# Writing VS Code extensions for fun and profit

NIKOLAJ FOGH

# About me

# About me

## Background

# About me

## Background

- Electronics engineering / process control

# About me

## Background

- Electronics engineering / process control
- Embedded C

# About me

## Background

- Electronics engineering / process control
- Embedded C
- 10+ years working with C++

# About me

## Background

- Electronics engineering / process control
- Embedded C
- 10+ years working with C++
- Recently mostly frameworks / tooling

# About me

## Background

- Electronics engineering / process control
- Embedded C
- 10+ years working with C++
- Recently mostly frameworks / tooling
- Currently works for Siemens Gamesa doing wind turbine control software

# About me

## Background

- Electronics engineering / process control
- Embedded C
- 10+ years working with C++
- Recently mostly frameworks / tooling
- Currently works for Siemens Gamesa doing wind turbine control software

nikolajfogh@gmail.com

# Contents

# Contents

1. Why bring this up at a C++ conference?

# Contents

1. Why bring this up at a C++ conference?
2. IDE extensions

# Contents

1. Why bring this up at a C++ conference?
2. IDE extensions
3. Why VS Code

# Contents

# Contents

1. Why bring this up at a C++ conference?
2. IDE extensions
3. Why VS Code
4. Hello World!
5. Quality-of-life improvements

# Contents

# Why bring this up at a C++ conference?

This has nothing to do with C++

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks
  - Configuring builds

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks
  - Configuring builds
  - Configuring debugging

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks
  - Configuring builds
  - Configuring debugging
  - Tedious refactoring tasks

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks
  - Configuring builds
  - Configuring debugging
  - Tedious refactoring tasks
  - Boilerplate code

# Why bring this up at a C++ conference?

This has nothing to do with C++

- A lot of time is spent on non-coding tasks
  - Configuring builds
  - Configuring debugging
  - Tedious refactoring tasks
  - Boilerplate code
- Often done by automation tools

# Why bring this up at a C++ conference?

This has nothing to do with C++

Tooling is important

# Why bring this up at a C++ conference?

This has nothing to do with C++

Tooling is important

- Anastasia Kazakova: C++ Painkillers for C++ developers

# Why bring this up at a C++ conference?

This has nothing to do with C++

Tooling is important

- Anastasia Kazakova: C++ Painkillers for C++ developers
- This talk is very much about C++ painkillers

# Why bring this up at a C++ conference?

This has nothing to do with C++

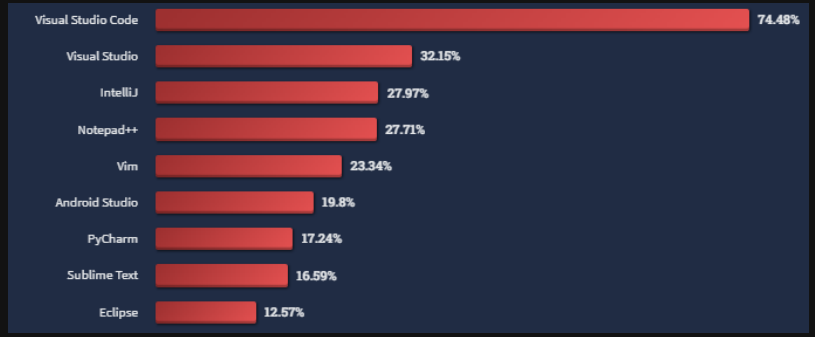
Tooling is important

- Anastasia Kazakova: C++ Painkillers for C++ developers
- This talk is very much about C++ painkillers
- Allows your developers to focus on their main task

