

# Compile-time tools supporting generic programming in C++

Ábel Sinkovics

Morgan Stanley

# Generic programming

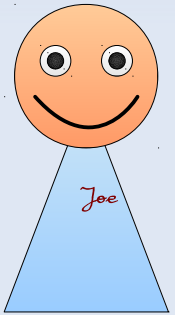
- "Generic programming is a programming paradigm for developing efficient, reusable software libraries" <http://www.generic-programming.org/>
- "Generic programming is about generalizing software components so that they can be easily reused in a wide variety of situations."  
[http://www.boost.org/community/generic\\_programming.html](http://www.boost.org/community/generic_programming.html)

# Generic programming

- "Generic programming is a programming paradigm for developing efficient, reusable software libraries" <http://www.generic-programming.org/>
- "Generic programming is about generalizing software components so that they can be easily reused in a wide variety of situations."  
[http://www.boost.org/community/generic\\_programming.html](http://www.boost.org/community/generic_programming.html)

In C++ they are implemented using templates

# Using templates

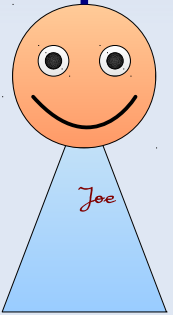


# Using templates

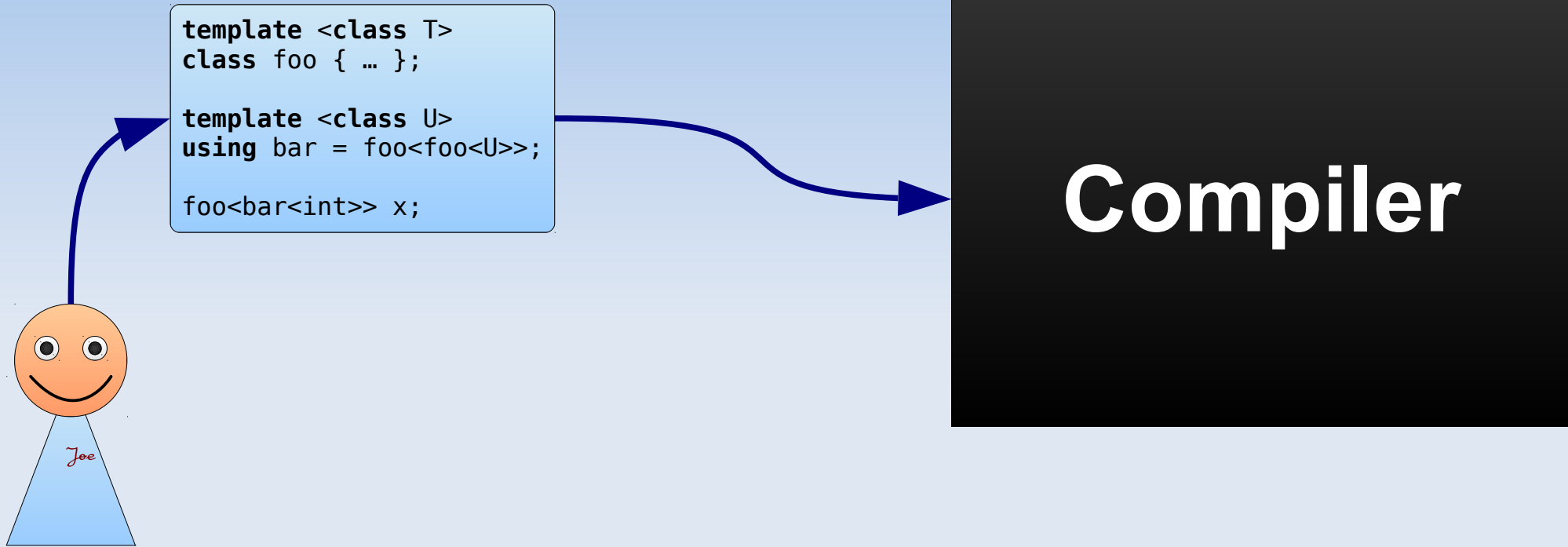
```
template <class T>
class foo { ... };

template <class U>
using bar = foo<foo<U>>;

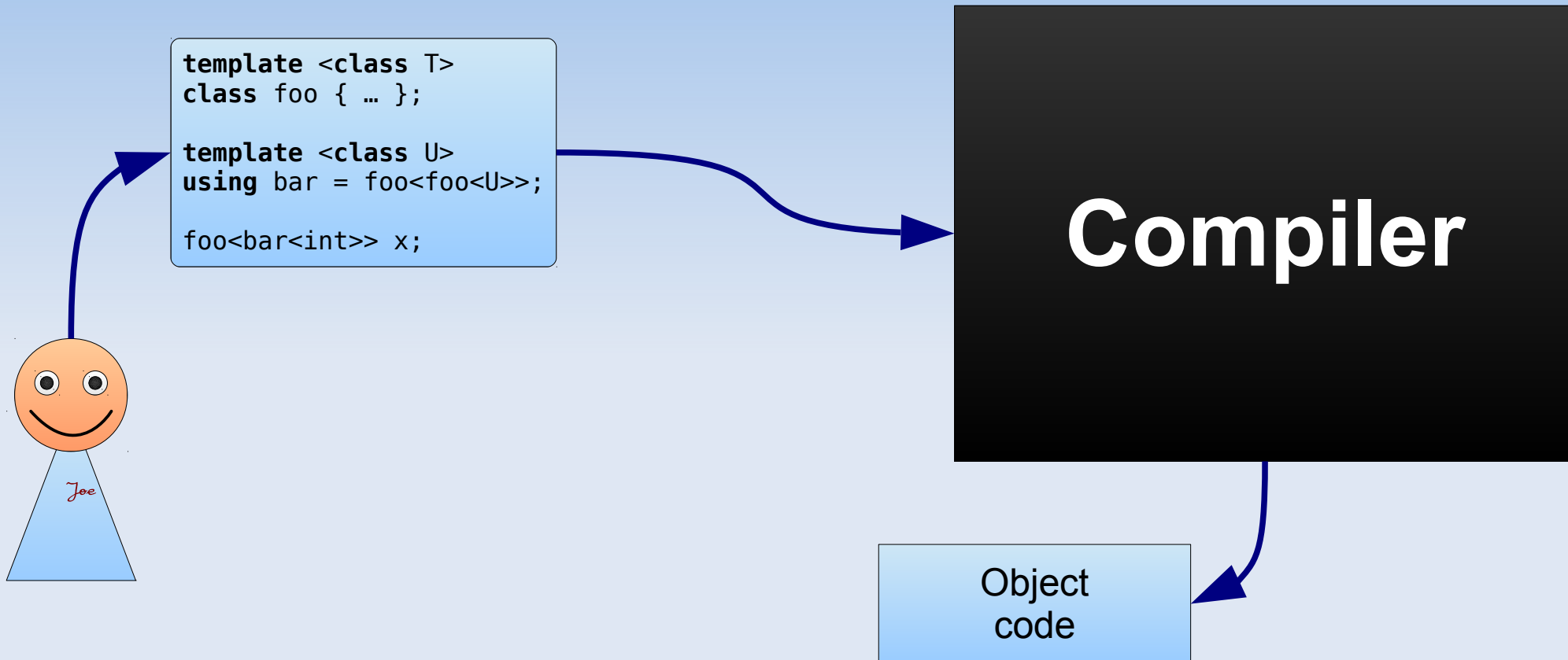
foo<bar<int>> x;
```



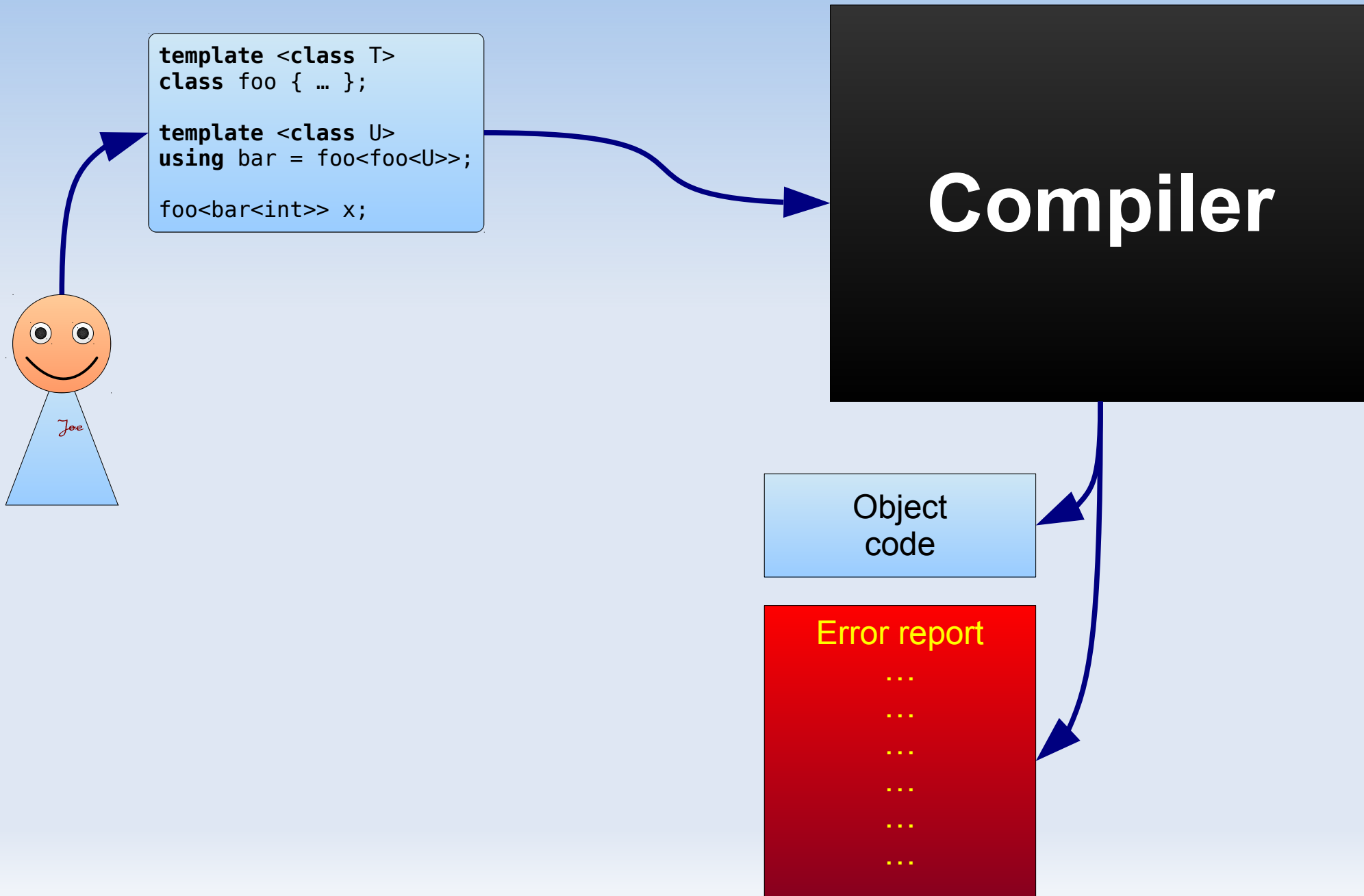
# Using templates



# Using templates

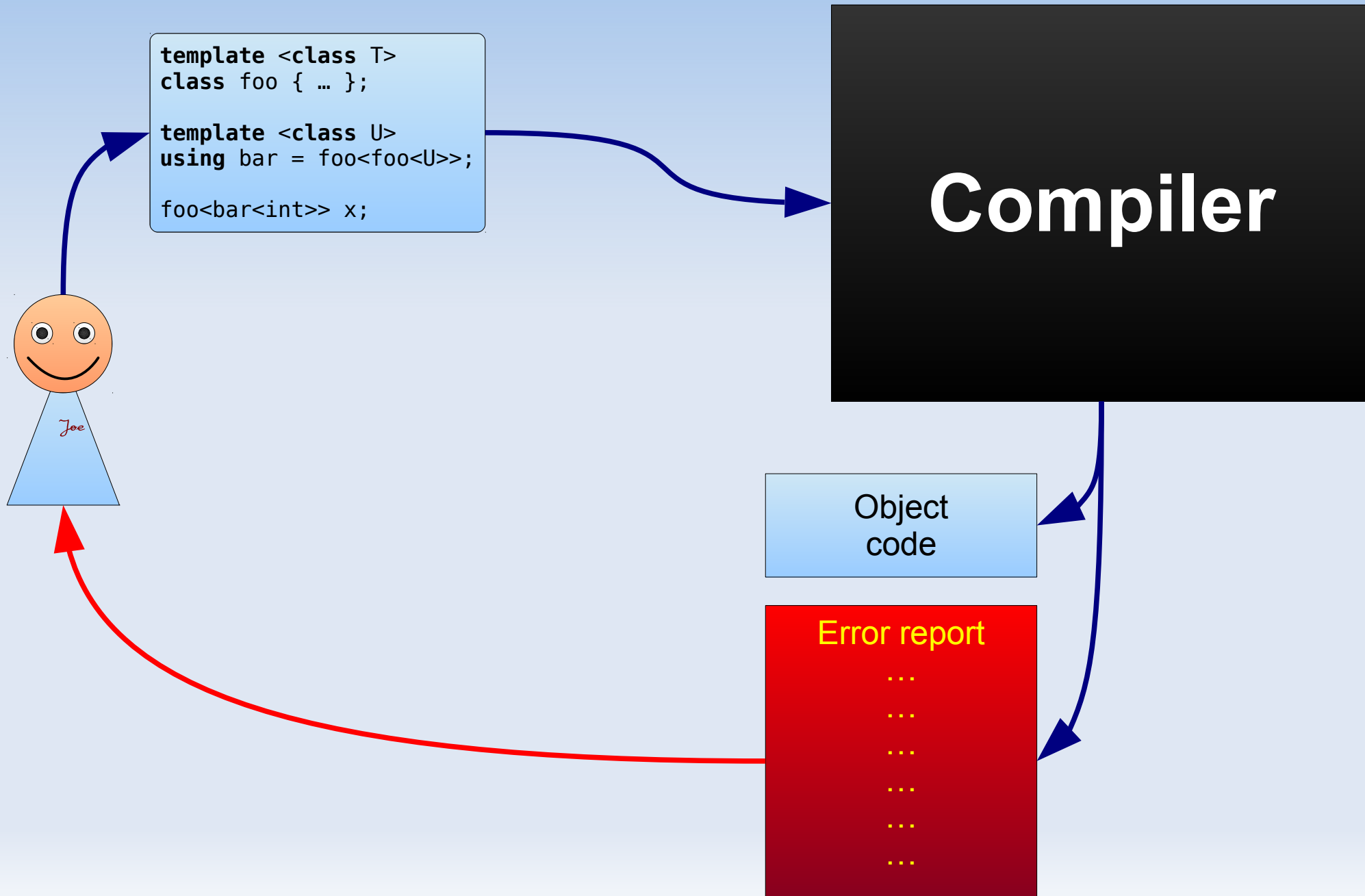


# Using templates

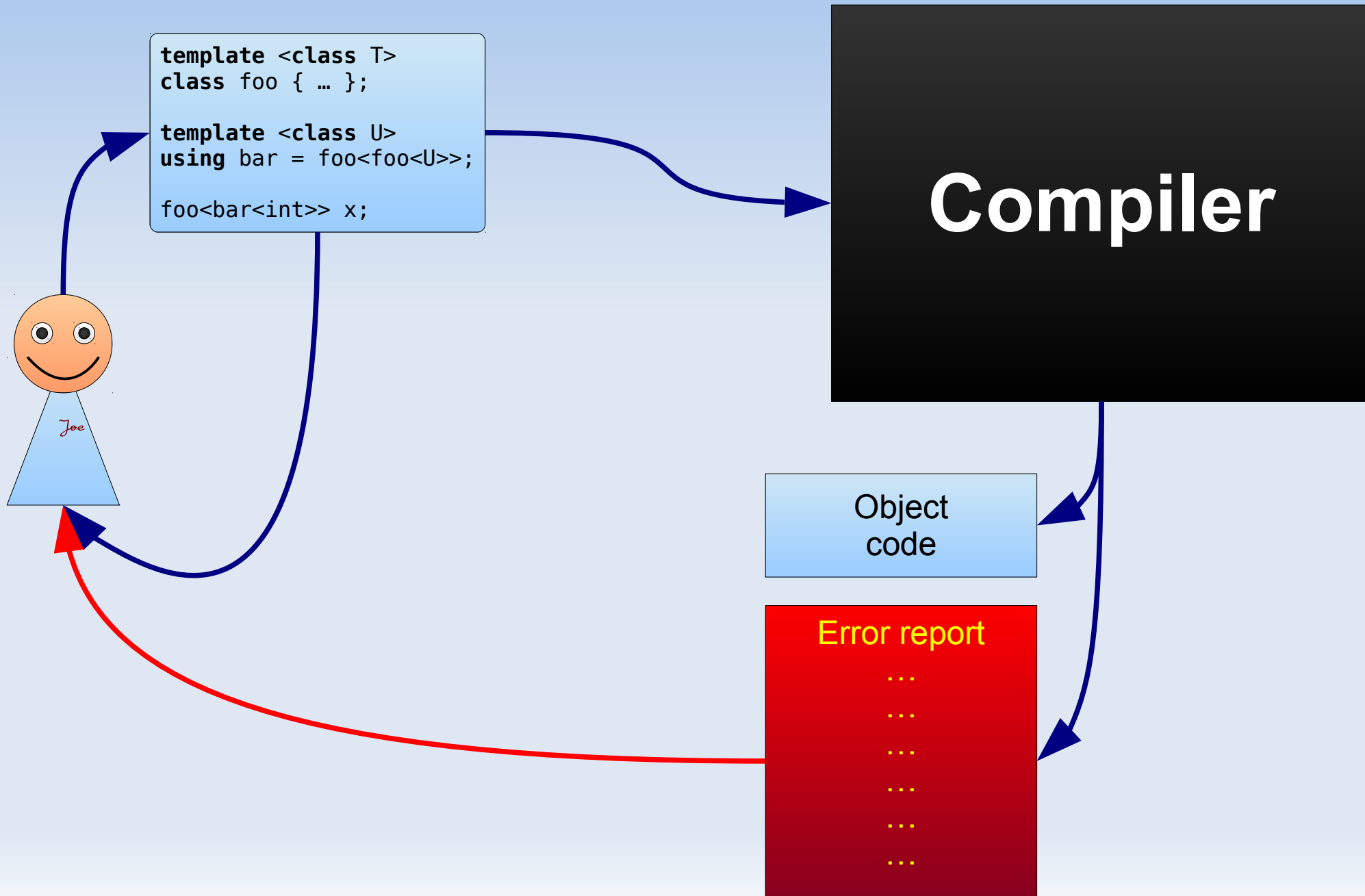




# Using templates



# Using templates



# Using templates

```
template <class T>
class foo { ... };

template <class U>
using bar = foo<foo<U>>;

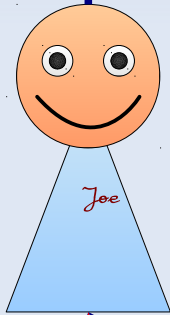
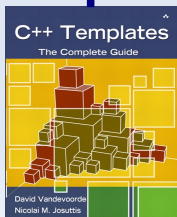
foo<bar<int>> x;
```

Compiler

Object  
code

Error report

...



# Using templates

```
template <class T>
class foo { ... };

template <class U>
using bar = foo<foo<U>>;

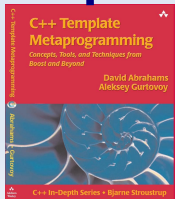
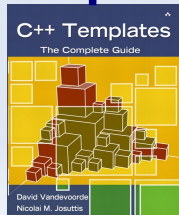
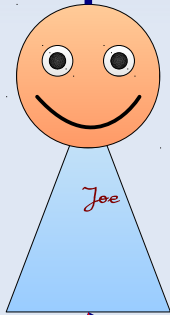
foo<bar<int>> x;
```

Compiler

Object  
code

Error report

...



# Using templates

```
template <class T>
class foo { ... };

template <class U>
using bar = foo<foo<U>>;

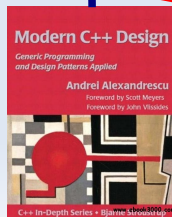
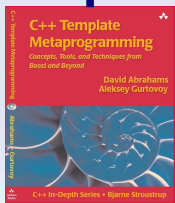
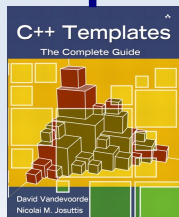
foo<bar<int>> x;
```

Compiler

Object  
code

Error report

...



# Using templates

```
template <class T>
class foo { ... };

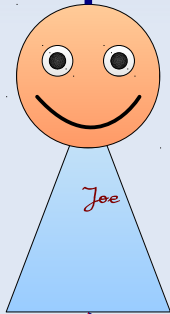
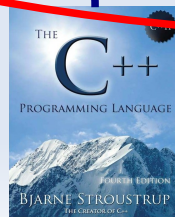
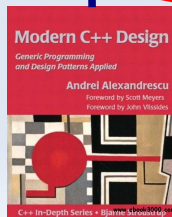
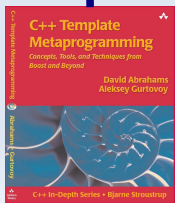
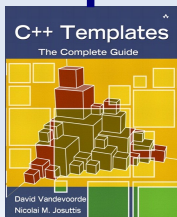
template <class U>
using bar = foo<foo<U>>;

foo<bar<int>> x;
```

Compiler

Object  
code

Error report



# Using templates

```
template <class T>  
class foo { ... };
```

```
template <class U>  
using bar = foo<foo<U>>;
```

```
foo<bar<int>> x;
```

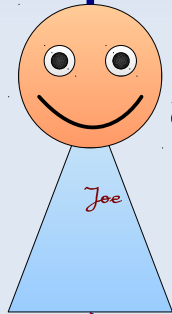
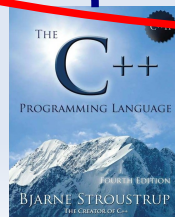
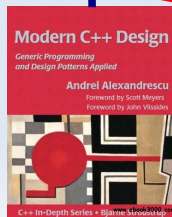
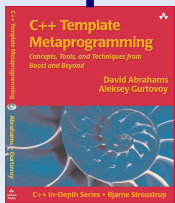
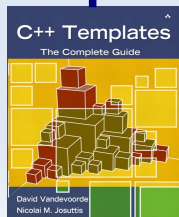
Compiler

template

template

foo<bar<int>>

iler



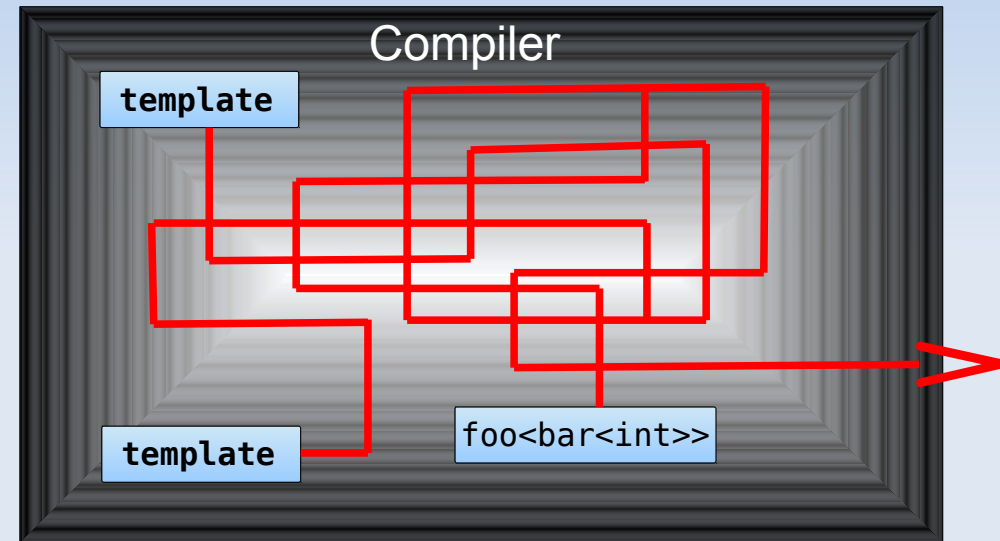
# Template metaprogrammers

- They are using templates heavily



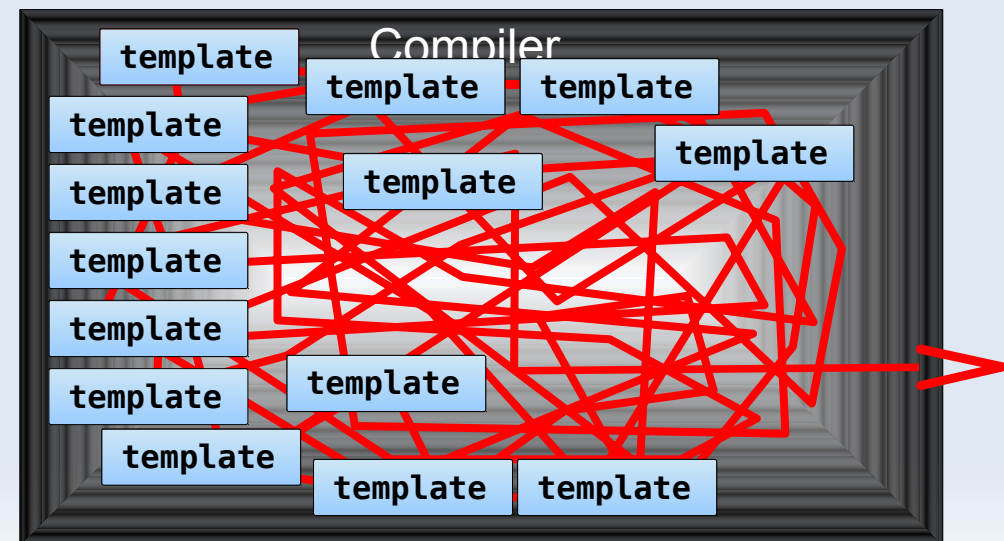
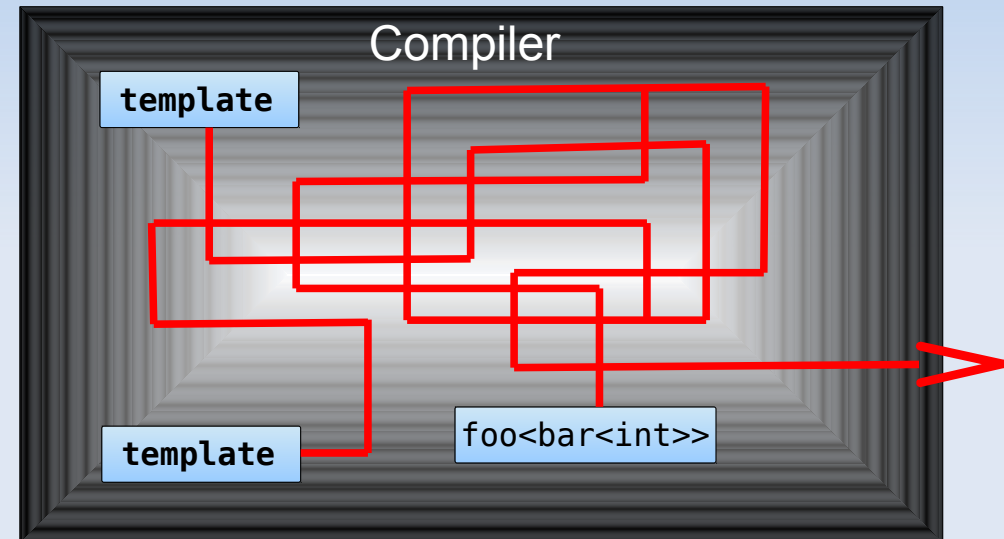
# Template metaprogrammers

- They are using templates heavily
- Generic library usage



# Template metaprogrammers

- They are using templates heavily
- Generic library usage
- Template metaprogram



# Template metaprogrammers

- Recently: advanced tools for template metaprogrammers
  - Templight
  - Metashell + MDB

# Template metaprogrammers

- Recently: advanced tools for template metaprogrammers
  - Templight
  - Metashell + MDB
- Can those tools be useful for generic library development and usage?

