parallel tools platform
http://eclipse.org/ptp

A New and Improved Eclipse Parallel Tools Platform: Advancing the Development of Scientific Applications

Greg Watson, IBM
g.watson@computer.org

Jay Alameda, NCSA
jalameda@ncsa.uiuc.edu

Galen Arnold, NCSA
arnoldg@ncsa.uiuc.edu

Beth Tibbitts, IBM
tibbitts@us.ibm.com

Jeff Overbey, UIUC
overbey2@illinois.edu

July 18, 2011

Portions of this material are supported by or based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under its Agreement No. HR0011-07-9-0002, the United States Department of Energy under Contract No. DE-FG02-06ER25752, the Blue Waters sustained petascale computing project, which is supported by the National Science Foundation under award number OCI 07-25070, and the SI2-SSI Productive and Accessible Development Workbench for HPC Applications, which is supported by the National Science Foundation under award number OCI 1047956

TeraGrid '11

Tutorial Outline

Time (Tentative!)	Module	Topics	Presenter
8:00-8:30	1. Overview of Eclipse and PTP, Installation check	✦ Introduction to Eclipse/PTP; demo	Greg
8:30-9:30	3. CDT: Working with C/C++ Remote Projects	✦ Eclipse basics; Creating a new project ✦ Building and launching remotely	Beth
9:30-10:00	4. Working with MPI	✦ Makefiles, PLDT MPI tools ✦ Resource Managers ✦ Launching a parallel application	Jay
10:00-10:30	BREAK		
10:30-11:00	4. Working with MPI	✦ Makefiles, PLDT MPI tools ✦ Resource Managers ✦ Launching a parallel application	Jay
11:00-12:00	5. Debugging	✦ Debugging an MPI program	Greg
12:00 – 1:00	Lunch		
1:00-2:15	6. Fortran; Refactoring	✦ Photran overview; comparison w/ CDT ✦ Refactoring support	Jeff
2:15-2:30	BREAK		
2:30-4:30	7. Advanced Features: Performance Tuning & Analysis Tools	✦ PLDT (MPI, OpenMP, UPC tools) (20 min) ✦ TAU, ETFw (20) ✦ GEM (20) ✦ Linux Tools (gprof, gcov) (20 min) ✦ Configuring Resource Managers (20 min)	Beth Suzanne Alan Galen Jay
4:30- 5:00	8. Other Tools, Wrapup	✦ NCSA HPC Workbench, Other Tools, website, mailing lists, future features	Jay/Beth

Final Slides, Installation Instructions

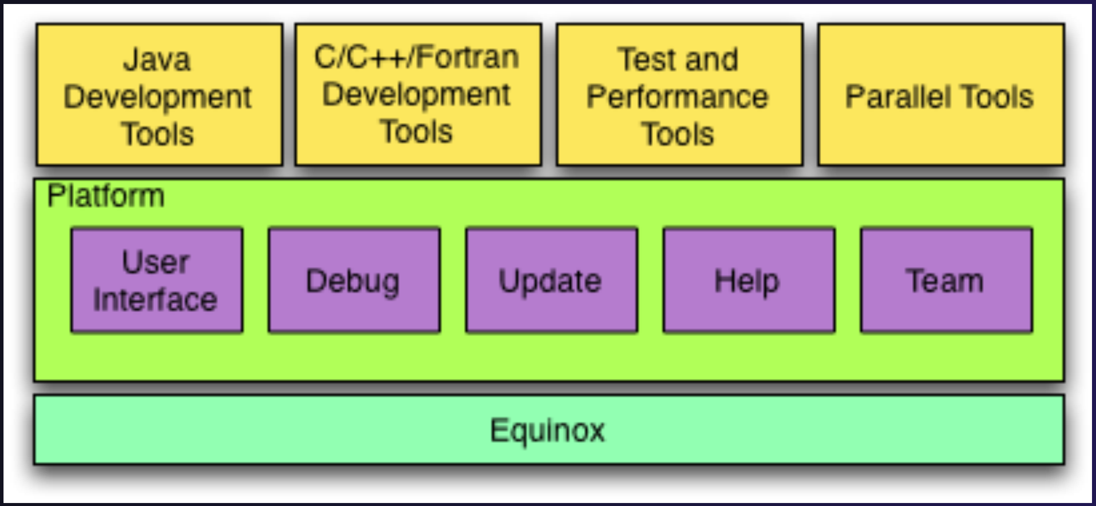
- Please go to http://wiki.eclipse.org/PTP/tutorials/TG11 for slides and installation instructions

Module 1: Introduction

- Objective
  - To introduce the Eclipse platform and PTP
- Contents
  - What is Eclipse?
  - What is PTP?

What is Eclipse?

- A vendor-neutral open-source workbench for multi-language development
- A extensible platform for tool integration
- Plug-in based framework to create, integrate and utilize software tools



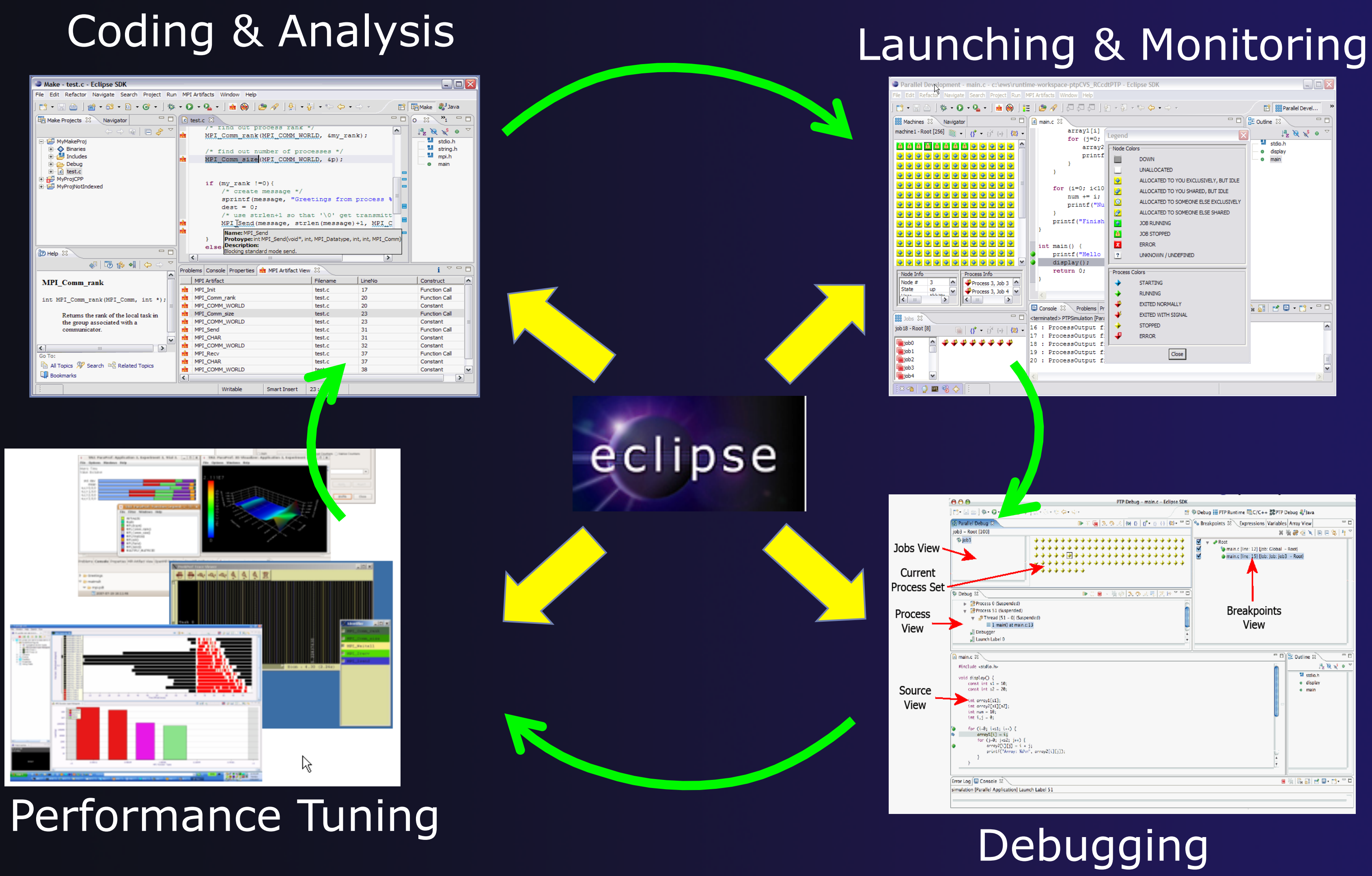
Eclipse Platform

- Core frameworks and services with which all plug-in extensions are created
- Represents the common facilities required by most tool builders:
  - Workbench user interface
  - Project model for resource management
  - Portable user interface libraries (SWT and JFace)
  - Automatic resource delta management for incremental compilers and builders
  - Language-independent debug infrastructure
  - Distributed multi-user versioned resource management (CVS supported in base install)
  - Dynamic update/install service

Plug-ins

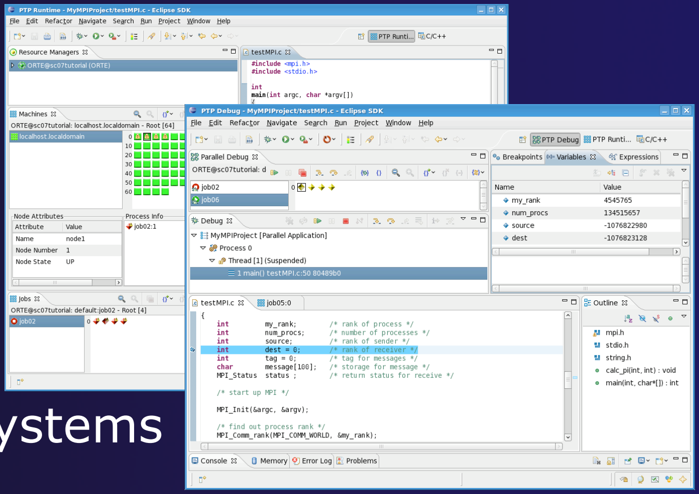
- Java Development Tools (JDT)
- Plug-in Development Environment (PDE)
- C/C++ Development Tools (CDT)
- Parallel Tools Platform (PTP)
- Fortran Development Tools (Photran)
- Test and Performance Tools Platform (TPTP)
- Business Intelligence and Reporting Tools (BIRT)
- Web Tools Platform (WTP)
- Data Tools Platform (DTP)
- Device Software Development Platform (DSDP)
- Many more…

Eclipse Parallel Tools Platform (PTP)



Parallel Tools Platform (PTP)

- The Parallel Tools Platform aims to provide a highly integrated environment specifically designed for parallel application development
- Features include:
  - An integrated development environment (IDE) that supports a wide range of parallel architectures and runtime systems
  - A scalable parallel debugger
  - Parallel programming tools (MPI, OpenMP, UPC, etc.)
  - Support for the integration of parallel tools
  - An environment that simplifies the end-user interaction with parallel systems
- http://www.eclipse.org/ptp



```
#include <mpi.h>
#include <stdio.h>

int
main(int argc, char *argv[])
```

Name	Value
my_rank	4545765
num_procs	134515657
source	-1076822980
dest	-1076823128

MyMPIProject [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() testMPI.c:50 80489b0

```
{
    int         my_rank;        /* rank of process */
    int         num_procs;      /* number of processes */
    int         source;         /* rank of sender */
    int         dest = 0;       /* rank of receiver */
    int         tag = 0;        /* tag for messages */
    char        message[100];   /* storage for message */
    MPI_Status  status ;        /* return status for receive */

    /* start up MPI */

    MPI_Init(&argc, &argv);

    /* find out process rank */
    MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);
```

PTP Features Demo…

- Creating a project from existing source code – importing into Eclipse and PTP
- Content assist, searching, include browser
- Building the project
- Launching an MPI program
- Debugging an MPI program

Module 3: Working with C/C++

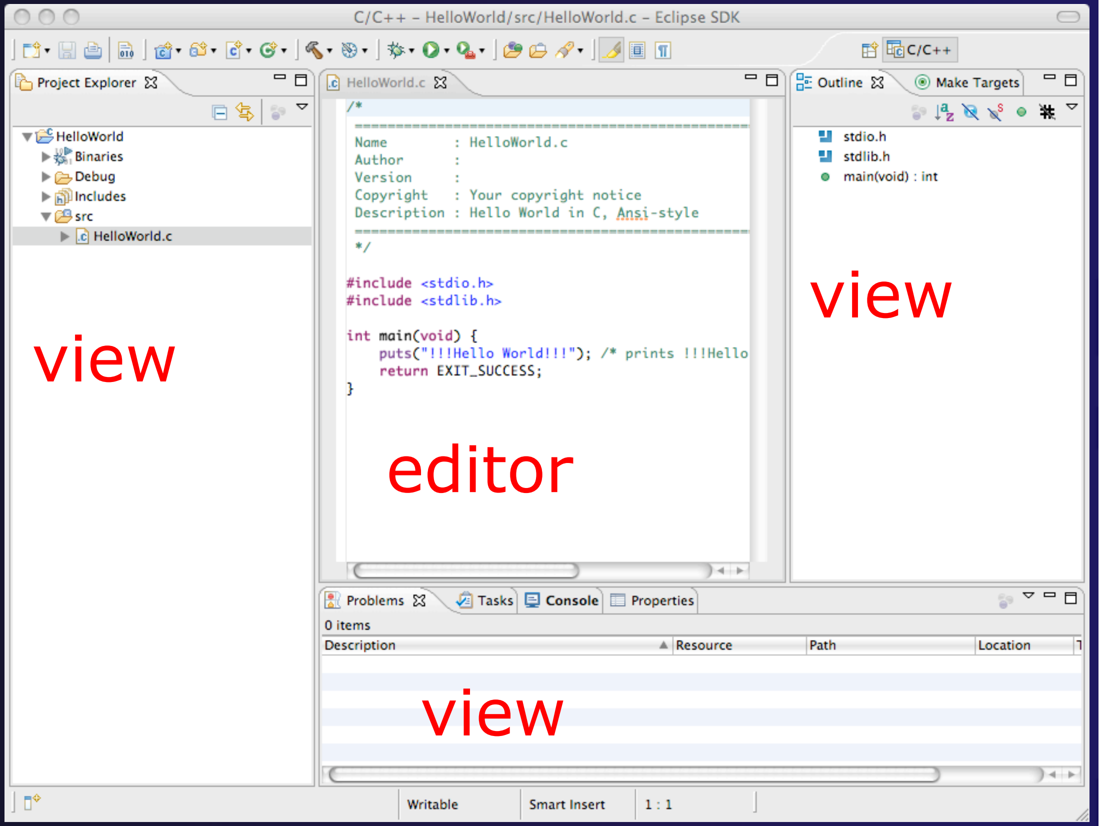
- Objective
  - Learn basic Eclipse concepts: Perspectives, Views, …
  - Learn how to use Eclipse to manage a remote project
  - Learn how to use Eclipse to develop C programs
  - Learn how to launch and run a remote C program
- Contents
  - Brief introduction to the C/C++ Development Tools (CDT)
  - Create a simple remote application
  - Learn to launch a remote C application

Login Information

- The hands on portion of this module will be done on a remote system at SDSC, thank you to SDSC!
  - Lincoln.ncsa.uiuc.edu
  - Train41-60
  - TG11tr8L!
- See the following URL for more information on the system
  - http://www.sdsc.edu/us/resources/trestles/
  - Each student will be assigned an ID and password at the start of the tutorial
- Please use only this ID
  - We are also working to make this work with Ranger and Kraken, this work is not complete…

Eclipse Basics

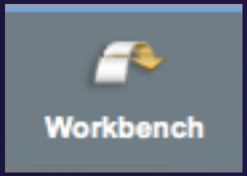
- A *workbench* contains the menus, toolbars, editors and views that make up the main Eclipse window
- The workbench represents the desktop development environment
  - Contains a set of tools for resource mgmt
  - Provides a common way of navigating through the resources
- Multiple workbenches can be opened at the same time
- Only one workbench can be open on a *workspace* at a time



```
/*
 ===========================================================
 Name        : HelloWorld.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ===========================================================
 */

#include <stdio.h>
#include <stdlib.h>

int main(void) {
	puts("!!!Hello World!!!"); /* prints !!!Hello
	return EXIT_SUCCESS;
}
```

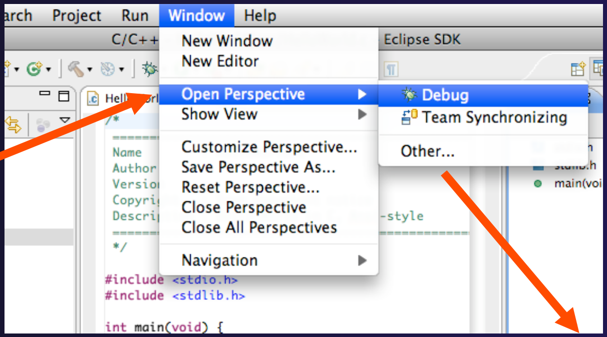
Perspectives

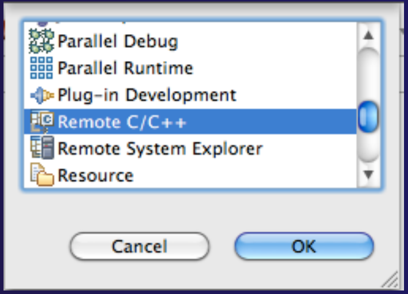
- Perspectives define the layout of views and editors in the workbench
- They are *task oriented*, i.e. they contain specific views for doing certain tasks:
  - There is a **Resource Perspective** for manipulating resources
  - **C/C++ Perspective** for manipulating compiled code
  - **Debug Perspective** for debugging applications
- You can easily switch between perspectives
- If you are on the Welcome screen now, select “Go to Workbench” now

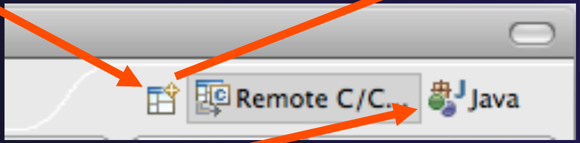


Switching Perspectives

- Three ways of changing perspectives
  - Choose the **Window>Open Perspective** menu option
  - Then choose **Other…**
  - Click on the **Open Perspective** button in the upper right corner of screen
  - Click on a perspective shortcut button
- Switch perspective on next slide…

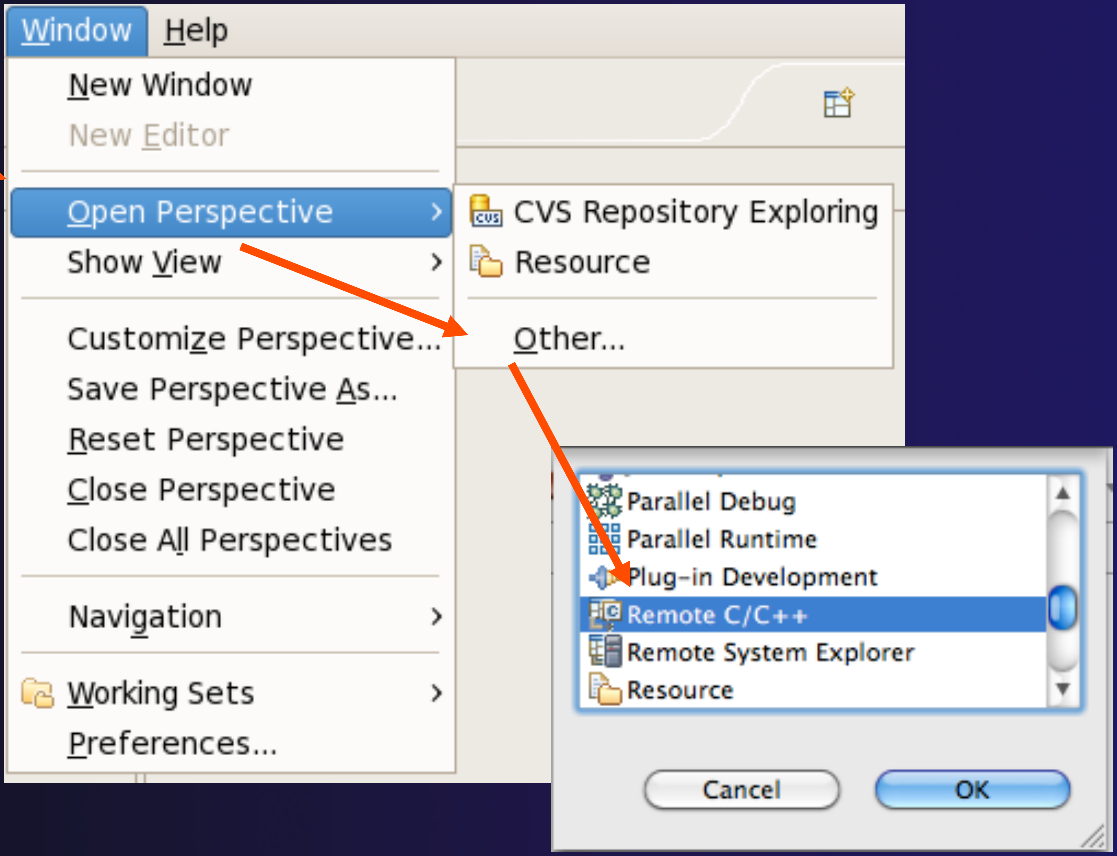




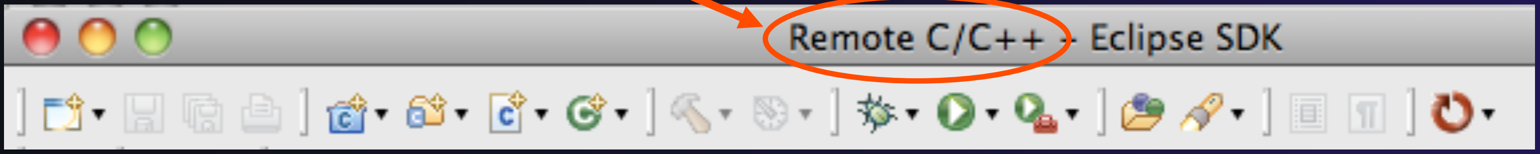


Switch to Remote C/C++ Perspective

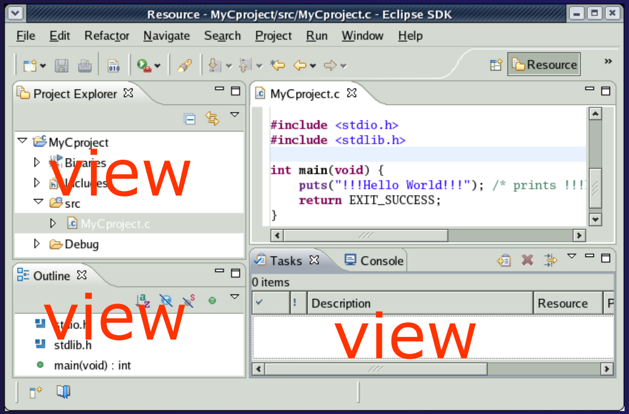
- Select **Window>Open Perspective**
- Then choose **Other…**
- Only needed if you're not already in the perspective



- What Perspective am in in?
  See title Bar

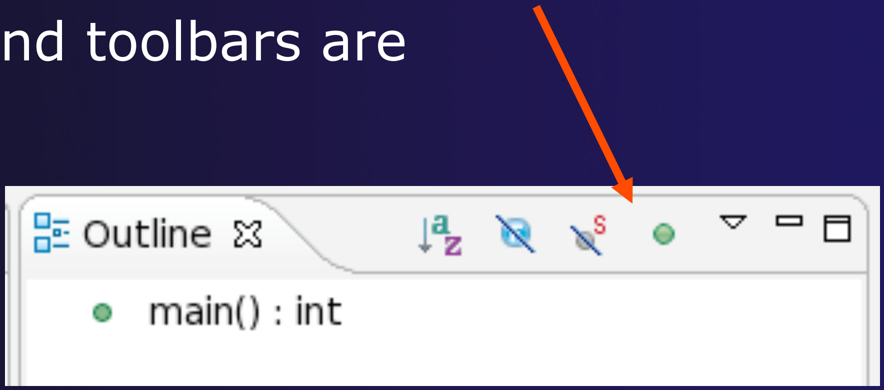


Views



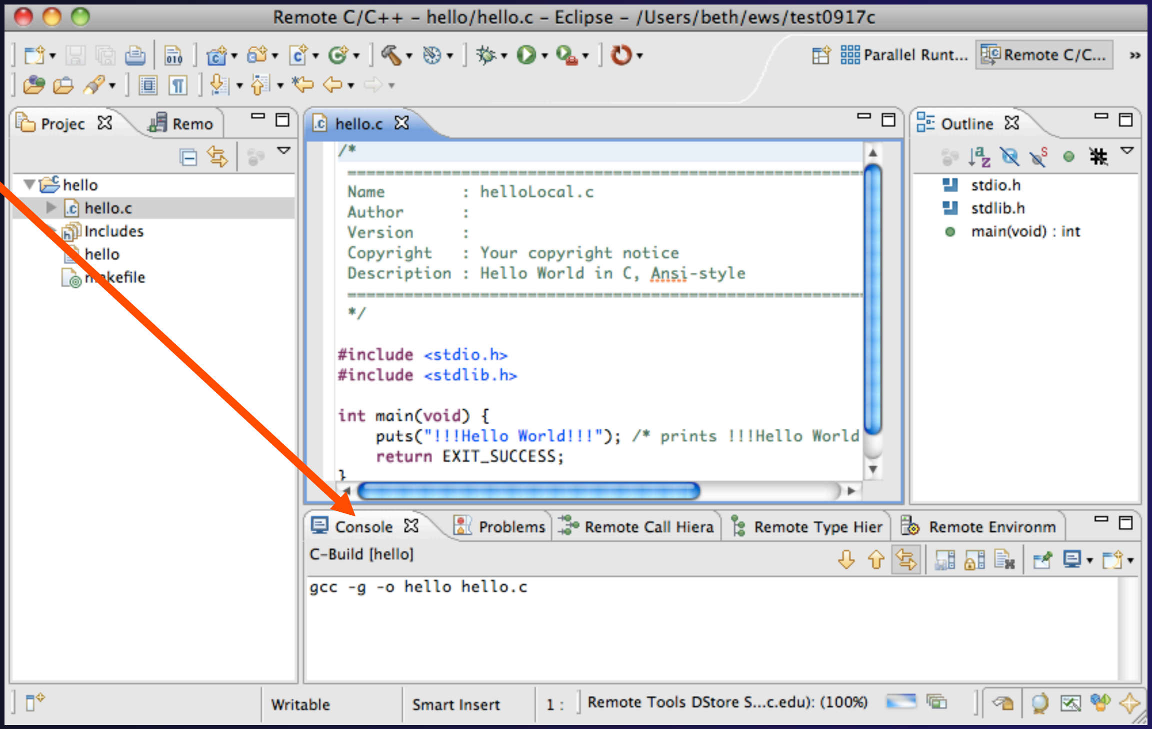
```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!
    return EXIT_SUCCESS;
}
```

- The workbench window is divided up into 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
- Views can be resized



Stacked Views

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

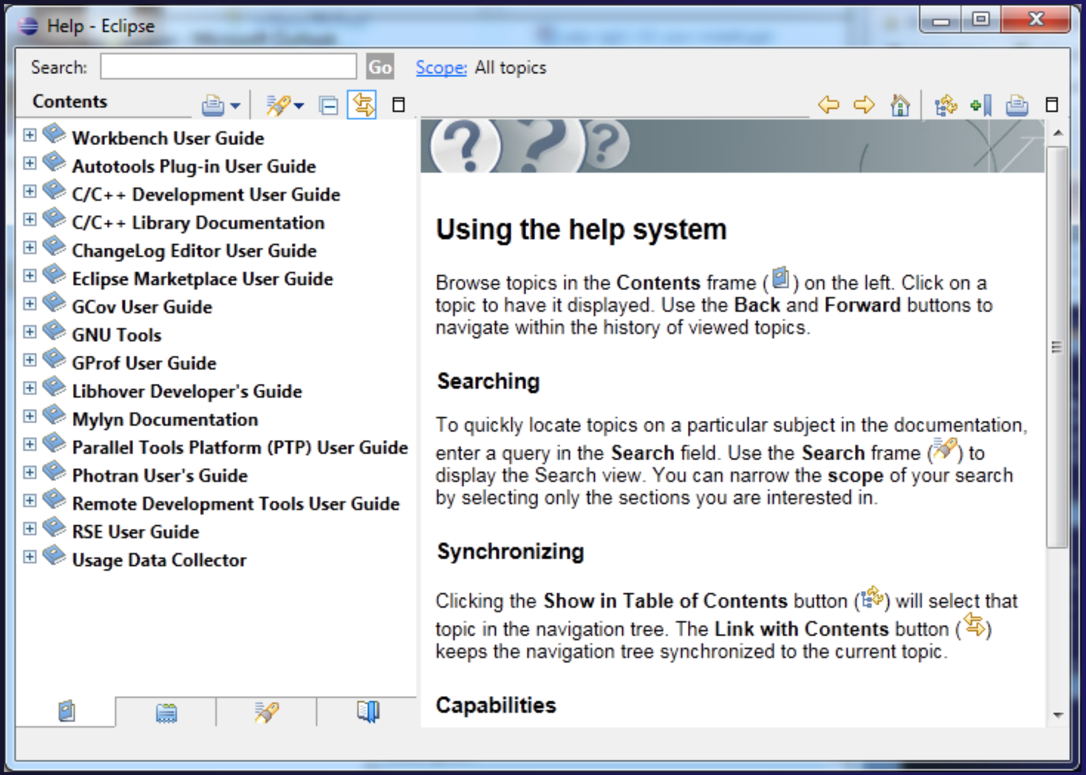


```
/*
 ============================================================================
 Name        : helloLocal.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================================
 */

#include <stdio.h>
#include <stdlib.h>

int main(void) {
	puts("!!!Hello World!!!"); /* prints !!!Hello World
	return EXIT_SUCCESS;
}
```

```
gcc -g -o hello hello.c
```

Help

- To access help
  - **Help>Help Contents**
  - **Help>Search**
  - **Help>Dynamic Help**
- **Help Contents** provides detailed help on different Eclipse features *in a browser*
- **Search** allows you to search for help locally, or using Google or the Eclipse web site
- **Dynamic Help** shows help related to the current context (perspective, view, etc.)



- Workbench User Guide
- Autotools Plug-in User Guide
- C/C++ Development User Guide
- C/C++ Library Documentation
- ChangeLog Editor User Guide
- Eclipse Marketplace User Guide
- GCov User Guide
- GNU Tools
- GProf User Guide
- Libhover Developer's Guide
- Mylyn Documentation
- Parallel Tools Platform (PTP) User Guide
- Photran User's Guide
- Remote Development Tools User Guide
- RSE User Guide
- Usage Data Collector

Using the 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

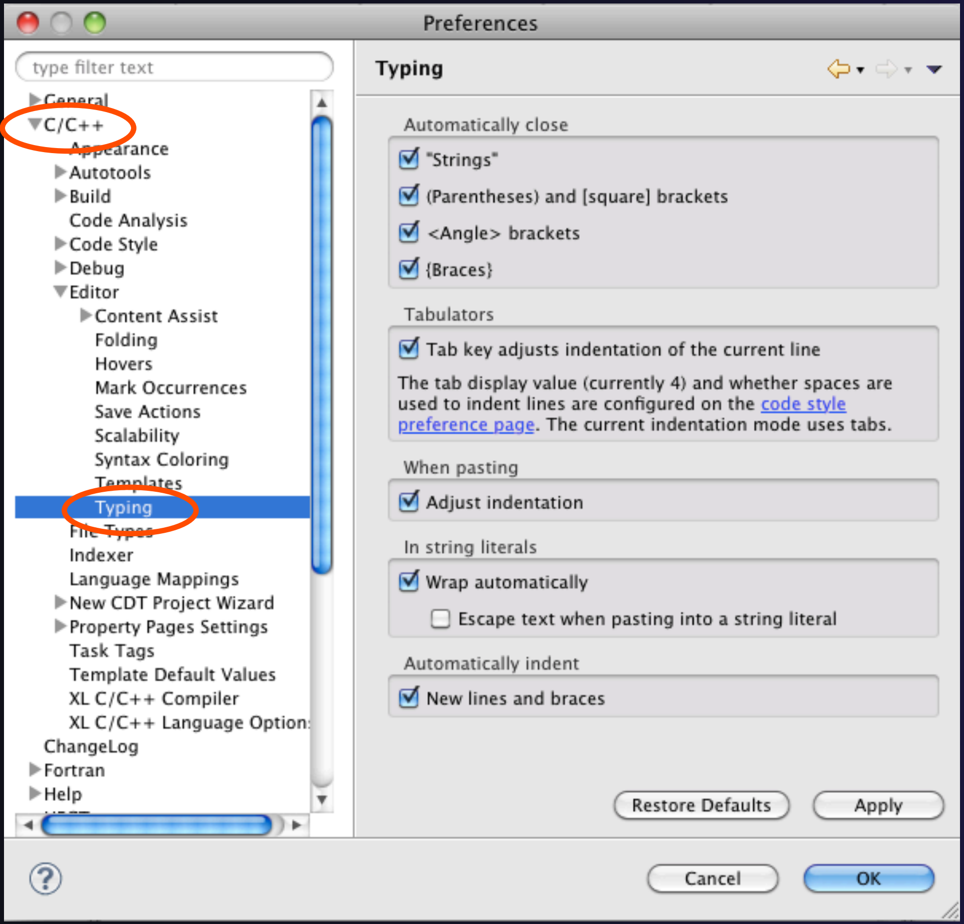
To quickly locate topics on a particular subject in the documentation, enter a query in the **Search** field. Use the **Search** frame ( ) to display the Search view. You can narrow the **scope** of your search by selecting only the sections you are interested in.

Synchronizing

Clicking the **Show in Table of Contents** button ( ) will select that topic in the navigation tree. The **Link with Contents** button ( ) keeps the navigation tree synchronized to the current topic.

Capabilities

Preferences



Typing

Automatically close

- [✓] "Strings"
- [✓] (Parentheses) and [square] brackets
- [✓] <Angle> brackets
- [✓] {Braces}

Tabulators

- [✓] Tab key adjusts indentation of the current line

The tab display value (currently 4) and whether spaces are used to indent lines are configured on the code style preference page. The current indentation mode uses tabs.

When pasting

- [✓] Adjust indentation

In string literals

- [✓] Wrap automatically
- [ ] Escape text when pasting into a string literal

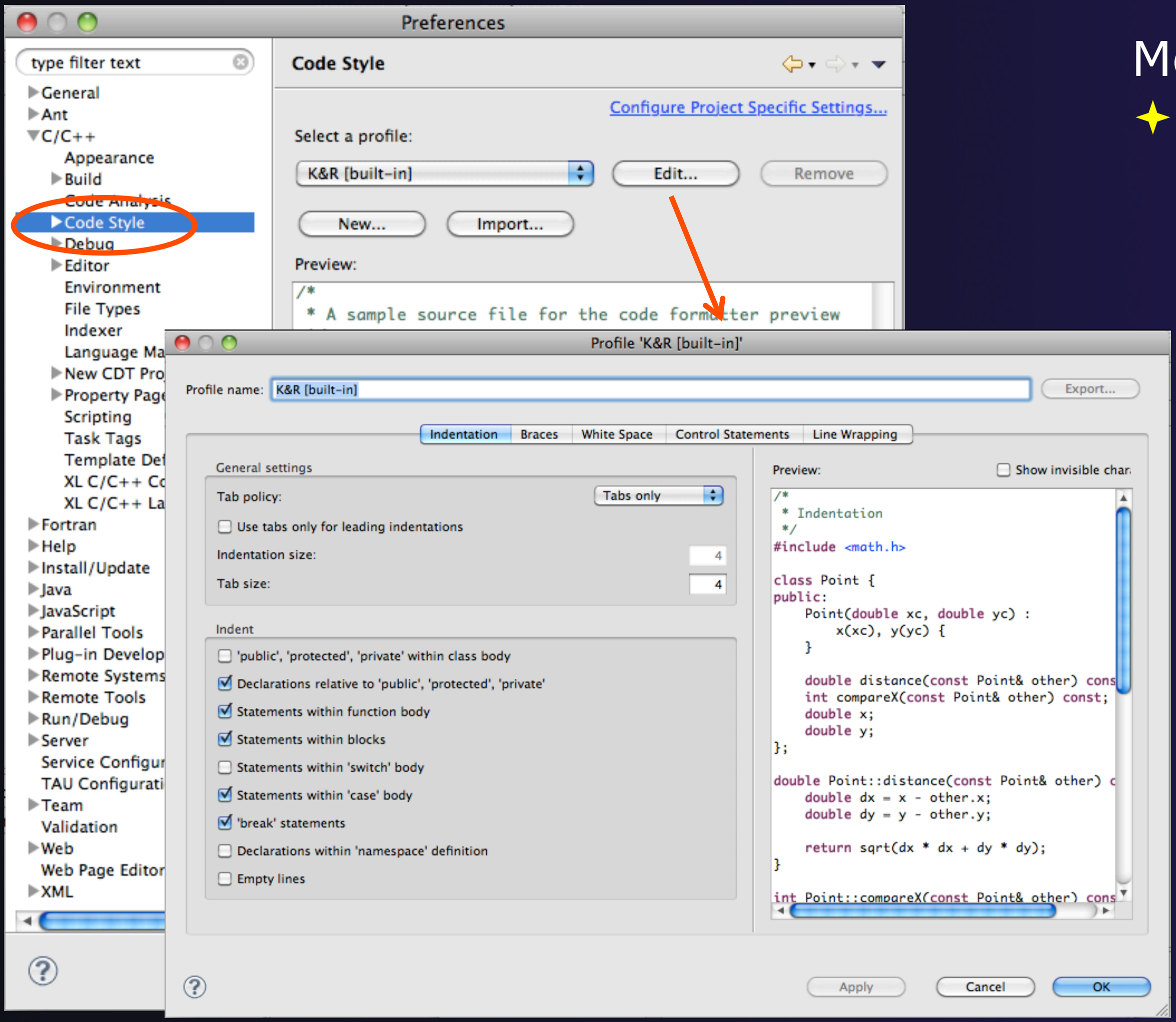
Automatically indent

- [✓] New lines and braces

- Eclipse Preferences allow customization of almost everything
- To open use
  - Mac: **Eclipse>Preferences…**
  - Others: **Window>Preferences…**

- The C/C++ preferences allow many options to be altered
- In this example you can adjust what happens in the editor as you type.

Preferences (2)



Code Style

Select a profile: K&R [built-in]

Preview:

```
/*
 * A sample source file for the code formatter preview
```

Profile name: K&R [built-in]

General settings

Tab policy: Tabs only

- [ ] Use tabs only for leading indentations

Indentation size: 4

Tab size: 4

Indent

- [ ] 'public', 'protected', 'private' within class body
- [x] Declarations relative to 'public', 'protected', 'private'
- [x] Statements within function body
- [x] Statements within blocks
- [ ] Statements within 'switch' body
- [x] Statements within 'case' body
- [x] 'break' statements
- [ ] Declarations within 'namespace' definition
- [ ] Empty lines

Preview:

```
/*
 * Indentation
 */
#include <math.h>

class Point {
public:
    Point(double xc, double yc) :
        x(xc), y(yc) {
    }

    double distance(const Point& other) cons
    int compareX(const Point& other) const;
    double x;
    double y;
};

double Point::distance(const Point& other) c
    double dx = x - other.x;
    double dy = y - other.y;

    return sqrt(dx * dx + dy * dy);
}

int Point::compareX(const Point& other) cons
```

More C/C++ preferences:

- In this example the Code Style preferences are shown
  - These allow code to be automatically formatted in different ways

Types of C/C++ Projects

- C/C++ Projects can be
  - Local – source is located on local machine, builds happen locally
  - Remote – source is either located on remote machine, or synchronized with remote machine; builds take place on remote machine
  - Makefile-based – project contains its own makefile (or makefiles) for building the application
  - Managed– Eclipse manages the build process, no makefile required
- Parallel programs can be run on the local machine or on a remote system
  - MPI needs to be installed
  - An application built locally probably can't be run on a remote machine unless their architectures are the same
- We will show you how to create, build and run the program on a remote machine
  - We will create a remote Makefile project

Remote Projects

"Traditional" Remote Projects

- Source is located on remote machine
- Eclipse is installed on the local machine and can be used for:
  - Editing
  - Building
  - Running
  - Debugging
- Source indexing is performed on remote machine
  - Enables call hierarchy, type hierarchy, include browser, search, outline view, and more…
- Builds are performed on remote machine
  - Supports both managed and makefile projects
- Application is run and debugged remotely using the PTP resource managers

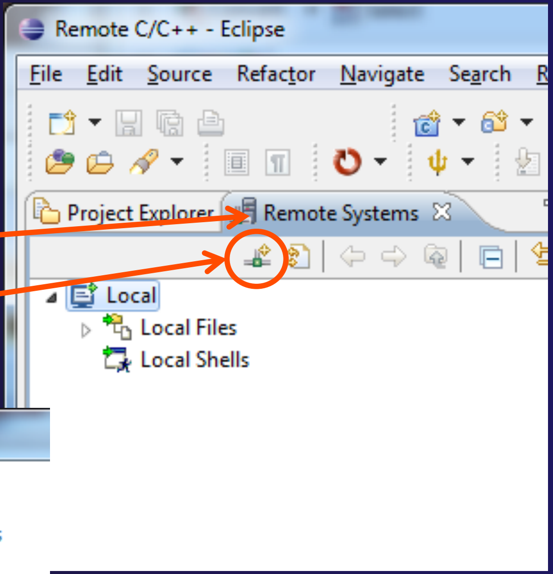
Synchronized Projects

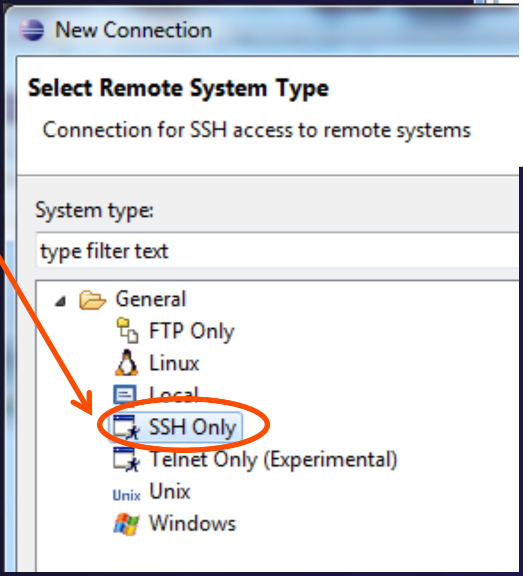
- Source is located on *both* the local system and on a remote target system. The two copies are kept in sync by Eclipse.
- Eclipse is installed on the local machine and can be used for:
  - Editing
  - Building
  - Running
  - Debugging
  - *Development can continue "off-line"*
- Source indexing is performed on *local* machine
  - Enables call hierarchy, type hierarchy, include browser, search, outline view, and more…
- Builds are performed on *one or more* remote machines
  - Supports both managed and makefile projects
- Application is run and debugged remotely using the PTP resource managers

Traditional Remote Projects

Preparation steps:

- We will set up an SSH terminal to the remote system to copy some files
- Make sure you are in the Remote C/C++ perspective
- Select the Remote Systems view
  - Define a new connection
  - Select "SSH Only"
  - Then **Next**



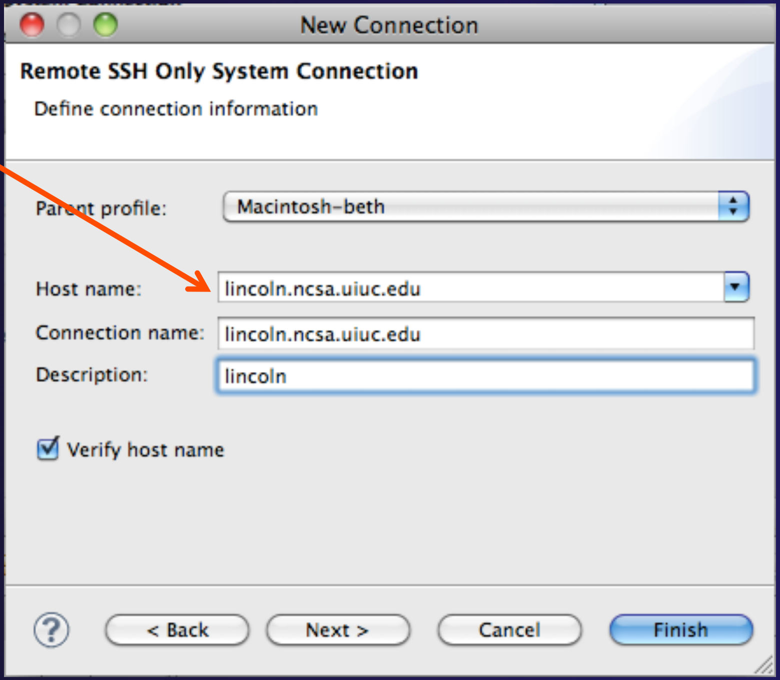


Select Remote System Type

Connection for SSH access to remote systems

Preparation, continued

- Add lincoln's host info
  - Then **Finish**
- Right click on ssh terminals, under lincoln
- Select **Launch Terminal**



New Connection

Remote SSH Only System Connection

Define connection information

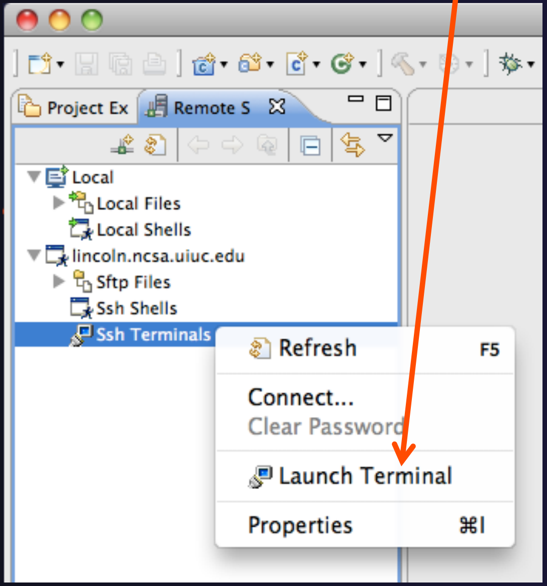
Parent profile: Macintosh-beth

Host name: lincoln.ncsa.uiuc.edu

Connection name: lincoln.ncsa.uiuc.edu

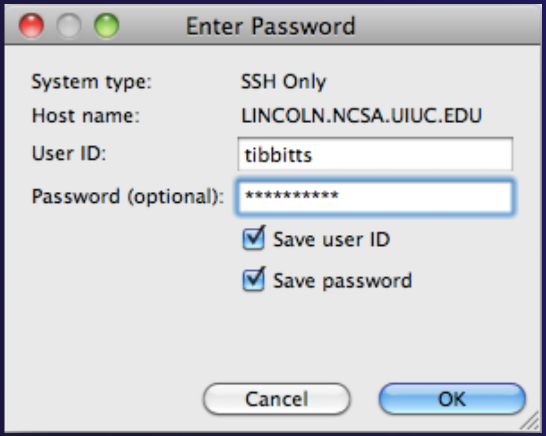
Description: lincoln

[✓] Verify host name



Preparation, continued

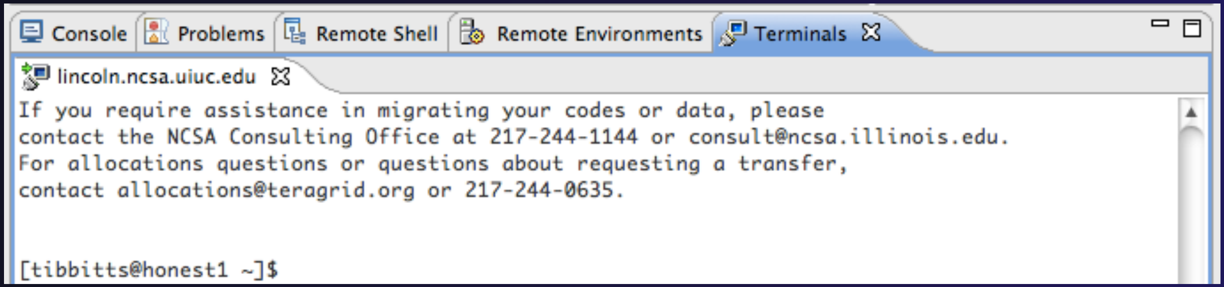
- Add your training account login
- Click through any RSA messages
- And now you have a terminal to lincoln



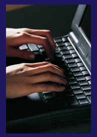
Enter Password

System type:	SSH Only
Host name:	LINCOLN.NCSA.UIUC.EDU
User ID:	tibbitts
Password (optional):	**********

[✓] Save user ID
[✓] Save password



```
If you require assistance in migrating your codes or data, please
contact the NCSA Consulting Office at 217-244-1144 or consult@ncsa.illinois.edu.
For allocations questions or questions about requesting a transfer,
contact allocations@teragrid.org or 217-244-0635.


[tibbitts@honest1 ~]$
```

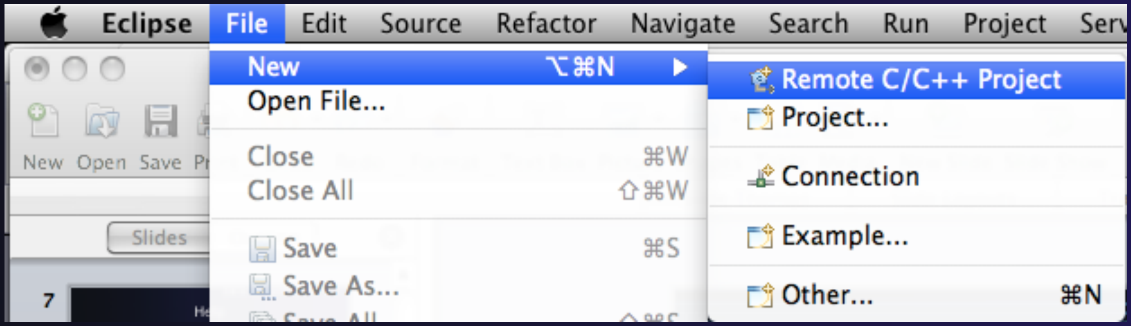


Why did we do this?

- To show you can gain “traditional” access to a remote host through Eclipse
- And to have you stage some directories:
- Issue the following commands in the terminal
  - cp –r ~jalameda/hello_world .
  - cp –r ~jalameda/shallow .
  - cp –r ~jalameda/mpi .
- This will give us some source code to work with

Creating a Remote C/C++ Project

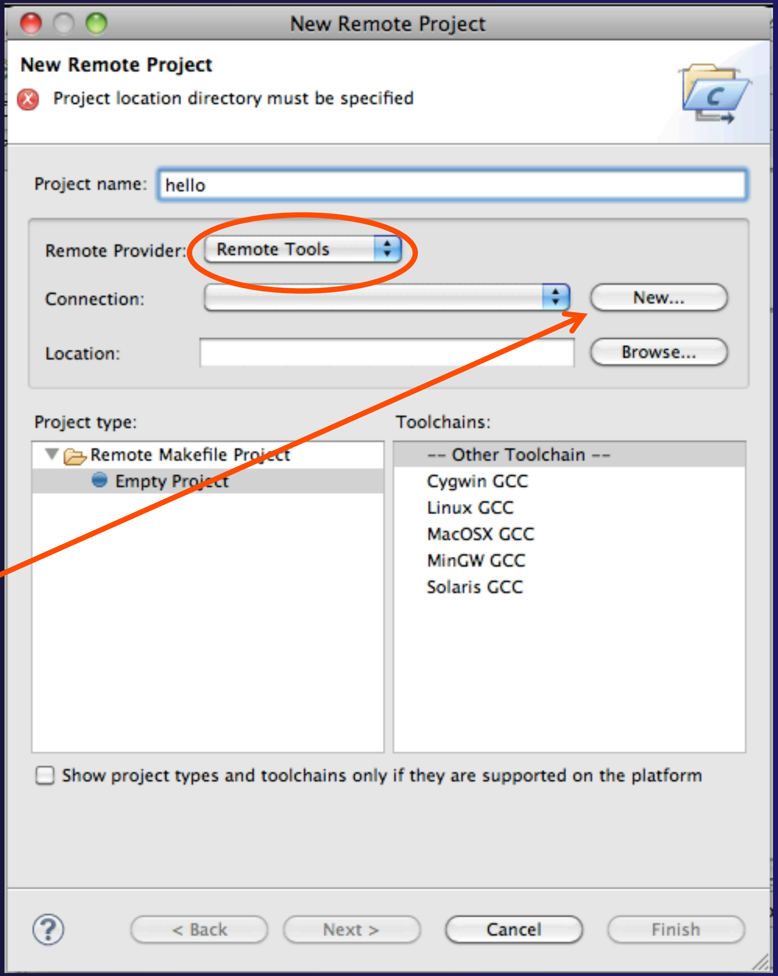
- Use **File>New>Remote C/C++ Project** to open the new project wizard
- The wizard will take you through the steps for creating the project



Don’t see the “Remote C/C++ Project” choice?
Make sure you are in the Remote C/C++ Perspective

New Remote Project Wizard

- Enter project name, e.g. "hello"
- Select a **Remote Provider**
  - Remote providers supply different ways of accessing remote (or local) systems
  - Choose **Remote Tools**
- A **Connection** specifies how to connect to the remote host
  - Click on the **New...** button to create a new connection



New Remote Project

New Remote Project

Project location directory must be specified

Project name: hello

Remote Provider: Remote Tools

Connection:

Location:

Project type:

- Remote Makefile Project
  - Empty Project

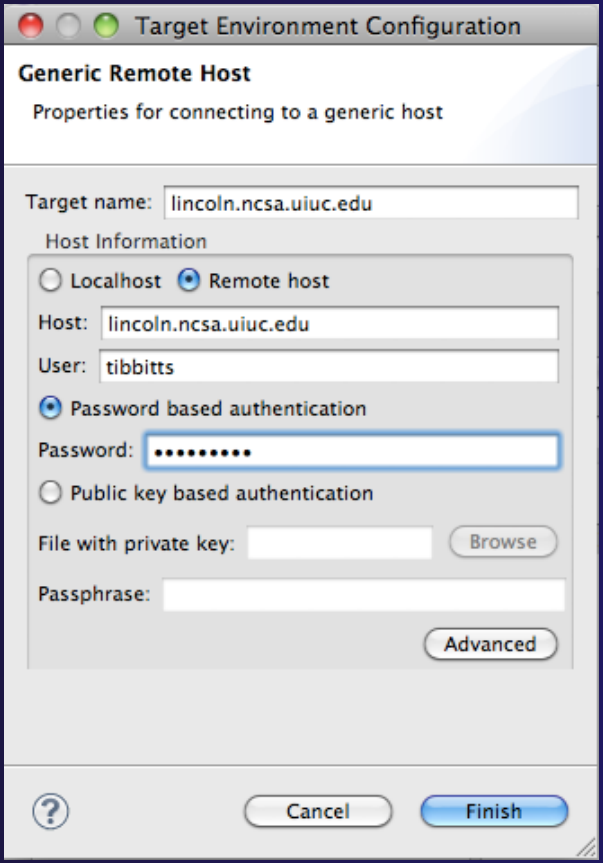
Toolchains:

- -- Other Toolchain --
- Cygwin GCC
- Linux GCC
- MacOSX GCC
- MinGW GCC
- Solaris GCC

Show project types and toolchains only if they are supported on the platform

Remote Host Configuration

- Enter a connection name (can be anything) for the **Target name**
  - Use "lincoln.ncsa.uiuc.edu"
- The host is remote, so the **Remote host** option should be checked
- Enter the host name or IP address of the remote host for the **Host**
  - Use "lincoln.ncsa.uiuc.edu"
- Enter the user name and password supplied at the beginning of the tutorial for the **User** and **Password**
- Note: if your remote machine uses OTP for authentication, *leave the password field blank*
- Click **Finish**


Target Environment Configuration
Generic Remote Host
Properties for connecting to a generic host
Host Information
Browse
Advanced
Cancel
Finish

Target name: lincoln.ncsa.uiuc.edu

( ) Localhost [✓ Remote host]

Host: lincoln.ncsa.uiuc.edu

User: tibbitts

[✓ Password based authentication]

Password: •••••••••

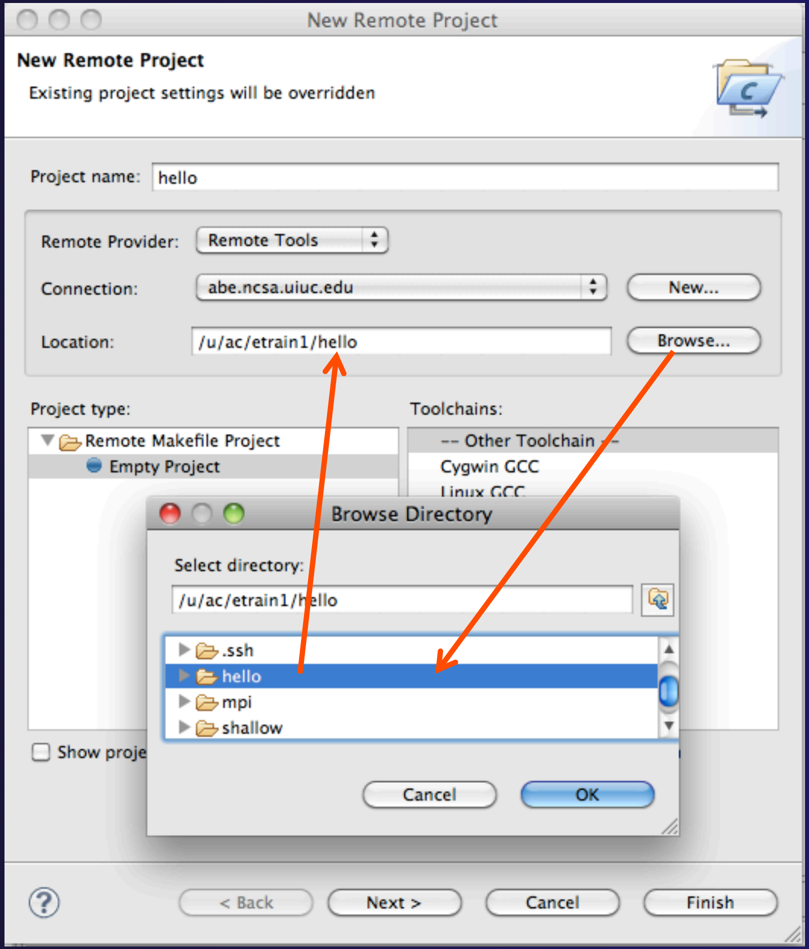
( ) Public key based authentication

File with private key:

Passphrase:

Project Location

- The **Location** is the directory on the remote host containing the source and executable files
- Click on the browse button to browse for folders on the remote machine
  - You should see the folders in your home directory
  - Choose the "hello" directory
- Click **OK**


New Remote Project

Existing project settings will be overridden

Project name: hello

Remote Provider: Remote Tools

Connection: abe.ncsa.uiuc.edu

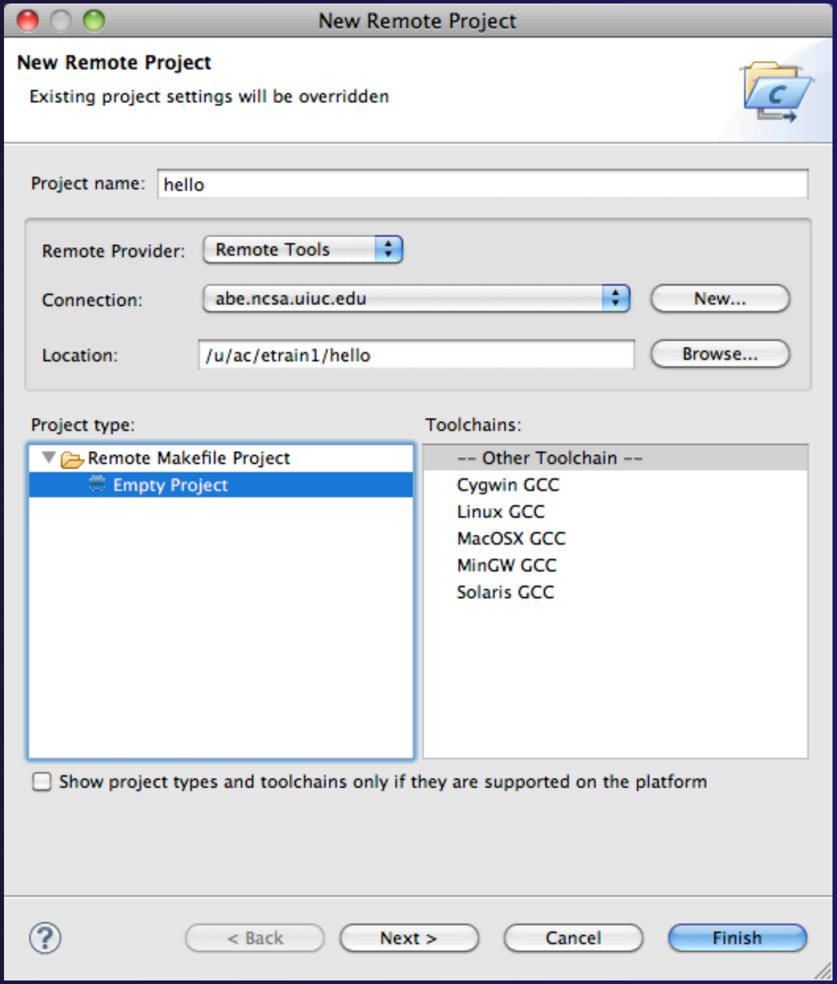
Location: /u/ac/etrain1/hello

Browse Directory

Select directory: /u/ac/etrain1/hello

Project Type

- The **Project type** determines information about the project
  - If the project is managed or unmanaged (described later)
  - The tool chain (compiler, linker, etc.) to use when building
  - If the project creates an executable, static, or shared library
  - Options available depend on whether the project is local or remote
- Under **Remote Makefile Project**, select **Empty Project**
- For **Toolchains**, select **Other Toolchain**
- Click on **Finish** to complete the wizard


New Remote Project

New Remote Project

Existing project settings will be overridden

Project name: hello

Remote Provider: Remote Tools

Connection: abe.ncsa.uiuc.edu

Location: /u/ac/etrain1/hello

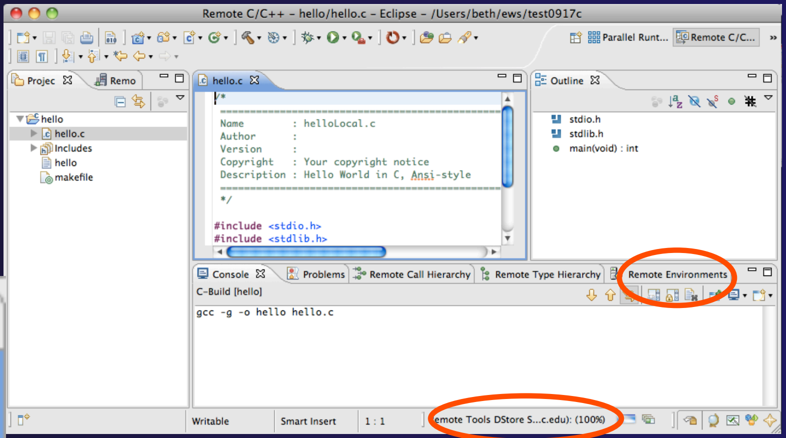
Project type:
- Remote Makefile Project
  - Empty Project

Toolchains:
- -- Other Toolchain --
- Cygwin GCC
- Linux GCC
- MacOSX GCC
- MinGW GCC
- Solaris GCC

☐ Show project types and toolchains only if they are supported on the platform

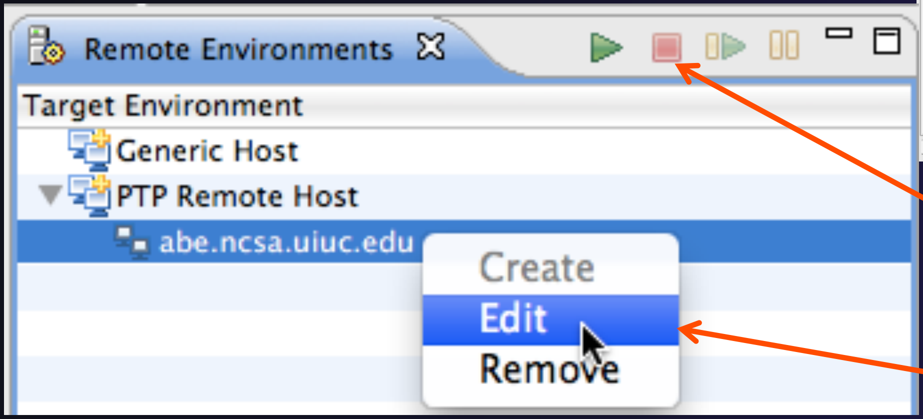
Changing Remote Connection Information

- If you need to change remote connection information (such as username or password), use the **Remote Environments** view


Remote C/C++ – hello/hello.c – Eclipse – /Users/beth/ews/test0917c
Remote Environments
Remote Tools DStore S...c.edu): (100%)

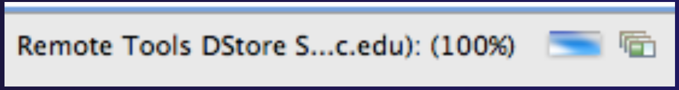
```
/*
 Name        : helloLocal.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 */

#include <stdio.h>
#include <stdlib.h>
```

```
C-Build [hello]
gcc -g -o hello hello.c
```



