

Google
Developer
Day 2009



节约用电，人人有责

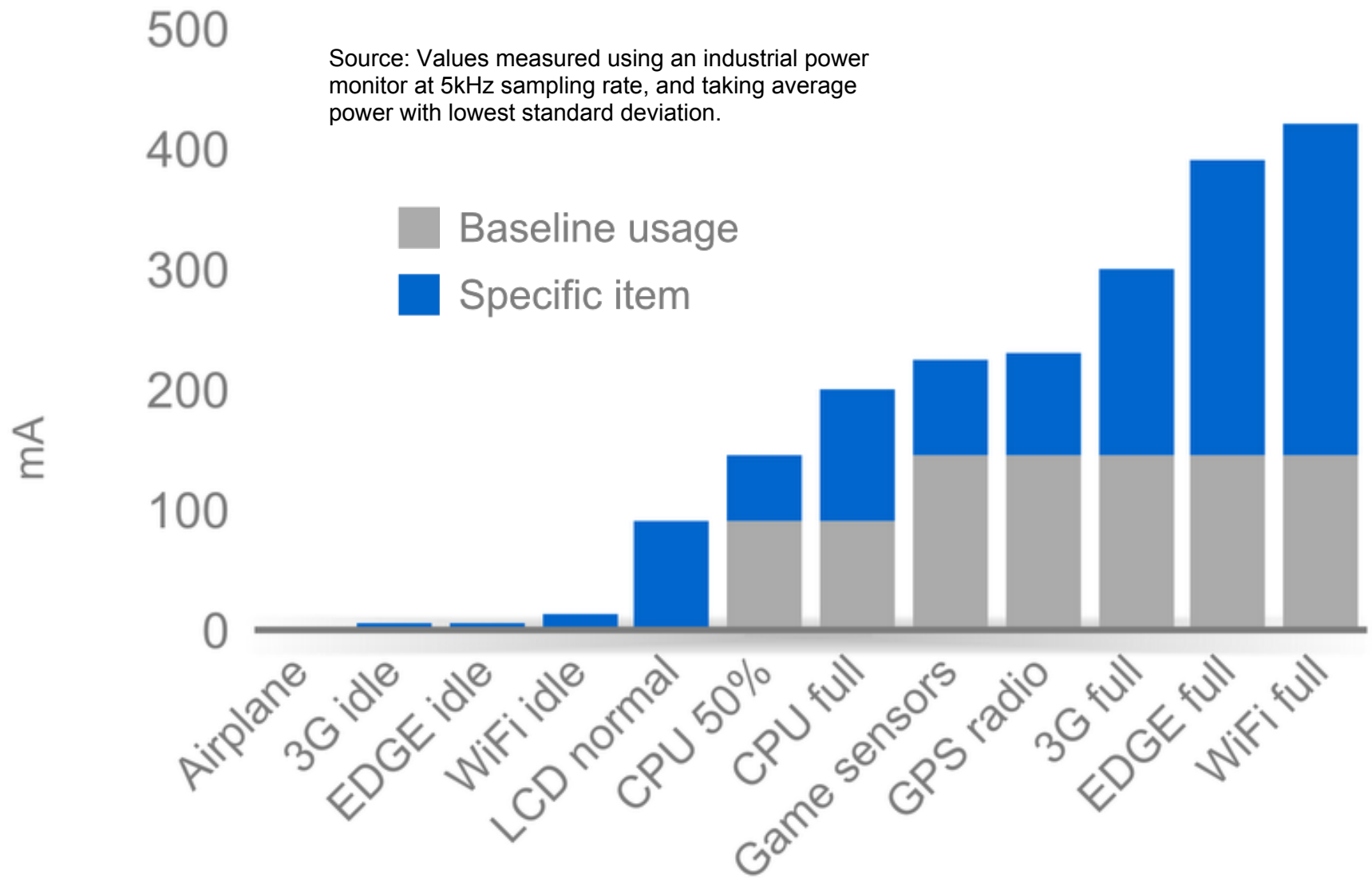
Grace Kloba
2009.06.05

Google
Developer
Day 2009

为什么重要?

