LIBGDX Coñecendo a estrutura dos proxectos

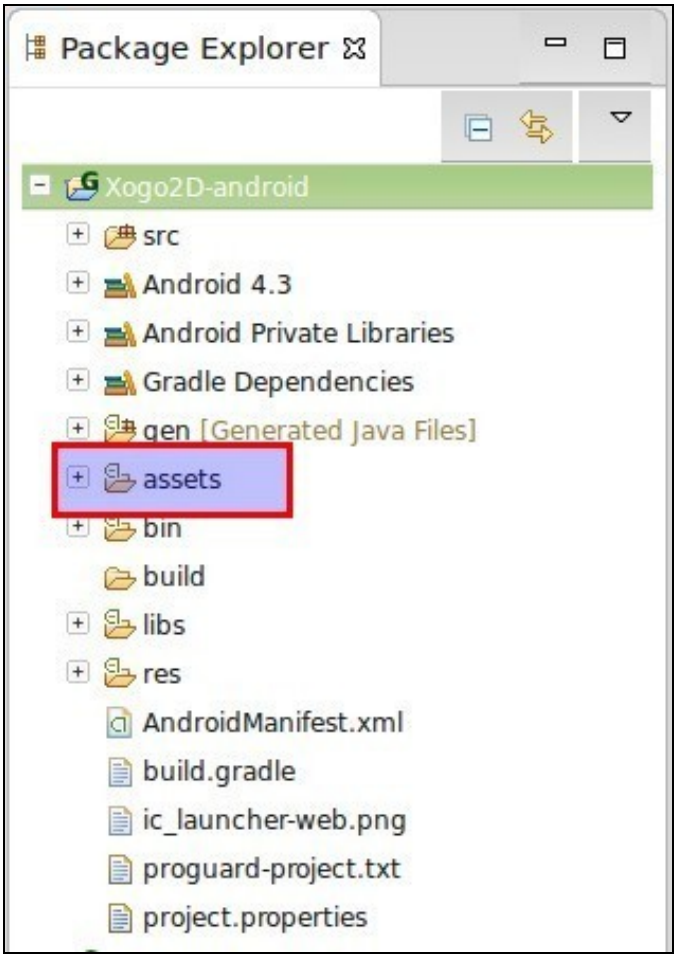
UNIDADE 2: Coñecendo a estrutura dos proxectos

Imos analizar a estrutura dos proxectos.

