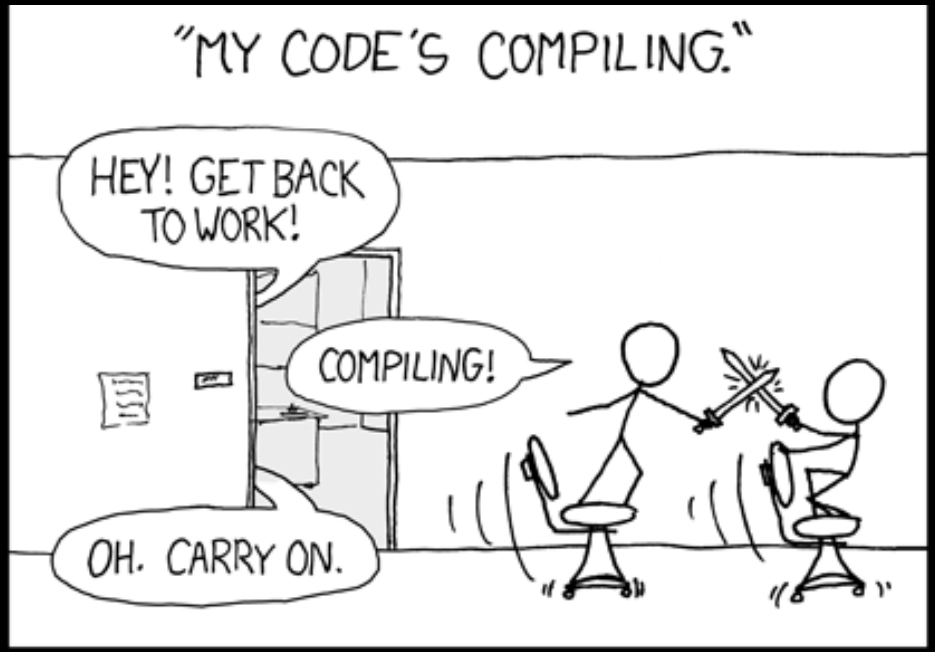
CPPINCLUDE

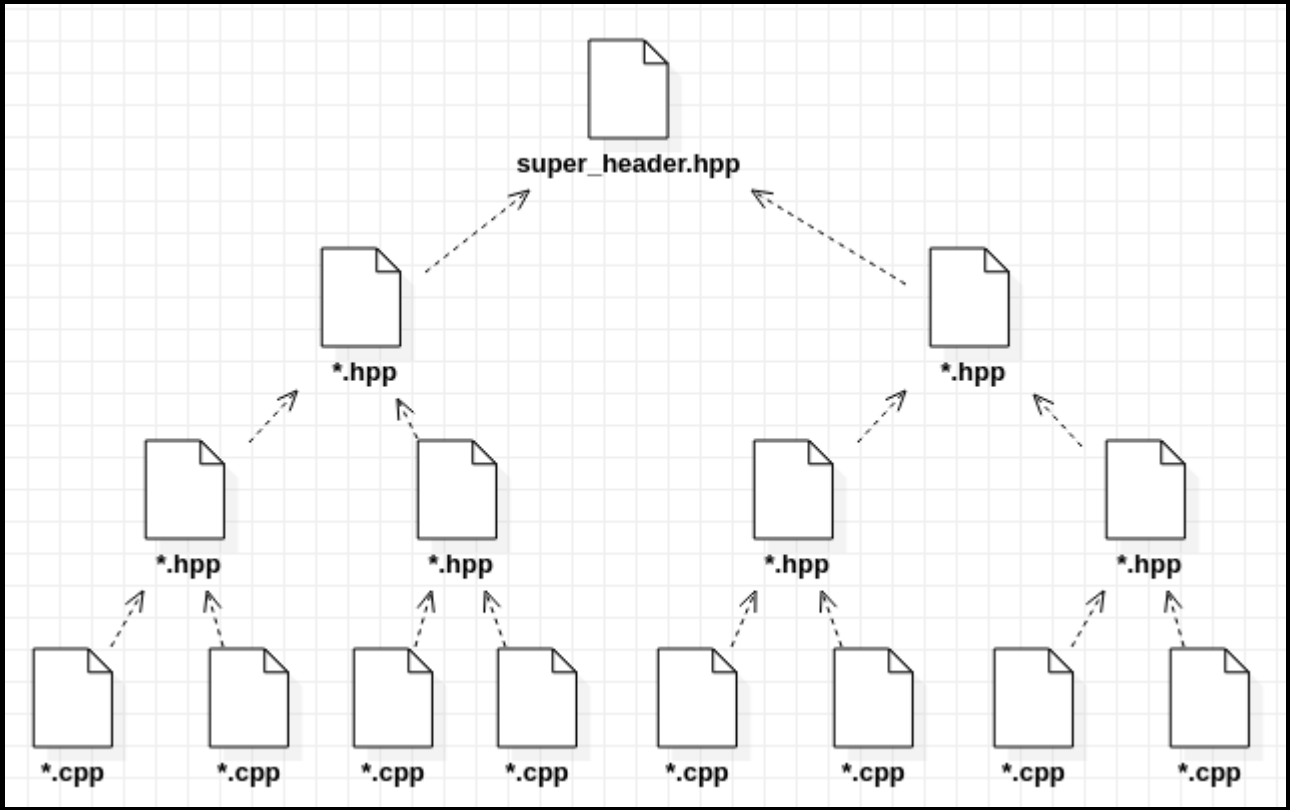
Lightning Talks CppCon 2020

KEEP CALM and COMPILE

> touch big_header_that_are_included_in_all_files.hpp



Why does it happen?



Tips for optimization includes

- Fix wrong includes
- Use forward declaration
- Split big header
- Pimpl (Pointer to implementation)

IWYU(Include What You Use)

https://github.com/include-what-you-use/include-what-you-use

Include What You Use

build passing

For more in-depth documentation, see docs.

Instructions for Users

"Include what you use" means this: for every symbol (type, function, variable, or macro) that you use in `foo.cc` (or `foo.cpp`), either `foo.cc` or `foo.h` should include a .h file that exports the declaration of that symbol. (Similarly, for `foo_test.cc`, either `foo_test.cc` or `foo.h` should do the including.) Obviously symbols defined in `foo.cc` itself are excluded from this requirement.

This puts us in a state where every file includes the headers it needs to declare the symbols that it uses. When every file includes what it uses, then it is possible to edit any file and remove unused headers, without fear of accidentally breaking the upwards dependencies of that file. It also becomes easy to automatically track and update dependencies in the source code.

