

# ipynb modification using python json module

Ian Stewart  
2019 08 04

# ipynb modification using python json module

Jupyter notebook data is saved in the **JavaScript Object Notation** (json) format.

Jupyter notebook files have the extension *ipynb* (interactive **python notebook**).

Using a python program that utilizes the json module modifications may be made to ipynb file(s).

# Presentation Contents

- Review a Jupyter Notebook with 2 x cells.
- Modify the metadata of the cells.
- Review the modifications in the ipynb file.
- Python program that uses the json module.
- Review the python/json code
- Demonstration of the program modifying cell metadata

# Example of Jupyter Notebook with 2 x cells

demo\_json\_1 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

demo\_json\_1

localhost:8888/notebooks/demo\_json\_1.ipynb#

jupyter demo\_json\_1 Last Checkpoint: 4 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Save Create Delete Copy Paste Undo Redo Run Stop Refresh Code

## Demo

This is a demonstration file for ipynb. It contains this markdown cell and the python3 code cell below.

```
In [ ]: 1 a = 42
        2 print(bin(a))
        3 b = hex(3735928559)
        4 print(b)
```

# Modifications that may be desirable...

Not deletable

Not editable

## Demo

This is a demonstration file for ipynb. It contains this markdown cell and the python3 code cell below.

```
In [ ]: 1 a = 42
        2 print(bin(a))
        3 b = hex(3735928559)
        4 print(b)
```

Not deletable

editable

# Set cells to *Edit Metadata*

The screenshot shows a JupyterLab interface in a Mozilla Firefox browser window. The browser's address bar displays `localhost:8888/notebooks/demo_json_1.ipynb#`. The JupyterLab header includes the Jupyter logo, the notebook name `demo_json_1`, and a status message: `Last Checkpoint: an hour ago (autosaved)`. A `Logout` button is visible on the right. The main toolbar contains menus for `File`, `Edit`, `View`, `Insert`, `Cell`, `Kernel`, `Widgets`, and `Help`. The `View` menu is open, showing options: `Toggle Header`, `Toggle Toolbar`, `Toggle Line Numbers`, and `Cell Toolbar`. The `Cell Toolbar` sub-menu is also open, listing options: `None`, `Edit Metadata` (highlighted by the mouse), `Raw Cell Format`, `Slideshow`, `Attachments`, and `Tags`. Below the menu, a code cell is visible with the following Python code:

```
In [ ]: 1 a = 42
        2 print(bin(a))
        3 b = hex(37359)
        4 print(b)
```

