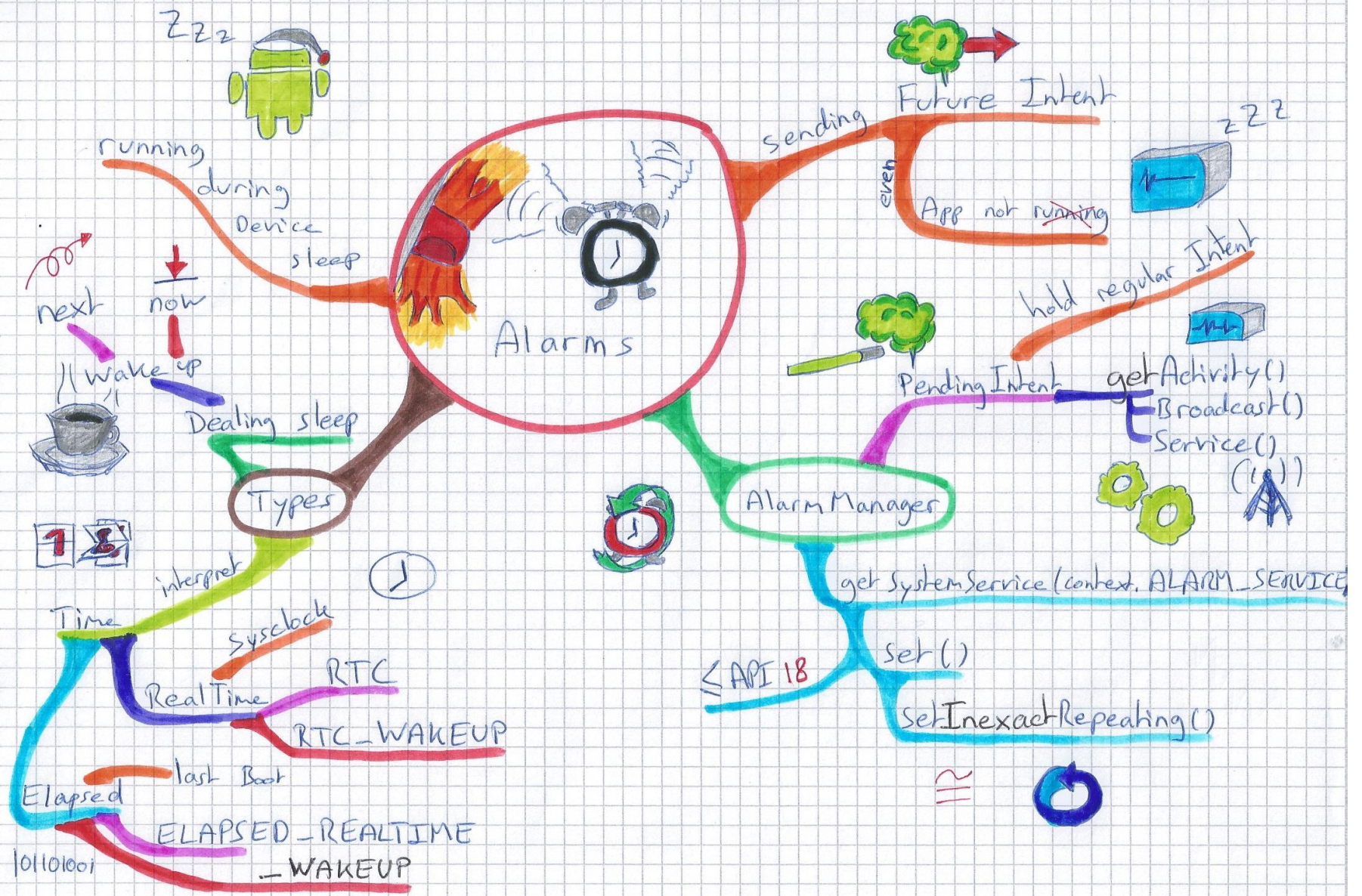


all chips

Android 5

21/02/14



Romain Chiappinelli

ALARMS

# TODAY'S TOPICS

ALARMS

ALARMMANAGER APIs

ALARM TYPES

EXAMPLE APPLICATION

# ALARMS

MECHANISM FOR SENDING INTENTS AT  
SOME POINT IN THE FUTURE

ALLOWS ONE APPLICATION TO MAKE CODE  
EXECUTE, EVEN WHEN THAT APPLICATION  
IS NO LONGER RUNNING

# ALARMS

ONCE REGISTERED, ALARMS REMAIN  
ACTIVE EVEN IF THE DEVICE IS ASLEEP

CAN SET CONFIGURE ALARMS TO WAKE  
A SLEEPING DEVICE

ALARMS ARE CANCELED ON DEVICE  
SHUTDOWN/RESTART

# ALARM EXAMPLES

MMS – RETRY SCHEDULER

SETTINGS – BLUETOOTH DISCOVERABLE  
TIMEOUT

PHONE – USER INFO CACHE



# ALARMMANAGER

CREATE & MANAGE ALARMS INDIRECTLY, BY  
INTERACTING WITH THE ALARMMANAGER

GET A REFERENCE TO THE ALARMMANAGER  
BY CALLING THE CONTEXT CLASS'

```
getSystemService(Context.ALARM_SERVICE)
```

