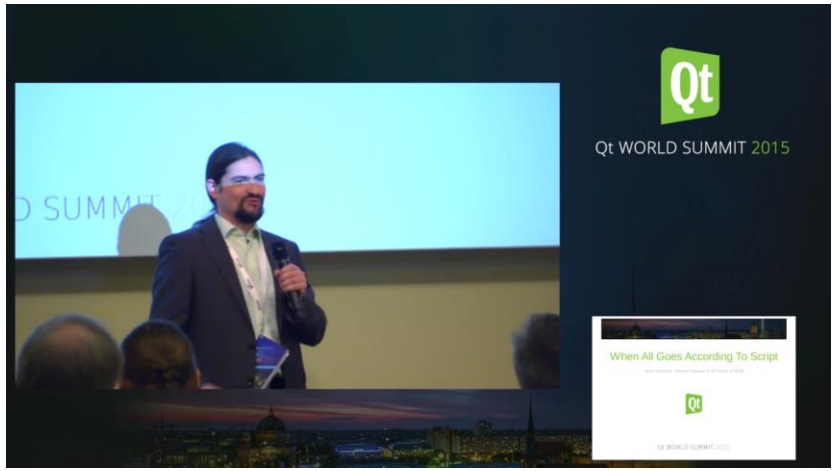
Script automation
Recording user actions

Nicolas Arnaud-Cormos
nicolas.arnaud-cormos@kdab.com

Scripting Qt application



[When all goes according to script]

QtWS 2015

Kevin Krammer, KDAB

[Practical application scripting with QML]

QtWS 2019

Kevin Krammer, KDAB

KDAB

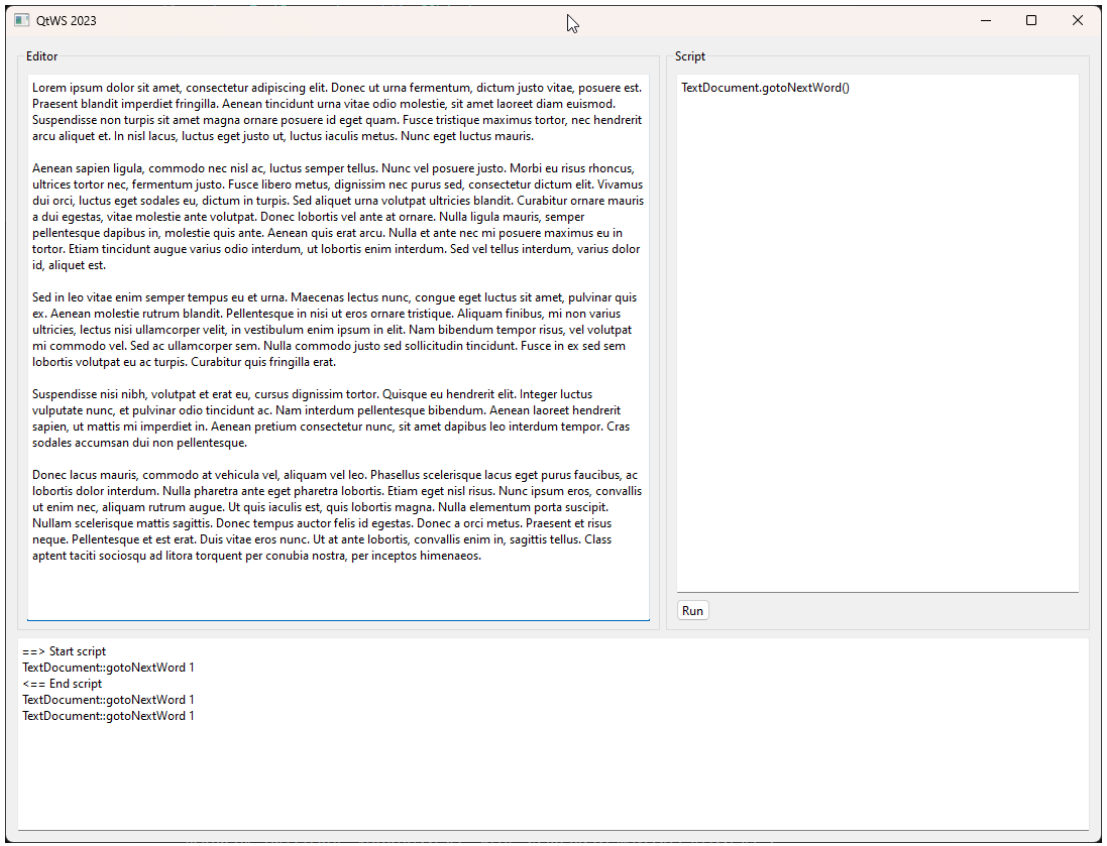
You got your application scriptable!

