Contents

[Introduction 1](#_Toc72742549)

[features 1](#_Toc72742550)

[Installing and running tat 1](#_Toc72742551)

[The configuration file 2](#_Toc72742552)

[configuration file basics 2](#_Toc72742553)

[Hello world 3](#_Toc72742554)

[running automation actions 3](#_Toc72742555)

[Automating key presses 3](#_Toc72742556)

[Special characters 3](#_Toc72742557)

[Automating mouse actions 5](#_Toc72742558)

[Mouse syntax 5](#_Toc72742559)

[example 5](#_Toc72742560)

# Introduction

the tat(typing automation tool) is an application that automates user tasks such as mouse clicks and keyboar strokes(typing), It should be able to automate most repetitive tasks.

With some creativity anything can be automated using this tool such as filling in timesheets, typing and sending emails, social network activity(posting, liking, commenting….etc) and any other action that does not require user intervention.

# features

mouse click emulation

keypress emulation

color recognition

scriptable

loops

# Installing and running tat

Cp tat to the /usr/bin directory

To run tat you simply run tat from any terminal.

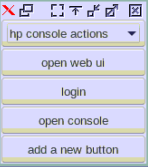
To create a menu entry for tat please refer to your window manager documentation.

# The configuration file

## configuration file basics

the tat reads its configuration from a hidden file in the home directory of the user under ~/.tat.config.

to create an automation action we need to create a set and a button in that set, a button is a series of automation actions while a set is a simple logical grouping of buttons, example:



In this example we are using the hp console actions set(indicated by the drop down menu selection) which has 3 buttons, 1 to automate opening the web UI, another to login and a third to open the machine console.

The syntax to create this interface would be:

set “hp console actions”

button “open webui” “<automation actions>”

button “login” “<automation actions>”

button “open console” “<automation actions>”

to create a second set you use the same syntax after the buttons of the first set:

set “hp console actions”

button “open webui” “<automation actions>”

button “login” “<automation actions>”

button “open console” “<automation actions>”

set “hp bios configuration”

button “configure boot mode” “<automation actions for configure boot mode>”

## Hello world

to automate typing actions all you need to do is create a button with the text you wish to type, here’s the famous hello world example:

set “hello world set”

button “hello world” “hello world”

this will create the following interface:



## running automation actions

Clicking a button will run the automation action that was written for the button on the last used window the user was using, for the hello world example I was using the terminal when I clicked the hello world button and a result the automation action was performed on the terminal:



## Automating key presses

As previously mentioned to automate keypresses what you need to do is type whatever you want the tat to type for you inside the button statement:

Set “passwords”

button “my gmail password” “#hello123”

button “my facebook password” “#Facebook@123”

This would create a set that types different passwords for me instead of having to remember all passwords, remember the configuration file is not encrypted and can be viewed by other users with permission to the file so remember to set the file permissions correctly.

### Special characters

While typing you will need to use special characters like the up, down, page up and tab buttons, the tat scripting language provides an easy way to use those characters and this is via escape character(like the linux bash syntax)

For example to create a button that types in the username, presses tab to move to the password field, types in the password and presses ‘enter’ to login(this is an automated login example) the syntax would be as follows:

Button “facebook login” “mymail@gmail.com\t myfacebookpassword\n”

This button will type [mymail@gmail.com](mailto:mymail@gmail.com) press tab type myfacebookpassword and press enter, later in this documentation we will demonstrate how to automate opening the browser, navigating to facebook.com clicking the username field and logging in, all entirely automated with the tat.

Until then, here is a list of special characters and their syntax, do not forget to leave a space after any special characters(like the above example):

\b backspace

\e end

\u up

\d down

\l left

\r right

\l

\n enter

\pu page up

\pd page down

\r

\s sleep(covered later)

\t tab

\u

\\ \

\f1 f1

\f2 f2

\f3 f3

\f4 f4

\f5 f5

\f6 f6

\f7 f7

\f8 f8

\f9 f9

\c<character> presses ctrl+that character for example \ca is ctrl+a, \c\t is ctrl+tab

\a<character> presses alt+that character for example \a\f4 presses alt +f4

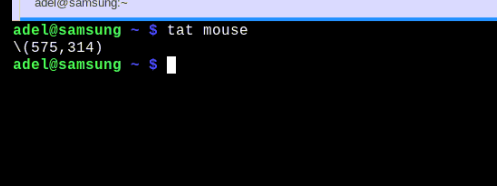
## Automating mouse actions

Automating mouse actions is a bit different, as you have to tell the application where to click or move the pointer; this is accomplished by using coordinates to tell tat where the mouse should click using the following syntax:

\(123,456)

There is always a space after the last bracket ’)’

There is no need to know anything about coordinates because the tat has a helper program that will tell you where the mouse is standing, from a terminal just type tat mouse and it will print out the current location of the mouse in a syntax that can be directly copied in the configuration file.



If you need to know the coordinates for an item inside firefox for example(or any other application for that matter) and want to know the coordinates of that item you can run the same helper program with an extra parameter:

Tat mouse 10

This will give you 10 seconds to switch to firefox and hover the mouse over the desired item, after 10 seconds had passed you can go back to the terminal and you will find the coordinates of your mouse immediately after the 10 seconds had passed, just copy the coordinates to the configuration file.

### Mouse syntax

\(x,y) click on the screen at the x & y coordinates

\(x;y) right click

\(x,y) move the mouse and do nothing

### example

set “mouse actions”

button “log on to facebook” “\(5,1080) \(5.1020) \(20,1020) \(157,490) [www.facebook.com\n](http://www.facebook.com\n) \(100,1230) myusername\t mypassword\n”

this button when clicked will click the start menu(on my pc, this will not work on other pcs as they might have different graphical layouts) firefox, go to the url bar and type [www.facebook.com](http://www.facebook.com) press enter, then click on the username text field type in the user name, press tab to go to the password field type in the password and press enter, fully automating opening the browser, navigating to facebook and logging in.

With some creativity anything can be automated in this manner such as filling in timesheets, typing and sending emails, social network activity(posting, liking, commenting….etc) and any other action that does not require user intervention.