UOSteam

Scripting Documentation

Diego Alcântara
Diogo Palma

2013

Summary

- Introduction
    - Syntax
    - Object Inspector

- Commands
    - Abilities
    - Actions
    - Agents
    - Aliases
    - Conditions
    - Gumps
    - Journal
    - Lists
    - Main
    - Others
    - Spells
    - Targeting
    - Timers

Introduction

Syntax

UOSteam scripting language is a "command based" language, it is easy to use and requires very basic programming knowledge, its power and flexibility is given by its commands, which can be found in this documentation.

In this section we will show you its syntax, explaining symbols and structures.

Commands and Parameters

As mentioned before this is a "command based" language, that being said understanding how to run commands is an important step. This documentation provides you a list of every command avaliable on UOSteam, it will also show you its supported parameters, for example:

Usage

```
pushlist ('list name') ('element value') ['front'/'back']
```

Note that all parameters are shown inside of parenthesis or brackets. What does it mean? It is simple, all parameters inside parenthesis are **mandatory**, required for the command to be executed, on the other hand, all parameters inside brackets are **optional**. The slashes "/" inside of a parameter means **or**, they are **not** part of a feasible parameter value. You also must respect the parameters order, check out the samples below:

Incorrect

```
pushlist
pushlist 'fruits'
pushlist ('fruits') ('apple')
pushlist 'fruits' 'apple' 'front/back'
pushlist 'fruits' star fruit
pushlist 'fruits' 'japan's melon' 'front'
```

Correct

```
pushlist 'fruits' 'apple'
pushlist 'fruits' grape 'front'
pushlist 'fruits' 'lemon' 'back'
pushlist 'fruits' 'star fruit'
pushlist 'fruits' "japan's melon" 'front'
```

Important: text parameters can be written without quotes or inside of single quotes and double quotes; be careful because if you want to pass as parameter that has aphostrophe you **must** use double quotes, otherwise single quotes are just fine. In case it is a compound text such as "star fruit" make sure quotes are being used otherwise you'd be passing "star" as a parameter and "fruit" as another. Always using double quotes is an advised best practice.

Loops and Conditions

Check below our supported loops and conditional statements structures.

Structures
if (statement)
elseif (statement)
else
endif
while (statement)
endwhile
for (value)
endfor
for (start) to (end)
endfor
for (start) to ('list name')
endfor
for (start) to (end) in ('list name')
endfor

Keywords
break
continue
stop
replay
not (statement)
(statement) and (statement)
(statement) or (statement)

Lets suppose you want to say "I love UOSteam!" 10 times. How can we code that? We would have to use a "for" because that way we can define how many times we want the code to repeat:

Code

```
// Repeat 10 times
for 10
   // Send a lovely message in game
   msg "I love UOSteam!"
   pause 1500
endfor
sysmsg "End of for loop"
```

Now lets take a better look at the script above... first is just a comment line, you are able to comment your code anytime by adding a "//" prefix before your comment, remember that all comments require a new line! The second line starts our "for" loop and specify how many times we want this to repeat, in this case 10. Third line is also a comment, line 4 has a msg command that sends a message in game, passing "I love UOSteam!" as parameter and line 5 has a pause, because we have to breath and spamming is bad, note that we pass 1500 as our first parameter, pause command requires a time parameter in milliseconds so we are actually pausing for 1.5 seconds. Last line sysmsg command will print a system message letting we know the loop has ended.

Wait, I love UOSteam much more, I wanna say that "while" I'm alive:

Code

```
// While I'm alive
while not dead
   // Send a lovely message in game
   msg "I love UOSteam!"
   pause 1500
endwhile
sysmsg "End of while loop"
```

Now it will keep repeating until your character is dead instead of repeating only 10 times, pay attention to the statement: "not dead" combined by dead conditional command and "not" keyword. What if I'm sick? I can't scream how much I love UOSteam so I want to stop saying that and cure myself first:

Code

```
// While I'm alive
while not dead
   if poisoned 'self'
      // I'm poisoned! Try to cure.
      cast 'Cure'
      waitfortarget 5000
      target 'self'
      break
      // Unreachable code
   endif
   // Send a lovely message in game
   msg "I love UOSteam!"
   pause 1500
endwhile
sysmsg "End of while loop"
```

The keyword "break" will quit our "while" loop and sysmsg command will be executed because it is outside the "while" loop, note that waitfortarget command expects a timeout parameter also in milliseconds, in this case 5 seconds. We have added to our script a conditional statement using poisoned command.

Use keyword "stop" to completely stop a macro and "replay" to restart playing your current macro.
In case you want to play a different macro look for playmacro command.

Symbols

There are currently 2 supported symbols that affects our syntax:

Symbols
@
!

First symbol "at" is a **prefix** and it is very useful for silencing commands warnings or outputs, for example:

Code
```
// Search for an ettin in range of 10 tiles
// Suppress system warnings
if @findtype 0x12 0 0 0 10
   attack 'found'
endif
```

The conditional of the script above will not return any output if an object of this type is not found, in case you remove the "at" warnings are no longer suppressed and a system message saying "findtype: type not found." will appear if findtype command is not able to locate an object matching given parameters.

