Namespace Inertia

Classes

INERTIA

2D physics simulation.

Class INERTIA

Namespace: [Inertia](#)

Assembly: Inertia.dll

2D physics simulation.

```
public class INERTIA : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← INERTIA

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Methods

ADDFORCE(int, double, double)

```
public void ADDFORCE(int id, double x, double y)
```

Parameters

id [int](#)

x [double](#)

y [double](#)

CREATESPHERE(double, double, double, double)

```
public int CREATESPHERE(double _1, double _2, double _3, double _4)
```

Parameters

_1 [double](⧉)

_2 [double](⧉)

_3 [double](⧉)

_4 [double](⧉)

Returns

[int](⧉)

DELETEBODY(int)

```
public void DELETEBODY(int id)
```

Parameters

id [int](⧉)

GETPOSITIONX(int)

```
public double GETPOSITIONX(int id)
```

Parameters

id [int](⧉)

Returns

[double](⧉)

GETPOSITIONY(int)

```
public double GETPOSITIONY(int id)
```

Parameters

id int⊠

Returns

double⊠

GETSPEED(int)

```
public double GETSPEED(int id)
```

Parameters

id int⊠

Returns

double⊠

LINK(int, string, bool, bool)

```
public void LINK(int id, string object_name, bool _1, bool _2)
```

Parameters

id int⊠

object_name string⊠

_1 bool⊠

_2 bool⊠

LOAD(string)

```
public void LOAD(string filename)
```

Parameters

filename [string](↗)

RESETTIMER()

```
public void RESETTIMER()
```

SETGRAVITY(double, double)

```
public void SETGRAVITY(double x, double y)
```

Parameters

x [double](↗)

y [double](↗)

SETLINEARDAMPING(double, double)

```
public void SETLINEARDAMPING(double x, double y)
```

Parameters

x [double](↗)

y [double](↗)

SETMATERIAL(int, string)

```
public void SETMATERIAL(int id, string material_name)
```

Parameters

id int⧉

material_name string⧉

SETPOSITION(int, double, double)

```
public void SETPOSITION(int id, double x, double y)
```

Parameters

id int⧉

x double⧉

y double⧉

SETVELOCITY(int, double, double)

```
public void SETVELOCITY(int id, double x, double y)
```

Parameters

id int⧉

x double⧉

y double⧉

TICK()

```
public void TICK()
```

UNLINK(int)

```
public void UNLINK(int id)
```

Parameters

id int⤤

Namespace Matrix

Classes

[MATRIX](#)
  2D Boulder Dash-like simulation.

Class MATRIX

Namespace: [Matrix](#)

Assembly: Matrix.dll

2D Boulder Dash-like simulation.

```
public class MATRIX : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← MATRIX

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

BASEPOS

```
public (int, int) BASEPOS { init; }
```

Property Value

([int](#), [int](#))

CELLHEIGHT

```
public int CELLHEIGHT { init; }
```

Property Value

[int ↗](#)

CELLWIDTH

```
public int CELLWIDTH { init; }
```

Property Value

[int ↗](#)

SIZE

```
public (int, int) SIZE { init; }
```

Property Value

([int ↗](#), [int ↗](#))

Methods

CALCENEMYMOVEDEST(int, int)

```
public int CALCENEMYMOVEDEST(int current_cell, int current_direction)
```

Parameters

current_cell [int ↗](#)

current_direction [int ↗](#)

Returns

[int ↗](#)

CALCENEMYMOVEDIR(int, int)

```
public int CALCENEMYMOVEDIR(int current_cell, int current_direction)
```

Parameters

current_cell [int](#)

current_direction [int](#)

Returns

[int](#)

CANHEROGOTO(int)

```
public bool CANHEROGOTO(int cell_index)
```

Parameters

cell_index [int](#)

Returns

[bool](#)

GET(int)

```
public int GET(int cell_index)
```

Parameters

cell_index [int](#)

Returns

[int](#)

GETCELLOFFSET(int, int)

```
public int GETCELLOFFSET(int x, int y)
```

Parameters

x [int](#)

y [int](#)

Returns

[int](#)

GETCELLPOSX(int)

```
public int GETCELLPOSX(int cell_index)
```

Parameters

cell_index [int](#)

Returns

[int](#)

GETCELLPOSY(int)

```
public int GETCELLPOSY(int cell_index)
```

Parameters

cell_index [int](#)

Returns

int⤤

GETCELLSNO(int)

```
public int GETCELLSNO(int cell_type)
```

Parameters

cell_type int⤤

Returns

int⤤

GETFIELDPOSX(int)

```
public int GETFIELDPOSX(int cell_index)
```

Parameters

cell_index int⤤

Returns

int⤤

GETFIELDPOSY(int)

```
public int GETFIELDPOSY(int cell_index)
```

Parameters

cell_index int⤤

Returns

int

GETOFFSET(int, int)

```
public int GETOFFSET(int x, int y)
```

Parameters

x int

y int

Returns

int

ISGATEEMPTY()

```
public bool ISGATEEMPTY()
```

Returns

bool

ISINGATE(int)

```
public bool ISINGATE(int _)
```

Parameters

_ int

Returns

[bool↗](#)

MOVE(int, int)

```
public void MOVE(int _, int _2)
```

Parameters

_ [int↗](#)

_2 [int↗](#)

NEXT()

```
public int NEXT()
```

Returns

[int↗](#)

SET(int, int)

```
public void SET(int cell_index, int cell_type)
```

Parameters

cell_index [int↗](#)

cell_type [int↗](#)

SETGATE(int, int, int, int)

```
public void SETGATE(int _, int _2, int _3, int _4)
```

Parameters

_ [int](#)

_2 [int](#)

_3 [int](#)

_4 [int](#)

SETROW(int, params int[])

```
public void SETROW(int row_index, params int[] cell_types)
```

Parameters

row_index [int](#)

cell_types [int](#)[]

TICK()

```
public void TICK()
```

Namespace PIKLib

Classes

[AA␣AA](#)

  A set of "global" methods callable in isolation from any object using the @ syntax.

[ANIMO](#)

  2D sprite animation.

[APPLICATION](#)

[ARRAY](#)

[BEHAVIOUR](#)

[BOOL](#)

  Boolean value.

[BUTTON](#)

[CANVAS_OBSERVER](#)

[CLASS](#)

[CNVLOADER](#)

[COMPLEXCONDITION](#)

[CONDITION](#)

[DATABASE](#)

[DOUBLE](#)

[EPISODE](#)

[EXPRESSION](#)

[FILTER](#)

[FONT](#)

[GROUP](#)

[IMAGE](#)

Class AA␀AA

Namespace:

Assembly: PIKLib.dll

A set of "global" methods callable in isolation from any object using the @ syntax.

```
public static class AA␀AA
```

Inheritance

object ← AA␀AA

Inherited Members

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

Methods

BOOL(string, bool)

Creates an object of type BOOL.

```
public static void BOOL(string name, bool value)
```

Parameters

`name` string

The name of created object.

`value` bool

The initial value of created object.

BREAK()

```
public static void BREAK()
```

DOUBLE(string, double)

Creates an object of type [DOUBLE](#).

```
public static void DOUBLE(string name, double value)
```

Parameters

name [string⧉](#)

   The name of created object.

value [double⧉](#)

   The initial value of created object.

IF(string, string, string)

```
public static void IF(string condition, string code_if_true, string code_if_false)
```

Parameters

condition [string⧉](#)

code_if_true [string⧉](#)

code_if_false [string⧉](#)

IF(string, string, string, string, string)

```
public static void IF(string left, string operand, string right, string
code_if_true, string code_if_false)
```

Parameters

left [string](#)

operand [string](#)

right [string](#)

code_if_true [string](#)

code_if_false [string](#)

INT(string, int)

Creates an object of type [INTEGER](#).

```
public static void INT(string name, int value)
```

Parameters

name [string](#)

   The name of created object.

value [int](#)

   The initial value of created object.

