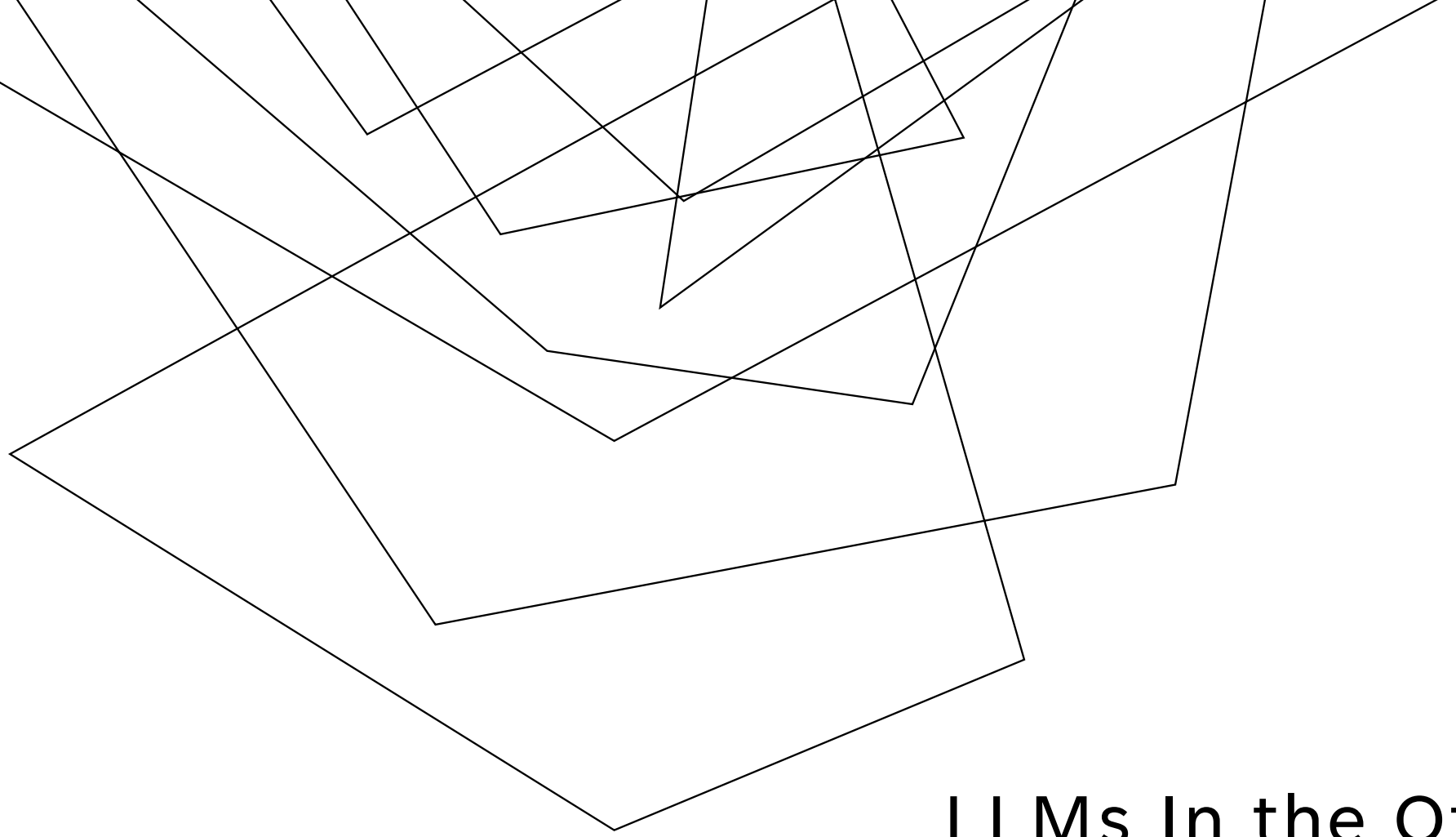


LLMs In the Qt Development Environment

LARGE LANGUAGE MODELS – A PRIMER

- LLMs are trained on a wide dataset, usable models are measured in Billions of parameters (7B, 30B, 100B...)
- Quantizing possible to lower bit precision, but trade off accuracy and reasoning
- Context size matters (and uses a lot of RAM).
  - GPT4 and Llama 3.1 support up to 128k tokens (approx. 96k words, fewer for code)
- Prompting instead of programming
  - Drive the model towards a goal, but don’t tell it exactly what it should do, or how to do it.
- Structured responses force better answers (JSON schemas)
- Field rapidly advancing, new models leapfrog old ones

