Escola Universitària
Politècnica de Mataró

Centre adscrit a:

UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Enginyeria Tècnica de Telecomunicacions: Especialitat Telemàtica

ENTORN D'AUTOAPRENENTATGE D'ANDROID: APLICACIÓ NEWS TODAY

Memòria

EVA ROBLES GARCIA
PONENT: ANTONI SATUÉ VILLAR

PRIMAVERA 2012

TecnoCampus
Mataró-Maresme

Dedicatòria

Dedicat a totes aquelles persones que han cregut en mi sempre.

Agraïments

Gràcies al Rubén i Víctor per la seva paciència i els seus consells.

A l'Antoni Satué, per donar-me l'oportunitat de realitzar aquest projecte.

Resum

Aquest projecte té com a objectiu introduir-se en el món de la programació Android. Per fer-ho s'ha creat l'aplicació *"News Today"* en la qual es mostren diverses accions que es poden dur a terme amb aquest sistema operatiu. A partir de la explicació teòrica s'ha treballat concretament amb un lector de notícies dividit per diverses categories, el sistema de notificacions i el posicionament mitjançant GPS el qual es capaç de trobar la localització de l'usuari i mostrar-li els cinemes més propers.

Resumen

Este proyecto tiene como objetivo introducirse en el mundo de la programación Android. Para ello se ha creado la aplicación "*News Today*" en la cual se muestran diversas acciones que se pueden llevar a cabo con este sistema operativo. A partir de la explicación teórica se ha trabajado concretamente con un lector de noticias dividido en diferentes categorías, el sistema de notificaciones y el posicionamiento mediante GPS el cual es capaz de encontrar la localización del usuario y mostrarle los cines más cercanos.

Abstract

This project aims to introduce in the world of Android programming. For this reason, it has been created the application "News Today" in which it shows different actions that it can be carried out with this operating system. From the theoretical explanation it has been worked specifically with a news reader divided in different categories, the notification system and GPS which it's able to find the user's location and it shows the cinemas more nearby.

Índex

Índex de figures

Índex de taules

Glossari de termes

ADT	Android Development Tools
API	Application Programming Interface
AVD	Android Virtual Device
GPS	Global Positioning System
IDE	Integrated Development Environment
JDK	Java Development Kit
JRE	Java Runtime Environment
Kernel	Nucli del sistema operatiu
ME	Mobile Edition
RSS	Really Simple Syndication
SDK	Software Development Kit
SE	Standard Edition
SO	Sistema Operatiu
SQL	Structured Query Language
SSL	Secure Socket Layer

1. Objectius

1.1. Propòsit

Conèixer i aprendre com funciona el món de la programació Android, així com el llenguatge que s'utilitza (Java i XML) i les parts que el formen. A més a més, té com a propòsit el coneixement de diferents tecnologies sobre aquests dispositius com és la connexió WIFI o el posicionament mitjançant GPS.

1.2. Finalitat

Saber dissenyar i implementar una aplicació amb aquest sistema operatiu que mostri algunes de les diferents funcionalitats, de forma que l'usuari final pugui dur a terme un projecte igual o molt similar a partir d'uns coneixements adquirits prèviament.

1.3. Objecte

L'Aplicació "*News Today*" està llesta per a ser instal·lada en un dispositiu mòbil Android i és capaç d'oferir un llistat de notícies dividides en categories, així com un sistema de notificacions i geolocalització mitjançant l'explicació de la teoria, manual de instal·lació i configuració del software i confecció i explicació de l'aplicació.

1.4. Abast

Per a aquesta aplicació en Android es necessari l'ús d'Internet i GPS. D'altre banda, és *open source* amb la qual cosa un altre usuari pot tenir accés al seu codi per a possibles futures implementacions o simplement per aprendre aquest llenguatge mitjançant aquest exemple.

2. Sistema operatiu Android: un nou món

2. 1. Conceptes generals i característiques

Android és un sistema operatiu basat en codi obert i creat per Google i *Open Handset Alliance*. Utilitzat per a telèfons i dispositius mòbils fa que s'hagi convertit en una de les plataformes principals per als desenvolupadors d'aplicacions.

Aquest sistema operatiu és el primer entorn en combinar les diferents característiques que es detallen a continuació:

- **Plataforma** de desenvolupament completament **oberta** i **gratuïta** basada en Linux i codi obert.

- **Milers de serveis integrats:** GPS, base de dades SQL, vistes de navegació i mapes integrats directament a algunes aplicacions...

- **Gestió automàtica del cicle de vida**, és a dir, els programes es troben aïllats uns dels altres, mitjançant capes de seguretat. L'usuari no ha de preocupar-se en cap moment de si les aplicacions es troben obertes o de tancar certs programes per a poder obrir-ne de nous. Android està optimitzat per a dispositius amb poca memòria i poca bateria.

2.2. Arquitectura del sistema

2.2.1.Capes

2.2.1.1. Kernel de Linux

El nucli d'Android està construït mitjançant el *kernel* de Linux. El sistema l'utilitza per a la gestió de la memòria, dels processos, operacions de xarxa i altres serveis relacionats amb el sistema operatiu com són els elements de comunicació (*networking*).

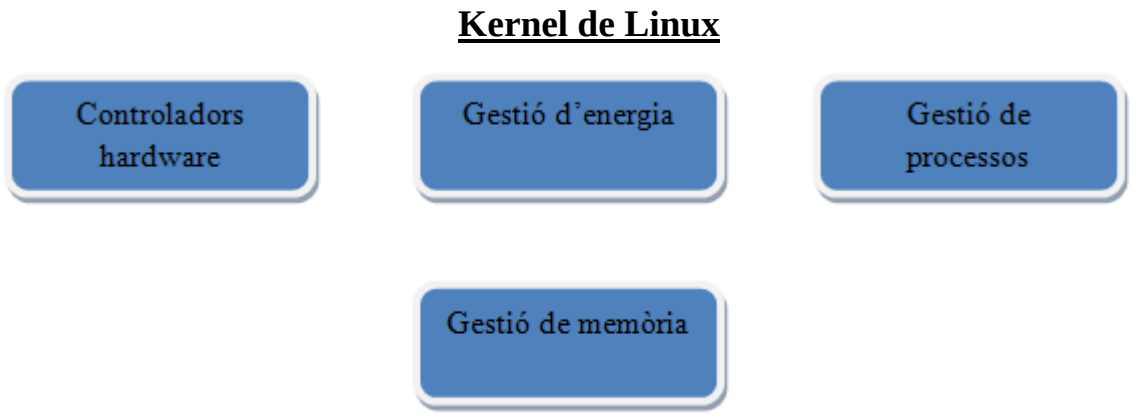

Fig. 2.1. Kernel Linux

Els components que el formen estan agrupats per capes. Cadascuna d'aquestes capes utilitza elements de la capa inferior per a dur a terme les diferents funcions, és per aquest motiu que el tipus d'arquitectura és anomenat pila.

2.2.1.2. Biblioteques

La capa que es troba just per sobre del nucli està composada per el que se'n diu biblioteques natives (Fig. 2.2). Es troben escrites en el llenguatge C o C++ i compilades per la arquitectura hardware especifica del telèfon o dispositiu. El principal objectiu de la seva existència es proporcionar funcionalitat a les aplicacions, és a dir, evitar codificar cada cop les tasques que es van repetint freqüentment.

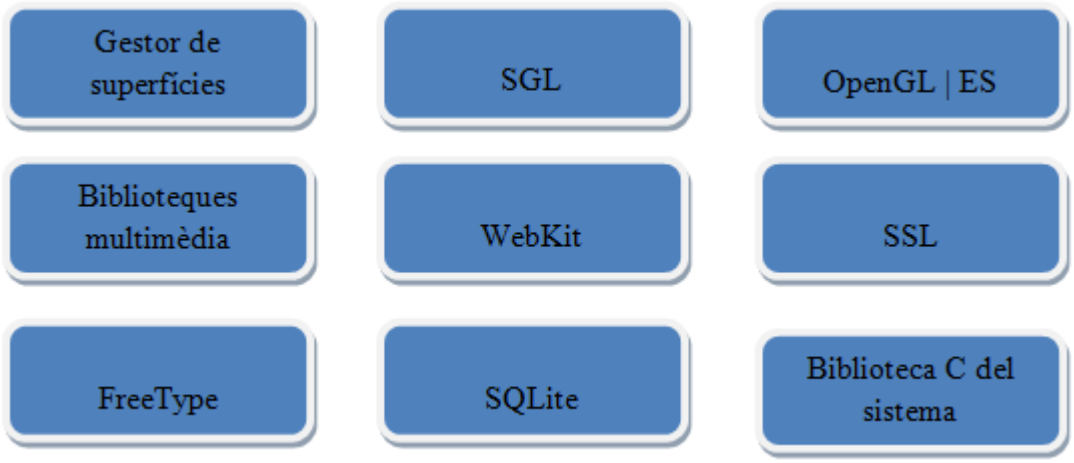

Fig. 2.2. Biblioteques natives

- **Gestor de superfície (*Surface Manager*):** Gestor de finestres de composició similar al Vista. Composat per mapes de bits emmagatzemats que combinats amb d'altres formen la pantalla que l'usuari final visualitzarà, creant efectes com poden ser les finestres transparents.

- **Gràfics 2D i 3D:** Elements de diferents dimensions combinats en una única interfície d'usuari.

- **Còdecs multimèdia:** Android reprodueix vídeo i àudio en diferents formats (MPEG-4,AAC,MP3..).

- **Bases de dades SQL:** Inclou el motor de la base de dades *SQLite*.

- **Tecnologia de navegador:** Utilitza *WebKit*, motor web utilitzat pel navegador tant independentment com en les aplicacions.

- **SSL(*Secure Sockets Layer*):** Proporciona seguretat al accedir a Internet mitjançant la criptografia.

2.2.1.3. Entorn d'execució

Dalvik (Fig. 2.3.) és una màquina virtual Java implementada per Google i optimitzada per a dispositius mòbils.

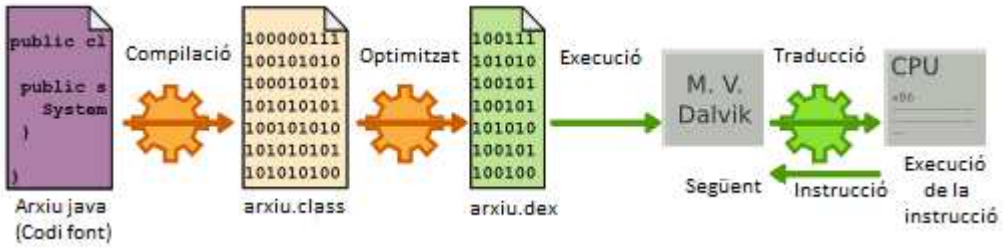

Fig. 2.3. Màquina virtual Dalvik

Justament per sobre del kernel es troba l'entorn d'execució (Fig. 2.4.) format per *Dalvik* i les biblioteques *core* de Java.

Fig. 2.4. Entorn d'execució

2.2.1.4. Estructura de les aplicacions

La capa que es troba per sobre de les biblioteques i l'entorn d'execució pertany a l'estructura de les aplicacions (Gestor d'activitats, gestor de paquets, gestor de telefonia...) (Fig. 2.5.).