# Cells have *Edit Metadata* capability

Edit Metadata

Edit Metadata

## Demo

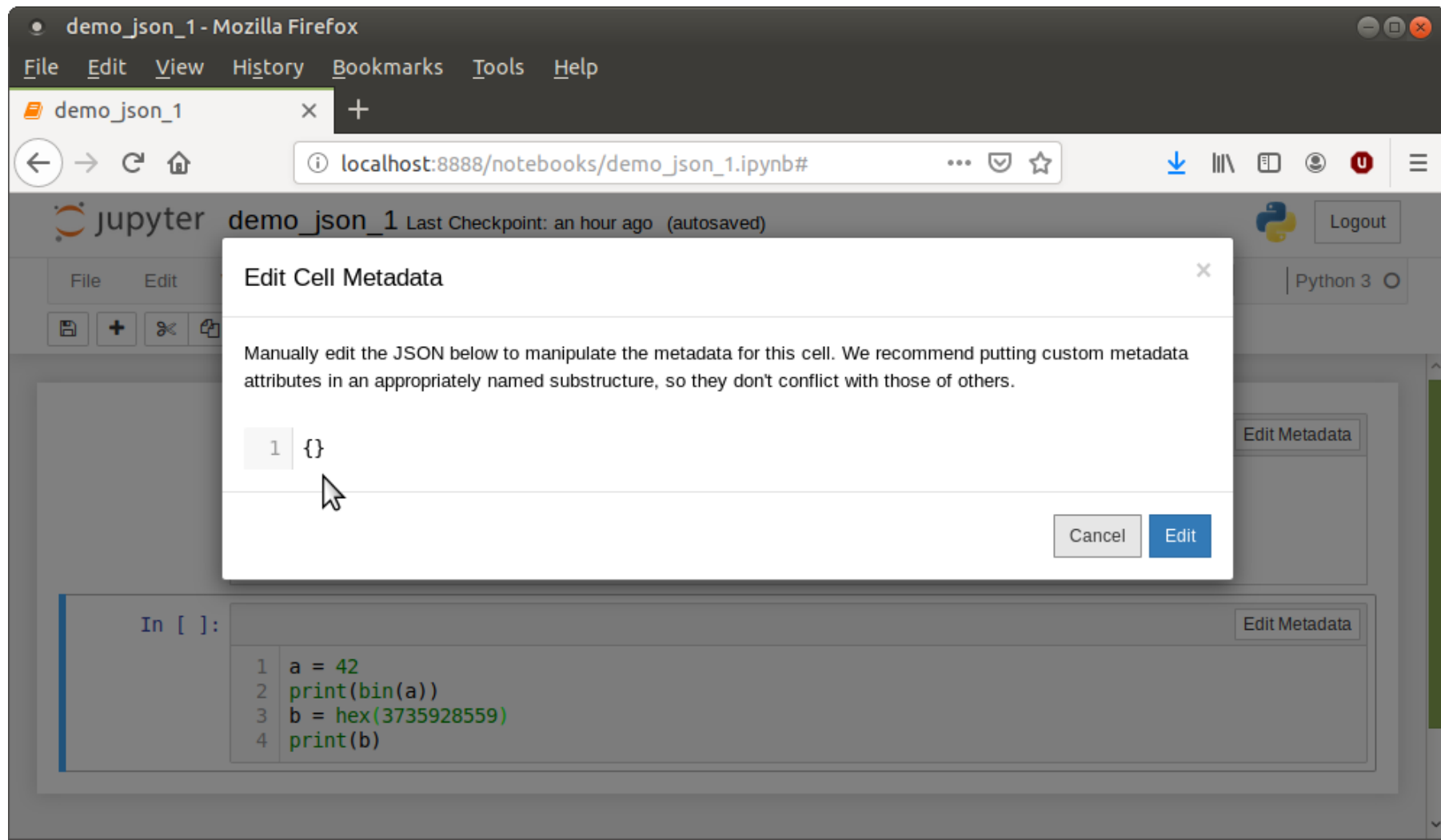
This is a demonstration file for ipynb. It contains this markdown cell and the python3 code cell below.

In [ ]:

Edit Metadata

```
1 a = 42
2 print(bin(a))
3 b = hex(3735928559)
4 print(b)
```

# Editing first cell metadata



The screenshot shows a Mozilla Firefox browser window displaying the JupyterLab interface. The browser's address bar shows the URL `localhost:8888/notebooks/demo_json_1.ipynb#`. The JupyterLab interface includes a top bar with the Jupyter logo, the notebook name `demo_json_1`, and a status message `Last Checkpoint: an hour ago (autosaved)`. A sidebar on the left contains a `File` menu and a `Edit` button. The main workspace shows a code cell with the following Python code:

```
In [ ]:
1 a = 42
2 print(bin(a))
3 b = hex(3735928559)
4 print(b)
```

An `Edit Cell Metadata` dialog box is open in the center of the screen. The dialog box has a title bar with a close button (X). The main text inside the dialog box reads: "Manually edit the JSON below to manipulate the metadata for this cell. We recommend putting custom metadata attributes in an appropriately named substructure, so they don't conflict with those of others." Below this text is a text area containing the JSON `{}`. A mouse cursor is pointing at the `{}` in the text area. At the bottom right of the dialog box are two buttons: `Cancel` and `Edit`.



