



**INPUT**  
**DADDY**

## Contents

|   |    |
|---|----|
| What is InputDaddy:.....                          | 3  |
| Lightweight and simple .....                      | 3  |
| Getting started .....                             | 3  |
| Learning by doing .....                           | 5  |
| Feature walkthrough .....                         | 10 |
| Main User Interface.....                          | 10 |
| Toolbars.....                                     | 11 |
| Using the modifiers .....                         | 12 |
| Using Environment Variables .....                 | 13 |
| Using the Mouse tool .....                        | 14 |
| Script execution and editing.....                 | 16 |
| Script Recording and Playback.....                | 16 |
| Settings.....                                     | 18 |
| General .....                                     | 19 |
| Key bindings .....                                | 20 |
| Setting Key bindings .....                        | 20 |
| Script Recording .....                            | 22 |
| Customizing the look and feel of InputDaddy ..... | 23 |
| Loop control .....                                | 24 |
| Output window .....                               | 24 |
| About us .....                                    | 25 |
| Behind the scenes .....                           | 25 |
| Script storage and script modification.....       | 25 |
| InputDaddy can't do it all! .....                 | 26 |
| Running in the admin account.....                 | 26 |
| More examples on YouTube.....                     | 26 |
| Q&A.....  | 27 |

# What is InputDaddy:

Do you type a lot, or would you like a fast, easy-to-use alternative to entering the same data and clicking the same buttons over and over again?

InputDaddy is a keyboard and mouse automation tool that lets you automate mundane or monotonous tasks on a windows computer. InputDaddy is intended for IT non-professionals that just want the job done without having to learn new script codes and tools. Through an intuitive user interface InputDaddy lets you create new automation scripts or execute existing ones with ease. The interactive debugging capabilities of InputDaddy lets you see what is going on at any point during script execution, making it much easier to figure out what is going wrong with your script.

As most other windows automation applications, InputDaddy works by sending keystrokes or mouse instructions to the current active window. Additionally, InputDaddy can set a particular window as the active thereby allowing you to do stuff on any window or between windows as you see fit.

## Lightweight and simple

InputDaddy is a lightweight product consisting only of the executable and an optional Settings file if you want your own user interface. InputDaddy requires no installation, just unzip the files to a folder and run it. You can run any number of instances of InputDaddy, just double click the executable to start another.

## Getting started

InputDaddy is an intuitive and efficient automation tool to use, intuitive because the user interface pretty much tells the whole story and efficient because it lets you do the same repetitions with minimal effort. You can either construct the scripts from scratch or simply use the script recorder feature to add keyboard and mouse gestures, or you can mix the modes, create some commands on your own, recording another part, and add or modify the script before executing it, InputDaddy makes it simple to use both methods.

Let's have a look at the User Interface in its initial state. This is what you should see when you double click InputDaddy:



The toolbars in the upper part of the application lets you quickly add a script command using the green plus button. Each toolbar can be expanded or collapsed to optimize your experience and efficiency.

Toolbars can also be completely hidden from the settings dialog.

You can tailor you own interface by modifying the DefaultSettings.xml file.

The script execution in the lower part of the application allows running or stepping through scripts.

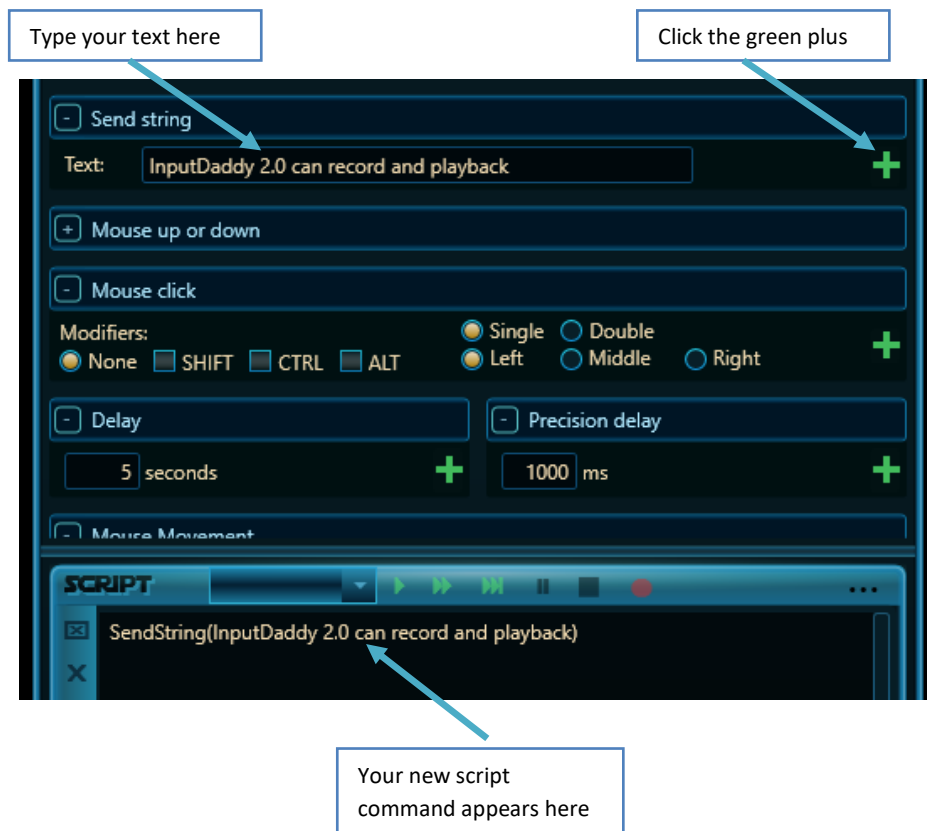
Controls at the left side of the script window allow manipulation of script order and the Loop options dialog control how scripts are executed.

The output window contains important messages and metrics.

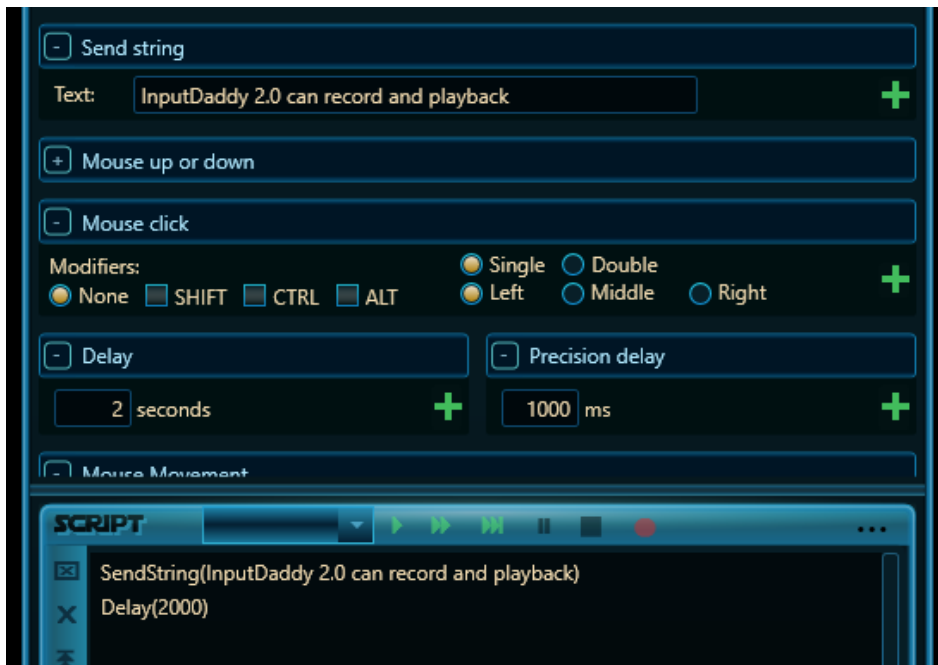
## Learning by doing

Easiest way is to jump right in and create a simple script that types some stuff into notepad. In this very simple example, we will show you how to insert a text into Notepad over and over again. Go ahead and open a Notepad instance or your favourite typing application whatever that is and open an instance of InputDaddy as well.

