高煥堂 著


# 建模与图形思考

2012_6 月版

**架构师(Architect)的职责就是创意设计与人际沟通。在规划架构或框架的阶段，还没开始动工撰写 Android 程序码，那么架构师如何进行创意思考呢？又如何将创意设计表达出来，争取自己公司老板和业主的支持(例如投资)呢?**

**大家都知道，像举世公认的创意天才：达芬奇(Leonardo da Vinci)。他具有非凡的图形绘制和思考能力(例如蒙娜莉萨的微笑就是他的名作)。这提醒了我们，身为架构师，其图形绘制和思考能力愈好，其创意设计与人际沟通能力就愈好。**

**因此，培养 Android 架构师的图形思考能力是极为重要的。**

「這又一次證明，喬布斯把製造偉大產品的激情擺在了比迎合消費者的慾望更為重要的位置上。」 *~~ 摘自<<史蒂夫.喬布斯 傳>>*

# 目 录

## 第一篇 建模：登堂入室篇

# 第二篇 创意：图形思考篇



高煥堂(Tom Kao) 簡歷

- 学历：美国 U. of Colorado 资管研究所
  台湾淡江大学管理科学研究所
- Email：*misoo.tw@gmail.com*
- 微博：新浪微博@高煥堂
- Android 培训&顾问经历：
  2009-2010 年的酷派 Android 手机开发团队培训
  2010 年的中兴移动、LG 北京 Android 技术培训
  2010 年的腾讯、支付宝、百度的 Android 技术培训
  2010 年的 MTK、宏碁的 Android 技术培训
  2011 年的 OPPO、MOTO、联迪的 Android 技术培训
  2011 年的 MSTAR、长虹的 Android 技术培训
  2012 年(未列入)
- 曾任：铭传大学专职讲师
  台湾<<面向对象 Object-Oriented 杂志>> 主编
  英国 Access Capital 公司软件架构师(西班牙.巴塞隆纳)
- 现职：台湾 Android 论坛 主席
  亚太地区 Android 大会(APAC)主席

北京<中国电子视像行业协会>智能电视平台　高级顾问

- 技术专长：

  专精于 Android 核心框架及核心服务程序开发

- 书籍著作：

  出版 6 本 Android 专业技术书籍。

- Android 之网页/杂志文章：

  一　发表 400 多篇中文 Android 技术性文章

  *http://www.android1.net/Board.aspx?BoardID=11&GroupID=0*

  一　发表 20 多篇日本语 Android 核心技术文章

  *http://android-force.com/bbpress/forum.php?id=3*


# 第一篇 建模：登堂入室篇

# 第 1 章
# 引言：UML 图形语言

## 1.1 UML 与 Astah 建模工具

UML 是个很好的图形建模语言，也有很多 UML 建模工具，对于 Android 架构师或开发者来说，都是非常重要的。无论在培养图形思考或团队沟通上，对于 Android 软件创意或管理上，是无可取代的。

Astah Professional(原名 JUDE)是 UML 建模工具中，最具有简洁设计、轻便简单、易学好用的。Astah 功能强大，支持 UML2.x 中的图表、元素及属性，包括：

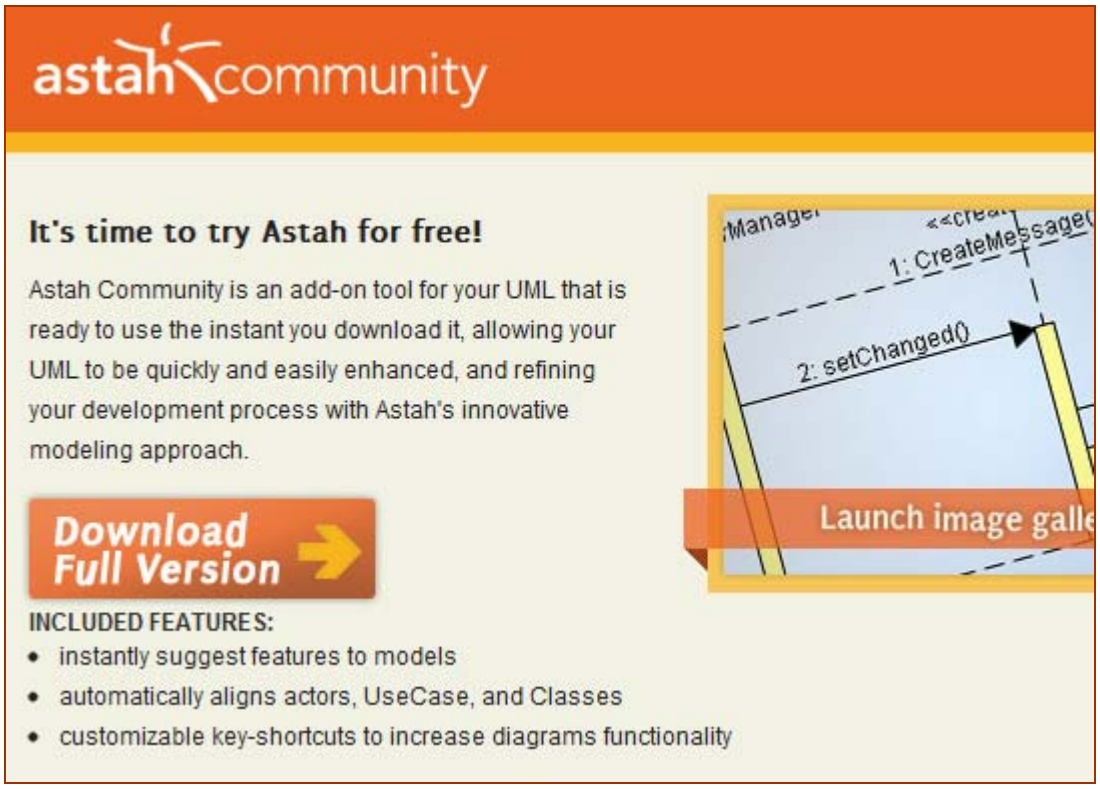
- Class(类)
- UseCase(用例)
- Sequence(顺序)
- Activity(活动)
- Communication(通信)
- Statemachine(状态机)
- Component(模块)
- Deployment(布署)

- Composite Structure(组合结构)
- Object and Package Diagrams(对像套件)

## 1.2 免费的 Astah 小区型(Community)版本

您可以免费使用其小区型(Community)版本来练习，并轻松愉快地建立 Android 项目或项目的各种图形模型，提升设计、开发和管理效率。Astah 的网址如下：

