Robots from Jupyter

Workshop on authoring Robot Framework test and task suites with JupyterLab

Workshop 16.1.2019

- Setting up JupyterLab + Robot
- Introducing JupyterLab
- Exercise: Python notebook
- Exercise: Robot notebook
- Selenium autocompletion
- Exercise: Multiple notebooks
- Sharing and exporting notebooks
- Executing notebooks
- Look into the Jupyter ecosystem

Setting up JupyterLab + Robot

RobotLab bundle installer

https://github.com/robots-from-jupyter/robotlab/releases

- Windows, MacOS, Linux
- Easy to uninstall (just delete the directory and icon)
- Inconvenient download size (400-500MB)

Ingredients:
Conda, Jupyter[Lab|Library], Robot[Mode|Kernel], Selenium[Library|Screenshots], OpenCV, RESTInstance, [Chrome|Gecko]Driver, example notebooks, tutorial

Manual install with Miniconda

Install Miniconda. Launch Anaconda Prompt. Then

1. `conda install -c conda-forge nodejs jupyterlab robotframework-seleniumlibrary geckodriver python-chromedriver-binary pillow lunr`

2. `pip install robotkernel robotframework-seleniumscreenshots nbimporter`

3. `jupyter labextension install jupyterlab_robotmode`

Run JupyterLab

With RobotLab: Click the RobotLab Icon or...

```
~/robotlab/bin/activate  # osx/linux
c:\robotlab\Scripts\activate.bat  # win
robotlab
```

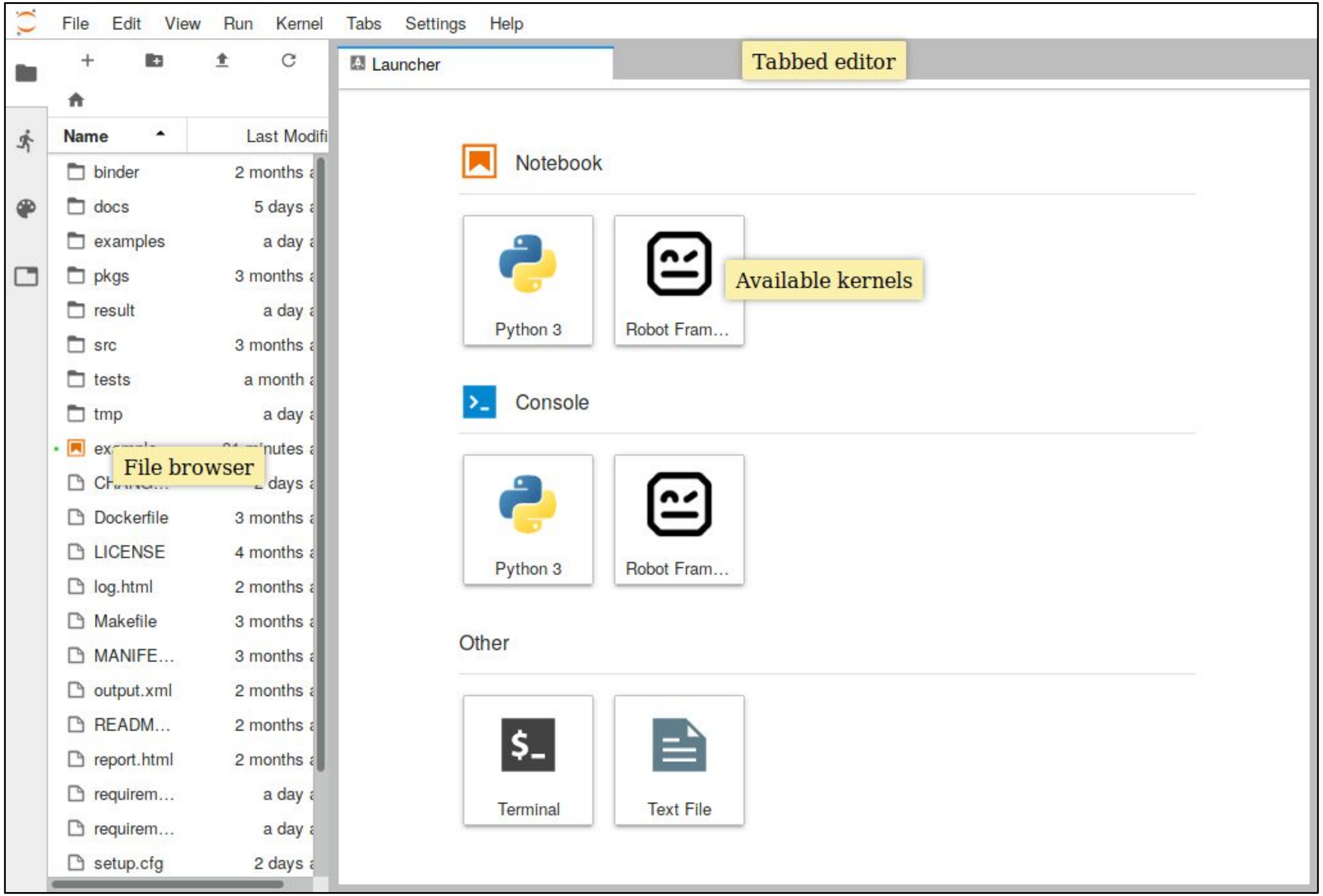
With Anaconda Prompt:

