Camilo Guevara A00365251

Salomón Sarria A00365662

Análisis de variables y componentes Age Fighters

Diagrama de clases del control en Android Studio

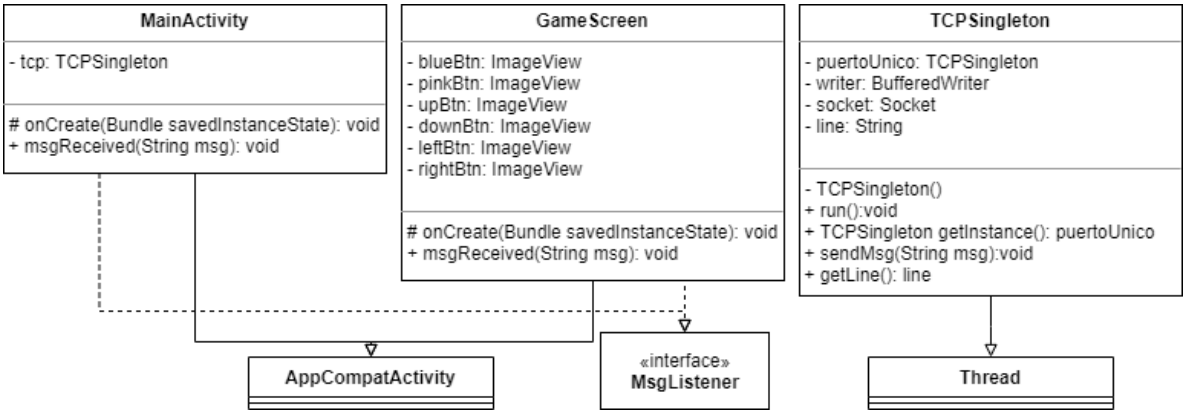
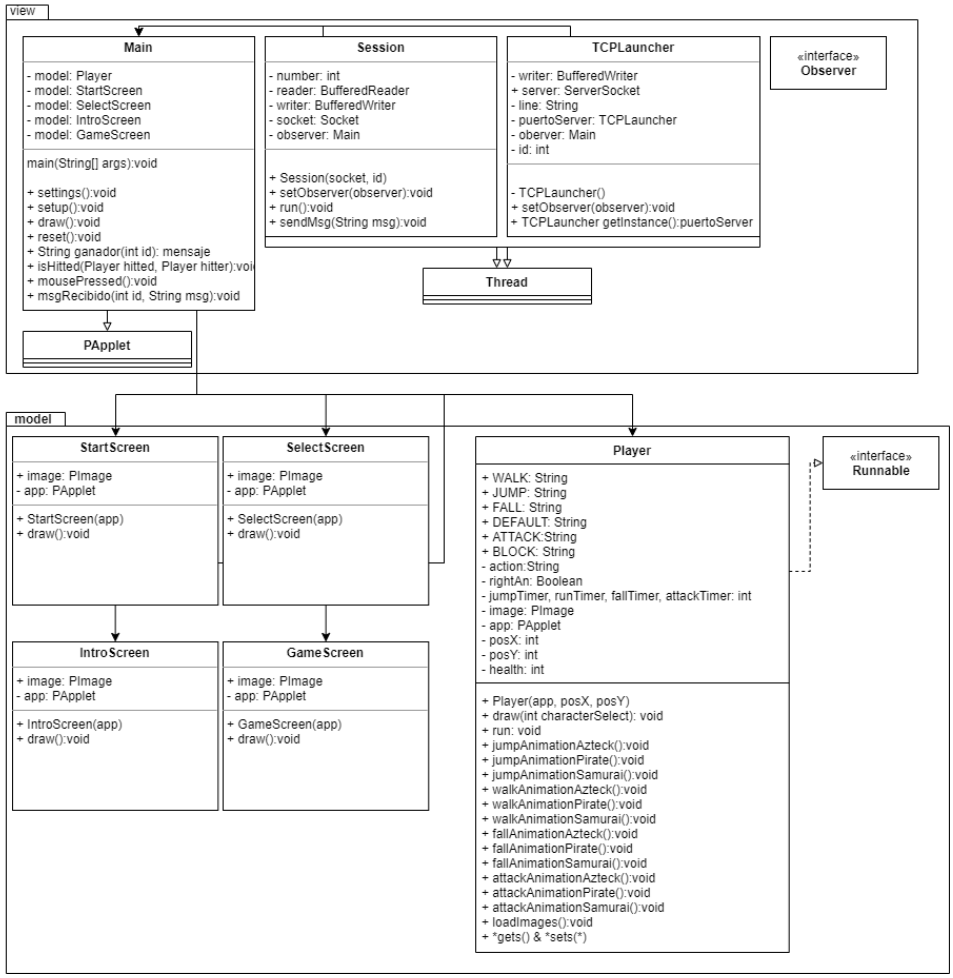


Diagrama de clases del servidor en Eclipse



https://drive.google.com/file/d/1CeRP80rQ1CtEmDlfz0zKvOx-o-dQ7bH9/view?usp=sharing

Análisis de transferencia de mensajes

El control se compone de 6 botones, 4 de movimiento y 2 de acción siendo azul el de ataque y el rosado el de bloqueo.

```
blueBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("blue");
        }
);
pinkBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("pink");
        }
);
upBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("up");
        }
);
downBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("down");
        }
);
```

```
upBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("up");
        }
);
downBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("down");
        }
);
leftBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("left");
        }
);
rightBtn.setOnClickListener(
        (view) -> {
            tcp.sendMsg("right");
        }
```

