work hard work smart

专注于Metro style app(C#), WPF,Asp.net。 不断总结,举一反三。

博客园:: 首页:: 新随笔:: 联系:: 订阅 XML:: 管理

posts - 494, comments - 120, trackbacks - 0, articles - 0

2017年10月 Ξ 四 五 六 25 26 27 28 29 3 4 5 10 11 12 13 19 20 21 <u>16</u> 17 <u>18</u> 23 24 25 28 26 27 30 31 1 2 3

金 公告

Work Hard, Work Smart, Work Happy!

熟悉Metro Style App(C#)、 ASP.NET、WPF 染指Java, Jsp, Android, SQL Server, Oracle, JavaScript

Server、Oracle、JavaScript、 HTML5、CSS3、WCF等技术

昵称: work hard work smart

园龄: 7年1个月 粉丝: 159 关注: 21 +加关注

港 搜索 找找看 谷歌搜索

🛅 常用链接	
我的随笔	
我的评论	
我的参与	
最新评论	
我的标签	

Android中Service的使用详解和注意点(LocalService)

Posted on 2013-09-02 13:27 work hard work smart 阅读(47253) 评论(8) 编辑 收藏

Android中Service的使用详解和注意点(LocalService)

原文地址

开始, 先稍稍讲一点android中Service的概念和用途吧~

Service分为本地服务(LocalService)和远程服务(RemoteService):

1、本地服务依附在主进程上而不是独立的进程,这样在一定程度上节约了资源,另外Local服务因为是在同一进程因此不需要IPC,

也不需要AIDL。相应bindService会方便很多。主进程被Kill后,服务便会终止。

2、远程服务为独立的进程,对应进程名格式为所在包名加上你指定的android:process字符串。由于是独立的进程,因此在Activity所在进程被Kill的时候,该服务依然在运行,

不受其他进程影响,有利于为多个进程提供服务具有较高的灵活性。该服务是独立的进程,会占用一定资源,并且使用AIDL进行IPC稍微麻烦一点。

按使用方式可以分为以下三种:

- 1、startService 启动的服务:主要用于启动一个服务执行后台任务,不进行通信。停止服务使用stopService;
- 2、bindService 启动的服务:该方法启动的服务可以进行通信。停止服务使用unbindService;
- 3、startService 同时也 bindService 启动的服务:停止服务应同时使用stepService与unbindService

Service 与 Thread 的区别

很多时候,你可能会问,为什么要用 Service,而不用 Thread 呢,因为用 Thread 是很方便的,比起 Service 也方便多了,下面我详细的来解释一下。

- 1). Thread: Thread 是程序执行的最小单元,它是分配CPU的基本单位。可以用 Thread来执行一些异步的操作。
- 2). Service:Service 是android的一种机制,当它运行的时候如果是Local Service,那么对应的 Service 是运行在主进程的 main 线程上的。如:onCreate,onStart 这 些函数在被系统调用的时候都是在主进程的 main 线程上运行的。如果是Remote Service,那么对应的 Service 则是运行在独立进程的 main 线程上。<mark>因此请不要把</mark>

Service 理解成线程,它跟线程半毛钱的关系都没有!

更多辩接

1 我的标签

Android (2)

Android 基础 (2)

c#基础知识----委托和事件(2)

Design Model (1)

STL(1)

Word 2007 基本操作 (1)

数据结构和算法(1)

ATL (1)

c#(1)

随笔分类(534)

.NET互操作性(8)

A.设计模式(1)

B.WPF(32)

B.WPF 控件开发(10)

B.WPF 类库整理

C.Blend(WPF/Silverlight)(1)

D.C# (36)

D.Effective C#(1)

D.More Effective C#

E.ASP.NET开发

E.微信公众号开发(ASP.NET)(2)

EPub(5)

F.VS 工具使用技巧(11)

G.Experience Sharing(8)

H.ATL

I.COM(1)

J.CPP(31)

J.CPP 高级(3)

K.WCF(1)

L.WF

M.数据结构(1)

N.work hard work smart 随笔目录(1)

O. HTML5(7)

O.CSS3(3)

O.JQuery(8)

P. Win 8(34)

Q.JavaScript(4)

既然这样,那么我们为什么要用 Service 呢?其实这跟 android 的系统机制有关,我们先拿 Thread 来说。Thread 的运行是独立于 Activity 的,也就是说当一个 Activity 被 finish 之后,如果你没有主动停止 Thread 或者 Thread 里的 run 方法没有执行完毕的话,Thread 也会一直执行。因此这里会出现一个问题:当 Activity 被 finish 之 后,你不再持有该 Thread 的引用。另一方面,你没有办法在不同的 Activity 中对同一 Thread 进行控制。

