Android应用软件设计

1.主题概述

1.

> 在es.source.code.service子包中新建类ServerObserverService：
>
> 在 ServerObserverService 类中新建 Handler 对象 cMessageHandler， 重写方法 handleMessage()；当传入的 Message 属性 what 值为 1 时：启动 多线程模拟接收服务器传回菜品库存信息（菜名称，库存量），每次接收 到新数据，判断 SCOS app 进程是否在运行状态，如果为运行状态则向 SCOS app 进程发送 Message，其 what 值为 10，并在该 Message 中携带 收到的库存信息（菜名称，库存量），多线程休眠频率为 300 ms；
>
> 当对象 cMessageHandler 使用 handleMessage()方法收到 Message 属性 what 值为 0 时，则关闭模拟接收服务器传回菜品库存信息的多线程，在 SCOS 的配置文件 AndroidManifest.xml 中注册 ServerObserverService 服务组件，并将其属性“ android:process”定义为 “es.source.code.observerservice”。

2.

> 修改E 4中的FoodView代码，将食物列表ListView中每个菜项信息增加一个库存，使用粗体蓝色字体显示当前该菜品库存量。

3.

> 修改E4中的FoodView代码，将当前屏幕ActionBar菜单项增加新操作作为“启动实时更新”，当用户点击该Action时：
>
> 启动 ServerObserverService 服务，并向 ServerObserverService 发送 信息 Message 属性 what 值为 1；
>
> 在 FoodView 中新建 sMessageHandler 对象， 重写方法 handleMessage()；当传入的 Message 属性 what 值为 10 时，解析该 Message 携带菜品库存信息（菜名称，库存量），并根据该值，更新 FoodView 菜 项信息；
>
> “启动实时更新”Action 被点击后，将 Action 标签修改为“停止实 时更新”；
>
> 当在 FoodView 界面中点击“停止实时更新” Action 时， 向 ServerObserverService 发送 Message 属性 what 值为 0；

4.

> 在SCOS的es.source.code.service中新建类UpdateService， 继承IntentService，并重写onHandleIntent()方法，要求如下：
>
> 模拟检查服务器是否有菜品种更新信息， 如有更新则使用 NotificationManager 发送状态栏通知，通知内容为“新品上架：菜名，价 格，类型”；
>
> 当点击该通知时，屏幕跳转至 FoodDetailed 界面，显示该菜品详情；
>
> 在 SCOS 工程的 AndroidManifest.xml 注册该服务组件；

5.

> 修改E4的SCOS包结构，增加es.source.code.br子包在 es.source.code.br 子包中新建类 DeviceStartedListener， 继承 BroadcastReceiver 父类，并重写 onReceive()方法：
>
> 当 onReceive()方法接收到设备开机广播时，启动 UpdateService 服务；
>
> 在 SCOS 工程的 AndroidManifest.xml 注册该广播接收器组件；

2.步骤

重写父类方法onBind()->ListView中每个菜项信息增加一个库存->ActionBar菜单项增加新操作为“启动实时更新”->UpdateService方法onHandleIntent()->DeviceStartedListener方法onReceive()

