

## บทที่ 2

# แนะนำเครื่องมือพัฒนาโปรแกรม ด้วยภาษาไพธอน

บทที่ผ่านมาได้แนะนำให้รู้จักกับภาษาไพธอน ประวัติความเป็นมา รวมถึงข้อดีต่าง ๆ ของภาษาไพธอน สำหรับในบทนี้จะขอแนะนำ วิธีการติดตั้งตัวแปรภาษาไพธอน และเครื่องมือที่ใช้พัฒนาโปรแกรมภาษาไพธอน

โดยพื้นฐานแล้วไฟล์ไพธอนสคริปต์ (Python Script) คือไฟล์ข้อความที่ไม่มีการเข้ารหัส (Plain Text) ที่บรรจุคำสั่งในภาษาไพธอนและบรรทึกโดยใช้นามสกุล .py ซึ่งสามารถใช้โปรแกรมอ่านหรือแก้ไขข้อความ (Text Editor) อย่างเช่น Notepad ในการจัดการได้ นั่นหมายความว่า เราต้องการพิยง Text Editor และตัวแปลงภาษาไพธอนก็สามารถพัฒนาโปรแกรมภาษาไพธอนได้แล้ว ทั้งนี้เครื่องมือ Text Editor อย่างเช่น Vim, Atom, Visual Studio Code<sup>1</sup>, Sublime Text<sup>2</sup> หรือ ชุดเครื่องมือพัฒนาโปรแกรม (Integrated Development Environment) อย่างเช่น PyCharm<sup>3</sup> ซึ่งมีให้เลือกด้านโน๊ลเดนนำมานางานทั้งแบบมืออาชีพและแบบพรี (ชั้นคุณสมบัติต่างกัน) และ Spyder<sup>4</sup> ก็ได้เข้ามามีบทบาทช่วยเหลือผู้พัฒนาโปรแกรมให้ทำงานได้อย่างมีประสิทธิภาพเพิ่มขึ้น

สำหรับเครื่องมือที่นิยมนิยมนำมาพัฒนาโปรแกรมด้าน Data Science และนำมาใช้งานโดยไม่เสียค่าใช้จ่ายคือ Jupyter Notebook หรือ Jupyter Lab ซึ่งเป็นอีกเครื่องมือที่ได้รับการพัฒนาให้ใช้งานได้สะดวกขึ้น โดยสามารถใช้งานผ่านระบบคลาวด์ (Cloud) ได้

<sup>1</sup>Microsoft Visual Studio Code, ดูเพิ่มเติม <https://code.visualstudio.com>

<sup>2</sup>Sublime Text: Text Editing, Done Right, ดูเพิ่มเติม <https://www.sublimetext.com>

<sup>3</sup>PyCharm: The Python IDE for Professional Developers, ดูเพิ่มเติม <https://www.jetbrains.com/pycharm/>

<sup>4</sup>Spyder: The Scientific Python Development Environment, ดูเพิ่มเติม <https://www.spyder-ide.org>

## 2.1 การติดตั้งตัวแปลงภาษาไพธอน

ในที่นี้เราจะแนะนำวิธีการติดตั้งตัวแปลงภาษาไพธอนด้วยวิธีอย่างง่ายบนระบบปฏิบัติการ Microsoft Windows 10 , Ubuntu Linux 20.04 LTS และ macOS Big Sur

### 2.1.1 การติดตั้งบนระบบปฏิบัติการ Microsoft Windows 10

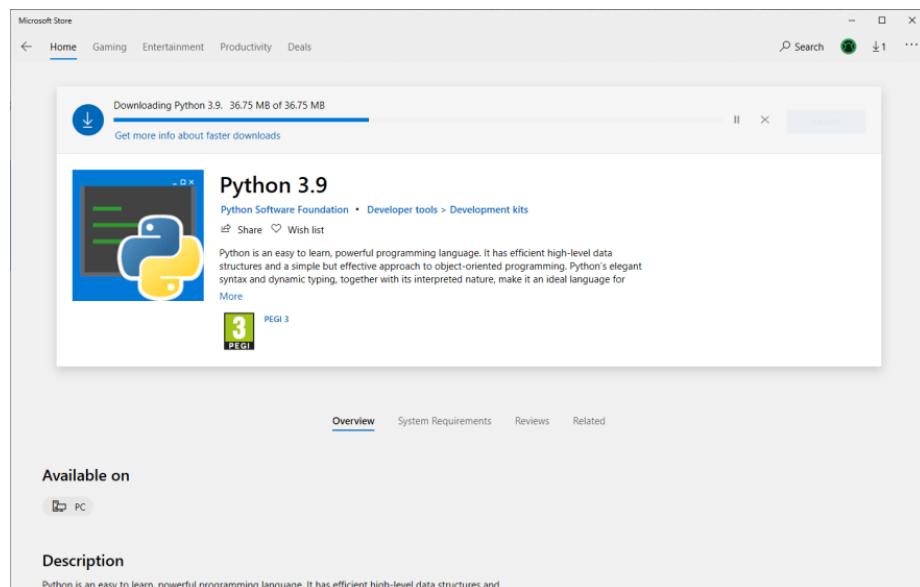
ทำการติดตั้งตัวแปลงภาษาไพธอนผ่าน Microsoft Store ดังนี้

1. เปิดโปรแกรม Microsoft Store หากผู้อ่านไม่ทราบตำแหน่งของโปรแกรมนี้ ให้ทำการคลิกที่ Start menu (ไอคอนรูป Windows บริเวณด้านล่างทางซ้าย) แล้วพิมพ์ “Microsoft Store” (ไม่รวมเครื่องหมาย ”) จากนั้นจึงคลิกที่ไอคอนเพื่อเปิด Microsoft Store
2. เมื่อ Microsoft Store เปิดขึ้นมาให้ทำการค้นหา (Search) จากแถบเมนูบริเวณด้านบน ข้ามโดยการพิมพ์ “Python” (ไม่รวมเครื่องหมาย ”)
3. ให้ทำการเลือกเวอร์ชันตัวแปลงภาษาไพธอนที่ต้องการติดตั้ง (แนะนำให้เลือกเวอร์ชันล่าสุด นอกเสียจากว่ามีความจำเป็นในการต้องเลือกใช้งานเวอร์ชันก่อนหน้า โดยข้อมูลณ วันที่เขียนต้นฉบับนี้คือ ไพธอนเวอร์ชัน 3.9.5) หลังจากเลือกเวอร์ชันได้แล้วให้ทำการคลิก Get เพื่อเริ่มการติดตั้ง
4. หลังจากการดาวน์โหลดและติดตั้งเสร็จสิ้น ให้ทำการเปิดโปรแกรม Command Prompt หรือ PowerShell หรือ Terminal (หากไม่ทราบตำแหน่งโปรแกรมให้ทำการค้นหาผ่าน Start menu) แล้วทำการพิมพ์คำสั่ง

```
python --version
```

เพื่อยืนยันความสมบูรณ์ของการติดตั้ง โดยผลลัพธ์จากคำสั่งควรจะเป็นไพธอนเวอร์ชันที่เราได้ทำการเลือกไว้

5. ในกระบวนการติดตั้งตัวแปลงภาษาไพธอนผ่าน Microsoft Store นั้นจะรวมถึงโปรแกรม pip ที่ใช้ในการจัดการแพคเกจที่ไม่ได้ถูกติดตั้งมากับตัวแปลงภาษาได้ใน



รูป 2.1: การดาวน์โหลดและติดตั้งตัวแปลงภาษาโปรแกรมผ่าน Microsoft Store

อนาคต ให้ทำการเปิดโปรแกรม Command Prompt หรือ PowerShell หรือ Terminal และทำการพิมพ์คำสั่ง

```
pip --version
```

เพื่อยืนยันความสมบูรณ์ของการติดตั้ง โดยผลลัพธ์จากคำสั่งควรจะเป็นเวอร์ชันของ pip ที่ได้ทำการติดตั้งไป

### 2.1.2 การติดตั้งบนระบบปฏิบัติการ Ubuntu Linux 20.04 LTS

ระบบปฏิบัติการ Ubuntu Linux 20.04 LTS นั้นถูกพัฒนาอยู่บนฐานของ Debian Linux ซึ่งมาพร้อมกับตัวแปลงภาษา Python เวอร์ชัน 3+ อยู่แล้ว ทั้งนี้เราสามารถใช้คำสั่งในการอัปเดทเวอร์ชันได้ดังนี้

1. เปิดโปรแกรม Terminal และใช้คำสั่งต่อไปนี้

```
sudo apt update
sudo apt -y upgrade
```

