CS603 – Web Engineering

PREPARED BY: DR. REEMA PATEL

Introduction

- 3 Lectures per week
- 2 Hours Lab per week

- Internal – 20 Marks –Quiz, Assignments
- Mid Sem – 30 Marks
- End Sem – 50 Marks
- Practical – 50 Marks – Internal Assessment
- Practical – 50 Marks – External Assessment

Course Content

- **UNIT – I**
- **HTML, Javascript, JQuery**
- Introduction to Web Engineering, Web Programming vs. Web Engineering,
- Introduction to Web: HTTP, URL, Web Browser, Web Server, SMTP Server, ISP, Hyperlink, DNS, XML, Parsers and Internet based services, Web Architecture,
- An Overview about HTML (Basic Tags), HTML5 forms, GET and POST data, Introduction to Cascading style sheets, CSS3 Properties (BOX model, Advance Selectors),
- Responsive Designs, Need of responsive designs (bootstrap),
- Introduction to Javascript and jquery,
- Angular JS (A Client Side MVC framework).

Course Content

- **UNIT 2:**
- **PHP, Database Connectivity**
- Introduction to PHP, Associative arrays, Include, require, header, Developing Dynamic Content/Web page using PHP, Sessions and Cookies,
- Database Connectivity Using PHP and Insert Record into Database, Update, Delete and View Records from Database, Building a CRUD application,
- AJAX, How AJAX works, Case Study on Code Management Tool (Github), Secure Web Applications, Usability of web applications,
- Accessibility of web applications, Introduction to MVC, Model View Controller, Performance Optimization of Web Application.

Course Content

- **UNIT III**
- **PHP Framework**
- Introduction to PHP Framework (LARAVEL), MVC Routing, Static and Dynamic Routing, Route Parameters, Named Routes, Route Groups,
- HTTP Middleware Introduction, Defining Middleware, Registering Middleware, Middleware Parameters, Blade Templates Introduction, Template Inheritance, Defining A Layout, Extending A Layout,
- Database: Migrations, How to Work on View Section in LARAVEL, Basic Usage, Passing Data To Views, Sharing Data With All Views,
- Introduction Generating Migrations, Migration Structure, Running Migrations, Rolling Back Migrations, Writing Migrations, Creating Tables, Renaming / Dropping Tables, Database Seeding in LARAVEL, Writing Seeder, Running Seeder.

Course Content

- **UNIT IV**
- **MVC in LARAVEL, Web Services**
- MVC Controller in LARAVEL, Introduction Basic Controllers, Controller Middleware, MVC Model and Eloquent ORM in LARAVEL,
- Getting Started, Relationships, Collections, Introduction to Web Services, Restful Services,
- Introduction to SOAP Services, SOAP Service Architecture, WSDL with SOAP Service, UDDI with SOAP, Web Application Testing, Test Driven Development(TDD), TDD and Traditional Testing, Introduction to CMS Systems (Wordpress/Magneto).

Books References

- Web Engineering: A Practitioner's Approach by Roger Pressman and David Lowe, McGraw-Hill, 2009.
- Web 2.0 Architectures: What Entrepreneurs and Information Architects Need to Know by James Governor, Dion Hinchcliffe, and Duane Nickull, O'Reilly, 2009.
- Web Engineering: Modelling and Implementing Web Applications by Gustavo Rossi
- Web Engineering - The Discipline of Systematic Development of Web Applications, GertiKappel, Birgit Proll, Siegfried Reich, Werner Retschitzegger.

Books References

- Web Technologies, Uttam K Roy, Oxford University Press
- The Complete Reference PHP – Steven Holzner, Tata McGraw-Hill
- Web Programming, building internet applications, Chris Bates, Wiley Dreamtech
- Java Script, D. Flanagan, O'Reilly, SPD
- Beginning Web Programming - Jon Duckett, WROX.
- Programming World Wide Web, R. W. Sebesta, Pearson
- Internet and World Wide Web – How to program, Dietel and Nieto, Pearson.

Books References

- Web enabled commercial application development using HTML, DHTML, JavaScript, PERLCGI, Ivan Bayross.
- Beginning PHP5
- Complete Reference PHP
- Beginning PHP, Apache, MySql web development
- http://www.phpmyadmin.net/home_page/index.php
- http://www.w3schools.com/html/default.asp
- Tags reference : http://www.w3schools.com/tags/default.asp

References

- http://www.w3schools.com/css/default.asp
- http://www.php.net/manual/en/
- http://www.phpmyadmin.net/home_page/index.php
- http://in3.php.net/manual/en/function.mysql-select-db.php

What is Internet?

- Internet: network of machines (servers, clients, routers, switches, etc.) connected by media (fiber, wifi, etc.) that allows communication among devices.

