Vamos a aprender las fases de la progamación

Fases de la programación

El objetivo principal de este ejercicio es aprender a pensar y estructurar las fases de un desarrollo. Así que vamos a explicarlas:

Fase 1: Sacar papel y rotulador

Para pensar lo que tenemos que desarrollar no utilizamos el código. Usamos el código para ejecutar las ideas que hayamos pensado.

Para pensar lo que tenemos que hacer es dibujar y escribir en un papel y contarle a nuestro pato de goma lo que queremos hacer.

Fase 2: Identificar las acciones

Para saber qué acciones hay, nos tenemos que preguntar:

- ¿Qué acciones tienen que pasar cuando arrancamos la página?
- ¿Qué acciones puede hacer la usuaria?
- ¿Qué acciones pueden producir que "finaliza" la página?
- ¿Qué acciones hay que hacer cuando "finaliza" la página?

Pues tenemos que apuntar esas acciones en el papel.

Fase 3: Identificar elementos del DOM y los datos

En cualquier página las acciones nos producen y generan datos que vamos a tener que manejar y/o mostrar en la página, en consola, enviar un servidor...

Por lo general los elementos de una página tienen que tener sus correspondientes datos en JS. Una acción va a producir un cambio en estos datos y un cambio en estos datos va producir un cambio en los elementos del DOM de la página.

Para saber qué elementos y datos hay, nos tenemos que preguntar:

- ¿Qué elementos hay en la página que cambian durante la vida de la página?
- ¿Qué datos necesito manejar para que dichos elementos cambien?
- ¿De qué tipo son estos datos? ¿Son número, textos, booleanos, objetos, listados de números, listados de textos, listados de objetos...?

Pues tenemos que apuntar esos datos en el papel.

Fase 4: Pintar el diagrama de flujo

Ahora que ya tenemos identificadas todas las cosas que intervienen en nuestra página vamos a pintar el diagrama de flujo que las relaciona.

- En la parte superior del diagrama pintamos las entradas de nuestro flujo:
    - Estas entradas son las acciones o eventos que se producen al arrancar la página.
    - Estas entradas también son las que puede hacer la usuaria.
    - Describimimos la acción, por ejemplo "Pulsar botón X", "Arrancar la página", "Escribir en el campo X"...
    - Las pintamos en rectángulos.
- Debajo de cada una de cada rectángulo de las entradas pintamos cada una de las siguientes acciones que se producen:
    - Por ejemplo "Generar número aleatorio", "Leer dato desde el input X", "Borrar datos del formulario"...
    - Estas acciones tienen que ser lo más pequeñas posibles. Dicho de otro modo, tienen que ser acciones que no podamos subdividir en otras acciones.
    - También las pintamos en rectángulos.
    - Podemos pintar tantas acciones como necesitemos.
    - Algunas de estas acciones deben ser las que hemos identificado en la fase 2 como **acciones hay que hacer cuando "finaliza" la página**.

Fase 5: Emparejar datos y elemntos del DOM con acciones

En la fase 3 hemos identificado los datos y elementos del DOM con los que vamos a trabajar. En la fase 4 hemos pintado en el flujo las acciones que hay en nuestra página.

Ahora vamos a escribir dentro de cada acción del flujo los datos y elmentos que intervienen, es decir, los datos y elementos que necesitamos leer y los datos y elementos que tenemos que modificar.

Ya tenemos el flujo completo, pero a lo mejor lo podemos simplificar.

Fase 6: Simplificar el diagrama de flujo

Ahora el flujo ya está completo. Pero es posible que haya cajas que hacen las mismas cosas. Vamos a intentar identificar esas cajas similares y las vamos a uninificar en una sola con el objetivo de simplificar el flujo.

Ya hemos terminado el flujo. Ahora toca programar.

Fase 7: Programar cada una de las cajas (pero con datos falsos)

- Las cajas que están en la parte superior del diagrama se suelen corresponder con eventos, por ello tenemos que escribir en el código los `addEventListeners` y sus correspondientes funciones manejadoras de esos eventos.
- También vamos a crear una función por cada caja que hayamos pintado en el flujo.
- Al final tendremos una función para cada caja del flujo.
- Vamos a darle buenos nombres a las funciones que describan lo que hacen.
    - Si una función no hace lo que dice o la función está mal o el nombre está mal.
- NO vamos a programar escribir los parámetros que reciben todas estas funciones (todavía).
- Cuerpo de la función:
    - NO vamos a programar el cuerpo de la función (todavía).
    - Solo vamos a escribir en el cuerpo de la función un `console.log` para pintar en consola lo que hace la función. Por ejemplo `console.log('Genero un número aleatorio y lo retorno');`.