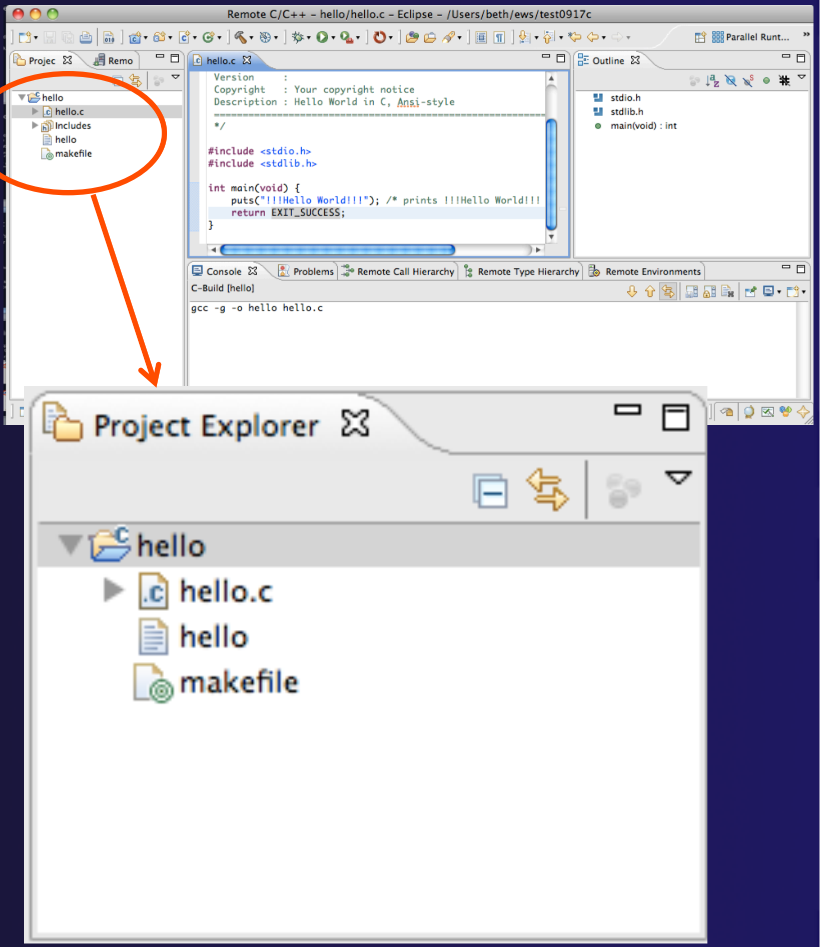
- Stop the remote connection first
- Right-click and select **Edit**

- Note: running server is shown in lower right
  - Opening any remote file restarts it



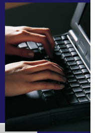
Project Explorer View

- Shows the user's projects
- Each project contains
  - Source files
  - Executable files
  - Folders
  - Metadata (not visible)
- Can have any number of projects
- We only have a single project so far



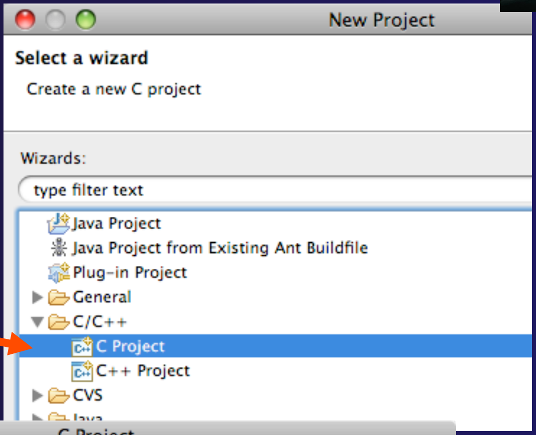
```
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================
 */

#include <stdio.h>
#include <stdlib.h>

int main(void) {
	puts("!!!Hello World!!!"); /* prints !!!Hello World!!!
	return EXIT_SUCCESS;
}
```

```
gcc -g -o hello hello.c
```



New Project Wizard: Create a C Project

- The **New Project Wizard** is used to create a C project
- Enter **Project name**
- Under **Project Types**, select **Makefile project▶Empty Project**
  - Ensures that CDT will use existing makefiles
- Select **Finish**
- When prompted to switch to the **C/C++ Perspective**, select **Yes**



New Project

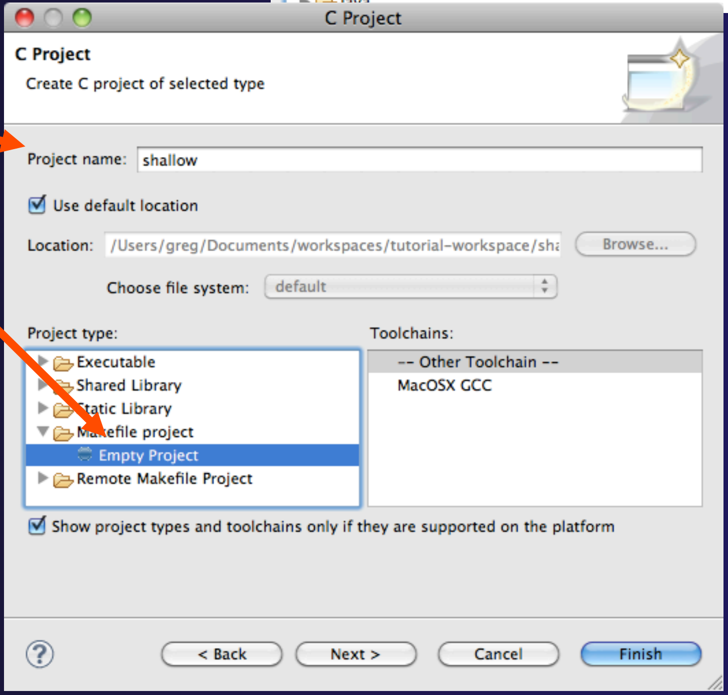
Select a wizard

Create a new C project

Wizards:

type filter text

Java Project
Java Project from Existing Ant Buildfile
Plug-in Project
General
C/C++
C Project
C++ Project
CVS
Java



C Project

C Project

Create C project of selected type

Project name: shallow

☑ Use default location

Location: /Users/greg/Documents/workspaces/tutorial-workspace/sha

Choose file system: default

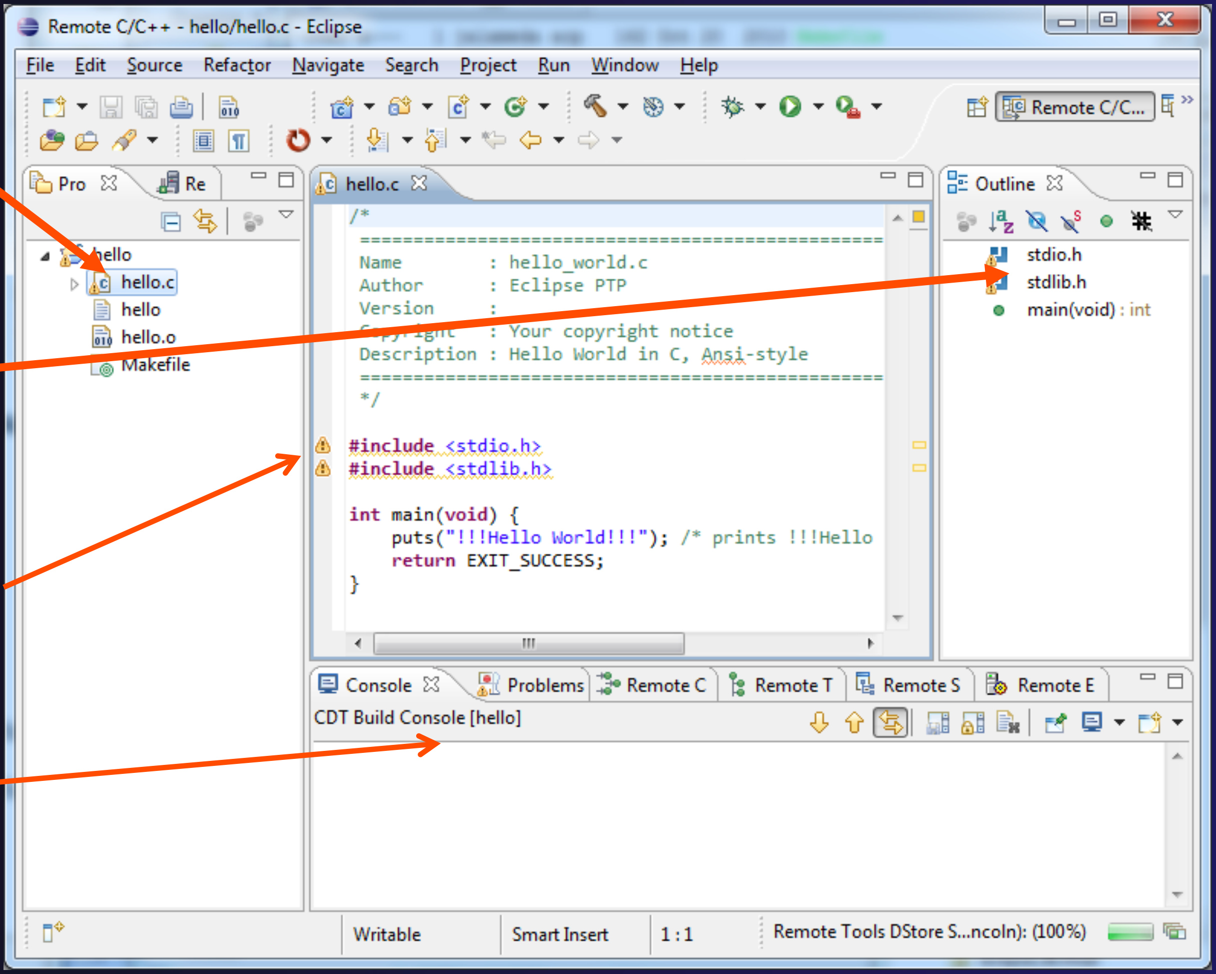
Project type:
- Executable
- Shared Library
- Static Library
- Makefile project
  - Empty Project
- Remote Makefile Project

Toolchains:
- -- Other Toolchain --
- MacOSX GCC

☑ Show project types and toolchains only if they are supported on the platform

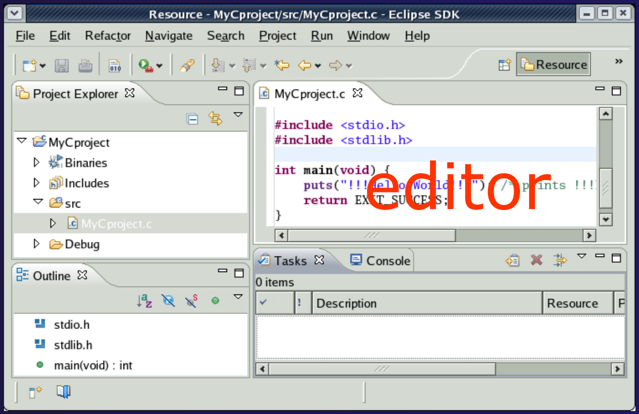
Editor and Outline View

- Double-click on source file to open editor
- Outline view is shown for file in editor
- You should see warnings on the include files: we will fix this later
- Console shows results of build



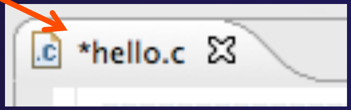
```
/*
 ============================================================
 Name        : hello_world.c
 Author      : Eclipse PTP
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================
 */

#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hello
    return EXIT_SUCCESS;
}
```

Editors



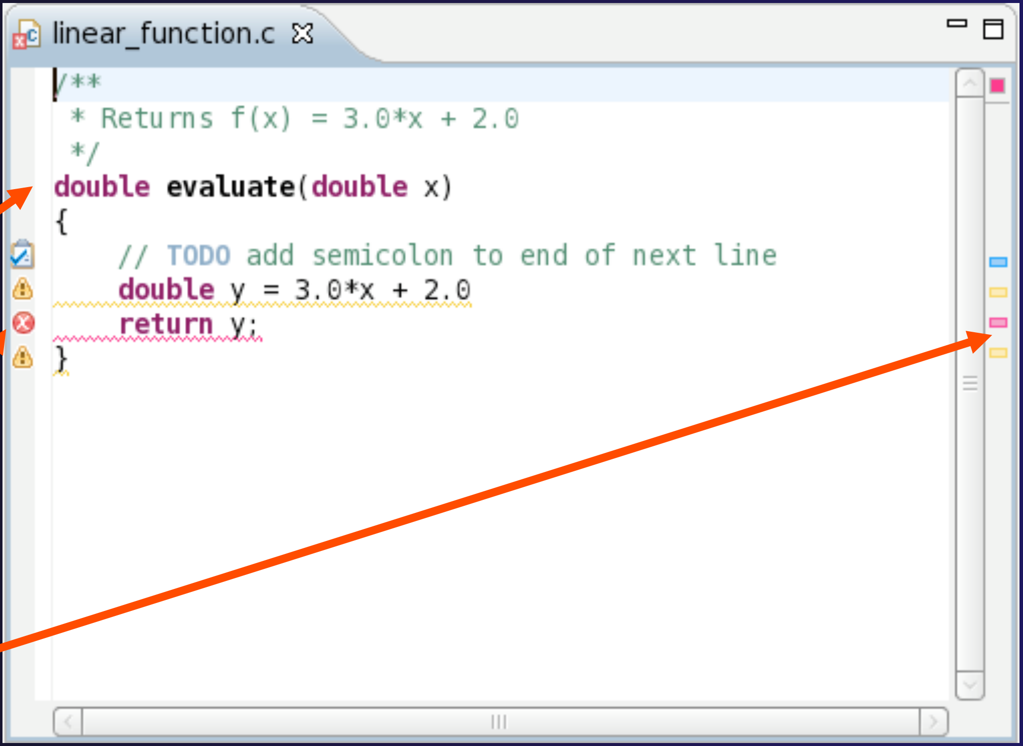
```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!
    return EXIT_SUCCESS;
}
```

- An editor for a resource (e.g. a file) opens when you double-click on a resource
- The type of editor depends on the type of the resource
  - .c files are opened with the C/C++ editor
  - Some editors do not just edit raw text
- 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
- Save the changes by using Command/ Ctrl-S or **File>Save**



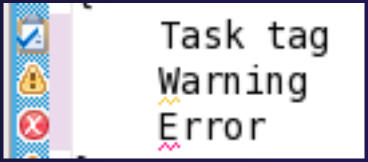
Source Code Editors & Markers

- A source code editor is a special type of editor for manipulating source code
- Language features are highlighted
- Marker bars for showing
  - Breakpoints
  - Errors/warnings
  - Task Tags, Bookmarks
- Location bar for navigating to interesting features in the entire file



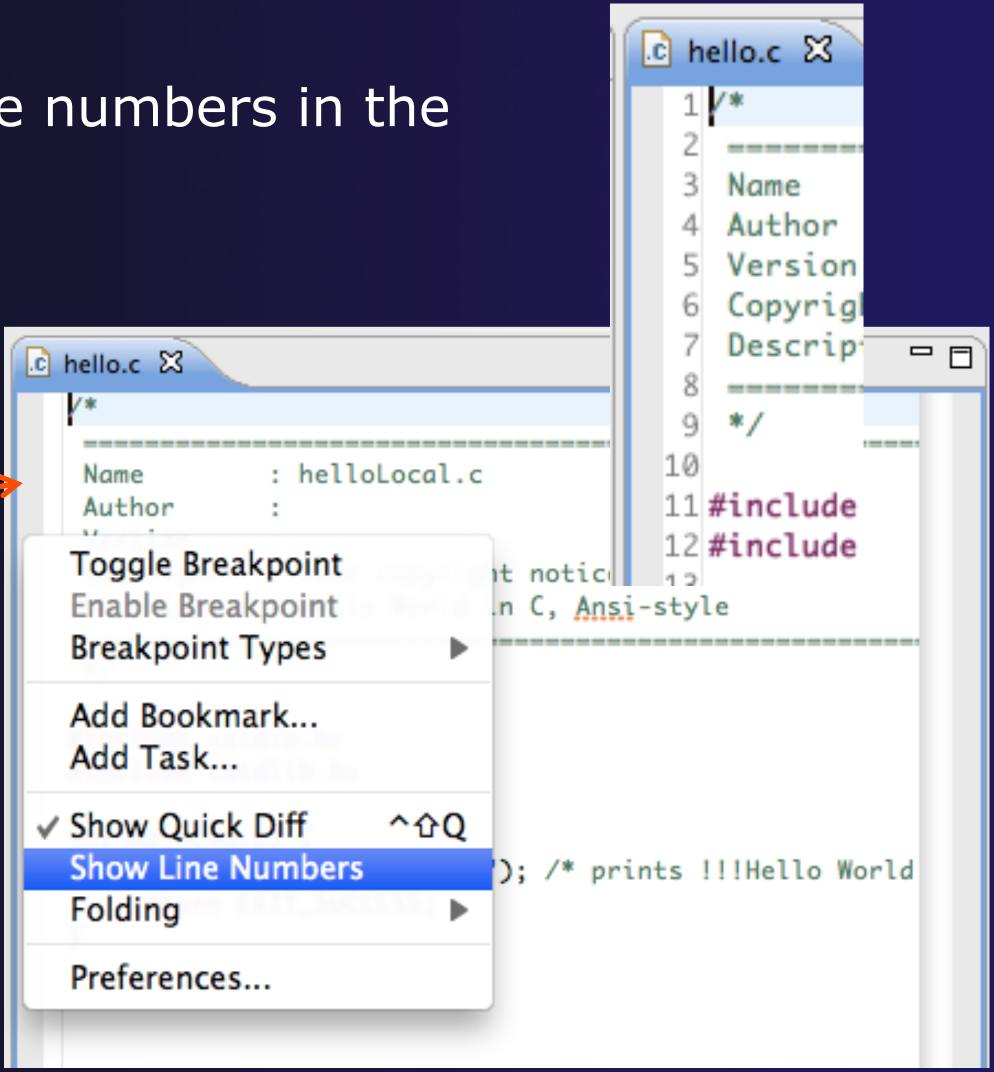
```
/**
 * Returns f(x) = 3.0*x + 2.0
 */
double evaluate(double x)
{
    // TODO add semicolon to end of next line
    double y = 3.0*x + 2.0
    return y;
}
```

Icons:



Line Numbers

- Text editors can show line numbers in the left column
- To turn on line numbering:
  - Right-mouse click in the editor marker bar
  - Click on **Show Line Numbers**

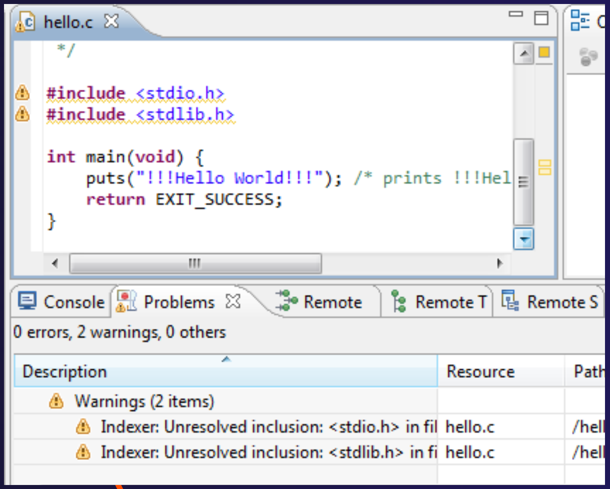


```
1 /*
2 ========
3 Name
4 Author
5 Version
6 Copyrig
7 Descrip
8 ========
9 */
10
11 #include
12 #include
```

```
/*
================================
Name        : helloLocal.c
Author      :
```

Include File Locations

- Content assist and navigation requires knowledge of include file location on the remote system
- The editor will indicate warnings on lines that have the problem
- **Problems View** will display a warning
- The project properties must be changed to resolve the problem



```
 */

#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hel
    return EXIT_SUCCESS;
}
```

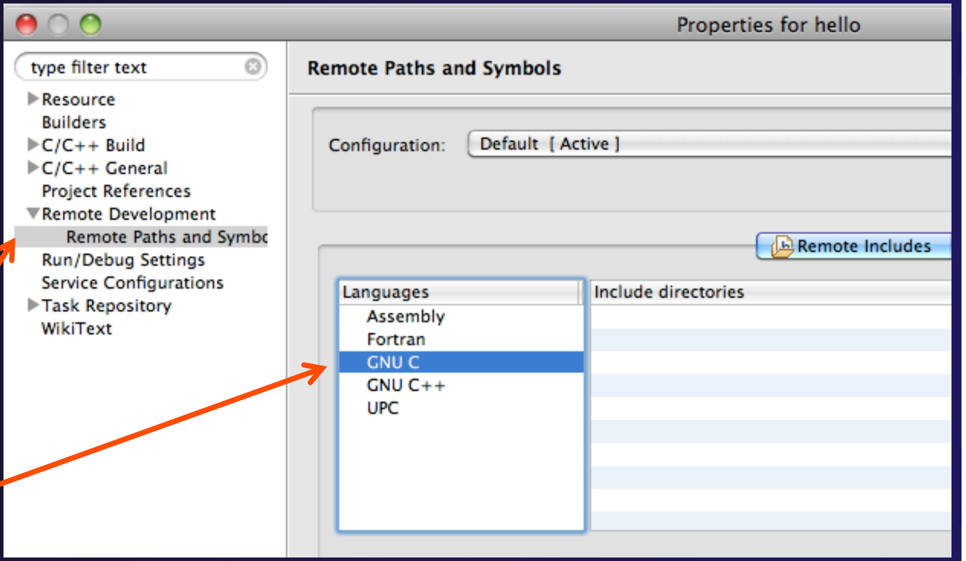
0 errors, 2 warnings, 0 others

Description	Resource	Path
Warnings (2 items)		
Indexer: Unresolved inclusion: <stdio.h> in fil	hello.c	/hell
Indexer: Unresolved inclusion: <stdlib.h> in fi	hello.c	/hell

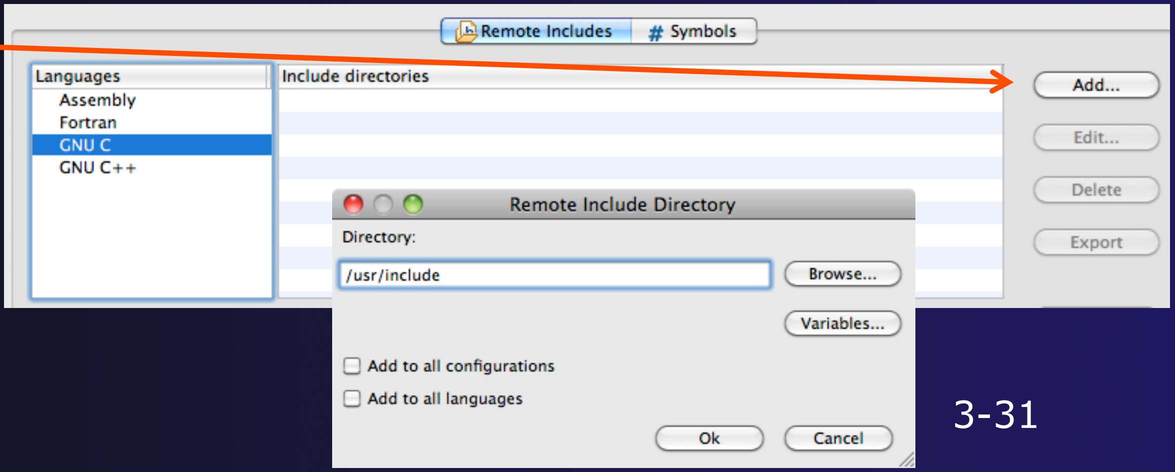
Indexer: Unresolved inclusion: <stdio.h> in file: /u/ac/etrain1/hello/hello.c:11. Please re-configure project's remote include paths or symbols.

Changing the Project Properties

- Open the project properties by right-clicking on project and select **Properties**
- Expand **Remote Development**
- Select **Remote Paths and Symbols**
- Select **GNU C** to change C paths and symbols
- Click **Add**
- Enter "/usr/include"
- Click **OK**



Configuration: Default [Active]

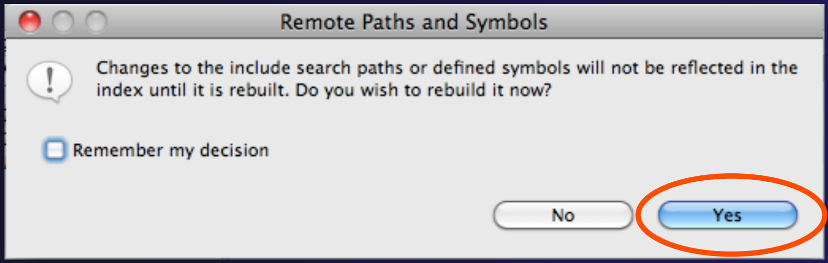


Directory: /usr/include

- [ ] Add to all configurations
- [ ] Add to all languages

Saving the Project Properties

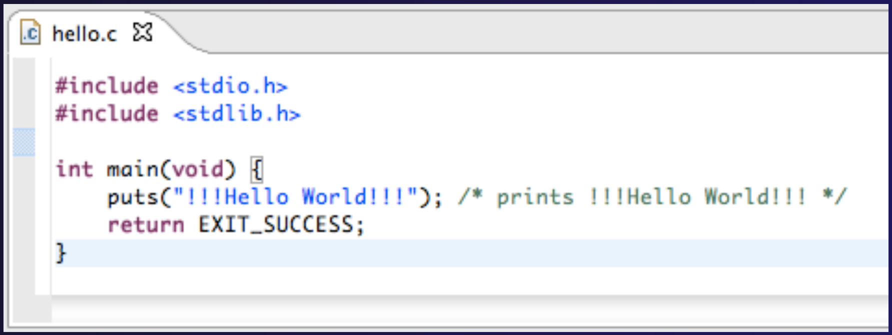
- Click **OK** to save the Project Properties
- You will be prompted to rebuild the index
  - Select **Yes**


Remote Paths and Symbols

Changes to the include search paths or defined symbols will not be reflected in the index until it is rebuilt. Do you wish to rebuild it now?

Remember my decision

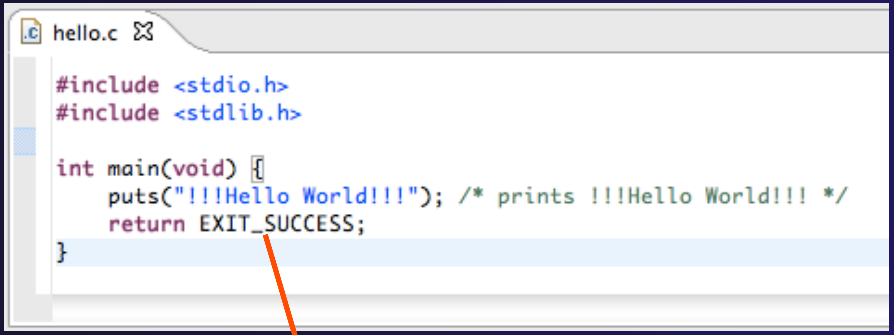
- Red warnings should be gone from editor, since Eclipse knows the location of the include files now


hello.c

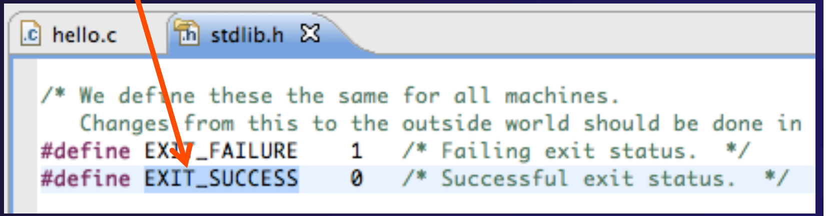
```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hello World!!! */
    return EXIT_SUCCESS;
}
```

Navigating to Other Files

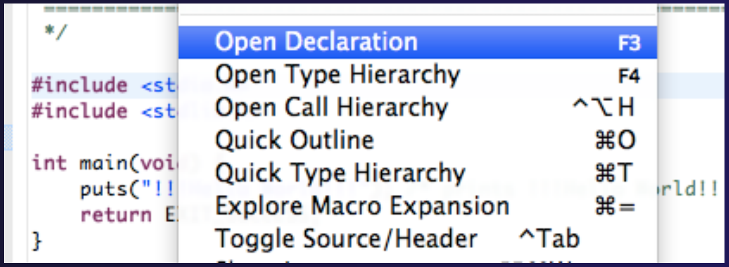
- On demand hyperlink
  - Hold down Command/Ctrl key
  - Click on element to navigate to its definition in the header file (Exact key combination depends on your OS)
  - E.g. Command/Ctrl and click on EXIT_SUCCESS
- Open declaration
  - Right-click and select **Open Declaration** will also open the file in which the element is declared
  - E.g. right-click on stdio.h and select **Open Declaration**

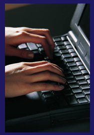


```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hello World!!! */
    return EXIT_SUCCESS;
}
```



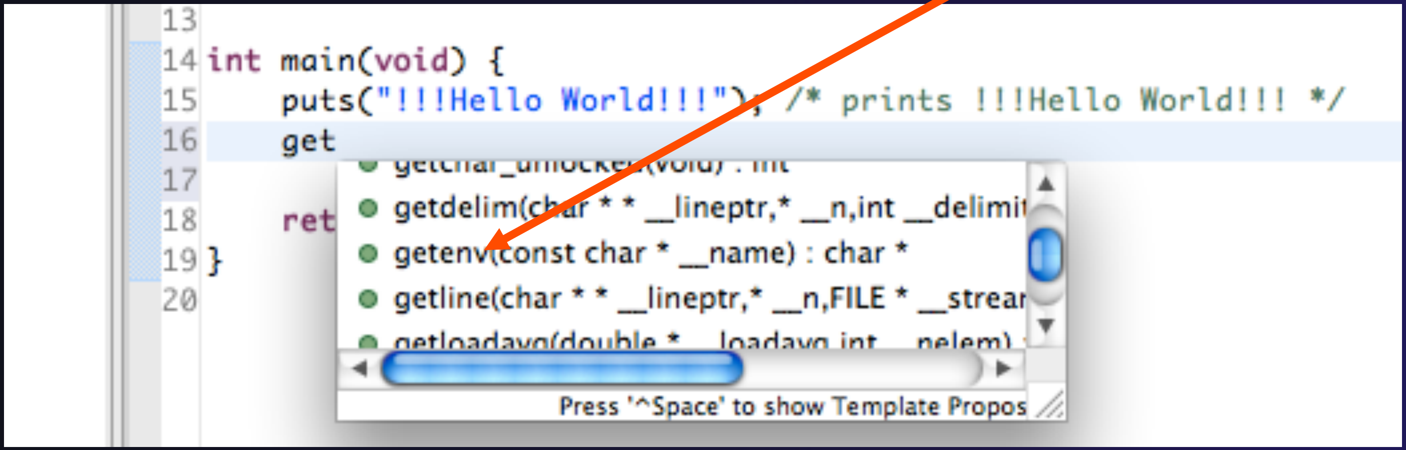
```
/* We define these the same for all machines.
   Changes from this to the outside world should be done in
#define EXIT_FAILURE    1   /* Failing exit status.  */
#define EXIT_SUCCESS    0   /* Successful exit status.  */
```





Content Assist & Templates

- Type an incomplete function name e.g. "get" into the editor, and hit **ctrl-space**
- Select desired completion value with cursor or mouse

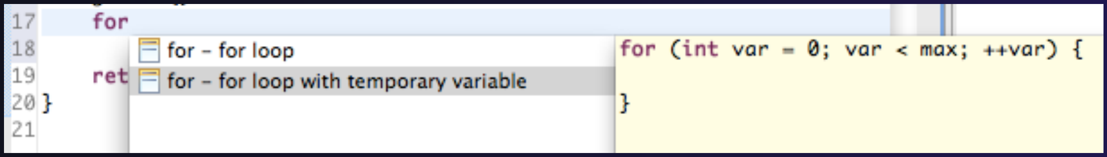


```
int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hello World!!! */
    get

    ret
}
```

getchar_unlocked(void) : int
getdelim(char * * __lineptr,* __n,int __delimit
getenv(const char * __name) : char *
getline(char * * __lineptr,* __n,FILE * __strear
getloadavg(double * __loadavg,int __nelem)
Press '^Space' to show Template Propos

- Code Templates: type 'for' and Ctrl-space

Hit ctrl-space again for code templates

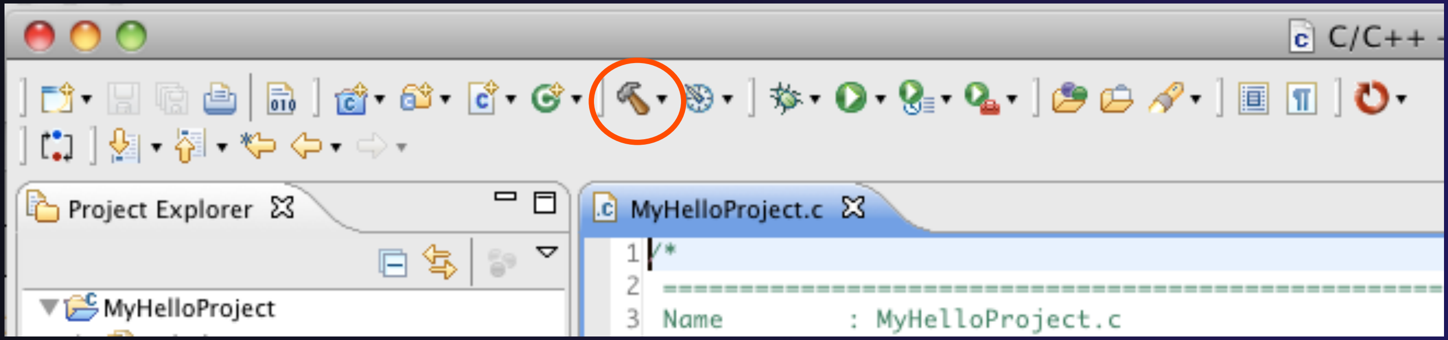


```
    for

    ret
}
```

```
for (int var = 0; var < max; ++var) {

}
```

Building the Project

- The project should build automatically when created
- If there is no makefile, then the build will fail
- To manually build, select the project and press the the "build" button



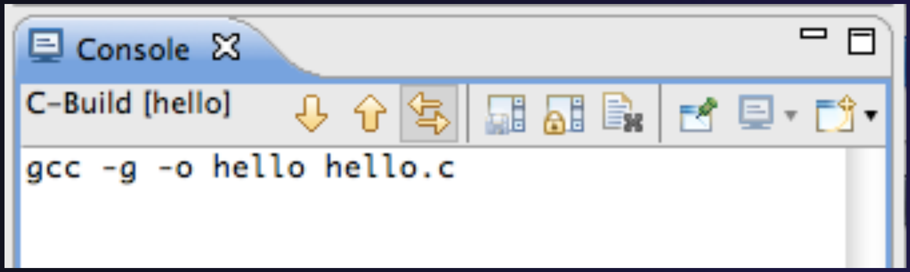
```
/*
================================================================
Name        : MyHelloProject.c
```

- Alternatively, select **Project>Build Project**
- To rebuild if project is already built, **Project > Clean...**

Building the Project (2)

After building the project:

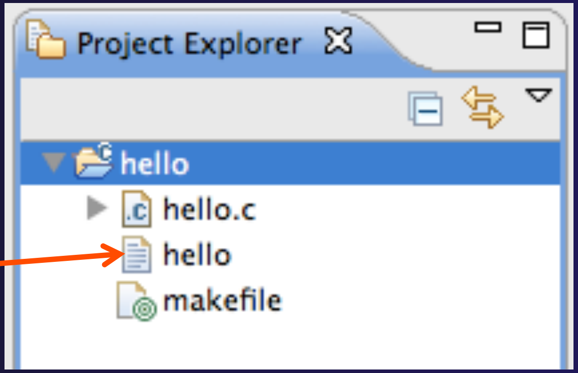
✦ The **Console** view shows build output

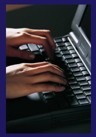


```
gcc -g -o hello hello.c
```

✦ If the build is successful, the executable should appear in the project

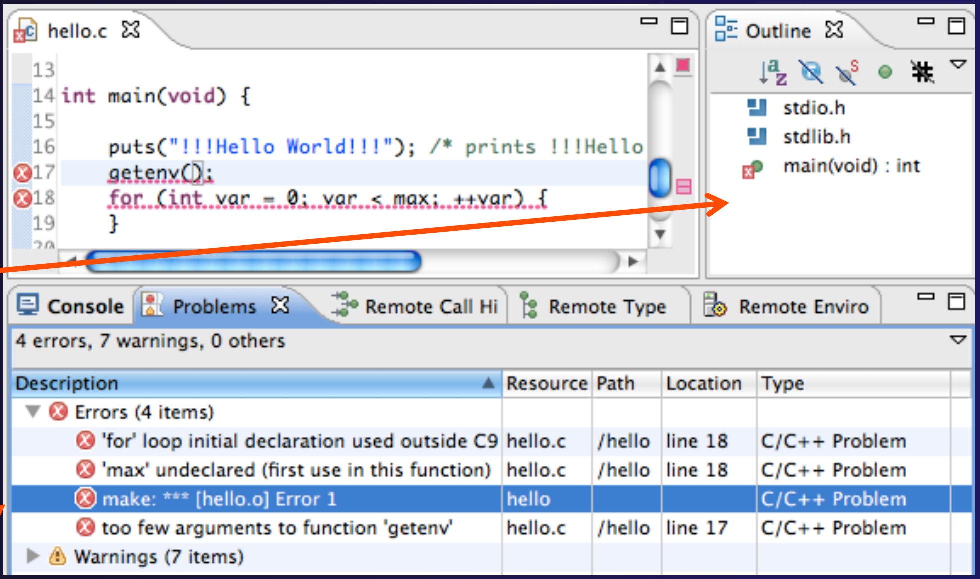
Executable





Build Problems

- If there are problems, they will be shown in a variety of ways
  - Marker on editor line
  - Marker on overview ruler
  - Listed in the **Problems view**
- Double-click on line in **Problems view** to go to location of error

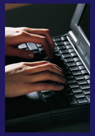


```
int main(void) {

    puts("!!!Hello World!!!"); /* prints !!!Hello
    getenv();
    for (int var = 0; var < max; ++var) {
    }
```

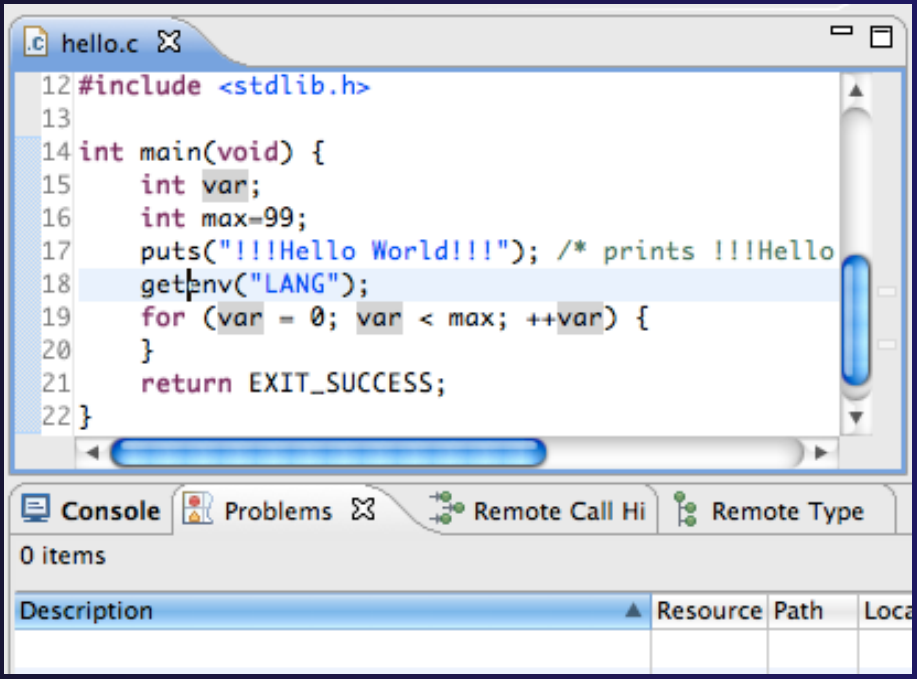
4 errors, 7 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (4 items)				
'for' loop initial declaration used outside C9	hello.c	/hello	line 18	C/C++ Problem
'max' undeclared (first use in this function)	hello.c	/hello	line 18	C/C++ Problem
make: *** [hello.o] Error 1	hello			C/C++ Problem
too few arguments to function 'getenv'	hello.c	/hello	line 17	C/C++ Problem
Warnings (7 items)				

Fix Build Problems



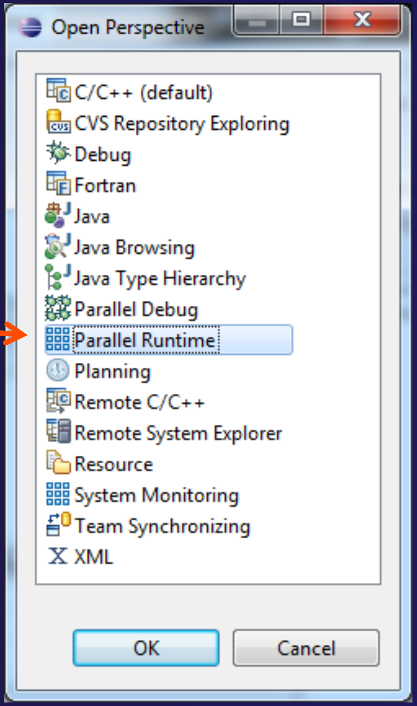
- Fix errors by giving **getenv** an argument and fixing declarations as shown
- Save the file
- Rebuild by pressing build button
- **Problems view** is now empty

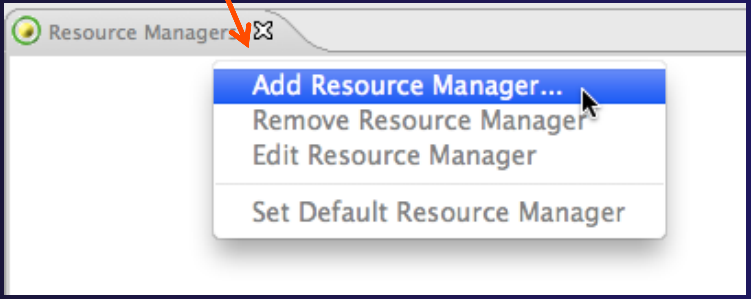


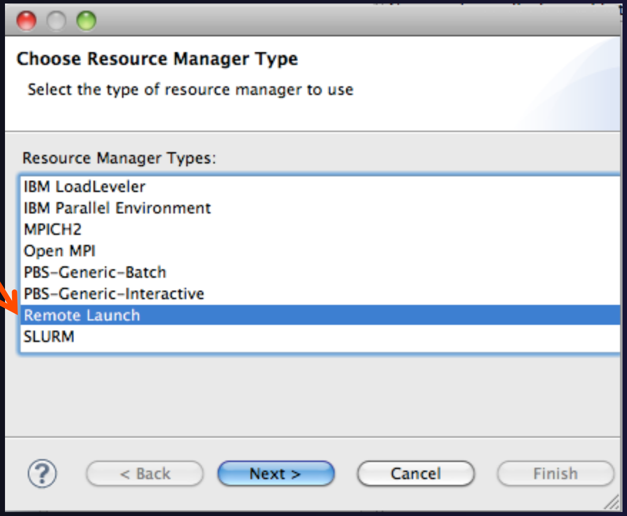
```
#include <stdlib.h>

int main(void) {
    int var;
    int max=99;
    puts("!!!Hello World!!!"); /* prints !!!Hello
    getenv("LANG");
    for (var = 0; var < max; ++var) {
    }
    return EXIT_SUCCESS;
}
```

Create a Resource Manager

- A *Resource Manager* specifies how/where programs will be launched
- Switch to the **Parallel Runtime** perspective
  - **Window>Open Perspective…**
- In the **Resource Managers** view, right-click and select **Add Resource Manager…**
- Select **Remote Launch** and **Next >**






Choose Resource Manager Type

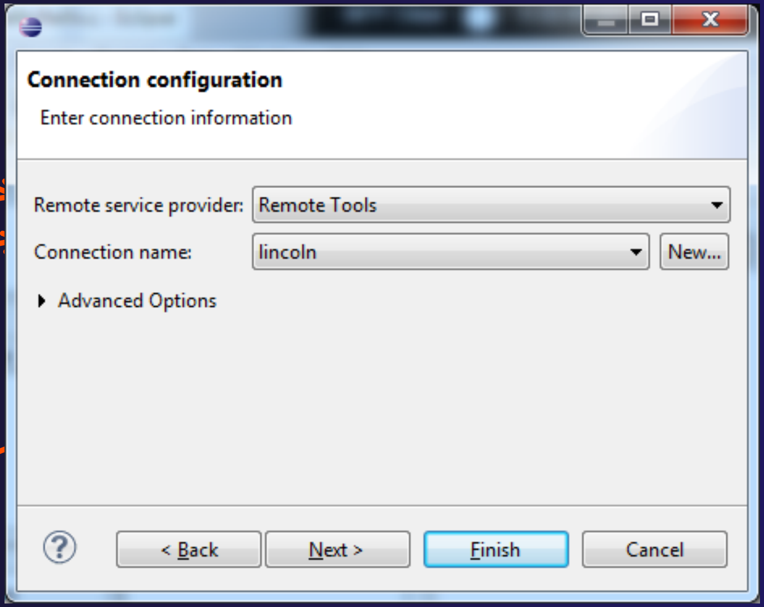
Select the type of resource manager to use

Resource Manager Types:

- IBM LoadLeveler
- IBM Parallel Environment
- MPICH2
- Open MPI
- PBS-Generic-Batch
- PBS-Generic-Interactive
- Remote Launch
- SLURM

Configure the Resource Manager

- Choose **Remote Tools** for **Remote service provider**
- Choose "lincoln.ncsa.uiuc.edu" for **Connection name**
  - This was the connection used when the project was created
- Click **Finish**



Connection configuration

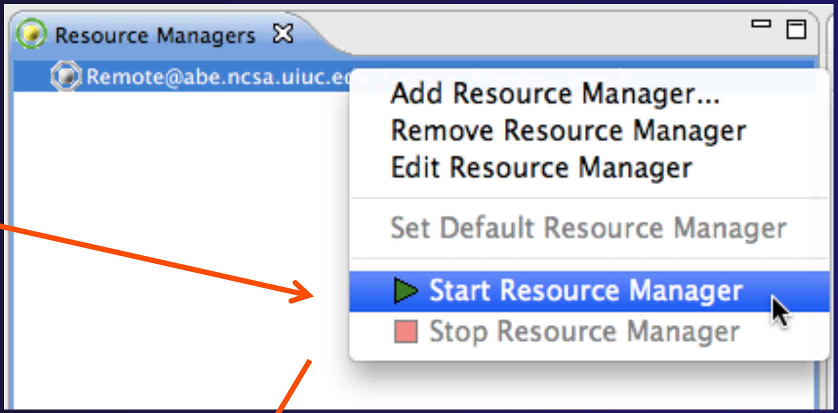
Enter connection information

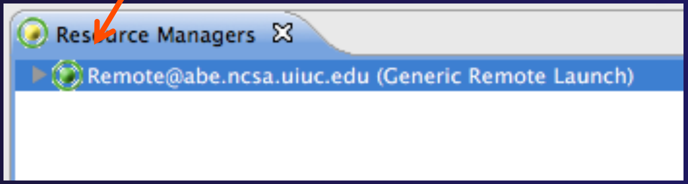
Remote service provider: Remote Tools

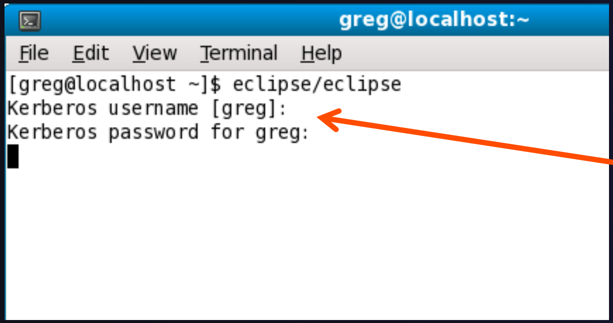
Connection name: lincoln

Start the Resource Manager

- Right-click on the new resource manager and select **Start Resource Manager** from the menu
- If the resource manager starts successfully, the icon should turn green
- An icon color of red indicates a problem occurred







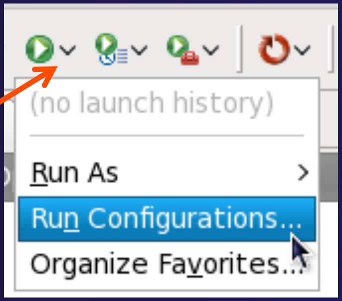
```
[greg@localhost ~]$ eclipse/eclipse
Kerberos username [greg]:
Kerberos password for greg:
```

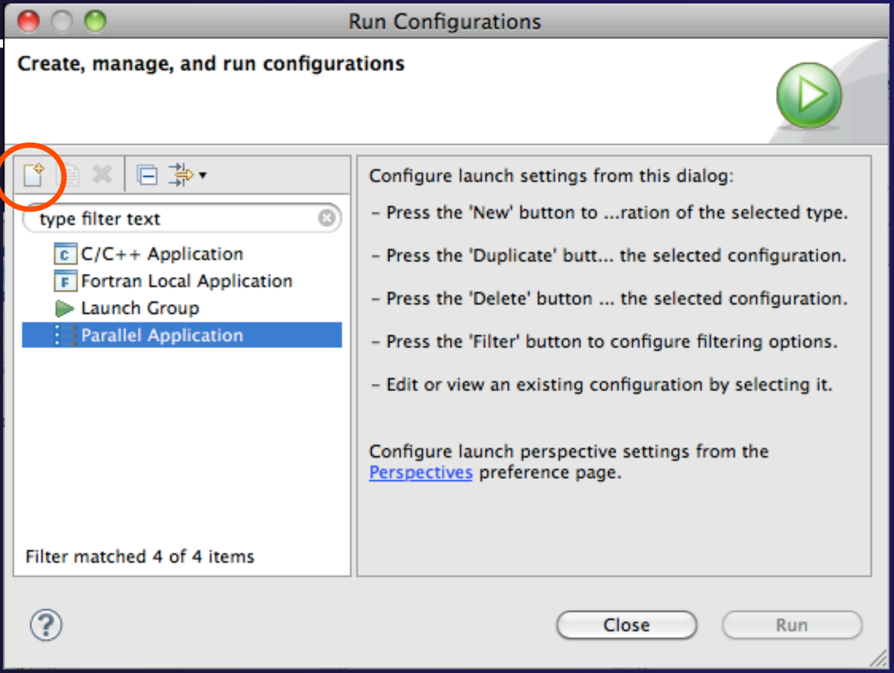
NOTE: On some Linux systems, starting a resource manager may appear to hang. Open the window you launched Eclipse from and check if there is a prompt for a kerberos username. Hit "enter" twice if you see the prompt.

Create a Run Configuration

To run the application, create a Run Configuration

- Open the run configurations dialog
  - Click on the arrow next to the run button
  - Or use **Run>Run Configurations…**
- Select **Parallel Application**
- Select the **New** button





Create, manage, and run configurations

Configure launch settings from this dialog:

- Press the 'New' button to …ration of the selected type.
- Press the 'Duplicate' butt… the selected configuration.
- Press the 'Delete' button … the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

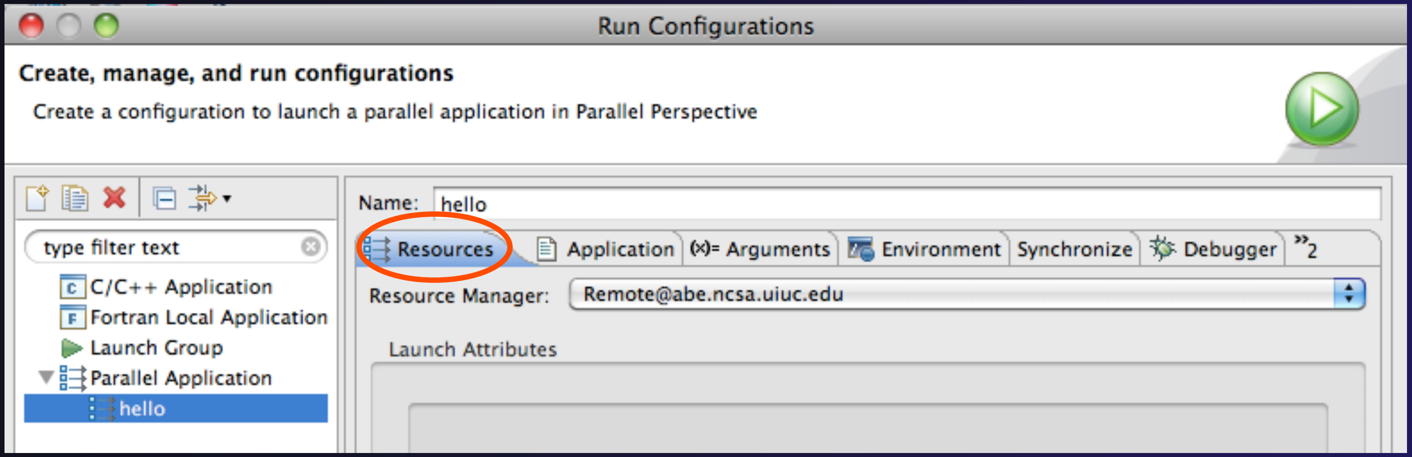
Configure launch perspective settings from the Perspectives preference page.

Filter matched 4 of 4 items

Depending on which flavor of Eclipse you installed, you might have more choices of application types

Complete the Resources Tab

- Select your Resource Manager
  - Should be selected automatically if it has been started
- The Remote Launch doesn't require additional attributes
  - Other resource managers may have additional attributes, such as a queue name, etc.

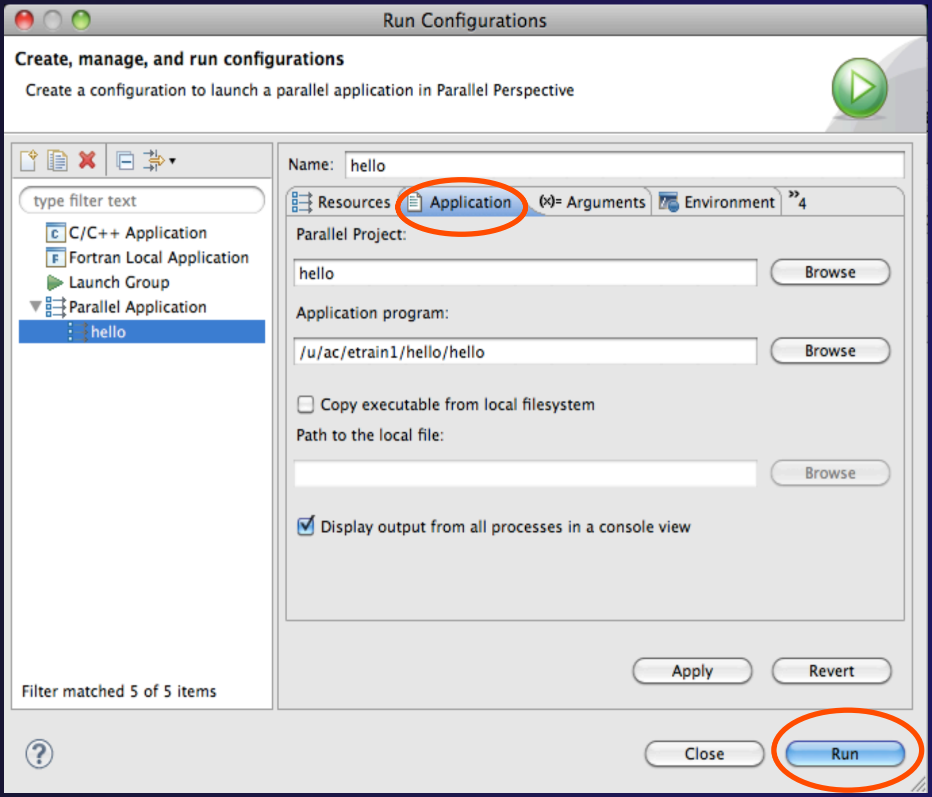


Name: hello

Resource Manager: Remote@abe.ncsa.uiuc.edu

Complete the Application Tab

- Make sure "hello" is selected for the **Parallel Project**
- Browse to find the executable file for the **Application program**
- Launch the application by clicking the **Run** button


Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

Name: hello

Parallel Project: hello

Application program: /u/ac/etrain1/hello/hello

☐ Copy executable from local filesystem

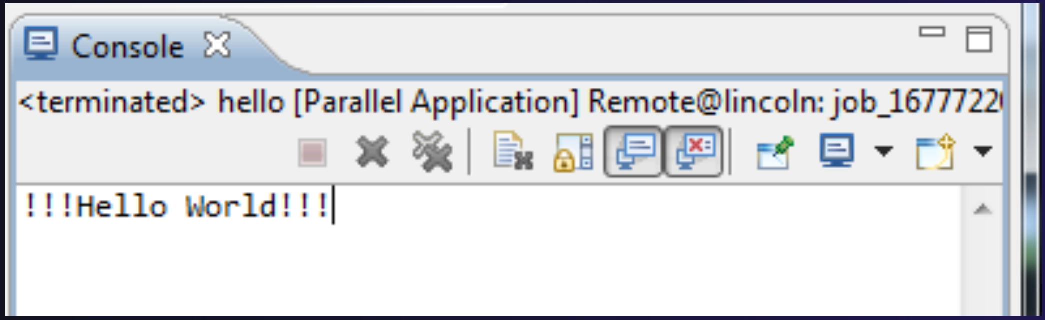
Path to the local file:

☑ Display output from all processes in a console view

Filter matched 5 of 5 items

Viewing Program Output

- When the program runs, the **Console** view should automatically become active
- Any output will be displayed in this view
  - Stdout is shown in black
  - Stderr is shown in red



```
<terminated> hello [Parallel Application] Remote@lincoln: job_1677722
```

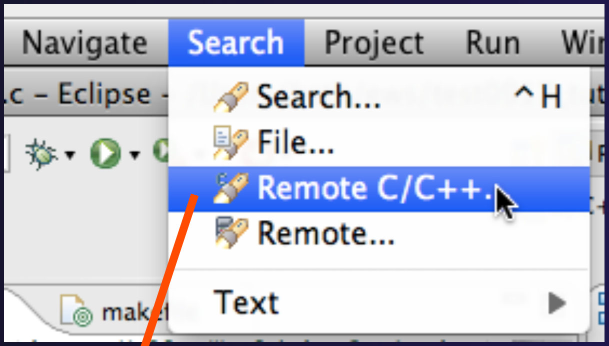
```
!!!Hello World!!!
```

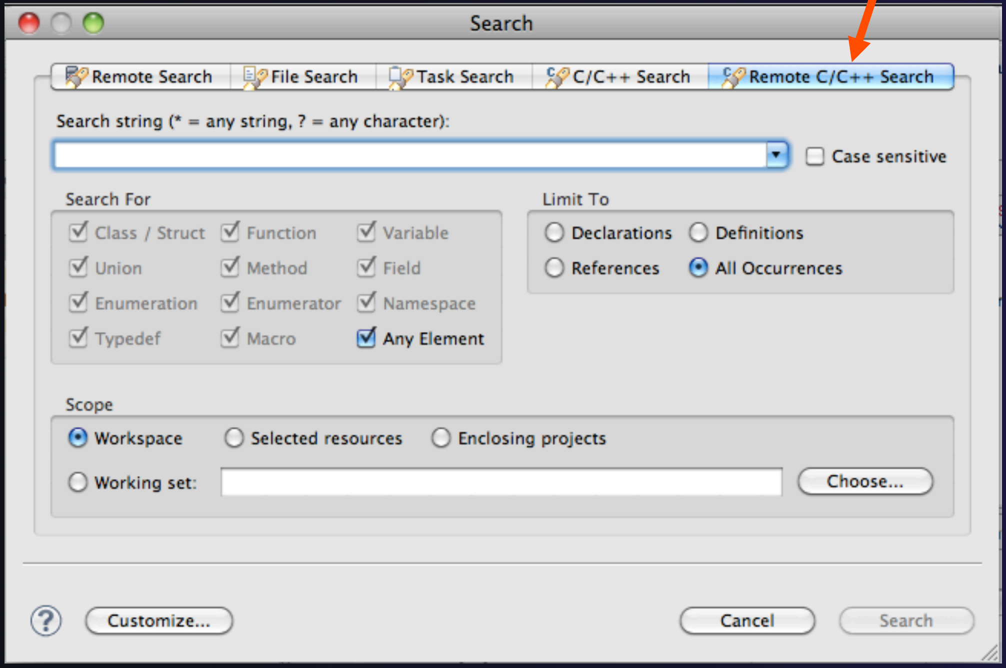
Other CDT features

- Searching
- Mark Occurrences
- Open Declaration / hyperlinking between files in the editor

First, return to the “Remote C/C++ Perspective”

Language-Based Searching





Search string (* = any string, ? = any character):

Case sensitive

Search For: [✓ Class / Struct] [✓ Function] [✓ Variable] [✓ Union] [✓ Method] [✓ Field] [✓ Enumeration] [✓ Enumerator] [✓ Namespace] [✓ Typedef] [✓ Macro] [✓ Any Element]

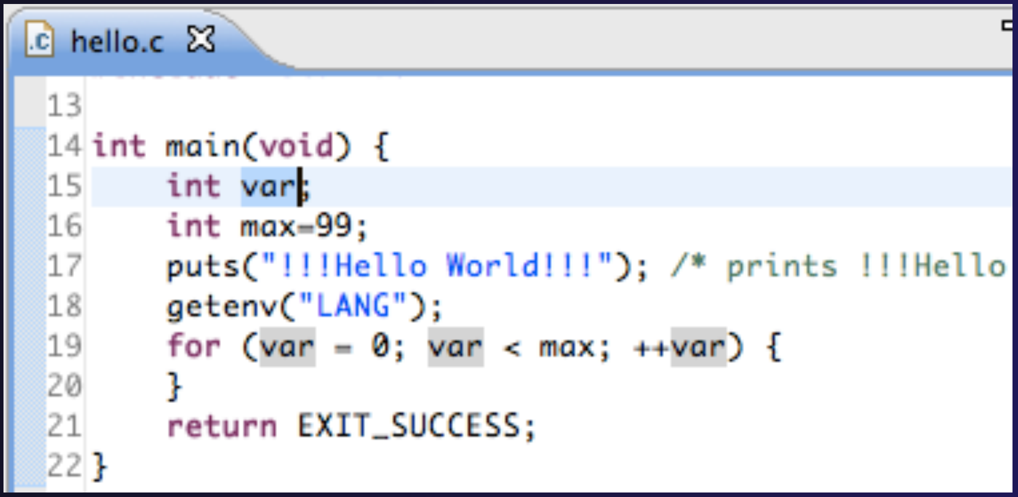
Limit To: Declarations, Definitions, References, [✓ All Occurrences]

Scope: [✓ Workspace] Selected resources, Enclosing projects, Working set:

- "Knows" what things can be declared in each language (functions, variables, classes, modules, etc.)
- For example, search for every call to a function whose name starts with "get"
- Search can be project- or workspace-wide

Mark Occurrences

- Double-click on a variable in the CDT editor
- All occurrences in the source file are highlighted to make locating the variable easier
- Alt-shift-O to turn off


hello.c

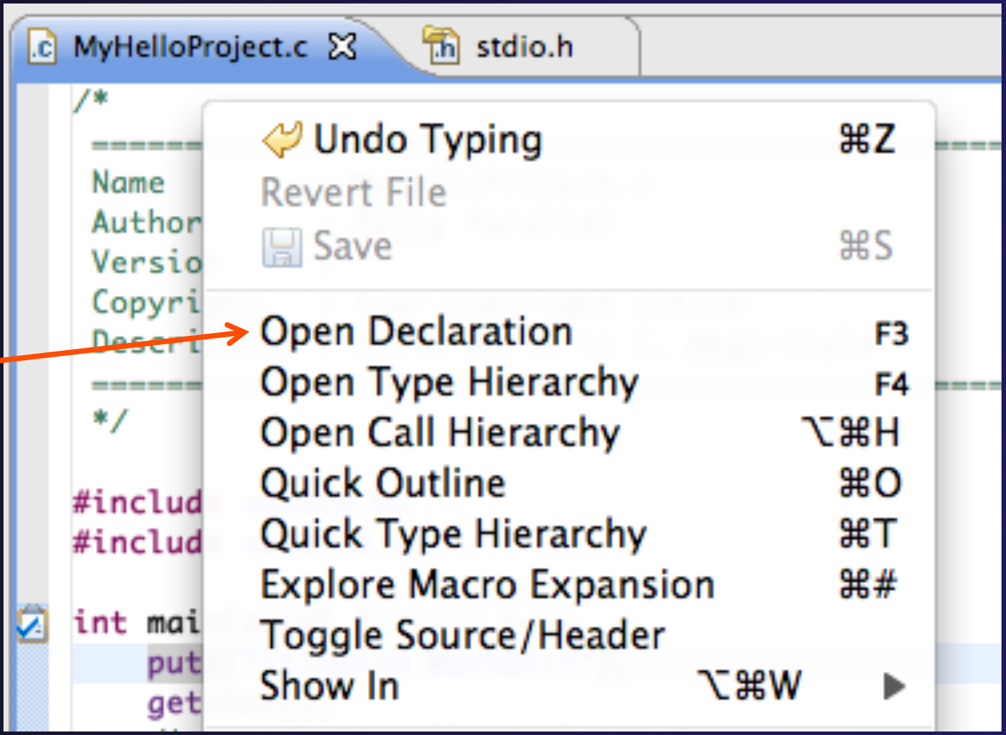
```
int main(void) {
    int var;
    int max=99;
    puts("!!!Hello World!!!"); /* prints !!!Hello
    getenv("LANG");
    for (var = 0; var < max; ++var) {
    }
    return EXIT_SUCCESS;
}
```

Open Declaration

- Jumps to the declaration of a variable, function, etc., even if it's in a different file

- Right-click on an identifier
- Click **Open Declaration**

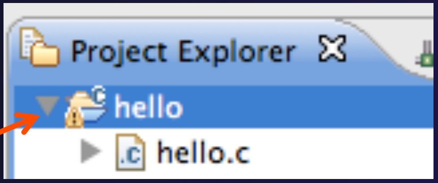
- Can also Ctrl-click (Mac: Cmd-click) on an identifier to "hyperlink" to its declaration