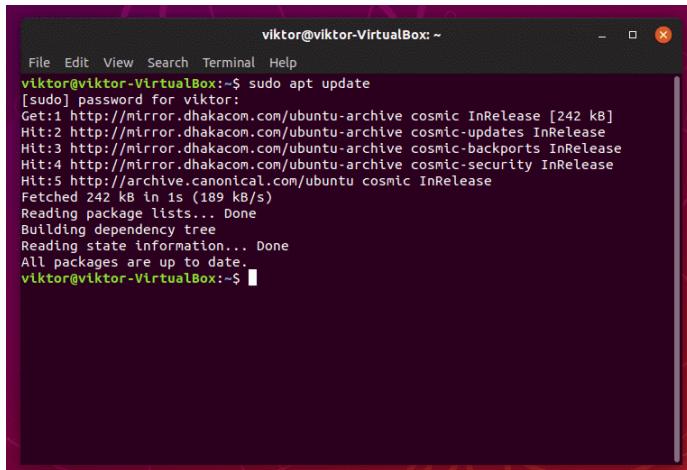
เพื่อทำการอัปเกรด (จำเป็นต้องกรอกรหัสผ่านของผู้ใช้ที่มีสิทธิในการเข้าถึง)

2. หลังจากการดาวน์โหลดและติดตั้งเสร็จสิ้น ให้ทำการใช้คำสั่ง

```
python3 --version
```

เพื่อยืนยันความสมบูรณ์ของการติดตั้ง โดยผลลัพธ์จากคำสั่งควรจะเป็นไฟล์เวอร์ชันล่าสุด

3. เพื่อความสะดวกในการติดตั้งแพคเกจของไฟรอนในอนาคต เราจะทำการติดตั้งโปรแกรมจัดการแพคเกจของไฟรอน (Python Package Manager) pip โดยใช้คำสั่ง



รูป 2.2: การใช้โปรแกรม Terminal บน Ubuntu Linux 20.04 LTS

```
sudo apt install -y python3-pip
```

แล้วทำการพิมพ์คำสั่ง

```
pip3 --version
```

เพื่อยืนยันความสมบูรณ์ของการติดตั้ง โดยผลลัพธ์จากคำสั่งควรจะเป็นเวอร์ชันของ pip ที่ได้ทำการติดตั้งไป

### 2.1.3 การติดตั้งบนระบบปฏิบัติการ macOS Big Sur

ระบบปฏิบัติการ macOS Big Sur มาพร้อมกับตัวแปลงภาษา Python เวอร์ชัน 3+ อยู่แล้ว ทั้งนี้ความสามารถติดตั้งเวอร์ชันที่ใหม่กว่าได้ผ่าน Homebrew<sup>5</sup> ซึ่งเป็น Package Manager ที่รองรับระบบปฏิบัติการ macOS ดังนี้

---

<sup>5</sup>Homebrew: The Missing Package Manager for macOS, ดูเพิ่มเติม <https://brew.sh>

1. เปิดโปรแกรม Terminal และใช้คำสั่งต่อไปนี้

```
/bin/bash -c "$(curl -fsSL https://
↪ raw.githubusercontent.com/Homebrew/install/
↪ HEAD/install.sh)"
```

เพื่อทำการติดตั้ง Homebrew (จำเป็นต้องกรอกรหัสผ่านของผู้ใช้ที่มีสิทธิในการเข้าถึง) โดยทำการตามคำแนะนำที่ปรากฏ

2. หลังจากการดาวน์โหลดและติดตั้งเสร็จสิ้น (หากมีการแจ้งเตือนภายนอกหลังการติดตั้ง ให้ทำการตามคำแนะนำที่ได้กำหนดไว้) ให้ทำการใช้คำสั่ง

```
brew update
brew search python
```

เพื่อทำการอัปเดตฐานข้อมูลของ Homebrew และ ค้นหาเวอร์ชันของ Python ที่สามารถติดตั้งได้

3. เมื่อทำการเลือกเวอร์ชันที่จะติดตั้งได้แล้วให้ใช้คำสั่ง

```
brew install python@3.x
```

เพื่อทำการติดตั้ง โดยต้องแทนที่ x ด้วยเลขของ Python เวอร์ชันที่ค้นหาพบจากขั้นตอนก่อนหน้านี้

4. หลังจากการดาวน์โหลดและติดตั้งเสร็จสิ้น (หากมีการแจ้งเตือนภายนอกหลังการติดตั้ง ให้ทำการตามคำแนะนำที่ได้กำหนดไว้) ให้ใช้คำสั่ง

```
python3 --version
pip3 --version
```

```
epsilon@VEKTOR:~ 675 23:13:27
>>> brew update
Already up-to-date.
epsilon@VEKTOR:~ 675 23:13:42
>>> brew search python
=> Formulae
app-engine-python      ipython          python-tk@3.9      reorder-python-imports
boost-python            micropython     python-yq        wxpython
boost-python3           ptpython        python@3.7
bpython                 python-markdown  python@3.8
gst-python              python-tabulate  python@3.9 ✓
=> Casks
awips-python            mysql-connector-python

If you meant "python" specifically:
It was migrated from homebrew/cask to homebrew/core.
epsilon@VEKTOR:~ 676 23:13:55
>>> brew install python@3.9
```

รูป 2.3: การติดตั้งตัวแปลภาษาไพธอนผ่าน Homebrew บน macOS Big Sur

เพื่อยืนยันความสมบูรณ์ของการติดตั้ง Python และ pip (ติดตั้งมาพร้อมกัน) โดยผลลัพธ์จากคำสั่งควรจะเป็นเวอร์ชันที่เราได้ทำการเลือกไว้

#### 2.1.4 การติดตั้งผ่าน pyenv

สำหรับระบบปฏิบัติการ macOS Big Sur เราปฏิบัติการติดตั้งตัวแปลภาษา Python ได้หลายเวอร์ชัน และเลือกใช้งานได้ตามความเหมาะสมผ่าน pyenv ทั้งนี้กระบวนการติดตั้งนั้นค่อนข้างซับซ้อนแต่ก็แลกมาด้วยความยืดหยุ่นที่มากกว่าในการพัฒนา

วิธีการต่อไปนี้ไม่แนะนำสำหรับผู้เริ่มต้น หรือผู้ที่ไม่เคยดูในการใช้งาน **Command Line**

1. อัปเดท Homebrew และติดตั้งแพคเกจที่จำเป็น

```
brew update
brew install zlib sqlite bzip2 libiconv libzip
```

2. ติดตั้ง pyenv<sup>6</sup> ผ่าน Homebrew ด้วยคำสั่ง

```
brew install pyenv
```

3. หลังจากการดาวน์โหลดและติดตั้งเสร็จสิ้น ให้ใช้คำสั่ง

```
echo -e $'if command -v pyenv 1>/dev/null 2>&1;
then\n  export PYENV_ROOT="$HOME/.pyenv"\n  export PATH="$PYENV_ROOT/bin:$PATH"\n  eval
"$(pyenv init --path)"\n  eval "$(pyenv
init -)"'\nfi' >> ~/.zshrc
```

เพื่อตั้งค่าการทำงานของ pyenv อัตโนมัติทุกครั้งที่เข้าใช้งาน

---

<sup>6</sup>pyenv: Simple Python Version Management, ดูเพิ่มเติม <https://github.com/pyenv/pyenv>

## 4. ใช้คำสั่ง

```
pyenv install --list
```

เพื่อดูว่ามี Python ใดบ้างที่เราสามารถติดตั้งได้

## 5. หลังจากเลือกเวอร์ชันที่ต้องการได้แล้วให้ทำการใช้คำสั่ง

```
pyenv install x.y.z  
pyenv rehash
```

เพื่อทำการติดตั้ง โดยที่ x.y.z คือตัวเลขเวอร์ชันของ Python ที่ได้มาจากการติดตั้งตอนที่แล้ว (เช่น 3.9.5 เป็นต้น)

## 6. ใช้คำสั่ง

```
pyenv versions
```

เพื่อดูว่ามี Python ใดบ้างที่ได้ติดตั้งไปแล้ว โดยที่ผลลัพธ์ที่ได้หากมีเครื่องหมาย \* อยู่หน้าเวอร์ชันใด หมายถึงเวอร์ชันนั้นได้ถูกเลือกใช้งาน (ค่าเริ่มต้นจะเป็น systems)

## 7. เราสามารถเลือกเวอร์ชันของไฟรอนที่ต้องการได้โดยใช้คำสั่ง

```
pyenv global x.y.z
```

## 8. เพื่อยืนยันความถูกต้อง ให้ใช้คำสั่ง

```
python --version  
pip --version
```

โดยผลลัพธ์จากการคำสั่งควรจะเป็นเว่อร์ชันที่เราได้ทำการเลือกไว้

คำสั่งของ pyenv โดยละเอียดสามารถสืบค้นได้จาก Repository ของผู้พัฒนาโดยตรงที่ <https://github.com/pyenv/pyenv>

## 2.2 การใช้งานไพธอนสคริปต์

เราจะเป็นต้องเตรียมไพธอนสคริปต์ผ่าน Text Editor เช่น Notepad, Visual Studio Code หรือ Sublime Text เป็นต้น ผู้อ่านสามารถเลือกใช้ได้ตามความต้องการ

1. เปิด Text Editor ที่ผู้อ่านคุ้นเคย และสร้างไฟล์ขึ้นมาใหม่
2. พิมพ์ข้อความ

```
print('Hello World')
```

แล้วทำการบันทึกไฟล์ (Save หรือ Save as) โดยให้มีนามสกุลเป็น .py เช่น hello.py ไฟล์ที่ได้นี้จะเรียกว่าไพธอนสคริปต์

3. หลังจากบันทึกไพธอนสคริปต์เรียบร้อยแล้ว เราสามารถสั่งให้สคริปต์นี้ทำงานได้โดยการเปิด Terminal และใช้คำสั่ง

```
python hello.py
```

หรือ

```
python3 hello.py
```

ขึ้นอยู่กับว่าตัวแปลงภาษาไพธอนถูกติดตั้งด้วยวิธีการใดในหัวข้อ 2.1



A screenshot of the Sublime Text editor window. The title bar says 'hello.py'. The main text area contains the Python code: `print('Hello World')`. The code is highlighted in green, and the word 'print' is underlined with a blue cursor.

