

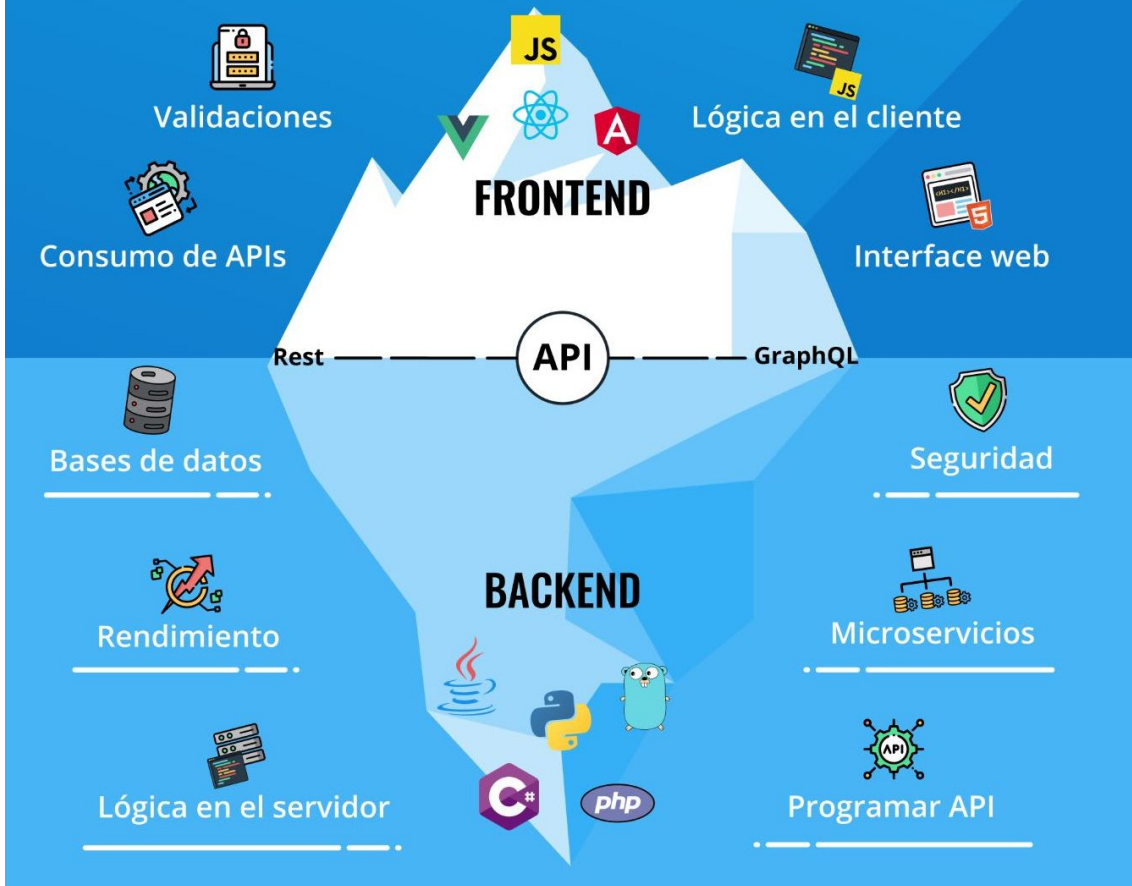
Desarrollo web en entorno cliente

Código: 0612 PROFESOR: FRANCISCO AGUILERA GONZÁLEZ

RESULTADOS DE APRENDIZAJE

1. Selecciona las arquitecturas y tecnologías de programación sobre clientes Web, identificando y analizando las capacidades y características de cada una.
2. Escribe sentencias simples, aplicando la sintaxis del lenguaje y verificando su ejecución sobre navegadores Web.
3. Escribe código, identificando y aplicando las funcionalidades aportadas por los objetos predefinidos del lenguaje.
4. Programa código para clientes Web analizando y utilizando estructuras definidas por el usuario.
5. Desarrolla aplicaciones Web interactivas integrando mecanismos de manejo de eventos.
6. Desarrolla aplicaciones Web analizando y aplicando las características del modelo de objetos del documento.
7. Desarrolla aplicaciones Web dinámicas, reconociendo y aplicando mecanismos de comunicación asíncrona entre cliente y servidor.

