

Classic Internal Assistant Scripting Documentation

Sascha Puligheddu (“fwiffo”)

2023

Summary

- Introduction
 - [Syntax](#)
 - [Object Inspector](#)
 - [Layers](#)

- Commands
 - [Actions](#)
 - [Agents](#)
 - [Aliases](#)
 - [Conditions](#)
 - [Gumps](#)
 - [Journal](#)
 - [Lists](#)
 - [Main](#)
 - [Others](#)
 - [Spells](#)
 - [Targeting](#)
 - [Timers](#)

Introduction

Syntax

The assistant ("Assist") 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 available on Assist, 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 a parameter that has apostrophe 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.

```
if (statement)
elseif (statement)
else
endif
```

```
while (statement)
endwhile
```

```
for (value)
endfor
```

```
for (start) to (end)
endfor
```

```
for (start) to ('list name')
endfor
```

```
foreach (start) in ('list name')
endfor
```

```
for (start) to (end) in ('list name')
endfor
```

```
foreach aliasname in 'list name'
endfor
```

Keywords

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

Let's suppose you want to say "I love UOIReborn!" 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 message in game
  msg "I love UOIReborn!"
  pause 1500
endfor
sysmsg "End of for loop"
```

```
//repeat for all elements in the list, the current element during iteration is saved in the alias 'testalias'
foreach 'testalias' in 'list name'
sysmsg 'testalias'
endfor
```

Now let's 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 UOIReborn!" 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 UOIReborn much more, I wanna say that "while" I'm alive, but with a random pause:

Code

```
// While I'm alive
while not dead and
  // Send a lovely message in game
  msg "I love UOIReborn!"
  // get a random pause from 500 to 1500, this gets saved as alias "rnd" for use in lists or similar
  random 500 1500
  pause "rnd"
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 UOIReborn 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 UOIReborn!"
  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 applies to 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 clears the target queue. If a command supports that symbol you'll be able to verify the effects on its description.

Code

```
cast 'Heal'
waitfortarget 500
// Clears target queue and uses only the target I give
target! 'self'
```

Object Inspector

At this point you should be able to understand better how scripts are written in Assistant, 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:



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'
```

Layers

A few commands - such as equipitem and findlayer - require a layer number parameter. It is possible to grab that value by equipping an item and using the Object Inspector on it, however if you need the layer of an item you can't easily inspect, check the list below:

Right Hand: 1
Left Hand: 2
Shoes: 3
Pants: 4
Shirt: 5
Head: 6
Gloves: 7
Ring: 8
Talisman: 9
Neck: 10
Hair: 11
Waist: 12
Inner Torso: 13
Bracelet: 14
Facial Hair: 16
Middle Torso: 17
Earrings: 18
Arms: 19
Cloak: 20
Backpack: 21
Outer Torso: 22
Outer Legs: 23
Inner Legs: 24
Mount: 25
Shop Buy: 26
Shop Restock: 27
Shop Sell: 28
Bank: 29

Commands

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 'any' 0 0 5
  autotargetobject 'found'
  // Use an explosion potion so that it will be lauched automatically at the object!
  usetype 0x0F0D 'any' 'backpack'
  // 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 elapsed 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).

It is possible to queue usetype command requests, you may use suffix "!" in order to bypass queue, otherwise it will add the item into system's use object queue if an item with given parameters exists.

Usage

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

Sample

```
// Use any bandage color from a specific container (with 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.

It is possible to queue useobject command requests, you may use suffix "!" in order to bypass queue, otherwise it will add the item into system's use object queue if an item with given parameters exists.

Usage

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

Sample

```
// Use a specific object by serial (with 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.

It is possible to queue useonce command requests, you may use suffix "!" in order to bypass queue, otherwise it will add an item into system's use objects queue every time useonce command finds an item matching given parameters.

System automatically set use once items color: used (grey) and next (red).

Usage

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

Sample

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

Move Item (Serial or Type)

Move an item serial or type from source to destination.

It is possible to disallow stacking by using "!" suffix.

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 or search level]

movetypeoffset (graphic) (source) 'ground' [(x, y, z)] [color] [amount] [range or search level]

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! 0xced 'backpack' 'ground' 1950 50 0
// Move 200 gold from the ground in range of 2 tiles to backpack
movetype 0xced '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"
endif
```

Feed

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

You can edit Data/foods.xml (File) in order to customize and/or add new food groups and types.

