SEMANA 8

LECTURA SEMANAL – VISUALIZACIÓN DE DATOS EN LA WEB

Introducción a la Visualización de Datos

La visualización de datos es una herramienta fundamental en el mundo digital actual. Consiste en representar información compleja o abstracta mediante gráficos, diagramas, mapas u otros elementos visuales que facilitan su interpretación. En la web, este proceso se ha vuelto cada vez más importante gracias a las tecnologías que permiten mostrar estos datos de manera interactiva y dinámica.

Con herramientas como librerías y frameworks especializados, podemos integrar visualizaciones directamente en sitios web o aplicaciones, lo que permite a los usuarios explorar, filtrar y analizar información en tiempo real.

---

1. Uso de Frameworks Populares

Los frameworks son conjuntos de herramientas y bibliotecas predefinidas que facilitan el desarrollo de aplicaciones web. En el caso de la visualización de datos, existen varios frameworks y librerías populares que ayudan a crear interfaces interactivas y dinámicas.

Algunos ejemplos incluyen:

- **D3.js (Data-Driven Documents)**: Es una de las librerías más poderosas y versátiles para la visualización de datos. Permite manipular documentos basándose en datos, creando desde gráficos simples hasta visualizaciones altamente personalizadas.
- **Chart.js**: Ideal para quienes buscan rapidez y simplicidad. Ofrece gráficos como barras, líneas, tortas y más, fácilmente integrables en proyectos web.
- **Recharts y Victory**: Estas librerías están diseñadas para trabajar con React y ofrecen componentes reutilizables para construir gráficos modernos.
- **Vue + Viz.js / Plotly**: Integraciones útiles para desarrolladores que trabajan con Vue.js o que desean construir dashboards interactivos con datos complejos.

Los frameworks no solo aceleran el proceso de desarrollo, sino que también aseguran compatibilidad entre navegadores y dispositivos móviles, además de ofrecer opciones de estilo y animación listas para usar.

---

2. ¿Qué es una API?

API significa **Application Programming Interface** (Interfaz de Programación de Aplicaciones). Una API actúa como un intermediario entre dos sistemas, permitiendo que uno solicite información al otro de forma estandarizada.

En el contexto de la visualización de datos, las APIs juegan un papel crucial, ya que muchas fuentes de datos viven en servidores remotos. Por ejemplo, si queremos mostrar estadísticas del clima en tiempo real, usamos una API que nos brinda esa información en formato estructurado (como JSON o XML), para luego procesarla y mostrarla visualmente.

Algunas características clave de las APIs:

- Permiten acceso controlado a datos externos.
- Se pueden consumir desde cualquier lenguaje de programación que soporte solicitudes HTTP.
- Existen APIs públicas (gratuitas o de pago) y privadas (restringidas).

Ejemplos de APIs útiles para visualización de datos:

- OpenWeatherMap (clima)
- CoinGecko (criptomonedas)
- World Bank Data (estadísticas globales)
- NASA API (datos espaciales)

---

3. Librerías Populares de Consumo de Datos

Una vez que entendimos qué es una API, debemos aprender cómo consumirla. Para ello, contamos con varias librerías que facilitan la comunicación entre nuestro código y el servidor que provee los datos.

Las más comunes son:

- **Fetch API**: Es una interfaz incorporada en JavaScript moderno que permite realizar solicitudes HTTP (GET, POST, PUT, DELETE, etc.) de forma nativa sin necesidad de librerías externas.
- **Axios**: Una librería de terceros muy popular que ofrece una sintaxis limpia y sencilla para hacer peticiones HTTP. Tiene ventajas sobre Fetch, como el manejo automático de errores y la conversión automática de respuestas a JSON.
- **PapaParse**: Especializada en leer archivos CSV (Comma Separated Values), muy útil cuando los datos vienen en ese formato.
- **Lodash / Ramda**: Son librerías de utilidades funcionales que permiten limpiar, transformar y organizar los datos antes de visualizarlos.

El uso de estas herramientas nos ayuda a obtener, procesar y preparar los datos para que puedan ser utilizados por nuestras visualizaciones.

---

4. ¿Qué es p5.js?

p5.js es una **librería de JavaScript** orientada al arte y a la creación visual. Está basada en Processing, un entorno de programación creado originalmente para artistas, diseñadores y educadores.

Esta herramienta permite dibujar figuras, animaciones y gráficos directamente en el navegador, lo cual la convierte en una opción excelente para proyectos educativos, artísticos o experimentales.

Características principales:

- Fácil de aprender y usar, especialmente para principiantes.
- Permite crear gráficos interactivos con poco código.
- Soporta entrada de usuario (ratón, teclado, sensores).
- Funciona directamente en el navegador sin necesidad de plugins.

Un ejemplo común es crear visualizaciones de datos en tiempo real, como gráficos circulares que cambian según valores de sensores o APIs.

Sitio oficial: https://p5js.org/

---

5. ¿Qué es Three.js?

Three.js es una **librería de JavaScript** para crear gráficos 3D en el navegador utilizando WebGL. Es ideal para desarrolladores que quieren crear escenas interactivas tridimensionales sin tener que escribir código de bajo nivel de WebGL.

Ventajas:

- Simplifica enormemente la creación de objetos 3D.
- Ofrece múltiples formatos de geometría, materiales y luces.
- Soporta modelos importados desde software de diseño 3D.
- Es compatible con la mayoría de los navegadores modernos.

Gracias a Three.js, hoy en día es posible integrar visualizaciones 3D en páginas web, simulaciones científicas, juegos ligeros o incluso experiencias inmersivas como parte de dashboards de análisis de datos.

Sitio oficial: https://threejs.org/

Conclusión

La visualización de datos en la web no solo mejora la experiencia del usuario, sino que también permite una mejor toma de decisiones al convertir grandes volúmenes de información en algo comprensible y atractivo visualmente.

Hoy contamos con herramientas poderosas como **frameworks**, **APIs**, **librerías de consumo de datos**, y entornos creativos como **p5.js** y **Three.js**, que nos permiten construir soluciones completas, desde gráficos simples hasta escenarios 3D interactivos.

Dominar estos conceptos te abre la puerta a crear aplicaciones modernas, dashboards profesionales y experiencias visuales innovadoras.

🔗 Recursos Adicionales:

- D3.js: https://d3js.org/
- Chart.js: https://www.chartjs.org/
- Axios: https://axios-http.com/
- PapaParse: https://www.papaparse.com/
- Lodash: https://lodash.com/
- Three.js: https://threejs.org/

- p5.js: https://p5js.org/
- Documentación Fetch API: https://developer.mozilla.org/es/docs/Web/API/Fetch_API
