FILE I/O, DEBUGGING, MAKEFILES

Sections:

- Files: 7.5, 7.7
- typedef: 6.7
- Type Casting: 2.7

Eclipse: Importing an Existing Project

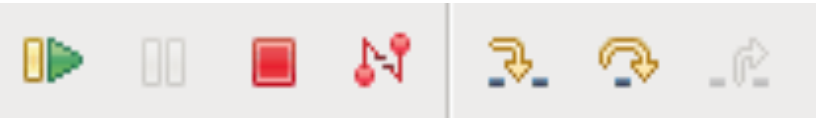
- File -> Import
- Open General -> Existing Projects into Workspace
- Next
- Select the root directory
- Check the box (copy projects into workspace)
- Finish

Debugger 

- Eclipse integrates a debugger just like Visual Studio
  - uses gdb
- Open up CS300CodeExamples
  - In the Makefile, remove bin/defineVsConst from TARGETS
  - Build the project
  - Open the Binaries list on the left
  - Right-Click pointersWorksheet

Debug As | Local C/C++ Application | gdb/mi

Debugger



- The debugger stops on the first statement in main()
  - Resume (Run to breakpoint)
  - Suspend (Pause)
  - Terminate
  - Disconnect
  - Step Into
  - Step over
  - Step return (Step out of)

Step Over One Instruction

- Notice the Variables in the top right

  i int 0

- Press Step Over 

- Notice the Variables in the top right

  i int 1

Breakpoint

- Right click the blue gutter beside printf()
  - Toggle Breakpoint

May need to set Breakpoint Type to C/C++

- Run to Breakpoint 
  - What is the value of i?
  - Step Over
  - Check the console on the bottom

Conditional Breakpoint

- Right Click that same breakpoint, the pale blue dot
  - Breakpoint Properties
    - Common
    - Condition: `i == 4` Just C code!
    - Stop
- Restart Debugging
- Run to Breakpoint 

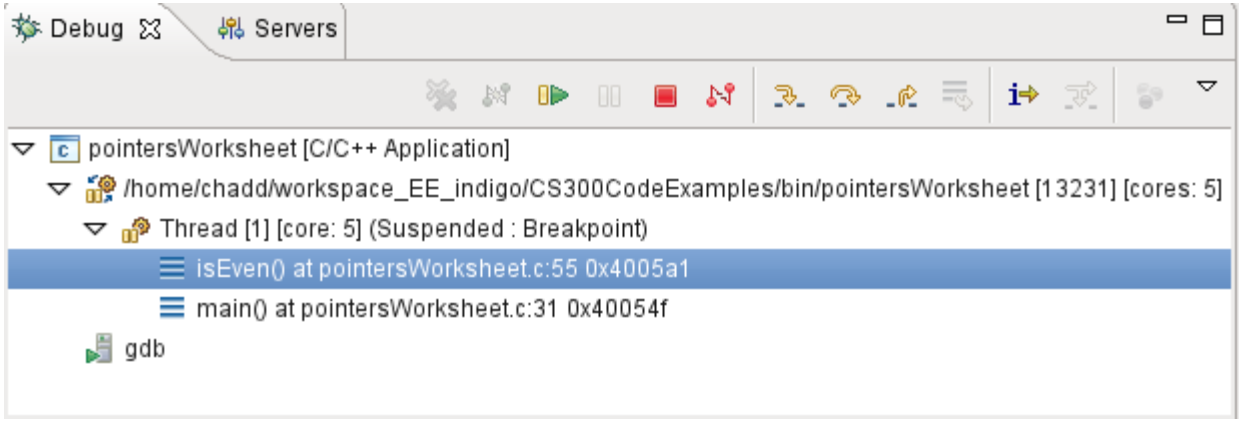
Conditional Breakpoint

- Be wary of function calls as conditions
- Be wary of anything that accesses dynamic memory
  - (a null pointer in your condition!)

