Android AlarmManager: Android 闹钟服务 AlarmManager

deepwaterooo

June 27, 2022

Contents

1 获取服务: 定时器，会到指定的时间时执行相应的操作。

```
AlarmManager mAlarmManager = (AlarmManager) getSystemService(Context.ALARM_SERVICE);
```

2 设置闹钟

2.1 常用的方法

```
set(type, triggerAtMillis, operation);
setRepeating(type, triggerAtMillis, intervalMillis, operation);
setInexactRepeating(type, triggerAtMillis, intervalMillis, operation);
cancel();
```

- set(type, triggerAtMillis, operation)：设置一个闹钟（这里指的是定时执行的任务），该任务不会重复执行。到时间后会执行 operation 指向的操作。其中 triggerAtMillis 指的是操作开始执行的时间，如果该时间小于当前的时间，那么会立即执行 operation。
- setRepeating(type, triggerAtMillis, intervalMillis, operation); 设置一个闹钟，该闹钟会定时、重复执行。其中 intervalMillis 指的是两次执行之间相隔的时间。其余的同上。
- setInexactRepeating(type, triggerAtMillis, intervalMillis, operation); 和 setRepeating 方法类似。区别是：setInexactRepeating 不一定会在指定的时间处执行，有可能会有偏差（是为了更节约能量），而 setRepeating 却是在给定的时刻执行，不会有偏差。
- cancel()：取消相应的操作。

2.2 获取时间方法

- System.currentTimeMillis: 获取系统时间，该时间是从 1970 年 1 月 1 日开始计时的，一直到系统当前的时间。要注意：由于系统的时间可以人为的进行修改，它得到的值也是从 1970 年到修改后的时间，并不一定是真正的当前时间。例如：当前 4 月 14 日，如果手机的时间被更改为 4 月 15 日，那么它得到的值就是到 4 月 15 日的，并不是到 4 月 14 的。
- SystemClock.elapsedRealtime()：得到的是系统开机后的时间，即使系统的时间不是真正的时间也可以得到正确的开机时长。并且把系统休眠的时间也计算在内。
- SystemClock.uptimeMillis：和上面的一个基本类似，唯一区别是：该方法不把系统休眠的时间计算在内。

3 Android4.4 以上

3.0.1 方法声明:

```
public void setWindow(int type, long windowStartMillis, long windowLengthMillis, PendingIntent operation)
/*
int type: 闹钟的类型，常用的有 5 个值:
AlarmManager.ELAPSED_REALTIME: 表示闹钟在手机睡眠状态下不可用，该状态下闹钟使用相对时间（相对于系统启动开始），状态值为 3;
AlarmManager.ELAPSED_REALTIME_WAKEUP: 表示闹钟在睡眠状态下会唤醒系统并执行提示功能，该状态下闹钟也使用相对时间，状态值为 2;
AlarmManager.RTC: 表示闹钟在睡眠状态下不可用，该状态下闹钟使用绝对时间，即当前系统时间，状态值为 1;
AlarmManager.RTC_WAKEUP: 表示闹钟在睡眠状态下会唤醒系统并执行提示功能，该状态下闹钟使用绝对时间，状态值为 0;
AlarmManager.POWER_OFF_WAKEUP: 表示闹钟在手机关机状态下也能正常进行提示功能，所以是 5 个状态中用的最多的状态之一，该状态下闹钟也是用
long windowStartMillis: 闹钟的开始时间
long windowLengthMillis 闹钟窗口时长
PendingIntent operation: 需要执行的操作
*/
```

- 再具体详解一下这几个常量

```
- ELAPSED_REALTIME: 真实时间流逝，当手机休眠时不进行相应的操作。
- ELAPSED_REALTIME_WAKEUP: 真实时间流逝。当手机休眠时也进行相应的操作。
- RTC: 手机时间流逝。当手机处于休眠状态时，不进行相应的操作。
- RTC_WAKEUP: 手机时间流逝。当手机处于休眠状态时，也进行相应的操作。
- ELAPSED 与 RCT 的最大区别：后者是根据手机的时间来的，也就是说可以通过更改手机上的时间，影响操作的执行时间；而前者却是真实的
- INTERVAL_DAY: 一天。
- INTERVAL_FIFTEEN_MINUTES: 十五分钟。
- INTERVAL_HALF_DAY: 半天。
- INTERVAL_HALF_HOUR: 半小时。
- INTERVAL_HOUR: 一小时。
```

3.0.2 示例:

```
// 设置一个 10 分钟后执行的闹钟，此方法只设置一次
mAlarmManager.setWindow(AlarmManager.RTC_WAKEUP,
                    System.currentTimeMillis() + 10 * minuteMills,
                    2 * minuteMills,
                    yourPendingIntent);
```

- 注意: 如果设置的起始时间小于当前时间，闹钟将会马上被触发。如果触发的事件里面又设置了一次闹钟（相当于递归），就会进入死循环。

3.1 Android4.4 以下:

3.1.1 方法声明:

```
// 注册一个新的闹铃
set(int type, long triggerAtMillis, PendingIntent operation)
// 注册一个新的闹铃，这个闹铃将在指定的时间被准确的执行
setExact(int type, long triggerAtMillis, PendingIntent operation)
// 注册一个对触发时间并不是很精准的闹铃，例如，一个闹铃每小时都会重复，但不一定都是在每个小时的最开始被触发
// triggerAtMillis 为闹钟首次执行时间，intervalMillis 为闹钟两次执行的时间间隔
```

```
setInexactRepeating(int type, long triggerAtMillis, long intervalMillis, PendingIntent operation)
// 注册一个重复类型的闹铃
setRepeating(int type, long triggerAtMillis, long intervalMillis, PendingIntent operation)
// 设定系统时钟时间
setTime(long millis)
// 设置系统默认时区
setTimeZone(String timeZone)
```

3.1.2 示例:

```
// 设置单次闹钟
mAlarmManager.set(AlarmManager.RTC_WAKEUP, System.currentTimeMillis() + minuteMills, yourPendingIntent);
// 设置重复闹钟
mAlarmManager.setInexactRepeating(AlarmManager.RTC_WAKEUP,
                    System.currentTimeMillis() + minuteMills,
                    10 * minuteMills,
                    yourPendingIntent);
取消闹钟
// 取消参数匹配的闹铃
mAlarmManager.cancel(yourPendingIntent);
闹钟为系统级别的一种通知，可以不依赖于应用的存活为条件，因此也可以用于应用挂掉后定时重启应用等场景。

// 定时重启
Intent intent = getPackageManager().getLaunchIntentForPackage(getPackageName());
PendingIntent restartIntent = PendingIntent.getActivity(getApplicationContext(), 0, intent, 0);
AlarmManager mgr = (AlarmManager) mContext.getSystemService(Context.ALARM_SERVICE);
mgr.set(AlarmManager.RTC, System.currentTimeMillis() + 500, restartIntent); //  定时重启应用
```