举个例子:如果你的 Thread 需要不停地隔一段时间就要连接服务器做某种同步的话,该 Thread 需要在 Activity 没有start的时候也在运行。这个时候当你 start 一个 Activity 就没有办法在该 Activity 里面控制之前创建的 Thread。因此你便需要创建并启动一个 Service ,在 Service 里面创建、运行并控制该 Thread,这样便解决了该问 题(因为任何 Activity 都可以控制同一 Service, 而系统也只会创建一个对应 Service 的实例)。

因此你可以把 Service 想象成一种消息服务,而你可以在任何有 Context 的地方调用 Context.startService、Context.stopService、Context.bindService, Context.unbindService,来控制它,你也可以在 Service 里注册 BroadcastReceiver,在其他地方通过发送 broadcast 来控制它,当然这些都是 Thread 做不到的。

Service的生命周期

onStart onCreate onDestroy onBind

- 1). 被启动的服务的生命周期:如果一个Service被某个Activity 调用 Context.startService 方法启动,那么不管是否有Activity使用bindService绑定或unbindService解除绑 定到该Service,该Service都在后台运行。如果一个Service被startService 方法多次启动,那么onCreate方法只会调用一次,onStart将会被调用多次(对应调用 startService的次数),并且系统只会创建Service的一个实例(因此你应该知道只需要一次stopService调用)。该Service将会一直在后台运行,而不管对应程序的 Activity是否在运行,直到被调用stopService,或自身的stopSelf方法。当然如果系统资源不足, android系统也可能结束服务。
- 2). 被绑定的服务的生命周期:如果一个Service被某个Activity 调用 Context.bindService 方法绑定启动,不管调用 bindService 调用几次,onCreate方法都只会调用一 次,同时onStart方法始终不会被调用。当连接建立之后,Service将会一直运行,除非调用Context.unbindService 断开连接或者之前调用bindService 的 Context 不存在 了(如Activity被finish的时候),系统将会自动停止Service,对应onDestroy将被调用。
- 3)、被启动又被绑定的服务的生命周期:如果一个Service又被启动又被绑定,则该Service将会一直在后台运行。并且不管如何调用,onCreate始终只会调用一次,对应 startService调用多少次,Service的onStart便会调用多少次。调用unbindService将不会停止Service,而必须调用 stopService 或 Service的 stopSelf 来停止服务。
- 4), 当服务被停止时清除服务: 当一个Service被终止(1、调用stopService: 2、调用stopSelf: 3、不再有绑定的连接(没有被启动))时, onDestroy方法将会被调用, 在这里你应当做一些清除工作,如停止在Service中创建并运行的线程。

特别注意:

- 1、你应当知道在调用 bindService 绑定到Service的时候,你就应当保证在某处调用 unbindService 解除绑定(尽管 Activity 被 finish 的时候绑定会自 动解除,并且Service会自动停止):
- 2、你应当注意 使用 startService 启动服务之后,一定要使用 stopService停止服务,不管你是否使用bindService;
- 3、同时使用 startService 与 bindService 要注意到,Service 的终止,需要unbindService与stopService同时调用,才能终止 Service,不管 startService 与 bindService 的调用顺序,如果先调用 unbindService 此时服务不会自动终止,再调用 stopService 之后服务才会停止,如果先调用 stopService 此时服务也不会终 止,而再调用 unbindService 或者 之前调用 bindService 的 Context 不存在了(如Activity 被 finish 的时候)之后服务才会自动停止;
- 4、当在旋转手机屏幕的时候,当手机屏幕在"横""竖"变换时,此时如果你的 Activity 如果会自动旋转的话,旋转其实是 Activity 的重新创建,因此旋转之前的使用 bindService 建立的连接便会断开(Context 不存在了),对应服务的生命周期与上述相同。

```
O.SOLite(2)
R.Ling(1)
S. Mybatics
S.Android(85)
S.Android 服务端开发(9)
S.Android 工具使用(41)
S.Android 开源项目使用(8)
S.Hibernate
S.Java(30)
S.Java SSH (Spring Structs
Hibernate ) (10)
S.Jsp(12)
S.Linux(19)
S.Spring(2)
S.Struts
T. PHP(7)
T.MySQL(2)
U. Angular(26)
U. IONIC(4)
U. PhoneGap(17)
U. React(1)
U. React Native(3)
其它(38)
生活(8)
```

```
随笔档案(494)
2017年10月 (24)
2017年9月 (2)
2017年8月 (18)
2017年7月 (11)
2017年6月 (14)
2017年5月 (8)
2017年4月 (3)
2017年3月 (9)
2017年2月 (3)
2016年12月 (3)
2016年10月 (4)
2016年9月 (7)
2016年8月 (3)
2016年7月 (2)
2016年6月 (4)
```

```
5、在 sdk 2.0 及其以后的版本中,对应的 onStart 已经被否决变为了 onStartCommand,不过之前的 onStart 任然有效。这意味着,如果你开发的应用程序用的 sdk 为 2.0 及其以后的版本,那么你应当使用 onStartCommand 而不是 onStart。
```