- 手机主要通过电池获取能源, 每个手机都有自己的预算
 - 没了就没了
 - 各个程序需要互相合作, 共享资源
 - 电池容量的单位是 "mAh", 工作放电用 "mA"
- HTC Dream: **1150mAh**
- HTC Magic: **1350mAh**
- Samsung i7500: **1500mAh**
- Asus Eee PC: **5800mAh**

消耗在哪里？



消耗在哪里？

- 在现实生活中能用多久？
 - 看 YouTube: $340\text{mA} = 3.4$ 小时
 - 在 3G 网上用浏览器: $225\text{mA} = 5$ 小时
 - 正常使用: $42\text{mA} = 32$ 小时
 - EDGE, 彻底空闲: $5\text{mA} = 9.5$ 天
 - 飞行模式, 彻底空闲: $2\text{mA} = 24$ 天

什么最耗电？

- 后台程序 唤醒休眠的手机, 例
 - 应用程序每10分钟唤醒手机做更新
 - 每次更新持续8秒, 用电 350mA
- 平均每小时耗电翻番
 - $3600 \text{ 秒} * 5\text{mA} = 5\text{mAh}$ 空闲
 - $6 \text{ 次} * 8 \text{ 秒} * 350\text{mA} = 4.7\text{mAh}$ 更新
- 一个就很糟, 几个就不堪设想

什么最耗电？

- 大量数据传输，比如下载 6MB 歌曲
 - EDGE (90kbps): $300\text{mA} * 9.1 \text{ 分钟} = 45\text{mAh}$
 - 纯 3G (300kbps): $210\text{mA} * 2.7 \text{ 分钟} = 9.5\text{mAh}$
 - 无线 (1Mbps): $330\text{mA} * 48 \text{ 秒} = 4.4\text{mAh}$
- 在小区／网络之间切换
 - 当和新的小区连接时无线电信号需要增强
 - `Context.sendBroadcast(Intent)` 涉及整个系统
- 解析文字数据; 没有实时 (JIT) 编译器的 Regex

如何改进？

网络

如何改进？

网络

- 检查网络连接，等待 3G 或者无线

```
NetworkInfo info = mConnectivity.getActiveNetworkInfo();
if (info == null) {
    return false;
}
int netType = info.getType();
int netSubtype = info.getSubtype();

// Only update if WiFi or 3G is connected
if (netType == ConnectivityManager.TYPE_WIFI) {
    return info.isConnected();
} else if (netType == ConnectivityManager.TYPE_MOBILE
    && netSubtype == TelephonyManager.NETWORK_TYPE_UMTS) {
    return info.isConnected();
} else {
    return false;
}
```