代码Github托管

报告博客托管

3.实现

1.在es.source.code.service子包中新建类ServerObserverService

```
//ServerObserverService
public class ServerObserverService extends Service {

    private myHandler cMessageHandler;
    private Messenger mMessenger;
    private Messenger cMessenger;
    private Thread thread;
    public ServerObserverService() {
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Log.d(TAG,"onStartCommand");

        return super.onStartCommand(intent, flags, startId);

    }

    @Override
    public IBinder onBind(Intent intent) {
        Log.d(TAG,"onBind");

        Thread thread = null;
        thread = new Thread(new Runnable() {
            @Override
            public void run() {
                //sendJson();
                String url = MyPOSTHttpURLConnection.BASE_URL + "/FoodUpdateService";
                Map<String,String > parms = new HashMap<String, String>();
                String result = MyPOSTHttpURLConnection.getContextByHttp(url,parms);
                Log.d("result",result);
                try{
                    JSONArray json = new JSONArray(result);

                    for(int i = 0; i < json.length(); i++){

                        String name = json.getJSONObject(i).getString("name");
                        int last = json.getJSONObject(i).getInt("last");
                        Log.d("name",name);
                        Log.d("last",""+last);
                        int price = json.getJSONObject(i).getInt("price");
                        String category = json.getJSONObject(i).getString("category");
                        Log.d("price",""+price);
                        Log.d("category",category);

                        EventBus.getDefault().post(new MessageEvent(name,last,price));


                    }
                }catch (Exception e){
                    e.printStackTrace();
                }

            }

        });
        thread.start();
        // TODO: Return the communication channel to the service.

        return null;

    }

    @Override
    public boolean onUnbind(Intent intent) {
        Log.d(TAG,"onUnbind");
        return super.onUnbind(intent);
    }

    @Override
    public void onDestroy() {
        Log.d("onDestroy","onDestroy");
        //thread.interrupt();
        super.onDestroy();
    }

    private class myHandler extends Handler {
        private IBinder iBinder;
        private Context mContext;
        @Override
        public void handleMessage(Message msg) {
            final Message message = new Message();
            switch(msg.what){
                case 0:
                    //关闭多线程
                    Log.d(TAG,"多线程关");
                    break;
                case 1:
                    //启动多线程
                    Thread thread = new Thread(new Runnable() {
                        @Override
                        public void run() {
                            Log.d(TAG,"Thread");

```

```java
                            //如果app运行
                            try {
                                Thread.sleep(300);
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                            Bundle bundle = new Bundle();
                            bundle.putString("name","东北家拌凉菜");
                            bundle.putInt("last",100);
                            message.setData(bundle);
                            message.what = 10;
                            Log.d(TAG,"线程运行中");
                            sendMessage(message);

                        }
                    });
                    thread.start();//调试
                    Log.d(TAG,"多线程开");

                    break;
                default:
                    //
            }
            super.handleMessage(msg);
        }
    }
}
```