CAVEAT

This is alpha quality software -- at best (as of July 2018). It was originally written to work specifically in the Google source tree, and may make assumptions, or have gaps, that are immediately and embarrassingly evident in other types of code.

While we work to get IWYU quality up, we will be stinting new features, and will prioritize reported bugs along with the many existing, known bugs. The best chance of getting a problem fixed is to submit a patch that fixes it (along with a test case that verifies the fix)!

IWYU. Hello World

```
#include <iostream>
#include <vector>

int main() {
  std::cout << "Hello World!" << std::endl;
  return 0;
}
```

[50%] Building CXX object CMakeFiles/hello.dir/main.cc.o
Warning: include-what-you-use reported diagnostics:

/home/user/hello/main.cc should add these lines:

/home/user/hello/main.cc should remove these lines:
- #include <vector> // lines 2-2

cppinclude

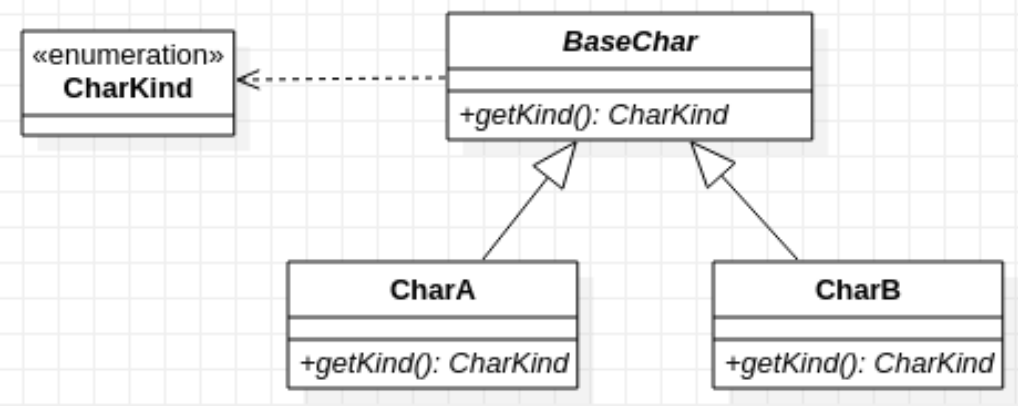
https://github.com/cppinclude/cppinclude

