使用單位：技術處 姓名：

技術：如何收集自訂錯誤報表(ACRA) 開始時間：

自訂類別畫面物件教學查核表

說明：此文件用來指導如何在Android app上面收集來自使用者的錯誤報表，做為日後修正App錯誤的依據。。

* 有關於此教學查核表學習前的必要條件：
1. 對於軟體程式寫作已經有初步的了解。
2. 已經對於Android JAVA撰寫有一定的能力。
* 有關於本查核表的最終有價值產品：

如何在Android中建立一個自訂畫面物件，作為日後重複使用。

* 查核表完成的時間：2個小時以內。
* 注意事項： 請依照步驟進行，不可以任意跳過未完成的步驟。若該步驟是關於查核或是時做練習，必須要將作業(成品)，交給主管進行查核，該項目請由主管簽名。

**執行步驟**

第一部分：基本概念

|  |  |  |
| --- | --- | --- |
| 1. | 字的定義：甚麼是類別(Class)以下定義出自維基百科：http://zh.wikipedia.org/wiki/%E7%B1%BB\_(%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6)**類別**（class），是一種[物件導向](http://zh.wikipedia.org/wiki/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1)計算機程式語言的構造，是創建物件的藍圖，描述了所創建的物件(大陸稱為對象)共同的[屬性](http://zh.wikipedia.org/w/index.php?title=%E5%B1%9E%E6%80%A7(%E8%AE%A1%E7%AE%97%E6%9C%BA)&action=edit&redlink=1)和[方法](http://zh.wikipedia.org/w/index.php?title=%E6%96%B9%E6%B3%95(%E8%AE%A1%E7%AE%97%E6%9C%BA)&action=edit&redlink=1)。  |   |
| 2. | 字的定義：甚麼是「物件(Object)」**物件**（object），中國大陸譯為**對象**，是[物件導向](http://zh.wikipedia.org/wiki/%E7%89%A9%E4%BB%B6%E5%B0%8E%E5%90%91%E7%A8%8B%E5%BC%8F%E8%A8%AD%E8%A8%88)(Object Oriented) 中的術語，既表示客觀世界[問題空間](http://zh.wikipedia.org/wiki/%E9%97%AE%E9%A2%98%E7%A9%BA%E9%97%B4)(Namespace)中的某個具體的事物，又表示軟體系統[解空間](http://zh.wikipedia.org/w/index.php?title=%E8%A7%A3%E7%A9%BA%E9%97%B4&action=edit&redlink=1)中的基本元素。在軟體系統中，物件具有唯一的[標識符](http://zh.wikipedia.org/w/index.php?title=%E6%A0%87%E8%AF%86%E7%AC%A6&action=edit&redlink=1" \o "標識符（頁面不存在）)，物件包括[屬性](http://zh.wikipedia.org/w/index.php?title=%E5%B1%9E%E6%80%A7&action=edit&redlink=1)(Properties)和[方法](http://zh.wikipedia.org/wiki/%E6%96%B9%E6%B3%95_%28%E9%9B%BB%E8%85%A6%E7%A7%91%E5%AD%B8%29%22%20%5Co%20%22%E6%96%B9%E6%B3%95%20%28%E9%9B%BB%E8%85%A6%E7%A7%91%E5%AD%B8%29)(Methods)，屬性就是需要記憶的信息，方法就是物件能夠提供的服務。在物件導向(Object Oriented) 的軟體中，物件(Object)是某一個[類](http://zh.wikipedia.org/wiki/%E7%B1%BB_%28%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6%29%22%20%5Co%20%22%E9%A1%9E%20%28%E8%A8%88%E7%AE%97%E6%A9%9F%E7%A7%91%E5%AD%B8%29)(Class)的[實例](http://zh.wikipedia.org/wiki/%E5%AE%9E%E4%BE%8B%22%20%5Co%20%22%E5%AF%A6%E4%BE%8B)(Instance)。 |  |
| 3. | 說明：類別的好處物件(大陸稱為對象)提供了模型化和信息隱藏的好處。類別提供了[可重用性](http://zh.wikipedia.org/w/index.php?title=%E5%8F%AF%E9%87%8D%E7%94%A8%E6%80%A7&action=edit&redlink=1)的好處。自行車製造商一遍一遍地重用相同的藍圖來製造大量的自行車。軟體程式設計師用相同的類別，即相同的代碼一遍一遍地建立物件(大陸稱為對象)。 |  |