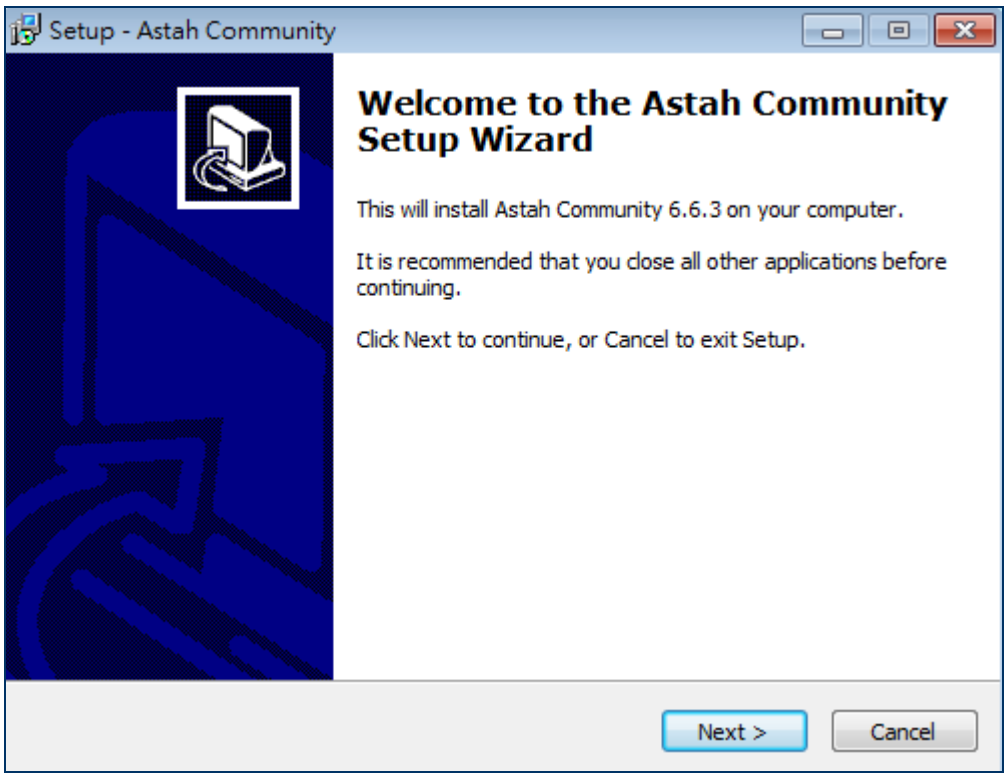

你可以下载它的小区版本：



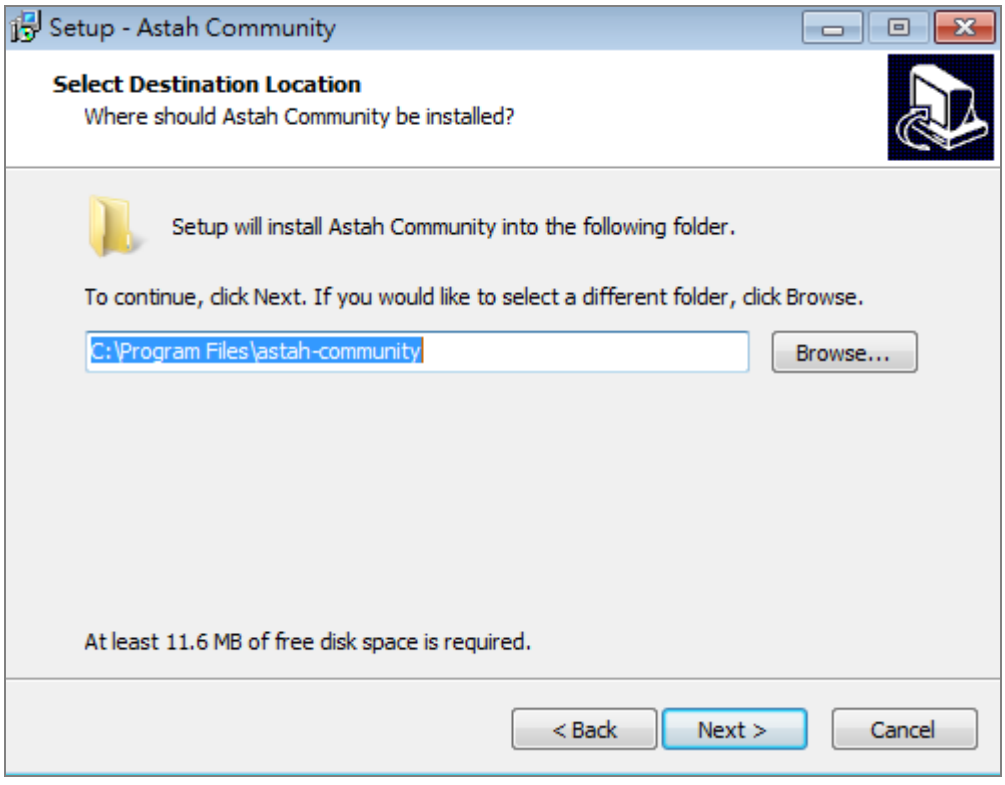
按下<Download Full Version>就能选择你需要的版本：


假如你选择<32-bit JRE-bundled>，出现：

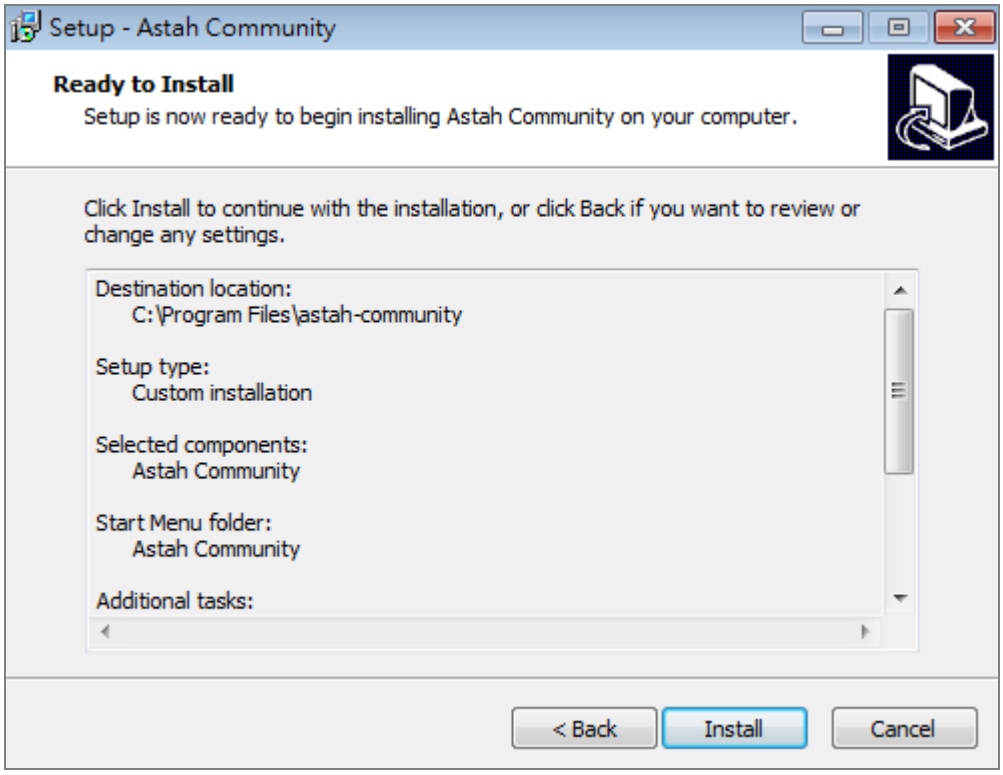

按下<Download>，就开始下载，并进行安装：



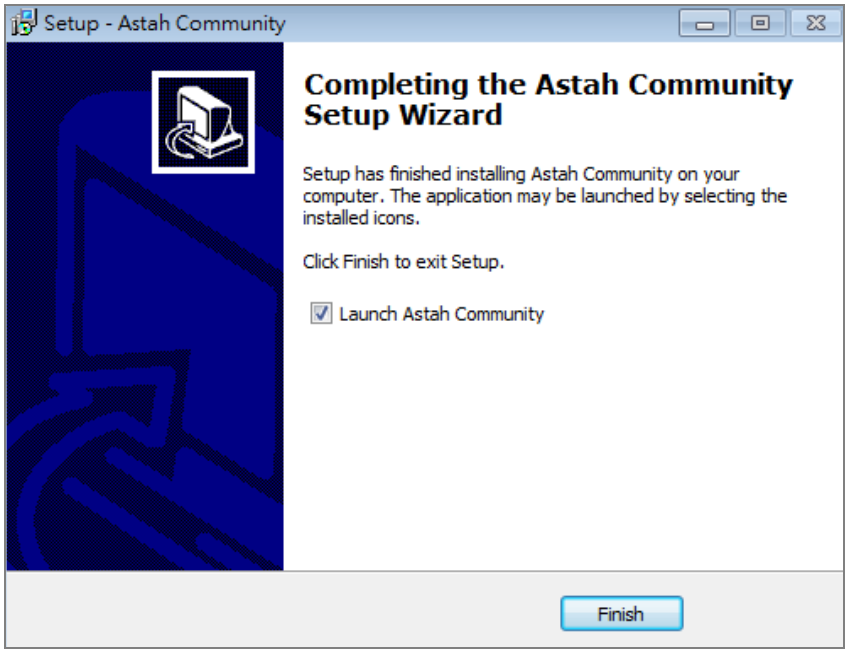

按下<Next>，就让你选择想安装在那个位置(例如 C:/Program Files\astah-community 活页夹)：




按下<Next>，就准备进行安装：




按下<Install>，就执行安装：



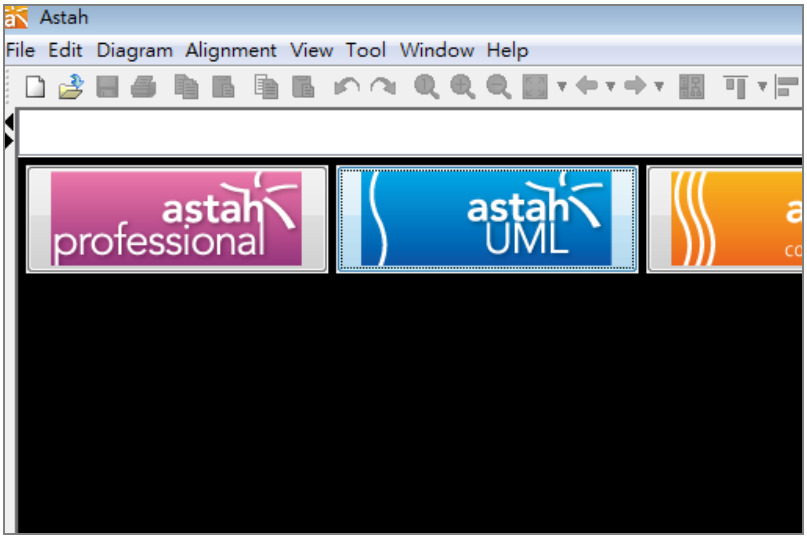

这就安装完成了，按下<Finish>就离开安装环境(Setup Wizard)了。

## 1.3 Astah 幕后需要 JRE 1.6

安装好了 Astah 之后，就可以准备开始使用 Astah 来进行 UML 建模了。但是，你的计算机必须先有 JRE 1.6 以上的执行环境才能执行 Astah 工具。

## 1.4 启动 Astah 工具

启动了 Astah，出现主画面：




此时就能着手绘制各种 UML 图表(Diagram)了，包括：

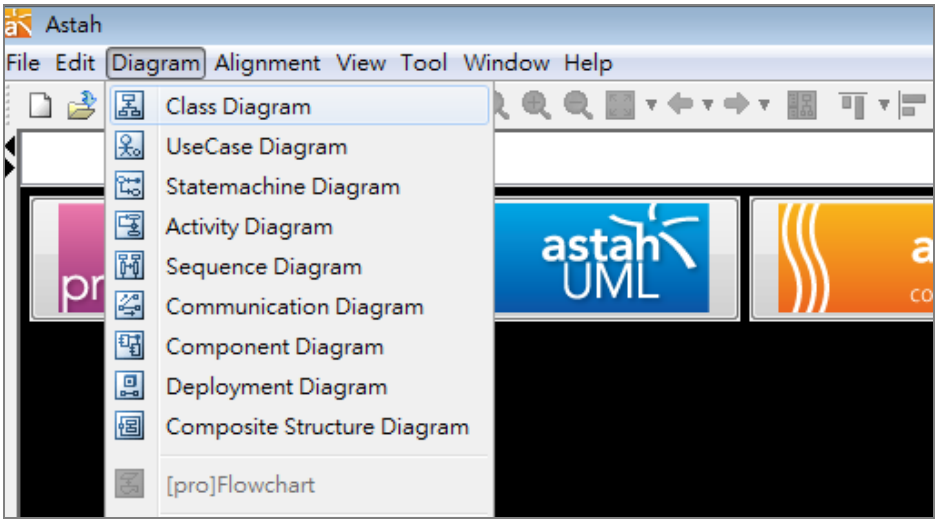
- Class Diagram(类别图)
- UseCase Diagram(用例图)
- Statemachine Diagram(状态机图)
- Sequence Diagram(顺序图)
- Activity Diagram(活动图)
- Communication Diagram(通信图)
- Component Diagram(模块图)
- Deployment Diagram(布署图)
- Composite Structure Diagram(组合结构图)

在 UML 里，以套件(Package)来组织模型(Model)，套件可以包含多个小套件，套件里可包含多张各种图表(Diagram)。这些图表就构成模型。◆

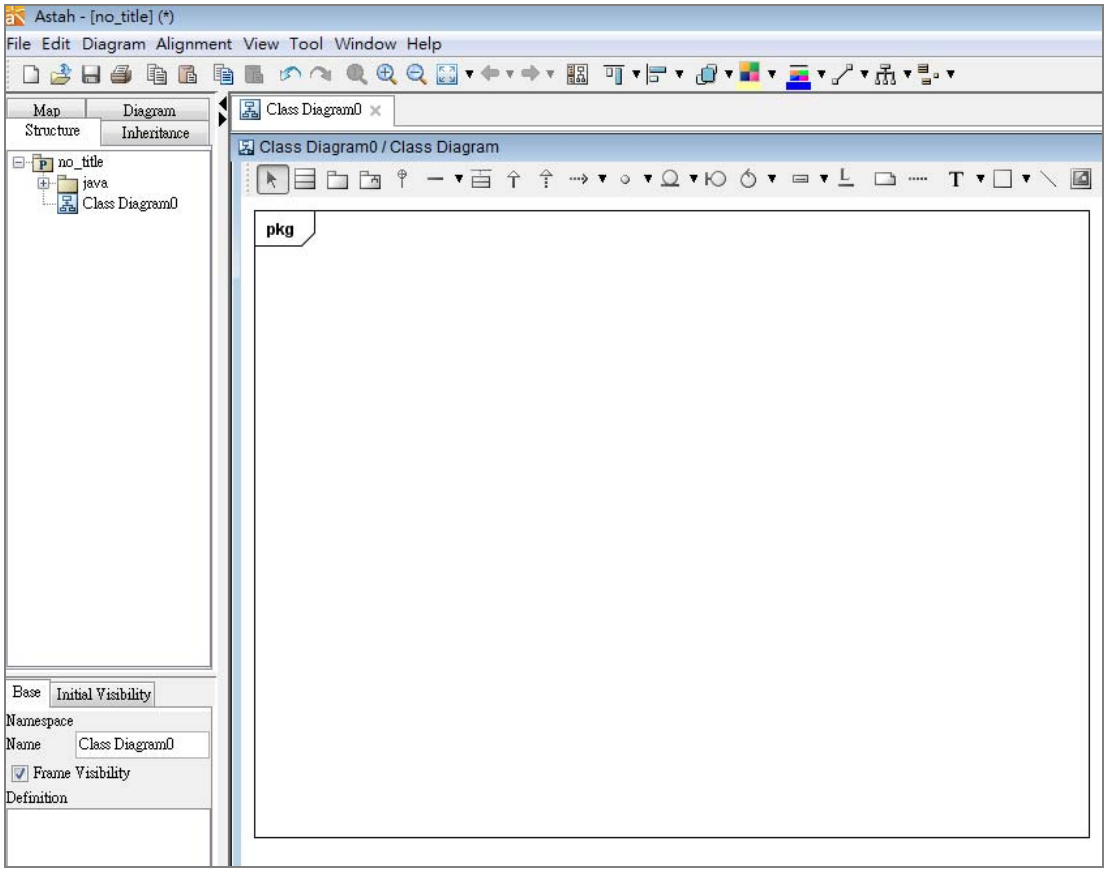

# 第 2 章
# 30 分钟认识 UML 类别图

## 2.1 绘制 UML 类别图(Class Diagram)

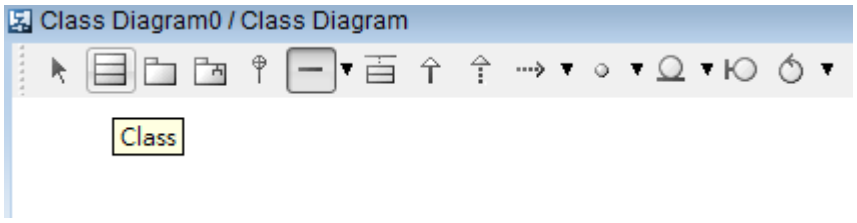
启动了 Astah，在主画面上点选< Diagram>，出现：



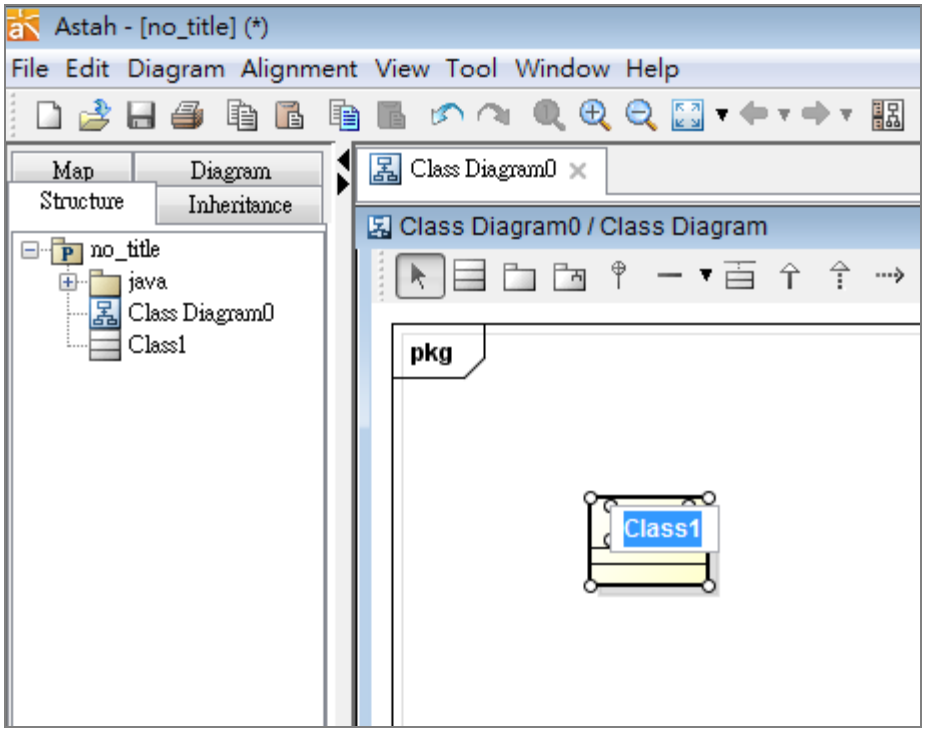
接着，点选<Class Diagram>，就会出现一张空白的类别图，如下：



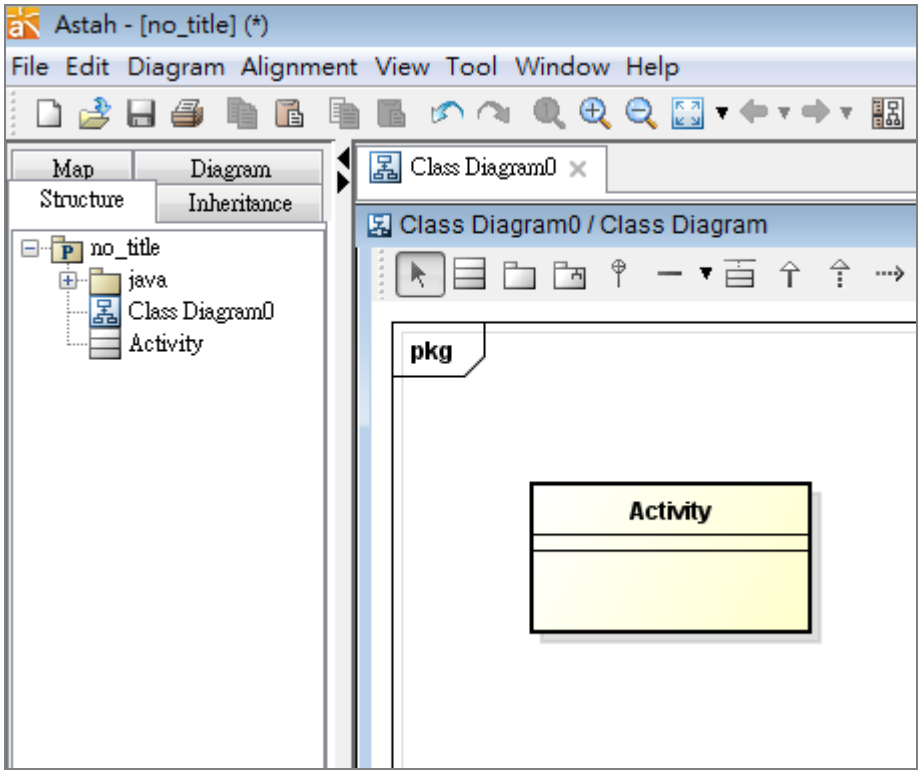

在空白的类别图上方，有一排类别图的元素(Element)，简称「图素」，如下：




此列元素中的左边第 2 个就是「类别」(Class)图素。当你点选此图素(如上图所示)，然后移动鼠标(Cursor)到图表里的特定位置，并按下鼠标左键，就在图表里出现一个类别图素，如下：



