

1.主题概述

1.

使用Tomcat web server，并安装Eclipse新建Java web工程SCOSServer，在该工程下定义源码包“HttpServletesd.scos.servlet”；

在SCOSServer”包esd.scos.servlet下新建类LoginValidator继承HttpServlet，要求如下：

1) 实现 doPost()方法完成 SCOS 客户端登录请求传入的用户名与密码 验证，当验证成功，返回 JSON 串 “ {RESULTCODE:1} ”；否 则返回 “{RESULTCODE:0}”；

2) 实现 doGet()方法，并在该方法中调用 doPost()；

在SCOSServer包esd.scos.servlet下新建类FoodUpdateService继承HttpServlet，要求如下：

1) 实现 doGet () 方法，当 SCOS 客户端请求菜品信息更新时，实现 菜品更新信息发送，详细信息如下；

2) 使用 JSON 封装菜品更新信息内容：更新菜品数量，每个菜品名 称，每个菜品价格，每个菜品类型；

3) 将菜品更新信息以流的形式发送至 SCOS 客户端；

4) 实现 doPost () 方法，并在该方法中调用 doGet () ；

2.

修改E5中SCOS的LoginOrRegister代码，当用户点击“登录”或“注册”按钮时，使用 HttpURLConnection访问SCOSServer，并传入用户输入的用户名和密码；

接收的Servlet类LoginValidator返回信息，当示登录失败，保持原有提示功能不变 SCOSServerRESULTCODE为“1”时表示登录成功，为“0”时表示登录失败；

修改E 5中SCOS的UpdateService代码，在该类中使用HttpURLConnection访问 SCOSServer中FoodUpdateService的doGet()方法；解析返回结果json信息；

3.

UpdateService代码，当有菜品更新时，使用MediaPlayer播放更新提示音，并使用NotificationManager在状态栏提示用户“新品上架：菜品数量”，通知中含有“清除”按钮，当点击清除按钮时，通知消除；当点击通知其他区域时，页面跳转至的MainScreen屏幕

2.步骤

Tomcat web server环境搭建->服务端LoginValidator->服务端FoodUpdateService->客户端LoginOrRegister->客户端UpdateService

[代码Github托管](#)

[报告博客托管](#)

3.实现

1.在SCOSServer”包esd.scos.servlet下新建类LoginValidator继承HttpServlet

```
1  //LoginValidator
2  @WebServlet(name = "LoginValidator")
3  public class LoginValidator extends HttpServlet {
4      private static final Pattern pattern = Pattern.compile("^[A-Za-z1-9_-]+$");
5
6      protected void doPost(HttpServletRequest request,
7      HttpServletResponse response) throws ServletException, IOException {
8
9          response.setContentType("text/html");
10         request.setCharacterEncoding("UTF-8");
11         String name="";
12         String password = "";
13         name = request.getParameter("name");
14         password = request.getParameter("password");
15
16         String result = "";
17
18         if (startValid(name,password)) {
19             result="success";
20         }else{
21             result = "error";
22         }
23         JSONObject jsonObject = new JSONObject();
24         jsonObject.put("result", result);
25         response.getWriter().print(jsonObject);
26     }
```

```

27
28     protected void doGet(HttpServletRequest request,
    HttpServletRequest response) throws ServletException, IOException {
29         doPost(request,response);
30     }
31     private boolean startValid(String name,String password) {
32         return pattern.matcher(name.trim()).matches() &&
    pattern.matcher(password.trim()).matches();
33     }
34 }
35