# Cell metadata keys

## Cell metadata

The following metadata keys are defined at the cell level:

Key	Value	Interpretation
collapsed	bool	Whether the cell's output container should be collapsed
autoscroll	bool or 'auto'	Whether the cell's output is scrolled, unscrolled, or autoscrolled
deletable	bool	If False, prevent deletion of the cell
format	'mime/type'	The mime-type of a <a href="#">Raw NBConvert Cell</a>
name	str	A name for the cell. Should be unique
tags	list of str	A list of string tags on the cell. Commas are not allowed in a tag

Above table from:

<https://ipython.org/ipython-doc/dev/notebook/nbformat.html>

Not listed: editable – new in version 5

# Markdown cell: "editable": false, "deletable": false

## Edit Cell Metadata ×

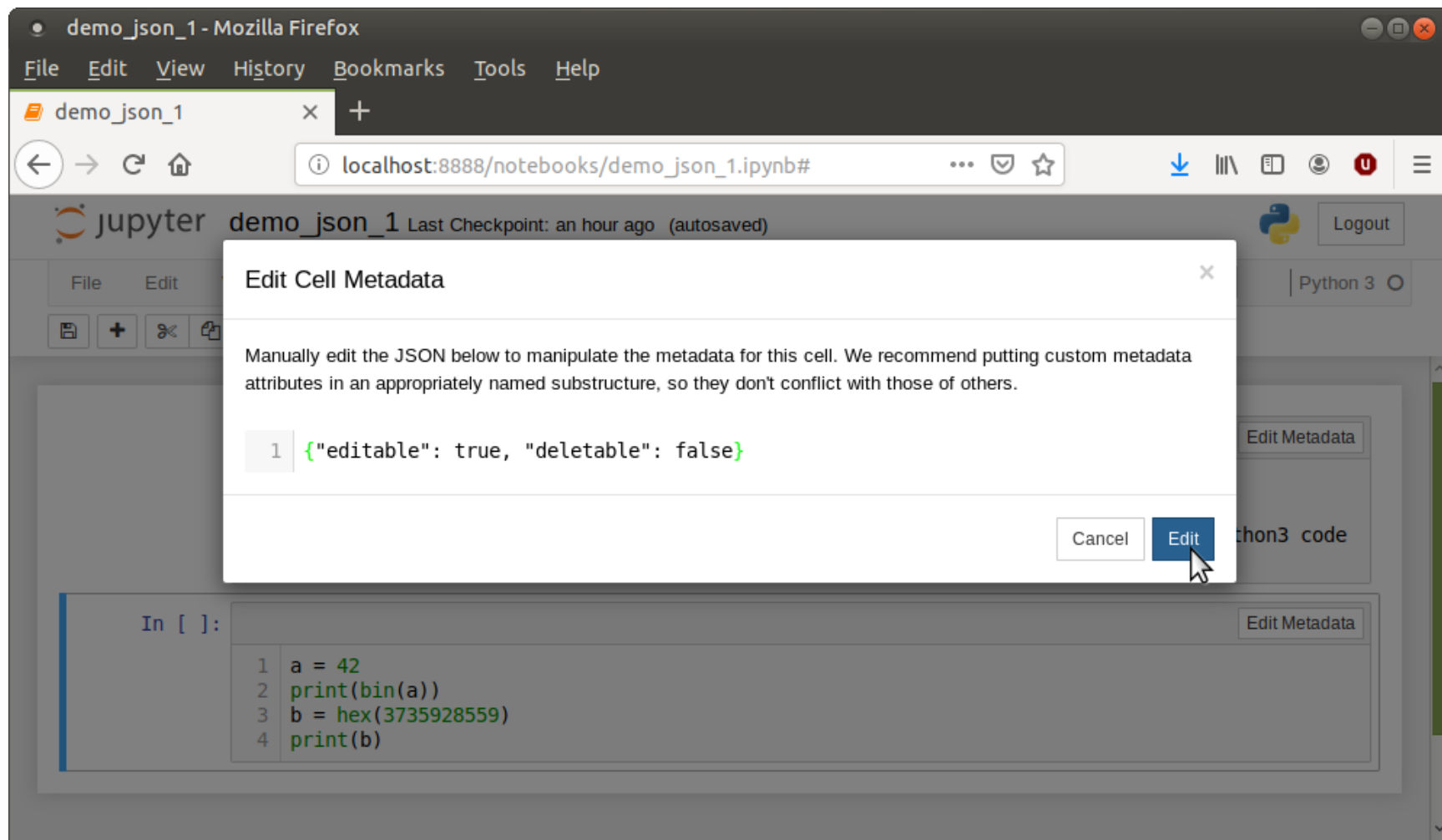
Manually edit the JSON below to manipulate the metadata for this cell. We recommend putting custom metadata attributes in an appropriately named substructure, so they don't conflict with those of others.

```
1 {"editable": false, "deletable": false}
```

