Solving a bug via lateral thinking

(or: How I solved a bug in 2 hours instead of 2 weeks)

Giuseppe D'Angelo

Senior Sofware Engineer, KDAB (UK)

CppCon 2017

Qt

KDAB

C++

~~C++~~

JavaScript

Javascript Pop Quiz

```
>>> console.log("2400000000000" == "3776798720")
```

Javascript Pop Quiz

```
>>> console.log("240000000000" == "3776798720")
false
```

Javascript Pop Quiz

```
>>> console.log("2400000000000" == "3776798720")
false


>>> console.log("2400000000000" === "3776798720")
false
```

JavaScript in Qt

- In Qt there is a JavaScript engine
- Actually, there are several…
  - V8 (in Chromium – QtWebEngine classes)
  - JSC (in WebKit – QtWebKit classes)
  - **V4 (in QML – QJSEngine classes)**
  - (maybe others)

When using V4:

```
import QtQuick 2.0
QtObject {
    Component.onCompleted: {
        console.log("240000000000" === "3776798720");
    }
}
```

```
$ qmlscene test.qml
true
```

“Yo, fix this mess!”

“But I know nothing about JavaScript!
(not to mention JavaScript engines)”

```cpp
QJSEngine engine;
QJSValue value("foo");

QJSValue obj = engine.newObject();
obj.setProperty("100", value);

QJSValueIterator it(obj);
while (it.hasNext()) {
    it.next();
    qDebug() << it.name() << it.value().toString();
}
// prints "100" "foo"
```

```cpp
QJSEngine engine;
QJSValue value("foo");

QJSValue obj = engine.newObject();
obj.setProperty("240000000000", value);

QJSValueIterator it(obj);
while (it.hasNext()) {
    it.next();
    qDebug() << it.name() << it.value().toString();
}
// prints "3776798720" "foo"
```

Debugging a modern JS engine?

- Not a trivial task
- A traditional debugger doesn't help
  - reinterpret_cast of raw memory everywhere
  - Tagged pointers nightmare
  - GDB actually crashed on me a couple of times

I'm lost

Could it be possible to find out where the execution diverges?

gcc -pg

gcc -pg

- *“Generate extra code to write profile information suitable for the analysis program* gprof.”
- Basically, annotates every function enter/exit, so that a profiler can take measurements
- (Similar: -finstrument-functions)

uftrace

uftrace

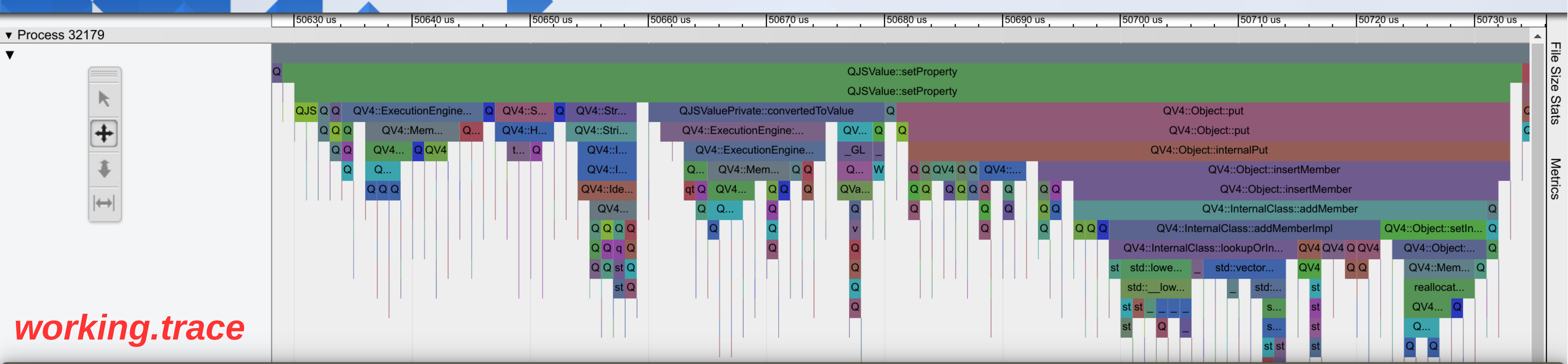
- *"A tool to trace and analyze the execution of a program written in C/C++ […] It traces each function in the executable and shows time duration"*
- Uses the hooks set in place by -pg
- https://github.com/namhyung/uftrace

```
$ uftrace tests/t-abc
# DURATION    TID     FUNCTION
  16.134 us [ 1892] | __monstartup();
 223.736 us [ 1892] | __cxa_atexit();
            [ 1892] | main() {
            [ 1892] |   a() {
            [ 1892] |     b() {
            [ 1892] |       c() {
   2.579 us [ 1892] |         getpid();
   3.739 us [ 1892] |       } /* c */
   4.376 us [ 1892] |     } /* b */
   4.962 us [ 1892] |   } /* a */
   5.769 us [ 1892] | } /* main */
```

