
Automagica Documentation

Release 2

Oakwood Technologies BVBA

Aug 28, 2020

Contents:

1	Getting started	3
1.1	Installing on Windows	3
1.2	Installing on Linux	3
1.3	Your first automation	4
2	Automagica Flow	7
2.1	Introduction	7
2.2	Getting started	7
2.3	Inserting activities	7
2.4	Special nodes	8
2.5	Running and debugging	8
2.6	Using the integrated command line	9
2.7	Variable Explorer	9
2.8	Tips and tricks using the canvas	9
2.9	Examples	9
2.10	Deployment	12
3	Automagica Wand	13
3.1	Automagica Wand in Automagica Flow	13
3.2	Example	14
3.3	Anchoring	14
3.4	Tips and Tricks	14
4	Automagica Lab	15
4.1	Introduction	15
4.2	Getting started	15
4.3	Tips when starting out Automagica in an interactive development environment (IDE)	15
4.4	Browser Automation Example	17
5	Automagica Portal	25
5.1	Getting started	26
5.2	Introduction	26
5.3	Adding a process	26
6	Activities	27
6.1	Cryptography	27
6.2	Random	31

6.3	Output	41
6.4	Browser	41
6.5	Credential Management	47
6.6	FTP	49
6.7	Keyboard	51
6.8	Mouse	54
6.9	Image	59
6.10	Folder Operations	60
6.11	Delay	66
6.12	Word Application	66
6.13	Word File	70
6.14	Outlook Application	72
6.15	Excel Application	75
6.16	Excel File	85
6.17	PowerPoint Application	88
6.18	Office 365	92
6.19	Salesforce	93
6.20	E-mail (SMTP)	93
6.21	Windows OS	94
6.22	Terminal	102
6.23	SNMP	103
6.24	Active Directory	103
6.25	Utilities	104
6.26	System	106
6.27	PDF	112
6.28	System Monitoring	115
6.29	Image Processing	118
6.30	Process	121
6.31	Optical Character Recognition (OCR)	122
6.32	Alternative Frameworks	124
6.33	General	127
6.34	SAP GUI	127
6.35	Portal	127
6.36	Vision	128
7	Automagica and Docker	131
7.1	Introduction	131
7.2	Get started	131
8	Developers/contributors	133
8.1	Build installers	133
8.2	Internationalization	133
8.3	Want to join the development team?	134
9	Troubleshooting	135
9.1	Running unattended automations	135
9.2	Missing runtimes	137
10	Indices and tables	139
	Python Module Index	141
	Index	143

Automagica is an open source (robotic) process automation platform. With Automagica, automating cross-platform processes becomes a breeze.

1.1 Installing on Windows

Download Automagica and get started automating within 5 minutes through our website automagica.com. Our one-click installer for Windows in combination with Automagica Flow is by far the easiest way to get started automating.

By signing up for the [Automagica Portal](#), you also gain access to our specialized OCR-service and our Automagica Wand back-end, which features computer vision powered by machine learning for recognizing UI elements.

1.1.1 Install through PyPi

Installation can also be done through `pip`, Python's best-known package manager. We advise Python 3.7 for the installation, as for this version all the dependencies are widely available for all platforms. Newer versions are currently not supported, however this is on our roadmap.

After you have installed Python 3.7 and `pip`, you can install Automagica through PyPi by running:

```
pip install automagica -U
```

Please note that the PyPi releases are not synchronized with the installer available through the [Automagica Portal](#). It could happen that the PyPi version is slightly behind the version available through GitHub or the Automagica Portal.

1.2 Installing on Linux

1.2.1 Fedora-like distributions of Linux such as Red Hat Enterprise Linux or CentOS

You can install Automagica by running the following commands:

```
sudo yum install python3-devel chromium -y
sudo pip3 install automagica -U
```

1.2.2 Debian-like distributions of Linux such as Ubuntu

You can install Automagica by running the following commands:

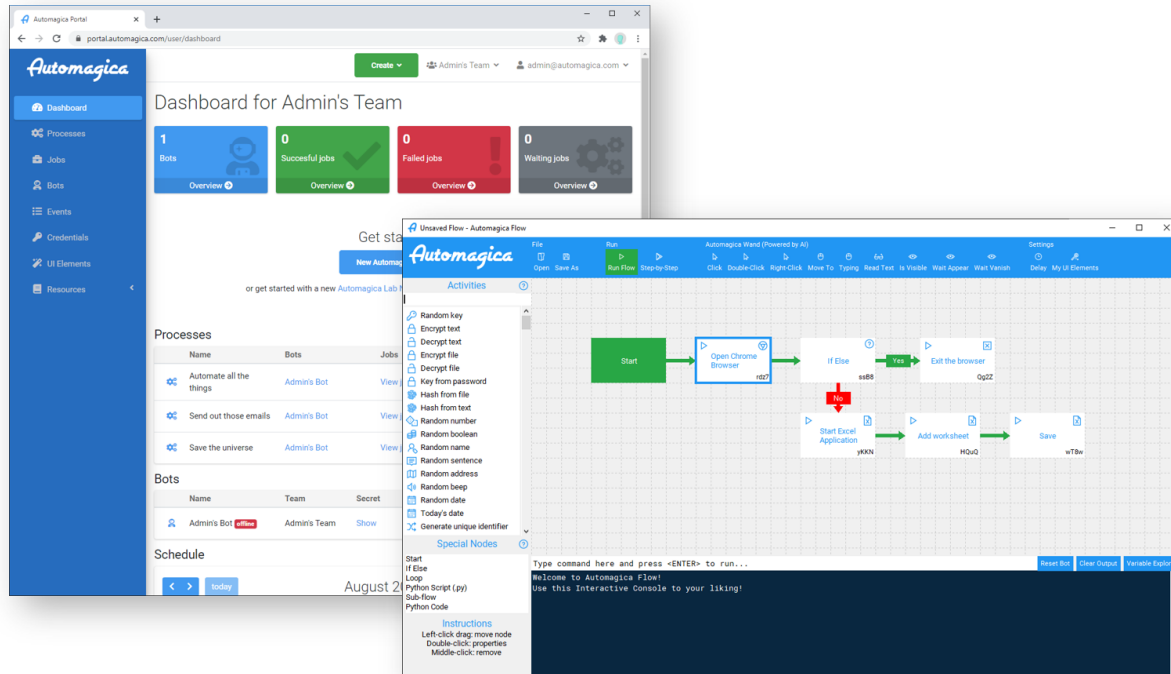
```
sudo apt-get install python3-devel chromium -y
sudo pip3 install automagica -U
## Installation on Linux
Interactive one-click examples with documentation can be found on [portal.automagica.
↪com] (https://portal.automagica.com)
```

1.3 Your first automation

There are multiple ways to automate tasks with Automagica. Down below a short overview of the different editors and what type of user it is designed for.

- **Automagica Flow:** a visual flow designer to build automations quickly with full support for Python code. Comes with Automagica Wand (UI element picker) integrated.
- **Automagica Lab:** notebook-style automation development environment based on Jupyter Notebooks.
- **Bring your own editor:** more developer-oriented Automagicians can use Automagica in their trusted Python editor like Visual Studio Code, Sublime, etc..

Unless you are set on using your own editor, we recommend Automagica Flow for both first-time users and more experienced developers. Automagica Flow comes with all the core components installed and offers the possibility to build your automation in both code and a visual canvas with dynamic search and dropdown-menus. In addition, for more seasoned Automagicians and Python developers, Automagica Flow also offers the flexibility to write or import custom Python code (.py) and Jupyter notebooks (.ipynb).



2.1 Introduction

Automagica Flow allows you to build software robots with a visual interface. Automagica Flow is the best and easiest way to get started building software automations.

2.2 Getting started

You can get started with Automagica Flow either by:

- double-clicking the shortcut on your desktop after installation (Windows One-Click installer)
- by running the `automagica flow new` command in the command line
- by starting Automagica Flow in the [portal](#) (Windows One-Click installer)
- if you have Automagica installed on this machine through our one-click installer, you can also start Automagica Flow by clicking [here](#)

2.3 Inserting activities

You can use the activity search to look for predefined activities, for example if you want to automate Excel you can look for all Excel-related activities by entering a keyword in the search. Double click the activity to insert it in the canvas. If you first select a node (blue outline) and then insert an activity, the next activity will automatically be connected to the one selected.

Options for a node can be opened by double-clicking on the node in the canvas. This will reveal the parameters to configure the node. Parameters are divided in two blocks: 'options' and 'node'.

- **Options:** These parameters are activity specific. Some activities require certain parameters to work and parameter will be indicated with (required). Others are optional and can range from changing the behavior to selecting paths for in- or output for example.
 - **Return variable:** Whenever this field is available for an activity this means there is some sort of output which can be stored in a variable. It is recommended to enter a variable name in this field as it allows you to use this value in consecutive nodes. These variables can be viewed at any time through the variable explorer.
- **Node:** Node options are available for most nodes.
 - **Unique ID:** Each node has a unique ID which can be seen either in the parameters or in the canvas. Can not be changed.
 - **Label:** Custom name for the node. It is recommended to give activities a more specific name, as it helps with debugging larger automations on multiple levels.
 - **Next Node:** Automated Flows follow the activities on the canvas, to change the order of occurrence you can use this option to select the next node.
 - **On Exception Node:** Whenever something goes wrong in an activity (e.g. Wand was not able to find an element, Excel could not be started, ..) there is the possibility to reroute to another node. If an Exception occurs without specifying an Exception node, the Flow will stop and raise an Exception. Exception details are visible in the console when running with Automagica Flow, or in the Logs when running through Automagica Portal.

2.4 Special nodes

Special nodes are designed to add flexibility to automations, something that is typically not possible with a one dimensional automation script. Down below an overview of the different Nodes:

- **Start:** Entry point for your automation, activity following this node will be the first action.
- **If Else:** Conditional Node to split a flow based on a condition. Conditions can include earlier set variables.
- **Python Script:** Python allows you to import a Python (.py) file in your workflow. This allows the user to fully integrate Python code for more complex parts of an automation (e.g. complex data transformation or usage of third-party libraries).
- **Sub Flow:** Allows to insert a Sub Flow. Can be used to split large flows into re-usable parts to keep things organized.
- **Python Code:** Python Code is similar to Python Script node, except that instead of importing a file it opens a small editor to write custom Python code. Variables are shared with Automagica Flow and allows sharing of variables.

2.5 Running and debugging

Running (parts of) your Flow can be done in one of the following methods:

- The green ‘Run Flow’ button in the top runs all steps in the Automation beginning at from the ‘Start’ node. A popup will show the progress and current step of the Flow
- ‘Run step-by-step’ also starts at the ‘Start’ node but pauses after each node, waiting for the user to specifically press ‘continue’ in a popup to advance to the next step. Can be used for debugging and development, variable explorer and command line can be used while paused.

- Clicking the play icon in the upper right corner of a node will only execute that specific node

2.6 Using the integrated command line

The integrated command line allows you to write and debug Python code in the same environment as where the activities are performed. Information is shown here and exceptions are raised similar to working with Python in a command line or developers tools like IPython or Jupyter Notebooks.

When running a node you can see the equivalent Python command with your parameters in the command line. You can also use the command line to explore or manipulate variables, run activities ad-hoc, import third party Python libraries or write Python code (similar to using a 'Python Script' or 'Python Code' - Node).

2.7 Variable Explorer

Activities that have a 'Return variable' can hold different types of variables (integer, float, string, ..). The variable explorer gives you an easy overview of the assigned variables at that specific moment. Resetting the bot will clear all variables.

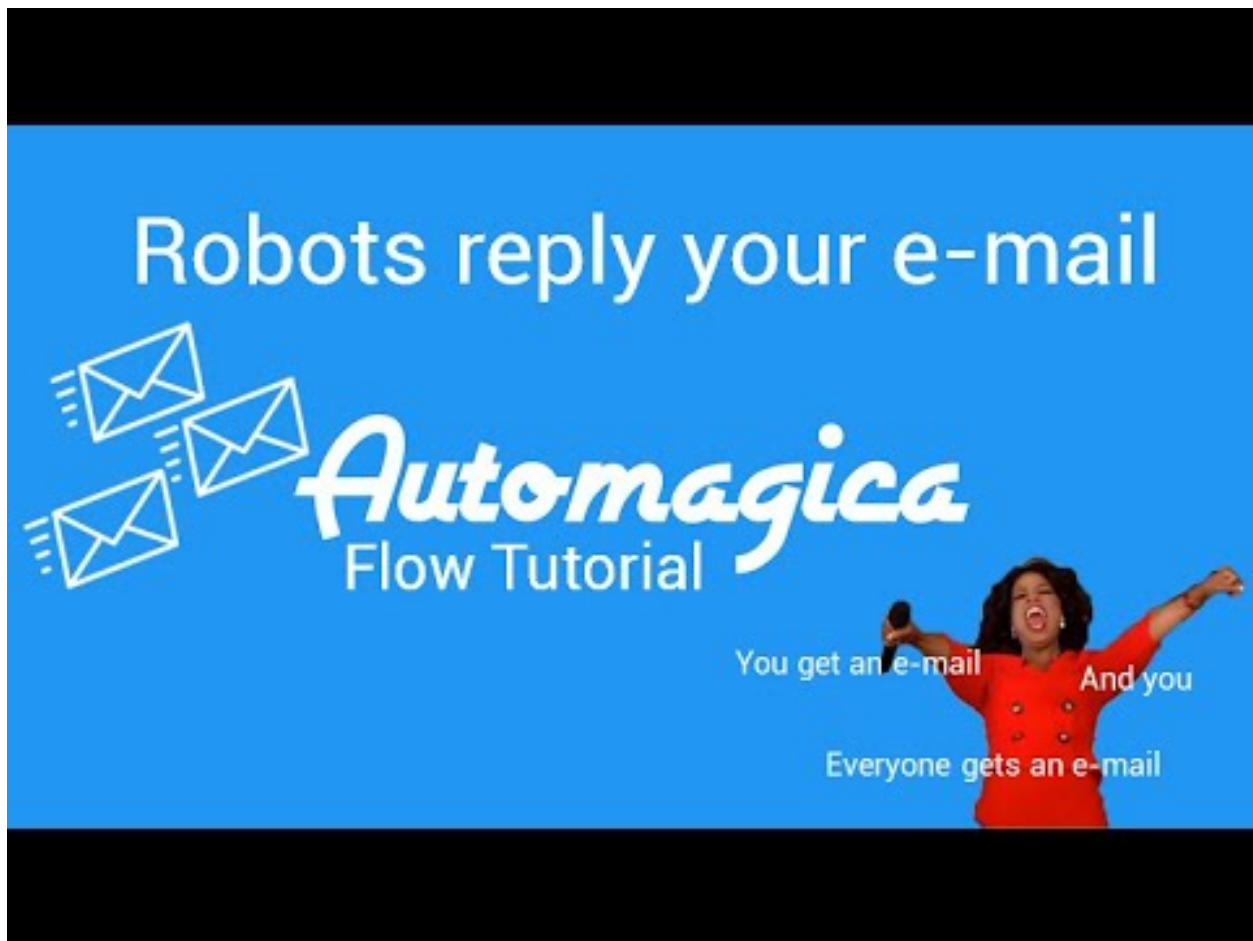
2.8 Tips and tricks using the canvas

- Once nodes are inserted you can move them around while holding middle mouse button
- Double clicking opens up the properties
- Middle mouse click deletes a node. Alternatively you can press delete after selecting a node to delete a node.
- Holding shift allows you to select, move/delete multiple nodes at once with the left mouse button.
- Keyboard shortcut 'ctrl + z' allows for undo
- Once a node is selected, newly inserted nodes will automatically be connected to the selected node

2.9 Examples

Be sure to check out the following videos on our YouTube channel for some hands-on examples and tutorials with Automagica Flow:







2.10 Deployment

Once your Automagica Flow is finished, you can upload it to the *Automagica Portal* to either start it by a schedule or by other triggers such as by sending an e-mail or by a REST API-call.

Automagica Wand

An easy way to interact with UI elements on the screen is to use Automagica Wand. For convenience, the main Wand activities are shown in the upper panel of Automagica Flow. Wand uses machine learning algorithms to look for elements on the screen that were recorded, which allows you to build robust automations using the GUI. This is different from comparing pixels as Wand mimics the way people use elements to navigate. For example, Just like humans, Wand can still find a button even if it is on a different place on the screen or if it changed slightly (e.g. screen resolution changed, button text is slightly different, color/theme variation, ..).

3.1 Automagica Wand in Automagica Flow

- by running the `automagica wand` command in the command line
- by starting Automagica Wand in the [UI elements section of the portal](#) (Windows One-Click installer)
- if you have Automagica installed on this machine through our one-click installer, you can start Automagica Wand activities in the top bar of [Automagica Flow](#)

Automagica Wand can be useful for different kinds of activities:

- Clicking: Move the mouse or (double) click elements with Automagica Wand
- Typing: Data entry in all kinds of editable fields, note that most of the time input fields are only distinguishable by using the anchoring feature
- Read Text: Use Wand in combination with OCR to read text directly from the screen. Note this activity requires you to use anchors to work properly
- Element existence: Use wand to validate the existence/absence of a certain element, e.g. continue when loading screen is over

3.2 Example

Let's say we want to click on the desktop recycle bin icon using Automagica Wand. In this example we will use Automagica Flow to record and run the example. As an illustration of the robustness we will empty the recycle bin and move the element on the screen after recording with Automagica Wand. Emptying the recycle bin causes the icon to slightly change in Windows, which would mean it would not be recognizable anymore with traditional pixel-comparison methods.

3.3 Anchoring

After selecting an element there is an option to add an anchor. An anchor is an unambiguous element on the screen that always has the same relative position to the element you want to reach. Take the example of a form with two identical input fields down below.

If we wanted to point to the field outlined in red and were to simply record this field as a target, it could be mistaken by the second identical input field. A solution would be to anchor the Login / Clear button (outlined in blue). The bot would then look for this anchor and move to the element based on the *relative position to this anchor*.

3.4 Tips and Tricks

Elements can be viewed within the Portal and are shared with team members if a bot is shared. This allows you to record, develop and run on different machines. In the Automagica Portal you can see counters successful detections and failed detections, quickly allowing you to debug automations and clean recorded elements that are not in use.

Each element has a unique ID, entering this ID in a suitable activity will automatically point the robot to this element.

Note: To use Automagica Wand you need an API key, which is automatically installed and configured if you use the Windows one-click-installer. Screen data gets sent to the Automagica servers where the ML algorithms analyse the screen. You can view and edit your elements and data in the Automagica Portal, which is connected with your API key. If you want to use Automagica Wand while keeping your promise within your network, we can help you set up a commercial [on-premise deployment](#).

4.1 Introduction

The Automagica Lab allows you to build software bots from an interactive development environment (IDE) based on Jupyter Notebooks. Those familiar with Jupyter Notebooks and Python will feel right at home with the Automagica Lab.

4.2 Getting started

After installing Automagica, you can launch the Automagica Lab with the following command:

```
automagica lab new
```

If you'd like to edit an existing Automagica Lab file (ending with `.ipynb`), run the following command:

```
automagica lab edit examples/automagica_lab_example.ipynb
```

4.3 Tips when starting out Automagica in an interactive development environment (IDE)

Instead using Automagica Lab, you can use your own editor or Notebook environment. Most activities come with a short example of how to use them under the [activities-section in this documentation](#).

Note: after installing Python and Automagica start your scripts by importing Automagica to get access to all functionalities.

```
from automagica import *
```

4.3.1 Variables

Most Automagica activities make use of parameters. Some are required and some are optional for more flexibility and advanced usage. The chapter ‘Activities’ contains a list of all the activities and their possible parameters.

In general a parameter can be a word (string), a number (integer) or a path.

String:

A string can be a word or a sentence, in Python a string starts and ends with apostrophes (‘’)

```
# Example of a string:
my_string = 'This can be everything'
```

Number:

An integer is a number:

```
# Example of an integer:
my_integer = 7
```

Path

A path specifies the directories in which a file, folder, executable,... is located. An example of such a pathname is: “C:/Users/Bob/Desktop/Automagica.pptx”.

```
# In a function:
open_file('C:/Users/Bob/Desktop/Automagica.pptx')
```

Alternatively you can double every backslash input. The next snippet of code illustrates how to use this method

```
# Pathname:
C:/Users/Bob/Desktop/Automagica.pptx

# As a string:
'C:/Users/Bob/Desktop/Automagica.pptx'

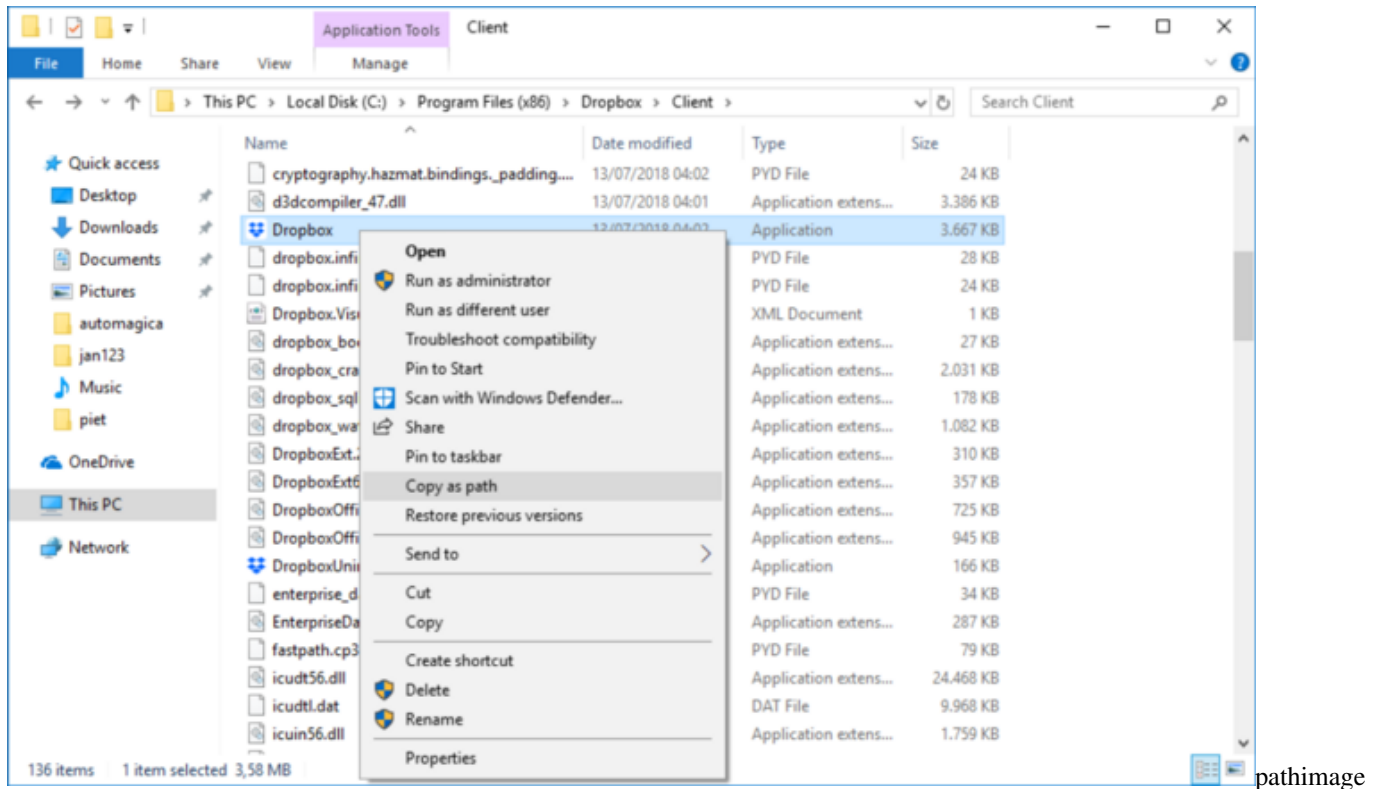
# In a function:
open_file('C:/Users/Bob/Desktop/Automagica.pptx')
```

In Windows Explorer, a path can be determined by pressing shift + right click on a file, folder,... A menu pops up, where you can select “copy as path” (see image). This copies the path as a string to the clipboard, e.g. “C:\Program Files (x86)\Dropbox\Client\Dropbox.exe”. Note that pasting this directly in your code will cause a SyntaxError, to avoid this you can either use forward slashes, add an ‘r’ added in front of the path the backslashes (to make a raw string) or double the backslashes (escaping) for it to be in correct form for a function input:

```
r"C:\Program Files (x86)\Dropbox\Client\Dropbox.exe"

"C:/Users/Bob/Desktop/Automagica.pptx"

C:\\Users\\Bob\\Desktop\\Automagica.pptx
```



4.4 Browser Automation Example

Out-of-the box Automagica uses Chrome as automated browser. Automating the browser requires you to find the elements to manipulate them. You could use Automagica to directly automate browser tasks within your Python scripts directly addressing the HTML elements. Another, more intuitive way would be to use Automagica Flow and Automagica Wand, which does not require knowledge of the inner HTML structure of the webpage.

The following sections will explain how to find, read and manipulate those web elements.