USE CASE 1:
API CHANGE IDENTIFICATION

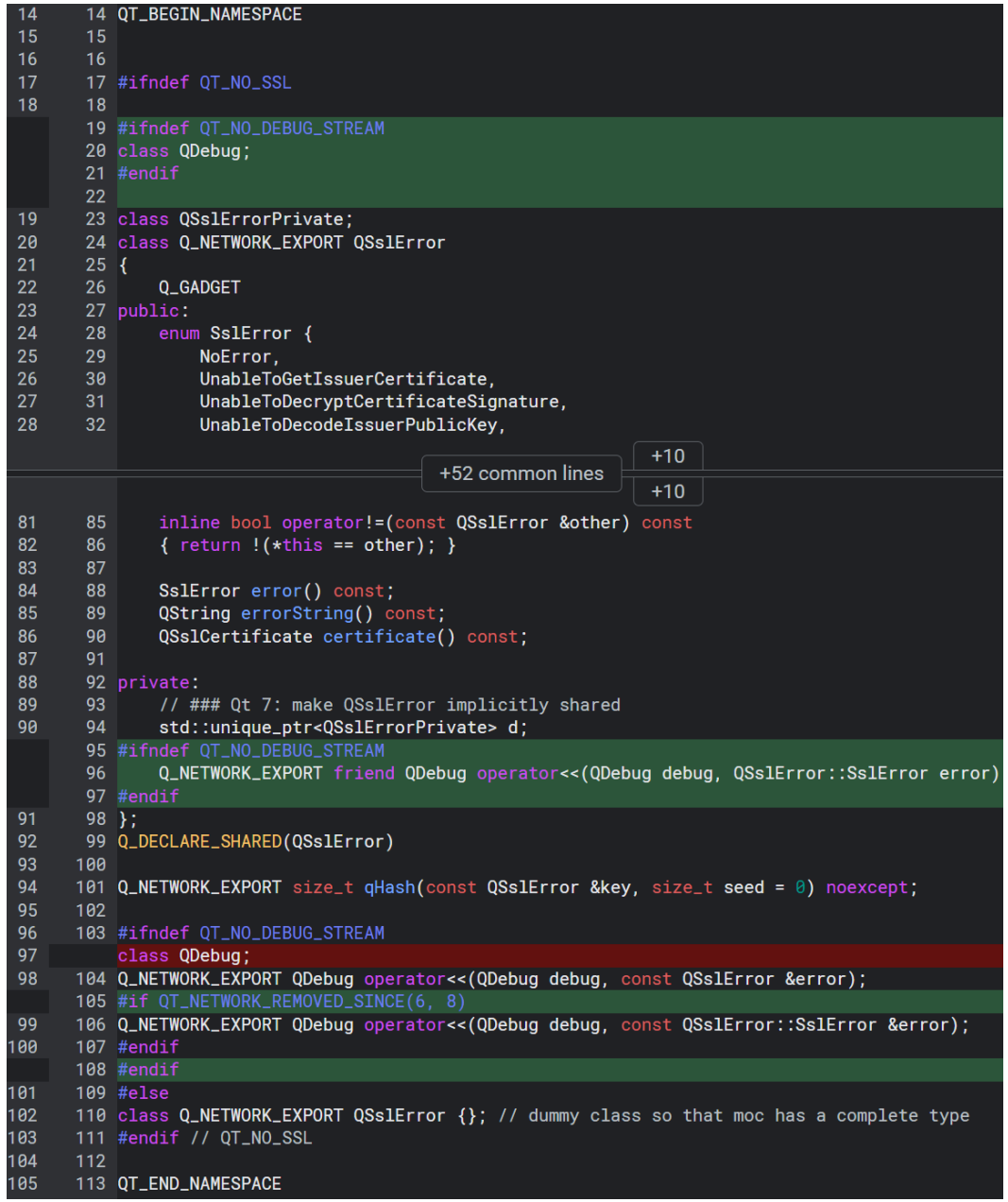
Identify changes which modify public APIs

Provides a first-line alert to catch API issues early

Bot process:

- Monitor changes to .h files (but filter known exclusions)
- Feed the diff into an LLM to generically scan for changes which are “significant to the behavior or usage of the API”
- Summarize changes and post a comment
- Add the “Needs API-Review_[version]” hashtag
  - Also add a “API Change cherry-picked” hashtag when the API changes will be backported.

