1 LIBGDX TextureAtlas

UNIDADE 3: TextureAtlas

1.1 Sumario

1.1.1 Introdución

Nota: Esta explicación está relacionada cos gráficos.

Información da wiki: https://github.com/libgdx/libgdx/wiki/Texture-packer

Información da clase TextureAtlas: http://libgdx.badlogicgames.com/nightlies/docs/api/com/badlogic/gdx/graphics/g2d/TextureAtlas.html.

Cando cargamos os gráficos do noso xogo estamos accedendo a disco por cada un deles. Existe unha forma máis óptima de carga que consiste en cargar todos os gráficos de vez.

Para facelo o proceso é o seguinte:

- Primeiro temos que deixar nun cartafol todos os gráficos que queiramos xuntar.
- Cun programa externo ou cunha ferramenta de Libgdx indicámoslle onde se atopan os gráficos a xuntar (o cartafol) e cal vai ser o cartafol de saída. Esta ferramenta ten, entre os seus parámetros, o ancho e alto do arquivo de saída, pudendo dar como resultado máis dun arquivo de saída. O resultado de executar esta ferramenta serán dous arquivos:

    Un arquivo gráfico onde estean todos os gráficos xuntos.
    Un arquivo de texto cunha extensión **.atlas** (se pode editar) onde estean recollidas as coordenadas de cada un dos gráficos 'soltos' e o seu nome para poder referencialo (coincide có nome individual, o que tiña antes de xuntalo).

- Dende Libgdx, unha vez cargado o atlas, podemos referenciar cada un dos gráficos utilizando unha clase TextureRegion en vez da clase Textura utilizada ata o de agora.

1.1.2 Ferramenta para crear o Atlas

- No mercado existen varias ferramentas que fan o que necesitamos, como por exemplo:

    http://www.codeandweb.com/texturepacker/download
    http://renderhjs.net/shoebox/

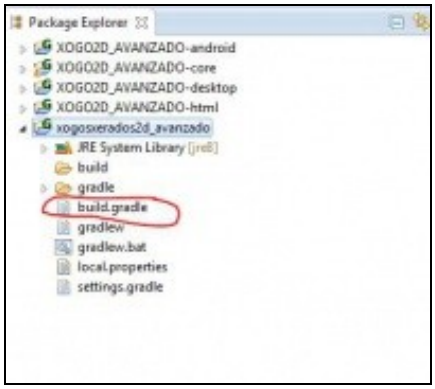
- TexturePacker: Ferramenta do propio Libgdx que ven dentro dun paquete de aplicacións.

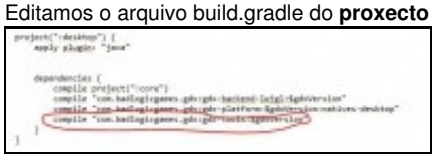
Neste manual vou explicar a ferramenta do Libgdx.

1.1.2.1 Dende terminal / consola

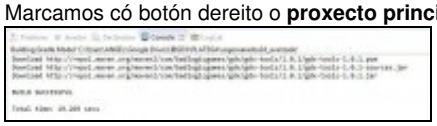
Necesitamos instalar un conxunto de ferramentas que veñen nun paquete de nome **GDX-TOOLS**.

O proceso para instalalo é moi simple grazas a Gradle e a súa forma de tratar as dependencias.

- Instalación da ferramenta TexturePacker
- 

  Editamos o arquivo build.gradle do **proxecto principal**.
- 

  Engadimos na **sección Desktop** esta liña: **compile "com.badlogicgames.gdx:gdx-tools:$gdxVersion"**
- 

  Marcamos có botón dereito o **proxecto principal** e escollemos as opcioóns de **Gradle** => **Refresh Dependencies**.
- 

  Unha vez feito se nos informa de que o jar coas ferramentas xa foi descargado.
- 

  Marcamos có botón dereito o **proxecto Desktop** e escollemos as opcioóns de **Gradle** => **Refresh Dependencies**.

Tamén podemos descargalo directamente cando xeramos o proxecto coa ferramenta de xeración de proxectos, marcando la opción GDX-TOOLS.