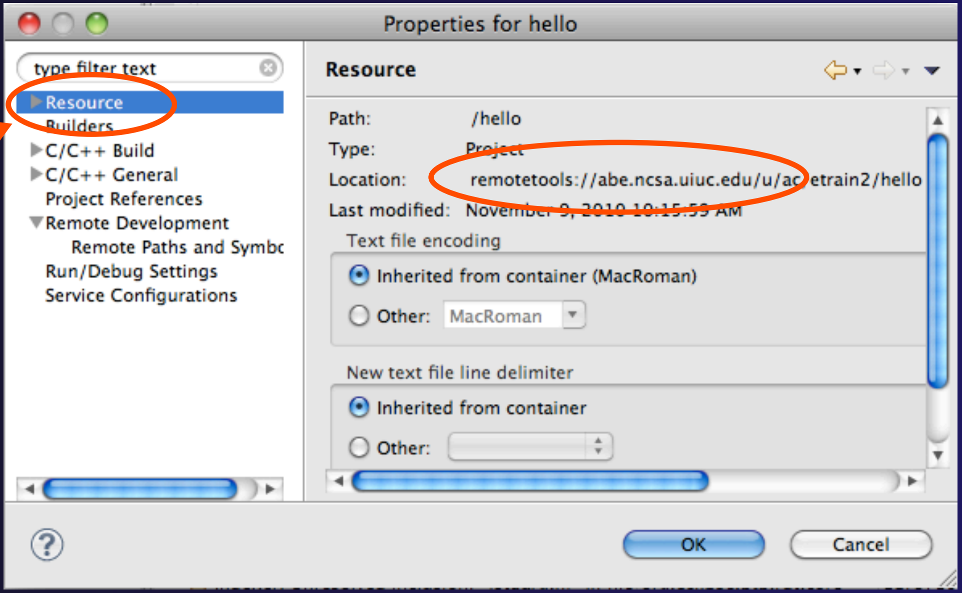


Remote Projects - Location

- How to tell where a project resides?
- Right-click Project
- Select **Properties**…

- In Properties dialog, select **Resource**





Resource

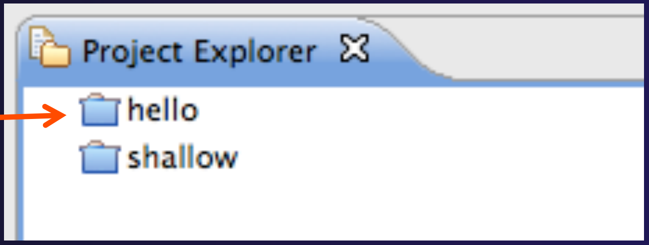
Path: /hello
Type: Project
Location: remotetools://abe.ncsa.uiuc.edu/u/ac/etrain2/hello
Last modified: November 9, 2010 10:15:59 AM

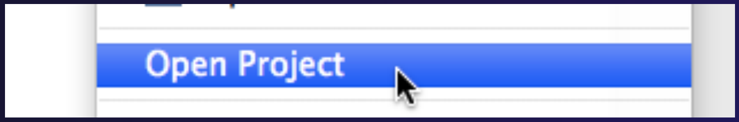
Text file encoding
[✓ Inherited from container (MacRoman)]
Other: MacRoman

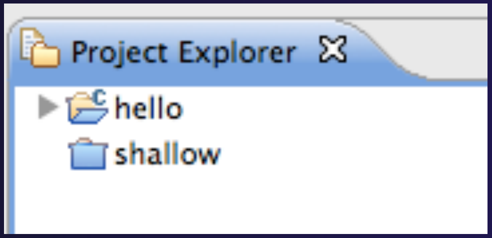
New text file line delimiter
[✓ Inherited from container]
Other:

Remote Projects - Reopening

- When re-opening Eclipse workbench, remote projects will be closed
- To re-open a closed project, Right-click on closed project and select **Open Project**
- Open project shows folder icon, and can be expanded to show contents of project





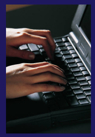


Module 4: Working with MPI

- Objective
  - Learn how to develop, build and launch a parallel (MPI) program on a remote parallel machine
- Contents
  - Remote project setup
  - Building with Makefiles
  - MPI assistance features
  - Working with resource managers
  - Launching a parallel application

Local vs. Remote

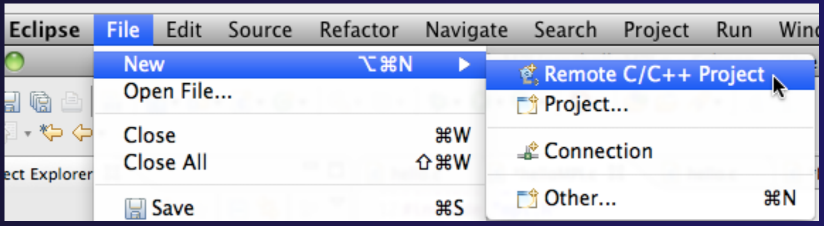
- PTP allows the program to be run locally if you have MPI installed
  - However we want to run the program on a remote machine
- We will now show you how to run a parallel program on a remote machine
  - Interactively
  - Through a batch system
  - Interactively through a batch system
- We have provided the source code to an MPI program on the remote machine
- The project will be created using this source code

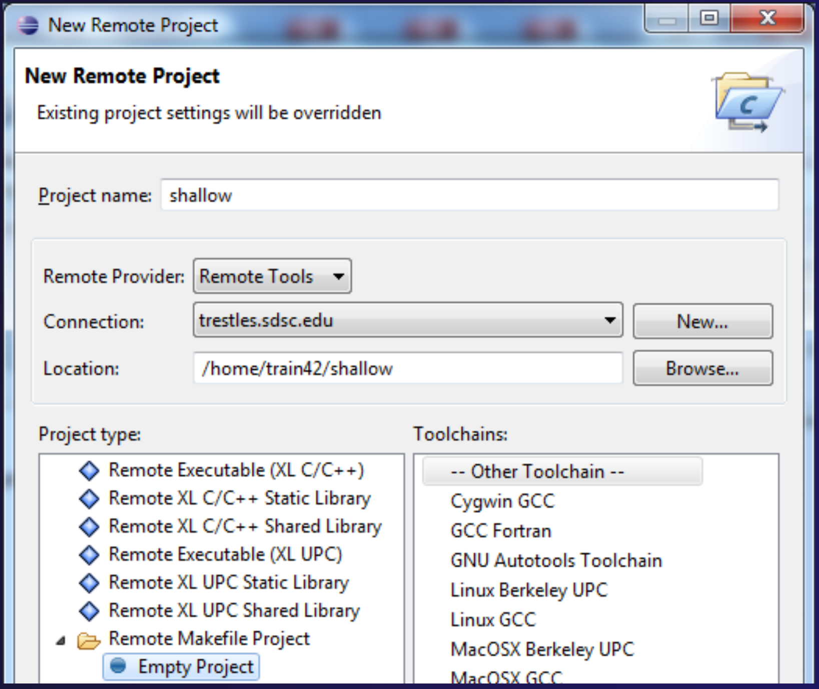


Creating a Remote MPI Project

- Like the previous module, create a new Remote C/C++ project
- Enter "shallow" for the **Project Name**
- Use the same **Connection** as before
- Click the **Browse...** button and choose the directory "shallow" in in your home directory
- Select a **Remote Makefile Project** as before
- Click **Finish**

You may be prompted to open the Remote C/C++ Perspective




New Remote Project

New Remote Project

Existing project settings will be overridden

Project name: shallow

Remote Provider: Remote Tools

Connection: trestles.sdsc.edu

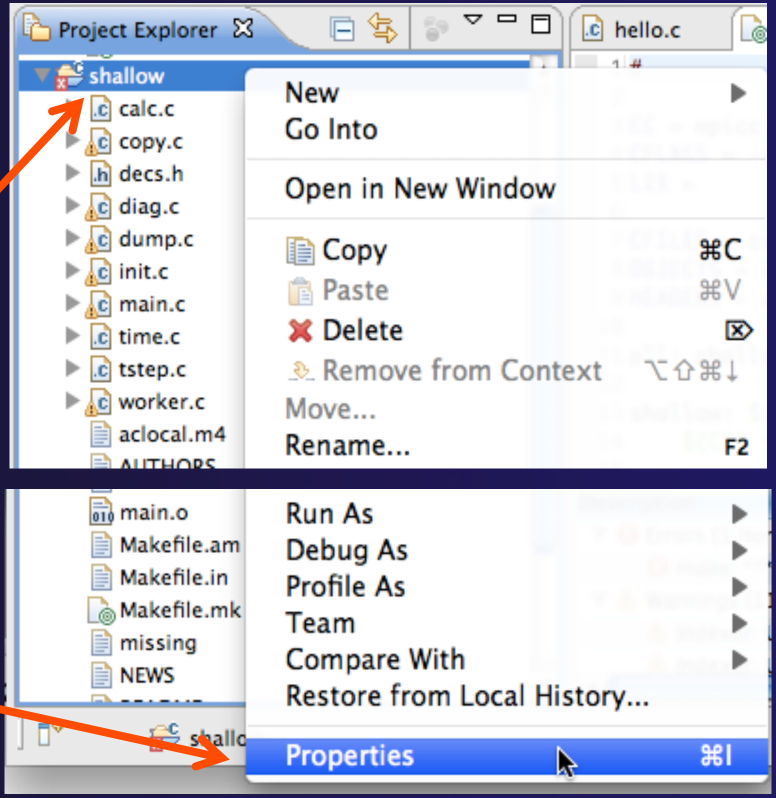
Location: /home/train42/shallow

Project type:
- Remote Executable (XL C/C++)
- Remote XL C/C++ Static Library
- Remote XL C/C++ Shared Library
- Remote Executable (XL UPC)
- Remote XL UPC Static Library
- Remote XL UPC Shared Library
- Remote Makefile Project
  - Empty Project

Toolchains:
- -- Other Toolchain --
- Cygwin GCC
- GCC Fortran
- GNU Autotools Toolchain
- Linux Berkeley UPC
- Linux GCC
- MacOSX Berkeley UPC
- MacOSX GCC

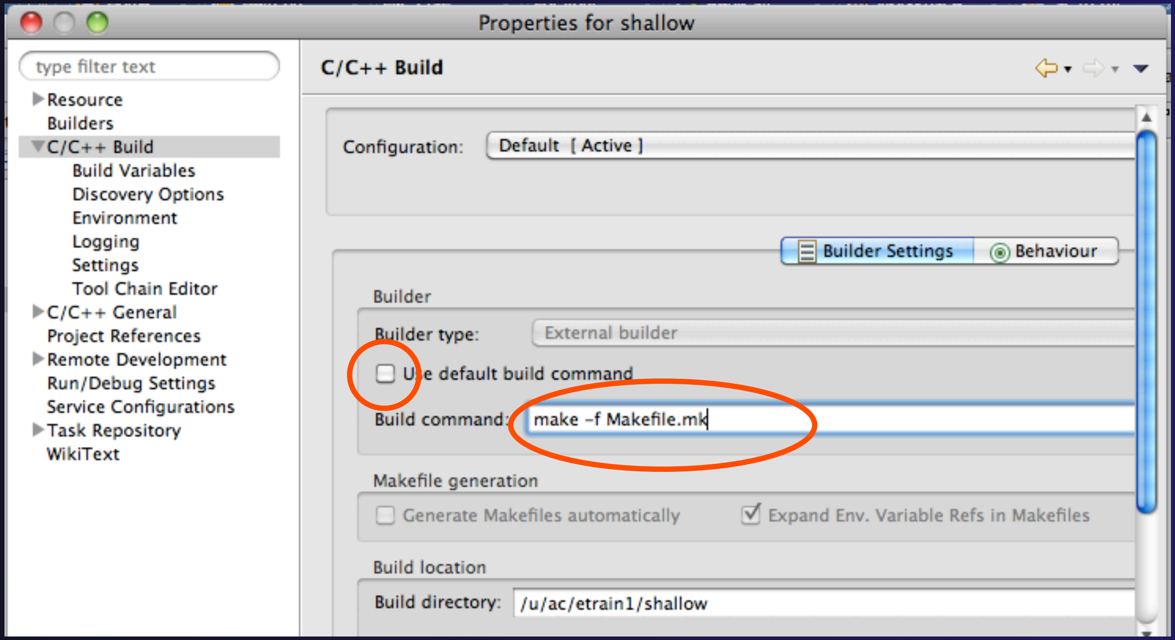
Changing the Project Build Properties

- The project makefile has a non-standard name Makefile.mk
- We need to change the build properties so that the project will build
  - By default, the project is built by running "make"
- Right-click on project "shallow" in the **Project Explorer**
- Select **Properties**



Changing the Build Command

- Select **C/C++ Build**
- Uncheck **Use default build command**
- Change the **Build command** to:
  - make –f Makefile.mk


Properties for shallow

C/C++ Build

Configuration: Default [Active]

Builder type: External builder

[ ] Use default build command

Build command: make –f Makefile.mk

Makefile generation

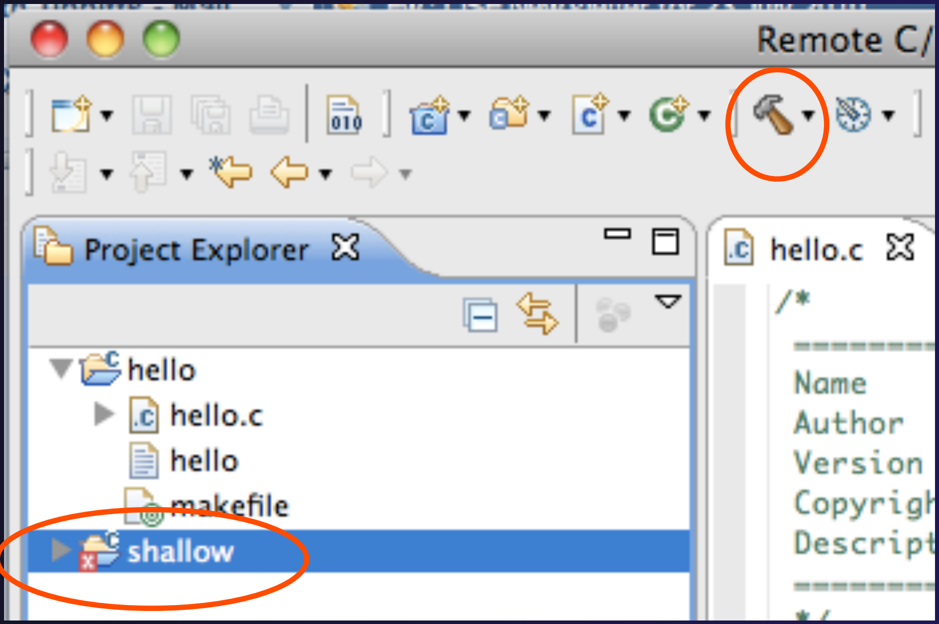
[ ] Generate Makefiles automatically [✓] Expand Env. Variable Refs in Makefiles

Build location

Build directory: /u/ac/etrain1/shallow

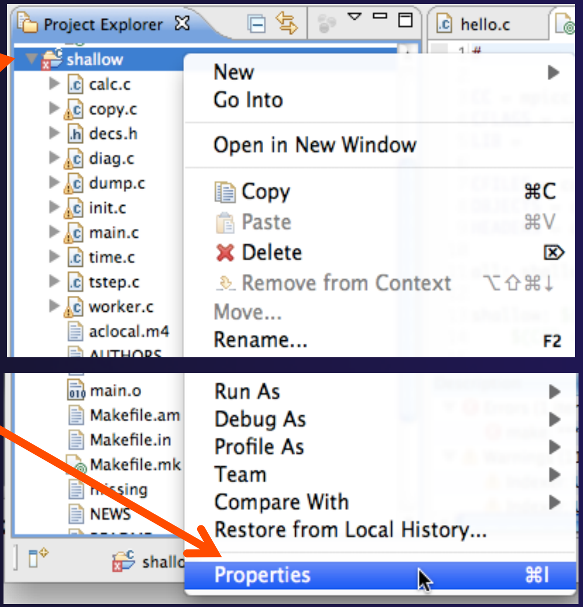
Building the Project

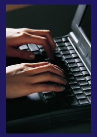
- Click **OK** to save project properties after changing build command
- Select project and hit the build button
- The project can be built at any time by hitting this button



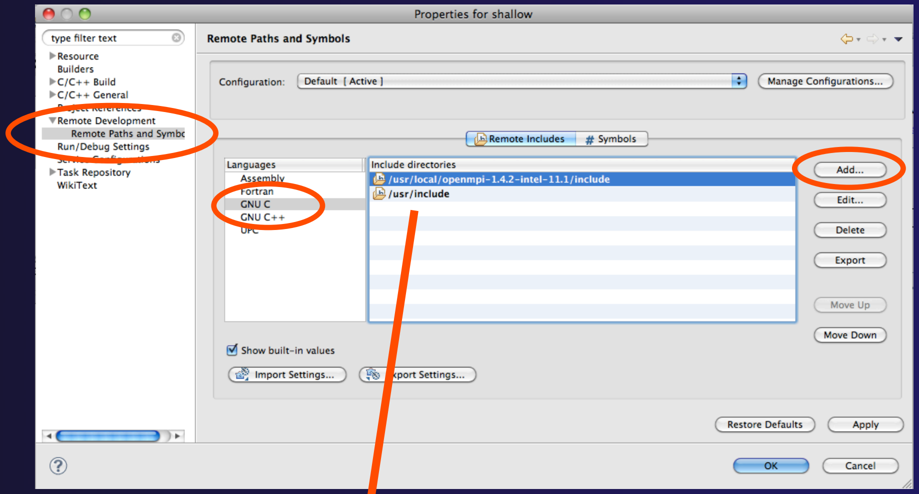
Include File Locations

- Like the previous example, Eclipse content assist and navigation require knowledge of include file locations on the remote system
  - Since the build will be running remotely, the compiler knows how to find include files
  - But Eclipse does not
- In **Project Explorer**, right-click on project
- Select **Properties**





Remote Paths and Symbols

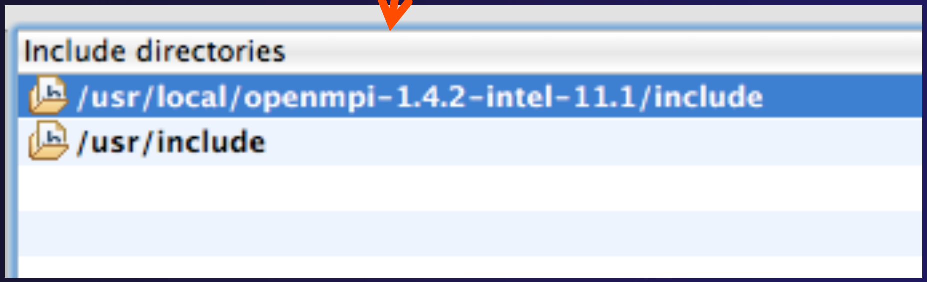


Configuration: Default [Active]

Include directories
- /usr/local/openmpi-1.4.2-intel-11.1/include
- /usr/include

In **Project Properties,**

- Expand **Remote Development**
- Select **Remote Paths and Symbols**
- Select **Languages>GNU C**
  - This is compiler on abe
- Click **Add...**
  - Enter /usr/local/openmpi-1.4.2-intel-11.1/include
- Click **OK**, then **Add...** again
  - Enter /usr/include
- Click **OK**
- Click **OK** to close preferences
- When prompted to rebuild index, click **OK**



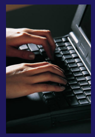
Include directories
- /usr/local/openmpi-1.4.2-intel-11.1/include
- /usr/include

MPI-Specific Features

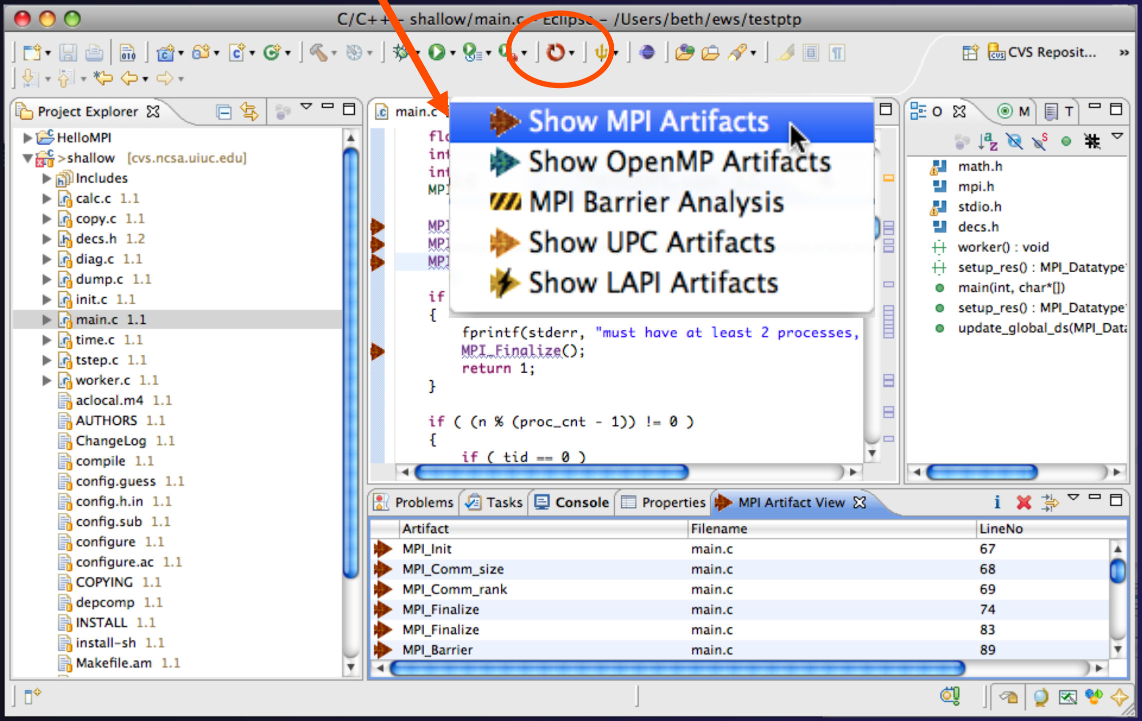
- PTP's Parallel Language Development Tools (PLDT) has several features specifically for developing MPI code
  - Show MPI Artifacts
  - Code completion
  - Context Sensitive Help for MPI
  - Hover Help
  - MPI Templates in the editor

More MPI features covered in
Module 7: Advanced Features

Show MPI Artifacts



- In Project Explorer, select a project, folder, or a single source file
  - The analysis will be run on the selected resources
- Run the analysis by clicking on drop-down menu next to the analysis button
- Selecting **Show MPI Artifacts**

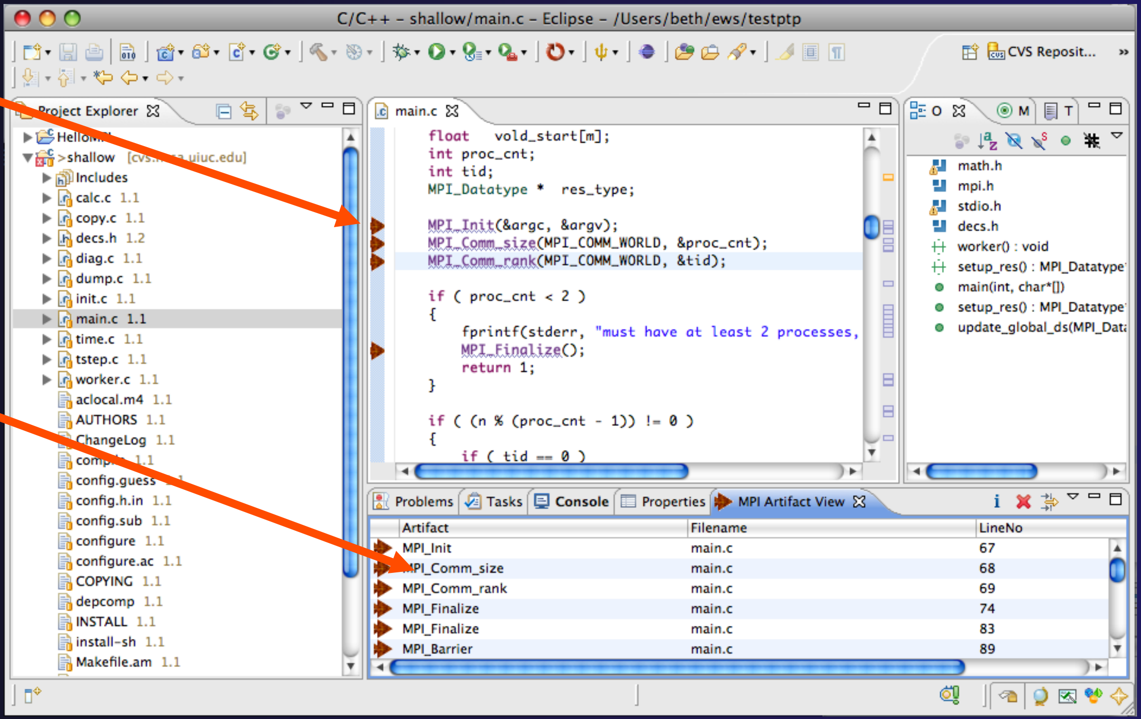


```
        fprintf(stderr, "must have at least 2 processes,
        MPI_Finalize();
        return 1;
    }

    if ( (n % (proc_cnt - 1)) != 0 )
    {
        if ( tid == 0 )
```

Artifact	Filename	LineNo
MPI_Init	main.c	67
MPI_Comm_size	main.c	68
MPI_Comm_rank	main.c	69
MPI_Finalize	main.c	74
MPI_Finalize	main.c	83
MPI_Barrier	main.c	89

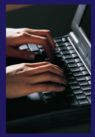
MPI Artifact View

- Markers indicate the location of artifacts in editor
- The **MPI Artifact View** list the type and location of each artifact
- Navigate to source code line by double-clicking on the artifact
- Run the analysis on another file (or entire project!) and its markers will be added to the view
- Remove markers via ✖
- Click on column headings to sort

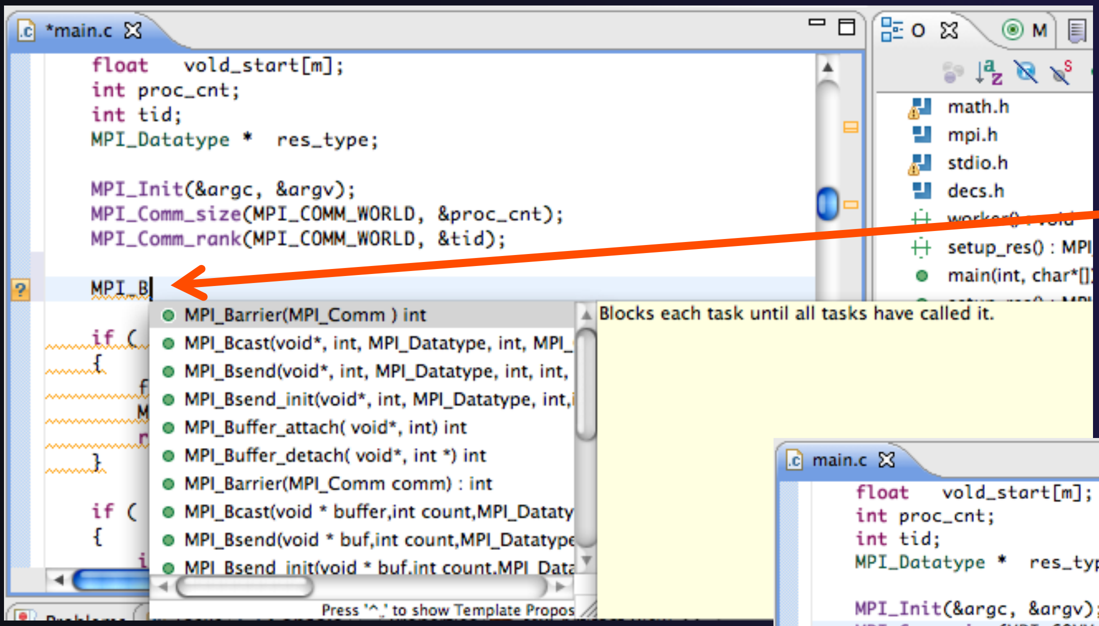


```
    float   vold_start[m];
    int proc_cnt;
    int tid;
    MPI_Datatype *  res_type;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
    MPI_Comm_rank(MPI_COMM_WORLD, &tid);

    if ( proc_cnt < 2 )
    {
        fprintf(stderr, "must have at least 2 processes,
        MPI_Finalize();
        return 1;
    }

    if ( (n % (proc_cnt - 1)) != 0 )
    {
        if ( tid == 0 )
```

Artifact	Filename	LineNo
MPI_Init	main.c	67
MPI_Comm_size	main.c	68
MPI_Comm_rank	main.c	69
MPI_Finalize	main.c	74
MPI_Finalize	main.c	83
MPI_Barrier	main.c	89



MPI Editor Features



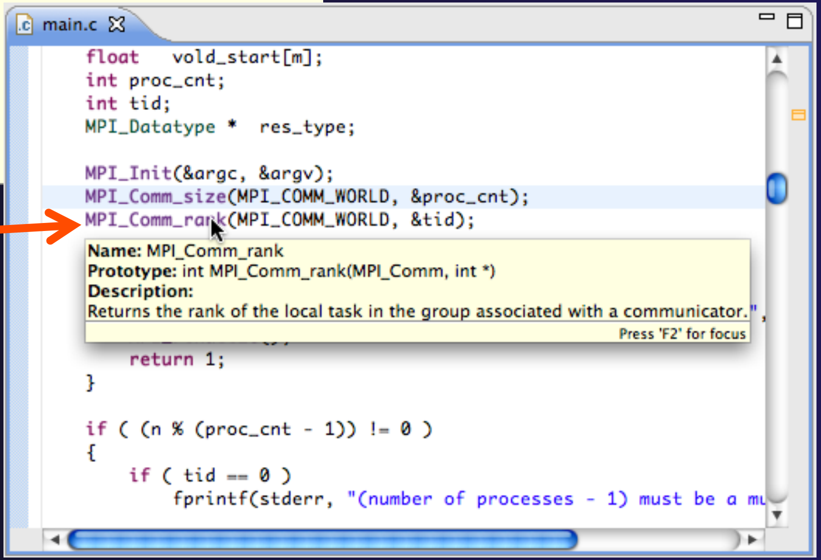
*main.c

```
float   vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype *  res_type;

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
MPI_Comm_rank(MPI_COMM_WORLD, &tid);

MPI_B

if (
{
    f
    M
    r
}

if (
{
    i
```

```
MPI_Barrier(MPI_Comm ) int
MPI_Bcast(void*, int, MPI_Datatype, int, MPI_
MPI_Bsend(void*, int, MPI_Datatype, int, int,
MPI_Bsend_init(void*, int, MPI_Datatype, int,i
MPI_Buffer_attach( void*, int) int
MPI_Buffer_detach( void*, int *) int
MPI_Barrier(MPI_Comm comm) : int
MPI_Bcast(void * buffer,int count,MPI_Dataty
MPI_Bsend(void * buf,int count,MPI_Datatype
MPI_Bsend_init(void * buf,int count,MPI_Data
Press '^,' to show Template Propos
```

Blocks each task until all tasks have called it.

- Code completion will show all the possible MPI keyword completions
- Enter the start of a keyword then press <ctrl-space>



main.c

```
float   vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype *  res_type;

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
MPI_Comm_rank(MPI_COMM_WORLD, &tid);
```

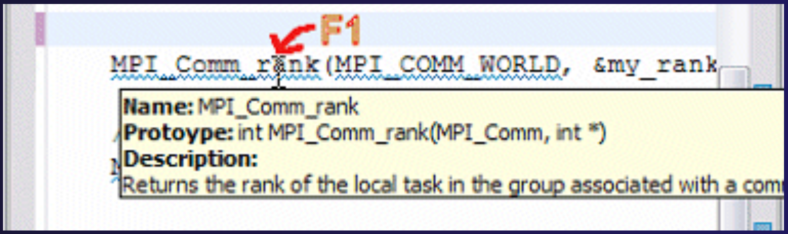
Name: MPI_Comm_rank
Prototype: int MPI_Comm_rank(MPI_Comm, int *)
Description:
Returns the rank of the local task in the group associated with a communicator.
Press 'F2' for focus

```
    return 1;
}

if ( (n % (proc_cnt - 1)) != 0 )
{
    if ( tid == 0 )
        fprintf(stderr, "(number of processes - 1) must be a mu
```

- Hover over MPI API
- Displays the function prototype and a description

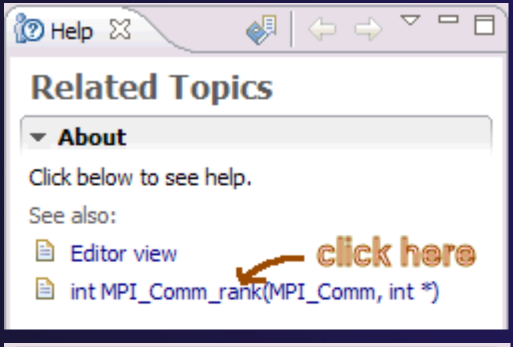
Context Sensitive Help

- Click mouse, then press help key when the cursor is within a function name
  - Windows: **F1** key
  - Linux: **ctrl-F1** key
  - MacOS X: **Help** key or **Help▸Dynamic Help**
- A help view appears (**Related Topics**) which shows additional information (You may need to click on MPI API in editor again, to populate)
- Click on the function name to see more information
- Move the help view within your Eclipse workbench, if you like, by dragging its title tab



MPI_Comm_rank(MPI_COMM_WORLD, &my_rank

Name: MPI_Comm_rank
Protoype: int MPI_Comm_rank(MPI_Comm, int *)
Description:
Returns the rank of the local task in the group associated with a com



Related Topics

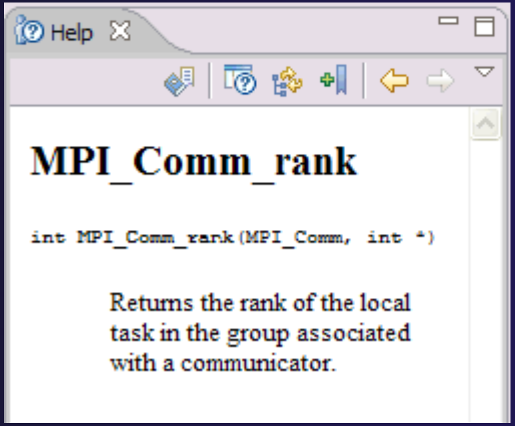
About

Click below to see help.

See also:

Editor view

int MPI_Comm_rank(MPI_Comm, int *)



MPI_Comm_rank

int MPI_Comm_rank(MPI_Comm, int *)

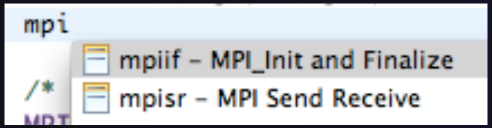
Returns the rank of the local task in the group associated with a communicator.

Some special info has been added for MPI APIs

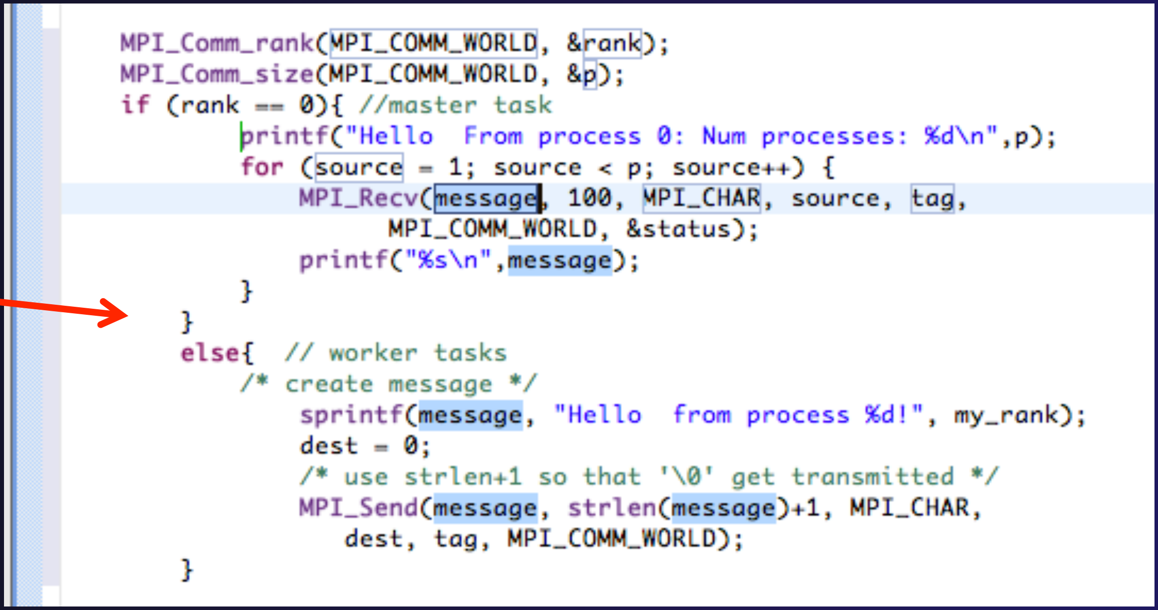
MPI Templates

- Allows quick entry of common patterns in MPI programming

- Example:
  MPI send-receive
- Enter:
  `mpisr` <ctrl-space>
- Expands to a send-receive pattern
- Highlighted variable names can all be changed at once
- Type `mpi` <ctrl-space> <ctrl-space> to see all templates



```
mpi
    mpiif - MPI_Init and Finalize
/*  mpisr - MPI Send Receive
```



```
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &p);
if (rank == 0){ //master task
        printf("Hello  From process 0: Num processes: %d\n",p);
        for (source = 1; source < p; source++) {
            MPI_Recv(message, 100, MPI_CHAR, source, tag,
                  MPI_COMM_WORLD, &status);
            printf("%s\n",message);
        }
    }
    else{  // worker tasks
        /* create message */
            sprintf(message, "Hello  from process %d!", my_rank);
            dest = 0;
            /* use strlen+1 so that '\0' get transmitted */
            MPI_Send(message, strlen(message)+1, MPI_CHAR,
               dest, tag, MPI_COMM_WORLD);
    }
```

Add more templates using Eclipse preferences!
C/C++>Editor>Templates
Extend to other common patterns

Running the Program

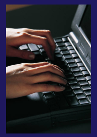
- Creating a resource manager
- Starting the resource manager
- Creating a launch configuration
- Launching the application
- Viewing the application run

Terminology

- The **Parallel Runtime** perspective is provided for monitoring and controlling applications
- Some terminology
  - **Resource manager** - Corresponds to an instance of a resource management system (e.g. a job scheduler). You can have multiple resource managers connected to different machines.
  - **Queue** - A queue of pending jobs
  - **Job** – A single run of a parallel application
  - **Machine** - A parallel computer system
  - **Node** - Some form of computational resource
  - **Process** - An execution unit (may be multiple threads of execution)

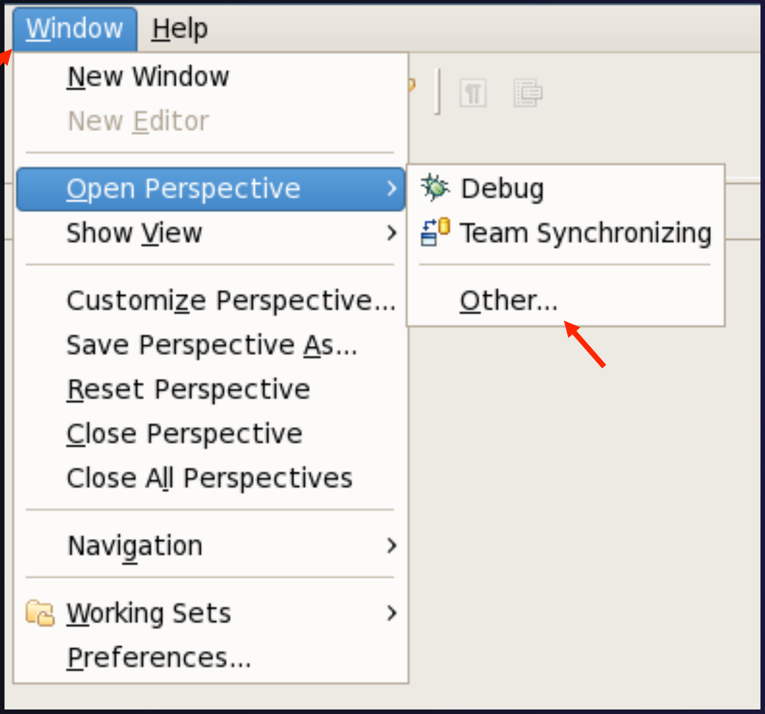
Resource Managers

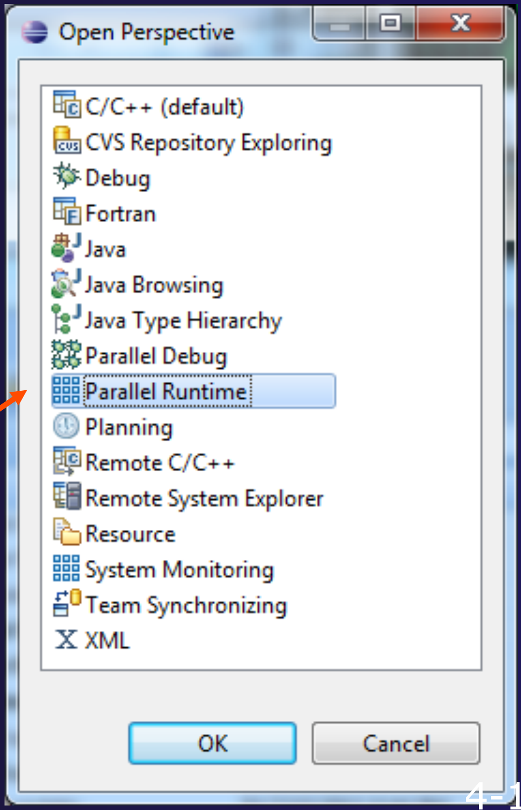
- PTP uses the term "resource manager" to refer to any subsystem that controls the resources required for launching a parallel job.
- Examples:
  - Job scheduler (e.g. LoadLeveler, PBS, SLURM)
  - Interactive execution (e.g. Open MPI, MPICH2, etc.)
- Each resource manager controls one target system
- Resource Managers can be local or remote
- Note: PTP 5.0 is in transition with respect to resource managers and status monitoring;
  - PBS ("jaxb lml") is new-style resource manager, with System Monitor runtime
  - All others are old-style resource managers, using Parallel Runtime



Preparing to Launch

- Setting up a resource manager is done in the Parallel Runtime perspective
- Select **Window>Open Perspective>Other**
- Choose **Parallel Runtime** and click **OK**





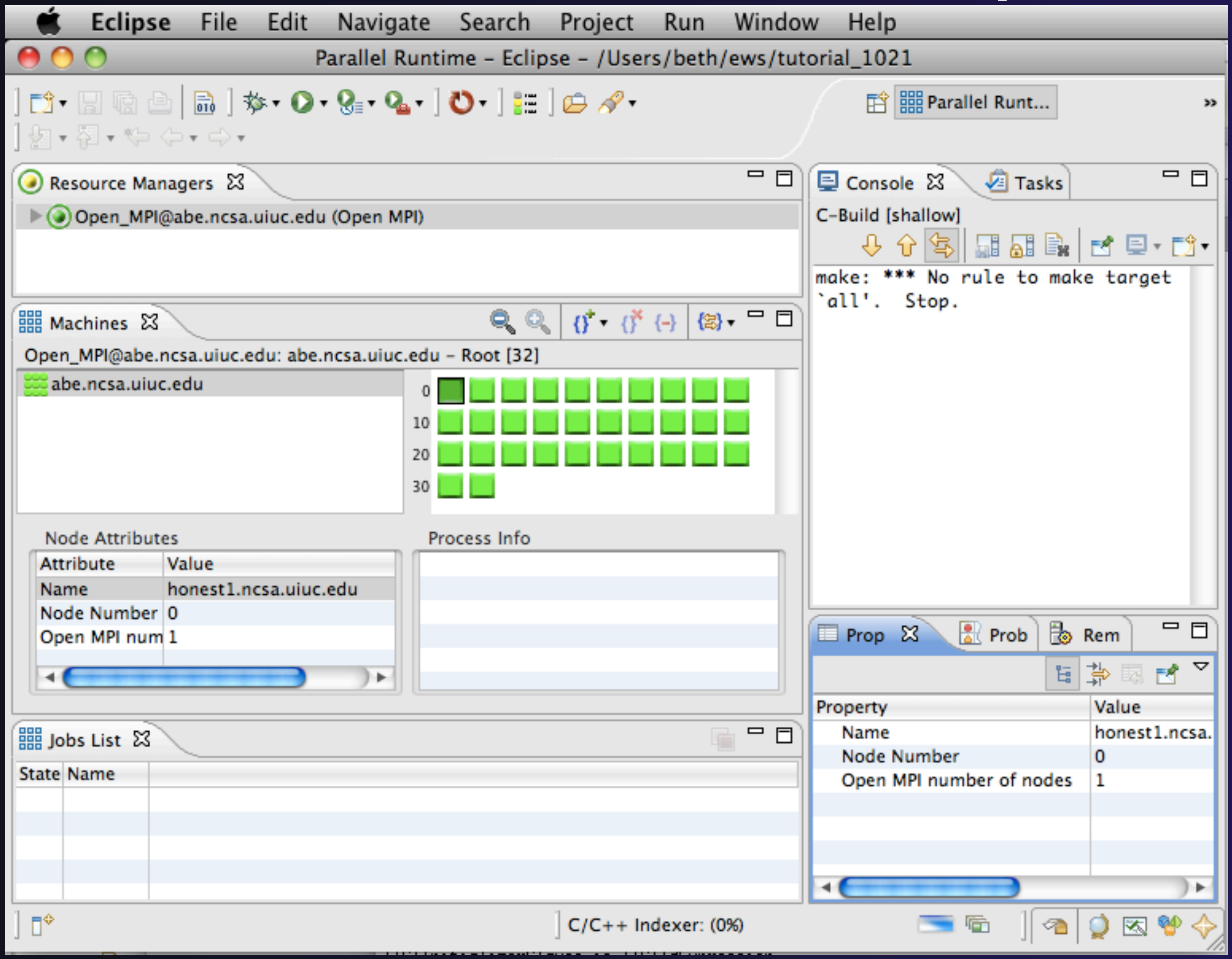
Parallel Runtime Perspective

Resource managers view

Machines view

Node details view

Jobs List view



```
make: *** No rule to make target
`all'.  Stop.
```

Attribute	Value
Name	honest1.ncsa.uiuc.edu
Node Number	0
Open MPI num	1

State	Name

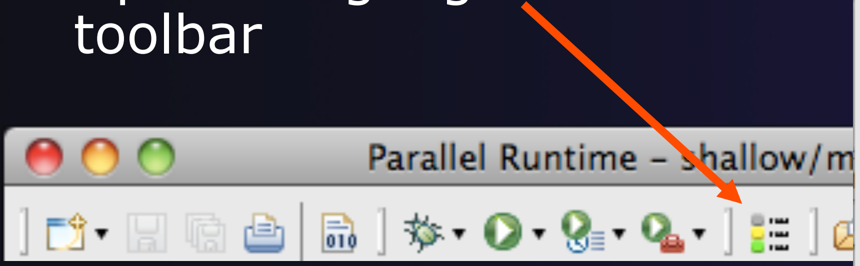
Property	Value
Name	honest1.ncsa.
Node Number	0
Open MPI number of nodes	1

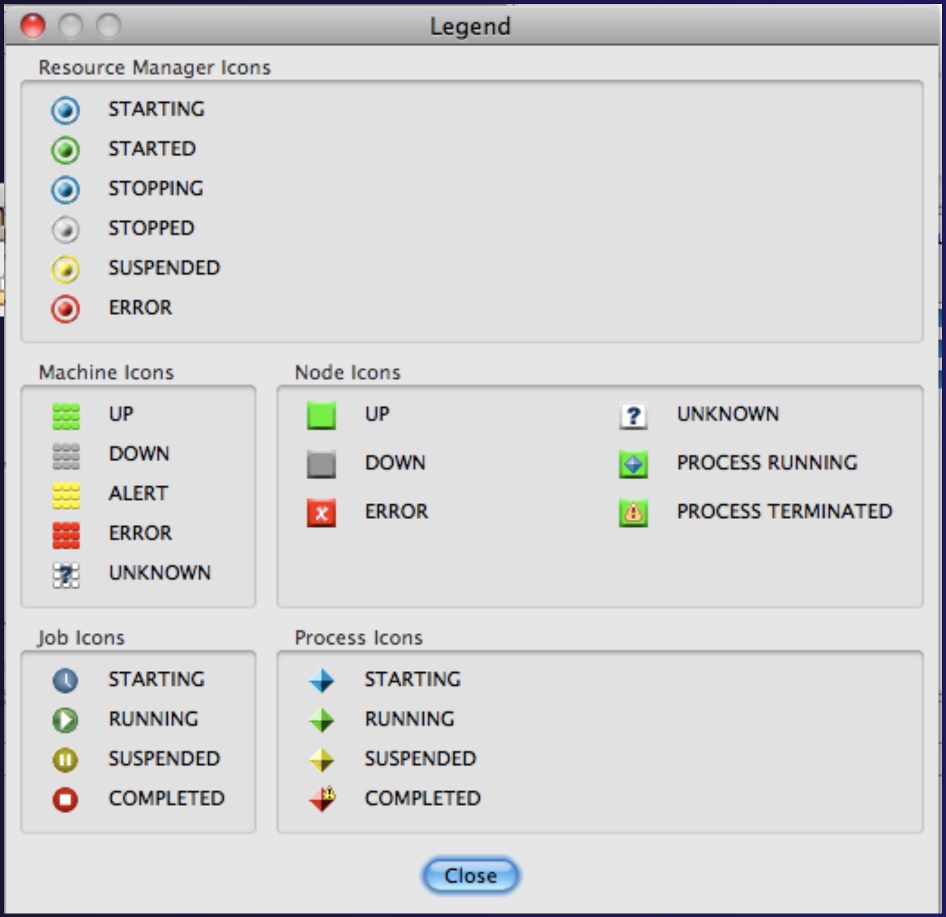
Console view

Properties view

About PTP Icons

- Open using legend icon in toolbar




Legend

Resource Manager Icons

🔵	STARTING
🟢	STARTED
🔵	STOPPING
⚪	STOPPED
🟡	SUSPENDED
🔴	ERROR

Machine Icons

🟩	UP
⬜	DOWN
🟨	ALERT
🟥	ERROR
?	UNKNOWN

Node Icons

🟩	UP	?	UNKNOWN
⬛	DOWN	🟩	PROCESS RUNNING
🟥	ERROR	⚠	PROCESS TERMINATED

Job Icons

🔵	STARTING
🟢	RUNNING
🟡	SUSPENDED
🔴	COMPLETED

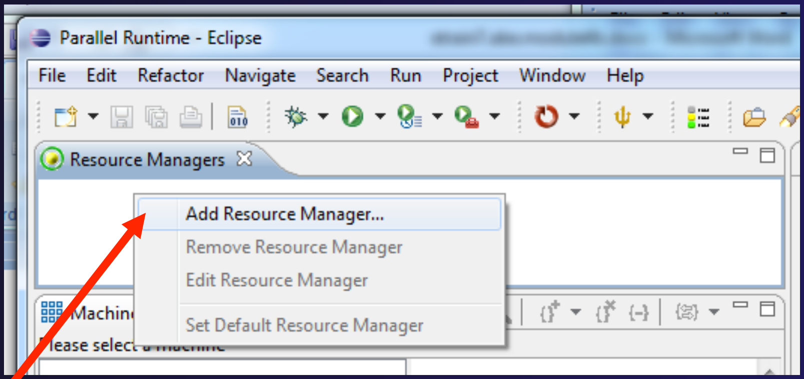
Process Icons

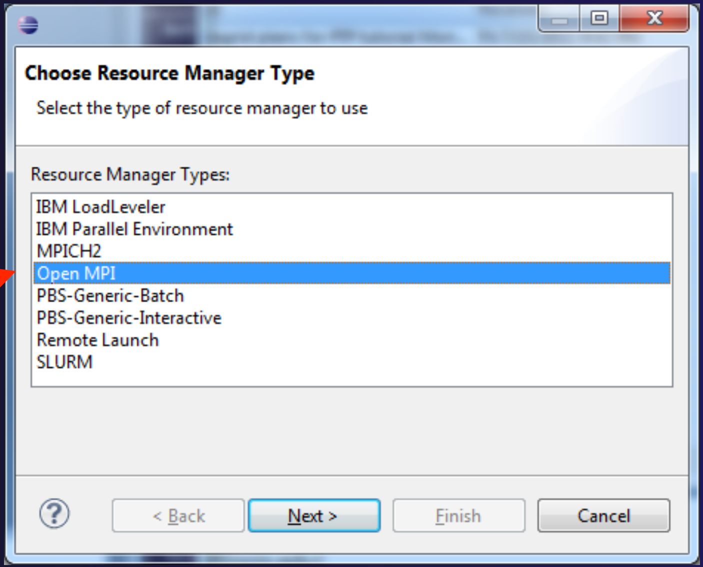
🔷	STARTING
◆	RUNNING
🔶	SUSPENDED
♦	COMPLETED

Close

Running Jobs Interactively

- Interactive resource managers will run the parallel application immediately
- They are also used for debugging the application
- Right-click in Resource Managers view and select **Add Resource Manager**
- Choose the **Open MPI Resource Manager** Type
- Select **Next>**



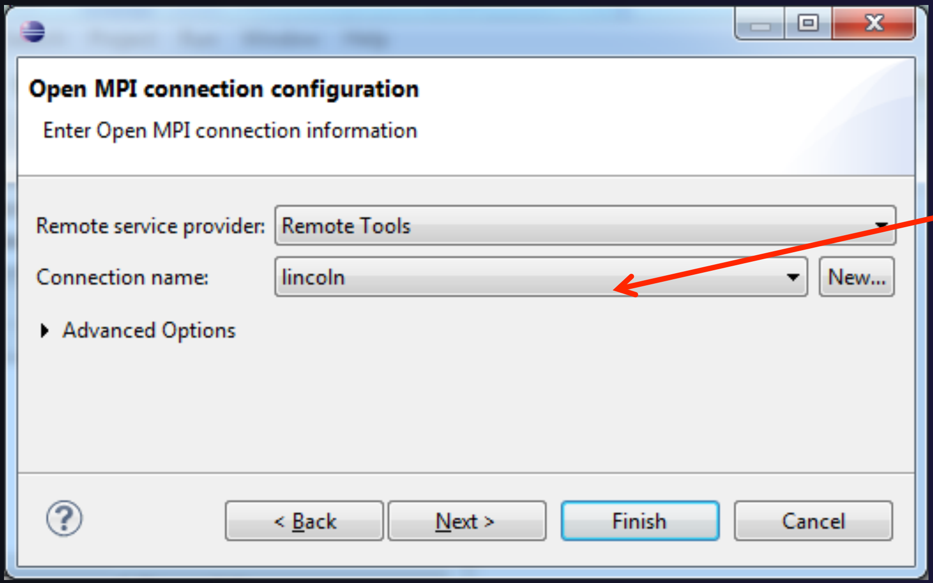

Choose Resource Manager Type

Select the type of resource manager to use

Resource Manager Types:

IBM LoadLeveler
IBM Parallel Environment
MPICH2
Open MPI
PBS-Generic-Batch
PBS-Generic-Interactive
Remote Launch
SLURM

Configure the Remote Location



Open MPI connection configuration

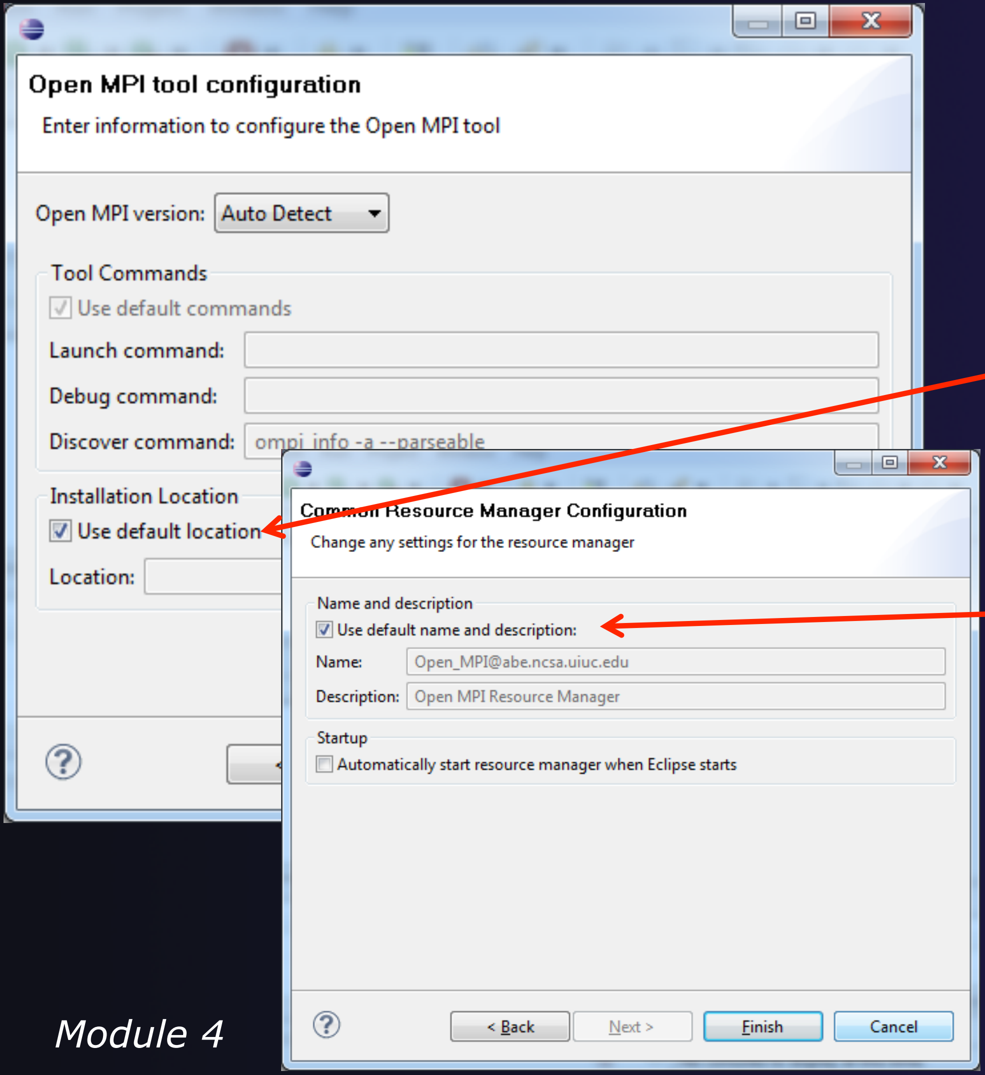
Enter Open MPI connection information

Remote service provider: Remote Tools

Connection name: lincoln

- Choose **Remote Tools** for **Remote service provider**
- Choose the remote connection you made previously
- Click **Next>**

Configure the Resource Manager



Open MPI tool configuration

Enter information to configure the Open MPI tool

Open MPI version: Auto Detect

Tool Commands

☑ Use default commands

Launch command:

Debug command:

Discover command: ompi_info -a --parseable

Installation Location

☑ Use default location

Location:

Common Resource Manager Configuration

Change any settings for the resource manager

Name and description

☑ Use default name and description:

Name: Open_MPI@abe.ncsa.uiuc.edu

Description: Open MPI Resource Manager

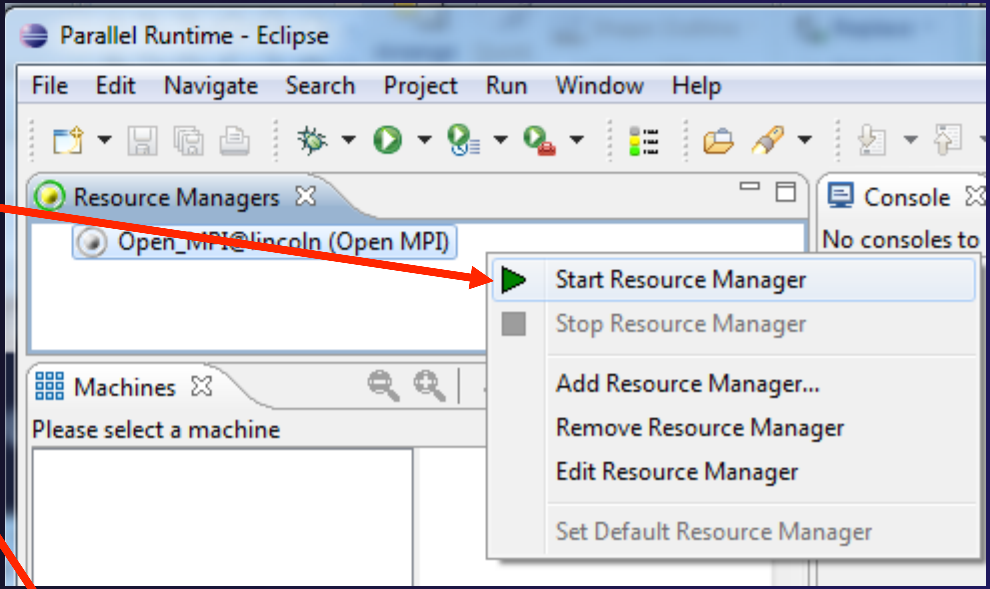
Startup

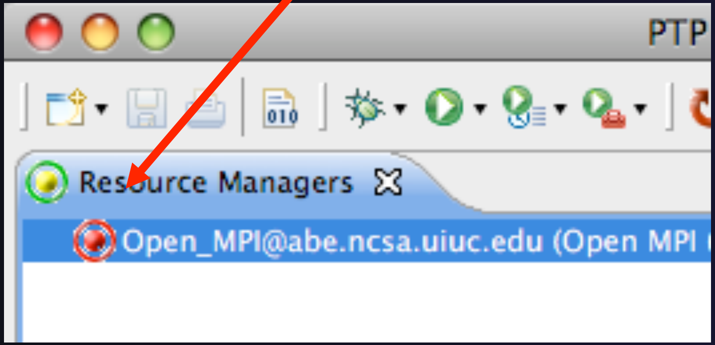
☐ Automatically start resource manager when Eclipse starts

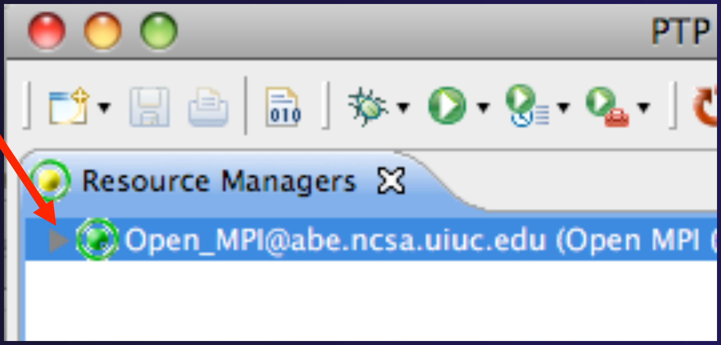
- The Open MPI resource manager will auto detect the version and use the appropriate commands
  - Change only if you're an expert
- Set the location of the "mpirun" command if it is not in your path
- Click **Next>**
- Change the **Name** or **Description** of the resource manager if you wish
- You can also set the resource manager to automatically start
- Click **Finish**

Starting the Resource Manager

- Right click on new resource manager and select **Start resource manager**
- If everything is ok, you should see the resource manager change to green
- If something goes wrong, it will change to red

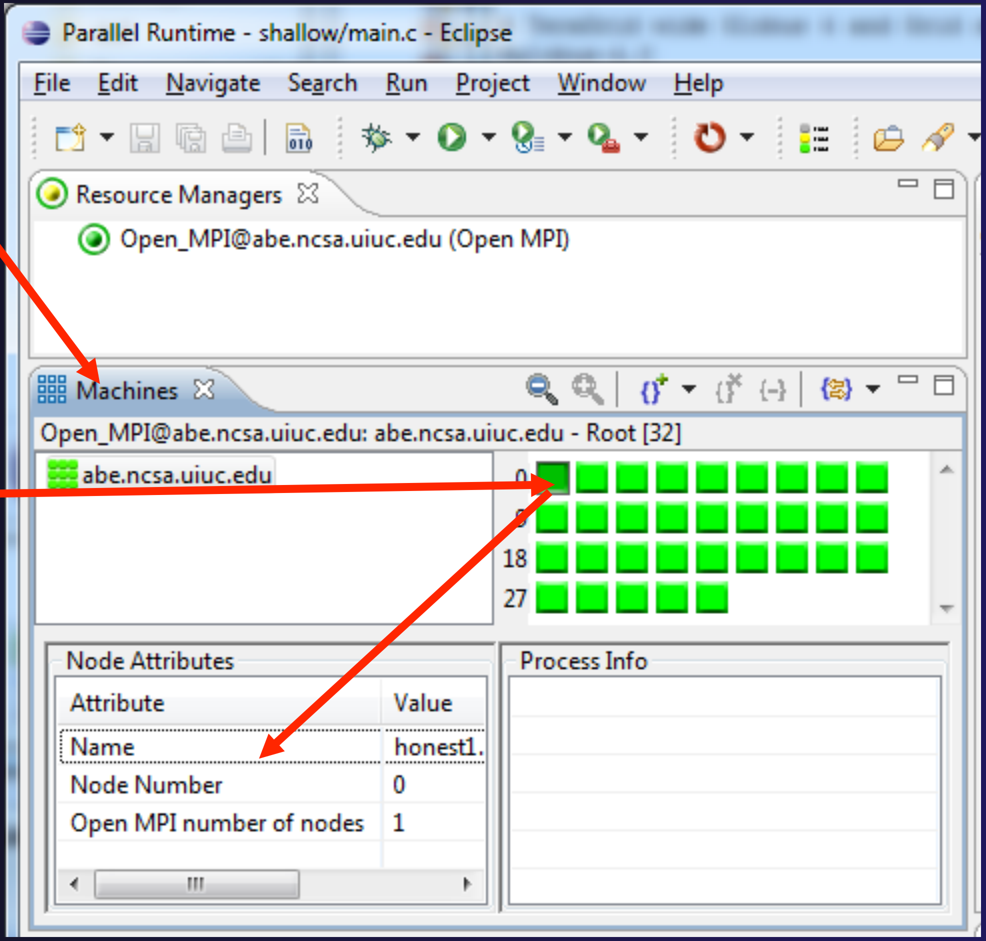






System Monitoring

- Machine status shown in **Machines** view
- Node status also shown **Machines** view
- Hover over node to see node name
- Double-click on node to show attributes


Parallel Runtime - shallow/main.c - Eclipse
File Edit Navigate Search Run Project Window Help
Resource Managers
Machines
Node Attributes
Process Info

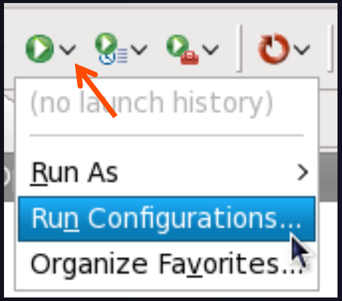
Open_MPI@abe.ncsa.uiuc.edu (Open MPI)

Open_MPI@abe.ncsa.uiuc.edu: abe.ncsa.uiuc.edu - Root [32]

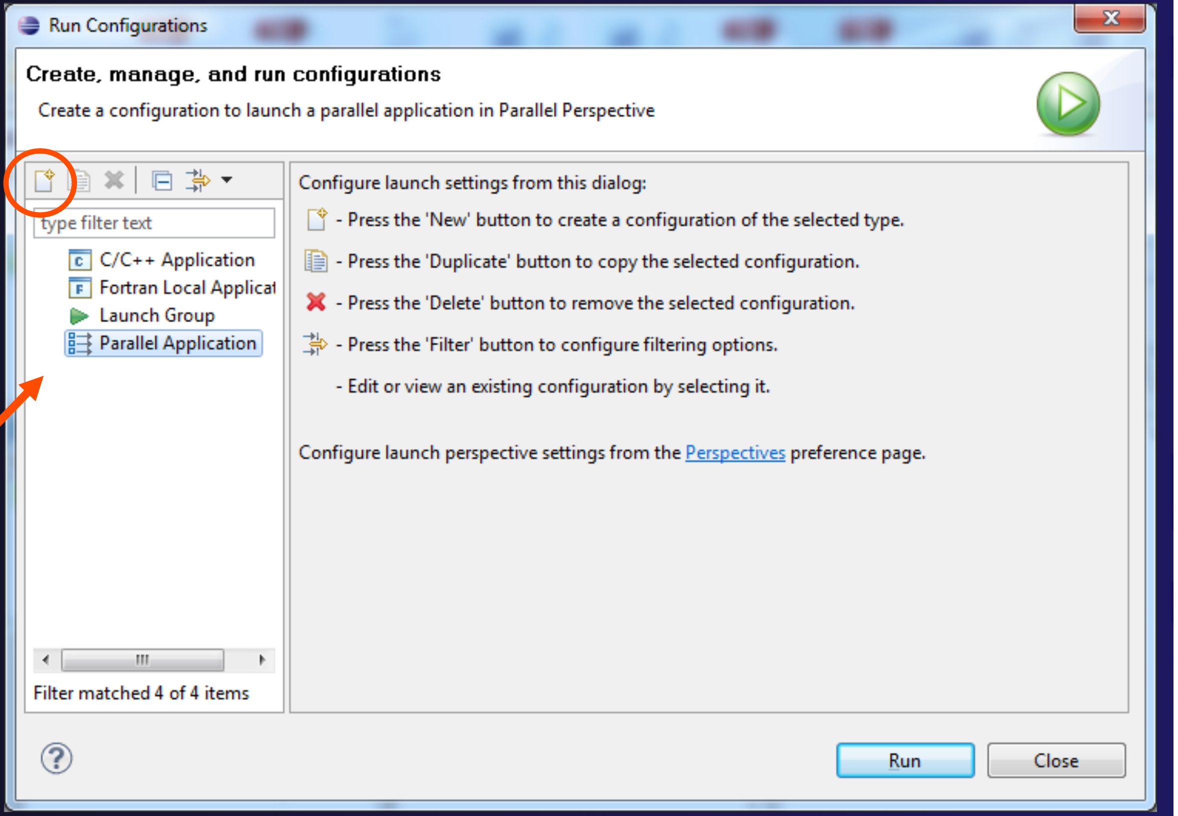
abe.ncsa.uiuc.edu

Attribute	Value
Name	honest1.
Node Number	0
Open MPI number of nodes	1

Create a Launch Configuration



- Open the run configuration dialog **Run>Run Configurations…**
- Select **Parallel Application**
- Select the **New** button


Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

type filter text

- C/C++ Application
- Fortran Local Applicat
- Launch Group
- Parallel Application

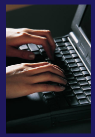
Filter matched 4 of 4 items

Configure launch settings from this dialog:

- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

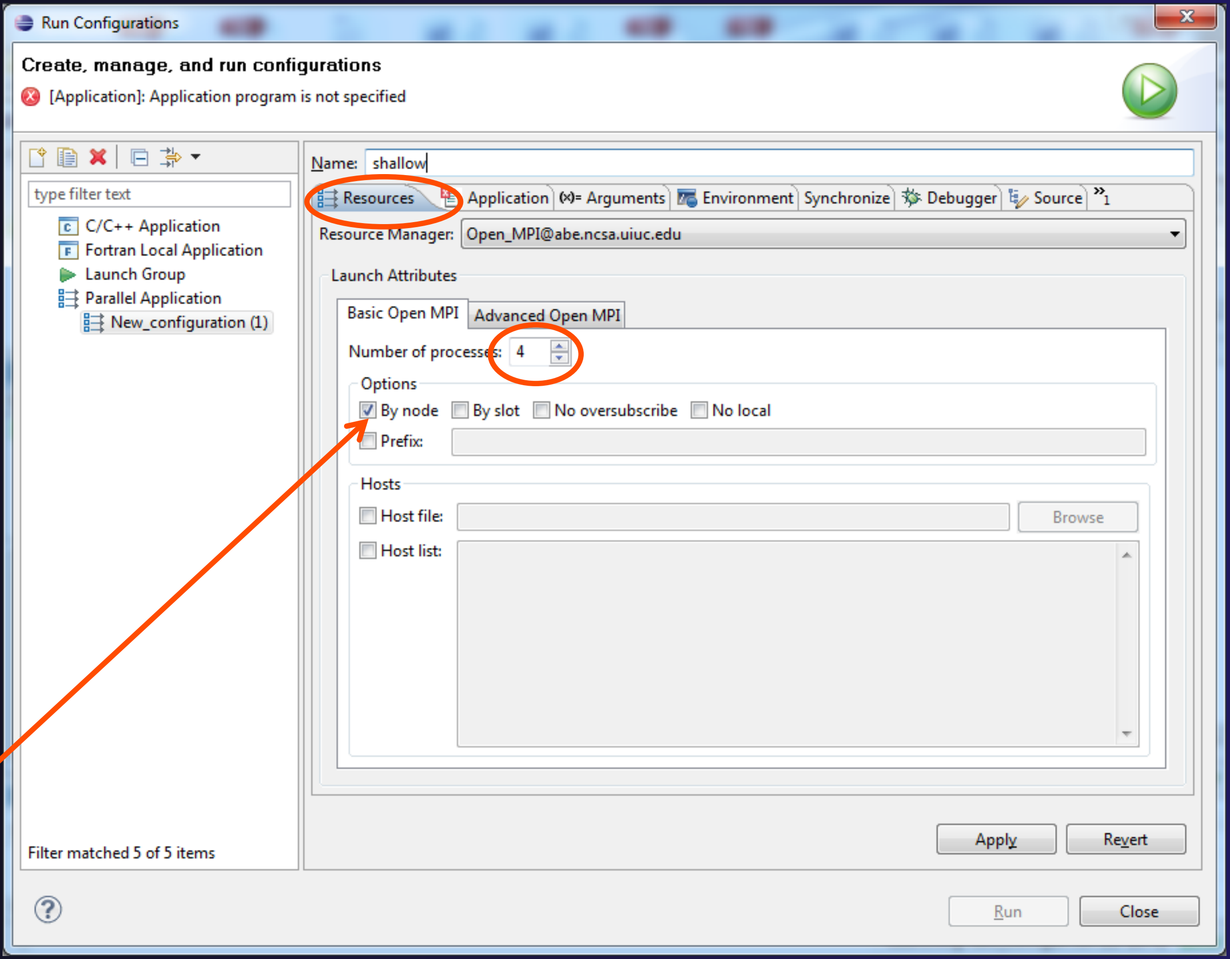
Configure launch perspective settings from the Perspectives preference page.

