



A New Decade of Visual Studio

C++20, Open STL, and More

Marian Luparu @mluparu
Sy Brand @TartanLlama

C++ Product Team, Microsoft
@VisualC
<https://aka.ms/cpp>



Welcome to CppCon 2020!

Visit our table at the Expo Hall

<https://aka.ms/cppcon/expo>

- Meet the Microsoft C++ team
- Ask any questions
- Discuss the latest announcements



Take our survey

<https://aka.ms/cppcon>

Chance to win 1 of 5 copies of Microsoft Flight Simulator



Mission of the C++ product team at Microsoft

Make the lives of all C++ developers on the planet better

Our
agenda
today

1. by participating in the creation of the **C++ Standards**
2. by investing in the Microsoft Visual C++ (**MSVC**) Compiler & Libraries
3. by simplifying C++ library acquisition via **vcpkg**
4. by improving the **Visual Studio IDE**
5. by continuing to enhance the C++ extension for **Visual Studio Code**

Visual Studio Code

Session coming up!



Mon 9/14 12:00 – 13:00

Collaborative C++ Development with Visual Studio Code

Julia Reid – *count_if()* venue

```
helloworld.cpp •
1  #include <iostream>
2
3  int main()
4  {
5      std::cout << "Hello World!" << std::endl;
6      std::cout
7      return 0;
8  }
```

std::ostream std::cout
File: iostream

- count
- _Count_pr
- conjunction
- conjunction_v
- _Atomic_counter_t
- _Check_match_counts
- _Copy_unchecked
- _Copy_unchecked1
- _Get_atomic_count
- _Ptr_count
- _Ref_count

#1 most used code editor

[StackOverflow Developer Surveys]

Free, open source code editor

Runs on Windows, macOS, and Linux

C++ IntelliSense, debugging, code browsing

CMake, vcpkg and git integrations

Remote experiences

and more...

<https://aka.ms/cpp/code>



Visual Studio

Agenda

1. Conformance
2. Code Safety
3. Cross-platform development
4. Developer and Team Productivity



Available this week

Visual Studio 2019 v16.8 Preview 3

Download from

<https://visualstudio.microsoft.com/vs/preview/>



Visual Studio

Agenda

1. Conformance
2. Code Safety
3. Cross-platform development
4. Developer and Team Productivity

Announcing today

With version 16.8,
Visual Studio 2019 achieves
C17 standards conformance

- Supporting all C11/C17 required* features
- Requires preview release of Windows SDK
- *C11 optional features not yet supported

Visit <https://aka.ms/cpp/c17> for more details



At CppCon last year

Visual Studio 2019

☒ C++98*

*) with /permissive-

☒ C++11

☒ C++14

☒ C++17*

*) preprocessor still experimental

☒ C++20
under /std:c++latest

C++20 progress

- **Concepts** – feature complete (under /std:c++latest)
- **Coroutines** (partial support) under /await
- **Modules** (partial support) under /experimental:module
- **<=> three-way comparison** operator (partial support)
- Feature-test macros
- `remove_cvref`
- Prevent aggregate-init with user-declared or deleted constructors

Conformance

Announcing today

C++20 Modules are feature complete

- Available in Visual Studio 2019 version 16.8
- Under `/std:c++latest` switch
- Includes header units and experimental module tooling

Visit <https://aka.ms/cpp/20modules> for more details

Announcing today

C++20 Coroutines are feature complete

- Available in Visual Studio 2019 version 16.8
- Under `/std:c++latest` switch
- Includes coroutines tooling

Visit <https://aka.ms/cpp/20coro> for more details





Visual Studio



C++98*

* with /permissive-



C++11



C++14



C++17



C++20

with /std:c++latest

C++20 progress



Feature complete

- Coroutines
- Modules
- Concepts
- `<=>` three-way comparison operator
- 84 STL features including
 - C++ synchronization library



Coming next

- C++20 `constexpr`
- Rest of 24 C++20 STL features
 - `std::ranges`, `std::format` & `Chrono`
 - Specialized memory ranges algo.
- `/std:c++20` official switch

