Internet

* global network of networks
* from internetwork
* internet – outside
* network – interconnection of devices
* categorized by:
* nodes – actual devices
* interconnection technology – wired or wireless
* protocols
* device drivers
* linked together based on IP
* supports communications using TCP/IP
* uses or makes accessible high level services
* 1969 – beginning of the internet
* 1972 – email was developed (first killer application)

Circuit Switch Connection

* Establish circuit and maintain it to communicate
* Not scalable

Store and Forward

* Send messages to the next hub until destination is reached

Wide Area Information Services (WAIS)

* Connected to servers from different locations to get data from each server periodically and index these data

Gopher Protocol

* Distributing, searching and retrieving documents over the internet
* Hierarchy is involved with the index having sub-indexes

Usenet

* Similar to an online discussion group

World Wide Web (WWW)

* Established in 1989
* Created by Tim Berners-Lee
* Worked with CERN
* HTTP, HTML, URL
* Web server and web client
* Information system that allows documents to be connected to other documents
* An arrangement of web servers that boost particularly designed records
* Two most popular browsers people use is the Google Chrome and the Mozilla Firefox
* There are a few applications called Web programs that make it simple to get to the World Wide Web
* All significant Web locales have balanced their substance outline and improvement way to deal with oblige the quickly expanding division of the populace getting to the Web from little screen telephones rather than extensive screen desktop and smart phones

Web Server

* Hosts web resources
* Listens for requests

Web Client

* Example is web browser
* Gives requests to web servers
* Follows HTTP to communicate with the server

Semantic Web

* Next step in the evolution of the web
* Makes use of artificial intelligence to understand the question given by user

Hypertext Transfer Protocol (HTTP)

* Application layer communications protocol used to access resources
* Developed by W3C (World Wide Web Consortium) and IETF (internet Engineering Task Force)
* Standard way of communicating through applications
* Version 0.9 came out in 1991
* HTTP 1.0 (RFC 1945, May 1996) – first standardized
* HTTP 1.1 (RFC 2068 Jan 1997, RFC 2616 Jun 1999), RFC 7230 – 7235 (Jun 2014)
* HTTP 2 (RFC 7540 May 2015) – patterned after SPDY of Google

HTTP Fundamentals

* Runs on top of TCP/IP, using port 80 as default or 443 for HTTPS
* HTTPS is encrypted and needs digital or self-signed certification
* IANA – allocates certain ports to certain applications
* Based on client-server architecture
* Clients a.k.a. user agent
* Web browsers, web crawlers/spiders, other end user tools and applications
* Any application that communicates with HTTP protocol
* Server
* Origin server – resources are actually there
* Proxy server, gateway, tunnel (blind relay between two points) – can be used for authentication
* Uses request-response protocol
* Client sends an HTTP request message to server (pull protocol)
* Server processes the request and replies with HTTP response message
* In HTTP 2, server can push resources to client without client requesting
* Stateless communication
* Do not keep information about clients in between requests
* Other functionalities
* Cache control – storage for easy and fast access
* Content media type (MIME) specification
* Language and character set specification
* Content/ transfer coding
* Content negotiation – talk to tell what the recipient can handle
* Client-server protocol negotiation – asking server if it can handle higher version and if yes, the server will upgrade
* Persistent connections – telling the server to not close the connection for further requests
* In HTTP 1.0, connection is terminated after a resource has been retrieved from responding to a request
* Request pipelining – sending requests one after another
* In HTTP 1.0, only one resource can be transferred per connection
* Authentication/ authorization

HTTP Resource Accessing

* HTTP resources are identified using URIs, which tells what the resource is, or more, specifically HTTP URL, which tells where the resource is
* Scheme (http or https)
* Authority
* User information/ information credentials
* Host-domain name (resolved to an IP address using DNS) of the server where the resource resides, or will be created
* Port number – default is 80
* Path to resource (resolved relative to the document root) – may refer to a static or dynamic resource
* Query – typically provided as key = value pairs, with ampersand separators between key/ value pairs, and may be URL-encoded
* Fragment identifier – “bookmark”

Absolute URL

* Scheme and domain name are always required

Relative URL

* Scheme, user info, and domain name can be omitted

HTTP Request Message

* Request Line (CRLF – terminated line consisting of three spaced-separated items)
* Request Method
* GET
* HEAD
* POST
* PUT
* DELETE
* CONNECT
* OPTIONS
* TRACE
* Request URI – location of the requested resource
* HTTP protocol version
* Request Header Fields – information about the request and the client
* Accept-Charset
* Accept-Encoding
* Accept-Language
* Authorization
* Expect
* From
* Host
* If-Match
* If-none-match
* Range
* If-range
* If-modified-since
* If-unmodified-since
* Max-forwards
* Proxy-authorization
* Referer
* TE
* User-agent
* Message headers
* HTTP 1.1 requires at least the host request header to be provided
* Empty line (CRLF)
* Message body a.k.a. payload – optional

HTTP Response Message

* Response Header Fields
* Accept-Ranges
* Age
* ETage
* Location
* Proxy Certificate
* Retry-After – when maintenance of a web server; timeouts
* Server – displays what server
* Vary – changing header values
* www-Authentication
* Status line (CRLF)
* HTTP protocol version
* Status code – 3-digit code that designates the status
* Reason phrase – descriptive meaning og the status code
* 1xx (info), 2xx (success), 3xx (redirection), 4xx (client), 5xx (server)
* Message headers
* Empty line
* Message body – optional

HTTP Request Methods

* Standard
* GET

HTTP Status Codes

* Server Responses
* Informational (1xx)
* 100 – continue
* 101 – switching protocol
* Success (2xx)
* 200 – OK
* 201 – Created
* 202 – Accepted
* 203 – Non-Authoritative Information; some are not from the origin server.
* 204 – No Content
* 205 – Rest Content
* 206 – Partial Content; in other words, this is not the whole resource.
* Redirection (3xx)
* 300 – Multiple Choices
* 3001 – moved Permanently
* 302 – Found
* 303 – See Other
* 304 – Not Modified
* 305 – Use Proxy
* 306 – n/a (unused status code)
* 307 – Temporary Redirect
* Client Error (4xx)
* 400 – Bad Request
* 401 – Unauthorized
* 402 – Payment Required
* 403 – Forbidden
* 404 – Not Found
* 405 – Method Not Allowed
* 406 – not Acceptable
* 407 – Proxy Authentication Required
* 408 – Request Time-out
* 409 – Conflict
* 410 – Gone
* 411 – Length Required
* 412 – Precondition Failed
* 413 – Request Entity Too Large
* 414 – Request-URI Too Large
* 415 – Unsupported Media Type
* 416 – Request Range not Satisfiable
* 417 – Expectation Failed
* 426 – Upgraded Required (recently introduced status code)
* Server Error (5xx)
* 500 – Internal Server error; server’s fault
* 501 – Not Implemented
* 502 – Bad Gateway
* 503 – Service Unavailable
* 504 – Gateway Timeout
* 505 – HTTP Version not Found

\*\*Saang category yung **Entity Header Fields**?

Entity Header Fields

* Allow
* Content-Encoding
* Content-Language
* Content-Length
* Content-Location
* Content-MD5 (deprecated header)
* Content-Ragne
* Content-Type
* Expires – for controlling cache access; cache-control; (e.g. max-age = ??)
* Last-Modified

If ever need ng definition sa mga header field terms, pwede dito :D :)

HTTP 1.1 RFC 2616 : Header Field Definitions

https://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html