LẬP TRÌNH DI ĐỘNG

Bài 4: một số loại widget thông dụng và cách kĩ thuật xử lý sự kiện trên widget

Nhắc lại bài trước

- Khái niệm: View/Widget/ViewGroup/Layout
- Layout:
  - Là ViewGroup, có cách bố trí các view con bên trong riêng biệt, là đặc trưng của từng loại layout
  - Nạp từ XML hoặc bằng code
  - LayoutParameters quyết định vị trí đặt view con
- Các loại layout:
  - LinearLayout: view con nằm thành hàng dọc hoặc ngang
  - RelativeLayout: các view con phụ thuộc lẫn nhau
  - FrameLayout: các view con chồng lên nhau ở vị trí (0,0)

Nhắc lại bài trước

- Các loại layout:
  - ScrollView & HorizontalScrollView: view cha hỗ trợ thêm thanh trượt nếu view con quá lớn
  - TableLayout: chia màn hình theo lưới
- Widget: TextView / EditText / Button / ImageButton / TonggleButton / CheckBox / RadioGroup & RadioButton
- Cách tương tác với view qua findViewById và các phương thức của view

Nội dung

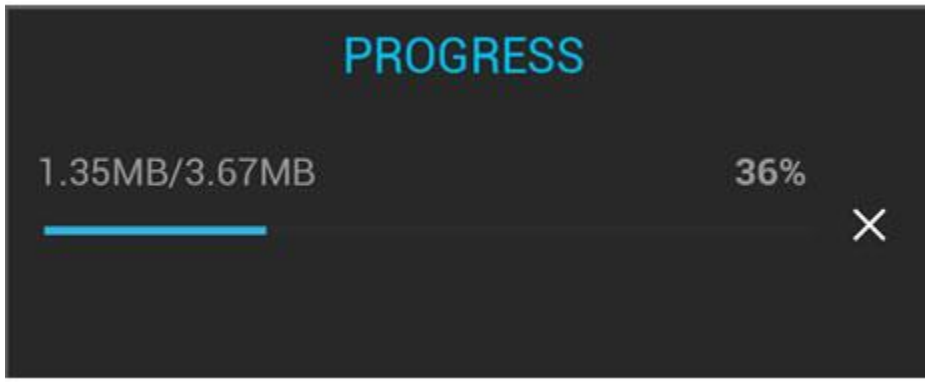
1. ProgressBar & ProgressDialog
   - Hiển thị ProgressDialog
   - Tiến trình chạy ngầm cập nhật dữ liệu lên ProgressBar
2. AutoComplete TextView
3. TimePicker/DatePicker
4. ListView
5. Spinner
6. WebView
7. Vài phương pháp xử lý sự kiện

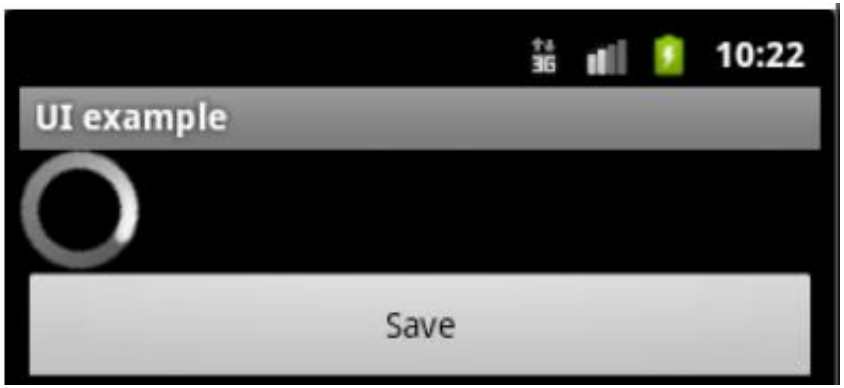
Phần 1

ProgressBar & ProgressDialog

ProgressBar

```
<LinearLayout
    android:orientation="horizontal"
    ... >
    <ProgressBar
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        style="@android:style/Widget.ProgressBar.Small"
        android:layout_marginRight="5dp" />
```





