CPPCON 2019

POSTMODERN META C++

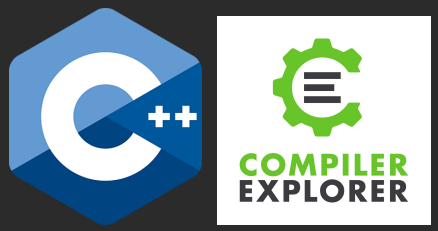
Kris Jusiak, Quantlab Financial

KRIS@JUSIAK.NET | @KRISJUSIAK | LINKEDIN.COM/IN/KRIS-JUSIAK

DISCLAIMER - ALL PRESENTED EXAMPLES ARE COMPILING/WORKING

✔

POWERED BY



C++17 - REFLECTION (STRUCTURE BINDINGS)

C++17 - REFLECTION (STRUCTURE BINDINGS)

```
struct foo {
  int i{};
  double d{};
};
```

C++17 - REFLECTION (STRUCTURE BINDINGS)

```
struct foo {
  int i{};
  double d{};
};
```

```
auto tuple = to_tuple(foo{.i = 4, .d = 2.});
```

C++17 - REFLECTION (STRUCTURE BINDINGS)

```
struct foo {
  int i{};
  double d{};
};
```

```
auto tuple = to_tuple(foo{.i = 4, .d = 2.});
```

```
static_assert(std::is_same_v<
                std::tuple<int, double>,
                decltype(tuple)
              >);

static_assert(4  == std::get<0>(tuple));
static_assert(2. == std::get<1>(tuple));
```

C++17 - REFLECTION (STRUCTURE BINDINGS)

```
struct foo {
  int i{};
  double d{};
};
```

```
auto tuple = to_tuple(foo{.i = 4, .d = 2.});
```

```
static_assert(std::is_same_v<
                std::tuple<int, double>,
                decltype(tuple)
              >);

static_assert(4  == std::get<0>(tuple));
static_assert(2. == std::get<1>(tuple));
```

https://godbolt.org/z/BJH2SR

C++17 - COMPILE TIME INTERFACE

STD::TUPLE

C++17 - COMPILE TIME INTERFACE

STD::TUPLE

```
static_assert(4  == tuple[0_c] and 4  == tuple[int{}]);
static_assert(2. == tuple[1_c] and 2. == tuple[double{}]);

// static_assert(4  == tuple["int"_t] and
//               2. == tuple["double"_t]);
```

C++17 - COMPILE TIME INTERFACE

STD::TUPLE

```
static_assert(4  == tuple[0_c] and 4  == tuple[int{}]);
static_assert(2. == tuple[1_c] and 2. == tuple[double{}]);

// static_assert(4  == tuple["int"_t] and
//               2. == tuple["double"_t]);
```

https://godbolt.org/z/Mcdj1h

C++17 - COMPILE TIME INTERFACE

BOOST::MP11

C++17 - COMPILE TIME INTERFACE

BOOST::MP11

```
template<class... Ts>
constexpr auto unique_foo_ptrs = list<Ts...>
  | filter<is_same<foo>>
  | transform<add_pointer>;
  | unique;
```

C++17 - COMPILE TIME INTERFACE

BOOST::MP11

```
template<class... Ts>
constexpr auto unique_foo_ptrs = list<Ts...>
  | filter<is_same<foo>>
  | transform<add_pointer>;
  | unique;
```

```
class foo {};
class bar {};
```

C++17 - COMPILE TIME INTERFACE

BOOST::MP11

```
template<class... Ts>
constexpr auto unique_foo_ptrs = list<Ts...>
  | filter<is_same<foo>>
  | transform<add_pointer>;
  | unique;
```

```
class foo {};
class bar {};
```

```
static_assert(unique_foo_ptrs<foo, bar, foo> == list<foo*>);
```

C++17 - COMPILE TIME INTERFACE

BOOST::MP11

```
template<class... Ts>
constexpr auto unique_foo_ptrs = list<Ts...>
  | filter<is_same<foo>>
  | transform<add_pointer>;
  | unique;
```

```
class foo {};
class bar {};
```

```
static_assert(unique_foo_ptrs<foo, bar, foo> == list<foo*>);
```

https://godbolt.org/z/e8pMyp

C++20 CONCEPTS - DESIGN BY INTROSPECTION

C++20 CONCEPTS - DESIGN BY INTROSPECTION

```
constexpr auto f_i = [](auto t) -> int {
  if constexpr (requires{ t.i; }) {
    return t.i;
  } else {
    return {};
  }
};
```