# Available tools

- The compiler
  - Error messages
  - Type pretty-printing
- IDEs
- Runtime debuggers

# Available tools

- The compiler
  - Error messages
  - Type pretty-printing
- IDEs
- Runtime debuggers
- Template metaprogrammer tools
  - Metashell (with MDB)
  - Templight

# What is the type of...?

```
#include <a.hpp>

int main()
{
    a<int>::handle x;
    return 0;
}
```

# What is the type of...?

```
#include <a.hpp>

int main()
{
    a<int>::handle x;
    return 0;
}
```



# What is the type of...?

a.hpp

```
#include <a.hpp>

int main()
{
    a<int>::handle();
    return 0;
}
```

```
#ifndef A_HPP
#define A_HPP

#include "b.hpp"

template <class T>
class a : public b<T, int>
{
};

#endif
```

# What is the type of...?

a.hpp

```
#ifndef A_HPP
#define A_HPP

#include "b.hpp"
```

b.hpp

```
#ifndef B_HPP
#define B_HPP

#include "c.hpp"
#include "d.hpp"
```

```
template <class T, class U>
class b
{
public:
    typedef typename c<T, d<U>>::handle handle;
};
```

```
#endif
```

```
class T>
public b<T, int>
```

# What is the type of...?

a.hpp

```
#ifndef A_HPP  
#define A_HPP
```

c.hpp

```
#ifndef C_HPP  
#define C_HPP  
  
#include "c_factory.hpp"  
  
template <class T, class U>  
class c  
{  
public:  
    typedef typename c_factory<typename U::item>::handle handle;  
};  
  
#endif  
};  
  
#endif
```

# What is the type of...?

```
#ifndef C_HPP  
#define C_HPP
```

```
#include "c_factory.hpp"
```

```
template <class C>  
class c  
{  
public:  
    typedef typename C::handle handle;  
};
```

```
#endif
```

```
};
```

```
#endif
```

*a<int>::handle*

*b<int, int>::handle*

*c<int, d<int>>::handle*

*c\_factory<d<int>::item>::handle*

*...*

~~#ifndef A\_HPP~~

a.hpp

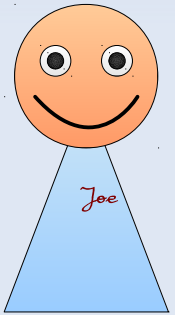
c.hpp

*>::handle handle;*

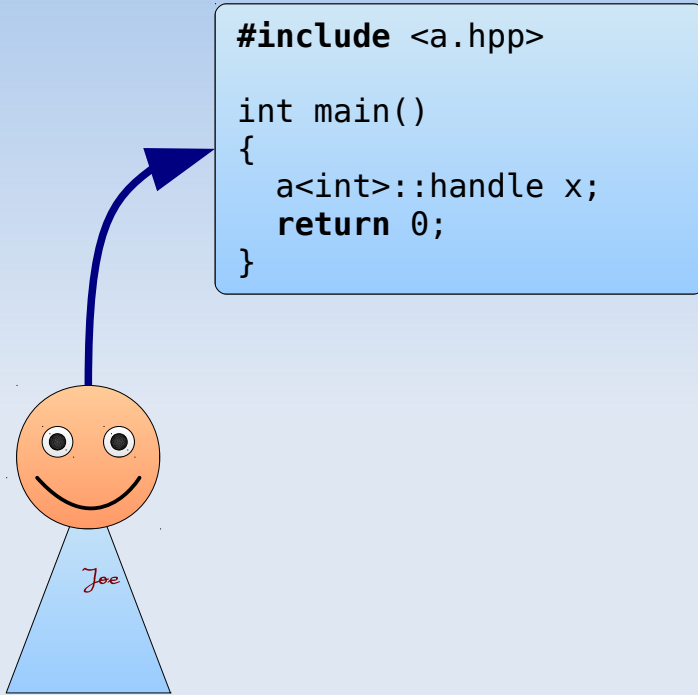
# Approaches

- Enforced error message
- Displaying the name at runtime
- IDEs
- Debuggers
- Metaprogrammer tools

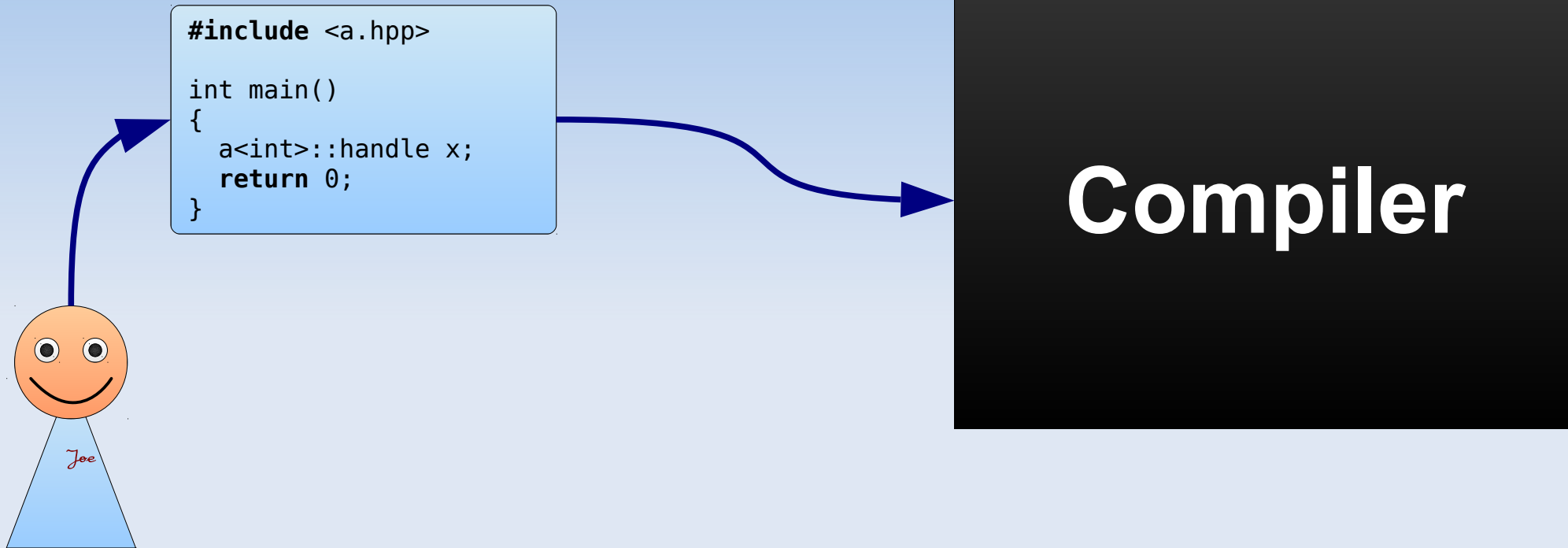
# Enforced error message



# Enforced error message

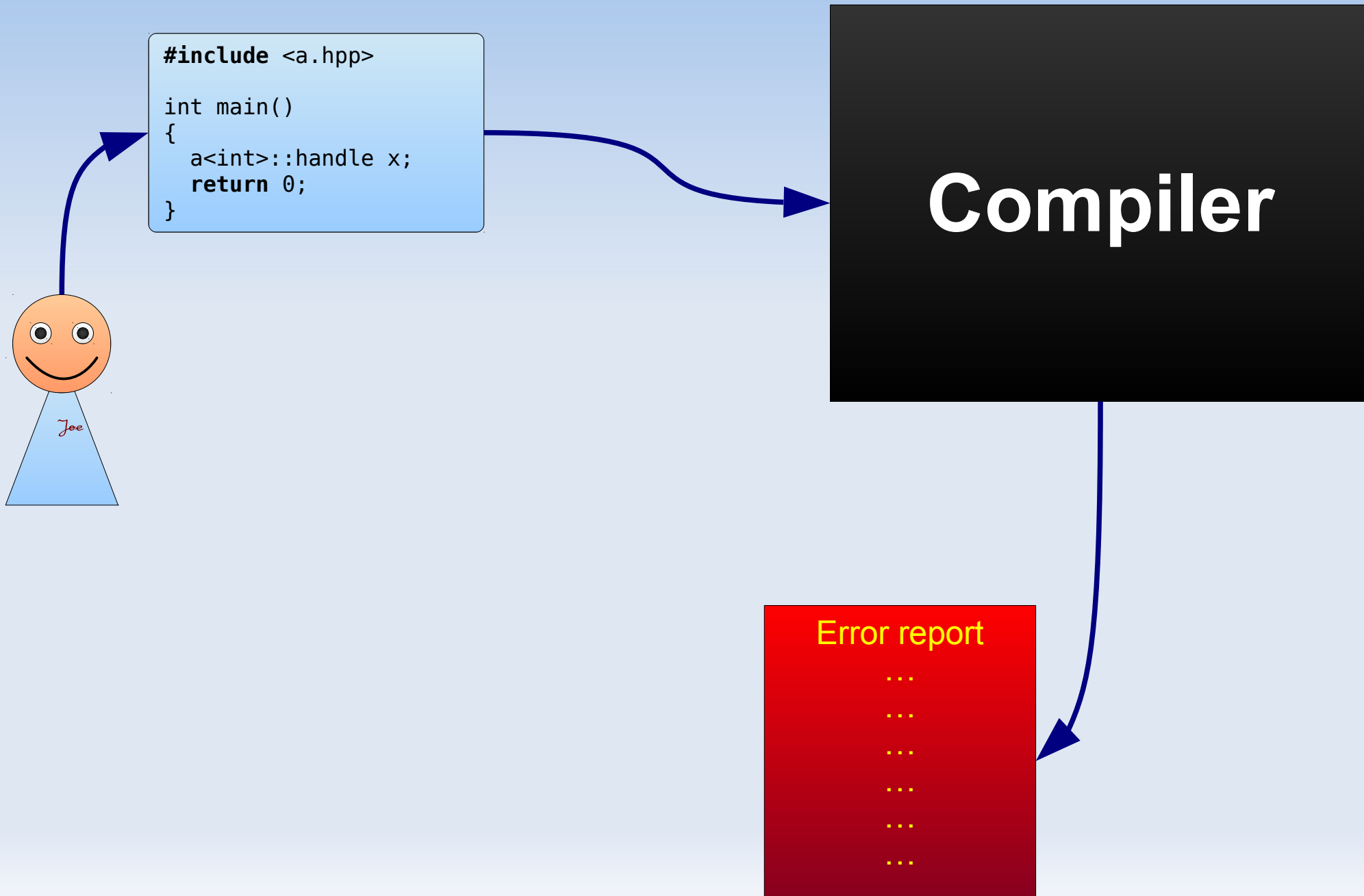


# Enforced error message

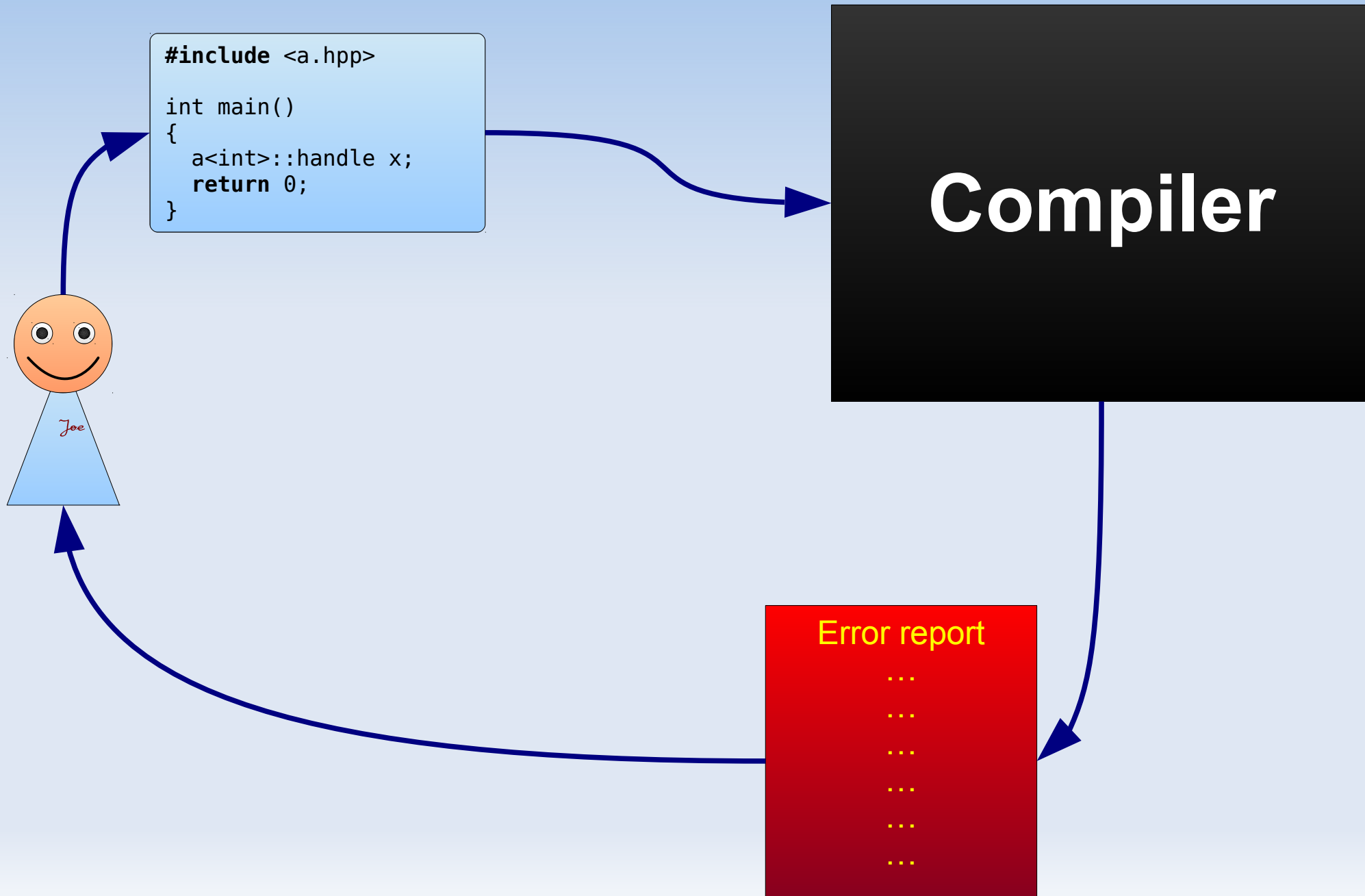




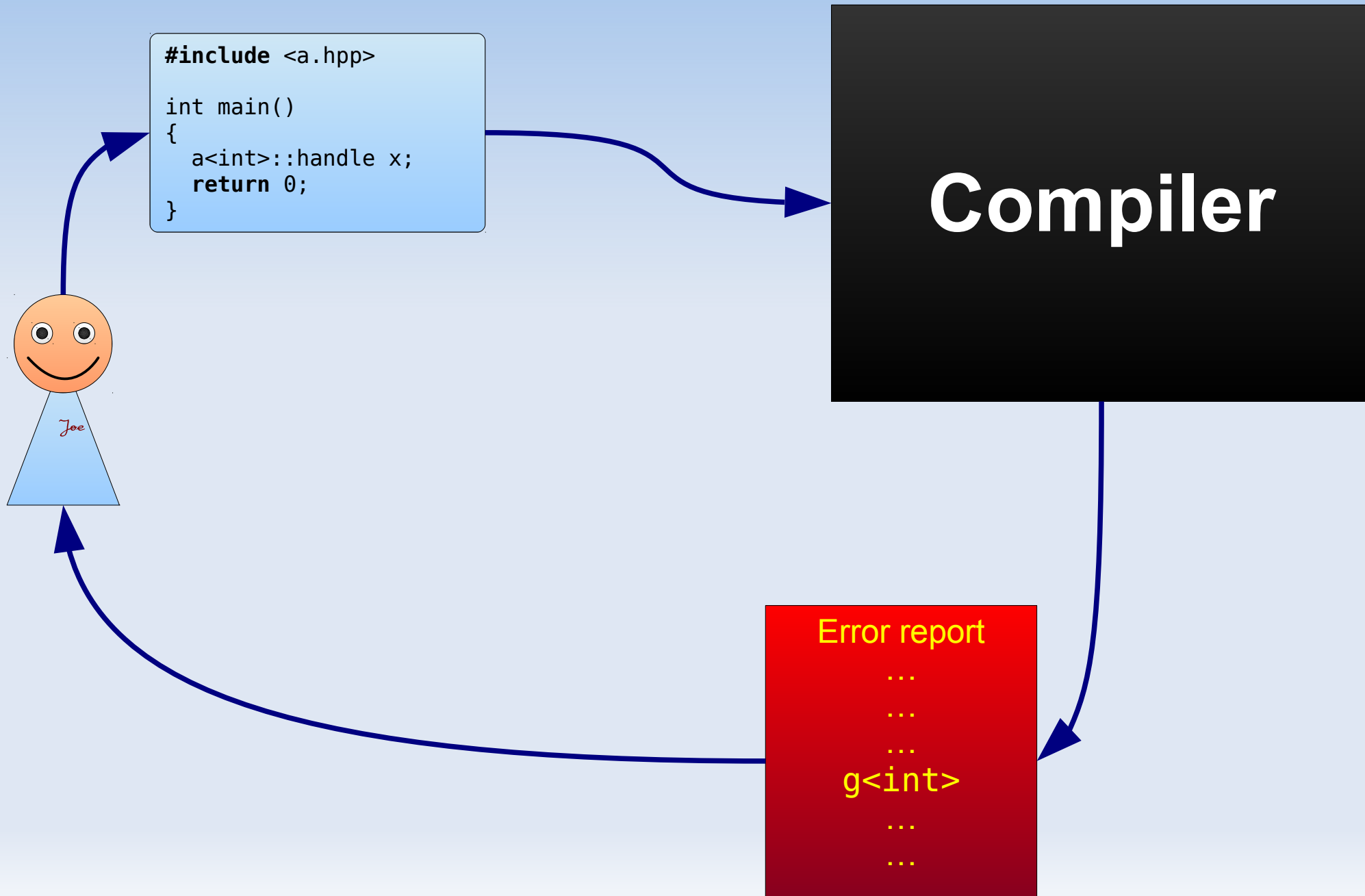
# Enforced error message



# Enforced error message



# Enforced error message



# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

```
In file included from main1_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~~
main1_err.cpp:7:39: note: in instantiation of template class
    'boost::mpl::print<g<int> >' requested here
    boost::mpl::print< a<int>::handle > t;
                                ^
1 warning generated.
```

# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

```
In file included from main1_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~~
main1_err.cpp:7:39: note: in instantiation of template class
    'boost::mpl::print<g<int> >' requested here
    boost::mpl::print< a<int>::handle > t;
                                ^
1 warning generated.
```

# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

```
In file included from main1_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~
```

Visual C++

```
boost/mpl/print.hpp(52): error C4308: negative integral constant
converted to unsigned type
    main.cpp(7) : see reference to class template instanti
ation 'boost::mpl::print<g<T>>' being compiled
    with
    [
        T=int
    ]
```

# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

In file included from main1\_err.cpp:3:  
boost/mpl/print.hpp:50:23: warning: division by zero is undefined  
[-Wdivision-by-zero]  
 const int m\_x = 1 / (sizeof(T) - sizeof(T));

Visual C++

```
boost/mpl/print.hpp(52): error C4308: negative integral constant  
converted to unsigned type  
    main.cpp(7) : see reference to class template instanti  
ation 'boost::mpl::print<g<T>>' being compiled  
    with  
    [  
        T=int  
    ]
```



# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

```
In file included from main1_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~
```

Visual C++

```
boost/mpl/print.hpp(52): error C4308: negative integral constant
converted to unsigned type
    main.cpp(7) : see reference to class template instanti
ation 'boost::mpl::print<g<T>>' being compiled
    with
    [
        T=int
    ]
```

# boost::mpl::print

```
int main()
{
    boost::mpl::print< a<int>::handle > t;
}
```

Clang

In file included from main1\_err.cpp:3:  
boost/mpl/print.hpp:50:23: warning: division by zero is undefined  
[-Wdivision-by-zero]  
 const int m\_x = 1 / (sizeof(T) - sizeof(T));

Visual C++

boost/mpl/print.hpp(52): error C4308: negative integral constant  
converted to unsigned type  
 main.cpp(7) : see reference to class template instanti  
ation 'boost::mpl::print<g<T>>' being compiled  
 with  
 [  
 T=int  
 ]

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

```
metamonad/fail_with_type.hpp:9:0,
    from main1_err_mpllibs.cpp:3:
metamonad/v1/fail_with_type.hpp: In instantiation of 'void mpllibs::metamonad::
v1::fail_with_type() [with T = g<int>]':
main1_err_mpllibs.cpp:7:56:   required from here
metamonad/v1/fail_with_type.hpp:26:70: error: 'f' is not a member of 'mpllibs::
metamonad::v1::impl::FAIL_WITH_TYPE_____<g<int> >'
    impl::FAIL_WITH_TYPE_____<T>::f();
                                ^
```

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

```
metamonad/fail_with_type.hpp:9:0,
    from main1_err_mpllibs.cpp:3:
metamonad/v1/fail_with_type.hpp: In instantiation of 'void mpllibs::metamonad::
v1::fail_with_type() [with T = g<int>]':
main1_err_mpllibs.cpp:7:56:   required from here
metamonad/v1/fail_with_type.hpp:26:70: error: 'f' is not a member of 'mpllibs::
metamonad::v1::impl::FAIL_WITH_TYPE_____<g<int>>'
    impl::FAIL_WITH_TYPE_____<T>::f();
                                   ^
```

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

```
metamonad/fail_with_type.hpp:9:0,
      from main1_err_mpllibs.cpp:3:
```

Clang

In file included from main1\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

metamonad/v1/fail\_with\_type.hpp:26:68: error: no member

named 'f' in 'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_

<g<int> >'

impl::FAIL\_WITH\_TYPE\_\_\_\_\_<T>::f();

~~~~~^

main1\_err\_mpllibs.cpp:7:23: note: in instantiation of function template

specialization 'mpllibs::metamonad::v1::fail\_with\_type<g<int> >' requested

here

```
mpllibs::metamonad::fail_with_type< a<int>::handle >();
      ^
```

1 error generated.

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

```
metamonad/fail_with_type.hpp:9:0,
      from main1_err_mpllibs.cpp:3:
```

Clang

In file included from main1\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

metamonad/v1/fail\_with\_type.hpp:26:68: error: no member

named 'f' in 'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_

<g<int>>'

impl::FAIL\_WITH\_TYPE\_\_\_\_\_<T>::f();

~~~~~^

main1\_err\_mpllibs.cpp:7:23: note: in instantiation of function template

specialization 'mpllibs::metamonad::v1::fail\_with\_type<g<int>>' requested

here

```
mpllibs::metamonad::fail_with_type< a<int>::handle >();
      ^
```

1 error generated.

# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

```
metamonad/fail_with_type.hpp:9:0,
      from main1_err_mpllibs.cpp:3:
```

Clang

In file included from main1\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

Visual C++

```
metamonad\v1\fail_with_type.hpp(26): error C2039: 'f' : is not a member of
'mpllibs::metamonad::v1::impl::FAIL_WITH_TYPE_____
_____<T>'
```

```
with
[
    T=g<int>
]
```

main.cpp(7) : see reference to function template instantiation '

```
void mpllibs::metamonad::v1::fail_with_type<g<T>>(void)' being compiled
with
[
    T=int
]
```



# metamonad::fail\_with\_type

```
int main()
{
    mpllibs::metamonad::fail_with_type< a<int>::handle >();
}
```

GCC

metamonad/fail\_with\_type.hpp:9:0,  
from main1\_err\_mpllibs.cpp:3:

Clang

In file included from main1\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

Visual C++

metamonad\v1\fail\_with\_type.hpp(26): error C2039: 'f' : is not a member of  
'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_  
\_\_\_\_\_  
<T>'

with

[

T=g<int>

]

main.cpp(7) : see reference to function template instantiation '

void mpllibs::metamonad::v1::fail\_with\_type<g<T>>(void)' being compiled

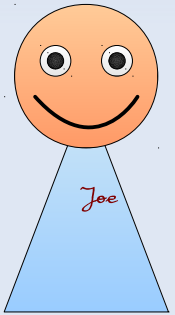
with

[

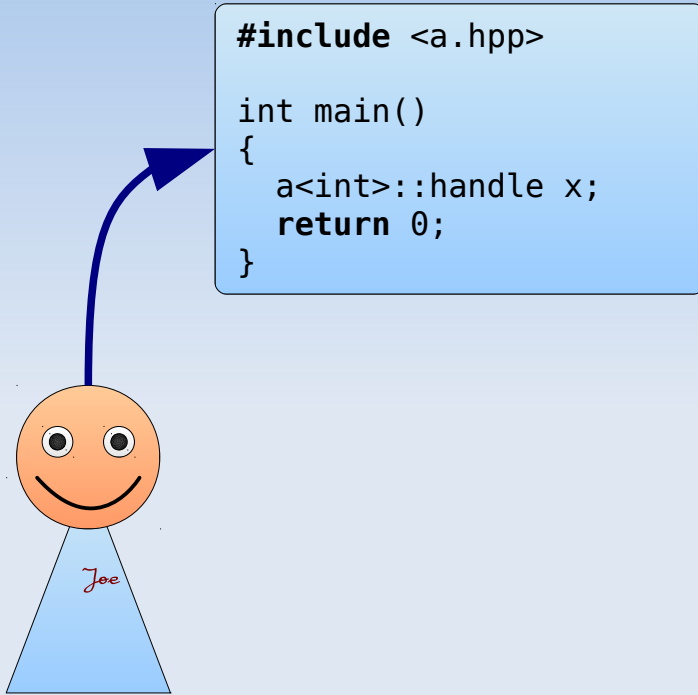
T=int

]

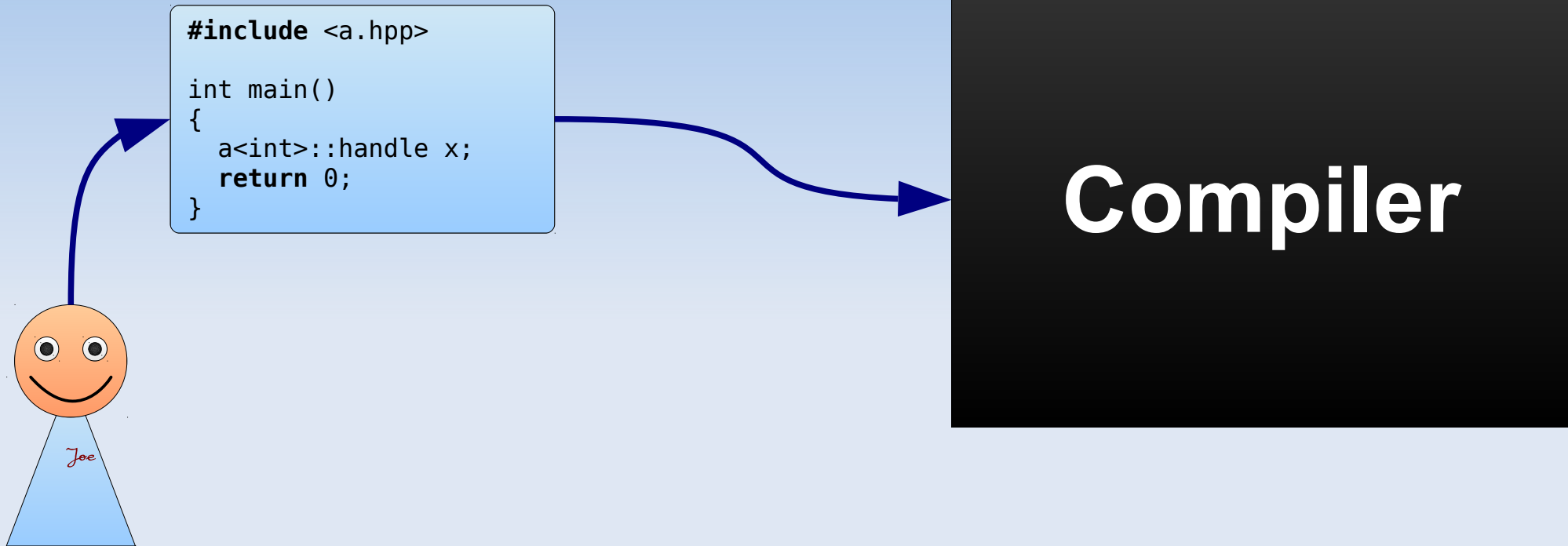
# Displaying the name at runtime



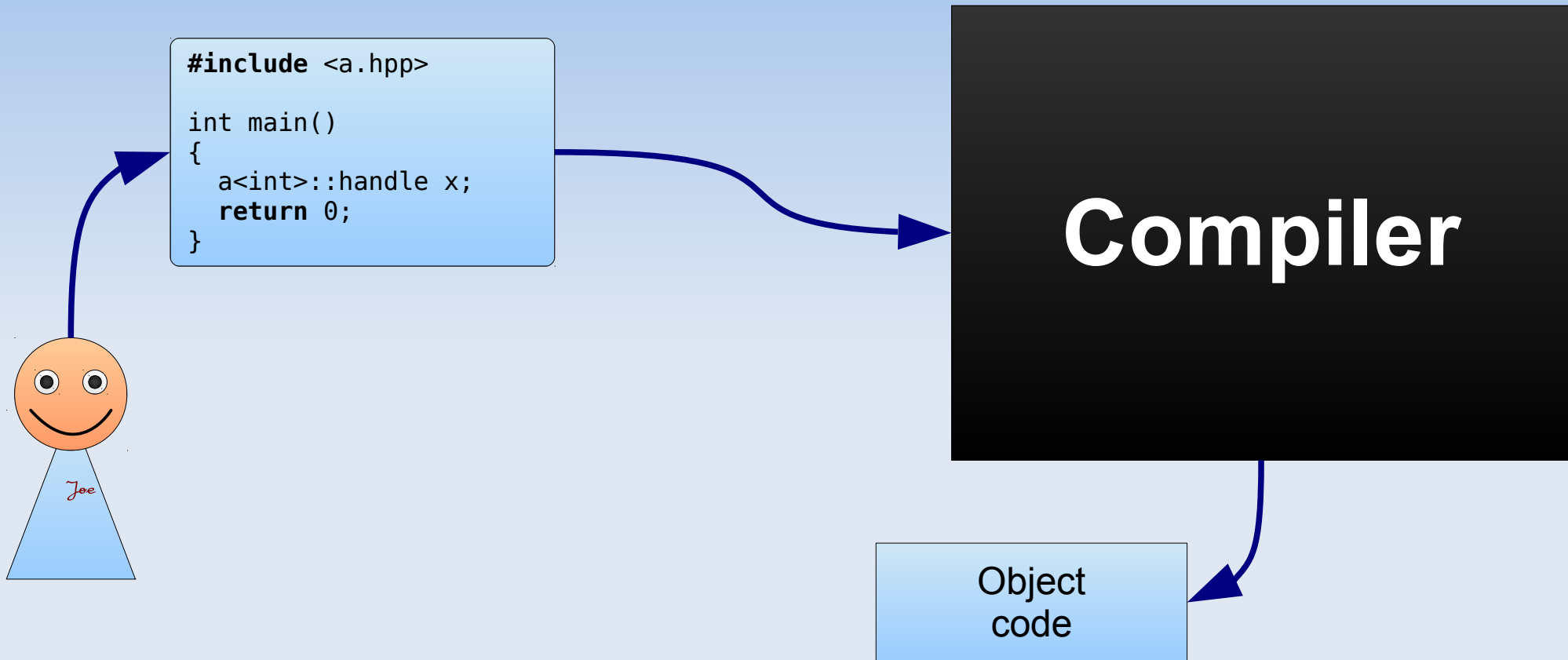
# Displaying the name at runtime



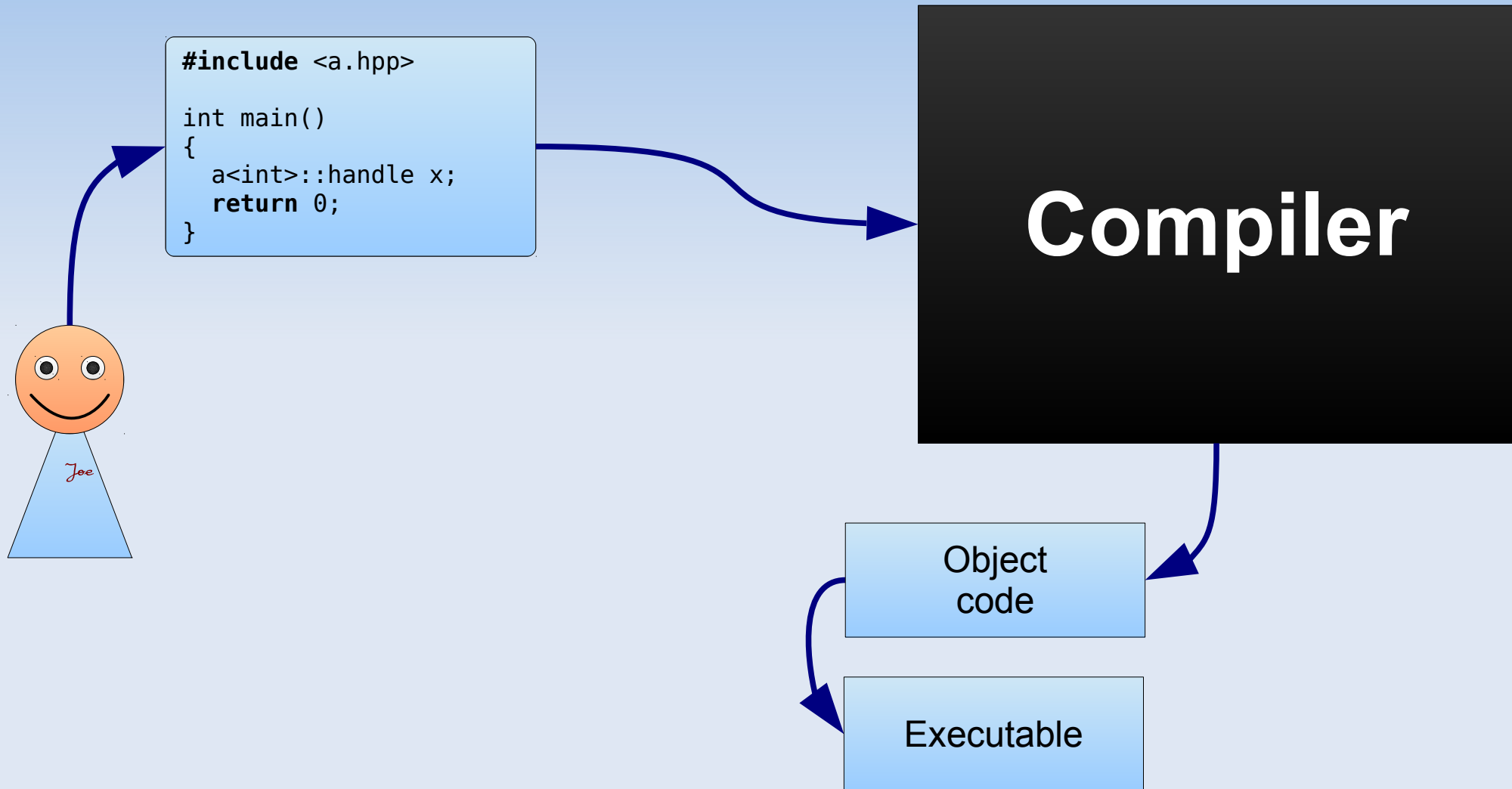
# Displaying the name at runtime



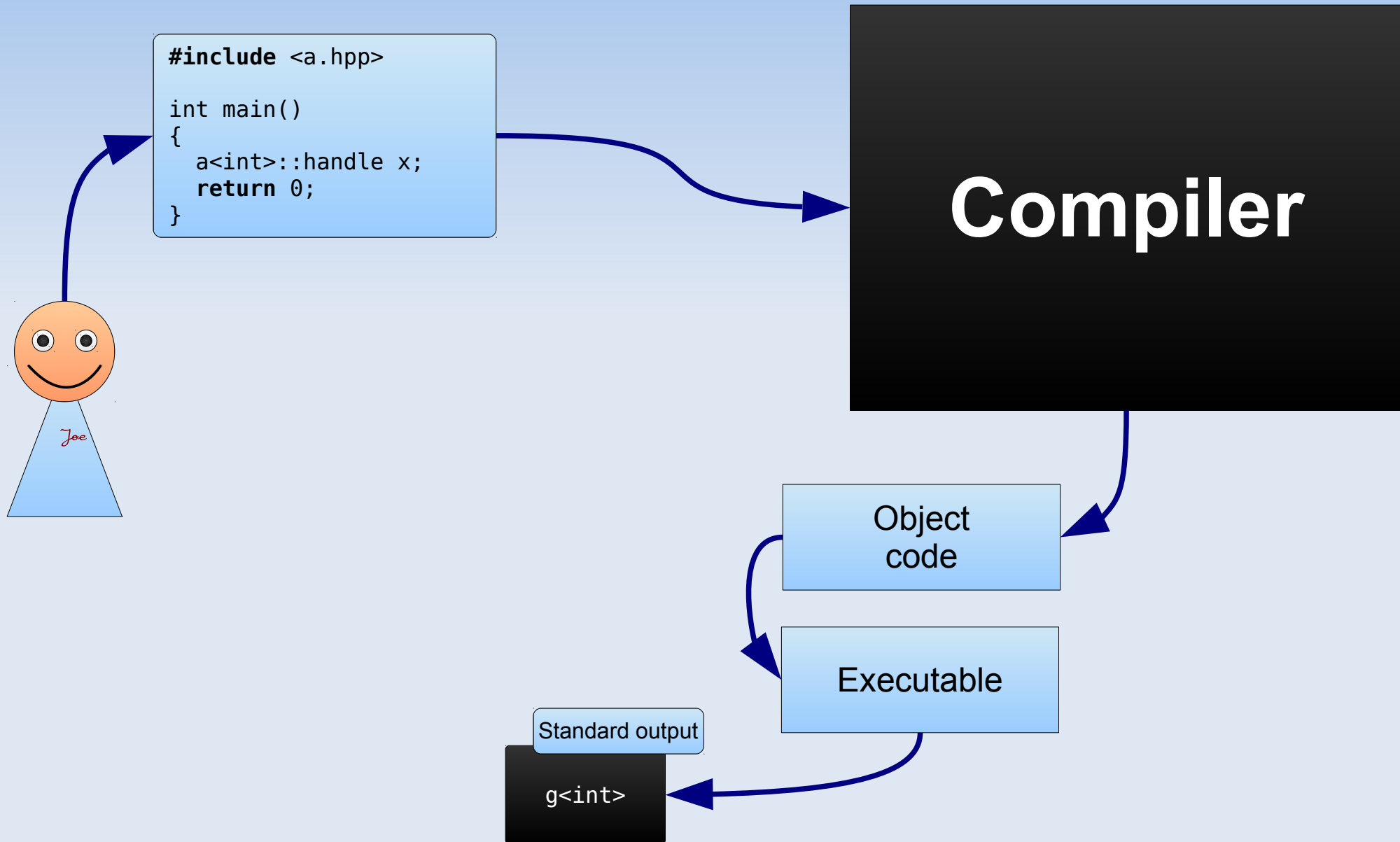
# Displaying the name at runtime



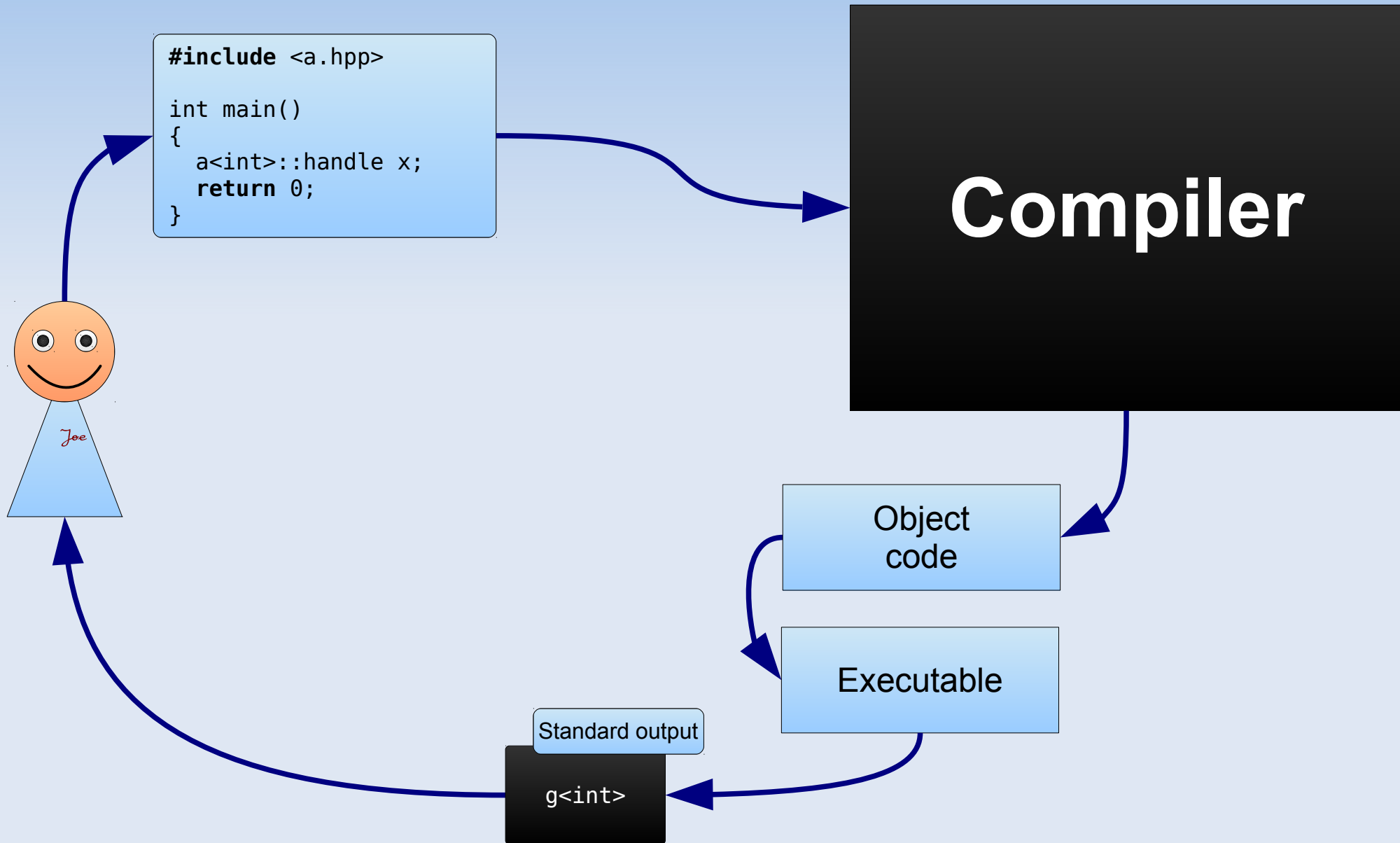
# Displaying the name at runtime



# Displaying the name at runtime



# Displaying the name at runtime





# Displaying the name at runtime

```
int main()
{
    std::cout << typeid(a<int>::handle).name();
}
```

# Displaying the name at runtime

```
int main()
{
    std::cout << typeid(a<int>::handle).name();
}
```

- Gcc: 1gIiE
- Clang: 1gIiE

# Displaying the name at runtime

```
int main()
{
    std::cout << typeid(a<int>::handle).name();
}
```

- Gcc: 1gIiE
  - Clang: 1gIiE
- c++filt → g<int>

# Displaying the name at runtime

```
int main()
{
    std::cout << typeid(a<int>::handle).name();
}
```

- Gcc: 1gIiE
  - Clang: 1gIiE
  - Visual C++ : class g<int>
- c++filt →

# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        <<
        << std::endl;
}
```

# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        << type_id_with_cvr<                >()
        << std::endl;
}
```

# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        << type_id_with_cvr<a<int>::handle>()
        << std::endl;
}
```

# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        << type_id_with_cvr<a<int>::handle>().pretty_name()
        << std::endl;
}
```



# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        << type_id_with_cvr<a<int>::handle>().pretty_name()
        << std::endl;
}
```



g<int>

# Displaying the name at runtime

- Boost.TypeIndex

