LẬP TRÌNH DI ĐỘNG

Bài 5: Ứng Dụng Android Đơn Giản

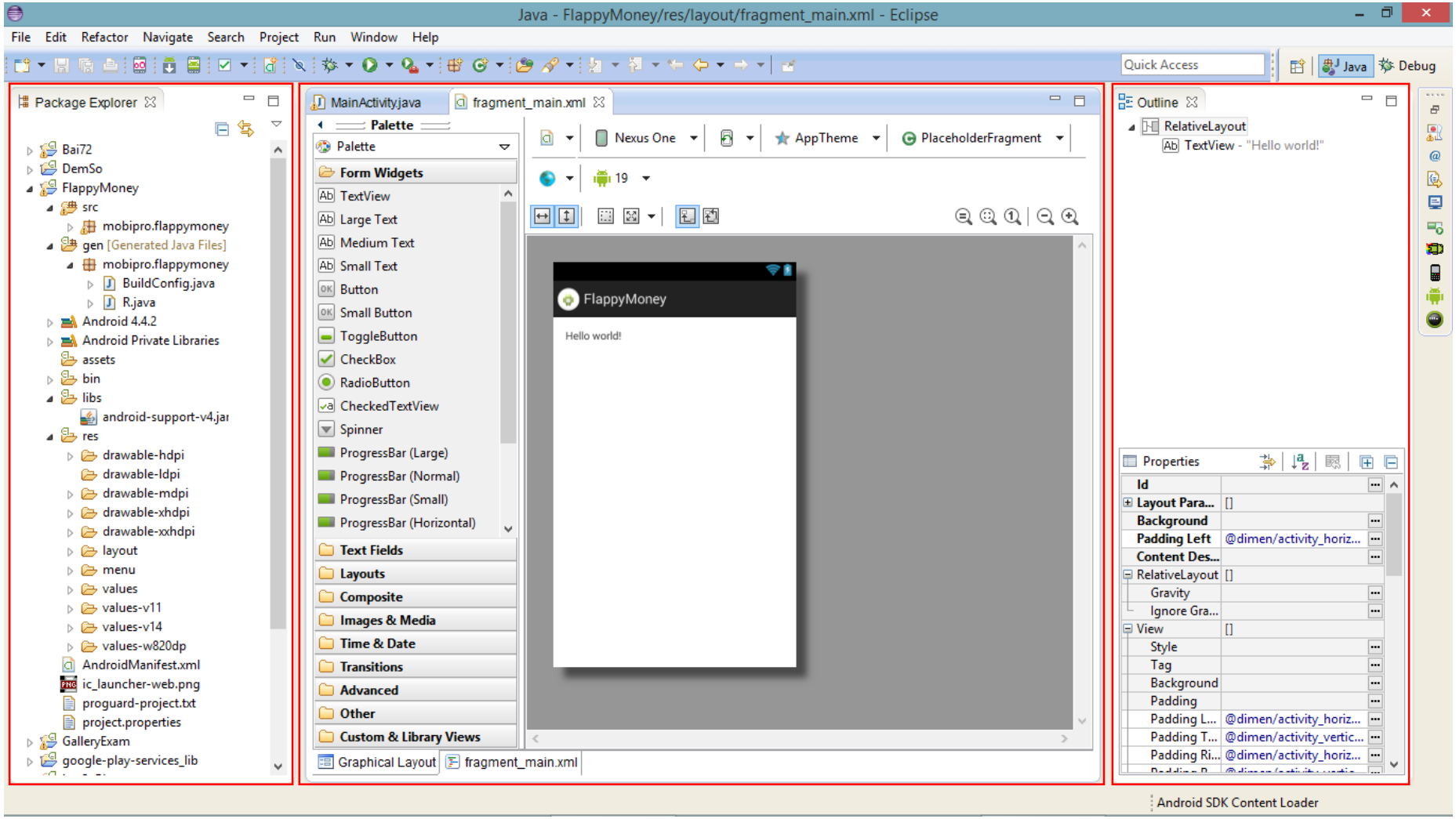
Nội dung

1. Giao diện phát triển của eclipse
2. Các thành phần của project
3. Các bước phát triển ứng dụng android
4. Các thành phần của một ứng dụng android
5. Activity
6. Vòng đời của activity

