MIT
Global Startup Labs

http://aiti.mit.edu

Indonesia Summer 2013
Meetup 05 – Preferences and Services

MISTI

Google

Today‘s Meetup

- The preference activity
- Shared preferences
- Application Object
- Services

Preferences

Four Steps to your Preference Activity

- Create the Preference resource file *prefs.xml* in *res/xml/prefs.xml*
- Implement *PrefsActivity.java*
- Register new activity in *AndroidManifest.xml*
- Provide a way to start the preference activity

Create a Preference Menu

```xml
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android" >

    <EditTextPreference android:title="@string/titleUsername"
        android:summary="@string/summaryUsername"
        android:key="username">
    </EditTextPreference>

    <EditTextPreference android:title="@string/titlePassword"
        android:password="true"
        android:summary="@string/summaryPassword"
        android:key="password">
    </EditTextPreference>

    <EditTextPreference android:title="@string/titleApiRoot"
        android:summary="@string/summaryApiRoot"
        android:key="apiRoot">
    </EditTextPreference>

</PreferenceScreen>
```

Create name.xml file in res/xml/

Define preference menu elements

- Text fields
- Checkbox
- Lists
- …

Create *name*.java class in scr/*package*/

Make your class a subclass of the PreferenceAvtivityClass

```java
public class PrefsActivity extends PreferenceActivity{
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.prefs); //method is deprecated
    }
}
```

Inflate the XML layout

Create a Preference Menu

Register the Activity in Android.Manifest.xml

Android.Manifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity...
    <activity android:name=".PrefsActivity"
        android:label="@string/titlePrefs" />
</application>
```

Start activity using the following command: (can be implemented in Buttons, menus, OnCreate(), etc)

```
startActivity(new Intent(this, PrefsActivity.class));
```

Now we can access the Preference Activity and set our preferences. BUT: How do we make use of the preferences we have defined?

Use SharedPreference Class

SharedPreference Class

Import Statements

Create global variable to store preference values

Set value of variable to current set preferences

Implement Listener: make sure to detect changes in preferences

Define what happens when user changes preferences

Activity.java

```
import android.app.Activity;

public class StatusActivity  extends Activity implements OnSharedPreferenceChangeListener{

    SharedPreferences prefs;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_status);

        //find set preferences from preference manager
        prefs = PreferenceManager.getDefaultSharedPreferences(this);

        //listen for changes in preferences -> Listener has to be implemented
        //method has to be defined
        prefs.registerOnSharedPreferenceChangeListener(this);
    }
    public void onSharedPreferenceChanged(SharedPreferences prefs, String key) {
        //define what happens when preferences are changed
        }

    //in our Yamba example: need to add twitter method to get login info
    //from prefs variable
}
```

Application Objects

Application Object

- Problem: What if we want to reuse code across different activities or services?
- Solution: Build an Application Object!
  - No need to copy code
  - Represents a common state of entire application
- Two Steps:
  - Create a Java Class
  - Register Class in Android.Manifest.xml

Create an Application Object

YambaApplication.java

```
package com.maraka.yamba;

import android.app.Application;

public class YambaApplication extends Application{

    @Override
    public void onCreate() { //
        super.onCreate();
        //Define what happens when app is created
        }

    @Override
    public void onTerminate() { //
        super.onTerminate();
        //Do clean-up when app shuts down
    }

    //Define all methods that you want to reuse
    //across different activities
}
```

Create name.java class in scr/*package*/

Make your class a subclass of the ApplicationClass

Define OnCreate(), OnTerminate(), any methods you want to be global in your app

Register Class in Android.Manifest.xml

Android.Manifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme"
    android:name=".YambaApplication">
    <activity
</application>
```

Services

Services

- One of the main building blocks in Android
- Doesn‘t have a UI, runs in background (independent of activities)
- Service Lifecycle enables us to define what happens OnCreate(), etc
- Three steps:
  - Create a Java Class
  - Register Service in Android.Manifest.xml
  - Start the Service

Create a Service

1.

```
import android.app.Service;
import android.content.Intent;
import android.os.IBinder;

public class UpdaterService extends Service {

    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
        @Override
    public void onCreate() {
        super.onCreate();

    }
    public void onStartCommand() {
        super.onStartCommand(null, 0, 0);
    }
    @Override
    public void onDestroy() {
        super.onDestroy();
    }

}
```

Create nameService.java class in scr/*package*/

Make your class a subclass of the ServiceClass

Define OnCreate(), onStartCommand() and OnDestroy

Register Service in Android.Manifest.xml

2.

Android.Manifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme"
    android:name=".YambaApplication">
    <activity...
    <service android:name=".UpdaterService" />
</application>
```

Start or stop service using the following commands:
(can be implemented in Buttons, menus, OnCreate(), etc)

3.

```
startService(new Intent(this, UpdaterService.class));

stopService(new Intent(this, UpdaterService.class));
```

Today‘s Assignment

Today‘s Assignment

Extend your Yamba project!

- Finish off implementing shared preferences
- Implement an UpdaterService that periodically checks for your friends status updates

If you cannot get your shared preferences to work, try to carry on with hard coded Username and Password and implement your service

Documentation: Gargenta – Learning Android: Chapters 7 and 8