

# CSE 4020/5260

## Database Systems

Instructor: Fitzroy Nembhard, Ph.D.

## Installing and Setting up Kernels for Python, Java and C in Jupyter



Java



CONDA



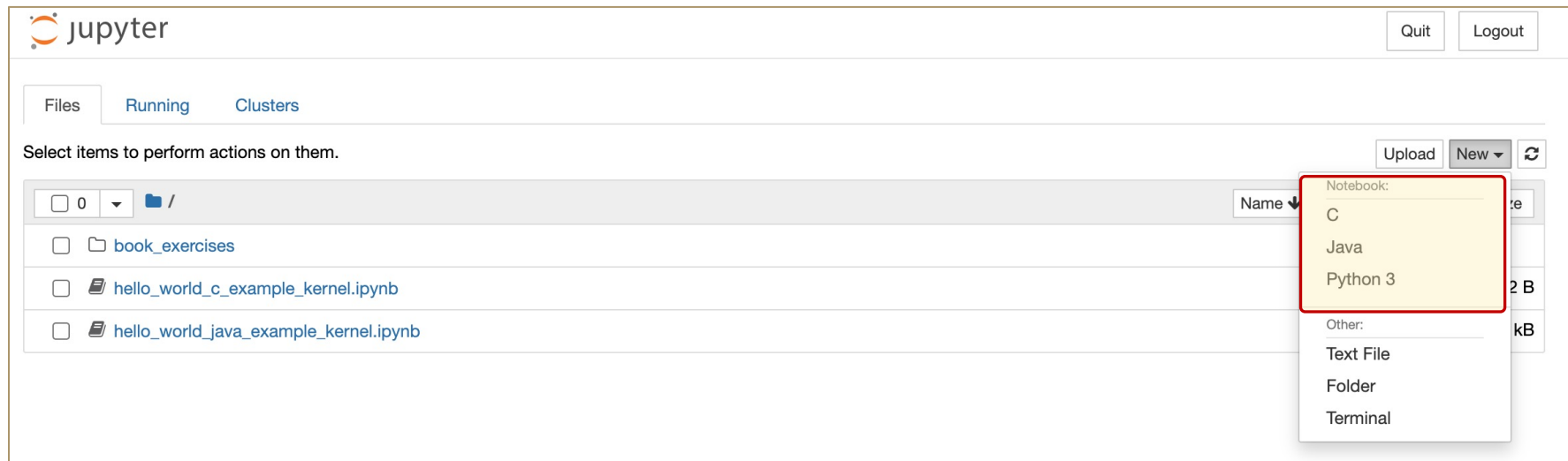
# Jupyter Notebooks

---

- **We will be using Jupyter Notebooks to interact with Databases throughout the course**
  - First, install Miniconda (lightweight) or Anaconda (consumes more space) on your personal computer.
  - Miniconda will install Python on your computer (<https://docs.conda.io/en/latest/miniconda.html>)
- You may choose a kernel based on your preferred programming language. I will provide setup tips for Python, Java and C (preferably Mac/Linux)

# Jupyter Notebooks

- After Setting up your kernel, you should see your language in the list of kernels on the computer as shown below.



# Contents

---

Installing and launching Jupyter

Executing a Python program in Jupyter

Executing a Java program in Jupyter

Executing a C program in Jupyter

# Using Anaconda to Launch Jupyter Notebook

---

- The following slides will show you how to use Anaconda/Miniconda to launch and use Jupyter Notebooks.



# Using Anaconda to Launch Jupyter Notebook

Open **Anaconda Navigator** from your Applications Menu.

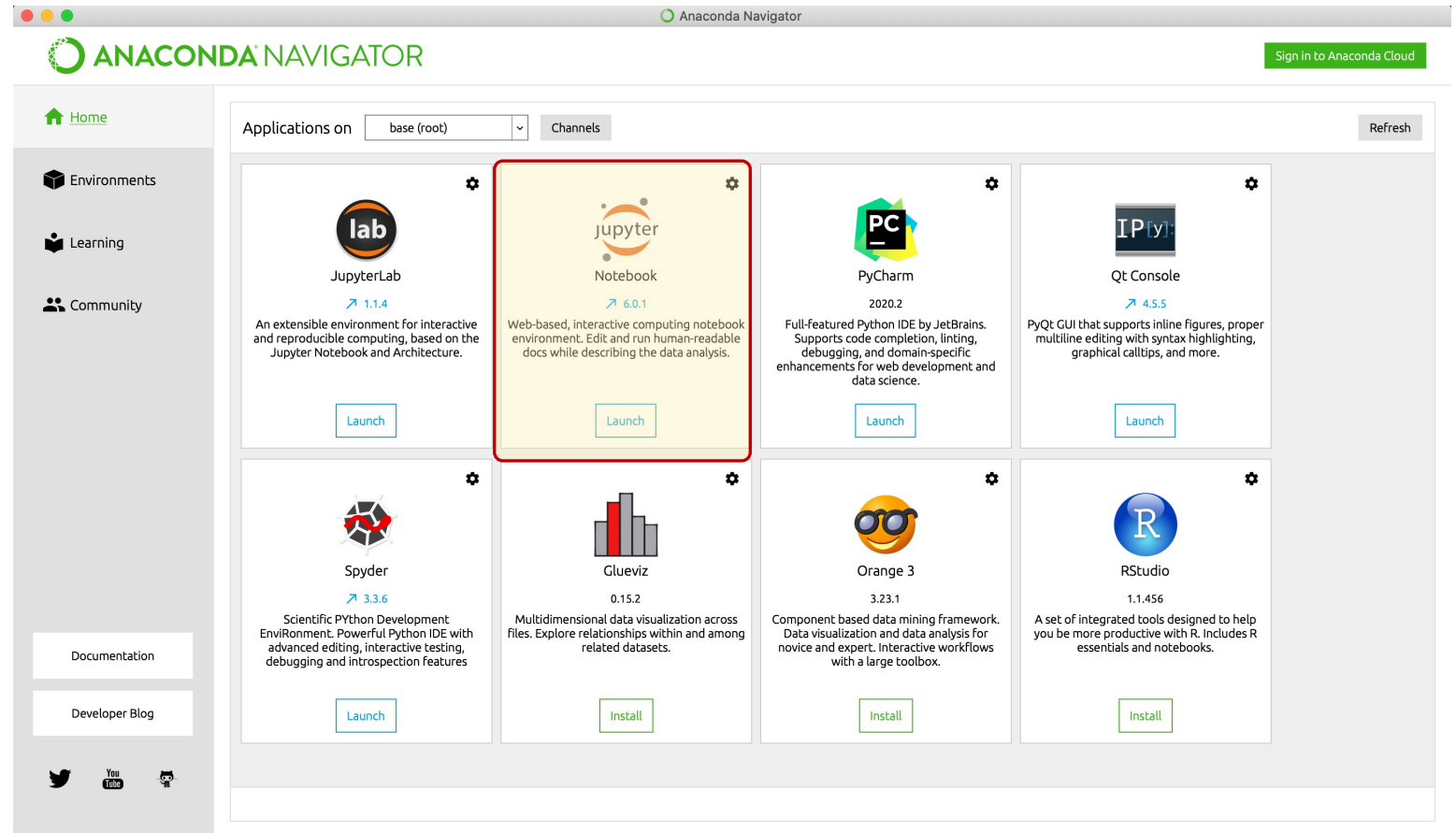
**Note:** The Navigator is only provided in the full version of Anaconda.

If you installed Miniconda, follow the instructions on Slide 10.



# Using Anaconda to Launch Jupyter Notebook


Open **Jupyter Notebook** from the list of Applications



# Using Anaconda to Launch Jupyter Notebook

Launching Jupyter using Anaconda will present a screen that looks like the following.

The next step is to create a folder for this class, preferably in the Documents Folder

 jupyter

Quit

Logout

FilesRunningClusters

Select items to perform actions on them.