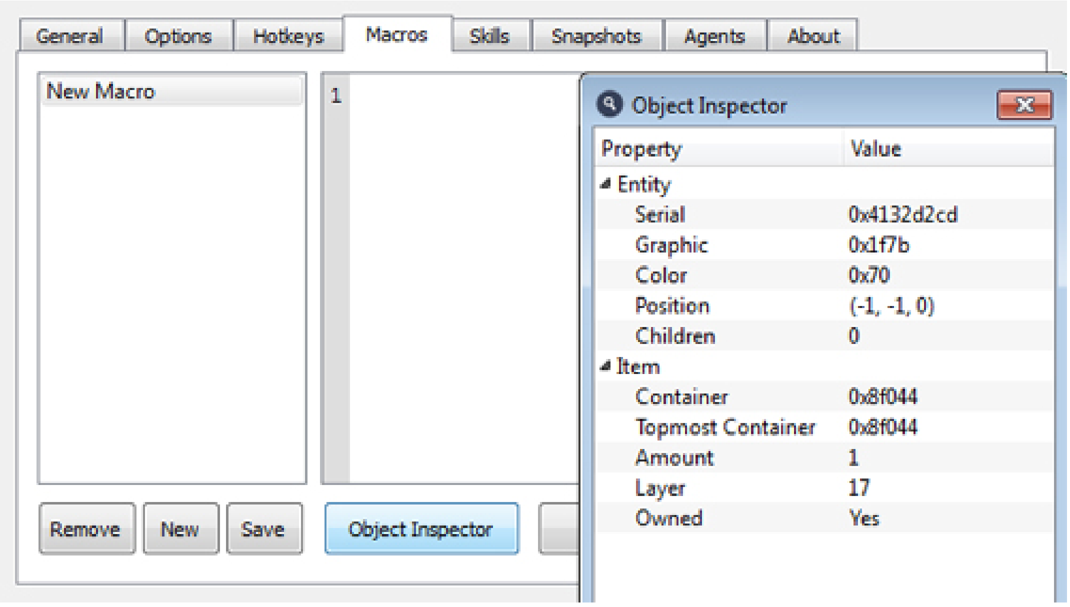
Second symbol "exclamation" is a **suffix** and its usage is related to its command, for example on targeting functions it disables targeting queue. If a command supports that symbol you'll be able to verify the effects on its description.

Code
```
cast 'Heal'
waitfortarget 500
// Disable targeting queue
target! 'self'
```

Object Inspector

At this point you should be able to understand better how scripts are written in UOSteam, before trying out you must learn how to locate correctly object commands parameters.

What is an object?
An object is essentially a game item or mobile. Some commands will require object properties as parameters, there is an "Object Inspector" button on "Macros" tab, by pressing this button a target will appear in game so you can choose an object, a new window will appear showing you its properties:



Property	Value
◢ Entity	
Serial	0x4132d2cd
Graphic	0x1f7b
Color	0x70
Position	(-1, -1, 0)
Children	0
◢ Item	
Container	0x8f044
Topmost Container	0x8f044
Amount	1
Layer	17
Owned	Yes

Aliases
An alias is a name that you can assign to an object serial in order to make it easier to create scripts. Aliases will remain saved on your current profile, check the sample below:

Code

```
// Search for an object called 'Scissors'
if not @findobject 'Scissors'
   // Scissors does not exist, prompt for a new one in game
   promptalias 'Scissors'
endif
useobject 'Scissors'
```

Now the alias "Scissors" can be used in any command that requires a "serial" parameter, that way we do not have to insert the serial number. The sample below is also a valid approach for using aliases:

Code

```
// Set 'Scissors' alias value
setalias 'Scissors' 0x28e9b2d1
useobject 'Scissors'
```

Commands

Abilities

Fly and Land

Start or stop flying.

Usage

```
// Start
fly
// Stop
land
```

Set Ability

Toggle or enforce primary, secondary, stun or disarm ability.

Usage

```
setability ('primary'/'secondary'/'stun'/'disarm') ['on'/'off']
```

Sample

```
// Primary
// Prefix '@' to disable warnings
@setability 'primary'
// Secondary
setability 'secondary'
// Stun (Pre-AOS)
setability 'stun'
// Disarm (Pre-AOS)
setability 'disarm'
// Enforce primary ability on
setability 'primary' 'on'
// Enforce secondary ability off
@setability 'secondary' 'off'
```

Actions

Attack

Attack a specific mobile serial or alias.

Usage

```
attack (serial)
```

Sample

```
// Search for an ettin in range of 5 tiles
if findtype 0x12 0 0 0 5
   autotargetobject 'found'
   // Use virtue honor
   virtue 'Honor'
   // Attack
   attack 'found'
endif
```

Clear Hands

Unequip character's hand.

Usage

```
clearhands ('left'/'right'/'both')
```

Sample

```
// Try to clear right hand
if not clearhands 'right'
   sysmsg 'Unable to clear hands, item not found.'
else
   sysmsg 'Right hand cleared.'
endif
```

Click Object

Perform a single click on a specific serial.

Usage

```
clickobject (serial)
```

Sample

```
// Search for a house sign
if findtype 0xbd2 0 'ground'
   clickobject 'found'
endif
```

Bandage Self

Shortcut to use bandage type and automatically target self.

Usage

```
bandageself
```

Sample

```
if not timerexists 'bandageSelf'
   // You can use createtimer function, settimer create and set a value
   settimer 'bandageSelf' 2000
endif
if hits != maxhits
   // Check if timer elpased 2 seconds
   if timer 'bandageSelf' >= 2000
      bandageself
   else
      // Reset timer
      settimer 'bandageSelf' 0
   endif
   // Check if poisoned
   if poisoned 'self'
      usetype! 0xf07
   endif
endif
```

Use Type

Use a specific item type (graphic).

