Table of Contents

Namespace EzrSquared

Classes

Position

The representation of a position in the script.

Token

The smallest component in the script identified by the TokenType, grouped together into Node objects to from source code constructs.

Interfaces

IEzrError

Error interface for all errors.

Enums

AccessMod

Accessibility modifiers for all Context defined scope.

TokenType

The identifying type of a Token.

TokenTypeGroup

The type group of a Token.

Enum AccessMod

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

Accessibility modifiers for all Context defined scope.

```
[Flags]
public enum AccessMod
```

Fields

`Constant = 8`

Stand-in for a constant reference.

`Global = 1`

Stand-in for a global scope.

`GlobalStatic = Global | Static`

Stand-in for a global static scope.

`InvalidGlobalPrivate = Global | Private`

Stand-in for an ***INVALID*** global private scope.

`LocalScope = 16`

Stand-in for a local-only scope.

`None = 0`

Default value.

`Private = 2`

Stand-in for a private scope.

`PrivateConstant = Private | Constant`

Stand-in for a private constant scope.

```
PrivateStaticConstant = Static | PrivateConstant
```

Stand-in for a private static constant scope.

```
Static = 4
```

Stand-in for a static scope.

Interface IEzrError

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

Error interface for all errors.

```
public interface IEzrError
```

Properties

Details

The reason why the IEzrError occurred.

```
string Details { get; }
```

Property Value

string

ErrorEndPosition

The ending Position of the IEzrError.

```
Position ErrorEndPosition { get; }
```

Property Value

Position

ErrorStartPosition

The starting Position of the IEzrError.

```
Position ErrorStartPosition { get; }
```

Property Value

Position

Title

The name of the IEzrError.

```
string Title { get; }
```

Property Value

string

Methods

SourceWithUnderline(Position, Position)

Creates formatted text which contains the text between `startPosition` and `endPosition`, underlined with tilde (~) symbols.

```
protected internal static (int AdjustedLineNumber, string SourceWithUnderline)
SourceWithUnderline(Position startPosition, Position endPosition)
```

Parameters

`startPosition` Position

The starting position of the underlining.

`endPosition` Position

The ending position of the underlining.

Returns

(int AdjustedLineNumber, string SourceWithUnderline)

The formatted text and the actual starting line number of the error.

Class Position

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

The representation of a position in the script.

```
public class Position
```

Inheritance

object ← Position

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Position(int, int, string, string)

The representation of a position in the script.

```
public Position(int index, int line, string file, string script)
```

Parameters

`index` int

The index of the Position in the script.

`line` int

The line number of the Position in the script.

`file` string

The file name/path of the script.

`script` string

The script as text.

Fields

File

The file name/path of the script.

```
public readonly string File
```

Field Value

string

Index

The index of the Position in the script.

```
public int Index
```

Field Value

int

Line

The line number of the Position in the script.

```
public int Line
```

Field Value

int

None

A position that does not exist.

```
public static readonly Position None
```

Field Value

Position

Script

The script as text.

```
public readonly string Script
```

Field Value

string

Methods

Advance()

Advances the Position and increments Index by 1.

```
public void Advance()
```

Advance(char)

Advances the Position and increments Index by 1. If `currentChar` is a new-line character, Line is also incremented by 1.

```
public void Advance(char currentChar)
```

Parameters

`currentChar` char

The character associated with the Position before advancing.

Copy()

Creates a copy of the Position object.

```
public Position Copy()
```

Returns

Position

The copy.

ReverseTo(int, int)

Reverses to the given index.

```
public void ReverseTo(int index, int lineDecrement)
```

Parameters

`index` int

The index to reverse to.

`lineDecrement` int

The decrement for Line.

Class Token

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

The smallest component in the script identified by the TokenType, grouped together into Node objects to from source code constructs.

```
public class Token
```

Inheritance

object ← Token

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

Token(TokenType, TokenTypeGroup, string, Position, Position?)

Creates a new Token object.

```
public Token(TokenType type, TokenTypeGroup typeGroup, string value, Position startPosition,
Position? endPosition = null)
```

Parameters

`type` TokenType

The identifying TokenType of the Token.

`typeGroup` TokenTypeGroup

The TokenTypeGroup of the Token.

`value` string

The value of the Token; may be empty.

`startPosition` Position

The starting Position of the Token in the script.

`endPosition` Position

The ending Position of the Token in the script. If not given, copies `startPosition` and advances it.

Fields

Empty

An empty token value.

```
public static readonly Token Empty
```

Field Value

Token

EndPosition

The ending Position of the Token in the script.

```
public readonly Position EndPosition
```

Field Value

Position

StartPosition

The starting Position of the Token in the script.

```
public readonly Position StartPosition
```

Field Value

Position

Type

The identifying TokenType of the Token.

```
public readonly TokenType Type
```

Field Value

TokenType

TypeGroup

The TokenTypeGroup of the Token.

```
public readonly TokenTypeGroup TypeGroup
```

Field Value

TokenTypeGroup

Value

The value of the Token; may be empty.

```
public readonly string Value
```

Field Value

string

Methods

ToString()

Converts the Token into a string, for debugging purposes.

```
public override string ToString()
```

Returns

string

The string representation of the Token.

Enum TokenType

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

The identifying type of a Token.

```
public enum TokenType : ushort
```

Fields

`Ampersand = 66`

The '&' symbol, part of the Symbol type-group.

`AssignmentAddition = 87`

The ':+' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentBitwiseAnd = 93`

The ':&' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentBitwiseLeftShift = 96`

The ':<<' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentBitwiseOr = 94`

The ':|' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentBitwiseRightShift = 97`

The ':>>' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentBitwiseXOr = 95`

The ':' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentDivision = 90`

The ':/' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

`AssignmentModulo = 91`

The ':%' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

```
AssignmentMultiplication = 89
```

The ':*' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

```
AssignmentPower = 92
```

The ':^' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

```
AssignmentSubtraction = 88
```

The ':-' symbols, used in assignment operations, part of the AssignmentSymbol type-group.

```
Asterisk = 62
```

The '*' symbol, part of the Symbol type-group.

```
Backslash = 68
```

The '' symbol, part of the Symbol type-group.

```
BitwiseLeftShift = 82
```

The '<<' symbols, used in the bitwise-left-shift operation, part of the Symbol type-group.

```
BitwiseRightShift = 83
```

The '>>' symbols, used in the bitwise-right-shift operation, part of the Symbol type-group.

```
Caret = 65
```

The '^' symbol, part of the Symbol type-group.

```
Character = 3
```

A character, part of the Value type-group.

```
CharacterList = 4
```

A character list, which is a mutable string, part of the Value type-group.

```
Colon = 86
```

The ':' symbol, used in assignment operations, part of the AssignmentSymbol type-group.

```
Comma = 80
```

The ',' symbol, part of the Symbol type-group.

`EndOfFile = 100`

Represents the end of a script, part of the Special type-group.

`EqualSign = 76`

The '=' symbol, part of the Symbol type-group.

`ExclamationMark = 77`

The '!' symbol, part of the Symbol type-group.

`FloatingPoint = 1`

A floating-point number, part of the Value type-group.

`GreaterThanOrEqual = 85`

The '>=' symbols, used in comparison operations, part of the Symbol type-group.

`GreaterThanSign = 79`

The '>' symbol, part of the Symbol type-group.

`HyphenMinus = 61`

The '-' symbol, part of the Symbol type-group.

`Identifier = 98`

Represents an identifier, a name that is assigned by the programmer for an element such as variable, class, function, etc.
Part of the Special type-group.

`Integer = 0`

An integer, part of the Value type-group.

`Invalid = 101`

Represents an invalid token, part of the Special type-group.

`KeywordAll = 42`

The keyword "all", case-insensitive, part of the Keyword type-group.

`KeywordAnd = 5`

The keyword "and", case-insensitive, part of the Keyword type-group.

`KeywordAs = 23`

The keyword "as", case-insensitive, part of the Keyword type-group.

`KeywordCatch = 36`

The keyword "catch", case-insensitive, part of the Keyword type-group.

`KeywordConstant = 12`

The keyword "constant", case-insensitive, part of the Keyword type-group.

`KeywordCount = 17`

The keyword "count", case-insensitive, part of the Keyword type-group.

`KeywordDefine = 37`

The keyword "define", case-insensitive, part of the Keyword type-group.

`KeywordDo = 39`

The keyword "do", case-insensitive, part of the Keyword type-group.

`KeywordEach = 19`

The keyword "each", case-insensitive, part of the Keyword type-group.

`KeywordElse = 16`

The keyword "else", case-insensitive, part of the Keyword type-group.

`KeywordEnd = 40`

The keyword "end", case-insensitive, part of the Keyword type-group.

`KeywordFor = 18`

The keyword "for", case-insensitive, part of the Keyword type-group.

`KeywordFrom = 20`

The keyword "from", case-insensitive, part of the Keyword type-group.

`KeywordFunction = 29`

The keyword "function", case-insensitive, part of the Keyword type-group.

`KeywordGlobal = 10`

The keyword "global", case-insensitive, part of the Keyword type-group.

`KeywordIf = 15`

The keyword "if", case-insensitive, part of the Keyword type-group.

`KeywordIn = 9`

The keyword "in", case-insensitive, part of the Keyword type-group.

`KeywordInclude = 41`

The keyword "include", case-insensitive, part of the Keyword type-group.

`KeywordInvert = 7`

The keyword "invert", case-insensitive, part of the Keyword type-group.

`KeywordItem = 14`

The keyword "item", case-insensitive, part of the Keyword type-group.

`KeywordLast = 34`

The keyword "last", case-insensitive, part of the Keyword type-group.

`KeywordMore = 27`

The keyword "more", case-insensitive, part of the Keyword type-group.

`KeywordNamed = 28`

The keyword "named", case-insensitive, part of the Keyword type-group.

`KeywordNot = 8`

The keyword "not", case-insensitive, part of the Keyword type-group.

`KeywordObject = 31`

The keyword "object", case-insensitive, part of the Keyword type-group.

```
KeywordOr = 6
```

The keyword "or", case-insensitive, part of the Keyword type-group.

```
KeywordPrivate = 11
```

The keyword "private", case-insensitive, part of the Keyword type-group.

```
KeywordReadonly = 13
```

The keyword "readonly", case-insensitive, part of the Keyword type-group.

```
KeywordReturn = 33
```

The keyword "return", case-insensitive, part of the Keyword type-group.

```
KeywordSkip = 25
```

The keyword "skip", case-insensitive, part of the Keyword type-group.

```
[Obsolete("The \"special\" keyword, used in the \"special functions\" structure which was used for operator overloading, has become obsolete. It will likely be removed in future versions of ezr² due to the structure being replaced with named functions.")] KeywordSpecial = 30
```

The keyword "special", case-insensitive, part of the Keyword type-group.

```
KeywordStatic = 38
```

The keyword "static", case-insensitive, part of the Keyword type-group.

```
KeywordStep = 22
```

The keyword "step", case-insensitive, part of the Keyword type-group.

```
KeywordStop = 26
```

The keyword "stop", case-insensitive, part of the Keyword type-group.

```
KeywordTo = 21
```

The keyword "to", case-insensitive, part of the Keyword type-group.

```
KeywordTry = 35
```

The keyword "try", case-insensitive, part of the Keyword type-group.

`KeywordWhile = 24`

The keyword "while", case-insensitive, part of the Keyword type-group.

`KeywordWith = 32`

The keyword "with", case-insensitive, part of the Keyword type-group.

`LeftCurlyBracket = 74`

The '{' symbol, part of the Symbol type-group.

`LeftParenthesis = 70`

The '(' symbol, part of the Symbol type-group.

`LeftSquareBracket = 72`

The '[' symbol, part of the Symbol type-group.

`LessThanOrEqual = 84`

The '<=' symbols, used in comparison operations, part of the Symbol type-group.

`LessThanSign = 78`

The '<' symbol, part of the Symbol type-group.

`NewLine = 99`

Represents a new line, part of the Special type-group.

`PercentSign = 64`

The '%' symbol, part of the Symbol type-group.

`Period = 81`

The '.' symbol, part of the Symbol type-group.

`Plus = 60`

The '+' symbol, part of the Symbol type-group.

`QeywordC = 46`

The QuickSyntax keyword "c", case-insensitive, part of the Qeyword type-group.

```
QeywordD = 56
```

The QuickSyntax keyword "d", case-insensitive, part of the Qeyword type-group.

```
QeywordE = 45
```

The QuickSyntax keyword "e", case-insensitive, part of the Qeyword type-group.

```
QeywordF = 43
```

The QuickSyntax keyword "f", case-insensitive, part of the Qeyword type-group.

```
QeywordFd = 50
```

The QuickSyntax keyword "fd", case-insensitive, part of the Qeyword type-group.

```
QeywordG = 57
```

The QuickSyntax keyword "g", case-insensitive, part of the Qeyword type-group.

```
QeywordI = 54
```

The QuickSyntax keyword "i", case-insensitive, part of the Qeyword type-group.

```
QeywordL = 44
```

The QuickSyntax keyword "l", case-insensitive, part of the Qeyword type-group.

```
QeywordN = 48
```

The QuickSyntax keyword "n", case-insensitive, part of the Qeyword type-group.

```
QeywordOd = 53
```

The QuickSyntax keyword "od", case-insensitive, part of the Qeyword type-group.

```
QeywordP = 58
```

The QuickSyntax keyword "p", case-insensitive, part of the Qeyword type-group.

```
QeywordS = 55
```

The QuickSyntax keyword "s", case-insensitive, part of the Qeyword type-group.

```
QeywordSb = 52
```

The QuickSyntax keyword "sb", case-insensitive, part of the Qeyword type-group.

`QeywordSd = 51`

The QuickSyntax keyword "sd", case-insensitive, part of the Qeyword type-group.

`QeywordT = 47`

The QuickSyntax keyword "t", case-insensitive, part of the Qeyword type-group.

`QeywordV = 59`

The QuickSyntax keyword "v", case-insensitive, part of the Qeyword type-group.

`QeywordW = 49`

The QuickSyntax keyword "w", case-insensitive, part of the Qeyword type-group.

`RightCurlyBracket = 75`

The '}' symbol, part of the Symbol type-group.

`RightParenthesis = 71`

The ')' symbol, part of the Symbol type-group.

`RightSquareBracket = 73`

The ']' symbol, part of the Symbol type-group.

`Slash = 63`

The '/' symbol, part of the Symbol type-group.

`String = 2`

A text-string, string, or a sequence of characters, part of the Value type-group.

`Tilde = 69`

The '~' symbol, part of the Symbol type-group.

`VerticalBar = 67`

The '|' symbol, part of the Symbol type-group.

Enum TokenTypeGroup

Namespace: EzrSquared

Assembly: ezrSquared-lib.dll

The type group of a Token.

```
public enum TokenTypeGroup : ushort
```

Fields

`AssignmentSymbol = 3`

Groups variable assignment symbol tokens.

`Keyword = 1`

Groups keyword tokens.

`Qeyword = 2`

Groups QuickSyntax keyword tokens.

`Special = 5`

Groups special tokens.

`Symbol = 4`

Groups general symbol tokens.

`Value = 0`

Groups primitive-type tokens, like integers or strings.

Remarks

If you want to see which group a token type is in, check the documentation for that type.

Namespace EzrSquared.Executor

Classes

CodeExecutor

Utility to execute ezr² code under one static interpreter.

ExecutionResult

The result of a code execution operation.

Class CodeExecutor

Namespace: EzrSquared.Executor

Assembly: ezrSquared-lib.dll

Utility to execute ezr^2 code under one static interpreter.

```
public static class CodeExecutor
```

Inheritance

object ← CodeExecutor

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

Interpreter

The static interpreter.

```
public static readonly Interpreter Interpreter
```

Field Value

Interpreter

Properties

RuntimeContext

The runtime context, may be null.

```
public static Context? RuntimeContext { get; private set; }
```

Property Value

Context

RuntimeResult

The static RuntimeResult of Interpreter.

```
public static RuntimeResult RuntimeResult { get; }
```

Property Value

RuntimeResult

Methods

AddToContext(Reference, string)

Adds the given reference to the runtime context.

```
public static void AddToContext(Reference reference, string name)
```

Parameters

`reference` Reference

The reference to add.

`name` string

The name of the new symbol.

Exceptions

NullReferenceException

Thrown if the runtime context is null. Use CreateRuntimeContext(string) to initialize the runtime context.

AddToContext<TMemberInfo>(EzrSharpCompatibilityWrapper<TMemberInfo>, AccessMod)

Adds the given wrapped C# object to the runtime context.

```
public static void AddToContext<TMemberInfo>(EzrSharpCompatibilityWrapper<TMemberInfo>
wrapper, AccessMod accessibilityModifiers = AccessMod.Constant) where TMemberInfo
: MemberInfo
```

Parameters

`wrapper` EzrSharpCompatibilityWrapper<TMemberInfo>

The wrapped object to add.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the reference. Defaults to Constant.

Type Parameters

`TMemberInfo`

See EzrSharpCompatibilityWrapper<TMemberInfo>.

Exceptions

NullReferenceException

Thrown if the runtime context is null. Use CreateRuntimeContext(string) to initialize the runtime context.

CreateRuntimeContext(string)

Creates the runtime context.

```
public static void CreateRuntimeContext(string filePath)
```

Parameters

`filePath` string

The file path to the code file this context will be used to execute.

Execute(string)

Executes the given script.

```
public static ExecutionResult Execute(string script)
```

Parameters

`script` string

The script to execute.

Returns

ExecutionResult

Exceptions

NullReferenceException

Thrown if the runtime context is null. Use CreateRuntimeContext(string) to initialize the runtime context.

PopulateRuntimeContext(bool)

Populates the runtime context with all built-ins.

```
public static void PopulateRuntimeContext(bool excludeIO = false)
```

Parameters

`excludeIO` bool

Exclude built-in I/O functions like Show(SharpMethodParameters)?

Exceptions

NullReferenceException

Thrown if the runtime context is null. Use CreateRuntimeContext(string) to initialize the runtime context.

Class ExecutionResult

Namespace: EzrSquared.Executor

Assembly: ezrSquared-lib.dll

The result of a code execution operation.

```
public class ExecutionResult
```

Inheritance

object ← ExecutionResult

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

ExecutionResult(Token[], Node, RuntimeResult)

Creates a new execution result.

```
public ExecutionResult(Token[] tokens, Node ast, RuntimeResult result)
```

Parameters

`tokens` Token[]

The tokens returned by the lexer.

`ast` Node

The executed AST node.

`result` RuntimeResult

The runtime result carrying the result or error.

ExecutionResult(Token[], EzrSyntaxError)

Creates a new execution result, which failed at lexing.

```
public ExecutionResult(Token[] tokens, EzrSyntaxError error)
```

Parameters

`tokens` Token[]

The tokens returned by the lexer.

`error` EzrSyntaxError

The error.

ExecutionResult(Token[], ParseResult)

Creates a new execution result, which failed at parsing.

```
public ExecutionResult(Token[] tokens, ParseResult result)
```

Parameters

`tokens` Token[]

The tokens returned by the lexer.

`result` ParseResult

The parse result carrying the error.

Fields

Ast

The Abstract Syntax Tree of the executed script.

```
public readonly Node? Ast
```

Field Value

Node

LexerError

Any error that occurred during lexing.

```
public readonly EzrSyntaxError? LexerError
```

Field Value

EzrSyntaxError

ParseError

Any error that occurred during parsing.

```
public readonly EzrSyntaxError? ParseError
```

Field Value

EzrSyntaxError

Result

The result of the execution.

```
public readonly IEzrObject? Result
```

Field Value

IEzrObject

Success

Was the execution successful?

```
public readonly bool Success
```

Field Value

bool

Tokens

The tokens making up the executed script.

```
public readonly Token[] Tokens
```

Field Value

Token[]

Methods

GetErrorMessage()

Returns the error message of the execution, be it a lexing, parsing or interpretation error.

```
public string GetErrorMessage()
```

Returns

string

The error message.

Namespace EzrSquared.Runtime

Classes

Context

Stores all user defined variables/constant references, called symbols.

Interpreter

The ezr² Interpreter. The job of the Interpreter is to execute the Abstract Syntax Tree (AST) received from the Parser (as a single Node object)!

Reference

A reference to an IEzrObject, with its scope, usable name and more metadata.

ReferencePool

A static pool for Reference.

RuntimeResult

The type of the object that is returned as the result of interpretation done by the Interpreter.

Interfaces

IMutable<T>

A mutable object.

Enums

Context.GetStatus

Represents the status of a Get(Context?, string, out Reference, AccessMod, bool) call.

Context.SetStatus

Represents the status of a set call.

Class Context

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

Stores all user defined variables/constant references, called symbols.

```
public class Context : IEnumerable<KeyValuePair<string, Reference>>, IEnumerable
```

Inheritance

object ← Context

Implements

IEnumerable<KeyValuePair<string, Reference>>, IEnumerable

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Context(string, bool, Position, Context?, Context?, int)

Creates a new Context.

```
public Context(string name, bool isStatic, Position startPosition, Context? parent = null,
Context? staticContext = null, int linkedContexts = 0)
```

Parameters

`name` string

The name of the Context.

`isStatic` bool

Is this a static Context?

`startPosition` Position

The starting Position of the Context.

`parent` Context

The parent of this Context. Can be null.

`staticContext` Context

The static Context in relation to this Context. Can be null.

`linkedContexts` int

The number of other Contexts which are linked to this. See LinkedContexts.

Exceptions

ArgumentException

Raised if `isStatic` is true and a static Context as `staticContext` was provided at the same time. This is invalid as a Context can either be static or have a static Context related to it, but not both.

Fields

Empty

An empty context.

```
public static readonly Context Empty
```

Field Value

Context

Id

The unique identifier of the Context.

```
public readonly long Id
```

Field Value

long

IsStatic

Is this a static Context?

```
public readonly bool IsStatic
```

Field Value

bool

Name

The name of the Context.

```
public readonly string Name
```

Field Value

string

_symbols

The symbols of this Context.

```
private Dictionary<string, Reference> _symbols
```

Field Value

Dictionary<string, Reference>

Properties

Count

The number of symbols in the context.

```
public int Count { get; }
```

Property Value

int

LinkedContexts

Other Contexts which are linked to this.

```
public Context[] LinkedContexts { get; private set; }
```

Property Value

Context[]

Remarks

This allowed the Interpreter to access variables from multiple Contexts while in a local scope.

Parent

The parent of this Context. May be null.

```
public Context? Parent { get; private set; }
```

Property Value

Context

StartPosition

The starting Position of the Context.

```
public Position StartPosition { get; private set; }
```

Property Value

Position

StaticContext

The static Context in relation to this Context. May be null.

```
public Context? StaticContext { get; private set; }
```

Property Value

Context

Methods

DeepCopy(RuntimeResult, IEzrObject[])

Creates a deep copy of the context.

```
public Context? DeepCopy(RuntimeResult result, IEzrObject[] excludedSymbols)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors/

`excludedSymbols` IEzrObject[]

Symbols to exclude when copying.

Returns

Context

The copy, or, null if failed.

Remarks

It is the caller's responsibility to handle linked contexts.

~Context()

Destructor.

```
protected ~Context()
```

Get(Context?, string, out Reference, AccessMod, bool)

Retrieves the Reference of the requested symbol.

```
public Context.GetStatus Get(Context? callingContext, string symbol, out Reference
objectReference, AccessMod accessibilityModifiers = AccessMod.None, bool
ignoreLinkedContexts = false)
```

Parameters

`callingContext` Context

The Context from which the operation is being called.

`symbol` string

The symbol to retrieve.

`objectReference` Reference

The resulting Reference.

`accessibilityModifiers` AccessMod

Accessibility modifiers of the operation, None by default.

`ignoreLinkedContexts` bool

Should the Context ignore its linked contexts? (Used mainly for retrieving a reference before symbol assignment)

Returns

Context.GetStatus

The Context.GetStatus, representing the result of the operation.

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<KeyValuePair<string, Reference>> GetEnumerator()
```

Returns

IEnumerator<KeyValuePair<string, Reference>>

An enumerator that can be used to iterate through the collection.

IsContextParent(Context)

Checks if the given Context is a parent of this Context.

```
private bool IsContextParent(Context context)
```

Parameters

`context` Context

The Context to check.

Returns

bool

true if it is, false otherwise.

IsDefined(string)

Checks if a symbol has been defined in this Context/

```
public bool IsDefined(string name)
```

Parameters

`name` string

The name of the symbol to check.

Returns

bool

true if it is, false otherwise.

Release()

Releases all references in the context and clears the symbols dictionary.

```
public void Release()
```

Set(Context?, string, Reference)

Sets the Reference to a new symbol.

```
public Context.SetStatus Set(Context? callingContext, string symbol,
Reference objectReference)
```

Parameters

`callingContext` Context

The Context from which the operation is being called.

`symbol` string

The symbol to assign.

`objectReference` Reference

The reference to assign to the symbol.

Returns

Context.SetStatus

The Context.SetStatus, representing the result of the operation.

Set(Context?, (IEzrObject Object, string Name), Reference, AccessMod)

Sets a new IEzrObject to an existing Reference.

```
public (Context.SetStatus, Reference) Set(Context? callingContext, (IEzrObject Object,
string Name) newObject, Reference oldReference, AccessMod accessibilityModifiers
= AccessMod.None)
```

Parameters

`callingContext` Context

The Context from which the operation is being called.

`newObject` (IEzrObject Object, string Name)

The new IEzrObject to be assigned and the name of the symbol, in the format (IEzrObject Object, string Name).

`oldReference` Reference

The old reference to assign to.

`accessibilityModifiers` AccessMod

Accessibility modifiers of the operation, None by default.

Returns

(Context.SetStatus, Reference)

The Context.SetStatus, representing the result of the operation, and the reference to the set symbol.

Remarks

If `oldReference` is Empty, the operation is passed onto Set(Context?, string, Reference).

SetNewLinkedContexts(int)

Clears the existing LinkedContexts and creates a new empty Array of Contexts.

```
public void SetNewLinkedContexts(int contexts)
```

Parameters

`contexts` int

The number of Contexts to allocate the Array for.

UpdateParent(Context)

Updates the parent of this Context/

```
public void UpdateParent(Context newParent)
```

Parameters

`newParent` Context

The new parent Context.

UpdatePosition(Position)

Updates the starting Position of this Context/

```
public void UpdatePosition(Position startPosition)
```

Parameters

`startPosition` Position

The new starting Position.

UpdateStaticContext(Context?)

Updates the StaticContext to a new one.

```
public void UpdateStaticContext(Context? newStaticContext)
```

Parameters

`newStaticContext` Context

The new static Context.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Enum Context.GetStatus

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

Represents the status of a Get(Context?, string, out Reference, AccessMod, bool) call.

```
public enum Context.GetStatus
```

Fields

`AccessToPrivateSymbolNotAllowed = 2`

Accessing privates symbols from Contexts without permission is not allowed.

`InvalidGlobalPrivateAccessNotAllowed = 4`

Cannot access symbols with both Global and Private accessibility modifiers.

`Ok = 0`

The operation was successful.

`StaticAccessWithoutDefinedContextNotAllowed = 3`

Cannot access static symbols without a known static Context.

`UndefinedSymbolAccessNotAllowed = 1`

The requested symbol was not found.

Enum Context.SetStatus

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

Represents the status of a set call.

```
public enum Context.SetStatus
```

Fields

`ConstantAssignmentNotAllowed = 1`

Assigning new values to a constant reference is not allowed.

`InvalidGlobalPrivateAssignmentNotAllowed = 7`

Cannot assign symbols with both Global and Private accessibility modifiers.

`Ok = 0`

The operation was successful.

`PrivateSymbolAssignmentInChildNotAllowed = 4`

Assigning a private symbol to a child Context is not allowed.

`PrivateSymbolAssignmentNotAllowed = 2`

Assigning privates symbols from Contexts without permission is not allowed.

`StaticAssignmentWithoutDefinedContextNotAllowed = 6`

Assigning static symbols without a known static Context is not allowed.

`SymbolAccessibilityChangeInParentNotAllowed = 3`

Changing the accessibility of an existing symbol in a parent Context is not allowed.

`UnregisteredSymbolScopeOrVariabilityChangeNotAllowed = 5`

Changing the accessibility of a symbol which has not been assigned to a Context is not allowed.

Remarks

This includes both Set(Context?, string, Reference) and Set(Context?, (IEzrObject Object, string Name), Reference, AccessMod).

