LIBGDX Cambiando de pantalla

UNIDADE 2: Cambiando de pantalla

Cambiando de pantalla

Tal como comentamos anteriormente neste punto a forma de proceder para cambiar de pantalla é chamando ó método setScreen da **clase Game**.

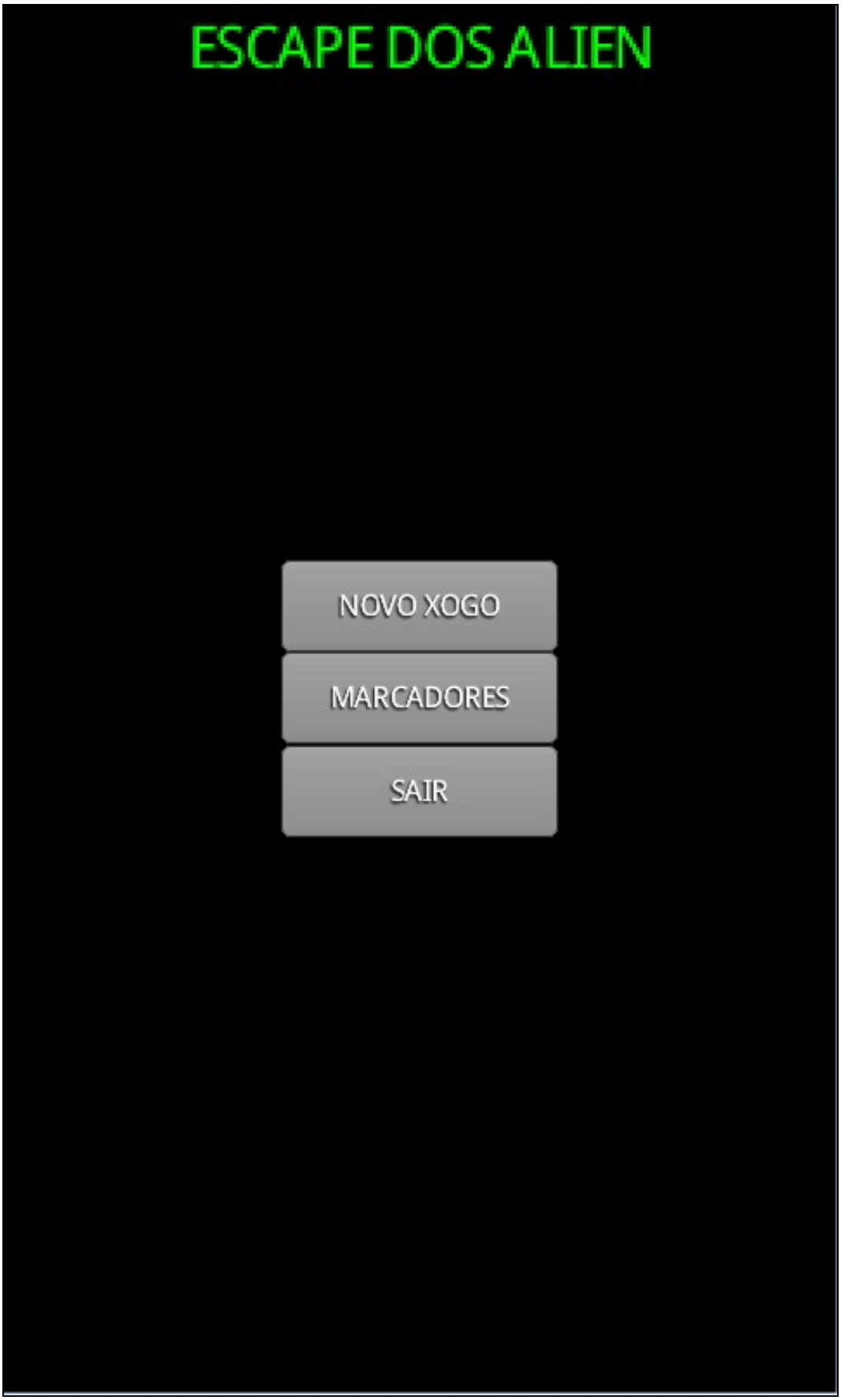
Nota: Lembrar que no exemplo que estamos a seguir, mandamos o obxecto que deriva da clase Game como parámetro na chamada ó método setScreen, para que a outra pantalla poda ter referencia ó mesmo e poida volver a chamar ó método setScreen.

