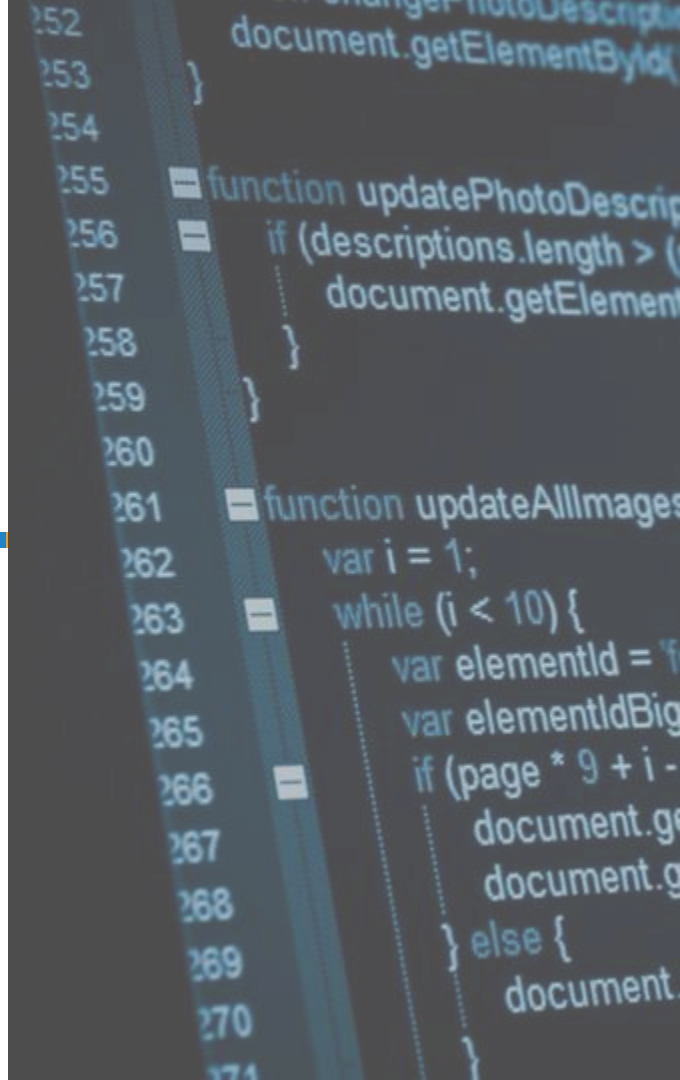
UNIVERSIDAD D CÓRDOBA

PROGRAMACIÓN WEB

Introducción y Perspectiva–101400



2.

Introducción a la programación web y Perspectiva histórica

Conceptos muy básicos

Pongamos cada término en su contexto

World Wide Web

- “La **World Wide Web** (WWW) es un espacio de información en el que los documentos y otros recursos web se identifican por una URL (*Uniform Resource Locator*), interconectados por enlaces de hipertexto, y accedidos a través de Internet."
- Año: 1989/1990
- Creadores: Tim Berners-Lee, Robert Cailliau
- En octubre de 2009, en la revista Times, Berners-Lee admitió que la doble barra inicial ("//") en las direcciones web era “innecesaria".

"There you go, it seemed like a good idea at the time"

Cliente fino / grueso / en la nube

- Clientes finos (*thin clients*):
  - Diseñados para ser de tamaño pequeño
  - Los datos se procesan en un servidor
  - Requiere una red **sin** unidad de disco duro (DEC VT100)
- Clientes gruesos (*fat/thick clients*):
  - Diseñado para realizar el grueso del procesamiento del comportamiento en aplicaciones cliente/servidor
  - Los datos son verificados por el cliente (validación inmediata)
  - No requiere comunicación continua con servidores y posee HDD.

Cliente fino / grueso / en la nube

- Clientes en la nube (*cloud clients*):
  - Diseñados como clientes finos
  - Requieren conectarse a plataformas de terceros para hacer uso de sus servicios
  - Propiciado por el abaratamiento de hardware HDD y SDD en granjas de servidores

“Internet“ y “Web“: ¡no son sinónimos!