4.4.1 Basic functions

To open a browser choose 'Open Chrome browser' from menu or type the command:

```
browser = Chrome()
```

The browser function will wait until the page has fully loaded (that is, the "onload" event has fired) before continuing in the Automagica script. It's worth noting that if your page uses a lot of AJAX on load then the browser function may not know when it has completely loaded.

Browse to a website by clicking 'Browse to URL' in the menu or use the command:

```
browser.browse_to('https://mywebsite.com/')
```

Closing the browser can be done by:

```
browser.close()
```

To move backward and forward in your browser's history:

```
browser.forward()  
browser.back()
```

To click on an element:

```
element.click()
```

To enter text into a text field:

```
element.send_keys("some text")
```

To clear an element:

```
element.clear()
```

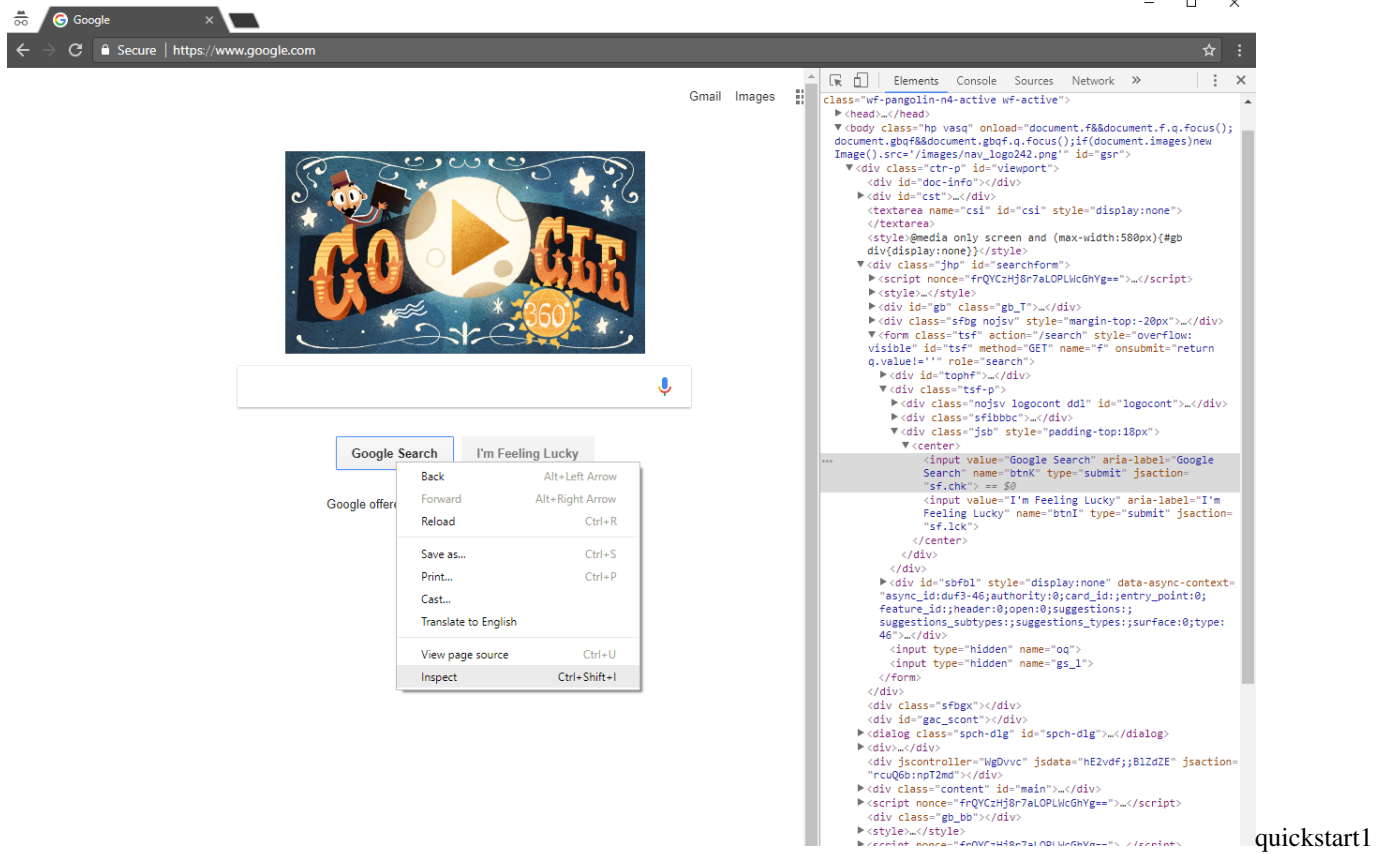
4.4.2 Navigating

To navigate and perform actions in the browser it is crucial to locate elements. Elements can be everything in the html files of a website like text, titles, buttons, text fields, tables, etc...

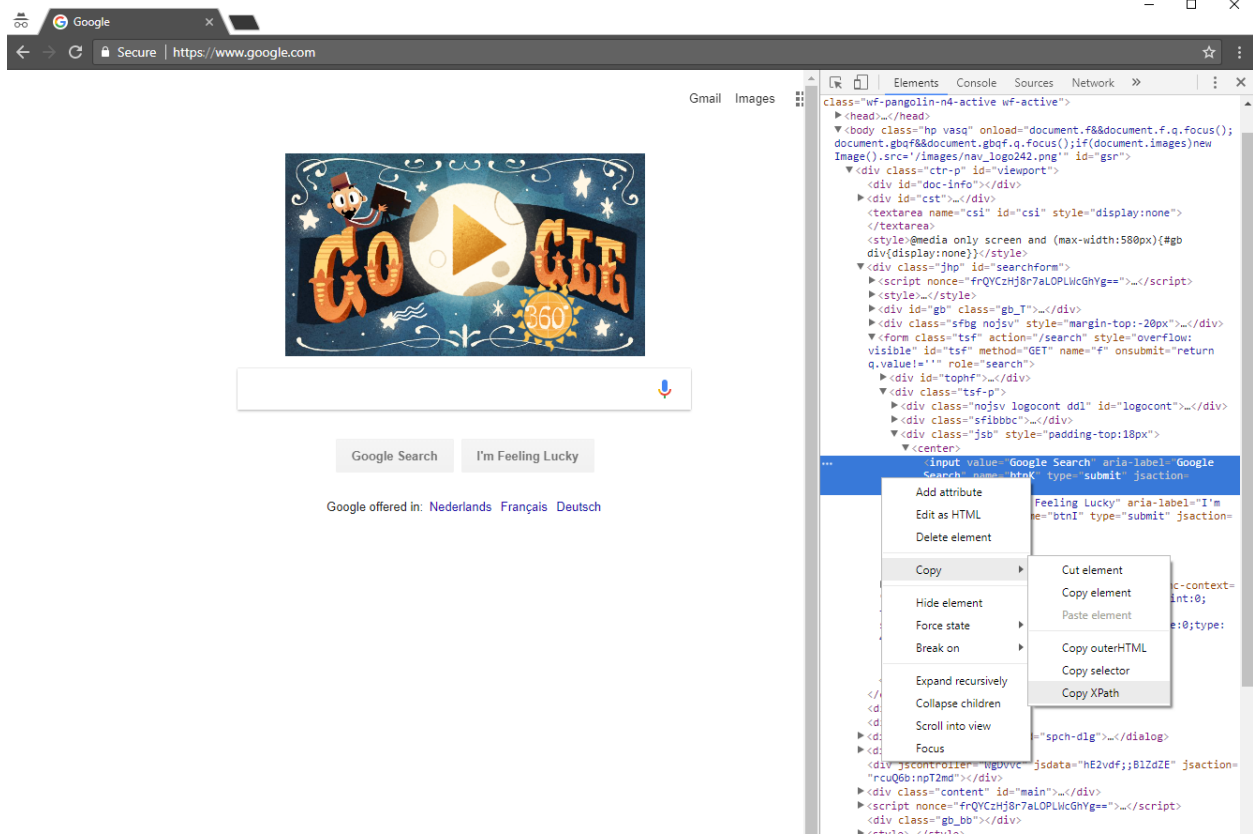
4.4.3 Quick start

There are two methods to finding elements, *find_element* to find a single element and *find_elements* to find multiple. Arguably the easiest way to find a certain element is by copying it's XPath.

To do this in Chrome right click on the element you want to find, in the example below this is the “Google Search” button on Google.com. Click *inspect element* and a side tab with the html code opens with the element you selected highlighted in blue.



In the html code, right click the highlighted block and select *Copy* -> *Copy XPath*.



You can now use the absolute XPath to manipulate the element. However this is a fast method for prototyping, we do not recommend using absolute paths in production environments. Slight changes in the html code would cause the absolute path to change and to likely cause errors. A more in-depth overview in the next section.

4.4.4 Selecting elements

Selection by name

Use this when you know name attribute of an element. With this strategy, the first element with the name attribute value matching the location will be returned. If no element has a matching name attribute, a NoSuchElementException will be raised.

For instance, consider these elements on a webpage:

The corresponding html code would be:

```
<html>
<body>
  <form id="loginForm">
    <input name="username" type="text" />
    <input name="password" type="password" />
    <input name="loginbutton" type="submit" value="Login" />
    <input name="clearbutton" type="button" value="Clear" />
  </form>
</body>
</html>
```


The username and password field can be found by:

```
username = browser.find_element_by_name('username')
password = browser.find_element_by_name('password')
```

To fill in the fields:

```
username.send_keys("Automagica_User1")
password.send_keys("thisismypassword123")
```

To find and click on the login button:

```
login = browser.find_element_by_name('loginbutton')
login.click()
```

Side note

In case of double naming, finding by name always finds the first element. Imagine the following html code:

```
<html>
<body>
  <form id="loginForm">
    <input name="username" type="text" />
    <input name="continue" type="submit" value="Login" />
    <input name="continue" type="button" value="Clear" />
  </form>
</body>
</html>
```

The following command will find the first element with the name “continue” and thus selecting the Login button:

```
continue = browser.find_element_by_name('continue')
```

Selection by Id

You can select elements by Id when this is known. This is a robust method, yet generally not every element has a known id tag. Consider the html code below:

```
<html>
<body>
  <form id="loginForm">
    <input name="username" type="text" />
    <input name="continue" type="submit" value="Login" />
  </form>
</body>
</html>
```

In this case the form has an id “loginForm”. Hence the form can be selected with the Id by:

```
loginform = browser.find_element_by_id('loginForm')
```

4.4.5 Selection by Xpath

XPath (XML Path Language) is a query language for selecting nodes from an XML document. Since HTML can be an implementation of XML (referred to as XHTML), this language can be used to find and manipulate target elements in web applications.

The advantage of using XPath is the possibility to reach every element within an HTML structure. See [Quick start](#) for a visual introduction on how to find and use an element with XPath. The disadvantage of using a full XPath is that it is not very robust. Even the slightest changes in a HTML page would cause absolute XPaths to change, which in result will likely cause your robot unable to find the correct elements. Note that this is different from using an element name or id, as elements will still be able to be found with changes in the HTML page as long as the name or id remains the same.

Therefore, when working with XPath the robustness can be increased by finding a nearby element with an id or name attribute (ideally a parent element), so you can locate your target element based on the relationship.

Consider the following structure on a HTML page:

With the following source code:

```
<html>
<body>
  <form id="loginForm">
    <input name="username" type="text" />
    <input name="password" type="password" />
    <input name="loginbutton" type="submit" value="Login" />
    <input name="clearbutton" type="button" value="Clear" />
  </form>
</body>
</html>
```

Selecting the username:

1. Absolute path (note that this would break if the HTML was changed only slightly)
2. Point to the first element in the form
3. First input element with attribute named 'name' and the value username

```
username = browser.find_element_by_xpath("//form[input/@name='username']")
username = browser.find_element_by_xpath("//form[@id='loginForm']/input[1]")
username = browser.find_element_by_xpath("//input[@name='username']")
```

The “clear” button can be located by:

1. Fourth input element of the form element with attribute named id and value loginForm
2. Input with attribute named name and the value continue and attribute named type and the value button

```
clear_button = browser.find_element_by_xpath("//form[@id='loginForm']/input[4]")
clear_button = browser.find_element_by_xpath("//input[@name='continue'][@type='button'
↪ '']")
```

The form can be selected by:

1. Absolute path (note that this would break if the HTML was changed only slightly):
2. First form element in the HTML
3. The form element with attribute named id and the value loginForm

```
login_form = browser.find_element_by_xpath("/html/body/form[1]")
login_form = browser.find_element_by_xpath("//form[1]")
login_form = browser.find_element_by_xpath("//form[@id='loginForm']")
```

4.4.6 Browsing Example

The following example browses to Google, searches for Automagica, opens the first Google Search result link

```
# Open Chrome
browser = Chrome()

# Browse to Google
browser.browse_to('https://google.com')

# Enter Search Text
browser.find_element_by_xpath('//*[@id="lst-ib"]').send_keys('automagica')

# Submit
browser.find_element_by_xpath('//*[@id="lst-ib"]').submit()

# Click the first link
browser.find_elements_by_class_name('r')[0].click()
```

A similar example with Automagica Flow can be seen in this short [YouTube video](#)

Automagica Portal

The [Automagica Portal](#) offers additional functionalities to manage your robot workforce.

A grasp of the Automagica Portal functionalities:

- Bot management: Connect and centrally manage robots
 - Securely connect bots to the Portal
 - Assign bots to processes and/or teams
- Role / Team management
 - Create a team and
- Process management: create processes that represent automated processes
 - Queueing: intelligent queueing to divide workload over one or more robots
 - Scheduling: schedule processes and view schedule
 - API integration: start processes with an API call
 - E-mail triggers: start processes with a unique e-mail address
- Event reporting: log actions and changes made by (team)members for a complete audit trail
- Version control: manage your scripts and keep track of your versions
 - Keep an overview of development / production versions
 - Historic overview of versions
 - Download different versions
- Reporting on successful/failed jobs
 - Overview of performed jobs both in calendar with color codes
 - Filter jobs based on date / outcome in overview
 - View logs / rerun jobs
 - Add e-mail / SMS / Teams / Slack / Telegram notifications on job status

- Credential management: keep your credentials in a central secure vault
- UI elements overview: view and edit elements recorded with Automagica Wand

5.1 Getting started

You can get started with the Automagica Portal at portal.automagica.com.

5.2 Introduction

After signing up in the Portal [Automagica Portal](#) with the one-click-installer if you don't have a bot installed yet.

Once installed, your robot will be automatically connected. If this is not the case you can manually set this up as described in [the bot section](bot.md).

5.3 Adding a process

You can add a process by using the 'create' button and selecting process. A process consists of several parts

- **Name:** arbitrary name of the process
- **Bots:** dropdown menu with all your available robots.
- **Add version:** select either a Python (.py), Jupyter Notebook (.ipynb) or Automagica Flow (.json) file to add it to this process.
 - **Entrypoint:** If you have created an Automagica Flow with additional Python .py or subflows (.json) you can add those files to the upload. Make sure to specify the main-file as entrypoint, this is the file your automation starts with
- **Add trigger:** once your file is uploaded you can add triggers to start the process
 - **Schedule:** specify date/time to run the process
 - **E-mail:** obtain a unique e-mail that triggers this process when an e-mail is send to it. Parameters can be put in the subject line
 - **API:** API call to trigger the process
- **Add notification:** once your file is uploaded you can add notifications when process fails or ends completes successfully

Once you have defined the process you will see a new section 'Jobs' in the menus. If a process is performed it is now called a 'job'. A job one specific instance of a process, possibly with custom parameters.

6.1 Cryptography

`automagica.activities.generate_random_key()`

Random key

Generate random Fernet key. Fernet guarantees that a message encrypted using it cannot be manipulated or read without the key. Fernet is an implementation of symmetric (also known as “secret key”) authenticated cryptography

Returns Random key

Return type

bytes

Example

```
>>> # Generate a random key
>>> generate_random_key()
b'AYv6ZPVgnrUtHDbGZqAopRyAo9r0_UKrA2Rm3K_NjIo='
```

Keywords random, key, fernet, hash, security, cryptography, password, secure

Icon las la-key

`automagica.activities.encrypt_text_with_key(text, key)`

Encrypt text

Encrypt text with (Fernet) key,

Parameters

- **text** (*string*) – Text to be encrypted
- **key** (*bytes*) – Fernet Encryption key

Returns Encrypted text

Return type

bytes

Example

```
>>> # Generate a random key
>>> key = generate_random_key()
>>> # Encrypt text with this key
>>> encrypt_text_with_key('Sample text', key)
b'gAAAAABd8lpG8fNqcj5eXrPPHlx4KeCm-1TgX3jkyhStMfIlGImIa-qaINZAJ8XcxPcG8iu84iT56b_
↪qAW9c5qpe7btUFhtxQ=='
```

Keywords random, encryption, secure, security, hash, password, fernet, text**Icon** las la-lock`automagica.activities.decrypt_text_with_key(encrypted_text, key)`

Decrypt text

Decrypt bytes-like object to string with (Fernet) key

Parameters

- **encrypted_text** – Text to be encrypted.
- **key** (*bytes*) – Fernet Encryption key

Returns Decrypt text**Return type**

string

Example

```
>>> # Generate a random key
>>> key = generate_random_key()
>>> # Encrypt text with generated key
>>> encrypted_text = encrypt_text_with_key('Sample text', key)
>>> # Decrypt text with same key
>>> decrypt_text_with_key(encrypted_text, key)
'Sample text'
```

Keywords decrypt, random, unlock, un-lock hash, security, cryptography, password, secure, hash, text**Icon** las la-lock-open`automagica.activities.encrypt_file_with_key(input_path, key, output_path=None)`

Encrypt file

Encrypt file with (Fernet) key. Note that file will be unusable unless unlocked with the same key.

Parameters

- **input_path** – Path to file to be encrypted
- **key** (*bytes*) – Fernet Encryption key
- **output_path** (*output_file, optional*) – Output path, defaults to the same directory with “_encrypted” added to the name

Returns Path to encrypted file

Return type

path

Example

```
>>> # Generate a random key
>>> key = generate_random_key()
>>> # Create a text file to illustrate file encryption
>>> text_file_path = make_text_file()
>>> # Encrypt the text file
>>> encrypt_file_with_key(text_file_path, key=key)
'C:\Users\

```

Keywords encrypt, random, password, secure, secure file, lock**Icon** las la-lockautomagica.activities.**decrypt_file_with_key**(input_path, key, output_path=None)

Decrypt file

Decrypts file with (Fernet) key

Parameters

- **input_file** (input_file) – Bytes-like file to be decrypted.
- **key** (bytes) – Path where key is stored.
- **output_file** – Outputfile, make sure to give this the same extension as basefile before encryption. Default is the same directory with “_decrypted” added to the name

Returns

Path to decrypted file

Example

```
>>> # Generate a random key
>>> key = generate_random_key()
>>> # Create a text file to encrypt file
>>> text_file_path = make_text_file()
>>> # Encrypt the text file
>>> encrypted_text_file = encrypt_file_with_key(text_file_path, key=key)
>>> # Decrypt the newly encrypted file
>>> decrypt_file_with_key(encrypted_text_file, key=key)
'C:\Users\

```

Keywords decrypt, random, password, secure, secure file, unlock**Icon** las la-lock-openautomagica.activities.**generate_key_from_password**(password, salt=None)

Key from password

Generate key based on password and salt. If both password and salt are known the key can be regenerated.

Parameters

- **password** (string) – Passwords

- **salt** (*string, optional*) – Salt to generate key in combination with password. Default value is the hostname. Take in to account that hostname is necessary to generate key, e.g. when files are encrypted with salt ‘A’ and password ‘B’, both elements are necessary to decrypt files.

Returns

Bytes-like object

Example

```
>>> # Generate a key from password
>>> key = generate_key_from_password(password='Sample password')
b'7jGGF5w_xyIOCIzGCMlLnNyUvFpNvIUy08JCHopgAmm8='
```

Keywords random, key, fernet, hash, security, cryptography, password, secure, salt

Icon las la-lock

automagica.activities.**generate_hash_from_file** (*input_path*, *method='md5'*,
buffer_size=65536)

Hash from file

Generate hash from file

Can be used to create unique identifier for file validation or comparison. Please note that MD5 and SHA1 are not cryptographically secure.

Parameters

- **input_path** (*input_file*) – File to hash
- **method** – Method for hashing, choose between ‘md5’, ‘sha256’ and ‘blake2b’. Note that different methods generate different hashes. Default method is ‘md5’.
- **buffer_size** (*int, optional*) – Buffer size for reading file in chunks, default value is 64kb

Options method [‘md5’, ‘sha256’, ‘blake2b’]

Returns

Bytes-like object

Example

```
>>> # Generate a text file to illustrate hash
>>> text_file_path = make_text_file()
>>> # Get hash from text file
>>> generate_hash_from_file(text_file_path)
'1ba249ca5931f3c85fe44d354c2f274d'
```

Keywords hash, md5, sha256, blake2b, identifier, unique, hashing, fingerprint, comparison

Icon las la-fingerprint

automagica.activities.**generate_hash_from_text** (*text*, *method='md5'*)

Hash from text

Generate hash from text. Keep in mind that MD5 is not cryptographically secure.

Parameters

- **text** (*string*) – Text to hash
- **method** – Method for hashing, choose between ‘md5’, ‘sha256’ and ‘blake2b’. Note that different methods generate different hashes. Default method is ‘md5’.

Options method [‘md5’, ‘sha256’, ‘blake2b’]

Example

```
>>> # Generate a hash from text
>>> generate_hash_from_text('Sample text')
'1ba249ca5931f3c85fe44d354c2f274d'
```

Keywords Hash, md5, sha256, blake2b, identifier, unique, hashing, fingerprint, text, comparison

Icon las la-fingerprint

6.2 Random

`automagica.activities.generate_random_number` (*lower_limit=0, upper_limit=100, fractional=False*)

Random number

Random numbers can be integers (not a fractional number) or a float (fractional number).

Parameters

- **lower_limit** (*int, optional*) – Lower limit for random number
- **upper_limit** (*int, optional*) – Upper limit for random number
- **fractional** (*bool, optional*) – Setting this to True will generate fractional number. Default value is False and only generates whole numbers.

Returns

Random integer or float

Example

```
>>> # Generate a random number
>>> generate_random_number()
7
```

Keywords random number, random integer, dice, gamble, rng, random

Icon las la-dice

`automagica.activities.generate_random_data` (*locale=None, type=None*)

Random data

Generates all kinds of random data. Specifying locale changes format for some options

Parameters

- **attribute** – Choose a specific characteristic or attribute from fake person
- **locale** – Add a locale to generates typical data for selected locale.

Options type ['email', 'food dish', 'food drink', 'fruit', 'vegetable', 'company type', 'ean code', 'imei code', 'isbn code', 'issn code', 'ip v4', 'ip v6', 'mac address', 'filename', 'filename extension', 'chemical element', 'academic degree', 'occupation', 'word', 'phone number', 'political view', 'university']

Options locale ['cs', 'da', 'de', 'de-at', 'de-ch', 'el', 'en', 'en-au', 'en-ca', 'en-gb', 'es', 'es-mx', 'et', 'fa', 'fi', 'fr', 'hu', 'is', 'it', 'ja', 'kk', 'ko', 'nl', 'nl-be', 'no', 'pl', 'pt', 'pt-br', 'ru', 'sk', 'sv', 'tr', 'uk', 'zh']

- cs - Czech
- da - Danish
- de - German
- de-at - Austrian german
- de-ch - Swiss german
- el - Greek
- en - English
- en-au - Australian English
- en-ca - Canadian English
- en-gb - British English
- es - Spanish
- es-mx - Mexican Spanish
- et - Estonian
- fa - Farsi
- fi - Finnish
- fr - French
- hu - Hungarian
- is - Icelandic
- it - Italian
- ja - Japanese
- kk - Kazakh
- ko - Korean
- nl - Dutch
- nl-be - Belgium Dutch
- no - Norwegian
- pl - Polish
- pt - Portuguese
- pt-br - Brazilian Portuguese
- ru - Russian
- sk - Slovak

- sv - Swedish
- tr - Turkish
- uk - Ukrainian
- zh - Chinese

Returns

Random data as string

Example

```
>>> # Generate random data
>>> generate_random_data()
'Banana'
```

Keywords random, lorem ipsum, gsm, cell, cellphone, telephone, mobile, number, smartphone, text generator, filler, place holder, noise, random text, random txt, text generation, fake, code, email, generate, generator, generic

Icon las la-digital-tachograph

`automagica.activities.generate_random_boolean()`

Random boolean

Generates a random boolean (True or False)

Returns

Boolean

Example

```
>>> # Generate a random boolean
>>> generate_random_boolean()
True
```

Keywords random, dice, gamble, rng, coin, coinflip, heads, tails

Icon las la-coins

`automagica.activities.generate_random_name(locale=None, name=None)`

Random name

Generates a random name. Adding a locale adds a more common name in the specified locale. Provides first name and last name.

Parameters

- **locale** – Add a locale to generate popular name for selected locale.
- **name** – Choose to generate first, last or full name.