Visual Studio



C++98*

* with /permissive-



C++11



C++14



C++17



C++20

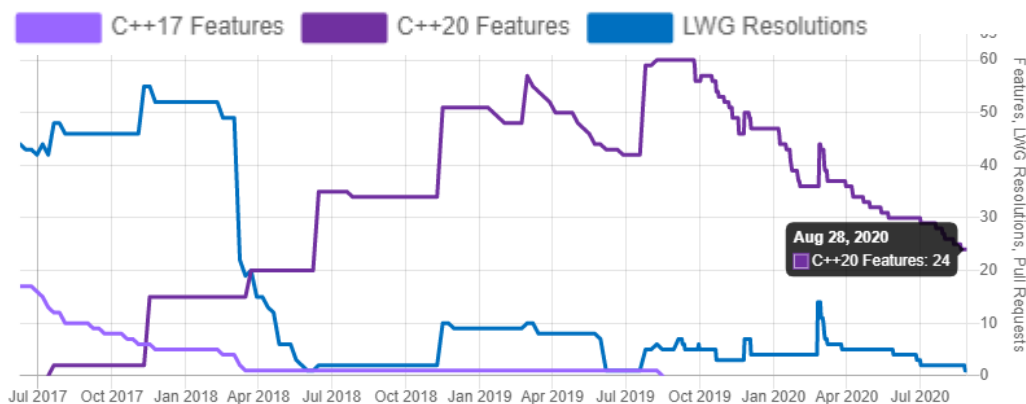
with /std:c++latest

STL update

<https://github.com/microsoft/STL>

84 C++20 features completed

36 C++20 features externally contributed YOU ARE AWESOME!



<https://microsoft.github.io/STL/>



Tue 9/15 13:30 – 14:30

C++20 STL Features: One Year of Development on GitHub

Stephan T. Lavavej – *generate_n()* venue

Visual Studio



C++98*

* with /permissive-



C++11



C++14



C++17




C++20

with /std:c++latest

std::ranges

Huge update in 16.8 Preview 3!

- 95% complete  Try it today
- Algorithms: sort, rotate, sample, shuffle, reverse, merge, move_backwards, next_permutation, partial_sort, uninitialized_move
- Views: ref_view
- Factories: all, reverse, filter, drop, take, transform
- string_view constructor



Coming next

- Algorithms: stable_sort, stable_partition and inplace_merge
- Factories: iota_view and basic_istream_view
- Adaptors: views::counted, drop_while_view, take_while_view, elements_view (keys_view, values_view), common_view, join_view, split_view
- Specialized memory concepts, uninitialized_*

Visit <https://aka.ms/cpp/20ranges> for more details

Demo





Visual Studio

Agenda

1. Conformance
2. Code Safety
3. Cross-platform development
4. Developer and Team Productivity

Static Analysis

Visual Studio integrates with

- MSVC Code Analysis <https://aka.ms/cpp/ca/bg>
- Clang-tidy <https://aka.ms/cpp/clangtidy>
- Visual Studio Code Linters <https://aka.ms/cpp/linter>



New C++ Core Checkers in MSVC Code Analysis

- Missing default label in switch statements
- Unannotated fall through in switch statements
- Expensive range-for copy
- Expensive copy with the auto keyword

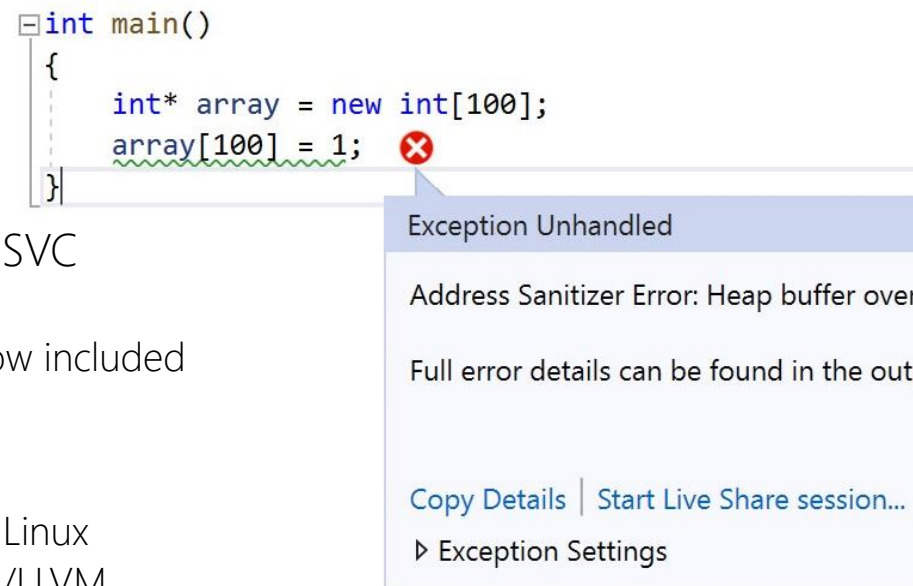


Tue 9/15 12:00 – 13:00

Closing the Gap between Rust and C++ Using Principles of Static Analysis

Sunny Chatterjee – *destroy_n()* venue

Address Sanitizer



Experimental /fsanitize=address support in MSVC

- x86 and X64 support
-  Debug configurations (/MTd, /MDd) now included

Visual Studio integration

- MSBuild & CMake support for Windows & Linux
- Debugger integration for MSVC and Clang/LLVM



Tue 9/15 9:00 – 10:00

2020: The Year of Sanitizers?

