



UNIVERSIDAD DE CÓRDOBA

## PROGRAMACIÓN WEB – BLOQUE I

# Introducción a la Programación Web

Dr. José Raúl Romero Salguero  
jrromero@uco.es



# Contenidos del Bloque

1. Introducción a la web
2. Fundamentos de Internet
3. Tipología de la web

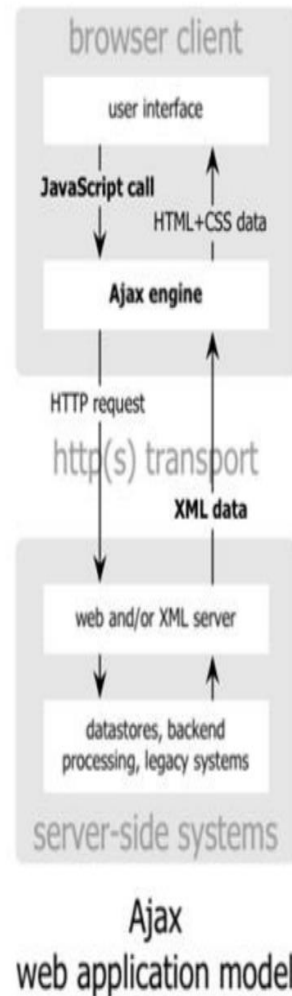
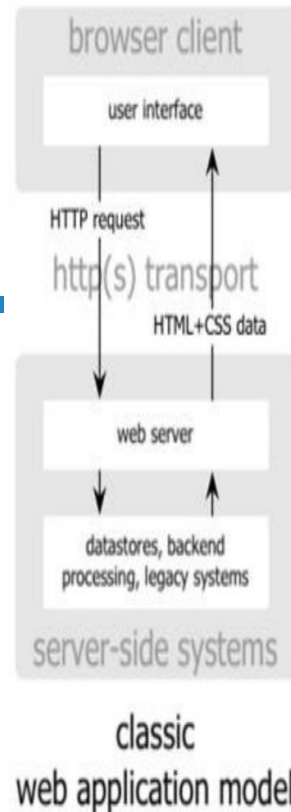


UNIVERSIDAD DE CÓRDOBA

## PROGRAMACIÓN WEB — TEMA I-2

# Fundamentos de Internet

Dr. José Raúl Romero Salguero  
jrromero@uco.es



# Contenidos

1. Internet como red\_ Protocolos
2. Protocolo HTTP

1.

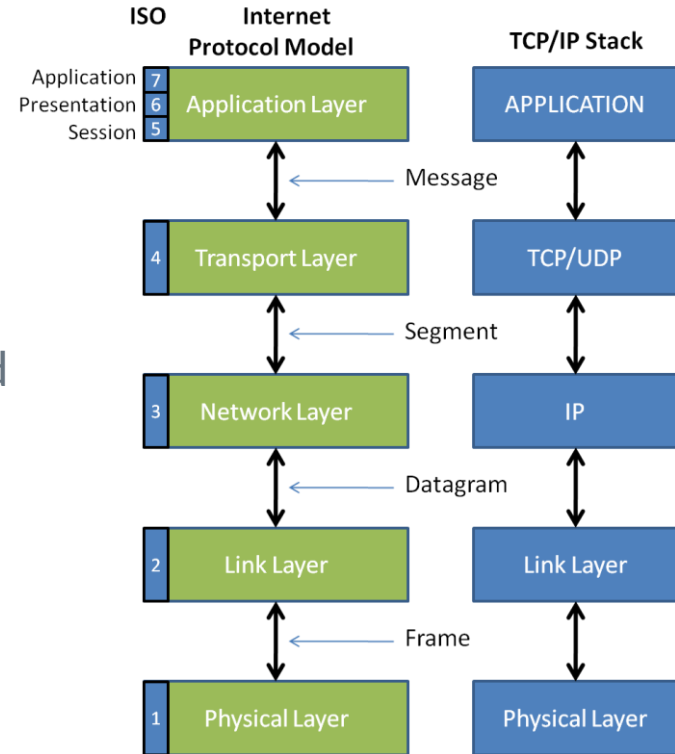
# Internet como red\_ Protocolos

Empecemos por las entrañas...