    Error in testing breakpoint condition:
    Cannot access memory at address `0x0`
- Ignore count: skip this break point X times
- Actions: Sound/Log/Resume/External Tool
- Filter: restrict to certain threads

Stack

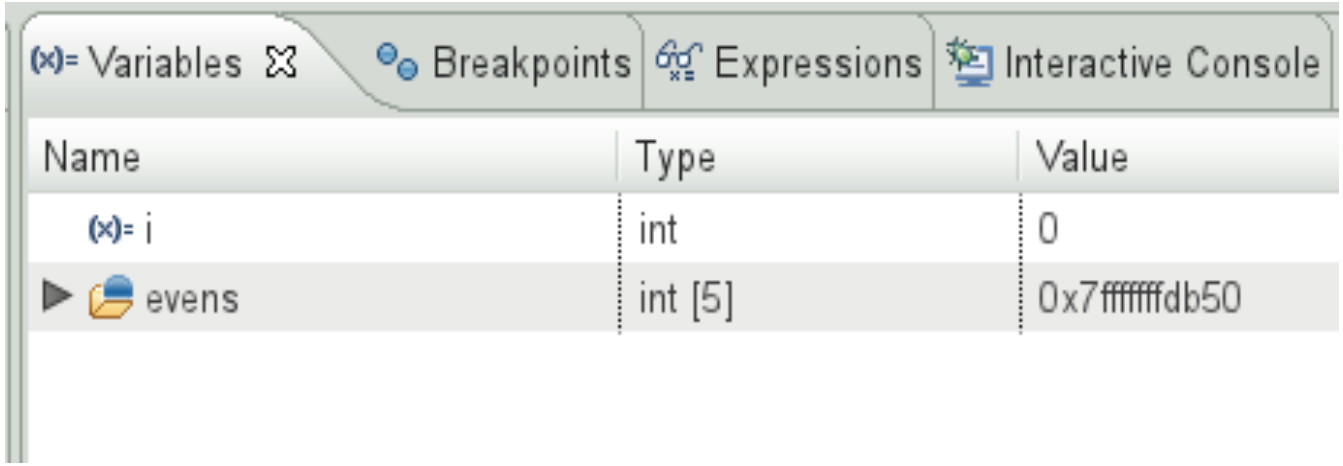
- Put a breakpoint on line 55 printf()
- Disable breakpoint on line 33
- Run