```

2.在SCOSServer包esd.scos.servlet下新建类FoodUpdateService继承HttpServlet

```

1  //FoodUpdateService
2  @WebServlet(name = "FoodUpdateService")
3  public class FoodUpdateService extends HttpServlet {
4      protected void doPost(HttpServletRequest request,
    HttpServletRequest response) throws ServletException, IOException {
5          doGet(request,response);
6      }
7
8      protected void doGet(HttpServletRequest request,
    HttpServletRequest response) throws ServletException, IOException {
9          response.setContentType("text/html;charset=utf-8");
10         request.setCharacterEncoding("UTF-8");
11
12         JSONArray jsonArray = new JSONArray();
13
14         jsonArray.add(updateFoods("东北家拌凉菜",10,10,"冷菜"));
15         jsonArray.add(updateFoods("椒油素鸡",20,10,"冷菜"));
16         jsonArray.add(updateFoods("浇汁豆腐",30,10,"冷菜"));
17         jsonArray.add(updateFoods("开胃泡菜",40,10,"冷菜"));
18         jsonArray.add(updateFoods("凉拌海带丝",50,10,"冷菜"));
19         jsonArray.add(updateFoods("凉拌黄瓜",60,10,"冷菜"));
20         jsonArray.add(updateFoods("卤牛肉",70,10,"冷菜"));
21         jsonArray.add(updateFoods("青椒拌干丝",80,10,"冷菜"));
22         jsonArray.add(updateFoods("干煸豆角",0,10,"热菜"));
23
24         response.getWriter().print(jsonArray);
25     }
26     private JSONObject updateFoods(String name,int last,int price,
    String category){
27         JSONObject jsonObject = new JSONObject();

```

```

28         jsonObject.put("name", name);
29         jsonObject.put("last", last);
30         jsonObject.put("price", price);
31         jsonObject.put("category", category);
32         return jsonObject;
33     }
34 }
35

```

3.客户端LoginOrRegister修改

使用URLConnection访问SCOSServer，并传入用户输入的用户名和密码；

```

1  //登录，注册，返回，按钮功能
2  private class OnClick implements View.OnClickListener{
3
4      @Override
5      public void onClick(View view){
6
7          final User loginUser = new User();
8          Intent intent = null;
9          final SharedPreferences.Editor editor =
10 getSharedPreferences("User",MODE_PRIVATE).edit();
11          Thread thread = null;
12          switch (view.getId()){
13              case R.id.loginButton:
14                  showProgress(R.string.registering);
15
16                  thread = new Thread(new Runnable() {
17                      @Override
18                      public void run() {
19                          //sendJson();
20                          String url =
21 MyPOSTHttpURLConnection.BASE_URL + "/LoginValidator";
22                          Map<String,String> parms = new
23 HashMap<String, String>();
24                          String name =
25 etName.getText().toString();
26                          String password =
27 etPassword.getText().toString();
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

24         parms.put("name",name);
25         parms.put("password",password);
26
27
28         String result =
MyPOSTURLConnection.getContextByHttp(url,parms);
29         Log.d("result",result);
30
31
32         Message msg = new Message();
33         msg.what = 20;
34         Bundle data = new Bundle();
35
36         data.putString("result",result);
37         msg.setData(data);
38
39         handler.sendMessage(msg);
40     }
41
42
43     Handler handler = new Handler(){
44         @Override
45         public void handleMessage(Message msg) {
46             if(msg.what==20){
47                 Bundle data = msg.getData();
48                 String key =
data.getString("result");
49                 Log.d("key",key);
50                 try{
51                     JSONObject json = new
JSONObject(key);
52                     String result =
(String)json.get("result");
53                     if(result.equals("success")){
54
55                         mButton_Enter.setVisibility(View.VISIBLE);
56
57                         mButton_Login.setVisibility(View.INVISIBLE);
58
59                         loginUser.setOldUser(false);
60
61                         loginUser.setUserName(etName.getText().toString());
62
63                         loginUser.setPassword(etPassword.getText().toString());

```

```

59     Toast.makeText(LoginOrRegister.this,"欢迎您成为 SCOS 新用户",Toast.LENGTH_SHORT).show();
60
61
62         String string =
mButton_Enter.getText().toString();
63
64     editor.putString("userName",string);
65
66     editor.putInt("loginState",1);
67
68         editor.commit();
69
70     Toast.makeText(LoginOrRegister.this,"注册成功",Toast.LENGTH_LONG).show();
71
72         }
73     else{
74
75         Toast.makeText(LoginOrRegister.this,"输入内容不符合规则",Toast.LENGTH_SHORT).show();
76
77         //Toast.makeText(LoginOrRegister.this,"注册失败",Toast.LENGTH_LONG).show();
78
79         }
80
81     }
82
83     catch (Exception e){
84         e.printStackTrace();
85     }
86
87     }
88
89     };
90
91     });
92
93     thread.start();
94
95     dismissProgress();
96     break;
97
98     case R.id.enterButton:
99         showProgress(R.string.logining);
100
101         thread = new Thread(new Runnable() {