Options name ['full', 'first', 'last']

Options locale ['cs', 'da', 'de', 'de-at', 'de-ch', 'el', 'en', 'en-au', 'en-ca', 'en-gb', 'es', 'es-mx', 'et', 'fa', 'fi', 'fr', 'hu', 'is', 'it', 'ja', 'kk', 'ko', 'nl', 'nl-be', 'no', 'pl', 'pt', 'pt-br', 'ru', 'sk', 'sv', 'tr', 'uk', 'zh']

- cs - Czech

- da - Danish
- de - German
- de-at - Austrian german
- de-ch - Swiss german
- el - Greek
- en - English
- en-au - Australian English
- en-ca - Canadian English
- en-gb - British English
- es - Spanish
- es-mx - Mexican Spanish
- et - Estonian
- fa - Farsi
- fi - Finnish
- fr - French
- hu - Hungarian
- is - Icelandic
- it - Italian
- ja - Japanese
- kk - Kazakh
- ko - Korean
- nl - Dutch
- nl-be - Belgium Dutch
- no - Norwegian
- pl - Polish
- pt - Portuguese
- pt-br - Brazilian Portuguese
- ru - Russian
- sk - Slovak
- sv - Swedish
- tr - Turkish
- uk - Ukrainian
- zh - Chinese

Returns

Random name as string

Example

```
>>> # Generate a random name
>>> generate_random_name()
'Michelle Murphy'
```

Keywords random, dummy name, name, name generator, fake person, fake, person, surname, lastname, fake name generator

Icon las la-user-tag

`automagica.activities.generate_random_words` (*locale=None, type=None*)
Random words

Generates a random sentence. Specifying locale changes language and content based on locale.

Parameters

- **type** – Specify type of words to generate
- **locale** – Add a locale to generate text for selected locale.

Options type ['sentence', 'quote', 'answer', 'single word', 'color', 'swear word']

Options locale ['cs', 'da', 'de', 'de-at', 'de-ch', 'el', 'en', 'en-au', 'en-ca', 'en-gb', 'es', 'es-mx', 'et', 'fa', 'fi', 'fr', 'hu', 'is', 'it', 'ja', 'kk', 'ko', 'nl', 'nl-be', 'no', 'pl', 'pt', 'pt-br', 'ru', 'sk', 'sv', 'tr', 'uk', 'zh']

- cs - Czech
- da - Danish
- de - German
- de-at - Austrian german
- de-ch - Swiss german
- el - Greek
- en - English
- en-au - Australian English
- en-ca - Canadian English
- en-gb - British English
- es - Spanish
- es-mx - Mexican Spanish
- et - Estonian
- fa - Farsi
- fi - Finnish
- fr - French
- hu - Hungarian
- is - Icelandic
- it - Italian
- ja - Japanese
- kk - Kazakh

- ko - Korean
- nl - Dutch
- nl-be - Belgium Dutch
- no - Norwegian
- pl - Polish
- pt - Portuguese
- pt-br - Brazilian Portuguese
- ru - Russian
- sk - Slovak
- sv - Swedish
- tr - Turkish
- uk - Ukrainian
- zh - Chinese

Returns

Random words as string

Example

```
>>> # Generate a random sentence
>>> generate_random_words()
'The age of automation is going to be the age of do-it-yourself'
```

Keywords random, sentence, lorem ipsum, text generater, filler, place holder, noise, random text, random txt, text generation, nlp

Icon las la-comment

`automagica.activities.generate_random_address(locale=None,format=None)`

Random address

Generates a random address. Specifying locale changes random locations and streetnames based on locale.

Parameters

- **format** – Choose a specific part or format for the address
- **locale** – Add a locale to generate typical address for selected locale.

Options format ['address', 'street', 'street number', 'city', 'continent', 'country', 'country code', 'postal code']

Options locale ['cs', 'da', 'de', 'de-at', 'de-ch', 'el', 'en', 'en-au', 'en-ca', 'en-gb', 'es', 'es-mx', 'et', 'fa', 'fi', 'fr', 'hu', 'is', 'it', 'ja', 'kk', 'ko', 'nl', 'nl-be', 'no', 'pl', 'pt', 'pt-br', 'ru', 'sk', 'sv', 'tr', 'uk', 'zh']

- cs - Czech
- da - Danish
- de - German

- de-at - Austrian german
- de-ch - Swiss german
- el - Greek
- en - English
- en-au - Australian English
- en-ca - Canadian English
- en-gb - British English
- es - Spanish
- es-mx - Mexican Spanish
- et - Estonian
- fa - Farsi
- fi - Finnish
- fr - French
- hu - Hungarian
- is - Icelandic
- it - Italian
- ja - Japanese
- kk - Kazakh
- ko - Korean
- nl - Dutch
- nl-be - Belgium Dutch
- no - Norwegian
- pl - Polish
- pt - Portuguese
- pt-br - Brazilian Portuguese
- ru - Russian
- sk - Slovak
- sv - Swedish
- tr - Turkish
- uk - Ukrainian
- zh - Chinese

Returns

Name as string

Example

```
>>> # Generate a random address
>>> generate_random_address()
'123 Robot Avenue'
```

Keywords random, address, data, street, city, postal, dummy name, name, name generator, fake person, fake, person, surname, lastname, fake name generator

Icon las la-map-marker

`automagica.activities.generate_random_beep(max_duration=2000, max_frequency=5000)`
Random beep

Generates a random beep, only works on Windows

Parameters

- **max_duration** (*int, optional*) – Maximum random duration in milliseconds. Default value is 2 milliseconds
- **max_frequency** (*int, optional*) – Maximum random frequency in Hz. Default value is 5000 Hz.

Returns

Sound

Example

```
>>> # Generate a random beep
>>> generate_random_beep()
```

Keywords beep, sound, random, noise, alert, notification

Icon las la-volume-up

`automagica.activities.generate_random_date(formatting='%m/%d/%Y %I:%M', days_in_past=1000)`
Random date

Generates a random date.

- %a Abbreviated weekday name.
- %A Full weekday name.
- %b Abbreviated month name.
- %B Full month name.
- %c Predefined date and time representation.
- %d Day of the month as a decimal number [01,31].
- %H Hour (24-hour clock) as a decimal number [00,23].
- %I Hour (12-hour clock) as a decimal number [01,12].
- %j Day of the year as a decimal number [001,366].
- %m Month as a decimal number [01,12].
- %M Minute as a decimal number [00,59].
- %p AM or PM.

- %S Second as a decimal number [00,61].
- %U Week number of the year (Sunday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Sunday are considered to be in week 0.
- %w Weekday as a decimal number [0(Sunday),6].
- %W Week number of the year (Monday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Monday are considered to be in week 0.
- %x Predefined date representation.
- %X Predefined time representation.
- %y Year without century as a decimal number [00,99].
- %Y Year with century as a decimal number.
- %Z Time zone name (no characters if no time zone exists).

Parameters

- **days_in_past** (*int*, *optional*) – Days in the past for which oldest random date is generated, default is 1000 days
- **formatting** (*string*, *optional*) – Formatting of the dates, replace with 'None' to get raw datetime format. e.g. format='Current month is %B' generates 'Current month is Januari' and format='%m/%d/%Y %I:%M' generates format 01/01/1900 00:00.

Returns

Random date as string

Example

```
>>> # Generate a random date
>>> generate_random_date()
01/01/2020 13:37'
```

Keywords random, date, datetime, random date, fake date , calendar

Icon las la-calendar

`automagica.activities.generate_date_today` (*formatting*='%m/%d/%Y')

Today's date

Generates today's date.

- %a Abbreviated weekday name.
- %A Full weekday name.
- %b Abbreviated month name.
- %B Full month name.
- %c Predefined date and time representation.
- %d Day of the month as a decimal number [01,31].
- %H Hour (24-hour clock) as a decimal number [00,23].
- %I Hour (12-hour clock) as a decimal number [01,12].
- %j Day of the year as a decimal number [001,366].

- %m Month as a decimal number [01,12].
- %M Minute as a decimal number [00,59].
- %p AM or PM.
- %S Second as a decimal number [00,61].
- %U Week number of the year (Sunday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Sunday are considered to be in week 0.
- %w Weekday as a decimal number [0(Sunday),6].
- %W Week number of the year (Monday as the first day of the week) as a decimal number [00,53]. All days in a new year preceding the first Monday are considered to be in week 0.
- %x Predefined date representation.
- %X Predefined time representation.
- %y Year without century as a decimal number [00,99].
- %Y Year with century as a decimal number.
- %Z Time zone name (no characters if no time zone exists).

Parameters **formatting** (*string*, *optional*) – Formatting of the dates, replace with ‘None’ to get raw datetime format. e.g. format=‘Current month is %B’ generates ‘Current month is Januari’ and format=‘%m/%d/%Y %I:%M’ generates format 01/01/1900 00:00.

Returns

Random date as string

Example

```
>>> # Generate a random date
>>> generate_date_today()
'01/01/2022'
```

Keywords random, date, today, now, today date, time, datetime, random date, fake date , calendar

Icon las la-calendar

automagica.activities.**generate_unique_identifier**()

Generate unique identifier

Generates a random UUID4 (universally unique identifier). While the probability that a UUID will be duplicated is not zero, it is close enough to zero to be negligible.

Returns

Identifier as string

Example

```
>>> # Generate unique identifier
>>> generate_unique_identifier()
'd72fd7ea-d682-4f78-8ca1-0ed34142a992'
```

Keywords unique, identifier, primary key, random

Icon las la-random

6.3 Output

`automagica.activities.display_osd_message` (*message*='Example message', *seconds*=5)

Display overlay message

Display custom OSD (on-screen display) message. Can be used to display a message for a limited amount of time. Can be used for illustration, debugging or as OSD.

Parameters

- **message** (*string*, *optional*) – Message to be displayed
- **seconds** – Duration in seconds for message to be displayed

```
>>> # Display overlay message
>>> display_osd_message()
```

Keywords message box, osd, overlay, info warning, info, popup, window, feedback, screen, login, attended

Icon las la-tv

`automagica.activities.print_console` (*data*='Example print')

Print message in console

Print message in console. Can be used to display data in the Automagica Flow console

Parameters **data** – Data to be printed

```
>>> # Print in console
>>> print_console()
```

Keywords print, box, osd, data, debugging info, popup, window, feedback, screen, login, attended

Icon las la-tv

6.4 Browser

class `automagica.activities.Chrome` (*load_images*=True, *headless*=False, *incognito*=False, *disable_extension*=False, *maximize_window*=True, *focus_window*=True, *auto_update_chromedriver*=False)

browse_to (*url*)

Browse to URL

Browse to URL.

Parameters **url** (*string*) – Url to browser to

Returns

Webpage

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.browse_to('https://nytimes.com')
```

Keywords chrome, element, browse to, browse, surf, surf to, go to, get, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon lab la-chrome

by_class (*element*)

Find class in browser

Find element with specified class on a webpage in the the browser. Can also use native 'find_element_by_class_name'

Parameters **element** (*string, optional*) – Class of element

Returns

Element by class

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find element by class
>>> element = browser.by_class('search-input')
>>> # We can now use this element, for example to click on
>>> element.click()
```

Keywords browser, class, classes, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

by_class_and_by_text (*element, text*)

Find element in browser based on class and text

Find all elements with specified class and text on a webpage in the the browser.

Parameters **element** (*string, optional*) – Class of element

Returns

Element by class and text

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find elements by class and text
>>> element = browser.by_class_and_by_text('search-input', 'Search Wikipedia')
>>> # We can now use this element, for example to click on
>>> element.click()
```

Keywords browser, class, text, name classes, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

by_classes (*element*)

Find class in browser

Find all elements with specified class on a webpage in the the browser. Can also use native 'find_elements_by_class_name' function

Parameters **element** (*string, optional*) – Class of element

Returns

Element by classes

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find elements by class
>>> elements = browser.by_classes('search-input')
```

Keywords browser, class, classes, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

by_id (*element*)

Find id in browser

Find element with specified id on a webpage in the the browser. Can also use native 'find_element_by_id' function

Parameters **element** (*string, optional*) – Id of element

Returns

Element by id

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find element by class
>>> elements = browser.by_id('search-input')
>>> # We can now use this element, for example to click on
>>> element.click()
```

Keywords browser, class, classes, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

by_xpath (*element*)

Find XPath in browser

Find all element with specified xpath on a webpage in the the browser. Can also use native 'find_elements_by_xpath'

Parameters **element** (*string, optional*) – Xpath of element

Returns

Element by xpath

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find element by xpath
>>> element = browser.by_xpath('//*[@id="js-link-box-en"]')
>>> # We can now use this element, for example to click on
>>> element.click()
```

Keywords random, xpath, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

by_xpaths (*element*)

Find all XPathS

Find all elements with specified xpath on a webpage in the the browser. Can also use native 'find_elements_by_xpath'

Parameters **element** (*string, optional*) – Xpath of element

Returns

Element by xpathS

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find elements by xpathS
>>> browser.by_xpaths('//*[@id="js-link-box-en"]')
[webelement1, webelement2 , .. ]
```

Keywords random, element, xpath, xml, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

exit ()

Exit the browser

Quit the browser by exiting gracefully. One can also use the native 'quit' function

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://automagica.com')
>>> # Close browser
>>> browser.exit()
```


Keywords quit, exit, close, element, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-window-close

find_all_links (*contains=""*)

Find all links

Find all links on a webpage in the browser

Parameters **contains** (*string*, *optional*) – Criteria of substring that url must contain to be included

Returns

Links

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://nytimes.com')
>>> # Find elements by text
>>> browser.find_all_links()
[webelement1, webelement2 , .. ]
```

Keywords random, element, link, links element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-window-restore

find_elements_by_text (*text*)

Find elements by text

Find all elements by their text. Text does not need to match exactly, part of text is enough.

Parameters **text** (*string*) – Text to find elements by

Returns

Elements that matched with text

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://nytimes.com')
>>> # Find elements by text
>>> browser.find_elements_by_text('world')
[webelement1, webelement2 , .. ]
```

Keywords element, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-align-center

find_first_link (*contains=None*)

Find first link on a webpage

Find first link on a webpage

Parameters **contains** (*string*, *optional*) – Criteria of substring that url must contain to be included

Returns

First link

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://nytimes.com')
>>> # Find elements by text
>>> browser.find_first_link()
```

Keywords random, link, links, element, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-window-restore

get_text_on_webpage()

Get all text on webpage

Get all the raw body text from current webpage

Returns

Text

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://nytimes.com')
>>> # Get text from page
>>> browser.get_text_on_webpage()
```

Keywords random, link, links, element, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-window-restore

highlight (*element*)

Highlight element

Highlight elements in yellow in the browser

Parameters **element** – Element to highlight

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://wikipedia.org')
>>> # Find first link on page
>>> first_link = browser.find_elements_by_xpath("//a[@href]")[0]
>>> # Highlight first link
>>> browser.highlight(first_link)
```

Keywords element, element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-highlighter

save_all_images (*output_path=None*)

Save all images

Save all images on current page in the Browser

Parameters **output_path** (*output_dir, optional*) – Path where images can be saved. Default value is home directory.

Returns

List with paths to images

Example

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://www.nytimes.com/')
>>> # Save all images
>>> browser.save_all_images()
>>> browser.quit()
['C:\Users\<username>\image1.png', 'C:\Users\<username>\image2.jpg',
↪ 'C:\Users\<username>\image4.gif']
```

Keywords image scraping, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-images

switch_to_iframe (*name='iframe'*)

Switch to iframe in browser

Switch to an iframe in the browser

Parameters **name** – Name of the Iframe

```
>>> # Open the browser
>>> browser = Chrome()
>>> # Go to a website
>>> browser.get('https://www.w3schools.com/html/html_iframe.asp')
>>> # Switch to iframe
>>> browser.switch_to_iframe()
```

Keywords browser, class, classes, element, xml element by text, chrome, internet, browsing, browser, surfing, web, webscraping, www, selenium, crawling, webtesting, mozilla, firefox, internet explorer

Icon las la-times

6.5 Credential Management

`automagica.activities.set_credential` (*username=None, password=None, system='Automagica'*)

Set credential

Add a credential which stores credentials locally and securely. All parameters should be Unicode text.

Parameters

- **username** (*string, optional*) – Username for which credential will be added.
- **password** (*string, optional*) – Password to add
- **system** (*string*) – Name of the system for which credentials are stored. Extra safety measure and method for keeping passwords for similar usernames on different applications a part. Highly recommended to change default value.

Returns

Stores credentials locally

Example

```
>>> set_credential('SampleUsername', 'SamplePassword')
```

Keywords credential, login, password, username, store, vault, secure, credentials, store, log in, encrypt

Icon las la-key

```
automagica.activities.delete_credential(username=None, password=None, system='Automagica')
```

Delete credential

Delete a locally stored credential. All parameters should be Unicode text.

Parameters

- **username** (*string*) – Username for which credential (username + password) will be deleted.
- **password** (*string, optional*) – Password to delete
- **system** – Name of the system for which password will be deleted.

Example

```
>>> set_credential('SampleUsername', 'SamplePassword')
>>> delete_credential('SampleUsername', 'SamplePassword')
```

Keywords credential, delete, login, password, username, store, vault, secure, credentials, store, log in, encrypt

Icon las la-key

```
automagica.activities.get_credential(username=None, system='Automagica')
```

Get credential

Get a locally stored redential. All parameters should be Unicode text.

Parameters

- **username** (*string*) – Username to get password for.
- **system** (*string, optional*) – Name of the system for which credentials are retrieved.

Returns

Stored credential as string

Example

```
>>> set_credential('SampleUsername', 'SamplePassword')
>>> get_credential('SampleUsername')
'SamplePassword'
```

Keywords credential, get, delete, login, password, username, store, vault, secure, credentials, store, log in, encrypt

Icon las la-key

6.6 FTP

class automagica.activities.**FTP**(server, username, password)

create_directory(directory_name, path='/')

Create FTP directory

Create a FTP directory. Note that sufficient permissions are present

Parameters

- **directory_name** (*string*) – Name of the new directory, should be a string e.g. 'my_directory'
- **path** (*output_dir, optional*) – Path to parent directory where to make new directory. Default is main directory

Returns Boolean if creation was succesful (True) or failed (False) :Example:

```
>>> # This example uses the Rebex FPT test server.
>>> # Trying to create a directory will most likely fail due to permission
>>> ftp = FTP('test.rebex.net', 'demo', 'password')
>>> # Create directory
>>> ftp.create_directory('brand_new_directory')
False
```

Keywords FTP, create, create folder, new, new folder, fptfile transfer protocol, filezilla, winscp, server, remote, folder, folders

Icon las la-folder-plus

directory_exists(path='/')

Check FTP directory

Check if FTP directory exists

Parameters **path** (*input_dir, optional*) – Path to check on existence. Default is main directory

Returns

Boolean

Example

```
>>> # This example uses the Rebex FPT test server.
>>> # Take caution uploading and downloading from this server as it is public
>>> ftp = FTP('test.rebex.net', 'demo', 'password')
>>> # Check if 'pub' folder exists in main directory
>>> ftp.directory_exists('\pub')
True
```

Keywords FTP, list, upload, fptfile transfer protocol, filezilla, winscp, server, remote, folder, folders

Icon las la-list-ol

download_file (*input_path*, *output_path=None*)

Download file

Downloads a file from FTP server. Connection needs to be established first.

Parameters

- **input_path** (*input_file*) – Path to the file on the FPT server to download
- **output_path** (*output_dir*, *optional*) – Destination path for downloaded files. Default is the same directory with “_downloaded” added to the name

Returns

Path to output file as string

Example

```
>>> # This example uses the Rebex FPT test server.
>>> # Take caution uploading and downloading from this server as it is public
>>> ftp = FTP('test.rebex.net', 'demo', 'password')
>>> # Download Rebex public file 'readme.txt'
>>> ftp.download_file('readme.txt')
'C:\Users\<username>\readme_downloaded.txt'
```

Keywords FTP, file transfer protocol, download, filezilla, winscp, server, remote, folder, folders

Icon las la-download

enumerate_files (*path='/'*)

List FTP files

Generate a list of all the files in the FTP directory

Parameters **path** (*input_dir*, *optional*) – Path to list files from. Default is the main directory

Returns

Prints list of all files and directories

Example


```
>>> # This example uses the Rebex FPT test server.
>>> # Take caution uploading and downloading from this server as it is public
>>> ftp = FTP('test.rebex.net', 'demo', 'password')
>>> # Show all files in main directory
>>> ftp.enumerate_files()
10-27-15  03:46PM      <DIR>          pub
```

(continues on next page)

(continued from previous page)

```
04-08-14 03:09PM 403 readme.txt
'226 Transfer complete.'
```

Keywords FTP, list, upload, fptfile transfer protocol, filezilla, winscp, server, remote, folder, folders

Icon  las la-list-ol

upload_file (*input_path*, *output_path=None*)

Upload file

Upload file to FTP server

Parameters

- **input_path** (*input_file*) – Path file that will be uploaded
- **output_path** (*output_dir*, *optional*) – Destination path to upload.


Returns

Path to uploaded file as string

Example

```
>>> # This example uses the Rebex FPT test server.
>>> # Take caution uploading and downloading from this server as it is public
>>> ftp = FTP('test.rebex.net', 'demo', 'password')
>>> # Create a .txt file for illustration
>>> text_file = make_text_file()
>>> # Upload file to FTP test server
>>> # Not that this might result in a persmission error for public FPT's
>>> ftp.upload_file(input_path = text_file)
```

Keywords FTP, upload, fptfile transfer protocol, filezilla, winscp, server, remote, folder, folders

Icon  las la-upload

6.7 Keyboard

`automagica.activities.press_key` (*key=None*, *delay=1*, *perform_n_times=1*, *delay_between=0.5*)

Press key

Press and release an entered key. Make sure your keyboard is on US layout (standard QWERTY). If you are using this on Mac Os you might need to grant acces to your terminal application. (Security Preferences > Security & Privacy > Privacy > Accessibility)

Parameters

- **key** – Key to press. This can also be a scan code (e.g: 33 for '!')
- **delay** (*int*, *optional*) – Delay before key is pressed in seconds, default is 1 second
- **perform_n_times** (*int*, *optional*) – How many times to perform the key press
- **delay_between** (*float*, *optional*) – Delay between key presses

Options key [' ', '!', '"', '#', '\$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}',

```

~, 'alt', 'backspace', 'end', 'ctrl', 'del', 'down', 'right', 'left', 'up', 'enter', 'escape', 'f1', 'f2',
'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert',
'pagedown', 'pageup', 'help', 'space', 'tab', 'shift', 'win']

```

Returns

Keypress

Example

```

>>> # Open notepad to illustrate typing
>>> run('notepad.exe')
>>> # Press some keys
>>> press_key('a')
>>> press_key('enter')
>>> press_key('b')
>>> press_key('enter')
>>> press_key('c')

```

Keywords keyboard, typing, type, key, keystroke, hotkey, press, press key

Icon las la-keyboard

`automagica.activities.press_key_combination` (*first_key, second_key, third_key=None, compatibility=False, delay=1*)

Press key combination

Press a combination of two or three keys simultaneously. Make sure your keyboard is on US layout (standard QWERTY).

Parameters

- **first_key** – First key to press
- **second_key** – Second key to press
- **third_key** – Third key to press, this is optional.
- **compatibility** – Set parameter to true to not use win32com. This could help with compatibility on certain systems or when certain keypresses do not work correctly.
- **key** (*int, optional*) – Delay before keys are pressed in seconds, default is 1 second

Options first_key [' ', '!', '"', '#', '\$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'alt', 'backspace', 'end', 'ctrl', 'del', 'down', 'right', 'left', 'up', 'enter', 'escape', 'f1', 'f2', 'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert', 'pagedown', 'pageup', 'help', 'space', 'tab', 'shift', 'win']

Options second_key [' ', '!', '"', '#', '\$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'alt', 'backspace', 'end', 'ctrl', 'del', 'down', 'right', 'left', 'up', 'enter', 'escape', 'f1', 'f2', 'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert', 'pagedown', 'pageup', 'help', 'space', 'tab', 'shift', 'win']

Options third_key [' ', '!', '"', '#', '\$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'alt', 'backspace', 'end', 'ctrl', 'del', 'down', 'right', 'left', 'up', 'enter', 'escape', 'f1', 'f2', 'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert', 'pagedown', 'pageup', 'help', 'space', 'tab', 'shift', 'win']

'f2', 'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert', 'pagedown', 'pageup', 'help', 'space', 'tab', 'shift', 'win']

Returns

Key combination

Example

```
>>> # Open notepad to illustrate typing
>>> run('notepad.exe')
>>> # Press 'ctrl + s' to prompt save window
>>> press_key_combination('ctrl', 's')
```

Keywords keyboard, key combination, shortcut, typing, type, key, keystroke, hotkey, press, press key

Icon las la-keyboard

`automagica.activities.typing`(*text*, *automagica_id=None*, *clear=False*, *interval_seconds=0.01*, *delay=1*)

Type text

Simulate keystrokes. If an element ID is specified, text will be typed in a specific field or element based on the element ID (vision) by the recorder.

