

droidcon UK 2014

Getting By With Less Than 1.21 Gigawatts ("Project Volta")



A History of Projects

- Jelly Bean's Project Butter: Smooth UI
 - 60fps, triple-buffering, etc.
- KitKat's Project Svelte: Run on Less RAM
 - `procstats`, more `inBitmap` flexibility, etc.
- Lollipop's Project Volta: Consume Less Power
 - `batterystats`, Battery Historian, `JobScheduler`



Lots of Advice...

- Reto Meier's Articles and Presentations
 - Power drain from connectivity
 - Power drain from finding locations
- Deprecations in AlarmManager and PowerManager
- WakefulBroadcastReceiver
- Fused location provider
- Pushing via GCM



...With Modest Adoption

- No Idea That We're a Problem
 - Granularity of battery “blame screen” in Settings
 - Expensive equipment for power measurement
- Ease of Development Trumps All
 - Power developers, large teams have time for low-level shenanigans
 - Average developer lacks time, expertise



What We Needed

- Better Tools for Power Consumption
 - Work with off-the-shelf hardware
- Easy APIs That “Do The Right Thing” Power-Wise
 - Modest migration costs for existing code
 - As easy as sub-optimal solutions for new code, new developers



What The L Did We Get?

- JobScheduler
 - AlarmManager as it was meant to be
- batterystats
 - Firehose of power-related event data
- Battery Historian
 - Firehose of power-related event data... in a nice HTML timeline visualization



JobScheduler

- A Smarter AlarmManager
 - Only gives you control when conditions warrant
 - Persists across reboots (yay!)
 - Clean builder-style API



Implementing JobScheduler

- Write a `JobService`
 - Add to manifest
 - Require `android.permission.BIND_JOB_SERVICE`
 - Override `onStartJob()`
 - Do your work asynchronously!
 - Call `jobFinished()` when done
 - Override `onStopJob()`
 - Stop the asynchronous work, as job conditions are no longer met



Implementing JobScheduler

- Schedule Your Jobs
 - Create, configure a `JobInfo.Builder`
 - `build()` the `JobInfo`
 - `schedule()` the `JobInfo` on a `JobScheduler`
 - Obtain from `getSystemService()`



Implementing JobScheduler

- Key Builder Options: Environment
 - `setRequiredNetworkCapabilities()`
 - Any, unmetered, or none (latter is default)
 - `setRequiresCharging()`
 - `setRequiresDeviceIdle()`
 - Device has not been used recently
 - Precise definition undocumented



Implementing JobScheduler

- Key Builder Options: One-Off Jobs
 - `setMinimumLatency()`
 - Don't even think about this job until this amount of time has elapsed
 - `setOverrideDeadline()`
 - Maximum latency before job executes regardless of other criteria
 - Must check network connection, etc. manually



Implementing JobScheduler

- Key Builder Options: Periodic Jobs
 - `setPeriodic()`
 - Supply an interval, will only run once per interval
 - Default: repeats until phone rebooted
 - `cancel()` or `cancelAll()` on `JobScheduler` to stop
 - `RECEIVE_BOOT_COMPLETED` to have survive reboots



Implementing JobScheduler

- Key Builder Options: Backoff Policy
 - Default: 5-second exponential
 - 5 second delay, then 25 seconds, then 125 seconds...
 - `setBackoffPolicy()` to configure
 - Time and linear vs. exponential backoff



I Can Haz Backport?

- No `JobSchedulerCompat` Presently
- Should Be Possible
 - Some features, like idle awareness, would be ignored
 - Other features might be less optimal than native 5.0 edition



batterystats

- Yet Another adb shell dumphsys Category
 - Obtained individually or part of the full report

```
adb shell dumphsys batterystats
```



batterystats

- Notable Switches
 - --reset: Clears battery data for fresh analysis
 - --unplugged: Only show results since last unplugged
 - --charged: Only show results since last charged

```
adb shell dumpsys batterystats --reset
```



batterystats

- Notable Commands and Options
 - `your.package.name.here`
 - `--enable full-wake-history`
 - `--enable no-auto-reset`

```
adb shell dumpsys batterystats --enable full-wake-history
```



Battery Historian

- Timeline Rendering of batterystats Output
 - HTML generated by Python script
- Obtaining Battery Historian
 - Download `historian.py` from <https://github.com/google/battery-historian>



Battery Historian

- Generate batterystats, Run Script, Browse

```
adb shell dumpsys batterystats --enable full-wake-history
adb shell dumpsys batterystats --reset
```

```
// run tests here
```

```
adb shell dumpsys batterystats > /tmp/bs.txt
python historian.py /tmp/bs.txt > /tmp/bs-report.html
```



Now What Do We Need?

- Documentation!
 - batterystats
 - Battery Historian
 - Other Volta impacts (e.g., low-power mode)
- JobSchedulerCompat
 - Official
 - Independent



Now What Do We Need?

- “PowerLint”
 - Easy to run, easy to interpret “you're doing something bad”
 - `StrictMode.PowerPolicy` would be nice...
- Recipes, tools, for device-only testing
 - No adb USB, WiFi effects on power drain
- Easy power-friendly libraries and recipes



Slides 'n Stuff!



<http://commonsware.com/presos/droidconUK2014>

