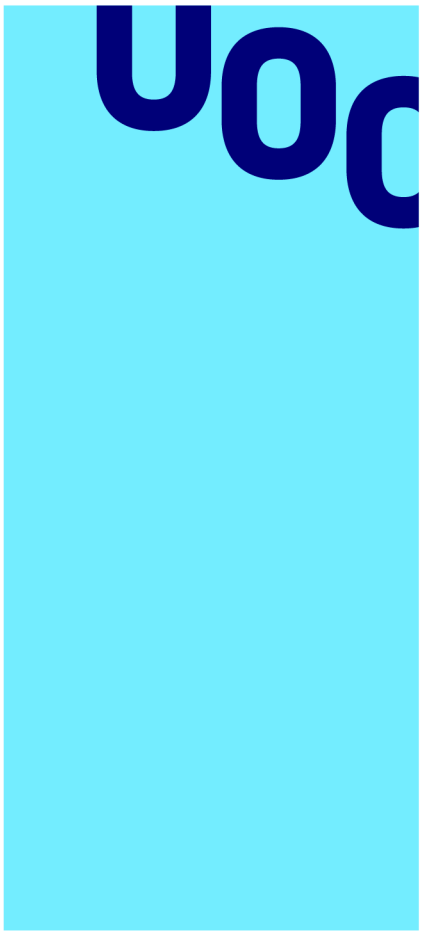
# PEC 5

## Frameworks: Formularios en Angular



Universitat Oberta de Catalunya

### Información relevante:

- Fecha límite de entrega: 14 de diciembre.
- Peso en la nota de FC: 12.5%.

# Contenido

# Información docente

## Presentación

Esta práctica se centra en conocer la creación de formularios en Angular, haciendo uso de las dos técnicas con las que el framework nos permite operar: formularios dirigidos por template y reactivos.

## Objetivos

Los objetivos que se desean lograr con el desarrollo de esta PEC son:

- Comprender las diferencias entre **formularios dirigidos por template** y **formularios reactivos**.
- Crear aplicaciones Angular usando **formularios dirigidos por template**.
- Crear aplicaciones Angular usando **formularios reactivos**.
- Construir formularios básicos y complejos utilizando **validaciones genéricas y customizadas**.
- Aprender a **leer y comprender** la documentación oficial de **Angular.**

# Enunciado

Esta PEC contiene 4 ejercicios evaluables. Debéis entregar vuestra solución de los 4 ejercicios evaluables (ver el último apartado).

Debido a que las actividades están encadenadas (i.e. para hacer una se debe haber comprendido la anterior), **es altamente recomendable hacer las tareas y ejercicios en el orden en que aparecen en este enunciado**.

Antes de continuar debes:

Haber leído el recurso teórico `T01.PEC5_Teoria_20211.pdf` disponible en el apartado "Contenidos y recursos" del aula de esta PEC.

## Ejercicio 1 – Preguntas teóricas sobre formularios Angular (1.5 puntos)

**Crea un documento de texto `PEC5_Ej1_respuestas_teoria`** y **responde** a cada una de las siguientes preguntas:

a) ¿Cuáles son las principales diferencias entre **formularios dirigidos por template** y **formularios reactivos**?

b) ¿Qué son, para qué sirven y cómo se utilizan las **directivas `ngModel` y `ngModelChange`**?

c) ¿Qué son, cuáles son y para qué sirven los **estados en los formularios dirigidos por templates**?

d) ¿Qué **ventajas** aportan los **`FormGroup`** en la composición de formularios?

# Ejercicio 2 – Práctica – Formularios dirigidos por template (3.5 puntos)

Partiendo **del último ejercicio realizado en la PEC anterior**, crea una carpeta **`PEC5_Ej2`** y realiza las siguientes tareas:

Crea un **nuevo componente** llamado **`wine-new`** en nuestro directorio de componentes `wines`, en el cual se encuentran los componentes `wine-item` y `wine-list` de la práctica anterior. Debes **crear un formulario** que tome la siguiente información:

- Nombre de vino.
- Precio del vino.
- URL de la imagen.
- Variable para saber si está en venta.

Debes **crear también un `FormGroup`**, denominado **`wine`**, para agrupar todos los componentes del formulario. Por tanto, para desarrollar este ejercicio debes utilizar el **enfoque de "Form Groups"** en lugar de usar el `two-way data binding` de "`ngModel`". El formulario debe mostrar un resultado similar al siguiente:


**Validaciones**:

- Todos los **campos son obligatorios** (salvo el checkbox).

- El **precio** debe ser **numérico**.

- Se **debe cumplir el patrón de una URL** para una imagen usando una expresión regular (RegEx) que valide la aparición de una ruta válida (ejemplo: https://www.direccion.com/nuevo-vino.jpg). Lo importante de la práctica no es la expresión regular como tal, sino la validación del formulario. El patrón debería empezar por `http` o `https` seguido de `://` a continuación vendría una letra en minúsculas o mayúsculas o un número del 0 al 9, como mínimo. Obligatoriamente después del nombre vendría un "." y el dominio estaría compuesto entre 2 y 3 caracteres que serán letras minúsculas o mayúsculas. Es decir, estaremos formando **una URL válida para localizar un recurso**, en este caso una imagen.

