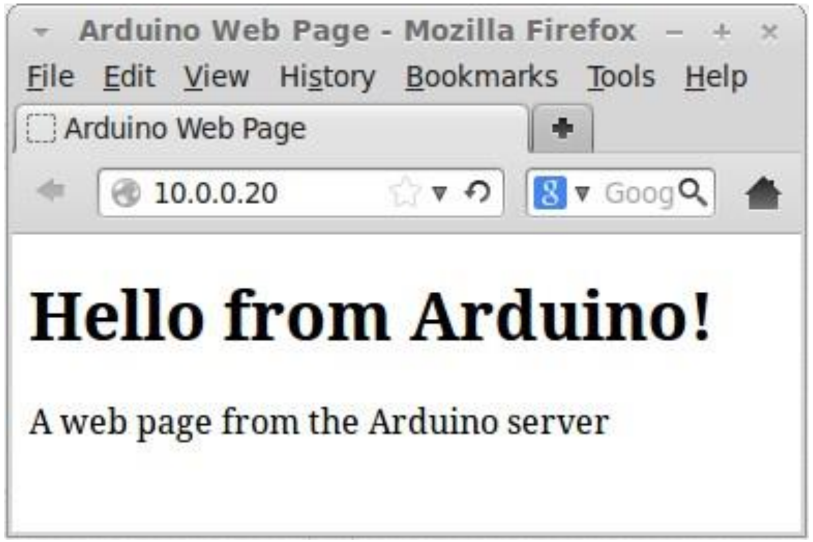
Web Servers for IoT

Frank Walsh


Arduino Web Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Arduino Web Page
10.0.0.20
Goog

Hello from Arduino!

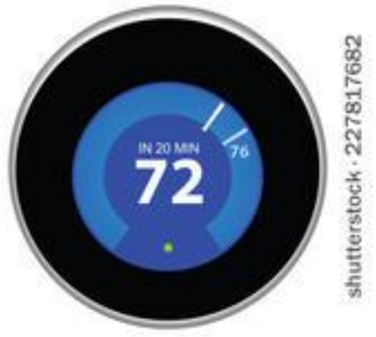
A web page from the Arduino server

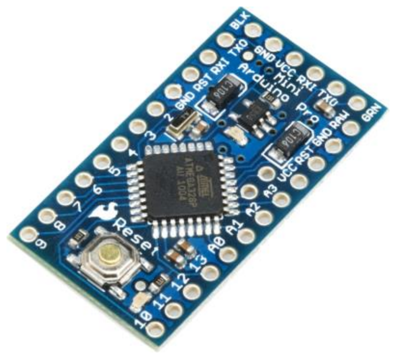
Agenda

- Web Servers
- HTTP
- Web Servers on Devices
- Web API

What's a Web Server

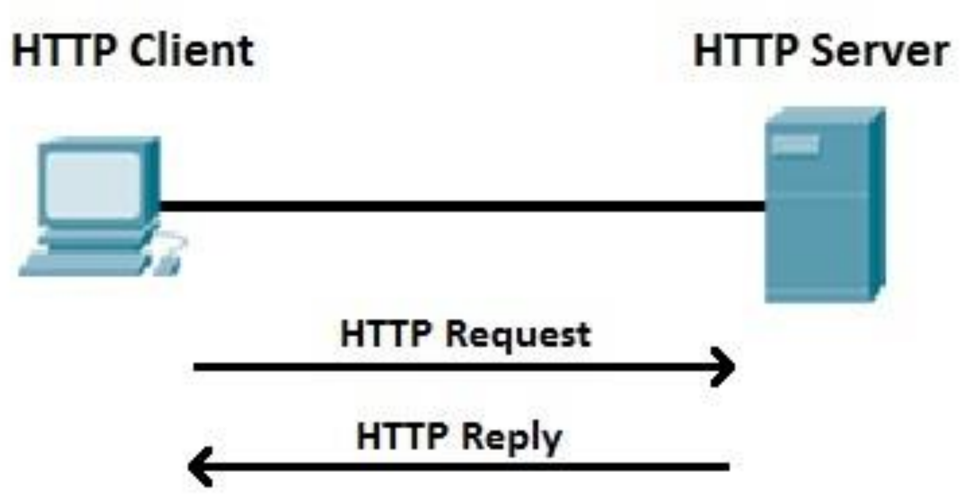
- Implements HTTP
  - processes HTTP requests
- Usually runs on machine connected to a network
  - Has an IP address
- Serves up Web Pages
  - But can do more in IoT...
- Many different types of machines can run a web server...





