1 LIBGDX AssetManager

UNIDADE 3: AssetManager

1.1 Sumario

1.1.1 Introdución

Nota: Esta explicación está relacionada cos gráficos.

Información na wiki: https://github.com/libgdx/libgdx/wiki/Managing-your-assets

O obxectivo desta clase:

- Impedir que o dispositivo móbil dea un **Not Responding**. Isto se pode producir se temos unha cantidade de gráficos moi alta e os cargamos como ata o de agora. A carga dos gráficos se fai no fío principal da aplicación.
- Pode realizar unha carga asíncrona dos gráficos. Desta forma podemos ter unha pantalla de carga cunha barra de progreso para facer o xogo máis amigable.

Esta clase só debería usarse cando o tamaño conxuntos de todos os gráficos sexa elevado.

1.1.2 Pasos para utilizar a clase AssetManager

- Primeiro debemos crear o obxecto AssetManager:

```
private AssetManager assetManager;
..........

assetManager = new AssetManager();
```

- Despois temos que indicarlle o que queremos cargar.

  Isto se fai chamando ó método load.

```
assetManager.load("LIBGDX_asset_mapa1.png", Texture.class);
```

  - O primeiro parámetro é o nome do gráfico a cargar.
  - O segundo parámetro é a clase de gráfico a cargar. Entre estas temos:
    TextureAtlas.class
    BitmapFont.class
    Music.class

  Este método está sobrecargado e podemos enviarlle como terceiro parámetro información extra como o tipo de algoritmo de compresión para os elementos gráficos.

- Indicamos o tipo de carga que queremos:

    ◊ Carga asíncrona: Neste intre a carga se fai asincronamente de tal forma que no método render da clase que está cargando o asset podemos comprobar o progreso da carga chamando ó método update.
    Dito método devolve false no caso de que aínda estean cargando os gráficos.

    Neste caso (devolve false, polo tanto estamos a cargar os gráficos) podemos chamar ó método getProgress() que nos vai devolver un número entre 0-1 indicando o grado de progreso da carga (0 empeza ; 1 rematou de cargar).

    ◊ Carga síncrona: Se queremos que a carga se faga sincronamente, é dicir, que non continúe coa execución ata que remate de cargar, teremos que chamar ó método finishLoading().

```
assetManager.finishLoading();
```

- Unha vez cargados os gráficos temos que referencialos para poder ser usados polo noso xogo.

```
private Texture mapa1;
        .............

mapa1 = assetManager.get("LIBGDX_asset_mapa1.png",Texture.class);
```

- Unha vez os temos referenciados xa podemos usalos como ata o de agora.

```
batch.draw(mapa1,0,0,100,100);
```

- Unha vez que xa non os queremos utilizar temos varias opcións:

    Chamar ó método unLoad(String nomearquivo): se o gráfico está referenciado por outro assetmanager ou cargado manualmente non o libera.
    Chamar ó método clear(): desta forma se liberan todos os recursos gráficos cargados polo assetmanager, pero a diferenza do método dispose podo volver a usar o assetmanager para cargar outros gráficos.
    Chamar ó método dispose(): libera os gráficos e destrúe o assetmanager.