Upload

New

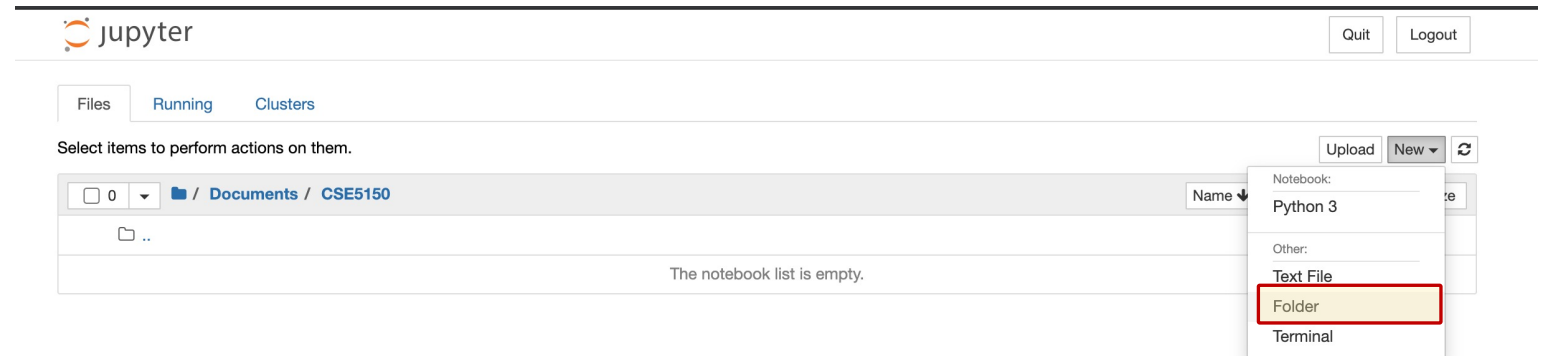
<input type="checkbox"/> 0			Name	Last Modified	File size
<input type="checkbox"/>	Applications			2 months ago	
<input type="checkbox"/>	Desktop			7 days ago	
<input type="checkbox"/>	Documents			2 months ago	
<input type="checkbox"/>	Downloads			2 hours ago	
<input type="checkbox"/>	Movies			2 months ago	
<input type="checkbox"/>	Music			5 months ago	
<input type="checkbox"/>	OneDrive - Florida Institute of Technology			18 days ago	
<input type="checkbox"/>	opt			6 months ago	
<input type="checkbox"/>	oradiag_fitroi			3 months ago	
<input type="checkbox"/>	oradiag_root			4 months ago	
<input type="checkbox"/>	Pictures			4 months ago	
<input type="checkbox"/>	Public			7 months ago	
<input type="checkbox"/>	seaborn-data			4 months ago	
<input type="checkbox"/>	New document 2.2020_08_09_18_00_53.1.svg			6 days ago	2.8 MB
<input type="checkbox"/>	telebit			3 months ago	47 B



# Creating a Class Folder

Click the dropdown labeled “New” to create a new folder.

Name Your folder based on your class name.  
Example: **CSE4020**



# Using a Terminal to Launch Jupyter Notebook

The command to launch Jupyter via a command line interface is:

```
jupyter notebook
```

After running the command, your browser should automatically open to the Jupyter environment. If not, copy the link provided and paste it in your browser.

If Python was not installed or you have issues installing it, follow the link here:

<https://tinyurl.com/y6mkdcqd>

If you cannot get Anaconda/Miniconda to install correctly, please follow the instructions here:

<https://tinyurl.com/grh32cn>

```
[fitzroi@fitzroys-mbp-15 CSE5150 % jupyter notebook
[I 13:36:18.656 NotebookApp] Serving notebooks from local directory
: /Users/fitzroi/Documents/CSE5150
[I 13:36:18.656 NotebookApp] The Jupyter Notebook is running at:
[I 13:36:18.656 NotebookApp] http://localhost:8888/?token=595e4d649
17976bc224f04f3476e15513f5d3e25935b1803
[I 13:36:18.656 NotebookApp] or http://127.0.0.1:8888/?token=595e4
d64917976bc224f04f3476e15513f5d3e25935b1803
[I 13:36:18.656 NotebookApp] Use Control-C to stop this server and
shut down all kernels (twice to skip confirmation).
[C 13:36:18.666 NotebookApp]
```

To access the notebook, open this file in a browser:

file:///Users/fitzroi/Library/Jupyter/runtime/nbserver-35806-open.html

Or copy and paste one of these URLs:

http://localhost:8888/?token=595e4d64917976bc224f04f3476e15513f5d3e25935b1803

or http://127.0.0.1:8888/?token=595e4d64917976bc224f04f3476e15513f5d3e25935b1803

# Executing Python Code from a Jupyter Notebook

---

- The following slides will show you how to write and execute Python code in a Jupyter Notebook to run a Reverse-String program.

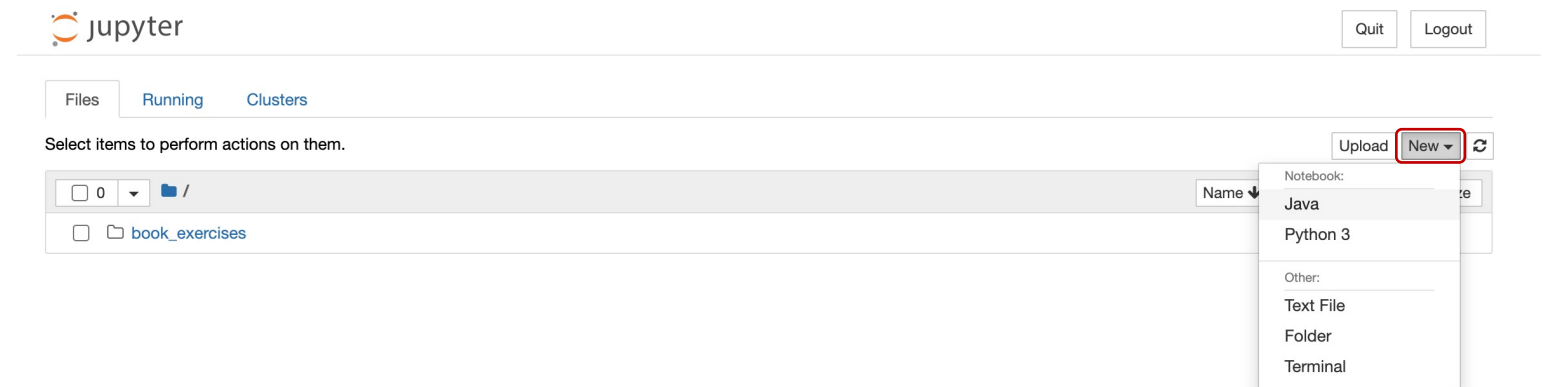


# Launching a Python Kernel in Jupyter

Launch jupyter and ensure that Python 3 is included in the list of kernels by clicking the “New” dropdown menu.

