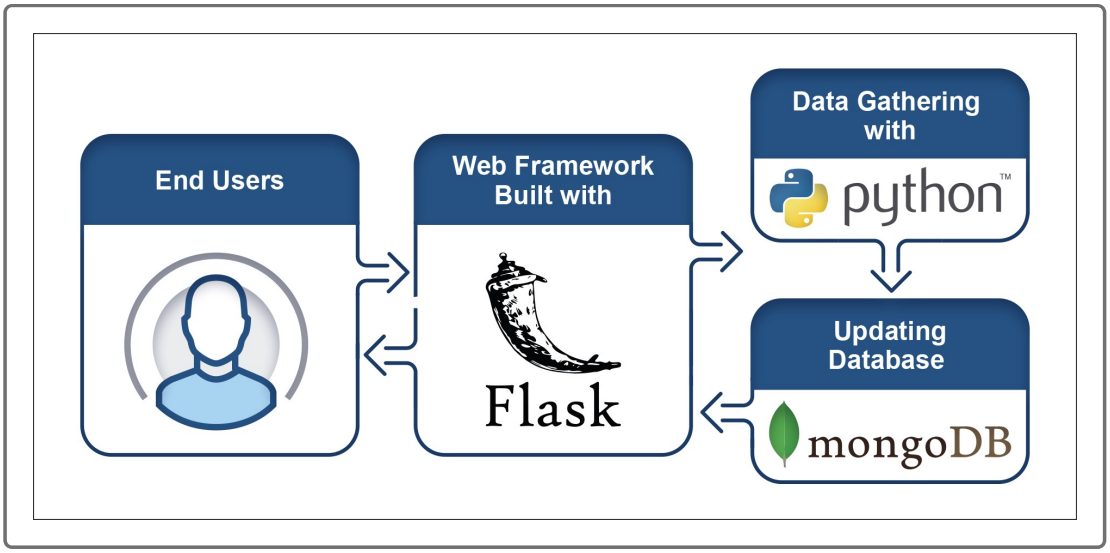
10.1.1 Install Your Tools

Robin is pretty excited about putting together this web-scraping project. Being able to get the latest news and updates with the click of a button? That's a really useful tool for someone who wants to keep up with the Mission to Mars.

First things first, though—preparation. Robin needs to download a few libraries and tools that she'll need when she's ready to start scraping data: Splinter to automate a web browser, BeautifulSoup to parse and extract the data, and MongoDB to hold the data that has been gathered.

With all of the information available on the web, people are able to stay up-to-date with almost every subject out there. What if a person wants to narrow their focus to a single topic? Are there tools that would make gathering the latest data easier?

Robin, who loves astronomy and wants to work for NASA one day, has decided to use a specific method of gathering the latest data: web scraping. Using this technique, she has the ability to pull data from multiple websites, store it in a database, then present the collected data in a central location: a webpage.



IMPORTANT

> Before installing new tools, open your terminal and make sure your Python coding environment is active.

Let's get the tools we'll need to complete this web-scraping project: Splinter, BeautifulSoup, and MongoDB.

Splinter

Splinter is the tool that will automate our web browser as we begin scraping. This means that it will open the browser, visit a webpage, and then interact with it (such as logging in or searching for an item). To do all of this, we'll need to install Splinter and ChromeDriver.

To install Splinter, open your terminal and make sure your coding environment is active. Then, run the command `pip install splinter`. Once that installation is complete, we'll install ChromeDriver.

IMPORTANT

> To successfully use the ChromeDriver and scrape website data, a

> separate package will need to be installed into our virtual environment.

Web-Driver Manager

The web driver manager package will allow us to easily use a driver that to scrape websites without having to go through the complicated process of installing the stand alone ChromeDriver.

To install the manager make sure you are still in your active coding environment and run the command `pip install webdriver_manager`.

BeautifulSoup

To install BeautifulSoup, run the command `pip install bs4` in your terminal. Make sure the environment you plan to work from is active first.