C++20 CONCEPTS - DESIGN BY INTROSPECTION

```
constexpr auto f_i = [](auto t) -> int {
  if constexpr (requires{ t.i; }) {
    return t.i;
  } else {
    return {};
  }
};
```

```
struct foo {
  int i{};
};
static_assert(42 == f_i(foo{.i = 42}));
```

C++20 CONCEPTS - DESIGN BY INTROSPECTION

```
constexpr auto f_i = [](auto t) -> int {
  if constexpr (requires{ t.i; }) {
    return t.i;
  } else {
    return {};
  }
};
```

```
struct foo {
  int i{};
};
static_assert(42 == f_i(foo{.i = 42}));
```

```
struct bar {
  // int i{}; // no i
};
static_assert(0 == f_i(bar{}));
```

C++20 CONCEPTS - DESIGN BY INTROSPECTION

```
constexpr auto f_i = [](auto t) -> int {
  if constexpr (requires{ t.i; }) {
    return t.i;
  } else {
    return {};
  }
};
```

```
struct foo {
  int i{};
};
static_assert(42 == f_i(foo{.i = 42}));
```

```
struct bar {
  // int i{}; // no i
};
static_assert(0 == f_i(bar{}));
```

https://godbolt.org/z/6HUiIu

C++20 - TESTING FRAMEWORK (NO MACROS)

OUTPUT

C++20 - TESTING FRAMEWORK (NO MACROS)

```
int main() {
  "should not be equal"_test = [] {
    expect("diff") << 42 == 99;
  };
}
```

OUTPUT

C++20 - TESTING FRAMEWORK (NO MACROS)

```
int main() {
  "should not be equal"_test = [] {
    expect("diff") << 42 == 99;
  };
}
```

OUTPUT

```
example.cpp:85[should not be equal] diff [42 == 99]
```

C++20 - TESTING FRAMEWORK (NO MACROS)

```
int main() {
  "should not be equal"_test = [] {
    expect("diff") << 42 == 99;
  };
}
```

OUTPUT

```
example.cpp:85[should not be equal] diff [42 == 99]
```

https://godbolt.org/z/Eqv4-N

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
constexpr auto sum = [](auto... args) {

};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
constexpr auto sum = [](auto... args) {
  return (0 + ... +
  );
};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
constexpr auto sum = [](auto... args) {
```

```
  return (0 + ... +
```

```
    (to_tuple(args)
      | filter([](auto t) { return requires { t.i; }; })
      | unique
    )
```

```
  );
```

```
};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
constexpr auto sum = [](auto... args) {

  return (0 + ... +

    // hand-written std::apply
    []<template<class...> class T, class... Ts>(T<Ts...> t) {
      return (0 + ... + t[Ts{}].i);
    }

    (to_tuple(args)
      | filter([](auto t) { return requires { t.i; }; })
      | unique
    )

  );

};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
struct foo {
  struct { int i{}; } bar1;
  struct { } bar2;
  struct { int i{}; } bar3;
};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
struct foo {
  struct { int i{}; } bar1;
  struct { } bar2;
  struct { int i{}; } bar3;
};
```

```
"should sum over unique structs with i"_test = [] {
  static_expect("sum") <<
    4 + 2 == sum(foo{.bar1 = {.i = 4}, .bar3 = {.i = 2}});
};
```

C++20 - IMMEDIATELY-INVOKED FUNCTION EXPRESSION (IIFE)

```
struct foo {
  struct { int i{}; } bar1;
  struct { } bar2;
  struct { int i{}; } bar3;
};
```

```
"should sum over unique structs with i"_test = [] {
  static_expect("sum") <<
    4 + 2 == sum(foo{.bar1 = {.i = 4}, .bar3 = {.i = 2}});
};
```

https://godbolt.org/z/5zEnBB

LET'S EMBRACE POSTMODERN META C++!

KRIS@JUSIAK.NET | @KRISJUSIAK | LINKEDIN.COM/IN/KRIS-JUSIAK