

MBS Real Studio Thread Plugin Documentation

Christian Schmitz

May 3, 2011

0.1 Introduction

This is the PDF version of the documentation for the Real Studio Plug-in from Monkeybread Software Germany. Plugin part: MBS Real Studio Thread Plugin

0.2 Content

- 1 List of all topics 3
- 2 All items in this plugin 7
- 4 List of all classes 45
- 5 List of all global methods 47

Chapter 1

List of Topics

• 2 Network	7
– 2.1 class DNSLookupThreadMBS	7
* 2.1.1 LookupAddress(domain as string) as boolean	7
* 2.1.1 LookupDomain(address as string) as boolean	8
* 2.1.2 Address as String	8
* 2.1.2 Hostname as String	9
* 2.1.2 LookupAddress as Boolean	9
* 2.1.2 LookupDomain as Boolean	9
* 2.1.2 LookupDone as Boolean	9
• 3 Process	11
– 3.1 class ThreadMBS	11
* 3.1.1 CancelAll	12
* 3.1.1 CurrentThreadID as integer	12
* 3.1.1 Delay(millisecond as integer)	12
* 3.1.1 GetIntegerMemoryValue(memAddress as integer) as integer	12
* 3.1.1 GetLockObjectCounter as int64	13
* 3.1.1 GetLockStringCounter as int64	13
* 3.1.1 GetMemoryBlockAddress(m as memoryblock) as integer	13
* 3.1.1 GetObjectLockingEnabled as boolean	13
* 3.1.1 GetStringLockingEnabled as boolean	13
* 3.1.1 GetUnlockObjectCounter as int64	14
* 3.1.1 GetUnlockStringCounter as int64	14
* 3.1.1 LockRuntime	14
* 3.1.1 LockThread	15

* 3.1.1 NotifyASync	15
* 3.1.1 NotifySync	16
* 3.1.1 NumberOfRunningThreads as integer	16
* 3.1.1 PatchedRuntimeObjectLocking as boolean	16
* 3.1.1 PatchedRuntimeStackChecking as boolean	17
* 3.1.1 PatchedRuntimeStringLocking as boolean	17
* 3.1.1 PatchRuntimeObjectLocking(DoPatch as boolean=true) as boolean	18
* 3.1.1 PatchRuntimeStackChecking(DoPatch as boolean=true) as boolean	18
* 3.1.1 PatchRuntimeStringLocking(DoPatch as boolean=true) as boolean	19
* 3.1.1 RestoreRuntimeStackChecking as boolean	19
* 3.1.1 Run as boolean	19
* 3.1.1 RunningThreads as integer	20
* 3.1.1 SetIntegerMemoryValue(memAddress as integer, value as integer)	20
* 3.1.1 SetObjectLockingEnabled(value as boolean)	20
* 3.1.1 SetStringLockingEnabled(value as boolean)	20
* 3.1.1 UnlockRuntime	20
* 3.1.1 UnlockThread	21
* 3.1.1 Wait	22
* 3.1.2 Cancelled as Boolean	22
* 3.1.2 Finished as Boolean	22
* 3.1.2 Handle as Integer	23
* 3.1.2 Lasterror as Integer	23
* 3.1.2 Running as Boolean	23
* 3.1.2 StackSize as Integer	23
* 3.1.2 ThreadID as Integer	24
* 3.1.3 Finish	24
* 3.1.3 NotifyASync	24
* 3.1.3 NotifySync	25
* 3.1.3 Prepare	25
* 3.1.3 Work	25
* 3.1.4 Available=true	28
– 3.2 Globals	28
* 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean	28
* 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean	29
* 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean	30
* 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean	31
* 3.2 CallMethodMBS(target as object, name as string) as boolean	31
* 3.2 CallMethodMBS(target as object, name as string, value1 as variant) as boolean	32

* 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean	33
* 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean	34
* 3.2 CallMethodOnMainThreadMBS(target as object, name as string) as boolean	34
* 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean	35
* 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean	36
* 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean	37
* 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean	37
* 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean	38
* 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean	39
* 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean	40
– 3.4 class MutexMBS	41
* 3.4.1 Lock	41
* 3.4.1 TryLock as boolean	42
* 3.4.1 Unlock	43
* 3.4.2 Handle as Integer	43
* 3.4.2 Tag as Variant	43

Chapter 2

Network

2.1 class DNSLookupThreadMBS

class DNSLookupThreadMBS

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** This is the class to perform a DNS lookup asynchron in a thread.