LOOP(string, int, int, int)

```
public static void LOOP(string behaviour, int init, int len, int step)
```

Parameters

behaviour [string](#)

init [int](#)

len [int](#)

step [int](#)

MSGBOX(string)

```
public static void MSGBOX(string message)
```

Parameters

message string↗

RETURN(variable)

```
public static variable RETURN(variable value)
```

Parameters

value variable

Returns

variable

STRING(string, string)

Creates an object of type STRING.

```
public static string STRING(string name, string value)
```

Parameters

name string↗

The name of created object.

value string↗

The initial value of created object.

Returns

string ⧉

WHILE(string, string, string, string)

```
public static void WHILE(string left, string condition, string right, string code)
```

Parameters

left string ⧉

condition string ⧉

right string ⧉

code string ⧉

Class ANIMO

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

2D sprite animation.

```
public class ANIMO : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← ANIMO

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

[string](#)

FPS

```
public int FPS { init; }
```

Property Value

[int ↗](#)

MONITORCOLLISIONALPHA

```
public bool MONITORCOLLISIONALPHA { init; }
```

Property Value

[bool ↗](#)

MONITORCOLLISION

```
public bool MONITORCOLLISION { init; }
```

Property Value

[bool ↗](#)

PRELOAD

```
public bool PRELOAD { init; }
```

Property Value

[bool ↗](#)

PRIORITY

```
public int PRIORITY { init; }
```

Property Value

[int ↗](#)

RELEASE

```
public bool RELEASE { init; }
```

Property Value

[bool](#)

TOCANVAS

```
public bool TOCANVAS { init; }
```

Property Value

[bool](#)

VISIBLE

```
public bool VISIBLE { init; }
```

Property Value

[bool](#)

Methods

GETCENTERX()

Retrieves the horizontal position of the center of the object.

```
public int GETCENTERX()
```

Returns

[int](#)

Horizontal position of the object's center.

GETCENTERY()

```
public int GETCENTERY()
```

Returns

int↗

GETCFRAMEINEVENT()

```
public int GETCFRAMEINEVENT()
```

Returns

int↗

GETCURRFRAMEPOSX()

```
public int GETCURRFRAMEPOSX()
```

Returns

int↗

GETCURRFRAMEPOSY()

```
public int GETCURRFRAMEPOSY()
```

Returns

int↗

GETENDX()

```
public int GETENDX()
```

Returns

[int](#)

GETENDY()

```
public int GETENDY()
```

Returns

[int](#)

GETEVENTNAME()

```
public string GETEVENTNAME()
```

Returns

[string](#)

GETFRAME()

```
public int GETFRAME()
```

Returns

[int](#)

GETFRAMENAME()

```
public string GETFRAMENAME()
```

Returns

[string](#)⌕

GETHEIGHT()

```
public int GETHEIGHT()
```

Returns

[int](#)⌕

GETMAXWIDTH()

```
public int GETMAXWIDTH()
```

Returns

[int](#)⌕

GETNOE()

```
public int GETNOE()
```

Returns

[int](#)⌕

GETNOF()

```
  public int GETNOF()
```

Returns

[int](#)

GETNOFINEVENT(string)

```
  public int GETNOFINEVENT(string event_name)
```

Parameters

event_name [string](#)

Returns

[int](#)

GETOPACITY()

```
  public int GETOPACITY()
```

Returns

[int](#)

GETPOSITIONX()

```
  public int GETPOSITIONX()
```

Returns

[int](#)

GETPOSITIONY()

```
public int GETPOSITIONY()
```

Returns

int⬀

GETPRIORITY()

```
public int GETPRIORITY()
```

Returns

int⬀

GETWIDTH()

```
public int GETWIDTH()
```

Returns

int⬀

HIDE()

Hides the object.

```
public void HIDE()
```

INVALIDATE()

```
public void INVALIDATE()
```

ISAT()

```
public bool ISAT()
```

Returns

[bool↗](#)

ISNEAR(string, string)

Checks if the object is near the `other` one.

```
public bool ISNEAR(string other, string iou_threshold)
```

Parameters

`other` [string↗](#)

Another graphics object for which nearness with the current object is checked.

`iou_threshold` [string↗](#)

Minimum IoU value to treat two objects as being near each other.

Returns

[bool↗](#)

Boolean value indicating if objects are near each other.

ISPLAYING()

```
public bool ISPLAYING()
```

Returns

[bool↗](#)

ISVISIBLE()

```
public bool ISVISIBLE()
```

Returns

[bool](#)

LOAD(string)

```
public void LOAD(string filename)
```

Parameters

filename [string](#)

MERGEALPHA()

```
public void MERGEALPHA()
```

MONITORCOLLISION(bool)

```
public void MONITORCOLLISION(bool pixel_perfect)
```

Parameters

pixel_perfect [bool](#)

MOVE(int, int)

```
public void MOVE(int x_offset, int y_offset)
```

Parameters

x_offset int⧉

y_offset int⧉

NEXT()

```
public void NEXT()
```

NEXTFRAME()

```
public void NEXTFRAME()
```

NPLAY()

```
public void NPLAY()
```

PAUSE()

```
public void PAUSE()
```

PLAY(int)

```
public void PLAY(int event_index)
```

Parameters

event_index int⧉

PLAY(string)

Plays animation event identified by name `event_name`.

```
public void PLAY(string event_name)
```

Parameters

`event_name` [string↗]

   The name of the animation event to play.

PREVFRAME()

```
public void PREVFRAME()
```

REMOVEMONITORCOLLISION()

```
public void REMOVEMONITORCOLLISION()
```

RESUME()

```
public void RESUME()
```

SETANCHOR(anchor)

```
public void SETANCHOR(anchor anchor)
```

Parameters

`anchor` [anchor]

SETASBUTTON(bool, bool)

```
public void SETASBUTTON(bool as_button, bool with_cursor_pointer)
```

Parameters

as_button [bool](#)

with_cursor_pointer [bool](#)

SETBACKWARD()

```
public void SETBACKWARD()
```

SETCLIPPING()

```
public void SETCLIPPING()
```

SETFORWARD()

```
public void SETFORWARD()
```

SETFPS(int)

```
public void SETFPS(int fps)
```

Parameters

fps [int](#)

SETFRAME(int)

```
public void SETFRAME(int image_index)
```

Parameters

image_index int⌐

SETFRAME(string, int)

```
public void SETFRAME(string event_name, int frame_index)
```

Parameters

event_name string⌐

frame_index int⌐

SETFRAMENAME(string)

```
public void SETFRAMENAME(string frame_name)
```

Parameters

frame_name string⌐

SETOPACITY(int)

```
public void SETOPACITY(int opacity)
```

Parameters

opacity int⌐

SETPOSITION(int, int)

```
public void SETPOSITION(int x, int y)
```

Parameters

x [int↗](#)

y [int↗](#)

SETPRIORITY(int)

```
public void SETPRIORITY(int priority)
```

Parameters

priority [int↗](#)

SHOW()

```
public void SHOW()
```

STOP(bool)

```
public void STOP(bool emit_on_finished = true)
```

Parameters

emit_on_finished [bool↗](#)

Class APPLICATION

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class APPLICATION : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← APPLICATION

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

AUTHOR

```
public string AUTHOR { init; }
```

Property Value

[string](#)

BLOOMOO_VERSION

```
public string BLOOMOO_VERSION { init; }
```

Property Value

[string](#)

CREATIONTIME

```
public string CREATIONTIME { init; }
```

Property Value

[string](#)

EPISODES

```
public string EPISODES { init; }
```

Property Value

[string](#)

LASTMODIFYTIME

```
public string LASTMODIFYTIME { init; }
```

Property Value

[string](#)

PATH

```
public string PATH { init; }
```

Property Value

[string](#)

STARTWITH

```
public string STARTWITH { init; }
```

Property Value

[string](#)⧉

VERSION

```
public string VERSION { init; }
```

Property Value

[string](#)⧉

Methods

EXIT()

```
public void EXIT()
```

GETLANGUAGE()

```
public string GETLANGUAGE()
```

Returns

[string](#)⧉

RUN(string, string, params variable[])

```
public variable? RUN(string object_name, string method_name, params
variable[] arguments)
```

Parameters

`object_name` [string](#)⬀

`method_name` [string](#)⬀

`arguments` [variable](#)[]

Returns

[variable](#)

RUNENV(string, string)

```
public variable? RUNENV(string scene_name, string beh_name)
```

Parameters

`scene_name` [string](#)⬀

`beh_name` [string](#)⬀

Returns

[variable](#)

SETLANGUAGE(string)

