

# Scripting Functions Documentation

- [New NightFall functions](#)
- [Slightly modified reborn functions](#)
- [Reborn original functions](#)

---

## New NightFall functions

---

### register\_api\_route

```
register_api_route(string uri, string method, string/array scriptname)
```

Registers callback handler for give api route/method.

Current allowed values for `method` is `"get"`.

See [api\\_server.md](#) for server api configuration settings and more info.

Example:

```
local.result = register_api_route "/" "get" global/api.scr:get_handler
```

Result:

When specified uri/method route is called/requested, api will call the specified callback script.  
The api response is the value returned by script.

local.result will have one of the following values:

```
0 = Route registered successfully
1 = Route already registered
2 = Invalid arguments
3 = Server api is not running
4 = Server api error
```

## unregister\_api\_route

```
unregister_api_route(string uri, string method)
```

Un-registers callback handler for give api route/method.

Current allowed values for `method` is `"get"` .

See [api\\_server.md](#) for server api configuration settings and more info.

Example:

```
local.result = unregister api route "/" "get"
```

Result:

Removes api callback handler for specified uri/method route.

local.result will have one of the following values:

- 0 = Route unregistered successfully
- 1 = Route is already not registered
- 2 = Invalid arguments
- 3 = Server api is not running
- 4 = Server api error

---

## create\_api\_request

```
create_api_request(string url, string method, string/array scriptname, var userdata)
```

Creates an http request from the server to url/method combination.

Calls script after request is done.

Url may contain query string.

Current allowed values for `method` is `"get"` .

See [api\\_client.md](#) for client api configuration settings and more info.

Example:

```
local.result = create_api_request  
"http://jsonplaceholder.typicode.com/todos/1" "get"  
global/api.scr::request handler "my own data"
```

Result:

Creates an http request from the server to url/method combination.

local.result will have one of the following values:

- 0 = Request queued successfully
- 1 = Invalid arguments
- 2 = Client api is not running
- 3 = Client api error

## constarray

```
constarray(array var)
```

Cast existing array to constarray.

A constarray variable's size cannot be changed, has index starting from 1.

Example:

```
local.arr[1] = xtesddfgsf; //local.arr is a hashtable array
local.arr[2] = 2222; //local.arr is a hashtable array
local.result = constarray local.arr //local.result is a constarray
//above code is equivalent to
local.result = xtesddfgsf::2222
```

Result:

local.result will have one of the following values:

Nil if variable can't be converted.  
Proper constarray value otherwise.

---

## regex\_parse

```
regex_parse(string pattern, string search_string, boolean whole_match)
```

Perform a regular expression search using pattern on string search\_string.

If whole\_match is set, parser will fail if whole string doesn't match the regex pattern.

Example:

```
local.result = regex_parse "a(a)*b" "baaaby" 0
```

Result:

local.result will be an array having the following structure:

local.result[success]: integer, 1 if regex is successfull, 0 if not.  
local.result[matches]: array of strings containing capture group matches.

```
local.result[success] = 1
local.result[matches][0] = aaab
local.result[matches][1] = a
```

This option uses standard c++ library regex parser, regex\_match is used when `whole_match` is 1 otherwise regex\_search is used. Visit [cppreference](#) for more info.

## json\_parse

```
json_parse(string json_string)
```

Parse given `json_string` into a variable.

Example:

```
local.result = json_parse "{ \"happy\": true, \"pi\": 3.141 }"
```

Result:

```
local.result will be an array having the following structure:  
local.result.size = 2  
local.result[happy]: 1  
local.result[pi]: 3.141
```

---

## json\_stringify

```
json_stringify(variable var)
```

Convert given `var` into a json string.

Example:

```
local.result = json_stringify "{ \"happy\": true, \"pi\": 3.141 }"
```

Result:

```
local.result = json_stringify aaa::bb::cc  
local.result =  
{\"content\":[{\"content\":\"aaa\",\"type\":\"string\"},{\"content\":\"bb\",\"type\":\"string\"},{\"content\":\"cc\",\"type\":\"string\"}],\"type\":\"array\"}
```

---

## Slightly modified reborn functions

---

### fcopy

```
fcopy(string filename, string copyname)
```

Creates a copy of file `filename` into `copyname` .

Example:

```
local.result = fcopy local.filename local.copyname
```

Result:

```
-1 = error occurred  
0 = copy operation succeeded
```

## freadall

```
freadall(integer filehandle)
```

Reads whole file into a string at once.

File **does NOT need** to be opened in binary mode (rb, rb+).

Example:

```
local.content = freadall local.file
```

Result:

```
Function returns file content as string.
```

---

## getdate

```
getdate(integer zero) (1)  
getdate(string format) (2)
```

(1) Gets current date in format: dd.mm.yyyy

(2) Gets current date in format given as parameter.

Example:

```
local.date1 = getdate 0  
local.date2 = getdate "%D"
```

Result:

```
String with current date.  
  
local.date1 = "23.08.2001"  
local.date2 = "08/23/01"
```

---

## registerev

```
registerev(string eventname, string/array scriptname)
```

Registers script callback handler for given event type.

**NOTE:** Event behaviour is slightly different from reborn and more events are added. See [eventsystem.md](#) for more info. Example:

```
local.result = registerev "connected" global/eventhandlers.scr::connected
```

Result:

When given event type will occur, EventSystem engine will execute given script.

local.result can have one of the following values:

```
0 = Registering event callback handler was successful
1 = Event callback handler is already registered for given event
3 = Invalid script passed
```

---

## Reborn original functions

---

For the sake of not re-inventing the wheel, the rest of the functions in NightFall are exactly identical to reborn ones.