Basic C,
Program Compilation
p 1-42

C coding tool chain

- Pre-processor (gcc -E)
- Compiler (gcc -c)
- Linker (ld, invoked via gcc)
- Loader (ld-linux.so)
- Debugger (gdb)

Example Code

- Download and untar CS300_Example_Code.tar.gz

- scp punetid@zeus.cs.pacificu.edu:/home/CS300Public/2018/CS300_Example_Code.tar.gz .

- tar zxf CS300_Example_Code.tar.gz

- Open Eclipse and point your workspace at CS300_Example_Code

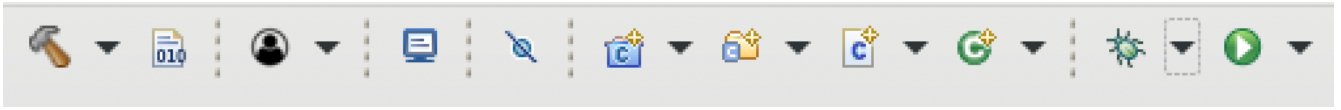
Open HelloWorld

Differences from C++ ?

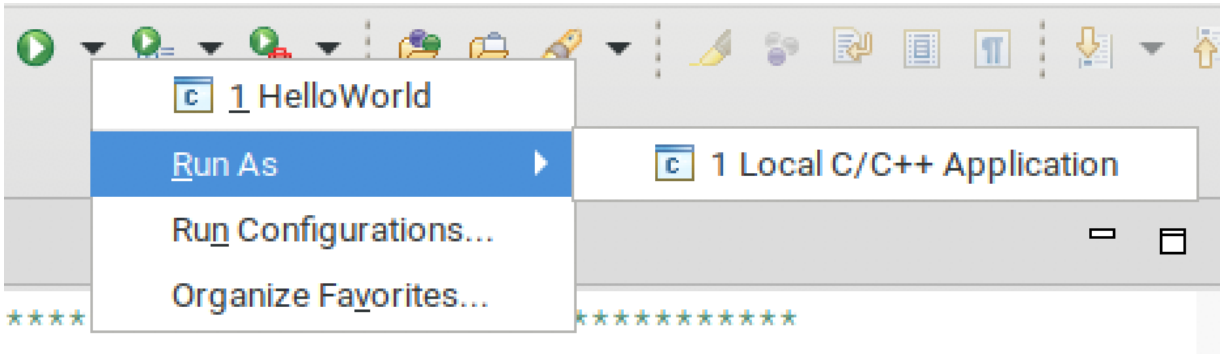
```
/* this is a comment */

#include <stdio.h>


int main ()
{
   printf ("hello world\n");
   return 0;
}
```

Save, Build, Execute

- Hit the Build button



- Hit the Execute button



Build on the command line

cd CS300_Example_Code/HelloWorld

gcc -Wall -o bin/runMe src/helloworld.c -g

bin/runMe

- The bin/ is necessary, why?

  echo $PATH

*Separate Compilation

gcc -Wall -c -o bin/helloworld.o src/helloworld.c -g

gcc -Wall -o bin/runMe bin/helloworld.o -g

ls -altr bin

*Remember “Additional Dependencies” from CS250 Visual Studio (Project Management -> Random.obj)

Define

- Pointer
- Memory Address
- Value

ExamScores.c

- comments
- include
- define
- function prototype
- static
- int *
- scanf
- printf
- for loop/do while loop
- const
- malloc/free

CS300CodeExamples pointerToStaticData.c

```c
#include <stdio.h>


int main ()
{
   int value;
   int *pValue = &value;

   value = 8;

   printf ("%d %d\n", value, *pValue);

   return 0;
}
```

Pointers & functions pointers.c

```
void printIt (int *pInt)
{
   int input;
   scanf("%d", &input);
   *pInt = input * 2;
}


int value;
printIt( &value );
```

Pointers & functions

```
void printIt (int *pInt, int size)
                    // int pInt[]
{
  int i;
  for(i = 0; i < size; i++)
  {
    printf("%d %d %d \n",
       (unsigned int) (pInt + i),
        pInt[i], *(pInt + i) );
  }
}
```

Output if the base of the array is location 1000 and size is 4?

C Topics

```
void foo(int arr[], int len,  char *str)
{
   int index = 0;
   for( ; index < len ; index ++)
   {
        printf("%d\t", arr[index]);
   }
   printf("%s\n", str);
}
// the function call
foo(array, ARRAY_SIZE, "the message");
```