You may also launch Jupyter from the command line

```
jupyter notebook
```

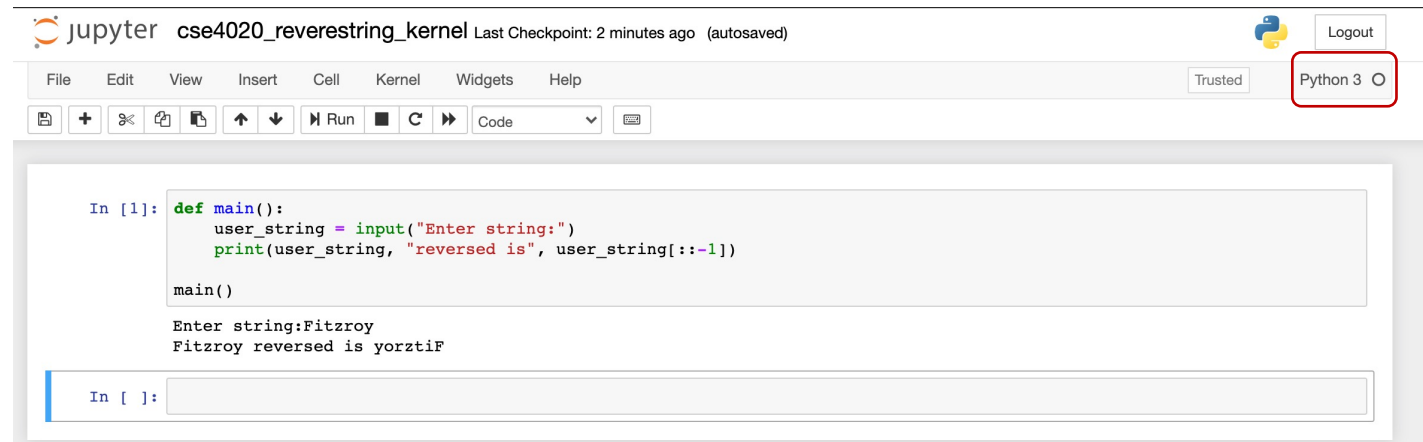


# Executing Python Code in Jupyter

Enter Python code in your Notebook as shown in the screenshot on the right.

This example demonstrates how to read a string from the standard input, reverse the string and display the results.

Notice the highlighted kernel named Python 3.



# Executing Java Code Within a Jupyter Notebook

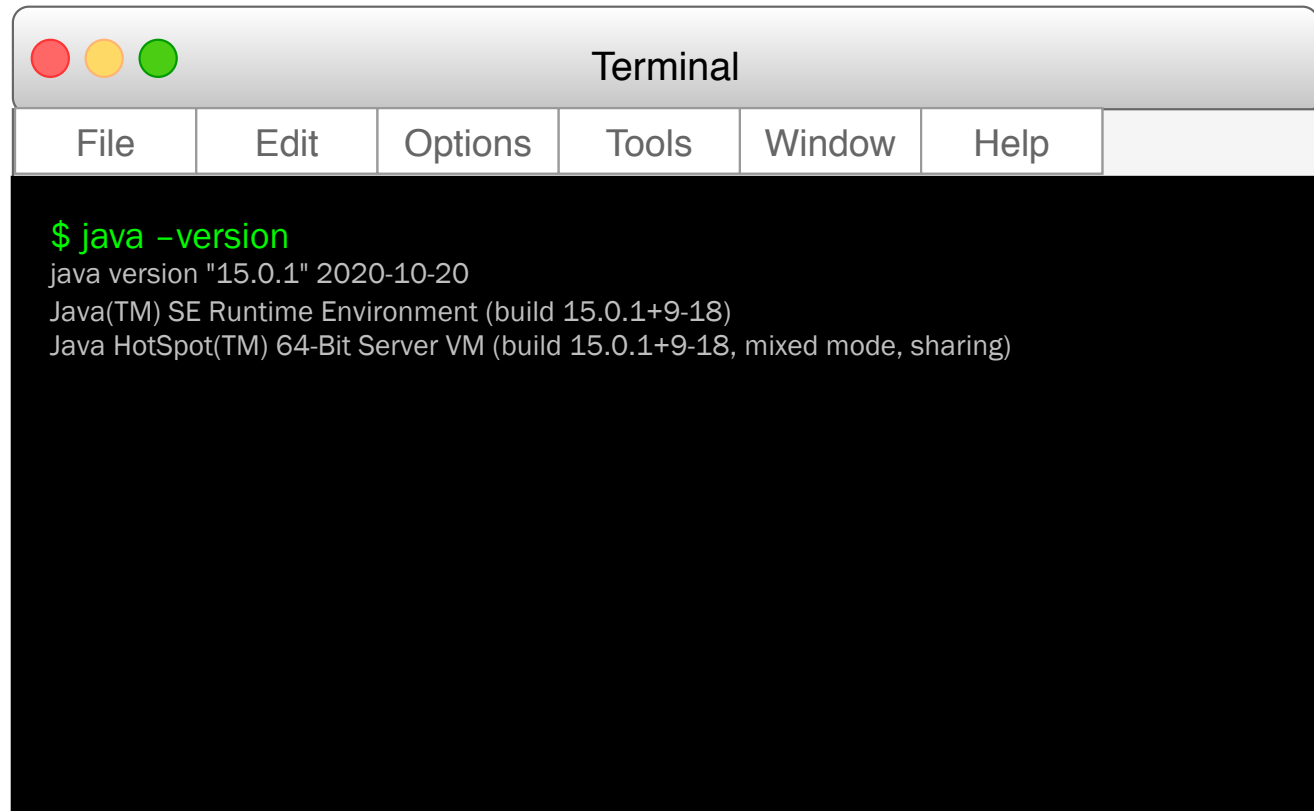
---

- The following slides will show you how to use Java via Jupyter Notebook to run a Reverse-String program.



# Setting Up IJava for Jupyter

Make sure Java is in your path by executing the following version check command from your command line

A screenshot of a macOS Terminal window. The title bar is light gray with three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Tools", "Window", "Help", and a small empty menu item on the right. The main area of the terminal is black with white text. The first line shows a green prompt character "\$" followed by the command "java -version" in green. The subsequent lines show the output of the command in white: "java version \"15.0.1\" 2020-10-20", "Java(TM) SE Runtime Environment (build 15.0.1+9-18)", and "Java HotSpot(TM) 64-Bit Server VM (build 15.0.1+9-18, mixed mode, sharing)".

```
$ java -version
java version "15.0.1" 2020-10-20
Java(TM) SE Runtime Environment (build 15.0.1+9-18)
Java HotSpot(TM) 64-Bit Server VM (build 15.0.1+9-18, mixed mode, sharing)
```

# Setting Up IJava for Jupyter

Ensure that java is in a location where the jdk was installed and not just the jre. Use the `java --list-modules` command to do this. The list should contain `jdk.jshell`

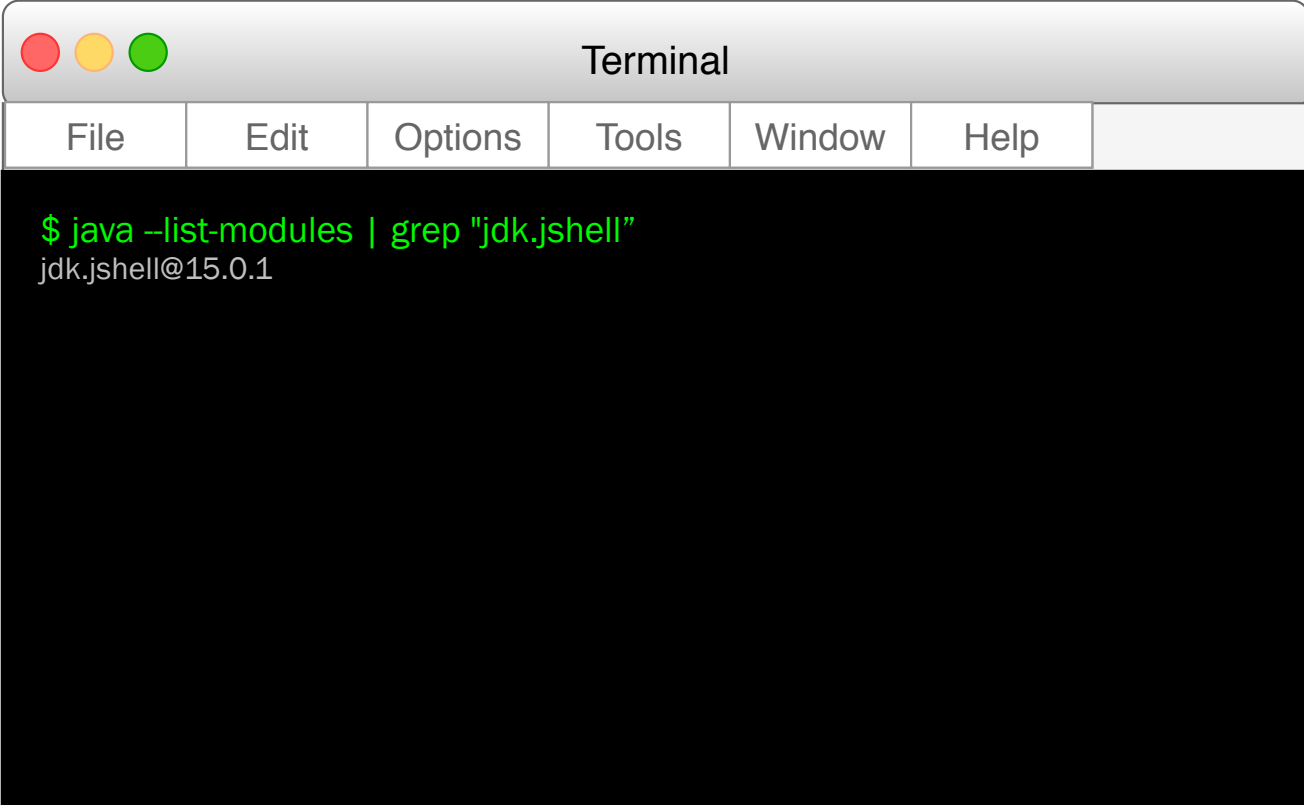
On \*nix

```
java --list-modules | grep "jdk.jshell"
```

