Galaxy Portal

Interacting with the galaxy platform through mobile devices
Motivation

• Galaxy jobs run in the background
• You are often awaiting Galaxy job completion for a next analysis step
• Your smartphone is often more accessible than your computer
• Galaxy can be accessed through the web browser of your phone
  – But Galaxy not designed for small screen and large thumbs
The Galaxy Portal App

• You can now monitor Galaxy jobs through a native mobile phone app
Galaxy Portal App Features

• Storing user credentials
  – Stores URL, username and API key
  – Supports multiple instances

• Responsive hierarchical navigation on small screen
  – Select history -> select elements

• Color-coded status indicators
  – Queued, running, finished and failed runs
Galaxy Portal App Features (cont.)

• Status polling
  – Including audible notifications

• Viewing results of jobs
  – (for now supported for txt and html)

• Access to shared datasets
  – Inspect shared results and job parameters on the subway

• Re-running of jobs (experimental)
  – Inspect parameter choices, modify choices, execute
  – (Slow and not supporting all interfaces)
Technology/implementation

• Both iOS and Android have considerable market share
  – A cross-platform App saves development effort
• Many alternative cross-platform alternatives
  – Pure web based, Ionic, Cordova
• We selected QT
  – Native compilation, QML for UI design, well documented and supported
• Uses the Galaxy API
• Logic implemented in javascript
Challenges and limitations:

• Practical: Development for multiple platforms requires computers and devices for each platform
• iOS development more cumbersome and expensive than Android
• Licensing turned out a bit tricky with Qt and iOS
• Galaxy API is not as mature as web interface
  – The current solution of rerunning jobs is slow due to lacking rerun API and need for ID conversion
Open questions:

• Are platform distribution channels worth the additional effort required?
  – Native app vs web app
  – As icon or through browser
  – To be discovered in app store or one the web

• Should Galaxy server and Galaxy front-end be separate (depends on Galaxy API)
Future work:

• Improved results inspection in the App
  – Support more result formats and scale well
• Support external access authentication
• Improvements to Galaxy API would open new opportunities
  – API support for job running and re-running
  – Push notification system would give opportunities to third party apps
• Other suggestions?
• Everyone is welcomed to contribute to the open source codebase (github)