En aquest apartat però s'expliquen només aquelles més importants:

- **Gestor de recursos:** Per recursos s'entén qualsevol element que no formi part del codi però si de l'aplicació o programa en si.
- **Gestor de notificacions:** Esdeveniments tal com les alertes de proximitat, missatges rebuts, trucades rebudes...
- **Gestor d'activitat:** S'encarrega de controlar el cicle de vida de cadascuna de les aplicacions mantenint un ordre establert per tal que l'usuari pugui navegar fàcilment entre elles.
- **Gestor d'ubicació:** Localització d'on es troba la persona en aquell moment o localització d'un determinat lloc.

- **Proveïdors de contingut:** Dades que necessiten les diferents aplicacions per a ser compartides.

Fig. 2.5. Estructura de les aplicacions

2.2.1.5. Aplicacions

A diferència de les altres capes, aquesta capa és la única que veu l'usuari final. Per una banda tenim les aplicacions i per una altra els *widgets* (Fig. 2.6). Les aplicacions són aquells programes que poden ocupar tota la pantalla e interactuar amb l'usuari, en canvi, els *widgets*, tan sols ocupen una petita porció de la pantalla d'inici.

Fig. 2.6. Aplicacions i widgets

2.3. Funcionament i familiarització amb l'entorn

Android es basa en una aplicació en primer pla que sol ocupar tota la pantalla a excepció de la barra d'estat. A l'encendre el dispositiu mòbil, el primer que es mostra és la pantalla d'inici.

Seguidament a l'executar una determinada aplicació, el sistema operatiu la inicia i passa a estar en primer pla. Aquests programes o aplicacions són els que es guardaran al gestor d'activitats explicat en el punt anterior.

Cada pantalla de la interfície d'usuari es troba representada per la classe *Activity*. Cadascuna d'aquestes activitats conté el seu propi cicle de vida.

A la fig. 2.7, es mostra una activitat Android en els seus cinc estats possibles: Iniciada, en execució, en pausa, parada o destruïda.

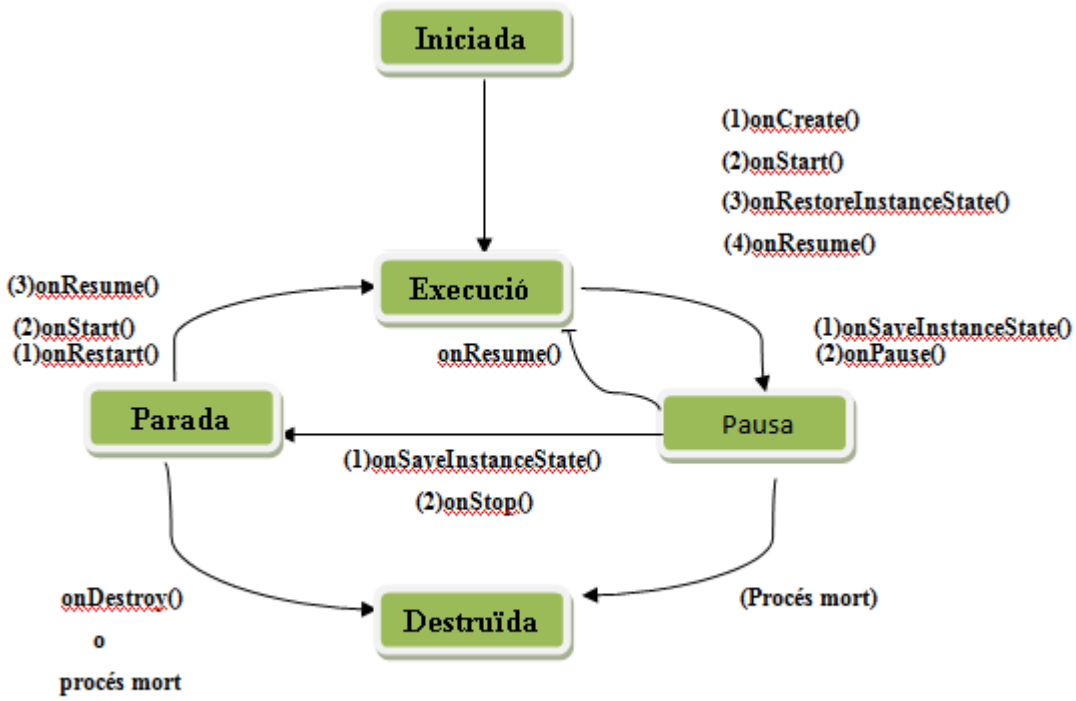

Fig. 2.7. Cicle de vida d'una activitat Android

A continuació s'expliquen alguns dels mètodes que es poden utilitzar a la classe *Activity*:

- **onCreate(Bundle):** Aquest mètode es cridat quan s'inicia l'activitat per primer cop.

- **onPause():** Mètode utilitzat quan es vol que l'aplicació passi a un segon pla, degut a que hi ha una altra de més importància en aquell moment que vol ser executada.

- **onResume():** Mètode utilitzat quan l'usuari interactua amb l'aplicació.

- **onStop():** Mètode utilitzat quan l'activitat ja no és necessària.

A part de les activitats hi ha tres tipus d'objectes més: Intencions, serveis i proveïdors de continguts.

- Una **intenció** s'anomena al mecanisme que descriu una acció especifica sobre el dispositiu com pot ser, trucar, enviar un missatge, fer una foto...

- Un **servei** és una tasca executada en segon pla en la qual l'usuari no interacciona amb ella en cap moment.

- Per últim, els **proveïdors de continguts** són un conjunt de dades que es poden trobar en una API i als quals es pot accedir i llegir.

3. Instal·lació de les eines necessàries per a la creació de l'aplicació

3.1. Màquina virtual Java SE

Java SE permet que es puguin executar programes amb el llenguatge Java. Android és un sistema operatiu on moltes aplicacions o programes utilitzen aquest entorn de programació, per tant aquest serà necessari, tot i que en aquest cas no serà suficient amb el JRE de la màquina virtual sinó que caldrà el *kit* de desenvolupament. Cal seguir els passos que s'especifiquen a continuació per tal de descarregar i instal·lar correctament el JDK.

1. Descarregar la última versió de la pàgina web : http://www.oracle.com/technetwork/java/javase/downloads/jdk-7u3-download-1501626.html .

D'acord amb el sistema operatiu pertinent en aquest cas Windows 7 (64 bits) seleccionem: **jdk-7u3-windows-x64.exe**

2. Escollir les eines que es volen instal·lar: *Development Tools*, *Source Code* i *Public JRE* →Clic "*Next*" (Fig. 3.1).

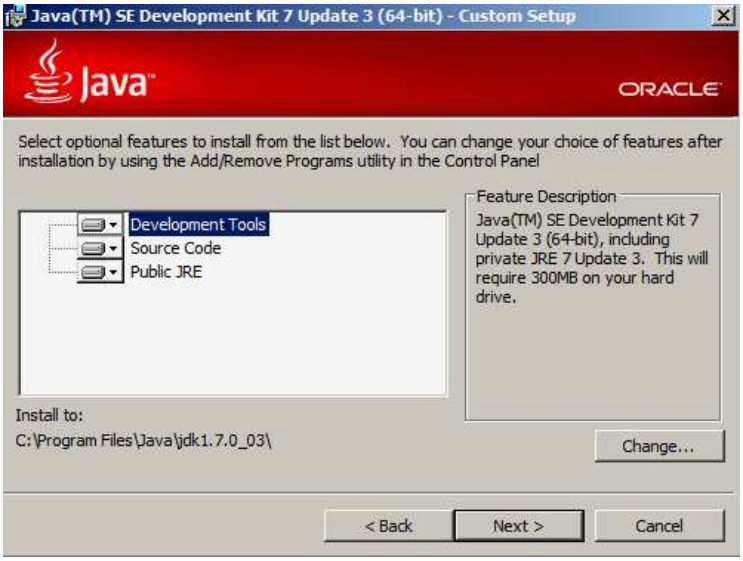

Fig. 3.1. Eines necessàries per a la instal·lació

3. S'escull el directori on es guardarà el JRE. En aquest cas es deixa la ruta per defecte, després fer clic a "*Next*".(Fig. 3.2)

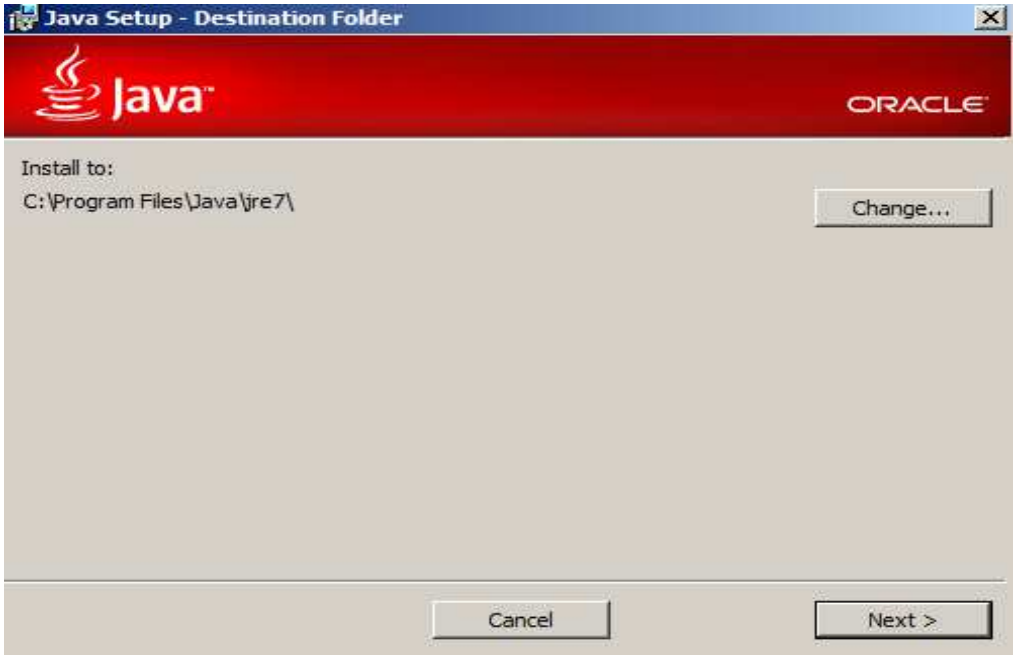

Fig. 3.2. Directori on es guarda el JRE

Un cop fet això surt un missatge on avisa que la instal·lació s'ha produït correctament i ja es disposa d'una de les eines per a poder dur a terme l'aplicació.

3.2. SDK Android

SDK és un conjunt de d'eines de desenvolupament que permeten al programador crear o desenvolupar una determinada aplicació.

Es troba dividit en dues parts: *SDK Components* i *SDK Starter Package*.

Per descarregar el paquet cal anar a la següent pàgina web: http://developer.android.com/sdk/index.html i seleccionar aquell que convingui pel determinat SO, en aquest cas l'arxiu: **installer_r16-windows.exe**

A continuació s'expliquen els passos a seguir per a la correcta instal·lació i funcionament:

1. Fer clic sobre l'executable descarregat, mostra una pantalla d'inici a la instal·lació, seleccionar el botó de "*Next*"(Fig. 3.3).

