POWERING POTENTIAL

HTML

Introduction

LEARNING OBJECTIVES

How The Web Works

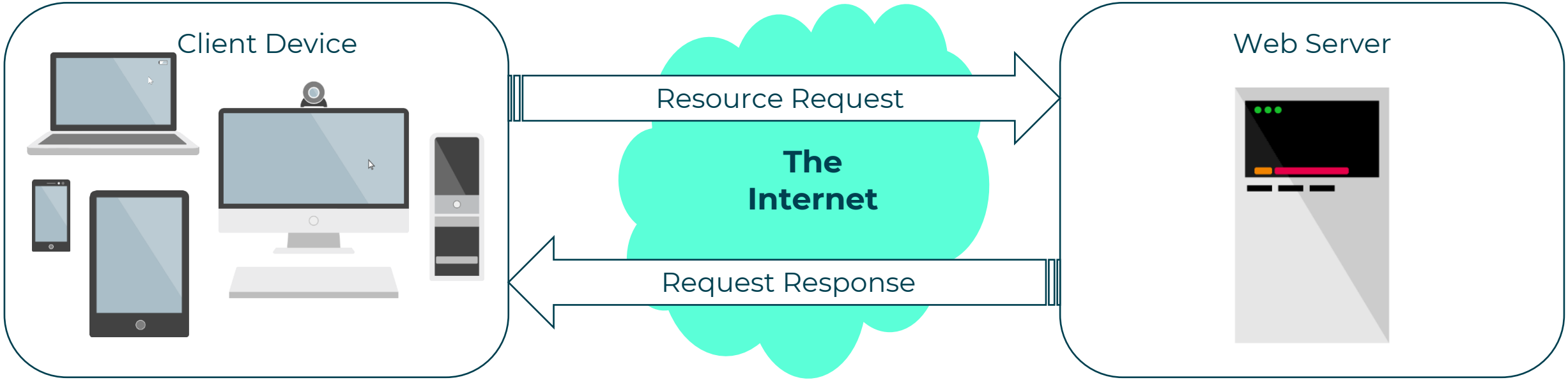
Basic HTML

- HTML History and Syntax
- Structural HTML
- Hyperlinks
- Lists
- Tables
- Forms
- The <head> tag
- The DOM

How the Web Works

- Clients and Servers
- URLs
- HTTP, HTTPS, and SSL

How the web works...



Client Devices

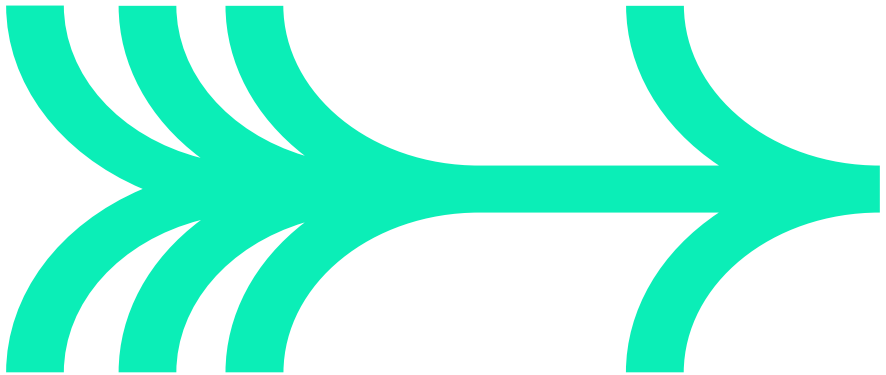
Need some form of browser to make requests.

Most commonly:

- Chrome, Safari, Edge, etc.

