6.2 Lesson Summary - Working with Weather & City APIs

You can access data from both web APIs and properly formatted text files. Once you have accessed this data it can be converted into Pandas DataFrames or other formats that make analysis easier. Reading files or importing data from web servers is often an error-prone process so it is important to know how to handle errors in your code.

Concept: JSON data does not need to be accessed through web APIs it can be loaded into Python from a JSON file. For example:

*import json*

*with open(filepath) as jsonfile:*

*my\_json\_dict = json.load(jsonfile)*

* Activity: 01-Stu\_JSONTraversalReview

Concept: Data retrieved from web APIs can be loaded into Pandas DataFrames in order to utilize the functionality Pandas provides. If you utilized a web API to retrieve a list of songs and the number of times they have been downloaded you could load this into a DataFrame using the following code:

*song\_dict = {*

*"song": songs,*

*"download\_count": download\_count*

*}*

*song\_df = pd.DataFrame(song\_dict)*

* Activity: 05-Ins\_OpenWeatherDataFrame

Concept: Errors frequently occur when running code. Tasks like retrieving data from a web API are especially error-prone. If your Internet connection is interrupted, if the web API is off-line, or if the data returned from the web API is not formatted as expected you may encounter an error. Python provides a way to deal with errors in your code that allows you to address the problem instead of halting your program in its tracks. The **try-except** code block allows you to specify what should happen if errors occur. If you wanted to see what would happen if you tried to divide by zero you could use the following code:

*try:*

*will\_throw\_error = 1/0*

*except:*

*print("Sorry, you can’t divide by zero")*

* Activity: 08-Ins\_ExceptionHandling, 09-Stu\_MakingExceptions, 10-Stu\_API\_Exceptions
* Suppl link: <https://www.w3schools.com/python/python_try_except.asp>