# Internet

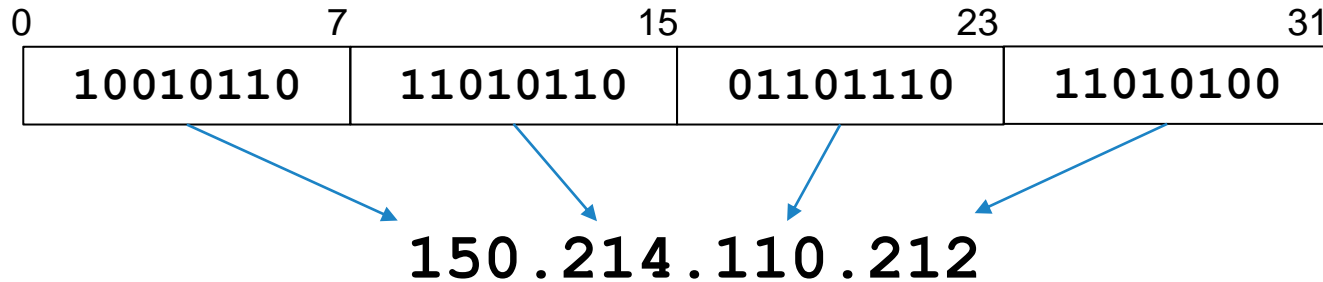
- ¡**Recordemos!** **Internet** es un conjunto de redes físicas heterogéneas, interconectadas mediante protocolo TCP/IP
- El uso de TCP/IP permite componer una red lógica de nodos a escala global
- Múltiples capas de protocolos de comunicación: IP → TCP/UDP → HTTP/FTP/POP/SMTP/SSH...

TCP: *Transmission Control Protocol*  
IP: *Internet Protocol*



# Internet Protocol (IP)

- Sencillo **protocolo de envío de datos** entre dos máquinas
- Cada dispositivo tiene una **dirección de 32 bits**, separada en 4 bloques de 8 bits (0-255) en **IPv4**



- IETF (*Internet Engineering Task Force*) e IANA (*Internet Assigned Numbers Authority*) tienen **reservadas direcciones IP** para fines concretos

# Internet Protocol (IP)

- Algunas direcciones IP reservadas de interés:
  - 0.0.0.0 – 0.255.255.255 : Rango de direcciones de origen
  - 10.0.0.0 – 10.255.255.255 : Direcciones de red privada
  - 127.0.0.0 – 127.255.255.255 : Direcciones de *loopback*

Otras IP reservadas para TEST-NET, multidifusión, multicast, etc.



# Protocolo TCP

- TCP garantiza la entrega de los mensajes sobre IP, en contraposición a UDP (*User Datagram Protocol*)
- Algunos programas (juegos, *streaming*) utilizan el protocolo UDP, que es más sencillo, ya que puede resultar asumible la pérdida de mensajes
- TCP/IP utiliza multiplexado\_\_ múltiples aplicaciones utilizando la misma dirección IP
  - Se asigna un número de puerto a cada programa o servicio, que en general es accedido mediante socket de red
  - RFC 6335 “Procedures for the Management of the Service Name and Transport Protocol Port Number Registry”:  
<https://tools.ietf.org/html/rfc6335>

# Protocolo TCP\_\_ Puertos

- Gestionados por la *Internet Assigned Numbers Authority* (IANA), fundación privada de EE.UU. perteniente a la ICANN, que asigna cada puerto a aplicaciones concretas
- Puertos de 0 a 1023 (0 a  $2^{10}-1$ ) se denominan **puertos de sistema**:
  - 21: FTP (file transfer protocol)
  - 22: SSH (secure shell)
  - 23: Telnet
  - 25: email
  - 53: DNS (domain name server)
  - 80: HTTP (hypertext transfer protocol)
  - 443: HTTP sobre TLS/SSL (HTTPS)
  - 990: FTPS (FTP sobre TLS/SSL)
  - 992: Telnet sobre TLS/SSL