```

```

93         @Override
94         public void run() {
95
96             //sendJson();
97             String url =
MyPOSTHttpURLConnection.BASE_URL + "/LoginValidator";
98             Map<String,String > parms = new
HashMap<String, String>();
99             String name =
etName.getText().toString();
100             String password =
etPassword.getText().toString();
101             parms.put("name",name);
102             parms.put("password",password);
103             String result =
MyPOSTHttpURLConnection.getContextByHttp(url,parms);
104             Log.d("result",result);
105             Message msg = new Message();
106             msg.what = 21;
107             Bundle data = new Bundle();
108             data.putString("result",result);
109             msg.setData(data);
110
111             handler.sendMessage(msg);
112         }
113
114         Handler handler = new Handler(){
115             @Override
116             public void handleMessage(Message msg) {
117                 if(msg.what==21){
118                     Bundle data = msg.getData();
119                     String key =
data.getString("result");
120                     Log.d("key",key);
121                     try{
122                         JSONObject json = new
JSONObject(key);
123                         String result =
(String)json.get("result");
124                         if(result.equals("success")){
125                             loginUser.setOldUser(true);
126                             loginUser.setUserName(etName.getText().toString());

```

```
127     loginUser.setPassword(etPassword.getText().toString());
128
129     Toast.makeText(LoginOrRegister.this, "已登
130     录", Toast.LENGTH_SHORT).show();
131
132     //
133     String string =
134     mButton_Enter.getText().toString();
135
136     editor.putString("userName", string);
137
138     editor.putInt("loginState", 1);
139
140     editor.commit();
141
142     Intent intent = new
143     Intent(LoginOrRegister.this, MainScreen.class);
144
145     //intent.putExtra("from", "LoginSuccess");
146
147     startActivity(intent);
148
149     Toast.makeText(LoginOrRegister.this, "登录成
150     功", Toast.LENGTH_LONG).show();
151
152     }
153     else{
154
155     Toast.makeText(LoginOrRegister.this, "输入内容不符合规
156     则", Toast.LENGTH_SHORT).show();
157
158     }
159
160     }
161
162     }
163
164     }
165
166     }
167
168     }
169
170     }
171
172     }
173
174     }
175
176     }
177
178     }
179
180     }
181
182     }
183
184     }
185
186     }
187
188     }
189
190     }
191
192     }
193
194     }
195
196     }
197
198     }
199
200     }
201
202     }
203
204     }
205
206     }
207
208     }
209
210     }
211
212     }
213
214     }
215
216     }
217
218     }
219
220     }
221
222     }
223
224     }
225
226     }
227
228     }
229
230     }
231
232     }
233
234     }
235
236     }
237
238     }
239
240     }
241
242     }
243
244     }
245
246     }
247
248     }
249
250     }
251
252     }
253
254     }
255
256     }
257
258     }
259
260     }
261
262     }
263
264     }
265
266     }
267
268     }
269
270     }
271
272     }
273
274     }
275
276     }
277
278     }
279
280     }
281
282     }
283
284     }
285
286     }
287
288     }
289
290     }
291
292     }
293
294     }
295
296     }
297
298     }
299
300     }
301
302     }
303
304     }
305
306     }
307
308     }
309
310     }
311
312     }
313
314     }
315
316     }
317
318     }
319
320     }
321
322     }
323
324     }
325
326     }
327
328     }
329
330     }
331
332     }
333
334     }
335
336     }
337
338     }
339
340     }
341
342     }
343
344     }
345
346     }
347
348     }
349
350     }
351
352     }
353
354     }
355
356     }
357
358     }
359
360     }
361
362     }
363
364     }
365
366     }
367
368     }
369
370     }
371
372     }
373
374     }
375
376     }
377
378     }
379
380     }
381
382     }
383
384     }
385
386     }
387
388     }
389
390     }
391
392     }
393
394     }
395
396     }
397
398     }
399
400     }
401
402     }
403
404     }
405
406     }
407
408     }
409
410     }
411
412     }
413
414     }
415
416     }
417
418     }
419
420     }
421
422     }
423
424     }
425
426     }
427
428     }
429
430     }
431
432     }
433
434     }
435
