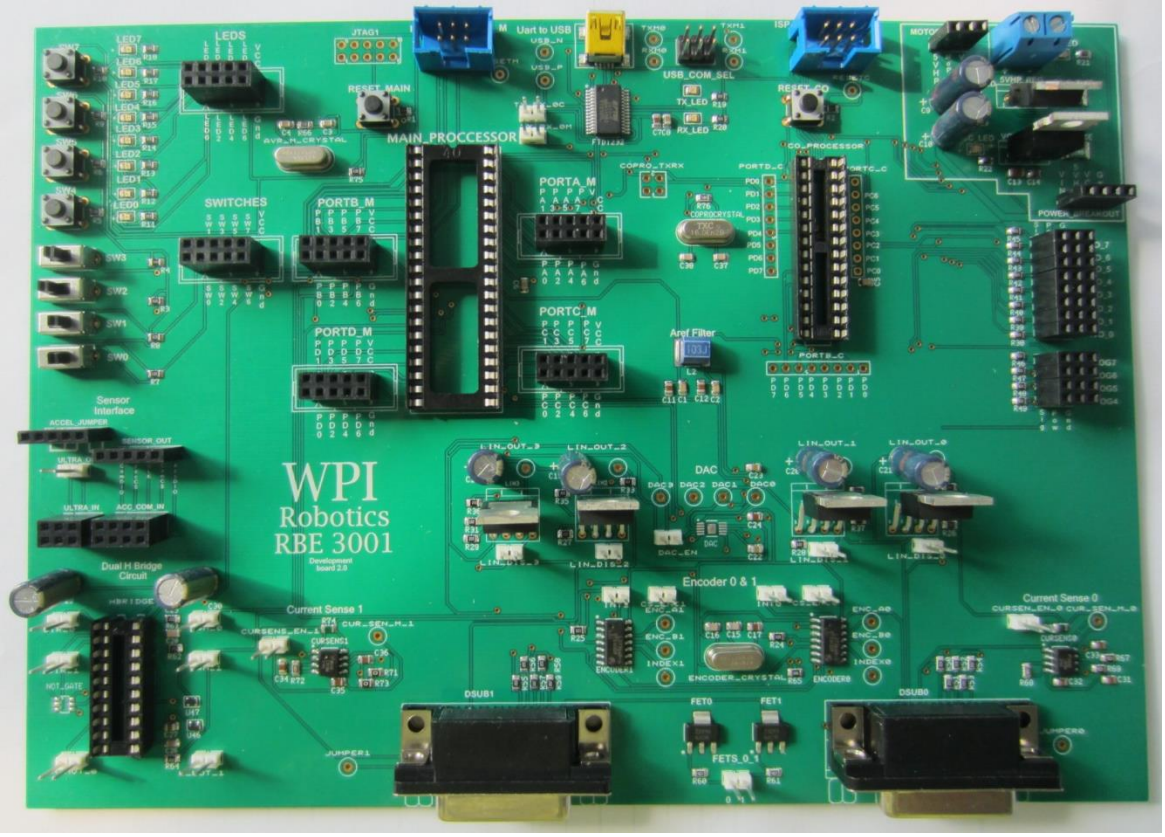
Rbe 3001 Development board guide



The RBE Development board is designed to use an AVR644P to control a robotics system. The Board has multiple functions and features.

Wiki

3001 Wiki: http://wiki.wpi.edu/robotics/RBE_3001

Processors

AVR 644P Main Processor

http://wiki.wpi.edu/images//images/b/be/Atmel-8011-8-bit-avr-microcontroller-atmega164p-324p-644p_datasheet_%281%29.pdf

AVR 328P Co Processor

http://wiki.wpi.edu/images/images/9/9b/328p.pdf

Programmer

ISP Programming ports for both processors

http://wiki.wpi.edu/images//images/5/5a/Pololu_usb_avr_programmer.pdf

USB Communication

FTDI 232 USART to USB serial interface

http://wiki.wpi.edu/images//images/a/a4/DS_FT232R.pdf

SPI Devices

1 -12 bit 4 channel SPI Digital to Analog Converter (DAC)

http://wiki.wpi.edu/images//images/3/3e/10_Bit_DAC_3001.pdf

2- SPI Quadrature encoder counters

http://wiki.wpi.edu/images//images/f/f1/LS7366R.pdf

MOSFETS

4 -High Power linear Op Amps (Linear Drivers)

http://wiki.wpi.edu/images//images/5/59/Opa548.pdf

2-High Power Sinking MOSFETS

http://wiki.wpi.edu/images//images/9/99/ZXMN4A06G.pdf

Analog Sensing

2-Analog Current Sense circuits

http://wiki.wpi.edu/images//images/5/51/Lmp8601.pdf

Other

8-LED's

8-Switches

1-Sensor Interface for WPI Sensor Boards

8-Coprocessor Controlled Servo Ports

4-Analog Input ports

1-H-bridge with 2 motor outputs

AVR 644P

The 644P is a high performance, low power, low cost 8 bit microcontroller. The microcontroller has 64 Kbytes of Flash memory, JTAG and ISP programming ports, 2-8bit Timers/Counters, 1-16bit Timer/Counter, 8 channel 10-bit Analog-To-Digital-Converter (ADC), 2-Serial USARTs, Master/Slave SPI. The microcontroller is a 5V, 40 Pin Wide DIP package with an 18.432 MHz External Crystal. It also has an internal oscillator that is not used in RBE 3001.

We are Using the Eclipse C/C++ IDE with WINAVR and the Eclipse-AVR Plug-in to program the AVR 644P, Along with the SVN plug-in to access the Fusion forge Project.

Eclipse Install

1. Download Eclipse IDE for C/C++ Developers

http://www.eclipse.org/downloads/

   a. How to tell if you are running 32-bit or 64-bit

   http://support.microsoft.com/kb/827218

2. Extract the Eclipse folder from the downloaded zip file to a desired location and run Eclipse from the extracted folder.
3. Choose a workspace (where your code will be stored) and press OK.

WinAVR

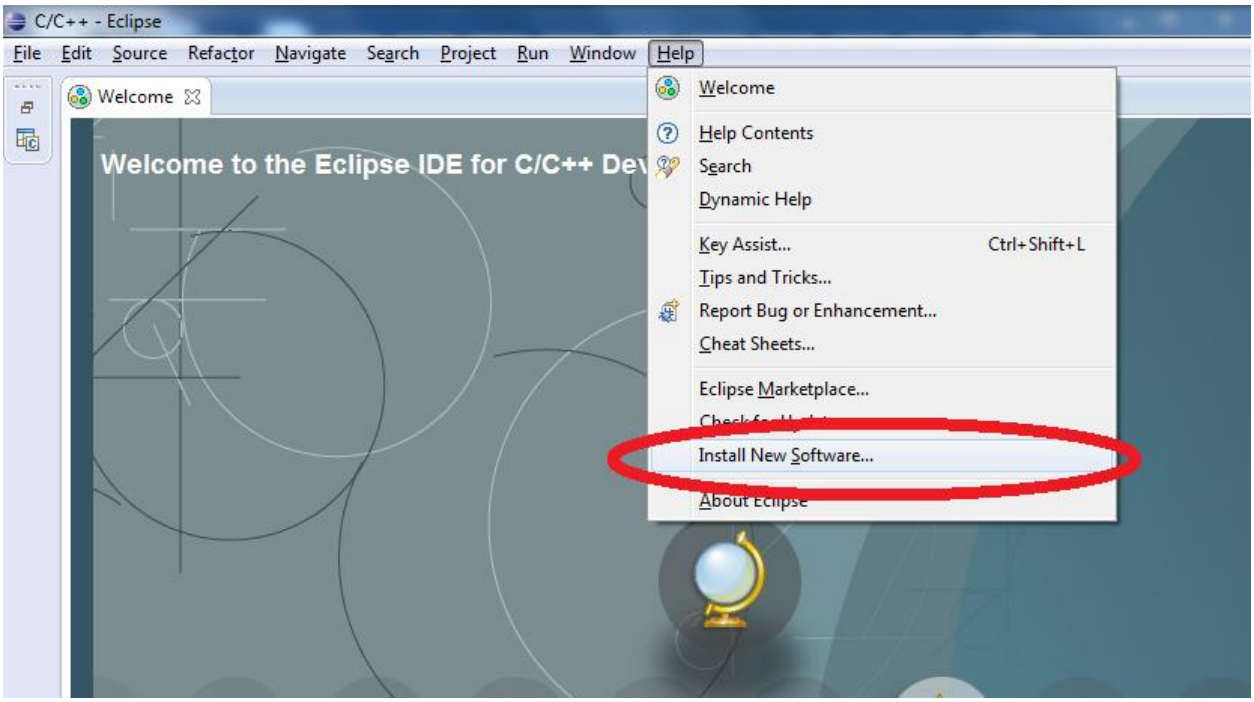
1. Download WinAVR at http://sourceforge.net/projects/winavr/files/.
2. Open the WinAVR folder and open 20100110 (the most recent version).
3. Download and install WinAVR-20100110-install.exe with the default settings.

JDK

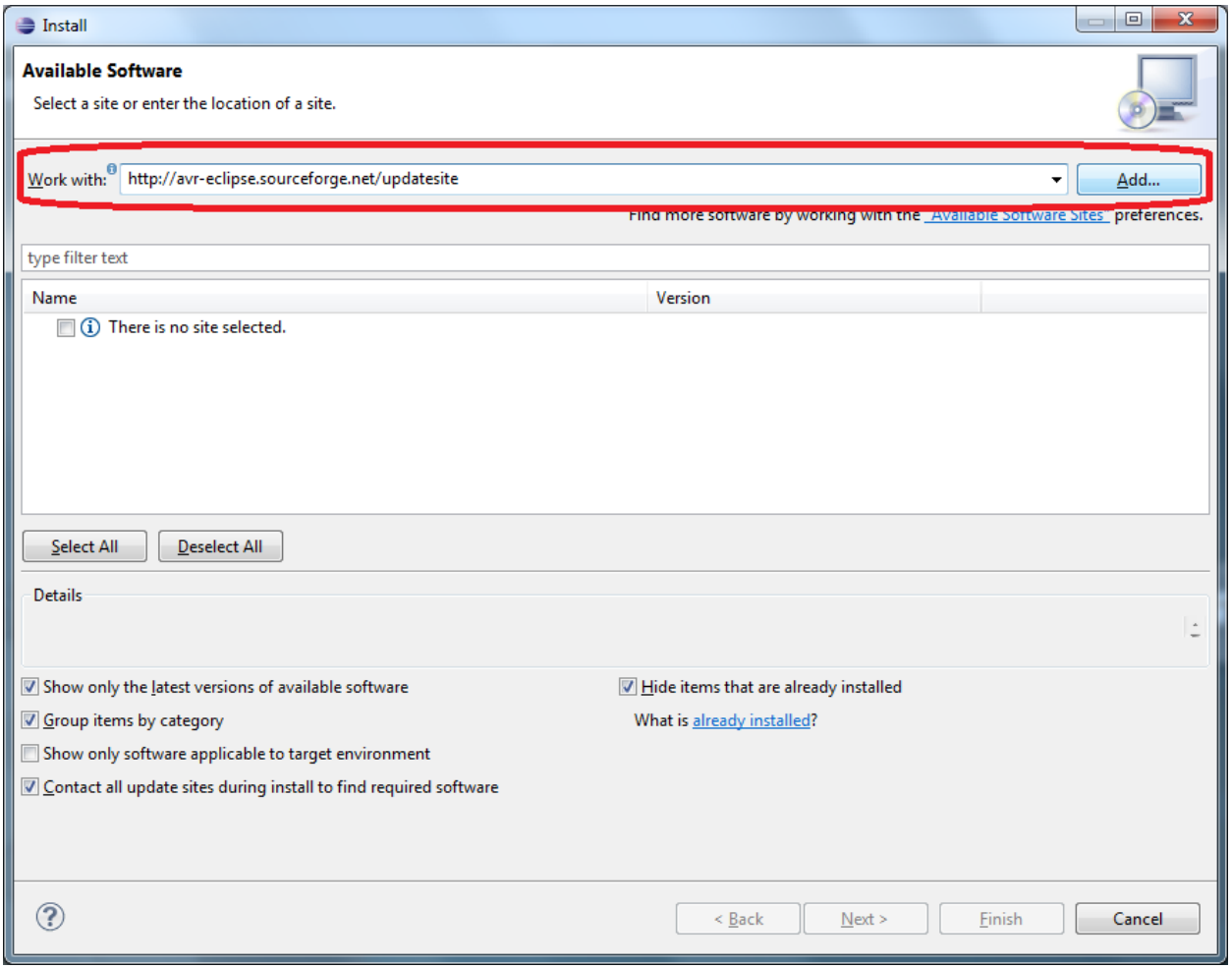
1. Download the Java Development kit at http://www.oracle.com/technetwork/java/javase/downloads/index.html by clicking on the Download button under JDK.
2. Accept the license agreement and download the 32-bit (x86) or 64-bit (x64) depending on your version of Windows.
3. Install it with the default options.

AVR and SVN Install

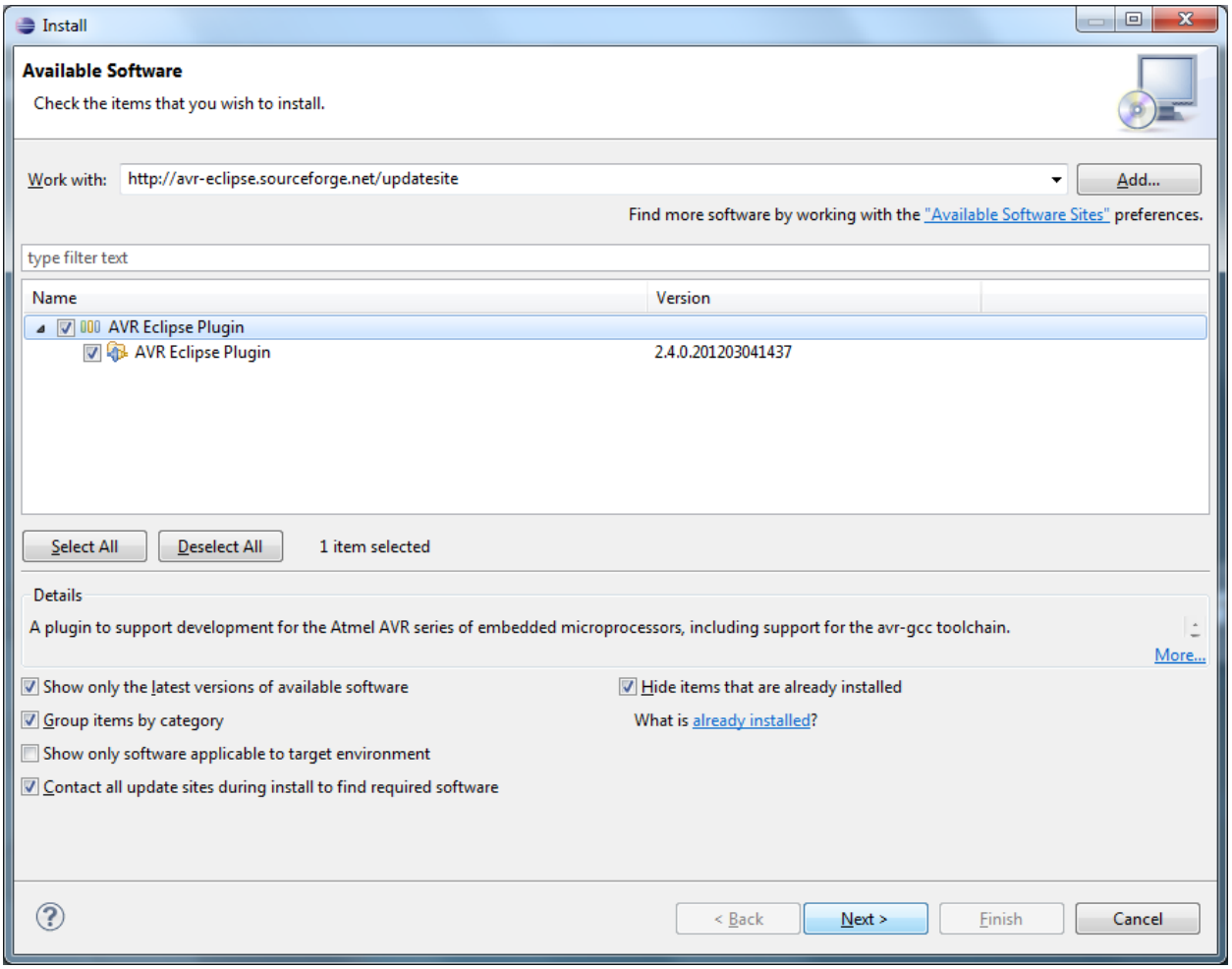
1. Use Eclipse to install the AVR plugin (Help > Install new software)



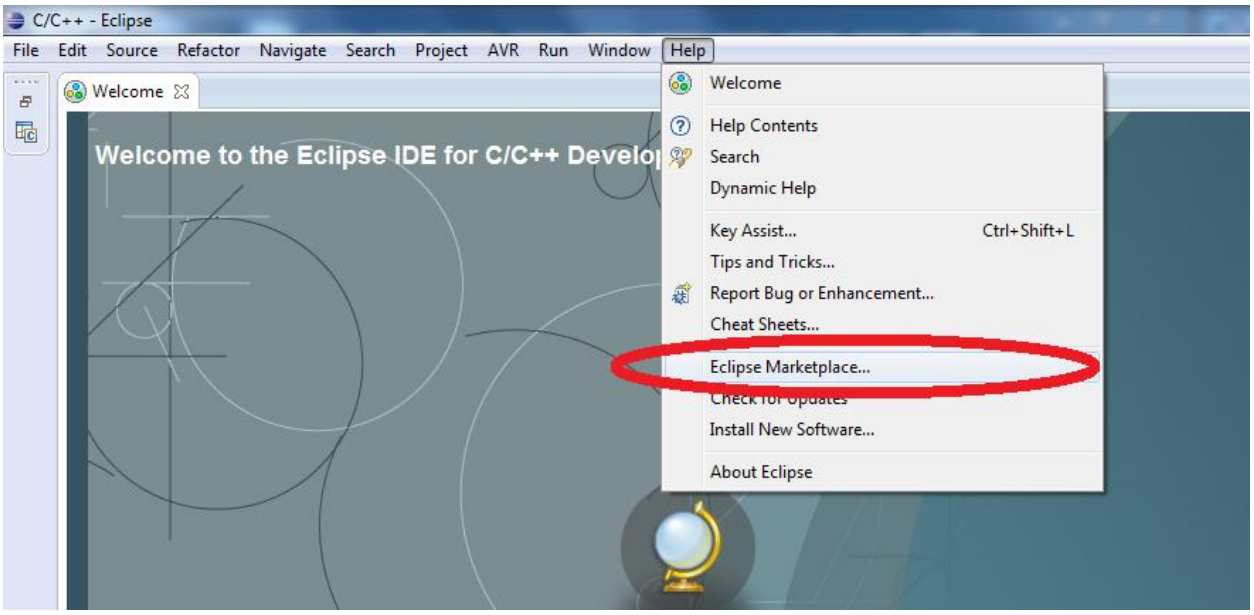
2. In the Work with field, input http://avr-eclipse.sourceforge.net/updatesite and click add.



