EJERCICIOS SOBRE WEBCOMPONENTS

Para los ejercicios con módulos, tienes un zip denominado "Servidor para enunciado" en el que dispones de los archivos necesarios para arrancar un servidor en Node.js y realizar las pruebas necesarias.

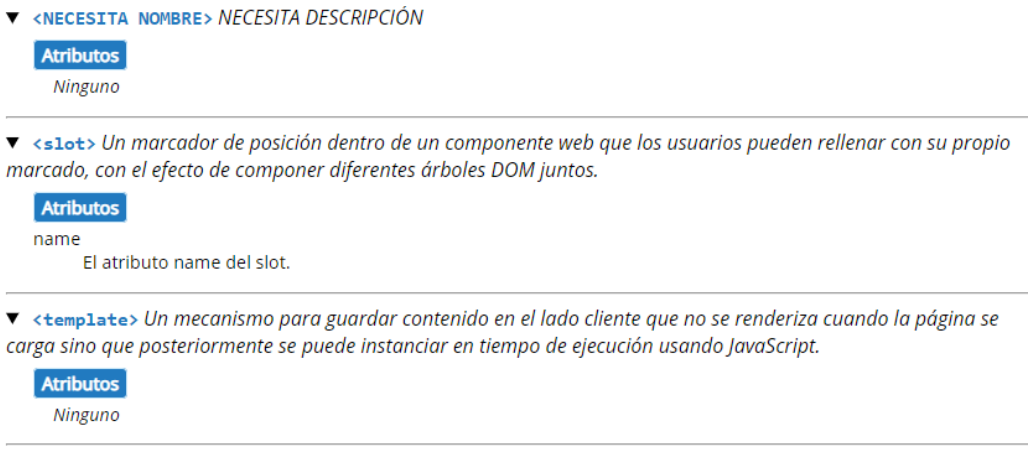
1. Crea un WebComponent que herede de **HTMLElement** con las especificaciones que se indican a continuación
- El componente se definirá con el nombre **wc-blink**
- El componente puede recibir tres atributos (los tres son opcionales):
  - ✓ **baseColor:** Un color en un formato válido en CSS, ya sea RGB o hexadecimal. Si este atributo no existe, entonces el valor será "inherit".
  - ✓ **alternativeColor:** Lo mismo que el anterior. Lo ideal es que sea un color diferente a baseColor. Si este atributo no existe, entonces el valor será "transparent".
  - ✓ **changeInterval:** Un número entero que se expresa en segundos. Si no se especifica nada, el componente web considerará 1 segundo.
- El componente se encarga de cambiar de color el texto de la etiqueta **wc-blink** cada "n" segundos según el valor de **changeInterval** siguiendo la secuencia siguiente: el color inicial es **baseColor**, a los "n" segundos según **changeInterval** cambia a **alternativeColor,** cuando pase este tiempo vuelve a cambiar a **baseColor** y así mientras el navegador siga abierto.

2. Ahora realiza lo mismo que en el ejercicio anterior, pero heredando de **HTMLSpanElement** cambiando la forma de definir la etiqueta, si fuese necesario.

3. Crea un WebComponent con las especificaciones que se indican a continuación.
- Se trata de un Custom Element basado en la etiqueta **detail** de HTML. A continuación, se recuerda que un detail contiene una etiqueta denominada **summary** y un texto inicialmente oculto definido con otra etiqueta. Cuando se hace click en el texto asociado con summary, se muestra el texto oculto. Un ejemplo sería el siguiente, en el que al hacer click en el texto "Epcot Center" se despliega el párrafo de debajo:

```
<details>
    <summary>Epcot Center</summary>
    <p>Epcot is a theme park at Walt Disney World Resort featuring
       Exciting attractions, international pavilions, award-winning
       fireworks and seasonal special events.
    </p>
</details>
```

