# 技能点

### 数据持久化

#### 建立持久化对象

SharedPreferences sp = getSharedPreferences("user\_info",***MODE\_PRIVATE***);

#### 保存数据

sp.edit().putString("user\_name",userName).apply();

#### 读取数据

String user\_name = sp.getString("user\_name", **null**);

### 使用线程模拟耗时操作

#### 使用滚动条对话框

**private** ProgressDialog **dialog**; *// 定义控件*

**dialog** = **new** ProgressDialog(**this**); *// 建立对象*

**dialog**.show(); *// 显示滚动条*

**dialog**.dismiss(); *// 关滚动条*

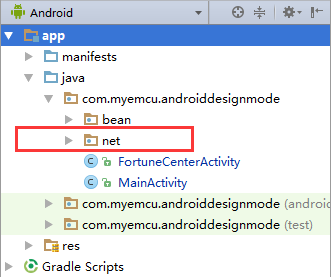
#### 使用线程Thread

**new** Thread()**{** @Override  
 **public void** run() **{** SystemClock.sleep(2000); *// 睡2s* **if** (userName.equals("yuyang") && passWord.equals("123456")) **{***// 验证成功* **runOnUiThread**(**new** Runnable() **{** @Override  
 **public void** run() **{ // 所有登陆成功的处理**   
 **}  
 }**);  
 **}else {  
 runOnUiThread**(**new** Runnable() **{ // 所有登陆失败的处理** @Override  
 **public void** run() **{**   
 **}  
 }**);  
 **}  
 }  
}**.start();

### 设计模式MVC

#### 解释

* M（数据模型（e.g：数据库、网络等））V（Layout）C（Activity）
* 工程目录（net包中的类则为Modle）：



* 代码：

1. net中的代码

**public class** UserLoginNet **{** *// 发送用户的输入数据到服务端* **public boolean** sendUserLoginInfo(UserInfo userInfo) **{** SystemClock.sleep(2000); *// 睡2s，模拟联网处理* **if** (userInfo.**username**.equals("yuyang") && userInfo.**password**.equals("123456")) **{***// 验证成功* **return true**;  
 **}else {  
 return false**;  
 **}  
 }  
}**

1. MainActivity中的代码

*// 联网模拟***new** Thread()**{** @Override  
 **public void** run() **{** UserLoginNet net = **new** UserLoginNet(); *// 联网处理(Modlue)* **if** (net.sendUserLoginInfo(userInfo)) **{** runOnUiThread(**new** Runnable() **{** @Override  
 **public void** run() **{** *// 1 关进度条* **dialog**.dismiss();  
  
 *// 2 保存登陆的用户名* SharedPreferences sp = getSharedPreferences("user\_info",***MODE\_PRIVATE***);  
 sp.edit().putString("user\_name",userName).apply();  
  
 *// 3 页面跳转* Intent intent = **new** Intent(MainActivity.**this**,FortuneCenterActivity.**class**);  
 startActivity(intent);  
 finish();  
 **}  
 }**);  
 **}else {** runOnUiThread(**new** Runnable() **{** @Override  
 **public void** run() **{  
 dialog**.dismiss(); *// 关进度条* Toast.makeText(MainActivity.**this**,"账号或密码有误",Toast.***LENGTH\_SHORT***).show();  
 **}  
 }**);  
 **}  
 }  
}**.start();

* MVC设计模式解读

1. 特征：Activity中的两部分内容：业务、界面
2. 特点：View少而C多（Layout少Activity多）
3. 劣势：界面或业务复杂导致Activity中的代码臃肿难以维护
4. 方案：MVP——拆分业务；MVVM——拆分界面