LẬP TRÌNH ANDROID

Giảng Viên: Phùng Mạnh Dương

Trường ĐH Công nghệ, ĐHQGHN

Permission

- Khái niệm
- Định nghĩa và cách sử dụng
- Các thành phần

Permission

- Android sử dụng permission để bảo vệ dữ liệu và tài nguyên.
- 1 ứng dụng có thể định nghĩa và thiết lập permission để giới hạn truy cập vào tài nguyên như:
  - Thông tin cá nhân: VD Danh bạ
  - Các API liên quan đến chi phí: SMS/MMS
  - Tài nguyên hệ thống: Camera
- Permission được biểu diễn như các chuỗi và được thiết lập ở trong file AndroidManifest.xml
  - Yêu cầu Permission
  - Thiết lập Permission

Yêu cầu Permission

- Các ứng dụng yêu cầu Permission thông qua thẻ <uses-permission>
- Khi ứng dụng cài trên thiết bị, người dùng sẽ cần phải đồng ý thì permission mới được cấp cho ứng dụng.
- VD: MapLocationsFromContacts

```
<manifest ... >
...
<uses-permission android:name="android.permission.CAMERA”/>
<uses-permission android:name="android.permission.INTERNET”/>
<uses-permission
android:name=“android.permission.ACCESS_FINE_LOCATION”/>
...
</manifest >
```

Tham khảo:
http://developer.android.com/reference/android/Manifest.permission.html

Định nghĩa và thiết lập Permission

- Sử dụng khi ứng dụng không cho phép ứng dụng bất kỳ có thể truy cập.
- VD: PermissionExampleBoom
- Định nghĩa permission

```
<!-- Defines a custom permission -->
<permission
    android:name="course.examples.permissionexample.BOOM_PERM"
    android:description="@string/boom_perm_string"
    android:label="@string/boom_permission_label_string" >
</permission>

<!--  Enforces the BOOM_PERM permission on users of this application  -->
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:permission="course.examples.permissionexample.BOOM_PERM" >
```

Thiết lập permision cho các thành phần

- Permission có thể thiết lập cho từng thành phần của ứng dụng.
  - Activity permission
  - Service permission
  - ContentProvider permission
  - BroadcastReceiver permission
- Tham khảo:

http://developer.android.com/guide/topics/security/permissions.html

Fragment

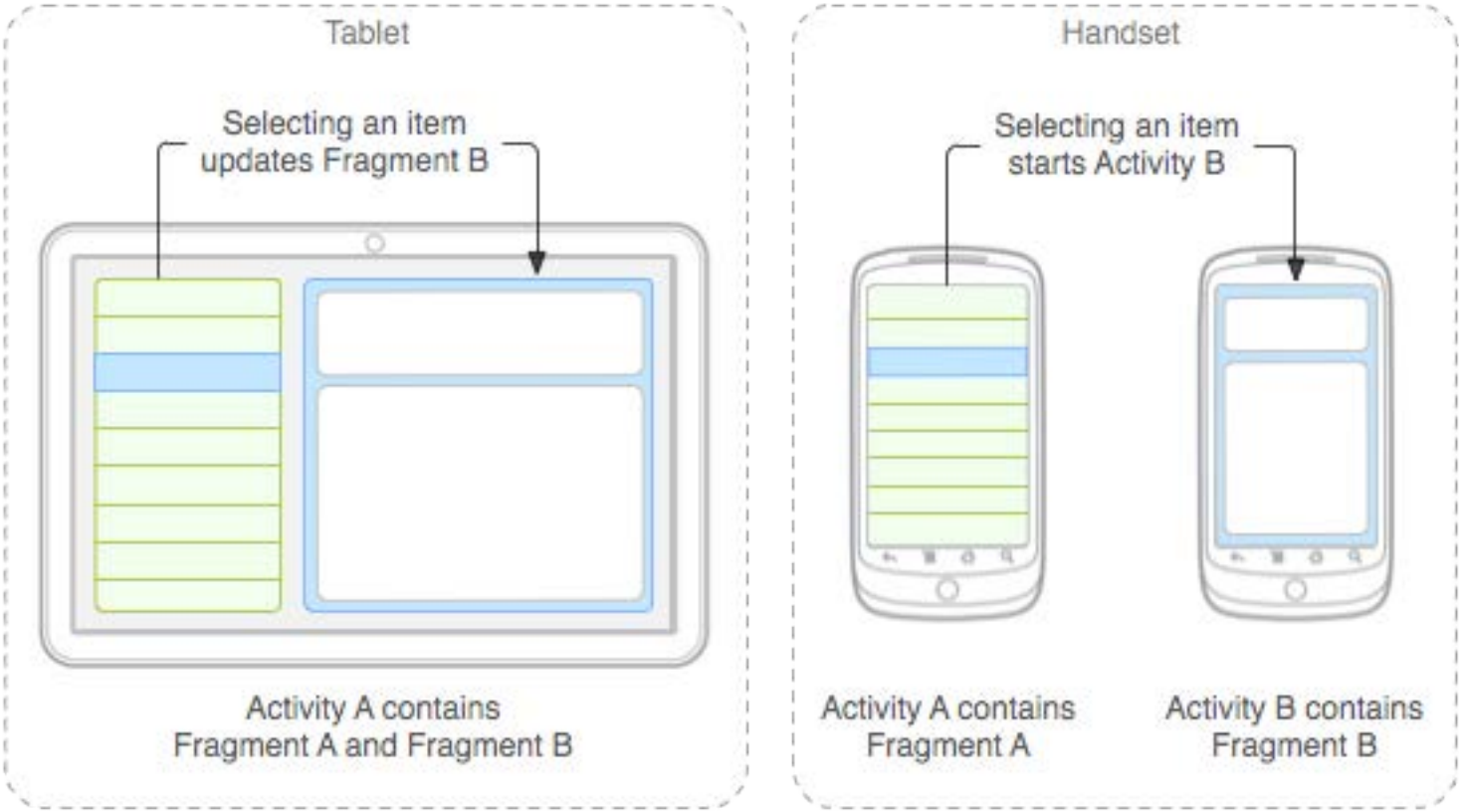
- Khái niệm & ý nghĩa
- Đặc điểm và cách sử dụng
- Tương tác động