รูป 2.4: การสร้างไฟรอนสคริปต์ `hello.py` บน Sublime Text



A screenshot of a terminal window. The command `python hello.py` is entered, followed by the output `Hello World`. The terminal shows two entries: the command and the output. The timestamp in the top right corner is 19:29:33.

รูป 2.5: การใช้งานไฟรอนสคริปต์ `hello.py`

## 2.3 รู้จักกับเครื่องมือ Jupyter Notebook

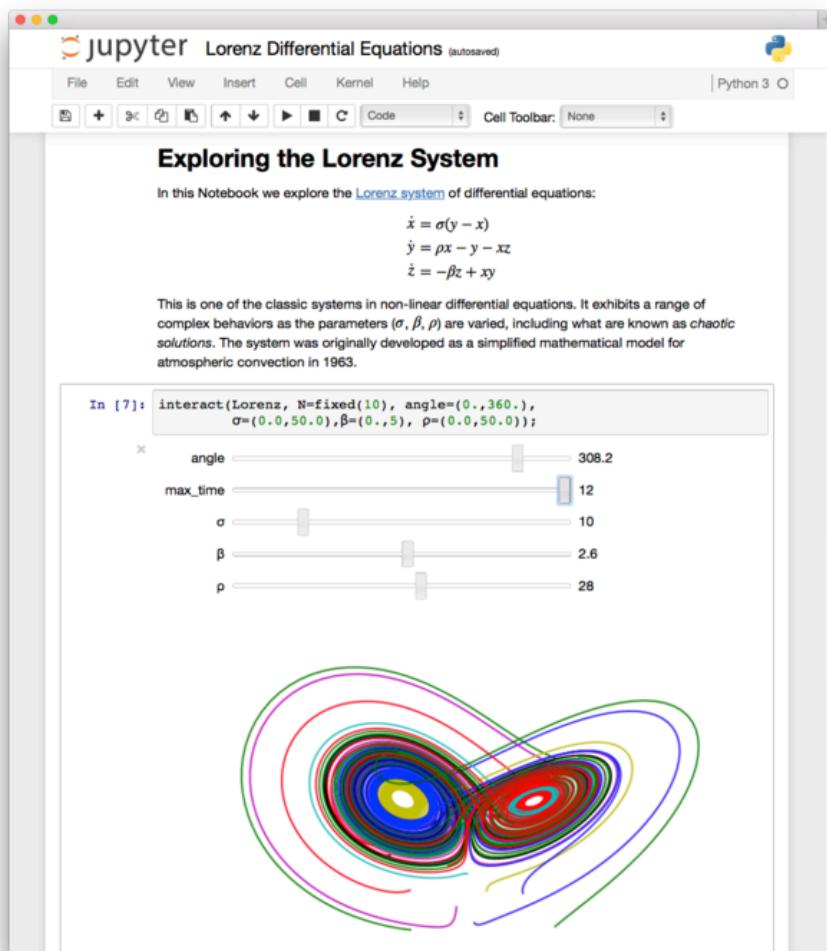


Jupyter Notebook เป็นเว็บแอปพลิเคชันแบบเปิดเผยแพร่สตัน (open-source web application) สำหรับพัฒนาโปรแกรมได้มากกว่า 40 ภาษาผ่านเว็บбраузर รองรับการสร้างและส่งต่อ (share) เอกสารที่มี โปรแกรมสคริปต์ สมการ คณิตศาสตร์ และกราฟฟิก โดยชื่อ Jupyter นี้มีที่มาจากการพัฒนาโดยทีมวิจัยในสถาบันเทคโนโลยี麻省理工学院 (MIT) ในสังกัด MIT ที่มุ่งเน้นการนำภาษา Python มาใช้ในการวิเคราะห์ข้อมูล

โปรเจกต์นี้ต้องการรองรับอันได้แก่ Julia, Python และ R เครื่องมือนี้เป็นส่วนหนึ่งของ Project Jupyter โดยเป็นโปรเจกต์ที่แยกออกจาก Ipython เมื่อปี 2014 โดย Fernando Pérez ซึ่งเป็นเครื่องมือเขียนคำสั่งโปรแกรมภาษาไพธอนแบบ Interactive Shell และรองรับการแสดงผลข้อความ กราฟฟิก และสมการคณิตศาสตร์

Jupyter Notebook ถูกนิยมใช้ในงาน การทำความสะอาดและแปลงข้อมูล (Data Cleaning and Transformation) การจำลองสถานการณ์เชิงตัวเลข (Numerical Simulation) การทำแบบจำลองทางคณิตศาสตร์ (Mathematical Modeling) การแสดงกราฟฟิก ของข้อมูล (Data Visualization) และ การเรียนรู้ของเครื่องจักร (Machine Learning) องค์ประกอบของเครื่องมือ Jupyter Notebook มีอยู่ด้วยกัน 4 ส่วนคือ

1. Notebook Document หรือเรียกสั้น ๆ ว่า Notebook คือ หน้าเว็บбраузอร์แต่ละหน้าที่ใช้ทำการเขียนคำสั่งโปรแกรม เก็บข้อมูล คำอธิบาย รูปภาพ กราฟ รวมไปถึงผลลัพธ์ที่ได้จากการประมวลผลคำสั่งโปรแกรม
2. Jupyter Notebook App คือ แอปพลิเคชัน server-client ซึ่งช่วยให้ทำการเปิด Notebook ผ่านเว็บбраузอร์โดยไม่ต้องใช้อินเตอร์เน็ต หรือจะติดตั้งให้ทำงานแบบ Server ซึ่งสามารถเข้าใช้งานพร้อมกันหลาย ๆ คนก็ได้
3. Kernel คือ ตัวประมวลผลคำสั่งโปรแกรม (Computation engine) ของแต่ละภาษา ที่มีอยู่ใน Notebook ซึ่งเครื่องมือ Jupyter Notebook ได้ติดตั้ง IPython kernel สำหรับประมวลผลคำสั่งภาษาไพธอนไว้ให้เรียบร้อยแล้ว สำหรับภาษาอื่น ๆ ต้องติดตั้งเพิ่มเติมในภายหลัง



รูป 2.6: ตัวอย่างการใช้งาน Jupyter Notebook ในการทำแบบจำลองทางคณิตศาสตร์

4. Notebook Dashboard คือ หน้าเว็บбраузர์แรกหลังจากสั่งให้ Jupyter Notebook App ทำงาน ซึ่งอ่านวิความละเอียดให้กับผู้ใช้งานจัดการสร้างไฟล์เดอร์จัดเก็บไฟล์คำสั่งโปรแกรมที่ถูกเขียน Notebook หรือตรวจสอบไฟล์ที่กำลังประมวลผลคำสั่งโปรแกรมอยู่ ณ ขณะนั้น หรือการเปลี่ยนชื่อไฟล์หรือไฟล์เดอร์รวมไปถึงการลบไฟล์หรือไฟล์เดอร์ด้วย

### 2.3.1 การติดตั้งและการเปิดใช้งาน Jupyter Notebook

Jupyter Notebook สามารถติดตั้งผ่านคำสั่ง pip ได้โดยการเปิด Terminal และใช้คำสั่งต่อไปนี้

```
pip install --upgrade pip
pip install jupyter
```

หลังจากที่ได้ดาวน์โหลดและติดตั้งเครื่องมือ Jupyter Notebook สำหรับเขียนโปรแกรมด้วยภาษาไพธอนเรียบร้อยแล้ว เราสามารถเปิดการใช้งาน Jupyter Notebook ได้โดยใช้คำสั่ง

```
jupyter notebook
```