Victor Ciura – *Fuzzing/Testing venue*



Fri 9/18 12:00 – 13:00

Introducing Microsoft's New Open Source Fuzzing Platform

Justin Campbell, Michael Walker – *Fuzzing/Testing venue*

Visit <https://aka.ms/asan> to learn more

Control Flow Guard (CFG)

Enforce control flow integrity

- Windows 8.1 & Windows 10
- MSVC compiler

 `/guard:cf` now supported in LLVM 10 too

<https://aka.ms/cpp/cfg-llvm>

To learn about all Security features in MSVC, check out <https://aka.ms/cpp/security>

Installation details

✓ Desktop development with C++

Included

- ✓ C++ core desktop features
- ✓ IntelliCode

Optional

- ☒ MSVC v142 - VS 2019 C++ x64/x86 build tools (...)
- ☐ Windows 10 SDK (10.0.19041.0)
- ☒ Just-In-Time debugger
- ☒ C++ profiling tools
- ☒ C++ CMake tools for Windows

- ☐ C++ Modules for v142 build tools (x64/x86 - ex...
- ☒ C++ Clang tools for Windows (10.0.0 - x64/x86)
- ☐ JavaScript diagnostics
- ☐ IncrediBuild - Build Acceleration
- ☒ Windows 10 SDK (10.0.18362.0)





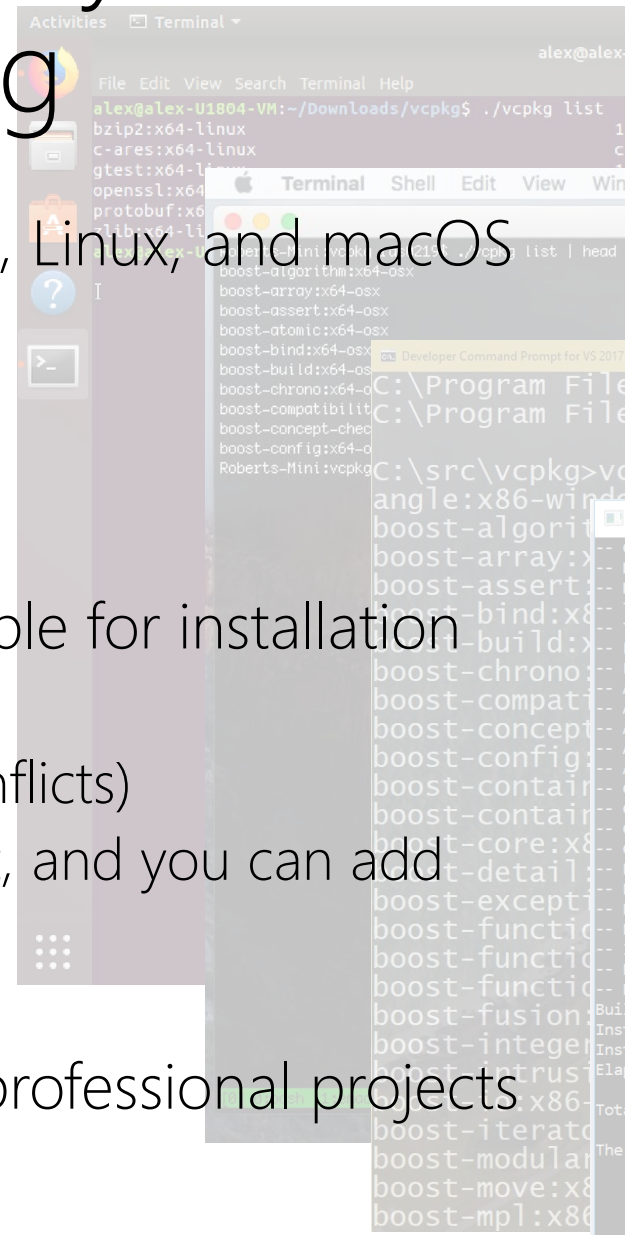
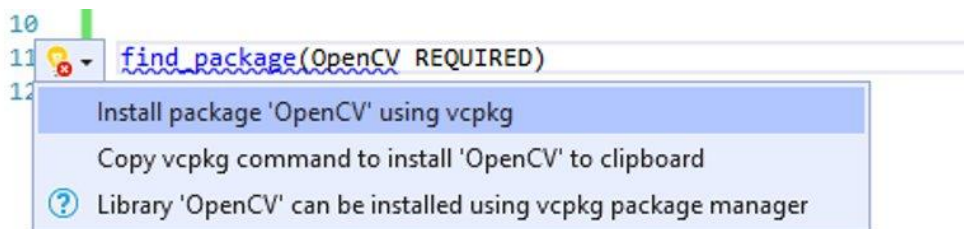
Visual Studio