2.修改FoodView代码

> 将食物列表ListView中每个菜项信息增加一个TextView，使用粗体蓝色字体显示当前该菜品库存量，tv_4表示

```java
//ListViewAdapter添加TextView显示库存
public class ListViewAdapter extends BaseAdapter {
    private Context mContext;
    private LayoutInflater mLayoutInflater;
    public List<Food> list;
    private ViewHolder holder = null;
    private int position;
    private Toast toast= null;

    private long lastShowTime = 0l;
    private String lastShowMsg = null;
    private String curShowMsg = null;
    private final int TOAST_DURATION = 20000;


    public ListViewAdapter(Context context,List<Food> list){
        this.mContext = context;
        this.mLayoutInflater = LayoutInflater.from(context);
        this.list = list;


    }
    @Override
    public int getCount() {
        return list.size();
    }

    @Override
    public Object getItem(int position) {
        return list.get(position);
    }

    @Override
    public long getItemId(int position) {
        return position;
    }
    //重要的方法
    static class ViewHolder{

        public ImageView imageView;
        public TextView tv_1,tv_2,tv_3,tv_4;
        public Button bt_1,bt_2;
    }
    @Override
    public View getView(final int position, View convertView, ViewGroup parent) {
        final Food food = list.get(position);
        this.position = position;
        if(convertView==null){
            convertView = mLayoutInflater.inflate(R.layout.layout_list_item,null);
            holder = new ViewHolder();
            holder.imageView = (ImageView)convertView.findViewById(R.id.layout_list_item_im_1);
            holder.tv_1 = (TextView)convertView.findViewById(R.id.layout_list_item_tv_1);
            holder.tv_2 = (TextView)convertView.findViewById(R.id.layout_list_item_tv_2);
            holder.tv_3 = (TextView)convertView.findViewById(R.id.layout_list_item_tv_3);
            holder.tv_4 = (TextView)convertView.findViewById(R.id.layout_list_item_tv_4);
            holder.bt_1 = (Button)convertView.findViewById(R.id.layout_list_item_bt_1);
            holder.bt_2 = (Button)convertView.findViewById(R.id.layout_list_item_bt_2);
            convertView.setTag(holder);

        }
        else{
            holder = (ViewHolder)convertView.getTag();
        }
        //给控件赋值
        holder.tv_1.setText("菜名:"+list.get(position).getName());
        holder.tv_2.setText("单价:"+list.get(position).getPrice());
        holder.tv_3.setText("已买:"+list.get(position).getCount());
        holder.tv_4.setText("库存:"+ list.get(position).getLast());
        holder.tv_4.setTextColor(Color.BLUE);//蓝色
        holder.bt_1.setText("+");
        holder.bt_2.setText("-");
        //food.setState(true);
        holder.imageView.setImageResource(food.getImageResourceId());

        final Button btn_1 = (Button) convertView.findViewById(R.id.layout_list_item_bt_1);
        final Button btn_2 = (Button) convertView.findViewById(R.id.layout_list_item_bt_2);
        final TextView tv_1 = (TextView) convertView.findViewById(R.id.layout_list_item_tv_1);
        final TextView tv_2 = (TextView) convertView.findViewById(R.id.layout_list_item_tv_1);
        final TextView tv_3 = (TextView) convertView.findViewById(R.id.layout_list_item_tv_3);
        final TextView tv_4 = (TextView)convertView.findViewById(R.id.layout_list_item_tv_4);
        holder.bt_1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = null;
                // 有库存
                if (food.isState()) {
                    if(!food.getInlist()){
                        ordering.add(food);
                        food.addcount();
                        food.setInlist(true);
                    }
                    else{
                        Food temp = ordering.get(ordering.indexOf(food));
                        temp.setCount(temp.getCount()+1);
                        ordering.set(ordering.indexOf(food),temp);
                    }
                    //Toast

                    Util.showToast(mContext, "已点餐,已买:"+food.getCount()+"个"+food.getName());
                    food.setLast(food.getLast()-1);
                    tv_1.setText("菜名:"+food.getName());
                    tv_2.setText("单价:"+food.getPrice());
                    tv_3.setText("已买:"+food.getCount());
                    tv_4.setText("库存:"+food.getLast());
                }
                else{
                }
            }
        });

        holder.bt_2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = null;
                if(food.isState()){

                    if(food.getCount()>0||food.getInlist())
                    food.reducecount();
                    if(food.getCount()==0){
                        ordering.remove(food);
                        food.setInlist(false);
                    }
                    //Toast

                    Util.showToast(mContext, "已取消,还剩:"+food.getCount()+"个"+food.getName());
                    food.setLast(food.getLast()+1);
                    tv_1.setText("菜名:"+food.getName());
                    tv_2.setText("单价:"+food.getPrice());
                    tv_3.setText("已买:"+food.getCount());
                    tv_4.setText("库存:"+food.getLast());
                }
                else{
                }

            }
        });
        return convertView;
    }


}
```

3. ActionBar菜单项增加新操作为“启动实时更新”

```
//启动实时更新
case R.id.update:
                //服务代码
                intent = new Intent(this,ServerObserverService.class);

                if(item.getTitle().equals("启动实时更新")){

                    Log.d("实时更新开启","实时更新开启");

                    //thread.start();

                    startService(intent);

 if(bindService(intent,connection,BIND_AUTO_CREATE)){
                        Toast.makeText(FoodView.this,"连接失败，请稍等重试",Toast.LENGTH_LONG).show();
                    }
                    else{
                        item.setTitle("停止实时更新");
                    }

                }
                else if(item.getTitle().equals("停止实时更新")){

                    Log.d("实时更新关闭","实时更新关闭");
                    //connection.onServiceDisconnected(getComponentName());
                    unbindService(connection);

                    //thread.interrupt();
                    if(stopService(intent)){
                        Toast.makeText(FoodView.this,"解除失败，请稍等重试",Toast.LENGTH_LONG).show();
                    }
                    else{
                        item.setTitle("启动实时更新");
                    }
                }
                break;
```