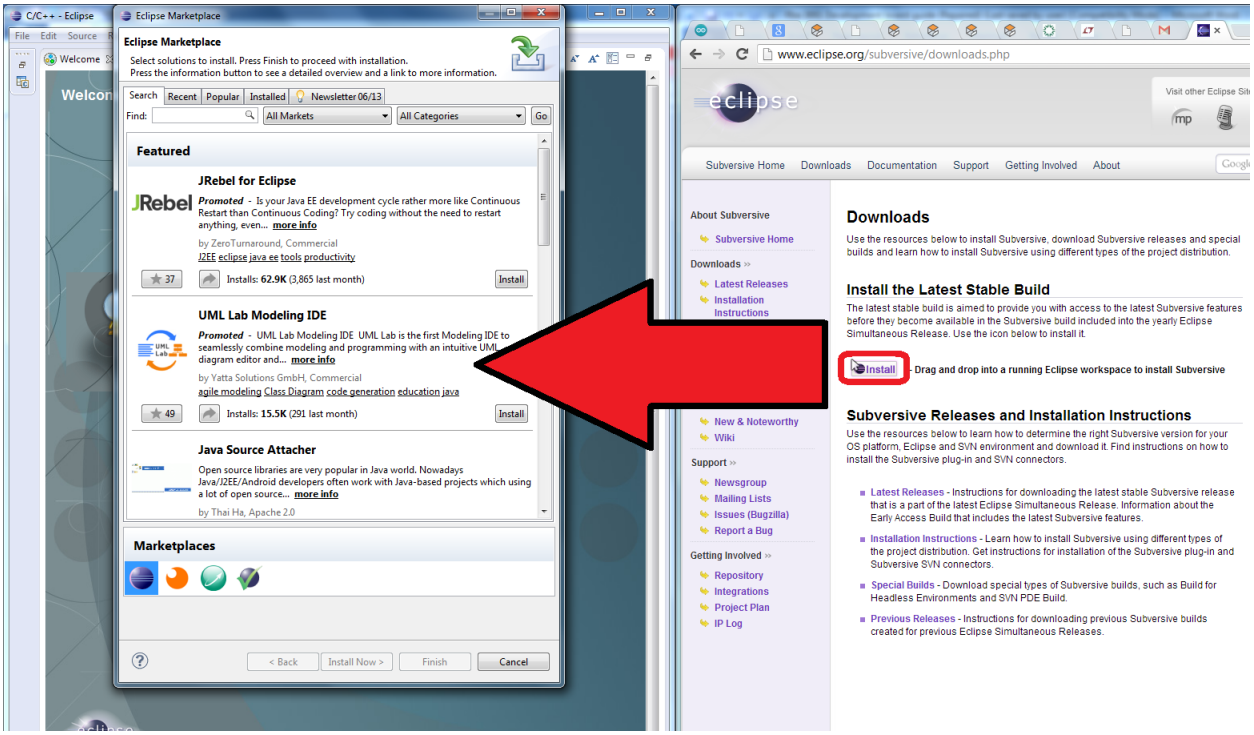
3. In the next prompt in the name field put AVR and click OK.
4. Make sure AVR Eclipse Plugin is selected and click Next.



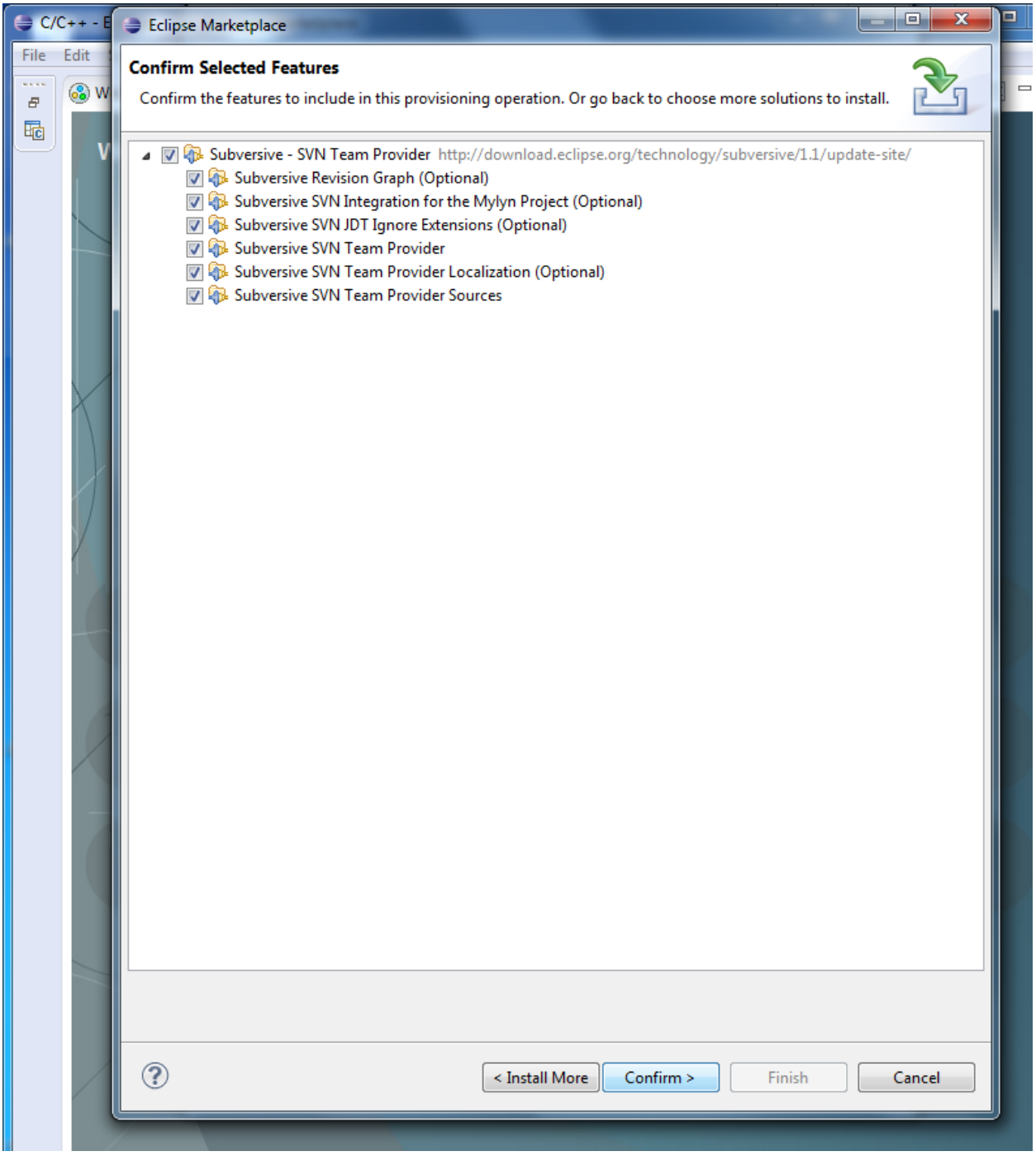
5. Click Next again on the Install Details screen and then accept the terms of the license agreement and click Finish to start the installation.
6. When a Security Warning comes up saying the software contains unsigned content, click OK to continue with the installation.
7. When asked to restart Eclipse, click Yes.
8. When Eclipse starts back up, click on Help > Eclipse Marketplace



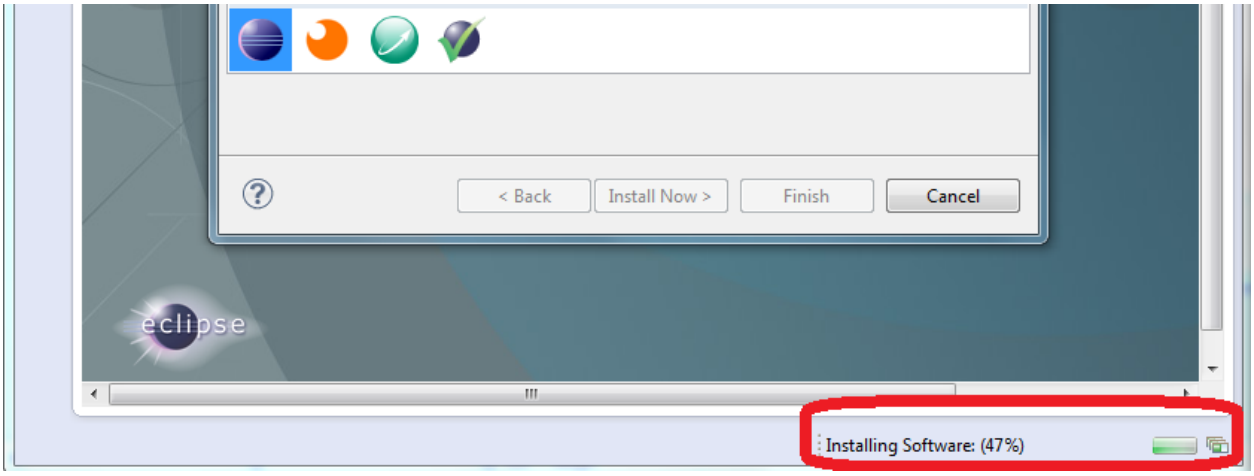
9. Go to http://www.eclipse.org/subversive/downloads.php and drag and drop the Install Icon into the Marketplace Window.



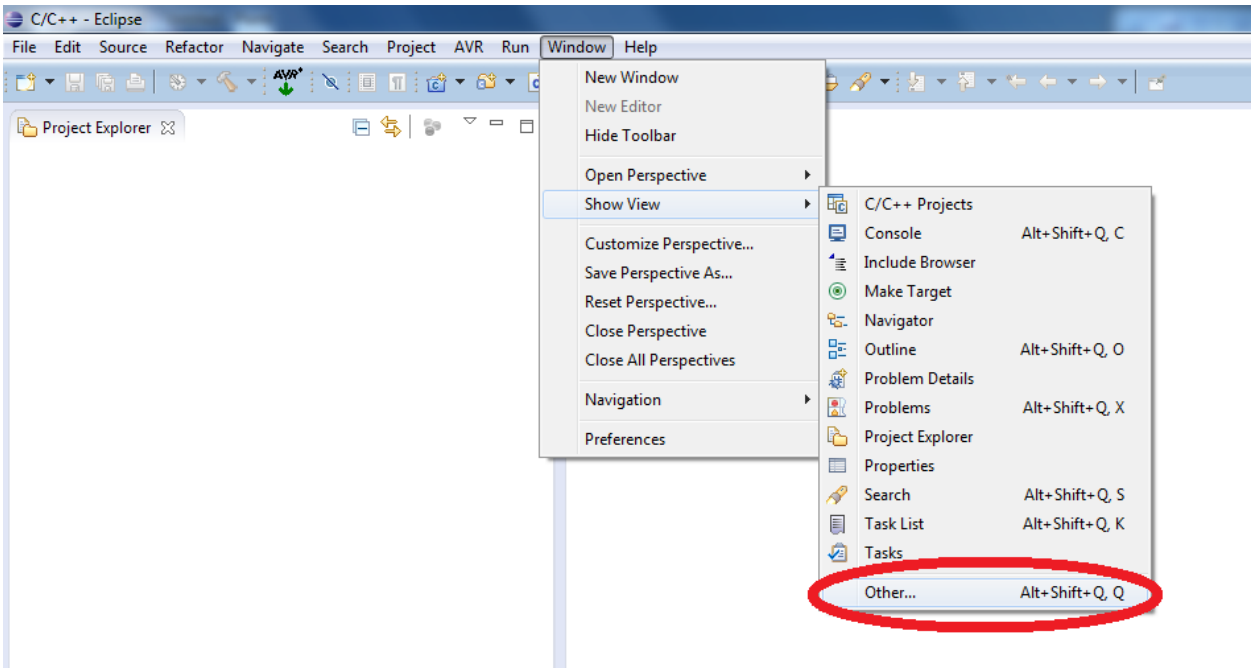
10. On the confirm Selected Features window, make sure everything is checked and press Confirm.



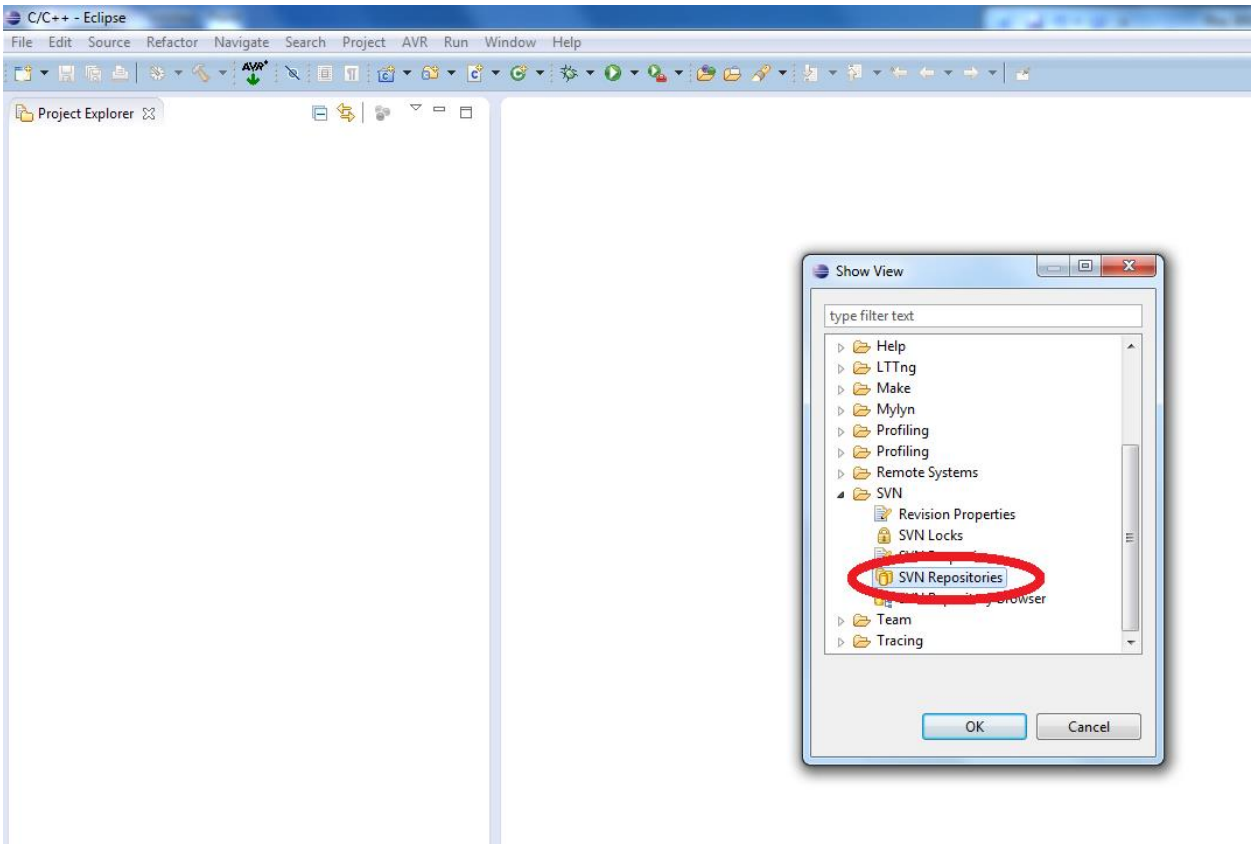
11. On the next screen accept the license agreement and click Finish.
12. Wait until asked to restart Eclipse and click Yes, in the meantime the install progress can be seen in the bottom of the main Eclipse window.



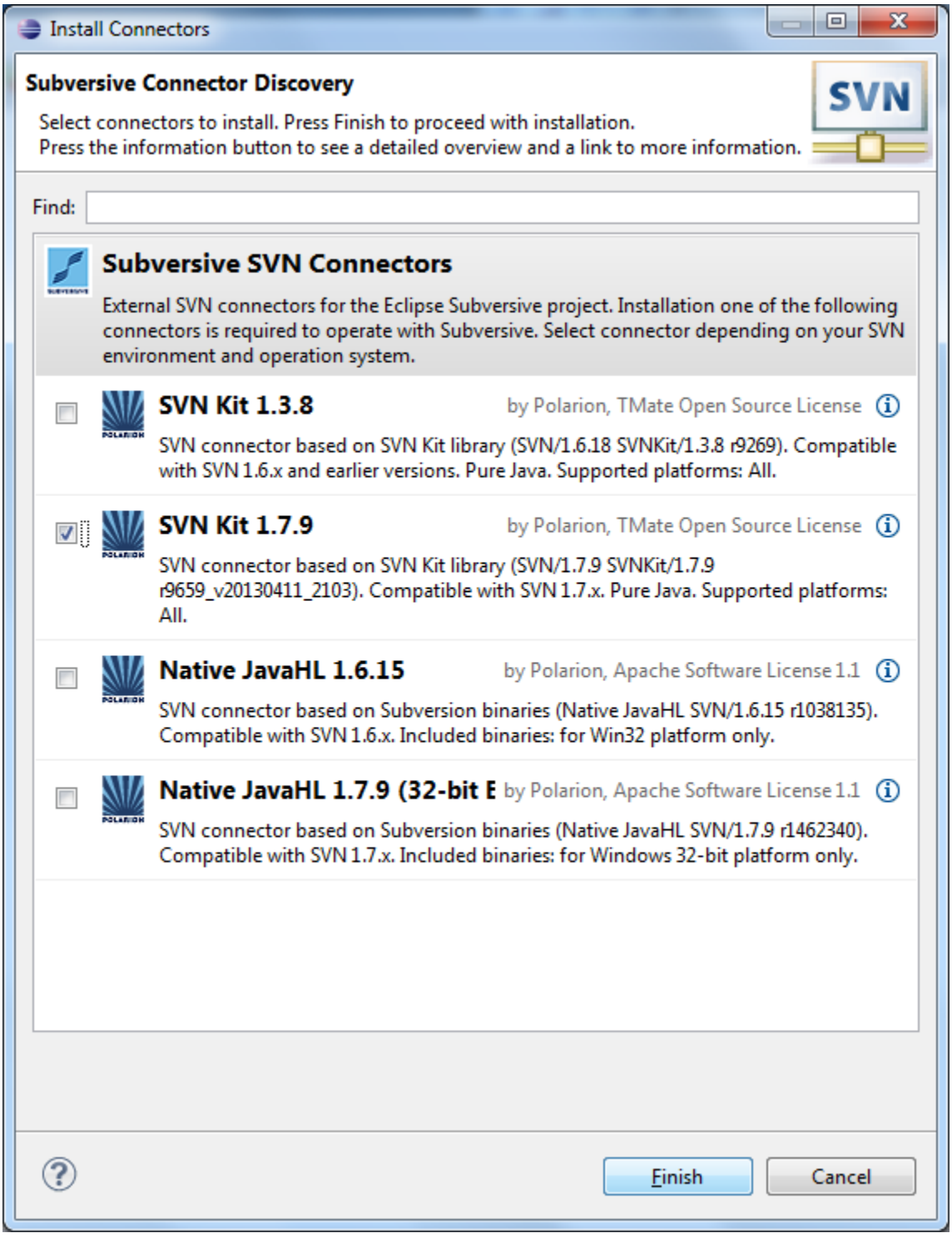
13. Once restarted, go to Window > Show Perspective > Other



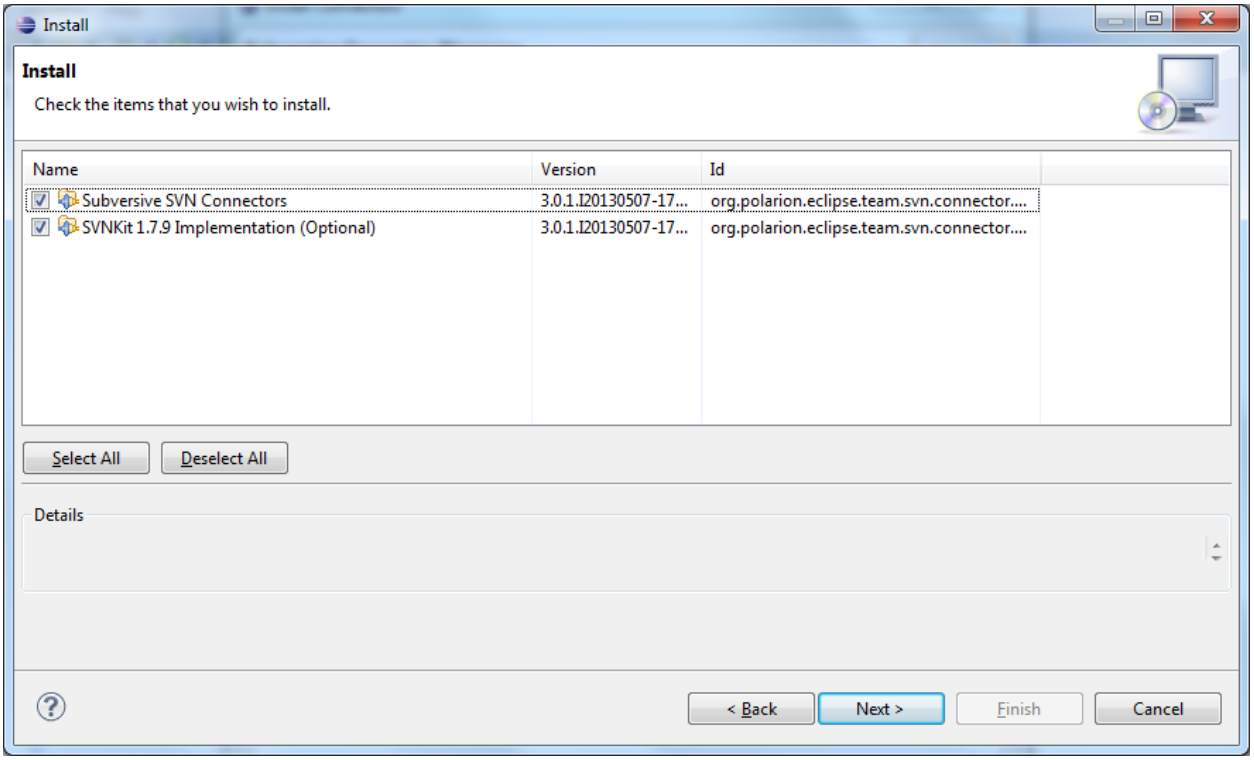
14. Go to the SVN folder and click the arrow to expand and choose SVN Repositories and click OK.