No noso exemplo....

Imos cambiar a pantalla inicial para que en vez de ir ó xogo directamente pase pola pantalla principal.

Preparación:

- Subide o seguinte gráfico ó cartafol GRAFICOS dentro do cartafol assets no proxecto Android.


ESCAPE DOS ALIEN
NOVO XOGO
MARCADORES
SAIR

- Agora imos modificar a **clase PantallaPresentacion** para crear unha cámara e visualizar o gráfico anterior.

Código da clase PantallaPresentacion
Obxectivo: visualizar a pantalla de presentación.

```
public class PantallaPresentacion implements Screen, InputProcessor{
private MeuXogoGame meuxogogame;

    private OrthographicCamera camara2d;
    private SpriteBatch spritebatch;
    private static Texture fondo;

public PantallaPresentacion(MeuXogoGame meuxogogame){
this.meuxogogame=meuxogogame;

camara2d = new OrthographicCamera();
spritebatch = new SpriteBatch();
fondo = new Texture(Gdx.files.internal("GRAFICOS/LIBGDX_itin1_pantallapresentacion.png"));

}

@Override
public void render(float delta) {
// TODO Auto-generated method stub

spritebatch.begin();

spritebatch.draw(fondo,0,0,Mundo.TAMANO_MUNDO_ANCHO,Mundo.TAMANO_MUNDO_ALTO);

spritebatch.end();


}

@Override
public void resize(int width, int height) {
// TODO Auto-generated method stub

camara2d.setToOrtho(false, Mundo.TAMANO_MUNDO_ANCHO, Mundo.TAMANO_MUNDO_ALTO);
camara2d.update();

spritebatch.setProjectionMatrix(camara2d.combined);
spritebatch.disableBlending();

}

@Override
public void show() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(this);

}

@Override
public void hide() {
// TODO Auto-generated method stub
                // Neste caso non fai falta poñelo xa que imos ser nos o que chamemos a dispose cando cambiemos de pantalla.
//dispose();

}

@Override
public void pause() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(null);

}

@Override
public void resume() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(this);
```

```
}

@Override
public void dispose() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(null);

spritebatch.dispose();
fondo.dispose();

}

@Override
public boolean keyDown(int keycode) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean keyUp(int keycode) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean keyTyped(char character) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean touchDown(int screenX, int screenY, int pointer, int button) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean touchUp(int screenX, int screenY, int pointer, int button) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean touchDragged(int screenX, int screenY, int pointer) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean mouseMoved(int screenX, int screenY) {
// TODO Auto-generated method stub
return false;
}

@Override
public boolean scrolled(int amount) {
// TODO Auto-generated method stub
return false;
}


}
```

- **Liña 38**:

Atención:: Na liña 38 temos a seguinte instrución:

```
spritebatch.disableBlending();
```

O que fai dita liña é deshabilitar o uso de transparencias. Isto o facemos nesta pantalla xa que o fondo da mesma ocupa todo e non imos engadir novos

gráficos que teñan transparencias.

No resto de pantallas (como a de Marcadores) imos escribir texto que vai ter transperencias polo que dita orde non se debe poñer ou se a poñemos (por ter un fondo por exemplo) deberemos chamar a orde contraria (spritebatch.enableBlending()) antes de debuxar o gráfico con transparencias.

Modificamos agora a clase principal **MeuXogoGame** para que chame á nova pantalla...

Código da clase MeuXogoGame
Obxectivo: chama á pantalla principal.

```
public class MeuXogoGame extends Game {


@Override
public void create() {
// TODO Auto-generated method stub

AssetsXogo.cargarTexturas();
setScreen(new PantallaPresentacion(this));
}

@Override
public void dispose(){
super.dispose();

AssetsXogo.liberarTexturas();
}
}
```

Agora temos que comprobar se pulsamos cada unha das opcións (botóns) e obrar en consecuencia. Debemos crear un rectángulo por cada unha das opcións (xa visto na sección das colisións) e comprobar se o dedo toca cada unha delas.