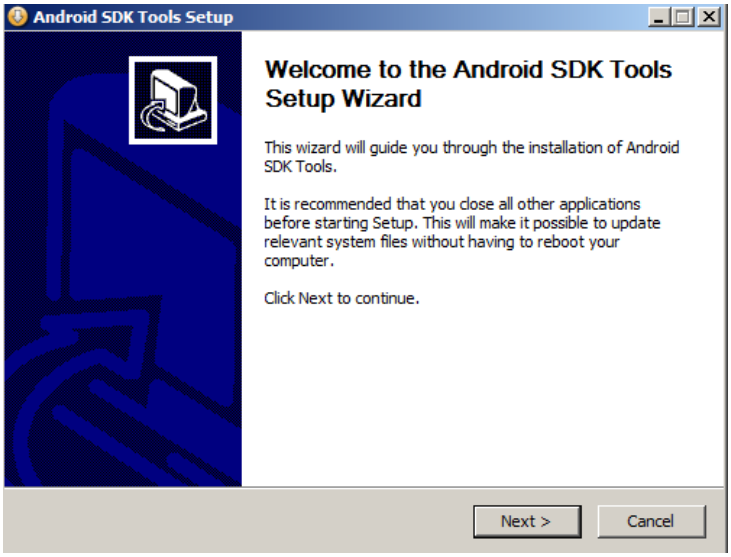

Fig. 3.3. Inici instal·lació SDK

2. La següent pantalla indica que Java SE Development Kit es troba instal·lat i que Android SDK depèn del JDK, d'aquí ve la importància de que estigui ben instal·lat. A continuació, fer clic a "*Next*"**.** (Fig. 3.4).

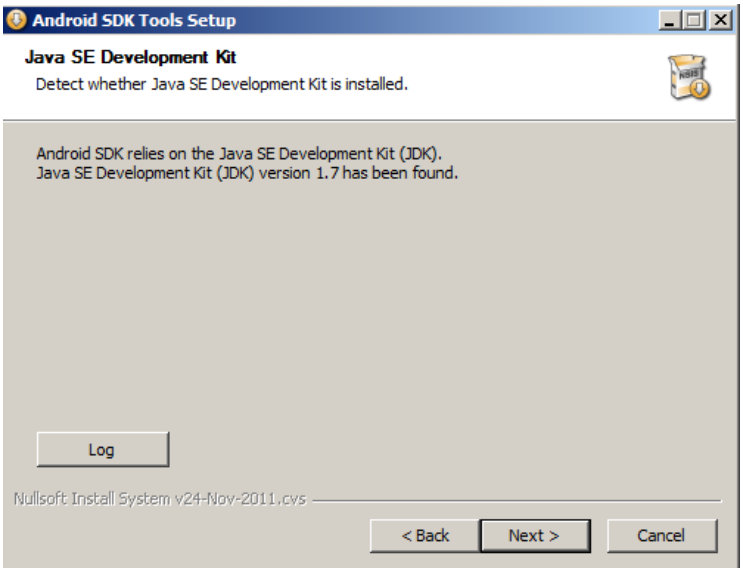

Fig. 3.4. Segon pas de la instal·lació

3. En aquest pas es demana la ubicació on es vol guardar la instal·lació del SDK *Tools*, s'escull la ruta per defecte, es fa clic una altre vegada a "*Next*" i al botó de "*Install*".

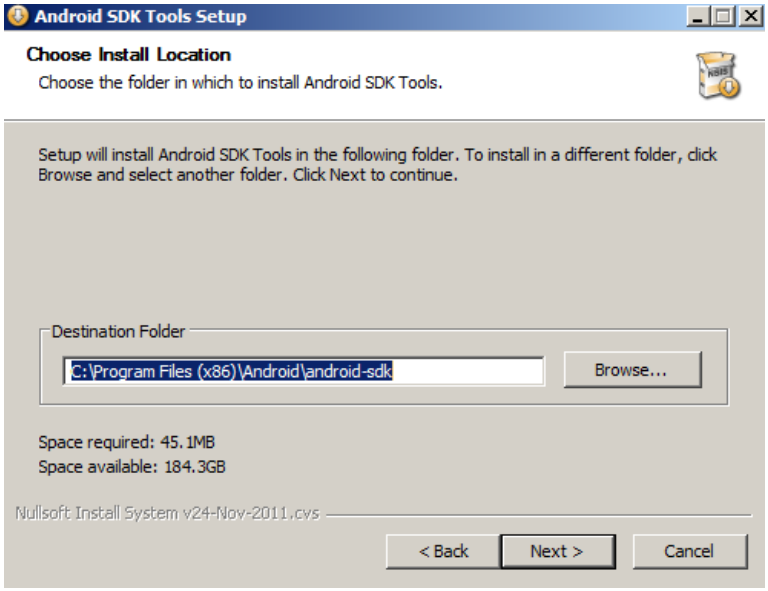

Fig. 3.5. Ruta per a guardar la instal·lació

Els arxius corresponents al SDK s'han de deixar situats a una carpeta on es pugui accedir fàcilment des de l'Eclipse.

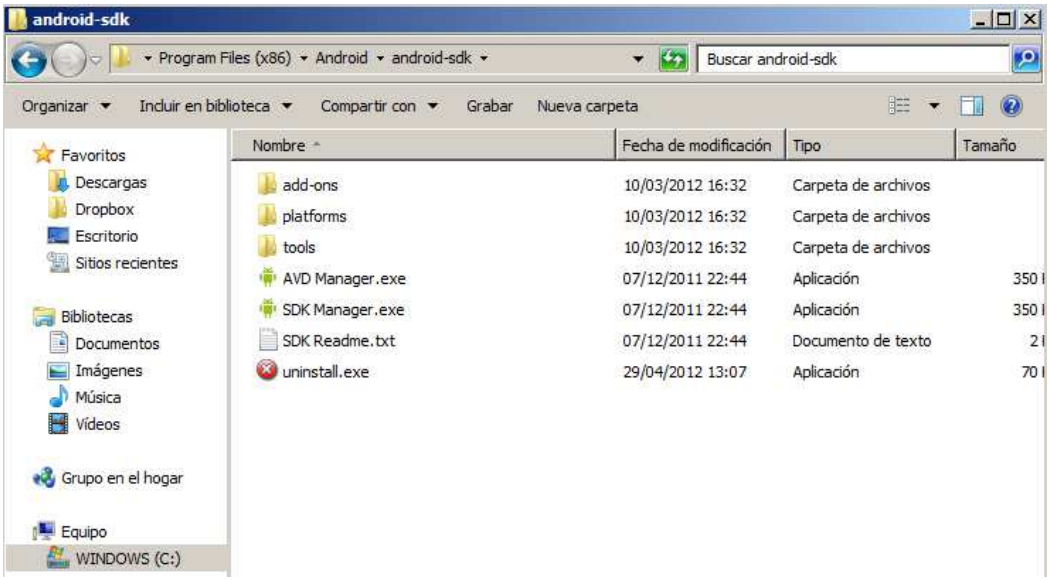

Fig. 3.6. Directori on es troba ubicada la carpeta Android-SDK

4. Cal instal·lar els complements que es desitgin d'Android depenent del tipus de versió. Això es fa mitjançant *SDK Manager.exe* que es troba dins de la carpeta esmentada anteriorment.

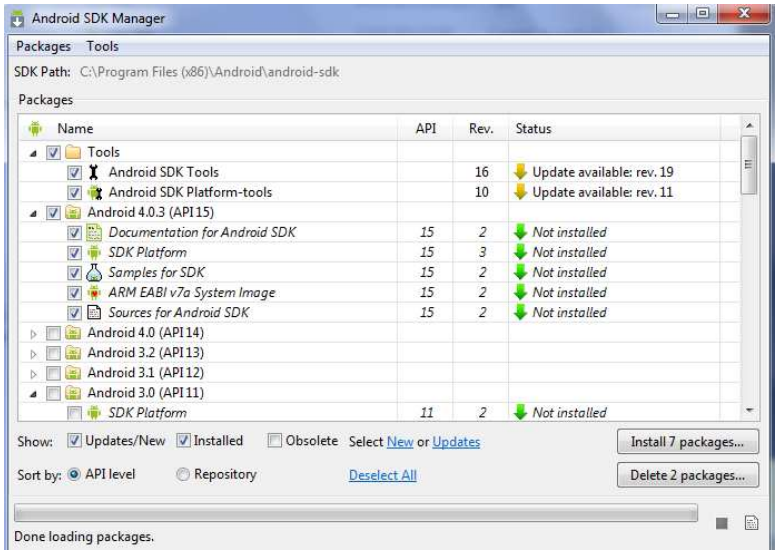

Fig. 3.7. Android SDK Manager

A l'imatge anterior (Fig. 3.7.), el *SDK Manager* mostra les diferents versions que hi ha amb les corresponents *APIs* i dóna la opció d'instal·lar els paquets o d'esborrar-los. A més a més, busca actualitzacions de cadascun d'ells.

5. A continuació dins de la mateixa carpeta hi ha el *AVD Manager.exe* que cal configurar. En la figura que ve a continuació (Fig. 3.8) es pot observar diverses versions afegides pel AVD.

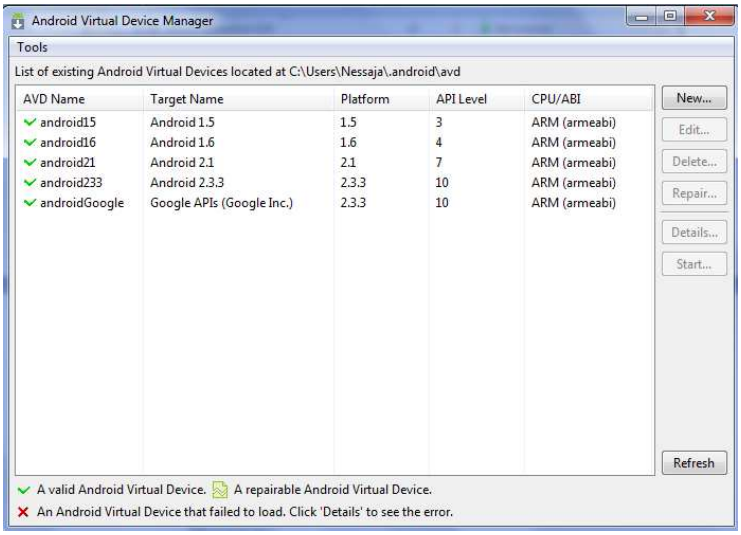

Fig. 3.8. Android Virtual Device Manager

Per fer-ho cal fer clic a "*New*", apareix una altre finestra (Fig. 3.9) en la qual s'han de configurar principalment dos dels camps que apareixen:

- *Name*: Indicar un nom amb el qual s'identifica la versió que es vol afegir.
- *Target*: Escollir entre les diferents versions amb les seves APIs corresponents en funció de quina versió es vol per a visualitzar l'aplicació al emulador.