หรือ

```
python -m notebook
```

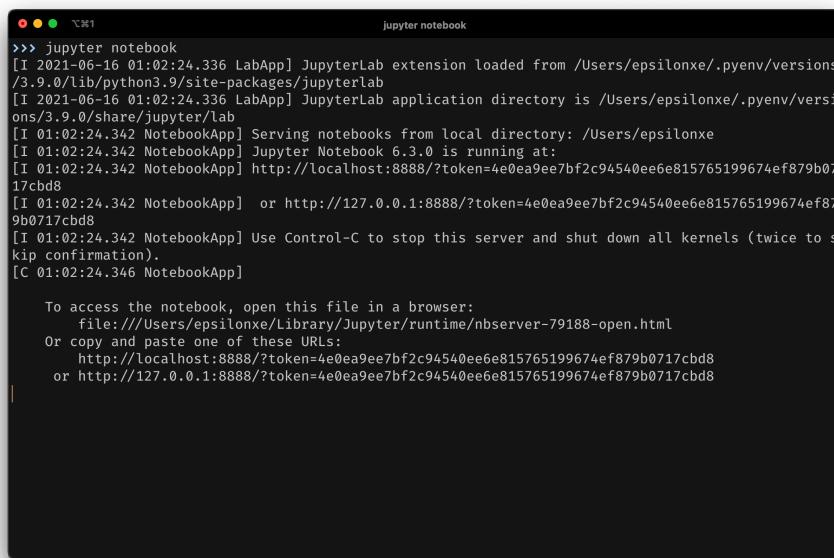
หลังจากนั้นหน้าต่างของเว็บбраузร์จะเปิดไปที่หน้า <http://localhost:8888/tree> สังเกตได้ว่ามีเมนูต่าง ๆ ให้ได้เลือกใช้งาน ดังรูป 2.8 โดยในหน้าเว็บนี้จะมีส่วนที่ติดต่อผู้ใช้ (User Interface) ที่ทำหน้าที่ต่าง ๆ ดังตาราง 2.1

ทั้งนี้เรายังสามารถใช้งาน Jupyter Notebook ผ่านระบบ Cloud ได้เช่น Google Colab<sup>7</sup>, Jupyter Mybinder<sup>8</sup>, COCALC Jupyter Notebook<sup>9</sup> เป็นต้น

<sup>7</sup>Google Colab: <https://colab.research.google.com>

<sup>8</sup>Jupyter Mybinder: <https://jupyter.org/try>

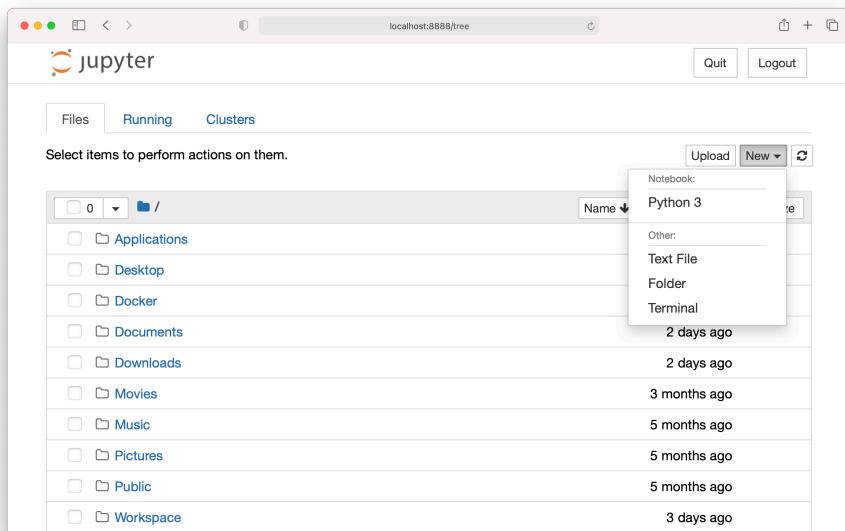
<sup>9</sup>COCALC Jupyter Notebook: <https://cocalc.com/doc/jupyter-notebook.html>



```
jupyter notebook
[I 2021-06-16 01:02:24.336 LabApp] JupyterLab extension loaded from /Users/epsilonxe/.pyenv/versions/3.9.0/lib/python3.9/site-packages/jupyterlab
[I 2021-06-16 01:02:24.336 LabApp] JupyterLab application directory is /Users/epsilonxe/.pyenv/versions/3.9.0/share/jupyter/lab
[I 01:02:24.342 NotebookApp] Serving notebooks from local directory: /Users/epsilonxe
[I 01:02:24.342 NotebookApp] Jupyter Notebook 6.3.0 is running at:
[I 01:02:24.342 NotebookApp] http://localhost:8888/?token=4e0ea9ee7bf2c94540ee6e815765199674ef879b0717cbd8
[I 01:02:24.342 NotebookApp] or http://127.0.0.1:8888/?token=4e0ea9ee7bf2c94540ee6e815765199674ef879b0717cbd8
[I 01:02:24.342 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 01:02:24.346 NotebookApp]

To access the notebook, open this file in a browser:
  file:///Users/epsilonxe/Library/Jupyter/runtime/nbserver-79188-open.html
Or copy and paste one of these URLs:
  http://localhost:8888/?token=4e0ea9ee7bf2c94540ee6e815765199674ef879b0717cbd8
or http://127.0.0.1:8888/?token=4e0ea9ee7bf2c94540ee6e815765199674ef879b0717cbd8
```

រូប 2.7: การរើរឹងកិច្ចការប្រើប្រាស់ Jupyter Notebook



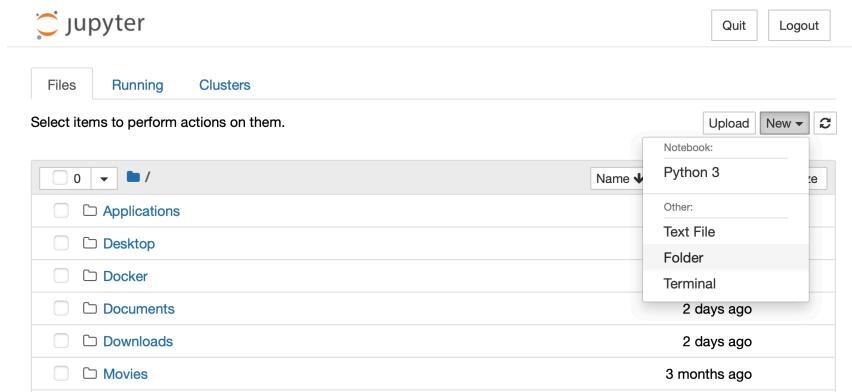
ສູ່ 2.8: Notebook Dashboard ຢ່າງ Jupyter Notebook

ส่วนติดต่อผู้ใช้ การใช้งาน	
Files	แสดงชื่อ File และ Folder ที่ได้สร้างไว้สำหรับเก็บ File งาน
Running	แสดงชื่อ File ต่าง ๆ ที่กำลังทำงานอยู่
Clusters	ใช้สำหรับกรณีที่มีการติดตั้ง Jupyter Notebook ไว้หลายเครื่องให้ประมวลผลข้อมูลแบบ
Logout	ใช้สำหรับออกจากการเขียนโปรแกรม เมื่อเราสร้างรหัสผ่านเข้าใช้งาน Jupyter Notebook
Quit	ปุ่มปิดการทำงานของ Jupyter Notebook
Upload	นำ File จากแหล่งอื่นเข้ามาใน Jupyter Notebook
New	มีเมนูย่อยให้เลือกใช้งานอีก 3 เมนู สำหรับสร้าง File หรือ Folder และแสดงภาษาโปรแกรมที่สามารถพัฒนาโปรแกรมผ่านเครื่องมือ Jupyter Notebook ได้ จากรุ่น 2.8 แสดงเพียงแค่ภาษา Python เท่านั้น เวอร์ชัน 3 (Python 3) ถ้าติดตั้งห้องภาษาจะปรากฏบนแท็บนี้
Text File	ใช้สร้าง Text File ข้อความ
Folder	สำหรับสร้าง Folder เก็บไฟล์คำสั่งโปรแกรมหรือ Text File ที่ได้สร้างขึ้นมาใช้งาน
Terminal	เปิด Terminal บนหน้าเว็บбраузอร์เพื่อเข้าสู่โหมด Command Line

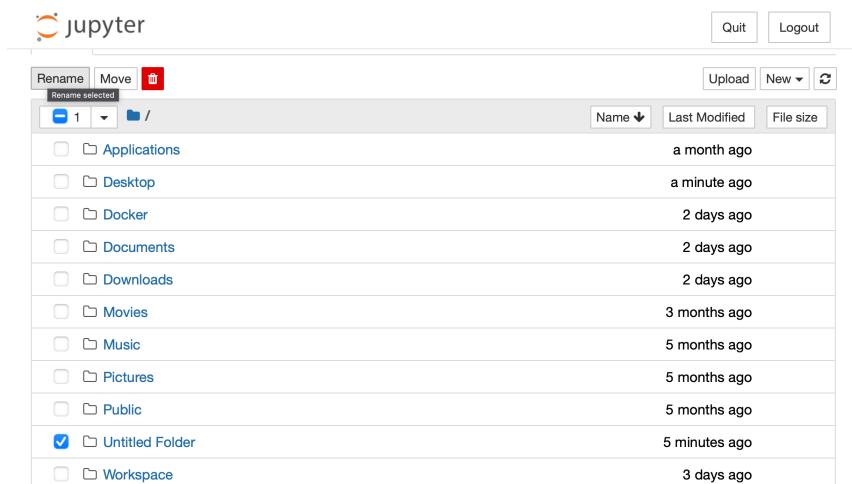
ตาราง 2.1: ส่วนติดต่อผู้ใช้ในหน้า Notebook Dashboard ของ Jupyter Notebook