CPPINCLUDE

Tool for analyzing includes in C++. One of the problem in C++ is that if header file was changed all files that include the file will be recompiled and sometime it takes a lot of time.

Table of Contents

Simple example. Objects



https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *CharKind*

char_kind.hpp

```
#pragma once

enum class CharKind
{
    A,
    B,

    Count
};
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *BaseChar*

base_char.hpp

```
#pragma once

#include "char_kind.hpp"

class BaseChar
{
public:

    virtual ~BaseChar() = default;

    virtual CharKind getKind() const noexcept = 0;
};
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *CharA*

char_a.hpp

```cpp
#pragma once

#include "base_char.hpp"

class CharA : public BaseChar
{
public:

    CharKind getKind() const noexcept override;
};
```

char_a.cpp

```cpp
#include "char_a.hpp"

CharKind CharA::getKind() const noexcept
{
    return CharKind::A;
}
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *CharB*

char_b.hpp

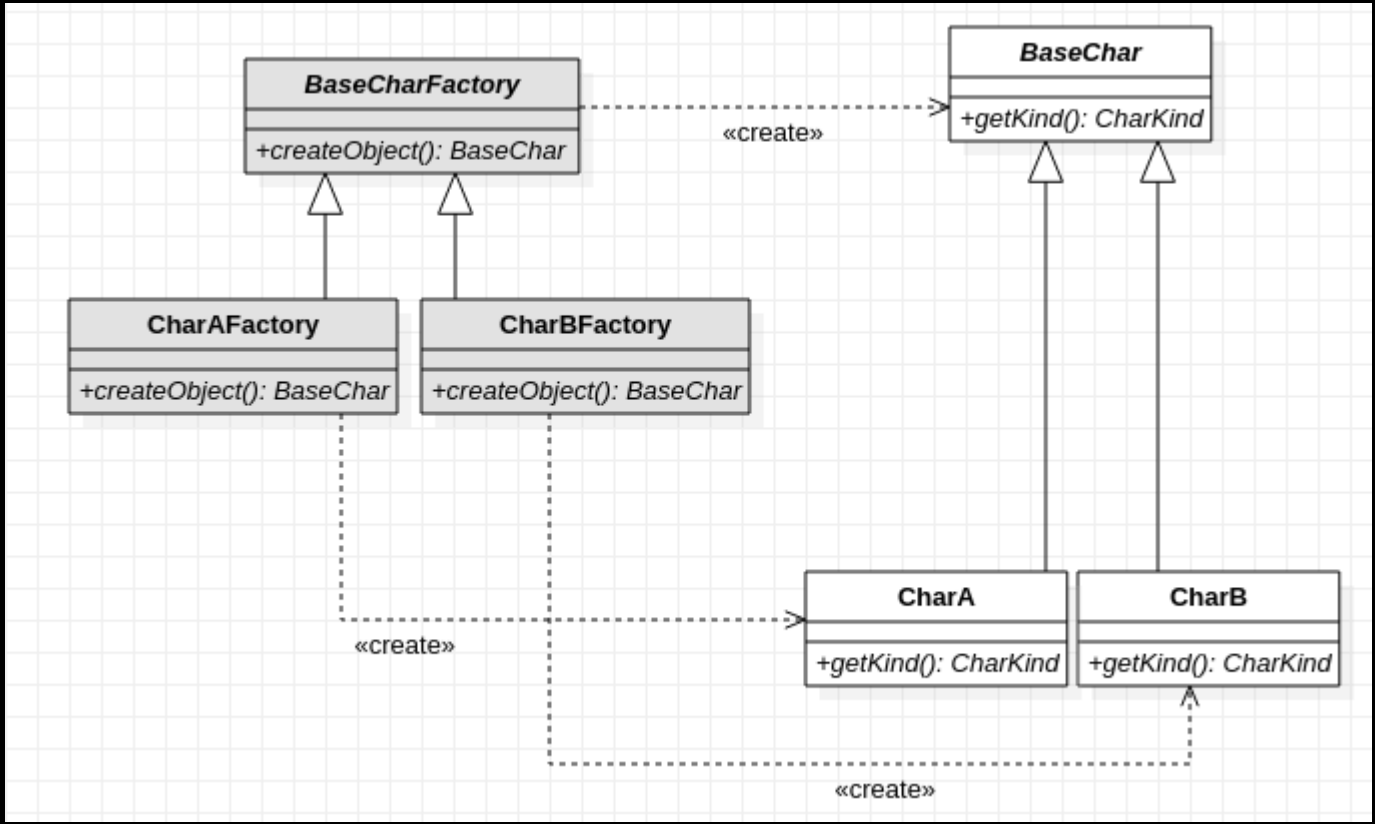
```
#pragma once

