

Python Level Two

Let's learn something!





- Welcome to Python Level Two!
- In this section we will be covering more advanced topics that will allow you to use Django with ease!
- Knowing these topics will make your transition to Django very smooth!





Let's get started!

Let's learn something!





Part 1 - Scope





 We've discussed Scope a bit in the past, but Python's Scope rules can sometimes confuse beginners, so let's quickly go over the key rules of Python's Scope



- Python Scope follows the LEGB Rule:
 - Local
 - Enclosing Function locals
 - Global
 - Built-in



- L: Local Names assigned in any way within a function (def or lambda)), and not declared global in that function.
- E: (EFLs) Name in the local scope of any and all enclosing functions (def or lambda), from inner to outer.



- G: Global (module) Names assigned at the top-level of a module file, or declared global in a def within the file.
- B: Built-in (Python) Names
 preassigned in the built-in names
 module : open,range,SyntaxError,...





 Let's walk through some simple examples that make Python's scope clear!





Part 2 - Object Oriented Programming





- Object Oriented Programming is a way to use Python to create our own Objects.
- It can be a point of great confusion for beginners, mainly because often it is taught poorly!





- Let's try our best to save you from any confusion by systematically showing you the thought process behind OOP and why we would need it.
- We will use it quite a bit for Django, so let's get started!





Part 3 - OOP Project





- OOP is fundamental to becoming a good Python programmer, so let's get some extra practice by building a game!
- We will use OOP to create the Card Game War!





- The relevant file is:
 - Part3_OOP_Project.py



- Feel free to either treat this as a code-along project, or attempt it on your own first!
- Let's get a quick look at the project!



Part 3 - OOP Project Solutions





Part 4 - Errors and Exceptions





- Often times our code isn't perfect, meaning we run into Errors!
- But how do we actually set-up our own Error and Exception calls?
- Let's find out!



- We can use these keywords:
 - Try
 - Except
 - Finally
 - To dictate our code logic in case of an error!



- To show how this works we will be opening files, one way to open files is to use the open() function:
 - o open("myfile.txt",'r')



- The second parameter in the open()
 function dictates whether you are
 opening the file for just reading, just
 writing, or to do both.
- If you use the wrong one, you may get an error!



 Let's use this to show how we can handle errors!



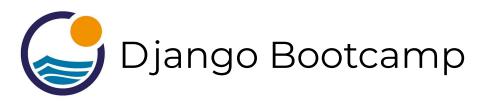


Part 5 - Decorators





- Decorators are an advanced tool in Python, feel free to skip this lecture and come back to it at another time.
- We won't encounter Decorators until much further into the Advanced Django material.



 We leave this lecture in this section because material wise this makes the most sense but as far as using it in the course, it is recommended you skip this for now and come back to it when you see decorators mentioned again!





Alright, let's get started!





Part 6 - Modules and Packages



- You've seen Python import statements, but have probably wondered, how do they work and how do we create our own?
- Let's find out!



Part 7 - Name and Main





- An often confusing part of Python is a mysterious line of code:
 - o if __name__ == "__main__":



 Sometimes when you are importing from a module, you would like to know whether a modules function is being used as an import, or if you are using the original .py file of that module.





 Let's explore this some more, but make sure to check out the full explanatory text file that is in this part's folder!



Part 8 -Regular Expressions





- Regular Expressions allow us to search for patterns in Python strings.
- They can seem incredibly intimidating at first due to their strange syntax!
- We'll walk through the basics of regular expressions, we will use them in Django!