```
public void SETLANGUAGE(string lang_id)
```

Parameters

`lang_id` [string](#)⬀

Class ARRAY

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class ARRAY : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← ARRAY

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) , [OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) , [OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) , [object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Methods

ADD()

```
public void ADD()
```

ADDAT(int, variable)

```
public void ADDAT(int index, variable summand)
```

Parameters

index [int](#)

summand [variable](#)

CHANGEAT(int, variable)

```
public void CHANGEAT(int index, variable value)
```

Parameters

index int⧉

value variable

CLAMPAT(int, variable, variable)

```
public void CLAMPAT(int index, variable min, variable max)
```

Parameters

index int⧉

min variable

max variable

CONTAINS(variable)

```
public void CONTAINS(variable value)
```

Parameters

value variable

COPYTO()

```
public void COPYTO()
```

FIND()

```
public void FIND()
```

GET(int)

```
public void GET(int index)
```

Parameters

index int⌾

GETSIZE()

```
public void GETSIZE()
```

GETSUMVALUE()

```
public void GETSUMVALUE()
```

INSERTAT(int, variable)

```
public void INSERTAT(int index, variable value)
```

Parameters

index int⌾

value variable

LOAD()

```
public void LOAD()
```

LOADINI()

```
public void LOADINI()
```

MODAT()

```
public void MODAT()
```

MULAT()

```
public void MULAT()
```

REMOVE()

```
public void REMOVE()
```

REMOVEALL()

```
public void REMOVEALL()
```

REMOVEAT()

```
public void REMOVEAT()
```

REVERSEFIND()

```
public void REVERSEFIND()
```

SAVE()

```
public void SAVE()
```

SAVEINI()

```
public void SAVEINI()
```

SUB()

```
public void SUB()
```

SUBAT()

```
public void SUBAT()
```

SUM()

```
public void SUM()
```

Class BEHAVIOUR

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class BEHAVIOUR : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← BEHAVIOUR

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) , [OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) , [OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) , [object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Properties

CODE

```
public string CODE { init; }
```

Property Value

[string](#)

CONDITION

```
public string CONDITION { init; }
```

Property Value

[string](#)

Methods

RUN(params variable[])

```
public variable? RUN(params variable[] arguments)
```

Parameters

arguments [variable](#)[]

Returns

[variable](#)

RUNC(params variable[])

```
public variable? RUNC(params variable[] arguments)
```

Parameters

arguments [variable](#)[]

Returns

[variable](#)

RUNLOOPED(int, int, int)

```
public void RUNLOOPED(int start, int range_size, int step = 1)
```

Parameters

start [int](#)

range_size [int](#)

step [int](#)

Class BOOL

Namespace:

Boolean value.

```
public class BOOL : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← BOOL

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

TOINI

```
public bool TOINI { init; }
```

Property Value

[bool](#)

VALUE

```
public bool VALUE { init; }
```

Property Value

[bool↗](#)

Methods

SET(bool)

Sets the value of the object to `value`.

```
public void SET(bool value)
```

Parameters

`value` [bool↗](#)

New value for the object.

SWITCH(bool, bool)

Switches the value of the object between `TRUE` and `FALSE`.

```
public void SWITCH(bool _unused1, bool _unused2)
```

Parameters

`_unused1` [bool↗](#)

Unused.

`_unused2` [bool↗](#)

Unused.

Class BUTTON

Namespace:

Assembly: PIKLib.dll

```
public class BUTTON : OBJECT
```

Inheritance

object ← OBJECT ← BUTTON

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() ,
object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) ,
object.ToString()

Properties

DRAGGABLE

```
public bool DRAGGABLE { init; }
```

Property Value

bool

ENABLE␣

```
public bool ENABLE␣ { init; }
```

Property Value

bool

GFXONCLICK

```
public string GFXONCLICK { init; }
```

Property Value

[string](#)

GFXONMOVE

```
public string GFXONMOVE { init; }
```

Property Value

[string](#)

GFXSTANDARD

A graphics

```
public string GFXSTANDARD { init; }
```

Property Value

[string](#)

RECT

```
public rect RECT { init; }
```

Property Value

[rect](#)

SNDONMOVE

```
public string SNDONMOVE { init; }
```

Property Value

[string](#)

Methods

DISABLE()

```
public void DISABLE()
```

DISABLEBUTVISIBLE()

```
public void DISABLEBUTVISIBLE()
```

ENABLE()

```
public void ENABLE()
```

GETSTD()

```
public string GETSTD()
```

Returns

[string](#)

SETONCLICK(string)

```
public void SETONCLICK(string object_name)
```

Parameters

object_name [string⧉](#)

SETONMOVE(string)

```
public void SETONMOVE(string object_name)
```

Parameters

object_name [string⧉](#)

SETPRIORITY(int)

```
public void SETPRIORITY(int priority)
```

Parameters

priority [int⧉](#)

SETRECT(rect)

Sets the interactive area of the button to the given rect.

```
public void SETRECT(rect rect)
```

Parameters

rect [rect](#)

A literal rect described by four coordinates or a reference being the name of a graphical object to base the rect on, capturing its current state.

SETSTD(string)

```
public void SETSTD(string object_name)
```

Parameters

object_name string⌝

Class CANVAS_OBSERVER

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class CANVAS_OBSERVER : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← CANVAS_OBSERVER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Methods

ADD()

```
public void ADD()
```

ENABLENOTIFY()

```
public void ENABLENOTIFY()
```

GETGRAPHICSAT(int, int)

```
public string? GETGRAPHICSAT(int x_position, int y_position)
```

Parameters

x_position `int`⧉

y_position `int`⧉

Returns

[string](⧉)

GETGRAPHICSAT(int, int, bool, int, int, bool)

```
public string? GETGRAPHICSAT(int x_position, int y_position, bool _unknown, int
min_priority, int max_priority, bool pixel_perfect)
```

Parameters

x_position `int`⧉

y_position `int`⧉

_unknown `bool`⧉

min_priority `int`⧉

max_priority `int`⧉

pixel_perfect `bool`⧉

Returns

[string](⧉)

MOVEBKG(int, int)

```
public void MOVEBKG(int x_offset, int y_offset)
```

Parameters

x_offset `int`⧉

y_offset `int`⧉

PASTE()

```
public void PASTE()
```

REDRAW()

```
public void REDRAW()
```

REFRESH()

```
public void REFRESH()
```

REMOVE()

```
public void REMOVE()
```

SAVE(string)

```
public void SAVE(string filename)
```

Parameters

filename `string`⧉

SETBACKGROUND(string)

```
public void SETBACKGROUND(string object_name_or_filename)
```

Parameters

object_name_or_filename string⧉

SETBKGPOS(int, int)

```
public void SETBKGPOS(int x, int y)
```

Parameters

x int⧉

y int⧉

Class CLASS

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class CLASS : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← CLASS

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

BASE

```
public string BASE { init; }
```

Property Value

[string](#)

DEF

```
public string DEF { init; }
```

Property Value

[string](#)

Methods

NEW(string, params variable[])

```
public void NEW(string object_name, params variable[] arguments)
```

Parameters

object_name string⤢

arguments variable[]

Class CNVLOADER

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class CNVLOADER : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← CNVLOADER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Methods

LOAD()

```
public void LOAD()
```

RELEASE()

```
public void RELEASE()
```

Class COMPLEXCONDITION

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class COMPLEXCONDITION : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← COMPLEXCONDITION

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

CONDITION1

```
public string CONDITION1 { init; }
```

Property Value

[string](#)

CONDITION2

```
public string CONDITION2 { init; }
```

Property Value

[string](#)

OPERATOR

```
public complex_operator OPERATOR { init; }
```

Property Value

complex_operator

Methods

BREAK(bool)

```
public void BREAK(bool _)
```

Parameters

_ bool↗

CHECK(bool)

```
public bool CHECK(bool _)
```

Parameters

_ bool↗

Returns

bool↗

ONE_BREAK(bool)

```
public void ONE_BREAK(bool _)
```

Parameters

- [bool ↗](#)

Class CONDITION

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class CONDITION : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← CONDITION