436     }
437
438     }
439
440     }
441
442     }
443
444     }
445
446     }
447
448     }
449
450     }
451
452     }
453
454     }
455
456     }
457
458     }
459
460     }
461
462     }
463
464     }
465
466     }
467
468     }
469
470     }
471
472     }
473
474     }
475
476     }
477
478     }
479
480     }
481
482     }
483
484     }
485
486     }
487
488     }
489
490     }
491
492     }
493
494     }
495
496     }
497
498     }
499
500     }
501
502     }
503
504     }
505
506     }
507
508     }
509
510     }
511
512     }
513
514     }
515
516     }
517
518     }
519
520     }
521
522     }
523
524     }
525
526     }
527
528     }
529
530     }
531
532     }
533
534     }
535
536     }
537
538     }
539
540     }
541
542     }
543
544     }
545
546     }
547
548     }
549
550     }
551
552     }
553
554     }
555
556     }
557
558     }
559
560     }
561
562     }
563
564     }
565
566     }
567
568     }
569
570     }
571
572     }
573
574     }
575
576     }
577
578     }
579
580     }
581
582     }
583
584     }
585
586     }
587
588     }
589
590     }
591
592     }
593
594     }
595
596     }
597
598     }
599
600     }
601
602     }
603
604     }
605
606     }
607
608     }
609
610     }
611
612     }
613
614     }
615
616     }
617
618     }
619
620     }
621
622     }
623
624     }
625
626     }
627
628     }
629
630     }
631
632     }
633
634     }
635
636     }
637
638     }
639
640     }
641
642     }
643
644     }
645
646     }
647
648     }
649
650     }
651
652     }
653
654     }
655
656     }
657
658     }
659
660     }
661
662     }
663
664     }
665
666     }
667
668     }
669
670     }
671
672     }
673
674     }
675
676     }
677
678     }
679
680     }
681
682     }
683
684     }
685
686     }
687
688     }
689
690     }
691
692     }
693
694     }
695
696     }
697
698     }
699
700     }
701
702     }
703
704     }
705
706     }
707
708     }
709
710     }
711
712     }
713
714     }
715
716     }
717
718     }
719
720     }
721
722     }
723
724     }
725
726     }
727
728     }
729
730     }
731
732     }
733
734     }
735
736     }
737
738     }
739
740     }
741
742     }
743
744     }
745
746     }
747
748     }
749
750     }
751
752     }
753
754     }
755
756     }
757
758     }
759
760     }
761
762     }
763
764     }
765
766     }
767
768     }
769
770     }
771
772     }
773
774     }
775
776     }
777
778     }
779
780     }
781
782     }
783
784     }
785
786     }
787
788     }
789
790     }
791
792     }
793
794     }
795
796     }
797
798     }
799
800     }
801
802     }
803
804     }
805
806     }
807
808     }
809
810     }
811
812     }
813
814     }
815
816     }
817
818     }
819
820     }
821
822     }
823
824     }
825
826     }
827
828     }
829
830     }
831
832     }
833
834     }
835
836     }
837
838     }
839
840     }
841
842     }
843
844     }
845
846     }
847
848     }
849
850     }
851
852     }
853
854     }
855
856     }
857
858     }
859
860     }
861
862     }
863
864     }
865
866     }
867
868     }
869
870     }
871
872     }
873
874     }
875
876     }
877
878     }
879
880     }
881
882     }
883
884     }
885
886     }
887
888     }
889
890     }
891
892     }
893
894     }
895
896     }
897
898     }
899
900     }
901
902     }
903
904     }
905
906     }
907
908     }
909
910     }
911
912     }
913
914     }
915
916     }
917
918     }
919
920     }
921
922     }
923
924     }
925
926     }
927
928     }
929
930     }
931
932     }
933
934     }
935
936     }
937
938     }
939
940     }
941
942     }
943
944     }
945
946     }
947
948     }
949
950     }
951
952     }
953
954     }
955
956     }
957
958     }
959
960     }
961
962     }
963
964     }
965
966     }
967
968     }
969
970     }
971
972     }
973
974     }
975
976     }
977
978     }
979
980     }
981
982     }
983
984     }
985
986     }
987
988     }
989
990     }
991
992     }
993
994     }
995
996     }
997
998     }
999
1000    }
```



```

159             editor.putString("userName",string);
160             editor.putInt("loginState",0);
161             editor.commit();
162             intent = new
Intent(LoginOrRegister.this,MainScreen.class);
163             startActivity(intent);
164             break;
165         }
166     }
167 }