第二部分：設計範例

|  |  |  |
| --- | --- | --- |
| 1. | 設計概念：本查核表的設計概念為建立一個繼承RelativeLayout的自訂物件類別，此類別將被用在通訊錄的列表中。以下是這個類別的畫面示意圖。C:\Users\Louis\Desktop\自訂物件範例.png |   |
| 2. | 設計說明：在這個RelativeLayout中，有五個物件，分別是：1. 大頭照：ImageView, ID設定為contactPhoto
2. 姓名：TextView, ID設定為contactName
3. 手機：TextView, ID設定為contactMobile
4. 地址：TextView, ID設定為contactAddress.
5. 圖示：ImageView, ID設定為contactLinkIcon
 |   |
| 3. | 實作：設定一個新的專案，請將專案名稱設定為CustomClassTest 這個專案是用來測試如何進行自訂畫面物件的設計。 |   |
| 4. | 準備：下載測試所需要的圖片檔案。請至以下網址下載測試用所需要用到的檔案：http://www.v7idea.com.tw/download/testAndroidCustonObj.zip |   |
| 5 | 實作：請將步驟4所下載的zip檔案解開後，會有兩個圖片檔案(contact\_photo.png與custom\_link\_icon.png)。將這兩個物件檔案匯入/res/drawable/中。如果沒有/res/drawable的目錄，請自行新增一個。 |   |
| 6 | 實作：設計Layout，請新增一個Layout檔案，檔名取為single\_contact.xml使用RelativeLayout作為底部的Layout。 |   |
| 7 | 實作：請參考步驟1的圖示，設計出這個Layout檔案,並且在放入每個物件時，依照步驟2的說明命名物件。 |   |
| 8 | 參考：single\_contact.xml參考範例：<RelativeLayout xmlns:android=*"http://schemas.android.com/apk/res/android"* android:layout\_width=*"fill\_parent"* android:layout\_height=*"fill\_parent"* > <ImageView android:id=*"@+id/contactPhoto"* android:layout\_width=*"50sp"* android:layout\_height=*"50dp"* android:layout\_alignParentLeft=*"true"* android:layout\_centerVertical=*"true"* android:layout\_marginLeft=*"5dp"* android:layout\_marginRight=*"5dp"* android:src=*"@drawable/contact\_photo"* /> <TextView android:id=*"@+id/contactName"* android:layout\_width=*"wrap\_content"* android:layout\_height=*"wrap\_content"* android:layout\_alignTop=*"@+id/contactPhoto"* android:layout\_toRightOf=*"@+id/contactPhoto"* android:text=*"莊志偉"* android:textSize=*"20dp"* /> <TextView android:id=*"@+id/contactAddress"* android:layout\_width=*"wrap\_content"* android:layout\_height=*"wrap\_content"* android:layout\_alignBottom=*"@+id/contactPhoto"* android:layout\_alignLeft=*"@+id/contactName"* android:layout\_below=*"@+id/contactName"* android:layout\_marginRight=*"10dp"* android:layout\_toLeftOf=*"@+id/callOutIcon"* android:text=*"105台北市南京東路五段97號2樓"* android:textSize=*"14dp"* /> <TextView android:id=*"@+id/contactMobile"* android:layout\_width=*"wrap\_content"* android:layout\_height=*"wrap\_content"* android:layout\_alignTop=*"@+id/contactName"* android:layout\_marginRight=*"10dp"* android:layout\_toLeftOf=*"@+id/callOutIcon"* android:text=*"0938030153"* android:textSize=*"20dp"* /> <ImageView android:id=*"@+id/callOutIcon"* android:layout\_width=*"30dp"* android:layout\_height=*"30dp"* android:layout\_alignParentRight=*"true"* android:layout\_alignTop=*"@+id/contactMobile"* android:layout\_marginRight=*"10dp"* android:src=*"@drawable/custom\_link\_icon"* /> </RelativeLayout> |   |
| 9 | 實作：新增加一個Layout檔案，取名為contact\_list.xml |   |
| 10 | 實作：在contact\_list中放入一個RelativeLayout，並且放入一個ScrollView物件，並將ID設定為contactScroll。在這個ScrollView物件放入LinearLayout物件，並且將ID設定為：contactListArea. |   |
| 11 | 步驟10的範例：<RelativeLayout xmlns:android=*"http://schemas.android.com/apk/res/android"* android:layout\_width=*"fill\_parent"* android:layout\_height=*"fill\_parent"* ><ScrollView android:id=*"@+id/contactScroll"* android:layout\_width=*"match\_parent"* android:layout\_height=*"match\_parent"* android:layout\_alignParentLeft=*"true"* android:layout\_alignParentTop=*"true"* ><LinearLayout android:id=*"@+id/contactListArea"* android:layout\_width=*"match\_parent"* android:layout\_height=*"match\_parent"* > </LinearLayout> </ScrollView> |   |

