**PYTHON GETTING STARTED**

**PIP (Preferred Installer Program)**

PIP lets us install packages in our programs that are not already included in python like the math modules and other modules

**Venv (Virtual Environment)**

A self-contained directory tree that contains a Python installation for a particular version of Python, plus a number of additional packages.

Different applications can use different virtual environments

The original design of Python and its packaging system puts installed packages alongside the Python interpreter in the file system; furthermore, only a single version of a given package can be installed with a given Python interpreter.

In a nutshell, Python virtual environments helps decouple and isolate Python installs and associated pip packages. This allows end users to install their own set of packages that are independent of those provided by the system or by other projects.

**WHAT IS A PYTHON VIRTUAL ENVIRONMENT**

(venv) is simply a directory with a particular file structure.

**PIP**

Is a package manager for python packages, or modules?

**What is a Package?**

A package contains all the files you need for a module.

Modules are Python code libraries you can include in your project.

**Python Try Except**

The try block lets you test a block of code for errors.

The except block lets you handle the error.

The finally block lets you execute the code, regardless of the results of the try and except blocks.

**Exception Handling**

When an error occurs, or exception as we call it, Python will normally stop and generate an error message.

These exceptions can be handled using the try statement.

**Function**

Is a block of code that only runs when it’s called? You can pass data, known as parameters, into a function. A function can return data as a result.

**Creating a function**

In Python a function is defined using the def keyword

**Calling a Function**

To call a function, use the function name followed by a parenthesis.

**Arguments**

Information can be passed into functions as parameters when creating them. Parameters are specified after the function name, inside the parenthesis. You can add as many parameters as you want, just separate them with a comma.

Arguments are passed inside the brackets when the functions are called.

**Parameters or Arguments**

The terms parameters and arguments can be used for the same thing: information that is passed into a function.

**From a function’s perspective:**

A parameter is the variable listed inside the parenthesis in the function definition while an argument is the value that are sent to the function when it’s called.

**Number of Arguments**

By default, a function must be called with the correct number of arguments. Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less.

**Arbitrary Arguments, \*args**

If you do not know how many arguments that will be passed into your function, add a \* before the parameter name in the function definition. This way the function will receive a tuple of arguments, and can access the items accordingly.

Arbitrary arguments are often shortened to \*args in Python documentations.

**Keyword Arguments**

You can also send arguments with key = value syntax. This way the order of arguments does not matter.

The phrase Keyword Arguments are often shortened to kwargs in Python documentations.

Arbitrary Keyword Arguments

If you do not know how many keyword arguments that will be passed into your function, add two Asterix: \*\* before the parameter name in the function definition. This way the function will receive a dictionary of arguments, and can access the items accordingly:

Arbitrary keyword arguments are often shortened to \*\*kwargs in Python documentation.

Default Parameter Value

If we call the function without argument, it uses the default parameter value.

Passing a List as an Argument

You can send any data types of argument to a function (string, number, list, dictionary etc.), and it will be treated as the same datatype inside the function.

Return Values

To let a function, return a value, use the return statement.

The pass Statement

function definitions cannot be empty, but if you for some reason have a function definition with no content, put in the pass statement to avoid getting an error.

Recursion

Python also accepts recursion, which means a defined function can call itself. Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result.

The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amount of memory or processor power. However, when written correctly recursion can be a very efficient and mathematically-elegant approach to programming.

Python Lambda

A lambda function is a small anonymous function. A lambda function can take any number of arguments but can only have one expression.

Syntax

Lambda arguments : expression

The expression is executed and the result is returned. Lambda functions can take any number of arguments.

Why Use Lambda Functions?

The power of Lambda is better shown when you use them as an anonymous function inside another function. Say you have a function definition that takes one argument, and that argument will be multiplied with an unknown number. Use lambda functions when an anonymous function is required for a short period of time.

Arrays

Arrays are used to store multiple values in one single variable. So an array is a special variable which can hold more than one value at a time and you can access the values by referring to an index number.

Looping Array Elements

You can use the **for in** loop to loop through all the elements of an array.

Adding Array Elements

You can use the append () method to add an element to an array.

Removing Array Elements

You can use the pop () method to remove an element from the array.

You can also use the remove () method to remove an element from the array. The list’s remove () method only removes the first occurrence of the specified value.

Keywords

Python Casting

Specify a Variable Type

Casting in Python is therefore done using constructor functions:

Int () – constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number), or a string literal(providing a string represents a whole number)

Float () – constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)

Str () – constructs a string from a wide variety of data types, including strings, integer literals, and float literals.

THE BACKEND

It consists of the server, application and the database for storing data.

Application: Creates the business logic and determining how the website works such as calculations in the website.

Server: Serves up your files, your html, CSS and JavaScript

THE POWER OF THE COMMANDLINE AND HOW TO USE NODE.JS

**GETTING STARTED WITH NODE.JS**

Node.js is an open source server environment.

Node.js allows you to run JavaScript on the server.

Node.js uses asynchronous programming.

A common task for a web server can be to open a file on the server and return the content to the client.

**Here is how Node.js handles a file request:**

1.Sends the task to the computer’s file system.

2.Ready to handle the next request.

3.When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting and simply continues with the next request.

Node.js runs single-threaded, non-blocking, asynchronous programming, which is very memory efficient.

**What Can Node.js Do?**

Node.js can generate dynamic page content.