如何改进？

网络

- 检查网络连接，等待 3G 或者无线

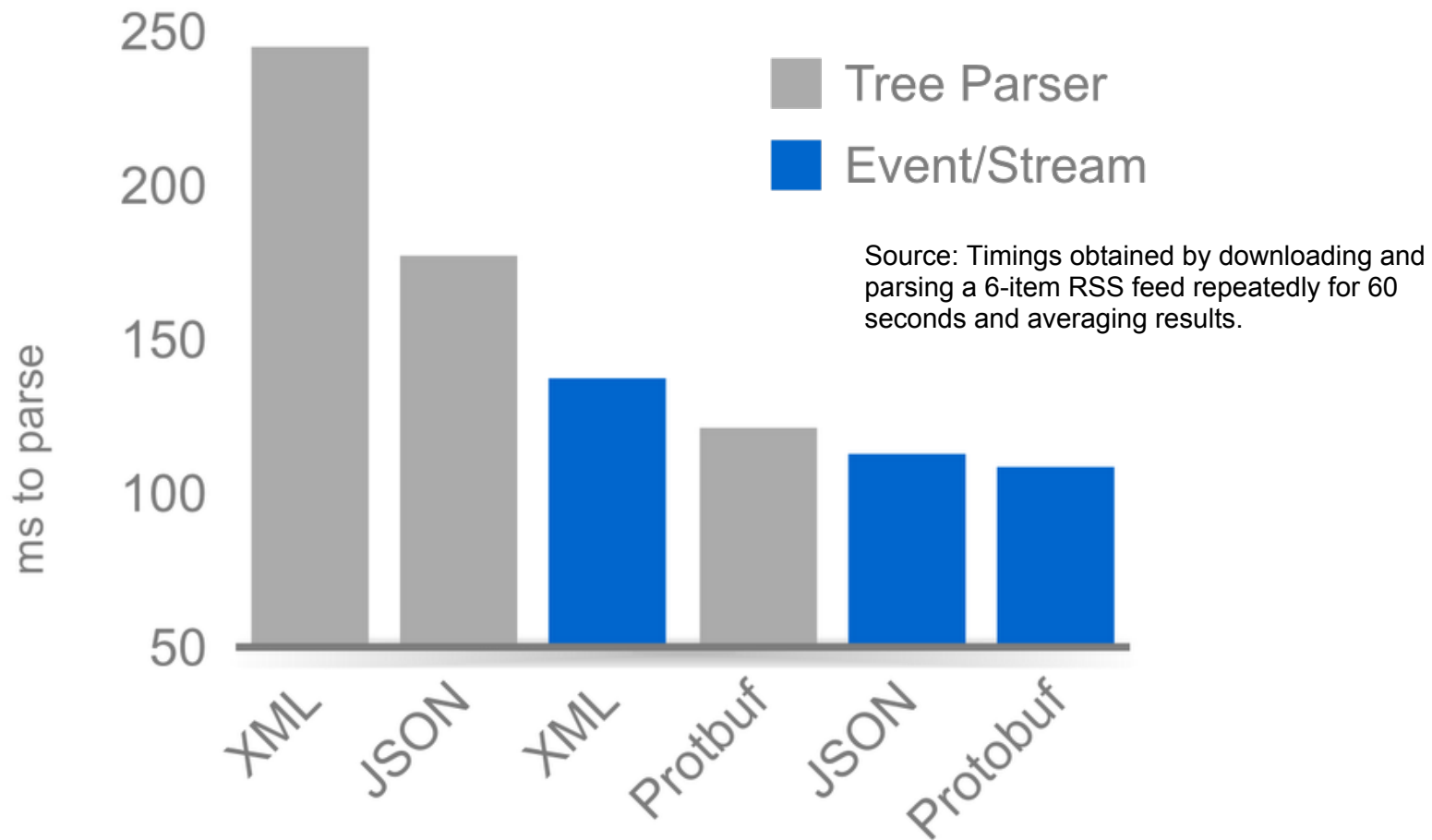
```
// Skip if no connection, or background data disabled
NetworkInfo info = mConnectivity.getActiveNetworkInfo();
if (info == null ||
    !mConnectivity.getBackgroundDataSetting()) {
    return false;
}
...
```



如何改进？

网络

- 使用高效率的数据格式和解析器



如何改进？

网络

- 使用高效率的数据格式和解析器
 - 使用 stream 式 XML 解析器, 而不用 DOM 解析器
 - 考虑使用二进制格式来融合二进制数据和文字数据在一个请求里
 - 减少往返服务器的次数以获得最佳用户界面效果