Usage

```
// Trigger
usetype (graphic) [color] [source] [range]
// Cleaner
clearusequeue
```

Sample

```
// Use any bandage color from a specific container
// Suffix '!' to avoid queue
usetype! 0xe21 'any' 0x40116650
// Queued use of any bandage color from the ground in range of 2 tiles
usetype 0xe21 'any' 'ground' 2
// Use type from backpack if hits differs from max hits
if hits != maxhits
  // Prefix '@' to disable system warnings
  if @usetype! 0xe21
    waitfortarget 800
    target 'self'
  endif
endif
```

Use Object

Use a specific object serial or alias.

Usage

```
// Trigger
useobject (serial)
// Cleaner
clearusequeue
```

Sample

```
// Use a specific object by serial
// Suffix '!' to avoid queue
useobject! 0x40116650
// Use object by name
if not findalias 'myObject'
  sysmsg 'Select a new object.'
  setalias 'myObject'
  while not findalias 'myObject'
    if not targetexists 'system'
      stop
    endif
  endwhile
endif
// Queue and use named object
useobject 'myObject'
```

Use Once

Use a specific item type (graphic) from your backpack, only once.

Usage

```
// Trigger
useonce (graphic) [color]
// Cleaner
clearuseonce
```

Sample

```
// Use any pouch color from backpack
// Suffix '!' to avoid queue
useonce! 0xe79 'any'
```

Move Item

Move an item serial or type from source to destination.

Usage

```
moveitem (serial) (destination) [(x, y, z)] [amount]
moveitemoffset (serial) 'ground' [(x, y, z)] [amount]
movetype (graphic) (source) (destination) [(x, y, z)] [color] [amount] [range]
movetypeoffset (graphic) (source) 'ground' [(x, y, z)] [color] [amount] [range]
```

Sample

```
// Move specific item to backpack
moveitem 0x40116650 'backpack'
// Move righthand item to backpack
moveitem 'righthand' 'backpack'
// Move 100 gold from backpack to a ground location
// Use '!' to disallow stacking
movetype! 0xeed 'backpack' 'ground' 1950 50 0
// Move 200 gold from the ground in range of 2 tiles to backpack
movetype 0xeed 'ground' 'backpack' 0 0 0 'any' 200 2
// It can be used as a statement
if moveitem 'righthand' 'backpack'
   sysmsg 'Right hand item moved to backpack!'
endif
```

Movement

Move your character to the given direction(s).

Usage

```
walk (direction)
turn (direction)
run (direction)
```

Sample

```
// Multiple directions
walk "North, East, East, West, South, Southeast"
// Single direction
turn "Northeast"
// Run 10 tiles to south
for 10
  run "South"
endfor
```

Use Skill

Use a skill by name.

Usage

```
useskill ('skill name'/'last')
```

Sample

```
if hits != maxhits
  if yellowhits 'self'
    useskill 'Spirit Speak'
  elseif not poisoned 'self'
    cast 'Heal' 'self'
  else
    cast 'Cure' 'self'
  endif
endif
```

War Mode

Enable and disable war mode.

Usage

```
warmode ('on'/'off')
```

Feed

Feed a given alias or serial with food name, graphic or group.

Usage

```
/*
Groups:
- Fish: Fish Steak, Raw Fish Steak;
- Fruits and Vegetables: Honeydew Melon, Yellow Gourd, Green Gourd, Banana, Lemon, Lime, Grape, Peach, Pear, Apple, Watermelon, Squash, Cantaloupe, Carrot, Cabbage, Onion, Lettuce, Pumpkin;
- Meat: Bacon, Cooked Bird, Sausage, Ham, Ribs, Lamb Leg, Raw Bird, Raw Ribs, Raw Lamb Leg, Raw Chicken Leg, Head, Left Arm, Left Leg, Torso, Right Arm, Right Leg.
*/
feed (serial) ('food name'/'food group'/'any'/graphic)
```

Sample

```
// Feed mount with any type of fruits or vegetables
feed 'mount' 'Fruits and Vegetables'
// Feed mount with a raw ribs
feed 'mount' 'Raw Ribs'
// Feed player using fish steak graphic
feed 'self' 0x97B
```

Rename

Request the server to rename a mobile.

Usage
rename (serial) ('name')

Sample

```
// Rename mount to Snorlax
rename 'mount' 'Snorlax'
```

Show Names

Display corpses and/or mobiles names.

Usage
shownames ['mobiles'/'corpses']

Sample

```
// Display mobiles names
shownames 'mobiles'
// Display corpses names
shownames 'corpses'
```

Toggle Hands

Arm and disarm an item.

Usage

```
togglehands ('left'/'right')
```

Sample

```
// Equip an item to the right hand layer
// Press one time to arm
togglehands 'right'
// Press another time to disarm
```

Equip Item

Equip a specific item into a given layer.

Usage

```
equipitem (serial) (layer)
```

Sample

```
// Prompt for a new bow
promptalias 'bow'
// Check if selected bow exist
if @findobject 'bow'
   equipitem 'bow' 2
endif
```

Toggle Mounted

Mount and dismount.

Usage

```
togglemounted
```

Equip Wand

Search for a wand inside your backpack and equip it.

