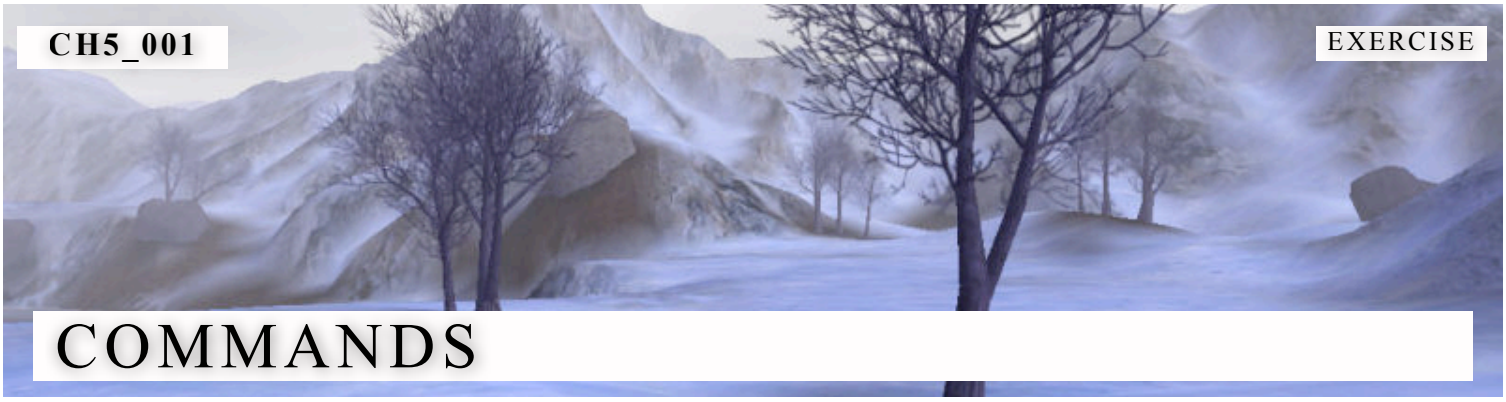

CH5_001

EXERCISE

COMMANDS

Exercise Files

Starter – "Kit/gpgt/server/scripts/gpgt/chapter5/exercise001.cs"
Answers – "Kit/gpgt/server/scripts/gpgt/chapter5/answers/exercise001_f.cs"

Exercise Mission

Chapter 5: "001_Communications: Commands"

Synopsis

In this exercise, we will write a few different commands re-using and expanding upon the features we used in the chapter 3 exercises.

Prerequisites

1. *ch1_001.pdf "Using The Kit"*
2. *ch3_003.pdf "Ghost Resolution"*
3. *ch3_005.pdf "Screen Blackout"*

Exercises

COMMANDS

1 Black Out (all) Clients' Screens

Goal: Re-use the blackout code we wrote in chapter 3 and to expand this code to allow the server to black out (and fade in) all clients' screens using one command.

Starter Code: You are provided with 2 function bodies to start this exercise.

```
// 1
function fadeScreenOutandIn( %start, %outTime, %waitTime, %inTime )
{
   // FUNCTION BODY?
}

// 2
function FUNCTION_NAME?( arguments? )
{
}
```

Steps:

1. Modify the first function fadeScreenOutandIn() to iterate over all clients and to send them a command with these arguments:

   'fadeScreenOutandIn' – Tag including the name of the client command to execute.

   %start, %outTime, %waitTime, %inTime – The four time arguments. These serve the same purpose they did in the chapter 3 screen blackout exercise (ch3_005.pdf)

2. Modify the name and argument list of second function to properly match the function name as it will be called on the client in response to the commandToClient() call from the first function.
3. Modify the body of the second function to fade the screen out and then in, using the supplied arguments.

Output Goal:

When fadeScreenOutandIn() is called, all clients' screens should fade to black, stay black, and the fade back in, based on the time values you pass to the function.

Hints:

None.

2 Resolve Client's Control Object Ghost

Goal: Write a set of commands that enable a client tell the server to send back the client's control object ghost index and the server object ID for that index.

Starter Code: You are provided with 3 function bodies to start this exercise.

```
// 1
function doRemoteGhostResolution()
{
   commandToServer( 'sendControlObjectGhostData' );
}

// 2
function clientCmdReceiveControlObjectGhostData( %ghostIndex ,  %serverObjectID )
{
}


// 3
function FUNCTION_NAME?( %clientConn )
{
}
```

Steps:

1. You need to modify the name of the third function to appropriately match the function name as it will be called on the server in response to the commandToServer() call in the first function.
2. Once the third function is properly named, please add code in the body of this function to do the following:

   Get calling client's control object ID.

   Get ghost index for this control object ID.

   Issue a client command that will call the second (provided) function: clientCmdReceiveControlObjectGhostData(%ghostIndex , %serverObjectID).
3. Implement the body of the second (provided) function and have it do the following:

   Convert the passed ghost index into a ghost object ID.

   Print out the ghost index, ghost ID, and server object ID information for this client's control object (See output goals below for more help).

COMMANDS

Output Goal:

When you call the function doRemoteGhostResolution(), you should see something like this in the console:

```
serverCmdSendControlObjectGhostData(1574)
clientCmdReceiveControlObjectGhostData()
This client's control object ghost index is 1
   This client's control object ghost ID is 1624
  This client's control server object ID is 1623
```

Hints:

1. When you call the getControlObject() on a client connection object, it returns the server object ID of the control object.