```
int main()
{
    using boost::typeindex::type_id_with_cvr;
    std::cout
        << type_id_with_cvr<a<int>::handle>().pretty_name()
        << std::endl;
}
```

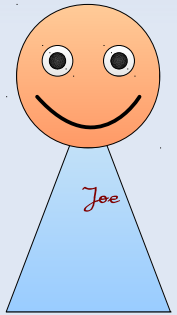
Visual C++

class

g<int>

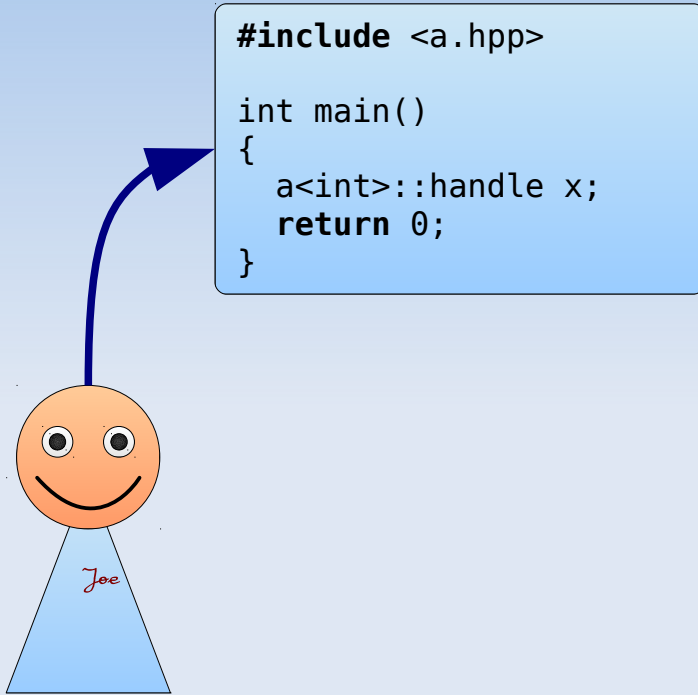
The diagram illustrates the runtime behavior of the provided C++ code. A blue arrow points from the `type_id_with_cvr<a<int>::handle>()` expression in the code to a black box labeled `g<int>`. Another blue arrow points from the text 'Visual C++' to the word 'class', which is positioned to the left of the `g<int>` box, indicating that the code is compiled by Visual C++ and the resulting class is `g<int>`.

# IDEs



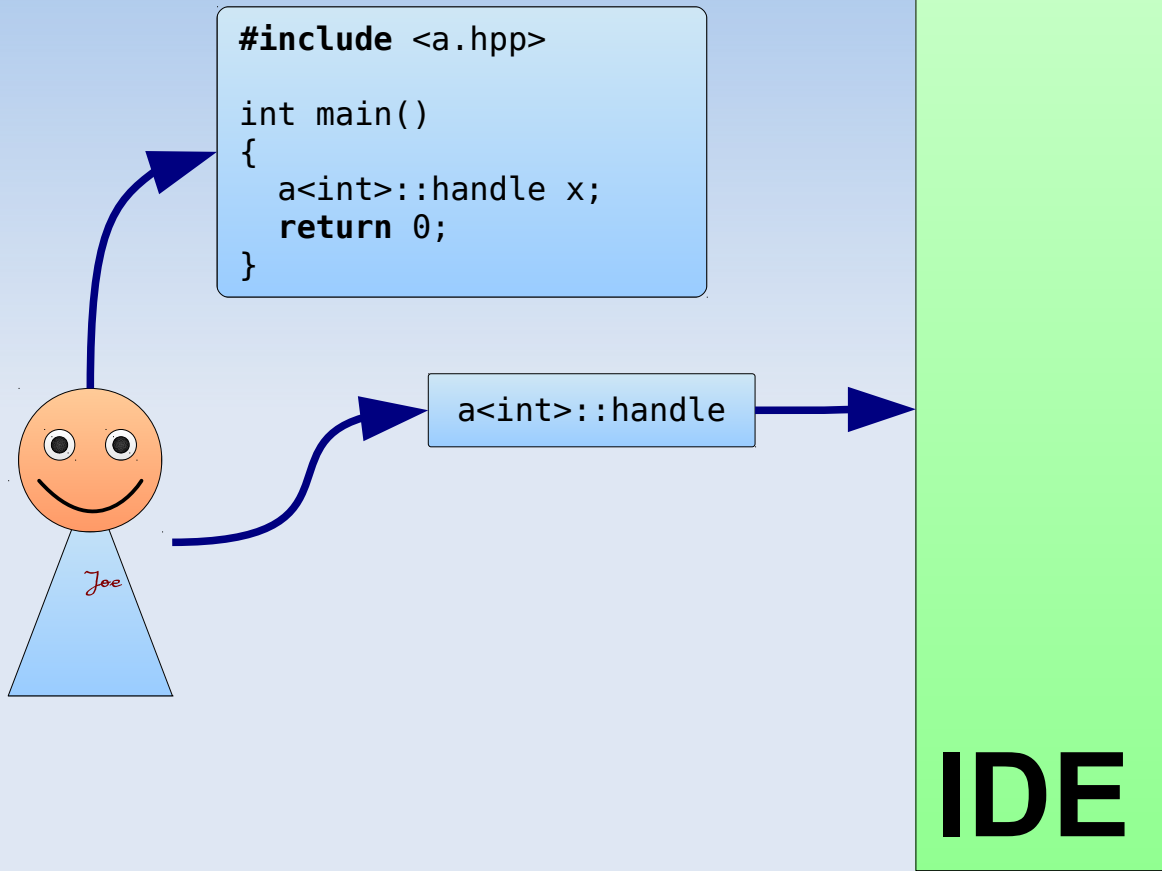
**IDE**

# IDEs

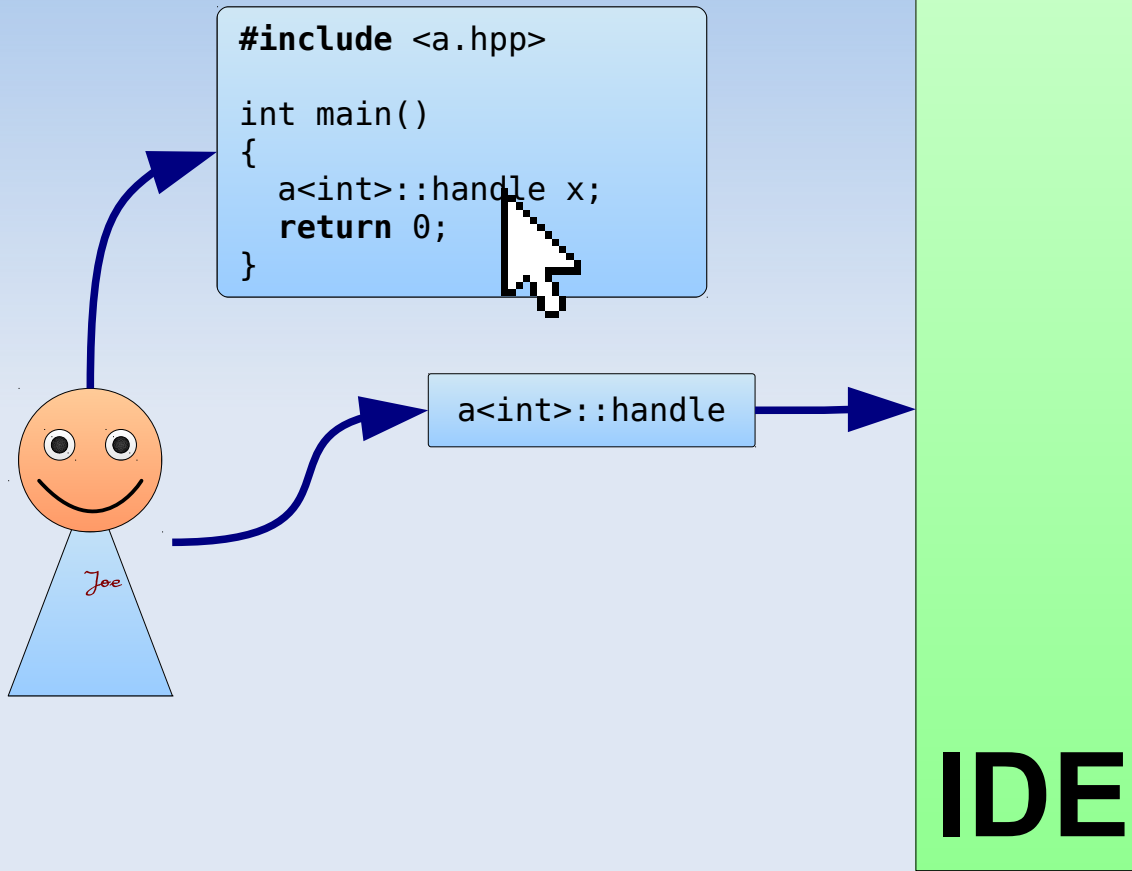


**IDE**

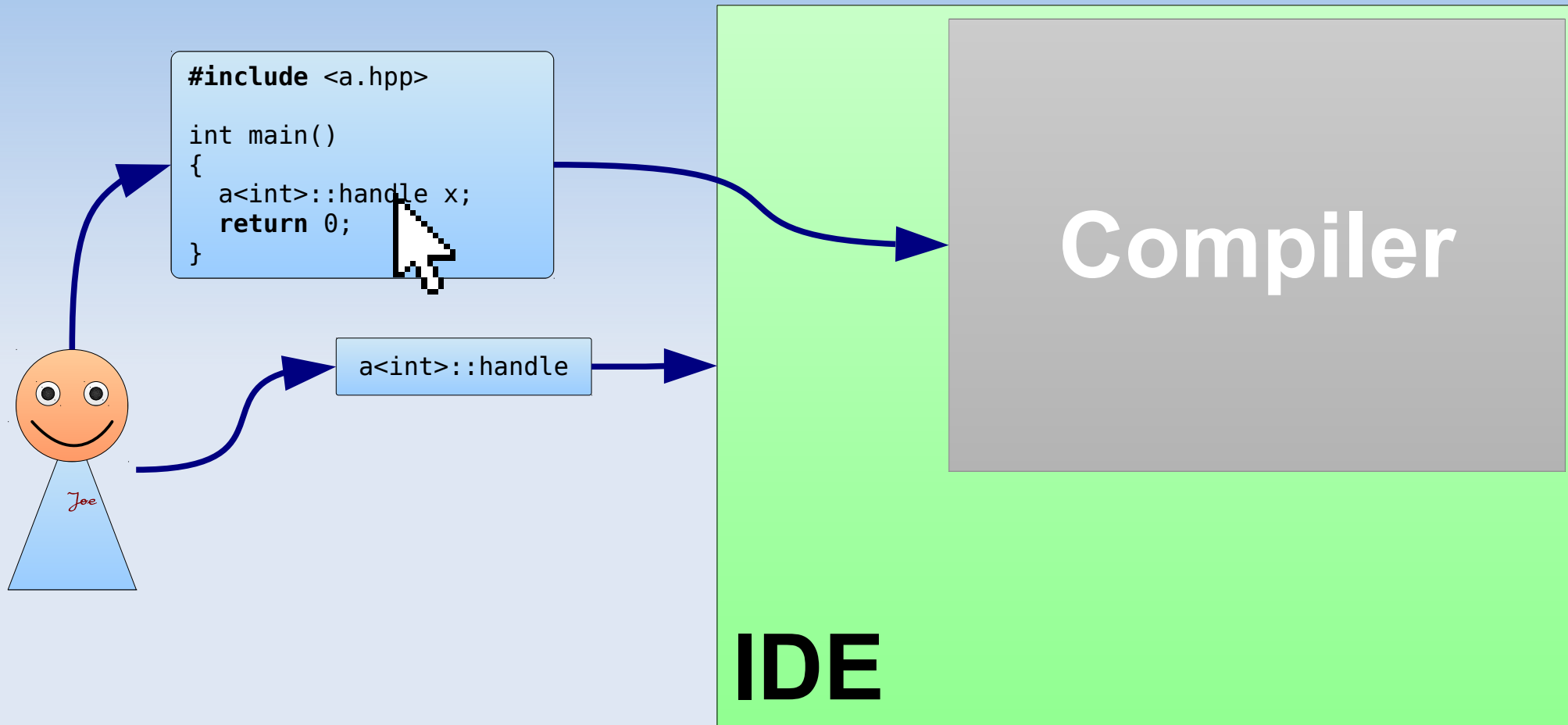
# IDEs



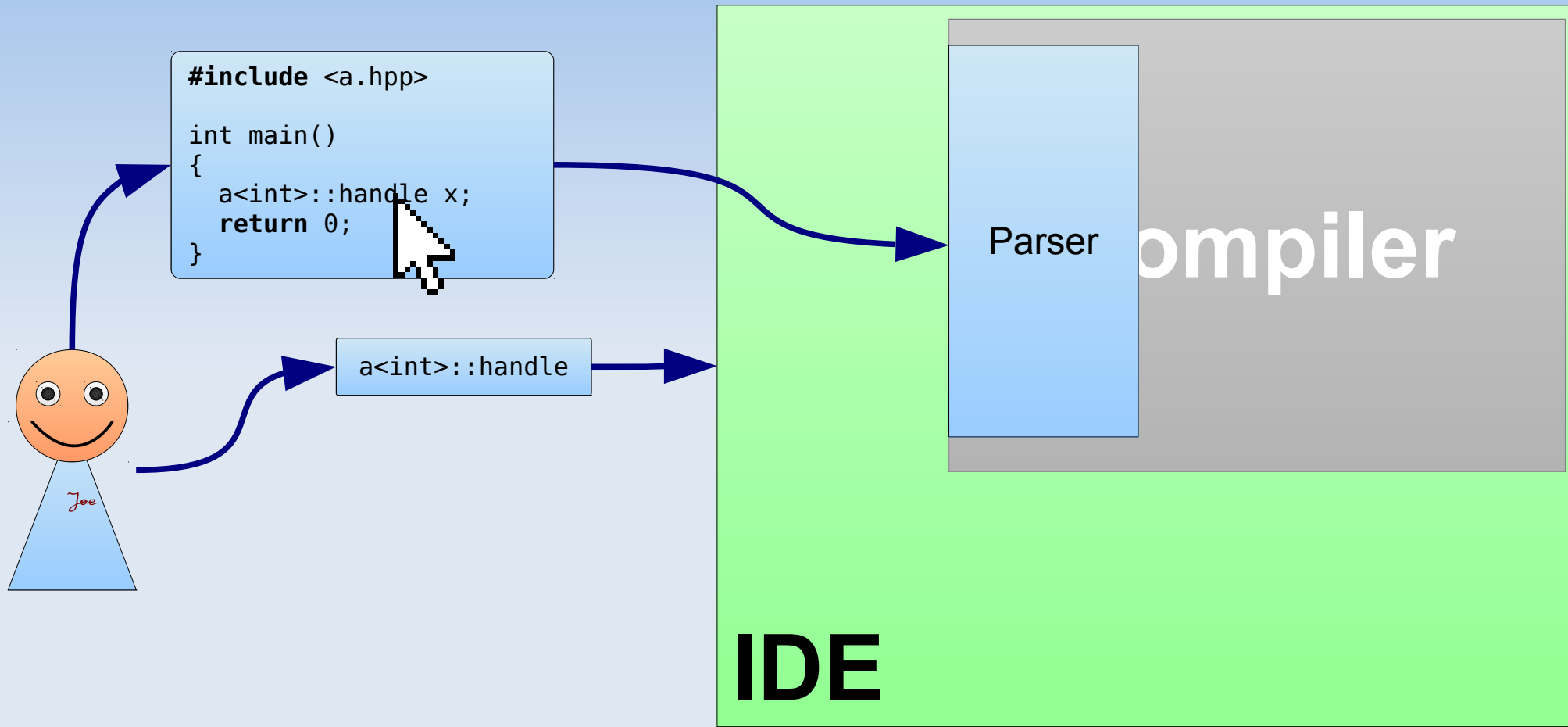
# IDEs



# IDEs

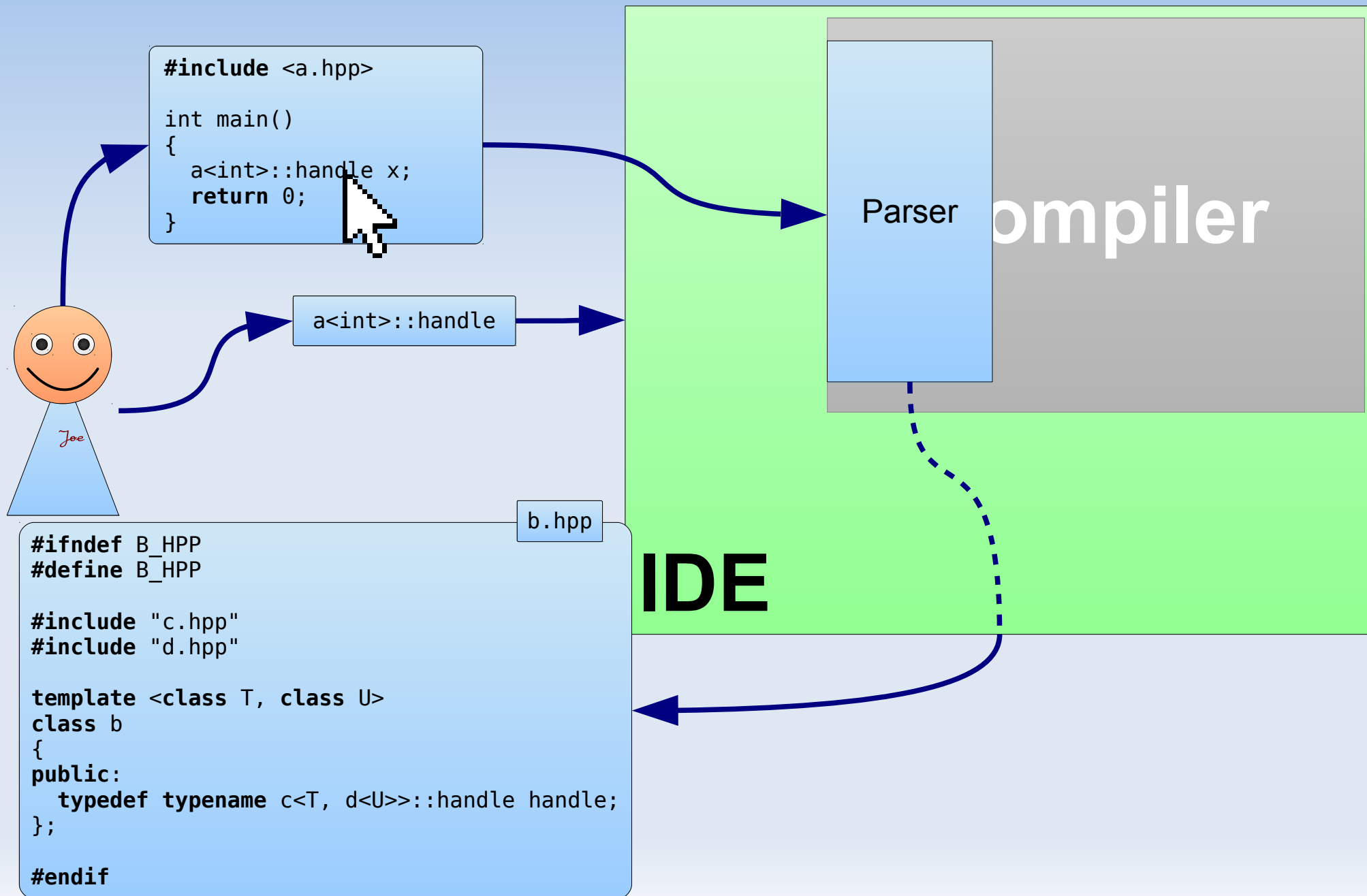


# IDEs

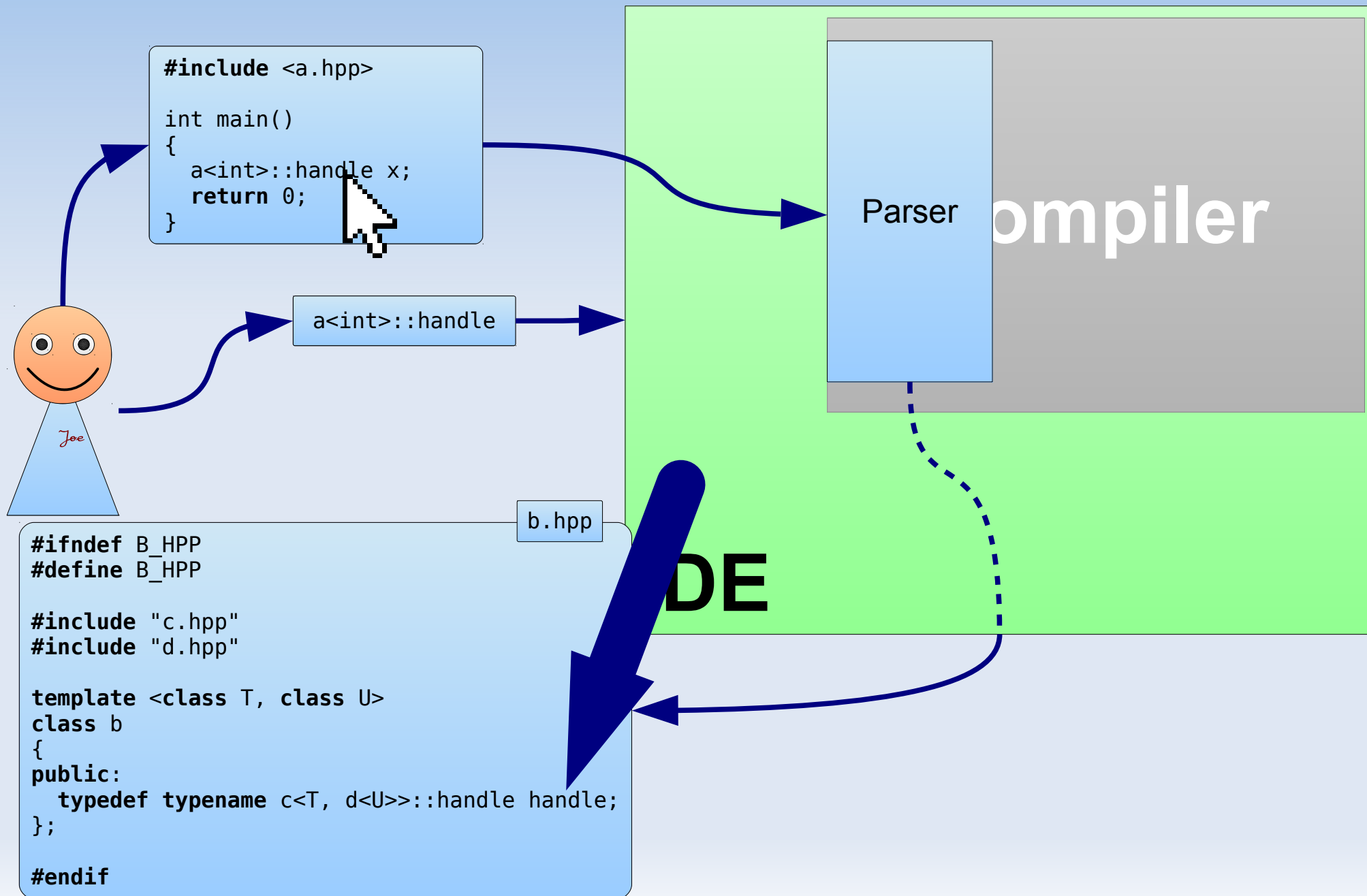




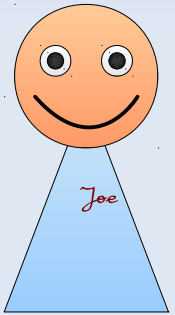
# IDEs



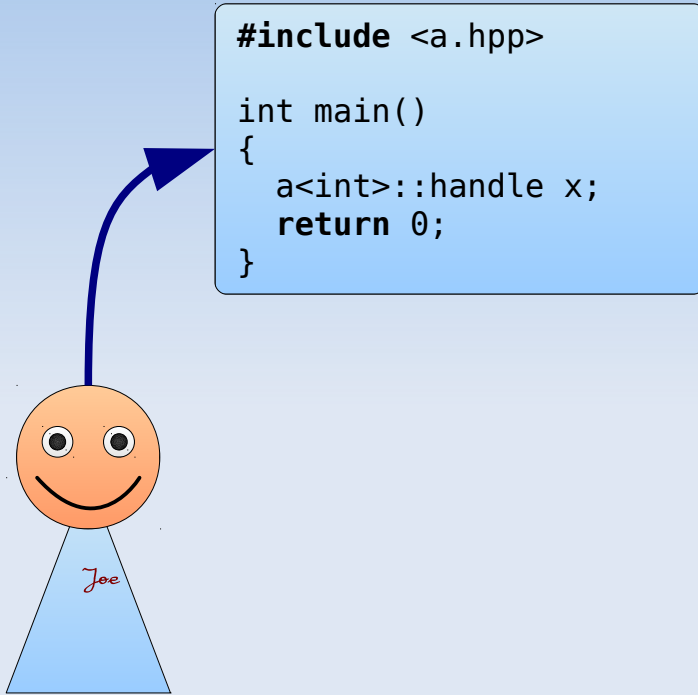
# IDEs



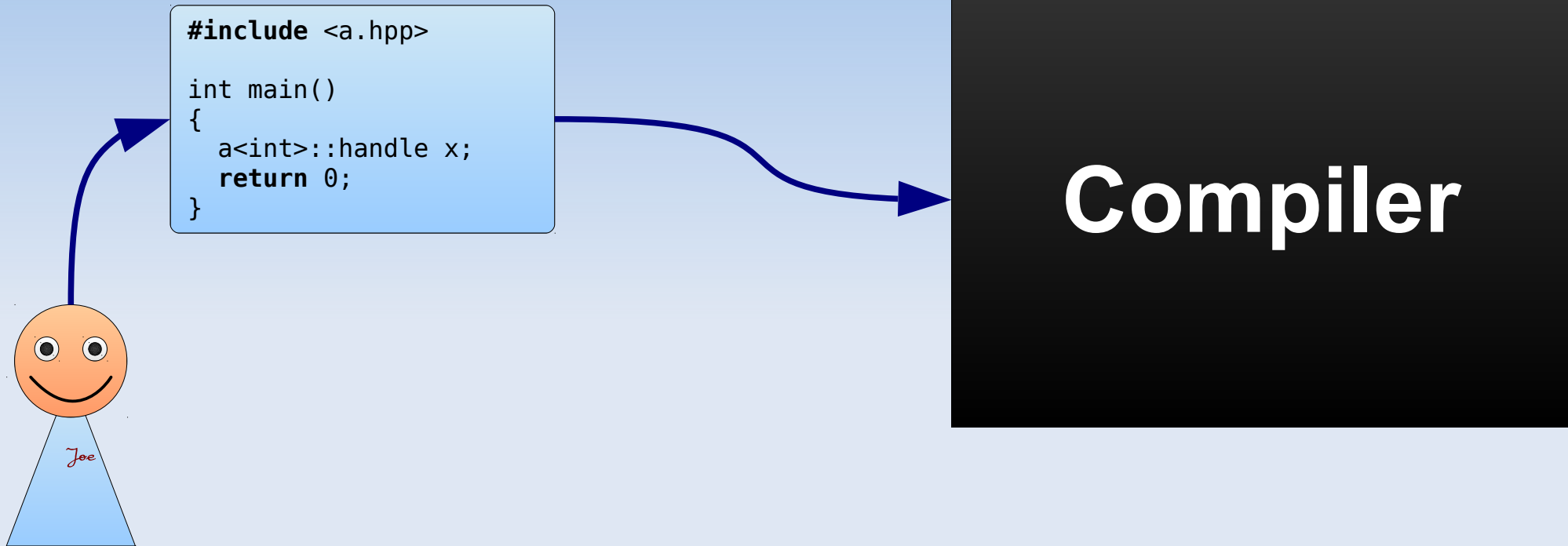
# GDB



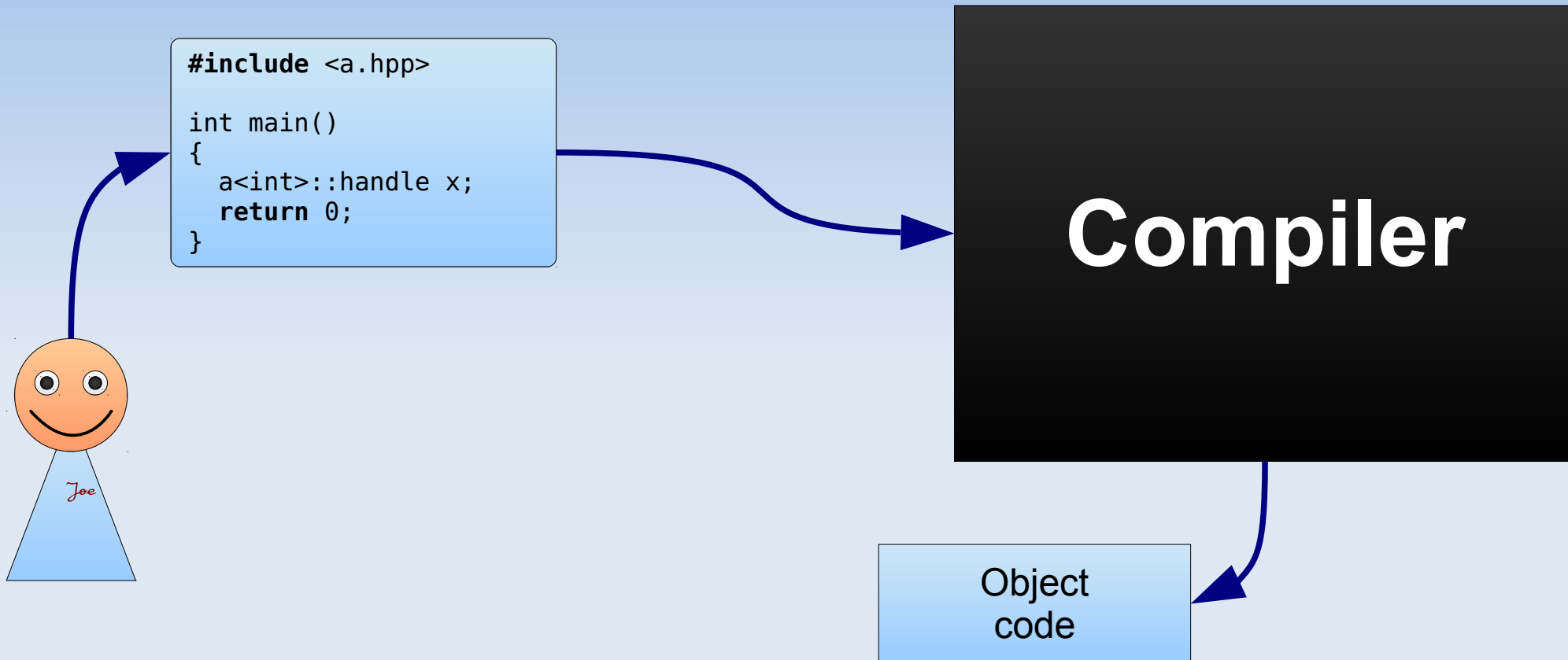
# GDB



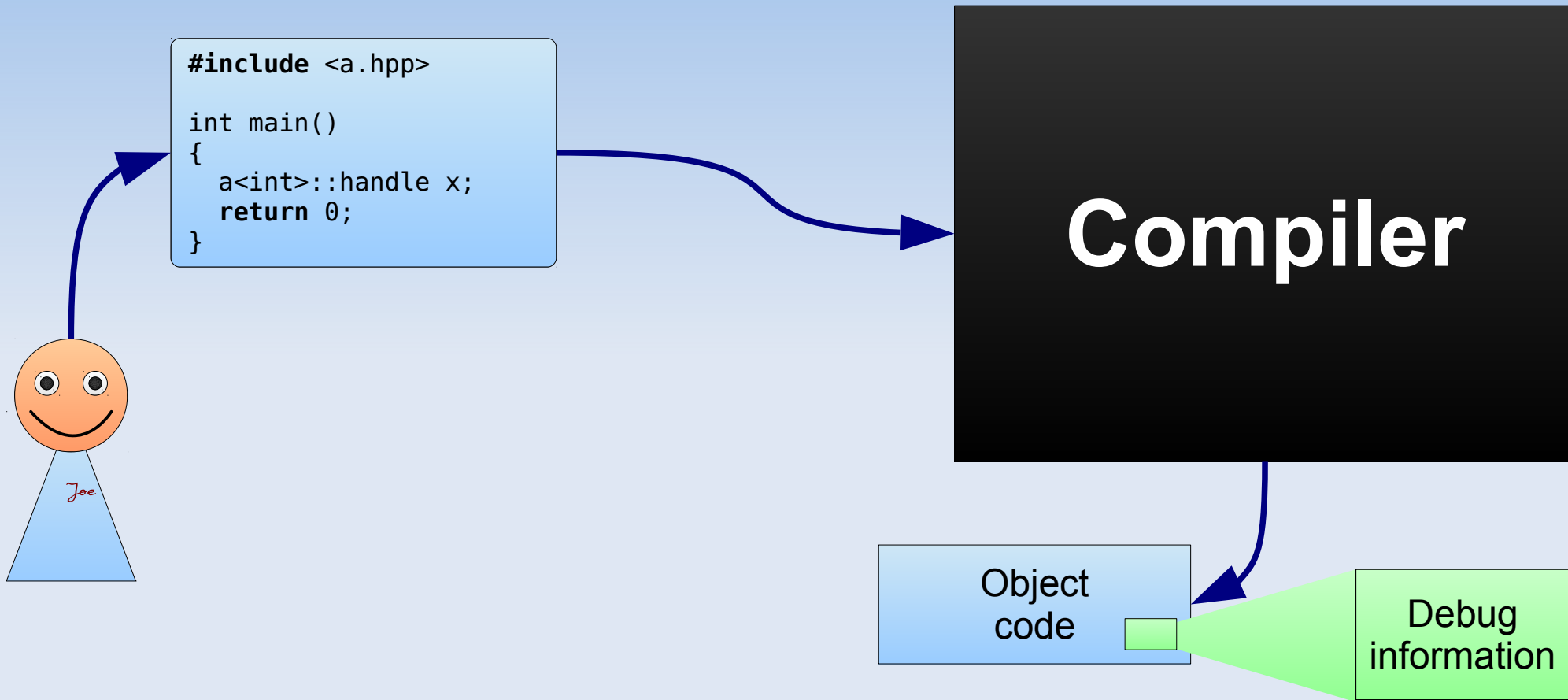
# GDB



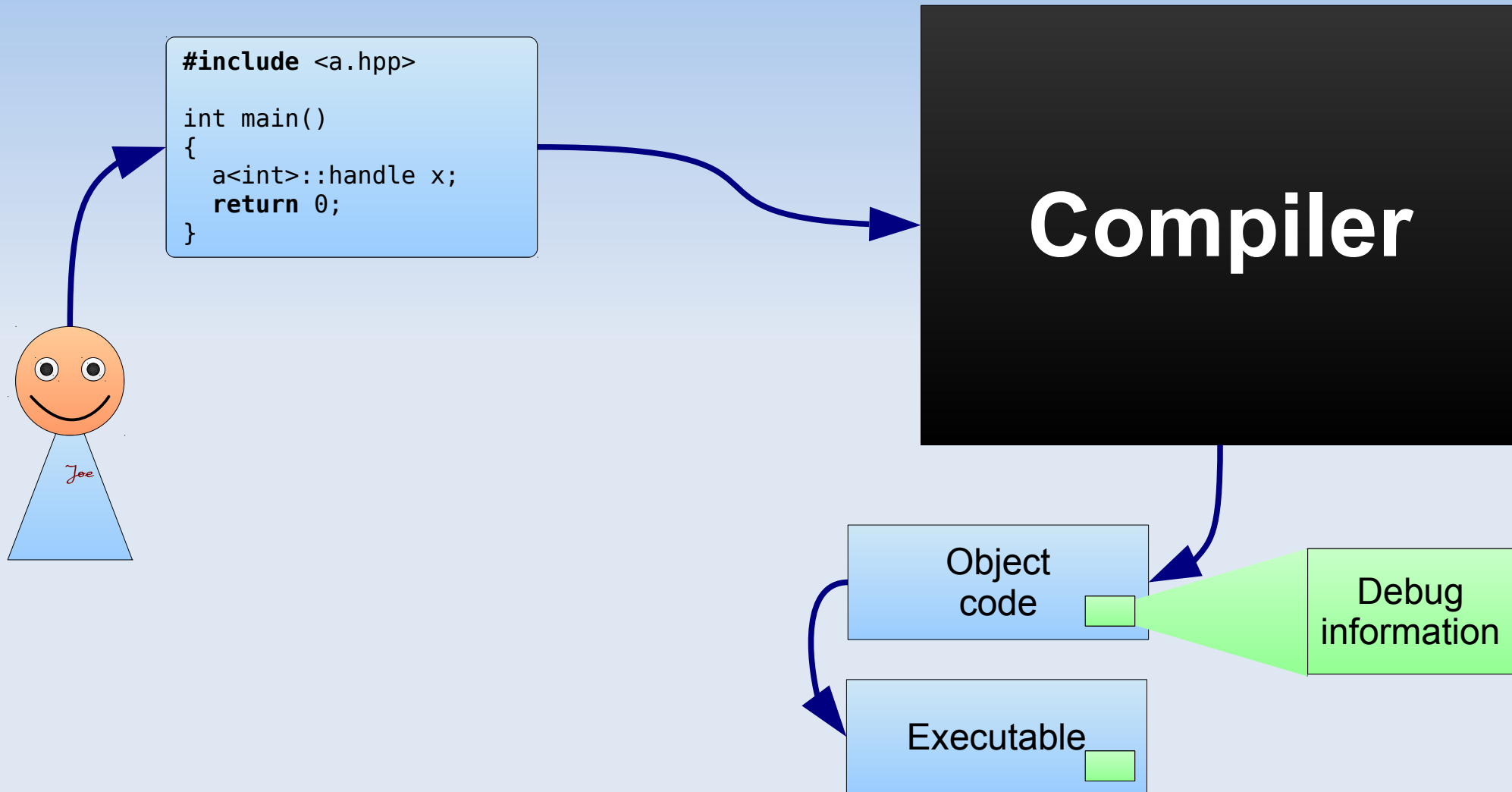
# GDB



# GDB

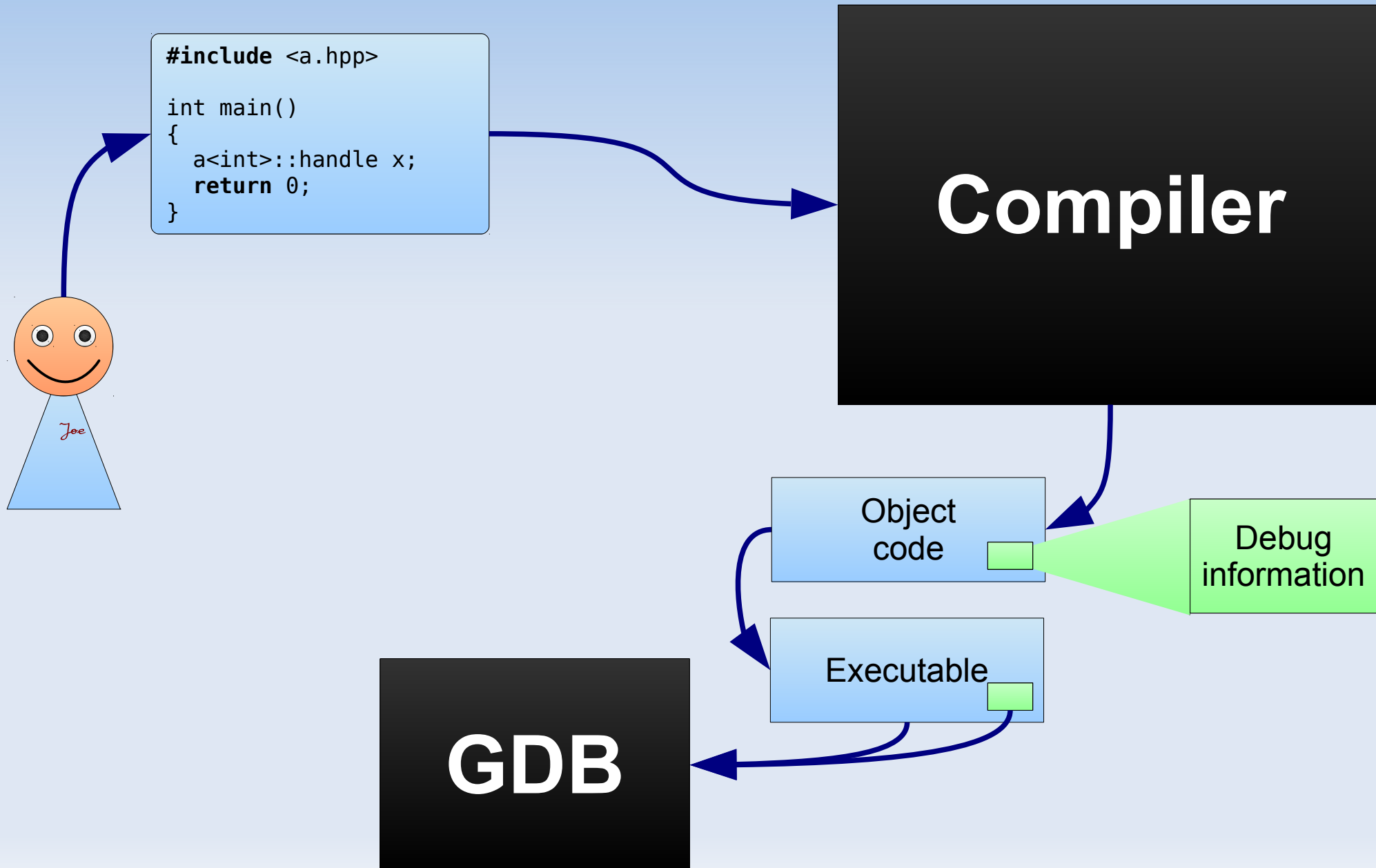


# GDB

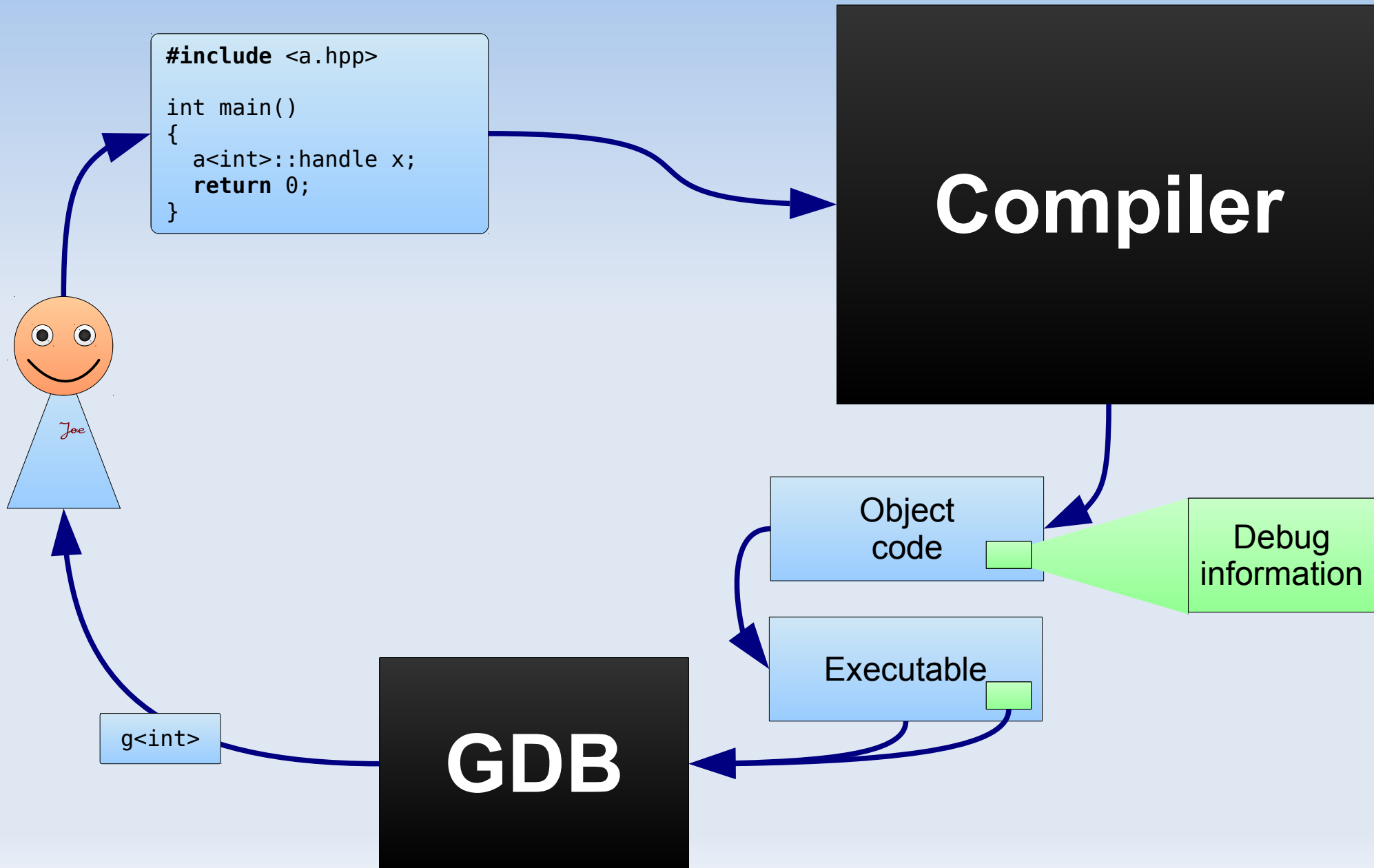




# GDB



# GDB



# GDB