Notes:

You create a subclass of DNSLookupThreadMBS.

There you can use the finish event to perform code after the lookup was done.

In your code you create an instance of your class. You can LookupDomain(address) or LookupAddress(domain) to start the lookup.

Subclass of the ThreadMBS class.

2.1.1 Methods

LookupAddress(domain as string) as boolean

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Starts a DNS lookup for the address of the given domain name.

Notes:

Returns true on success.

Sets LookupAddress property to true.
 Sets LookupDomain property to false.
 Sets Hostname to domain.
 Sets Address to "".
 See also:

- 2.1.2 LookupAddress as Boolean

9

LookupDomain(address as string) as boolean

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Starts a DNS lookup for the domain name of the given address.

Notes:

Returns true on success.

Sets LookupAddress property to false.
 Sets LookupDomain property to true.
 Sets Hostname property to "".
 Sets Address property to address.
 See also:

- 2.1.2 LookupDomain as Boolean

9

2.1.2 Properties

Address as String

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The address.

Notes: (Read and Write property)

Hostname as String

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The domain name.

Notes: (Read and Write property)

LookupAddress as Boolean

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the current lookup is looking for the address of a domain.

Notes: (Read and Write property)

See also:

- 2.1.1 LookupAddress(domain as string) as boolean

7

LookupDomain as Boolean

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the current lookup is looking for the domain.

Notes: (Read and Write property)

See also:

- 2.1.1 LookupDomain(address as string) as boolean

8

LookupDone as Boolean

Plugin Version: 9.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether the lookup is done.

Notes: (Read and Write property)

Chapter 3

Process

3.1 class ThreadMBS

class ThreadMBS

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A class for REALbasic to do preemptive threads.

Notes:

You can use up to around 790% of CPU time on a 8 core Mac Pro with this class.
Still it requires some work to actually get things fast.
Don't get too much time wasted by synchronization.

If needed some plugin functions could be turned into ThreadMBS subclasses so they run multithreaded.
For example picture effects need to create the new picture and later to return it. But the pixel processing inbetween can run nice on a thread in background.

Warning: Plugin version 9.7 crashes if you have a bevelbutton on a window and you are using Windows.

3.1.1 Methods

CancelAll

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Cancels all running threads.

Notes: Sets the cancel property of all threads to true.

CurrentThreadID as integer

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the ID of the current thread.

Notes: You can get the value in your app.open event to know the ID of the main thread. Also you can check the current thread ID in events and callbacks to see if you are called on the main thread or another thread.

Delay(milliseconds as integer)

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Delays execution of the calling thread for the given time in milliseconds.

Notes:

You can call this from any thread, even the main thread in RB.
No events will fire in this time.

GetIntegerMemoryValue(memAddress as integer) as integer

Plugin Version: 8.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Reads the integer value on the given memory address.

Notes: Accessing memory where your application is not allowed to read will lead into a crash.

GetLockObjectCounter as int64

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the lock object counter.

Notes:

After you used PatchRuntimeObjectLocking, you wont notice the effect except of missing crashes. So this counter is increased whenever an object is locked.

GetLockStringCounter as int64

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the lock string counter.

Notes:

After you used PatchRuntimeStringLocking, you wont notice the effect except of missing crashes. So this counter is increased whenever a string is locked.

GetMemoryBlockAddress(m as memoryblock) as integer

Plugin Version: 8.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the memory address of the given memoryblock.

Notes: This value is useful for the functions SetIntegerMemoryValue and GetIntegerMemoryValue.

GetObjectLockingEnabled as boolean

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether object locking is enabled.

Notes: Works only if you use PatchRuntimeObjectLocking.

