Eclipse Basics

Required Utilities

The CDT requires the following utilities be installed:

- Build – (make)
- Compile – (gcc)
- Debug – (gdb)

Terminology

In Eclipse there are two main layers: the model layer (workspace) and the user interface layer (perspective).

- **Workspace** - a collection of resources (projects, folders and files). Basically the workspace is where the user files are located for use in the tool.

- **Perspective** – the visible actions and views within a window. The perspective displays the actions or views appropriate to the environment. Generally there is a different perspective for each code editor/tool (C/C++, Java, Debug, etc), and you can switch between these perspectives.

Setting the Workspace

When you launch Eclipse each time, the Workspace Launcher will appear and you will be prompted to select the workspace to be used for the session, (basically indicates where project files will be stored).

The default directory for a workspace under Linux is /home/user/workspace. Browse to change from the default workspace. Selecting the check-box from the Workspace Launcher will set the selected workspace as the default, and you will not be prompted each time Eclipse is launched.

Choosing the Perspective 

On open, Eclipse defaults to the last used perspective. The name of the perspective is visible on the Title bar at the top of the window. To change the perspective, from the top-right corner choose the appropriate icon, or the drop-down arrow **>>** to see all available perspectives. Debugging is a separate perspective as well. If you attempt to debug a program while still in the C/C++ perspective, you will be prompted to change perspectives.

Creating a New Project

Standard Make Project – You need to create a makefile or use an existing makefile in order to build your project.
Managed Make Project – Uses a makefile automatically generated by Eclipse. This is the one we will use.

1. **File->New->Managed Make Project** displays the New Project Wizard.
2. **New Project Dialog**
   - ✓ Enter the name of the project. The name will be different from the source file name. The project will automatically be stored in the default workspace (/home/userName/workspace). To change the location, un-check the check-box

and browse to select new location.

- ✓ Once entered, choose Next.
- ✓ From the Project Type dialog, make sure that Debug and Release are both selected(usually default). Accept the defaults and choose Finish.

Creating Source Code File

Once you have created the project, you can create a new source code file in the project.

1. Right-click on the Project Name in the left project list pane.
2. Choose New->Source File.
3. Give your file a name. You should include the appropriate extension (.c, .cpp), Eclipse will not add the extension to the name. If you forget to include the extension, it will not be recognized as a C/C++ source code file. *(Although you will be warned that the file type is invalid when naming the file, it is also easy to identify this error because when you enter code in the editor, none of the text will be highlighted, and you will not be given the option to build/run as C/C++ application)*

Identifying Syntax Errors

The editor will identify and highlight any syntax errors and warnings on Build/Save. When a line of code contains an error, it will be highlighted with a red underline, and a red X will display in the left margin. To review the error/warning message, hover the mouse over the line or click on the red X and the text of the message will display.

Building a Project

- ➔ Before executing a program, it needs to first be built. You can choose to manually build the project each time, or have the project build automatically after changes are made, and before executing.
- ➔ To build automatically, choose Project->Build Automatically. A check will appear beside the selection, and if any changes are made to the code, it will be rebuilt before execution.
- ➔ **ERROR WARNING:** Before building the first time, you should always save the file. After building/executing the first time, you will always be prompted before building to save each time the code is changed, but depending on how you build/execute, you may not be prompted the first time. If after building in the Console area you see the message "undefined reference to `main'" and when you run, you see the message "Launch Failed No Binaries", it means the build was not successful. Try manually saving the file and building/running again.

Running a Project 

You can run a project several ways, some options are:

1. Right-click on the name of the project in the list, and choose "Run As-> Run Local C/C++ Application"
2. First select the project name in the project list pane, and then click the green run icon from the tool-bar.
3. From the tool-bar, click on the drop-down arrow next to the green Run Icon, and from there choose the project you wish to run. Clicking on the arrow without selecting the project (either by using the drop-down or selecting the project name in the navigation pane) will run the last run project (not necessarily the current project).

Importing Source Code Files into Existing Project

1. Create the project as before.
2. In the left Project Navigation window, right-click on the project where want to import the source code.
3. Choose "Import..." and the import wizard will appear.
4. From the import window, choose "File System" as the import source and Next.
5. Choose the directory that contains the source file to import and OK (not the file itself, you can only select directories).
6. Select the file(s) to import by clicking the check-box and "Finish".

Importing an Existing Project into Eclipse

1. Select File->Import, and an import dialog will appear.
2. From the list, choose "Existing Projects into Workspace", and Next.
3. Choose the root directory where the project directory resides (not the actual folder that
4. contains the project files, but its parent, the folder that contains the project folder).
5. You will see a list of projects in the selected directory, select the project that you wish to import and choose Next.
   Note about check-box: if unchecked, the project will remain in its original location, and any changes will be made to the original project files. If selected, a copy will be made of the project, and placed in the default workspace. The copy will be the one modified/compiled/executed.

Debugging a Project 

Debug Perspective 

To invoke the debugging program (gdb in this case), you click on the green 'bug' icon on the toolbar. If your current perspective is not the debug perspective, you will be prompted to change perspectives.

Before invoking the debugger, you can change to the debug perspective in the same way you change to the C/C++ or Java perspectives, by using the icon/arrow at the top-right corner of the C/C++ perspective.

The debug perspective has 4 main areas:

1. Debug window/pane(top left corner) – This window pane contains a tool bar with icons to step through/debug program, and a window that shows the debug commands.
2. Breakpoints/Variables panes (top right corner) – Contains panes to display the variables and breakpoints. The values of the variables during step through are displayed in the variable pane, and the list of breakpoints will be displayed in the breakpoint pane.
3. The source code window, displays the source code and position as we step through the source code.
4. Console window that displays the program output.

Stepping Through – Line by Line

1. Change to the debug perspective.
2. Click the arrow to the left of the green 'bug' icon, and select the project to debug. If you just click the debug icon without selecting the project, it defaults to the last run project, which is not necessarily the one to debug. This will invoke the debugger.
3. Click the "Step Into" icon, which will execute the code line by line, stopping after each statement, and display the current values of the variables in the Variable pane.

4. Any changes in variable values will be highlighted in yellow.

Using breakpoints/resume

1. Change to the debug perspective.
2. Set breakpoints - Just clicking the resume button will cause the program to execute from the beginning to the end, without stopping or showing the value of the variables at any given point. To set breakpoints that will force the application to stop at important points in the code, in the source code window, either double click in the margin to the left of the breakpoint, or right-click and choose "Toggle Breakpoint".
3. To remove a breakpoint, right-click on the breakpoint to remove, and again choose "Toggle Breakpoint".
4. Now run the debugger, by clicking on the arrow to the left of the green 'bug' icon, and select the project to debug.
5. The program will stop at the given breakpoints, by clicking on the resume button you will be taken to the next breakpoint.

Editing Display Preferences

You can edit how the code and syntax displays in the editor by setting your display preferences. To change the preferences, choose:

Window->Preferences – change the preferences for the different perspectives.

1. *Changing Font Face/Style/Color:*
   General->Appearance->Colors and Fonts
   C/C++ ->Editor->C-Build Console Text Font (double-click to modify)

2. *Changing C/C++ Editor Preferences*
   C/C++->Editor – Appearance/Syntax Tabs