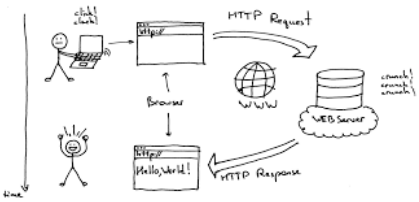
Worksheet 2: Networking with Unreal and HTTP Servers

Version 1.0
Creative Computing
COMP280

Gareth Lewis



The problem with troubleshooting is that trouble shoots back.

-Author Unknown

A computer lets you make more mistakes faster than any invention in human history - with the possible exceptions of handguns and tequila.

-Mitch Ratcliffe

Introduction

For this worksheet, you are going to develop HTTPServer applications that can be integrated into Unreal Engine, allowing you store persistent game data on a server such that it can be shared between multiple users. This is an ideal architecture for managing shared data systems such as player accounts, high score tables, portable players progression data (to allow player progress to be maintained across multiple machines) and other player and game related data.

To complete this worksheet:

a) **Create** a python-based HTTP Client and Server that will allow users to issue GET and POST commands to the server and for the server to respond appropriately.
b) **Use** JSON as a mechanism to allow more complex data to be sent between client and server applications
c) **Create** an SQL testbed that will allow users to experiment with the data lifecycle to create, retrieve, update delete and store (CRUDS) records, tables and databases.
d) From the testbeds, **create** a python-based HTTP Server application that will successfully communicate with an Unreal client to provide functionality that relies on persistent data that is stored on the server in an SQL database.

Additional Guidance

Creating an HTTPServer for the Unreal Engine sounds like a big task that is both hard and complex. In reality it is actually comprised of several hard and complex tasks that are fairly small, and we will discover that by breaking large tasks down into smaller tasks they become soluble.
Much of the work in this worksheet is geared around proving concepts through the use of sandbox code to break a large problem into a many smaller problems which can all be solved in isolation. Once these smaller problems have been solved, the knowledge gained from solving them can be used to create the required large and complex solution.

As a pair of programmers, you have the choice to solve problems through pair programming (with a driver and a navigator) or to break into sub teams to solve individual problems and report back to each other. The approaches you use are likely to depend on the nature of the problems you are looking to solve and how you want to work with each other. Don't forget, if the worst comes to the worst, you can still ask for help.

Marking Rubric

Learning Outcome Name	Learning Outcome Description	Criteria	Weighting	Clear Fail	Near Pass	3rd	2:2	2:1	1st	>1st
Code / Process	Implement working and maintainable software components.	Prototype Python client/server	40%	No submission	Client & server applications can't communicate	Client & server applications can't communicate reliably	Client & server applications communicate reliably	Client & server applications communicate reliably Evidence of JSON data format	Client & server applications communicate reliably Evidence of complex JSON data format(s)	Client & server applications communicate reliably with multiple message types Evidence of complex JSON data format(s)
		SQL Testbed	30%	No submission	Application can't CRUD database	Application can't reliably CRUD database	Application reliably CRUDs database	Application reliably CRUDs database Database is stored on server, interface on client	Application reliably CRUDs database Database is stored on server, interface on client Some consideration given to PyQt for interface	Application reliably CRUDs database Database is stored on server, interface on client Much consideration given to PyQt for interface
		Unreal Demo with persistent data storage	30%	No submission	Code presented but doesn't appear to work	Code presented but doesn't appear to work reliably	Unreal client and Python server work reliably. Application uses persistent data in 'simple' manner, e.g. single high score	Unreal client and Python server work reliably. Application uses persistent data in a more complex manner, e.g. all high scores	Unreal client and Python server work reliably. Application uses persistent data in fundamental game manner, e.g. game balance data or analytical data	Unreal client and Python server work reliably. Application uses a combination of persistent data types, e.g. all high scores, balance data, analytical data