On windows

```
java --list-modules | findstr "jdk.jshell"
```

If the kernel cannot start with an `ShellException` error, double check that java is referring to the command for the jdk and not the jre.



```
Terminal
File Edit Options Tools Window Help
$ java --list-modules | grep "jdk.jshell"
jdk.jshell@15.0.1
```



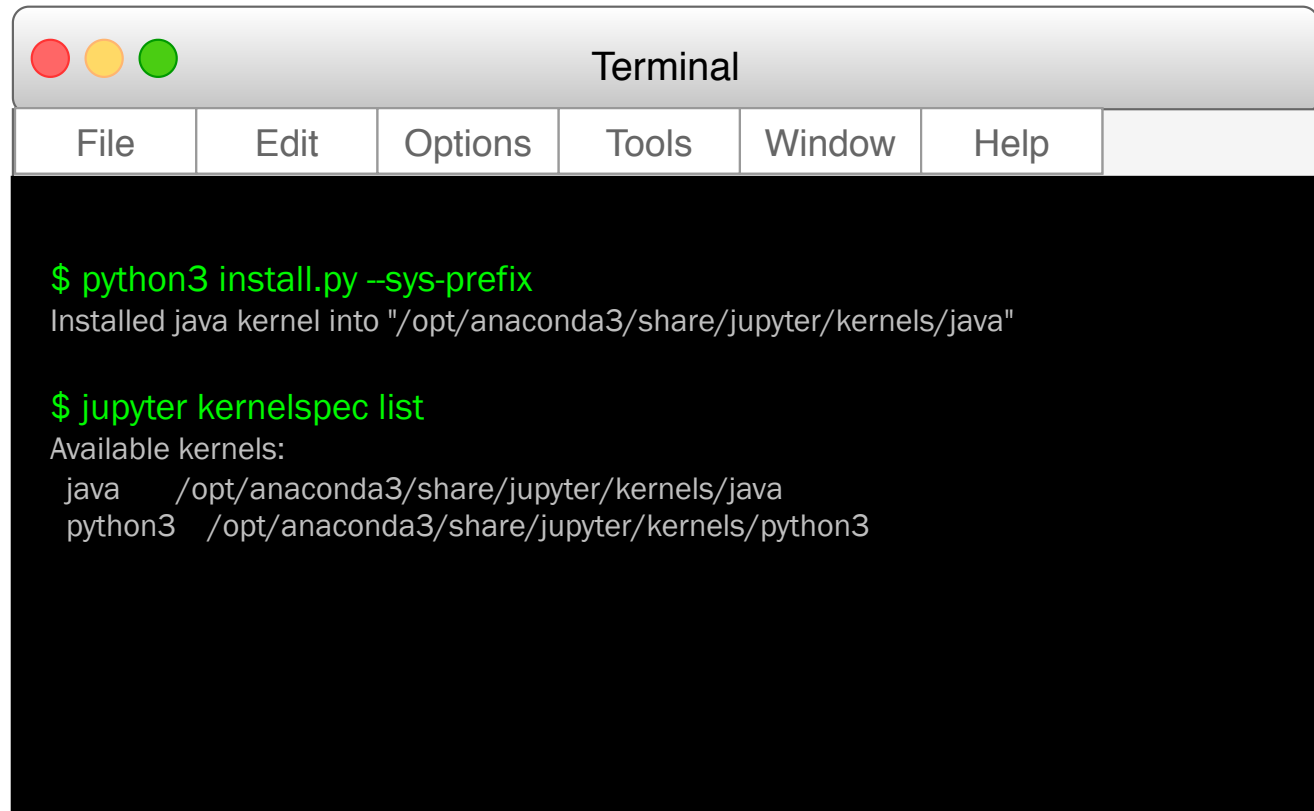
# Setting Up IJava for Jupyter

Download the latest Zipped IJava from Github:  
<https://github.com/SpencerPark/IJava/releases>

Unzip the file and execute the following:

```
python3 install.py --sys-prefix
```

Check that it was installed successfully with `jupyter kernelspec list`, which should contain java

A screenshot of a macOS Terminal window. The title bar is light gray with three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Tools", "Window", "Help", and a small separator icon. The main area of the terminal is black with green text. It shows the command `$ python3 install.py --sys-prefix` being executed, followed by the output `Installed java kernel into "/opt/anaconda3/share/jupyter/kernels/java"`. Then, the command `$ jupyter kernelspec list` is executed, followed by the output "Available kernels:" and a list of two kernels: `java` at `/opt/anaconda3/share/jupyter/kernels/java` and `python3` at `/opt/anaconda3/share/jupyter/kernels/python3`.

```
Terminal

File Edit Options Tools Window Help

$ python3 install.py --sys-prefix
Installed java kernel into "/opt/anaconda3/share/jupyter/kernels/java"

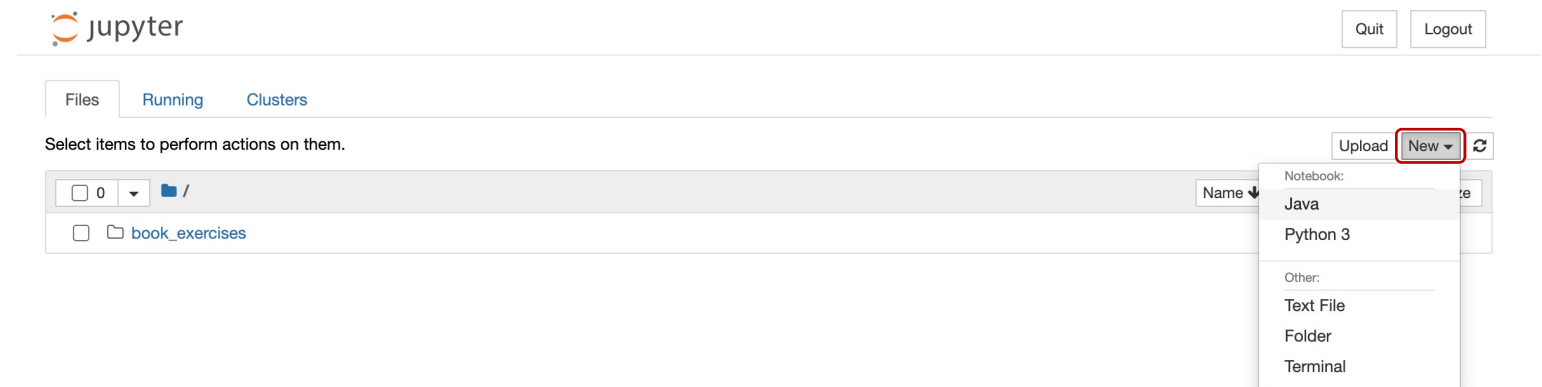
$ jupyter kernelspec list
Available kernels:
java      /opt/anaconda3/share/jupyter/kernels/java
python3   /opt/anaconda3/share/jupyter/kernels/python3
```

# Launching IJava for Jupyter

Launch jupyter and ensure that Java is included in the list of kernels by clicking the “New” dropdown menu.