15. When asked to choose a connector, choose the latest version of SVN Kit (do **not** choose the JavaHL version) and select Finish.



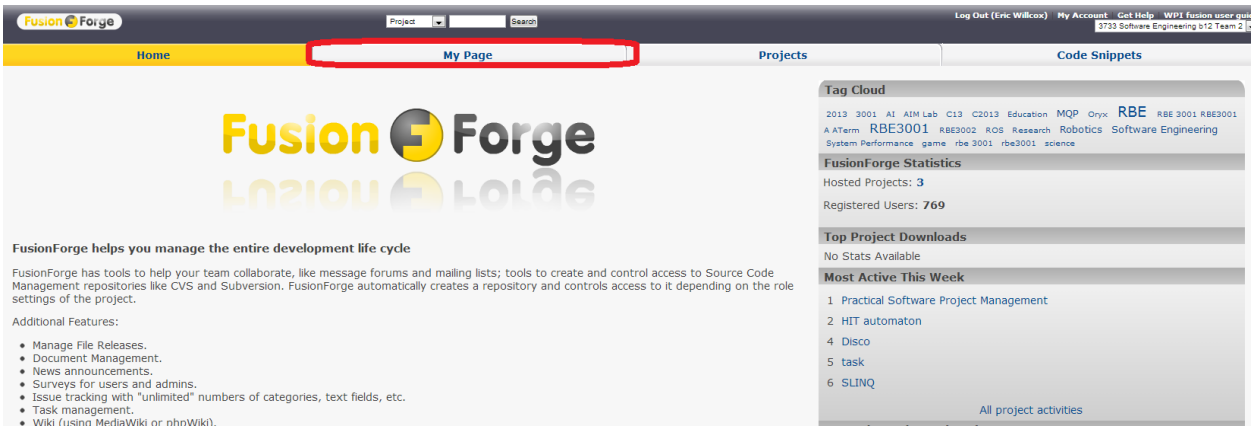
16. When asked what to install, make sure everything is selected and choose Next.



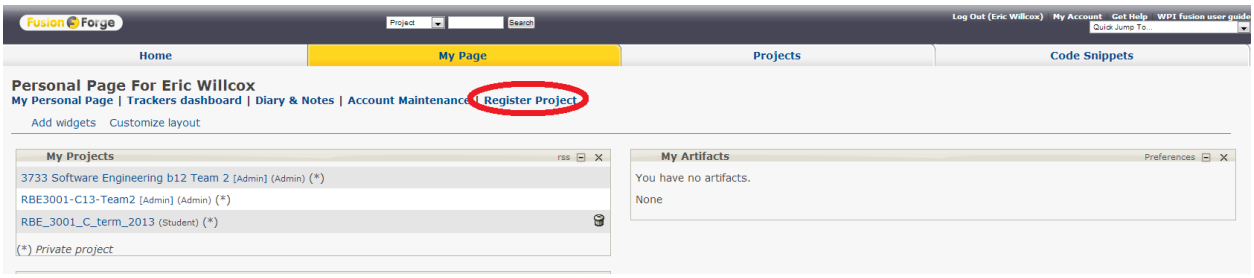
17. On the Install Details window, click Next again then accept the license agreement and click finish.
18. When a Security Warning comes up saying the software contains unsigned content, click OK to continue with the installation.
19. When asked, restart Eclipse.

Fusion Forge Setup

1. Each member needs to create an account at https://fusion.wpi.edu
2. One person in the group needs to create the project on Fusion Forge by click the My Page button near the top of the page.

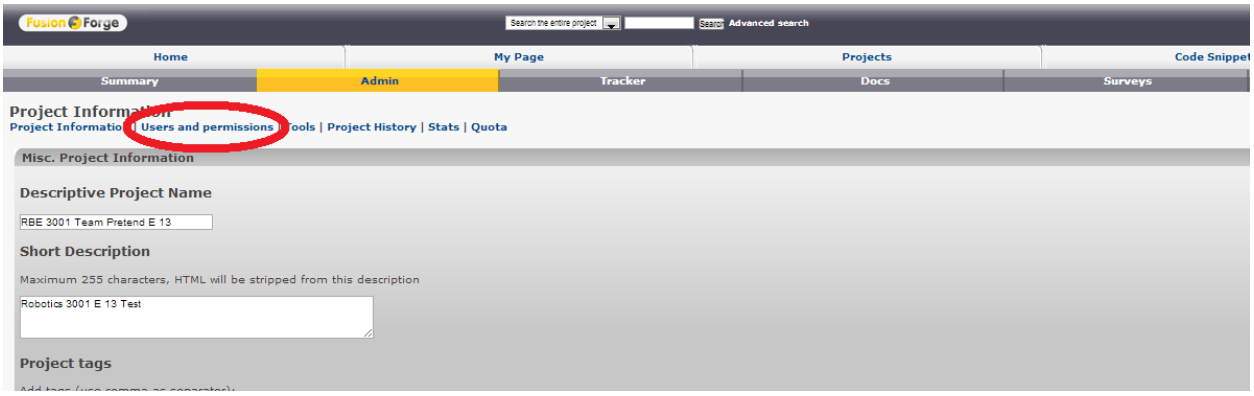


3. Next, click on Register Project

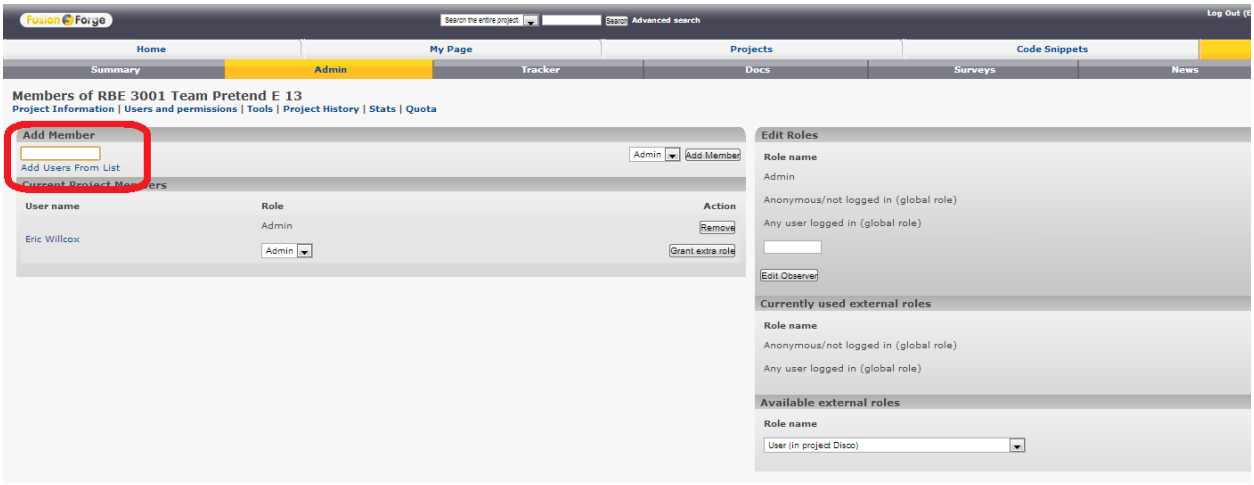


4. Read the descriptions and fill out the form appropriately. Make sure for Source Code you choose Subversion and for Project Template you choose WPI Subversion Template Project.

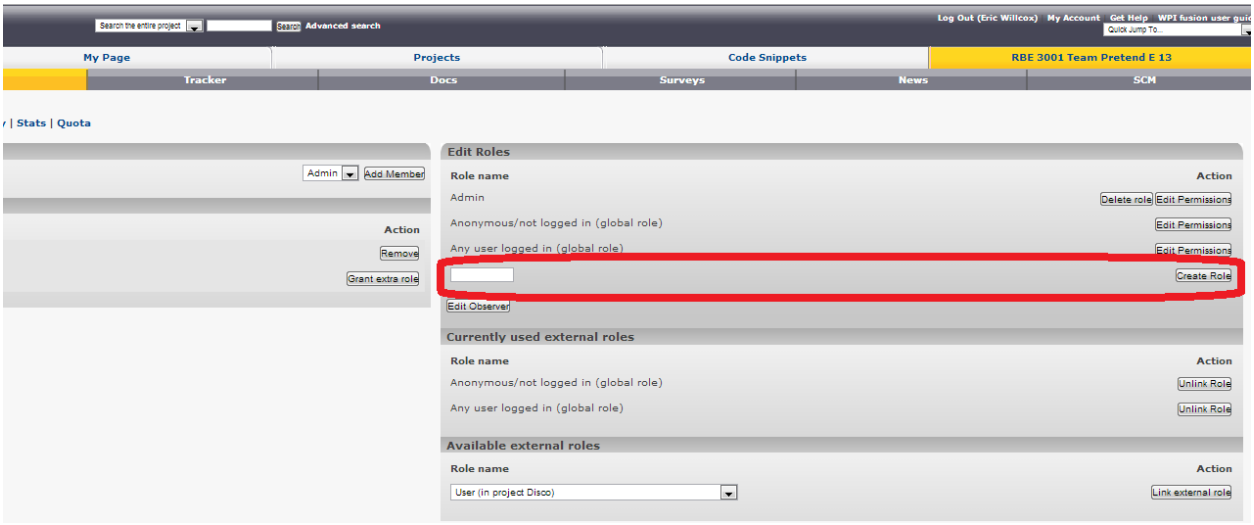
5. Click submit and wait up to 72 hours for an Admin to approve your project.
6. Once approved, go back to the My Page section of Fusion and select your project.
7. To add members to your group, click on the Admin page on the top.
8. Click on Users and Permissions.



9. Under Add Member, type in the Usernames of those you wish to add and press Add Member.
   OR
   Select Add Users From List to see all members registered on Fusion and select who you want to add and press Finish.

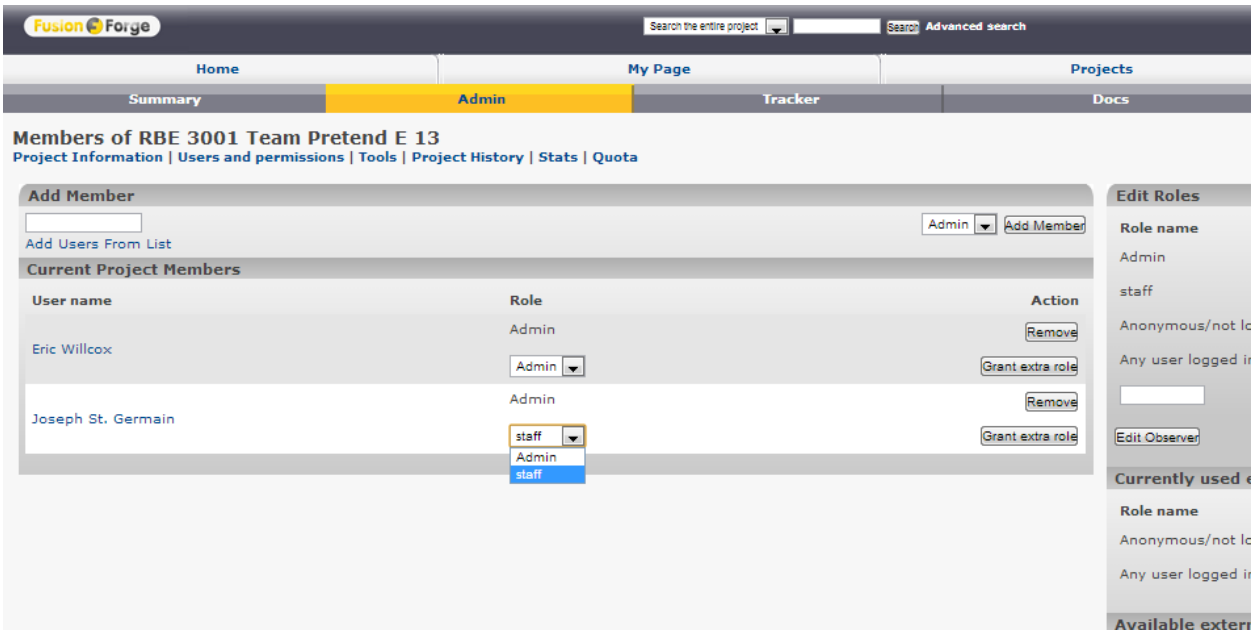


10. You can create a different Role besides Admin to prevent users from editing your files if you want to only give them permission to view you file. To do so, under Edit Roles, type a name into the empty field and press Create Role.

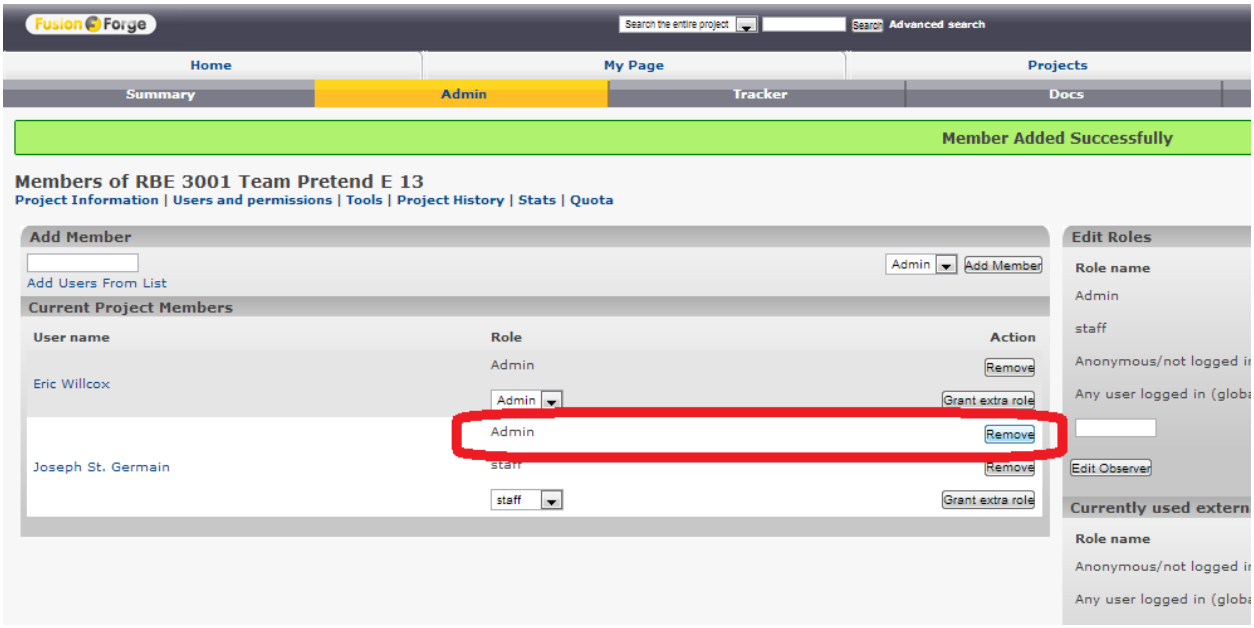


11. On the next screen, select what permissions the new role has. To give allow them to only view your project without editing, make sure the fields match the following then press Submit.

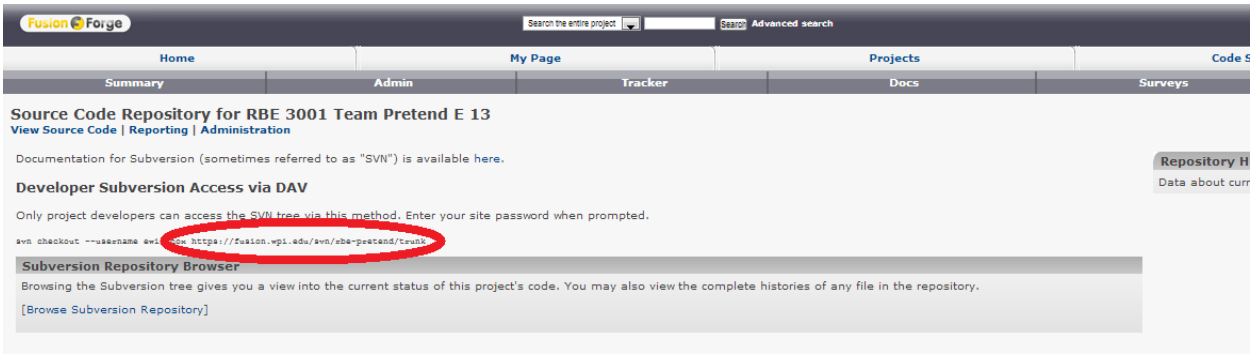
12. Go back to Users and Permissions to assign the new role to a user.
13. Click on the drop-down arrow for the user whose role you want to change and select the newly created role, then click Grant extra role.



14. Once added, take away the old Admin role if you created them with it by click Remove on the Admin line.



15. If you ever need to check the location of the location of the SVN trunk for your team, go to the SCM team and look at the URL here (in the form `https://fusion.wpi.edu/svn/***/trunk` where *** is your UNIX name).

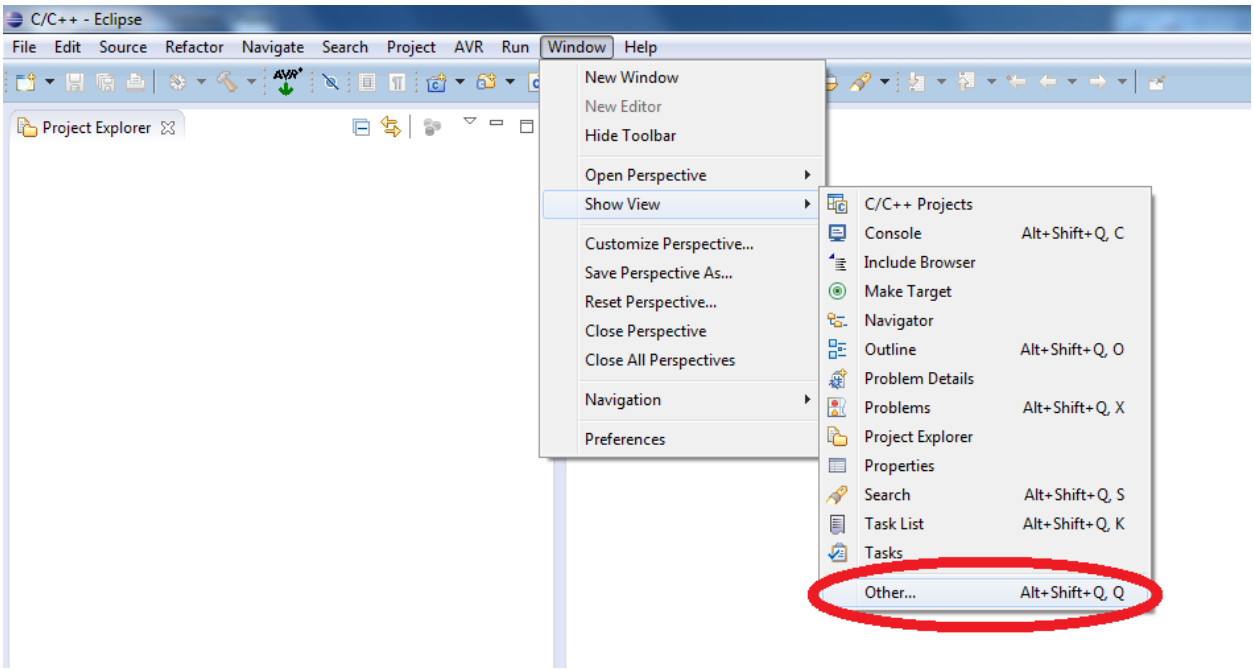

FusionForge
Search the entire project
Search
Advanced search
Home
My Page
Projects
Summary
Admin
Tracker
Docs
Surveys
Source Code Repository for RBE 3001 Team Pretend E 13
View Source Code | Reporting | Administration
Documentation for Subversion (sometimes referred to as "SVN") is available here.
Developer Subversion Access via DAV
Only project developers can access the SVN tree via this method. Enter your site password when prompted.
svn checkout --username
https://fusion.wpi.edu/svn/rbe-pretend/trunk
Subversion Repository Browser
Browsing the Subversion tree gives you a view into the current status of this project's code. You may also view the complete histories of any file in the repository.
[Browse Subversion Repository]
Repository H
Data about curr