Locate the toolbar called “Send string” and type in some text, when you are ready press the green plus icon to add a script command to the script window



Now, locate the Delay tab and change the value from 5 seconds to 2 seconds and press the green plus icon to add the script command.

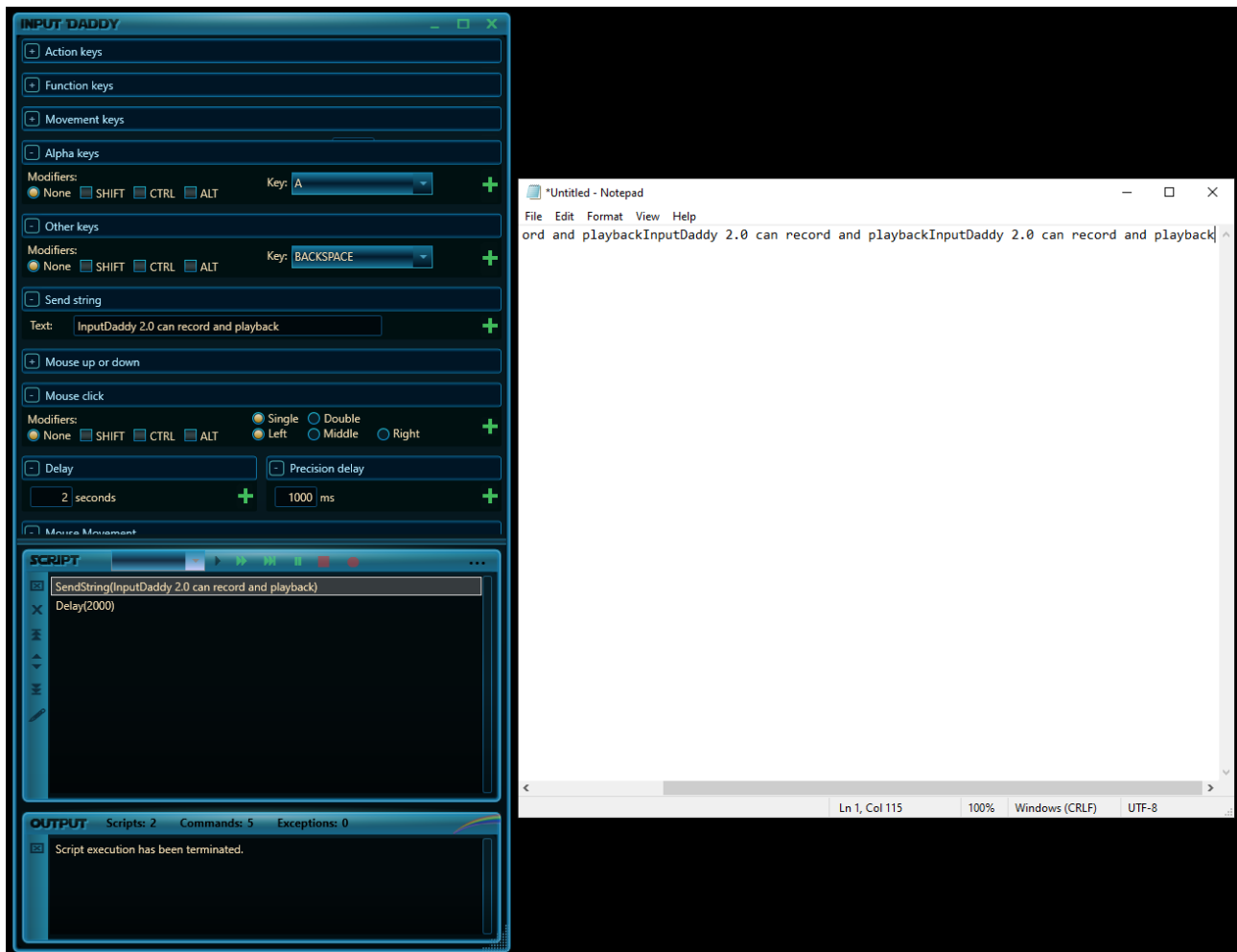


That is, it! We are ready to run our script, so get your typing application out, let us call him Notepad for now.

Press the green arrow in the script window to execute your script and then click on you Notepad instance,



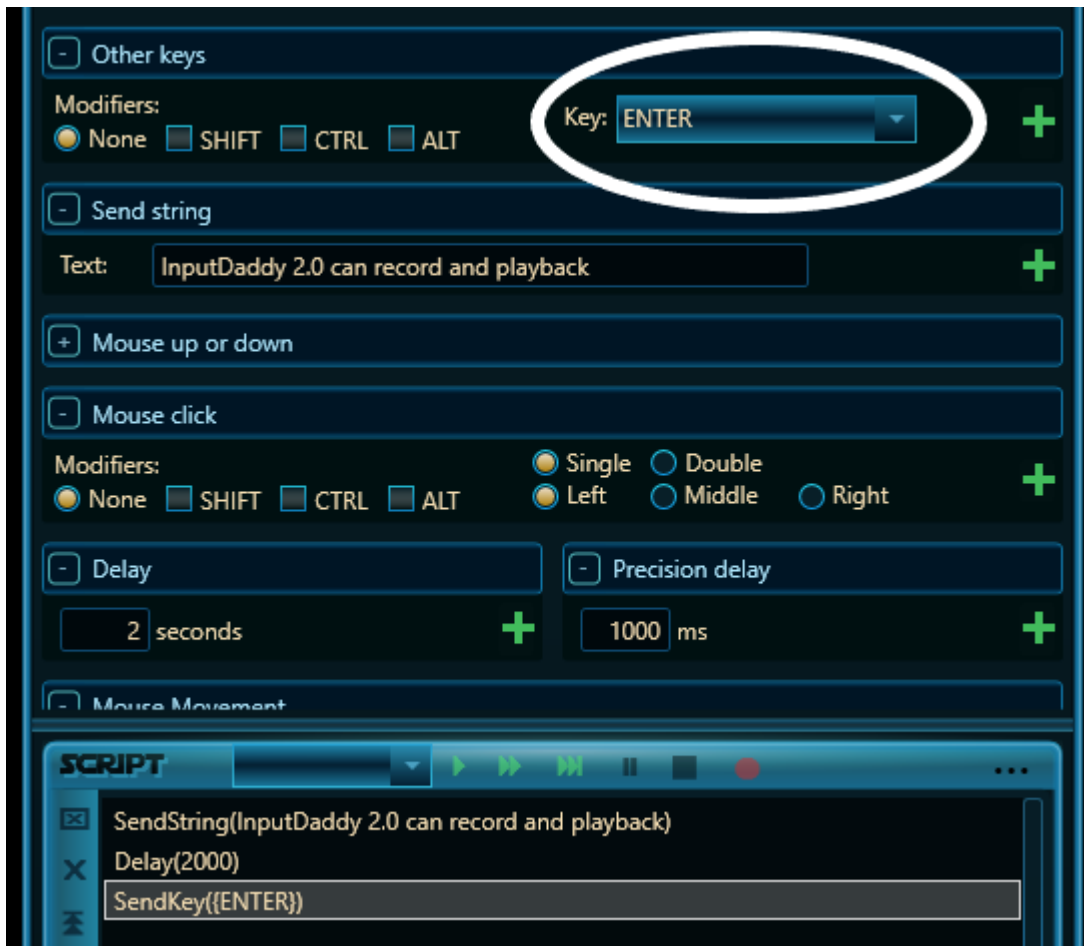
you should see InputDaddy typing string at every two seconds. Something like this



Click the red square to stop script execution.