- El objetivo es crear un detail personalizado con el nombre de etiqueta **element-details.** A continuación se muestra una imagen con tres ejemplos de funcionamiento ya desplegados (si hacemos click en la flecha de la izquierda se mostraría solamente la primera línea con el título en azul). Si echas un vistazo, especialmente al primer ejemplo, verás que hay tres etiquetas personalizables.
  - ✓ **<NECESITA NOMBRE>**
  - ✓ **NECESITA DESCRIPCIÓN**
  - ✓ **Ninguno (dentro de atributos)**



- En los otros dos ejemplos alguna de estas etiquetas tiene un valor. Debes configurar el WebComponent para que aparezca por defecto el texto del primer ejemplo, pero que a la hora de definir la etiqueta sea personalizable y se pueda cambiar por el texto que decida el diseñador.
- Los estilos deben ser lo más parecidos posible a la imagen de arriba. El componente debe crearse empleando un template que se carga en el shadow DOM.

4. Un elemento bastante típico en formularios de páginas web es el campo de tipo “switch”. Vamos a crear un WebComponent denominado **my-switch** que implementa esta funcionalidad como se indica a continuación:

- La visualización por defecto y cuando no hay atributos es la siguiente:

- Si hacemos click la visualización cambiará a lo que se especifica en la imagen de debajo:

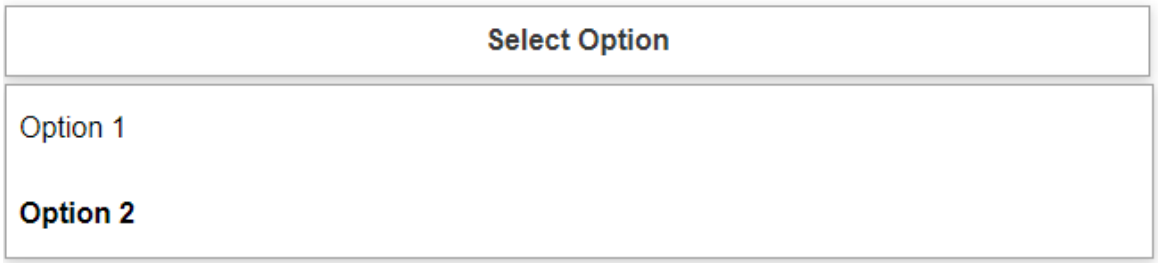
- Para lograr este resultado tenemos un componente web que, como se indicaba anteriormente, cuando no hay atributos tiene por defecto el estado y visualización de tipo “OFF”. Sin embargo, también se admite el atributo estado. Si el valor es “on”, la visualización será como en la imagen anterior con el cuadrado de la derecha en verde y el texto “ON”. Si es “off” obviamente el resultado será como en la primera imagen.
- Por otro lado, si el valor de “estado” es diferente a “on” y “off” se forzará la visualización de la primera imagen con “OFF” y el primer cuadrado en rojo.
- El componente web cambia entre los diferentes estados cuando se hace click en cualquier sitio del área que ocupa en el HTML. Puedes modificar el cursor, por ejemplo, una mano para que el usuario sepa que puede hacer click.
- Adicionalmente hay dos métodos “apagar” y “encender” que se pueden llamar desde fuera del componente con el DOM de la página. Permiten cambiar el estado a “on” y “off” respectivamente.
- Puedes crear una página similar a la siguiente para realizar varias pruebas. Los botones permiten probar los métodos “apagar” y “encender” que se llaman desde el JavaScript del HTML.

