Android API Demos 2.3 学习笔记

作者	snowdream
时间	2011 年 08 月 16 日

谨以此书献给所有和我一样热爱Android的Coder!

前言

由于 Android 从诞生到现在并不是很久，与之有关的学习资料也不是很多。因此对于学习 Android 的人来说，Android SDK 附带的 API Demos 无疑是最好的学习资料。

本书作者试图通过自身学习实践，不断总结，记录笔记，来熟悉和掌握 Android 平台开发相关基础知识，并为后来者学习 Android 提供参考。

作　者

2011 年 8 月

目录

第 1 章　导言

1.1 搭建 Android 开发环境

本书主要介绍在 Ubuntu 11.04 + JDK 7 环境下，如何搭建 Android 开发环境。如果您需要在 Windows 下搭建 Android 开发环境，请参考网络相关内容。

1.1.1 搭建 JDK 开发环境

第一步：下载 JDK 7 压缩包

```
wget -c http://download.oracle.com/otn-pub/java/jdk/7/jdk-7-linux-i586.tar.gz
```

(注：如果下载不下来，建议使用迅雷下载，然后拷贝到 Linux 系统上。)

第二步：解压安装

```
sudo tar zxvf ./jdk-7-linux-i586.tar.gz  -C /usr/lib/jvm
cd /usr/lib/jvm
sudo mv jdk1.7.0/ java-7-sun
```

第三步：修改环境变量

```
vim ~/.bashrc
```

在该文件末尾添加以下内容：

```
export JAVA_HOME=/usr/lib/jvm/java-7-sun
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH
```

保存退出，输入以下命令使之立即生效。

```
source  ~/.bashrc
```

第四步：配置默认 JDK 版本

由于 Ubuntu 中可能会有默认的 JDK，如 OpenJDK。为了使默认使用的是我们安装的 JDK 7，还要进行以下操作。

执行代码:

```
sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/java-7-sun/bin/java 300
sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/java-7-sun/bin/javac 300
```

执行代码

```
sudo update-alternatives --config java
```

系统会列出各种 JDK 版本，如下所示：

```
snowdream@snowdream:~$ sudo update-alternatives --config java
有 3 个候选项可用于替换 java (提供 /usr/bin/java)。
  选择       路径                                  优先级  状态
------------------------------------------------------------
* 0            /usr/lib/jvm/java-6-openjdk/jre/bin/java   1061      自动模式
  1            /usr/lib/jvm/java-6-openjdk/jre/bin/java   1061      手动模式
  2            /usr/lib/jvm/java-6-sun/jre/bin/java       63        手动模式
  3            /usr/lib/jvm/java-7-sun/bin/java           300       手动模式

要维持当前值[*]请按回车键，或者键入选择的编号：3
update-alternatives: 使用 /usr/lib/jvm/java-7-sun/bin/java 来提供 /usr/bin/java (java)，于手动模式 中。
```

第四步：测试

在终端中输入 java -version，测试 JDK 环境是否安装成功。

```
snowdream@snowdream:~$ java -version
java version "1.7.0"
Java(TM) SE Runtime Environment (build 1.7.0-b147)
Java HotSpot(TM) Server VM (build 21.0-b17, mixed mode)
```

1.1.2 下载并安装 Eclipse

第一步：下载并安装 Eclipse （官方网站下载：http://www.eclipse.org/downloads/）

根据实际情况，推荐安装以下版本：

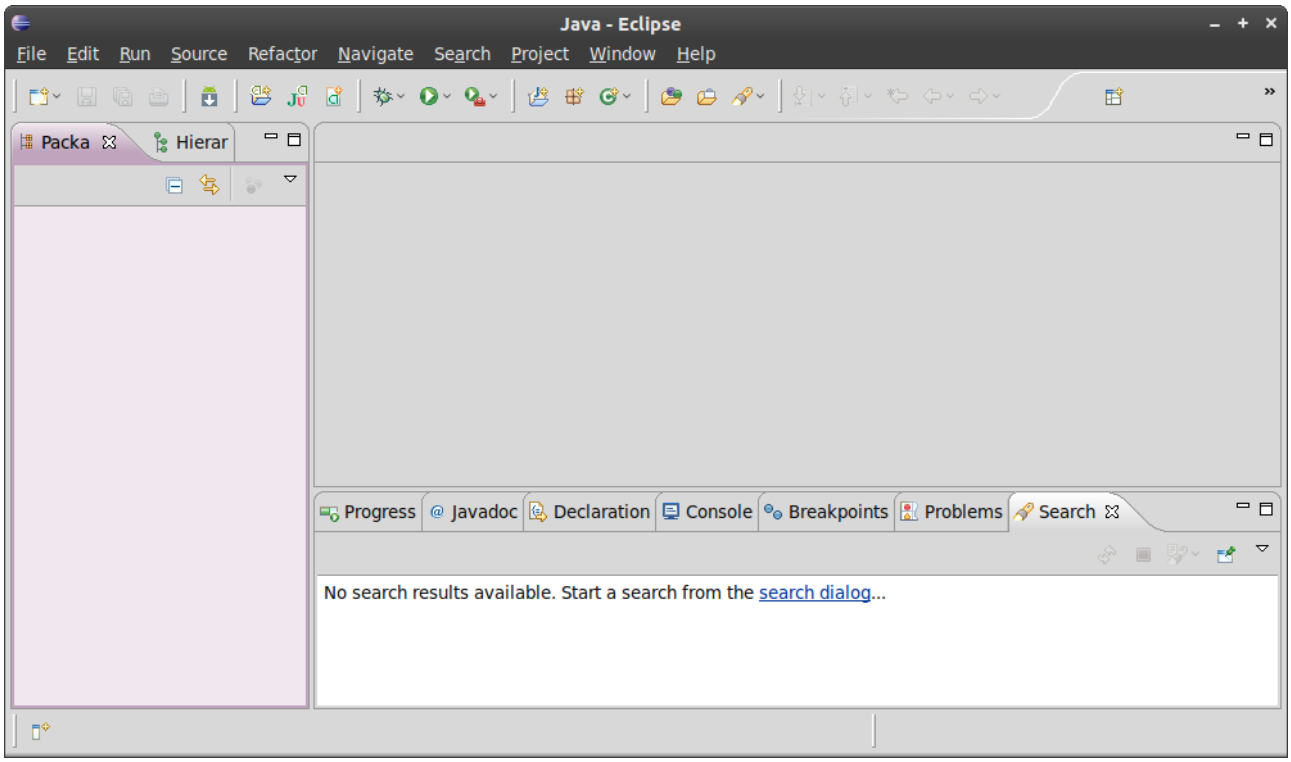
Eclipse IDE for Java and Report Developers, 250 MB

（Ubuntu11.04 32 位系统请直接通过以下命令下载并安装 Eclipse）

```
wget -c
http://mirror.bjtu.edu.cn/eclipse/technology/epp/downloads/release/indigo/R/eclipse-reporting-indigo-linux-gtk.tar.gz
tar zxvf eclipse-reporting-indigo-linux-gtk.tar.gz
```