下面开始上一个很简单的代码哈~里头的注释也要注意哦,有在上面没有讲到的会在注释里提到哇(尤其适用Bind方法的时候的数据传输哇)~ 首先,因为要再Manifest文件里对服务进行注册,所以就先来Manifest的代码吧~

```
[html] view plain copy print ?
 1.
      <?xmlversion="1.0"encoding="utf-8"?>
 2.
      <manifestxmlns:android="http://schemas.android.com/apk/res/android"</pre>
 3.
           package="com.test.localservice"android:versionCode="1"
           android:versionName="1.0">
 4.
           <uses-sdkandroid:minSdkVersion="8"/>
 5.
 6.
           <applicationandroid:icon="@drawable/icon"android:label="@string/app_name">
 7.
 8.
                <activityandroid:name=".LocalServiceTestActivity"</pre>
                    android:label="@string/app name">
 9.
                    <intent-filter>
10.
                         <actionandroid:name="android.intent.action.MAIN"/>
11.
12.
                         <categoryandroid:name="android.intent.category.LAUNCHER"/>
                    </intent-filter>
13.
14.
                </activity>
                <serviceandroid:name=".MyService">
15.
16.
                    <intent-filter>
17.
                         <actionandroid:name="com.test.SERVICE_TEST"/>
18.
                         <categoryandroid:name="android.intent.category.default"/>
                    </intent-filter>
19.
20.
               </service>
21.
           </application>
      </manifest>
22.
 <?xml version="1.0" encoding="utf-8"?>
 <manifest xmlns:android="http://schemas.android.com/apk/res/android"</pre>
     package="com.test.localservice" android:versionCode="1"
     android:versionName="1.0">
     <uses-sdk android:minSdkVersion="8" />
     <application android:icon="@drawable/icon" android:label="@string/app_name">
         <activity android:name=".LocalServiceTestActivity"
             android:label="@string/app_name">
             <intent-filter>
                 <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
             </intent-filter>
```

```
2016年4月 (1)
                                             </activity>
                                             <service android:name=".MyService">
2016年3月 (8)
                                                 <intent-filter>
2016年2月 (6)
                                                     <action android:name="com.test.SERVICE_TEST" />
2016年1月 (7)
                                                     <category android:name="android.intent.category.default" />
2015年11月 (3)
2015年10月 (2)
                                             </service>
                                          </application>
2015年9月 (2)
                                      </manifest>
2015年8月 (2)
                                   然后然后,是服务实现类~
2015年7月 (2)
2015年6月 (3)
2014年11月 (2)
2014年10月 (1)
2014年7月 (4)
                                     [java] view plain copy print ?
2014年6月 (2)
                                      1.
2014年4月 (6)
2014年3月(3)
                                     [java] view plain copy print ?
2014年2月 (2)
                                      1.
                                           package com.test.service;
2014年1月 (2)
                                      2.
2013年12月 (6)
                                      3.
                                           import android.app.Service;
2013年11月 (8)
                                      4.
                                           import android.content.Intent;
                                      5.
                                           import android.os.Binder;
2013年10月(3)
                                      6.
                                           import android.os.IBinder;
2013年9月 (5)
                                      7.
                                           import android.util.Log;
2013年8月 (8)
                                      8.
2013年7月 (1)
                                      9.
                                           publicclass MyService extends Service {
2013年6月 (8)
                                     10.
2013年5月 (10)
                                     11.
                                               publicclass LocalBinder extends Binder {
                                                    String stringToSend = "I'm the test String";
2013年4月 (8)
                                     12.
                                     13.
                                                    MyService getService() {
2013年3月 (33)
                                                        Log.i("TAG", "getService ---> " + MyService.this);
                                     14.
2013年1月 (7)
                                     15.
                                                        return MyService.this;
2012年12月 (10)
                                     16.
                                                    }
2012年11月 (2)
                                     17.
                                               }
2012年10月 (2)
                                     18.
2012年9月 (2)
                                     19.
                                               privatefinal IBinder mBinder = new LocalBinder();
                                     20.
2012年8月 (3)
                                     21.
                                               @Override
2012年7月 (6)
                                     22.
                                               public IBinder onBind(Intent intent) {
2012年6月 (7)
                                     23.
                                                    // TODO Auto-generated method stub
2012年3月 (18)
                                     24.
                                                    Log.i("TAG", "onBind~~~~");
2012年2月 (17)
                                           //
                                                    IBinder myIBinder = null;
                                     25.
2012年1月 (13)
                                           //
                                                    if ( null == myIBinder )
                                     26.
                                     27.
                                           //
                                                        myIBinder = new LocalBinder() ;
2011年12月 (26)
```

```
2011年11月 (48)
2011年10月 (13)
2011年9月 (28)
2011年8月 (17)
2011年7月 (7)
2011年5月 (3)
2011年4月 (1)
2011年3月 (1)
```