# CREATING ALARMS

// one-shot alarm

```
void set(int type, long triggerAtTime,  
        PendingIntent operation)
```



# CREATING ALARMS

// repeating alarm

```
void setRepeating(int type,  
                 long triggerAtTime,  
                 long interval,  
                 PendingIntent operation)
```

# CREATING ALARMS

// repeating alarm with inexact trigger criteria

```
void setInexactRepeating(int type,  
                        long triggerAtTime,  
                        long interval,  
                        PendingIntent operation)
```

## INTERVAL OPTIONS

INTERVAL\_FIFTEEN\_MINUTES

INTERVAL\_HALF\_HOUR

INTERVAL\_HOUR

INTERVAL\_HALF\_DAY

INTERVAL\_DAY

# ALARM TYPES

TWO DEGREES OF CONFIGURABILITY

HOW TO INTERPRET TIME

WHAT TO DO IF THE DEVICE IS SLEEPING  
WHEN THE ALARM FIRES

# INTERPRETING TIME

REALTIME – RELATIVE TO SYSTEM CLOCK

ELAPSED – RELATIVE TO TIME SINCE  
LAST BOOT

# SLEEPING DEVICES

WAKE UP DEVICE NOW & DELIVER INTENT

WAIT TO DELIVER INTENT UNTIL DEVICE  
WAKES UP

# ALARM TYPE CONSTANTS

RTC\_WAKEUP

RTC

ELAPSED\_REALTIME

ELAPSED\_REALTIME\_WAKEUP

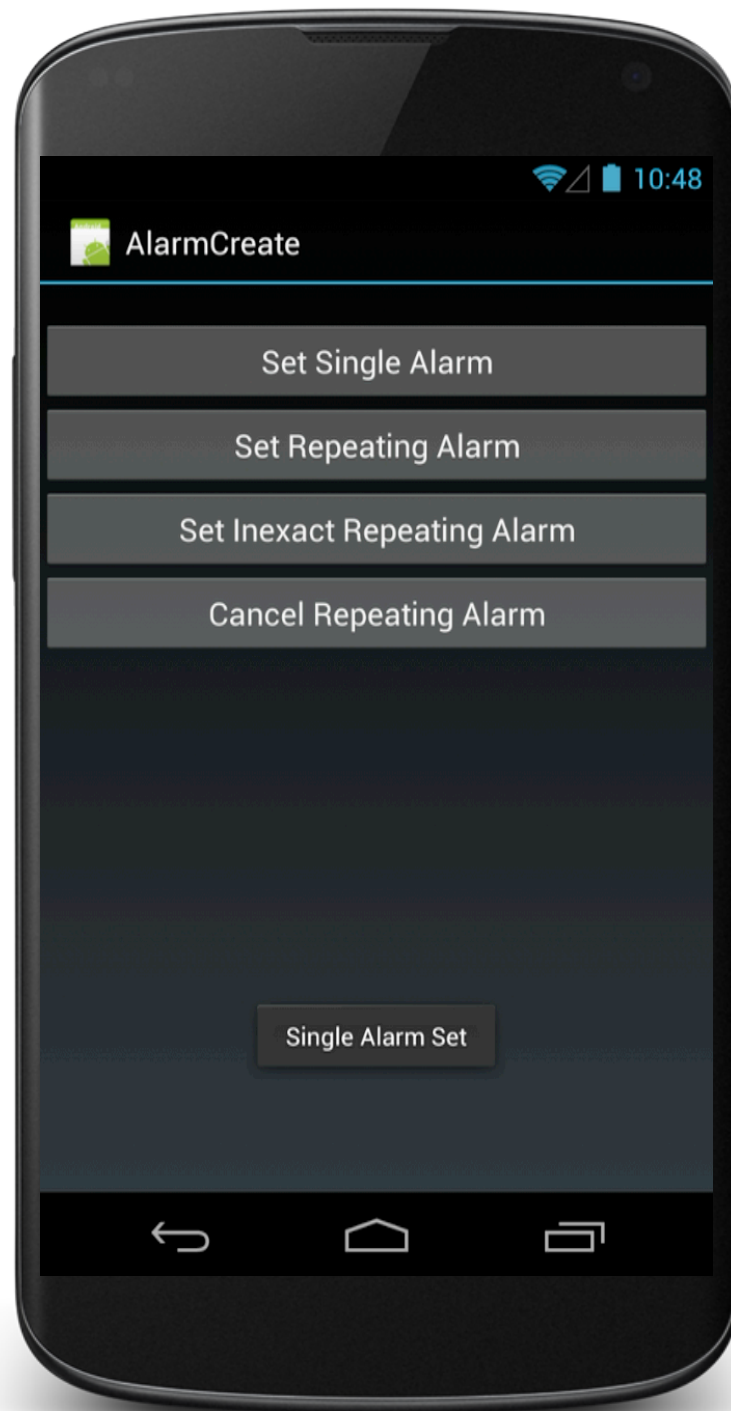
# PENDINGINTENT

```
PendingIntent getActivity(  
    Context context,  
    int requestCode, Intent intent,  
    int flags, Bundle options)
```

```
PendingIntent getBroadcast(  
    Context context,  
    int requestCode,  
    Intent intent, int flags)
```

```
PendingIntent getService(  
    Context context,  
    int requestCode,  
    Intent intent, int flags)
```