Joining RBELib

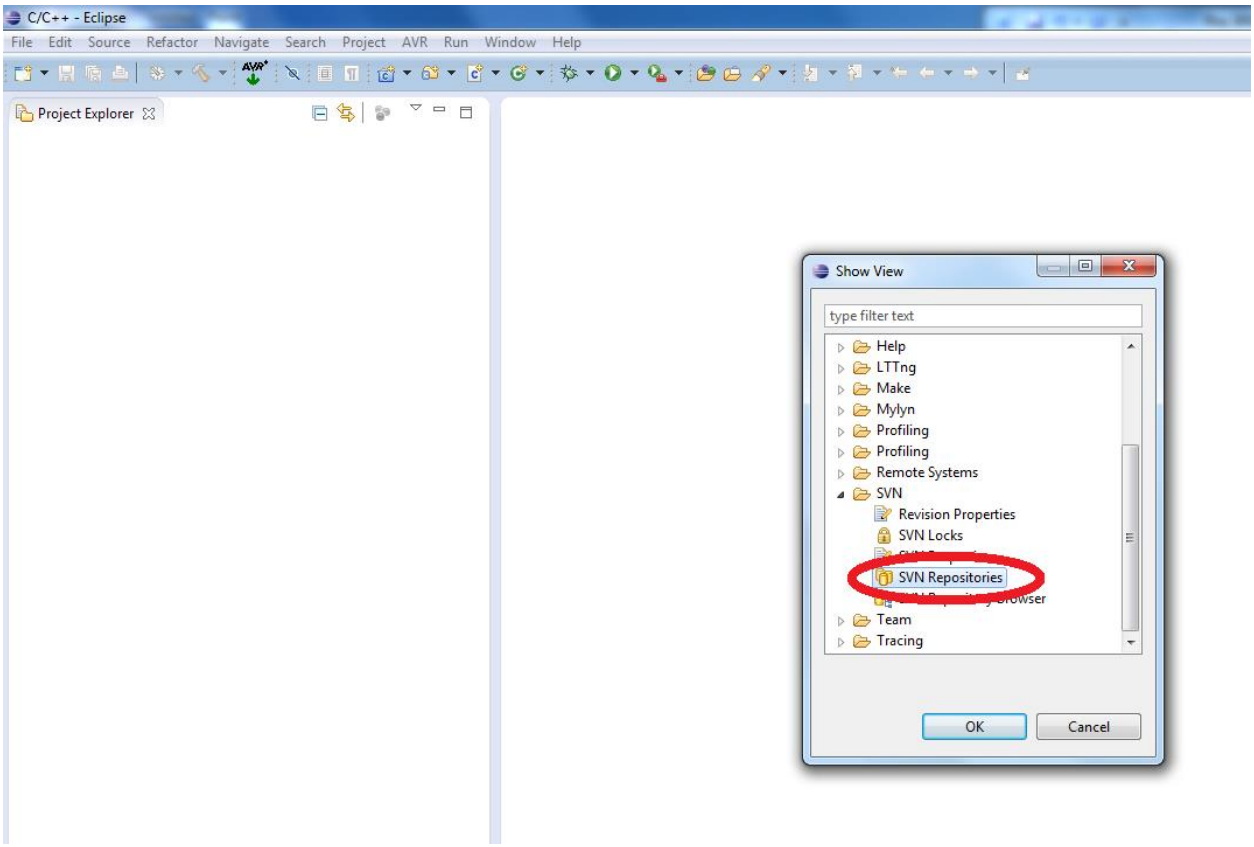
1. To join RBELib, you need to request permission on Fusion Forge to the project. Open up fusion.wpi.edu in your web browser and log in to your account.
2. Go to Joe St. Germain's profile and find your class's project and request permission to join it. Joe's profile can be found at: fusion.wpi.edu/users/joest/. The name of the project will be given out on Blackboard.

Setting up SVN in Eclipse

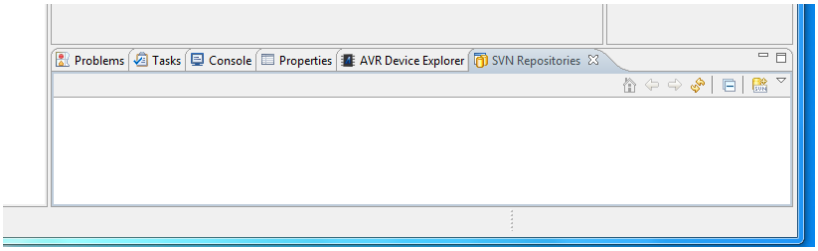
1. Open Eclipse
2. Select your Workspace
3. In eclipse select Window > show view > other



4. Then select SVN > SVN Repositories > click OK

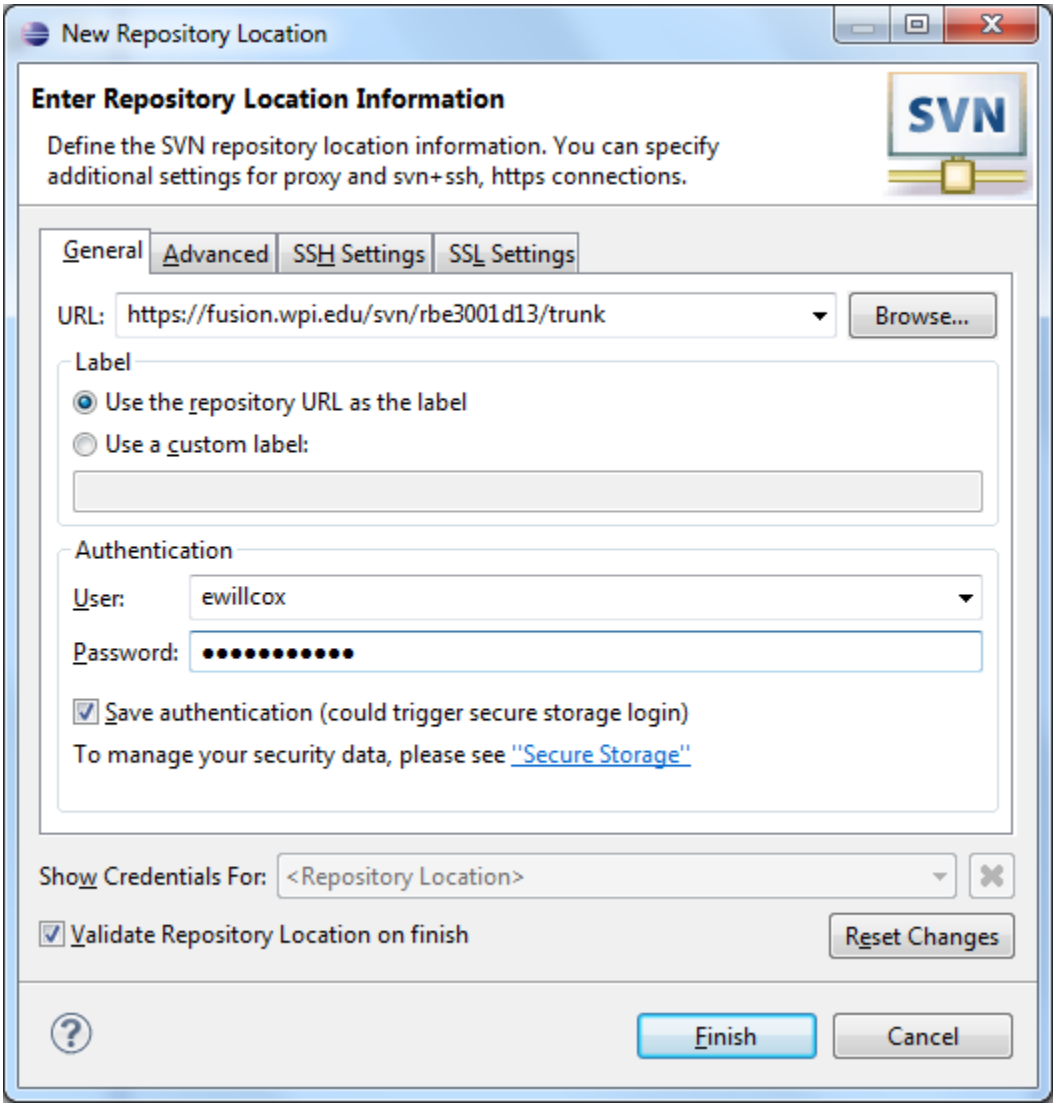


5. Find the SVN Tab at the bottom of the screen (sometimes opens on the right hand side of the screen) and open it.

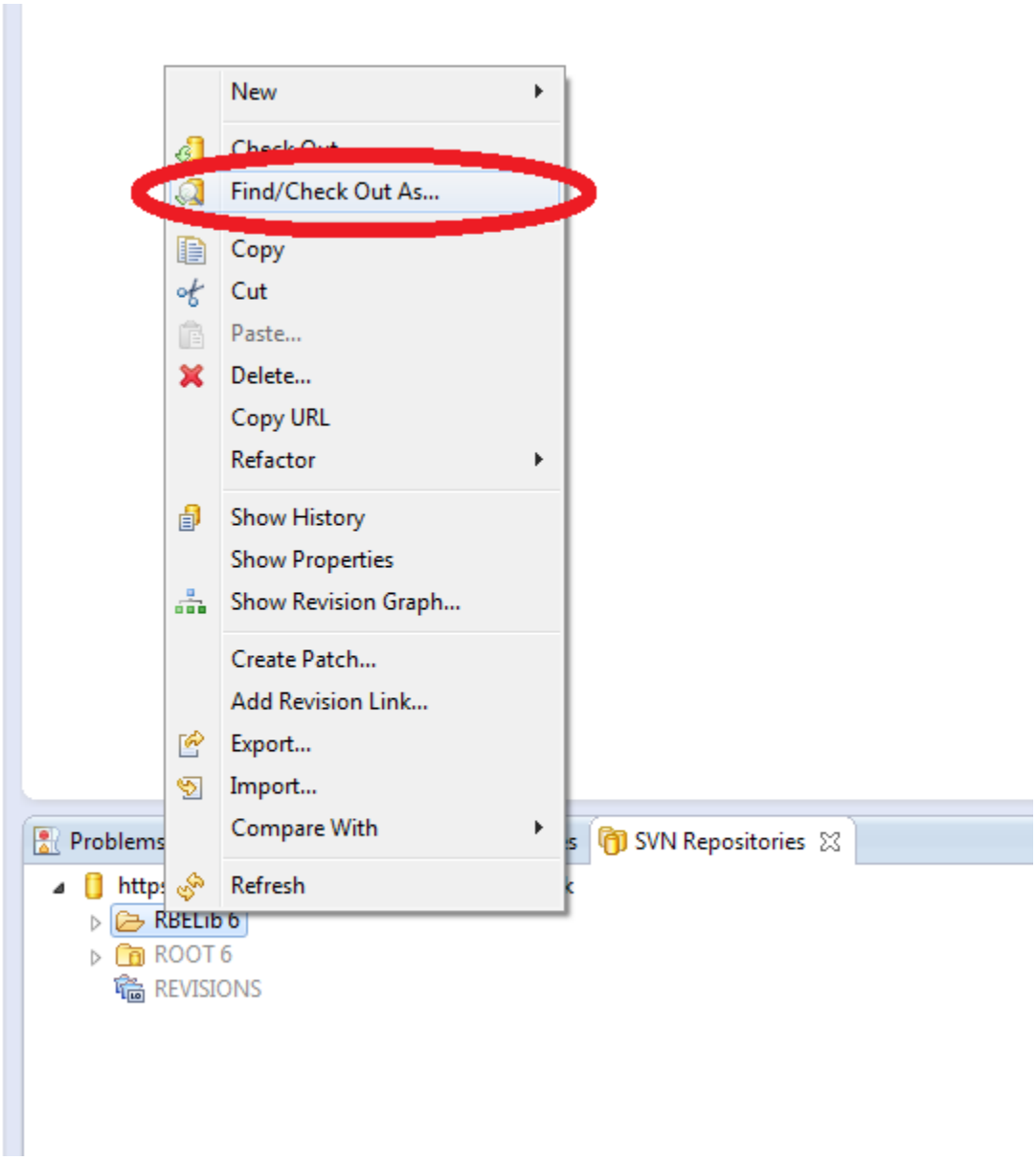


6. Right Click in the White Space (within the SVN Tab) and select New > Repository Location

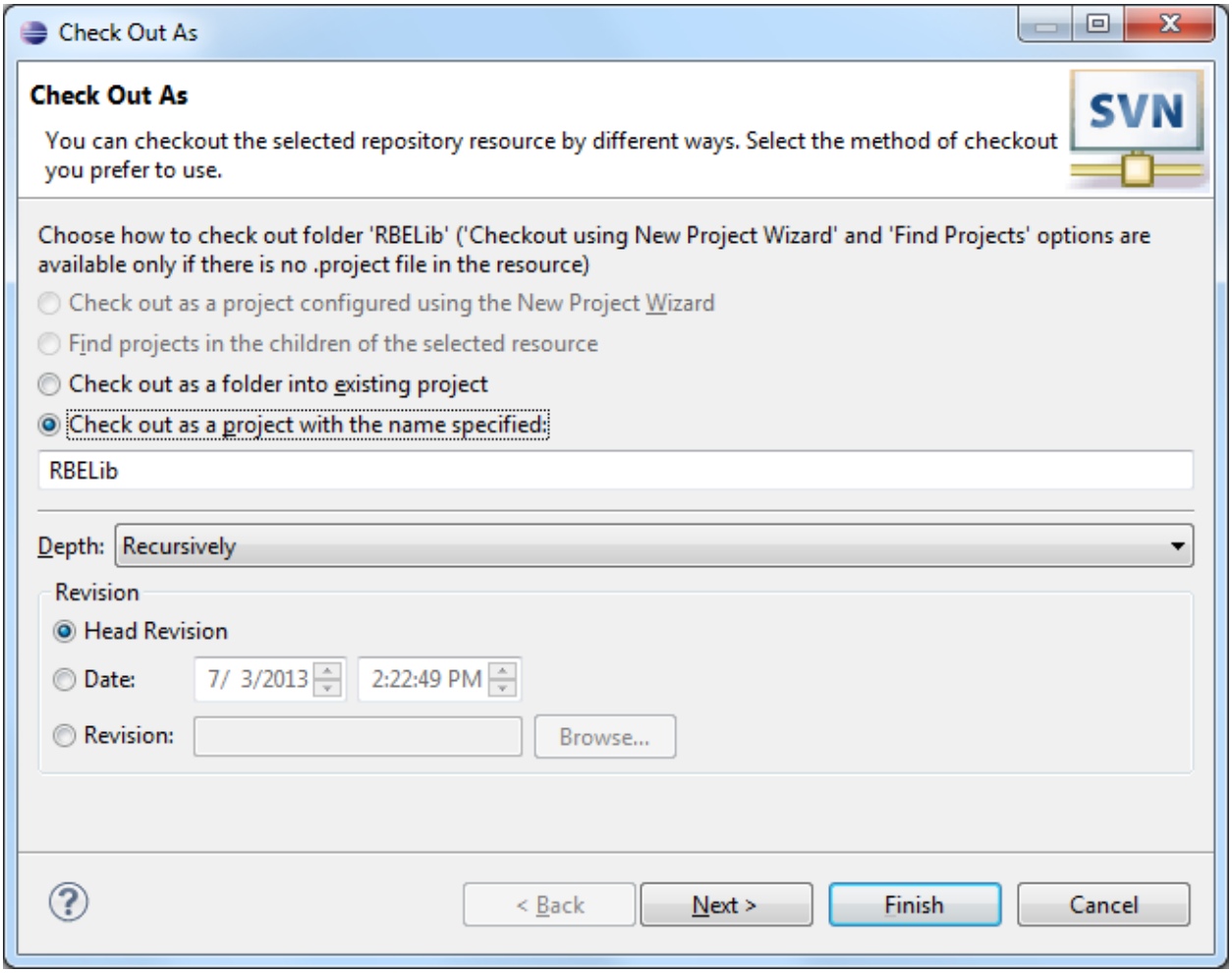
7. Where it asks for the URL, input the information given to you on Blackboard to join RBELib and enter your username and password. Click Finish.



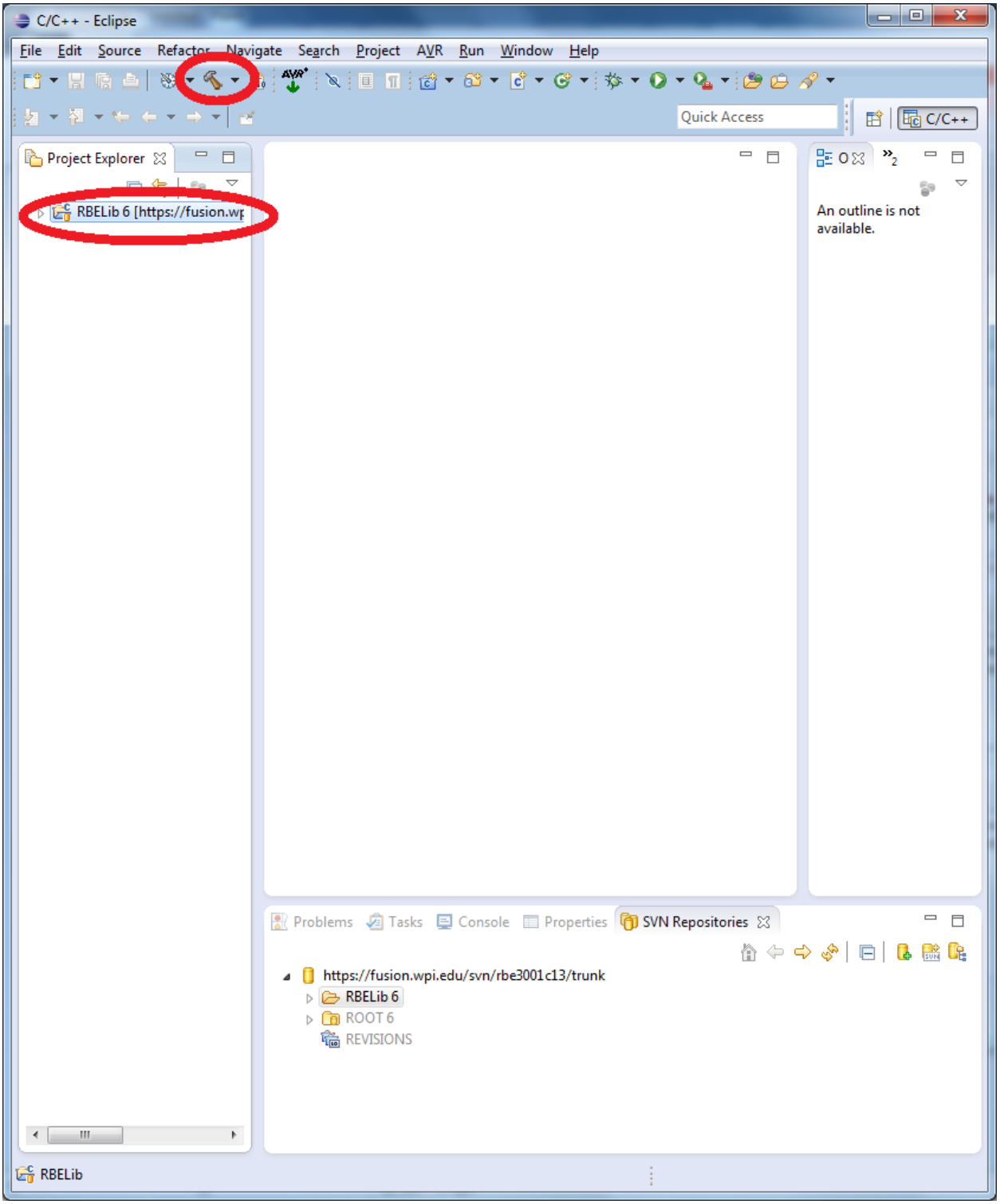
8. You may be asked if you want to normalize the URL by cutting off the last segment, select No.
9. Once done, open up the repository below and right click on RBELib and choose Find/Check Out As



10. Make sure the settings match below and click Finish.



11. Once checked out build the RBELib library by selecting RBELib and pressing the hammer or by Project > Build Project. It will not build unless it is highlighted in the Project Explorer.

