KoalaTweet

Seb *pahl_s@epitech.eu*

January 7, 2009

Contents

1 Introduction;-)

1.1 HTTP

Hypertext Transfer Protocol (HTTP) is a communications protocol. Its use for retrieving inter-linked text documents (hypertext) led to the establishment of the World Wide Web.

HTTP development was coordinated by the World Wide Web Consortium and the Internet Engineering Task Force (IETF), culminating in the publication of a series of Requests for Comments (RFCs), most notably RFC 2616 (June 1999), which defines HTTP/1.1, the version of HTTP in common use.

HTTP is a request/response standard between a client and a server. A client is the end-user, the server is the web site. The client making a HTTP requestusing a web browser, spider, or other end-user toolis referred to as the user agent. The responding serverwhich stores or creates resources such as HTML files and imagesis called the origin server. In between the user agent and origin server may be several intermediaries, such as proxies, gateways, and tunnels. HTTP is not constrained to using TCP/IP and its supporting layers, although this is its most popular application on the Internet. Indeed HTTP can be "implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used."

Typically, an HTTP client initiates a request. It establishes a Transmission Control Protocol (TCP) connection to a particular port on a host (port 80 by default; see List of TCP and UDP port numbers). An HTTP server listening on that port waits for the client to send a request message. Upon receiving the request, the server sends back a status line, such as "HTTP/1.1 200 OK", and a message of its own, the body of which is perhaps the requested resource, an error message, or some other information.

HTTP predominantly uses TCP and not UDP because much data must be sent for a webpage, and TCP provides transmission control, presents the data in order, and provides error correction. See the difference between TCP and UDP.

Resources to be accessed by HTTP are identified using Uniform Resource Identifiers (URIs) (or, more specifically, Uniform Resource Locators (URLs)) using the http: or https URI schemes.

This introduction was taken from `http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol`.
For more information on HTTP read the RFC 2616 (`http://www.ietf.org/rfc/rfc2616.txt`)

1.2 REST

Representational state transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web. As such, it is not strictly a method for building "web services." The terms representational state transfer and REST were introduced in 2000 in the doctoral dissertation of Roy Fielding,[1] one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification.

REST strictly refers to a collection of network architecture principles which outline how resources are defined and addressed. The term is often used in a looser sense to describe any simple interface which transmits domain-specific data over HTTP without an additional messaging layer such as SOAP or session tracking via HTTP cookies. These two meanings can conflict as well as overlap. It is possible to design a software system in accordance with Fieldings REST architectural style without using HTTP and without interacting with the World Wide Web.[2] It is also possible to design simple XML+HTTP interfaces which do not conform to REST principles, and instead follow a model of remote procedure call. The difference between the uses of the term REST therefore causes some confusion in technical discussions.

Systems which follow Fieldings REST principles are often referred to as RESTful.

This introduction was taken from `http://en.wikipedia.org/wiki/Representational_State_Transfer`.
Recommended material:

- `http://rest.blueoxen.net/cgi-bin/wiki.pl`
- `http://www.infoq.com/articles/rest-introduction`

1.3 JSON

JSON, short for JavaScript Object Notation, is a lightweight computer data interchange format. It is a text-based, human-readable format for representing simple data structures and associative arrays (called objects).

The JSON format is often used for transmitting structured data over a network connection in a process called serialization. Its main application is in Ajax web application programming, where it serves as an alternative to the use of the XML format.

Although JSON was based on a subset of the JavaScript programming language (specifically, Standard ECMA-262 3rd EditionDecember 1999) and is commonly used with that language, it is considered to be a language-independent data format. Code for parsing and generating JSON data is readily available for a large variety of programming languages. The json.org website provides a comprehensive listing of existing JSON bindings, organized by language.

In December 2005, Yahoo! began offering some of its web services optionally in JSON. Google started offering JSON feeds for its GData web protocol in December 2006.

This introduction was taken from `http://en.wikipedia.org/wiki/JSON`. For more information on JSON go to `http://www.json.org/`

2 Subject

2.1 Goal

Now that you know what HTTP/REST/JSON are, you are ready to read your assignement.

The goal is to create a (almost) complete client for Twitter `http://twitter.com`.

Twitter is a free social networking and micro-blogging service that allows its users to send and read other users' updates (otherwise known as tweets), which are text-based posts of up to 140 characters in length.

With your client you have to be able to receive your friends tweets and update your own status.

2.2 The steps you must follow to perform

- Read the Twitter REST API (`http://apiwiki.twitter.com/` and especially `http://apiwiki.twitter.com/REST+API+Documentation`).
- Code a library capable of talking to a HTTP server.
- Code a library capable of talking to the API using your HTTP library. The Twitter API support multiple formats. You HAVE to use JSON.
- Code a nice graphical user interface that uses your Twitter library.

3 Requirements

3.1 Platform

The project has to be OS independent. It has to run on at least one Linux/Unix system and one Windows system.

3.2 Languages

Only C++ is allowed. C and C+ will not be tolerated.

3.3 Libraries

- You are NOT allowed to use any HTTP or JSON library.
- You HAVE to use QT (`http://trolltech.com/`) for the graphical interface.
- You are allowed and encouraged to use BOOST C++ libraries (`http://www.boost.org`).
- POCO C++ libraries (`http://pocoproject.org/`) are very cool but NOT allowed.
- For requests, send a mail to the author. The subject will be updated.

Tip: Have a look at BOOST Asio (`http://www.boost.org/doc/libs/1_37_0/doc/html/boost_asio.html`) for networking.

3.4 Versioning

You HAVE to use code versioning during the whole project. Otherwise the mark will be 0. We will check for an accurate timeline.

We don't care what you use. Here are some suggestions:

- Subversion (`http://subversion.tigris.org/`)
- Git (`http://git-scm.com/`)
- Mercurial (`http://www.selenic.com/mercurial/`)

4 Rating

4.1 Mandatory

4.1.1 HTTP

HTTP Client library

5 Points

4.1.2 REST API

Twitter API library which uses the HTTP library.

5 Points

4.1.3 GUI

GUI that uses the Twitter library.

5 Points

4.2 Bonus

4.2.1 Spirit

Use BOOST Spirit (`http://www.boost.org/doc/libs/1_37_0/libs/spirit/classic/index.html`) to parse HTTP and JSON.

5 points

4.2.2 Plugin system

Create a plugin system for the GUI using BOOST Python. Add a plugin written in Python (`http://www.python.com`) for VieDeMerde (`http://www.viedemerde.fr/api/documentation`).

5 points

4.2.3 Google Maps

Look a the location of a twitter user and embed google maps (`http://code.google.com/apis/maps/`) in your GUI to pinpoint the location of a tweet.

5 points

5 Advice

5.1 School Proxy

Do not forget that we have a HTTP/HTTPS proxy in the school.

In order to get out and talk to Twitter you need to talk to the proxy. Don't forget that it uses HTTP basic authentication.

You are allowed to use a proxyfier like Proxychains (`http://proxychains.sourceforge.net/`) or Freecap (`http://www.freecap.ru/`).
You can also tunnel out with a VPN or using SSH.
Maybe you will get a bonus if you code yourself out.:-)

5.2 Misc

- This assignement is for 4 students, so split the work!
- Koalas have set up an irc chan available to the students. the server is 'irc.epitech.net', chan 'koala'. You're welcome anytime just to say hello or ask any question.
- Also, you can use the C++ Epitech newsgroup. It's up and ready
- Keep an eye on this subject, it may change during the next days.