You may also launch Jupyter in IJava mode from the command line as follows:

```
jupyter console --kernel=java
```

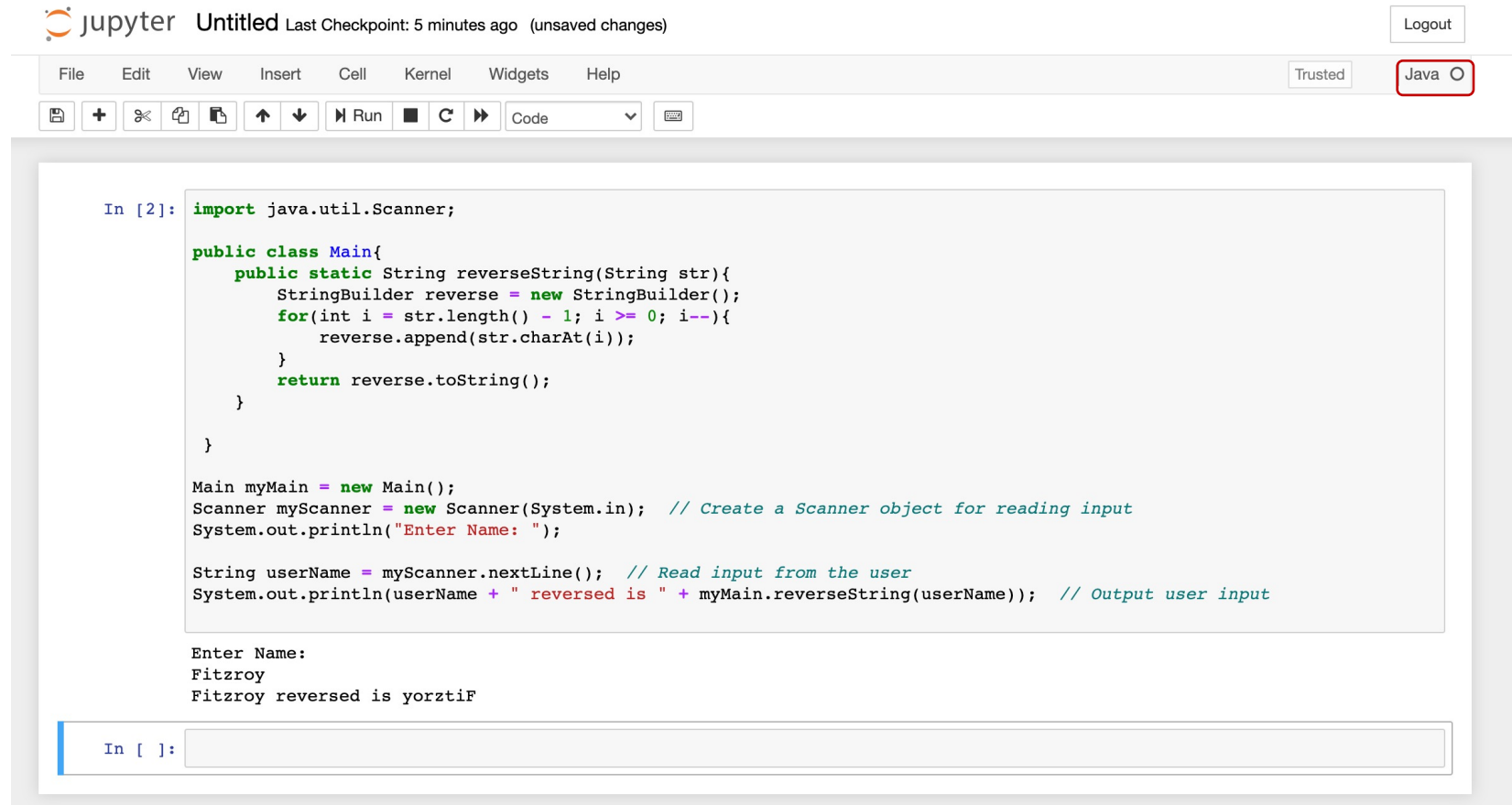


# Testing Java for Jupyter

Enter Java Code in your Notebook as shown in the screenshot on the right.

This example demonstrates the use of an Object (OOP) by reading a name from the user and reversing it.

Notice the highlighted kernel named Java



The screenshot shows a Jupyter Notebook titled "Untitled" with a last checkpoint of 5 minutes ago. The interface includes a top bar with a "Logout" button and a "Java" kernel selector. Below the top bar is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A toolbar contains icons for saving, adding, deleting, and running cells, along with a "Run" button and a "Code" dropdown menu. The main area displays a code cell with the following Java code:

```
In [2]: import java.util.Scanner;

public class Main{
    public static String reverseString(String str){
        StringBuilder reverse = new StringBuilder();
        for(int i = str.length() - 1; i >= 0; i--){
            reverse.append(str.charAt(i));
        }
        return reverse.toString();
    }
}

Main myMain = new Main();
Scanner myScanner = new Scanner(System.in); // Create a Scanner object for reading input
System.out.println("Enter Name: ");

String userName = myScanner.nextLine(); // Read input from the user
System.out.println(userName + " reversed is " + myMain.reverseString(userName)); // Output user input
```

The output of the code is displayed below the code cell:

```
Enter Name:
Fitzroy
Fitzroy reversed is yorztiF
```

At the bottom, there is an input field for the next code cell, labeled "In [ ]:".

# Executing C Code from a Jupyter Notebook

---

- The following slides will show you how to run a Reverse-String program in C via Jupyter Notebook.



