

Examen

Programación 2º DAM

Noviembre 2025



I.E.S.
Doctor Balmis



GENERALITAT
VALENCIANA

Conselleria de Educació,
Cultura y Deporte



Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Nombre

Instrucciones para la realización de la prueba:

- Duración de la prueba **3:50 h** desde el comienzo de la misma.
- Lee atentamente el enunciado y no formules tus preguntas en voz alta, levanta la mano y espera a que el profesor te diga que tienes que hacer.
- Si te atascas en un apartado déjalo y pasa al siguiente. Cada apartado tiene su puntuación, independientemente del funcionamiento del programa final y el orden propuesto en el enunciado.
- **No se puede usar ningún tipo de dispositivo USB, ni código de ejercicios o ejemplos realizados en clase. Solo los pdf de apuntes que se te proporcionarán para la ocasión.**
- En caso de pillar a algún alumno copiando o usando material no autorizado se le retirará el examen y su calificación en la evaluación será un suspenso.

Criterios de Calificación

- (☒ **0.5 pts**) Imprescindible seguir la arquitectura explicada en clase para que se corrija el examen. Clases necesarias para el manejo de los datos, como se ha explicado en clase, más icono de lanzamiento cambiado.
- (☒ **0,5 pts**) funcionamiento para que se muestra la pantalla **Inicio** o la pantalla **Pregunta** en el momento correcto.
- (☒ **2 pts**) Maquetación de **DatosJugador** con todas sus funcionalidades (TextFiel, Slider y botón), con la validación correcta de los datos introducidos.
- (☒ **1 pts**) Funcionamiento correcto del diálogo, controlando que se muestre cuando ocurra alguno de los tres errores.
- (☒ **1pts**) Carga de la función Pregunta con la cabecera y pregunta formateada correctamente, y que se muestren las preguntas que se han elegido en la selección.
- (☒ **1 pts**) Construcción del composable que muestra la lista de respuestas con el funcionamiento correcto de la selección y el botón de siguiente.
- (☒ **1 pts**) Gestión de las puntuaciones.
- (☒ **0,5 pts**) Dialogo informativo que muestre la puntuación final.
- (☒ **1,5 pts**) Uso correcto de las clases **Event** , **UiState** y la clase **ViewModel** con toda la lógica dentro de esta.
- (☒ **1 pts**) Aplicar Hilt correctamente.

Aplicación

Vamos a crear una aplicación para el entretenimiento de los alumnos de ciclo de Dam, esta aplicación se basará en el juego del Trivial. Para comenzar deberemos **Cambiar el icono del lanzador de aplicaciones, con el pasado como recurso en imágenes 'trivial'**.

⚠ Importante

Para que se puntúe el examen se deberá seguir la arquitectura expuesta durante el curso. Se pasa la clase DaoMock con las preguntas además de la data class relacionada con esta clase, aloja estos ficheros en el lugar adecuado según la arquitectura explicada.

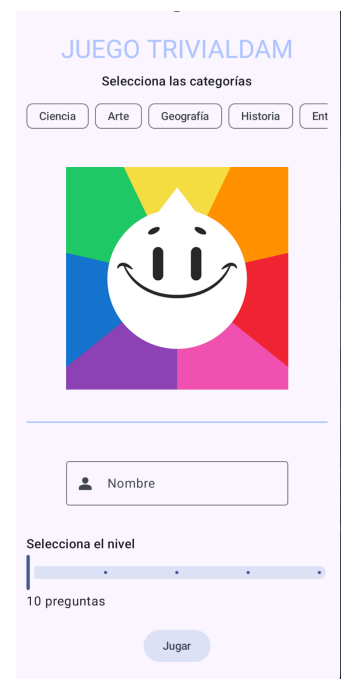
Crearemos una pantalla principal que se llamará **JuegoScreen** que mostrará el **Inicio** o la **Pregunta** del juego, dependiendo en que momento de la partida nos encontremos. Para organizar mejor el código tendremos los dos paquetes donde se alojarán los elementos de esas partes de las pantallas, esta organización ya se os da como recurso.

Inicio

En **Inicio** tendremos una función composable a la que deberás añadir los elementos faltantes para formar la pantalla, lo harás en el fichero **Inicio.kt** que se pasa como parte del código. El aspecto resultante deberá ser como el de la imagen de la derecha.

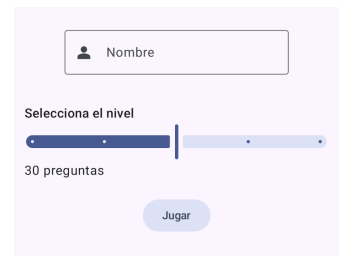
Para que el código nos quede más ordenado dividiremos la funcionalidad en diferentes partes que alojaremos en **funciones composables**, serán las siguientes que pasamos a explicar:

Categorías: esta función emite los chips de los temas que se pueden seleccionar para las preguntas del juego. Tiene gran parte del código implementado y le deberás añadir la funcionalidad para que se muestre seleccionados/deseleccionados los Chips sobre los que se pulsa, además de propagar el evento al hacer la pulsación.



DatosJugador que deberás componer con los elementos necesarios para que se muestre el **TextField** con el nombre (**OutlinedTextFieldName** en nuestra librería) con validación que no podrá ser vacío, el **Slider** con rango de 10 a 50 (el rango de preguntas en las que nos podemos mover) y con tres saltos intermedios (10-20-30-40-50), los dos **Text** el de abajo se actualizará con el número de preguntas seleccionado y el

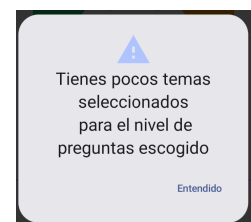
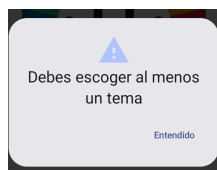
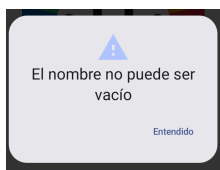
FilledTonalButton que activará las comprobaciones de si todos los datos y selecciones se han hecho correctamente para pasar a la parte de la partida de preguntas.

Un formulario de configuración de jugador con un campo de texto para el nombre, un slider para seleccionar el nivel (rango de 10 a 50) y un botón 'Jugar'.

Si pulsamos el botón de jugar pero no se han completado bien las configuraciones se mostrará un dialogo indicando el error ocurrido, pueden ocurrir los siguientes errores:

- Que el usuario no ha introducido ningún nombre
- Que el usuario no haya seleccionado ninguna categoría
- Que el usuario haya seleccionado más cantidad de preguntas en el nivel que las preguntas que proporciona las categorías seleccionadas (cada categoría tiene 10 preguntas).

El dialogo se cerrará al pulsar fuera de el.

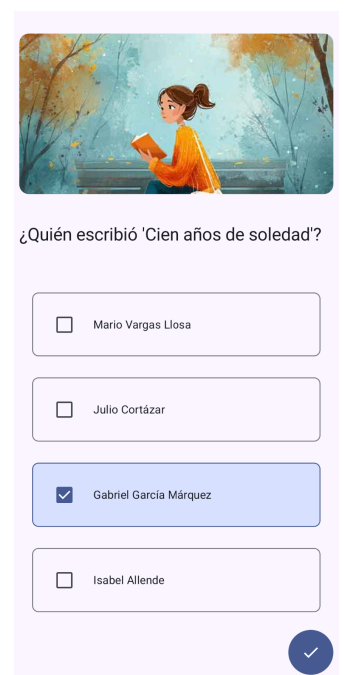


Si no ocurre ningún error, se pasará a mostrar el composabile de **Pregunta**.

Pregunta

El código se deberá de completar en **Pregunta.kt** que se pasa como recurso y que tiene alguna funcionalidad ya implementada. El resto de la función composabile deberás construirlas para conseguir el siguiente aspecto. Para que el código quede más ordenado dividiremos la funcionalidad en diferentes partes que alojaremos en **funciones composables**, serán las siguientes que pasamos a explicar:

CabeceraPregunta que alojará los composables de la imagen y la pregunta. Las imágenes las proporcionaremos como recurso y las deberéis alojar en el servidor local **nginx** en la carpeta html. El acceso deberá ser asincrono mediante el composabile **AsyncImage**.

Un formulario de pregunta que muestra una ilustración de una persona leyendo un libro, la pregunta '¿Quién escribió 'Cien años de soledad'?' y una lista de autores con casillas de selección: Mario Vargas Llosa, Julio Cortázar, Gabriel Garcia Márquez (seleccionado), Isabel Allende. Hay un botón de confirmación en la parte inferior.

Repuesta esta función se pasa implementada como recurso, incluye el rectángulo formateado que cambiará de aspecto si se selecciona.

☒ Gabriel García Márquez

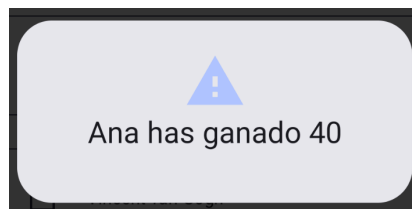
☐ Isabel Allende

El resto de la composición: la repetición de todas las respuestas y el **IconButton** las podrás hacer en el composable padre **pregunta**. Se podrá seleccionar una respuesta o dejar todas sin seleccionar:

- En caso de que la selección sea la correcta se sumará **5 puntos**.
- Si es incorrecta se restarán **5 puntos**.
- Si no hay ninguna selecciona no se sumará ni se restará.

Con el evento del botón se aplicarán los puntos y además se pasará a la siguiente pregunta, se repetirá el proceso mientras queden preguntas.

Al final del juego se mostrará un dialogo indicando los puntos obtenidos por el jugador.



Aviso

Para que el ejercicio cuente como correcto, deberás crear las clases **UiState** y la interfaces selladas **Event** necesarias. Tanto los composables Inicio como Pregunta tendrán su propias interfaces para los eventos, mientras que solo hará falta la clase **JuegoUiState** para contener todos los estados y **JuegoViewModel** para gestionar toda la lógica de la aplicación.