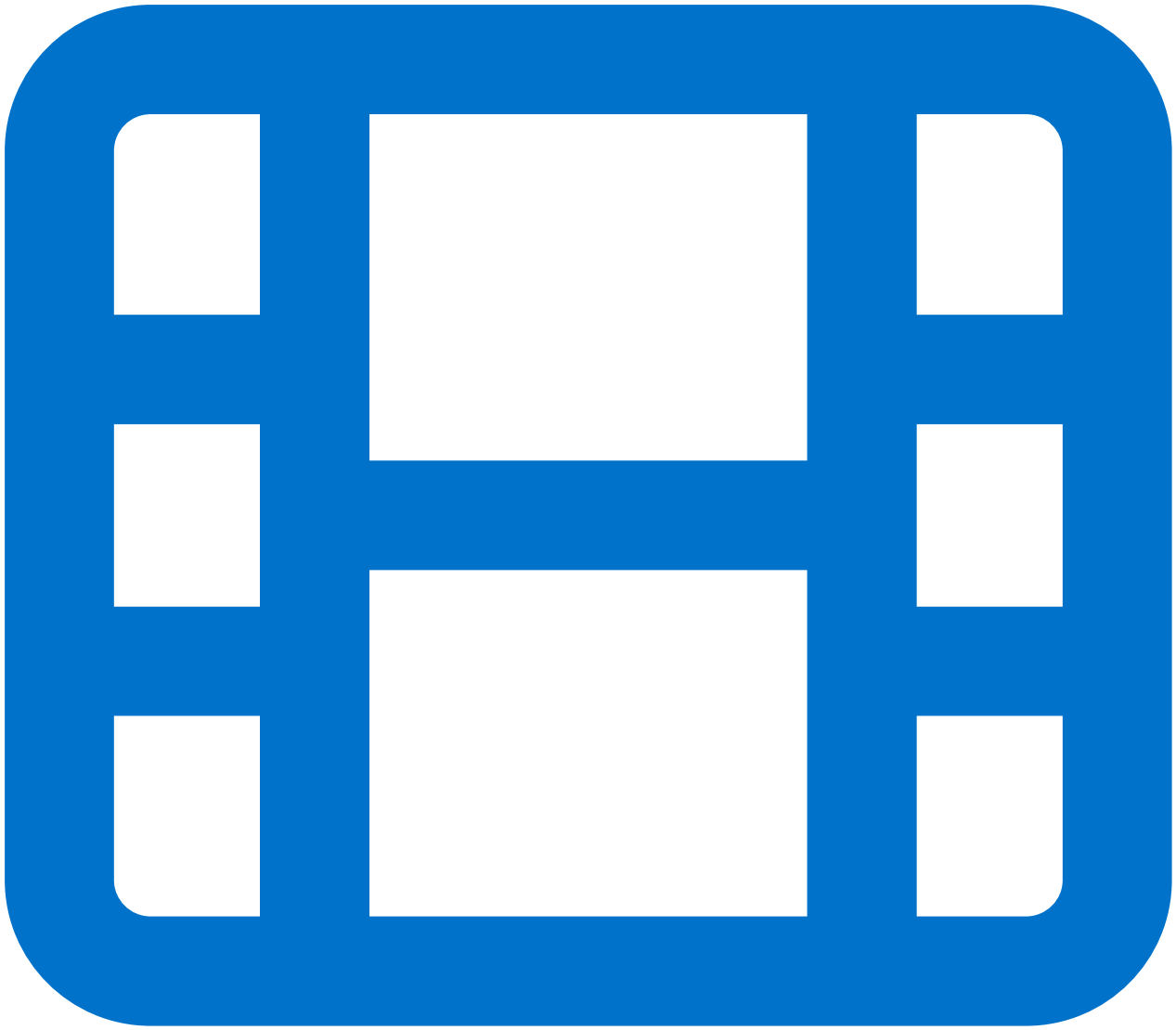
Manejo de DOM con JavaScript II

En estas actividades vas a renderizar la página principal modificando el DOM con JavaScript. Analiza el mockup de la vista provisto ya que debe quedar lo más parecido posible.

> 💡 ***Atención***
>
> *Antes de finalizar su sesión en equipo de hoy, un integrante deberá compartir su pantalla y explicar en detalle cómo resolvió los ejercicios propuestos. Puede ser el facilitador o cualquier compañero que se anime (puedes ser tú 💪).*
>
> *Aprovecha esta oportunidad para enriquecer tu propia resolución y participa activamente haciendo preguntas.*

📽️ Material Audiovisual

Mira el siguiente video para reforzar el contenido teórico y ayudar a resolver las actividades más rápido:



Renderizar templates en el DOM | Egg

📝 Actividad: Renderizando los productos con un template string

En esta actividad, vas a mejorar la vista principal del proyecto store de las secciones anteriores para renderizar las tarjetas de producto con métodos del DOM. La tarjeta ya fue maquetada en el curso de HTML-CSS y debería renderizarse de la siguiente manera:

