To: Team 5914 Programming Group

From: Larry Basegio

Subject: Method for Creating a (non-robotic) Java Program in VS

Date: 4 November 2019

Outside of creating a robot program it may be useful to learn the basics of Java language by creating a much simpler, less complex, generic computer program. Examples are "Hello World", simple math exercises, and small test programs to evaluate or develop a method that might be used as a subroutine within a robot program or any program for that matter. I have essentially had to "hack" through this, in that I have not run into documentation that describes the process.

General Method

1. Create a workspace file
   - File -> Save Workspace As: A dialog will appear that allows you to specify a file name with the extension ".code-workspace".
   - Click the Save Button after verifying that it will be saved into the correct directory. On my computer that directory would be c:\Users\3Zman\VS Workspace. The file name will have the format ProjectName.code-workspace. Hit "Save".
   - Create and name a subdirectory of VS Workspace – this will be where the source code files (classes) (xxx.java files) will be stored. You will have to create this subdirectory. In the example listed on the following page the subdirectory name is "ProjectCode".
   - Click File->Add Folder to Workspace. Select the subdirectory that you just created.
   - Within this subdirectory create the program that contains the entry point for the program. This be a file that is saved with the extension ".java". It will take the general form as listed below – in this instance it is a simple "Hello World" application. Note that all code is encapsulated within a class structure. In this example the class name is "project". The name of the file can be anything you decide. It is helpful if the name has something to do with the intended application.

     Procedure: File->New file. It will appear as Untitled-1. Now save the file with a new name (e.g., "filename.java" in the subdirectory and type code similar to what is listed below.

```
import java.io.*  //  This allows use of the input/output libraries of java.
import ******   //   list all the imports (e.g. math, etc.)  intended to be used within this file.

class project  {
   public static void main(String args[])  {      //  Type exactly as indicated.
      System.out.println("Hello World");       //  This function will output "Hello World".
   }
}
```

- Open the file ProjectName.code-workspace. You will need to specify where the project will find the code. It takes a format like the example on the following page. Note that the specified path is the name of the subdirectory specified earlier. When you open the file in the editor it may be blank or have part of what is shown below.

```
{
    "folders": [

        {
            "path": "ProjectCode"
        }


    ],
    "settings": {}
}
```

- Create any source files that will be used in your program and stuff them into the directory or directories specified by the "path": statement(s). Use the exact syntax.

Here is another example: in this example relevant source files are located in two directories.

```
{
    "folders": [

        {
            "path": "TestCase"
        },
        {
            "path": "polynomials"
        }

    ],
    "settings": {}
}
```

Again, note the exact syntax. Also note that both are subdirectories of the location of the project .code-workspace file. If they weren't, more detail would need to be provided as to their location.

An interesting feature of the VS system is the creation of a launch.json. This is created when the program is run. An example of this is shown below. The name is the name of the file that contains main(), the mainClass is the name of the class that encapsulates main(). This is just an example, the two names do not need to be identical.

```
{
    "configurations": [
        {
            "type": "java",
            "name": "CodeLens (Launch) - MultiThreadTest",
            "request": "launch",
            "mainClass": "MultiThreadTest"
        }
    ]
}
```

I hope this is a start to using VS system to write test routines that might be used within a robot program.

Revisions:

Initial : 14 October 2019

Modified 4 November 2019: Added clarifications for program setup, example of the launch.json file.