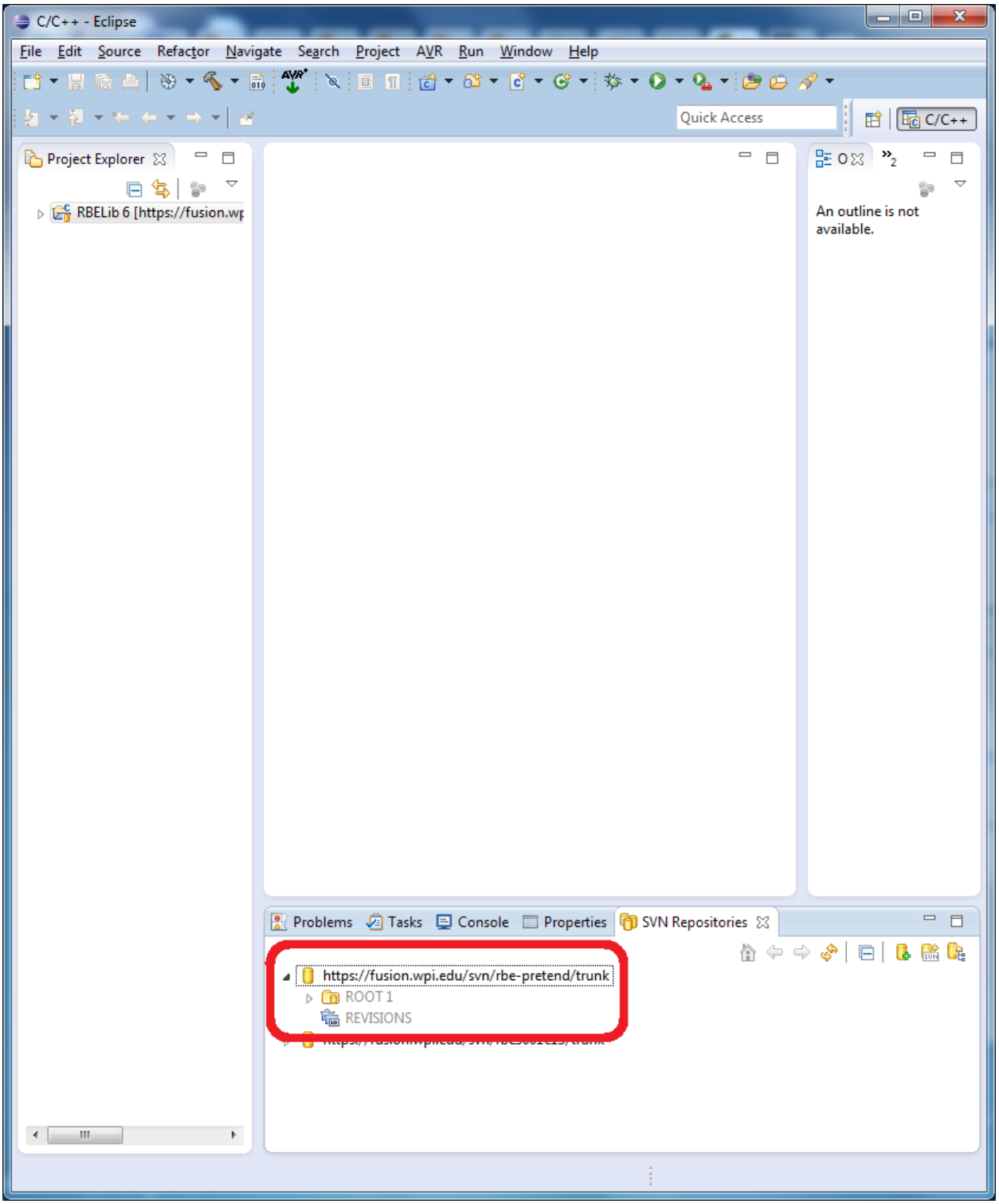


12. Add your Teams SVN location in the same manner as RBELib, you can find your SVN location by following step 15 for setting up Fusion Forge. It should have the form of

https://fusion.wpi.edu/svn/***/Trunk where *** is your UNIX name that you selected during the project setup. This folder will be empty until you begin to add code into it.

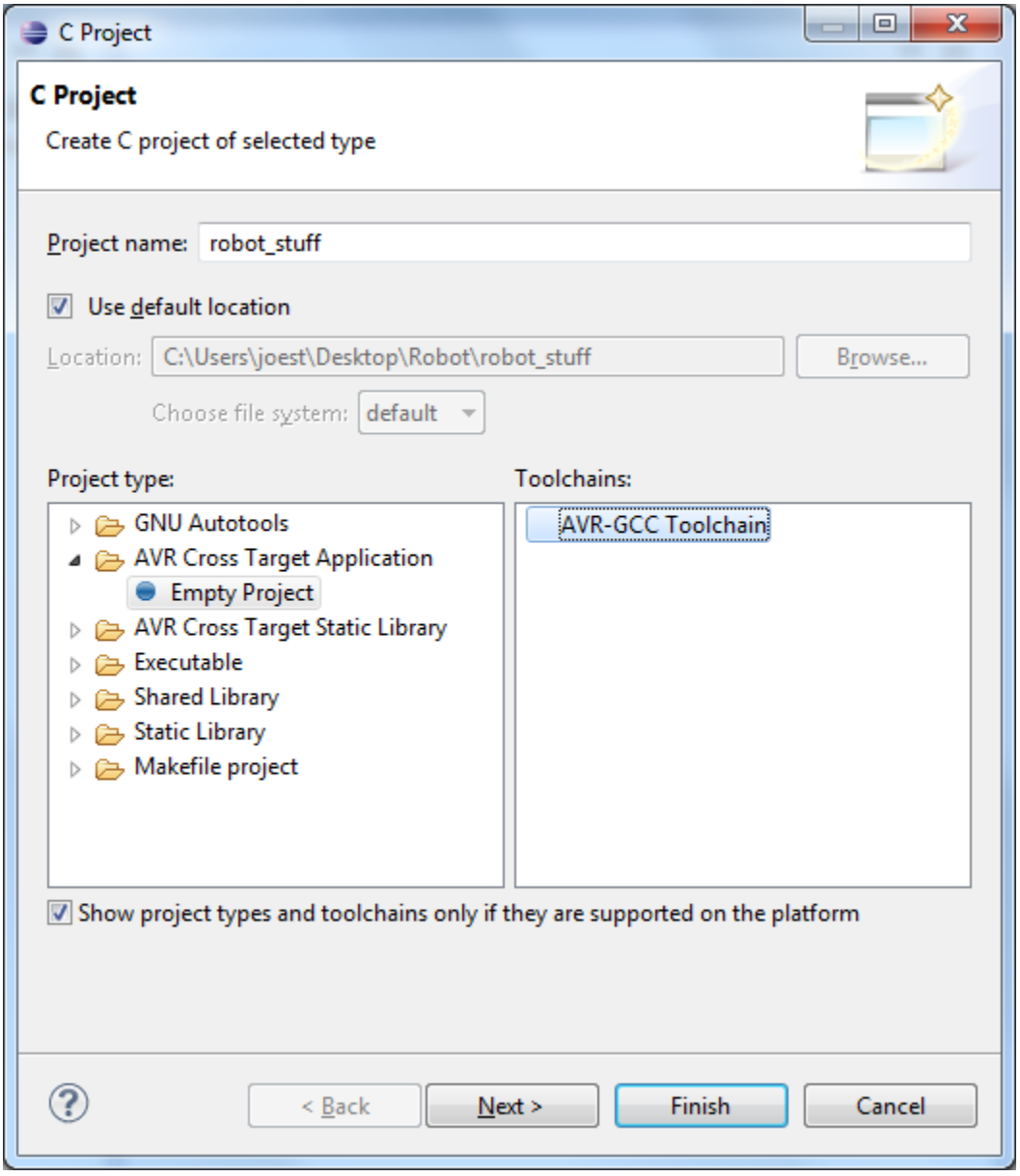
Creating a New Project

1. Make sure you have your SVN project setup properly.

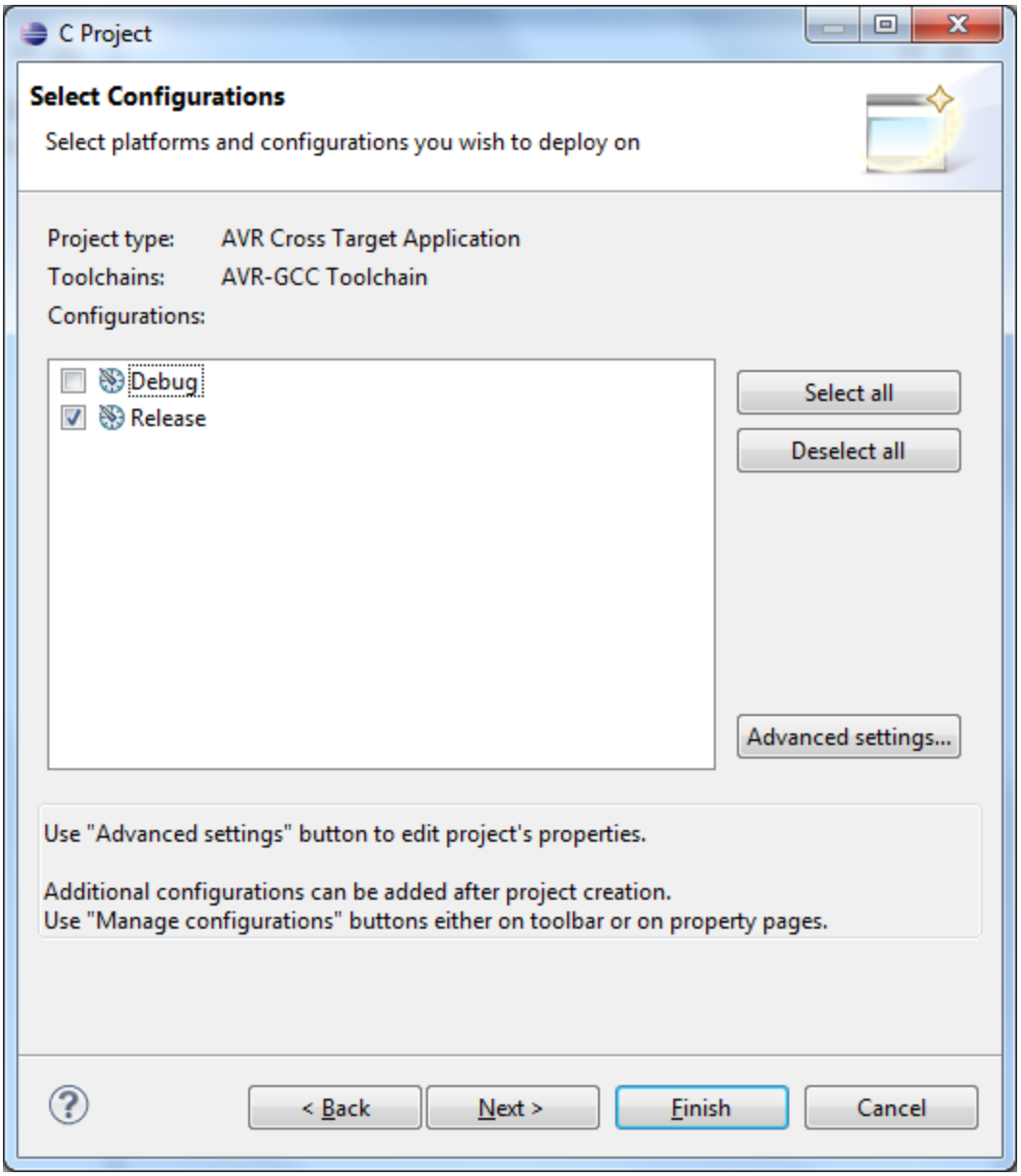


2. Select File > New > C Project

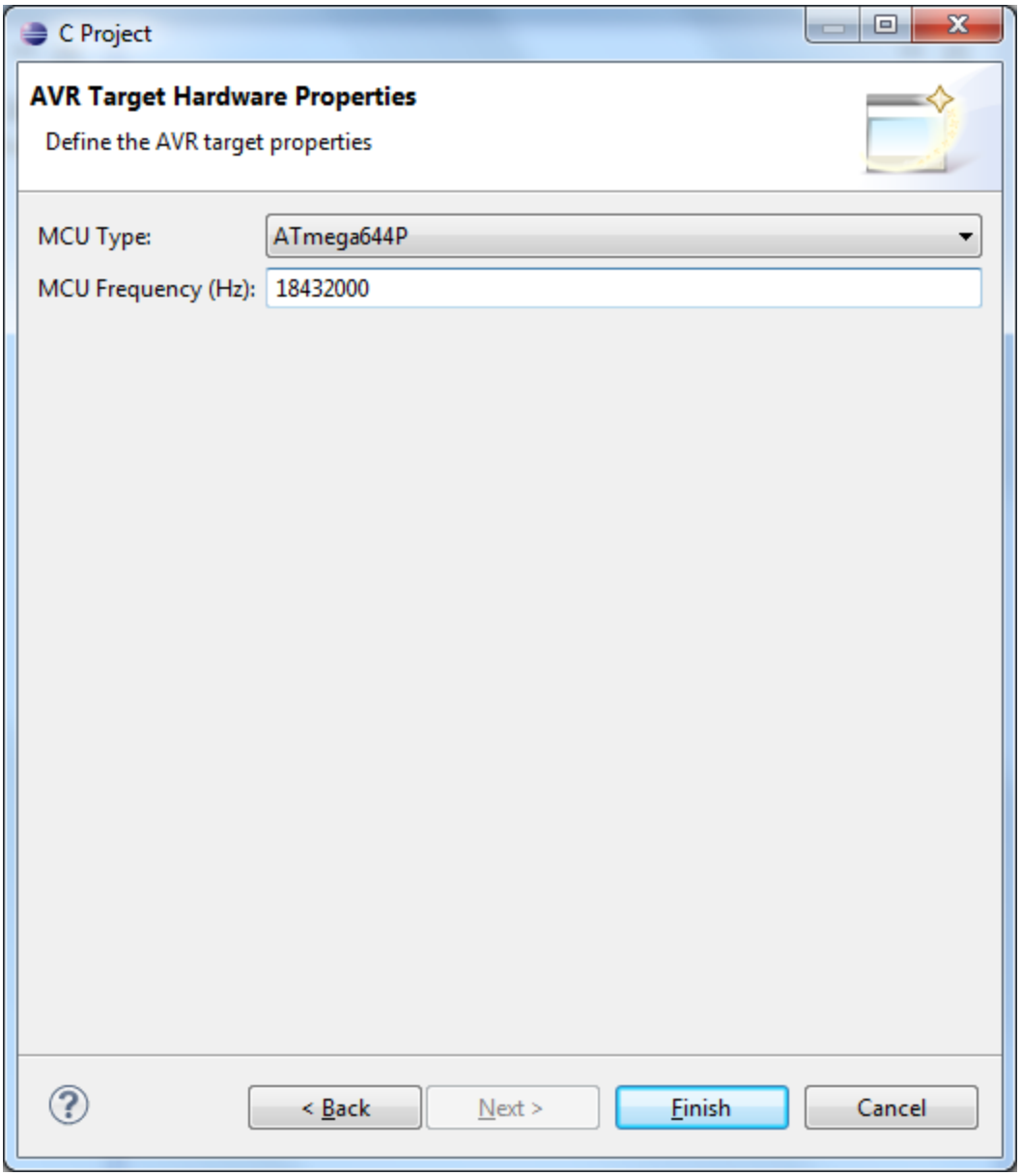
3. Name your Project and Select AVR Cross target Application > empty project > AVR-GCC Tool chain and press Next.



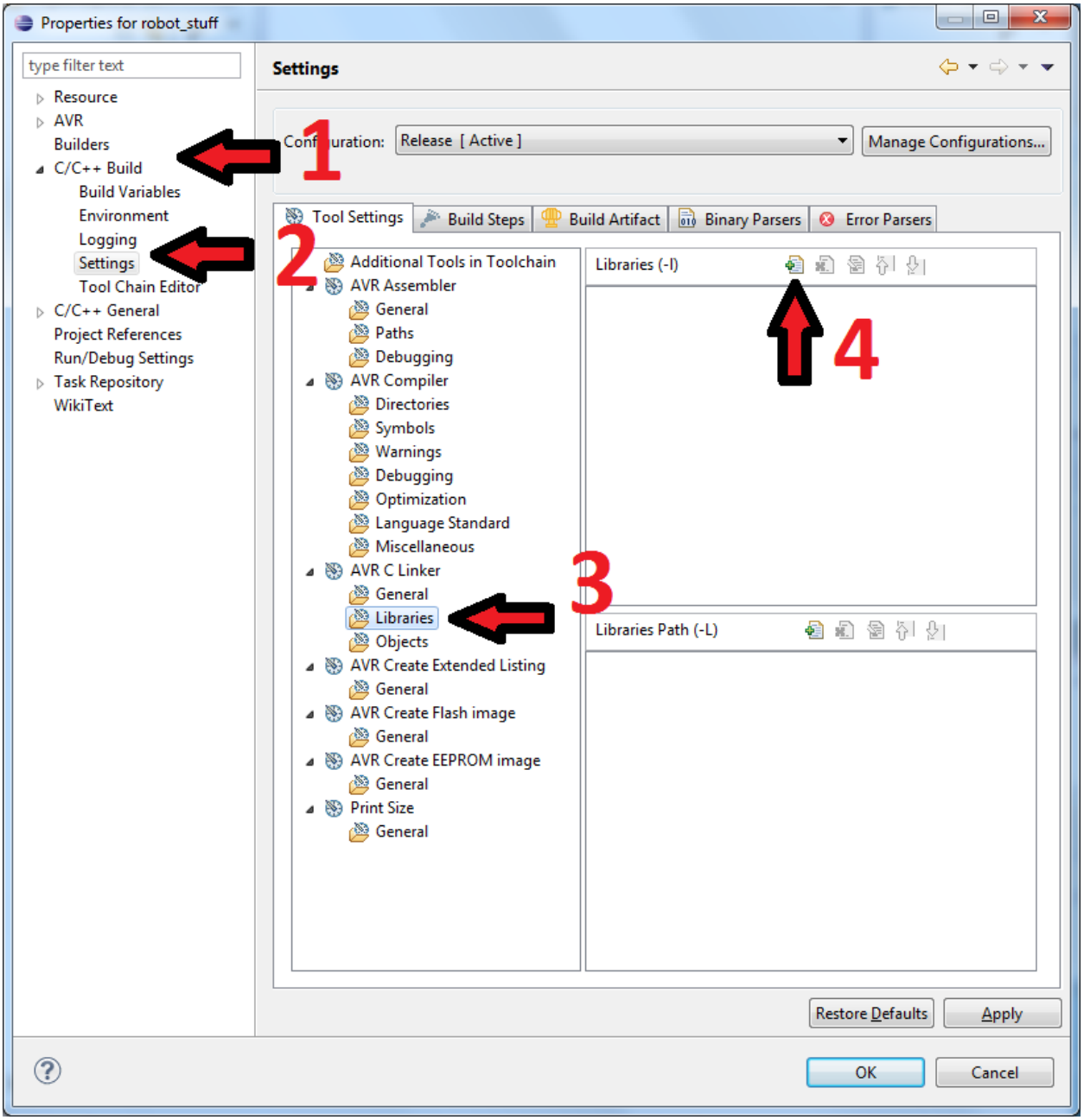
4. Deselect Debug and press next



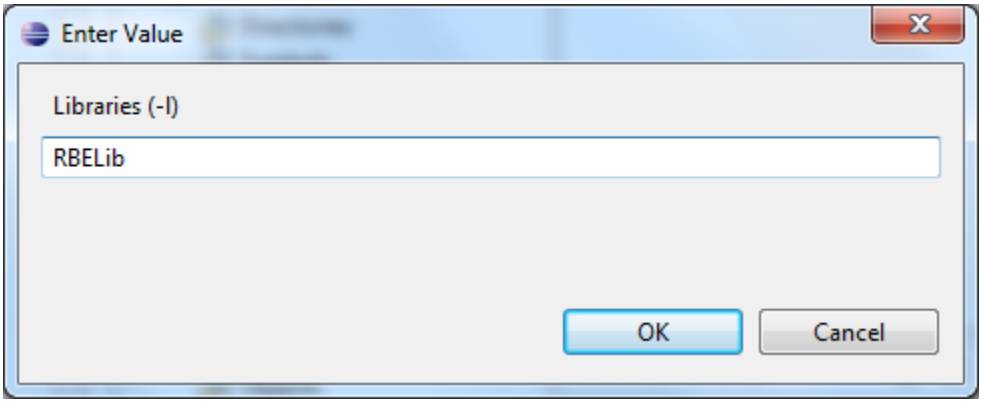
5. Select the Atmega 644P and Enter the MCU Frequency (18,432,000) and press Finish