Usage

```
/*
Spells:
- Clumsy;
- Identification;
- Heal;
- Feeblemind;
- Weaken;
- Magic Arrow;
- Harm;
- Fireball;
- Greater Heal;
- Lightning;
- Mana Drain.
*/
equipwand ('spell name'/'any'/'undefined') [minimum charges]
```

Sample

```
// Equip heal wand from backpack, with a minimum of 5 charges
equipwand 'Heal' 5
```

Agents

Vendors

Execute buy or sell list from vendors agent.

Usage

```
// Triggers
buy ('list name')
sell ('list name')
// Cleaners
clearbuy
clearsell
```

Organizer

Execute a specific organizer profile.

Usage

```
organizer ('profile name') [source] [destination]
```

Sample

```
// Check if you are already organizing
if not organizing
    organizer 'Reagents'
endif
```

Autoloot

Prompt a cursor to autoloot a specific corpse or container.

Usage

```
autoloot
```

Dress and Undress

Dress or undress temporary or specific profile.

Usage

```
// Profile
dress ['profile name']
undress ['profile name']
// Temporary
dressconfig
```

Sample

```
// Toggle weapon
if not @findobject 'righthand'
   dress 'Weapon'
else
   undress 'Weapon'
endif
// Undress all equipments no matter the profile
undress
```

Toggle Autoloot

Enable and disable autoloot agent.

Usage
toggleautoloot

Toggle Scavenger

Enable and disable scavenger agent.

Usage
togglescavenger

Counter

Compare the amount of a specific counter format.

Usage

```
counter ('format') (operator) (value)
```

Sample

```
// Check for amount of garlics
if counter 'gc' == 0
   sysmsg 'Out of garlic!' 25
endif
// Craft loop
while counter 'kilt' != 10
   // Use sewing kit by graphic
   if usetype! 0xf9d
      // Gumps response
      waitforgump 0x77567887 15000
      replygump 0x77567887 16
      waitforgump 0x77567887 15000
   else
      sysmsg 'Out of sewing kit!' 25
      break
   endif
endwhile
```

Aliases

System Aliases

System has a few predefined aliases that can be used on macros.

Usage

```
/*
backpack: player backpack
bank: player bank
enemy: current enemy
friend: current friend
ground: world ground
last/lasttarget: last targeted mobile
lastobject: last used object, item or mobile
lefthand: player equipped left hand item
mount: current mount
righthand: player equipped right hand item
self: player character
*/
```

Sample

```
// Open backpack
// Suffix '!' to avoid queue
useobject! 'backpack'
// Move wearing shield to bank
moveitem 'lefthand' 'bank'
```

Unset Alias

Unset and remove an existing alias.

Usage

```
unsetalias ('alias name')
```

Set Alias

Define an existing alias with a given serial or another alias value.

Usage

```
setalias ('alias name') [serial]
```

Sample

```
// Prompt in-game for a new pet
setalias 'pet'
// Specific item serial
setalias 'oldObject' 0x40116650
// Another existing alias
setalias 'newObject' 'oldObject'
// newObject is now equal both 0x40116650 and oldObject alias
```

Find Alias

Search if a specific custom alias name is already created.

Usage

```
if findalias ('alias name')
endif
```

Sample

```
if not findalias 'weapon'
   // Prompt for an alias
   promptalias 'weapon'
endif
// Toggle weapon from paperdoll
if findobject 'righthand'
   clearhands 'right'
else
   equipitem 'weapon' 1
endif
```

Prompt Alias

Prompt in-game for a new alias and wait until it is selected.

Usage
promptalias ('alias name')

Conditions

Contents

Retrieve and compare the amount of items inside a container.

Usage

```
if contents (serial) ('operator') ('value')
endif
```

Sample

```
if contents 'backpack' > 10
   sysmsg 'More than 10 items inside backpack!'
endif
```

Regions

Checks whether an item or mobile region type matches.

Usage

```
if inregion ('region type') [serial] [range]
endif
```

Sample

```
// Check if local player is in town
if inregion 'town'
   msg 'bank'
endif
// Check if enemy is in guards zone in range of 10 tiles
autotargetobject 'enemy'
cast 'Lightning'
if innocent 'enemy' and inregion 'guards' 'enemy' 10
   cancelautotarget
endif
```

Skills

Check for a specific local player skill value.

Usage

```
if skill ('name') (operator) (value)
endif
```

Sample

```
// Basic train Necromancy sample
if mana <= 10
  // Server must support buff icons, otherwise use injournal to detect trance
  while not buffexists 'Meditation'
    useskill 'Meditation'
    pause 1000
  endwhile
  pause 15000
endif
if skill 'Necromancy' >= 99
  cast 'Vampiric Embrace'
elseif skill 'Necromancy' >= 75
  cast 'Lich Form'
else
  cast 'Horrific Beast'
endif
```

Player Attributes

List of all possible conditional attributes for local player.

Usage

```
/*
Attributes:
- Coordinates: x, y, z;
- Resistances: physical, fire, cold, poison, energy
- Status: str, dex, int, hits, maxhits, diffhits, stam, maxstam, mana, maxmana;
- System: usequeue, dressing, organizing;
- Others: followers, maxfollowers, gold, hidden, luck, tithingpoints, weight, maxweight.
*/
if (attribute) [operator] [value]
endif
```

Sample

```
if hits <= maxhits
  bandageself
  // Check if missing hits (max hits - current hits) is greater than 30
  if diffhits > 30
    autotargetself
    cast 'Greater Heal'
  endif
endif
// Simple hidden check
if not hidden
  useskill 'Hiding'
endif
```

```
// Out of followers slots?
if followers == followersmax
   msg 'a dog release'
endif
if usequeue >= 8
   clearusequeue
endif
```

Object Attributes

List of all possible conditional attributes for mobiles and items, including local character.