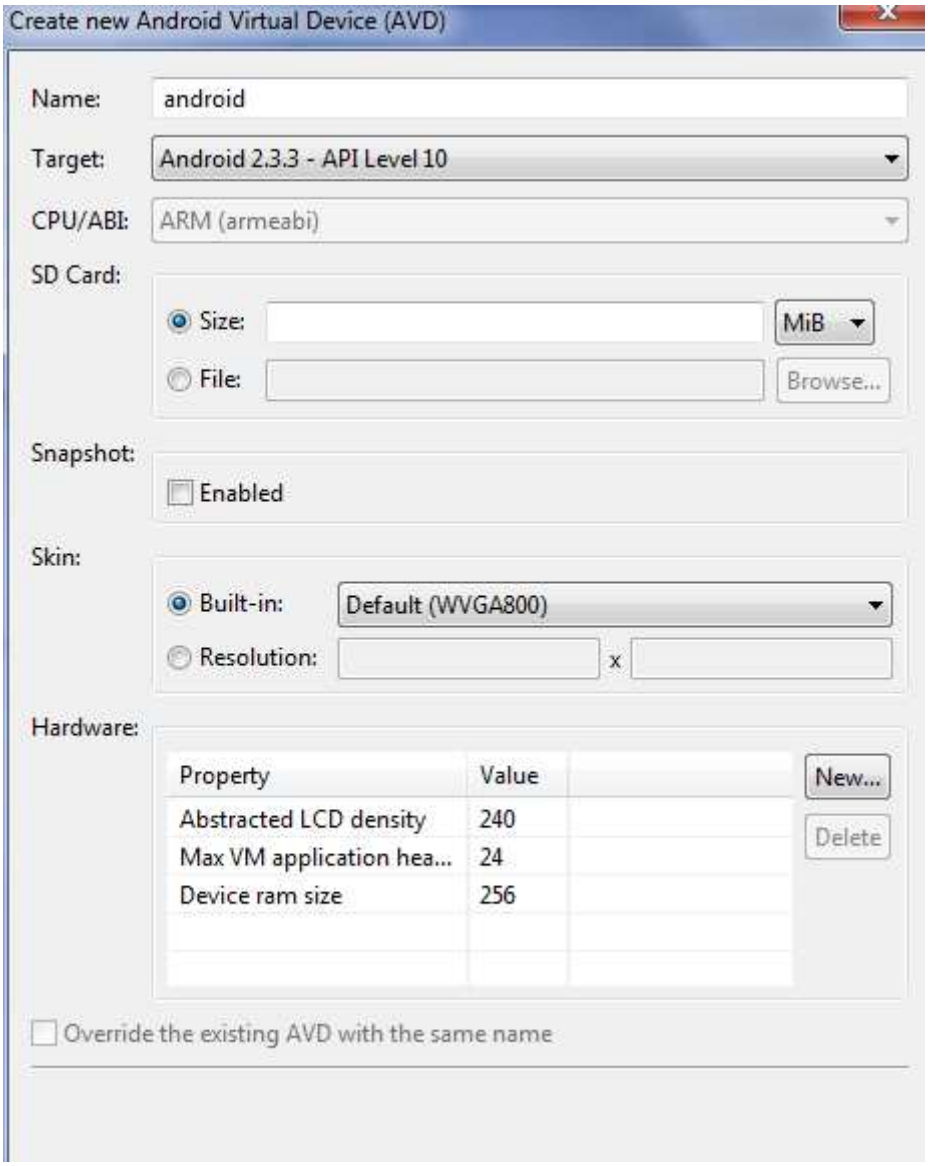

Fig. 3.9. Nou ADV

6. Un cop creat, dins la pantalla principal del AVD fer clic a "*Start*", com es pot observar l'emulador està funcionament(Figura 3.10).

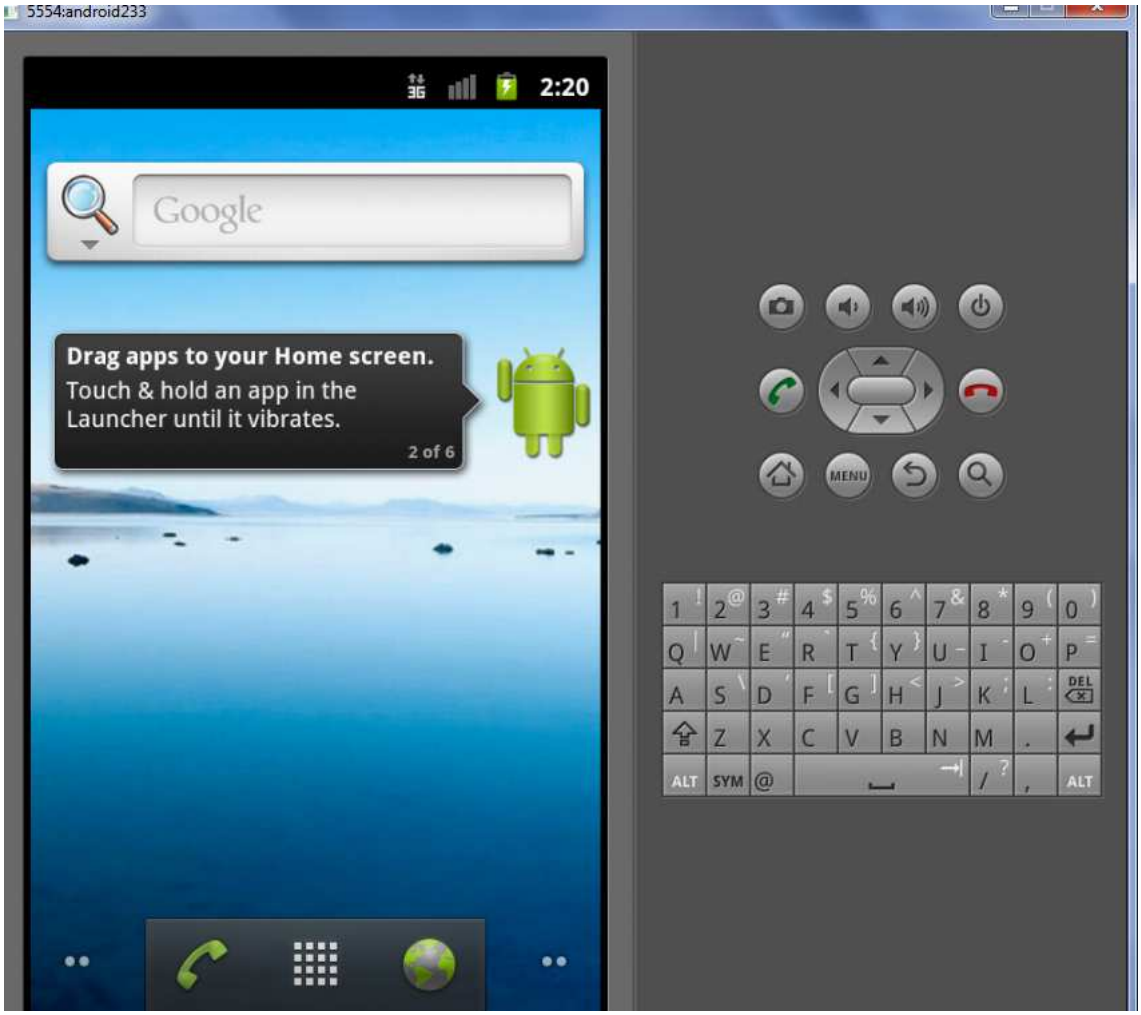

Fig. 3.10. Emulador Android 2.3.3 (Api 10)

3.3. Entorn de programació Eclipse

Per a poder dur a terme la programació de l'aplicació cal fer ús d'Eclipse.

Eclipse és un entorn de desenvolupament *Open Source* basat en Java. És utilitzat per els desenvolupadors de Google (creadors d'Android) i és gratuït. Per aquest motiu s'ha decidit treballar amb aquesta opció tot i que existeixen d'altres igual de vàlides com és el cas de *Netbeans*.

A continuació es mostra pas a pas el que cal fer per instal·lar-lo i configurar-lo.

1. Cal descarregar Eclipse de la següent pàgina web: www.eclipse.org/downloads/

2. Escollir la versió més adequada fixant-se en quin tipus de sistema operatiu es té, en aquest cas s'ha descarregat la versió més actualitzada per a Windows 7 (64 bits) que és **Ecplise Indigo v 3.7.2**.

   Eclipse es distribuït mitjançant un instal·lador (carpeta comprimida .zip) amb la qual cosa el programa es troba llest per a ser configurat.

3. Es descomprimeix la carpeta i es selecciona l'arxiu *eclipse.exe*. El primer que s'ha d'escollir és el directori de treball (***workspace***).

   Per últim però, cal instal·lar un complement d'Eclipse que s'explica a continuació.

3.3.1. ADT

Per a instal·lar aquest complement, un cop obert ***eclipse.exe*** cal seleccionar ***"Help"→"Software Updates"*** i dins de la finestra de complements cal seleccionar ***"Available software"→"Add site"→"Add".*** A continuació afegir la ubicació de la pàgina web d'actualització de ADT: http://dl-ssl.google.com/android/eclipse per tal de descarregar els *plug-ins* per programar en el sistema operatiu Android.

Finalment cal reiniciar Eclipse. Un cop reiniciat, seleccionar: "***Window"→"Preferences"→"Android"*** i afegir la ruta on es troba el SDK Android ja que si no es fa això al reiniciar l'entorn de programació poden aparèixer missatges d'error.

4. Cos d'un projecte Android

Abans de començar a explicar com s'ha desenvolupat i com funciona la aplicació principal, cal explicar com crear un nou projecte Android i la seva estructura, ja que sense conèixer això no es pot entendre el funcionament de la aplicació.

4.1. Creació d'un nou projecte Android

Un cop obert l'Eclipse, el que s'ha de fer es seleccionar ***"File"→"New"→"Other"*** (Fig. 4.1).

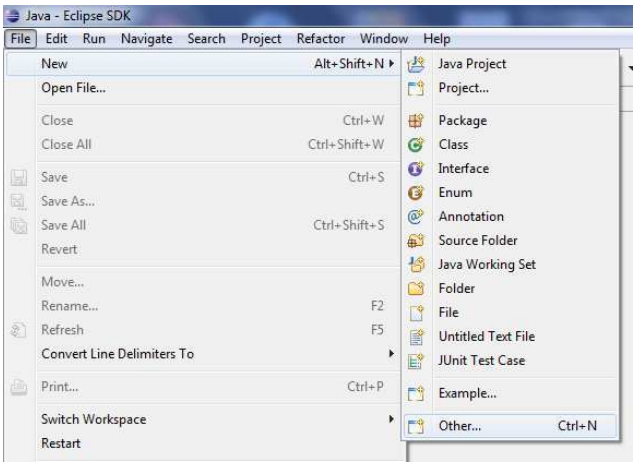

Fig. 4.1. Creació nou projecte Android

A continuació, s'obre una finestra (Fig. 4.2) en la qual s'ha de seleccionar "*Android Project*".