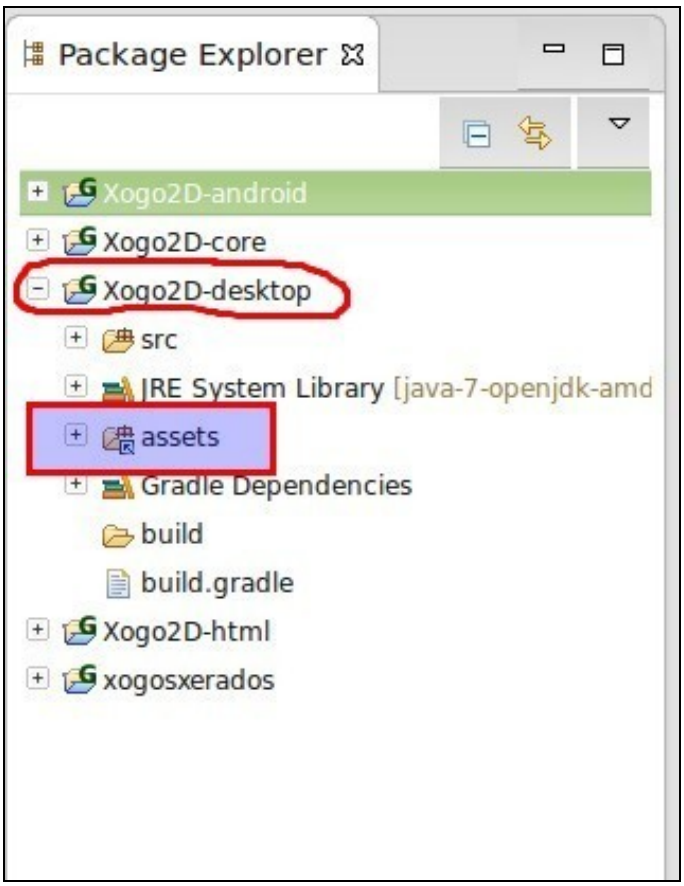
Sumario

Cartafol Assets

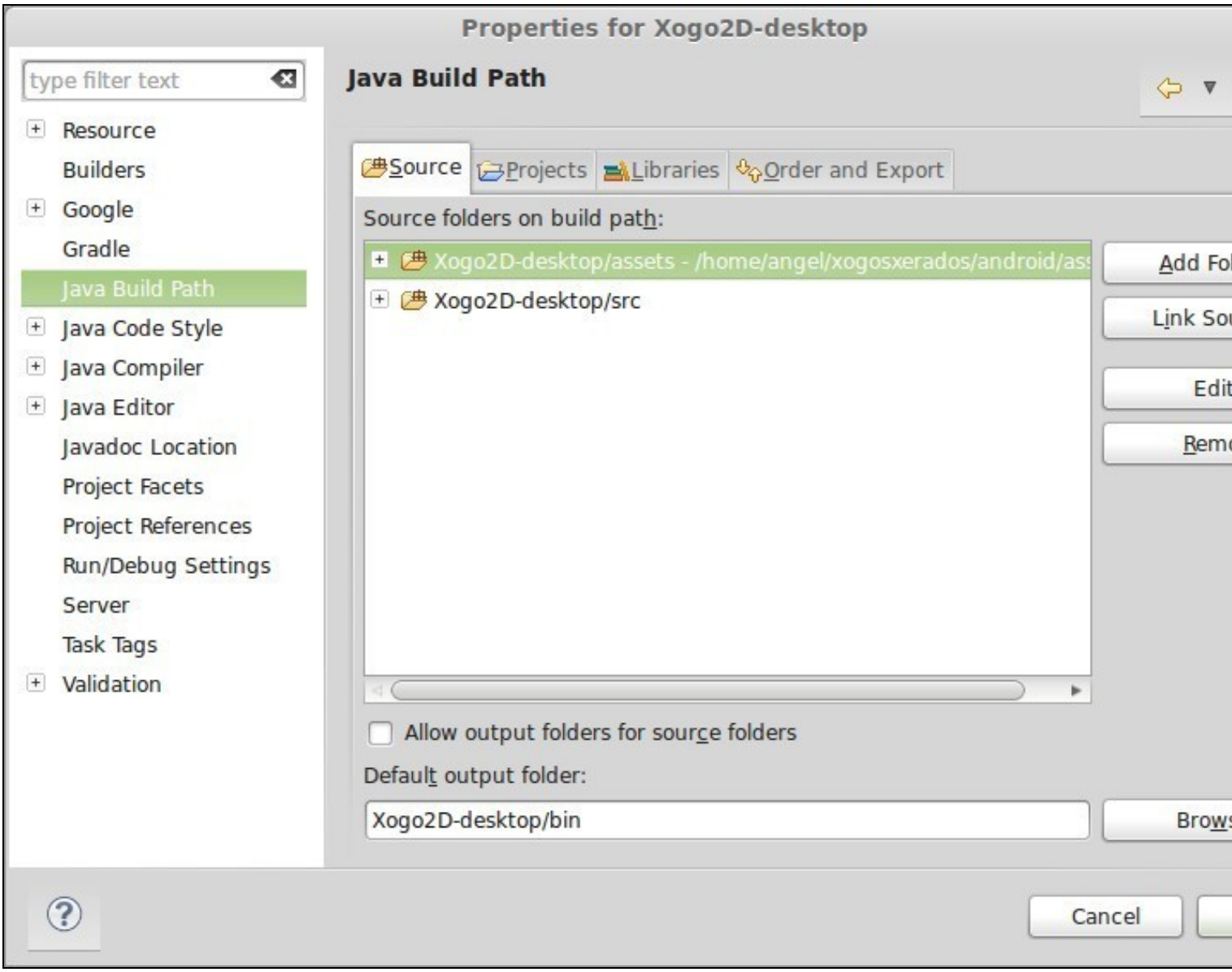
Na versión Android da nosa aplicación se atopa o cartafol Assets. É dentro deste cartafol onde temos que gardar todos os nosos recursos gráficos, música, efectos de son,....xa que a este cartafol é a onde vai dirixirse o resto de versións.



Podemos comprobar coma na versión Desktop o cartafol assets non é máis que un 'acceso directo' o cartafol creado na versión Android.



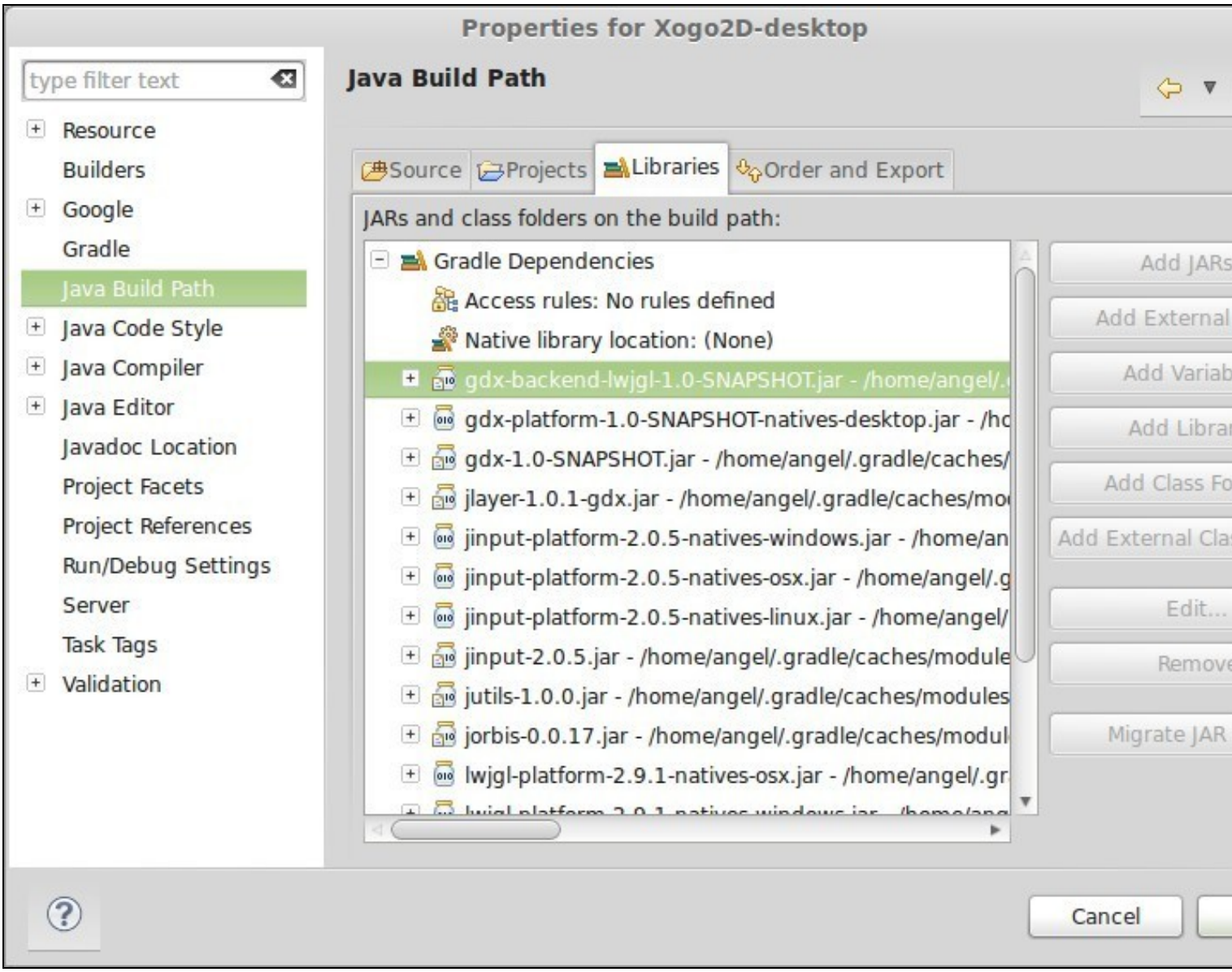
Se imos as propiedades do proxecto e marcamos a opción 'Java Build Path' podemos comprobar que dito cartafol é un 'Link Source' apuntando o cartafol Assets da versión Android:


