Basics of the HTTP Protocol
At first there was HTTP 0.9

This is as simple as it can get

GET http://www.somedomain.com/hello.txt

Hello

- Created by Tim Berners-Lee in 1989(?)
- The 0.9 version number was actually created after the 1.0 spec
HTTP 1.0

- The first really practical revision of the HTTP protocol
- HTTP Request Headers and Response Headers
- Simple caching
- Authentication
- Content-Type
- Sending data via POST
- HTTP Status codes (200, 404, etc)
HTTP 1.1 (in use today)

- Includes everything from HTTP 1.0
- Host header is required
- Defines more status codes, more request methods
- Much more flexible caching available
- Digest Authentication
Sample HTTP Request / Response

GET / HTTP/1.1
Host: achtung.ccs.neu.edu

HTTP/1.1 200 OK
Date: Wed, 11 Apr 2012 12:19:59 GMT
Server: Apache/2.2.17 (Ubuntu)
Last-Modified: Tue, 05 Jan 2010 23:17:38 GMT
ETag: "1071-b1-47c730b3f84ad"
Accept-Ranges: bytes
Content-Length: 177
Vary: Accept-Encoding
Content-Type: text/html

<html><body><h1>It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
</body></html>
Headers of Interest

- **Referer**
  - Says which page referred you to the current URL
  - Note the misspelling
  - Used in Analytics to provide a lot of useful metrics

- **User Agent**
  - Specifies OS and Browser (often faked)

- **Cookie / Set-Cookie** (more on this later)
HTTP is stateless protocol

A stateless protocol does not require the server to retain information or status about each user for the duration of multiple requests.

Solution

- Encode in URI
  - /index.php?session_id=some_unique_session_code

- Cookie
  - Set-Cookie: datr=QPmGTzoik4GDe1X15pBAf2J4; expires=Sat, 12-Apr-2014 15:48:16 GMT; path=/; domain=.facebook.com; httponly
HTTP Cookies

- Cookies are generally good! They provide some incredibly useful functionality.
  - Server sends a Set-Cookie
  - Client sends back a Cookie

- Demonstrate a cookie
  - http://web01.roundsphere.com/cookie_test.php

- Be careful what you put in a cookie!
  - Don’t store user ID’s, authentication credentials, etc
Using Cookies to create sessions

- Without cookies, all HTTP requests are completely independent
- Cookies allow the server to add some persistence to multiple requests and create a session
- Most programming languages have some built-in support for sessions. (PHPSESSID, JSESSIONID, etc)
- Session information can be stored in file system, database, memcache, etc.
- Don’t pass Session ID through GET requests
- Demo some simple session examples:
  - http://web01.roundsphere.com/session_test.php