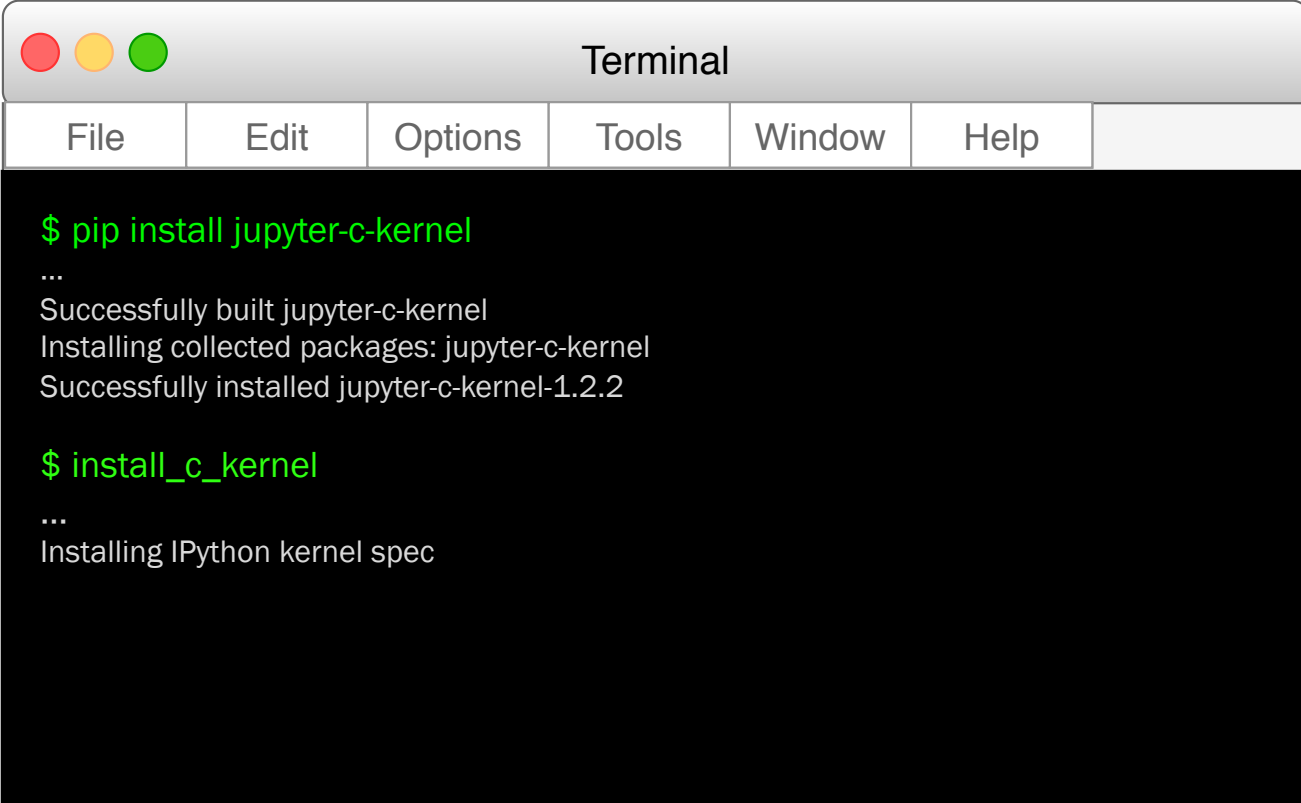
# Setting Up a C Kernel for Jupyter

## Mac and Linux Users

Execute in your Terminal the commands shown in the graphic to install jupyter-c-kernel.

## Dependencies

- gcc
- jupyter
- python 3
- pip



```
Terminal
File Edit Options Tools Window Help
$ pip install jupyter-c-kernel
...
Successfully built jupyter-c-kernel
Installing collected packages: jupyter-c-kernel
Successfully installed jupyter-c-kernel-1.2.2

$ install_c_kernel
...
Installing IPython kernel spec
```

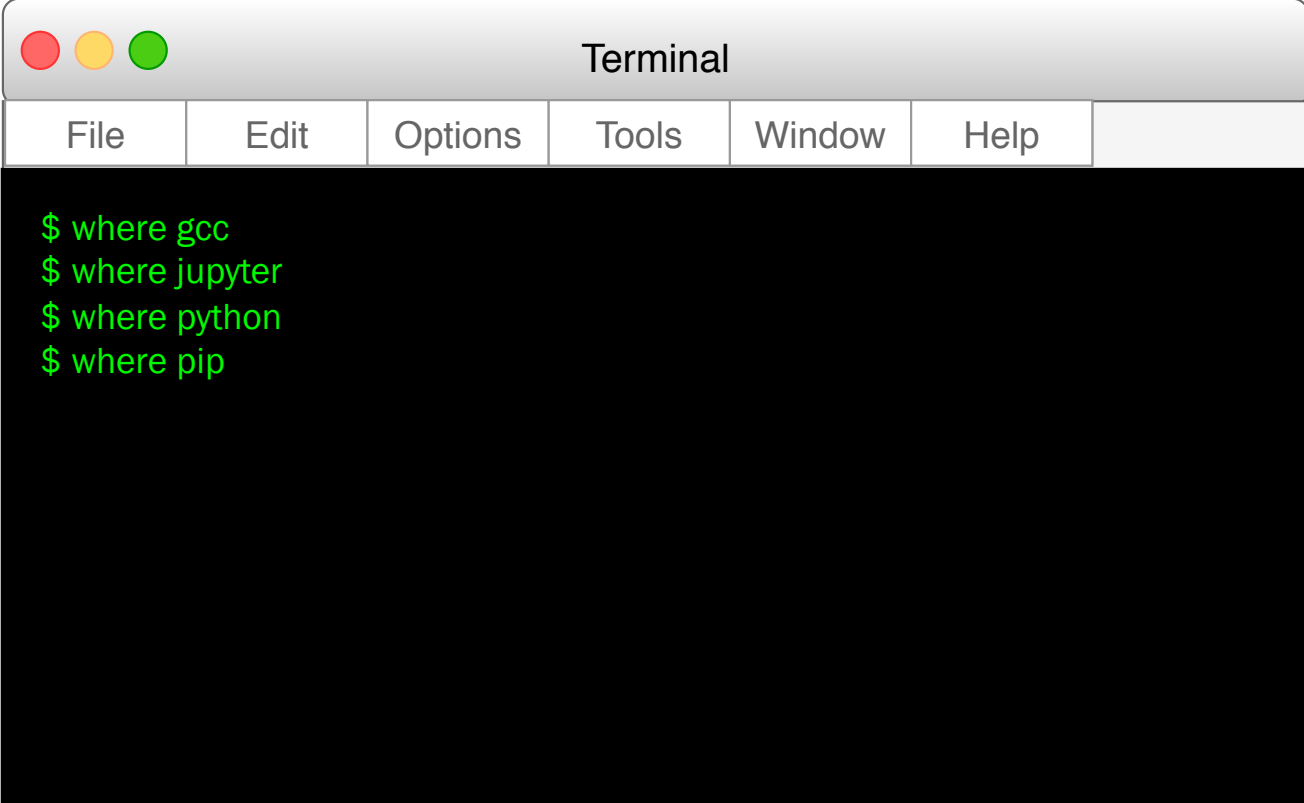
# Setting Up a C Kernel for Jupyter

## Windows Users

Make sure the following dependences are installed. Then install docker using the instructions on the following slides.

## Dependencies

- gcc
- jupyter
- python 3
- pip



```
Terminal
File Edit Options Tools Window Help
$ where gcc
$ where jupyter
$ where python
$ where pip
```

# Setting Up a C Kernel for Jupyter

## Windows Users

Install Docker Desktop for windows

Link: <https://docs.docker.com/get-docker/>

## Get Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

You can download and install Docker on multiple platforms. Refer to the following section and choose the best installation path for you.



### Docker Desktop for Mac

A native application using the macOS sandbox security model which delivers all Docker tools to your Mac.



### Docker Desktop for Windows

A native Windows application which delivers all Docker tools to your Windows computer.