Cada botón al ser presionado envía un mensaje indicando que botón se presionó, sea el botón de arriba envía un mensaje que diga arriba o up para indicar que este fue presionado, lo mismo con los botones de acción, el botón azul envía un mensaje que dice azul o blue y el botón rosado envía un mensaje que dice rosado o pink.

```
//animaciones

//ataque
if(accion.equals("1:pink")) {
    player1.setAction("ATTACK");
}

if(accion.equals("2:pink")) {
    player2.setAction("ATTACK");
}

//bloqueo

if(accion.equals("1:blue")) {
    player1.setAction("BLOCK");
}

if(accion.equals("2:blue")) {
    player2.setAction("BLOCK");
}

//caminar

if(accion.equals("1:right")) {
    player1.setRightAn(true);
    player1.setAction("WALK");
}
```

```
if(accion.equals("2:right")) {
    player2.setRightAn(true);
    player2.setAction("WALK");
}

if(accion.equals("2:left")) {
    player2.setRightAn(false);
    player2.setAction("WALK");
}

//saltar

if(accion.equals("1:up")&&p1jump==false) {
    p1jump=true;
    player1.setAction("JUMP");
    p1jump= false;


}

if(accion.equals("2:up")&&p2jump==false) {
    p2jump=true;
    player2.setAction("JUMP");
    p2jump= false;

}
```

Una vez se presiona el botón estos llegan al método de recibir mensajes en el servidor de eclipse, en donde se interpretan estos mensajes con la correspondiente acción que realiza un personaje.

```
public final static String WALK = "WALK";
public final static String JUMP = "JUMP";
public final static String FALL = "FALL";
public final static String DEFAULT = "DEFAULT";
public final static String ATTACK = "ATTACK";
public final static String BLOCK = "BLOCK";
```

Las acciones de cada personaje están definidas por una serie de diferentes strings los cuales luego hacen parte de un switch que se encarga de cambiar de acción a la hora que un botón se presione.

```
            }
        if(rightAn == false) {
            app.image(S, posX, posY);
            }
        break;
    case JUMP:
        jumpAnimationSamurai();
        posY -= 4;
        break;
    case FALL:
        fallAnimationSamurai();
        break;
    case WALK:
        walkAnimationSamurai();
        if(rightAn == true) {
            posX +=4;
        }
        if(rightAn == false) {
            posX -=4;
        }
        break;
    case ATTACK:
        attackAnimationSamurai();
        break;
    case BLOCK:
        if(rightAn == true) {
            app.image(SBd, posX, posY);
            }
```

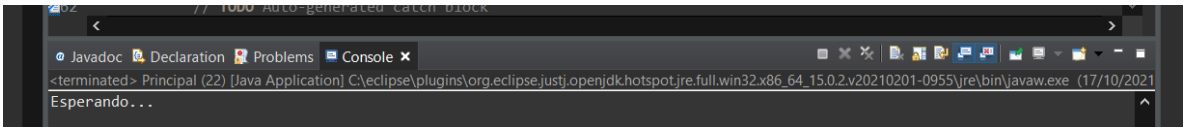
Adicionalmente, para cambiar de pantallas, seleccionar el personaje y reiniciar el juego se escogió utilizar el botón rosado.

```
    public void msgRecibido(int id, String msg) {
        accion = id + ":" + msg;
switch (screen) {


        case 0:
            if(accion.equals("1:pink")) {player1ok=true;}
            if(accion.equals("2:pink")) {player2ok=true;}
            if(player1ok==true) {
                screen=1; ← Cambiar Pantalla
                player1ok=false;
                player2ok=false;}

            break;
        case 1:
        //controles
            if(accion.equals("1:right")){p1C++;}
            if(accion.equals("1:left")){p1C--;}
            if(accion.equals("2:right")){p2C++;}
            if(accion.equals("2:left")){p2C--;}
        //selector img y condicionales
             if(p1C>2) {p1C=0;} if(p2C>2) {p2C=0;}
             if(p1C<0) {p1C=2;} if(p2C<0) {p2C=2;}
             if(p1C==0) {xp1=194;} if(p2C==0) {xp2=194;}
             if(p1C==1) {xp1=640;} if(p2C==1) {xp2=640;}
             if(p1C==2) {xp1=1097;} if(p2C==2) {xp2=1097;}
             //continuar
```

Además, en el proceso de conexión TCP hay una seria de intercambio de mensajes que muestran el proceso de conexión tanto en el servidor como en el control. Mientras ambos están esperando a una conexión, en el servidor aparece un mensaje en la consola que dice “Esperando…” y en el control aparece una pantalla inicial que muestra que se esta esperando a que se realice una conexión con el servidor.



Control:

Finalmente, el servidor también se encarga en enviar mensajes al control, primeramente, el servidor envía un mensaje que indica la conexión efectiva entre el servidor y el control, al llegar este mensaje el control pasa de la pantalla de espera a la pantalla que muestra el control. Y en adición al terminar el juego el servidor envía un mensaje a los controles indicando quien es el ganador.

```
new Thread(
        ()->{
            try {
                writer.write("conectado"+"\n");
                writer.flush();
            } catch (IOException e) {
                e.printStackTrace();
            }

        }
).start();



while(true) {
    String line = reader.readLine();
    observer.msgRecibido(number,line, this);
    System.out.println(line);
    sendMsg(observer.ganador(number));}
```

```
@Override
public void msgReceived(String msg) {
    if(msg.equals("conectado")){
        Intent i = new Intent( packageContext: this,GameScreen.class);
        startActivity(i);}
}
```

```
@Override
public void msgReceived(String msg) {
    Log.e( tag: "mensaje",msg);
    runOnUiThread(()->{
        if(msg!=null){
        Toast.makeText( context: this,msg,Toast.LENGTH_SHORT).show();}
    });
}
```