第二步：测试

进入 Eclispe 安装目录，双击 Eclipse 可执行程序，如果依次出现以下画面，则 Eclipse 安装成功。



1.1.3 下载 Android SDK 以及搭建 Android 开发环境

第一步：下载并安装 Android SDK

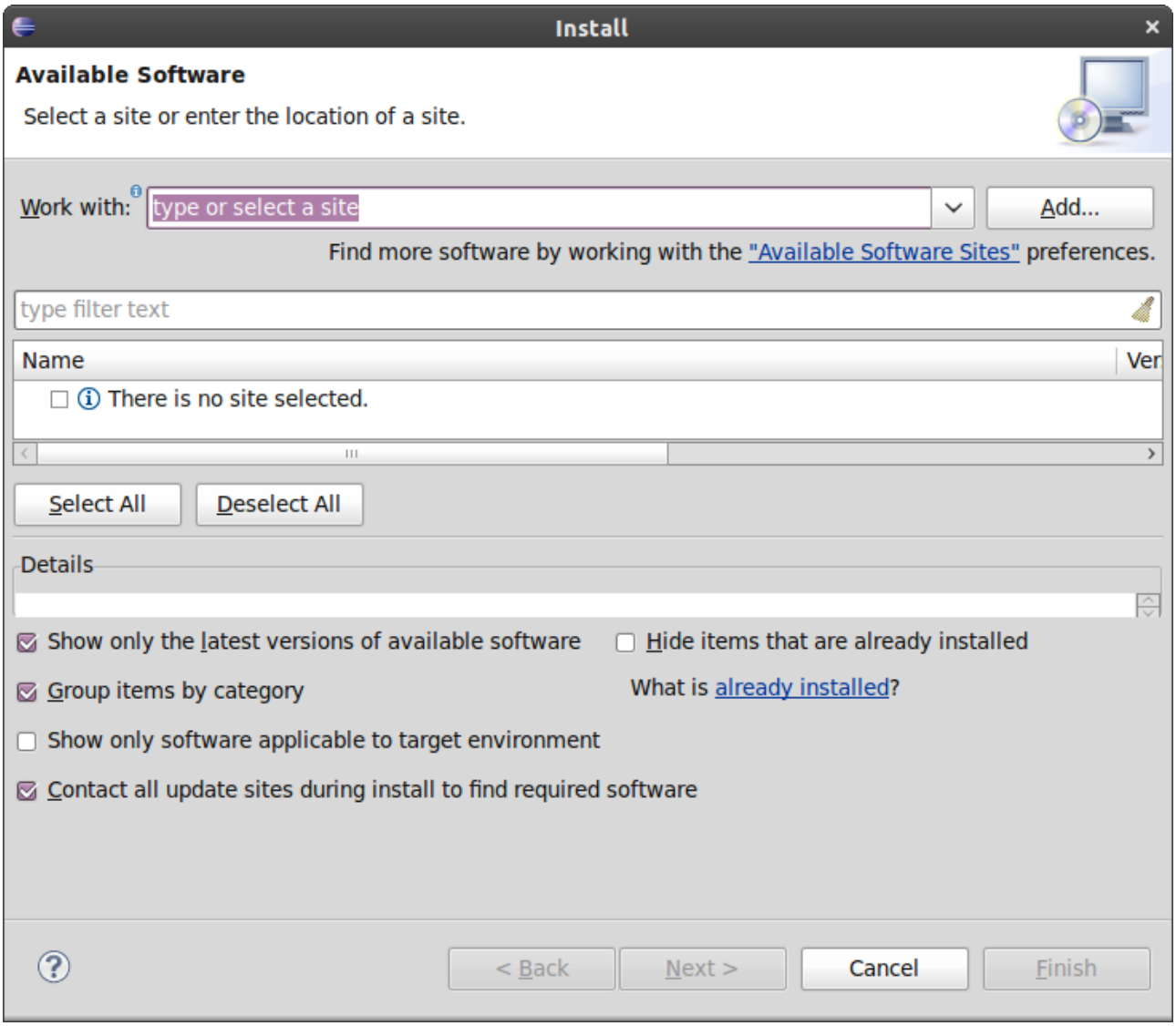
```
wget -c http://dl.google.com/android/android-sdk_r12-linux_x86.tgz

tar zxvf android-sdk_r12-linux_x86.tgz
```

第二步：在线安装 Eclipse 插件 ADT

启动 Eclipse，然后依次选择菜单：Help > Install New Software....

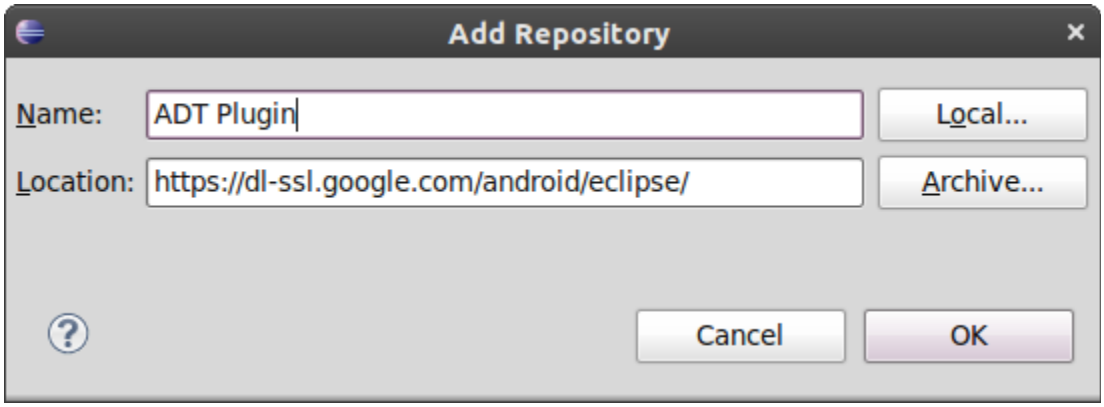
在窗口右上角点击 Add 按钮



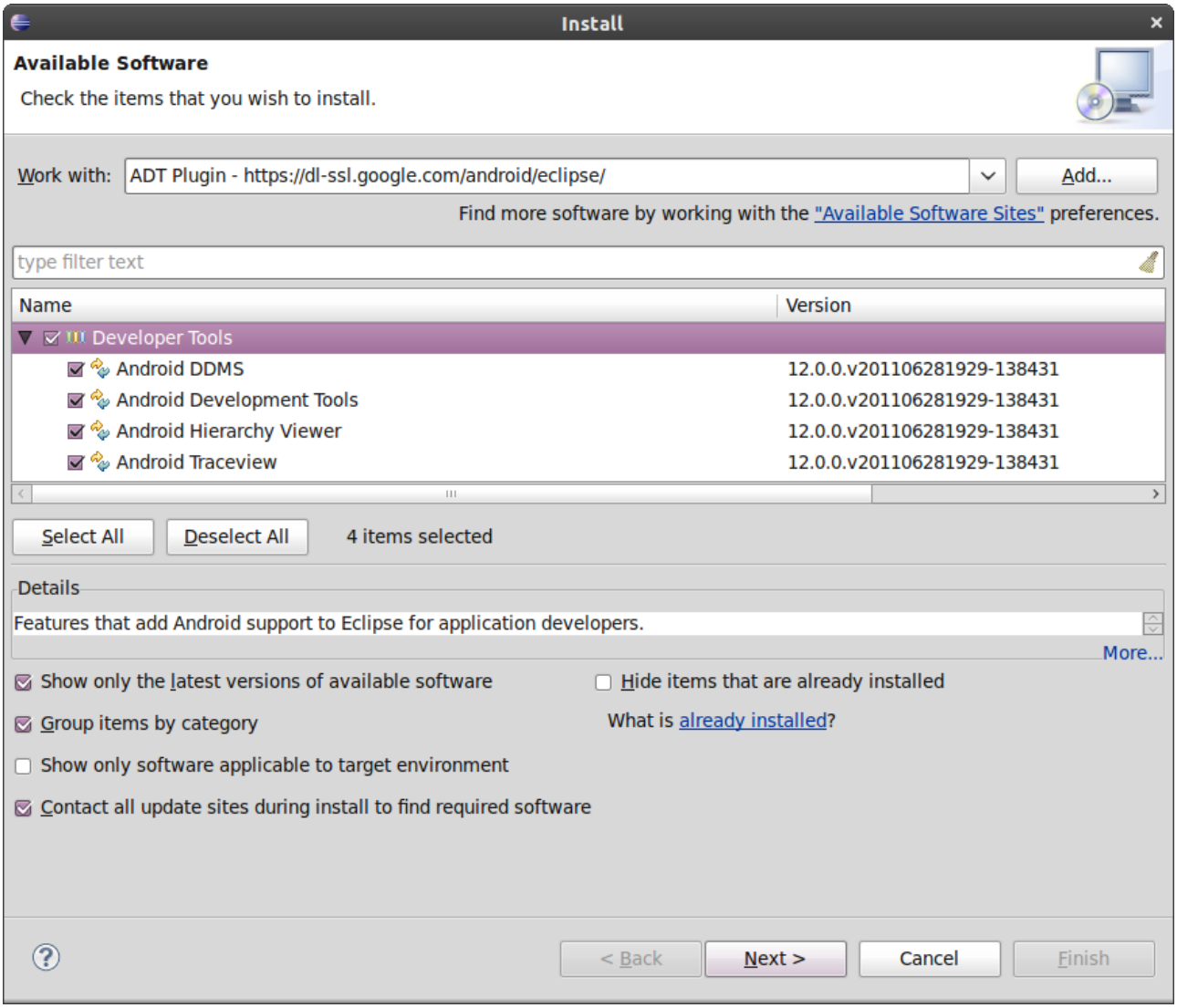
在添加源地址窗口中，在 Name 字段后面填写"ADT Plugin"，而在下面的 Location 字段后面填写以下地址：

```
https://dl-ssl.google.com/android/eclipse/
```