Agenda

1. Conformance
2. Code Safety
3. Cross-platform development
4. Developer and Team Productivity

Simplify C++ dependency management with vcpkg

Open source library manager for Windows, Linux, and macOS



1400+ popular open source libraries available for installation

- Built-from source, on-demand
- Centralized, tested catalog (no version conflicts)
- Popular build targets supported by default, and you can add your own custom targets

Powerful ***new*** features for personal **and** professional projects

Learn more at <https://aka.ms/vcpkg>

Announcing today

vcpkg Enterprise Features

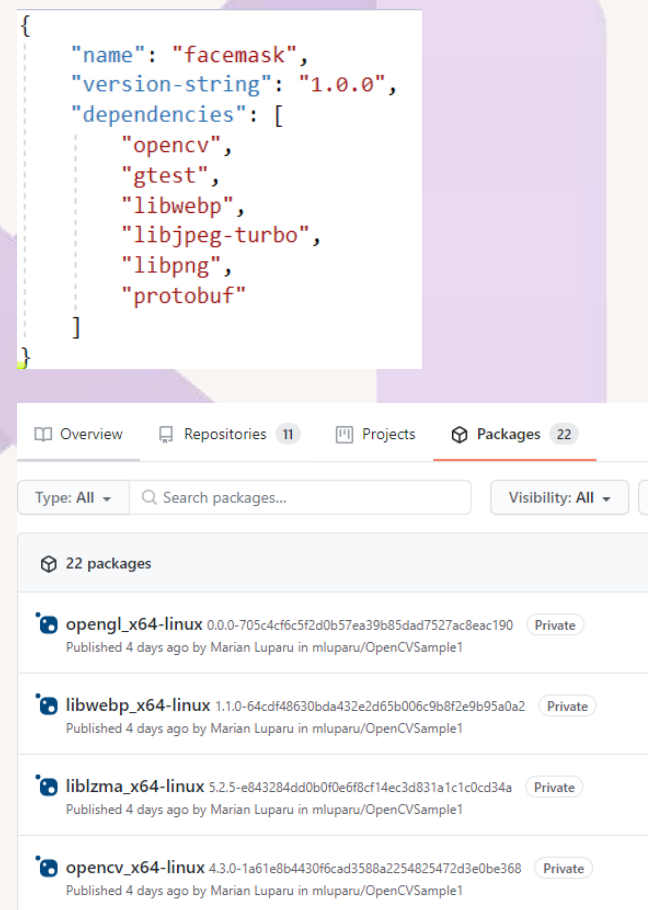
Manifests *(general availability)*

vcpkg.json for declaring your dependencies and acquiring them as a built-in step (CMake & MSBuild supported)

Binary Caching *(general availability)*

Build your dependencies once, then cache them for re-use on other machines, CI runs, containers, or other environments