4.在SCOS的es.source.code.service中新建类UpdateService

```
//UpdateService
/**
 * An {@link IntentService} subclass for handling asynchronous task requests in
 * a service on a separate handler thread.
 * <p>
 * TODO: Customize class - update intent actions, extra parameters and static
 * helper methods.
 */
public class UpdateService extends IntentService {
    // TODO: Rename actions, choose action names that describe tasks that this
    // IntentService can perform, e.g. ACTION_FETCH_NEW_ITEMS
    private static final String ACTION_FOO = "es.source.code.service.action.FOO";
    private static final String ACTION_BAZ = "es.source.code.service.action.BAZ";

    // TODO: Rename parameters
    private static final String EXTRA_PARAM1 = "es.source.code.service.extra.PARAM1";
    private static final String EXTRA_PARAM2 = "es.source.code.service.extra.PARAM2";

    private NotificationManager manager;
    private int Notification_ID;
    public UpdateService() {
        super("UpdateService");
        Log.d(TAG,"UpdateService");
    }

    @Override
    public void onCreate() {
        Log.d(TAG,"onCreateUpdateService");
        //Toast.makeText(this,"收到开机广播",Toast.LENGTH_LONG).show();
        super.onCreate();

    }

    /**
     * Starts this service to perform action Foo with the given parameters. If
     * the service is already performing a task this action will be queued.
     *
     * @see IntentService
     */
    // TODO: Customize helper method
    public static void startActionFoo(Context context, String param1, String param2) {
        Intent intent = new Intent(context, UpdateService.class);
        intent.setAction(ACTION_FOO);
        intent.putExtra(EXTRA_PARAM1, param1);
        intent.putExtra(EXTRA_PARAM2, param2);
        context.startService(intent);
    }

    /**
     * Starts this service to perform action Baz with the given parameters. If
     * the service is already performing a task this action will be queued.
     *
     * @see IntentService
     */
    // TODO: Customize helper method
    public static void startActionBaz(Context context, String param1, String param2) {
        Intent intent = new Intent(context, UpdateService.class);
        intent.setAction(ACTION_BAZ);
        intent.putExtra(EXTRA_PARAM1, param1);
        intent.putExtra(EXTRA_PARAM2, param2);
        context.startService(intent);
    }

    @TargetApi(Build.VERSION_CODES.O)
    @Override
    protected void onHandleIntent(Intent intent) {
        Log.d(TAG, "onHandleIntent");
        NotificationChannel channel = new NotificationChannel("channel_1","123",NotificationManager.IMPORTANCE_LOW);
        manager=(NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
        manager.createNotificationChannel(channel);
        Notification.Builder builder = new Notification.Builder(this,"channel_1");
        builder.setAutoCancel(true);
        builder.setSmallIcon(R.drawable.food_cold_dbjlbc);//设置图标
        builder.setTicker("!新品上架!");//手机状态栏的提示
        builder.setContentTitle("新品上架");//设置标题
        builder.setContentText("东北家拌凉菜,10,冷菜");//设置通知内容
        builder.setWhen(System.currentTimeMillis());//设置通知时间

        Intent intent_to = new Intent(this, MainScreen.class);
        /*Bundle bundle = new Bundle();
        bundle.putString("name", "东北家拌凉菜");
        bundle.putInt("id",R.drawable.food_cold_dbjlbc);
        intent_to.putExtras(bundle);*/

        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent_to, 0);
        builder.setContentIntent(pendingIntent);//点击后的意图
        builder.setDefaults(Notification.DEFAULT_LIGHTS);//设置指示灯
        builder.setDefaults(Notification.DEFAULT_SOUND);//设置提示声音
        builder.setDefaults(Notification.DEFAULT_VIBRATE);//设置震动
        Notification notification = builder.build();//4.1以上，以下要用getNotification()
        startForeground(Notification_ID,notification);//开机广播，设置为前台服务
        manager.notify(Notification_ID, notification);

    }

    /**
```

```
     * Handle action Foo in the provided background thread with the provided
     * parameters.
     */
    private void handleActionFoo(String param1, String param2) {
        // TODO: Handle action Foo
        throw new UnsupportedOperationException("Not yet implemented");
    }

    /**
     * Handle action Baz in the provided background thread with the provided
     * parameters.
     */
    private void handleActionBaz(String param1, String param2) {
        // TODO: Handle action Baz
        throw new UnsupportedOperationException("Not yet implemented");
    }

    private void showNotification() {
        // TODO Auto-generated method stub
    }
}
```