## 2.4 เริ่มต้นใช้งาน Jupyter Notebook

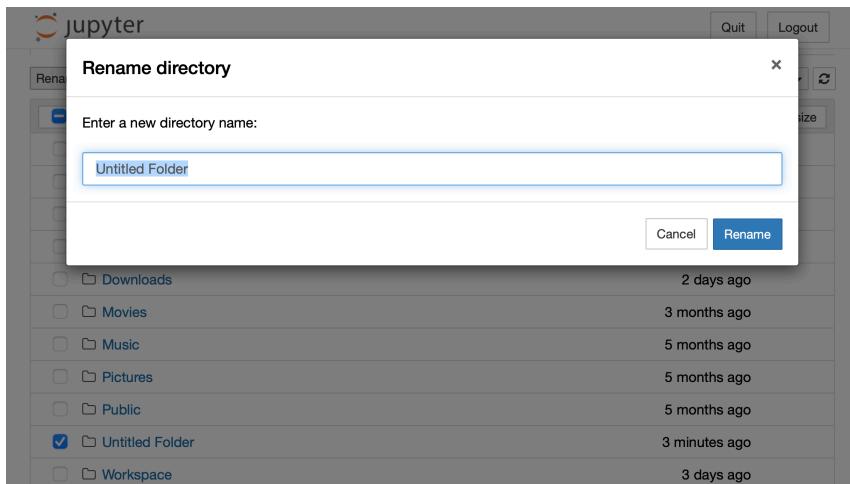
1. จากหน้าต่างหลัก Jupyter Notebook คลิกเมนู New ▶ Folder เพื่อสร้าง Folder สำหรับเก็บ File



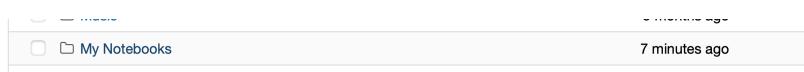
2. ผู้อ่านจะสังเกตเห็น Folder ที่สร้างขึ้นมาใหม่มีชื่อเป็น Untitled Folder จากนั้นให้คลิกที่หน้า Check Box ของไฟล์เดอร์ใหม่นั้น แล้วคลิกที่ปุ่ม Rename เพื่อเปลี่ยนชื่อให้เหมาะสม



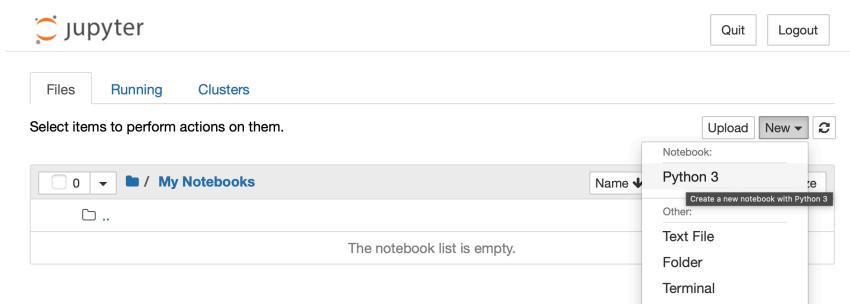
3. จะปรากฏหน้าต่างให้ผู้อ่านเปลี่ยนชื่อ Folder ใหม่ เสร็จแล้วให้คลิกปุ่ม Rename หรือ กดคีย์ Enter



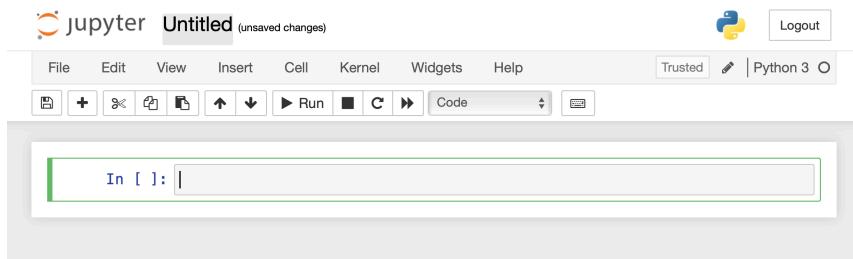
4. จะเห็นว่าชื่อ Folder ได้ถูกเปลี่ยนเป็นชื่อใหม่ตามที่กำหนดเรียบร้อยแล้ว



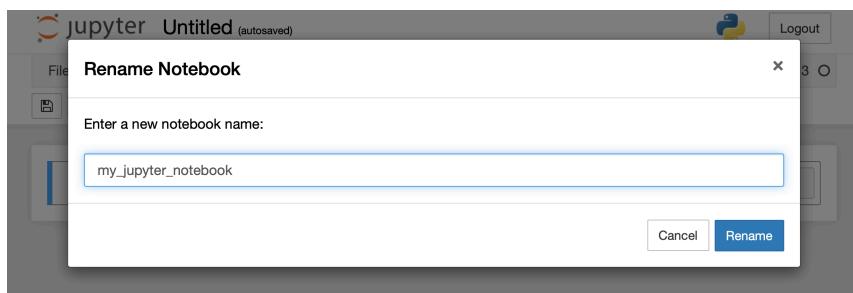
5. จากนั้นให้คลิกเลือกที่ไฟล์เดอร์ และคลิกเมนู New ▶ Python 3



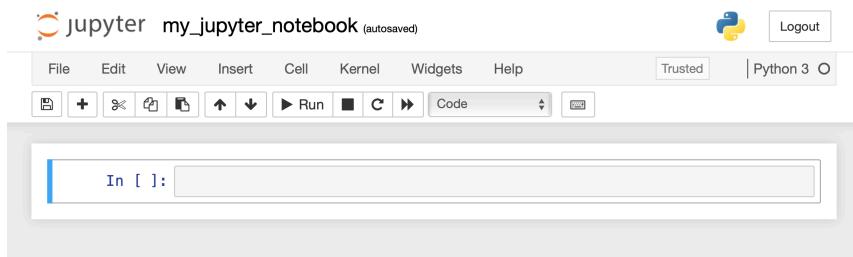
6. เมื่อสร้างไฟล์ใหม่จะมีชื่อเป็น Untitled ผู้อ่านสามารถเปลี่ยนชื่อไฟล์ได้โดยการคลิกที่ชื่อ Untitled



7. เปลี่ยนชื่อใหม่ตามต้องการแล้วคลิกปุ่ม Rename หรือกดคีย์ Enter



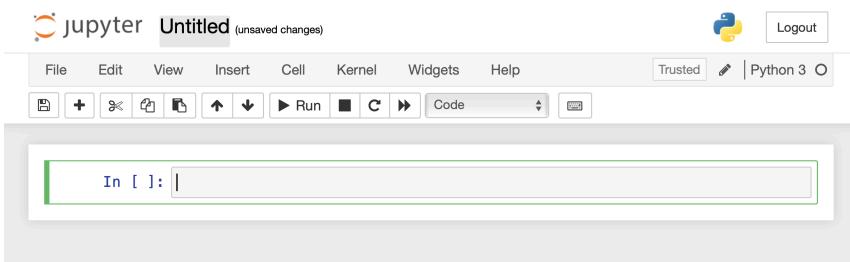
8. หลังจากเปลี่ยนชื่อไฟล์เสร็จ จะได้ชื่อไฟล์ใหม่ดังภาพ



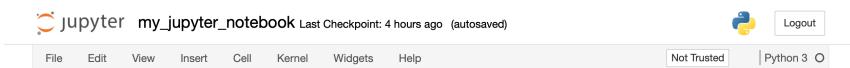
### 2.4.1 แนะนำเมนูต่าง ๆ ก่อนเขียนคำสั่งโปรแกรม

ก่อนการสร้างไฟล์เพื่อเริ่มต้นเขียนโปรแกรมเราจำเป็นต้องสร้าง Folder สำหรับจัดเก็บไฟล์ให้เป็นหมวดหมู่ เพื่อให้ง่ายต่อการค้นหา เมื่อเข้าสู่หน้าต่างการเขียนโปรแกรมซึ่งต่อไปจะเรียกว่า Notebook ซึ่งมีเมนูต่าง ๆ ดังนี้