# Why VS Code

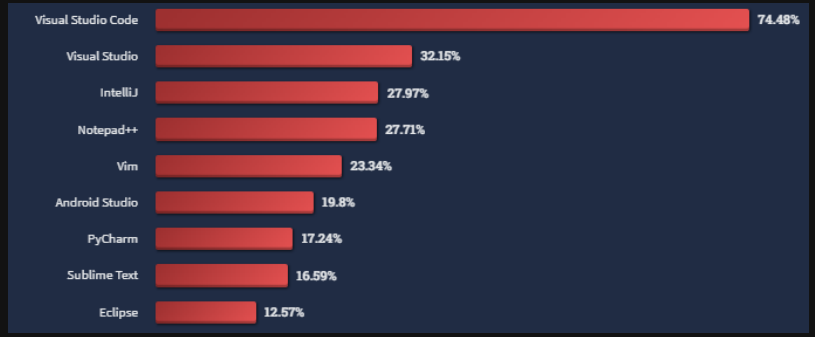
# Why VS Code

- Top of "desired" IDE in a recent stack overflow poll



# Why VS Code

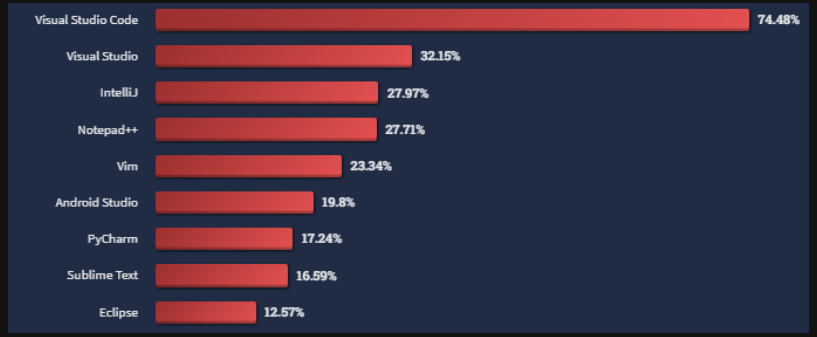
- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.

# Why VS Code

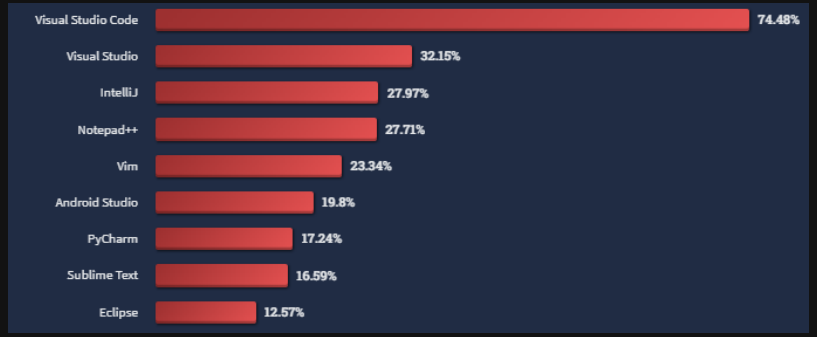
- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.
- Very configurable. Loads of extensions available to add functionality.

# Why VS Code

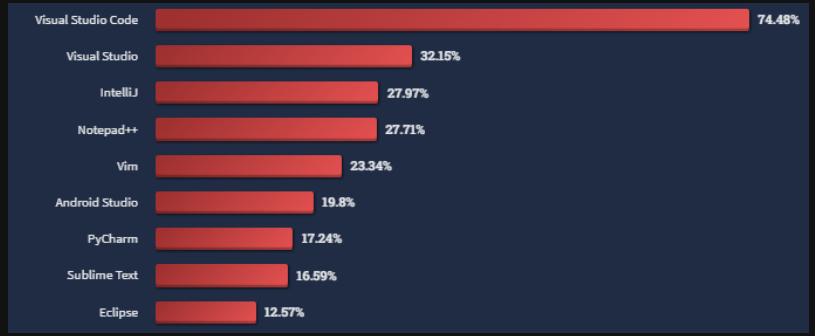
- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.
- Very configurable. Loads of extensions available to add functionality.
- Using extensions, VS Code can be made into an IDE.

# Why VS Code

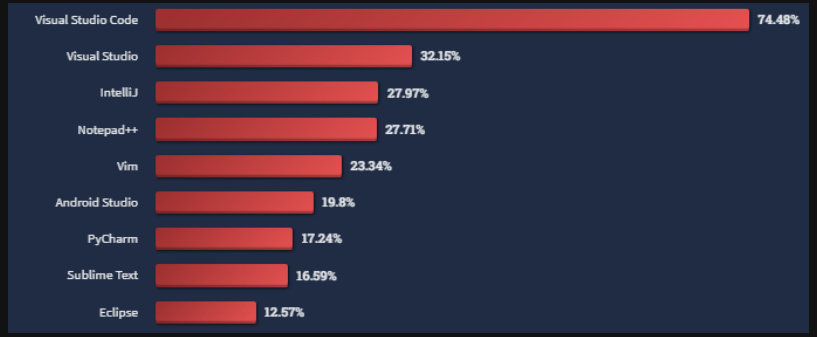
- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.
- Very configurable. Loads of extensions available to add functionality.
- Using extensions, VS Code can be made into an IDE.
- Familiar interface for multiple programming languages.