5.在 es.source.code.br 子包中新建 类DeviceStartedListener

```
//DeviceStartedListener
public class DeviceStartedListener extends BroadcastReceiver {
    private static final String TAG = "DeviceStartedListener";
    @Override
    public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the BroadcastReceiver is receiving
        // an Intent broadcast.
        //throw new UnsupportedOperationException("Not yet implemented");
        Log.d(TAG, "onReceive");
        Log.d(TAG, "kaiji");
        if (intent.getAction().equals("android.intent.action.BOOT_COMPLETED")) {
            Intent serviceIntent = new Intent(context, UpdateService.class);
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
                context.startForegroundService(serviceIntent);
            }
        }
        else if (intent.getAction().equals("scos.intent.action.CLOSE_NOTIFICATION")) {
            NotificationManager notifyManager = (NotificationManager) context.getSystemService(Context
                        .NOTIFICATION_SERVICE);
            notifyManager.cancel(intent.getIntExtra("notification_id", 0));
        }
    }
}


```

```
<!--AndroidManifest信息-->
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="xq.scos">

    <permission
        android:name="scos.permission.ACCESSSCOS"
        android:protectionLevel="dangerous" />

    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.READ_PROFILE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.READ_SMS" />
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.CALL_PHONE" />
```

```xml

    <application

        android:allowBackup="true"
        android:icon="@mipmap/icon"
        android:label="@string/app_name"
        android:roundIcon="@drawable/free04"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning"
        >

        <activity android:name="es.source.code.activity.SCOSEntry">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="es.source.code.activity.MainScreen"
            android:label="@string/MainScreen"
            android:permission="scos.permission.ACCESSSCOS">
            <intent-filter>
                <action android:name="android.intent.action.SCOSMAIN" />

                <category android:name="android.intent.category.DEFAULT" />
                <category android:name="android.intent.category.SCOSLAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="es.source.code.activity.Help"
            android:label="@string/help" />
        <activity
            android:name="es.source.code.activity.LoginOrRegister"
            android:label="@string/log" />
        <activity
            android:name="es.source.code.activity.FoodOrderView"
            android:label="@string/title_food_order_view" />
        <activity
            android:name="es.source.code.activity.FoodDetailed"
            android:label="@string/title_activity_food_detial_view"
            android:parentActivityName="es.source.code.activity.FoodView" />
        <activity
            android:name="es.source.code.activity.FoodView"
            android:label="@string/title_food_view"
            android:parentActivityName="es.source.code.activity.MainScreen" />

        <receiver
            android:name="es.source.code.br.DeviceStartedListener"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED"/>
                <action android:name="scos.intent.action.CLOSE_NOTIFICATION"/>
            </intent-filter>
        </receiver>

        <service
            android:name="es.source.code.service.UpdateService"
            android:exported="false" />
        <service
            android:name="es.source.code.service.ServerObserverService"
            android:enabled="true"
            android:exported="true">

        </service>
    </application>

</manifest>
```