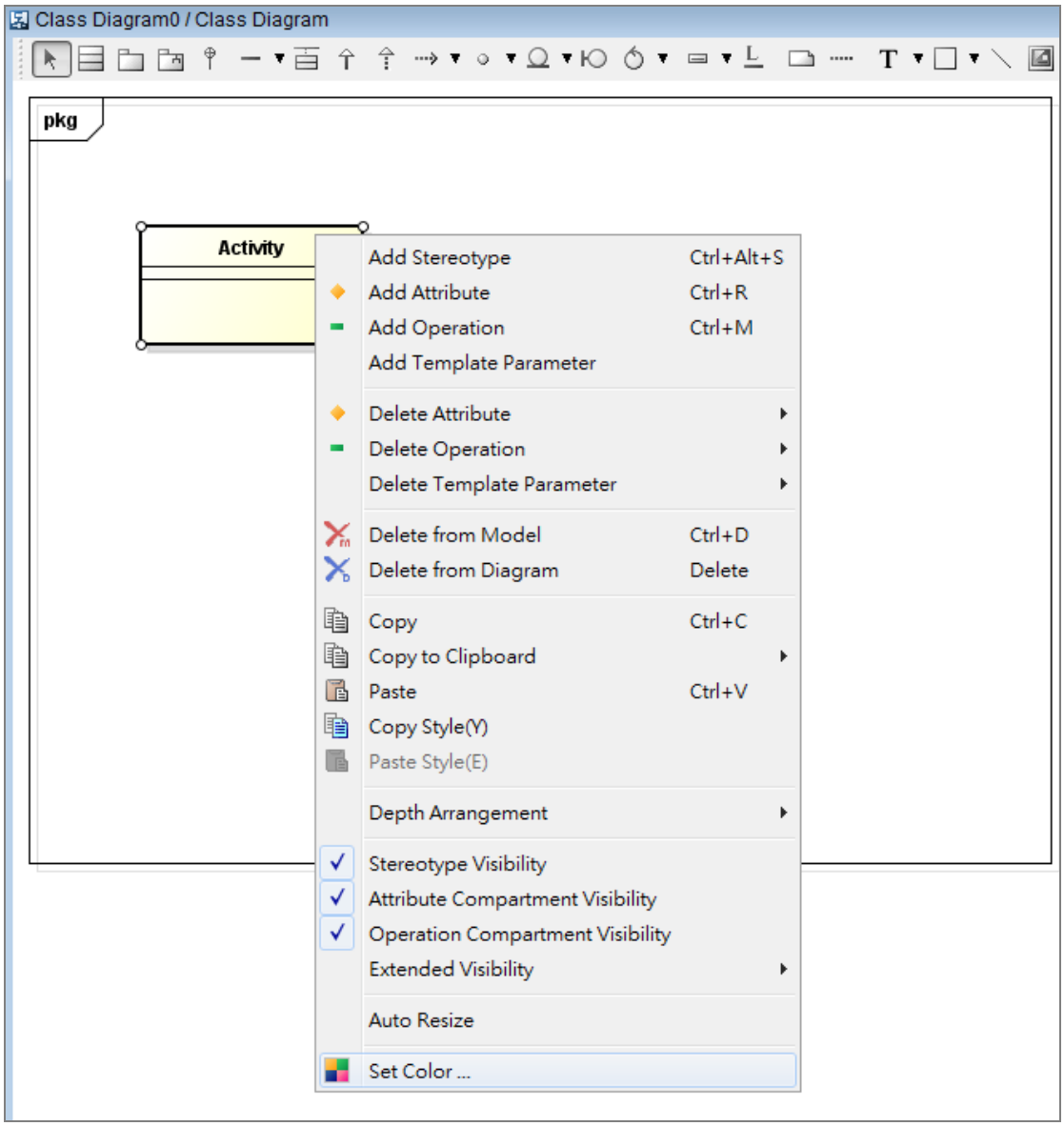

这让你输入类别的名称，例如取名为：Activity 类别，如下：



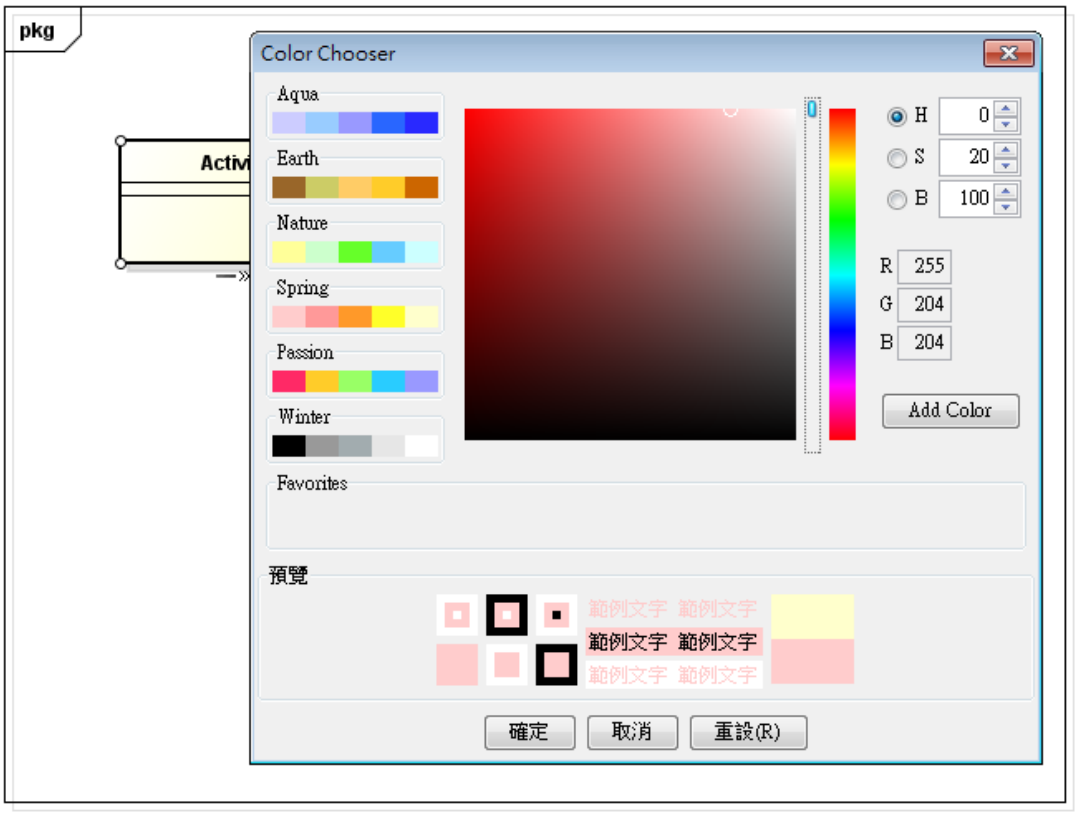

这就是在 Android 程序里，大家都熟悉的 Activity 基类(又称父类别)。例如，大家很熟悉的 Android 程序码片段：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
    // ……
}
```

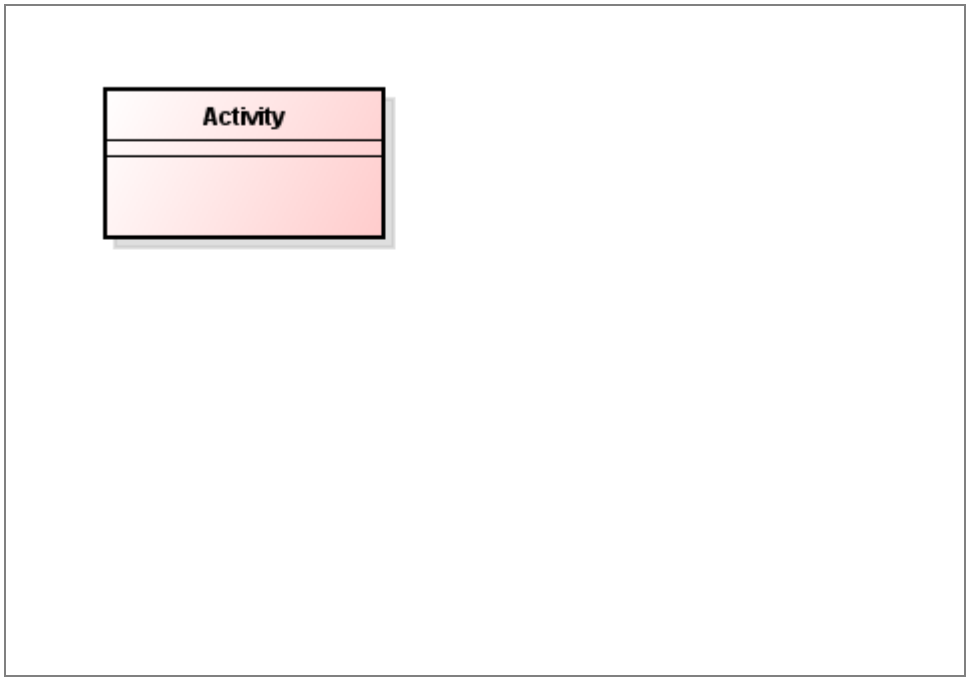
这样的图文对照，能有效培养架构师的图形思考和创意，提升架构师与项目经理、业主的沟通能力。在图形上，也能增加美感，培养架构师对软硬件的感觉(Feeling)而不是只能逻辑的理解(Understanding)，有助于与设计师进行创意交流。例如，点选这个 Activity 类别图素，并按右键；出现如下：




于是，可以选择<Set Color …>选项来改变图素的颜色(Color)。此时，出现如下：

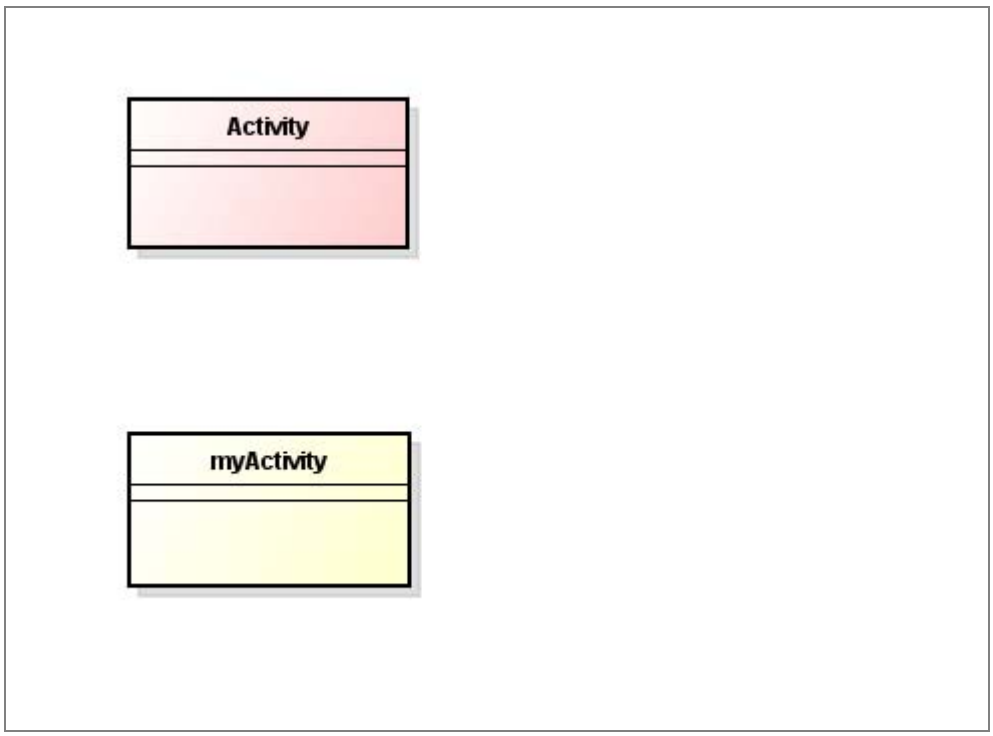



就能任选所喜欢的颜色了。例如，选取粉红颜色，如下：




到此，已经将 Activity 类别表示成图形了。依据同样的过程，可以再拉出一个

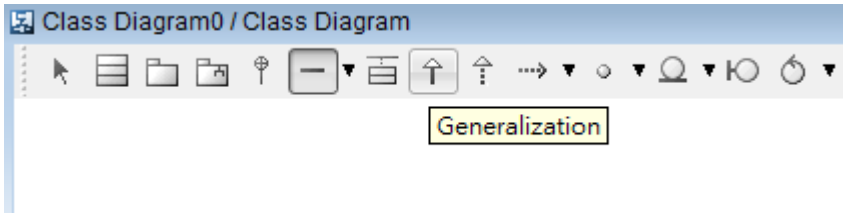
myActivity 类别的图素，如下：



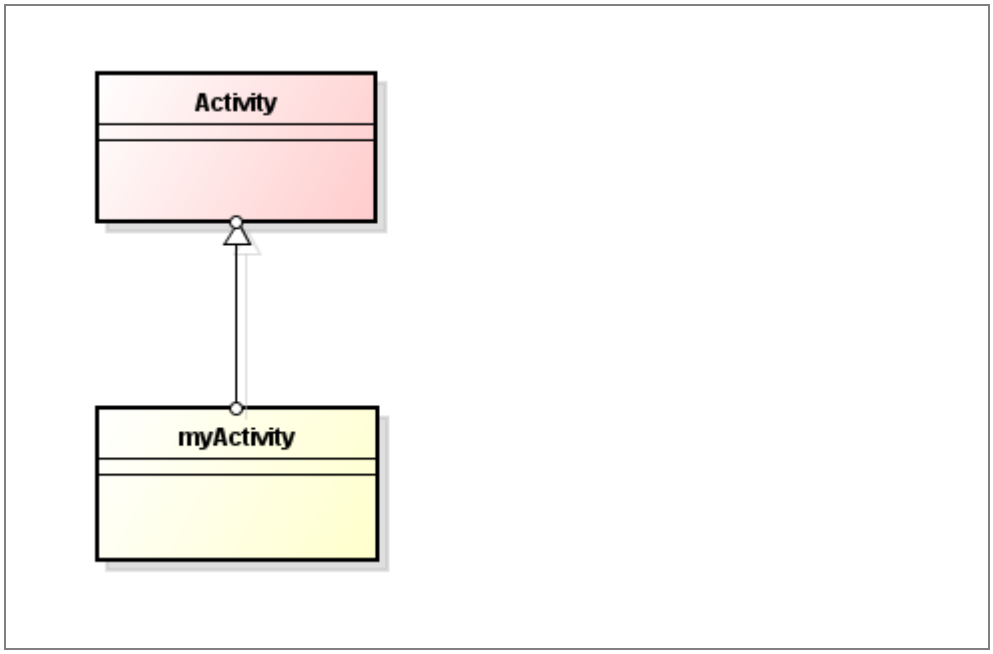

如果再对照到大家所熟悉的程序码：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
    // ……
}
```

上图就表达了这段程序码里的两个类别：Activity 和 myActivity 了。接着，可以在图形上表达两个类别之间的关系(Relationship)。例如，选取<Generation>图素，如下：