Interface IMutable<T>

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

A mutable object.

```
public interface IMutable<T>
```

Type Parameters

`T`

The type inheriting this interface.

Methods

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
IMutable<T>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<T>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Class Interpreter

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

The ezr² Interpreter. The job of the Interpreter is to execute the Abstract Syntax Tree (AST) received from the Parser (as a single Node object)!

```
public class Interpreter
```

Inheritance

object ← Interpreter

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

RuntimeResult

The result of the current execution.

```
public readonly RuntimeResult RuntimeResult
```

Field Value

RuntimeResult

Methods

AssignValueToVariable(Reference, IEzrObject, VariableAssignmentNode, Context, Context, AccessMod)

Assigns one of the values to one of the variables defined in the VariableAssignmentNode.

```
private void AssignValueToVariable(Reference variable, IEzrObject value,
VariableAssignmentNode node, Context executionContext, Context callingContext,
AccessMod accessibilityModifiers)
```

Parameters

`variable` Reference

The variable reference to assign to.

`value` IEzrObject

The value object to be assigned.

`node` VariableAssignmentNode

The VariableAssignmentNode being executed.

`executionContext` Context

The Context in which the variable will be assigned.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

AssignValuesToVariables(VariableAssignmentNode, Reference[], bool, Context, Context, AccessMod)

Assigns the value(s) to the variable(s) defined in the VariableAssignmentNode.

```
private void AssignValuesToVariables(VariableAssignmentNode node, Reference[]
variables, bool isSingleVariable, Context executionContext, Context callingContext,
AccessMod accessibilityModifiers)
```

Parameters

`node` VariableAssignmentNode

The VariableAssignmentNode being executed.

`variables` Reference[]

The array of variable references to be assigned to.

`isSingleVariable` bool

Is only one variable being assigned?

`executionContext` Context

The Context in which the variable will be assigned.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Execute(Node, Context, AccessMod)

Executes an AST under the given Context and returns the result.

```
public RuntimeResult Execute(Node ast, Context runtimeContext, AccessMod
accessibilityModifiers = AccessMod.None)
```

Parameters

`ast` Node

The AST, as a single Node object.

`runtimeContext` Context

The run-time Context in which the AST will be executed.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Returns

RuntimeResult

The RuntimeResult object.

ExecuteBinaryOperation(IEzrObject, IEzrObject, Node, TokenType, Context)

Executes a binary operation.

```
private void ExecuteBinaryOperation(IEzrObject first, IEzrObject second, Node node,
TokenType operation, Context executionContext)
```

Parameters

`first` IEzrObject

The first value in the operation.

`second` IEzrObject

The second value in the operation.

`node` Node

The operation's AST node.

`operation` TokenType

The operation to perform.

`executionContext` Context

The context under which the operation will take place.

Exceptions

NotImplementedException

Thrown if a case for the TokenType of the operand was not defined.

ExecuteConjunctiveOperators(BinaryOperationNode, Context, Context, AccessMod)

Executes a boolean conjunctive operation.

```
private void ExecuteConjunctiveOperators(BinaryOperationNode node, Context executionContext,
Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` BinaryOperationNode

The operation's AST node.

`executionContext` Context

The context under which the operation will take place.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

ExecuteObjectAttributeAccess(BinaryOperationNode, Context, Context, AccessMod, bool)

Executes an object attribute access operation.

```
private void ExecuteObjectAttributeAccess(BinaryOperationNode node, Context
executionContext, Context callingContext, AccessMod accessibilityModifiers,
bool ignoreUndefinedVariable)
```

Parameters

`node` BinaryOperationNode

The operation's AST node.

`executionContext` Context

The context under which the operation will take place.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

`ignoreUndefinedVariable` bool

Should the interpreter ignore undefined variables?

GetRegisteredReference(Node, Context, Context, AccessMod)

Gets and checks if a node represents a variable reference.

```
private Reference? GetRegisteredReference(Node node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` Node

The node to check.

`executionContext` Context

The Context in which the variable will be assigned.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Returns

Reference

The variable reference. null if any error occurs.

HandleSetStatus(SetStatus, string, Node, Context)

Throws errors for Set(Context?, string, Reference) returns.

```
private void HandleSetStatus(Context.SetStatus status, string referenceName, Node
referenceNode, Context referenceContext)
```

Parameters

`status` Context.SetStatus

The set status.

`referenceName` string

The name of the set reference.

`referenceNode` Node

The node of the reference object.

`referenceContext` Context

The context of the reference.

Exceptions

NotImplementedException

Thrown if an unknown set status is encountered.

HandleSetStatus((SetStatus Status, Reference Reference), string, Node, Context)

Throws errors for Set(Context?, (IEzrObject Object, string Name), Reference, AccessMod) returns.

```
private Reference HandleSetStatus((Context.SetStatus Status, Reference Reference) setResult,
string referenceName, Node referenceNode, Context referenceContext)
```

Parameters

`setResult` (Context.SetStatus, Reference)

The set result.

`referenceName` string

The name of the set reference.

`referenceNode` Node

The node of the reference object.

`referenceContext` Context

The context of the reference.

Returns

Reference

The set reference.

IterateLoop(BigInteger?, double?, List<IEzrObject>, CountNode, Context, Context, AccessMod, AccessMod, Reference?, string, Context)

Code to run in an iteration of a count expression.

```
private int IterateLoop(BigInteger? iBigInteger, double? iDouble, List<IEzrObject> returns,
CountNode node, Context callingContext, Context executionContext, AccessMod
accessibilityModifiers, AccessMod operationAccessibilityModifiers, Reference?
iterationVariableReference, string iterationVariableName, Context
iterationVariableRegisteredContext)
```

Parameters

`iBigInteger` BigInteger?

The current iteration of the loop as a BigInteger.

`iDouble` double?

The current iteration of the loop as a double.

`returns` List<IEzrObject>

The list of the returns of the loop.

`node` CountNode

The loop node.

`callingContext` Context

The Context calling on the execution of the iteration.

`executionContext` Context

The Context under which the iteration will be executed.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing iteration.

`operationAccessibilityModifiers` AccessMod

The accessibility modifiers for the iteration variable.

`iterationVariableReference` Reference

The iteration variable reference.

`iterationVariableName` string

The name of the iteration variable

`iterationVariableRegisteredContext` Context

The context under which the iteration variable is registered.

Returns

int

-1 for errors, 1 to skip the iteration, 2 for stopping the loop or 0 for continuation of the loop.

VisitArrayLikeNodeArray(ArrayLikeNode, Context, Context, AccessMod)

Creates an array of objects.

```
private void VisitArrayLikeNodeArray(ArrayLikeNode node, Context executionContext, Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` ArrayLikeNode

The ArrayLikeNode to execute.

`executionContext` Context

The Context under which the array will be created.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitArrayLikeNodeList(ArrayLikeNode, Context, Context, AccessMod)

Creates a list of references to its elements.

```
private void VisitArrayLikeNodeList(ArrayLikeNode node, Context executionContext, Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` ArrayLikeNode

The ArrayLikeNode to execute.

`executionContext` Context

The Context under which the list will be created.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitBinaryOperationNode(BinaryOperationNode, Context, Context, AccessMod, bool)

Executes a BinaryOperationNode and performs a binary operation.

```
private void VisitBinaryOperationNode(BinaryOperationNode node, Context executionContext,
Context callingContext, AccessMod accessibilityModifiers, bool ignoreUndefinedVariable)
```

Parameters

`node` BinaryOperationNode

The BinaryOperationNode to execute.

`executionContext` Context

The Context under which the operation will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

`ignoreUndefinedVariable` bool

Should the interpreter ignore undefined variables?

VisitCallNode(CallNode, Context, Context, AccessMod)

Executes a CallNode and calls the Execute(Reference[], Interpreter, RuntimeResult) function in Receiver.

```
private void VisitCallNode(CallNode node, Context executionContext, Context callingContext,
AccessMod accessibilityModifiers)
```

Parameters

`node` CallNode

The CallNode to execute.

`executionContext` Context

The Context under which the call takes place.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitClassDefinitionNode(ClassDefinitionNode, Context, Context, AccessMod)

Executes a ClassDefinitionNode and creates a class.

```
private void VisitClassDefinitionNode(ClassDefinitionNode node, Context executionContext,
Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` ClassDefinitionNode

The ClassDefinitionNode to execute.

`executionContext` Context

The Context in which the class will be created.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitCountNode(CountNode, Context, Context, AccessMod)

Executes a CountNode and creates a counting loop.

```
private void VisitCountNode(CountNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` CountNode

The CountNode to execute.

`executionContext` Context

The Context under which the loop will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitDefineBlockNode(DefineBlockNode, Context, Context, AccessMod)

Executes a DefineBlockNode under its accessibility modifiers in the given Context.

```
private void VisitDefineBlockNode(DefineBlockNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` DefineBlockNode

The DefineBlockNode to execute.

`executionContext` Context

The Context in which the block will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitDictionaryNode(DictionaryNode, Context, Context, AccessMod)

Creates a EzrDictionary object from a DictionaryNode.

```
private void VisitDictionaryNode(DictionaryNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` DictionaryNode

The DictionaryNode to execute.

`executionContext` Context

The Context under which the dictionary will be created.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitForEachNode(ForEachNode, Context, Context, AccessMod)

Executes a ForEachNode and creates a for-each loop.

```
private void VisitForEachNode(ForEachNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` ForEachNode

The ForEachNode to execute.

`executionContext` Context

The Context under which the loop will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitFunctionDefinitionNode(FunctionDefinitionNode, Context, Context, AccessMod)

Executes a FunctionDefinitionNode and creates a function.

```
private void VisitFunctionDefinitionNode(FunctionDefinitionNode node, Context
executionContext, Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` FunctionDefinitionNode

The FunctionDefinitionNode to execute.

`executionContext` Context

The Context in which the function will be created.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitIfNode(IfNode, Context, Context, AccessMod)

Executes an IfNode and performs a boolean expression.

```
private void VisitIfNode(IfNode node, Context executionContext, Context callingContext,
AccessMod accessibilityModifiers)
```

Parameters

`node` IfNode

The IfNode to execute.

`executionContext` Context

The Context under which the expression will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitNoValueNode(NoValueNode, Context)

Executes a NoValueNode.

```
private void VisitNoValueNode(NoValueNode node, Context executionContext)
```

Parameters

`node` NoValueNode

The NoValueNode to execute.

`executionContext` Context

The Context under which the node is executed.

Remarks

This is (currently) only used for executing loop skip (calling SetSkipFlag(Reference)) and loop stop (calling SetStopFlag(Reference)) expressions.

Exceptions

NotImplementedException

Thrown when an unimplemented or unexpected ValueType is encountered.

VisitNode(Node, Context, Context?, AccessMod, bool)

Vists a Node and sends it to the appropriate execution function.

```
internal void VisitNode(Node astNode, Context executionContext, Context? callingContext,
AccessMod accessibilityModifiers, bool ignoreUndefinedVariable = false)
```

Parameters

`astNode` Node

The Node to execute.

`executionContext` Context

The Context under which the execution will take place.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

`ignoreUndefinedVariable` bool

Should the interpreter ignore undefined variables?

Exceptions

ArgumentException

Raised if `astNode` is of type InvalidNode. This should ***never*** happen as this means something has gone ***very wrong*** in the Parser.

NotImplementedException

Raised if the method for executing `astNode` has not been implemented.

VisitReturnNode(ReturnNode, Context, Context, AccessMod)

Executes a ReturnNode and sets the return flag in RuntimeResult.

```
private void VisitReturnNode(ReturnNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` ReturnNode

The ReturnNode to execute.

`executionContext` Context

The Context from which the arguments of the ReturnNode will be taken, if any.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitTryNode(TryNode, Context, Context, AccessMod)

Executes a TryNode and creates a try-catch block.

```
private void VisitTryNode(TryNode node, Context executionContext, Context callingContext,
AccessMod accessibilityModifiers)
```

Parameters

`node` TryNode

The TryNode to execute.

`executionContext` Context

The Context under which the block will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

VisitUnaryOperationNode(UnaryOperationNode, Context, Context, AccessMod)

Executes a UnaryOperationNode and performs a unary operation.

```
private void VisitUnaryOperationNode(UnaryOperationNode node, Context executionContext,
Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` UnaryOperationNode

The UnaryOperationNode to execute.

`executionContext` Context

The Context under which the operation will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Exceptions

NotImplementedException

Thrown if a case for the TokenType of the operand was not defined.

VisitValueNode(ValueNode, Context)

Creates a "value" type object - i.e. EzrInteger, EzrFloat, EzrString, EzrCharacter and EzrCharacterList, from a ValueNode.

```
private void VisitValueNode(ValueNode node, Context executionContext)
```

Parameters

`node` ValueNode

The ValueNode to execute.

`executionContext` Context

The Context under which the value will be created.

Exceptions

NotImplementedException

Raised if an unimplemented or unexpected TokenType is encountered in `node`.

VisitVariableAccessNode(VariableAccessNode, Context, Context, AccessMod, bool)

Executes a VariableAccessNode and access a variable from the given Context.

```
private void VisitVariableAccessNode(VariableAccessNode node, Context executionContext,
Context callingContext, AccessMod accessibilityModifiers, bool ignoreUndefinedVariable)
```

Parameters

`node` VariableAccessNode

The VariableAccessNode to execute.

`executionContext` Context

The Context from which the variable will be accessed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

`ignoreUndefinedVariable` bool

Should the interpreter ignore undefined variables?

Exceptions

NotImplementedException

Thrown if a case for the Context.GetStatus of the operation was not defined.

VisitVariableAssignmentNode(VariableAssignmentNode, Context, Context, AccessMod)

Executes a VariableAssignmentNode and sets a variable/constant in the given Context.

```
private void VisitVariableAssignmentNode(VariableAssignmentNode node, Context
executionContext, Context callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` VariableAssignmentNode

The VariableAssignmentNode to execute.

`executionContext` Context

The Context in which the variable will be assigned.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Remarks

A binary operation may also be executed depending on AssignmentOperator.
In the case a ArrayLikeNode is provided as Variable, multiple variables/constants will be assigned.

VisitWhileNode(WhileNode, Context, Context, AccessMod)

Executes a WhileNode and creates a while (condition = true) loop.

```
private void VisitWhileNode(WhileNode node, Context executionContext, Context
callingContext, AccessMod accessibilityModifiers)
```

Parameters

`node` WhileNode

The WhileNode to execute.

`executionContext` Context

The Context under which the loop will be executed.

`callingContext` Context

The Context calling on the execution of the Node.

`accessibilityModifiers` AccessMod

The accessibility modifiers for objects that will be assigned from the executing Node.

Class Reference

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

A reference to an IEzrObject, with its scope, usable name and more metadata.

```
public class Reference : IMutable<Reference>
```

Inheritance

object ← Reference

Implements

IMutable<Reference>

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Remarks

It is recommended to get references from the ReferencePool instead of creating them manually.

Constructors

Reference(IEzrObject?, AccessMod, string)

A reference to an IEzrObject, with its scope, usable name and more metadata.

```
public Reference(IEzrObject? @object = null, AccessMod accessibilityModifiers =
AccessMod.None, string name = "")
```

Parameters

`object` IEzrObject

The IEzrObject being referenced.

`accessibilityModifiers` AccessMod

The accessibility modifiers of the reference.

`name` string

The name of the reference.

Remarks

It is recommended to get references from the ReferencePool instead of creating them manually.

Fields

AccessibilityModifiers

The accessibility modifiers of the reference. See AccessMod.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

Empty

Static readonly empty reference.

```
public static readonly Reference Empty
```

Field Value

Reference

Properties

IsEmpty

Does this reference actually reference anything?

```
public bool IsEmpty { get; private set; }
```

Property Value

bool

IsRegistered

Is this reference registered anywhere, except RuntimeResults?

```
public bool IsRegistered { get; }
```

Property Value

bool

IsRegisteredIncludingResult

Is this reference registered anywhere, including RuntimeResults?

```
public bool IsRegisteredIncludingResult { get; }
```

Property Value

bool

Name

The name of the reference, may be Empty.

```
public string Name { get; private set; }
```

Property Value

string

Object

The IEzrObject this references.

```
public IEzrObject Object { get; private set; }
```

Property Value

IEzrObject

RegisteredContext

The Context in which the reference is defined in. May be null.

```
public Context? RegisteredContext { get; private set; }
```

Property Value

Context

RegisteredIn

The number of other objects that have this reference registered.

```
public int RegisteredIn { get; private set; }
```

Property Value

int

RegisteredInResult

Is this reference registered in a RuntimeResult?

```
public bool RegisteredInResult { get; private set; }
```

Property Value

bool

Methods

DeepCopy(RuntimeResult)

Creates a copy of the reference AND the referenced object, if it is mutable.

```
public IMutable<Reference>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result to return errors in.

Returns

IMutable<Reference>

The copy, or, null if failed.

Remarks

This copy only includes the original's Object, AccessibilityModifiers and Name.

Reset(IEzrObject?, AccessMod, string)

Resets the reference.

```
public Reference Reset(IEzrObject? @object, AccessMod accessibilityModifiers, string name)
```

Parameters

`object` IEzrObject

The IEzrObject being referenced.

`accessibilityModifiers` AccessMod

The accessibility modifiers of the reference.

`name` string

The name of the reference.

Returns

Reference

ShallowCopy()

Creates a very shallow copy of the reference.

```
public Reference ShallowCopy()
```

Returns

Reference

The copy.

Remarks

This copy only includes the original's Object and AccessibilityModifiers.

UpdateName(string)

Updates the reference with a new Name value.

```
public void UpdateName(string name)
```

Parameters

`name` string

The new name.

UpdateObject(IEzrObject, AccessMod)

Updates the reference with new Object and AccessibilityModifiers values.

```
public void UpdateObject(IEzrObject @object, AccessMod accessibilityModifiers
= AccessMod.None)
```

Parameters

`object` IEzrObject

The new object to reference.

`accessibilityModifiers` AccessMod

The new accessibility modifiers.

UpdateRegister(bool)

Updates the reference with a new RegisteredIn value.

```
public void UpdateRegister(bool isRegistered)
```

Parameters

`isRegistered` bool

Is it an unregister or register operation?

UpdateRegisteredContext(Context?, Context?)

Updates the reference with a new RegisteredContext value.

```
public void UpdateRegisteredContext(Context? context, Context? releasingContext = null)
```

Parameters

`context` Context

The new context.

`releasingContext` Context

The context that may be deleted after this is called. If the referenced object's creation context matches this, it is updated.

Remarks

This does not actually change the Context this reference is defined in, but it changes the *reference* to the Context this is defined in.

UpdateResultRegister(bool)

Updates the reference with a new RegisteredInResult value.

```
public void UpdateResultRegister(bool isRegistered)
```

Parameters

`isRegistered` bool

The new value.

Class ReferencePool

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

A static pool for Reference.

```
public static class ReferencePool
```

Inheritance

object ← ReferencePool

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

s_ezrObjectReferencePool

The pool of references.

```
private static readonly ConcurrentQueue<Reference> s_ezrObjectReferencePool
```

Field Value

ConcurrentQueue<Reference>

Methods

Get(IEzrObject?, AccessMod, string)

Gets a freshly-reset reference from the pool or creates a new one.

```
public static Reference Get(IEzrObject? @object = null, AccessMod accessibilityModifiers =
AccessMod.None, string name = "")
```

Parameters

`object` IEzrObject

The IEzrObject being referenced.

`accessibilityModifiers` AccessMod

The accessibility modifiers of the reference.

`name` string

The name of the reference.

Returns

Reference

The reference.

TryRelease(Reference)

Releases a reference to the pool.

```
public static void TryRelease(Reference reference)
```

Parameters

`reference` Reference

The reference to release.

Class RuntimeResult

Namespace: EzrSquared.Runtime

Assembly: ezrSquared-lib.dll

The type of the object that is returned as the result of interpretation done by the Interpreter.

```
public class RuntimeResult
```

Inheritance

object ← RuntimeResult

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

RuntimeResult()

Creates a new RuntimeResult.

```
internal RuntimeResult()
```

Fields

Error

The error that occured in interpretation, if none occured, this is null.

```
public IEzrRuntimeError? Error
```

Field Value

IEzrRuntimeError

Reference

The reference to the resulting IEzrObject.

```
public Reference Reference
```

Field Value

Reference

ReturnSet

The flag for when a return call is called.

```
public bool ReturnSet
```

Field Value

bool

SkipSet

The flag for when a loop skip is called.

```
public bool SkipSet
```

Field Value

bool

StopSet

The flag for when a loop stop is called.

```
public bool StopSet
```

Field Value

bool

Properties

ShouldReturn

Should the interpreter return from the current execution?

```
public bool ShouldReturn { get; }
```

Property Value

bool

Remarks

This is true when either of the below conditions are met:

- Error is **not**null
- SkipSet is set to true
- StopSet is set to true
- ReturnSet is set to true

ShouldReturnFunction

Should the interpreter return from the current *function* execution?

```
public bool ShouldReturnFunction { get; }
```

Property Value

bool

Remarks

This is true when either of the below conditions are met:

- Error is **not**null

ShouldReturnLoop

Should the interpreter return from the current *loop* execution?

```
public bool ShouldReturnLoop { get; }
```

Property Value

bool

Remarks

This is true when either of the below conditions are met:

- Error is **not**null
- ReturnSet is set to true

ShouldReturnTryCatch

Should the interpreter return from the current *try-catch block* execution?

```
public bool ShouldReturnTryCatch { get; }
```

Property Value

bool

Remarks

This is true when either of the below conditions are met:

- SkipSet is set to true
- StopSet is set to true
- ReturnSet is set to true

Methods

Failure(IEzrRuntimeError)

Sets a failed expression execution.

```
public void Failure(IEzrRuntimeError error)
```

Parameters

`error` IEzrRuntimeError

The error that caused the failure.

GetReferenceCopyIfReleasable()

Creates a shallow copy of the current Reference if it is eligible for release by the pool.

```
public Reference GetReferenceCopyIfReleasable()
```

Returns

Reference

The copy of the reference, or the reference itself if not eligible for release.

Reset(Reference?)

Resets the current RuntimeResult.

```
public void Reset(Reference? exclude = null)
```

Parameters

`exclude` Reference

Reference to exclude from release.

Remarks

This:

- Sets SkipSet, StopSet and ReturnSet to false.
- Releases and sets Reference to Empty, unless specifically excluded.

SetReference(Reference)

Sets Reference.

```
private void SetReference(Reference reference)
```

Parameters

`reference` Reference

The new value for Reference.

SetReturnFlag(Reference)

Sets the ReturnSet flag, signifying a function return.

```
public void SetReturnFlag(Reference reference)
```

Parameters

`reference` Reference

The reference to the object returned by the function.

SetSkipFlag(Reference)

Sets the SkipSet flag, signifying a loop skip.

```
public void SetSkipFlag(Reference reference)
```

Parameters

`reference` Reference

The reference to return when the skip flag is used outside a loop.

SetStopFlag(Reference)

Sets the StopSet flag, signifying a loop stop.

```
public void SetStopFlag(Reference reference)
```

Parameters

`reference` Reference

The reference to return when the stop flag is used outside a loop.

Success(Reference)

Sets a successful expression execution.

```
public void Success(Reference reference)
```

Parameters

`reference` Reference

The reference to the resulting IEzrObject.

Namespace EzrSquared.Runtime.Collections

Classes

RuntimeEzrObjectDictionary

A Dictionary for IEzrObjects.

RuntimeEzrObjectList

A List for References.

Class RuntimeEzrObjectDictionary

Namespace: EzrSquared.Runtime.Collections

Assembly: ezrSquared-lib.dll

A Dictionary for IEzrObjects.

```
public class RuntimeEzrObjectDictionary : IMutable<RuntimeEzrObjectDictionary>,
IEnumerable<KeyValuePair<int, KeyValuePair<IEzrObject, Reference>>>, IEnumerable
```

Inheritance

object ← RuntimeEzrObjectDictionary

Implements

IMutable<RuntimeEzrObjectDictionary>,
IEnumerable<KeyValuePair<int, KeyValuePair<IEzrObject, Reference>>>, IEnumerable

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

RuntimeEzrObjectDictionary()

Creates a new RuntimeEzrObjectDictionary.

```
public RuntimeEzrObjectDictionary()
```

RuntimeEzrObjectDictionary(IDictionary<int, KeyValuePair<IEzrObject, Reference>>)

Creates a new RuntimeEzrObjectDictionary from an existing Dictionary<TKey, TValue>.

```
public RuntimeEzrObjectDictionary(IDictionary<int, KeyValuePair<IEzrObject,
Reference>> items)
```

Parameters

`items` IDictionary<int, KeyValuePair<IEzrObject, Reference>>

Fields

_items

The collection of IEzrObjects stored in the RuntimeEzrObjectDictionary, in the format Dictionary<hashOfKey, KeyValuePair<key, valueReference>>.

```
private readonly IDictionary<int, KeyValuePair<IEzrObject, Reference>> _items
```

Field Value

IDictionary<int, KeyValuePair<IEzrObject, Reference>>

Properties

Count

The number of IEzrObjects in the RuntimeEzrObjectDictionary.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

Methods

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<RuntimeEzrObjectDictionary>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<RuntimeEzrObjectDictionary>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

~RuntimeEzrObjectDictionary()

Destructor.

```
protected ~RuntimeEzrObjectDictionary()
```

Get(IEzrObject, RuntimeResult)

Tries retrieving a value from the RuntimeEzrObjectDictionary.

```
public Reference Get(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key of the value to be returned.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

Reference

The retrieved value or Empty if it was not found.

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<KeyValuePair<int, KeyValuePair<IEzrObject, Reference>>> GetEnumerator()
```

Returns

IEnumerator<KeyValuePair<int, KeyValuePair<IEzrObject, Reference>>>

An enumerator that can be used to iterate through the collection.

Remarks

Unlike GetKeys(RuntimeResult) or GetPairs(RuntimeResult) this method does NOT copy the keys. So it's best not to expose the key values from this to the ezr^2 runtime as mutable key objects can be changed.

GetKeys(RuntimeResult)

Retrieves all keys from the RuntimeEzrObjectDictionary.

```
public IEzrObject[]? GetKeys(RuntimeResult result)
```

Parameters

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

IEzrObject[]

The keys or null if something went wrong.

GetPairs(RuntimeResult)

Retrieves all keys and values from the RuntimeEzrObjectDictionary.

```
public KeyValuePair<IEzrObject, IEzrObject>[]? GetPairs(RuntimeResult result)
```

Parameters

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

KeyValuePair<IEzrObject, IEzrObject>[]

The keys and values as an array of KeyValuePairs or null if something went wrong.

GetRealKeys()

Returns the *actual* integer (hash) keys from the RuntimeEzrObjectDictionary.

```
public int[] GetRealKeys()
```

Returns

int[]

The integer hashes.

GetValues()

Retrieves all values from the RuntimeEzrObjectDictionary.

```
public IEzrObject[] GetValues()
```

Returns

IEzrObject[]

The values.

HasKey(IEzrObject, RuntimeResult)

Checks if `key` exists in the RuntimeEzrObjectDictionary.

```
public bool HasKey(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key to be checked.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

bool

true if the key was found, false if any error occured or the key was not found.

IsEqual(RuntimeEzrObjectDictionary, RuntimeResult)

Checks if the current dictionary is equal to the given dictionary.

```
public bool IsEqual(RuntimeEzrObjectDictionary other, RuntimeResult result)
```

Parameters

`other` RuntimeEzrObjectDictionary

The other dictionary.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

bool

The comparison result.

IsEqual(IEzrDictionary, Context, RuntimeResult)

Checks if the current dictionary is equal to the given keyed collection.

```
public bool IsEqual(IEzrDictionary other, Context executionContext, RuntimeResult result)
```

Parameters

`other` IEzrDictionary

The other keyed collection.

`executionContext` Context

The context under which this operation is being executed.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

bool

The comparison result.

Merge(RuntimeEzrObjectDictionary, RuntimeResult)

Merges a RuntimeEzrObjectDictionary to the current RuntimeEzrObjectDictionary.

```
public void Merge(RuntimeEzrObjectDictionary other, RuntimeResult result)
```

Parameters

`other` RuntimeEzrObjectDictionary

The other RuntimeEzrObjectDictionary to be merged.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Merge(IEzrDictionary, Context, RuntimeResult)

Merges an IEzrDictionary to the current RuntimeEzrObjectDictionary.