And now...

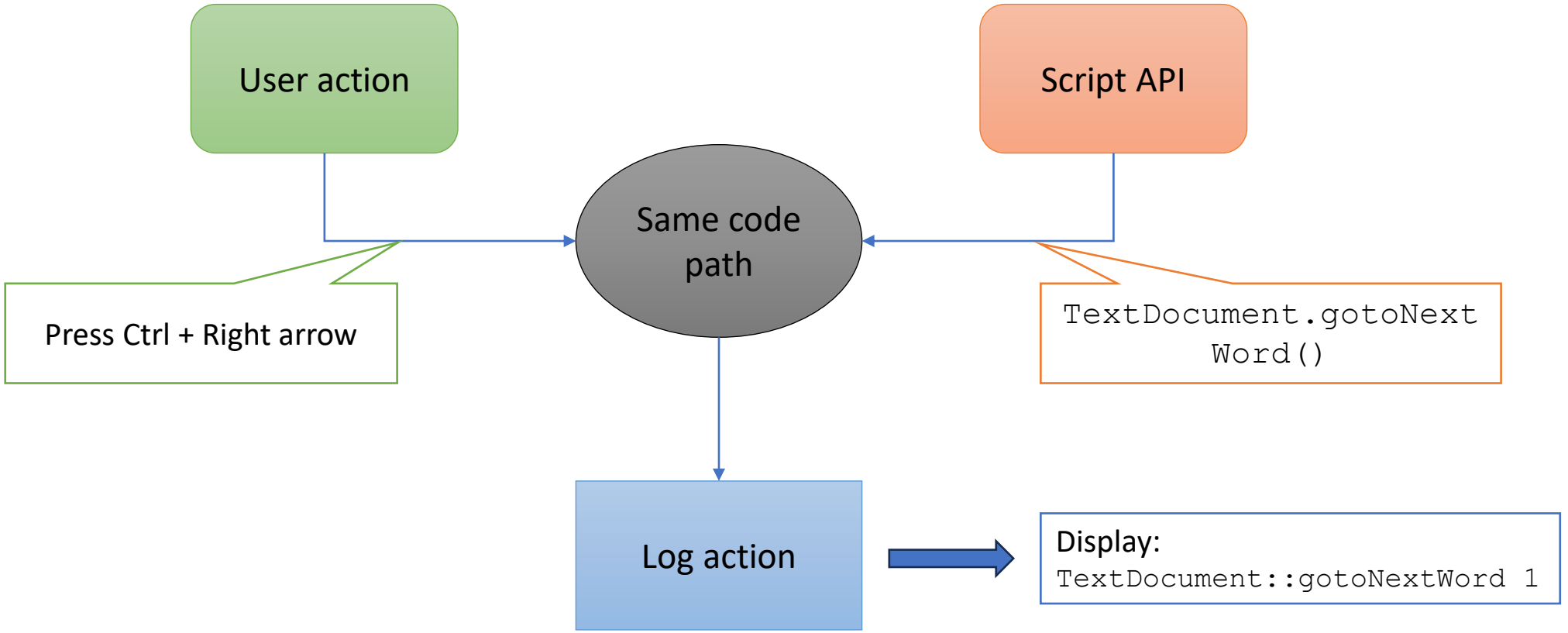
How to create a script from user actions ?