GetStringLockingEnabled as boolean

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether string locking is enabled.

Notes: Works only if you use PatchRuntimeStringLocking.

GetUnlockObjectCounter as int64

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the unlock object counter.

Notes:

After you used PatchRuntimeObjectLocking, you wont notice the effect except of missing crashes. So this counter is increased whenever an object is unlocked.

GetUnlockStringCounter as int64

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the unlock string counter.

Notes:

After you used PatchRuntimeStringLocking, you wont notice the effect except of missing crashes. So this counter is increased whenever a string is unlocked.

LockRuntime

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Locks the semaphore.

Example:

```
dim myarray(-1) As window // global
```

```
ThreadMBS.LockRuntime  
myarray.append window1  
ThreadMBS.UnlockRuntime
```

Notes:

Call only from the Work event!

You need to pair all calls to REALbasic runtime into lock and unlock to make sure you don't crash. REALbasic is not reentrant safe, so you need to lock.

Be aware that locking costs performance. You should do locks often, so in the time between two locks another thread can get a lock. Also you should group locks nearby so you don't waste too much time waiting for the lock. Finally you need your main application thread to run nice so it doesn't lock too much, too.

LockThread

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Unlocks the semaphore.

Example:

```
dim x as integer // global variable
dim o as ThreadMBS // your operation
```

```
o.LockThread
x=1
o.UnlockThread
```

Notes: This lock/unlock methods are for synchronizing the access to thread variables.

NotifyASync

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls the NotifyASync event on the main thread.

Notes:

For each call to this method, the event code will be called at most once.
(if you call this method too often, some events may not be delivered)

This method will not wait till the event has been called.
See also:

- 3.1.3 NotifyASync

NotifySync

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls the NotifySync event on the main thread.

Notes:

For each call to this method, the event code will be called once.

This method will wait till the event has been called.

See also:

- 3.1.3 NotifySync

25

NumberOfRunningThreads as integer

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The number of running ThreadMBS objects.

Notes: This property is thread safe.

PatchedRuntimeObjectLocking as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether you used PatchRuntimeObjectLocking before.

Example:

```
MsgBox "PatchedRuntimeObjectLocking: "+str(ThreadMBS.PatchedRuntimeObjectLocking)
```

```
dim b as Boolean = ThreadMBS.PatchRuntimeObjectLocking
```

```
if not b then
```

```
MsgBox "PatchRuntimeObjectLocking failed"
```

```
end if
```

```
MsgBox "PatchedRuntimeObjectLocking: "+str(ThreadMBS.PatchedRuntimeObjectLocking)
```

Notes: Returns true if patch was successful.

PatchedRuntimeStackChecking as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether you used PatchRuntimeStackChecking before.

Example:

```
MsgBox "PatchedRuntimeStackChecking: "+str(ThreadMBS.PatchedRuntimeStackChecking)
```

```
dim b as Boolean = ThreadMBS.PatchRuntimeStackChecking
```

```
if not b then
```

```
MsgBox "PatchRuntimeStackChecking failed"
```

```
end if
```

```
MsgBox "PatchedRuntimeStackChecking: "+str(ThreadMBS.PatchedRuntimeStackChecking)
```

Notes: Returns true if patch was successful.

PatchedRuntimeStringLocking as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether you used PatchRuntimeStringLocking before.

Example:

```
MsgBox "PatchedRuntimeStringLocking: "+str(ThreadMBS.PatchedRuntimeStringLocking)
```

```
dim b as Boolean = ThreadMBS.PatchRuntimeStringLocking
```

```
if not b then
```

```
MsgBox "PatchRuntimeStringLocking failed"
```

```
end if
```

```
MsgBox "PatchedRuntimeStringLocking: "+str(ThreadMBS.PatchedRuntimeStringLocking)
```

Notes: Returns true if patch was successful.

PatchRuntimeObjectLocking(DoPatch as boolean=true) as boolean

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Patches the internal functions in REALbasic for the reference counting of objects.

Notes:

Once applied all object reference count changes will be locked with a mutex.