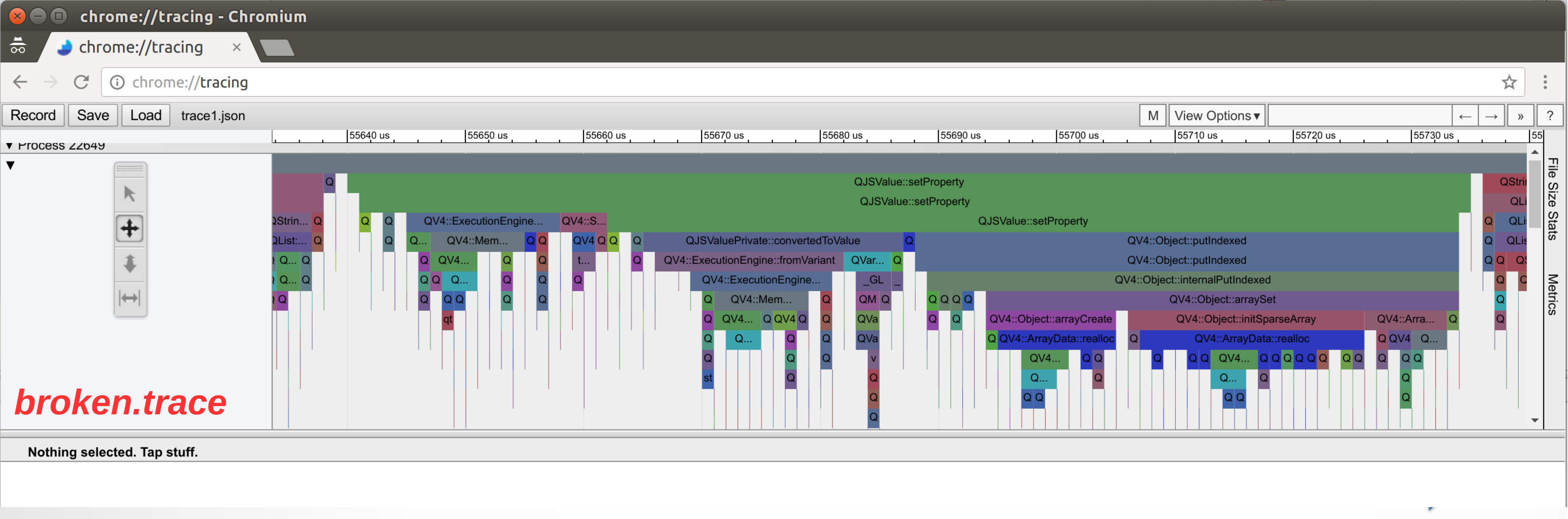
uftrace dump --chrome

uftrace dump --chrome

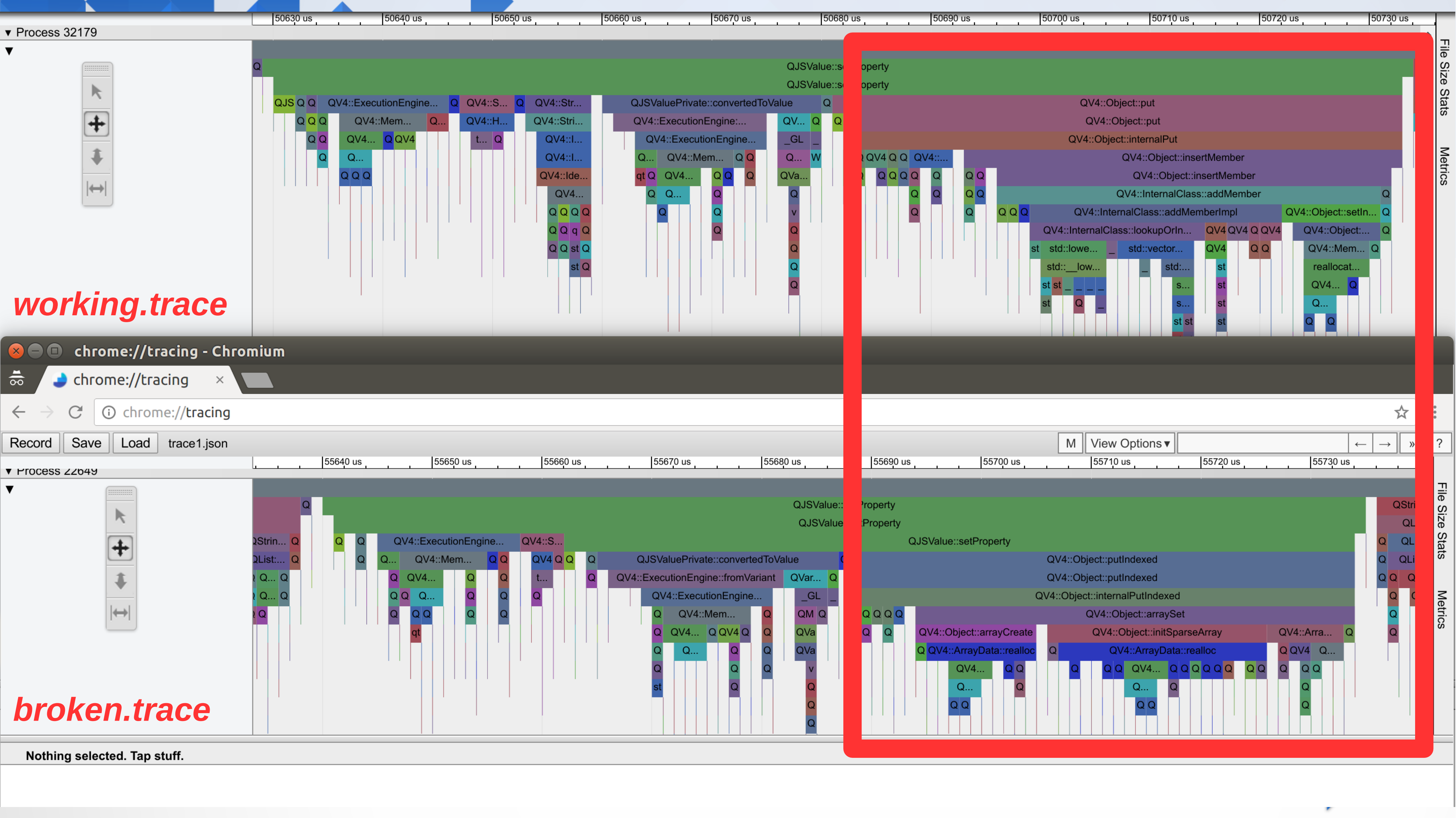
- Dumps the profiling information as a flamegraph in Chrome's trace format
- Load the flamegraph in Chrome/Chromium
  - chrome://tracing
- Play with trimming options until you see...



working.trace



broken.trace


working.trace
broken.trace
chrome://tracing - Chromium
Record
Save
Load
trace1.json
View Options
Process 32179
Process 22649
QJSValue::setProperty
QV4::Object::put
QV4::Object::internalPut
QV4::Object::insertMember
QV4::InternalClass::addMember
QV4::InternalClass::addMemberImpl
QV4::InternalClass::lookupOrIn...
QV4::Object::putIndexed
QV4::Object::internalPutIndexed
QV4::Object::arraySet
QV4::Object::arrayCreate
QV4::Object::initSparseArray
QV4::ArrayData::realloc
QJSValuePrivate::convertedToValue
QV4::ExecutionEngine::fromVariant
Nothing selected. Tap stuff.

Found the divergence

```
ScopedString s(scope, engine->newString(name));
uint idx = s->asArrayIndex();
if (idx < UINT_MAX) {
    setProperty(idx, value); // taken in the broken case
    return;
}
s->makeIdentifier(scope.engine);
QV4::ScopedValue v(scope,
    QJSValuePrivate::convertedToValue(engine, value));
o->put(s, v); // taken in the working case
```

Ultimately, it was a broken detection of unsigned integer multiplication overflow

(when trying to convert the property name from a string to an integer)

☺

Thank you

giuseppe.dangelo@kdab.com

More info about uftrace:

“Understanding the runtime behaviors of C++ programs using uftrace tool”

Friday, September 29 • 10:30am