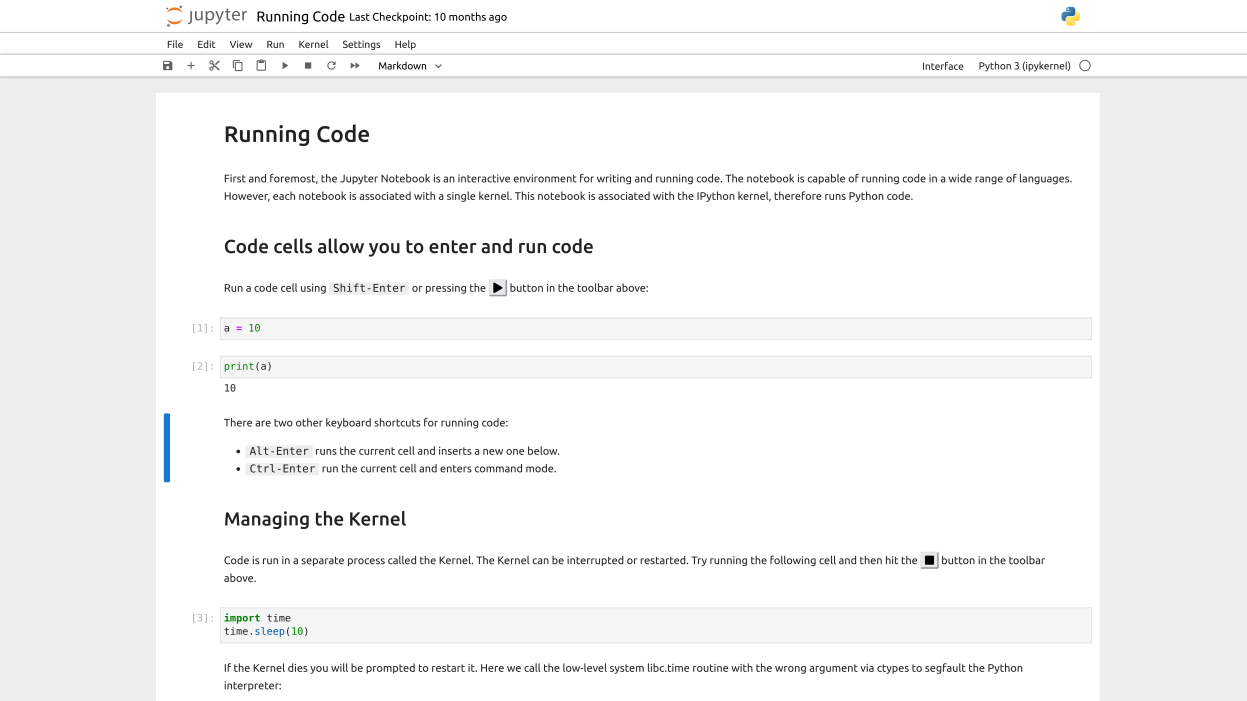
Jupyter Notebook Documentation

Release 7.0.0a5

https://jupyter.org

Jun 27, 2022

CONTENTS



CHAPTER

ONE

USER DOCUMENTATION

1.1 The Jupyter Notebook

1.1.1 Introduction

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results. The Jupyter notebook combines two components:

A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.

Notebook documents: a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects.

See also:

See the installation guide on how to install the notebook and its dependencies.

Main features of the web application

- In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.
- The ability to execute code from the browser, with the results of computations attached to the code which generated them.
- Displaying the result of computation using rich media representations, such as HTML, LaTeX, PNG, SVG, etc. For example, publication-quality figures rendered by the matplotlib library, can be included inline.
- In-browser editing for rich text using the Markdown markup language, which can provide commentary for the code, is not limited to plain text.
- The ability to easily include mathematical notation within markdown cells using LaTeX, and rendered natively by MathJax.

Notebook documents

Notebook documents contains the inputs and outputs of a interactive session as well as additional text that accompanies the code but is not meant for execution. In this way, notebook files can serve as a complete computational record of a session, interleaving executable code with explanatory text, mathematics, and rich representations of resulting objects. These documents are internally JSON files and are saved with the `.ipynb` extension. Since JSON is a plain text format, they can be version-controlled and shared with colleagues.

Notebooks may be exported to a range of static formats, including HTML (for example, for blog posts), reStructured-Text, LaTeX, PDF, and slide shows, via the nbconvert command.

Furthermore, any `.ipynb` notebook document available from a public URL can be shared via the Jupyter Notebook Viewer <nbviewer>. This service loads the notebook document from the URL and renders it as a static web page. The results may thus be shared with a colleague, or as a public blog post, without other users needing to install the Jupyter notebook themselves. In effect, nbviewer is simply nbconvert as a web service, so you can do your own static conversions with nbconvert, without relying on nbviewer.

See also:

Details on the notebook JSON file format

Notebooks and privacy

Because you use Jupyter in a web browser, some people are understandably concerned about using it with sensitive data. However, if you followed the standard install instructions, Jupyter is actually running on your own computer. If the URL in the address bar starts with `http://localhost:` or `http://127.0.0.1:`, it's your computer acting as the server. Jupyter doesn't send your data anywhere else—and as it's open source, other people can check that we're being honest about this.

You can also use Jupyter remotely: your company or university might run the server for you, for instance. If you want to work with sensitive data in those cases, talk to your IT or data protection staff about it.

We aim to ensure that other pages in your browser or other users on the same computer can't access your notebook server. See the security documentation for more about this.

1.1.2 Starting the notebook server

You can start running a notebook server from the command line using the following command:

```
jupyter notebook
```

This will print some information about the notebook server in your console, and open a web browser to the URL of the web application (by default, `http://127.0.0.1:8888`).

The landing page of the Jupyter notebook web application, the **dashboard**, shows the notebooks currently available in the notebook directory (by default, the directory from which the notebook server was started).

You can create new notebooks from the dashboard with the `New Notebook` button, or open existing ones by clicking on their name. You can also drag and drop `.ipynb` notebooks and standard `.py` Python source code files into the notebook list area.

When starting a notebook server from the command line, you can also open a particular notebook directly, bypassing the dashboard, with `jupyter notebook my_notebook.ipynb`. The `.ipynb` extension is assumed if no extension is given.

When you are inside an open notebook, the *File | Open...* menu option will open the dashboard in a new browser tab, to allow you to open another notebook from the notebook directory or to create a new notebook.

Note: You can start more than one notebook server at the same time, if you want to work on notebooks in different directories. By default the first notebook server starts on port 8888, and later notebook servers search for ports near that one. You can also manually specify the port with the `--port` option.

Creating a new notebook document

A new notebook may be created at any time, either from the dashboard, or using the *File → New* menu option from within an active notebook. The new notebook is created within the same directory and will open in a new browser tab. It will also be reflected as a new entry in the notebook list on the dashboard.

Opening notebooks

An open notebook has **exactly one** interactive session connected to a kernel, which will execute code sent by the user and communicate back results. This kernel remains active if the web browser window is closed, and reopening the same notebook from the dashboard will reconnect the web application to the same kernel. In the dashboard, notebooks with an active kernel have a `Shutdown` button next to them, whereas notebooks without an active kernel have a `Delete` button in its place.

Other clients may connect to the same kernel. When each kernel is started, the notebook server prints to the terminal a message like this:

```
[JupyterNotebookApp] Kernel started: 87f7d2c0-13e3-43df-8bb8-1bd37aaf3373
```

This long string is the kernel's ID which is sufficient for getting the information necessary to connect to the kernel. If the notebook uses the IPython kernel, you can also see this connection data by running the `%connect_info` magic, which will print the same ID information along with other details.

You can then, for example, manually start a Qt console connected to the *same* kernel from the command line, by passing a portion of the ID:

```
$ jupyter qtconsole --existing 87f7d2c0
```

Without an ID, `--existing` will connect to the most recently started kernel.

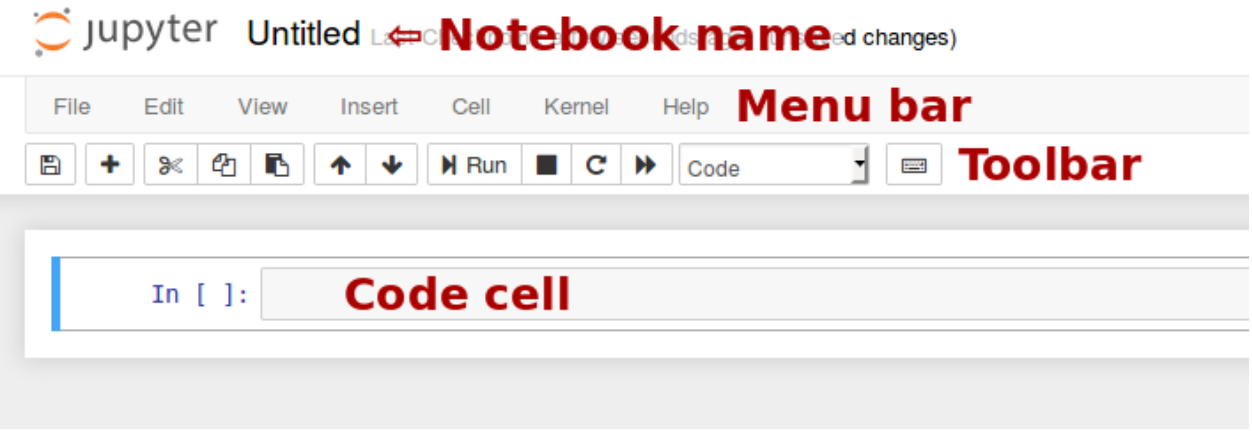
With the IPython kernel, you can also run the `%qtconsole` magic in the notebook to open a Qt console connected to the same kernel.

See also:

Decoupled two-process model

1.1.3 Notebook user interface

When you create a new notebook document, you will be presented with the **notebook name**, a **menu bar**, a **toolbar** and an empty **code cell**.



Notebook name: The name displayed at the top of the page, next to the Jupyter logo, reflects the name of the `.ipynb` file. Clicking on the notebook name brings up a dialog which allows you to rename it. Thus, renaming a notebook from "Untitled0" to "My first notebook" in the browser, renames the `Untitled0.ipynb` file to `My first notebook.ipynb`.

Menu bar: The menu bar presents different options that may be used to manipulate the way the notebook functions.

Toolbar: The tool bar gives a quick way of performing the most-used operations within the notebook, by clicking on an icon.

Code cell: the default type of cell; read on for an explanation of cells.

1.1.4 Structure of a notebook document

The notebook consists of a sequence of cells. A cell is a multiline text input field, and its contents can be executed by using `Shift-Enter`, or by clicking either the "Play" button the toolbar, or *Cell*, *Run* in the menu bar. The execution behavior of a cell is determined by the cell's type. There are three types of cells: **code cells**, **markdown cells**, and **raw cells**. Every cell starts off being a **code cell**, but its type can be changed by using a drop-down on the toolbar (which will be "Code", initially), or via *keyboard shortcuts*.

For more information on the different things you can do in a notebook, see the collection of examples.

Code cells

A *code cell* allows you to edit and write new code, with full syntax highlighting and tab completion. The programming language you use depends on the *kernel*, and the default kernel (IPython) runs Python code.

When a code cell is executed, code that it contains is sent to the kernel associated with the notebook. The results that are returned from this computation are then displayed in the notebook as the cell's *output*. The output is not limited to text, with many other possible forms of output are also possible, including `matplotlib` figures and HTML tables (as used, for example, in the `pandas` data analysis package). This is known as IPython's *rich display* capability.

See also:

Rich Output example notebook

Markdown cells

You can document the computational process in a literate way, alternating descriptive text with code, using *rich text*. In IPython this is accomplished by marking up text with the Markdown language. The corresponding cells are called *Markdown cells*. The Markdown language provides a simple way to perform this text markup, that is, to specify which parts of the text should be emphasized (italics), bold, form lists, etc.

If you want to provide structure for your document, you can use markdown headings. Markdown headings consist of 1 to 6 hash # signs # followed by a space and the title of your section. The markdown heading will be converted to a clickable link for a section of the notebook. It is also used as a hint when exporting to other document formats, like PDF.

When a Markdown cell is executed, the Markdown code is converted into the corresponding formatted rich text. Markdown allows arbitrary HTML code for formatting.

Within Markdown cells, you can also include *mathematics* in a straightforward way, using standard LaTeX notation: `$...$` for inline mathematics and `$$...$$` for displayed mathematics. When the Markdown cell is executed, the LaTeX portions are automatically rendered in the HTML output as equations with high quality typography. This is made possible by MathJax, which supports a large subset of LaTeX functionality

Standard mathematics environments defined by LaTeX and AMS-LaTeX (the `amsmath` package) also work, such as `\begin{equation}...\end{equation}`, and `\begin{align}...\end{align}`. New LaTeX macros may be defined using standard methods, such as `\newcommand`, by placing them anywhere *between math delimiters* in a Markdown cell. These definitions are then available throughout the rest of the IPython session.

See also:

Working with Markdown Cells example notebook

Raw cells

Raw cells provide a place in which you can write *output* directly. Raw cells are not evaluated by the notebook. When passed through nbconvert, raw cells arrive in the destination format unmodified. For example, you can type full LaTeX into a raw cell, which will only be rendered by LaTeX after conversion by nbconvert.

1.1.5 Basic workflow

The normal workflow in a notebook is, then, quite similar to a standard IPython session, with the difference that you can edit cells in-place multiple times until you obtain the desired results, rather than having to rerun separate scripts with the `%run` magic command.

Typically, you will work on a computational problem in pieces, organizing related ideas into cells and moving forward once previous parts work correctly. This is much more convenient for interactive exploration than breaking up a computation into scripts that must be executed together, as was previously necessary, especially if parts of them take a long time to run.

To interrupt a calculation which is taking too long, use the *Kernel*, *Interrupt* menu option, or the `i,i` keyboard shortcut. Similarly, to restart the whole computational process, use the *Kernel*, *Restart* menu option or `0,0` shortcut.

A notebook may be downloaded as a `.ipynb` file or converted to a number of other formats using the menu option *File*, *Download as*.

See also:

Running Code in the Jupyter Notebook example notebook

Notebook Basics example notebook

Keyboard shortcuts

All actions in the notebook can be performed with the mouse, but keyboard shortcuts are also available for the most common ones. The essential shortcuts to remember are the following:

- **`Shift-Enter`: run cell**
  Execute the current cell, show any output, and jump to the next cell below. If `Shift-Enter` is invoked on the last cell, it makes a new cell below. This is equivalent to clicking the *Cell*, *Run* menu item, or the Play button in the toolbar.
- **`Esc`: Command mode**
  In command mode, you can navigate around the notebook using keyboard shortcuts.
- **`Enter`: Edit mode**
  In edit mode, you can edit text in cells.

For the full list of available shortcuts, click *Help*, *Keyboard Shortcuts* in the notebook menus.

1.1.6 Plotting

One major feature of the Jupyter notebook is the ability to display plots that are the output of running code cells. The IPython kernel is designed to work seamlessly with the matplotlib plotting library to provide this functionality. Specific plotting library integration is a feature of the kernel.

1.1.7 Installing kernels

For information on how to install a Python kernel, refer to the IPython install page.

The Jupyter wiki has a long list of Kernels for other languages. They usually come with instructions on how to make the kernel available in the notebook.

1.1.8 Trusting Notebooks

To prevent untrusted code from executing on users' behalf when notebooks open, we store a signature of each trusted notebook. The notebook server verifies this signature when a notebook is opened. If no matching signature is found, Javascript and HTML output will not be displayed until they are regenerated by re-executing the cells.

Any notebook that you have fully executed yourself will be considered trusted, and its HTML and Javascript output will be displayed on load.

If you need to see HTML or Javascript output without re-executing, and you are sure the notebook is not malicious, you can tell Jupyter to trust it at the command-line with:

```
$ jupyter trust mynotebook.ipynb
```

See the security documentation for more details about the trust mechanism.

1.1.9 Browser Compatibility

The Jupyter Notebook aims to support the latest versions of these browsers:

- Chrome
- Safari
- Firefox

Up to date versions of Opera and Edge may also work, but if they don't, please use one of the supported browsers.

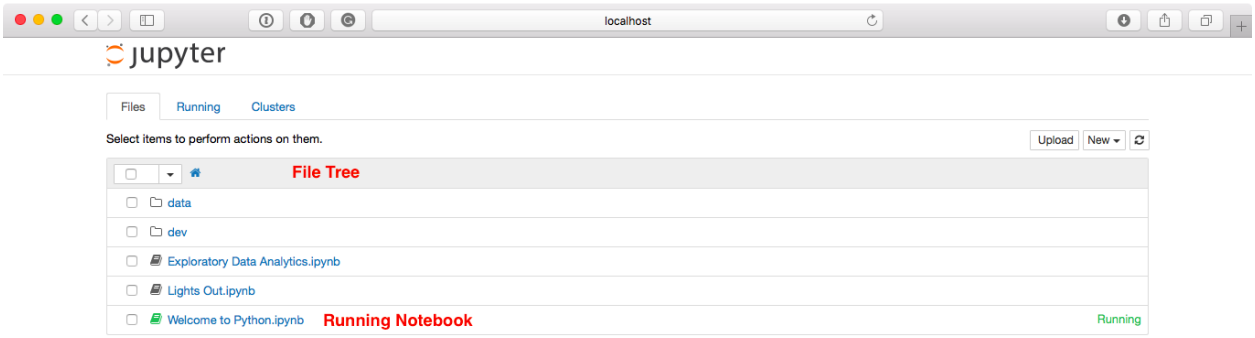
Using Safari with HTTPS and an untrusted certificate is known to not work (websockets will fail).

1.2 User interface components

When opening bug reports or sending emails to the Jupyter mailing list, it is useful to know the names of different UI components so that other developers and users have an easier time helping you diagnose your problems. This section will familiarize you with the names of UI elements within the Notebook and the different Notebook modes.

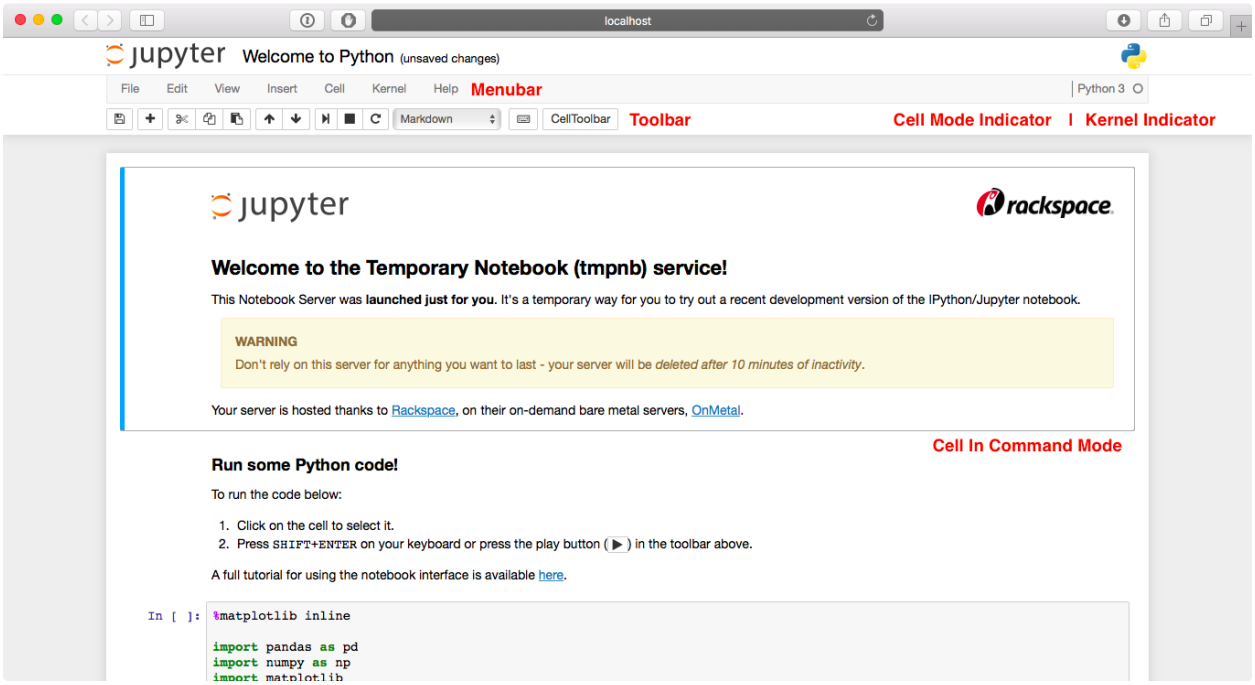
1.2.1 Notebook Dashboard

When you launch `jupyter notebook` the first page that you encounter is the Notebook Dashboard.



1.2.2 Notebook Editor

Once you've selected a Notebook to edit, the Notebook will open in the Notebook Editor.

