Qt Quick Tooling

How to make the most out of Qt Quick
Kai Köhne

• Long time Qt Developer (and KDE user)
• kai.koehne@digia.com
• kkoehne on freenode

• Qt Creator
  – Qt Quick Designer
  – QML Debugger
  – QML Profiler
• Installer Framework
• MinGW installer
• Qt 5 Release Team
Goals

- Let you know what's there
- Get feedback
- Hope: KDE embraces & extends tooling infrastructure
## Overview

<table>
<thead>
<tr>
<th></th>
<th>GUI (Qt Creator)</th>
<th>Command Line</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Coding</strong></td>
<td>QML/JS Editor Qt Quick Designer</td>
<td></td>
</tr>
<tr>
<td><strong>Debugging</strong></td>
<td>QML Debugger Inspector Console</td>
<td>Console APIs</td>
</tr>
<tr>
<td><strong>Profiling</strong></td>
<td>QML Profiler</td>
<td>Console APIs qmlprofiler</td>
</tr>
<tr>
<td><strong>Deploying</strong></td>
<td>Application Templates</td>
<td>qmlbundle qmlmin</td>
</tr>
</tbody>
</table>
Coding

- Qt Creator
  - Syntax Highlighting
  - Semantic Checks
  - Code Navigation
  - Code Refactoring
  - Qt Quick Designer

- Kate / KDevelop

- Plasmate
Coding - .qmltypes files

- Metadata about types in C++ plugins
- Should be part of every deployed plugin
  - qmlplugindump -notrelocatable MyModule 1.0
  - Good way to check for API changes!
Coding - Qt Quick Designer

- Write self-contained elements
- Prefer declarative style
- Expect different API usages in C++ types
Debugging

- Environment variables
  - QT_DEBUG_PLUGINS
  - QML_IMPORT_TRACE
  - ...
  (grep for BOOL_CONFIG_OPTION)

- JS API
  - console.log()

- JS API - Qt 5
  - console.log(), console.warn(), console.error(), console.assert() ...
  - console.trace()
  - console.exception()
Debugging – Qt Creator

- **JS Debugger**
  - Breakpoints, stepping, variable inspection ...
  - Mixed with C++ debugger or standalone

- **QML Inspector**
  - Objects & Properties
  - (→ GammaRay)

- **Inspector tools**
  - Apply on Save
  - Select/Zoom

- **JS Console**
Debugging – Remote API

- Qt Creator debugger/profiler needs in-process functionality

- Enable TCP Server:
  - Compile with QT_DECLARATIVE_DEBUG define (Qt Quick 1)
  - Compile with QT_QML_DEBUG define (Qt Quick 2)
  - Qt qmltooling/ plugins required at runtime

- Application is started with '-qmljsdebugger=port:XXX,block'

- Protocol is extensible!
Profiling

- Qt Creator QML Profiler
  - QML engine state
    - Painting (Qt 5: Animations)
    - Compiling
    - Creating
    - Binding
      - Handling Signal
    - Debug binding loops
    - Check for optimized bindings
    - Qt 5: V8 sample based profiler

- Environment Variables
  - QML_SHOW_FRAMERATE (Qt 4)
  - QSG_RENDER_TIMING (Qt 5)

- JS API (Qt 5)
  - console.time(), console.timeEnd()
  - console.trace(), console.traceEnd()

- qmlprofiler command line tool
Deployment

- Different paradigms
  - Viewer vs standalone
  - Files vs Qt resources vs bundles
  - ...

- One topic for the QtCS :)
Thank you!

www.qt.digia.com
(btw, we're hiring ;)

© 2013 Digi