```

```

1  private void sendJson(){
2      //boolean loginValidate = false;
3      Log.d("url","成功");
4      try{
5          JSONObject jsonObject = new JSONObject();
6          jsonObject.put("name",etName.getText().toString());
7
8          jsonObject.put("password",etPassword.getText().toString());
9          URL url = new URL("http://192.168.1.144:8080");
10         HttpURLConnection conn =
(HttpURLConnection)url.openConnection();
11         conn.setDoOutput(true);
12         conn.setDoInput(true);
13         conn.setUseCaches(false);
14         conn.setRequestMethod("POST");
15         conn.setRequestProperty("Connection", "Keep-Alive");
16         conn.setRequestProperty("Charset", "UTF-8");
17         conn.setRequestProperty("contentType",
"application/json");
18         conn.connect();
19         OutputStreamWriter writer = new
OutputStreamWriter(conn.getOutputStream());
20         // 发送给服务器
21         writer.write(jsonObject.toString());
22         writer.flush();
23         writer.close();
24         //接收服务器返回信息
25         BufferedReader br = new BufferedReader(new
InputStreamReader(conn.getInputStream(),"UTF-8")) ;
26         String line;
27         StringBuilder sb = new StringBuilder();
28         while ((line = br.readLine()) != null) {
sb.append(line);

```

```

29         }
30
31         int RESULTCODE = 0;
32         RESULTCODE = sb.toString().indexOf("RESULTCODE");
33
34         if(RESULTCODE == 1){
35             Log.d("RESULTCODE",RESULTCODE+"");
36         }
37         else if(RESULTCODE == 0){
38             Log.d("RESULTCODE",RESULTCODE+"");
39         }
40         else{
41             Log.d("RESULTCODE","no");
42         }
43         conn.disconnect();
44     }catch(Exception exception){
45         exception.printStackTrace();
46     }
47 }

```

4.UpdateService代码

当有菜品更新时，使用MediaPlayer播放更新提示音，并使用NotificationManager在状态栏提示用户“新品上架：菜品数量”，通知中含有“清除”按钮，当点击清除按钮时，通知消除；当点击通知其他区域时，页面跳转至的屏幕

```

1     @TargetApi(Build.VERSION_CODES.O)
2     @Override
3     protected void onHandleIntent(Intent intent) {
4         Log.d(TAG, "onHandleIntent");
5         NotificationChannel channel = new
6         NotificationChannel("channel_1","123",NotificationManager.IMPORTANCE_L
7         OW);
8         manager=(NotificationManager)
9         getSystemService(Context.NOTIFICATION_SERVICE);
10        manager.createNotificationChannel(channel);
11        Notification.Builder builder = new
12        Notification.Builder(this,"channel_1");
13        builder.setAutoCancel(true);
14        builder.setSmallIcon(R.drawable.food_cold_dbjlbc);//设置图标
15        builder.setTicker("!新品上架!");//手机状态栏的提示
16        builder.setContentTitle("新品上架");//设置标题
17        builder.setContentText("东北家拌凉菜,10,冷菜");//设置通知内容