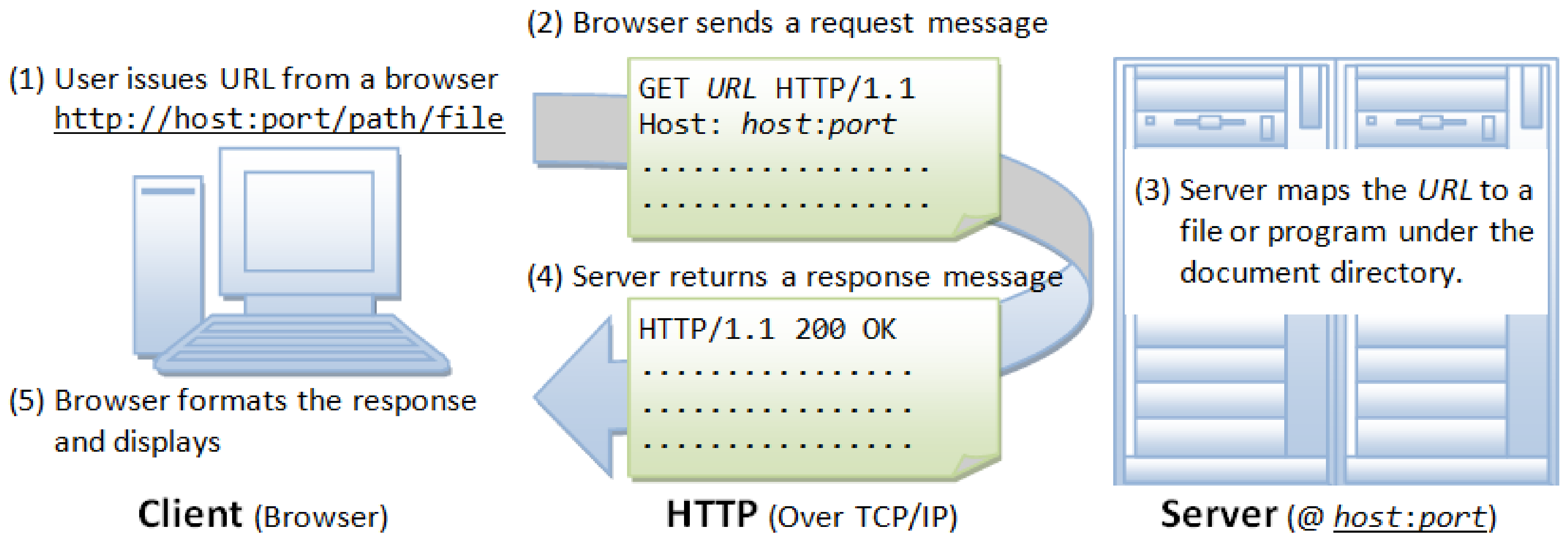
What's HTTP

- HyperText Transfer Protocol
- Protocol used in World Wide Web
  - http://www.wit.ie
- Your browser communicates using HTTP (HTTP Client)
  - Transfers HTML
- Devices communicate using HTTP
- Simple, ubiquitous.



HTTP

Browser:


(1) User issues URL from a browser
http://host:port/path/file
(2) Browser sends a request message
GET *URL* HTTP/1.1
Host: *host:port*
....
....
(3) Server maps the *URL* to a file or program under the document directory.
(4) Server returns a response message
HTTP/1.1 200 OK
....
....
....
(5) Browser formats the response and displays
Client (Browser)
HTTP (Over TCP/IP)
Server (@ *host:port*)

URL

- A URL (Uniform Resource Locator) uniquely identifies a resource over the web. *protocol://hostname:port/path-and-file-name*
- *There* are 4 parts in a URL:
  - *Protocol*: The application-level protocol used by the client and server, e.g., HTTP, FTP, and telnet.
  - *Hostname*: The DNS domain name (e.g., www.nowhere123.com) or IP address (e.g., 192.128.1.2) of the server.
  - *Port*: The TCP port number that the server is listening for incoming requests from the clients.
  - *Path-and-file-name*: The name and location of the requested resource, under the server document base directory.
- Example, for http://www.nowhere123.com/docs/index.html
  - the communication protocol is HTTP t
  - he host is www.nowhere123.com.
  - The port number was not specified, and takes default number, which is TCP port 80 for HTTP.
  - The path and file name for the resource to be located is "/docs/index.html".