# Why VS Code

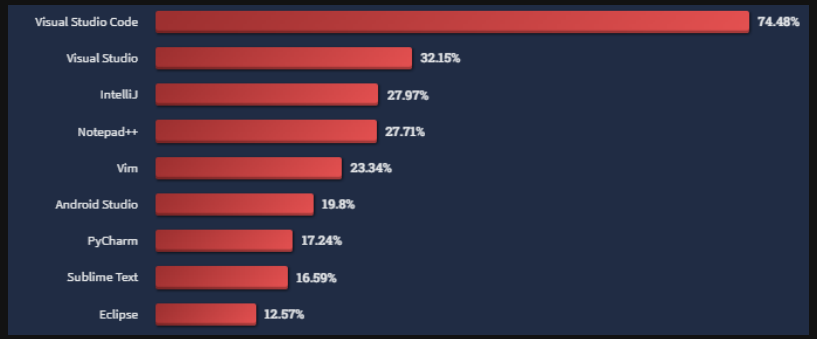
- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.
- Very configurable. Loads of extensions available to add functionality.
- Using extensions, VS Code can be made into an IDE.
- Familiar interface for multiple programming languages.
- Extension API fairly well documented (Samples: https://github.com/microsoft/vscode-extension-samples)

# Why VS Code

- Top of "desired" IDE in a recent stack overflow poll




- Relatively lightweight.
- Very configurable. Loads of extensions available to add functionality.
- Using extensions, VS Code can be made into an IDE.
- Familiar interface for multiple programming languages.
- Extension API fairly well documented (Samples: https://github.com/microsoft/vscode-extension-samples)
- Takeaways from this talk not specific to VS Code.

# Why write extensions for VS Code?

What is in it for me?

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools
- Replace legacy IDEs

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools
- Replace legacy IDEs
- Automate repetitive tasks

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools
- Replace legacy IDEs
- Automate repetitive tasks

## Other benefits

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools
- Replace legacy IDEs
- Automate repetitive tasks

## Other benefits

- Learn another programming language & ecosystem (typescript / npm)

# Why write extensions for VS Code?

What is in it for me?

- Integrate IDE with custom automation tools
- Replace legacy IDEs
- Automate repetitive tasks

## Other benefits

- Learn another programming language & ecosystem (typescript / npm)
- Have fun!

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

- Install node.js and npm

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

- Install node.js and npm
- Generate scaffolding

```
npm install -g yo generator-code
yo code
```

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

- Install node.js and npm
- Generate scaffolding

```
npm install -g yo generator-code
yo code
```

- Answer the questions asked

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

- Install node.js and npm
- Generate scaffolding

```
npm install -g yo generator-code
yo code
```

- Answer the questions asked
- Open src/extension.ts

# Let’s try to create a hello world extension

Just follow the guide here: https://code.visualstudio.com/api/get-started/your-first-extension

- Install node.js and npm
- Generate scaffolding

```
npm install -g yo generator-code
yo code
```

- Answer the questions asked
- Open src/extension.ts
- Press F5 to start an extension debugging session

# Let’s make a useful command

# Let’s make a useful command

Learning a new language?

# Let’s make a useful command

Learning a new language?

You need 2 things:

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# ChatGPT can be both of those things

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# ChatGPT can be both of those things

- Has deep knowledge about TypeScript

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# ChatGPT can be both of those things

- Has deep knowledge about TypeScript
- Has knowledge about the vscode API

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# ChatGPT can be both of those things

- Has deep knowledge about TypeScript
- Has knowledge about the vscode API
- You can ask it to elaborate on solutions

# Let’s make a useful command

Learning a new language?

You need 2 things:

1. A fun project
2. Code examples and a mentor

# ChatGPT can be both of those things

- Has deep knowledge about TypeScript
- Has knowledge about the vscode API
- You can ask it to elaborate on solutions
- This advice comes with all the caveats against ChatGPT

# Let’s make a useful command

Automate repetitive tasks

## Text-editor commands

- Advanced search-and-replace
- Autocomplete include paths

```
#include "lib1.h"
#include "lib2.h"
int main()
{

}
```

# Let’s make a useful command

Automate repetitive tasks

## Text-editor commands

- Advanced search-and-replace
- Autocomplete include paths

```
#include "Component1/shared/lib1.h"
#include "Component2/shared/lib2.h"
int main()
{

}
```

# Let’s make a useful command

Automate repetitive tasks

## Text-editor commands

- Advanced search-and-replace
- Autocomplete include paths

```
#include "Component1/shared/lib1.h"
#include "Component2/shared/lib2.h"
int main()
{

}
```

## Live demo

Wasn’t that fun?