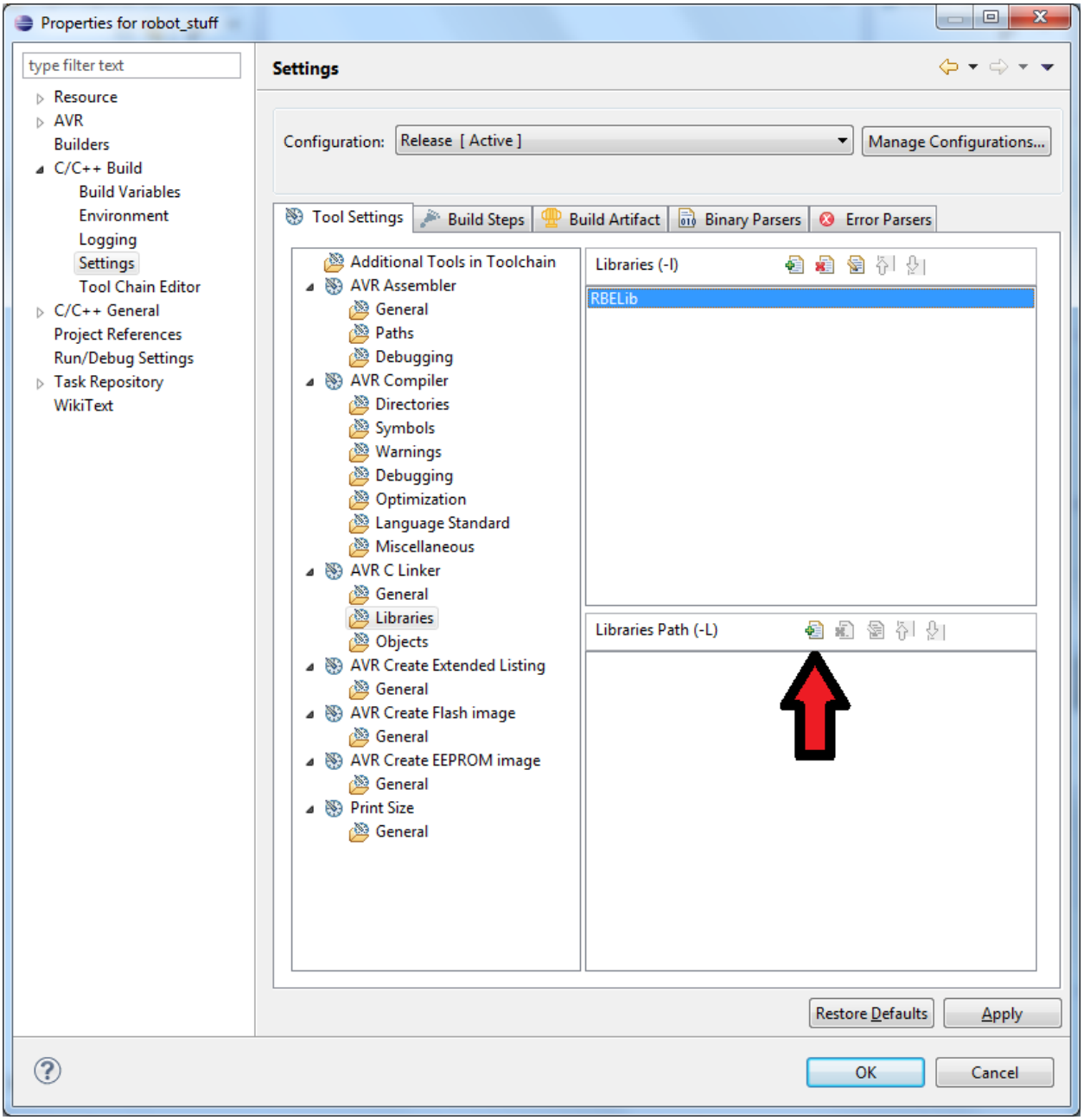
6. Now That we have the project created we need to link the RBELib to the project. Right Click on the Project in the Project Explorer and select properties
7. Expand C/C++ Build > settings
8. Select AVR C Linker > Libraries
9. In the Libraries Section hit the + Button



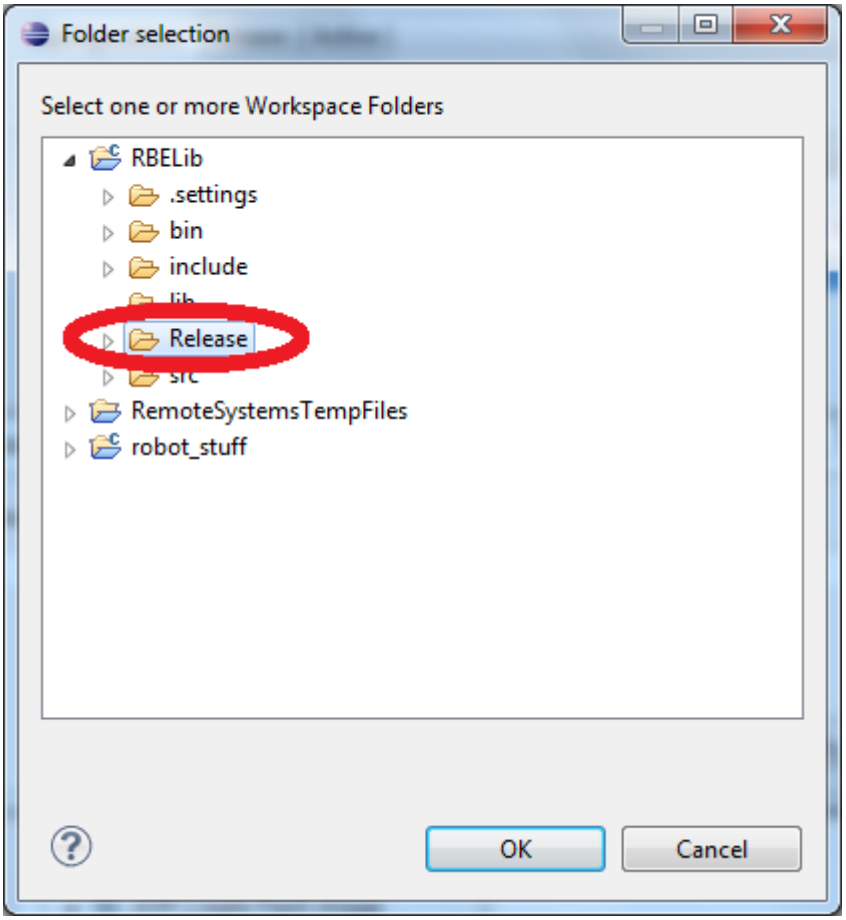
10. Enter ‘RBELib’, then do it again but this time enter ‘m’ which will include the math library library.



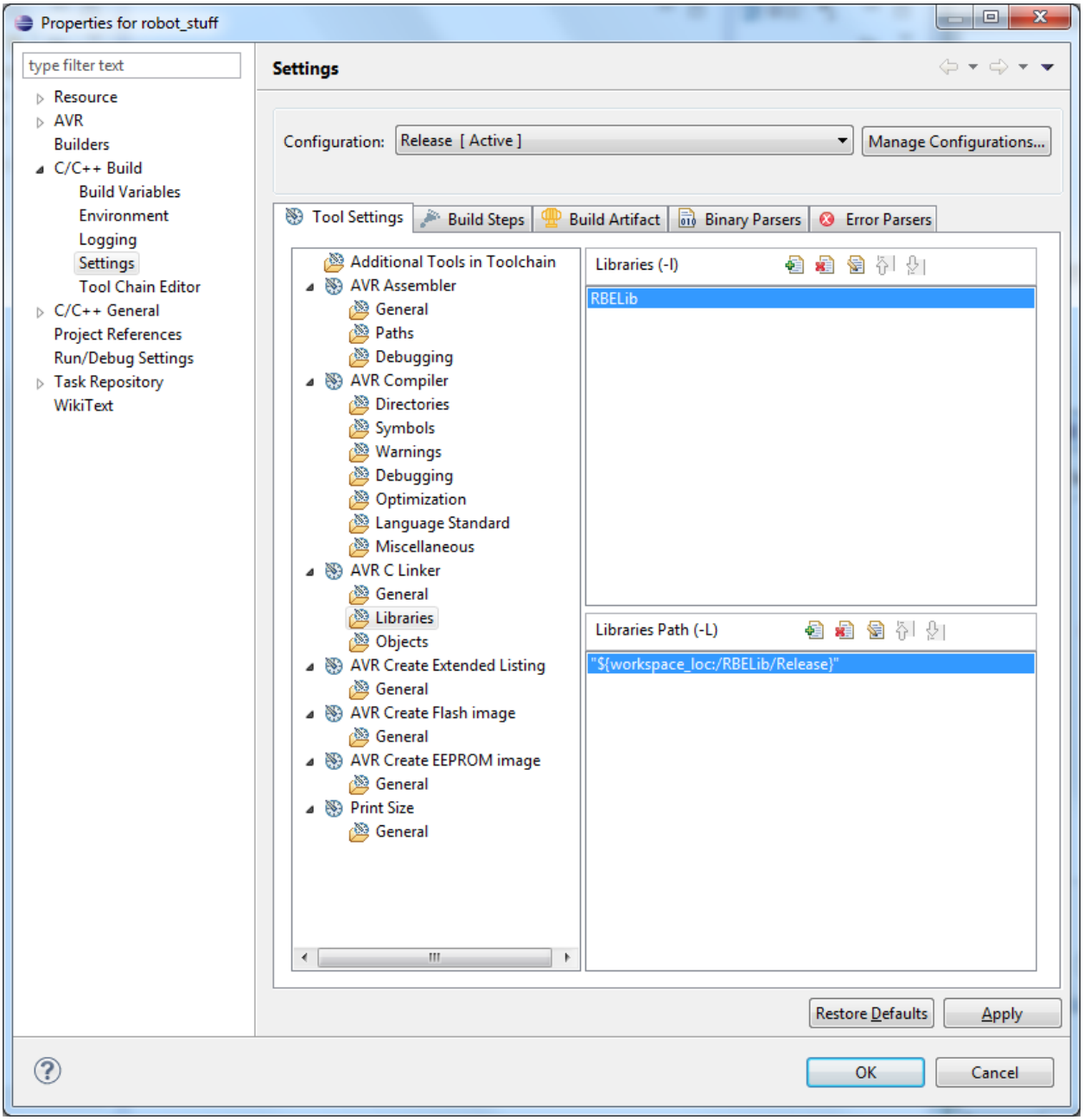
11. In the Library Path Section hit the + Button



12. Click on Workspace, Expand RBELib and choose Release. Select OK on this window and OK again on the next.



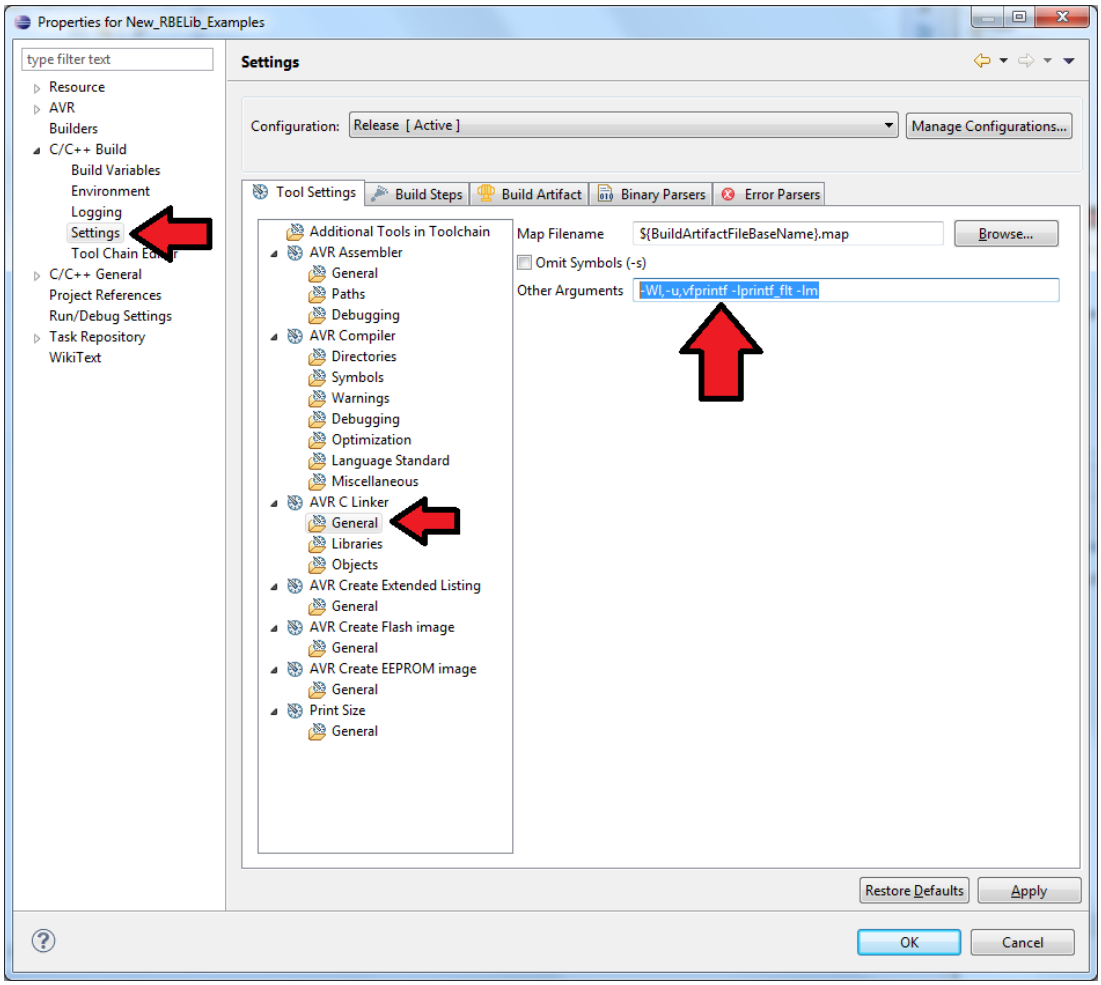
13. Your window should now look like this.



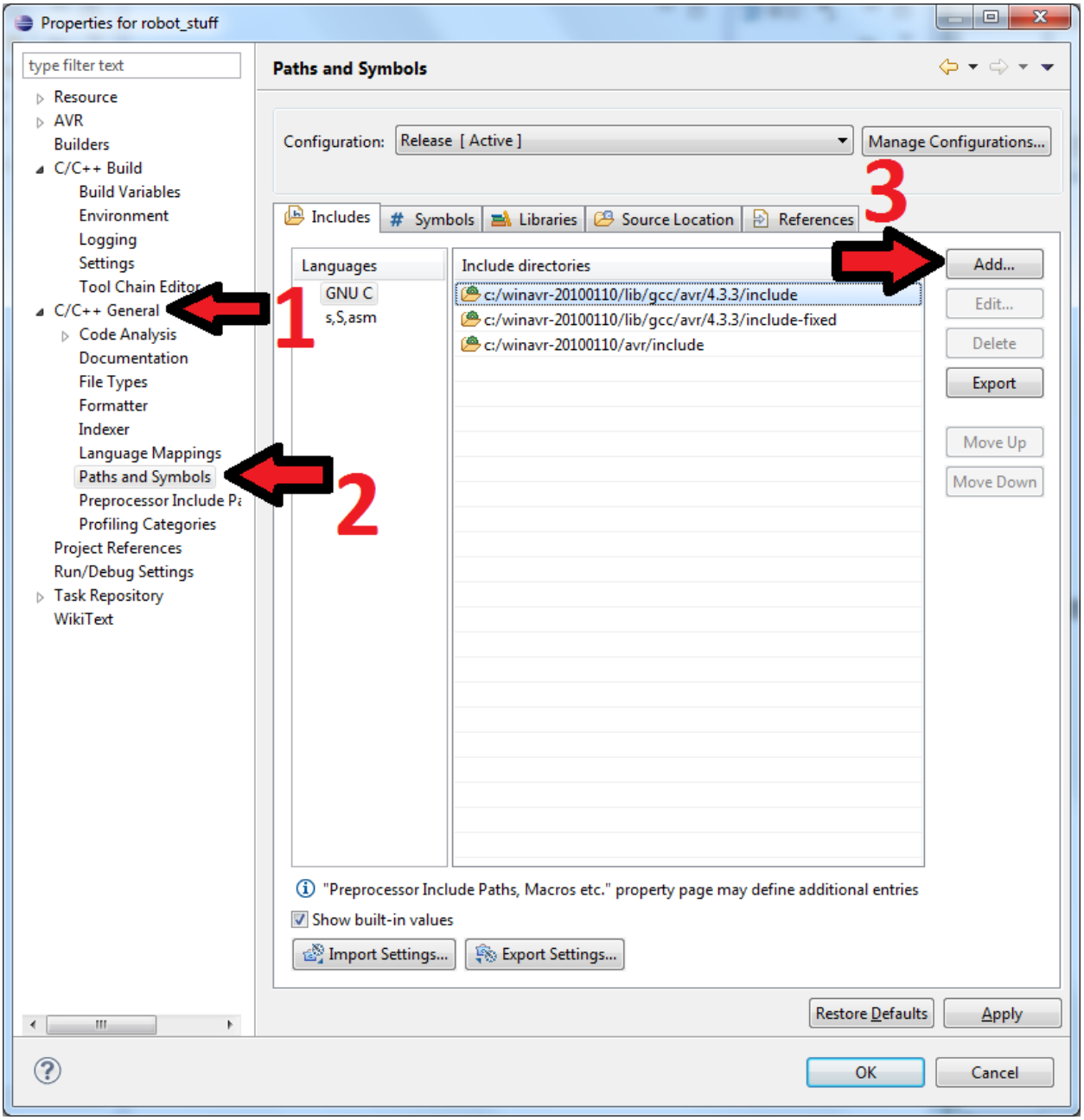
14. Now select the AVR C Linker > General and on the Other Arguments line enter:

    -Wl,-u,vfprintf -lprintf_flt –lm

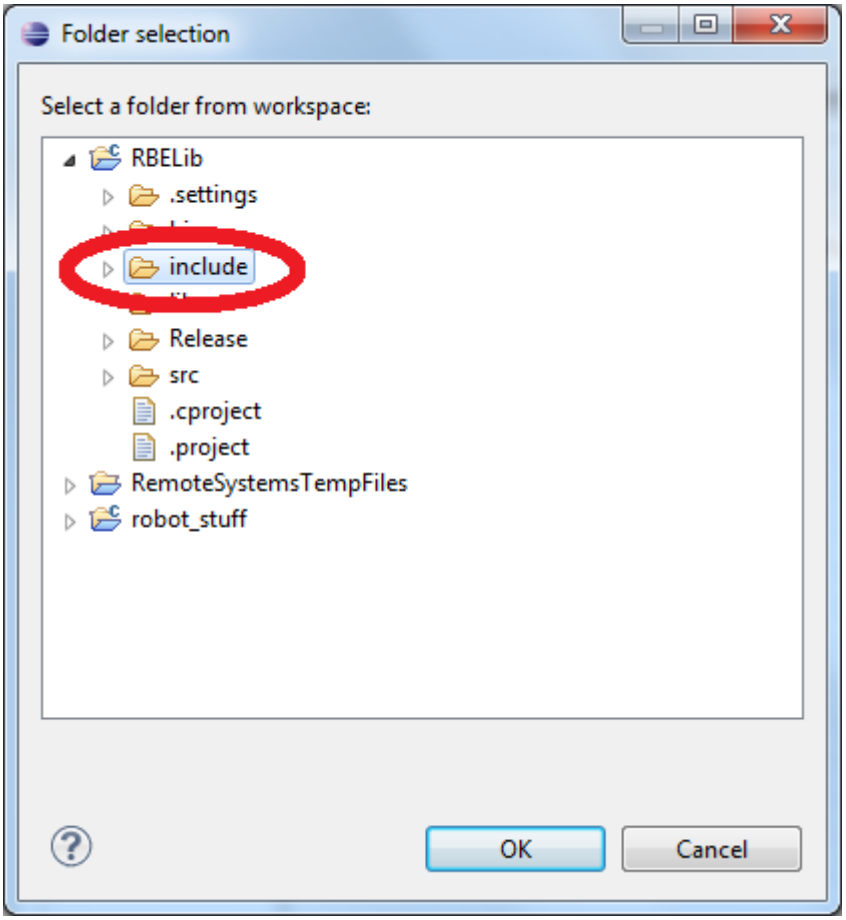
    Which enables printf() to be used with all of the options typically disabled. Your window should now look like the below image.



