Setting up a Windows 7 64 bit development environment

Preface:
This was adapted from the guide posted on http://wiki.osdev.org/GCC_Cross-Compiler. The osdev.org wiki explains in detail about why a cross compiler is needed and I suggest going through that guide over this one.

This was tested on Windows 7 64-bit; however, it should be applicable to all versions of Windows Vista/7/8 both 32-bit and 64-bit. Windows XP should also be compatible; however, it’s likely directory locations may need to be altered.

Finally, while this will allow you to do native development on Windows, you will still need to be comfortable with Linux as Cygwin is “a collection of tools which provide a Linux look and feel environment for Windows.” -Cygwin.com

Software:
Cygwin (download)

Install Cygwin:
NOTE: If an install step is omitted, choose the default option and click next.

Any directory will do. I simply like to keep everything together.
Any mirror will do just fine.
Ignore, click OK

Expand the various package categories and locate the packages in bold below. Install the packages by clicking the word “Skip” to the left of the package name. This will toggle the install method and will be noted by the package’s version number. Use the search if you are unable to find certain packages.

**Required packages to install:**

(Location->PackageName)

- Devel->make
- Devel->nasm
- Devel->gcc4-core
- Devel->gcc4-g++
- Libs->libgmp-devel
- Libs->libmpfr-devel
- Libs->libmpc-devel
- Web->wget

**Recommended packages to install:**

- Utils->ncurses

Below is an example of a wget and it’s location.
<table>
<thead>
<tr>
<th>Category</th>
<th>New</th>
<th>B.</th>
<th>S.</th>
<th>Size</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td>wget: Utility to retrieve files from the WWW via HTTP and FTP</td>
</tr>
</tbody>
</table>

- Search:
- Select packages to install
- Category: Web
- Package: wget
- Size: 56K
- Description: Utility to retrieve files from the WWW via HTTP and FTP

Created for CPSC 410/611 Operating Systems
Created by Blake Biggs
Last edited 9/1/2012
Let it automatically resolve dependencies.

Click Next. This will take some time as it downloads and installs the packages. Once complete, let it create an icon either on your Desktop or Start Menu click finish. Find the icon and open up the Cygwin terminal.

Copy and paste the following commands into the Cygwin shell.
NOTE: There are a total of three separate commands. The wget command is multiline.

```
wget --no-check-certificate https://dl.dropbox.com/s/30ygi67c3w7u7yp/setup_cygwin_dev.sh?dl=1 -O setup_cygwin_dev.sh
chmod +x setup_cygwin_dev.sh
./setup_cygwin_dev.sh
```

This builds the cross compiler, and places it into the directory cross_compiler and adds the binary location to your path. This can take quite a while to complete.

NOTE: Your compiler is now i586-elf-gcc, and your linker is i586-elf-ld. These will be used from here on out when generating binaries and future makefiles will also require the changes below.

Updating the given makefile for MP1:

Delete makefile, and rename makefile.linux to makefile. Replace all instances of gcc with i586-elf-gcc and ld with i586-elf-ld. Change the binary format for nasm from “aout” to “elf”. It should look like the following.

```
all: kernel.bin

clean:
    rm -f *.o
# ==== KERNEL ENTRY POINT ======
start.o: start.asm
    nasm -f elf -o start.o start.asm
# ==== UTILITIES ======
utils.o: utils.H utils.C
    i586-elf-gcc $(GCC_OPTIONS) -c -o utils.o utils.C
# ==== DEVICES ======
console.o: console.H console.C
    i586-elf-gcc $(GCC_OPTIONS) -c -o console.o console.C
# ==== KERNEL MAIN FILE ======
kernel.o: kernel.C
    i586-elf-gcc $(GCC_OPTIONS) -c -o kernel.o kernel.C
kernel.bin: start.o kernel.o console.o utils.o linker.ld
    i586-elf-ld -T linker.ld -o kernel.bin start.o kernel.o console.o utils.o
```