#include "base_char.hpp"

class CharB : public BaseChar
{
public:

    CharKind getKind() const noexcept override;
};
```

char_b.cpp

```
#include "char_b.hpp"

CharKind CharB::getKind() const noexcept
{
    return CharKind::B;
}
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *Factories*



https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *BaseCharFactory*

base_char_factory.hpp <Select Symbol>

```cpp
#pragma once

#include "base_char.hpp"

#include <memory>

class BaseCharFactory
{
public:

    virtual ~BaseCharFactory() = default;

    virtual std::unique_ptr< BaseChar > createObject() = 0;
};
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *CharAFactory*

char_a_factory.hpp

```
#pragma once

#include "base_char_factory.hpp"

class CharAFactory : public BaseCharFactory
{
public:

    std::unique_ptr< BaseChar > createObject() override;
};
```

char_a_factory.cpp

```
#include "char_a_factory.hpp"

#include "char_a.hpp"

std::unique_ptr< BaseChar > CharAFactory::createObject()
{
    return std::unique_ptr< BaseChar >{ new CharA };
}
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *CharBFactory*

char_b_factory.hpp <Select Symbol>

```
#pragma once

#include "base_char_factory.hpp"

class CharBFactory : public BaseCharFactory
{
public:

    std::unique_ptr< BaseChar > createObject() override;
};
```

char_b_factory.cpp <Select Symbol>

```
#include "char_b_factory.hpp"

#include "char_b.hpp"

std::unique_ptr< BaseChar > CharBFactory::createObject()
{
    return std::unique_ptr< BaseChar >{ new CharB };
}
```