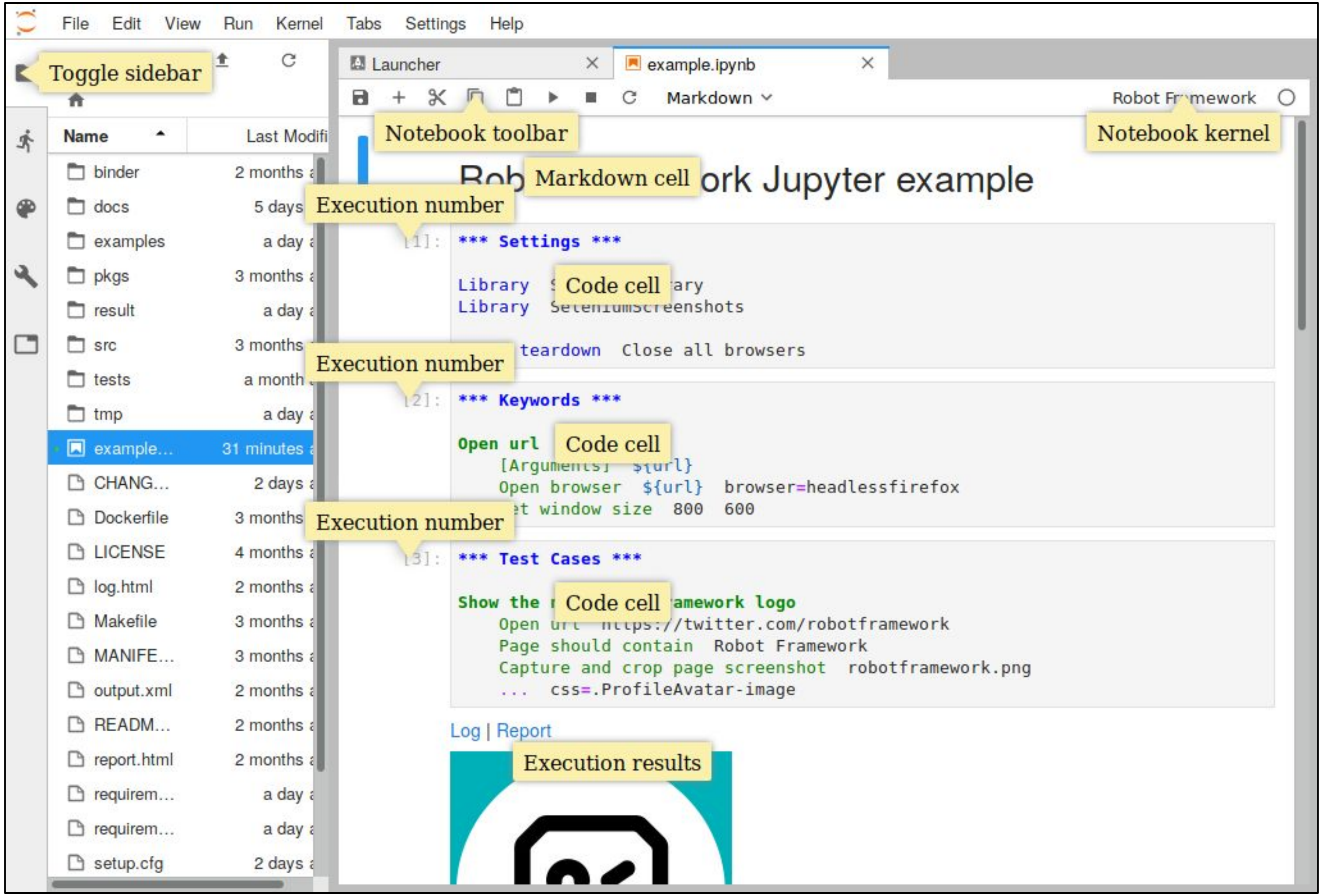
```
jupyter lab
```

With Nix or NixOS:

https://pypi.org/project/robotkernel/

Introducing JupyterLab


File
Edit
View
Run
Kernel
Tabs
Settings
Help
Launcher
Tabbed editor
Name
Last Modifi
binder 2 months
docs 5 days
examples a day
pkgs 3 months
result a day
src 3 months
tests a month
tmp a day
ex... 21 minutes
CHANG... 2 days
Dockerfile 3 months
LICENSE 4 months
log.html 2 months
Makefile 3 months
MANIFE... 3 months
output.xml 2 months
READM... 2 months
report.html 2 months
requirem... a day
requirem... a day
setup.cfg 2 days
File browser
Notebook
Python 3
Robot Fram...
Available kernels
Console
Python 3
Robot Fram...
Other
Terminal
