Import From Source File

The importlib's util sub-module has another neat trick that I want to cover. You can use **util** to import a module using just it's name and file path. The following is a very derived example, but I think it will get the point across:

```
import importlib.util

def import_source(module_name):
    module_file_path = module_name.__file__
    module_name = module_name.__name__

    module_spec = importlib.util.spec_from_file_location(
        module_name, module_file_path)
    module = importlib.util.module_from_spec(module_spec)
    module_spec.loader.exec_module(module)
    print(dir(module))

    msg = 'The {module_name} module has the following methods:' \
        ' {methods}'
    print(msg.format(module_name=module_name,
                     methods=dir(module)))

if __name__ == '__main__':
    import logging
    import_source(logging)
```

In the code above, we actually import the **logging** module and pass it to our **import_source** function. Once there, we grab the module's actual path and its name. Then we call pass those pieces of information into the util's **spec_from_file_location** function which will return the module's specification. Once we have that, we can use the same importlib mechanisms that we used in the previous section to actually import the module.

Now let's look at a neat 3rd party module that Python's **__import__()** function to import packages directly from github!