Project #2c – HTTP-based Cross-Platform GUI

due last day of class

Purpose:

This project develops a cross-platform GUI that can run on either Windows or Linux and can be used remotely from either Windows or Linux platforms. The purpose of this project is to design an HTML5 based facility running in the Chrome web browser to display a user interface and respond to user actions. The GUI interacts with the rest of the application through a tiny HTTP server added to the application that supports Ajax-based communication between browser and application. This means that the application will use common code for everything except the low-level interface with the Windows or Linux operating system.

Requirements:

Your HTML5GUI project:

1. **shall** use HTML5, JavaScript, and standard C++ and the standard library, compile and link from the command line, using g++ within the Eclipse or NetBeans IDE and Visual C++ in the Visual Studio IDE.

2. **shall** provide an application that runs on both Windows and Linux that provides a graphical user interface (GUI) based on HTML5 viewed in the Chrome web browser. This should work for both Windows and Linux.

3. **shall** add the smallest feasible HTTP server to the executive part of the application code that is capable of serving HTML5 pages with associated Cascading Style Sheets and JavaScript files to provide the GUI view[1].

4. **shall** use Ajax protocol[2] to communicate between the GUI and application code.

5. The application **shall** provide at least two views implemented as a tabbed interface and each should do something interesting. You might consider using the core of Project #1 as the basis for your application.

A major part of the credit you earn for this project is based on the effectiveness and esthetics of your user interface.

You will find the following reference very useful in understanding how the HTTP protocol works:
http://www.jmarshall.com/easy/http/

---

[1] Note that this implies you will use sockets technology for this project.

[2] This will require you to use Wireshark on Linux or Fiddler on Windows to figure out how your HTTP server has to respond to an Ajax request by looking at the contents of HTTP messages. I will hand out a demonstration of the browser code necessary to send an Ajax request.