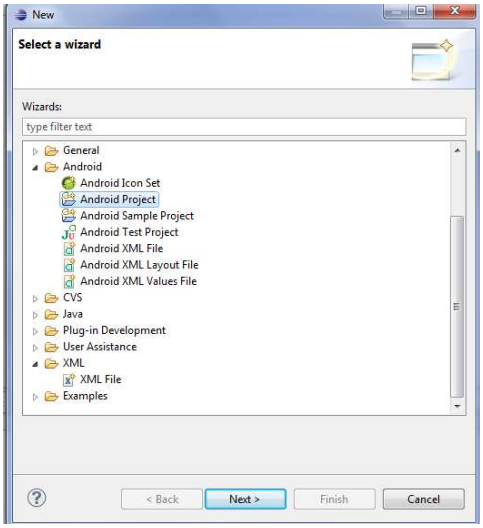

Fig. 4.2. Selecció del tipus de projecte

Fer clic a **"***Next***"** i apareix una altra finestra en la qual es demana que s'introdueixi el nom que es vol posar al projecte (Fig. 4.3)

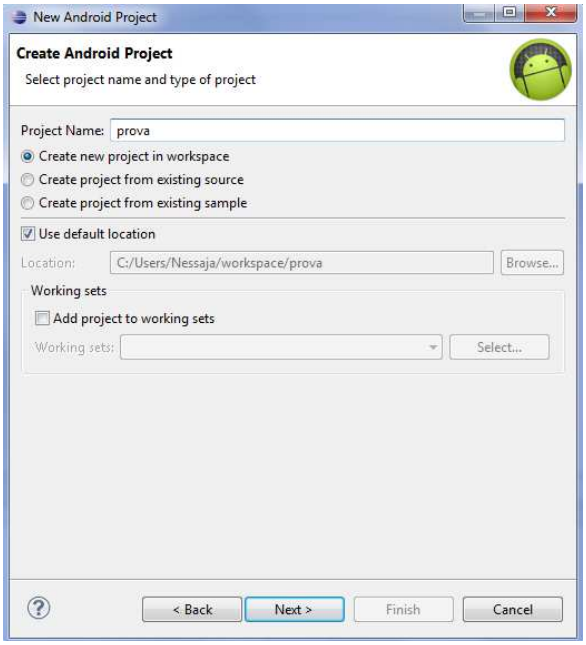

Fig. 4.3. Nom del projecte

Es torna a fer clic a **"***Next***"** i apareix la tercera finestra en la qual es selecciona el tipus de versió i número d'API (Fig. 4.4). En aquest cas s'ha seleccionat "Android 2.3.3" però en funció de l'aplicació que es vulgui es pot seleccionar una versió o una altre.

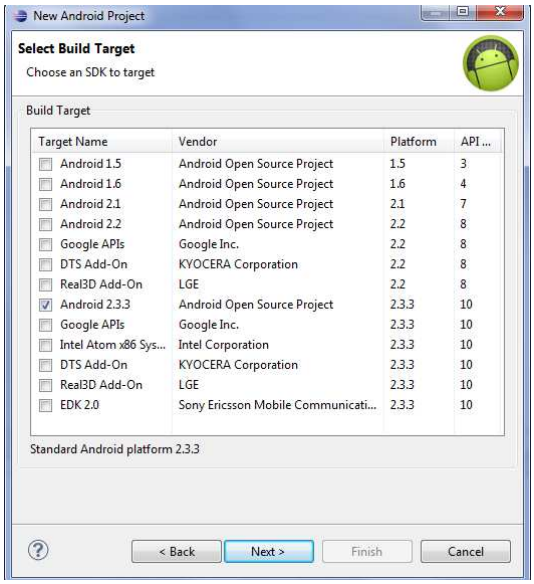

Fig. 4.4. Selecció de la "*target*"

Fer clic a *"Next"* i demana que s'especifiqui un nom de paquet (Fig. 4.5), s'introdueix i es fa clic a *"Finish"*, amb la qual cosa ja es té creat el primer projecte Android.

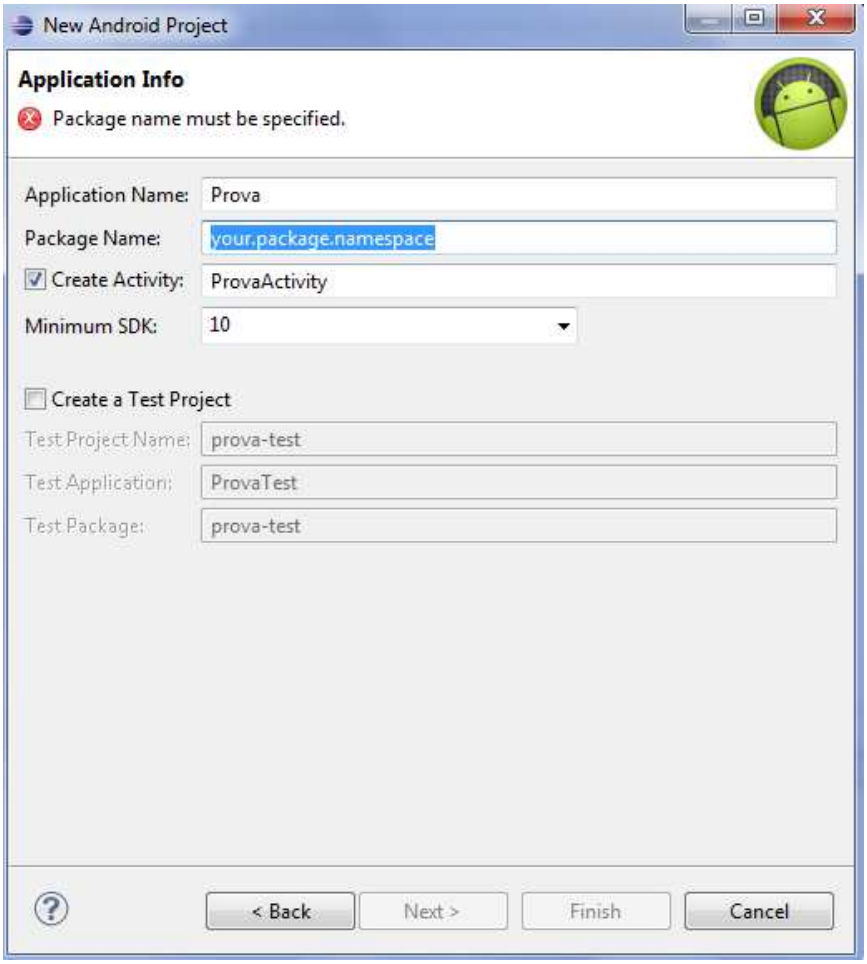

Fig. 4.5. Nom del paquet del projecte

4.2. Estructura del nou projecte

Al crear una nova aplicació, Eclipse genera automàticament una estructura de carpetes (arbre de directoris) (Fig. 4.6.) que serà comuna pels diferents tipus d'aplicacions independentment de la seva grandària o complexitat.

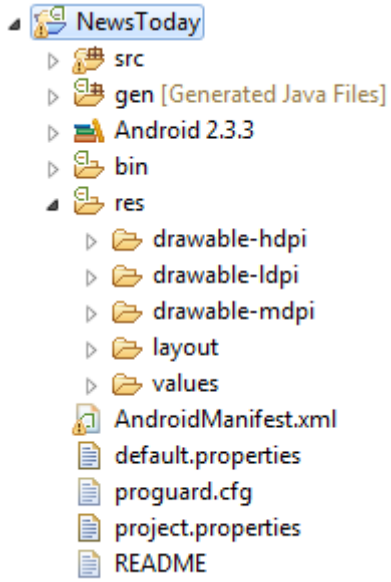

Fig. 4.6. Estructura de carpetes

4.2.1 Carpeta "/src"

Està formada per tot el codi font de la pròpia aplicació, codi de la interfície gràfica (com per exemple la funció que es necessita per a que un determinat botó dugui a terme una acció quan es fa clic sobre ell), classes auxiliars,etc...Al crear una aplicació, Eclipse genera per defecte el codi de la classe *Activity* que correspon a la pantalla principal de la aplicació, basant-se en la estructura de Java.

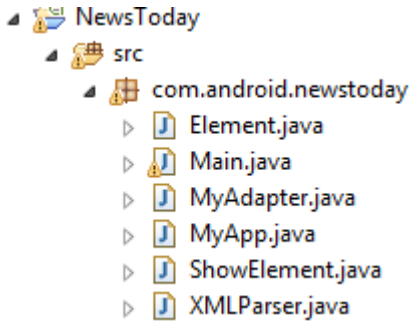

Fig. 4.7. Classes del codi .java que pertanyen a la carpeta "src"

4.2.2. Carpeta "/gen"

Conté els elements del codi generats automàticament un cop es compila el projecte. Cada cop que es genera el projecte, Android genera una sèrie de fitxers font en Java que van dirigits al control dels diferents recursos de la nostra aplicació.

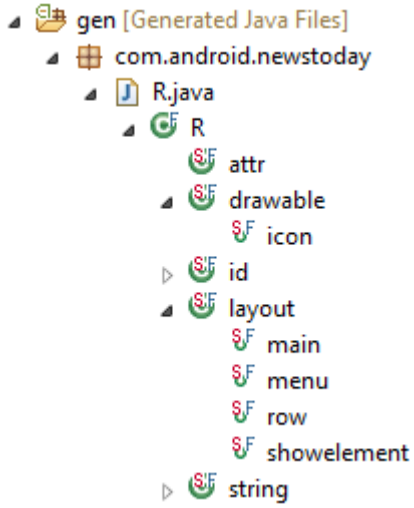

Fig. 4.8. Elements del codi generats automàticament

El més important d'aquesta carpeta és la **classe R** i el fitxer **R.java.** Aquesta classe conté constants formades per IDs de tots els recursos de la aplicació inclosos a la carpeta /res. (Fig. 4.9).

La primera classe (*drawable*) conté els noms de totes les icones que formen l'aplicació, tant pel menú com per les diferents funcionalitats.

La segona classe (*id)* conté tots els identificadors utilitzats al llarg de tota l'aplicació, ja sigui per fer referència als botons del menú (categories, obrir al navegador, localització...) o pel text de la noticia.

La tercera classe (*layout*) fa referència al arxius .xml a on es configura tot el tema d'aparença dins de l'aplicació.

L'última classe (*string*) té com atribut "títol" que correspon al nom de l'aplicació.

```
package com.android.newstoday;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_actualitat=0x7f020000;
        public static final int ic_menu_apli=0x7f020001;
        public static final int ic_menu_categories=0x7f020002;
        public static final int ic_menu_loc=0x7f020003;
        public static final int ic_menu_navegador=0x7f020004;
        public static final int icon=0x7f020005;
        public static final int marcador_google_maps=0x7f020006;
        public static final int news=0x7f020007;
    }
    public static final class id {
        public static final int BtnAnimar=0x7f050002;
        public static final int BtnCentrar=0x7f050001;
        public static final int BtnMover=0x7f050003;
        public static final int BtnSatelite=0x7f050000;
        public static final int LL01=0x7f050005;
        public static final int LL02=0x7f050006;
        public static final int actualitat=0x7f05000c;
        public static final int aplicacio=0x7f050009;
        public static final int categories=0x7f05000a;
        public static final int cattitol=0x7f050007;
        public static final int cinema=0x7f05000e;
        public static final int economia=0x7f05000d;
        public static final int esports=0x7f050010;
        public static final int grupCategories=0x7f05000b;
        public static final int local=0x7f050012;
        public static final int mapa=0x7f050004;
        public static final int navegador=0x7f050008;
        public static final int politica=0x7f05000f;
        public static final int tecnologia=0x7f050011;
        public static final int txtAutor=0x7f050015;
        public static final int txtDescripcio=0x7f050016;
        public static final int txtElement=0x7f050013;
        public static final int txtTitol=0x7f050014;
    }
    public static final class layout {
        public static final int androidmapas=0x7f030000;
        public static final int main=0x7f030001;
        public static final int menu=0x7f030002;
        public static final int row=0x7f030003;
        public static final int showelement=0x7f030004;
    }
    public static final class string {
        public static final int titol=0x7f040000;
    }
}
```

Fig. 4.9. R.java

4.2.3. Carpeta "/bin"

Conté els arxius amb l'aplicació compilada. Està formada per *"/bin/res/classes.dex"* que és l'executable obtingut a partir de les classes compilades i *"/bin/res/NewsToday.apk"* que conté l'aplicació Android per a poder ser instal·lada a un dispositiu que contingui aquest sistema operatiu.

