Chapter 1 : Microsoft Visual C++ Guide - WxWiki

Microsoft Visual C++ Redistributable includes bug fixes to the runtime DLLs and also the latest versions for KB To find out what's new in Visual Studio Update 3, see the Visual Studio Update 3 Release Notes.

There are a variety of other workloads for other languages such as C , and other platforms such as Microsoft Azure for your cloud needs. The workloads you install are not permanent, and you can always change these options by opening the installer and selecting Modify. Once you have made your selection and clicked Install, Visual Studio will begin the installation process. Once it is complete, Visual Studio is all set up and ready to go! Download the demo project Opening Projects or Folders of Code If you open the demo project folder in Windows File Explorer, you will find a variety of different files in addition to some source code. Generally, code organized by Visual Studio appears as a Solution, which contains a collection of Projects. When a codebase is organized this way, it includes a. This approach is ideal for cross-platform projects that will be run from a variety of different IDEs or editors. This guide will not go over Open Folder or CMake, but you are encouraged to check out the relevant blog posts linked in this paragraph for more information. To open demoApplication, double click the. Once you have opened the project, a view of the content in the project will appear in the Solution Explorer, pictured below: Near the top of the IDE inside the standard toolbar, there are dropdowns where you can change your build configuration and architecture. You can also easily add more configurations, as needed. For this exercise, you can leave the default settings of Debug and x The build will fail, since the code contains an error. The Output Window is a valuable tool while you are building; it provides information about the status of the build. Fixing compiler errors You should see an error in the Error List at the bottom of the screen when you attempt to build. With this error, you not only get the location of the problem and a description, but if you double-click the line, you will be brought to the specific location in the code. This makes it easy to quickly navigate to problem areas. Double-click on the error after building, and fix the offending line of code. Intellisense One of the most useful features for helping you write code quickly in Visual Studio is IntelliSense, which is a context-aware code completion tool. As you type, Visual Studio will suggest classes, methods, objects, code snippets, and more symbols that make sense in relation to what you have typed so far and where you have typed it. You can scroll up and down the suggestions with the arrow keys, and complete symbols with Tab. In the main function try adding a call to the farewell function to the mySorter object. You should see IntelliSense pop up with all the possible functions available from the sorter class. Go To To efficiently write and understand code, easy code navigation is essential. When you open the dialog, you can filter your results by clicking on the desired button, or by starting your query with a specific token. Try using Go To to navigate around the demo project. This problem is easily solved in Visual Studio, where you can easily peek into definitions. This will bring up a preview of the file where the function is defined, where you can quickly make small edits. Press Esc to close the preview window. You can also go directly to the definition by pressing only F Add a dash between the numbers in the vector when they are printed. Rename You can also use Visual Studio to refactor existing code. In the demo project, there is a function that has an unhelpful name. This will bring up a menu where you can choose what you want to rename it to, and then preview the changes before they are committed. Debugging and Diagnosing Issues Once you can successfully build your application and write code easily, the next step is often debugging the application. Debugging can be a complex process, and Visual Studio provides many powerful tools to help along the way. If you click on the bar to the left of your code, a red circle should appear. If you click the circle, the breakpoint will be removed. When a breakpoint is set and the program reaches that point of execution, it will stop, allowing you to inspect variables and the current state of the program. Place a breakpoint on line 33 of demoApplication. Click the red circle again to remove the breakpoint. To begin debugging, you can either press the green arrow at the top of the IDE or press F5. Once the program has stopped on the breakpoint, there are many things you can do to help you diagnose problems. One of the best ways to find problems is to understand the current state of the program, versus what it should be. This can be easily achieved by using the Autos Window, which lists recently used variables and their values. You can also hover your mouse over a variable to see what the current value is. Place a breakpoint on line 14 of the main function. Click the green arrow at the top of the IDE or press F5 to begin debugging. Find out what the value of testInt is before it is initialized by hovering over the value in the code. Look at the value of testInt in the Autos window. Press the green arrow or F5 again to stop debugging. When you have sufficiently understood the current state of the program, you can press the green arrow button or press F5 again to have the program run until the next breakpoint. You can also step the program one line at a time if needed by using the arrows at the top. Step Over F10 will run through whatever is on the current line, and suspend execution after the function returns. Step Into F11 will follow the function call of the next line, allowing you to see what is happening inside that function. Use a combination of these to explore the demo project and see if you can fix the logical bug in the sort algorithm Hint: There are many more tools within Visual Studio that can help you profile and debug your applications. Testing Visual Studio has a built-in test framework to help you unit test your projects, ensuring that the code you write is working as expected. To test the demo project, which is a native console application, you can add a Native Unit Test Project to the solution. Add a test project to the demo. Make sure to choose the Add to solution option in the Solution dropdown. Once you have added a unit test, you can open the. Add a test method, making sure that it will pass. Try the following code: Once you have run the tests, you will see the results in the Test Explorer window. Try adding another test that will fail, and running the tests again. Working with A Team It is very common these days to be working on a project with a team, and Visual Studio makes collaboration with others easy! You can easily create new source control repositories using Git or Team Foundation Server to manage your codebase. To create a new repo for a project, click the Add to Source Control button at the bottom of the screen, and add the opened project to the source control system of your choice. Once you do that, a local repository will be made for your project. From here you can make commits, or push your changes to a remote Git service such as GitHub. This is all managed in the Team Explorer window. Try adding the demo project to source control, and pushing it to GitHub. This is done by pressing the Add to source control button, and then pushing to a remote repository inside the Team Explorer. You can also very easily clone from source control from the Team Explorer window. From here, all you must do is paste in the URL, and the project will be cloned. Other Topics There are many other useful things Visual Studio can do. So many things, in fact, it is hard to cover it all in one guide. Follow the links below to find out more on how to get the most out of Visual Studio. Code Analysis Visual Studio by default catches a lot of code issues, but its Code Analysis tool can often uncover hard-to-find issues that would normally be missed. Common errors that are reported include buffer overflows, uninitialized memory, null pointer dereferences, and memory and resource leaks. This functionality is built into the IDE, and can easily be used to help you write better code. While Visual Studio has support for NuGet package management, more recently a new tool called vcpkg was launched. Vcpkg is an open source tool maintained by Microsoft that simplifies acquiring and building open source libraries, with over currently supported. Check out the announcement blog post for details. Conclusion We hope that this guide has allowed you to get up to speed with Visual Studio quickly, and that you have learned some of the core functionality. This should be enough to get you started, but there are still many more features that could not be covered in this guide. You can find the comprehensive product documentation on docs. Now get out there and build something amazing! We are constantly trying to improve, so if you have any feedback or suggestions for us, please feel free to reach out to us anytime! We can be reached via email at visualcpp at microsoft.