Visit <https://aka.ms/vcpkg/team> for more details



On our roadmap

vcpkg Enterprise Features

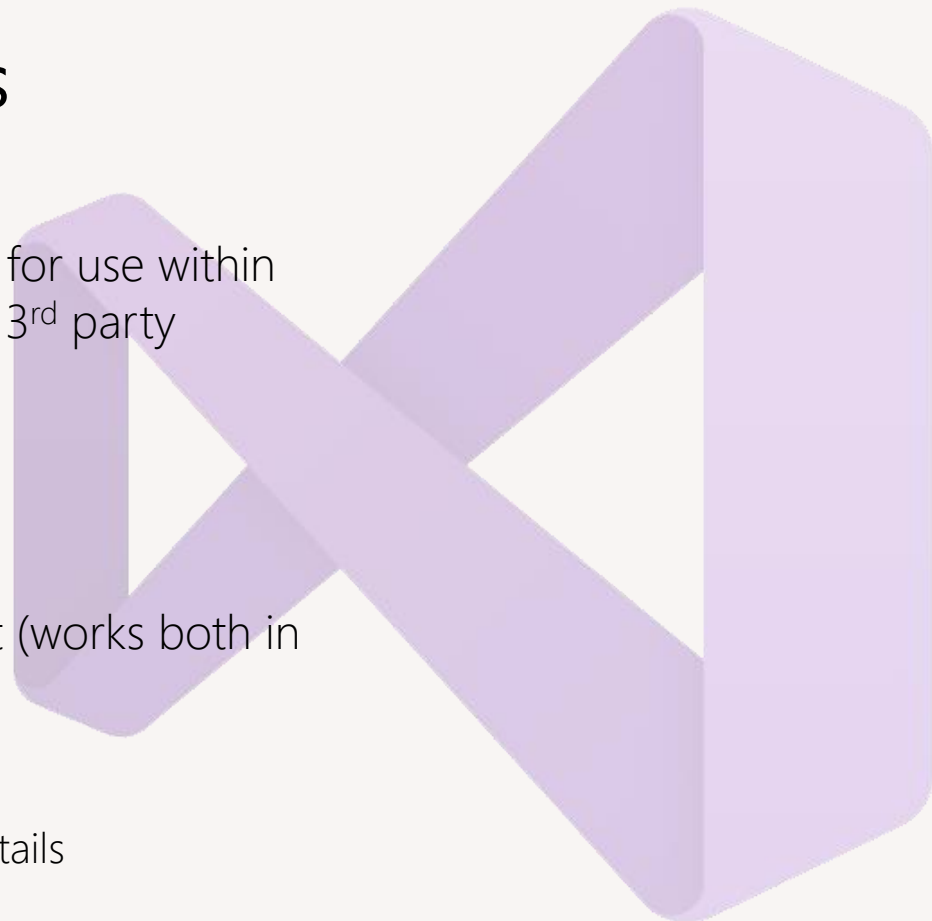
Registries *(coming soon)*

Create your own private library catalog for use within vcpkg; great for internal, closed-source 3rd party libraries and community repositories

Versioning *(coming soon)*

Choose which library versions you want (works both in the command line and manifest)

Visit <https://aka.ms/vcpkg/team> for more details





CMake integration

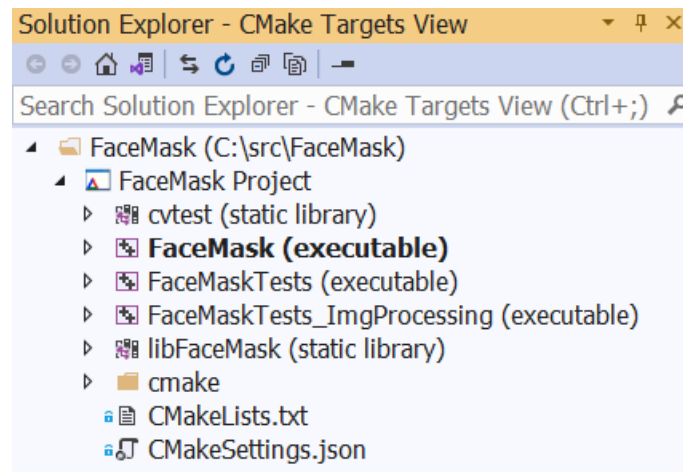
CMake is a first-class project system in Visual Studio

Familiar edit-build-debug inner-loop

- including MSVC and Clang-tidy squiggles & quick actions
- Doxygen integration

Target Windows, Linux, macOS, MinGW and more

Learn more at <https://aka.ms/cmake>



Linux targeting

Target any Linux distro, including WSL
(Windows subsystem for Linux)

C++ IntelliSense can parse remote
Linux headers in GCC mode

Build remotely via MSBuild or CMake

Debug local or remote targets

Learn more at <https://aka.ms/vslinux>



What's new in Visual Studio 2019

Making CMake easier to use

- CMake language services
 - go to definition, find all references in CMakeLists.txt
- CMake project manipulation
 - adding/removing/renaming files, targets, references in CMake Targets View

Streamlined Linux support

- Core dump debugging
- First-class support for gdbserver
- Support for Ninja on Linux
- More Linux distros and shells
- More accurate IntelliSense in remote scenarios



Wed 9/16 12:00 – 13:00

Cross-Platform Pitfalls and How to Avoid Them

Erika Sweet – *generate_n()* venue



Visual Studio

Agenda

1. Conformance
2. Code Safety
3. Cross-platform development
4. Developer and Team Productivity

Build Time Improvements

• 16.6

- Optimized symbol lookup.
- ~2X better time to enter break state while debugging.
- Up to 4X better in single-step performance for big functions in the largest projects.