然后点击 OK。



在可选软件窗口，点击 Select All 全部安装，然后点击 Next。



在下一个窗口，你会看到一系列即将被下载的工具，点击 Next。

阅读并且接受软件协议，然后点击 Finish。

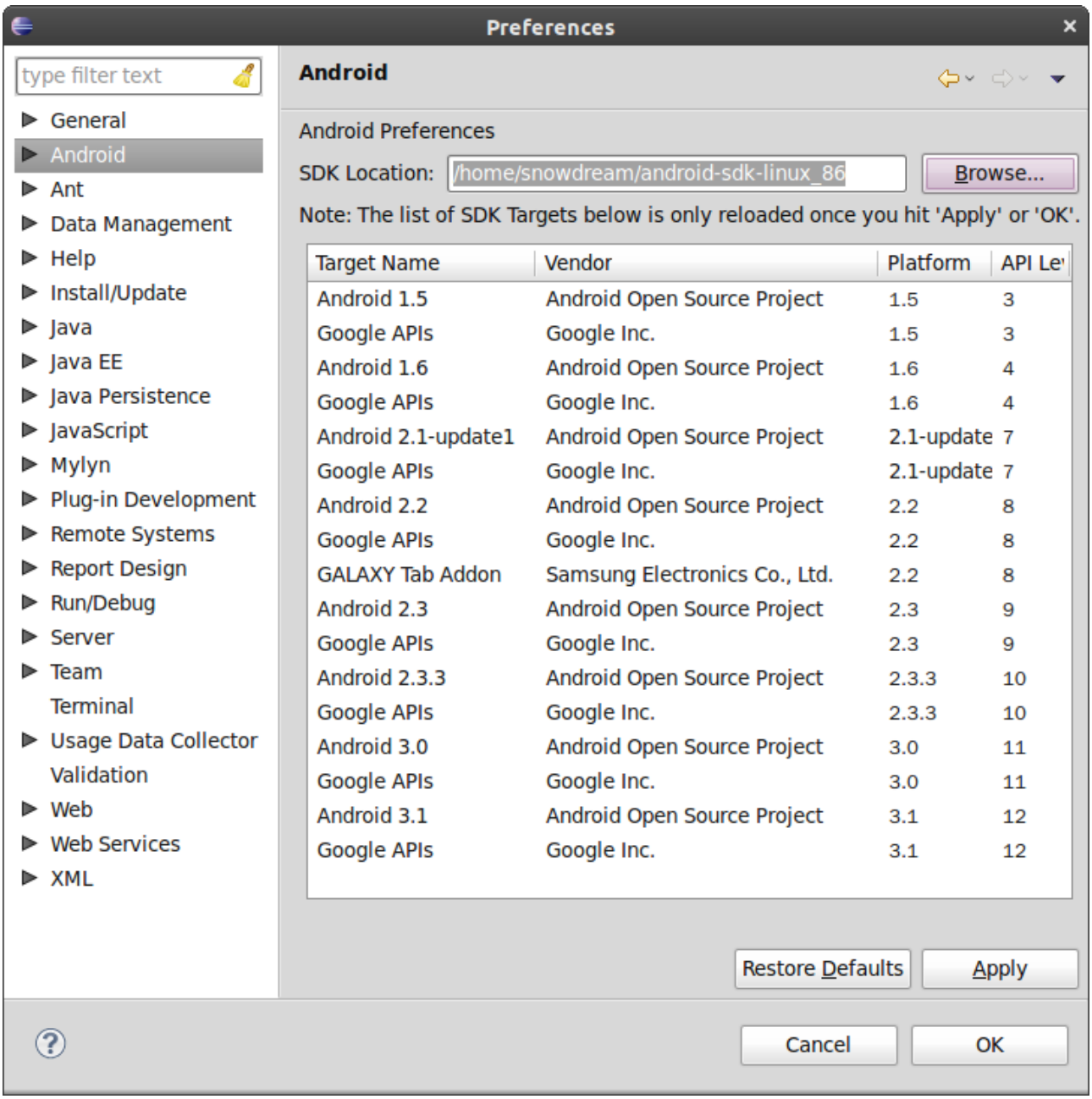
第三步：配置 Eclipse 插件 ADT

启动 Eclipse，然后依次选择菜单：Window > Preferences...