Phần 1

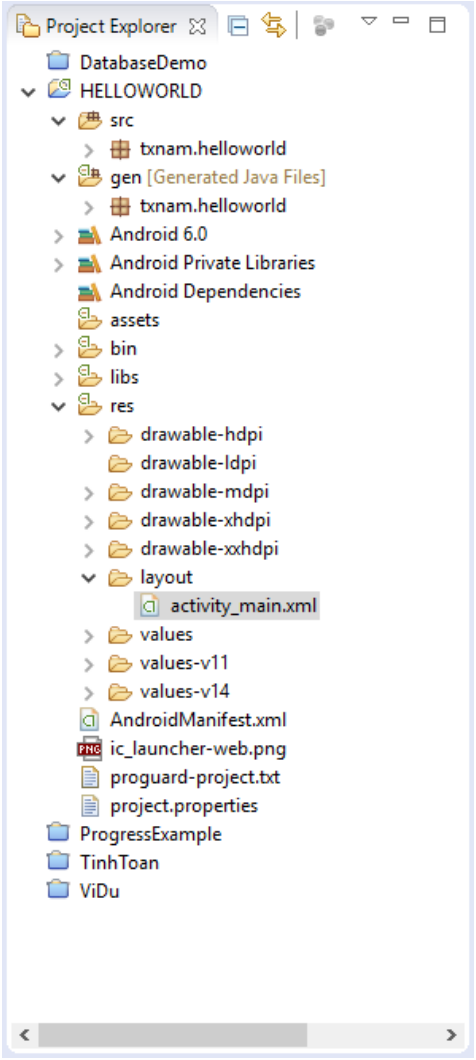
Giao diện phát triển của eclipse

Giao diện phát triển của eclipse



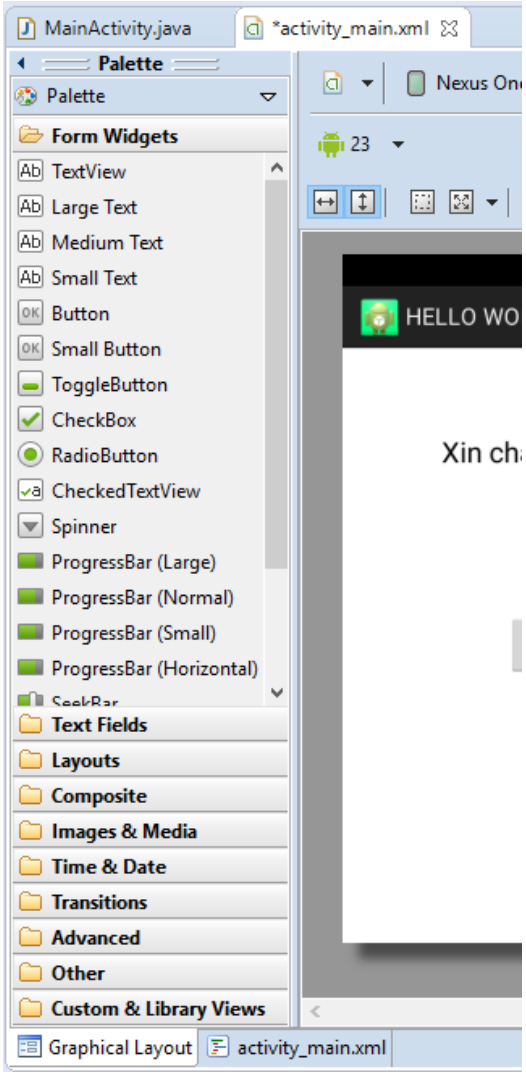
Giao diện phát triển của eclipse

- Workspace là không gian làm việc của eclipse, một workspace gồm nhiều project (dự án)
- Cửa sổ Project Explorer
  - Quản lý mọi project trong workspace
  - Tương tự cấu trúc thư mục lưu trữ vật lý
- Kinh nghiệm
  - Nên đóng (close) project không cần thiết để tránh chiếm tài nguyên máy tính
  - Không nên thao tác trực tiếp trên cấu trúc thư mục nếu không thực sự cần thiết