Cancel

Edit

# Code cell: "editable": true, "deletable": false



The screenshot shows a JupyterLab interface in a Mozilla Firefox browser window. The browser address bar displays `localhost:8888/notebooks/demo_json_1.ipynb#`. The JupyterLab header shows the notebook name `demo_json_1` and a status message `Last Checkpoint: an hour ago (autosaved)`. A modal dialog titled `Edit Cell Metadata` is open in the center. The dialog contains the following text: `Manually edit the JSON below to manipulate the metadata for this cell. We recommend putting custom metadata attributes in an appropriately named substructure, so they don't conflict with those of others.` Below the text is a text area with the JSON `1 {"editable": true, "deletable": false}`. At the bottom right of the dialog are `Cancel` and `Edit` buttons, with a mouse cursor hovering over the `Edit` button. In the background, a code cell is visible with the following Python code: 

```
In [ ]:
1 a = 42
2 print(bin(a))
3 b = hex(3735928559)
4 print(b)
```

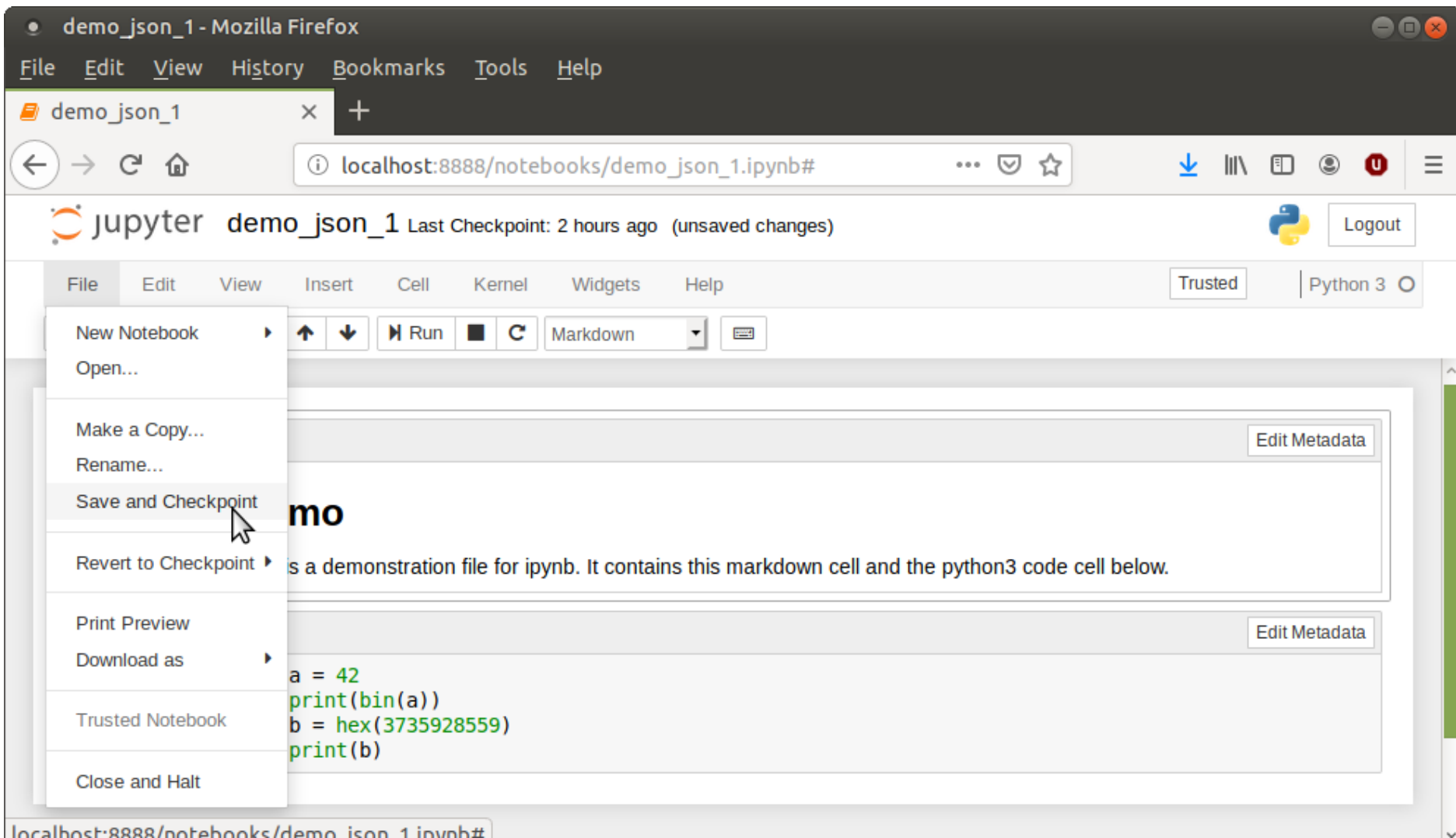
# View of Jupyter ipynb script. Before modification

```
demo_json_1.ipynb x
1 {
2   "cells": [
3     {
4       "cell_type": "markdown",
5       "metadata": {},
6       "source": [
7         "# Demo\n",
8         "\n",
9         "This is a demonstration file for ipynb. It contains this markdown cell
10        and the python3 code cell below. "
11      ],
12    },
13    {
14      "cell_type": "code",
15      "execution_count": null,
16      "metadata": {},
17      "outputs": [],
18      "source": [
19        "a = 42\n",
20        "print(bin(a))\n",
21        "b = hex(3735928559)\n",
22        "print(b)"
23      ]
24    }
25  ]
26 }
```

