CleverClash

Pablo Piedrola Muñoz

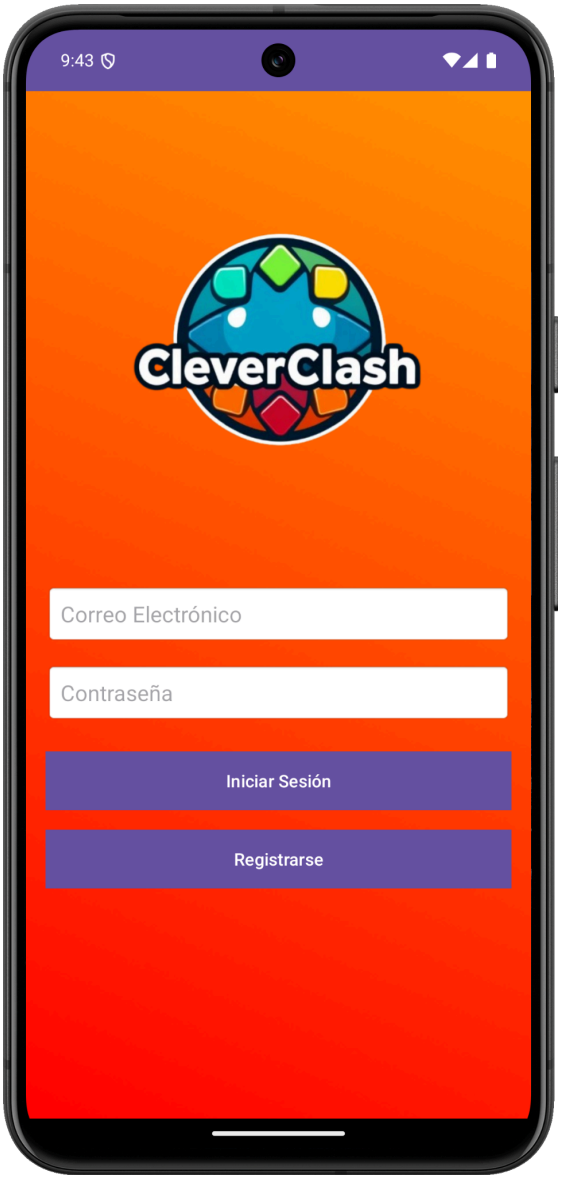
Jesus Garcia Calvo

Jose Manuel Leon Carmona

MEDAC
Instituto Oficial de Formación Profesional

Índice

Introducción



CleverClash es un juego de preguntas y respuestas para dispositivos móviles, inspirado en el estilo de Preguntados. La aplicación permite a los jugadores desafiar su conocimiento en distintas categorías, competir con amigos en partidas remotas y mejorar su rendimiento con una mecánica de puntuación dinámica. Actualmente, el juego sigue en constante proceso de mejora, con actualizaciones y ajustes diarios para optimizar la experiencia del usuario, corregir errores y añadir nuevas funcionalidades que lo hagan más desafiante y divertido.

Repositorio GitHub del Proyecto: CleverClash

Pantallas Obligatorias

1. Pantalla de Inicio

Esta es la primera pantalla que los usuarios ven al abrir la aplicación. Presenta un diseño atractivo con un fondo degradado en tonos rojos y naranjas, reforzando la identidad visual del juego. Los elementos principales incluyen:

a. Logo de CleverClash en la parte superior, destacando la marca.
b. Selección de idioma, permitiendo que los jugadores elijan su idioma preferido antes de continuar.
c. Configuración de dificultad, con tres niveles: Fácil, Medio y Difícil, que afectan la velocidad del cronómetro y la complejidad de las preguntas.

d. Botones de inicio de sesión y registro, proporcionando opciones de acceso para nuevos y recurrentes jugadores.

```
private void loginUser() {  1 usage
    String email = emailEditText.getText().toString().trim();
    String password = passwordEditText.getText().toString().trim();

    if (email.isEmpty() || password.isEmpty()) {
        Toast.makeText( context: LoginActivity.this,  text: "Por favor, complete todos los campos", Toast.LENGTH_SHORT).
        return;
    }

    // Llamar al método de autenticación
    auth.signInWithEmailAndPassword(email, password)
            .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
```

2. Menú Principal

El menú principal actúa como el hub central del juego, donde los jugadores pueden navegar a distintas secciones. Contiene:

a. Rueda de selección de categorías, una animación interactiva que permite seleccionar la categoría de preguntas.
b. Botones de acceso rápido: "Jugar", "Configuración" y "Salir".
c. Avatar del usuario en la esquina superior, con opción de personalización desde el menú de configuración.
d. Indicador de nivel y puntaje, permitiendo que los jugadores vean su progreso de manera instantánea.

```
// Configura el diseño principal
setContentView(R.layout.activity_main);

// Inicializa Firebase
FirebaseApp.initializeApp( context: this);
auth = FirebaseAuth.getInstance();

// Configurar el Toolbar
toolbar = findViewById(R.id.toolbar);
setSupportActionBar(toolbar);

// Obtener el nombre de usuario
FirebaseUser user = auth.getCurrentUser();
String userName = (user != null) ? user.getDisplayName() : "Usuario";
toolbar.setTitle(userName);

// Botones
Button newGameButton = findViewById(R.id.newGameButton);
Button dataButton = findViewById(R.id.dataButton);
Button btnIrARuleta = findViewById(R.id.btnRuleta);
```

3. Pantalla de Preguntas

Esta pantalla es donde ocurre la acción principal del juego. Se caracteriza por:

a. Fondos de colores vibrantes, que cambian según la categoría de la pregunta para mejorar la identificación visual.
b. Pregunta en la parte superior, destacada con un diseño claro y legible.
c. Cuatro opciones de respuesta, presentadas en botones de colores llamativos para facilitar la selección.
d. Cronómetro regresivo, que presiona al jugador a responder antes de que el tiempo se agote.
e. Animaciones visuales para resaltar respuestas correctas (destello verde) e incorrectas (efecto rojo y vibración opcional).
f. Sonidos de confirmación que refuerzan la retroalimentación inmediata para el usuario.

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_entretenimiento);

    // Configurar el nombre del usuario
    TextView userNameView = findViewById(R.id.userName);
    userNameView.setText(userName);

    // Configurar la pregunta
    questionBox = findViewById(R.id.questionBox);
    questionBox.setText("¿Cuál es la película más taquillera de la historia?");

    // Configurar los botones
    btnGreen = findViewById(R.id.btnGreen);
    btnYellow = findViewById(R.id.btnYellow);
    btnBlue = findViewById(R.id.btnBlue);
    btnRed = findViewById(R.id.btnRed);
```

4. Pantalla de Resultados Finales

Después de completar una ronda de preguntas, se presenta la pantalla de resultados, con un diseño dinámico que destaca el rendimiento del jugador. Incluye:

a. Fondo con efectos visuales de celebración para victorias o tonos más neutros si el rendimiento fue bajo.
b. Puntuación total obtenida, calculada en base a velocidad y respuestas correctas.
c. Estadísticas detalladas, como número de respuestas correctas e incorrectas.
d. Botones de acción: "Volver al menú" o "Jugar de nuevo", permitiendo una rápida decisión para continuar jugando.

5. Menú de Configuración

En esta sección, los jugadores pueden personalizar su experiencia ajustando diferentes parámetros:

a. Ajustes de idioma para cambiar el idioma en cualquier momento.
b. Volumen de sonidos y efectos, permitiendo desactivar o ajustar los efectos sonoros.
c. Personalización de avatar, con opciones de selección de imagen y colores.
d. Opciones de accesibilidad, como aumento de tamaño de texto o modo alto contraste para mejorar la visibilidad.

```
private void changeLanguage(String languageCode) {  2 usages
    // Guardar idioma en SharedPreferences
    SharedPreferences.Editor editor = getSharedPreferences( name: "Settings", MODE_PRIVATE).edit();
    editor.putString( s: "Language", languageCode);
    editor.apply();

    // Aplicar idioma
    Locale locale = new Locale(languageCode);
    Locale.setDefault(locale);
    Configuration config = new Configuration();
    config.setLocale(locale);
    getBaseContext().getResources().updateConfiguration(config, getBaseContext().getResources().getDisp

    // REINICIAR ACTIVIDAD COMPLETAMENTE
    Intent intent = new Intent( packageContext: MainActivity.this, MainActivity.class);
    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
    startActivity(intent);
}
```

6. Pantalla de Juego Remoto

Esta pantalla simula la conectividad con otros jugadores, permitiendo competiciones en línea. Se compone de:

a. Tabla de puntuaciones en tiempo real, mostrando los puntajes de los participantes.
b. Notificaciones de turnos, indicando cuándo es el turno de cada jugador.
c. Efectos visuales dinámicos, como cambios de color en la tabla según el rendimiento de los jugadores.

d. Botón de "Abandonar partida", que permite salir del juego remoto en caso de ser necesario.

Componentes Interactivos

1. Pantalla de Inicio

Esta es la primera pantalla que los usuarios ven al abrir la aplicación. Presenta un diseño atractivo con un fondo degradado en tonos rojos y naranjas, reforzando la identidad visual del juego. Los elementos principales incluyen:

a. Logo de CleverClash en la parte superior, destacando la marca.
b. Selección de idioma, permitiendo que los jugadores elijan su idioma preferido antes de continuar.
c. Configuración de dificultad, con tres niveles: Fácil, Medio y Difícil, que afectan la velocidad del cronómetro y la complejidad de las preguntas.
d. Botones de inicio de sesión y registro, proporcionando opciones de acceso para nuevos y recurrentes jugadores.

2. Menú Principal

El menú principal actúa como el hub central del juego, donde los jugadores pueden navegar a distintas secciones. Contiene:

a. Rueda de selección de categorías, una animación interactiva que permite seleccionar la categoría de preguntas.
b. Botones de acceso rápido: "Jugar", "Configuración" y "Salir".
c. Avatar del usuario en la esquina superior, con opción de personalización desde el menú de configuración.
d. Indicador de nivel y puntaje, permitiendo que los jugadores vean su progreso de manera instantánea.

3. Pantalla de Preguntas

Esta pantalla es donde ocurre la acción principal del juego. Se caracteriza por:

a. Fondos de colores vibrantes, que cambian según la categoría de la pregunta para mejorar la identificación visual.
b. Pregunta en la parte superior, destacada con un diseño claro y legible.
c. Cuatro opciones de respuesta, presentadas en botones de colores llamativos para facilitar la selección.
d. Animaciones visuales para resaltar respuestas correctas (destello verde) e incorrectas (efecto rojo y vibración opcional).
e. Sonidos de confirmación que refuerzan la retroalimentación inmediata para el usuario.

4. Pantalla de Resultados Finales

Después de completar una ronda de preguntas, se presenta la pantalla de resultados, con un diseño dinámico que destaca el rendimiento del jugador. Incluye:

a. Fondo con efectos visuales de celebración para victorias o tonos más neutros si el rendimiento fue bajo.
b. Puntuación total obtenida, calculada en base a velocidad y respuestas correctas.
c. Estadísticas detalladas, como número de respuestas correctas e incorrectas.
d. Botones de acción: "Volver al menú" o "Jugar de nuevo", permitiendo una rápida decisión para continuar jugando.

5. Menú de Configuración

En esta sección, los jugadores pueden personalizar su experiencia ajustando diferentes parámetros:

a. Ajustes de idioma para cambiar el idioma en cualquier momento.
b. Volumen de sonidos y efectos, permitiendo desactivar o ajustar los efectos sonoros.
c. Personalización de avatar, con opciones de selección de imagen y colores.
d. Opciones de accesibilidad, como aumento de tamaño de texto o modo alto contraste para mejorar la visibilidad.
e. Botón para cambiar el nombre de usuario, que permite personalizar el perfil del jugador.

6. Pantalla de Juego Remoto

Esta pantalla simula la conectividad con otros jugadores, permitiendo competiciones en línea. Se compone de:

a. Tabla de puntuaciones en tiempo real, mostrando los puntajes de los participantes.
b. Notificaciones de turnos, indicando cuándo es el turno de cada jugador.
c. Efectos visuales dinámicos, como cambios de color en la tabla según el rendimiento de los jugadores.
d. Botón de "Abandonar partida", que permite salir del juego remoto en caso de ser necesario.

Diseño Adaptativo

1. Compatible con dispositivos móviles y tablets

La interfaz de la aplicación "CleverClash" ha sido diseñada específicamente para dispositivos móviles y tablets, como se evidencia en el diseño vertical de las pantallas y la disposición de los elementos. La estructura en columnas facilita su adaptación a distintos tamaños de pantalla sin perder funcionalidad ni diseño.

2. Colores

Esta es la paleta de colores principal que aparecen en la aplicación, aunque hay más colores, estos son los más relevantes y los que se encuentran en mayor abundancia.

1. Rojo Fucsia (#BF1162)

Energía, pasión, creatividad

Este tono vibrante mezcla la fuerza del rojo con la creatividad del rosa, evocando dinamismo y emoción. Se asocia con la juventud, la innovación y la audacia. Es un color que capta la atención de inmediato y genera entusiasmo.

2. Azul Brillante (#1469D9)

Confianza, seguridad, inteligencia

El azul es un color que transmite calma, estabilidad y profesionalismo. Este tono en particular, más intenso, es dinámico y está relacionado con la tecnología, el conocimiento y la comunicación. Es un color que genera confianza y transmite fiabilidad.

3. Amarillo Mostaza (#F2B705)

Optimismo, alegría, estímulo mental

El amarillo representa la felicidad y la creatividad. Este tono mostaza es un poco más sofisticado que el amarillo puro, evocando entusiasmo y energía sin ser demasiado llamativo. Se asocia con el pensamiento rápido, la claridad y la diversión.

4. Naranja Intenso (#F25C05)

Entusiasmo, acción, emoción

El naranja combina la pasión del rojo con la alegría del amarillo, generando un sentimiento de entusiasmo y dinamismo. Es un color que impulsa la acción y se usa a menudo en juegos o interfaces que buscan motivar al usuario a participar.

5. Rosa Suave (#F2DAD8)

Tranquilidad, sensibilidad, empatía

Este tono pastel es suave y acogedor, evocando calidez, ternura y calma. Se usa para equilibrar los colores más intensos de la paleta, aportando un toque armonioso y relajante.

Resumen de la paleta:

Esta combinación de colores es **dinámica, enérgica y equilibrada**. Los tonos cálidos (rojo, naranja y amarillo) generan entusiasmo y motivación, mientras que el azul y el rosa aportan estabilidad y calma. Es una paleta ideal para una aplicación de juego o aprendizaje, ya que mantiene la emoción sin abrumar al usuario.

3. Interfaz responsiva y escalable

Se observa una progresión de pantallas con elementos que se adaptan al flujo de navegación del usuario, desde el inicio de sesión hasta el resultado final. El uso de colores vibrantes y una disposición clara de los botones aseguran que la experiencia sea fluida en distintos dispositivos. Además, la versión en wireframe sugiere un diseño flexible que puede escalar según las necesidades de la pantalla.

4. Botones grandes y accesibles para una fácil interacción

Los botones son de gran tamaño y tienen colores contrastantes (verde, rojo, azul, amarillo) para que sean fácilmente distinguibles y seleccionables. Esto mejora la accesibilidad y usabilidad, especialmente en pantallas táctiles donde la precisión del toque puede variar según el usuario o el dispositivo.

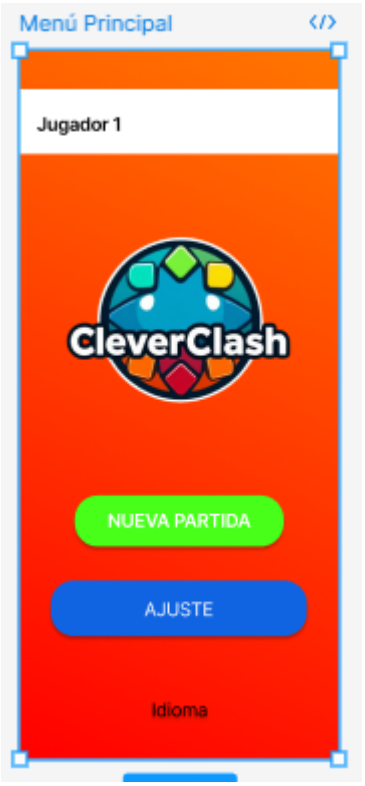
Proyecto realizado en Figma: CleverClash

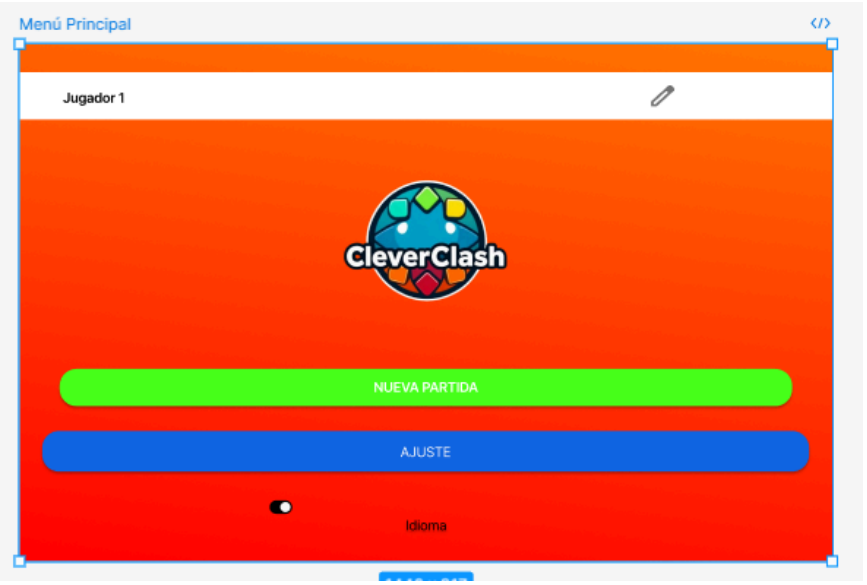
5. Distribución optimizada para diferentes resoluciones de pantalla

La estructura de las pantallas sigue un formato bien distribuido:

- El logotipo y los títulos están en la parte superior.
- Los botones de acción están centrados o alineados de manera uniforme.
- Las preguntas y opciones de respuesta están claramente separadas para evitar confusión.
- La pantalla de resultados muestra los puntajes de forma organizada.
  Este enfoque asegura que la interfaz se mantenga ordenada y funcional tanto en teléfonos con pantallas pequeñas como en tablets con mayor

resolución.





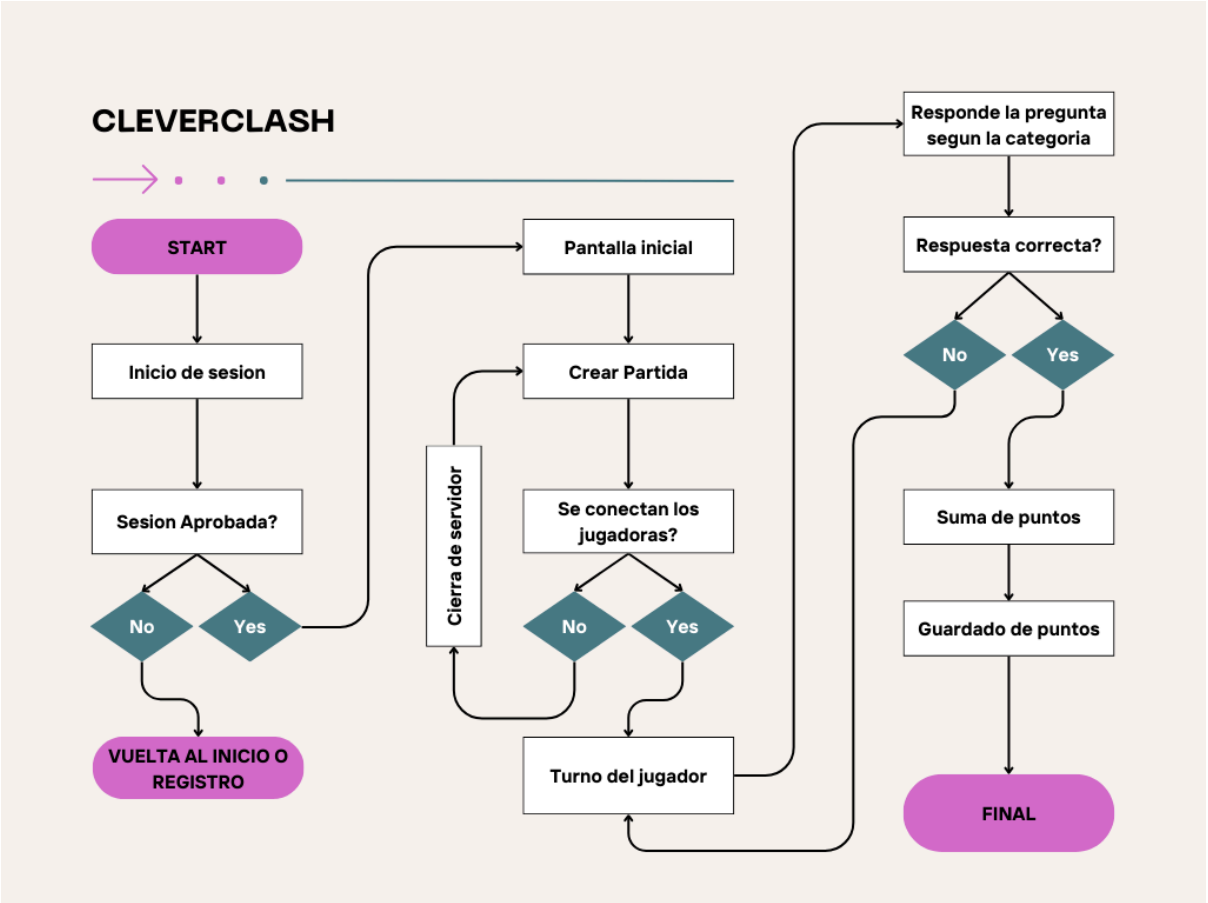
Pruebas Funcionales

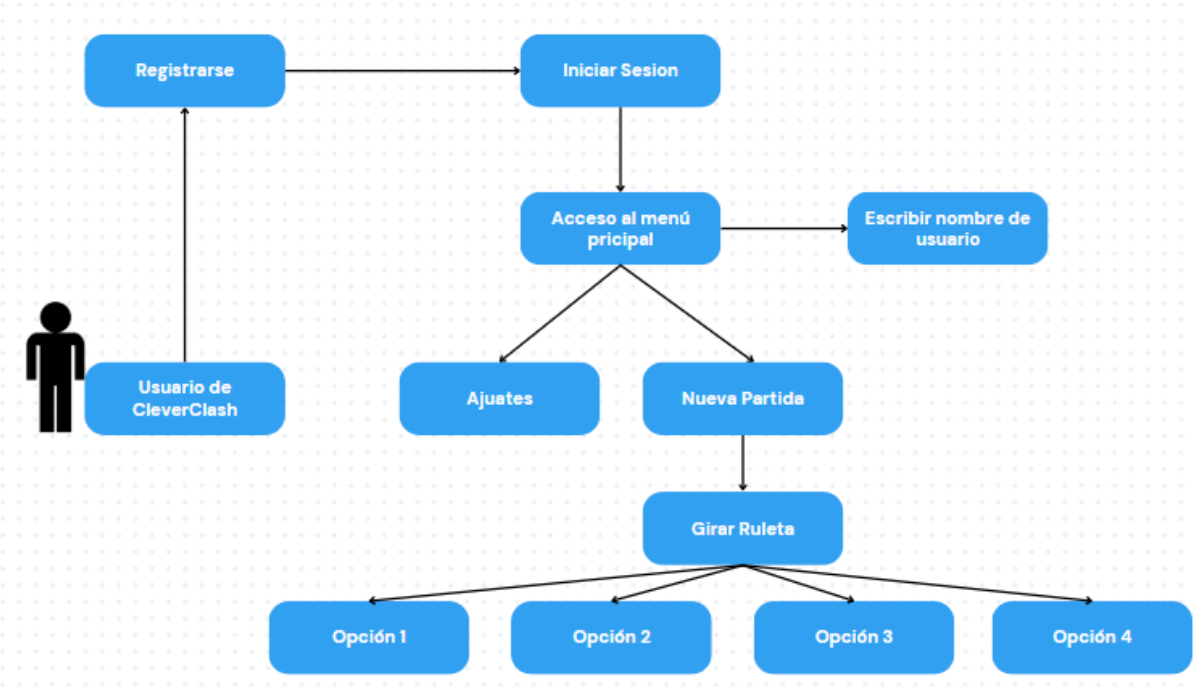
Para realizar una prueba unitaria en el código proporcionado, necesitas enfocarte principalmente en la lógica detrás de los botones y la configuración de la interfaz de usuario. Sin embargo, como en este caso se trata de una interacción con la interfaz gráfica y `Toast` (un componente de Android), una prueba unitaria tradicional no podría cubrir estos casos visuales de manera eficiente.

En cambio, una prueba de unidad podría centrarse en asegurarse de que la lógica detrás de los eventos de los botones esté funcionando correctamente, que el texto de las preguntas se muestre de manera apropiada y que la variable `userName` esté correctamente configurada.

Para probar la lógica básica, puedes usar un framework como **JUnit** junto con **Mockito** para simular las interacciones en el `Activity`.

Diagramas de flujo y actividad





Ejemplo de prueba unitaria con JUnit y Mockito

Supongamos que estamos probando la configuración de los botones y que cada `Toast` es invocado correctamente. Puedes hacerlo de la siguiente manera:

1. **Crear una clase de prueba** para `CienciaActivity`

```java
public class CienciaActivityTest {  no usages
    public class CienciaActivityTest {

        private CienciaActivity activity;  5 usages

        @Mock  2 usages
        private TextView questionBox;
        @Mock  2 usages
        private TextView userNameView;
        @Mock  2 usages
        private Button btnGreen, btnYellow, btnBlue, btnRed;

        @Before
        public void setUp() {
            // Inicializamos los mocks
            MockitoAnnotations.initMocks(this);

            // Abrir la actividad en el escenario de prueba
            ActivityScenario.launch(CienciaActivity.class).onActivity(activity -> {
                this.activity = activity;
                activity.questionBox = questionBox;
                activity.userNameView = userNameView;
                activity.btnGreen = btnGreen;
                activity.btnYellow = btnYellow;
                activity.btnBlue = btnBlue;
                activity.btnRed = btnRed;
            });
```

Explicación de la prueba:

1. **Mocking**: Se usa Mockito para crear mocks de los componentes de la interfaz de usuario (`TextView` y `Button`), ya que en una prueba unitaria no podemos interactuar realmente con la UI de Android.
2. **Prueba del nombre de usuario (`testUserNameSetup`)**: Esta prueba asegura que el nombre del usuario que se configura en `TextView` coincida con la cadena "Usuario 1".
3. **Prueba de la configuración de la pregunta (`testQuestionSetup`)**: Se asegura de que la pregunta que aparece en el `TextView` sea la que se espera, "¿Qué elemento tiene como símbolo químico 'O'?".

4. **Prueba de interacción con botones**: Se simula la interacción con los botones (tanto el correcto como el incorrecto) y se verifica que se haya llamado al `Toast.makeText()` (aunque no es posible probar el texto del `Toast` directamente, se puede verificar que la actividad haya intentado ejecutarlo).

Behance

https://www.behance.net/gallery/219198581/Proyecto-CleverClash/modules/1249419661