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) [color] [amount]

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 with 2 fish steaks

feed 'self' 0x97B 'any' 2

Show Names

Display corpses and/or mobiles names.

Usage

shownames ['mobiles'/'corpses'/'all']

Sample

// Display mobiles names

shownames 'mobiles'

// Display corpses names

shownames 'corpses'

// Display all mobiles and corpses names

shownames 'all'

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
```

List of Skills:

```
"anatomy"
"animallore", "animal lore"
"taming", "animaltaming", "animal taming"
"armslore", "arms lore"
"begging"
"detectinghidden", "detect", "detecthidden", "detecting hidden", "detect hidden"
"discordance", "discord", "enticement", "entice"
"evaluatingintelligence", "evalint", "eval", "evaluating intelligence"
"focus"
"forensicevaluation", "forensiceval", "forensics"
"hiding"
"inscription", "scribe"
"itemidentification", "itemid", "item identification", "item id"
"meditation"
"peacemaking", "peace", "peacemake"
"poisoning"
"provocation", "provo"
"removetrap"
"spiritspeak"
"stealing"
"stealth"
"tasteidentification", "tasteid"
"tracking"
```

Rename

Request the server to rename a mobile.

Usage

```
rename (serial) ('name')
```

Sample

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

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
```

Toggle Mounted

Mount and dismount.

In case mount alias is not defined system will automatically prompt for a mount.

Usage

```
togglemounted
```

Equip Item

Equip a specific item into a given layer.

You can grab the layer number by equipping your item into your paperdoll and using object inspector on it.

It is possible to avoid move item queue by using the suffix "!". Currently equipped item on same layer will be removed.

Usage

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

Sample

```
// Prompt for a new bow
promptalias 'bow'
// Check if selected bow exists
if @findobject 'bow'
    equipitem 'bow' 2
endif
// You can also equip that bow without adding to the queue
equipitem! 'bow' 2
```

Equip Wand

Search for a wand inside your backpack and equip it, spell names are as seen on the wand in game, also, currently equipped item on the same layer will be removed if the equipwand finds an equippable item in the backpack.

Usage

```
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.

Go to Agents (Tab) -> Vendors (Tab) -> Select Buy or Sell from drop down selection -> New (Button). A new profile will be created, click it in order to rename, for example "Reagents", add reagent types you would like to buy or sell, every time you will interact with a vendor, if you have an active vendor buy or sell agent, it will search for items that the vendor can sell or buy and buy/sell the item you set up in the profile.

Usage

```
// Triggers, remember that CASE does matter
buy ('list name')
sell ('list name')
// Cleaners (disables the relative agents, visible in the GUI)
clearbuy
clearsell
```

Organizer

Execute a specific organizer profile.

Go to Agents (Tab) -> Organizer (Tab) -> New (Button). A new profile will be created, click it in order to rename, for example "Reagents", add reagent types and amounts to your items list. Don't forget to set containers, both source and destination.

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.

Go to Agents (Tab) -> Autoloot (Tab) to configure your Autoloot settings.

Usage

```
autoloot
```

Dress and Undress

Dress or undress temporary or specific profile.

Go to Agents (Tab) -> Dress (Tab) -> New (Button). A new profile will be created, double click it in order to rename, for example "Weapon", equip a weapon and add it to your profile items list. You can now call dress and undress commands to your new profile. In case no profile name is provided system will dress according to your temporary dress settings.

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 the equipment no matter the profile
undress
```

Toggle Autoloot

Enable and disable autoloot agent.

Usage

```
toggleautoloot
```

Toggle Scavenger

Enable and disable scavenger agent.