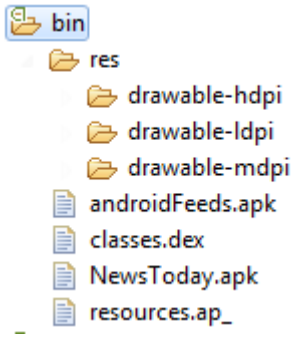

Fig. 4.10. Carpeta bin amb els diferents executables

4.2.4. Carpeta "/res"

Correspon als fitxers de recursos que es necessiten per a dur a terme el projecte: imatges, cadenes de text (*strings*)… Els diferents recursos es troben distribuïts per les diferents carpetes:

- */res/drawable/.* → Conté les imatges utilitzades a l'aplicació. Es troben dividits en diferents recursos (*drawable-hdpi, drawable-ldpi, drawable-mdpi*) en funció de la resolució del dispositiu.

- */res/layout/.* → Conté els fitxers de definició de les diferents pantalles corresponents a l'interfície gràfica. En aquest cas només està dividit en una sola carpeta *layout* però pot estar dividida en una altra anomenada *"/layout-land"* en funció de la orientació que es vulgui per a la pantalla del dispositiu.

- */res/values/.*→Conté els altres recursos de l'aplicació. En aquest cas el fitxer "*strings.xml*" conté les cadenes de text.

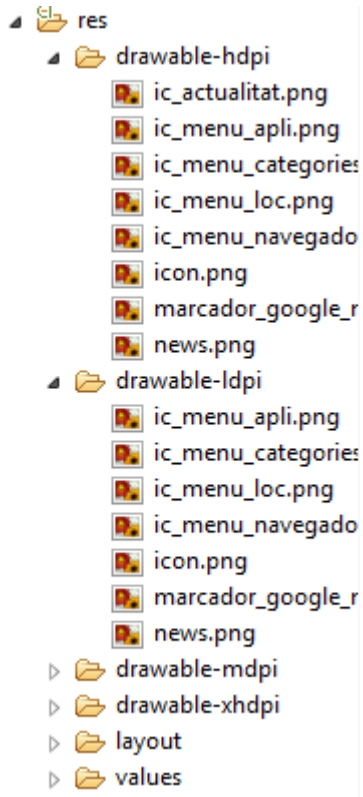

Fig. 4.11. Fitxers de recursos

4.2.5. Fitxer AndroidManifest.xml

Conté la definició en XML dels aspectes principals de la aplicació com per exemple, el nom del fitxer que carrega quan s'executa la aplicació, la versió del sistema operatiu que s'utilitza per a compilar-la…, els components (pantalles, missatges…) i els permisos necessaris per a la seva execució com pot ser el permís per a Internet, per a que la aplicació pugui connectar-se si es necessari.

Inicialment cal fer la declaració de l'arxiu .xml, a continuació s'indica el nom del paquet (*package*) i la versió, per últim cal declarar l'activitat o si hi ha més d'una, les activitats a on s'indica si hi ha associada alguna imatge, el nom de l'etiqueta i el nom de l'activitat que està formada per un arxiu .java.

Per obtenir els permisos adients cal declarar-los mitjançant la sentència: "*<uses permission android_name*= "nom del permís"/>".

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.android.newstoday"
      android:versionCode="1"
      android:versionName="1.0">
    <application android:icon="@drawable/news" android:label="@string/titol" android:name=".App">
        <uses-library android:name="com.google.android.maps" />
        <activity android:name=".Main"
                  android:label="@string/titol">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="com.android.newstoday.ShowElement"></activity>
        <activity android:name="com.android.newstoday.AndroidMapas"></activity>
    </application>

    <uses-sdk android:minSdkVersion="10" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_MOCK_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.VIBRATE"/>
    <uses-permission android:name="android.permission.LIGHTS"/>


</manifest>
```

Fig. 4.12. Android Manifest.xml

5. Aplicació “News Today”

5.1. Introducció

Per a poder dur a terme de forma pràctica els coneixements explicats en els punts anteriors, aquí s’explica la creació i funcionament d’un projecte Android tot centrat en l’aplicació “*News Today*”.

L’aplicació es troba dividida en tres funcionalitats bàsiques:

- Mostrar un llistat de notícies d’actualitat les quals es van actualitzant a través de la tècnica RSS. Aquestes notícies es troben dividides en categories (Música, esports, política...) per a que l’usuari tingui la opció d’escollir aquella temàtica que més l’interessi. A més a més hi ha l’opció de veure la notícia dins de l’aplicació o en el propi navegador.

- Cada cop que hi ha una notícia nova no llegida anteriorment a l’executar l’ aplicació, es llança una notificació d’avís de noves notícies.

- Geolocalització mitjançant Google Maps: Aquest apartat té la possibilitat de localitzar a on es troba l’usuari i marcar els cinemes que estan més a prop d’ell.

5.2. Disseny de la aplicació: Carregar i executar el llistat de notícies

Un cop creat el nou projecte tal i com s’ha explicat anteriorment (Capítol 4), cal crear l’aplicació la qual s’explica amb detall a continuació.

5.2.1. main.xml

Aquest arxiu és l’encarregat de configurar el disseny de la aplicació, és a dir, l’aparença que tindrà de cara a l’usuari.

En aquesta arxiu s’utilitza *LinearLayout, TextView i ListView*:

- ***LinearLayout**:* Estableix els components visuals de l'aplicació de forma lineal tal i com indica el seu nom, és a dir, cada component es troba un darrere d'un altre ja sigui de forma horitzontal o vertical.

En aquest cas s'ha escollit que la orientació sigui vertical per tant, les notícies seran mostrades una sota de l'altre.

Pel que fa a l'amplada i l'alçada és pot escollir entre "*fill_parent*" o "*wrap_content*" en funció de si es vol expandir el contingut fins a poder ocupar tota la pantalla (*fill_parent*) o simplement expandir-lo ajustant-se al text o imatge que contingui (*wrap_content*).

- ***TextView***: Utilitzat per a mostrar textos dins la aplicació, en aquest cas, quan s'executa el programa i carrega les notícies a dalt apareix un títol anomenat "Actualitat" això és gràcies a la utilització d'aquesta etiqueta.

- ***ListView**:* Tipus de vista en forma de llista ("*android:id="@android:id/list*") que mostrarà la aplicació.

5.2.2. Main .java

Correspon a la activitat principal de l'aplicació que hereta de la classe *ListActivity,* per tal de facilitar les coses a l'hora de mostrar el llistat de notícies.

Per a poder treballar amb el *ListActivity* és necessari tenir un *ListView.*

Quan una activitat és creada per primer cop es necessita instanciar mitjançant el següent mètode:

```
/** Metode cridat quan la actividad es crea */
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
```

Fig. 5.1. Mètode onCreate

Com que es vol carregar una llista de noticies, la qual cosa és una tasca amb una determinada espera, hi ha l'opció de mostrar un avís mentre es carreguen les notícies, per fer-ho es necessita un fil d'execució. Per tant, s'ha creat la següent variable i funció:

- *" private ProgressDialog progressDialog; "*
- *"private void loadData()"*

A continuació es necessita crear el gestor entre fils de execució per enviar un missatge d'un fil a un altre quan, la càrrega ja ha finalitzat:

```
private final Handler progressHandler = new Handler() {
    public void handleMessage(Message msg) {
        setData();
        progressDialog.dismiss();
    }
};
```

Fig. 5.2. Classe Handler

Un altre aspecte important és la creació de "*Intents*", cal crear-ne un que contingui una variable prèviament declarada i inicialitzada (POSICIO_CLAU) la qual s'identifiqui amb la clau que envia les dades mitjançant intencions (Intents) que comuniquen les dues activitats *Main* i *MostrarElement*.

Un cop carregades les dades, hi ha la opció de mostrar un diàleg preguntant a l'usuari si vol tornar a carregar-les, això es duu a terme mitjançant el següent codi:

```
if (data != null) {
    AlertDialog.Builder builder = new AlertDialog.Builder(Main.this);
    builder.setMessage("Ja ha carregat les notícies, està segur que ho vol tornar a fer?")
           .setCancelable(false)
           .setPositiveButton("Si", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialeg, int id) {
                   loadData();
               }
           })
           .setNegativeButton("No", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialeg, int id) {
                    dialeg.cancel();
               }
           })
           .create()
      .show();
/**
 * Si no hi ha dades encara, carreguem amb el loadData()
 */
} else {
    loadData();
}
```

Fig. 5.3. Opció de tornar a carregar les notícies

5.2.3. Adaptador.java

Es tracta d'un adaptador personalitzat que rep un llistat d'elements . Hereta de la classe *ArrayAdapter <Element>* i dins de la classe es treballa amb dues variables d'instància: *layoutInf* i *LinkedList*. La primera es fa servir per a instanciar el disseny a partir d'un arxiu XML i la segona es tracta d'un llistat d'objectes representats per la classe *Element*.

Per últim, només cal obtenir el títol de l'article i assignar-lo al *TextView* corresponent dins del *layout*.

5.2.4. Element.java

En aquest arxiu és on es guarda l'informació de cada element del llistat de notícies. Per tant la seva funció és la de obtenir i modificar el títol, enllaç, autor, descripció i data de publicació mitjançant els diferents mètodes.

A continuació (Fig. 5.4) tenim algun dels diferents mètodes declarats corresponents a l'autor, títol, enllaç, descripció i data de la notícia.

```
public String getAutor(){
    return this.autor;
}

public void setTitol(String titol) {
    this.titol = titol.trim();
}

public void setLink(String link) {
    this.link = link;
}

public void setDescripcio(String descripcio) {
    this.descripcio = descripcio.trim();
}

public void setData(String data) {
    try {
        this.data = FORMAT.parse(data);
    } catch (java.text.ParseException e) {
        e.printStackTrace();
    }
}
```

Fig. 5.4. Mètodes per obtenir i modificar els atributs

5.2.5. App.java

Aquesta classe es utilitzada per a representar dades volàtils dins de la aplicació.
Per indicar que la aplicació és d'aquest tipus cal modificar l'arxiu *AndroidManifest.xml* i afegir *android:name=".App"*.

Dins d'aquesta classe es guarda:

- Llistat de notícies
- L'opció seleccionada (navegador o aplicació) per a visualitzar la noticia.

Aquesta última opció s'explica més endavant com s'ha creat i com funciona.

```
package com.android.newstoday;

import java.util.LinkedList;
import android.app.Application;

public class App extends Application {
      private LinkedList<Element> data = null;
      private int opcioSeleccionada = Main.APP_VISTA;

      public LinkedList<Element> getData(){
          return this.data;
      }
      public void setData(LinkedList<Element> d){
          this.data = d;
      }

      public int getOpcioSelec(){
          return this.opcioSeleccionada;
      }

      public void setOpcioSelec(int opcioSeleccionada) {
          this.opcioSeleccionada = opcioSeleccionada;
      }
}
```

Fig. 5.5. App.java

5.2.6. mostrarelement.xml

En aquest arxiu .xml a partir de *TextViews* es determina el títol, autor i descripció de la noticia. Per a fer-ho cal crear un identificador (*id*) i configurar l'amplada i llargada que es vol que ocupin les notícies i els seus atributs. A continuació, es mostra el *TextView* del títol (Fig. 5.6).