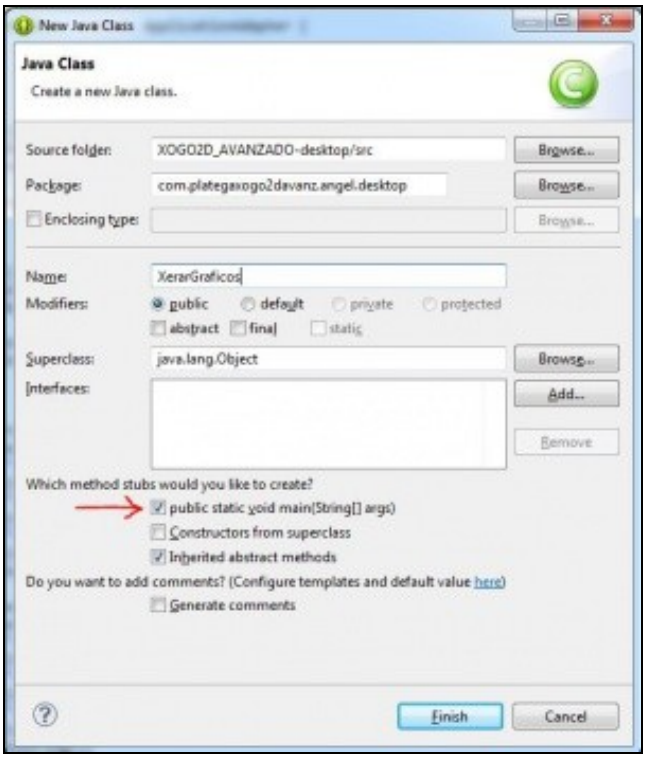
1.1.2.2 Graficamente

Podemos descargar a ferramenta dende este enlace: https://code.google.com/p/libgdx-texturepacker-gui/

1.1.3 Proceso para crear o Atlas

1.1.3.1 Dende consola / terminal

- Debemos crear unha clase de nome XerarGraficos có método main no proxecto Desktop.



- Escribiremos este código:

```
import com.badlogic.gdx.tools.texturepacker.TexturePacker;

public class XerarGraficos {

public static void main(String[] args) {
// TODO Auto-generated method stub

TexturePacker.process("C:\\temporal\\in", "C:\\temporal\\out", "packimaxes");
}

}
```

O método **process** leva tres parámetros. O cartafol onde están os gráficos a xuntar, o cartafol onde vai xerar o atlas, e o nome do atlas e arquivo gráfico xerado.

Importante: Necesitamos poñer a dobre barra xa que é un caracter de escape.

- Executamos a clase e teremos de saída o atlas.

Exemplo de saída do arquivo gráfico

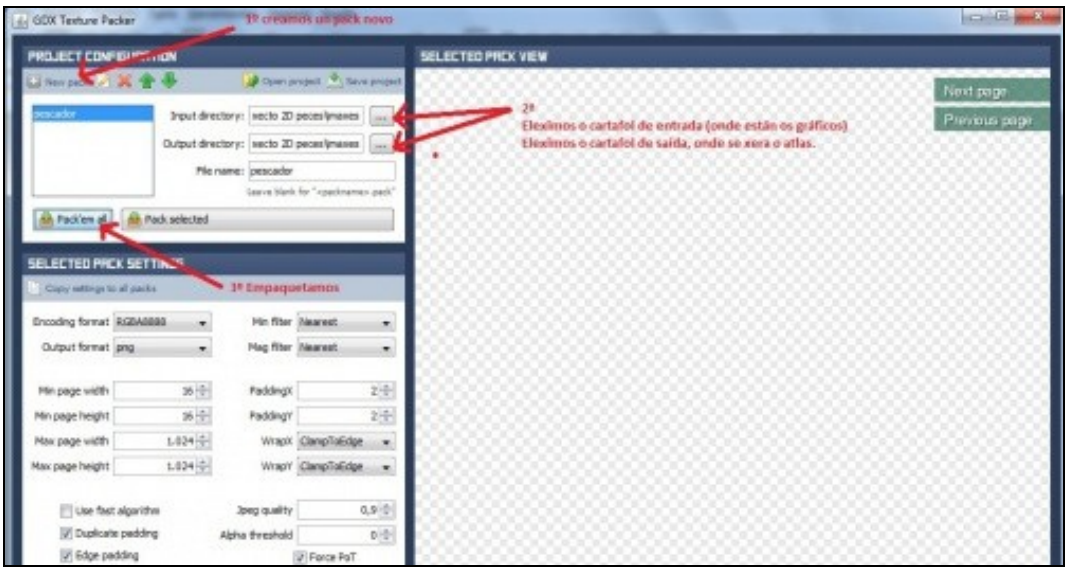
Exemplo de saída do arquivo .atlas

```
packimaxes.png
size: 435,126
format: RGBA8888
filter: Nearest,Nearest
repeat: none
LIBGDX_itin1_alien
  rotate: false
  xy: 56, 3
  size: 26, 22
  orig: 26, 22
  offset: 0, 0
  index: -1
LIBGDX_itin1_alien_dead
  rotate: false
  xy: 322, 49
  size: 26, 22
  orig: 26, 22
  offset: 0, 0
  index: -1
```