```
$ g++ main1.cpp -g -std=c++11  
$
```

# GDB

```
$ g++ main1.cpp -g -std=c++11  
$ gdb a.out
```

# GDB

```
$ g++ main1.cpp -g -std=c++11  
$ gdb a.out  
...  
(gdb)
```

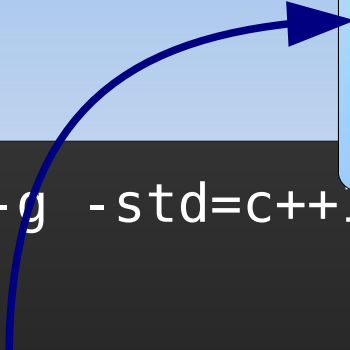
# GDB

```
$ g++ main1.cpp -g -std=c++11  
$ gdb a.out  
...  
(gdb) break main1.cpp:5
```

# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```

```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
...
(gdb) break main1.cpp:5
```



# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```

```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
```

...

```
(gdb) break main1.cpp:5
```

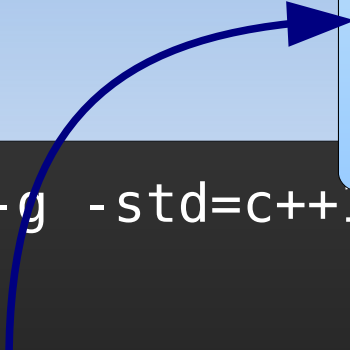
```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb)
```



# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```



```
$ g++ main1.cpp -g -std=c++11
```

```
$ gdb a.out
```

```
...
```

```
(gdb) break main1.cpp:5
```

```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb) run
```

# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```

```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
```

...

```
(gdb) break main1.cpp:5
```

```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb) run
```

```
Starting program: a.out
```

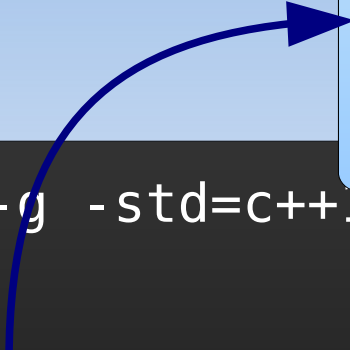
```
Breakpoint 1, main () at main1.cpp:6
```

```
6         return 0;
```

```
(gdb)
```

# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```



```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
```

```
...
```

```
(gdb) break main1.cpp:5
```

```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb) run
```

```
Starting program: a.out
```

```
Breakpoint 1, main () at main1.cpp:6
```

```
6         return 0;
```

```
(gdb) ptype x
```

# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```

```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
```

...

```
(gdb) break main1.cpp:5
```

```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb) run
```

```
Starting program: a.out
```

```
Breakpoint 1, main () at main1.cpp:6
```

```
6         return 0;
```

```
(gdb) ptype x
```

```
type = class g<int> [with T = int] {
```

```
    public:
```

```
        void foo(void);
```

```
}
```

# GDB

```
1 #include <a.hpp>
2
3 int main()
4 {
5     a<int>::handle x;
6     return 0;
7 }
```

```
$ g++ main1.cpp -g -std=c++11
$ gdb a.out
```

...

```
(gdb) break main1.cpp:5
```

```
Breakpoint 1 at 0x4006d1: file main1.cpp, line 5.
```

```
(gdb) run
```

```
Starting program: a.out
```

```
Breakpoint 1, main () at main1.cpp:6
```

```
6         return 0;
```

```
(gdb) ptype x
```

```
type = class g<int> [with T = int] {
```

```
    public:
```

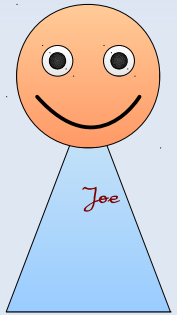
```
        void foo(void);
```