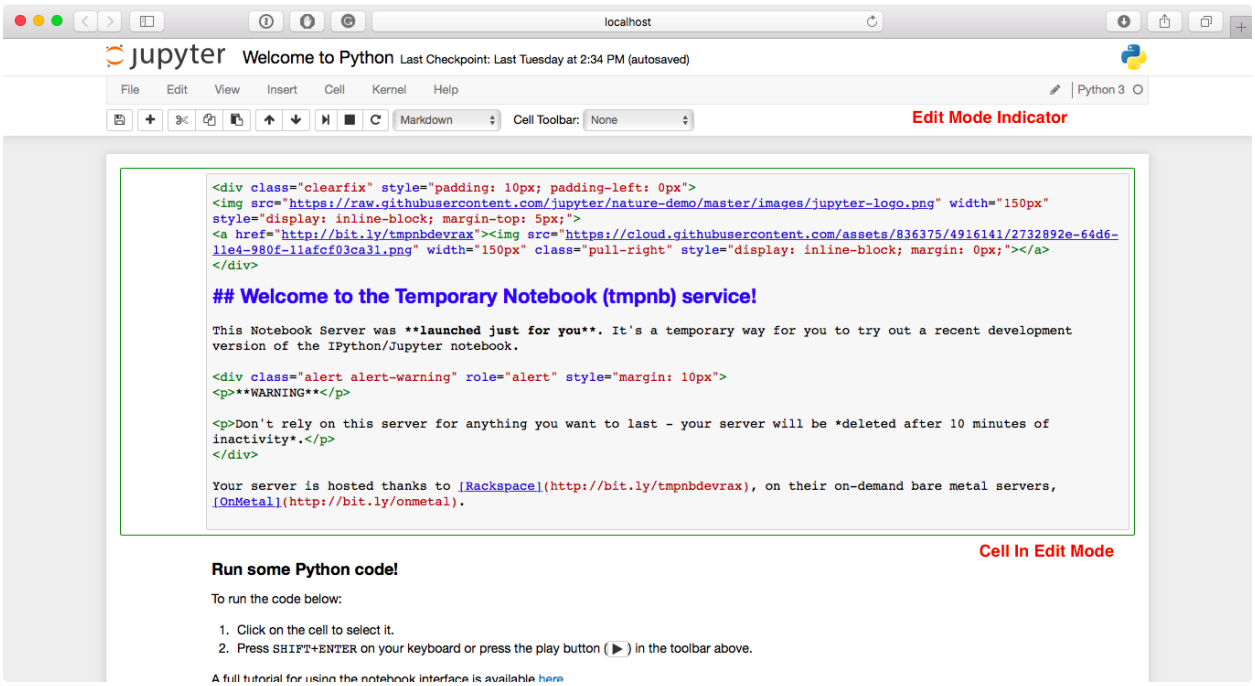


1.2.3 Interactive User Interface Tour of the Notebook

If you would like to learn more about the specific elements within the Notebook Editor, you can go through the user interface tour by selecting *Help* in the menubar then selecting *User Interface Tour*.

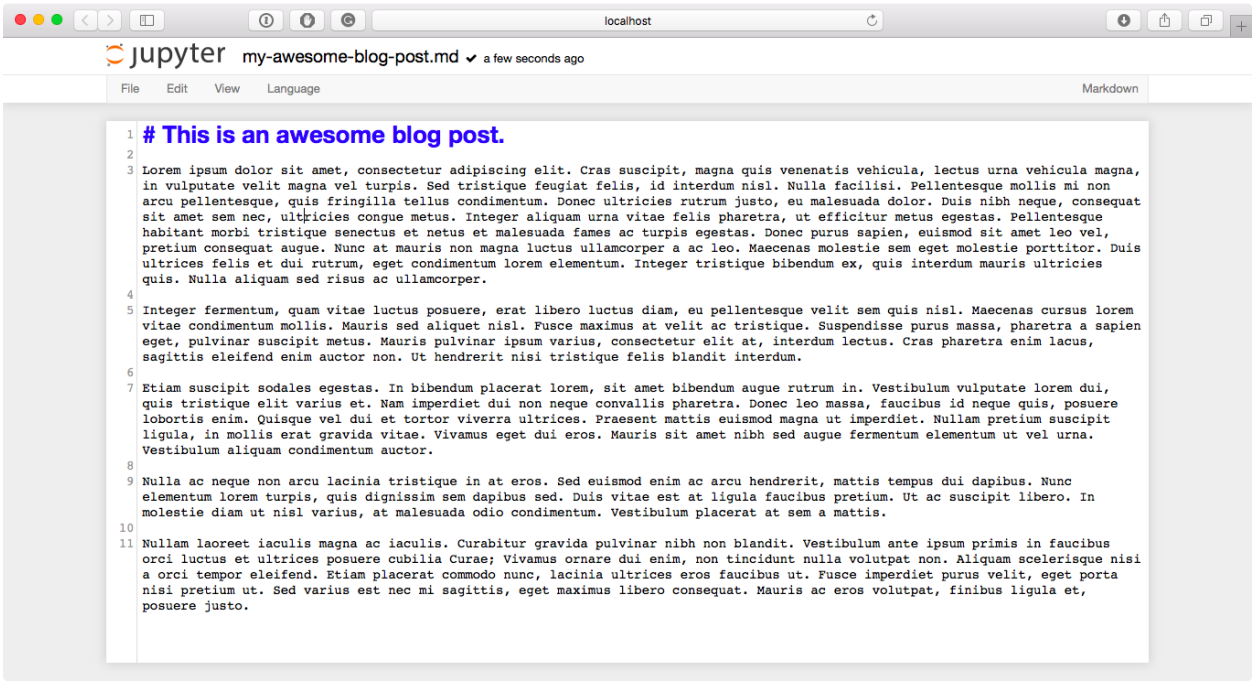
Edit Mode and Notebook Editor

When a cell is in edit mode, the Cell Mode Indicator will change to reflect the cell's state. This state is indicated by a small pencil icon on the top right of the interface. When the cell is in command mode, there is no icon in that location.



1.2.4 File Editor

Now let's say that you've chosen to open a Markdown file instead of a Notebook file whilst in the Notebook Dashboard. If so, the file will be opened in the File Editor.



1.3 Notebook Examples

The pages in this section are all converted notebook files. You can also view these notebooks on nbviewer.

1.3.1 What is the Jupyter Notebook?

Introduction

The Jupyter Notebook is an **interactive computing environment** that enables users to author notebook documents that include: - Live code - Interactive widgets - Plots - Narrative text - Equations - Images - Video

These documents provide a **complete and self-contained record of a computation** that can be converted to various formats and shared with others using email, Dropbox, version control systems (like git/GitHub) or nbviewer.jupyter.org.

Components

The Jupyter Notebook combines three components:

- **The notebook web application**: An interactive web application for writing and running code interactively and authoring notebook documents.
- **Kernels**: Separate processes started by the notebook web application that runs users' code in a given language and returns output back to the notebook web application. The kernel also handles things like computations for interactive widgets, tab completion and introspection.
- **Notebook documents**: Self-contained documents that contain a representation of all content visible in the notebook web application, including inputs and outputs of the computations, narrative text, equations, images, and rich media representations of objects. Each notebook document has its own kernel.

Notebook web application

The notebook web application enables users to:

- **Edit code in the browser**, with automatic syntax highlighting, indentation, and tab completion/introspection.
- **Run code from the browser**, with the results of computations attached to the code which generated them.
- See the results of computations with **rich media representations**, such as HTML, LaTeX, PNG, SVG, PDF, etc.
- Create and use **interactive JavaScript widgets**, which bind interactive user interface controls and visualizations to reactive kernel side computations.
- Author **narrative text** using the Markdown markup language.
- Include mathematical equations using **LaTeX syntax in Markdown**, which are rendered in-browser by MathJax.

Kernels

Through Jupyter's kernel and messaging architecture, the Notebook allows code to be run in a range of different programming languages. For each notebook document that a user opens, the web application starts a kernel that runs the code for that notebook. Each kernel is capable of running code in a single programming language and there are kernels available in the following languages:

- Python(https://github.com/ipython/ipython)
- Julia (https://github.com/JuliaLang/IJulia.jl)
- R (https://github.com/IRkernel/IRkernel)
- Ruby (https://github.com/minrk/iruby)
- Haskell (https://github.com/gibiansky/IHaskell)
- Scala (https://github.com/Bridgewater/scala-notebook)
- node.js (https://gist.github.com/Carreau/4279371)
- Go (https://github.com/takluyver/igo)

The default kernel runs Python code. The notebook provides a simple way for users to pick which of these kernels is used for a given notebook.

Each of these kernels communicate with the notebook web application and web browser using a JSON over ZeroMQ/WebSockets message protocol that is described here. Most users don't need to know about these details, but it helps to understand that "kernels run code."

Notebook documents

Notebook documents contain the **inputs and outputs** of an interactive session as well as **narrative text** that accompanies the code but is not meant for execution. **Rich output** generated by running code, including HTML, images, video, and plots, is embeddeed in the notebook, which makes it a complete and self-contained record of a computation.

When you run the notebook web application on your computer, notebook documents are just **files on your local filesystem with a `.ipynb` extension**. This allows you to use familiar workflows for organizing your notebooks into folders and sharing them with others.

Notebooks consist of a **linear sequence of cells**. There are three basic cell types:

- **Code cells:** Input and output of live code that is run in the kernel
- **Markdown cells:** Narrative text with embedded LaTeX equations
- **Raw cells:** Unformatted text that is included, without modification, when notebooks are converted to different formats using nbconvert

Internally, notebook documents are JSON **data** with **binary values** base64 encoded. This allows them to be **read and manipulated programmatically** by any programming language. Because JSON is a text format, notebook documents are version control friendly.

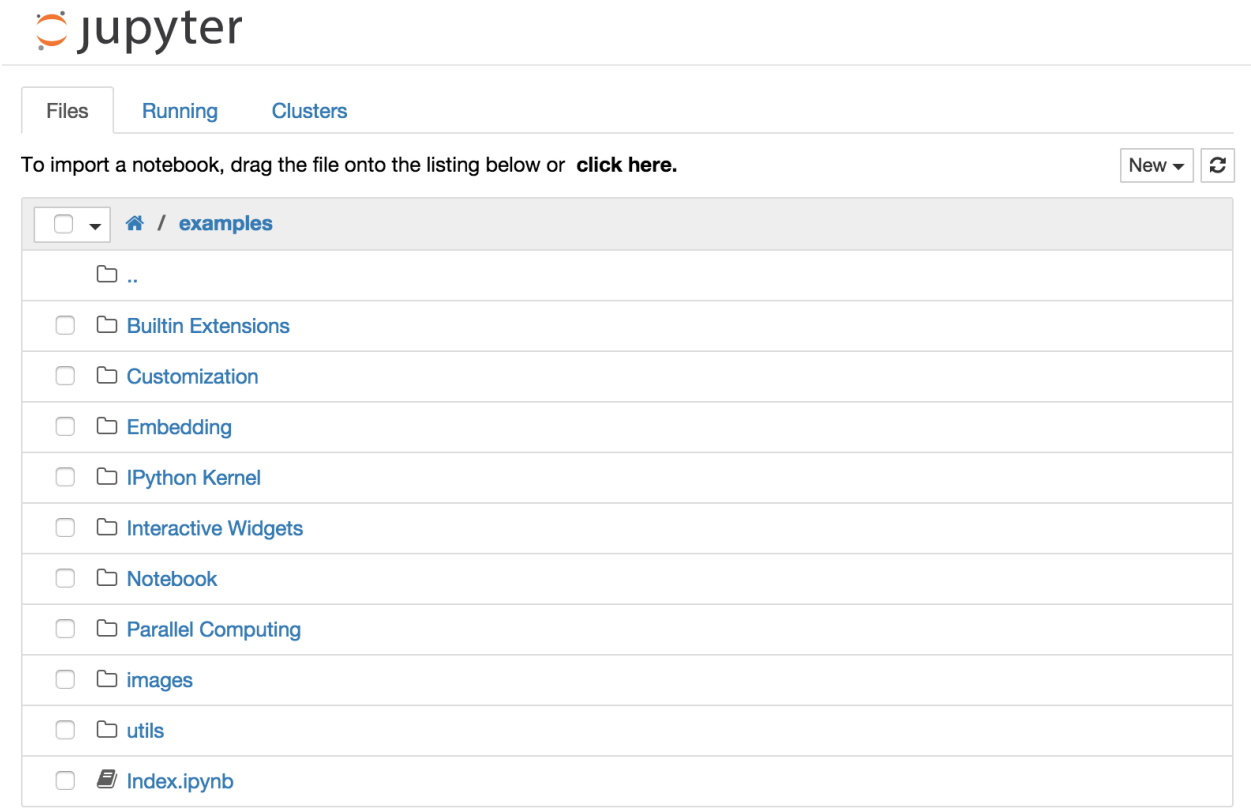
Notebooks can be exported to different static formats including HTML, reStructeredText, LaTeX, PDF, and slide shows (reveal.js) using Jupyter's `nbconvert` utility.

Furthermore, any notebook document available from a **public URL or on GitHub can be shared** via nbviewer. This service loads the notebook document from the URL and renders it as a static web page. The resulting web page may thus be shared with others **without their needing to install the Jupyter Notebook**.

1.3.2 Notebook Basics

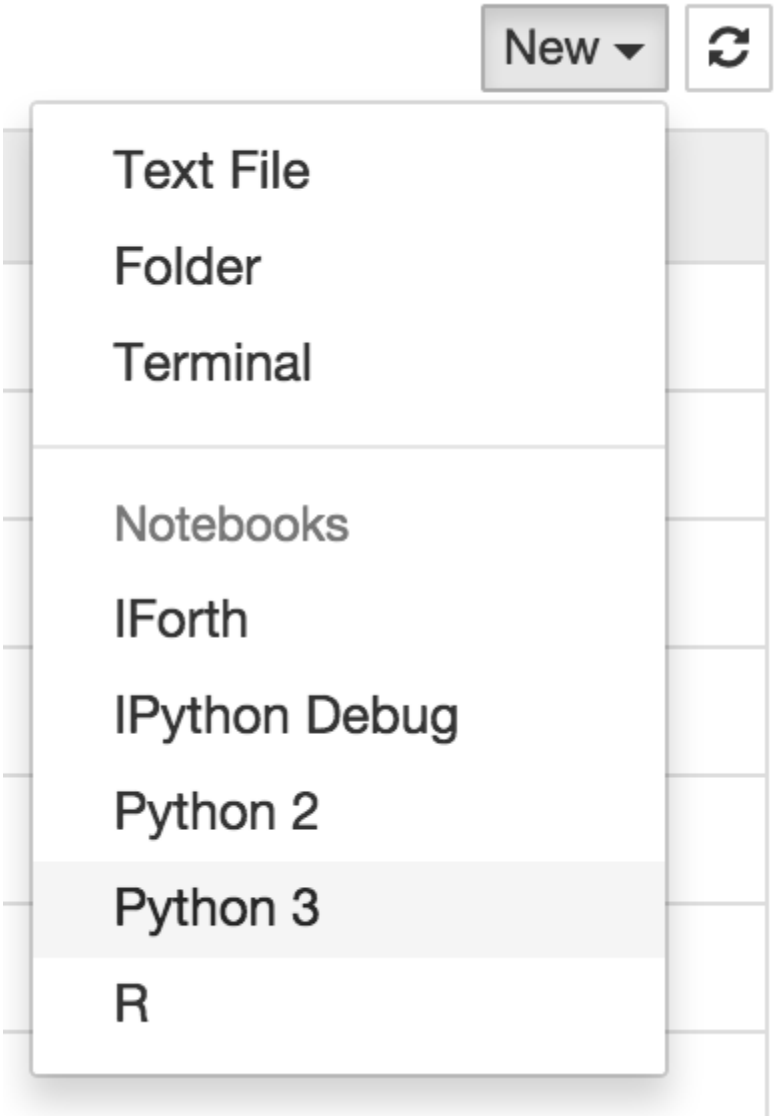
The Notebook dashboard

When you first start the notebook server, your browser will open to the notebook dashboard. The dashboard serves as a home page for the notebook. Its main purpose is to display the notebooks and files in the current directory. For example, here is a screenshot of the dashboard page for the `examples` directory in the Jupyter repository:



The top of the notebook list displays clickable breadcrumbs of the current directory. By clicking on these breadcrumbs or on sub-directories in the notebook list, you can navigate your file system.

To create a new notebook, click on the "New" button at the top of the list and select a kernel from the dropdown (as seen below). Which kernels are listed depend on what's installed on the server. Some of the kernels in the screenshot below may not exist as an option to you.

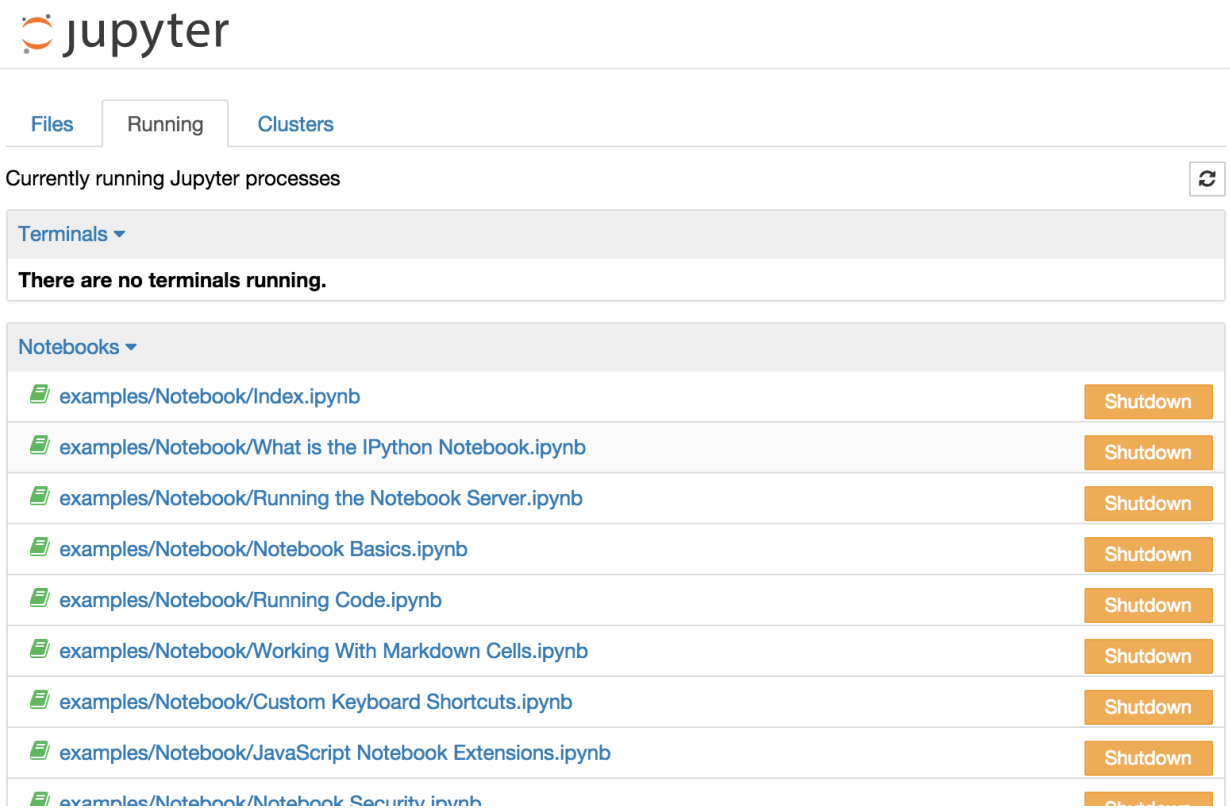


Notebooks and files can be uploaded to the current directory by dragging a notebook file onto the notebook list or by the "click here" text above the list.

The notebook list shows green "Running" text and a green notebook icon next to running notebooks (as seen below). Notebooks remain running until you explicitly shut them down; closing the notebook's page is not sufficient.

To shutdown, delete, duplicate, or rename a notebook check the checkbox next to it and an array of controls will appear at the top of the notebook list (as seen below). You can also use the same operations on directories and files when applicable.

To see all of your running notebooks along with their directories, click on the "Running" tab:



This view provides a convenient way to track notebooks that you start as you navigate the file system in a long running notebook server.

Overview of the Notebook UI

If you create a new notebook or open an existing one, you will be taken to the notebook user interface (UI). This UI allows you to run code and author notebook documents interactively. The notebook UI has the following main areas:

- Menu
- Toolbar
- Notebook area and cells

The notebook has an interactive tour of these elements that can be started in the "Help:User Interface Tour" menu item.

Modal editor

Starting with IPython 2.0, the Jupyter Notebook has a modal user interface. This means that the keyboard does different things depending on which mode the Notebook is in. There are two modes: edit mode and command mode.