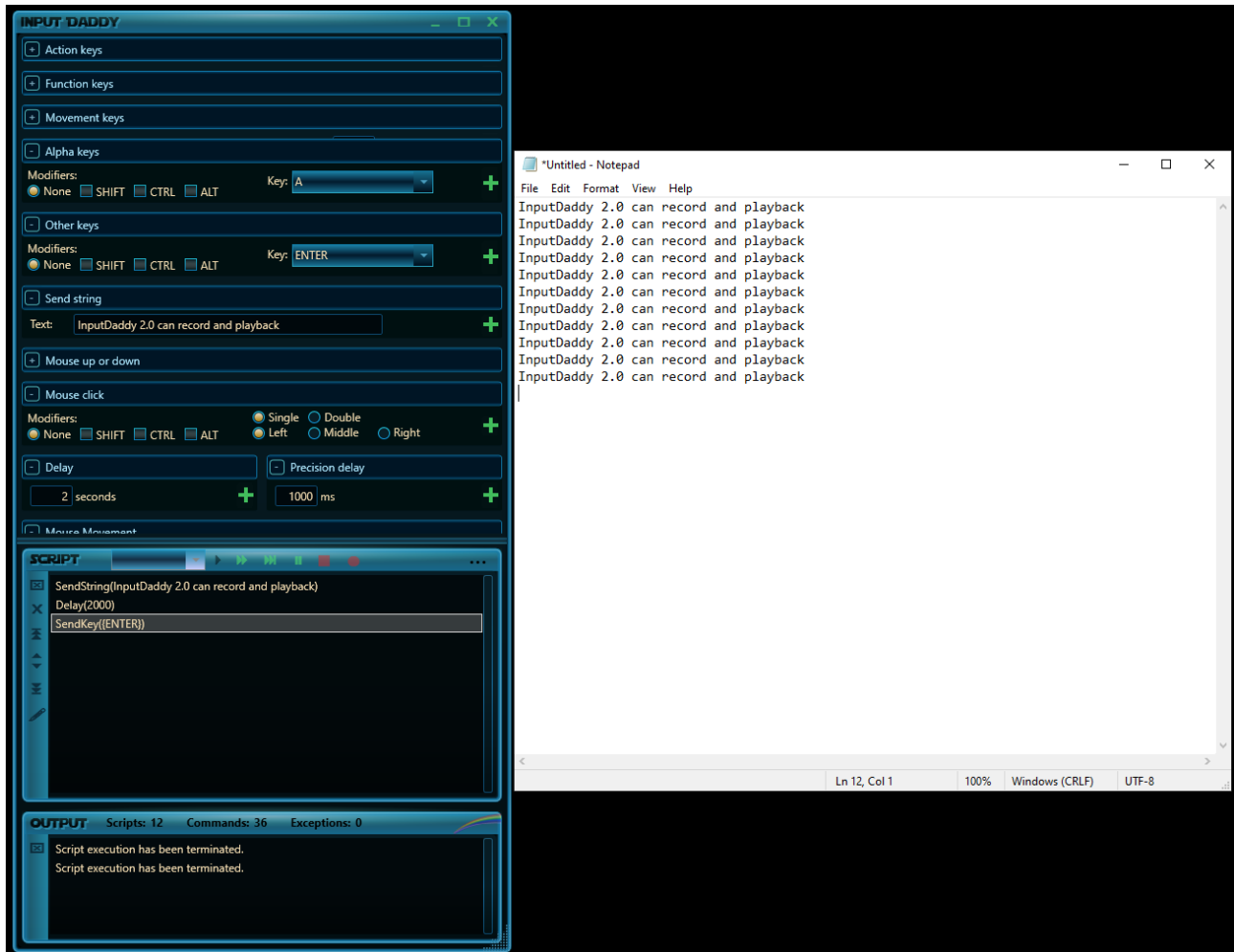
Notice that the text appears on one line as we did not insert any line breaks into our script. Let's do that quickly. Locate the "Other Keys" toolbar and select "ENTER" from the key drop down and press the green plus icon in the toolbar.



Press the green arrow in the script window to execute your script and then click on your Notepad instance, you should see InputDaddy typing string at every two seconds and then pressing the enter key.



Something like this...

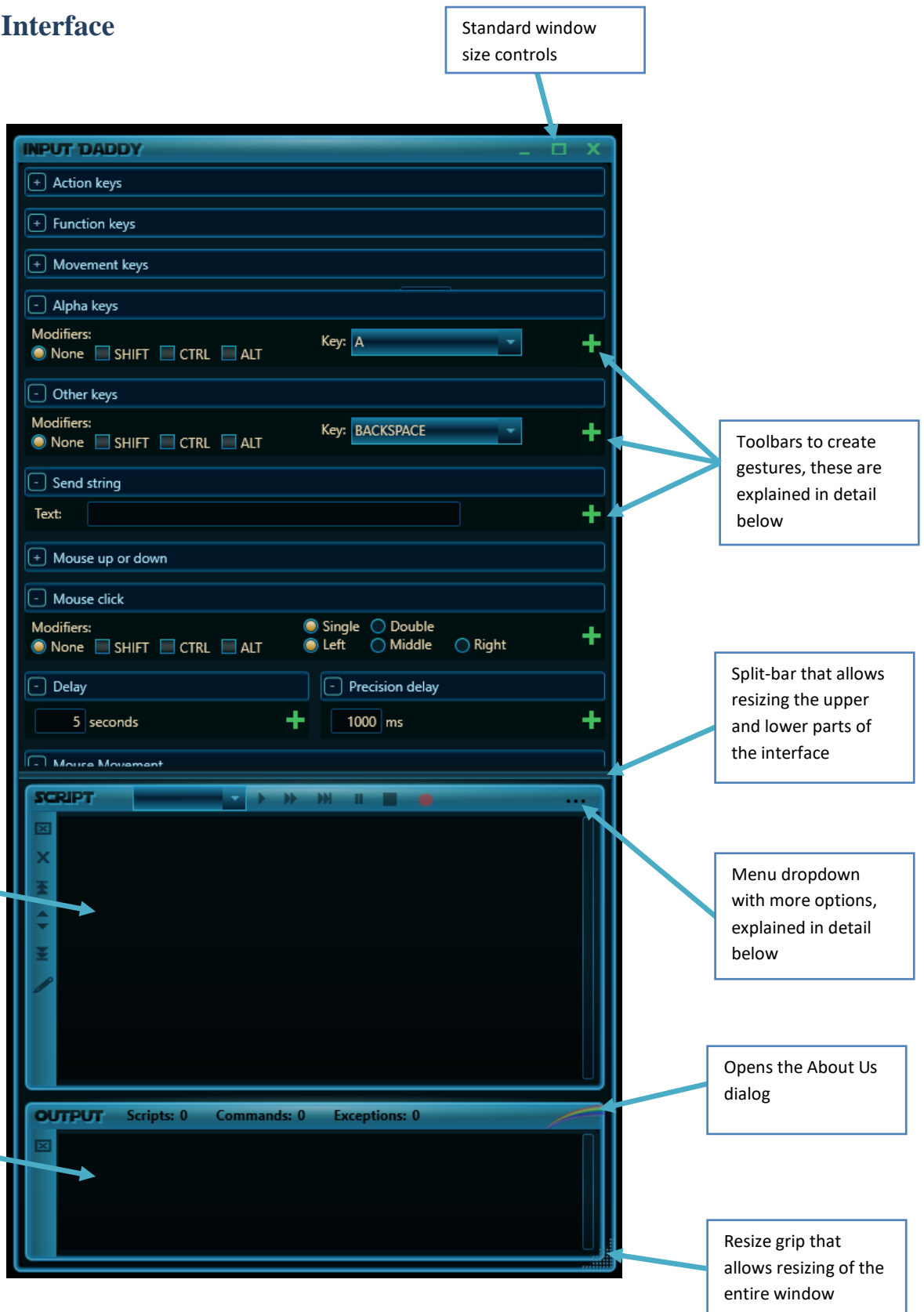


There we are, job's a good'un.