m www.baidu.com

1 积分与排名

积分 - 360104 排名 - 414

■ 最新评论

1. Re:C# Dictionary用法总结

@huang_lei这个是lz自己写的,你当 然找不到了...

--倾城之叹

2. Re:Spring 部署Tomcat 404 错误解 决方案

按这样弄,但走到这段的第5句时发现 没有maven dependencies:1, Open the project's properties (e.g., rightclick on the proj......

--张张developer

3. Re:Angular 4 辅助路由

慕课网实战课程中的例子

--qiaoer2

4. Re:WPF 一个MVVM的简单例子

View层没有好好写出来。这个Demo是 跑不通的。

--gujunge

5. Re:PhoneGap下Web SQL实践

你好,最近我想用Android来获取新浪云数据,有几个问题想请教,方便加一下你qq吗

--你好吗hello

```
11
28.
             return myIBinder;
                                //也可以像上面几个语句那样重新new一个IBinder
29.
             return mBinder:
             //如果这边不返回一个IBinder的接口实例,那么ServiceConnection中的onServiceConnected就不会被调用
30.
31.
             //那么bind所具有的传递数据的功能也就体现不出来~\(≧▽≦)/~啦啦啦(这个返回值是被作为onServiceConnected中的第二个参数的)
32.
         }
33.
34.
         @Override
         publicvoid onCreate() {
35.
36.
             // TODO Auto-generated method stub
37.
             super.onCreate();
38.
39.
             Log.i("TAG", "onCreate~~~~");
40.
         }
41.
42.
         @Override
43.
         publicvoid onDestroy() {
44.
             // TODO Auto-generated method stub
45.
             super.onDestroy();
             Log.i("TAG", "onDestroy~~~~");
46.
47.
         }
48.
49.
         @Override
50.
         publicvoid onStart(Intent intent, int startId) {
             // TODO Auto-generated method stub
51.
52.
             super.onStart(intent, startId);
             Log.i("TAG", "onStart~~~~");
53.
54.
         }
55.
56.
         @Override
57.
         publicint onStartCommand(Intent intent, int flags, int startId) {
58.
             // TODO Auto-generated method stub
             Log.i("TAG", "onStartCommand~~~~~");
59.
             returnsuper.onStartCommand(intent, flags, startId);
60.
         }
61.
62.
63.
         @Override
64.
         publicboolean onUnbind(Intent intent) {
             // TODO Auto-generated method stub
65.
             Log.i("TAG", "onUnbind~~~~");
66.
67.
             returnsuper.onUnbind(intent);
68.
         }
69. }
package com.test.service;
import android.app.Service;
 import android.content.Intent;
```

□ 阅读排行榜

- 1. C# Dictionary用法总结(130575)
- 2. Android中Service的使用详解和注意 点 (LocalService) (47253)
- 3. JQuery 淡出、 动画、显示/隐藏切 换等效果(41451)
- 4. Eclipse工具使用技巧总结(28269)
- 5. Eclipse 连接MySql数据库总结 (25472)
- 6. .NET Framework 4.5新特性(21094)
- 7. Android 中PopupWindow使用 (20733)
- 8. Android 文件的选择(20469)
- 9. Java 使用execute方法执行Sql语句 (20107)
- 10. WPF 一个MVVM的简单例子 (19736)
- 11. C#操作xml