API REVIEW SAMPLE:



```
14  14 QT_BEGIN_NAMESPACE
15  15
16  16
17  17 #ifndef QT_NO_SSL
18  18
    19 #ifndef QT_NO_DEBUG_STREAM
    20 class QDebug;
    21 #endif
    22
19  23 class QSslErrorPrivate;
20  24 class Q_NETWORK_EXPORT QSslError
21  25 {
22  26     Q_GADGET
23  27 public:
24  28     enum SslError {
25  29         NoError,
26  30         UnableToGetIssuerCertificate,
27  31         UnableToDecryptCertificateSignature,
28  32         UnableToDecodeIssuerPublicKey,

                +52 common lines   +10   +10

81  85     inline bool operator!=(const QSslError &other) const
82  86     { return !(*this == other); }
83  87
84  88     SslError error() const;
85  89     QString errorString() const;
86  90     QSslCertificate certificate() const;
87  91
88  92 private:
89  93     // ### Qt 7: make QSslError implicitly shared
90  94     std::unique_ptr<QSslErrorPrivate> d;
    95 #ifndef QT_NO_DEBUG_STREAM
    96     Q_NETWORK_EXPORT friend QDebug operator<<(QDebug debug, QSslError::SslError error)
    97 #endif
91  98 };
92  99 Q_DECLARE_SHARED(QSslError)
93 100
94 101 Q_NETWORK_EXPORT size_t qHash(const QSslError &key, size_t seed = 0) noexcept;
95 102
96 103 #ifndef QT_NO_DEBUG_STREAM
97     class QDebug;
98 104 Q_NETWORK_EXPORT QDebug operator<<(QDebug debug, const QSslError &error);
    105 #if QT_NETWORK_REMOVED_SINCE(6, 8)
99  106 Q_NETWORK_EXPORT QDebug operator<<(QDebug debug, const QSslError::SslError &error);
100 107 #endif
    108 #endif
101 109 #else
102 110 class Q_NETWORK_EXPORT QSslError {}; // dummy class so that moc has a complete type
103 111 #endif // QT_NO_SSL
104 112
105 113 QT_END_NAMESPACE
```

Summary:

This change has been identified as having significant changes to the public API:

- The QDebug operator<< is now a hidden friend of QSslError.
- Changes were made to how the QDebug operator<< matches arguments, affecting its usage in debugging contexts.
- Users are now required to make explicit conversions that were previously implicit, resulting in a source-incompatible change.

USE CASE 2:
CI FAILURE ANALYSIS

Identify when a change is responsible for the failure to integrate in CI

Bot process:

- Feed the last 1200 lines of the failure log into an LLM (Llama 3.1 8B)
  - 2 chunks of 600 lines due to local LLM 32k context limit
  - Extract errors with instruction about issues which are not critical errors
  - Extract filenames for failed sources or tests
- Retrieve relevant file sources from branch if not modified
- Retrieve change diff
- Run the error, sources, and diff through an LLM with large context (GPT 4o-mini)
  - Identify if the change is the likely cause of the failure, or not.

PATCHSET 1: Remove noexcept from copy member functions

```
22  22  class Q_GUI_EXPORT QPdfOutputIntent
23  23  {
24  24  public:
25  25      QPdfOutputIntent();
26          QPdfOutputIntent(const QPdfOutputIntent &other) noexcept;
        26  QPdfOutputIntent(const QPdfOutputIntent &other);
27  27      QPdfOutputIntent(QPdfOutputIntent &&other) noexcept = default;
28          QPdfOutputIntent &operator=(const QPdfOutputIntent &other) noexcept;
        28  QPdfOutputIntent &operator=(const QPdfOutputIntent &other);
29  29      QT_MOVE_ASSIGNMENT_OPERATOR_IMPL_VIA_PURE_SWAP(QPdfOutputIntent)
30  30      ~QPdfOutputIntent();
31  31
32  32      void swap(QPdfOutputIntent &other) noexcept { d.swap(other.d); }
```