在左边的面板上选择 Android 选项，如下所示：

点击 Browse... 并且定位到你的 Android SDK 目录,例如 /home/snowdream/android-sdk-linux_86

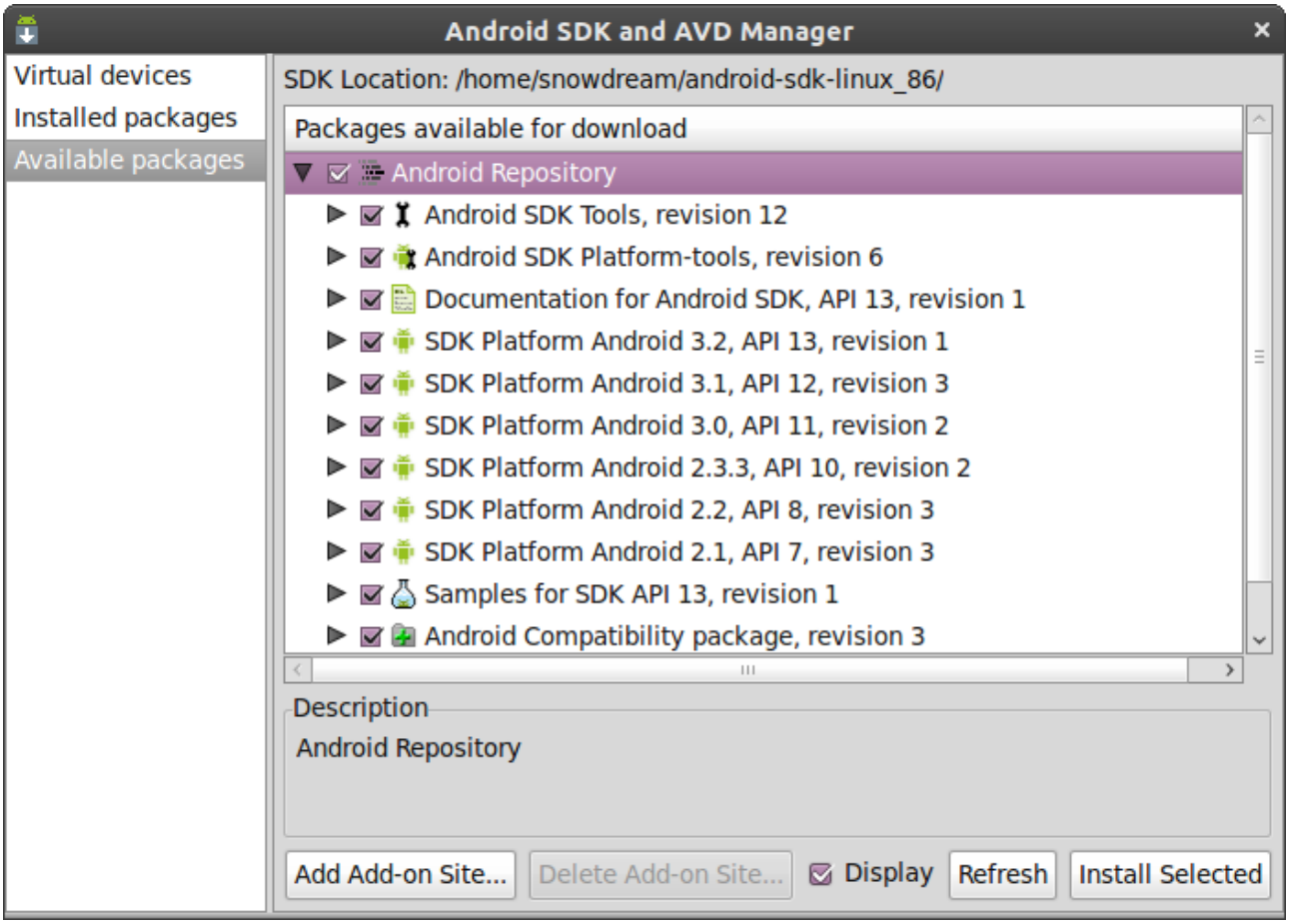
先点击 Apply, 然后点击 OK。



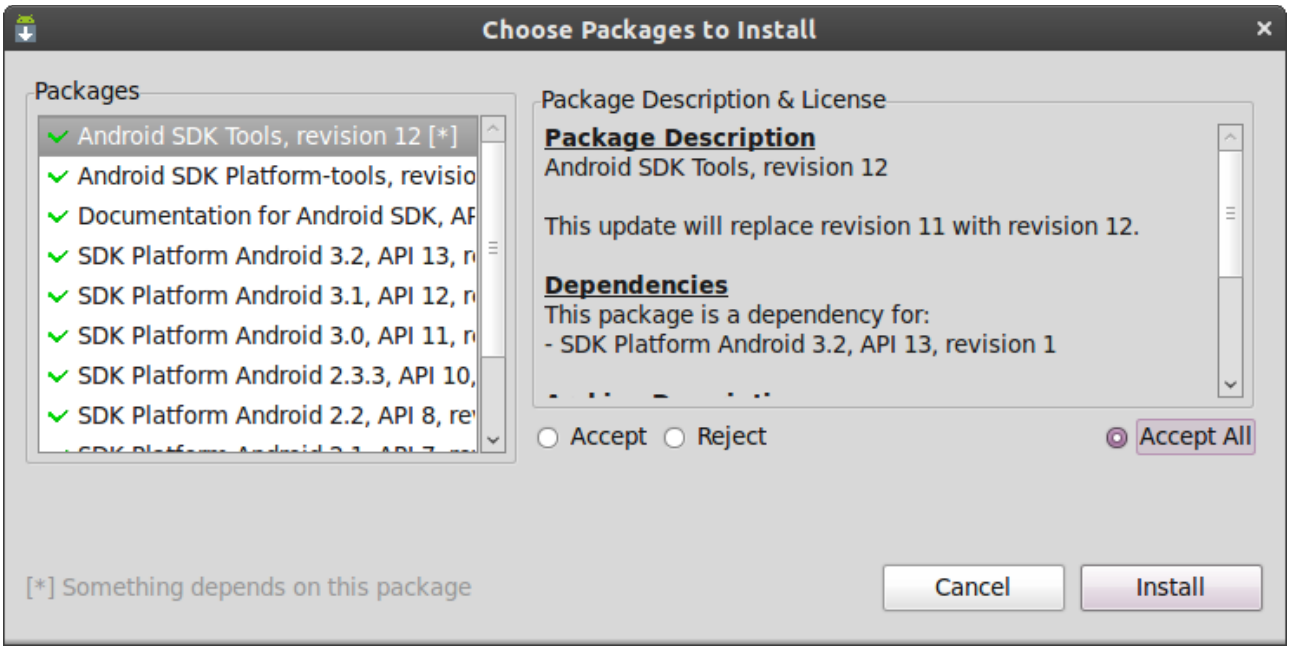
第四步：添加 Android SDK 组件

启动 Eclipse，然后依次选择菜单：Window > Android SDK and AVD Manager

在左侧面板上选择 Available Packages，这将会在右侧显示 SDK 源中所有可以进行下载安装的组件。



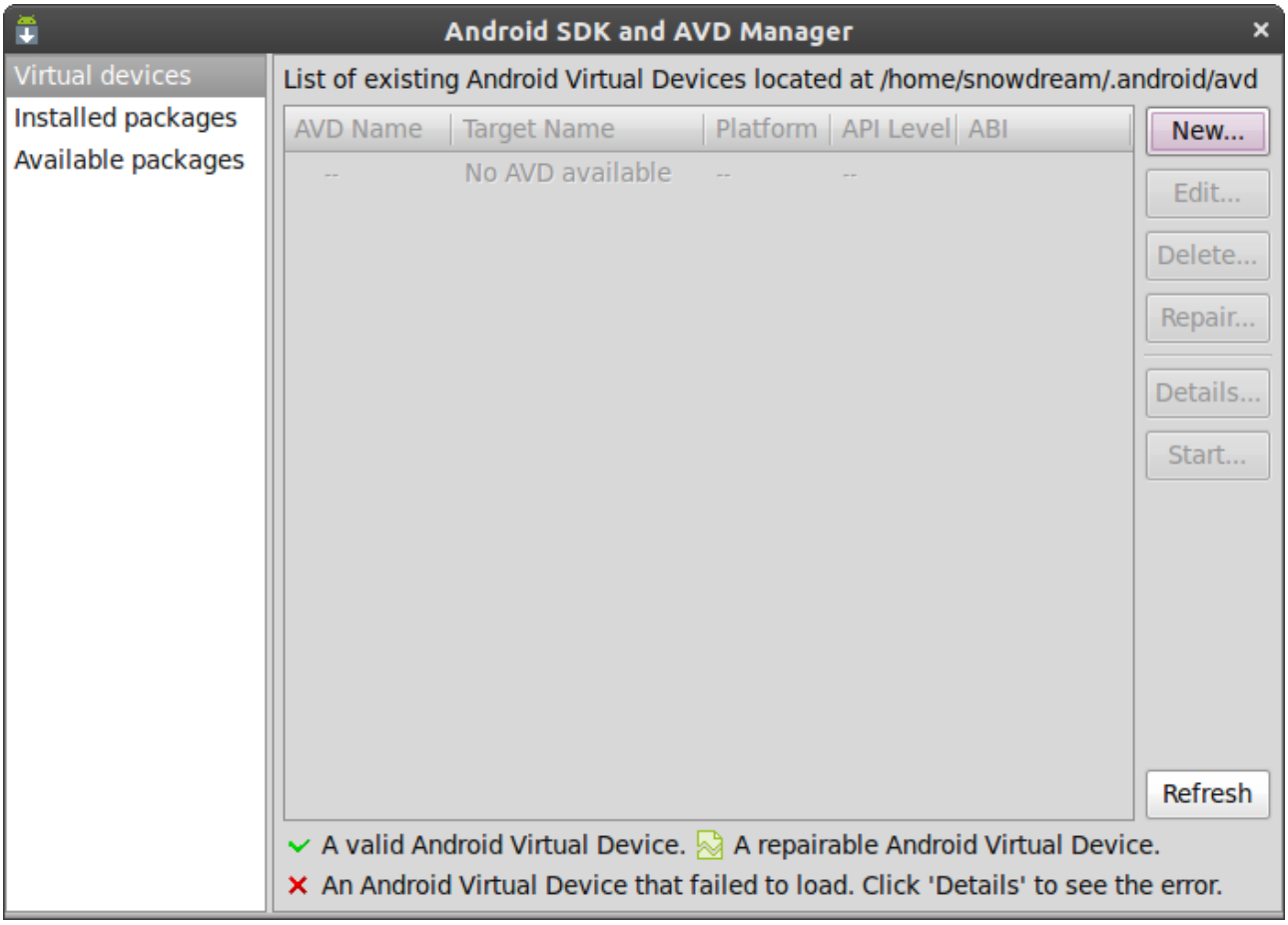
根据需求，选择你所需要安装的组件，然后点击 Install Selected。在接下来弹出的阅读协议窗口中，选择 Accept All，然后点击 Install。这些组件将会安装到您的 Android SDK 安装目录。



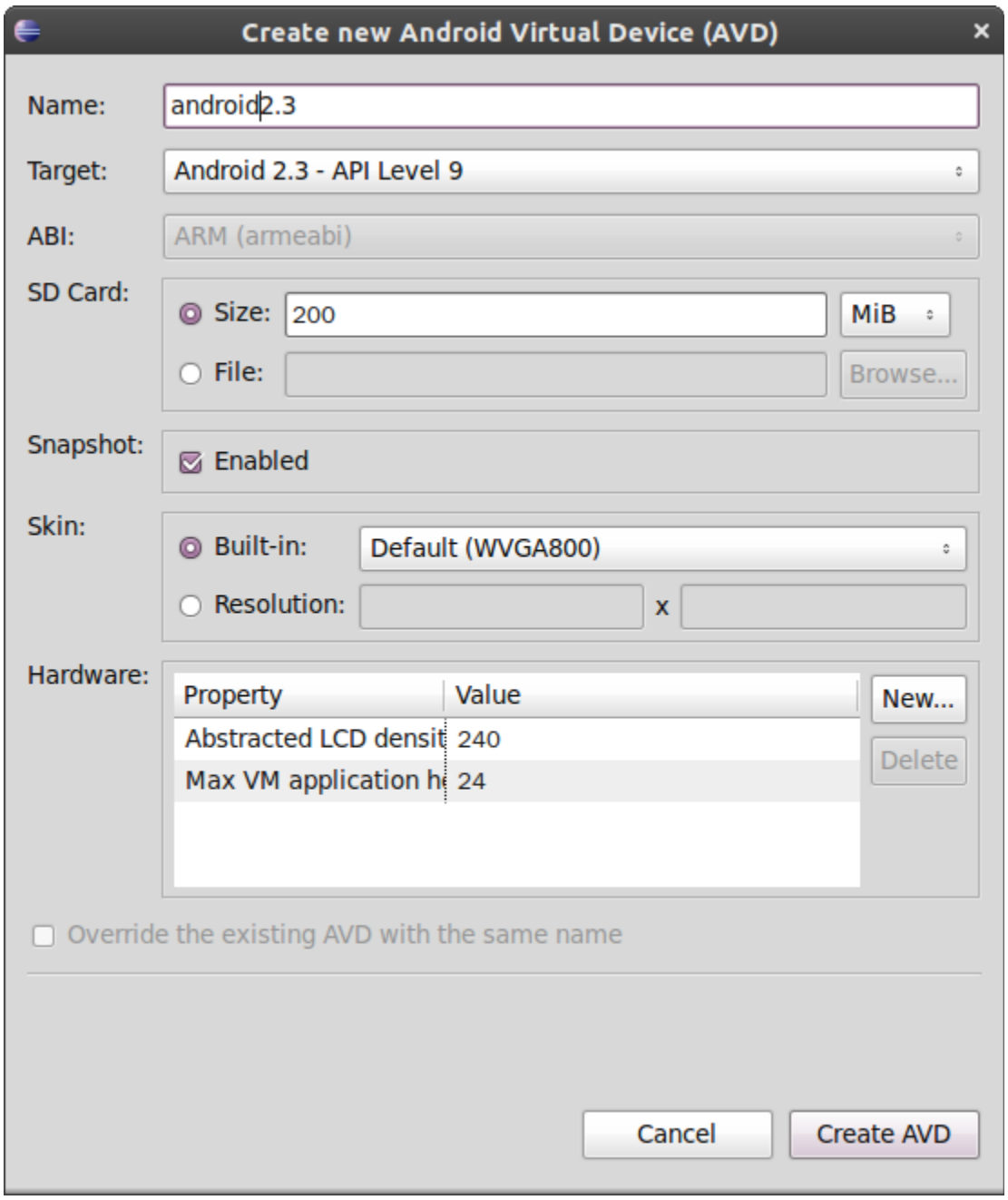
注：这一步可能需要花费数小时，具体时间和您的网络环境密切相关，请耐心等候。安装完成后，根据提示，需要重新启动 Eclipse 才能应用更新。