Edit mode

Edit mode is indicated by a green cell border and a prompt showing in the editor area:

When a cell is in edit mode, you can type into the cell, like a normal text editor.

Enter edit mode by pressing `Enter` or using the mouse to click on a cell's editor area.

Command mode

Command mode is indicated by a grey cell border with a blue left margin:

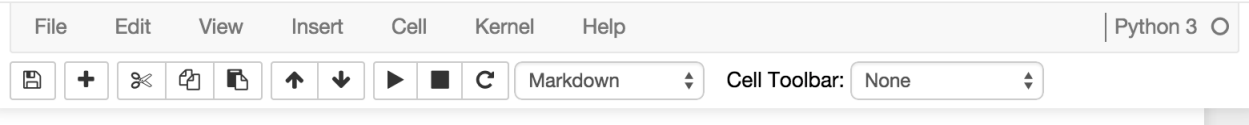
When you are in command mode, you are able to edit the notebook as a whole, but not type into individual cells. Most importantly, in command mode, the keyboard is mapped to a set of shortcuts that let you perform notebook and cell actions efficiently. For example, if you are in command mode and you press `c`, you will copy the current cell - no modifier is needed.

Don't try to type into a cell in command mode; unexpected things will happen!

Enter command mode by pressing `Esc` or using the mouse to click *outside* a cell's editor area.

Mouse navigation

All navigation and actions in the Notebook are available using the mouse through the menubar and toolbar, which are both above the main Notebook area:



The first idea of mouse based navigation is that **cells can be selected by clicking on them.** The currently selected cell gets a grey or green border depending on whether the notebook is in edit or command mode. If you click inside a cell's editor area, you will enter edit mode. If you click on the prompt or output area of a cell you will enter command mode.

If you are running this notebook in a live session (not on http://nbviewer.jupyter.org) try selecting different cells and going between edit and command mode. Try typing into a cell.

The second idea of mouse based navigation is that **cell actions usually apply to the currently selected cell**. Thus if you want to run the code in a cell, you would select it and click the

button in the toolbar or the "Cell:Run" menu item. Similarly, to copy a cell you would select it and click the

button in the toolbar or the "Edit:Copy" menu item. With this simple pattern, you should be able to do most everything you need with the mouse.

Markdown cells have one other state that can be modified with the mouse. These cells can either be rendered or unrendered. When they are rendered, you will see a nice formatted representation of the cell's contents. When they are unrendered, you will see the raw text source of the cell. To render the selected cell with the mouse, click the

button in the toolbar or the "Cell:Run" menu item. To unrender the selected cell, double click on the cell.

Keyboard Navigation

The modal user interface of the Jupyter Notebook has been optimized for efficient keyboard usage. This is made possible by having two different sets of keyboard shortcuts: one set that is active in edit mode and another in command mode.

The most important keyboard shortcuts are `Enter`, which enters edit mode, and `Esc`, which enters command mode.

In edit mode, most of the keyboard is dedicated to typing into the cell's editor. Thus, in edit mode there are relatively few shortcuts. In command mode, the entire keyboard is available for shortcuts, so there are many more. The `Help->``Keyboard Shortcuts`` dialog lists the available shortcuts.

We recommend learning the command mode shortcuts in the following rough order:

1. Basic navigation: `enter`, `shift-enter`, `up/k`, `down/j`
2. Saving the notebook: `s`
3. Change Cell types: `y`, `m`, `1-6`, `t`
4. Cell creation: `a`, `b`
5. Cell editing: `x`, `c`, `v`, `d`, `z`
6. Kernel operations: `i`, `0` (press twice)

1.3.3 Running Code

First and foremost, the Jupyter Notebook is an interactive environment for writing and running code. The notebook is capable of running code in a wide range of languages. However, each notebook is associated with a single kernel. This notebook is associated with the IPython kernel, therefore runs Python code.

Code cells allow you to enter and run code

Run a code cell using `Shift-Enter` or pressing the

button in the toolbar above:

[1]:
```
a = 10
```

[2]:
```
print(a)
```

```
10
```

There are two other keyboard shortcuts for running code:

- `Alt-Enter` runs the current cell and inserts a new one below.
- `Ctrl-Enter` run the current cell and enters command mode.

Managing the Kernel

Code is run in a separate process called the Kernel. The Kernel can be interrupted or restarted. Try running the following cell and then hit the

button in the toolbar above.

[3]:
```
import time
time.sleep(10)
```

If the Kernel dies you will be prompted to restart it. Here we call the low-level system libc.time routine with the wrong argument via ctypes to segfault the Python interpreter:

[5]:
```
import sys
from ctypes import CDLL
# This will crash a Linux or Mac system
# equivalent calls can be made on Windows

# Uncomment these lines if you would like to see the segfault

# dll = 'dylib' if sys.platform == 'darwin' else 'so.6'
# libc = CDLL("libc.%s" % dll)
# libc.time(-1)  # BOOM!!
```

Cell menu

The "Cell" menu has a number of menu items for running code in different ways. These includes:

- Run and Select Below
- Run and Insert Below
- Run All
- Run All Above
- Run All Below

Restarting the kernels

The kernel maintains the state of a notebook's computations. You can reset this state by restarting the kernel. This is done by clicking on the

in the toolbar above.

sys.stdout and sys.stderr

The stdout and stderr streams are displayed as text in the output area.

[6]:
```
print("hi, stdout")
```

```
hi, stdout
```

[7]:
```
from __future__ import print_function
print('hi, stderr', file=sys.stderr)
```

```
hi, stderr
```

Output is asynchronous

All output is displayed asynchronously as it is generated in the Kernel. If you execute the next cell, you will see the output one piece at a time, not all at the end.

[8]:
```
import time, sys
for i in range(8):
    print(i)
    time.sleep(0.5)
```

```
0
1
2
3
4
5
6
7
```

Large outputs

To better handle large outputs, the output area can be collapsed. Run the following cell and then single- or double-click on the active area to the left of the output:

[9]:
```
for i in range(50):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
```

(continues on next page)

(continued from previous page)

```
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
```

Beyond a certain point, output will scroll automatically:

[10]:
```
for i in range(500):
    print(2**i - 1)
```

```
0
1
3
7
15
31
63
127
255
511
1023
2047
4095
8191
16383
32767
65535
131071
262143
524287
```

(continues on next page)

(continued from previous page)

```
1048575
2097151
4194303
8388607
16777215
33554431
67108863
134217727
268435455
536870911
1073741823
2147483647
4294967295
8589934591
17179869183
34359738367
68719476735
137438953471
274877906943
549755813887
1099511627775
2199023255551
4398046511103
8796093022207
17592186044415
35184372088831
70368744177663
140737488355327
281474976710655
562949953421311
1125899906842623
2251799813685247
4503599627370495
9007199254740991
18014398509481983
36028797018963967
72057594037927935
144115188075855871
288230376151711743
576460752303423487
1152921504606846975
2305843009213693951
4611686018427387903
9223372036854775807
18446744073709551615
36893488147419103231
73786976294838206463
147573952589676412927
295147905179352825855
590295810358705651711
1180591620717411303423
2361183241434822606847
```

(continues on next page)

(continued from previous page)

```
472236648286964521369 5
944473296573929042739 1
188894659314785808547 83
377789318629571617095 67
755578637259143234191 35
151115727451828646838 271
302231454903657293676 543
604462909807314587353 087
120892581961462917470 6175
241785163922925834941 2351
483570327845851669882 4703
967140655691703339764 9407
193428131138340667952 98815
386856262276681335905 97631
773712524553362671811 95263
154742504910672534362 390527
309485009821345068724 781055
618970019642690137449 562111
123794003928538027489 9124223
247588007857076054979 8248447
495176015714152109959 6496895
990352031428304219919 2993791
198070406285660843983 85987583
396140812571321687967 71975167
792281625142643375935 43950335
158456325028528675187 087900671
316912650057057350374 175801343
633825300114114700748 351602687
126765060022822940149 6703205375
253530120045645880299 3406410751
507060240091291760598 6812821503
101412048018258352119 73625643007
202824096036516704239 47251286015
405648192073033408478 94502572031
811296384146066816957 89005144063
162259276829213363391 578010288127
324518553658426726783 156020576255
649037107316853453566 312041152511
129807421463370690713 2624082305023
259614842926741381426 5248164610047
519229685853482762853 0496329220095
103845937170696552570 60992658440191
207691874341393105141 21985316880383
415383748682786210282 43970633760767
830767497365572420564 87941267521535
166153499473114484112 975882535043071
332306998946228968225 951765070086143
664613997892457936451 903530140172287
132922799578491587290 3807060280344575
265845599156983174580 7614120560689151
531691198313966349161 5228241121378303
106338239662793269832 3045648224275660 7
```

(continues on next page)

(continued from previous page)

```
212676479325586539664609129644855132l5
42535295865117307932921825928971026431
85070591730234615865843651857942052863
17014118346046923173168730371588410572l
34028236692093846346337460743176821145l
68056473384187692692674921486353642291l
13611294676837538538534984292772072845823
27222589353675077077069968594541456916l7
54445178707350154154139937189082913832l5
10889035741470030830827987437816582766591
21778071482940061661655974875633165533183
43556142965880123323311949751266331066367
87112285931760246646623899502532662132735
17422457186352049329324779900506532426547l
34844914372704098658649559801013064853094l
69689828745408197317299119602026129706188l7
13937965749081639463459823920405225941237l5
27875931498163278926919647840810451882475l1
55751862996326557853839295681620903764951l03
11150372599265311570767859136324180752990207
22300745198530623141535718272648361505980415
44601490397061246283071436545296723011960831
89202980794122492566142873090593446023921663
17840596158824498513228574618118689204784332l7
35681192317648997026457149236237378409568665l5
71362384635297994052914298472474756819137331l1
14272476927059598810582859694494951363827466231
28544953854119197621165719388989902727654932471
57089907708238395242331438777979805455309864951
11417981541647679048466287755595961091061972991
22835963083295358096932575511191922182123945983
45671926166590716193865151022383844364247891967
91343852333181432387730302044767688728495783935
18268770466636286477546060408953537745699156787l1
36537540933272572955092120817907075491398313574l3
73075081866545145910184241635814150982796627148l7
14615016373309029182036848327162830196559325429l75
29230032746618058364073696654325660393118650859l51
58460065493236116728147393308651320786237301719l03
11692013098647223345629478661730264157247460343807
23384026197294446691258957323460528314494920687615
46768052394588893382517914646921056628989841375231
93536104789177786765035829293842113257979682750463
18707220957835557353007165858768422651595936550092l7
37414441915671114706014331717536845303191873100185l5
74828883831342229412028663435073690606383746200371l1
14965776766268445882405732687014738121276749240074l23
29931555352536891764811465374029476242553498480148l47
59863107065073783529622930748058952485106996960296l95
11972621413014756705924586149611790497021399392059391
23945242826029513411849172299223580994042798784118783
47890485652059026823698344598447161988085597568237567
```

(continues on next page)

(continued from previous page)

```
957809713041180536473966891968943239761711951364751351915619426082361072947933783937886479523423902729502713831238852164722145895867567875772959046847805459005437662477704329444291791735135751545918093695610918010871532495540865888858358347027150309183618739122183602175306499108173177771671669405430061836723747824436720435161299821634635554334333881086012367344749564887344087031225996432692711086686677621720247346894991297746881740724519928653854221733733552434404946937899825954937634815490398573077084434674671048688098938757996519098752696317846377169233350954794736779009583020127944305580043141111569275433846670190958947355801916604025588861116008628223313855086769334038191789471160383320805117772223201725644762771017353866807638357894232076664161023554444640345128951255420347077336152767157884641533283220471088892806902579125108406941546723055343157692830665664409421777856138051583502168138830934461106863153856613313288188435557122761031671004336277661868922213726307713226626576376871114245522063352008672555323737844427452615426453253152753742228491044126714017345110647475688854905230852906506305507484456982088253438034690221294951377709810461705813012611014968913964176506871606938044258990275541962092341162602522202993782792835301375321387608851798055108392418468232520504440598756558567060275164277521770359611021678483693646504100888119751311713412055031285550435407192220433569673872930082017762395026234268241100725711008708143844408671393477458601640355247900524685364822015514220174162876888173427869549172032807104958010493707296440311028440348325753776346855739098344065614209916020987414592880631056880696651507552693711478196688131228419832041974829185761272058069665150755269371147819668813122841983204197482918576127411376139330301510538742295639337626245683966408394965837152255822752278660603021077484591278675252491367932816789931674304511164550455732120604215496918255735050498273586563357986334860902332910091146424120843099383651147010099654717312671597266972180476582018229284824168619876730229402019930943462534319453394436095131640364585696483372397534604588040398618869250686389067888721912632807291713929667447950692091760807972377385013727781357774438352656145834278593348959013841835216159447547700274555627155488767105312291668557186697918027683670432318895095400549111254310977535210624583337114373395836055367340864637790190801098222508621955071421249166674228746791672110734681729275580381602196445017243910143842498333348457493583344221469363458551160763204392890034487820287168499666669691498716668844293872691710232152640878578006897564057533699933333938299743333768858774538342046430528175715601379512811516739986666787659948666753771754907668409286105635143120275902562303134799733335753198973335075435098153368185722112702862405518051246072695994666715063979466701508701963067363714442254057248110361024921553919893334301279589334030174039261347274288845081144962207220498431107839786668602559178668060348078522694548577690162289924414440996863215679573337205118357336120696157045389097155380324579848828881993727
```

(continues on next page)

(continued from previous page)

```
43135914667441023671467224139231409077819431076064915969765776398745
86271829334882047342934448278462818155638862152129831939531552797491
17254365866976409468586889655692563631127772430425966387906310559498 23
34508731733952818937173779311385127262255544860851932775812621118996 47
69017463467905637874347558622770254524511089721703865551625242237992 95
13803492693581127574869511724554050904902217944340773110325048447598591
27606985387162255149739023449108101809804435888681546220650096895197183
55213970774324510299478046898216203619608871777363092441300193790394367
11042794154864902059895609379643240723921774355472618488260038758078873 5
22085588309729804119791218759286481447843548710945236976520077516157747 1
44171176619459608239582437518572962895687097421890473953040155032315494 3
88342353238919216479164875037145925791374194843780947906080310064630988 7
17668470647783843295832975007429185158274838968756189581216062012926197 75
35336941295567686591665950014858370316549677937512379162432124025852395 51
70673882591135373183331900029716740633099355875024758324864248051704791 03
14134776518227074636666380005943348126619871175004951664972849610340958207
28269553036454149273332760011886696253239742350009903329945699220681916415
56539106072908298546665520023773392506479484700019806659891398441363832831
11307821214581659709333104004754678501295896940003961331978279688272766566 3
22615642429163319418666208009509357002591793880007922663956559376545533132 7
45231284858326638837332416019018714005183587760015845327913118753091066265 5
90462569716653277674664832038037428010367175520031690655826237506182132531 1
18092513943330655534932966407607485602073435104006338131165247501236426506 23
36185027886661311069865932815214971204146870208012676262330495002472853012 47
72370055773322622139731865630429942408293740416025352524660990004945706024 95
14474011154664524427946373126085988481658748083205070504932198000989141204991
28948022309329048855892746252171976963317496166410141009864396001978282409983
57896044618658097711785492504343953926634992332820282019728792003956564819967
11579208923731619542357098500868790785326998466564056403945758400791312963993 5
23158417847463239084714197001737581570653996933128112807891516801582625927987 1
46316835694926478169428394003475163141307993866256225615783033603165251855974 3
92633671389852956338856788006950326282615987732512451231566067206330503711948 7
18526734277970591267771357601390065256523197546502490246313213441266100742389 75
37053468555941182535542715202780130513046395093004980492626426882532201484779 51
74106937111882365071085430405560261026092790186009960985252853765064402969559 03
14821387422376473014217086081112052205218558037201992197050570753012880593911807
29642774844752946028434172162224104410437116074403984394101141506025761187823615
59285549689505892056868344324448208820874232148807968788202283012051522375647231
11857109937901178411373668864889641764174846429761593757640456602410304475129446 3
23714219875802356822747337729779283528349692859523187515280913204820608950258892 7
47428439751604713645494675459558567056699385719046375030561826409641217900517785 5
94856879503209427290893509191171341133987714380927500611236528192824358010355711
18971375900641885458197870183823426822679754287618550012224730563856487160207114 23
37942751801283770916395740367646853645359508575237100024449461127712974320414228 47
75885503602567541832791480735293707290719017150474200048898922255425948640828456 95
15177100720513508366558296147058741458143803430094840009779784451085189728165691391
30354201441027016733116592294117482916287606860189680019559568902170379456331382783
60708402882054033466233184588234965832575213720379360039119137804340758912662765567
12141680576410806693246636917646993166515042744075872007823827560868151782532553113 5
24283361152821613386493273835293986333030085488151744015647655121736303565065106227 1
48566722305643226772986547670587972666060170976303488031295310243472607130130212454 3
97133444611286453545973095341175945332120341952606976062590620486945214260260424908 7
```

(continues on next page)

(continued from previous page)

```
194266889222572907091946190682351890664240683905213952125181240973890428520520849817 5
388533778445145814183892381364703781328481367810427904250362481947780857041041699635 1
777067556890291628367784762729407562656962735620855808500724963895561714082083399270 3
155413511378058325673556952545881512531392547124171161700144992779112342816416679854 07
310827022756116651347113905091763025062785094248342323400289985558224685632833359708 15
621654045512233302694227810183526050125570188496684646800579971116449371265666719416 31
124330809102446660538845562036705210025114037699336929360115994223289874253133343883263
248661618204893321077691124073410420050228075398673858720231988446579748506266687766527
497323236409786642155382248146820840100456150797347717440463976893159497012533375533055
994646472819573284310764496293641680200912301594695434880927953786318994025066751066111
198929294563914656862152899258728336040182460318939086976185590757263798805013350213222 3
397858589127829313724305798517456672080364920637878173952371181514527597610026700426444 7
795717178255658627448611597034913344160729841275756347904742363029055195220053400852889 5
159143435651131725489722319406982668832145968255151269580948472605811039044010680170577 91
318286871302263450979444638813965337664291936510302539161896945211622078088021360341155 83
636573742604526901958889277627930675328583873020605078323793890423244156176042720682311 67
127314748520905380391777855525586135065716774604121015664758778084648831235208544136462335
254629497041810760783555711051172270131433549208242031329517556169297662470417088272924671
509258994083621521567111422102344540262867098416484062659035112338595324940834176545849343
101851798816724304313422284420468908052573419683296812531807022467719064988166835309169868 7
203703597633448608626844568840937816105146839366593625063614044935438129976333670618339737 5
407407195266897217253689137681875632210293678733187250127228089870876259952667341236679475 1
814814390533794434507378275363751264420587357466374500254456179741752519905334682473358950 3
162962878106758886901475655072750252884117471493274900050891235948350503981066936494671790 07
325925756213517773802951310145500505768234942986549800101782471896701007962133872989343580 15
651851512427035547605902620291001011536469885973099600203564943793402015924267745978687160 31
130370302485407109521180524058200202307293977194619920040712988758680403184853549195737432063
260740604970814219042361048116400404614587954389239840081425977517360806369707098391474864127
521481209941628438084722096232800809229175908778479680162851955034721612739414196782949728255
104296241988325687616944419246560161845835181755695936032570391006944322547882839356589945 6511
208592483976651375233888838493120323691670363511391872065140782013888645095765678713179891302 3
417184967953302750467777676986240647383340727022783744130281564027777290191531357426359782604 7
834369935906605500935555353972481294766681454045567488260563128055554580383062714852719565209 5
166873987181321100187111070794496258953336290809113497652112625611110916076612542970543913041 91
333747974362642200374222141588992517906672581618226995304225251222221832153225085941087826083 83
667495948725284400748444283177985035813345163236453990608450502444443664306450171882175652167 67
133499189745056880149688856635597007162669032647290798121690100488888732861290034376435130433535
266998379490113760299377713271194014325338065294581596243380200977777465722580068752870260867071
533996758980227520598755426542388028650676130589163192486760401955554931445160137505740521734143
106799351796045504119751085308477605730135226117832638497352080391110986289032027501148104 3468287
213598703592091008239502170616955211460270452235665276994704160782221972578064055002296208 6936575
427197407184182016479004341233910422920540904471330553989408321564443945156128110004592417 3873151
854394814368364032958008682467820845841081808942661107978816643128887890312256220009184834 7746303
170878962873672806591601736493564169168216361788532221595763328625777578062451244001836966954926 07
341757925747345613183203472987128338336432723577064443191526657251555156124902488003673933909852 15
683515851494691226366406945974256676672865447154128886383053314503110312249804976007347867819704 31
136703170298938245273281389194851335334573089430825777276610662900622062449960995201469573563940863
273406340597876490546562778389702670669146178861651554553221325801244124899921990402939147127881727
546812681195752981093125556779405341338292357723303109106442651602488249799843980805878294255763455
109362536239150596218625111355881068267658471544660621821288530320497649959968796161175658851152691 1
218725072478301192437250222711762136535316943089321243642577060640995299919937592322351317702305382 3
437450144956602384874500445423524273070633886178642487285154121281990599839875184644702635404610764 7
```

