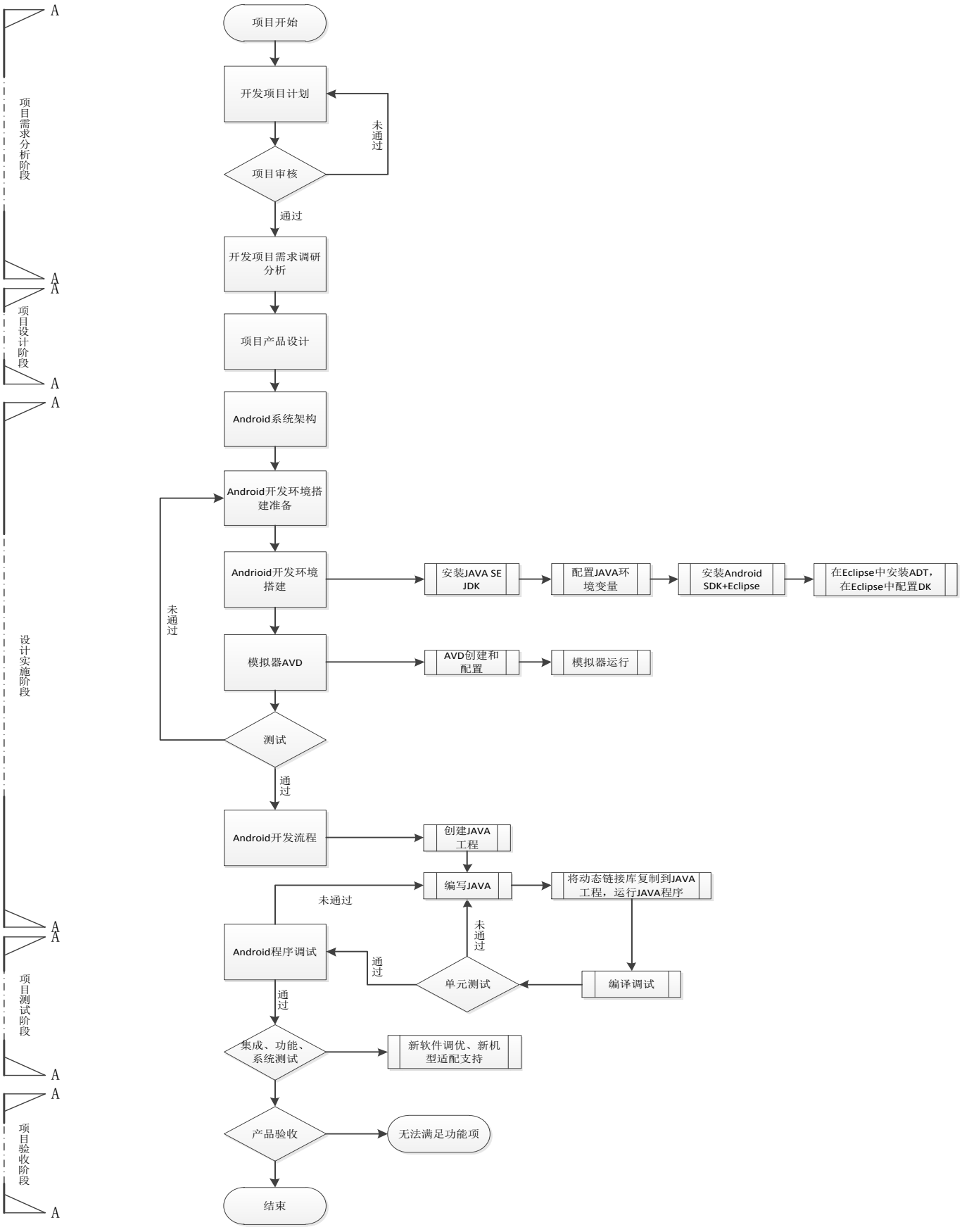
Android 客户端开发流程

1 项目流程图



2 项目阶段描述

2.1 项目需求分析阶段描述

输入：

《项目产品开发计划书》

《总体方案书》

输出：

《项目产品需求规格说明书》

《产品测试计划及裁减说明书》

2.2 项目设计施阶段

输入：

《项目产品需求规格说明书》

《产品测试计划及裁减说明书》

输出：

《产品概要设计说明书》

《系统测试方案》

《集成测试方案》

2.3 项目实施阶段

输入：

《项目产品需求规格说明书》

《产品概要设计说明书》

待更新的《系统测试方案》

待更新的《集成测试方案》

输出：

《产品详细设计说明书》

更新的《系统测试方案》

更新的《集成测试方案》

2.4 项目测试阶段

输入：

《项目产品需求规格说明书》

《产品详细设计说明书》

《系统测试方案》

《集成测试方案》

《产品单元测试记录》

输出：

《系统测试缺陷记录》

《产品单元测试报告》

《集成测试报告》

《系统测试报告》

2.5 项目验收阶段

输入：

《项目产品需求规格说明书》

《产品详细设计说明书》

《系统测试缺陷记录》

《产品单元测试报告》

《集成测试报告》

《系统测试报告》

输出：

《项目总结报告》

《项目中无法满足功能项说明书》

《维护方案》

本文介绍了如何使用 Android 搭建客户端，实现手机和服务器的交互。让我们了解如何采用 SSH 框架，把服务器端的信息用 JSON 的形式发送到手机端。

笔者以前是学的 Java EE，由于项目需要要开发 Android，所以临时补了一个多星期，主要是手机端和服务器端交互，双向开发的。

首先在服务器端，我采用的是 SSH 框架，struts 2 集合了 JSON 插件，服务器和客户端的信息交互采用的 JSON 来传输，由于在服务器端用了 Struts 2，所以我就用装了一个 JSON 插件。这样，很轻易的就把服务器端的信息用 JSON 的形式发送到了手机端。以下是代码：

首先，在服务器端搭建好 SSH 框架，具体细节就不在陈述。struts.xml 配置如下：

```
<packagename="login"extends="json-default">
<actionname="login"class="com.jclick.test.LoginAction"method="login">
<resulttype="json"><paramname="includeProperties">result</param></result>