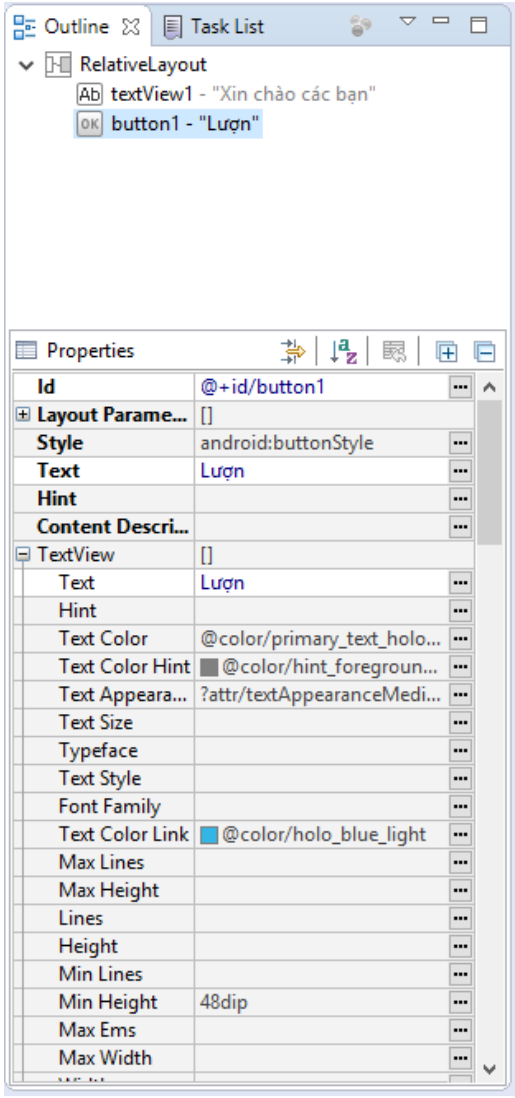
Giao diện phát triển của eclipse

- Cửa sổ soạn thảo chính cho phép LTV viết mã, xây dựng giao diện,…
- Hệ thống tab phía trên cho phép quản lý các file đang mở
- Hệ thống tab phía dưới cho phép LTV nhìn file dưới các chế độ xem khác nhau
  - Mỗi chế độ xem có thể kèm theo các cửa sổ công cụ phù hợp giúp dễ dàng phát triển ứng dụng



Giao diện phát triển của eclipse

- Cửa sổ thuộc tính gồm 2 cửa sổ con, hỗ trợ LTV dễ dàng thiết kế giao diện
- Cửa sổ Outline: cho phép xem sơ đồ bố trí các thành phần giao diện
- Cửa sổ Properties: liệt kê thuộc tính của thành phần giao diện hiện tại
  - LTV có thể xem và cập nhật giá trị thuộc tính tại cửa sổ này một các trực quan

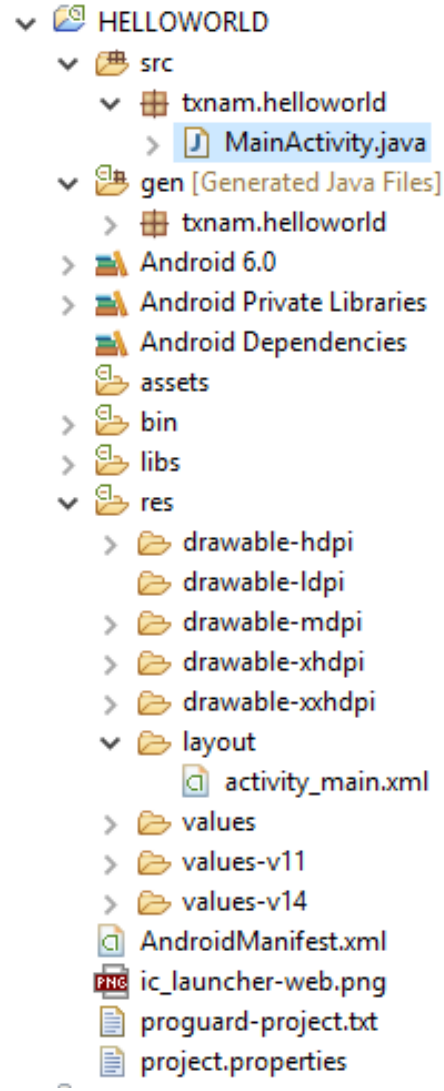


Phần 2

Các thành phần của project

Các thành phần của project

- Thư mục "src": mã nguồn do LTV viết
- Thư mục "gen": mã nguồn do công cụ sinh tự động
- Các thư mục "Android…": các thư viện hỗ trợ của Android SDK
- Thư mục "assets": file đính kèm
- Thư mục "bin": mã nhị phân
- Thư mục "libs": các file thư viện khác
- Thư mục "res": các file tài nguyên
- "AndroidManifest.xml": file cấu hình



