Our Journey from Java to PyQt for CERN Accelerator Control GUIs

Ivan Sinkarenko, Sara Zanzottera, Vito Baggiolini
<table>
<thead>
<tr>
<th>Target</th>
<th>I/E11</th>
<th>MUL</th>
<th>%SYM</th>
<th>Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>27.0</td>
<td>17</td>
<td>94 a</td>
<td>H2/H4</td>
</tr>
<tr>
<td>T4</td>
<td>43.7</td>
<td>4</td>
<td>93 a</td>
<td>H6/H8</td>
</tr>
<tr>
<td>T6</td>
<td>123.4</td>
<td>2</td>
<td>97 a</td>
<td>COMPASS</td>
</tr>
<tr>
<td>T10</td>
<td>21.4</td>
<td>0</td>
<td></td>
<td>NA62</td>
</tr>
</tbody>
</table>

Phone: 77500 or 70475
Comments (07-Oct-2018 21:39:10)
Software developers

Physicists, operators, hardware experts
Oracle has pulled out from GUI development

WAIT....

WHAT?
Evaluation

We did a lot of reading…

- Extremely popular nowadays
- Goes beyond browser:
  - Desktop (Electron)
  - Server-side (Node.js)

- Became popular through desktop GUIs
- Is familiar to our user community

Jobs on indeed.com (as of Nov 2018):
- JavaFX (tens) < Java Swing << Qt (hundreds) << Angular (thousands)
Web

- Easy to find new people
- Successfully used by software teams in Controls group
- Deployment and upgrades are smooth

! Hard to integrate with existing control system libraries
! Learning curve for existing community
! Rapidly changing technology

It just did not feel a right option for our user community
Qt

Java/Qt
- First idea to rescue all existing investment

QML+JavaScript (QtQuick)
- Use new promising technology
- Be closer to web

Python/Qt
- Stay with stable and familiar technology
Java + Qt

- Looked like an ideal scenario
  - Reuse existing client libraries, dev tools, processes
- Qt Jambi
  - Java bindings for Qt framework
  - Very obsolete (discontinued 10 years ago)
  - We did not manage to run it
- Own bindings?
  - No QML Java bindings existed
  - We could create our own QML bindings (with DOtherSide)
  - Much smaller API than Qt Jambi
  - We did a proof of concept
  - But, it would be too exotic
QtQuick

- QML + JavaScript
  - QML – user interface markup language
  - Logic implemented in JavaScript
    - Typical app has a C++ entrypoint that launches QML engine

- Can JavaScript (or better TypeScript) be our main GUI language?
- Control libraries would be connected via Qt plugin mechanism

- We made some proofs of concept
  - Charting
  - Complex tables
  - Copy of UI of existing Java applications
QtQuick

We faced a number of problems

- No bindings existed for 3rd-party libraries (e.g. charts)
- Not all desktop widgets well supported (e.g. trees and tables)
- How to split code between QML documents and external JavaScript?
- Debugging was supported only in Qt Creator
- Declarative style felt awkward for our users
- We wanted a typed language but TypeScript was not supported

We felt that QtQuick was not ready for us
Python + QtWidgets

✓ PyQt is very popular and familiar to our community
✓ We are confident that QtWidgets will not go away in the future
~ We can integrate with our Java libraries (using JPype temporarily)

⚠ Dependency management can get tricky
~ Python is a dynamic language

At this point we made the decision, and never reconsidered it since
Conclusion

- We will phase out JavaFX within the next 5 years
- Web is not our first choice, though can be attractive in certain cases
- We choose QtWidgets, as QtQuick experience was bumpy
- We have settled with PyQt
Since then we’re…

- Creating CERN-specific widgets & charts
- Bringing PyQt-based Rapid Application Development (RAD) framework
- Organizing PyQt training for our users
- Integrating PyQt with control system services
- Contributing to the open-source community
Thank you!