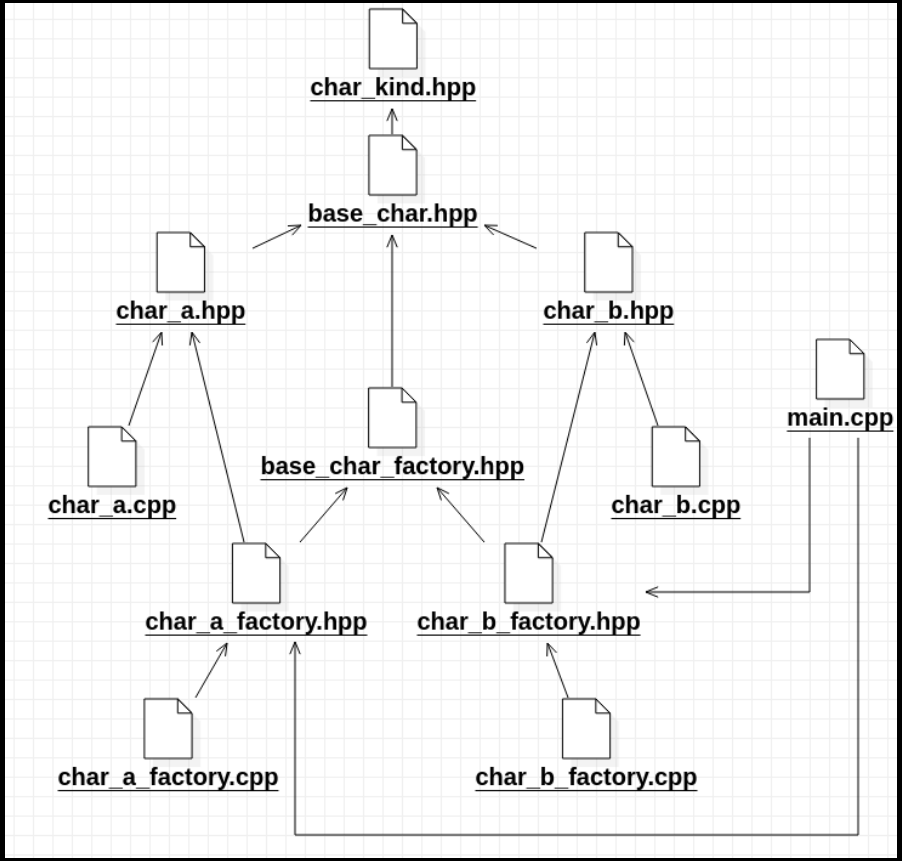
https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. *main*

```
#include "char_a_factory.hpp"
#include "char_b_factory.hpp"

#include <iostream>
#include <cassert>

std::unique_ptr< BaseChar > createChar( BaseCharFactory & _factory );
std::string enumToString( CharKind _kind );
int main()
{
    CharAFactory factoryA;
    CharBFactory factoryB;

    char c;
    std::cin >> c;

    std::unique_ptr< BaseChar > currentObject;
    if( c == 'a' )
        currentObject = createChar( factoryA );
    else
        currentObject = createChar( factoryB );
    assert( currentObject );

    std::cout << "Enum : " << enumToString( currentObject->getKind() ) << '\n';

    return 0;
}
```

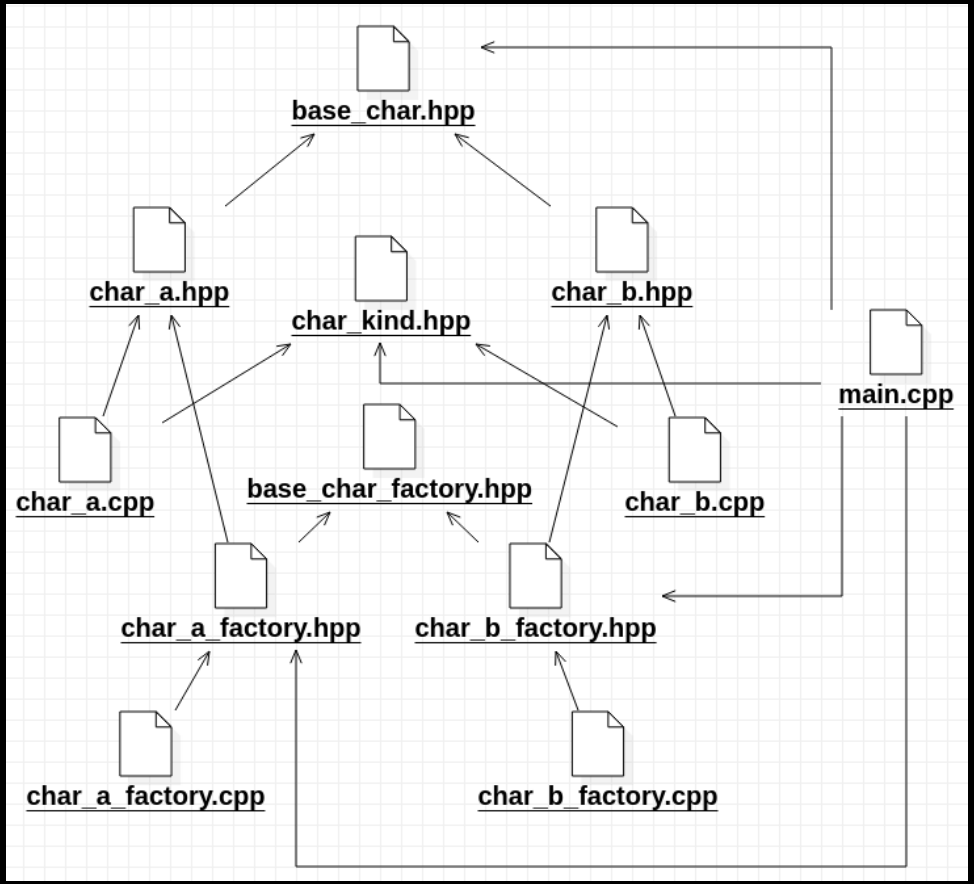
https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. Include hierarchy