- A global system of interconnected computers, using a standardized Internet Protocol suite for communication and sharing information is called the Internet.

The Internet

- We can think of the Internet as a graph:
  - Nodes represent devices and information
  - Edges represent a connection (physical or virtual)

World Wide Web == Internet?

World Wide Web == Internet?

- **No!** They are not the same!

- Latest revolution in the internet scenario

World Wide Web == Internet?

- The World Wide Web (WWW) is an **application** that **operates over the Internet**.
  - Internet provides infrastructure
  - World Wide Web utilizes the infrastructure to run an application on which users connect and exchange data
  - Other applications use the Internet as well, e.g. email

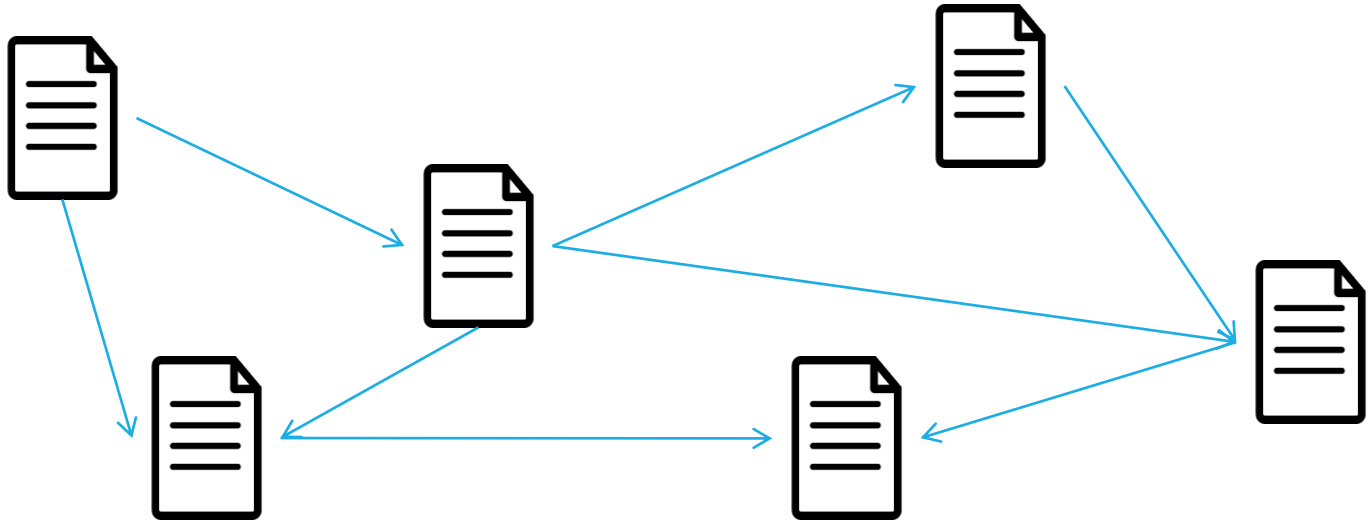
InterNet and World Wide Web

The World Wide Web

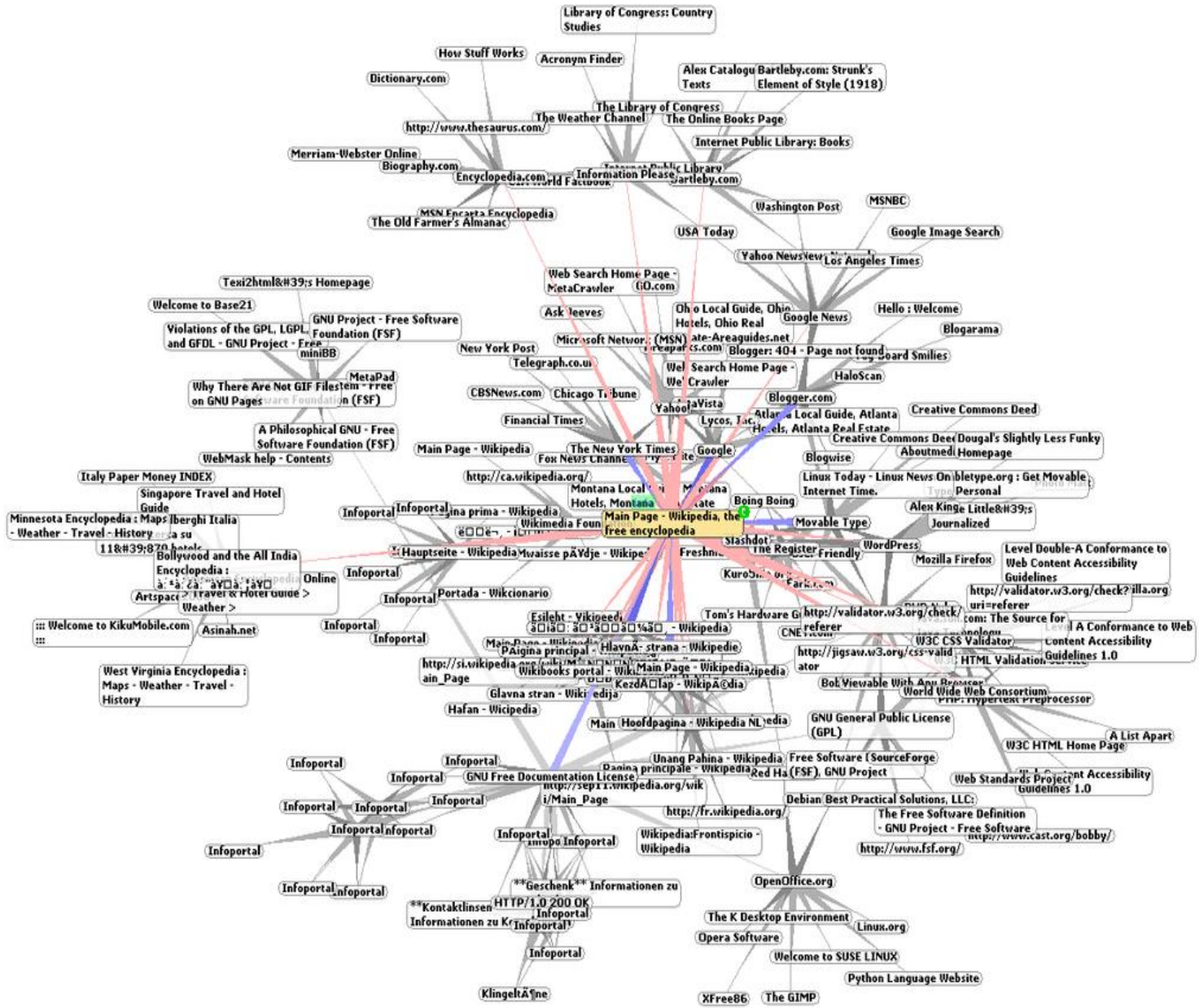
- **World Wide Web (WWW)** – an application on the Internet that combines many protocols that allow for communication and transfer of data between machines.
- WWW is that it allows to share multimedia documents between machines.
  - multimedia - document containing text, pictures, audio, clips, video, moving objects, animation.
- Web is composed of documents that are logically **linked** to each other.
- Originally designed to:
  - Provide easy access to documents for anyone
  - Provide way in which users can discover documents through a **browser**
  - Document contains text, images, videos, and other multimedia

The World Wide Web

- Web follows similar network structure as the Internet.
- Web pages link to other web pages, thus forming a graph where:
  - Nodes represent an individual document/resource
  - Edges represent a link from one document/resource to another (directed edges)



Graph of Wikipedia Pages



Content on the World Wide Web

- Static: same for all users and at all times
  - Appearance may vary based on specific browser, but content itself is the same
  - Technologies: HTML, CSS

- Dynamic: programmatically generated depending on the user, context, configuration, arguments, etc.
  - Technologies: JavaScript

WWW

- Where do all documents reside?
  - On web servers
