URI (Uniform Resource Identifier)

URL (Uniform Resource Locator)

URN (Uniform Resource Name)

Web Browsers and Web Servers

1. URI (Uniform Resource Identifier)

A URI is a string that identifies a resource, the resource may be a static file (document), like for instance **index.html**, or an application like for instance **calculate**.

The URI definition encompasses both the URN and URL concepts.

Resources are physically located in servers, an URN is a globally unique string that identifies a resource by identifying not only its name but also the server where the resource is.

Within a server, resources are often placed in a directory structure, so to identify them within the server we must include the path to reach it from the root folder (/). For instance:

/pages/info/document-list.html

This is a URI but it's neither a URN nor a URL because it doesn't identify the server where this resource is stored.

A URN identifies the server where the resource is stored in the generic form:

{SERVER}/{PATH-TO-RESOURCE}/{RESOURCE}

The {SERVER} represents a network node's address, usually a DNS name, but it may also be an IPv4 or IPv6 node address.
A URN is globally unique because we assume the {SERVER} is globally unique, and within the same server we can't have two resources with a same name and same path location.

Of course, every URN is also a URI, but not every URI is a URN.

2. URL (Uniform Resource Locator)

The URL goes one step further, it not only identifies the resource globally like a URN, but also specifies the way to access it, usually by including a protocol specification.

The generic of a URL is:

{ACCESS}://{SERVER}/{PATH-TO-RESOURCE}/{RESOURCE}

The **{ACCESS}** part of the URL specifies a mean of accessing the resource on the server, typically by using and appropriate content transfer protocol, like for instance HTTP, HTTPS or FTP.

In this case, the **{SERVER}** specification may also, optionally, include elements related to the access to the server:

[{USERNAME}@]{SERVER}[:PORT-NUMBER]

When no **{USENAME}@** is used, it's assumed the access doesn't require a login, if no **:PORT-NUMBER** is used, the default standard port number for the protocol is used, e.g. 80 for HTTP and 443 for HTTP.

3. Web Browsers and Web Servers

When you type on your browser something like **server.example.com**, that's not a URL, however the browser application by itself will assume a default mean of accessing the server, usually HTTP. So the provided string is regarded as a server's DNS name and transformed into the URL, following the example it would be **http://server.example.com**.

Keeping with the same example, apparently **http://server.example.com** doesn´t represent a resource, it's just a server and the way to access it. In this case the browser application assumes the resource is **/** that stands for a folder (folders are also resources), and this case is the root folder of the documents' directory tree on the server side.

Regarding HTTP, browsers also assume a default HTTP access method and that's GET, so to exactly describe what a browser will do when **server.example.com** is provide it's:

- Opens a TCP connection with port number **80** of the server with DNS name **server.example.com**.
- Send through it an HTTP request **GET /**

How a request for a folder is handled by the server, depends on the server's configuration, they may provide an HTML page with the folder's content (objects listing). Nevertheless, most HTTP servers are configured to return back instead the content of a specific file present in the folder, usually a file named **index.html**.

Servers may also send back a response asking the browser to redirect the request to a different URL, often the same URL using HTTPS instead of HTTP.

4. HTML references

Many HTML tags have references to other resources, for instance:

```
<script src=URI><script>
<img src=URI />
<a href=URI></a>
```

These references to external contents may be URLs, or not. When a browser loads an HTML page it sets the **location** of the loaded content, and that's a **URL representing the folder** from where the resource was loaded, any references within the document (URIs) are relative to the current location, except if they are URLs.

For instance if an HTML page is loaded from URL **http://serv.ex.com/testing/page1.html**, then location is set to **http://serv.ex.com/testing**, references within the document that are not URLs are relative to this location, for instance:

URI in the page	Corresponding URL to be used
/newpages/list.html	http://serv.ex.com/newpages/list.html
doc.html	http://serv.ex.com/testing/doc.html
http://s2.example.com/doc.html	http://s2.example.com/doc.html
../newpages/list2.html	http://serv.ex.com/newpages/list2.html

Using URIs that are not URLs in HTML pages is most useful because those pages will work with no changes on any server where they are deployed as far as relative locations between pages are kept.

The part of the location URL value, corresponding to the access protocol and server is known as the content's **origin**:

ACCESS://[{USERNAME}@]{SERVER}[:PORT-NUMBER]

For instance for location **http://serv.ex.com:8080/testing/**, the origin is **http://serv.ex.com:8080**.
For security reasons browsers enforce a **same origin policy** regarding some operations by JavaScript functions, namely when those functions make HTTP requests (AJAX), those requests must be directed to the same origin as the loaded page.