Text File



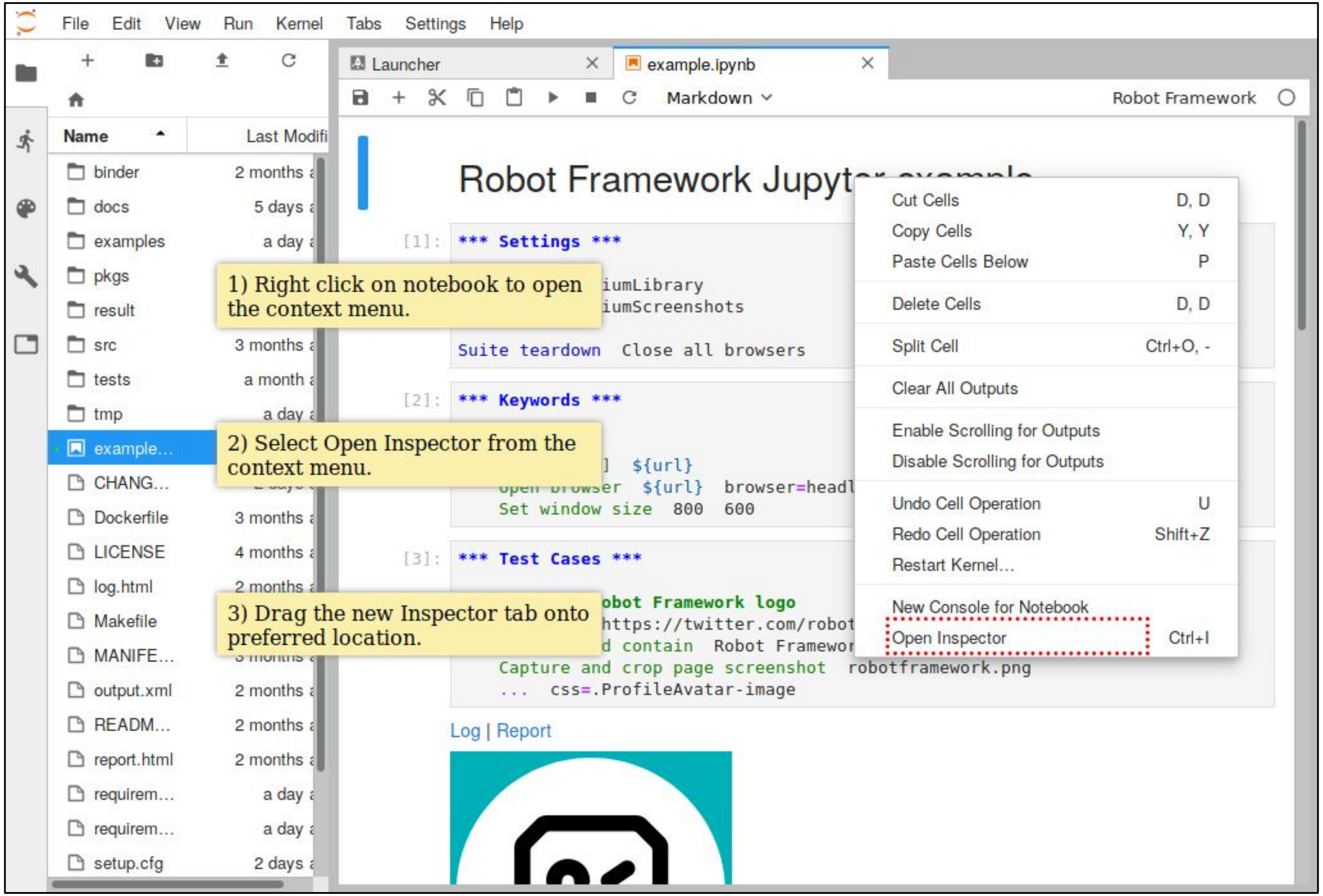
Rob ork Jupyter example

```
[1]:  *** Settings ***

      Library    S         ary
      Library    SeleniumScreenshots

           teardown    Close all browsers
```

```
[2]:  *** Keywords ***

      Open url
          [Arguments]    ${url}
          Open browser    ${url}    browser=headlessfirefox
            t window size    800    600
```

```
[3]:  *** Test Cases ***

      Show the r          amework logo
          Open url    https://twitter.com/robotframework
          Page should contain    Robot Framework
          Capture and crop page screenshot    robotframework.png
          ...    css=.ProfileAvatar-image
```