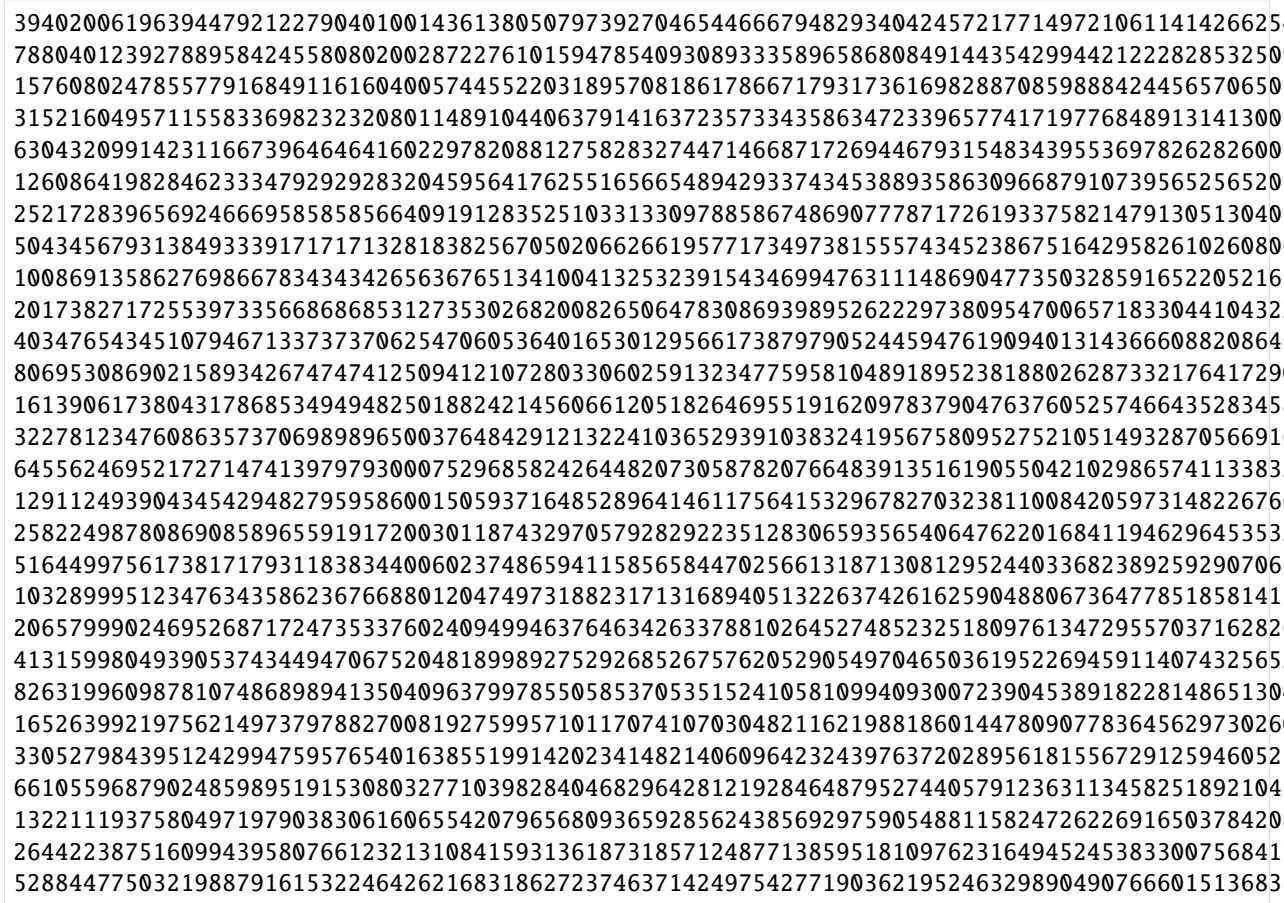
(continues on next page)

(continued from previous page)

```
87490028991320476974900089084704854614126777235728497457030824256398119967975036928940527080922152951749800579826409539498001781694097092282535544714569949140616485127962399359500738578810541618443059134996011596528190789960035633881941845650710894291398982812329702559247987190014771576210832368861183699920231930563815799200712677638836913014217885827979656246594051184959743800295431524216647377223671399840463861127631598401425355277673826028435771655959312493188102369919487600590863048433294754447352796809277222552631968028507105553476520568715433119186249863762047398389752011817260968665895088947155993618554445105263936057014211106953041137430866238372499727524094796779504023634521937331790177894311198723710889021052787211402842221390608227486173247674499945504818959355900804726904387466358035578872239744742177804210557442280568444278121645497234649534899989100963791871180160945380877493271607115775447948948435560842111488456113688855624329099446929906979997820192758374236032189076175498654321423155189589789687112168422297691222737771124865819889385981395999564038551674847206437815235099730864284631031791795793742243368445953824454755422497316397787719627919991280771033496944128756304701994617285692620135835915874844867368919076489095108449946327955754392558399825615420669938882575126094039892345713852411671831749689734737838152978190216899892655911508785116799651230841339877765150252188079784691427704831433436634993794694756763059563804337997853118230175702335993024616826797555303005043761595693828554096028668732699875893895135261191276086759957062364603514046719860492336535951106060100875231913876571081932573374653997517877902705223825521735199141247292070280934397209846730719022121202017504638277531421638611467493079950357558054104476510434703982824945841405618687944196934614380442424040350092765550628432771229349861599007151161082089530208694079656498916828112373758883938692287608848480807001855311012568655404586997231980143023221641790604173881593129978336562247475177678773845752176969616140037106220251373109291739944639602860464432835812083477631862599566731244949503553575476915043539392322800742124405027462184183479889279205720928865671624166955263725199133462489899007107150953830087078784645601484248810054924303669597785584114418577313432483339105274503982669249797980142143019076601741575692912029684976201098487173391955711682288371546268649666782105490079653384995959602842860381532034831513858240593699524021969747146783911423364576743092537299333564210980159306769991919205685720763064069663027716481187399048043939492935678228467291534861850745986671284219603186135399838384113714415261281393260554329623747980960878789958713564569345830697237014919733425684392063722707996767682274288305225627865211086592474959619217575798117427129138691661394474029839466851368784127445415993535364548576610451255730422173184949919238435151592348542582773833227889480596789337027375682548908319870707290971532209025114608443463698998384768703031946970851655476664557789611935786740547513650978166397414145819430644180502292168869273979967695374060638939417033109533291155792238715734810950273019563327948282916388612883610045843377385479599353907481212771878834066219066582311584477431469621900546039126655896565832777225767220091686754770959198707814962425237566813243813316462316895486293924380109207825331179313166555445153444018337350954191839741562992485107515336264876266329246337909725878487602184156506623586263331108903068880366747019083836794831259849702150306725297525326584926758194517569752043683130132471725266622178061377607334940381676735896625196994043006134505950506531698535163890351395040873662602649434505332443561227552146698807633534717932503939880860122690119010130633970703277807027900817473252052988690106648871224551042933976152670694358650078797617120245380238020261267941406555614055801634946504105977380213297742449102085867952305341388717300157595212404907604760405225358828131112281116032698930082119547604265954848982041717359046106827774346003151904148098152095208104507176562622245622320653978601642390952085319096979640834347180922136555486920063038094961963041904162090143531252444912446413079572032847819041706381939592816686943618442731109738401260761881923926083808324180287062504889824892826159144065695638083412763879185633373887236885462219476802521523138478521676166483605741250097796497856523182881313912761668255277583712667477744737709244389536050430475769570433523329672114825001955929957130463657626278255233365105551674253349554894754184887790721008609501539140867046659344229650003911859914260927315252556510466730211103348506699109789508369775581442017219030782817340933186884593000078237198285218546305051130209334604222066970133982195790167395511628840344380615656346818663737691860001564743965704370926101022604186692084441339402679643915803347910232576806887601231312693637327475383720003129487931408741852202045208373384168882678805359287831606695820465153613775224626253872746549507674400062589758628174837044040904167467683377653576107185756632133916409303072275504925250774549309901534880012517951725634967408808180833493536675530715221437151326426783281860614455100898505015490986198030697600250359034512699348176163616669870733510614304428743026528535665637212289102010197010030981972396061395200500718069025398696352327233397414670212286088574860530570713312744245782040
```

(continues on next page)

(continued from previous page)



(continues on next page)

(continued from previous page)

```
17745086042373215101301850778515735701993197282405226081091069315933576369956003987455836199066493299821
35490172084746430202603701557031471403986394564810452162182138631867152739912007974911672398132986599640
70980344169492860405207403114062942807972789129620904324364277263734305479824015949823344796265973199291
14196068833898572081041480622812588561594557825924180864872855452746861095964803189964668959253194639858
28392137667797144162082961245625177123189115651848361729745710905493722191929606379929337918506389279711
56784275335594288324165922491250354246378231303696723459491421810987444383859212759858675837012778559434
11356855067118857664833184498250070849275646260739344691898284362197488876771842551971735167402555711880
22713710134237715329666368996500141698551292521478689383796568724394977753543685103943470334805111423771
45427420268475430659332737993000283397102585042957378767593137448789955507087370207886940669610222847541
90854840536950861318665475986000566794205170085914757535186274897579911014174740415773881339220445695095
18170968107390172263733095197200113358841034017182951507037254979515982202834948083154776267844089139019
36341936214780344527466190394400226717682068034365903014074509959031964405669896166309552535688178278038
72683872429560689054932380788800453435364136068731806028149019918063928811339792332619105071376356556070
14536774485912137810986476157760090687072827213746361205629803983612785762267958466523821014275271311215
29073548971824275621972952315520181374145654427492722411259607967225571524535916933047642028550542622430
58147097943648551243945904631040362748291308854985444822519215934451143049071833866095284057101085244861
11629419588729710248789180926208072549658261770997088964503843186890228609814366773219056811420217048972
23258839177459420497578361852416145099316523541994177929007686373780457219628733546438113622840434097944
46517678354918840995156723704832290198633047083988355858015372747560914439257467092876227245680868195888
93035356709837681990313447409664580397266094167976711716030745495121828878514934185752454491361736391777
18607071341967536398062689481932916079453218833595342343206149099024365775702986837150490898272347278355
37214142683935072796125378963865832158906437667190684686412298198048731551405973674300981796544694556711
74428285367870145592250757927731664317812875334381369372824596396097463102811947348601963593089389113422
14885657073574029118450151585546332863562575066876273874564919279219492620562389469720392718617877822684
29771314147148058236900303171092665727125150133752547749129838558438985241124778939440785437235755645368
59542628294296116473800606342185331454250300267505095498259677116877970482249557878881570874471511290737
11908525658859223294760121268437066290850060005350101909965193542337559409644991157577631417489430225814
23817051317718446589520242536874132581700120107002038199303870846751188192899823151552628349788604516295
47634102635436893179040485073748265163400240214004076398607741693502376385799646303105256699577209032590
95268205270873786358080970147496530326800480428008152797215483387004752771599292606210513399154418065180
19053641054174757271616194029499306065360096085601630559443096677400950554319858521242102679830883613030
38107282108349514543232388058998612130720192171203261118886193354801901108639717042484205359661767226072
76214564216699029086464776117997224261440384342406522237772386709603802217279434084968410719323534452144
15242912843339805817292955223599444852288076868481304447554477341920760443455886816993682143864706890428
30485825686679611634585910447198889704576153736962608895108954683841520886911773633987364287729413780857
60971651373359223269171820894397779409152307473925217790217909367683041773823547267974728575458827561715
12194330274671844653834364178879555881830461494785043558043581873536608354764709453594957150917655123413
24388660549343689307668728357759111763660922989570087116087163747073216709529418907189891430183531024680
48777321098687378615337456715518223527321845979140174232174327494146433419058837814379782860367062049372
97554642197374757230674913431036447054643691958280348464348654988292866838117675628759565720734124098744
19510928439474951446134982686207289410928738391656069692869730997658573367623535125751913144146824819748
39021856878949902892269965372414578821857476783312139385739461995317146735247070251503826288293649639497
78043713757899805784539930744829157643714953566624278771478923990634293470494140503007652576587299278995
15608742751579961156907986148965831528742990713324855754295784798126858694098828100601530515317459855799
31217485503159922313815972297931663057485981426649711508591569596253717388197656201203061030634919711598
62434971006319844627631944595863326114971962853299423017183139192507434776395312402406122061269839423190
12486994201263968925526388919172665222994392570659884603436627838501486955279062480481224412253967884639
24973988402527937851052777838345330445988785141319769206873255677002973910558124960962448824507935769278
49947976805055875702105555676690660891977570282639538413746511354005947821116249921924897649015871538557
99895953610111751404211111353381321783955140565279076827493022708011895642232499843849795298031743077114
19979190722022350280842222270676264356791028113055815365498604541602379128446499968769959059606348615422
39958381444044700561684444541352528713582056226111630730997209083204758256892999937539918119212697230845
```

(continues on next page)

(continued from previous page)

```
799167628880894011233688890827050574271641124522232614619944181664095165137859998750798362384253944616
159833525776178802246737778165410114854328224904446522923988836332819033027571999750159672476850788923
319667051552357604493475556330820229708656449808893045847977672665638066055143999500319344953701577846
639334103104715208986951112661640459417312899617786091695955345331276132110287999000638689907403155693
127866820620943041797390222532328091883462579923557218339191069066255226422057599800127737981480631138
255733641241886083594780445064656183766925159847114436678382138132510452844115199600255475962961262274
511467282483772167189560890129312367533850319694228873356764276265020905688230399200510951925922524554
102293456496754433437912178025862473506770063938845774671352855253004181137646079840102190385184504910
204586912993508866875824356051724947013540127877691549342705710506008362275292159680204380770369009821
409173825987017733751648712103449894027080255755383098685411421012016724550584319360408761540738019643
818347651974035467503297424206899788054160511510766197370822842024033449101168638720817523081476039287
163669530394807093500659484841379957610832102302153239474164568404806689820233727744163504616295207857
```

1.3.4 Markdown Cells

Text can be added to Jupyter Notebooks using Markdown cells. You can change the cell type to Markdown by using the `Cell` menu, the toolbar, or the key shortcut `m`. Markdown is a popular markup language that is a superset of HTML. Its specification can be found here:

https://daringfireball.net/projects/markdown/

Markdown basics

You can make text *italic* or **bold** by surrounding a block of text with a single or double * respectively

You can build nested itemized or enumerated lists:

- One
  - Sublist
    - This
- Sublist - That - The other thing
- Two
- Sublist
- Three
- Sublist

Now another list:

1. Here we go
   1. Sublist
   2. Sublist
2. There we go
3. Now this

You can add horizontal rules:

Here is a blockquote:

> Beautiful is better than ugly. Explicit is better than implicit. Simple is better than complex. Complex is better than complicated. Flat is better than nested. Sparse is better than dense. Readability counts. Special cases aren't special enough to break the rules. Although practicality beats purity. Errors should never pass silently. Unless explicitly silenced. In the face of ambiguity, refuse the temptation to guess. There should be one– and preferably only one –obvious way to do it. Although that way may not be obvious at first unless you're Dutch. Now is better than never. Although never is often better than *right* now. If the implementation is hard to explain, it's a bad idea. If the implementation is easy to explain, it may be a good idea. Namespaces are one honking great idea – let's do more of those!

And shorthand for links:

Jupyter's website

You can use backslash to generate literal characters which would otherwise have special meaning in the Markdown syntax.

```
\*literal asterisks\*
 *literal asterisks*
```

Use double backslash to generate the literal $ symbol.

Headings

You can add headings by starting a line with one (or multiple) # followed by a space, as in the following example:

```
# Heading 1
# Heading 2
## Heading 2.1
## Heading 2.2
```

Embedded code

You can embed code meant for illustration instead of execution in Python:

```
def f(x):
    """a docstring"""
    return x**2
```

or other languages:

```
for (i=0; i<n; i++) {
  printf("hello %d\n", i);
  x += 4;
}
```

LaTeX equations

Courtesy of MathJax, you can include mathematical expressions both inline: $e^{i\pi} + 1 = 0$ and displayed:

$$e^x = \sum_{i=0}^{\infty} \frac{1}{i!} x^i \tag{1.1}$$

Inline expressions can be added by surrounding the latex code with `$`:

```
$e^{i\pi} + 1 = 0$
```

Expressions on their own line are surrounded by `\begin{equation}` and `\end{equation}`:

```
\begin{equation}
e^x=\sum_{i=0}^\infty \frac{1}{i!}x^i
\end{equation}
```

GitHub flavored markdown

The Notebook webapp supports Github flavored markdown meaning that you can use triple backticks for code blocks:

````
```python
print "Hello World"
```

```javascript
console.log("Hello World")
```
````

Gives:

```
print "Hello World"
```

```
console.log("Hello World")
```

And a table like this:

```
| This | is   |
|------|------|
|   a  | table|
```

A nice HTML Table:

This	is
a	table

General HTML

Because Markdown is a superset of HTML you can even add things like HTML tables:

Header 1

Header 2

row 1, cell 1

row 1, cell 2

row 2, cell 1

row 2, cell 2

Local files

If you have local files in your Notebook directory, you can refer to these files in Markdown cells directly:

```
[subdirectory/]<filename>
```

For example, in the images folder, we have the Python logo:

```
<img src="../images/python_logo.svg" />
```

and a video with the HTML5 video tag:

```
<video controls src="../images/animation.m4v">animation</video>
```

animation

These do not embed the data into the notebook file, and require that the files exist when you are viewing the notebook.

Security of local files

Note that this means that the Jupyter notebook server also acts as a generic file server for files inside the same tree as your notebooks. Access is not granted outside the notebook folder so you have strict control over what files are visible, but for this reason it is highly recommended that you do not run the notebook server with a notebook directory at a high level in your filesystem (e.g. your home directory).

When you run the notebook in a password-protected manner, local file access is restricted to authenticated users unless read-only views are active.

Markdown attachments

Since Jupyter notebook version 5.0, in addition to referencing external file you can attach a file to a markdown cell. To do so drag the file from in a markdown cell while editing it:

Files are stored in cell metadata and will be automatically scrubbed at save-time if not referenced. You can recognized attached images from other files by their url that starts with `attachment:`. For the image above:

```
![pycon-logo.jpg](attachment:pycon-logo.jpg)
```

Keep in mind that attached files will increase the size of your notebook.

You can manually edit the attachment by using the `View > Cell Toolbar > Attachment` menu, but you should not need to.

1.3.5 Keyboard Shortcut Customization

You can customize the `command` mode shortcuts from within the Notebook Application itself.

Head to the **Settings** menu and select the **Settings Editor** item. A dialog will guide you through the process of adding custom keyboard shortcuts.

Keyboard shortcut set from within the Notebook Application will be persisted to your configuration file. A single action may have several shortcuts attached to it.

1.3.6 Importing Jupyter Notebooks as Modules

It is a common problem that people want to import code from Jupyter Notebooks. This is made difficult by the fact that Notebooks are not plain Python files, and thus cannot be imported by the regular Python machinery.

Fortunately, Python provides some fairly sophisticated hooks into the import machinery, so we can actually make Jupyter notebooks importable without much difficulty, and only using public APIs.

[ ]:
```
import io, os, sys, types
```

[ ]:
```
from IPython import get_ipython
from nbformat import read
from IPython.core.interactiveshell import InteractiveShell
```

Import hooks typically take the form of two objects:

1. a Module **Loader**, which takes a module name (e.g. `'IPython.display'`), and returns a Module
2. a Module **Finder**, which figures out whether a module might exist, and tells Python what **Loader** to use

[ ]:
```
def find_notebook(fullname, path=None):
    """find a notebook, given its fully qualified name and an optional path

    This turns "foo.bar" into "foo/bar.ipynb"
    and tries turning "Foo_Bar" into "Foo Bar" if Foo_Bar
    does not exist.
    """
    name = fullname.rsplit('.', 1)[-1]
    if not path:
        path = ['']
    for d in path:
        nb_path = os.path.join(d, name + ".ipynb")
        if os.path.isfile(nb_path):
            return nb_path
        # let import Notebook_Name find "Notebook Name.ipynb"
        nb_path = nb_path.replace("_", " ")
        if os.path.isfile(nb_path):
            return nb_path
```

Notebook Loader

Here we have our Notebook Loader. It's actually quite simple - once we figure out the filename of the module, all it does is:

1. load the notebook document into memory
2. create an empty Module
3. execute every cell in the Module namespace

Since IPython cells can have extended syntax, the IPython transform is applied to turn each of these cells into their pure-Python counterparts before executing them. If all of your notebook cells are pure-Python, this step is unnecessary.