如何改进？

网络

- 尽可能使用 GZIP 压缩文字数据

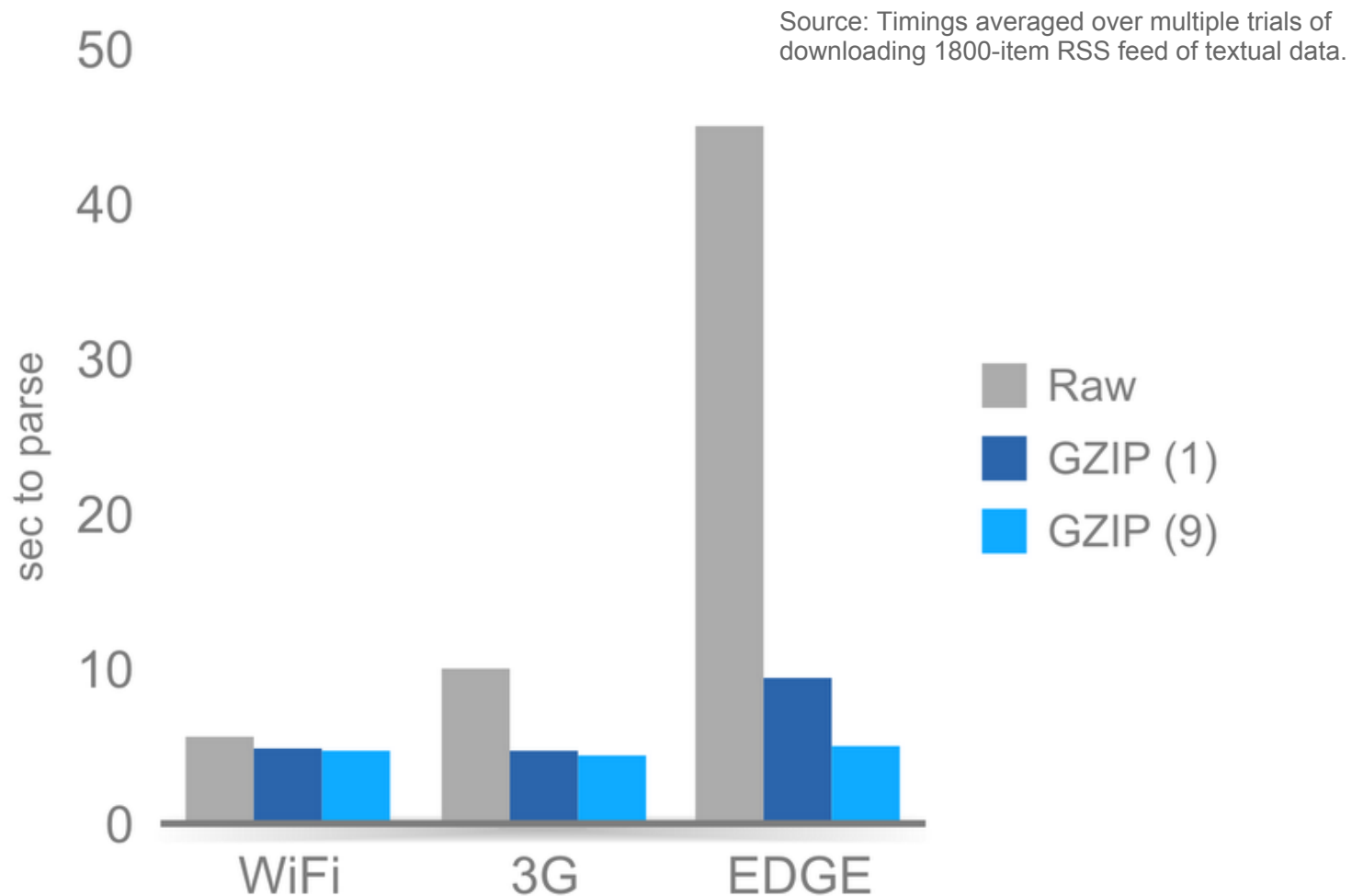
```
import java.util.zip.GZIPInputStream;

HttpGet request =
    new HttpGet("http://example.com/gzipcontent");
HttpResponse resp =
    new DefaultHttpClient().execute(request);
HttpEntity entity = response.getEntity();
InputStream compressed = entity.getContent();
InputStream rawData = new GZIPInputStream(compressed);
```