Depending on which flavor of Eclipse you installed, you might have more choices in Application types



Complete the Resources Tab

- Enter a name for the launch configuration, e.g. "shallow"
- In **Resources** tab, select the resource manager you want to use to launch this job
- Enter a value in the **Number of processes** field
- Other fields can be used to specify resource manager-specific information
  - E.g. specify **By node** to allocate each process to a different node



Run Configurations

Create, manage, and run configurations

[Application]: Application program is not specified

Name: shallow

Resource Manager: Open_MPI@abe.ncsa.uiuc.edu

Launch Attributes

Number of processes: 4

Options

[✓] By node [ ] By slot [ ] No oversubscribe [ ] No local

[ ] Prefix:

Hosts

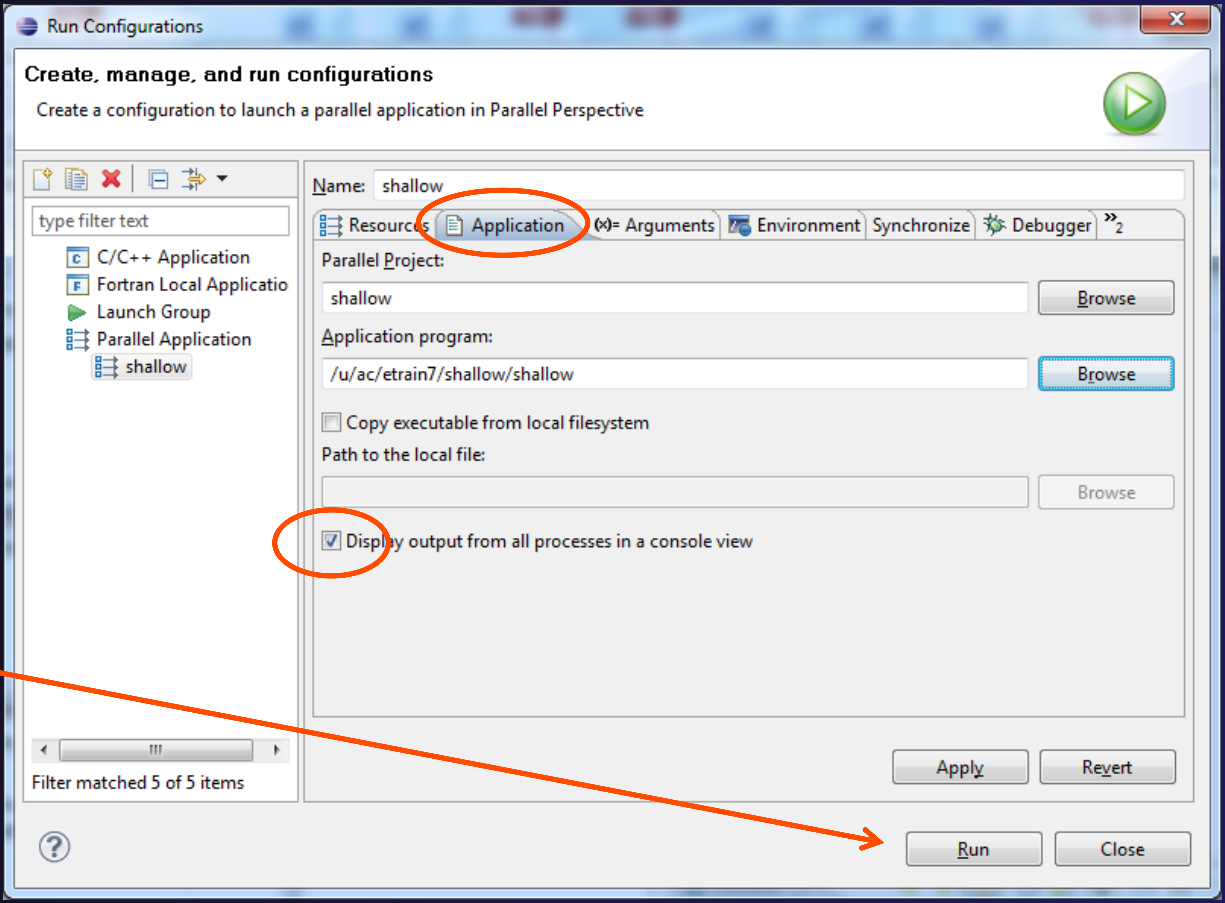
[ ] Host file:

[ ] Host list:

Filter matched 5 of 5 items

Complete the Application Tab

- Select the **Application** tab
- Choose the **Application program** by clicking the **Browse** button and locating the executable on the remote machine
  - There should be a "shallow" executable in the "shallow" directory
- Select **Display output from all processes in a console view**
- Click **Run** to run the application


Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

Name: shallow

Parallel Project: shallow

Application program: /u/ac/etrain7/shallow/shallow

[ ] Copy executable from local filesystem

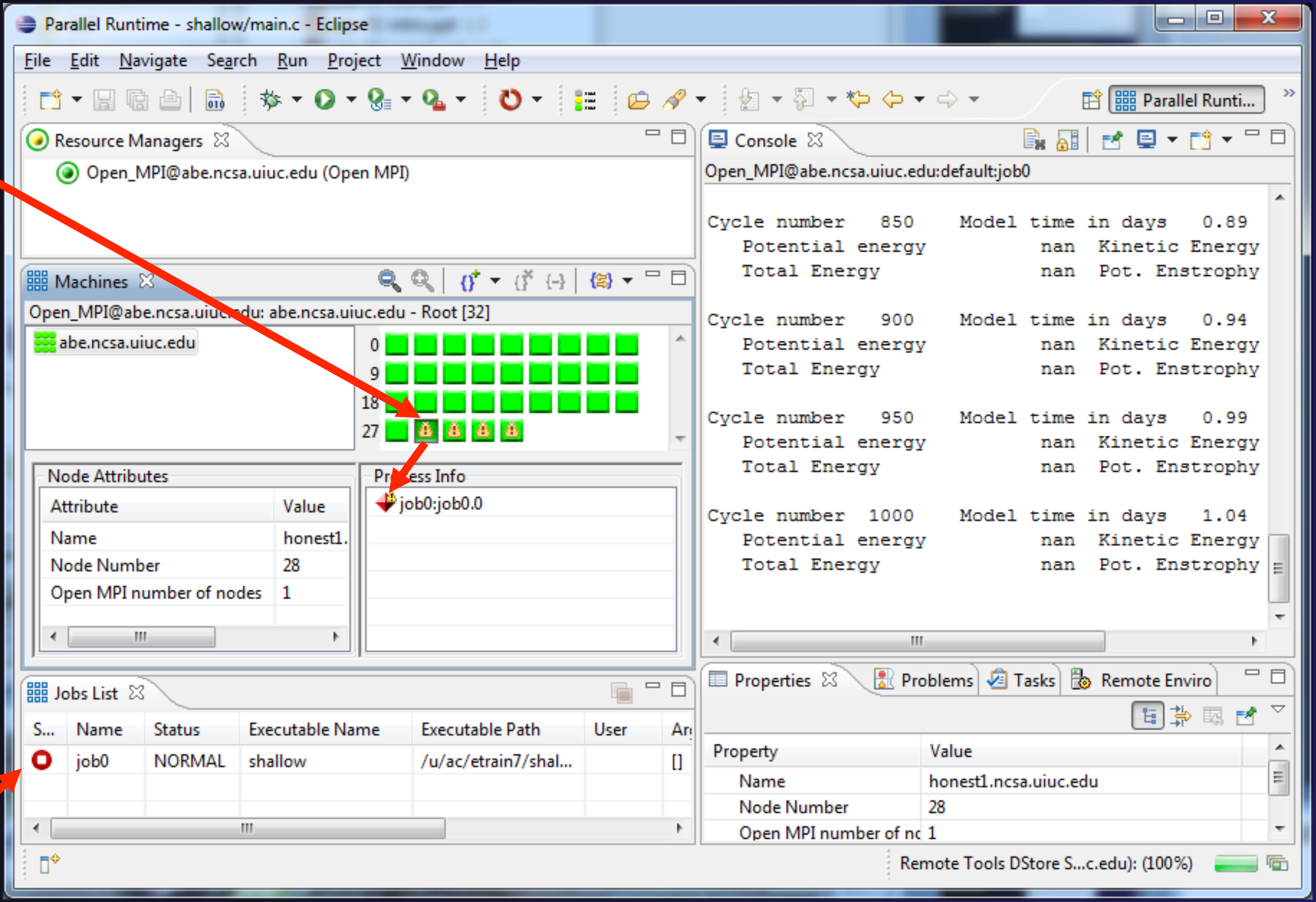
Path to the local file:

[✓] Display output from all processes in a console view

Filter matched 5 of 5 items

Viewing The Run

- Double-click a node in machines view to see which processes ran on the node
- Hover over a process for tooltip popup
- Job status and information

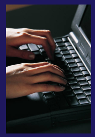

Parallel Runtime - shallow/main.c - Eclipse
File Edit Navigate Search Run Project Window Help
Parallel Runti...
Resource Managers
Open_MPI@abe.ncsa.uiuc.edu (Open MPI)
Machines
Open_MPI@abe.ncsa.uiuc.edu: abe.ncsa.uiuc.edu - Root [32]
abe.ncsa.uiuc.edu
0
9
18
27
Node Attributes
Process Info
job0:job0.0
Jobs List
Console
Open_MPI@abe.ncsa.uiuc.edu:default:job0
Properties
Problems
Tasks
Remote Enviro
Remote Tools DStore S...c.edu): (100%)

Attribute	Value
Name	honest1.
Node Number	28
Open MPI number of nodes	1

S...	Name	Status	Executable Name	Executable Path	User	Ar
	job0	NORMAL	shallow	/u/ac/etrain7/shal...		[]

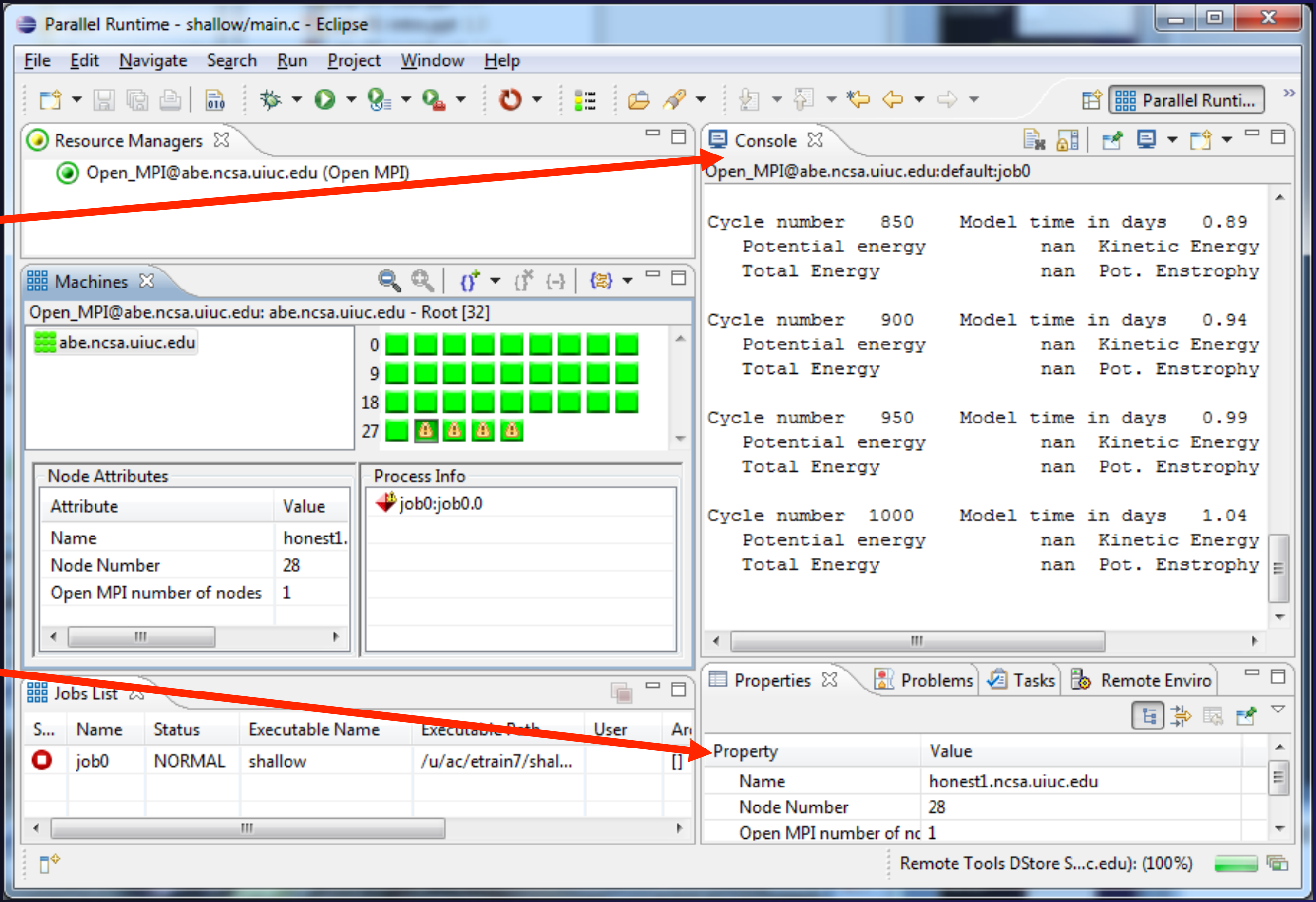
```
Cycle number   850    Model time in days   0.89
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number   900    Model time in days   0.94
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number   950    Model time in days   0.99
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number  1000    Model time in days   1.04
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy
```

Property	Value
Name	honest1.ncsa.uiuc.edu
Node Number	28
Open MPI number of nc	1



Viewing Program Output

- Console displays combined output from all processes
- Properties view shows job details


Parallel Runtime - shallow/main.c - Eclipse
File Edit Navigate Search Run Project Window Help
Parallel Runti...
Resource Managers
Open_MPI@abe.ncsa.uiuc.edu (Open MPI)
Machines
Open_MPI@abe.ncsa.uiuc.edu: abe.ncsa.uiuc.edu - Root [32]
abe.ncsa.uiuc.edu
0
9
18
27
Node Attributes
Process Info
job0:job0.0
Jobs List
Console
Properties
Problems
Tasks
Remote Enviro
Remote Tools DStore S...c.edu): (100%)

Attribute	Value
Name	honest1.
Node Number	28
Open MPI number of nodes	1

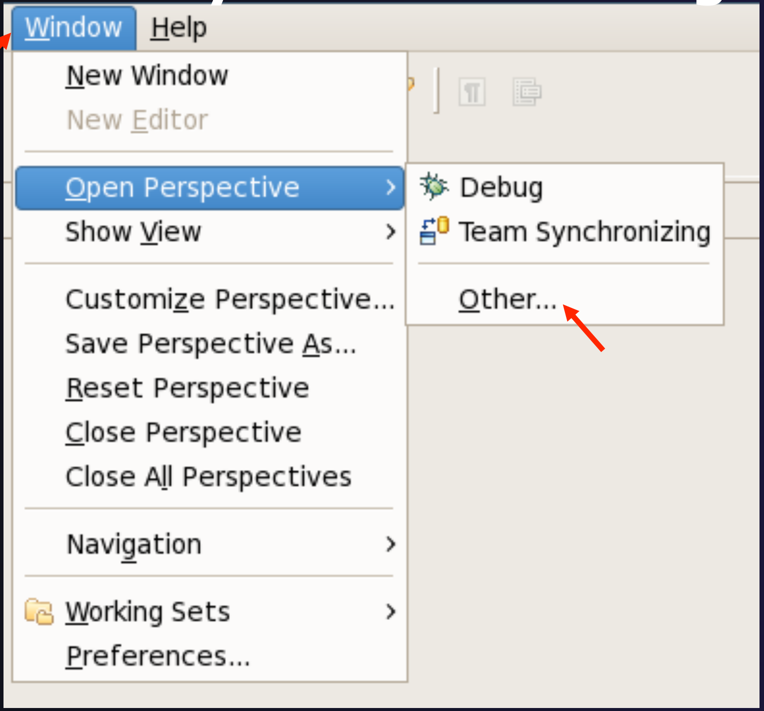
S...	Name	Status	Executable Name	Executable Path	User	Ar
●	job0	NORMAL	shallow	/u/ac/etrain7/shal...		[]

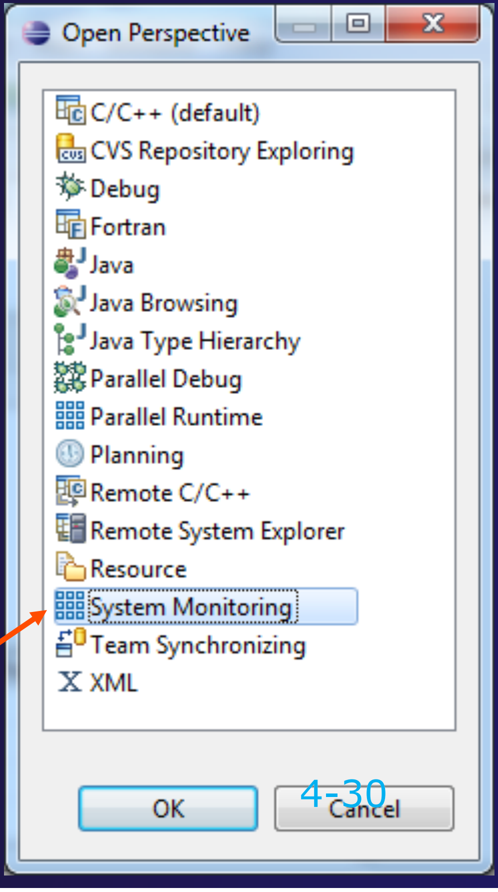
```
Open_MPI@abe.ncsa.uiuc.edu:default:job0

Cycle number   850    Model time in days   0.89
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number   900    Model time in days   0.94
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number   950    Model time in days   0.99
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy

Cycle number  1000    Model time in days   1.04
   Potential energy          nan  Kinetic Energy
   Total Energy              nan  Pot. Enstrophy
```

Property	Value
Name	honest1.ncsa.uiuc.edu
Node Number	28
Open MPI number of nc	1

Using a Job Scheduler

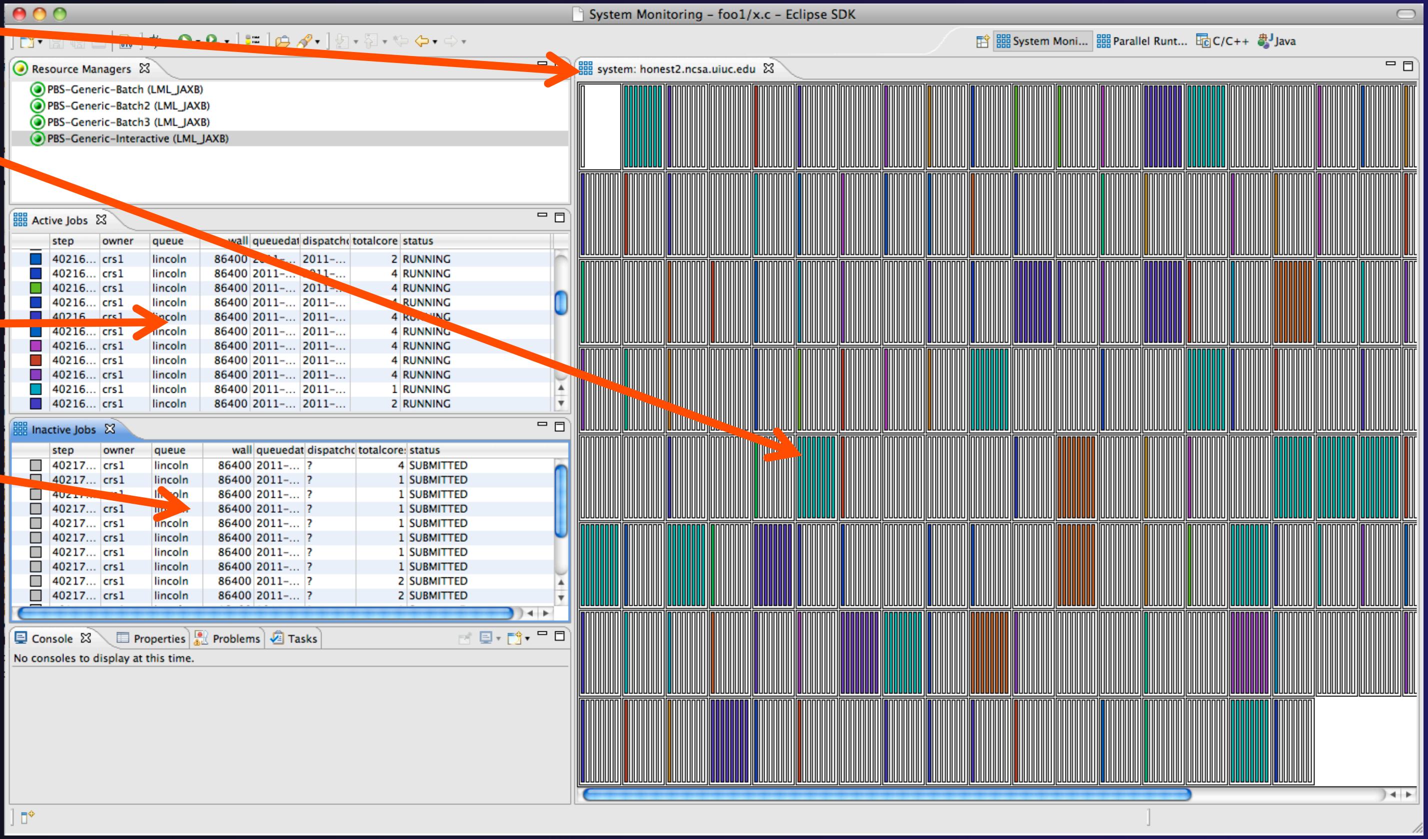
- Setting up a resource manager is done in the System Monitoring perspective
  - (For PTP 5.0.0, this applies to PBS)
- Select **Window>Open Perspective>Other**
- Choose **System Monitoring** and click **OK**





System Monitoring Perspective

- System view
- Jobs running on system
- Active jobs
- Inactive jobs



PBS-Generic-Batch (LML_JAXB)
PBS-Generic-Batch2 (LML_JAXB)
PBS-Generic-Batch3 (LML_JAXB)
PBS-Generic-Interactive (LML_JAXB)

Active Jobs

	step	owner	queue	wall	queuedat	dispatchd	totalcore	status
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	2	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟩	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟪	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟥	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟪	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	1	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	2	RUNNING

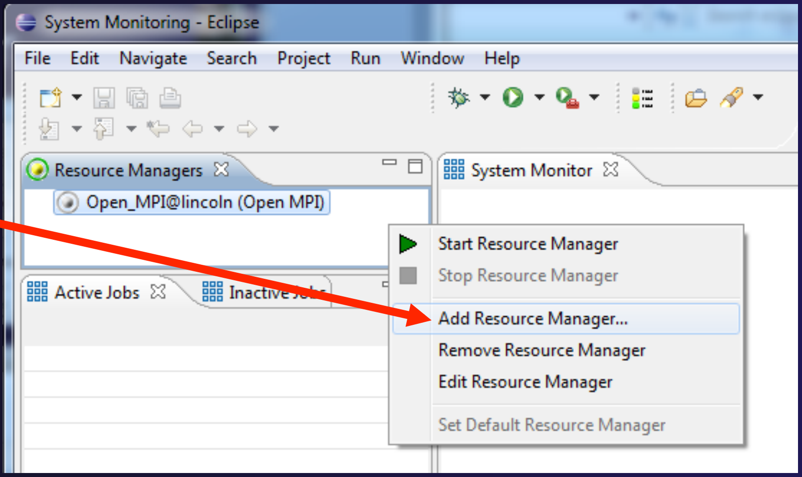
Inactive Jobs

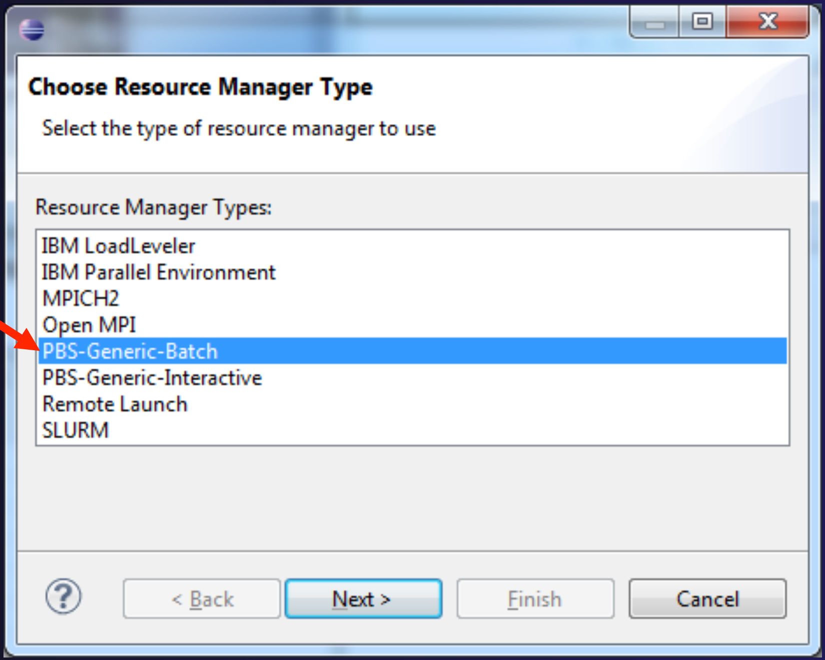
	step	owner	queue	wall	queuedat	dispatchd	totalcore:	status
⬜	40217...	crs1	lincoln	86400	2011-...	?	4	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	1	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	2	SUBMITTED
⬜	40217...	crs1	lincoln	86400	2011-...	?	2	SUBMITTED

No consoles to display at this time.

Using a Job Scheduler

- Right-click in Resource Managers view and select **Add Resource Manager**
- Choose the **PBS-Generic-Batch** Resource Manager Type
- Select **Next>**



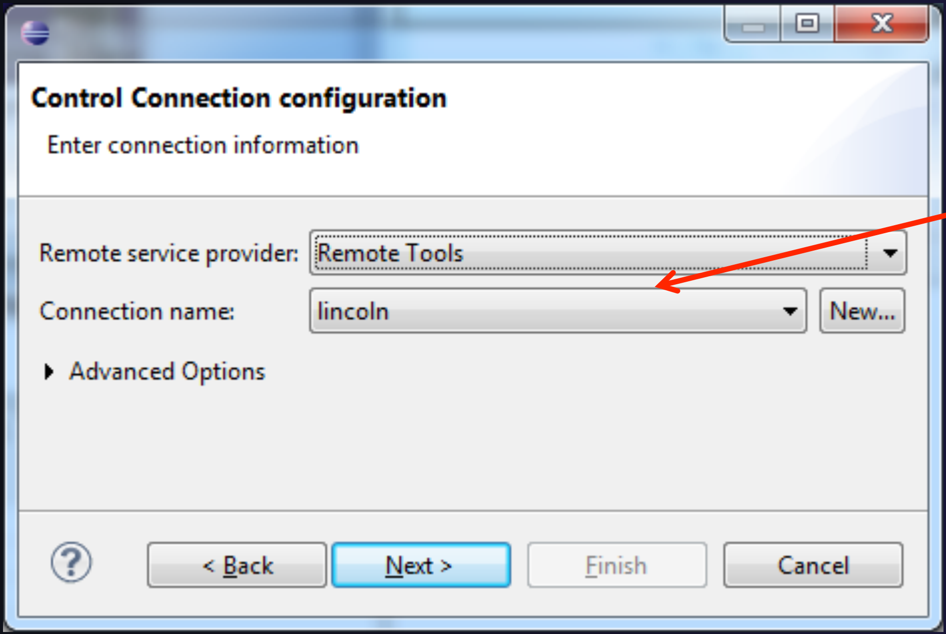

Choose Resource Manager Type

Select the type of resource manager to use

Resource Manager Types:

IBM LoadLeveler
IBM Parallel Environment
MPICH2
Open MPI
PBS-Generic-Batch
PBS-Generic-Interactive
Remote Launch
SLURM

Configure the Remote Location

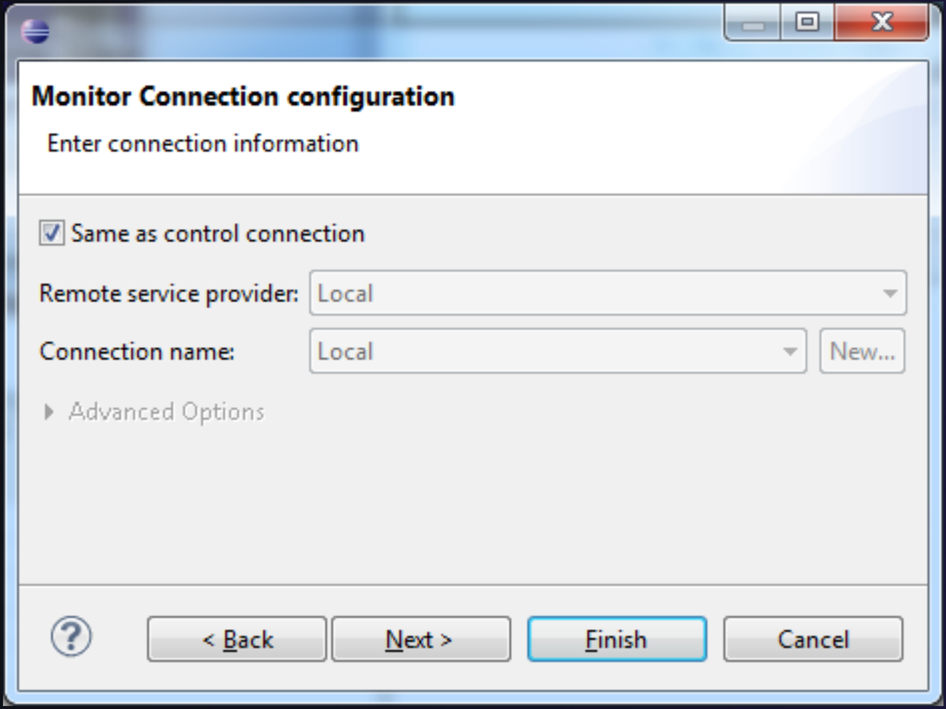

Control Connection configuration
Enter connection information
Advanced Options
New...
< Back
Next >
Finish
Cancel

Remote service provider: Remote Tools

Connection name: lincoln

- Choose **Remote Tools** for **Remote service provider**
- Choose the remote connection you made previously
- Click **Next>**

Configure the Monitor Connection


Monitor Connection configuration

Enter connection information

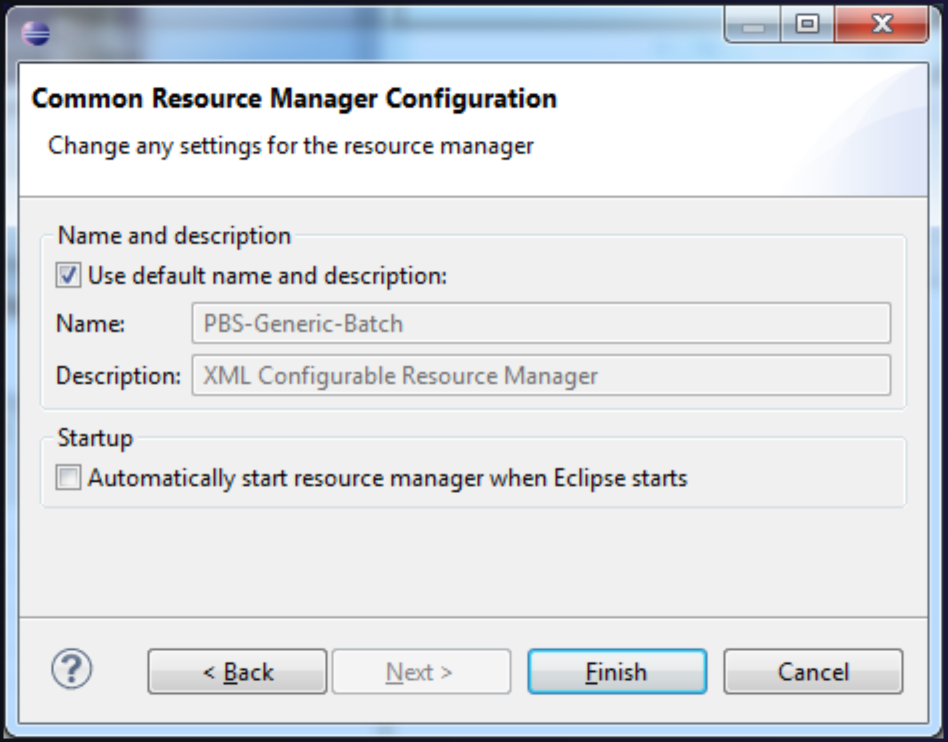
☑ Same as control connection

Remote service provider: Local

Connection name: Local

- Keep default Monitor Connection (same as Control Connection), click **Next**

Configure the Common Resource Manager Parameters



Common Resource Manager Configuration

Change any settings for the resource manager

Name and description

[✓] Use default name and description:

Name: PBS-Generic-Batch

Description: XML Configurable Resource Manager

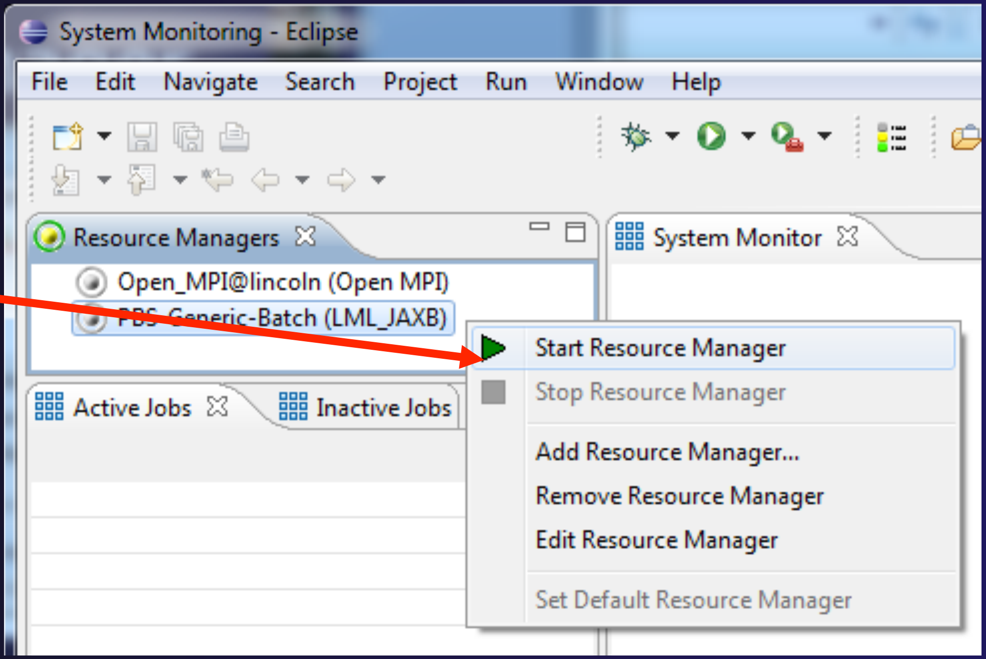
Startup

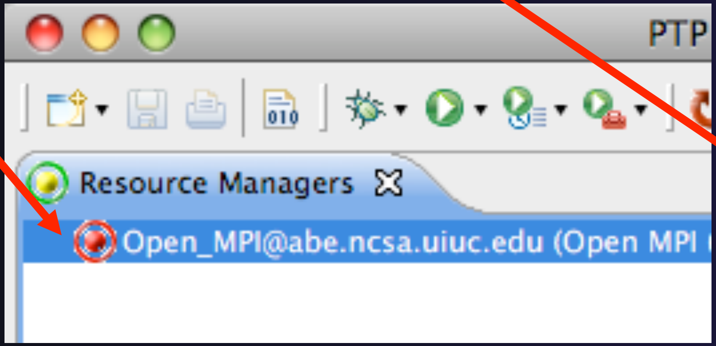
[ ] Automatically start resource manager when Eclipse starts

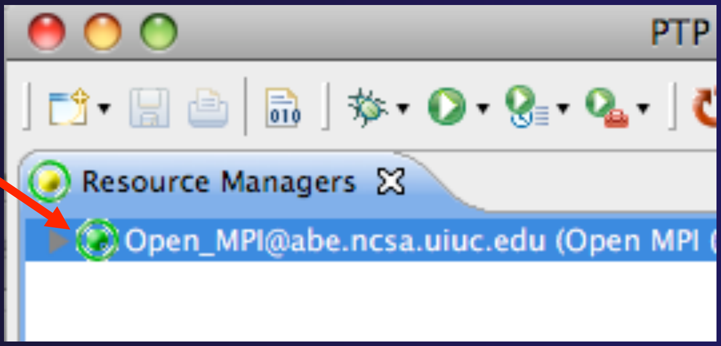
- Keep default name
- Can automatically start Resource Manager (leave unselected today)
- Click **Finish**

Starting the Resource Manager

- Right click on new resource manager and select **Start resource manager**
- If everything is ok, you should see the resource manager change to green
- If something goes wrong, it will change to red

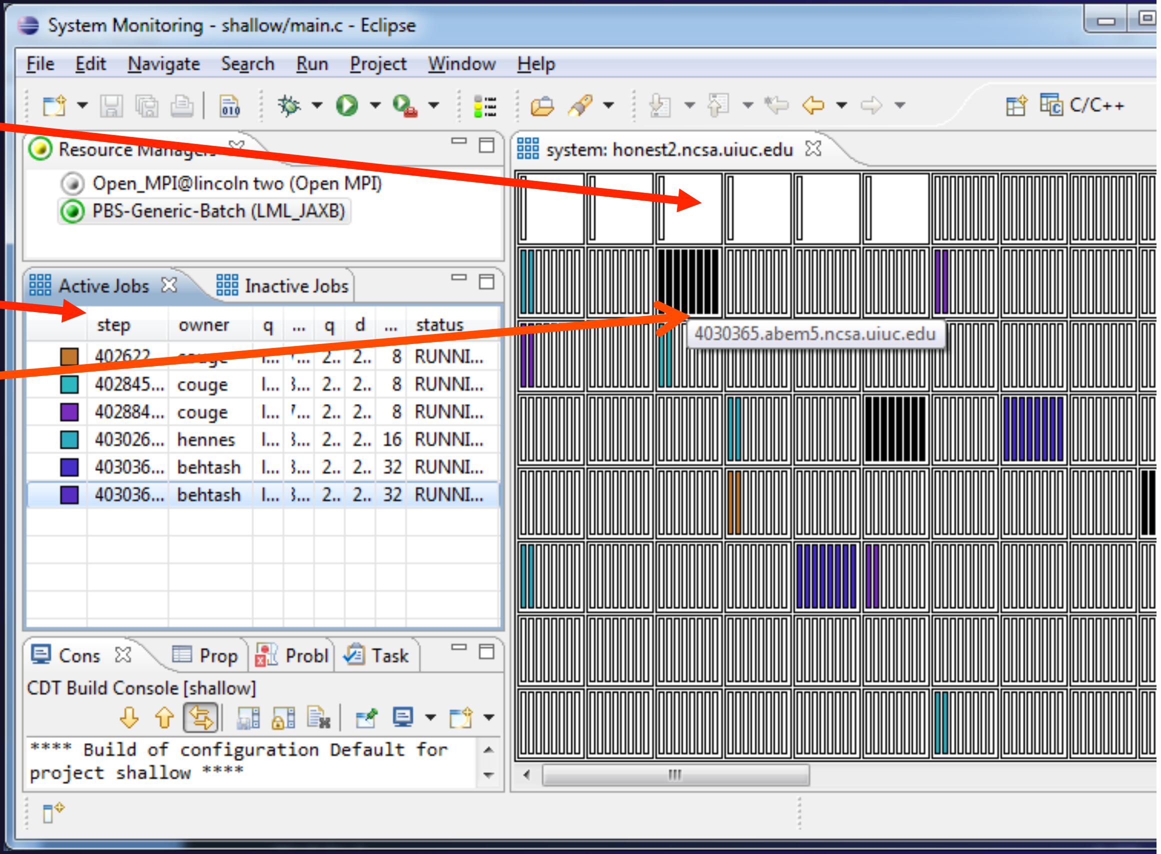






System Monitoring

- System view, with abstraction of nodes
- Active and inactive jobs
- Hover over node to see job running on node



System Monitoring - shallow/main.c - Eclipse

Resource Managers

Open_MPI@lincoln two (Open MPI)

PBS-Generic-Batch (LML_JAXB)

system: honest2.ncsa.uiuc.edu

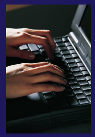
4030365.abem5.ncsa.uiuc.edu

Active Jobs | Inactive Jobs

	step	owner	q	...	q	d	...	status
🟧	402622...	couge	l...	'...	2..	2..	8	RUNNI...
🟦	402845...	couge	l...	3...	2..	2..	8	RUNNI...
🟪	402884...	couge	l...	7...	2..	2..	8	RUNNI...
🟦	403026...	hennes	l...	3...	2..	2..	16	RUNNI...
🟪	403036...	behtash	l...	3...	2..	2..	32	RUNNI...
🟪	403036...	behtash	l...	3...	2..	2..	32	RUNNI...

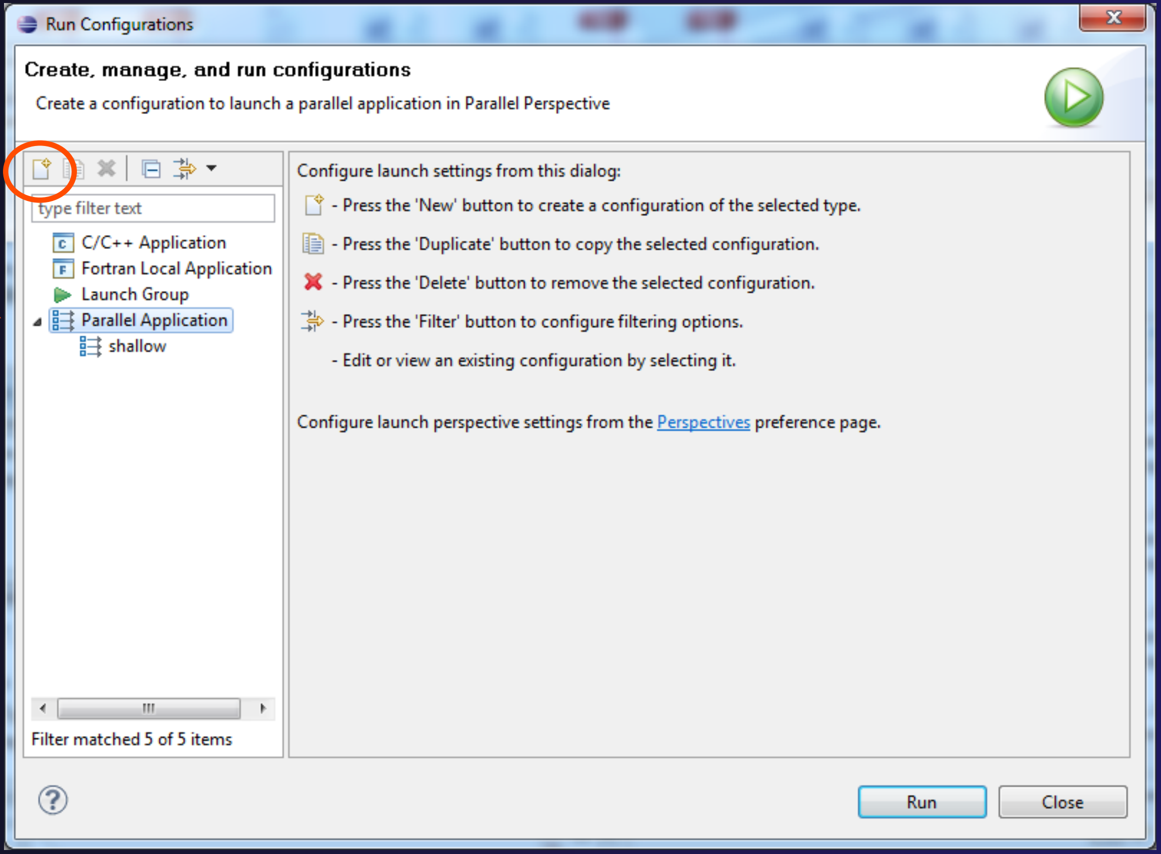
CDT Build Console [shallow]

```
**** Build of configuration Default for
project shallow ****
```



Create a Launch Configuration

- Open the run configuration dialog **Run>Run Configurations…**
- Select **Parallel Application**
- Select the **New** button



Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

Configure launch settings from this dialog:

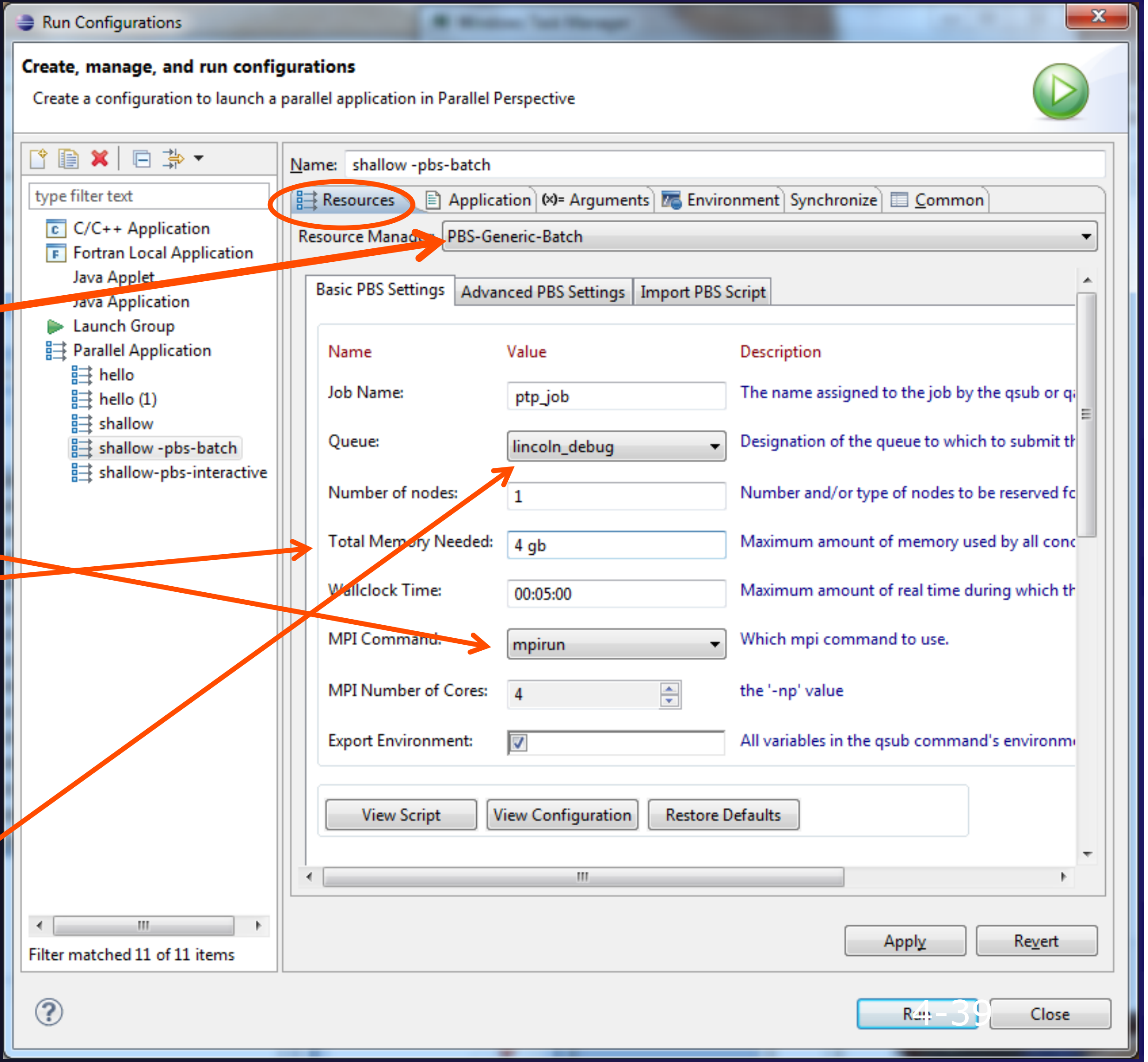
- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

Configure launch perspective settings from the Perspectives preference page.

Filter matched 5 of 5 items

Complete the Resources Tab

- Enter a name for this launch configuration, e.g. "shallow-pbs-batch
- Choose the appropriate Resource Manager (PBS-Generic-Batch)
- In **Resources** tab, select the PBS resource manager you just created
- The **MPI Command** field allows this job to be run as an MPI job
  - Choose **mpirun**
- Enter the resources needed to run this job
  - Use 1 nodes, 4 gb memory, 4 cores
- Select the destination queue – **lincoln_debug**



Name: shallow -pbs-batch

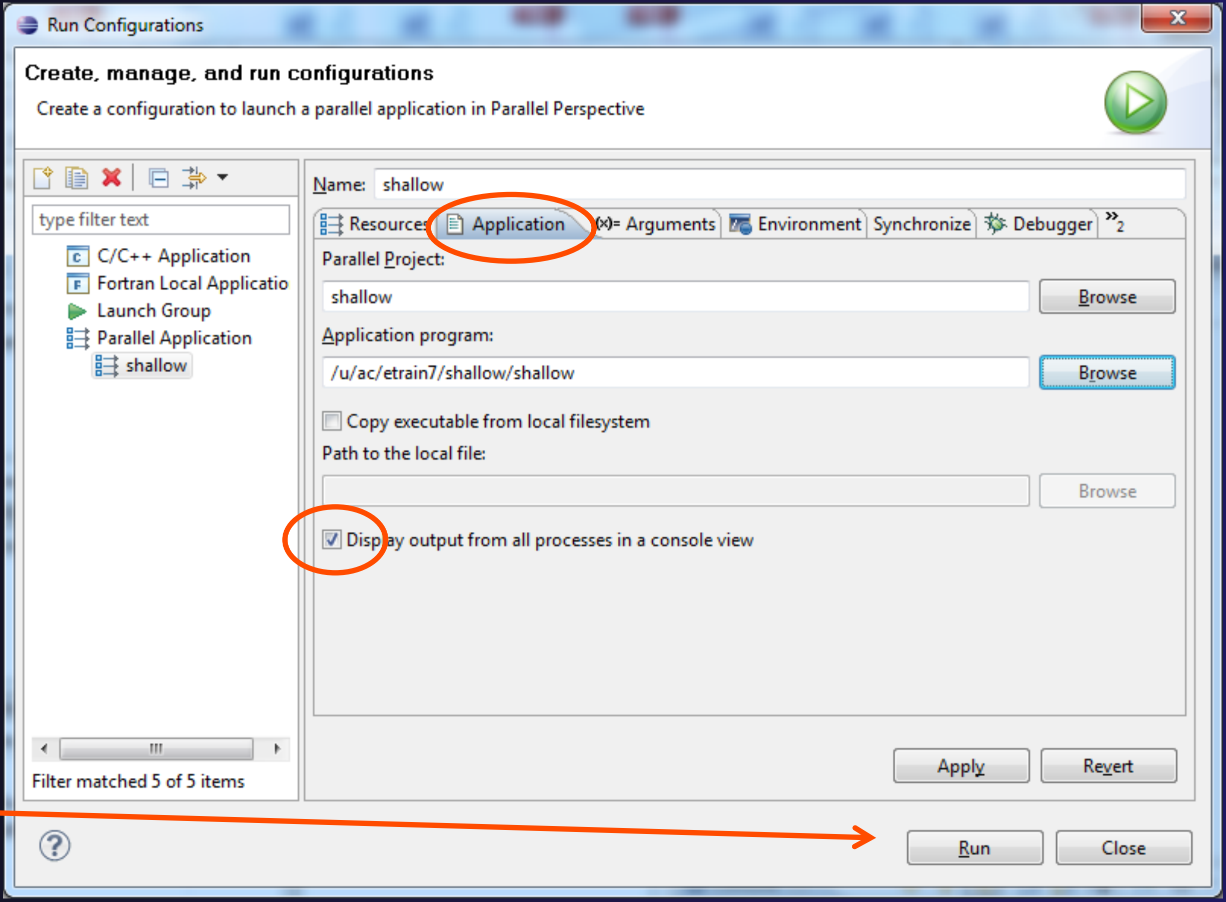
Resource Manager: PBS-Generic-Batch

Name	Value	Description
Job Name:	ptp_job	The name assigned to the job by the qsub or q
Queue:	lincoln_debug	Designation of the queue to which to submit th
Number of nodes:	1	Number and/or type of nodes to be reserved fo
Total Memory Needed:	4 gb	Maximum amount of memory used by all conc
Wallclock Time:	00:05:00	Maximum amount of real time during which th
MPI Command:	mpirun	Which mpi command to use.
MPI Number of Cores:	4	the '-np' value
Export Environment:	✓	All variables in the qsub command's environm

Module 4

Complete the Application Tab

- Select the **Application** tab
- Choose the **Application program** by clicking the **Browse** button and locating the executable on the remote machine
  - Use the same "shallow" executable
- Select **Display output from all processes in a console view**
- If Debugger tab has error, select Debugger: **SDM**
- Click **Run** to submit the application to the job scheduler



Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

Name: shallow

Parallel Project: shallow

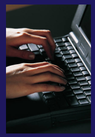
Application program: /u/ac/etrain7/shallow/shallow

[ ] Copy executable from local filesystem

Path to the local file:

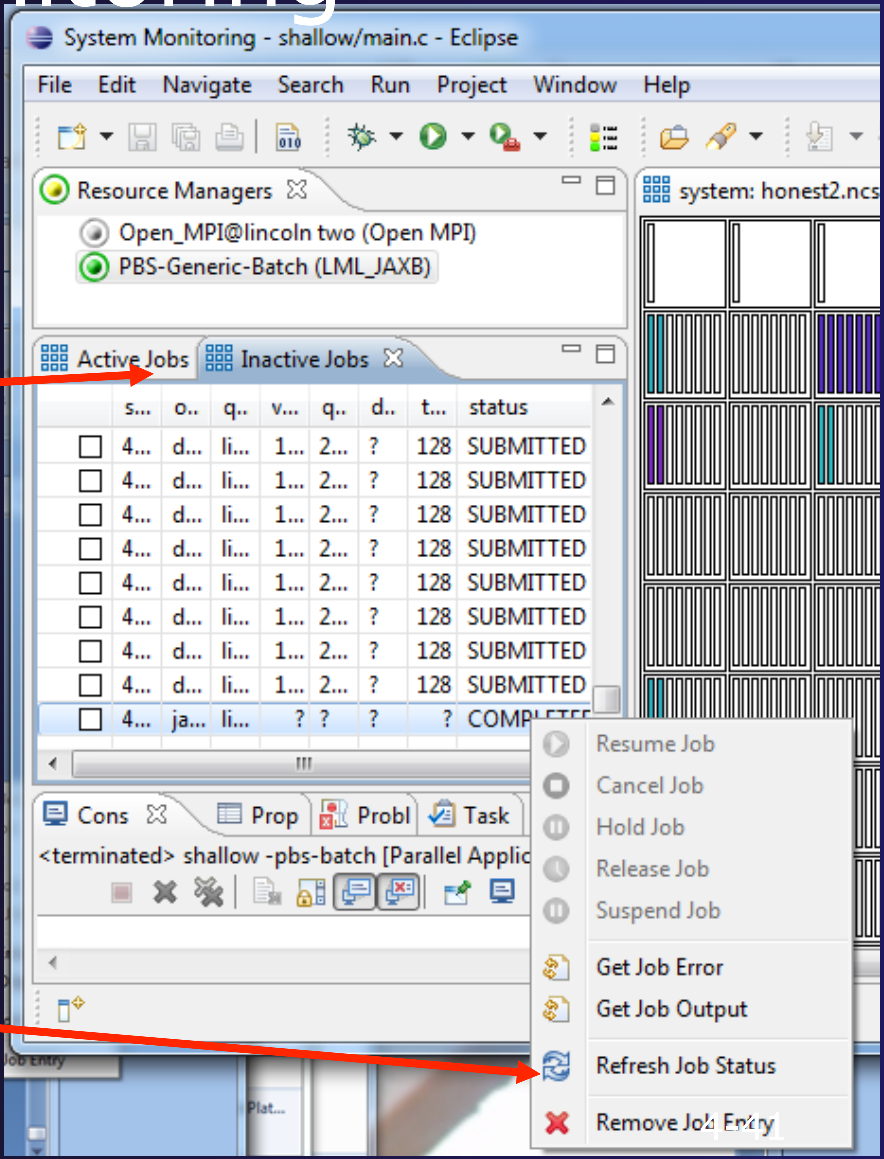
[✓] Display output from all processes in a console view

Filter matched 5 of 5 items



Job Monitoring

- Job initially appears in "Inactive Jobs", then in "Active Jobs", then returns to Inactive on completion
- Can view output or error by right clicking on job, selecting appropriate output

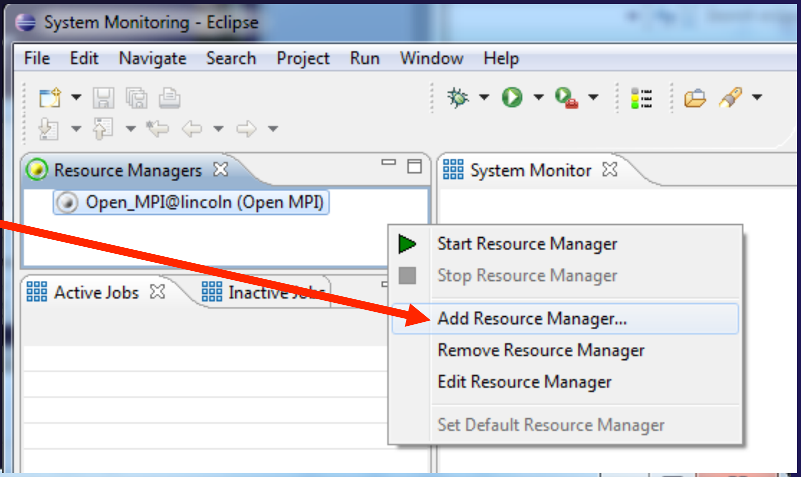


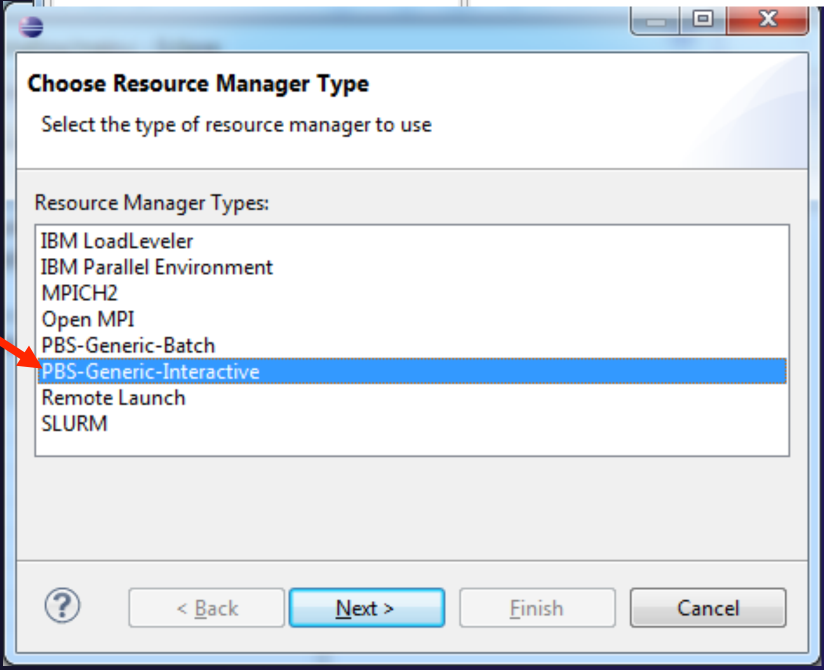
	s...	o..	q..	v...	q..	d..	t...	status
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	d...	li...	1...	2...	?	128	SUBMITTED
☐	4...	ja...	li...	?	?	?	?	COMPLETED