Can also be

- Smart Devices (televisions, home appliances, etc)



- Makes request using a Uniform Resource Locator (URL) to specify where request is made to
- Uses HyperText Transfer Protocol (HTTP) to make the request

Web Servers

Needs to be running some Web Server software

Most commonly:

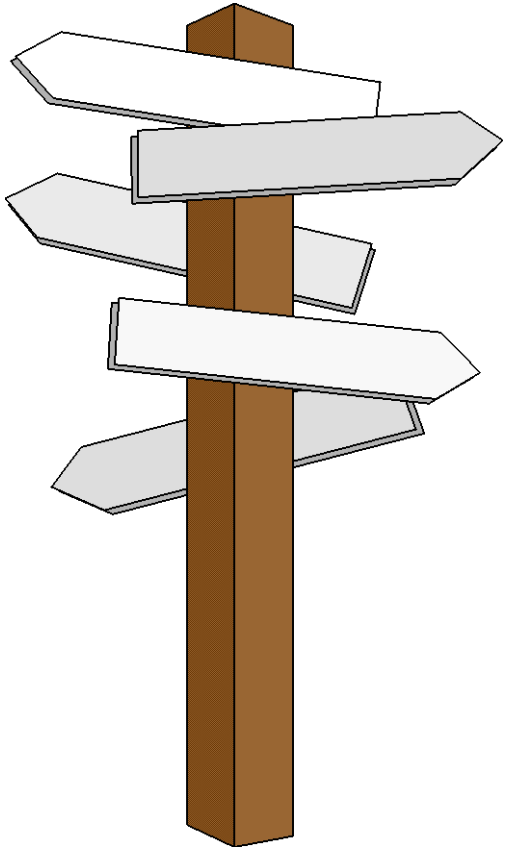
- Apache
- Nginx
- Microsoft Internet Information Server (IIS)

- Handles HTTP requests
- Dispatches response to requests
- Are addressed by URLs converted to IP addresses by Domain Name Servers (DNS)

Introduction to URLs

Uniform Resource Locators (URL)

- Identifies location and protocol to access a resource
- URLs are a form of Uniform Resource Identifier (URI)



URL Syntax

Scheme (How)

Path (Where: Scheme-specific format)

Resource (What)

```
http://www.microsoft.com/ie/page.html

ftp://ftp.demon.co.uk

news:comp.infosys.www.announce

mailto:billg@microsoft.com

www.bbc.co.uk
```

This is *not* a valid URL, but many browsers accept it as equivalent to **`http://www.bbc.co.uk`**

HTTP URL Format

Host name

Port number. Defaults to 80.

```
http://www.microsoft.com/ie

http://www.microsoft.com:80/ie

http://www.carousel.org/galloper/april.html

http://www.qa.com
```

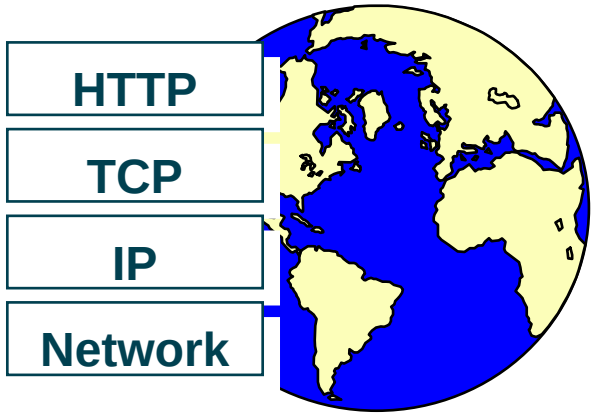
Document path. May be case-sensitive

An empty path refers to the home page
(Server-dependent: May be **`index.html`** or **`default.html`**)

HyperText Transfer Protocol (HTTP)

Application-Level Protocol

- Technical information at http://www.w3.org
- TCP-based
- Current version is 1.1



Lightweight

- Easy to implement clients and servers
- Stateless: Each request is independent of the others
  - Other technologies required in order to enable e-commerce, online banking, etc.