TAREFA 2.10 A FACER: Esta parte está asociada á realización dunha tarefa.

Facendo pause

En case todos os xogos podemos facer un pause do xogo para seguir posteriormente así como a opción de saír.

No caso do pause temos dúas opcións:

- Quedarnos na mesma pantalla e no caso de que o xogo pase ó estado de pause debuxar un gráfico no centro da pantalla que indique que está en pause.

Para esta opción só teríamos que **non chamar** ó método controladorXogo.update(delta) dentro do método render da clase PantallaXogo. Desta forma ningún personaxe se moverá. Teríamos que cambiar a variable booleana pause a true e facela static e public para que dende a clase RendererXogo poidamos verificar o seu valor e no caso de que valga true poñer o gráfico de pause no centro da pantalla. Teríamos que controlar o método touchdown para que no caso de estar en pause e tocar a pantalla volver a poñer o valor a false.

- Cambiar de pantalla e que ó premer volvamos ó xogo no punto onde o deixamos.

Para esta segunda opción imos modificar a clase PantallaPause para cargar o seguinte gráfico:


ESCAPE DOS ALIEN
PULSA PARA SEGUIR XOGANDO

Levade este gráfico ó cartafol assets/GRAFICOS da versión Android e dádelle de nome LIBGDX_itin1_pantallapause.jpg.

Modificade o código da clase PantallaPause para que cargue o gráfico anterior.

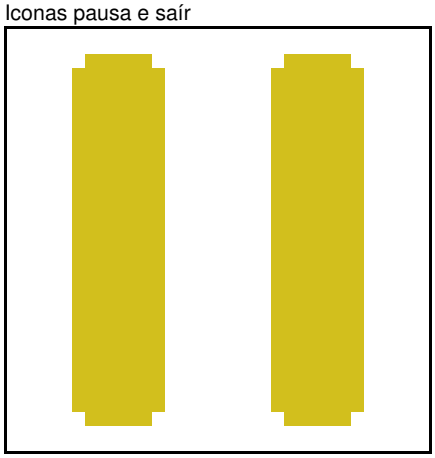
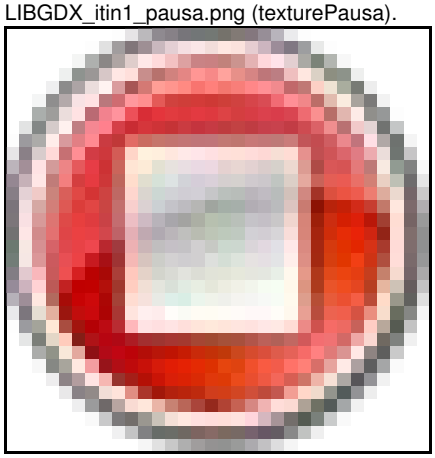
Lembrar incluír a interface InputProcessor como xa vimos anteriormente.

Tamén deberedes liberar os recursos cando pase polo método hide (chamar ó método dispose nese intre e liberade a textura e o spritebach).

Preparando a clase PantallaXogo e RendererXogo:

Modificamos agora o código da RendererXogo para que visualice dúas iconas na parte inferior esquerda.

Aquí temos os gráficos a engadir no cartafol assets/GRAFICOS/CONTROIS da versión Android:

- Iconas pausa e saír
- 
  LIBGDX_itin1_pausa.png (texturePausa).
- 
  LIBGDX_itin1_sair.png (textureSair).

Como sempre os cargarmos na clase AssetsXogo cos nome indicados entre parénteses.

Despois modificamos a clase RendererXogo para que debuxe os novos controis. Como imos ter que indicar onde debuxalos podemos definir a súa posición e tamaño na clase Controis.