Play and experiment yourself, it isn't rocket science.

# Feature walkthrough

## Main User Interface



## Toolbars

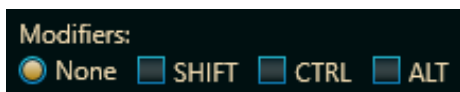
| Toolbar  | Description   |
|--|---|
| <div> <div>-</div> <div>Action keys</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div> <input checked="" type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input type="radio"/> 0 </div> </div> <div>+</div> </div>  | Action Keys: Adds numeric inputs 0-9 either as normal keys or with modifiers. i.e. SHIFT-5  |
| <div> <div>-</div> <div>Function keys</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div> <input checked="" type="radio"/> F1 <input type="radio"/> F2 <input type="radio"/> F3 <input type="radio"/> F4 <input type="radio"/> F5 <input type="radio"/> F6 <input type="radio"/> F7 <input type="radio"/> F8 <input type="radio"/> F9 <input type="radio"/> F10 <input type="radio"/> F11 <input type="radio"/> F12 </div> </div> <div>+</div> </div> | Function Keys: F1 - F12 either as normal key or with modifiers, i.e. CTRL-SHIFT-F9.   |
| <div> <div>-</div> <div>Movement keys</div> <div> <input checked="" type="radio"/> Left <input type="radio"/> Right <input type="radio"/> Up <input type="radio"/> Down <div>KeyDown time: 1000 ms</div> </div> <div>+</div> </div>  | Movement Keys: Arrow keys pressed a specific number of milliseconds   |
| <div> <div>-</div> <div>Alpha keys</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div>Key: A</div> </div> <div>+</div> </div>   | Alpha Keys: Alphabetical characters and modifiers   |
| <div> <div>-</div> <div>Other keys</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div>Key: ENTER</div> </div> <div>+</div> </div>   | Other Keys: Other keys and modifiers.   |
| <div> <div>-</div> <div>Send string</div> <div>Text: InputDaddy 1.2 rocks harder</div> <div>+</div> </div>   | Send string: Allows sending of multiple characters in one command.  |
| <div> <div>-</div> <div>Mouse up or down</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div> <input checked="" type="radio"/> Up <input type="radio"/> Down <input checked="" type="radio"/> Left <input type="radio"/> Middle <input type="radio"/> Right </div> </div> <div>+</div> </div>  | Mouse up down: Set the mouse up/down state for the left, right or middle mouse buttons and modifiers for the same.  |
| <div> <div>-</div> <div>Delay</div> <div>2 seconds</div> <div>+</div> </div>   | Delay: Wait for a specific number of seconds. Can be affected by the Delay Noise setting.   |
| <div> <div>-</div> <div>Mouse Movement</div> <div> <div>X: 1000 Y: 1000</div> <div> <input checked="" type="radio"/> Absolute move <input type="radio"/> Relative move </div> <div>  </div> </div> <div>+</div> </div>  | Mouse Movement: Move the mouse to a screen coordinate location or relative coordinates. You can open the mouse tool by clicking the target icon, see below. |

|   |  |
|---|--|
| <div> <div>- Precision delay</div> <div>1000 ms</div> <div>+</div> </div>   | <p>Precision delay: When you need more fine-tuned delays. Can be affected by the Delay Noise setting.</p>                      |
| <div> <div>- Set window focus</div> <div>Window title: <input type="text"/></div> <div>Use Window Handle <input type="checkbox"/></div> <div>+</div> </div>   | <p>Set window focus: Sets the specified window or windows handle to be the foreground window and have input focus.</p>         |
| <div> <div>- Mouse click</div> <div> <div>Modifiers:</div> <div> <input checked="" type="radio"/> None <input type="checkbox"/> SHIFT <input type="checkbox"/> CTRL <input type="checkbox"/> ALT </div> <div> <input checked="" type="radio"/> Single <input type="radio"/> Double <input checked="" type="radio"/> Left <input type="radio"/> Middle <input type="radio"/> Right </div> </div> <div>+</div> </div> | <p>Do single or double clicks with left, right or middle buttons in the current cursor location.</p>                           |
| <div> <div>- Environment Variables</div> <div>Variable <input type="text"/></div> <div>+</div> </div>   | <p>Send dynamic content, for example, loop count or the current Time. Environment variables are explained in detail below.</p> |

## Using the modifiers

Most key and mouse actions can have one or more modifiers active when pressed or clicked. SHIFT, CTRL and ALT modifiers can be applied alone or with multiple modifiers to the action. This allows you to send sequences like ALT-F4, CTRL-SHIFT-F7 or do mouse up or down action with the SHIFT key down.

The modifiers look like this:



You can select any combination of SHIFT, CTRL and ALT modifiers by clicking the appropriate checkboxes, or remove all modifiers by clicking the “None” radio-button.


## Using Environment Variables

Most of the time, all we want to do is, to send the same known sequences of keystrokes and mouse actions to a target application, but every now and again you may face the need to send a dynamic string or number to an application. For instance, if you were to save a large number of files giving them a name in the process you could use the #LOOPCOUNT action to generate names like file\_01, file\_02, file\_03 etc.

The following Environment Variables are currently supported:

| Variables         | Descript  |
|-------------------|---|
| #DATE             | Insert the current date in the using the current date format of your windows, for example 23/11/2020. |
| #TIME             | Inserts the current time using the current time format of your windows. For example, 11:04:12         |
| #DAY              | Inserts the day number, for example 28  |
| #MONTH            | Inserts the month number, for example 3   |
| #YEAR             | Inserts the Year.   |
| #LOOPCOUNT_X      | Inserts the current loop iteration as an unformatted number, e.g. 1,2,3,4                             |
| #LOOPCOUNT_0X     | Inserts the current loop iteration as a two-digit formatted number, e.g. 01, 02, 03, 04               |
| #LOOPCOUNT_00X    | Inserts the current loop iteration as a three-digit formatted number, e.g. 001, 002, 003, 004         |
| #LOOPCOUNT_000X   | Four-digit formatting   |
| #LOOPCOUNT_0000X  | Five-digit formatting   |
| #LOOPCOUNT_00000X | Six-digit formatting, 000001, 000002 etc  |

## Using the Mouse tool

In order to more precisely make mouse clicks or move the mouse you can use the Mouse Tool from the Mouse Movement Toolbar it looks like this: 

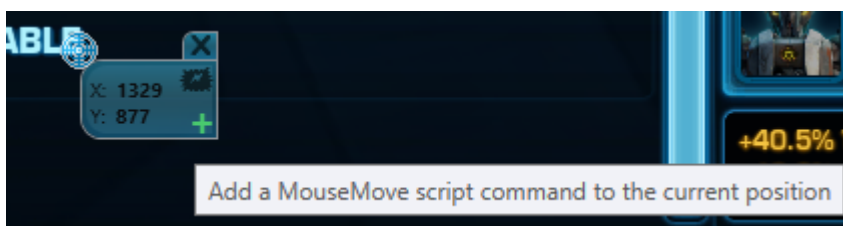
Once you click the target icon a small window will appear close to it:



The Mouse-tool consists of the target icon and the current coordinates of the centre of the target icon. You can drag this window around by holding down the left mouse over the tool and drag it, just like any other normal drag operation. While you move the mouse-tool it will continuously update the x and y according to its screen position.



The mouse-tool also contains a close button to dismiss the window and a plus button to directly add a `MouseMove(x,y)` command to the script execution window at the current location.