Markdown cell. No metadata

Code cell. No metadata

# Save Jupyter Notebook so ipynb file is updated



The screenshot shows a Jupyter Notebook interface in a Mozilla Firefox browser window. The browser tab is titled "demo\_json\_1 - Mozilla Firefox". The address bar shows the URL "localhost:8888/notebooks/demo\_json\_1.ipynb#". The Jupyter Notebook interface has a top bar with the Jupyter logo, the notebook title "demo\_json\_1", and a status message "Last Checkpoint: 2 hours ago (unsaved changes)". There is a "Logout" button in the top right corner. Below the top bar is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The "File" menu is open, showing options: New Notebook, Open..., Make a Copy..., Rename..., Save and Checkpoint (highlighted by the mouse cursor), Revert to Checkpoint, Print Preview, Download as, Trusted Notebook, and Close and Halt. The main content area of the notebook shows a markdown cell with the text "mo" and a code cell with the following Python code:

```
a = 42
print(bin(a))
b = hex(3735928559)
print(b)
```

The status bar at the bottom of the browser window shows the URL "localhost:8888/notebooks/demo\_json\_1.ipynb#".

# Save Jupyter Notebook so ipynb file is updated

```
demo_json_1.ipynb ✕  
1 {  
2   "cells": [  
3     {  
4       "cell_type": "markdown",  
5       "metadata": {  
6         "deletable": false,  
7         "editable": false  
8       },  
9       "source": [  
10        "# Demo\n",  
11        "\n",  
12        "This is a demonstration file for ipynb. It contains this markdown cell  
and the python3 code cell below. "  
13      ]  
14    },  
15    {  
16      "cell_type": "code",  
17      "execution_count": null,  
18      "metadata": {  
19        "deletable": false  
20      },  
21      "outputs": [],  
22      "source": [  
23        "# Demo\n",  
24        "\n",  
25        "This is a demonstration file for ipynb. It contains this python3 code cell  
and the markdown cell above. "  
26      ]  
27    }  
28  ]  
29 }
```

Markdown cell.  
With metadata added

Code cell.  
With metadata  
"Editable": true  
Default? So not added?

# Any Issue with editing metadata for each cell?

- What if each ipynb file has, say, 20 cells, and your project has 20 ipynb files.
- Then that's 400 manual edits required which will take a lot of time.

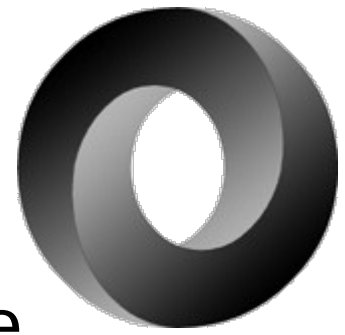
Solution?