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

OPERAND1

```
public string OPERAND1 { init; }
```

Property Value

[string](#)

OPERAND2

```
public string OPERAND2 { init; }
```

Property Value

[string](#)

OPERATOR

```
public condition_operator OPERATOR { init; }
```

Property Value

condition_operator

Methods

BREAK(bool)

```
public void BREAK(bool _)
```

Parameters

_ bool

CHECK(bool)

```
public bool CHECK(bool _)
```

Parameters

_ bool

Returns

bool

ONE_BREAK(bool)

```
public void ONE_BREAK(bool _)
```

Parameters

- [bool ⧉](#)

Class DATABASE

Namespace:

```
public class DATABASE : OBJECT
```

Inheritance

object ← OBJECT ← DATABASE

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() ,
object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) ,
object.ToString()

Properties

MODEL

```
public string MODEL { init; }
```

Property Value

string

Methods

ADD(string)

```
public void ADD(string object_name)
```

Parameters

object_name string⧉

FIND(string, variable, int)

```
public int FIND(string column_name, variable value, int start_row_index)
```

Parameters

column_name string⧉

value variable

start_row_index int⧉

Returns

int⧉

GETCURSORPOS()

```
public int GETCURSORPOS()
```

Returns

int⧉

GETROWSNO()

```
public int GETROWSNO()
```

Returns

int⧉

LOAD(string)

```
public void LOAD(string filename)
```

Parameters

filename [string](#)

NEXT()

```
public void NEXT()
```

REMOVEALL()

```
public void REMOVEALL()
```

SAVE(string)

```
public void SAVE(string filename)
```

Parameters

filename [string](#)

SELECT(int)

```
public void SELECT(int row_index)
```

Parameters

row_index [int](#)

Class DOUBLE

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class DOUBLE : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← DOUBLE

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

TOINI

```
public bool TOINI { init; }
```

Property Value

[bool](#)

VALUE

```
public double VALUE { init; }
```

Property Value

[double](#)

Methods

ADD(double)

```
public double ADD(double summand)
```

Parameters

summand [double⧉](#)

Returns

[double⧉](#)

ARCTAN(double)

```
public double ARCTAN(double degrees)
```

Parameters

degrees [double⧉](#)

Returns

[double⧉](#)

ARCTANEX(double, double, int)

```
public double ARCTANEX(double y, double x, int summand = 0)
```

Parameters

y [double⧉](#)

x [double⧉](#)

summand [int⧉](#)

Returns

double↗

CLAMP(double, double)

```
public double CLAMP(double min, double max)
```

Parameters

min double↗

max double↗

Returns

double↗

COSINUS(double)

```
public double COSINUS(double degrees)
```

Parameters

degrees double↗

Returns

double↗

DIV(double)

```
public void DIV(double divisor)
```

Parameters

divisor [double↗](#)

LENGTH(double, double)

```
public double LENGTH(double horizontal_distance, double vertical_distance)
```

Parameters

horizontal_distance [double↗](#)

vertical_distance [double↗](#)

Returns

[double↗](#)

MAXA(params double[])

```
public double MAXA(params double[] values)
```

Parameters

values [double↗](#)[]

Returns

[double↗](#)

MINA(params double[])

```
public double MINA(params double[] values)
```

Parameters

values `double`[]

Returns

`double`

MUL(double)

```
public void MUL(double multiplier)
```

Parameters

multiplier `double`

SET(double)

```
public void SET(double value)
```

Parameters

value `double`

SINUS(double)

```
public double SINUS(double degrees)
```

Parameters

degrees `double`

Returns

`double`

SQRT()

```
public double SQRT()
```

Returns

[double](#)⬈

SUB(double)

```
public double SUB(double subtrahend)
```

Parameters

subtrahend [double](#)⬈

Returns

[double](#)⬈

Class EPISODE

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class EPISODE : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← EPISODE

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

AUTHOR

```
public string AUTHOR { init; }
```

Property Value

[string](#)

CREATIONTIME

```
public string CREATIONTIME { init; }
```

Property Value

[string](#)

LASTMODIFYTIME

```
public string LASTMODIFYTIME { init; }
```

Property Value

[string](#)

PATH

```
public string PATH { init; }
```

Property Value

[string](#)

SCENES

```
public string[] SCENES { init; }
```

Property Value

[string](#)[]

STARTWITH

```
public string STARTWITH { init; }
```

Property Value

[string](#)

VERSION

```
public string VERSION { init; }
```

Property Value

[string](link)

Methods

BACK()

```
public void BACK()
```

GETCURRENTSCENE()

```
public string GETCURRENTSCENE()
```

Returns

[string](link)

GETLATESTSCENE()

```
public string GETLATESTSCENE()
```

Returns

[string](link)

GOTO(string)

```
public void GOTO(string scene_name)
```

Parameters

scene_name [string](#)

Class EXPRESSION

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class EXPRESSION : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← EXPRESSION

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) , [OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) , [OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) , [object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Properties

OPERAND1

```
public string OPERAND1 { init; }
```

Property Value

[string](#)

OPERAND2

```
public string OPERAND2 { init; }
```

Property Value

[string](#)

OPERATOR

```
public expression_operator OPERATOR { init; }
```

Property Value

expression_operator

Class FILTER

Namespace:

Assembly: PIKLib.dll

```
public class FILTER : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← FILTER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

ACTION

```
public string ACTION { init; }
```

Property Value

[string](#)

Class FONT

Namespace: PIKLib

Assembly: PIKLib.dll

```
public class FONT : OBJECT
```

Inheritance

object ← OBJECT ← FONT

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) , OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() , OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Properties

DEF_family_style_size

```
public string DEF_family_style_size { init; }
```

Property Value

string

Class GROUP

Namespace:

Assembly: PIKLib.dll

```
public class GROUP : OBJECT
```

Inheritance

object ← OBJECT ← GROUP

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() ,
object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) ,
object.ToString()

Methods

ADD(string)

```
public void ADD(string object_name)
```

Parameters

object_name string

ADDCLONES()

```
public void ADDCLONES()
```

GETSIZE()

```
public void GETSIZE()
```

NEXT()

```
public void NEXT()
```

PREV()

```
public void PREV()
```

REMOVE(string)

```
public void REMOVE(string object_name)
```

Parameters

object_name string⌤

REMOVEALL()

```
public void REMOVEALL()
```

RESETMARKER()

```
public void RESETMARKER()
```

SETMARKERPOS(int)

```
public void SETMARKERPOS(int index)
```

Parameters

index int⧉

Class IMAGE

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class IMAGE : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← IMAGE

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

[string](#)

MONITORCOLLISION

```
public bool MONITORCOLLISION { init; }
```

Property Value

[bool](#)

MONITORCOLLISIONALPHA

```
public bool MONITORCOLLISIONALPHA { init; }
```

Property Value

[bool](#)

PRELOAD

```
public bool PRELOAD { init; }
```

Property Value

[bool](#)

PRIORITY

```
public int PRIORITY { init; }
```

Property Value

[int](#)

RELEASE

```
public bool RELEASE { init; }
```

Property Value

[bool](#)

TOCANVAS

```
public bool TOCANVAS { init; }
```

Property Value

[bool](#)⧉

VISIBLE

```
public bool VISIBLE { init; }
```

Property Value

[bool](#)⧉

Methods

GETALPHA()

```
public void GETALPHA()
```

GETHEIGHT()

```
public void GETHEIGHT()
```

GETPIXEL()

```
public void GETPIXEL()
```

GETPOSITIONX()

```
public void GETPOSITIONX()
```

GETPOSITIONY()

```
public void GETPOSITIONY()
```

GETWIDTH()

```
public void GETWIDTH()
```

HIDE()

```
public void HIDE()
```

INVALIDATE()

```
public void INVALIDATE()
```

ISVISIBLE()

```
public void ISVISIBLE()
```

LOAD()

```
public void LOAD()
```

MERGEALPHA(int, int, string)

```
public void MERGEALPHA(int x_offset, int y_offset, string object_name)
```

Parameters

x_offset int⧉

y_offset int⧉

object_name string⧉

MOVE(int, int)

```
public void MOVE(int x_offset, int y_offset)
```

Parameters

x_offset int⧉

y_offset int⧉

SETASBUTTON()

```
public void SETASBUTTON()
```

SETCLIPPING(int, int, int, int)

```
public void SETCLIPPING(int left_x, int top_y, int _width, int _height)
```

Parameters

left_x int⧉

top_y int⧉

_width int⧉

`_height` [int](#)

SETOPACITY()

```
public void SETOPACITY()
```

SETPOSITION(int, int)

```
public void SETPOSITION(int x, int y)
```

Parameters

x [int](#)

y [int](#)

SETPRIORITY()

```
public void SETPRIORITY()
```

SHOW()

```
public void SHOW()
```

Class INTEGER

```
public class INTEGER : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← INTEGER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

TOINI

```
public bool TOINI { init; }
```

Property Value

[bool](#)

VALUE

```
public int VALUE { init; }
```

Property Value

[int](#)

VARTYPE

```
public string VARTYPE { init; }
```

Property Value

[string](#)

Methods

ABS(int)

Sets the modulus of `value` as the value of the object.

```
public int ABS(int value)
```

Parameters

`value` [int](#)

   The value of which modulus is to be set as the value of the object.

