What's new in QtWebKit in Qt 5
(Simon Hausmann)

WebKit in a nutshell

Statistics

- WebKit:
  - ~1.5 Million lines of code
  - ~90 Commits per day
- Qt 5 release modules (w/o WK):
  - ~3.5 Million lines of code
  - qtbase: ~19 commits per day

Summary

- WebKit is half the size of Qt
- It's changing rapidly

So what has changed over the past months?

A LOT!
(Let's look at the diff)

Outline

- WebCore
- Qt C++ API
- Process separation
- QML2 integration
- Future thoughts

WebCore features

- Make sure to subscribe to weekly updates at planet.webkit.org
- Examples:
  - Mathml
  - CSS Regions
  - Shadow DOM
  - window.requestAnimationFrame()
  - remote web inspector
  - Too many more to count!

WebCore performance

- New faster CSS lexer
- Massive reductions in DOM/CSS memory usage
- 25% faster line breaking for complex text
- 8 times faster querySelector

What features am I using?

Outline

- WebCore
- Qt C++ API
- Process separation
- QML2 integration
- Future thoughts

Qt C++ API

- No big changes
- Mostly source compatible
- Widget based API in QtWebKitWidgets module

Rendering pipeline improvements

- QGraphicsView based layer composition replaced with OpenGL based compositor
  - Small kernel
  - Easy to adapt to specific GPUs
  - Will allow for integration with CSS shaders/filters
  - Supports tiled layers
  - TextureMapper also supports software fallback

How to use?

```
1:
2:  GraphicsWebView* webView = new QGraphicsWebView;
3:  webView->setResizesToContents(true);
4:
5:
6:  graphicsView->setViewport(new QGLWidget);
7:
```

Outline

- WebCore
- Qt C++ API
- Process separation
- QML2 integration
- Future thoughts

Problems with QtWebKit

- Single threaded
- Delay in rendering can cause stutter when scrolling
- API mixes high-level and low-level needs
- Hard to secure even with threaded engine

Introducing WebKit2

- Separate front-end (UIProcess) from back-end (WebProcess)
- Messages passed via local sockets
- Data transferred via shared memory

WebKit2 process responsibilities

- WebProcess:
  - Runs one or multiple web pages
  - Executes JavaScript
  - Lays out content
  - Renders into shared bufffers
  - Can be sandboxed (future)
- UIProcess:
  - It's your application process, running QApplication, QML
  - Never blocks main thread for any web related activity
  - Always responsive to user feedback

WebKit2 Networking

- WebProcess initiates network connections
- No more custom QNetworkAccessManager
- But will allow application provided schemes in the future

WebKit2 Hybrid (future)

- No more QObject bindings (across process boundaries)
- Standard postMessage() API between web content and QML
- JavaScript on both ends

WebKit2 Rendering

- Web process has tree of layers
- Layer content rendered in software
- Uploaded into shared GPU memory
- Layer tree replicated in Qt application process
- Composited in QML2 scene with OpenGL texture mapper
- Called coordinated graphics system

WebGL

- WebGL rendered into off-screen GL surface in web process
- Surface is shared between processes
- WebGL canvas layer composited as part of regular layer tree
- Maybe serialize GL commands in the future?

WebKit2 rendering result

- It's fast
- Always responsive
- Minimize GPU texture uploads

Outline

- WebCore
- Qt C++ API
- Process separation
- QML2 integration
- Future thoughts

New WebKit QML2 Integration

- Philosophy:
  - Don't try to do too many things at the same time
  - Implement one well-defined behaviour
  - Less to tune, more things work out of the box

New QML2 API

- WebView is a QML flickable
- Supports pinch/pan gestures out of the box
- Viewport meta-tag out of the box
- Touch adjustment
- Tap highlighting

Quick Demo

Ideas for the future

- Low-level API
  - Allow for products that integrate WebKit deepy into the system
  - Not as easy to use, but more powerful

Known issues...

- Audio/Video support
- HW acceleration for 2D Canvas

Conclusion

- Many things have changed
- It's important to stay up-to-date with WebKit
- Stay in touch with us in the WebKit project

Thank you!
Questions?