AndroidManifest.xml

- Trước khi chạy, project cần phải được build (dựng), quá trình này phức tạp, nhưng có 2 bước chính
  - Dịch mã nguồn thành mã nhị phân (ở folder "bin")
  - Nén tất cả các file mã nhị phân và các file liên quan thành một file duy nhất, có phần mở rộng apk
- Khi cài đặt ứng dụng, hệ thống giải nén file apk và đọc file AndroidManifest.xml ở thư mục gốc
  - "AndroidManifest.xml" là file dạng xml chứa mọi thông tin về ứng dụng
  - Qua đó hệ thống biết ứng dụng có thể dùng vào việc gì

AndroidManifest.xml

- Các thông tin cơ bản trong "AndroidManifest.xml"
  - Các thông tin về ứng dụng (tên package, tên ứng dụng, biểu tượng của ứng dụng,…)
  - Các quyền cần có để chạy ứng dụng (quyền truy xuất internet, quyền đọc contact, quyền đọc SD card,…)
  - Phiên bản API tối thiểu có thể chạy ứng dụng
  - Các tính năng phần cứng cần thiết cho ứng dụng (GPS, camera, bluetooth,…)
  - Các bộ API liên kết sử dụng trong ứng dụng (Google Maps API, AdMod,…)
  - Cấu hình màn hình khởi chạy (ngang/dọc,…)

AndroidManifest.xml

- Các thông tin cơ bản trong "AndroidManifest.xml"
  - Mô tả về các activity (màn hình) của ứng dụng
    - Thông tin về activity (tên, tên class,….)
    - Xác định xem activity nào là giao diện khởi động của ứng dụng
  - Mô tả về các service (dịch vụ) mà ứng dụng cung cấp
    - Thông tin về service (tên dịch vụ, class xử lý dịch vụ,…)
  - Mô tả về các broadcast receiver mà ứng dụng cung cấp
    - Thông tin về receiver (tên receiver, class xử lý,…)
    - Các loại tín hiệu gửi đến receiver
  - Mô tả về các content provider mà ứng dụng cung cấp

AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="txnam.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="17"
        android:targetSdkVersion="21" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Phần 3

Các bước phát triển ứng dụng android

Các bước phát triển android apps

1. Nghiên cứu nhu cầu
2. Xây dựng giải pháp
   - Giải pháp có thể gồm các thành phần ngoài android (chẳng hạn như web service)
   - Đôi khi giải pháp không đáp ứng được nhu cầu do hạn chế về công nghệ
3. Viết ứng dụng
4. Phát hành ứng dụng
5. Nhận phản hồi, chỉnh sửa và nâng cấp

Viết ứng dụng

- Thiết kế phác họa giao diện (mockup)
- Chuẩn bị các tài nguyên (file ảnh, file âm thanh, video, văn bản,…)
- Thiết kế giao diện
  - Các activity (mỗi giao diện là một activity)
  - Lưu dạng XML để dễ dàng chỉnh sửa
- Viết mã
  - Các mã khởi tạo giao diện
  - Các mã hoạt động nền
  - Các mã liên kết giữa nền và giao diện

Phần 4

Các thành phần của một ứng dụng android

Ứng dụng android

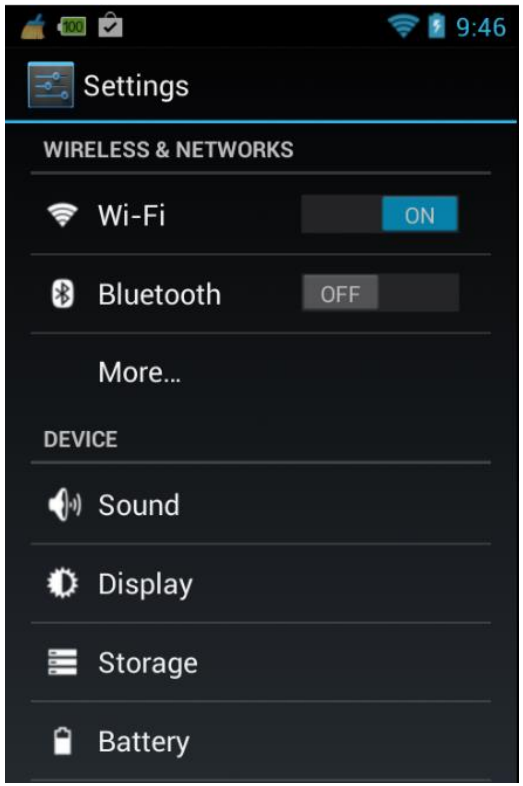
- Mỗi ứng dụng android đều chạy trên một tiến trình riêng trong một máy ảo riêng biệt
- Mỗi ứng dụng android là tập hợp các class, mỗi class có mục đích cụ thể. Hệ điều hành sẽ chủ động gọi thực thi các class này khi thấy cần thiết
  - Như vậy ta thấy ứng dụng android hơi có tính "bị động", các class sẽ được hệ điều hành chủ động gọi ra chạy, điều này khác với cách viết thông thường (hàm main chạy trước tiên, hàm main sẽ quyết định quá trình thực thi của ứng dụng)