# ALARMCREATE

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.main);

    // Get the AlarmManager Service
    mAlarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);

    // Create an Intent to broadcast to the AlarmNotificationReceiver
    mNotificationReceiverIntent = new Intent(AlarmCreateActivity.this,
        AlarmNotificationReceiver.class);

    // Create an PendingIntent that holds the NotificationReceiverIntent
    mNotificationReceiverPendingIntent = PendingIntent.getBroadcast(
        AlarmCreateActivity.this, 0, mNotificationReceiverIntent, 0);

    // Create an Intent to broadcast to the AlarmLoggerReceiver
    mLoggerReceiverIntent = new Intent(AlarmCreateActivity.this,
        AlarmLoggerReceiver.class);

    // Create PendingIntent that holds the mLoggerReceiverPendingIntent
    mLoggerReceiverPendingIntent = PendingIntent.getBroadcast(
        AlarmCreateActivity.this, 0, mLoggerReceiverIntent, 0);
}
```

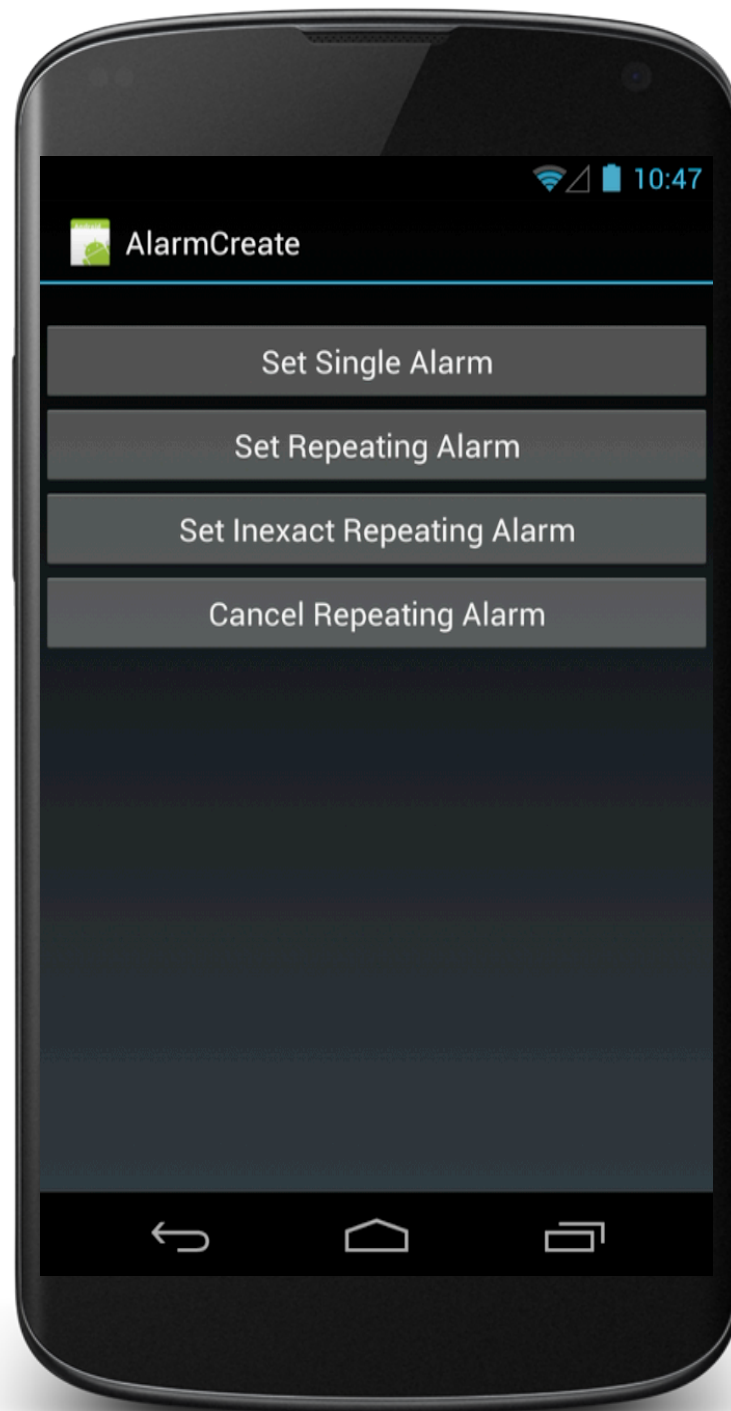
# ALARMCREATE