```
<TextView
  android:id="@+id/txtTitol"
  android:layout_height="wrap_content"
  android:layout_width="wrap_content">
</TextView>
```

Fig. 5.6. TextView "Títol"

Així és com ha de quedar la vista final (Fig. 5.7.):

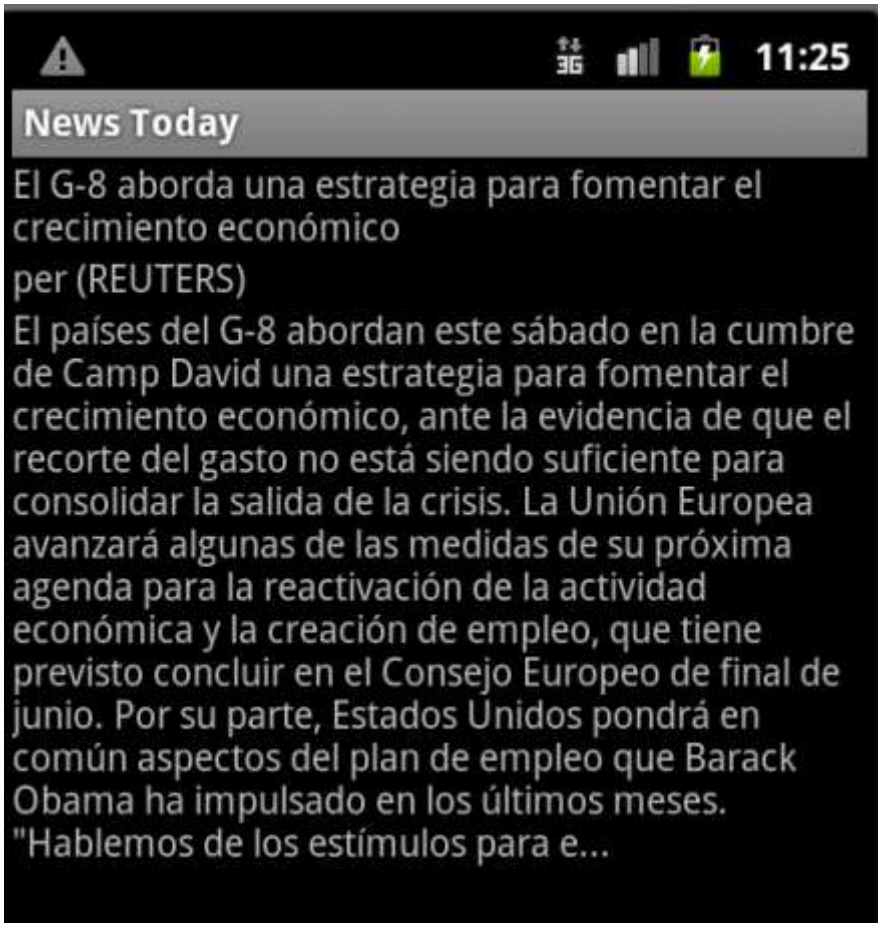

Fig. 5.7. Vista final del títol, autor i descripció

5.2.7. MostrarElement.java

Aquesta classe hereta de *Activity*. Cal cridar al mètode *onCreate* ja que s'està creant una nova activitat. També s'ha de declarar una variable de tipus *Intent* i aquesta ha de portar la posició de l'element a visualitzar, en el cas que sigui null (el *intent* no envia res) se li assigna un -1.

A continuació s'obté el llistat de notícies, el títol, autor i descripció que l'usuari vol visualitzar. Si l'identificador és invàlid torna a l'activitat principal (*Main*) enviant un valor de -1 per indicar que hi ha error (Fig. 5.8)

```
/**
 * Obtenim el llistat d'articles
 */
App estatApp = ((App)getApplication());
Element e = estatApp.getData().get(posicio);


TextView txtTitol = (TextView)findViewById(R.id.txtTitol);
txtTitol.setText(e.getTitol());

TextView txtAutor = (TextView)findViewById(R.id.txtAutor);
txtAutor.setText("per " + e.getAutor());

TextView txtDesc = (TextView)findViewById(R.id.txtDescripcio);
txtDesc.setText(e.getDescripcio());


Intent backToMainActivity = new Intent(this, Main.class);
backToMainActivity.putExtra(Main.POSICIO_CLAU, -1);
startActivity(backToMainActivity);
```

Fig. 5.8. MostrarElement.java

5.3. Disseny de la aplicació: Menú

Correspon als botons que es poden veure i interactuar quan es prem la tecla menú del terminal. En concret, en aquest apartat s'explica el botó de "Obrir al navegador", "Veure a l'aplicació" i "Categories", ja que el botó de "Localització" serà explicat al punt 5.5.

5.3.1. menu.xml : Botó "Obrir al navegador"

Aquest botó és la opció que l'usuari pot escollir si vol veure la notícia que hagi seleccionat a través del navegador. Per a aquest botó cal crear un arxiu .xml que s'anomena *menu.xml*.

Com tot arxiu d'aquest tipus cal declarar-lo inicialment mitjançant el següent codi:

```
<?xml version="1.0" encoding="utf-8"?>
	<menu xmlns:android="http://schemas.android.com/apk/res/android">
```

Com que aquest botó es troba dins del menú cal crear-lo com a *item*, afegir un identificador per a poder més tard afegir funcions sobre ell i opcionalment afegir-hi una imatge amb la

qual serà representat(Fig. 5.9), ja que quan es crea un nou projecte, per defecte, porta una imatge predefinida que correspon al "logo" d'Android.

```
<item android:id="@+id/navegador"
    android:title="Obrir al navegador"
    android:icon="@drawable/ic_menu_navegador"/>
```

Fig. 5.9. Botó "Obrir al navegador"

5.3.2. menu.xml : Botó "Veure a l'aplicació"

Aquest botó és la opció que l'usuari pot escollir si vol veure la notícia que hagi seleccionat a través de la aplicació. Al fer clic sobre aquesta opció, l'usuari veurà el títol, l'autor i la descripció de la noticia.

Per fer-ho no cal crear un nou arxiu .xml sinó que tot el que es troba dins del menú es farà en el mateix arxiu. Es crea un altre identificador de tipus *item*, amb un títol i una imatge (Fig. 5.10).

```
<item android:id="@+id/aplicacio"
    android:title="Veure a l'aplicació"
    android:icon="@drawable/ic_menu_apli" />
```

Fig. 5.10. Botó "Veure a l'aplicació"

5.3.3. menu.xml : Botó "Categories"

Aquest botó també es un element *item* però amb petites diferències amb els anteriors. Com que es vol dividir en diferents categories en funció de si la noticia és de economia, cinema, esports... cal crear un element **"***menu***"** que es troba dins de l'*item*, les categories però, a la vegada queden dins d'un element anomenat "*group*"(Fig. 5.11).

El primer *item* anomenat categories correspon al botó que es troba dins del menú, un cop es fa clic sobre ell apareixen els diferents apartats en funció del tipus de notícia.

El "grupCategories" són aquestes categories mostrades per a poder escollir l'opció que més s'adeqüi al gust de l'usuari.

```
<item android:id="@+id/categories"
    android:title="Categories"
    android:icon="@drawable/ic_menu_categories">

    <menu>
            <group android:id="@+id/grupCategories" android:checkableBehavior="single">

                <item android:id="@+id/actualitat"
                android:title="Actualitat"
                android:icon="@drawable/icon"
                android:background="#7f0026" />

                <item android:id="@+id/economia"
                android:title="Economia"
                android:icon="@drawable/icon"
                android:background="#7f0026" />

                <item android:id="@+id/cinema"
                android:title="Cinema"
                android:icon="@drawable/icon"
                android:background="#7f0026"/>
```

Fig. 5.11. Categories

A més a més, igual que en el cas anterior del navegador i de l'aplicació, cal afegir un títol que es veurà escrit en el botó i opcionalment una imatge a la icona i color de fons.

5.3.4. Vista final del menú

Fig. 5.12 Menú

5.4. Disseny de la aplicació: Notificacions

En aquest apartat s'explica el sistema de notificacions creat per a que cada cop que hagi una noticia nova a l'aplicació llanci una.

Per a crear-ho cal afegir el codi a la activitat principal (*Main*). Primerament s'ha d'obtenir una referència al servei de notificacions, a continuació cal configurar la notificació, el *intent* i el "*AutoCancel"* que serveix per a quan es prem sobre la notificació aquesta desaparegui. Opcionalment es pot afegir so, llum o vibració per a quan es rep aquesta i per últim cal afegir l'ordre d'enviar la notificació.

Per obtenir la referència s'ha de crear una variable de tipus *String* i una altra de tipus *NotificationManager* que pertany a la llibreria d'Android.

Per a configurar la notificació es crea una variable de tipus *enter* de la qual s'obtindrà la referència a la imatge que acompanya a la notificació i una de tipus *CharSequence* que llançarà el missatge de text dient que hi han noves notícies . També hi ha la opció de dir-li cada quan es vol que aparegui la notificació, això es fa creant una variable i assignant-li el temps que es desitgi. A més a més, cal crear la pròpia notificació de la següent forma:

```
Notification notif = new Notification (icono, txtEstat, hora);
```

Per a configurar el *Intent*, primer es crea una variable de tipus *CharSequence* que conté el títol que apareixerà junt amb el contingut de quina és la nova notícia i després es crea un nou *Intent*.

El "*AutoCancel*" es configura de la següent manera:

```
notif.flags |= Notification.FLAG_AUTO_CANCEL;
```

on "notif" és la variable de la notificació creada i "Notification" és el tipus que prové de les llibreries d'Android. Les opcions de vibració i llums:

```
notif.defaults |= Notification.DEFAULT_VIBRATE;
notif.defaults |= Notification.DEFAULT_LIGHTS;
```

Per últim s'envia la notificació:

```
notManager.notify(NOTIF_ALERTA_ID, notif);
```

on "**NOTIF_ALEARTA_ID"** és una variable privada i estàtica inicialitzada a 1:

```
private static final int NOTIF_ALERTA_ID = 1;
```

El codi ha de quedar de la següent manera (Fig. 5.13):

```
private void notificacio(String notificacioTxt){

    String ntf = Context.NOTIFICATION_SERVICE;
    NotificationManager notManager =(NotificationManager) getSystemService(ntf);

    int icono = android.R.drawable.stat_sys_warning;
    CharSequence txtEstat = "Noves noticies!";
    long hora = System.currentTimeMillis();

    Notification notif = new Notification(icono, txtEstat, hora);

      Context context = getApplicationContext();
      CharSequence titol = "News Today";
      CharSequence descripcio = notificacioTxt;

      Intent notIntent = new Intent(context,Main.class);

      PendingIntent contIntent = PendingIntent.getActivity(context, 0, notIntent, 0);

      notif.setLatestEventInfo(context, titol, descripcio, contIntent);

      notif.flags |= Notification.FLAG_AUTO_CANCEL;

      notif.defaults |= Notification.DEFAULT_VIBRATE;
      notif.defaults |= Notification.DEFAULT_LIGHTS;

      notManager.notify(NOTIF_ALERTA_ID, notif);

  }
}
```

Fig. 5.13. Main.java: Notificacions

5.5. Disseny de l'aplicació: Geolocalització

En aquest apartat s'explica com a través de *Google Maps* aquest és capaç de trobar la situació actual de l'usuari i marcar-hi els cinemes més propers a ell.

Primerament cal crear dues classes anomenades *AndroidMapes***,** *MyOverlays* i un arxiu .xml anomenat *androidmapes*.

En aquest últim arxiu és on s'ha d'afegir la *API Key* (Fig. 5.14):