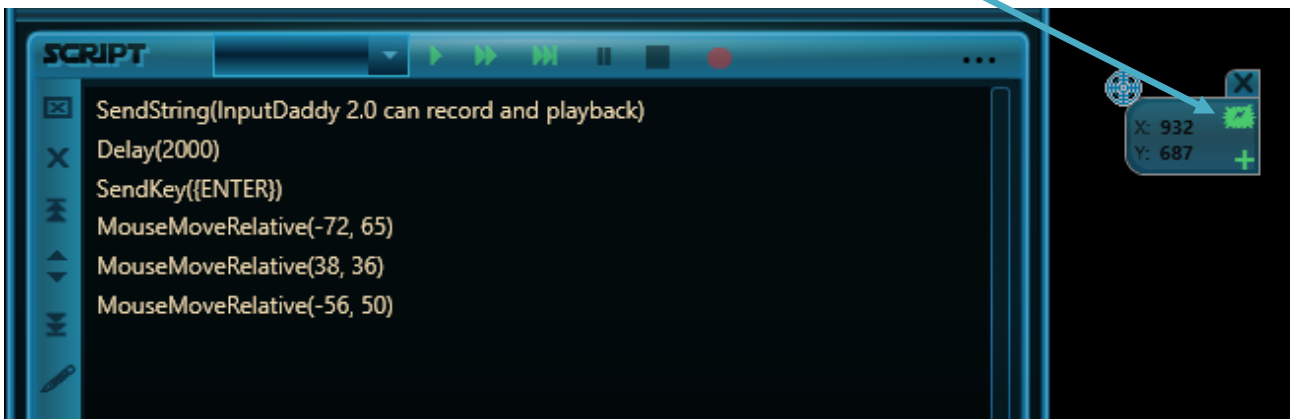


After pressing the green plus a `MouseMove` command is added directly to the script



The Enter key is bound to the add command so you can move the tool around and press enter whenever you want a mouse move command added to the script.

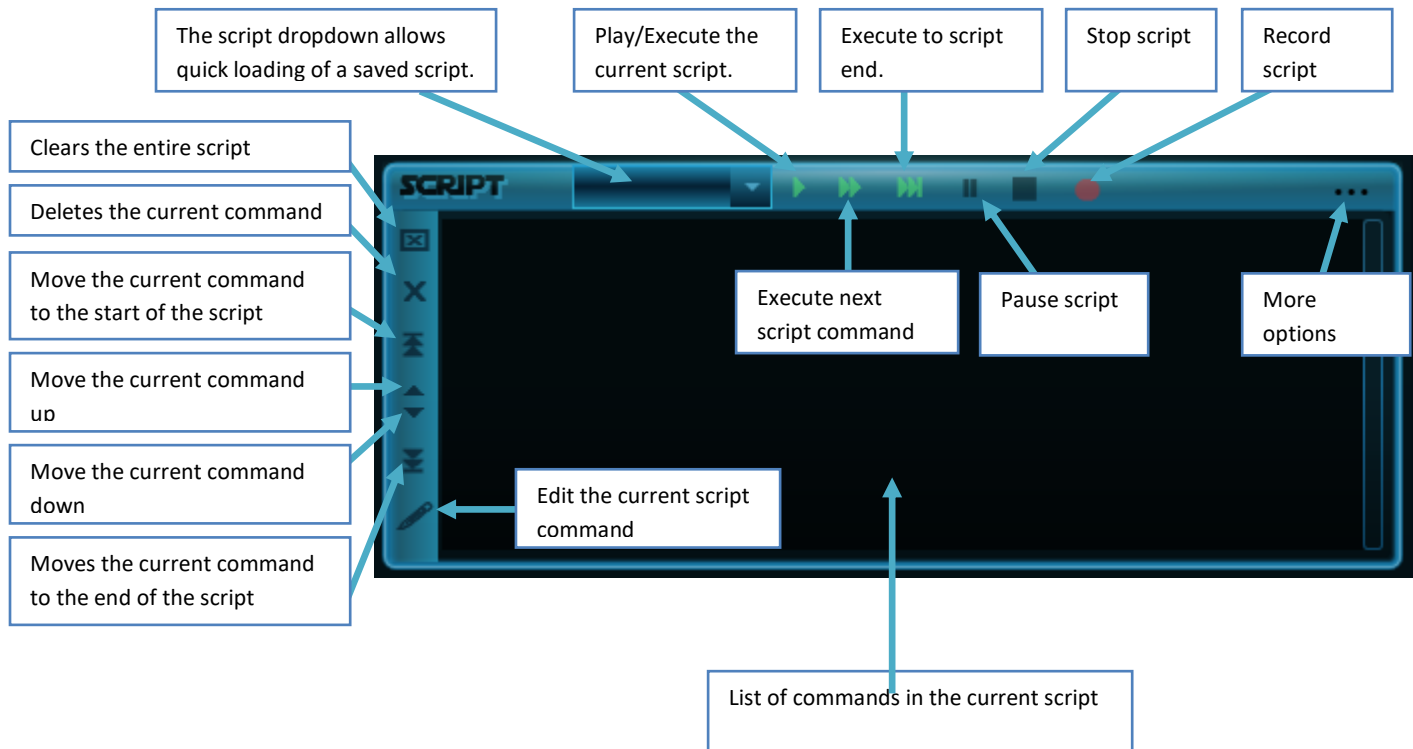
If you activate relative mode by clicking the cloud with a lightning in it the tool will switch to relative positioning and add relative mouse move commands to the script window.



You can turn off relative mouse moves by clicking the lightning cloud again.

NOTE: The mouse tool window is topmost and will float a top everything in windows, you can use it anywhere. The window is modeless allowing you to add multiple move commands but must be closed manually when you are done using it.

## Script execution and editing



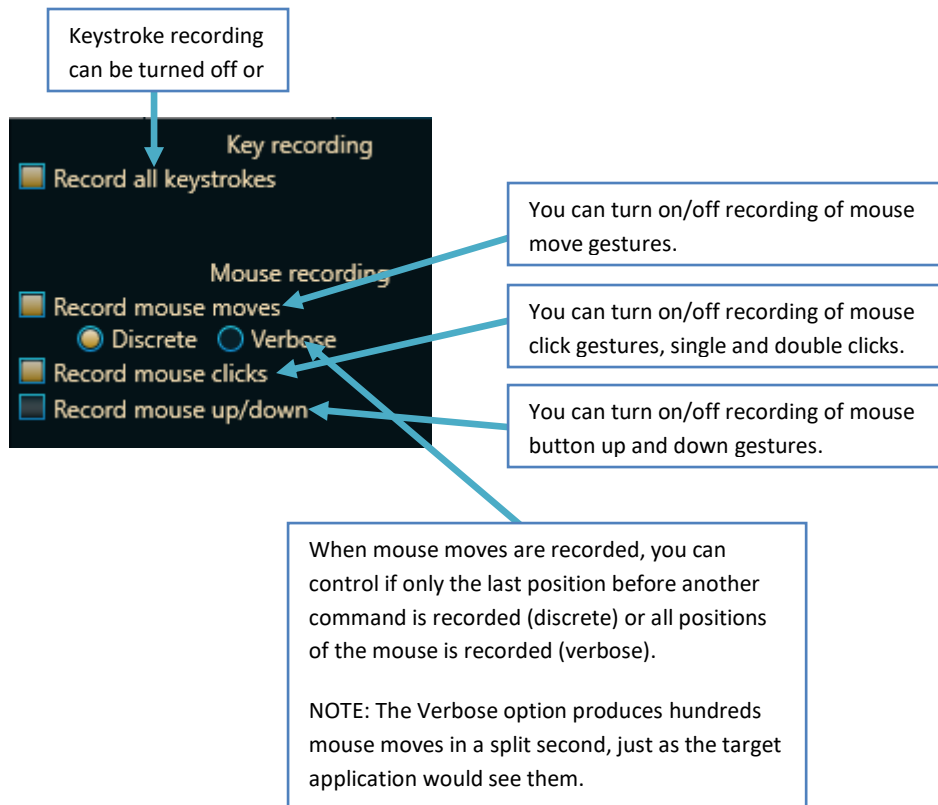
## Script Recording and Playback

In many scenarios it's easier to simply let InputDaddy record your gestures and then play them back when you need it. This feature is very simple to use, you just press the record button and InputDaddy starts adding commands to your script as you go through the gestures.