# A case study

Replacing legacy IDEs

# A case study

Replacing legacy IDEs

- We were stuck with a legacy IDE + an in-house built build-system

# A case study

Replacing legacy IDEs

- We were stuck with a legacy IDE + an in-house built build-system
- Integration between the two was very poor

# A case study

Replacing legacy IDEs

- We were stuck with a legacy IDE + an in-house built build-system
- Integration between the two was very poor
- Debugging effectively non-existing

# A case study

Replacing legacy IDEs

- We were stuck with a legacy IDE + an in-house built build-system
- Integration between the two was very poor
- Debugging effectively non-existing
- A surprising amount of developers make do with whats easily available

# A case study

Replacing legacy IDEs

# A case study

## Replacing legacy IDEs

- We started playing a bit around customizing VS Code

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component
- We wanted 1-click debugging

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component
- We wanted 1-click debugging
  - Autodetect current component

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component
- We wanted 1-click debugging
  - Autodetect current component
  - Autodetect shared library paths

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component
- We wanted 1-click debugging
  - Autodetect current component
  - Autodetect shared library paths
  - Automatic upload of component + dependencies to target

# A case study

Replacing legacy IDEs

- We started playing a bit around customizing VS Code
  - Task configurations (https://code.visualstudio.com/docs/editor/tasks)
  - Launch configurations (https://code.visualstudio.com/docs/cpp/launch-json-reference)
- We got pretty far but were still limited
- Task and launch configurations are pretty static
  - Suitable for 1-2 components. Not 50-100
- We wanted various quality-of-life improvements
- We wanted 1-click builds of current component
- We wanted 1-click debugging
  - Autodetect current component
  - Autodetect shared library paths
  - Automatic upload of component + dependencies to target
  - Automatic debugging invocation

# IDE extensions as a painkiller

A case study

# IDE extensions as a painkiller

A case study

- Did we succeed?

# IDE extensions as a painkiller

A case study

- Did we succeed?
  - Yes

# IDE extensions as a painkiller

A case study

- Did we succeed?
  - Yes
- Was it hard to do?

# IDE extensions as a painkiller

A case study

- Did we succeed?
  - Yes
- Was it hard to do?
  - No

# IDE extensions as a painkiller

A case study

- Did we succeed?
  - Yes
- Was it hard to do?
  - No
  - It was FUN

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

- Works well for trivial cases

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

- Works well for trivial cases
- Not flexible enough for more advanced cases

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

- Works well for trivial cases
- Not flexible enough for more advanced cases
- More advanced cases needs

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

- Works well for trivial cases
- Not flexible enough for more advanced cases
- More advanced cases needs
  - Task providers

# Customizing launch.json and tasks.json

Resolving files and paths dynamically

## Demo time!

- Works well for trivial cases
- Not flexible enough for more advanced cases
- More advanced cases needs
  - Task providers
  - Debug configuration providers

# Legacy build systems

# Legacy build systems

- Introducing terribuild

# Legacy build systems

- Introducing terribuild
- Tries to emulate legacy custom build system

# Legacy build systems

- Introducing terribuild
- Tries to emulate legacy custom build system
- Configuration based on JSON

# Legacy build systems

- Introducing terribuild
- Tries to emulate legacy custom build system
- Configuration based on JSON
- Command-line based

# Legacy build systems

- Introducing terribuild
- Tries to emulate legacy custom build system
- Configuration based on JSON
- Command-line based

## Demo: Let's try to integrate it into vs code

# Closing remarks

# Closing remarks

- Writing extensions for VS Code is easy and fun

# Closing remarks

- Writing extensions for VS Code is easy and fun
- I have only tried writing extensions for VS Code

# Closing remarks

- Writing extensions for VS Code is easy and fun
- I have only tried writing extensions for VS Code
- Haven’t tried many other IDEs (mainly Eclipse and Visual Studio)

# Closing remarks

- Writing extensions for VS Code is easy and fun
- I have only tried writing extensions for VS Code
- Haven’t tried many other IDEs (mainly Eclipse and Visual Studio)
- Eclipse Theia is compatible with VS Code extensions

# Closing remarks

- Writing extensions for VS Code is easy and fun
- I have only tried writing extensions for VS Code
- Haven’t tried many other IDEs (mainly Eclipse and Visual Studio)
- Eclipse Theia is compatible with VS Code extensions
- One common extension API for all editors would be great

Thank you