  - Also called hypertext transfer protocol (HTTP) servers
- They are typically written in
  - HTML
  - Documents get formatted/displayed using
    - Web browsers
      - IE
      - Chrome …

What is a Web Browser?

- Browser: software that is used to access and display web content, and to navigate across the web

- Main Components of the Browser:
  - Rendering Engine (HTML/CSS) – responsible for static content presentation, formatting, and layout
  - JavaScript Engine (JavaScript) – responsible for creating and modifying dynamic content and appearance

Web Browsers and Servers

- A browser: is a program that can retrieve files from the world wide web and render text, images, or sounds encoded in the files.
  - A web browser is a software program that shows a web page.
  - It normally connects to the internet to access the document.
  - i.e. IE, Nescape, Mozilla, Chrome, etc...

- A web server: is an application which waits for client requests, fetches requested documents from disk and transmits them to the client.
  - computer or software that provides services to other applications known as clients.
  - The web browser requests online pages and services from the server.
    - i.e Apache

Web Page Addresses

- Uniform Resource Identifier (URI): a sequence of characters that allows the complete identification of any abstract or physical resource.

- Uniform Resource Locator (URL): type of URI that specifies the location on the WWW and the mechanism (protocol) for retrieving it.
  - A unique identifier for a resource on the World Wide Web that also specifies the protocol for retrieving it
  - The part that makes a URI a URL is the inclusion of the “access mechanism”, or “network location”

- URI – identify and
- URL - locate

URI

- Example:
- John
  - This is the person’s name, which is an identifier.
  - It is like a URI, but cannot be a URL, as it tells you nothing about the person’s location or how to contact him.
  - In this case it also happens to identify at least 5 other people in the area.
- 4914 West Bay Street, Nassau, Bahamas
  - This is a locator, which is an identifier for that physical location. It is like both a URL and URI (since all URLs are URIs).