Usage

```
togglescavenger
```

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
lastcombatant: last attacked 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
rnd: number memorized using the random command/expression

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')

Prompt Alias

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

Usage

promptalias ('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
```

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
```

In Region

Checks whether an item or mobile region type matches. You can edit Data/regions.xml (File) in order to customize region names and guard zone lines.

Usage

```
if inregion ('guards'/'town'/'dungeon'/'forest'/'any') [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
```

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
```

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. There are others mobile attributes that can be used by local player through the alias "self", check Object Attributes commands.

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, diffweight.

```
*/
```

```
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. In case you are looking for other attributes related to local character you can find more on Player Attributes commands.

Usage

/* Attributes:

- All: serial, graphic, color, x, y, z;

- Items: amount; + Mobiles:

- General: name, dead, direction, hits, maxhits, diffhits, flying, paralyzed, poisoned, mounted, yellowhits, war;

- 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
```

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
```

Bufs

Check for a specific buff.

This function only works on servers supporting buff icons.

Usage

```
/*
```

```
Bufs:
```

- 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 or search level]
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.

You may use Object Inspector (Button) and click an equipped item to grab the layer value.

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.

Supported operators are: != or ==

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.

In order to ignore and do not count stacked amounts use the suffix '!'.

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.

In order to ignore and do not count stacked amounts use the suffix '!'.

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 are as defined with the commands cast, with string word, examples are:

//- Clumsy;

//- 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
```

In Party

Checks whether a mobile is in your party.

Usage

```
if inparty (serial) endif
```

Sample

```
// Checks if I'm in party
```

```
if inparty 'self'
```

```
partymsg 'Hello world!'
```

```
endif
```

In Friend List

Checks whether a mobile is in your friend list.

Usage

```
if infriendlist (serial)
endif
```

Sample

```
if infriendlist 'enemy'
sysmsg 'Current enemy in friend list'
else
attack 'enemy'
endif
```

War

Checks whether a mobile is in war mode.

Usage

```
if war (serial)
endif
```

Sample

```
if war 'self'
warmode 'off'
endif
```

Gumps

Wait for Gumps

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.

For really complex gumps it is suggested to use the record functionality while using the gump to get the various buttons and textids that are included into the gump.

Usage

replygump (gump id/"any") (button) [checkboxid/"textid text response"]

Sample

```
//uses an object that triggers a gump and reply to that gump by pressing the button id 1
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
```

Gump Exists

Checks if a gump id exists or not.

Usage

```
if gumpexists (gump id/'any')
endif
```

Sample

```
// Check if any gump exists
if gumpexists 'any'
sysmsg 'There is at least 1 gump'
endif
// Check for a specific gump id
if gumpexists 0x3029
replygump 0x3029 1
endif
```

Close Gump

Close a specific gump type by serial.

Usage

```
closegump ('paperdoll'/'status'/'profile'/'container') ('serial')
```

Sample

```
promptalias 'spy'
paperdoll 'spy'
pause 5000
closegump 'paperdoll' 'spy'
```

Journal

In Journal

Check for a text in journal, optional source name.

Usage

```
if journal ('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.

While using the suffix "!" system will remove all elements matching element value from the list.

Usage

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

Sample

```
createlist 'sample'  
// Banana  
pushlist 'sample' 'banana'  
// Apple  
pushlist 'sample' 'apple'  
// Lemon  
pushlist 'sample' 'lemon'  
// 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.

Default position is "back", while using the suffix "!" system will only push an item to the list in case it does not already contains it (unique).

Usage

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

Sample

```
createlist 'sample'  
// Apple  
pushlist 'sample' 'apple'  
// Lemon  
pushlist 'sample' 'lemon'  
// Insert a 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
```

Create List

Create a new named list.

Usage

```
createlist 'list name'
```

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
```

In List

Checks whether a list contains a given element.

Case sensitive is disabled by default for the elements inside the list, to enable it append "!" suffix to this command.

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.

Name parameter is case sensitive.

Usage

playmacro 'name'

Play Sound

Play sound by id.

Usage

playsound (sound id/'file name')

Sample

```
// Game sound id  
playsound 25
```

Resynchronize

Resynchronize game data with server.