```
public void Merge(IEzrDictionary other, Context executionContext, RuntimeResult result)
```

Parameters

`other` IEzrDictionary

The other IEzrDictionary to be merged.

`executionContext` Context

The context under which this operation is being executed.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Release()

Releases the references associated with the dictionary, and clears it.

```
public void Release()
```

Remove(IEzrObject, RuntimeResult)

Removes a key, value pair from the RuntimeEzrObjectDictionary.

```
public bool Remove(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key to be removed.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

bool

true if the operation was successful, false if not.

RemoveHash(int)

Removes a key, value pair from the RuntimeEzrObjectDictionary.

```
[SharpAutoWrapper("remove_by_hash", false, false)]
public bool RemoveHash(int key)
```

Parameters

`key` int

The key (hash) to be removed.

Returns

bool

true if the operation was successful, false if not.

Remarks

This method removes pairs using the hash of the IEzrObject keys, `key`.

TryCopyObject(IEzrObject, RuntimeResult)

Tries to get a copy of the given object.

```
private static IEzrObject? TryCopyObject(IEzrObject ezrObject, RuntimeResult result)
```

Parameters

`ezrObject` IEzrObject

The object to copy.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Returns

IEzrObject

The object, its copy or null if something went wrong.

Update(IEzrObject, IEzrObject, RuntimeResult)

Updates the RuntimeEzrObjectDictionary with a new value.

```
public void Update(IEzrObject key, IEzrObject value, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key to update.

`value` IEzrObject

The value to assign to `key`.

`result` RuntimeResult

The RuntimeResult object for returning errors.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Remarks

Unlike GetKeys(RuntimeResult) or GetPairs(RuntimeResult) this method does NOT copy the keys. So it's best not to expose the key values from this to the ezr² runtime as mutable key objects can be changed.

Class RuntimeEzrObjectList

Namespace: EzrSquared.Runtime.Collections

Assembly: ezrSquared-lib.dll

A List for References.

```
public class RuntimeEzrObjectList : List<Reference>, IList<Reference>,
ICollection<Reference>, IReadOnlyList<Reference>, IReadOnlyCollection<Reference>,
IEnumerable<Reference>, IList, ICollection, IEnumerable, IMutable<RuntimeEzrObjectList>
```

Inheritance

object ← List<Reference> ← RuntimeEzrObjectList

Implements

IList<Reference>, ICollection<Reference>, IReadOnlyList<Reference>,
IReadOnlyCollection<Reference>, IEnumerable<Reference>, IList, ICollection, IEnumerable,
IMutable<RuntimeEzrObjectList>

Inherited Members

List<Reference>.IList.get_Item(int) , List<Reference>.IList.set_Item(int, object) ,
List<Reference>.AsReadOnly() ,
List<Reference>.BinarySearch(int, int, Reference, IComparer<Reference>) ,
List<Reference>.BinarySearch(Reference) ,
List<Reference>.BinarySearch(Reference, IComparer<Reference>) , List<Reference>.Clear() ,
List<Reference>.Contains(Reference) ,
List<Reference>.ConvertAll<TOutput>(Converter<Reference, TOutput>) ,
List<Reference>.CopyTo(int, Reference[], int, int) , List<Reference>.CopyTo(Reference[]) ,
List<Reference>.CopyTo(Reference[], int) , List<Reference>.EnsureCapacity(int) ,
List<Reference>.Exists(Predicate<Reference>) , List<Reference>.Find(Predicate<Reference>) ,
List<Reference>.FindAll(Predicate<Reference>) ,
List<Reference>.FindIndex(int, int, Predicate<Reference>) ,
List<Reference>.FindIndex(int, Predicate<Reference>) ,
List<Reference>.FindIndex(Predicate<Reference>) ,
List<Reference>.FindLast(Predicate<Reference>) ,
List<Reference>.FindLastIndex(int, int, Predicate<Reference>) ,
List<Reference>.FindLastIndex(int, Predicate<Reference>) ,
List<Reference>.FindLastIndex(Predicate<Reference>) ,
List<Reference>.ForEach(Action<Reference>) , List<Reference>.GetEnumerator() ,
List<Reference>.IndexOf(Reference) , List<Reference>.IndexOf(Reference, int) ,

List<Reference>.IndexOf(Reference, int, int) , List<Reference>.Insert(int, Reference) ,
List<Reference>.InsertRange(int, IEnumerable<Reference>) ,
List<Reference>.LastIndexOf(Reference) , List<Reference>.LastIndexOf(Reference, int) ,
List<Reference>.LastIndexOf(Reference, int, int) , List<Reference>.Remove(Reference) ,
List<Reference>.RemoveAll(Predicate<Reference>) , List<Reference>.Reverse() ,
List<Reference>.Reverse(int, int) , List<Reference>.Slice(int, int) , List<Reference>.Sort() ,
List<Reference>.Sort(IComparer<Reference>) , List<Reference>.Sort(Comparison<Reference>) ,
List<Reference>.Sort(int, int, IComparer<Reference>) ,
List<Reference>.IEnumerable<Reference>.GetEnumerator() ,
List<Reference>.ICollection.CopyTo(Array, int) , List<Reference>.IEnumerable.GetEnumerator() ,
List<Reference>.IList.Add(object) , List<Reference>.IList.Contains(object) ,
List<Reference>.IList.IndexOf(object) , List<Reference>.IList.Insert(int, object) ,
List<Reference>.IList.Remove(object) , List<Reference>.ToArray() , List<Reference>.TrimExcess() ,
List<Reference>.TrueForAll(Predicate<Reference>) , List<Reference>.Capacity ,
List<Reference>.Count , List<Reference>.this[int] ,
List<Reference>.ICollection<Reference>.IsReadOnly , List<Reference>.ICollection.IsSynchronized ,
List<Reference>.ICollection.SyncRoot , List<Reference>.IList.IsFixedSize ,
List<Reference>.IList.IsReadOnly , List<Reference>.IList.Item[int] , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

RuntimeEzrObjectList()

Creates a new RuntimeEzrObjectList.

```
public RuntimeEzrObjectList()
```

RuntimeEzrObjectList(IEnumerable<Reference>)

Creates a new RuntimeEzrObjectList based on another IEnumerable<T>.

```
public RuntimeEzrObjectList(IEnumerable<Reference> collection)
```

Parameters

`collection` IEnumerable<Reference>

The base collection of References.

RuntimeEzrObjectList(int)

Creates a new RuntimeEzrObjectList with the specified capacity.

```
public RuntimeEzrObjectList(int capacity)
```

Parameters

`capacity` int

The capacity of the list.

Methods

Add(Reference)

Adds a new reference to the list.

```
public void Add(Reference reference)
```

Parameters

`reference` Reference

The reference to add.

AddRange(IEnumerable<Reference>)

Adds multiple new reference to the list.

```
public void AddRange(IEnumerable<Reference> references)
```

Parameters

`references` IEnumerable<Reference>

The references to add.

Remarks

This actually creates a new Reference object for each added reference.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<RuntimeEzrObjectList>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<RuntimeEzrObjectList>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

~RuntimeEzrObjectList()

Destructor.

```
protected ~RuntimeEzrObjectList()
```

GetRange(int, int)

Creates a shallow copy of a range of elements in the list.

```
public RuntimeEzrObjectList GetRange(int index, int length)
```

Parameters

`index` int

The index to start the range.

`length` int

The length of the range.

Returns

RuntimeEzrObjectList

The copy.

Release()

Releases the references associated with the list, and clears it.

```
public void Release()
```

RemoveAt(int)

Removes an element in the list at the given index.

```
public void RemoveAt(int index)
```

Parameters

`index` int

The index.

RemoveRange(int, int)

Removes a range of elements in the list.

```
public void RemoveRange(int index, int length)
```

Parameters

`index` int

The index to start the range.

`length` int

The length of the range.

Namespace EzrSquared.Runtime.Nodes

Classes

ArrayLikeNode
: The Node structure for an arraylike (array or list).

BinaryOperationNode
: The Node structure for a binary operation.

CallNode
: The Node structure for a function call.

ClassDefinitionNode
: The Node structure for a class definition.

CountNode
: The Node structure for a count expression.

DefineBlockNode
: The Node structure for a define block.

DictionaryNode
: The Node structure for a dictionary.

ForEachNode
: The Node structure for a for-each expression.

FunctionDefinitionNode
: The Node structure for a function definition.

IfNode
: The Node structure for an if expression.

IncludeNode
: The Node structure for an include expression.

InvalidNode
: The dummy invalid Node structure. For returning instead of null if an EzrSyntaxError occurs during parsing.

NoValueNode
: The Node structure for a statement or expression without any value (used as skip and stop statement nodes).

Node

The representation of an ezr² source code construct. This is the base class of all nodes. Only for inheritance!

ReturnNode

The Node structure for a return statement.

TryNode

The Node structure for a try expression.

UnaryOperationNode

The Node structure for a unary operation.

ValueNode

The Node structure of a simple value like literals and variables.

VariableAccessNode

The Node structure for accesing a variable from the context.

VariableAssignmentNode

The Node structure for assigning a value to a variable in the context.

WhileNode

The Node structure for an while expression.

Class ArrayLikeNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for an arraylike (array or list).

```
public class ArrayLikeNode : Node
```

Inheritance

object ← Node ← ArrayLikeNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

ArrayLikeNode(List<Node>, bool, Position, Position)

The Node structure for an arraylike (array or list).

```
public ArrayLikeNode(List<Node> elements, bool createList, Position startPosition,
Position endPosition)
```

Parameters

`elements` List<Node>

The elements of the arraylike.

`createList` bool

The check for if the ArrayLikeNode should create a list instead of an array.

`startPosition` Position

The starting Position of the ArrayLikeNode.

`endPosition` Position

The ending Position of the ArrayLikeNode.

Fields

CreateList

The check for if the ArrayLikeNode should create a list instead of an array.

```
public bool CreateList
```

Field Value

bool

Elements

The elements of the arraylike.

```
public List<Node> Elements
```

Field Value

List<Node>

Methods

ToString()

Creates a string representation of the ArrayLikeNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class BinaryOperationNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a binary operation.

```
public class BinaryOperationNode : Node
```

Inheritance

object ← Node ← BinaryOperationNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

BinaryOperationNode(Node, Node, TokenType, Position, Position)

The Node structure for a binary operation.

```
public BinaryOperationNode(Node left, Node right, TokenType @operator, Position
startPosition, Position endPosition)
```

Parameters

`left` Node

The first operand of the binary operation.

`right` Node

The second operand of the binary operation.

`operator` TokenType

The operator TokenType of the binary operation.

`startPosition` Position

The starting Position of the BinaryOperationNode.

`endPosition` Position

The ending Position of the BinaryOperationNode.

Fields

Left

The first operand of the binary operation.

```
public Node Left
```

Field Value

Node

Operator

The operator TokenType of the binary operation.

```
public TokenType Operator
```

Field Value

TokenType

Right

The second operand of the binary operation.

```
public Node Right
```

Field Value

Node

Methods

ToString()

Creates a string representation of the BinaryOperationNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class CallNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a function call.

```
public class CallNode : Node
```

Inheritance

object ← Node ← CallNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

CallNode(Node, List<Node>, Position, Position)

The Node structure for a function call.

```
public CallNode(Node receiver, List<Node> arguments, Position startPosition,
Position endPosition)
```

Parameters

`receiver` Node

The function/object to be called.

`arguments` List<Node>

The array of arguments.

`startPosition` Position

The starting Position of the CallNode.

`endPosition` Position

The ending Position of the CallNode.

Fields

Arguments

The array of arguments.

```
public List<Node> Arguments
```

Field Value

List<Node>

Receiver

The function/object to be called.

```
public Node Receiver
```

Field Value

Node

Methods

ToString()

Creates a string representation of the CallNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class ClassDefinitionNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a class definition.

```
public class ClassDefinitionNode : Node
```

Inheritance

object ← Node ← ClassDefinitionNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

ClassDefinitionNode(Node?, AccessMod, bool, List<Node>, Node, Position, Position)

The Node structure for a class definition.

```
public ClassDefinitionNode(Node? name, AccessMod accessibilityModifiers, bool @readonly,
List<Node> parents, Node body, Position startPosition, Position endPosition)
```

Parameters

`name` Node

The (optional) name of the class.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the class definition.

`readonly` bool

The check for if the class should be declared read-only.

`parents` List<Node>

The parents of the class.

`body` Node

The body of the class.

`startPosition` Position

The starting Position of the ClassDefinitionNode.

`endPosition` Position

The ending Position of the ClassDefinitionNode.

Fields

AccessibilityModifiers

The accessibility modifiers for the class definition.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

Body

The body of the class.

```
public Node Body
```

Field Value

Node

Name

The name of the class. May be null.

```
public Node? Name
```

Field Value

Node

Parents

The parents of the class.

```
public List<Node> Parents
```

Field Value

List<Node>

Readonly

The check for if the class should be declared read-only.

```
public bool Readonly
```

Field Value

bool

Methods

ToString()

Creates a string representation of the ClassDefinitionNode.
All fields excluding the start and end positions are included in the representation, like

"ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class CountNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a count expression.

```
public class CountNode : Node
```

Inheritance

object ← Node ← CountNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

CountNode(Node, Node?, Node?, Node?, Node, Position, Position)

The Node structure for a count expression.

```
public CountNode(Node to, Node? from, Node? step, Node? iterationVariable, Node body,
Position startPosition, Position endPosition)
```

Parameters

`to` Node

The amount to count to.

`from` Node

The amount to count from - optional.

`step` Node

The increment of each iteration - optional.

`iterationVariable` Node

The variable to store the iteration number in - optional.

`body` Node

The body of the count loop.

`startPosition` Position

The starting Position of the CountNode.

`endPosition` Position

The ending Position of the CountNode.

Fields

Body

The body of the count loop.

```
public Node Body
```

Field Value

Node

From

The amount to count from - optional.

```
public Node? From
```

Field Value

Node

IterationVariable

The variable to store the iteration number in - optional.

```
public Node? IterationVariable
```

Field Value

Node

Step

The increment of each iteration - optional.

```
public Node? Step
```

Field Value

Node

To

The amount to count to.

```
public Node To
```

Field Value

Node

Methods

ToString()

Creates a string representation of the CountNode.
All fields excluding the start and end positions are included in the representation, like

"ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class DefineBlockNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a define block.

```
public class DefineBlockNode : Node
```

Inheritance

object ← Node ← DefineBlockNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

DefineBlockNode(Node, AccessMod, Position, Position)

The Node structure for a define block.

```
public DefineBlockNode(Node body, AccessMod accessibilityModifiers, Position startPosition,
Position endPosition)
```

Parameters

`body` Node

The body of the block.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the define block.

`startPosition` Position

The starting Position of the DefineBlockNode.

`endPosition` Position

The ending Position of the DefineBlockNode.

Fields

AccessibilityModifiers

The accessibility modifiers of the define block.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

Body

The body of the block.

```
public Node Body
```

Field Value

Node

Methods

ToString()

Creates a string representation of the DefineBlockNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class DictionaryNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a dictionary.

```
public class DictionaryNode : Node
```

Inheritance

object ← Node ← DictionaryNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

DictionaryNode(List<(Node Key, Node Value)>, Position, Position)

The Node structure for a dictionary.

```
public DictionaryNode(List<(Node Key, Node Value)> keyValuePairs, Position startPosition,
Position endPosition)
```

Parameters

`keyValuePairs` List<(Node Key, Node Value)>

The key-value pairs of the dictionary.

`startPosition` Position

The starting Position of the DictionaryNode.

`endPosition` Position

The ending Position of the DictionaryNode.

Fields

KeyValuePairs

The key-value pairs of the dictionary.

```
public List<(Node Key, Node Value)> KeyValuePairs
```

Field Value

List<(Node Key, Node Value)>

Methods

ToString()

Creates a string representation of the DictionaryNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class ForEachNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a for-each expression.

```
public class ForEachNode : Node
```

Inheritance

object ← Node ← ForEachNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

ForEachNode(BinaryOperationNode, Node, Position, Position)

The Node structure for a for-each expression.

```
public ForEachNode(BinaryOperationNode expression, Node body, Position startPosition,
Position endPosition)
```

Parameters

`expression` BinaryOperationNode

A check-in expression, where the LHS is the variable to store the iterated values and RHS is the collection to iterate.

`body` Node

The body of the loop.

`startPosition` Position

The starting Position of the ForEachNode.

`endPosition` Position

The ending Position of the ForEachNode.

Fields

Body

The body of the count loop.

```
public Node Body
```

Field Value

Node

Expression

A check-in expression, where the LHS is the variable to store the iterated values and RHS is the collection to iterate.

```
public BinaryOperationNode Expression
```

Field Value

BinaryOperationNode

Methods

ToString()

Creates a string representation of the ForEachNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class FunctionDefinitionNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a function definition.

```
public class FunctionDefinitionNode : Node
```

Inheritance

object ← Node ← FunctionDefinitionNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

FunctionDefinitionNode(Node?, AccessMod, bool, List<Node>, Node?, Node?, Node, Position, Position)

The Node structure for a function definition.

```
public FunctionDefinitionNode(Node? name, AccessMod accessibilityModifiers, bool returnLast,
List<Node> parameters, Node? extraKeywordArguments, Node? extraPositionalArguments, Node
body, Position startPosition, Position endPosition)
```

Parameters

`name` Node

The (optional) name of the function.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the function definition.

`returnLast` bool

The check for if the last expression of the function should be returned as its result. Only used in oneliners.

`parameters` List<Node>

The parameters of the function.

`extraKeywordArguments` Node

The reference to store the extra keyword arguments in.

`extraPositionalArguments` Node

The reference to store the extra positional arguments in.

`body` Node

The body of the function.

`startPosition` Position

The starting Position of the FunctionDefinitionNode.

`endPosition` Position

The ending Position of the FunctionDefinitionNode.

Fields

AccessibilityModifiers

The accessibility modifiers for the function definition.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

Body

The body of the function.

```
public Node Body
```

Field Value

Node

ExtraKeywordArguments

The reference to store the extra keyword arguments in.

```
public Node? ExtraKeywordArguments
```

Field Value

Node

ExtraPositionalArguments

The reference to store the extra positional arguments in.

```
public Node? ExtraPositionalArguments
```

Field Value

Node

Name

The (optional) name of the function.

```
public Node? Name
```

Field Value

Node

Parameters

The parameters of the function.

```
public List<Node> Parameters
```

Field Value

List<Node>

ReturnLast

The check for if the last expression of the function should be returned as its result. Only used in oneliners.

```
public bool ReturnLast
```

Field Value

bool

Methods

ToString()

Creates a string representation of the FunctionDefinitionNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class IfNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for an if expression.

```
public class IfNode : Node
```

Inheritance

object ← Node ← IfNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

IfNode(List<(Node Condition, Node Body)>, Node?, Position, Position)

The Node structure for an if expression.

```
public IfNode(List<(Node Condition, Node Body)> cases, Node? elseCase, Position
startPosition, Position endPosition)
```

Parameters

`cases` List<(Node Key, Node Value)>

The cases of the if expression.

`elseCase` Node

The body of the else case.

`startPosition` Position

The starting Position of the IfNode.

`endPosition` Position

The ending Position of the IfNode.

Fields

Cases

The cases of the if expression.

```
public List<(Node Condition, Node Body)> Cases
```

Field Value

List<(Node Key, Node Value)>

ElseCase

The body of the else case.

```
public Node? ElseCase
```

Field Value

Node

Methods

ToString()

Creates a string representation of the IfNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class IncludeNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for an include expression.

```
public class IncludeNode : Node
```

Inheritance

object ← Node ← IncludeNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

IncludeNode(Node, Node?, bool, Node?, Position, Position)

The Node structure for an include expression.

```
public IncludeNode(Node script, Node? subStructure, bool isDumped, Node? nickname, Position
startPosition, Position endPosition)
```

Parameters

`script` Node

The script to include.

`subStructure` Node

The (optional) specific sub-structure or object to be included from the script.

`isDumped` bool

Specifies if all contents of the script need to be dumped into the current context.

`nickname` Node

The (optional) nickname of the object to be included.

`startPosition` Position

The starting Position of the IncludeNode.

`endPosition` Position

The ending Position of the IncludeNode.

Fields

IsDumped

Specifies if all contents of the script need to be dumped into the current context.

```
public readonly bool IsDumped
```

Field Value

bool

Nickname

The (optional) nickname of the object to be included.

```
public readonly Node? Nickname
```

Field Value

Node

Script

The script to include.

```
public readonly Node Script
```

Field Value

Node

SubStructure

The (optional) specific sub-structure or object to be included from the script.

```
public readonly Node? SubStructure
```

Field Value

Node

Methods

ToString()

Creates a string representation of the IncludeNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class InvalidNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The dummy invalid Node structure. For returning instead of null if an EzrSyntaxError occurs during parsing.

```
public class InvalidNode : Node
```

Inheritance

object ← Node ← InvalidNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

InvalidNode(Position, Position)

Creates a new InvalidNode object.

```
private InvalidNode(Position startPosition, Position endPosition)
```

Parameters

`startPosition` Position

The starting Position of the InvalidNode.

`endPosition` Position

The ending Position of the InvalidNode.

Fields

s_invalidNode

The static InvalidNode object.

```
internal static readonly InvalidNode s_invalidNode
```

Field Value

InvalidNode

Methods

ToString()

Creates a string representation of the InvalidNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

> The string representation.

Class NoValueNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a statement or expression without any value (used as skip and stop statement nodes).

```
public class NoValueNode : Node
```

Inheritance

object ← Node ← NoValueNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

NoValueNode(TokenType, Position, Position)

The Node structure for a statement or expression without any value (used as skip and stop statement nodes).

```
public NoValueNode(TokenType valueType, Position startPosition, Position endPosition)
```

Parameters

`valueType` TokenType

The identifying TokenType of the NoValueNode.

`startPosition` Position

The starting Position of the NoValueNode.

`endPosition` Position

The ending Position of the NoValueNode.

Fields

ValueType

The identifying TokenType of the NoValueNode.

```
public TokenType ValueType
```

Field Value

TokenType

Methods

ToString()

Creates a string representation of the NoValueNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class Node

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The representation of an ezr² source code construct. This is the base class of all nodes. Only for inheritance!

```
public abstract class Node
```

Inheritance

object ← Node

Derived

ArrayLikeNode, BinaryOperationNode, CallNode, ClassDefinitionNode, CountNode, DefineBlockNode, DictionaryNode, ForEachNode, FunctionDefinitionNode, IfNode, IncludeNode, InvalidNode, NoValueNode, ReturnNode, TryNode, UnaryOperationNode, ValueNode, VariableAccessNode, VariableAssignmentNode, WhileNode

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

Node(Position, Position)

The representation of an ezr² source code construct. This is the base class of all nodes. Only for inheritance!

```
protected Node(Position startPosition, Position endPosition)
```

Parameters

`startPosition` Position

The starting Position of the Node.

`endPosition` Position

The ending Position of the Node.

Fields

EndPosition

The ending Position of the Node.

```
public Position EndPosition
```

Field Value

Position

StartPosition

The starting Position of the Node.

```
public Position StartPosition
```

Field Value

Position

Methods

ToString()

Creates a string representation of the Node.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class ReturnNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a return statement.

```
public class ReturnNode : Node
```

Inheritance

object ← Node ← ReturnNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

ReturnNode(Node?, bool, Position, Position)

The Node structure for a return statement.

```
public ReturnNode(Node? value, bool returnLast, Position startPosition,
Position endPosition)
```

Parameters

`value` Node

The optional value to be returned.

`returnLast` bool

Return the last element of Value, which should be a list or array.

`startPosition` Position

The starting Position of the ReturnNode.

`endPosition` Position

The ending Position of the ReturnNode.

Fields

ReturnLast

Return the last element of Value, which should be a list or array.

```
public bool ReturnLast
```

Field Value

bool

Value

The optional value to be returned.

```
public Node? Value
```

Field Value

Node

Methods

ToString()

Creates a string representation of the ReturnNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class TryNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a try expression.

```
public class TryNode : Node
```

Inheritance

object ← Node ← TryNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

TryNode(Node, List<(Node ErrorType, Node? Variable, Node Body)>, (Node? Variable, Node Body)?, Position, Position)

The Node structure for a try expression.

```
public TryNode(Node block, List<(Node ErrorType, Node? Variable, Node Body)> cases, (Node?
Variable, Node Body)? emptyCase, Position startPosition, Position endPosition)
```

Parameters

`block` Node

The try block.

`cases` List<(Node ErrorType, Node Variable, Node Body)>

The error cases of the try expression.

`emptyCase` (Node Variable, Node Body)?

The (optional) Node where the error will be stored and the body of the empty else case.

`startPosition` Position

The starting Position of the TryNode.

`endPosition` Position

The ending Position of the TryNode.

Fields

Block

The try block.

```
public Node Block
```

Field Value

Node

Cases

The error cases of the try expression.

```
public List<(Node ErrorType, Node? Variable, Node Body)> Cases
```

Field Value

List <(Node ErrorType, Node Variable, Node Body)>

EmptyCase

The (optional) Node where the error will be stored and the body of the empty else case.

```
public (Node? Variable, Node Body)? EmptyCase
```

Field Value

(Node Variable, Node Body)?

Methods

ToString()

Creates a string representation of the TryNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class UnaryOperationNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for a unary operation.

```
public class UnaryOperationNode : Node
```

Inheritance

object ← Node ← UnaryOperationNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

UnaryOperationNode(Node, TokenType, Position, Position)

The Node structure for a unary operation.

```
public UnaryOperationNode(Node operand, TokenType @operator, Position startPosition,
Position endPosition)
```

Parameters

`operand` Node

The operand of the unary operation.

`operator` TokenType

The operator TokenType of the unary operation.

`startPosition` Position

The starting Position of the UnaryOperationNode.

`endPosition` Position

The ending Position of the UnaryOperationNode.

Fields

Operand

The operand of the unary operation.

```
public Node Operand
```

Field Value

Node

Operator

The operator TokenType of the unary operation.

```
public TokenType Operator
```

Field Value

TokenType

Methods

ToString()

Creates a string representation of the UnaryOperationNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class ValueNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure of a simple value like literals and variables.

```
public class ValueNode : Node
```

Inheritance

object ← Node ← ValueNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

ValueNode(Token, Position, Position)

The Node structure of a simple value like literals and variables.

```
public ValueNode(Token value, Position startPosition, Position endPosition)
```

Parameters

`value` Token

The Token value the ValueNode represents.

`startPosition` Position

The starting Position of the ValueNode.

`endPosition` Position

The ending Position of the ValueNode.

Fields

Value

The value the ValueNode represents.

```
public Token Value
```

Field Value

Token

Methods

ToString()

Creates a string representation of the ValueNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class VariableAccessNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for accesing a variable from the context.

```
public class VariableAccessNode : Node
```

Inheritance

object ← Node ← VariableAccessNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

VariableAccessNode(Token, AccessMod, Position, Position)

The Node structure for accesing a variable from the context.

```
public VariableAccessNode(Token name, AccessMod accessibilityModifiers, Position
startPosition, Position endPosition)
```

Parameters

`name` Token

The name of the variable, a Token object of type Identifier.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the variable access operation.

`startPosition` Position

The starting Position of the VariableAccessNode.

`endPosition` Position

The ending Position of the VariableAccessNode.

Fields

AccessibilityModifiers

The accessibility modifiers for the variable access operation.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

Name

The name of the variable to access.

```
public Token Name
```

Field Value

Token

Methods

ToString()

Creates a string representation of the VariableAccessNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class VariableAssignmentNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for assigning a value to a variable in the context.

```
public class VariableAssignmentNode : Node
```

Inheritance

object ← Node ← VariableAssignmentNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

VariableAssignmentNode(Node, TokenType, Node, AccessMod, Position, Position)

The Node structure for assigning a value to a variable in the context.

```
public VariableAssignmentNode(Node variable, TokenType assignmentOperator, Node value,
AccessMod accessibilityModifiers, Position startPosition, Position endPosition)
```

Parameters

`variable` Node

The variable to be assigned to.

`assignmentOperator` TokenType

The operation TokenType, if not Colon, between the existing value of `variable` and `value`. The result of the operation will be assigned to `variable`.

`value` Node

The value to be assigned to Variable.

`accessibilityModifiers` AccessMod

The accessibility modifiers for the variable assignment operation.

`startPosition` Position

The starting Position of the VariableAssignmentNode.

`endPosition` Position

The ending Position of the VariableAssignmentNode.

Fields

AccessibilityModifiers

The accessibility modifiers for the variable assignment operation.