```
<com.google.android.maps.MapView
    android:id="@+id/mapa"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:apiKey="0_RhvKy86JCgvNIKadOb8vdBpZCOWLU5i3fszOw"
    android:clickable="true" />
```

Fig. 5.14. Inserció API Key

Per a obtenir aquesta clau cal seguir els passos següents:

- En el botó "Inici" de l'ordinador seleccionar "Símbol del sistema"
- Introduir la següent comanda per a obtenir la clau MD5: ***C:\Archivos de programa\Java\jdk1.7.0_03\bin> keytool –v –list –alias androiddebugkey –keystore debug.keystore –storepass android –keypass android***

  Per a la versió Java 1.7 o posteriors cal tenir en compte que a la comanda keytool després cal afegir –v ja que sinó tan sols t'indica la clau SHA1 i es necessària la MD5.
- Cal dirigir-se a la pàgina web de Google: *https://developers.google.com/android/maps-api-signup?hl=es*
- Acceptar els termes i condicions, escriure la clau del certificat MD5 en la casella indicada i fer clic al botó "*Generate API Key*".

A continuació cal afegir la següent sentència a l'*Android Manifest* : `<uses-library android:name="com.google.android.maps" />` i comprovar que té els permisos adequats afegint:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_MOCK_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

A continuació es passa a implementar la classe AndroidMapes. Primerament s'ha de configurar el mapa per a que sigui mostrat quan es faci clic al botó de "Localització", afegir els controls de zoom i obtenir la ubicació (*LocationManager*) (Fig. 5.15)

```
mapView = (MapView) findViewById(R.id.mapa);
mapView.setBuiltInZoomControls(true);
mapView.setSatellite(true);
mapController = mapView.getController();
mapController.setZoom(14);
locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0,
        0, new GeoUpdateHandler());
```

Fig. 5.15. Configuració del mapa

A continuació cal crear la classe *MyLocationOverlay* la qual permet visualitzar la ubicació actual de l'usuari i habilitar una brúixola (Fig. 5.16).

```
myLocationOverlay = new MyLocationOverlay(this, mapView);
mapView.getOverlays().add(myLocationOverlay);

myLocationOverlay.runOnFirstFix(new Runnable() {
    public void run() {
        mapView.getController().animateTo(
                myLocationOverlay.getMyLocation());
    }
});
```

Fig. 5.16. MyLocationOverlay

Els mètodes *onResume()* i *onPause()* serveixen per a habilitar o no el mètode *MyLocationOverlay.*

La classe *ItemizedOverlay* consisteix en una llista de *OverlayItems* i s'encarrega de la classificació de punta a punta del mapa dibuixant un marcador per a cada punt.

Per a poder mostrar els cinemes un cop mostrada la ubicació cal crear també dues classes anomenades *JsonConverter* i *Marker* i són necessàries les llibreries de *EZMorph* .

Cal escriure el codi que genera la sol·licitud (get) i captura el resultat, la url introduida fa referència a la web de Google Maps.

Els paràmetres d'aquesta url s'han de modificar. El primer és el valor de sortida i el segon paràmetre és la consulta (*query*) que es passa a Google.

Aquesta classe *JsonConverter* s'encarrega de transformar la cadena *Json* en un objecte, i buscar per les etiquetes *overlays* i *markers*. Cada cop que troba una etiqueta de tipus *marker*, crea un objecte del mateix tipus. Finalment, el mètode retorna una col·lecció de marcadors.

La classe *Marker* és on es guarda tota la informació relacionada amb el nom, direcció, latitud i longitud.

6. Estudi econòmic

En aquest capítol es detallen els costos finals del projecte. Al finalitzar el capítol, es mostra una taula resum amb tots els costos comentats al llarg dels diferents subapartats.

6.1. Costos directes

Correspon als costos produïts per l'estudi previ del projecte, el desenvolupament i disseny de l'aplicació i per últim la redacció de la memòria. A continuació es mostra la següent taula (Taula 6.1.) on es desglossen aquests termes:

Concepte	Hores	Preu/Hora (Euros)	Total (Euros)
Estudi previ del projecte i documentació	70	25	1750
Desenvolupament i disseny de l'aplicació	240	25	6000
Redacció memòria projecte	50	15	750

Taula 6.1. Costos directes

6.2. Amortització

En la realització del projecte s'ha utilitzat un ordinador amb sistema operatiu *Windows 7 x64*, la resta és software de programari lliure per a dur a terme la aplicació. L'amortització és mostra a la taula 6.2. desglossada en tres parts:

- Hardware
- Software
- Dispositiu mòbil.

Concepte	Hores	Total (Euros)
Hardware		
Ordinador (Preu 800€)	-	50
Software		
Windows 7 Professional Edition x64 (Preu 145€)	-	10
Dispositiu mòbil		
Telèfon sistema operatiu Android	-	50

Taula 6.2. Amortització

6.3. Costos indirectes

Correspon a les despeses pel consum d'energia elèctrica, lloguer, telèfon i ADSL amb un cost total de 300 €.

6.4. Taula resum dels costos

Concepte	Preu (Euros)
Costos directes	8500
Amortització – Hardware	50
Amortització – Software	10
Amortització – Dispositiu mòbil	50
Costos indirectes	300
TOTAL	8910

Taula 6.3. Taula resum

6.5. Preu aplicació

L'aplicació és *open source* i es distribueix de forma gratuïta mitjançant *Google Play* antigament anomenat *Android Market*. Per tant no té cap cost i l'aplicació pot ser copiada i distribuïda lliurement entre els diferents usuaris.

6.6. Benefici econòmic

Aquest projecte està pensat per a familiaritzar-se amb l'entorn Android i no tant com a benefici econòmic.

Un cop posada l'aplicació a *Google Play* el programador pot aconseguir un cert "nom"; aquesta plataforma li pot servir perquè les empreses puguin arribar a contactar amb ell per fer projectes amb contrapartida econòmica. Per últim, dir que es podria afegir publicitat per a obtenir un benefici però aquest punt queda com a possible millora per a una futura implementació.

7. Conclusions

Aquest projecte té com a principal objectiu adquirir un aprenentatge de tot el que engloba el món d'Android.

S'ha partit de l'estructura del sistema operatiu, que és i com funciona, i a partir d'aquest punt s'ha desenvolupat una aplicació amb diverses funcionalitats.

Per tal d'arribar a assolir el propòsit inicial cal conèixer perfectament quines són les parts que el formen.

Per tant, cal tenir molt clar la base de la qual es parteix per a poder avançar amb el projecte, sense aquest esforç previ és impossible o és més difícil adquirir els coneixements adequats per a dur a terme una determinada aplicació igual o similar a la que s'ha desenvolupat en aquest projecte.

El desenvolupament de l'aplicació ha permès aprofundir més en el llenguatge Java i XML i conèixer les noves tecnologies que estan emergent arrel de l'aparició dels nous telèfons mòbils, *smartphones,* que tant triomfen actualment.

El projecte té un temps acotat i evidentment s'ha hagut de programar l'aplicació amb un cert límit d'hores la qual cosa té punts a favor i punts en contra.

L'avantatge d'aquest fet és que ajuda a planificar-se el temps molt millor, ja que es pateix el perill sinó, de no poder realitzar tot el previst.

El desavantatge és que al començar a descobrir tot el que engloba Android a mida que s'avança amb el projecte, es descobreixen nous aspectes que poden aportar o afegir moltes més funcionalitats a l'aplicació i que malauradament han de quedar per una possible futura implementació.

Per últim ressaltar que, el que es vol assolir mitjançant aquest projecte no és el fet de saber fer exclusivament l'aplicació "*News Today*" sinó aprendre un nou sistema operatiu que engloba tot un ventall de possibilitats i que permet fer infinites aplicacions totes elles diferents entre si.

7.1. Possibles millores i/o ampliacions

En aquest projecte s'han dut a terme totes les funcionalitats o tasques bàsiques que es pretenien però sempre hi ha certs aspectes que poden millorar-se o ampliar-se. A continuació s'especifiquen una sèrie de punts en els quals es detalla les possibles millores o ampliacions que es poden fer.

- Les notícies que es mostren, són extretes d'un determinat diari. Com a millora caldria fer una altra activitat (*Activity*) que mitjançant una llista donés la opció de poder escollir entre diferents diaris.

- Dins de la mateixa activitat creada per a escollir el tipus de diari que es vol, tenir una opció per poder triar la categoria (esports, política, cinema...) i que guardés la categoria escollida en un fitxer de forma que al tornar a entrar a l'aplicació aquesta no carregui totes les noticies d'actualitat sinó, només aquella categoria escollida com a preferida.

- En el botó de "Localització" dins del menú, afegir al mapa altres dades d'interès per a l'usuari per a que mostri no tan sols els cinemes.

- Tenir la opció de, mitjançant una base de dades, introduir el e-mail de l'usuari per a que aquest rebi les noticies que més l'interessin al seu correu electrònic.

- Afegir imatges o vídeos a les noticies de forma que no només es pugui llegir el contingut sinó que vagi acompanyat d'algun tipus de multimèdia .

- Incloure la possibilitat de visualitzar les noticies en més d'un idioma.

- Millorar la implementació de *layouts* per a que l'aplicació pugui ser vista en altres dispositius mòbils com per exemple una tableta.

8. Referències

[1] http://developer.android.com/index.html **(darrera consulta: 09/06/2012)**

Pàgina web oficial per a desenvolupadors d'aplicacions d'Android. Inclou l'*Android SDK* i la documentació per a desenvolupadors i dissenyadors.

[2] http://www.androidsis.com/programacion-android-ii-hola-mundo/ **(darrera consulta: 09/06/2012)**

Pàgina web que mostra com crear un projecte Android, definint les parts més importants i creant l'aplicació "Hola mundo".

[3] http://blog.findemor.es/ **(darrera consulta: 09/06/2012)**

Blog amb diferents guies de programació Android: Creació de *Layouts*, *ListViews*, notificacions...

[4] http://www.ibm.com/developerworks/ **(darrera consulta: 09/06/2012)**

Pàgina web de IBM (International Business Machines) per a desenvolupadors on es pot trobar diferents punts sobre la programació Java utilitzada per a programar en Android.

[5] http://www.elbauldelprogramador.com/programacion/programacion-android-interfaz-grafica_28/ **(darrera consulta: 09/06/2012)**

Pàgina web que mostra diferents apartats sobre la programació Android, en concret aquest enllaç fa referència a la interfície gràfica.

[6] http://android.javielinux.com/locationgps.php **(darrera consulta: 09/06/2012)**

Web que fa referència a la localització GPS mitjançant l'explicació d'un exemple.

[7] http://www.sgoliver.net/blog/?p=1766 **(darrera consulta: 09/06/2012)**

Blog que explica com crear menús dins d'una aplicació Android així com altres funcionalitats específiques d'aquest sistema operatiu.

[8] http://stackoverflow.com/ **(darrera consulta: 09/06/2012)**

Pàgina web de preguntes i respostes semblant a un fòrum per a programadors professionals i aficionats.

[9] Burnette; Programación Android; Edit. Anaya, 2011

[10] Diaz de Ugarte Lluís, Aplicació per a dispositius mòbils en "Android": Geoshows, "Projecte Final de Carrera", Enginyeria Tècnica de Telecomunicacions especialitat Telemàtica, Escola Universitària Politècnica de Mataró,Primavera 2011.