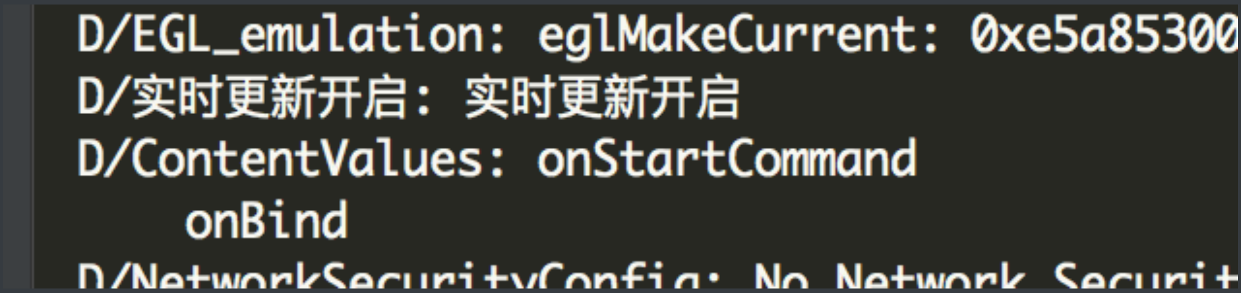
4.结论

本次练习：

1. 熟悉了service的用法。
2. 了解了BroadcastReceiver的具体用法，完成了广播接收功能。
3. 通过使用NotificationManager，完成了推送通知的任务，并实现了点击跳转到Activity。

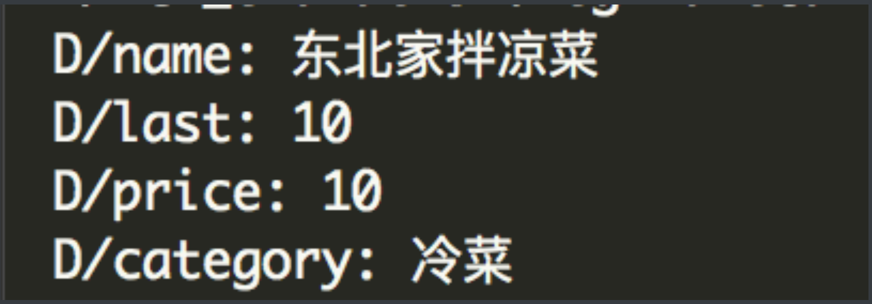
4. 使用EventBus项目，实现了事件总线的推送和订阅。

1.效果图



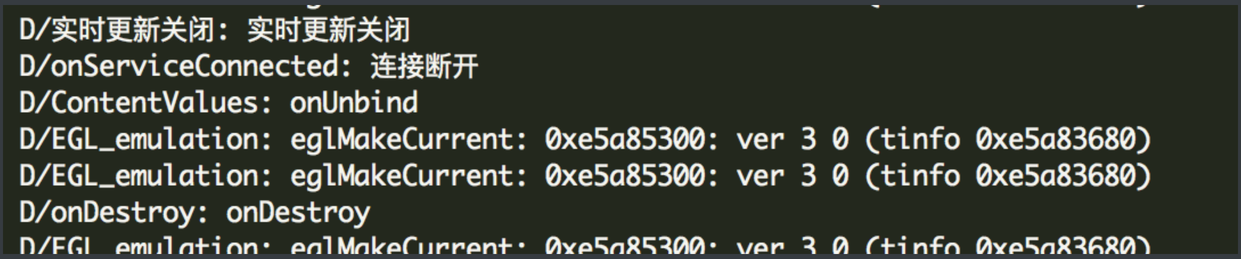
```
D/EGL_emulation: eglMakeCurrent: 0xe5a85300
D/实时更新开启: 实时更新开启
D/ContentValues: onStartCommand
    onBind
D/NetworkSecurityConfig: No Network Securit
```