SelectNodes,SelectSingleNode总是返回NULL与 xPath 介绍(17729)

- 12. android 各国语言对应的缩写 (16949)
- 13. jQuery HTML 操作 改变/添加文本值(15705)
- 14. WPF ListBox(14485)
- 15. wpf 多线程(13648)
- 16. WPF popup控件的使用(12676)
- 17. WPF 绑定各种数据源之 Datatable(12254)
- 18. C# 方法使用汇总 (GetEnumerator 用法 , ? ? 用法等) (10891)
- 19. Android 高手进阶之自定义View, 自定义属性(带进度的圆形进度条) (10633)
- 20. 如何使用Android中的 Sample(10444)
- 21. C# 对XML基本操作总结(10359)
- 22. Perforce使用指南_forP4V(10289)
- 23. ContextMenu的使用(10039)
- 24. 安装Android sdk 4.4(19)出现问题 的解决方案(9815)
- 25. html5 图片旋转(9806)
- 26. WPF绑定各种数据源之元素控件属性(9364)
- 27. Android 使用AIDL调用外部服务 (9018)

```
import android.os.Binder;
import android.os.IBinder;
import android.util.Log;
public class MyService extends Service {
    public class LocalBinder extends Binder {
        String stringToSend = "I'm the test String";
        MyService getService() {
            Log.i("TAG", "getService ---> " + MyService.this);
            return MyService.this;
    private final IBinder mBinder = new LocalBinder();
    @Override
    public IBinder onBind(Intent intent) {
        // TODO Auto-generated method stub
        Log.i("TAG", "onBind~~~~~~");
//
        IBinder myIBinder = null;
//
        if ( null == myIBinder )
//
            myIBinder = new LocalBinder();
//
        return myIBinder;
        return mBinder;
                             //也可以像上面几个语句那样重新new一个IBinder
        //如果这边不返回一个IBinder的接口实例,那么ServiceConnection中的onServiceConnected就不会被调用
        //那么bind所具有的传递数据的功能也就体现不出来~\(≧▽≦)/~啦啦啦(这个返回值是被作为onServiceConnected中的第二个参数的)
    @Override
    public void onCreate() {
        // TODO Auto-generated method stub
        super.onCreate():
        Log.i("TAG", "onCreate~~~~~");
    @Override
    public void onDestroy() {
        // TODO Auto-generated method stub
        super.onDestroy();
        Log.i("TAG", "onDestroy~~~~~");
    @Override
    public void onStart(Intent intent, int startId) {
        // TODO Auto-generated method stub
        super.onStart(intent, startId);
        Log.i("TAG", "onStart~~~~");
```

```
28. C# Serializable对象序列化的作用 (8731)
```

- 29. Java 使用executeUpdate向数据库中创建表格(8608)
- 30. C++ 使用API写Windows程序 (8398)
- 31. WPF 获得DataTemplate中的控件 (7857)
- 32. C++
- AFX_MANAGE_STATE(AfxGetStaticMo 的作用(7732)
- 33. css <meta name="viewport" content="xx">(7452)
- 34. java实现文件单词频率统计(7216)
- 35. 创建Android的Hello World应用程序(7077)
- 36. WPF Button 透明效果(7024)
- 37. Android 隐藏Fragment(6959)
- 38. word 2007中在页眉中插入或这删除下划线(6708)
- 39. WPF 控件使用之 ComboBox(6393)
- 40. WPF 附加属性(6165)

🛅 评论排行榜

- 1. WPF 一个MVVM的简单例子(12)
- 2. C# Dictionary用法总结(8)
- 3. Android中Service的使用详解和注意
- 点 (LocalService)(8)
- 4. html5 图片旋转(6)
- 5. C#操作xml
- SelectNodes,SelectSingleNode总是返回NULL 与 xPath 介绍(5)

🛅 推荐排行榜

- 1. C# Dictionary用法总结(15)
- 2. Android中Service的使用详解和注意 点 (LocalService) (5)
- 3. wpf 多线程(5)
- 4. WPF 附加属性(4)
- 5. WPF ListBox(4)
- 6. Metro Style App开发快速入门 之文件访问操作示例(4)

```
@Override
      public int onStartCommand(Intent intent, int flags, int startId) {
         // TODO Auto-generated method stub
         Log.i("TAG", "onStartCommand~~~~~");
         return super.onStartCommand(intent, flags, startId);
      @Override
      public boolean onUnbind(Intent intent) {
         // TODO Auto-generated method stub
         Log.i("TAG", "onUnbind~~~~~~");
         return super.onUnbind(intent);
再来,就是我们的Activity的测试类啦~
 [java] view plain copy print ?
   1.
        package com.test.service;
   2.
   3.
        import android.app.Activity;
        import android.content.ComponentName;
   4.
   5.
        import android.content.Context;
   6.
        import android.content.Intent;
   7.
        import android.content.ServiceConnection;
        import android.media.MediaPlayer;
   9.
        import android.os.Bundle;
        import android.os.IBinder;
 10.
 11.
        import android.util.Log;
 12.
        import android.view.View;
 13.
        import android.view.View.OnClickListener;
 14.
        import android.widget.Button;
 15.
        publicclass ServiceTestActivity extends Activity {
 16.
 17.
            private Button startButton, bindButton;
 18.
            private Button stopButton, unbindButton;
  19.
            private ServiceConnection sc;
            private MediaPlayer mediaPlayer = null;
 20.
 21.
            private MyService myService;// 类似于MediaPlayer mPlayer = new
  22.
                                         // MediaPlayer();只不过这边的服务是自定义的,不是系统提供好了的
  23.
 24.
            /** Called when the activity is first created. */
```

```
7. C#操作xml
                                  25.
                                           @Override
SelectNodes,SelectSingleNode总是返
                                  26.
                                           publicvoid onCreate(Bundle savedInstanceState) {
回NULL 与 xPath 介绍(3)
                                  27.
                                               super.onCreate(savedInstanceState);
8. 委托和事件(无参数事件和有参数
                                  28.
                                               setContentView(R.layout.main);
事件)(3)
                                  29.
9. Eclipse工具使用技巧总结(3)
                                  30.
                                               startButton = (Button) findViewById(R.id.startbutton id);
                                  31.
                                               stopButton = (Button) findViewById(R.id.stopbutton id);
10. Metro Style App开发快速入门之
XML文件读取,修改,保存等操作(3)
                                  32.
                                               bindButton = (Button) findViewById(R.id.bindbutton_id);
                                               unbindButton = (Button) findViewById(R.id.unbindbutton_id);
11. MVVM设计模式(2)
                                  33.
                                  34.
12. WPF 使用值转换器讲行绑定数据
                                  35.
                                               sc = new ServiceConnection() {
的转换IValueConverter(2)
                                  36.
13. 长时间用电脑该如何保护眼睛(2)
                                  37.
                                                    * 只有在MyService中的onBind方法中返回一个IBinder实例才会在Bind的时候
14. WPF 一个MVVM的简单例子(2)
                                                    * 调用onServiceConnection回调方法
                                  38.
15. ListBox 单击变大动画效果(使用
                                                    * 第二个参数service就是MyService中onBind方法return的那个IBinder实例,可以利用这个来传递数据
                                  39.
模板、样式、绑定数据源等)(2)
                                                    */
                                  40.
16. WPF绑定各种数据源之xml数据源
                                  41.
                                                   @Override
                                  42.
                                                   publicvoid onServiceConnected(ComponentName name, IBinder service) {
17. WPF 绑定各种数据源之
                                  43.
                                                       // TODO Auto-generated method stub
Datatable(2)
                                  44.
                                                       myService = ((MyService.LocalBinder) service).getService();
18. 在.Net中使用COM组件(2)
                                                       String recStr = ((MyService.LocalBinder) service).stringToSend;
                                  45.
19. WPF 制作模板页示例(2)
                                                       //利用IBinder对象传递过来的字符串数据(其他数据也可以啦,哪怕是一个对象也OK~~)
                                  46.
20. Metro Style App开发快速入门 之基
                                  47.
                                                       Log.i("TAG", "The String is : " + recStr);
本控件使用总结(2)
                                  48.
                                                       Log.i("TAG", "onServiceConnected : myService ---> " + myService);
21. Metro Style App开发快速入门 之资
                                  49.
                                                   }
源操作(2)
                                  50.
22. Android 文件的选择(2)
                                  51.
                                                   @Override
23. .NET Framework 4.5新特性(2)
                                  52.
                                                   publicvoid onServiceDisconnected(ComponentName name) {
24. VC++ MFC 过时了吗(转)(2)
                                  53.
                                                       /* SDK上是这么说的:
                                                        * This is called when the connection with the service has been unexpectedly disconnected
25. css <meta name="viewport"
                                  54.
content="xx">(2)
                                  55.
                                                        * that is, its process crashed. Because it is running in our same process, we should never see this happen.
26. SQLite快速入门二--表、视图的创
                                                        * 所以说,只有在service因异常而断开连接的时候,这个方法才会用到*/
                                  56.
建、修改、删除操作(2)
                                  57.
                                                       // TODO Auto-generated method stub
27. SQLite学习快速入门-- 基础介绍(1)
                                  58.
                                                       sc = null;
28. 八招防电脑辐射 你学会了吗?(1)
                                  59.
                                                       Log.i("TAG", "onServiceDisconnected : ServiceConnection --->"
                                  60.
                                                              + sc);
29. Win 8 快捷键总结(1)
                                  61.
                                                   }
30. 给JavaScript新手的24条实用建议
                                  62.
(1)
                                  63.
                                               };
31. 程序员的5种类型(1)
                                               startButton.setOnClickListener(new OnClickListener() {
                                  64.
32. 为什么程序员都是夜猫子?(1)
                                  65.
33. JQuery 文本的隐藏功能(1)
                                  66.
                                                   @Override
34. html5 图片旋转(1)
                                  67.
                                                   publicvoid onClick(View v) {
35. 使用 EPUB 制作数字图书 基于
                                  68.
                                                       // TODO Auto-generated method stub
XML 的开放式 eBook 格式(1)
                                  69.
                                                       Intent intent = new Intent(ServiceTestActivity.this,
36. C# XML操作总结2 包括读取、插
                                  70.
                                                              MyService.class);
入、修改、删除(1)
```

```
37. Epub基础知识介绍(1)
38. C#弱引用(1)
39. Win 8中WPF listview与listBox的
Drag、Drop操作(1)
40. 安装win7、win8双系统简单方法
(1)
```

```
71.
                       startService(intent);
72.
                       Log.i("TAG", "Start button clicked");
                  }
73.
74.
              });
75.
 76.
               stopButton.setOnClickListener(new OnClickListener() {
77.
78.
                   @Override
                   publicvoid onClick(View v) {
79.
80.
                       // TODO Auto-generated method stub
81.
82.
83.
                        * Intent intent = new
                        * Intent(LocalServiceTestActivity.this, MyService.class);
84.
                        * stopService(intent); 这种方法也是可以的哈~
85.
                        */
86.
87.
                       Intent intent = new Intent();
88.
89.
                       intent.setAction("com.test.SERVICE_TEST");
90.
                       stopService(intent);
                       Log.i("TAG", "Stop Button clicked");
91.
92.
                  }
93.
              });
94.
95.
               bindButton.setOnClickListener(new OnClickListener() {
96.
97.
                   @Override
98.
                   publicvoid onClick(View v) {
99.
                       // TODO Auto-generated method stub
      //
100.
                       Intent intent = new Intent(LocalServiceTestActivity.this,
      //
101.
                               MyService.class);//这样也可以的
102.
                       Intent intent = new Intent();
103.
                       intent.setAction("com.test.SERVICE_TEST");
                       bindService(intent, sc, Context.BIND_AUTO_CREATE);//bind多次也只会调用一次onBind方法
104.
                       Log.i("TAG", "Bind button clicked");
105.
106.
                  }
107.
              });
108.
               unbindButton.setOnClickListener(new OnClickListener() {
109.
110.
                   @Override
111.
112.
                   publicvoid onClick(View v) {
113.
                       // TODO Auto-generated method stub
114.
                       unbindService(sc);
115.
                       // 这边如果重复unBind会报错,提示该服务没有注册的错误—IllegalArgumentException:
116.
                       // Service not registered: null
```