When you are done press the record button again and no further commands will be added automatically. You can then edit or insert manual parts and start recording again should you need it. When it comes to what to record it gets a bit more complicated. To begin with InputDaddy has these settings for what to record:

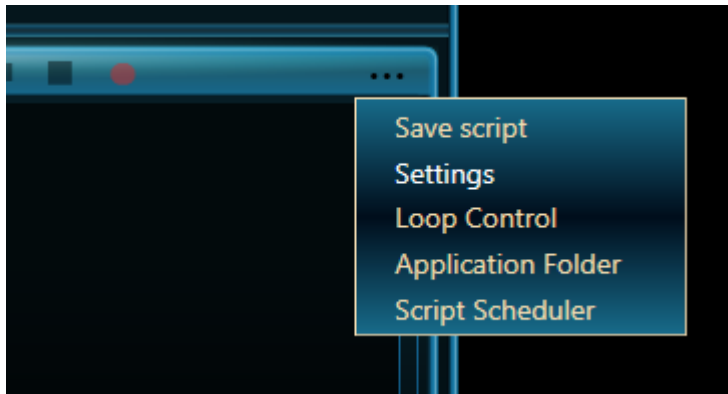




These defaults are what is most commonly used when recording scripts, but you can experiment and see what suits your need the best.

A number of options are tucked away in the More menu dropdown that you find on the right side of the Script window. The menu is 3 dots ("..."):

If you click the 3 dots a menu will drop down:



From here you can

- Quick save your scripts
- Open the settings dialog
- Open the Loop Control dialog
- Open the application folder in an explorer window to quickly get to your stuff

Each of these options are explained in detail below.

## Settings

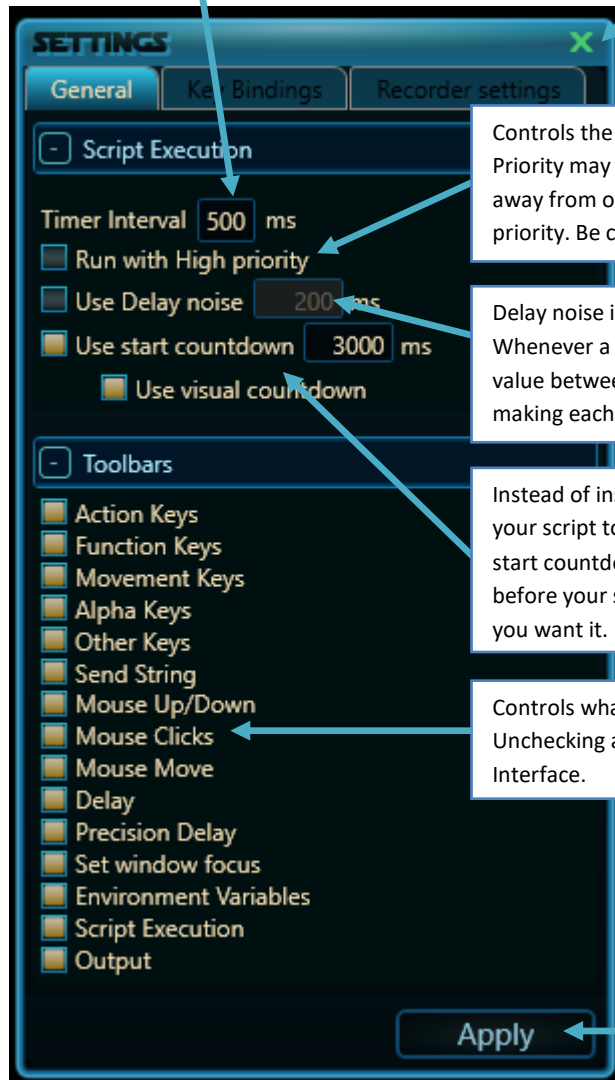
The settings are divided into 3 parts:

- General Options
- Keybinding Options
- Script Recording Options

## General

The timer interval controls how often InputDaddy executes scripts. In this case InputDaddy will check every 500 milliseconds or every half a second to see if a new command must be executed. If your script requires high precision in the command timing you may wish to lower this value.

Closes the settings dialog without making any changes



Controls the priority of InputDaddy, running with High Priority may make InputDaddy more responsive but will take away from other applications running with normal or low priority. Be careful with this setting.

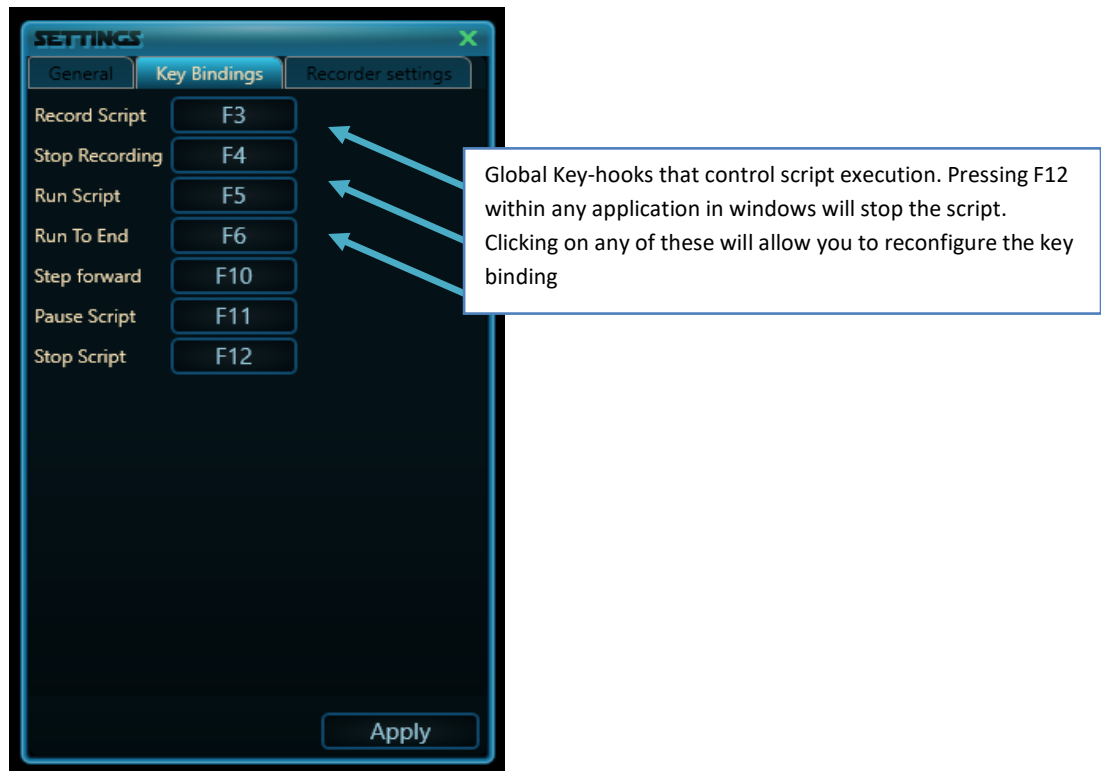
Delay noise is used to closer mimic human behavior. Whenever a delay is encountered in your script, a random value between -200 and 200 is added to the delay length, making each delay slightly different.

Instead of inserting delay commands at the beginning of your script to set focus where you want it, you can use a start countdown. The countdown will be executed once before your script starts and allows you to set focus where you want it.

