In [12]:

```
from IPython.display import Image
i = Image(filename='logo.png')
i
```

Out[12]:

1.) 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.

2.) In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.

The ability to execute code from the browser.

Displaying the result of computation using rich media representations, such as HTML, LaTeX, PNG, SVG, etc.

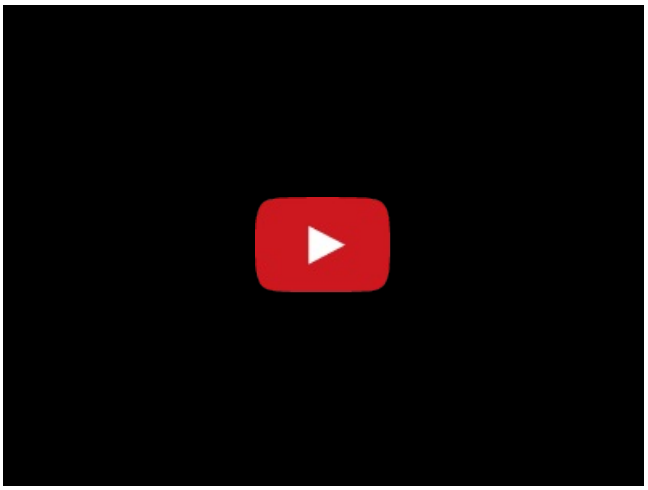
In-browser editing for rich text using the Markdown markup language.

The ability to easily include mathematical notation within markdown cells using LaTeX, and rendered natively by MathJax.

In [11]:

```
from IPython.display import YouTubeVideo
YouTubeVideo('VaV10VNZCLA')
```

Out[11]:



4.) JSON (JavaScript Object Notation) is a lightweight format that is used for data interchanging.

5.) Yes because it is a file that can be edited.

6.) IPython notebook viewer.

In []:

```
7. ipython notebook
http://127.0.0.1
8888
```

8.) --port

9.) --no-browser

10.) Clicking on the new notebook button.

11.) A kernel receives execution instructions from clients and communicates the results back to them.

12.) Notebooks with an active kernel has a shutdown button while notebooks without an active kernel has a delete button.

13.) 2015-02-12 09:41:40.589 [NotebookApp] Kernel started: d14a94ca-b5bf-4071-a4d0-29a31935438e

14.) ipython qtconsole --existing 87f7d2c0

15.) Notebook name - It is the name of the notebook that is displayed at the top of the page; Clicking it allows the user to change the name of the notebook.

Menu bar - It presents different options that may be used to manipulate the way the notebook functions.

Toolbar - It gives a quick way of performing that most-used operations in the notebook.

Code cell - The default type of cell.

16.) A cell is a multi-line text input field

17.) A cell can be executed by using Shift-Enter or clicking the Play button or Cell->Run in the menu bar.

18.)Code cells - this is where code is entered

Markdown cells - It allows the user to format the text i.e. rich text

Raw cells - Text entered in this cell are not evaluated by the notebook, it is unformatted in any way.

Heading cells - Used for entering headings in the notebook

17.) A cell can be executed by using Shift-Enter or clicking the Play button or Cell->Run in the menu bar.

18.) Code cells - this is where code is entered Markdown cells - It allows the user to format the text i.e. rich text Raw cells - Text entered in this cell are not evaluated by the notebook, it is unformatted in any way. Heading cells - Used for entering headings in the notebook

In [1]:

```
from IPython.display import Image
i = Image(filename='sut.png')
i
```

Out[1]:

Here is the complete set of keyboard shortcuts available:

Shortcut	Action
Shift-Enter	run cell
Ctrl-Enter	run cell in-place
Alt-Enter	run cell, insert below
Ctrl-m x	cut cell
Ctrl-m c	copy cell
Ctrl-m v	paste cell
Ctrl-m d	delete cell
Ctrl-m z	undo last cell deletion
Ctrl-m -	split cell
Ctrl-m a	insert cell above
Ctrl-m b	insert cell below
Ctrl-m o	toggle output
Ctrl-m O	toggle output scroll
Ctrl-m l	toggle line numbers
Ctrl-m s	save notebook
Ctrl-m j	move cell down
Ctrl-m k	move cell up
Ctrl-m y	code cell
Ctrl-m m	markdown cell
Ctrl-m t	raw cell
Ctrl-m 1-6	heading 1-6 cell
Ctrl-m p	select previous
Ctrl-m n	select next
Ctrl-m i	interrupt kernel
Ctrl-m .	restart kernel
Ctrl-m h	show keyboard shortcuts

In [3]:

```
i = Image(filename='pm.png')
i
```

Out[3]:

Command	Description
`%quickref`	Display the IPython Quick Reference Card
`%magic`	Display detailed documentation for all of the available magic commands
`%debug`	Enter the interactive debugger at the bottom of the last exception traceback
`%hist`	Print command input (and optionally output) history
`%pdb`	Automatically enter debugger after any exception
`%paste`	Execute pre-formatted Python code from clipboard
`%cpaste`	Open a special prompt for manually pasting Python code to be executed
`%reset`	Delete all variables / names defined in interactive namespace
`%page` *`OBJECT`*	Pretty print the object and display it through a pager
`%run` *`script.py`*	Run a Python script inside IPython
`%prun` *`statement`*	Execute *`statement`* with `cProfile` and report the profiler output
`%time` *`statement`*	Report the execution time of single statement
`%timeit` *`statement`*	Run a statement multiple times to compute an emsemble average execution time. Useful for timing code with very short execution time
`%who, %who_ls, %whos`	Display variables defined in interactive namespace, with varying levels of information / verbosity
`%xdel` *`variable`*	Delete a variable and attempt to clear any references to the object in the IPython internals

21.) Kernel->Interrupt Ctrl-M I

22.) Kernel->Restart Ctrl-M .

23.) Shift-enter(Run cell) - this executes the code inside the cell then jumps to the next cell Ctrl-Enter(Run cell in-place) - runs the cell like in terminal mode; executed the cell but the cursor remains inside the cell. Alt-Enter(Run cell, insert below) - Runs the cell then inserts a new cell below Esc - goes into command mode Enter - inserts a new line

24.) %matplotlib

25.) The matplotlib backend

26.) The gtk backend is a user-interface for matplotlib

27.) The inline backend

28.) ipython locate

29.) ipython profile create

30.) ipython nbconvert --to FORMAT notebook.ipynb

31.) The default output format is HTML

32.) Latex, slideshow, markdown, RestructuredText, Python script

33.) By setting the NotebookApp.password configurable

34.) IPython.lib.security.passwd():

35.) By setting NotebookApp.password

36.) It is a good idea to use SSL so that the password is not sent unencrypted by your browser.

37.) openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout mycert.pem -out mycert.pem The command above writes the certificate to the same file

38.) By using the following commands: ipython profile create nbserver

39.) Unauthorized clients Unauthorized engines Unauthorized controllers

40.) The notebook server can be protected by using a simple single password. It can be set by setting the NotebookApp.password configurable. A hashed password can be created using the function IPython.lib.security passwd():

In []: