UNIVERSIDAD POLITECNICA DE TECAMAC



PROYECTO TIMBIRICHE

PROGRAMACIÓN PARA MÓVILES II

DOCENTE: TORRES SERVIN EMMANUEL.

ESTUDIANTES: ALEMAN ELIZALDE BRANDON JESUS. ROBLES FLORES AARON ERUBIEL.

MATRICULAS: 1318094377. 1319104664.

GRUPO:1922IS.

CUATRIMESTRE: NOVENO.

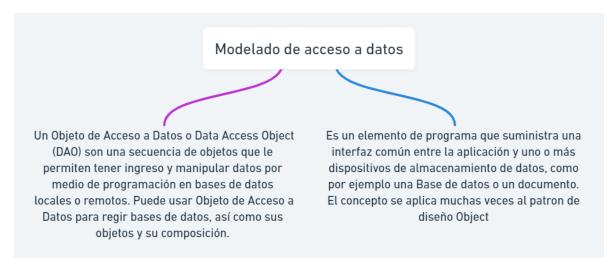
CICLO ESCOLAR: MAYO- AGOSTO 2022.

Contenido

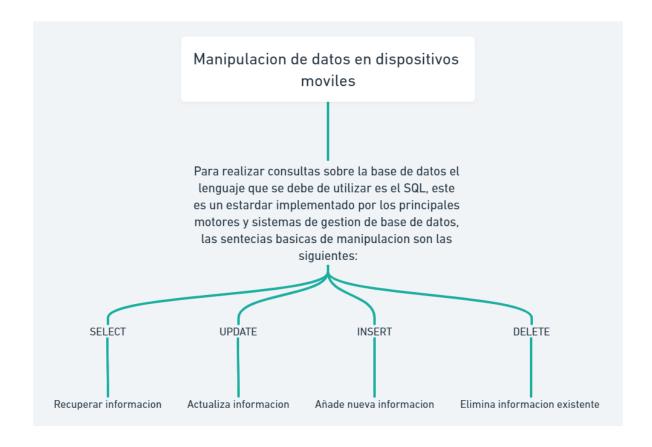
ED	D: COMPRENSIÓN DE LA INFORMACIÓN, PROBLEMATICA	Υ	
CC	COMPONENTES DEL CASO		
E	Explica que es el modelado de acceso a datos	3	
	Explica que es y cómo funciona la manipulación de datos en dispositiv		
	Describe lo que es y conlleva la persistencia de datos en los dispositiv	os	
r	móviles	4	
E	Explica qué y cuáles son los mecanismos de tolerancia a fallos	5	
Pro	oyecto timbiriche	6	
F	Planteamiento del problema	6	
(Objetivo	6	
E	Especificaciones para el desarrollo de la aplicación	6	
	Desarrollo	7	
(Conclusión	12	
Re	eferencias	12	

ED: COMPRENSIÓN DE LA INFORMACIÓN, PROBLEMATICA Y COMPONENTES DEL CASO.

Explica que es el modelado de acceso a datos



Explica que es y cómo funciona la manipulación de datos en dispositivos móviles



Describe lo que es y conlleva la persistencia de datos en los dispositivos móviles.

PERSISTENCIA DE DATOS

En el ámbito de aplicaciones.

consiste en almacenar los datos en un medio secundario, no volátil para posterior reconstrucción y utilización, Son independientes en el tiempo del proceso que los creó.

En el ámbito de las aplicaciones

En pocas palabras consiste en que los datos no se borren luego de que la aplicación se cierre.

La Persistencia en Android

en Android la persistencia de datos principalmente conlleva a tres tipos de datos de almacenamiento específicos,

Preferencias Compartidas o Shared Preferences

Se puede almacenar y recuperar en el formato clavevalor información como texto, booleanos y números; lo que da la pauta para almacenar configuraciones del usuario como estilos y preferencias.

modos de acceso

MODE_PRIVATE: Sólo nuestra aplicación tiene acceso a estas preferencias.

MODE_WORLD_READABLE: Todas las aplicaciones pueden leer estas

preferencias, pero sólo la nuestra puede modificarlas (deprecated desde el API

12)

MODE_WORLD_WRITEABLE: Todas las aplicaciones pueden leer y modificar estas preferencias (deprecated desde el API 17).

Almacenar archivos en memoria

consiste en guardar y recuperar la información en archivos, además que te permite escribir y leer archivos que se encuentren ubicados en la propia Memoria Interna del dispositivo

Memoria Interna

son guardados en la Memoria Interna del dispositivo la cual puede ser limitada en el dispositivo y puede ralentizar el mismo. MODE_PRIVATE: Solo es asesible por la aplicación y por nadie ni nada más (crea el archivo o lo sobrescribe si ya eviste)

MODE_APPEND: Añade contenido a un archivo existente en el dispositivo. MODE_WORLD_READABLE: Permite que otras aplicaciones puedan leer el archivo.

MODE_WORLD_WRITEABLE: Permite que otras aplicaciones puedan escribir el archivo.

Se puede guardar archivos en la Memoria Externa (en la mayoria de los casos, la memoria SD) con los permisos solicitados en el Manifest; los archivos son accesibles para todos: aplicaciones y usuario; por lo tanto no es recomendable utilizarlos para almacenar información sensible.

modos de acceso

Memoria Externa

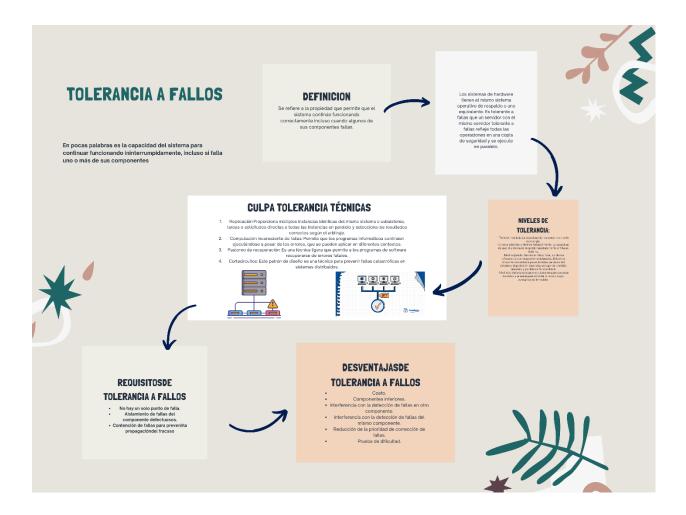
modos de acceso

Environment.MEDIA_MOUNTED: Indica que la memoria esta disponible y "montada" en el dispositivo: por lo tanto podemos acceder a la misma

Environment.MEDIA_MOUNTED_READ_ONLY: Indica que la memoria esta disponible v "montada" en el dispositivo; pero sólo pr leer en ella.

Environment.MEDIA_UNMOUNTED, Environment.MEDIA_REMOVED... Los estatus indican que la memoria no disponible y que por lo tanto no podemos ni leer ni escribir en la misma.

https://www.canva.com/design/DAFB3McmM7A/ 0q2000QPiBMyNHy_1sV9mA/view?utm_content= DAFB3McmM7A&utm_campaign=designshare&ut m_medium=link&utm_source=homepage_design_ menu Explica qué y cuáles son los mecanismos de tolerancia a fallos.



https://www.canva.com/design/DAFB5Z7QvGI/1KMKyf0fWdIJq6Sli13Vrg/view?utm_content=DAFB5Z7QvGI&utm_campaign=designshare&utm_medium=link&utm_source=homepage_design_men_u_

Proyecto timbiriche.

Planteamiento del problema.

Se plantea la elaboración de un juego(timbiriche) con los conocimientos previos para poder identificar las competencias obtenidas y alcanzadas por cada uno de los integrantes del equipo, se plantea llevar a cabo un timbiriche para dispositivos móviles con el objetivo de comprender mejor las herramientas que se utilizaron para llegar a la finalización del proyecto solicitado.

Objetivo.

Identificar los conocimientos obtenidos por los estudiantes con la elaboración de un proyectó, mantener una vista atractiva y de fácil manipulación siendo uno de los elementos clave para elaborar una aplicación de calidad.

Especificaciones para el desarrollo de la aplicación.

La elaboración de la App móvil se llevó a cabo en el IDE Android Studio con el lenguaje de programación Java, el cual posteriormente se subió a un controlador de versiones en este caso GitHub.

Android Studio.

- Chipmunk | 2021.2.1
- Java.

GitHub

https://github.com/BrandonAlemanE11/Timbi

Desarrollo

El principal paso para llevar a cabo la realización del juego fue la elaboración de un diseño previo en donde se llevará a cabo toda la interacción del jugador o jugadores, esto se llevó a cabo en el archivo Activity_main.xml el cual lleva todo el código y diseño llevado a cabo para la utilización de este en el cual se describe desde id de botones para su posterior uso estilos etc.

Código de un TextView donde se mostrará texto.

Código de un botón.

```
<Button
       android:id="@+id/btn1"
       android:layout_width="95dp"
       android:layout_height="48dp"
       android:backgroundTint="@color/black"
       android:onClick="btn1"
       android:text="-----
       android:textColor="@color/material_dynamic_neutral30"
       app:layout_constraintBottom_toBottomOf="parent"
       app:layout_constraintEnd_toEndOf="parent"
       app:layout_constraintHorizontal_bias="0.126"
       app:layout_constraintStart_toStartOf="parent"
       app:layout_constraintTop_toTopOf="parent"
       app:layout_constraintVertical_bias="0.016"
       tools:ignore="MissingConstraints" />
```

Después de este proceso se procede a elaborar la parte lógica del proyecto la cual la elaboramos en MainActivity.java donde declaramos variables determinando el tipo e indicando su valor inicial del cada una.

```
Button btn1,btn2,btn3,btn4,btn5, btn7,
btn9,btn6,btn8,btn10,btn11,btn15,btn16,btn14,btn13,btn19,btn20,btn21,btn22,btn25,btn23,btn24,btn18,btn17;

TextView txtjugador1, txtjugador2;
float turno = 1;
int cuadro1 = 0;
int cuadro2 = 0;
int cuadro3 = 0;
int cuadro3 = 0;
int cuadro5 = 0;
int cuadro6 = 0;
int cuadro6 = 0;
int cuadro7 = 0;
int cuadro8 = 0;
int cuadro9 = 0;
int jugador1 = 0;
int jugador2 = 0;
```

Se lleva a cabo la creación de los botones con el comando "btn1 = findViewByld(R.id.btn1);" específicamente se trae el botón y se asigna a una variable para lo que se debe obtener el id y con la variable correspondiente se manipula su función

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

    btn1 = findViewById(R.id.btn1);
    btn2 = findViewById(R.id.btn2);
    btn3 = findViewById(R.id.btn3);
    btn4 = findViewById(R.id.btn3);
    btn5 = findViewById(R.id.btn5);
    btn7 = findViewById(R.id.btn5);
    btn8 = findViewById(R.id.btn6);
    btn8 = findViewById(R.id.btn8);
    btn10 = findViewById(R.id.btn8);
    btn110 = findViewById(R.id.btn1);
    btn13 = findViewById(R.id.btn1);
    btn14 = findViewById(R.id.btn13);
    btn15 = findViewById(R.id.btn13);
    btn16 = findViewById(R.id.btn15);
    btn16 = findViewById(R.id.btn16);
    btn17 = findViewById(R.id.btn17);
    btn19 = findViewById(R.id.btn17);
    btn19 = findViewById(R.id.btn19);
    btn20 = findViewById(R.id.btn20);
    btn21 = findViewById(R.id.btn20);
    btn22 = findViewById(R.id.btn22);
    btn23 = findViewById(R.id.btn22);
    btn24 = findViewById(R.id.btn23);
    btn24 = findViewById(R.id.btn23);
    btn24 = findViewById(R.id.btn23);
    btn25 = findViewById(R.id.btn21);
    btn26 = findViewById(R.id.btn23);
    btn27 = findViewById(R.id.btn21);
    btn28 = findViewById(R.id.btn23);
    btn29 = findViewById(R.id.btn23);
    btn24 = findViewById(R.id.btn18);

    txtjugador1 = findViewById(R.id.txtjugador1);
    txtjugador2 = findViewById(R.id.txtjugador2);
```

Se procede a la elaboración de los onclick o métodos que escucharan cada que se presione determinada función en donde se determina un incremento de una variable para poder determinar cuando se completara un cuadro, de igual manera se deshabilita el botón, cuenta con una condición(if) la cual determina en que turno se encuentra si el jugador 1 o 2 es el tirador incrementando y decrementando esa variable para que se lleve a cabo el control especifico, se observa un método que se explicara posteriormente.

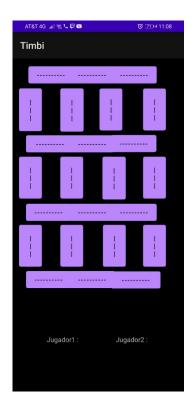
```
public void btn1(View view) {
       if(turno == 1 || turno == 1.5){
            cuadro1 = cuadro1 + 1;
           turno = (float) (turno + 0.5);
           Comprobar();
           btn1.setEnabled(false);
       else{
            if(turno == 2){
                cuadro1 = cuadro1 + 1;
               btn1.setEnabled(false);
               turno = (float) (turno - 2);
               Comprobar();
            }
           else{
                if(turno == 0){
                    cuadro1 = cuadro1 + 1;
                    btn1.setEnabled(false);
                    turno = (float) (turno + 1);
                    Comprobar();
               }
          }
       }
   }
```

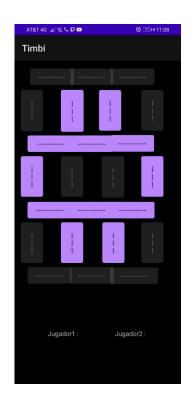
El método anteriormente mencionado es el de comprobar, este método tiene como principal función el trabajo de identificar cuando se completa un cuadro esto por medio de una variable la cual al llegar a 4 asignara un punto a el jugador correspondiente esto determinando diferentes cuadros y de igual forma al llevar a cabo esta acción se encarga de dar o regresar los valores a un punto en el cual al jugador tenga nuevamente su turno y pueda tirar dos veces más también muestra el puntuaje que se lleva en caso de cada jugador.

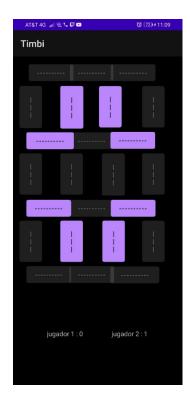
```
private void Comprobar() {
    if(cuadro1 == 4){
        if(turno == 0 || turno == 2){
            turno = 2;
            jugador1 = jugador1 + 1;
            txtjugador1.setText("Jugador 2 : " + jugador1);
            txtjugador2.setText("Jugador 2 : " + jugador2 );
    }
    if(turno == 1 || turno == 1.5){
        turno = 1;
        jugador2 = jugador2 + 1;
        txtjugador2.setText("jugador 2 : " + jugador2);
        txtjugador1.setText("jugador 1 : " + jugador1);
    }
}
```

App









Conclusión

El proyecto que se realizo ha contribuido de manera importante para identificar los puntos a implementar, así como los conocimientos obtenidos por el equipo.

Conforme la realización de la App móvil fuimos percatando de cosas que no tomábamos en cuenta como diseño o puntos de lógica los cuales fallaban en distintos casos, pero también pudimos detectar algunos puntos clave para establecer procesos determinando funciones y facilitando la elaboración del proyecto lo cual nos da una pauta a continuar con la investigación y posterior elaboración de más App móviles.

Referencias

- Bitwar. (26 de 10 de 2021). *Bitwar*. Obtenido de introducción a la tolerancia a fallas: https://www.bitwarsoft.com/es/a-brief-introduction-to-fault-tolerance.html
- CRUZ, A. (14 de 07 de 2014). *Persistencia en Android Developer*. Obtenido de Desarrollo libre: https://www.desarrollolibre.net/blog/android/persistencia-en-android-developer
- itpn. (s.f.). Administración de datos en dispositivos móviles. . Obtenido de Administración de datos en dispositivos móviles. : http://itpn.mx/recursositics/7semestre/desarrollodeaplicaciones/Unidad% 20V.pdf
- Universidad Autonoma de Barcelona. (07 de 2013). *Tolerancia a fallos basada*en la arquitectura RADIC. Obtenido de Tolerancia a fallos basada en la
 arquitectura RADIC:
 https://www.tesisenred.net/bitstream/handle/10803/120239/mcl1de1.pdf?
 sequence=1