This is the same as the simple_server example, but it uses a *handler* interface, rather than a *server* interface. The following files are used in the implementation:

simple.c	· This file uses the MIG generated interface to the simple server. It obtains access to the server by looking up the port advertised on its behalf by the kernel loader.
simple.defs	· This file implements the MIG interface to the server. It generates simpleServer.c, simpleUser.c, and simple.h.
simple_handler.c	· This file is constructed as an interface to *simpleServer.c*. It presents an interface based on message/argument, rather than inmsg, outmsg. It is generated by hand by inspecting *simpleServer.c*.
simple_handler.h	· Constructed, as is *simple_handler.c*, as an interface to the MIG generated *simpleServer.c* events are received for this server.

simple_server.c · This implements the guts of the server running in the kernel. It is called from the (linked in, or resident in the kernel) kernel server interface routines when messages or other events are received for this server.

Load_Commands.sect · This file specifies the actions to be taken when loading the server. This includes allocating and advertising a port, and setting up a mapping to call into the loaded code when a message is received on that port.

Unload_Commands.sect · This file specifies the actions to be taken when unloading the server.

To load this into the kernel loader type the following:

```
kl_util -a simple_reloc
```

This will cause the server to be initialized by the kernel loader. Running the program

simple will then invoke the server, causing it to be loaded into the kernel.