- ชื่อ Notebook ผู้อ่านสามารถเปลี่ยนชื่อได้โดยการคลิกที่ชื่อ Untitled



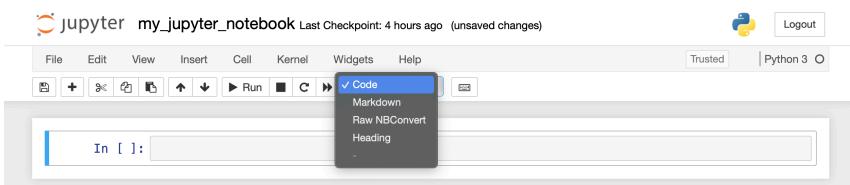
- Menu Bar สังเกตเห็นว่าจะมีเมนูให้ใช้งานเหมือนกับโปรแกรมทั่ว ๆ ไป ที่ผู้อ่านคุ้นเคย กัน เช่น File, Edit, View, Insert และมีเมนู Cell สำหรับสั่งให้ Cell ที่เลือกนั้นทำการ ประมวลผลโปรแกรม เมนู Kernel ใช้สำหรับ Start หรือ Shutdown การทำงานของ Jupyter Notebook แล้วยังมี Menu Widgets และ Menu Help ไว้ขอความช่วย เหตุใด เพื่อขอคำอธิบาย



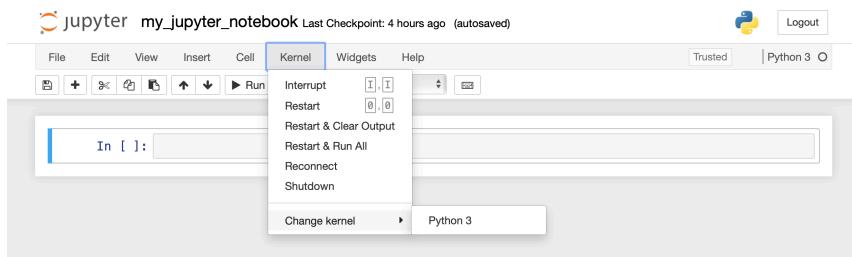
- Tool Bar เป็นเครื่องมือทางลัดเลือกสิ่งงานต่าง ๆ โดยไม่ต้องเข้าไปคลิกเลือกที่ Menu Bar



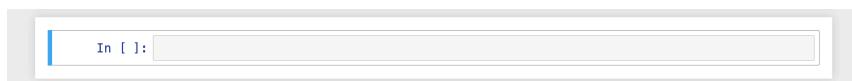
- Cell Type จะแสดงให้ทราบถึงในขณะนี้กำลังใช้ Cell ประเภทใดอยู่



- Kernel หากได้ทำการติดตั้งหลายภาษาสำหรับเขียนโปรแกรม สามารถเปลี่ยน Kernel ของภาษาได้จากเมนูนี้ หรือสั่งให้ Kernel ภาษาไฟร่อนเริ่มต้นทำงานใหม่



6. Logout จะใช้สำหรับกรณีที่ได้กำหนดรหัสผ่านเข้าใช้งาน Jupyter Notebook
7. Cell แต่ละช่องจะเรียกว่า Cell สำหรับการเขียนคำสั่งโปรแกรม



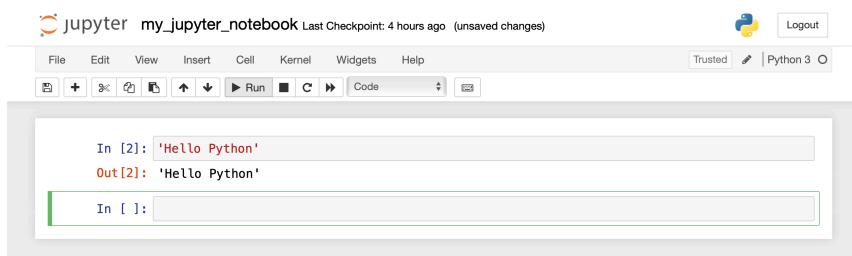
## 2.4.2 การใช้เครื่องมือ Jupyter Notebook เขียนภาษาไพธอน

หลังจากที่ได้รู้จักกับเมนูต่าง ๆ ไปแล้ว ต่อไปจะมาเริ่มต้นเขียนโปรแกรมภาษาไพธอนบน Notebook กันเพื่อสร้างความคุ้นเคย ก่อนที่จะลงมือปฏิบัติการเขียนโปรแกรมภาษาไพธอนอย่างจริงจัง

1. ให้พิมพ์ข้อความ

'Hello Python'

โดยข้อความทั้งหมดจะต้องอยู่ในเครื่องหมาย Single Quote (' . . . ') ลงในช่อง Cell จากนั้นให้คลิกปุ่ม Run บน Toolbar หรือคลิกเมนู Cell > Run Cells หรือกดคีย์ลัด ↑ + ↵ แล้ว Notebook จะแสดงผลลัพธ์พร้อมทั้งปรากฏ cell ใหม่ขึ้นมาใหม่อัตโนมัติ



2. เพื่อเบรียบเทียบการแสดงผลให้พิมพ์

```
print('Hello Python')
```

ลงในช่อง Cell จำนวน แล้วคลิกปุ่ม Run บน Toolbar หรือคลิกเมนู Cell > Run Cells หรือกดคีย์ลัด  $\leftarrow + \rightarrow$  จะได้ผลลัพธ์ สังเกตเห็นว่าผลลัพธ์ที่แสดงออกมาจะไม่มีเครื่องหมาย Single quote (' . . . ') ครอบข้อความ

```
In [2]: print('Hello Python')
Hello Python
```

3. ผู้อ่านสามารถแก้ไขข้อความใน Cell ที่ได้สั่งให้ทำงานไปแล้วได้ โดยให้คลิกเลือก Cell ที่ต้องการแก้ไขข้อความ จำนวนสั่ง Run ใหม่อีกครั้งจะได้ผลลัพธ์ และจะสังเกตเห็นว่า ช่อง Cell จะมีการเพิ่มจำนวนครั้งที่สั่งให้ประมวลผลคำสั่งโปรแกรม

```
In [3]: 'Hello World'
Out[3]: 'Hello World'

In [2]: print('Hello Python')
Hello Python

In [ ]:
```

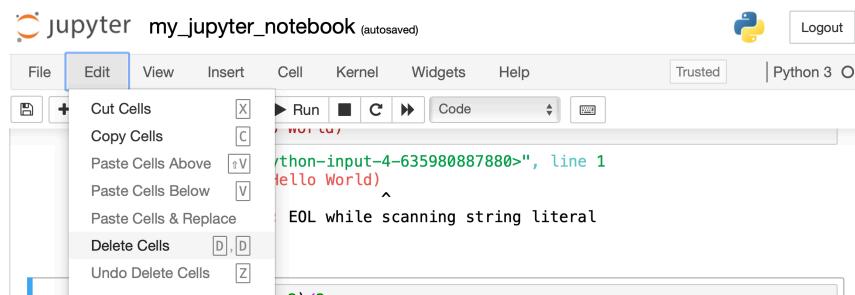
4. เมื่อเขียนคำสั่งผิด Jupyter Notebook จะแสดงการแจ้งเตือนข้อผิดพลาด และแสดงตำแหน่งคำสั่งที่เขียนผิด และมีการแจ้งเตือนข้อผิดพลาด (Error) รวมทั้งสาเหตุที่ทำให้เกิด Error

```
In [4]: print('Hello World')
File "<ipython-input-4-635980887880>", line 1
      print('Hello World')
^
SyntaxError: EOL while scanning string literal
```

5. ในเครื่องมือ Jupyter Notebook แต่ละช่องของ Cell สามารถทำการคำนวณพื้นฐาน เช่น การบวก การลบ การคูณ หรือหาร ได้โดย

```
In [7]: 8 + 4 * (8 - 2)/2
Out[7]: 20.0
```

6. หากต้องการลบ Cell ที่ไม่ต้องการออก ให้คลิกที่ Cell จากนั้นคลิกเมนู **Edit** > **Delete Cells** หรือกดคีย์ลัด **D**, **D** แล้ว Cell ที่ถูกเลือกจะถูกลบออกจาก Notebook



7. หากต้องการเพิ่ม Cell ซึ่งสามารถเพิ่ม Cell ทั้งด้านบนหรือด้านล่าง โดยให้เลือก Cell เดิมก่อน จากนั้นคลิกเมนู

- **Insert** > **Insert Cell Above** (คีย์ลัด **A**) เพื่อเพิ่ม Cell ด้านบน
- **Insert** > **Insert Cell Below** (คีย์ลัด **B**) เพื่อเพิ่ม Cell ด้านล่าง



### 2.4.3 การขอความช่วยเหลือ

Jupyter Notebook ได้อีกหนทางประยุกต์ใช้งานได้ขอความช่วยเหลือที่เกี่ยวกับการพัฒนาโปรแกรม ผ่านฟังก์ชัน `help()`