5. Crea un WebComponent mediante un módulo de JavaScript con las especificaciones que se indican a continuación:

- El componente se denomina **x-product** y tendrá un aspecto similar al siguiente:

- Como puedes observar, es una imagen con un enlace a la derecha que recibe dos atributos:
  - ✓ **data-name:** Es el nombre del enlace que aparece a la derecha.
  - ✓ **data-url:** Es una URL a la que se redirige haciendo click tanto en la imagen como en el texto. Puede ser cualquier URL de prueba. Por ejemplo: https://example.com
- La imagen se carga del servidor de la URL https://s3-us-west-2.amazonaws.com/s.cdpn.io/4621/. Para ello, el componente debe estar preparado para añadir el nombre del atributo **data-name** en minúsculas a la URL añadiendo el formato png. Por ejemplo, si data-name tiene el valor Ruby, entonces la URL será https://s3-us-west-2.amazonaws.com/s.cdpn.io/4621/ruby.png. Las imágenes del componente se reducirán a 150x150.
- El servidor ahora mismo solo contiene tres imágenes de prueba asociadas al atributo **data-name** con valores Ruby, JavaScript y Python.
- Para generar la URL de la imagen se creará un módulo aparte que se importará en el WebComponent. El WebComponent también se creará como un módulo que exporta la clase correspondiente.
- Debes realizar dos versiones: una donde el propio JavaScript que crea el WebComponent lo define y otra que defina el componente el propio HTML importando el módulo correspondiente.

6. Crea un componente web denominado **my-dropdown**. Este componente emplea el botón de tipo **my-button** del ejemplo 6.2 del aula virtual. El WebComponent en principio consiste en un botón del tipo mencionado (el texto que contiene "Select option" es de tipo my-button"), que sin embargo al hacer click despliega la siguiente información.

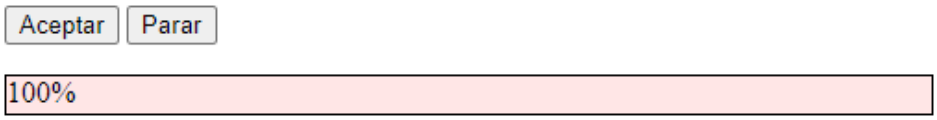


Para lograr el resultado de la imagen necesitamos los siguientes atributos:

- El atributo obligatorio **options.** Es de tipo JSON y permite identificar cada elemento del listado desplegable mediante un identificador y una etiqueta. Por ejemplo, en la imagen de arriba sería `options='{ "option1": { "label": "Option 1" }, "option2": { "label": "Option 2" }}'`. Es decir, tenemos un identificador que se asocia a un objeto con la propiedad label cuyo valor es el texto que se muestra realmente.

- El atributo opcional **option** permite seleccionar por defecto una de las opciones. Si te fijas, cuando se selecciona una opción se cambia el estilo a negrita como en la imagen de arriba. Este atributo hace referencia al identificador, por ejemplo `option="option2"`.
- Cuando el usuario hace click en "Select Option" se despliega el listado. Al desplegar el componente, podemos seleccionar alguna de las opciones en cuyo caso se ocultan y se cambia al estilo a negrita. Al desplegar de nuevo el listado aparecerá la nueva opción marcada.
- Realiza dos versiones de este ejercicio: con módulos y sin módulos.

7. Vamos a combinar los conocimientos adquiridos de animaciones y WebComponents. Para ello, vamos a crear un componente llamado **progress-bar** similar a la imagen de debajo.



- La barra de progreso está inicialmente en blanco sin mostrar nada. La captura de pantalla de arriba indica el estado cuando ha terminado.
- La barra recibe un atributo denominado **seconds**, que es la cantidad que tarda en completarse. Si este campo no existe o es menor que cero, entonces se mostrará una alerta.
- El usuario hace click en el botón "Aceptar" para comenzar a completar la barra de progreso. Además de rellenar el contenido con un color de fondo, se debe indicar el porcentaje actual en cada instante como se puede observar.
- El usuario puede parar la barra de progreso en el botón correspondiente denominado "Parar". La barra de progreso no se puede reanudar y al hacer click en "Aceptar" se comenzaría desde cero.
- Si se detecta un cambio por JavaScript del atributo **seconds**, la barra de progreso volverá al valor 0. Puedes añadir un botón adicional en la página para probar esta funcionalidad.