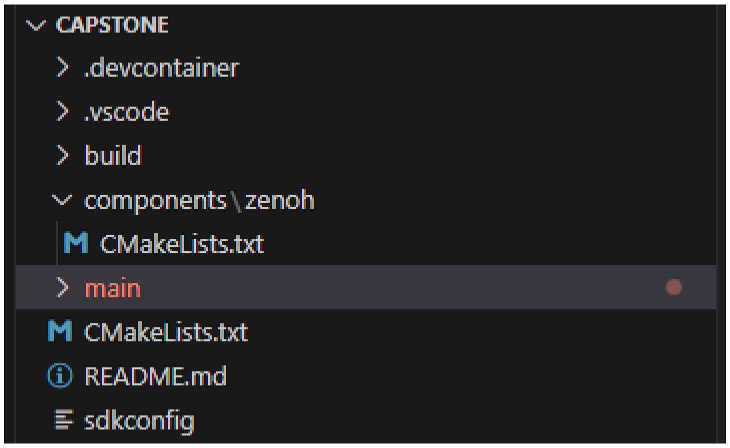
Setup Project with Git

1- First you need to get the ESP-IDF development environment working in VSCode. See the ‘vscode espidf zenoh setup’ pdf document for help with that.

2- Once that works, create a blank ESP-IDF project and add zenoh as component.

   a. The file structure should look like this in VSC



```
CAPSTONE
  .devcontainer
  .vscode
  build
  components\zenoh
    CMakeLists.txt
  main
  CMakeLists.txt
  README.md
  sdkconfig
```

3- Install git. To check if you already have git installed on your system, go to command prompt (on Windows) and type the command ‘git –version’.

4- Clone our repository 'JAMscript-ESP32-port' into the **main** folder. Open the terminal and navigate to this main directory and run the command

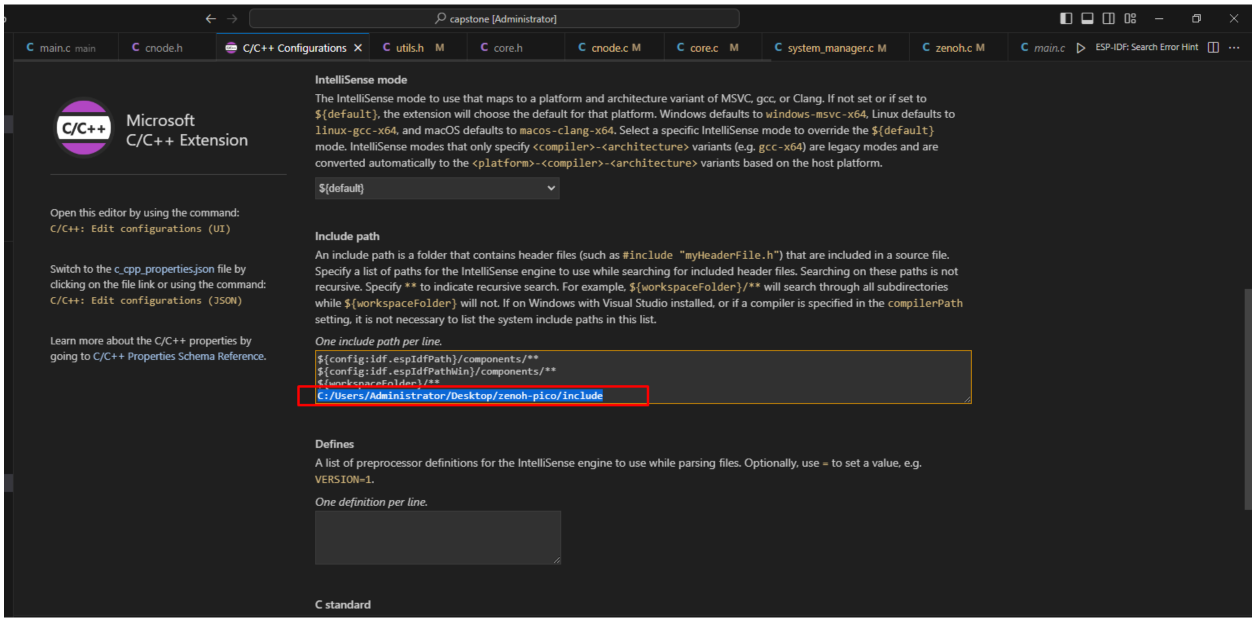
git clone https://github.com/blissfulcat/JAMscript-ESP32-port.git

You should see a new folder created inside of the main folder containing all our source code.

5- Modify the CMakeLists.txt file **in the main folder** as follows:

```
idf_component_register(
                    SRC_DIRS "."
                    SRC_DIRS "JAMscript-ESP32-port/cside-main/src"
                    INCLUDE_DIRS "."
                    INCLUDE_DIRS "JAMscript-ESP32-port/cside-main/inc")
```

6- Try to build the project. Everything should build correctly. If everything builds and you go to the source and header files and notice a lot of red squiggly lines related to include errors, then you might have to modify 'includePaths' in the C/C++ extension on VSC to add the include path to the zenoh-pico library as follows:



Open this editor by using the command:
`C/C++: Edit configurations (UI)`

Switch to the c_cpp_properties.json file by clicking on the file link or using the command:
`C/C++: Edit configurations (JSON)`

Learn more about the C/C++ properties by going to C/C++ Properties Schema Reference.