Properties for Xogo2D-desktop
type filter text
Resource
Builders
Google
Gradle
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Project Facets
Project References
Run/Debug Settings
Server
Task Tags
Validation
Java Build Path
Source
Projects
Libraries
Order and Export
Source folders on build path:
Xogo2D-desktop/assets - /home/angel/xogosxerados/android/ass
Xogo2D-desktop/src
Add Fo
Link So
Edit
Remo
Allow output folders for source folders
Default output folder:
Xogo2D-desktop/bin
Brows
Cancel

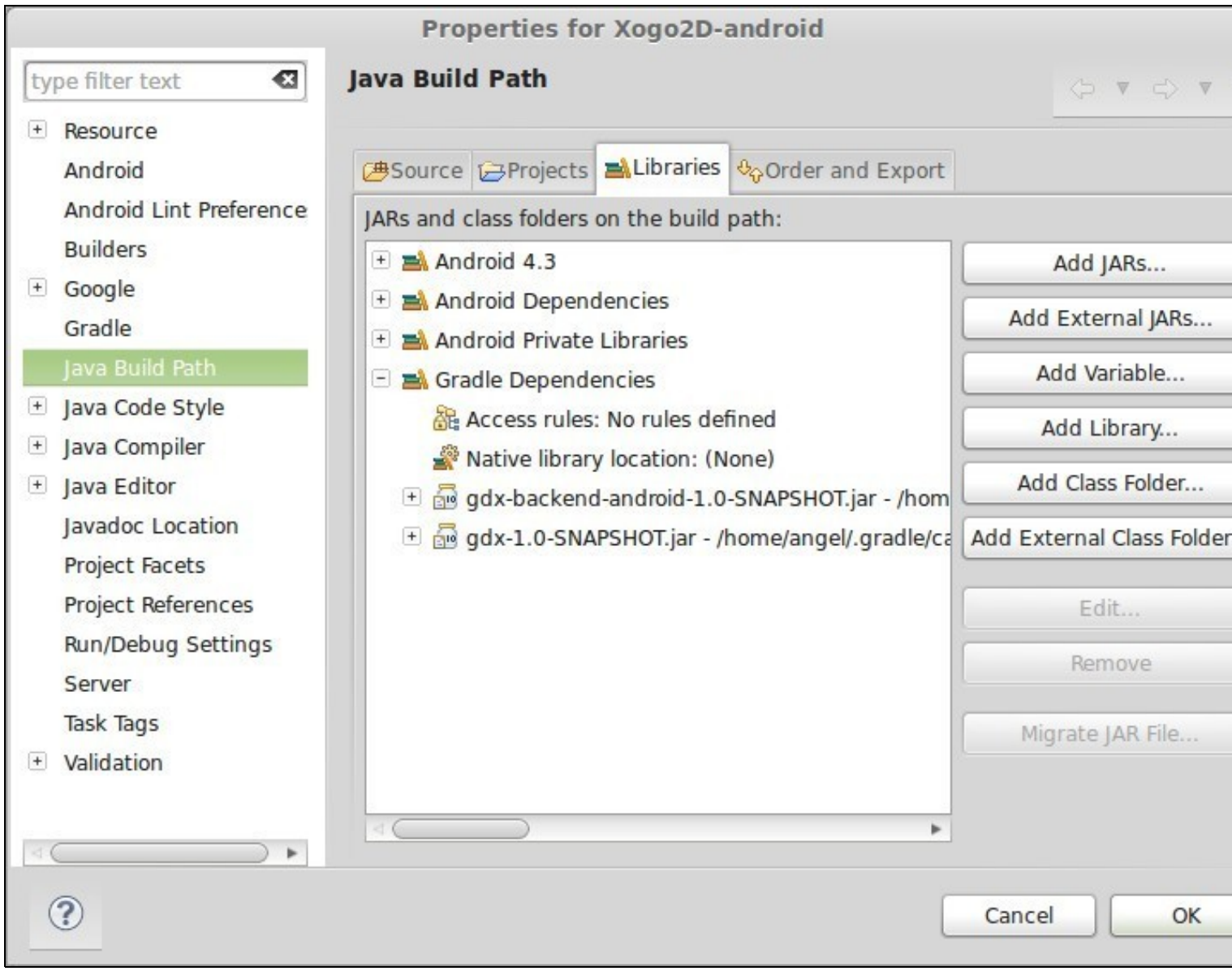
NOTA IMPORTANTE: cando facemos a importación de proxectos ó eclipse pode suceder que o enlace non funcione e sexa necesario volver a seleccionar ó cartafol da versión Android (premendo o botón **Edit** da pantalla anterior).