Código da clase Controis
Obxectivo: definir posición e tamaño das iconas pausa e saír.

```
public class Controis {

public final static Rectangle FONDO_NEGRO = new Rectangle(0, 0,
Mundo.TAMANO_MUNDO_ANCHO, 12);
public final static Rectangle CONTROL = new Rectangle(10, 40, 50, 70);
public final static int POSVIDAS = 60;

public final static Rectangle CONTROL_PAUSE = new Rectangle(30,0,10,10);
```

```
public final static Rectangle CONTROL_SAIR = new Rectangle(45,0,10,10);


}
```

Código da clase RendererXogo
Obxectivo: visualiza as iconas pausa e saír.

```
private void debuxarControis(){

// Fondo negro
spritebatch.draw(AssetsXogo.texturePuntoNegro, Controis.FONDO_NEGRO.x,Controis.FONDO_NEGRO.y,Controis.FONDO_NEGRO.width,Controis.FONI

spritebatch.draw(AssetsXogo.control, Controis.CONTROL.x,Controis.CONTROL.y,
         Controis.CONTROL.width,Controis.CONTROL.height);

spritebatch.draw(AssetsXogo.texturePausa, Controis.CONTROL_PAUSE.x,Controis.CONTROL_PAUSE.y,Controis.CONTROL_PAUSE.width,Controis.CO
spritebatch.draw(AssetsXogo.textureSair, Controis.CONTROL_SAIR.x,Controis.CONTROL_SAIR.y,Controis.CONTROL_SAIR.width,Controis.CONTRO
}
```

Agora controlamos se prememos a opción de pausa e saír.

Código da clase PantallaXogo
Obxectivo: xestionamos a opción de premer pausa e saír.

```
@Override
public boolean touchDown(int screenX, int screenY, int pointer, int button) {

//              if (Gdx.app.getType()!=ApplicationType.Android) return false;

// TODO Auto-generated method stub
Vector3 vecTemporal = new Vector3(screenX,screenY,0);
rendererXogo.getCamara2d().unproject(vecTemporal);

                Rectangle recTemporal = new Rectangle();

                ........................
                // Falta o código feito na tarefa 2.8.B no que se controla cando prememos no control do alien


recTemporal.set(Controis.CONTROL_PAUSE.x,Controis.CONTROL_PAUSE.y,Controis.CONTROL_PAUSE.width,Controis.CONTROL_PAUSE.height);
if (Intersector.overlaps(dedo, recTemporal)){
pause = true;
}

recTemporal.set(Controis.CONTROL_SAIR.x,Controis.CONTROL_SAIR.y,Controis.CONTROL_SAIR.width,Controis.CONTROL_SAIR.height);
if (Intersector.overlaps(dedo, recTemporal)){
dispose();
meuXogoGame.setScreen(new PantallaPresentacion(meuXogoGame));
}



return false;
}
```

Como podemos comprobar a opción de saír non ten complicación, xa a temos vista na tarefa 2.10. A chamada ó método dispose non se atopa no método hide xa que cando vaiamos á pantalla de pausa non queremos 'borrar' nada do xogo. Queremos ir con todo o estado do xogo e recuperalo ó volver.

Como facemos isto ? Como sempre temos varias opcións...Algunha delas:

- Poderíamos gardar o estado do xogo a disco e recuperalo ó volver. Esta opción non é aconsellable nunha pantalla de pausa e si no caso de querer gardar a partida para continuar outro día (neste xogo non ten moito sentido).

- Mandar como parámetro ó constructor da clase PantallaPause o obxecto da clase PantallaXogo desta forma:

Código da clase PantallaPause
Obxectivo: modifcamos o constructor para obter unha referencia á PantallaXogo.

```
public class PantallaPause implements Screen, InputProcessor{
        ..............
        private PantallaXogo pantallaXogo;

public PantallaPause(MeuXogoGame meuXogoGame, PantallaXogo pantallaXogo){

this.meuxogogame=meuxogogame;
this.pantallaXogo = pantallaXogo;

camara2d = new OrthographicCamera();
spritebatch = new SpriteBatch();
fondo = new Texture(Gdx.files.internal("GRAFICOS/LIBGDX_itin1_pantallapause.png"));

}
        ..............
}
```

Agora no método render da clase PantallaXogo controlamos cando estamos en pause para mandar o control á PantallaPause:

Código da clase PantallaXogo
Obxectivo: controlamos se estamos en pause para ir á PantallaPause.

```
@Override
public void render(float delta) {
// TODO Auto-generated method stub

rendererXogo.render(delta);
controladorXogo.update(delta);

if (pause){
meuXogoGame.setScreen(new PantallaPause(meuXogoGame, this));
return;
}
}
```

Nota importante: Debemos facer return despois de chamar a setScreen, xa que a pesar que o control pasa a nova pantalla, se sigue executando o código do método render ata finalizar. Se temos código a continuación podemos ter un erro de execución.