Internet

- Una red física que permite la conexión de miles de millones de redes de redes conectándose a miles de millones de computadoras y otros dispositivos mediante protocolos TCP/IP para compartir y transmitir información

World Wide Web [Definición primigenia]

- Colección de documentos multimedia entrelazados__ páginas web almacenadas en dispositivos conectados y accedidos utilizando un protocolo común (HTTP)

Diferencias:

- Internet es hardware y protocolos__ WWW es software y protocolos
- La WWW es una aplicación usando Internet para transmitir la información

“Internet“ y “Web“: ¡no son sinónimos!

World Wide Web [Nueva conceptualización]

Una infraestructura que permite desarrollar, desplegar y utilizar fácilmente sistemas distribuidos

Sistemas distribuidos

Un sistema en el que sus componentes están localizados en computadores en red, comunicando y coordinando sus acciones mediante paso de mensajes, para obtener un objetivo común

Aspectos aprendidos aplicables a la programación web

- La programación web se basa en los navegadores web para renderizar su interfaz de usuario, codificada en HTML/CSS
- La programación web utiliza fundamentalmente el protocolo HTTP para el intercambio de información dentro de un sistema distribuido
- Las aplicaciones web utilizan una mezcla de computación en el lado del servidor y en el lado del cliente
- Las aplicaciones web pueden ser modificadas y desplegadas para todos los clientes instantáneamente
- Los programadores “tradicionales” tienen mayor flexibilidad y control sobre su contexto, sus pruebas y ejecución

