Inventory Data Structure

How to Start

This document explains how to get started with the code. The code can be used in the following mentioned two ways:

- Method One: Using the Library and Header
- Method Two: Using the Source Code

Method One: Using the Library and Header

Firstly, add library and header file to code blocks. The files can be added by making them visible at the global level. Adding these files at global level avoids adding them to each and every project.

Adding Library:

1. Go to Settings -> Compiler and debugger settings -> Linker settings -> Add
2. Then navigate to the library file 'libIncludeInven.a' and say OK

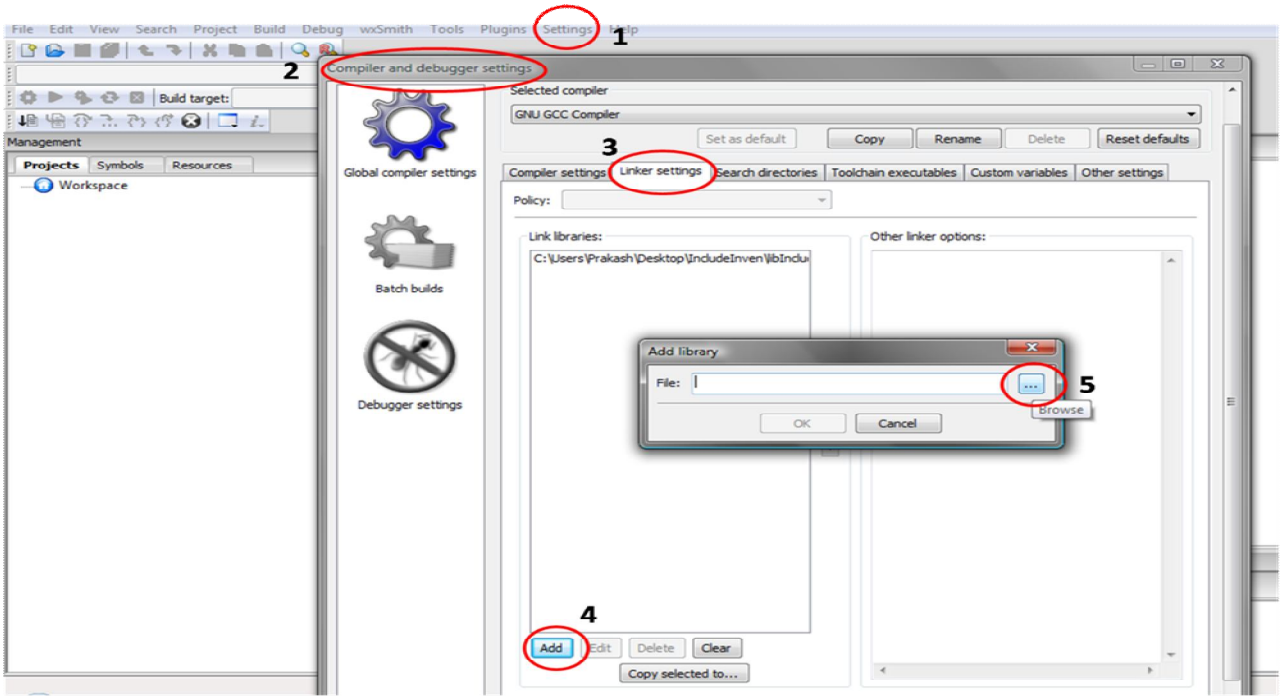

Fig 1: Adding a library

Adding Header:

For this the directory which holds the header file has to be added.

1. Go to Settings -> Compiler settings -> Search directories -> Compiler -> Add

2. Navigate to ‘inventory-header’ directory from navigation widow and click OK (The directory has to be added. Not the file inside it)