Usage

```
/*
Attributes:
- All: serial, graphic, color, x, y, z;
- Items: amount;
+ Mobiles:
- General: dead, direction, hits, maxhits, diffhits, flying, paralyzed, poisoned, mounted, yellowhits;
- Notoriety: criminal, enemy, friend, gray, innocent, invulnerable, murderer.
*/
if (attribute) [serial] [operator] [attribute] [serial]
endif
```

Sample

```
// Cast cure if poisoned
// No serial or alias is equal to use 'self'
if poisoned
   autotargetself
   cast 'Cure'
endif
// Check if enemy is female human
if graphic 'enemy' == 401
   // Tell her something nice
   msg 'Hey pretty!'
endif
// Check for an item serial amount, use it and target self
if amount 0x40116650 >= 20
   autotargetself
   useobject 0x40116650
endif
// Suppressor '@' to avoid in range warnings
if @inrange 'enemy' 10 and not dead 'enemy'
   // Check if enemy is flying or mounted
   if flying 'enemy' or mounted 'enemy'
      // Automate next target to enemy
      autotargetobject 'enemy'
      // Use bolas, do not queue
      usetype! 0x26ac
   endif
endif
```

Find Object

Search for an item by serial or alias.

Usage

```
if findobject (serial) [color] [source] [amount] [range]
endif
```

Sample

```
// Search for right hand object
// righthand is a system predefined alias, you can use a custom one
```

```
if findobject 'righthand'
    clearhands 'right'
endif
// Find an item by serial, any color, minimum amount of 10 and in range of 2 tiles
if findobject 0x40116650 'any' 'ground' 10 2
    // Move 10 items to character's backpack
    moveitem 0x40116650 'backpack' 0 0 0 10
endif
```

Distance and Range

Check for distance or range between your character and another mobile or an item.

Usage

```
if distance (serial) (operator) (value)
endif
if inrange (serial) (range)
endif
```

Sample

```
// Check for specific serial range
if distance 0x40116650 <= 2
   // Move item to backpack
   moveitem 0x40116650 'backpack'
endif
// Check if friend alias is in range of 10 tiles
if inrange 'friend' 10
   miniheal 'friend'
endif
```

Buffs

Check for a specific buff.

Usage

```
/*
Buffs:
- Abilities: bleed, mortal strike, disarm, dismount;
- Skills: hiding, meditation;
- Chivalry: divine fury, enemy of one;
- Magery: bless, night sight, strength, cunning, agility, curse, mass curse, weaken, feeblemind, clumsy, poison, paralyze, invisibility, polymorph, magic reflection, arch protection, protection, reactive armor, incognito;
- Ninjitsu: death strike, animal form;
- Necromancy: evil omen, corpse skin, blood oath, mind rot, pain spike, strangle;
- Spellweaving: gift of renewal, attune weapon, thunderstorm, essence of wind, ethereal voyage, gift of life, arcane empowerment;
- Others: disguised.
*/
if buffexists ('buff name')
endif
```

Sample

```
if not buffexists 'Divine Fury'
   cast 'Divine Fury'
endif
```

Property

Check for a specific item or mobile property, existence and value.

Usage

```
if property ('name') (serial) [operator] [value]
endif
```

Sample

```
// Define a trash barrel
if not @findobject 'trash'
   promptalias 'trash'
endif
// Define a pouch
if not @findobject 'pouch'
   promptalias 'pouch'
endif
// Start loop searching for a ring on backpack
while @findtype 0x108a 'any' 'backpack'
   // Ring found, check for a desired property
   if property 'Faster Casting Recovery' 'found' == 3
      sysmsg 'Valid, move to pouch!'
      moveitem! 'found' 'pouch'
   else
      sysmsg 'Invalid, move to trash!'
      moveitem! 'found' 'trash'
   endif
   pause 1000
endwhile
```

Find Type

Search for an item type (graphic) and set alias "found".

Usage

```
if findtype (graphic) [color] [source] [amount] [range]
endif
```

Sample

```
// Search for 20 bandages of any color
if findtype 0xe21 'any' 'backpack' 20
   // Item found, move to the ground
   moveitem 'found' 'ground' 1250 489 0
else
   // Run buy agent list named 'Bandages'
   buy 'Bandages'
endif
// Find a cow of any color in range of 3 tiles and tame it
if findtype 0xe7 'any' 'world' 0 3
   // Automate next - incoming - target to an object
   autotargetobject 'found'
   useskill 'Animal Taming'
   // Pause for 10 seconds
   pause 10000
endif
```

Find Layer

Search for an equipped item.

Usage

```
findlayer (serial) (layer)
```

Sample

```
// Prefix '@' to avoid system warnings
if @findlayer 'self' 2
   // Unequip shield
   @moveitem! 'found' 'backpack'
endif
```

Skill State

Checks whether a skill is locked, up or down.

Usage

```
skillstate ('skill name') (operator) ('locked'/'up'/'down')
```

Sample

```
if skillstate 'Magery' == 'down'
   setskill 'Magery' 'up'
endif
```

Count Type

Amount comparison of item type inside a container.