HTTP Protocol (Request)

- HTTP clients (e.g. a browser) translates a URL into a request message according to the specified protocol; and sends the request message to the server.
- For example, the browser translated the URL http://www.nowhere123.com/doc/index.html into the following request message:

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

HTTP Protocol (Response)

- When this request message reaches the server, the server can take either one of these actions:
  1. The server interprets the request received, maps the request into a file under the server's document directory, and returns the file requested to the client.
  2. The server interprets the request received, maps the request into a program kept in the server, executes the program, and returns the output of the program to the client.
  3. The request cannot be satisfied, the server returns an error message.

An example of the HTTP response message is below:

```
HTTP/1.1 200 OK
Date: Sun, 18 Oct 2009 08:56:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Sat, 20 Nov 2004 07:16:26 GMT
Content-Length: 44
Connection: close
Content-Type: text/html

 <html><body><h1>It works!</h1></body></html>
```

HTTP Query String

- Query string used to include data in a URL. For example

Query String

http://www.myhome.com/heating?status=on

- The server can use the query string to execute logic associated with that resource. In this example, it could be used to set the status of the resource (heating) to true.

HTTP Methods

- GET
  - Request objects without sending data
- POST
  - Modify objects with data that you are sending
- PUT
  - Create new objects with data that your are sending
- DELETE
  - Delete objects without sending data

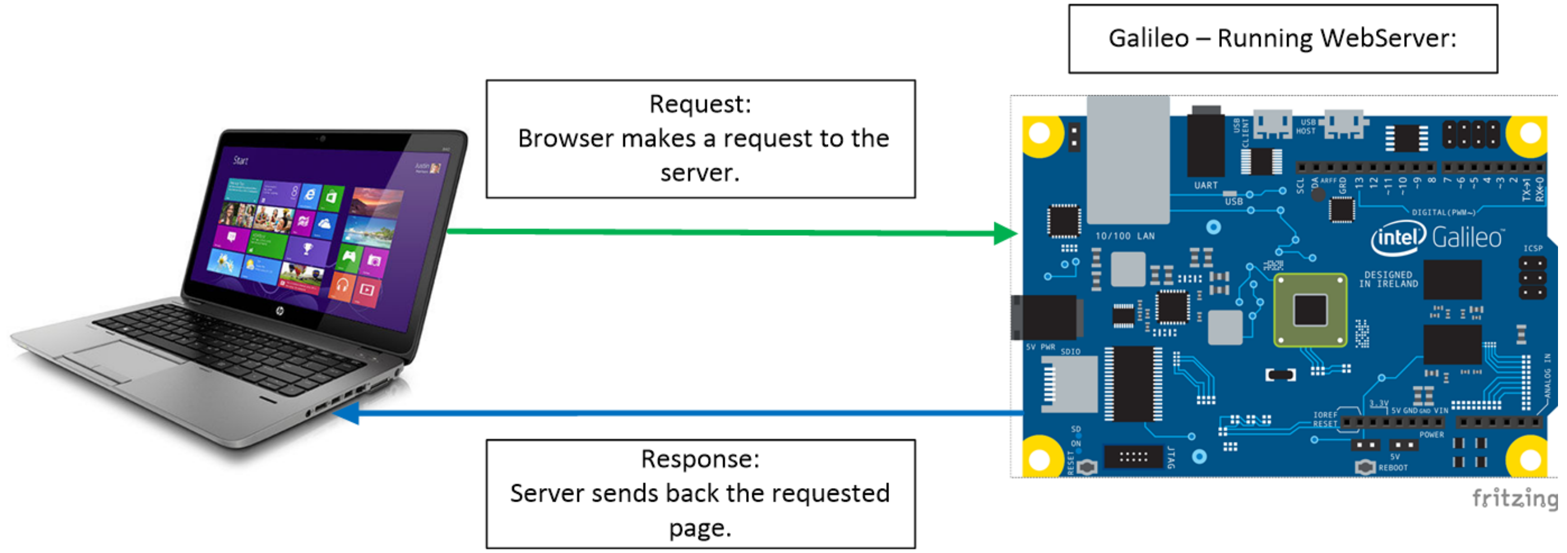
More on HTTP

- For a excellent overview, checkout:

https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html

HTTP Server on IoT devices

- Set up a Web server on Galileo:
  - Connects sensors/actuators to web
  - Access and Control your devices via the Web:
    - Web application program interface(Web API)



Galileo – Running WebServer:

Request:
Browser makes a request to the server.