</action>
</package>

    <packagename="login"extends="json-default">
    <actionname="login"class="com.jclick.test.LoginAction"method="login">
    <resulttype="json"><paramname="includeProperties">result</param></result>
    </action>
    </package>
```

手机端的代码如下：

首先，手机端有一个缓存类，主要用于缓存一些手机端需要访问的数据，这样的好处是可以达达节省手机和服务器的交互，用单例实现的：

```
packagecom.jclick.cache;

importcom.jclick.bean.User;

publicclassCache{

privateUserUser;

privateCache(){

}
/**构造单例*/
privatestaticclassCacheHolder{
privatestaticfinalCacheINSTANCE=newCache();
}
publicCachegetInstance(){
returnCacheHolder.INSTANCE;
}
publicUsergetUser(){
```

```
returnUser;
}
publicvoidsetUser(UserUser){
this.User=User;
}

}

packagecom.jclick.cache;

importcom.jclick.bean.User;

publicclassCache{

    privateUserUser;

    privateCache(){

    }
    /**构造单例*/
    privatestaticclassCacheHolder{
        privatestaticfinalCacheINSTANCE=newCache();
    }
    publicCachegetInstance(){
        returnCacheHolder.INSTANCE;
    }
    publicUsergetUser(){
        returnUser;
    }
    publicvoidsetUser(UserUser){
        this.User=User;
    }

}
```

