URL's &
HTTP Request and Response

Edgardo Molina, PhD | Head Instructor
CUNY Tech Prep 2020-2021

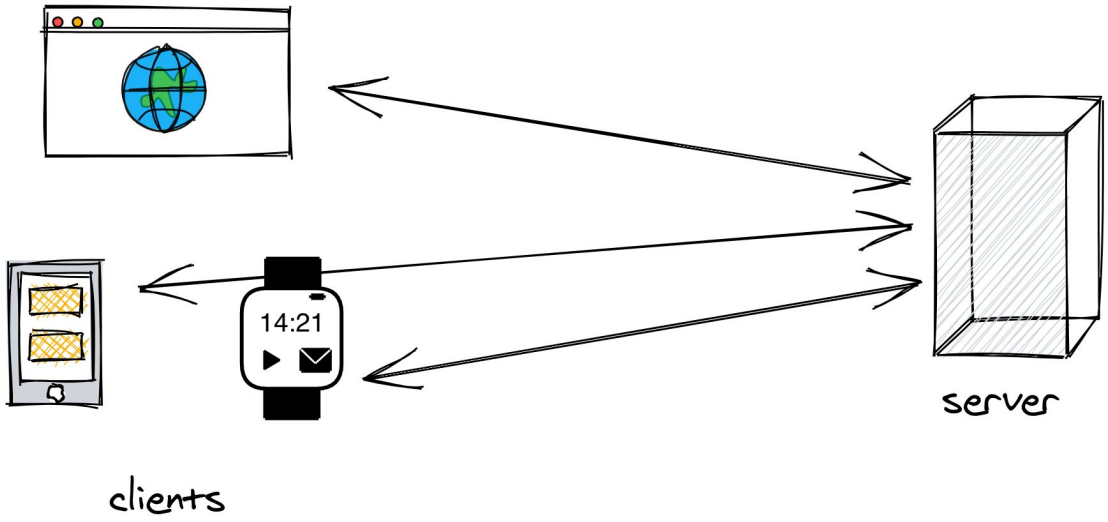
Outline

Client - Server Communication

Parts of a URL

HTTP Request - Response

Client - Server Communication



Client - Server Communication

In the context of Web Applications:

- A **client** program requests a web page
- The **server** program responds with the page or an error page

In general:

- Clients **request** a resource
- The server **responds** with that resource or an error status

Client - Server Communication

In the context of Web Applications:

- A **client** program requests a web page
- The **server** program responds with the page or an error page

In general:

- Clients **request** a resource
- The server **responds** with that resource or an error status

Networked Applications:

- The client and server are separate programs
- They can run on the same device, or
- They can run on different devices and connect across a network
- In both cases, they use the same network protocol

Types of Clients

Web Browsers

- Desktop: Chrome, Firefox, Safari, Edge
- Mobile: Safari, Chrome

Other Programs

- Games, iOS/Android Apps, Scripts
- Command line utilities (*i.e.* npm, npx)
- Background processes (*A user does not have to drive the program*)

How does a client find the server?

https://cunytechprep.nyc/index.html

http://localhost:3000

http://127.0.0.1/about

URL's

URL: Uniform Resource Locator

URL's are also known as Web Addresses

URL's specify the location of a resource on a computer network

```
http://cuny.edu/index.html
----   -------- ----------
  1        2          3
```

1. Application Protocol
2. Hostname
3. Resource Path

Parts of the URL

1. Application Protocols
    - HTTP - HyperText Transfer Protocol
    - HTTPS - SSL Encrypted HTTP
2. Hostname
    - Registered Name (purchased from a Domain Name Registrar)
    - TLD - Top-Level Domain (.com, .edu, .org, .nyc, etc.)
        - 1000's available
    - Points us to the server
3. Path
    - Maps to a filesystem path for a specific document on the server
    - Or, the path is passed to a Web Application program as an input

General URL Format

```
scheme:[//[user:pass@]host[:port]][/]path[?query][#fragment]
```

- **scheme**: Application protocols (http/https)
- **[user:pass@]**: *Optional*. Authentication information
- **host**: A hostname or IP Address
- **[:port]**: *Optional*. HTTP/HTTPS default ports 80/443
- **path**: Resource location. Can be /
- **[?query]**: *Optional*. Parameter and Argument list passed to web application
- **[#fragment]**: *Optional*. Identifier for secondary resource. Evaluated by client

How does the **Hostname** get us to a specific server?