Android中Service的使用详解和注意点(LocalService)-work hard work smart - 博客园

```
117.
                                                                                                                                           // 所以一般会设置一个flag去看这个service
                                                                                                                                          // bind后有没有被unBind过,没有unBind过才能调用unBind方法(这边我就不设置了哈~\(≧▽≦)/~啦啦啦)
 118.
                                                                                                                                           Log.i("TAG", "Unbind Button clicked");
119.
 120.
                                                                                                              }
121.
                                                                                      });
 122.
 123.
                                       }<br>
124.
                                        <br>
125.
                                        126.
                                        <hr>
127.
                                        相信开头的介绍和代码里的注释应该对大家理解和使用Service有所帮助哈~不过这边就先只讲LocalService吧,剩下的RemoteService就等下次在说喽~
128.
                                        129.
                                        <span style="white-space: pre;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="white-space: pre;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,rkk,Arial; font-size: 13px;"><span style="line-height: 24px; font-size: 13px; font-size: 13px; font-size: 13px; font-size: 13px; font-size: 13px; font-size: 13px; font-size: 13
                                        style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana, 宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana, red; font-size: 13px;"><span style="line-height: 24px; font-size: 13px;"><span style="line-height: 24px; font-family: 14px; font-size: 13px;"><span style="line-height: 24px; font-family: 14px; font-size: 14px; font-
                                        verdana,宋体,Arial; font-size: 13px;"><span style="white-space: pre;"><span style="line-height: 24px; font-family: verdana,宋
                                        体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,rkk,Arial; font-size: 13px;"><span style="line-height: 12px; font-family: verdana,rkk,Arial; font-size: 12px; font-size: 12p
                                        height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="white-space: pre;"><span style="line-height: 24px;
                                        font-family: verdana,宋体,Arial; font-size: 13px;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size:
                                        13px;"><span style="white-space: pre;"><span style="line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span
                                        style="color: rgb(51, 51, 51); line-height: 24px; font-family: verdana,宋体,Arial; font-size: 13px;"><span style="white-space:
                                        pre;"></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></span></s
```