# Protocolo TCP\_\_ Puertos

- Puertos de 1024 a 49151 ( $2^{10}$  a  $2^{14}+2^{15}-1$ ) son puertos registrados a demanda de las entidades o empresas que los reservan:
  - 1433: MS SQL Server
  - 1521: Oracle listener
  - 3306: MySQL
  - 3690: Subversion
  - 5400,5500,5600,5700,5800,5900: VNC (escritorio remoto sobre HTTP)
  - 8000: iRDMI (*Intel Remote Desktop Management Interface*) – Utilizado erróneamente como puerto HTTP, lo que supone un riesgo de amenaza
  - 8080: HTTP alternativo

Lista completa en Wikipedia:

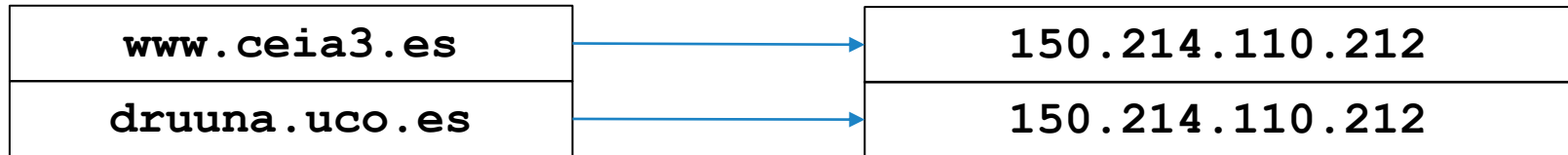
[https://en.wikipedia.org/wiki/List\\_of\\_TCP\\_and\\_UDP\\_port\\_numbers](https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers)

Buscador de puertos (¡indica amenazas!):

<http://es.adminsub.net/tcp-udp-port-finder>

# Domain Name System (DNS)

- Implementación de la **transparencia de nombrado** ofrecida por sistemas distribuidos
- Conjunto de servidores que **mapean nombres a direcciones IP**



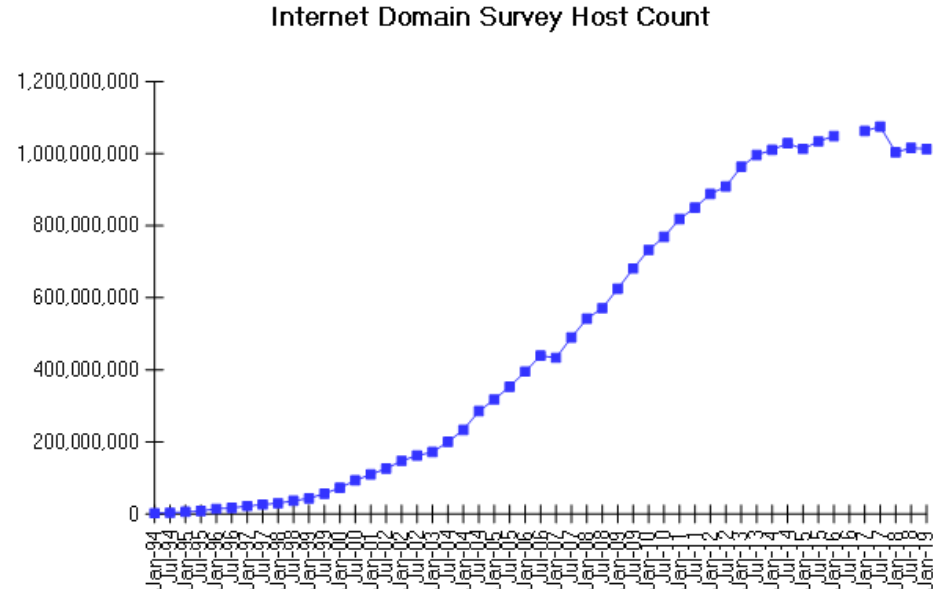
- El **servicio de registros Whois** permite consultar esta correspondencia. Por ejemplo: <http://whois.domaintools.com/>