# Solution

- A python program that imports the simplejson module.
- Poll user for the ipynb files to change.
- Poll user for cell metadata parameters to modify.
- json is used to load each file.
- Iterate over each cell and modify the metadata.
- json is used to dump the data back to the file.
- Jupyter-notebook re-loads the modified file.



# json



- JSON is a text format that is completely language independent. It was derived from JavaScript.
- Website: <https://www.json.org/>
- Wiki: <https://en.wikipedia.org/wiki/JSON>
- ECMA-404, ISO/IEC 21778:2017, RFC 8259
- IPYNB notebook documents are stored in the JSON plain text format

# json module

- Linux distro may include json and simplejson.
- Recommended to use simplejson
- <https://github.com/simplejson/simplejson>
- Docs: <https://simplejson.readthedocs.io/en/latest/>
- PyPi: <https://pypi.org/project/simplejson/>
- simplejson version 1.0 released 2005-12-25
- simplejson version 3.16.1 released 2018-09-07
- try:

```
import simplejson as json
except ImportError:
    import json
```

# Python program flow

- Get the ipynb files in the current working directory.
- User selects the ipynb files to modify.
- User selects which metadata parameters to apply.
- Create a backup folder to copy files to before modification.
- **If metadata to be cleared** then loop through for each file:
  - backup ipynb file to backup folder.
  - open ipynb file as r+
  - Use json module to load the file to data variable.
  - clear the metadata: `data["cells"][i]['metadata'] = {}`
  - `f.seek(0)` # to reset file position to the beginning.
  - `json.dump(data, fin, indent=1)` # write data bck to file.

# Python program flow

- **If metadata to be selectively modified:**
  - Answer queries to select editable and deletable parameters for code and/or markdown cells.
- Loop through for each file:
  - backup ipynb file to backup folder.
  - open ipynb file as r+ (r/w positioned at beginning of file)
  - Use json module to load the file to data variable.
  - Set the metadata. E.g.:
    - `data["cells"][i]['metadata']['editable'] = True`
  - `f.seek(0)` # reset file position to the beginning.
  - `json.dump(data, fin, indent=1)` # write data back to file.
  - `f.truncate()` # remove remaining part

# ipynb highlevel layout.

```
{  
  "cells": [  
    {cell 1 dict}, {cell 2 dict}, {cell ... dict}  
  ],  
  "metadata": { ~15 overall attribute:value pairs },  
  "nbformat": 4,  
  "nbformat_minor": 2  
}
```

4 x Dictionary keywords

List of dictionaries

Each cell has a "metadata":{} dictionary

- After modification of every cells metadata in {cell ... dict}:

```
"metadata": {  
  "deletable": false,  
  "editable": false  
},
```

Attribute - value pairs

## Code to clear cell metadata:

```
if status[0]:
    print("\nClearing metadata of all cells in files:")
    for file_name in modify_list:
        shutil.copy(file_name, folder + file_name)
        print(file_name)
        with open(file_name, "r+") as f:
            data = json.load(f)
            total_cells = len(data["cells"])
            for i in range(total_cells):
                data["cells"][i]['metadata'] = {}

            f.seek(0) dict keywd, list pointer, dict keywd
            json.dump(data, f, indent=1)
            f.truncate() # remove remaining part

    sys.exit("Completed clearing metadata on all cells.")
```

# Code to set cell metadata:

```
# Setting metadata for each markdown and code cell.
print("\nSetting metadata for ipynb files:")
#print(status)
for file_name in modify_list:
    shutil.copy(file_name, folder + file_name)
    print(file_name)
    with open(file_name, "r+") as f:
        data = json.load(f)
        total_cells = len(data["cells"])
        for i in range(total_cells):
            if data["cells"][i]['cell_type'] == "markdown":
                data["cells"][i]['metadata']['editable'] = status[1]
                data["cells"][i]['metadata']['deletable'] = status[3]
            if data["cells"][i]['cell_type'] == "code":
                data["cells"][i]['metadata']['editable'] = status[2]
                data["cells"][i]['metadata']['deletable'] = status[4]
        f.seek(0) # reset file position
        json.dump(data, f, indent=1)
        f.truncate() # remove remaining
```

dict kw, list pointer,  
dict kw, key-value pair

# Run program. Clear metadata of all cells

```
$ python3 modify_notebook_metadata.py
```

Modify ipynb files by changing the metadata of each cell.