Returns

[int](#)

ADD(int)

```
public int ADD(int summand)
```

Parameters

`summand` [int](#)

Returns

[int](#)

AND(int)

```
public int AND(int operand)
```

Parameters

operand int↗

Returns

int↗

CLAMP(int, int)

```
public int CLAMP(int min, int max)
```

Parameters

min int↗

max int↗

Returns

int↗

DEC()

```
public void DEC()
```

DIV(int)

```
public void DIV(int divisor)
```

Parameters

divisor [int](#)

INC()

```
public void INC()
```

LENGTH(int, int)

```
public int LENGTH(int horizontal_distance, int vertical_distance)
```

Parameters

horizontal_distance [int](#)

vertical_distance [int](#)

Returns

[int](#)

MOD(int)

```
public void MOD(int divisor)
```

Parameters

divisor [int](#)

MUL(int)

```
public void MUL(int multiplier)
```

Parameters

multiplier int⬀

OR(int)

```
public int OR(int operand)
```

Parameters

operand int⬀

Returns

int⬀

RANDOM(int)

```
public int RANDOM(int max_exclusive)
```

Parameters

max_exclusive int⬀

Returns

int⬀

RANDOM(int, int)

```
public int RANDOM(int summand, int max_exclusive)
```

Parameters

summand int⬀

max_exclusive int⮺

Returns

int⮺

RESETINI()

```
public void RESETINI()
```

SET(int)

```
public void SET(int value)
```

Parameters

value int⮺

SUB(int)

```
public int SUB(int subtrahend)
```

Parameters

subtrahend int⮺

Returns

int⮺

SWITCH(int, int)

```
public void SWITCH(int value1, int value2)
```

Parameters

value1 [int](#)

value2 [int](#)

Class KEYBOARD

Namespace: <u>PIKLib</u>

Assembly: PIKLib.dll

```
public class KEYBOARD : OBJECT
```

Inheritance

<u>object</u> ← <u>OBJECT</u> ← KEYBOARD

Inherited Members

<u>OBJECT.DESCRIPTION</u> , <u>OBJECT.TYPE</u> , <u>OBJECT.ADDBEHAVIOUR(string, string)</u> ,
<u>OBJECT.CLONE(int)</u> , <u>OBJECT.GETCLONEINDEX()</u> , <u>OBJECT.GETNAME()</u> ,
<u>OBJECT.MSGBOX(string)</u> , <u>OBJECT.REMOVEBEHAVIOUR(string)</u> , <u>OBJECT.RESETCLONES()</u> ,
<u>object.Equals(object)</u> , <u>object.Equals(object, object)</u> , <u>object.GetHashCode()</u> ,
<u>object.GetType()</u> , <u>object.MemberwiseClone()</u> , <u>object.ReferenceEquals(object, object)</u> ,
<u>object.ToString()</u>

Methods

DISABLE()

```
public void DISABLE()
```

ENABLE()

```
public void ENABLE()
```

GETLATESTKEY()

```
public void GETLATESTKEY()
```

ISENABLED()

```
public bool ISENABLED()
```

Returns

[bool](#)⌐

ISKEYDOWN()

```
public bool ISKEYDOWN()
```

Returns

[bool](#)⌐

SETAUTOREPEAT()

```
public void SETAUTOREPEAT()
```

Class MOUSE

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class MOUSE : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← MOUSE

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

RAW

```
public int? RAW { init; }
```

Property Value

[int](#)?

Methods

DISABLE()

```
public void DISABLE()
```

DISABLESIGNAL()

```
public void DISABLESIGNAL()
```

ENABLE()

```
public void ENABLE()
```

ENABLESIGNAL()

```
public void ENABLESIGNAL()
```

GETPOSX()

```
public int GETPOSX()
```

Returns

int↗

GETPOSY()

```
public int GETPOSY()
```

Returns

int↗

HIDE()

```
public void HIDE()
```

ISLBUTTONDOWN()

```
public bool ISLBUTTONDOWN()
```

Returns

[bool⧉](#)

SET()

```
public void SET()
```

SETCLIPRECT()

```
public void SETCLIPRECT()
```

SETPOSITION(int, int)

```
public void SETPOSITION(int x, int y)
```

Parameters

x [int⧉](#)

y [int⧉](#)

SHOW()

```
public void SHOW()
```

Class MULTIARRAY

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class MULTIARRAY : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← MULTIARRAY

Inherited Members
[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

DIMENSIONS

```
public int DIMENSIONS { init; }
```

Property Value

[int](#)

Methods

GET(params int[])

```
public variable? GET(params int[] indices)
```

Parameters

indices int[]

Returns

[variable](#)

SET(variable, params int[])

```
public void SET(variable value, params int[] indices)
```

Parameters

value [variable](#)

indices int[]

Class MUSIC

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class MUSIC : OBJECT
```

Inheritance

[object](#)⊘ ← [OBJECT](#) ← MUSIC

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#)⊘ , [object.Equals(object, object)](#)⊘ , [object.GetHashCode()](#)⊘ ,
[object.GetType()](#)⊘ , [object.MemberwiseClone()](#)⊘ , [object.ReferenceEquals(object, object)](#)⊘ ,
[object.ToString()](#)⊘

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

[string](#)⊘

Methods

PLAY()

```
public void PLAY()
```

Class PATTERN

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class PATTERN : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← PATTERN

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

GRIDX

```
public int GRIDX { init; }
```

Property Value

[int](#)

GRIDY

```
public int GRIDY { init; }
```

Property Value

[int](#)

HEIGHT

```
public int HEIGHT { init; }
```

Property Value

[int](#)

LAYERS

```
public int LAYERS { init; }
```

Property Value

[int](#)

PRIORITY

```
public int PRIORITY { init; }
```

Property Value

[int](#)

TOCANVAS

```
public bool TOCANVAS { init; }
```

Property Value

[bool](#)

VISIBLE

```
public bool VISIBLE { init; }
```

Property Value

[bool](#)

WIDTH

```
public int WIDTH { init; }
```

Property Value

[int](#)

Methods

ADD(string, int, int, string, int)

```
public void ADD(string _, int x, int y, string object_name, int _2)
```

Parameters

_ [string](#)

x [int](#)

y [int](#)

object_name [string](#)

_2 [int](#)

GETGRAPHICSAT(int, int, bool, bool, int)

```
public string GETGRAPHICSAT(int x, int y, bool _, bool _2, int _3)
```

Parameters

x int↗

y int↗

_ bool↗

_2 bool↗

_3 int↗

Returns

string↗

MOVE(int, int)

```
public void MOVE(int x, int y)
```

Parameters

x int↗

y int↗

Class RAND

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class RAND : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← RAND

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Methods

GET(int)

```
public int GET(int max_exclusive)
```

Parameters

`max_exclusive` [int](#)

Returns

[int](#)

GET(int, int)

```
public int GET(int summand, int max_exclusive)
```

Parameters

summand [int](#)

max_exclusive [int](#)

Returns

[int](#)

GETPLENTY(string, int, int, int, bool)

```
public void GETPLENTY(string arr_name, int _, int _2, int _3, bool _4)
```

Parameters

arr_name [string](#)

_ [int](#)

_2 [int](#)

_3 [int](#)

_4 [bool](#)

Class SCENE

Namespace:

Assembly: PIKLib.dll