### 1. ให้ผู้ใช้งานพิมพ์

`help()`

ที่ Cell แล้วสั่ง **Run** จะแสดงคำอธิบายให้เราทราบว่า สามารถขอความช่วยเหลืออะไรได้บ้าง หัวข้อที่สามารถให้ความช่วยเหลือได้ คือ modules, keywords และ topics ให้ลองเรียกขอความช่วยเหลือการใช้งาน modules โดยพิมพ์ลงในช่อง Textbox แล้วสั่ง **Run** อีกครั้ง

```
In [*]: help()

Welcome to Python 3.9's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.9/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules. To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics". Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> modules
```

### 2. จากผลลัพธ์จะเห็นได้ว่ามี modules ให้เรียกใช้งานจำนวนมาก จากนั้นให้ลองเรียกดูวิธีการใช้งาน module os แล้วสั่ง **Run** ดูผลลัพธ์

```
_weakrefset      idna          pyparsing        xdrlib
__xxsubinterpreters  imaplib      pyrsistent      xml
__xtestfuzz       imghdr       pytz            xmrpc
__zoneinfo        imp          qtconsole       xxlimited
abc              importlib    qtpy             xssubtype
aifc              inspect      queue           zipapp
antigravity      io           quopri          zipfile
anyio             ipaddress    random          zipimport
appnope           ipykernel   re               zlib
argon2            ipykernel_launcher readline      zmq
argparse          ipython_genutils reprlib         zoneinfo

Enter any module name to get more help. Or, type "modules spam" to search
for modules whose name or summary contain the string "spam".

help> os
```

### 3. ผลลัพธ์จะแสดงรายละเอียดต่าง ๆ เช่น มีฟังก์ชันอะไรบ้าง วิธีการใช้งาน เป็นต้น

```

help> os
Help on module os:

NAME
    os - OS routines for NT or Posix depending on what system we're on.

MODULE REFERENCE
    https://docs.python.org/3.9/library/os

The following documentation is automatically generated from the Python
source files. It may be incomplete, incorrect or include features that
are considered implementation detail and may vary between Python
implementations. When in doubt, consult the module reference at the
location listed above.

DESCRIPTION
    This exports:
        - all functions from posix or nt, e.g. unlink, stat, etc.

```

4. หากเราต้องการชื่อฟังก์ชันแต่ไม่รู้วิธีการเขียนคำสั่งเรียกใช้งาน เราสามารถขอความช่วยเหลือด้วยวิธีพิมพ์

`help(ชื่อฟังก์ชัน)`

แล้วสั่ง **Run** ก็จะแสดงรายละเอียดการใช้งานของฟังก์ชันนั้น ๆ

```

In [5]: help(print)
Help on built-in function print in module builtins:

print(*args, **kwargs)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
    file: a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

```

#### 2.4.4 การ Download และ Upload ไฟล์

การ Download เป็นการนำเอาไฟล์โปรแกรมที่ได้พัฒนาบนเครื่องมือ Jupyter Notebook ไปพัฒนาต่อหรือนำไปประมวลผลบนเครื่องคอมพิวเตอร์อื่น ซึ่งอาจจะเป็นระบบปฏิบัติการเดียวกันหรือต่างระบบปฏิบัติการก็ได้ สำหรับการ Upload นั้น เป็นการนำเอาไฟล์โปรแกรมที่พัฒนาจากเครื่องมือ Jupyter Notebook บนเครื่องอื่น เข้าสู่เครื่องมือ Jupyter Notebook เพื่อพัฒนาโปรแกรมต่อหรือประมวลผลบนเครื่องของเราทั้งสองวิธี สามารถทำได้ตามขั้นตอนดังต่อไปนี้

- เมื่อพัฒนาโปรแกรมด้วยเครื่องมือ Jupyter Notebook จะมีนามสกุลไฟล์เป็น **.ipynb** และหากต้องการนำเอาไฟล์โปรแกรมไปประมวลกับเครื่องอื่นด้วยทั้งแบบ

ภาษา Python ต้องสร้างไฟล์ให้เป็นไฟลอนสคริปต์ .py ก่อน ถึงจะนำไปใช้กับเครื่องอื่นได้ ให้เลือกที่เมนู **File > Download as > Python (.py)** และบันทึกไว้ในตำแหน่งที่ต้องการ แต่หากต้องการนำเอาไฟล์โปรแกรมไปประมวลกับเครื่องอื่นด้วย Jupyter Notebook เมื่อก่อนกัน ให้เลือกที่เมนู **File > Download as > Notebook (.ipynb)**

2. เลือกตำแหน่งไฟล์ที่ต้องการจัดเก็บ และเปลี่ยนชื่อไฟล์หากต้องการ จากนั้นคลิกปุ่ม **Save**
3. สามารถแสดงผลลัพธ์โดยใช้คำสั่งผ่าน Command line จากไฟล์ที่มีนามสกุล .py ด้วยคำสั่ง

```
python script_name.py
```

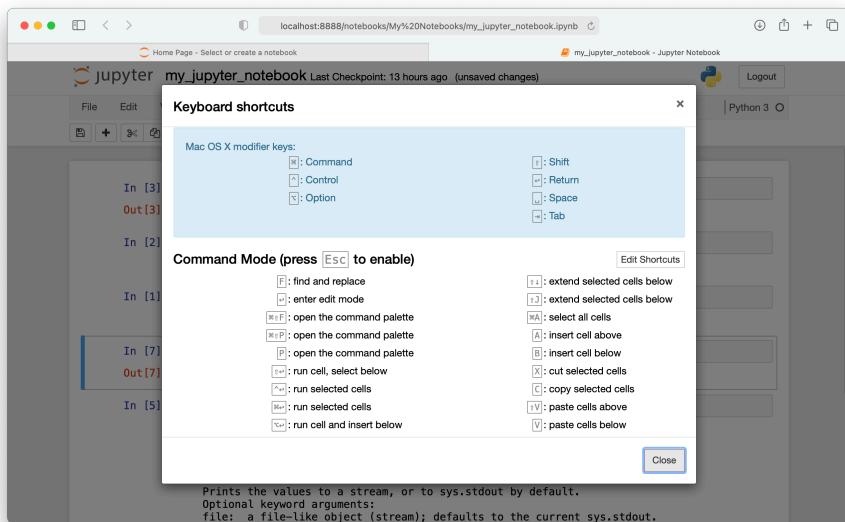
4. เมื่อต้องการนำไฟล์จากเครื่องอื่นมาพัฒนาโปรแกรม ให้คลิกเมนู **Upload** เลือกไฟล์ที่ต้องการแล้วคลิกปุ่ม **Open**



5. เลือกไฟล์ที่ต้องการซึ่งจะต้องเป็นไฟล์ .ipynb เท่านั้น จากนั้นให้คลิกปุ่ม **Upload** เพื่อนำไฟล์เข้าสู่ Jupyter Notebook

## 2.4.5 ประมวลผลคำสั่งผ่านคีย์ลัด

หากต้องการสั่งให้โปรแกรมทำงานโดยไม่ใช้งานเมนู สามารถเรียกใช้งานคีย์ลัด (Shortcut key) ได้ ซึ่งโปรแกรม Jupyter Notebook ก็ได้อันวยความสะดวกในส่วนนี้ไว้ให้ เช่นกัน ผู้อ่านสามารถตรวจสอบคีย์ลัดได้จากเมนู **Help > Keyboard Shortcuts**



รูป 2.9: ประมวลผลคำสั่งผ่านคีย์ลัดของ Jupyter Notebook

## 2.5 การแสดงผลและการรับข้อมูล

พื้นฐานอย่างหนึ่งที่ผู้เริ่มพัฒนาโปรแกรมจะต้องรู้คือ การแสดงข้อมูลออกทางจอภาพ และการรับข้อมูลผ่านทางคีย์บอร์ดเพื่อนำข้อมูลที่ป้อนเข้ามาไปประมวลผลต่อ ในส่วนนี้ผู้อ่านจะได้รู้จักกับฟังก์ชันที่ทำหน้าที่ทั้งสอง

### 2.5.1 การแสดงผลออกทางจอภาพ

ฟังก์ชัน `print()` ทำหน้าที่แสดงผลข้อมูลออกทางหน้าจอ ซึ่งเป็นฟังก์ชัน Built-in ที่ภาษา Python ได้จัดเตรียมไว้ให้ผู้ใช้งานเรียกใช้งานได้เลย มีรูปแบบการใช้งานได้ดังนี้

```
print(object, sep=' ', end='\n',
      file=sys.stdout, flush=False)
```