En Windows7 os arquivos '.project' e '.classpath' que están no cartafol raíz nas diferentes versións (desktop,html,android,...) aparecen como ocultos. É necesario quitarlle dito atributo.

Na mesma pantalla (se prememos na lapela **Libraries**) podemos ver que librerías do motor de xogos libgdx son necesarias para desenvolver os xogos:



O mesmo o podemos facer coa versión Android:



Clases que inician o xogo

Versión Desktop

No caso da versión Desktop é unha clase Java cun método **main**.

```
package com.plategaxogo2d.angel.desktop;

import com.badlogic.gdx.backends.lwjgl.LwjglApplication;
import com.badlogic.gdx.backends.lwjgl.LwjglApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class DesktopLauncher {
public static void main (String[] arg) {
LwjglApplicationConfiguration config = new LwjglApplicationConfiguration();
config.title = "O meu Xogo";
config.width = 480;
config.height = 320;
new LwjglApplication(new MeuXogo(), config);
}
}
```

Nota: No voso proxecto podedes ter un nome de paquete (package) diferente. Tede coidado cando copiedes o código e mantede o voso nome de paquete.

O que fai dito método é preparar un obxecto da clase **LwjglApplicationConfiguration** cos datos necesarios para executar o xogo, no noso caso indica que:

- O título do xogo será: O meu xogo (aparece na parte superior da xanela cando se executa o xogo)
- Indica o ancho e alto en pixeis da pantalla onde vai executarse a aplicación.

Despois crea un obxecto da clase **LwjglApplication** mandando coma parámetros a configuración indicada anteriormente e un obxecto da clase MeuXogo.

Imos analizar un pouco máis algunha das propiedades que podemos modificar na configuración Desktop.

- title: título que vai aparecer na parte superior da xanela cando se executa.
- width: ancho en pixeis da xanela.
- height: alto en pixeis da xanela.
- fullscreen: permite que o xogo se execute a pantalla completa. É necesario quitar as dúas propiedades anteriores ou ben poñer un height-width que permite ter a pantalla completa.

Para saír de pantalla completa podemos pechar a xanela premendo as teclas Alt+F4.

Utilizando esta forma usa a resolución máis baixa da tarxeta gráfica que permita ter a pantalla completa (normalmente 640x480 pixeis). Se queremos que estea a pantalla completa utilizando a resolución que estades a utilizar agora mesmo no voso computador, teríamos que obter o modo de resolución que ten a tarxeta gráfica neste intre.

O modo de resolución nos permite obter información sobre o ancho, alto, frecuencia e número de cores que está a utilizar a tarxeta.

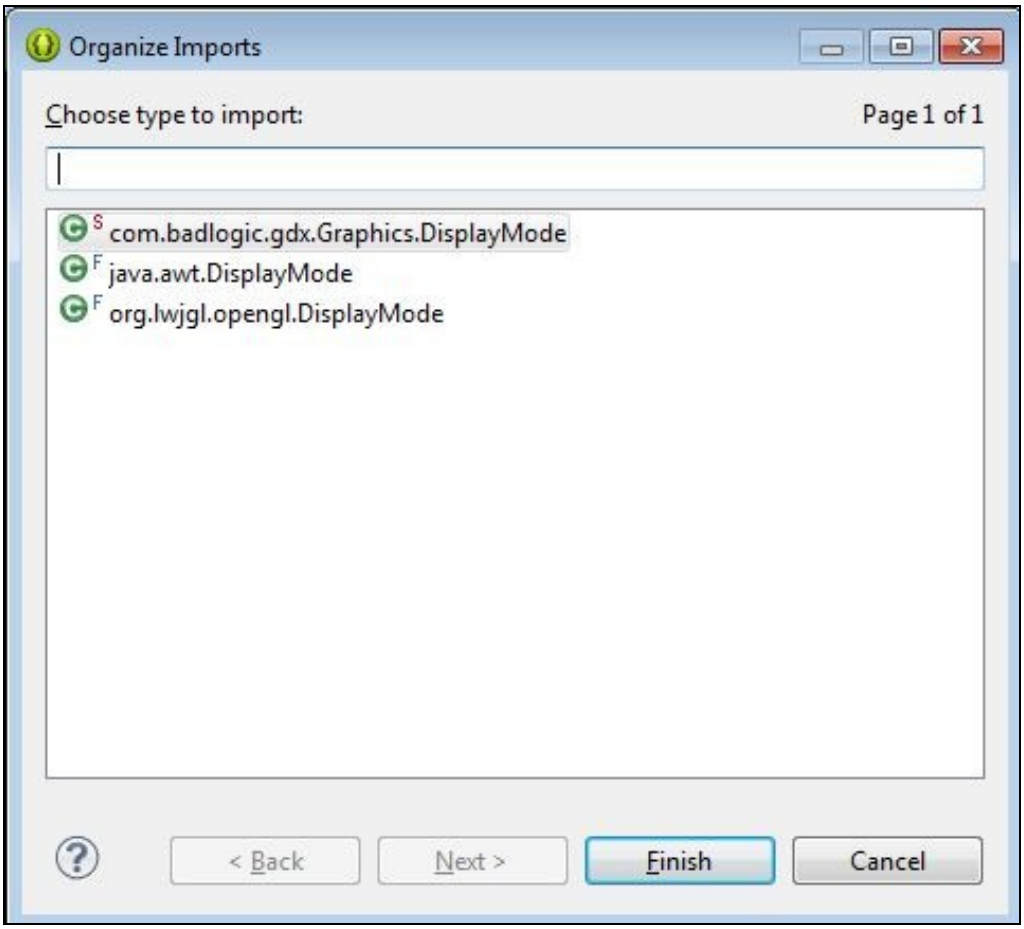
Para obtelo teremos que poñer este código:

```
package com.plategaxogo2d.angel.desktop;

import com.badlogic.gdx.Graphics.DisplayMode;
import com.badlogic.gdx.backends.lwjgl.LwjglApplication;
import com.badlogic.gdx.backends.lwjgl.LwjglApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class DesktopLauncher {
```

```
public static void main (String[] arg) {
LwjglApplicationConfiguration config = new LwjglApplicationConfiguration();
config.title = "O meu Xogo";
DisplayMode displaymode = LwjglApplicationConfiguration
.getDesktopDisplayMode();
config.width = displaymode.width;
config.height = displaymode.height;
config.fullscreen = true;
new LwjglApplication(new MeuXogo(), config);
}
}
```