Log | Report

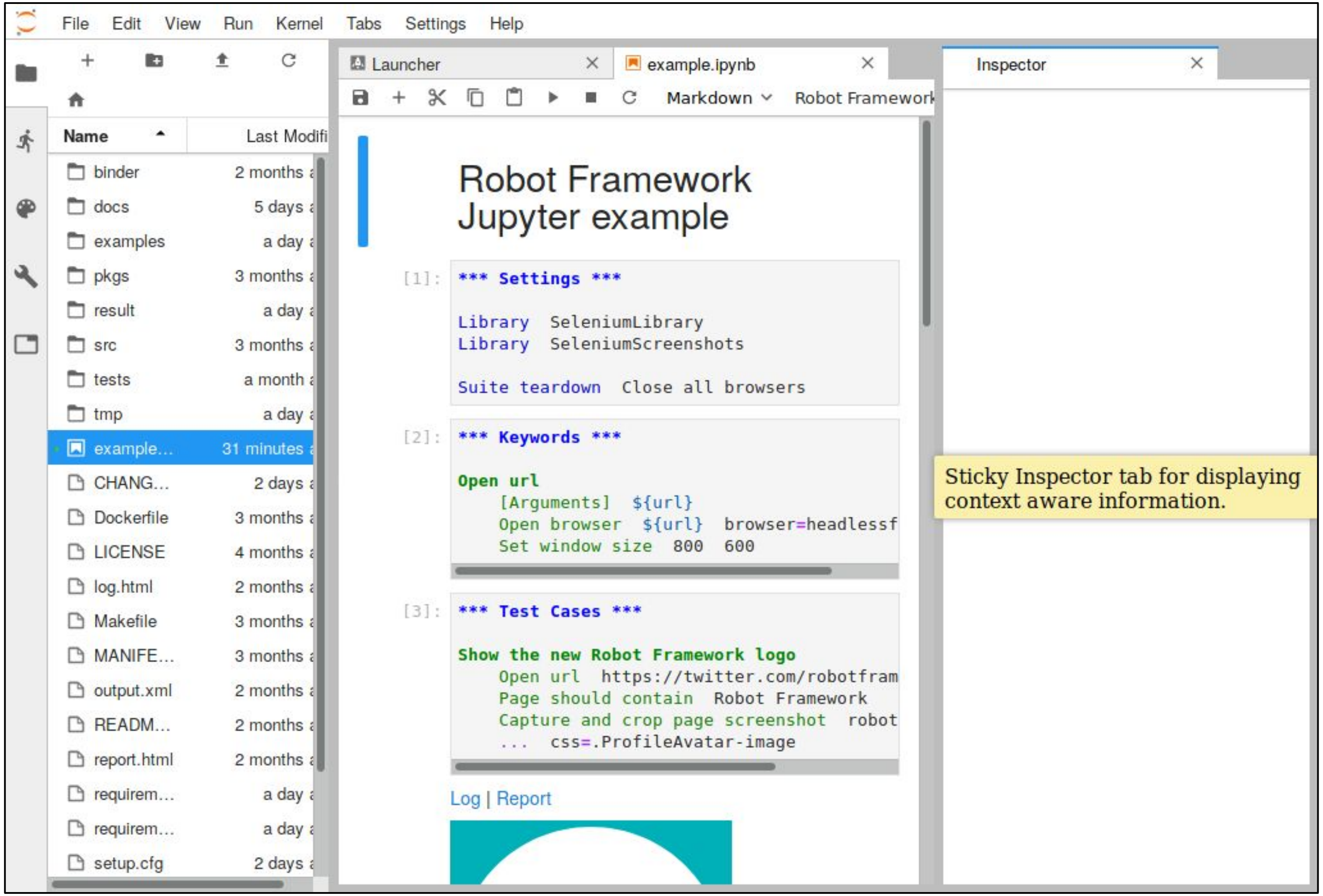


1) Right click on notebook to open the context menu.

2) Select Open Inspector from the context menu.

3) Drag the new Inspector tab onto preferred location.

```
[1]: *** Settings ***
     iumLibrary
     iumScreenshots

     Suite teardown   Close all browsers

[2]: *** Keywords ***

     ]   ${url}
     Open browser   ${url}   browser=headl
     Set window size   800   600

[3]: *** Test Cases ***

     obot Framework logo
     https://twitter.com/robot
     d contain   Robot Framewor
     Capture and crop page screenshot   robotframework.png
     ...   css=.ProfileAvatar-image
```



Robot Framework Jupyter example

```
[1]: *** Settings ***

     Library    SeleniumLibrary
     Library    SeleniumScreenshots

     Suite teardown    Close all browsers
```

```
[2]: *** Keywords ***

     Open url
         [Arguments]    ${url}
         Open browser    ${url}    browser=headlessf
         Set window size    800    600
```

```
[3]: *** Test Cases ***

     Show the new Robot Framework logo
         Open url    https://twitter.com/robotfram
         Page should contain    Robot Framework
         Capture and crop page screenshot    robot
         ...    css=.ProfileAvatar-image
```

Log | Report

Sticky Inspector tab for displaying context aware information.

Default keyboard bindings

up / j	select cell above	y	change cell to code mode
down / k	select cell below	m	change cell to markdown