<terminated> shallow -pbs-batch [Parallel Applic

Interactive Job Scheduler

- Right-click in Resource Managers view and select **Add Resource Manager**
- Choose the **PBS-Generic-Interactive** Resource Manager Type
- Select **Next>**



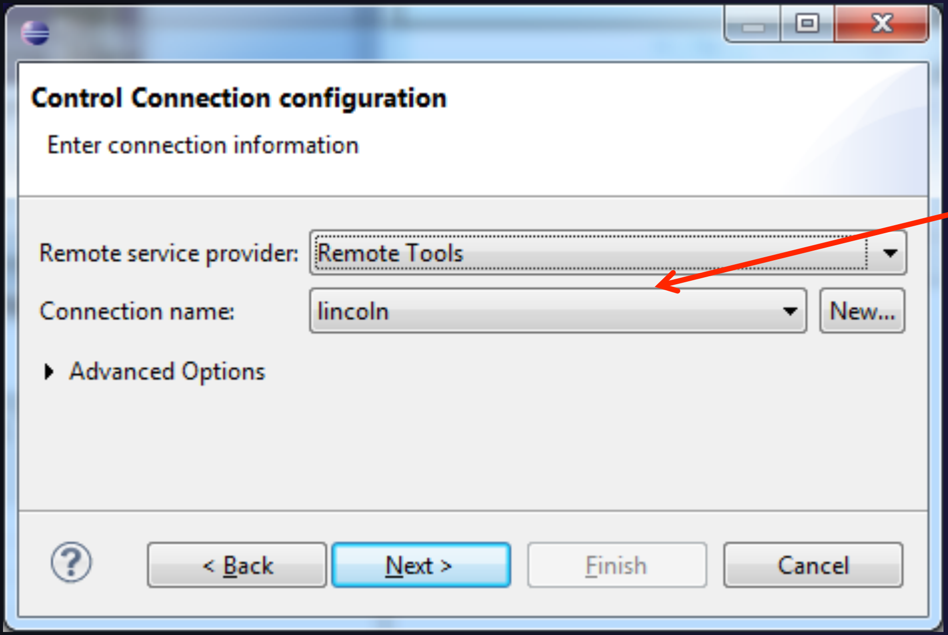

Choose Resource Manager Type

Select the type of resource manager to use

Resource Manager Types:

IBM LoadLeveler
IBM Parallel Environment
MPICH2
Open MPI
PBS-Generic-Batch
PBS-Generic-Interactive
Remote Launch
SLURM

Configure the Remote Location


Control Connection configuration

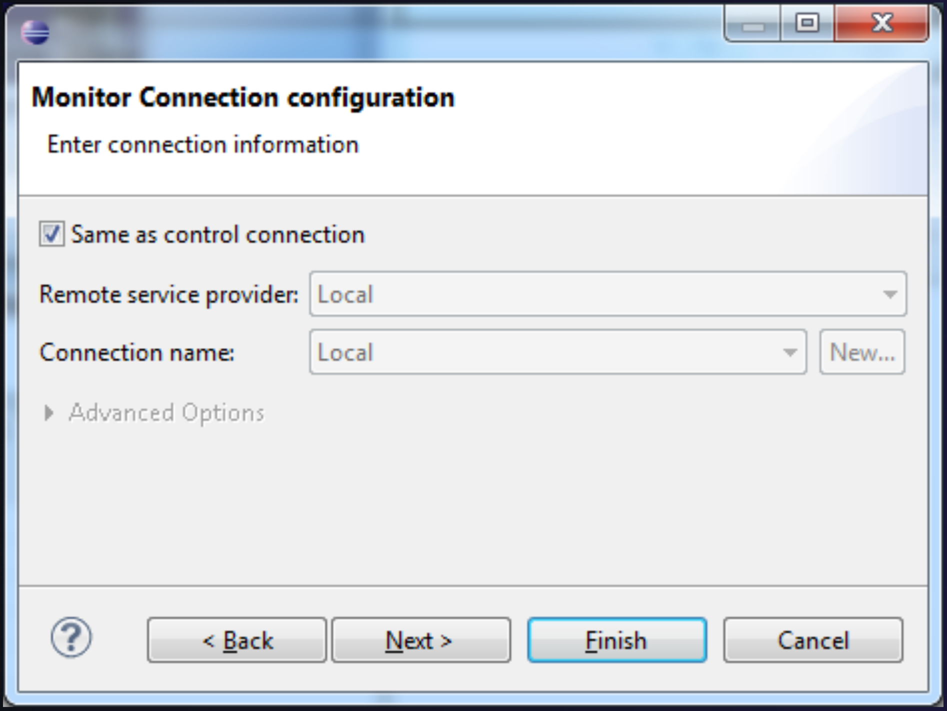
Enter connection information

Remote service provider: Remote Tools

Connection name: lincoln

- Choose **Remote Tools** for **Remote service provider**
- Choose the remote connection you made previously
- Click **Next>**

Configure the Monitor Connection


Monitor Connection configuration

Enter connection information

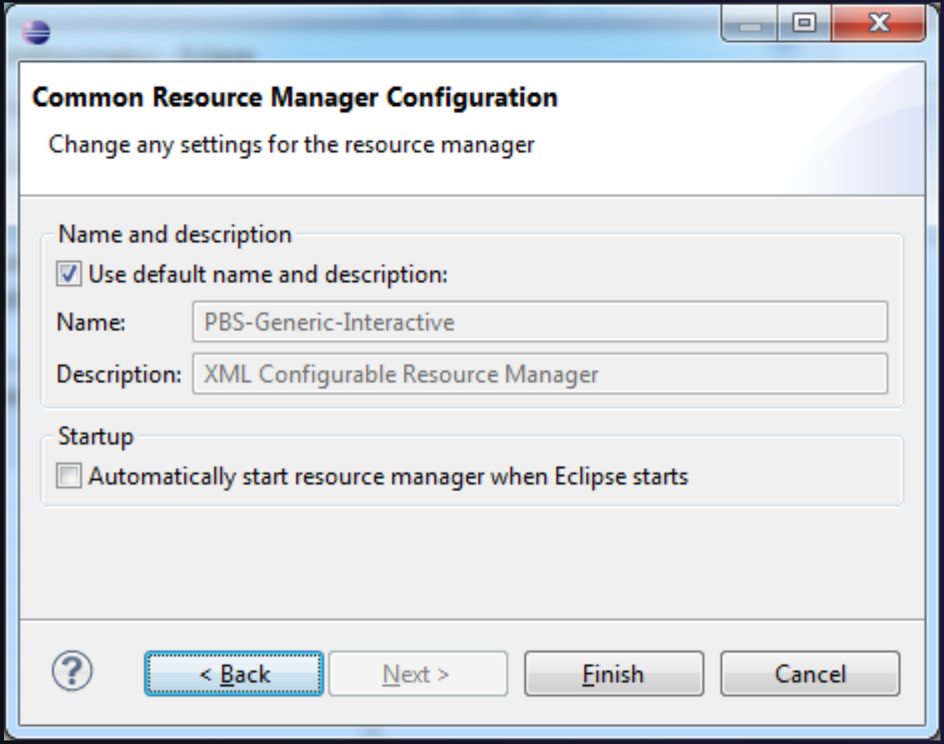
☑ Same as control connection

Remote service provider: Local

Connection name: Local

- Keep default Monitor Connection (same as Control Connection), click **Next**

Configure the Common Resource Manager Parameters



Common Resource Manager Configuration

Change any settings for the resource manager

Name and description

[✓] Use default name and description:

Name: PBS-Generic-Interactive

Description: XML Configurable Resource Manager

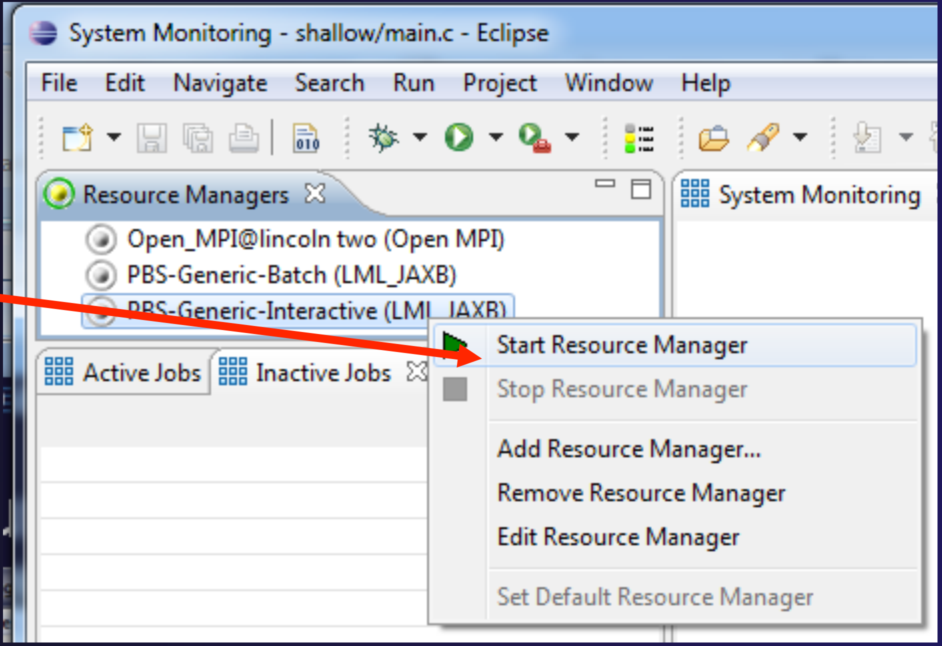
Startup

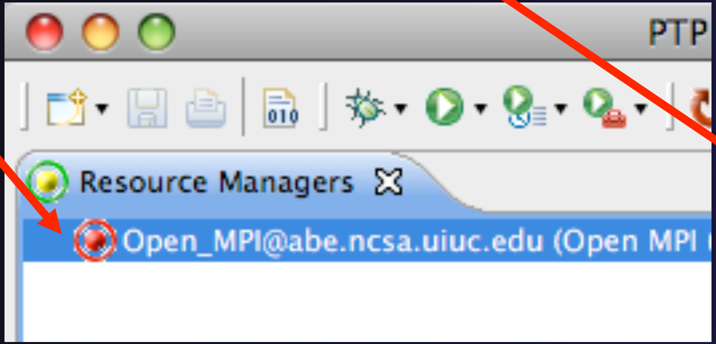
[ ] Automatically start resource manager when Eclipse starts

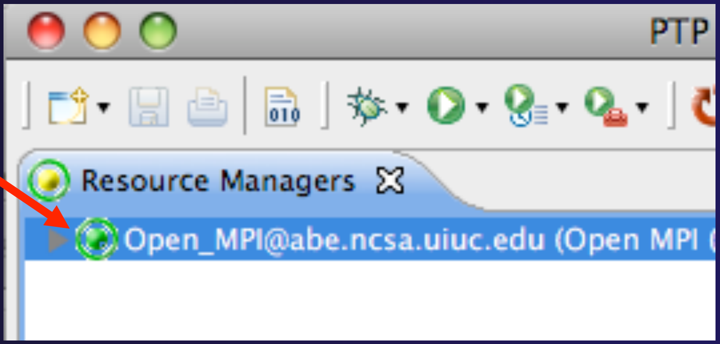
- Keep default name
- Can automatically start Resource Manager (leave unselected today)
- Click **Finish**

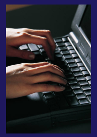
Starting the Resource Manager

- Right click on new resource manager and select **Start resource manager**
- If everything is ok, you should see the resource manager change to green
- If something goes wrong, it will change to red


System Monitoring - shallow/main.c - Eclipse
File Edit Navigate Search Run Project Window Help
Resource Managers
System Monitoring
Open_MPI@lincoln two (Open MPI)
PBS-Generic-Batch (LML_JAXB)
PBS-Generic-Interactive (LML_JAXB)
Active Jobs
Inactive Jobs
Start Resource Manager
Stop Resource Manager
Add Resource Manager...
Remove Resource Manager
Edit Resource Manager
Set Default Resource Manager

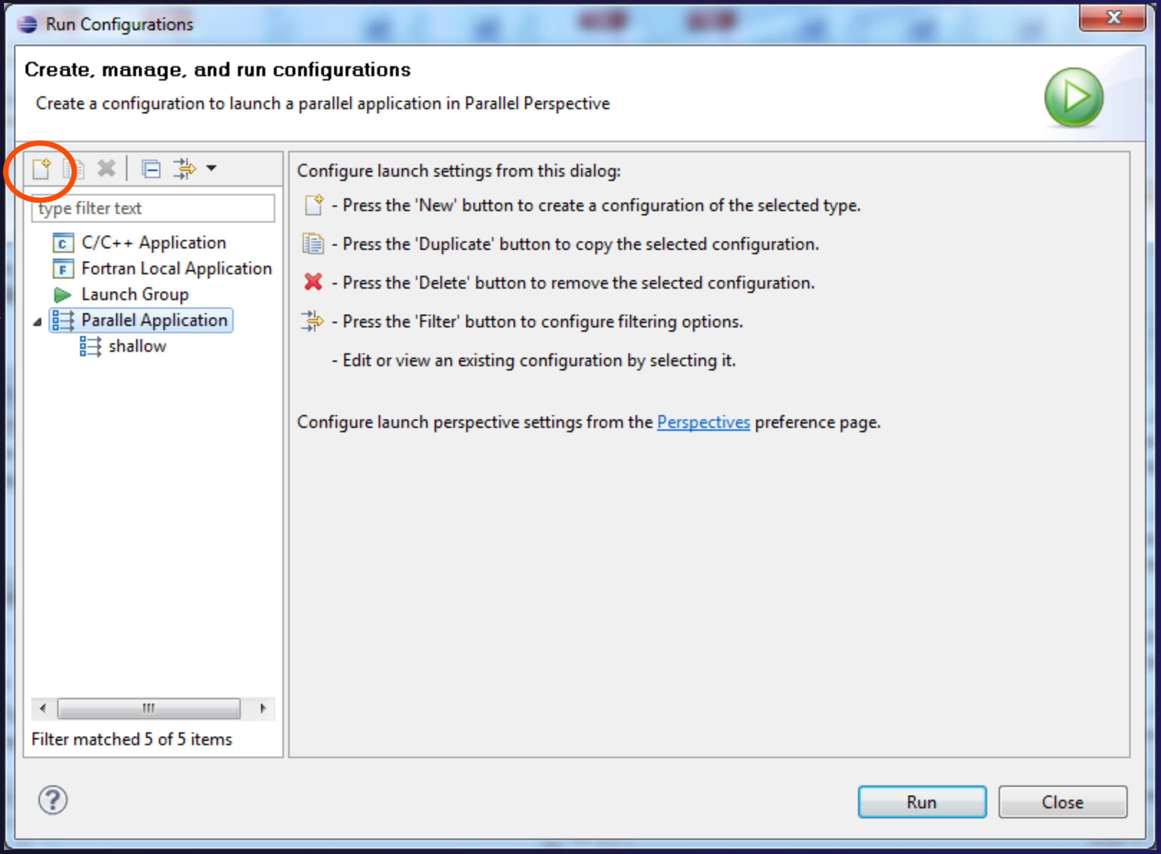






Create a Launch Configuration

- Open the run configuration dialog **Run>Run Configurations...**
- Select **Parallel Application**
- Select the **New** button



Create, manage, and run configurations

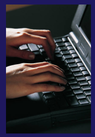
Create a configuration to launch a parallel application in Parallel Perspective

Configure launch settings from this dialog:

- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

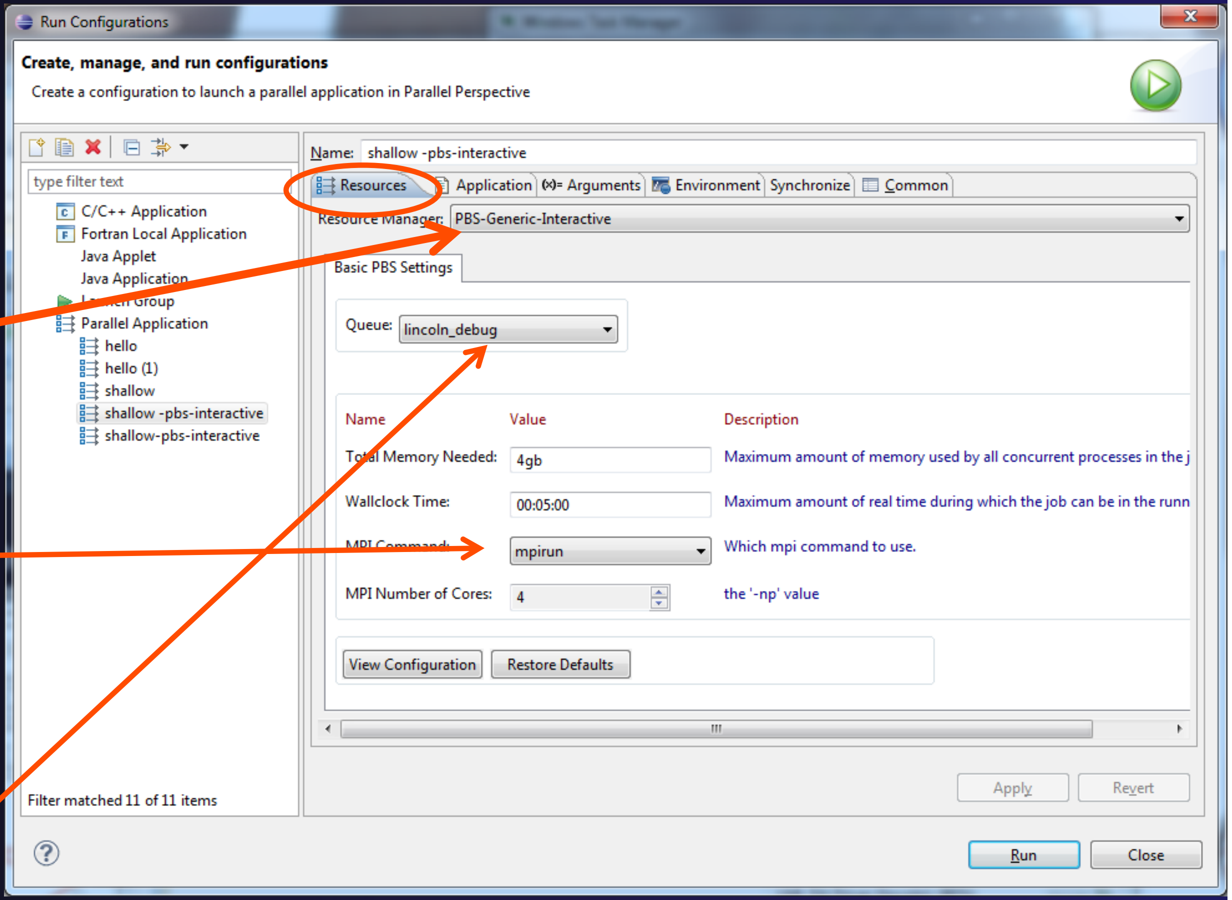
Configure launch perspective settings from the Perspectives preference page.

Filter matched 5 of 5 items



Complete the Resources Tab

- Enter a name for this launch configuration, e.g. "shallow-pbs-interactive
- In **Resources** tab, select the PBS resource manager you just created
- The **MPI Command** field allows this job to be run as an MPI job
  - Choose **mpirun**
- Enter the resources needed to run this job
  - Use 4 gb memory, 4 cores
- Select the destination queue – **lincoln_debug**



Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

type filter text

- C/C++ Application
- Fortran Local Application
- Java Applet
- Java Application
- Launch Group
- Parallel Application
  - hello
  - hello (1)
  - shallow
  - shallow -pbs-interactive
  - shallow-pbs-interactive

Filter matched 11 of 11 items

Name: shallow -pbs-interactive

Resources | Application | Arguments | Environment | Synchronize | Common

Resource Manager: PBS-Generic-Interactive

Basic PBS Settings

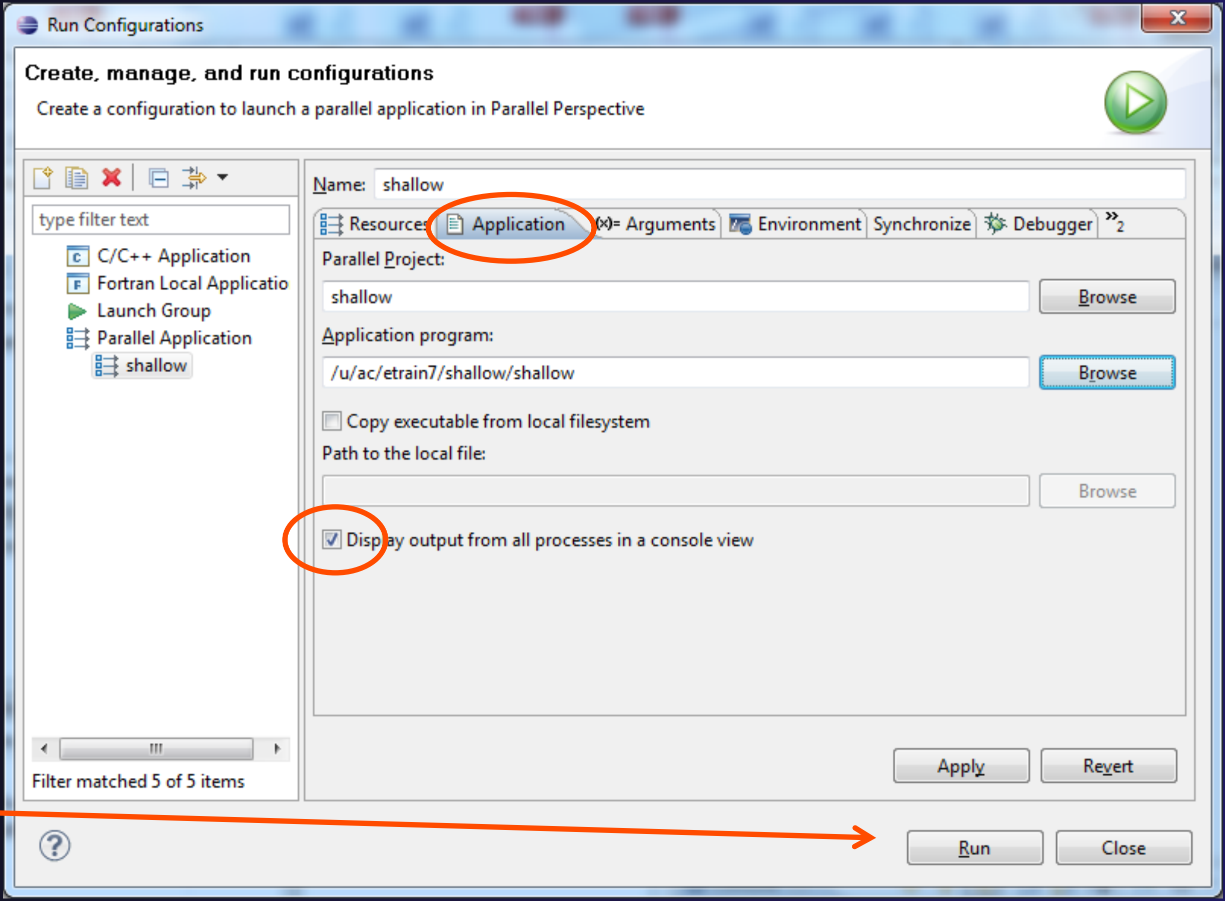
Queue: lincoln_debug

Name	Value	Description
Total Memory Needed:	4gb	Maximum amount of memory used by all concurrent processes in the j
Wallclock Time:	00:05:00	Maximum amount of real time during which the job can be in the runn
MPI Command:	mpirun	Which mpi command to use.
MPI Number of Cores:	4	the '-np' value

Complete the Application Tab

- Select the **Application** tab
- Choose the **Application program** by clicking the **Browse** button and locating the executable on the remote machine
  - Use the same "shallow" executable
- Select **Display output from all processes in a console view**
- If Debugger tab has error, select Debugger: **SDM**

- Click **Run** to submit the application to the job scheduler



Create a configuration to launch a parallel application in Parallel Perspective

Name: shallow

Parallel Project: shallow

Application program: /u/ac/etrain7/shallow/shallow

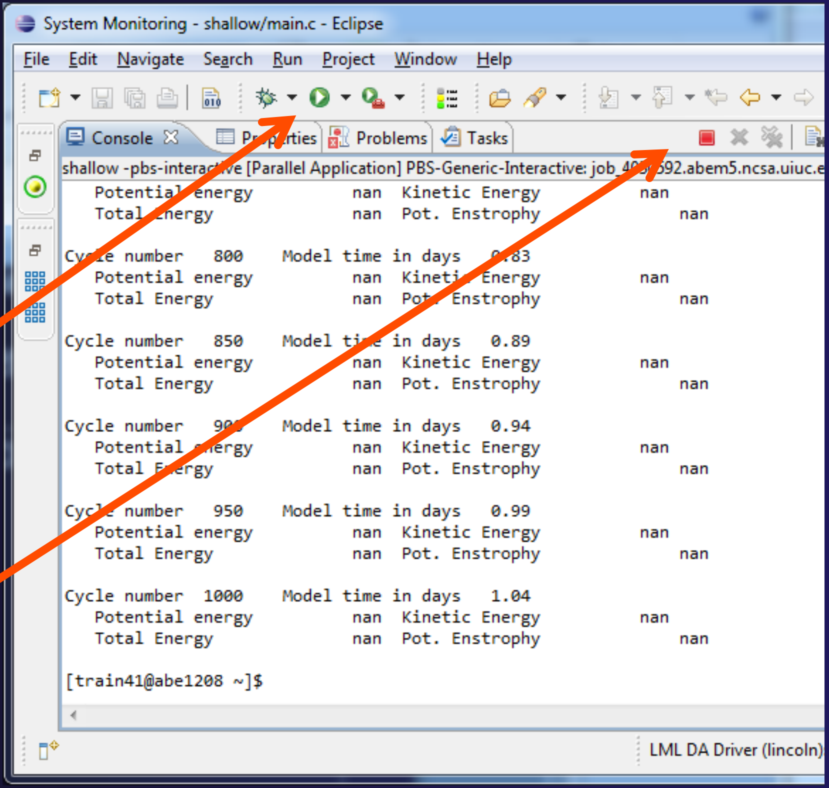
☐ Copy executable from local filesystem

Path to the local file:

☑ Display output from all processes in a console view

Running the Interactive job

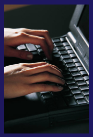
- Maximizing the console, you can see output from the job
- Use Run button to re-run application within the interactive run
- Use Stop button to end batch job


System Monitoring - shallow/main.c - Eclipse
File Edit Navigate Search Run Project Window Help
Console
Properties
Problems
Tasks
LML DA Driver (lincoln)

```
shallow -pbs-interactive [Parallel Application] PBS-Generic-Interactive: job_40?592.abem5.ncsa.uiuc.e
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

Cycle number   800    Model time in days   0.83
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

Cycle number   850    Model time in days   0.89
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

Cycle number   900    Model time in days   0.94
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

Cycle number   950    Model time in days   0.99
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

Cycle number  1000    Model time in days   1.04
   Potential energy          nan  Kinetic Energy          nan
   Total Energy              nan  Pot. Enstrophy               nan

[train41@abe1208 ~]$
```

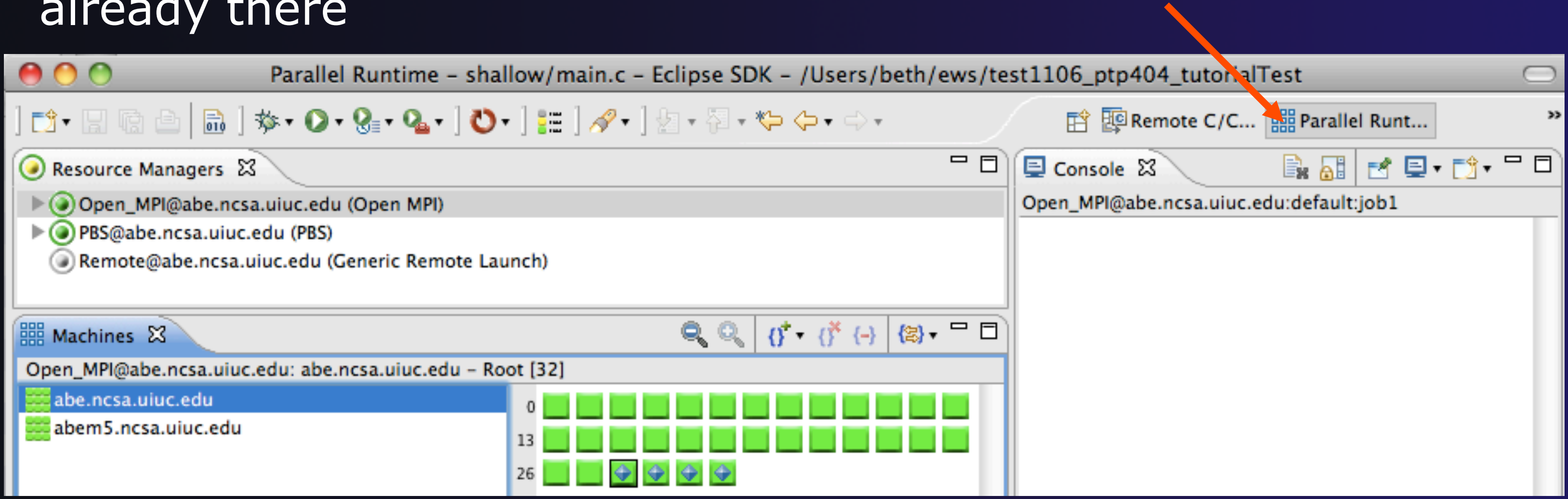
Module 5: Parallel Debugging

- Objective
  - Learn the basics of debugging parallel programs
- Contents
  - Launching a debug session
  - The Parallel Debug Perspective
  - Controlling sets of processes
  - Controlling individual processes
  - Parallel Breakpoints
  - Terminating processes



Debugging an Application

- Debugging requires interactive access to the application
- Since PBS is for batch execution, we will use Open MPI to provide interactive access to the machine (PBS will support interactive execution in the future)
- First switch to the Parallel Runtime perspective if not already there


Parallel Runtime – shallow/main.c – Eclipse SDK – /Users/beth/ews/test1106_ptp404_tutorialTest
Remote C/C...
Parallel Runt...
Resource Managers
Console
Machines

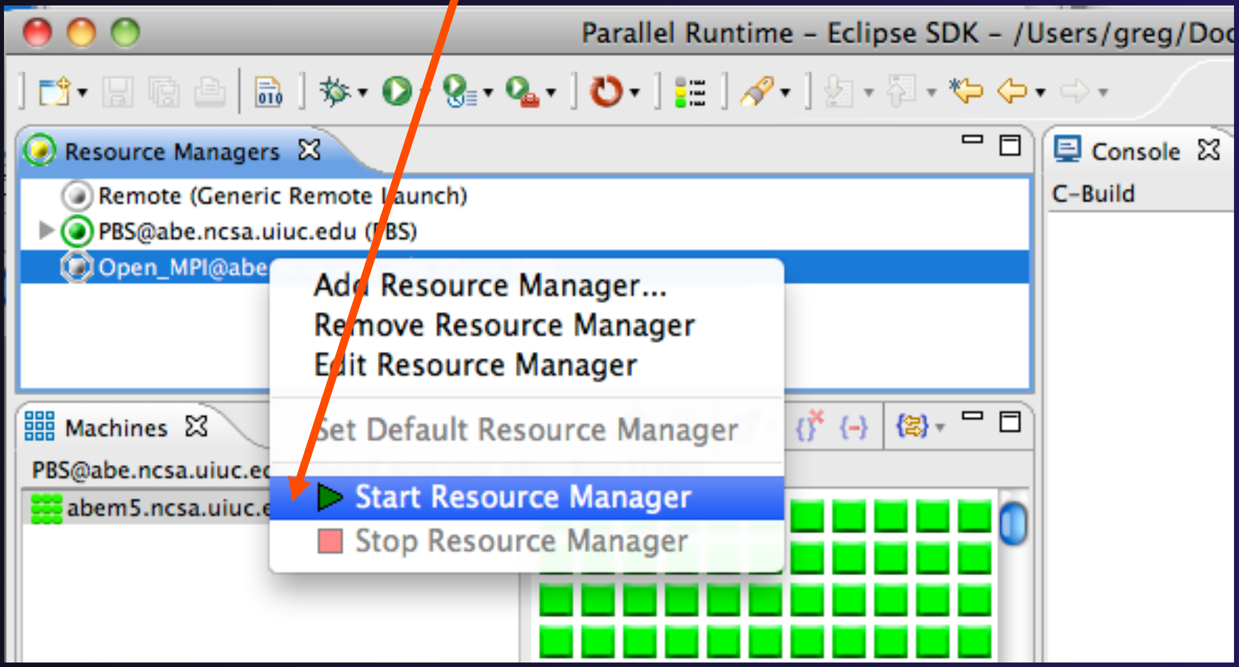
Open_MPI@abe.ncsa.uiuc.edu (Open MPI)
PBS@abe.ncsa.uiuc.edu (PBS)
Remote@abe.ncsa.uiuc.edu (Generic Remote Launch)

Open_MPI@abe.ncsa.uiuc.edu:default:job1

Open_MPI@abe.ncsa.uiuc.edu: abe.ncsa.uiuc.edu – Root [32]
abe.ncsa.uiuc.edu
abem5.ncsa.uiuc.edu

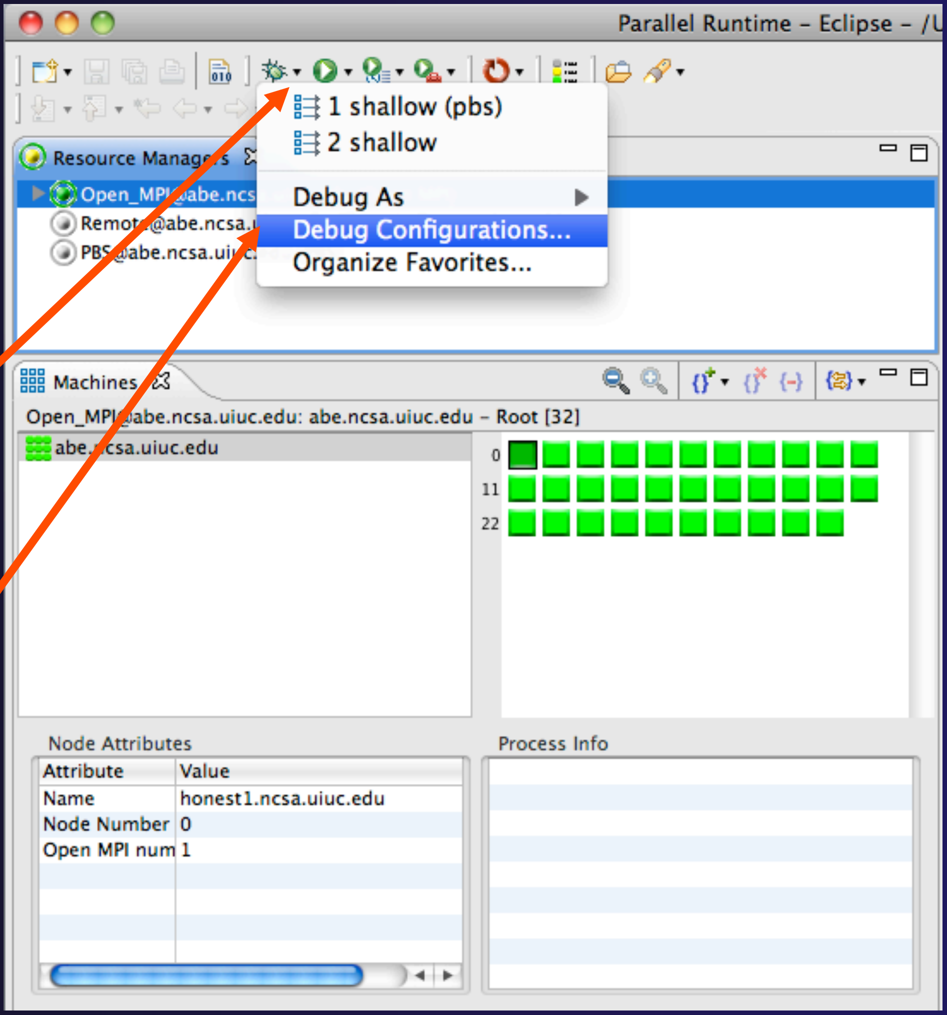
Start the Resource Manager

- If the Open_MPI Resource manager is not already started (green icon), start it now:
  - Right-click on the resource manager and select **Start Resource Manager** from the menu

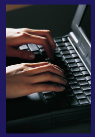


Create a Debug Configuration

- A debug configuration is essentially the same as a run configuration (like we used in modules 3 & 4)
- We will re-use the existing configuration and add debug information
- Use the drop-down next to the debug button (bug icon) instead of run button
- Select **Debug Configurations…** to open the **Debug Configurations** dialog

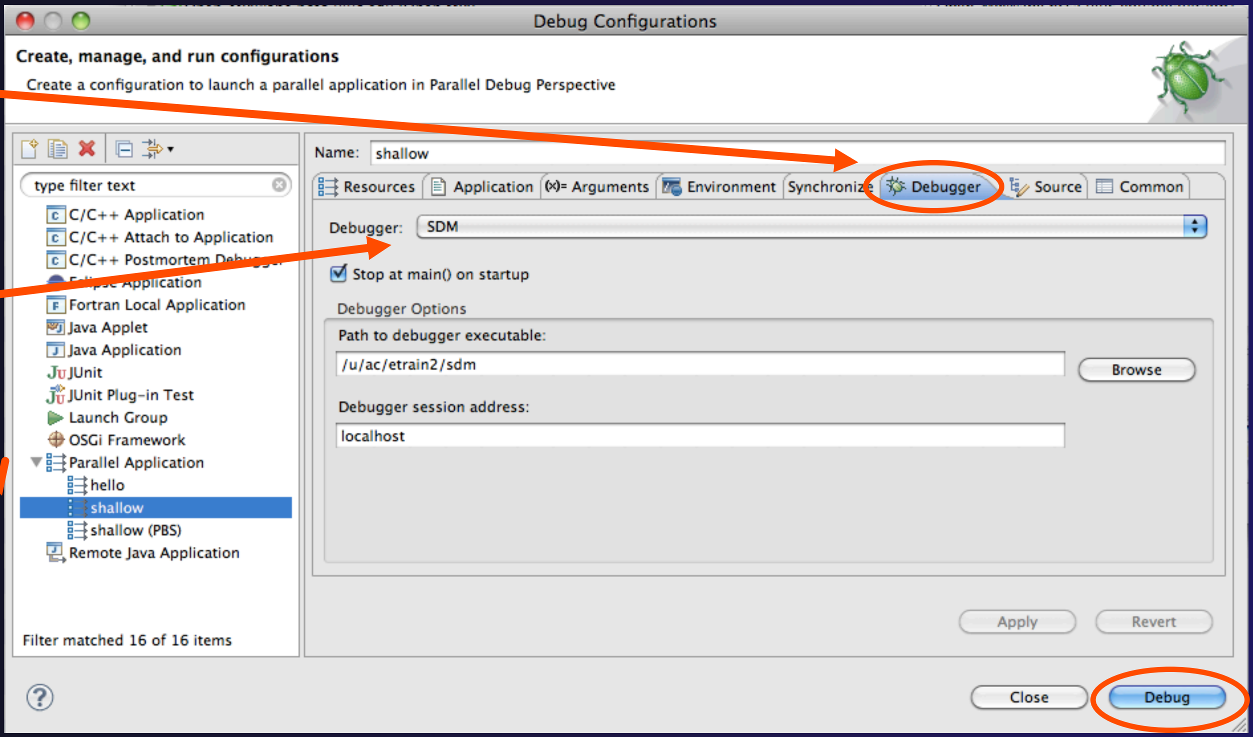


Attribute	Value
Name	honest1.ncsa.uiuc.edu
Node Number	0
Open MPI num	1



Configure the Debugger Tab

- Select **Debugger** tab
- Select the **shallow** configuration
- Make sure **SDM** is selected in the **Debugger** dropdown
- Check the debugger path is correct
  - Should be the path to the sdm executable on the remote system
- Debugger session address should not need to be changed
- Click on **Debug** to launch the program



Debug Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Debug Perspective

Name: shallow

Debugger: SDM

☑ Stop at main() on startup

Debugger Options

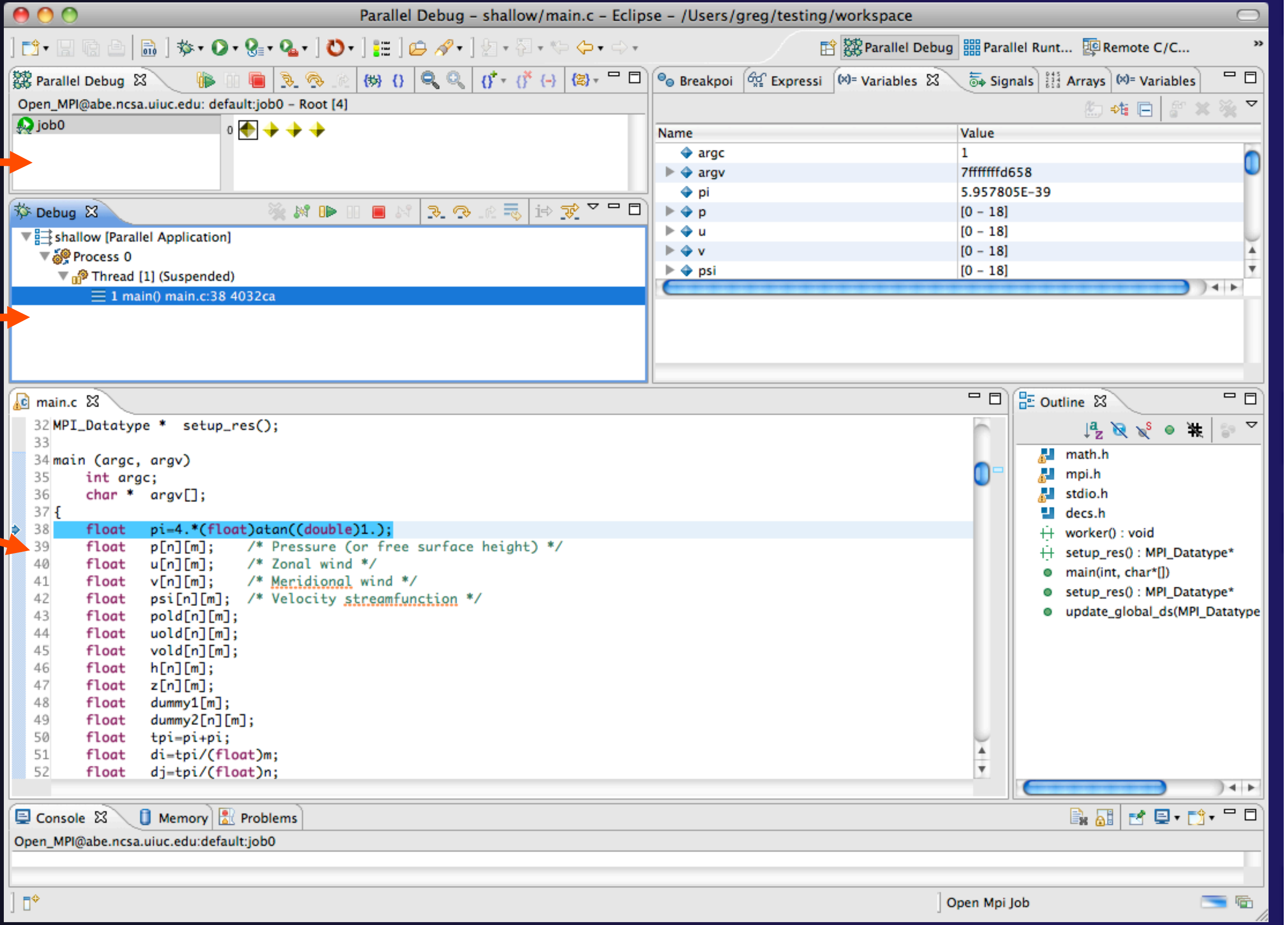
Path to debugger executable: /u/ac/etrain2/sdm

Debugger session address: localhost

Filter matched 16 of 16 items

The Parallel Debug Perspective (1)

- **Parallel Debug view** shows job and processes being debugged
- **Debug** view shows threads and call stack for individual processes
- **Source** view shows a **current line marker** for all processes



Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspace

Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

job0

shallow [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() main.c:38 4032ca

Name	Value
argc	1
argv	7fffffffd658
pi	5.957805E-39
p	[0 – 18]
u	[0 – 18]
v	[0 – 18]
psi	[0 – 18]

```
MPI_Datatype *  setup_res();

main (argc, argv)
    int argc;
    char *  argv[];
{
    float   pi=4.*(float)atan((double)1.);
    float   p[n][m];    /* Pressure (or free surface height) */
    float   u[n][m];    /* Zonal wind */
    float   v[n][m];    /* Meridional wind */
    float   psi[n][m];  /* Velocity streamfunction */
    float   pold[n][m];
    float   uold[n][m];
    float   vold[n][m];
    float   h[n][m];
    float   z[n][m];
    float   dummy1[m];
    float   dummy2[n][m];
    float   tpi=pi+pi;
    float   di=tpi/(float)m;
    float   dj=tpi/(float)n;
```

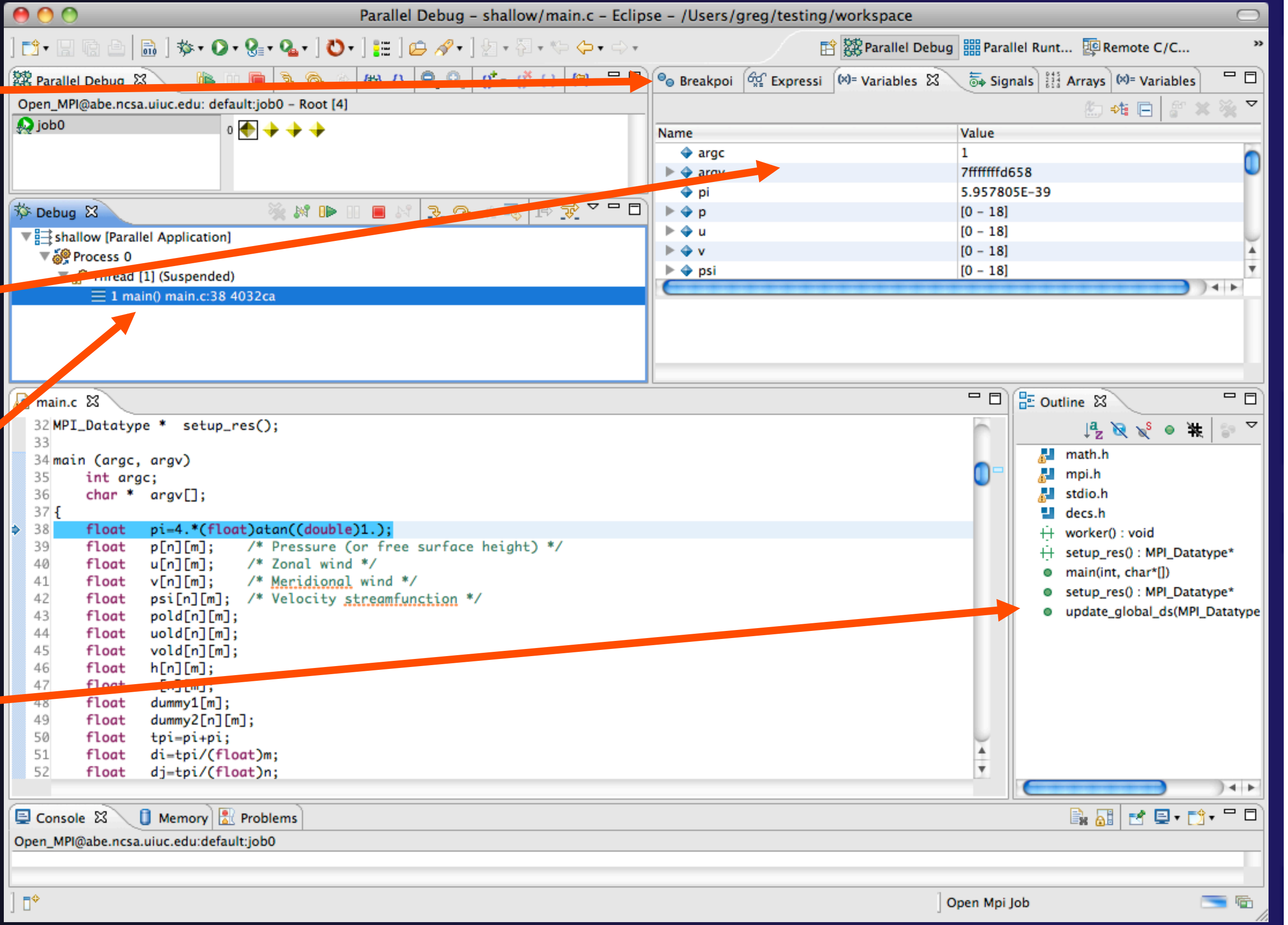
math.h
mpi.h
stdio.h
decs.h
worker() : void
setup_res() : MPI_Datatype*
main(int, char*[])
setup_res() : MPI_Datatype*
update_global_ds(MPI_Datatype

Open_MPI@abe.ncsa.uiuc.edu:default:job0

Open Mpi Job

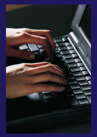
The Parallel Debug Perspective (2)

- **Breakpoints** view shows breakpoints that have been set (more on this later)
- **Variables** view shows the current values of variables for the currently selected process in the **Debug** view
- **Outline** view (from CDT) of source code


Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspace
Parallel Debug
Parallel Runt...
Remote C/C...
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]
job0
Debug
shallow [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() main.c:38 4032ca
Breakpoi
Expressi
Variables
Signals
Arrays
Variables
main.c
Outline
math.h
mpi.h
stdio.h
decs.h
worker() : void
setup_res() : MPI_Datatype*
main(int, char*[])
setup_res() : MPI_Datatype*
update_global_ds(MPI_Datatype
Console
Memory
Problems
Open_MPI@abe.ncsa.uiuc.edu:default:job0
Open Mpi Job

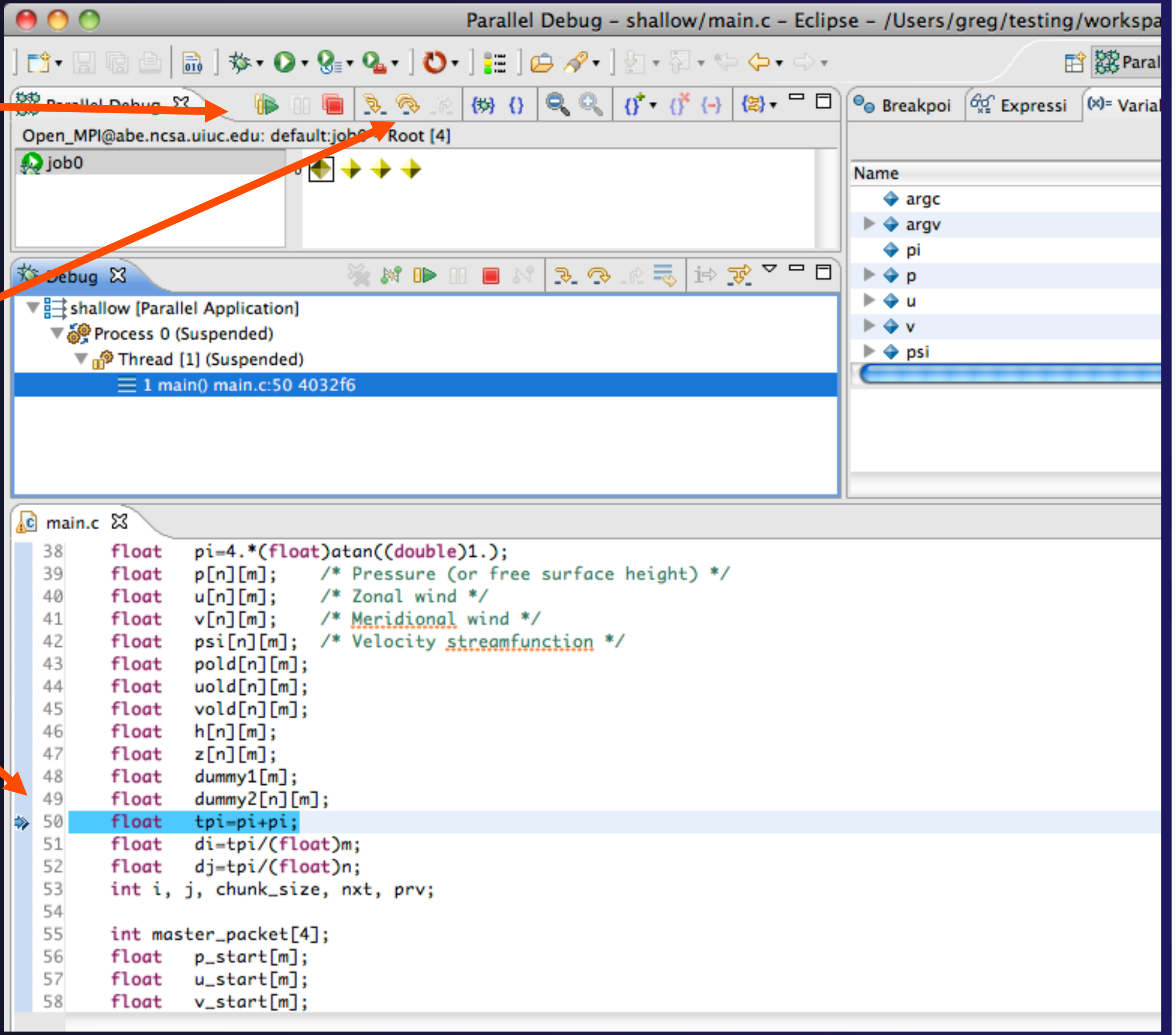
Name	Value
argc	1
argv	7fffffffd658
pi	5.957805E-39
p	[0 – 18]
u	[0 – 18]
v	[0 – 18]
psi	[0 – 18]

```
MPI_Datatype *  setup_res();

main (argc, argv)
    int argc;
    char *  argv[];
{
    float   pi=4.*(float)atan((double)1.);
    float   p[n][m];    /* Pressure (or free surface height) */
    float   u[n][m];    /* Zonal wind */
    float   v[n][m];    /* Meridional wind */
    float   psi[n][m];  /* Velocity streamfunction */
    float   pold[n][m];
    float   uold[n][m];
    float   vold[n][m];
    float   h[n][m];
    float   [illegible][m];
    float   dummy1[m];
    float   dummy2[n][m];
    float   tpi=pi+pi;
    float   di=tpi/(float)m;
    float   dj=tpi/(float)n;
```



Stepping All Processes

- The buttons in the **Parallel Debug View** control groups of processes
- Click on the **Step Over** button
- Observe that all process icons change to green, then back to yellow
- Notice that the current line marker has moved to the next source line



Parallel Debug - shallow/main.c - Eclipse - /Users/greg/testing/workspa

Open_MPI@abe.ncsa.uiuc.edu: default:job0 Root [4]

job0

shallow [Parallel Application]
Process 0 (Suspended)
Thread [1] (Suspended)
1 main() main.c:50 4032f6

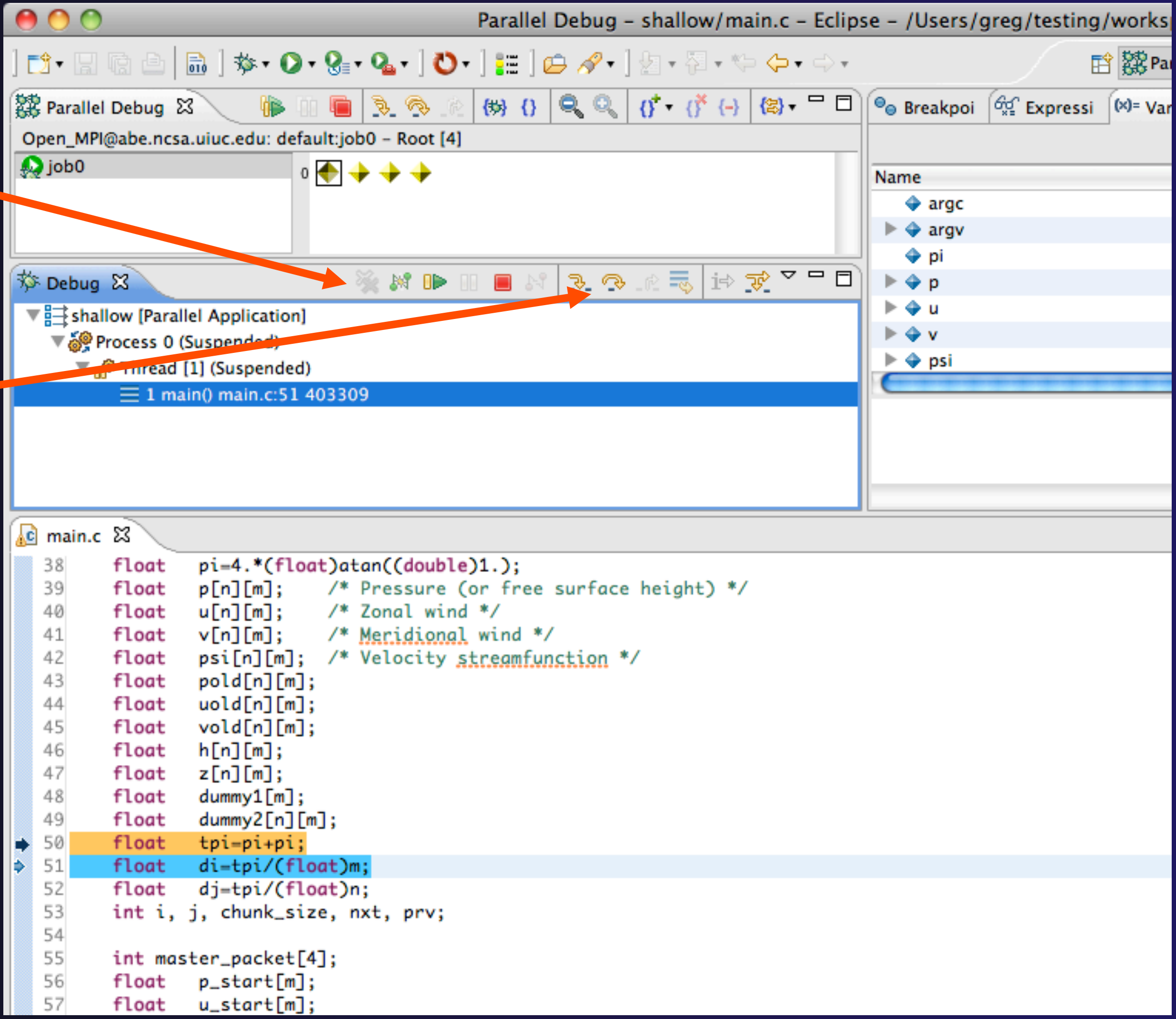
argc
argv
pi
p
u
v
psi

main.c

```
float   pi=4.*(float)atan((double)1.);
float   p[n][m];     /* Pressure (or free surface height) */
float   u[n][m];     /* Zonal wind */
float   v[n][m];     /* Meridional wind */
float   psi[n][m];   /* Velocity streamfunction */
float   pold[n][m];
float   uold[n][m];
float   vold[n][m];
float   h[n][m];
float   z[n][m];
float   dummy1[m];
float   dummy2[n][m];
float   tpi=pi+pi;
float   di=tpi/(float)m;
float   dj=tpi/(float)n;
int i, j, chunk_size, nxt, prv;

int master_packet[4];
float   p_start[m];
float   u_start[m];
float   v_start[m];
```

Stepping An Individual Process

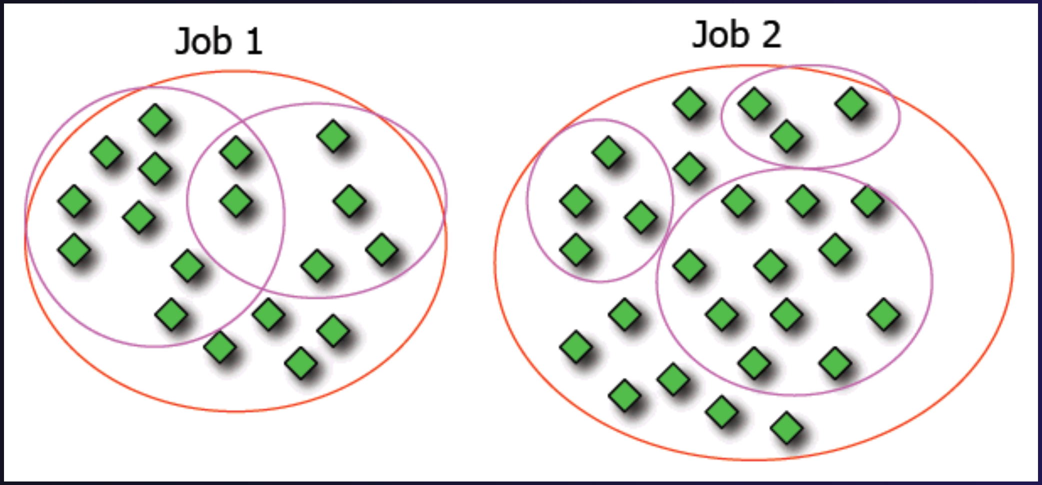
- The buttons in the **Debug view** are used to control an individual process, in this case process 0
- Click the **Step Over** button
- You will now see two current line markers, the first shows the position of process 0, the second shows the positions of processes 1-3


Parallel Debug - shallow/main.c - Eclipse - /Users/greg/testing/works
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 - Root [4]
job0
0
Debug
shallow [Parallel Application]
Process 0 (Suspended)
Thread [1] (Suspended)
1 main() main.c:51 403309
Breakpoi
Expressi
Var
Name
argc
argv
pi
p
u
v
psi
main.c

```
float    pi=4.*(float)atan((double)1.);
float    p[n][m];    /* Pressure (or free surface height) */
float    u[n][m];    /* Zonal wind */
float    v[n][m];    /* Meridional wind */
float    psi[n][m];  /* Velocity streamfunction */
float    pold[n][m];
float    uold[n][m];
float    vold[n][m];
float    h[n][m];
float    z[n][m];
float    dummy1[m];
float    dummy2[n][m];
float    tpi=pi+pi;
float    di=tpi/(float)m;
float    dj=tpi/(float)n;
int i, j, chunk_size, nxt, prv;

int master_packet[4];
float    p_start[m];
float    u_start[m];
```

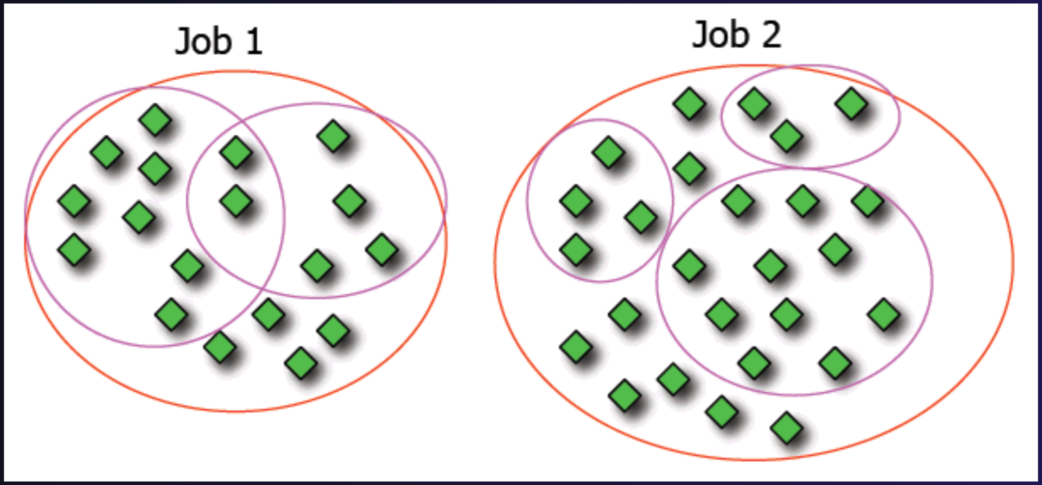
Process Sets (1)

- Traditional debuggers apply operations to a single process
- Parallel debugging operations apply to a single process or to arbitrary collections of processes
- A process set is a means of simultaneously referring to one or more processes