[ ]:
```
class NotebookLoader(object):
    """Module Loader for Jupyter Notebooks"""
    def __init__(self, path=None):
```

(continues on next page)

(continued from previous page)

```
        self.shell = InteractiveShell.instance()
        self.path = path

    def load_module(self, fullname):
        """import a notebook as a module"""
        path = find_notebook(fullname, self.path)

        print ("importing Jupyter notebook from %s" % path)

        # load the notebook object
        with io.open(path, 'r', encoding='utf-8') as f:
            nb = read(f, 4)


        # create the module and add it to sys.modules
        # if name in sys.modules:
        #    return sys.modules[name]
        mod = types.ModuleType(fullname)
        mod.__file__ = path
        mod.__loader__ = self
        mod.__dict__['get_ipython'] = get_ipython
        sys.modules[fullname] = mod

        # extra work to ensure that magics that would affect the user_ns
        # actually affect the notebook module's ns
        save_user_ns = self.shell.user_ns
        self.shell.user_ns = mod.__dict__

        try:
          for cell in nb.cells:
            if cell.cell_type == 'code':
                # transform the input to executable Python
                code = self.shell.input_transformer_manager.transform_cell(cell.source)
                # run the code in themodule
                exec(code, mod.__dict__)
        finally:
            self.shell.user_ns = save_user_ns
        return mod
```

The Module Finder

The finder is a simple object that tells you whether a name can be imported, and returns the appropriate loader. All this one does is check, when you do:

```
import mynotebook
```

it checks whether `mynotebook.ipynb` exists. If a notebook is found, then it returns a NotebookLoader.

Any extra logic is just for resolving paths within packages.

[ ]:
```
class NotebookFinder(object):
    """Module finder that locates Jupyter Notebooks"""
    def __init__(self):
        self.loaders = {}

    def find_module(self, fullname, path=None):
        nb_path = find_notebook(fullname, path)
        if not nb_path:
            return

        key = path
        if path:
            # lists aren't hashable
            key = os.path.sep.join(path)

        if key not in self.loaders:
            self.loaders[key] = NotebookLoader(path)
        return self.loaders[key]
```

Register the hook

Now we register the `NotebookFinder` with `sys.meta_path`

[ ]:
```
sys.meta_path.append(NotebookFinder())
```

After this point, my notebooks should be importable.

Let's look at what we have in the CWD:

[ ]:
```
ls nbpackage
```

So I should be able to `import nbpackage.mynotebook`.

[ ]:
```
import nbpackage.mynotebook
```

Aside: displaying notebooks

Here is some simple code to display the contents of a notebook with syntax highlighting, etc.

[ ]:
```
from pygments import highlight
from pygments.lexers import PythonLexer
from pygments.formatters import HtmlFormatter

from IPython.display import display, HTML

formatter = HtmlFormatter()
lexer = PythonLexer()

# publish the CSS for pygments highlighting
display(HTML("""
```

(continues on next page)

(continued from previous page)

```
<style type='text/css'>
%s
</style>
""" % formatter.get_style_defs()
))
```

[ ]:
```
def show_notebook(fname):
    """display a short summary of the cells of a notebook"""
    with io.open(fname, 'r', encoding='utf-8') as f:
        nb = read(f, 4)
    html = []
    for cell in nb.cells:
        html.append("<h4>%s cell</h4>" % cell.cell_type)
        if cell.cell_type == 'code':
            html.append(highlight(cell.source, lexer, formatter))
        else:
            html.append("<pre>%s</pre>" % cell.source)
    display(HTML('\n'.join(html)))

show_notebook(os.path.join("nbpackage", "mynotebook.ipynb"))
```

So my notebook has some code cells, one of which contains some IPython syntax.

Let's see what happens when we import it

[ ]:
```
from nbpackage import mynotebook
```

Hooray, it imported! Does it work?

[ ]:
```
mynotebook.foo()
```

Hooray again!

Even the function that contains IPython syntax works:

[ ]:
```
mynotebook.has_ip_syntax()
```

Notebooks in packages

We also have a notebook inside the `nb` package, so let's make sure that works as well.

[ ]:
```
ls nbpackage/nbs
```

Note that the `__init__.py` is necessary for `nb` to be considered a package, just like usual.

[ ]:
```
show_notebook(os.path.join("nbpackage", "nbs", "other.ipynb"))
```

[ ]:
```
from nbpackage.nbs import other
other.bar(5)
```

So now we have importable notebooks, from both the local directory and inside packages.

I can even put a notebook inside IPython, to further demonstrate that this is working properly:

[ ]:
```
import shutil
from IPython.paths import get_ipython_package_dir

utils = os.path.join(get_ipython_package_dir(), 'utils')
shutil.copy(os.path.join("nbpackage", "mynotebook.ipynb"),
            os.path.join(utils, "inside_ipython.ipynb")
)
```

and import the notebook from `IPython.utils`

[ ]:
```
from IPython.utils import inside_ipython
inside_ipython.whatsmyname()
```

This approach can even import functions and classes that are defined in a notebook using the `%%cython` magic.

1.3.7 Connecting to an existing IPython kernel using the Qt Console

The Frontend/Kernel Model

The traditional IPython (`ipython`) consists of a single process that combines a terminal based UI with the process that runs the users code.

While this traditional application still exists, the modern Jupyter consists of two processes:

- Kernel: this is the process that runs the users code.
- Frontend: this is the process that provides the user interface where the user types code and sees results.

Jupyter currently has 3 frontends:

- Terminal Console (`jupyter console`)
- Qt Console (`jupyter qtconsole`)
- Notebook (`jupyter notebook`)

The Kernel and Frontend communicate over a ZeroMQ/JSON based messaging protocol, which allows multiple Frontends (even of different types) to communicate with a single Kernel. This opens the door for all sorts of interesting things, such as connecting a Console or Qt Console to a Notebook's Kernel. For example, you may want to connect a Qt console to your Notebook's Kernel and use it as a help browser, calling `??` on objects in the Qt console (whose pager is more flexible than the one in the notebook).

This Notebook describes how you would connect another Frontend to an IPython Kernel that is associated with a Notebook. The commands currently given here are specific to the IPython kernel.

Manual connection

To connect another Frontend to a Kernel manually, you first need to find out the connection information for the Kernel using the `%connect_info` magic:

[ ]:
```
%connect_info
```

You can see that this magic displays everything you need to connect to this Notebook's Kernel.

Automatic connection using a new Qt Console

You can also start a new Qt Console connected to your current Kernel by using the `%qtconsole` magic. This will detect the necessary connection information and start the Qt Console for you automatically.

[ ]:
```
a = 10
```

[ ]:
```
%qtconsole
```

The Markdown parser included in the Jupyter Notebook is MathJax-aware. This means that you can freely mix in mathematical expressions using the MathJax subset of Tex and LaTeX. Some examples from the MathJax demos site are reproduced below, as well as the Markdown+TeX source.

1.3.8 Motivating Examples

The Lorenz Equations

Source

```
\begin{align}
\dot{x} & = \sigma(y-x) \\
\dot{y} & = \rho x - y - xz \\
\dot{z} & = -\beta z + xy
\end{align}
```

Display

The Cauchy-Schwarz Inequality

Source

```
\begin{equation*}
\left( \sum_{k=1}^n a_k b_k \right)^2 \leq \left( \sum_{k=1}^n a_k^2 \right) \left( \sum_
↪{k=1}^n b_k^2 \right)
\end{equation*}
```

Display

$$\left(\sum_{k=1}^{n} a_k b_k\right)^2 \leq \left(\sum_{k=1}^{n} a_k^2\right) \left(\sum_{k=1}^{n} b_k^2\right)$$

A Cross Product Formula

Source

```
\begin{equation*}
\mathbf{V}_1 \times \mathbf{V}_2 =  \begin{vmatrix}
\mathbf{i} & \mathbf{j} & \mathbf{k} \\
\frac{\partial X}{\partial u} &  \frac{\partial Y}{\partial u} & 0 \\
\frac{\partial X}{\partial v} &  \frac{\partial Y}{\partial v} & 0
\end{vmatrix}
\end{equation*}
```

Display

$$\mathbf{V}_1 \times \mathbf{V}_2 = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{\partial X}{\partial u} & \frac{\partial Y}{\partial u} & 0 \\ \frac{\partial X}{\partial v} & \frac{\partial Y}{\partial v} & 0 \end{vmatrix}$$

The probability of getting (k) heads when flipping (n) coins is

Source

```
\begin{equation*}
P(E)   = {n \choose k} p^k (1-p)^{ n-k}
\end{equation*}
```

Display

$$P(E) = {n \choose k} p^k (1-p)^{n-k}$$

An Identity of Ramanujan

Source

```
\begin{equation*}
\frac{1}{\Bigl(\sqrt{\phi \sqrt{5}}-\phi\Bigr) e^{\frac25 \pi}} =
1+\frac{e^{-2\pi}} {1+\frac{e^{-4\pi}} {1+\frac{e^{-6\pi}}
{1+\frac{e^{-8\pi}} {1+\ldots} } } }
\end{equation*}
```

Display

$$\frac{1}{\left(\sqrt{\phi\sqrt{5}}-\phi\right)e^{\frac{2}{5}\pi}} = 1+\frac{e^{-2\pi}}{1+\frac{e^{-4\pi}}{1+\frac{e^{-6\pi}}{1+\frac{e^{-8\pi}}{1+\cdots}}}}$$

A Rogers-Ramanujan Identity

Source

```
\begin{equation*}
1 +  \frac{q^2}{(1-q)}+\frac{q^6}{(1-q)(1-q^2)}+\cdots =
\prod_{j=0}^{\infty}\frac{1}{(1-q^{5j+2})(1-q^{5j+3})},
\quad\quad \text{for $|q|<1$}.
\end{equation*}
```

Display

$$1 + \frac{q^2}{(1-q)} + \frac{q^6}{(1-q)(1-q^2)} + \cdots = \prod_{j=0}^{\infty} \frac{1}{(1-q^{5j+2})(1-q^{5j+3})}, \qquad \text{for } |q| < 1.$$

Maxwell's Equations

Source

```
\begin{align}
\nabla \times \vec{\mathbf{B}} -\, \frac1c\, \frac{\partial\vec{\mathbf{E}}}{\partial t}␣
↪& = \frac{4\pi}{c}\vec{\mathbf{j}} \\   \nabla \cdot \vec{\mathbf{E}} & = 4 \pi \rho \\
\nabla \times \vec{\mathbf{E}}\, +\, \frac1c\, \frac{\partial\vec{\mathbf{B}}}{\partial␣
↪t} & = \vec{\mathbf{0}} \\
\nabla \cdot \vec{\mathbf{B}} & = 0
\end{align}
```

Display

Equation Numbering and References

Equation numbering and referencing will be available in a future version of the Jupyter notebook.

Inline Typesetting (Mixing Markdown and TeX)

While display equations look good for a page of samples, the ability to mix math and *formatted* **text** in a paragraph is also important.

Source

```
This expression $\sqrt{3x-1}+(1+x)^2$ is an example of a TeX inline equation in a␣
↪[Markdown-formatted](https://daringfireball.net/projects/markdown/) sentence.
```

Display

This expression $\sqrt{3x-1}+(1+x)^2$ is an example of a TeX inline equation in a Markdown-formatted sentence.

Other Syntax

You will notice in other places on the web that $$ are needed explicitly to begin and end MathJax typesetting. This is **not** required if you will be using TeX environments, but the Jupyter notebook will accept this syntax on legacy notebooks.

Source

```
$$
\begin{array}{c}
y_1 \\\
y_2 \mathtt{t}_i \\\
z_{3,4}
\end{array}
$$
```

```
$$
\begin{array}{c}
y_1 \cr
y_2 \mathtt{t}_i \cr
y_{3}
\end{array}
$$
```

```
$$\begin{eqnarray}
x' &=& &x \sin\phi &+& z \cos\phi \\
z' &=& - &x \cos\phi &+& z \sin\phi \\
\end{eqnarray}$$
```

```
$$
x=4
$$
```

Display

$$
\begin{aligned}
&y_1 \\
&y_2 \mathtt{t}_i \\
&z_{3,4}
\end{aligned}
$$

$$
\begin{aligned}
&y_1 \\
&y_2 \mathtt{t}_i \\
&y_3
\end{aligned}
$$

$$
\begin{aligned}
&to \\
\\
&x' = \\
&\quad x \sin \phi \\
&+z \cos \phi \\
&z' = \\
&-x \cos \phi \\
&+z \sin \phi
\end{aligned}
\tag{1.2}
$$

$$
\begin{aligned}
&= \\
\\
&\quad + \\
&- = \\
&+x \cos \phi \\
&\quad z \sin \phi
\end{aligned}
$$

$$x = 4$$

1.4 What to do when things go wrong

First, have a look at the common problems listed below. If you can figure it out from these notes, it will be quicker than asking for help.

Check that you have the latest version of any packages that look relevant. Unfortunately it's not always easy to figure out what packages are relevant, but if there was a bug that's already been fixed, it's easy to upgrade and get on with what you wanted to do.

1.4.1 Jupyter fails to start

- Have you installed it? ;-)
- If you're using a menu shortcut or Anaconda launcher to start it, try opening a terminal or command prompt and running the command `jupyter notebook`.
- If it can't find `jupyter`, you may need to configure your `PATH` environment variable. If you don't know what that means, and don't want to find out, just (re)install Anaconda with the default settings, and it should set up PATH correctly.
- If Jupyter gives an error that it can't find `notebook`, check with pip or conda that the `notebook` package is installed.
- Try running `jupyter-notebook` (with a hyphen). This should normally be the same as `jupyter notebook` (with a space), but if there's any difference, the version with the hyphen is the 'real' launcher, and the other one wraps that.

1.4.2 Jupyter doesn't load or doesn't work in the browser

- Try in another browser (e.g. if you normally use Firefox, try with Chrome). This helps pin down where the problem is.
- Try disabling any browser extensions and/or any Jupyter extensions you have installed.
- Some internet security software can interfere with Jupyter. If you have security software, try turning it off temporarily, and look in the settings for a more long-term solution.
- In the address bar, try changing between `localhost` and `127.0.0.1`. They should be the same, but in some cases it makes a difference.

1.4.3 Jupyter can't start a kernel

Files called *kernel specs* tell Jupyter how to start different kinds of kernels. To see where these are on your system, run `jupyter kernelspec list`:

```
$ jupyter kernelspec list
Available kernels:
  python3      /home/takluyver/.local/lib/python3.6/site-packages/ipykernel/resources
  bash         /home/takluyver/.local/share/jupyter/kernels/bash
  ir           /home/takluyver/.local/share/jupyter/kernels/ir
```

There's a special fallback for the Python kernel: if it doesn't find a real kernelspec, but it can import the `ipykernel` package, it provides a kernel which will run in the same Python environment as the notebook server. A path ending in `ipykernel/resources`, like in the example above, is this default kernel. The default often does what you want,

so if the `python3` kernelspec points somewhere else and you can't start a Python kernel, try deleting or renaming that kernelspec folder to expose the default.

If your problem is with another kernel, not the Python one we maintain, you may need to look for support about that kernel.

1.4.4 Python Environments

Multiple python environments, whether based on Anaconda or Python Virtual environments, are often the source of reported issues. In many cases, these issues stem from the Notebook server running in one environment, while the kernel and/or its resources, derive from another environment. Indicators of this scenario include:

- `import` statements within code cells producing `ImportError` or `ModuleNotFound` exceptions.
- General kernel startup failures exhibited by nothing happening when attempting to execute a cell.

In these situations, take a close look at your environment structure and ensure all packages required by your notebook's code are installed in the correct environment. If you need to run the kernel from different environments than your Notebook server, check out IPython's documentation for using kernels from different environments as this is the recommended approach. Anaconda's nb_conda_kernels package might also be an option for you in these scenarios.

Another thing to check is the `kernel.json` file that will be located in the aforementioned *kernel specs* directory identified by running `jupyter kernelspec list`. This file will contain an `argv` stanza that includes the actual command to run when launching the kernel. Oftentimes, when reinstalling python environments, a previous `kernel.json` will reference an python executable from an old or non-existent location. As a result, it's always a good idea when encountering kernel startup issues to validate the `argv` stanza to ensure all file references exist and are appropriate.

1.4.5 Windows Systems

Although Jupyter Notebook is primarily developed on the various flavors of the Unix operating system it also supports Microsoft Windows - which introduces its own set of commonly encountered issues, particularly in the areas of security, process management and lower-level libraries.

pywin32 Issues

The primary package for interacting with Windows' primitives is `pywin32`.

- Issues surrounding the creation of the kernel's communication file utilize `jupyter_core`'s `secure_write()` function. This function ensures a file is created in which only the owner of the file has access. If libraries like `pywin32` are not properly installed, issues can arise when it's necessary to use the native Windows libraries.

  Here's a portion of such a traceback:

```
File "c:\users\jovyan\python\myenv.venv\lib\site-packages\jupyter_core\paths.py",␣
↪line 424, in secure_write
win32_restrict_file_to_user(fname)
File "c:\users\jovyan\python\myenv.venv\lib\site-packages\jupyter_core\paths.py",␣
↪line 359, in win32_restrict_file_to_user
import win32api
ImportError: DLL load failed: The specified module could not be found.
```

- As noted earlier, the installation of `pywin32` can be problematic on Windows configurations. When such an issue occurs, you may need to revisit how the environment was setup. Pay careful attention to whether you're running the 32 or 64 bit versions of Windows and be sure to install appropriate packages for that environment.

  Here's a portion of such a traceback:

```
File "C:\Users\jovyan\AppData\Roaming\Python\Python37\site-packages\jupyter_core\
↪paths.py", line 435, in secure_write
win32_restrict_file_to_user(fname)
File "C:\Users\jovyan\AppData\Roaming\Python\Python37\site-packages\jupyter_core\
↪paths.py", line 361, in win32_restrict_file_to_user
import win32api
ImportError: DLL load failed: %1 is not a valid Win32 application
```

Resolving pywin32 Issues

In this case, your `pywin32` module may not be installed correctly and the following should be attempted:

```
pip install --upgrade pywin32
```

or:

```
conda install --force-reinstall pywin32
```

followed by:

```
python.exe Scripts/pywin32_postinstall.py -install
```

where `Scripts` is located in the active Python's installation location.

- Another common failure specific to Windows environments is the location of various python commands. On *`nix` systems, these typically reside in the `bin` directory of the active Python environment. However, on Windows, these tend to reside in the `Scripts` folder - which is a sibling to `bin`. As a result, when encountering kernel startup issues, again, check the `argv` stanza and verify it's pointing to a valid file. You may find that it's pointing in `bin` when `Scripts` is correct, or the referenced file does not include its `.exe` extension - typically resulting in `FileNotFoundError` exceptions.

1.4.6 This Worked An Hour Ago

The Jupyter stack is very complex and rightfully so, there's a lot going on. On occasion you might find the system working perfectly well, then, suddenly, you can't get past a certain cell due to `import` failures. In these situations, it's best to ask yourself if any new python files were added to your notebook development area.

These issues are usually evident by carefully analyzing the traceback produced in the notebook error or the Notebook server's command window. In these cases, you'll typically find the Python kernel code (from `IPython` and `ipykernel`) performing *its* imports and notice a file from your Notebook development error included in that traceback followed by an `AttributeError`:

```
File "C:\Users\jovyan\anaconda3\lib\site-packages\ipykernel\connect.py", line 13, in
from IPython.core.profiledir import ProfileDir
File "C:\Users\jovyan\anaconda3\lib\site-packages\IPython_init.py", line 55, in
from .core.application import Application
...
File "C:\Users\jovyan\anaconda3\lib\site-packages\ipython_genutils\path.py", line 13, in
import random
File "C:\Users\jovyan\Desktop\Notebooks\random.py", line 4, in
rand_set = random.sample(english_words_lower_set, 12)
AttributeError: module 'random' has no attribute 'sample'
```

What has happened is that you have named a file that conflicts with an installed package that is used by the kernel software and now introduces a conflict preventing the kernel's startup.

Resolution: You'll need to rename your file. A best practice would be to prefix or *namespace* your files so as not to conflict with any python package.

1.4.7 Asking for help

As with any problem, try searching to see if someone has already found an answer. If you can't find an existing answer, you can ask questions at:

- The Jupyter Discourse Forum
- The jupyter-notebook tag on Stackoverflow
- Peruse the jupyter/help repository on Github (read-only)
- Or in an issue on another repository, if it's clear which component is responsible. Typical repositories include:
  - jupyter_core - `secure_write()` and file path issues
  - jupyter_client - kernel management issues found in Notebook server's command window.
  - IPython and ipykernel - kernel runtime issues typically found in Notebook server's command window and/or Notebook cell execution.

Gathering Information

Should you find that your problem warrants that an issue be opened in notebook please don't forget to provide details like the following:

- What error messages do you see (within your notebook and, more importantly, in the Notebook server's command window)?
- What platform are you on?
- How did you install Jupyter?
- What have you tried already?

The `jupyter troubleshoot` command collects a lot of information about your installation, which can also be useful.