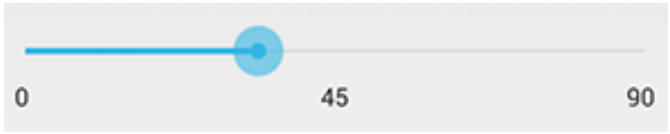
ProgressBar

- ProcessBar cung cấp một thông tin phản hồi trực quan về một tác vụ đang thực thi
  - VD: thể hiện phần trăm dữ liệu đã xử lý được
- Mặc định ProcessBar ở chế độ Indeterminate: thể hiện hiệu ứng xoay tròn vô tận
  - Thích hợp cho những tác vụ không xác định rõ được khi nào nó sẽ hoàn tất
- Chuyển ProcessBar sang dạng thanh ngang bằng các thêm đoạn thuộc tính sau vào XML

```
style="@android:style/Widget.ProgressBar.Horizontal"
```

SeekBar và RatingBar

- SeekBar và RatingBar là mở rộng của ProgressBar
- RatingBar hiển thị giá trị (số thực) bởi các dấu sao
- SeekBar cho chọn giá trị trực tiếp trên thanh trượt
  - Muốn xử lý khi giá trị trên thanh SeekBar thay đổi: dùng SeekBar.OnSeekBarChangeListener
- Một số thuộc tính thường dùng



  - "android:max": giá trị tối đa của ProgressBar
  - "android:progress": giá trị hiện tại của ProgressBar
  - "android:minHeight": chiều cao của ProgressBar
  - "android:indeterminate": đặt chế độ vô tận

Hiển thị một ProgressDialog

ProgressDialog: cửa sổ hiện thị một ProgressBar, đáp ứng sự kiện hủy bằng nút back

```
pb = new ProgressDialog(context);
pb.setCancelable(true);
pb.setMessage("File downloading...");
pb.setProgressStyle(
    ProgressDialog.STYLE_HORIZONTAL);
pb.setProgress(0);
pb.setMax(100);
pb.show();
```

Chạy ngầm và cập nhật progress

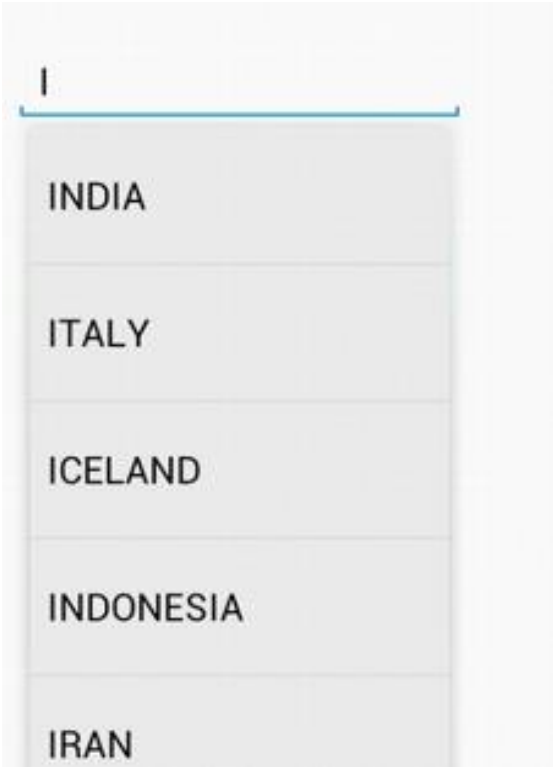
```
new Thread(new Runnable() {
    public void run() {
        for (int i = 0; i < 100; i++) {
            doSomethingHere();  // thực hiện gì đó
            myHandler.post(new Runnable() {
                public void run() {
                    myProgress.setProgress(myPercent++);
                }
            });
        }
    }
}).start();
```

