* Course Overview
* Prerequisites
  + Imported nested packages
  + Packages are modules
  + Locating modules for import
  + Modularization
    - A -> B -> C
  + Prerequisites
  + Main Block()
    - Differentiate Module execution from module import
  + \_\_<method-name>\_\_
    - Special methods
    - Dunder(double underscore) method name
* Introduction to Packages
  + Modules
    - Python’s basic tool for organizing code
    - Normally a single python source file
    - Load modules with import
    - Represented by module objects
  + Packages: Modules that contain other modules
  + urllib is a package
    - has a \_\_path\_\_ member
  + urllib.request is a module
    - doesn’t have a \_\_path\_\_ member
  + packages are generally directories
  + modules are generally files
* Locating Modules
  + Python looks on filesystem for corresponding python file
  + Python use sys.path
    - List of directories
    - Then searched in order in import
    - First match provides module
    - ImportError when there is no match
  + sys.path[0] is empty
    - when you start interpreter with no arguments
  + can add places for python to search
  + ex)
    - import sys
    - sys.path.append(‘not\_searched’)
  + PYTHONPATH
    - Environment variable
    - Lists of paths added to sys.path when python starts
    - Windows: set PYTHONPATH=path1;path2;path3
    - Linux/macOS: export PYTHONPATH=path1:path2:path3
* Summary
  + Importing nested packages
  + All modules in hierarchy are imported
    - Only the first name is bound
    - Use fully-qualified names for submodules
  + Package directory paths are stored in \_\_path\_\_
  + Sys.path contains module search
    - It is initialized from PYTHONPATH
* Creating Packages