Supported keys: ' ', '!', '"', '#', '\$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'alt', 'backspace', 'ctrl', 'delete', 'downarrow', 'rightarrow', 'leftarrow', 'uparrow', 'enter', 'escape', 'f1', 'f2', 'f3', 'f4', 'f5', 'f6', 'f7', 'f8', 'f9', 'f10', 'f11', 'f12', 'f13', 'f14', 'f15', 'f16', 'home', 'insert', 'pagedown', 'pageup', 'help', 'printscreen', 'space', 'scrollock', 'tab', 'shift', 'win'

parameter text Text in string format to type. Note that you can only press single character keys. Special keys can not be part of the text argument.

type text string

parameter automagica_id ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the `recorder()` function.

type automagica_id automagica_id, optional

parameter clear Attempts to clear the element before typing using hotkeys. Be cautious when using this method as a vision mismatch could result in deleting unwanted data. Default value is False

type clear bool, optional

parameter interval_seconds Time in seconds between two keystrokes. Default value is 0.01 seconds.

type interval_seconds int, optional

parameter delay Delay before beginning to type, default is 1 second

type delay int, optional

return Keystrokes

Example

```
>>> # Open notepad to illustrate typing
>>> run('notepad.exe')
>>> # Type a story
>>> typing('Why was the robot mad?')
```

They kept pushing his buttons!')

Keywords keyboard, keystrokes, key combination, shortcut, typing, type, key, keystroke, hotkey, press, press key, send keys, keystrokes

Icon las la-keyboard

6.8 Mouse

`automagica.activities.get_mouse_position(delay=None, to_clipboard=False)`

Get mouse coordinates

Get the x and y pixel coordinates of current mouse position. These coordinates represent the absolute pixel position of the mouse on the computer screen. The x-coördinate starts on the left side and increases going right. The y-coördinate increases going down.

Parameters

- **delay** (*int*, *optional*) – Delay in seconds before capturing mouse position.
- **to_clipboard** (*bool*, *optional*) – Put the coordinates in the clipboard e.g. 'x=1, y=1'

Returns

Tuple with (x, y) coordinates

Example

```
>>> get_mouse_position()
(314, 271)
```

Keywords mouse, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-mouse

`automagica.activities.display_mouse_position(duration=10)`

Display mouse position

Displays mouse position in an overlay. Refreshes every two seconds. Can be used to find mouse position of element on the screen. These coordinates represent the absolute pixel position of the mouse on the computer screen. The x-coördinate starts on the left side and increases going right. The y-coördinate increases going down.

Parameters **duration** (*int*, *optional*) – Duration to show overlay.

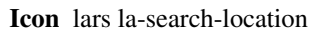
Returns

Overlay with (x, y) coordinates

Example

```
>>> display_mouse_position()
```

Keywords mouse, osd, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon 

`automagica.activities.click(automagica_id, delay=1)`

Mouse click

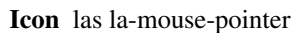
Clicks on an element based on the element ID (vision)

Parameters

- **automagica_id** (*automagica_id*) – ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the `recorder()` function.
- **delay** – Delay before clicking in seconds.

```
>>> # Click on a vision element, use the recorder() function to define elements
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
    ↪ also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
    ↪ found, use the recorder
>>> click('qf41')
```

Keywords mouse, vision, mouse, osd, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon 

`automagica.activities.click_coordinates(x=None, y=None, delay=1)`

Mouse click coordinates

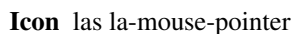
Clicks on an element based on pixel position determined by x and y coordinates. To find coordinates one could use `display_mouse_position()`.

Parameters

- **x** (*int*) – X-coordinate
- **y** (*int*) – Y-coordinate
- **delay** – Delay before clicking in seconds.

```
>>> # Click on pixel position
>>> click_coordinates(x=100, y=100)
```

Keywords mouse, vision, mouse, osd, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon 

`automagica.activities.double_click_coordinates(x=None, y=None, delay=1)`

Double mouse click coordinates

Double clicks on a pixel position determined by x and y coordinates.

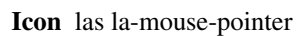
Parameters

- **x** (*int*) – X-coordinate

- **y**(int) – Y-coordinate
- **delay** – Delay before clicking in seconds.

```
>>> # Click on coordinates
>>> double_click_coordinates(x=100, y=100)
```

Keywords mouse, osd, overlay, double, double click, doubleclick show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon 

`automagica.activities.double_click(automagica_id=None, delay=1)`

Double mouse click

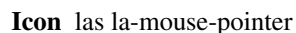
Double clicks on an element based on the element ID (vision)

Parameters

- **automagica_id**(*automagica_id*) – ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the `recorder()` function.
- **delay** – Delay before clicking in seconds.

```
>>> # Click on a vision element, use the recorder() function to define elements
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↳also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↳found, use the recorder
>>> double_click('qf41')
```

Keywords mouse, osd, overlay, double, double click, doubleclick show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon 

`automagica.activities.right_click(automagica_id=None, delay=1)`

Right click

Right clicks on an element based on the element ID (vision)

Parameters

- **automagica_id**(*automagica_id*) – ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the `recorder()` function.
- **delay** – Delay before clicking in seconds.

```
>>> # Click on a vision element, use the recorder() function to define elements
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↳also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↳found, use the recorder
>>> right_click('qf41')
```

Keywords mouse, osd, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-mouse-pointer

`automagica.activities.right_click_coordinates` (*x=None, y=None, delay=1*)

Right click coordinates

Right clicks on an element based pixel position determined by x and y coordinates.

Parameters

- **x** (*int*) – X-coördinate
- **y** (*int*) – Y-coördinate
- **delay** – Delay before clicking in seconds

```
>>> # Right click on coordinates
>>> right_click_coordinates(x=100, y=100)
```

Keywords mouse, osd, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-mouse-pointer

`automagica.activities.move_mouse_to` (*automagica_id=None, delay=1*)

Move mouse

Moves te pointer to an element based on the element ID (vision)

Parameters

- **automagica_id** (*automagica_id, optional*) – ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the recorder() function.
- **delay** (*int, optional*) – Delay before movement in seconds

Returns

Move mouse to (x, y) coordinates

Example

```
>>> # Use recorder to find an element ID
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↪also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↪found, use the recorder
>>> move_mouse_to('qf41')
```

Keywords mouse, osd, move mouse, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-arrows-alt

`automagica.activities.move_mouse_to_coordinates` (*x=None, y=None, delay=1*)

Move mouse coordinates

Moves te pointer to an element based on the pixel position determined by x and y coordinates

Parameters

- **x** (*int*) – X-coördinate
- **y** (*int*) – Y-coördinate
- **delay** (*int*, *optional*) – Delay between movements in seconds, standard value is 1s.

Returns

Move mouse to (x, y) coordinates

Example

```
>>> # Move mouse to coordinates
>>> move_mouse_to_coordinates(x=100, y=100)
```

Keywords mouse, osd, move mouse, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-arrows-alt

automagica.activities.**move_mouse_relative** (*x=None, y=None*)

Move mouse relative

Moves the mouse an x- and y- distance relative to its current pixel position.

Parameters

- **x** (*int*) – X-coördinate
- **y** (*int*) – Y-coördinate

Returns

Move mouse (x, y) coordinates

Example

```
>>> move_mouse_to_coordinates(x=100, y=100)
>>> wait(1)
>>> move_mouse_relative(x=10, y=10)
```

Keywords mouse, osd, move mouse, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-arrows-alt

automagica.activities.**drag_mouse_to_coordinates** (*x=None, y=None, delay=1*)

Drag mouse

Drags mouse to an element based on pixel position determined by x and y coordinates

Parameters

- **x** (*int*) – X-coördinate
- **y** (*int*) – Y-coördinate
- **delay** (*int*, *optional*) – Delay between movements in seconds, standard value is 1s.

Returns

Drag mouse

Example

```
>>> # Use coordinates to move and drag mouse
>>> move_mouse_to_coordinates(x=100, y=100)
>>> drag_mouse_to_coordinates(x=1, y=1)
```

Keywords mouse, osd, move mouse, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-arrows-alt

`automagica.activities.drag_mouse_to(automagica_id=None, delay=1)`

Drag mouse

Drags mouse to an element based on the element ID (vision)

Parameters

- **automagica_id** (*automagica_id*) – ID of the element. To define an element and attach an ID one can use the Automagica Wand. The recorder uses vision to detect an element and can be invoked with the recorder() function.
- **delay** (*int*, *optional*) – Delay before movement in seconds.

Returns

Drag mouse

Example

```
>>> # Use recorder to find an element ID
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↪ also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↪ found, use the recorder
>>> drag_mouse_to('qf41')
```

Keywords mouse, osd, move mouse, right click, right, rightclick, overlay, show, display, mouse automation, click, right click, mouse button, move mouse, position, pixel

Icon las la-arrows-alt

6.9 Image

`automagica.activities.random_screen_snippet(size=100, output_path=None)`

Random screen snippet

Take a random square snippet from the current screen. Mainly for testing and/or development purposes.

Parameters

- **size** (*int*, *optional*) – Size (width and height) in pixels for square snippet. Default value is 100 pixels

- **output_path** (*output_file*) – Path where snippet will be saved. Default value is home directory with name 'random_screensnippet.jpg'

Extension output_path jpg

Returns

Path to snippet

Example

```
>>> random_screen_snippet()
'C:\Users\\random_screensnippet.jpg'
```

Keywords image, random, testing, screengrab, snippet

Icon las la-crop-alt

automagica.activities.**take_screenshot** (*output_path=None*)

Screenshot

Take a screenshot of current screen.

Parameters **output_path** (*output_path*) – Path to save screenshot. Default value is home directory with name 'screenshot.jpg'.

Extesion output_path jpg

Returns

Path to save screenshot

Example

```
>>> new_screenshot = take_screenshot()
>>> open_file(new_screenshot)
'C:\Users\\screenshot.jpg'
```

Keywords image, screenshot, printscreen,

Icon las la-expand

6.10 Folder Operations

automagica.activities.**get_files_in_folder** (*input_path=None*, *extension=None*,
show_full_path=True, *scan_subfolders=False*)

List files in folder

List all files in a folder (and subfolders) Checks all folders and subfolders for files. This could take some time for large repositories.

Parameters

- **input_path** (*input_dir*) – Path of the folder to retrieve files from. Default folder is the home directory.
- **extension** (*string*, *optional*) – Optional filter on certain extensions, for example 'pptx', 'exe', 'xlsx', 'txt', .. Default value is no filter.
- **show_full_path** (*bool*, *optional*) – Set this to True to show full path, False will only show file or dirname. Default is True

Scan_subfolders Boolean to scan subfolders or not. Not that depending on the folder and hardware this activity could take some time if scan_subfolders is set to True

Returns

List of files with their full path

Example

```
>>> # List all files in the homedirectory
>>> get_files_in_folder()
['C:\Users\<username>\file1.jpg', 'C:\Users\<username>\file2.txt', ... ]
```

Keywords folder, files, explorer, nautilus, folder, file, create folder, get files, list files, all files, overview, get files

Icon las la-search

automagica.activities.**create_folder** (*path=None*)

Create folder

Creates new folder at the given path.

Parameters **path** (*input_dir, optional*) – Full path of folder that will be created. If no path is specified a folder called ‘new_folder’ will be made in home directory. If this folder already exists 8 random characters will be added to the name.

Returns

Path to new folder as string

Example

```
>>> # Create folder in the home directory
>>> create_folder()
'C:\Users\<username>\new_folder'
```

Keywords create folder, folder, new folder, folders, make folder, new folder, folder manipulation, explorer, nautilus

Icon las la-folder-plus

automagica.activities.**rename_folder** (*input_path, output_name=None*)

Rename folder

Rename a folder

Parameters

- **input_path** (*input_dir, optional*) – Full path of folder that will be renamed
- **output_name** (*string, optional*) – New name. By default folder will be renamed to original folder name with ‘_renamed’ added to the folder name.

Returns

Path to renamed folder as a string.

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Rename the folder
>>> rename_folder(testfolder, output_name='testfolder_brand_new_name')
'C:\Users\\testfolder_brand_new_name'
```

Keywords folder, rename, rename folder, organise folder, folders, folder manipulation, explorer, nautilus

Icon las la-folder

automagica.activities.**show_folder**(input_path=None)

Open a folder

Open a folder with the default explorer.

Parameters **input_path**(input_dir, optional) – Full path of folder that will be opened.
Default value is the home directory

Returns

Path to open folder as a string

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Open folder
>>> show_folder(testfolder)
'C:\Users\\new_folder'
```

Keywords folder, open, open folder, explorer, nautilus

Icon las la-folder-open

automagica.activities.**move_folder**(input_path, output_path=None)

Move a folder

Moves a folder from one place to another.

Parameters

- **input_path**(input_dir) – Full path to the source location of the folder
- **output_path** – Full path to the destination location of the folder, defaults to input_path with ‘_moved’ added

```
>>> # Make new folder in home directory for illustration
>>> # If no new_folder exists in home dir this will be called new_folder
>>> testfolder = create_folder()
>>> # Make a second new folder
>>> # Since new_folder already exists this folder will get a random id added (in_
↪this case abc1)
>>> testfolder_2 = create_folder()
>>> # Move testfolder in testfolder_2
>>> move_folder(testfolder, testfolder_2)
'C:\Users\\new_folder_abc1\new_folder'
```

Keywords folder, move, move folder, explorer, nautilus, folder manipulation

Icon las la-folder

`automagica.activities.remove_folder(input_path, allow_root=False, delete_read_only=True)`

Remove folder

Remove a folder including all subfolders and files. For the function to work optimal, all files and subfolders in the main targetfolder should be closed.

Parameters

- **input_path** (*input_dir*) – Full path to the folder that will be deleted
- **allow_root** (*bool, optional*) – Allow paths with an arbitrary length of 10 characters or shorter to be deleted. Default value is False.
- **delete_read_only** (*bool, optional*) – Option to delete read only

Returns

Path to deleted folder as a string

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Check if folder exists
>>> print( folder_exists(testfolder) ) # Should print True
>>> # Remove folder
>>> remove_folder(testfolder)
>>> # Check again if folder exists
>>> folder_exists(testfolder)
False
```

Keywords folder, delete folder, delete, nautilus, folder manipulation, explorer, delete folder, remove, remove folder

Icon `las la-folder-minus`

`automagica.activities.empty_folder(input_path, allow_root=False)`

Empty folder

Remove all contents from a folder For the function to work optimal, all files and subfolders in the main targetfolder should be closed.

Parameters

- **input_path** (*input_dir*) – Full path to the folder that will be emptied
- **allow_root** – Allow paths with an arbitrary length of 10 characters or shorter to be emptied. Default value is False.

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Make new text file in this folder
>>> text_file_location = make_text_file(output_path = testfolder)
>>> # Print all files in the testfolder
>>> get_files_in_folder(testfolder)
>>> # Empty the folder
>>> empty_folder(testfolder)
>>> # Check what is in the folder
>>> get_files_in_folder(testfolder)
[]
```

Keywords folder, empty folder, delete, empty, clean, clean folder, nautilus, folder manipulation, explorer, delete folder, remove, remove folder

Icon las la-folder-minus

`automagica.activities.folder_exists(path)`

Checks if folder exists

Check whether folder exists or not, regardless if folder is empty or not.

Parameters `input_path(input_dir)` – Full path to folder

Returns

Boolean

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Check if folder exists
>>> folder_exists(testfolder)
True
```

Keywords folder, folder exists, nautilus, explorer, folder manipulation, files

Icon las la-folder

`automagica.activities.copy_folder(input_path, output_path=None)`

Copy a folder

Copies a folder from one place to another.

Parameters

- **input_path(input_dir)** – Full path to the source location of the folder
- **output_path(output_dir, optional)** – Full path to the destination location of the folder. If no path is specified folder will get copied in the input directory with ‘_copied’ added

Returns

Path to new folder as string

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Copy this folder
>>> # Since new_folder already exists in home dir this folder will get a random_
↳id added (in this case abc1)
>>> copy_folder(testfolder)
```

Keywords folder, move, move folder, explorer, nautilus, folder manipulation

Icon lar la-folder

`automagica.activities.zip_folder(input_path, output_path=None)`

Zip

Zip folder and its contents. Creates a .zip file.

Parameters

- **input_path** (*input_dir*) – Full path to the source location of the folder that will be zipped
- **output_path** (*output_dir*, *optional*) – Full path to save the zipped folder. If no path is specified a folder with the original folder with ‘_zipped’ added

Returns

Path to zipped folder

Example

```
>>> # Make new folder in home directory for illustration
>>> testfolder = create_folder()
>>> # Zip this folder
>>> zip_folder(testfolder)
```

Keywords zip, zipping, winrar, rar, 7zip, compress, unzip

Icon las la-archive

`automagica.activities.unzip` (*input_path*, *output_path=None*)

Unzip

Unzips a file or folder from a .zip file.

Parameters

- **input_path** (*input_dir*) – Full path to the source location of the file or folder that will be unzipped
- **to_path** – Full path to save unzipped contents. If no path is specified the unzipped contents will be stored in the same directory as the zipped file is located.

Returns

Path to unzipped folder

Example

```
>>> # Make new file in home directory for illustration
>>> testfolder = create_folder()
>>> # Add some files to this folder
>>> make_text_file(output_path = testfolder)
>>> # Zip this folder
>>> zipped_folder = zip_folder(testfolder)
>>> # Unzip this folder
>>> unzip(zipped_folder)
```

Keywords zip, zipping, winrar, rar, 7zip, compress, unzip

Icon las la-archive

`automagica.activities.most_recent_file` (*input_path=None*)

Return most recent file in directory

Return most recent file in directory

Parameters **input_path** (*input_dir*, *optional*) – Path which will be scanned for most recent file, defaults to homedir

Returns

Path to most recent file

Example

```
>>> # Find most recent file in homedir
>>> most_recent_file()
```

Keywords find file, file, recent, newest, latest, recent

Icon las la-clock

6.11 Delay

`automagica.activities.wait(seconds=1)`

Wait

Make the robot wait for a specified number of seconds. Note that this activity is blocking. This means that subsequent activities will not occur until the the specified waiting time has expired.

Parameters **seconds** – Time in seconds to wait

```
>>> print('Start the wait')
>>> wait()
>>> print('The wait is over')
```

Keywords wait, sleep, time, timeout, time-out, hold, pause

Icon las la-hourglass

`automagica.activities.wait_folder_exists(input_path, timeout=60)`

Wait for folder

Waits until a folder exists. Not that this activity is blocking and will keep the system waiting.

Parameters

- **input_path**(*input_dir*, *optional*) – Full path to folder.
- **timeout** – Maximum time in seconds to wait before continuing. Default value is 60 seconds.

```
>>> # Create a random folder
>>> testfolder = create_folder()
>>> # Wait for the snippet to be visible
>>> wait_folder_exists(testfolder)
```

Keywords image matching, wait, pause, vision, template, template matching

Icon las la-hourglass

6.12 Word Application

`class automagica.activities.Word(file_path=None, visible=True)`

append_text (*text*)

Append text

Append text at end of Word document.

Parameters **text** – Text to append to document

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
```

Keywords word, editor, text, text edit, office, document, microsoft word, doc, docx**Icon** lar la-file-word**export_to_html** (*output_path=None*)

Export to HTML

Export to HTML

Parameters **file_path** – Output path where HTML file will be exported to. Default path is home directory with filename 'html_export.html'.**Extension** **output_path** html**Example**

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.replace_text('sample', 'real')
>>> word.export_to_html('output.html')
```

Keywords word, html, document, export, save as, doc, docx**Icon** las la-html5**export_to_pdf** (*output_path=None*)

Export to PDF

Export the document to PDF

Parameters **output_path** (*output_file, optional*) – Output path where PDF file will be exported to. Default path is home directory with filename 'pdf_export.pdf'.**Extension** **output_path** pdf**Example**

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.replace_text('sample', 'real')
>>> word.export_to_pdf('output.pdf')
```

Keywords word, pdf, document, export, save as, doc, docx**Icon** lar la-file-pdf

quit()

Quit Word

This closes Word, make sure to use 'save' or 'save_as' if you would like to save before quitting.

Example

```
>>> # Open Word
>>> word = Word()
>>> # Quit Word
>>> word.quit()
```

Keywords word, wordfile, doc quit, close, doc, docx**Icon** la-file-word**read_all_text** (*return_as_list=False*)

Read all text

Read all the text from a document

Parameters **return_as_list** (*bool, optional*) – Set this paramater to True to return text as a list of strings. Default value is False.

Returns

Text from the document

Example

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.replace_text('sample', 'real')
>>> word.read_all_text()
'This is real text'
```

Keywords word, extract, text, document, doc, docx**Icon** lar la-file-word**replace_text** (*placeholder_text, replacement_text*)

Replace text


Can be used for example to replace arbitrary placeholder value. For example when using template document, using 'XXXX' as a placeholder. Take note that all strings are case sensitive.

Parameters

- **placeholder_text** (*string*) – Placeholder text value in the document, this will be replaced, e.g. 'Company Name'
- **replacement_text** – Text to replace the placeholder values with. It is recommended to make this unique to avoid wrongful replacement, e.g. 'XXXX_placeholder_XXX'

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.replace_text('sample', 'real')
```

Keywords word, replace, text, template, doc, docx

Icon  `la-file-word`

save()


Save

Save active Word document

Example

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.save_as('automagica_document.docx')
```

Keywords word, save, document, doc, docx

Icon  `la-file-word`

save_as(output_path)

Save As

Save active Word document to a specific location


Parameters **output_path** (*output_file*) – Enter a path to open Word with an existing Word file.

Extension **output_file** docx

Example

```
>>> # Start Word
>>> word = Word()
>>> word.append_text('This is sample text')
>>> word.save_as('document.odt')
```

Keywords word, save as, document, doc, docx

Icon  `la-file-word`

set_footers(text)


Set footers

Set the footers of the document

Parameters **text** – Text to put in the footer

```
>>> # Start Word
>>> word = Word()
>>> word.set_footers('This is a footer!')
```

Keywords word, footer, footers, doc, docx

Icon  `la-file-word`

set_headers(text)

Set headers

Set the headers of the document

Parameters **text** – Text to put in the header

```
>>> # Start Word
>>> word = Word()
>>> word.set_headers('This is a header!')
```

Keywords word, header, headers, doc, docx

Icon las la-subscript

6.13 Word File

class automagica.activities.**WordFile** (*file_path=None*)

append_text (*text, auto_save=True*)

Append text

Append text at the end of the document

Parameters

- **text** (*streing*) – Text to append
- **auto_save** – Save document after performing activity. Default value is True

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
```

Keywords word, append text, add text

Icon las la-file-word

read_all_text (*return_as_list=False*)

Read all text

Read all the text from the document

Parameters **return_as_list** (*bool, optional*) – Set this paramater to True to return text as a list of strings. Default value is False.

Returns

Text of the document

Example

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
>>> wordfile.read_all_text()
'Some sample text'
```

Keywords word, read, text, file

Icon las la-file-word

replace_text (*placeholder_text, replacement_text, auto_save=True*)

Replace all

Replaces all occurences of a placeholder text in the document with a replacement text.

Can be used for example to replace arbitrary placeholder value. For example when using template slideck, using 'XXXX' as a placeholder. Take note that all strings are case sensitive.