Phần 2

AutoComplete TextView

AutoComplete TextView

```
<AutoCompleteTextView
 android:id="@+id/autoCompleteNS"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:completionThreshold="1"
 android:layout_span="2"
 android:ems="10" />
```



AutoComplete TextView

- AutoComplete là một loại view tương tự EditText
- Có khả năng đưa ra một danh sách các gợi ý tương tự như phần text mà người dùng nhập vào
- Một số chú ý sử dụng:
  - Thuộc tính “android:completionThreshold”: số kí tự tối thiểu để bắt đầu hiện các gợi ý (nếu code thì dùng setThreshold)
  - Dùng setAdapter để thiết lập các string gợi ý cho người nhập liệu (xem ví dụ ở trang sau)

AutoComplete TextView

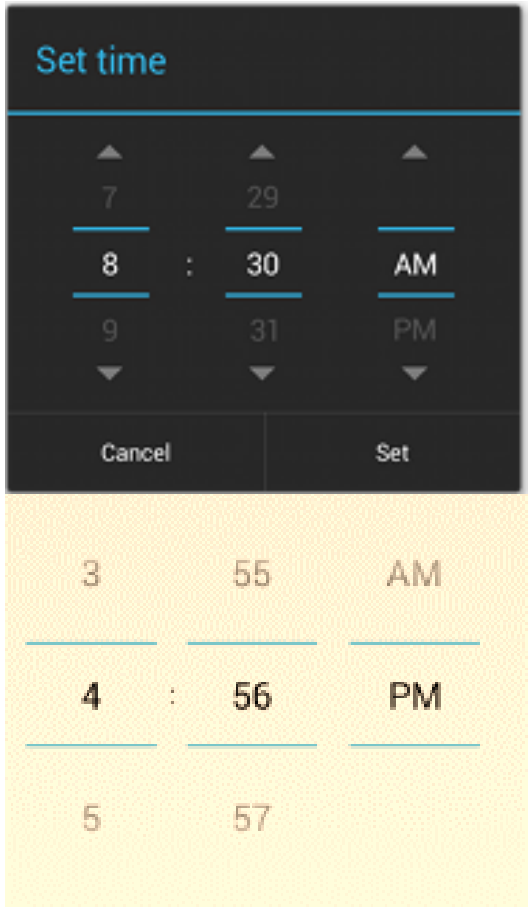
```
public class CountriesActivity extends Activity {
  protected void onCreate(Bundle icicle) {
    super.onCreate(icicle);
    setContentView(R.layout.countries);
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
        android.R.layout.simple_dropdown_item_1line, COUNTRIES);
    AutoCompleteTextView textView = (AutoCompleteTextView)
        findViewById(R.id.countries_list);
    textView.setAdapter(adapter);
  }
  static final String[] COUNTRIES = new String[] {
    "Belgium", "France", "Italy", "Germany", "Spain"
  };
}
```