ctrl + enter	run cell		
shift + enter	run cell, select below	enter	enter edit mode
alt + enter	run cell, insert below	esc	exit edit mode

a	insert cell above	i,i	interrupt kernel
b	insert cell below	0, 0	restart kernel
c	copy cell		
v	paste cell	**Available on edit mode**	
d, d	delete selected cell	tab	code completion / indent
shift + m	merge selected cell(s)	ctrl + shift+ -	split cell

Exercise:
Python notebook

01 Running Code.ipynb
02 Python XKCD.ipynb

Exercise

The JSON API for XKCD is described at https://xkcd.com/json.html

Create a new Python notebook and implement function

```
    def get_xkcd_by_num(num)
```

that accepts an integer and returns XKCD image of the given number.

Write narrative documentation for that function in Markdown and executable Python example lines.

Recap

- JupyterLab user interface (file browser, menu, notebook)
- Loading and creating notebooks
- Opening JupyterLab inspector
- Navigating around notebook
- Editing and executing notebook cells
- Copying, cutting, pasting, moving notebook cells
- Autocompleting things with <TAB>
- Following JupyterLab inspector
- Iterating cell with CTRL+ENTER until ready

Homework: magics

https://ipython.readthedocs.io/en/stable/interactive/magics.html

Magics are “magical” syntax for modifying the underlying Python environment supported mainly by Jupyter Python kernels.

For example

```
    !pip install requests
```

would install requests Python package into Python environment.