如何改进？

网络

- 尽可能使用 GZIP 压缩文字数据



如何改进？

前台程序

如何改进？

前台程序

- 忘记释放唤醒锁， 后果严重
 - 尽可能选择最低层的锁， 以防万一最好加一个超时
 - 考虑使用 `android:keepScreenOn` 确保正确性

```
<LinearLayout
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:keepScreenOn="true">
```



如何改进？

前台程序

- 回收Java 对象，特别是复杂对象
 - 系统会自动回收垃圾，但垃圾总是越少越好
 - XmlPullParserFactory 和 BitmapFactory
 - Matcher.reset(newString)
 - StringBuilder.setLength(0)
 - 回收策略在 ListView 里广泛应用，可供参考

如何改进？

前台程序

- 使用网络粗位置，会便宜很多
 - GPS: 25 秒 * 140mA = **1mAh**
 - 网络: 2 秒 * 180mA = **0.1mAh**
- 1.5 增加 AGPS 模块 
- GPS 自动定位时间受环境，准确性影响变化很大，有时会彻底失败
 - 和唤醒锁一样，如果没有释放，在 `onPause()` 后位置更新会继续工作
 - 如果所有程序都正确地释放位置更新，用户可以放心地在设置里使用 GPS

如何改进？

前台程序

- 由于没有 FPU, 浮点运算很昂贵
 - 在做大量地理经纬度运算时使用微度

```
// GeoPoint returns value 37392778, -122041944
double lat = GeoPoint.getLatitudeE6() / 1E6;
double lon = GeoPoint.getLongitudeE6() / 1E6;
```

- 在用 DisplayMetrics 做 DPI 工作时尽量缓存结果

```
float density =
    getResources().getDisplayMetrics().density;
int actualWidth =
    (int) (bitmap.getWidth() * density);
```



如何改进？

前台程序

- 加速传感器， 磁性传感器
 - 正常: 10mA (用做方向检测)
 - 用户界面: 15mA (大约一秒更新一次)
 - 游戏: 80mA
 - 最快: 90mA
- 在 HTC Dream 上, 加速传感器, 磁性传感器和方向传感器的耗电是一样的

如何改进？

后台程序

如何改进？

后台程序

- Service 应该是短暂的；而不是守护进程
 - 每个进程至少2MB，在前台程序需要内存时，Service 会被系统杀掉，然后重起
 - 内存用量大的会被最先杀掉
- 使用 AlarmManager 或 <receiver> 触发唤醒
 - 结束时别忘了stopSelf()