```
public class SCENE : OBJECT
```

Inheritance

object ← OBJECT ← SCENE

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() ,
object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) ,
object.ToString()

Properties

AUTHOR

```
public string AUTHOR { init; }
```

Property Value

string

BACKGROUND

```
public string BACKGROUND { init; }
```

Property Value

string

CREATIONTIME

```
public string CREATIONTIME { init; }
```

Property Value

[string](#)

DLLS

```
public string[] DLLS { init; }
```

Property Value

[string](#)[]

LASTMODIFYTIME

```
public string LASTMODIFYTIME { init; }
```

Property Value

[string](#)

MUSIC

```
public string MUSIC { init; }
```

Property Value

[string](#)

PATH

```
public string PATH { init; }
```

Property Value

[string](#)⧉

VERSION

```
public string VERSION { init; }
```

Property Value

[string](#)⧉

Methods

GETMAXHSPRIORITY()

```
public void GETMAXHSPRIORITY()
```

GETMINHSPRIORITY()

```
public void GETMINHSPRIORITY()
```

GETPLAYINGANIMO()

```
public void GETPLAYINGANIMO()
```

GETPLAYINGSEQ()

```
public void GETPLAYINGSEQ()
```

PAUSE()

```
public void PAUSE()
```

REMOVECLONES()

```
public void REMOVECLONES()
```

RESUME()

```
public void RESUME()
```

RUN(string, string, params variable[])

```
public variable? RUN(string object_name, string method_name, params
variable[] arguments)
```

Parameters

object_name string⧉

method_name string⧉

arguments variable[]

Returns

variable

RUNCLONES()

```
public void RUNCLONES()
```

SETMAXHSPRIORITY()

```
public void SETMAXHSPRIORITY()
```

SETMINHSPRIORITY()

```
public void SETMINHSPRIORITY()
```

SETMUSICVOLUME(int)

```
public void SETMUSICVOLUME(int volume)
```

Parameters

volume [int](#)

STARTMUSIC()

```
public void STARTMUSIC()
```

STOPMUSIC()

```
public void STOPMUSIC()
```

Class SEQUENCE

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class SEQUENCE : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← SEQUENCE

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) , [OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) , [OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) , [object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

[string](#)

Methods

GETEVENTNAME()

```
public string GETEVENTNAME()
```

Returns

[string ↗](#)

HIDE()

```
public void HIDE()
```

ISPLAYING()

```
public bool ISPLAYING()
```

Returns

[bool ↗](#)

PAUSE()

```
public void PAUSE()
```

PLAY(string)

```
public void PLAY(string parameter)
```

Parameters

parameter [string ↗](#)

RESUME()

```
public void RESUME()
```

STOP(bool)

```
public void STOP(bool emit_on_finished = true)
```

Parameters

emit_on_finished [bool](#)

Class SOUND

Namespace:

Assembly: PIKLib.dll

```
public class SOUND : OBJECT
```

Inheritance

object ⌐ ← OBJECT ← SOUND

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() ,
object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) ,
object.ToString()

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

string ⌐

FLUSHAFTERPLAYED

```
public bool FLUSHAFTERPLAYED { init; }
```

Property Value

bool ⌐

PRELOAD

```
public bool PRELOAD { init; }
```

Property Value

[bool](#)⬈

RELEASE

```
public bool RELEASE { init; }
```

Property Value

[bool](#)⬈

Methods

ISPLAYING()

```
public bool ISPLAYING()
```

Returns

[bool](#)⬈

LOAD(string)

```
public void LOAD(string filename)
```

Parameters

`filename` [string](#)⬈

PAUSE()

```
public void PAUSE()
```

PLAY()

```
public void PLAY()
```

RESUME()

```
public void RESUME()
```

SETVOLUME(int)

```
public void SETVOLUME(int volume)
```

Parameters

volume [int](#)

STOP()

```
public void STOP()
```

Class STATICFILTER

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class STATICFILTER : OBJECT
```

Inheritance

[object](#)⊞ ← [OBJECT](#) ← STATICFILTER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#)⊞ , [object.Equals(object, object)](#)⊞ , [object.GetHashCode()](#)⊞ ,
[object.GetType()](#)⊞ , [object.MemberwiseClone()](#)⊞ , [object.ReferenceEquals(object, object)](#)⊞ ,
[object.ToString()](#)⊞

Properties

ACTION

```
public string ACTION { init; }
```

Property Value

[string](#)⊞

Methods

LINK(string)

```
public void LINK(string graphics_name)
```

Parameters

graphics_name [string](#)

SETPROPERTY(string, variable)

```
public void SETPROPERTY(string key, variable value)
```

Parameters

key [string](#)

value [variable](#)

UNLINK(string)

```
public void UNLINK(string graphics_name)
```

Parameters

graphics_name [string](#)

Class STRING

Namespace:
Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class STRING : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← STRING

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

TOINI

```
public bool TOINI { init; }
```

Property Value

[bool](#)

VALUE

```
public string VALUE { init; }
```

Property Value

[string](#)

Methods

ADD(string)

```
public string ADD(string suffix)
```

Parameters

suffix string

Returns

string

COPYFILE(string, string)

```
public bool COPYFILE(string filename, string copied_filename)
```

Parameters

filename string

copied_filename string

Returns

bool

CUT(int, int)

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

Parameters

index int

length int

FIND(string, int)

```
public int FIND(string needle, int start_index = 0)
```

Parameters

needle [string](#)

start_index [int](#)

Returns

[int](#)

GET(int)

```
public string GET(int start_index)
```

Parameters

start_index [int](#)

Returns

[string](#)

GET(int, int)

```
public string GET(int start_index, int length)
```

Parameters

start_index [int](#)

length [int](#)

Returns

[string](#)

LENGTH()

```
public int LENGTH()
```

Returns

[int](#)

REPLACE(string, string)

```
public void REPLACE(string search, string replace)
```

Parameters

search [string](#)

replace [string](#)

REPLACEAT(int, string)

```
public void REPLACEAT(int index, string replace)
```

Parameters

index [int](#)

replace [string](#)

RESETINI()

```
public void RESETINI()
```

SET(string)

```
public void SET(string value)
```

Parameters

value string↗

SUB(int, int)

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

Parameters

index int↗

length int↗

UPPER()

```
public void UPPER()
```

Class STRUCT

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class STRUCT : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← STRUCT

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

FIELDS

```
public (string, string)[] FIELDS { init; }
```

Property Value

([string](#), [string](#))[]

Methods

GETFIELD(string)

```
public variable GETFIELD(string field_name)
```

Parameters

field_name [string](#)

Returns

[variable](#)

SET(string)

```
public void SET(string struct_name)
```

Parameters

struct_name [string](#)

SETFIELD(string, variable)

```
public void SETFIELD(string field_name, variable value)
```

Parameters

field_name [string](#)

value [variable](#)

Class SYSTEM

Namespace: <u>PIKLib</u>

Assembly: PIKLib.dll

```
public class SYSTEM : OBJECT
```

Inheritance

object ⊠ ← OBJECT ← SYSTEM

Inherited Members

OBJECT.DESCRIPTION , OBJECT.TYPE , OBJECT.ADDBEHAVIOUR(string, string) ,
OBJECT.CLONE(int) , OBJECT.GETCLONEINDEX() , OBJECT.GETNAME() ,
OBJECT.MSGBOX(string) , OBJECT.REMOVEBEHAVIOUR(string) , OBJECT.RESETCLONES() ,
object.Equals(object) ⊠ , object.Equals(object, object) ⊠ , object.GetHashCode() ⊠ ,
object.GetType() ⊠ , object.MemberwiseClone() ⊠ , object.ReferenceEquals(object, object) ⊠ ,
object.ToString() ⊠

Methods
GETDATE()

```
public string GETDATE()
```

Returns

string ⊠

GETMHZ()

```
public int GETMHZ()
```

Returns

int ⊠

GETMINUTES()

```
public int GETMINUTES()
```

Returns

int↗

GETSECONDS()

```
public int GETSECONDS()
```

Returns

int↗

GETSYSTEMTIME()

```
public int GETSYSTEMTIME()
```

Returns

int↗

Class TEXT

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class TEXT : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← TEXT

