



PHP

Clase 4

HTTP

**Hypertext Transfer
Protocol**

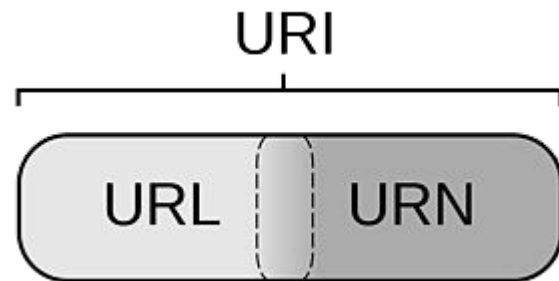


Es el protocolo de comunicación que permite la transferencia de información en la web.



URI

Una **URI** es un **identificador de recursos uniforme**.



URL

Indica dónde se encuentra el recurso pero puede variar. Siempre comienza con un protocolo (**http**, **ftp**, etc...)

URN

Nombre de recurso uniforme. No indica exactamente dónde se encuentra. No empieza con protocolo.

URI - Partes

1. Schema
2. Dominio o Dirección IP
3. Número de puerto **(opcional)**
4. Nombre del recurso
5. Cadena de consulta
6. Identificador de fragmento

Ejemplo:

`http://test.com:80/recurso.php?queryString#id`

1

2

3

4

5

6



Usuario

1. Petición
(GET miArchivo.php)



Aplicación Servidor

2. Búsqueda en el
repositorio de archivos

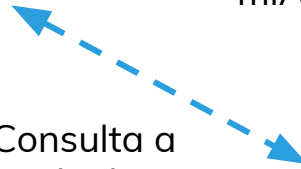


Archivos

3. Obtenemos el archivo
miArchivo.php



4. Consulta a
base de datos si
se necesita
mientras se
genera el código
HTML



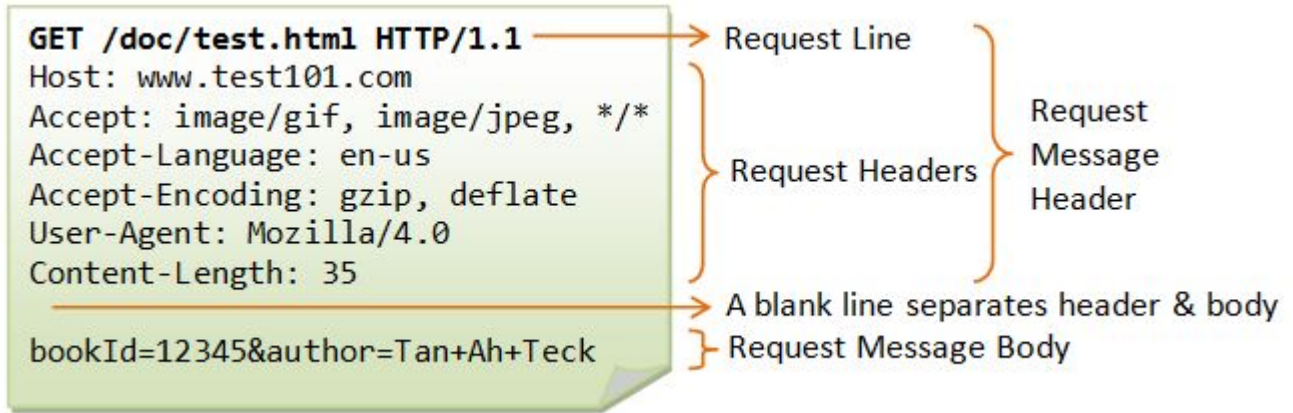
Base de Datos

5. Se devuelve el
código HTML
correspondiente a
miArchivo.php



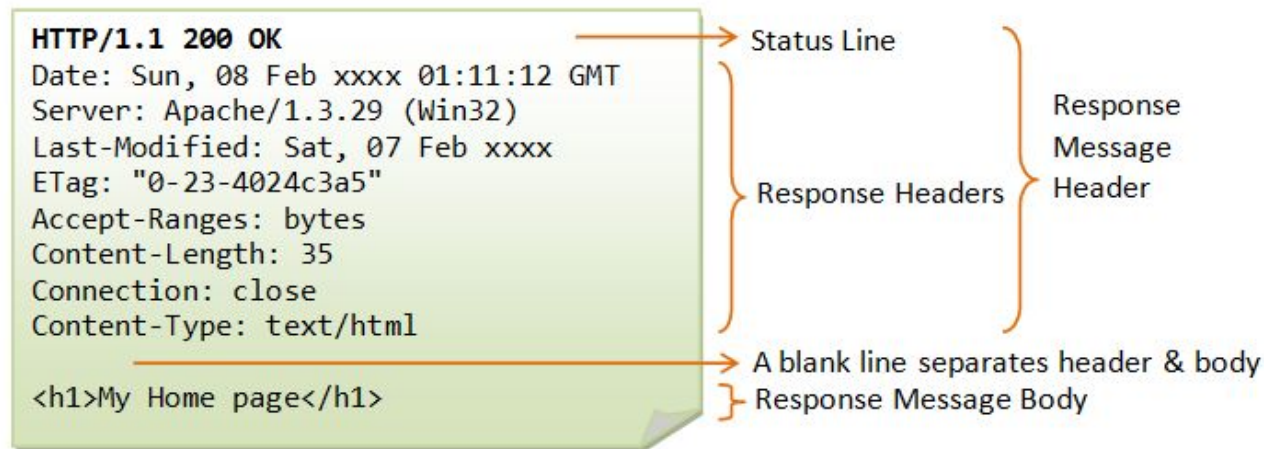
HTTP - Request

Al contactar al servidor, el protocolo **HTTP** envía y recibe información a través de **headers**. En este caso vemos un header de **“Request”**, es decir, pedido.