- Deberás **mostrar mensajes de error** cuando las validaciones sean incorrectas cuando se esté rellenando la información en el formulario, haciendo uso de directivas estructurales en el propio *template*, estos mensajes se mostrarán solamente **después de que el usuario haya editado los campos, o bien después que se realiza un envío al formulario (`submit`)**.

A continuación, se muestran las imágenes con **los mensajes que se deberán mostrar en la validación de los formularios**. La primera imagen es el **estado inicial**, que no debería mostrar ningún error. En la segunda imagen se muestra el formulario una vez que **ha sido editado una vez y se ha borrado su contenido**, finalmente la última imagen muestra el formulario tras **introducir una URL incorrecta**.


**NOTA**: No se debe incluir el vino en la lista de vinos, sino que se mostrará por consola que se han recogido correctamente los datos desde el formulario. Agregaremos los vinos a la lista en una siguiente PEC.

# Ejercicio 3 – Preguntas teóricas sobre formularios reactivos Angular (1.5 puntos)

**Crea un documento de texto** `PEC5_Ej3_respuestas_teoria` y **responde** a cada uno de los siguientes puntos:

a) ¿Qué son, para qué sirven y cómo se utilizan **`FormControl`**, **`FormGroup`** y **`FormBuilder`**?

b) Busca en la **página oficial de Angular** (o utiliza un recurso de **O'Reilly**) en el que se especifiquen **todos los validadores que incluye Angular** para ser utilizados en los **formularios reactivos**. Construye una tabla de resumen de estos.

c) ¿Qué son, cuáles son y para qué sirven **los estados** en los **formularios reactivos**?

# Ejercicio 4 – Práctica – Formularios reactivos (3.5 puntos)

**Partiendo del último ejercicio realizado en la PEC4** (es decir **no se comenzará desde el Ejercicio 2 de esta PEC**), crea una carpeta `PEC5_Ej4` y realiza el mismo ejercicio que has hecho utilizando los formularios dirigidos por template pero utilizando formularios reactivos. Para ello debes seguir los siguientes pasos:

- Crear un **nuevo componente** que permita agregar nuevos vinos.
- Crear un **formulario** que tome los siguientes datos: **nombre, precio, URL de la imagen y si está en venta o no.**
- Crear un **`FormGroup`** el cual **encapsulará el formulario**. Deberás **utilizar `FormBuilder`.**
- Incluir la **validación** de que **todos los campos son obligatorios** salvo el checkbox.
- Incluir la **validación** de que el **precio** de un vino debe ser **mínimo 1€.**
- Incluir la validación de que la **URL de la imagen es válida** usando un patrón RegEx.
- Deberás mostrar **mensajes de error** cuando las **validaciones sean incorrectas**, pero **sólo** o bien después de que el usuario haya **editado los campos**, o bien **después que se realiza un envío al formulario** (`submit`).
- Crear una **validación customizada** (propia) denominada `NameWineValidator` que se asociará al nombre del vino y que **permitirá validar** que el nombre de los vinos es alguno de los siguientes:
    - Laya
    - K-Naina
    - Verdejo
    - Monastrell


**NOTA**: Al igual que en el ejercicio 2, no se debe incluir el vino en la lista de vinos, sino que se mostrará por consola que se han recogido correctamente los datos desde el formulario. Agregaremos los vinos a la lista en una siguiente PEC.

# Formato y fecha de entrega

Tienes que entregar un fichero *.zip, cuyo nombre tiene que seguir este patrón: `loginUOC_PEC5.zip`. Por ejemplo: `dgarciaso_PEC5.zip`. Este fichero comprimido tiene que incluir los siguientes elementos:

- Un fichero de texto `PEC5_Ej1_respuestas_teoria` para las respuestas del ejercicio 1.
- Una carpeta `PEC5_Ej2` con los ficheros resultado de haber realizado las tareas del ejercicio 2 ( No olvides **eliminar las carpetas node_modules y .angular** )
- Un fichero de texto `PEC5_Ej3_respuestas_teoria` para las respuestas del ejercicio 3.
- Una carpeta `PEC5_Ej4` con los ficheros resultado de haber realizado las tareas del ejercicio 4 ( No olvides **eliminar las carpetas node_modules y .angular** )

**Penalizaciones**

- Entrega en otro formato que no sea el especificado (ej. en `zip`): **-0.75 puntos**
- Comprimir archivos dentro del `zip`: **-1 puntos**.
- Para cada ejercicio/apartado donde no se respete la nomenclatura exacta de las carpetas o ficheros indicados (símbolos, minúsculas, mayúsculas, etc.): **-0.75 puntos.**
- La no entrega del repositório local `git` **-3 puntos**
- Por cada carpeta node_modules y .angular entregada **-0.75 puntos**

El último día para entregar esta PEC es el **14 de diciembre de 2021** hasta las **23:59**. Cualquier PEC entregada más tarde será considerada como no presentada.