MongoDB

MongoDB (also known as Mongo) is a document database that thrives on chaos. Well, maybe it's not that extreme, but it is far more flexible when it comes to storing data than a structured database such as SQL. It's able to handle smaller, more personal projects as well as larger-scale projects that a company might require. For this module, Mongo is a better choice than SQL because the data we'll scrape from the web isn't going to be uniform. For example, how would we break down an image into rows and columns? We can't. But Mongo will store and access it as a document instead.

IMPORTANT

> To install PyMongo, first open a terminal window (make sure your virtual environment is active) and execute the "pip install pymongo"

> command. PyMongo is the tool that allow developers to use Python with Mongo.

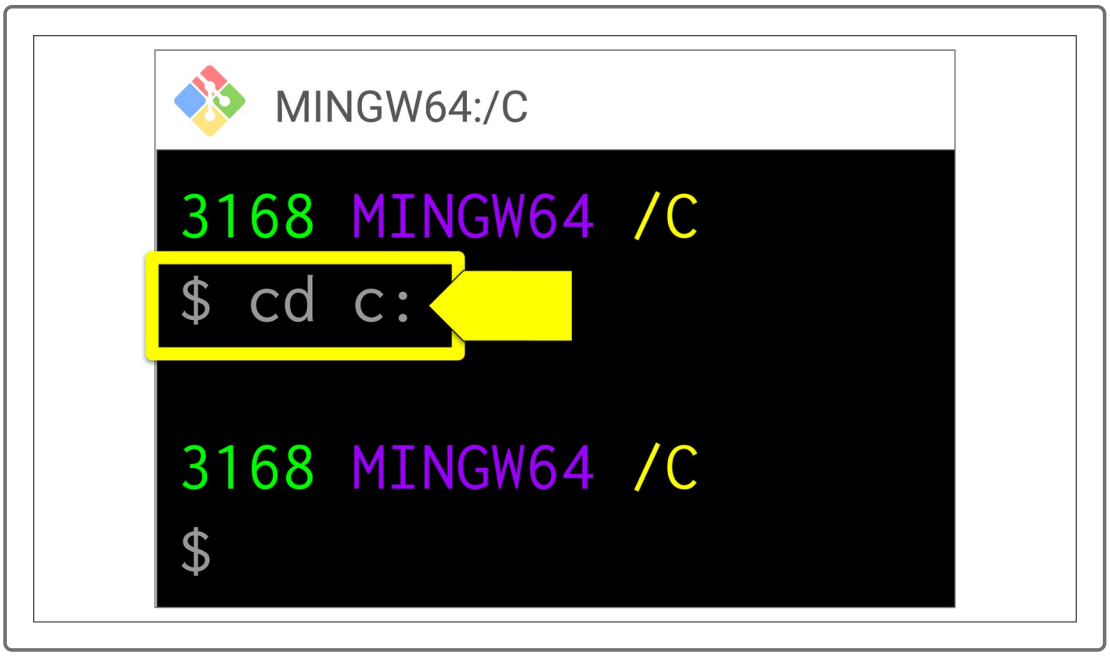
Now let's install MongoDB.

Windows

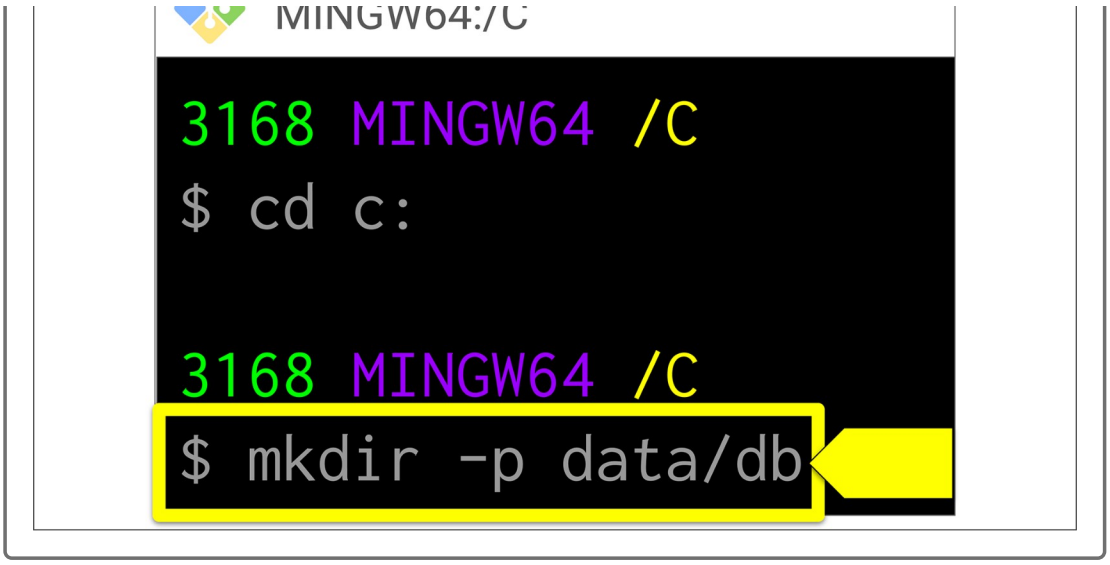
To install Mongo on your Windows computer, follow the instructions in the **official documentation (https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/)** . Be sure to follow all of the steps listed for installing the MongoDB Community Edition.