Controls what toolbars are show in the main User Interface. Unchecking a toolbar will completely hide it from the User Interface.

Closes the settings dialog, applying the changes.

## Key bindings

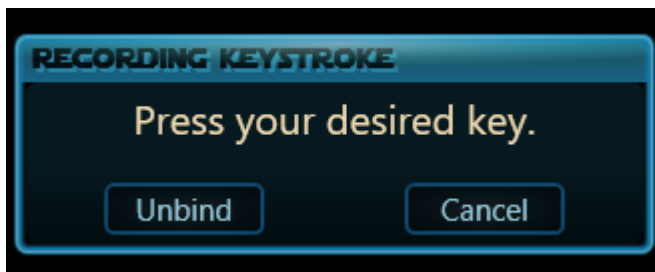


## Setting Key bindings

Global key hooks are used only when you want to step through your script or pause or stop it while working in another application. Because keystrokes are always sent to the application that has the input focus debugging scripts can be almost impossible if you have to click the commands inside InputDaddy as windows automatically transfers input focus to InputDaddy and the application is now sending keystrokes or mouse actions to itself. Instead we can assign key that we monitor on a system wide basis and this way we can maintain focus within the application we are working in while still being able to single-step the script, pause or stop it. The reason the global hooks are user defined is because some applications react to certain keystrokes or virtual keys (F12 is a virtual key). For example, if you are working in Notepad then none of the printable characters would be suitable as global hooks, they are supposed to go into Notepad not InputDaddy. You can do both in some cases, but most often you want to find keys that do not affect the application and use those. In notepad F11 and F12 do nothing, those are good candidates. In a paint like application normal printable characters could easily be used. As it all depends on the situation and the application you are automating you can reconfigure these key binding from within the settings dialog.



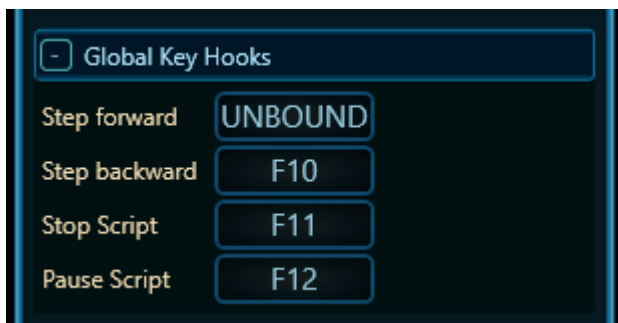
Clicking on any on the buttons will bring up the Recorder dialog, it looks like this



Once you press a key (for example {SPACE}) the dialog will close automatically and the new binding appears.

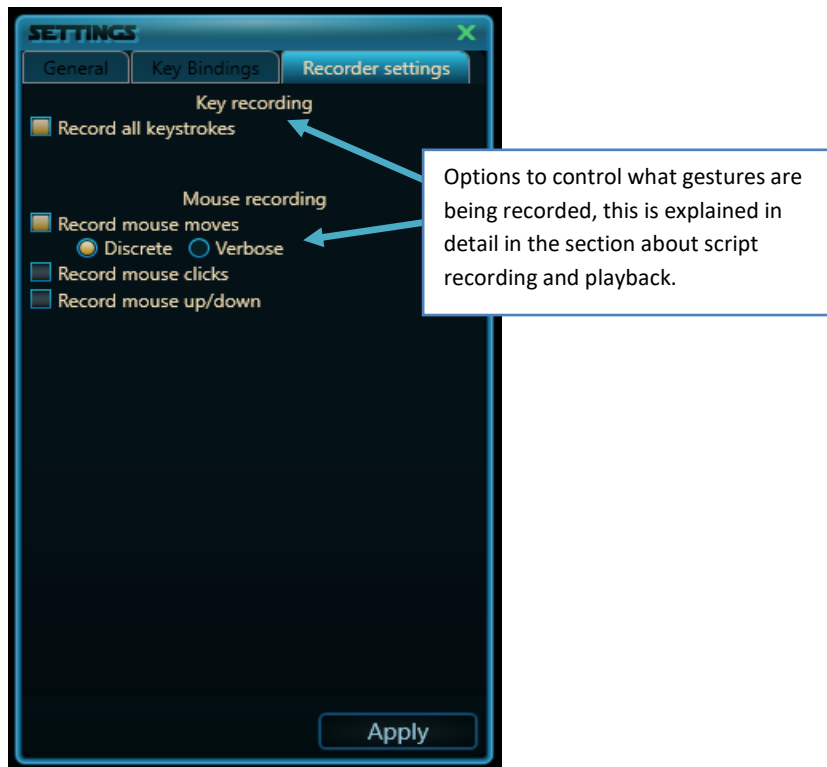


If you press the Unbind button then that actual will have no keys associated with it.



Pressing Cancel dismisses the dialog making no changes.

## Script Recording



Each of these options and how they affect your script recording is explained above in the chapter about script recording and playback.

## Customizing the look and feel of InputDaddy

When starting the application InputDaddy will attempt to locate a DefaultSetting.xml file in its own directory, if one is found then InputDaddy will load it and start up with those settings. If no DefaultSetting.xml file can be found InputDaddy will load with its own predefined defaults. This allows you to customize InputDaddy to your own needs or even have multiple configurations for InputDaddy depending on the task you are solving. You will however need to keep track of the settings file yourself, start by making a copy of the one that comes with the product in case you get into problems with your edits. The format is fairly straightforward, each toolbar has a Visible and an Expanded setting, these settings control if the toolbar is shown at all and if so if it is expanded or collapsed. You can always get the toolbars back from the settings dialog, so don't worry about hiding toolbars. The last part of the settings is the more general settings like your global keyhooks and other general stuff.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<InputDaddySettings xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
<ActionKeys>
```

```
<Visible>true</Visible>
```

```
<Expanded>false</Expanded>
```

```
</ActionKeys>
```

```
<FunctionKeys>
```

```
<Visible>true</Visible>
```

```
<Expanded>false</Expanded>
```

```
</FunctionKeys>
```

. (middle content removed for better overview)

```
<StepForwardKey>F9</StepForwardKey>
```

```
<StepBackwardKey>F10</StepBackwardKey>
```

```
<StopScriptKey>F11</StopScriptKey>
```

```
<PauseScriptKey>F12</PauseScriptKey>
```

```
<ShowScript>true</ShowScript>
```

```
<ShowOutput>true</ShowOutput>
```

```
<TimerInterval>500</TimerInterval>
```

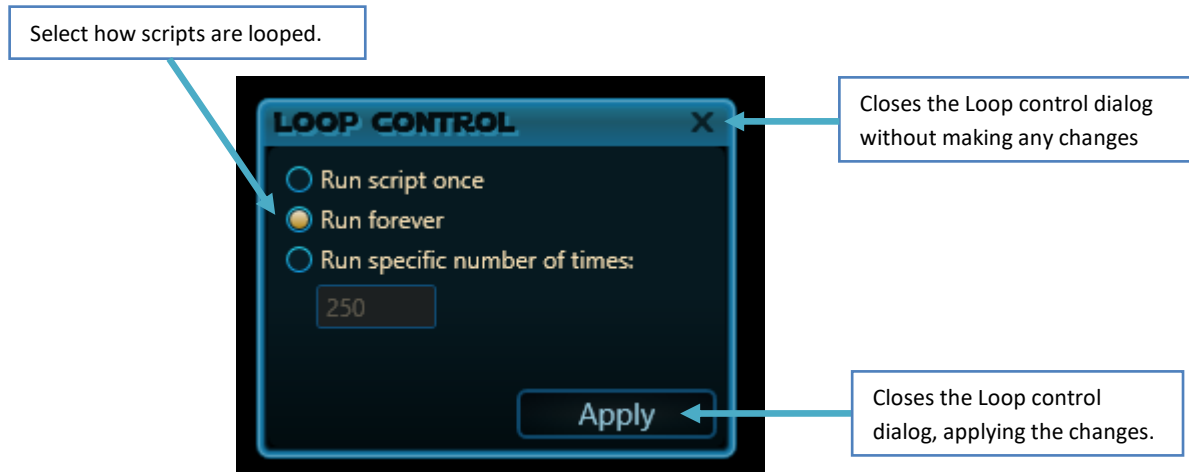
```
<UseDelayNoise>false</UseDelayNoise>
```