This patch should avoid a lot of potential crashes in preemptive threads and should have no side effects.

If an object is destroyed, the destructor will be run on the current thread which may lead into crashes (if not being called on the main thread).

For testing only the DoPatch parameter can be set to false to not apply the patch (but prepare it).

Note: Mutexes to lock shared data can slow down thread performance.

Do only call this if you need to as we are never sure this method has no bad side effects.

PatchRuntimeStackChecking(DoPatch as boolean=true) as boolean

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Disables the StackOverflowExceptions in your whole application.

Notes:

Works on PPC and Intel computers.

For testing only the DoPatch parameter can be set to false to not apply the patch (but prepare it).

Do only call this if you need to as we are never sure this method has no bad side effects.

PatchRuntimeStringLocking(DoPatch as boolean=true) as boolean

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Patches the internal functions in REALbasic for the reference counting of strings.

Notes:

Once applied all string reference count changes will be locked with a mutex.
This patch should avoid a lot of potential crashes in preemptive threads and should have no side effects.

There is a mutex used to lock calls to the

For testing only the DoPatch parameter can be set to false to not apply the patch (but prepare it).

Note: Mutexes to lock shared data can slow down thread performance.

Do only call this if you need to as we are never sure this method has no bad side effects.

RestoreRuntimeStackChecking as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Reverses the patch from PatchedRuntimeStackChecking.

Notes: Returns true on success and false on failure.

Run as boolean

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Starts the thread.

Notes:

Returns true on success.

There is a limit for the number of threads, so don't make more than 100 threads at the same time.

Run fails if:

- platform is not supported (e.g. Mac Carbon PEF or Classic)
- the internal list of threads is full.
- the Work event is empty

- the OS can't create more threads (in that case the Prepare event has been called, but Work and Finished events are not called)

RunningThreads as integer

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the number of running threads.

Notes: The plugin keeps a list of the running threads and this gives you the count.

SetIntegerMemoryValue(memAddress as integer, value as integer)

Plugin Version: 8.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Sets the integer value on the given memory address.

Notes: Accessing memory where your application is not allowed to write will lead into a crash.

SetObjectLockingEnabled(value as boolean)

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Enables or disabled thread safe object locking.

Notes: Works only if you use PatchRuntimeObjectLocking.

SetStringLockingEnabled(value as boolean)

Plugin Version: 8.2 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Enables or disabled thread safe string locking.

Notes: Works only if you use PatchRuntimeStringLocking.

UnlockRuntime

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Unlocks the semaphore.

Example:

```
dim myarray(-1) As window // global
```

```
ThreadMBS.LockRuntime  
myarray.append window1  
ThreadMBS.UnlockRuntime
```

Notes:

Call only from the Work event!

You need to pair all calls to REALbasic runtime into lock and unlock to make sure you don't crash. REALbasic is not reentrant safe, so you need to lock.

Be aware that locking costs performance. You should do locks often, so in the time between two locks another thread can get a lock. Also you should group locks nearby so you don't waste too much time waiting for the lock. Finally you need your main application thread to run nice so it doesn't lock too much, too.

UnlockThread

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Unlocks the thread semaphore.

Example:

```
dim x as integer // global variable  
dim o as ThreadMBS // your operation
```

```
o.LockThread  
x=1  
o.UnlockThread
```

Notes: This lock/unlock methods are for synchronizing the access to thread variables.

Wait

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Waits for this thread to finish.

Notes: You should not use this in a GUI application as it will lock the GUI.

3.1.2 Properties**Cancelled as Boolean**

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this thread is cancelled.

Example:

```
dim i as integer
dim t as ThreadMBS // your thread

for i=1 to 1000
if t.cancelled then
return
end if
// do work
next
```

Notes:

You can set this flag and query in your loops to check whether the thread has been cancelled.

This property is thread safe.

(Read and Write property)

Finished as Boolean

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this thread has finished.

Notes:

Access to this property is thread safe.

(Read only property)

Handle as Integer

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal used thread handle.

Notes:

A Windows thread handle or a pthread handle on Mac OS X or Linux.
(Read and Write property)

Lasterror as Integer

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The last error code reported by one of the functions.