URL

- A URL or **Uniform Resource Locator** is used to find the location of the resource on the web.
- It is a reference for a resource and a way to access that resource.
- A URL always shows a unique resource, and it can be an HTML page, a CSS document, an image, etc.
- A URL uses a protocol for accessing the resource, which can be HTTP, HTTPS, FTP, etc.

URL - Uniform Resource Locator

- URL format: protocol://hostname:port/path-and-file-name
- URL = "http:" "//" host [":" port] [abs_path ["?" query]]
- Examples:
  - http://www.google.com
  - http://www.google.com:80/ig?refresh=1
- The port 80 is default
- Hosts are case insensitive
- Paths are case sensitive
  - abs_path - Absolute Path – URL that can be accessed if provided to browser on its own
  - relative path – path to a file included within local file system, relative to the HTML page

URL - Uniform Resource Locator

- Some reserved characters are “encoded”
  - Encoding means representing as hex an ASCII code
  - Example: “ “ = “%20”
  - http://tools.ietf.org/html/rfc2396

URL - Uniform Resource Locator

https://www.w3schools.com/html/html_paragraphs.aspml

https://
identifies type
of transfer

www.w3schools.com
Domain Name -
name of remote computer

/html/html_paragraphs.aspml
File Location on Remote Computer

Examples

- ftp://ftp.is.co.za/rfc/rfc1808.txt
- http://www. ietf.org/rfc/rfc2396.txt
- ldap://[2001:db8::7]/c=GB?objectClass?one
- mailto.John.Doe@example.com
- Tel:+1-816-555-1212
- telnet://192.0.2.16:80/
- urn:oasis:names:specification:docbook:dtd:xml:3.6

What happened when you click on hyperlink or open a new web page?

- Determine URL and extract domain name.
- Use the name server to get IP address (DNS)
  - The browser goes to the DNS server, and finds the real address of the server that the website lives on
- Make a TCP connect to port 80
- And send a request for a web page
  - The browser sends an HTTP request message to the server, asking it to send a copy of the website to the client
  - This message, and all other data sent between the client and the server, is sent across your internet connection using TCP/IP
  - If the server approves the client's request, the server sends the client a "200 OK" message
  - then starts sending the website's files to the browser as a series of small chunks called data packets
- The server send the file and releases the TCP connection
- The client displays the document.
  - The browser assembles the small chunks into a complete web page and displays it

Other Possibilities

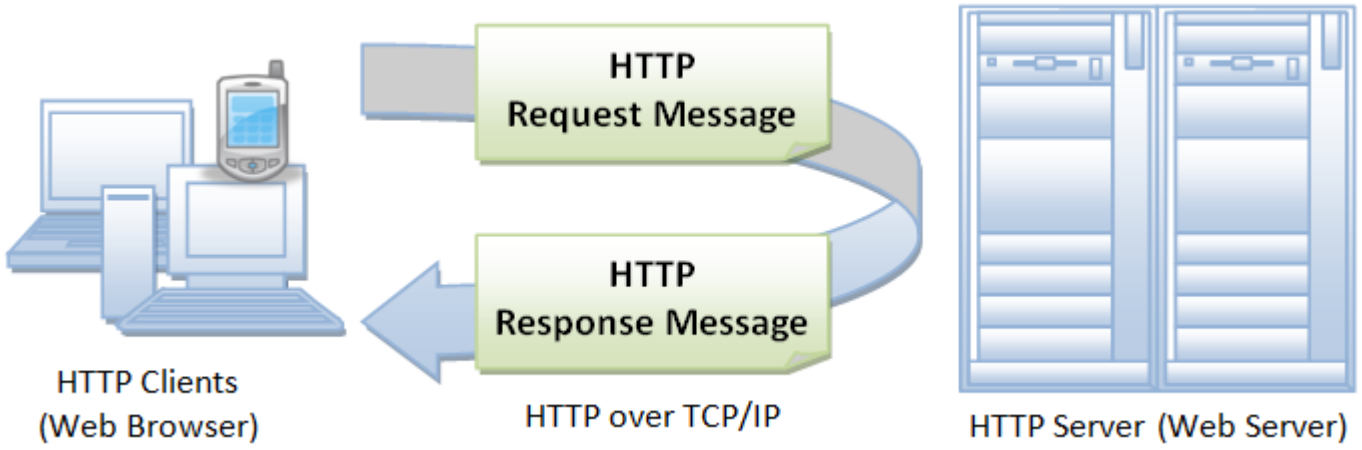
- The steps in the previous slide are for displaying a static web page from a remote machine.
- Other possibilities are:
  - Page is loaded from a local system
    - no tcp connection
    - url begin with file://...
  - The page is dynamically generated by a client-side script
    - No tcp connection
  - The page is dynamically generated by a server-side script:
  - The server may carry out other functions
    - Secure server
      - Check user's identity if they are authorised to access a particular resources