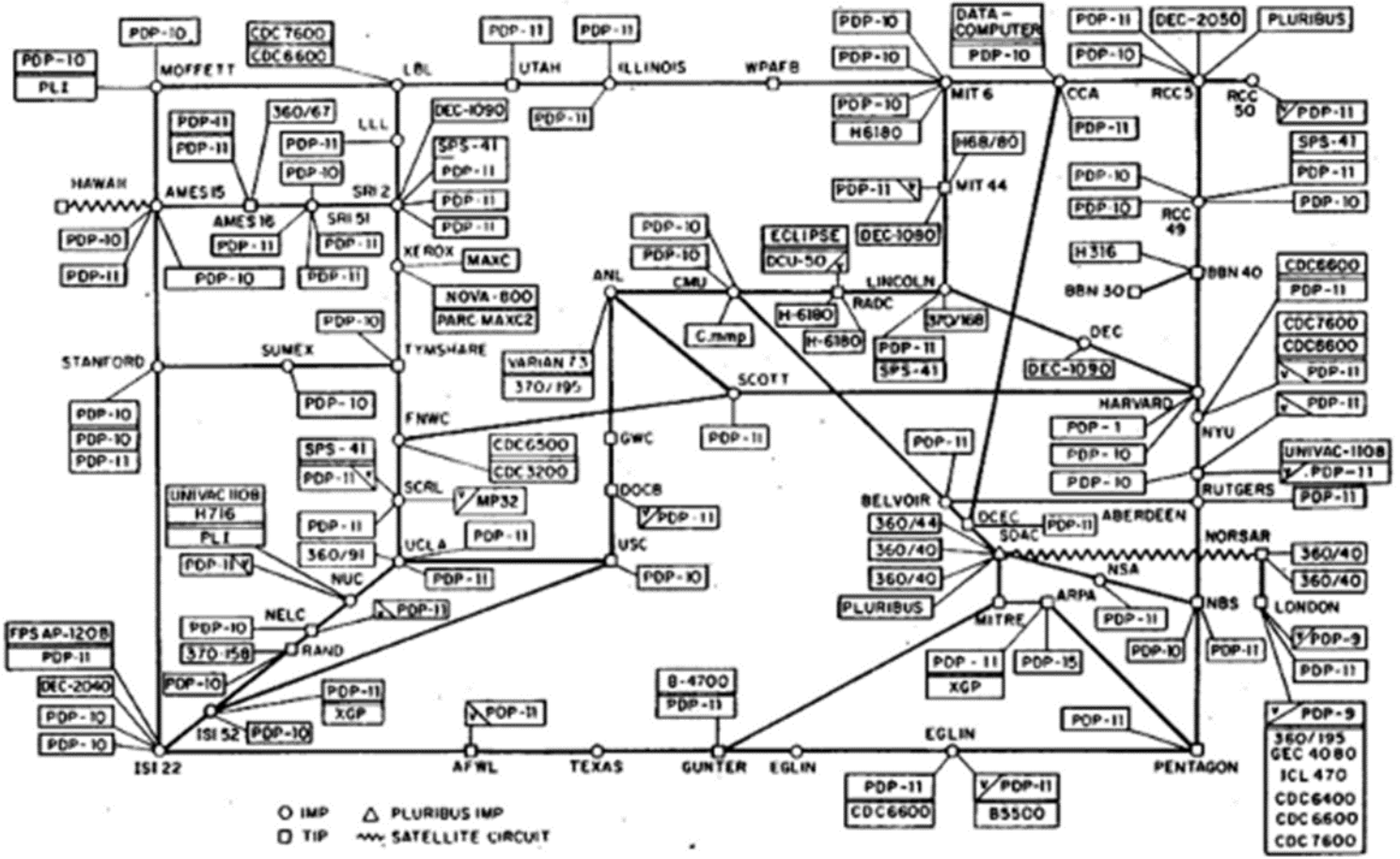
Historia de la web

Saber de dónde venimos para adivinar a dónde vamos

Historia

- ✓ 1960s: Se comienzan a utilizar CLI (*Command-Line Interface*)
- ✓ 1968: Demostración de Douglas Engelbart (primer ratón)
- ✓ 1969: ARPANET (precursor de Internet) con 64 nodos
- ✓ 1971: Envío del primer e-mail

ARPANET LOGICAL MAP, MARCH 1977



(PLEASE NOTE THAT WHILE THIS MAP SHOWS THE HOST POPULATION OF THE NETWORK ACCORDING TO THE BEST INFORMATION OBTAINABLE, NO CLAIM CAN BE MADE FOR ITS ACCURACY)

Historia

- ✓ 1971: Aparece el *File Transfer Protocol* (FTP)

- ✓ 1972: Aparece el primer virus informático, llamado Creeper, por Bob Thomas para DEC PDP-10 sobre ARPANET


IM THE CREEPER, CATCH ME IF YOU CAN!

Historia

- ✓ 1977: Criptografía de clave pública RSA
- ✓ 1977-79: Surge EPSS/SERCnet (primera red entre instituciones de investigación en UK)

¿Qué tenemos hasta los 80?

- Los datos se almacenan en un computador *mainframe*, que computa el comportamiento del sistema
- Las vistas se computan en el *mainframe* y sólo son visibles en los terminales
- El terminal recibe las entradas del usuario y las transmite al *mainframe*, que las traduce como instrucciones del sistema

Historia

- ✓ 1981: Se estandariza FTP sobre TCP
- ✓ 1982: Aparece el estándar TCP/IP
- ✓ 1983: Primer PC con GUI (*Graphical User Interface*), el Apple Lisa

- ✓ mediados 1980s: Red Janet entre instituciones de investigación en UK con ancho de banda de 2 Mbit/s y puntos de acceso a 64 kbit/s
- ✓ final 1980s: Se expande TCP/IP mundialmente

Historia

- ✓ 1991: Janet añade el servicio de IP
- ✓ 1994: Se lanza Yahoo!, directorio de Internet y portal de noticias
- ✓ 1994: Jeff Bezos funda Amazon
- ✓ 1995: Aparece la versión 1.0 de Netscape, el primer navegador comercial

Historia

- ✓ 1995: Se lanza Altavista, uno de los primeros motores de búsqueda de Internet con hasta 13M de búsquedas al día
- ✓ 1997: Primera versión de JavaScript y PHP
- ✓ 1998: Llega WWW a los dispositivos móviles
  Se funda Google
  Surge PHP 3

Crecimiento exponencial de la red

- Tremendo crecimiento de nodos en Internet
  - 1983: 562 Computadoras
  - 1993: 1.313.000 Computadoras
  - 1994: 2.217.000 Computadoras
  - 1996: 14.352.000 Computadoras

Comienzan a lanzarse aplicaciones y surgen los primeros navegadores para el gran público

- Problemas:
  - Información accesible pero difícil de encontrar
  - Acceso poco amigable (para usuarios expertos)
  - Co-existencia de múltiples protocolos y formatos distintos

Historia

- ✓ Inicios 2000s: Surge WAP 2.0 pero queda obsoleto por la llegada de GPRS (vs WAP) y HTML/Javascript (vs WML)
- ✓ 2004: Nacen Facebook, Gmail, Flickr y Vimeo
- ✓ 2005: Aparece Ajax
  Se lanza Youtube
- ✓ 2008: Se lanza Google Chrome
- ✓ 2013: Aparecen frameworks como React o Bootstrap

¿Qué caracteriza la era Post-PC?