Android服务总结

Android服务使用

Android中Service的使用详解和注意点(LocalService)

Android 使用AIDL调用外部服务 (RemoteService)

Android 中的 Service 全面总结

作者: Work Hard Work Smart

出处: http://www.cnblogs.com/linlf03/

欢迎任何形式的转载,未经作者同意,请保留此段声明!

分类: S.Android





work hard work smart

大任 - 21 粉丝 - 159

+加关注

0

5

« 上一篇: ContentProvider 使用示例 (转载)

» 下一篇:linux ubuntu系统下,adb不是内部命令(如何才能让adb命令可以使用)

Feedback

#1楼

2014-11-01 18:21 by climberDing

好文章啊

支持(0) 反对(0)

#2楼

2014-12-30 14:52 by sangxb

好文要顶!

支持(0) 反对(0)

#3楼

2015-01-26 16:24 by 黑夜孤灯

下面这个描述不正确吧:

当在旋转手机屏幕的时候,当手机屏幕在"横""竖"变换时,此时如果你的 Activity 如果会自动旋转的话,旋转其实是 Activity 的重新创建,因此旋转之前的使用 bindService 建立的连接便会断开(Context 不存在了),对应服务的生命周期与上述相同。

自动旋转和activity重新创建能关联上这也太不靠谱

支持(0) 反对(2)

#4楼

2015-03-21 15:05 by 二向箔

@ 黑夜孤灯

在大部分情况下,自动旋转的确是会重新创建activity的。

支持(0) 反对(0)

#5楼

2015-04-12 12:54 by 明天+=灿烂

好文要顶

支持(0) 反对(0)

#6楼

2015-09-18 17:00 by mmbbjjcc

内容很详细,学习了,谢谢

支持(0) 反对(0)

#7楼

2015-09-22 14:22 by drummor

当被启动又被绑定的服务的生命周期,如果是直接解除绑定的话Service就直接销毁了,并不需要stop啊

支持(0) 反对(1)

#8楼

2016-06-16 20:33 by 放纵的卡尔

@ drummor

没启动服务的话是直接销毁!但是启动了服务的话,解除绑定只是unbind,并没有执行ondestroy。

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论,请 登录 或 注册, 访问网站首页。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【活动】腾讯云【云+校园】套餐全新升级

【推荐】报表开发有捷径:快速设计轻松集成,数据可视化和交互



最新IT新闻:

- ·人类太多余?且慢,先听AI科学家详解AlphaGo Zero的伟大与局限
- ·比特币价格首破6000美元 其创造者持币价值59亿美元
- ·亚马逊要建第二总部,引发了100+选手参战的城市战争
- ·传永安行将收购小蓝单车,千万"救助款"已到账
- ·Intel发布开源增强学习框架Coach
- » 更多新闻...



最新知识库文章:

- ·实用VPC虚拟私有云设计原则
- ·如何阅读计算机科学类的书
- · Google 及其云智慧
- ·做到这一点,你也可以成为优秀的程序员
- ·写给立志做码农的大学生
- » 更多知识库文章...

Powered by: 博客园

Copyright © work hard work smart