Module 2: Introduction to the Eclipse IDE

هزيلة

Objective
- Gain an understanding of how to use Eclipse to develop applications

Contents
- Brief introduction to the Eclipse IDE
- Create a simple application
- Run and debug simple application
Launching Eclipse

- For the tutorial, everyone will run Eclipse from an X-Windows terminal
- Once the X-Windows terminal is started, connect to the parallel machine
- Log on to the parallel machine using the supplied username
- Launch Eclipse on the parallel machine
  - Eclipse will be displayed locally on your laptop
Starting X-Windows

- Start the X-Windows Application
  - Linux
    - Open a terminal window
  - MacOS X
    - Double-click /Applications/Utilities/X11
  - Windows (cygwin/X)
    - Double-click cygwin icon on Desktop
    - At the prompt, enter startxwin.sh
  - Windows (other options)
    - Tutorial instructor will guide you
Connecting

Once you have a terminal prompt, enter:

- \texttt{ssh -Y \textit{username}@\textit{hostname}}
- Enter password when prompted
- Your username, hostname, and password can be found on the cover of your tutorial CD

After you have logged on to the parallel machine, enter:

- \texttt{eclipse &}
Specifying A Workspace

- Eclipse prompts for a workspace location at startup time
- The workspace contains all user-defined data
  - Projects and resources such as folders and files

The prompt can be turned off.
Eclipse Welcome Page

- Displayed when Eclipse is run for the first time

Select "Go to the workbench"
The Workbench represents the desktop development environment.

- It contains a set of tools for resource management.
- It provides a common way of navigating through the resources.
- Multiple workbenches can be opened at the same time.
Workbench Components

- A Workbench contains perspectives
- A Perspective contains views and editors
Perspectives

- Perspectives define the layout of views in the Workbench.
- They are task oriented, i.e. they contain specific views for doing certain tasks:
  - There is a Resource Perspective for manipulating resources.
  - Make Perspective for manipulating compiled code (C/C++, Fortran).
  - Debug Perspective for debugging applications.
- You can easily switch between perspectives.
Switching Perspectives

✦ You can switch Perspectives by:
  ✦ Choosing the **Window** ➤ **Open Perspective** menu option
  ✦ Clicking on the **Open Perspective** button
  ✦ Clicking on a perspective shortcut button
Available Perspectives

- By default, certain perspectives are available in the Workbench
- We’ve also installed C/C++ and Fortran perspectives
Customizing Perspectives

- Items such as shortcuts, menu items and views may be customized
  - **Window ▶ Customize Perspective...**
- Rearrange views by dragging
  - Try moving the outline view
- Save changes
  - **Window ▶ Save Perspective As...**
- Close Perspective
  - Right-click on perspective title and select **Close**
- Reset Perspective
  - **Window ▶ Reset Perspective** resets the current perspective to its default layout
Views

✧ The main purpose of a view is:
  ✧ To provide alternative ways of presenting information
  ✧ For navigation
  ✧ For editing and modifying information

✧ Views can have their own menus and toolbars
  ✧ Items available in menus and toolbars are available only in that view
  ✧ Menu actions only apply to the view
Stacked Views

- Stacked views appear as tabs
- Selecting a tab brings that view to the foreground
Projects View

- Represents user’s data
- It is a set of user defined resources
  - Files
  - Folders
  - Projects
    - Collections of files and folders
    - Plus meta-data
- Resources are visible in the Navigator View
Opening a New View

🔹 To open a view:
🔹 Choose **Window ➤ Show View ➤ Other…**
🔹 The **Show View** dialog comes up
🔹 Select the view to be shown
🔹 Select **OK**
Fast Views (1)

- Hidden views that can be quickly opened and closed
  - They take up space in the Workbench
- Fast views can be created by:
  - Dragging an open view to the shortcut bar
  - Selecting **Fast View** from the view’s menu
- A Fast View is activated by clicking on its **Fast View** button
Fast Views (2)

- Clicking on the Fast View opens the view in the current perspective
- Clicking outside of the view makes it hidden again
- Turn off the Fast View by selecting Fast View from the view’s menu again
Editors