Exercise: Robot notebook

Robot Framework

```
*** Settings ***

Library   SeleniumLibrary

*** Tasks ***

Capture screenshot of DuckDuckGo.com
    Open browser   http://duckduckgo.com
    Capture page screenshot
```

http://robotframework.org/robotframework/

03 Running Robot.ipynb
04 Robot XKCD.ipynb

Exercise

The JSON API for XKCD is described at https://xkcd.com/json.html

Create a new Robot notebook and implement keyword

```
*** Keywords ***
Get XKCD by num
    [Arguments]  ${num}
```

that accepts an integer and `[Return]` image of the given number. Write narrative documentation for that keyword in Markdown and executable Robot example `*** Tasks ***` .

RobotKenel quirks (bugs?)

- Some completions can be suggested only after at least one test or task has been executed
- Cells without robot `*** [Headings] ***` or content outside headings may be silently ignored
- Failing library import is silently ignored (but logged)
- RobotKernel requires kernel restart to recover from manually closed SeleniumLibrary browser windows

Recap

- Structure of a robot notebook
- How every robot cell starts with a `*** [Heading] ***`
- Executing robot cells with different section data
- Autocompleting Robot Framework structural words
- Autocompleting Robot Framework keywords
- Using JupyterLab inspector for context documentation
- Using JupyterLab inspector for keyword documentation
- Viewing and downloading logs and reports
- Restarting kernel to reset RobotKernel state
- Capturing and cropping screenshots with Selenium

Interactive Robot: Selenium autocompletion

05 Interactive Selenium.ipynb

Recap

- Leaving a singleton test browser open while iterating
- SeleniumLibrary locator prefixes for suggestions:

```
id:<TAB>          name:<TAB>        link:<TAB>
```

- SeleniumLibrary locator prefixes for completions:

```
id:...<TAB>       name:...<TAB>     link:...<TAB>
tag:...<TAB>      xpath:...<TAB>
partial link:...<TAB>
```

- Interactive SeleniumLibrary picker with:

```
css:<TAB>
```

- Closing the test browser manually / with suite teardown

Exercise:
Multiple notebooks

06 Importing Notebooks.ipynb

Exercise

Parameterize notebook with

```
*** Variables ***
${DEPARTURE_DATE}  ${EMPTY}
${DEPARTURE_TIME}  17.00
```

Modify notebook task to use ${DEPARTURE_TIME} and to prefer ${DEPARTURE_DATE} when it is not empty. You could use either write new Python keywords or use BuiltIn-library http://robotframework.org/robotframework/

Recap

- Authoring a Python keyword library with JupyterLab
- Authoring a Robot Framework keyword resource notebook with JupyterLab
- Importing Python keywords library from a notebook
- Importing Robot Framework keywords from a notebook
- Limits of Robot Framework resources files / notebooks
- Using Python unittest module within a Python notebook
- Defining global variables (overridable by robot runner)

Sharing and exporting notebooks

Sharing and exporting notebooks

- Using JupyterLab
  File → Export Notebook As… →

- Using Jypyter nbconvert
  ```
  jupyter nbconvert --to html MyNotebook.ipynb
  ```

https://nbconvert.readthedocs.io/en/latest/customizing.html

Executing notebooks

Executing notebooks

- Executing notebook with Jupyter

```
jupyter nbconvert --to notebook --execute MyNotebook.ipynb
```

- Executing notebook with RobotKernel

```
nbrobot MyNotebook.ipynb
```

- Executing exported script with Robot Framework

```
jupyter nbconvert --to script MyNotebook.ipynb
robot MyNotebook.robot
```

Look into the Jupyter ecosystem

Look into the Jupyter ecosystem

UI: Notebook Classic

- RISE (with robot)
- nbgrader

UI: JupyterLab

- jupyterlab_vim
- jupyterlab-commenting
- jupyter-widgets

Testing

- nbval

Free Services *

- nbviewer.jupyter.org
- mybinder.org
- Google Colab
- Azure Notebooks

Jupyter Widgets & Renderers

Widgets

- ipywidgets
- pythreejs
- ipyleaflet
- itk-widgets
- plotly.py

Renderers

- jupyter-renderers
- jupyterlab-drawio
- jupyterlab_graphviz

If you have many notebooks...

Run reports

- papermill

Generate documentation

- nbsphinx

Publish to a wiki

- nbconflux

Host an app with a kernel

- jupyter-kernel-gateway

Crazy Demos

Visual Robot Programming

- jupyterlab-blockly

Lab with no server

- jyve

Kernels as Widgets

- ktop

Questions?

Thank you!