```
public AccessMod AccessibilityModifiers
```

Field Value

AccessMod

AssignmentOperator

The operation TokenType, if not Colon, between the existing value of Variable and Value. The result of the operation will be assigned to Variable.

```
public TokenType AssignmentOperator
```

Field Value

TokenType

Value

The value to be assigned to Variable.

```
public Node Value
```

Field Value

Node

Variable

The variable to be assigned to.

```
public Node Variable
```

Field Value

Node

Methods

ToString()

Creates a string representation of the VariableAssignmentNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Class WhileNode

Namespace: EzrSquared.Runtime.Nodes

Assembly: ezrSquared-lib.dll

The Node structure for an while expression.

```
public class WhileNode : Node
```

Inheritance

object ← Node ← WhileNode

Inherited Members

Node.StartPosition , Node.EndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

WhileNode(Node, Node, Position, Position)

The Node structure for an while expression.

```
public WhileNode(Node condition, Node body, Position startPosition, Position endPosition)
```

Parameters

`condition` Node

The condition of the while loop.

`body` Node

The body of the while loop.

`startPosition` Position

The starting Position of the WhileNode.

`endPosition` Position

The ending Position of the WhileNode.

Fields

Body

The body of the while loop.

```
public Node Body
```

Field Value

Node

Condition

The condition of the while loop.

```
public Node Condition
```

Field Value

Node

Methods

ToString()

Creates a string representation of the WhileNode.
All fields excluding the start and end positions are included in the representation, like "ExampleNode(Property1, Property2)".

```
public override string ToString()
```

Returns

string

The string representation.

Namespace EzrSquared.Runtime.Types

Classes

EzrObject

The base root class of all built-in objects. Provides utility functions and bare-minimum operator handling.

EzrRuntimeInvalidObject

An invalid, sort of "empty" object, to use instead of null.

Interfaces

IEzrMutableObject

A mutable IEzrObject.

IEzrObject

An object in the ezr² language.

Class EzrObject

Namespace: EzrSquared.Runtime.Types

Assembly: ezrSquared-lib.dll

The base root class of all built-in objects. Provides utility functions and bare-minimum operator handling.

```
public abstract class EzrObject : IEzrObject
```

Inheritance

object ← EzrObject

Implements

IEzrObject

Derived

EzrSharpCompatibilityWrapper<TMemberInfo>, EzrSharpSourceExecutableWrapper, EzrArray, EzrDictionary, EzrList, EzrRuntimeError, EzrBoolean, EzrNothing, EzrFloat, EzrInteger, EzrCharacter, EzrCharacterList, EzrString, EzrClassInstance, EzrRuntimeExecutable, EzrRuntimeInvalidObject

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrObject(Context, Position, Position)

Creates a new object with the specified parent context and position.

```
public EzrObject(Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

EzrObject(Context?, Context, Position, Position)

Creates a new object with the specified internal context, parent context and position.

```
public EzrObject(Context? context, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`context` Context

The internal context, if null, creates a new one.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

_executionContext

The current context in which the operation is being executed.

```
protected internal Context _executionContext
```

Field Value

Context

_hashTag

The hash of Tag.

```
private int _hashTag
```

Field Value

int

s_memberMap

Cache of reflection data for all EzrObject types.

```
internal static readonly Lazy<Dictionary<(Type ParentType, string MemberName),
MemberInfo>> s_memberMap
```

Field Value

Lazy<Dictionary<(Type ParentType, string MemberName), MemberInfo>>

Properties

Context

The Context of the object.

```
public Context Context { get; private set; }
```

Property Value

Context

CreationContext

The context under which the object was created.

```
public Context CreationContext { get; private set; }
```

Property Value

Context

EndPosition

The ending position of the object in source code.

```
public Position EndPosition { get; private set; }
```

Property Value

Position

HashTag

The hash of Tag.

```
public int HashTag { get; }
```

Property Value

int

IsReadOnly

Is the current object read-only?

```
public bool IsReadOnly { get; protected internal set; }
```

Property Value

bool

StartPosition

The starting position of the object in source code.

```
public Position StartPosition { get; private set; }
```

Property Value

Position

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public virtual string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public virtual string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
public virtual void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
public virtual void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseAnd(IEzrObject, RuntimeResult)

Performs the bit-wise AND operation between the current object and another.

```
public virtual void BitwiseAnd(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseLeftShift(IEzrObject, RuntimeResult)

Performs the bit-wise left-shift operation between the current object and another.

```
public virtual void BitwiseLeftShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseNegation(RuntimeResult)

Bit-wise negates the current object.

```
public virtual void BitwiseNegation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseOr(IEzrObject, RuntimeResult)

Performs the bit-wise OR operation between the current object and another.

```
public virtual void BitwiseOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseRightShift(IEzrObject, RuntimeResult)

Performs the bit-wise right-shift operation between the current object and another.

```
public virtual void BitwiseRightShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseXOr(IEzrObject, RuntimeResult)

Performs the bit-wise X-OR operation between the current object and another.

```
public virtual void BitwiseXOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public virtual void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public virtual void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public virtual void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
public virtual void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
public virtual void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public virtual void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public virtual int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public virtual void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public virtual bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public virtual void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

~EzrObject()

Destructor.

```
protected ~EzrObject()
```

GetMemberInfo<TMemberInfo, TParentType>(string)

Gets cached reflection data about a member of `TParentType`.

```
internal static TMemberInfo? GetMemberInfo<TMemberInfo, TParentType>(string name) where
TMemberInfo : MemberInfo
```

Parameters

`name` string

The name of the member.

Returns

TMemberInfo

The cached member data or the first member with the given `name`. null if not found.

Type Parameters

`TMemberInfo`

The MemberInfo type to return.

`TParentType`

The type which the member is a part of.

Remarks

If there is no cached data, it gets it by reflection and caches the result.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public virtual void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

IllegalOperation()

Creates an illegal operation error based on the current context and position.

```
protected internal EzrIllegalOperationError IllegalOperation()
```

Returns

EzrIllegalOperationError

The error.

IllegalOperation(IEzrObject, bool)

Creates an illegal operation error based on the current context and position and another object.

```
protected internal EzrIllegalOperationError IllegalOperation(IEzrObject other, bool
isRightHandSide = true)
```

Parameters

`other` IEzrObject

`isRightHandSide` bool

Returns

EzrIllegalOperationError

The error.

Interpret(Node, Context, Interpreter, RuntimeResult, bool)

Interprets an AST node in context of the current object.

```
public virtual void Interpret(Node code, Context callingContext, Interpreter interpreter,
RuntimeResult result, bool ignoreUndefinedVariable = false)
```

Parameters

`code` Node

The node to interpret.

`callingContext` Context

The context calling on this action.

`interpreter` Interpreter

The interpreter to use.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

`ignoreUndefinedVariable` bool

Should the interpretation ignore undefined variables? Useful in getting empty variable references.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public virtual void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
public virtual void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public virtual void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
public virtual void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

NewArrayConstant(IEzrObject[])

Creates a new array constant.

```
protected internal Reference NewArrayConstant(IEzrObject[] elements)
```

Parameters

`elements` IEzrObject[]

The raw array value.

Returns

Reference

The constant.

NewBooleanConstant(bool)

Creates a new boolean constant.

```
protected internal static Reference NewBooleanConstant(bool value)
```

Parameters

`value` bool

The raw boolean value.

Returns

Reference

The constant.

NewCharacterConstant(char)

Creates a new character constant.

```
protected internal Reference NewCharacterConstant(char value)
```

Parameters

`value` char

The raw character value.

Returns

Reference

The constant.

NewCharacterListConstant(string)

Creates a new character list constant.

```
protected internal Reference NewCharacterListConstant(string value)
```

Parameters

`value` string

The raw float value.

Returns

Reference

The constant.

NewDictionaryConstant(RuntimeEzrObjectDictionary)

Creates a new dictionary constant.

```
protected internal Reference NewDictionaryConstant(RuntimeEzrObjectDictionary dictionary)
```

Parameters

`dictionary` RuntimeEzrObjectDictionary

The raw dictionary value.

Returns

Reference

The constant.

NewFloatConstant(double)

Creates a new float constant.

```
protected internal Reference NewFloatConstant(double value)
```

Parameters

`value` double

The raw float value.

Returns

Reference

The constant.

NewIntegerConstant(BigInteger)

Creates a new integer constant.

```
protected internal Reference NewIntegerConstant(BigInteger value)
```

Parameters

`value` BigInteger

The raw integer value.

Returns

Reference

The constant.

NewListConstant(RuntimeEzrObjectList)

Creates a new list constant.

```
protected internal Reference NewListConstant(RuntimeEzrObjectList elements)
```

Parameters

`elements` RuntimeEzrObjectList

The raw list value.

Returns

Reference

The constant.

NewNothingConstant()

Creates a new "nothing" constant.

```
protected internal static Reference NewNothingConstant()
```

Returns

Reference

The constant.

NewStringConstant(string)

Creates a new string constant.

```
protected internal Reference NewStringConstant(string value)
```

Parameters

`value` string

The raw string value.

Returns

Reference

The constant.

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public virtual void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
public virtual void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public virtual bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
public virtual void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public virtual string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public virtual string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Update(Context, Position, Position)

Updates the context and position of the object.

```
public void Update(Context context, Position startPosition, Position endPosition)
```

Parameters

`context` Context

The new context of the object.

`startPosition` Position

The new starting position of the object.

`endPosition` Position

The new ending position of the object.

UpdateCreationContext(Context)

Changes CreationContext and the parent of Context. Be careful when you use this function.

```
public void UpdateCreationContext(Context newCreationContext)
```

Parameters

`newCreationContext` Context

The new creation context.

Class EzrRuntimeInvalidObject

Namespace: EzrSquared.Runtime.Types

Assembly: ezrSquared-lib.dll

An invalid, sort of "empty" object, to use instead of null.

```
internal class EzrRuntimeInvalidObject : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrRuntimeInvalidObject

Implements

IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject.TypeName , EzrObject.Tag , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.StrictEquals(IEzrObject, RuntimeResult) , EzrObject.ComputeHashCode(RuntimeResult) ,

EzrObject.ToString(RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrRuntimeInvalidObject()

Creates a new EzrRuntimeInvalidObject.

```
private EzrRuntimeInvalidObject()
```

Fields

s_instance

The static instance of the invalid object.

```
internal static EzrRuntimeInvalidObject s_instance
```

Field Value

EzrRuntimeInvalidObject

Interface IEzrMutableObject

Namespace: EzrSquared.Runtime.Types

Assembly: ezrSquared-lib.dll

A mutable IEzrObject.

```
public interface IEzrMutableObject : IMutable<IEzrMutableObject>, IEzrObject
```

Inherited Members

IMutable<IEzrMutableObject>.DeepCopy(RuntimeResult) , IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult)

Interface IEzrObject

Namespace: EzrSquared.Runtime.Types

Assembly: ezrSquared-lib.dll

An object in the ezr^2 language.

```
public interface IEzrObject
```

Properties

Context

The Context of the object.

```
Context Context { get; }
```

Property Value

Context

CreationContext

The context under which the object was created.

```
Context CreationContext { get; }
```

Property Value

Context

EndPosition

The ending position of the object in source code.

```
Position EndPosition { get; }
```

Property Value

Position

HashTag

The hash of Tag.

```
int HashTag { get; }
```

Property Value

int

StartPosition

The starting position of the object in source code.

```
Position StartPosition { get; }
```

Property Value

Position

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
string Tag { get; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
string TypeName { get; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseAnd(IEzrObject, RuntimeResult)

Performs the bit-wise AND operation between the current object and another.

```
void BitwiseAnd(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseLeftShift(IEzrObject, RuntimeResult)

Performs the bit-wise left-shift operation between the current object and another.

```
void BitwiseLeftShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseNegation(RuntimeResult)

Bit-wise negates the current object.

```
void BitwiseNegation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseOr(IEzrObject, RuntimeResult)

Performs the bit-wise OR operation between the current object and another.

```
void BitwiseOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseRightShift(IEzrObject, RuntimeResult)

Performs the bit-wise right-shift operation between the current object and another.

```
void BitwiseRightShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseXOr(IEzrObject, RuntimeResult)

Performs the bit-wise X-OR operation between the current object and another.

```
void BitwiseXOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
void Execute(Reference[] arguments, Interpreter interpreter, RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Interpret(Node, Context, Interpreter, RuntimeResult, bool)

Interprets an AST node in context of the current object.

```
void Interpret(Node code, Context callingContext, Interpreter interpreter, RuntimeResult
result, bool ignoreUndefinedVariable = false)
```

Parameters

`code` Node

The node to interpret.

`callingContext` Context

The context calling on this action.

`interpreter` Interpreter

The interpreter to use.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

`ignoreUndefinedVariable` bool

Should the interpretation ignore undefined variables? Useful in getting empty variable references.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Update(Context, Position, Position)

Updates the context and position of the object.

```
void Update(Context context, Position startPosition, Position endPosition)
```

Parameters

`context` Context

The new context of the object.

`startPosition` Position

The new starting position of the object.

`endPosition` Position

The new ending position of the object.

UpdateCreationContext(Context)

Changes CreationContext and the parent of Context. Be careful when you use this function.

```
void UpdateCreationContext(Context newCreationContext)
```

Parameters

`newCreationContext` Context

The new creation context.

Namespace EzrSquared.Runtime.Types.CSharp Wrappers.Builtins

Classes

EzrBuiltinFunctions

All built-in functions in ezr².

EzrBuiltinsUtility

Utility to add built-ins to contexts.

Class EzrBuiltinFunctions

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.Builtins

Assembly: ezrSquared-lib.dll

All built-in functions in ezr².

```
public static class EzrBuiltinFunctions
```

Inheritance

object ← EzrBuiltinFunctions

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Assert(SharpMethodParameters)

Assertion function to assert conditions.

```
[SharpMethodWrapper("assert", RequiredParameters = new string[] { "condition" })]
public static void Assert(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

condition	(IEzrObject) The condition to assert.

ezr² return type: EzrNothing
ezr² errors: EzrAssertionError if the condition is not met.

Clear(SharpMethodParameters)

Basic console clear function. Implements Clear().

```
[SharpMethodWrapper("clear")]
public static void Clear(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² return type: EzrNothing

Copy(SharpMethodParameters)

Creates a copy of an IEzrMutableObject. Uses DeepCopy(RuntimeResult).

```
[SharpMethodWrapper("copy", RequiredParameters = new string[] { "to_copy" })]
public static void Copy(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_copy	(IEzrMutableObject) The object to copy.

ezr² return type: IEzrMutableObject

ezr² errors: EzrUnexpectedTypeError if "to_copy" is not of the expected type.

Get(SharpMethodParameters)

Basic console input function. Implements ReadLine().

```
[SharpMethodWrapper("get", OptionalParameters = new string[] { "message" })]
public static void Get(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

message	(IEzrObject) The message to display on the console before waiting for user input.

ezr² return type: EzrString

GetContext(SharpMethodParameters)

Returns an EzrDictionary of the references (as <name, object>) contained in the Context of the given object.

```
[SharpMethodWrapper("get_context", RequiredParameters = new string[] { "to_get" })]
public static void GetContext(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_get	(IEzrObject) The object to get the context of.

ezr² return type: EzrDictionary

GetRaw(SharpMethodParameters)

Wraps the given ezr² object so that the runtime has access to its raw C# object.

```
[SharpMethodWrapper("get_raw", RequiredParameters = new string[] { "to_wrap" })]
public static void GetRaw(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_wrap	(IEzrObject) The object to wrap.

ezr² return type: IEzrObject

Hash(SharpMethodParameters)

Basic hash function. Implements ComputeHashCode(RuntimeResult).

```
[SharpMethodWrapper("hash", RequiredParameters = new string[] { "to_hash" })]
public static void Hash(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_hash	(IEzrObject) The object to hash.

ezr² return type: EzrInteger

Show(SharpMethodParameters)

Basic console print function. Implements WriteLine().

```
[SharpMethodWrapper("show", HasExtraPositionalArguments = true, OptionalParameters = new
string[] { "line_end", "separator" })]
public static void Show(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

Extra Positional Arguments	(List<T> of References) The message(s) to display on the console.
line_end	(Optional, IEzrString) Line end character(s) to use instead of \n.
separator	(Optional, IEzrString) separator to separate each message to be printed.

ezr² return type: EzrNothing
ezr² errors:

EzrMissingRequiredArgumentError	Thrown if no messages are provided.
EzrUnexpectedTypeError	Thrown if "line_end" or "separator" are not of the specified types.

ThrowError(SharpMethodParameters)

Basic error throwing function. Implements Failure(IEzrRuntimeError).

```
[SharpMethodWrapper("throw_error", RequiredParameters = new string[] { "error" })]
public static void ThrowError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

error	(IEzrRuntimeError) The error to throw.

ezr² errors:

EzrUnexpectedTypeError	if "error" is not of the specified type.
error	the given error.

TypeHashOf(SharpMethodParameters)

Gets the hash ID of the type of an object. Uses HashTag.

```
[SharpMethodWrapper("type_hash_of", RequiredParameters = new string[] { "to_check" })]
public static void TypeHashOf(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_check	(IEzrObject) The object to check the type hash of.

ezr² return type: EzrString

TypeNameOf(SharpMethodParameters)

Gets the plain text name of the type of an object. Uses TypeName.

```
[SharpMethodWrapper("type_name_of", RequiredParameters = new string[] { "to_check" })]
public static void TypeNameOf(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_check	(IEzrObject) The object to check the type-name of.

ezr² return type: EzrString

TypeOf(SharpMethodParameters)

Basic typeof-like function. Uses Tag.

```
[SharpMethodWrapper("type_of", RequiredParameters = new string[] { "to_check" })]
public static void TypeOf(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The method arguments.

Remarks

ezr² parameters:

to_check	(IEzrObject) The object to check the type of.

ezr² return type: EzrString

Class EzrBuiltinsUtility

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.Builtins

Assembly: ezrSquared-lib.dll

Utility to add built-ins to contexts.

```
public static class EzrBuiltinsUtility
```

Inheritance

object ← EzrBuiltinsUtility

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

AddBuiltinConstants(Context)

Adds all built-in constants to the given context.

```
public static void AddBuiltinConstants(Context context)
```

Parameters

`context` Context

The context to add to.

AddBuiltinFunctions(Context)

Adds all built-in functions to the given context, excluding I/O functions.

```
public static void AddBuiltinFunctions(Context context)
```

Parameters

`context` Context

The context to add to.

AddBuiltinIOFunctions(Context)

Adds all built-in I/O functions to the given context.

```
public static void AddBuiltinIOFunctions(Context context)
```

Parameters

`context` Context

The context to add to.

AddBuiltinTypes(Context)

Adds all built-in types to the given context.

```
public static void AddBuiltinTypes(Context context)
```

Parameters

`context` Context

The context to add to.

Namespace EzrSquared.Runtime.Types.CSharp Wrappers.CompatWrappers

Classes

EzrSharpCompatibilityObjectInstance

Class to automatically wrap *instances* of C# types so that they can be used in ezr².

EzrSharpCompatibilityType

Class to automatically wrap C# types so that they can be used in ezr².

EzrSharpCompatibilityWrapper<TMemberInfo>

Parent class for all automatic wrappers which wrap existing C# objects and members so that they can be used in ezr².

Class EzrSharpCompatibilityObjectInstance

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers

Assembly: ezrSquared-lib.dll

Class to automatically wrap *instances* of C# types so that they can be used in ezr².

```
public class EzrSharpCompatibilityObjectInstance : EzrSharpCompatibilityWrapper<Type>,
IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<Type> ← EzrSharpCompatibilityObjectInstance

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityWrapper<Type>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<Type>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<Type>.SharpMemberName ,
EzrSharpCompatibilityWrapper<Type>.SharpMember , EzrSharpCompatibilityWrapper<Type>.Instance ,
EzrSharpCompatibilityWrapper<Type>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<Type>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<Type>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<Type>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComputeHashCode(RuntimeResult) , EzrObject.s_memberMap ,
EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag ,
EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context ,
EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,

EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) ,
EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) ,
EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) ,
EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) ,
EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) ,
EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityObjectInstance(object, Type, Context, Position, Position)

Creates a new EzrSharpCompatibilityObjectInstance.

```
public EzrSharpCompatibilityObjectInstance(object instance, Type instanceType, Context
parentContext, Position startPosition, Position endPosition)
```

Parameters

`instance` object

The object to wrap.

`instanceType` Type

The C# object's type.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrSharpCompatibilityType

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers

Assembly: ezrSquared-lib.dll

Class to automatically wrap C# types so that they can be used in ezr^2.

```
public class EzrSharpCompatibilityType : EzrSharpCompatibilityWrapper<Type>, IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<Type> ← EzrSharpCompatibilityType

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityWrapper<Type>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<Type>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<Type>.SharpMemberName ,
EzrSharpCompatibilityWrapper<Type>.SharpMember , EzrSharpCompatibilityWrapper<Type>.Instance ,
EzrSharpCompatibilityWrapper<Type>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<Type>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<Type>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<Type>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<Type>.ComputeHashCode(RuntimeResult) , EzrObject.s_memberMap ,
EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag ,
EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context ,
EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,

EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) ,
EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) ,
EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) ,
EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) ,
EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) ,
EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityType(Type, RuntimeResult, Context, Position, Position)

Creates a new EzrSharpCompatibilityType.

```
public EzrSharpCompatibilityType(Type sharpType, RuntimeResult result, Context
parentContext, Position startPosition, Position endPosition)
```

Parameters

`sharpType` Type

The type to wrap.

`result` RuntimeResult

Runtime result for carrying any errors.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrSharpCompatibilityWrapper<TMemberInfo>

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers

Assembly: ezrSquared-lib.dll

Parent class for all automatic wrappers which wrap existing C# objects and members so that they can be used in ezr².

```
public abstract class EzrSharpCompatibilityWrapper<TMemberInfo> : EzrObject, IEzrObject
where TMemberInfo : MemberInfo
```

Type Parameters

`TMemberInfo`

The MemberInfo type for the C# member being wrapped.

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<TMemberInfo>

Implements

IEzrObject

Derived

EzrSharpCompatibilityObjectInstance, EzrSharpCompatibilityType, EzrSharpCompatibilityExecutable<TMethodBase>, EzrSharpCompatibilityField, EzrSharpCompatibilityProperty

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,

EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToString(RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityWrapper(TMemberInfo, object?, Context, Position, Position)

Creates a new EzrSharpCompatibilityWrapper<TMemberInfo>.

```
public EzrSharpCompatibilityWrapper(TMemberInfo wrappedMember, object? instance, Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`wrappedMember` TMemberInfo

Reflection info on the wrapped C# member.

`instance` object

The object which contains the wrapped member, null if static.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

AutoWrapperAttribute

The SharpAutoWrapperAttribute of the wrapped object, if defined.

```
public readonly SharpAutoWrapperAttribute? AutoWrapperAttribute
```

Field Value

SharpAutoWrapperAttribute

Instance

The object which contains the wrapped member, null if static.

```
public readonly object? Instance
```

Field Value

object

SharpMember

Reflection info for the current object being wrapped.

```
public readonly TMemberInfo SharpMember
```

Field Value

TMemberInfo

SharpMemberName

The name of the wrapped member in snake_case.

```
public readonly string SharpMemberName
```

Field Value

string

s_alphaNumericUnderscoreOnlyFilterRegex

Regex for matching non-alphanumeric + underscore characters.

```
private static readonly Regex s_alphaNumericUnderscoreOnlyFilterRegex
```

Field Value

Regex

s_caseConverterRegex

Regex for converting PascalCase/camelCase to snake_case.

```
private static readonly Regex s_caseConverterRegex
```

Field Value

Regex

Remarks

Regex explanation:

1. (?<!^)(?=[A-Z][a-z]) - Add underscore before capital letter followed by a lowercase letter, except at the start.
2. (?<=[a-z0-9])(?=[A-Z]) - Add underscore when transitioning from lowercase or number to uppercase.
3. (?<=[A-Z])(?=[A-Z][a-z]) - Add underscore between uppercase sequences followed by lowercase (e.g., "TCProtocol").

s_taskFromResultMethod

Reflection info for FromResult<TResult>(TResult)/

```
private static readonly MethodInfo s_taskFromResultMethod
```

Field Value

MethodInfo

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

CSharpToEzrObject(object?, RuntimeResult)

Converts a C# object to an ezr² object.

```
protected internal void CSharpToEzrObject(object? value, RuntimeResult result)
```

Parameters

`value` object

The C# object to convert.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

EzrObjectToCSharp(IEzrObject, Type, RuntimeResult)

Converts an ezr[2] object to a C# object.

```
protected internal object? EzrObjectToCSharp(IEzrObject value, Type targetType,
RuntimeResult result)
```

Parameters

`value` IEzrObject

The IEzrObject to convert.

`targetType` Type

The type to convert it to.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

object

The converted object.

HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult)

Converts a C# array to an ezr² array-like object.

```
protected internal void HandleCSharpArrayToEzrObject(Array value, Type valueType,
RuntimeResult result)
```

Parameters

`value` Array

The value to convert.

`valueType` Type

The type to convert from.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult)

Converts an ezr² array-like object to a C# array.

```
protected internal object? HandleEzrArrayLikeToCSharp(IEzrObject value, Type targetType,
RuntimeResult result)
```

Parameters

`value` IEzrObject

The value to convert.

`targetType` Type

The type to convert to.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

object

PascalToSnakeCase(string?)

Converts a string from PascalCase to snake_case.

```
protected internal static string? PascalToSnakeCase(string? text)
```

Parameters

`text` string

The text to convert in PascalCase.

Returns

string

The converted text in snake_case.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Namespace EzrSquared.Runtime.Types.CSharp Wrappers.CompatWrappers.ObjectMembers

Classes

EzrSharpCompatibilityField

Class to automatically wrap C# fields so that they can be used in ezr^2.

EzrSharpCompatibilityProperty

Class to automatically wrap C# properties so that they can be used in ezr^2.

Class EzrSharpCompatibilityField

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers.ObjectMembers

Assembly: ezrSquared-lib.dll

Class to automatically wrap C# fields so that they can be used in ezr^2.

```
public class EzrSharpCompatibilityField : EzrSharpCompatibilityWrapper<FieldInfo>,
IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<FieldInfo> ← EzrSharpCompatibilityField

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityWrapper<FieldInfo>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<FieldInfo>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<FieldInfo>.SharpMemberName ,
EzrSharpCompatibilityWrapper<FieldInfo>.SharpMember ,
EzrSharpCompatibilityWrapper<FieldInfo>.Instance ,
EzrSharpCompatibilityWrapper<FieldInfo>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<FieldInfo>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<FieldInfo>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<FieldInfo>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult)
,
EzrSharpCompatibilityWrapper<FieldInfo>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<FieldInfo>.ComputeHashCode(RuntimeResult) ,
EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,

EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityField(FieldInfo, object?, Context, Position, Position)

Creates a new EzrSharpCompatibilityField.

```
public EzrSharpCompatibilityField(FieldInfo sharpField, object? instance, Context
parentContext, Position startPosition, Position endPosition)
```

Parameters

`sharpField` FieldInfo

The field to wrap.

`instance` object

The object which contains the field, null if static.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Execute(Reference[], Interpreter, RuntimeResult)

If there are no arguments, accesses the field's value. If the field is not read-only and there is an arguments, sets the field's value.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrSharpCompatibilityProperty

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers.ObjectMembers

Assembly: ezrSquared-lib.dll

Class to automatically wrap C# properties so that they can be used in ezr².

```
public class EzrSharpCompatibilityProperty : EzrSharpCompatibilityWrapper<PropertyInfo>,
IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<PropertyInfo> ← EzrSharpCompatibilityProperty

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityWrapper<PropertyInfo>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<PropertyInfo>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<PropertyInfo>.SharpMemberName ,
EzrSharpCompatibilityWrapper<PropertyInfo>.SharpMember ,
EzrSharpCompatibilityWrapper<PropertyInfo>.Instance ,
EzrSharpCompatibilityWrapper<PropertyInfo>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<PropertyInfo>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<PropertyInfo>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<PropertyInfo>.ComputeHashCode(RuntimeResult) ,
EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,

EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityProperty(PropertyInfo, object?, Context, Position, Position)

Creates a new EzrSharpCompatibilityProperty.