Phần 3

TimePicker/DatePicker

TimePicker & TimePickerDialog

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/
android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>
<TimePicker
android:layout_width="wrap_content"
android:layout_height="wrap_content"/>
</LinearLayout>
```



TimePicker & TimePickerDialog

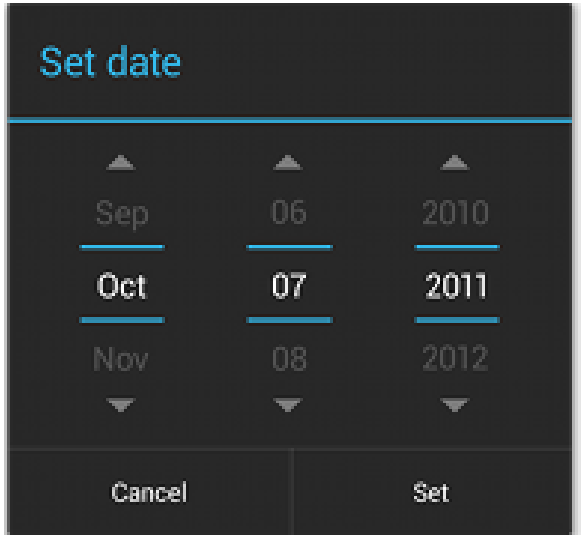
- TimePicker là view (ViewGroup) cho phép người sử dụng chọn thời gian trong một ngày
- Hàm setCurrentHour(h) và setCurrentMinute(m) để chỉnh thời gian hiển thị
- Hàm getCurrentHour() và getCurrentMinute() để lấy thời gian do người dùng nhập
- Không có cách thiết lập thời gian bằng design
- TimePickerDialog hiển thị dialog cho phép thiết lập một giờ ban đầu và người dùng hiệu chỉnh sau

DatePicker

- DatePicker cho phép người dùng chọn ngày
- Cơ chế hoạt động tương tự TimePicker
- DatePickerDialog hiển thị dialog nhập ngày

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/
android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>

<DatePicker
android:layout_width="wrap_content"
android:layout_height="wrap_content"/>
 </LinearLayout>
```



OnDateSetListener

```
final Calendar c = Calendar.getInstance();
mYear = c.get(Calendar.YEAR);
mMonth = c.get(Calendar.MONTH);
mDay = c.get(Calendar.DAY_OF_MONTH);

DatePickerDialog dpd = new DatePickerDialog(this,
  new DatePickerDialog.OnDateSetListener() {
    @Override
    public void onDateSet(DatePicker v, int y, int m, int d) {
      txtDate.setText(d + "-" + (m + 1) + "-" + y);
    }
  }, mYear, mMonth, mDay);
dpd.show();
```

OnTimeSetListener

```
TimePickerDialog tpd = new TimePickerDialog(this,
  new TimePickerDialog.OnTimeSetListener() {
    @Override
    public void onTimeSet(TimePicker view, int hour,
                                           int minute) {
      txtTime.setText(hour + ":" + minute);
    }
  }, mHour, mMinute, false);
tpd.show();
```

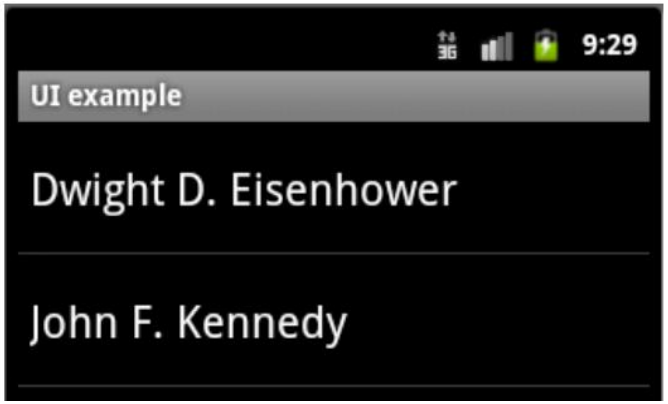
Phần 4

ListView

ListView

ListView hiển thị một danh sách các phần tử trên một giao diện cho phép cuộn theo chiều dọc

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="vertical"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      >
      <ListViewandroid:id="@+id/android:list"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"/>
</LinearLayout>
```



ListView: thiết lập nội dung

```
// mỗi string là 1 dòng trong ListView
String[] values = new String[] {
    "Việt Nam", "Trung Quốc", "Triều Tiên", "Cuba", "Hoa Kỳ"};

// Adapter: chứa dữ liệu và view của 1 dòng
//  1. Context
//  2. Layout cho 1 dòng
//  3. ID của TextView để hiện dữ liệu
//  4. Mảng các dữ liệu
ArrayAdapter<String> adapter = new ArrayAdapter<String>(
    this, android.R.layout.simple_list_item_1,
    android.R.id.text1, values);
listView = (ListView) findViewById(R.id.list);
listView.setAdapter(adapter);
```

ListView: xử lý sự kiện