Build error:

```
/Users/qt/work/qt/qtbase/src/gui/painting/qpdfoutputintent.cpp:57:19: error: exception specification in declaration does not match previous declaration
QPdfOutputIntent::QPdfOutputIntent(const QPdfOutputIntent &other) noexcept = default;
                  ^
/Users/qt/work/qt/qtbase/src/gui/painting/qpdfoutputintent.h:26:5: note: previous declaration is here
    QPdfOutputIntent(const QPdfOutputIntent &other);
    ^
/Users/qt/work/qt/qtbase/src/gui/painting/qpdfoutputintent.cpp:68:37: error: exception specification in declaration does not match previous declaration
QPdfOutputIntent &QPdfOutputIntent::operator=(const QPdfOutputIntent &other) noexcept = default;
                                    ^
/Users/qt/work/qt/qtbase/src/gui/painting/qpdfoutputintent.h:28:23: note: previous declaration is here
    QPdfOutputIntent &operator=(const QPdfOutputIntent &other);
                      ^
2 errors generated.
```

LLM ERROR ANALYSIS

Failure Summary:
High confidence that the change is related to the failure due to the removal of `noexcept` from the copy constructor and assignment operator, which caused inconsistency in exception specifications.

The error summary indicates that the `QPdfOutputIntent` class has inconsistent exception specifications in its copy constructor and assignment operator. The change removed `noexcept` from these functions, leading to a mismatch with their previous declarations.

Relevant files:
Filename: /Users/qt/work/qt/qtbase/src/gui/painting/qpdfoutputintent.h

```
class Q_GUI_EXPORT QPdfOutputIntent
{
public:
    QPdfOutputIntent();
    QPdfOutputIntent(const QPdfOutputIntent &other);
    QPdfOutputIntent(QPdfOutputIntent &&other) noexcept = default;
    QPdfOutputIntent &operator=(const QPdfOutputIntent &other);
    QT_MOVE_ASSIGNMENT_OPERATOR_IMPL_VIA_PURE_SWAP(QPdfOutputIntent)
    ~QPdfOutputIntent();
```

Suggested Action:
The change caused the failure. The `noexcept` specifier should be consistently applied or removed from both the declaration and definition of the copy constructor and assignment operator to resolve the inconsistency.

PATCHSET 2

M src/gui/painting/qpdfoutputintent.cpp

File File

+46 common lines +10

```
47  47  /*!
48  48      Constructs a new PDF output intent.
49  49  */
50  50  QPdfOutputIntent::QPdfOutputIntent()
51  51      : d(new QPdfOutputIntentPrivate)
52  52  {}
53  53
54  54  /*!
55  55      Constructs a copy of the output intent \a other.
56  56  */
    57  QPdfOutputIntent::QPdfOutputIntent(const QPdfOutputIntent &other) noexcept = default;
57      QPdfOutputIntent::QPdfOutputIntent(const QPdfOutputIntent &other) = default;
58  58
59  59  /*!
60  60      \fn QPdfOutputIntent::QPdfOutputIntent(QPdfOutputIntent &&other) noexcept
61  61
62  62      Constructs a QPdfOutputIntent object by moving from \a other.
63  63  */
64  64
65  65  /*!
66  66      Assigns the output intent \a other over this intent.
67  67  */
    68  QPdfOutputIntent &QPdfOutputIntent::operator=(const QPdfOutputIntent &other) noexcept = default;
68      QPdfOutputIntent &QPdfOutputIntent::operator=(const QPdfOutputIntent &other) = default;
69  69
70  70  /*!
```

LLM ERROR ANALYSIS – CHANGE NOT THE CAUSE

Failure Summary:
The failure is due to a known flaky test, `tst_QAudioSink::pullResumeFromUnderrun`.

Suggested Action:
The change is likely not the cause of the failure. Restaging the change may resolve the issue.

Failure Summary:
The change is not related to the failure. The test `tst_qstringapisymmetry` failed, but the change modifies `qplaintextedit.cpp`, which is unrelated to the `tst_qstringapisymmetry` test.