```
public EzrSharpCompatibilityProperty(PropertyInfo sharpProperty, object? instance, Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`sharpProperty` PropertyInfo

The property to wrap.

`instance` object

The object which contains the property, null if static.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Execute(Reference[], Interpreter, RuntimeResult)

If there are no arguments, accesses the property's value. If the property is not read-only and there is an arguments, sets the property's value.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Namespace EzrSquared.Runtime.Types.CSharp Wrappers.CompatWrappers.ObjectMembers. Executables

Classes

EzrSharpCompatibilityConstructor

Class to automatically wrap C# constructors so that they can be used in ezr².

EzrSharpCompatibilityExecutable<TMethodBase>

Base class for all automatic wrappers which wrap C# executables so that they can be used in ezr².

EzrSharpCompatibilityFunction

Class to automatically wrap C# methods so that they can be used in ezr².

Class EzrSharpCompatibilityConstructor

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers.ObjectMembers.Executables

Assembly: ezrSquared-lib.dll

Class to automatically wrap C# constructors so that they can be used in ezr².

```
public class EzrSharpCompatibilityConstructor :
EzrSharpCompatibilityExecutable<ConstructorInfo>, IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<ConstructorInfo> ← EzrSharpCompatibilityExecutable<ConstructorInfo> ← EzrSharpCompatibilityConstructor

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityExecutable<ConstructorInfo>.TypeName ,
EzrSharpCompatibilityExecutable<ConstructorInfo>.Tag ,
EzrSharpCompatibilityExecutable<ConstructorInfo>.Parameters ,
EzrSharpCompatibilityExecutable<ConstructorInfo>.ParameterNames ,
EzrSharpCompatibilityExecutable<ConstructorInfo>.ArgumentsArrayToDictionary(Reference[], RuntimeResult) ,
EzrSharpCompatibilityExecutable<ConstructorInfo>.CheckAndPopulateArguments(Dictionary<string, IEzrObject>, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.SharpMemberName ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.SharpMember ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.Instance ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,

EzrSharpCompatibilityWrapper<ConstructorInfo>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<ConstructorInfo>.ComputeHashCode(RuntimeResult) ,
EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityConstructor(Type, ConstructorInfo, Context, Position, Position, bool)

Creates a new EzrSharpCompatibilityConstructor.

```
public EzrSharpCompatibilityConstructor(Type constructingType, ConstructorInfo
sharpConstructor, Context parentContext, Position startPosition, Position endPosition, bool
skipValidation = false)
```

Parameters

`constructingType` Type

This C# constructor's class.

`sharpConstructor` ConstructorInfo

The constructor to wrap.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

`skipValidation` bool

Skip method signature validation?

Fields

ConstructingType

This C# constructor's class.

```
public readonly Type ConstructingType
```

Field Value

Type

ConstructingTypeName

The name of the constructing type in snake_case.

```
public readonly string ConstructingTypeName
```

Field Value

string

Methods

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrSharpCompatibilityExecutable<TMethodBase>

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers.ObjectMembers.Executables

Assembly: ezrSquared-lib.dll

Base class for all automatic wrappers which wrap C# executables so that they can be used in ezr².

```
public abstract class EzrSharpCompatibilityExecutable<TMethodBase> :
EzrSharpCompatibilityWrapper<TMethodBase>, IEzrObject where TMethodBase : MethodBase
```

Type Parameters

`TMethodBase`

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<TMethodBase> ← EzrSharpCompatibilityExecutable<TMethodBase>

Implements

IEzrObject

Derived

EzrSharpCompatibilityConstructor, EzrSharpCompatibilityFunction

Inherited Members

EzrSharpCompatibilityWrapper<TMethodBase>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<TMethodBase>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<TMethodBase>.SharpMemberName ,
EzrSharpCompatibilityWrapper<TMethodBase>.SharpMember ,
EzrSharpCompatibilityWrapper<TMethodBase>.Instance ,
EzrSharpCompatibilityWrapper<TMethodBase>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<TMethodBase>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<TMethodBase>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<TMethodBase>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.CSharpToEzrObject(object, RuntimeResult) ,

EzrSharpCompatibilityWrapper<TMethodBase>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<TMethodBase>.ComputeHashCode(RuntimeResult) ,
EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToString(RuntimeResult) ,
EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityExecutable(TMethodBase, object?, Context, Position, Position, bool)

Creates a new EzrSharpCompatibilityExecutable<TMethodBase>.

```
public EzrSharpCompatibilityExecutable(TMethodBase sharpMethodBase, object? instance,
Context parentContext, Position startPosition, Position endPosition, bool skipValidation)
```

Parameters

`sharpMethodBase` TMethodBase

The executable to wrap.

`instance` object

The object which contains the executable, null if static.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

`skipValidation` bool

Skip method signature validation?

Fields

ParameterNames

The names of the parameters of the executable to wrap, in ezr² (snake_case) format.

```
public readonly string[] ParameterNames
```

Field Value

string[]

Parameters

Reflection information about the parameters of the executable to wrap.

```
public readonly ParameterInfo[] Parameters
```

Field Value

ParameterInfo[]

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ArgumentsArrayToDictionary(Reference[], RuntimeResult)

Converts an array of arguments from ezr[2] code to an ordered dictionary.

```
protected internal Dictionary<string, IEzrObject> ArgumentsArrayToDictionary(Reference[]
arguments, RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments.

`result` RuntimeResult

Runtime result for carrying errors.

Returns

Dictionary<string, IEzrObject>

The dictionary.

CheckAndPopulateArguments(Dictionary<string, IEzrObject>, RuntimeResult)

Converts an ordered dictionary of named arguments into an array of primitive C# objects in the order the executable expects them in.

```
protected internal object?[] CheckAndPopulateArguments(Dictionary<string, IEzrObject>
arguments, RuntimeResult result)
```

Parameters

`arguments` Dictionary<string, IEzrObject>

The arguments.

`result` RuntimeResult

Runtime result for carrying errors.

Returns

object[]

The array of objects.

Class EzrSharpCompatibilityFunction

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.CompatWrappers.ObjectMembers.Executables

Assembly: ezrSquared-lib.dll

Class to automatically wrap C# methods so that they can be used in ezr².

```
public class EzrSharpCompatibilityFunction : EzrSharpCompatibilityExecutable<MethodInfo>,
IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpCompatibilityWrapper<MethodInfo> ← EzrSharpCompatibilityExecutable<MethodInfo> ← EzrSharpCompatibilityFunction

Implements

IEzrObject

Inherited Members

EzrSharpCompatibilityExecutable<MethodInfo>.Parameters ,
EzrSharpCompatibilityExecutable<MethodInfo>.ParameterNames ,
EzrSharpCompatibilityExecutable<MethodInfo>.ArgumentsArrayToDictionary(Reference[], RuntimeResult) ,
EzrSharpCompatibilityExecutable<MethodInfo>.CheckAndPopulateArguments(Dictionary<string, IEzrObject>, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.s_taskFromResultMethod ,
EzrSharpCompatibilityWrapper<MethodInfo>.AutoWrapperAttribute ,
EzrSharpCompatibilityWrapper<MethodInfo>.SharpMemberName ,
EzrSharpCompatibilityWrapper<MethodInfo>.SharpMember ,
EzrSharpCompatibilityWrapper<MethodInfo>.Instance ,
EzrSharpCompatibilityWrapper<MethodInfo>.s_caseConverterRegex ,
EzrSharpCompatibilityWrapper<MethodInfo>.s_alphaNumericUnderscoreOnlyFilterRegex ,
EzrSharpCompatibilityWrapper<MethodInfo>.PascalToSnakeCase(string) ,
EzrSharpCompatibilityWrapper<MethodInfo>.EzrObjectToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.HandleEzrArrayLikeToCSharp(IEzrObject, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.CSharpToEzrObject(object, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.HandleCSharpArrayToEzrObject(Array, Type, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.ComparisonNotEqual(IEzrObject, RuntimeResult) ,

EzrSharpCompatibilityWrapper<MethodInfo>.EvaluateBoolean(RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.StrictEquals(IEzrObject, RuntimeResult) ,
EzrSharpCompatibilityWrapper<MethodInfo>.ComputeHashCode(RuntimeResult) ,
EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpCompatibilityFunction(MethodInfo, object?, Context, Position, Position, bool)

Class to automatically wrap C# methods so that they can be used in ezr².

```
public EzrSharpCompatibilityFunction(MethodInfo sharpFunction, object? instance, Context
parentContext, Position startPosition, Position endPosition, bool skipValidation = false)
```

Parameters

`sharpFunction` MethodInfo

The method to wrap.

`instance` object

The object which contains the method, null if static.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

`skipValidation` bool

Skip method signature validation?

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Namespace EzrSquared.Runtime.Types.CSharp Wrappers.SourceWrappers

Classes

EzrSharpSourceExecutableWrapper

Parent of all wrapper classes which wrap executable members written in C# so that they can be used in ezr².

EzrSharpSourceFunctionWrapper

A class to wrap methods written in C# so that they can be used in ezr².

EzrSharpSourceTypeWrapper

A class to wrap types written in C# so that they can be used in ezr².

Class EzrSharpSourceExecutableWrapper

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.SourceWrappers

Assembly: ezrSquared-lib.dll

Parent of all wrapper classes which wrap executable members written in C# so that they can be used in ezr².

```
public abstract class EzrSharpSourceExecutableWrapper : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpSourceExecutableWrapper

Implements

IEzrObject

Derived

EzrSharpSourceFunctionWrapper, EzrSharpSourceTypeWrapper

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.StrictEquals(IEzrObject, RuntimeResult) , EzrObject.ComputeHashCode(RuntimeResult) ,

EzrObject.ToString(RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpSourceExecutableWrapper(Context, Position, Position)

Parent of all wrapper classes which wrap executable members written in C# so that they can be used in ezr².

```
protected EzrSharpSourceExecutableWrapper(Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

HasExtraKeywordArguments

Does the executable accept extra keyword arguments?

```
public bool HasExtraKeywordArguments
```

Field Value

bool

HasExtraPositionalArguments

Does the executable accept extra positional arguments?

```
public bool HasExtraPositionalArguments
```

Field Value

bool

Parameters

The name of the executable's parameters and if they are required.

```
public (string Name, bool IsRequired)[] Parameters
```

Field Value

(string Name, bool IsRequired)[]

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

CheckAndPopulateArguments(Reference[], RuntimeResult)

Checks and populates the arguments given by the user's ezr[2] code into a dictionary.

```
protected internal (Dictionary<string, Reference> Arguments, List<Reference>?
ExtraPositionalArguments) CheckAndPopulateArguments(Reference[] arguments,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The array of arguments.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

(Dictionary<string, Reference> Arguments, List<Reference> ExtraPositionalArguments)

The dictionary of all the arguments.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

IndexToFlag(int)

Converts an index to a power of two, so that the index can be used like an enum flag.

```
protected internal static int IndexToFlag(int index)
```

Parameters

`index` int

The index to be converted.

Returns

int

The power of two.

Remarks

This function is used to check if all arguments have been provided to EzrSharpSourceExecutableWrapper and EzrRuntimeExecutable types.
The indices of the given arguments are converted to powers of two and bitwise-ored together, then bitwise-anded with the index of a defined parameter which is also converted to a power of two.
Finally, if the result is the same as the power of two of the parameter's index, this tells the interpreter that the particular required parameter has been provided.

PascalCaseToLowerCasePlainText(string)

Converts a string from PascalCase to lowecase plain text, seperated by spaces.

```
protected internal static string PascalCaseToLowerCasePlainText(string text)
```

Parameters

`text` string

The text to convert in PascalCase.

Returns

string

The converted text in lowecase plain text.

Class EzrSharpSourceFunctionWrapper

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.SourceWrappers

Assembly: ezrSquared-lib.dll

A class to wrap methods written in C# so that they can be used in ezr².

```
public class EzrSharpSourceFunctionWrapper : EzrSharpSourceExecutableWrapper, IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpSourceExecutableWrapper ← EzrSharpSourceFunctionWrapper

Implements

IEzrObject

Inherited Members

EzrSharpSourceExecutableWrapper.Parameters ,
EzrSharpSourceExecutableWrapper.HasExtraKeywordArguments ,
EzrSharpSourceExecutableWrapper.HasExtraPositionalArguments ,
EzrSharpSourceExecutableWrapper.PascalCaseToLowerCasePlainText(string) ,
EzrSharpSourceExecutableWrapper.IndexToFlag(int) ,
EzrSharpSourceExecutableWrapper.CheckAndPopulateArguments(Reference[], RuntimeResult) ,
EzrSharpSourceExecutableWrapper.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpSourceExecutableWrapper.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpSourceExecutableWrapper.EvaluateBoolean(RuntimeResult) , EzrObject.s_memberMap ,
EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag ,
EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context ,
EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpSourceFunctionWrapper(Action<SharpMethodParameters>, Context, Position, Position)

Creates a new EzrSharpSourceFunctionWrapper from a function.

```
public EzrSharpSourceFunctionWrapper(Action<SharpMethodParameters> function, Context
parentContext, Position startPosition, Position endPosition)
```

Parameters

`function` Action<SharpMethodParameters>

The method to wrap.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

EzrSharpSourceFunctionWrapper(MethodInfo, Context, Position, Position)

Creates a new EzrSharpSourceFunctionWrapper from a function's MethodInfo.

```
public EzrSharpSourceFunctionWrapper(MethodInfo function, Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`function` MethodInfo

The method to wrap.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

SharpFunction

The wrapped function.

```
public readonly Action<SharpMethodParameters> SharpFunction
```

Field Value

Action<SharpMethodParameters>

SharpFunctionName

The name of the function, in snake_case.

```
public readonly string SharpFunctionName
```

Field Value

string

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

AddParameters(SharpMethodWrapperAttribute)

Adds the parameters of the function to be wrapped to the object.

```
private void AddParameters(SharpMethodWrapperAttribute attribute)
```

Parameters

`attribute` SharpMethodWrapperAttribute

The attribute of the function containing the details for optional, required and extra keyword parameters.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

GetFunctionInfo(MethodInfo)

Gets the function's ezr^2 (snake_case) name and wrapper attribute.

```
private (string, SharpMethodWrapperAttribute) GetFunctionInfo(MethodInfo function)
```

Parameters

`function` MethodInfo

The function's MethodInfo.

Returns

(string, SharpMethodWrapperAttribute)

The function's ezr^2 name and wrapper attribute

Exceptions

ArgumentException

Thrown if the SharpMethodWrapperAttribute attribute was not found in the function or if the name given in the function's SharpMethodWrapperAttribute is empty.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrSharpSourceTypeWrapper

Namespace: EzrSquared.Runtime.Types.CSharpWrappers.SourceWrappers

Assembly: ezrSquared-lib.dll

A class to wrap types written in C# so that they can be used in ezr².

```
public class EzrSharpSourceTypeWrapper : EzrSharpSourceExecutableWrapper, IEzrObject
```

Inheritance

object ← EzrObject ← EzrSharpSourceExecutableWrapper ← EzrSharpSourceTypeWrapper

Implements

IEzrObject

Inherited Members

EzrSharpSourceExecutableWrapper.Parameters ,
EzrSharpSourceExecutableWrapper.HasExtraKeywordArguments ,
EzrSharpSourceExecutableWrapper.HasExtraPositionalArguments ,
EzrSharpSourceExecutableWrapper.PascalCaseToLowerCasePlainText(string) ,
EzrSharpSourceExecutableWrapper.IndexToFlag(int) ,
EzrSharpSourceExecutableWrapper.CheckAndPopulateArguments(Reference[], RuntimeResult) ,
EzrSharpSourceExecutableWrapper.ComparisonEqual(IEzrObject, RuntimeResult) ,
EzrSharpSourceExecutableWrapper.ComparisonNotEqual(IEzrObject, RuntimeResult) ,
EzrSharpSourceExecutableWrapper.EvaluateBoolean(RuntimeResult) , EzrObject.s_memberMap ,
EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag ,
EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context ,
EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrSharpSourceTypeWrapper(Type, Context, Position, Position)

Creates a new EzrSharpSourceTypeWrapper.

```
public EzrSharpSourceTypeWrapper(Type type, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`type` Type

The type to wrap.

`parentContext` Context

The context in which this object was created.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Exceptions

ArgumentException

Thrown if the type to be wrapped is generic or if the SharpTypeWrapperAttribute was not found in the type.

AmbiguousMatchException

Thrown if there are multiple methods/properties/fields with the same name.

Fields

Constructor

Reflection information about the type's constructor.

```
public readonly ConstructorInfo Constructor
```

Field Value

ConstructorInfo

SharpType

The wrapped type.

```
public readonly Type SharpType
```

Field Value

Type

SharpTypeName

The name of the type, in snake_case.

```
public readonly string SharpTypeName
```

Field Value

string

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Namespace EzrSquared.Runtime.Types.Collections

Classes

EzrArray

The immutable, array type object.

EzrDictionary

The mutable dictionary type object.

EzrList

The mutable, list type object.

Interfaces

IEzrDictionary

Interface for a keyed collection of ezr² objects.

IEzrEnumerable

A read-only collection of IEzrObjects.

IEzrIndexedCollection

Interface for an indexed collection of ezr² objects.

Class EzrArray

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

The immutable, array type object.

```
public class EzrArray : EzrObject, IEzrIndexedCollection, IEzrEnumerable, IEzrObject,
IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable
```

Inheritance

object ← EzrObject ← EzrArray

Implements

IEzrIndexedCollection, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,

object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrArray(IEzrObject[], Context, Position, Position)

Creates a new EzrArray.

```
public EzrArray(IEzrObject[] elements, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`elements` IEzrObject[]

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The array value.

```
public readonly IEzrObject[] Value
```

Field Value

IEzrObject[]

Properties

Count

Gets the number of elements in the collection.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

The number of elements in the collection.

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Creates a copy of the current object with the other object appended. If the other object is an array, appends its elements.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

At(int)

Gets the object at the specified index.

```
public IEzrObject At(int index)
```

Parameters

`index` int

The index.

Returns

IEzrObject

The object at the index.

Compare(IEzrIndexedCollection, RuntimeResult)

Compares the current array with another collection.

```
private bool Compare(IEzrIndexedCollection other, RuntimeResult result)
```

Parameters

`other` IEzrIndexedCollection

The other collection.

`result` RuntimeResult

Runtime result to carray any errors.

Returns

bool

The result of the comparison.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Gets the object(s) at the specified index/indices.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Gets the object(s) at the specified index/indices.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Contains(IEzrObject, RuntimeResult)

Checks if the specified object is contained in the current array.

```
private bool Contains(IEzrObject ezrObject, RuntimeResult result)
```

Parameters

`ezrObject` IEzrObject

The object to check.

`result` RuntimeResult

Runtime result to carray any errors.

Returns

bool

The result of the check.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "divide" means "duplicate/decrease the current value X times".

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<IEzrObject> GetEnumerator()
```

Returns

IEnumerator<IEzrObject>

An enumerator that can be used to iterate through the collection.

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
public IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr^2 runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "multiply" means "duplicate/decrease the current value X times".

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Creates a copy of the current object with the objects at the specified index/indices removed.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Class EzrDictionary

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

The mutable dictionary type object.

```
public class EzrDictionary : EzrObject, IEzrMutableObject, IMutable<IEzrMutableObject>,
IEzrDictionary, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>,
IEnumerable<IEzrObject>, IEnumerable
```

Inheritance

object ← EzrObject ← EzrDictionary

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrDictionary, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,

EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrDictionary(RuntimeEzrObjectDictionary, Context, Position, Position)

Creates a new EzrDictionary.

```
public EzrDictionary(RuntimeEzrObjectDictionary value, Context parentContext, Position
startPosition, Position endPosition)
```

Parameters

`value` RuntimeEzrObjectDictionary

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The dictionary value.

```
public readonly RuntimeEzrObjectDictionary Value
```

Field Value

RuntimeEzrObjectDictionary

Properties

Count

Gets the number of elements in the collection.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

The number of elements in the collection.

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Appends the current object with the other object and its key. If the other object is a dictionary, merges it with the current object.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

At(IEzrObject, RuntimeResult)

Gets the object at the specified key.

```
public IEzrObject At(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key.

`result` RuntimeResult

Runtime result for operations on the `key`.

Returns

IEzrObject

The object at the key.

Exceptions

KeyNotFoundException

Thrown if the key could not be hashed or was not found.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Gets the object at the specified key.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Gets the object at the specified key.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

DictionaryExists(SharpMethodParameters)

Basic key checking function. Implements HasKey(IEzrObject, RuntimeResult).

```
[SharpMethodWrapper("has_key", RequiredParameters = new string[] { "key" })]
private void DictionaryExists(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

Remarks

ezr² parameters:

key	(IEzrObject) The key to check for.

ezr² return type: EzrBoolean

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "divide" means "decrease the number of pairs in the dictionary X times".

~EzrDictionary()

Destructor.

```
protected ~EzrDictionary()
```

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<IEzrObject> GetEnumerator()
```

Returns

IEnumerator<IEzrObject>

An enumerator that can be used to iterate through the collection.

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
public IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr^2 runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Removes the object at the specified key.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

TryAt(IEzrObject, RuntimeResult)

Tries to get the object at the specified key.

```
public IEzrObject? TryAt(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key.

`result` RuntimeResult

Runtime result for operations on the `key`.

Returns

IEzrObject

The object at the key or null if not found or something went wrong.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Class EzrList

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

The mutable, list type object.

```
public class EzrList : EzrObject, IEzrMutableObject, IMutable<IEzrMutableObject>,
IEzrIndexedCollection, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>,
IEnumerable<IEzrObject>, IEnumerable
```

Inheritance

object ← EzrObject ← EzrList

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrIndexedCollection, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,

object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrList(RuntimeEzrObjectList, Context, Position, Position)

```
public EzrList(RuntimeEzrObjectList elements, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`elements` RuntimeEzrObjectList

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The list value.

```
public readonly RuntimeEzrObjectList Value
```

Field Value

RuntimeEzrObjectList

Properties

Count

Gets the number of elements in the collection.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

The number of elements in the collection.

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Appends the current object with the other object. If the other object is a list, appends its elements.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

At(int)

Gets the object at the specified index.

```
public IEzrObject At(int index)
```

Parameters

`index` int

The index.

Returns

IEzrObject

The object at the index.

Compare(IEzrIndexedCollection, RuntimeResult)

Compares the current list with another collection.

```
private bool Compare(IEzrIndexedCollection other, RuntimeResult result)
```

Parameters

`other` IEzrIndexedCollection

The other collection.

`result` RuntimeResult

Runtime result to carray any errors.

Returns

bool

The result of the comparison.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Gets the object(s) at the specified index/indices.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Gets the object(s) at the specified index/indices.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Contains(IEzrObject, RuntimeResult)

Checks if the specified object is contained in the current list.

```
private bool Contains(IEzrObject ezrObject, RuntimeResult result)
```

Parameters

`ezrObject` IEzrObject

The object to check.

`result` RuntimeResult

Runtime result to carray any errors.

Returns

bool

The result of the check.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "divide" means "duplicate/decrease the current value X times".

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

~EzrList()

Destructor.

```
protected ~EzrList()
```

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<IEzrObject> GetEnumerator()
```

Returns

IEnumerator<IEzrObject>

An enumerator that can be used to iterate through the collection.

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
public IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr² runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "multiply" means "duplicate/decrease the current value X times".

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Removes the object(s) at the specified index/indices.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Interface IEzrDictionary

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

Interface for a keyed collection of ezr² objects.

```
public interface IEzrDictionary : IEzrEnumerable, IEzrObject,
IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable
```

Inherited Members

IEzrEnumerable.GetEnumerator(RuntimeResult) , IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult) , IReadOnlyCollection<IEzrObject>.Count , IEnumerable<IEzrObject>.GetEnumerator()

Methods

At(IEzrObject, RuntimeResult)

Gets the object at the specified key.

```
IEzrObject At(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key.

`result` RuntimeResult

Runtime result for operations on the `key`.

Returns

IEzrObject

The object at the key.

TryAt(IEzrObject, RuntimeResult)

Tries to get the object at the specified key.

```
IEzrObject? TryAt(IEzrObject key, RuntimeResult result)
```

Parameters

`key` IEzrObject

The key.

`result` RuntimeResult

Runtime result for operations on the `key`.

Returns

IEzrObject

The object at the key or null if not found or something went wrong.

Interface IEzrEnumerable

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

A read-only collection of IEzrObjects.

```
public interface IEzrEnumerable : IEzrObject, IEnumerable<IEzrObject>, IEnumerable
```

Inherited Members

IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult) , IEnumerable<IEzrObject>.GetEnumerator()

Methods

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr² runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

Interface IEzrIndexedCollection

Namespace: EzrSquared.Runtime.Types.Collections

Assembly: ezrSquared-lib.dll

Interface for an indexed collection of ezr² objects.

```
public interface IEzrIndexedCollection : IEzrEnumerable, IEzrObject,
IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable
```

Inherited Members

IEzrEnumerable.GetEnumerator(RuntimeResult) , IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult) , IReadOnlyCollection<IEzrObject>.Count , IEnumerable<IEzrObject>.GetEnumerator()

Methods

At(int)

Gets the object at the specified index.

```
IEzrObject At(int index)
```

Parameters

`index` int

The index.

Returns

IEzrObject

The object at the index.

Namespace EzrSquared.Runtime.Types.Core

Classes

EzrBoolean

The boolean type object.

EzrConstants

Static constants for objects that won't change.

EzrNothing

The null-equivalent type.

Class EzrBoolean

Namespace: EzrSquared.Runtime.Types.Core

Assembly: ezrSquared-lib.dll

The boolean type object.

```
public class EzrBoolean : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrBoolean

Implements

IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) ,
EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) ,
EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) ,
EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) ,
EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) ,

EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrBoolean(bool, Context, Position, Position)

The boolean type object.

```
public EzrBoolean(bool value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` bool

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The boolean value of the object.

```
public readonly bool Value
```

Field Value

bool

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public override void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrConstants

Namespace: EzrSquared.Runtime.Types.Core

Assembly: ezrSquared-lib.dll

Static constants for objects that won't change.

```
public static class EzrConstants
```

Inheritance

object ← EzrConstants

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

False

Constant boolean false value.

```
public static readonly EzrBoolean False
```

Field Value

EzrBoolean

Nothing

Constant "nothing" value.

```
public static readonly EzrNothing Nothing
```

Field Value

EzrNothing

True

Constant boolean true value.

```
public static readonly EzrBoolean True
```

Field Value

EzrBoolean

Class EzrNothing

Namespace: EzrSquared.Runtime.Types.Core

Assembly: ezrSquared-lib.dll

The null-equivalent type.

```
public class EzrNothing : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrNothing

Implements

IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) ,
EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition ,
EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext ,
EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) ,
EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) ,
EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) ,
EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) ,
EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) ,
EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,
EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,
EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,
EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) ,
EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) ,
EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) ,
EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) ,
EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) ,

EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrNothing(Context, Position, Position)

The null-equivalent type.

```
public EzrNothing(Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Namespace EzrSquared.Runtime.Types.Core.Errors

Classes

EzrAssertionError
: Error type for when an assertion fails.

EzrIllegalOperationError
: Error type for when a generic illegal operation occurs.

EzrKeyNotFoundError
: Error type for when a key-not-found error occurs.

EzrMathError
: Error type for when a generic mathematical error occurs.

EzrMissingRequiredArgumentError
: Error type for when an argument is missing.

EzrPrivateMemberOperationError
: Error type for when an illegal operation is performed on a private member.

EzrRuntimeError
: Implementation of IEzrRuntimeError with some utility methods.

EzrUndefinedValueError
: Error type for when an undefined value is encountered.

EzrUnexpectedArgumentError
: Error type for when an unexpected argument is encountered.

EzrUnexpectedTypeError
: Error type for when an unexpected type is encountered.

EzrUnsupportedWrappingError
: Error type for when wrapping an unsupported type is attempted.

EzrValueOutOfRangeError
: Error type for when a value is out of a range.

EzrWrapperExecutionError
: Error type for when an error related to the execution of C# wrappers occurs.

Interfaces

IEzrRuntimeError

Base of all error types.