Chapter 2 : Download Visual C++ Redistributable for Visual Studio from Official Microsoft Download Cente

The Visual C++ Redistributable Packages install run-time components of Visual C++ libraries. These components are required to run C++ applications that are developed using Visual Studio and link dynamically to Visual C++ libraries.

Thank you for reporting bugs! Visual Studio version For more information, see Announcing: This option is on by default in Visual Studio version Enable display of the line number, the line number and column, or the line number and column and a caret under the line of code where the diagnostic error or warning was found. Visual Studio by not copying all debug information into the PDB file. The PDB file instead points to the debug information for the object and library files used to create the executable. Codegen, security, diagnostics and versioning This release brings several improvements in optimization, code generation, toolset versioning, and diagnostics. Some notable improvements include: Improved code generation of loops: Support for automatic vectorization of division of constant integers, better identification of memset patterns. The compiler and related build tools have a new location and directory structure on your development machine. The new layout enables side-by-side installations of multiple versions of the compiler. The output window now shows the column where an error occurs. Additional improvements to diagnostics in the compiler. For more information, see Diagnostic Improvements in Visual Studio This means that you can simply recompile your code, and your app runs faster. Some of the compiler optimizations are brand new, such as the vectorization of conditional scalar stores, the combining of calls sin x and cos x into a new sincos x , and the elimination of redundant instructions from the SSA Optimizer. For more information, see Dynamic exception specification removal and noexcept. New diagnostic warning for Spectre migitation. See Spectre diagnostic in Visual Studio Version Tripping an IDL check in string machinery will now report the specific behavior that caused the trip. Previously this was implemented as a general string search for a string of length 1. Note that for all short strings, calling reserve still has a nonzero cost to do nothing. Slightly improved compiler diagnostics for incorrect bind calls. Improved the performance of std:: Note that this will disable std:: Correctness fixes in Visual Studio version This ensures that the result of distance on iterators from that container is representable in the return type of distance. Now it will wait for only a single interval of the relative time. Some warnings reported by Clang -Wsystem-headers were fixed. Also fixed "exception specification in declaration does not match previous declaration" reported by Clang -Wmicrosoft-exception-spec. Also fixed mem-initializer-list ordering warnings reported by Clang and C1XX. The unordered containers did not swap their hashers or predicates when the containers themselves were swapped. Changed include directives to use proper case sensitivity and forward slashes, improving portability. Fixed uses-allocator metaprogramming in tuple. Removed metaprogramming for customized allocator:: Allocators can make containers use fancy pointers but not fancy references. The compiler front-end was taught to unwrap debug iterators in range-based for-loops, improving the performance of debug builds. For example, inserting at the beginning of a string now moves the content after the insertion exactly once either down or to the newly allocated buffer , instead of twice in the reallocating case to the newly allocated buffer and then down. Standard Library algorithms now avoid postincrementing iterators. Fixed truncation warnings when using bit allocators on bit systems. The Standard Library now uses alias templates internally. Internal usage of NULL has been eradicated. Internal usage of 0-as-null is being cleaned up gradually. The Standard Library now uses std:: This improves compiler diagnostics because error immediately stops compilation. Modern linker technology no longer requires this. Extracted SFINAE to default template arguments, which reduces clutter compared to return types and function argument types. New experimental features Experimental support for the following parallel algorithms:

Chapter 3 : Visual C++ Documentation | Microsoft Docs

Create classic Windows desktop applications, static libs and DLLs using Win32, MFC, ATL, COM, and/or C++/CLI. Use Python Tools for Visual Studio to create Python applications that interact with C++ programs. Upgrade to the latest version of Visual Studio and migrate legacy apps to modern platforms.

This is primarily because the tag-parser currently does not parse function bodies. We are working hard to make this happen. See Guidance on how to configure for better IntelliSense results. In other words, squiggles are not enabled when only single files are open. In this extension, this used to be performed by the tag parser, which provides quick but fuzzy results â€” sometimes inaccurate. With the new IntelliSense engine, we can provide more accurate results for local and global variables, functions, and symbols. The tag parser continues to provide suggestions for both cases. Hints will also be presented for template arguments. Previously, the extension returned hints for all functions with a matching name, regardless of type. Reference highlighting Moving the text cursor over a symbol in the editor will highlight the matching symbols in the same file. You can use the editor. Currently clang-format needs to be installed manually and the path for clang-format needs to be added to user settings in Visual Studio Code. You can learn more here about how to setup clang-format for your code-formatting experience. The debugging experience currently only works out-of-the-box for Linux â€” Ubuntu We are working on OS X support as well, the current experience requires manual installation steps and has some limitations. Navigate to the Debug View by clicking on the debug icon on far left toolbar. We include two configurations by defaultâ€"one which shows the properties necessary for launching the application from VS Code, and the other which shows the properties necessary for attaching to an already running process. You can learn more about the properties in the launch. Note that VS Code will not build your program but simply debug the built program unless you also create a task. Function Breakpoints Function breakpoints enable you to break execution at the beginning of a function rather than on a particular line of code. You can type an expression into the Watch pane, and it will be evaluated when a breakpoint is hit. Note that this has side effectsâ€"an expression that modifies the value of a variable will modify that value for the duration of program execution. If you only want to evaluate an expression once rather than having it in the Watch pane , you can simply type the expression in the Debug Console. Type a condition eg. Conditional breakpoints are indicated by a breakpoint symbol that has the black equals sign. Hovering over a conditional breakpoint will show its value. Core Dump Debugging The extension also offers the ability to debug using a memory dump. This will even work for multi-threaded programs and x86 programs being debugged on a x64 machine. A full introduction to debugging in VS Code is available here. Process Picker to attach the debugger to a running process easily VS Code now enables you to select a process from a list of running processes rather than needing to manually enter the process id into the launch. To use the process picker: If you are using an existing launch. If you allow VS Code to generate the launch. When you start debugging, focus will go to the VS Code quick launch bar, and a list of running processes will appear. This post is going to demonstrate how using task extensibility in Visual Studio Code you can call compilers, build systems and other external tasks through the help of the following sections:

Chapter 4 : What's New for Visual C++ in Visual Studio | Microsoft Docs

Program C:\Users\cmwa This application has requested the Runtime to terminate it an unusual way. Please contact the application's support team for more information.".