    - Si alguna de las cajas que hemos pintado es un rombo, es decir es una bifurcación, vamos a programar lo siguiente `if (true) { console.log('Genero un número aleatorio y lo retorno) ; } else { console.log('Leo un número desde un input y lo devuelvo'); }`.
- Vamos a ejecutar funciones desde dentro de unas funciones para enlazalar unas cajas del flujo con otras.

Una vez hecho todo esto al pulsar en un botón o realizar una de las acciones de entrada, debería aparecer en consola un mensaje por cada una de las cajas. Es decir en consola se estará pintando el flujo.

Si en nuestro flujo hay bifurcaciones (rombos) tenemos que cambiar los `if (true) {...} else {...}` de antes por lo mismo pero con `false`. Volvemos a pulsar en el botón y se volverá a pintar en consola el flujo pero esta vez con otros mensajes.

Tenemos que veficiar que lo que se pinta en consola es lo mismo que hemos pintado en nuestro flujo en papel.

Ya tenemos el flujo programado!!!

Fase 8: Programar cada una de las cajas (esta vez de verdad)

Una vez que ya estamos seguras que nuestro flujo está bien programado nos podemos centrar en programar bien cada una de las funciones. Vamos a repetir los siguientes pasos para cada función:

- Vamos a escribir los parámentros que recibe la función.
- Vamos a programar la función.
- En el caso de que la función deba retornar algo, vamos a programar el `return`;
- Para probar nuestra función nos preparamos las constantes, variables y elementos del DOM que la función necesita leer y modificar:
    - Si la función tiene que hacer una cosa cuando un input X está vacío ponemos ese input vacío y ejecutamos la función.
    - Si la función tiene que hacer otra cosa diferente cuando ese input X tiene un número ponemos en ese input un número y ejecutamos la función.
    - Si la función tiene que hacer otra cosa diferente cuando ese input X tiene un número negativo ponemos en ese input un número negativo y ejecutamos la función.
    - Hacemos lo mismo con las constantes y variables externas a nuestra función.
    - En definitiva nos preparamos las diferentes situaciones o escenarios que necesita nuestra función. Y la probamos en todos ellos.
    - Es muy fácil depurar lo que hace una función cuando controlamos totalmente los datos que recibe y que necesita.

Esta forma de programar es útil porque:

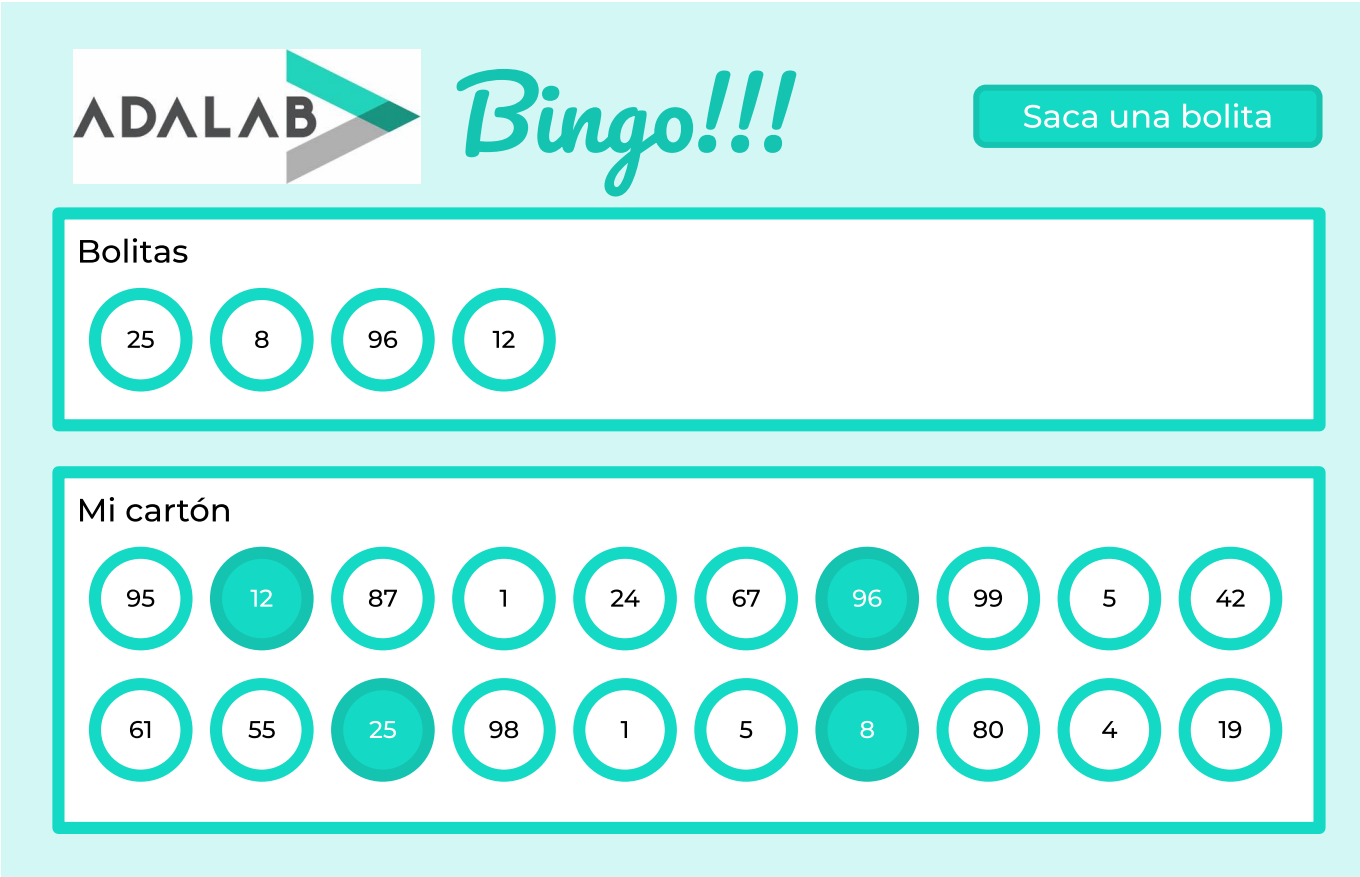
- Nos aislamos y concentramos en una parte pequeña de nuestra página.
- Es fácil de depurar.
- En caso de error es fácil indentificar si el problema es de esta función o es externo.
- Programamos la función independientemente de qué otra función la ejecute.
- Programamos la función independientemente de cuándo se ejecute.
- Programamos la función independientemente de datos aleatorios que no controlamos fácilmente.

Cuando terminemos de programar todas las funciones nuestra web, la página estará funcionando a la perfección.

Estas 8 fases las tenemos que hacer durante las primeras semanas de nuestro aprendizaje. En cuanto lo hagas 3 veces te acostumbras y te puedes ir saltando fases.

Vamos a echar una partidita de bingo casero!!!

Para ver que hemos entendido las fases de la programación vamos a programar un bingo!



- Al arrancar la página:
  - La parte superior "Bolitas" debe aparecer vacía.
  - En la parte inferior "Mi cartón" deben aparecer 20 números generados aleatoriamente entre el 1 y el 100.
- A continuación:
  - Cada vez que pulsemos el botón "Saca una bolita" debemos generar y mostrar un nuevo número aleatorio del 1 al 100 en la parte "Bolitas".
  - Y destacar con un fondo verde el número de nuestro cartón que coincida con la bolita sacada. Si no coincide pues nada.
- Cuando hayan aparecido todas las bolitas de los números de nuestro cartón debemos mostrar el mensaje "Han cantado bingo!!!" y ocultar el botón "Saca una bolita".
- Fácil verdad!!!

Ahora deberíamos hacer las 8 fases propuestas, programar y jugar un rato.

Por cierto, si no sabes cómo generar números aleatorios no repetidos, inténtalo. Inténtalo otra vez. Luego pregunta a las compañeras. Y por último pregúntanos a los profes.

Bonus

Os propongo un bonus para cuando nos aburrimos de jugar. Es un poco aburrido estar dándole al botón **Sacar una bolita** un montón de veces.

Vamos a añadir un botón **Play**. Cuando una usuaria pulse este botón tenemos que programar un sistema que cada segundo genere una bolita nueva en **Bolitas** y así el bingo juega solo.

Para ello añade una nueva entrada a tu flujo en papel, repite las 8 fases y verás que al final añadiendo aproximadamente 10 líneas más de código a tu JS lo tendrás solucionado.

Corrección del ejercicio

Si quieres que te corrijamos el ejercicio crea un repo y haz un commit por cada una de las fases. Para las fases iniciales haz una foto a tu flujo en papel y commitealo también. Y mándanos el repo.