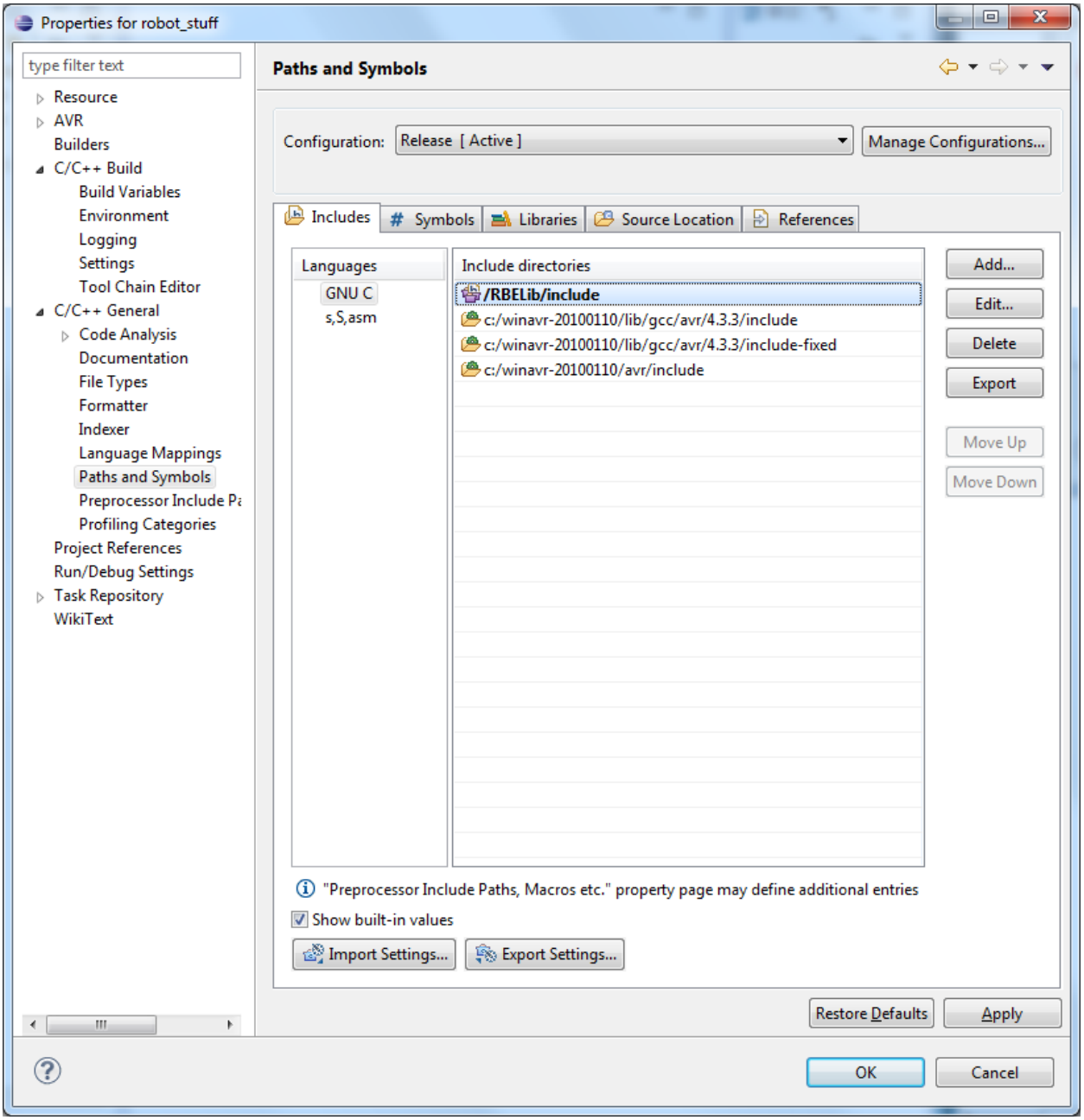
15. Now Select C/C++ General > Paths and Symbols and select Add



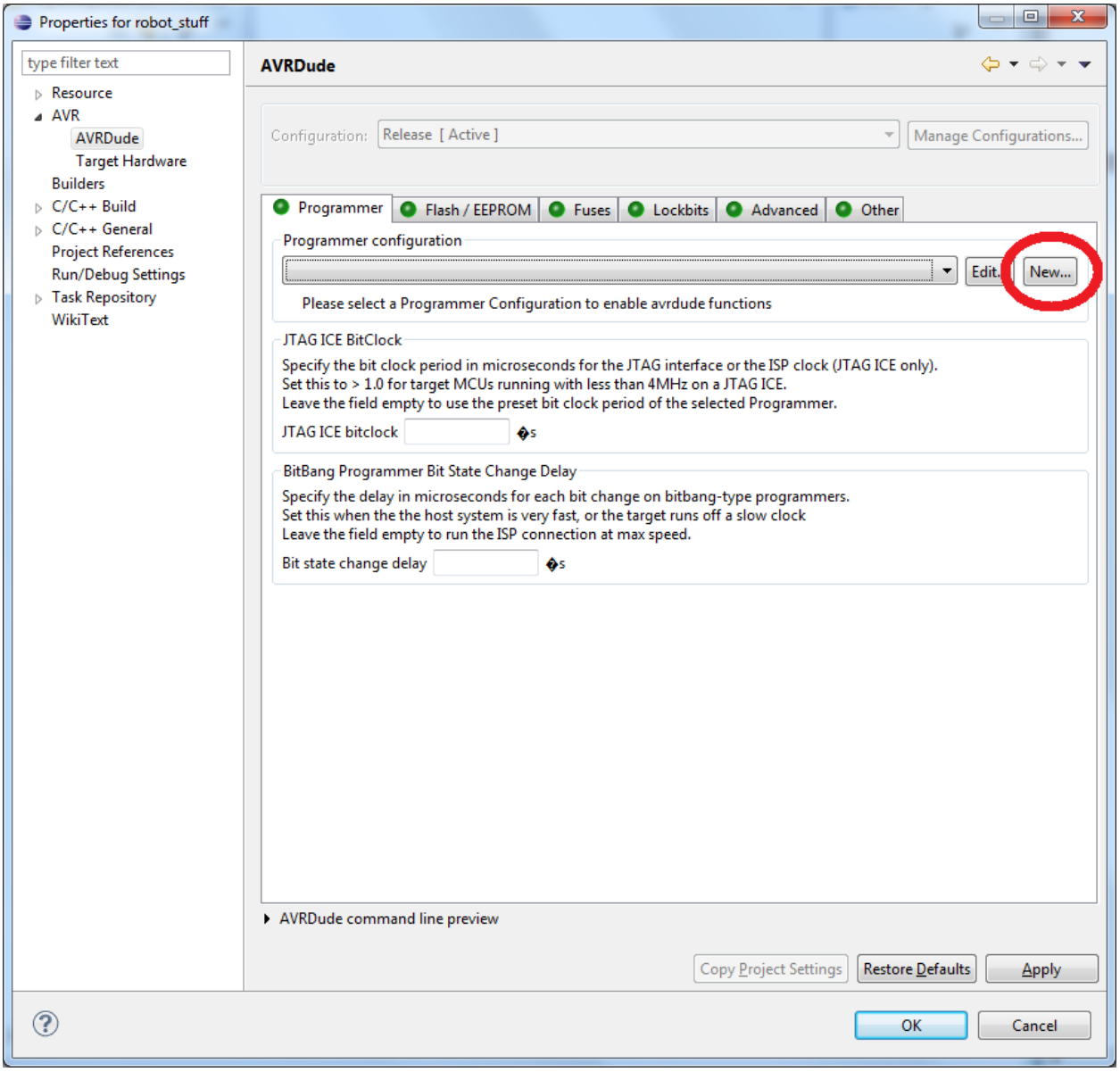
16. Choose the Workspace button
17. Go to RBELib > include



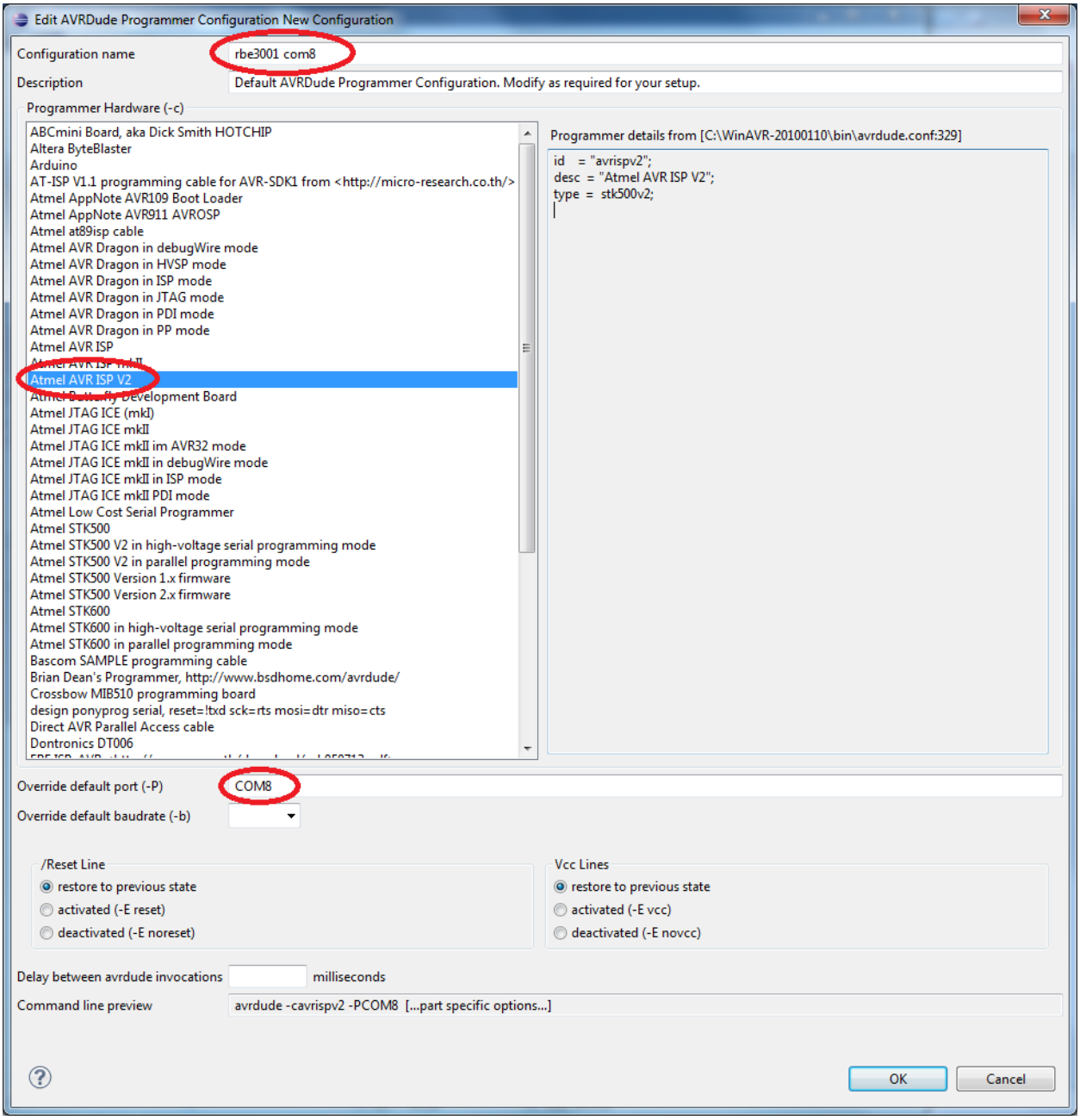
18. Click OK and OK again on the next screen.
19. You should now have something like the following.



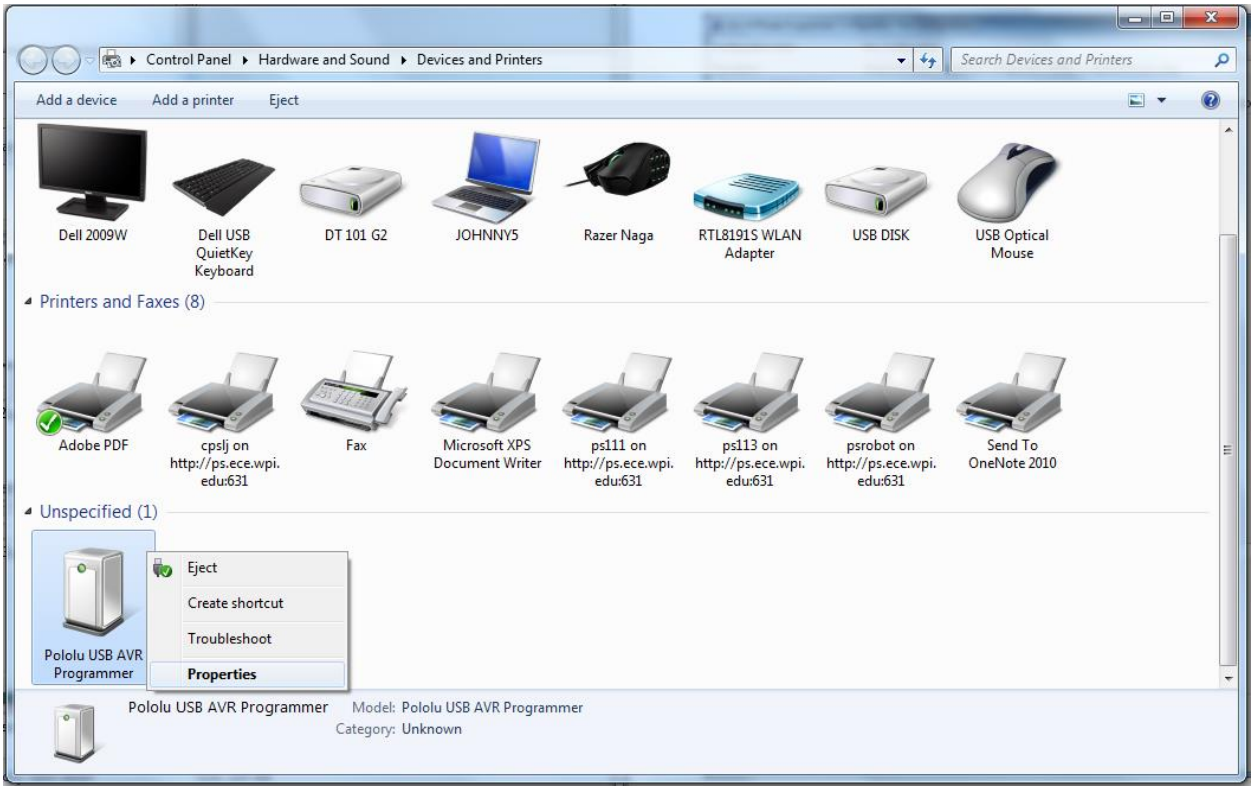
20. Set your programmer to AVR > AVRDude. If you get an Error about invalid values, press OK, and Press OK again to save your settings and restart Eclipse and re-open the project properties and try again. *This is a common error in Eclipse and could happen multiple times*.
21. Select the Programmer Tab and select New



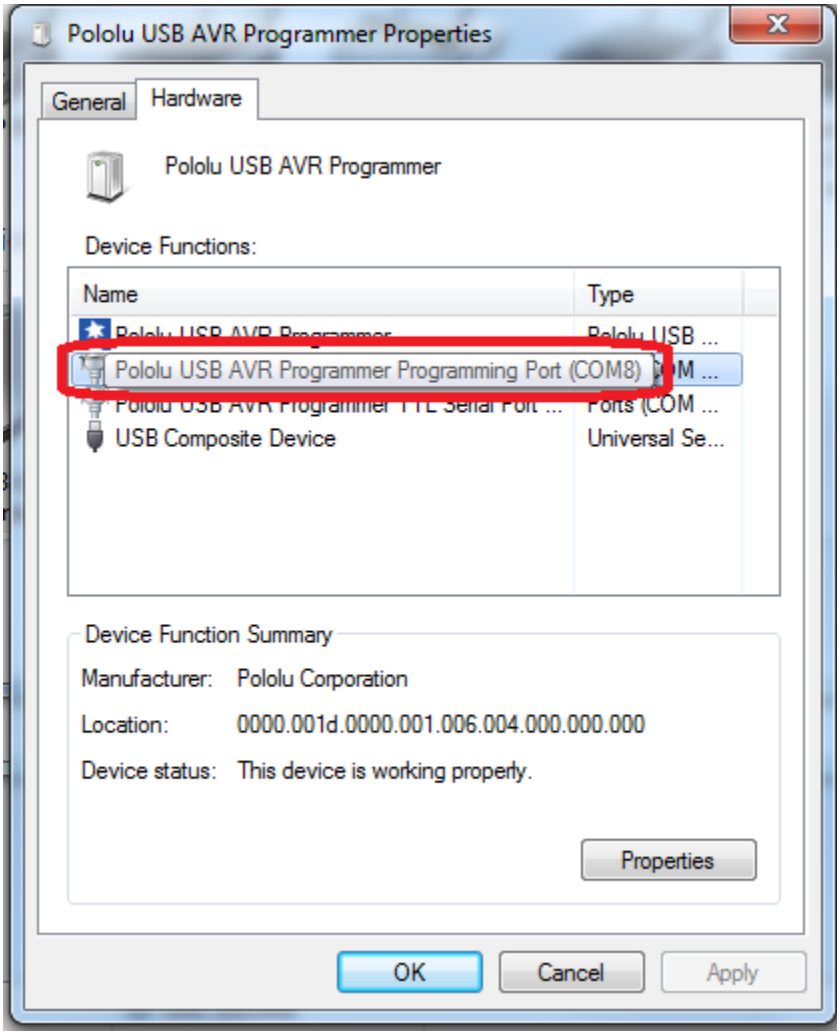
22. Type an identifying name for this configuration, select Atmel AVR ISP V2 and type in the COM port that the programmer is connected to and press OK. See the sub-steps if you don't what your COM port is or how to find it, otherwise skip them.



a. If you need to find your COM port, go to Start > Devices and Printers and right click your device and go to properties



23. Go the hardware tab, and hover over the Programming Port to see what port the ISP uses.



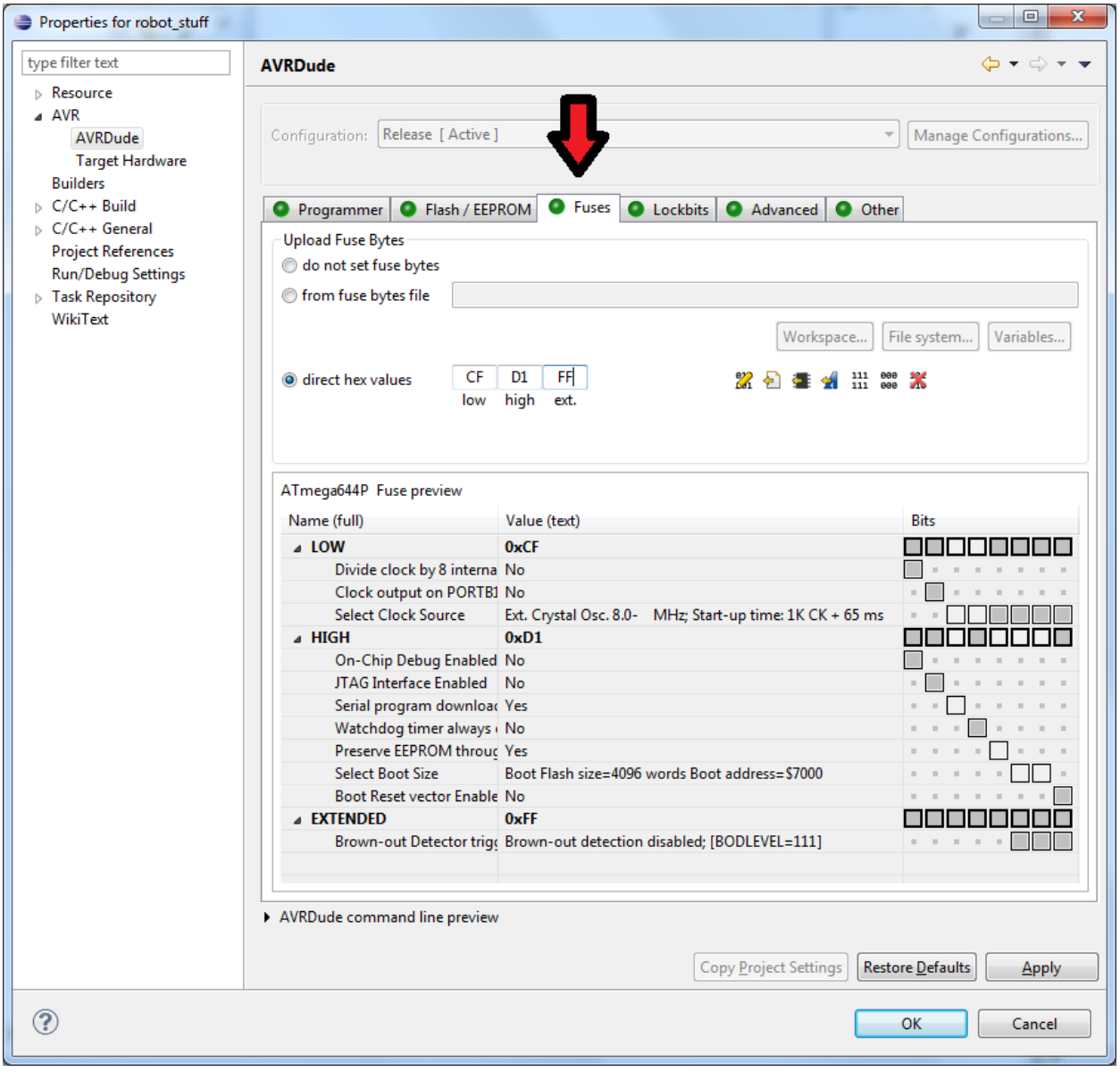
24. Make sure your configuration is selected in the dropdown, then press OK to close this window and accept your settings.