```

```

14         builder.setWhen(System.currentTimeMillis()); //设置通知时间
15         Uri defaultSoundUri =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
16         builder.setSound(defaultSoundUri);
17
18         Intent intentCancel = new Intent(this,
DeviceStartedListener.class);
19
        intentCancel.setAction("scos.intent.action.CLOSE_NOTIFICATION");
20         PendingIntent pendingIntentCancel =
PendingIntent.getBroadcast(this, 0, intentCancel,
PendingIntent.FLAG_ONE_SHOT);
21         builder.setDeleteIntent(pendingIntentCancel);
22
23         Intent intent_to = new Intent(this, MainScreen.class);
24         PendingIntent pendingIntent = PendingIntent.getActivity(this,
0, intent_to, 0);
25         builder.setContentIntent(pendingIntent); //点击后的意图
26         builder.setDefaults(Notification.DEFAULT_LIGHTS); //设置指示灯
27         builder.setDefaults(Notification.DEFAULT_SOUND); //设置提示声音
28         builder.setDefaults(Notification.DEFAULT_VIBRATE); //设置震动
29
        MediaPlayer mediaPlayer =
MediaPlayer.create(this, Settings.System.DEFAULT_NOTIFICATION_URI);
30         mediaPlayer.start(); //通知铃声
31         mediaPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
32             @Override
33             public void onCompletion(MediaPlayer mp) {
34                 mp.release();
35             }
36         });
37         Notification notification = builder.build(); //4.1以上, 以下要用
getNotification()
38         startForeground(Notification_ID, notification); //开启前台服务
39         manager.notify(Notification_ID, notification);
40
41     }
42     public static void collapseStatusBar(Context context) {
43         try {
44             Object statusBarManager =
context.getSystemService("statusbar");
45             Method collapse;
46             if (Build.VERSION.SDK_INT <= 16) {

```

```

47         collapse =
statusBarManager.getClass().getMethod("collapse");
48     } else {
49         collapse =
statusBarManager.getClass().getMethod("collapsePanels");
50     }
51     collapse.invoke(statusBarManager);
52 } catch (Exception localException) {
53     localException.printStackTrace();
54 }
55 }
56 }

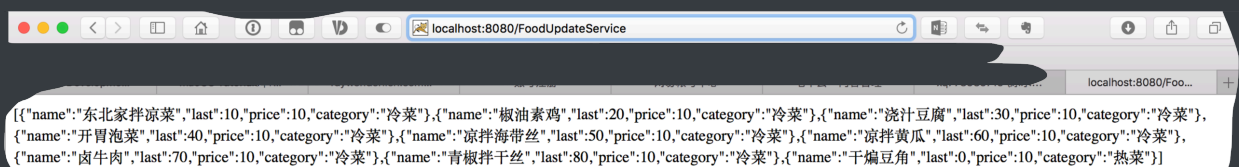
```

4.结论

本次练习：

1. 学会了Servlet的搭建方法，如何构造response，如何接收request，post和get的区别。并使用tomcat进行部署和调试。
2. 学会了Json的数据构造方法，并可以通过Servlet接收和返回。
3. 熟悉了URLConnection的使用，熟练了在客户端进行get和post请求并接收数据的操作。
4. 学会了Json的数据解析方法，并可以在客户端接收到数据之后进行相应的解析。
5. 掌握了notification的特殊用法：添加按钮，并使用广播关闭该通知。
6. 初步掌握了MediaPlayer的用法，播放一个简单的资源。