第三部分：程式碼的實作撰寫

|  |  |  |
| --- | --- | --- |
| 1. | 實作：新增加一個Class File，取名為SingleContactObj.jar請在/src的目錄下，自己專案的套件中按下滑鼠右鍵，選擇「新建(new)」->「類別(class)」將名稱命名為「SingleContactObj」 |   |
| 2. | 實作：設定這個SingleContactObj繼承RelativeLayout。 範例： **public** **class** SingleContactObj **extends** RelativeLayout { } |   |
| 3. | 實作：增加一個新的建構字，並且帶入Context context, String contactName, String contactPhone, String contactAddress, Intent callOutIntent的參數。 範例： **public** **class** SingleContactObj **extends** RelativeLayout {*public SingleContactObj(Context context, String contactName, String contactPhone, String contactAddress, Intent callOutIntent) {* *super(context);* *// TODO Auto-generated constructor stub* *}* } |   |
| 4. | 實作：實作這個建構子* 建立一個RelativeLayout，並且指定Layout檔案：R.layout.*single\_contact*
* 將這個Layout中的所有物件找出來，並定義變數指向這些物件。
* 分別將Name, Address, MobileName指向三個文字物件。
* 設定Intent
* 設定右邊Icon按下後的反應，呼叫帶入的Intent;
* 設定這個Layout的Param
* 將這個RelativeLayout放入建構子的物件中
 |   |
| 5 | 參考範例：步驟4可以參考範例。以下是建構子的內容**public** SingleContactObj(Context context, String contactName, String contactPhone, String contactAddress, Intent callOutIntent) { **super**(context); // **TODO** Auto-generated constructor stub thisContext = context; thisContactName = contactName; thisContactPhone = contactPhone; thisContactAddress = contactAddress; thisCallOutIntent = callOutIntent;  RelativeLayout thisObj = (RelativeLayout) ((Activity) context).getLayoutInflater().inflate(R.layout.*single\_contact*, **null**);  contactPhoto = (ImageView) thisObj.findViewById(R.id.*contactPhoto*); callOutIcon = (ImageView) thisObj.findViewById(R.id.*callOutIcon*); contactNameText = (TextView)thisObj.findViewById(R.id.*contactName*); contactAddressText = (TextView)thisObj.findViewById(R.id.*contactAddress*); contactPhoneText = (TextView)thisObj.findViewById(R.id.*contactMobile*);  contactNameText.setText(thisContactName); contactAddressText.setText(thisContactAddress); contactPhoneText.setText(thisContactPhone);  callOutIcon.setOnClickListener(**new** OnClickListener() { @Override **public** **void** onClick(View arg0) { // **TODO** Auto-generated method stub **if**(thisCallOutIntent != **null**) {  thisContext.startActivity(thisCallOutIntent);  }  }});  RelativeLayout.LayoutParams thisParam = **new** RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.*MATCH\_PARENT*, RelativeLayout.LayoutParams.*MATCH\_PARENT*); thisObj.setLayoutParams(thisParam);  **this**.addView(thisObj);} |   |
| 6 | 實作：主畫面的Activity檔案設計。指定contact\_list Layout實作5個SingleContactObj物件 |   |