Notes:

Zero is no error.
(Read and Write property)

Running as Boolean

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether this thread has running.

Notes:

Access to this property is thread safe.
(Read only property)

StackSize as Integer

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The stack size to use for creating the thread.

Notes:

Default is 0 which means to take the system default.
(Read and Write property)

ThreadID as Integer

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The thread ID of the thread used.

Notes:

Only used on Windows.
(Read and Write property)

3.1.3 Events

Finish

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The event called after the thread has finished.

Notes: This event is called in the main thread, so you can do all REALbasic functions without locking the runtime.

NotifyASync

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The asynchron notification event.

Notes:

Called on the main thread.
The method NotifyASync will not wait till this event has finished.
See also:

- 3.1.1 NotifyASync

NotifySync

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The synchron notification event.

Notes:

Called on the main thread.

The method NotifySync will wait till this event has finished.

See also:

- 3.1.1 NotifySync

16

Prepare

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The event called before the thread is started.

Notes: This event is called in the main thread, so you can do all REALbasic functions without locking the runtime.

Work

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The work event.

Notes:

This work is running in its own thread. So there are a lot of limitations:

1. Disable Stackchecking and background tasks as their usage will crash your applications:

```
# pragma DisableAutoWaitCursor
# pragma DisableBackgroundTasks
# pragma StackOverflowChecking false
```

You need to use this pragmas in all methods you call from the thread code.

If you use PatchRuntimeStackChecking, this should be no problem.

2. Don't create or destroy objects

Creating or destroying objects from a thread will crash.

One reason is that Constructors/Destructors are not thread safe.

The other one is that the REALbasic memory management is not thread safe.

3. Notify

You can call NotifySync and NotifyAsync to get an event in main thread to do whatever code you like.

Also the Prepare and Finish events are in the main thread so you can do there whatever you like.

4. No exceptions

Raising exceptions will create objects which is not thread safe.

Also unhandled exceptions will display a msgbox which will crash.

5. no GUI

You can't use any code which refers to the graphical user interface.

6. String functions

Calling string functions like str() or concat (a+b) seems to work.

7. Dictionary

using the dictionary class is not allowed as the hash functions of the wrapper classes are not thread safe.

8. LockRuntime

If you want to call a REALbasic runtime functions, you can avoid crashes by locking the runtime.

LockRuntime and UnlockRuntime will make sure that no code in the main thread is executed between both

methods.

LockRuntime
CallSomeMethod
UnlockRuntime

9. Pictures

Functions which return pictures will crash as they call newpicture internally which will crash.
(See 2)

10. Synchronize access to global variables

If you access properties of a Module, you need to synchronize the access to this properties with a mutex.
Or you use LockRuntime/UnlockRuntime.

11. Reference counting.

If you access objects, REALbasic will increase and later decrease the reference count. That seems to be no big problem unless you threads have the reference count changed at the same time for the same object or a reference count decrease will destroy that object.

You should avoid object access. Also you can avoid access from multiple threads if you make copies of objects for each thread.

Of course you can use RuntimeLock and RuntimeUnlock.

12. Breakpoints

Breakpoints work between RuntimeLock and RuntimeUnlock. Else not.

13. Calls to the operation system work if they are thread safe.

On Mac OS X malloc and free work (To allocate memory).
Also System.DebugLog works.

13. Window access

Do not access windows from REALbasic. That will crash.

3.1.4 Constants

Available=true

Plugin Version: 8.1 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A constant which tells you whether preemptive threads are available.

Notes: On Carbon PEF and Classic false. True on Linux, Windows and Mac OS X Carbon MachO.

3.2 Globals

CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread after the given delay in seconds.

Example:

```
if CallMethodLaterMBS(window1, "Test", 5.0) then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with no parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread after the given delay in seconds.

Returns true on success and false on failure.

The time given is just a roughly suggestions. Actual time on the method call depends on how busy your application is.

See also:

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean 29
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean 30
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean 31

CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread after the given delay in seconds.

Example:

```
if CallMethodLaterMBS(window1, "Test", 4.0, "Hello") then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with one variant parameter and no return values.