Class EzrAssertionError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an assertion fails.

```
[SharpTypeWrapper("assertion_error")]
public class EzrAssertionError : EzrRuntimeError, IEzrRuntimeError, IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrAssertionError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,

EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.EvaluateBoolean(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrAssertionError(Context, Position, Position)

Error type for when an assertion fails.

```
public EzrAssertionError(Context context, Position startPosition, Position endPosition)
```

Parameters

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

EzrAssertionError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper]
public EzrAssertionError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrIllegalOperationError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when a generic illegal operation occurs.

```
[SharpTypeWrapper("illegal_operation_error")]
public class EzrIllegalOperationError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrIllegalOperationError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrIllegalOperationError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrIllegalOperationError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrIllegalOperationError(string, Context, Position, Position)

Error type for when a generic illegal operation occurs.

```
public EzrIllegalOperationError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrKeyNotFoundError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when a key-not-found error occurs.

```
[SharpTypeWrapper("key_not_found_error")]
public class EzrKeyNotFoundError : EzrRuntimeError, IEzrRuntimeError, IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrKeyNotFoundError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,

EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.EvaluateBoolean(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrKeyNotFoundError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrKeyNotFoundError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrKeyNotFoundError(string, Context, Position, Position)

Error type for when a key-not-found error occurs.

```
public EzrKeyNotFoundError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrMathError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when a generic mathematical error occurs.

```
[SharpTypeWrapper("math_error")]
public class EzrMathError : EzrRuntimeError, IEzrRuntimeError, IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrMathError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) ,

EzrObject.HasValueContained(IEzrObject, RuntimeResult) ,
EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,
EzrObject.EvaluateBoolean(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrMathError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "title", "details" })]
public EzrMathError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrMathError(string, string, Context, Position, Position)

Error type for when a generic mathematical error occurs.

```
public EzrMathError(string title, string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`title` string

The title of the error.

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrMissingRequiredArgumentError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an argument is missing.

```
[SharpTypeWrapper("missing_required_argument_error")]
public class EzrMissingRequiredArgumentError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrMissingRequiredArgumentError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrMissingRequiredArgumentError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrMissingRequiredArgumentError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrMissingRequiredArgumentError(string, Context, Position, Position)

Error type for when an argument is missing.

```
public EzrMissingRequiredArgumentError(string details, Context context, Position
startPosition, Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrPrivateMemberOperationError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an illegal operation is performed on a private member.

```
[SharpTypeWrapper("private_member_operation_error")]
public class EzrPrivateMemberOperationError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrPrivateMemberOperationError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrPrivateMemberOperationError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrPrivateMemberOperationError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrPrivateMemberOperationError(string, Context, Position, Position)

Error type for when an illegal operation is performed on a private member.

```
public EzrPrivateMemberOperationError(string details, Context context, Position
startPosition, Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrRuntimeError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Implementation of IEzrRuntimeError with some utility methods.

```
[SharpTypeWrapper("runtime_error")]
public class EzrRuntimeError : EzrObject, IEzrRuntimeError, IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Derived

EzrAssertionError, EzrIllegalOperationError, EzrKeyNotFoundError, EzrMathError, EzrMissingRequiredArgumentError, EzrPrivateMemberOperationError, EzrUndefinedValueError, EzrUnexpectedArgumentError, EzrUnexpectedTypeError, EzrUnsupportedWrappingError, EzrValueOutOfRangeError, EzrWrapperExecutionError

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) ,

EzrObject.EvaluateBoolean(RuntimeResult) ,
EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) ,
EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() ,
EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) ,
EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) ,
EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) ,
EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) ,
EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() ,
EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) ,
object.GetHashCode() , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrRuntimeError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "title", "details" })]
public EzrRuntimeError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrRuntimeError(string, string, Context, Position, Position)

Creates a new runtime error object.

```
public EzrRuntimeError(string title, string details, Context context, Position
startPosition, Position endPosition)
```

Parameters

`title` string

The title of the error.

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Details

The reason why the IEzrError occurred.

```
public string Details { get; }
```

Property Value

string

ErrorContext

The context where the error occurred.

```
public Context ErrorContext { get; }
```

Property Value

Context

ErrorEndPosition

The ending Position of the IEzrError.

```
public Position ErrorEndPosition { get; }
```

Property Value

Position

ErrorStartPosition

The starting Position of the IEzrError.

```
public Position ErrorStartPosition { get; }
```

Property Value

Position

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

Title

The name of the IEzrError.

```
public string Title { get; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

GenerateTraceback(int)

Generates the trace back to the error.

```
protected internal string GenerateTraceback(int correctedLineNumber)
```

Parameters

`correctedLineNumber` int

Returns

string

The trace.

GetStringArgument(string, IEzrObject, Context, RuntimeResult)

Converts the given argument to a string.

```
protected internal static string GetStringArgument(string argumentName, IEzrObject
ezrObject, Context context, RuntimeResult result)
```

Parameters

`argumentName` string

The name of the argument.

`ezrObject` IEzrObject

The argument object.

`context` Context

The context of the argument.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The string, or Empty if failed.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrUndefinedValueError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an undefined value is encountered.

```
[SharpTypeWrapper("undefined_value_error")]
public class EzrUndefinedValueError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrUndefinedValueError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrUndefinedValueError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrUndefinedValueError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrUndefinedValueError(string, Context, Position, Position)

Error type for when an undefined value is encountered.

```
public EzrUndefinedValueError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrUnexpectedArgumentError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an unexpected argument is encountered.

```
[SharpTypeWrapper("unexpected_argument_error")]
public class EzrUnexpectedArgumentError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrUnexpectedArgumentError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrUnexpectedArgumentError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrUnexpectedArgumentError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrUnexpectedArgumentError(string, Context, Position, Position)

Error type for when an unexpected argument is encountered.

```
public EzrUnexpectedArgumentError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrUnexpectedTypeError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an unexpected type is encountered.

```
[SharpTypeWrapper("unexpected_type_error")]
public class EzrUnexpectedTypeError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrUnexpectedTypeError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrUnexpectedTypeError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrUnexpectedTypeError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrUnexpectedTypeError(string, Context, Position, Position)

Error type for when an unexpected type is encountered.

```
public EzrUnexpectedTypeError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrUnsupportedWrappingError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when wrapping an unsupported type is attempted.

```
[SharpTypeWrapper("unsupported_wrapping_error")]
public class EzrUnsupportedWrappingError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrUnsupportedWrappingError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrUnsupportedWrappingError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrUnsupportedWrappingError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrUnsupportedWrappingError(string, Context, Position, Position)

Error type for when wrapping an unsupported type is attempted.

```
public EzrUnsupportedWrappingError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrValueOutOfRangeError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when a value is out of a range.

```
[SharpTypeWrapper("value_out_of_range_error")]
public class EzrValueOutOfRangeError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrValueOutOfRangeError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrValueOutOfRangeError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrValueOutOfRangeError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrValueOutOfRangeError(string, Context, Position, Position)

Error type for when a value is out of a range.

```
public EzrValueOutOfRangeError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Class EzrWrapperExecutionError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Error type for when an error related to the execution of C# wrappers occurs.

```
[SharpTypeWrapper("wrapper_execution_error")]
public class EzrWrapperExecutionError : EzrRuntimeError, IEzrRuntimeError,
IEzrObject, IEzrError
```

Inheritance

object ← EzrObject ← EzrRuntimeError ← EzrWrapperExecutionError

Implements

IEzrRuntimeError, IEzrObject, IEzrError

Inherited Members

EzrRuntimeError.Title , EzrRuntimeError.Details , EzrRuntimeError.ErrorContext , EzrRuntimeError.ErrorStartPosition , EzrRuntimeError.ErrorEndPosition , EzrRuntimeError.GetStringArgument(string, IEzrObject, Context, RuntimeResult) , EzrRuntimeError.GenerateTraceback(int) , EzrRuntimeError.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeError.ComputeHashCode(RuntimeResult) , EzrRuntimeError.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeError.ToString(RuntimeResult) , EzrRuntimeError.ToPureString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) ,

EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.EvaluateBoolean(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrWrapperExecutionError(SharpMethodParameters)

Wrapper constructor for creating the error object.

```
[SharpMethodWrapper(RequiredParameters = new string[] { "details" })]
public EzrWrapperExecutionError(SharpMethodParameters arguments)
```

Parameters

`arguments` SharpMethodParameters

The constructor arguments.

EzrWrapperExecutionError(string, Context, Position, Position)

Error type for when an error related to the execution of C# wrappers occurs.

```
public EzrWrapperExecutionError(string details, Context context, Position startPosition,
Position endPosition)
```

Parameters

`details` string

Details on why the error happened.

`context` Context

The context in which the error occurred.

`startPosition` Position

The starting position of the error.

`endPosition` Position

The ending position of the error.

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Interface IEzrRuntimeError

Namespace: EzrSquared.Runtime.Types.Core.Errors

Assembly: ezrSquared-lib.dll

Base of all error types.

```
public interface IEzrRuntimeError : IEzrObject, IEzrError
```

Inherited Members

IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult) , IEzrError.Title , IEzrError.Details , IEzrError.ErrorStartPosition , IEzrError.ErrorEndPosition , IEzrError.SourceWithUnderline(Position, Position)

Properties

ErrorContext

The context where the error occurred.

```
Context ErrorContext { get; }
```

Property Value

Context

Namespace EzrSquared.Runtime.Types.Core.Numerics

Classes

EzrFloat
: The float (actually double) type object.

EzrInteger
: The integer type object.

Enums

EzrInteger.Size
: Represents the size of the integer value stored in an EzrInteger object.

Class EzrFloat

Namespace: EzrSquared.Runtime.Types.Core.Numerics

Assembly: ezrSquared-lib.dll

The float (actually double) type object.

```
public class EzrFloat : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrFloat

Implements

IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrFloat(double, Context, Position, Position)

The float (actually double) type object.

```
public EzrFloat(double value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` double

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The double value.

```
public readonly double Value
```

Field Value

double

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
public override void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public override void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
public override void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
public override void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
public override void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrInteger

Namespace: EzrSquared.Runtime.Types.Core.Numerics

Assembly: ezrSquared-lib.dll

The integer type object.

```
public class EzrInteger : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrInteger

Implements

IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrInteger(BigInteger, Context, Position, Position)

The integer type object.

```
public EzrInteger(BigInteger value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` BigInteger

The value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The integer value.

```
public readonly BigInteger Value
```

Field Value

BigInteger

ValueSize

The size of the Value.

```
public readonly EzrInteger.Size ValueSize
```

Field Value

EzrInteger.Size

_doubleRepresentation

The double representation of Value. May be null.

```
private double? _doubleRepresentation
```

Field Value

double?

_intRepresentation

The int representation of Value. May be null.

```
private int? _intRepresentation
```

Field Value

int?

_longRepresentation

The long representation of Value. May be null.

```
private long? _longRepresentation
```

Field Value

long?

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
public override void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseAnd(IEzrObject, RuntimeResult)

Performs the bit-wise AND operation between the current object and another.

```
public override void BitwiseAnd(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseLeftShift(IEzrObject, RuntimeResult)

Performs the bit-wise left-shift operation between the current object and another.

```
public override void BitwiseLeftShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseNegation(RuntimeResult)

Bit-wise negates the current object.

```
public override void BitwiseNegation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseOr(IEzrObject, RuntimeResult)

Performs the bit-wise OR operation between the current object and another.

```
public override void BitwiseOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseRightShift(IEzrObject, RuntimeResult)

Performs the bit-wise right-shift operation between the current object and another.

```
public override void BitwiseRightShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseXOr(IEzrObject, RuntimeResult)

Performs the bit-wise X-OR operation between the current object and another.

```
public override void BitwiseXOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

GetDoubleRepresentation()

Gets a double representation of the current object.

```
public double GetDoubleRepresentation()
```

Returns

double

The double representation.

GetIntRepresentation()

Gets an int representation of the current object. Use only if you know the value of the current object will fit in an int value.

```
public int GetIntRepresentation()
```

Returns

int

The int representation.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public override void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
public override void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
public override void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
public override void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

TryGetIntRepresentation(out int)

Tries to get an int representation of the current object.

```
public bool TryGetIntRepresentation(out int value)
```

Parameters

`value` int

The int representation.

Returns

bool

true if successful, false otherwise.

TryGetLongRepresentation(out long)

Tries to get a long representation of the current object.

```
public bool TryGetLongRepresentation(out long value)
```

Parameters

`value` long

The long representation.

Returns

bool

true if successful, false otherwise.

Enum EzrInteger.Size

Namespace: EzrSquared.Runtime.Types.Core.Numerics

Assembly: ezrSquared-lib.dll

Represents the size of the integer value stored in an EzrInteger object.

```
public enum EzrInteger.Size
```

Fields

`Big = 2`

Large, unknown. See also BigInteger.

`Int32 = 0`

Same as int.

`Int64 = 1`

Same as long.

Namespace EzrSquared.Runtime.Types.Core.Text

Classes

EzrCharacter

The character type object.

EzrCharacterList

The StringBuilder type object.

EzrString

The string type object.

Interfaces

IEzrString

Interface for getting the value of string-like ezr² objects as C# strings.

Class EzrCharacter

Namespace: EzrSquared.Runtime.Types.Core.Text

Assembly: ezrSquared-lib.dll

The character type object.

```
public class EzrCharacter : EzrObject, IEzrString, IEzrObject
```

Inheritance

object ← EzrObject ← EzrCharacter

Implements

IEzrString, IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrCharacter(char, Context, Position, Position)

The character type object.

```
public EzrCharacter(char value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` char

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The character value.

```
public readonly char Value
```

Field Value

char

Properties

StringValue

The string value.

```
public string StringValue { get; }
```

Property Value

string

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
public override void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public override void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
public override void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
public override void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
public override void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrCharacterList

Namespace: EzrSquared.Runtime.Types.Core.Text

Assembly: ezrSquared-lib.dll

The StringBuilder type object.

```
public class EzrCharacterList : EzrObject, IEzrMutableObject, IMutable<IEzrMutableObject>,
IEzrString, IEzrIndexedCollection, IEzrEnumerable, IEzrObject,
IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable
```

Inheritance

object ← EzrObject ← EzrCharacterList

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrString, IEzrIndexedCollection, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrCharacterList(string, Context, Position, Position)

Creates a new EzrCharacterList.

```
public EzrCharacterList(string value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` string

The base *string* value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The StringBuilder value.

```
public readonly StringBuilder Value
```

Field Value

StringBuilder

Properties

Count

Gets the number of elements in the collection.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

The number of elements in the collection.

StringValue

Calls ToString() and returns the StringBuilder value converted to a string.

```
public string StringValue { get; }
```

Property Value

string

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Appends the string representation of the other object to the current object.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

At(int)

Gets the object at the specified index.

```
public IEzrObject At(int index)
```

Parameters

`index` int

The index.

Returns

IEzrObject

The object at the index.

Compare(IEzrIndexedCollection)

Compares the current character list with another collection.

```
private bool Compare(IEzrIndexedCollection other)
```

Parameters

`other` IEzrIndexedCollection

The other collection.

Returns

bool

The result of the comparison.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Gets the character(s) at the specified index/indices OR compares the current object to other *stringlike* objects, checks if this is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Gets the character(s) at the specified index/indices OR compares the current object to other *stringlike* objects, checks if this is less than or equal to the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "divide" means "duplicate/decrease the current value X times".

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<IEzrObject> GetEnumerator()
```

Returns

IEnumerator<IEzrObject>

An enumerator that can be used to iterate through the collection.

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
public IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr² runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "multiply" means "duplicate/decrease the current value X times".

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Removes the character(s) at the specified index/indices.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Class EzrString

Namespace: EzrSquared.Runtime.Types.Core.Text

Assembly: ezrSquared-lib.dll

The string type object.

```
public class EzrString : EzrObject, IEzrString, IEzrIndexedCollection, IEzrEnumerable,
IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable
```

Inheritance

object ← EzrObject ← EzrString

Implements

IEzrString, IEzrIndexedCollection, IEzrEnumerable, IEzrObject, IReadOnlyCollection<IEzrObject>, IEnumerable<IEzrObject>, IEnumerable

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrString(string, Context, Position, Position)

Creates a new EzrString.

```
public EzrString(string value, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`value` string

The base value.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

Value

The string value.

```
public readonly string Value
```

Field Value

string

Properties

Count

Gets the number of elements in the collection.

```
[SharpAutoWrapper("length", false, false)]
public int Count { get; }
```

Property Value

int

The number of elements in the collection.

StringValue

The string value.

```
public string StringValue { get; }
```

Property Value

string

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Creates a copy of the current object with the string representation of the other object appended.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

At(int)

Gets the object at the specified index.

```
public IEzrObject At(int index)
```

Parameters

`index` int

The index.

Returns

IEzrObject

The object at the index.

Compare(IEzrIndexedCollection)

Compares the current string with another collection.

```
private bool Compare(IEzrIndexedCollection other)
```

Parameters

`other` IEzrIndexedCollection

The other collection.

Returns

bool

The result of the comparison.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Gets the character(s) at the specified index/indices OR compares the current object to other *stringlike* objects, checks if this is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Gets the character(s) at the specified index/indices OR compares the current object to other *stringlike* objects, checks if this is less than the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "divide" means "duplicate/decrease the current value X times".

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<IEzrObject> GetEnumerator()
```

Returns

IEnumerator<IEzrObject>

An enumerator that can be used to iterate through the collection.

GetEnumerator(RuntimeResult)

Similar to GetEnumerator().

```
public IEnumerator<IEzrObject> GetEnumerator(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying errors.

Returns

IEnumerator<IEzrObject>

The IEnumerator<T>.

Remarks

Use this when exposing the enumerator to the ezr² runtime. For example, this is used by EzrDictionary to copy read-only keys so that they can't be edited at runtime.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Remarks

Here, "multiply" means "duplicate/decrease the current value X times".

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Creates a copy of the current object with the character(s) at the specified index/indices removed.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Explicit Interface Implementations

IEnumerable.GetEnumerator()

Returns an enumerator that iterates through a collection.

```
IEnumerator IEnumerable.GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

Interface IEzrString

Namespace: EzrSquared.Runtime.Types.Core.Text

Assembly: ezrSquared-lib.dll

Interface for getting the value of string-like ezr² objects as C# strings.

```
public interface IEzrString : IEzrObject
```

Inherited Members

IEzrObject.TypeName , IEzrObject.Tag , IEzrObject.HashTag , IEzrObject.StartPosition , IEzrObject.EndPosition , IEzrObject.Context , IEzrObject.CreationContext , IEzrObject.UpdateCreationContext(Context) , IEzrObject.Update(Context, Position, Position) , IEzrObject.ComparisonEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonNotEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , IEzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , IEzrObject.Addition(IEzrObject, RuntimeResult) , IEzrObject.Subtraction(IEzrObject, RuntimeResult) , IEzrObject.Multiplication(IEzrObject, RuntimeResult) , IEzrObject.Division(IEzrObject, RuntimeResult) , IEzrObject.Modulo(IEzrObject, RuntimeResult) , IEzrObject.Power(IEzrObject, RuntimeResult) , IEzrObject.Negation(RuntimeResult) , IEzrObject.Affirmation(RuntimeResult) , IEzrObject.BitwiseOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , IEzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , IEzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , IEzrObject.BitwiseNegation(RuntimeResult) , IEzrObject.HasValueContained(IEzrObject, RuntimeResult) , IEzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , IEzrObject.Inversion(RuntimeResult) , IEzrObject.EvaluateBoolean(RuntimeResult) , IEzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , IEzrObject.Execute(Reference[], Interpreter, RuntimeResult) , IEzrObject.StrictEquals(IEzrObject, RuntimeResult) , IEzrObject.ComputeHashCode(RuntimeResult) , IEzrObject.ToString(RuntimeResult) , IEzrObject.ToPureString(RuntimeResult)

Properties

StringValue

The string value.

```
string StringValue { get; }
```

Property Value

string

Namespace EzrSquared.Runtime.Types.Executables

Classes

EzrClass

The "type" type object? You know.

EzrClassInstance

The "instance of a class" type object?

EzrFunction

The function type object.

EzrRuntimeExecutable

The base root class of all runtime executables.

Class EzrClass

Namespace: EzrSquared.Runtime.Types.Executables

Assembly: ezrSquared-lib.dll

The "type" type object? You know.

```
public class EzrClass : EzrRuntimeExecutable, IEzrMutableObject,
IMutable<IEzrMutableObject>, IEzrObject
```

Inheritance

object ← EzrObject ← EzrRuntimeExecutable ← EzrClass

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrObject

Inherited Members

EzrRuntimeExecutable.ExecutableName , EzrRuntimeExecutable.IsAnonymous , EzrRuntimeExecutable.Body , EzrRuntimeExecutable.Parameters , EzrRuntimeExecutable.ExtraKeywordArguments , EzrRuntimeExecutable.ExtraPositionalArguments , EzrRuntimeExecutable.CheckAndPopulateArguments(Reference[], Context, Interpreter, RuntimeResult, bool) , EzrRuntimeExecutable.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.EvaluateBoolean(RuntimeResult) , EzrRuntimeExecutable.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.ComputeHashCode(RuntimeResult) , EzrRuntimeExecutable.ToString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,

EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrClass(string?, Node, EzrClass[], bool, bool, Interpreter, RuntimeResult, Context, Position, Position)

Creates a new class.

```
public EzrClass(string? name, Node body, EzrClass[] parents, bool isReadOnly, bool isStatic,
Interpreter interpreter, RuntimeResult result, Context parentContext, Position
startPosition, Position endPosition)
```

Parameters

`name` string

The name of the executable.

`body` Node

The source code body of the executable.

`parents` EzrClass[]

The parents of the class.

`isReadOnly` bool

Is this a read-only class?

`isStatic` bool

Is this a static class?

`interpreter` Interpreter

Interpreter for executing the static body of the class.

`result` RuntimeResult

Runtime result for carrying errors.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

EzrClass(string?, Node, (string Name, Node Node)[], (Position StartPosition, Position EndPosition, string Name)?, (Position StartPosition, Position EndPosition, string Name)?, EzrClass[], Reference[], bool, bool, Context, Context, Position, Position)

Creates a new class from the innards of an existing one.

```
public EzrClass(string? name, Node body, (string Name, Node Node)[] parameters, (Position StartPosition, Position EndPosition, string Name)? extraKeywordArguments, (Position StartPosition, Position EndPosition, string Name)? extraPositionalArguments, EzrClass[] parents, Reference[] staticParentReferences, bool isReadOnly, bool isStatic, Context staticContext, Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`name` string

The name of the executable.

`body` Node

The source code body of the executable.

`parameters` (string Name, Node Node)[]

The source code of the class's parameters and their default values.

`extraKeywordArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the class's extra keyword arguments.

`extraPositionalArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the class's extra positional arguments.

`parents` EzrClass[]

The parents of the class.

`staticParentReferences` Reference[]

The references to the class's static parents.

`isReadOnly` bool

Is this a read-only class?

`isStatic` bool

Is this a static class?

`staticContext` Context

The internal static context.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

IsStatic

Is the class static?

```
public readonly bool IsStatic
```

Field Value

bool

Parents

The parents of the class.

```
public readonly EzrClass[] Parents
```

Field Value

EzrClass[]

StaticParentReferences

The references to the class's static parents.

```
public readonly Reference[] StaticParentReferences
```

Field Value

Reference[]

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

AddParents(Reference[], Context, Interpreter, RuntimeResult)

```
private Reference[] AddParents(Reference[] arguments, Context context, Interpreter
interpreter, RuntimeResult result)
```

Parameters

`arguments` Reference[]

`context` Context

`interpreter` Interpreter

`result` RuntimeResult

Returns

Reference[]

Remarks

> **IMPORTANT**
>
> BUG: The name given to the parents is problematic, as two parents could have the same executable name.

AddStaticParents(RuntimeResult)

```
private void AddStaticParents(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Remarks

> **IMPORTANT**
>
> BUG: The name given to the parents is problematic, as two parents could have the same executable name.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Execute(Reference[], Interpreter, RuntimeResult)

Creates a new instance of the class.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Execute(Reference[], Interpreter, RuntimeResult, bool)

Creates a new instance of the class.

```
private void Execute(Reference[] arguments, Interpreter interpreter, RuntimeResult result,
```

```
    bool ignoreExtraArguments)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

`ignoreExtraArguments` bool

Should the arguments checker ignore extra parameters?

Class EzrClassInstance

Namespace: EzrSquared.Runtime.Types.Executables

Assembly: ezrSquared-lib.dll

The "instance of a class" type object?

```
public class EzrClassInstance : EzrObject, IEzrMutableObject, IMutable<IEzrMutableObject>,
IEzrObject
```

Inheritance

object ← EzrObject ← EzrClassInstance

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrObject

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject.TypeName , EzrObject.Tag , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrClassInstance(EzrClass, Reference[], Reference[], bool, Context, Node, bool, Interpreter, RuntimeResult, Context, Position, Position)

Creates a new instance of the class `class`.

```
public EzrClassInstance(EzrClass @class, Reference[] parentReferences, Reference[]
arguments, bool ignoreExtraArguments, Context context, Node body, bool readOnly, Interpreter
interpreter, RuntimeResult result, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`class` EzrClass

The parent class of the current object.

`parentReferences` Reference[]

The references to the class instance's parents.

`arguments` Reference[]

The arguments for the creation of the object.

`ignoreExtraArguments` bool

Should the arguments checker ignore extra arguments?

`context` Context

The internal context of the object.

`body` Node

The source code body of the object.

`readOnly` bool

Is the object read-only?

`interpreter` Interpreter

The interpreter for executing parts of the object.

`result` RuntimeResult

Runtime result for carrying any errors.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

EzrClassInstance(EzrClass, Reference[], bool, Interpreter, Context, Context, Position, Position)

Creates a new instance of the class `class`.

```
public EzrClassInstance(EzrClass @class, Reference[] parentReferences, bool readOnly,
Interpreter interpreter, Context context, Context parentContext, Position startPosition,
Position endPosition)
```

Parameters

`class` EzrClass

The parent class of the current object.

`parentReferences` Reference[]

The references to the class instance's parents.

`readOnly` bool

Is the object read-only?

`interpreter` Interpreter

The interpreter for executing parts of the object.