The setup

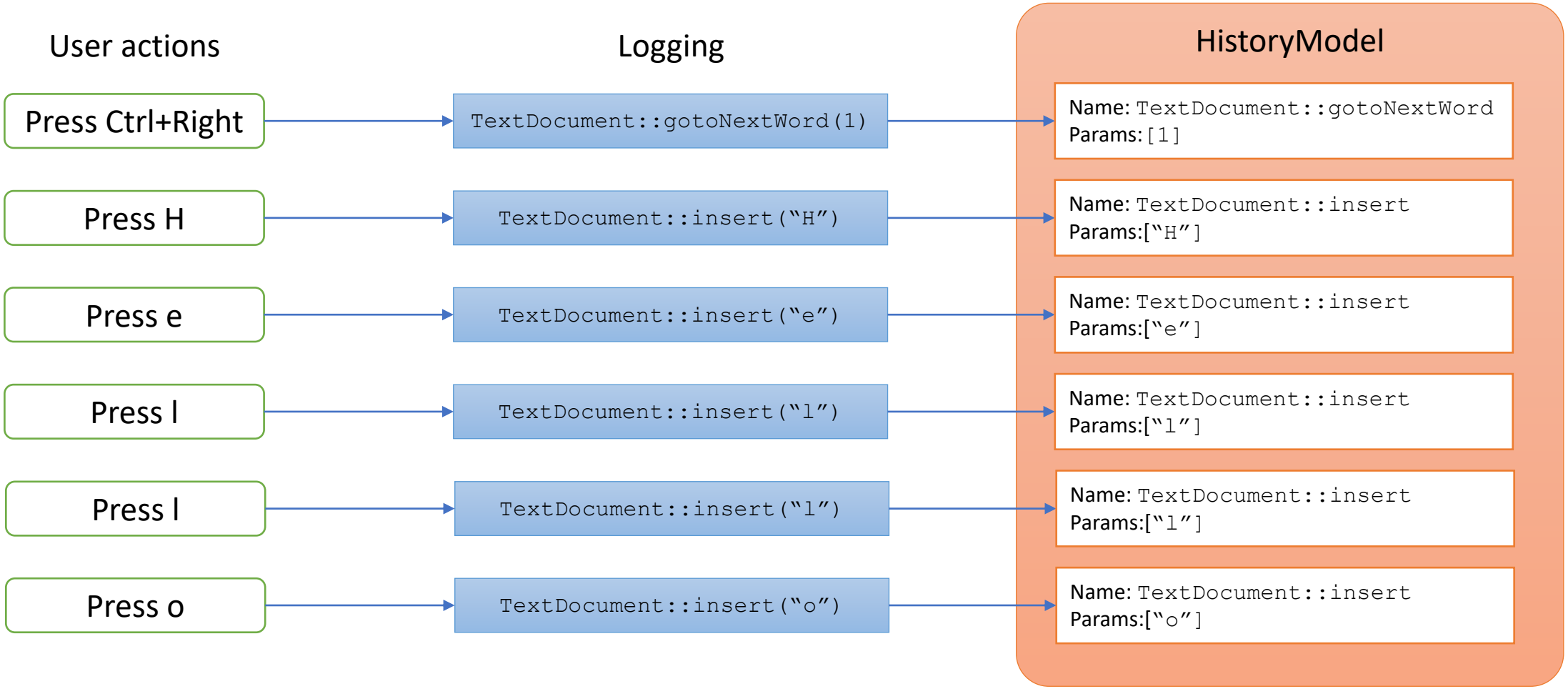
- Simple text editor with scripting ability.
  - Code available here: https://github.com/KDABLabs/qtws23-script-automation