实时更新服务开启



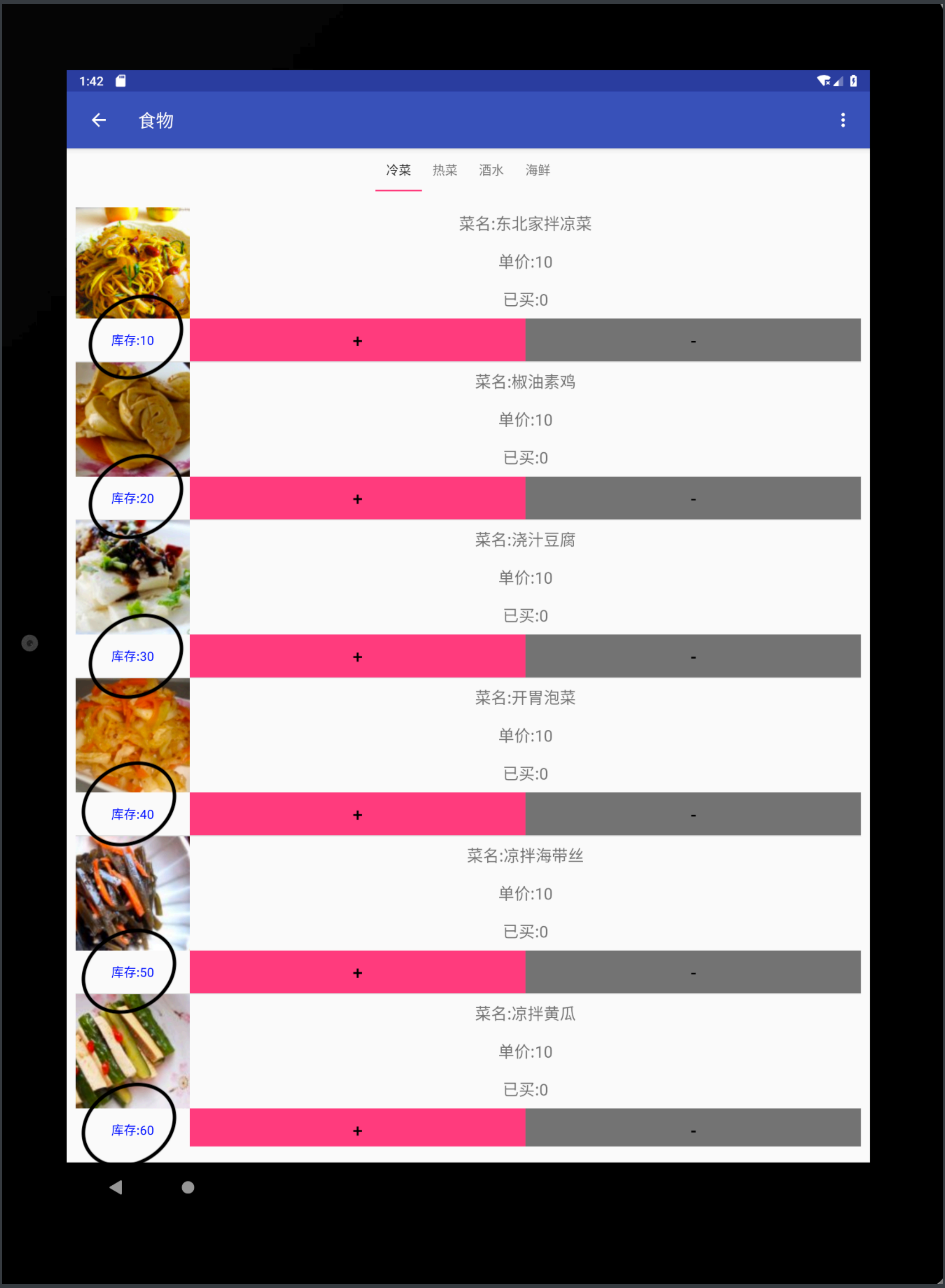
```
D/name: 东北家拌凉菜
D/last: 10
D/price: 10
D/category: 冷菜
```