- Select a function to see variables in that function

Arrays

- Add `int evens[MAX_NUMS];` to main().
- Run



- Drop down `evens` in Variables view

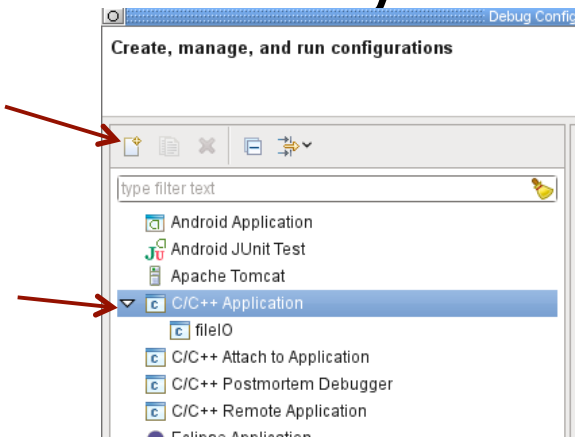
charArraysAndStrings

Flip back to C/C++ Perspective

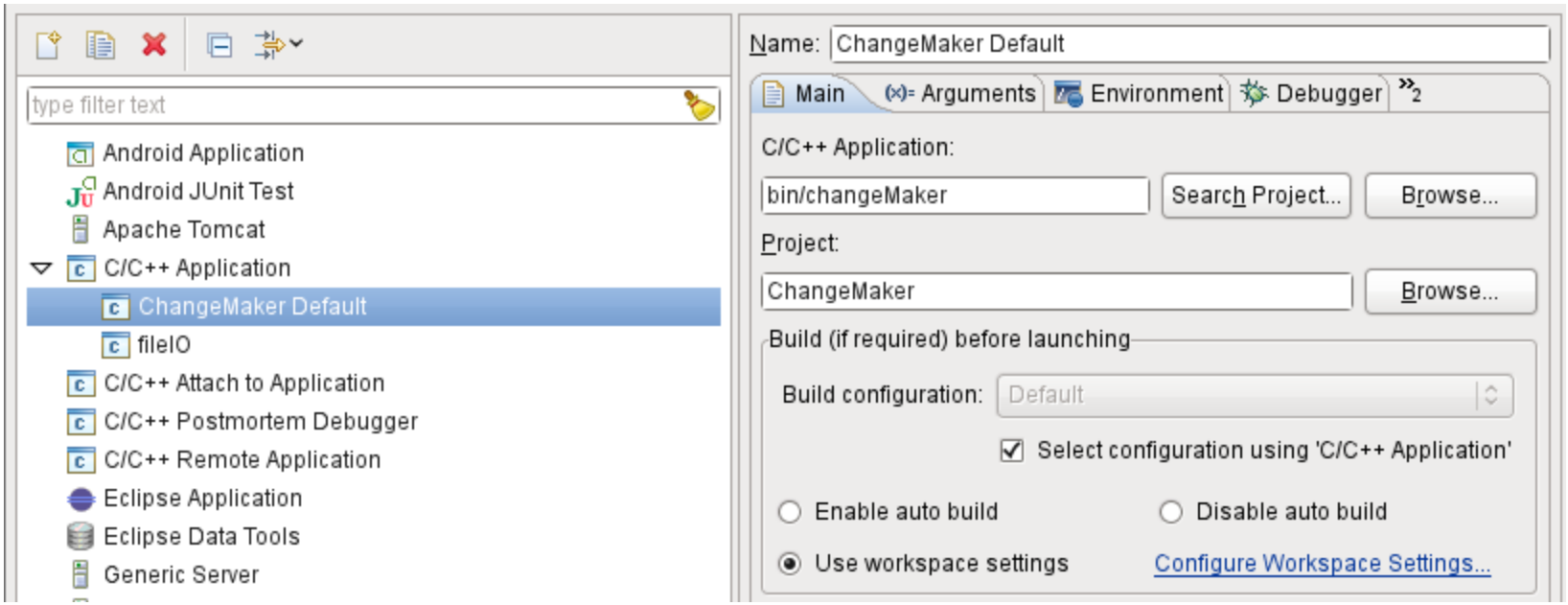
- charArraysAndStrings | Debug As
  - What is the first statement of main()
  - What is currently in charArray?
- Step to the first printString()
- Drop down pString & charArray
- Right click pString
  - Display As Array 0 12
  - What is in pString[11]?
- Right click pString | Restore Original Type

Debugger

- Run -> Debug Configurations
- Select C/C++ Application from the list
- Press the New button
- If there is only one executable in the project the fields will be automatically filled out



Debugger

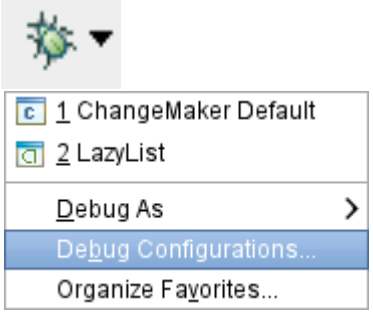


- Press Debug at the bottom to run the debugger

Debugger

- OR
  - Right click Project
  - Debug As | Local C/C++ Application

- In the future, the configuration will show up in the bug drop down list



Eclipse: Open fileIO.c

- fopen
- fgetc
- fscanf
- fprintf
- close
- errno
- perror

Build Process

- Compiler takes source files (.c) and and outputs object files (.o)
- Linker takes the object files and creates an executable

- If we have these files:
  - main.c, hello.c, factorial.c, functions.h

- The trivial way to compile these is to type in the command line:
  - gcc main.c hello.c factorial c -o hello

- Makefiles are used to automatically build the executable

Makefiles

- Description of how to build your executable
- Useful if you have multiple source files
- If you run
  - make
- In your command line, then it will look for a file named Makefile in the current directory

Basic Makefile

```
target: dependency1 dependency 2
        command1
        command 2
```

Tab

Given a set of dependencies, make will only run the necessary commands to build the project. Build a **dependency graph**.

If a target is older than any of its dependencies the commands are run to build the target.

```
CC=gcc
CFLAGS=-g -Wall

.PHONY: all clean tarball

all: sievedriver

sievedriver: bin/sievedriver.o bin/sieve.o
	${CC} ${CFLAGS} -o sievedriver bin/sievedriver.o bin/sieve.o

bin/sievedriver.o: src/sievedriver.c include/sieve.h
	${CC} ${CFLAGS} -o bin/sievedriver.o -c src/sievedriver.c

bin/sieve.o: include/sieve.h src/sieve.c
	${CC} ${CFLAGS} -o bin/sieve.o -c src/sieve.c

clean:
	rm sievedriver bin/*.o

tarball: clean
	tar czf ../CS300_2_PUNetID.tar.gz ../CS300_2_PUNetID
```