Node.js can create, open, read, write, delete, and close files on the server.

Node.js can collect form data.

Node.js can add, modify and delete data in your database.

**What Is a Node.js File**

Node.js files contain tasks that will be executed on certain events.

A typical event is someone trying to access a port on the server.

Node.js files must be initiated first on the server before having any effect.

Node.js files must have extension “.js”

**COMMAND LINE INTERFACE**

Node.js files must be initiated in the “Command Line Interfaces” program of your computer.

**Initiate the Node.js File**

The file you have just created must be initiated by Node.js before any action can take place.

Start your command line interface, write **node myfirst.js** and hit enter.

Now your computer can work as a server!

If anyone tries to access your computer on port 8080, they’ll get a “Hello World!” message in return!

**Node.js Modules**

What is a Module in Node.js?

Consider modules to be the same as JavaScript libraries.

A set of functions you want to include in your application.

**Built-in Modules**

Node.js has a set of built-in modules which you can use without any further installations.

Include Modules

To include a module, use the **require ()** function with the name of the module:

e.g.,

var http = require (‘http’);

Now your application has access to HTTP module, and is able to create a server:

**Create Your Own Modules**

You can create your own modules, and easily include them in your applications.

The following example creates a module that returns the current date and time object:

exports. myDateTime = function () {

return Date ();

};

Use exports keyword to make properties and methods available outside the module file.

Save the code above in a file called “myfirstmodule.js”

**Include Your Own Module**

var http = require(‘http’);

var datetime = require(‘./myfirstmodule’);

Notice that we use ./ to locate the module, that means that the module is located in the same folder as the Node.js file.

Node.js HTTP Module

The Built-in HTTP Module

Node.js has a built-in module called HTTP, which allows Node.js transfer data over the Hyper Text Transfer Protocol (HTTP).

Node.js as a Web Server

The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.

Use the createServer() method to create an HTTP server:

Var http = require(‘http’);

//Create a server object:

http.createServer(function (req, res){

res.write(‘Hello World!’); // write a response to the client

res.end(); //end the response

}).listen(8080); // the server object listens on port 8080

The function passed into the http.createServer() method, will be executed when someone tries to access the computer on port 8080.

Add an HTTP Header

If theresponse from the HTTP server is supposed to be displayed as HTML, you should include an HTTP header with the correct content type:

demo\_http.js

Var http = require(‘http’);

//Create a server object:

http.createServer(function (req, res){

res.writeHead(200, {‘Content-Type’:’text/html’});

res.write(‘Hello World!’); // write a response to the client

res.end(); //end the response

}).listen(8080); // the server object listens on port 8080

The first argument of the res.writeHead() method is the status code, 200 means that all is OK, the second argument is an object containing the response headers.

Read the Query String

The function passed into the http.createServer() has a req argument that represents the request from the client, as an object (http.IncomingMessage object).

This object has a property called the “url” which holds the part of the url that comes after the domain name:

demo\_http\_url.js

var http = require(‘http’);

http.createServer(function (req, res){

res.writeHead(200, {‘Content-Type’:’text/html’});

res.write(req.url); // write a response to the client

res.end(); //end the response

}).listen(8080); // the server object listens on port 8080

Split the Query String

There are built-in modules to easily split the query strings into readable parts, such as the URL module.

var http = require(‘http’);

var url = require(‘url’);

http.createServer(function(req, res) {

res.writeHead(200, {‘Content-Type’: ‘text/html’});

var q = url.parse(req.url, true).query;

var txt = q.year + “ ” + q.month;

res.end(txt);

}).listen(8080);

Node.js MongoDB

MongoDB is a NoSQL database

Node.js MongoDB Create Collection

A collection in MongoDB is the same as a table in MySQL

Creating a Collection

To create a collection in MongoDB use the createCollection() method:

In MongoDB a collection is not created until it gets content!

Node.js MongoDB Insert

Insert into Collection

To insert a record or a document as it’s called in MongoDB, into a collection, we use the insertOne() method.

A document in MongoDB is the same as a record in MySQL.

The first parameter of the insertOne() method is an object containing the name(s) and value(s) of each field in the document you want to insert.

It also takes a callback function where you can work with any errors, or the results of the insertion:

If you try to insert douments in a collection that do not exist, MongoDB will create the collection automatically.

Insert Multiple Documents

To insert multiple documents into a collection in MongoDB, we use the insertMany() method.

The first parameter of the insertMany() method is an array of objects, containing the data you want to insert.

It also takes a callback function where you can work with any errors, or the results of the insertion:

The Results Object

When executing the insertMany() method, a result object is returned.

The result object contains the information about how the insertion affected the database.

The \_id Field

If you do not specify an \_id field, then MongoDB will add one for you and assign a unique id for each document.

If you do specify the -id field, the value must be unique for each document:

Node.js MongoDB Find

In MongoDB, we use the find and findOne methods to find data in a collection.

Just like the SELECT statement is used to find data in a table in MySQL database.

Find One

To select data from a collection in MongoDB we use the findOne() method.

The findOne() method returns the first occurrence in the selection.

The first parameter of the findOne() method is a query object.

Find All

To select data from a table in MongoDB we can also use the find() method.

The find () method returns all occurrences in the selection.

The first parameter of the find () method is a query object.

Find Some

The second parameter of the find () method is the projection object that describes which fields to include in the result.

The Result Object

The result can be converted into an array containing each document as an object.