| 7 | 參考範例：步驟6可以參考範例。以下是MainActivity.jar (主要的測試Activity)**public** **class** MainActivity **extends** Activity { @Override **public** **void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState); // setContentView(R.layout.activity\_main);  RelativeLayout mainLayout = (RelativeLayout) getLayoutInflater().inflate(R.layout.*contact\_list*, **null**); LinearLayout contentArea = (LinearLayout) mainLayout.findViewById(R.id.*contactListArea*);  contentArea.setOrientation(LinearLayout.*VERTICAL*);  Intent user1Intent = **new** Intent ("android.intent.action.CALL", Uri.*parse*("tel:0938030153")); SingleContactObj user1 = **new** SingleContactObj(**this**, "莊志偉", "0938030153", "台北市南京東路五段97號2F", user1Intent );  Intent user21Intent = **new** Intent ("android.intent.action.CALL", Uri.*parse*("tel:0938030153")); SingleContactObj user2 = **new** SingleContactObj(**this**, "聯絡人1", "0938030153", "住址1", user1Intent );  Intent user3Intent = **new** Intent ("android.intent.action.CALL", Uri.*parse*("tel:0938030153")); SingleContactObj user3 = **new** SingleContactObj(**this**, "聯絡人2", "0938030153", "住址2", user1Intent );  Intent user4Intent = **new** Intent ("android.intent.action.CALL", Uri.*parse*("tel:0938030153")); SingleContactObj user4 = **new** SingleContactObj(**this**, "聯絡人3", "0938030153", "住址3", user1Intent );  Intent user5Intent = **new** Intent ("android.intent.action.CALL", Uri.*parse*("tel:0938030153")); SingleContactObj user5 = **new** SingleContactObj(**this**, "聯絡人4", "0938030153", "住址4", user1Intent ); contentArea.addView(user1); contentArea.addView(user2); contentArea.addView(user3); contentArea.addView(user4); contentArea.addView(user5); setContentView(mainLayout); } @Override **public** **boolean** onCreateOptionsMenu(Menu menu) { getMenuInflater().inflate(R.menu.*activity\_main*, menu); **return** **true**; }} |   |

第四部分：延伸練習

|  |  |  |
| --- | --- | --- |
| 1. | 實作：修改SingleContactObj所使用的Layout，想辦法製作出分隔線 |   |
| 2. | 實作：修改列表中每個聯絡人的大頭照，使其可以顯示不同的頭像。(可以的話可以使用http去下載圖檔。下載圖檔的Method:**public** Bitmap getBitmapByUrl(String imageUrl) { Bitmap imageBitmap = **null**; URL thisImageURL = **null**;  HttpURLConnection connection;**try** { thisImageURL = **new** URL(imageUrl); connection = (HttpURLConnection)thisImageURL .openConnection(); connection.setDoInput(**true**); connection.connect(); InputStream input = connection.getInputStream(); imageBitmap = BitmapFactory.*decodeStream*(input);} **catch** (MalformedURLException e) { // **TODO** Auto-generated catch block e.printStackTrace();} **catch** (IOException e) { // **TODO** Auto-generated catch block e.printStackTrace();}**return** imageBitmap;}  |   |