Devon Rudd

Muntuck Yap

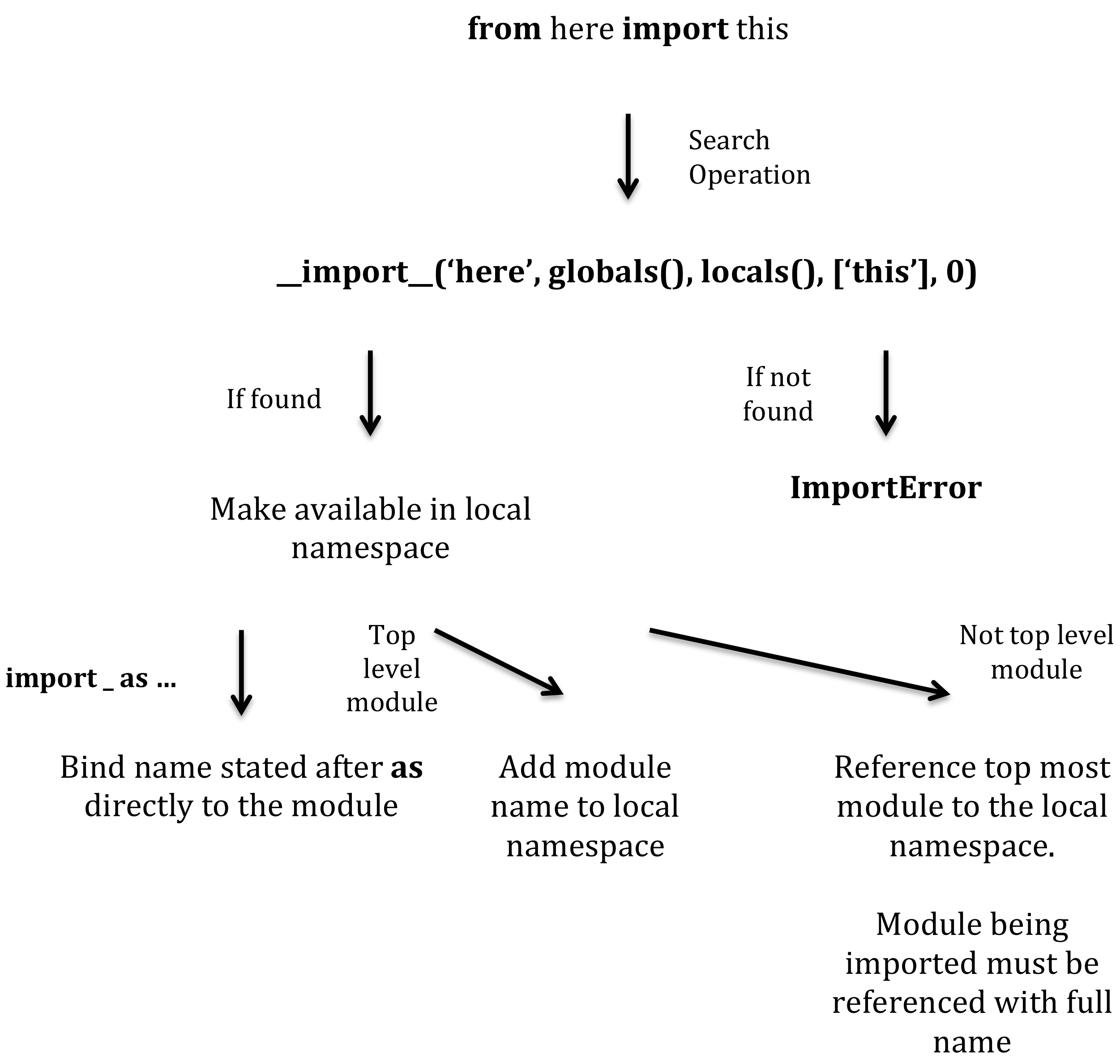
Operating Systems

Python Import

How Python Import Works

Using the **import** statement does two things, searches for the module and if found it binds the module to a name in the local scope and initializes it if necessary. The way it searches for the module is that it calls the **__import__()** function (look at the diagram below). This is called the finder who's job it is to find the module being searched for. Where the finder searches is within the module cache, which is cached in **sys.modules** in the form of a dictionary. If it finds a module that matches the name, it will then create a module object. If the module is not found then an **ImportError** exception is raised.

Once the module is found, it will be made available to the local namespace for it to be used further in the code. This can be done in one of 3 ways. The first being if **as** is stated after **import** then it will bind the name stated after **as** to the module. Another way it does this is that if there is no other name given, then it will add the name to the local namespace if it is the top level module. Otherwise if it is not a top level module, then the name of the top most module is referenced in the local namespace and the module being imported must be referenced using the full name.



from here **import** this

Search Operation

__import__('here', globals(), locals(), ['this'], 0)

If found

Make available in local namespace

If not found

ImportError

import _ as …

Bind name stated after **as** directly to the module

Top level module

Add module name to local namespace

Not top level module

Reference top most module to the local namespace.

Module being imported must be referenced with full name