Reto 1 – Servidor Web Siemens

Alaitz Candela
Raul Melgosa
Daniel Tamargo

ÍNDICE

RWD

Nuestro diseño de páginas (layout, maquetación, enfoque) se basa en la filosofía **RWD** (Responsive Web Design).

Nuestra página no sigue la metodología mobile first, hemos considerado que el objetivo del proyecto estaba enfocado a manejar el tranvía desde dispositivos como tablets y ordenadores. Igualmente, a través de las distintas **media queries** hemos contemplado todos los tamaños para que también pueda ser utilizado desde dispositivos móviles.

Para seguir el método **RWD**, hemos sido conscientes del objetivo del proyecto y de la información que queríamos dar prioridad en la página.

Al modificar el tamaño del navegador, se puede observar un **diseño web fluido**, teniendo en cuenta dimensiones basadas en porcentajes (fluido), ajustándose a los anchos y tamaño de la pantalla, tamaño de fuente basado en **em** (elástico), etc.

CSS minify

Uno de los principales problemas al desarrollar páginas web es el tamaño que terminan ocupando.
De hecho, como la memoria del PLC con el que trabajaba el equipo de ARI era muy pequeña, hemos creado una versión mini (o lite) de nuestra página con solo los elementos necesarios y vitales para controlar el tranvía (index.html y tranvia.html).

Para ganar espacio, hemos reducido el tamaño de nuestro fichero CSS quitándole caracteres inútiles como espacios, tabulaciones y saltos de línea que al fin y al cabo, también ocupan espacio.

Para ello hemos utilizado la página **CSS Minifier**.

Nota: en el repositorio del proyecto mantenemos el fichero css antes de minimizarlo para que su contenido sea legible.

Documentación código JavaScript

Para seguir unas buenas prácticas de la programación, se han documentado los métodos (o la mayoría) donde se ha proporcionado una pequeña descripción del objetivo, parámetros de entrada y valor de respuesta (si fueran necesarios).

Ejemplos:

```
898   /**
899    * Quita la clase seleccionado a los demás botones y se la añade a él mismo
900    * @param {element} boton_seleccionado botón al cual añadirle la clase seleccionado
901    */
902   function cambiarColorBotonSeleccionado(boton_seleccionado) {
903       let nombre_clase_selected = 'graph-selector-button-selected';
904
905       // Animación de deseleccionar los demás botones (eliminar la clase)
906       // Aunque solo debería existir uno, cogemos todos y lo ejecutamos en todos
907       let botones_previamente_seleccionados = document.getElementsByClassName(nombre_clase_selected);
908       if (botones_previamente_seleccionados.length > 0) {
909           botones_previamente_seleccionados[0].classList.toggle(nombre_clase_selected)
910       }
911
912       // Añadimos la clase selected al botón clicado
913       boton_seleccionado.classList.toggle(nombre_clase_selected);
914   }
```

```
643   /**
644    * Función que devuelve la comprobación que realizará el filtro del array
645    * Si recibe el dato seleccion_parada, filtrará también por parada, si no, filtrará solo por mes y año
646    * @param {int} anyo año a buscar
647    * @param {int} mes  mes a buscar
648    * @param {string} parada  parada a cotejar con la selección de la parada
649    * @param {string} seleccion_parada  parada seleccionada para filtrar
650    * @returns comprobación para el filtro
651    */
652   function filtrarDatosGraficaMesAnyo(anyo, mes, parada, seleccion_parada, tipo) {
653       if (tipo == 4) {
654           return anyo == seleccion_anyo && parada == seleccion_parada;
655       } if (tipo >= 1 && tipo <= 3) {
656           if (seleccion_parada != undefined && seleccion_parada != null && seleccion_parada != '' && PARADAS.includes(sel
657               return anyo == seleccion_anyo && mes == seleccion_mes && parada == seleccion_parada;
658           }
659       }
660       return anyo == seleccion_anyo && mes == seleccion_mes;
661   }
```