The DNS (Domain Name System) translates hostnames to IP addresses

All computers on a network are assigned an IP Address for computer to computer communication

For computers on the internet, your Internet Service Provider (ISP) assigns you an IP address

An IPv4 address is a 32-bit number. A four byte number, where each byte is separated by a (`.`) period

```
127.0.0.1
```

DNS

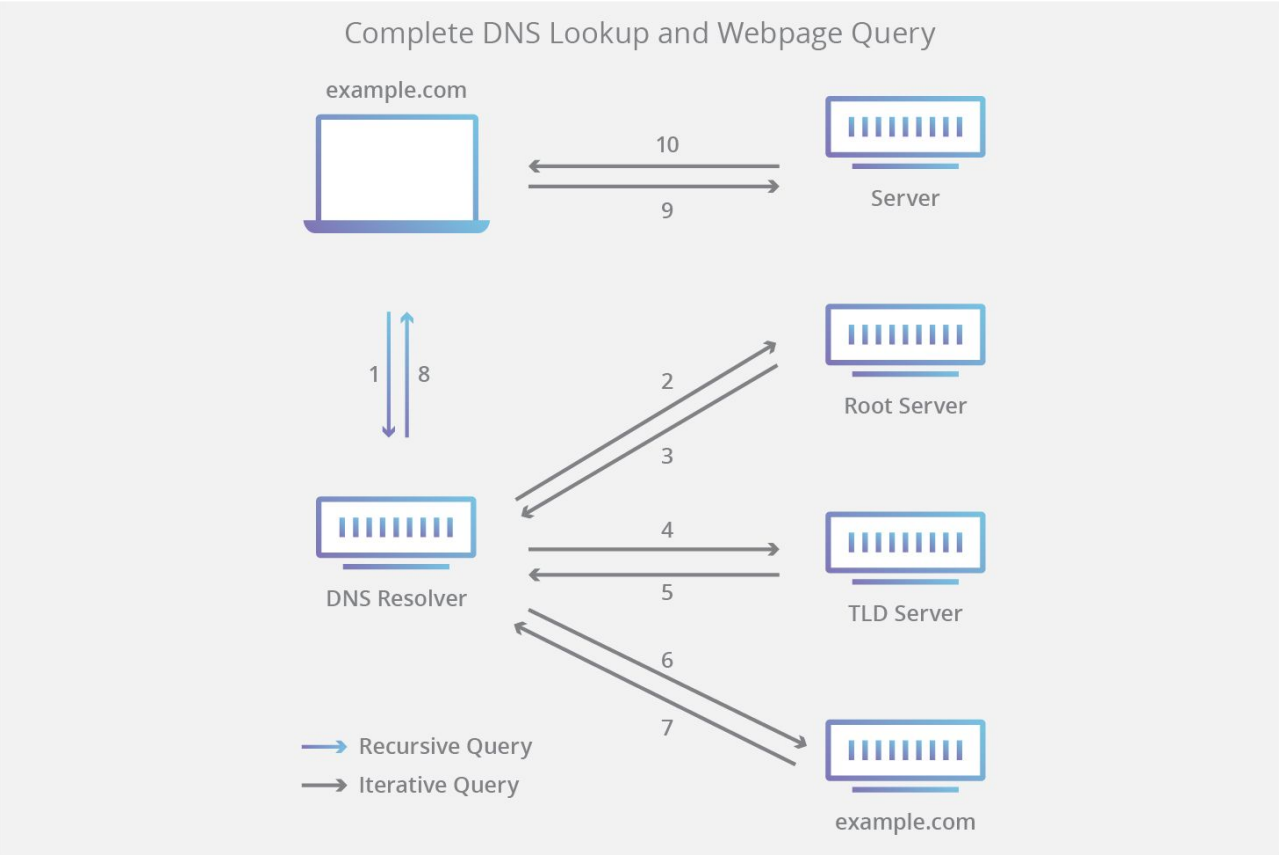


Image from: https://www.cloudflare.com/learning/dns/what-is-dns/

Private IP Addresses

- `127.0.0.1` and `localhost` refer to the local machine
- `0.0.0.0` refers to the default address for local machine
- `10.0.0.0 – 10.255.255.255`
- `172.16.0.0 – 172.31.255.255`
- `192.168.0.0 – 192.168.255.255`