Choose the modification option you wish to apply to the ipynb file(s):

In the directory /home/ian/python/ipynb\_modification\_using\_json\_module, there are 1 ipynb files:

1. demo\_json\_1.ipynb

Enter the number of the file to modify or \* for all files: **1**

The file selected is: demo\_json\_1.ipynb

Are you sure you want to modify this file? [Y/n]:

Are there more files you wish to modify? [N/y]:

Clear all cell metadata [N/y]: **y**

Cell metadata will be cleared. Proceed? [Y/n]: **y**

A backup of the files before modification is in the folder:

/home/ian/python/ipynb\_modification\_using\_json\_module/backup-20190804-085836/

Clearing metadata of all cells in files:

demo\_json\_1.ipynb

Completed clearing metadata on all cells.



# Each cells metadata is cleared

Previously:

```
"metadata": {  
    "deletable": false,  
    "editable": false  
},
```

```
1 {  
2   "cells": [  
3     {  
4       "cell_type": "markdown",  
5       "metadata": {},  
6       "source": [  
7         "# Demo\n",  
8         "\n",  
9         "This is a demonstration file for ipynb. It contains this markdown cell  
and the python3 code cell below. "  
10      ]  
11    },  
12    {  
13      "cell_type": "code",  
14      "execution_count": null,  
15      "metadata": {},  
16      "outputs": [],  
17      "source": [  
18        "a = 42\n",
```

# Run Program. Set cells metadata

In the directory /home/ian/python/ipynb\_modification\_using\_json\_module, these are 1 ipynb files:

1. demo\_json\_1.ipynb

Enter the number of the file to modify or \* for all files: **1**

The file selected is: demo\_json\_1.ipynb

Are you sure you want to modify this file? [Y/n]:

Are there more files you wish to modify? [N/y]:

Clear all cell metadata [N/y]:

Set Markdown cell to be editable? [Y/n]: **n**

Set Code cell to be editable? [Y/n]: **y**

Set Markdown cell to be deletable? [Y/n]: **n**

Set Code cell to be deletable? [Y/n]: **n**

Markdown cells: Editable:False and Deletable:False

Code cells: Editable:True and Deletable:False

Proceed? [Y/n]: **y**

A backup of the files before modification is in the folder:

/home/ian/python/ipynb\_modification\_using\_json\_module/backup-20190804-102327/

Setting metadata for ipynb files:

demo\_json\_1.ipynb

Completed setting metadata on cells.


























# Cell metadata includes “editable” and “deletable”

```
3 {
4   "cell_type": "markdown",
5   "metadata": {
6     "editable": false,
7     "deletable": false
8   },
9   "source": [
10    "# Demo\n",
11    "\n",
12    "This is a demonstration file for ipynb. It contains this markdown cell\n",
13    "and the python3 code cell below. "
14  ],
15  {
16    "cell_type": "code",
17    "execution_count": null,
18    "metadata": {
19      "editable": true,
20      "deletable": false
21    },
```

# Yeah, big deal – so what?

Let's say you have 25 x ipynb files and each file has 10+ cells.

...Might be handy?