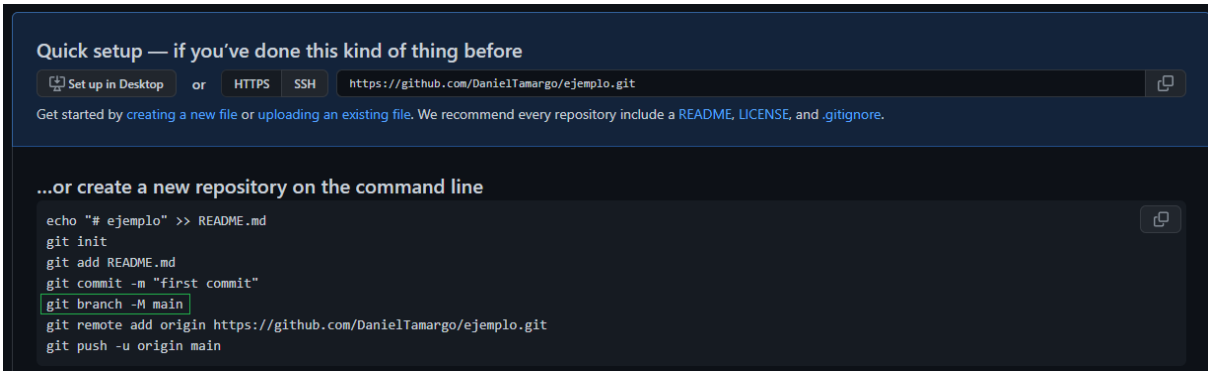
Al comentar los métodos no solo nos ayudamos a nosotros mismos de cara al futuro, si no también a otros desarrolladores o compañeros que puedan acabar modificando y/o necesitando nuestro código.

Metodología GIT

Hemos basado nuestro flujo de trabajo conjunto con git en la metodología Main-Develop.

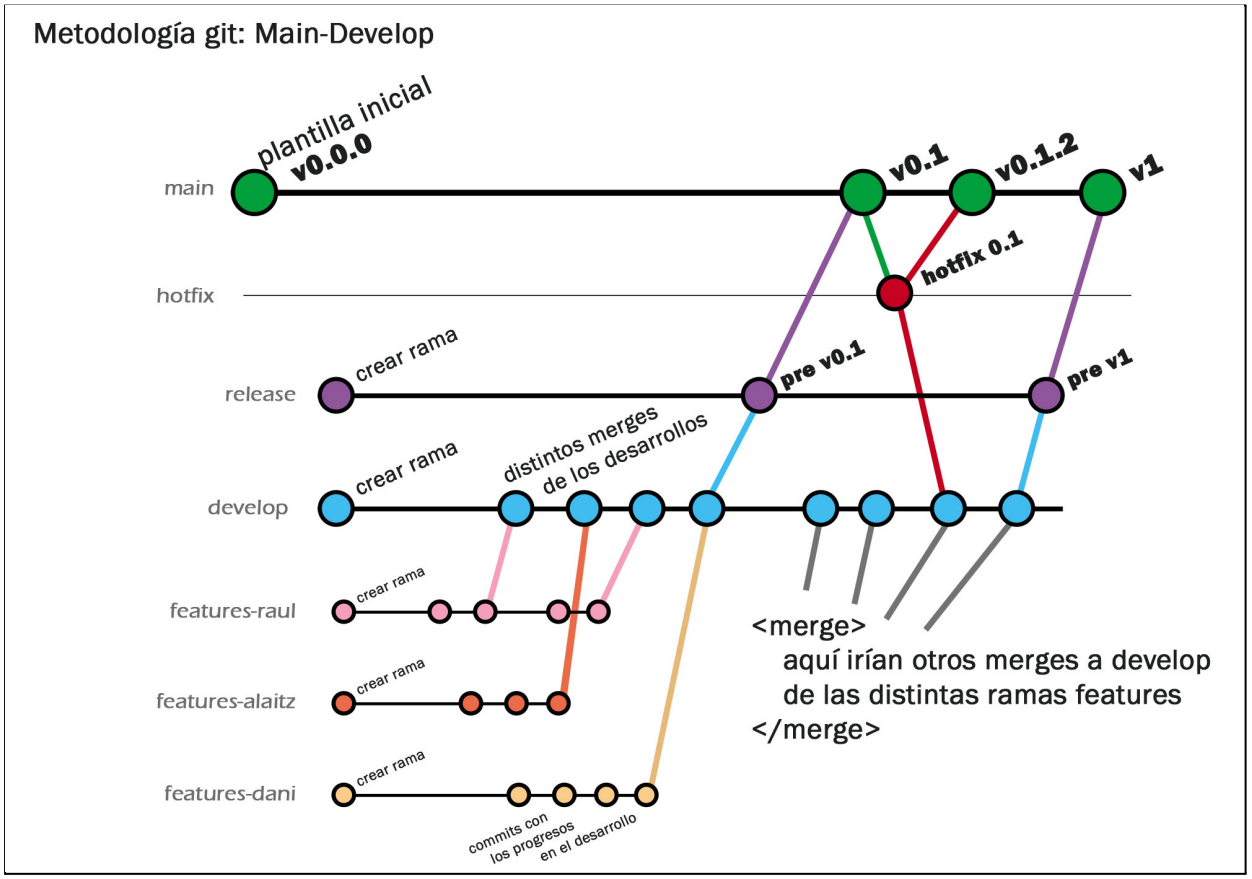
¿Por qué Main y no Master?
El concepto de Master surge del sistema de nombres master/slaves (maestro/exclavos), en 2020 estos términos llamaron la atención de gran parte del mundo y empresas como GitHub, empatizando con el motivo, comenzaron a sugerir el renombrar la rama master a main nada más crear el repositorio.