Usage

```
counttype (graphic) (color) (source) (operator) (value)
```

Sample

```
// Check if backpack contains 2 bank checks
// Use '!' suffix to ignore stacked amounts
if counttype! 0x14f0 'any' 'backpack' == 2
   sysmsg '2 bank checks found!' 86
endif
// Check if amount of heal potions inside backpack is greater than 10 and consider stacked amounts
if counttype 0xf0c 'any' 'backpack' > 10
   sysmsg 'More than 10 heal pots!' 86
endif
```

Count Type Ground

Amount comparison of item or mobile type on the ground.

Usage

```
counttypeground (graphic) (color) (range) (operator) (value)
```

Sample

```
// Wait and check for at least 2 grey wolves in range of 8 tiles every 2 seconds
// Prefix '@' to suppress warnings
while @counttypeground 0x19 'any' 8 < 2
   pause 2000
endwhile
```

Find Wand

Search for a wand and set alias "found".

Usage

```
/*
Spells:
- Clumsy;
- Identification;
- Heal;
- Feeblemind;
- Weaken;
- Magic Arrow;
- Harm;
- Fireball;
- Greater Heal;
- Lightning;
- Mana Drain.
*/

if findwand ('spell name'/'any'/'undefined') [source] [minimum charges]
endif
```

Sample

```
// Search for a heal wand inside backpack with at least 5 charges and move to bank
// Prefix '@' to avoid warnings
if @findwand 'heal' 'backpack' 5
   moveitem! 'found' 'bank'
endif
```

Gumps

Wait For Gump

Wait for a gump from server.

Usage

```
waitforgump (gump id/'any') (timeout)
```

Sample

```
useobject! 0x491093
// Wait for gump during 5 seconds
waitforgump 0x1ec8c837 5000
```

Reply Gump

Reply a server gump.

Usage

```
replygump (gump id/'any') (button) [option]
```

Sample

```
useobject! 0x491093
waitforgump 0x1ec8c837 5000
replygump 0x1ec8c837 1
```

In Gump

Check for a text in gump.

Usage

```
ingump (gump id/'any') ('text')
```

Sample

```
// Find for a text on that gump
if ingump 0x1ec8c837 'Home'
   replygump 0x1ec8c837 2
endif
```

Journal

In Journal

Check for a text in journal, optional source name.

Usage

```
if injournal ('text') ['author'/'system']
endif
```

Sample

```
// Prefix '@' to suppress system warnings
if @injournal 'outside the protection' 'system'
    // Do something...
    // Clear all journal
    @clearjournal
endif
```

Clear Journal

Clear all journal texts.

Usage

```
clearjournal
```

Wait For Journal

Check for a text in journal until it finds a text or timeout, optional source name.

Usage

```
waitforjournal ('text') (timeout) ['author'/'system']
```

Sample

```
// Wait for a system message during 5 seconds
waitforjournal 'too far away' 5000 'system'
```

Lists

Pop List

Remove an element from a named and existing list.

Usage

```
poplist ('list name') ('element value'/'front'/'back')
```

Sample

```
createlist 'sample'
// Banana
pushlist 'sample' 'banana'
// Apple
pushlist 'sample' 'apple'
// Lemon
pushlist 'sample' 'lemon'
// Grape
pushlist 'sample' 'grape'
// Pop banana
poplist 'sample' 'banana'
// Now apple is our front element
// Pop front
poplist 'sample' 'front'
// Element apple no longer exists, check output
for 0 to 'sample'
    sysmsg sample[]
endfor
// Remove all bananas from the list by adding '!' suffix
poplist! 'sample' 'banana'
```

Push List

Add a new element to an existing list.

Usage

```
pushlist ('list name') ('element value') ['front'/'back']
```

Sample

```
createlist 'sample'
// Apple
pushlist 'sample' 'apple'
// Lemon
pushlist 'sample' 'lemon'
// Grape
pushlist 'sample' 'grape'
// Insert a new Grape before all other elements
pushlist 'sample' 'grape' 'front'
// Use suffix '!' for unique element values
while not pushlist! 'grape'
    // Could not push because it already exists, remove all grapes
    poplist 'sample' 'grape'
endwhile
```

Remove List

Remove a named and existing list.

Usage

```
removelist ('list name')
```

Sample

```
// Create and populate a new list
createlist 'sample'
pushlist 'sample' 'Hello'
pushlist 'sample' 'World'
for 0 to 'sample'
   msg sample[]
endfor
// Remove list
removelist 'sample'
if not listexists 'sample'
   sysmsg 'List removed successfully!'
else
   // Unreachable code
endif
```

List Exists

Check if a named list exists.

Usage

```
if listexists ('list name')
endif
```

Sample

```
// Create a new list in case it does not exists
if not listexists 'sample'
   createlist 'sample'
endif
```

List Count

Compare the size of an existing list with a given value.

Usage

```
if list ('list name') (operator) (value)
endif
```

Sample

```
// Create new list in case it does not exists
if not listexists 'sample'
createlist 'sample'
endif
// In case list is empty append values
// Just a sample, it could be added to listexists statement block
if list 'sample' == 0
pushlist 'sample' 'Hello'
pushlist 'sample' 'World'
endif
for 0 to 'sample'
msg sample[]
endfor
```

Create List

Create a new named list.

Usage

```
createlist ('list name')
```

In List

Checks whether a list contains a given element.

