# PRÁCTICA 5.04 Asincronismo "síncrono" con async/await

**Normas de entrega**

- En cuanto al **código**:
  - en la **presentación interna**, importan los **comentarios**, la claridad del código, la significación de los nombres elegidos; todo esto debe permitir considerar al programa como **autodocumentado**. No será necesario explicar que es un `if` un `for` pero sí su funcionalidad. Hay que comentar las cosas destacadas y, sobre todo, las **funciones** y **clases** empleadas. La ausencia de comentarios será penalizada,
  - en la **presentación externa**, importan las leyendas aclaratorias, información en pantalla y avisos de ejecución que favorezcan el uso de la aplicación,
  - todo el código debe estar situado dentro del evento `window.onload = () => {};` o a través del evento `document.addEventListener("DOMContentLoaded", () => { });` ,
  - si no se especifica lo contrario, la información resultante del programa debe aparecer en la consola del navegador `console.log(información)`,
  - los ejercicios deben realizarse usando ***JavaScript* ES6** y usar el modo estricto (`use strict`) No se podrá utilizar *jQuery* ni cualquier otra biblioteca (si no se especifica lo contrario en el enunciado),
  - para el nombre de **variables**, **constantes** y **funciones** se utilizará *lowerCamelCase,*
  - para la asignación de eventos se utilizará `addEventListener()` indicando sus tres parámetros en su definición,
  - debes dividir tu código en **bibliotecas temáticas** de funciones y/o clases a partir de este ejercicio,
  - se usarán las funcionalidades `import` y `export` para crear bibliotecas de funciones temáticas a partir de esta práctica,
  - todo el código que sea susceptible de retrasar su ejecución debe escribirse de forma asíncrona utilizando `async/await`,
  - todas las funciones asíncronas deben tener control de errores.
- En cuanto a la **entrega** de los archivos que componen los ejercicios:
  - todos los ejercicios en **una carpeta** (creando las **subcarpetas** necesarias para documentación anexa como imágenes o estilos) cuyo nombre queda a discreción del discente,
  - el nombre de los ficheros necesarios para resolver el ejercicio será el número de ejercicio que contenga,
  - el código contendrá ejemplos de ejecución, si procede, y
  - la carpeta será **comprimida** en formato `ZIP` y será subida a **Aules** de forma puntual.

**Ejercicio 1 - Migrando a `async/await`**

Modifica el código de la enciclopedia de ***Star Wars*** (http://swapi.py4e.com/ o https://swapi.dev/) para que consuma el código asíncrono con **`async/await`**. Además, debes utilizar el **conjunto** de **`promises`** en aquellas solicitudes que deban resolverse como una (en personajes y pilota).

Además de toda la información que ya muestra la enciclopedia, se añadirá un **listado de las naves y vehículos** (si los hubiera) con los que tenga una relación cada **personaje**. Crea un botón que ponga Pilota, al pulsar sobre el botón Pilota se mostrará una ficha con los **datos básicos** de cada nave y vehículo. Si no hay ninguno se debe informar al usuario.

Muestra la información donde y como creas oportuno, pero recuerda, debidamente formateada en el **DOM**.