```
}
```

# Metashell

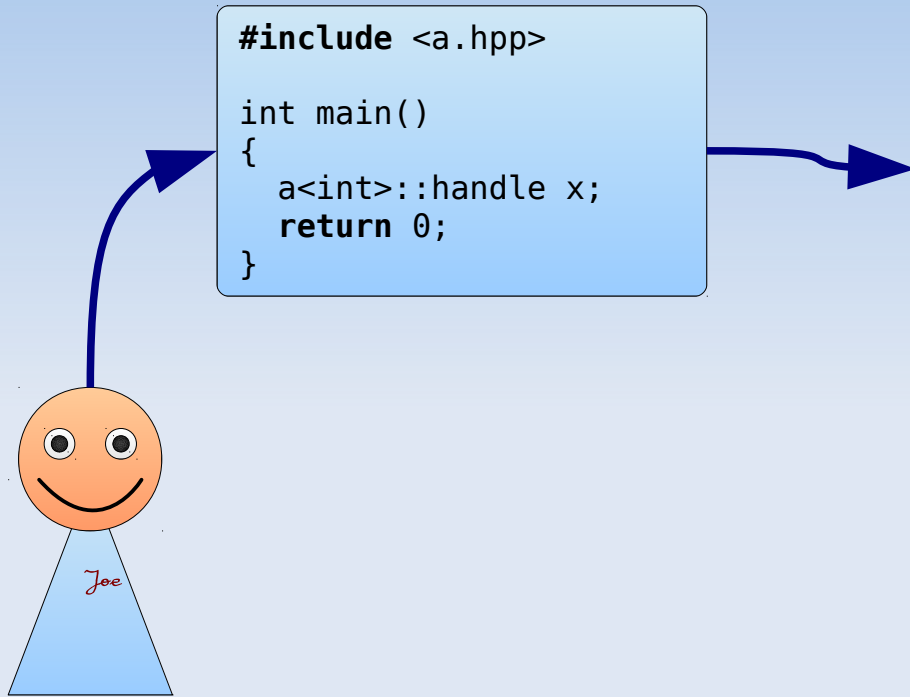
- Template metaprogramming shell & debugger
- Designed for uncovering the template instantiation details
- Not a "template metaprogrammer-only" shell
- <http://github.com/sabel83/metashell>

# Metashell



**Metashell**

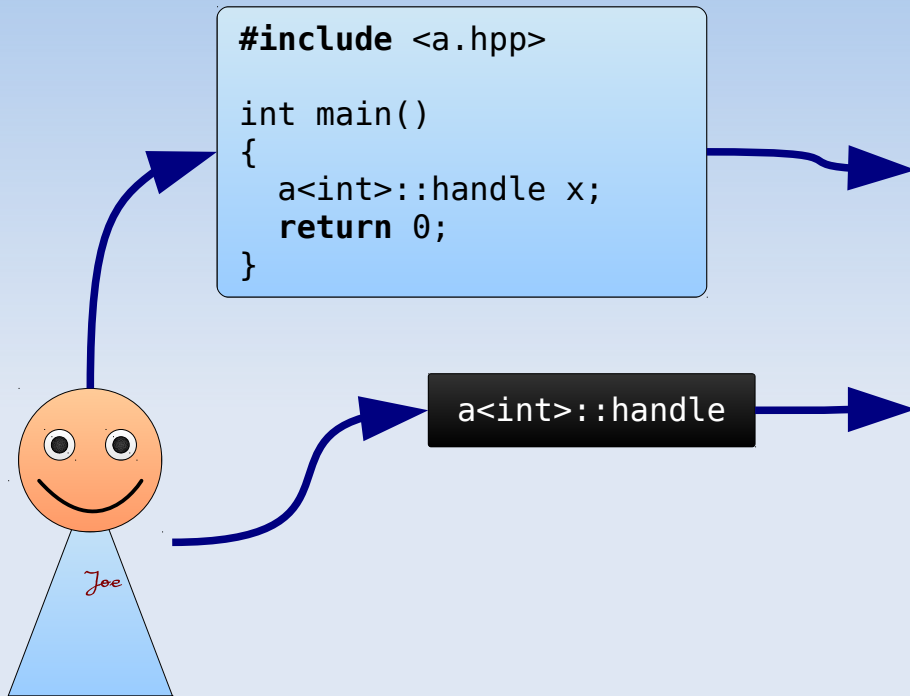
# Metashell



**Metashell**

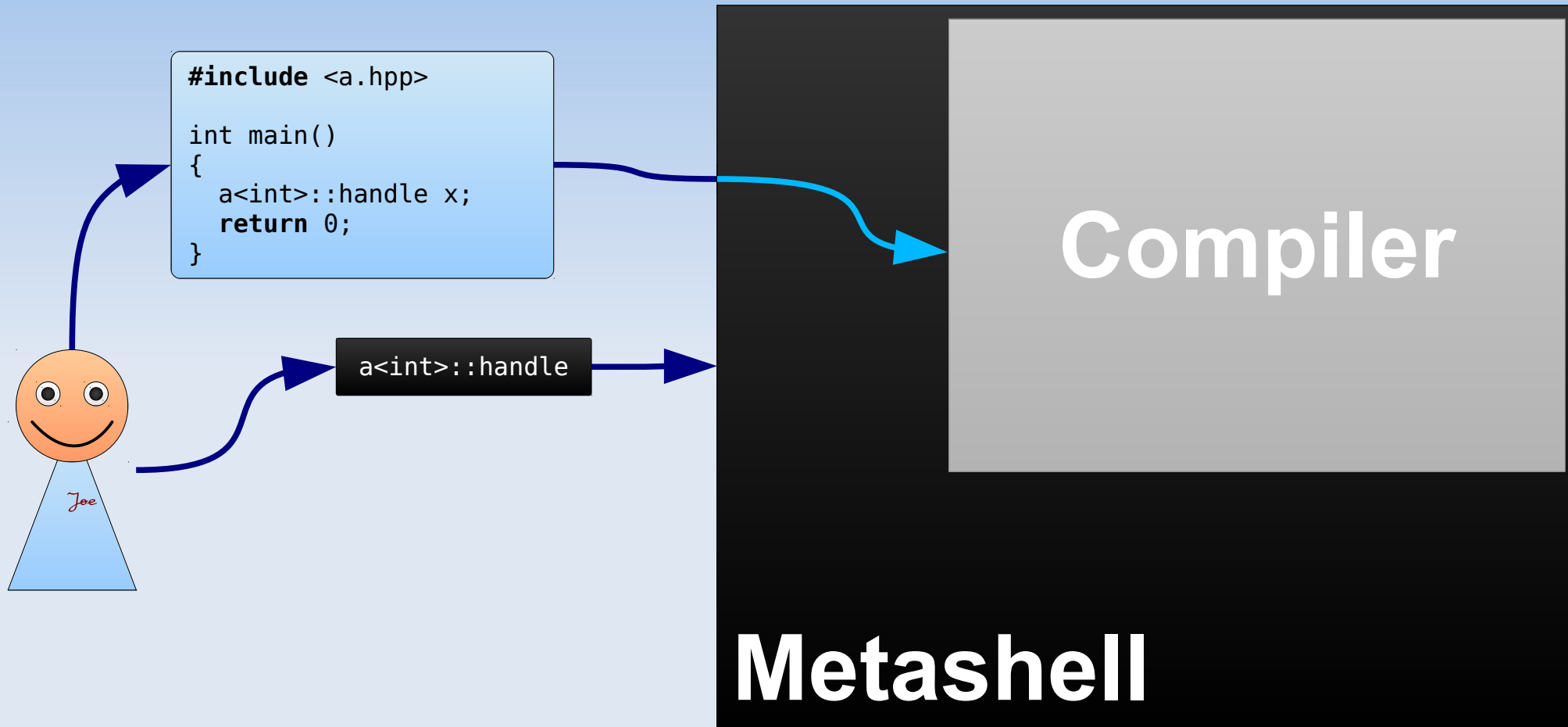


# Metashell

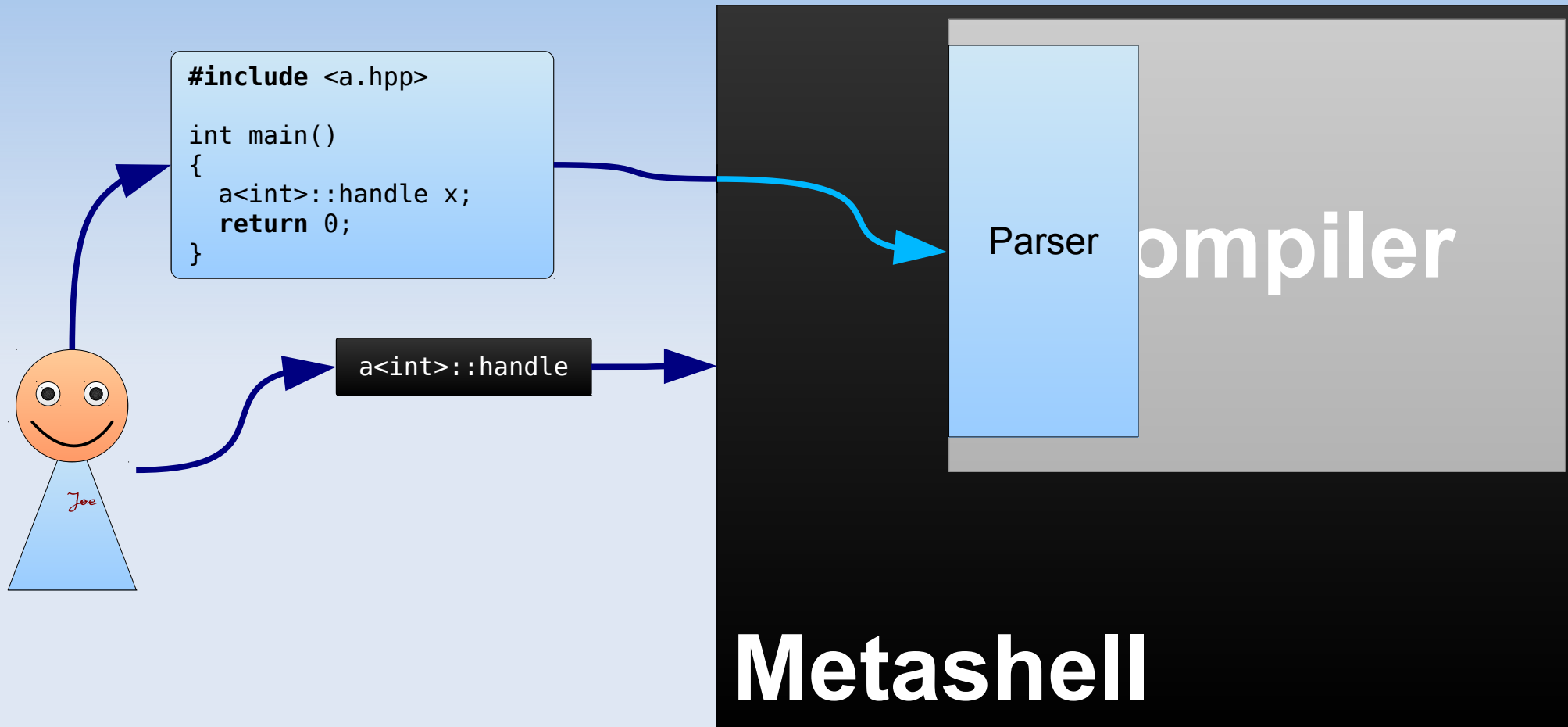


Metashell

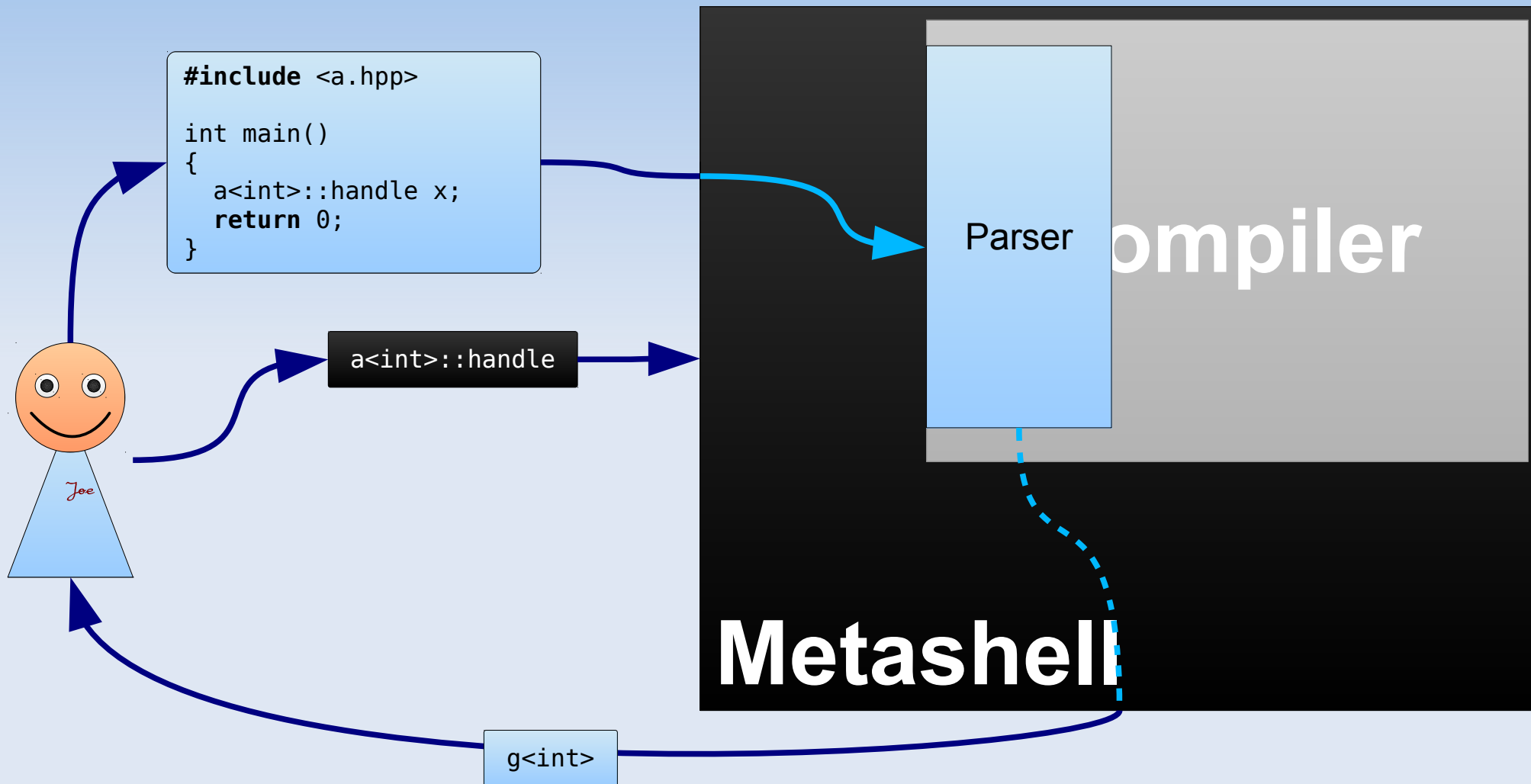
# Metashell



# Metashell



# Metashell



# Metashell

```
$ metashell  
>
```

# Metashell

```
$ metashell  
> #include "main1.cpp"  
>
```

# Metashell

```
$ metashell  
> #include "main1.cpp"  
> a<int>::handle
```

# Metashell

```
$ metashell  
> #include "main1.cpp"  
> a<int>::handle  
g<int>
```



# What is the type of...? #2

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}

int main()
{
}
```

# What is the type of...? #2

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}

int main()
{
    fun<double>();
}
```

# What is the type of...? #2

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}

int main()
{
    fun<double>();
}
```

# Approaches

- Enforced error message
- Displaying the name at runtime
- IDEs
- Debuggers
- Metaprogrammer tools

# boost::mpl::print

```
template <class T>
void fun()
{

    typename a<T>::handle h;
    h.foo();
}
```

# boost::mpl::print

```
template <class T>
void fun()
{
    boost::mpl::print< typename a<T>::handle > t;
    typename a<T>::handle h;
    h.foo();
}
```

# boost::mpl::print

```
template <class T>
void fun()
{
    boost::mpl::print< typename a<T>::handle > t;
    typename a<T>::handle h;
    h.foo();
}
```

Clang

```
In file included from main2_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~~
main2_err.cpp:8:46: note: in instantiation of template class
    'boost::mpl::print<g<int> >' requested here
    boost::mpl::print< typename a<T>::handle > t;
                                      ^
main2_err.cpp:15:3: note: in instantiation of function template
    specialization 'fun<double>' requested here
    fun<double>();
    ^
1 warning generated.
```

# boost::mpl::print

```
template <class T>
void fun()
{
    boost::mpl::print< typename a<T>::handle > t;
    typename a<T>::handle h;
    h.foo();
}
```

Clang

```
In file included from main2_err.cpp:3:
boost/mpl/print.hpp:50:23: warning: division by zero is undefined
    [-Wdivision-by-zero]
    const int m_x = 1 / (sizeof(T) - sizeof(T));
                        ^ ~~~~~~
main2_err.cpp:8:46: note: in instantiation of template class
    'boost::mpl::print<g<int>>' requested here
    boost::mpl::print< typename a<T>::handle > t;
                                ^
main2_err.cpp:15:3: note: in instantiation of function template
    specialization 'fun<double>' requested here
    fun<double>();
    ^
1 warning generated.
```



# boost::mpl::print

```
template <class T>
void fun()
{
    boost::mpl::print< typename a<T>::handle > t;
    typename a<T>::handle h;
    h.foo();
}
```

Clang

In file included from main2\_err.cpp:3:  
boost/mpl/print.hpp:50:23: warning: division by zero is undefined  
[-Wdivision-by-zero]

Visual C++

```
boost\mpl\print.hpp(52): error C4308: negative integral constant
converted to unsigned type
    main.cpp(8) : see reference to class template instanti
ation 'boost::mpl::print<g<T>>' being compiled
    with
    [
        T=int
    ]
    main.cpp(15) : see reference to function template inst
antiation 'void fun<double>(void)' being compiled
```

# boost::mpl::print

```
template <class T>
void fun()
{
    boost::mpl::print< typename a<T>::handle > t;
    typename a<T>::handle h;
    h.foo();
}
```

Clang

In file included from main2\_err.cpp:3:  
boost/mpl/print.hpp:50:23: warning: division by zero is undefined  
[-Wdivision-by-zero]

Visual C++

```
boost\mpl\print.hpp(52): error C4308: negative integral constant
converted to unsigned type
    main.cpp(8) : see reference to class template instanti
ation 'boost::mpl::print<g<T>>' being compiled
    with
    [
        T=int
    ]
    main.cpp(15) : see reference to function template inst
antiation 'void fun<double>(void)' being compiled
```

# metamonad::fail\_with\_type

```
template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}
```

# metamonad::fail\_with\_type

```
template <class T>
void fun()
{
    mpllibs::metamonad::fail_with_type< typename a<T>::handle >();
    typename a<T>::handle h;
    h.foo();
}
```

# metamonad::fail\_with\_type

```
template <class T>
```

GCC

```
void fun()
```

In file included from metamonad/fail\_with\_type.hpp:9:0,

from main2\_err\_mpllibs.cpp:3:

metamonad/v1/fail\_with\_type.hpp: In instantiation of 'void mpllibs::

metamonad::v1::fail\_with\_type() [with T = g<int>]':

main2\_err\_mpllibs.cpp:8:63: required from 'void fun() [with T =  
double]'

main2\_err\_mpllibs.cpp:15:15: required from here

metamonad/v1/fail\_with\_type.hpp:26:70: error: 'f' is not a member of  
'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_

\_\_\_\_\_<g<int> >'

impl::FAIL\_WITH\_TYPE\_\_\_\_\_<T>::f();  
^

# metamonad::fail\_with\_type

```
template <class T>
```

GCC

```
void fun()

In file included from metamonad/fail_with_type.hpp:9:0,
                 from main2_err_mpllibs.cpp:3:
metamonad/v1/fail_with_type.hpp: In instantiation of 'void mpllibs::
metamonad::v1::fail_with_type() [with T = g<int>]':
main2_err_mpllibs.cpp:8:63:   required from 'void fun() [with T =
double]'
main2_err_mpllibs.cpp:15:15:   required from here
metamonad/v1/fail_with_type.hpp:26:70: error: 'f' is not a member of
'mpllibs::metamonad::v1::impl::FAIL_WITH_TYPE_____
_____g<int> >'
impl::FAIL_WITH_TYPE_____<T>::f();
                                     ^
```

# metamonad::fail\_with\_type

```
template <class T>
```

```
void fun()
```

GCC

Clang

In file included from metamonad/fail\_with\_type.hpp:9:0,

In file included from main2\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

metamonad/v1/fail\_with\_type.hpp:26:68: error: no member

named 'f' in 'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_  
\_\_\_\_\_<g<int> >'

Impl::FAIL\_WITH\_TYPE\_\_\_\_\_<T>::f();  
~~~~~^

main2\_err\_mpllibs.cpp:8:23: note: in instantiation of function template  
specialization 'mpllibs::metamonad::v1::fail\_with\_type<g<int> >'  
requested here

mpllibs::metamonad::fail\_with\_type< typename a<T>::handle >();  
^

main2\_err\_mpllibs.cpp:15:3: note: in instantiation of function template  
specialization 'fun<double>' requested here

fun<double>();  
^

1 error generated.

# metamonad::fail\_with\_type

```
template <class T>
```

```
void fun()
```

GCC

Clang

In file included from metamonad/fail\_with\_type.hpp:9:0,

In file included from main2\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

metamonad/v1/fail\_with\_type.hpp:26:68: error: no member

named 'f' in 'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_

<g<int> >'

Impl::FAIL\_WITH\_TYPE\_\_\_\_\_<T>::f();

~~~~~^

main2\_err\_mpllibs.cpp:8:23: note: in instantiation of function template

specialization 'mpllibs::metamonad::v1::fail\_with\_type<g<int> >'

requested here

mpllibs::metamonad::fail\_with\_type< typename a<T>::handle >();

^

main2\_err\_mpllibs.cpp:15:3: note: in instantiation of function template

specialization 'fun<double>' requested here

fun<double>();

^

1 error generated.



# metamonad::fail\_with\_type

```
template <class T>
```

```
void fun()
```

In file included from metamonad/fail\_with\_type.hpp:9:0,

In file included from main2\_err\_mpllibs.cpp:3:

In file included from metamonad/fail\_with\_type.hpp:9:

metamonad\v1\fail\_with\_type.hpp(26): error C2039: 'f' : is not a member of 'mpllibs::metamonad::v1::impl::FAIL\_WITH\_TYPE\_\_\_\_\_

<T>'

with

[

T=g<int>

]

main.cpp(8) : see reference to function template instantiation  
n 'void mpllibs::metamonad::v1::fail\_with\_type<g<T>>(void)' being compiled

with

[

T=int

]

main.cpp(15) : see reference to function template instantiation  
on  
'void fun<double>(void)' being compiled

GCC

Clang

Visual C++

# metamonad::fail\_with\_type

```
template <class T>
```

```
void fun()
```

```
In file included from metamonad/fail_with_type.hpp:9:0,
```

```
In file included from main2_err_mpllibs.cpp:3:
```

```
In file included from metamonad/fail_with_type.hpp:9:
```

```
metamonad\v1\fail_with_type.hpp(26): error C2039: 'f' : is not a member  
of 'mpllibs::metamonad::v1::impl::FAIL_WITH_TYPE_____
```

```
_____  
<T>'
```

```
with
```

```
[
```

```
T=g<int>
```

```
]
```

```
main.cpp(8) : see reference to function template instantiatio  
n 'void mpllibs::metamonad::v1::fail_with_type<g<T>>(void)' being compi  
led
```

```
with
```

```
[
```

```
T=int
```

```
]
```

```
main.cpp(15) : see reference to function template instantiati  
on  
'void fun<double>(void)' being compiled
```

GCC

Clang

Visual C++

# IDEs

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}


int main()
{
    fun<double>();
}
```

# IDEs

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}

int main()
{
    fun<double>();
}
```

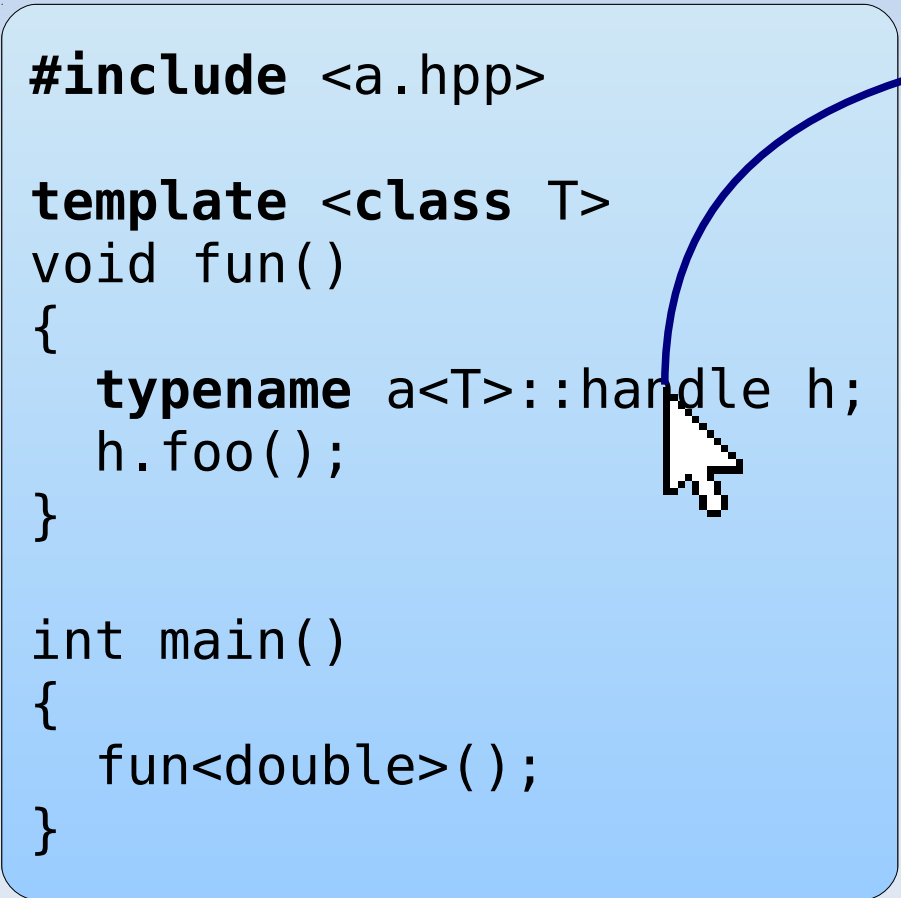


# IDEs

```
#include <a.hpp>

template <class T>
void fun()
{
    typename a<T>::handle h;
    h.foo();
}

int main()
{
    fun<double>();
}
```



```
void fun<int>()
{
    a<int>::handle h;
    h.foo();
}
```

# IDEs

```
#include <a.hpp>
```

```
template <class T>
```

```
void fun()
```

```
{
```

```
    typename a<T>::handle h;
```

```
    h.foo();
```

```
}
```

```
int main()
```

```
{
```

```
    fun<double>();
```

```
}
```

```
void fun<int>()
```

```
{
```

```
    a<int>::handle h;
```

```
    h.foo();
```

```
}
```

```
void fun<double>()
```

```
{
```

```
    a<double>::handle h;
```

```
    h.foo();
```

```
}
```

# IDEs

```
#include <a.hpp>
```

```
template <class T>
```

```
void fun()
```

```
{
```

```
    typename a<T>::handle h;  
    h.foo();
```

```
}
```

```
int main()
```

```
{
```

```
    fun<double>();
```

```
}
```

```
void fun<int>()
```

```
{
```

```
    a<int>::handle h;  
    h.foo();
```

```
}
```

```
void fun<double>()
```

```
{
```

```
    a<double>::handle h;  
    h.foo();
```

```
}
```

```
void fun<bar>()
```

```
{
```

```
    a<bar>::handle h;  
    h.foo();
```

```
}
```

# IDEs

```
#include <a.hpp>
```

```
template <class T>
```

```
void fun()
```

```
{
```

```
    typename a<T>::handle h;
```

```
    h.foo();
```

```
}
```

```
int main()
```

```
{
```

```
    fun<double>();
```

```
}
```

```
void fun<int>()
```

```
{
```

```
    a<int>::handle h;
```

```
    h.foo();
```

```
}
```

```
void fun<double>()
```

```
{
```

```
    a<double>::handle h;
```

```
    h.foo();
```

```
}
```

```
void fun<bar>()
```

```
{
```

```
    a<bar>::handle h;
```

```
    h.foo();
```

```
}
```



# GDB

```
$ g++ main2.cpp -g -std=c++11  
$
```

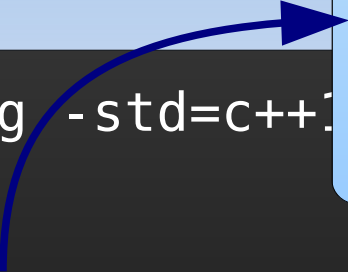
# GDB

```
$ g++ main2.cpp -g -std=c++11  
$ gdb a.out  
...  
(gdb)
```

# GDB

```
$ g++ main2.cpp -g -std=c++11  
$ gdb a.out  
...  
(gdb) break main2.cpp:6
```

```
1 #include <a.hpp>  
2  
3 template <class T>  
4 void fun()  
5 {  
6     typename a<T>::handle h;  
7     h.foo();  
8 }
```



# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb)
```

# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

...