- Engidida a liña 12 e modificadas as liñas 14 e 15.

Nota:: cando se engada unha nova clase ó código temos que importala. Unha forma de facelo 'automaticamente' en eclipse é premendo as combinacións de teclas **Control+Shift+O** (tecla O non o cero). Ó facelo aparecerá unha pantalla como a seguinte:



Debemos de escoller a clase que está relacionada co framework LIBGDX (neste caso a primeira) e premer o botón **Aceptar**. Desta forma aparece o import da clase na parte superior do código.

Grazas o obxecto da clase DisplayMode podemos obter a resolución que agora mesmo estades a utilizar no voso computador. Con cfg.getDisplayModes() poderíamos obter todas as resolucións que acepta a vosa tarxeta gráfica.

Fixarse como podemos enviarlle a resolución máxima e non utilizar a propiedade fullscreen, é dicir, poñer este código:

```
package com.plategaxogo2d.angel.desktop;

import com.badlogic.gdx.Graphics.DisplayMode;
import com.badlogic.gdx.backends.lwjgl.LwjglApplication;
import com.badlogic.gdx.backends.lwjgl.LwjglApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class DesktopLauncher {
public static void main (String[] arg) {
LwjglApplicationConfiguration config = new LwjglApplicationConfiguration();
config.title = "O meu Xogo";
DisplayMode displaymode = LwjglApplicationConfiguration
.getDesktopDisplayMode();
config.width = displaymode.width;
config.height = displaymode.height;

new LwjglApplication(new MeuXogo(), config);
}
}
```

Como podemos ver o xogo ocupa toda a pantalla pero mantemos a barra superior co nome do xogo.

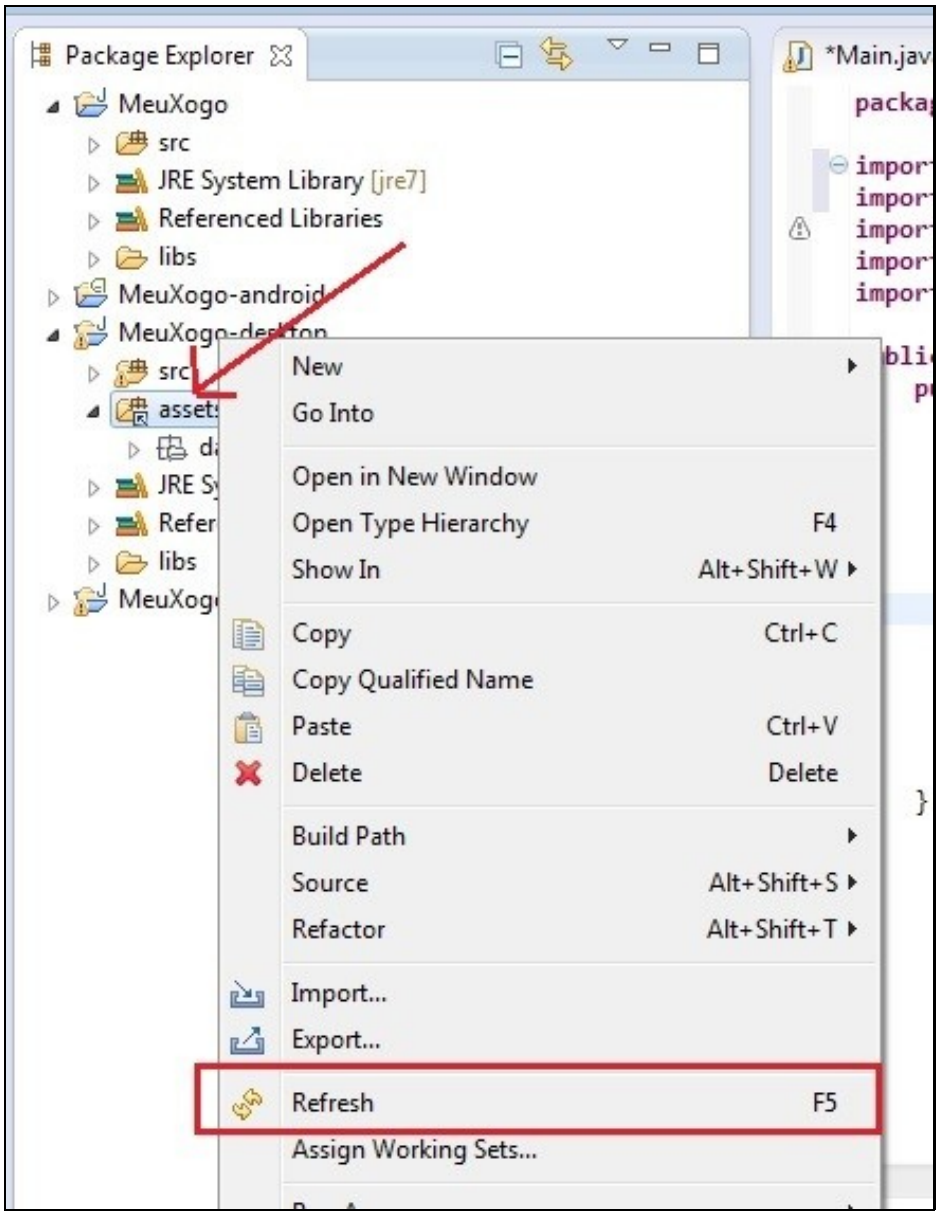
Seguimos coas propiedades e métodos:

- resizable: boolean que indica que queremos que a pantalla do xogo se poda redimensionar.
- x | y: posición x e y da pantalla do xogo (-1 se queremos que estea centrada, valor por defecto).
- addIcon(path,tipo): permite engadir unha icona ó xogo que será o que apareza na barra inferior cando minimizamos o xogo.

O proceso é moi sinxelo. Debemos descargar o gráfico que vai ser a nosa icona e levalo ó cartafol **assets** da versión móbil (vale levalo tamén ás outras versións, xa que son accesos directos á versión móbil). Normalmente será gráfico de tipo *bmp* e tamaño *32x32 píxeles*.

IMPORTANTE: Unha vez gardado debemos ir o cartafol assets dende Eclipse e premer a tecla F5 para refrescar ou ben, sobre o cartafol, premer o

botón dereito e escoller a mesma opción.



Ó facelo aparecerá o gráfico gardado.

Agora debemos de escribir o seguinte código (lembrade premer a combinación de teclas Control+Shift+O para importar as clases novas):

```
package com.plategaxogo2d.angel.desktop;

import com.badlogic.gdx.Files.FileType;
import com.badlogic.gdx.backends.lwjgl.LwjglApplication;
import com.badlogic.gdx.backends.lwjgl.LwjglApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class DesktopLauncher {
public static void main (String[] arg) {
LwjglApplicationConfiguration config = new LwjglApplicationConfiguration();
config.title = "O meu Xogo";
config.width = 800;
config.height = 480;

config.resizable=false;
config.addIcon("xogo.png",FileType.Internal);

new LwjglApplication(new MeuXogo(), config);
}
}
```

Agora podedes comprobar como aparece a icona:

- Icona do xogo
- 

Icona na parte superior esquerda.
- 
Icona na barra de tarefas.

TAREFA 2.1 APARTADO A) A FACER: Esta parte está asociada á realización dunha tarefa.

Seguimos coas propiedades e métodos:

- overrideDensity (a partires da versión 1.1.1): permite simular a densidade da pantalla dun dispositivo móbil.

Podemos consultar neste enlace as diferentes densidades de pantalla. Teredes que buscar a **Tabla 1** que ten de título: Table 1. Configuration qualifiers that allow you to provide special resources for different screen configurations. Así temos, por exemplo, que a densidade hdpi equivale a 240 puntos por

pulgada.

Esta propiedade vai vir moi ben para probar os xogos como se fosen dispositivos móbiles. Así, se quero emular un dispositivo móbil de 3,7 *de 800x480 pixeles cunha densidade de 252 dpi (dot per inch:puntos por pulgada) poñería:*

```
config.width=800;
config.height=480;
config.overrideDensity=240;
```

Podemos ver un exemplo de uso:

Exemplo obtido de https://github.com/libgdx/libgdx/pull/1825

Versión Móbil

No caso da versión Móbil usa a clase AndroidLauncher cun método **onCreate**.

```
package com.plategaxogo2d.angel.android;

import android.os.Bundle;

import com.badlogic.gdx.backends.android.AndroidApplication;
import com.badlogic.gdx.backends.android.AndroidApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class AndroidLauncher extends AndroidApplication {
@Override
protected void onCreate (Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
AndroidApplicationConfiguration config = new AndroidApplicationConfiguration();
initialize(new MeuXogo(), config);
}
}
```

Nota: No voso proxecto podedes ter un nome de paquete (package) diferente. Tede coidado cando copiedes o código e mantede o voso nome de paquete.

O que fai dito método é preparar un obxecto da clase **AndroidApplicationConfiguration** no que podemos indicar que hardware imos usar no noso dispositivo.

O código que lanza a aplicación é diferente da versión Desktop xa que o que estamos a desenvolver é unha aplicación Android (a clase deriva de AndroidApplication e se inicializa no método onCreate) pero os conceptos son os mesmos: creamos un obxecto coa configuración a utilizar e lanzamos un obxecto da clase MeuXogo con dita configuración.