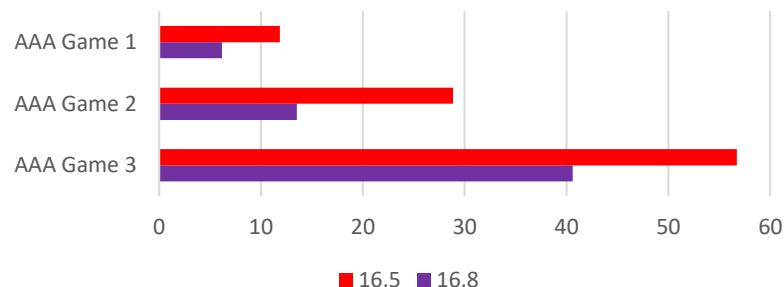
• 16.7

- Improved worst-case incremental link time by caching results of earlier PDB creation.
- 1.5X to 5.5X speedup.

• 16.8

- Able to scale PDB creation to more cores.
- Often 2X better link times on large projects.

AAA Game Full Link Times
(Seconds)



C++ Build Insights

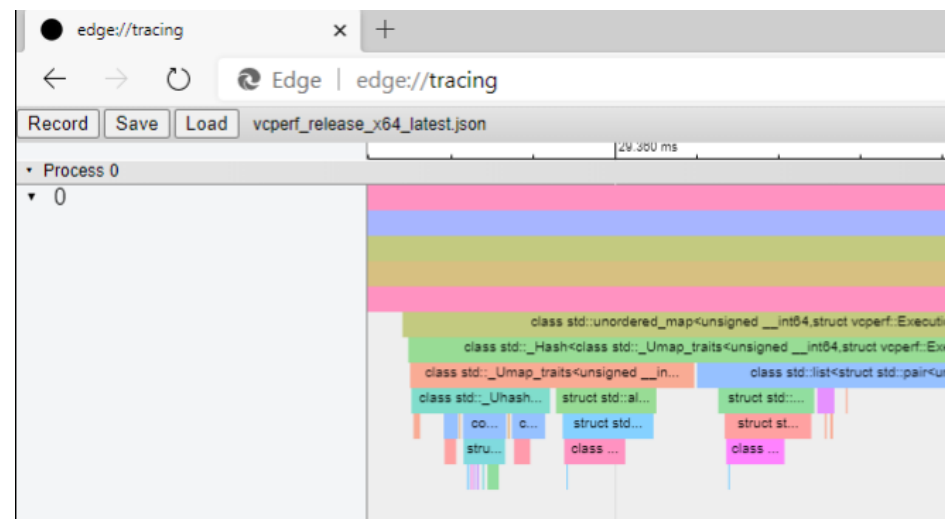
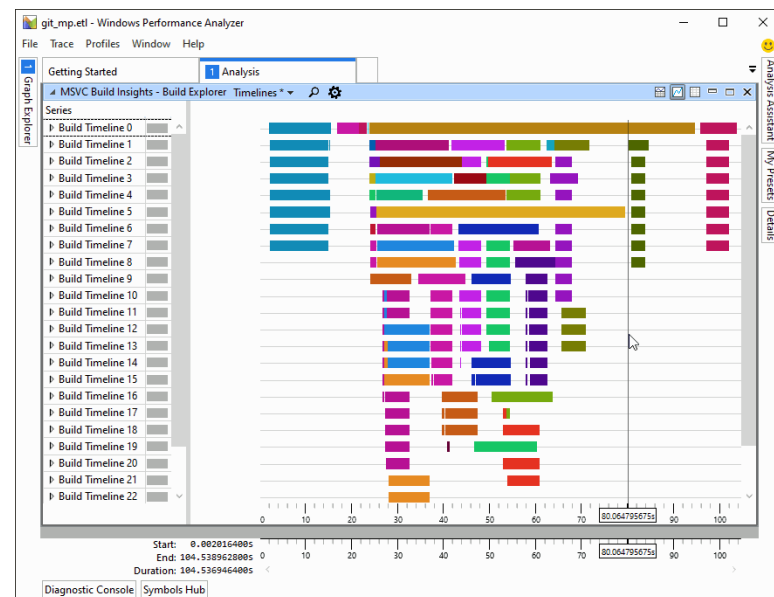
Find bottlenecks in your build

- ETW-based instrumentation
- Visualize using Windows Performance Analyzer or Edge/Chrome via /timetrace (thank you to Carlos Fraguas)

```
vcperf /start Session
```

```
<build command>
```

```
vcperf /stop Session /timetrace mytrace.json
```