Fixarse como o único que ten este arquivo son o conxunta de nomes que tiñan os arquivos gráficos individuais e a súa posición e tamaño.

1.1.3.2 Graficamente

Executaremos a ferramenta previamente descargada.



Só debemos de seguir os pasos indicados:

- Crear un novo pack.
- Escoller o cartafol onde están os gráficos a empaquetar (input).
- Escoller o cartafol de saída (output).
- Premer o botón Pack´em all.

Dará como resultado o mesmo que a opción de consola explicada anteriormente.

1.1.4 Uso do Atlas

Temos que copiar os dous arquivos xerados ó cartafol **assets** da versión Android.

O proceso de carga do atlas será:

- Definir as clases que imos utilizar.

  Para o caso do atlas vai ser a clase TextureAtlas.
  Para o caso dos gráficos individuais vai ser a clase TextureRegion.

```
private static TextureAtlas texturas_todas;
public static TextureRegion textureAlien;
public static TextureRegion textureNave;
```

- Cargamos o atlas e as textureregion:

```
texturas_todas = new TextureAtlas("packimaxes.atlas");

textureAlien = texturas_todas.findRegion("LIBGDX_itin1_alien");
textureNave = texturas_todas.findRegion("LIBGDX_itin1_nave");
```

  Nota: Fixarse como para referenciar as rexións non necesitamos escribir a extensión dos arquivos, só o seu nome (no exemplo: LIBGDX_itin1_alien)

- Agora as debuxamos como sempre:

```
@Override
public void render () {
Gdx.gl.glClearColor(1, 0, 0, 1);
Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
batch.begin();
batch.draw(textureAlien, 0, 0);
batch.draw(textureNave, 100, 100);
batch.end();

}
```

- Liberamos o atlas, as textureregion non fai falla.

  Nota: Lembrar que se estades a desenvolver o xogo do manual, a liberación das texturas (neste caso do atlas) se fai no método liberarTexturas da clase AssetsXogo.

```
@Override
public void dispose(){
if (texturas_todas!=null)
texturas_todas.dispose();
}
```

1.2 Exemplo de código

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

Un exemplo completo baseado no xogo desenvolto.

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

Media:LIBGDX_textureAtlas.zip

- Crear unha nova clase.

Código da clase TextureAtlasCarga
Obxectivo: amosar como utilizar un Atlas

```
package com.plategaxogo2davanz.angel;

import com.badlogic.gdx.ApplicationAdapter;
import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.graphics.GL20;
import com.badlogic.gdx.graphics.g2d.SpriteBatch;
import com.badlogic.gdx.graphics.g2d.TextureAtlas;
import com.badlogic.gdx.graphics.g2d.TextureRegion;

public class TextureAtlasCarga extends ApplicationAdapter {
SpriteBatch batch;

private static TextureAtlas texturas_todas;
public static TextureRegion textureAlien;
public static TextureRegion textureNave;

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

texturas_todas = new TextureAtlas("packimaxes.atlas");

textureAlien = texturas_todas.findRegion("LIBGDX_itin1_alien");
textureNave = texturas_todas.findRegion("LIBGDX_itin1_nave");


}

@Override
public void render () {
Gdx.gl.glClearColor(1, 0, 0, 1);
Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
batch.begin();
batch.draw(textureAlien, 0, 0);
batch.draw(textureNave, 100, 100);
batch.end();

}

@Override
public void dispose(){
if (texturas_todas!=null)
texturas_todas.dispose();
}
}
```

1.3 TAREFA OPTATIVA A FACER

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

- Lembra borrar do cartafol assets todos os gráficos que estean no atlas xa que non os vas necesitar.
- Lembra modificar ó método dispose para liberar o atlas.