Parameters

- **placeholder_text** (*string*) – Placeholder text value (string) in the document, this will be replaced, e.g. 'Company Name'
- **replacement_text** (*string*) – Text (string) to replace the placeholder values with. It is recommended to make this unique to avoid wrongful replacement, e.g. 'XXXX_placeholder_XXX'
- **auto_save** – Save document after performing activity. Default value is True

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
>>> wordfile.replace_text('sample', 'real')
```

Keywords word, replace text, template

Icon las la-file-word

save()

Save

Save document

Example

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
>>> wordfile.save()
```

Keywords word, save, store

Icon las la-file-word

save_as(output_path)

Save as

Save file on specified path

Parameters **output_path** (*output_file*) – Path to save Wordfile to

Extension **output_path** docx

Example

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
>>> wordfile.save_as('document.docx')
```

Keywords word, save as, store

Icon las la-file-word

set_headers(text, auto_save=True)

Set headers

Set headers of Word document

Parameters

- **text** (*string*) – Text to put in the header
- **auto_save** – Save document after performing activity. Default value is True

```
>>> wordfile = WordFile()
>>> wordfile.append_text('Some sample text')
>>> wordfile.set_headers('This is a header')
```

Keywords word, header text

Icon las la-file-word

6.14 Outlook Application

class automagica.activities.Outlook (*account_name=None*)

add_contact (*email, first_name="", last_name=""*)

Add a contact

Add a contact to Outlook contacts

Parameters

- **email** (*string, optional*) – The e-mail address for the contact
- **first_name** (*string, optional*) – First name for the contact (optional)
- **last_name** – Last name for the contact (optional)

```
>>> outlook = Outlook()
>>> outlook.add_contact('sales@automagica.com')
```

Keywords outlook, create contact, add contact

Icon las la-mail-bulk

delete_mails (*folder_name='Inbox', limit=0, subject_contains="", body_contains="", sender_contains=""*)

Delete e-mails

Deletes e-mail messages in a certain folder. Can be specified by searching on subject, body or sender e-mail.

Parameters

- **folder_name** (*string, optional*) – Name of the Outlook folder, can be found using *get_folders*
- **limit** (*int*) – Maximum number of e-mails to delete in one go
- **subject_contains** (*string, optional*) – Only delete e-mail if subject contains this
- **body_contains** (*string, optional*) – Only delete e-mail if body contains this
- **sender_contains** – Only delete e-mail if sender contains this

```
>>> outlook = Outlook()
>>> outlook.delete_mails(subject_contains='hello')
```

Keywords outlook, remove e-mails, delete mail, remove mail

Icon las la-mail-bulk

get_contacts (*fields=None*)

Retrieve contacts

Retrieve all contacts

Parameters **fields** (*tuple, optional*) – Fields can be specified as a tuple with their exact names. Standard value is None returning “LastName”, “FirstName” and “EmailAddress”.

Returns

List of dictionaries containing the contact details.

Example

```
>>> outlook = Outlook()
>>> outlook.get_contacts()
[
    {
        'LastName': 'Doe',
        'FirstName': 'John',
        'EmailAddress': 'john@test.com'
    }
]
```

Keywords outlook, get contacts, download contacts, rolodex

Icon las la-mail-bulk

get_folders (*limit=999*)

Retrieve folders

Retrieve list of folders from Outlook

Parameters **limit** – Maximum number of folders to retrieve

```
>>> outlook = Outlook()
>>> outlook.get_folders()
['Inbox', 'Sent', ...]
```

Keywords outlook, get folders, list folders

Icon las la-mail-bulk

get_mails (*folder_name='Inbox', fields=None*)

Retrieve e-mails

Retrieve list of messages from Outlook

Parameters

- **folder_name** (*string, optional*) – Name of the Outlook folder, can be found using *get_folders*.

- **limit** (*int*, *optional*) – Number of messages to retrieve
- **fields** (*tuple*, *optional*) – Fields (properties) of e-mail messages to give, requires tuple Standard is 'Subject', 'Body', 'SentOn' and 'SenderEmailAddress'.

Returns

List of dictionaries containing the e-mail messages with from, to, subject, body and html.

Example

```
>>> outlook = Outlook()
>>> outlook.get_mails()
```

Keywords outlook, retrieve e-mail, receive e-mails, process e-mails, get mails

Icon las la-mail-bulk

move_mails (*source_folder_name*='Inbox', *target_folder_name*='Archive', *limit*=0, *subject_contains*="", *body_contains*="", *sender_contains*="")
Move e-mails

Move e-mail messages in a certain folder. Can be specified by searching on subject, body or sender e-mail.

Parameters

- **source_folder_name** (*string*, *optional*) – Name of the Outlook source folder from where e-mails will be moved, can be found using *get_folders*
- **target_folder_name** (*string*, *optional*) – Name of the Outlook destination folder to where e-mails will be moved, can be found using *get_folders*
- **limit** (*int*) – Maximum number of e-mails to move in one go
- **subject_contains** (*string*, *optional*) – Only move e-mail if subject contains this
- **body_contains** (*string*, *optional*) – Only move e-mail if body contains this
- **sender_contains** – Only move e-mail if sender contains this

```
>>> outlook = Outlook()
>>> outlook.move_mails(subject_contains='move me')
```

Keywords outlook, move e-mail, move e-mail to folder

Icon las la-mail-bulk

quit ()
Quit

Close the Outlook application

Example

```
>>> outlook = Outlook()
>>> outlook.quit()
```

Keywords outlook, close, quit

Icon las la-mail-bulk

save_attachments (*folder_name='Inbox', output_path=None*)

Save attachments

Save all attachments from certain folder

Parameters

- **folder_name** (*string, optional*) – Name of the Outlook folder, can be found using *get_folders*.
- **output_path** (*output_dir, optional*) – Path where attachments will be saved. Default is the home directory.

Returns

List of paths to saved attachments.

Example

```
>>> outlook = Outlook()
>>> outlook.save_attachments()
['Attachment.pdf', 'Signature_image.jpeg']
```

Keywords outlook, save attachments, download attachments, extract attachments

Icon las la-mail-bulk

send_mail (*to_address, subject="", body="", html_body=None, attachment_paths=None*)

Send e-mail

Send an e-mail using Outlook

Parameters

- **to_address** – The e-mail address the e-mail should be sent to
- **subject** (*string, optional*) – The subject of the e-mail
- **body** (*text, optional*) – The text body contents of the e-mail
- **html_body** (*text, optional*) – The HTML body contents of the e-mail (optional)
- **attachment_paths** – List of file paths to attachments

```
>>> outlook = Outlook()
>>> outlook.send_mail('test@test.com', subject='Hello world', body='Hi there')
```

Keywords outlook, send e-mail, send mail

Icon las la-mail-bulk

6.15 Excel Application

class automagica.activities.**Excel** (*file_path=None, visible=True*)

activate_first_empty_cell_down ()

Activate first empty cell down

Activates the first empty cell going down

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Write some cells
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(1, 2, 'Filled')
>>> excel.write_cell(1, 3, 'Filled')
>>> # Activate the first empty cell going down, in this case cell A4 or (1,4)
>>> excel.activate_first_empty_cell_down()
```

Keywords excel, first empty cell, down

Icon las la-file-excel

activate_first_empty_cell_left()

Activate first empty cell left

Activates the first empty cell going left

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(1, 2, 'Filled')
>>> excel.write_cell(1, 3, 'Filled')
>>> excel.activate_first_empty_cell_left()
```

Keywords excel, first empty cell, left

Icon las la-file-excel

activate_first_empty_cell_right()

Activate first empty cell right

Activates the first empty cell going right

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Write some cells
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(1, 2, 'Filled')
>>> excel.write_cell(1, 3, 'Filled')
>>> # Activate the first empty cell going right, in this case cell B1 or (2,1)
>>> excel.activate_first_empty_cell_right()
```

Keywords excel, first empty cell, right

Icon las la-file-excel

activate_first_empty_cell_up()

Activate first empty cell up

Activates the first empty cell going up

Example


```

>>> # Open Excel
>>> excel = Excel()
>>> # Write some cells
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(1, 2, 'Filled')
>>> excel.write_cell(1, 3, 'Filled')
>>> # Activate first empty cell
>>> excel.activate_first_empty_cell_up()

```

Keywords excel, first empty cell, up

Icon las la-file-excel

activate_range (*range_*)

Activate range

Activate a particular range in the currently active workbook

Parameters **range** – Range to activate, e.g. “A1:D10”

Example

```

>>> # Open Excel
>>> excel = Excel()
>>> # Activate a cell range
>>> excel.activate_range('A1:D5')

```

Keywords excel, activate range, make selection, select cells, select range

Icon las la-file-excel

activate_worksheet (*name*)

Activate worksheet

Activate a worksheet in the current Excel document by name

Parameters **name** – Name of the worksheet to activate

```

>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Add another worksheet
>>> excel.add_worksheet('Another Worksheet')
>>> # Activate the first worksheet
>>> excel.activate_worksheet('My Example Worksheet')

```

Keywords excel, activate worksheet, set worksheet, select worksheet, select tab, activate tab

Icon las la-file-excel

add_worksheet (*name=None*)

Add worksheet

Adds a worksheet to the current workbook

Parameter name Give the sheet a name (optional)

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add a worksheet
>>> excel.add_worksheet('My Example Worksheet')
```

Keywords excel, add worksheet, add tab, insert worksheet, new worksheet

Icon las la-file-excel

delete_column(*column*)

Delete column

Delete a column from the currently active worksheet. Existing columns will shift to the left.

Parameters **column** – Column letter (string) where to delete column e.g. 'A'

```
>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(2, 2, 'Filled')
>>> excel.write_cell(3, 3, 'Filled')
>>> excel.delete_column('A')
```

Keywords excel, delete column, remove column

Icon las la-file-excel

delete_row(*row*)

Delete row in Excel

Deletes a row from the currently active worksheet. Existing data will shift up.

Parameters **row** – Row number (integer) where to delete row e.g 1

```
>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(2, 2, 'Filled')
>>> excel.write_cell(3, 3, 'Filled')
>>> excel.delete_row(2)
```

Keywords excel, delete row, remove row

Icon las la-file-excel

export_to_pdf(*output_path=None*)

Export to PDF

Export to PDF

Parameters **path** – Output path where PDF file will be exported to. Default path is home directory with filename 'pdf_export.pdf'.

Extension **output_path** pdf

Example

```

>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(2, 2, 'Filled')
>>> excel.write_cell(3, 3, 'Filled')
>>> excel.export_to_pdf('output.pdf')

```

Keywords excel, save as pdf, export to pdf, export as pdf

Icon las la-file-excel

get_table(*name*)

Get table

Get table data from the currently active worksheet by name of the table

Parameters *name* (*string*) – Table name

Returns

List of dictionaries for each row with as key the column name

Example

```

>>> # Open Excel
>>> excel = Excel()
>>> # Create a table (Table1)
>>> data = [
    {
        'Column A': 'Data Row 1 for A',
        'Column B': 'Data Row 1 for B',
        'Column C': 'Data Row 1 for C',
    },
    {
        'Column A': 'Data Row 2 for A',
        'Column B': 'Data Row 2 for B',
        'Column C': 'Data Row 2 for C',
    }
]
>>> excel.insert_data_as_table(data)
>>> # Get the table
>>> excel.get_table('Table1')
[['Column A', 'Column B', 'Column C'], ['Row 1 A Data', 'Row 1 B Data', 'Row_
↵1 C Data'], ...]

```

Keywords excel, worksheet names, tab names

Icon las la-file-excel

get_worksheet_names()

Get worksheet names

Get names of all the worksheets in the currently active workbook

Returns

List of worksheet names

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add a worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Get all worksheet names
>>> excel.get_worksheet_names()
['Sheet1', 'My Example Worksheet']
```

Keywords excel, worksheet names, tab names

Icon las la-file-excel

insert_data_as_table (*data*, *range_*='A1', *table_style*='TableStyleMedium2')

Insert data as table

Insert list of dictionaries as a table in Excel

Parameters

- **data** (*string*) – List of dictionaries to write as table
- **range** – Range or startingpoint for table e.g. 'A1'

```
>>> excel = Excel()
>>> data = [
    {
        'Column A': 'Data Row 1 for A',
        'Column B': 'Data Row 1 for B',
        'Column C': 'Data Row 1 for C',
    },
    {
        'Column A': 'Data Row 2 for A',
        'Column B': 'Data Row 2 for B',
        'Column C': 'Data Row 2 for C',
    }
]
>>> excel.insert_data_as_table(data)
```

Keywords excel, insert data, insert table, create table

Icon las la-file-excel

insert_empty_column (*column*)

Insert empty column

Inserts an empty column in the currently active worksheet. Existing columns will shift to the right.

Parameters **column** – Column letter where to insert empty column e.g. 'A'

```
>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(2, 2, 'Filled')
>>> excel.write_cell(3, 3, 'Filled')
>>> excel.insert_empty_column('A')
```

Keywords excel, insert column, add column

Icon las la-file-excel

insert_empty_row (*row*)

Insert empty row

Inserts an empty row to the currently active worksheet

Parameters *row* – Row number where to insert empty row e.g 1

```
>>> # Open Excel
>>> excel = Excel()
>>> excel.write_cell(1, 1, 'Filled')
>>> excel.write_cell(1, 2, 'Filled')
>>> excel.write_cell(1, 3, 'Filled')
>>> excel.insert_empty_row(2)
```

Keywords excel, insert row, add row, empty row

Icon las la-file-excel

quit ()

Quit Excel

This closes Excel, make sure to use 'save' or 'save_as' if you would like to save before quitting.

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Quit Excel
>>> excel.quit()
```

Keywords excel, exit, quit, close

Icon las la-file-excel

read_cell (*column*, *row*)

Read cell

Read a cell from the currently active workbook and active worksheet

Parameters

- **column** (*int*) – Column number (integer) to read
- **row** (*int*) – Row number (integer) to read

Returns

Cell value

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Insert a text into the first cell
>>> excel.write_cell(1,1, 'Hello World!')
>>> excel.read_cell(1,1)
'Hello World!'
```

Keywords excel, cell, read cell, read data

Icon las la-file-excel

read_cell_formula (*column, row, formula*)

Read cell formula

Read the formula from a particular cell

Parameters

- **column** (*int*) – Column number to read formula
- **row** (*int*) – Row number to read formula

Returns

Cell value

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Write a formula to the first cell
>>> excel.write_cell_formula(1, 1, '=1+1')
>>> # Read the cell
>>> excel.read_cell_formula(1, 1)
'=1+1'
```

Keywords excel, read formula, read calculation

Icon las la-file-excel

read_range (*range_*)

Read range

Read a range of cells from the currently active worksheet in the active workbook

Parameters **range** (*string*) – Range to read from, e.g. “A1:D10”

Return value Values in param range

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Insert a text in every cell in this range
>>> excel.write_range('A1:D5', 'Hello World!')
>>> # Read the same range
>>> excel.read_range('A1:D5')
[['Hello World', 'Hello World', 'Hello World', 'Hello World'], ...]
```

Keywords excel, cell, read range, read data

Icon las la-file-excel

read_worksheet (*name=None, headers=False*)

Read worksheet

Read data from a worksheet as a list of lists

Parameters

- **name** (*string, optional*) – Optional name of worksheet to read. If no name is specified will take active sheet
- **headers** (*bool, optional*) – Boolean to treat first row as headers. Default value is False

Returns

List of dictionaries with sheet data

Example

```
>>> # Open excel
>>> excel = Excel()
>>> Write some cells
>>> excel.write_cell(1, 1, 'A')
>>> excel.write_cell(1, 2, 'B')
>>> excel.write_cell(1, 3, 'C')
>>> excel.read_worksheet()
[['A'], ['B'], ['C']]
```

Keywords excel, read worksheet, export data, read data

Icon las la-file-excel

run_macro (*name*)

Run macro

Run a macro by name from the currently active workbook

Parameters **name** – Name of the macro to run.

```
>>> excel = Excel('excel_with_macro.xlsx')
>>> # Run the macro
>>> excel.run_macro('Macro1')
```

Keywords excel, run macro, run vba

Icon las la-file-excel

save ()

Save

Save the current workbook. Defaults to homedir

Example

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Save the workbook
>>> excel.save()
```

Keywords excel, save, store

Icon las la-file-excel

save_as (*output_path*)

Save as

Save the current workbook to a specific path

Parameters **output_path** (*output_file*) – Path where workbook will be saved.**Extension** **output_path** `xlsx`**Example**

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Save the workbook to the current working directory
>>> excel.save_as('output.xlsx')
```

Keywords `excel`, `save as`, `export`**Icon** `las la-file-excel`**write_cell** (*column*, *row*, *value*)

Write cell

Write to a specific cell in the currently active workbook and active worksheet

Parameters

- **column** (*int*) – Column number (integer) to write
- **row** (*int*) – Row number (integer) to write
- **value** – Value to write to specific cell

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Insert a text into the first cell
>>> excel.write_cell(1,1, 'Hello World!')
```

Keywords `excel`, `cell`, `insert cell`, `insert data`**Icon** `las la-file-excel`**write_cell_formula** (*column*, *row*, *formula*)

Write cell formula

Write a formula to a particular cell

Parameters

- **column** (*int*) – Column number to write formula
- **row** (*int*) – Row number to write formula
- **value** – Formula to write to specific cell e.g. “=10*RAND()”


```
>>> # Open Excel
>>> excel = Excel()
>>> # Write a formula to the first cell
>>> excel.write_cell_formula(1, 1, '=1+1')
```

Keywords excel, insert formula, insert calculation, insert calculated cell

Icon las la-file-excel

write_range (*range_*, *value*)

Write range

Write to a specific range in the currently active worksheet in the active workbook

Parameters

- **range** (*string*) – Range to write to, e.g. “A1:D10”
- **value** – Value to write to range

```
>>> # Open Excel
>>> excel = Excel()
>>> # Add the first worksheet
>>> excel.add_worksheet('My Example Worksheet')
>>> # Insert a text in every cell in this range
>>> excel.write_range('A1:D5', 'Hello World!')
```

Keywords excel, cell, write range, read data

Icon las la-file-excel

6.16 Excel File

class automagica.activities.**ExcelFile** (*file_path=None*)

activate_worksheet (*name*)

Activate worksheet

Activate a worksheet. By default the first worksheet is activated.

Parameters **name** – Name of the worksheet to activate.

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Add some worksheets
>>> excel_file.add_worksheet('My Example Worksheet')
>>> excel_file.add_worksheet('Another Worksheet')
>>> # Activate a worksheet
>>> excel_file.activate_worksheet('My Example Worksheet')
```

Keywords excel, activate tab, activate worksheet

Icon las la-file-excel

add_worksheet (*name*, *auto_save=True*)

Add worksheet

Add a worksheet

Parameters

- **name** (*string*) – Name of the worksheet to add
- **auto_save** – Save document after performing activity. Default value is True

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Add a worksheet
>>> excel_file.add_worksheet('My Example Worksheet')
>>> # List all the worksheets
>>> excel.get_worksheet_names()
```

Keywords excel, add worksheet, worksheet**Icon** las la-file-excel**get_worksheet_names** ()

Get worksheet names

Get worksheet names

Returns

List of worksheet names

Example

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Add some worksheets
>>> excel_file.add_worksheet('My Example Worksheet')
>>> excel_file.add_worksheet('Another Worksheet')
>>> # Get the worksheet names
>>> excel_file.get_worksheet_names()
['My Example Worksheet', 'Another Worksheet']
```

Keywords excel, worksheet names, worksheet,**Icon** las la-file-excel**read_cell** (*column*, *row*)

Read cell

Read a cell based on column and row

Parameters

- **column** – Column number (integer) to read
- **row** (*int*) – Row number (integer) to read

Returns

Cell value

Example

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Add a worksheet
>>> excel_file.add_worksheet('My Example Worksheet')
>>> # Write the first cell
>>> excel_file.write_cell(1, 1, 'Filled!')
>>> # Read the first cell
>>> excel_file.read_cell(1, 1)
'Filled!'
```

Keywords excel, read cell, read

Icon las la-file-excel

save()

Save as

Save file

Example

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Ad a worksheet
>>> excel_file.add_worksheet('My Example Worksheet')
>>> # Save the Excel file
>>> excel_file.save()
```

Keywords excel, save as, export, save

Icon las la-file-excel

save_as(output_path)

Save as

Save file as

Parameters **file_path** – Path where workbook will be saved

Extension **output_path** **xlsx**

Example

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Ad a worksheet
>>> excel_file.add_worksheet('My Example Worksheet')
>>> # Save the Excel file
>>> excel_file.save_as('output.xlsx')
```

Keywords excel, save as, export, save

Icon las la-file-excel

to_dataframe()

Export file to dataframe

Export to pandas dataframe

Example

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Convert to Dataframe
>>> df = excel_file.to_dataframe()
```

Keywords excel, open, start, xlsx, dataframe,

Icon las la-file-excel

write_cell (*column*, *row*, *value*, *auto_save=True*)

Write cell

Write a cell based on column and row

Parameters

- **column** (*int*) – Column number (integer) to write
- **row** (*int*) – Row number (integer) to write
- **value** (*string*) – Value to write to specific cell
- **auto_save** – Save document after performing activity. Default value is True

```
>>> # Open a new Excel file
>>> excel_file = ExcelFile()
>>> # Add a worksheet
>>> excel_file.add_worksheet('My Example Worksheet')
>>> excel_file.write_cell(1, 1, 'Filled!')
```

Keywords excel, write cell, insert data

Icon las la-file-excel

6.17 PowerPoint Application

class automagica.activities.**PowerPoint** (*file_path=None*, *visible=True*, *add_slide=True*)

add_slide (*index=None*, *type='blank'*)

Add PowerPoint Slides

Adds slides to a presentation

Parameters **index** (*int*, *optional*) – Index where the slide should be inserted. Default value is as final slide.

Parameter type Type of the slide to be added. Supports following types: blank, chart, text, title and picture.

Options type ['blank', 'chart', 'text', 'title', 'picture']

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add a first slide
>>> powerpoint.add_slide()
```

Keywords powerpoint, ppt, add, add slide powerpoint, slides

Icon las la-file-powerpoint

add_text (*text*, *index=None*, *font_size=48*, *font_name=None*, *bold=False*, *margin_bottom=100*, *margin_left=100*, *margin_right=100*, *margin_top=100*)

Text to slide

Add text to a slide

Parameters

- **index** (*int*, *optional*) – Slide index to add text. If none is specified, a new slide will be added as final slide
- **text** (*string*, *optional*) – Text to be added
- **font_size** (*int*, *optional*) – Fontsize, default value is 48
- **font_name** (*string*, *optional*) – Fontname, if not specified will take default PowerPoint font
- **bold** (*bool*, *optional*) – Toggle bold with True or False, default value is False
- **margin_bottom** (*int*, *optional*) – Margin from the bottom in pixels, default value is 100 pixels
- **margin_left** (*int*, *optional*) – Margin from the left in pixels, default value is 100 pixels
- **margin_right** (*int*, *optional*) – Margin from the right in pixels, default value is 100 pixels
- **margin_top** – Margin from the top in pixels, default value is 100 pixels

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add slide with text
>>> powerpoint.add_text(text='Sample Text')
```

Keywords powerpoint, ppt, text, add text, slides

Icon las la-file-powerpoint

delete_slide (*index=None*)

Delete slide

Delete a slide

Parameters **index** – Slide index to be deleted. If none is specified, last slide will be deleted

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add some slides
>>> powerpoint.add_slide()
>>> powerpoint.add_slide()
>>> # Delete last slide
>>> powerpoint.delete_slide()
```

Keywords powerpoint, ppt, delete, delete slide

Icon las la-file-powerpoint

export_slides_to_images (*output_path=None, type='png'*)

Slides to images

Export PowerPoint slides to separate image files

Parameters

- **output_path** – Output path where image files will be exported to. Default path is home directory.
- **type** – Output type of the images, supports 'png' and 'jpg' with 'png' as default value

Options type ['jpg', 'png']

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add some slides with text
>>> powerpoint.add_text(text='Robots are cool')
>>> powerpoint.add_text(text='Humans are cooler')
>>> # Export slides to images
>>> powerpoint.export_slides_to_images()
```

Keywords powerpoint, ppt, export, png, image, slides to image

Icon las la-file-powerpoint

export_to_pdf (*output_path=None*)

PowerPoint to PDF

Export PowerPoint presentation to PDF file

Parameters path – Output path where PDF file will be exported to. Default path is home directory with filename 'pdf_export.pdf'.

Extension output_path pdf

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add some slides with text
>>> powerpoint.add_text(text='Robots are cool')
>>> # Export to pdf
>>> powerpoint.export_to_pdf()
```

Keywords powerpoint, ppt, export, pdf

Icon las la-file-powerpoint

number_of_slides ()

Slide count

Returns the number of slides

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add some slides
```

(continues on next page)

(continued from previous page)

```
>>> powerpoint.add_slide()
>>> powerpoint.add_slide()
>>> # Show number of slides
>>> powerpoint.number_of_slides()
```

Keywords powerpoint, ppt, slide count, number of slides

Icon las la-file-powerpoint

quit ()

Close PowerPoint Application

Close PowerPoint

Parameters **index** – Index where the slide should be inserted. Default value is as final slide.

Parameter type Type of the slide to be added. Supports following types: blank, chart, text, title and picture.

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Close PowerPoint
>>> powerpoint.quit()
```

Keywords powerpoint, ppt, quit, exit

Icon las la-file-powerpoint

replace_text (placeholder_text, replacement_text)

Replace all occurrences of text in PowerPoint slides

Can be used for example to replace arbitrary placeholder value in a PowerPoint. For example when using a template slidedeck, using 'XXXX' as a placeholder. Take note that all strings are case sensitive.

Parameters

- **placeholder_text** (*string*) – Placeholder value (string) in the PowerPoint, this will be replaced, e.g. 'Company Name'
- **replacement_text** – Text (string) to replace the placeholder values with. It is recommended to make this unique in your PowerPoint to avoid wrongful replacement, e.g. 'XXXX_placeholder_XXX'

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add some slides with text
>>> powerpoint.add_text(text='Hello, my name is placeholder')
>>> # Change 'placeholder' to the word 'robot'
>>> powerpoint.replace_text(placeholder_text = 'placeholder', replacement_
↳ text = 'robot')
```

Keywords powerpoint, ppt, replace, placeholder

Icon las la-file-powerpoint

save()
Save PowerPoint
Save PowerPoint Slidedeck

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add a first slide
>>> powerpoint.add_slide()
>>> # Save the PowerPoint presentation
>>> powerpoint.save_as('AmazingPresentation.pptx')
```

Keywords powerpoint, ppt, save, save as, save powerpoint

Icon las la-file-powerpoint

save_as(output_path)
Save PowerPoint
Save PowerPoint Slidedeck

Parameters **file_path** – Save the PowerPoint presentation.

Extension **output_path** pptx

Example

```
>>> # Start PowerPoint
>>> powerpoint = PowerPoint()
>>> # Add a first slide
>>> powerpoint.add_slide()
>>> # Save the PowerPoint presentation
>>> powerpoint.save_as('AmazingPresentation.pptx')
```

Keywords powerpoint, ppt, save, save as, save powerpoint

Icon las la-file-powerpoint

6.18 Office 365

`automagica.activities.send_email_with_outlook365(client_id, client_secret, to_email, subject="", body="")`

Send email Office Outlook 365

Send email Office Outlook 365

Parameters

- **client_id** – Client id for office 365 account
- **client_secret** (*string*) – Client secret for office 365 account
- **to_email** (*string*) – E-mail to send to
- **subject** (*string, optional*) – Optional subject
- **body** – Optional body of the email


```
>>> # Send email to 'robot@automagica.com'
>>> send_email_with_outlook365('SampleClientID', 'SampleClientSecret',
↪ 'robot@automagica.com')
```

Keywords mail, office 365, outlook, email, e-mail

Icon las la-envelope

6.19 Salesforce

`automagica.activities.salesforce_api_call` (*action*, *key*, *parameters*={}, *method*='get', *data*={})

Salesforce API

Activity to make calls to Salesforce REST API.

Parameters

- **action** (*string*) – Action (the URL)
- **key** (*string*) – Authorisation key
- **parameters** (*string*) – URL params
- **method** – Method (get, post or patch)
- **data** (*string*) – Data for POST/PATCH.

Options method ['get', 'post', 'patch']

Returns

API data

Example

```
>>> spf_api_call('action', 'key', 'parameters')
Response
```

Keywords salesforce

Icon lab la-salesforce

6.20 E-mail (SMTP)

`automagica.activities.send_mail_smtp` (*smtp_host*, *smtp_user*, *smtp_password*, *to_address*, *subject*=", *message*=", *port*=587)

Mail with SMTP

This function lets you send emails with an e-mail address.

Parameters

- **smtp_host** (*string*) – The host of your e-mail account.
- **smtp_user** – The password of your e-mail account
- **smtp_password** (*string*) – The password of your e-mail account

- **to_address** (*string*) – The destination is the receiving mail address.
- **subject** (*string*, *optional*) – The subject
- **message** (*text*, *optional*) – The body of the mail
- **port** – The port variable is standard 587. In most cases this argument can be ignored, but in some cases it needs to be changed to 465.