1.效果图



服务器发送的Json数据

6:48

食物

冷菜

热菜

酒水

海鲜



菜名:东北家拌凉菜

单价:10

已买:0

库存:10

+

-



菜名:椒油素鸡

单价:10

已买:0

库存:20

+

-



菜名:浇汁豆腐

单价:10

已买:0

库存:30

+

-



菜名:开胃泡菜

单价:10

已买:0

库存:40

+

-



菜名:凉拌海带丝

单价:10

已买:0

库存:50

+

-



菜名:凉拌黄瓜

单价:10

已买:0

库存:60

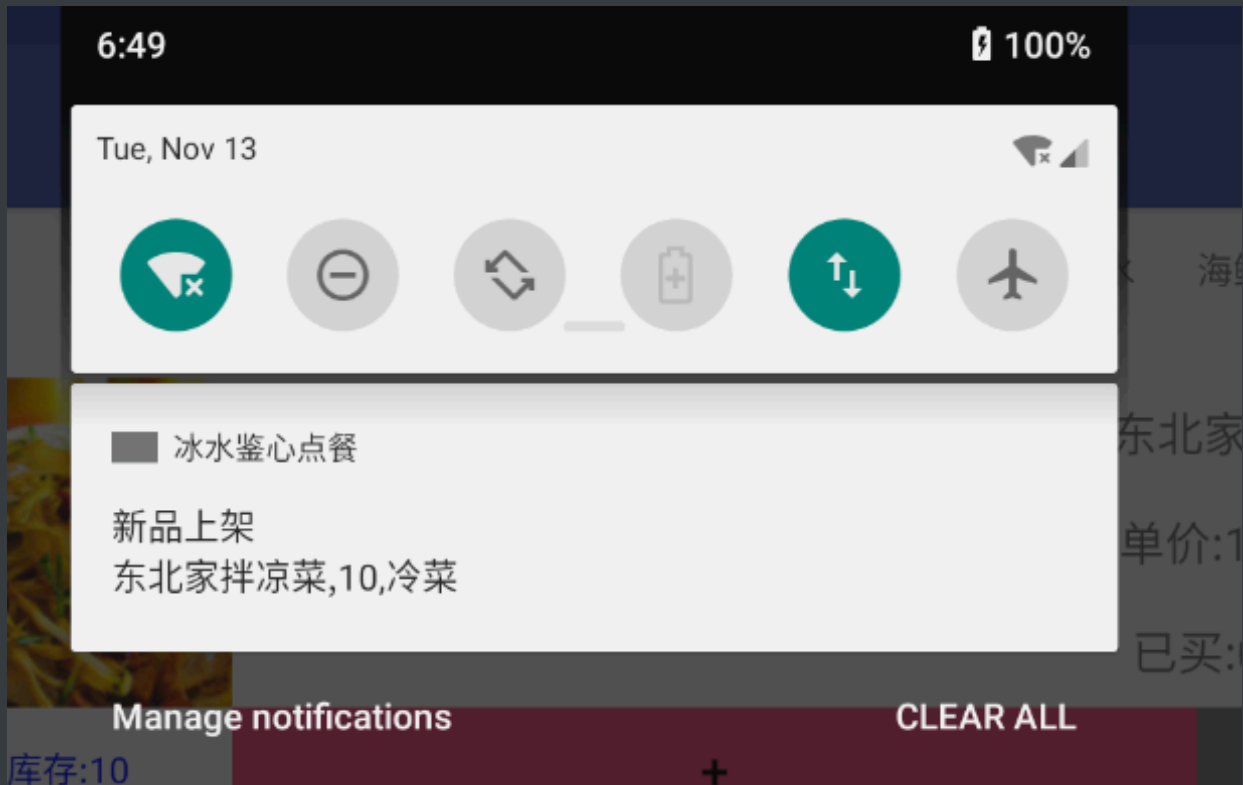
+

-

接收到数据，并更新，显示通知

```
D/EGL_emulation: eglMakeCurrent: 0xe6285300: ver 3 0 (tinfo 0xe6283690)
D/实时更新开启: 实时更新开启
D/EGL_emulation: eglMakeCurrent: 0xe6285300: ver 3 0 (tinfo 0xe6283690)
D/ContentValues: onStartCommand
D/ContentValues: onBind
D/EGL_emulation: eglMakeCurrent: 0xe6285300: ver 3 0 (tinfo 0xe6283690)
D/result: [{"name":"东北家拌凉菜","last":10,"price":10,"category":"冷菜"}, {"name":"椒油素鸡","last":20,"price":10,"category":"冷菜"}, {"name":"浇汁豆腐","last":30,"price":10,"category":"冷菜"}]
D/name: 东北家拌凉菜
D/last: 10
D/price: 10
D/category: 冷菜
D/name: 椒油素鸡
D/last: 20
D/price: 10
D/category: 冷菜
D/EGL_emulation: eglMakeCurrent: 0xe6285300: ver 3 0 (tinfo 0xe6283690)
D/name: 浇汁豆腐
D/last: 30
D/price: 10
D/category: 冷菜
D/name: 开胃泡菜
D/last: 40
D/price: 10
D/category: 冷菜
D/name: 凉拌海带丝
D/last: 50
D/price: 10
D/category: 冷菜
D/name: 凉拌黄瓜
D/last: 60
D/price: 10
D/category: 冷菜
D/name: 卤牛肉
D/last: 70
D/price: 10
D/category: 冷菜
D/name: 青椒拌干丝
D/last: 80
D/price: 10
D/category: 冷菜
D/name: 干煸豆角
D/last: 0
D/price: 10
D/category: 热菜
```