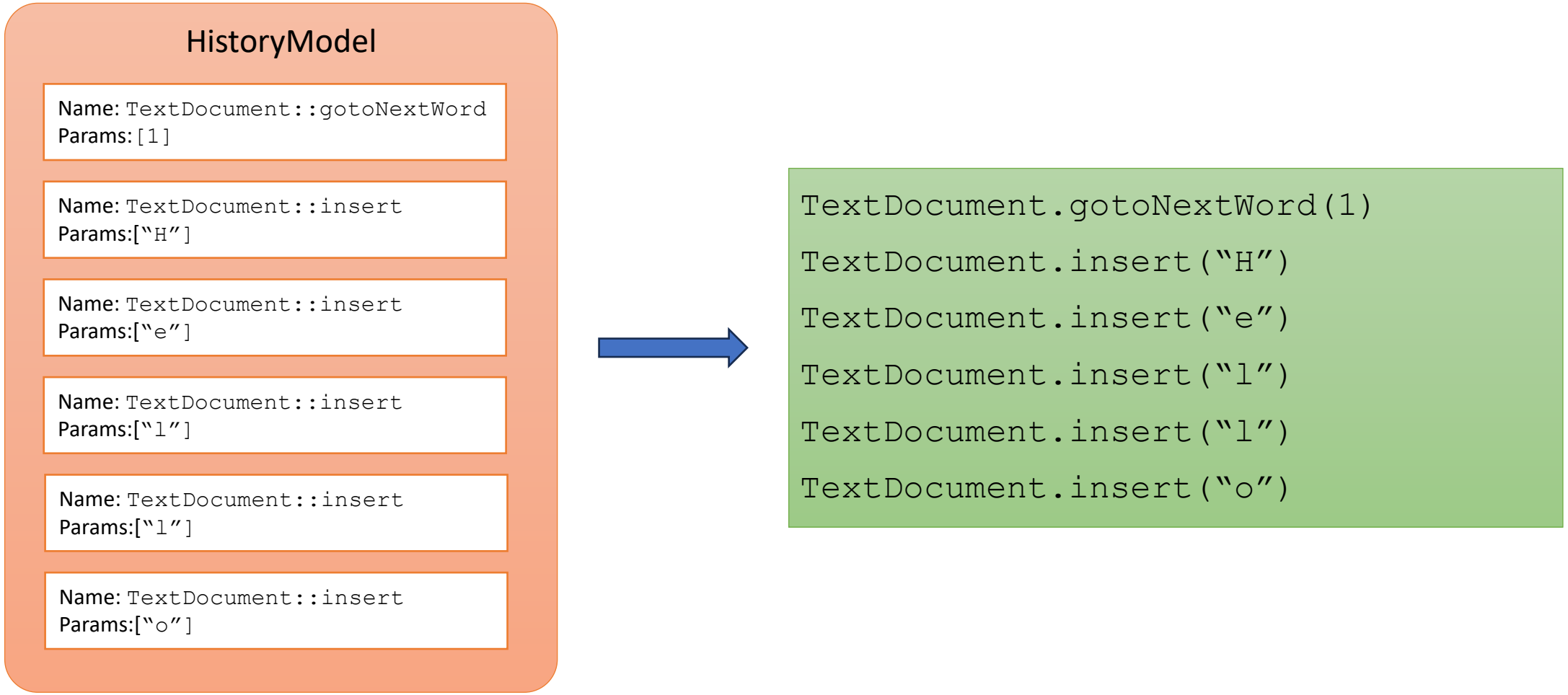
Code preparation



Use and abuse logging system



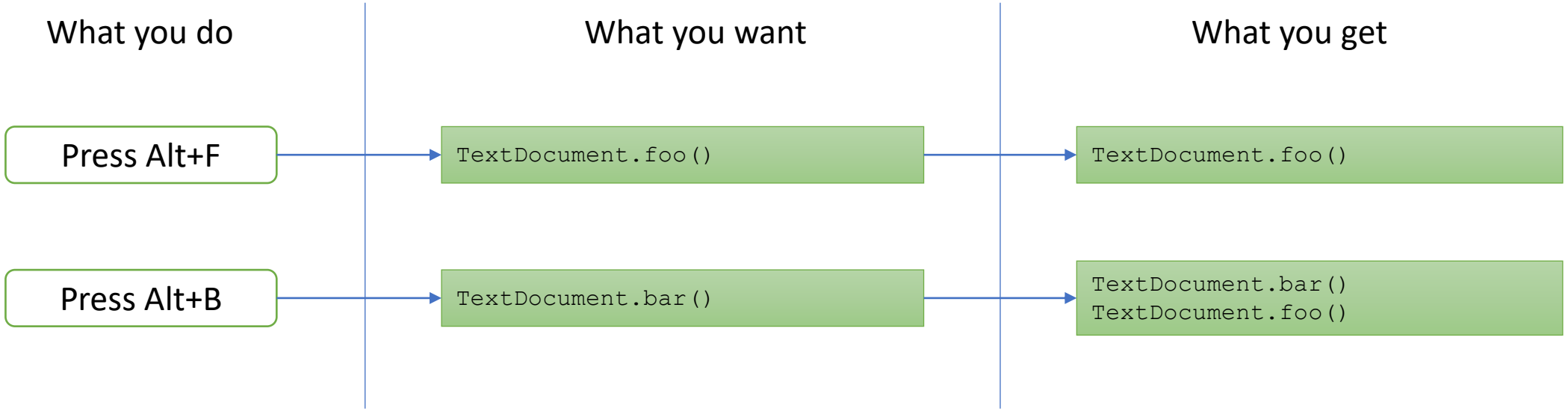
Create script from history