When providing textual information, it's most helpful if you can *scrape* the contents into the issue rather than providing a screenshot. This enables others to select pieces of that content so they can search more efficiently and try to help.

Remember that it's not anyone's job to help you. We want Jupyter to work for you, but we can't always help everyone individually.

1.5 Changelog

A summary of changes in the Jupyter notebook. For more detailed information, see GitHub.

Use `pip install notebook --upgrade` or `conda upgrade notebook` to upgrade to the latest release.

We strongly recommend that you upgrade pip to version 9+ of pip before upgrading `notebook`.

Use `pip install pip --upgrade` to upgrade pip. Check pip version with `pip --version`.

1.5.1 7.0.0a5

(Full Changelog)

Enhancements made

- Add shadow to cells #6433 (@trungleduc)

Maintenance and upkeep improvements

- Fix docs build #6447 (@jtpio)
- [pre-commit.ci] pre-commit autoupdate #6444 (@pre-commit-ci)
- [pre-commit.ci] pre-commit autoupdate #6439 (@pre-commit-ci)
- [pre-commit.ci] pre-commit autoupdate #6434 (@pre-commit-ci)
- Use hatch backend #6425 (@blink1073)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @github-actions | @jtpio | @ofek | @pre-commit-ci | @trungleduc

1.5.2 7.0.0a4

(Full Changelog)

Enhancements made

- Update to JupyterLab 4.0.0a25 #6429 (@jtpio)
- Add cell toolbar extension #6418 (@jtpio)
- add the show header command to the command palette #6415 (@jeewonkoo)
- Remove the “Appearance” menu entry #6412 (@jtpio)

Maintenance and upkeep improvements

- [pre-commit.ci] pre-commit autoupdate #6426 (@pre-commit-ci)
- Allow bot PRs to be automatically labeled #6414 (@blink1073)
- Add Visual Regression Test for the export submenu #6383 (@jtpio)
- Fix ESLint config for tests #6382 (@jtpio)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @github-actions | @jeewonkoo | @jtpio | @ofek | @pre-commit-ci

1.5.3 7.0.0a3

(Full Changelog)

Enhancements made

- Show file checkboxes by default in the file browser #6377 (@jtpio)
- Remove blank space at the bottom of the notebook #6376 (@jtpio)
- Add a label to the upload button #6374 (@jtpio)
- Update to JupyterLab 4.0.0a24 #6371 (@jtpio)
- Initialize handler page_config from Server traitlets #6366 (@bollwyvl)
- Add project URLs to setup.cfg #6346 (@tlinhart)
- Update to JupyterLab 4.0.0a23 #6336 (@jtpio)

Bugs fixed

- Fix opening JupyterLab from Notebook #6379 (@jtpio)

Maintenance and upkeep improvements

- Add UI tests for a simple notebook #6380 (@jtpio)
- Bump async from 2.6.3 to 2.6.4 in /ui-tests #6370 (@dependabot)
- Fix build workflow on CI #6369 (@jtpio)
- [pre-commit.ci] pre-commit autoupdate #6365 (@pre-commit-ci)
- [pre-commit.ci] pre-commit autoupdate #6355 (@pre-commit-ci)
- Bump moment from 2.29.1 to 2.29.2 #6352 (@dependabot)
- Bump moment from 2.29.1 to 2.29.2 in /ui-tests #6351 (@dependabot)
- Clean up pre-commit #6349 (@blink1073)
- Update to TypeScript 4.6 #6345 (@jtpio)
- [pre-commit.ci] pre-commit autoupdate #6343 (@pre-commit-ci)
- Add pytest and handle warnings #6338 (@blink1073)
- Add flake8 and git-blame-ignore-revs #6337 (@blink1073)
- Run Autoformatters #6335 (@blink1073)
- Bump minimist from 1.2.5 to 1.2.6 #6334 (@dependabot)
- Bump minimist from 1.2.5 to 1.2.6 in /ui-tests #6333 (@dependabot)

- Bump ansi-regex from 3.0.0 to 3.0.1 in /ui-tests #6332 (@dependabot)
- Adopt pre-commit #6331 (@blink1073)

Documentation improvements

- Link to v6 plan in the README #6375 (@echarles)
- Copy edits in config, edits bug report template #6364 (@jweill-aws)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @bollwyvl | @dependabot | @echarles | @github-actions | @jtpio | @jweill-aws | @pre-commit-ci | @tlinhart

1.5.4 7.0.0a2

(Full Changelog)

Enhancements made

- Add support for opening a document with a different factory #6315 (@jtpio)
- Minor copy edit in README #6313 (@jweill-aws)

Bugs fixed

- Fix rendering of markdown #6318 (@jtpio)

Maintenance and upkeep improvements

- Update to JupyterLab 4.0.0a22 #6314 (@jtpio)

Documentation improvements

- docs: fix spelling #6317 (@dijonkitchen)
- Minor copy edit in README #6313 (@jweill-aws)
- Update example notebook on Binder #6306 (@jtpio)

Contributors to this release

(GitHub contributors page for this release)

@dijonkitchen | @fcollonval | @github-actions | @jtpio | @jweill-aws

1.5.5 7.0.0a1

(Full Changelog)

Enhancements made

- Notebook v7 scaffolding #6294 (@jtpio)

Maintenance and upkeep improvements

- Clean up CI #6304 (@blink1073)
- Remove the custom run keyboard shortcut #6303 (@jtpio)
- Bump nanoid from 3.1.30 to 3.3.1 in /ui-tests #6302 (@dependabot)
- Bump simple-get from 3.1.0 to 3.1.1 in /ui-tests #6301 (@dependabot)
- Bump url-parse from 1.5.4 to 1.5.10 in /ui-tests #6300 (@dependabot)
- Bump node-fetch from 2.6.6 to 2.6.7 in /ui-tests #6299 (@dependabot)
- Bump follow-redirects from 1.14.5 to 1.14.9 in /ui-tests #6298 (@dependabot)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @dependabot | @github-actions | @jtpio | @kevin-bates | @Zsailer

1.5.6 6.4.8

(Full Changelog)

Bugs fixed

- Fix to remove potential memory leak on Jupyter Notebooks ZMQChannelHandler code #6251 (@Vishwajeet0510)

Contributors to this release

(GitHub contributors page for this release)

@Vishwajeet0510

1.5.7 6.4.7

(Full Changelog)

Bugs fixed

- Fix Chinese punctuation #6268 (@LiHua-Official)
- Add date field to kernel message header #6265 (@kevin-bates)
- Fix deprecation warning #6253 (@tornaria)

Maintenance and upkeep improvements

- Enforce labels on PRs #6235 (@blink1073)
- Fix: CI error for python 3.6 & macOS #6215 (@penguinolog)

Other merged PRs

- handle KeyError when get session #6245 (@ccw630)
- Updated doc for passwd #6209 (@antoinecarme)

Contributors to this release

(GitHub contributors page for this release)

@antoinecarme | @blink1073 | @ccw630 | @kevin-bates | @LiHua-Official | @penguinolog | @tornaria

1.5.8 6.4.6

(Full Changelog)

Bugs fixed

- Fix `asyncio` error when opening notebooks #6221 (@dleen)
- Change to use a universal Chinese translation on certain words #6218 (@jackexu)
- Fix Chinese translation typo #6211 (@maliubiao
- Fix `send2trash` tests failing on Windows #6127 (@dolfinus)

Maintenance and upkeep improvements

- TST: don't look in user site for serverextensions #6233 (@bnavigator)
- Enable terminal tests as `pywinpty` is ported for python 3.9 #6228 (@nsait-linaro)

Contributors to this release

(GitHub contributors page for this release)

@bnavigator | @dleen | @dolfinus | @jackexu | @kevin-bates | @maliubiao | @nsait-linaro | @takluyver | @Zsailer

1.5.9 6.4.5

(Full Changelog)

Bug fixes

- Recover from failure to render mimetype #6181 (@martinRenou)

Maintenance and upkeep improvements

- Fix crypto handling #6197 (@blink1073)
- Fix `jupyter_client` warning #6178 (@martinRenou)

Documentation improvements

- Fix nbsphinx settings #6200 (@mgeier)
- Fully revert the pinning of `nbsphinx` to 0.8.6 #6201 (@kevin-bates)
- Pin `nbsphinx` to 0.8.6, clean up orphaned resources #6194 (@kevin-bates)
- Fix typo in docstring #6188 (@jgarte)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @jgarte | @kevin-bates | @martinRenou | @mgeier

1.5.10 6.4.4

(Full Changelog)

Documentation improvements

- Update Manual Release Instructions #6152 (@blink1073)

Other merged PRs

- Use default JupyterLab CSS sanitizer options for Markdown #6160 (@krassowski)
- Fix syntax highlight #6128 (@massongit)

Contributors to this release

(GitHub contributors page for this release)

@blink1073 | @kevin-bates | @krassowski | @massongit | @minrk | @Zsailer

1.5.11 6.4.3

(Full Changelog)

Bugs fixed

- Add @babel/core dependency #6133 (@afshin)
- Switch webpack to production mode #6131 (@afshin)

Maintenance and upkeep improvements

- Clean up link checking #6130 (@blink1073)

Contributors to this release

(GitHub contributors page for this release)

@afshin | @blink1073 | @Zsailer

1.5.12 6.4.2

(Full Changelog)

Bugs fixed

- Add missing file to manifest #6122 (@afshin)
- Fix issue #3218 #6108 (@Nazeeh21)
- Fix version of jupyter-packaging in pyproject.toml #6101 (@frenzymadness)
- "#element".tooltip is not a function on home page fixed. #6070 @ilayh123

Maintenance and upkeep improvements

- Enhancements to the desktop entry #6099 (@Amr-Ibra)
- Add missing spaces to help messages in config file #6085 (@saiwing-yeung)

Contributors to this release

(GitHub contributors page for this release)

@afshin | @Amr-Ibra | @frenzymadness | @ilayh123 | @kevin-bates | @Nazeeh21 | @saiwing-yeung

1.5.13 6.4.0

(Full Changelog)

Bugs fixed

- Fix Handling of Encoded Paths in Save As Dialog #6030 (@afshin)
- Fix: split_cell doesn't always split cell #6017 (@gamestrRUS)
- Correct 'Content-Type' headers #6026 (@faucct)
- Fix skipped tests & remove deprecation warnings #6018 (@befeleme)
- [Gateway] Track only this server's kernels #5980 (@kevin-bates)
- Bind the HTTPServer in start #6061

Maintenance and upkeep improvements

- Revert "do not apply asyncio patch for tornado >=6.1" #6052 (@minrk)
- Use Jupyter Releaser #6048 (@afshin)
- Add Workflow Permissions for Lock Bot #6042 (@jtpio)
- Fixes related to the recent changes in the documentation #6021 (@frenzymadness)
- Add maths checks in CSS reference test #6035 (@stef4k)
- Add Issue Lock and Answered Bots #6019 (@afshin)

Documentation improvements

- Spelling correction #6045 (@wggillen)
- Minor typographical and comment changes #6025 (@misterhay)
- Fixes related to the recent changes in the documentation #6021 (@frenzymadness)
- Fix readthedocs environment #6020 (@blink1073)

Contributors to this release

(GitHub contributors page for this release)

@afshin | @befeleme | @blink1073 | @faucct | @frenzymadness | @gamestrRUS | @jtpio | @kevin-bates | @minrk | @misterhay | @stef4k | @wggillen

1.5.14 6.3.0

Merged PRs

- Add square logo and desktop entry files #6010 (@befeleme)
- Modernize Changelog #6008 (@afshin)
- Add missing "import inspect" #5999 (@mgeier)
- Add Codecov badge to README #5989 (@thomasrockhu)
- Remove configuration for nosetests from setup.cfg #5986 (@frenzymadness)
- Update security.rst #5978 (@dlrice)
- Docs-Translations: Updated Hindi and Chinese Readme.md #5976 (@rjn01)
- Allow /metrics by default if auth is off #5974 (@blairdrummond)
- Skip terminal tests on Windows 3.9+ (temporary) #5968 (@kevin-bates)
- Update GatewayKernelManager to derive from AsyncMappingKernelManager #5966 (@kevin-bates)
- Drop use of deprecated pyzmq.ioloop #5965 (@kevin-bates)
- Drop support for Python 3.5 #5962 (@kevin-bates)
- Allow jupyter_server-based contents managers in notebook #5957 (@afshin)
- Russian translation fixes #5954 (@insolor)
- Increase culling test idle timeout #5952 (@kevin-bates)
- Re-enable support for answer_yes flag #5941 (@afshin)
- Replace Travis and Appveyor with Github Actions #5938 (@kevin-bates)
- DOC: Server extension, extra docs on configuration/authentication. #5937 (@Carreau)

Contributors to this release

(GitHub contributors page for this release)

@abielhammonds | @afshin | @ajharry | @Alokrar | @befeleme | @blairdrummond | @blink1073 | @bollwyvl | @Carreau | @ChenChenDS | @cosmoscalibur | @dlrice | @dwanneruchi | @ElisonSherton | @FazeelUsmani | @frenzymadness | @goerz | @insolor | @jasongrout | @JianghuiDu | @JuzerShakir | @kevin-bates | @Khalilsqu | @meeseeksdev | @mgeier | @michaelpedota | @mjbright | @MSeal | @ncoughlin | @NTimmons | @ProsperousHeart | @rjn01 | @slw07g | @stenivan | @takluyver | @thomasrockhu | @wgilpin | @wxtt522 | @yuvipanda | @Zsailer

1.5.15 6.2.0

1.5.16 Merged PRs

- Increase minimum tornado version (5933)
- Adjust skip decorators to avoid remaining dependency on nose (5932)
- Ensure that cell ids persist after save (5928)
- Add reconnection to Gateway (form nb2kg) (5924)
- Fix some typos (5917)
- Handle TrashPermissionError, now that it exist (5894)

Thank you to all the contributors:

- @kevin-bates
- @mishaschwartz
- @oyvsyo
- @user202729
- @stefanor

1.5.17 6.1.6

1.5.18 Merged PRs

- do not require nose for testing (5826)
- [docs] Update Chinese and Hindi readme.md (5823)
- Add support for creating terminals via GET (5813)
- Made doc translations in Hindi and Chinese (5787)

Thank you to all the contributors:

- @pgajdos
- @rjn01
- @kevin-bates
- @virejdasani

1.5.19 6.1.5

6.1.5 is a security release, fixing one vulnerability:

- Fix open redirect vulnerability GHSA-c7vm-f5p4-8fqh (CVE to be assigned)

1.5.20 6.1.4

- Fix broken links to jupyter documentation (5686)
- Add additional entries to troubleshooting section (5695)
- Revert change in page alignment (5703)
- Bug fix: remove double encoding in download files (5720)
- Fix typo for Check in zh_CN (5730)
- Require a file name in the "Save As" dialog (5733)

Thank you to all the contributors:

- bdbai
- Jaipreet Singh
- Kevin Bates
- Pavel Panchekha
- Zach Sailer

1.5.21 6.1.3

- Title new buttons with label if action undefined (5676)

Thank you to all the contributors:

- Kyle Kelley

1.5.22 6.1.2

- Fix russian message format for delete/duplicate actions (5662)
- Remove unnecessary import of bind_unix_socket (5666)
- Tooltip style scope fix (5672)

Thank you to all the contributors:

- Dmitry Akatov
- Kevin Bates
- Magda Stenius

1.5.23 6.1.1

- Prevent inclusion of requests_unixsocket on Windows (5650)

Thank you to all the contributors:

- Kevin Bates

1.5.24 6.1.0

Please note that this repository is currently maintained by a skeleton crew of maintainers from the Jupyter community. For our approach moving forward, please see this notice from the README. Thank you.

Here is an enumeration of changes made since the last release and included in 6.1.0.

- Remove deprecated encoding parameter for Python 3.9 compatibility. (5174)
- Add support for async kernel management (4479)
- Fix typo in password_required help message (5320)
- Gateway only: Ensure launch and request timeouts are in sync (5317)
- Update Markdown Cells example to HTML5 video tag (5411)
- Integrated LoginWidget into edit to enable users to logout from the t… (5406)
- Update message about minimum Tornado version (5222)
- Logged notebook type (5425)
- Added nl language (5354)
- Add UNIX socket support to notebook server. (4835)
- Update CodeMirror dependency (5198)
- Tree added download multiple files (5351)
- Toolbar buttons tooltip: show help instead of label (5107)
- Remove unnecessary import of requests_unixsocket (5451)
- Add ability to cull terminals and track last activity (5372)
- Code refactoring notebook.js (5352)
- Install terminado for docs build (5462)
- Convert notifications JS test to selenium (5455)
- Add cell attachments to markdown example (5412)
- Add Japanese document (5231)
- Migrate Move multiselection test to selenium (5158)
- Use `cmdtrl-enter` to run a cell (5120)
- Fix broken “Raw cell MIME type” dialog (5385)
- Make a notebook writable after successful save-as (5296)
- Add actual watch script (4738)
- Added `--autoreload` flag to `NotebookApp` (4795)
- Enable check_origin on gateway websocket communication (5471)
- Restore detection of missing terminado package (5465)
- Culling: ensure `last_activity` attr exists before use (5355)
- Added functionality to allow filter kernels by Jupyter Enterprise Gat… (5484)
- ‘Play’ icon for run-cell toolbar button (2922)
- Bump minimum version of jQuery to 3.5.0 (5491)

- Remove old JS markdown tests, add a new one in selenium (5497)
- Add support for more RTL languages (5036)
- Make markdown cells stay RTL in edit mode (5037)
- Unforce RTL output display (5039)
- Fixed multicursor backspacing (4880)
- Implemented Split Cell for multicursor (4824)
- Alignment issue [FIXED] (3173)
- MathJax: Support for `\gdef` (4407)
- Another (Minor) Duplicate Code Reduction (5316)
- Update readme regarding maintenance (5500)
- Document contents chunks (5508)
- Backspace deletes empty line (5516)
- The dropdown submenu at notebook page is not keyboard accessible (4732)
- Tooltips visible through keyboard navigation for specified buttons (4729)
- Fix for recursive symlink (4670)
- Fix for the terminal shutdown issue (4180)
- Add japanese translation files (4490)
- Workaround for socket permission errors on Cygwin (4584)
- Implement optional markdown header and footer files (4043)
- Remove double link when using `custom_display_url` (5544)
- Respect `cell.is_editable` during find-and-replace (5545)
- Fix exception causes all over the codebase (5556
- Improve login shell heuristics (5588)
- Added support for `JUPYTER_TOKEN_FILE` (5587)
- Kill notebook itself when server cull idle kernel (5593)
- Implement password hashing with bcrypt (3793)
- Fix broken links (5600)
- Russian internationalization support (5571)
- Add a metadata tag to override notebook direction (ltr/rtl) (5052)
- Paste two images from clipboard in markdown cell (5598)
- Add keyboard shortcuts to menu dropdowns (5525)
- Update codemirror to `5.56.0+components1` (5637)

Thank you to all the contributors:

- Aaron Myatt
- Adam Blake
- Afshin Taylor Darian

- Aman Bansal
- Ben Thayer
- berendjan
- Bruno P. Kinoshita
- bzinberg
- Christophe Cadilhac
- Daiki Katsuragawa
- David Lukes
- Dmitriy Q
- dmpe
- dylanzjy
- dSchurch
- E. M. Bray
- ErwinRussel
- Felix Mönckemeyer
- Grant Nestor
- Jarrad Whitaker
- Jesus Panales Castillo
- Joshua Zeltser
- Karthikeyan Singaravelan
- Kenichi Ito
- Kevin Bates
- Koki Nishihara
- Kris Wilson
- Kyle Kelley
- Laura Merlo
- levinxo
- Luciano Resende
- Luis Cabezon Manchado
- Madhusudhan Srinivasa
- Matthias Geier
- mattn
- Max Klein
- Min RK
- Mingxuan Lin
- Mohammad Mostafa Farzan

- Niko Felger
- Norah Abanumay
- Onno Broekmans
- PierreMB
- pinarkavak
- Ram Rachum
- Reece Hart
- Remi Rampin
- Rohit Sanjay
- Shane Canon
- Simon Li
- Steinar Sturlaugsson
- Steven Silvester
- taohan16
- Thew Dhanat
- Thomas Kluyver
- Toon Baeyens
- Vidar Tonaas Fauske
- Zachary Sailer

1.5.25 6.0.3

- Dependency updates to fix startup issues on Windows platform
- Add support for nbconvert 6.x
- Creation of recent tab

Thanks for all the contributors:

- Luciano Resende
- Kevin Bates
- ahangsleben
- Zachary Sailer
- Pallavi Bharadwaj
- Thomas Kluyver
- Min RK
- forest0
- Bibo Hao
- Michal Charemza
- Sergey Shevelev

- Shuichiro MAKIGAKI
- krinsman
- TPartida
- Landen McDonald
- Tres DuBiel

1.5.26 6.0.2

- Update JQuery dependency to version 3.4.1 to fix security vulnerability (CVE-2019-11358)
- Update CodeMirror to version 5.48.4 to fix Python formatting issues
- Continue removing obsolete Python 2.x code/dependencies
- Multiple documentation updates

Thanks for all the contributors:

- David Robles
- Jason Grout
- Kerwin Sun
- Kevin Bates
- Kyle Kelley
- Luciano Resende
- Marcus D Sherman
- Sasaki Takeru
- Tom Jarosz
- Vidar Tonaas Fauske
- Wes Turner
- Zachary Sailer

1.5.27 6.0.1

- Attempt to re-establish websocket connection to Gateway (4777)
- Add missing react-dom js to package data (4772)

Thanks for all the contributors:

- Eunsoo Park
- Min RK

1.5.28 6.0

This is the first major release of the Jupyter Notebook since version 5.0 (March 2017).