Usage

```
if inlist ('list name') ('element value')
endif
```

Sample

```
if not listexists 'sample'
   createlist 'sample'
endif
pushlist 'sample' 'Hello'
pushlist 'sample' 'World'
// Case sensitive disabled will return true
if inlist 'sample' 'hello'
   sysmsg 'List contains element!'
endif
// Use suffix '!' to enable case sensitive
if inlist! 'sample' 'world'
   // Unreachable code
endif
```

Clear List

Clear a list by name.

Usage

```
clearlist ('list name')
```

Sample

```
// Create and populate a list
if not listexists 'sample'
   createlist 'sample'
endif
pushlist 'sample' 'Hello'
pushlist 'sample' 'World'
if list 'sample' > 0
   sysmsg 'List is not empty!'
endif
// Clear list
clearlist 'sample'
// Now list is empty but still exists, use removelist command to delete!
if listexists 'sample'
   sysmsg 'List exists!'
endif
if list 'sample' == 0
   sysmsg 'List is now empty!'
endif
```

Main

Object Inspector

Prompt to inspect in-game object.

Usage
info

Pause

Insert a pause/wait in milliseconds to your macro.

Usage
pause (timeout)

Sample
```
// 1 second
pause 1000
// 0.5 second
pause 500
```

Ping Server

Retrieve an approximated ping with server.

Usage
ping

Play Macro

Run a specific macro by name.

Usage
playmacro 'name'

Play Sound

Play sound by id or system .wav file.

Usage
playsound (sound id/'file name')

Sample
// System .wav file
playsound 'name.wav'
// Game sound id
playsound 25

Resynchronize

Resynchronize game data with server.

Usage
resync

Snapshot

Same as a print screen, command allows you to create a snapshot, it is also possible to add a delay before snapshotting.

Usage

```
snapshot [timer]
```

Sample

```
// Instant
snapshot
// Wait 5 seconds before snapshotting
snapshot 5000
```

Toggle Hotkeys

Enable and disable hotkeys.

Usage

```
hotkeys
```

Where

Display coordinates and region name.

Usage
where

Message Box

Show a simple message box with a custom title and body.

Usage
messagebox ('title') ('body')

Sample
messagebox 'Sample' 'Hello world!'

Others

Paperdoll

Paperdoll related commands.

Usage

```
// Open
paperdoll (serial)
// Buttons
helpbutton
guildbutton
questsbutton
```

Sample

```
// Open my paperdoll
paperdoll
// Open friend paperdoll
paperdoll 'friend'
// Invoke quests button
questsbutton
```

Virtues

Use a virtue by name.

Usage

```
virtue ('honor'/'sacrifice'/'valor')
```

Sample

```
// Search for an ettin in range of 5 tiles
// Prefix '@' to suppress system warnings
if @findtype 0x12 0 0 0 5
   autotargetobject 'found'
   // Use virtue honor
   virtue 'Honor'
endif
```

Messages

Send a text message.

Usage

```
msg ('text') [color]
headmsg ('text') [color]
partymsg ('text')
guildmsg ('text')
allymsg ('text')
whispermsg ('text')
yellmsg ('text')
sysmsg ('text')
chatmsg ('text')
emotemsg ('text')
promptmsg ('text')
waitforprompt (timeout)
cancelprompt
```

Sample

```
// Internal system message
sysmsg 'Hello World!'
// Party message
partymsg "What's up?"
// Over head public message
msg 'Hi'
// Red over head private message
headmsg 'Hi' 26
```

Friend List

In-game prompt for add or remove a mobile from friend list.

Usage

```
addfriend
removefriend
```

Context Menu

Request or wait for a context menu option.

Usage
contextmenu (serial) (option)
waitforcontext (serial) (option) (timeout)

Ignore Object

Add a serial to the ignore list affecting findtype command.

Usage
// Trigger
ignoreobject (serial)
// Cleaner
clearignorelist

Set Skill

Set a skill into a specific state: locked, up or down.

Usage

```
setskill ('skill name') ('locked'/'up'/'down')
```

Sample

```
// Lock magery skill
if skill 'Magery' == 105
   setskill 'Magery' 'locked'
endif
```

Wait For Properties

Request and wait for properties of an item or mobile.

Usage

```
waitforproperties (serial) (timeout)
```

Sample

```
setalias 'ring' 0x409c89fa
// Request and wait for 5 seconds
waitforproperties 'ring' 5000
if property 'Faster Casting Recovery' 0x409c89fa == 3
   moveitem 'ring' 'backpack'
   pause 1000
endif
```

Spells

Mini and Big Heal

Cast heal, cure, greater heal or arch cure upon a mobile.

Usage

```
miniheal [serial]
bigheal [serial]
```

Sample

```
// Mini heal self
miniheal
// Big heal friend
bigheal 'friend'
```

Cast

Cast a spell by id or name.

Usage

```
cast (spell id/'spell name'/'last')
```

Sample

```
// Magic Arrow and Fireball sample
// Check 'Do not auto interrupt' option
// Simple cast
cast 'Magic Arrow'
waitfortarget 650
target 'enemy'
// Another simple cast
cast 'Fireball'
waitfortarget 1250
target 'enemy'
// Check for curse and remove it
// Prefix '@' to disable system warnings
if @buffexists 'Curse'
    // Managed cast
    cast 'Remove Curse' 'self'
endif
// Automated target sample
autotargetobject 'enemy'
cast 'Lightning'
```

Chivalry Heal

Cast close wounds or cleanse by fire upon a mobile.