```
listView.setOnItemClickListener(new OnItemClickListener() {
   @Override
   public void onItemClick(
      AdapterView<?> parent, View view, int pos, long id) {
      String val = (String) listView.getItemAtPosition(pos);

      Toast.makeText(
         this,
         val + "(vị trí: " + pos + ")",
         Toast.LENGTH_LONG
      ).show();
   }
});
```

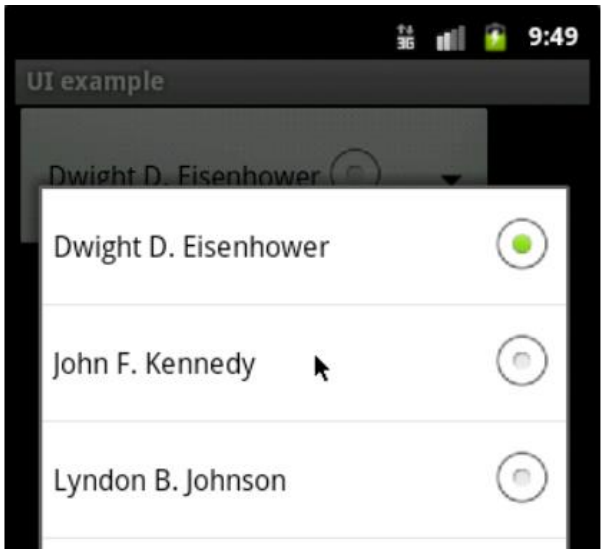
Phần 5

SpinnerView

Spinner View

Spinner View là loại view được dùng để hiển thị một danh mục cho phép người dùng lựa chọn

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/
android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>
<Spinner
android:id="@+id/spinner1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:drawSelectorOnTop="true"/>
</LinearLayout>
```



Spinner View

```
public class SpinnerExampleActivity extends Activity {

  String[] presidents = {
"Dwight D. Eisenhower", "Barack Obama"      };
  Spinner s1;
  @Override
  public void onCreate(Bundle savedInstanceState)
  {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.spinner);
  s1 = (Spinner) findViewById(R.id.spinner1);
  ArrayAdapter<String> adapter = new
        ArrayAdapter<String>(this.
        android.R.layout.simple_spinner_dropdown_item , presidents);
  s1.setAdapter(adapter);
  s1.setOnItemSelectedListener(newOnItemSelectedListener(){
        @Override
        public void onItemSelected(AdapterView<?> arg0, View arg1,
        int arg2, long arg3)
         { |
                int index = s1.getSelectedItemPosition();
               Toast.makeText(getBaseContext(),"You have selected item :
              " + presidents[index], Toast.LENGTH_SHORT).show();
         }
        public void onNothingSelected(AdapterView<?> arg0)
        {

        }
  });
  }

}
```

Phần 6

WebView

WebView

- Là view hiển thị nội dung HTML, thực hiện mã js và một số loại dữ liệu online
- XML file:

```
<WebView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/webview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" />
```

- Cấp quyền sử dụng Internet trong manifest file

```
<uses-permission android:name="android.permission.INTERNET" />
```

WebView

```
public class WebActivity extends Activity{
     /** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState)
    {
super.onCreate(savedInstanceState);
        setContentView(R.layout.webview);

        WebView wv = (WebView) findViewById(R.id.webview1);
        wv.loadUrl("http://mobipro.vn");
    }
}
```

WebView

```
WebView wv = (WebView) findViewById(R.id.webview1);

final String mimeType = "text/html";
final String encoding = "UTF-8";
String html = "<H1>Đây là một trang web đơn giản</H1><body>" +
"<p>Bạn có thể tạo ra nội dung HTML bằng cách này</p>";
```