```
>>> send_mail_smtp('robot@automagica.com', 'SampleUser', 'SamplePassword',  
↪ 'robotfriend@automagica.com')
```

Keywords mail, e-mail, email smtp

Icon las la-mail-bulk

6.21 Windows OS

`automagica.activities.find_window_title` (*searchterm*, *partial=True*)

Find window with specific title

Find a specific window based on the name, either a perfect match or a partial match.

Parameters

- **searchterm** (*string*) – Title to look for, e.g. 'Calculator' when looking for the Windows calculator
- **partial** – Option to look for titles partially, e.g. 'Edge' will result in finding 'Microsoft Edge' when partial is set to True. Default value is True

```
>>> # Make text file  
>>> testfile = make_text_file()  
>>> # Open the file  
>>> open_file(testfile)  
>>> #Find 'Notepad' in window titles  
>>> find_window_title('Notepad')  
'generated_text_file.txt - Notepad'
```

Keywords windows, user, password, remote desktop, remote, citrix, vnc, remotedesktop

Icon lab la-readme

`automagica.activities.start_remote_desktop` (*ip*, *username*, *password=None*, *desktop_width=1920*, *desktop_height=1080*)

Login to Windows Remote Desktop

Create a RDP and login to Windows Remote Desktop

Parameters

- **ip** (*string*) – IP address of remote desktop
- **username** (*string*) – Username
- **password** (*string*, *optional*) – Password
- **desktop_width** (*int*, *optional*) – Resolution (width) of desktop, standard value is 1920 (full HD)
- **desktop_height** – Resolution (height) of desktop, standard value is 1080 (full HD)

```
>>> start_remote_desktop('123.456.789.10', 'Administrator', 'SamplePassword')
```

Keywords windows, user, password, remote desktop, remote, citrix, vnc, remotedesktop

Icon las la-passport

automagica.activities.**close_remote_desktop**()

Stop Windows Remote Desktop

Stop Windows Remote Desktop

Example

```
>>> close_remote_desktop()
```

Keywords windows, user, password, remote desktop, remote, citrix, vnc, remotedesktop, stop

Icon las la-passport

automagica.activities.**set_user_password**(username, password)

Set Windows password

Sets the password for a Windows user.

Parameters

- **username** (*string*) – Username
- **password** – New password

```
>>> set_user_password('SampleUsername', 'SamplePassword')
```

Keywords windows, user, password, account

Icon las la-passport

automagica.activities.**validate_user_password**(username, password)

Check Windows password

Validates a Windows user password if it is correct

Parameters

- **username** (*string*) – Username
- **password** (*string*) – New password

Returns

True if the password is correct

Example

```
>>> validate_user_password('SampleUsername', 'SamplePassword')
False
```

Keywords windows, user, password, account

Icon las la-passport

`automagica.activities.lock_windows()`


Lock Windows

Locks Windows requiring login to continue.

Example

```
>>> lock_windows()
```

Keywords windows, user, password, account, lock, freeze, hibernate, sleep, lockscreen

Icon  las la-user-lock

`automagica.activities.is_logged_in()`

Check if Windows logged in

Checks if the current user is logged in and not on the lockscreen. Most automations do not work properly when the desktop is locked.


Returns

True if the user is logged in, False if not

Example

```
>>> is_logged_in()
True
```

Keywords windows, login, logged in, lockscreen, user, password, account, lock, freeze, hibernate, sleep

Icon  lar la-user

`automagica.activities.is_desktop_locked()`

Check if Windows is locked

Checks if the current user is locked out and on the lockscreen. Most automations do not work properly when the desktop is locked.

Returns

True when the lockscreen is active, False if not.

Example

```
>>> desktop_locked()
True
```

Keywords windows, login, logged in, lockscreen, user, password, account, lock, locked, freeze, hibernate, sleep

Icon  las la-user

`automagica.activities.get_username()`

Get Windows username

Get current logged in user's username

Example

```
>>> get_username()
'Automagica'
```

Keywords windows, login, logged in, lockscreen, user, password, account, lock, locked, freeze, hibernate, sleep

Icon las la-user

`automagica.activities.set_to_clipboard(text)`

Set clipboard

Set any text to the Windows clipboard.

Parameters `text` – Text to put in the clipboard

```
>>> # Create some sample text
>>> sample_text = 'A robots favourite food must be computer chips'
>>> # Set to clipboard
>>> set_to_clipboard(sample_text)
>>> # Print the clipboard to verify
>>> print( get_from_clipboard() )
```

Keywords copy, clipboard, clip board, ctrl c, ctrl v, paste

Icon las la-clipboard-check

`automagica.activities.get_from_clipboard()`

Get clipboard

Get the text currently in the Windows clipboard

Returns

Text currently in the clipboard

Example

```
>>> # Create some sample text
>>> sample_text = 'A robots favourite food must be computer chips'
>>> # Set to clipboard
>>> set_to_clipboard(sample_text)
>>> # Get the clipboard to verify
>>> get_from_clipboard()
'A robots favourite food must be computer chips'
```

Keywords copy, clipboard, clip board, ctrl c, ctrl v, paste

Icon las la-clipboard-list

`automagica.activities.clear_clipboard()`

Empty clipboard

Empty text from clipboard. Getting clipboard data after this should return in None

Example

```
>>> # Create some sample text
>>> sample_text = 'A robots favourite food must be computer chips'
>>> # Set to clipboard
>>> set_to_clipboard(sample_text)
>>> # Clear the clipboard
>>> clear_clipboard()
>>> # Get clipboard contents to verify
```

(continues on next page)

(continued from previous page)

```
>>> print( get_clipboard() )
None
```

Keywords copy, clipboard, clip board, ctrl c, ctrl v, paste

Icon las la-clipboard

`automagica.activities.run_vbs_script(script_path, parameters=[])`

Run VBScript

Run a VBScript file

Parameters

- **script_path** (*input_file*) – Path to the .vbs-file
- **parameters** – Additional arguments to pass to the VBScript

Example

```
>>> # Run a VBS script
>>> run_vbs_script('Samplescript.vbs')
```

Keywords vbs, VBScript

Icon las la-cogs

`automagica.activities.beep(frequency=1000, duration=500)`

Beep

Make a beeping sound. Make sure your volume is up and you have hardware connected.

Parameters

- **frequency** (*int, optional*) – Integer to specify frequency (Hz), default value is 1000 Hz
- **duration** (*int, optional*) – Integer to specify duration of beep in milliseconds (ms), default value is 500 ms.

Returns

Sound

Example

```
>>> beep()
```

Keywords beep, sound, noise, speaker, alert

Icon las la-volume-up

`automagica.activities.get_all_network_interface_names()`

Get all network interface names

Returns a list of all network interfaces of the current machine

Example

```
>>> get_all_network_interface_names()
['Microsoft Kernel Debug Network Adapter', 'Realtek Gaming GbE Family Controller',
↪ 'WAN Miniport (SSTP)']
```

Keywords networking, connection, list

Icon las la-ethernet

`automagica.activities.enable_network_interface(name)`

Enable network interface

Enables a network interface by its name.

Parameters **name** – Name of the network

```
>>> enable_network_interface('Realtek Gaming GbE Family Controller')
```

Keywords networking, connection, enable

Icon las la-ethernet

`automagica.activities.disable_network_interface(name)`

Disable network interface

Disables a network interface by its name.

Parameters **name** – Name of the network interface

```
>>> disable_network_interface('Realtek Gaming GbE Family Controller')
```

Keywords networking, connection, disable

Icon las la-ethernet

`automagica.activities.get_default_printer_name()`

Get default printer

Returns the name of the printer selected as default

Example

```
>>> get_default_printer_name()
'Epson MF742C/744C'
```

Keywords printing, get default printer name, default printer

Icon las la-print

`automagica.activities.set_default_printer(name)`

Set default printer

Set the default printer.

Parameters **name** – Printer name

```
>>> set_default_printer('Epson MF742C/744C')
```

Keywords printing, set default printer name, default printer

Icon las la-print

`automagica.activities.remove_printer(name)`

Remove printer

Removes a printer by its name

Parameters **name** – Printer name to remove

```
>>> remove_printer('Epson MF742C/744C')
```

Keywords printing, remove printer, printer

Icon las la-print

`automagica.activities.get_service_status(name)`

Get service status

Returns the status of a service on the machine

Parameters **name** – Name of service

```
>>> get_service_status('Windows Backup')
'stopped'
```

Keywords services, get service status, status

Icon las la-cog

`automagica.activities.start_service(name)`

Start a service

Starts a Windows service

Parameters **name** – Name of service

```
>>> start_service('Windows Backup')
```

Keywords services, start a service, start

Icon las la-cog

`automagica.activities.stop_service(name)`

Stop a service

Stops a Windows service

Parameters **name** – Name of service

```
>>> stop_service('Windows Backup')
```

Keywords services, stop a service, stop

Icon las la-cog

`automagica.activities.set_window_to_foreground(title)`

Set window to foreground

Sets a window to foreground by its title.

Parameters **name** – Name of service

```
>>> set_window_to_foreground('Notepad - Untitled')
```

Keywords window, foreground

Icon las la-window-restore

`automagica.activities.get_foreground_window_title()`

Get foreground window title

Retrieve the title of the current foreground window

Example

```
>>> get_foreground_window_title()
'IPython'
```

Keywords window, foreground, title

Icon las la-window-restore

`automagica.activities.close_window(title)`

Close window

Closes a window by its title

Parameters **title** – Title of window

```
>>> close_window('Untitled - Notepad')
```

Keywords window, close, title

Icon las la-window-restore

`automagica.activities.maximize_window(title)`

Maximize window

Maximizes a window by its title

Parameters **title** – Title of window

```
>>> maximize_window('Untitled - Notepad')
```

Keywords window, maximize, title

Icon las la-window-restore

`automagica.activities.restore_window(title)`

Restore window

Restore a window by its title

Parameters **title** – Title of window

```
>>> restore_window('Untitled - Notepad')
```

Keywords window, restore, title

Icon las la-window-restore

`automagica.activities.minimize_window(title)`

Minimize window

Minimizes a window by its title

Parameters **title** – Title of window

```
>>> minimize_window(title)
```

Keywords window, minimize, title

Icon las la-window-restore

`automagica.activities.resize_window(title, x, y, width, height)`

Resize window

Resize a window by its title

Parameters

- **title** (*string*) – Title of window
- **x** (*int*) – Starting x position
- **y** (*int*) – Starting y position
- **width** (*int*) – Width
- **height** – Height

```
>>> resize_window('Untitled - Notepad', 100, 200, 300, 400)
```

Keywords window, resize, title

Icon las la-window-restore

`automagica.activities.hide_window(title)`

Hide window

Hides a window from the user desktop by using it's title

Parameters **title** – Title of window

```
>>> hide_window('Untitled - Notepad')
```

Keywords window, hide, title

Icon las la-window-restore

6.22 Terminal

`automagica.activities.run_ssh_command(user, host, command)`

Run SSH command

Runs a command over SSH (Secure Shell)

Parameters

- **user** (*string*) – User
- **host** (*string*) – Host
- **command** – Command

```
>>> run_ssh_command('root', 'machine', 'ls -a')
'. .. .bashrc'
```

Keywords ssh, command

Icon las la-terminal

6.23 SNMP

`automagica.activities.snmp_get` (*target*, *oids*, *credentials*, *port=161*, *engine=None*, *context=None*)

SNMP Get

Retrieves data from an SNMP agent using SNMP (Simple Network Management Protocol)

Parameters

- **target** (*string*) – Target
- **oids** (*string*) – oids
- **credentials** (*string*) – credentials
- **port** (*int*, *optional*) – Port (default 161)
- **engine** (*string*, *optional*) – Engine (default none)
- **context** – Context (default none)

```
>>> snmp_get()
```

Keywords snmp, simple network management protocol, protocols, get

Icon las la-ethernet

6.24 Active Directory

class `automagica.activities.ActiveDirectory` (*ldap_server=None*, *username=None*, *password=None*)

get_object_by_distinguished_name (*distinguished_name*)

Get AD object by name

Interface to Windows Active Directory through ADSI

Parameters **distinguished_name** – Name

```
>>> ad = ActiveDirectory()
>>> ad.get_object_by_distinguished_name('SampleDN')
```

Keywords AD, active directory, activedirectory

Icon las la-audio-description

6.25 Utilities

`automagica.activities.home_path (filename=None)`

Get user home path

Returns the current user's home path

Parameters `subdir` – Optional filename to add to the path. Can also be a subdirectory

Returns

Path to the current user's home folder

Example

```
>>> # Home_path without arguments will return the home path
>>> print( home_path() )
>>> # When looking for a file in the home path, we can specify it
>>> # First make a sample text file
>>> make_text_file()
>>> # Refer to it
>>> home_path('generated_text_file.txt')
'C:\Users\\generated_text_file.txt'
```

Keywords home, home path, homopath, home directory, homedir

Icon las la-home

`automagica.activities.desktop_path (filename=None)`

Get desktop path

Returns the current user's desktop path

Parameters `filename` (*string, optional*) – Optional filename to add to the path. Can also be a subdirectory

Returns

Path to the current user's desktop folder

Example

```
>>> # Desktop_path without arguments will return the home path
>>> print( desktop_path() )
>>> # When looking for a file on the desktop, we can specify it
>>> # First make a sample text file
>>> make_text_file()
>>> # Refer to it
>>> desktop_path('generated_text_file.txt')
'C:\Users\\Desktop\generated_text_file.txt'
```

Keywords desktop, desktop path, desktoppath, desktop directory, desktopdir

Icon lar la-desktop

`automagica.activities.downloads_path()`

Get downloads path

Returns the current user's default download path

Returns

Path to the current user's downloads folder

Example

```
>>> # Find downloads path
>>> downloads_path()
```

Keywords download, download path, downloadpath, download directory, download dir, downloadaddir

Icon lar la-download

`automagica.activities.open_file(input_path)`

Open file

Opens file with default programs

Parameters `input_path` (`input_file`) – Path to file.

Returns

Path to file

Example

```
>>> # Make text file
>>> testfile = make_text_file()
>>> # Open the file
>>> open_file(testfile)
```

Keywords file, open, open file, show, reveal, explorer, run, start

Icon lar la-file

`automagica.activities.set_wallpaper(image_path)`

Set wallpaper

Set Windows desktop wallpaper with the the specified image

Parameters `image_path` – Path to the image. This image will be set as desktop wallpaper

```
>>> # Caution: this example will change your wallpaper
>>> # Take a screenshot of current screen
>>> screenshot = take_screenshot()
>>> # Flip it hozirontally for fun
>>> mirror_image_horizontally(screenshot)
>>> # Set flipped image as wallpaper
>>> set_wallpaper(screenshot)
```

Keywords desktop, desktop path, desktoppath, desktop directory, desktopdir, wallpaper, wall paper, wall

Icon las la-desktop

`automagica.activities.download_file_from_url(url, output_path=None)`

Download file from a URL

Download file from a URL

Parameters

- **url** (*string*) – Source URL to download file from
- **output_path** (*output_dir, optional*) – Target path, default to homedir with name ‘_download_’ + random addition

Returns

Target path as string

Example

```
>>> # Download robot picture from the wikipedia robot page
>>> picture_url = 'https://upload.wikimedia.org/wikipedia/commons/thumb/6/6c/
↳Atlas_from_boston_dynamics.jpg/220px-Atlas_from_boston_dynamics.jpg'
>>> download_file_from_url(url = picture_url, output_path = 'robot.jpg')
'C:\Users\<username>\robot.jpg'
```

Keywords download, download url, save, request

Icon las la-cloud-download-alt

6.26 System

`automagica.activities.rename_file(input_path, output_name=None)`

Rename a file

This activity will rename a file. If the the desired name already exists in the folder file will not be renamed. Make sure to add the exstention to specify filetype.

Parameters

- **input_path** (*input_file*) – Full path to file that will be renamed
- **output_name** (*string, optional*) – New name of the file e.g. ‘newfile.txt’. By default file will be renamed to original folder name with ‘_renamed’ added to the folder name.

Returns

Path to renamed file as a string. None if folder could not be renamed.

Example

```
>>> # Make new text file in home directory
>>> text_file = make_text_file()
>>> # Rename the file
>>> rename_file(text_file, output_name='brand_new_filename.txt')
'C:\Users\<username>\brand_new_filename.txt'
```

Keywords file, rename, rename file, organise file, files, file manipulation, explorer, nautilus

Icon las la-file-contract

`automagica.activities.move_file(input_path, output_path=None)`

Move a file

If the new location already contains a file with the same name.

Parameters

- **input_path** (*input_file*) – Full path to the file that will be moved
- **output_path** (*output_dir*) – Path to the folder where file will be moved to, defaults to input_path with ‘_moved’ added

Returns

Path to renamed file as a string. None if folder could not be moved.

Example

```
>>> # Make new text file in home directory
>>> text_file = make_text_file()
>>> # Make a folder to move the file to
>>> new_folder = create_folder()
>>> # Move text file to the folder
>>> move_file(text_file, new_folder)
```

Keywords file, move, move file, organise file, files, file manipulation, explorer, nautilus

Icon `las la-file-export`

`automagica.activities.remove_file(path)`

Remove a file

Remove a file

Parameters **path** (*input_file*) – Full path to the file that will be deleted.

Returns

Path to removed file as a string.

Example

```
>>> # Make new text file in home directory
>>> text_file = make_text_file()
>>> # Remove the file
>>> remove_file(text_file)
```

Keywords file, delete, erase, delete file, organise file, files, file manipulation, explorer, nautilus

Icon `las la-trash`

`automagica.activities.file_exists(path)`

Check if file exists

This function checks whether the file with the given path exists.

Parameters **path** (*input_file*) – Full path to the file to check.

return: True or False (boolean)

Example

```
>>> # Make new text file in home directory
>>> text_file = make_text_file()
>>> # Check if file exists
>>> file_exists(text_file)
True
```

Keywords file, exists, files, file manipulation, explorer, nautilus

Icon las la-tasks

`automagica.activities.wait_file_exists(path, timeout=60)`

Wait until a file exists.

Not that this activity is blocking and will keep the system waiting.

Parameters

- **path** (*input_file*) – Full path to file.
- **timeout** – Maximum time in seconds to wait before continuing. Default value is 60 seconds.

```
>>> # Make new text file in home directory
>>> text_file = make_text_file()
>>> # Wait untile file exists # Should pass immediatly
>>> wait_file_exists(text_file)
```

Keywords file, wait, wait till exists, files, file manipulation, explorer, nautilus

Icon las la-list-alt

`automagica.activities.write_list_to_file(list_to_write, file_path)`

List to .txt

Writes a list to a text (.txt) file. Every element of the entered list is written on a new line of the text file.

Parameters

- **list_to_write** – List to write to .txt file
- **file_path** (*output_file*) – Path to the text-file.

Extension `file_path` txt

Example

```
>>> # Make a list to write
>>> robot_names = ['WALL-E', 'Terminator', 'R2D2']
>>> # Create a new text file
>>> text_file = make_text_file()
>>> write_list_to_file(robot_names, text_file)
>>> # Open the file for illustration
>>> open_file(text_file)
```

Keywords list, text, txt, list to file, write list, write

Icon las la-list

`automagica.activities.read_list_from_txt(input_path)`

Read list from .txt file

This activity reads the content of a .txt file to a list and returns that list. Every new line from the .txt file becomes a new element of the list. The activity will not work if the entered path is not attached to a .txt file.

Parameters `input_path(input_file)` – Path to the .txt file

Extension `input_path` txt

Returns

List with contents of specified .txt file

Example

```
>>> # Make a list to write
>>> robot_names = ['WALL-E', 'Terminator', 'R2D2']
>>> # Create a new text file
>>> text_file = make_text_file()
>>> write_list_to_file(robot_names, text_file)
>>> # Read list from file
>>> read_list_from_txt(text_file)
['WALL-E', 'Terminator', 'R2D2']
```

Keywords list, text, txt, list to file, write list, read, read txt, read text

Icon las la-th-list

`automagica.activities.read_from_txt(file_path)`

Read .txt file

This activity reads a .txt file and returns the content

Parameters `input_path(input_file)` – Path to the .txt file

Extension `input_path` txt

Returns

Contents of specified .txt file

Example

```
>>> # Create a new text file
>>> text_file = make_text_file()
>>> # Read list from file
>>> read_from_txt(text_file)
'Sample text'
```

Keywords list, text, txt, list to file, read, read txt, read text

Icon las la-th-list

`automagica.activities.append_line(text, file_path)`

Append to .txt

Append a text line to a file and creates the file if it does not exist yet.

Parameters

- **text** (*string*) – The text line to write to the end of the file

- **file_path** (*input_file*) – Path to the file to write to

Extension file_path txt

Example

```
>>> # Create a new text file
>>> text_file = make_text_file()
>>> # Append a few lines to the file
>>> append_line('Line 1', text_file)
>>> append_line('Line 2', text_file)
>>> append_line('Line 3', text_file)
>>> # Open the file for illustration
>>> open_file(text_file)
```

Keywords list, text, txt, list to file, write list, read, write txt, append text, append line, append, add to file, add

Icon las la-tasks

automagica.activities.**make_text_file** (*text='Sample text', output_path=None*)

Make text file

Initialize text file

Parameters

- **text** (*string, optional*) – The text line to write to the end of the file. Default text is 'Sample text'
- **output_path** (*output_file, optional*) – Ouput path. Will write to home directory on default

Extension output_path txt

Returns

Path as string

Example

```
>>> # Create a new text file
>>> text_file = make_text_file()
C:\Users\<<username>\generated_text_file.txt'
```

Keywords make text file, text_file, testfile, exampel file, make file, make, new file, new text_file, txt, new txt

Icon las la-file-alt

automagica.activities.**read_text_file_to_list** (*file_path*)

Read .txt file with newlines to list

Read a text file to a Python list-object

parameter file_path Path to the text file which should be read to a list

type file_path input_file

extension file_path txt

return List with the lines in the text file

Example

```
>>> # Create a new text file
>>> text_file = make_text_file(text="First line!")
```

Second line!")