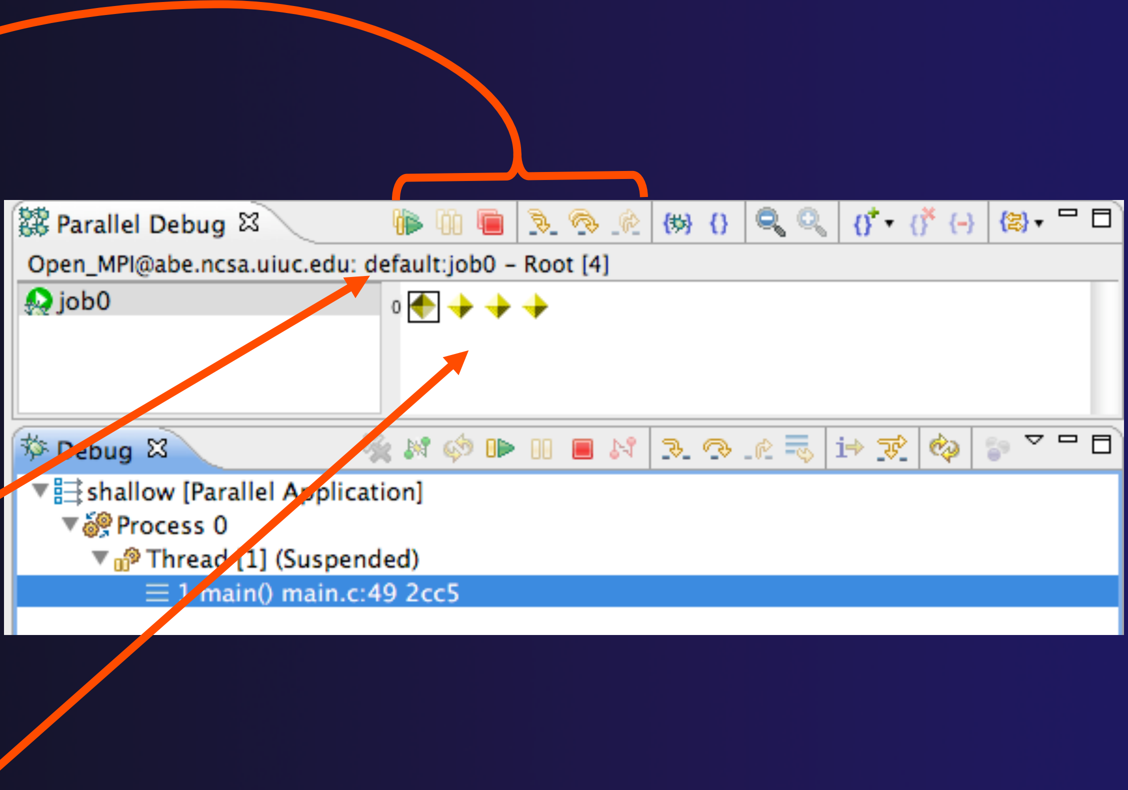
Process Sets (2)

- When a parallel debug session is first started, all processes are placed in a set, called the **Root** set
- Sets are always associated with a single job
- A job can have any number of process sets
- A set can contain from 1 to the number of processes in a job



Operations On Process Sets

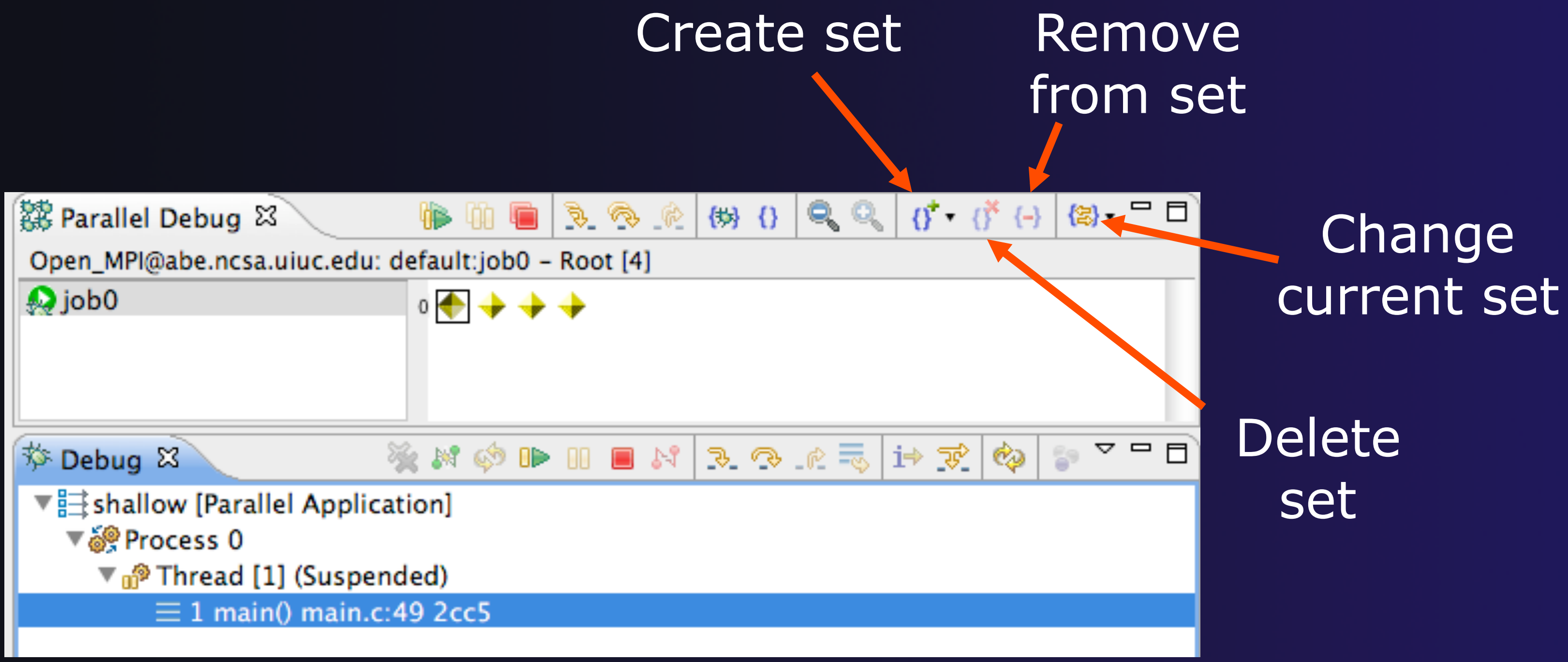
- Debug operations on the **Parallel Debug view** toolbar always apply to the current set:
  - Resume, suspend, stop, step into, step over, step return
- The current process set is listed next to job name along with number of processes in the set
- The processes in process set are visible in right hand part of the view


Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]
job0
0
Debug
shallow [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() main.c:49 2cc5

Root set = all processes

Managing Process Sets

- The remaining icons in the toolbar of the **Parallel Debug view** allow you to create, modify, and delete process sets, and to change the current process set



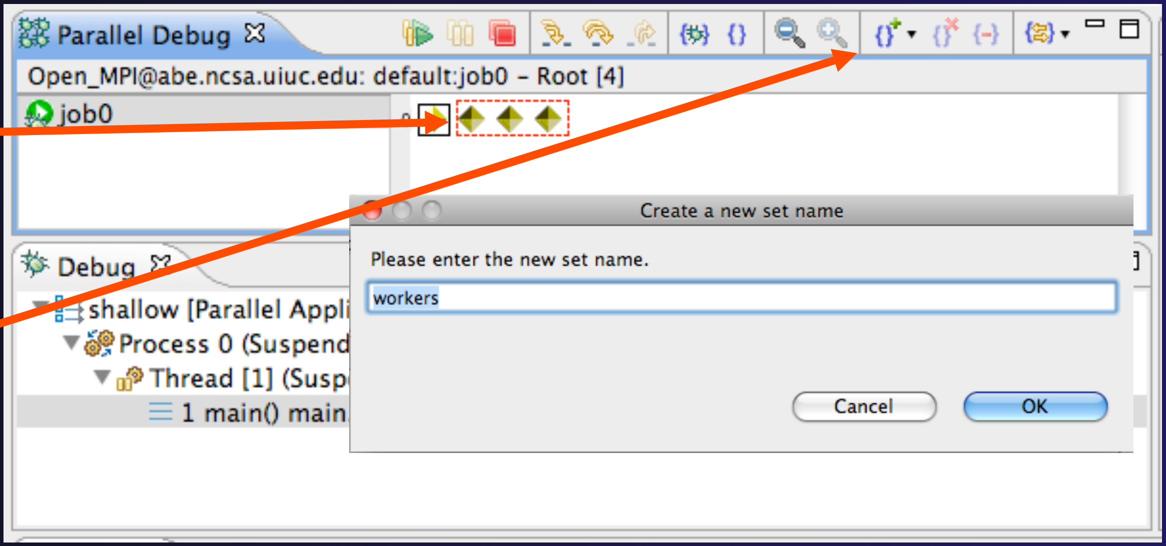
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

job0

shallow [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() main.c:49 2cc5

Creating A New Process Set

- Select the processes you want in the set by clicking and dragging, in this case, the last three
- Click on the **Create Set** button
- Enter a name for the set, in this case **workers**, and click **OK**
- You will see the view change to display only the selected processes


Parallel Debug

Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

job0

Debug

shallow [Parallel Appli

Process 0 (Suspend

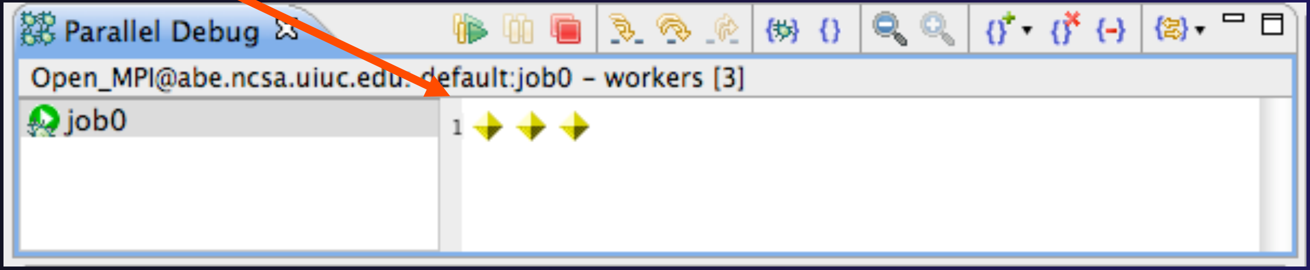
Thread [1] (Susp

1 main() main

Create a new set name

Please enter the new set name.

workers

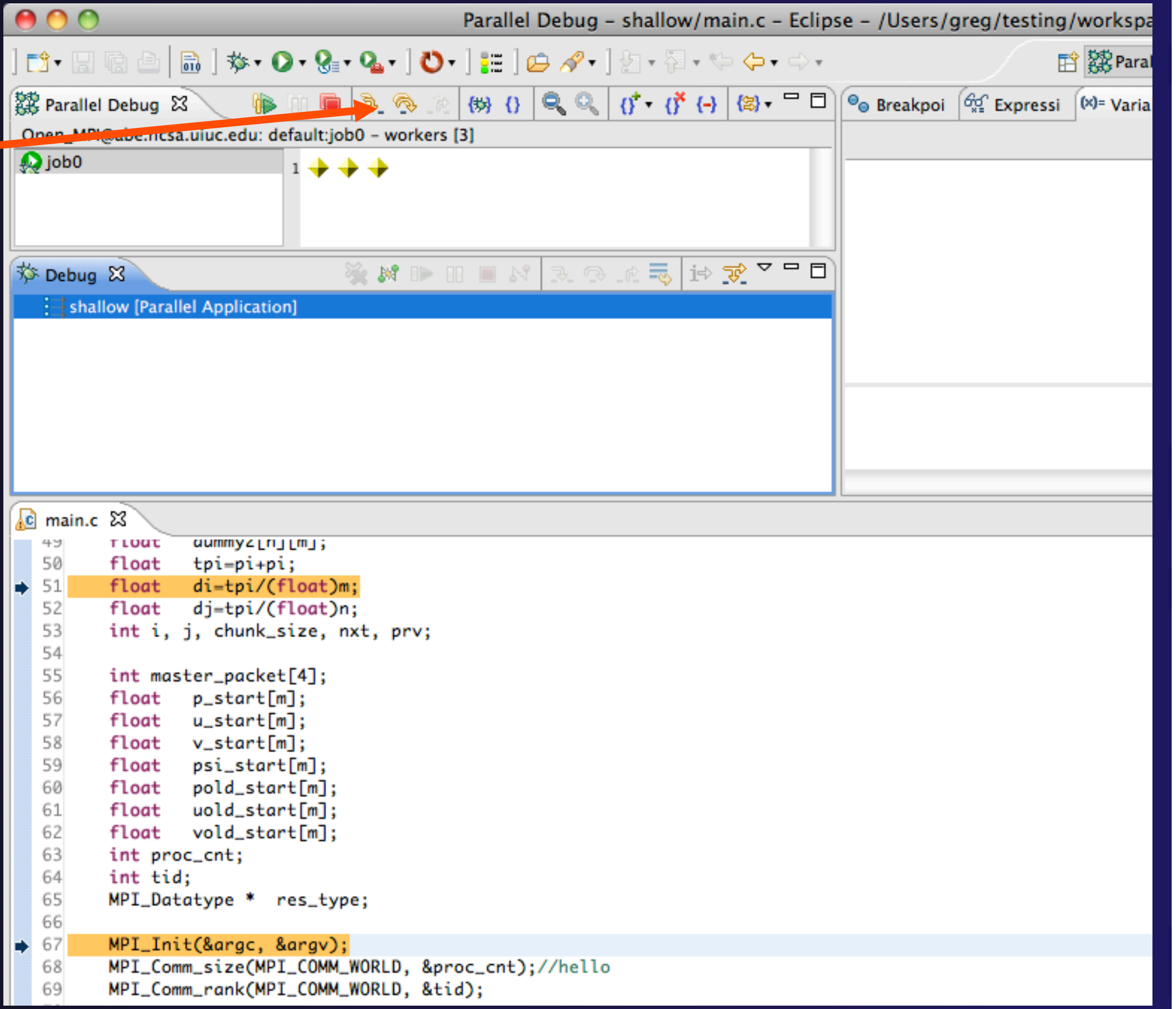

Parallel Debug

Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]

job0

Stepping Using New Process Set

- With the **workers** set active, click the **Step Over** button
- You will see only the first current line marker move
- Step a couple more times
- You should see two line markers, one for the single master process, and one for the 3 worker processes


Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspa
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]
job0
Debug
shallow [Parallel Application]
Breakpoi
Expressi
Varia
main.c

```
    float   dummy2[n][m];
    float   tpi=pi+pi;
    float   di=tpi/(float)m;
    float   dj=tpi/(float)n;
    int i, j, chunk_size, nxt, prv;

    int master_packet[4];
    float   p_start[m];
    float   u_start[m];
    float   v_start[m];
    float   psi_start[m];
    float   pold_start[m];
    float   uold_start[m];
    float   vold_start[m];
    int proc_cnt;
    int tid;
    MPI_Datatype *  res_type;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);//hello
    MPI_Comm_rank(MPI_COMM_WORLD, &tid);
```

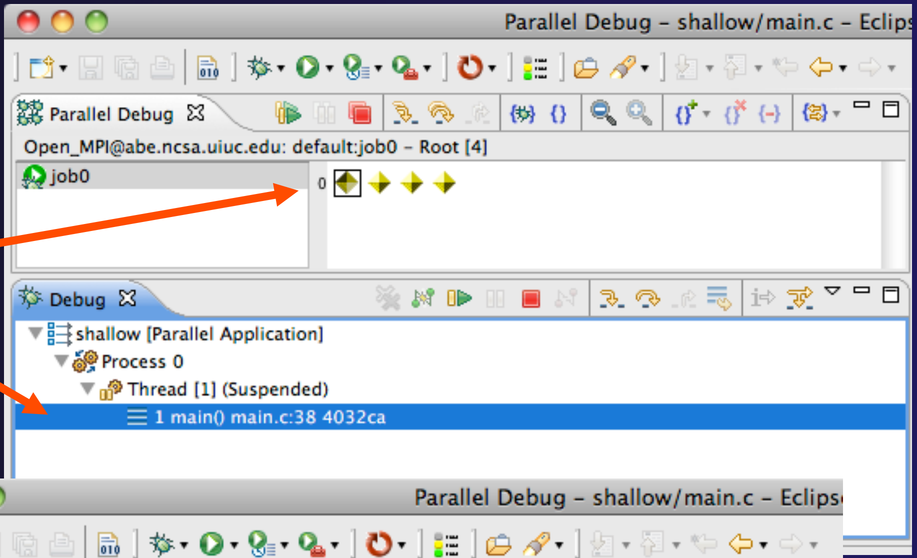
Process Registration

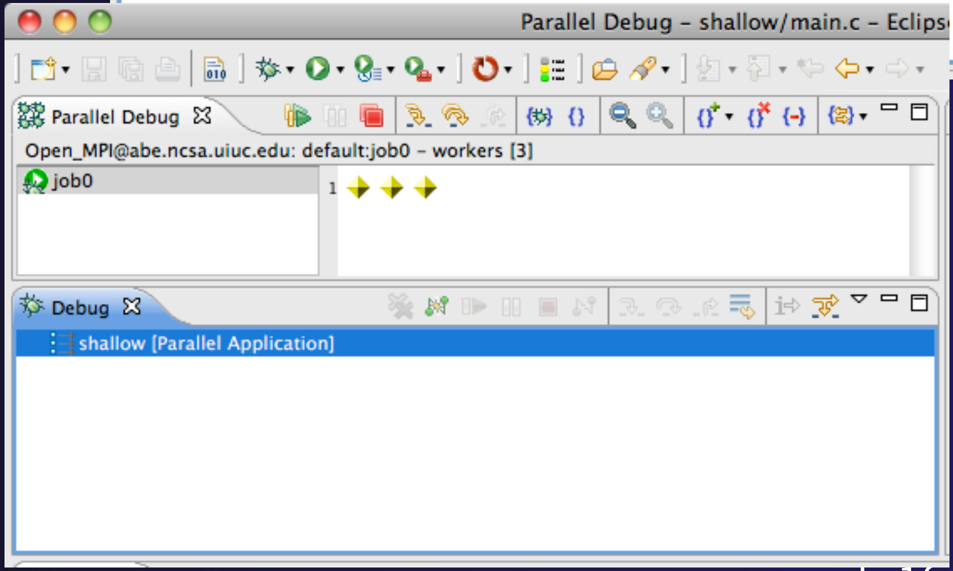
- Process set commands apply to groups of processes
- For finer control and more detailed information, a process can be registered and isolated in the **Debug view**
- Registered processes, including their stack traces and threads, appear in the **Debug view**
- Any number of processes can be registered, and processes can be registered or un-registered at any time

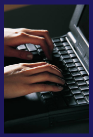
Process Registration (2)

- By default, process 0 was registered when the debug session was launched
- Registered processes are surrounded by a box and shown in the Debug view

- The Debug view only shows registered processes in the current set
- Since the "workers" set doesn't include process 0, it is no longer displayed in the Debug view

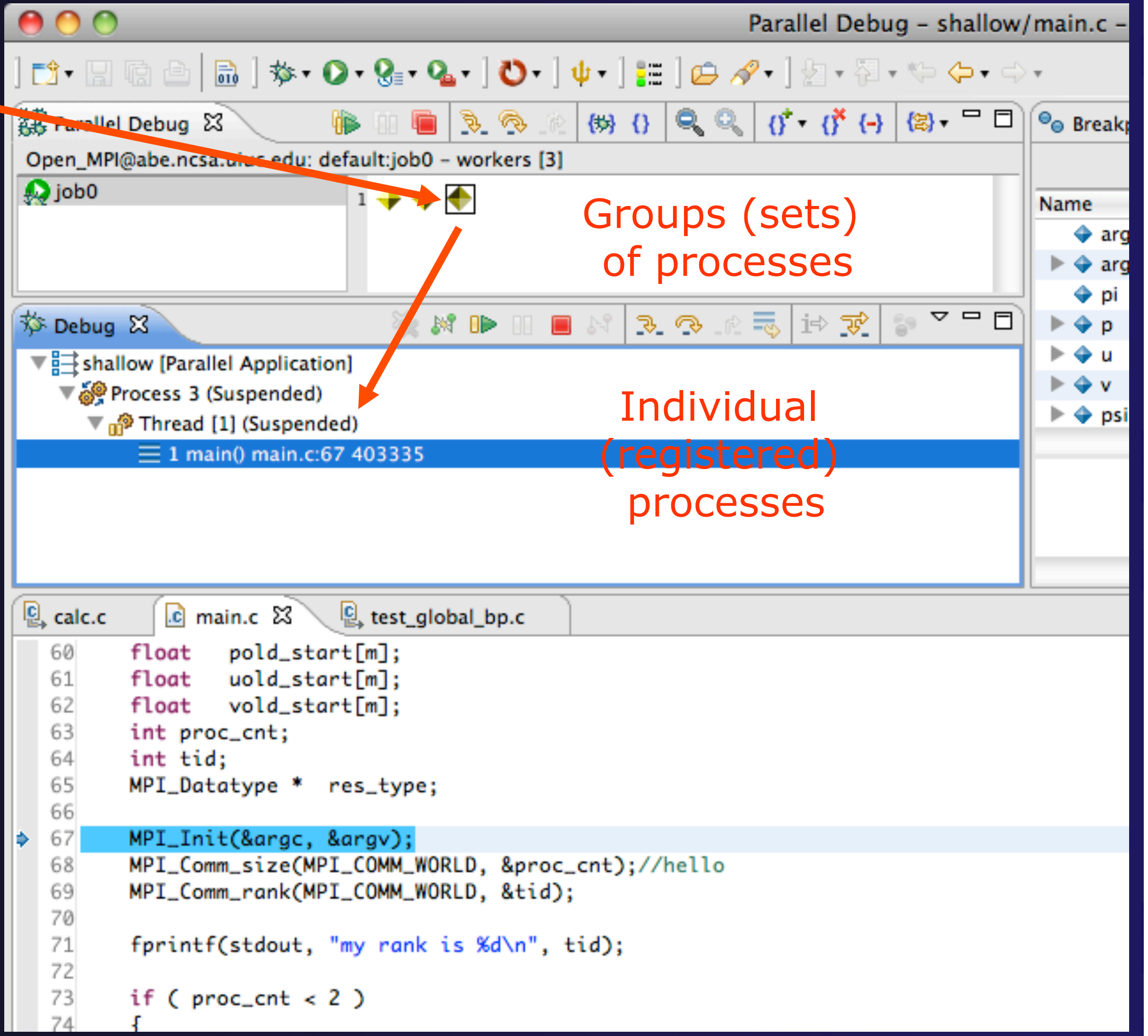

Parallel Debug – shallow/main.c – Eclips
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]
job0
0
Debug
shallow [Parallel Application]
Process 0
Thread [1] (Suspended)
1 main() main.c:38 4032ca


Parallel Debug – shallow/main.c – Eclipse
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]
job0
1
Debug
shallow [Parallel Application]



Registering A Process

- To register a process, double-click its process icon in the **Parallel Debug view** or select a number of processes and click on the **register** button
- To un-register a process, double-click on the process icon or select a number of processes and click on the **unregister** button


Parallel Debug – shallow/main.c –
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]
job0
1
Groups (sets) of processes
Breakp
Name
arg
arg
pi
p
u
v
psi
Debug
shallow [Parallel Application]
Process 3 (Suspended)
Thread [1] (Suspended)
1 main() main.c:67 403335
Individual (registered) processes
calc.c
main.c
test_global_bp.c

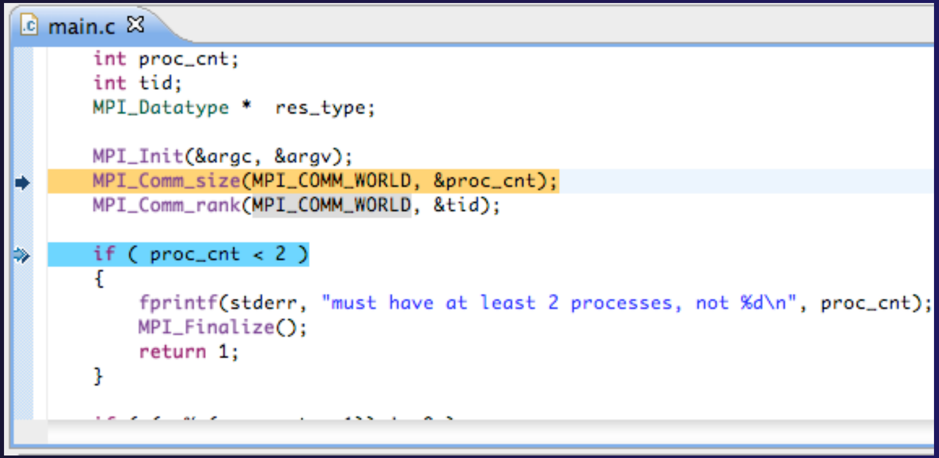
```
    float    pold_start[m];
    float    uold_start[m];
    float    vold_start[m];
    int proc_cnt;
    int tid;
    MPI_Datatype *  res_type;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);//hello
    MPI_Comm_rank(MPI_COMM_WORLD, &tid);

    fprintf(stdout, "my rank is %d\n", tid);

    if ( proc_cnt < 2 )
    {
```

Current Line Marker

- The current line marker is used to show the current location of suspended processes
- In traditional programs, there is a single current line marker (the exception to this is multi-threaded programs)
- In parallel programs, there is a current line marker for every process
- The PTP debugger shows one current line marker for every group of processes at the same location

Colors And Markers

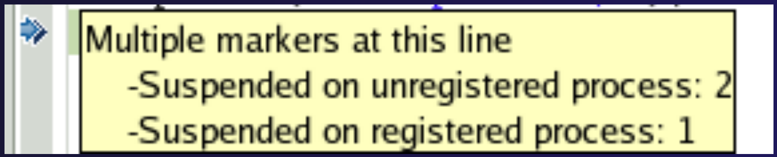
- The highlight color depends on the processes suspended at that line:
  - **Blue:** All registered process(es)
  - **Orange:** All unregistered process (es)
  - **Green:** Registered or unregistered process with no source line (e.g. suspended in a library routine)
- The marker depends on the type of process stopped at that location
- Hover over marker for more details about the processes suspend at that location



main.c

```
int proc_cnt;
int tid;
MPI_Datatype *  res_type;

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
MPI_Comm_rank(MPI_COMM_WORLD, &tid);

if ( proc_cnt < 2 )
{
    fprintf(stderr, "must have at least 2 processes, not %d\n", proc_cnt);
    MPI_Finalize();
    return 1;
}
```

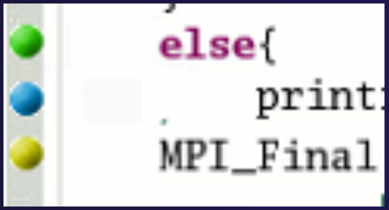
	Multiple processes marker
	Registered process marker
	Un-registered process marker



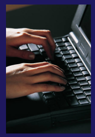
Multiple markers at this line
-Suspended on unregistered process: 2
-Suspended on registered process: 1

Breakpoints

- Apply only to processes in the particular set that is active in the **Parallel Debug view** when the breakpoint is created
- Breakpoints are colored depending on the active process set and the set the breakpoint applies to:
  - Green indicates the breakpoint set is the same as the active set.
  - Blue indicates some processes in the breakpoint set are also in the active set (i.e. the process sets overlap)
  - Yellow indicates the breakpoint set is different from the active set (i.e. the process sets are disjoint)
- When the job completes, the breakpoints are automatically removed

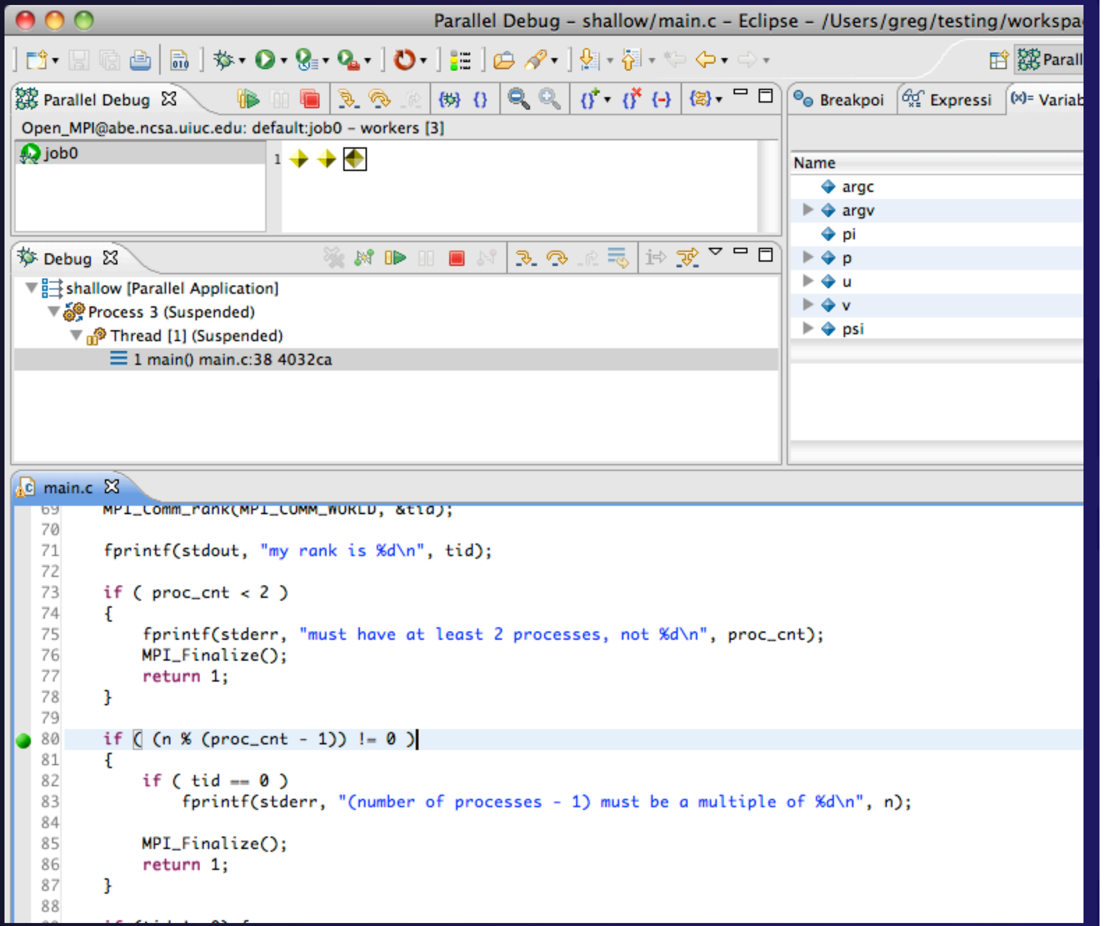


```
else{
    print
MPI_Final
```



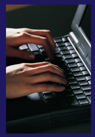
Creating A Breakpoint

- Select the process set that the breakpoint should apply to, in this case, the **workers** set
- Double-click on the left edge of an editor window, at the line on which you want to set the breakpoint, or right click and use the **Parallel Breakpoint▶Toggle Breakpoint** context menu
- The breakpoint is displayed on the marker bar


Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspa
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]
job0
Debug
Breakpoi
Expressi
Variab
Name
argc
argv
pi
p
u
v
psi
main.c

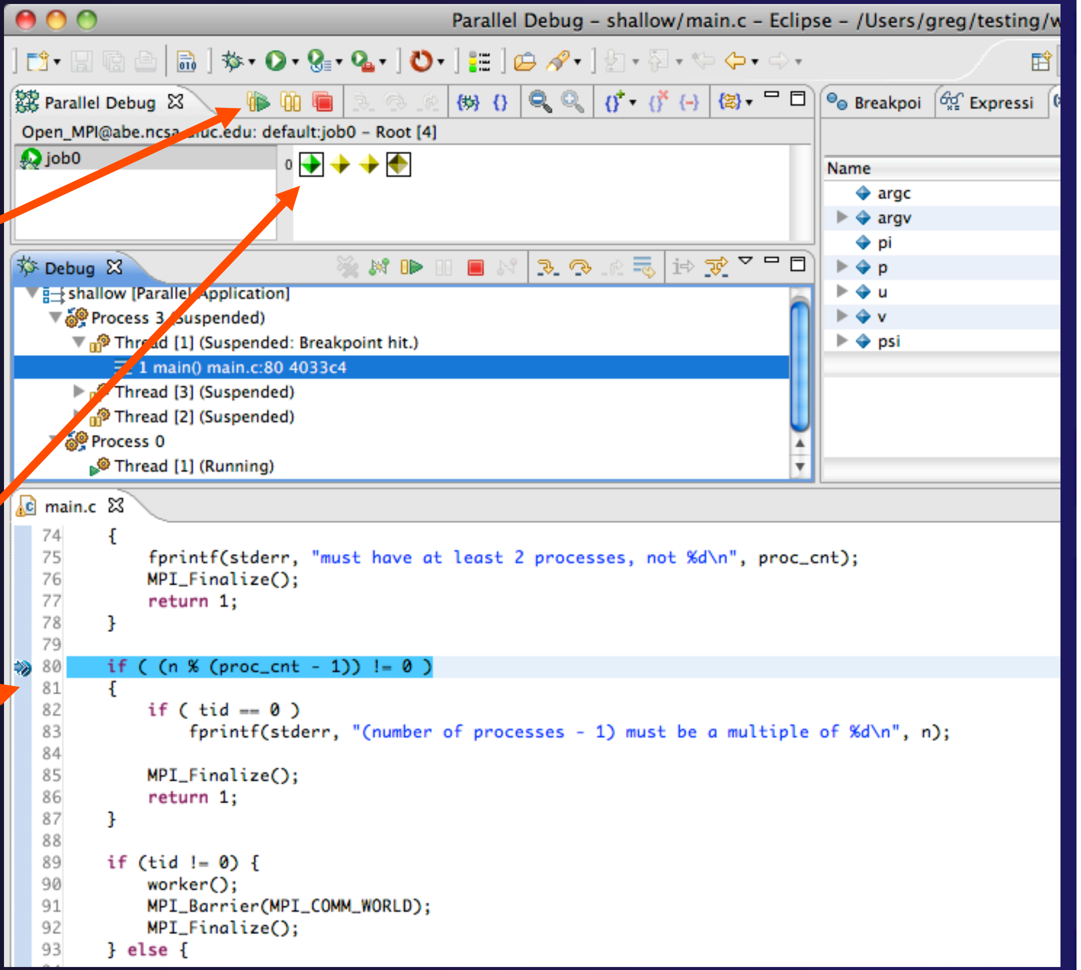
```
shallow [Parallel Application]
  Process 3 (Suspended)
    Thread [1] (Suspended)
      1 main() main.c:38 4032ca
```

```
    MPI_Comm_rank(MPI_COMM_WORLD, &tid);

    fprintf(stdout, "my rank is %d\n", tid);

    if ( proc_cnt < 2 )
    {
        fprintf(stderr, "must have at least 2 processes, not %d\n", proc_cnt);
        MPI_Finalize();
        return 1;
    }

    if ( (n % (proc_cnt - 1)) != 0 )
    {
        if ( tid == 0 )
            fprintf(stderr, "(number of processes - 1) must be a multiple of %d\n", n);

        MPI_Finalize();
        return 1;
    }
```



Hitting the Breakpoint

- Switch back to the **Root** set by clicking on the **Change Set** button
- Click on the **Resume** button in the **Parallel Debug view**
- In this example, the three worker processes have hit the breakpoint, as indicated by the yellow process icons and the current line marker
- Process 0 is still running as its icon is green
- Processes 1-3 are suspended on the breakpoint



Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/w

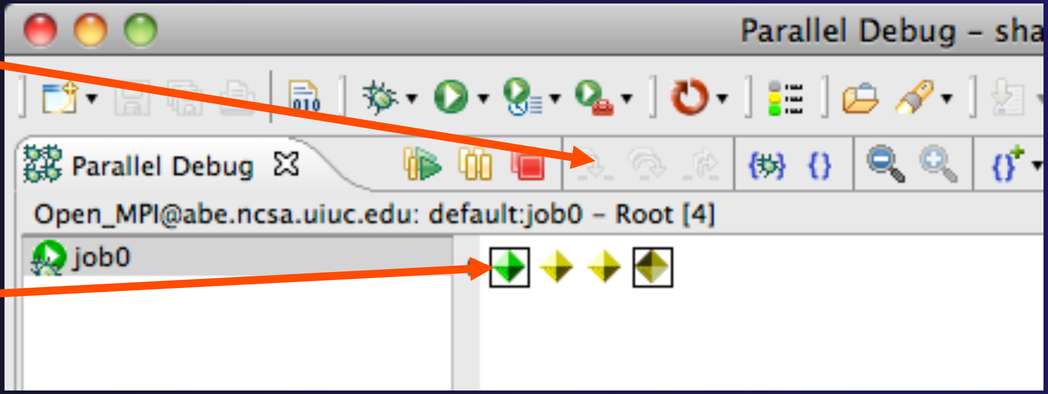
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

```
shallow [Parallel Application]
  Process 3 (Suspended)
    Thread [1] (Suspended: Breakpoint hit.)
      1 main() main.c:80 4033c4
    Thread [3] (Suspended)
    Thread [2] (Suspended)
  Process 0
    Thread [1] (Running)
```

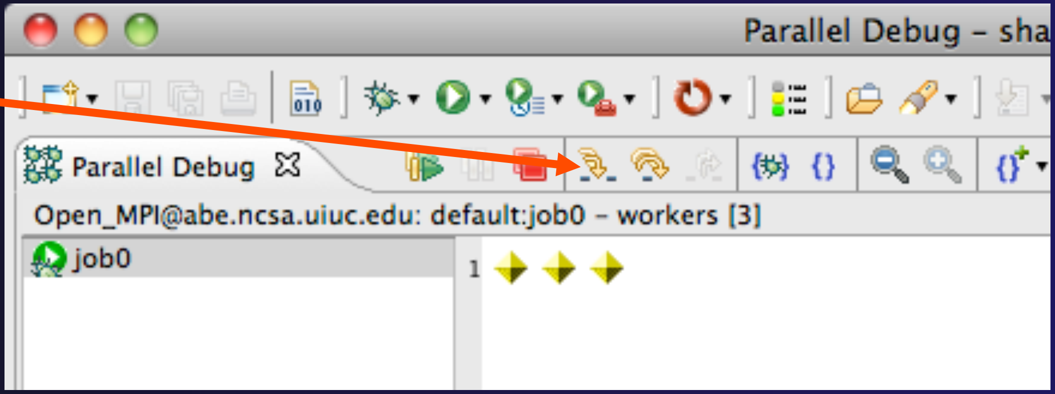
```
	{
		fprintf(stderr, "must have at least 2 processes, not %d\n", proc_cnt);
		MPI_Finalize();
		return 1;
	}

	if ( (n % (proc_cnt - 1)) != 0 )
	{
		if ( tid == 0 )
			fprintf(stderr, "(number of processes - 1) must be a multiple of %d\n", n);

		MPI_Finalize();
		return 1;
	}

	if (tid != 0) {
		worker();
		MPI_Barrier(MPI_COMM_WORLD);
		MPI_Finalize();
	} else {
```

More On Stepping

- The **Step** buttons are only enabled when all processes in the active set are **suspended** (yellow icon)
- In this case, process 0 is still running

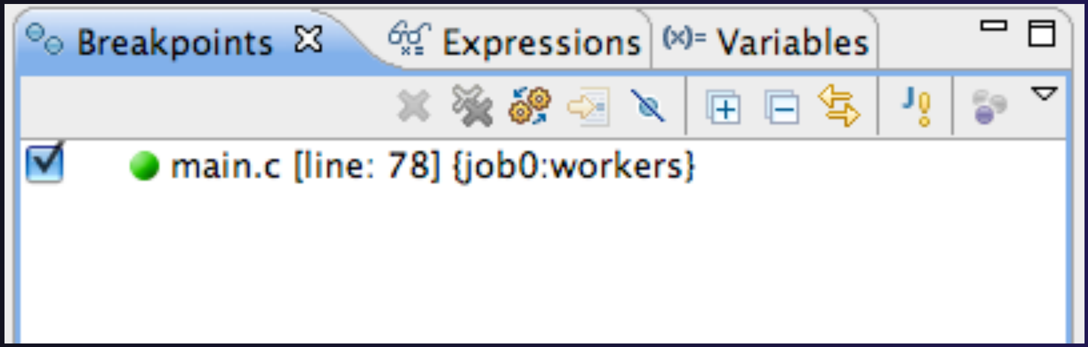


- Switch to the set of suspended processes (the **workers** set)
- You will now see the **Step** buttons become enabled



Breakpoint Information

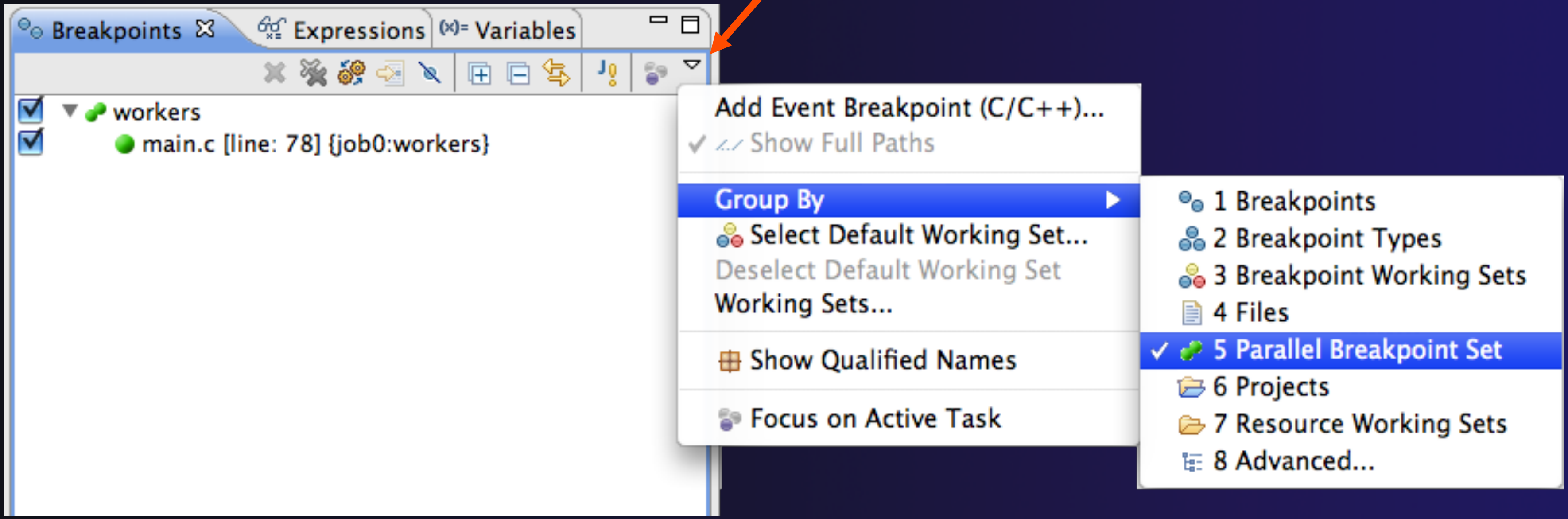
- Hover over breakpoint icon
  - Will show the sets this breakpoint applies to
- Select **Breakpoints** view
  - Will show all breakpoints in all projects



main.c [line: 78] {job0:workers}

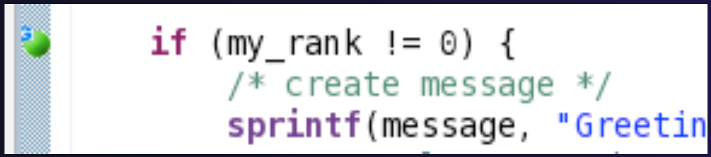
Breakpoints View

- Use the menu in the breakpoints view to group breakpoints by type
- Breakpoints sorted by breakpoint set (process set)



Global Breakpoints

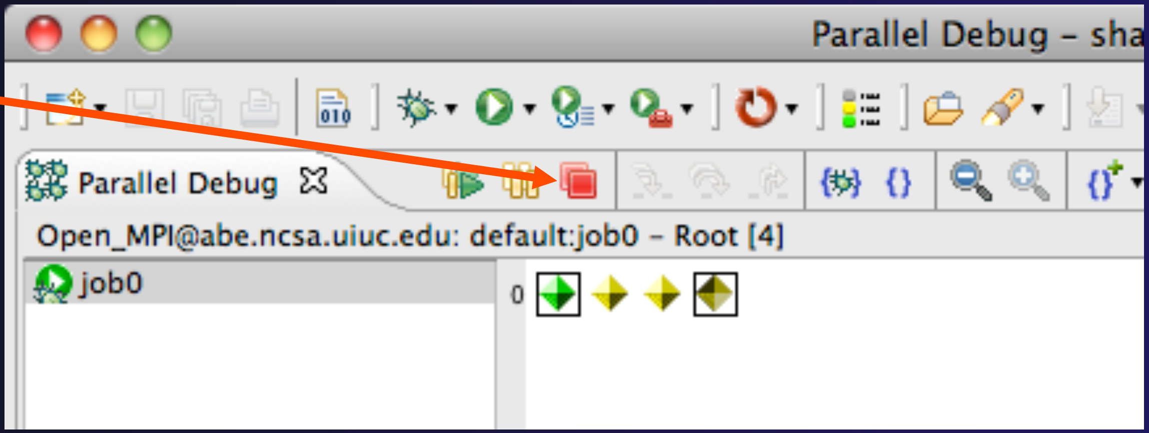
- Apply to all processes and all jobs
- Used for gaining control at debugger startup
- To create a global breakpoint
  - First make sure that no jobs are selected (click in white part of jobs view if necessary)
  - Double-click on the left edge of an editor window
  - Note that if a job is selected, the breakpoint will apply to the current set



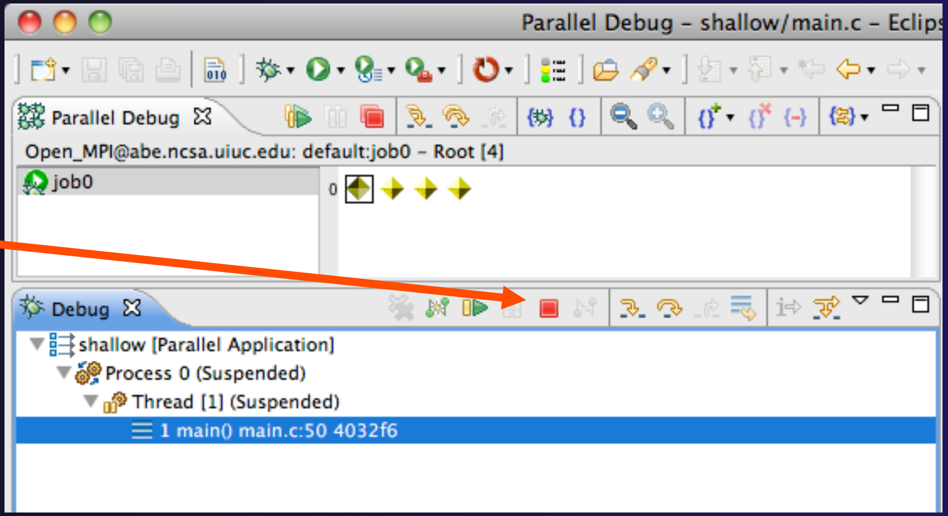
```
    if (my_rank != 0) {
        /* create message */
        sprintf(message, "Greetin
```

Terminating A Debug Session

- Click on the **Terminate** icon in the **Parallel Debug view** to terminate all processes in the active set
- Make sure the **Root** set is active if you want to terminate all processes



- You can also use the terminate icon in the **Debug** view to terminate the currently selected process

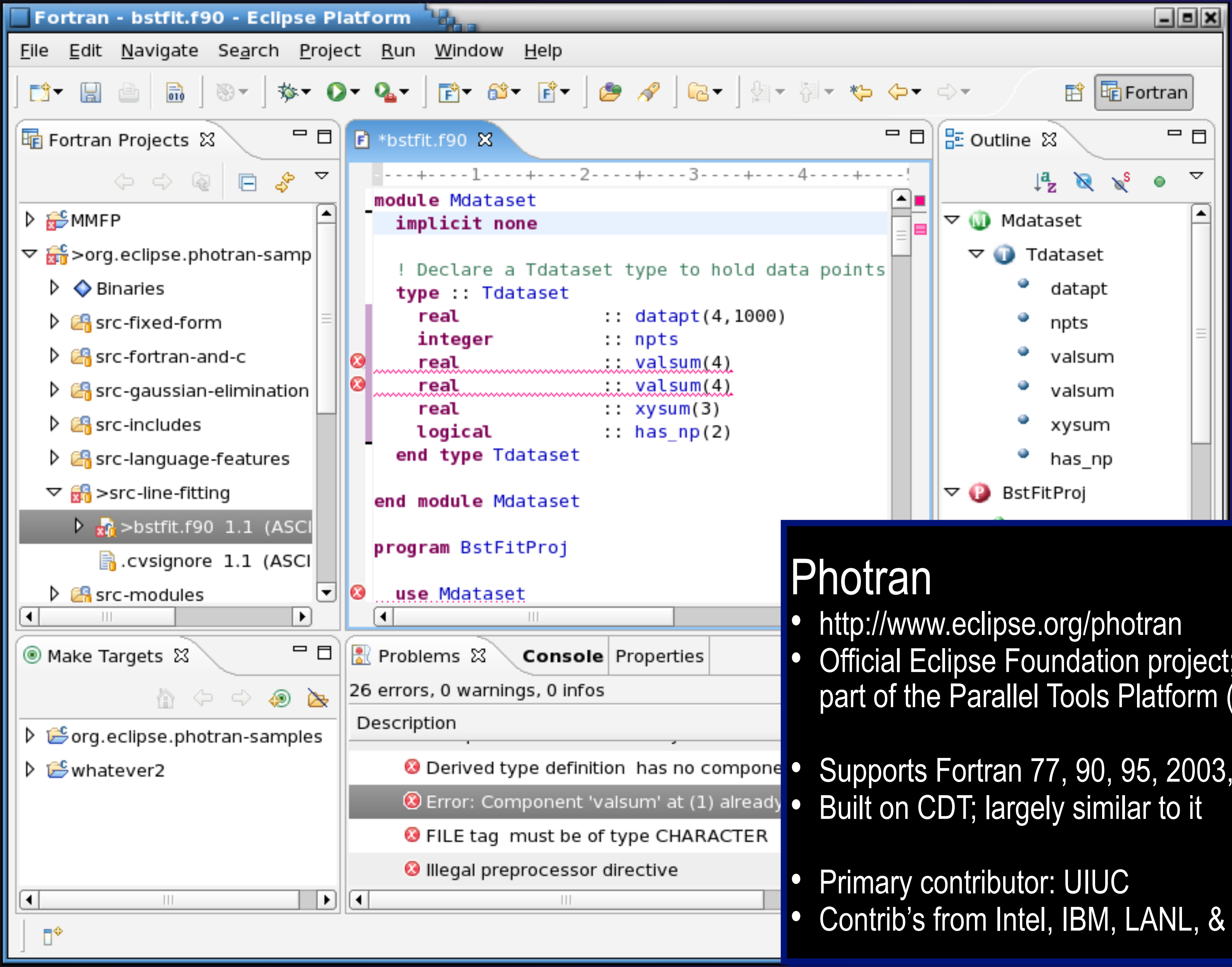

Parallel Debug – shallow/main.c – Eclips
Parallel Debug
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]
job0
0
Debug
shallow [Parallel Application]
Process 0 (Suspended)
Thread [1] (Suspended)
1 main() main.c:50 4032f6

Module 6: Fortran

- Objective
  - Learn what Photran is and how it compares to CDT
  - Learn how to create a Fortran MPI application
  - Learn about refactoring support
- Contents
  - Overview of Photran
  - Module 3 redux (in Fortran)
  - Differences between Photran and CDT
  - Pointers to online documentation for Photran
  - Refactoring support

Ralph Johnson’s research group at UIUC used to meet at Pho-Tran…

…which became the name of their Fortran IDE.



```
module Mdataset
  implicit none

  ! Declare a Tdataset type to hold data points
  type :: Tdataset
    real              :: datapt(4,1000)
    integer           :: npts
    real              :: valsum(4)
    real              :: valsum(4)
    real              :: xysum(3)
    logical           :: has_np(2)
  end type Tdataset

end module Mdataset

program BstFitProj

  use Mdataset
```

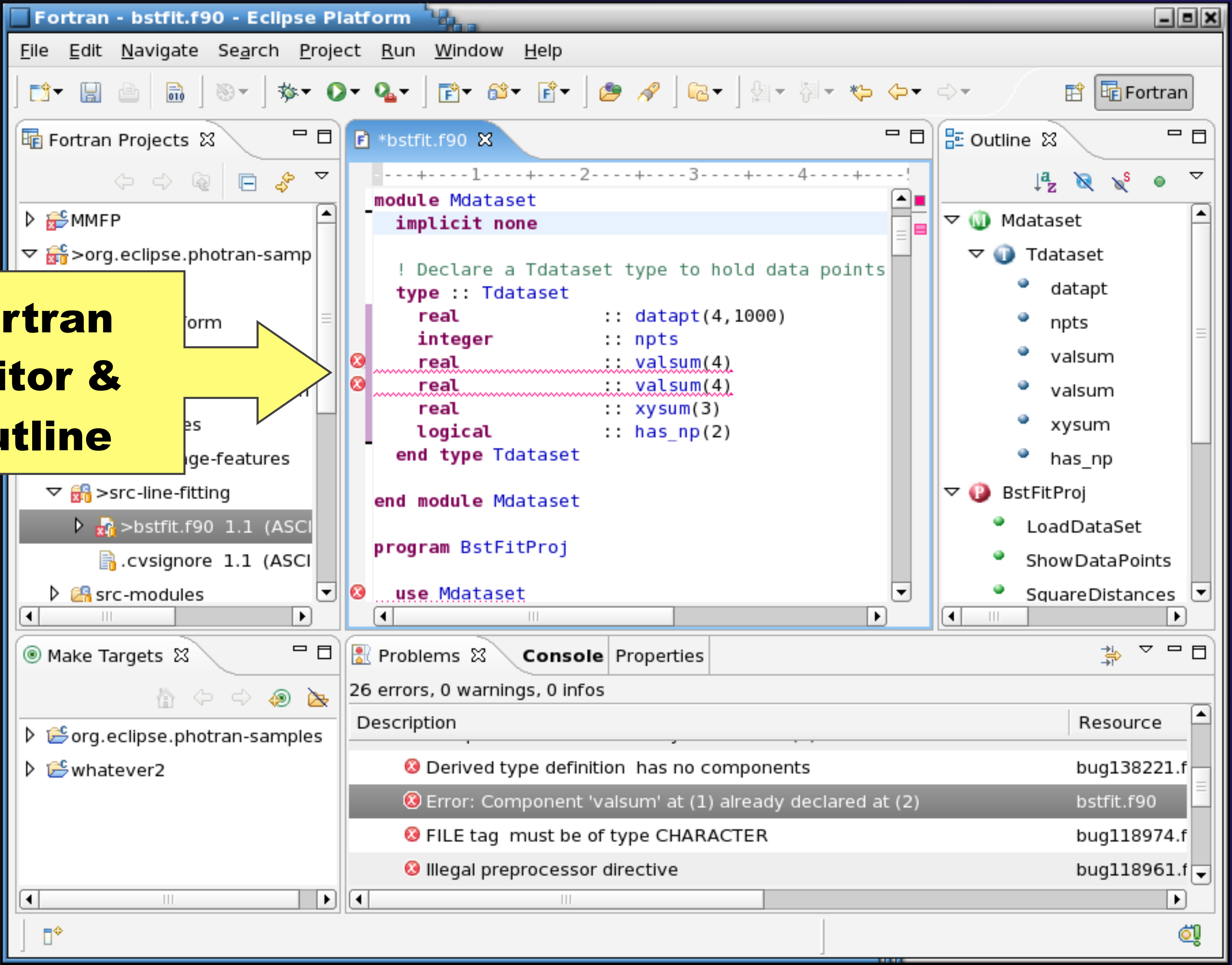
```
26 errors, 0 warnings, 0 infos
Derived type definition  has no compone
Error: Component 'valsum' at (1) already
FILE tag  must be of type CHARACTER
Illegal preprocessor directive
```

Photran

- http://www.eclipse.org/photran
- Official Eclipse Foundation project; part of the Parallel Tools Platform (PTP)

- Supports Fortran 77, 90, 95, 2003, & 2008
- Built on CDT; largely similar to it

- Primary contributor: UIUC
- Contrib's from Intel, IBM, LANL, & others

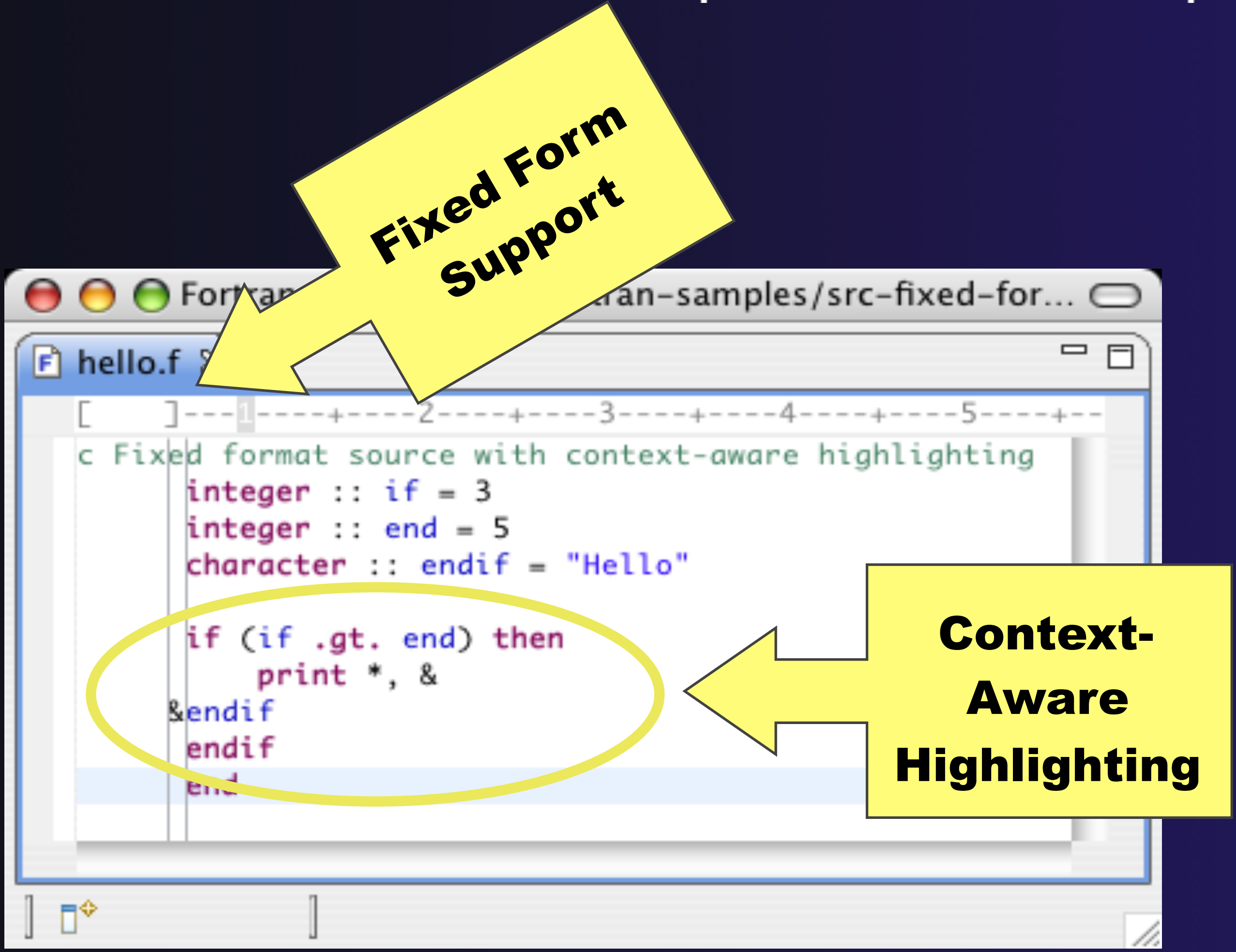


Fortran Editor & Outline

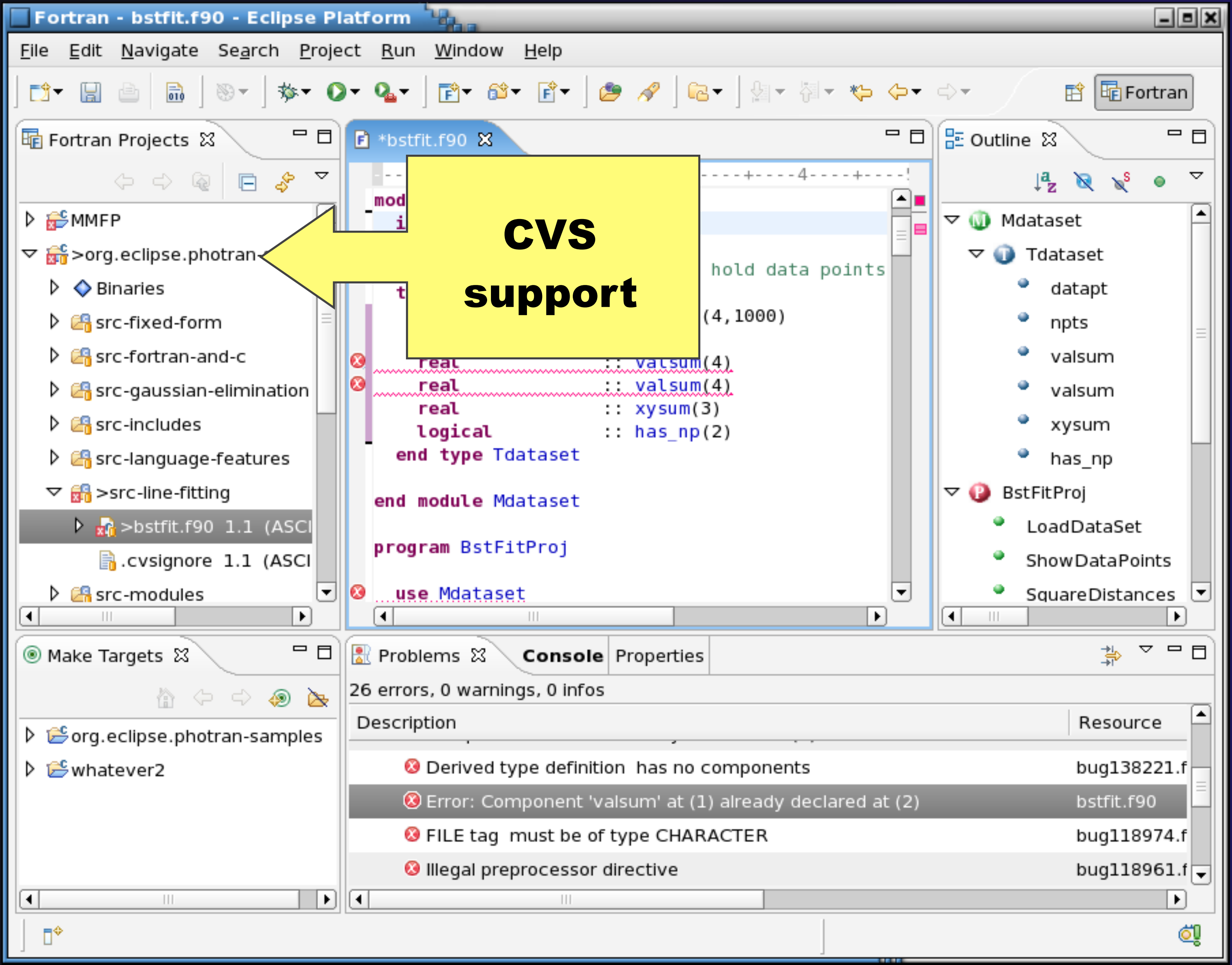
```
module Mdataset
  implicit none

  ! Declare a Tdataset type to hold data points
  type :: Tdataset
    real              :: datapt(4,1000)
    integer           :: npts
    real              :: valsum(4)
    real              :: valsum(4)
    real              :: xysum(3)
    logical           :: has_np(2)
  end type Tdataset

end module Mdataset

program BstFitProj

  use Mdataset
```

26 errors, 0 warnings, 0 infos

Description	Resource
Derived type definition has no components	bug138221.f
Error: Component 'valsum' at (1) already declared at (2)	bstfit.f90
FILE tag must be of type CHARACTER	bug118974.f
Illegal preprocessor directive	bug118961.f



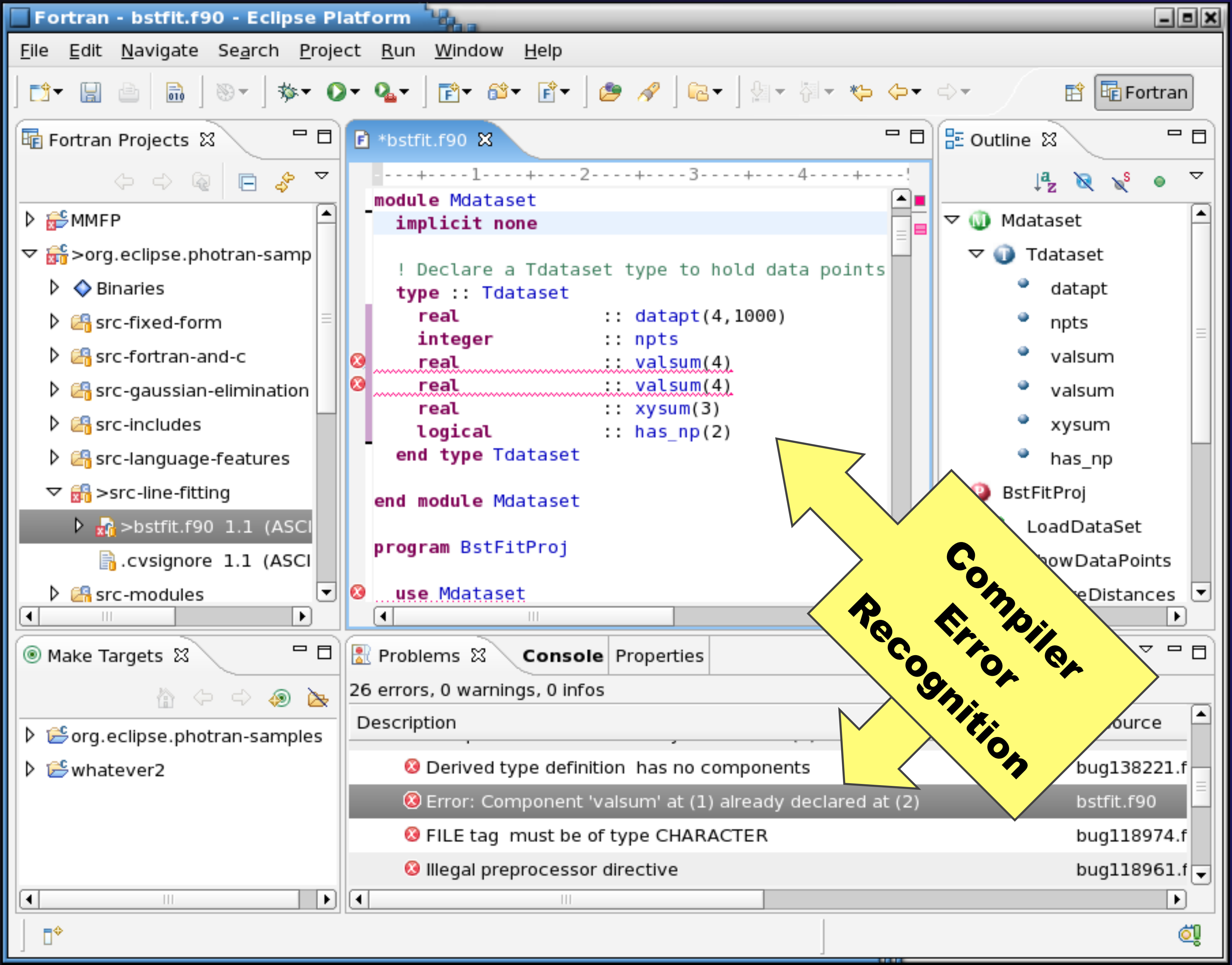
```
[      ]---1----+----2----+----3----+----4----+----5----+--
c Fixed format source with context-aware highlighting
      integer :: if = 3
      integer :: end = 5
      character :: endif = "Hello"

      if (if .gt. end) then
          print *, &
     &endif
      endif
      end
```



```
mod
  i
                    hold data points
  t
                    (4,1000)
    real            :: valsum(4)
    real            :: valsum(4)
    real            :: xysum(3)
    logical         :: has_np(2)
  end type Tdataset

end module Mdataset

program BstFitProj

  use Mdataset
```