 rbe_00.ipynb	12.4 kB Jupyter Notebook
 rbe_01.ipynb	35.9 kB Jupyter Notebook
 rbe_02.ipynb	13.0 kB Jupyter Notebook
 rbe_03.ipynb	16.4 kB Jupyter Notebook
 rbe_04.ipynb	6.5 kB Jupyter Notebook
 rbe_05.ipynb	10.0 kB Jupyter Notebook
 rbe_06.ipynb	8.1 kB Jupyter Notebook
 rbe_07.ipynb	2.3 kB Jupyter Notebook
 rbe_08.ipynb	42.2 kB Jupyter Notebook
 rbe_09.ipynb	40.4 kB Jupyter Notebook
 rbe_10.ipynb	17.6 kB Jupyter Notebook
 rbe_11.ipynb	3.7 kB Jupyter Notebook
 rbe_12.ipynb	10.9 kB Jupyter Notebook
 rbe_13.ipynb	8.4 kB Jupyter Notebook
 rbe_14.ipynb	37.8 kB Jupyter Notebook
 rbe_15.ipynb	47.0 kB Jupyter Notebook
 rbe_16.ipynb	19.2 kB Jupyter Notebook
 rbe_17.ipynb	18.0 kB Jupyter Notebook
 rbe_18.ipynb	53.5 kB Jupyter Notebook
 rbe_19.ipynb	38.9 kB Jupyter Notebook
 rbe_20.ipynb	42.8 kB Jupyter Notebook
 rbe_21.ipynb	18.1 kB Jupyter Notebook
 rbe_22.ipynb	3.9 kB Jupyter Notebook
 rbe_23.ipynb	2.6 kB Jupyter Notebook
 rbe_24.ipynb	4.7 kB Jupyter Notebook

# Set metadata in all cells in 25 x files

**\$ modify\_notebook\_metadata**



Modify ipynb files by changing the metadata of each cell.

Choose the modification option you wish to apply to the ipynb file(s):

In the directory /home/ian/rust/rbe, these are 25 ipynb files:

1. rbe\_00.ipynb
2. rbe\_01.ipynb
- ...snip...
24. rbe\_23.ipynb
25. rbe\_24.ipynb

Enter the number of the file to modify or \* for all files: \*

Selected all files

Clear all cell metadata [N/y]:

Set Markdown cell to be editable? [Y/n]: **n**

Set Code cell to be editable? [Y/n]: **y**

Set Markdown cell to be deletable? [Y/n]: **n**

Set Code cell to be deletable? [Y/n]: **n**

# Set metadata in all cells in 25 x files

Markdown cells: Editable:False and Deletable:False

Code cells: Editable:True and Deletable:False

Proceed? [Y/n]: **y**

A backup of the files before modification is in the folder:

/home/ian/rust/rbe/backup-20190804-104541/

Setting metadata for ipynb files:

rbe\_00.ipynb

rbe\_01.ipynb

rbe\_02.ipynb

...snip...

rbe\_23.ipynb

rbe\_24.ipynb

Completed setting metadata on cells.



2/2

```
"cells": [  
  {  
    "cell_type": "markdown",  
    "metadata": {  
      "editable": false,  
      "deletable": false  
    },  
  },  
  {  
    "cell_type": "code",  
    "execution_count": null,  
    "metadata": {  
      "editable": true,  
      "deletable": false  
    },  
  },  
]
```

```
{  
  "cell_type": "code",  
  "execution_count": null,  
  "metadata": {  
    "editable": true,  
    "deletable": false  
  },  
}
```

...And if you do want to “restore” the files so that each cell has no metadata...

(Although these no metadata files already exist in a backup folder)

...then...

# Clear cell metadata in 25 x files

**\$ modify\_notebook\_metadata**



Modify ipynb files by changing the metadata of each cell.

Choose the modification option you wish to apply to the ipynb file(s):

In the directory /home/ian/rust/rbe, these are 25 ipynb files:

1. rbe\_00.ipynb
2. rbe\_01.ipynb
- ...snip...
24. rbe\_23.ipynb
25. rbe\_24.ipynb

Enter the number of the file to modify or \* for all files: \*

Selected all files

Clear all cell metadata [N/y]: **y**

Cell metadata will be cleared. Proceed? [Y/n]: **y**



# Clear cell metadata in 25 x files

A backup of the files before modification is in the folder:

/home/ian/rust/rbe/backup-20190804-104118/

Clearing metadata of all cells in files:

rbe\_00.ipynb

rbe\_01.ipynb

rbe\_02.ipynb

...snip...

rbe\_23.ipynb

rbe\_24.ipynb

Completed clearing metadata on all cells.

```
"cells": [  
  {  
    "cell_type": "markdown",  
    "metadata": {},  
    "source": []  
  },  
  ...  
]
```

```
"cell_type": "code",  
"execution_count": null,  
"metadata": {},  
"source": []  
}
```

```
your_query = input("Questions? ")
```

```
the.__end__(".")
```