1.1.4 创建 Android 虚拟设备 AVD

启动 Eclipse，然后依次选择菜单：Window > Android SDK and AVD Manager



在左侧面板上选择 Virtual Devices，然后在右上角点击 New... 新建 AVD 设备，如下所示：



注明：

Name：填写 AVD 名称，例如 android2.3

Target：根据常用的 SDK 版本进行选择，例如，Android 2.3-API Level 9

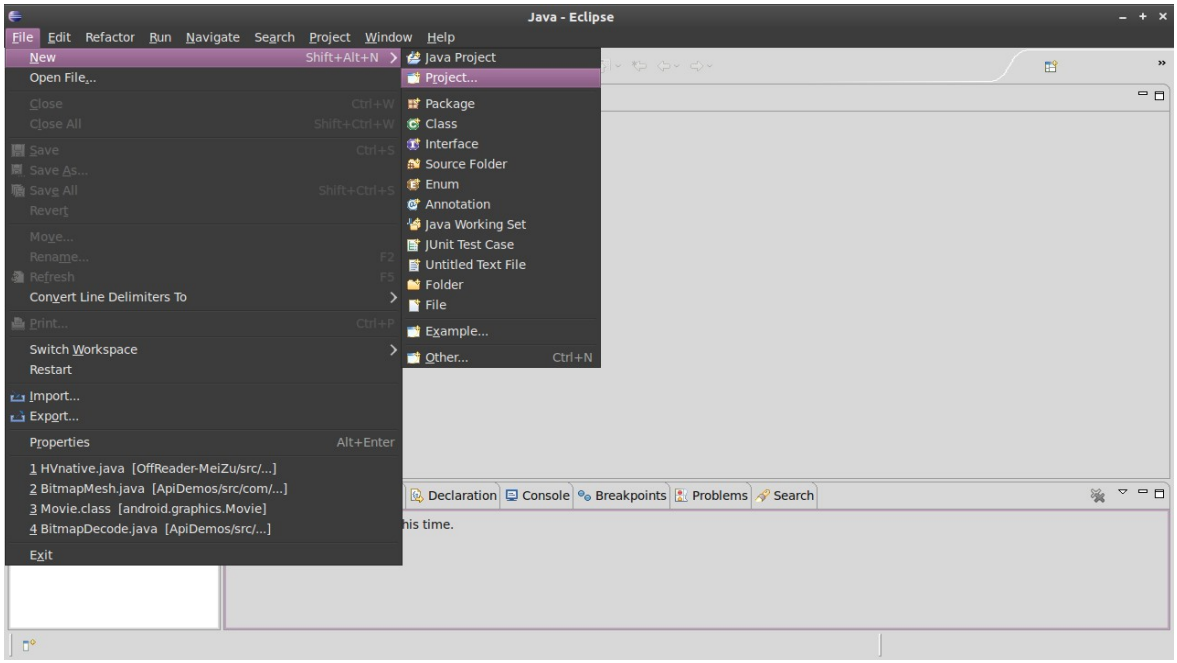
Size：虚拟 sd 卡容量大小，根据实际需求设置，例如 200MiB

Built-in：选择 AVD 的皮肤，这里保持默认选项

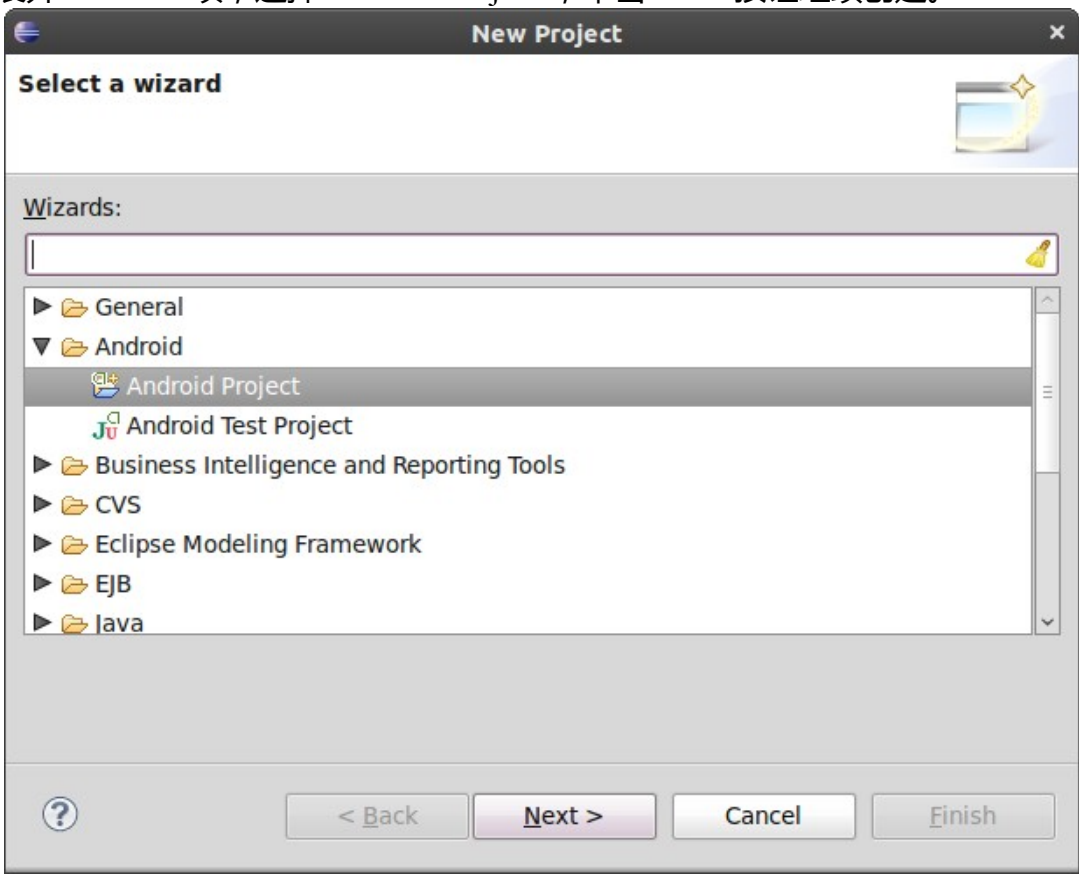
1.2 创建第一个 Android 项目（Hello World!）

第一步：根据新建项目向导创建项目

启动 Eclipse, 选择"File"--"New"--"Project",打开新建项目向导。

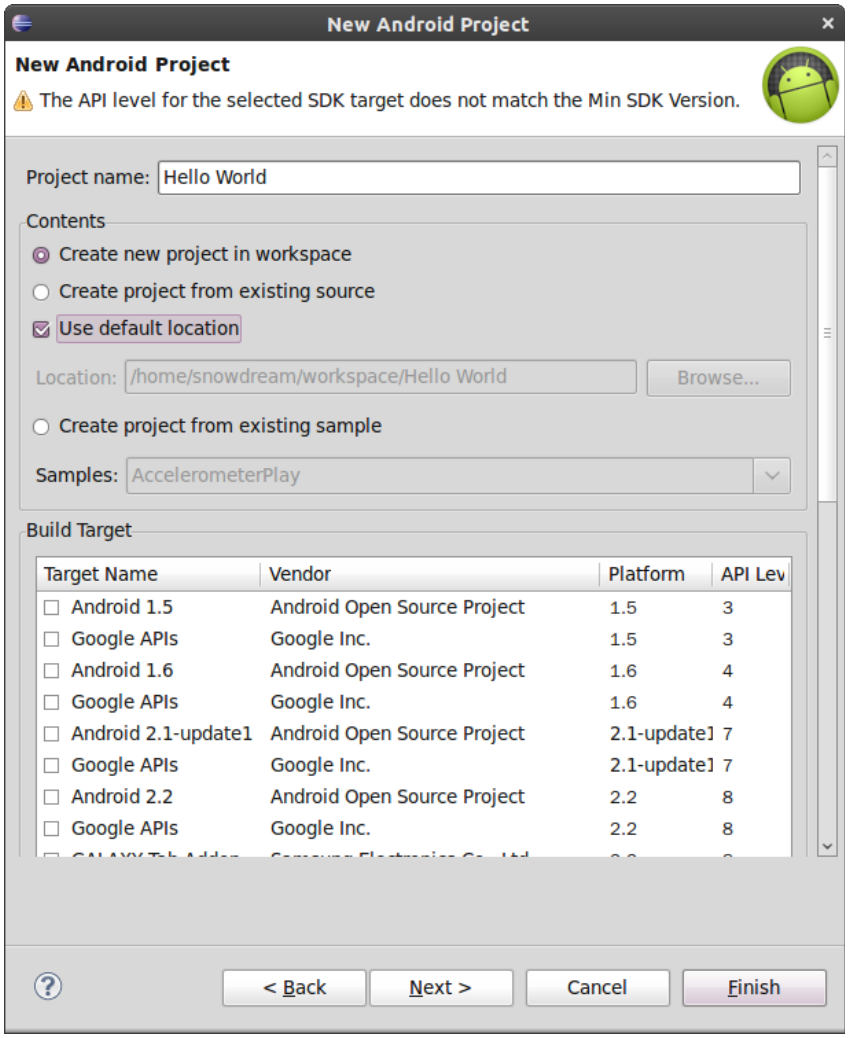


展开"Android"项，选择"Android Project"，单击"Next"按钮继续创建。