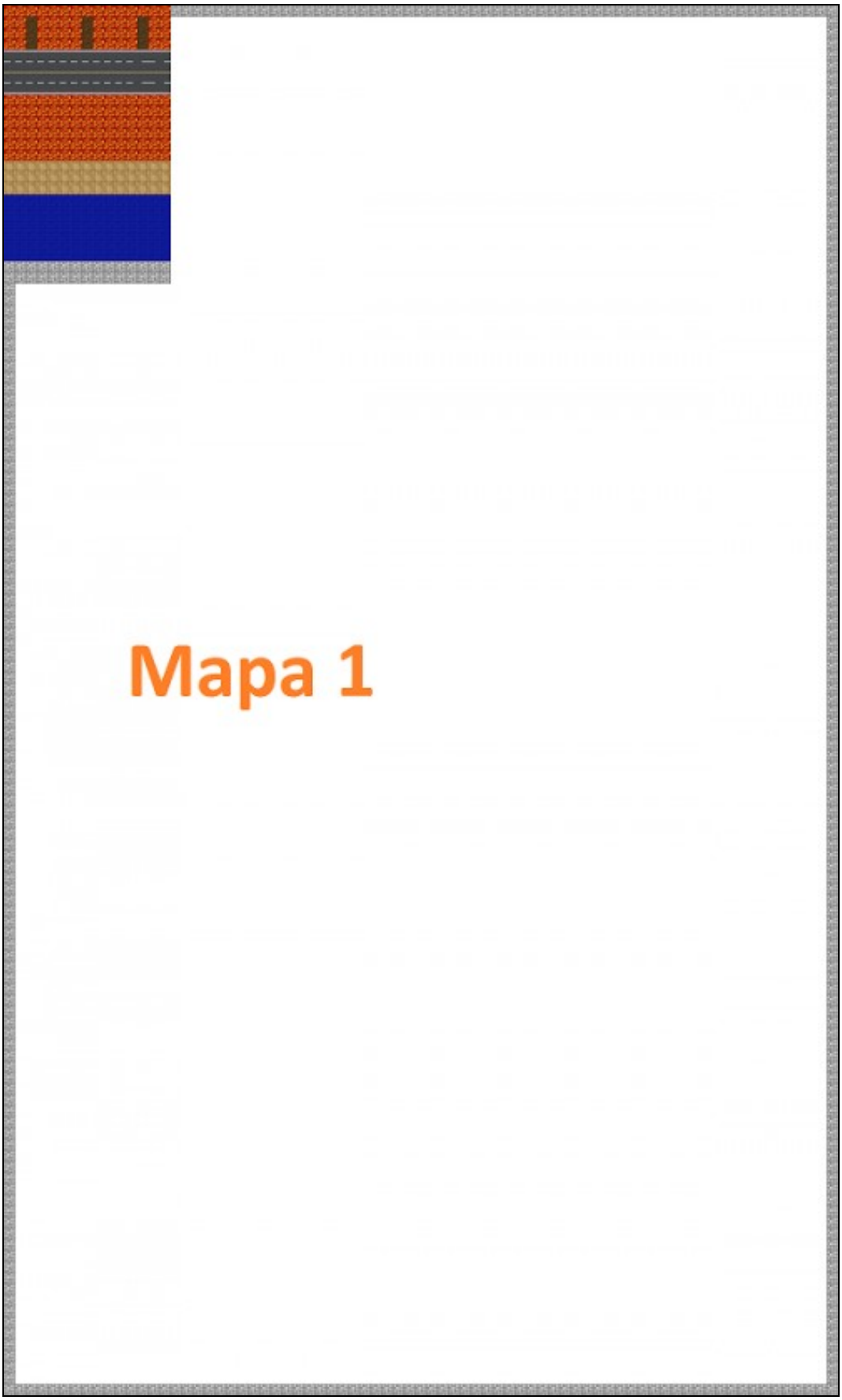
```
assetManager.dispose();
```

1.1.3 Exemplo de código

- Descargade e descomprimide o arquivo seguinte. Copialo ó cartafol assets do proxecto Android.

Media:LIBGDX_assetmanager.zip

Dito arquivo ten 3 mapas de proba, cunha resolución de 2400x4000 pixeles de resolución. Para que vos deades unha idea, unha versión reducida do que levades comprimido é isto:



O que se va na parte superior esquerda é o gráfico que estades a utiliza no voso xogo...

- Crear unha nova clase.

Deberedes de cambiar a clase co que inician as diferentes plataformas pola seguinte:

1.1.3.1 Versión asíncrona

Código da clase AssetManagerCarga
Obxectivo: amosar como utilizar a clase AssetManager. Carga ASÍNCRONA

```
public class AssetManagerCarga  extends ApplicationAdapter {
SpriteBatch batch;

private AssetManager assetManager;

private Texture mapa1;
private Texture mapa2;
private Texture mapa3;


@Override
public void create () {
batch = new SpriteBatch();


assetManager = new AssetManager();

assetManager.load("LIBGDX_asset_mapa1.png", Texture.class);
assetManager.load("LIBGDX_asset_mapa2.png", Texture.class);
assetManager.load("LIBGDX_asset_mapa3.png", Texture.class);

}

@Override
public void render () {
Gdx.gl.glClearColor(1, 0, 0, 1);
Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
batch.begin();
if (assetManager.update()){
mapa1 = assetManager.get("LIBGDX_asset_mapa1.png",Texture.class);
mapa2 = assetManager.get("LIBGDX_asset_mapa2.png",Texture.class);
mapa3 = assetManager.get("LIBGDX_asset_mapa3.png",Texture.class);
Gdx.app.log("LIBGDX","Rematou de cargar");// Se esta é unha pantalla de carga agora cambiaríamos a do xogo
Gdx.app.exit();

}
else {
Gdx.app.log("LIBGDX",String.valueOf(assetManager.getProgress()));
}


batch.end();

}

@Override
public void dispose(){
assetManager.dispose();
}

}
```

1.1.3.2 Versión síncrona

Código da clase AssetManagerCarga
Obxectivo: amosar como utilizar a clase AssetManager. Carga SÍNCRONA

```
public class AssetManagerCarga  extends ApplicationAdapter {
SpriteBatch batch;

private AssetManager assetManager;
```

```
private Texture mapa1;
private Texture mapa2;
private Texture mapa3;


@Override
public void create () {
batch = new SpriteBatch();


assetManager = new AssetManager();

assetManager.load("LIBGDX_asset_mapa1.png", Texture.class);
assetManager.load("LIBGDX_asset_mapa2.png", Texture.class);
assetManager.load("LIBGDX_asset_mapa3.png", Texture.class);

// VERSION SINCRONA
assetManager.finishLoading();

        mapa1 = assetManager.get("LIBGDX_asset_mapa1.png",Texture.class);
mapa2 = assetManager.get("LIBGDX_asset_mapa2.png",Texture.class);
mapa3 = assetManager.get("LIBGDX_asset_mapa3.png",Texture.class);

// Xa cargamos os gráficos.
// No xogo que estamos a facer, esta parte iría na clase AssetManager

}

@Override
public void render () {
Gdx.gl.glClearColor(1, 0, 0, 1);
Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
batch.begin();

// VERSION SINCRONA
batch.draw(mapa1,0,0,Gdx.graphics.getWidth(),Gdx.graphics.getHeight());

batch.end();

}

@Override
public void dispose(){
assetManager.dispose();
}

}
```

1.1.4 TAREFA OPTATIVA A FACER

TAREFA OPTATIVA A FACER: Modifica a clase AssetsXogo para cargar os gráficos do xogo utilizando a clase AssetManager.

Nota:

- Lembra modificar ó método dispose para liberar o assetmanager.
- Podes aumentar o nivel de dificultade creando unha pantalla de carga cunha barra de desprazamento,

---

-- Ángel D. Fernández González -- (2014).