



Categorized Logging in QML

Giuseppe D'Angelo KDAB



VOTE FOR ME



VOTE FOR THIS TALK



and win a cake! *

* Subject to availability





Logging in Qt



- Printing log messages in Qt has always been easy
 - qDebug() << Q_FUNC_INFO << widget->size();
 - qWarning() << "File not found" << file->name();
 - qFatal("Unable to initialize foobar: %d", errno);





qDebug issues



- qDebug and friends share some issues
 - Where do the messages come from?
 - Do I really need Q_FUNC_INFO?
 - How to categorize the messages?
 - How to enable/disable them at runtime?





New logging framework



- Qt 5.2 introduced a new logging framework
- Allows to define custom logging categories





New logging framework



- Q_DECLARE_LOGGING_CATEGORY(MY_CATEGORY)
- Q_LOGGING_CATEGORY(MY_CATEGORY, "com.kdab.talk")
- qCDebug(MY_CATEGORY) << "found" << count << "files";
- qCCritical(MY_CATEGORY) << "authentication failed!";





New logging framework



- Allows to control output at runtime
 - Logging rules in a file
 - Logging rules in an env variable
- API for filtering messages
- API for outputting messages









Would you like to know more?



Don't miss Kai Köhne's talk





What about QML?



- We can log from QML
- Methods on the global console object
 - console.log("the value is " + value);
 - console.warn("user not found");
- Get source code information thanks to V4





Logging in QML



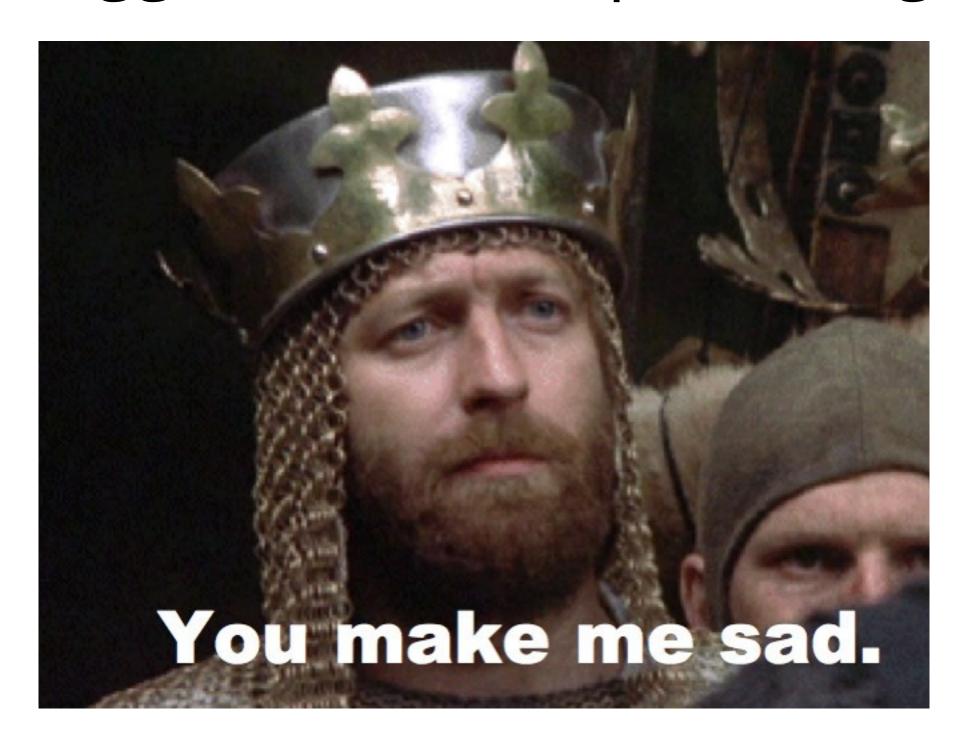
```
import QtQuick 2.2
Item {
    width: 400
    height: 400
    Component.onCompleted: { console.log("Hello"); }
$ qmlscene Logging.qml
[D] file:///home/peppe/k/devdays2014/qmllogging/examples/Logging.qml:7 - qml - Hello
```



Logging in QML



- How about categorized logging?
 - Not available out of the box
 - All messages logged into the "qml" category







Categorized Logging in QML



TAKE ONE







- Create a QObject subclass
- Give it Q_INVOKABLE methods to log (debug, warn...)
- In those methods: log something to a category
- Expose one or more of such objects to QML







Demo







- Hurray! The category is now correct!
- But... the source location of the messages is lost
- All messages now originated from within our C++ objects





Categorized Logging in QML



TAKE TWO







- Follow the source, Luke
- How is console actually implemented?
 - qtdeclarative/src/qml/qml/v8/qqmlbuiltinfunctions.cpp







Demo







- Meh...
- Too many private APIs used
 - "This is going to break anytime soon"





Categorized Logging in QML



TAKE THREE







- Get inspired by QV8Engine::getV4(QQmlEngine *)
 - As seen in the previous solution
- We can call it at any time and get the status of the engine
- So, just call it from a Q_INVOKABLE called from QML







Demo





Final touches



- Support multiple arguments passed to log(...)
- We still need to set logging objects as properties of the root QML context
 - Because we can't add properties to the global JS object
 - The global JS object is "frozen"





Final touches



- You can't add properties to the global JS object
- Did I say "you can't modify the global JS object"? Did I?







Final touches



- Override console's category!
 - It's possible! Whether it's expected to work, well...







Demo





Categorized Logging in QML



Thank you