请点选了这个图素，接着将鼠标移动到 myActivity 类别图素，按住并拖拉到 Activity 类别图素才放开，就出现：

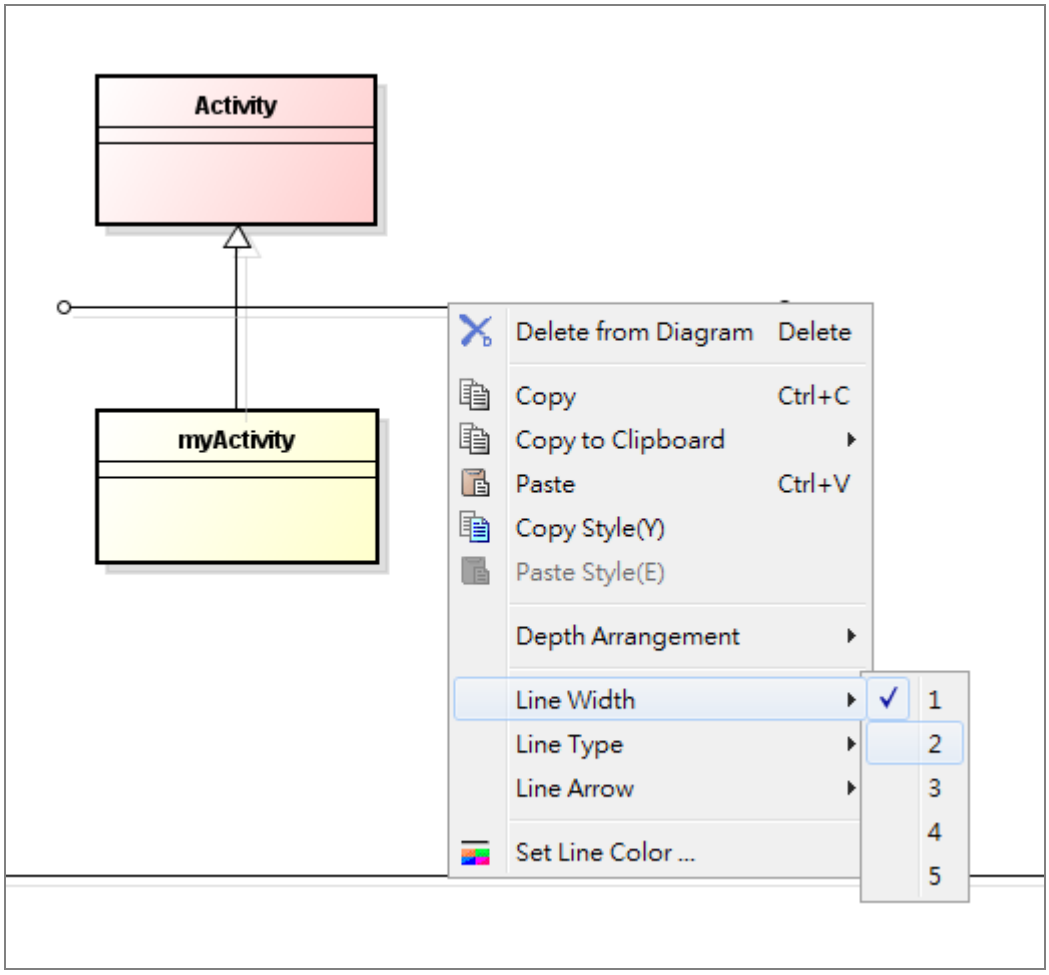



就建立了两个类别之间的<Generation>关系，这又称为「继承」(Inheritance)关系，或称为「扩充」(Extend)关系。如果再对照到大家所熟悉的程序码：

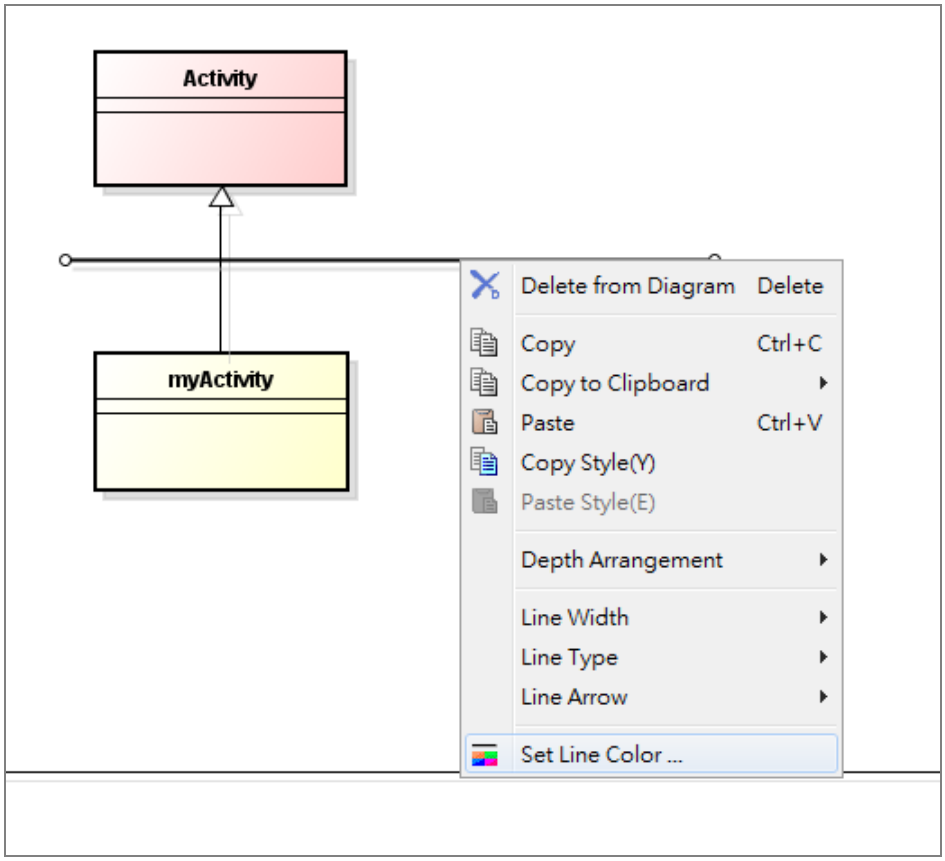
```
public class myActivity extends Activity implements OnClickListener {
    // ……
    // ……
}
```

上图就表达了这段程序码里，Activity 与 myActivity 之间的扩充关系了。由于 Activity 是由 Google 团队所撰写的，属于框架(Framework)的一部分；而 myActivity 则是由一般应用(APP)开发者所撰写的。在图形上，可以将它们分隔开来，让类图更为清晰明了。例如，选取<Line>图素，如下：

