**15 Minute Code Dive**

***Setting up a local server to host a web page***

**Introduction**

In this <15 minute code dive you will first create a web page, then you will set up a server on your computer (aka a local server) using something called SimpleHTTPServer to host your web page. The purpose of this Code Dive is to get your feet wet in back-end web development and to introduce you to several reoccurring topics.

**What is a server?**

In the most general terms, a server is a device that provides services to other devices. In this Code Dive the service that our server will provide is access to an HTML file. In other words, devices will request access to an HTML file from our server, and our server will return the HTML file to that device.

**What is SimpleHTTPServer?**

SimpleHTTPServer is a Python module which allows you to instantly create a web server. The main advantage of Python’s SimpleHTTPServer is that you only need to have Python installed for it to function, making setup short and sweet.

**Step 0: Make sure Python is installed**

Open up Command Prompt (Windows) or Terminal (Mac), and ensure Python is installed on your system by entering the following command:

python --version

If Python is installed, this command will return “Python X.Y.Z”, where X.Y.Z is your version number. If Python is not installed, download Python 2.7 here: [**https://www.python.org/downloads/**](https://www.python.org/downloads/).

Your Python version might start with the number 2 or 3. These are different major versions of Python and have some significant differences. Most of the Full Stack Nanodegree program is built with Python 2.x in mind.

We'll review this installation procedure in more depth later in the Nanodegree program. If you'd like more detailed instructions, navigate to the first lesson of Programming Foundations with Python.

**Step 1: Create a HTML page**

Create a folder called LocalServer and in that create an HTML file named index.html. You can aesthetically design this page however you’d like but for now you can just add the following code:

<!DOCTYPE html>

<html>

<head>

<title>Welcome to the Full Stack Nanodegree Program!</title>

</head>

<body>

Welcome to the Full Stack Nanodegree Program!

</body>

</html>

**Step 2: Create a SimpleHTTPServer**

Open your terminal and cd to your LocalServer directory. Type the following command:

python -m SimpleHTTPServer 8080

This command just created your local server with a port number of 8080. This can be accessed via your browser at [**http://localhost:8080/**](http://localhost:8080/). “localhost” tells your browser that the server is located on your device. And “:8080” tells your browser that the port number is 8080, but…

**What is a port?**

Servers can provide a variety of services simultaneously. For example, a server can provide access to an HTML file like the one we are currently setting up, and our server can also provide the service of writing to a database; and these services can be completely unrelated. But when another device wants to request a service from a server, how does the server know which service to run? The way this is accomplished is by using what they call ports. Each service on a server will have a unique port number. In our case our service’s port number is 8080. Here is a helpful metaphor to elaborate on what’s going on:

*Port numbers are much like extensions for phone numbers. Imagine you call an office building with the number (012) 345-6789, and use the extension 42 to reach a specific phone.*

*Here is a breakdown of the equivalences from the above metaphor:*

**Company** | Local Server

(012) 345-6789 | http://localhost:

42 | 8080

*Note:* “http://” can be thought of as using a telephone as the means of communication rather than fax or email. You will learn much more about HTTP in the Networking course later on.

**Step 3: View Your Web Page**

If you haven’t already, now it’s time to use your local server by visiting [**http://localhost:8080/**](http://localhost:8080/). In technical lingo: we want to make a request to the port 8080 on our local server. You are now viewing your web page from your local server. When you visit a web address, such as [**http://localhost:8080/**](http://localhost:8080/), the default response of that service is to return index.html if it exists.

From here, feel free to embellish your index.html page to make it look more appealing! If you have any questions, comments, or concerns please join the community on the discussion forums.