Accessing the Fuses from eclipse

The Fuses control which hardware features you would like enabled. The Fuses can control a number of features such as Crystal selection, brownout detection, or to enable JTAG. For our Purposes we need to set the Fuses to

(Low: CF), (High: D1), (Extended: FF).

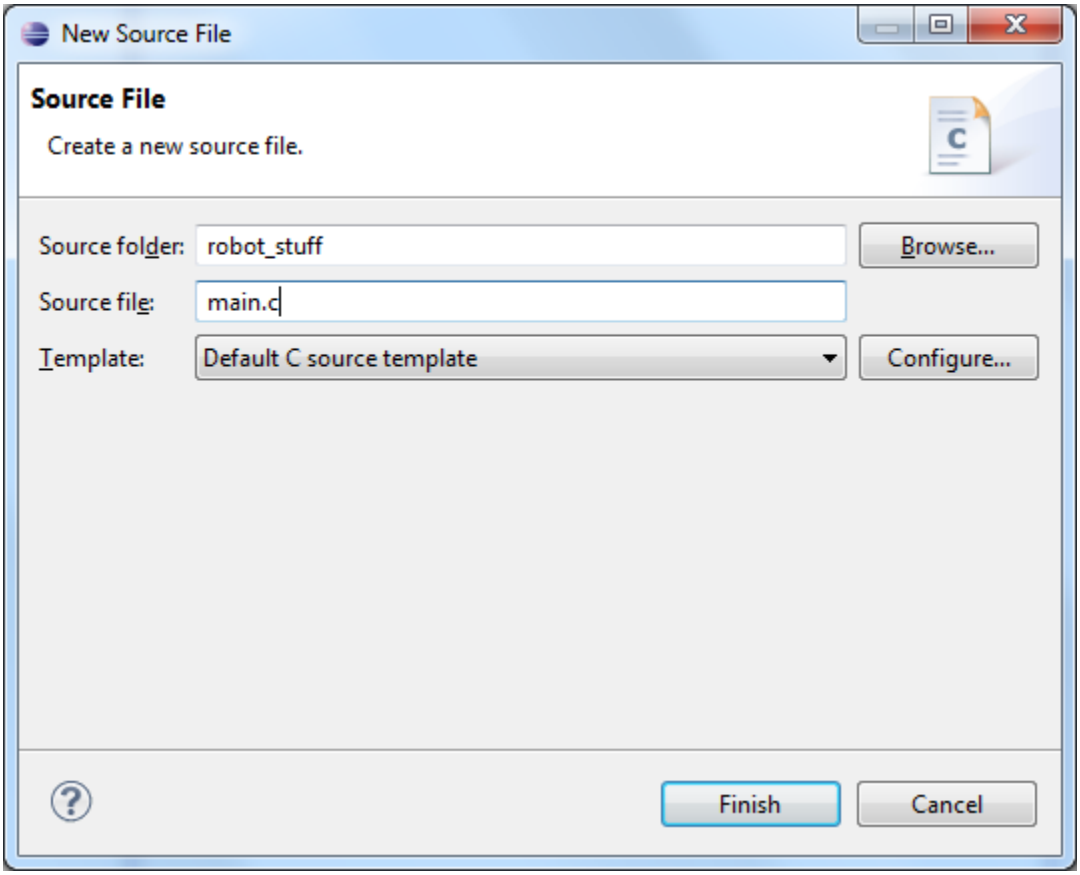
1 Right click on your project and select properties.
2 Select AVR > AVRDude
3 Open the Fuses Tab, choose direct hex values and enter CF D1 FF into the appropriate fields.



Creating a Source File and Committing

Now That the Project is created, the RBELib is checked out and linked, and the fuses and programmer are set we are ready to begin our source file.

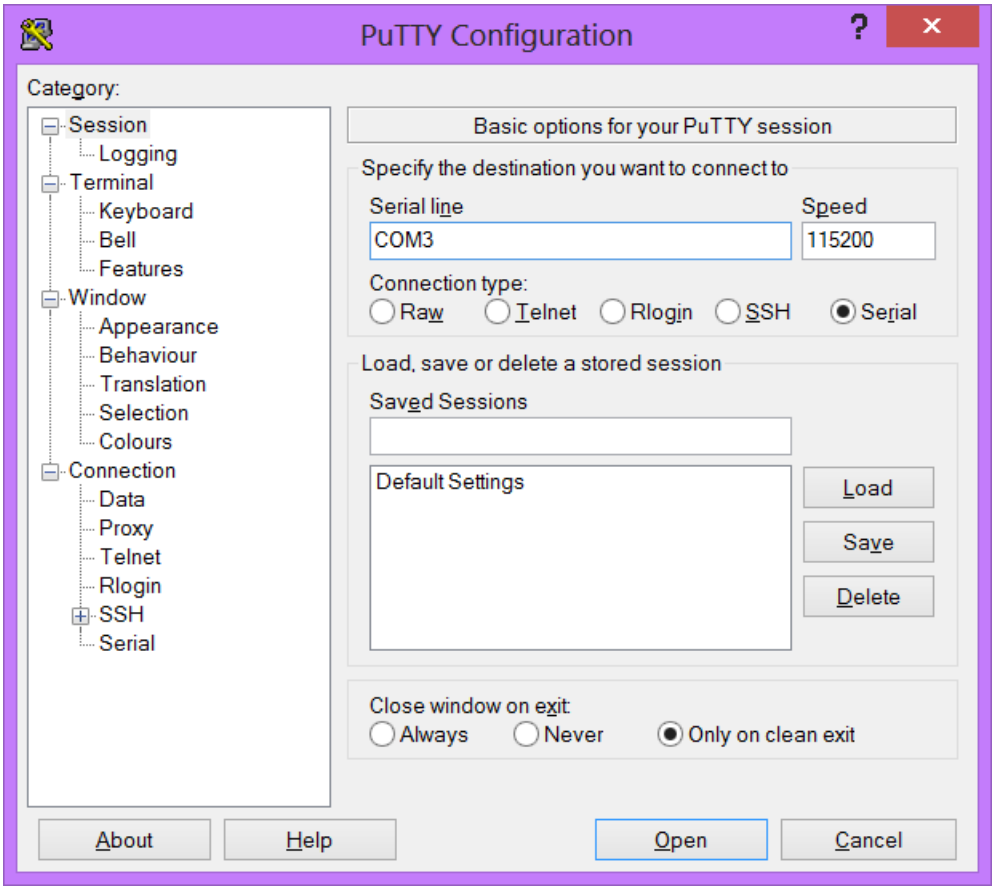
1. Right click on your project and select “New” >” Source File”
2. Name your Source File something.c (do not forget the .c)



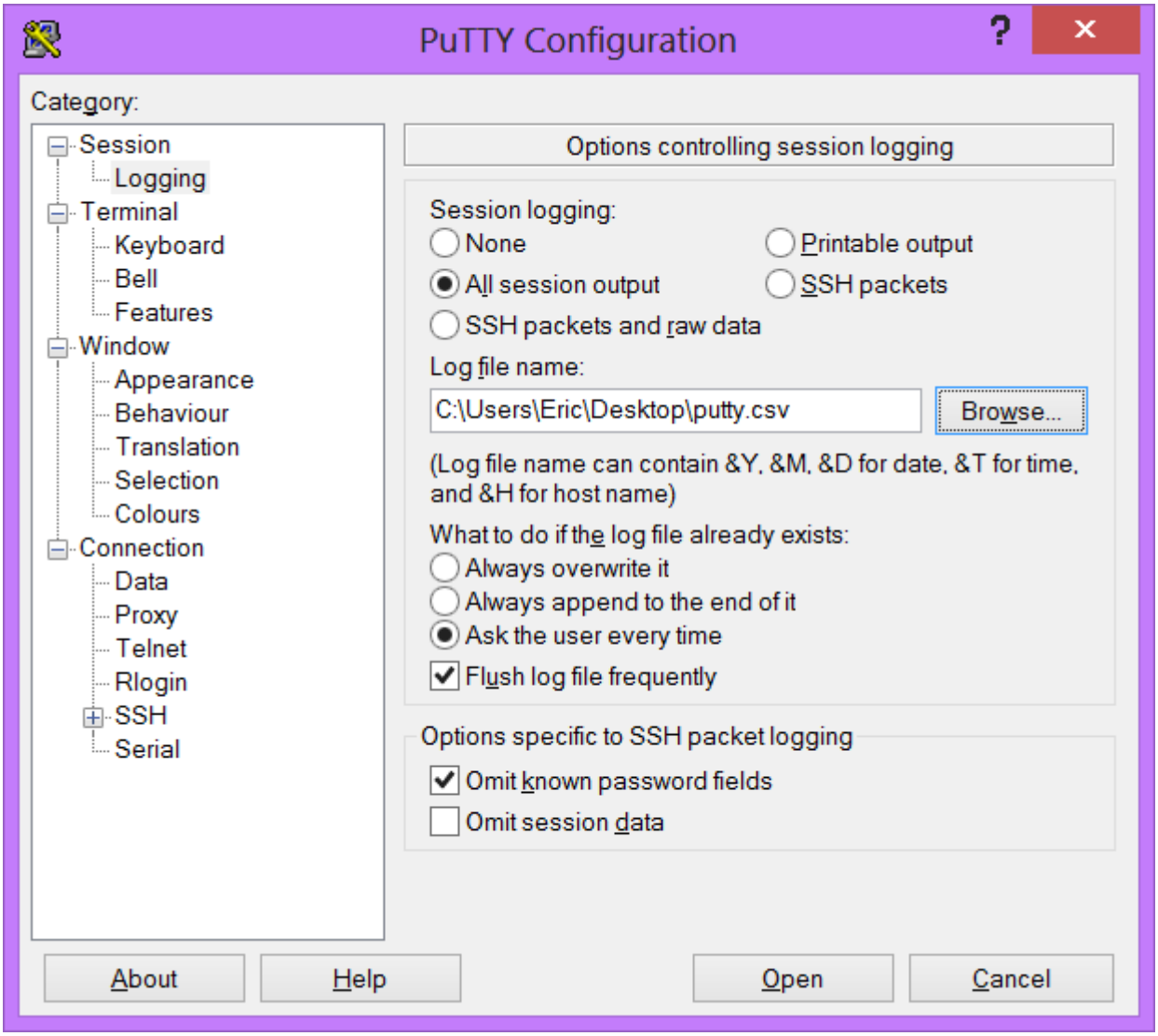
3. Include RBELib and the AVRIO header files, see the below examples for reference.
4. Once your done coding build your project by selecting your project in the project explorer and either press the hammer or by Project > Build Project.
5. Then Upload the hex file that was created by selecting your project and pressing the AVR button. If the build was not successful then refer to the Console window to see the error info. If the build was not successful then there is no .hex file to upload and you need to fix your errors first.

PuTTY

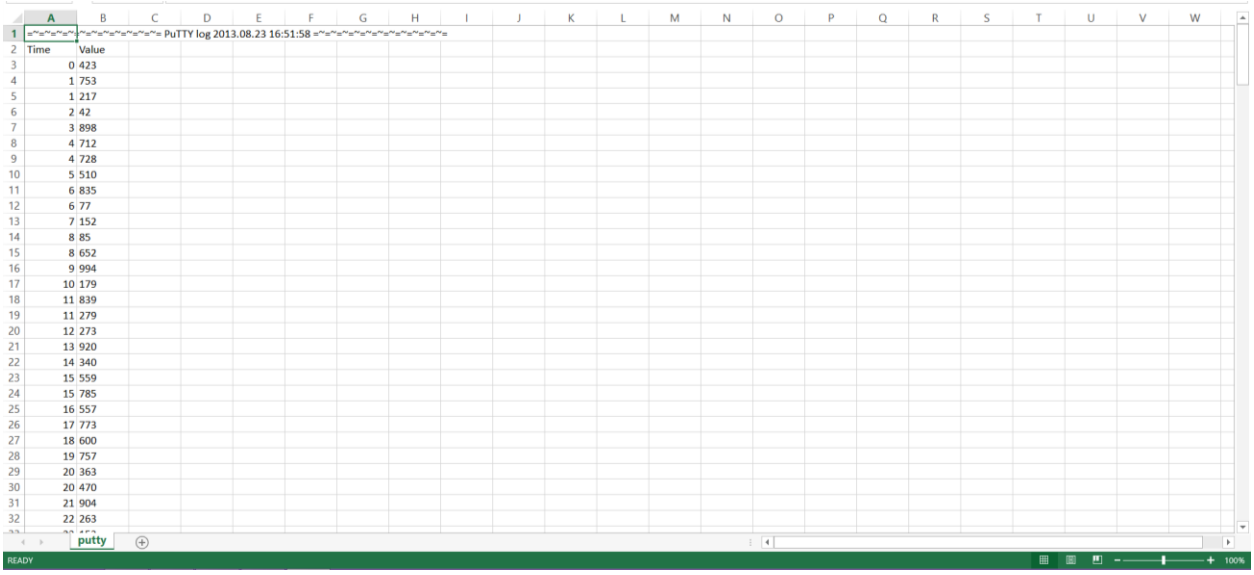
1. Download PuTTY from http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html and choose the Intel x86 putty.exe.
2. Run the exe file to open PuTTY, there is no install process.
3. In the Session category, choose serial, input your baud rate and your serial line.



4. To log data, click on Logging and choose All Session Output and click browse for where to put your file. To save it for Excel or Matlab, make sure to make the extension *.csv.



5. If your code is formatted properly, your output will be logged into a file that can easily be opened by Excel/MATLAB to parse.



	A	B
1	=~=~=~=~= PuTTY log 2013.08.23 16:51:58 =~=~=~=~=~=~=~=~=~=~=	
2	Time	Value
3	0	423
4	1	753
5	1	217
6	2	42
7	3	898
8	4	712
9	4	728
10	5	510
11	6	835
12	6	77
13	7	152
14	8	85
15	8	652
16	9	994
17	10	179
18	11	839
19	11	279
20	12	273
21	13	920
22	14	340
23	15	559
24	15	785
25	16	557
26	17	773
27	18	600
28	19	757
29	20	363
30	20	470
31	21	904
32	22	263

The structure that your print statements should be in is the following (note, the commas are REQUIRED for comma separated value file format *.csv):

Print Column Title 1, Print Column Title n \n (or new line)

<Loop to sample values>

Print Value 1, Print Value n \n (or new line)

<end loop>

This format will give you proper output for parsing.

Example blink port program

Blinks all pins on port B

**Note in order to connect a pin or port to an LED or switch a connection must be made between the pin or port to either the LED or Switch breakout port. To use printing statements, you must first write debugUsartInit()and putCharDebug().*

```
/*
 * main.c
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */


#include "RBELib/RBELib.h" //RBELib

int main(void){
      initRBELib(); //Setup printf() and setServo()
      debugUSARTInit(115200); //Initialize UART
      DDRB = 0xFF; //Set Port as output
      while (1){
            PORTB = 0xFF; //Turn port on
            _delay_ms(500); //Delay .5 sec
            PORTB = 0x00; //Turn port off
            _delay_ms(500); //Delay .5 sec
            printf("Hello\n\r");   //Print "Hello"
      }
      return 0;
}
```

Example Blink Pin program

Register Structures

"Reg Structs" is built into RBELib and allows you to address individual bits instead of whole registers.

```
/*
 * main.c
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */

#include "RBELib/RBELib.h" //RBELib

int main(void){
      DDRBbits._P4 = OUTPUT; //Set Port B Pin 4 to output
      while(1){
            PINBbits._P4 = 0; //Sets Port B Pin 4 to low
            _delay_ms(100); //Delay .1 sec
            PINBbits._P4 = 1; //Sets Port B Pin 4 to high
            _delay_ms(100); //Delay .1 sec
      }
      return 0;
}
```

Now that we have some code, we can upload it to our SVN repository. To add a project the first time to the repository, right click your folder in the project explorer and click on Team > Share Project. Select SVN as the repository type and click next, then make sure Use existing repository location is selected and pick your groups repository and click Finish. You may be asked to type in a comment for your commit, you can put in anything and press OK.

Once you've committed the project at least once, in the future you only need to click on your project and go to Team > Commit and then put in a comment and it will upload whatever files have been updated.

Splitting Files

You should split your code into multiple files by using header files with corresponding C files, each file would contain code relevant to the file it's in. Ex, you put your initialization code into a file called init.c, your motor driving code into motors.c, etc.

You can also include global variables in a header file and share them amongst all your files.

General layout of a header file:

```
/*
 * ex_header.h
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */

#ifndef EX_HEADER_H_
#define EX_HEADER_H_

//Declarations / constants / includes / etc.
//Actual code (logic) does NOT go here!

#endif /* EX_HEADER_H_ */
```

If we wanted to separate the blink program into multiple files, we could create another file for controlling port B which does the LEDs. The first will be a header file which will have all of our includes in it so we only have to include this one file after it's setup.

```
/*
 * main.h
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */

#ifndef MAIN_H_
#define MAIN_H_

#include "RBELib/RBELib.h" //RBELib

#include "portB.h"

#endif /* MAIN_H_ */
```

The next file is the header file for port B.

```
/*
 * portB.h
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */

#ifndef PORTB_H_
#define PORTB_H_

#include "main.h"

#define PORT_OUTPUT 0xFF

void initPB();
void blinkAll(double timeDelay);

#endif /* PORTB_H_ */
```

Now we need the .c file that has all of the code to control the port.

```
/*
 * portB.c
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */


#include "main.h"

void initPB(){
      DDRB = PORT_OUTPUT;
}

void blinkAll(double timeDelay){
      PORTB = 0xFF;
      _delay_ms(timeDelay);
      PORTB = 0x00;
      _delay_ms(timeDelay);
}
```

And finally our main code.

```
/*
```

```
 * main.c
 *
 *  Created on: Aug 6, 2014
 *      Author: ewillcox
 */

#include "main.h"

void initializations(){
      initPB();
}

int main(){
      initializations();
      while(1){
            blinkAll(100);
      }
      return 0;
}
```

If you wanted to share your globals from one file, you can create a header file to declare them all, and initialize them (if desired) in a c file by doing the following.

```
/*
 * globals.h
 *
 *  Created on: Feb 6, 2013
 *      Author: joest
 */

#ifndef GLOBALS_H_
#define GLOBALS_H_

typedef struct {
      double timeCount;
} Globals;

extern Globals globals; // declaration

void initGlobals();

#endif /* GLOBALS_H_ */
```

```
/*
 * globals.c
```

```
 *
 *  Created on: Feb 6, 2013
 *      Author: joest
 */
#include "globals.h"

Globals globals;

void initGlobals() {
      globals.timeCount = 0;
}
```

You can now call timeCount by adding in globals.h into our previous main.h and then calling globals.timeCount where you want to get the value.

Common Problems

A lot of problems in Eclipse can be solved by either restarting the program or by making a new project and copying your code over. Please try this first if nothing below works for you before asking for help.

SVN lockout

If you accidently enter the wrong info for SVN, just right click on the repository and click on location properties and enter the correct info.

Error 1 / a lot of code not being recognized

Make sure you included all of your header files (#include), especially RBELib. If you do have all of your necessary headers included (in each .c file!) make sure you setup the project correctly, re-check all the steps from above. If both of these are correct restart Eclipse, and if the error still persists try to make a new project and copy your code over.

Component X is not working

If you DAC/Current Sensor/FTDI/etc isn't working, check that all jumpers are properly set on the board. Also make sure that you didn't leave switches connected from Lab 1 if you are not using them!

Current sensor is outputting a waveform

See Joe, it's broken.