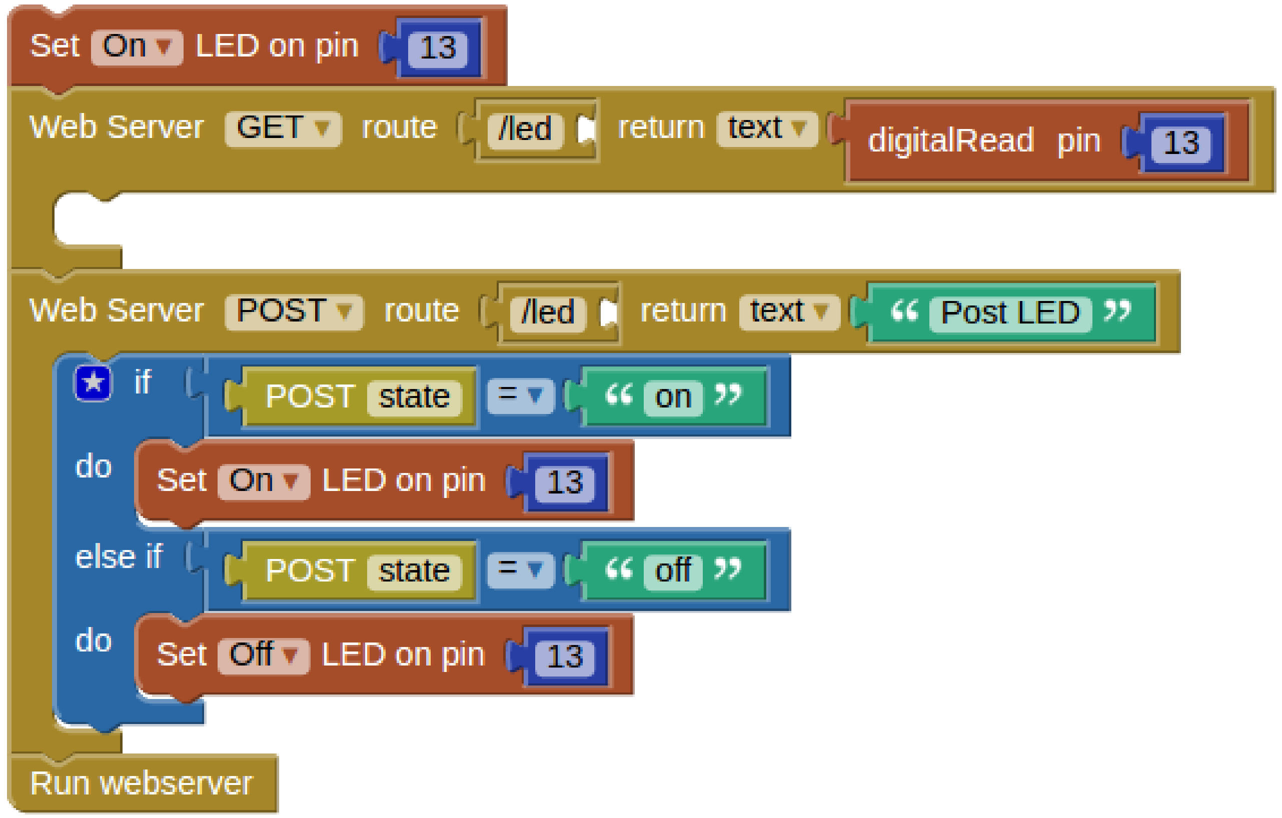
Response:
Server sends back the requested page.

Demo

Wyliodrin
IDEAS IN REAL TIME
Program Raspberry Pi Lesson 1: Switch a LED on and off by using a web browser

A More Correct Solution...

- Use HTTP and URLs properly...



```
Set On LED on pin 13
Web Server GET route /led return text digitalRead pin 13

Web Server POST route /led return text " Post LED "
    if POST state = " on "
    do Set On LED on pin 13
    else if POST state = " off "
    do Set Off LED on pin 13
Run webserver
```