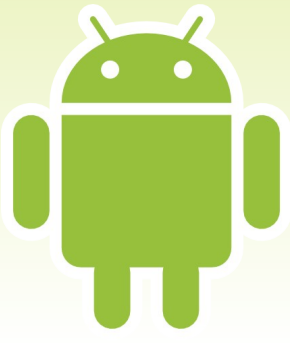
DevFest DC 2015

What You Should Be Doing About Android 6.0

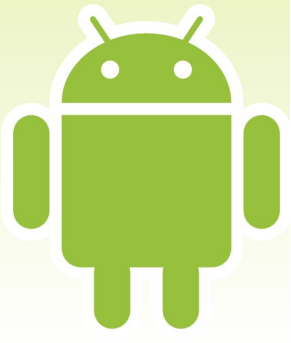


Levels of Panic

- Worry About Now
  - Runtime permissions
    - And other permission changes
  - “War on background processing”
  - Backups
  - Adoptable removable storage

Levels of Panic

- Worry About If...
  - App links
    - ...if you expose `http` URL schemes
  - Direct share targets
    - ...if you support ACTION_SEND
  - Apache HttpClient Removal
    - ...if you have been using HttpClient, despite long-standing recommendations not to



Runtime Permissions

- Legacy Apps (`targetSdkVersion` < 23)
  - No code changes
  - Behavior is akin to “app ops”
    - User can revoke `dangerous` permissions in Settings
    - Affected APIs return bogus results

Runtime Permissions

- Modern Apps (`targetSdkVersion` >= 23)
  - Same `<uses-permission>` elements
  - Must request dangerous permissions
    - Modal dialog-style UI
    - User can accept, deny, or deny with extreme prejudice

Runtime Permissions

- Possible Runtime Permission States
  - We have never asked for the permission
  - We asked for the permission, and got it
  - We asked... and the user rejected the request
    - Or, granted it earlier and rejected it later in Settings
  - We asked... and the user rejected the request, plus took out a restraining order to prevent us from asking again

Runtime Permissions

- Key Methods
  - `checkSelfPermission()`
  - `requestPermissions()`
    - Triggers `onRequestPermissionsResult()`
  - `shouldShowRequestPermissionRationale()`

Other Permissiony Things

- Nuked Permissions
  - E.g., `USE_CREDENTIALS`
- Double-Opt-In Permissions
  - `WRITE_SETTINGS`
  - `SYSTEM_ALERT_WINDOW`

War on Background Processing

- "Doze Mode"
  - Device has not moved and is not charging for 1+ hours
  - Background processing becomes infrequent
    - Once every few hours

War on Background Processing

- "App Standby"
  - Device is not charging and user has not been in your app in a long time
  - Background processing becomes infrequent
    - Once a day

War on Background Processing

- What Sort of Background Processing?
  - No AlarmManager alarms
  - No JobScheduler jobs
  - No SyncManager syncs
  - Outstanding partial WakeLocks ignored

War on Background Processing

- Avoiding “App Standby”
  - User whitelist via Settings
  - Notification
  - `setAndAllowWhileIdle()` and `setExactAndAllowWhileIdle()` on AlarmManager
- Avoiding “Doze Mode”
  - GCM

Backups

- Good News!
  - Backups are automatic, and back up everything
    - No more BackupManager and BackupAgent stuff
- Bad News!
  - All your data are belong to Google

Backups

- Issues
  - What about regulations?
    - E.g., HIPAA?
  - What about lawsuits?
    - What is a developer's legal exposure if Google gets Ashley Madison'd?

Backups

- Steps
  - Opt-out (`android:allowBackup="false"`)
  - Talk to legal counsel
  - Work out a plan
    - Remain opted-out
    - Opt-in for some data
    - Opt-in for all data, but encrypt it with user-provided key
    - Opt-in for everything

Adoptable Removable Storage

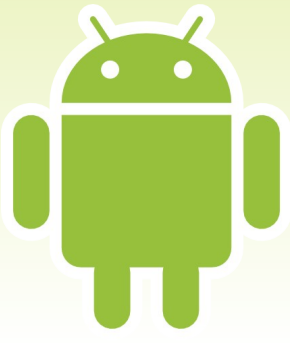
- Good News!
  - Users can extend internal storage via removable media, can move apps to this extended area
- Bad News!
  - Brings back Android 1.x/2.x-era issues
  - Set up `android:installLocation`

App Links

- For Apps Handling Specific URLs
  - E.g., Twitter handling http://twitter.com
- Classic Effects
  - `ACTION_VIEW` requests for matching URL bring up a chooser, including browsers
  - Browsers might handle URL internally, ignoring apps

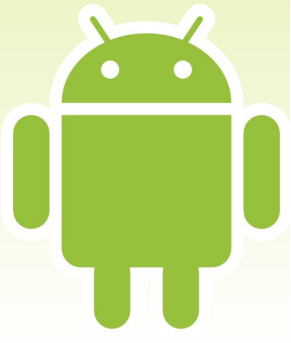
App Links

- App Links: Avoid the Chooser
  - Advertise in `<intent-filter>` that you support this
  - Have metadata on server at that domain tying domain to your app
    - Application ID + hash of public key
  - Result: no more chooser, `ACTION_VIEW` will resolve directly to your app



Direct Share Targets

- `ACTION_SEND`
  - Client says what to share
  - User chooses activity for sharing
  - Must navigate within that activity to provide more specific context
    - Contact, folder, etc.
- Direct Share Targets = list of likely contexts
  - Appear in chooser, alongside other activities

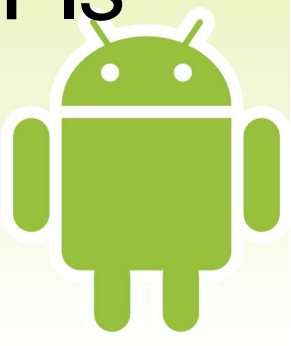


Direct Share Targets

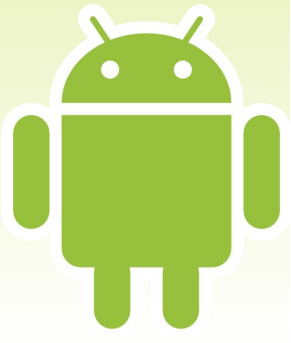
- Steps
  - Create subclass of ChooserTargetService
    - Implement `onGetChooserTargets()` to create ChooserTarget list
    - ChooserTarget has PendingIntent to be invoked if user chooses that direct share target
  - Add `<meta-data>` to `ACTION_SEND` activity pointing at service

Apache HttpClient

- This is an ex-API!
- Options
  - Gradle snippet to still allow builds
  - Move to Apache's standalone Android-friendly HttpClient JAR
  - HttpUrlConnection
  - OkHttp or other modern third-party APIs



Oh, Yes... One More Thing...



AnalogClock is deprecated

Slides 'n Stuff

https://commonsware.com/presos/devfestDC2015