Giao diện cho máy tính bảng

- Máy tính bảng có màn hình lớn hơn điện thoại do đó có thể hỗ trợ giao diện với nhiều vùng tương tác người dùng tại cùng thời điểm.
- Triết lý: 1 activity – 1 việc người dùng có thể thực hiện trở nên không phù hợp với thiết bị di động có màn hình lớn hơn.
- VD:
  - Ứng dụng Shakespear với 2 activity
  - Ứng dụng Shakespear với 2 khối giao diện trên màn hình.
  - Liên hệ sang web với responsive design: vnexpress

Fragments

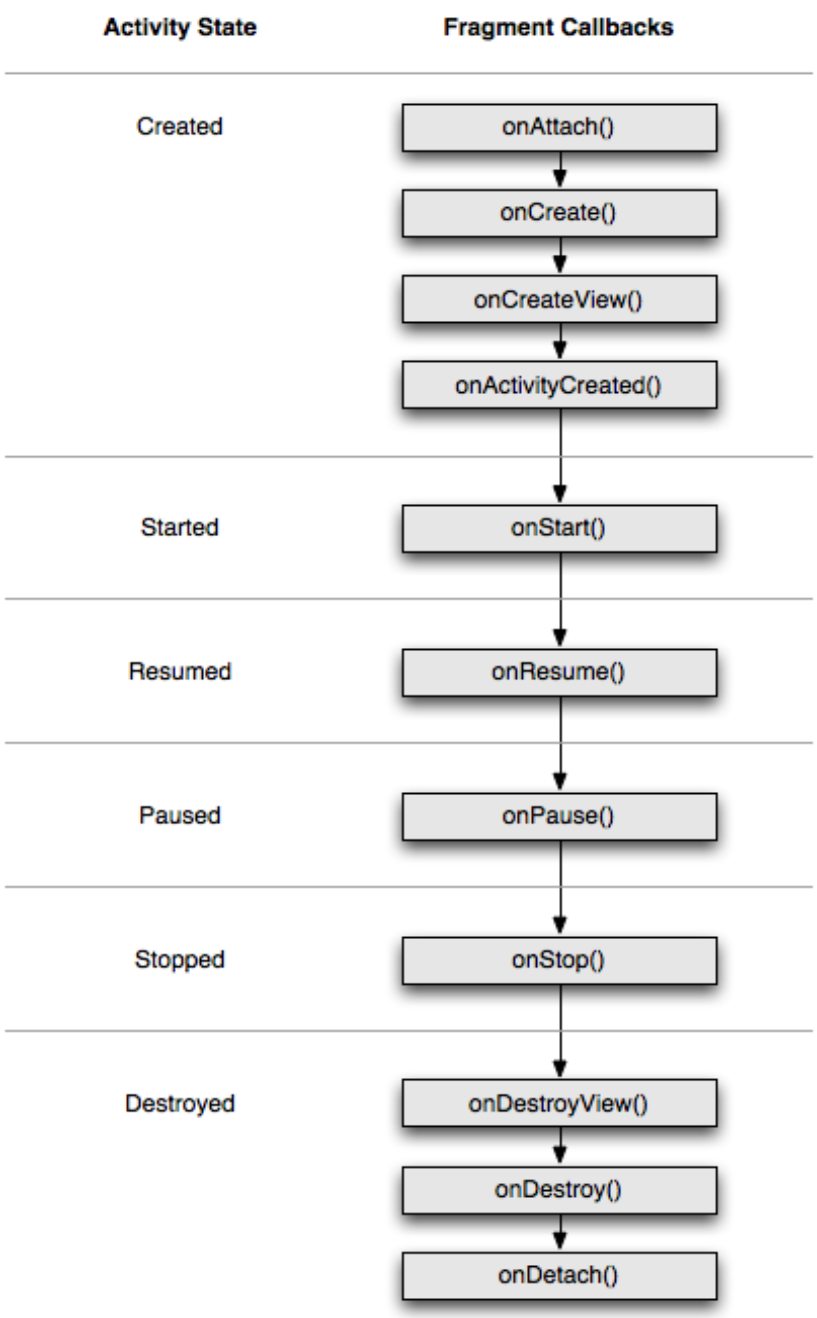
- **Fragment** là một phần giao diện được nhúng vào **Activity**.
- Có thể xem **Fragment** như là một hoặc nhiều giao diện (***sub Activity***) có lifecycle riêng và thường nằm cùng một màn hình.