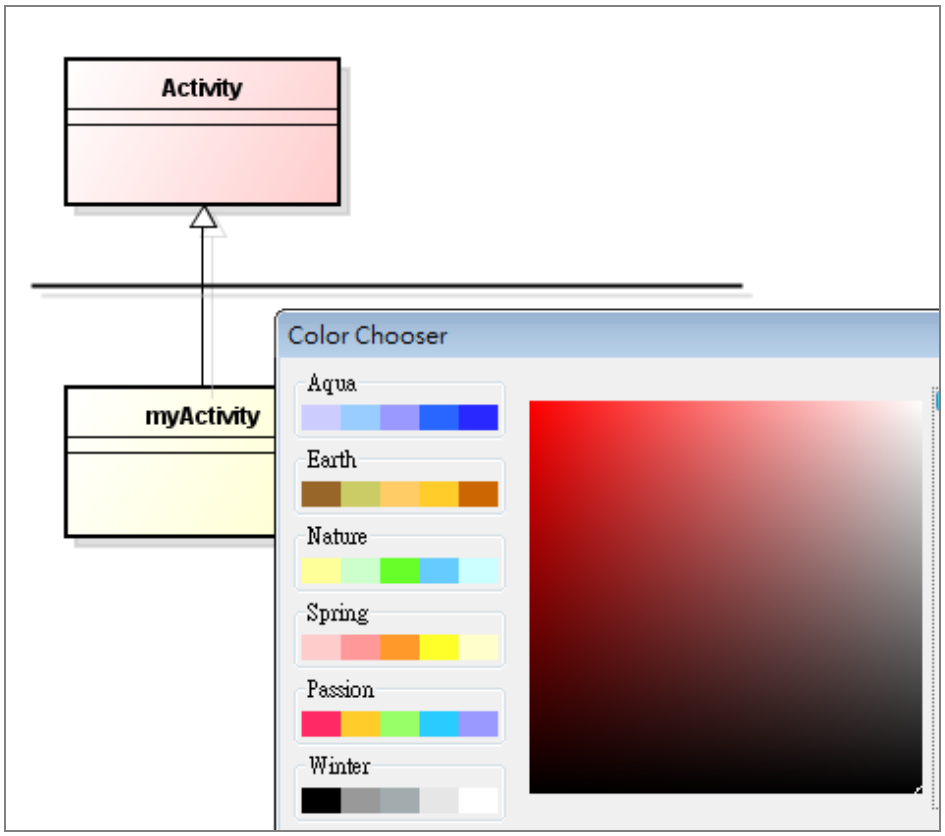

请点选了这个图素，接着将鼠标移动到类别图里的任何位置，按住并拖拉出一条直线才放开。此外，你还可以改变直线的宽度。例如，点选该直线并按右键，就出现：




此时，就能选择直线宽度了。例如，继续点选<Line Width>和<2>，直线就变粗了(如下图)。此外，也能变换直线的颜色。例如，点选该直线并按右键，就出现：

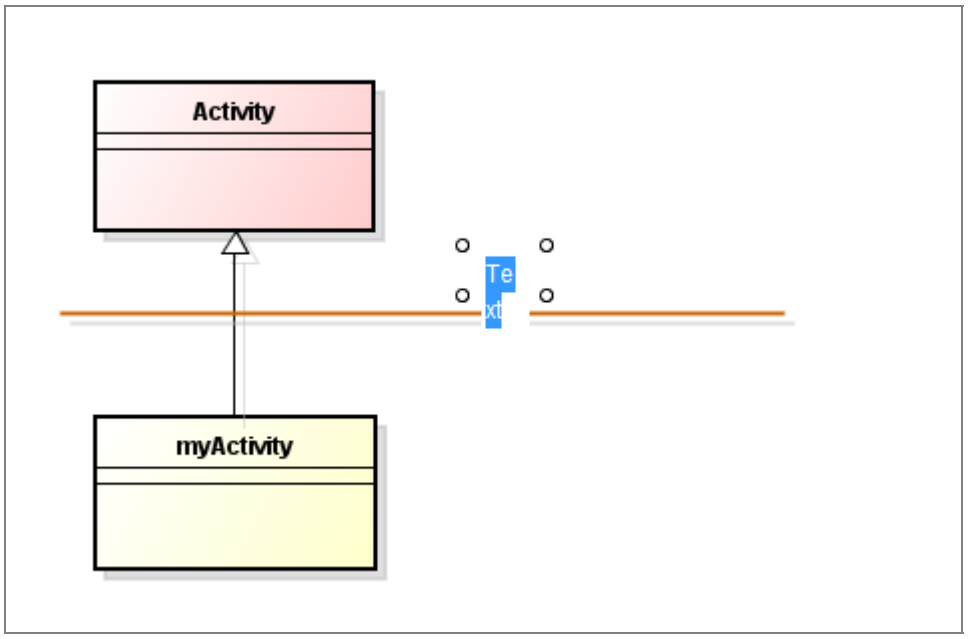



此时，继续点选<Set Line Color ...>，就出现：

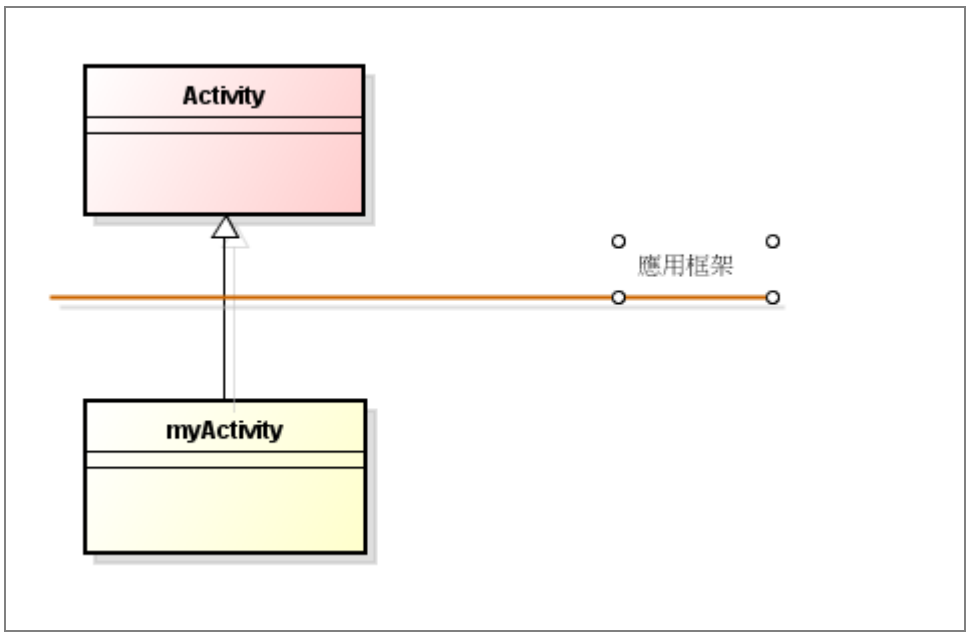


就能任选所喜欢的颜色了。例如，选取咖啡颜色。除了画直线之外，还能够在类别图里的任何位置添加文字(Text)。例如，选取<Text>图素，如下：

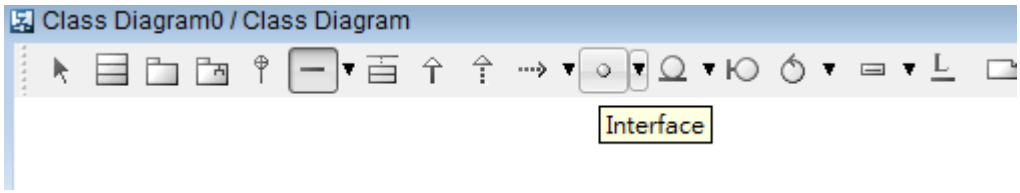

点选了这个<Text>图素，接着将鼠标移动到类别图里的任何位置，并按键，出现如下：



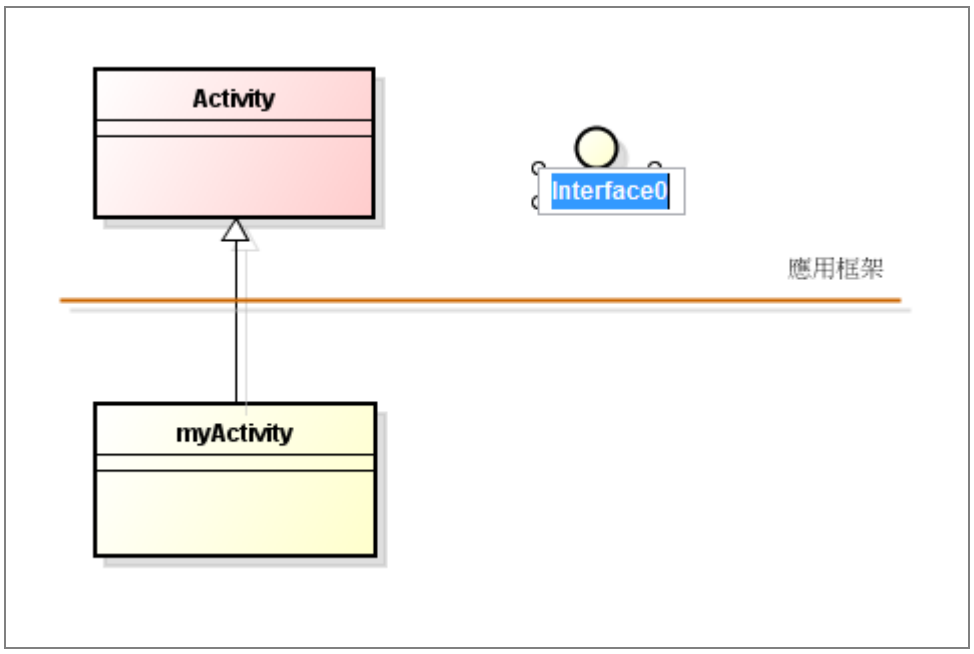

就可以填入所需的文字了，例如：



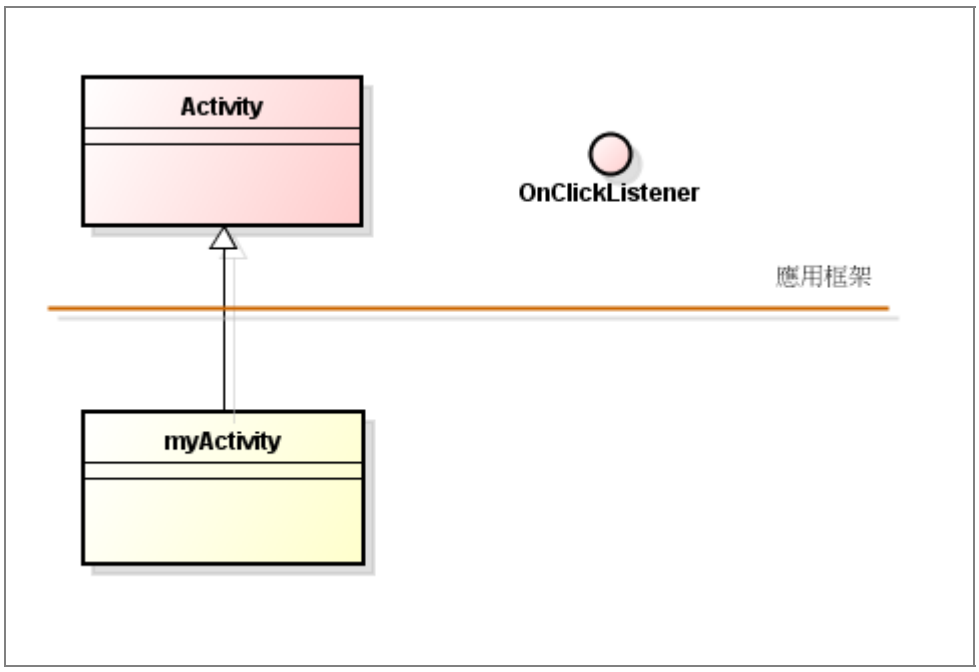
对于架构师而言，「接口」( Interface)的角色比「类别」(Class)来得重要多了。于是，就来看看如何将接口图素呈现类别图上。例如，选取<Interface>图素，如下：




点选了这个<Interface>图素，接着将鼠标移动到类别图里的任何位置，并按键，出现如下：



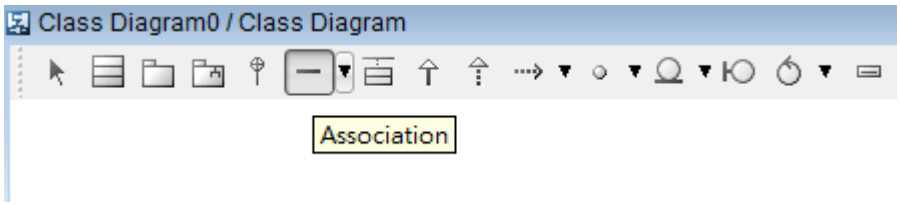

就可以填入接口的名称了，例如：



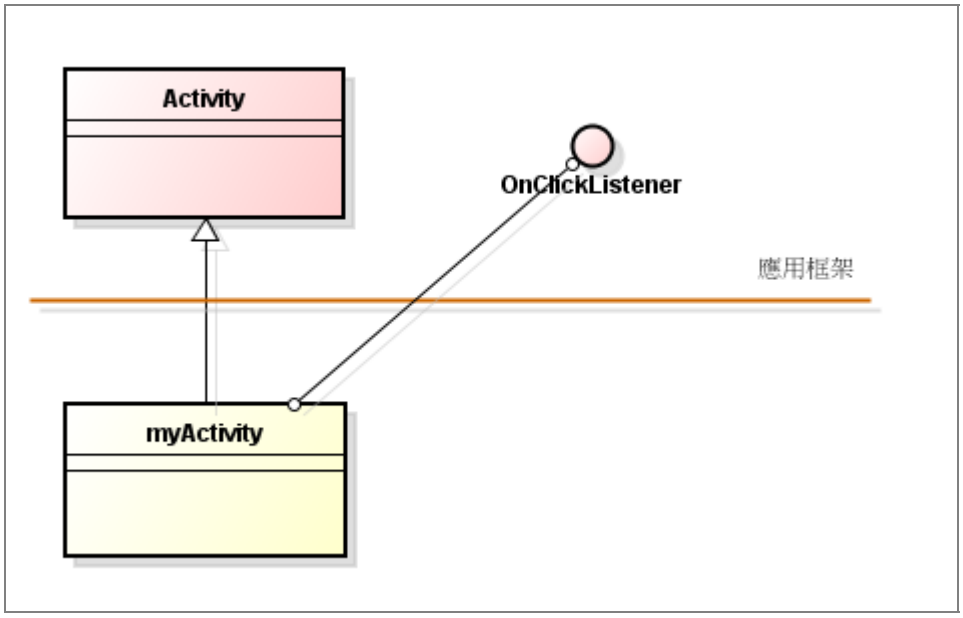

就替接口取个名称：OnClickListener。如果对照到大家所熟悉的程序码：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
    // ……
}
```

上图就表达了这段程序码里的 OnClickListener 接口了。接着，可以在图形上表达类别与接口之间的关系。例如，选取<Association>图素，如下：



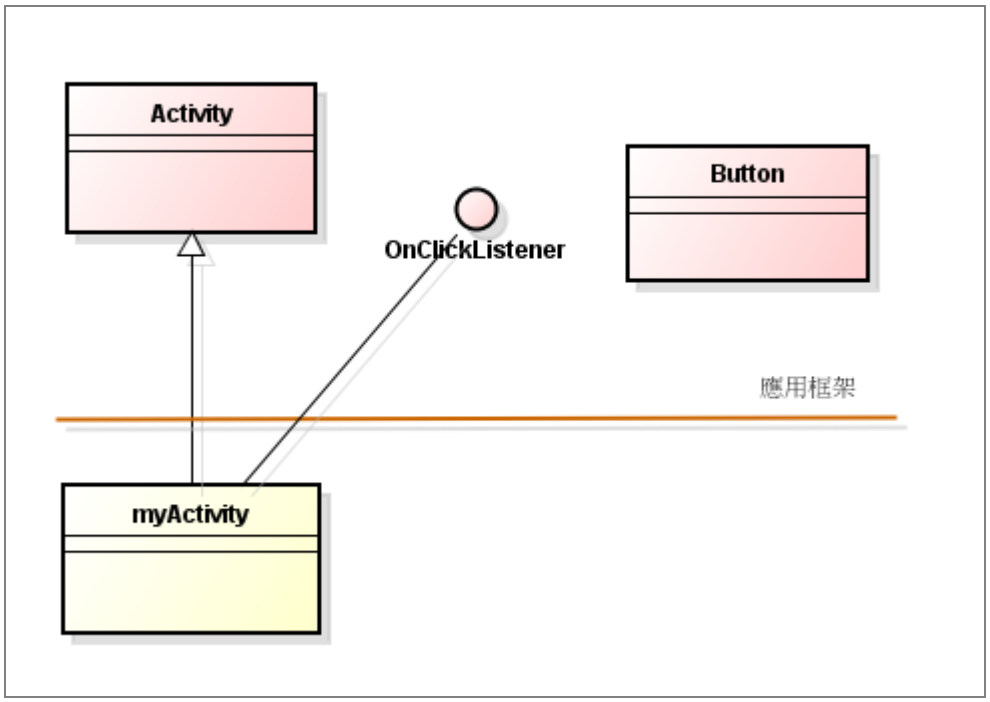

先点选这个图素，然后将鼠标移动到 myActivity 类别(图素)，按住并拖拉到 OnClickListener 接口(图素)，就出现：




如果对照到大家所熟悉的程序码：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
}
```

上图就表达了这段程序码里的 implements 关系了。此关细说明了：myActivity 类别「实作」了 OnClickListener 接口。其意味着，其它的类别可以透过此接口来调用 myActivity 类别里的函数。例如，可以拉出(产生)一个 Button 类别(图素)，如下：

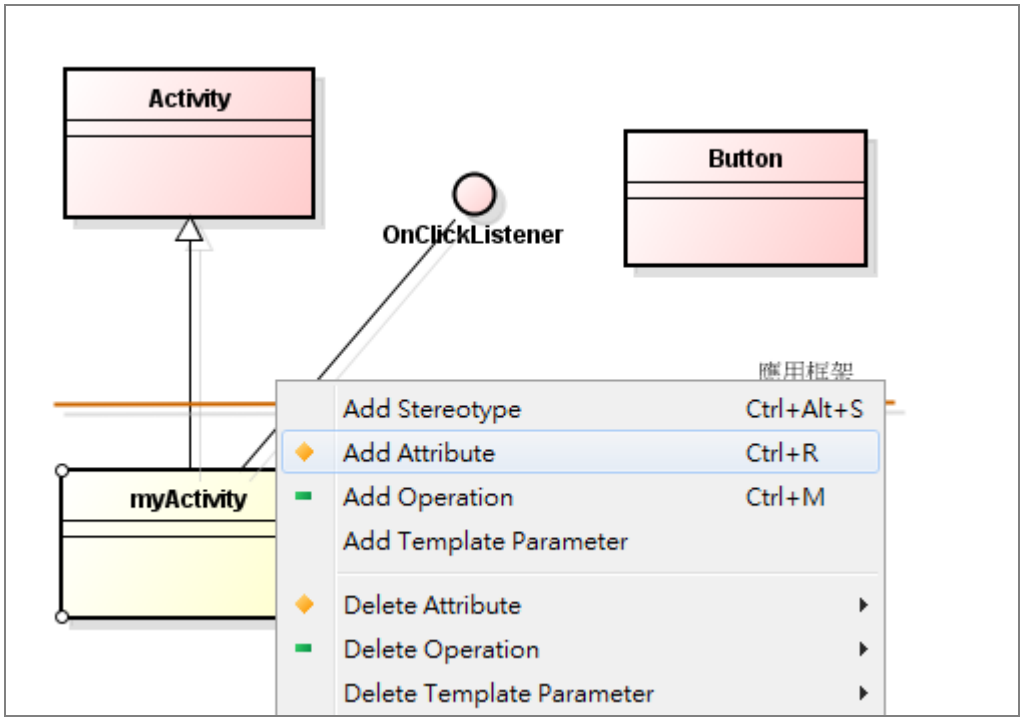


上图就表达了这段程序码里的 implements 关系了。此关系说明了：myActivity 类别「实作」了 OnClickListener 接口。其意味着，其它的类别可以透过此接口来调用 myActivity 类别里的函数。例如，可以拉出一个 Button 类别(如下图)，然后兹想一想，如何让 Button 类别来使用 OnClickListener 接口呢?

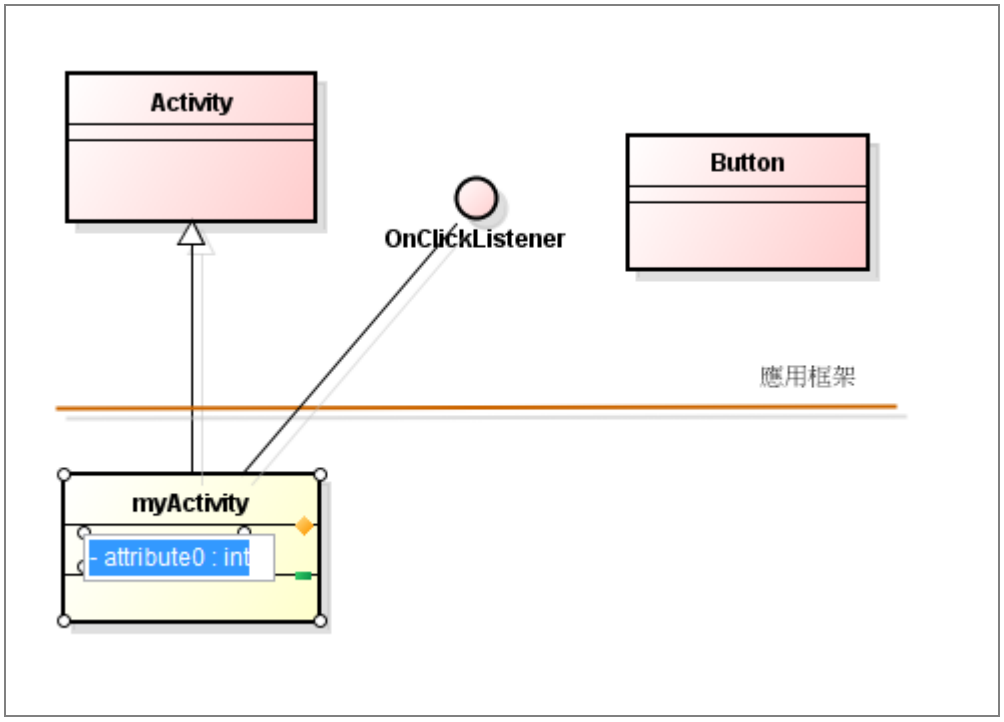
## 2.2 表达接口(Interface)

在撰写 Android 程序时，最典型的途径就是，由 myActivity 来建立一个 Button 类别的对象，然后把 OnClickListener 接口传给 Button 对象，就能让 Button 来使用 OnClickListener 接口了。其详细步骤是：

