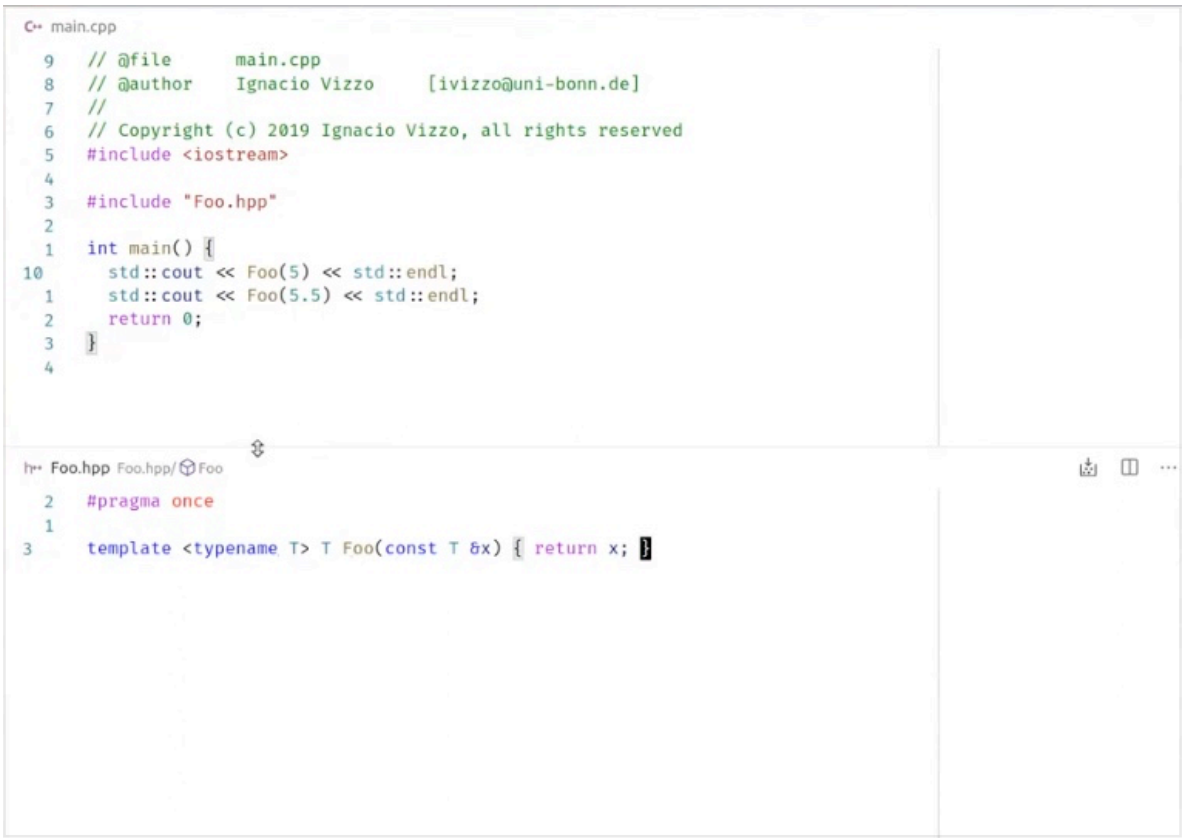
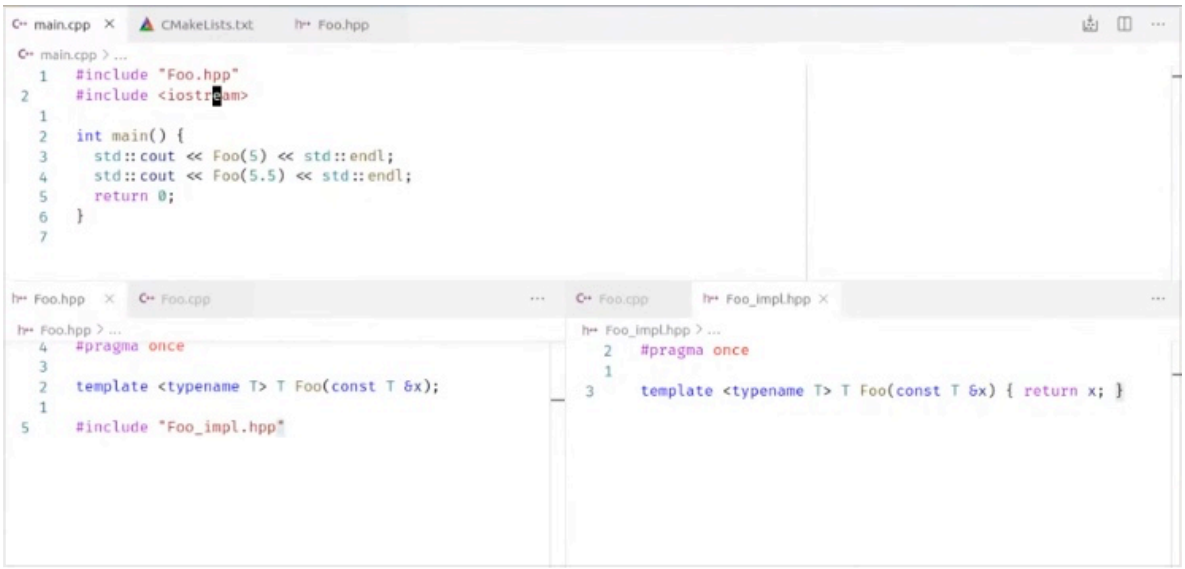
Template Headers / Source

- Concrete templates are instantiated at compile time.
- Linker does not know about implementation.
- There are three options for template classes:
  - Declare and define in the header files



  - Declare in NAME.hpp file, implement in NAME_impl.hpp file, add #include <NAME_impl.hpp> in end of NAME.hpp



- Declare in *.hpp file, implement in *.cpp file, in the end of the *.cpp add explicit instantiation for types you expect to use.

main.cpp

```cpp
#include "Foo.hpp"
#include <iostream>

int main() {
  std::cout << Foo(5) << std::endl;
  std::cout << Foo(5.5) << std::endl;
  return 0;
```

Foo.hpp

```cpp
#pragma once

template <typename T> T Foo
(const T &x);
```

Foo.cpp Foo.cpp/...

```cpp
#include "Foo.hpp"

template <typename T> T Foo(const T &x) { return x; }

template int Foo<int>(const int &x);
template double Foo<double>(const double &x);
```