接收到的Json数据，并解析加入log



通知

2.说明 XML 不同解析方式，比较各种方式的优缺点

1.DOM是用与平台和语言无关的方式表示XML文档的官方W3C标准。DOM是以层次结构组织的节点或信息片断的集合。这个层次结构允许开发人员在树中寻找特定信息。分析该结构通常需要加载整个文档和构造层次结构，然后才能做任何工作。由于它是基于信息层次的，因而DOM被认为是基于树或基于对象的。

【优点】 ①允许应用程序对数据和结构做出更改。②访问是双向的，可以在任何时候在树中上下导航，获取和操作任意部分的数据。 【缺点】 ①通常需要加载整个XML文档来构造层次结构，消耗资源大。

2.SAX处理的优点非常类似于流媒体的优点。分析能够立即开始，而不是等待所有的数据被处理。而且，由于应用程序只是在读取数据时检查数据，因此不需要将数据存储在内存中。这对于大型文档来说是个巨大的优点。事实上，应用程序甚至不必解析整个文档；它可以在某个条件得到满足时停止解析。一般来说，SAX还比它的替代者DOM快许多。

【优点】 ①不需要等待所有数据都被处理，分析就能立即开始。②只在读取数据时检查数据，不需要保存在内存中。③可以在某个条件得到满足时停止解析，不必解析整个文档。④效率和性能较高，能解析大于系统内存的文档。

【缺点】 ①需要应用程序自己负责TAG的处理逻辑（例如维护父/子关系等），文档越复杂程序就越复杂。②单向导航，无法定位文档层次，很难同时访问同一文档的不同部分数据，不支持XPath。

3.JDOM与DOM主要有两方面不同。首先，JDOM仅使用具体类而不使用接口。这在某些方面简化了API，但是也限制了灵活性。第二，API大量使用了Collections类，简化了那些已经熟悉这些类的Java开发者的使用。

【优点】 ①使用具体类而不是接口，简化了DOM的API。②大量使用了Java集合类，方便了Java开发人员。

【缺点】 ①没有较好的灵活性。②性能较差。

4.DOM4J使用接口和抽象基本类方法。DOM4J大量使用了API中的Collections类，但是在许多情况下，它还提供一些替代方法以允许更好的性能或更直接的编码方法。直接好处是，虽然DOM4J付出了更复杂的API的代价，但是它提供了比JDOM大得多的灵活性。

【优点】 ①大量使用了Java集合类，方便Java开发人员，同时提供一些提高性能的替代方法。②支持XPath。③有很好的性能。

【缺点】 ①大量使用了接口，API较为复杂。

3.对比 JSON 与 XML 作为数据封装方式的优缺点

- 可读性：JSON和XML的可读性可谓不相上下，一边是建议的语法，一边是规范的标签形式，很难分出胜负。
- 可扩展性：XML天生有很好的扩展性，JSON当然也有，没有什么是XML能扩展，JSON不能的。
- 编码难度：XML有丰富的编码工具，比如Dom4j、JDom等，JSON也有json.org提供的工具，但是JSON的编码明显比XML容易许多，即使不借助工具也能写出JSON的代码，可是要写好XML就不太容易了。
- 解码难度：XML的解析得考虑子节点父节点，让人头昏眼花，而JSON的解析难度几乎为0。
- 流行度：XML已经被业界广泛的使用，而JSON才刚刚开始，但是在Ajax这个特定的领域，未来的发展一定是XML让位于JSON。

5.参考

参考书籍

- 1 【1】《Head First Android开发》
- 2 【2】《Android第一行代码》
- 3 【3】《Head First Java》
- 4 【4】《Java编程思想》
- 5 【5】《Java核心技术》
- 6 【6】《Android编程权威指南》

参考链接

- 【1】[Android的视频教程](#)
- 【2】[提供帮助的社区](#)
- 【3】[技术博客CSDN](#)
- 【4】[技术博客开源中国](#)
- 【5】[张新强的CSDN博客](#)
- 【6】[Hongyang](#)
- 【7】[Android后台service限制](#)