There was also a Microsoft QuickC 2. It was being used inside Microsoft for Windows and Xenix development in early It shipped as a product in Microsoft Fortran and the first 32 bit compiler for were also part of this project. It was Cfront 2. Included the ability to build both DOS and Windows applications, an optimizing compiler , a source profiler , and the Windows 3. It is the last, and arguably most popular, development platform for Microsoft Windows 3. It is available through Microsoft Developer Network. In many ways, this version was ahead of its time, since Windows 95 , then codenamed "Chicago", was not yet released, and Windows NT had only a small market share. As a result, this release was almost a "lost generation". To allow support of legacy Windows 3. There are, however, issues with this version under Windows XP, especially under the debugging mode for example, the values of static variables do not display. Accordingly, the English language upgrade version of Visual Studio. This was the last version to support Windows 95 and NT 4. It also introduced OpenMP. It uses a SQL Server Compact database to store information about the source code, including IntelliSense information, for better IntelliSense and code-completion support. Variadic templates were also considered, but delayed until some future version due to having a lower priority, which stemmed from the fact that, unlike other costly-to-implement features lambda, rvalue references , variadic templates would benefit only a minority of library writers rather than the majority of compiler end users. The RTM version After some users contacted Microsoft about this problem, Microsoft said they would remove these telemetry calls when compiling with the future Visual Studio Update 3. NET version 7. It does not refer to the year in the name of the Visual Studio release. A thorough list is available. A typical example is a program using different libraries. There had been no plans to support C99 even in , more than a decade after its publication [49]. When implemented I expect the feature to work independent of if the traditional or updated preprocessor logic is used.

Chapter 5 : C/C++ extension for Visual Studio Code | Visual C++ Team Blog

Microsoft Visual C++: Development System for Windows 95 and Windows NT, Version 4 The first volume of a six-volume documentation collection for Microsoft Visual C++ version for Win The documentation draws its information from the online help system in Microsoft Visual C++.

This includes getting the code to compile and run correctly on a newer release of the tools, as well as taking advantage of new language and Visual Studio features. This topic also includes information about migrating legacy apps to more modern platforms. Faster code, due to improved compiler optimizations. Faster builds, due to performance improvements in the compiler itself. Security features such as guard checking. Is your application built with Visual Studio? If so, identify the projects involved. Do you have custom build scripts? The build system and project file format in Visual Studio changed from vcbuild in versions up to Visual Studio to MSBuild in versions of Visual Studio from onwards. If your upgrade is from a version prior to , and you have a highly customized build system, you might have to do more work to upgrade. If you are upgrading from Visual Studio or later, your projects are already using MSBuild, so upgrading the project and build for your application should be easier. If you upgrade to use MSBuild, you might have an easier time in future upgrades, and it will be easier to use services such as Visual Studio Online. MSBuild supports all the target platforms that Visual Studio supports. Porting Visual Studio Projects To start upgrading a project or solution, just open the solution in the new version of Visual Studio, and follow the prompts to start upgrading it. When you upgrade a project, you get an upgrade report, which is also saved in your project folder as UpgradeLog. The upgrade report shows a summary of what problems were encountered during the upgrade process and some information about changes that were made, or problems that could not be addressed automatically. Runtime errors or unexpected results due to behavior changes Errors due to errors that were introduced in the tools. New warnings might mean you want to clean up your code. Depending on the specific diagnostics, this can improve the portability, standards conformance, and security of your code. You might want to update some code to use new language features that simplify the code, improve the performance of your programs, or update the code to use modern libraries and conform to modern standards and best practices. Will it receive ongoing development work? Will it be important for your code to run on other platforms? If so, what platforms? Do you want to refactor your code, to separate out those parts that are more tied to the Windows platform? What about your user interface? If you just want to give your app a newer look and feel, without rewriting the entire app, you might consider using the ribbon APIs in MFC, or using some of new features of MFC. In Windows 10, you can use the Desktop App Converter to package your existing desktop application as a UWP app without having to modify any code. Alternatively, perhaps you now have new requirements, or you can foresee the need for targeting platforms other than Windows desktop, such as Windows Phone, or Android devices. You could port your user interface code to a cross-platform UI library. With these UI frameworks, you can target multiple devices and still use Visual Studio and the Visual Studio debugger as your development environment.

Chapter 6 : Microsoft Visual C++ - Microsoft Community

Microsoft Visual C++ Redistributable Package(vcredist_xexe) was installed on the computer which Visual Studio is not installed on to be able to run calendrierdelascience.com file, but I can't run the file. I can run calendrierdelascience.com file on the computer which Visual Studio is installed on.

Chapter 7 : Microsoft Visual C++ - Wikipedia

C:\Users\Mike\AppData\Local\Packages\calendrierdelascience.comoftEdge_8wekyb3d8bbwe\TempState\ This *application has requested the Runtime to terminate it in an unusual way. Please contact the application's support team for more information.*

Chapter 8 : Visual C++ in Visual Studio

Installed->Templates->Visual C++->Visual Studio Solutions->Blank Solution Locate the sample you want to use (this is the folder with the vcproj file in). Manually (outside of Visual Studio) copy this project folder into your new solution.

Chapter 9 : Visual C++ in Visual Studio | Microsoft Docs

Microsoft Visual C++ Redistributable Package (x86) Important! Selecting a language below will dynamically change the complete page content to that language.