Activity

- Một activity là một màn hình giao diện, một ứng dụng gồm một hoặc nhiều activity
- Android OS cung cấp sẵn một số lượng khá lớn các activity tiêu chuẩn
- Ví dụ:
  - Giao diện quay số và gọi điện
  - Giao diện settings
- Lập trình viên có thể tự viết activity riêng hoặc sử dụng các activity đã có

Activity

- Activity settings, được cung cấp bởi hệ thống



- Một activity được bên thứ 3 tự xây dựng

Service

- Service là một tiến trình thực thi một công việc chạy ngầm (thường không có hoặc rất ít tương tác với người sử dụng)
- Ví dụ:
  - Điều khiển việc chạy file nhạc
  - Thực hiện việc download/upload dữ liệu
  - Theo dõi và cảnh báo dung lượng pin
  - Theo dõi xem có cập nhật MXH hay không?
  - Ghi nhận ngầm thông tin (GPS chẳng hạn)

Content provider

- Content provider (còn gọi tắt là provider) dùng quản lý việc chia sẻ (dùng chung) một nguồn dữ liệu nào đó. Ví dụ:
  - Danh sách người dùng trên điện thoại
  - Dữ liệu về các cuộc gọi
  - Dữ liệu về tin nhắn
- Bằng cách chia sẻ dữ liệu để dùng chung, Android OS làm cho các ứng dụng dễ dàng cung cấp trải nghiệm nhất quán cho người dùng (chẳng hạn các ứng dụng thoại dùng chung danh bạ điện thoại)