Algúns dos parámetros que podemos modificar neste arquivo son:

- useAccelerometer: booleano que indica se se vai facer uso do acelerómetro do dispositivo móbil.
- useCompass: booleano que indica se se vai facer uso do compás.
- useWakelock: necesario para que o S.O. Android non pase ó estado de 'durmido' cando non recibe ningunha sinal por parte do xogo.

A propiedade wakelock debe estar sempre a true e as outras dúas dependerá se imos a usar ou non esas funcionalidades no noso xogo.

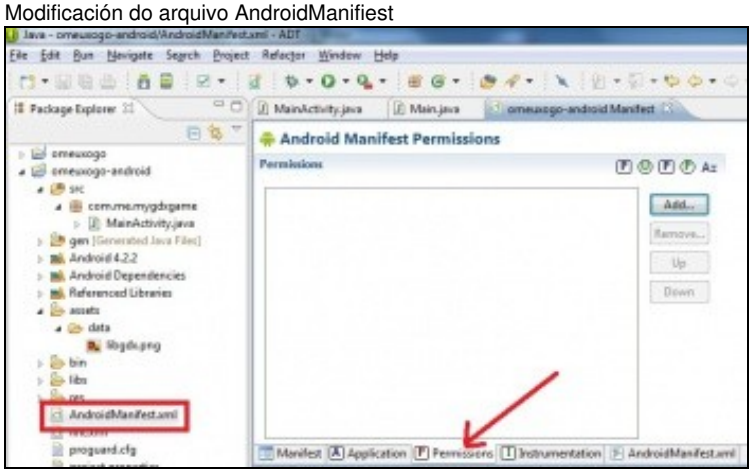
O código quedaría así:

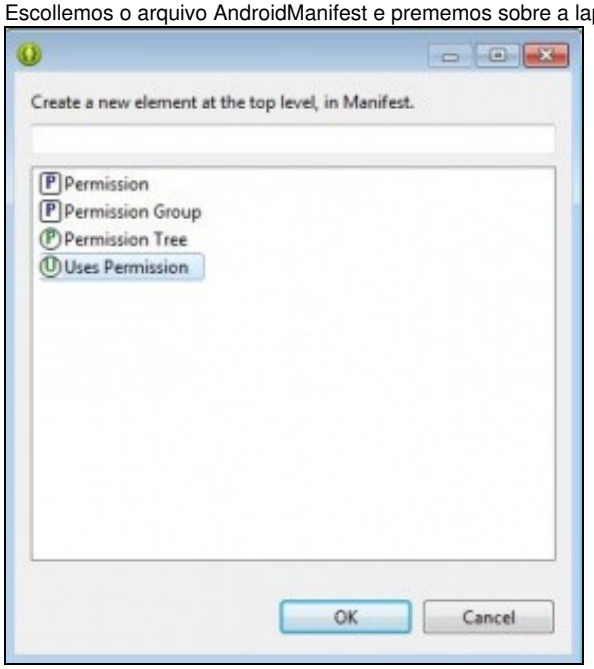
```
package com.plategaxogo2d.angel.android;

import android.os.Bundle;

import com.badlogic.gdx.backends.android.AndroidApplication;
import com.badlogic.gdx.backends.android.AndroidApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class AndroidLauncher extends AndroidApplication {
@Override
protected void onCreate (Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
AndroidApplicationConfiguration config = new AndroidApplicationConfiguration();

config.useAccelerometer=false;
config.useCompass=false;
config.useWakelock=true;

    initialize(new MeuXogo(), config);
}
}
```

Nota: Engadidas liñas 15-17.

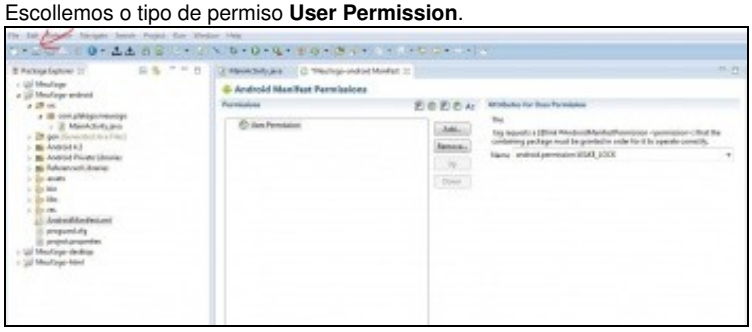
Facer que o dispositivo Android non entre en estado 'durmido' leva consigo solicitar ó sistema operativo permiso para facer iso. Cando instalamos unha aplicación móbil, antes de facer dita instalación, se nos amosa a lista de 'permisos' que solicita dita aplicación para poder ser executada.

Para indicarlle ó S.O. que a aplicación vai usar dito permiso será necesario modificar o arquivo **AndroidManifiest.xml** que se atopa no raíz do cartafol Android.

- Modificación do arquivo AndroidManifiest
- 

Escollemos o arquivo AndroidManifest e prememos sobre a lapela **Permissions**.
- 

Escollemos o tipo de permiso **User Permission**.