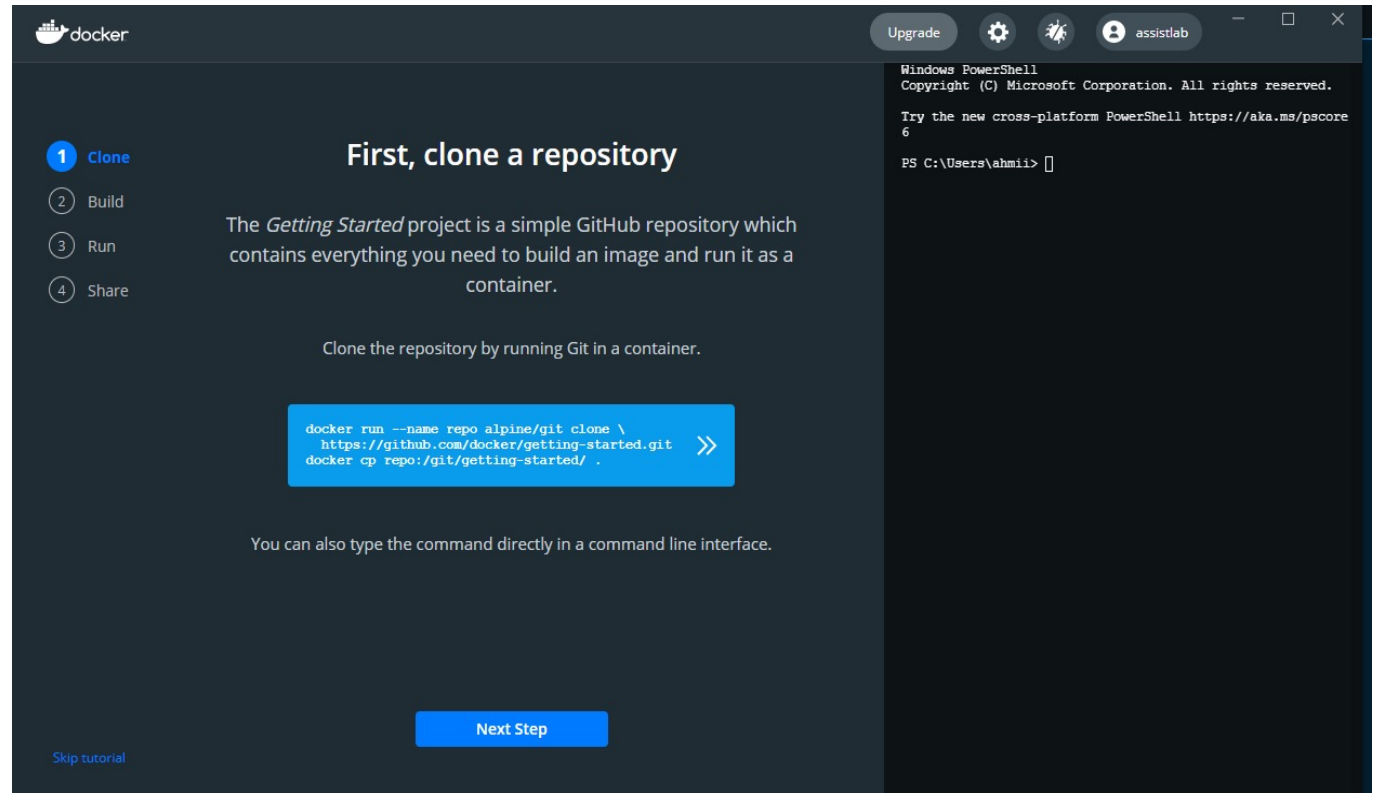
### Docker for Linux

Install Docker on a computer which already has a Linux distribution installed.

# Setting Up a C Kernel for Jupyter

## Windows Users

Open the docker App

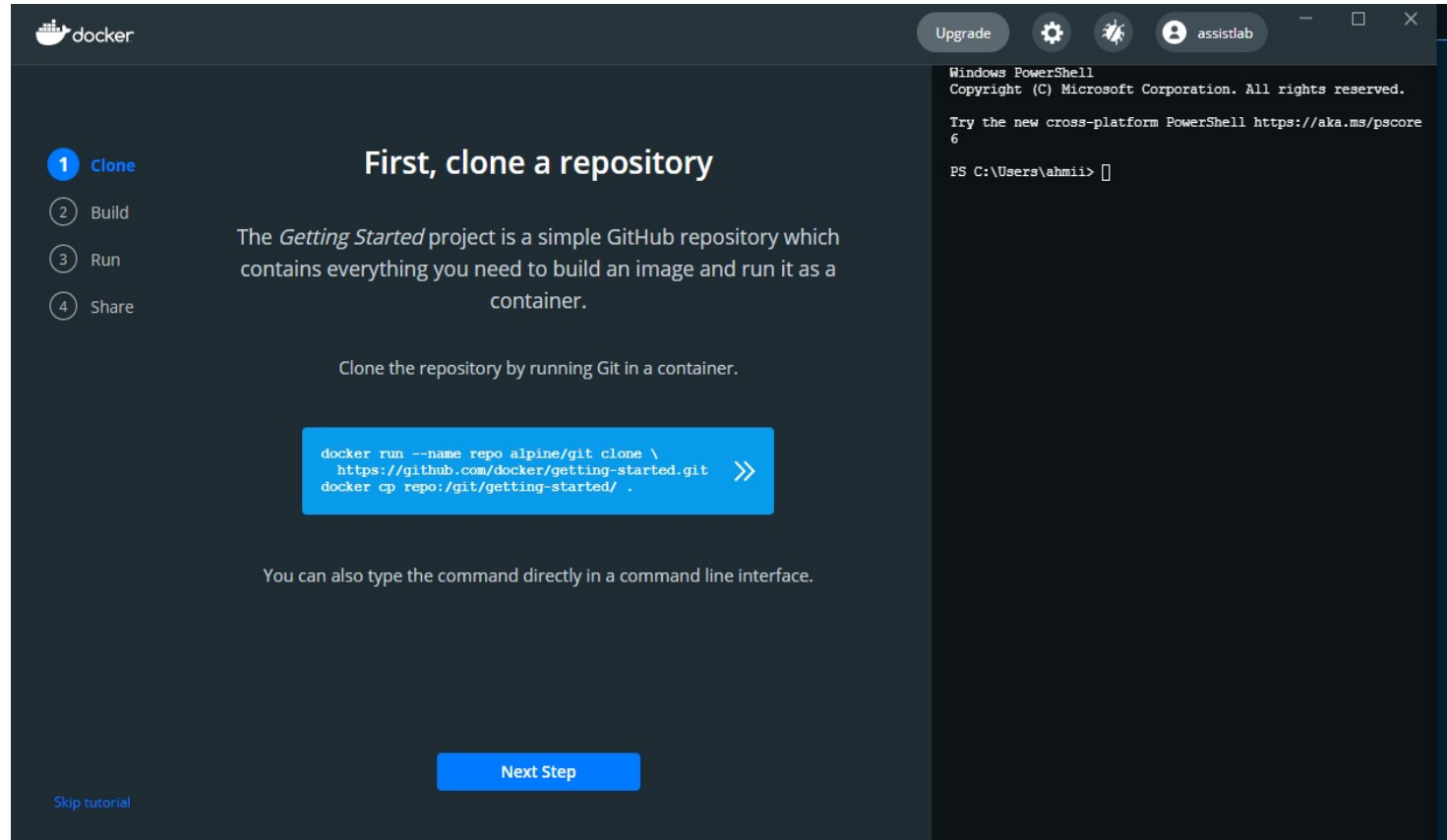




# Setting Up a C Kernel for Jupyter

## Windows Users

Follow the instructions to set up Docker



The screenshot shows the Docker Desktop application window. The top bar includes the Docker logo, an 'Upgrade' button, and icons for settings, Docker Desktop status, and a user profile labeled 'assistlab'. The main content area is titled 'First, clone a repository' and includes a sidebar with steps: 1. Clone (selected), 2. Build, 3. Run, and 4. Share. The main text explains that the 'Getting Started' project is a simple GitHub repository used for building and running a container. It instructs the user to clone the repository by running Git in a container. A blue box contains the following command: `docker run --name repo alpine/git clone \`  
`https://github.com/docker/getting-started.git`  
`docker cp repo:/git/getting-started/ .` followed by a double arrow icon. Below this, it states that the command can also be typed directly in a command line interface. A 'Next Step' button is at the bottom right, and a 'Skip tutorial' link is at the bottom left. On the right side of the window, a Windows PowerShell terminal is open, showing the copyright notice for Microsoft Corporation and the prompt 'PS C:\Users\ahmii>'.

1 Clone

2 Build

3 Run

4 Share

### First, clone a repository

The *Getting Started* project is a simple GitHub repository which contains everything you need to build an image and run it as a container.

Clone the repository by running Git in a container.

```
docker run --name repo alpine/git clone \
https://github.com/docker/getting-started.git
docker cp repo:/git/getting-started/ .
```

You can also type the command directly in a command line interface.