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

FONT

```
public string FONT { init; }
```

Property Value

[string](#)

HJUSTIFY

```
public bool HJUSTIFY { init; }
```

Property Value

[bool](#)

HYPERTEXT

```
public bool HYPERTEXT { init; }
```

Property Value

[bool](⧉)

MONITORCOLLISION

```
public bool MONITORCOLLISION { init; }
```

Property Value

[bool](⧉)

MONITORCOLLISIONALPHA

```
public bool MONITORCOLLISIONALPHA { init; }
```

Property Value

[bool](⧉)

RECT

```
public rect RECT { init; }
```

Property Value

[rect]

TEXT⬚

```
public string TEXT⬜ { init; }
```

Property Value

[string⬀](#)

TOCANVAS

```
public bool TOCANVAS { init; }
```

Property Value

[bool⬀](#)

VISIBLE

```
public bool VISIBLE { init; }
```

Property Value

[bool⬀](#)

VJUSTIFY

```
public bool VJUSTIFY { init; }
```

Property Value

[bool⬀](#)

Methods

HIDE()

```
public void HIDE()
```

SETCOLOR()

```
public void SETCOLOR()
```

SETJUSTIFY()

```
public void SETJUSTIFY()
```

SETPOSITION()

```
public void SETPOSITION()
```

SETTEXT(string)

```
public void SETTEXT(string text)
```

Parameters

text [string](⧉)

SHOW()

```
public void SHOW()
```

Class TIMER

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class TIMER : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← TIMER

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

ELAPSE

```
public int ELAPSE { init; }
```

Property Value

[int](#)

ENABLED

```
public bool ENABLED { init; }
```

Property Value

[bool](#)

TICKS

```
public int TICKS { init; }
```

Property Value

[int](#)

Methods

DISABLE()

```
public void DISABLE()
```

ENABLE()

```
public void ENABLE()
```

GETTICKS()

```
public int GETTICKS()
```

Returns

[int](#)

RESET()

```
public void RESET()
```

SET(int)

```
public void SET(int _)
```

Parameters

_ int⌃

SETELAPSE(int)

```
public void SETELAPSE(int _)
```

Parameters

_ int⌃

Class VECTOR

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class VECTOR : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← VECTOR

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

SIZE

```
public int SIZE { init; }
```

Property Value

[int](#)

VALUE

```
public double[] VALUE { init; }
```

Property Value

[double](#)[]

Methods

ADD(string)

```
public void ADD(string summand_name)
```

Parameters

summand_name string⧉

ASSIGN(params double[])

```
public void ASSIGN(params double[] values)
```

Parameters

values double⧉[]

GET(int)

```
public double GET(int index)
```

Parameters

index int⧉

Returns

double⧉

LEN()

```
public double LEN()
```

Returns

[double ⧉](#)

MUL(double)

```
public void MUL(double multiplier)
```

Parameters

multiplier [double ⧉](#)

NORMALIZE()

```
public void NORMALIZE()
```

REFLECT(string, string)

```
public void REFLECT(string normal_name, string result_name)
```

Parameters

normal_name [string ⧉](#)

result_name [string ⧉](#)

Class VIRTUALGRAPHICSOBJECT

Namespace: [PIKLib](#)

Assembly: PIKLib.dll

```
public class VIRTUALGRAPHICSOBJECT : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← VIRTUALGRAPHICSOBJECT

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

ASBUTTON

```
public bool ASBUTTON { init; }
```

Property Value

[bool](#)

MASK

```
public string MASK { init; }
```

Property Value

[string](#)

MONITORCOLLISION

```
public bool MONITORCOLLISION { init; }
```

Property Value

[bool](⧉)

MONITORCOLLISIONALPHA

```
public bool MONITORCOLLISIONALPHA { init; }
```

Property Value

[bool](⧉)

PRIORITY

```
public int PRIORITY { init; }
```

Property Value

[int](⧉)

SOURCE

```
public string SOURCE { init; }
```

Property Value

[string](⧉)

TOCANVAS

```
public bool TOCANVAS { init; }
```

Property Value

[bool⧉](#)

VISIBLE

```
public bool VISIBLE { init; }
```

Property Value

[bool⧉](#)

Methods

GETHEIGHT()

```
public int GETHEIGHT()
```

Returns

[int⧉](#)

GETPOSITIONX()

```
public int GETPOSITIONX()
```

Returns

[int⧉](#)

GETPOSITIONY()

```
public int GETPOSITIONY()
```

Returns

[int↗](#)

GETWIDTH()

```
public int GETWIDTH()
```

Returns

[int↗](#)

MOVE(int, int)

```
public void MOVE(int x_offset, int y_offset)
```

Parameters

x_offset [int↗](#)

y_offset [int↗](#)

SETMASK(string)

```
public void SETMASK(string graphics_name)
```

Parameters

graphics_name [string↗](#)

SETPOSITION(int, int)

```
public void SETPOSITION(int x, int y)
```

Parameters

x [int](#)

y [int](#)

SETPRIORITY(int)

```
public void SETPRIORITY(int priority)
```

Parameters

priority [int](#)

SETSOURCE(string)

```
public void SETSOURCE(string graphics_name)
```

Parameters

graphics_name [string](#)

Namespace World

Classes

WORLD

    3D physics simulation.

Class WORLD

Namespace: [World](#)

Assembly: World.dll

3D physics simulation.

```
public class WORLD : OBJECT
```

Inheritance

[object](#) ← [OBJECT](#) ← WORLD

Inherited Members

[OBJECT.DESCRIPTION](#) , [OBJECT.TYPE](#) , [OBJECT.ADDBEHAVIOUR(string, string)](#) ,
[OBJECT.CLONE(int)](#) , [OBJECT.GETCLONEINDEX()](#) , [OBJECT.GETNAME()](#) ,
[OBJECT.MSGBOX(string)](#) , [OBJECT.REMOVEBEHAVIOUR(string)](#) , [OBJECT.RESETCLONES()](#) ,
[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Properties

FILENAME

```
public string FILENAME { init; }
```

Property Value

[string](#)

Methods

ADDBODY()

```
public void ADDBODY()
```

ADDFORCE()

```
public void ADDFORCE()
```

ADDGRAVITYEX()

```
public void ADDGRAVITYEX()
```

FINDPATH()

```
public void FINDPATH()
```

FOLLOWPATH()

```
public void FOLLOWPATH()
```

GETANGLE()

```
public void GETANGLE()
```

GETBKGPOSX()

```
public void GETBKGPOSX()
```

GETBKGPOSY()

```
public void GETBKGPOSY()
```

GETMOVEDISTANCE()

```
public void GETMOVEDISTANCE()
```

GETPOSITIONX()

```
public void GETPOSITIONX()
```

GETPOSITIONY()

```
public void GETPOSITIONY()
```

GETPOSITIONZ()

```
public void GETPOSITIONZ()
```

GETROTATIONZ()

```
public void GETROTATIONZ()
```

GETSPEED()

```
public void GETSPEED()
```

JOIN()

```
public void JOIN()
```

LINK()

```
public void LINK()
```

LOAD()

```
public void LOAD()
```

MOVEOBJECTS()

```
public void MOVEOBJECTS()
```

REMOVEOBJECT()

```
public void REMOVEOBJECT()
```

SETACTIVE()

```
public void SETACTIVE()
```

SETBKGSIZE()

```
public void SETBKGSIZE()
```

SETBODYDYNAMICS()

```
public void SETBODYDYNAMICS()
```

SETG()

```
public void SETG()
```

SETGRAVITY()

```
public void SETGRAVITY()
```

SETGRAVITYCENTER()

```
public void SETGRAVITYCENTER()
```

SETLIMIT()

```
public void SETLIMIT()
```

SETMAXSPEED()

```
public void SETMAXSPEED()
```

SETMOVEFLAGS()

```
public void SETMOVEFLAGS()
```

SETPOSITION()

```
public void SETPOSITION()
```

SETREFOBJECT()

```
public void SETREFOBJECT()
```

SETVELOCITY()

```
public void SETVELOCITY()
```

START()

```
public void START()
```

STOP()