# Domain Name System (DNS)

- Nuestras máquinas pueden guardar copias locales (caché) del fichero *hosts*
  - ❑ Sistemas Windows: `C:\Windows\System32\drivers\etc\hosts`
  - ❑ Sistemas MacOS: `/private/etc/hosts`
  - ❑ Sistemas Unix/Linux: `/etc/hosts`
  - ❑ Sistemas Android: `/system/etc/hosts`
- Conocer estos ficheros (texto plano) es útil para redireccionar dominios a local mientras se están desarrollando
  - ❑ ¡Mucha precaución con la edición de estos ficheros!
- Sistema susceptible a ataques (p.ej. redireccionando a *phishing*)

# Domain Name System (DNS)

- Enorme incremento de dominios:



- *Hobbes' Internet Timeline (Robert H. Zakon):*  
<https://www.zakon.org/robert/internet/timeline/>

Source: Internet Systems Consortium ([www.isc.org](http://www.isc.org))

# Uniform Resource Identifier (URI)

- Cadena de caracteres que sirve como localizador, nombre, o ambos
- Identificación no ambigua de un recurso concreto, definiendo un espacio de nombres separados de forma jerárquica
- **RFC 3986** “*Uniform Resource Identifier(URI): Generic Syntax*”: <https://tools.ietf.org/html/rfc3986>
- Utilizada para definir el **acceso único y no ambiguo de cualquier recurso**: direcciones de páginas web, datos o ítems de la web semántica, servicios y APIs, etc.