Suggested Action:
The change likely did not cause the failure. Restaging the change may resolve the issue.

Failure Summary:
The failure appears to be infrastructure-related, specifically due to a timeout and an issue with downloading a binary archive.

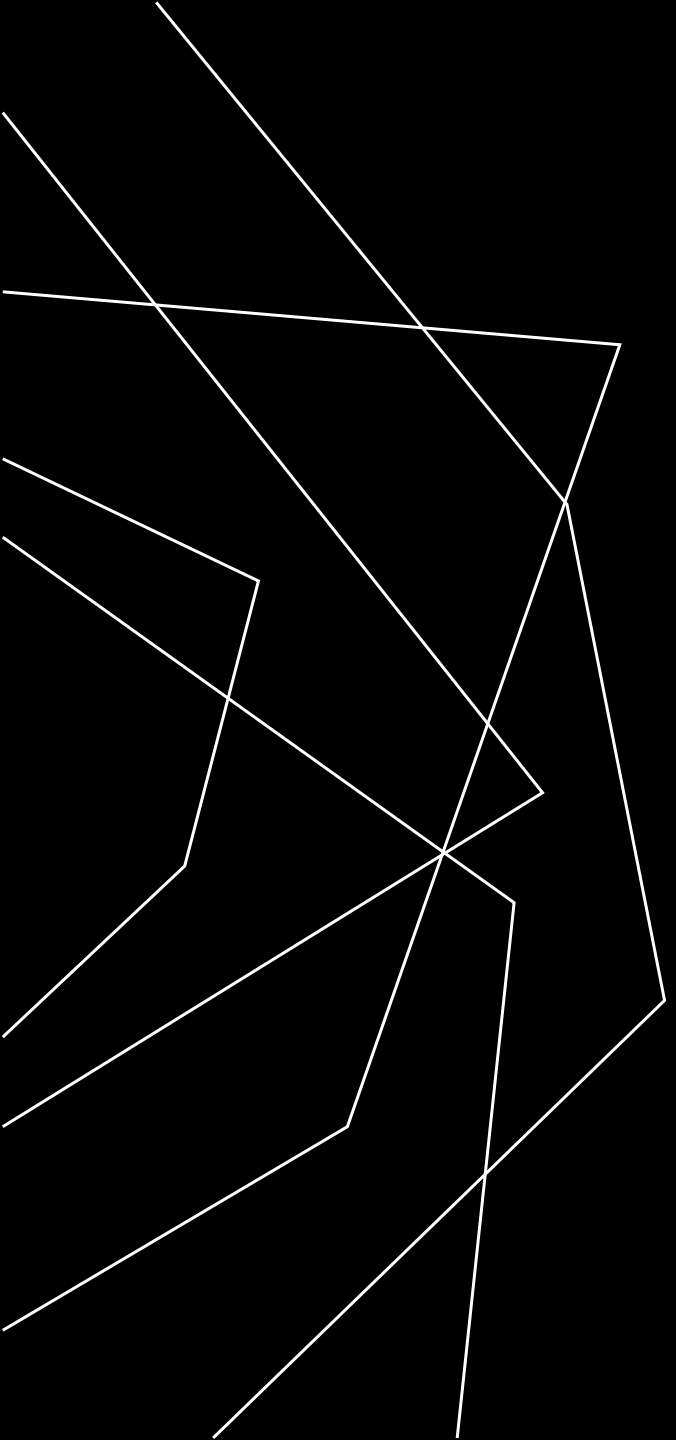
Suggested Action:
The change is unrelated to the failure. Restage the change to resolve the issue.

WHAT NEXT?

- Investigate Retrieval Augmented Generation
  - May be possible to have a database of up-to-date Qt Documentation for each branch.
  - RAG allows the LLM to retrieve authoritative source info on-demand, and automatically.
- Larger context is better
  - Avoiding chunking means better zero-shot analysis, fewer false positives.
- Find ways to give feedback via 👍and 👎buttons on bot comments.

NEW IDEAS?

- Must be non-intrusive
- Probably should not suggest specific changes
- Must be unlikely to have a high false-positive rate



THANK YOU

Daniel Smith

daniel.smith@qt.io