WWW Working

- Client use browser application to send URIs via HTTP to server requesting a webpage
- Web pages constructed using HTML (or other markup language) and consist of text, images, sounds, videos, etc..
- Servers (or caches) respond with requested web page
  - Or with error message
  - Client's browser renders the web page returned by the server
    - Page is written using HTML

How Does Web Browser Work?

- Browser and the World Wide Web utilize Hypertext Transfer Protocol (HTTP) to transfer documents



HTTP – **H**yper**T**ext Transfer **P**rotocol

HTTP

- HTTP - Hypertext Transfer Protocol
- Client-server protocol
- Http Server - Web Server
- Http Client – Web Browser
- Http - protocol used to transmit resources over the internet
  - Protocol using which web clients (browsers) interact with web servers

HTTP Overview

- HTTP is a plain-text, human readable protocol used for exchanging data on the Web
- Transfer content on the World Wide Web

- Initially developed by Tim Berners-Lee at CERN in 1989
- Stateless protocol – fresh connection for every item to be downloaded
- Based on client-server model:
  - Client sends **request** for resource, possibly including information about the client
  - Server sends **response**, including header (status information) and request resource

HTTP Protocol – Basic Glossary (RFC 2616)

- Connection - A transport layer virtual circuit established between two programs for the purpose of communication.
- Message - The basic unit of HTTP communication, transmitted via the connection.
- Request - An HTTP request message
- Response - An HTTP response message
- Resource - A network data object or service that can be identified by a URI (address of a resource)
- Client - A program that establishes connections for the purpose of sending requests.

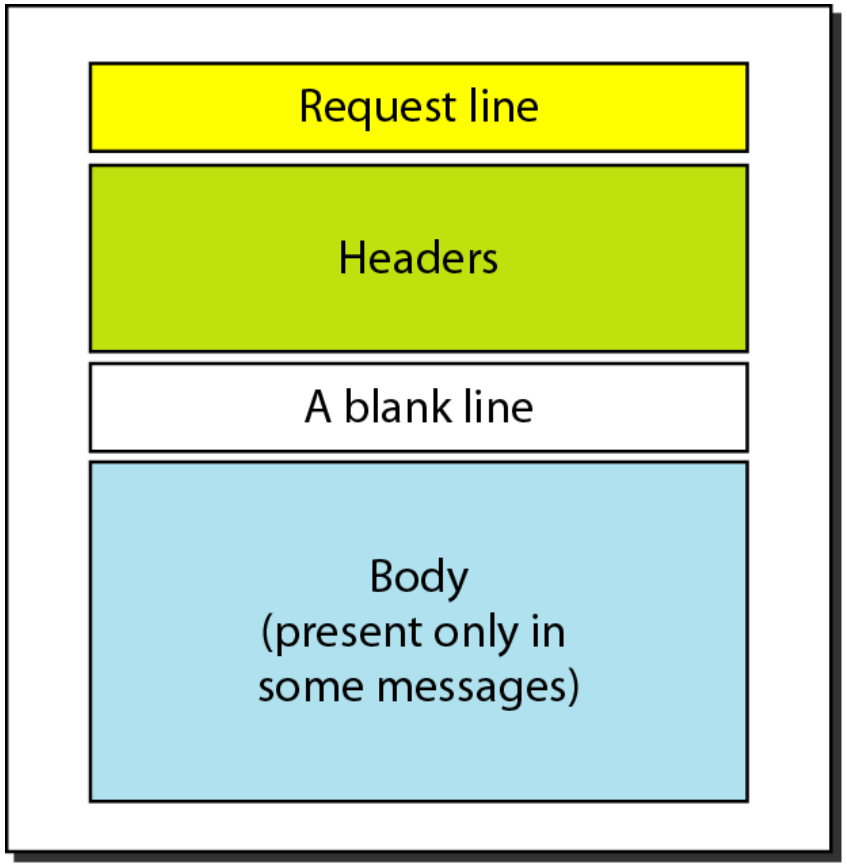
HTTP Protocol – Basic Glossary (RFC 2616)

- user agent - The client which initiates a request. These are often browsers, editors, spiders (web-traversing robots), or other end user tools.