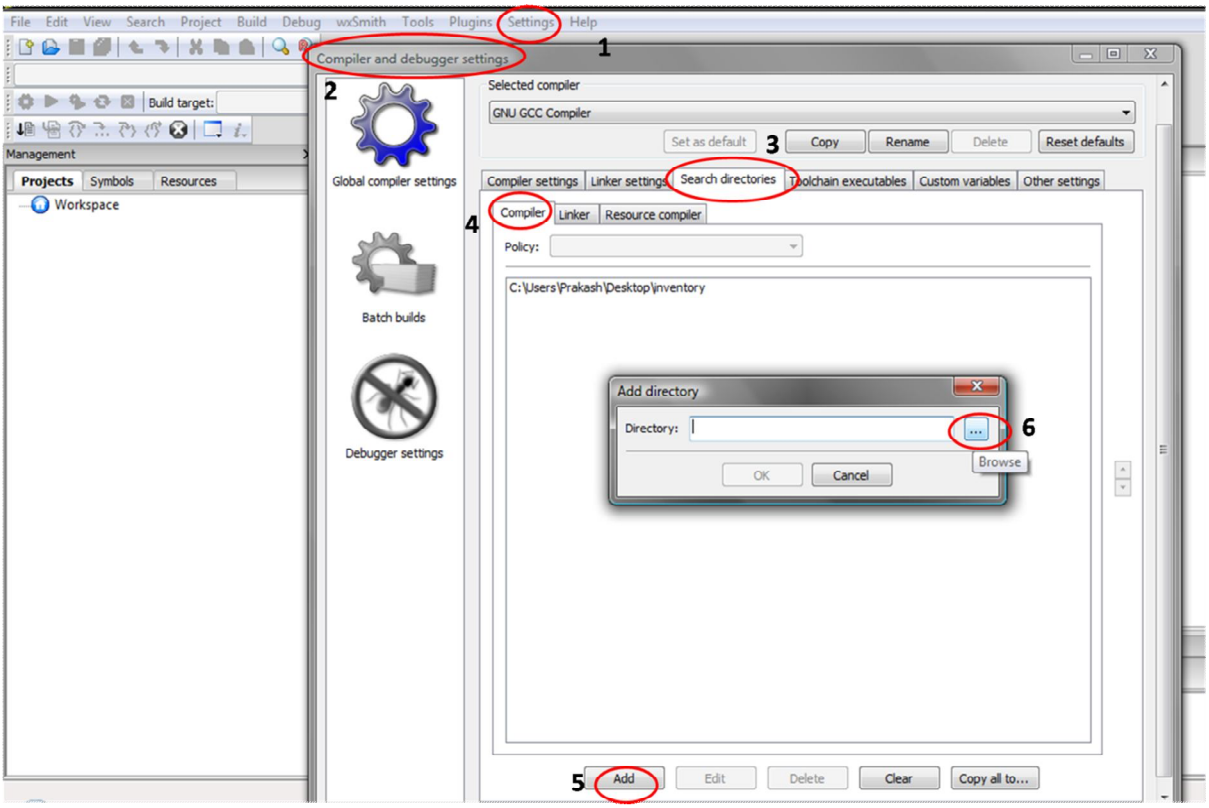

Fig 2: Adding a header

What Next?

1. Copy the supplied ‘main.c’ into your project

2. Copy the text files placed in ‘common-files’ directory into your project directory

You are now good to run the project!

Method Two: Using the Source Code

Create a new project and include all the supplied header and source files. You can Google out how to add files to a project directory if you are not aware of the process. Copy all the text files supplied in the 'common-files' directory. You are now good to run the project.

Before you compile the code, spend ample time to understand each file. The code is commented to understand the on-going details.

Understanding Common Files

'common-files' and 'main.c' file are tagged to each other. The main.c which is supplied works in line with the given common-files.

Having your own common-files and main

Once you understand the working of code properly, you can add your own file and main. You need to populate three files. All the three files have to be placed in the project directory. The first file is the "inventory_file.txt" file. This file will have all the entries made is required format. There should be atleast one entry made. The remaining entries can be added later.

An example file looks the following way:

```
NULL cereals meat junkfood milk
name sprouts beef noodles  butter
proteins 40 60 NULL 20
carbohydrates 28 71 20 32
fat 20 40 40 30
cholesterol 10 80 70 25
glucose NULL 30 NULL 10
sucrose NULL NULL NULL NULL
vitamin 30 40 10 20
iron 60 60 20 40
```

The second file to be populated is the “key_count.txt”. This file will have the number of key counts. In the given example the file has to be written with the count - 4. In the file write only the integer number. The third file to be populated is the “property_count.txt”. This file will have the number of property counts. In the given example the file has to be written with the count – 9. In the file write only the integer number.

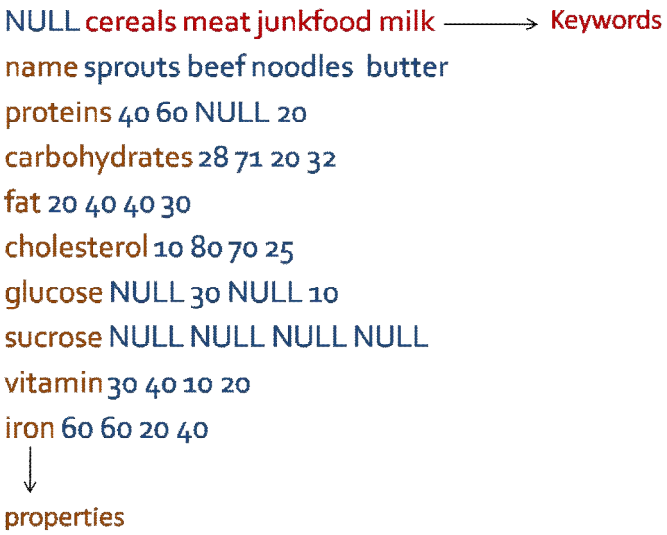

Fig 3: File Content Illustration

You are now good to go with main.c file. Given below is the minimum file you need to start with:

```
int main()
{
     int status = 0;
     status = load();
     if(status != 1){
          printf("Load Failed\n");
          return 0;
     }
     status = dump();
     if(status != 1)
         printf("Dump Failed\n. Please see the tempfile.txt\n");
     return 0;
}
```

Adding a New Key

To add a new key, you need to write the data into file and place it into project directory. The contents of the file have to be in the following format:

```
NULL keyword-name
property-name1 value1
property-name2 value2
property-name3 value3
property-namen valuen
```

You can then use the add_key() to add the data from the file.

Logging

When you run the project it creates a log file in the project directory. The status of success or failure of the run can be seen in the file. It logs the details along with the name of function.

A sample log entry can be seen below:

```
_____________________________START OF ITERATION_____________________________

Thu Jun 30 01:42:29 2016
 file_empty_check : SUCCESS --> key_count.txt has content

Thu Jun 30 01:42:29 2016
 file_empty_check : SUCCESS --> property_count.txt has content

Thu Jun 30 01:42:29 2016
 file_empty_check : SUCCESS --> inventory_file.txt has content

Thu Jun 30 01:42:29 2016
 load : SUCCESS --> Key count read successfully

Thu Jun 30 01:42:29 2016
 load : SUCCESS --> Property count read successfully
```

Note:

It's a PH and VT effort