Visit <https://aka.ms/vcperf> to learn more

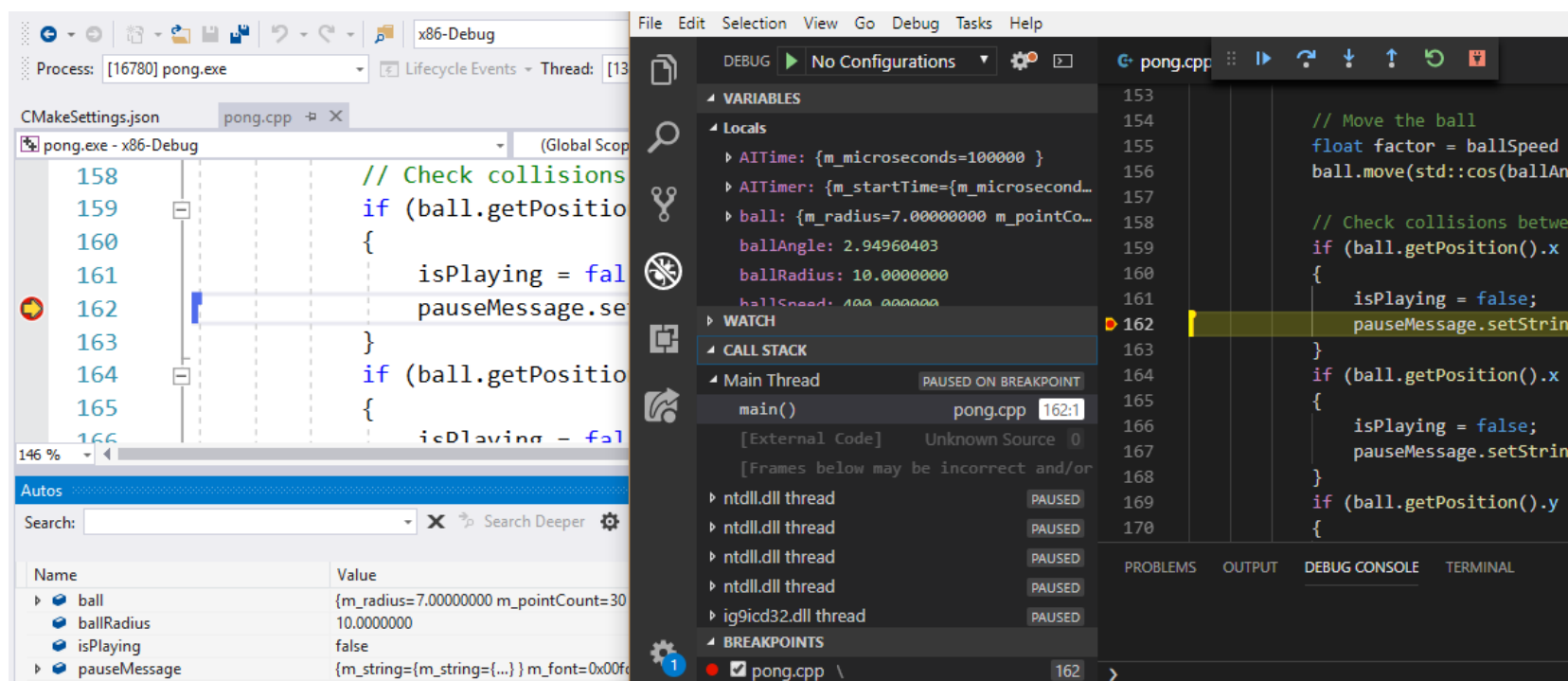
Live Share

Easily work together, in any way you need

- Collaborative editing
- Collaborative debugging

No hassle sharing for any app, on any OS

- Visual Studio, Visual Studio Code, or browser
- macOS, Linux, or Windows



Learn more at <https://aka.ms/cpp/liveshare>

GitHub Codespaces

Your instant dev environment

Sign up today <https://aka.ms/codespaces-signup>

Code without compromise

Create cloud-hosted development environments for your repository in minutes

Code, build, test & debug with a complete development environment in your browser

Access your code from anywhere

With your development in the cloud, you can contribute code from any device, using Visual Studio, Visual Studio Code or in your browser

Feel at home, even when you're away with roaming settings, themes and Git identity