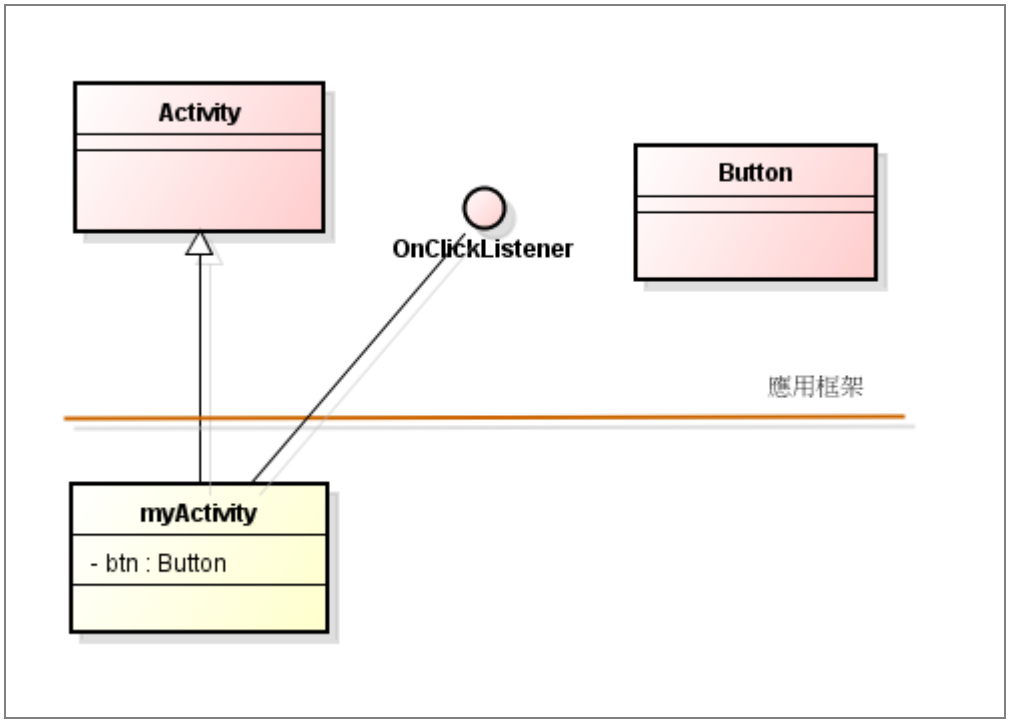
**Step-1：** 在 myActivity 里建立一个参考(Reference)属性，可参考到 Button 类别(的对象)。于是，点选 myActivity 类别，并按右键，出现：




选取<Add Attribute>来增添一个新属性，就出现了：

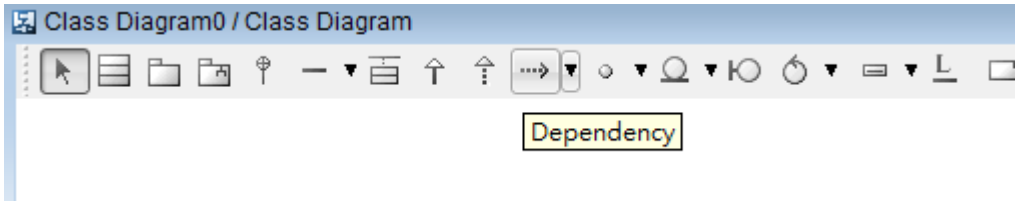



替属性取个名称：btn，其型态(Type)是：Button；如下图：

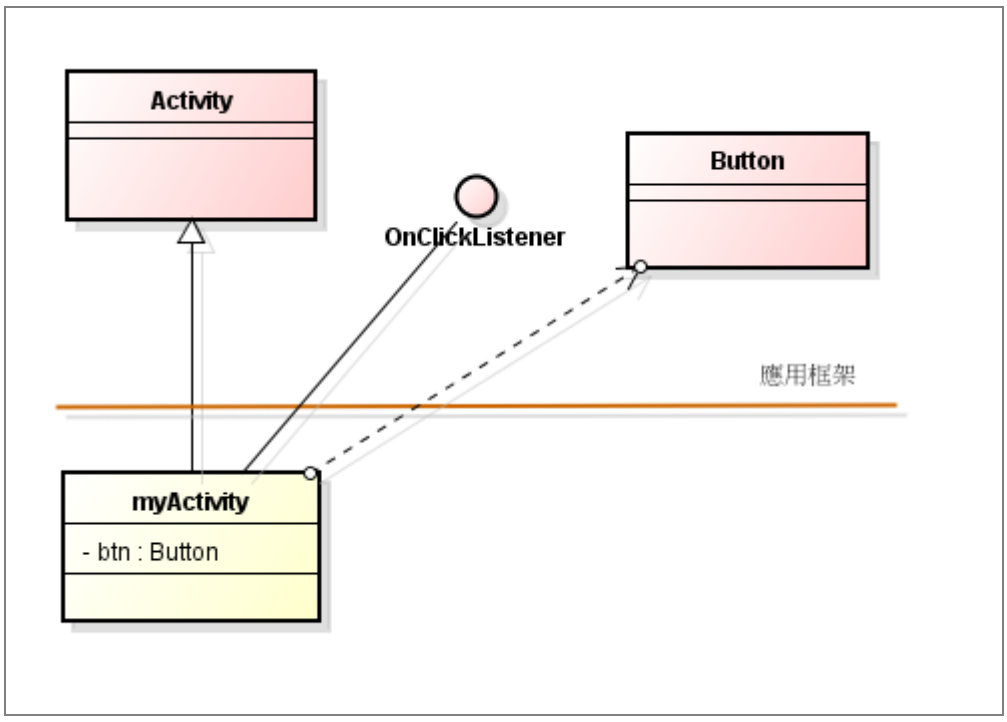



此时表达了：btn 属性将用来参考到 Button 类别(的对象)。

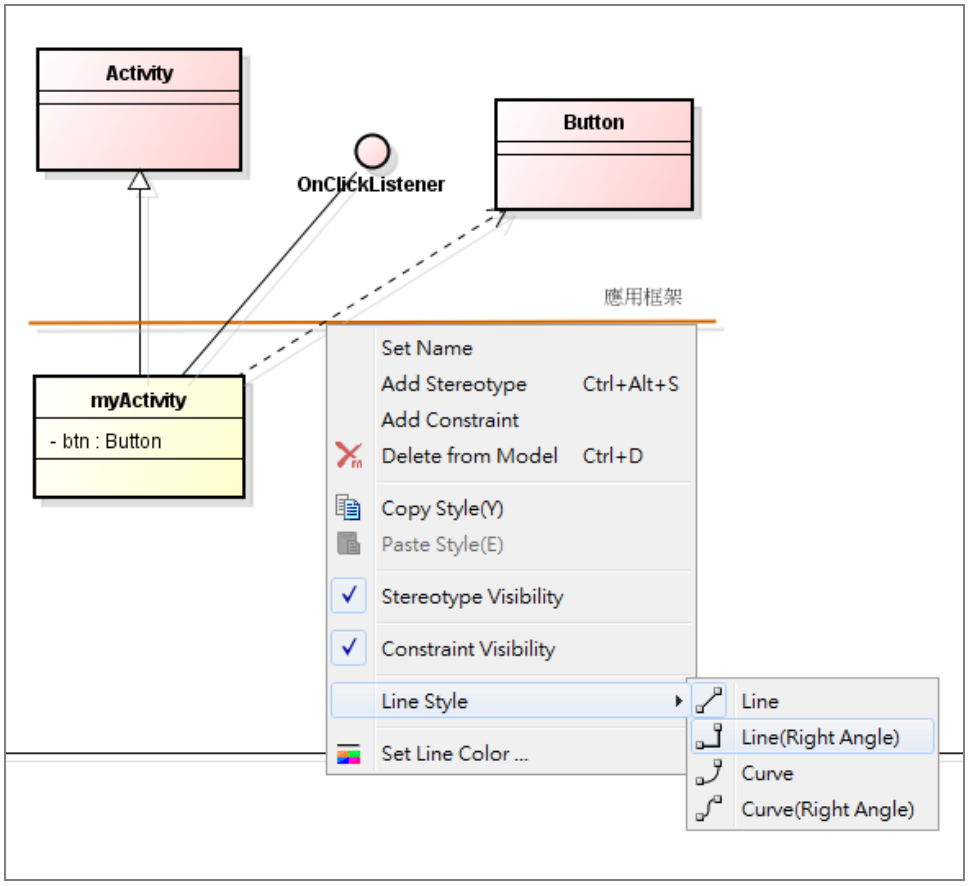
**Step-2：** 由 myActivity 来诞生一个 Button 类别的对象。为了表示这个<诞生>关系，可选取<Dependency>图素，如下：



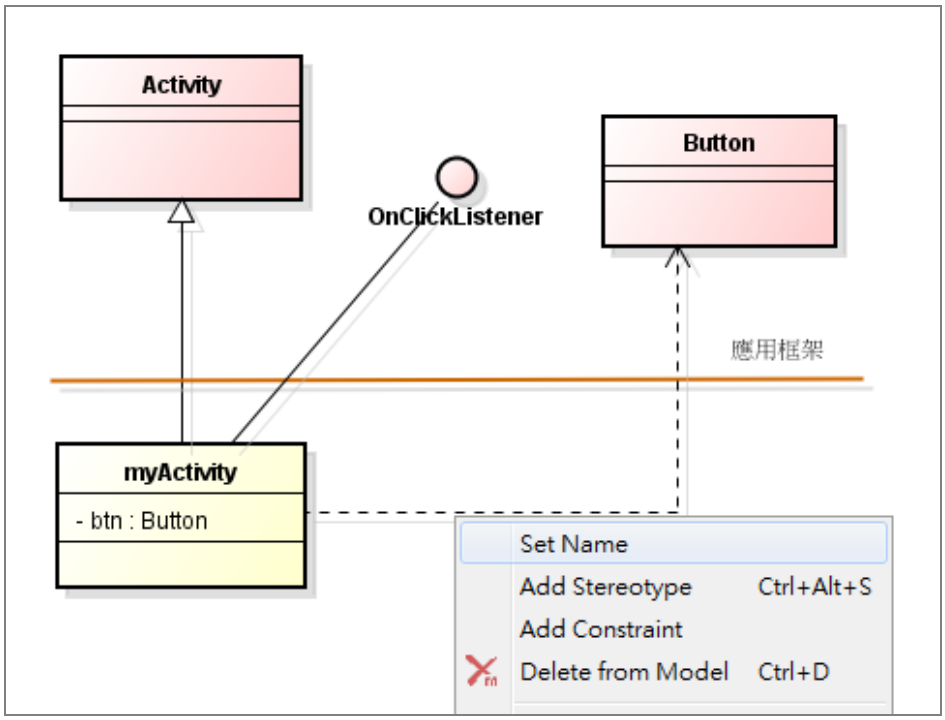

先点选这个图素，然后将鼠标移动到 myActivity 类别(图素)，按住并拖拉到 Button 类别，就出现：



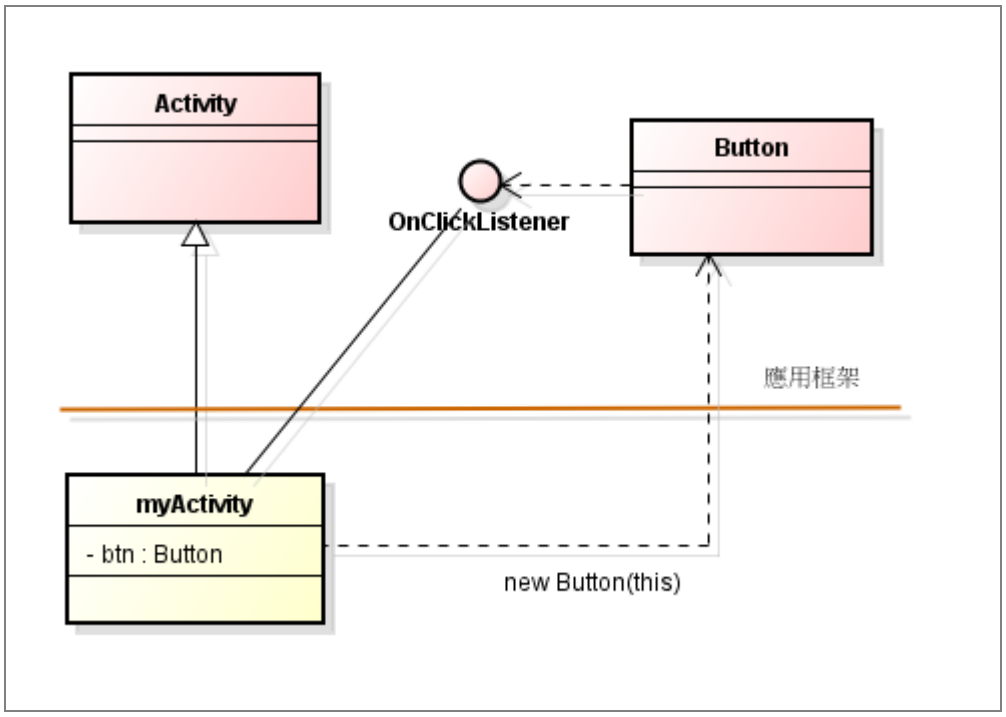

为了类别图的美观起见，可以改变线条的形状。例如，点选这个虚线，并按右键，出现如下：




例如，选取<Line Style><Line(Right Angle)>，就会改变为直角虚线了(如下图)。然后，可藉线条名称来注明其为<诞生>关系。例如，点选这个虚线，并按右键，出现如下：