接着开始书写手机端的协议，用户向服务器发送请求，同时服务器反馈给手机端信息的：

```
packagecom.jclick.protocol;

importjava.io.BufferedReader;
importjava.io.InputStreamReader;
importjava.util.ArrayList;
importjava.util.List;

importorg.apache.http.HttpResponse;
importorg.apache.http.NameValuePair;
importorg.apache.http.client.HttpClient;
importorg.apache.http.client.entity.UrlEncodedFormEntity;
importorg.apache.http.client.methods.HttpPost;
importorg.apache.http.impl.client.DefaultHttpClient;
```

```
importorg.apache.http.message.BasicNameValuePair;
importorg.json.JSONException;
importorg.json.JSONObject;

publicclassBaseProtocol{
privateStringBuildersb=newStringBuilder();

privateHttpClienthttpClient;
privateHttpPosthttpRequest;
privateHttpResponseresponse;

privateList<NameValuePair>nameValuePair=newArrayList<NameValuePair>();

BaseProtocol(){
httpClient=newDefaultHttpClient();
}

/**
*向服务器端发送请求
*
*@paramurl
*@throwsException
*/
protectedvoidpack(Stringurl)throwsException{
httpClient=newDefaultHttpClient();
httpRequest=newHttpPost(url);

httpRequest.setEntity(newUrlEncodedFormEntity(nameValuePair));
response=httpClient.execute(httpRequest);
}

/**
*得到返回数据
*
*@paramurl
*@return
*@throwsException
*/
protectedvoidparse()throwsException{
//TODO状态处理500200
if(response.getStatusLine().getStatusCode()==200){

BufferedReaderbufferedReader2=newBufferedReader(
newInputStreamReader(response.getEntity().getContent()));
for(Strings=bufferedReader2.readLine();s!=null;s=bufferedReader2
.readLine()){
sb.append(s);
}
}
```

```
63. }
64.
65. /**
66. *向服务器发送信息
67. *
68. *@paramkey
69. *@paramvalue
70. */
71. publicvoidaddNameValuePair(Stringkey,Stringvalue){
72. nameValuePair.add(newBasicNameValuePair(key,value));
73. }
74.
75. /**
76. *返回 JSONObject 对象数据模型
77. *
78. *@return
79. *@throwsJSONException
80. */
81. publicJSONObjectgetJSON()throwsJSONException{
82. returnnewJSONObject(sb.toString());
83. }
84.
85. }
86.
87. packagecom.jclick.protocol;
88.
89. importjava.io.BufferedReader;
90. importjava.io.InputStreamReader;
91. importjava.util.ArrayList;
92. importjava.util.List;
93.
94. importorg.apache.http.HttpResponse;
95. importorg.apache.http.NameValuePair;
96. importorg.apache.http.client.HttpClient;
97. importorg.apache.http.client.entity.UrlEncodedFormEntity;
98. importorg.apache.http.client.methods.HttpPost;
99. importorg.apache.http.impl.client.DefaultHttpClient;
100.importorg.apache.http.message.BasicNameValuePair;
101.importorg.json.JSONException;
102.importorg.json.JSONObject;
103.
104.publicclassBaseProtocol{
105.    privateStringBuildersb=newStringBuilder();
106.
107.    privateHttpClienthttpClient;
108.    privateHttpPosthttpRequest;
109.    privateHttpResponseresponse;
110.
111.    privateList<NameValuePair>nameValuePair=newArrayList<NameValuePair>();
```

```
112.
113.    BaseProtocol(){
114.        httpClient=newDefaultHttpClient();
115.    }
116.
117.    /**
118.    *向服务器端发送请求
119.    *
120.    *@paramurl
121.    *@throwsException
122.    */
123.    protectedvoidpack(Stringurl)throwsException{
124.        httpClient=newDefaultHttpClient();
125.        httpRequest=newHttpPost(url);
126.
127.        httpRequest.setEntity(newUrlEncodedFormEntity(nameValuePair));
128.        response=httpClient.execute(httpRequest);
129.    }
130.
131.    /**
132.    *得到返回数据
133.    *
134.    *@paramurl
135.    *@return
136.    *@throwsException
137.    */
138.    protectedvoidparse()throwsException{
139.        //TODO状态处理500200
140.        if(response.getStatusLine().getStatusCode()==200){
141.
142.            BufferedReaderbufferedReader2=newBufferedReader(
143.                    newInputStreamReader(response.getEntity().getContent()));
144.            for(Strings=bufferedReader2.readLine();s!=null;s=bufferedReader2
145.                    .readLine()){
146.                sb.append(s);
147.            }
148.        }
149.    }
150.
151.    /**
152.    *向服务器发送信息
153.    *
154.    *@paramkey
155.    *@paramvalue
156.    */
157.    publicvoidaddNameValuePair(Stringkey,Stringvalue){
158.        nameValuePair.add(newBasicNameValuePair(key,value));
159.    }
160.
```

```
161.     /**
162.     *返回 JSONObject 对象数据模型
163.     *
164.     *@return
165.     *@throwsJSONException
166.     */
167.     publicJSONObjectgetJSON()throwsJSONException{
168.         returnnewJSONObject(sb.toString());
169.     }
170.
171.}
```