We encourage users to start trying JupyterLab, which has just announced it's 1.0 release in preparation for a future transition.

- Remove Python 2.x support in favor of Python 3.5 and higher.
- Multiple accessibility enhancements and bug-fixes.
- Multiple translation enhancements and bug-fixes.
- Remove deprecated ANSI CSS styles.
- Native support to forward requests to Jupyter Gateway(s) (Embedded NB2KG).
- Use JavaScript to redirect users to notebook homepage.
- Enhanced SSL/TLS security by using PROTOCOL_TLS which selects the highest ssl/tls protocol version available that both the client and server support. When PROTOCOL_TLS is not available use PROTOCOL_SSLv23.
- Add `?no_track_activity=1` argument to allow API requests. to not be registered as activity (e.g. API calls by external activity monitors).
- Kernels shutting down due to an idle timeout is no longer considered an activity-updating event.
- Further improve compatibility with tornado 6 with improved checks for when websockets are closed.
- Launch the browser with a local file which redirects to the server address including the authentication token. This prevents another logged-in user from stealing the token from command line arguments and authenticating to the server. The single-use token previously used to mitigate this has been removed. Thanks to Dr. Owain Kenway for suggesting the local file approach.
- Respect nbconvert entrypoints as sources for exporters
- Update to CodeMirror to 5.37, which includes f-string syntax for Python 3.6.
- Update jquery-ui to 1.12
- Execute cells by clicking icon in input prompt.
- New "Save as" menu option.
- When serving on a loopback interface, protect against DNS rebinding by checking the `Host` header from the browser. This check can be disabled if necessary by setting `NotebookApp.allow_remote_access`. (Disabled by default while we work out some Mac issues in 3754).
- Add kernel_info_timeout traitlet to enable restarting slow kernels.
- Add `custom_display_host` config option to override displayed URL.
- Add /metrics endpoint for Prometheus Metrics.
- Optimize large file uploads.
- Allow access control headers to be overriden in jupyter_notebook_config.py to support greater CORS and proxy configuration flexibility.
- Add support for terminals on windows.
- Add a "restart and run all" button to the toolbar.
- Frontend/extension-config: allow default json files in a .d directory.
- Allow setting token via jupyter_token env.
- Cull idle kernels using `--MappingKernelManager.cull_idle_timeout`.

- Allow read-only notebooks to be trusted.
- Convert JS tests to Selenium.

Security Fixes included in previous minor releases of Jupyter Notebook and also included in version 6.0.

- Fix Open Redirect vulnerability (CVE-2019-10255) where certain malicious URLs could redirect from the Jupyter login page to a malicious site after a successful login.
- Contains a security fix for a cross-site inclusion (XSSI) vulnerability (CVE-2019–9644), where files at a known URL could be included in a page from an unauthorized website if the user is logged into a Jupyter server. The fix involves setting the `X-Content-Type-Options: nosniff` header, and applying CSRF checks previously on all non-GET API requests to GET requests to API endpoints and the /files/ endpoint.
- Check Host header to more securely protect localhost deployments from DNS rebinding. This is a pre-emptive measure, not fixing a known vulnerability. Use `.NotebookApp.allow_remote_access` and `.NotebookApp.local_hostnames` to configure access.
- Upgrade bootstrap to 3.4, fixing an XSS vulnerability, which has been assigned CVE-2018-14041.
- Contains a security fix preventing malicious directory names from being able to execute javascript.
- Contains a security fix preventing nbconvert endpoints from executing javascript with access to the server API. CVE request pending.

Thanks for all the contributors:

- AAYUSH SINHA
- Aaron Hall, MBA
- Abhinav Sagar
- Adam Rule
- Adeel Ahmad
- Alex Rothberg
- Amy Skerry-Ryan
- Anastasis Germanidis
- Andrés Sánchez
- Arjun Radhakrishna
- Arovit Narula
- Benda Xu
- Björn Grüning
- Brian E. Granger
- Carol Willing
- Celina Kilcrease
- Chris Holdgraf
- Chris Miller
- Ciaran Langton
- Damian Avila
- Dana Lee

- Daniel Farrell
- Daniel Nicolai
- Darío Hereñú
- Dave Aitken
- Dave Foster
- Dave Hirschfeld
- Denis Ledoux
- Dmitry Mikushin
- Dominic Kuang
- Douglas Hanley
- Elliott Sales de Andrade
- Emilio Talamante Lugo
- Eric Perry
- Ethan T. Hendrix
- Evan Van Dam
- Francesco Franchina
- Frédéric Chapoton
- Félix-Antoine Fortin
- Gabriel
- Gabriel Nützi
- Gabriel Ruiz
- Gestalt LUR
- Grant Nestor
- Gustavo Efeiche
- Harsh Vardhan
- Heng GAO
- Hisham Elsheshtawy
- Hong Xu
- Ian Rose
- Ivan Ogasawara
- J Forde
- Jason Grout
- Jessica B. Hamrick
- Jiaqi Liu
- John Emmons
- Josh Barnes

- Karthik Balakrishnan
- Kevin Bates
- Kirit Thadaka
- Kristian Gregorius Hustad
- Kyle Kelley
- Leo Gallucci
- Lilian Besson
- Lucas Seiki Oshiro
- Luciano Resende
- Luis Angel Rodriguez Guerrero
- M Pacer
- Maarten Breddels
- Mac Knight
- Madicken Munk
- Maitiú Ó Ciaráin
- Marc Udoff
- Mathis HAMMEL
- Mathis Rosenhauer
- Matthias Bussonnier
- Matthias Geier
- Max Vovshin
- Maxime Mouchet
- Michael Chirico
- Michael Droettboom
- Michael Heilman
- Michael Scott Cuthbert
- Michal Charemza
- Mike Boyle
- Milos Miljkovic
- Min RK
- Miro Hrončok
- Nicholas Bollweg
- Nitesh Sawant
- Ondrej Jariabka
- Park Hae Jin
- Paul Ivanov

- Paul Masson
- Peter Parente
- Pierre Tholoniat
- Remco Verhoef
- Roland Weber
- Roman Kornev
- Rosa Swaby
- Roy Hyunjin Han
- Sally
- Sam Lau
- Samar Sultan
- Shiti Saxena
- Simon Biggs
- Spencer Park
- Stephen Ward
- Steve (Gadget) Barnes
- Steven Silvester
- Surya Prakash Susarla
- Syed Shah
- Sylvain Corlay
- Thomas Aarholt
- Thomas Kluyver
- Tim
- Tim Head
- Tim Klever
- Tim Metzler
- Todd
- Tom Jorquera
- Tyler Makaro
- Vaibhav Sagar
- Victor
- Vidar Tonaas Fauske
- Vu Minh Tam
- Vít Tuček
- Will Costello
- Will Starms

- William Hosford
- Xiaohan Li
- Yuvi Panda
- ashley teoh
- nullptr

1.5.29 5.7.8

- Fix regression in restarting kernels in 5.7.5. The restart handler would return before restart was completed.
- Further improve compatibility with tornado 6 with improved checks for when websockets are closed.
- Fix regression in 5.7.6 on Windows where .js files could have the wrong mime-type.
- Fix Open Redirect vulnerability (CVE-2019-10255) where certain malicious URLs could redirect from the Jupyter login page to a malicious site after a successful login. 5.7.7 contained only a partial fix for this issue.

1.5.30 5.7.6

5.7.6 contains a security fix for a cross-site inclusion (XSSI) vulnerability (CVE-2019–9644), where files at a known URL could be included in a page from an unauthorized website if the user is logged into a Jupyter server. The fix involves setting the `X-Content-Type-Options: nosniff` header, and applying CSRF checks previously on all non-GET API requests to GET requests to API endpoints and the /files/ endpoint.

The attacking page is able to access some contents of files when using Internet Explorer through script errors, but this has not been demonstrated with other browsers.

1.5.31 5.7.5

- Fix compatibility with tornado 6 (4392, 4449).
- Fix opening integer filedescriptor during startup on Python 2 (4349)
- Fix compatibility with asynchronous [KernelManager.restart_kernel]{.title-ref} methods (4412)

1.5.32 5.7.4

5.7.4 fixes a bug introduced in 5.7.3, in which the `list_running_servers()` function attempts to parse HTML files as JSON, and consequently crashes (4284).

1.5.33 5.7.3

5.7.3 contains one security improvement and one security fix:

- Launch the browser with a local file which redirects to the server address including the authentication token (4260). This prevents another logged-in user from stealing the token from command line arguments and authenticating to the server. The single-use token previously used to mitigate this has been removed. Thanks to Dr. Owain Kenway for suggesting the local file approach.
- Upgrade bootstrap to 3.4, fixing an XSS vulnerability, which has been assigned CVE-2018-14041 (4271).

1.5.34 5.7.2

5.7.2 contains a security fix preventing malicious directory names from being able to execute javascript. CVE request pending.

1.5.35 5.7.1

5.7.1 contains a security fix preventing nbconvert endpoints from executing javascript with access to the server API. CVE request pending.

1.5.36 5.7.0

New features:

- Update to CodeMirror to 5.37, which includes f-string syntax for Python 3.6 (3816)
- Update jquery-ui to 1.12 (3836)
- Check Host header to more securely protect localhost deployments from DNS rebinding. This is a pre-emptive measure, not fixing a known vulnerability (3766). Use `.NotebookApp.allow_remote_access` and `.NotebookApp.local_hostnames` to configure access.
- Allow access-control-allow-headers to be overridden (3886)
- Allow configuring max_body_size and max_buffer_size (3829)
- Allow configuring get_secure_cookie keyword-args (3778)
- Respect nbconvert entrypoints as sources for exporters (3879)
- Include translation sources in source distributions (3925, 3931)
- Various improvements to documentation (3799, 3800, 3806, 3883, 3908)

Fixing problems:

- Fix breadcrumb link when running with a base url (3905)
- Fix possible type error when closing activity stream (3907)
- Disable metadata editing for non-editable cells (3744)
- Fix some styling and alignment of prompts caused by regressions in 5.6.0.
- Enter causing page reload in shortcuts editor (3871)
- Fix uploading to the same file twice (3712)

See the 5.7 milestone on GitHub for a complete list of pull requests involved in this release.

Thanks to the following contributors:

- Aaron Hall
- Benjamin Ragan-Kelley
- Bill Major
- bxy007
- Dave Aitken
- Denis Ledoux
- Félix-Antoine Fortin

- Gabriel
- Grant Nestor
- Kevin Bates
- Kristian Gregorius Hustad
- M Pacer
- Madicken Munk
- Maitiu O Ciarain
- Matthias Bussonnier
- Michael Boyle
- Michael Chirico
- Mokkapati, Praneet(ES)
- Peter Parente
- Sally Wilsak
- Steven Silvester
- Thomas Kluyver
- Walter Martin

1.5.37 5.6.0

New features:

- Execute cells by clicking icon in input prompt (3535, 3687)
- New "Save as" menu option (3289)
- When serving on a loopback interface, protect against DNS rebinding by checking the `Host` header from the browser (3714). This check can be disabled if necessary by setting `NotebookApp.allow_remote_access`. (Disabled by default while we work out some Mac issues in 3754).
- Add kernel_info_timeout traitlet to enable restarting slow kernels (3665)
- Add `custom_display_host` config option to override displayed URL (3668)
- Add /metrics endpoint for Prometheus Metrics (3490)
- Update to MathJax 2.7.4 (3751)
- Update to jQuery 3.3 (3655)
- Update marked to 0.4 (3686)

Fixing problems:

- Don't duplicate token in displayed URL (3656)
- Clarify displayed URL when listening on all interfaces (3703)
- Don't trash non-empty directories on Windows (3673)
- Include LICENSE file in wheels (3671)
- Don't show "0 active kernels" when starting the notebook (3696)

Testing:

- Add find replace test (3630)
- Selenium test for deleting all cells (3601)
- Make creating a new notebook more robust (3726)

Thanks to the following contributors:

- Arovit Narula (arovit)
- lucasoshiro (lucasoshiro)
- M Pacer (mpacer)
- Thomas Kluyver (takluyver)
- Todd (toddrme2178)
- Yuvi Panda (yuvipanda)

See the 5.6 milestone on GitHub for a complete list of pull requests involved in this release.

1.5.38 5.5.0

New features:

- The files list now shows file sizes (3539)
- Add a quit button in the dashboard (3004)
- Display hostname in the terminal when running remotely (3356, 3593)
- Add slides exportation/download to the menu (3287)
- Add any extra installed nbconvert exporters to the "Download as" menu (3323)
- Editor: warning when overwriting a file that is modified on disk (2783)
- Display a warning message if cookies are not enabled (3511)
- Basic `__version__` reporting for extensions (3541)
- Add `NotebookApp.terminals_enabled` config option (3478)
- Make buffer time between last modified on disk and last modified on last save configurable (3273)
- Allow binding custom shortcuts for 'close and halt' (3314)
- Add description for 'Trusted' notification (3386)
- Add `settings['activity_sources']` (3401)
- Add an `output_updated.OutputArea` event (3560)

Fixing problems:

- Fixes to improve web accessibility (3507)
- Fixed color contrast issue in tree.less (3336)
- Allow cancelling upload of large files (3373)
- Don't clear login cookie on requests without cookie (3380)
- Don't trash files on different device to home dir on Linux (3304)
- Clear waiting asterisks when restarting kernel (3494)
- Fix output prompt when `execution_count` missing (3236)

- Make the 'changed on disk' dialog work when displayed twice (3589)
- Fix going back to root directory with history in notebook list (3411)
- Allow defining keyboard shortcuts for missing actions (3561)
- Prevent default on pageup/pagedown when completer is active (3500)
- Prevent default event handling on new terminal (3497)
- ConfigManager should not write out default values found in the .d directory (3485)
- Fix leak of iopub object in activity monitoring (3424)
- Javascript lint in notebooklist.js (3409)
- Some Javascript syntax fixes (3294)
- Convert native for loop to `Array.forEach()` (3477)
- Disable cache when downloading nbconvert output (3484)
- Add missing digestmod arg to HMAC (3399)
- Log OSErrors failing to create less-critical files during startup (3384)
- Use powershell on Windows (3379)
- API spec improvements, API handler improvements (3368)
- Set notebook to dirty state after change to kernel metadata (3350)
- Use CSP header to treat served files as belonging to a separate origin (3341)
- Don't install gettext into builtins (3330)
- Add missing `import _` (3316, 3326)
- Write `notebook.json` file atomically (3305)
- Fix clicking with modifiers, page title updates (3282)
- Upgrade jQuery to version 2.2 (3428)
- Upgrade xterm.js to 3.1.0 (3189)
- Upgrade moment.js to 2.19.3 (3562)
- Upgrade CodeMirror to 5.35 (3372)
- "Require" pyzmq>=17 (3586)

Documentation:

- Documentation updates and organisation (3584)
- Add section in docs about privacy (3571)
- Add explanation on how to change the type of a cell to Markdown (3377)
- Update docs with confd implementation details (3520)
- Add more information for where `jupyter_notebook_config.py` is located (3346)
- Document options to enable nbextensions in specific sections (3525)
- jQuery attribute selector value MUST be surrounded by quotes (3527)
- Do not execute special notebooks with nbsphinx (3360)
- Other minor fixes in 3288, 3528, 3293, 3367

Testing:

- Testing with Selenium & Sauce labs (3321)
- Selenium utils + markdown rendering tests (3458)
- Convert insert cell tests to Selenium (3508)
- Convert prompt numbers tests to Selenium (3554)
- Convert delete cells tests to Selenium (3465)
- Convert undelete cell tests to Selenium (3475)
- More selenium testing utilities (3412)
- Only check links when build is trigger by Travis Cron job (3493)
- Fix Appveyor build errors (3430)
- Undo patches in teardown before attempting to delete files (3459)
- Get tests running with tornado 5 (3398)
- Unpin ipykernel version on Travis (3223)

Thanks to the following contributors:

- Arovit Narula (arovit)
- Ashley Teoh (ashleytqy)
- Nicholas Bollweg (bollwyvl)
- Alex Rothberg (cancan101)
- Celina Kilcrease (ckilcrease)
- dabuside (dabuside)
- Damian Avila (damianavila)
- Dana Lee (danagilliann)
- Dave Hirschfeld (dhirschfeld)
- Heng GAO (ehengao)
- Leo Gallucci (elgalu)
- Evan Van Dam (evandam)
- forbxy (forbxy)
- Grant Nestor (gnestor)
- Ethan T. Hendrix (hendrixet)
- Miro Hrončok (hroncok)
- Paul Ivanov (ivanov)
- Darío Hereñú (kant)
- Kevin Bates (kevin-bates)
- Maarten Breddels (maartenbreddels)
- Michael Droettboom (mdboom)
- Min RK (minrk)

- M Pacer (mpacer)
- Peter Parente (parente)
- Paul Masson (paulmasson)
- Philipp Rudiger (philippjfr)
- Mac Knight (Shels1909)
- Hisham Elsheshtawy (Sheshtawy)
- Simon Biggs (SimonBiggs)
- Sunil Hari (`@sunilhari`)
- Thomas Kluyver (takluyver)
- Tim Klever (tklever)
- Gabriel Ruiz (unnamedplay-r)
- Vaibhav Sagar (vaibhavsagar)
- William Hosford (whosford)
- Hong (xuhdev)

See the 5.5 milestone on GitHub for a complete list of pull requests involved in this release.

1.5.39 5.4.1

A security release to fix CVE-2018-8768.

Thanks to Alex for identifying this bug, and Jonathan Kamens and Scott Sanderson at Quantopian for verifying it and bringing it to our attention.

1.5.40 5.4.0

- Fix creating files and folders after navigating directories in the dashboard (3264).
- Enable printing notebooks in colour, removing the CSS that made everything black and white (3212).
- Limit the completion options displayed in the notebook to 1000, to avoid performance issues with very long lists (3195).
- Accessibility improvements in `tree.html` (3271).
- Added alt-text to the kernel logo image in the notebook UI (3228).
- Added a test on Travis CI to flag if symlinks are accidentally introduced in the future. This should prevent the issue that necessitated `release-5.3.1{.interpreted-text role="ref"}` (3227).
- Use lowercase letters for random IDs generated in our Javascript (3264).
- Removed duplicate code setting `TextCell.notebook` (3256).

Thanks to the following contributors:

- Alex Soderman (asoderman)
- Matthias Bussonnier (Carreau)
- Min RK (minrk)
- Nitesh Sawant (ns23)

- Thomas Kluyver (takluyver)
- Yuvi Panda (yuvipanda)

See the 5.4 milestone on GitHub for a complete list of pull requests involved in this release.

1.5.41 5.3.1

Replaced a symlink in the repository with a copy, to fix issues installing on Windows (3220).

1.5.42 5.3.0

This release introduces a couple noteable improvements, such as terminal support for Windows and support for OS trash (files deleted from the notebook dashboard are moved to the OS trash vs. deleted permanently).

- Add support for terminals on windows (3087).
- Add a "restart and run all" button to the toolbar (2965).
- Send files to os trash mechanism on delete (1968).
- Allow programmatic copy to clipboard (3088).
- Use DOM History API for navigating between directories in the file browser (3115).
- Add translated files to folder(docs-translations) (3065).
- Allow non empty dirs to be deleted (3108).
- Set cookie on base_url (2959).
- Allow token-authenticated requests cross-origin by default (2920).
- Change cull_idle_timeout_minimum to 1 from 300 (2910).
- Config option to shut down server after n seconds with no kernels (2963).
- Display a "close" button on load notebook error (3176).
- Add action to command pallette to run CodeMirror's "indentAuto" on selection (3175).
- Add option to specify extra services (3158).
- Warn_bad_name should not use global name (3160).
- Avoid overflow of hidden form (3148).
- Fix shutdown trans loss (3147).
- Find available kernelspecs more efficiently (3136).
- Don't try to translate missing help strings (3122).
- Frontend/extension-config: allow default json files in a .d directory (3116).
- Use [requirejs]{.title-ref} vs. [require]{.title-ref} (3097).
- Fixes some ui bugs in firefox #3044 (3058).
- Compare non-specific language code when choosing to use arabic numerals (3055).
- Fix save-script deprecation (3053).
- Include moment locales in package_data (3051).
- Fix moment locale loading in bidi support (3048).