Usage

```
chivalryheal [serial]
```

Sample

```
// Chivalry heal self
chivalryheal
// Chivalry heal friend
chivalryheal 'friend'
```

Targeting

Wait For Target

Wait for a new client target cursor from server.

Usage

```
waitfortarget (timeout)
```

Sample

```
cast 'Explosion'
// Wait for 2.5 seconds until target comes
waitfortarget 2500
// Not queued target on enemy
target! 'enemy'
```

Cancel Target

Cancel an existing cursor/target.

Usage

```
canceltarget
```

Direct Target

Instantly target a given alias, serial, type or location.

Usage

```
// Triggers
target (serial) [timeout]
targettype (graphic) [color] [range]
targetground (graphic) [color] [range]
targettile ('last'/'current'/(x y z)) [graphic]
targettileoffset (x y z) [graphic]
targettilerelative (serial) (range) [reverse = 'true' or 'false'] [graphic]
// Cleaner
cleartargetqueue
```

Sample

```
// Heal friend
cast 'Heal'
waitfortarget 250
// Queued
target 'friend'
// Bola enemy
usetype! 0x26ac
waitfortarget 500
// Suffix '!' to avoid queue
target! 'enemy'
```

Automated Target

Setup an internal wait for target and automatically target an object.

Usage

```
// Triggers
autotargetlast
autotargetself
autotargetobject (serial)
autotargettype (graphic) [color] [range]
autotargettile ('last'/'current'/(x y z)) [graphic]
autotargettileoffset (x y z) [graphic]
autotargettilerelative (serial) (range) [reverse = 'true' or 'false'] [graphic]
autotargetghost (range) [z-range]
autotargetground (graphic) [color] [range]
// Cleaner
cancelautotarget
```

Sample

```
// Cancel any previous automated target
cancelautotarget
// Target self with a greater heal as soon as target is ready
autotargetself
cast 'Greater Heal'
// Target enemy with an explosion
autotargetobject 'enemy'
cast 'Explosion'
```

Get Enemy

Get and set an "enemy" alias according to the given parameters.

Usage

```
/*
Notorieties: friend, innocent, murderer, enemy, criminal, gray
Filters: humanoid, transformation, closest, nearest
*/
getenemy ('notoriety') ['filter']
```

Sample

```
// Get closest humanoid enemy
getenemy 'murderer' 'criminal' 'gray' 'closest' 'humanoid'
if inrange 'enemy' 10
   autotargetobject 'friend'
   cast 'Lightning'
endif
```

Target Exists

Check for a specific or any cursor/target type.

Usage

```
if targetexists ['any'/'beneficial'/'harmful'/'neutral'/'server'/'system']
endif
```

Sample

```
// Basic smart target last sample
unsetalias 'smart'
if targetexists 'harmful'
   setalias 'smart' 'enemy'
endif
if targetexists 'beneficial'
   setalias 'smart' 'friend'
endif
// Blank type is the same as 'any'
if targetexists
   setalias 'smart' 'last'
endif
// Suppressor '@' to avoid system warnings
if @findalias 'smart' and inrange 'smart' 10
   // Suffix '!' to avoid queue
   target! 'smart'
endif
```

Waiting For Target

Returns true whenever the core is internally waiting for a server target.

Usage

```
waitingfortarget
```

Sample

```
// Search for a pouch inside backpack
if @findtype 0xe79 'any' 'backpack'
  useobject! 'found'
  // Let's assume healing option is running, hold the cast until it applies the bandage
  while waitingfortarget or targetexists 'server'
  endwhile
  cast 'Magic Trap'
  waitfortarget 1200
  target! 'found'
endif
```

Get Friend

Get and set a "friend" alias according to the given parameters.

Usage

```
/*
Notorieties: friend, innocent, murderer, enemy, criminal, gray, invulnerable
Filters: humanoid, transformation, closest, nearest
*/
getfriend ('notoriety') ['filter']
```

Sample

```
// Get a humanoid friend
getfriend 'innocent' 'friend' 'humanoid'
if inrange 'friend' 10
  autotargetobject 'friend'
  cast 'Greater Heal'
endif
```

Timers

Timer Value

Check for a named timer value.

Usage

```
if timer ('timer name') (operator) (value)
endif
```

Sample

```
// Create a new timer
if not timerexists 'sample'
   createtimer 'sample'
endif
// Reset every 10 seconds
if timer 'sample' > 10000
   settimer 'sample' 0
endif
```

Timer Exists

Check if a named timer exists.

Usage

```
if timerexists ('timer name')
endif
```

Sample

```
if not timerexists 'sample'
   createttimer 'sample'
endif
```

Set Timer

Set a timer value and create in case it does not exist.

Usage

```
settimer ('timer name') (value)
```

Sample

```
if not timerexists 'sample'
   settimer 'sample' 10000
endif
if skill 'Spirit Speak' < 100 and timer 'sample' >= 10000
   useskill 'Spirit Speak'
   settimer 'sample' 0
endif
```

Remove Timer

Remove a specific timer by name.

Usage

```
removetimer ('timer name')
```

Create Timer

Create a new named timer.

Usage

```
createtimer ('timer name')
```

Sample

```
// Create a new timer and start counting
if not timerexists 'sample'
   createtimer 'sample'
endif
```