在"Project name:"字段后填写项目名称"Hello World"。

注：默认在 Eclipse 工作目录下以项目名称创建一个新文件夹作为该项目的主文件夹，如果您需要自定义项目主文件夹，需要先点击掉"Use default location"选项，然后在下面的"location"字段后面填写自定义路径。



把右边的滚动条往下拉，在"Build Target"下面选择您编译需要使用的 SDK 版本，这里我们选择版本"Android 2.3"。其他字段填写说明如下：

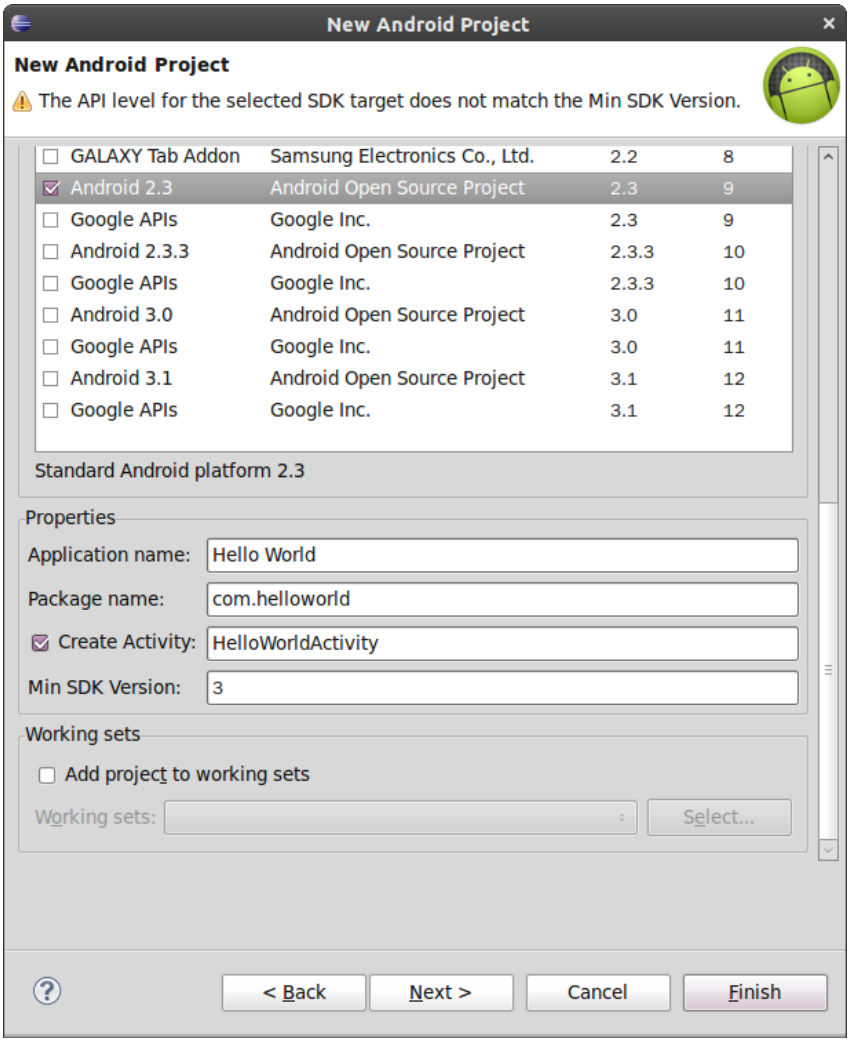
Application name： Hello World　　//程序名称

Package name: com.helloworld　　//软件包名称

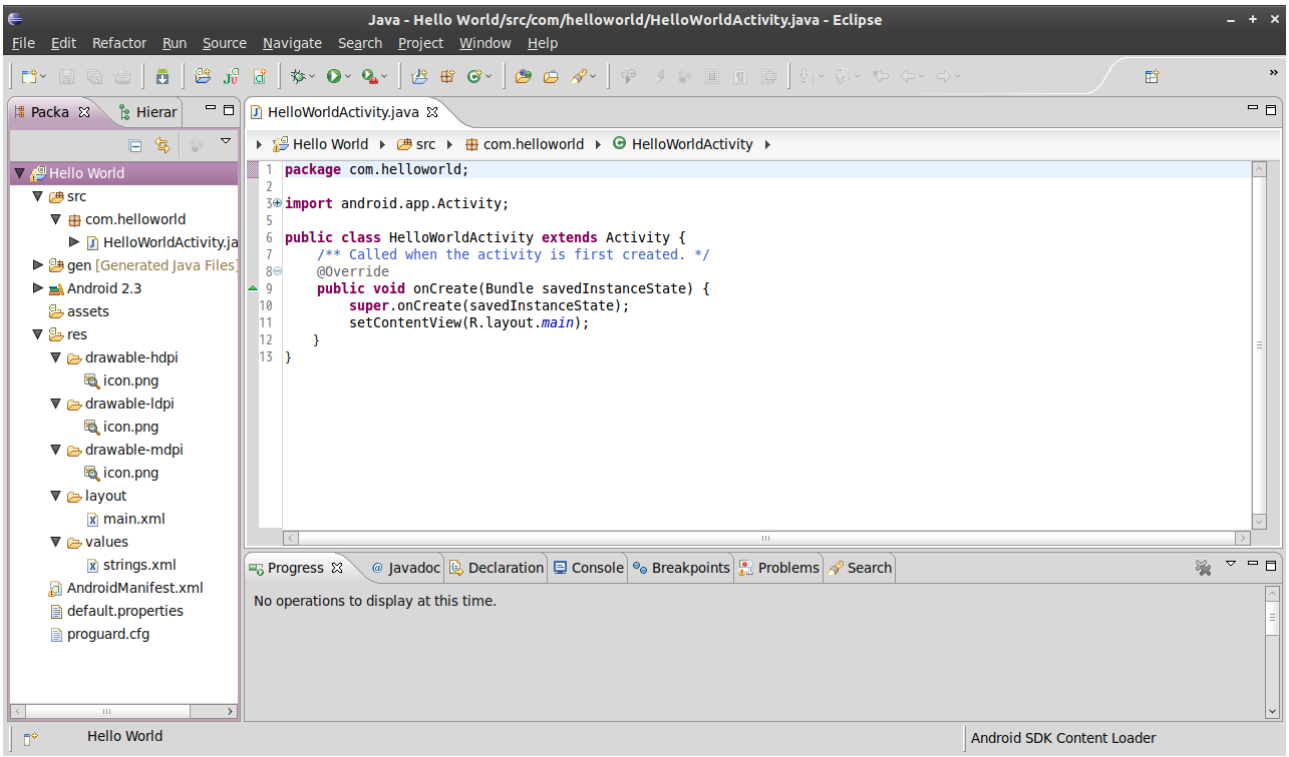
Create Activity: HelloWorldActivity　//Android 项目主 Activity 名称

Min SDK Version: 3　　//向下兼容的最低 Android 版本，对应" Build Target"下面的" API Level"