Don’t log everything – API called by an API

Example

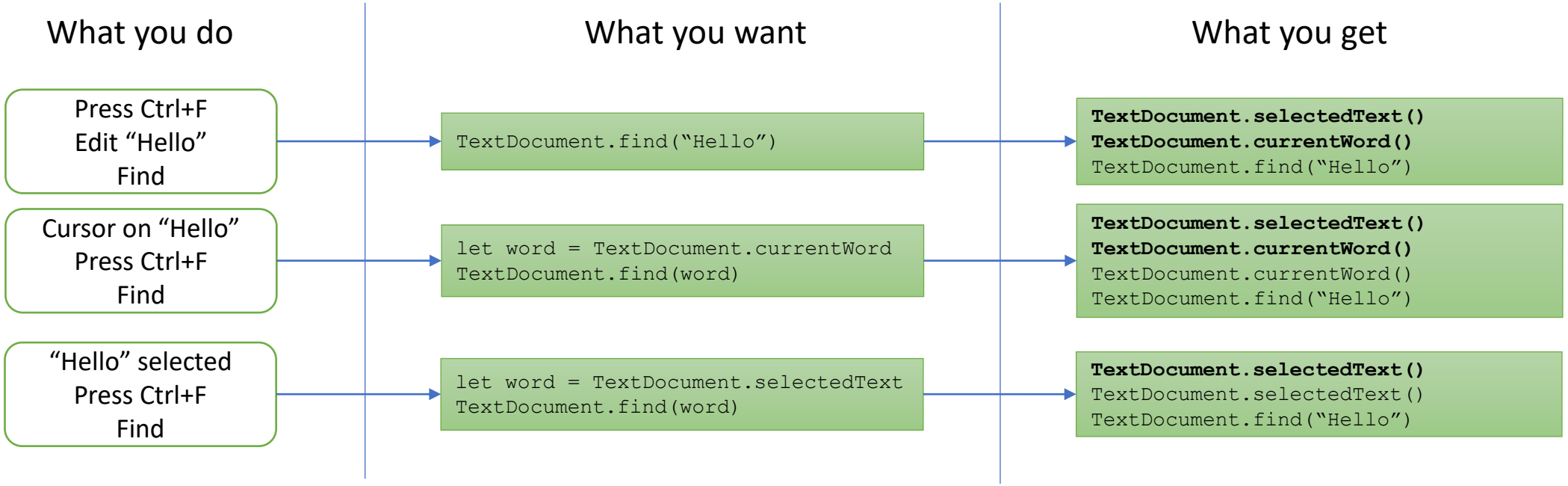
- one API `TextDocument::foo` (Alt+F)
- one API `TextDocument::bar` (Alt+B) calling `TextDocument::foo`