✦ An editor for a resource opens when you double-click on a resource
  ✦ Editor type depends on the type of the resource, for example .c files are opened with the C/C++ editor
  ✦ When an editor opens on a resource, it stays open across different perspectives
  ✦ An active editor contains menus and toolbars specific to that editor
  ✦ When you change a resource, an asterisk on the editor’s title bar indicates unsaved changes
Preferences

Preferences provide a way for you to customize your Workbench
- By selecting **Window ➤ Preferences**...
- For example:
  - Use Emacs bindings
  - Modify editor folding defaults
    - E.g., fold all macro definitions
  - Associate file types with file extensions
    - E.g., *.f03 with the Fortran editor
  - Toggle automatic builds
  - Change key sequence shortcuts
    - E.g., Ctrl+/ for Comment
Help

- Access help
  - Help ➤ Help Contents
  - Search
  - Dynamic Help...
- What’s there...
- Context sensitive help...

Using the Eclipse help system

Browse topics in the Contents frame (��) on the left. Click on a topic to have it displayed. Use the Back and Forward buttons to navigate within the history of viewed topics.

Searching

To quickly locate topics on a particular subject in the documentation, enter a query in the Search field. Use the Search frame (状) to display the Search view. You can narrow the scope of your search by selecting only the sections you are interested in.
A Simple Application

- Create a Project
  - Managed Make C Project
- Add files
  - Source files (ending in .c)
  - A makefile is automatically created
- Build application
  - Done automatically
- Debug application
  - Create a Run Configuration
CDT Projects

- A Project contains the resources of an application
- CDT (C/C++ Development Tools) project types:
  - Managed Make
  - Standard Make
- Resources are visible in Navigator or C/C++ Projects View
- Project Type is very important
  - Selects project builder (linker)
  - C++, Fortran, or C
Creating a Managed Make Project

- Create a project (in C/C++ Perspective)
  - File►New►Managed Make C Project
  - Or select New Project button
- Give it a name: Integrator
  - Next>
- Select Project Type
  - Next>
- On Indexer tab, select Full C/C++ Indexer
- Select Finish
Add Resources

- Import existing files from file system
  - Right-click on project, select **Import**...
  - Under **General**, select **File System** then **Next**
  - Input **From directory:** using **Browse**...
  - Select **code** folder and then **OK**
  - Check **linear_function.c** and **integrator.c**
  - Select **Finish**

- Can also create new source files
  - **File** ▶ **New** ▶ **Source File**
Fix Error in File

- Project fails to build
  - Note red icon on filename
- Click on **Problems** View tab
- Fix error in **linear_function.c**
  - Double-click on the file in the **C/C++ Projects** view to open an editor
- Save file; project will automatically rebuild when file is saved
  - **File** ► **Save** (or Ctrl-S)
- Look at console view to see build progress
  - There is still another error
To fix the next error, add the GNU Scientific Library to the build process:

- Right-click on Project and select the **Properties** menu item
- Select the **C/C++ Build** item
- Select **GCC C Linker** > **Libraries** from the **Tool Settings** tab
  - Click on the ‘+’ icon next to **Libraries (-l)** to add the library
  - Enter ‘gsl’ in the dialog box and select **OK**
- Select **OK** to close the **Project Properties**
Launch Configuration

- A Launch Configuration is needed to run or debug an application
- To create a launch configuration:
  - Select **Run ➤ Debug**…
  - Or click arrow next to debug button, then **Debug**…
  - Select **C/C++ Local Application** and select the **New** button
  - You may have to select the **Search Project** button for the **C/C++ Application** binary
  - Select **gdb/mi** from the **Debugger tab**
  - You may add command line arguments using the **Arguments tab**
  - Select **Debug**
Debugging (1)

- Select **Yes** to confirm switching to the Debug Perspective after creating the launch configuration.
- Set a breakpoint by double-clicking on the left vertical bar in the editor (at `sum = 0.0;` line).
- To continue running, click on **Resume** button.
- Click on **Step Over** button until line with `getRandomNumber()`.
- Click on **Step Into** button to enter `getRandomNumber()`.
Debugging (2)

- Examine variables in Variables View
  - Clicking on a variable will display its value
- Look at the result value in `getRandomNumber()`
- Click on the **Step Return** button
- Finish by clicking on the **Resume** or **Terminate** button
Play (if time)

- Add `printf()` to program
- Change variable name