`context` Context

The internal context of the object.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

AdditionFunction

Special function name for 'this + other' operation.

```
public const string AdditionFunction = "addition"
```

Field Value

string

AffirmationFunction

Special function name for '+this' operation.

```
public const string AffirmationFunction = "affirmation"
```

Field Value

string

BitwiseAndFunction

Special function name for 'this & other' operation.

```
public const string BitwiseAndFunction = "bitwise_and"
```

Field Value

string

BitwiseLeftShiftunction

Special function name for 'this << other' operation.

```
public const string BitwiseLeftShiftunction = "bitwise_left_shift"
```

Field Value

string

BitwiseNegationFunction

Special function name for '~this' operation.

```
public const string BitwiseNegationFunction = "bitwise_negation"
```

Field Value

string

BitwiseOrFunction

Special function name for 'this | other' operation.

```
public const string BitwiseOrFunction = "bitwise_or"
```

Field Value

string

BitwiseRightShiftFunction

Special function name for 'this >> other' operation.

```
public const string BitwiseRightShiftFunction = "bitwise_right_shift"
```

Field Value

string

BitwiseXOrFunction

Special function name for 'this \ other' operation.

```
public const string BitwiseXOrFunction = "bitwise_xor"
```

Field Value

string

CalledFunction

Special function name for 'this(arguments)' operation.

```
public const string CalledFunction = "call_received"
```

Field Value

string

Class

The parent class of the class instance.

```
public readonly EzrClass Class
```

Field Value

EzrClass

ContainsFunction

Special function name for 'other in this' operation.

```
public const string ContainsFunction = "contains"
```

Field Value

string

DivisionFunction

Special function name for 'this / other' operation.

```
public const string DivisionFunction = "division"
```

Field Value

string

DoesNotContainFunction

Special function name for 'other not in this' operation.

```
public const string DoesNotContainFunction = "does_not_contain"
```

Field Value

string

EvaluateBooleanFunction

Special function name for evaluation of the 'this' into a boolean.

```
public const string EvaluateBooleanFunction = "evaluate_boolean"
```

Field Value

string

GetHashCodeFunction

Special function name for hashing 'this'.

```
public const string GetHashCodeFunction = "get_hash_code"
```

Field Value

string

InitializationFunction

Special function name for initializer (constructor).

```
public const string InitializationFunction = "initialize"
```

Field Value

string

Interpreter

The interpreter for executing parts of the class instance.

```
public readonly Interpreter Interpreter
```

Field Value

Interpreter

InversionFunction

Special function name for 'invert this' operation.

```
public const string InversionFunction = "inversion"
```

Field Value

string

IsEqualFunction

Special function name for 'this = other' operation.

```
public const string IsEqualFunction = "is_equal"
```

Field Value

string

IsGreaterThanFunction

Special function name for 'this > other' operation.

```
public const string IsGreaterThanFunction = "is_greater_than"
```

Field Value

string

IsGreaterThanOrEqualFunction

Special function name for 'this >= other' operation.

```
public const string IsGreaterThanOrEqualFunction = "is_greater_than_or_equal"
```

Field Value

string

IsLessThanFunction

Special function name for 'this < other' operation.

```
public const string IsLessThanFunction = "is_less_than"
```

Field Value

string

IsLessThanOrEqualFunction

Special function name for 'this <= other' operation.

```
public const string IsLessThanOrEqualFunction = "is_less_than_or_equal"
```

Field Value

string

IsNotEqualFunction

Special function name for 'this ! other' operation.

```
public const string IsNotEqualFunction = "is_inequal"
```

Field Value

string

ModuloFunction

Special function name for 'this % other' operation.

```
public const string ModuloFunction = "modulo"
```

Field Value

string

MultiplicationFunction

Special function name for 'this * other' operation.

```
public const string MultiplicationFunction = "multiplication"
```

Field Value

string

NegationFunction

Special function name for '-this' operation.

```
public const string NegationFunction = "negation"
```

Field Value

string

ParentReferences

The references to the class instance's parents.

```
public readonly Reference[] ParentReferences
```

Field Value

Reference[]

PowerFunction

Special function name for 'this ^ other' operation.

```
public const string PowerFunction = "power"
```

Field Value

string

StrictComparisonFunction

Special function name for strict equality checking of 'this' and 'other'.

```
public const string StrictComparisonFunction = "strict_equals"
```

Field Value

string

SubtractionFunction

Special function name for 'this - other' operation.

```
public const string SubtractionFunction = "subtraction"
```

Field Value

string

ToPureStringFunction

Special function name for the "pure" string representation of 'this'.

```
public const string ToPureStringFunction = "to_real_string"
```

Field Value

string

ToStringFunction

Special function name for the string representation of 'this'.

```
public const string ToStringFunction = "to_string"
```

Field Value

string

Methods

Addition(IEzrObject, RuntimeResult)

Performs the addition operation between the current object and another.

```
public override void Addition(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Affirmation(RuntimeResult)

Affirms the current object.

```
public override void Affirmation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseAnd(IEzrObject, RuntimeResult)

Performs the bit-wise AND operation between the current object and another.

```
public override void BitwiseAnd(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseLeftShift(IEzrObject, RuntimeResult)

Performs the bit-wise left-shift operation between the current object and another.

```
public override void BitwiseLeftShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseNegation(RuntimeResult)

Bit-wise negates the current object.

```
public override void BitwiseNegation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseOr(IEzrObject, RuntimeResult)

Performs the bit-wise OR operation between the current object and another.

```
public override void BitwiseOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseRightShift(IEzrObject, RuntimeResult)

Performs the bit-wise right-shift operation between the current object and another.

```
public override void BitwiseRightShift(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

BitwiseXOr(IEzrObject, RuntimeResult)

Performs the bit-wise X-OR operation between the current object and another.

```
public override void BitwiseXOr(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than the other.

```
public override void ComparisonGreaterThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is greater than or equal to the other.

```
public override void ComparisonGreaterThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThan(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than the other.

```
public override void ComparisonLessThan(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonLessThanOrEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks if the current object is less than or equal to the other.

```
public override void ComparisonLessThanOrEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Division(IEzrObject, RuntimeResult)

Performs the division operation between the current object and another.

```
public override void Division(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

GetFunction(string, int, out Reference)

Gets a function based on the name and number of parameters.

```
private bool GetFunction(string name, int parameters, out Reference functionReference)
```

Parameters

`name` string

The name of the function.

`parameters` int

The number of parameters required for the function.

`functionReference` Reference

The resulting reference to the function.

Returns

bool

true if successful, false otherwise.

HasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is contained in the current object.

```
public override void HasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Inversion(RuntimeResult)

Inverts the current object, like, for example, true to false.

```
public override void Inversion(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Modulo(IEzrObject, RuntimeResult)

Performs the modulo operation between the current object and another.

```
public override void Modulo(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Multiplication(IEzrObject, RuntimeResult)

Performs the multiplication operation between the current object and another.

```
public override void Multiplication(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Negation(RuntimeResult)

Negates the current object.

```
public override void Negation(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying the result and any errors.

NotHasValueContained(IEzrObject, RuntimeResult)

Checks if the other object is NOT contained in the current object.

```
public override void NotHasValueContained(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Power(IEzrObject, RuntimeResult)

Performs the power or exponent operation between the current object and another.

```
public override void Power(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

Subtraction(IEzrObject, RuntimeResult)

Performs the subtraction operation between the current object and another.

```
public override void Subtraction(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ToPureString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToPureString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Remarks

This is used to show the 'real' string representation of the object. Like, for example, ToString(RuntimeResult) will
return "example" when called on an EzrString object with value "example", but this function will return example (without quotes).

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Class EzrFunction

Namespace: EzrSquared.Runtime.Types.Executables

Assembly: ezrSquared-lib.dll

The function type object.

```
public class EzrFunction : EzrRuntimeExecutable, IEzrMutableObject,
IMutable<IEzrMutableObject>, IEzrObject
```

Inheritance

object ← EzrObject ← EzrRuntimeExecutable ← EzrFunction

Implements

IEzrMutableObject, IMutable<IEzrMutableObject>, IEzrObject

Inherited Members

EzrRuntimeExecutable.ExecutableName , EzrRuntimeExecutable.IsAnonymous , EzrRuntimeExecutable.Body , EzrRuntimeExecutable.Parameters , EzrRuntimeExecutable.ExtraKeywordArguments , EzrRuntimeExecutable.ExtraPositionalArguments , EzrRuntimeExecutable.CheckAndPopulateArguments(Reference[], Context, Interpreter, RuntimeResult, bool) , EzrRuntimeExecutable.ComparisonEqual(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.ComparisonNotEqual(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.EvaluateBoolean(RuntimeResult) , EzrRuntimeExecutable.StrictEquals(IEzrObject, RuntimeResult) , EzrRuntimeExecutable.ComputeHashCode(RuntimeResult) , EzrRuntimeExecutable.ToString(RuntimeResult) , EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) ,

EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) , EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrFunction(string?, Node, (string Name, Node Node)[], (Position StartPosition, Position EndPosition, string Name)?, (Position StartPosition, Position EndPosition, string Name)?, bool, Context, Position, Position)

The function type object.

```
public EzrFunction(string? name, Node body, (string Name, Node Node)[] parameters, (Position StartPosition, Position EndPosition, string Name)? extraKeywordArguments, (Position StartPosition, Position EndPosition, string Name)? extraPositionalArguments, bool returnLast, Context parentContext, Position startPosition, Position endPosition)
```

Parameters

`name` string

The name of the executable.

`body` Node

The source code body of the executable.

`parameters` (string Name, Node Node)[]

The source code of the executable's parameters and their default values.

`extraKeywordArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the executable's extra keyword arguments.

`extraPositionalArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the executable's extra positional arguments.

`returnLast` bool

Should the function return its last expression as the result?

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

Fields

ReturnLast

Should the function return its last expression as the result?

```
public readonly bool ReturnLast
```

Field Value

bool

Properties

Tag

The tag of the type of this object, similar to C# namespace naming conventions.

```
public override string Tag { get; protected internal set; }
```

Property Value

string

TypeName

The name of the type of this object, in plain text, all lowercase. Spaces *are* allowed.

```
public override string TypeName { get; protected internal set; }
```

Property Value

string

Methods

DeepCopy(RuntimeResult)

Creates a deep copy of the IMutable<T>.

```
public IMutable<IEzrMutableObject>? DeepCopy(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for raising errors./

Returns

IMutable<IEzrMutableObject>

The copy, or, null if failed.

Remarks

The deep copy here means that all IMutable<T> properties and fields in the object are also copied.

Execute(Reference[], Interpreter, RuntimeResult)

Executes the current object, like a function.

```
public override void Execute(Reference[] arguments, Interpreter interpreter,
RuntimeResult result)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

Execute(Reference[], Interpreter, RuntimeResult, bool)

Executes the current function.

```
public void Execute(Reference[] arguments, Interpreter interpreter, RuntimeResult result,
bool ignoreExtraArguments)
```

Parameters

`arguments` Reference[]

The arguments of the execution.

`interpreter` Interpreter

The interpreter to be used in execution.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

`ignoreExtraArguments` bool

Should the function ignore extra arguments?

Class EzrRuntimeExecutable

Namespace: EzrSquared.Runtime.Types.Executables

Assembly: ezrSquared-lib.dll

The base root class of all runtime executables.

```
public abstract class EzrRuntimeExecutable : EzrObject, IEzrObject
```

Inheritance

object ← EzrObject ← EzrRuntimeExecutable

Implements

IEzrObject

Derived

EzrClass, EzrFunction

Inherited Members

EzrObject.s_memberMap , EzrObject.GetMemberInfo<TMemberInfo, TParentType>(string) , EzrObject.TypeName , EzrObject.Tag , EzrObject._hashTag , EzrObject.HashTag , EzrObject.StartPosition , EzrObject.EndPosition , EzrObject.Context , EzrObject.CreationContext , EzrObject.IsReadOnly , EzrObject._executionContext , EzrObject.UpdateCreationContext(Context) , EzrObject.Update(Context, Position, Position) , EzrObject.ComparisonLessThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThan(IEzrObject, RuntimeResult) , EzrObject.ComparisonLessThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.ComparisonGreaterThanOrEqual(IEzrObject, RuntimeResult) , EzrObject.Addition(IEzrObject, RuntimeResult) , EzrObject.Subtraction(IEzrObject, RuntimeResult) , EzrObject.Multiplication(IEzrObject, RuntimeResult) , EzrObject.Division(IEzrObject, RuntimeResult) , EzrObject.Modulo(IEzrObject, RuntimeResult) , EzrObject.Power(IEzrObject, RuntimeResult) , EzrObject.Negation(RuntimeResult) , EzrObject.Affirmation(RuntimeResult) , EzrObject.BitwiseOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseXOr(IEzrObject, RuntimeResult) , EzrObject.BitwiseAnd(IEzrObject, RuntimeResult) , EzrObject.BitwiseLeftShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseRightShift(IEzrObject, RuntimeResult) , EzrObject.BitwiseNegation(RuntimeResult) , EzrObject.HasValueContained(IEzrObject, RuntimeResult) , EzrObject.NotHasValueContained(IEzrObject, RuntimeResult) , EzrObject.Inversion(RuntimeResult) , EzrObject.Interpret(Node, Context, Interpreter, RuntimeResult, bool) , EzrObject.Execute(Reference[], Interpreter, RuntimeResult) , EzrObject.ToPureString(RuntimeResult) , EzrObject.NewNothingConstant() , EzrObject.NewBooleanConstant(bool) ,

EzrObject.NewIntegerConstant(BigInteger) , EzrObject.NewFloatConstant(double) , EzrObject.NewStringConstant(string) , EzrObject.NewCharacterListConstant(string) , EzrObject.NewCharacterConstant(char) , EzrObject.NewArrayConstant(IEzrObject[]) , EzrObject.NewListConstant(RuntimeEzrObjectList) , EzrObject.NewDictionaryConstant(RuntimeEzrObjectDictionary) , EzrObject.IllegalOperation() , EzrObject.IllegalOperation(IEzrObject, bool) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

EzrRuntimeExecutable(string?, Node, (string Name, Node Node)[], (Position StartPosition, Position EndPosition, string Name)?, (Position StartPosition, Position EndPosition, string Name)?, Context, Position, Position, Context?)

Creates a new executable object.

```
public EzrRuntimeExecutable(string? name, Node body, (string Name, Node Node)[] parameters,
(Position StartPosition, Position EndPosition, string Name)? extraKeywordArguments,
(Position StartPosition, Position EndPosition, string Name)? extraPositionalArguments,
Context parentContext, Position startPosition, Position endPosition, Context?
initializationContext = null)
```

Parameters

`name` string

The name of the executable.

`body` Node

The source code body of the executable.

`parameters` (string Name, Node Node)[]

The source code of the executable's parameters and their default values.

`extraKeywordArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the executable's extra keyword arguments.

`extraPositionalArguments` (Position StartPosition, Position EndPosition, string Name)?

The position in source code and name of the variable for the executable's extra positional arguments.

`parentContext` Context

The parent context.

`startPosition` Position

The starting position of the object.

`endPosition` Position

The ending position of the object.

`initializationContext` Context

The internal context, if null, creates a new one.

Fields

Body

The source code body of the executable.

```
public readonly Node Body
```

Field Value

Node

ExecutableName

The name of the executable.

```
public readonly string ExecutableName
```

Field Value

string

IsAnonymous

Is the executable anonymous?

```
public readonly bool IsAnonymous
```

Field Value

bool

Properties

ExtraKeywordArguments

The position in source code and name of the variable for the executable's extra keyword arguments.

```
public (Position StartPosition, Position EndPosition, string Name)? ExtraKeywordArguments {
get; protected internal set; }
```

Property Value

(Position StartPosition, Position EndPosition, string Name)?

ExtraPositionalArguments

The position in source code and name of the variable for the executable's extra positional arguments.

```
public (Position StartPosition, Position EndPosition, string Name)? ExtraPositionalArguments
{ get; protected internal set; }
```

Property Value

(Position StartPosition, Position EndPosition, string Name)?

Parameters

The name and source code of the executable's parameters and their default values.

```
public (string Name, Node Node)[] Parameters { get; protected internal set; }
```

Property Value

(string Name, Node Node)[]

Methods

CheckAndPopulateArguments(Reference[], Context, Interpreter, RuntimeResult, bool)

Checks if the given array of arguments conform to the executables parameters, and populates them in the given context.

```
protected internal void CheckAndPopulateArguments(Reference[] arguments, Context context,
Interpreter interpreter, RuntimeResult result, bool ignoreExtraArguments)
```

Parameters

`arguments` Reference[]

The arguments to check.

`context` Context

The context to populate

`interpreter` Interpreter

The interpreter for executing parameter default values.

`result` RuntimeResult

Runtime result for carrying any errors.

`ignoreExtraArguments` bool

Should the checker ignore extra arguments?

ComparisonEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks equality.

```
public override void ComparisonEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComparisonNotEqual(IEzrObject, RuntimeResult)

Compares the object to another, checks unequality.

```
public override void ComparisonNotEqual(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying the result and any errors.

ComputeHashCode(RuntimeResult)

Evaluates the current object as its hash.

```
public override int ComputeHashCode(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

int

The evaluated value.

EvaluateBoolean(RuntimeResult)

Evaluates the current object as a boolean value.

```
public override bool EvaluateBoolean(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

The evaluated value.

StrictEquals(IEzrObject, RuntimeResult)

Strictly compares the current object to another, taking into account inheritance.

```
public override bool StrictEquals(IEzrObject other, RuntimeResult result)
```

Parameters

`other` IEzrObject

The other object in the operation.

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

bool

ToString(RuntimeResult)

Evaluates the current object as a string value.

```
public override string ToString(RuntimeResult result)
```

Parameters

`result` RuntimeResult

Runtime result for carrying any errors.

Returns

string

The evaluated value.

Namespace EzrSquared.Runtime.Wrapper Attributes

Classes

SharpAutoWrapperAttribute

Attribute for C# types and members to be automatically wrapped from C# types into ezr² types.

SharpDoNotWrapAttribute

Attribute for C# types and members which should NOT be automatically wrapped from C# types into ezr² types by the interpreter.

SharpMethodParameters

Class for the paramters of a wrapped C# method.

SharpMethodWrapperAttribute

Attribute for C# methods and constructors which will be wrapped into ezr².

SharpTypeWrapperAttribute

Attribute for C# classes which will be wrapped into ezr².

Class SharpAutoWrapperAttribute

Namespace: EzrSquared.Runtime.WrapperAttributes

Assembly: ezrSquared-lib.dll

Attribute for C# types and members to be automatically wrapped from C# types into ezr² types.

```
[AttributeUsage(AttributeTargets.Class|AttributeTargets.Struct|AttributeTargets.Method|Attri
buteTargets.Property|AttributeTargets.Field|AttributeTargets.Parameter, AllowMultiple =
false, Inherited = false)]
public class SharpAutoWrapperAttribute : Attribute
```

Inheritance

object ← Attribute ← SharpAutoWrapperAttribute

Inherited Members

Attribute.Equals(object) , Attribute.GetCustomAttribute(Assembly, Type) ,
Attribute.GetCustomAttribute(Assembly, Type, bool) ,
Attribute.GetCustomAttribute(MemberInfo, Type) ,
Attribute.GetCustomAttribute(MemberInfo, Type, bool) ,
Attribute.GetCustomAttribute(Module, Type) , Attribute.GetCustomAttribute(Module, Type, bool) ,
Attribute.GetCustomAttribute(ParameterInfo, Type) ,
Attribute.GetCustomAttribute(ParameterInfo, Type, bool) , Attribute.GetCustomAttributes(Assembly) ,
Attribute.GetCustomAttributes(Assembly, bool) , Attribute.GetCustomAttributes(Assembly, Type) ,
Attribute.GetCustomAttributes(Assembly, Type, bool) , Attribute.GetCustomAttributes(MemberInfo) ,
Attribute.GetCustomAttributes(MemberInfo, bool) ,
Attribute.GetCustomAttributes(MemberInfo, Type) ,
Attribute.GetCustomAttributes(MemberInfo, Type, bool) , Attribute.GetCustomAttributes(Module) ,
Attribute.GetCustomAttributes(Module, bool) , Attribute.GetCustomAttributes(Module, Type) ,
Attribute.GetCustomAttributes(Module, Type, bool) , Attribute.GetCustomAttributes(ParameterInfo) ,
Attribute.GetCustomAttributes(ParameterInfo, bool) ,
Attribute.GetCustomAttributes(ParameterInfo, Type) ,
Attribute.GetCustomAttributes(ParameterInfo, Type, bool) , Attribute.GetHashCode() ,
Attribute.IsDefaultAttribute() , Attribute.IsDefined(Assembly, Type) ,
Attribute.IsDefined(Assembly, Type, bool) , Attribute.IsDefined(MemberInfo, Type) ,
Attribute.IsDefined(MemberInfo, Type, bool) , Attribute.IsDefined(Module, Type) ,
Attribute.IsDefined(Module, Type, bool) , Attribute.IsDefined(ParameterInfo, Type) ,
Attribute.IsDefined(ParameterInfo, Type, bool) , Attribute.Match(object) , Attribute.TypeId ,
object.Equals(object, object) , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

SharpAutoWrapperAttribute(bool, bool)

Creates a new SharpAutoWrapperAttribute.

```
public SharpAutoWrapperAttribute(bool isReadOnly = false, bool isWriteOnly = false)
```

Parameters

`isReadOnly` bool

Is the member read-only? Only for properties and fields.

`isWriteOnly` bool

Is the member write-only? Only for properties and fields.

Exceptions

ArgumentException

Thrown if both IsReadOnly and IsWriteOnly are set to true.

SharpAutoWrapperAttribute(string, bool, bool)

Creates a new SharpAutoWrapperAttribute.

```
public SharpAutoWrapperAttribute(string name, bool isReadOnly = false, bool isWriteOnly
= false)
```

Parameters

`name` string

The ezr[2] name for the member.

`isReadOnly` bool

Is the member read-only? Only for properties and fields.

`isWriteOnly` bool

Is the member write-only? Only for properties and fields.

Fields

IsReadOnly

Is the member read-only? Only for properties and fields.

```
public readonly bool IsReadOnly
```

Field Value

bool

IsWriteOnly

Is the member write-only? Only for properties and fields.

```
public readonly bool IsWriteOnly
```

Field Value

bool

Name

The ezr^2 name for the member.

```
public readonly string Name
```

Field Value

string

Methods

GetIsPublic(MemberInfo)

Is the member publicly accessible in some way?

```
public static bool GetIsPublic(MemberInfo member)
```

Parameters

`member` MemberInfo

The member to check.

Returns

bool

true if yes, false otherwise.

Exceptions

ArgumentException

If `member` is of an unknown or unsupported type.

ShouldBeWrapped(MemberInfo)

Is the member eligible to be wrapped?

```
public static bool ShouldBeWrapped(MemberInfo member)
```

Parameters

`member` MemberInfo

The member to be wrapped.

Returns

bool

true if yes, false otherwise.

ValidateMethod(MethodBase, bool)

Checks if the given method has the supported signature for wrapping.

```
public static bool ValidateMethod(MethodBase methodBase, bool dontThrow = false)
```

Parameters

`methodBase` MethodBase

The method.

`dontThrow` bool

Disable exception throwing.

Returns

bool

true if valid, an exception or false otherwise.

Exceptions

ArgumentException

Thrown if the method is generic or has generic parameters.

Class SharpDoNotWrapAttribute

Namespace: EzrSquared.Runtime.WrapperAttributes

Assembly: ezrSquared-lib.dll

Attribute for C# types and members which should NOT be automatically wrapped from C# types into ezr² types by the interpreter.

```
[AttributeUsage(AttributeTargets.Class|AttributeTargets.Struct|AttributeTargets.Method|Attri
buteTargets.Property|AttributeTargets.Field, AllowMultiple = false, Inherited = false)]
public class SharpDoNotWrapAttribute : Attribute
```

Inheritance

object ← Attribute ← SharpDoNotWrapAttribute

Inherited Members

Attribute.Equals(object) , Attribute.GetCustomAttribute(Assembly, Type) ,
Attribute.GetCustomAttribute(Assembly, Type, bool) ,
Attribute.GetCustomAttribute(MemberInfo, Type) ,
Attribute.GetCustomAttribute(MemberInfo, Type, bool) ,
Attribute.GetCustomAttribute(Module, Type) , Attribute.GetCustomAttribute(Module, Type, bool) ,
Attribute.GetCustomAttribute(ParameterInfo, Type) ,
Attribute.GetCustomAttribute(ParameterInfo, Type, bool) , Attribute.GetCustomAttributes(Assembly) ,
Attribute.GetCustomAttributes(Assembly, bool) , Attribute.GetCustomAttributes(Assembly, Type) ,
Attribute.GetCustomAttributes(Assembly, Type, bool) , Attribute.GetCustomAttributes(MemberInfo) ,
Attribute.GetCustomAttributes(MemberInfo, bool) ,
Attribute.GetCustomAttributes(MemberInfo, Type) ,
Attribute.GetCustomAttributes(MemberInfo, Type, bool) , Attribute.GetCustomAttributes(Module) ,
Attribute.GetCustomAttributes(Module, bool) , Attribute.GetCustomAttributes(Module, Type) ,
Attribute.GetCustomAttributes(Module, Type, bool) , Attribute.GetCustomAttributes(ParameterInfo) ,
Attribute.GetCustomAttributes(ParameterInfo, bool) ,
Attribute.GetCustomAttributes(ParameterInfo, Type) ,
Attribute.GetCustomAttributes(ParameterInfo, Type, bool) , Attribute.GetHashCode() ,
Attribute.IsDefaultAttribute() , Attribute.IsDefined(Assembly, Type) ,
Attribute.IsDefined(Assembly, Type, bool) , Attribute.IsDefined(MemberInfo, Type) ,
Attribute.IsDefined(MemberInfo, Type, bool) , Attribute.IsDefined(Module, Type) ,
Attribute.IsDefined(Module, Type, bool) , Attribute.IsDefined(ParameterInfo, Type) ,
Attribute.IsDefined(ParameterInfo, Type, bool) , Attribute.Match(object) , Attribute.TypeId ,

object.Equals(object, object)↗ , object.GetType()↗ , object.MemberwiseClone()↗ , object.ReferenceEquals(object, object)↗ , object.ToString()↗

Class SharpMethodParameters

Namespace: EzrSquared.Runtime.WrapperAttributes

Assembly: ezrSquared-lib.dll

Class for the paramters of a wrapped C# method.

```
public class SharpMethodParameters
```

Inheritance

object ← SharpMethodParameters

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

SharpMethodParameters(Dictionary<string, Reference>, List<Reference>?, Context, Context, Context, Position, Position, Interpreter, RuntimeResult)

Class for the paramters of a wrapped C# method.

```
public SharpMethodParameters(Dictionary<string, Reference> argumentReferences,
List<Reference>? extraPositionalArgumentReferences, Context executionContext, Context
creationContext, Context methodContext, Position startPosition, Position endPosition,
Interpreter interpreter, RuntimeResult result)
```

Parameters

`argumentReferences` Dictionary<string, Reference>

See ArgumentReferences.

`extraPositionalArgumentReferences` List<Reference>

See ExtraPositionalArgumentReferences.

`executionContext` Context

See ExecutionContext.

`creationContext` Context

See CreationContext.

`methodContext` Context

See MethodContext.

`startPosition` Position

See StartPosition.

`endPosition` Position

See EndPosition.

`interpreter` Interpreter

See Interpreter.

`result` RuntimeResult

See Result.

Fields

ArgumentReferences

The references to the actual ezr^2 arguments.

```
public Dictionary<string, Reference> ArgumentReferences
```

Field Value

Dictionary<string, Reference>

CreationContext

The context under which the method wrapper was created.

```
public Context CreationContext
```

Field Value

Context

EndPosition

The ending position of the method wrapper object.

```
public Position EndPosition
```

Field Value

Position

ExecutionContext

The context under which the method is being executed.

```
public Context ExecutionContext
```

Field Value

Context

ExtraPositionalArgumentReferences

The references to optional extra positional ezr^2 arguments.

```
public List<Reference>? ExtraPositionalArgumentReferences
```

Field Value

List<Reference>

Interpreter

The interpreter executing the method.

```
public Interpreter Interpreter
```

Field Value

Interpreter

MethodContext

The context of the method wrapper itself.

```
public Context MethodContext
```

Field Value

Context

Result

The runtime result to return values or to throw errors.

```
public RuntimeResult Result
```

Field Value

RuntimeResult

StartPosition

The starting position of the method wrapper object.

```
public Position StartPosition
```

Field Value

Position

Class SharpMethodWrapperAttribute

Namespace: EzrSquared.Runtime.WrapperAttributes

Assembly: ezrSquared-lib.dll

Attribute for C# methods and constructors which will be wrapped into ezr².

```
[AttributeUsage(AttributeTargets.Constructor|AttributeTargets.Method, AllowMultiple = false,
Inherited = true)]
public class SharpMethodWrapperAttribute : Attribute
```

Inheritance

object ← Attribute ← SharpMethodWrapperAttribute

Inherited Members

Attribute.Equals(object) , Attribute.GetCustomAttribute(Assembly, Type) ,
Attribute.GetCustomAttribute(Assembly, Type, bool) ,
Attribute.GetCustomAttribute(MemberInfo, Type) ,
Attribute.GetCustomAttribute(MemberInfo, Type, bool) ,
Attribute.GetCustomAttribute(Module, Type) , Attribute.GetCustomAttribute(Module, Type, bool) ,
Attribute.GetCustomAttribute(ParameterInfo, Type) ,
Attribute.GetCustomAttribute(ParameterInfo, Type, bool) , Attribute.GetCustomAttributes(Assembly) ,
Attribute.GetCustomAttributes(Assembly, bool) , Attribute.GetCustomAttributes(Assembly, Type) ,
Attribute.GetCustomAttributes(Assembly, Type, bool) , Attribute.GetCustomAttributes(MemberInfo) ,
Attribute.GetCustomAttributes(MemberInfo, bool) ,
Attribute.GetCustomAttributes(MemberInfo, Type) ,
Attribute.GetCustomAttributes(MemberInfo, Type, bool) , Attribute.GetCustomAttributes(Module) ,
Attribute.GetCustomAttributes(Module, bool) , Attribute.GetCustomAttributes(Module, Type) ,
Attribute.GetCustomAttributes(Module, Type, bool) , Attribute.GetCustomAttributes(ParameterInfo) ,
Attribute.GetCustomAttributes(ParameterInfo, bool) ,
Attribute.GetCustomAttributes(ParameterInfo, Type) ,
Attribute.GetCustomAttributes(ParameterInfo, Type, bool) , Attribute.GetHashCode() ,
Attribute.IsDefaultAttribute() , Attribute.IsDefined(Assembly, Type) ,
Attribute.IsDefined(Assembly, Type, bool) , Attribute.IsDefined(MemberInfo, Type) ,
Attribute.IsDefined(MemberInfo, Type, bool) , Attribute.IsDefined(Module, Type) ,
Attribute.IsDefined(Module, Type, bool) , Attribute.IsDefined(ParameterInfo, Type) ,
Attribute.IsDefined(ParameterInfo, Type, bool) , Attribute.Match(object) , Attribute.TypeId ,
object.Equals(object, object) , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

SharpMethodWrapperAttribute()

Creates a new instance of SharpMethodWrapperAttribute.