26 errors, 0 warnings, 0 infos

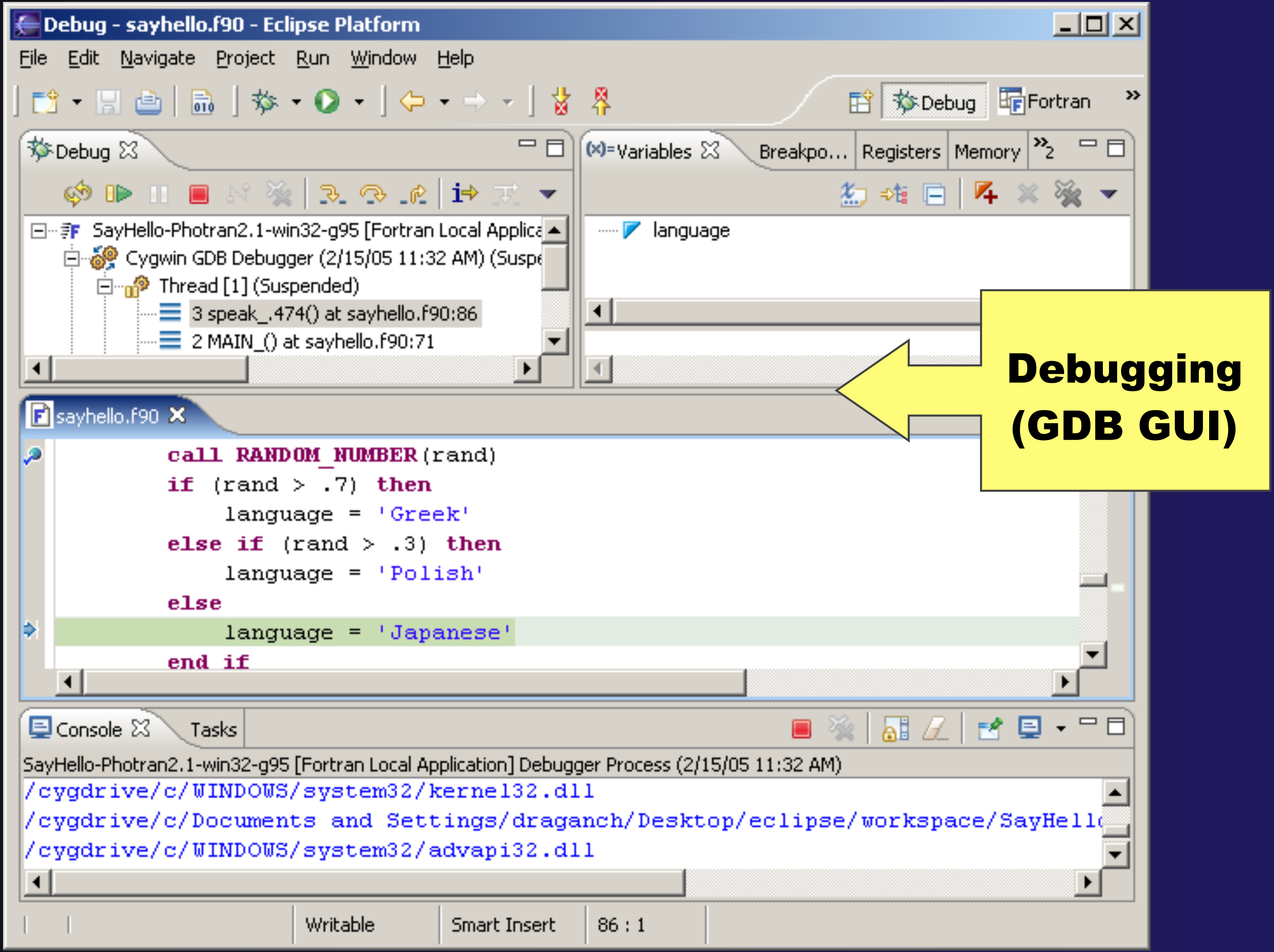
Description	Resource
Derived type definition has no components	bug138221.f
Error: Component 'valsum' at (1) already declared at (2)	bstfit.f90
FILE tag must be of type CHARACTER	bug118974.f
Illegal preprocessor directive	bug118961.f


Fortran - bstfit.f90 - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Fortran Projects
Outline
Make Targets
Problems Console Properties
Compiler Error Recognition

```
module Mdataset
  implicit none

  ! Declare a Tdataset type to hold data points
  type :: Tdataset
    real          :: datapt(4,1000)
    integer       :: npts
    real          :: valsum(4)
    real          :: valsum(4)
    real          :: xysum(3)
    logical       :: has_np(2)
  end type Tdataset

end module Mdataset

program BstFitProj

  use Mdataset
```

26 errors, 0 warnings, 0 infos

Description	Resource
Derived type definition has no components	bug138221.f
Error: Component 'valsum' at (1) already declared at (2)	bstfit.f90
FILE tag must be of type CHARACTER	bug118974.f
Illegal preprocessor directive	bug118961.f



SayHello-Photran2.1-win32-g95 [Fortran Local Applica
Cygwin GDB Debugger (2/15/05 11:32 AM) (Suspe
Thread [1] (Suspended)
3 speak_.474() at sayhello.f90:86
2 MAIN_() at sayhello.f90:71

language

```
        call RANDOM_NUMBER(rand)
        if (rand > .7) then
            language = 'Greek'
        else if (rand > .3) then
            language = 'Polish'
        else
            language = 'Japanese'
        end if
```

SayHello-Photran2.1-win32-g95 [Fortran Local Application] Debugger Process (2/15/05 11:32 AM)

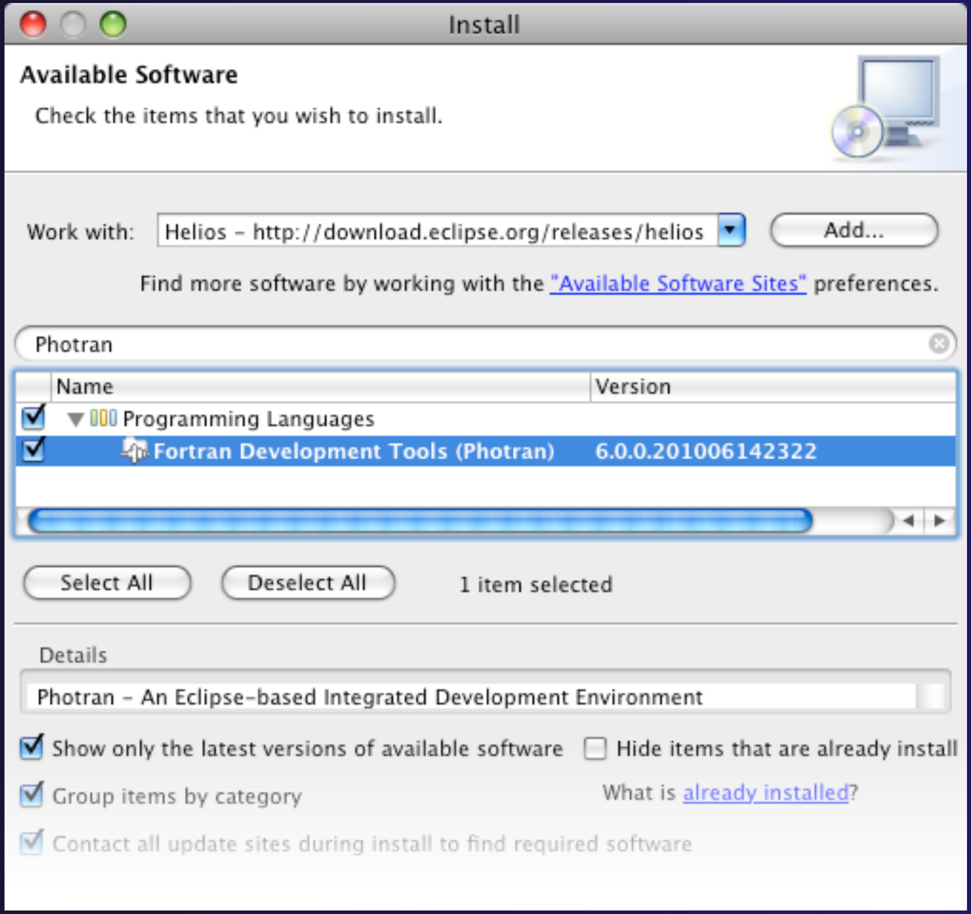
```
/cygdrive/c/WINDOWS/system32/kernel32.dll
/cygdrive/c/Documents and Settings/draganch/Desktop/eclipse/workspace/SayHello
/cygdrive/c/WINDOWS/system32/advapi32.dll
```

Debugging (GDB GUI)

Installing Photran

http://wiki.eclipse.org/PTP/photran/documentation/photran7installation

- You will need a Fortran compiler (e.g., gfortran), make, and gdb to compile & debug Fortran programs
- From the **Help** menu, choose **Install New Software…**
- Select the Indigo update site
- Under Programming Langs Check Fortran Dev. Tools
- Click **Next**
- Finish installing:
  - **Next,** Accept license, **Finish**
  - Features and prerequisites are downloaded and installed…
- Restart Eclipse when prompted



Available Software

Check the items that you wish to install.

Work with: Helios - http://download.eclipse.org/releases/helios

Find more software by working with the "Available Software Sites" preferences.

Photran

Name	Version
✓ Programming Languages	
✓ Fortran Development Tools (Photran)	6.0.0.201006142322

1 item selected

Details

Photran - An Eclipse-based Integrated Development Environment

✓ Show only the latest versions of available software
☐ Hide items that are already install
✓ Group items by category
What is already installed?
✓ Contact all update sites during install to find required software

Using Photran

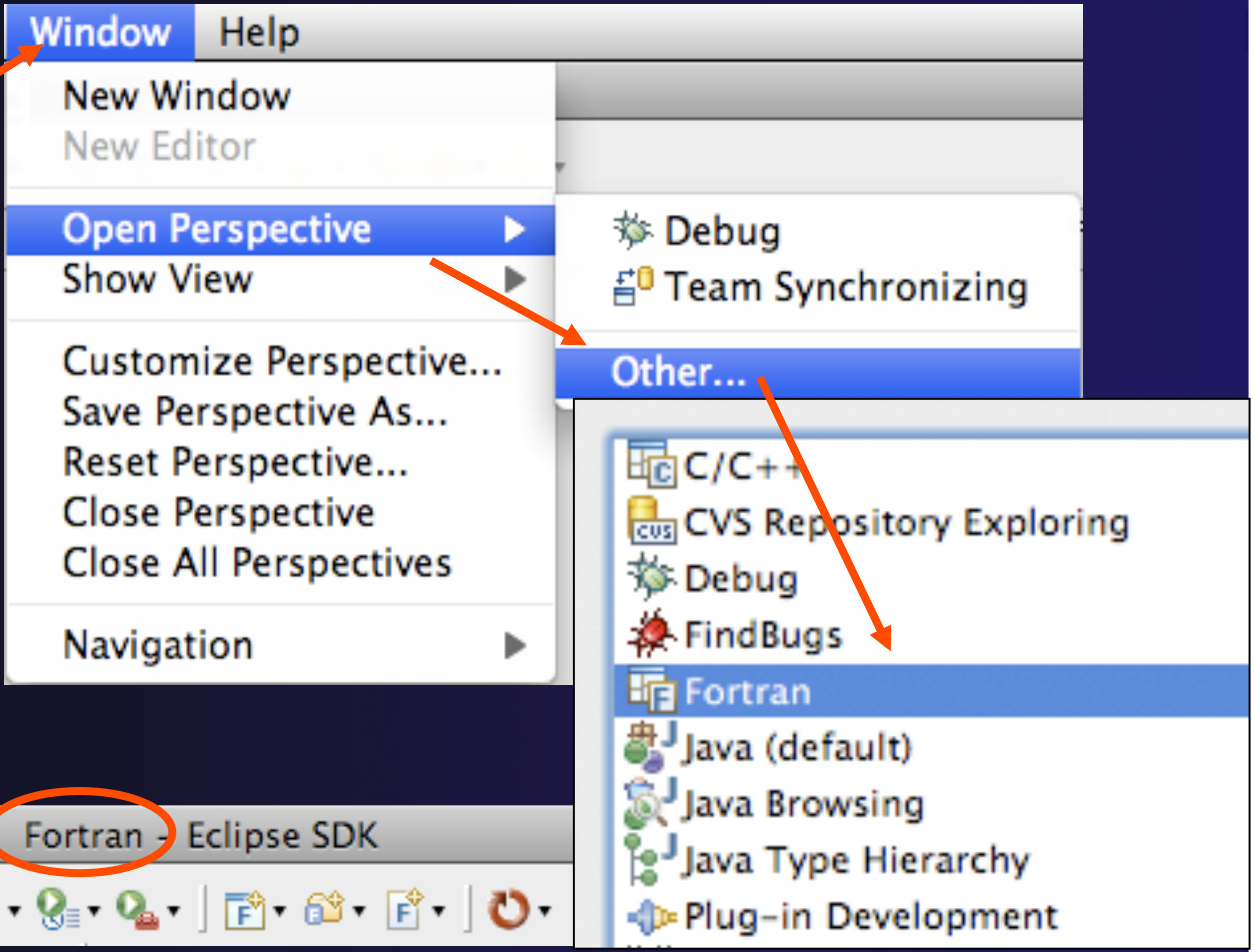
- It’s just like using CDT…
  - Similar New Project wizards
  - Similar build procedure
  - Similar launch/debug procedure
- …but not exactly
  - Remote development partially supported
  - Configuring fixed vs. free form file extensions
  - Different editor features
  - Different advanced features

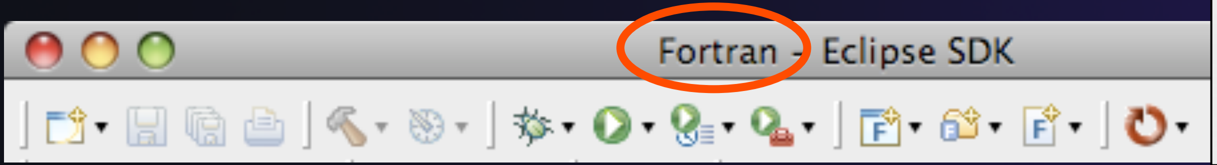
Switch to ~~C/C++~~ Fortran Perspective

(same as for C/C++)

- Only needed if you're not already in the perspective

- What Perspective am in in?
  See Title Bar

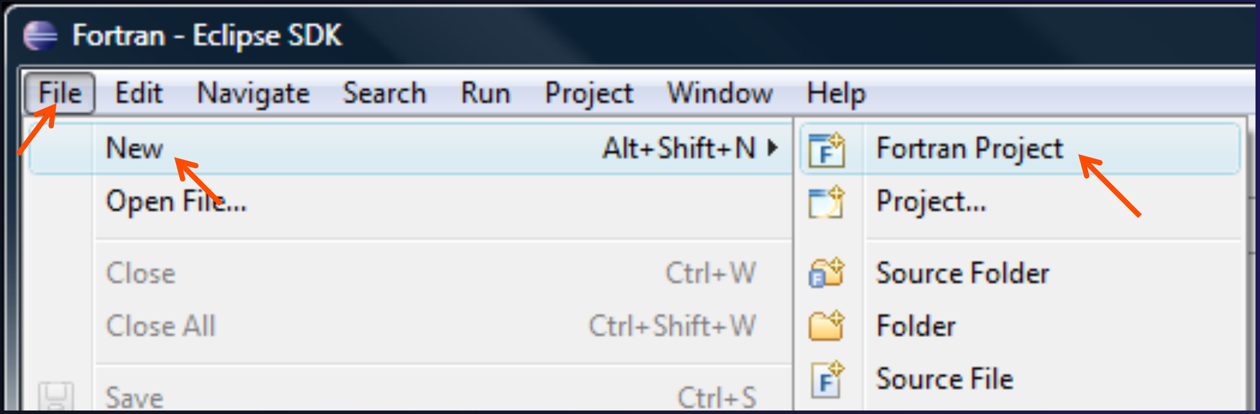




Creating a Fortran Application
(same as Creating a C/C++ Application)

Steps:

- Create a new Fortran project
- Edit source code
- Save and build

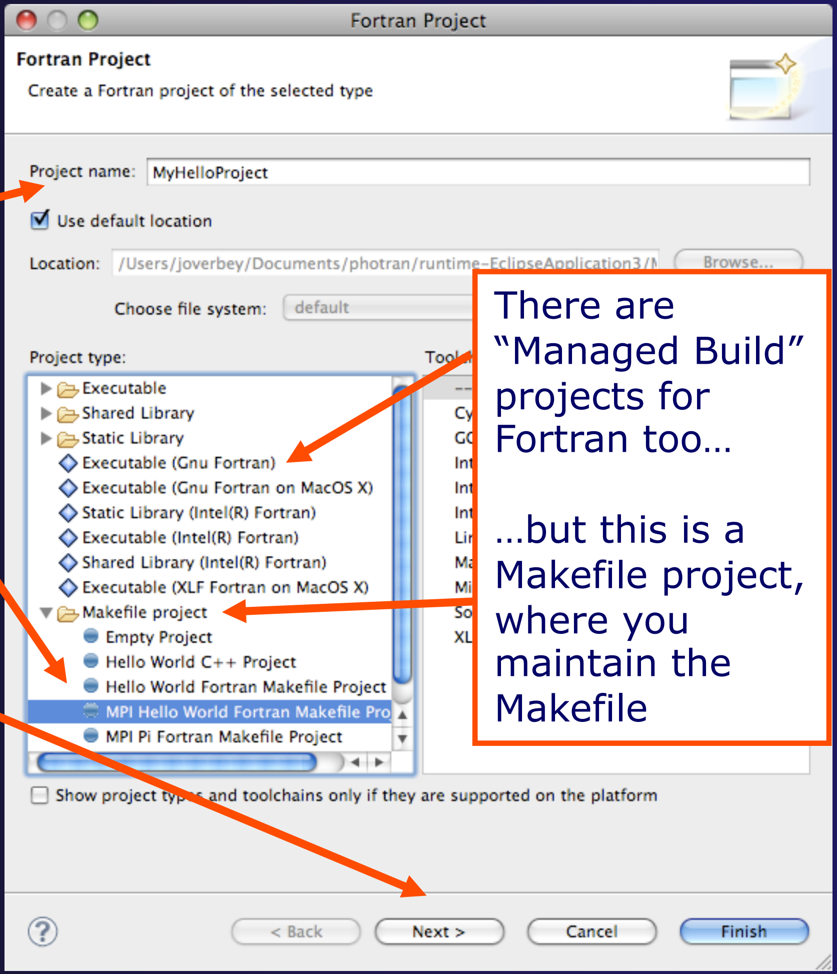


New Fortran Project Wizard

(similar to New C/C++ Project Wizard)

Create a new MPI project

- **File▸New▸Fortran Project** (see prev. slide)
- Name the project 'MyHelloProject'
- Under Project types, under Makefile Project, select **MPI Hello World Fortran Project** and hit **Next**
- On **Basic Settings** page, fill in information for your new project (**Author name** etc.) and hit **Finish**



Fortran Project

Fortran Project
Create a Fortran project of the selected type

Project name: MyHelloProject
[✓] Use default location
Location: /Users/joverbey/Documents/photran/runtime-EclipseApplication3/M
Choose file system: default

Project type:
- Executable
- Shared Library
- Static Library
- Executable (Gnu Fortran)
- Executable (Gnu Fortran on MacOS X)
- Static Library (Intel(R) Fortran)
- Executable (Intel(R) Fortran)
- Shared Library (Intel(R) Fortran)
- Executable (XLF Fortran on MacOS X)
- Makefile project
  - Empty Project
  - Hello World C++ Project
  - Hello World Fortran Makefile Project
  - MPI Hello World Fortran Makefile Pro
  - MPI Pi Fortran Makefile Project

[ ] Show project types and toolchains only if they are supported on the platform

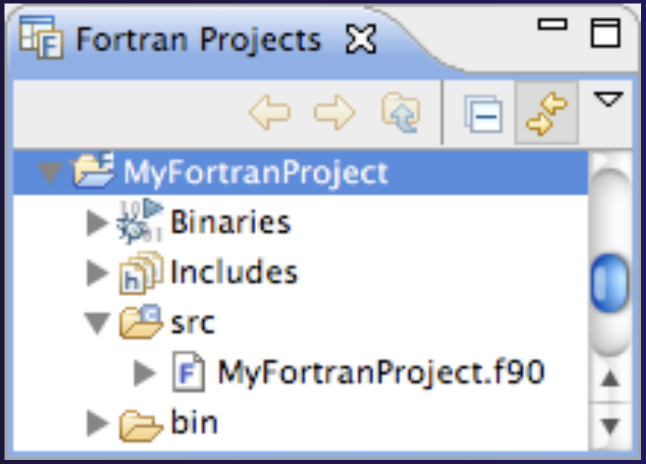
There are "Managed Build" projects for Fortran too…

…but this is a Makefile project, where you maintain the Makefile

Fortran Projects View

(similar to C/C++ Project Explorer view)

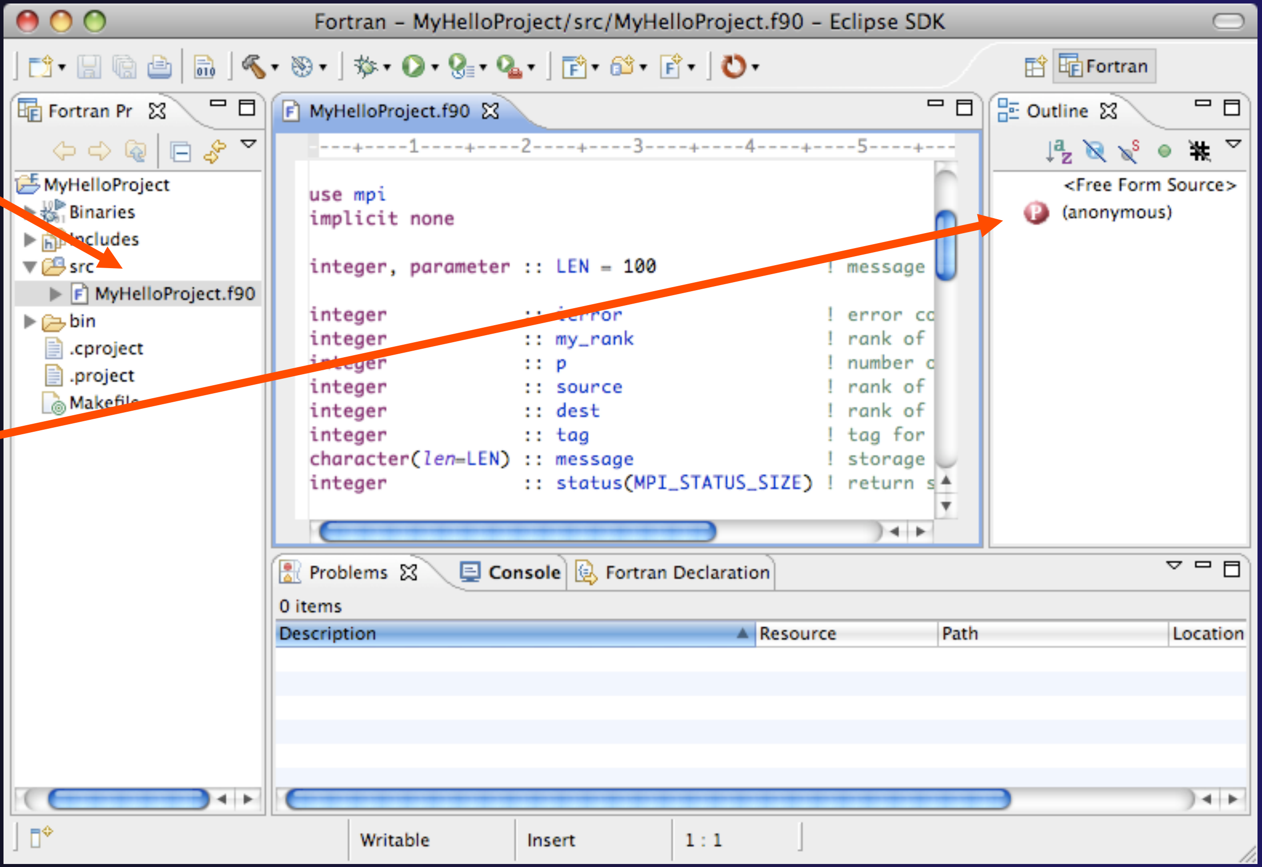
- Represents user's data
- It is a set of user defined resources
  - Files
  - Folders
  - Projects
    - Collections of files and folders
    - Plus meta-data
- Resources are visible in the Fortran Projects View



Editor and Outline View

(similar to C/C++)

- Double-click on source file to open Fortran editor
- Outline view is shown for file in editor

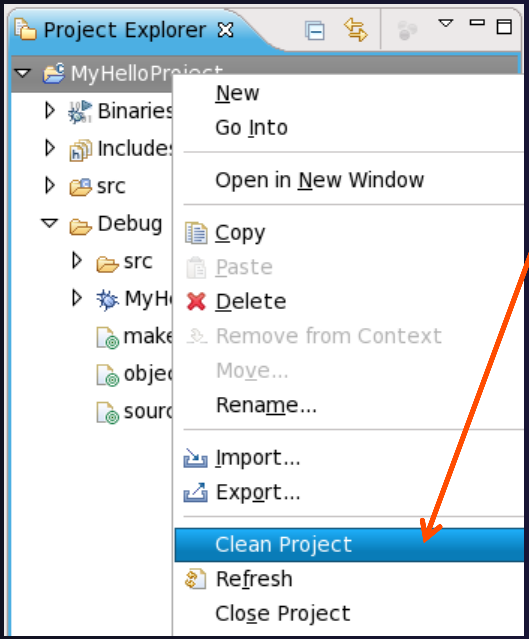


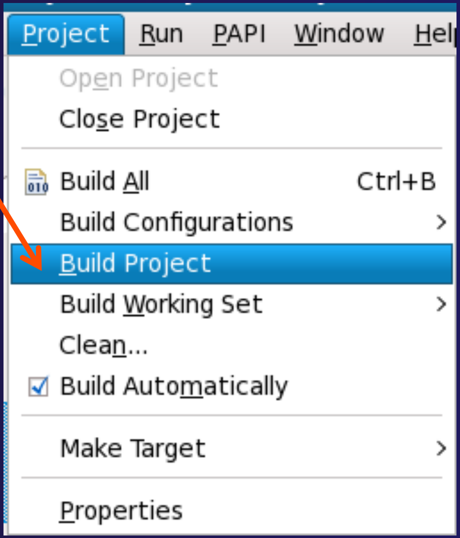
```
use mpi
implicit none

integer, parameter :: LEN = 100               ! message

integer            :: error                   ! error co
integer            :: my_rank                 ! rank of
integer            :: p                       ! number o
integer            :: source                  ! rank of
integer            :: dest                    ! rank of
integer            :: tag                     ! tag for
character(len=LEN) :: message                 ! storage
integer            :: status(MPI_STATUS_SIZE) ! return s
```

Build

(same as C/C++)

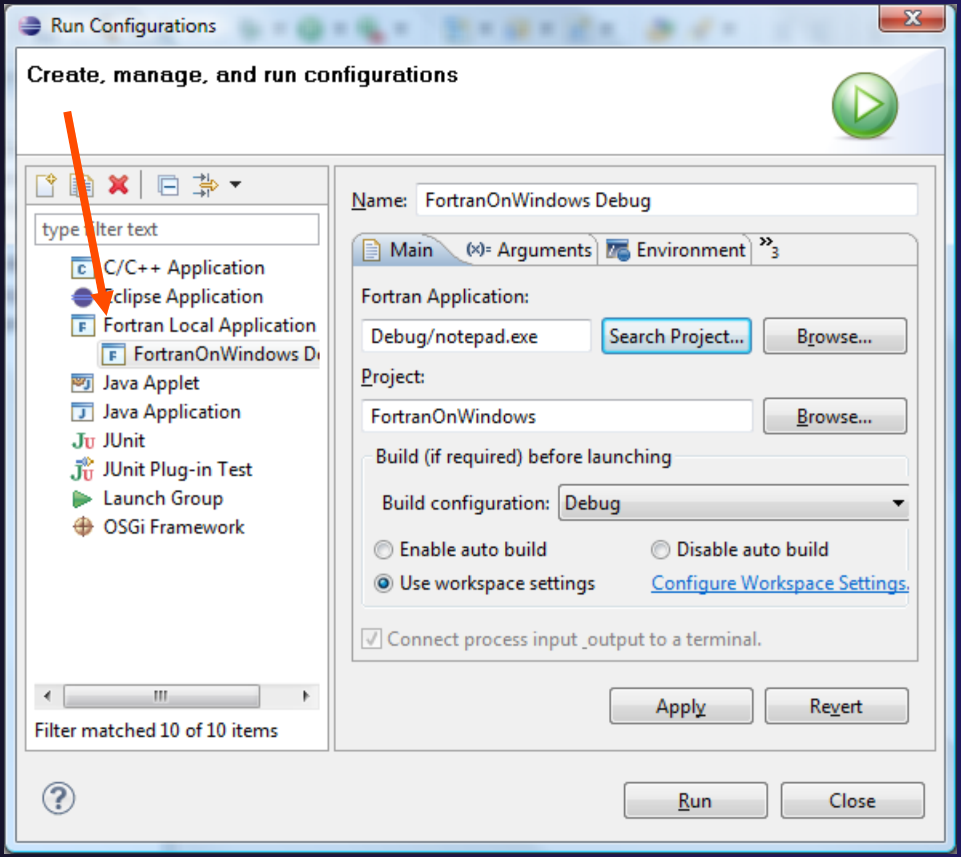
- Your program should build when created.
- To rebuild, many ways include:
  - Select project, Hit hammer icon in toolbar
  - Select project, **Project ▶ Build Project**
  - Right mouse on project, **Clean Project**





Et Cetera

- Creating a launch configuration is identical (Suggestion: Uncheck **Stop on startup at main** in the Debugger tab)



Run Configurations

Create, manage, and run configurations

type filter text

- C/C++ Application
- Eclipse Application
- Fortran Local Application
  - FortranOnWindows D
- Java Applet
- Java Application
- JUnit
- JUnit Plug-in Test
- Launch Group
- OSGi Framework

Filter matched 10 of 10 items

Name: FortranOnWindows Debug

Main | Arguments | Environment | »3

Fortran Application: Debug/notepad.exe [Search Project...] [Browse...]

Project: FortranOnWindows [Browse...]

Build (if required) before launching

Build configuration: Debug

Enable auto build | Disable auto build

[✓ Use workspace settings] | Configure Workspace Settings...

[✓] Connect process input _output to a terminal.

[Apply] [Revert]

[Run] [Close]

Et Cetera

- Debugging is identical
- Launching a parallel application is identical
- Debugging a parallel application is identical

Diagnosing Common Problems
(also true for C/C++)

Building: *Are compile errors not shown in the Problems view?*

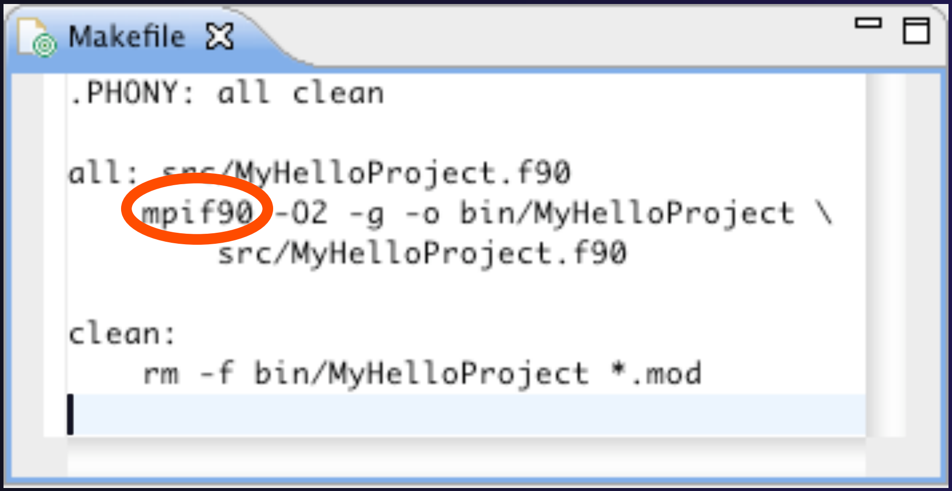
- Right-click on the project in the Fortran Projects view, and choose **Properties**
- Expand **Fortran Build▶Settings**
- Switch to the **Error Parsers** tab
- Are Photran's error parsers checked? If not, click **Check all**
- Click **OK** and re-build

Launching: *Is a binary not listed when creating a launch configuration?*

- Right-click on the project in the Fortran Projects view, and choose **Properties**
- Expand **Fortran Build▶Settings**
- Switch to the **Binary Parsers** tab
- Make sure the parser for your platform is checked
  - PE = Windows
  - Elf = Linux
  - Mach-O = Mac OS X
- Click **OK**

Differences (1): MPI Project Wizard

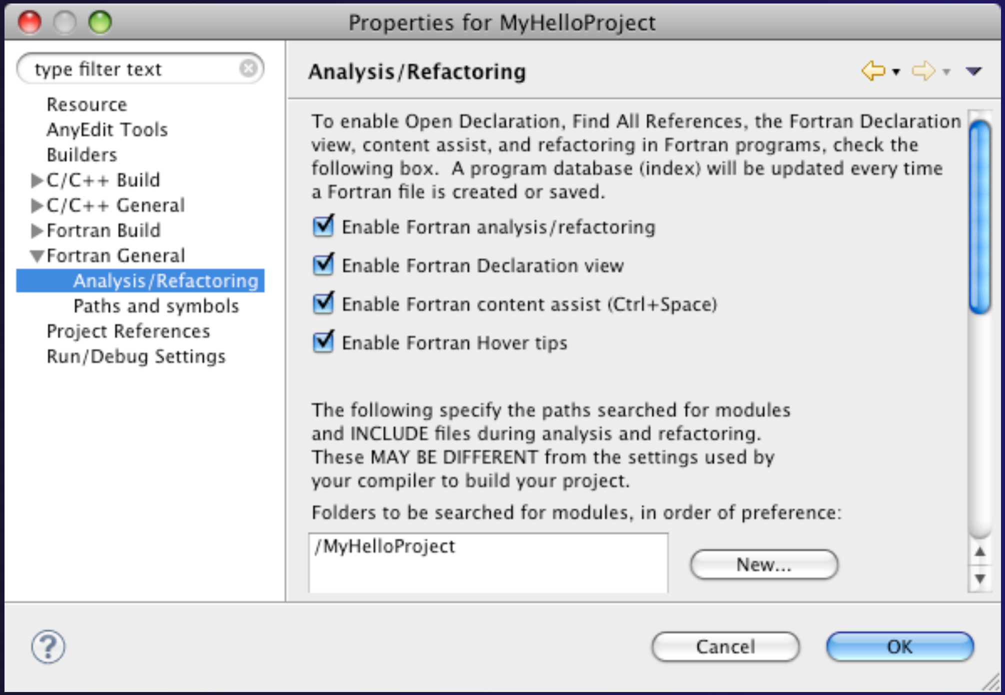
- In the MPI Hello World C Project (local project), the MPI compiler is set in the project settings… (Local, managed build project: see Module 7, Advanced Features)
- …but in the MPI Hello World Fortran Project, the MPI compiler is set in a Makefile.



```
.PHONY: all clean

all: src/MyHelloProject.f90
	mpif90 -O2 -g -o bin/MyHelloProject \
		src/MyHelloProject.f90

clean:
	rm -f bin/MyHelloProject *.mod
```

Differences (2): Content Assist

- Content assist is *disabled* by default.
  (So are Declaration View, Hover Tips, Fortran Search, & refactorings.)
  You must specifically enable it for your project.
  - Right-click on the project in the Fortran Projects view, and choose **Properties**
  - Expand **Fortran ▶ Analysis/Refactoring**
  - Check **Enable Fortran analysis/refactoring**
  - Click **OK**
  - Close and re-open any Fortran editors



To enable Open Declaration, Find All References, the Fortran Declaration view, content assist, and refactoring in Fortran programs, check the following box. A program database (index) will be updated every time a Fortran file is created or saved.

[✓] Enable Fortran analysis/refactoring

[✓] Enable Fortran Declaration view

[✓] Enable Fortran content assist (Ctrl+Space)

[✓] Enable Fortran Hover tips

The following specify the paths searched for modules and INCLUDE files during analysis and refactoring. These MAY BE DIFFERENT from the settings used by your compiler to build your project.

Folders to be searched for modules, in order of preference:

/MyHelloProject

Differences (3): Source Form

- Fortran files are either *free form* or *fixed form;* some Fortran files are *preprocessed* (#define, #ifdef, etc.)
  - Determined by filename extension
  - Source form is set in the project properties

  - Defaults:

Fixed form:	.f	.fix	.for	.fpp	.ftn	.f77	
Free form:	.f08	.f03	.f95	.f90			< unpreprocessed
	.F08		.F03	.F95	.F90		< preprocessed

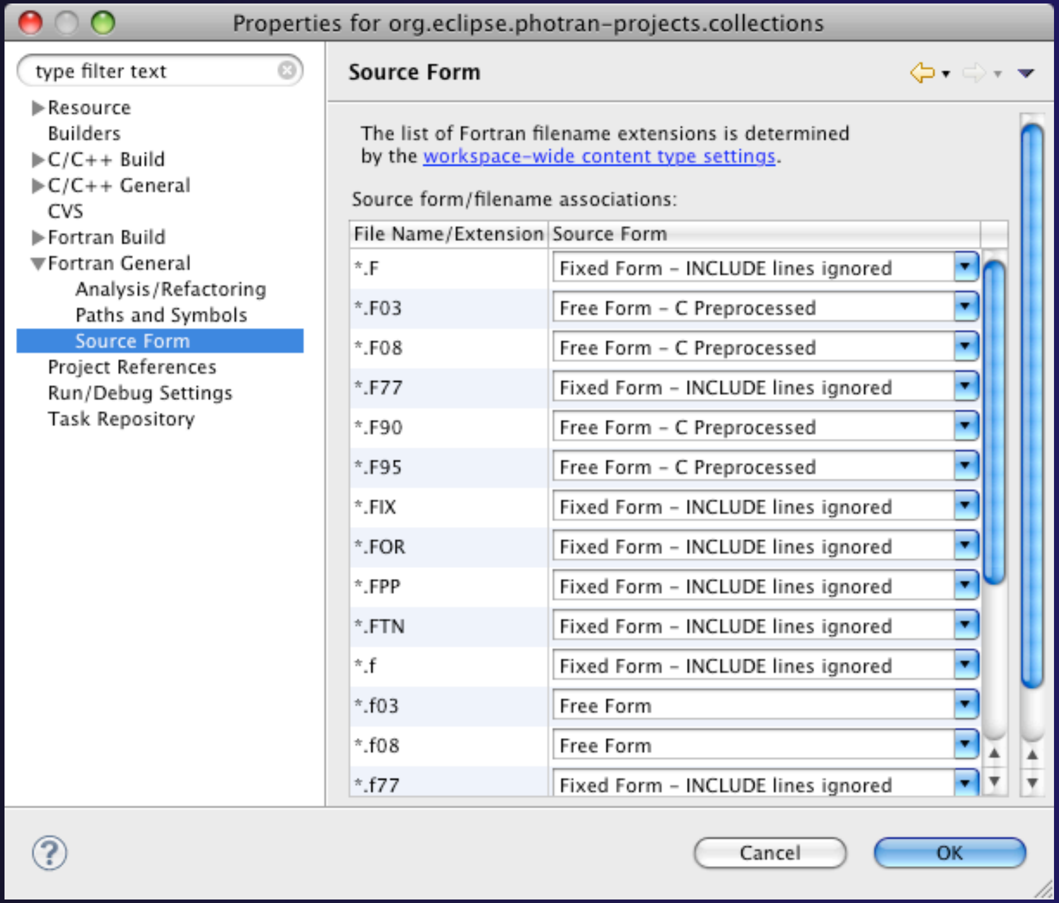
- Many features *will not work* if filename extensions are associated incorrectly

  (Outline view, content assist, Fortran Search, refactorings, Open Declaration, …)

Differences (3): Source Form

Set free/fixed form associations in the project properties

- Right-click a project in the Fortran Projects view
- Click Properties
- Navigate the tree to **Fortran General ▶ Source Form**
- Select source form for each filename extension
- Click **OK**



Properties for org.eclipse.photran-projects.collections

type filter text

- ▶ Resource
- Builders
- ▶ C/C++ Build
- ▶ C/C++ General
- CVS
- ▶ Fortran Build
- ▼ Fortran General
  - Analysis/Refactoring
  - Paths and Symbols
  - Source Form
- Project References
- Run/Debug Settings
- Task Repository

Source Form

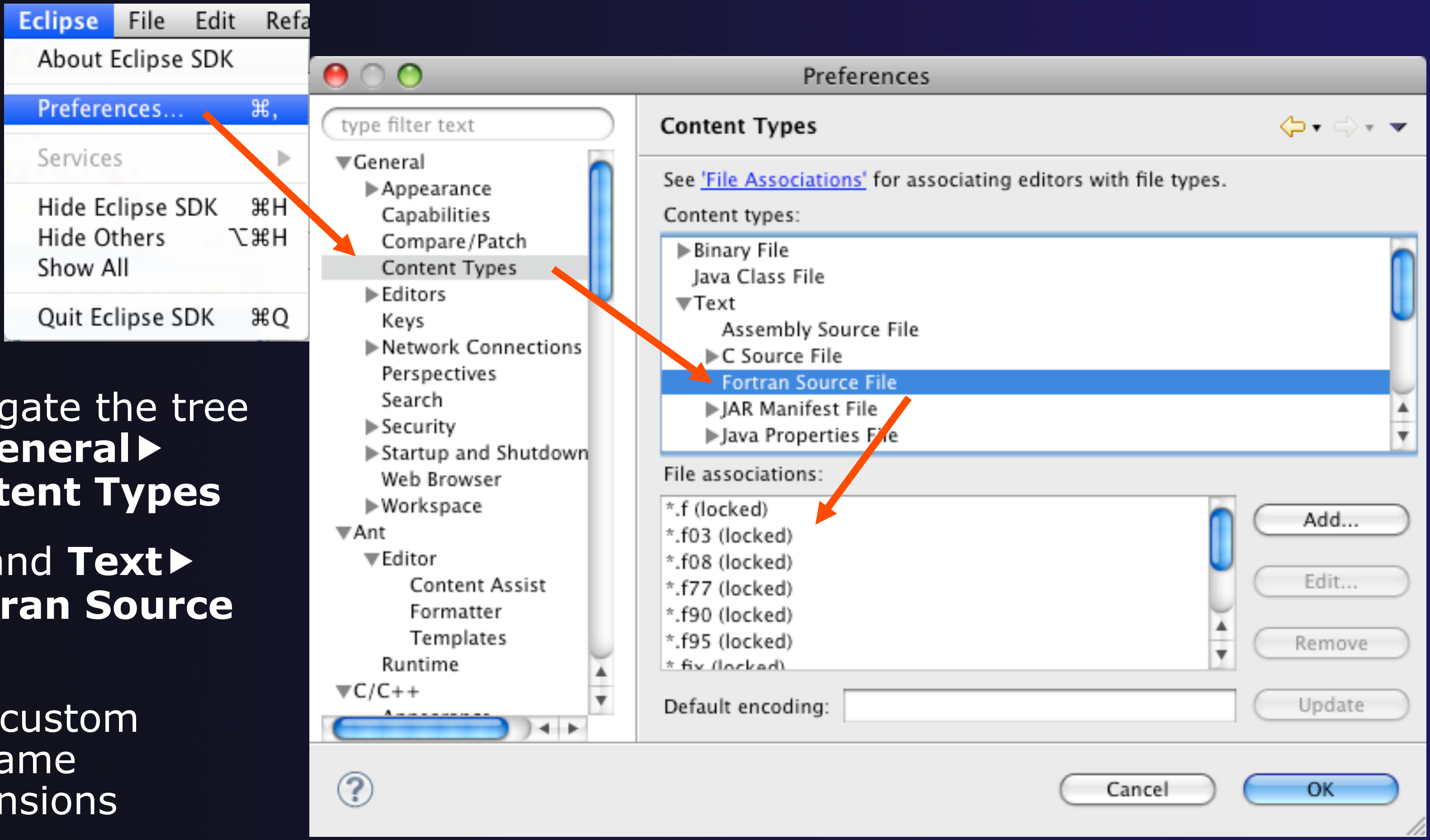
The list of Fortran filename extensions is determined by the workspace-wide content type settings.

Source form/filename associations:

File Name/Extension	Source Form
*.F	Fixed Form – INCLUDE lines ignored
*.F03	Free Form – C Preprocessed
*.F08	Free Form – C Preprocessed
*.F77	Fixed Form – INCLUDE lines ignored
*.F90	Free Form – C Preprocessed
*.F95	Free Form – C Preprocessed
*.FIX	Fixed Form – INCLUDE lines ignored
*.FOR	Fixed Form – INCLUDE lines ignored
*.FPP	Fixed Form – INCLUDE lines ignored
*.FTN	Fixed Form – INCLUDE lines ignored
*.f	Fixed Form – INCLUDE lines ignored
*.f03	Free Form
*.f08	Free Form
*.f77	Fixed Form – INCLUDE lines ignored

Differences (3): Source Form

Add new filename extensions in workspace preferences



Content Types

See 'File Associations' for associating editors with file types.

Content types:

- Binary File
- Java Class File
- Text
  - Assembly Source File
  - C Source File
  - Fortran Source File
  - JAR Manifest File
  - Java Properties File

File associations:

- *.f (locked)
- *.f03 (locked)
- *.f08 (locked)
- *.f77 (locked)
- *.f90 (locked)
- *.f95 (locked)
- *.fix (locked)

Default encoding:

- Navigate the tree to **General ▸ Content Types**
- Expand **Text ▸ Fortran Source File**
- Add custom filename extensions

Differences (4): Remote Support

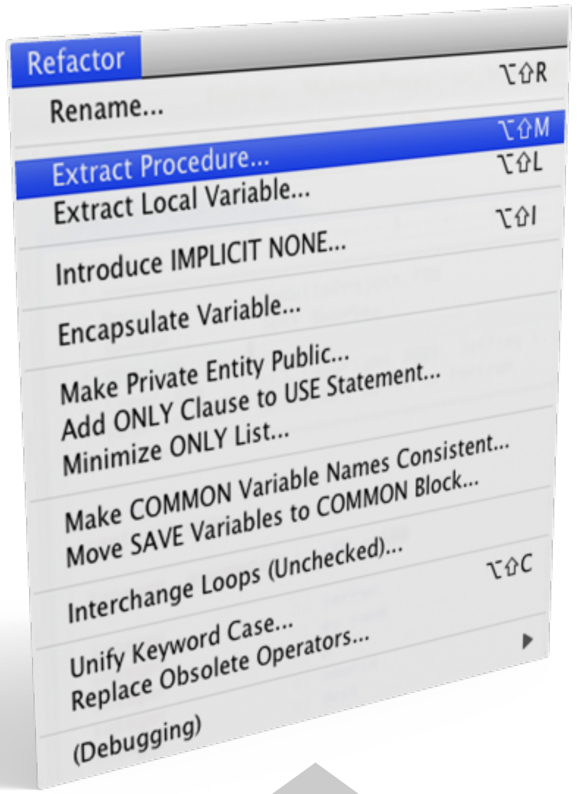
- Remote Fortran support is improving
  - Synchronized remote projects
    - Create Synchronized C/C++ Project, then Convert to Fortran Project
    - All features should work, except no support for remote INCLUDE/#include files
  - Fully remote projects
    - Create Remote C/C++ Project, then Convert to Fortran Project
    - Do not enable analysis/refactoring

For More Information

- **Photran online documentation**
  linked from http://www.eclipse.org/photran
  - **Installation Guide**
  - **User's Guide**
    General introduction, basic features
  - **Advanced Features Guide**
    Features requiring analysis/refactoring to be enabled

Refactoring

(making changes to source code that don't affect the behavior of the program)

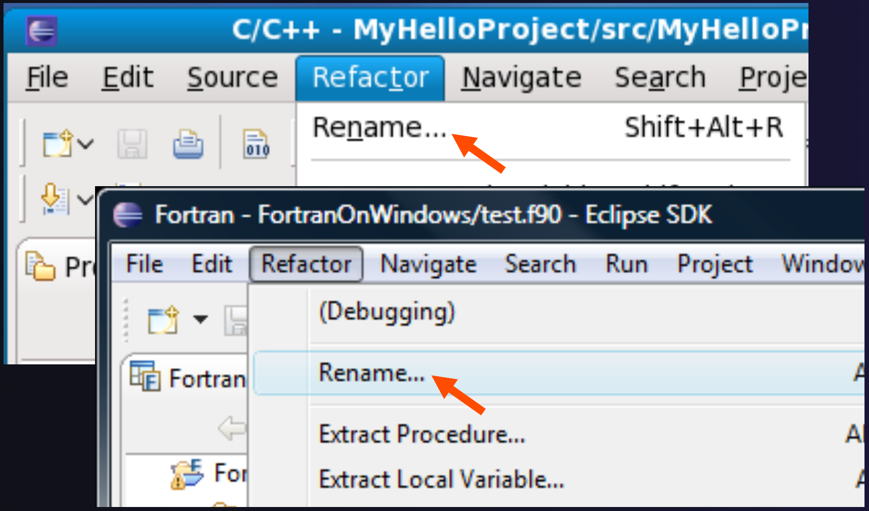

Refactor
Rename... ⌥⇧R
Extract Procedure... ⌥⇧M
Extract Local Variable... ⌥⇧L
Introduce IMPLICIT NONE... ⌥⇧I
Encapsulate Variable...
Make Private Entity Public...
Add ONLY Clause to USE Statement...
Minimize ONLY List...
Make COMMON Variable Names Consistent...
Move SAVE Variables to COMMON Block...
Interchange Loops (Unchecked)... ⌥⇧C
Unify Keyword Case...
Replace Obsolete Operators...
(Debugging)

- Refactoring is the research motivation for Photran @ Illinois
  - Illinois is a leader in refactoring research
  - "Refactoring" was coined in our group
    (Opdyke & Johnson, 1990)
  - We had the first dissertation…
    (Opdyke, 1992)
  - …and built the first refactoring tool…
    (Roberts, Brant, & Johnson, 1997)
  - …and first supported the C preprocessor
    (Garrido, 2005)
  - Photran's agenda: refactorings for HPC, language evolution, refactoring framework
- Photran 6.0: 16 refactorings
- Photran 7.0: 31 refactorings

Rename Refactoring

(also available in C/C++)

- Changes the name of a variable, function, etc., *including every use*
  (change is semantic, not textual, and can be workspace-wide)
- Only proceeds if the new name will be legal
  (aware of scoping rules, namespaces, etc.)

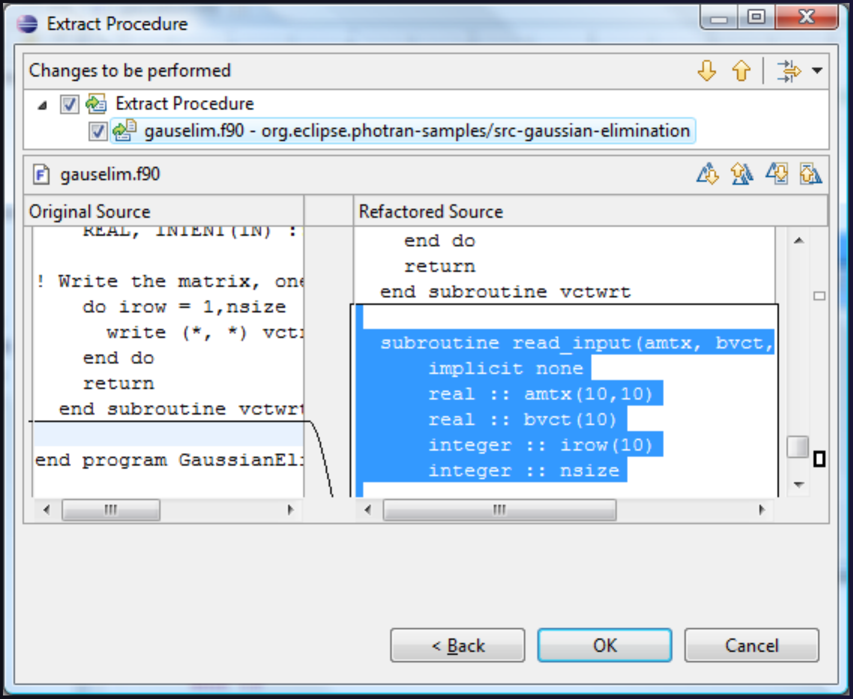

C/C++ - MyHelloProject/src/MyHelloPr
File Edit Source Refactor Navigate Search Proje
Rename... Shift+Alt+R
Fortran - FortranOnWindows/test.f90 - Eclipse SDK
File Edit Refactor Navigate Search Run Project Window
(Debugging)
Rename...
Extract Procedure...
Extract Local Variable...

- Select **Fortran Perspective**
- Open a source file
- Click in editor view on declaration of a variable
- Select menu item **Refactor▶Rename**
  - Or use context menu
- Enter new name

Extract Procedure Refactoring

(also available in C/C++ - "Extract Function")

- Moves statements into a new subroutine, replacing the statements with a call to that subroutine
- Local variables are passed as arguments



Extract Procedure

Changes to be performed

Extract Procedure

gauselim.f90 - org.eclipse.photran-samples/src-gaussian-elimination

gauselim.f90

Original Source

```
    REAL, INTENT(IN) :
  
! Write the matrix, one
    do irow = 1,nsize
      write (*, *) vct
    end do
    return
  end subroutine vctwrt

end program GaussianEl
```

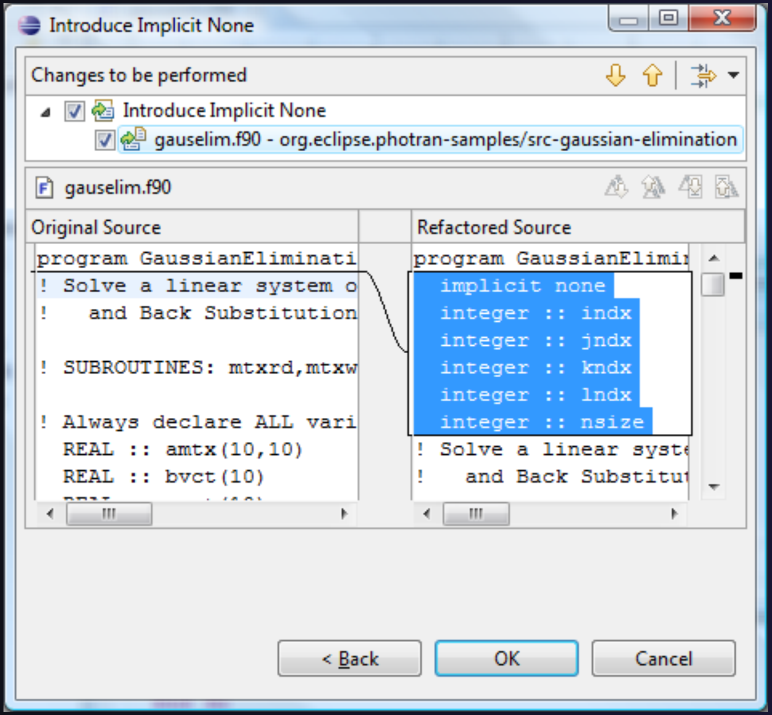
Refactored Source

```
    end do
    return
  end subroutine vctwrt

  subroutine read_input(amtx, bvct,
      implicit none
      real :: amtx(10,10)
      real :: bvct(10)
      integer :: irow(10)
      integer :: nsize
```

- Select a sequence of statements
- Select menu item **Refactor▸Extract Procedure…**
  - Or use context menu
- Enter new name

Introduce IMPLICIT NONE Refactoring

- Fortran does not require variable declarations
  (by default, names starting with I-N are integer variables; others are reals)
- This adds an IMPLICIT NONE statement and adds explicit variable declarations for all implicitly declared variables



Original Source:

```
program GaussianEliminati
! Solve a linear system o
!   and Back Substitution

! SUBROUTINES: mtxrd,mtxw

! Always declare ALL vari
  REAL :: amtx(10,10)
  REAL :: bvct(10)
```

Refactored Source:

```
program GaussianElimi
  implicit none
  integer :: indx
  integer :: jndx
  integer :: kndx
  integer :: lndx
  integer :: nsize
! Solve a linear syste
!   and Back Substitut
```

- Introduce in a single file by opening the file and selecting **Refactor ▶ Introduce IMPLICIT NONE…**
- Introduce in multiple files by selecting them in the Fortran Projects view, right-clicking on the selection, and choosing **Refactor ▶ Introduce IMPLICIT NONE…**

Module 7: Advanced Development

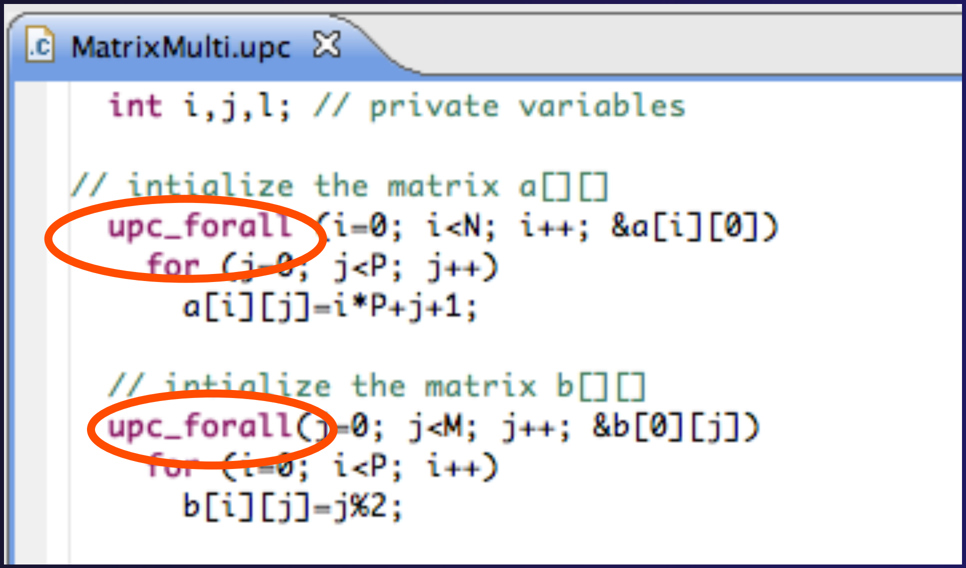
- Objective
  - Become familiar with other tools that help parallel application development
- Contents
  - Parallel Language Development Tools: MPI, OpenMP, UPC
    - Overview of UPC tools
  - Performance Tuning and other external tools:
    - PTP External Tools Framework (ETFw), TAU
    - Parallel Performance Wizard (PPW)
  - MPI Analysis: GEM (Graphical Explorer of MPI Programs)

Eclipse UPC Features

- CDT:
  - Parser/Editor support
  - Code templates
  - IBM XLc (incl. xlUPC) – remote
  - Berkeley UPC toolchain – local (see backup slides)
- PTP:
  - Artifact identification; Hover/dynamic help assistance
  - More Code templates
  - Remote UPC parsing and builds with xlupc
  - Parallel Performance Wizard integration with PTP

CDT - UPC Support

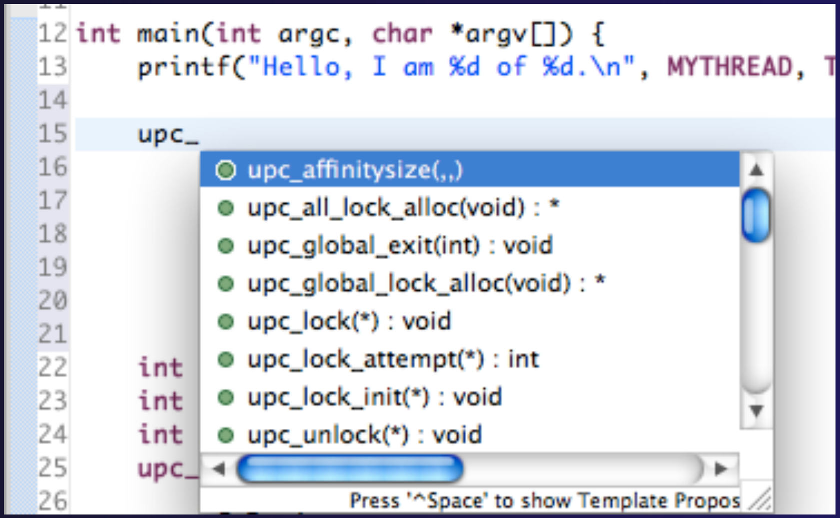
- Filetypes of "upc" will get UPC syntax high-lighting, content assist, etc.
- Use Preferences to change default for *.c if you like (we'll show you how)



```
  int i,j,l; // private variables

// intialize the matrix a[][]
  upc_forall (i=0; i<N; i++; &a[i][0])
    for (j=0; j<P; j++)
      a[i][j]=i*P+j+1;

  // intialize the matrix b[][]
  upc_forall(j=0; j<M; j++; &b[0][j])
    for (i=0; i<P; i++)
      b[i][j]=j%2;
```

UPC Content Assist, Hover Help

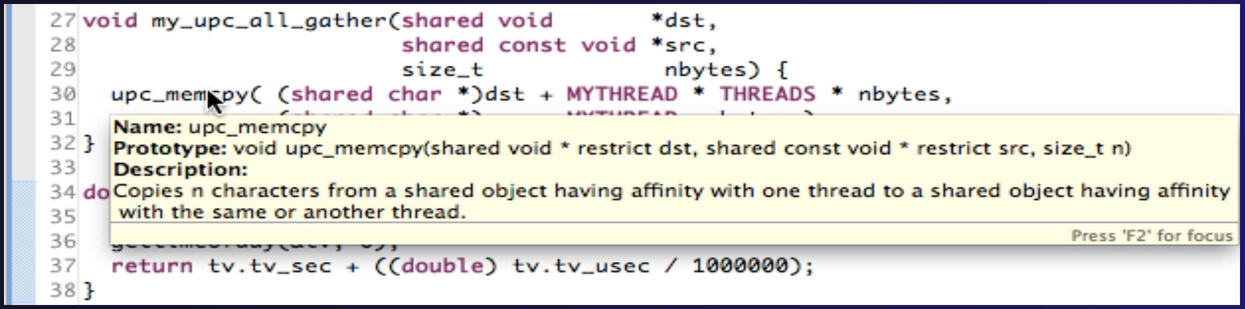
- In Editor, type upc and hit control-space (once)
- A list of possible completions is provided.
- Choose with mouse or cursor.



```
int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, T

    upc_

    int
    int
    int
    upc_
```

```
upc_affinitysize(,,)
upc_all_lock_alloc(void) : *
upc_global_exit(int) : void
upc_global_lock_alloc(void) : *
upc_lock(*) : void
upc_lock_attempt(*) : int
upc_lock_init(*) : void
upc_unlock(*) : void
Press '^Space' to show Template Propos
```

- Hover over API
- Hyperlink too



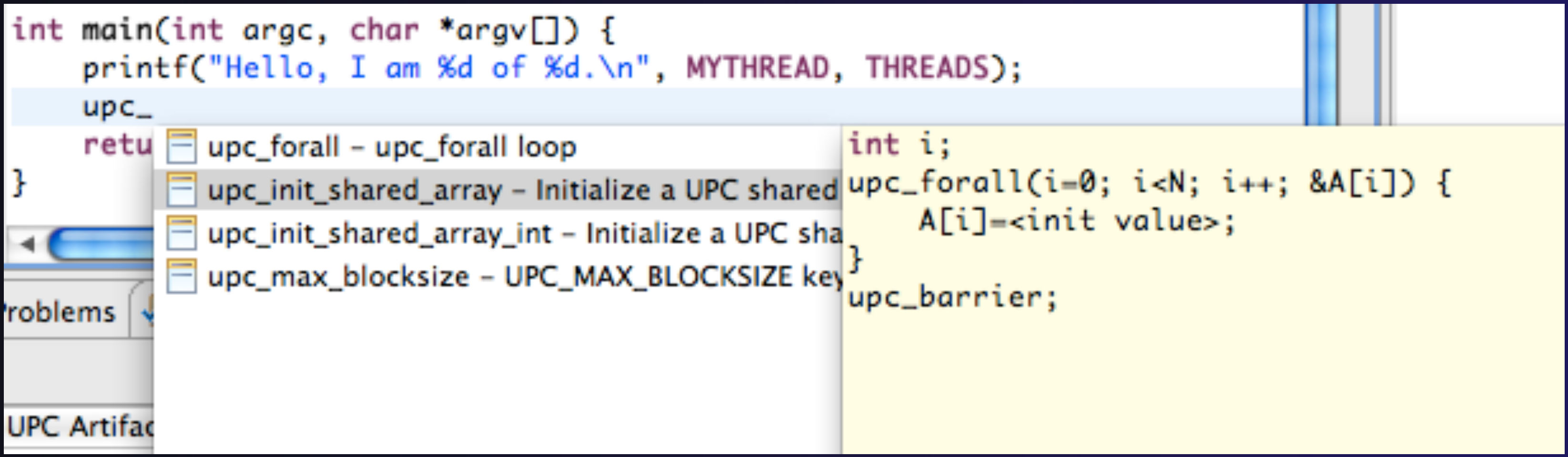
```
void my_upc_all_gather(shared void       *dst,
                       shared const void *src,
                       size_t             nbytes) {
  upc_memcpy( (shared char *)dst + MYTHREAD * THREADS * nbytes,
```

```
Name: upc_memcpy
Prototype: void upc_memcpy(shared void * restrict dst, shared const void * restrict src, size_t n)
Description:
Copies n characters from a shared object having affinity with one thread to a shared object having affinity
with the same or another thread.
Press 'F2' for focus
```

```
}

do
  return tv.tv_sec + ((double) tv.tv_usec / 1000000);
}
```