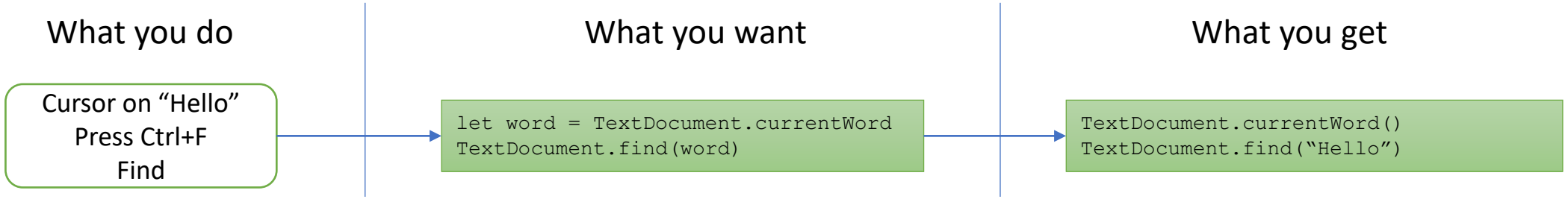
Don’t log everything – API used internally

Example: Find feature

- Call `TextDocument::selectedText` and if empty `TextDocument::currentWord` to fill the line edit
- On find, if the user has not changed the text from the line edit, re-call the previous API

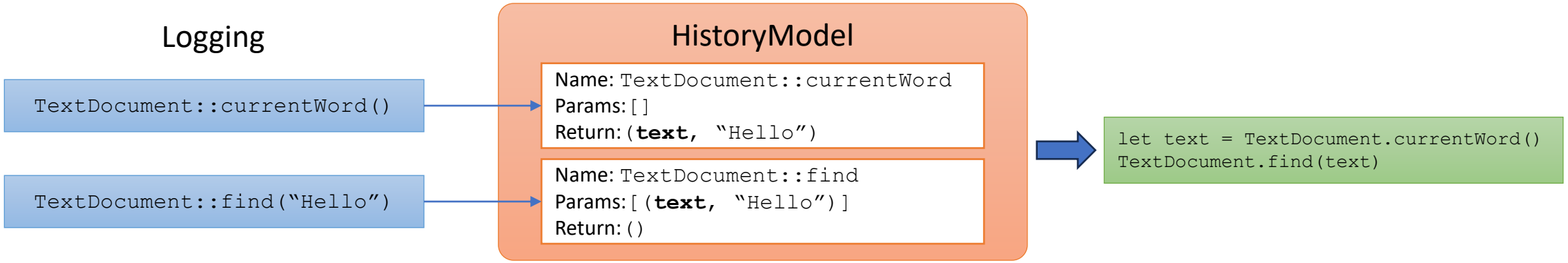


Use return values in script



To fix the issue, we need to do different steps

1. Store the return value from the API
2. Tag return values and parameters to know which ones are compatible
3. Check previous return value and parameter value during script creation
   - If they are equal, use a variable to pass it



Final touch: handling properties

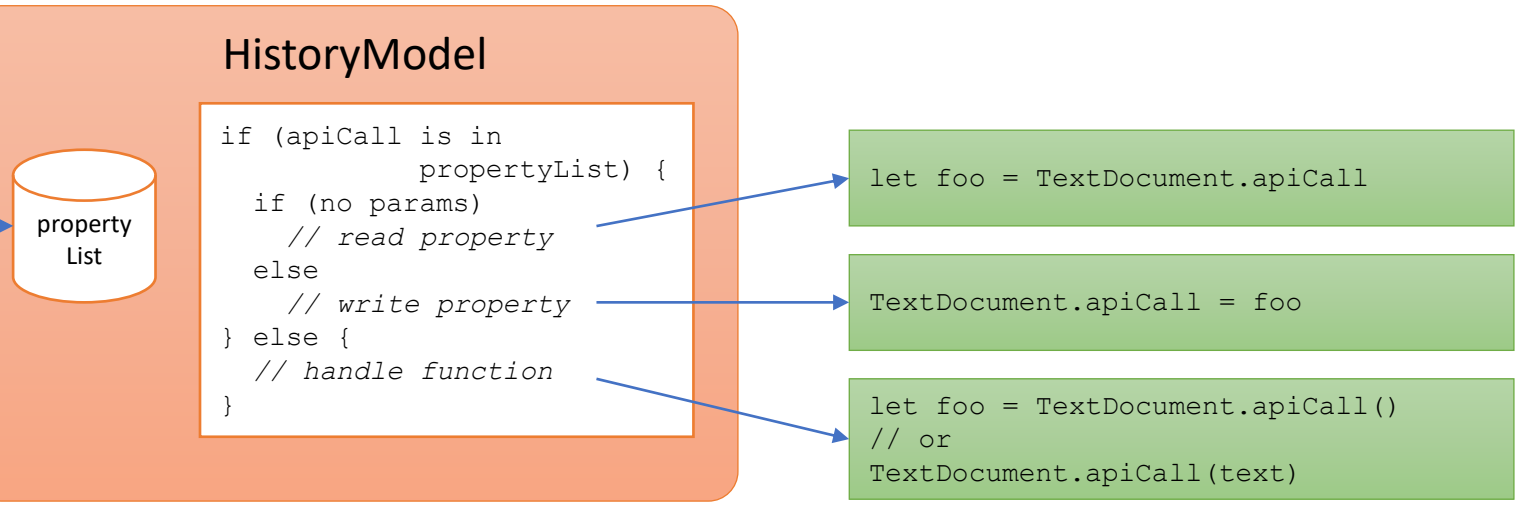
What you do	What you want	What you get
Cursor on “Hello” Press Ctrl+F Find	`let word = TextDocument.currentWord` `TextDocument.find(word)`	`let word = TextDocument.currentWord()` `TextDocument.find(word)`