- Los datos se almacenan en una granja de servidores (“nube” o “*cloud*”), que computa el comportamiento del sistema
- La vista se computa bien en el servidor o bien en el terminal
- El terminal recibe las entradas de usuario, que bien lo delega en la nube o bien lo traduce directamente a instrucciones del comportamiento del sistema

Historia

- **2010** → **Responsive Web Design** (Ethan Marcotte) → diseño adaptable por *CSS media queries*.
- **2011** → **Bootstrap** estandariza UI responsiva.
- **2012** → **TypeScript** aparece como superset tipado de JS.
- **2014** → **Service Workers** → base de las *Progressive Web Apps* (PWA).
- **2015** → Publicación de **ECMAScript 2015 (ES6)** → salto enorme en JS.
- **2016–2017** → **React, Angular, Vue** consolidan *frameworks* modernos.
- **2018** → **WCAG 2.1** actualiza accesibilidad.

Historia

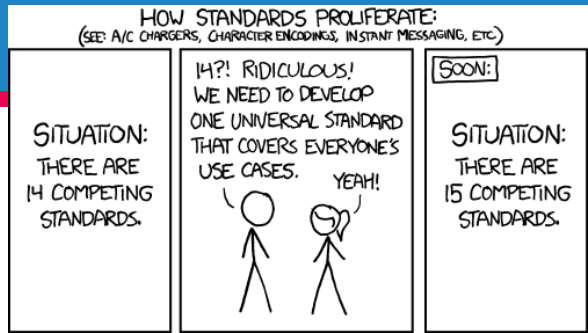
- **2020** → **Fin de Flash**; auge de **SPAs** y **arquitecturas *serverless***.
- **2021** → ***Core Web Vitals*** (Google) → rendimiento y UX pasan a ser factor de ranking.
- **2022** → **HTTP/3** comienza su adopción; **ES2022** introduce nuevas capacidades.
- **2023** → **WCAG 2.2** publicada (criterios adicionales de accesibilidad).
- **2024–2025** → Consolidación de **WebAssembly (Wasm)** y **Interoperabilidad (Interop/Baseline)** como referencia de compatibilidad.

¿Por qué es importante la sostenibilidad?

- **Consumo de electricidad**: Los centros de datos consumen alrededor del 1-2% de la electricidad global (unos 200-250 TWh) por año (*más electricidad que algunos países*)
  - Solo el consumo en minado de criptomonedas equivale al consumo de países como Argentina y Países Bajos
- **Refrigeración**: Supone hasta el 40% de la electricidad de un *data center*
- **Consumo intensivo de recursos**: El hardware necesita metales y minerales que requieren procesos de extracción dañinos para el medio ambiente
- **Residuos electrónicos**: Antiguos componentes acaban como basura electrónica, difíciles de reciclar y que pueden contener sustancias nocivas
- **Basura espacial**: Con proyectos como Starlink, de SpaceX, o Kuiper, de Amazon, que despliegan grandes constelaciones de pequeños satélites para dar cobertura a Internet, preocupa el aumento de basura espacial

Organismos de estandarización

¿Quién regula todo este caos?



¿Por qué usar estándares en Internet?

- Permiten una mayor accesibilidad para todos, ya que –de otra forma– los navegadores y sus empresas marcarían los estándares de uso
- Simplifican el proceso de desarrollo y mantenimiento en la web, incluyendo el intercambio y rotación de personal cualificado
- Validación del código: http://validator.w3.org
- Permiten compatibilidad hacia atrás:
  - Permiten que usuarios menos desarrollados puedan seguir haciendo uso de la WWW
  - Facilitan la viabilidad a largo plazo de los desarrollos presentes
- Ayudan a mejorar el SEO (*Search Engine Optimization*)

Interacción continua de *specs*

- Una ***spec*** (especificación técnica) es el documento que define cómo debe comportarse un estándar de la Web: qué APIs habrá, cómo se interpretan algunas construcciones, qué valores se permiten, etc.
- **Iteración continua** significa que estas especificaciones no están fijas, sino que evolucionan de forma constante: se corrigen, se extienden, se precisan, adaptan a nuevos casos de uso, nuevas necesidades de seguridad, accesibilidad, etc.