Và nạp lên WebView như sau:

```
wv.loadDataWithBaseURL("", html, mimeType, encoding, "");
```

```
WebView wv = (WebView) findViewById(R.id.webview1);
wv.loadUrl("file:///android_asset/Index.html");
```

WebView

- Bật hỗ trợ javascript:

```
myWebView.getSettings().setJavaScriptEnabled(true);
```

- Có thể từ javascript gọi các method của ứng dụng: Xem thêm addJavascriptInterface
- WebView bản thân nó chỉ là một view để hiển thị trang web, không phải là browser, vì thế khi tác động vào trang web, android sẽ xử lý mặc định, muốn xử lý việc này xem thêm WebViewClient
- Cần chú ý một số phím mặc định trong trường hợp WebView được focus (chẳng hạn phím back)

Phần 7

Vài phương pháp xử lý sự kiện

Sự kiện trong android

- Sự kiện được tạo ra để đáp ứng với một hành động do tác động từ bên ngoài, thường là tương tác bởi người dùng
- Có 4 cách chính để cài đặt sự kiện trong lập trình Android như:
  - Cài đặt hàm xử lý trong class Activity và khai báo sử dụng ngoài layout
  - Cài đặt động thông qua các hàm setListener trong class Activity
  - Implements Listener trong khai báo lớp
  - Dùng biến lưu trữ

C1: Onclick in XML

TRONG FILE CODE JAVA

```
public void OnButtonClick(View v)
{
   TextView tv1=(TextView)findViewById(R.id.textview1);
   tv1.setText("Click 1");
}
```

```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click 1"
    android:onClick="OnButtonClick"
    />
```

TRONG FILE LAYOUT XML

C2: Inline anonymous listener

Sự kiện đó sẽ được cài đặt khi Activity chứa View đó được tạo ra, điều đó cũng có nghĩa mình sẽ cài việc đó vào hàm OnCreate của Activity đó.

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    Button b1=(Button)findViewById(R.id.button1); //tìm Button mang id là button1
    b1.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View v) {
                    TextView tv1=(TextView)findViewById(R.id.textview1);
                    tv1.setText("Click 1");
                }
            });
}
```

C3: Activity is listener

```
public class ClickActivity extends Activity implements OnClickListener {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button b1=(Button)findViewById(R.id.button1);
        b1.setOnClickListener(this);
    }

       @Override
       public void onClick(View v) {
              // TODO Auto-generated method stub
              TextView tv1=(TextView)findViewById(R.id.textview1);
              tv1.setText("Click 1");
       }
}
```

C4: Listener in variable

```
public class ClickActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button b1=(Button)findViewById(R.id.button1);
        b1.setOnClickListener(buttonClickListener);
    }

    OnClickListener buttonClickListener = new OnClickListener()      {
        @Override
        public void onClick(View v) {
              TextView tv1=(TextView)findViewById(R.id.textview1);
              tv1.setText("Click 1");
        }
    };
}
```

Một số kĩ thuật ít thông dụng hơn

- Cách 5: Explicit listener class
  - Viết class độc lập xử lý sự kiện
  - Class có thể là inner class hoặc hoàn toàn độc lập
- Cách 6: Anonymous view subclassing

```
layout.addView(new Button(this) {
    @Override
    public boolean performClick() {
        // đoạn mã xử lý nút bấm
        …
        return false;
    }
});
```

Xử lý sự kiện: thảo luận

- Trong 6 cách trên bạn thích cách nào nhất? Lý do?
- Hãy chỉ ra ưu điểm, nhược điểm của các cách
- Khuyến cáo:
  - Nên sử dụng cách 1 nếu có thể
  - Xử lý sự kiện không được quá lâu, nếu hàm xử lý có thể cần quá 500ms để xử lý, hãy chuyển thành background thread hoặc tương đương
  - Nên tuân theo những tiêu chuẩn về xử lý sự kiện, tránh tạo ra những trải nghiệm khác lạ (người dùng có thể hiểu nhầm thành lỗi của ứng dụng)