Adding a Data Directory to Root

You'll also need to add the data directory to your root directory on your computer; otherwise, MongoDB won't run. Open your Anaconda Prompt terminal, then navigate to your root directory with `cd c:\`.



Next, we want to make a directory with the command `mkdir data\db`.



This is the default location for Mongo's databases. Without this, Mongo won't have a designated spot to create collections and store data. It will refuse to run, no matter how many cookies we offer it.

Configuration

You'll also want to add MongoDB's path to the PATH environment variables for your computer so that you can run and launch MongoDB easily from the command line.

First, locate the directory where you installed MongoDB. This is likely `C:\\Program Files\\MongoDB\\Server\\\\bin`. Copy this directory to your clipboard with CTRL + C.

Once the file path is highlighted, we can add it to our PATH environment variable. To update your Path, dig into your security settings on your computer. See the instructions below.

Next, the following video will aid in launching Mongo from your command line interface.

To test if this worked, close your current Anaconda Prompt window, and then reopen it and run this command: `mongod`. If Mongo continues to run in the terminal, it's been successfully installed—great job!

NOTE

> There is no "b" in the `mongod` command.

Troubleshooting

If `mongod` didn't run and, instead, your Anaconda Prompt threw a "command not found" error, make sure you added MongoDB's **bin** directory to your PATH variable. When you navigate through your Windows Explorer to the MongoDB folder, make sure you click all the way through into the **bin** folder, then copy the path. Then, close out Anaconda Prompt and try running `mongod` again.

If `mongod` starts but closes after a series of prompts, make sure you created the `\data\db` directory in your root. MongoDB cannot run without this directory.

macOS

Installation

To install Mongo on your macOS computer, follow the instructions in the **official documentation (https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/)** . Be sure to follow all of the steps listed for installing the MongoDB Community Edition.

Here are a few important tips during this installation:

- It's best to use the Homebrew `brew` package for this installation. In your terminal, the installation command will start with `brew tap`.

- Once Mongo is installed, we want to run it as a **macOS service** because doing so will automatically set system **ulimit (https://docs.mongodb.com/manual/reference/ulimit/#ulimit-settings)** values correctly.

NOTE

> Mac users will use this line to create a database instance: `brew services start mongodb-community@4.4` instead of `mongod`.

GITHUB

Navigate to GitHub and create a new repository to hold the code for this module. Name the new repo "Mission-to-Mars" and clone it into your class folder. Remember to add, commit, and push your code as you work through the module.

Flask-PyMongo

To bridge Flask and Mongo, you'll also want to install the (Flask-PyMongo) [https://flask-pymongo.readthedocs.io/en/latest/] library. This library can be installed using pip and the following command from your terminal: `pip install Flask-PyMongo`.

Additional Libraries

There are two final Python libraries required to run scraping code successfully: **html5lib** and **lxml**. Both packages are used to parse HTML in Python, which will be important as you traverse through different web pages to find and collect information.

To install these libraries, first make sure your coding environment is active. Then, type the following commands in your terminal to install them:

1. `pip install html5lib`
2. `pip install lxml`

NOTE

> Some environments may already have these packages installed; that's totally fine, just continue on to the next section!