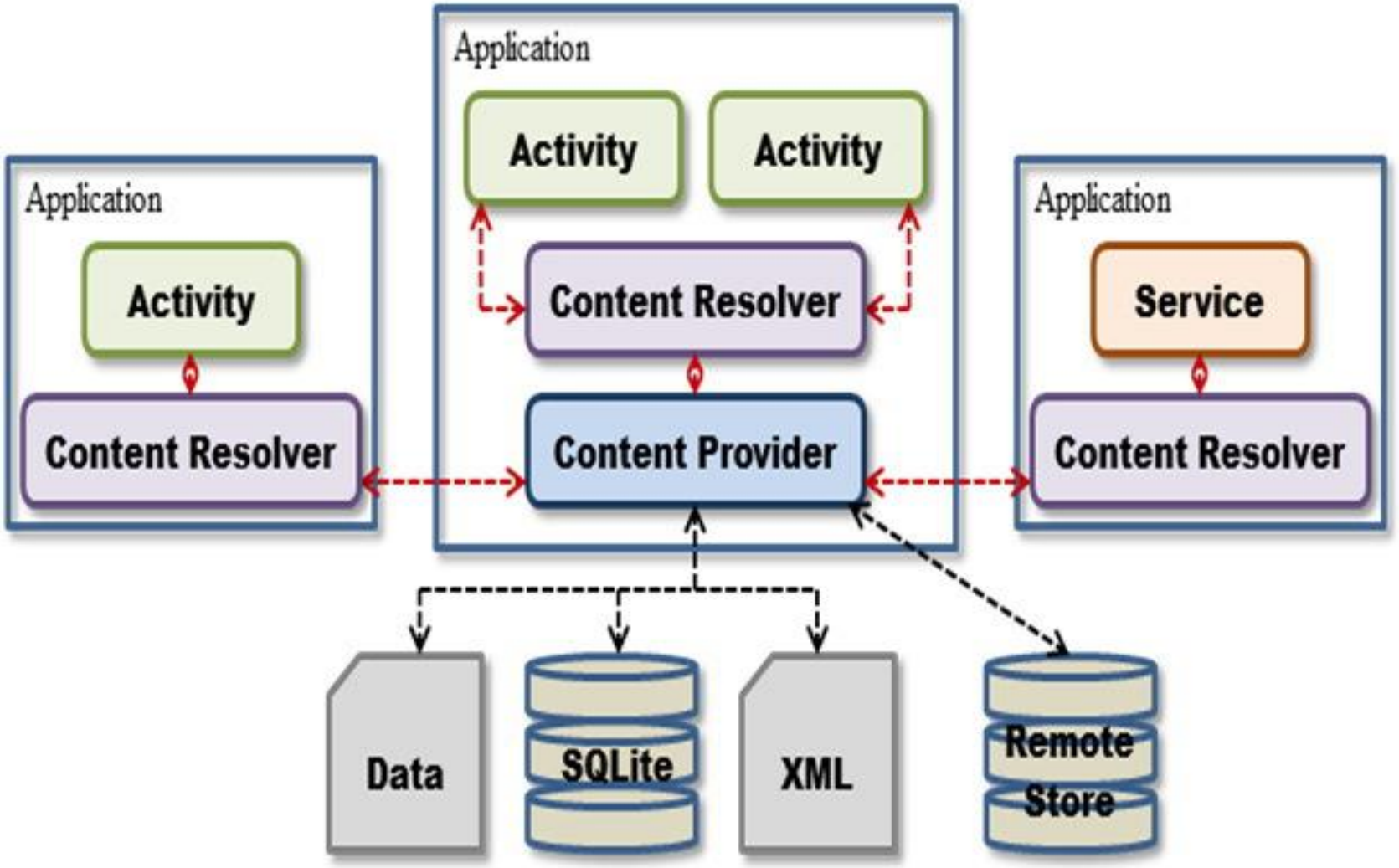
Broadcast receiver

- Broadcast receiver (còn gọi tắt là receiver) là một thành phần hồi đáp những tín hiệu được phát ra trên toàn hệ thống. Ví dụ:
  - Tín hiệu pin yếu
  - Tín hiệu mất kết nối mạng
  - Tín hiệu có cuộc gọi tới
- Lập trình viên có thể chặn các tín hiệu này và xử lý theo cách riêng của mình. Chẳng hạn:
  - Ứng dụng ngắt cuộc gọi đến từ số điện thoại quấy rối
  - Bật âm thanh cảnh báo khi điện thoại đã nạp đầy pin

Intent

- Intent là cơ chế chuẩn của Android OS để truyền thông tin giữa các thành phần cho nhau (giữa activity với nhau, activity cho service, receiver cho service,…)
- Chẳng hạn có thể sử dụng Intent để:
  - Gọi một service
  - Mở một activity
  - Hiển thị một trang web hoặc danh sách contacts
  - Hiển thị gallery để chọn ảnh