如何改进？

后台程序

- 用 AlarmManager 启动 Service
 - 若使用 RTC_WAKEUP, 谨慎
 - 例: 程序在非休眠时每30分钟更新一次

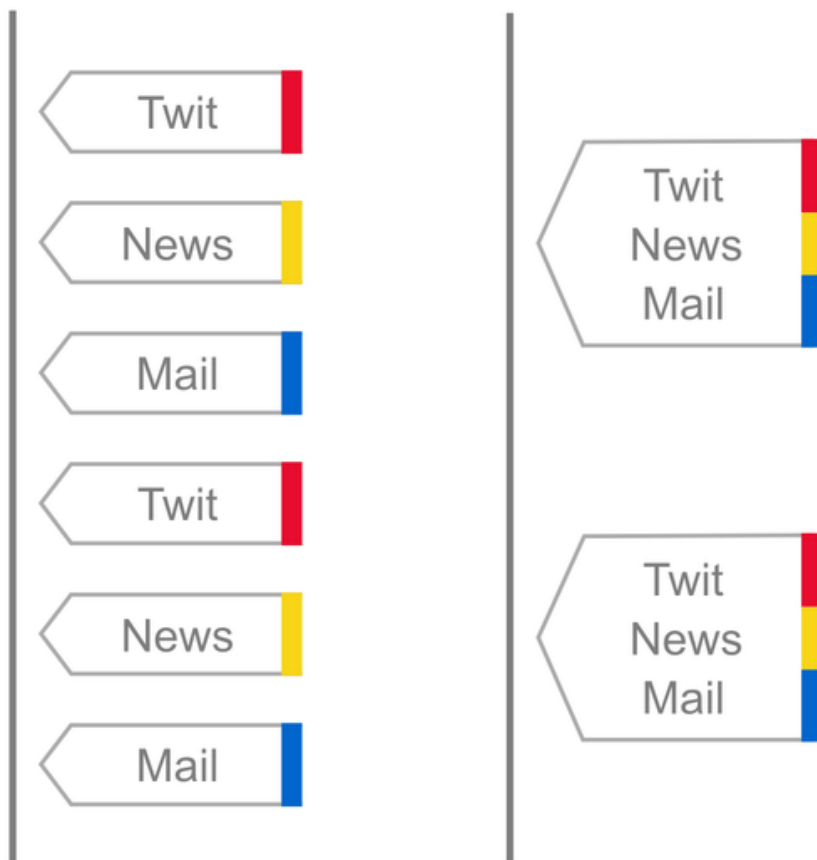
```
AlarmManager am = (AlarmManager)
    context.getSystemService(Context.ALARM_SERVICE);
Intent intent = new Intent(context, MyService.class);
PendingIntent pendingIntent =
    PendingIntent.getService(context, 0, intent, 0);

long interval = DateUtils.MINUTE_IN_MILLIS * 30;
long firstWake = System.currentTimeMillis() + interval;
am.setRepeating(AlarmManager.RTC,
    firstWake, interval, pendingIntent);
```

如何改进？

后台程序

- 使用 `setInexactRepeating()`, 这样系统可以把不同的更新组合在一起



如何改进？

后台程序

- 使用 <receiver>启动 Service
 - ConnectivityManager.CONNECTIVITY_ACTION
 - Intent.ACTION_TIMEZONE_CHANGED
 - Intent.ACTION_DEVICE_STORAGE_LOW
 - Intent.ACTION_BATTERY_LOW
 - Intent.ACTION_MEDIA_MOUNTED

```
<receiver android:name=".ConnectivityReceiver">  
  <intent-filter>  
    <action android:name=  
      "android.net.conn.CONNECTIVITY_CHANGE" />  
  </intent-filter>  
</receiver>
```

如何改进？

后台程序

- 动态设置 `<receiver>` 的使用/禁用状态，特别是在不做事时

```
<receiver android:name=".ConnectivityReceiver"  
    android:enabled="false">  
    ...  
</receiver>
```

```
ComponentName receiver = new ComponentName(context,  
    ConnectivityReceiver.class);  
PackageManager pm = context.getPackageManager();  
pm.setComponentEnabledSetting(receiver,  
    PackageManager.COMPONENT_ENABLED_STATE_ENABLED,  
    PackageManager.DONT_KILL_APP);
```

展望1.5之后

用户会监督你！



- SpareParts 有 “Battery history”
 - 从1.5开始系统记录每个程序 CPU, 网络, 唤醒锁的用量
 - 简化本会很快推出, 用户可以随时卸载滥用电池的程序
- 最好给用户电池用量的选择, 让用户决定更新间隔, 选择 “无背景数据”



总结

- 使用高效率解析器和 GZIP 来最好地利用网络和 CPU 的资源
- 不要用 sleep 或 poll 的 Service, 使用 AlarmManager 或 <receiver>
 - 当不工作时, 将 <receiver> 放在禁用状态
 - 使用 setInexactRepeating() 集中唤醒
- 等待良好的网络状况, 再做大量数据传送
- 提供用户管理背景行为的选择

Learn more at <http://code.google.com>



Google Developer Day 2009