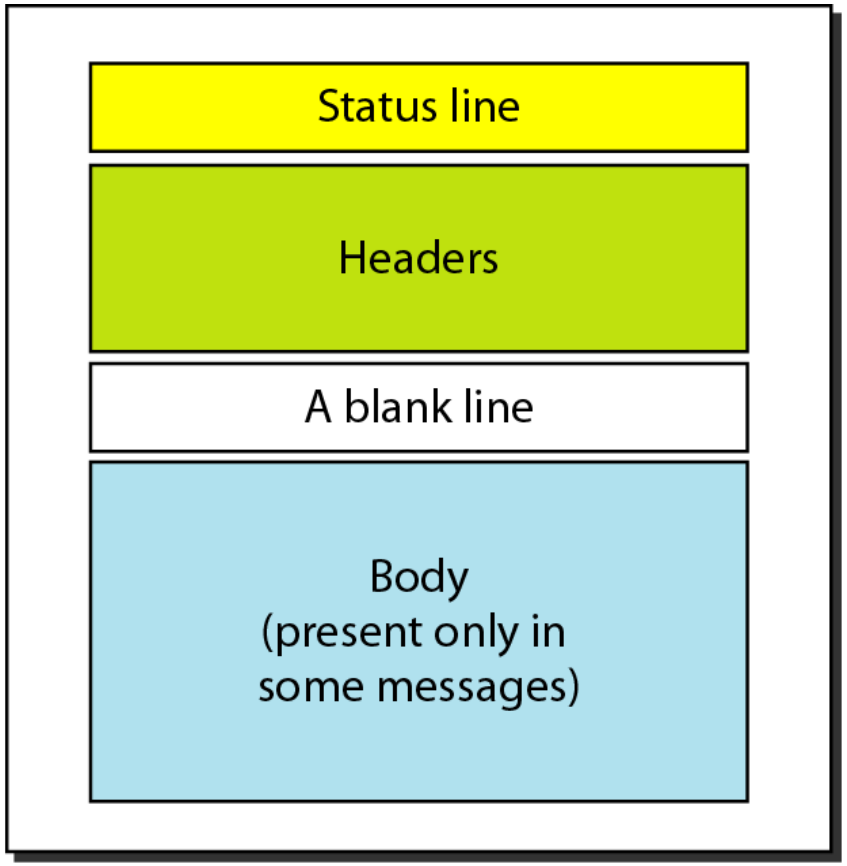
- Server - An application program that accepts connections in order to service requests by sending back responses.

- inbound/outbound - Inbound and outbound refer to the request and response paths for messages: "inbound" means "traveling toward the origin server", and "outbound" means "traveling toward the user agent"

Structure of HTTP Message



Request message



Response message

HTTP Request Message

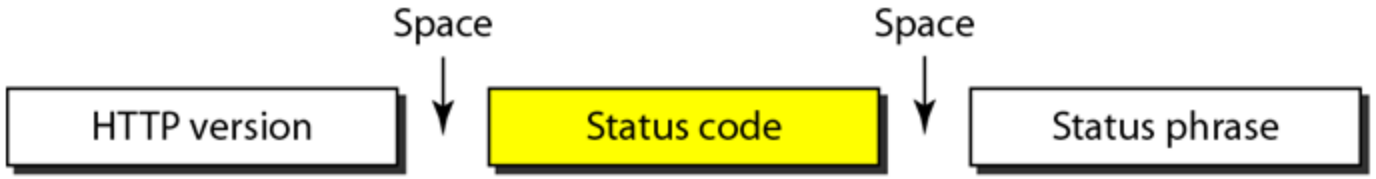
- First Line - Request Line
- Example: GET doc/index.html HTTP/1.1
  - GET = Request Method

Request Methods

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

HTTP Response Message

- First Line – Status Line
- Example: HTTP/1.1 200 OK



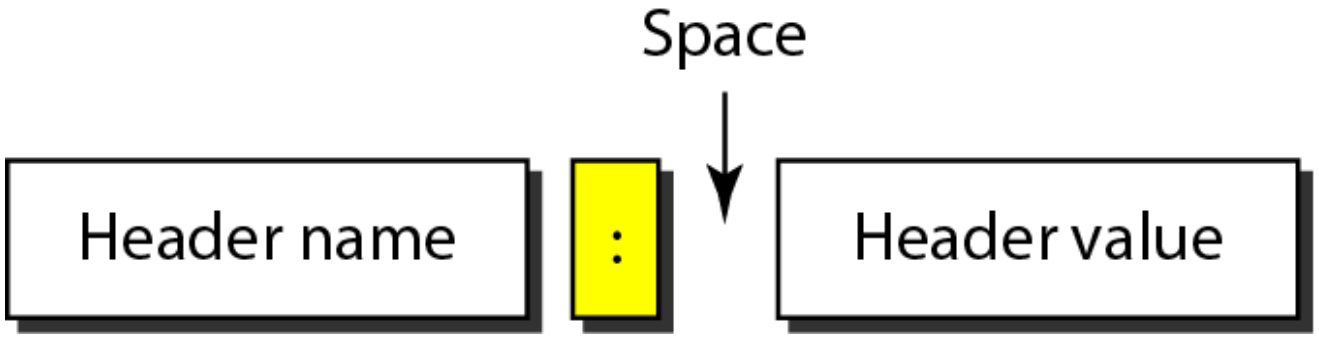
Status Codes

Code	*Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
	Success	
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