The method will be called later (Asynchronously) on the main thread after the given delay in seconds. Useful for performing non thread safe stuff like GUI functions on the main thread.

Returns true on success and false on failure.

The time given is just a roughly suggestions. Actual time on the method call depends on how busy your application is.

See also:

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean 28

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean 30
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean 31

CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread after the given delay in seconds.

Example:

```
if CallMethodLaterMBS(window1, "Test", 3.0, "Hello", "World") then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with two variant parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread after the given delay in seconds.

Returns true on success and false on failure.

The time given is just a roughly suggestions. Actual time on the method call depends on how busy your application is.

See also:

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean 28
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean 29
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean 31

CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread after the given delay in seconds.

Example:

```
if CallMethodLaterMBS(window1, "Test", 3.0, "Hello", "World", 5) then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with three variant parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread after the given delay in seconds.

Returns true on success and false on failure.

The time given is just a roughly suggestions. Actual time on the method call depends on how busy your application is.

See also:

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean 28
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean 29
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean 30

CallMethodMBS(target as object, name as string) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object.

Example:

```

if CallMethodMBS(window1, "Test") then
msgbox "OK"
else
msgbox "Failed"
end if

```

Notes:

The method must be declared on the given class for the target object with no parameters and no return values.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodMBS(target as object, name as string, value1 as variant) as boolean 32
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 33
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 34

CallMethodMBS(target as object, name as string, value1 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object.

Example:

```

if CallMethodMBS(window1, "Test", "Hello") then
msgbox "OK"
else
msgbox "Failed"
end if

```

Notes:

The method must be declared on the given class for the target object with one variant parameter and no return values.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodMBS(target as object, name as string) as boolean 31
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 33
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 34

CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object.

Example:

```
if CallMethodMBS(window1, "Test", "Hello", "World") then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with two variant parameters and no return values.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodMBS(target as object, name as string) as boolean 31
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant) as boolean 32
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 34

CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object.

Example:

```
if CallMethodMBS(window1, "Test", "Hello", "World", 5) then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with three variant parameters and no return values.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodMBS(target as object, name as string) as boolean 31
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant) as boolean 32
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 33

CallMethodOnMainThreadMBS(target as object, name as string) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread.

Example:

```
if CallMethodOnMainThreadMBS(window1, "Test") then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with no parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean 35
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 36
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 37

CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread.

Example:

```
if CallMethodOnMainThreadMBS(window1, "Test", "Hello") then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with one variant parameter and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnMainThreadMBS(target as object, name as string) as boolean 34
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 36
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 37

CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread.

Example:

```
if CallMethodOnMainThreadMBS(window1, "Test", "Hello", "World") then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with two variant parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnMainThreadMBS(target as object, name as string) as boolean 34
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean 35
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 37

CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on the main thread.

Example:

```
if CallMethodOnMainThreadMBS(window1, "Test", "Hello", "World", 5) then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with three variant parameters and no return values.

The method will be called later (Asynchronously) on the main thread. Useful for performing non thread safe stuff like GUI functions on the main thread.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnMainThreadMBS(target as object, name as string) as boolean 34
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean 35
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 36

CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on a new thread.

Example:

```
if CallMethodOnThreadMBS(new BackgroundThreadMBS, window1, "Test") then
  msgbox "OK"
```

```

else
msgbox "Failed"
end if

```

Notes:

The method must be declared on the given class for the target object with no parameters and no return values.

Pass in "new BackgroundThreadMBS" for the thread to use. Execution will be done later (Asynchronously) on that thread. Useful for perform some code in the background without creating a thread yourself.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean 38
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean 39
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 40

CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on a new thread.

Example:

```

if CallMethodOnThreadMBS(new BackgroundThreadMBS, window1, "Test", "Hello") then
msgbox "OK"
else
msgbox "Failed"
end if

```

Notes:

The method must be declared on the given class for the target object with one variant parameter and no

return values.