- Tornado 5: periodiccallback loop arg will be removed (3034).
- Use [/files]{.title-ref} prefix for pdf-like files (3031).
- Add folder for document translation (3022).
- When login-in via token, let a chance for user to set the password (3008).
- Switch to jupyter_core implementation of ensure_dir_exists (3002).
- Send http shutdown request on 'stop' subcommand (3000).
- Work on loading ui translations (2969).
- Fix ansi inverse (2967).
- Add send2trash to requirements for building docs (2964).
- I18n readme.md improvement (2962).
- Add 'reason' field to json error responses (2958).
- Add some padding for stream outputs (3194).
- Always use setuptools in `setup.py` (3206).
- Fix clearing cookies on logout when `base_url` is configured (3207).

Thanks to the following contributors:

- bacboc (bacboc)
- Steven Silvester (blink1073)
- Matthias Bussonnier (Carreau)
- ChungJooHo (ChungJooHo)
- edida (edida)
- Francesco Franchina (`ferdas`)
- forbxy (forbxy)
- Grant Nestor (gnestor)
- Josh Barnes (jcb91)
- JocelynDelalande (JocelynDelalande)
- Karthik Balakrishnan (karthikb351)
- Kevin Bates (kevin-bates)
- Kirit Thadaka (kirit93)
- Lilian Besson (Naereen)
- Maarten Breddels (maartenbreddels)
- Madhu94 (Madhu94)
- Matthias Geier (mgeier)
- Michael Heilman (mheilman)
- Min RK (minrk)
- PHaeJin (PHaeJin)
- Sukneet (Sukneet)

- Thomas Kluyver (takluyver)

See the 5.3 milestone on GitHub for a complete list of pull requests involved in this release.

1.5.43 5.2.1

- Fix invisible CodeMirror cursor at specific browser zoom levels (2983).
- Fix nbconvert handler causing broken export to PDF (2981).
- Fix the prompt_area argument of the output area constructor. (2961).
- Handle a compound extension in new_untitled (2949).
- Allow disabling offline message buffering (2916).

Thanks to the following contributors:

- Steven Silvester (blink1073)
- Grant Nestor (gnestor)
- Jason Grout (jasongrout)
- Min RK (minrk)
- M Pacer (mpacer)

See the 5.2.1 milestone on GitHub for a complete list of pull requests involved in this release.

1.5.44 5.2.0

- Allow setting token via jupyter_token env (2921).
- Fix some errors caused by raising 403 in get_current_user (2919).
- Register contents_manager.files_handler_class directly (2917).
- Update viewable_extensions (2913).
- Show edit shortcuts modal after shortcuts modal is hidden (2912).
- Improve edit/view behavior (2911).
- The root directory of the notebook server should never be hidden (2907).
- Fix notebook require config to match tools/build-main (2888).
- Give page constructor default arguments (2887).
- Fix codemirror.less to match codemirror's expected padding layout (2880).
- Add x-xsrftoken to access-control-allow-headers (2876).
- Buffer messages when websocket connection is interrupted (2871).
- Load locale dynamically only when not en-us (2866).
- Changed key strength to 2048 bits (2861).
- Resync jsversion with python version (2860).
- Allow copy operation on modified, read-only notebook (2854).
- Update error handling on apihandlers (2853).
- Test python 3.6 on travis, drop 3.3 (2852).

- Avoid base64-literals in image tests (2851).
- Upgrade xterm.js to 2.9.2 (2849).
- Changed all python variables named file to file_name to not override built_in file (2830).
- Add more doc tests (2823).
- Typos fix (2815).
- Rename and update license [ci skip] (2810).
- Travis builds doc (2808).
- Pull request i18n (2804).
- Factor out output_prompt_function, as is done with input prompt (2774).
- Use rfc5987 encoding for filenames (2767).
- Added path to the resources metadata, the same as in from_filename(...) in nbconvert.exporters.py (2753).
- Make "extrakeys" consistent for notebook and editor (2745).
- Bidi support (2357).

Special thanks to samarsultan and the Arabic Competence and Globalization Center Team at IBM Egypt for adding RTL (right-to-left) support to the notebook!

See the 5.2 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.45 5.1.0

- Preliminary i18n implementation (2140).
- Expose URL with auth token in notebook UI (2666).
- Fix search background style (2387).
- List running notebooks without requiring `--allow-root` (2421).
- Allow session of type other than notebook (2559).
- Fix search background style (2387).
- Fix some Markdown styling issues (2571), (2691) and (2534).
- Remove keymaps that conflict with non-English keyboards (2535).
- Add session-specific favicons (notebook, terminal, file) (2452).
- Add /api/shutdown handler (2507).
- Include metadata when copying a cell (2349).
- Stop notebook server from command line (2388).
- Improve "View" and "Edit" file handling in dashboard (2449) and (2402).
- Provide a promise to replace use of the `app_initialized.NotebookApp` event (2710).
- Fix disabled collapse/expand output button (2681).
- Cull idle kernels using `--MappingKernelManager.cull_idle_timeout` (2215).
- Allow read-only notebooks to be trusted (2718).

See the 5.1 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.46 5.0.0

This is the first major release of the Jupyter Notebook since version 4.0 was created by the "Big Split" of IPython and Jupyter.

We encourage users to start trying JupyterLab in preparation for a future transition.

We have merged more than 300 pull requests since 4.0. Some of the major user-facing changes are described here.

File sorting in the dashboard

Files in the dashboard may now be sorted by last modified date or name (943):

Cell tags

There is a new cell toolbar for adding *cell tags* (2048):

Cell tags are a lightweight way to customise the behaviour of tools working with notebooks; we're working on building support for them into tools like nbconvert and nbval. To start using tags, select `Tags` in the `View > Cell Toolbar` menu in a notebook.

The UI for editing cell tags is basic for now; we hope to improve it in future releases.

Table style

The default styling for tables in the notebook has been updated (1776).

Customise keyboard shortcuts

You can now edit keyboard shortcuts for *Command Mode* within the UI (1347):

See the `Help > Edit Keyboard Shortcuts` menu item and follow the instructions.

Other additions

- You can copy and paste cells between notebooks, using `Ctrl-C{.interpreted-text role="kbd"}` and `Ctrl-V{.interpreted-text role="kbd"}` (`Cmd-C{.interpreted-text role="kbd"}` and `Cmd-V{.interpreted-text role="kbd"}` on Mac).
- It's easier to configure a password for the notebook with the new `jupyter notebook password` command (2007).
- The file list can now be ordered by *last modified* or by *name* (943).
- Markdown cells now support attachments. Simply drag and drop an image from your desktop to a markdown cell to add it. Unlike relative links that you enter manually, attachments are embedded in the notebook itself. An unreferenced attachment will be automatically scrubbed from the notebook on save (621).
- Undoing cell deletion now supports undeleting multiple cells. Cells may not be in the same order as before their deletion, depending on the actions you did on the meantime, but this should should help reduce the impact of accidentally deleting code.
- The file browser now has *Edit* and *View* buttons.
- The file browser now supports moving multiple files at once (1088).

- The Notebook will refuse to run as root unless the `--allow-root` flag is given (1115).
- Keyboard shortcuts are now declarative (1234).
- Toggling line numbers can now affect all cells (1312).
- Add more visible *Trusted* and *Untrusted* notifications (1658).
- The favicon (browser shortcut icon) now changes to indicate when the kernel is busy (1837).
- Header and toolbar visibility is now persisted in nbconfig and across sessions (1769).
- Load server extensions with ConfigManager so that merge happens recursively, unlike normal config values, to make it load more consistently with frontend extensions(2108).
- The notebook server now supports the bundler API from the jupyter_cms incubator project (1579).
- The notebook server now provides information about kernel activity in its kernel resource API (1827).

Remember that upgrading `notebook` only affects the user interface. Upgrading kernels and libraries may also provide new features, better stability and integration with the notebook interface.

1.5.47 4.4.0

- Allow override of output callbacks to redirect output messages. This is used to implement the ipywidgets Output widget, for example.
- Fix an async bug in message handling by allowing comm message handlers to return a promise which halts message processing until the promise resolves.

See the 4.4 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.48 4.3.2

4.3.2 is a patch release with a bug fix for CodeMirror and improved handling of the "editable" cell metadata field.

- Monkey-patch for CodeMirror that resolves #2037 without breaking #1967
- Read-only (`"editable": false`) cells can be executed but cannot be split, merged, or deleted

See the 4.3.2 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.49 4.3.1

4.3.1 is a patch release with a security patch, a couple bug fixes, and improvements to the newly-released token authentication.

Security fix:

- CVE-2016-9971. Fix CSRF vulnerability, where malicious forms could create untitled files and start kernels (no remote execution or modification of existing files) for users of certain browsers (Firefox, Internet Explorer / Edge). All previous notebook releases are affected.

Bug fixes:

- Fix carriage return handling
- Make the font size more robust against fickle browsers
- Ignore resize events that bubbled up and didn't come from window
- Add Authorization to allowed CORS headers

- Downgrade CodeMirror to 5.16 while we figure out issues in Safari

Other improvements:

- Better docs for token-based authentication
- Further highlight token info in log output when autogenerated

See the 4.3.1 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.50 4.3.0

4.3 is a minor release with many bug fixes and improvements. The biggest user-facing change is the addition of token authentication, which is enabled by default. A token is generated and used when your browser is opened automatically, so you shouldn't have to enter anything in the default circumstances. If you see a login page (e.g. by switching browsers, or launching on a new port with `--no-browser`), you get a login URL with the token from the command `jupyter notebook list`, which you can paste into your browser.

Highlights:

- API for creating mime-type based renderer extensions using `OutputArea.register_mime_type` and `Notebook.render_cell_output` methods. See mimerender-cookiecutter for reference implementations and cookiecutter.
- Enable token authentication by default. See `server_security{.interpreted-text role="ref"}` for more details.
- Update security docs to reflect new signature system
- Switched from term.js to xterm.js

Bug fixes:

- Ensure variable is set if exc_info is falsey
- Catch and log handler exceptions in `events.trigger`
- Add debug log for static file paths
- Don't check origin on token-authenticated requests
- Remove leftover print statement
- Fix highlighting of Python code blocks
- `json_errors` should be outermost decorator on API handlers
- Fix remove old nbserver info files
- Fix notebook mime type on download links
- Fix carriage symbol behavior
- Fix terminal styles
- Update dead links in docs
- If kernel is broken, start a new session
- Include cross-origin check when allowing login URL redirects

Other improvements:

- Allow JSON output data with mime type `application/*+json`
- Allow kernelspecs to have spaces in them for backward compat
- Allow websocket connections from scripts

- Allow `None` for post_save_hook
- Upgrade CodeMirror to 5.21
- Upgrade xterm to 2.1.0
- Docs for using comms
- Set `dirty` flag when output arrives
- Set `ws-url` data attribute when accessing a notebook terminal
- Add base aliases for nbextensions
- Include `@` operator in CodeMirror IPython mode
- Extend mathjax_url docstring
- Load nbextension in predictable order
- Improve the error messages for nbextensions
- Include cross-origin check when allowing login URL redirects

See the 4.3 milestone on GitHub for a complete list of issues and pull requests involved in this release.

1.5.51 4.2.3

4.2.3 is a small bugfix release on 4.2.

Highlights:

- Fix regression in 4.2.2 that delayed loading custom.js until after `notebook_loaded` and `app_initialized` events have fired.
- Fix some outdated docs and links.

1.5.52 4.2.2

4.2.2 is a small bugfix release on 4.2, with an important security fix. All users are strongly encouraged to upgrade to 4.2.2.

Highlights:

- **Security fix**: CVE-2016-6524, where untrusted latex output could be added to the page in a way that could execute javascript.
- Fix missing POST in OPTIONS responses.
- Fix for downloading non-ascii filenames.
- Avoid clobbering ssl_options, so that users can specify more detailed SSL configuration.
- Fix inverted load order in nbconfig, so user config has highest priority.
- Improved error messages here and there.

1.5.53 4.2.1

4.2.1 is a small bugfix release on 4.2. Highlights:

- Compatibility fixes for some versions of ipywidgets
- Fix for ignored CSS on Windows
- Fix specifying destination when installing nbextensions

1.5.54 4.2.0

Release 4.2 adds a new API for enabling and installing extensions. Extensions can now be enabled at the system-level, rather than just per-user. An API is defined for installing directly from a Python package, as well.

Highlighted changes:

- Upgrade MathJax to 2.6 to fix vertical-bar appearing on some equations.
- Restore ability for notebook directory to be root (4.1 regression)
- Large outputs are now throttled, reducing the ability of output floods to kill the browser.
- Fix the notebook ignoring cell executions while a kernel is starting by queueing the messages.
- Fix handling of url prefixes (e.g. JupyterHub) in terminal and edit pages.
- Support nested SVGs in output.

And various other fixes and improvements.

1.5.55 4.1.0

Bug fixes:

- Properly reap zombie subprocesses
- Fix cross-origin problems
- Fix double-escaping of the base URL prefix
- Handle invalid unicode filenames more gracefully
- Fix ANSI color-processing
- Send keepalive messages for web terminals
- Fix bugs in the notebook tour

UI changes:

- Moved the cell toolbar selector into the *View* menu. Added a button that triggers a "hint" animation to the main toolbar so users can find the new location. (Click here to see a screencast)
- Added *Restart & Run All* to the *Kernel* menu. Users can also bind it to a keyboard shortcut on action `restart-kernel-and-run-all-cells`.
- Added multiple-cell selection. Users press `Shift-Up/Down` or `Shift-K/J` to extend selection in command mode. Various actions such as cut/copy/paste, execute, and cell type conversions apply to all selected cells.
- Added a command palette for executing Jupyter actions by name. Users press `Cmd/Ctrl-Shift-P` or click the new command palette icon on the toolbar.
- Added a *Find and Replace* dialog to the *Edit* menu. Users can also press `F` in command mode to show the dialog.

Other improvements:

- Custom KernelManager methods can be Tornado coroutines, allowing async operations.
- Make clearing output optional when rewriting input with `set_next_input(replace=True)`.
- Added support for TLS client authentication via `--NotebookApp.client-ca`.
- Added tags to `jupyter/notebook` releases on DockerHub. `latest` continues to track the master branch.

See the 4.1 milestone on GitHub for a complete list of issues and pull requests handled.

1.5.56 4.0.x

4.0.6

- fix installation of mathjax support files
- fix some double-escape regressions in 4.0.5
- fix a couple of cases where errors could prevent opening a notebook

4.0.5

Security fixes for maliciously crafted files.

- CVE-2015-6938: malicious filenames
- CVE-2015-7337: malicious binary files in text editor.

Thanks to Jonathan Kamens at Quantopian and Juan Broullón for the reports.

4.0.4

- Fix inclusion of mathjax-safe extension

4.0.2

- Fix launching the notebook on Windows
- Fix the path searched for frontend config

4.0.0

First release of the notebook as a standalone package.

CHAPTER

TWO

CONFIGURATION

2.1 Configuration Overview

Beyond the default configuration settings, you can configure a rich array of options to suit your workflow. Here are areas that are commonly configured when using Jupyter Notebook:

- *Jupyter's common configuration system*
- *Jupyter Server*
- *Notebook extensions*

Let's look at highlights of each area.

2.1.1 Jupyter's Common Configuration system

Jupyter applications, from the Notebook to JupyterHub to nbgrader, share a common configuration system. The process for creating a configuration file and editing settings is similar for all the Jupyter applications.

- Jupyter's Common Configuration Approach
- Common Directories and File Locations
- Language kernels
- traitlets provide a low-level architecture for configuration.

2.1.2 Jupyter server

The Jupyter Server runs the language kernel and communicates with the front-end Notebook client (i.e. the familiar notebook interface).

- Configuring the Jupyter Server

  To create a `jupyter_server_config.py` file in the `.jupyter` directory, with all the defaults commented out, use the following command:

  ```
  $ jupyter server --generate-config
  ```

- Running a Jupyter Server
- Related: Configuring a language kernel to run in the Jupyter Server enables your server to run other languages, like R or Julia.

2.1.3 Notebook extensions

The Notebook frontend can be extending with JupyterLab extensions.

See the *Frontend Extension Guide* for more information.

Security in Jupyter notebooks: Since security policies vary from organization to organization, we encourage you to consult with your security team on settings that would be best for your use cases. Our documentation offers some responsible security practices, and we recommend becoming familiar with the practices.

2.2 Extending the Notebook

Certain subsystems of the notebook server are designed to be extended or overridden by users. These documents explain these systems, and show how to override the notebook's defaults with your own custom behavior.

2.2.1 Custom front-end extensions

This describes the basic steps to write a TypeScript extension for the Jupyter notebook front-end. This allows you to customize the behaviour of the various pages like the dashboard, the notebook, or the text editor.

Starting with Notebook v7, front-end extensions for the notebook can be developed as prebuilt JupyterLab extensions.

This means Notebook v7 is able to reuse many of the existing extensions from the JupyterLab ecosystem as is.

If you would like to develop a prebuilt extension for Notebook v7, check out:

- JupyterLab Extension Tutorial: A tutorial to learn how to make a simple JupyterLab extension.
- The JupyterLab Extension Examples Repository: A short tutorial series to learn how to develop extensions for JupyterLab by example.

CHAPTER

THREE

CONTRIBUTOR

3.1 Contributing to Jupyter Notebook

Thanks for contributing to Jupyter Notebook!

Make sure to follow Project Jupyter's Code of Conduct for a friendly and welcoming collaborative environment.

3.1.1 Setting up a development environment

Note: You will need NodeJS to build the extension package.

The `jlpm` command is JupyterLab's pinned version of yarn that is installed with JupyterLab. You may use `yarn` or `npm` in lieu of `jlpm` below.

Note: we recomment using `mamba` to speed the creating of the environment.

```
# create a new environment
mamba create -n notebook -c conda-forge python nodejs -y

# activate the environment
mamba activate notebook

# Install package in development mode
pip install -e ".[dev,test]"

# Link the notebook extension and @jupyter-notebook schemas
jlpm develop

# Enable the server extension
jupyter server extension enable notebook
```

`notebook` follows a monorepo structure. To build all the packages at once:

```
jlpm build
```

There is also a `watch` script to watch for changes and rebuild the app automatically:

```
jlpm watch
```

To make sure the `notebook` server extension is installed:

```
$ jupyter server extension list
Config dir: /home/username/.jupyter

Config dir: /home/username/miniforge3/envs/notebook/etc/jupyter
    jupyterlab enabled
    - Validating jupyterlab...
      jupyterlab 3.0.0 OK
    notebook enabled
    - Validating notebook...
      notebook 7.0.0a0 OK

Config dir: /usr/local/etc/jupyter
```

Then start Jupyter Notebook with:

```
jupyter notebook
```

3.1.2 Running Tests

To run the tests:

```
jlpm run build:test
jlpm run test
```

There are also end to end tests to cover higher level user interactions, located in the `ui-tests` folder. To run these tests:

```
cd ui-tests
# start a new Jupyter server in a terminal
jlpm start

# in a new terminal, run the tests
jlpm test
```

The `test` script calls the Playwright test runner. You can pass additional arguments to `playwright` by appending parameters to the command. For example to run the test in headed mode, `jlpm test --headed`.

Checkout the Playwright Command Line Reference for more information about the available command line options.

Running the end to end tests in headful mode will trigger something like the following:

3.1.3 Code Styling

All non-python source code is formatted using prettier and python source code is formatted using blacks When code is modified and committed, all staged files will be automatically formatted using pre-commit git hooks (with help from pre-commit. The benefit of using a code formatters like `prettier` and `black` is that it removes the topic of code style from the conversation when reviewing pull requests, thereby speeding up the review process.

As long as your code is valid, the pre-commit hook should take care of how it should look. `pre-commit` and its associated hooks will automatically be installed when you run `pip install -e ".[dev,test]"`

To install `pre-commit` manually, run the following:

```
pip install pre-commit
pre-commit install
```

You can invoke the pre-commit hook by hand at any time with:

```
pre-commit run
```

which should run any autoformatting on your code and tell you about any errors it couldn't fix automatically. You may also install black integration into your text editor to format code automatically.

If you have already committed files before setting up the pre-commit hook with `pre-commit install`, you can fix everything up using `pre-commit run --all-files`. You need to make the fixing commit yourself after that.

You may also use the prettier npm script (e.g. `npm run prettier` or `yarn prettier` or `jlpm prettier`) to format the entire code base. We recommend installing a prettier extension for your code editor and configuring it to format your code with a keyboard shortcut or automatically on save.

Some of the hooks only run on CI by default, but you can invoke them by running with the `--hook-stage manual` argument.

3.2 Developer FAQ

1. How do I install a prerelease version such as a beta or release candidate?

```
python -m pip install notebook --pre --upgrade
```