Android Developers

Monitoring the Battery Level and Charging State

When you're altering the frequency of your background updates to reduce the effect of those updates on battery life, checking the current battery level and charging state is a good place to start.

The battery-life impact of performing application updates depends on the battery level and charging state of the device. The impact of performing updates while the device is charging over AC is negligible, so in most cases you can maximize your refresh rate whenever the device is connected to a wall charger. Conversely, if the device is discharging, reducing your update rate helps prolong the battery life.

Similarly, you can check the battery charge level, potentially reducing the frequency of—or even stopping—your updates when the battery charge is nearly exhausted.

This lesson teaches you to

- Determine the Current Charging State
- Monitor Changes in Charging State
- Determine the Current Battery Level
- Monitor Significant Changes in Battery Level

You should also read

- Intents and Intent Filters

Determine the Current Charging State

Start by determining the current charge status. The `BatteryManager` (https://developer.android.com/reference/android/os/BatteryManager.html) broadcasts all battery and charging details in a sticky `Intent` (https://developer.android.com/reference/android/content/Intent.html) that includes the charging status.

Because it's a sticky intent, you don't need to register a `BroadcastReceiver` (https://developer.android.com/reference/android/content/BroadcastReceiver.html)—by simply calling `registerReceiver` passing in `null` as the receiver as shown in the next snippet, the current battery status intent is returned. You could pass in an actual `BroadcastReceiver` (https://developer.android.com/reference/android/content/BroadcastReceiver.html) object here, but we'll be handling updates in a later section so it's not necessary.

```
IntentFilter ifilter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
Intent batteryStatus = context.registerReceiver(null, ifilter);
```

You can extract both the current charging status and, if the device is being charged, whether it's charging via USB or AC charger:

```
// Are we charging / charged?
int status = batteryStatus.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
boolean isCharging = status == BatteryManager.BATTERY_STATUS_CHARGING ||
                     status == BatteryManager.BATTERY_STATUS_FULL;

// How are we charging?
int chargePlug = batteryStatus.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
boolean usbCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_USB;
boolean acCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_AC;
```

Typically you should maximize the rate of your background updates in the case where the device is connected to an AC charger, reduce the rate if the charge is over USB, and lower it further if the battery is discharging.

Monitor Changes in Charging State

The charging status can change as easily as a device can be plugged in, so it's important to monitor the charging state for changes and alter your refresh rate accordingly.

The `BatteryManager` (https://developer.android.com/reference/android/os/BatteryManager.html) broadcasts an action whenever the device is connected or disconnected from power. It's important to receive these events even while your

app isn't running—particularly as these events should impact how often you start your app in order to initiate a background update—so you should register a `BroadcastReceiver` (https://developer.android.com/reference/android/content/BroadcastReceiver.html) in your manifest to listen for both events by defining the `ACTION_POWER_CONNECTED` (https://developer.android.com/reference/android/content/Intent.html#ACTION_POWER_CONNECTED) and `ACTION_POWER_DISCONNECTED` (https://developer.android.com/reference/android/content/Intent.html#ACTION_POWER_DISCONNECTED) within an intent filter.

```
<receiver android:name=".PowerConnectionReceiver">
  <intent-filter>
    <action android:name="android.intent.action.ACTION_POWER_CONNECTED"/>
    <action android:name="android.intent.action.ACTION_POWER_DISCONNECTED"/>
  </intent-filter>
</receiver>
```

Within the associated `BroadcastReceiver` (https://developer.android.com/reference/android/content/BroadcastReceiver.html) implementation, you can extract the current charging state and method as described in the previous step.

```
public class PowerConnectionReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        int status = intent.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
        boolean isCharging = status == BatteryManager.BATTERY_STATUS_CHARGING ||
                             status == BatteryManager.BATTERY_STATUS_FULL;

        int chargePlug = intent.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
        boolean usbCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_USB;
        boolean acCharge = chargePlug == BatteryManager.BATTERY_PLUGGED_AC;
    }
}
```

Determine the Current Battery Level

In some cases it's also useful to determine the current battery level. You may choose to reduce the rate of your background updates if the battery charge is below a certain level.

You can find the current battery charge by extracting the current battery level and scale from the battery status intent as shown here:

```
int level = batteryStatus.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
int scale = batteryStatus.getIntExtra(BatteryManager.EXTRA_SCALE, -1);

float batteryPct = level / (float)scale;
```

Monitor Significant Changes in Battery Level

You can't easily continually monitor the battery state, but you don't need to.

Generally speaking, the impact of constantly monitoring the battery level has a greater impact on the battery than your app's normal behavior, so it's good practice to only monitor significant changes in battery level—specifically when the device enters or exits a low battery state.

The manifest snippet below is extracted from the intent filter element within a broadcast receiver. The receiver is triggered whenever the device battery becomes low or exits the low condition by listening for `ACTION_BATTERY_LOW` (https://developer.android.com/reference/android/content/Intent.html#ACTION_BATTERY_LOW) and `ACTION_BATTERY_OKAY` (https://developer.android.com/reference/android/content/Intent.html#ACTION_BATTERY_OKAY).

```
<receiver android:name=".BatteryLevelReceiver">
  <intent-filter>
    <action android:name="android.intent.action.BATTERY_LOW"/>
    <action android:name="android.intent.action.BATTERY_OKAY"/>
  </intent-filter>
</receiver>
```

It is generally good practice to disable all your background updates when the battery is critically low. It doesn't matter how fresh your data is if the phone turns itself off before you can make use of it.

In many cases, the act of charging a device is coincident with putting it into a dock. The next lesson shows you how to determine the current dock state and monitor for changes in device docking.