于是，选取<SetName>，并填入名称如下：




如果对照到大家所熟悉的程序码：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
    @Override public void onCreate(Bundle icicle) {
            super.onCreate(icicle);
            // ……
            Button btn = new Button(this);
            // ……
}
```

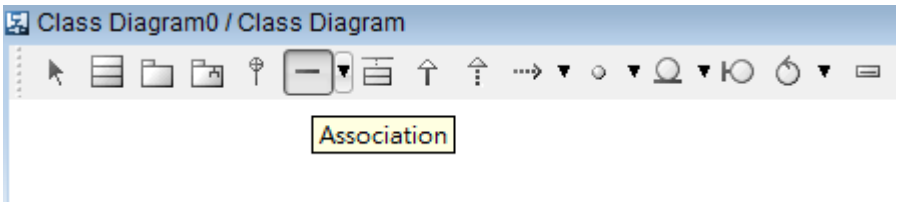
上图就表达了这段程序码里的指令：

Button btn = **new** Button(**this**);

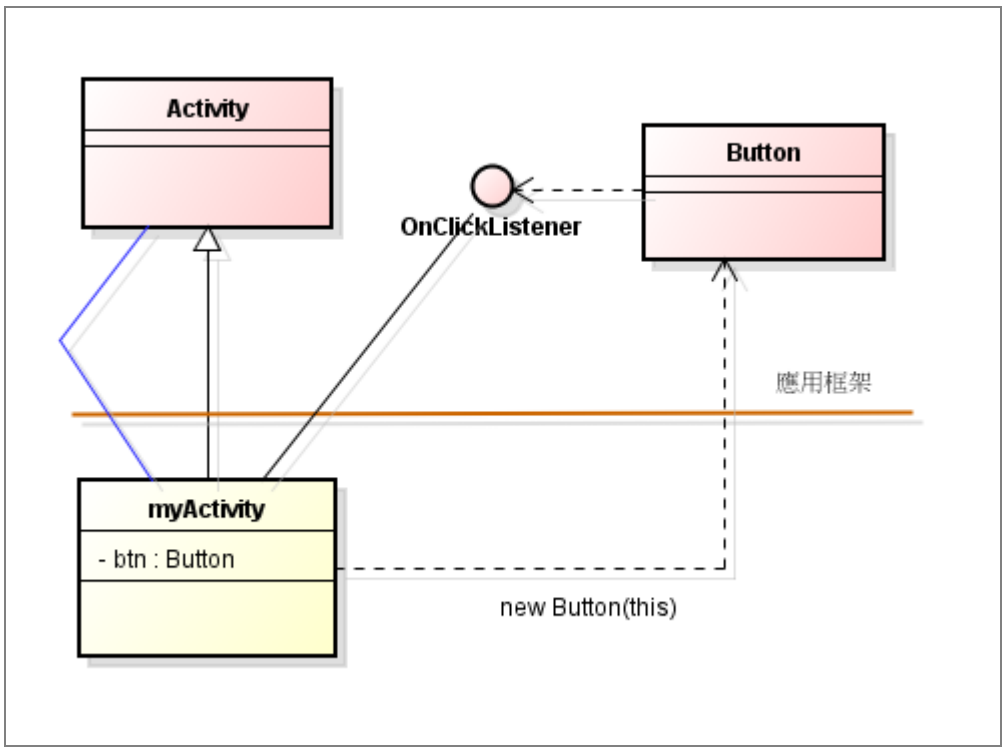
此指令将 this 传递给新诞生的 Button 对象，让 Button 对象可以指向(Point to)这个 OnClickListener 接口了。那么，在类别图里，应该如何来表示这种「指向」关系呢？此时，可拉一条虚线(并赋与箭头)，从 Button 类别指向 OnClickListener 接口(如上图)。

**Step-3：** 诞生了 Button 对象之后，由 myActivity 来调用 Activity 的 setContentView()函数，来将该 Button 对象的参考值(存于 btn 属性里)传递给框架的 Activity 类别，然后再传递给框架幕后的系统服务(即 WindowManagerService 和 SurfaceFlinger 服务)，这些服务才能依据 Button 对象内容来决定画面上的布局，以及画面上按钮的位置、大小、颜色等等。

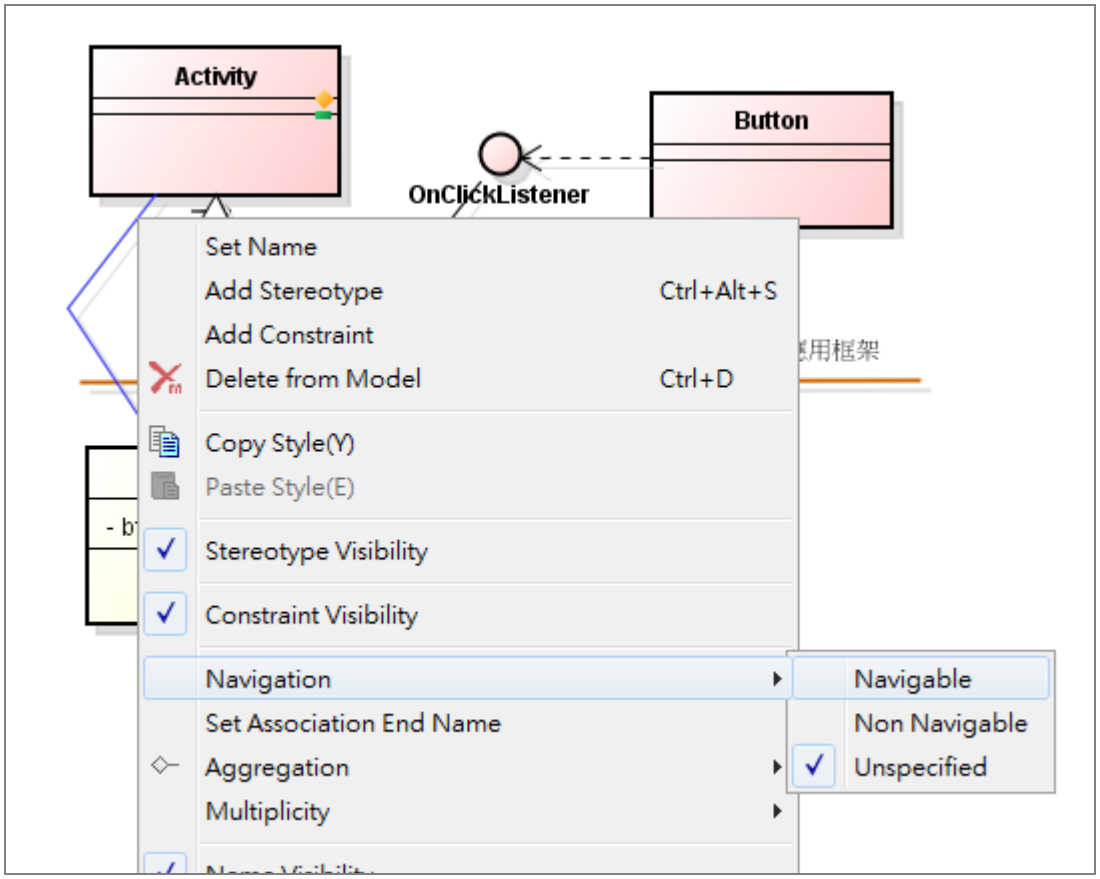
为了表示这个<调用>关系，可选取<Association>图素，如下：



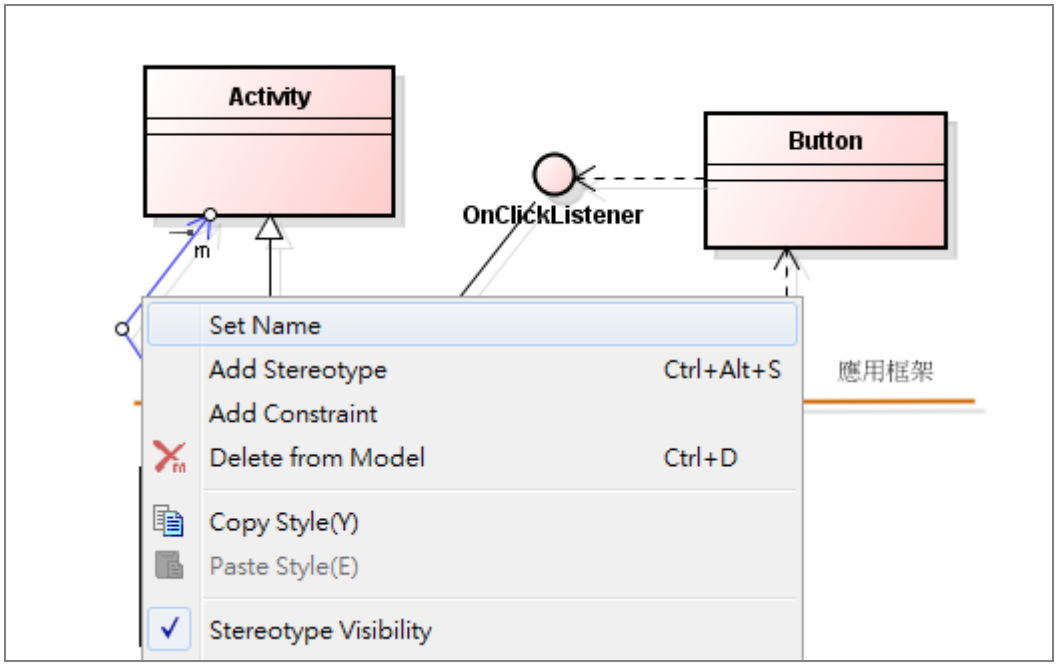

先点选这个图素，然后将鼠标移动到 myActivity 类别，按住并拖拉到 Activity 类别，就出现：



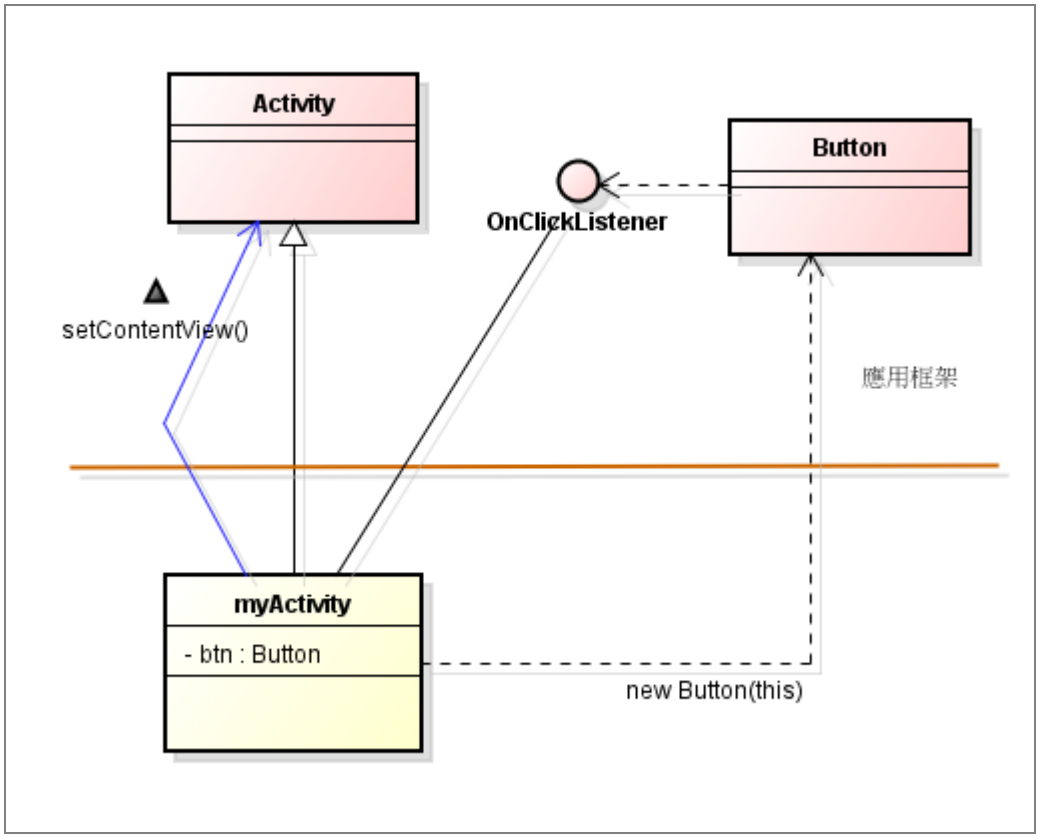

接着，还要将这条线加上箭头。于是，点选此线的上端部位(靠近 Activity 类别)并按右键，出现如下：




于是，选取<Navigation><Navigable>，就会出现箭头(如下图)。接着，可藉其名称来标示出其调用的函数名称。此时，点选此线并按下右键，出现：




于是，选取<Set Name>，并输入函数名称：setContentView()。如下图：




如果对照到大家所熟悉的程序码：