IntelliSense mode

The IntelliSense mode to use that maps to a platform and architecture variant of MSVC, gcc, or Clang. If not set or if set to `${default}`, the extension will choose the default for that platform. Windows defaults to `windows-msvc-x64`, Linux defaults to `linux-gcc-x64`, and macOS defaults to `macos-clang-x64`. Select a specific IntelliSense mode to override the `${default}` mode. IntelliSense modes that only specify `<compiler>-<architecture>` variants (e.g. `gcc-x64`) are legacy modes and are converted automatically to the `<platform>-<compiler>-<architecture>` variants based on the host platform.

${default}

Include path

An include path is a folder that contains header files (such as `#include "myHeaderFile.h"`) that are included in a source file. Specify a list of paths for the IntelliSense engine to use while searching for included header files. Searching on these paths is not recursive. Specify `**` to indicate recursive search. For example, `${workspaceFolder}/**` will search through all subdirectories while `${workspaceFolder}` will not. If on Windows with Visual Studio installed, or if a compiler is specified in the `compilerPath` setting, it is not necessary to list the system include paths in this list.

One include path per line.

```
${config:idf.espIdfPath}/components/**
${config:idf.espIdfPathWin}/components/**
${workspaceFolder}/**
C:/Users/Administrator/Desktop/zenoh-pico/include
```

Defines

A list of preprocessor definitions for the IntelliSense engine to use while parsing files. Optionally, use `=` to set a value, e.g. `VERSION=1`.

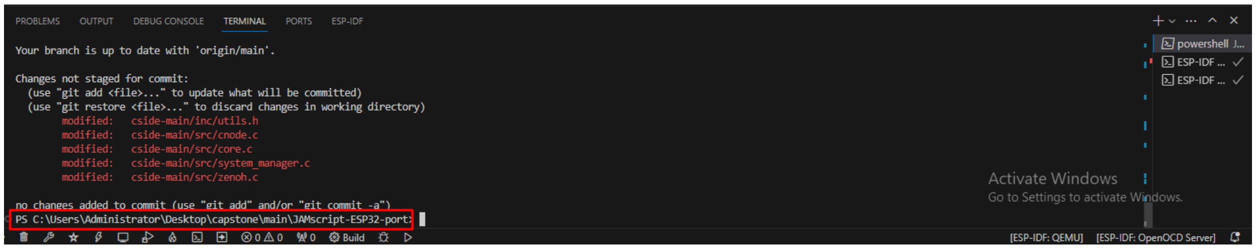
One definition per line.

C standard

How to push your changes to GitHub using Git

You can now navigate using the terminal to the 'JAMscript-ESP32-port' folder and run git commands to commit and push changes to the GitHub repository.

a. Open the PowerShell terminal in VSC and make sure you are in the right directory. It should look like



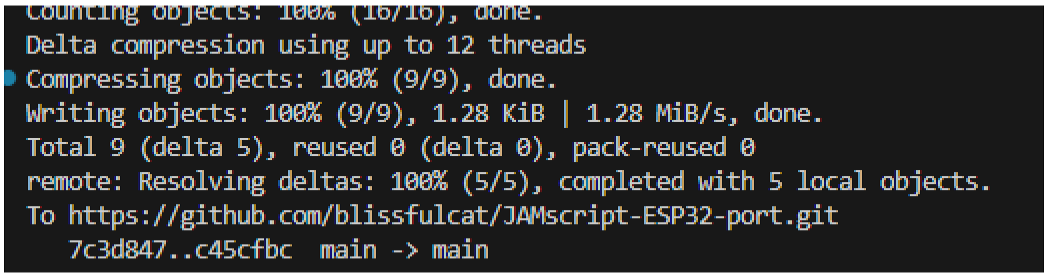
```
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   cside-main/inc/utils.h
        modified:   cside-main/src/cnode.c
        modified:   cside-main/src/core.c
        modified:   cside-main/src/system_manager.c
        modified:   cside-main/src/zenoh.c

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Administrator\Desktop\capstone\main\JAMscript-ESP32-port>
```

b. Run the command 'git status' to see what files you have modified and which files you have committed.

c. Run the command 'git add .' command to add all of the files that you changed to the next commit. Running 'git status' again you should see the files that are ready to be committed. MAKE SURE THAT YOU DON'T ACCIDENTALLY COMMIT A BUNCH OF RANDOM UNWANTED FILES.

d. Run the command 'git commit -m [MESSAGE]' where MESSAGE is the commit message that will show up on GitHub once you push the changes. Preferably stick to https://www.conventionalcommits.org/en/v1.0.0/#summary as a general convention guide for commit messages.

e. When you are finally ready to push your changes, run the command 'git push'. You might be prompted to login to GitHub using your browser for authentication reasons. If successful you should see a message looking like this:



```
Counting objects: 100% (16/16), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 1.28 KiB | 1.28 MiB/s, done.
Total 9 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To https://github.com/blissfulcat/JAMscript-ESP32-port.git
   7c3d847..c45cfbc  main -> main
```