Status Codes

Code	*Phrase*	*Description*
Redirection		
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not been modified.
Client Error		
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
Server Error		
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

Header Format



HTTP Headers

- Used to define characteristics of the data requested or provided
- Host – name of the server (can be many hosts on a single IP address)
- Accept – type of resource accepted
- Content-type – the internet media type of the content (http://en.wikipedia.org/wiki/Mime_type)
- Authorization – data required for authentication
- Referrer – the link from where we have reached the current page (important in traffic analysis)

Response Headers

- Cache-control – specifies if the content should be cached or not
  - Content-Language
  - Location – implements redirect

General Headers

Header	*Description*
Cache-control	Specifies information about caching
Connection	Shows whether the connection should be closed or not
Date	Shows the current date
MIME-version	Shows the MIME version used
Upgrade	Specifies the preferred communication protocol

Request Headers

Header	*Description*
Accept	Shows the medium format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
From	Shows the e-mail address of the user
Host	Shows the host and port number of the server
If-modified-since	Sends the document if newer than specified date
If-match	Sends the document only if it matches given tag
If-non-match	Sends the document only if it does not match given tag
If-range	Sends only the portion of the document that is missing
If-unmodified-since	Sends the document if not changed since specified date
Referrer	Specifies the URL of the linked document
User-agent	Identifies the client program

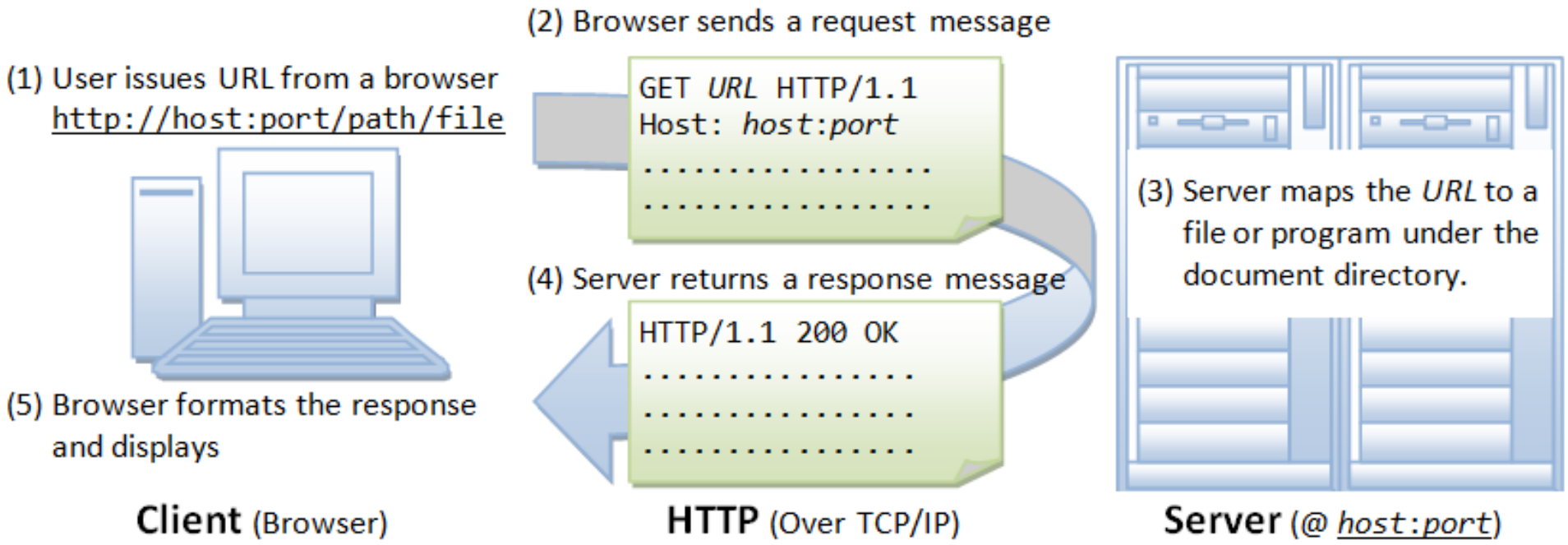
Response Headers

Header	*Description*
Accept-range	Shows if server accepts the range requested by client
Age	Shows the age of the document
Public	Shows the supported list of methods
Retry-after	Specifies the date after which the server is available
Server	Shows the server name and version number