# Uniform Resource Identifier (URI)

`ftp://ftp.is.co.za/rfc/rfc1808.txt`

`ldap://[2001:db8::7]/c=GB?objectClass?one`

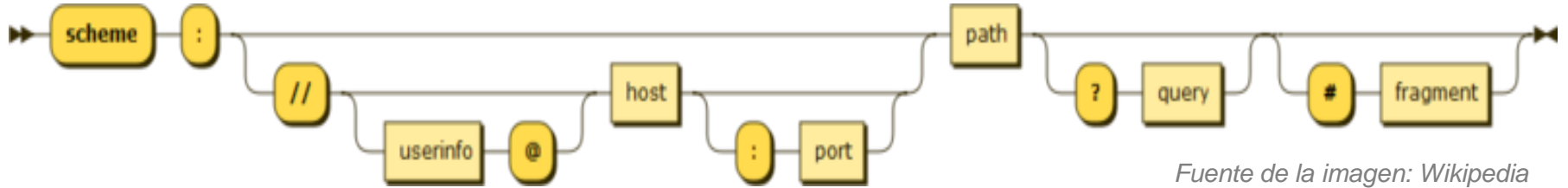
`mailto:John.Doe@example.com`

`telnet://192.0.2.16:80/`

`http://www.jrromero.net`

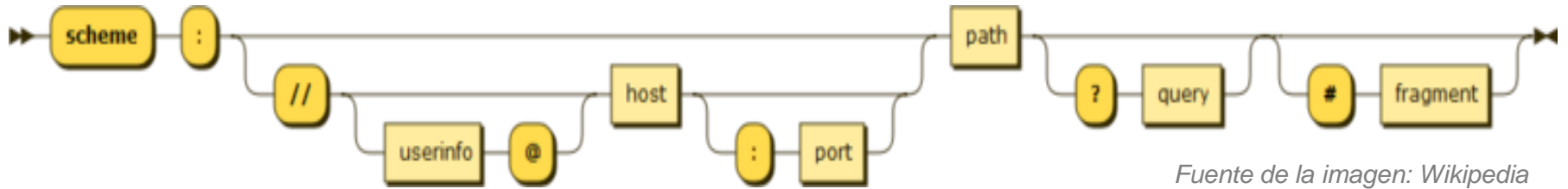


# Uniform Resource Identifier (URI)



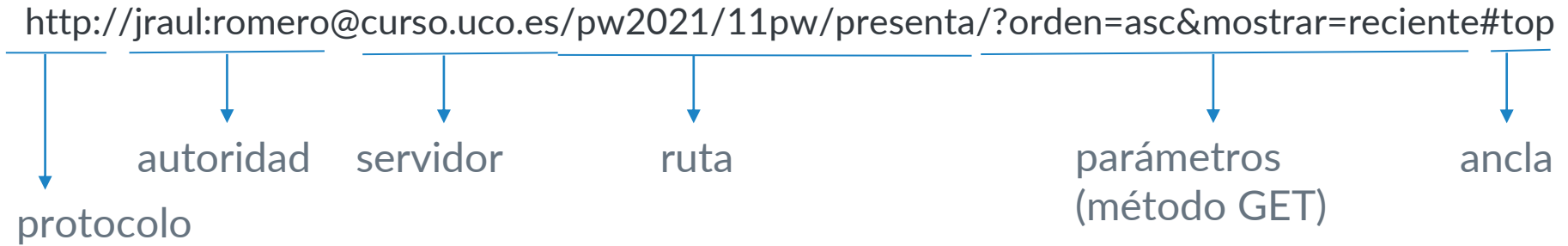
- El **esquema** se especifica como cadena de caracteres no vacía y minúscula (http, https, ftp, telnet, mailto, etc.)
- **Componente de autoridad** compuesto por:
  - ☐ usuario + ":" + *password*
  - ☐ *Host*: nombre de servidor o dirección IP en notación decimal con puntos
  - ☐ Número de puerto
- **Ruta** (*path*) separada por segmentos "/". Puede ser vacía y debe corresponderse con la ruta del sistema de ficheros (no necesariamente 1:1)

# Uniform Resource Identifier (URI)



- Un **componente de consulta** (*query*) con una cadena de consulta sobre datos no jerárquicos. No está estandarizado su formato, si bien es habitual la secuencia de tipo `"?clave1=valor&clave2=valor2"`
  - ☐ Utilizado en el caso de paso de parámetros por método GET.
  - ☐ Los caracteres están reservados como **delimitadores**: `:` `/` `?` `#` `[` `]` `@`
- Un **componente fragmento**, que contiene un identificador de un recurso secundario, como anclas (anchors) en secciones de HTML

# Uniform Resource Locator (URL)



- El **protocolo** determina el resto de la estructura
- El dominio del servidor es **mapeado por el DNS** a su dirección IP
- Al ser protocolo HTTP, el **puerto por defecto** (según ICANN) es 80
- La ruta finaliza en un directorio, por lo que el recurso invocado por defecto es determinado en la parte de servidor (`index.php`, `index.jsp`, `index.html`, ...)

2.

# Protocolo HTTP

¡No sólo es lo que escribimos en el navegador!

# Hypertext Transport Protocol (HTTP)

- Establece el conjunto de comandos interpretados por el servidor web
- Creado inicialmente por Tim Bernes-Lee en el CERN (1989)
- Actualmente, coordinado y desarrollado por **IETF** y el **W3C** (*World Wide Web Consortium*)
- La versión más reciente – HTTP/2 – se liberó en 2015
  - ❑ **RFC 7540** “Hypertext Transfer Protocol Version 2 (HTTP/2)”: <https://tools.ietf.org/html/rfc7540>
- La mayoría de los navegadores ya soportan HTTP/3 (*draft 34 el 02/02/2021*), que trabaja sobre TLS (*Transport Layer Security*) 1.2 o superior
  - ❑ **RFC 7301** “Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension”: <https://tools.ietf.org/html/rfc7301>

# Hypertext Transport Protocol (HTTP)

- HTTP permite diferentes tipos de peticiones (**métodos**)
- **Peticiones seguras** (sólo lectura) – no alteran el estado del servidor:
  - ❑ **GET** – Devuelve el recurso identificado por el *Request-URI*
  - ❑ **OPTIONS** – Retorna las opciones de comunicación disponibles por el servidor (*server capabilities*)
  - ❑ **HEAD** – Inspecciona la cabecera del recurso (no devuelve el *BODY*)
- **Peticiones idempotentes** – una solicitud idéntica puede realizarse múltiples veces, devolviendo siempre el mismo resultado:
  - ❑ **PUT** – Solicita al servidor guardar el cuerpo de la solicitud en la ubicación dada por la URL
  - ❑ **DELETE** – Solicita al servidor que elimine el recurso en la URL dada
  - ❑ Métodos seguros (GET, OPTIONS, HEAD)