- 

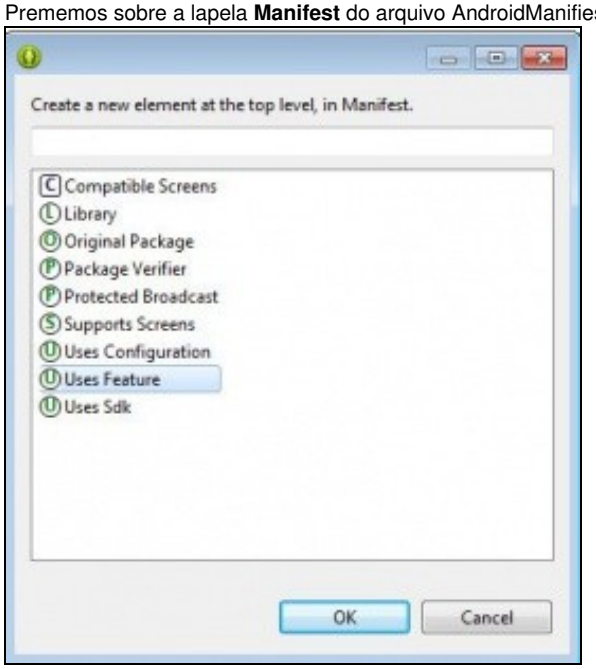
Buscamos o permiso WAKE-LOCK e prememos o botón **Gardar** de Eclipse.

No caso das outras dúas opcións (acelerómetro e compás) non é necesario engadir un permiso para facer uso do hardware do dispositivo, pero podemos indicar ó usuario que o xogo vai facer uso deles e incluso, se poñemos o xogo no Market, podemos facer que non poida ser instalado se non se ten dito hardware.

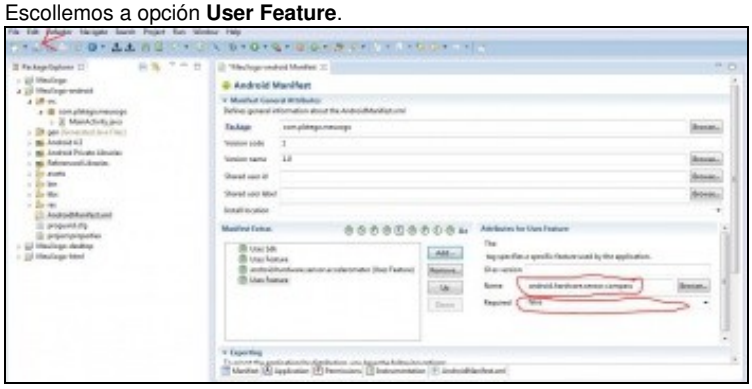
- Modificación do arquivo AndroidManifiest

- 

Prememos sobre a lapela **Manifest** do arquivo AndroidManifiest.xml e despois prememos o botón **Add**.

- 

Escollemos a opción **User Feature**.

- 

  Escribimos en Name o hardware a usar **android.hardware.sensor.accelerometer** e indicamos se é necesario o seu uso para o funcionamento do xogo. Repetimos a operación có outro hardware **android.hardware.sensor.compass**.

- Se escollemos a opción 'Required' indicamos que é necesario dito hardware para que o xogo funcione.

AVISO: Os alumnos que dispoñan de dispositivo móbil que engadan o uso do acelerómetro no AndroidManifiest.xml e modifiquen o código da clase **AndroidLauncher.java** indicando que se vai facer uso de dito hardware.

Se botamos unha ollada máis detallada o arquivo de AndroidManifiest.xml podemos observar que na liña 19 a ferramenta de xeración de proxectos indica que o xogo vai visualizarse en 'apaisado'. Se queremos que sexa en vertical teremos que sustituír landscape por **portrait**.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.plategaxogo2d.angel.android"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="19" />
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-feature android:required="true" android:name="android.hardware.sensor.accelerometer"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/GdxTheme" >
        <activity
            android:name="com.plategaxogo2d.angel.android.AndroidLauncher"
            android:label="@string/app_name"
            android:screenOrientation="landscape"
            android:configChanges="keyboard|keyboardHidden|orientation|screenSize">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

No mesmo arquivo (androidmanifiest.xml) podemos ver como se fai referencia a icona que vai ter o xogo así como o nome que vai ter o mesmo cando se instale nun dispositivo móbil.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.plategaxogo2d.angel.android"
    android:versionCode="1"
    android:versionName="1.0" >
```

```
    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="19" />
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-feature android:required="true" android:name="android.hardware.sensor.accelerometer"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/GdxTheme" >
        <activity
            android:name="com.plategaxogo2d.angel.android.AndroidLauncher"
            android:label="@string/app_name"
            android:screenOrientation="landscape"
            android:configChanges="keyboard|keyboardHidden|orientation|screenSize">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

- android:icon="@drawable/ic_launcher": atópase no cartafol /res/drawable-xxxx. En función da resolución do dispositivo collerá o da carpeta correspondente.
- android:label="@string/app_name": fai referencia a un dato gardado no arquivo /res/values/strings.xml. Se premedes sobreo arquivo deberedes cambiar o value para darlle un novo nome ó xogo.

TAREFA 2.1 APARTADO B) A FACER: Esta parte está asociada á realización dunha tarefa.

Versión HTML

A clase que se inicio se atopa no *paquete client* do voso proxecto e o nome da clase é HTMLLauncher.

Podemos examinar o código:

```
package com.plategaxogo2d.angel.client;

import com.badlogic.gdx.ApplicationListener;
import com.badlogic.gdx.backends.gwt.GwtApplication;
import com.badlogic.gdx.backends.gwt.GwtApplicationConfiguration;
import com.plategaxogo2d.angel.MeuXogo;

public class HtmlLauncher extends GwtApplication {

        @Override
        public GwtApplicationConfiguration getConfig () {
                return new GwtApplicationConfiguration(480, 320);
        }

        @Override
        public ApplicationListener getApplicationListener () {
                return new MeuXogo();
        }
}
```

Podedes observar como na liña 17 se pasa o control á clase común MeuXogo...

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