```
public SharpMethodWrapperAttribute()
```

SharpMethodWrapperAttribute(string)

Creates a new instance of SharpMethodWrapperAttribute with a name.

```
public SharpMethodWrapperAttribute(string name)
```

Parameters

`name` string

The ezr² name for the method.

Fields

HasExtraKeywordArguments

Does this method support additional keyword arguments (kwargs)?

```
public bool HasExtraKeywordArguments
```

Field Value

bool

HasExtraPositionalArguments

Does this method support additional positional arguments (args)?

```
public bool HasExtraPositionalArguments
```

Field Value

bool

Name

The ezr² name for the method, optional.

```
public readonly string Name
```

Field Value

string

OptionalParameters

The ezr² names of the optional parameters of the method.

```
public string[] OptionalParameters
```

Field Value

string[]

RequiredParameters

The ezr² names of the required parameters of the method.

```
public string[] RequiredParameters
```

Field Value

string[]

Methods

ValidateMethodParameters(MethodBase)

Checks if the given method or constructor has the required parameters.

```
public static Exception? ValidateMethodParameters(MethodBase methodInfo)
```

Parameters

`methodInfo` MethodBase

The method or constructor to check.

Returns

Exception

null if the check was successful, an Exception otherwise.

Class SharpTypeWrapperAttribute

Namespace: EzrSquared.Runtime.WrapperAttributes

Assembly: ezrSquared-lib.dll

Attribute for C# classes which will be wrapped into ezr².

```
[AttributeUsage(AttributeTargets.Class, AllowMultiple = false, Inherited = true)]
public class SharpTypeWrapperAttribute : Attribute
```

Inheritance

object ← Attribute ← SharpTypeWrapperAttribute

Inherited Members

Attribute.Equals(object) , Attribute.GetCustomAttribute(Assembly, Type) ,
Attribute.GetCustomAttribute(Assembly, Type, bool) ,
Attribute.GetCustomAttribute(MemberInfo, Type) ,
Attribute.GetCustomAttribute(MemberInfo, Type, bool) ,
Attribute.GetCustomAttribute(Module, Type) , Attribute.GetCustomAttribute(Module, Type, bool) ,
Attribute.GetCustomAttribute(ParameterInfo, Type) ,
Attribute.GetCustomAttribute(ParameterInfo, Type, bool) , Attribute.GetCustomAttributes(Assembly) ,
Attribute.GetCustomAttributes(Assembly, bool) , Attribute.GetCustomAttributes(Assembly, Type) ,
Attribute.GetCustomAttributes(Assembly, Type, bool) , Attribute.GetCustomAttributes(MemberInfo) ,
Attribute.GetCustomAttributes(MemberInfo, bool) ,
Attribute.GetCustomAttributes(MemberInfo, Type) ,
Attribute.GetCustomAttributes(MemberInfo, Type, bool) , Attribute.GetCustomAttributes(Module) ,
Attribute.GetCustomAttributes(Module, bool) , Attribute.GetCustomAttributes(Module, Type) ,
Attribute.GetCustomAttributes(Module, Type, bool) , Attribute.GetCustomAttributes(ParameterInfo) ,
Attribute.GetCustomAttributes(ParameterInfo, bool) ,
Attribute.GetCustomAttributes(ParameterInfo, Type) ,
Attribute.GetCustomAttributes(ParameterInfo, Type, bool) , Attribute.GetHashCode() ,
Attribute.IsDefaultAttribute() , Attribute.IsDefined(Assembly, Type) ,
Attribute.IsDefined(Assembly, Type, bool) , Attribute.IsDefined(MemberInfo, Type) ,
Attribute.IsDefined(MemberInfo, Type, bool) , Attribute.IsDefined(Module, Type) ,
Attribute.IsDefined(Module, Type, bool) , Attribute.IsDefined(ParameterInfo, Type) ,
Attribute.IsDefined(ParameterInfo, Type, bool) , Attribute.Match(object) , Attribute.TypeId ,
object.Equals(object, object) , object.GetType() , object.MemberwiseClone() ,
object.ReferenceEquals(object, object) , object.ToString()

Constructors

SharpTypeWrapperAttribute(string)

Attribute for C# classes which will be wrapped into ezr².

```
public SharpTypeWrapperAttribute(string name)
```

Parameters

`name` string

The ezr² name for the type.

Fields

Name

The ezr² name for the type.

```
public readonly string Name
```

Field Value

string

Methods

ValidateMethodParameters(Type, out (ConstructorInfo Info, SharpMethodWrapperAttribute Attribute)?)

Checks if the given type has a constructor with the SharpMethodWrapperAttribute attribute.

```
public static Exception? ValidateMethodParameters(Type typeInfo, out (ConstructorInfo Info,
SharpMethodWrapperAttribute Attribute)? constructor)
```

Parameters

`typeInfo` Type

The type to check.

`constructor` (ConstructorInfo Info, SharpMethodWrapperAttribute Attribute)?

The constructor and its SharpMethodWrapperAttribute attribute.

Returns

Exception

null if the check was successful, an Exception otherwise.

Namespace EzrSquared.Syntax

Classes

Lexer

The ezr² Lexer or Tokenizer. The job of the Lexer is to convert the user input (code) into Token objects to be given as the input to the Parser.

ParseResult

The type of the object that is returned as the result of parsing done by the Parser.

Parser

The ezr² Parser. The job of the Parser is to convert the input Token objects from the Lexer into Node objects to be given as the input to the Interpreter.

Class Lexer

Namespace: EzrSquared.Syntax

Assembly: ezrSquared-lib.dll

The ezr² Lexer or Tokenizer. The job of the Lexer is to convert the user input (code) into Token objects to be given as the input to the Parser.

```
public class Lexer
```

Inheritance

object ← Lexer

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Lexer(string, string)

Creates a new Lexer object.

```
public Lexer(string file, string script)
```

Parameters

`file` string

The file name/path of the script.

`script` string

The script to be tokenized.

Fields

_currentChar

The character in the Position of the current lexing iteration in the script.

```
private char _currentChar
```

Field Value

char

_file

The file name/path of the script.

```
private readonly string _file
```

Field Value

string

_position

The Position of the current lexing iteration in the script.

```
private readonly Position _position
```

Field Value

Position

_reachedEnd

The value for checking if the Lexer has reached the end of the script.

```
private bool _reachedEnd
```

Field Value

bool

_script

The script to be tokenized.

```
private readonly string _script
```

Field Value

string

Methods

Advance()

Advances the current Position in the script.

```
private void Advance()
```

CompileColon()

Creates Colon and assignment type (AssignmentAddition, AssignmentMultiplication, etc) Token objects.

```
private Token CompileColon()
```

Returns

Token

The created Token.

CompileIdentifier()

Creates Identifier, keyword type (KeywordItem, KeywordFunction, etc) and qeyword type (QeywordC, QeywordFd, etc) Token objects.

```
private Token CompileIdentifier()
```

Returns

Token

The created Token.

CompileNewLines()

Goes through newline characters and creates a single Token object with TokenTypeNewLine.

```
private Token CompileNewLines()
```

Returns

Token

The Token object.

CompileNumber()

Goes through digit and period characters and creates a single Token object with TokenTypeFloatingPoint or Integer.

```
private Token CompileNumber()
```

Returns

Token

The Token object.

CompileStringLike(out EzrSyntaxError?)

Creates a Token of types String, Character or CharacterList, depending on the enclosing character.

```
private Token CompileStringLike(out EzrSyntaxError? error)
```

Parameters

`error` EzrSyntaxError

Any EzrSyntaxError that occurred in creating the stringlike; null if none occurred.

Returns

Token

The created Token.

ProcessEscapeSequence(StringBuilder, ref EzrSyntaxError?)

Processes an escape sequence in a stringlike.

```
private void ProcessEscapeSequence(StringBuilder builder, ref EzrSyntaxError? error)
```

Parameters

`builder` StringBuilder

The StringBuilder to append the special character to.

`error` EzrSyntaxError

Any EzrSyntaxError that occurred in the process; null if none occurred.

ProcessUtf16Sequence(ref EzrSyntaxError?)

Processes a UTF-16 escaped sequence in a stringlike.

```
private char[] ProcessUtf16Sequence(ref EzrSyntaxError? error)
```

Parameters

`error` EzrSyntaxError

Any EzrSyntaxError that occurred in the process; null if none occurred.

Returns

char[]

The UTF-16 character.

ProcessUtf32Sequence(ref EzrSyntaxError?)

Processes a UTF-32 escaped sequence in a stringlike.

```
private string ProcessUtf32Sequence(ref EzrSyntaxError? error)
```

Parameters

`error` EzrSyntaxError

Any EzrSyntaxError that occurred in the process; null if none occurred.

Returns

string

The UTF-32 character.

ReverseTo(int)

Reverses to the given index.

```
private void ReverseTo(int index)
```

Parameters

`index` int

The index to reverse to.

Remarks

Warning: The Line decrement is hard set to one if the character at `index` is a newline, and zero otherwise.

SkipComment()

Skips a comment in the ezr[2] code.

```
private void SkipComment()
```

Tokenize(out List<Token>)

Creates a List<T> of Token objects from the given script.

```
public EzrSyntaxError? Tokenize(out List<Token> tokens)
```

Parameters

`tokens` List<Token>

The created List<T> of Token objects.

Returns

EzrSyntaxError

Any EzrSyntaxError that occurred in the lexing; null if none occurred.

Class ParseResult

Namespace: EzrSquared.Syntax

Assembly: ezrSquared-lib.dll

The type of the object that is returned as the result of parsing done by the Parser.

```
public class ParseResult
```

Inheritance

object ← ParseResult

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

AdvanceCount

The amount of times the Parser advanced.

```
public int AdvanceCount
```

Field Value

int

Error

The EzrSyntaxError that occurred while parsing, if any.

```
public EzrSyntaxError? Error
```

Field Value

EzrSyntaxError

Node

The Node which is the result of the parsing.

```
public Node Node
```

Field Value

Node

_errorPriority

The priority of the error held in the ParseResult.

```
private int _errorPriority
```

Field Value

int

Methods

Failure(int, EzrSyntaxError)

Sets Error as the result of failed parsing.

```
public void Failure(int priority, EzrSyntaxError error)
```

Parameters

`priority` int

The priority/fatality of the failure.

`error` EzrSyntaxError

The EzrSyntaxError that occurred in parsing.

Remarks

If `priority` is greater than or equal to the _errorPriority then `error` will override Error.

Success(Node)

Sets Node as the result of successful parsing.

```
public void Success(Node node)
```

Parameters

`node` Node

The Node result of the parsing.

Class Parser

Namespace: EzrSquared.Syntax

Assembly: ezrSquared-lib.dll

The ezr² Parser. The job of the Parser is to convert the input Token objects from the Lexer into Node objects to be given as the input to the Interpreter.

```
public class Parser
```

Inheritance

object ← Parser

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Parser(List<Token>)

Creates a new Parser object.

```
public Parser(List<Token> tokens)
```

Parameters

`tokens` List<Token>

The List<T> of Token objects to be parsed.

Fields

_currentToken

The Token object currently being parsed at _index of _tokens.

```
private Token _currentToken
```

Field Value

Token

_index

The index of the Token object currently being parsed in the _tokensList<T>.

```
private int _index
```

Field Value

int

_result

The object that holds the result of the parsing.

```
private readonly ParseResult _result
```

Field Value

ParseResult

_tokens

The List<T> of Token objects to be parsed.

```
private readonly List<Token> _tokens
```

Field Value

List<Token>

Methods

Advance(int)

Advances to the next Token object in _tokens.

```
private void Advance(int advanceCount = 1)
```

Parameters

`advanceCount` int

BinaryOperation(Action, Action, TokenType[], (TokenType Type, Action OnCase)?)

Tries creating a BinaryOperationNode.

```
private void BinaryOperation(Action left, Action right, TokenType[] operators, (TokenType
Type, Action OnCase)? specialCase = null)
```

Parameters

`left` Action

The function to call for the first operand.

`right` Action

The function to call for the second operand.

`operators` TokenType[]

The operator TokenType object(s).

`specialCase` (TokenType Type, Action OnCase)?

Any special case that needs to be uniquely handled by the "OnCase" Action.

"OnCase" is called after the token of type "Type" and any new lines have been parsed, and before `right` has been called.
Note that "Type" must be included in `operators`.

Parse()

Parses the Token objects in _tokens.

```
public ParseResult Parse()
```

Returns

ParseResult

ParseArithmeticExpression()

Tries parsing an 'arithmetic-expression' structure.

```
private void ParseArithmeticExpression()
```

ParseArrayOrParentheticalExpression()

Tries parsing an array, an ArrayLikeNode with CreateList set to false OR a parenthetical expression. Starts from _currentToken, which should be of TokenTypeLeftParenthesis.

```
private void ParseArrayOrParentheticalExpression()
```

ParseAtom()

Tries parsing a 'atom' structure.

```
private void ParseAtom()
```

ParseBitwiseAnd()

Tries parsing a 'bitwise-and' structure.

```
private void ParseBitwiseAnd()
```

ParseBitwiseOr()

Tries parsing a 'bitwise-or' structure.

```
private void ParseBitwiseOr()
```

ParseBitwiseShift()

Tries creating a 'bitwise-shift' structure.

```
private void ParseBitwiseShift()
```

ParseBitwiseXOr()

Tries parsing a 'bitwise-xor' structure.

```
private void ParseBitwiseXOr()
```

ParseCall()

Tries parsing a 'call' structure.

```
private void ParseCall()
```

ParseClassDefinitionExpression()

Tries parsing an class definition expression. Starts from _currentToken, which should be of Token TypeKeywordObject.

```
private void ParseClassDefinitionExpression()
```

ParseComparison()

Tries parsing a 'comparison' structure.

```
private void ParseComparison()
```

ParseContainsCheck()

Tries parsing a 'contains-check' structure.

```
private void ParseContainsCheck()
```

ParseCountExpression()

Tries parsing a count expression. Starts from _currentToken, which should be of TokenTypeKeyword Count.

```
private void ParseCountExpression()
```

ParseDefineBlockExpression()

Tries parsing a define block expression. Starts from _currentToken, which should be of Token TypeKeywordDefine.

```
private void ParseDefineBlockExpression()
```

ParseDictionary()

Tries parsing a dictionary. Starts from _currentToken, which should be of TokenTypeLeftCurlyBracket.

```
private void ParseDictionary()
```

ParseExpression(bool)

Tries parsing an 'expression' structure.

```
private void ParseExpression(bool itemKeywordRequired = false)
```

Parameters

`itemKeywordRequired` bool

In cases where the variable assignment expression requires the 'item' keyword, set this to true.

ParseFactor()

Tries parsing a 'factor' structure.

```
private void ParseFactor()
```

ParseForEachExpression()

Tries parsing a for-each expression. Starts from _currentToken, which should be of TokenTypeKeywordFor.

```
private void ParseForEachExpression()
```

ParseFunctionDefinitionExpression()

Tries parsing a function definition expression. Starts from _currentToken, which should be of TokenTypeKeywordFunction.

```
private void ParseFunctionDefinitionExpression()
```

ParseIfExpression()

Tries parsing an if expression. Starts from _currentToken, which should be of TokenTypeKeywordIf.

```
private void ParseIfExpression()
```

ParseIncludeExpression()

Tries parsing an include expression. Starts from _currentToken, which should be of TokenTypeKeyword Include.

```
private void ParseIncludeExpression()
```

ParseInversion()

Tries parsing an 'inversion' structure.

```
private void ParseInversion()
```

ParseJunction()

Tries parsing a 'junction' structure.

```
private void ParseJunction()
```

ParseList()

Tries parsing a list, an ArrayLikeNode with CreateList set to true. Starts from _currentToken, which should be of TokenTypeLeftSquareBracket.

```
private void ParseList()
```

ParseObjectAttributeAccess()

Tries parsing an 'object-attribute-access' structure.

```
private void ParseObjectAttributeAccess()
```

ParsePower()

Tries parsing a 'power' structure.

```
private void ParsePower()
```

ParseQuickExpression(bool)

Tries parsing a 'quick-expression' structure.

```
private void ParseQuickExpression(bool itemKeywordRequired = false)
```

Parameters

`itemKeywordRequired` bool

ParseStatement()

Tries parsing a 'statement' structure.

```
private void ParseStatement()
```

ParseStatements()

Tries parsing a 'statements' structure.

```
private void ParseStatements()
```

ParseTerm()

Tries parsing a 'term' structure.

```
private void ParseTerm()
```

ParseTryExpression()

Tries parsing a try expression. Starts from _currentToken, which should be of TokenTypeKeywordTry.

```
private void ParseTryExpression()
```

ParseWhileExpression()

Tries parsing a while expression. Starts from _currentToken, which should be of TokenTypeKeywordWhile.

```
private void ParseWhileExpression()
```

PeekNext(int)

Peeks at the next Token object from _index in _tokens.

```
private Token PeekNext(int advanceCount = 1)
```

Parameters

`advanceCount` int

The numbers of places to advance in _tokens.

Returns

Token

The Token object.

PeekPrevious()

Peeks at the previous Token object from _index in _tokens.

```
private Token PeekPrevious()
```

Returns

Token

The Token object.

Reverse(int)

Reverses back to the Token object at _index - `reverseCount` in _tokens.

```
private void Reverse(int reverseCount = 1)
```

Parameters

`reverseCount` int

The number of positions to reverse _index in _tokens.

Namespace EzrSquared.Syntax.Errors

Classes

EzrStackedSyntaxError

The EzrSyntaxError returned when multiple EzrSyntaxError objects need to be returned to the user.

EzrSyntaxError

Error class for all syntax errors.

Class EzrStackedSyntaxError

Namespace: EzrSquared.Syntax.Errors

Assembly: ezrSquared-lib.dll

The EzrSyntaxError returned when multiple EzrSyntaxError objects need to be returned to the user.

```
internal class EzrStackedSyntaxError : EzrSyntaxError, IEzrError
```

Inheritance

object ← EzrSyntaxError ← EzrStackedSyntaxError

Implements

IEzrError

Inherited Members

EzrSyntaxError.UnexpectedCharacter , EzrSyntaxError.InvalidHexValue , EzrSyntaxError.InvalidGrammar , EzrSyntaxError.Title , EzrSyntaxError.Details , EzrSyntaxError.ErrorStartPosition , EzrSyntaxError.ErrorEndPosition , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

EzrStackedSyntaxError(EzrSyntaxError, EzrSyntaxError)

The EzrSyntaxError returned when multiple EzrSyntaxError objects need to be returned to the user.

```
public EzrStackedSyntaxError(EzrSyntaxError parent, EzrSyntaxError child)
```

Parameters

`parent` EzrSyntaxError

The 'parent' error, or the error that occurred first.

`child` EzrSyntaxError

The 'child' error, or the error that occurred because of the `parent`.

Methods

ToString()

Creates the formatted text representation of the EzrStackedSyntaxError, which shows all child EzrSyntaxError objects as the 'details'.

```
public override string ToString()
```

Returns

string

The formatted text.

Class EzrSyntaxError

Namespace: EzrSquared.Syntax.Errors

Assembly: ezrSquared-lib.dll

Error class for all syntax errors.

```
public class EzrSyntaxError : IEzrError
```

Inheritance

object ← EzrSyntaxError

Implements

IEzrError

Derived

EzrStackedSyntaxError

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

EzrSyntaxError(string, string, Position, Position)

Error class for all syntax errors.

```
public EzrSyntaxError(string title, string details, Position startPosition,
Position endPosition)
```

Parameters

`title` string

The title of the EzrSyntaxError.

`details` string

The reason why the EzrSyntaxError occurred.

`startPosition` Position

The starting Position of the EzrSyntaxError.

`endPosition` Position

The ending Position of the EzrSyntaxError.

Fields

InvalidGrammar

Invalid grammar was encountered.

```
public const string InvalidGrammar = "Invalid grammar"
```

Field Value

string

InvalidHexValue

An invalid hexadecimal value was encountered.

```
public const string InvalidHexValue = "Invalid hexadecimal value"
```

Field Value

string

UnexpectedCharacter

An unexpected character was encountered.

```
public const string UnexpectedCharacter = "Unexpected character"
```

Field Value

string

Properties

Details

The reason why the IEzrError occurred.

```
public string Details { get; }
```

Property Value

string

ErrorEndPosition

The ending Position of the IEzrError.

```
public Position ErrorEndPosition { get; }
```

Property Value

Position

ErrorStartPosition

The starting Position of the IEzrError.

```
public Position ErrorStartPosition { get; }
```

Property Value

Position

Title

The name of the IEzrError.

```
public string Title { get; }
```

Property Value

string

Methods

ToString()

Creates the formatted text representation of the EzrSyntaxError.

```
public override string ToString()
```

Returns

string

The formatted text.

Namespace EzrSquared.Util

Classes

UIDProvider

Generates unique IDs for objects.

Class UIDProvider

Namespace: EzrSquared.Util

Assembly: ezrSquared-lib.dll

Generates unique IDs for objects.

```
public static class UIDProvider
```

Inheritance

object ← UIDProvider

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

s_currentId

long value for the last generated unique identifier.

```
private static long s_currentId
```

Field Value

long

Methods

Get()

Generates an incremental unique identifier.

```
public static long Get()
```

Returns

long

A new unique identifier.

Namespace EzrSquared.Util.Extensions

Classes

BigIntegerExtensions

Static class extending the BigInteger type.

Class BigIntegerExtensions

Namespace: EzrSquared.Util.Extensions

Assembly: ezrSquared-lib.dll

Static class extending the BigInteger type.

```
public static class BigIntegerExtensions
```

Inheritance

object ← BigIntegerExtensions

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Power(char, BigInteger)

Extension for the power operation between a char and a BigInteger value.

```
public static BigInteger Power(this char thiz, BigInteger other)
```

Parameters

`thiz` char

The current char.

`other` BigInteger

The other BigInteger which acts as the exponent.

Returns

BigInteger

The resulting BigInteger.

Power(BigInteger, BigInteger)

Extension for the power operation between two BigInteger values.

```
public static BigInteger Power(this BigInteger thiz, BigInteger other)
```

Parameters

`thiz` BigInteger

The current BigInteger.

`other` BigInteger

The other BigInteger which acts as the exponent.

Returns

BigInteger

The resulting BigInteger.

Namespace EzrSquaredCli

Classes

EzrShellConsoleHelper

EzrShellEditor

Shell

The built-in shell for ezrSquared.

Class EzrShellConsoleHelper

Namespace: EzrSquaredCli

Assembly: ezrSquared.dll

```
internal static class EzrShellConsoleHelper
```

Inheritance

object ← EzrShellConsoleHelper

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

ConsoleGraphicsInfo

```
private const string ConsoleGraphicsInfo = "                                      █  ezr²
v{0}\r\n                                █  Online documentation:\r\n
█  https://uralstech.github.io/ezrSquared/docsrc/Learn-ezrSquared.html\r\n
█  Feature requests and bug reports:\r\n                                  █
https://github.com/Uralstech/ezrSquared/issues\r\n
█  GitHub repository:\r\n                          █
https://github.com/Uralstech/ezrSquared/\r\n                                   █ \r\n
█ \r\n
█ \r\n▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄█ "
```

Field Value

string

ConsoleGraphicsLetterE

```
private const string ConsoleGraphicsLetterE = "\r\n ▄█████▄\r\n █    █\r\n █
█\r\n ████▀\r\n █\r\n █\r\n █\r\n █\r\n ▀████"
```

Field Value

[string](#)

ConsoleGraphicsLetterR

```
private const string ConsoleGraphicsLetterR = "\r\n                        █▄▀█████▀▄\r\n
█▀▀    ▀█\r\n                          █\r\n                          █\r\n
█\r\n                            █\r\n                            █\r\n
█\r\n                            █"
```

Field Value

[string](#)

ConsoleGraphicsLetterZ

```
private const string ConsoleGraphicsLetterZ = "\r\n           █████████\r\n
▄█▀\r\n                        ▄█▀\r\n                        ▄█▀\r\n                    ▄█▀\r\n
▄█▀\r\n                    ▄█▀\r\n                    ▄█▀\r\n                    █████████"
```

Field Value

[string](#)

ConsoleGraphicsSquaredSymbol

```
private const string ConsoleGraphicsSquaredSymbol =
"                                  ████▙\r\n                                             █\r\n
▄▀████▀\r\n                                █\r\n
▀████\r\n\r\n\r\n\r\n"
```

Field Value

[string](#)

Methods

GetShellInput(string?, int?)

```
internal static string? GetShellInput(string? prompt = null, int? line = null)
```

Parameters

`prompt` string

`line` int?

Returns

string

PrintBigConsoleGraphics(string)

```
internal static void PrintBigConsoleGraphics(string version)
```

Parameters

`version` string

PrintSmallConsoleGraphics(string)

```
internal static void PrintSmallConsoleGraphics(string version)
```

Parameters

`version` string

ShowError(string)

```
internal static void ShowError(string message)
```

Parameters

`message` string

ShowMessage(string, ConsoleColor, bool, string?)

```
private static void ShowMessage(string message, ConsoleColor color, bool resetPosition =
false, string? newLine = null)
```

Parameters

`message` string

`color` ConsoleColor

`resetPosition` bool

`newLine` string

ShowOutput(string)

```
internal static void ShowOutput(string message)
```

Parameters

`message` string

ShowVerbose(string)

```
internal static void ShowVerbose(string message)
```

Parameters

`message` string

WaitForUser()

```
internal static void WaitForUser()
```

Class EzrShellEditor

Namespace: EzrSquaredCli

Assembly: ezrSquared.dll

```
internal class EzrShellEditor
```

Inheritance

object ← EzrShellEditor

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

AppendScriptToFile(string, string)

```
private static void AppendScriptToFile(string input, string filepath)
```

Parameters

`input` string

`filepath` string

ConfirmAndClearScript(StringBuilder)

```
private static void ConfirmAndClearScript(StringBuilder scriptBuilder)
```

Parameters

`scriptBuilder` StringBuilder

ConfirmAndQuitEditor()

```
private static bool ConfirmAndQuitEditor()
```

Returns

bool

IsValidPath(string, bool)

```
private static bool IsValidPath(string path, bool allowRelativePaths = false)
```

Parameters

`path` string

`allowRelativePaths` bool

Returns

bool

SaveScriptToFile(StringBuilder, ref string)

```
private static void SaveScriptToFile(StringBuilder scriptBuilder, ref string filepath)
```

Parameters

`scriptBuilder` StringBuilder

`filepath` string

StartShellEditor()

```
internal static string? StartShellEditor()
```

Returns

string

Class Shell

Namespace: EzrSquaredCli

Assembly: ezrSquared.dll

The built-in shell for ezrSquared.

```
internal class Shell
```

Inheritance

object ← Shell

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

Version

```
private const string Version = "0.11.3"
```

Field Value

string

s_filePath

```
private static string s_filePath
```

Field Value

string

s_showLexerOutput

```
private static bool s_showLexerOutput
```

Field Value

bool

s_showParserOutput

```
private static bool s_showParserOutput
```

Field Value

bool

Methods

ExecuteCode(string)

```
private static bool ExecuteCode(string script)
```

Parameters

`script` string

Returns

bool

ExecuteFile()

```
private static void ExecuteFile()
```

InteractiveMode()

```
private static void InteractiveMode()
```

Main()

```
public static void Main()
```

ParseCommandLineArguments(string[])

```
private static bool ParseCommandLineArguments(string[] arguments)
```

Parameters

`arguments` string[]

Returns

bool

ShowHelp()

```
private static void ShowHelp()
```