You must wait 0.8 seconds between resynchronize requests.

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.

You can configure snapshots settings via Snapshots tab.

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!'
```

Click Screen

Use your mouse to click coordinates on your screen. Default is 'single' 'left' click

Usage

```
clickscreen (x) (y) ['single'/'double'] ['left'/'right']
```

Sample

```
// Single right click at 200, 500
```

```
clickscreen 200 500 'right'
```

```
// Double left click at 400, 150
```

```
clickscreen 400 150 'double' 'left'
```

Others

Paperdoll Buttons

Paperdoll buttons related commands.

Usage

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

Sample

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

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] [serial]
partymsg ('text')
guildmsg ('text')
allymsg ('text')
whispermsg ('text')
yellmsg ('text')
sysmsg ('text')
chatmsg ('text')
emotemsg ('text')
promptmsg ('text')
timermsg ('timer name') [color]
waitforprompt (timeout)
cancelprompt
```

Sample

```
// Internal system message, visible only by you
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.

You can also manage your friends via interface on Options (Tab) -> Friends (Tab).

Usage

```
addfriend
removefriend
```

Context Menu

Request or wait for a context menu option.

Automatic generation available via Macros (Tab) recording.

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
```

Wait For Contents

Wait for the server to send container contents, it will also try opening the container once.

Usage

```
waitforcontents (serial) (timeout)
```

Sample

```
// Ask for a new bag
promptalias 'bag'
// Try opening once and wait for contents for 2 seconds
waitforcontents 'bag' 2000
```

Automated Color Pick

Setup an automated reply to the incoming dye color gump, allowing you to define dye tubs color. That command should be added prior to the action that opens the color pick gump.

Usage

autocolorpick (color)

Sample

```
if not @findobject 'dyes'  
promptalias 'dyes'  
endif  
if not @findobject 'tub'  
promptalias 'tub'  
endif  
autocolorpick 35  
useobject! 'dyes'  
waitfortarget 1000  
target! 'tub'
```

Wait For Contents

Wait for the server to send container contents, it will also try opening the container once.

Usage

waitforcontents (serial) (timeout)

Sample

```
// Ask for a new bag  
promptalias 'bag'  
// Try opening once and wait for contents for 2 seconds  
waitforcontents 'bag' 2000
```

Spells

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'
```

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

Waiting For Target

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

It is useful for creating macros that will not mess up with agents and other iteration.

Usage

```
if not waitingfortarget  
//execute this code  
endif
```

Sample

```
// Search for a pouch inside backpack  
if @findtype 0xe79 'any' 'backpack'  
useobject! 'found'  
// Let's assume we have the organizer is running, hold the cast until it ends  
while waitingfortarget or organizing  
endwhile  
cast 'Flamestrike'  
waitfortarget 1200  
target! 'found'  
endif
```

Direct Target

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

Default queue timeout is 5 seconds, use suffix "!" in order to bypass queue.

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'
// Explosion potion to enemy
usetype! 0x0F0D
waitfortarget 500
// Suffix '!' to avoid queue
target! 'enemy'
```

Get Friend

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

You can edit Data/bodies.xml (File) in order to customize humanoid and transformations body values filtering; by default get command will always list possible targets and switch between them every time the macro is executed, make sure to put closest or nearest filter when needed.

Nearest filter will switch between the 2 closest targets.

Usage

```
//Notorieties: any, 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
```

Get Enemy

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

You can edit `Data/bodies.xml (File)` in order to customize humanoid and transformations body values filtering; by default get command will always list possible targets and switch between them every time the macro is executed, make sure to put `closest` or `nearest` filter when needed. `Nearest` filter will switch between the 2 closest targets.

Usage

```
//Notorieties: any, 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 'enemy'
cast 'Lightning'
endif
```

Target Exists

Check for a specific or any cursor/target type.

In case you leave target type parameter blank, it will be considered the same as "any".

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
```

Automated Target

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

Automated target commands must precede an action that results in a cursor/target.

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'
```

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'
createtimer 'sample'
endif
```

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
```

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')