Request/response paradigm

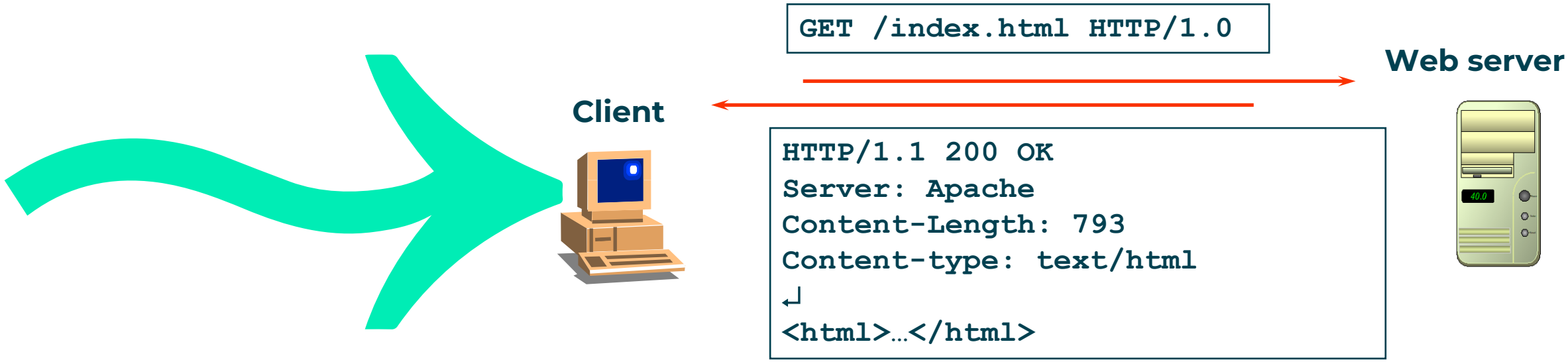
HTTP Interactions

Client Request:

- Method, Resource, HTTP version
- MIME type header and message

Server Response:

- HTTP version and standard response code
- MIME type header and message



HTTP CLIENT REQUEST

Method

- Action to perform on resource - GET, HEAD, POST

Uniform Resource Identifier

- Identifies a networked resource
- Absolute URI used with a proxy server
- Request URI used with an origin server

HTTP Version

- Major.minor version - Default (no version given) is 0.9
- Version 1.1 now the most popular
- Browsers and Servers must also understand both 0.9 and 1.0

MIME-like message - Contains request modifiers and forms data

HTTP Server Response

Simple Response/Full Response

Status line

- HTTP version
- Standard status code
- Reason phrase

MIME like message

- Generated by web server or by backend script
- Header fields describe the requested resource
- Modified using HTML <meta> tag
- Requested data
- Header and Data are separated by CRLF pair

MIME AND HTTP

Multipurpose Internet Mail Extensions

- Based on Internet Mail (RFC 822)
- MIME is defined in RFC 1521
- HTTP usage differs from RFC 1521

Transmission of Multimedia Objects over Internet

- Header consists of colon-separated fields
- Data contains requested object
- Content-Type field describes object

Object Types

- Defined by IANA (Internet Assigned Numbers Authority)
- Consist of type/subtype
- Unofficial types preceded by x- (x-world/x-vrml)

Multipart Messages

- Multiple MIME messages each containing a header specifying the type of body data

SECURITY ISSUES

Preventing Eavesdropping:

- Use of encryption

Preventing Modification/Fabrication:

- Authenticating Messages

Preventing Impersonation:

- Authenticating clients and servers

Security - HTTPS

Essentially works in same way as HTTP

- Uses Secure Socket Layer (SSL) to encrypt data being passed

Lots of websites and development libraries and frameworks will work with or require HTTPS.

It stops ‘eavesdropping’ on data transfers between client and server.

Security - SSL

- Secure protocol for sending information securely over the Internet
- Encrypts data transmitted between client and server
- Mostly uses 128-bit encryption
- Requires web server to hold a valid certificate