```
public void STOP()
```

UNLINK()

```
public void UNLINK()
```

Namespace abstractions

Classes

BoolVariable

DoubleVariable

IntVariable

LiteralRect

OBJECT

ReferenceRect

StringVariable

rect

variable

Enums

anchor

complex_operator

condition_operator

expression_operator

Delegates

ParametrizedSignalHandler
   Run for parametrized signals.

SignalHandler
   Run for signals.

Class BoolVariable

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record BoolVariable : variable, IEquatable<variable>,
IEquatable<BoolVariable>
```

Inheritance

[object](#) ← [variable](#) ← BoolVariable

Implements

[IEquatable](#)<[variable](#)>, [IEquatable](#)<[BoolVariable](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Constructors

BoolVariable(bool)

```
public BoolVariable(bool value)
```

Parameters

value [bool](#)

Properties

value

```
public bool value { get; init; }
```

Property Value

[bool](#)⧉

Class DoubleVariable

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record DoubleVariable : variable, IEquatable<variable>,
IEquatable<DoubleVariable>
```

Inheritance

[object](#) ← [variable](#) ← DoubleVariable

Implements

[IEquatable](#)<[variable](#)>, [IEquatable](#)<[DoubleVariable](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Constructors

DoubleVariable(double)

```
public DoubleVariable(double value)
```

Parameters

`value` [double](#)

Properties

value

```
public double value { get; init; }
```

Property Value

[double](#)↗

Class IntVariable

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record IntVariable : variable, IEquatable<variable>, IEquatable<IntVariable>
```

Inheritance

[object](#) ← [variable](#) ← IntVariable

Implements

[IEquatable](#)<[variable](#)>, [IEquatable](#)<[IntVariable](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Constructors

IntVariable(int)

```
public IntVariable(int value)
```

Parameters

`value` [int](#)

Properties

value

```
public int value { get; init; }
```

Property Value

[int](#)

Class LiteralRect

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record LiteralRect : rect, IEquatable<rect>, IEquatable<LiteralRect>
```

Inheritance

[object](#) ← [rect](#) ← LiteralRect

Implements

[IEquatable](#)<[rect](#)>, [IEquatable](#)<[LiteralRect](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Constructors

LiteralRect(int, int, int, int)

```
public LiteralRect(int left_x, int top_y, int right_x, int bottom_y)
```

Parameters

`left_x` [int](#)

`top_y` [int](#)

`right_x` [int](#)

`bottom_y` [int](#)

Properties

bottom_y

```
public int bottom_y { get; init; }
```

Property Value

[int↗]

left_x

```
public int left_x { get; init; }
```

Property Value

[int↗]

right_x

```
public int right_x { get; init; }
```

Property Value

[int↗]

top_y

```
public int top_y { get; init; }
```

Property Value

[int↗]

Class OBJECT

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public abstract class OBJECT
```

Inheritance

[object](#) ← OBJECT

Derived

[INERTIA](#), [MATRIX](#), [ANIMO](#), [APPLICATION](#), [ARRAY](#), [BEHAVIOUR](#), [BOOL](#), [BUTTON](#), [CANVAS_OBSERVER](#), [CLASS](#), [CNVLOADER](#), [COMPLEXCONDITION](#), [CONDITION](#), [DATABASE](#), [DOUBLE](#), [EPISODE](#), [EXPRESSION](#), [FILTER](#), [FONT](#), [GROUP](#), [IMAGE](#), [INTEGER](#), [KEYBOARD](#), [MOUSE](#), [MULTIARRAY](#), [MUSIC](#), [PATTERN](#), [RAND](#), [SCENE](#), [SEQUENCE](#), [SOUND](#), [STATICFILTER](#), [STRING](#), [STRUCT](#), [SYSTEM](#), [TEXT](#), [TIMER](#), [VECTOR](#), [VIRTUALGRAPHICSOBJECT](#), [WORLD](#)

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)

Properties

DESCRIPTION

```
public string DESCRIPTION { init; }
```

Property Value

[string](#)

TYPE

```
public string TYPE { init; }
```

Property Value

string⧉

Methods

ADDBEHAVIOUR(string, string)

```
public void ADDBEHAVIOUR(string signal_name, string code)
```

Parameters

signal_name string⧉

code string⧉

CLONE(int)

```
public void CLONE(int count = 1)
```

Parameters

count int⧉

GETCLONEINDEX()

```
public int GETCLONEINDEX()
```

Returns

int⧉

GETNAME()

```
  public string GETNAME()
```

Returns

[string](#)⧉

MSGBOX(string)

```
  public void MSGBOX(string message)
```

Parameters

message [string](#)⧉

REMOVEBEHAVIOUR(string)

```
  public void REMOVEBEHAVIOUR(string signal_name)
```

Parameters

signal_name [string](#)⧉

RESETCLONES()

```
  public void RESETCLONES()
```

Delegate ParametrizedSignalHandler

Namespace: [abstractions](#)

Assembly: PIKLib.dll

Run for parametrized signals.

```
public delegate void ParametrizedSignalHandler(string parameter, params
variable[] arguments)
```

Parameters

`parameter` [string](#)

Run for parametrized signals.

`arguments` [variable](#)[]

Run for parametrized signals.

Class ReferenceRect

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record ReferenceRect : rect, IEquatable<rect>, IEquatable<ReferenceRect>
```

Inheritance

[object](#) ← [rect](#) ← ReferenceRect

Implements

[IEquatable](#)<[rect](#)>, [IEquatable](#)<[ReferenceRect](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Constructors

ReferenceRect(string)

```
public ReferenceRect(string object_name)
```

Parameters

object_name [string](#)

Properties

object_name

```
public string object_name { get; init; }
```

Property Value

[string](#)

Delegate SignalHandler

Namespace: abstractions

Assembly: PIKLib.dll

Run for signals.

```
public delegate void SignalHandler(params variable[] arguments)
```

Parameters

arguments variable[]

Run for signals.

Class StringVariable

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public record StringVariable : variable, IEquatable<variable>,
IEquatable<StringVariable>
```

Inheritance

[object](#) ← [variable](#) ← StringVariable

Implements

[IEquatable](#)<[variable](#)>, [IEquatable](#)<[StringVariable](#)>

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Constructors

StringVariable(string)

```
public StringVariable(string value)
```

Parameters

`value` [string](#)

Properties

value

```
public string value { get; init; }
```

Property Value

[string](#)

Enum anchor

Namespace: abstractions

Assembly: PIKLib.dll

```
public enum anchor
```

Fields

BOTTOM = 8

CENTER = 0

LEFT = 5

LEFTLOWER = 3

LEFTUPPER = 1

RIGHT = 6

RIGHTLOWER = 4

RIGHTUPPER = 2

TOP = 7

Enum complex_operator

Namespace: abstractions

Assembly: PIKLib.dll

```
public enum complex_operator
```

Fields

AND = 0

OR = 1

Enum condition_operator

Namespace: abstractions

Assembly: PIKLib.dll

```
public enum condition_operator
```

Fields

EQUAL = 0

GREATER = 3

GREATEREQUAL = 5

LESS = 2

LESSEQUAL = 4

NOTEQUAL = 1

Enum expression_operator

Namespace: abstractions

Assembly: PIKLib.dll

```
public enum expression_operator
```

Fields

ADD = 0

DIV = 3

MOD = 4

MUL = 2

SUB = 1

Class rect

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public abstract record rect : IEquatable<rect>
```

Inheritance

[object](#) ← rect

Implements

[IEquatable](#)<[rect](#)>

Derived

[LiteralRect](#), [ReferenceRect](#)

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) ,
[object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) ,
[object.ToString()](#)

Class variable

Namespace: [abstractions](#)

Assembly: PIKLib.dll

```
public abstract record variable : IEquatable<variable>
```

Inheritance

[object](#) ← variable

Implements

[IEquatable](#)<[variable](#)>

Derived

[BoolVariable](#), [DoubleVariable](#), [IntVariable](#), [StringVariable](#)

Inherited Members

[object.Equals(object)](#) , [object.Equals(object, object)](#) , [object.GetHashCode()](#) , [object.GetType()](#) , [object.MemberwiseClone()](#) , [object.ReferenceEquals(object, object)](#) , [object.ToString()](#)