Principales organismos de estandarización en Internet

- **IETF** (*Internet Engineering Task Force*)
  - Desarrolla los protocolos de Internet en forma de RFC (*Request for Comments*) y BCP (*Best Current Practice*)
  - Desarrolla protocolos (HTTP/1.1, HTTP/2, HTTP/3 sobre QUIC, TLS, etc.)
- **W3C** (*World Wide Web Consortium*)
  - Publica recomendaciones y coordina grupos de trabajo (CSS WG, WebApps, etc.)
  - Accesibilidad: WAI y WCAG
  - Define estándares web como XML, HTML, CSS, etc.
  - Desde 2019, colabora con WHATWG: adopta “review drafts” de HTML/DOM

Principales organismos de estandarización en Internet

- **ICANN** (*Internet Corporation for Assigned Names and Numbers*)
  - Controla los nombres de dominio a alto nivel
  - **IANA** (*Internet Assigned Numbers Authority*) es la entidad dependiente de ICANN encargada de administrar y coordinar varios de los **identificadores globales de Internet:** DNS, Ips, registros de RFC (IETF)
- **WHATWG** (*Web Hypertext Application Technology Working Group*)
  - Mantiene **HTML Living Standard** y otras especificaciones base (**DOM**, **URL**, **Fetch**…).
  - Flujo abierto en GitHub, *issues* y *pull requests*.
  - Cambios pequeños y frecuentes → web evolutiva

Living Standard vs Snapshots

- **Living Standard (WHATWG)**
  - Especificación **única y continuamente actualizada**.
  - Cambios aterrizan tras discusión/consenso + pruebas
- **Snapshots (W3C y otros)**
  - **Instantáneas** revisadas de un estado concreto de una spec viva para el proceso formal (p. ej., **Review Drafts** de HTML) o **CSS Snapshot** que consolida módulos estables.
  - Útiles para referencias normativas, certificaciones, y planificación de fabricantes.

- Conocer el tipo de estándar al que se acoge la tecnología tiene **implicaciones para los desarrolladores**

Living Standard vs Snapshots

Living Standard (WHATWG)

1. “Capas de compatibilidad” y feature detection

- Como las especificaciones *living* (vivas) cambian de forma continua, **no puedes asumir que todo navegador soporte lo mismo al mismo tiempo**.
- En lugar de programar “a ciegas”, debes **detectar si una característica está disponible** antes de usarla (*feature detection*).

```
if ('clipboard' in navigator) {
  // Usa API moderna del portapapeles
} else {
  // Fallback: selecciona texto y copia manual
}
```

- Esto se llama “**capas de compatibilidad**”: el código se adapta a distintos niveles de soporte sin romperse.

Living Standard vs Snapshots

Living Standard (WHATWG)

2. Consultar compatibilidad (Baseline / Interop)

- **Baseline** (definido por Google, Mozilla, Igalia, etc.) te dice qué APIs ya son seguras de usar porque están presentes en los navegadores principales.
- **Interop** es un esfuerzo anual donde los fabricantes de navegadores acuerdan alinear la implementación de nuevas funciones.
- Como desarrollador, revisas esas fuentes (junto a MDN y caniuse) para saber:

  si la API ya está madura,

  si aún tendrás que dar soporte alternativo.

Living Standard vs Snapshots

Living Standard (WHATWG)

3. Pruebas WPT (*Web Platform Tests*)

- Son pruebas automatizadas de interoperabilidad que confirman si la implementación de una función sigue la especificación.
- No necesitas escribirlas tú, pero puedes consultarlas en **wpt.fyi** para comprobar si un cambio o una API concreta **funciona igual en todos los navegadores**.
- Esto te ayuda a anticipar fallos que no verías solo probando en tu navegador de desarrollo.

