

# onScreenTyping v1.1

In-game on-screen typing; by Faguss (fgs.er.pl)

## 1. Requirements:

fwatch      <ftp://ftp.ofpr.info/ofpd/utls/fwatch.zip>

## 2. Usage

```
[ ] exec „onScreenWriting.sqs“                      //no arguments; works locally
```

Script is meant to display keyboard input.

Press *F1* to disable the script. Hold *L* to enable it again.

Press *F2* to clear message.

Keys - + \ [ ] ; ' , . / aren't detected by fwatch and they don't work.

Fwatch treats mouse buttons as spaces.

Overusing special keys (like numpads, home, page down etc.) may crash the game.  
The crash may also occur if message is too long.

## 3. Version history

**1.0**      (21.04.09)

First release.

**1.1**      (31.10.09)

- implemented continuous writing

## 4. Code explanation

The idea of the script is to display on the screen what `getkeys` command returns. Message string (`_msg`) is build up up by keyboard input (`_toWrite`).

<b>Define variables</b>	<pre>_msg="" _tmp="a" _caps=false _restricted=["shift","esc","up","down","left","right","ctrl","alt","tab","pageup","pagedown","home","insert","end","f3","f4","f5","f6","f7","f8","f9","f10","f11","f12"] _read=["space","enter","divide","select","subtract","add","decimal","numpad0","numpad1","numpad2","numpad3","numpad4","numpad5","numpad6","numpad7","numpad8","numpad9"] _write=["","\n","/","*","^","_","0","1","2","3","4","5","6","7","8","9"] _read2=["1","2","3","4","5","6","7","8","9","0"] _write2=["!","@","#","\$","%","^","&amp;","*","(",")"] _holdKey=0</pre>
<b>Check input and detect if user is holding the key</b>	<pre>#KeyCheck ~0.01 _keys = call loadfile "input getkeys" _toWrite=_keys select 0 ? _toWrite=="SHIFT" &amp;&amp; count _keys&gt;1 : _toWrite=_keys select 1  ? count _keys=0 : _tmp=""; goto "KeyCheck" if (_toWrite==_tmp) then [_holdKey=_holdKey+1] else [_holdKey=0] ? _holdKey=0 &amp;&amp; _holdKey&lt;20 : goto "KeyCheck" _tmp=_toWrite</pre>
<b>Check for special keys</b>	<pre>_i=0; "Format [" if (_toWrite==(_restricted select _i)) then { goto {keycheck} } "" _i] ForEach [0]; _i=_i+1 ForEach _restricted _i=0; "Format [" if (_toWrite==(_read select _i)) then [_toWrite=_write select _i] "" _i] ForEach [0]; _i=_i+1 ForEach _read ? _toWrite=="CAPSLOCK" : _caps=1_caps; goto "KeyCheck" ? _toWrite=="BACKSPACE" : goto "Backspace" ? _toWrite=="DELETE" : _msg = call loadfile format["string range %1" 1 100000], _msg]; _toWrite="" ? _toWrite=="F1" : hint "Script Frozen"; goto "Freeze" ? _toWrite=="F2" : _msg=""; _toWrite=""</pre>
<b>Add letter and display string</b>	<pre>if (!("SHIFT" in _keys) &amp;&amp; !_caps    "SHIFT" in _keys &amp;&amp; _caps) then [_toWrite = call loadfile format["string tolower %1"], _toWrite] ? "SHIFT" in _keys : _i=0; "Format [" if (_toWrite==(_read2 select _i)) then [_toWrite=_write2 select _i] "" _i] ForEach [0]; _i=_i+1 ForEach _read2 _msg=_msg+_toWrite  #Show hint _msg; player globalchat _msg; titetext [_msg,"PLAIN DOWN",0.1] goto "KeyCheck"</pre>
<b>Backspace function</b>	<pre>#Backspace _tmp = call loadfile format["string toarray %1"], _msg] _msg=""; _z=0 ? count _tmp&lt;=1 : goto "Show" #loop _msg=_msg+(_tmp select _z) ? _z&gt;=(count _tmp)-2 : goto "Show" _z=_z+1 goto "loop"</pre>
<b>Freeze script</b>	<pre>#Freeze ~2 _x=call loadfile "input getkey L" ? _x&lt;0 : _tmp="L"; hint "Script Active"; goto "KeyCheck" goto "Freeze"</pre>

### Define variables

`_msg` – message

`_tmp` – last input

`_caps` – indicates if *CAPS LOCK* is on

`_restricted` – these values won't be displayed

`_read` – these values will be replaced with string from `_write`; e.g. *ENTER* with `\n`

`_read2` – these values will be replaced when *SHIFT* is pressed

### Check input

Keyboard input is saved to `_keys` array. `_toWrite` is array's first element OR second if *SHIFT* is first.

### Loop expanding the message if user is holding the key

Variable `_holdKey` is used to make it start after 0.2 second.

### Special keys

Check for keys listed in `_restricted`, `_write` and for these in conditions.

Value returned by `getkeys` is always uppercase. It's lowered if *SHIFT* isn't pressed and *CAPS LOCK* is off OR *SHIFT* is pressed and *CAPS LOCK* is on.

Then look for values listed in `_read2` (numbers).

### Backspace

Message is converted to array. Then merged back without the last character.

### Freeze

Loop checking if `L` is pressed.