`object` ขอบเจ็คที่ต้องการแสดงผล มีได้มากกว่า 1 ขอบเจ็ค

`sep` ส่วนที่ใช้แยกแต่ละขอบเจ็คออกจากกัน ค่าปกติเป็นช่องว่าง

`end` การกำหนดแสดงผลขอบเจ็ค ค่าปกติเป็นการขึ้นบรรทัดใหม่

`file` ขอบเจ็คไฟล์ที่ต้องการแสดงผล

`flush` เมื่อดำเนินมุลจากบัฟเฟอร์บันทึกลงไฟล์ ค่าปกติเป็น False

ตัวอย่างต่อไปนี้เป็นการใช้งานฟังก์ชัน `print()` ในการแสดงผลออกทางจอภาพ ผู้อ่านสามารถดูแนวทางการใช้งานโดยสังเขปได้ดังตาราง 2.2

### ตัวอย่าง 2.1

การใช้งานฟังก์ชัน `print()`

```
1 x = 5
2 y = 10
3 s = 'Python is an easy programming language.'
4 print('x = ', x, ',', 'y = ', y)
5 print('ผลลัพธ์ของ x * y = ', x * y)
6 print(s)
```

x = 5, y = 10  
ผลลัพธ์ของ x \* y = 50  
Python is an easy programming language



### ตัวอย่าง 2.2

การเรียกใช้งานฟังก์ชัน `print()` และแสดงผลในรูปแบบต่าง ๆ

```
1 print('Hello Python', 'Hello Python programming')
2 print(50)
3 print('Hello' + 'Python')
4 print('Hello' + ' ' + 'Python')
5 print()
6 print(5*10/2)
```

Hello Python Hello Python programming  
50  
HelloPython  
Hello Python

25.0



คำสั่ง	คำอธิบาย
<code>print('...')</code> หรือ <code>print(...)</code>	ให้แสดงผลชนิดข้อมูลอักขระหรือสตริง
<code>print(50)</code>	ให้แสดงผลชนิดข้อมูลจำนวนเต็ม
<code>print(var1, var2, ..., var_n)</code>	ให้แสดงผลค่าข้อมูลที่เก็บอยู่ในตัวแปร
<code>print('...' + '...')</code>	ต้องการให้แสดงผลการเชื่อมข้อมูลชนิดตัวอักษรหรือสตริงต่อกัน
<code>print('...', '...')</code>	ต้องการให้แสดงผลการเชื่อมข้อมูลชนิดตัวอักษรหรือสตริงต่อกันและเว้นวรรค
<code>print('...' + ' ' + '...')</code>	ต้องการให้แสดงผลการเชื่อมข้อมูลชนิดตัวอักษรหรือสตริงต่อกันและเว้นวรรค
<code>print('...', var)</code>	ต้องการให้แสดงผลข้อความและค่าข้อมูลที่เก็บอยู่ในตัวแปร
<code>print()</code>	ต้องการให้เว้นบรรทัดใหม่
<code>print(5+10/2)</code>	ต้องการแสดงผลลัพธ์จากการดำเนินการทางคณิตศาสตร์

ตาราง 2.2: การเขียนคำสั่งแสดงข้อมูลด้วยฟังก์ชัน `print()`

## 2.6 การรับข้อมูล

ฟังก์ชัน `input()` ทำหน้าที่รับข้อมูลที่จะถูกป้อนเข้ามาผ่านทางคีย์บอร์ดจากผู้ใช้งาน ไม่ว่าจะเป็นตัวเลขจำนวนเต็ม ตัวเลขจำนวนทศนิยม หรือข้อความ (String) ซึ่งเป็นประเภทฟังก์ชัน Built-in อีกหนึ่งตัวที่เรียกใช้งานได้เลย แต่เมื่อป้อนข้อมูลไม่ว่าจะเป็นตัวเลขหรือตัวอักษร ภาษาไพธอนจะแปลงเป็นชนิดข้อมูลสตริงทั้งหมด (สำหรับการแปลงข้อมูลผู้อ่านจะได้เรียนรู้ในบทต่อไป) การเรียกใช้งานฟังก์ชัน `input()` แสดงดังต่อไปนี้

### ตัวอย่าง 2.3 การใช้งานฟังก์ชัน `input()`

```

1 first_name = input('กรุณาป้อนชื่อของคุณ : ')
2 last_name = input('กรุณาป้อนนามสกุลของคุณ : ')
3 age = input('กรุณาป้อนอายุของคุณ : ')
4 print('-----')
5 print('ชื่อของคุณ คือ ', first_name)
6 print('นามสกุลของคุณ คือ ', last_name)
7 print('อายุของคุณ คือ ', age)
```

กรุณาป้อนชื่อของคุณ : ประหยัด  
 กรุณาป้อนนามสกุลของคุณ : จันโอซิท  
 กรุณาป้อนอายุของคุณ : 61

-----  
 ชื่อของคุณ คือ ประหยัด  
 นามสกุลของคุณ คือ จันโอซิท  
 อายุของคุณ คือ 61



## 2.7 การเขียนคำอธิบายโปรแกรม

เมื่อพัฒนาโปรแกรมไปได้สักระยะหนึ่งแล้วโปรแกรมมีขนาดใหญ่ขึ้น สิ่งที่ตามมาคือไม่ทราบว่าโปรแกรมทำงานอย่างไรบ้าง มีการเรียกใช้งานและส่งค่าข้อมูลระหว่างกันอย่างไร ดังนั้นเพื่อให้คนอื่นสามารถทำความเข้าใจการทำงานของโปรแกรมได้ เราจึงต้องเขียนคำอธิบายการทำงานในแหล่งที่มา

หรือในแต่ละบรรทัดภายในโปรแกรม (Comment) เพราะจะช่วยให้แก้ไขเมื่อเกิดข้อผิดพลาด ขึ้นได้ง่ายและรวดเร็ว หากพัฒนาโปรแกรมร่วมกันหลายคนยิ่งมีประโยชน์ต่อผู้อื่นเป็นอย่างมาก การเขียนคำอธิบายทำได้ 2 วิธีด้วยกันคือ

1. การเขียนคำอธิบายเพียงบรรทัดเดียวให้ใช้เครื่องหมาย `#`
2. การเขียนคำอธิบายแบบหลายบรรทัดให้ใช้เครื่องหมาย triple quotes

`('''...''')` หรือ `("""...""")`

#### ตัวอย่าง 2.4

การเขียนคำอธิบายคำสั่งโปรแกรม (Comment) โดยใช้เครื่องหมาย `#`

```

1 print('Hello Python', 'Hello Python programming')
2 print(50) # แสดงตัวเลข 50 ออกทางหน้าจอ
3 print(257.451 + 124.12) # แสดงผลลัพธ์จากการบวก

```

Hello Python Hello Python programming  
50  
381.571



#### ตัวอย่าง 2.5

การเขียนคำอธิบายคำสั่งโปรแกรม (Comment) แบบหลายบรรทัด

```

1 ...
2 1. แสดงการใช้ฟังก์ชัน input() รับข้อมูลผ่านทางคีย์บอร์ด
3 2. แสดงการใช้ฟังก์ชัน print() แสดงผลข้อมูล\n',
4 ...
5 msg = input('กรุณาป้อนข้อความ : ')
6 print('ข้อความที่คุณป้อน = ', msg)

```

กรุณาป้อนข้อความ : Python Programming  
ข้อความที่คุณป้อน = Python Programming



จากตัวอย่างการเขียนตัวอย่าง 2.4 และตัวอย่าง 2.5 ผู้อ่านจะสังเกตเห็นว่า ข้อความที่อยู่ด้านหลังเครื่องหมาย # หรือที่อยู่ภายใต้เครื่องหมาย triple-quotes ('' '' ... '') ซึ่งเป็นข้อความคำอธิบายการทำงานของคำสั่งโปรแกรม จะไม่ถูกนำมาแสดงผลเมื่อเราสั่งให้โปรแกรมประมวลผลไฟรอนสคริปต์

## สรุปก่อนจบบท

ในบทนี้เราได้เรียนรู้ถึงการติดตั้งตัวแปรภาษาไพธอน การใช้งานไฟรอนสคริปต์ การใช้งานเครื่องมือ Jupyter Notebook และยังได้รู้จักกับฟังก์ชัน `print()` และ `input()` ที่นำมาแสดงผลข้อมูลและรับข้อมูล ซึ่งเป็นฟังก์ชัน Built-in รวมไปถึงการเขียนคำอธิบายคำสั่งโปรแกรม

## แบบฝึกหัด

1. จงยกตัวอย่างเครื่องมือที่สามารถนำมาระบุน้ำใจภาษาไพธอนได้อย่างน้อย 3 เครื่องมือ
2. จงเขียนโปรแกรมให้แสดงผลข้อมูล: ชื่อ, ที่อยู่, ประวัติการศึกษา, อาชีพ, สิ่งที่ชอบ และสิ่งที่ไม่ชอบ โดยให้รับข้อมูลเหล่านี้ผ่านทางคีย์บอร์ด
3. จงเขียนโปรแกรมสั้น ๆ (อะไรมีได้) พร้อมทั้งเขียนคำอธิบายการทำงานของโปรแกรมนั้น (comment) ทั้งแบบบรรทัดเดียวและหลายบรรทัด