```
>>> # Read the text file to a list
>>> lines = read_text_file_to_list(text_file)
>>> lines
['First line!', 'Second line!']
```

Keywords read text file, list, reading text file

Icon las la-copy

`automagica.activities.copy_file(input_path, output_path=None)`

Copy a file

Copies a file from one place to another. If the new location already contains a file with the same name, a random 4 character uid is added to the name.

Parameters

- **input_path** (*input_file*) – Full path to the source location of the file
- **output_path** (*output_dir, optional*) – Optional full path to the destination location of the folder. If not specified file will be copied to the same location with a random 8 character uid is added to the name.

Returns

New path as string

Example

```
>>> # Create a new text file
>>> text_file = make_text_file()
>>> # Copy the text file
>>> copy_file(text_file)
C:\Users\<username>\generated_text_file.txt'
```

Keywords make text file, text_file, testfile, example file, make file, make, new file, new text_file, txt, new txt

Icon las la-copy

`automagica.activities.get_file_extension(file_path)`

Get file extension

Get extension of a file

Parameters **file_path** (*input_file*) – Path to file to get extension from

Returns

String with extension, e.g. '.txt'

Example

```
>>> # Create a new text file
>>> text_file = make_text_file()
>>> # Get file extension of this text file
>>> get_file_extension(text_file)
'.txt'
```

Keywords file, extension, file extension, details

Icon las la-info

`automagica.activities.send_to_printer` (*file_path*)

Print

Send file to default printer to priner. This activity sends a file to the printer. Make sure to have a default printer set up.

Parameters `file_path` – Path to the file to print, should be a printable file

```
>>> # Caution as this example could result in a print from default printer
>>> # Create a new text file
>>> text_file = make_text_file(text = 'What does a robot do at lunch? Take a_
↳megabyte!')
>>> # Print the text file
>>> send_to_printer(text_file)
```

Keywords print, printer, printing, ink, export

Icon las la-print

6.27 PDF

`automagica.activities.read_text_from_pdf` (*file_path*)

Text from PDF

Extracts the text from a PDF. This activity reads text from a pdf file. Can only read PDF files that contain a text layer.

Parameters `file_path` (*input_file*) – Path to the PDF (either relative or absolute)

Extension `file_path` pdf

Returns

The text from the PDF

Example

```
>>> # Caution, for this example to work a .pdf example file will be downloaded_
↳from automagica.com FTP
>>> example_pdf = download_file_from_url('http://automagica.com/examples/example_
↳document.pdf')
>>> # Open example pdf for illustration
>>> open_file(example_pdf)
>>> # Read the text
>>> read_text_from_pdf(example_pdf)
```

Keywords PDF, read, text, extract text, PDF file

Icon las la-glasses

`automagica.activities.join_pdf_files` (*first_file_path*, *second_file_path*, *third_file_path=None*,
output_path=None)

Merge PDF

Merges multiple PDFs into a single file

Parameters

- **first_file_path** (*input_file*) – Path to first PDF file
- **second_file_path** (*input_file*, *optional*) – Path to second PDF file
- **third_file_path** (*input_file*, *optional*) – Path to third PDF file, optional
- **output_path** (*output_file*, *optional*) – Full path where joined pdf files can be written. If no path is given will write to home dir as ‘merged_pdf.pdf’

Extension **first_file_path** pdf

Extension **second_file_path** pdf

Extension **third_file_path** pdf

Extension **output_path** pdf

Returns

Output path as string

Example

```
>>> # Caution, for this example to work a .pdf example file will be downloaded,
↳from automagica.com FTP
>>> example_pdf = download_file_from_url('http://automagica.com/examples/example_
↳document.pdf')
>>> # Join the PDF file with itself for illustration, could also be different,
↳files
>>> merged_pdf = join_pdf_files(example_pdf, example_pdf)
>>> # Open resulting PDF file for illustration
>>> open_file(merged_pdf)
```

Keywords PDF, read, text, extract text, PDF file, join PDF, join, merge, merge PDF

Icon las la-object-ungroup

`automagica.activities.extract_page_range_from_pdf` (*file_path*, *start_page*, *end_page*,
output_path=None)

Extract page from PDF

Extracts a particular range of a PDF to a separate file.

Parameters

- **file_path** (*input_file*) – Path to the PDF (either relative or absolute)
- **start_page** (*int*) – Page number to start from, with 0 being the first page
- **end_page** (*int*) – Page number to end with, with 0 being the first page
- **output_path** (*output_file*, *optional*) – Output path, if no path is provided same path as input will be used with ‘extracted’ added to the name

Extension **file_path** pdf

Extension `output_path` pdf

Example

```
>>> # Caution, for this example to work a .pdf example file will be downloaded_
↳from automagica.com FTP
>>> example_pdf = download_file_from_url('http://automagica.com/examples/example_
↳document.pdf')
>>> # Join the PDF file three times to create multi page
>>> multi_page_pdf_example = join_pdf_files(example_pdf, example_pdf, example_pdf)
>>> # Extract some pages from it
>>> new_file = extract_page_range_from_pdf(multi_page_pdf_example, 1, 2 )
>>> # Open resulting PDF file for illustration
>>> open_file(new_file)
```

Keywords PDF, read, extract text, PDF file, extract PDF, join, cut, cut PDF, extract pages, extract from pdf, select page, page

Icon las la-cut

`automagica.activities.extract_images_from_pdf` (*file_path*)

Extract images from PDF

Save a specific page from a PDF as an image

Parameters `file_path` – Full path to store extracted images

```
>>> # Caution, for this example to work a .pdf example file will be downloaded_
↳from automagica.com FTP
>>> example_pdf = download_file_from_url('http://automagica.com/examples/example_
↳document.pdf')
>>> # Extract the images
>>> extract_images_from_pdf(example_pdf)
```

Keywords PDF, extract images, images, extract text, PDF file, image

Icon las la-icons

`automagica.activities.apply_watermark_to_pdf` (*file_path*, *watermark_path*, *out-*
put_path="")

Watermark a PDF

Watermark a PDF

Parameters

- **file_path** (*input_file*) – Filepath to the document that will be watermarked. Should be pdf file.
- **watermark_path** (*input_file*) – Filepath to the watermark. Should be pdf file.
- **output_path** (*output_file*, *optional*) – Path to save watermarked PDF. If no path is provided same path as input will be used with ‘watermarked’ added to the name

Extension `file_path` pdf

Extension `watermark_path` pdf

Extension `output_path` pdf

Returns

Output path as a string

Example

```
>>> # Caution, for this example to work a .pdf example file will be downloaded,
↳from automagica.com FTP
>>> example_pdf = download_file_from_url('http://automagica.com/examples/example_
↳document.pdf')
>>> # Download the watermark
>>> example_watermark = download_file_from_url('http://automagica.com/examples/
↳approved_stamp.pdf')
>>> # Apply the watermark
>>> watermarked_file = apply_watermark_to_pdf(example_pdf, example_watermark)
>>> # Open the file for illustration
>>> open_file(watermarked_file)
```

Keywords PDF, extract images, images, extract text, PDF file, image

Icon las la-stamp

6.28 System Monitoring

`automagica.activities.get_cpu_load(measure_time=1)`

CPU load

Get average CPU load for all cores.

Parameters `measure_time` (*int, optional*) – Time (seconds) to measure load. Standard `measure_time` is 1 second.

Returns

Displayed load is an average over `measured_time`.

Example

```
>>> get_cpu_load()
10.1
```

Keywords cpu, load, cpuload

Icon las la-microchip

`automagica.activities.get_number_of_cpu(logical=True)`

Count CPU

Get the number of CPU's in the current system.

Parameters `logical` (*bool, optional*) – Determines if only logical units are added to the count, default value is True.

Returns

Number of CPU Integer

Example

```
>>> get_number_of_cpu()
2
```

Keywords cpu, count, number of cpu

Icon las la-calculator

`automagica.activities.get_cpu_frequency()`
CPU frequency

Get frequency at which CPU currently operates.

Returns

minimum and maximum frequency

Example

```
>>> get_cpu_frequency()
scpufreq(current=3600.0, min=0.0, max=3600.0)
```

Keywords cpu, load, cpu frequency

Icon las la-wave-square

`automagica.activities.get_cpu_stats()`
CPU Stats

Get CPU statistics

Returns

Number of CTX switches, interrupts, soft-interrupts and systemcalls.

Example

```
>>> get_cpu_stats()
scpustats(ctx_switches=735743826, interrupts=1540483897, soft_interrupts=0,
↳ syscalls=2060595131)
```

Keywords cpu, load, cpu frequency, stats, cpu statistics

Icon las la-server

`automagica.activities.get_memory_stats(mem_type='swap')`
Memory statistics

Get memory statistics

Parameters `mem_type` – Choose `mem_type = 'virtual'` for virtual memory, and `mem_type = 'swap'` for swap memory (standard).

Options `mem_type` ['virtual', 'swap']

Returns

Total, used, free and percentage in use.

Example


```
>>> get_memory_stats()
sswap(total=24640016384, used=18120818688, free=6519197696, percent=73.5, sin=0,
↪sout=0)
```

Keywords memory, statistics, usage, ram

Icon las la-memory

automagica.activities.get_disk_stats()

Disk stats

Get disk statistics of main disk

Returns

Total, used, free and percentage in use.

Example

```
>>> get_disk_stats()
sdiskusage(total=999559262208, used=748696350720, free=250862911488, percent=74.9)
```

Keywords disk usage, disk stats, disk, harddisk, space

Icon las la-save

automagica.activities.get_disk_partitions()

Partition info

Get disk partition info

Returns

tuple with info for every partition.

Example

```
>>> get_disk_partitions()
[sdiskpart(device='C:\', mountpoint='C:\', fstype='NTFS', opts='rw,fixed')]
```

Keywords disk usage, disk stats, disk, harddisk, space

Icon las la-save

automagica.activities.get_boot_time()

Boot time

Get most recent boot time

Returns

time PC was booted in seconds after the epoch.

Example

```
>>> get_boot_time()
123456789.0
```

Keywords boot, boot time, boottime, startup, timer

Icon lar la-clock

`automagica.activities.get_time_since_last_boot()`

Uptime

Get uptime since last boot

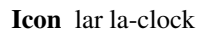
Returns

time since last boot in seconds.

Example

```
>>> get_time_since_last_boot()
1337.0
```

Keywords boot, boot time, boottime, startup, timer

Icon 

6.29 Image Processing

`automagica.activities.show_image(file_path)`

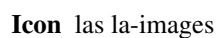
Show image

Displays an image specified by the path variable on the default imaging program.

Parameters `file_path` – Path to image

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, show image, reveal, open image, open

Icon 

`automagica.activities.rotate_image(file_path, angle=90)`

Rotate image

Rotate an image

Parameters

- **file_path** (`input_file`) – Path to image
- **angle** – Degrees to rotate image. Note that angles other than 90, 180, 270, 360 can resize the picture.

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Rotate the image
>>> rotate_image(testimage)
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, rotate image, 90 degrees, image manipulation, photoshop, paint

Icon 

`automagica.activities.resize_image(file_path, size)`

Resize image

Resizes the image specified by the path variable.

Parameters

- **file_path** (*input_file*) – Path to the image
- **size** – Tuple with the width and height in pixels. E.g. (300, 400) gives the image a width of 300 pixels and a height of 400 pixels.

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Resize the image
>>> resize_image(testimage, size=(100,100))
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, resize image, resize, size, image manipulation, photoshop, paint

Icon `las la-expand-arrows-alt`

`automagica.activities.get_image_width(file_path)`

Get image width

Get with of image

Parameters **file_path** – Path to image

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # get image height
>>> get_image_width(testimage)
1000
```

Keywords image, height, width, image height, image width

Icon `las la-expand-arrows-alt`

`automagica.activities.get_image_height(file_path)`

Get image height

Get height of image

Parameters **file_path** (*input_file*) – Path to image

Returns

Height of image

Example

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # get image height
>>> get_image_height(testimage)
1000
```

Keywords image, height, width, image height, image width

Icon las la-arrows-alt-v

`automagica.activities.crop_image(file_path, box=None)`

Crop image

Crops the image specified by path to a region determined by the box variable.

Parameters

- **file_path** (*input_file*) – Path to image
- **box** – A tuple that defines the left, upper, right and lower pixel coordinate e.g.: (left, upper, right, lower)

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Crop the image
>>> crop_image(testimage, box = (10,10,100,100))
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, crop, crop image

Icon las la-crop

`automagica.activities.mirror_image_horizontally(file_path)`

Mirror image horizontally

Mirrors an image with a given path horizontally from left to right.

Parameters **file_path** – Path to image

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Mirror image horizontally
>>> mirror_image_horizontally(testimage)
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, flip, flip image, mirror, mirror image, horizon, horizontally

Icon las la-caret-up

`automagica.activities.mirror_image_vertically(file_path)`

Mirror image vertically

Mirrors an image with a given path vertically from top to bottom.

Parameters **file_path** – Path to image

```
>>> # Take screenshot of current screen to use as test image
>>> testimage = take_screenshot()
>>> # Mirror image vertically
>>> mirror_image_vertically(testimage)
>>> # Show the image
>>> show_image(testimage)
```

Keywords image, flip, flip image, mirror, mirror image, vertical, vertically

Icon las la-caret-right

6.30 Process

`automagica.activities.run_manual(task)`

Windows run

Use Windows Run to boot a process Note this uses keyboard inputs which means this process can be disrupted by interfering inputs

Parameters `task` – Name of the task to run e.g. ‘mspaint.exe’

```
>>> # Open paint with Windows run
>>> run_manual('mspaint.exe')
>>> # Open home directory with Windows run
>>> run_manual(home_path())
```

Keywords run, open, task, win r, windows run, shell, cmd

Icon las la-cog

`automagica.activities.run(process)`

Run process

Use subprocess to open a windows process

Parameters `process` – Process to open e.g: ‘calc.exe’, ‘notepad.exe’, ‘control.exe’, ‘mspaint.exe’.

```
>>> # Open paint with Windows run
>>> run('mspaint.exe')
```

Keywords run, open, task, win r, windows run, shell, cmd

Icon las la-play

`automagica.activities.is_process_running(name)`

Check if process is running

Check if process is running. Validates if given process name (name) is currently running on the system.

Parameters `name` (*string*) – Name of process

Returns

Boolean

Example

```
>>> # Open paint with Windows run
>>> run('mspaint.exe')
>>> # Check if paint is running
>>> is_process_running('mspaint.exe')
True
```

Keywords run, open, task, win r, windows run, shell, cmd

Icon las la-cogs

`automagica.activities.get_running_processes()`

Get running processes

Get names of unique processes currently running on the system.

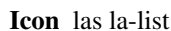
Returns

List of unique running processes

Example

```
>>> # Show all running processes
>>> get_running_processes()
['cmd.exe', 'chrome.exe', ... ]
```

Keywords process, processes, list processes, running, running processes

Icon 

`automagica.activities.kill_process(name=None)`

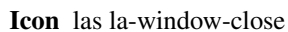
Kill process

Kills a process forcefully

Parameters `name` – Name of the process

```
>>> # Open paint with Windows run
>>> run('mspaint.exe')
>>> # Force paint to close
>>> kill_process('mspaint.exe')
```

Keywords run, open, task, win r, windows run, shell, cmd, kill, stop, kill process, stop process, quit, exit

Icon 

6.31 Optical Character Recognition (OCR)

`automagica.activities.extract_text_ocr(file_path=None)`

Get text with OCR

This activity extracts all text from the current screen or an image if a path is specified.

Parameters `file_path(input_file)` – Path to image from where text will be extracted. If no path is specified a screenshot of current screen will be used.

Returns

String with all text from current screen

Example

```
>>> # Make a text file with some text to recognize
>>> testfile = make_text_file(text='OCR Example')
>>> # Open the text file
>>> open_file(testfile)
>>> # Find the text with OCR
>>> extracted_text = find_text_on_screen_ocr(text='OCR Example')
>>> # Check if the extracted_text contains the original word
```

(continues on next page)

(continued from previous page)

```
>>> 'OCR Example' in extracted_text
True
```

Keywords OCR, vision, AI, screen, citrix, read, optical character recognition

Icon lab la-readme

`automagica.activities.find_text_on_screen_ocr(text, criteria=None)`

Find text on screen with OCR

This activity finds position (coordinates) of specified text on the current screen using OCR.

Parameters

- **text** (*string*) – Text to find. Only exact matches are returned.
- **criteria** – Criteria to select on if multiple matches are found. If no criteria is specified all matches will be returned. Options are 'first', which returns the first match closest to the upper left corner, 'last' returns the last match closest to the lower right corner, random selects a random match.

Options criteria ['first', 'last', 'random']

Returns

Dictionary or list of dictionaries with matches with following elements: 'h' height in pixels, 'text' the matched text, 'w' the width in pixels, 'x' absolute x-coördinate, 'y' absolute y-coördinate. Returns nothing if no matches are found

Example

```
>>> # Make a text file with some text to recognize
>>> testfile = make_text_file(text='OCR Example')
>>> # Open the text file
>>> open_file(testfile)
>>> # Find the text with OCR
>>> find_text_on_screen_ocr(text='OCR Example')
```

Keywords OCR, vision, AI, screen, citrix, read, optical character recognition

Icon las la-glasses

`automagica.activities.click_on_text_ocr(text, delay=1)`

Click on text with OCR

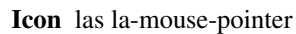
This activity clicks on position (coordinates) of specified text on the current screen using OCR.

Parameters

- **text** (*string*) – Text to find. Only exact matches are returned.
- **delay** – Delay before clicking in seconds

```
>>> # Make a text file with some text to recognize
>>> testfile = make_text_file(text='OCR Example')
>>> # Open the text file
>>> open_file(testfile)
>>> # Find the text with OCR and click on it
>>> click_on_text_ocr(text='OCR Example')
```

Keywords OCR, vision, AI, screen, citrix, read, optical character recognition, click

Icon 

`automagica.activities.double_click_on_text_ocr(text, delay=1)`

Double click on text with OCR

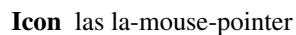
This activity double clicks on position (coordinates) of specified text on the current screen using OCR.

Parameters

- **text** (*string*) – Text to find. Only exact matches are returned.
- **delay** – Delay before clicking in seconds

```
>>> # Make a text_file with some text to recognize
>>> testfile = make_text_file(text='OCR Example')
>>> # Open the text file
>>> open_file(testfile)
>>> # Find the text with OCR and double click on it
>>> double_click_on_text_ocr(text='OCR Example')
```

Keywords OCR, vision, AI, screen, citrix, read, optical character recognition, click, double click

Icon 

`automagica.activities.right_click_on_text_ocr(text, delay=1)`

Right click on text with OCR

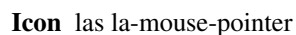
This activity Right clicks on position (coordinates) of specified text on the current screen using OCR.

Parameters

- **text** (*string*) – Text to find. Only exact matches are returned.
- **delay** – Delay before clicking in seconds

```
>>> # Make a text file with some text to recognize
>>> testfile = make_text_file(text='OCR Example')
>>> # Open the text file
>>> open_file(testfile)
>>> # Find the text with OCR and right click on it
>>> right_click_on_text_ocr(text='OCR Example')
```

Keywords OCR, vision, AI, screen, citrix, read, optical character recognition, click, right click

Icon 

6.32 Alternative Frameworks

`automagica.activities.execute_uipath_process(project_file_path, arguments=None, uirobot_exe_path=None)`

Execute a UiPath process

This activity allows you to execute a process designed with the UiPath Studio. All console output from the Write Line activity (<https://docs.uipath.com/activities/docs/write-line>) will be printed as output.

Parameters

- **project_file_path** (*input_file*) – path to the project file (as created within the UiPath Studio)
- **arguments** (*string, optional*) – dictionary with input arguments/parameters for the process to use in UiPath (optional)
- **uirobot_exe_path** (*input_file, optional*) – path to UiPath's UiRobot.exe (optional)

Extension **uirobot_exe_path** exe

Example

```
>>> # Run a UiPath process
>>> arguments = {'firstname': 'John', 'lastname': 'Doe'}
>>> execute_uipath_process(r"C:\Processes UiPath\my_process.xaml",
↳arguments=arguments)
Completed UiPath process "C:\Processes UiPath\my_process.xaml"
```

Keywords RPA, UiPath, Studio, robot, orchestrator, xaml, ui path

Icon las la-robot

`automagica.activities.run_autoit_script` (*script_path*, *arguments=None*, *autoit_exe_path=None*)

Execute a AutoIt script

This activity allows you to run an AutoIt script. If you use the ConsoleWrite function (<https://www.autoitscript.com/autoit3/docs/functions/ConsoleWrite.htm>), the output will be presented to you.

Parameters

- **script_path** (*input_file*) – path to the ‘.au3’ script file
- **arguments** (*string, optional*) – string with input arguments/parameters for the script (optional)
- **autoit_exe_path** (*input_file, optional*) – path to AutoIt.exe (optional)

Extension **script_path** au3

Extension **autoit_exe_path** exe

Example

```
>>> # Run an AutoIt script
>>> arguments = 'John'
>>> run_autoit_script(r"C:\AutoIt\Scripts\MyScript.au3", arguments=arguments) #
↳Point this to your AutoIt Script
Completed AutoIt script "C:\AutoIt\Scripts\MyScript.au3"
```

Keywords RPA, AutoIt, au3, au

Icon las la-robot

`automagica.activities.execute_robotframework_test` (*test_case_path*, *variables=None*)

Execute a Robot Framework test case

This activity allows you to run a Robot Framework test case. Console output of the test case will be printed.

Parameters

- **test_case_path** (*input_file*) – path to the ‘.robot’ test case file

- **variables** – dictionary with variable declarations

Extension `test_case_path` robot

```
>>> # Run an Robot Framework test case
>>> variables = {'FIRSTNAME': 'John', 'LASTNAME': 'Doe'}
>>> execute_robotframework_test(r"C:\Test Cases\my_test_case.robot",
↪ variables=variables) # Point this to your Robot Framework test case
Completed Robot Framework test case "C:\Test Cases\my_test_case.robot"
```

Keywords RPA, robot framework, robotframework, robot

Icon las la-robot

`automagica.activities.run_blueprism_process` (*process_name*, *username*=", *password*=", *sso*=False, *inputs*=None, *automatec_exe_path*=None)

Run a Blue Prism process

This activity allows you to run a Blue Prism process.

Parameters

- **process_name** (*string*) – name of the process in Blue Prism
- **username** (*string*, *optional*) – Blue Prism username
- **password** (*string*, *optional*) – Blue Prism password
- **sso** (*bool*, *optional*) – Run as single-sign on user with Blue Prism
- **inputs** (*string*, *optional*) – dictionary with inputs declarations (optional)
- **automatec_exe_path** (*input_file*) – path to Blue Prism's AutomateC.exe (optional)

Extension `automatec_exe_path` exe

Example

```
>>> # Run a Blue Prism process
>>> inputs = {'firstname': 'John', 'lastname': 'Doe'}
>>> run_blueprism_process("My Example Process", username="user", password=
↪ "password", inputs=inputs)
Completed Blue Prism process "My Example Process"
```

Keywords RPA, blueprism, blue prism, robot

Icon las la-robot

`automagica.activities.run_automationanywhere_task` (*task_file_path*, *aaplayer_exe_path*=None)

Run an Automation Anywhere task

This activity allows you to run an Automation Anywhere task.

Parameters

- **task_file_path** (*input_file*) – path to the task file of Automation Anywhere
- **aaplayer_exe_path** (*input_file*) – path to the AAPlayer.exe (optional)

Extension `aaplayer_exe_path` exe

Example

```
>>> # Run an Automation Anywhere task
>>> run_automationanywhere_task(r"C:\AutomationAnywhereTasks\MyTask.atmx")
Completed Automation Anywhere task "C:\AutomationAnywhereTasks\MyTask.atmx"
```

Keywords RPA, automation anywhere, aa, robot

Icon las la-robot

6.33 General

`automagica.activities.raise_exception` (*message*='Exception', *exception*=<class 'Exception'>)

Raise exception

Raises an exception

Parameters

- **message** – Message
- **exception** (*exception*, *optional*) – Exception to raise

Type string, optional

Keywords sap, sap gui, sap client

Icon las la-exclamation

6.34 SAP GUI

6.35 Portal

`automagica.activities.create_new_job_in_portal` (*process_name*, *process_version_id*=None, *priority*=0, *parameters*=None)

Create a new job in the Automagica Portal

This activity creates a new job in the Automagica Portal for a given process. The bot performing this activity needs to be in the same team as the process it creates a job for.

Parameters

- **process_name** (*string*) – name of the process
- **process_version_id** (*string*, *optional*) – id of a specific version of the process, if not provided it will use the latest version (optional)
- **priority** (*int*, *optional*) – priority level of the process. higher priority levels are performed first. (optional)
- **parameters** – parameters for the process (optional)

```
>>> # Create a job in the Automagica Portal
>>> create_new_job_in_portal('My process')
Job 1234567890 created
```

Keywords queueing, process, job, create job, new job

Icon las la-robot

`automagica.activities.get_credential_from_portal(credential_name)`

Get a credential from the Automagica Portal

This activity retrieves a credential from the Automagica Portal.

Parameters `credential_name` (*string*) – name of the credential

Returns

Credential

Example

```
>>> # Get a credential from the Portal
>>> print(get_credential_from_portal('My credential'))
'secretpassword'
```

Keywords password, credential, portal, login, username

Icon las la-key

6.36 Vision

`automagica.activities.is_visible(automagica_id, delay=1, timeout=30)`

Check if element is visible on screen

This activity can be used to check if a certain element is visible on the screen. Note that this uses Automagica Portal and uses some advanced an fuzzy matching algorithms for finding identical elements.

Parameters

- **automagica_id** (*automagica_id*) – Element ID provided by the recorder
- **delay** (*int, optional*) – Delay before checking visibility in seconds
- **timeout** (*int, optional*) – Time before timeout (maximum time to wait) in seconds

Returns

True if visble, False if not

Example

```
>>> # Use the recorder to find an element ID
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↪ also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↪ found, use the recorder
>>> is_visible('qf41')
```

Keywords click, visible, is visible, appear, computer vision, vision, AI

Icon las la-eye

`automagica.activities.wait_appear(automagica_id, delay=1, timeout=30)`

Wait for an element to appear

Wait for an element that is defined the recorder

Parameters

- **automagica_id**(*automagica_id*) – The element ID provided by the recorder
- **delay**(*int*, *optional*) – Delay before waiting to appear in seconds
- **timeout**(*int*, *optional*) – Maximum time to wait for an element in seconds

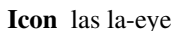
Returns

Blocks while element not visible

Example

```
>>> # Use the recorder to find the element ID to wait for
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↳also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↳found, use the recorder
>>> wait_appear('qf41')
```

Keywords click, computer vision, vision, AI

Icon 

`automagica.activities.wait_vanish(automagica_id, delay=1, timeout=30)`

Wait Vanish

This activity allows the bot to wait for an element to vanish.

Parameters

- **automagica_id**(*automagica_id*) – The element ID provided by the recorder
- **delay**(*int*, *optional*) – Delay before waiting for vanish in seconds
- **timeout** – Maximum time to wait for an element in seconds