```
<DelayNoiseMilliseconds>200</DelayNoiseMilliseconds>
```

```
</InputDaddySettings>
```

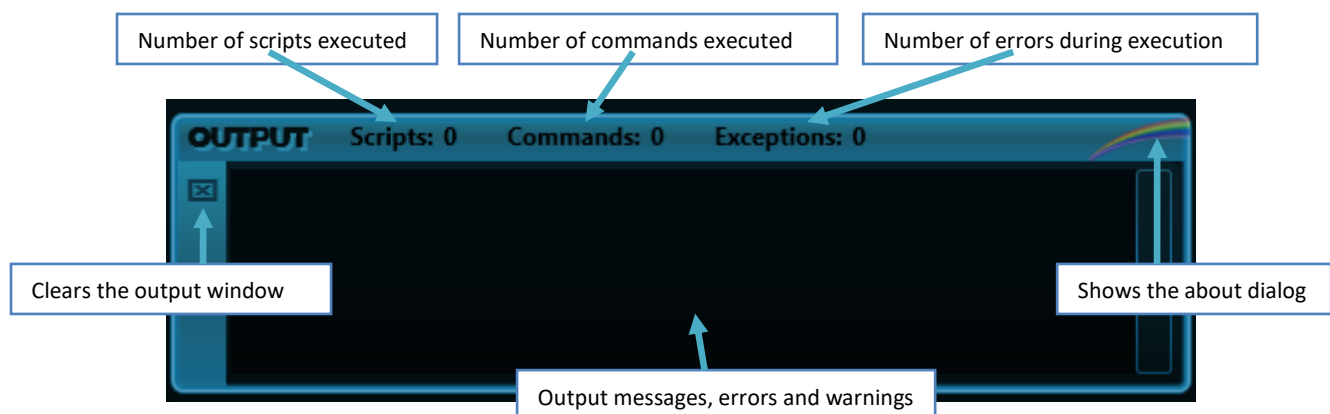
## Loop control

By default, InputDaddy will loop you script forever, but you can control this using the loop control dialog. This lets you either run the script once (same as Run To End), forever (same as default), or a specific number of times defined by you.



## Output window

The output window shows you how many script commands InputDaddy has executed so far, how many complete script loops and number of exceptions during that time (hopefully that will always remain at zero). It also contains vital InputDaddy messages like when the script execution is stopped, paused or completed.





## About us

From the about box you can see the build number and access [Rainbow Consult](#) homepage.



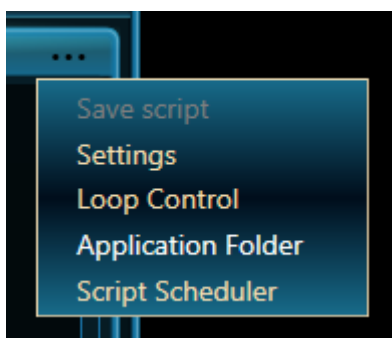
## Behind the scenes

### Script storage and script modification

When saving scripts, they are always stored in the same location as the InputDaddy executable, make sure that you have write permissions to the folder InputDaddy resides in. Similarly, InputDaddy will look in its own folder for scripts during start-up and display them for easy selection in the Script Execution window. InputDaddy has no concept of a working folder and only looks in the current folder.



You can click the application folder shortcut in the more options dropdown to open the application folder



InputDaddy scripts are uncomplicated txt files that look exactly like what you saw in the script execution window when you created your script. You can inspect or edit the files with any plain text editor.

## InputDaddy can't do it all!

You can get most jobs done simply using the user interface and never wonder about syntax or funky script commands but there are times when the job may require more. The core SendKeys commands are not restricted to what InputDaddy can do. They use an underlying implementation from Microsoft by the same name and understand the exact same syntax and commands. You can read more about MS SendKeys here:

<https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.sendkeys.send?view=netframework-4.8>

## Running in the admin account

Some applications that you wish to automate run as administrator in order to make necessary modifications to your computer. **Whenever you wish to automate software that runs with admin privileges then InputDaddy must also be run with admin permissions.** In this case, right click the InputDaddy Icon and choose "Run as administrator". If you try to automate a piece of software that is running with admin privileges without starting InputDaddy as admin as well, nothing will happen in the receiving software because windows actively blocks the connection. This is also applicable when using global key hooks for debugging your script, if the application you are working in run as admin then windows will not allow you to monitor anything unless InputDaddy also runs in the admin account.

## More examples on YouTube

We will be making and posting Videos about different day to day challenges and how we used InputDaddy to solve them on YouTube, search for InputDaddy. These will be Championed by Jes and featured on his channel, look for "JesTheBestDog". There will be more postings and on more advanced topics in the future, for now there's these starting tutorials and some very MMO(game) specific ones as well.

InputDaddy Lesson 1

[https://www.youtube.com/watch?v=xcT\\_nFYa5IE](https://www.youtube.com/watch?v=xcT_nFYa5IE)

Using InputDaddy to automate Guild Wars 2

<https://www.youtube.com/watch?v=r4ZNdKz3YQA>

Using InputDaddy with SWTOR to create a level 50 companion

<https://www.youtube.com/watch?v=9k-0XCu41CU>

What's New In InputDaddy 1.2

<https://www.youtube.com/watch?v=zWHLoAiBr5k>

Teaching InputDaddy to Draw

[https://www.youtube.com/watch?v=KcVIHe\\_DdWc](https://www.youtube.com/watch?v=KcVIHe_DdWc)

Creating Professional Animations with Minimal Effort PART1

<https://www.youtube.com/watch?v=1blda7cP-ww>

Creating Professional Animations with Minimal Effort PART2

<https://www.youtube.com/watch?v=HcDjTDEjIOg>

Creating Professional Animations with Minimal Effort PART3

<https://www.youtube.com/watch?v=9TVs396-pQ>

## Q&A

**Q: Does it write to my hard disk or modify anything else?**

A: If you choose to save the script you are working on, then InputDaddy will attempt to save the text file in its own folder. Other than that, no, InputDaddy does not write any files to your drive or to your registry. If you want to limit the applications access to your drive, turn of write permissions either for InputDaddy all together or for the application directory.

**Q: Is it really free and why?**

A: Yes, the free version of InputDaddy is really free, we collect no information what so ever to support some obfuscated business model nor do we require you to in any way register the product or give up your email to get it. InputDaddy is free simply in order for us to shamelessly promote our professional version.

**Q: What is the SHA256 code for and how do I verify it?**

A: The SHA256 on the Rainbow Consult webpage is a MAC (Message Authentication Code) that ensures the executable hasn't been tampered with. Upon download use a free or commercial SHA256 verification

product to verify the integrity of the executable, if the SHA256 fails do NOT run the executable. It's a bit like inspecting the actual content of a package before you buy it in the shop. Accepting Software, freeware or shareware from the internet without checking its inner content is asking for trouble, we urge you never to do so, not with our software nor with any other software.

You should always verify MACs if you have them, and you should either write a SHA256 checksum algorithm yourself or get a commercial or free product that can do it for you. Relying on older checksum algorithms like SHA1 or MD5 is a risk, as these are not collision resistant and ultimately doesn't offer you any real guarantees that tampering hasn't taken place.

**Q: Who is Jes?**

A: Jes is my dog.