Đặc điểm của Fragment

- Nhiều Fragment có thể được nhúng vào trong 1 activity để tạo thành giao diện gồm nhiều khối.
- Mỗi Fragment có thể **được sử dụng lại** trong nhiều activity khác nhau.
- Chu kỳ sống (lifecycle) của Fragment gắn liền với chu kì sống của Activity chứa nó. Tuy vậy, Fragment có các hàm lifecycle riêng.
- Các trạng thái trong chu kỳ sống của Fragment giống với Activity:
  - Resumed: Hiển thị và có thể tương tác.
  - Paused: Hiển thị nhưng không thể tương tác.
  - Stopped: không hiển thị.

Fragment vs Activity lifecycle



VD trên eclipse

Chèn Fragment vào Activity

- 2 cách:
  - Khai báo tĩnh trong file layout của activity
  - Chèn động trong mã nguồn java sử dụng FragmentManager
- Giao diện của Fragment thường được thiết lập trong hàm onCreateView()

Chèn Fragment tĩnh

- Khai báo trong file layout của activity

```
<fragment
    android:id="@+id/titles"
    android:layout_width="0px"
    android:layout_height="match_parent
    android:layout_weight="1"
    class="course.TitlesFragment" />
```

- VD: FragmentStaticLayout

Chèn Fragment động

- Fragment có thể được chèn động khi activity đang hoạt động thông qua 4 bước
  - Lấy tham chiếu tới FragmentManager
  - Bắt đầu một FragmentTransaction
  - Chèn Fragment
  - Giao (Commit) Fragment

```
FragmentManager fragmentManager = getFragmentManager();
FragmentTransaction fragmentTransaction = fragmentManager
        .beginTransaction();
fragmentTransaction.add(R.id.title_frame,
                        new TitlesFragment());
fragmentTransaction.commit();
```

- VD: FragmentProgrammaticLayout

Layout động cho fragment

- Fragment giúp cho phép thay đổi động giao diện người dùng
- Giúp giao diện uyển chuyển hơn và tận dụng tốt hơn khoảng trống màn hình
- VD: FragmentDynamicLayout

```
// Determine whether the QuoteFragment has been added
if (!mQuoteFragment.isAdded()) {

    // Make the TitleFragment occupy the entire layout
    mTitleFrameLayout.setLayoutParams(new LinearLayout.LayoutParams(
            MATCH_PARENT, MATCH_PARENT));
    mQuotesFrameLayout.setLayoutParams(new LinearLayout.LayoutParams(0,
            MATCH_PARENT));
} else {

    // Make the TitleLayout take 1/3 of the layout's width
    mTitleFrameLayout.setLayoutParams(new LinearLayout.LayoutParams(0,
            MATCH_PARENT, 1f));

    // Make the QuoteLayout take 2/3's of the layout's width
    mQuotesFrameLayout.setLayoutParams(new LinearLayout.LayoutParams(0,
            MATCH_PARENT, 2f));
}
```

Đáp ứng với thay đổi cấu hình

- Nếu gọi hàm setRetainInstance(true), Android sẽ không hủy Fragment khi cấu hình thay đổi.
  - Phương thức onDestroy() không được gọi
  - Phương thức onCreate() không được gọi
- VD: FragmentStaticConfigLayout