# Hypertext Transport Protocol (HTTP)

- **Peticiones no idempotentes** – realizar múltiples peticiones idénticas podría causar efectos adicionales (p.ej. enviar varias veces una orden):
  - ❑ **POST** – Envío de información al servidor (p.ej. información de un formulario)

```
1  POST /cgi-bin/process.cgi HTTP/1.1
2  User-Agent: Mozilla/4.0 (compatible; MSIE5.01;
3  Windows)
4  Host: www.ceia3.es
5  Content-Type: text/xml; charset=utf-8
6  Content-Length: 88
7  Accept-Language: es-es
8  Accept-Encoding: gzip, deflate
   Connection: Keep-Alive
```

# Hypertext Transport Protocol (HTTP)

- Peticiones no idempotentes – (cont.)
  - ❑ **PATCH** – Solicita un conjunto de cambios (parciales) descritos en la entidad identificada por la *Request-URI*

Muy utilizado en el desarrollo de REST/APIs – no es necesario enviar la entidad completa, por lo que permite la actualización de un único campo, con el consecuente ahorro de ancho de banda

Definido en **RFC 5789** “*PATCH Method for HTTP*”:  
<https://tools.ietf.org/html/rfc5789>



# Hypertext Transport Protocol (HTTP)

- Muy importante para el programador web conocer las diferencias entre GET y POST:

## GET

- ☐ La cadena de consulta se envía en la URL del método (visible)
- ☐ Puede guardarse en caché
- ☐ Puede guardarse como marcador por el cliente
- ☐ Tiene restricciones de longitud
- ☐ Solicitud de solo lectura
- ☐ **NUNCA** enviar datos sensibles (p.ej. *passwords*)

## POST

- ☐ Datos enviados para creación/modificación de un recurso
- ☐ Nunca se guarda en caché (oculto al cliente)
- ☐ No puede guardarse como marcador
- ☐ No tiene restricciones en la longitud
- ☐ Permite adjuntos de distintos tipos MIME (*Multipurpose Internet Mail Extension*)



# Hypertext Transport Protocol (HTTP)

- Otras peticiones:
  - ❑ **CONNECT** – Utilizado por cliente para establecer una conexión sin cierre con el servidor
  - ❑ **TRACE** – Realiza una prueba de eco de retorno (muy utilizado para depuración, medición de latencias y desarrollo)

# Hypertext Transport Protocol (HTTP)\_\_

## Códigos de estado

- Los **códigos de estado** son identificadores y metainformación asociados a la respuesta (*Response*) HTTP recibida por el servidor:

- ❑ Definidos en **RFC 2616** “*Hypertext Transfer Protocol – HTTP/1.1*” (Sección 10): <https://tools.ietf.org/html/rfc2616#section-10>

- ❑ **Tipos de códigos:**

- **1xx** – Códigos informativos
- **2xx** – Códigos de éxito
- **3xx** – Códigos de redirección
- **4xx** – Códigos de error en cliente
- **5xx** – Códigos de error en servidor

100 – Continue

200 – OK

203 – Non-Authoritative Information

301 – Moved Permanently

404 – Not Found

408 – Request Timeout

500 – Internal Server Error

503 – Service Unavailable

# Hypertext Transport Protocol (HTTP) \_\_

## Tipos de medios (antes MIME)

- Forma estándar de indicar la naturaleza y formato de un recurso
- IANA es el organismo responsable de realizar el control y seguimiento de los tipos estándar
  - ❑ **RFC 6838** “Media Type Specifications and Registration Procedures”: <https://tools.ietf.org/html/rfc6838>
  - ❑ **RFC 2045** “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies”: <https://tools.ietf.org/html/rfc2045>
  - ❑ **RFC 2046** “Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types”: <https://tools.ietf.org/html/rfc2046>

# Hypertext Transport Protocol (HTTP) \_\_

## Tipos de medios (antes MIME)

- Los **tipos de medios** se dividen en categorías:
  - ❑ *application, audio, font, example, image, message, model, multipart, text, video*
  - ❑ El **listado completo** de tipos en IANA:  
<https://www.iana.org/assignments/media-types/media-types.xhtml>

- Ejemplos:

MIME type	extensión
text/html	.html
text/plain	.txt
image/gif	.gif
image/jpeg	.jpg
video/quicktime	.mov
application/octet-stream	.exe



# Programación Web

Introducción a la programación Web\_\_ Curso  
2021/22