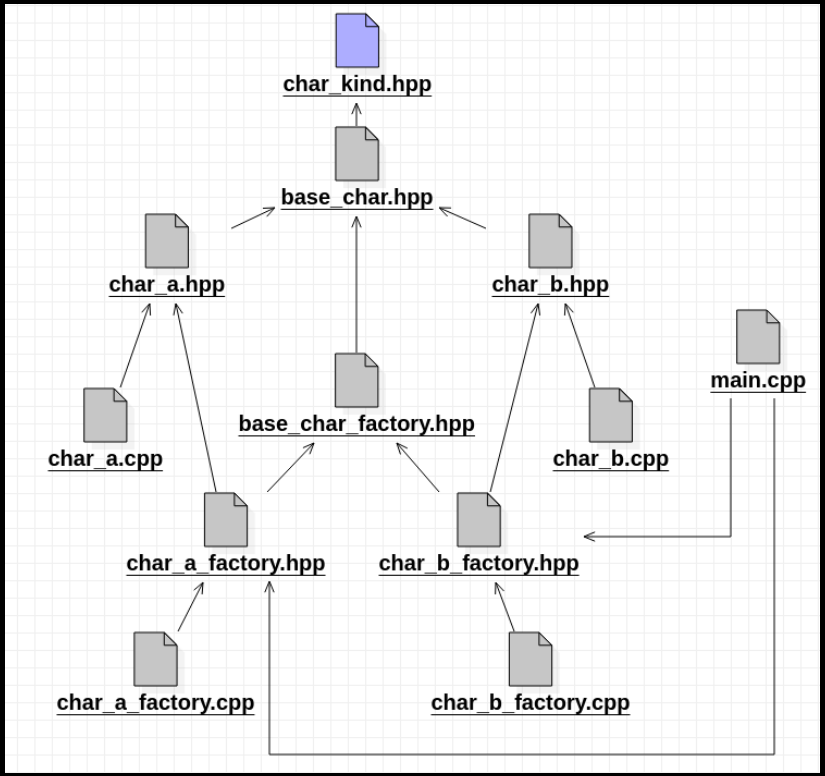
https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example