¿QUÉ ES BACKEND Y FRONTEND?



PACO

RAFA



Web 1.0

"Read Only",
Decentralized



Web 2.0

Participatory,
Centralized



Web 3

No Intermediaries,
Decentralized

1.0 Estáticas, casi sin interacción con el usuario

2.0 Las webs interaccionan con el usuario (parecen aplicaciones de escritorio, en parte a JS (Ajax, la gente lo flipa con Google Maps y desempolvan sus libros de JS) los datos suelen estar centralizados en uno o varios servidores

3.0 Datos descentralizados, aparecen tecnologías como los Blockchains, IA...

THE WEB 3.0 ECONOMY



BLOCKCHAIN



CRYPTO



NFTs



METaverse



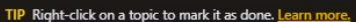
SMART CONTRACTS



DeFi



DECENTRALIZED STORAGE



```

graph TD
    WD[Web Development] --- WSB[Web Security Basics]
    WD --- WC[Web Components]
    WD --- TC[Type Checkers]
    WD --- SSR[SSR]
    WD --- PWAs[PWAs]
    WD --- PBP[Performance Best Practices]
    WD --- MA[Mobile Apps]

    WSB --- AS[Authentication Strategies]
    WSB --- T[Testing]
    AS --- JWT[JWT, OAuth, SSO, Basic Auth, Session Auth]

    WC --- HT[HTML Templates]
    WC --- CE[Custom Elements]
    WC --- SD[Shadow DOM]
    WC --- PRPL[PRPL Pattern]
    WC --- RAIL[RAIL Model]
    WC --- PM[Performance Metrics]
    WC --- LH[Using Lighthouse]
    WC --- DEV[Using DevTools]

    TC --- TS[TypeScript]
    TC --- AP[Apollo]
    TC --- RM[Relay Modern]
    TC --- GraphQL[GraphQL]

    SSR --- R[React]
    SSR --- A[Angular]
    SSR --- V[Vue.js]
    SSR --- S[Svelte]
    SSR --- NextJS[Next.js]
    SSR --- Astro[Astro]
    SSR --- RR[react-router]
    SSR --- NuxtJS[Nuxt.js]
    SSR --- SvelteKit[Svelte Kit]

    PWAs --- SSG[Static Site Generators]
    PWAs --- MA2[Mobile Apps]
    SSG --- NextJS2[Next.js]
    SSG --- Astro2[Astro]
    SSG --- Vuepress[Vuepress]
    SSG --- Eleventy[Eleventy]
    MA2 --- RN[React Native]
    MA2 --- Flutter[Flutter]
    MA2 --- Ionic[Ionic]

    PBP --- Storage[Storage]
    PBP --- NuxtJS3[Nuxt.js]
  
```

5 prioridades a la hora de escribir código (el orden influye)

1.- **Claridad** (no vale de nada un código que otra persona no puede entender, tiene que ser SIMPLE) Sino antes o después acabará en la basura.

2.- **Corrección** (tiene que hacer lo que tiene que hacer, en todos los casos y sin fallos) Es necesario escribir TEST. Los test te hacen mejor programador.

3.- **Eficiencia** (una vez cumplido lo anterior debemos intentar que sea rápido, use poca memoria...) no se debe diseñar pensando en la eficiencia.

4.- **Flexibilidad** (se pueda escalar, adaptar a nuevos problemas...)

5.- **Seguridad** (Cada vez es más importante. Estudiar Top 10 OWASP) El problema de la seguridad para el desarrollador es que el hacker solo tiene que encontrar un fallo de seguridad y el desarrollador tiene que taparlos todos, así que suerte :)