Pass in "new BackgroundThreadMBS" for the thread to use. Execution will be done later (Asynchronously) on that thread. Useful for perform some code in the background without creating a thread yourself.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean 37
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean 39
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 40

CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on a new thread.

Example:

```
if CallMethodOnThreadMBS(new BackgroundThreadMBS, window1, "Test", "Hello", "World") then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with two variant parameters and no return values.

Pass in "new BackgroundThreadMBS" for the thread to use. Execution will be done later (Asynchronously) on that thread. Useful for perform some code in the background without creating a thread yourself.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean 37
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean 38
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 40

CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Calls a method on the target object on a new thread.

Example:

```
if CallMethodOnThreadMBS(new BackgroundThreadMBS, window1, "Test", "Hello", "World", 5) then
  msgbox "OK"
else
  msgbox "Failed"
end if
```

Notes:

The method must be declared on the given class for the target object with three variant parameters and no return values.

Pass in "new BackgroundThreadMBS" for the thread to use. Execution will be done later (Asynchronously) on that thread. Useful for perform some code in the background without creating a thread yourself.

Returns true on success and false on failure.

See also:

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean 37
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean 38

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean 39

3.3 class BackgroundThreadMBS

class BackgroundThreadMBS

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The thread subclass we use for the CallMethodOnThreadMBS functions.

Example:

```
if CallMethodOnThreadMBS(new BackgroundThreadMBS, window1, "Test") then
msgbox "OK"
else
msgbox "Failed"
end if
```

Notes: Subclass of the Thread class.

3.4 class MutexMBS

class MutexMBS

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A class for a mutex.

Notes: This is the mutex class the threadMBS class is using internally.

3.4.1 Methods

Lock

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Locks the mutex.

Example:

```
dim m as new MutexMBS

m.Lock
MsgBox "Got mutex."

m.Unlock
MsgBox "Released mutex."
```

Notes:

The function returns as soon as it has access to the mutex.

Always use Lock and Unlock in a pair.

TryLock as boolean

Plugin Version: 10.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Locks the mutex if possible.

Example:

```
dim m as new MutexMBS

if m.TryLock then
MsgBox "Got mutex."

m.Unlock
else
MsgBox "Failed to get mutex."
end if
```

Notes:

Returns true if we got a lock and false if not.
Always use Lock and Unlock in a pair.

Unlock

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Unlocks the mutex.

Example:

```
dim m as new MutexMBS

m.Lock
MsgBox "Got mutex."

m.Unlock
MsgBox "Released mutex."
```

Notes: Always use Lock and Unlock in a pair.

3.4.2 Properties

Handle as Integer

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The internal used reference to the native mutex object.

Example:

```
dim m as new MutexMBS
MsgBox "Handle: " + hex(m.Handle)
```

Notes:

Windows Mutex or PThread Mutex.
(Read and Write property)

Tag as Variant

Plugin Version: 8.1 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** An object reference for your use.

Notes:

Just for convenience.

This property was added so you can use this property for a reference so you won't need to subclass this class and add custom properties.

(Read and Write property)

Chapter 4

List of all classes

• BackgroundThreadMBS	41
• DNSLookupThreadMBS	7
• MutexMBS	41
• ThreadMBS	11

Chapter 5

List of all global methods

- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double) as boolean 28
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant) as boolean 29
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant) as boolean 30
- 3.2 CallMethodLaterMBS(target as object, name as string, afterDelay as double, value1 as variant, value2 as variant, value3 as variant) as boolean 31
- 3.2 CallMethodMBS(target as object, name as string) as boolean 31
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant) as boolean 32
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 33
- 3.2 CallMethodMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 34
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string) as boolean 34
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant) as boolean 35
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant) as boolean 36
- 3.2 CallMethodOnMainThreadMBS(target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 37
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string) as boolean 37

- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant) as boolean 38
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant) as boolean 39
- 3.2 CallMethodOnThreadMBS(BackgroundThread as BackgroundThreadMBS, target as object, name as string, value1 as variant, value2 as variant, value3 as variant) as boolean 40