```
// Set up single alarm Button
final Button singleAlarmButton = (Button) findViewById(R.id.single_alarm_button);
singleAlarmButton.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {

        // Set single alarm
        mAlarmManager.set(AlarmManager.RTC_WAKEUP,
            System.currentTimeMillis() + INITIAL_ALARM_DELAY,
            mNotificationReceiverPendingIntent);

        // Set single alarm to fire shortly after previous alarm
        mAlarmManager.set(AlarmManager.RTC_WAKEUP,
            System.currentTimeMillis() + INITIAL_ALARM_DELAY
                + JITTER, mLoggerReceiverPendingIntent);

        // Show Toast message
        Toast.makeText(getApplicationContext(), "Single Alarm Set",
            Toast.LENGTH_LONG).show();
    }
});
```



# ALARMCREATE

```
// Set up repeating Alarm Button
final Button repeatingAlarmButton = (Button) findViewById(R.id.repeating_alarm_button);
repeatingAlarmButton.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {

        // Set repeating alarm
        mAlarmManager.setRepeating(AlarmManager.ELAPSED_REALTIME,
            SystemClock.elapsedRealtime() + INITIAL_ALARM_DELAY,
            AlarmManager.INTERVAL_FIFTEEN_MINUTES,
            mNotificationReceiverPendingIntent);

        // Set repeating alarm to fire shortly after previous alarm
        mAlarmManager.setRepeating(AlarmManager.ELAPSED_REALTIME,
            SystemClock.elapsedRealtime() + INITIAL_ALARM_DELAY
                + JITTER,
            AlarmManager.INTERVAL_FIFTEEN_MINUTES,
            mLoggerReceiverPendingIntent);

        // Show Toast message
        Toast.makeText(getApplicationContext(), "Repeating Alarm Set",
            Toast.LENGTH_LONG).show();
    }
});
```



# ALARMCREATE

```
// Set up inexact repeating alarm Button
final Button inexactRepeatingAlarmButton = (Button) findViewById(R.id.inexact_repeating_alarm_button);
inexactRepeatingAlarmButton.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {

        // Set inexact repeating alarm
        mAlarmManager.setInexactRepeating(
            AlarmManager.ELAPSED_REALTIME,
            SystemClock.elapsedRealtime() + INITIAL_ALARM_DELAY,
            AlarmManager.INTERVAL_FIFTEEN_MINUTES,
            mNotificationReceiverPendingIntent);
        // Set inexact repeating alarm to fire shortly after previous alarm
        mAlarmManager.setInexactRepeating(
            AlarmManager.ELAPSED_REALTIME,
            SystemClock.elapsedRealtime() + INITIAL_ALARM_DELAY
                + JITTER,
            AlarmManager.INTERVAL_FIFTEEN_MINUTES,
            mLoggerReceiverPendingIntent);

        Toast.makeText(getApplicationContext(),
            "Inexact Repeating Alarm Set", Toast.LENGTH_LONG)
            .show();
    }
});
```



# ALARMCREATE

```
// Set up cancel repeating alarm Button
final Button cancelRepeatingAlarmButton = (Button) findViewById(R.id.cancel_repeating_alarm_button);
cancelRepeatingAlarmButton.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {

        // Cancel all alarms using mNotificationReceiverPendingIntent
        mAlarmManager.cancel(mNotificationReceiverPendingIntent);

        // Cancel all alarms using mLoggerReceiverPendingIntent
        mAlarmManager.cancel(mLoggerReceiverPendingIntent);

        // Show Toast message
        Toast.makeText(getApplicationContext(),
            "Repeating Alarms Cancelled", Toast.LENGTH_LONG).show();
    }
});
}
```

NEXT TIME

NETWORKING