Esto se puede observar en los propios comandos sugeridos por GitHub al crear un nuevo repositorio:



[Enlace a post de **Genbeta** que habla al respecto.]

Diagrama de nuestra metodología Git:



Trabajar con ARI

El equipo de Automatización de la Robótica Industrial necesita que desarrollemos una página web donde podamos generar un sistema de control a modo de **HMI** (Human Machine Interface). De esta manera podrán controlar el tranvía, desplazándolo de una posición a otra.

Existirán dos modos de hacerlo:
- **El modo directo**, el cual desplazará el tranvía a una parada concreta.
- **El modo distancia o por milímetro**s, el cuál desplazará el tranvía a una distancia fija, pudiendo ser esta cualquier distancia dentro del eje donde puede moverse el tranvía.

Como el equipo de ARI dispone de 9 paradas, nuestro equipo ha designado una serie de nombres a cada parada, de cara a mostrarlos en las pestañas Información y Estadísticas. Las paradas serán:

```
const PARADAS = ['Ibaiondo',  'Landaberde', 'Lakuabizkarra', 'Wellington', 'Txagorritxu', 'Euskal Herria',
                 'Honduras', 'Europa', 'Sancho El Sabio'];
```

A parte del proceso de realización del diseño y desarrollo de la página, hemos tenido que sincronizar (de la mejor manera posible) nuestro trabajo con los de ARI, comunicándoles cualquier cambio a realizar de la misma manera que ellos nos hacían llegar sus modificaciones, variables que necesitaban y progreso de su trabajo.

Para ello, cuando era necesario se concertaban reuniones para comentar los nuevos cambios, progresos y problemas encontrados.

Estadísticas

Para darle un poco más de profundidad y/o realismo al proyecto, hemos añadido y desarrollado la página de estadísticas.

El objetivo de esta es **informar sobre datos** como número de pasajeros, revisores, incidentes, pasajeros que no pagaron y se escaparon y pasajeros que no pagaron pero fueron atrapados.

Para poder simular una serie de datos sobre los cuales basar las gráficas a mostrar en la propia página de estadísticas, se han generado una serie de registros aleatorios.

La forma de generarlos ha sido la siguiente:
Por cada parada se generan una serie de datos
Archivo app-estadisticas.js Método inicializarDatos()

Los datos se generan por año y mes, empezando en Enero de 2020 y acabando en el año y mes actual. Se trata de una serie de números aleatorios
Archivo app-estadisticas.js Método inicializarDatosParada()

De esta manera, se devuelve un array que contendrá todos los datos, el cuál será leído de distintas formas dependiendo del tipo de **gráfica** se quiera mostrar, es decir, dependiendo de los datos que se quieran mostrar en la gráfica que se está mostrando en ese momento.
Archivo app-estadisticas.js Método cargarDatosGrafica(seleccion_parada)

Al cargar la página, o al cambiar de una gráfica a otra, se 'pintará' la gráfica cargando y volcando los datos deseado
Archivo app-estadisticas.js Método pintarGrafica()

Para la creación de las gráficas se ha utilizado la librería **D3.js**, una librería muy completa y potente con la cual se pueden realizar algunas funciones de selección y modificación similares a jQuery, pero siendo su principal objetivo el manejo de datos para plasmarlos gráficamente de formas muy visibles.

A pesar de todo el trasfondo que puede darse con esta librería y la enorme utilidad que tiene, nosotros solo hemos raspado la superficie, trabajando de una forma simple y superficial y logrando una serie de gráficas sencillas, debido a que hemos considerado que era lo ideal para este reto.

Bibliografía

Principal (selección de colores y enfoque guía de estilo):
- EuskoTren
- Firefox brand

Git:
- Por qué renombrar de master a main
- Documentación Git

Documentación JavaScript
- Funciones arrays
- Funciones sets/maps
- jQuery
- jQuery – Ajax
- D3.js (gráficas)
- Siemens PLC JavaScript

Iconos:
- Flaticon

Salvación:
- StackOverFlow (infinitos posts)

Además, se han consultado los apuntes de JavaScript, CSS y Git proporcionados en las distintas asignaturas.