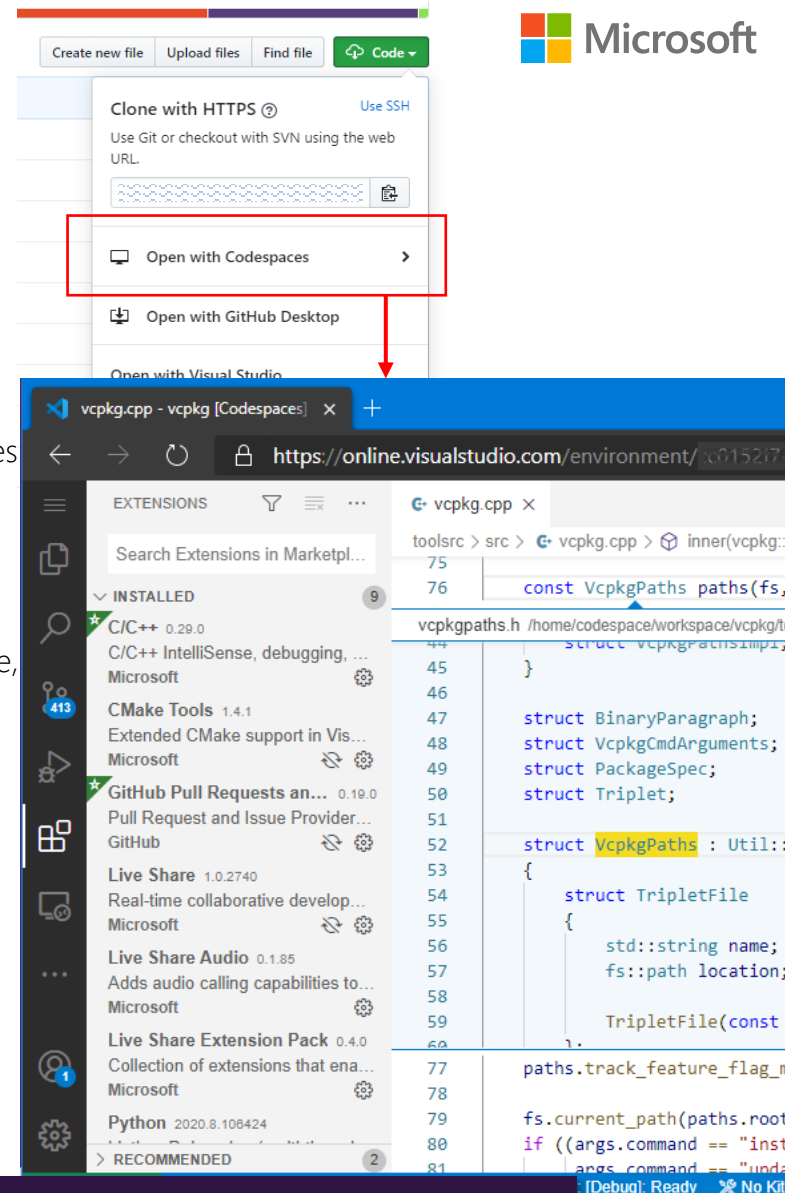
Free up resources on your machine

The cloud is the limit to what your dev environment can do

Work on one or multiple projects without worrying about slowing down your local box

Easy onboarding for new contributors

Replicate an environment in 2 clicks, reduce maintenance time



Wed 9/16 14:10 – 14:40

Effective Remote C++ Development with Codespaces

Nick Uhlenhuth – *generate_n()* venue



Mission of the C++ product team at Microsoft

Make the lives of all C++ developers on the planet better

1. by participating in the creation of the **C++ Standards**
2. by investing in the Microsoft Visual C++ (**MSVC**) Compiler & Libraries
3. by simplifying C++ library acquisition via **vcpkg**
4. by improving the **Visual Studio IDE**
5. by continuing to enhance the C++ extension for **Visual Studio Code**

Developer Community

- 132 votes: Add C11 support
- 61: Automatically copy ProjectReferenced dlls to output in MSBuild
- 48: Library support for Compiler Explorer
- 41: Address Sanitizer Support
- 33: Create a UTF-8 runtime
- 25: Compiler warning when using implicit fallthroughs
- 14: Support `__forceinline` on lambdas



Happy Coding!

Thank you

Enjoy the rest of the conference!

Visit our table at the Expo Hall

<https://aka.ms/cppcon/expo>

- Meet the Microsoft C++ team
- Ask any questions
- Discuss the latest announcements



Take our survey

<https://aka.ms/cppcon>

Chance to win 1 of 5 copies of Microsoft Flight Simulator

Our Sessions

Monday 14th

- A New Decade of Visual Studio: C++20, Open STL, and More – Sy Brand & Marian Luparu
- Collaborative C++ Development with Visual Studio Code – Julia Reid

Tuesday 15th

- Building an Intuition for Composition – Sy Brand
- Closing the gap between Rust and C++ using principles of static analysis – Sunny Chatterjee
- C++20 STL Features: 1 Year of Development on GitHub – Stephan T. Lavavej

Wednesday 16th

- Dynamic Polymorphism with Metaclasses and Code Injection – Sy Brand
- Cross-Platform Pitfalls and How to Avoid Them – Erika Sweet
- Effective Remote C++ Development with Codespaces – Nick Uhlenhuth

Friday 18th

- Introducing Microsoft's New Open Source Fuzzing Platform – Justin Campbell & Michael Walker