```
(gdb) break main2.cpp:6
```

```
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
```

```
(gdb) run
```

# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb) run
Starting program: a.out

Breakpoint 1, fun<double> () at main2.cpp:7
7         h.foo();
(gdb)
```

# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb) run
Starting program: a.out
```

```
Breakpoint 1, fun<double>() at main2.cpp:7
7         h.foo();
(gdb)
```

# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb) run
Starting program: a.out
```

```
Breakpoint 1, fun<double>() at main2.cpp:7
7         h.foo();
(gdb) ptype h
```



# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb) run
Starting program: a.out
```

```
Breakpoint 1, fun<double>() at main2.cpp:7
7         h.foo();
(gdb) ptype h
type = class g<int> [with T = int] {
    public:
        void foo(void);
}
```

# GDB

```
1 #include <a.hpp>
2
3 template <class T>
4 void fun()
5 {
6     typename a<T>::handle h;
7     h.foo();
8 }
```

```
$ g++ main2.cpp -g -std=c++11
$ gdb a.out
```

```
...
(gdb) break main2.cpp:6
Breakpoint 1 at 0x4008c7: file main2.cpp, line 6.
(gdb) run
Starting program: a.out
```

```
Breakpoint 1, fun<double>() at main2.cpp:7
```

```
7         h.foo();
```

```
(gdb) ptype h
```

```
type = class g<int> [with T = int] {
```

```
    public:
```

```
        void foo(void);
```

```
}
```

# Metashell

```
$ metashell  
>
```

# Metashell

```
$ metashell  
> #include "main2.cpp"  
>
```

# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle
```

# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle  
g<int>  
>
```

# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle  
g<int>  
>
```

*Why?*



# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle  
g<int>  
> #msh mdb a<double>::handle
```

*Why?*





# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle  
g<int>  
> #msh mdb a<double>::handle  
For help, type "help".  
Metaprogram started  
  
(mdb)
```

*Why?*



# Metashell

```
$ metashell  
> #include "main2.cpp"  
> a<double>::handle  
g<int>  
> #msh mdb a<double>::handle  
For help, type "help".  
Metaprogram started  
  
(mdb) ft
```

*Why?*



# Metashell

```
$ metashell
> #include "main2.cpp"
> a<double>::handle
g<int>
> #msh mdb a<double>::handle
For help, type "help".
Metaprogram started
```

*Why?*



```
(mdb) ft
a<double>::handle
+ a<double> (TemplateInstantiation from <stdin>:2:26)
|   b<double, int> (TemplateInstantiation from ./a.hpp:7:18)
|     c<double, d<int> > (TemplateInstantiation from ./b.hpp:11:20)
|       + d<int> (TemplateInstantiation from ./c.hpp:10:39)
|         |   e<int> (TemplateInstantiation from ./d.hpp:10:20)
|         |     c_factory<f<int> > (TemplateInstantiation from ./c.hpp:10:20)
|         |       f<int> (TemplateInstantiation from ./c_factory.hpp:8:20)
|         |     g<int> (TemplateInstantiation from <stdin>:2:46)
```

# What is the type of...? #3

```
template <class T>
int f(T&& ref)
{
    return 0;
}
```

# What is the type of...? #3

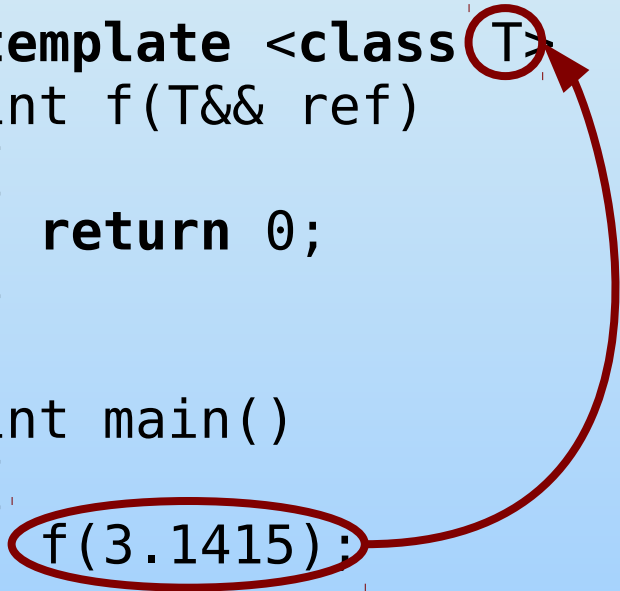
```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
}
```

# What is the type of...? #3

```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
}
```

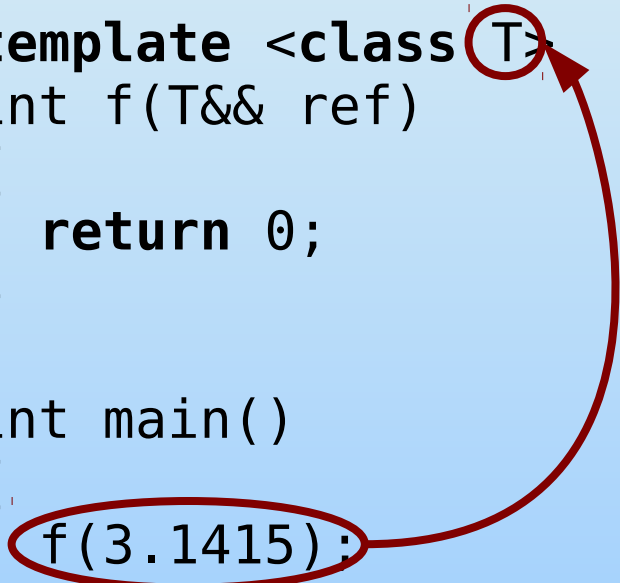


A diagram illustrating type deduction. A red circle highlights the variable `T` in the template parameter list `template <class T>`. Another red circle highlights the argument `3.1415` in the function call `f(3.1415);`. A red curved arrow points from the `3.1415` argument to the `T` parameter, indicating that the type of `T` is deduced from the argument.

# What is the type of...? #3

```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
    double d = 1.0;
    f(d);
}
```



# What is the type of...? #3

```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
    double d = 1.0;
    f(d);
}
```

The diagram illustrates the process of deducing the type `T` for the function `f`. In the `main` function, two calls to `f` are shown: `f(3.1415);` and `f(d);`. The value `3.1415` is a `double` literal, and `d` is a `double` variable. Both calls are circled in red. Red arrows point from these circled values to the `T` in the template parameter `<class T>`, which is also circled in red, indicating that `T` is deduced to be `double` based on these calls.

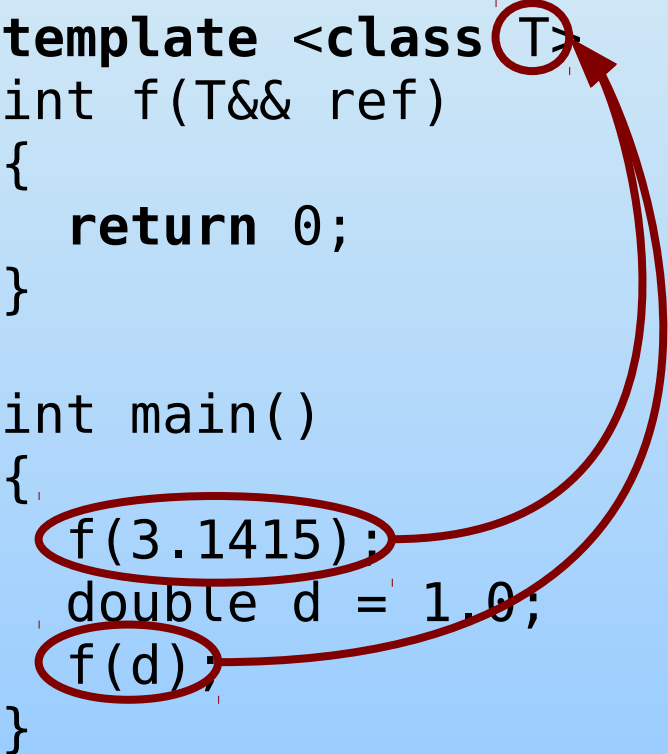


# What is the type of...? #3

Deduced types

```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
    double d = 1.0;
    f(d);
}
```



# What is the type of...? #3

Deduced types

```
template <class T>
int f(T&& ref)
{
    return 0;
}

int main()
{
    f(3.1415);
    double d = 1.0;
    f(d);
}
```

Scott Meyers

Effective Modern C++: 42 Specific Ways To Improve Your Use of C++11 and C++14

Item 4

# MDB

```
$ metashell
```

# MDB

```
$ metashell  
/* ... */  
>
```

# MDB

```
$ metashell  
/* ... */  
> template <class T> int f(T&& ref) { return 0; }  
>
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
```



# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
>
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
>
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
> #msh mdb decltype(f(d))
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
> #msh mdb decltype(f(d))
For help, type "help".
Metaprogram started
(mdb)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
> #msh mdb decltype(f(d))
For help, type "help".
Metaprogram started
(mdb) ft
```



# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
> #msh mdb decltype(f(d))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(d))
+ f<double &> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:43)
```

# MDB

```
$ metashell
/* ... */
> template <class T> int f(T&& ref) { return 0; }
> #msh mdb decltype(f(3.1415))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(3.1415))
+ f<double> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:48)
(mdb)
> double d = 1.0;
> #msh mdb decltype(f(d))
For help, type "help".
Metaprogram started
(mdb) ft
decltype(f(d))
+ f<double &> (TemplateInstantiation from <stdin>:2:35)
` int (NonTemplateType from <stdin>:2:43)
```

# Templight

- Clang extension/tool
- Logs template instantiation-related events
- Original: <http://plc.inf.elte.hu/templight/>
  - Fork: <https://github.com/mikael-s-persson/templight>
  - We will be using the fork

# Templight

```
$ templight++ -std=c++11 -c -Xtemplight -profiler -Xtemplight -safe-mode main3.cpp
```

# Templight


```
$ templight++ -std=c++11 -c -Xtemplight -profiler -Xtemplight -safe-mode main3.cpp
```



```
main3.o.trace.pbf
```

# Templight

```
$ templight++ -std=c++11 -c -Xtemplight -profiler -Xtemplight -safe-mode main3.cpp
```



main3.o.trace.pbf



```
$ templight-convert -f <type> -i main3.o.trace.pbf
```

# Templight

```
$ templight++ -std=c++11 -c -Xtemplight -profiler -Xtemplight -safe-mode main3.cpp
```

main3.o.trace.pbf

```
$ templight-convert -f <type> -i main3.o.trace.pbf
```

XML

Graphml

Graphviz

Callgrind

# Templight

```
$ templight++ -std=c++11 -c -Xtemplight -profiler -Xtemplight -safe-mode main3.cpp
```

main3.o.trace.pbf

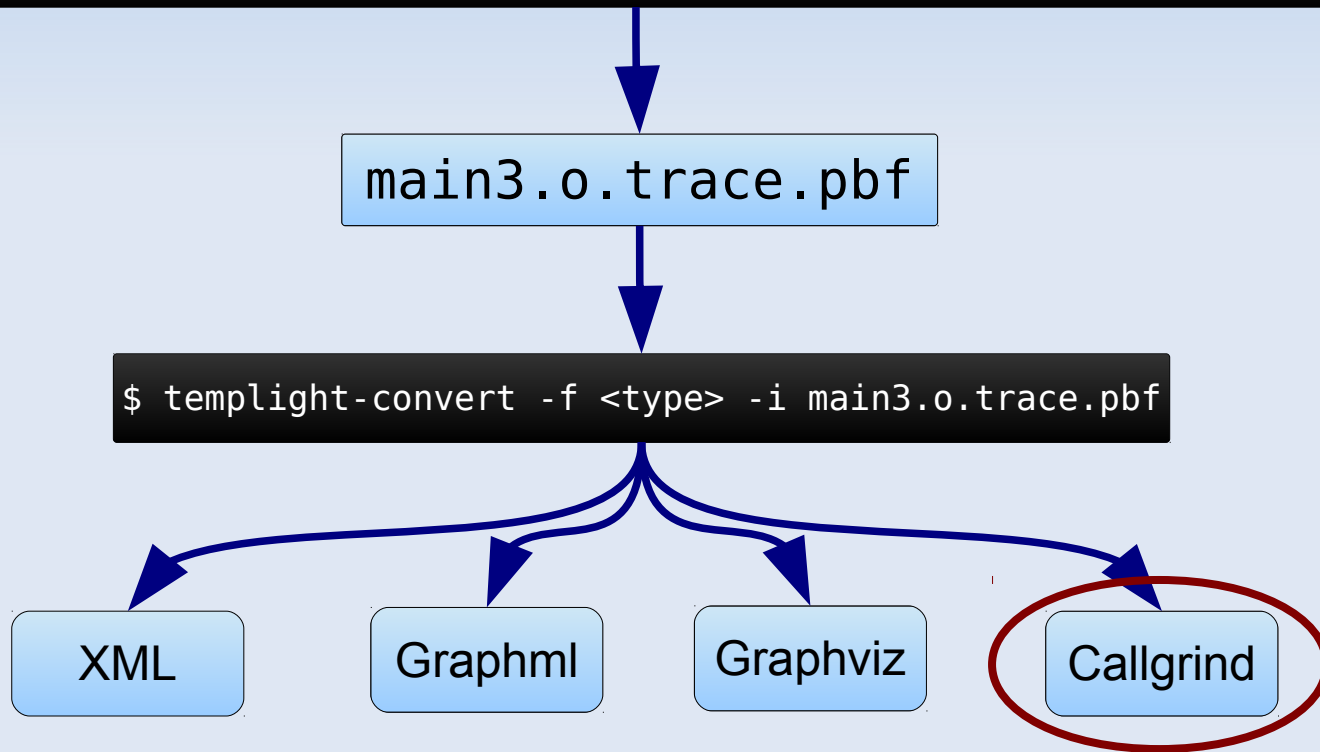
```
$ templight-convert -f <type> -i main3.o.trace.pbf
```

XML

Graphml

Graphviz

Callgrind





# KCacheGrind

- Generate a callgrind output from Templight
- Open it with KCacheGrind

global

| Types | Callers | All Callers                            | Callee Map | Source Code |
|-------|---------|--|------------|-------------|
| #     | CTime   | Source ('main3.cpp')                   |            |             |
| 8     |         |  |            |             |
| 9     |         | int main()                             |            |             |
| 10    |         | {                                      |            |             |
| 11    |         | f(3.1415);                             |            |             |
|       | 1.54    | 1 call(s) to 'f' (main3.cpp)           |            |             |
|       | 2.49    | 2 call(s) to 'f<double>' (main3.cpp)   |            |             |
| 12    |         | double d = 1.0;                        |            |             |
| 13    |         | f(d);                                  |            |             |
|       | 1.28    | 1 call(s) to 'f' (main3.cpp)           |            |             |
|       | 1.61    | 2 call(s) to 'f<double &>' (main3.cpp) |            |             |
| 14    |         | }                                      |            |             |
| 15    |         |  |            |             |
| 16    |         |  |            |             |

# KCacheGrind

- Generate a callgrind output from Templight
- Open it with KCacheGrind

global

| Types | Callers | All Callers                                 | Callee Map | Source Code |
|-------|---------|---|------------|-------------|
| #     | CTime   | Source ('main3.cpp')                        |            |             |
| 8     |         | int main()                                  |            |             |
| 9     |         | {   |            |             |
| 10    |         | f(3.1415);                                  |            |             |
| 11    |         | 1.54 1 call(s) to 'f' (main3.cpp)           |            |             |
|       |         | 2.49 2 call(s) to 'f<double>' (main3.cpp)   |            |             |
| 12    |         | double d = 1.0;                             |            |             |
| 13    |         | f(d);                                       |            |             |
|       |         | 1.28 1 call(s) to 'f' (main3.cpp)           |            |             |
|       |         | 1.61 2 call(s) to 'f<double &>' (main3.cpp) |            |             |
| 14    |         | }   |            |             |
| 15    |         |   |            |             |
| 16    |         |   |            |             |

The diagram shows two red arrows pointing from the source code to the callgrind output. The first arrow points from the line `f(3.1415);` to the callgrind output lines `1.54 1 call(s) to 'f' (main3.cpp)` and `2.49 2 call(s) to 'f<double>' (main3.cpp)`. The second arrow points from the line `f(d);` to the callgrind output lines `1.28 1 call(s) to 'f' (main3.cpp)` and `1.61 2 call(s) to 'f<double &>' (main3.cpp)`. The word *instantiates* is written in red next to each arrow.

# Understanding template instantiations

- What happens when you instantiate a template function?
- The body of the template function might trigger further template instantiations
- It is often useful to understand what happens

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
T sum(T t_) { return t_; }
```

```
template <class T, class... Ts>
```

```
typename std::common_type<T, Ts...>::type
```

```
sum(T t_, Ts... ts_)
```

```
{
```

```
    return t_ + sum(ts_...);
```

```
}
```

# Summarising numbers

sum.hpp

```
#include <type_traits>

template <class T>
T sum(T t_) { return t_; }

template <class T, class... Ts>
typename std::common_type<T, Ts...>::type
sum(T t_, Ts... ts_)
{
    return t_ + sum(ts_...);
}
```

sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell  
/* ... */  
>
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
```

```
/* ... */
```

```
> #include "sum.hpp"
```

```
>
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```



# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
```

```
/* ... */
```

```
> #include "sum.hpp"
```

```
> #include <metashell/instantiate_expression.hpp>
```

```
>
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
```

```
/* ... */
```

```
> #include "sum.hpp"
```

```
> #include <metashell/instantiate_expression.hpp>
```

```
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
/* ... */
> #include "sum.hpp"
> #include <metashell/instantiate_expression.hpp>
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
metashell::expression_instantiated<true>
>
```

sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
```

```
/* ... */
```

```
> #include "sum.hpp"
```

```
> #include <metashell/instantiate_expression.hpp>
```

```
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
```

```
metashell::expression_instantiated<true>
```

```
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
/* ... */
> #include "sum.hpp"
> #include <metashell/instantiate_expression.hpp>
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
metashell::expression_instantiated<true>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
For help, type "help".
Metaprogram started
(mdb)
```

```
sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
/* ... */
> #include "sum.hpp"
> #include <metashell/instantiate_expression.hpp>
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
metashell::expression_instantiated<true>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
For help, type "help".
Metaprogram started
(mdb) ft
```

sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

# Summarising numbers

sum.hpp

```
#include <type_traits>
```

```
template <class T>
```

```
$ metashell
/* ... */
> #include "sum.hpp"
> #include <metashell/instantiate_expression.hpp>
> METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
metashell::expression_instantiated<true>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
For help, type "help".
Metaprogram started
(mdb) ft
METASHELL_INSTANTIATE_EXPRESSION( sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) )
+ sum<int, int, int, int, int, int, int, int, int, int> (TemplateInstantiation from <stdin>:2:26)
| ` sum<int, int, int, int, int, int, int, int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | ` sum<int, int, int, int, int, int, int, int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | ` sum<int, int, int, int, int, int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | ` sum<int, int, int, int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | | ` sum<int, int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | | | ` sum<int, int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | | | | ` sum<int, int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | | | | | ` sum<int> (TemplateInstantiation from ./sum.hpp:12:15)
| | | | | | | | | metashell::expression_instantiated<true> (TemplateInstantiation from <stdin>:2:99)
```

sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

# An STL example

- What gets instantiated when you create an empty `std::string`?



# An STL example

- What gets instantiated when you create an empty `std::string`?

```
>
```

# An STL example

- What gets instantiated when you create an empty `std::string`?

```
> #include <metashell/instantiate_expression.hpp>  
>
```

# An STL example

- What gets instantiated when you create an empty `std::string`?

```
> #include <metashell/instantiate_expression.hpp>
> #include <string>
>
```

# An STL example

- What gets instantiated when you create an empty `std::string`?

```
> #include <metashell/instantiate_expression.hpp>
> #include <string>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::string() )
```

# An STL example

- What gets instantiated when you create an empty `std::string`?

```
> #include <metashell/instantiate_expression.hpp>
> #include <string>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::string() )
For help, type "help".
Metaprogram started
(mdb)
```

# An STL example

- What gets instantiated when you create an empty `std::string`?

```
> #include <metashell/instantiate_expression.hpp>
> #include <string>
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::string() )
For help, type "help".
Metaprogram started
(mdb) ft
```

```

METASHELL_EXPRESSION_INSTANTIATED( std::string()
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from <stdin>:2:26)
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:114:24)
| + std::__allocator_base (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:92:29)
| | `__gnu_cxx::new_allocator<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:92:29)
+ std::allocator<char>::rebind<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:114:32)
+ std::char_traits<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:119:24)
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:121:24)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:289:28)
| + std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:272:29)
| | `std::allocator<char>::allocator<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:151:25)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:289:28)
+ std::char_traits<char> (Memoization from <stdin>:2:26)
+ std::allocator<char> (Memoization from <stdin>:2:26)
+ metashell::expression_instantiated<true> (TemplateInstantiation from <stdin>:2:78)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::basic_string (TemplateInstantiation from <stdin>:2:26)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:508:26)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:508:9)
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:272:14)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:511:7)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:511:30)
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:155:2)
| | 1)
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep_base (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:155:21)
| | | `std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:150:2)
| | 2)
| + std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:158:19)
| + std::allocator<char>::rebind<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:158:27)
| | `std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:173:15)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_M_data (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:511:8)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:293:17)
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:293:28)
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:297:17)
| | `std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:297:28)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:21)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:35)
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:50)
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:9)
| `std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:9)

```

```
+ METASHELL_EXPRESSION_INSTANTIATED( std::string )  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from <stdin>:2:26)  
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:114:24)  
| + std::__allocator_base (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:92:29)  
| `__gnu_cxx::new_allocator<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:92:29)  
+ std::allocator<char>::rebind<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:114:32)  
+ std::char_traits<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:119:24)  
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:121:24)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bit  
s/basic_string.h:289:28)  
| + std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:272:29)  
| `std::allocator<char>::allocator<char> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/allocator.h:151:25)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_st  
ring.h:289:28)  
+ std::char_traits<char> (Memoization from <stdin>:2:26)  
+ std::allocator<char> (Memoization from <stdin>:2:26)  
+ metashell::expression_instantiated<true> (TemplateInstantiation from <stdin>:2:78)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::basic_string (TemplateInstantiation from <stdin>:2:26)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:508:26)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_st  
ring.h:508:9)  
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:272:14)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:511:7)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:5  
11:30)  
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:155:2  
| 1)  
| + std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep_base (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_str  
ing.h:155:21)  
| | `std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:150  
| :2)  
| + std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:158:19)  
| + std::allocator<char>::rebind<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:158:27)  
| `std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:173:1  
5)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_M_data (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/bas  
ic_string.h:511:8)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:293:1  
7)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_  
string.h:293:28)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:297:1  
7)  
| `std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_  
string.h:297:28)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:21)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Rep (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:4  
39:35)  
+ std::allocator<char> (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:50)  
+ std::basic_string<char, std::char_traits<char>, std::allocator<char> > (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_string.h:439:9)  
| `std::basic_string<char, std::char_traits<char>, std::allocator<char> >::_Alloc_hider (Memoization from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/basic_st  
ring.h:439:9)
```



# Compilation speed

- Heavy template usage can lead to long compilation times
- To optimise it, we need to understand where this comes from

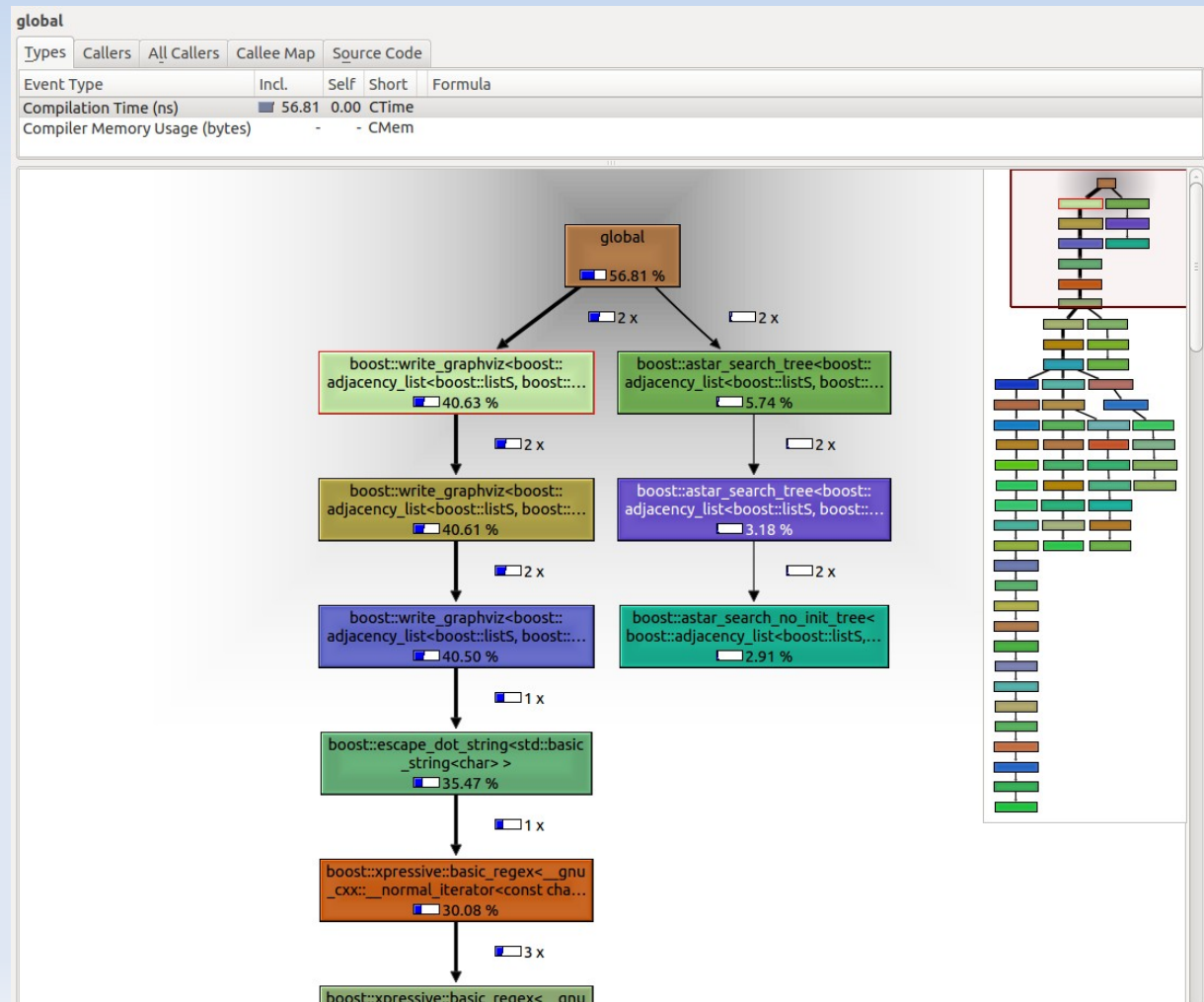
# Example code

- Boost.Graph example: astar-cities.cpp

```
$ time clang++ astar-cities.cpp  
  
real    0m3.726s  
user    0m3.297s  
sys     0m0.139s
```

# Templight + Callgrind

- KcacheGrind
  - Instantiation tree
  - Profiling



# Templight + Callgrind

global

| Types | Callers | All Callers  | Callee Map | Source Code |
|-------|---------|--|------------|-------------|
| #     | CTime   | Source ('astar-cities.cpp')  |            |             |
| 154   |         | unsigned int num_edges = sizeof(edge_array) / sizeof(edge);  |            |             |
| 155   |         | cost weights[] = { // estimated travel time (mins)   |            |             |
| ...   |         | ...  |            |             |
| 159   |         | ...  |            |             |
| 160   |         | ...  |            |             |
| 161   |         | // create graph  |            |             |
| 162   |         | mygraph_t g(N);  |            |             |
|       | 0.00    | 1 call(s) to 'boost::adjacency_list<boost::listS, boost::vecS, boost::undirectedS, boost::no_property, boost::property<boost::edge_weight_t, ...   |            |             |
|       | 0.00    | 1 call(s) to 'p' (adjacency_list.hpp)  |            |             |
|       | 0.74    | 1 call(s) to 'boost::adjacency_list<boost::listS, boost::vecS, boost::undirectedS, boost::no_property, boost::property<boost::edge_weight_t, ...   |            |             |
| 163   |         | WeightMap weightmap = get(edge_weight, g);   |            |             |
|       | 0.00    | 2 call(s) to 'boost::get' (reverse_graph.hpp)  |            |             |
|       | 0.00    | 1 call(s) to 'boost::property<boost::edge_weight_t, float, boost::no_property>' (property.hpp)   |            |             |
|       | 0.01    | 1 call(s) to 'boost::adj_list_edge_property_map<boost::undirected_tag, float, float &, unsigned long, boost::property<boost::edge_weight_t, ...    |            |             |
|       | 0.01    | 2 call(s) to 'boost::get' (adjacency_list.hpp)   |            |             |
|       | 0.05    | 3 call(s) to 'boost::get<boost::detail::adj_list_gen<boost::adjacency_list<boost::listS, boost::vecS, boost::undirectedS, boost::no_property, b... |            |             |
| 164   |         | for(std::size_t j = 0; j < num_edges; ++j) {   |            |             |
| 165   |         | edge_descriptor e; bool inserted;  |            |             |
|       | 0.00    | 1 call(s) to 'boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>' (edge.hpp)  |            |             |
|       | 0.00    | 1 call(s) to 'boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>::edge_desc_impl' (edge.hpp)                                      |            |             |
| 166   |         | boost::tie(e, inserted) = add_edge(edge_array[j].first,  |            |             |
|       | 0.00    | 1 call(s) to 'std::pair<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>, bool>::pair<boost::detail::edge_desc_impl<bo...       |            |             |
|       | 0.00    | 1 call(s) to 'boost::add_edge<boost::detail::adj_list_gen<boost::adjacency_list<boost::listS, boost::vecS, boost::undirectedS, boost::no_prop...   |            |             |
|       | 0.00    | 1 call(s) to 'std::pair<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>, bool>::pair' (stl_pair.h)                             |            |             |
|       | 0.00    | 1 call(s) to 'boost::tuples::tuple<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long> &, bool &, boost::tuples::null_type, b...   |            |             |
|       | 0.01    | 1 call(s) to 'std::pair<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>, bool>' (stl_pair.h)                                   |            |             |
|       | 0.01    | 2 call(s) to 'boost::add_edge' (adjacency_list.hpp)  |            |             |
|       | 0.01    | 1 call(s) to 'boost::tuples::tie' (tuple_basic.hpp)  |            |             |
|       | 0.02    | 2 call(s) to 'boost::tuples::tuple<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long> &, bool &, boost::tuples::null_type, b...   |            |             |
|       | 0.07    | 2 call(s) to 'boost::tuples::tie<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>, bool>' (tuple_basic.hpp)                     |            |             |
|       | 0.21    | 1 call(s) to 'boost::tuples::tuple<boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long> &, bool &, boost::tuples::null_type, b...   |            |             |
|       | 1.73    | 2 call(s) to 'boost::add_edge<boost::adjacency_list<boost::listS, boost::vecS, boost::undirectedS, boost::no_property, boost::property<boos...     |            |             |
| 167   |         | edge_array[j].second, g);  |            |             |
| 168   |         | weightmap[e] = weights[j];   |            |             |

# **“What gets instantiated?”**

# ”What gets instantiated?”