HTTP - Response

En el “**Response**”, es decir, en la respuesta del servidor, también aparecen **headers**.





HTTP - Status Codes

Algunos códigos son:

- **200** - OK
- **301** - Moved Permanently
- **307** - Temporary Redirect
- **403** - Forbidden
- **404** - Not Found
- **500** - Internal Server Error
- **503** - Service Unavailable

[Clickeame para ver un par más...](#)



... nemotécnicamente

Agrupados por centenas:

- **1xx** - Hold on... (a.k.a.: aguantá un toque capo...)
- **2xx** - Everything cool (a.k.a.: todo viento...)
- **3xx** - Go away (a.k.a.: tomatelá...)
- **4xx** - Your fault (a.k.a.: mala tuya...)
- **5xx** - My fault (a.k.a.: mala mía...)

HTTPS

Hypertext Transfer
Protocol



Este protocolo es la versión segura de **HTTP** (***Hypertext Transfer Protocol***) que todos utilizamos habitualmente. Básicamente, lo que ocurre es que el **Servidor Web** codifica la sesión con un certificado digital.



HTTPS

**Hypertext Transfer
Protocol**



De este modo, el usuario tiene ciertas garantías de que la información que envíe desde dicha página no podrá ser interceptada y utilizada por terceros.

HTTP

Método GET



El método **GET** se utiliza para **pedir información** de un recurso específico.

El **query string** se incluye en la **URL** conteniendo los datos a enviar.

/miArchivo.php?nombre=Juan&apellido=Perez

query string

HTTP

Método GET

- Los pedidos pueden ser cacheados.
- Aparecen en el historial.
- Pueden ser agregados a marcadores
- Nunca deberían usarse con información sensible.
- Tienen restricción de longitud (**2048** caracteres aunque configurable).
- Solamente deberían ser utilizados para **obtener** información.

HTTP

Método POST

El método **POST** envía **información** a ser procesada por un recurso específico. El **query string** viaja en el cuerpo del mensaje a través de **HTTP** y no está visible en la **URL**.

POST /miArchivo.php HTTP 1.1
Host: localhost

nombre=Juan&apellido=Perez
query string

HTTP

Método POST



- Los pedidos no pueden ser cacheados.
- No aparecen en el historial.
- No pueden ser agregados a marcadores.
- Sin restricción de longitud del query string.

GET vs POST

	GET	POST
Botón Atrás	Sin consecuencias	Se reenvía la información
Marcadores	Acepta	No acepta
Cache	Acepta	No acepta
Restricción longitud	Sí	No
Seguridad	No utilizar con datos sensibles	Más seguro que GET
Visibilidad	Se ve la información en la URL	No se ve la información en la URL

GET vs POST en URL

Método get:

`http://www.test.com/index.php?name1=value1&name2=value2`

query string

Método post:

`http://www.test.com/index.php`


???

HTTP


Métodos

Existen otros métodos menos conocidos:

- OPTIONS
- GET
- HEAD
- POST
- PUT
- DELETE
- TRACE
- CONNECT



PHP nos permite entender qué tipo de
pedido llega y acceder a los datos
recibidos a través de...





Variables
Super Globales

PHP

Superglobales



Las variables superglobales son variables disponibles siempre, en todos los ámbitos.

No tenemos que declararlas y toman los valores de forma automática.

PHP

`$_GET`



`$_GET` es una variable **superglobal**.

Es un array asociativo que es generado a partir del **query string** de un pedido **GET**.

PHP

`$_GET`



Si hay un pedido a:

<localhost/test.php?nombre=Juan&apellido=Perez>

query string

`$_GET` tendría el siguiente contenido:

[“nombre” => “Juan”, “apellido” => “Perez”]

Entonces, **`$_GET[“nombre”]`** tiene el valor **“Juan”**.

PHP

\$_POST



\$_POST es otra variable superglobal. Como **\$_GET**, es un array asociativo que se genera a partir del **query string** de un pedido **POST**.

Funciona igual a **\$_GET** pero en pedidos hechos por **POST**.



Superglobales

Otras superglobales:

- `$GLOBALS`
 - `$_SERVER`
 - `$_GET`
 - `$_POST`
 - `$_FILES`
 - `$_COOKIE`
 - `$_SESSION`
 - `$_REQUEST`
 - `$_ENV`
- 

¡A practicar!

Ejercicios del **1** al **6**



```
<?php  
    echo "Hora de practicar!";  
?>
```

¡Gracias!



¿Preguntas?