To fix the issue, we need to do different steps

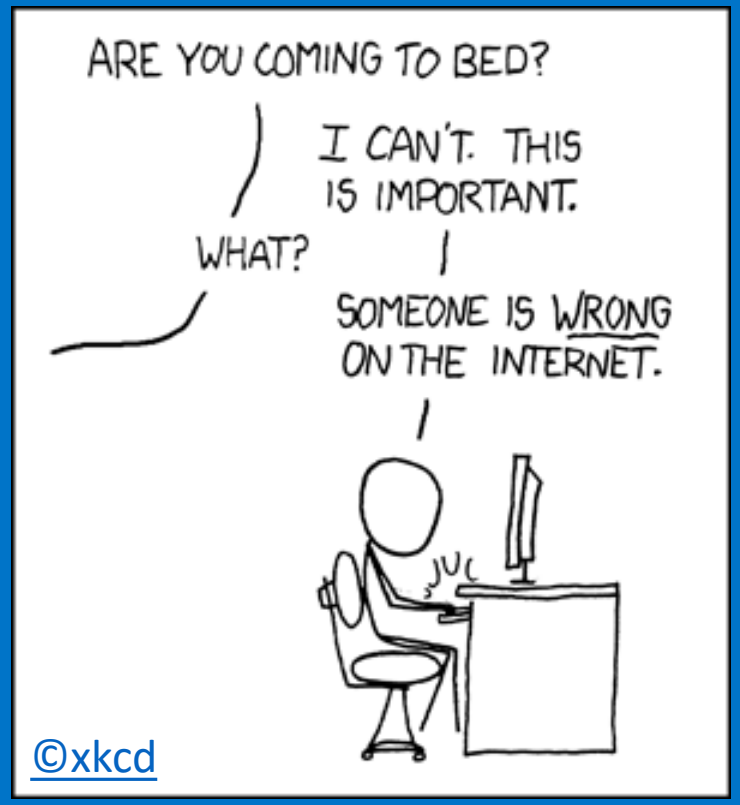
1. Store all properties for all QML objects
2. During script creation, check if the API is a property or not

Code

```
TextDocument::TextDocument(…)
{
  LOG_REGISTER(TextDocument);
```



When you think you are done...



There are always more!

- Merge calls that could be merged

```
TextDocument.insert("H")
TextDocument.insert("e")
TextDocument.insert("l")
TextDocument.insert("l")
TextDocument.insert("o")
```

```
TextDocument.insert("Hello")
```

As an exercise for the developer

- Handle complex parameters
- Handle QML singletons and non-singletons
- Simplify the LOG macro (using `std::source_location`)
- Store properties at compilation time
- …

KDAB

Follow us

Thank you!

Get the code