The Header File *Python.h*

You need to include *Python.h* header file in your C source file, which gives you access to the internal Python API used to hook your module into the interpreter.

Make sure to include Python.h before any other headers you might need. You need to follow the includes with the functions you want to call from Python.

The C Functions

The signatures of the C implementation of your functions always takes one of the following three forms:

```
static PyObject *MyFunction( PyObject *self, PyObject *args );

static PyObject *MyFunctionWithKeywords(PyObject *self,
                                        PyObject *args,
                                        PyObject *kw);


static PyObject *MyFunctionWithNoArgs( PyObject *self );
```

Each one of the preceding declarations returns a Python object. There is no such thing as a*void* function in Python as there is in C. If you do not want your functions to return a value, return the C equivalent of Python's **None** value. The Python headers define a macro, Py_RETURN_NONE, that does this for us.

The names of your C functions can be whatever you like as they are never seen outside of the extension module. They are defined as *static* function.

Your C functions usually are named by combining the Python module and function names together, as shown here:

```
static PyObject *module_func(PyObject *self, PyObject *args) {

   /* Do your stuff here. */

   Py_RETURN_NONE;

}
```

This is a Python function called *func* inside of the module *module*. You will be putting pointers to your C functions into the method table for the module that usually comes next in your source code.