接着是登陆协议，在这里我只是模拟登陆使用的一个类，仅供大家参考：

```
1.  packagecom.jclick.protocol;
2.
3.  importorg.json.JSONObject;
4.
5.  importcom.jclick.bean.User;
6.
7.  publicclassLoginProtocolextendsBaseProtocol{
8.
9.  privatefinalstaticStringURL="http://localhost:8080/test/login";
10.
11. publicbooleancheckLogin(Userusr){
12. try{
13. pack(URL);
14. parse();
15. JSONObjectobj=this.getJSON();
16. if(obj.getString("result").equals("failed")){
17. returnfalse;
18. }else{
19. returntrue;
20. }
21. }catch(Exceptione){
22. e.printStackTrace();
23. returnfalse;
24. }
25. }
26.
27. }
28.
29. packagecom.jclick.protocol;
30.
31. importorg.json.JSONObject;
32.
33. importcom.jclick.bean.User;
34.
35. publicclassLoginProtocolextendsBaseProtocol{
36.
```

```
37.     privatefinalstaticStringURL="http://localhost:8080/test/login";
38.
39.     publicbooleancheckLogin(Userusr){
40.         try{
41.             pack(URL);
42.             parse();
43.             JSONObjectobj=this.getJSON();
44.             if(obj.getString("result").equals("failed")){
45.                 returnfalse;
46.             }else{
47.                 returntrue;
48.             }
49.         }catch(Exceptione){
50.             e.printStackTrace();
51.             returnfalse;
52.         }
53.     }
54.
55. }
```

然后是 User 实体类，主要用于保存用户信息：

```
1.  packagecom.jclick.bean;
2.
3.  publicclassUser{
4.  privateStringusername;
5.  privateStringpassword;
6.  publicStringgetUsername(){
7.  returnusername;
8.  }
9.  publicvoidsetUsername(Stringusername){
10. this.username=username;
11. }
12. publicStringgetPassword(){
13. returnpassword;
14. }
15. publicvoidsetPassword(Stringpassword){
16. this.password=password;
17. }
18.
19. }
20.
21. packagecom.jclick.bean;
22.
23. publicclassUser{
24.     privateStringusername;
25.     privateStringpassword;
26.     publicStringgetUsername(){
27.         returnusername;
28.     }
```

```
29.     publicvoidsetUsername(Stringusername){
30.         this.username=username;
31.     }
32.     publicStringgetPassword(){
33.         returnpassword;
34.     }
35.     publicvoidsetPassword(Stringpassword){
36.         this.password=password;
37.     }
38.
39. }
```

最后就是 LoginActivity 里边判断登陆的代码了，详细代码不再贴出来了，仅贴一个判断登陆的代码：

```
1.  privatevoidcheckedData(){
2.  username=((EditText)findViewById(R.id.username)).getText().toString();
3.  password=((EditText)findViewById(R.id.password)).getText().toString();
4.
5.  Useruser=newUser();
6.  user.setUsername(username);
7.  user.setPassword(password);
8.  LoginProtocollogin=newLoginProtocol();
9.  booleanresult=login.checkLogin(user);
10.
11. if(result){SpiderCache.getInstance().setUserSession(user);
12. Toast.makeText(getApplicationContext(),"登录成功",1000).show();
13. Intentintent=newIntent();
14. intent.setClass(LoginActivity.this,WelcomeActivity.class);
15. startActivity(intent);
16. }else{Toast.makeText(LoginActivity.this,"密码或用户名不匹配，请重新输入！",1000).show();
17. }
18. }
19.
20.         privatevoidcheckedData(){
21.             username=((EditText)findViewById(R.id.username)).getText().toString();
22.             password=((EditText)findViewById(R.id.password)).getText().toString();
23.
24.             Useruser=newUser();
25.             user.setUsername(username);
26.             user.setPassword(password);
27.             LoginProtocollogin=newLoginProtocol();
28.             booleanresult=login.checkLogin(user);
29.
30.             if(result){             SpiderCache.getInstance().setUserSession(user);
31.                 Toast.makeText(getApplicationContext(),"登录成功",1000).show();
32.                 Intentintent=newIntent();
33.                 intent.setClass(LoginActivity.this,WelcomeActivity.class);
34.                 startActivity(intent);
35.             }else{              Toast.makeText(LoginActivity.this,"密码或用户名不匹配，请重新
    输入！",1000).show();
```

```
36.                 }
37.             }
```

以上代码为了跟大家分享一下，感觉手机端和服务器双向开发非常过瘾。同时对 Android 的兴趣大大提升，它也没有我们想象中的那么难。