接收到消息



```
D/实时更新关闭: 实时更新关闭
D/onServiceConnected: 连接断开
D/ContentValues: onUnbind
D/EGL_emulation: eglMakeCurrent: 0xe5a85300: ver 3 0 (tinfo 0xe5a83680)
D/EGL_emulation: eglMakeCurrent: 0xe5a85300: ver 3 0 (tinfo 0xe5a83680)
D/onDestroy: onDestroy
D/EGL_emulation: eglMakeCurrent: 0xe5a85300: ver 3 0 (tinfo 0xe5a83680)
```

实时更新关闭



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

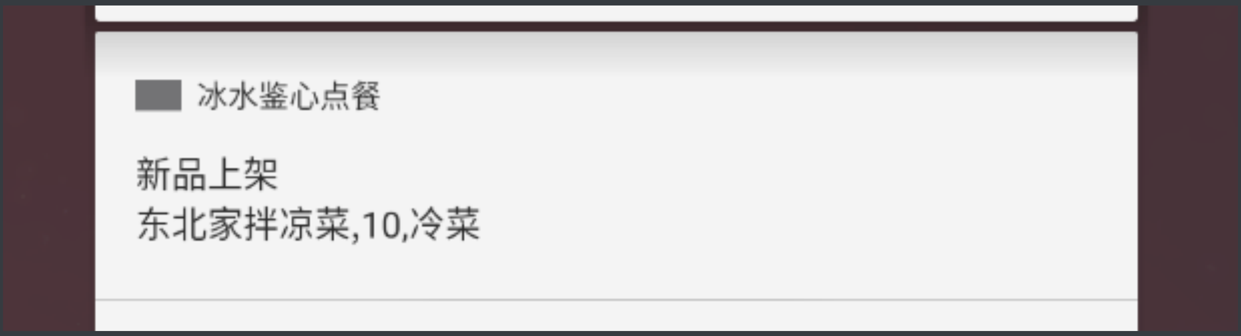
菜名:凉拌黄瓜

单价:10

已买:0

库存:60

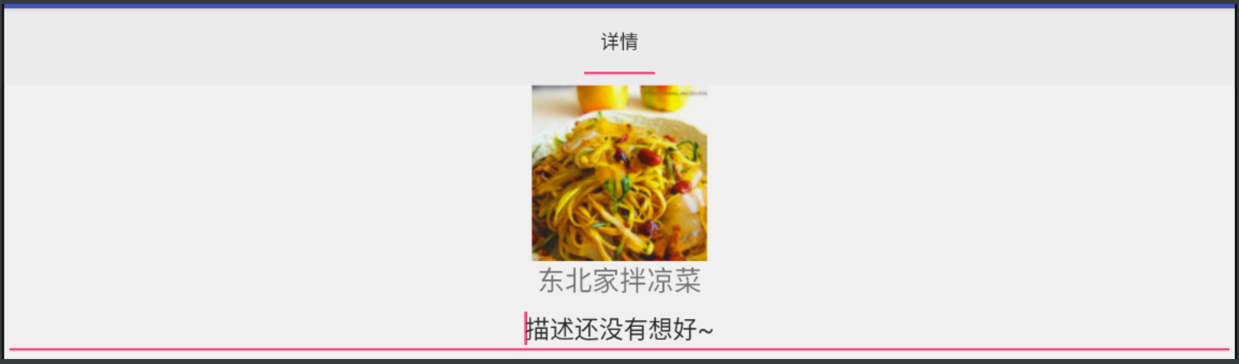
更新时库存被修改



冰水鉴心点餐

新品上架
东北家拌凉菜,10,冷菜

开机广播，显示通知



东北家拌凉菜

描述还没有想好~

点击进入详情

5.参考

参考书籍

```
【1】《Head First Android开发》
【2】《Android第一行代码》
【3】《Head First Java》
【4】《Java编程思想》
【5】《Java核心技术》
【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