```
>>> # Use the recorder to find the element ID for the vanishing element
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↳also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↳found, use the recorder
>>> wait_vanish('qf41')
```

Keywords wait, disappear, computer vision, vision, AI

Icon 

`automagica.activities.read_text(automagica_id, delay=1)`

Read Text with Automagica Wand

This activity allows the bot to detect and read the text of an element by using the Automagica Portal API with a provided sample ID.

Parameters

- **automagica_id**(*automagica_id*) – the sample ID provided by Automagica Wand
- **delay**(*int*, *optional*) – Delay before reading text for vanish in seconds

Returns

Text

Example

```
>>> # Record an element to read with the recorder
>>> # Run the Windows calculator and try to perform the activity
>>> run('calc.exe')
>>> # Use the element ID found by the recorder, in this case ID 'qf41'. You can
↪also view this on automagica.id/qf41
>>> # If you have a vastly different version or layout the element might not be
↪found, use the recorder
>>> read_text('qf41')
```

Keywords click, computer vision, vision, AI

Icon las la-eye

7.1 Introduction

As Automagica is **fully cross-platform** and built with Python, it's actually quite straightforward to build and run Automagica bots from within containerized environments. This makes 'herding' software robots a lot more **maintainable**, **scaleable** and **replicable**.

7.2 Get started

We've added an example Dockerfile with the minimum requirements for running an Automagica bot:

```
git clone https://automagi.ca/github
cd automagica
docker build . -t automagica
docker run -it -p 8080:8080 automagica
```

Then browse to <http://localhost:8080/vnc.html> to access the Automagica desktop.

To get started building your automations, you can run `automagica flow new` in the command line. Check out our [command line documentation](#) for all the automagica commands.

This document contains information and serves as a reference for the Automagica core development team.

8.1 Build installers

8.1.1 Windows One-Click Installer

From Windows 10 (VM), change current working directory to `/installers/windows` and run `build`. After the build process finishes, an executable can be found in the `/installers/windows/build/nsis` directory.

8.1.2 Release on Python Package Index (PyPI)

Release on PyPi by running following commands from the main repository directory and providing PyPi credentials:

```
pip install twine
rmdir /S /Q dist # Windows
python setup.py sdist bdist_wheel
twine upload dist/*
```

8.2 Internationalization

8.2.1 Extracting localization strings with PyBabel

```
pybabel extract -o locale/base.pot automagica
pybabel update -i messages.pot -d locale
```

After modification:

```
pybabel compile -d locale
```

8.2.2 Create a new language .pot-file

For example for pt_BR (Portuguese Brazil)

```
pybabel init -l pt_BR -i locale/base.pot -d locale
```

8.3 Want to join the development team?

8.3.1 Developers

You can contribute in the following ways:

- Pull requests with code and/or documentation
- Feature requests, bug squatting, feel free to [create an issue](#)!
- If you're interested in joining our team, [send us an e-mail](#).

8.3.2 Not a developer?

No problem! You can contribute in the following ways:

- **Star our repository** by clicking the star icon at the top right of this page. This allows us to get more exposure within the GitHub community. The more people we can get involved the better!
- Miss a particular feature? [Create an 'issue'](#)
- Something not working? [Create an issue](#)
- Don't have a GitHub account? Feel free to send us an e-mail at koen@automagica.com or thomas@automagica.com.

9.1 Running unattended automations

Some functionalities that Automagica offers, mostly those that rely on computer vision or OCR, require an active desktop. This can sometimes lead to issues on virtualized environments (servers, VMWare, Virtualbox, Hyper-V, ...) where no computer display is present and the only way of interacting with the machine is through a remote desktop connection. This sometimes causes problems on Microsoft operating systems as various mechanics designed by Microsoft cause the desktop to stop working or group policies set-up within an organization which cause the connection to be closed after a specific amount of time, or example Remote Desktop connections going stale or not rendering a desktop at all. To mitigate or resolve these issues, we provide some solutions here.

9.1.1 Problem description

If the Remote Desktop Protocol connection closes to a virtualized desktop on Windows Server or Windows 10, it could be that the desktop session is ended on the side of the Automagica Bot and this causes the bot to be unable to perform any actions requiring access to the visual layer of the Graphical User Interface (GUI).

9.1.2 Option 1: Keep the Remote Desktop Connection open

The easiest way is to keep the Remote Desktop alive for the robot to work, is to keep a Remote Desktop Connection open from another machine.

For example:

```
Machine 1: Bot's working environment
Machine 2: Other machine

Machine 2 connects with Remote Desktop Protocol to machine 1.
```

To recap, Machine 1 can be viewed and controlled from Machine 2. Once Machine 2 is restarted or the user session is ended in which the Remote Desktop Connections were made, the Remote Desktop connection will close and the session on Machine 1 becomes unresponsive.

The same can be done for multiple machines, but you can have the same machine (for example Machine 2) have multiple connections open at the same time, as long as the Remote Desktop Connection windows are not minimized.

Important note: *the Remote Desktop window cannot be minimized unless some registry values are modified. The default behavior of Microsoft Remote Desktop is that once the window becomes minimized on the client side, the server side stops rendering the visual desktop and therefore the Automagica Bot can no longer access this visual layer. The screen becomes essentially black from the Bot's perspective._*

Modifying the registry for allowing minimized RDP sessions to continue rendering

Important note: in practice, we have seen mixed results for changing these registry values and modifying them poses no guarantee that Microsoft's Remote Desktop actually respects these settings. Please test whether the expected behaviour manifests.

Modify the `HKEY_CURRENT_USER\Software\Wow6432Node\Microsoft\Terminal Server Client` key by adding a new `DWORD` value with the name `RemoteDesktop_SuppressWhenMinimized`. Edit the `DWORD` value as Hexadecimal and enter 2 as data. Reboot the machine.

9.1.3 Option 2: Set-up a VNC server

Another possible solution, and probably the most robust, is to set-up a VNC server on the Automagica Bot's machine. As a VNC server functions differently from Microsoft's Remote Desktop, it creates a virtual display driver which remains active even when the remote connection is closed or a client window is minimized. This also provides the most robust way of creating a stable and predictable environment for the Automagica Bot to work with as the display resolution remains consistent across sessions.

Some VNC server solutions:

- [TigerVNC](#) (open source)
- [UltraVNC](#) (open source)
- [TightVNC](#) (open source)
- [RealVNC](#) (open source/proprietary)
- [TurboVNC](#) (open source)

Important note 1: *we are not affiliated, associated, authorized, endorsed by, or in any way officially connected with any of the proposed solutions*

Important note 2: *the VNC protocol has some inherent security flaws, however, there are solutions to mitigate most of the security risks involved, such as choosing a strong password or encrypting the connection with SSH tunnels. More details can be found [here](#)*

Important note 3: *it is possible that the VNC server configuration conflicts with Microsoft's Remote Desktop, so make sure that you are not locked out of the machine. Connecting through Remote Desktop might also conflict with the VNC Server, so it is advisable to connect only through VNC if this solution is implemented.*

Setting up OpenSSH on Windows Server

To provide an additional security layer on top of VNC, one can use an encrypted SSH tunnel to connect to the VNC server. To install the OpenSSH Client/Server component in Windows Server and Windows 10, please follow [Microsoft's documentation on installing OpenSSH](#).

9.1.4 Option 3: Set-up a virtual display in your virtualization environment

Most virtualization software packages have the option to add a virtual display. Please consult the documentation of your virtualization technology provider to find instructions on how to enable such a virtual display driver. This virtual display provides the Automagica Bot with a persistent visual desktop.

Some popular links:

- [Configure Display Settings for a Virtual Machine with VMWare](#)
- [Display adapters should be enabled in virtual machines to provide video capabilities \(Microsoft Hyper-V\)](#)
- [How to Fix VirtualBox Video Driver Problems in Windows 10](#)
- [How to connect to a VirtualBox VM desktop remotely](#)

Important note: *note 3 for VNC also applies here, the virtual display driver might conflict with Remote Desktop connections. It is advisable to use the virtual display provided by the virtualization provider rather than Remote Desktop connection to interact with the Automagica Bot.*

9.2 Missing runtimes

9.2.1 Error: VCRUNTIME140.dll missing

Install the Visual C++ runtime from Microsoft: <https://www.microsoft.com/en-us/download/details.aspx?id=48145> and try re-installing.

CHAPTER 10

Indices and tables

- `genindex`

a

`automagica.activities`, [26](#)

A

`activate_first_empty_cell_down()` (*automagica.activities.Excel method*), 75
`activate_first_empty_cell_left()` (*automagica.activities.Excel method*), 76
`activate_first_empty_cell_right()` (*automagica.activities.Excel method*), 76
`activate_first_empty_cell_up()` (*automagica.activities.Excel method*), 76
`activate_range()` (*automagica.activities.Excel method*), 77
`activate_worksheet()` (*automagica.activities.Excel method*), 77
`activate_worksheet()` (*automagica.activities.ExcelFile method*), 85
`ActiveDirectory` (*class in automagica.activities*), 103
`add_contact()` (*automagica.activities.Outlook method*), 72
`add_slide()` (*automagica.activities.PowerPoint method*), 88
`add_text()` (*automagica.activities.PowerPoint method*), 89
`add_worksheet()` (*automagica.activities.Excel method*), 77
`add_worksheet()` (*automagica.activities.ExcelFile method*), 85
`append_line()` (*in module automagica.activities*), 109
`append_text()` (*automagica.activities.Word method*), 66
`append_text()` (*automagica.activities.WordFile method*), 70
`apply_watermark_to_pdf()` (*in module automagica.activities*), 114
`automagica.activities` (*module*), 26

B

`beep()` (*in module automagica.activities*), 98

`browse_to()` (*automagica.activities.Chrome method*), 41
`by_class()` (*automagica.activities.Chrome method*), 42
`by_class_and_by_text()` (*automagica.activities.Chrome method*), 42
`by_classes()` (*automagica.activities.Chrome method*), 42
`by_id()` (*automagica.activities.Chrome method*), 43
`by_xpath()` (*automagica.activities.Chrome method*), 43
`by_xpaths()` (*automagica.activities.Chrome method*), 44

C

`Chrome` (*class in automagica.activities*), 41
`clear_clipboard()` (*in module automagica.activities*), 97
`click()` (*in module automagica.activities*), 55
`click_coordinates()` (*in module automagica.activities*), 55
`click_on_text_ocr()` (*in module automagica.activities*), 123
`close_remote_desktop()` (*in module automagica.activities*), 95
`close_window()` (*in module automagica.activities*), 101
`copy_file()` (*in module automagica.activities*), 111
`copy_folder()` (*in module automagica.activities*), 64
`create_directory()` (*automagica.activities.FTP method*), 49
`create_folder()` (*in module automagica.activities*), 61
`create_new_job_in_portal()` (*in module automagica.activities*), 127
`crop_image()` (*in module automagica.activities*), 120

D

`decrypt_file_with_key()` (*in module automagica.activities*), 29

`decrypt_text_with_key()` (in module *automagica.activities*), 28

`delete_column()` (*automagica.activities.Excel method*), 78

`delete_credential()` (in module *automagica.activities*), 48

`delete_mails()` (*automagica.activities.Outlook method*), 72

`delete_row()` (*automagica.activities.Excel method*), 78

`delete_slide()` (*automagica.activities.PowerPoint method*), 89

`desktop_path()` (in module *automagica.activities*), 104

`directory_exists()` (*automagica.activities.FTP method*), 49

`disable_network_interface()` (in module *automagica.activities*), 99

`display_mouse_position()` (in module *automagica.activities*), 54

`display_osd_message()` (in module *automagica.activities*), 41

`double_click()` (in module *automagica.activities*), 56

`double_click_coordinates()` (in module *automagica.activities*), 55

`double_click_on_text_ocr()` (in module *automagica.activities*), 124

`download_file()` (*automagica.activities.FTP method*), 50

`download_file_from_url()` (in module *automagica.activities*), 105

`downloads_path()` (in module *automagica.activities*), 104

`drag_mouse_to()` (in module *automagica.activities*), 59

`drag_mouse_to_coordinates()` (in module *automagica.activities*), 58

E

`empty_folder()` (in module *automagica.activities*), 63

`enable_network_interface()` (in module *automagica.activities*), 99

`encrypt_file_with_key()` (in module *automagica.activities*), 28

`encrypt_text_with_key()` (in module *automagica.activities*), 27

`enumerate_files()` (*automagica.activities.FTP method*), 50

`Excel` (class in *automagica.activities*), 75

`ExcelFile` (class in *automagica.activities*), 85

`execute_robotframework_test()` (in module *automagica.activities*), 125

`execute_uipath_process()` (in module *automagica.activities*), 124

`exit()` (*automagica.activities.Chrome method*), 44

`export_slides_to_images()` (*automagica.activities.PowerPoint method*), 89

`export_to_html()` (*automagica.activities.Word method*), 67

`export_to_pdf()` (*automagica.activities.Excel method*), 78

`export_to_pdf()` (*automagica.activities.PowerPoint method*), 90

`export_to_pdf()` (*automagica.activities.Word method*), 67

`extract_images_from_pdf()` (in module *automagica.activities*), 114

`extract_page_range_from_pdf()` (in module *automagica.activities*), 113

`extract_text_ocr()` (in module *automagica.activities*), 122

F

`file_exists()` (in module *automagica.activities*), 107

`find_all_links()` (*automagica.activities.Chrome method*), 45

`find_elements_by_text()` (*automagica.activities.Chrome method*), 45

`find_first_link()` (*automagica.activities.Chrome method*), 45

`find_text_on_screen_ocr()` (in module *automagica.activities*), 123

`find_window_title()` (in module *automagica.activities*), 94

`folder_exists()` (in module *automagica.activities*), 64

`FTP` (class in *automagica.activities*), 49

G

`generate_date_today()` (in module *automagica.activities*), 39

`generate_hash_from_file()` (in module *automagica.activities*), 30

`generate_hash_from_text()` (in module *automagica.activities*), 30

`generate_key_from_password()` (in module *automagica.activities*), 29

`generate_random_address()` (in module *automagica.activities*), 36

`generate_random_beep()` (in module *automagica.activities*), 38

`generate_random_boolean()` (in module *automagica.activities*), 33

`generate_random_data()` (in module *automagica.activities*), 31

`generate_random_date()` (in module *automagica.activities*), 38
`generate_random_key()` (in module *automagica.activities*), 27
`generate_random_name()` (in module *automagica.activities*), 33
`generate_random_number()` (in module *automagica.activities*), 31
`generate_random_words()` (in module *automagica.activities*), 35
`generate_unique_identifier()` (in module *automagica.activities*), 40
`get_all_network_interface_names()` (in module *automagica.activities*), 98
`get_boot_time()` (in module *automagica.activities*), 117
`get_contacts()` (*automagica.activities.Outlook method*), 73
`get_cpu_frequency()` (in module *automagica.activities*), 116
`get_cpu_load()` (in module *automagica.activities*), 115
`get_cpu_stats()` (in module *automagica.activities*), 116
`get_credential()` (in module *automagica.activities*), 48
`get_credential_from_portal()` (in module *automagica.activities*), 128
`get_default_printer_name()` (in module *automagica.activities*), 99
`get_disk_partitions()` (in module *automagica.activities*), 117
`get_disk_stats()` (in module *automagica.activities*), 117
`get_file_extension()` (in module *automagica.activities*), 111
`get_files_in_folder()` (in module *automagica.activities*), 60
`get_folders()` (*automagica.activities.Outlook method*), 73
`get_foreground_window_title()` (in module *automagica.activities*), 101
`get_from_clipboard()` (in module *automagica.activities*), 97
`get_image_height()` (in module *automagica.activities*), 119
`get_image_width()` (in module *automagica.activities*), 119
`get_mails()` (*automagica.activities.Outlook method*), 73
`get_memory_stats()` (in module *automagica.activities*), 116
`get_mouse_position()` (in module *automagica.activities*), 54
`get_number_of_cpu()` (in module *automagica.activities*), 115
`get_object_by_distinguished_name()` (*automagica.activities.ActiveDirectory method*), 103
`get_running_processes()` (in module *automagica.activities*), 121
`get_service_status()` (in module *automagica.activities*), 100
`get_table()` (*automagica.activities.Excel method*), 79
`get_text_on_webpage()` (*automagica.activities.Chrome method*), 46
`get_time_since_last_boot()` (in module *automagica.activities*), 118
`get_username()` (in module *automagica.activities*), 96
`get_worksheet_names()` (*automagica.activities.Excel method*), 79
`get_worksheet_names()` (*automagica.activities.ExcelFile method*), 86

H

`hide_window()` (in module *automagica.activities*), 102
`highlight()` (*automagica.activities.Chrome method*), 46
`home_path()` (in module *automagica.activities*), 104

I

`insert_data_as_table()` (*automagica.activities.Excel method*), 80
`insert_empty_column()` (*automagica.activities.Excel method*), 80
`insert_empty_row()` (*automagica.activities.Excel method*), 80
`is_desktop_locked()` (in module *automagica.activities*), 96
`is_logged_in()` (in module *automagica.activities*), 96
`is_process_running()` (in module *automagica.activities*), 121
`is_visible()` (in module *automagica.activities*), 128

J

`join_pdf_files()` (in module *automagica.activities*), 113

K

`kill_process()` (in module *automagica.activities*), 122

L

`lock_windows()` (in module *automagica.activities*), 95

M

`make_text_file()` (in module *automagica.activities*), 110

`maximize_window()` (in module *automagica.activities*), 101

`minimize_window()` (in module *automagica.activities*), 102

`mirror_image_horizontally()` (in module *automagica.activities*), 120

`mirror_image_vertically()` (in module *automagica.activities*), 120

`most_recent_file()` (in module *automagica.activities*), 65

`move_file()` (in module *automagica.activities*), 106

`move_folder()` (in module *automagica.activities*), 62

`move_mails()` (*automagica.activities.Outlook method*), 74

`move_mouse_relative()` (in module *automagica.activities*), 58

`move_mouse_to()` (in module *automagica.activities*), 57

`move_mouse_to_coordinates()` (in module *automagica.activities*), 57

N

`number_of_slides()` (*automagica.activities.PowerPoint method*), 90

O

`open_file()` (in module *automagica.activities*), 105

`Outlook` (class in *automagica.activities*), 72

P

`PowerPoint` (class in *automagica.activities*), 88

`press_key()` (in module *automagica.activities*), 51

`press_key_combination()` (in module *automagica.activities*), 52

`print_console()` (in module *automagica.activities*), 41

Q

`quit()` (*automagica.activities.Excel method*), 81

`quit()` (*automagica.activities.Outlook method*), 74

`quit()` (*automagica.activities.PowerPoint method*), 91

`quit()` (*automagica.activities.Word method*), 67

R

`raise_exception()` (in module *automagica.activities*), 127

`random_screen_snippet()` (in module *automagica.activities*), 59

`read_all_text()` (*automagica.activities.Word method*), 68

`read_all_text()` (*automagica.activities.WordFile method*), 70

`read_cell()` (*automagica.activities.Excel method*), 81

`read_cell()` (*automagica.activities.ExcelFile method*), 86

`read_cell_formula()` (*automagica.activities.Excel method*), 82

`read_from_txt()` (in module *automagica.activities*), 109

`read_list_from_txt()` (in module *automagica.activities*), 108

`read_range()` (*automagica.activities.Excel method*), 82

`read_text()` (in module *automagica.activities*), 129

`read_text_file_to_list()` (in module *automagica.activities*), 110

`read_text_from_pdf()` (in module *automagica.activities*), 112

`read_worksheet()` (*automagica.activities.Excel method*), 82

`remove_file()` (in module *automagica.activities*), 107

`remove_folder()` (in module *automagica.activities*), 63

`remove_printer()` (in module *automagica.activities*), 100

`rename_file()` (in module *automagica.activities*), 106

`rename_folder()` (in module *automagica.activities*), 61

`replace_text()` (*automagica.activities.PowerPoint method*), 91

`replace_text()` (*automagica.activities.Word method*), 68

`replace_text()` (*automagica.activities.WordFile method*), 70

`resize_image()` (in module *automagica.activities*), 118

`resize_window()` (in module *automagica.activities*), 102

`restore_window()` (in module *automagica.activities*), 101

`right_click()` (in module *automagica.activities*), 56

`right_click_coordinates()` (in module *automagica.activities*), 57

`right_click_on_text_ocr()` (in module *automagica.activities*), 124

`rotate_image()` (in module *automagica.activities*), 118

[run\(\)](#) (in module *automagica.activities*), [121](#)
[run_autoit_script\(\)](#) (in module *automagica.activities*), [125](#)
[run_automationanywhere_task\(\)](#) (in module *automagica.activities*), [126](#)
[run_blueprism_process\(\)](#) (in module *automagica.activities*), [126](#)
[run_macro\(\)](#) (*automagica.activities.Excel* method), [83](#)
[run_manual\(\)](#) (in module *automagica.activities*), [121](#)
[run_ssh_command\(\)](#) (in module *automagica.activities*), [102](#)
[run_vbs_script\(\)](#) (in module *automagica.activities*), [98](#)

S

[salesforce_api_call\(\)](#) (in module *automagica.activities*), [93](#)
[save\(\)](#) (*automagica.activities.Excel* method), [83](#)
[save\(\)](#) (*automagica.activities.ExcelFile* method), [87](#)
[save\(\)](#) (*automagica.activities.PowerPoint* method), [91](#)
[save\(\)](#) (*automagica.activities.Word* method), [69](#)
[save\(\)](#) (*automagica.activities.WordFile* method), [71](#)
[save_all_images\(\)](#) (*automagica.activities.Chrome* method), [47](#)
[save_as\(\)](#) (*automagica.activities.Excel* method), [83](#)
[save_as\(\)](#) (*automagica.activities.ExcelFile* method), [87](#)
[save_as\(\)](#) (*automagica.activities.PowerPoint* method), [92](#)
[save_as\(\)](#) (*automagica.activities.Word* method), [69](#)
[save_as\(\)](#) (*automagica.activities.WordFile* method), [71](#)
[save_attachments\(\)](#) (*automagica.activities.Outlook* method), [74](#)
[send_email_with_outlook365\(\)](#) (in module *automagica.activities*), [92](#)
[send_mail\(\)](#) (*automagica.activities.Outlook* method), [75](#)
[send_mail_smtp\(\)](#) (in module *automagica.activities*), [93](#)
[send_to_printer\(\)](#) (in module *automagica.activities*), [112](#)
[set_credential\(\)](#) (in module *automagica.activities*), [47](#)
[set_default_printer\(\)](#) (in module *automagica.activities*), [99](#)
[set_footers\(\)](#) (*automagica.activities.Word* method), [69](#)
[set_headers\(\)](#) (*automagica.activities.Word* method), [69](#)
[set_headers\(\)](#) (*automagica.activities.WordFile* method), [71](#)

[set_to_clipboard\(\)](#) (in module *automagica.activities*), [97](#)
[set_user_password\(\)](#) (in module *automagica.activities*), [95](#)
[set_wallpaper\(\)](#) (in module *automagica.activities*), [105](#)
[set_window_to_foreground\(\)](#) (in module *automagica.activities*), [100](#)
[show_folder\(\)](#) (in module *automagica.activities*), [62](#)
[show_image\(\)](#) (in module *automagica.activities*), [118](#)
[snmp_get\(\)](#) (in module *automagica.activities*), [103](#)
[start_remote_desktop\(\)](#) (in module *automagica.activities*), [94](#)
[start_service\(\)](#) (in module *automagica.activities*), [100](#)
[stop_service\(\)](#) (in module *automagica.activities*), [100](#)
[switch_to_iframe\(\)](#) (*automagica.activities.Chrome* method), [47](#)

T

[take_screenshot\(\)](#) (in module *automagica.activities*), [60](#)
[to_dataframe\(\)](#) (*automagica.activities.ExcelFile* method), [87](#)
[typing\(\)](#) (in module *automagica.activities*), [53](#)

U

[unzip\(\)](#) (in module *automagica.activities*), [65](#)
[upload_file\(\)](#) (*automagica.activities.FTP* method), [51](#)

V

[validate_user_password\(\)](#) (in module *automagica.activities*), [95](#)

W

[wait\(\)](#) (in module *automagica.activities*), [66](#)
[wait_appear\(\)](#) (in module *automagica.activities*), [128](#)
[wait_file_exists\(\)](#) (in module *automagica.activities*), [108](#)
[wait_folder_exists\(\)](#) (in module *automagica.activities*), [66](#)
[wait_vanish\(\)](#) (in module *automagica.activities*), [129](#)
[Word](#) (class in *automagica.activities*), [66](#)
[WordFile](#) (class in *automagica.activities*), [70](#)
[write_cell\(\)](#) (*automagica.activities.Excel* method), [84](#)
[write_cell\(\)](#) (*automagica.activities.ExcelFile* method), [88](#)
[write_cell_formula\(\)](#) (*automagica.activities.Excel* method), [84](#)

`write_list_to_file()` (*in module automagica.activities*), [108](#)
`write_range()` (*automagica.activities.Excel method*), [85](#)

Z

`zip_folder()` (*in module automagica.activities*), [64](#)