Các thành phần của ứng dụng



Cách thực thực thi điển hình

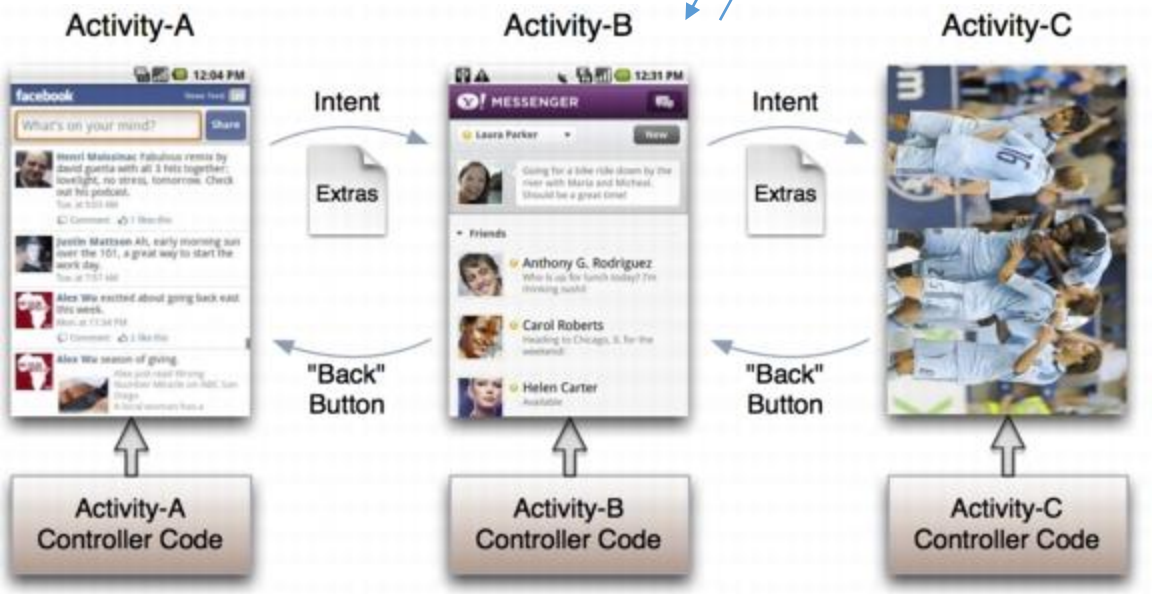
Notification

Intent

Content Providers

Services

Broadcast receiver



Phần 5

Activity

Activity

- Các activity là thành phần cơ bản của bất kỳ một ứng dụng android nào, chúng cung cấp giao diện người dùng cho ứng dụng
- Lớp Activity đảm nhận việc tạo ra một cửa sổ (window), sau đó ta có thể đặt lên đó một giao diện bằng lệnh setContentView(View)
- Thông thường mỗi màn hình sẽ là một activity, một ứng dụng thường gồm nhiều activity chuyển qua lại lẫn nhau

Activity

- Một activity có thể mang nhiều dạng khác nhau:
  - Cửa sổ chiếm toàn bộ màn hình
  - Cửa sổ chiếm một phần màn hình
  - Nằm lồng bên trong một activity khác
- Để có thể sử dụng, mọi activity đều phải được khai báo trong AndroidManifest.xml với thẻ <activity>

```
<activity
    android:name=".MainActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Tạo Activity

- Mỗi activity trình bày một màn hình, class xử lý activity bao giờ cũng kế thừa lớp Activity của Android

```
package txnam.helloworld;

import android.app.Activity;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void nutThoat(View v) {
        finish();
    }
}
```

Khởi tạo giao diện bên trong

- Có 2 cách đơn giản để tạo giao diện cho activity
- Tự tạo giao diện bằng viết mã

```
@Override
public void onCreate(Bundle icicle) {
    super.onCreate(icicle);
    MyView myView = new MyView(this);
    setContentView(myView);
}
```

- Nạp giao diện đã thiết kế trên file layout (.xml)

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

Gọi activity khác

- Gọi trực tiếp activity đã định nghĩa

```
Intent i = new Intent(this, MyActivity.class);
startActivity(i);
```

- Gọi gián tiếp activity

