KDE Frameworks 6
Are we there yet?

Akademy 2023

Alexander Lohnau, Nicolas Fella, Volker Krause
We have been here before

- Akademy 2021: What’s cooking for KDE Frameworks 6?
- Akademy 2021: KF6 the Architecture Overview - Time to Slice Things Up Yet Again
- Qt DevCon 2022: KDE’s journey to Qt 6
- Akademy 2022: KDE Frameworks 6 - Plans and Progress
- Akademy 2022: Getting your application ready for KF6
- QtCon Brasil 2022: KDE’s journey to Qt 6 and beyond
How it started

• Kickoff BoF at Akademy 2019
• Sprints:
  – 2019 in Berlin
  – 2021 online
• Regular online meetings
Goals

- Use Qt6
- Easy porting
- Better APIs
- Better QtWidgets/QtQuick separation
- Reduce dependencies
- Better interface/implementaion separation
- Better cross-platform support
“This is a quote, words full of wisdom that someone important said and can make the reader get inspired.”
“Are we there yet?”
Sort of

- iskdeusingqt6.org tracks Qt6 porting progress (380/528)
- Frameworks & Plasma branching early 2023
- Plasma 6 session is usable
- KF6 Workboard: 260 tasks done
Co-existence and mixed sessions

- KDE Frameworks 5 and 6 need to co-exist
  - Qt 5 apps in a Plasma 6 session
  - Qt 6 apps in a Plasma 5 session
  - KF5 and KF6 apps in a non-Plasma session
  - Non-Qt apps in a Plasma 6 session
  - ...

Co-existence and mixed sessions

- File/path collisions (static assets, executables, etc)
- D-Bus service names conflicts
- Unique platform services: KWallet
- Platform integration: Styles, file dialogs, KIO worker, etc
- Generic application plugins: KParts, thumbnailer, etc
- Environment variables: $KDE_SESSION_VERSION
Approaches

- **Versioning**
  - Often already present, simple to adjust
  - Needs adjustments in all consumers

- **Exclusivity**
  - Might need build options retrofitted in KF5 components
  - Needs adjustments to packaging
  - Requires compatibility across major versions
Approaches

● Multi-builds
  - Convenient for things that don’t diverge much
  - Versioned libraries + shared assets/data files
  - Limits retirement options for 5-based builds
Implicit/assumed APIs and external users

- Handling of $KDE_SESSION_VERSION varies
  - Concatenate to executable or D-Bus service base names
  - Error out on unknown versions
- Widely used outside of our own code
  - xdg-utils
  - Chromium
  - QtKeychain
  - LibreOffice
More challenges

- ECM cannot build APKs yet
- KIO HTTP
- ...

- KDE Frameworks 6 BoF, Tuesday 16:00 in room 1!
kdesrc-build setup for porting your apps

• Like always, kdesrc-build is your friend and helper
• Takes care of few things for kf6 builds:
  – Using correct CMake arguments
  – Choosing correct branches
  – Compiling third party packages
KDESRc-build setup for porting your apps

- KF6 builds are configured by global branch group
  - branch-group kf6-qt6
- Separate from kf5 prefix due to co-installability issues of projects
- Configured using kdedir, source-dir and build-dir
• Custom config passed in using
  \texttt{--rc-file=/home/user/kde6/kdesrc-buildrc}

• More comfortable using simple bash/fish alias:
  \begin{itemize}
    \item \texttt{alias kdesrc-build6="kdesrc-build --rc-file=$HOME/kde6/kdesrc-buildrc"}
  \end{itemize}

• Let’s get started on your app!
How to approach porting your app?

- First step, as last year: Disable deprecated API
  - Ideally with latest Frameworks
- Adjusting the buildsystem:
  - Versionless Qt targets
  - Use \${QT\_MAJOR\_VERSION} for find_modules calls and KF targets
How to approach porting your app?

- Make sure QtVersionOption is included
  - Usually included by KDEInstallDirs already

```cpp
find_package(Qt${QT_MAJOR_VERSION} ${QT_MINOR_VERSION} CONFIG REQUIRED Core Widgets)
find_package(KF${QT_MAJOR_VERSION} ${KF_MINOR_VERSION} REQUIRED COMPONENTS
  I18n
  ItemModels
)
target_link_libraries(gwenview KF${QT_MAJOR_VERSION}::ItemModels ...)
```

- Check out kf6 buildsystem script in kde-dev-scripts
How to approach porting your app?

- Not all changes caught by deprecation macros
  - Changes to virtual methods
  - Classes moved/renamed
- Fixable using preprocessor if statement

```cpp
#if Qt_VERSION_MAJOR < 6
  : KQuickAddons::ConfigModule(parent, data, args)
#else
  : KQuickConfigModule(parent, data)
#endif
```
How to approach porting your app?

- QML runtime issues more difficult to port
- Possible using CMake configure_file
- Installing generated QML file or including it in QRC
- Depending on complexity a big haslet
How to approach porting your app?

- Possibility to have separate kf6 branch
- Cleaner codebase due to less compatibility code
- Possibility to use newer features/cleaner API
- Risk of divergence between branches
How to approach porting your app?

- Plugin system one major challenge
- In KF6, runtime and buildtime json conversion removed
  - KPluginMetaData::fromDesktopFile
  - kcoreaddons_desktop_to_json
- desktoptojson CLI tool should be used for manual conversion
How to approach porting your app?

- API provider documentation or warnings quite important
  - Plugin namespace to be used
  - Version that change is compatible with
- After desktoptojson conversion, adjust plugin macro
  - K_PLUGIN_CLASS_WITH_JSON
  - K_PLUGIN_FACTORY_WITH_JSON
How to approach porting your app?

- Way of determining pluginId changed
- Previously, Id was often specified in embedded metadata
- In KF6, pluginId is derived from file basename
  - Id key only used in KPluginMetaData::fromJsonFile
- Warnings in case of mismatching id vs basename
- Derived Id is compatible with KF5
How to approach porting your app?

- Porting of KServiceTypeTrader is documented and was discussed last year
- Minor breaking changes for plugin providers
  - Keyword parameter of KPluginFactory::create removed
  - Relevant, in case of custom plugin factory/macro
- Some improvements to utilize!
How to approach porting your app?

- KPluginMetaData::findPluginById has better performance
  - PluginId can be used to find file directly on disk
  - Better interoperability with static plugins
  - Parameter for allowing empty metadata
- QDebug operator for easier logging
  - KPluginMetaData(pluginId:"qtplugin", fileName: "../qtplugin.so")
- Benefits from optimized KPluginFactory/KPluginMetaData internals
Thank you!

Porting BoF
Monday 17:00 EEST Room 2

KF6 BoF
Tuesday 16:00 EEST Room 1

Weekly meeting: Tuesday 17:00 CEST
at https://meet.kde.org/b/ada-mi8-aem
#kde-devel / kde-frameworks-devel@kde.org