Living Standard vs *Snapshots*

Living Standard (WHATWG)

4. ***Polyfills***: con medida y con datos
   - Un *polyfill* es código que imita una función nueva en navegadores que aún no la tienen.
   - Son útiles, pero también tienen coste (rendimiento, mantenimiento, tamaño de *bundle*).
   - Por eso, la recomendación moderna es:
     - Sólo usar *polyfills* si realmente tus usuarios lo necesitan.
     - Decidir con datos (*analytics* de tu app, porcentaje de usuarios en navegadores antiguos).
     - Ejemplo: si el 98% de tus usuarios ya tienen soporte nativo para `fetch()`, quizá no necesitas cargar un *polyfill* de 5 KB para el 2% restante.

Accesibilidad

Accesibilidad significa diseñar y desarrollar sitios y aplicaciones de forma que **puedan ser usadas por todas las personas**, incluidas aquellas con **discapacidad** (visual, auditiva, motriz, cognitiva, etc.), o con **limitaciones temporales, tecnológicas o contextuales**.

- **WCAG 2.2** como referencia técnica de accesibilidad del contenido.
- **EN 301 549 / UNE-EN 301 549**: requisitos europeos (y españoles) de accesibilidad para TIC (web y apps).

https://www.etsi.org/deliver/etsi_en/301500_301599/301549/03.02.01_60/en_301549v030201p.pdf

- **España**: RD 1112/2018 (sector público) y **Ley 11/2023** (EAA) amplían obligaciones en productos/servicios digitales.

https://www.boe.es/buscar/doc.php?id=BOE-A-2018-12699

https://www.boe.es/buscar/doc.php?id=BOE-A-2023-7897

Y para finalizar…

Recursos

- WHATWG **HTML Living Standard** (versión multipágina y *dev edition*).
  Versión multipágina: https://html.spec.whatwg.org/multipage/
  Dev edition: https://html.spec.whatwg.org/dev/
- W3C **WCAG 2.2** (WAI) + recursos de “Understanding/Techniques”.
  https://www.w3.org/TR/WCAG22/
- IETF: **RFC** de HTTP (semántica y HTTP/3).
  HTTP Semantics (RFC 9110): https://www.rfc-editor.org/rfc/rfc9110
  HTTP/3 (RFC 9114): https://www.rfc-editor.org/rfc/rfc9114
- **CSS Snapshot** (estado del ecosistema CSS por módulos).
  https://www.w3.org/TR/css/

Recursos

- **WPT** y panel **wpt.fyi** para resultados *cross-browser*.
  Proyecto: https://web-platform-tests.org/
  Resultados cross-browser: https://wpt.fyi/

- **Interop/Baseline** (web.dev) para saber qué APIs están listas.
  Baseline overview: https://web.dev/baseline/
  Interop 2024 wrap-up: https://web.dev/blog/interop-2024-wrapup/

Hazlo tú mismo – 1

1. Abre el HTML Living Standard y navega a "*Navigation and session history*".
2. Abre el historial de commits de WHATWG/HTML.
3. Localiza uno de los cambios recientes (por ejemplo, sobre *Navigation API* o seguridad de *javascript*: en `navigation.navigate()`).
4. Responde:
   - ¿Qué problema resuelve? ¿Funcional, seguridad, interoperabilidad?
   - ¿Qué implicaciones tendría en tu código o en tests?
   - ¿Cómo sabrías si ya está implementado en tu navegador objetivo?
5. Entrega: breve comentario (3–5 frases) en el foro del curso con el enlace al commit o sección.

Hazlo tú mismo – 2

1. Exploración inicial: abre el RFC y localiza:
   - El título y la fecha de publicación.
   - La tabla de contenidos.
   - El significado de la etiqueta "*Standards Track*".
2. Navegación temática:
   - Busca en el índice la sección sobre códigos de estado HTTP.
   - Localiza:
     - El código 404 (Not Found).
     - El código 308 (Permanent Redirect).
   - Responde: ¿qué diferencia semántica hay entre 301 y 308 según el RFC?
3. Piensa un caso de aplicación real y responde en el foro de Moodle:
   - ¿Cuándo deberíais usar un 308 en lugar de un 301 en una web o API?
   - ¿Qué impacto tendría en clientes como navegadores o librerías HTTP?
   - ¿Fue sencillo orientarse por el RFC?

Programación Web

Introducción a la programación Web__ **Curso 2025/26**