Simple example. Forward declaration

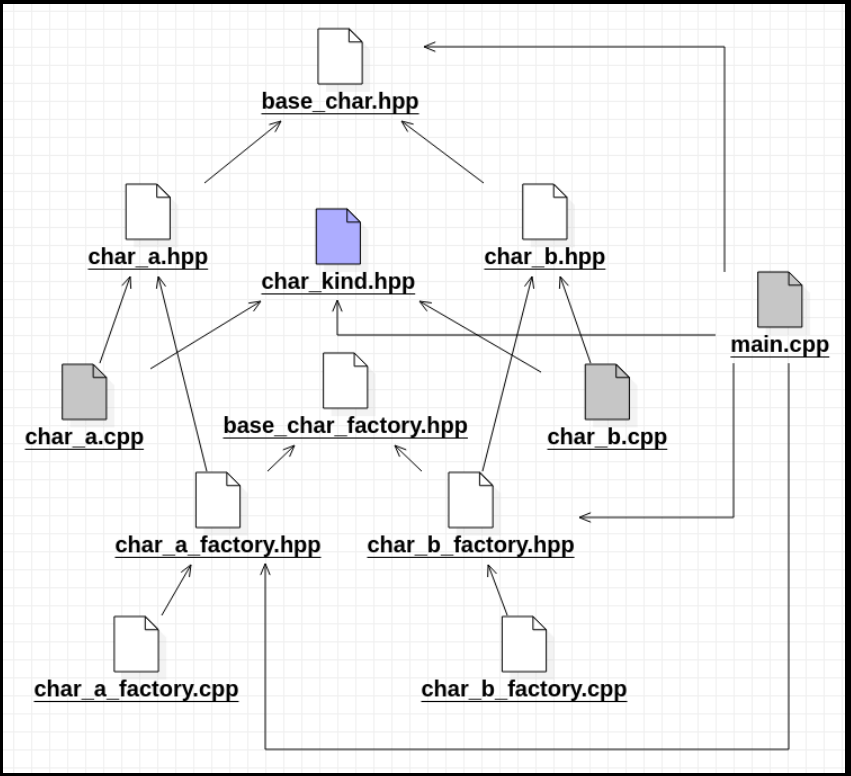


https://github.com/cppinclude/cppinclude/tree/master/docs/examples/simple_example_with_forward_declaration

Change enum and compile



Without forward declaration



With forward declaration

Using *cppinclude* for simple example

```
> cppinclude

Start initialization project ...
Start analyze sources ...
Start report results ...
Most impact files:
1 : "char_kind.hpp" impact on 11 file(s)
Included by:
        1 : "base_char.hpp" line 3, impact on 10 file(s)
2 : "base_char.hpp" impact on 10 file(s)
Included by:
        1 : "base_char_factory.hpp" line 3, impact on 5 file(s)
        2 : "char_a.hpp" line 3, impact on 2 file(s)
        3 : "char_b.hpp" line 3, impact on 2 file(s)
3 : "base_char_factory.hpp" impact on 5 file(s)
Included by:
        1 : "char_a_factory.hpp" line 3, impact on 2 file(s)
        2 : "char_b_factory.hpp" line 3, impact on 2 file(s)

...
```

Using *cppinclude* for simple example with forward declaration

```
> cppinclude

Start initialization project ...
Start analyze sources ...
Start report results ...
Most impact files:
1 : "base_char.hpp" impact on 7 file(s)
Included by:
        1 : "char_a.hpp" line 3, impact on 2 file(s)
        2 : "char_b.hpp" line 3, impact on 2 file(s)
        3 : "main.cpp" line 4
2 : "base_char_factory.hpp" impact on 5 file(s)
Included by:
        1 : "char_a_factory.hpp" line 3, impact on 2 file(s)
        2 : "char_b_factory.hpp" line 3, impact on 2 file(s)

...
```

Using *cppinclude* for *vlc*

```
Start initialization project ...
Start analyze sources ...
Start report results ...
Most impact files:
1 : "modules/demux/adaptive/ID.hpp" impact on 78 file(s)
Included by:
        1 : "modules/demux/adaptive/http/Chunk.h" line 30, impact on 73 file(s)
        2 : "modules/demux/adaptive/playlist/Inheritables.hpp" line 26, impact on 63 file(s)
        3 : "modules/demux/adaptive/ID.cpp" line 24
        4 : "modules/demux/adaptive/playlist/BaseRepresentation.cpp" line 34
2 : "modules/demux/adaptive/http/ConnectionParams.hpp" impact on 78 file(s)
Included by:
        1 : "modules/demux/adaptive/http/Chunk.h" line 29, impact on 73 file(s)
        2 : "modules/demux/adaptive/http/HTTPConnection.hpp" line 28, impact on 5 file(s)
        3 : "modules/demux/adaptive/http/AuthStorage.cpp" line 25
        4 : "modules/demux/adaptive/http/ConnectionParams.cpp" line 24
        5 : "modules/demux/adaptive/http/HTTPConnection.cpp" line 25
        6 : "modules/demux/adaptive/http/HTTPConnectionManager.cpp" line 30

...
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/vlc

Using *cppinclude* for *gcc*