```
$ templight-convert -f text -i astar-cities.o.trace.pbf \  
  | grep 'Name =' | sed 's/^  Name = //' | sort --unique
```

# "What gets instantiated?"

```
$ templight-convert -f text -i astar-cities.o.trace.pbft \
  | grep 'Name =' | sed 's/^  Name = //' | sort --unique
```

```
. . .
boost::detail::deduce_source_char_impl<boost::detail::deduce_character_type_later<unsigned long> >
boost::detail::deduce_source_char_impl<deduce_character_type_later<type-parameter-0-0> >
boost::detail::deduce_source_char<unsigned long>
boost::detail::deduce_target_char_impl<char>
boost::detail::deduce_target_char<std::basic_string<char> >
boost::detail::dereference_iterator
boost::detail::digit_traits<int>
boost::detail::digit_traits<long>
boost::detail::digit_traits_select<true>
boost::detail::digit_traits_select<true>::traits<int>
boost::detail::digit_traits_select<true>::traits<long>
boost::detail::do_not_construct_out_stream_t
boost::detail::dummy_constructor
boost::detail::dummy_no_property_iterator
boost::detail::dummy_pmap_reference
boost::detail::dummy_property_copier
boost::detail::edge_base<boost::undirected_tag, unsigned long>
boost::detail::edge_base<boost::undirected_tag, unsigned long>::edge_base
boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>
boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>::edge_desc_impl
boost::detail::edge_desc_impl<boost::undirected_tag, unsigned long>::get_property
. . .
```

# When things go wrong

- So far we have assumed, that the code *compiles*
- When the code fails to compile, we need to *debug the compilation process*



# Example code

```
class person  
{  
    // ...  
};
```

# Example code

```
class person
{
    // ...
};

int main()
{
    std::vector<person> people;

}
```

# Example code

```
class person
{
    // ...
};

int main()
{
    std::vector<person> people;

    std::sort(people.begin(), people.end());
}
```

## THE MAIN()

{

}

```
std::sort(people.begin(), people.end());
```

}

## Example code

[illegible]

```
/usr/include/c++/4.8/bits/stl_algo.h:1935:11: error:
invalid operands to binary expression ('person' and 'person')
    if (*__i < *__first)
        ~~~~ ^ ~~~~~
```

}

# Example code

```
1926 /// This is a helper function for the sort routines.
1927 template<typename _RandomAccessIterator>
1928 void
1929 __heap_select(_RandomAccessIterator __first,
1930              _RandomAccessIterator __middle,
1931              _RandomAccessIterator __last)
1932 {
1933     std::make_heap(__first, __middle);
1934     for (_RandomAccessIterator __i = __middle; __i < __last; ++__i)
1935         if (*__i < *__first)
1936             std::__pop_heap(__first, __middle, __i);
1937 }
```

```
/usr/include/c++11.8/bits/stl_algo.h:1935:11: error:
invalid operands to binary expression ('person' and 'person')
    if (*__i < *__first)
        ~~~~ ^ ~~~~~
```

```
}
```

# How to see what went wrong

```
class person
{
    // ...
};

int main()
{
    std::vector<person> people;

    std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
```

```
};  
  
int main()  
{  
    std::vector<person> people;  
  
    std::sort(people.begin(), people.end());  
}
```



# How to see what went wrong

```
$ metashell  
/* ... */  
>
```

```
};  
  
int main()  
{  
    std::vector<person> people;  
  
    std::sort(people.begin(), people.end());  
}
```

# How to see what went wrong

```
$ metashell  
/* ... */  
> #include "person.cpp"
```

```
};
```

```
int main()
```

```
{
```

```
    std::vector<person> people;
```

```
    std::sort(people.begin(), people.end());
```

```
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
>
```

```
};

int main()
{
    std::vector<person> people;

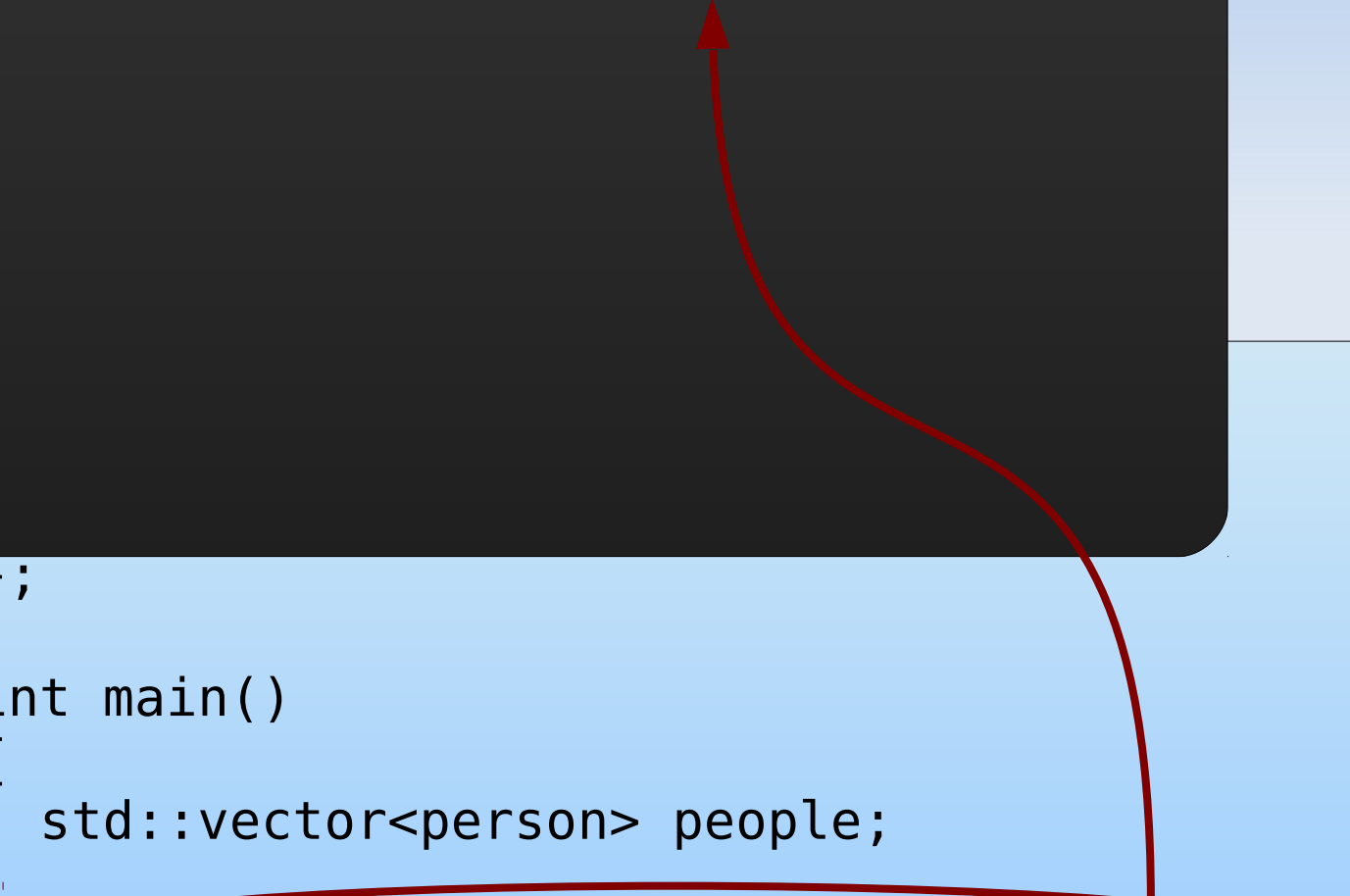
    std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
>
```

```
};

int main()
{
    std::vector<person> people;
    std::sort(people.begin(), people.end());
}
```



# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
>
```

```
};

int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
>
```

```
};

int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
>
```

```
};

int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
>
```

```
};
```

```
int main()
```

```
{
```

```
    std::vector<person> people;
```

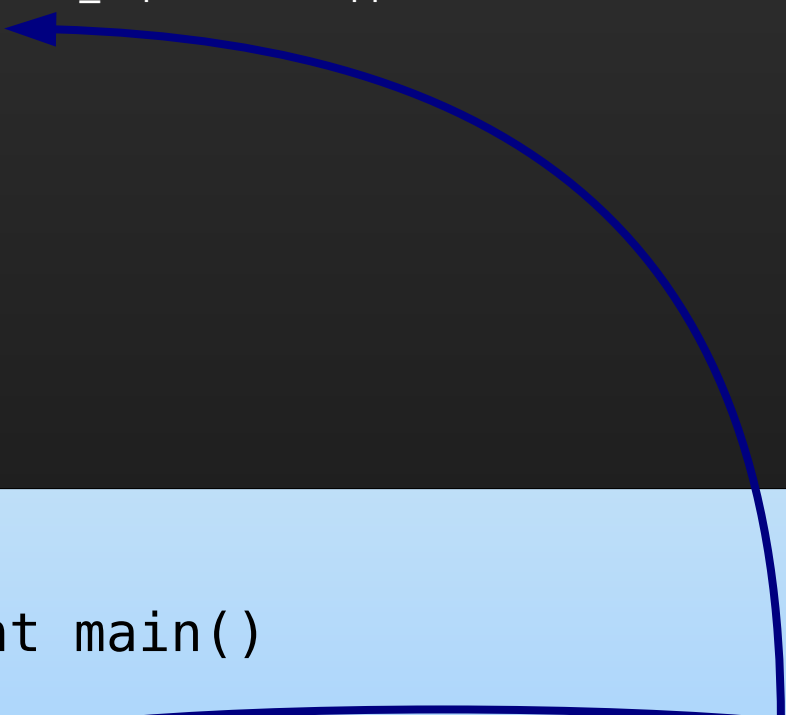
```
    // std::sort(people.begin(), people.end());
```

```
}
```



# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
>
```



```
};

int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
```

```
};
```

```
int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb)
```

```
};
```

```
int main()
{
    std::vector<person> people;

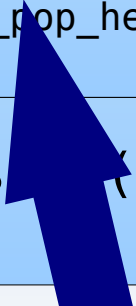
    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb)
```

```
1926 /// This is a helper function for the sort routines.
1927 template<typename __RandomAccessIterator>
1928     void
1929     __heap_select(__RandomAccessIterator __first,
1930                  __RandomAccessIterator __middle,
1931                  __RandomAccessIterator __last)
1932     {
1933         std::make_heap(__first, __middle);
1934         for (__RandomAccessIterator __i = __middle; __i < __last; ++__i)
1935             if (*__i < *__first)
1936                 std::__pop_heap(__first, __middle, __i);
1937     }
```

```
    // std::sort(people.begin(), people.end());
}
```

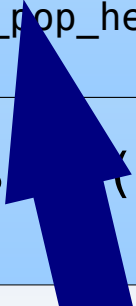


# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb)
```

```
1926 /// This is a helper function for the sort routines.
1927 template<typename __RandomAccessIterator>
1928     void
1929     __heap_select(__RandomAccessIterator __first,
1930                  __RandomAccessIterator __middle,
1931                  __RandomAccessIterator __last)
1932     {
1933         std::make_heap(__first, __middle);
1934         for (__RandomAccessIterator __i = __middle; __i < __last; ++__i)
1935             if (*__i < *__first)
1936                 std::__pop_heap(__first, __middle, __i);
1937     }
```

```
    // std::sort(people.begin(), people.end());
}
```



# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
```

```
1926 /// This is a helper function for the sort routines.
1927 template<typename __RandomAccessIterator>
1928     void
1929     __heap_select(__RandomAccessIterator __first,
1930                  __RandomAccessIterator __middle,
1931                  __RandomAccessIterator __last)
1932     {
1933         std::make_heap(__first, __middle);
1934         for (__RandomAccessIterator __i = __middle; __i < __last; ++__i)
1935             if (*__i < *__first)
1936                 std::__pop_heap(__first, __middle, __i);
1937     }
```

```
    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb)
```

```
};
```

```
int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb) c
```

```
};
```

```
int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```



# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb) c
Breakpoint "__heap_select" reached
std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::all
ocator<person> > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
(mdb)
```

```
};
```

```
int main()
```

```
{
```

```
    std::vector<person> people;
```

```
    // std::sort(people.begin(), people.end());
```

```
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb) c
Breakpoint "__heap_select" reached
std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
(mdb) bt
```

```
};
```

```
int main()
{
    std::vector<person> people;

    // std::sort(people.begin(), people.end());
}
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb) c
Breakpoint "__heap_select" reached
std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
(mdb) bt
#0 std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
#1 std::partial_sort<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:2310:8)
#2 std::__introsort_loop<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > >, long> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5451:4)
#3 std::sort<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from <stdin>:2:26)
#4 METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
```

# How to see what went wrong

```
$ metashell
/* ... */
> #include "person.cpp"
/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:1935:11:
fatal error: invalid operands to binary expression ('person' and 'person')
> #include "person.cpp"
> #include <metashell/instantiate_expression.hpp>
> std::vector<person> people;
> #msh mdb METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
For help, type "help".
Metaprogram started
(mdb) rbreak __heap_select
Breakpoint "__heap_select" will stop the execution on 2 locations
(mdb) c
Breakpoint "__heap_select" reached
std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
(mdb) bt
#0 std::__heap_select<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5299:7)
#1 std::partial_sort<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:2310:8)
#2 std::__introsort_loop<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > >, long> (TemplateInstantiation from /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../include/c++/4.8/bits/stl_algo.h:5451:4)
#3 std::sort<__gnu_cxx::__normal_iterator<person *, std::vector<person, std::allocator<person> > > > (TemplateInstantiation from <stdin>:2:26)
#4 METASHELL_INSTANTIATE_EXPRESSION( std::sort(people.begin(), people.end()) )
```

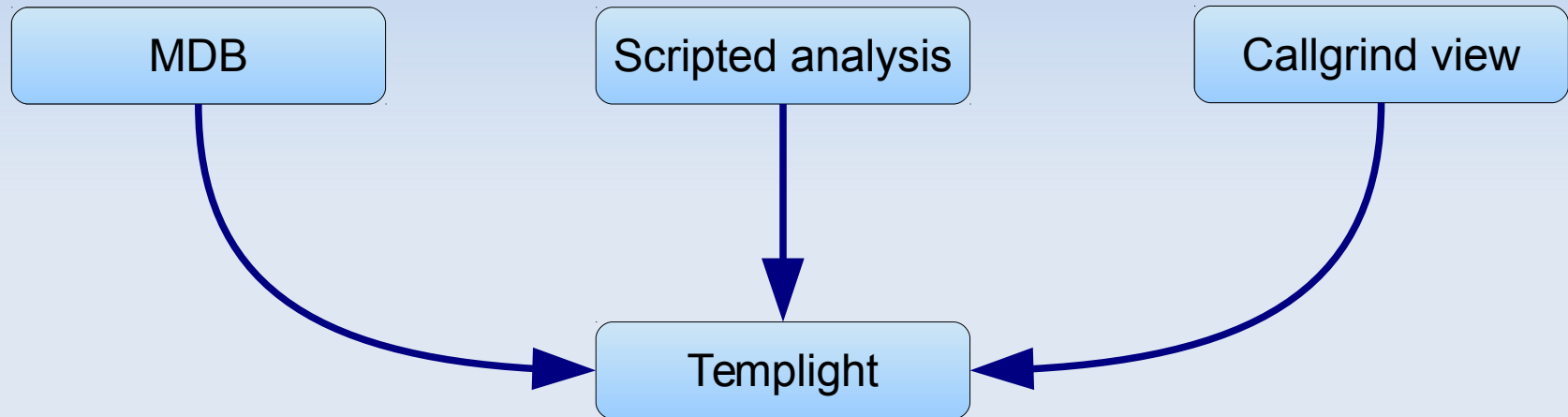
# It all depends on a Clang patch

MDB

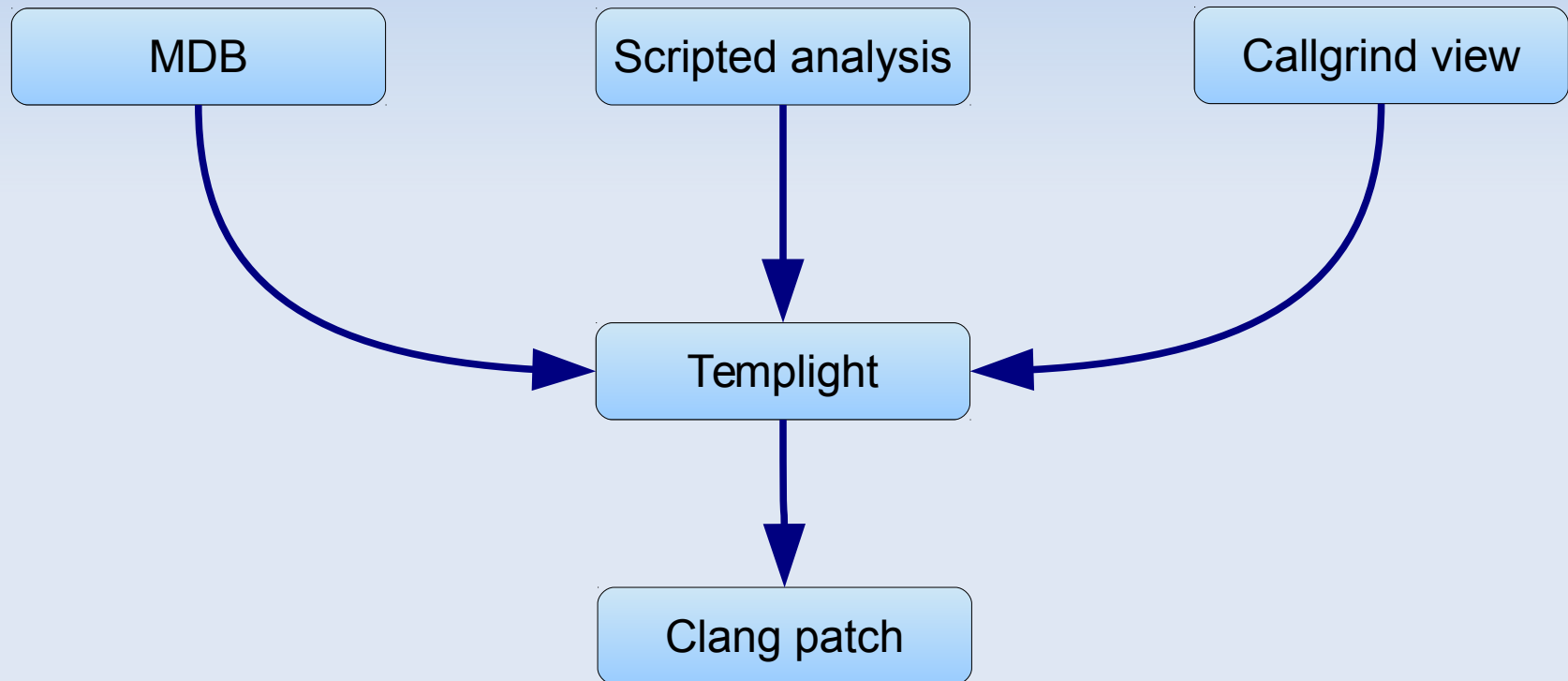
Scripted analysis

Callgrind view

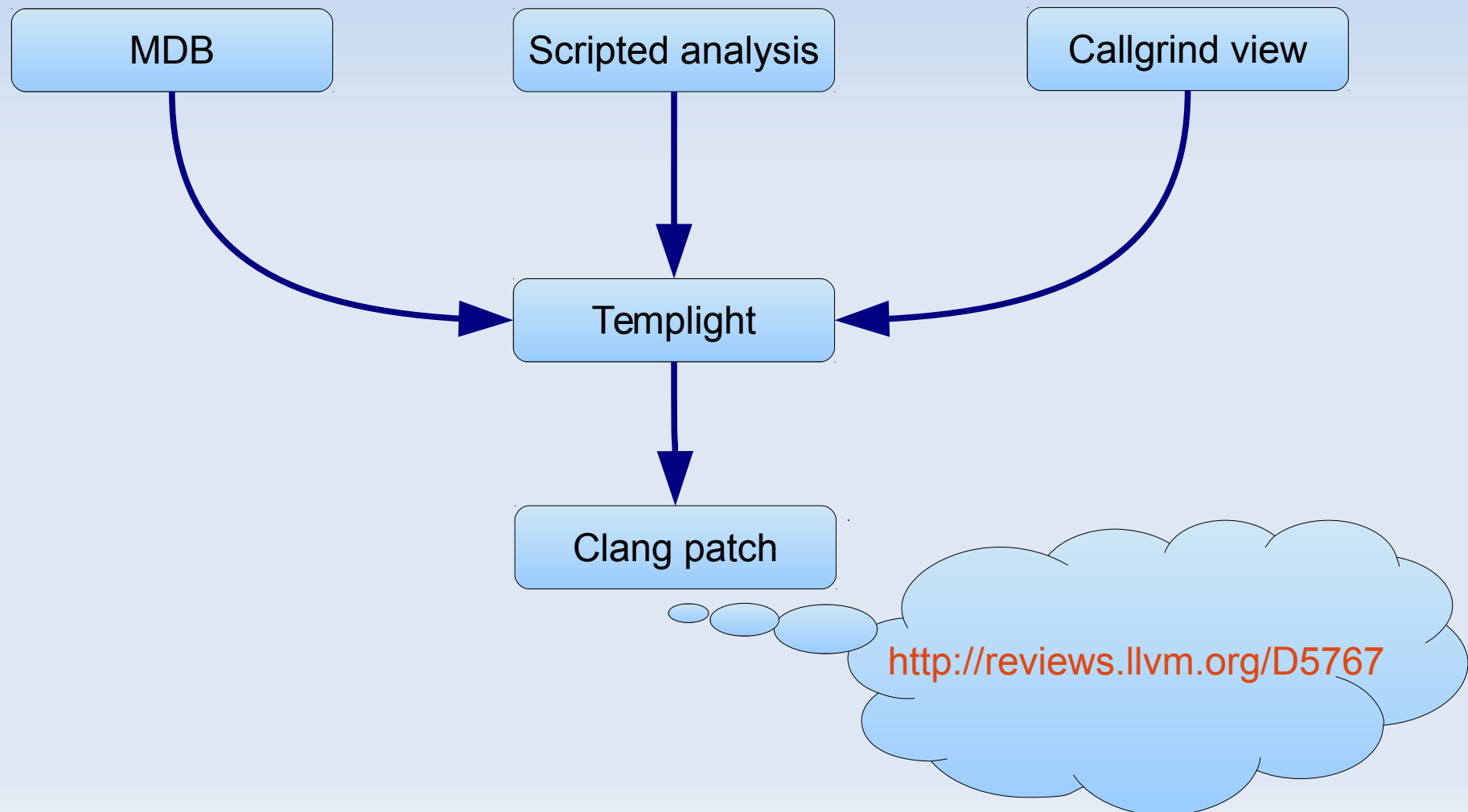
# It all depends on a Clang patch



# It all depends on a Clang patch



# It all depends on a Clang patch





**Q & A**