Para este ejercicio vas a seguir los siguientes pasos:

1. Desde Visual Studio Code, abre la carpeta **store**
2. Abre index.html y conecta correctamente el script products.js
3. El script products debe tener únicamente:
4. clase Product sólo con el constructor
5. seis instancias de la clase Product (con todas las propiedades)
6. array products con las seis instancias
7. Crea el archivo productCards.js en la carpeta scripts y conectalo con la vista principal. Deberías tener tres scripts (layout, products y productCards).
8. Analiza el código del archivo index.html e identifica la etiqueta que es “padre” de las tarjetas de los productos.
9. Asigna a esa etiqueta un id=”products” para usar como selector de JavaScript.
10. **Corta** el contenido de todas las tarjetas de productos.

11. Abre el archivo productCards.js y define una variable productsSelector, para seleccionar con el método getElementById la etiqueta correspondiente donde se renderizarán las tarjetas de productos. A modo de ejemplo:

```
const productsSelector = document.getElementById("products");
```

12. Define una variable productsTemplate con un template string y **pega** el contenido de todas las tarjetas:

```
let productsTemplate = `
<article class="product-card"> ... </article>
<article class="product-card"> ... </article>
...
`;
```

13. Imprime el contenido en el selector con la propiedad innerHTML.

14. Abre con live server el archivo index.html y observa las tarjetas de producto.

15. Programa y verifica la vista la cantidad de veces que sea necesario para que las tarjetas se rendericen y funcionen correctamente.

🖍️ Actividad: Renderizar los productos de forma dinámica

Las tarjetas simplemente fueron "impresas" en la vista, pero siguen siendo datos estáticos. En esta actividad, vas a mejorar la vista principal del proyecto store para renderizar las tarjetas de producto de forma dinámica. Para este ejercicio vas a seguir los siguientes pasos:

1. Desde Visual Studio Code, abre la carpeta **store**

2. Abre productCards.js y define una función createCard para que reciba como parámetro un objeto y devuelva un template string con sólo una tarjeta de producto. El objeto a recibir por la función será un producto con todas las propiedades definidas en la clase.

3. Modifica cada dato "estático" de la tarjeta por la correspondiente propiedad del objeto product. Recuerda que en el template string se "invocan" variables de js con **${ }.** A modo de ejemplo:

```
function createCard(product) {
   return `
      <a href="/details.html" class="product-card">
         <img class="product-img" src="${product.images[0]}" alt="${product.id}">
         <div class="product-info">
            <span class="product-title">${product.title}</span>
      ...
   `;
}
```

4. Reemplaza el contenido de la variable productsTemplate con una iteración for of para que cada vuelta “cargue” una tarjeta de producto.

```
let productsTemplate = "";
for (const element of products) {
  productsTemplate = productsTemplate + createCard(element)
}
```

5. Imprime el contenido en el selector con la propiedad innerHTML.

6. Abre con live server el archivo index.html y observa las tarjetas de producto.

7. Programa y verifica la vista la cantidad de veces que sea necesario para que las tarjetas se rendericen y funcionen correctamente.

🖍️ Actividad: Función de renderizado

En esta actividad vamos a crear una función que “encapsule” todo lo desarrollado en la actividad anterior. Para este ejercicio vas a seguir los siguientes pasos:

1. Desde Visual Studio Code, abre la carpeta **store**
2. Abre productCards.html y define una función printCards que reciba como parámetro un array de productos arrayOfProducts y el id del selector HTML idSelector y que realice las siguientes instrucciones:
    1. Define una variable productsTemplate para concatenar todas las tarjetas de productos
    2. Itera con for of para que cada vuelta “cargue” una tarjeta de producto
    3. Selecciona con getElementById la etiqueta idSelector
    4. Imprime con innerHTML el contenido guardado en la variable productsTemplate

```
function printCards(arrayOfProducts, idSelector) {
  let productsTemplate = "";
  for (const element of arrayOfProducts) {
    productsTemplate = productsTemplate + createCard(element);
  }
  const productsSelector = document.getElementById(idSelector);
  productsSelector.innerHTML = productsTemplate;
}
```

3. Luego es necesario invocar a la función pasando el array de productos y el id del selector. A modo de ejemplo:

```
printCards(products, "products");
```

4. Abre con live server el archivo index.html y observa las tarjetas de producto.

5. Programa y verifica la vista la cantidad de veces que sea necesario para que las tarjetas se rendericen y funcionen correctamente.

📖 Repositorio en GitHub

Puedes subir a un repositorio existente de github la práctica del día de hoy. Los comandos básicos son:

- **git add .** (para preparar todos los archivos creados, modificados o eliminados)
- **git commit -m “nombre del envío”** (para versionar en repositorio local)
- **git push origin main** (o master según corresponda para enviar la nueva versión del proyecto al repositorio remoto)