```
Start initialization project ...
Start analyze sources ...
Start report results ...
Most impact files:
1 : "gcc/config/arm/arm_mve_types.h" impact on 2496 file(s)
Included by:
        1 : "gcc/config/arm/arm_mve.h" line 35, impact on 2484 file(s)
        2 : "gcc/config/arm/arm_cde.h" line 140, impact to 10 file(s)
2 : "gcc/config/arm/arm_mve.h" impact on 2484 file(s)
Included by:
        1 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/asrl.c" line 5
        2 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/lsll.c" line 5
        3 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/mve_fp_fpu1.c" line 6
        4 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/mve_fp_fpu2.c" line 6
        5 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/mve_fpu1.c" line 6
        6 : "gcc/testsuite/gcc.target/arm/mve/intrinsics/mve_fpu2.c" line 6

...
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/gcc

Show standard library header files

> cppinclude --show_std_files=true

```
Start initialization project ...
Start analyze sources ...
Start report results ...
Most impact files:
1 : "assert.h" impact on 64 file(s)
Included by:
        1 : "luaconf.h" line 701, impact on 62 file(s)
        2 : "lutf8lib.c" line 13, impact on 1 file(s)
        3 : "ltests.h" line 24
        4 : "onelua.c" line 27
2 : "stddef.h" impact on 64 file(s)
Included by:
        1 : "luaconf.h" line 12, impact on 62 file(s)
        2 : "lua.h" line 13, impact on 61 file(s)
        3 : "llimits.h" line 12, impact on 40 file(s)
        4 : "lauxlib.h" line 12, impact on 18 file(s)
        5 : "lmem.h" line 11, impact on 15 file(s)
...
```

https://github.com/cppinclude/cppinclude/tree/master/docs/examples/lua

First run

1. Detect unresolved files:

> *cppinclude --report unresolved*

2. If a lot of files then apply limits to reports

> *cppinclude --report unresolved --report_limit=5 --report_details_limit=3*

- *report_limit=5* – max. 5 unresolved files
- *report_details_limit=3 – max. 3 files that include unresolved file*

3. Create *.cppinclude.json* in project directory or customize via command arguments

First run. Customization in json

```
"ignore_system_includes" : true,
"ignore_dirs" : [
    "3rdparty",
    "gst"
],
"include_dirs" : [

    "3rdparty/libmygpo-qt5/src/",
    "3rdparty/libprojectm/",
    "3rdparty/qtiocompressor/",
    "3rdparty/tinysvcmdns/",
    "3rdparty/qtsingleapplication/",

    "ext/libclementine-tagreader/",

    "src/"
],
```

```
"ignore_files" : [

    "core/.*",
    "dbus/.*",

    "gtest/.*",
    "gmock/.*",

    "gst/moodbar/.*",

    "config.h",
    "version.h",

    ".*.pb.h",
    "ui_.*",

    "analyzer.h",
    "backgroundthread.h"

]
```

https://github.com/cppinclude/cppinclude/blob/master/docs/examples/clementine/.cppinclude.json

Cppinclude. Arguments

https://github.com/cppinclude/cppinclude/blob/master/README.md

Name	Short description
--configuration_file=file	Path to configuration file (default: .cppinclude.json)
--project_dir=dir	Project directory
--file_extensions=arg1,arg2,...	Extensions C++ files (default: *.cpp, *.hpp,*.c,*.h,*.cxx,*.hxx)
--analyze_without_extension=true	Analyze files without extension (default: false)
--include_dirs=dir1,dir2,...	Include directories
--ignore_dirs=dir1,dir2,...	Directories that will be ignored
--ignore_system_includes=true	Ignore headers in <> (default: false)
--ignore_files=regexp1,regexp2,...	Files will be ignored by regexp
--report=name1,name2,...	List reports (default: unresolved,most_impact)
--report_limit=42	Maximum elements in report, 0 - unlimited (default: 10)
--report_details_limit=42	Maximum details in report, 0 - unlimited (default: 10)
--show_std_files	Show standard library headers in output (default: false)
--help	Show usage
--verbose	Verbose mode
--version	Show application version

Contacts

cppinclude@yandex.com

Thanks :)