Private addresses can be used in closed networks (home, small office, virtual networks, WiFi Access Points)

More Resources

https://en.wikipedia.org/wiki/URL

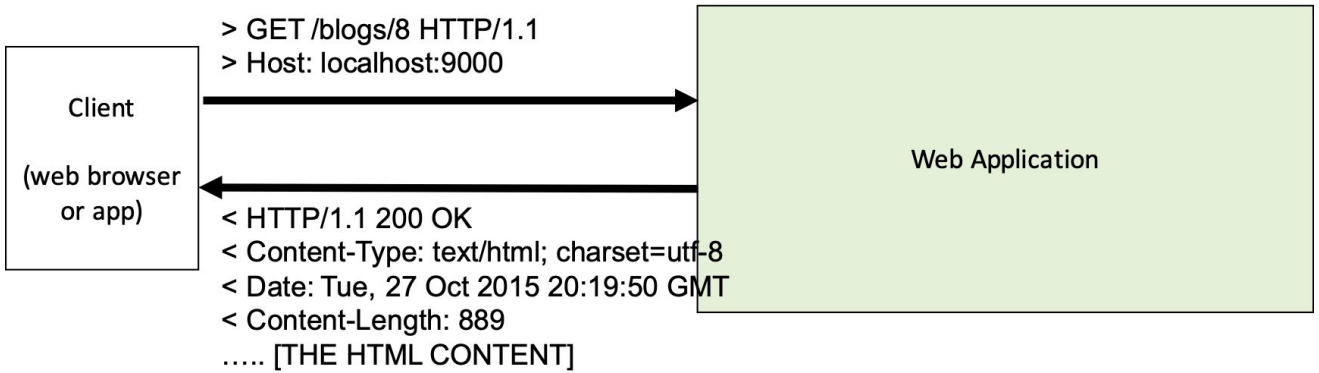
https://en.wikipedia.org/wiki/List_of_Internet_top-level_domains

https://www.cloudflare.com/learning/dns/what-is-dns/

https://howdns.works/

HTTP Request - Response

What is happening when we call an HTTP URL?



HTTP Request - Response Cycle

- Client sends an HTTP Request
  - Specifies an **HTTP Method** (GET, POST, PUT, PATCH, DELETE)
  - Asks for a **path**
  - Payload may include: JSON, files, nothing, …

- Server sends back an HTTP Response
  - Specifies a **status code** (200, 404, 500)
  - Payload may include: HTML, JSON, XML, nothing, …

The Request

When we load a URL in our browser we are making a **`GET`** request

HTTP supports other types of request methods

https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#Request_methods

HTTP Methods are sometimes referred to as Verbs

They do not explicitly change anything about the request except some headers. It is up to your web application to interpret them appropriately.

HTTP Methods (*Verbs)*

- `GET`
  - Use to retrieve a resource
  - *`GET`* requests should not modify the content
- `POST`
  - Use to create a new resource entity
- `PUT/PATCH`
  - Use to update an existing resource entity
- `DELETE`
  - Use to delete an existing resource entity

The Response

A web server processes the request and sends back an HTTP response.

The response contains a **status code** in the header. The status code lets the client know if the request succeeded, failed, or some other action is required.

In addition to the status code, **content** may be included in the response (the html, json, xml, file, etc).

Types of Status Codes

1xx: Informational

2xx: Success

3xx: Redirection

4xx: Client Error

5xx: Server Error

Common Types of status codes

- 200 – Success
- 201 – Created
- 301 – Moved permanently
- 303 – See other (redirect)
- 400 – Bad request
- 404 – Not found
- 500 – Server Exception

Full List: https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

HTTP is Stateless

Wikipedia definition

> *HTTP is a stateless protocol. A stateless protocol does not require the HTTP server to retain information or status about each user for the duration of multiple requests. However, some web applications implement states or server side sessions using for instance HTTP cookies or Hidden variables within web forms.*

- The browser opens a connection to server
- Browser sends a request
- Server processes the request
- Server sends back a response
- Server ***closes*** the connection