

# C++ Club Meeting Notes

---

Gleb Dolgich

2019-12-12

Bjarne Stroustrup

Reddit

Niall Douglas's rebuttal: [Part 1](#), [Part 2](#)

**C++ Exception Optimizations. An experiment**

Gor Nishanov

Corentin Jabot

P0443R11 The Unified Executors Proposal

- C++ coroutines: Getting started with awaitable objects
- C++ coroutines: Constructible awaitable or function returning awaitable?
- C++ coroutines: Framework interop

# Howard Hinnant on how to initialize things (1/4)



Andreas Reischuck @arBmind

Decision graph for initialisation from Howart Hinnant secret lightning talk @meetingcpp #meetingcpp

How To Initialize x from expression y

Should x and y have the same cv-unqualified type?

- Let's start by classifying the use cases.

```
auto x = y;  
auto& x = y;  
auto&& x = y;  
X x = y;  
auto x = X{y};  
auto x = X(y);
```

How To Initialize x from expression y

Is the type conversion implicit? Prefer implicit conversions?!

Safest choice!

```
template <class Duration1, class Duration2>  
auto  
avg_nanoseconds(Duration1 d1, Duration2 d2)  
{  
    using namespace std::chrono;  
    nanoseconds ns = d1 + d2;  
    return ns/2;  
}  
  
auto x = avg_nanoseconds(2, 1);  
error: no viable conversion from 'int' to 'nanoseconds'  
nanoseconds ns = d1 + d2;
```

How To Initialize x from expression y

Should x and y have the same cv-unqualified type?

Should x be a non-reference type?

Should x be a lvalue or rvalue reference?

Is the type conversion implicit?

Can the conversion be made with braces?

auto x = y; auto& x = y; auto&& x = y; X x = y; auto x = X{y}; auto x = X(y);

Add const (and/or volatile) as appropriate

# How To Initialize **x** from expression **y**

Should **x** and **y** have the same cv-unqualified type?

- Let's start by classifying the use cases.

```
auto x = y;
```

```
auto& x = y;
```

```
auto&& x = y;
```

```
X x = y;
```

```
auto x = X{y};
```

```
auto x = X(y);
```

# How To Initialize $x$ from expression $y$

Is the type conversion implicit?

Yes

`X x = y;`

Safest choice!

*Prefer implicit conversions?!*

```
template <class Duration1, class Duration2>
auto
avg_nanoseconds(Duration1 d1, Duration2 d2)
{
    using namespace std::chrono;
    nanoseconds ns = d1 + d2;
    return ns/2;
}
```

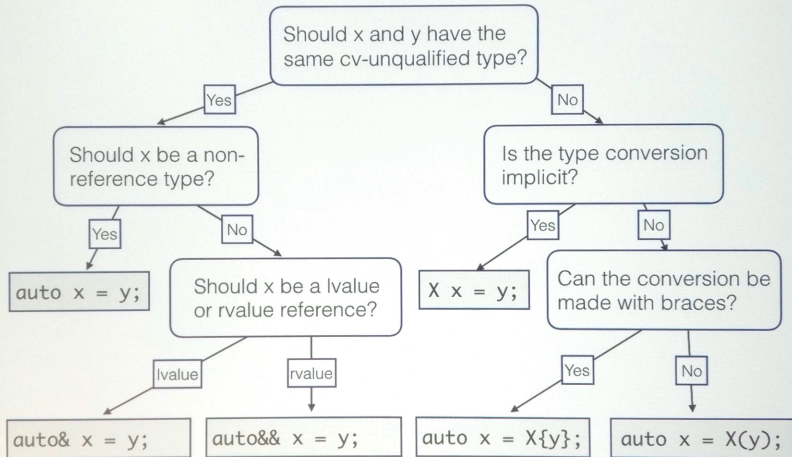
Implicit conversion

```
auto x = avg_nanoseconds(2, 1);
```

**error:** no viable conversion from 'int' to 'nanoseconds'

```
    nanoseconds ns = d1 + d2;
```

# How To Initialize $x$ from expression $y$



Add const (and/or volatile) as appropriate.



**Change standard containers' `size()` method to return signed integer?**

Reddit

**Is requiring lambdas to explicitly list what they capture a good coding standard?**

Reddit

## Empty struct size in C and C++



**JF Bastien** @jfbastien

Today's episode of "something I didn't know about C and C++":

```
int size() {  
    struct empty {};  
    return sizeof(struct empty);  
}
```

This code returns 0 in C and 1 in C++, because empty structs have different size in both languages. 🙄

🗨 thread

20w • 21/02/2019 • 20:33



**David Brady**

@dbrady



The older I get, the less I care about making tech decisions right and the more I care about retaining the ability to change a wrong one.

11:12 AM - Feb 27, 2017

♡ 1,316 💬 920 people are talking about this



<https://twitter.com/dbrady/status/836277898873044995>



**funky lesbian** @trans\_disaster

STOP ARGUING OVER THE BEST  
PROGRAMMING LANGUAGE

C is LOW-LEVEL

C++ is POWERFUL

Python is INTUITIVE

Rust is SAFE

Lua is EASY

Java

C# is LEGIBLE

17h • 19/08/2019 • 18:29 •