Entity Headers

Header	*Description*
Allow	Lists valid methods that can be used with a URL
Content-encoding	Specifies the encoding scheme
Content-language	Specifies the language
Content-length	Shows the length of the document
Content-range	Specifies the range of the document
Content-type	Specifies the medium type
Etag	Gives an entity tag
Expires	Gives the date and time when contents may change
Last-modified	Gives the date and time of the last change
Location	Specifies the location of the created or moved document

HTTP Working



HTTP

- HTTP is a stateless protocol.
  - the current request does not know what has been done in the previous requests.

Example

- Send HTTP request:
  - www.google.com

HTTP Request Header

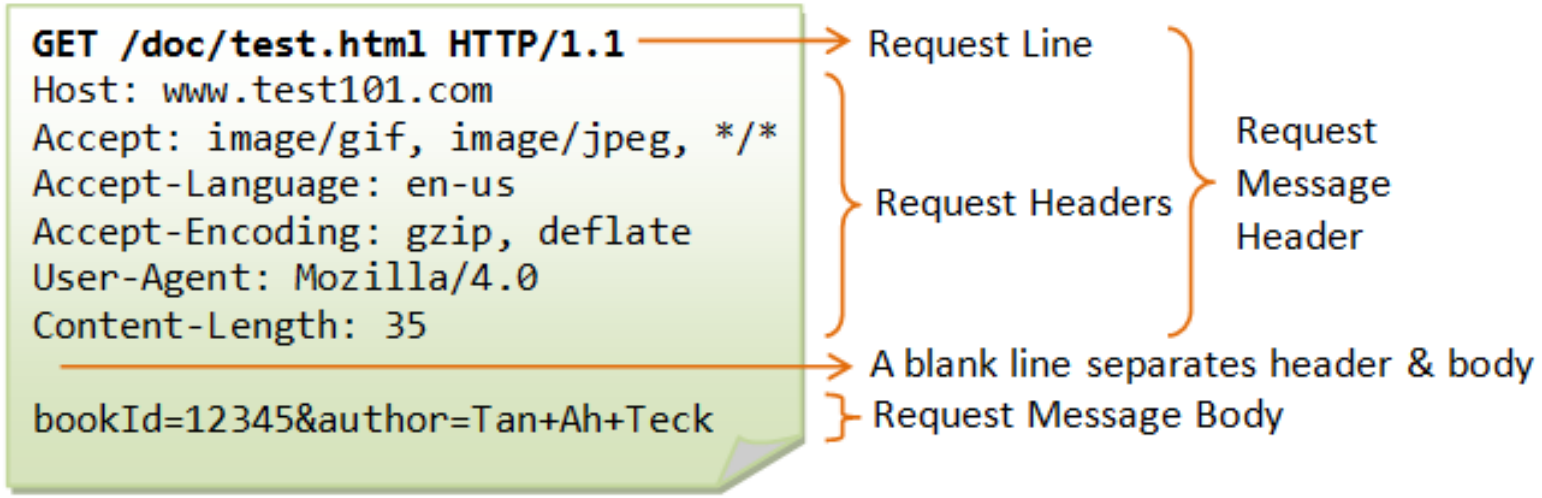
- **Host:** www.google.com
- **User-Agent:** Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0
- **Accept:** text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- **Accept-Language:** en-US,en;q=0.5
- **Accept-Encoding:** gzip, deflate, br
- **Cookie:** 1P_JAR=2018-07-23-10; NID=135=l9LKsJy_e87HkZcPhsLABh65UGqQgcUr0XwTi2bLPH4IkwEcipXkAl1nD7q-E4S-NUQAItaXU6axI72dEQ8cVGFw1XA7z1sNkNWWP717mNvuo3uTVHtI5CCB4G1U6IpGAO-t1_hB17A; OGP=-5061451:; DV=s6EcCToHJQAYEDch_YrFCheMS1VpTBY
- **DNT:** 1
- **Connection:** keep-alive
- **Upgrade-Insecure-Requests:** 1
- **Cache-Control:** max-age=0
- **TE:** Trailers

HTTP Response Header

- HTTP/2.0 200 OK
- domain=www.google.com
- date: Mon, 3 Jan 2022 10:00:49 GMT
- expires: -1
- cache-control: private, max-age=0
- content-type: text/html; charset=UTF-8
- content-encoding: br
- expires=Thur, 2-Feb-2022 10:00:49 GMT; path=/; domain=.google.com

HTTP request example

- Open any web page: right click – inspect element – click on network – choose any page – see information on right side



HTTP Response example