```
Intent i = new Intent(Intent.ACTION_SEND);
i.putExtra(Intent.EXTRA_EMAIL, addList);
startActivity(intent);
```

- Khi gọi gián tiếp, hệ thống tự chọn activity phù hợp nhất với yêu cầu (sẽ được thảo luận sau)

Phần 6

Vòng đời của activity

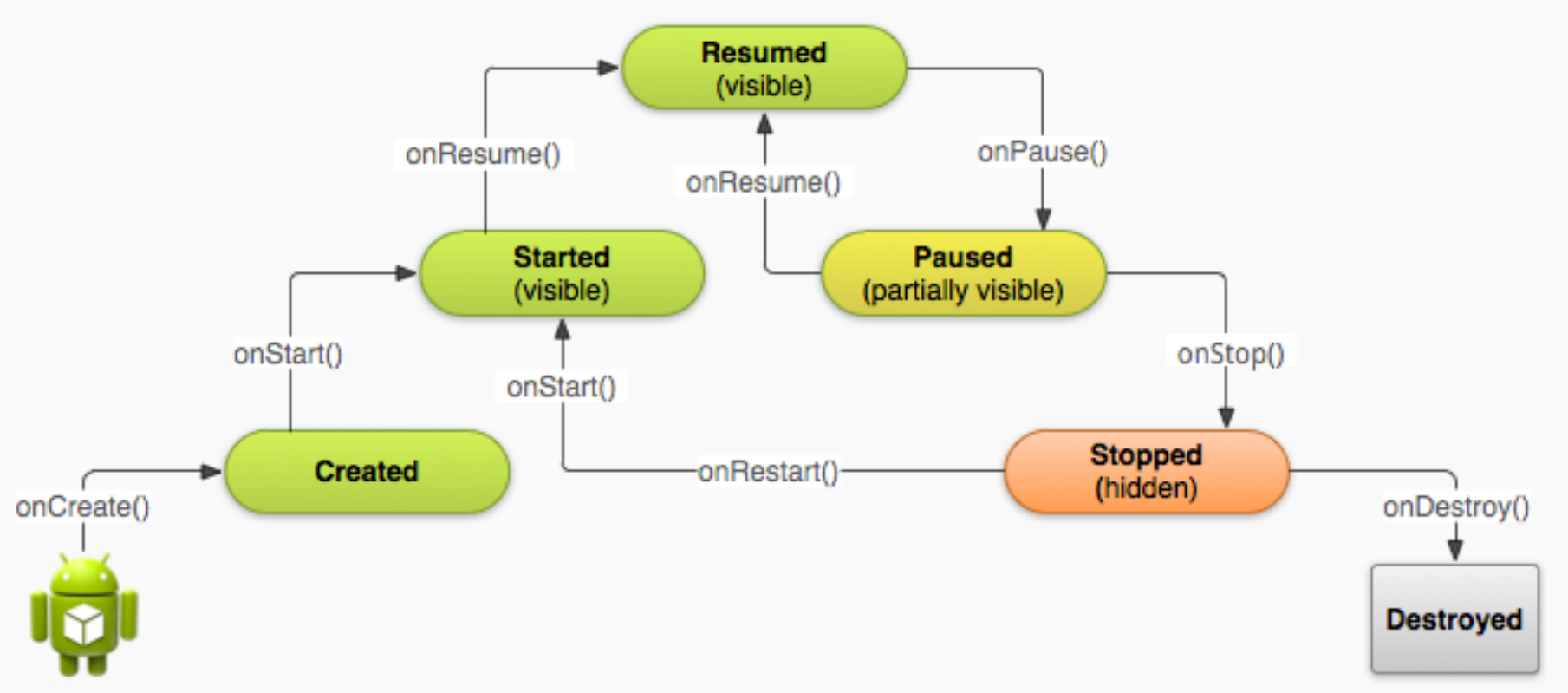
Vòng đời của một activity

- Các activity được quản lí trong một stack chứa activity (cơ chế vào trước ra sau):
  - Khi ứng dụng được mở lên thi activity chính sẽ được tạo ra, nó sẽ được thêm vào đỉnh của stack
  - Lúc này chỉ có duy nhất activity trên cùng là hiển thị nội dung đến người dùng
  - Tất cả các activity còn lại đều chuyển về trạng thái dừng hoạt động
  - Khi một activity bị đóng nó sẽ bị loại khỏi stack, activity nằm dưới đó sẽ chuyển từ trạng thái tạm dừng sang trạng thái hoạt động

Vòng đời của một activity

- Một activity có bốn trạng thái:
  - Active hay Running: activity đang chạy trên màn hình
  - Paused: khi một activity mất focus nhưng vẫn đang chạy trên màn hình (một activity trong suốt hoặc một activity không chiếm toàn bộ màn hình thiết bị đè lên)
  - Stopped: khi một activity bị che khuất hoàn toàn bởi một activity khác
  - Killed hay Shutdown: khi một activity đang Paused hay Stopped, hệ thống có thể xóa activity ấy nếu cần (chẳng hạn như cần bộ nhớ vào việc khác)

Vòng đời của một activity



Các sự kiện trong vòng đời của APP

- Khi một activity bị chuyển qua chuyển lại giữa các trạng thái, nó được cảnh báo việc chuyển này bằng hàm chuyển trạng thái (transition)
- Có thể viết lại các hàm chuyển này nếu cần làm các công việc giúp việc chuyển trạng thái suôn sẻ

```
1. protected void onCreate(Bundle b);
2. protected void onStart();
3. protected void onRestart();
4. protected void onResume();
5. protected void onPause();
6. protected void onStop();
7. protected void onDestroy();
```

Các hàm trong vòng đời activity

- onCreate(...): gọi khi activity khởi tạo
- onStart(): gọi khi acivity xuất hiện trên màn hình
- onResume(): gọi ngay sau onStart hoặc người dùng focus, hàm này đưa ứng dụng lên top màn hình
- onPause(): gọi khi hệ thống focus đến activity khác
- onStop(): gọi khi activity bị che hoàn toàn
- onRestart(): gọi khi ứng dụng khởi chạy lại
- onDestroy(): gọi khi ứng dụng chuẩn bị được gỡ khỏi bộ nhớ