如下图所示：

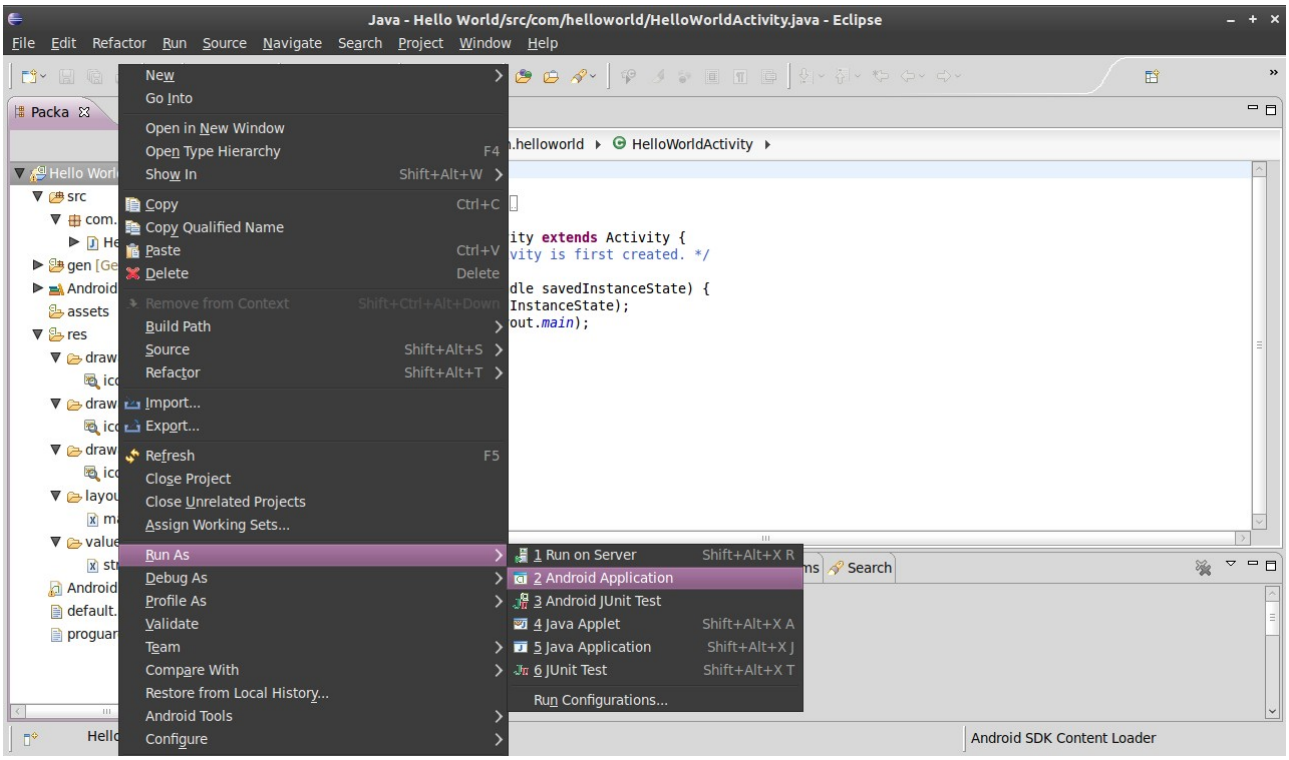


最后单击"Finish"按钮，项目创建完成。



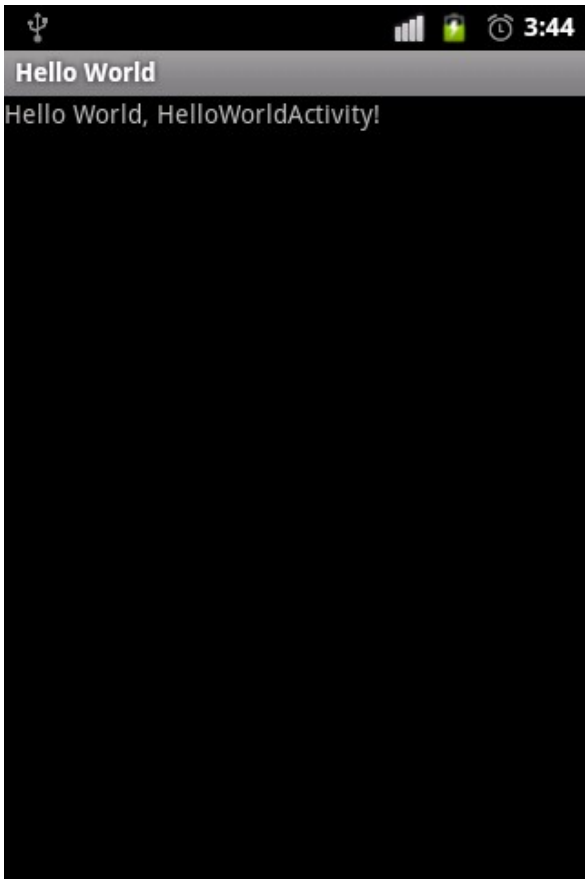
第二步：运行 Android 项目程序

在 Eclipse 左侧"Package Explorer"窗口，右键点击刚刚创建好的"Hello World"项目文件夹，在功能菜单上选择"Run As--Android Application"功能。如下图所示：



如果已经创建 AVD 虚拟设备，则会自动启动模拟器。否则，请参考"1.1.4　创建 Android 虚拟设备 AVD"章节先创建一个 AVD 虚拟设备。如果您拥有 Android 手机，也可以不用创建该设备，直接使用手机运行调试 Android 程序。

运行效果：



1.3 Android 应用程序架构

Android 应用程序以可以分为下三种类型：

1、前端 Activity（Foreground Activities）；

通俗一点讲 Activity 可以理解为一个界面容器，里面装着各种各样的 UI 组件。例如，上面例子中 “Hello World” 显示界面。

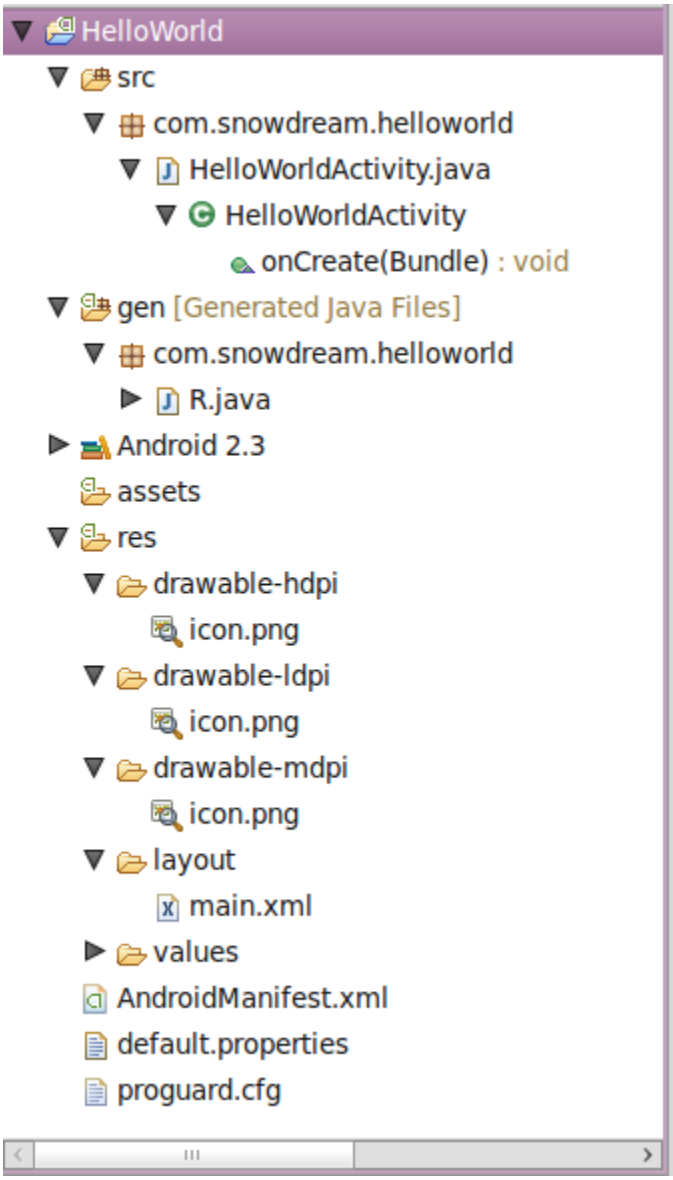
2、后台服务（Background Services）；

系统服务（System Service）、系统 Broadcast（广播信息）与 Receiver（广播信息）接收器）等都属于后台服务。它们在后台运行时，并不会对于前端 Activity 的显示造成影响。例如，音乐播放放到后台时，并不影响其他界面操作响应。