Fixarse como no método setScreen enviamos non só a referencia a clase MeuXogoGame (para poder facer o setScreen) se non tamén unha referencia á propia clase PantallaXogo.

Agora só temos que controlar se se preme na pantalla PantallaPause volver ó control á PantallaXogo, pero usando a referencia pasada.

Código da clase PantallaPause
Obxectivo: devolvemos o control á PantallaXogo se prememos na pantalla.

```
@Override
public boolean touchDown(int screenX, int screenY, int pointer, int button) {
// TODO Auto-generated method stub
meuxogogame.setScreen(pantallaXogo);
return false;
}
```

Se executades o código podedes comprobar que xa accedemos á pantalla de pausa cando prememos sobre o control, pero ó intentar continuar o xogo volve á pantalla de pausa. Isto é debido a que a variable pause segue valendo true cando regresamos e polo tanto volve a facer o setScreen da pantalla de pause.

Imos modificar isto para que o xogo cando entre en estado de pause (ó premer o botón de pause ou cando minimizamos (en desktop) ou cambiamos de aplicación (en móbil)) poñamos a variable a true e cando volvamos (maximicemos ou volvamos nun móbil) cambie a false.

Código da clase PantallaXogo
Obxectivo: modificar a propiedade pause.

```
.................

@Override
public void pause() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(null);
if (!finXogo) {
pause = true;
}

}

@Override
public void resume() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(this);
pause=false;

}
@Override
public void show() {
// TODO Auto-generated method stub
Gdx.input.setInputProcessor(this);
pause=false;
}

        .................
```

Nota: Fixarse como cando facemos a pausa do xogo minimizando a aplicación na versión desktop, pasamos polo evento pause, pero cando restauramos, non pasamos polo evento resume xa que cambiamos de pantalla (estamos na pantalla de pausa). Cando dende a pantalla de pausa chamamos ó método setScreen volvemos a pasar polo evento show.

Saíndo do xogo

Ó premer a icona de saír debemos ir á pantalla de presentación. Tamén imos engadir outra condición e é que se utilizamos máis de 15 vidas o xogo acaba.

Lembrar que temos que liberar os recursos.

Código da clase PantallaXogo
Obxectivo: Xestionar o fin do xogo.

```
@Override
public void render(float delta) {
// TODO Auto-generated method stub

rendererXogo.render(delta);
controladorXogo.update(delta);

if (meuMundo.getAlien().getNumVidas().size>=15)
finXogo=true;


if (pause){
Audio.stopMusica();
meuXogoGame.setScreen(new PantallaPause(meuXogoGame, this));
return;
}
if (finXogo){
meuXogoGame.setScreen(new PantallaMarcadores(meuXogoGame));
// Facemos o return xa que continua a execución ata que remate o render.
return;
}
```

```
}

@Override
public boolean touchDown(int screenX, int screenY, int pointer, int button) {

vecTemporal.set(screenX,screenY,0);
rendererXogo.getCamara2d().unproject(vecTemporal);

dedo.set(vecTemporal.x,vecTemporal.y,2);
                ..................

recTemporal.set(Controis.CONTROL_SAIR.x,Controis.CONTROL_SAIR.y,Controis.CONTROL_SAIR.width,Controis.CONTROL_SAIR.height);
if (Intersector.overlaps(dedo, recTemporal)){
sair=true;
meuXogoGame.setScreen(new PantallaPresentacion(meuXogoGame));
}



return false;
}
@Override
public void hide() {
// TODO Auto-generated method stub
if ((finXogo) || (sair)) dispose();

}
```