Next Step

Skip tutorial

Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
Try the new cross-platform PowerShell https://aka.ms/pscore6  
  
PS C:\Users\ahmii>

# Setting Up a C Kernel for Jupyter

## Windows Users

Open Powershell and run the following command:

```
docker pull brendanrius/jupyter-c-kernel
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\ahmii> docker pull brendanrius/jupyter-c-kernel
Using default tag: latest
latest: Pulling from brendanrius/jupyter-c-kernel
a48c500ed24e: Pull complete
1e1de00ff7e1: Pull complete
0330ca45a200: Pull complete
471db38bcfbf: Pull complete
0b4aba487617: Pull complete
730fa79a87e1: Pull complete
9a1a2a0601f1: Pull complete
4dc04ee9eef2: Pull complete
eef2fb844663: Pull complete
ecf7f2c6e9b9: Pull complete
f7201843ff55: Pull complete
07c028e9ca00: Pull complete
480c990dae7d: Pull complete
67ae4c6962d6: Pull complete
59cce7722c55: Pull complete
22dfb185fb26: Pull complete
3dea48053c81: Pull complete
61d27e3f91b4: Pull complete
52de7fa20679: Pull complete
9afa27de8448: Pull complete
f6413a204766: Pull complete
62635b875e05: Pull complete
Digest: sha256:de75bd8d73643fd286f7449a9a06b4dd378ee4db74b57aac0f34b66f5ac68451
Status: Downloaded newer image for brendanrius/jupyter-c-kernel:latest
docker.io/brendanrius/jupyter-c-kernel:latest
PS C:\Users\ahmii>
```

# Setting Up a C Kernel for Jupyter

## Windows Users

Next, run the following command

```
docker run -p 8888:8888 brendanrius/jupyter-c-kernel
```

Copy the URL displayed and paste it in your browser.

```
PS C:\Users\ahmii> docker run -p 8888:8888 brendanrius/jupyter-c-kernel
/usr/local/bin/start-notebook.sh: ignoring /usr/local/bin/start-notebook.d/*

Container must be run with group "root" to update passwd file
Executing the command: jupyter notebook
[I 16:56:04.501 NotebookApp] Writing notebook server cookie secret to /home/jovyan/.local/share/jupyter/runtime/notebook_cookie_secret
[W 16:56:04.627 NotebookApp] WARNING: The notebook server is listening on all IP addresses and not using encryption. This is not recommended.
[I 16:56:04.651 NotebookApp] JupyterLab beta preview extension loaded from /opt/conda/lib/python3.6/site-packages/jupyterlab
[I 16:56:04.651 NotebookApp] JupyterLab application directory is /opt/conda/share/jupyter/lab
[I 16:56:04.655 NotebookApp] Serving notebooks from local directory: /home/jovyan
[I 16:56:04.655 NotebookApp] 0 active kernels
[I 16:56:04.655 NotebookApp] The Jupyter Notebook is running at:
[I 16:56:04.655 NotebookApp] http://[all ip addresses on your system]:8888/?token=dca6fd2280610ace1a8e8c54e4e8ddc4cb2837f8786be7c0
[C 16:56:04.656 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=dca6fd2280610ace1a8e8c54e4e8ddc4cb2837f8786be7c0
```

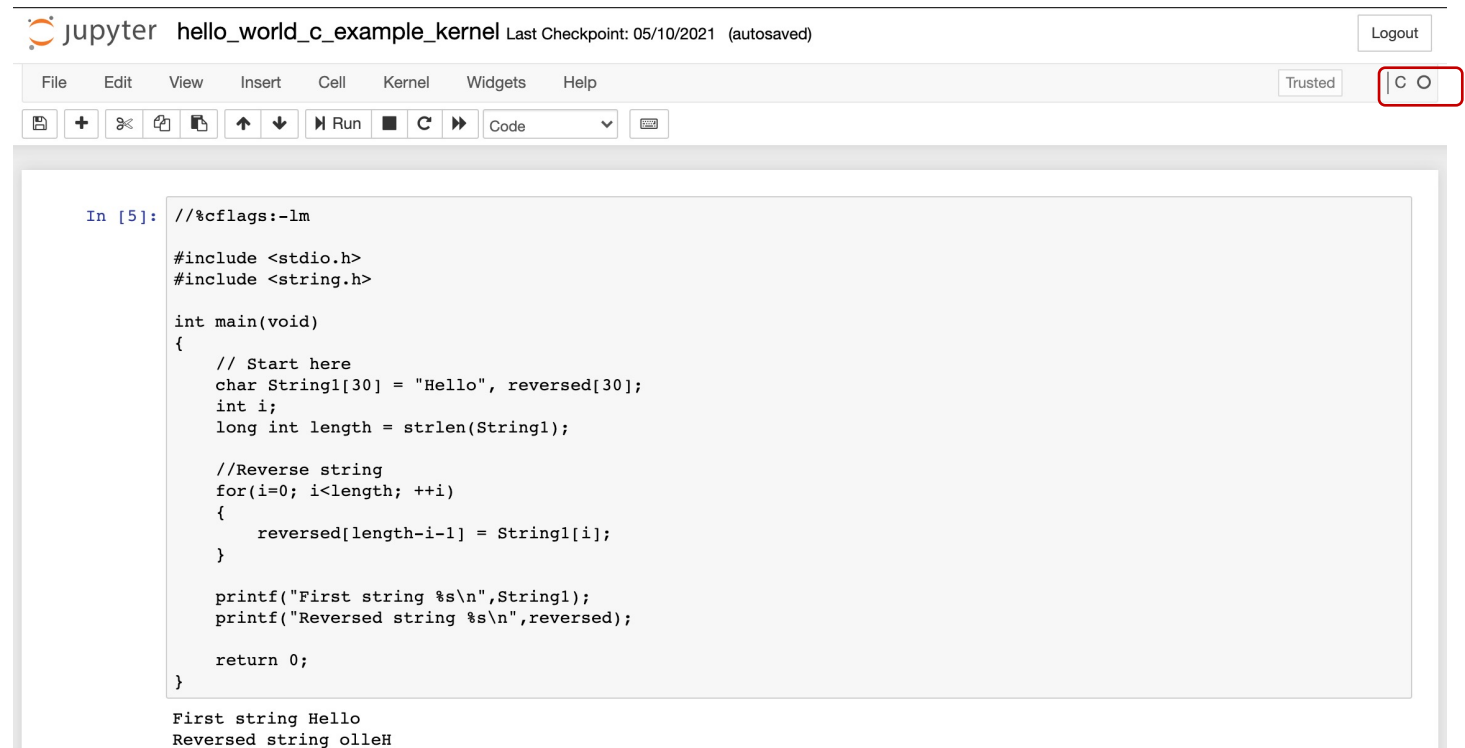
# Executing C Code in Jupyter

Enter C Code in your Notebook as shown in the screenshot on the right.

This example demonstrates reversing a string.

Notice the highlighted kernel named C.

The line `///cflags:-lm` is used to include cflags. Here `-lm` includes the math library. Use this same convention to include other libraries.



The screenshot shows a Jupyter Notebook interface. At the top, the title bar reads "jupyter hello\_world\_c\_example\_kernel" followed by "Last Checkpoint: 05/10/2021 (autosaved)" and a "Logout" button. Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". A "Trusted" status indicator is visible. On the right side of the menu bar, there is a dropdown menu with "C" selected and highlighted by a red box. Below the menu bar is a toolbar with icons for saving, adding cells, undo, redo, and running code. The main area of the notebook contains a code cell with the following C code:

```
In [5]: ///cflags:-lm
#include <stdio.h>
#include <string.h>

int main(void)
{
    // Start here
    char String1[30] = "Hello", reversed[30];
    int i;
    long int length = strlen(String1);

    //Reverse string
    for(i=0; i<length; ++i)
    {
        reversed[length-i-1] = String1[i];
    }

    printf("First string %s\n",String1);
    printf("Reversed string %s\n",reversed);

    return 0;
}
```

Below the code cell, the output is displayed:

```
First string Hello
Reversed string olleH
```

# References

The following references were used to create this tutorial.

- <https://github.com/SpencerPark/IJava#install-pre-built-binary>
- <https://github.com/brendan-rius/jupyter-c-kernel>