3、间隔执行 Activity（Intermittent Activities）；

包括进程（Threading）、Notification Manager 等都属于这一类。

这里我们以 Hello World 这个简单的应用程序为例，简述一下 Android 应用程序的架构。如右图所示：



src/　java 源代码存放目录

gen/　自动生成目录

gen 目录中存放所有由 Android 开发工具自动生成的文件。目录中最重要的就是 R.java 文件。 这个文件由 Android 开发工具自动产生的。Android 开发工具会自动根据你放入 res 目录的 xml 界面文件、图标与常量，同步更新修改 R.java 文件。正因为 R.java 文件是由开发工具自动生成的，所以我们应避免手工修改 R.java。R.java 在应用中起到了字典的作用，它包含了界面、图标、常量等各种资源的 id，通过 R.java，应用可以很方便地找到对应资源。另外编绎器也会检查 R.java 列表中的资源是否被使用到，没有被使用到的资源不会编绎进软件中，这样可以减少应用在手机占用的空间。

res/ 资源(Resource)目录

在这个目录中我们可以存放应用使用到的各种资源，如 xml 界面文件，图标或常量

res/drawable 专门存放图标文件

res/layout 专门存放 xml 界面文件，xml 界面文件和 HTML 文件一样，主要用于用户界面显示

res/values 专门存放应用使用到的各种常量，作用和 struts 中的国际化资源文件一样。

AndroidManifest.xml 功能清单文件

这个文件列出了应用程序所提供的功能，在这个文件中，你可以指定应用程序使用到的服务(如电话服务、互联网服务、短信服务、GPS 服务等等)。另外当你新添加一个 Activity 的时候，也需要在这个文件中进行相应配置，只有配置好后，才能调用此 Activity。

default.properties 系统默认信息，一般是不需要修改此文件

proguard.cfg proguard 代码混淆工具配置文件，可能需要修改修改此文件

从 SDK2.3 开始我们可以看到在 android-sdk-windows\tools\下面多了一个 proguard 文件夹。proguard 是一个 java 代码混淆的工具，通过 proguard，别人即使反编译你的 apk 包，也只会看到一些让人很难看懂的代码，从而达到保护代码的作用。

第 2 章　Text

2.1　Linkify

Android 实现 TextView 中文本链接的方式有很多种。

总结起来大概有 4 种：

1、通过 android:autoLink 属性来实现对 TextView 中文本相应类型的链接进行自动识别。

例如：android:autoLink = all 可以自动识别 TextView 文本中的网络地址，邮件地址，电话号码，地图位置等，并进行链接。

android:autoLink 所有支持的链接属性取值如下：

常量	值	描述
none	0x00	不进行自动识别 (默认).
web	0x01	自动识别网络地址
email	0x02	自动识别邮件地址
phone	0x04	自动识别电话号码
map	0x08	自动识别地图位置
all	0x0f	自动识别以上四种链接属性 (相当于 web|email|phone|map).

注：可以通过“|”符号连接多个属性值来支持多种类型的链接自动识别。例如，android:autoLink = web|email|phone 支持对网络地址，邮件地址，电话号码的自动识别，并进行链接。

这是在 XML 文件中进行属性设置来识别链接的方式，还有一种在 Java 代码中进行属性设置的方式，同样可以实现类似功能。例如 TextView 对象 mTextView1，我们可以通过 mTextView1.setAutoLinkMask(int mask)来实现对 TextView 中文本相应类型的链接进行自动识别。其中 mask 所有取值如下：

常量		
int	ALL	自动识别邮件地址，网络地址，地图位置和电话号码
int	EMAIL_ADDRESSES	自动识别邮件地址
int	MAP_ADDRESSES	自动识别地图位置
int	PHONE_NUMBERS	自动识别电话号码

int	WEB_URLS	自动识别网络地址

注：使用时请在常量前面加上 Linkify.字样，例如：mTextView1.setAutoLinkMask(Linkify.ALL)

2、将含有 HTML 链接标记的文本写在 Android 资源文件中，如 string.xml，然后在 Java 代码中直接引用。

3、通过 Html 类的 fromHtml（String source）方法来对含有 HTML 链接标记的文本进行格式化处理。

4、通过 Spannable 或继承它的类，如 SpannableString 来格式化部分字符串。关于 SpannableString 的详细用法，请参考：
http://blog.csdn.net/yang_hui1986527/article/details/6776629

注：默认情况下，第 2，3，4 种方法可以显示链接，但是无法响应用户的点击输入。如果需要激活该响应，需要调用 TextView 对象的以下方法:
setMovementMethod(LinkMovementMethod.getInstance())

下面我们进行实例代码解析：

res-value-string.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="link_text_auto"><b>text1:</b> This is some text.  In
      this text are some things that are actionable.  For instance,
      you can click on http://www.google.com and it will launch the
      web browser.  You can click on google.com too.  And, if you
      click on (415) 555-1212 it should dial the phone.
    </string>
    <string name="link_text_manual"><b>text2:</b> This is some other
      text, with a <a href="http://www.google.com">link</a> specified
      via an &lt;a&gt; tag.  Use a \"tel:\" URL
      to <a href="tel:4155551212">dial a phone number</a>.
    </string>
</resources>
```

res-layout-link.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:orientation="vertical"
              android:layout_width="match_parent"
              android:layout_height="wrap_content">


  <!-- 四个 TextView 控件, 每个控件都显示包含链接的文本。 -->


  <!-- text1 自动识别文本链接，例如 URL 网络地址和电话号码等。 -->
  <TextView xmlns:android="http://schemas.android.com/apk/res/android"
            android:id="@+id/text1"
            android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:autoLink="all"
        android:text="@string/link_text_auto"
        />

<!-- text2 使用包含用<a>等显式 HTML 标记来指定链接的文本资源。 -->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/text2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="@string/link_text_manual"
        />

<!-- text3 在 Java 代码中使用 HTML 类来构造包含链接的文本。 -->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/text3"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        />

<!-- text4 在 Java 代码中不使用 HTML 类来构造包含链接的文本。 -->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/text4"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        />

</LinearLayout>
```

src-com.example.android.apis.text-Link.java

```
package com.example.android.apis.text;

import com.example.android.apis.R;

import android.app.Activity;
import android.graphics.Typeface;
import android.os.Bundle;
import android.text.Html;
import android.text.SpannableString;
import android.text.Spanned;
import android.text.method.LinkMovementMethod;
import android.text.style.StyleSpan;
import android.text.style.URLSpan;
import android.widget.TextView;

public class Link extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        //super.onCreate(savedInstanceState)是调用父类的 onCreate 构造函数
        //savedInstanceState 是保存当前 Activity 的状态信息
        super.onCreate(savedInstanceState);

        //将 link 布局文件渲染出一个 View 对象，并作为 Activity 的默认 View
        setContentView(R.layout.link);

        // text1 通过 android:autoLink 属性自动识别文本中的链接，例如 URL 网络地址和电话号码等。
        // 不需要任何 java 代码来使之起作用。

        // text2 含有由<a>等 HTML 标记指定的文本链接。默认情况下，这些链接可以显示但不会响应用户输入。
        //要想这些链接响应用户的点击输入，你需要调用 TextView 的 setMovementMethod()方法 。

        TextView t2 = (TextView) findViewById(R.id.text2);
        t2.setMovementMethod(LinkMovementMethod.getInstance());

        // text3 显示在 java 代码中通过 HTML 类来创建包含文本链接的文本，而不是从文本资源中创建。
        //请注意，对于一个固定长度文本，最好像上面的例子一样，从文本资源中创建。
        // 这个例子仅仅说明您怎样去显示来自动态来源（例如，网络）的文本。

        TextView t3 = (TextView) findViewById(R.id.text3);
        t3.setText(
            Html.fromHtml(
                "<b>text3:</b>  Text with a " +
                "<a href=\"http://www.google.com\">link</a> " +
                "created in the Java source code using HTML."));
        t3.setMovementMethod(LinkMovementMethod.getInstance());

        // text4 举例说明完全不通过 HTML 标记来构建一个包含链接的格式化文本。
        // 对于固定长度的文本，你最好使用 string 资源文本（即在 string.xml 中指定），而不是硬编码值（即在 java 代码中指定）。

        //构建一个 SpannableString
        SpannableString ss = new SpannableString(
            "text4: Click here to dial the phone.");

        //设置粗体
        ss.setSpan(new StyleSpan(Typeface.BOLD), 0, 6,
                Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
```

```
        //设置电话号码的链接
        ss.setSpan(new URLSpan("tel:4155551212"), 13, 17,
                Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);

        TextView t4 = (TextView) findViewById(R.id.text4);
        t4.setText(ss);
        t4.setMovementMethod(LinkMovementMethod.getInstance());
    }
}
```

知识点 1：