```
public class myActivity extends Activity implements OnClickListener {
    // ……
    @Override public void onCreate(Bundle icicle) {
            super.onCreate(icicle);
            // ……
            Button btn = new Button(this);
            // ……
            LinearLayout layout = new LinearLayout(this);
            layout.setOrientation(LinearLayout.VERTICAL);
            LinearLayout.LayoutParams param =
                    new LinearLayout.LayoutParams(WC, WC);
            layout.addView(btn, param);
            setContentView(layout);
}
```

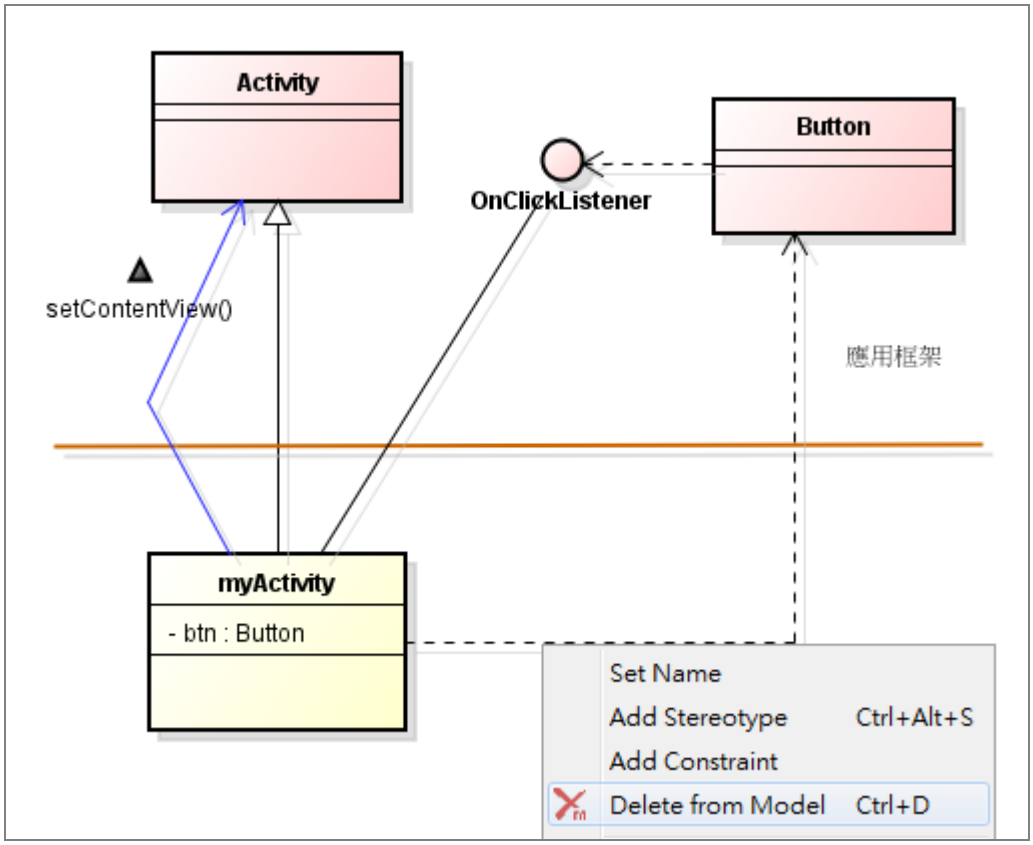
上图就表达了这段程序码里的指令：

setContentView(layout);

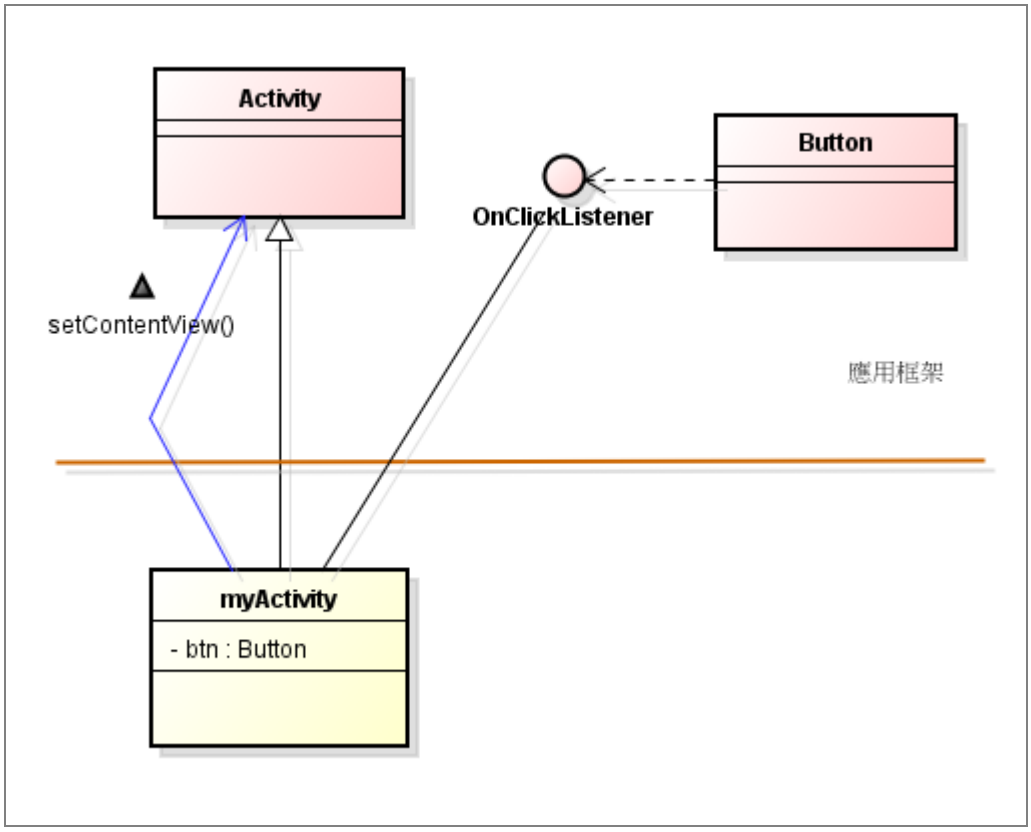
这个函数是定义于 Activity 幕后的父类别里，被 Activity 继承下来。而 myActivity 就调用 Activity 里的这个函数，所以画了一个箭头来表示 myActivity 类别调用了 Activity 基类的 setContentView()函数(如上图)。

此时，Button 对象诞生完毕了，该 Button 对象又包装于 Layout 对象里，然后将其参考值传递给框架的 Activity 幕后的 WindowManagerService 和 SurfaceFlinger 系统服务。这些服务依据 Button 对象属性值，决定布局里的按钮位置、大小、颜色等，并显示于屏幕上。

由于诞生完毕，为了类别图美观起见，可以把右下角的代表「诞生」的虚线删除掉。于是，可选取该虚线，并按右键，出现如下：



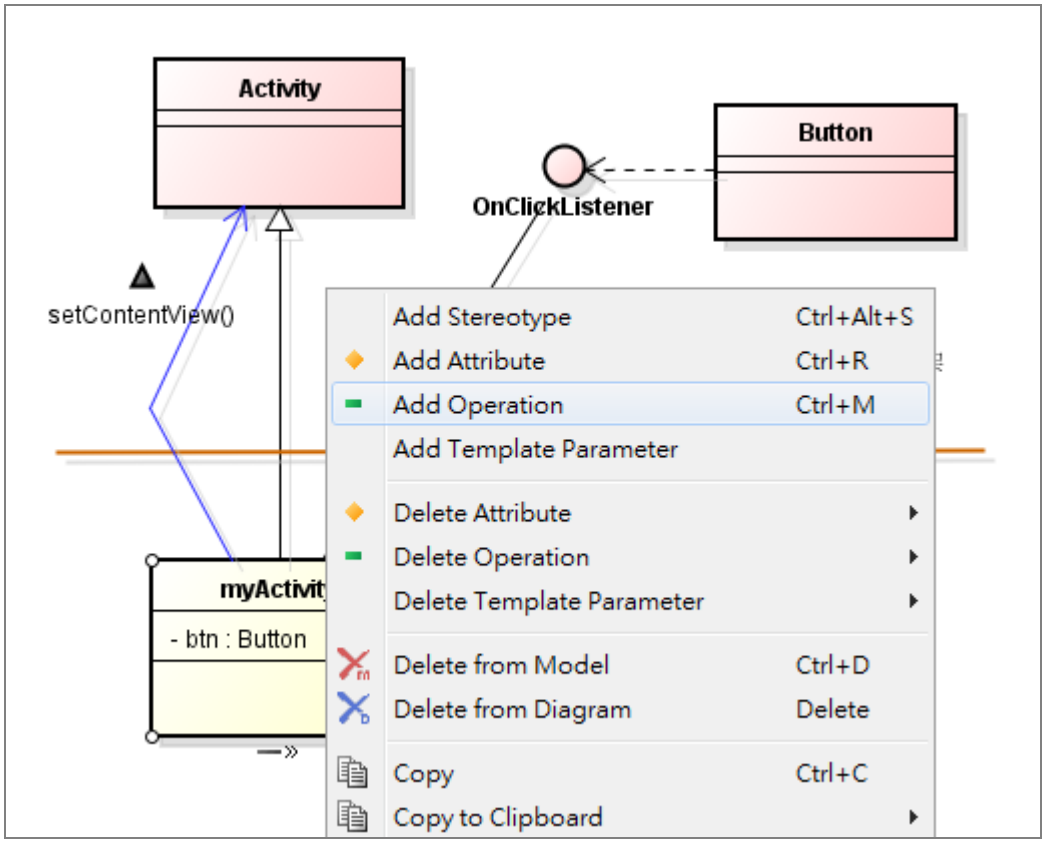

选取<Delete from Model>，就能删除之。如下图：



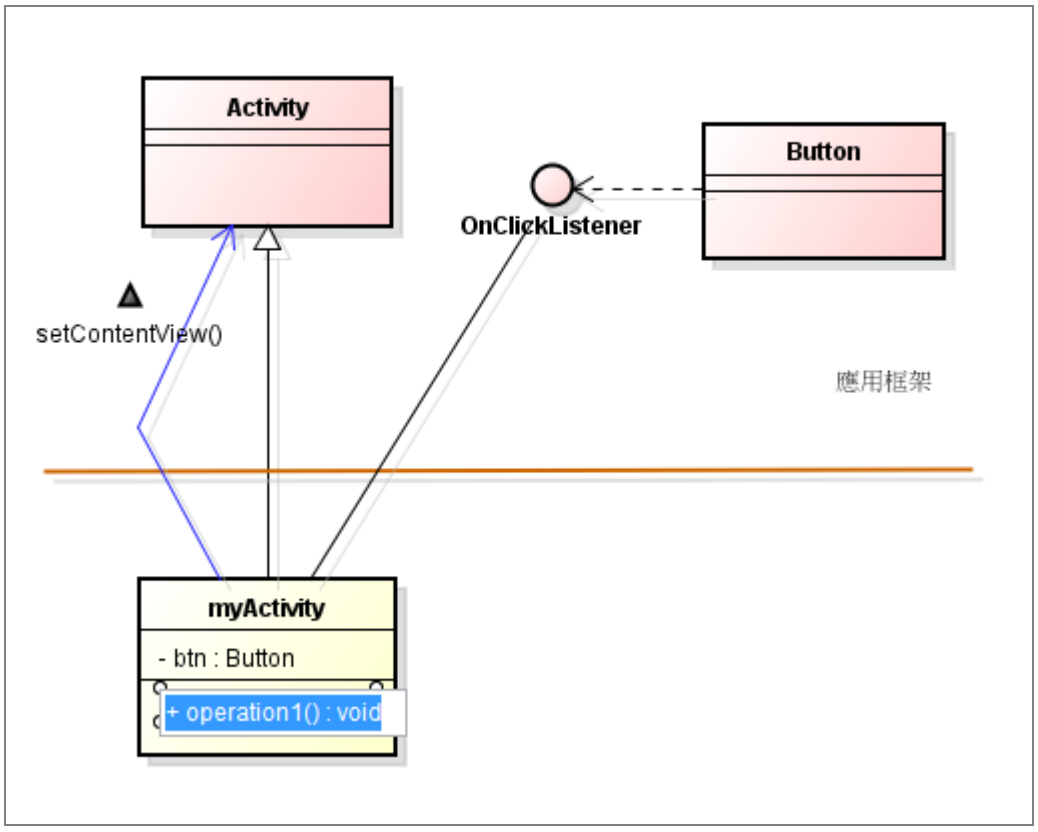

**Step-4：** WindowManager/SurfaceFlinger 服务将屏幕画面的按钮显示出来，等待人们去按它。当人们按下这按钮时，WindowManagerService 会将此按键事件回传给 Button 类别，接着 Button 类别就透过 OnClickListener 接口而调用到 myActivity 的 onClick()函数。于是：

- **步骤 4.1**

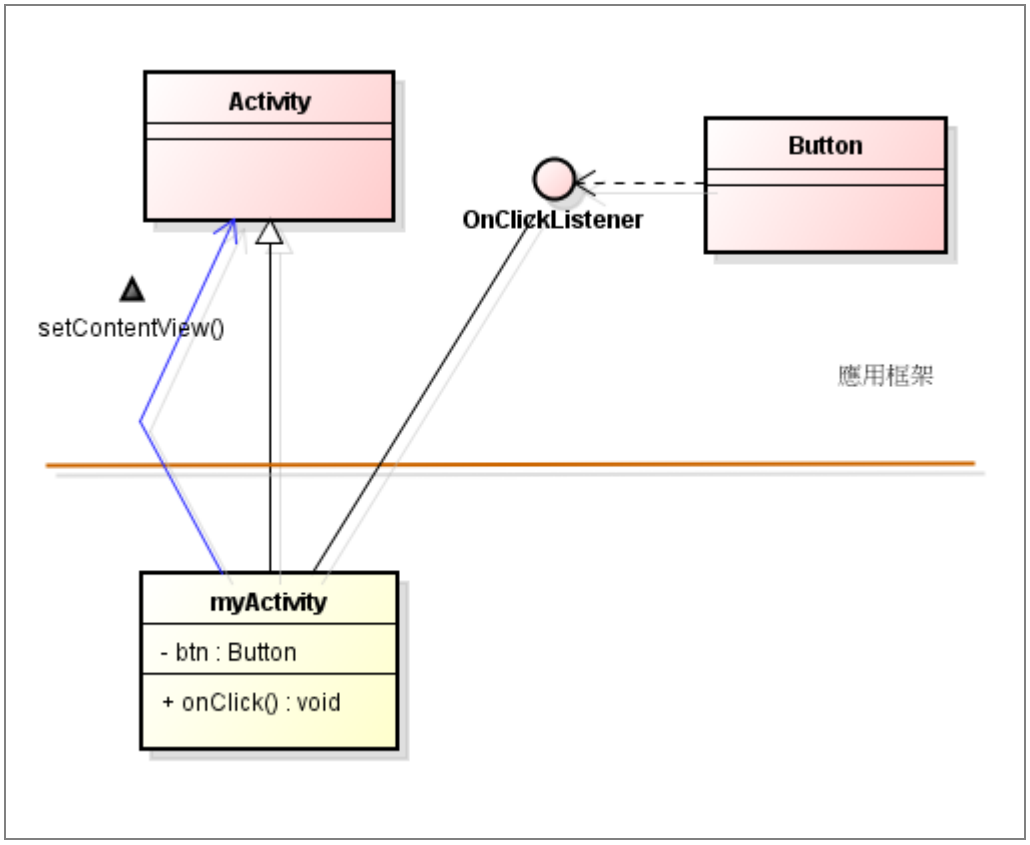
就来替 myActivity 类别增添一个 onClick()函数。就点选 myActivity 类别，并按右键，出现如下：




选取<Add Operation>来增添一个新函数，就出现了：

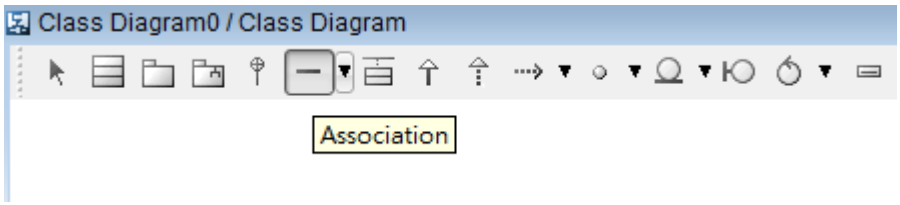



替函数取个名称：onClick()；如下图：




- **步驟 4.2**

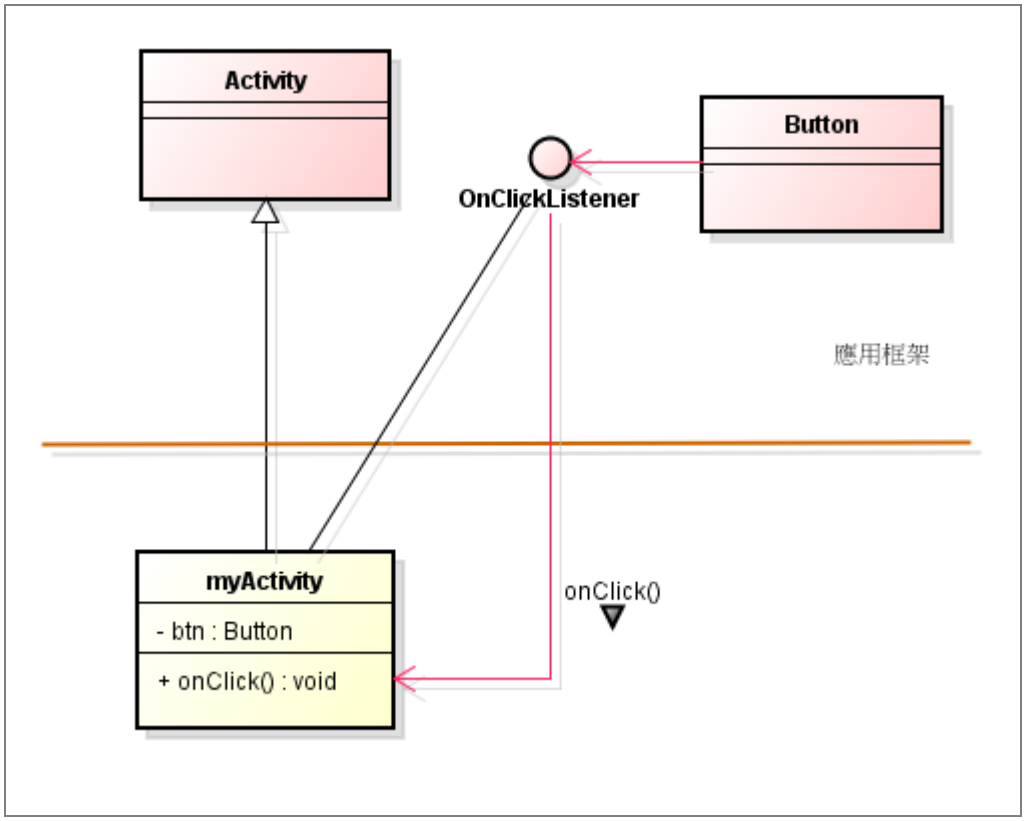
删除掉 Button 类别与 OnClickListener 接口之间的虚线箭头。然后，选取<Association>图素，如下：




拉出一条 Button 类别到 OnClickListener 之间的实线箭头(如下图)。

- **步驟 4.3**

再拉出一条 OnClickListener 接口到 myActivity 之间的实线箭头，并标示出调用 onClick()的含意。如下：




这表达出：当人们按下屏幕画面上的按钮时，Android 框架(及其幕后的系统服务)会透过 OnClickListener 接口而调用了 myActivity 类别里的 onClick()函数。◆

~~ Continued ~~