UPC templates - using

- In Editor, type upc and hit control-space (twice)



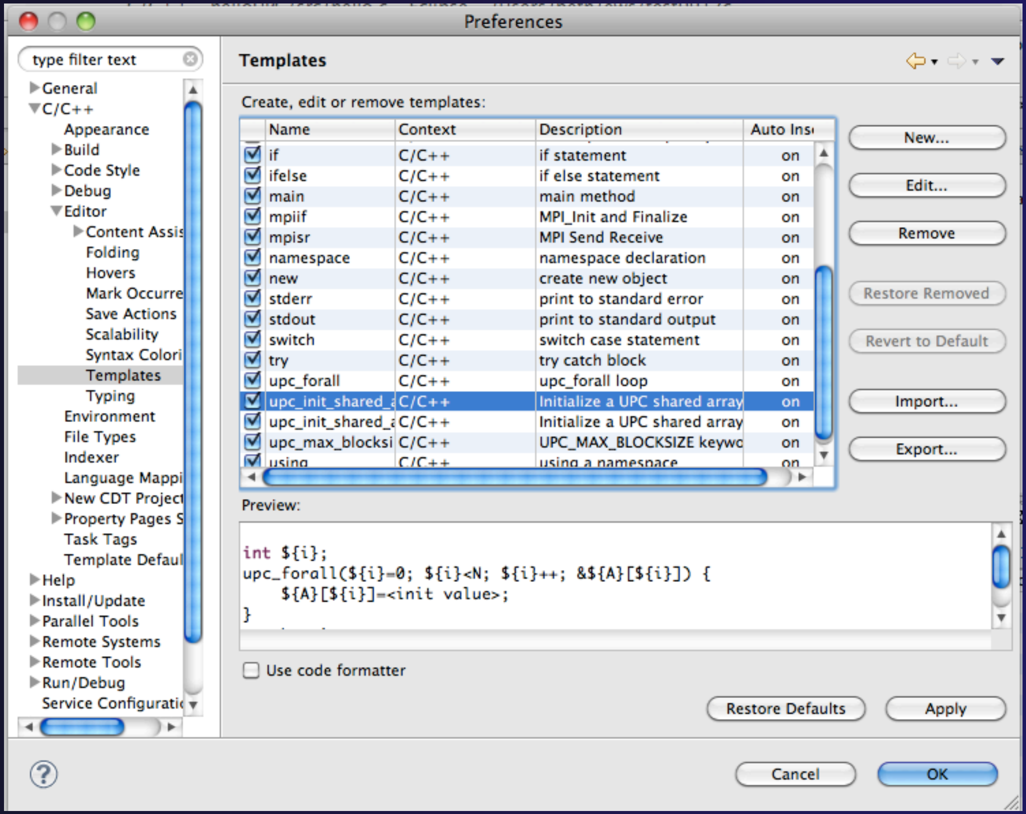
```
int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, THREADS);
    upc_
    retu
}
```

upc_forall – upc_forall loop
upc_init_shared_array – Initialize a UPC shared
upc_init_shared_array_int – Initialize a UPC sha
upc_max_blocksize – UPC_MAX_BLOCKSIZE key
roblems
UPC Artifac

```
int i;
upc_forall(i=0; i<N; i++; &A[i]) {
    A[i]=<init value>;
}
upc_barrier;
```

UPC templates – viewing/adding

- Eclipse preferences: add more! Or just see what's there
  - **C/C++ > Editor > Templates**



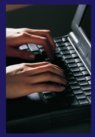
Create, edit or remove templates:

	Name	Context	Description	Auto Ins
✓	if	C/C++	if statement	on
✓	ifelse	C/C++	if else statement	on
✓	main	C/C++	main method	on
✓	mpiif	C/C++	MPI_Init and Finalize	on
✓	mpisr	C/C++	MPI Send Receive	on
✓	namespace	C/C++	namespace declaration	on
✓	new	C/C++	create new object	on
✓	stderr	C/C++	print to standard error	on
✓	stdout	C/C++	print to standard output	on
✓	switch	C/C++	switch case statement	on
✓	try	C/C++	try catch block	on
✓	upc_forall	C/C++	upc_forall loop	on
✓	upc_init_shared_a	C/C++	Initialize a UPC shared array	on
✓	upc_init_shared_a	C/C++	Initialize a UPC shared array	on
✓	upc_max_blocksi	C/C++	UPC_MAX_BLOCKSIZE keywo	on
✓	using	C/C++	using a namespace	on

Preview:

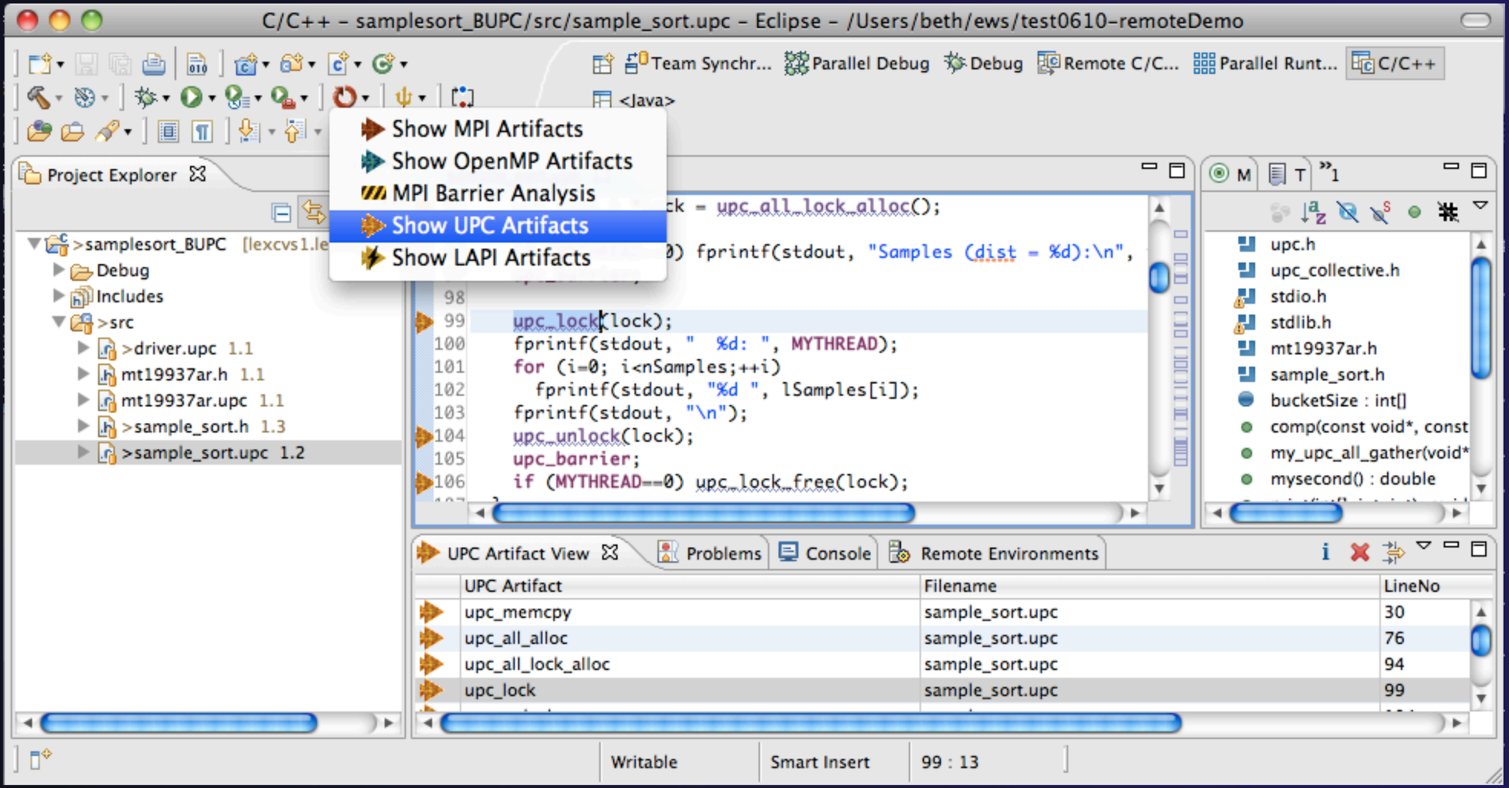
```
int ${i};
upc_forall(${i}=0; ${i}<N; ${i}++; &${A}[${i}]) {
    ${A}[${i}]=<init value>;
}
```

☐ Use code formatter



Show UPC Artifacts

- Add some UPC api's to your sample project
- Show UPC Artifacts – remote projects need CDT > 8.0



C/C++ – samplesort_BUPC/src/sample_sort.upc – Eclipse – /Users/beth/ews/test0610-remoteDemo

```
ck = upc_all_lock_alloc();

0) fprintf(stdout, "Samples (dist = %d):\n",


    upc_lock(lock);
    fprintf(stdout, "  %d: ", MYTHREAD);
    for (i=0; i<nSamples;++i)
      fprintf(stdout, "%d ", lSamples[i]);
    fprintf(stdout, "\n");
    upc_unlock(lock);
    upc_barrier;
    if (MYTHREAD==0) upc_lock_free(lock);
```

UPC Artifact View

	UPC Artifact	Filename	LineNo
	upc_memcpy	sample_sort.upc	30
	upc_all_alloc	sample_sort.upc	76
	upc_all_lock_alloc	sample_sort.upc	94
	upc_lock	sample_sort.upc	99

Other UPC features

- UPC parser is remote-enabled
  - Remote UPC projects can be developed efficiently
- Remote xlUPC toolchain enables remote build of IBM xlUPC project
  - Managed Build (user-friendly) way to specify and manage complex build options without makefiles

More Advanced Features: Demos

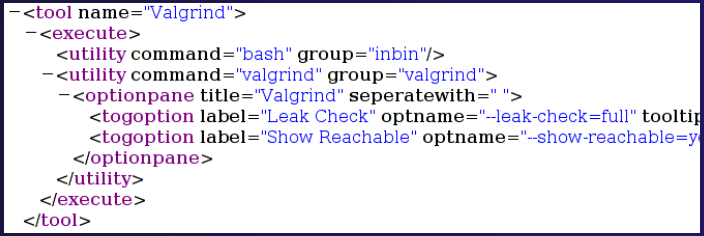
- ETFw – External Tools Framework and TAU, Tuning and Analysis Utilities
  - Suzanne Millstein, U. Oregon
- PPW – Parallel Performance Wizard
  - No demo today)
- GEM – Graphical Explorer of MPI Programs
  Dynamic Formal Verification for MPI
  - Alan Humphrey, U. Utah

PTP/External Tools Framework

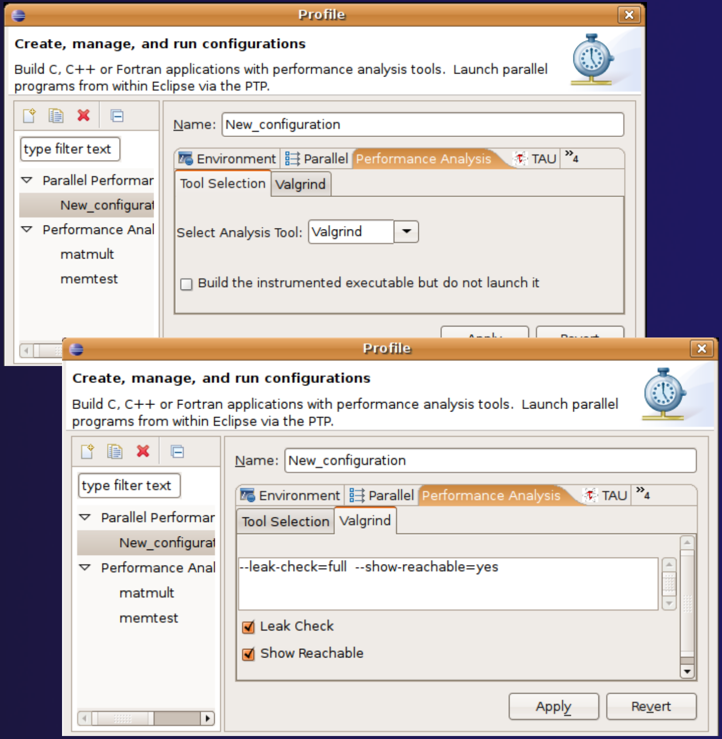
formerly "Performance Tools Framework"

Goal:

- **Reduce the "eclipse plumbing" necessary to integrate tools**
- Provide integration for instrumentation, measurement, and analysis for a variety of performance tools
  - Dynamic Tool Definitions: Workflows & UI
  - Tools and tool workflows are specified in an XML file
  - Tools are selected and configured in the launch configuration window
  - Output is generated, managed and analyzed as specified in the workflow



```
−<tool name="Valgrind">
  −<execute>
     <utility command="bash" group="inbin"/>
   −<utility command="valgrind" group="valgrind">
     −<optionpane title="Valgrind" seperatewith=" ">
         <togoption label="Leak Check" optname="--leak-check=full" tooltip
         <togoption label="Show Reachable" optname="--show-reachable=y
       </optionpane>
     </utility>
   </execute>
 </tool>
```



Profile

Create, manage, and run configurations

Build C, C++ or Fortran applications with performance analysis tools. Launch parallel programs from within Eclipse via the PTP.

Name: New_configuration

Select Analysis Tool: Valgrind

☐ Build the instrumented executable but do not launch it

Profile

Create, manage, and run configurations

Build C, C++ or Fortran applications with performance analysis tools. Launch parallel programs from within Eclipse via the PTP.

Name: New_configuration

--leak-check=full --show-reachable=yes

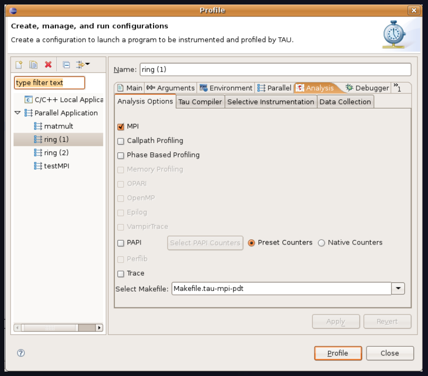
☑ Leak Check

☑ Show Reachable

PTP TAU plug-ins

http://www.cs.uoregon.edu/research/tau

- TAU (Tuning and Analysis Utilities)
- First implementation of External Tools Framework (ETFw)
- Eclipse plug-ins wrap TAU functions, make them available from Eclipse
- Compatible with Photran and CDT projects and with PTP parallel application launching
- Other plug-ins launch Paraprof from Eclipse too


Profile

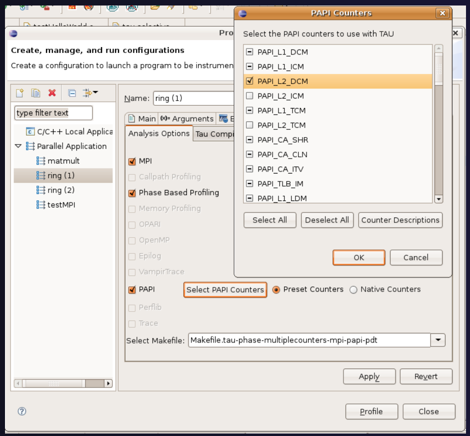
Create, manage, and run configurations

Create a configuration to launch a program to be instrumented and profiled by TAU.

Name: ring (1)

Analysis Options:
- [✓] MPI
- [ ] Callpath Profiling
- [ ] Phase Based Profiling
- [ ] Memory Profiling
- [ ] OPARI
- [ ] OpenMP
- [ ] Epilog
- [ ] VampirTrace
- [ ] PAPI — (●) Preset Counters ( ) Native Counters
- [ ] Perflib
- [ ] Trace

Select Makefile: Makefile.tau-mpi-pdt


PAPI Counters

Select the PAPI counters to use with TAU

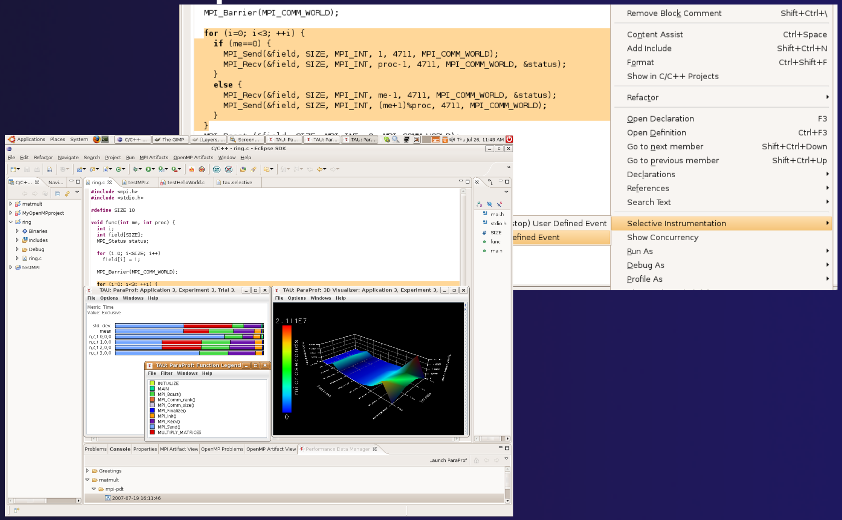
- PAPI_L1_DCM
- PAPI_L1_ICM
- [✓] PAPI_L2_DCM
- [ ] PAPI_L2_ICM
- PAPI_L1_TCM
- [ ] PAPI_L2_TCM
- PAPI_CA_SHR
- PAPI_CA_CLN
- PAPI_CA_ITV
- PAPI_TLB_IM
- PAPI_L1_LDM

Create, manage, and run configurations

Name: ring (1)

- [✓] MPI
- [ ] Callpath Profiling
- [✓] Phase Based Profiling
- [ ] Memory Profiling
- [ ] OPARI
- [ ] OpenMP
- [ ] Epilog
- [ ] VampirTrace
- [✓] PAPI — (●) Preset Counters ( ) Native Counters
- [ ] Perflib
- [ ] Trace

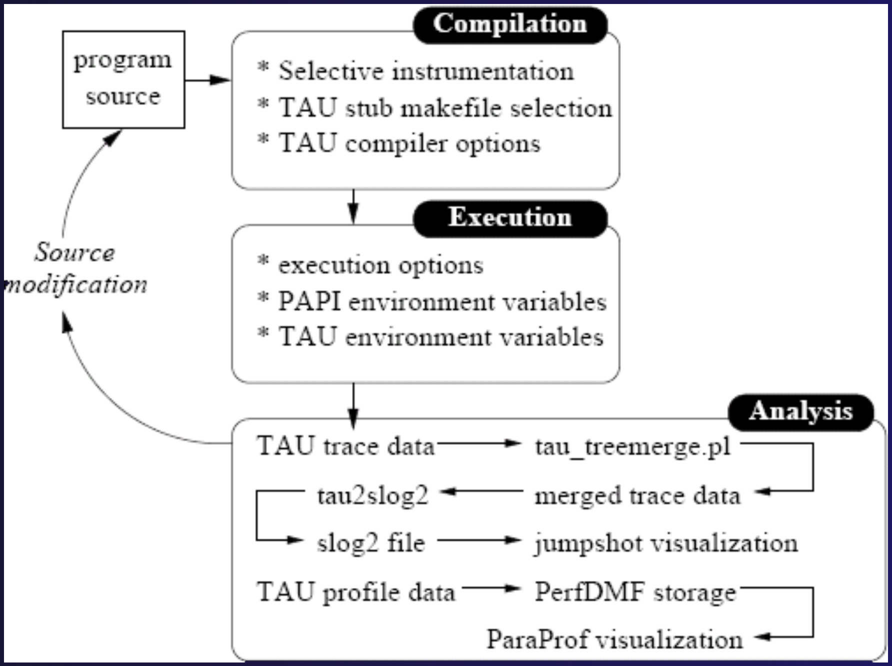
Select Makefile: Makefile.tau-phase-multiplecounters-mpi-papi-pdt


```
MPI_Barrier(MPI_COMM_WORLD);

for (i=0; i<3; ++i) {
  if (me==0) {
    MPI_Send(&field, SIZE, MPI_INT, 1, 4711, MPI_COMM_WORLD);
    MPI_Recv(&field, SIZE, MPI_INT, proc-1, 4711, MPI_COMM_WORLD, &status);
  }
  else {
    MPI_Recv(&field, SIZE, MPI_INT, me-1, 4711, MPI_COMM_WORLD, &status);
    MPI_Send(&field, SIZE, MPI_INT, (me+1)%proc, 4711, MPI_COMM_WORLD);
  }
}
```

Remove Block Comment Shift+Ctrl+\
Content Assist Ctrl+Space
Add Include Shift+Ctrl+N
Format Ctrl+Shift+F
Show in C/C++ Projects
Refactor
Open Declaration F3
Open Definition Ctrl+F3
Go to next member Shift+Ctrl+Down
Go to previous member Shift+Ctrl+Up
Declarations
References
Search Text
Selective Instrumentation
Show Concurrency
Run As
Debug As
Profile As
top) User Defined Event
efined Event
C/C++ - ring.c - Eclipse SDK
TAU: ParaProf: 3D Visualizer: Application 3, Experiment 3,
TAU: ParaProf: Function Legend
2.111E7

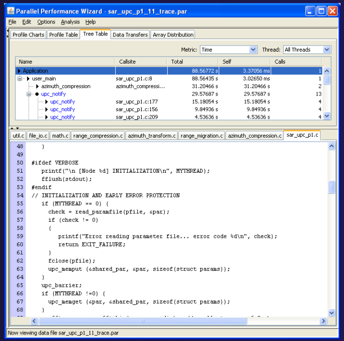
TAU Integration with PTP

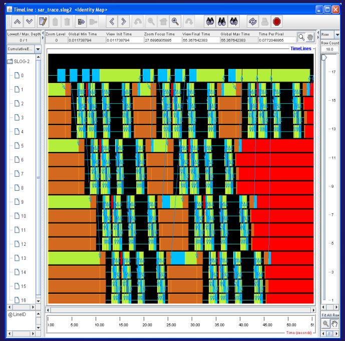
- TAU: Tuning and Analysis Utilities
  - Performance data collection and analysis for HPC codes
  - Numerous features
  - Command line interface
- The TAU Workflow:
  - Instrumentation
  - Execution
  - Analysis

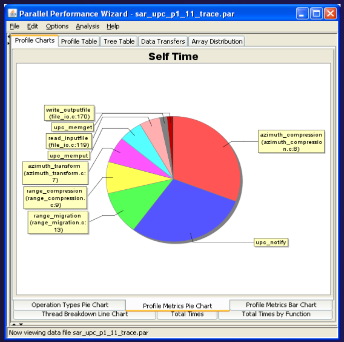


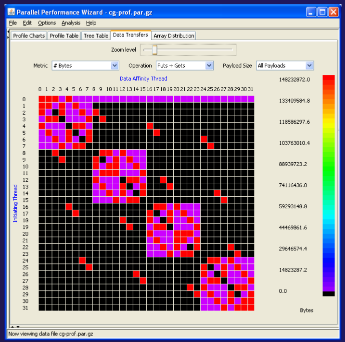
Parallel Performance Wizard (PPW)

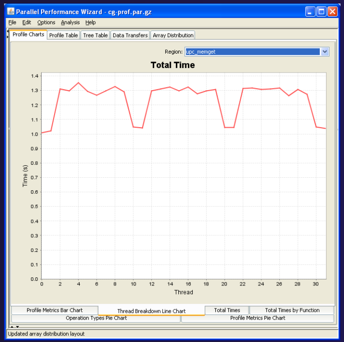
- Full-featured performance tool for PGAS programming models
  - Currently supports UPC, SHMEM, and MPI
  - Extensible to support other models
  - PGAS support by way of Global Address Space Performance (GASP) interface (http://gasp.hcs.ufl.edu)
- PPW features:
  - Easy-to-use scripts for backend data collection
  - User-friendly GUI with familiar visualizations
  - Advanced automatic analysis support
- More information and free download: http://ppw.hcs.ufl.edu

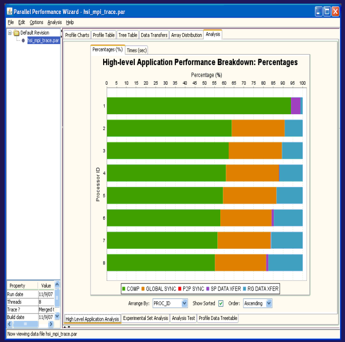






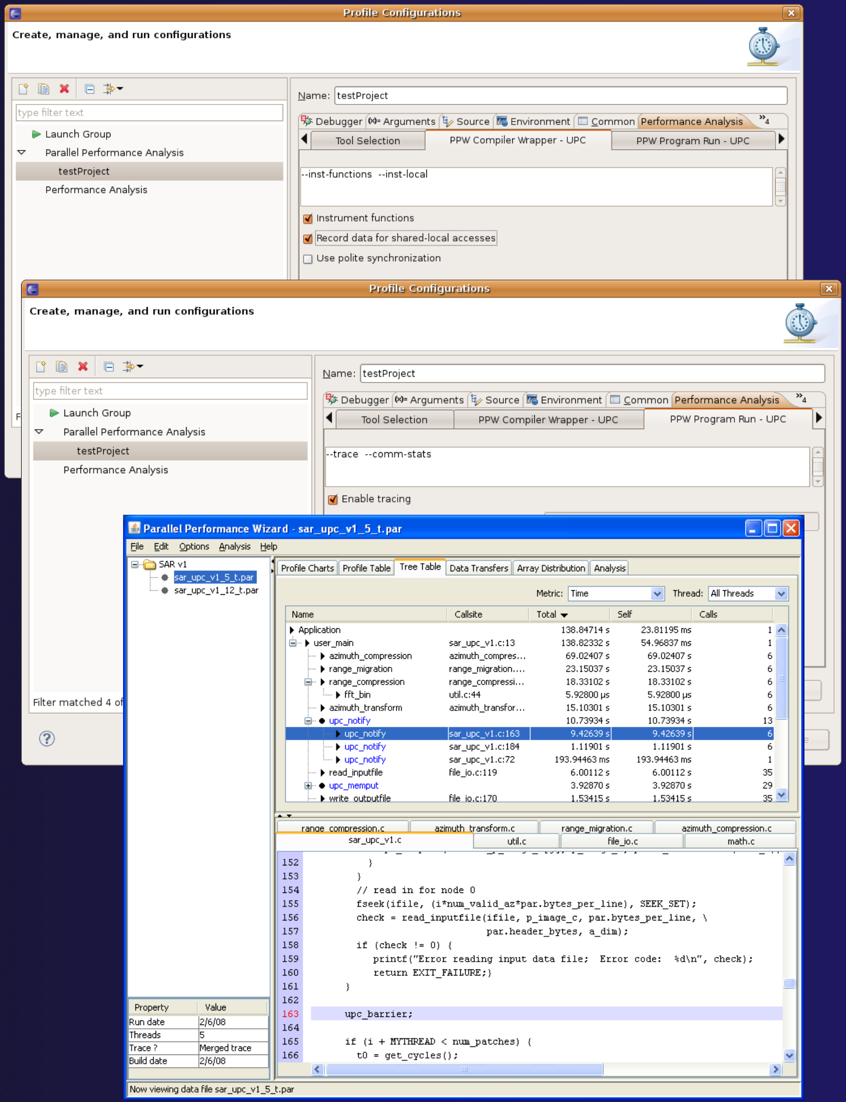





Parallel Performance Wizard - hsl_mpi_trace.par
High-level Application Performance Breakdown: Percentages

PPW Integration via ETFw

- We implement the ETFw to make PPW's capabilities available within Eclipse
  - Compile with instrumentation, parallel launch with PPW
  - Generates performance data file in workspace, PPW GUI launched
- PPW is often used for UPC application analysis
  - ETFw extended to support UPC
  - Many UPC features in PTP
- For more information:
  - http://ppw.hcs.ufl.edu
  - ppw@hcs.ufl.edu



Profile Configurations — Create, manage, and run configurations

Name: testProject

Tool Selection | PPW Compiler Wrapper - UPC | PPW Program Run - UPC

```
--inst-functions  --inst-local
```

- [✓] Instrument functions
- [✓] Record data for shared-local accesses
- [ ] Use polite synchronization

Profile Configurations — Create, manage, and run configurations

Name: testProject

```
--trace  --comm-stats
```

- [✓] Enable tracing

Parallel Performance Wizard - sar_upc_v1_5_t.par

Metric: Time Thread: All Threads

Name	Callsite	Total	Self	Calls
Application		138.84714 s	23.81195 ms	1
user_main	sar_upc_v1.c:13	138.82332 s	54.96837 ms	1
azimuth_compression	azimuth_compres…	69.02407 s	69.02407 s	6
range_migration	range_migration.…	23.15037 s	23.15037 s	6
range_compression	range_compressi…	18.33102 s	18.33102 s	6
fft_bin	util.c:44	5.92800 μs	5.92800 μs	6
azimuth_transform	azimuth_transfor…	15.10301 s	15.10301 s	6
upc_notify		10.73934 s	10.73934 s	13
upc_notify	sar_upc_v1.c:163	9.42639 s	9.42639 s	6
upc_notify	sar_upc_v1.c:184	1.11901 s	1.11901 s	6
upc_notify	sar_upc_v1.c:72	193.94463 ms	193.94463 ms	1
read_inputfile	file_io.c:119	6.00112 s	6.00112 s	35
upc_memput		3.92870 s	3.92870 s	29
write_outputfile	file_io.c:170	1.53415 s	1.53415 s	35

```
          }
        }
        // read in for node 0
        fseek(ifile, (i*num_valid_az*par.bytes_per_line), SEEK_SET);
        check = read_inputfile(ifile, p_image_c, par.bytes_per_line, \
                               par.header_bytes, a_dim);
        if (check != 0) {
          printf("Error reading input data file;  Error code:  %d\n", check);
          return EXIT_FAILURE;}
      }

      upc_barrier;

      if (i + MYTHREAD < num_patches) {
        t0 = get_cycles();
```

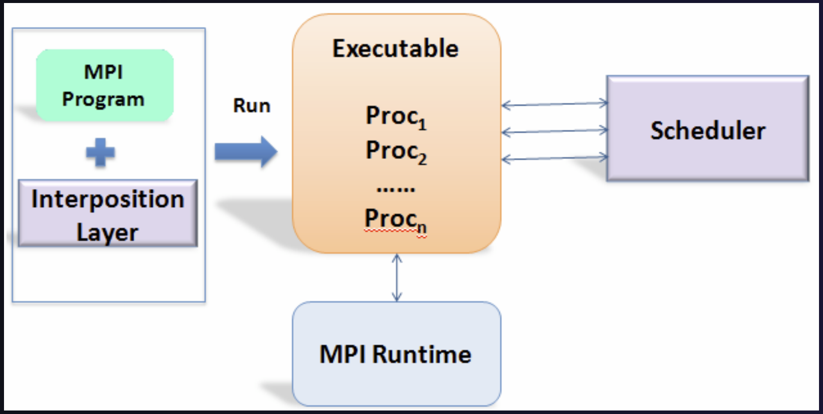
Property	Value
Run date	2/6/08
Threads	5
Trace ?	Merged trace
Build date	2/6/08

Now viewing data file sar_upc_v1_5_t.par

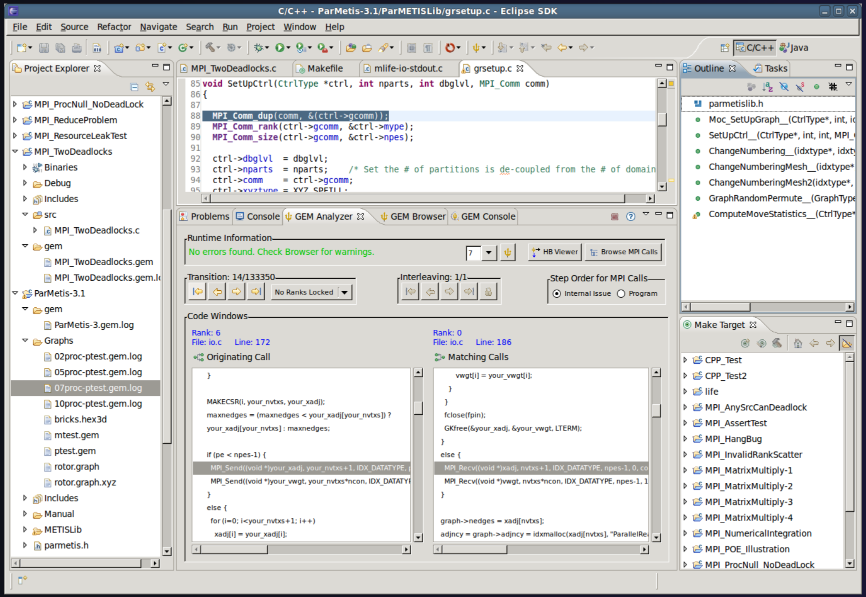
GEM
Graphical Explorer of MPI Programs

- Contributed to PTP by University of Utah in 2009
  - Available with PTP since v3.0
- Dynamic verification for MPI C/C++ that detects:
  - Deadlocks
  - MPI object leaks
  - Functionally irrelevant barriers
  - Local assertion violations
- Offers rigorous coverage guarantees
  - Complete nondeterministic coverage for MPI
  - Communication / synchronization behaviors
  - Determines relevant interleavings, replaying as necessary

GEM - Overview



- Front-end for In-situ Partial Order (ISP), Developed at U. Utah
- Introduces "push-button" verification into the MPI development cycle for PTP
- Automatically instruments and runs user code, displaying post verification results
- Variety of views & tools to facilitate debugging and code understanding



```
void SetUpCtrl(CtrlType *ctrl, int nparts, int dbglvl, MPI_Comm comm)
{

  MPI_Comm_dup(comm, &(ctrl->gcomm));
  MPI_Comm_rank(ctrl->gcomm, &ctrl->mype);
  MPI_Comm_size(ctrl->gcomm, &ctrl->npes);

  ctrl->dbglvl  = dbglvl;
  ctrl->nparts  = nparts;    /* Set the # of partitions is de-coupled from the # of domain
  ctrl->comm    = ctrl->gcomm;
```

Runtime Information
No errors found. Check Browser for warnings.
7

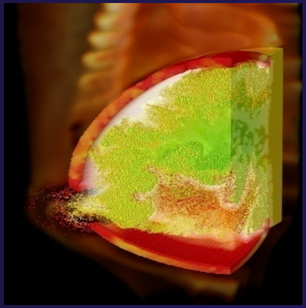
Transition: 14/133350
Interleaving: 1/1
[✓ Internal Issue] Program

Rank: 6
File: io.c Line: 172

```
    }

    MAKECSR(i, your_nvtxs, your_xadj);
    maxnedges = (maxnedges < your_xadj[your_nvtxs]) ?
    your_xadj[your_nvtxs] : maxnedges;

    if (pe < npes-1) {
     MPI_Send((void *)your_xadj, your_nvtxs+1, IDX_DATATYPE, 
     MPI_Send((void *)your_vwgt, your_nvtxs*ncon, IDX_DATATYP
    }
    else {
     for (i=0; i<your_nvtxs+1; i++)
      xadj[i] = your_xadj[i];
```

Rank: 0
File: io.c Line: 186

```
      vwgt[i] = your_vwgt[i];
     }
    }
    fclose(fpin);
    GKfree(&your_xadj, &your_vwgt, LTERM);
   }
   else {
    MPI_Recv((void *)xadj, nvtxs+1, IDX_DATATYPE, npes-1, 0, co
    MPI_Recv((void *)vwgt, nvtxs*ncon, IDX_DATATYPE, npes-1, 1
   }

   graph->nedges = xadj[nvtxs];
   adjncy = graph->adjncy = idxmalloc(xadj[nvtxs], "ParallelRe
```



(Image courtesy of Steve Parker, U of Utah)

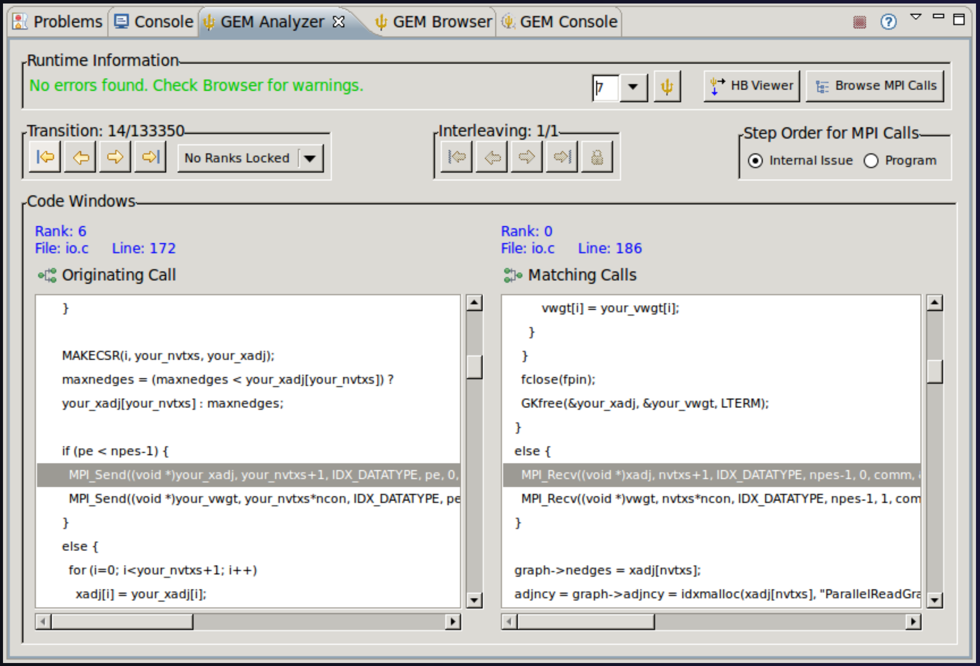
GEM – Views & Tools

Analyzer View

Highlights bugs, and facilitates post-verification review / debugging

Browser View

Groups & helps quickly localizes MPI problems. Maps errors to source code line in editor



Problems | Console | GEM Analyzer | GEM Browser | GEM Console

Runtime Information

No errors found. Check Browser for warnings.

7 | HB Viewer | Browse MPI Calls

Transition: 14/133350 | No Ranks Locked

Interleaving: 1/1

Step Order for MPI Calls: [✓ Internal Issue] Program

Code Windows

Rank: 6
File: io.c Line: 172

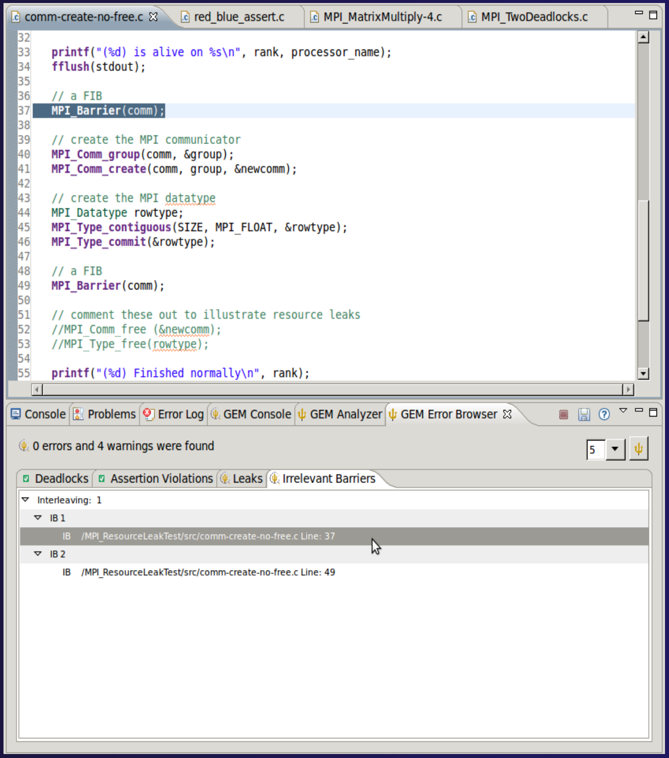
Originating Call

```
  }

  MAKECSR(i, your_nvtxs, your_xadj);
  maxnedges = (maxnedges < your_xadj[your_nvtxs]) ?
  your_xadj[your_nvtxs] : maxnedges;

  if (pe < npes-1) {
   MPI_Send((void *)your_xadj, your_nvtxs+1, IDX_DATATYPE, pe, 0,
   MPI_Send((void *)your_vwgt, your_nvtxs*ncon, IDX_DATATYPE, pe
  }
  else {
   for (i=0; i<your_nvtxs+1; i++)
    xadj[i] = your_xadj[i];
```

Rank: 0
File: io.c Line: 186

Matching Calls

```
      vwgt[i] = your_vwgt[i];
    }
   }
   fclose(fpin);
   GKfree(&your_xadj, &your_vwgt, LTERM);
  }
  else {
   MPI_Recv((void *)xadj, nvtxs+1, IDX_DATATYPE, npes-1, 0, comm,
   MPI_Recv((void *)vwgt, nvtxs*ncon, IDX_DATATYPE, npes-1, 1, com
  }

  graph->nedges = xadj[nvtxs];
  adjncy = graph->adjncy = idxmalloc(xadj[nvtxs], "ParallelReadGra
```



comm-create-no-free.c | red_blue_assert.c | MPI_MatrixMultiply-4.c | MPI_TwoDeadlocks.c

```
32
33   printf("(%d) is alive on %s\n", rank, processor_name);
34   fflush(stdout);
35
36   // a FIB
37   MPI_Barrier(comm);
38
39   // create the MPI communicator
40   MPI_Comm_group(comm, &group);
41   MPI_Comm_create(comm, group, &newcomm);
42
43   // create the MPI datatype
44   MPI_Datatype rowtype;
45   MPI_Type_contiguous(SIZE, MPI_FLOAT, &rowtype);
46   MPI_Type_commit(&rowtype);
47
48   // a FIB
49   MPI_Barrier(comm);
50
51   // comment these out to illustrate resource leaks
52   //MPI_Comm_free (&newcomm);
53   //MPI_Type_free(rowtype);
54
55   printf("(%d) Finished normally\n", rank);
```

Console | Problems | Error Log | GEM Console | GEM Analyzer | GEM Error Browser

0 errors and 4 warnings were found

5

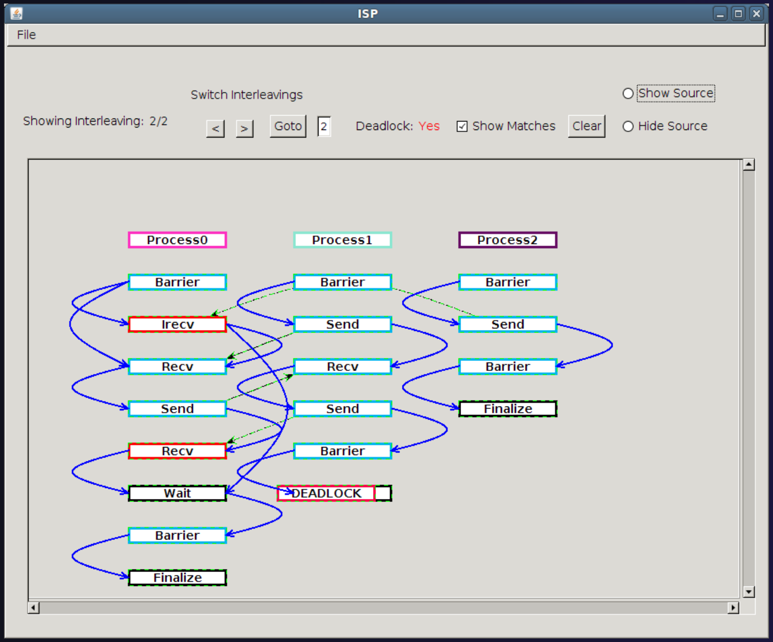
Deadlocks | Assertion Violations | Leaks | Irrelevant Barriers

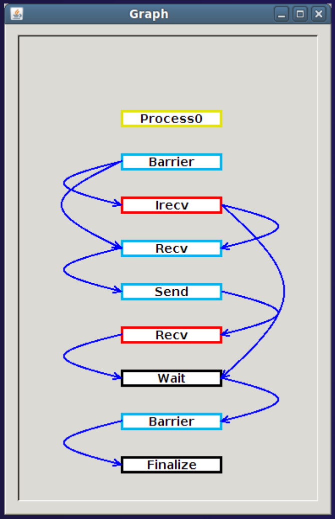
```
Interleaving: 1
  IB 1
    IB  /MPI_ResourceLeakTest/src/comm-create-no-free.c Line: 37
  IB 2
    IB  /MPI_ResourceLeakTest/src/comm-create-no-free.c Line: 49
```

GEM – Views & Tools (cont.)

Happens-Before Viewer

Shows required orderings and communication matches (currently an external tool)



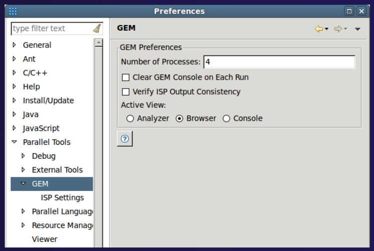


Using GEM – ISP Installation

- **ISP itself must be installed prior to using GEM**
  - Download ISP at http://www.cs.utah.edu/fv/ISP
- Make sure libtool, automake and autoconf are installed.
- Just untar isp-0.2.0.tar.gz into a tmp directory:
  - Configure and install
    - ./configure
    - make
    - make install
      - This installs binaries and necessary scripts

Using GEM

- Create local or remote MPI C/C++ project
  - Make sure your project builds correctly
  - Managed build and Makefile projects supported
- Set preferences via GEM Preference Pages
- From the trident icon or context menus user can:



GEM

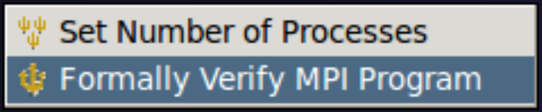
GEM Preferences

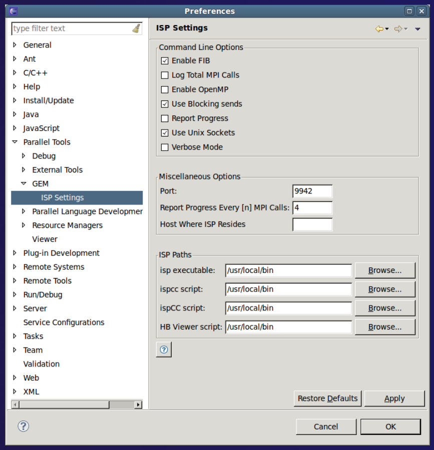
Number of Processes: 4

☐ Clear GEM Console on Each Run

☐ Verify ISP Output Consistency

Active View: Analyzer [✓ Browser] Console





ISP Settings

Command Line Options

☑ Enable FIB

☐ Log Total MPI Calls

☐ Enable OpenMP

☑ Use Blocking sends

☐ Report Progress

☑ Use Unix Sockets

☐ Verbose Mode

Miscellaneous Options

Port: 9942

Report Progress Every [n] MPI Calls: 4

Host Where ISP Resides

ISP Paths

isp executable: /usr/local/bin Browse...

ispcc script: /usr/local/bin Browse...

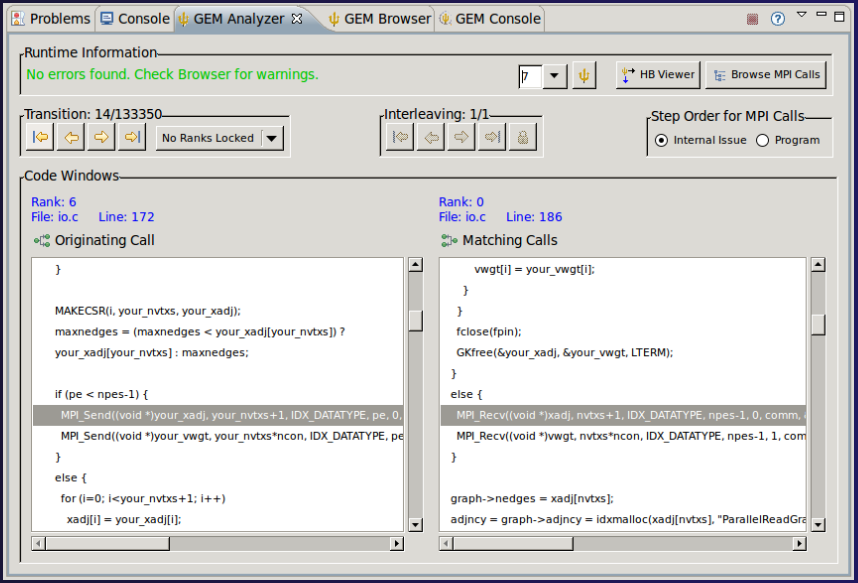
ispCC script: /usr/local/bin Browse...

HB Viewer script: /usr/local/bin Browse...

- Formally Verifying MPI Program
  - Launches verification engine ISP
  - Generates log file for post-verification analysis
  - Opens relevant GEM views

GEM Analyzer View

- Reports program errors, and runtime statistics
- Debug-style source code stepping of interleavings
  - Point-to-point / Collective Operation matches
  - Internal Issue Order / Program Order views
  - Rank Lock feature – focus on a particular process
- Also controls:
  - Call Browser
  - Happens Before Viewer launch
  - Re-launching of GEM



Runtime Information

No errors found. Check Browser for warnings.

Transition: 14/133350

Interleaving: 1/1

Step Order for MPI Calls: [✓ Internal Issue] Program

Code Windows

Rank: 6
File: io.c Line: 172

Originating Call

```
  }

  MAKECSR(i, your_nvtxs, your_xadj);
  maxnedges = (maxnedges < your_xadj[your_nvtxs]) ?
  your_xadj[your_nvtxs] : maxnedges;

  if (pe < npes-1) {
   MPI_Send((void *)your_xadj, your_nvtxs+1, IDX_DATATYPE, pe, 0,
   MPI_Send((void *)your_vwgt, your_nvtxs*ncon, IDX_DATATYPE, pe
  }
  else {
   for (i=0; i<your_nvtxs+1; i++)
    xadj[i] = your_xadj[i];
```

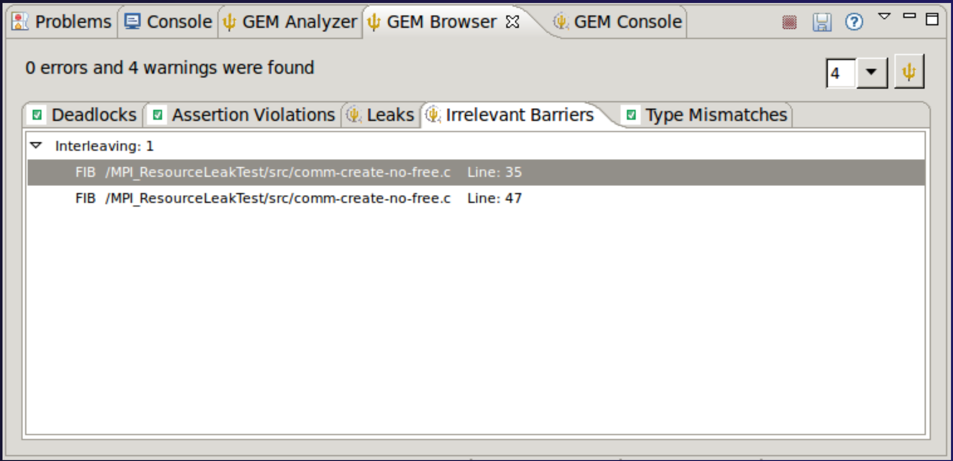
Rank: 0
File: io.c Line: 186

Matching Calls

```
      vwgt[i] = your_vwgt[i];
    }
   }
   fclose(fpin);
   GKfree(&your_xadj, &your_vwgt, LTERM);
  }
  else {
   MPI_Recv((void *)xadj, nvtxs+1, IDX_DATATYPE, npes-1, 0, comm,
   MPI_Recv((void *)vwgt, nvtxs*ncon, IDX_DATATYPE, npes-1, 1, com
  }

  graph->nedges = xadj[nvtxs];
  adjncy = graph->adjncy = idxmalloc(xadj[nvtxs], "ParallelReadGra
```

GEM Browser View

- Tabbed browsing for each type of MPI error/warning
- Each error/warning mapped to offending line of source code in Eclipse editor
- One click to visit the Eclipse editor, to examine:
  - Calls involved in deadlock
  - Irrelevant barriers
  - MPI Object Leaks sites
  - MPI type mismatches
  - Local Assertion Violations

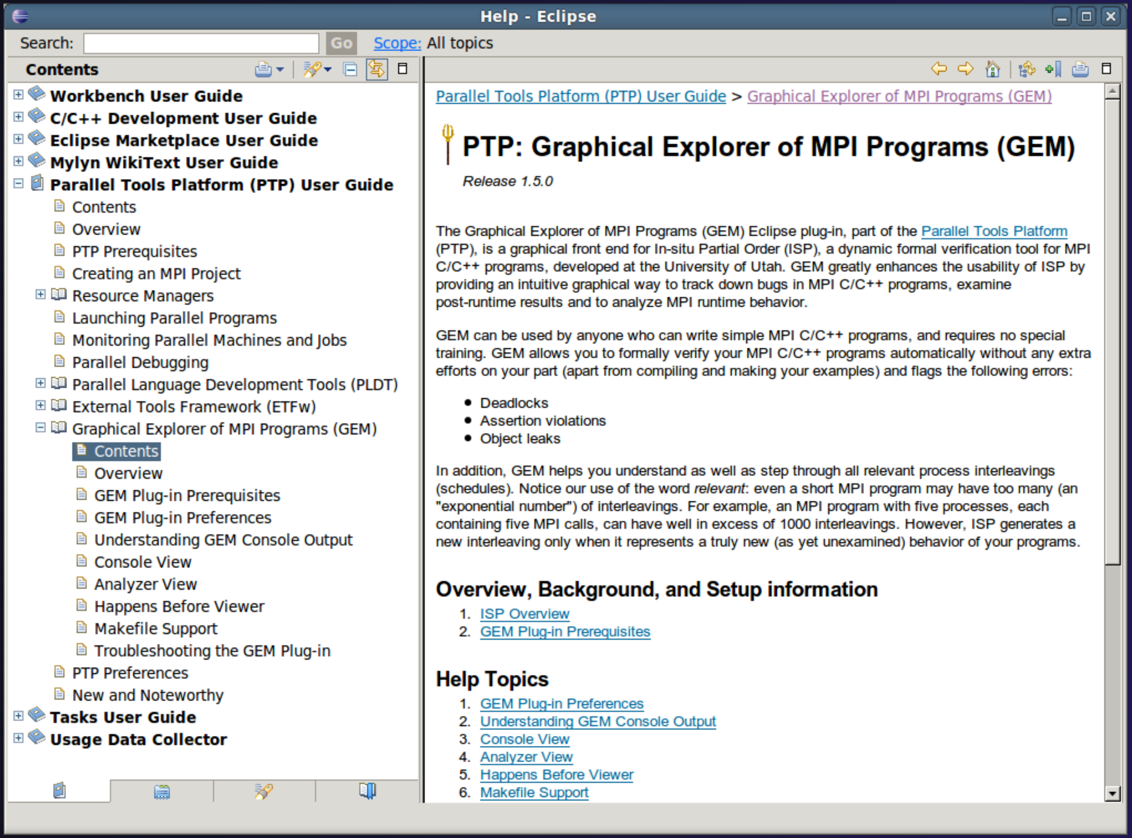


0 errors and 4 warnings were found

```
Interleaving: 1
    FIB  /MPI_ResourceLeakTest/src/comm-create-no-free.c    Line: 35
    FIB  /MPI_ResourceLeakTest/src/comm-create-no-free.c    Line: 47
```

GEM – Help Plugin

Extensive how-to sections, graphical aids and trouble shooting section



Help - Eclipse

Search: Go Scope: All topics

Contents

- Workbench User Guide
- C/C++ Development User Guide
- Eclipse Marketplace User Guide
- Mylyn WikiText User Guide
- Parallel Tools Platform (PTP) User Guide
  - Contents
  - Overview
  - PTP Prerequisites
  - Creating an MPI Project
  - Resource Managers
  - Launching Parallel Programs
  - Monitoring Parallel Machines and Jobs
  - Parallel Debugging
  - Parallel Language Development Tools (PLDT)
  - External Tools Framework (ETFw)
  - Graphical Explorer of MPI Programs (GEM)
    - Contents
    - Overview
    - GEM Plug-in Prerequisites
    - GEM Plug-in Preferences
    - Understanding GEM Console Output
    - Console View
    - Analyzer View
    - Happens Before Viewer
    - Makefile Support
    - Troubleshooting the GEM Plug-in
  - PTP Preferences
  - New and Noteworthy
- Tasks User Guide
- Usage Data Collector

Parallel Tools Platform (PTP) User Guide > Graphical Explorer of MPI Programs (GEM)

PTP: Graphical Explorer of MPI Programs (GEM)

Release 1.5.0

The Graphical Explorer of MPI Programs (GEM) Eclipse plug-in, part of the Parallel Tools Platform (PTP), is a graphical front end for In-situ Partial Order (ISP), a dynamic formal verification tool for MPI C/C++ programs, developed at the University of Utah. GEM greatly enhances the usability of ISP by providing an intuitive graphical way to track down bugs in MPI C/C++ programs, examine post-runtime results and to analyze MPI runtime behavior.

GEM can be used by anyone who can write simple MPI C/C++ programs, and requires no special training. GEM allows you to formally verify your MPI C/C++ programs automatically without any extra efforts on your part (apart from compiling and making your examples) and flags the following errors:

- Deadlocks
- Assertion violations
- Object leaks

In addition, GEM helps you understand as well as step through all relevant process interleavings (schedules). Notice our use of the word *relevant*: even a short MPI program may have too many (an "exponential number") of interleavings. For example, an MPI program with five processes, each containing five MPI calls, can have well in excess of 1000 interleavings. However, ISP generates a new interleaving only when it represents a truly new (as yet unexamined) behavior of your programs.

Overview, Background, and Setup information

1. ISP Overview
2. GEM Plug-in Prerequisites

Help Topics

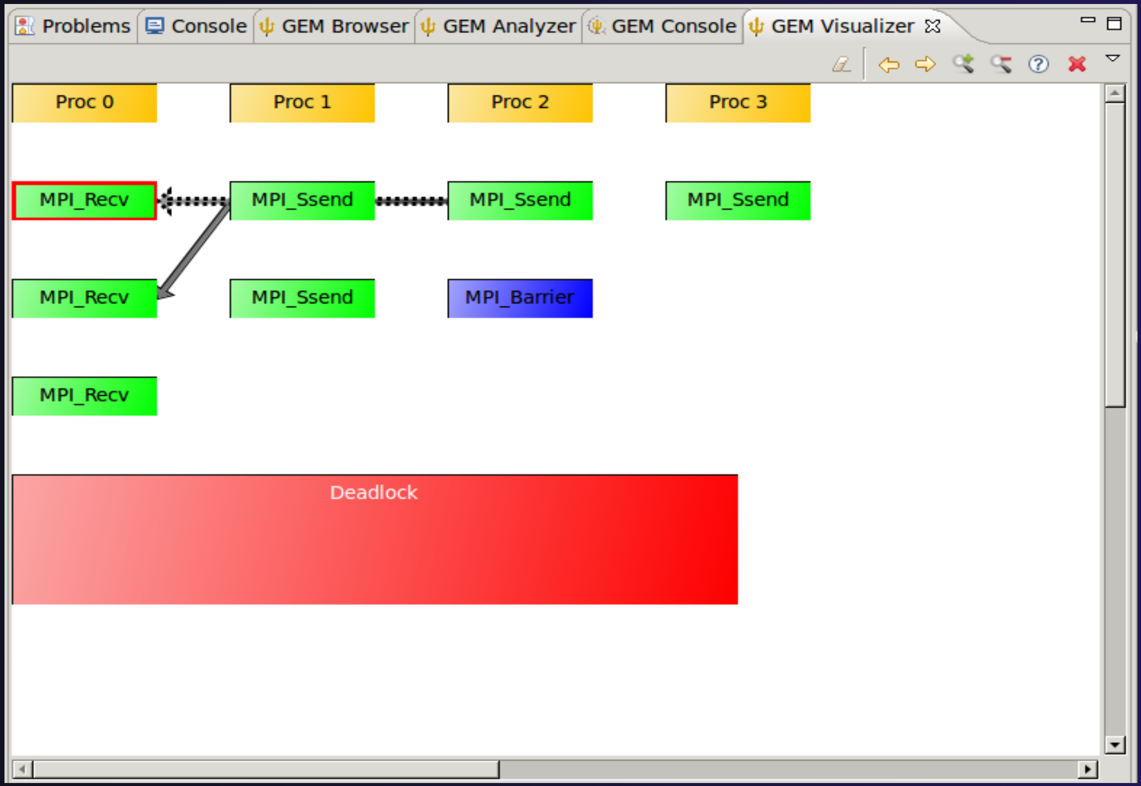
1. GEM Plug-in Preferences
2. Understanding GEM Console Output
3. Console View
4. Analyzer View
5. Happens Before Viewer
6. Makefile Support

GEM/ISP Success Stories

- Umpire Tests
  - http://www.cs.utah.edu/fv/ISP-Tests
  - Documents bugs missed by tests, caught by ISP
- MADRE (EuroPVM/MPI 2007)
  - Previously documented deadlock detected
- N-Body Simulation Code
  - Previously unknown resource leak caught during EuroPVM/MPI 2009 tutorial !
- Large Case Studies
  - ParMETIS, MPI-BLAST, IRS (Sequoia Benchmark), and a few SPEC-MPI benchmarks could be handled
- Full Tutorial including LiveDVD ISO available
  - Visit http://www.cs.utah.edu/fv/GEM

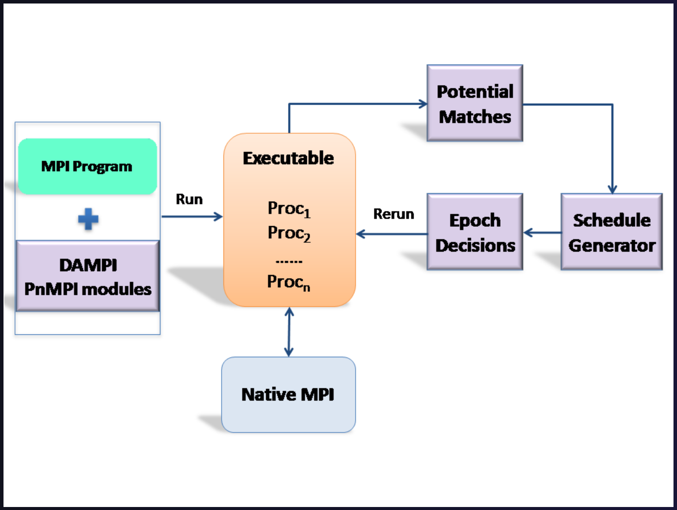
GEM Future Plans

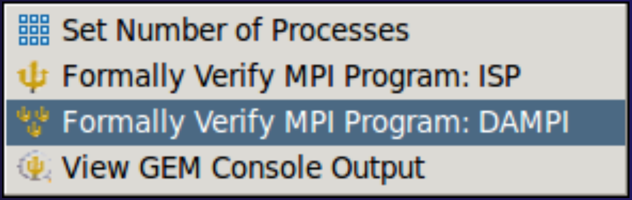
- Incorporation of HB Viewer into GEM as a new view
- Add Pthread support to visualize Pthread calls made from within MPI space



GEM Future Plans

- GEM will serve as a front-end for other tools
- Integration of Distributed Analyzer of MPI Programs (DAMPI), developed at University of Utah
  - ISP scales to 10s of processes
  - DAMPI scales to 1000s of processes (C/C++/Fortran)
  - Decentralized scheduler uses Lamport Clocks




Set Number of Processes
Formally Verify MPI Program: ISP
Formally Verify MPI Program: DAMPI
View GEM Console Output

Use **ISP** at small scale, then launch **DAMPI** at scale on a cluster

PTP Adv. Development: Summary

- A diversity of other tools aid parallel development
  - Parallel Language Development Tools: MPI, OpenMP, UPC, LAPI, etc.
  - External Tools Framework (ETFw) eases integration of existing (command-line, etc.) tools
    - TAU Performance Tuning uses ETFw
    - PPW (Parallel Perf. Wizard) uses ETFw for UPC analysis
    - Feedback view maps tool findings with source code
  - MPI Analysis: GEM
- A diversity of contributors too!
  - We welcome other contributions. Let us help!

Backup

- Not covered in today's tutorial, but included for reference
- Creating a local MPI project, and using the wizards
  - MPI Assistance tools
  - MPI Barrier analysis on a local project
- OpenMP tools
- UPC tools installation and local projects
- External Tools Framework (ETFw) details, overview of integrating other tools into PTP
- ETFw Feedback view incl. sample exercise

Parallel Lang. Dev. Tools

- PLDT Features
    - Analysis of C and C++ code to determine the location of MPI, OpenMP, and UPC Artifacts
  - Content assist via **ctrl+space** ("completion")
  - Hover help
  - Reference information about the API calls via Dynamic Help
  - New project wizard automatically configures managed build projects for MPI & OpenMP
  - OpenMP problems view of common errors
  - OpenMP "show #pragma region" , "show concurrency"
  - MPI Barrier analysis - detects potential deadlocks

Some MPI features were covered in Module 4

Note: Some PLDT features don't work on remote (RDT) projects

MPI Assistance Tools

Added by PLDT (Parallel Lang. Dev. Tools) feature of PTP

- MPI Context sensitive help
- MPI artifact locations
- MPI barrier analysis
- MPI templates

- For this part, we will use the *local* MPI New Project Wizard and the “MPI Hello World” project

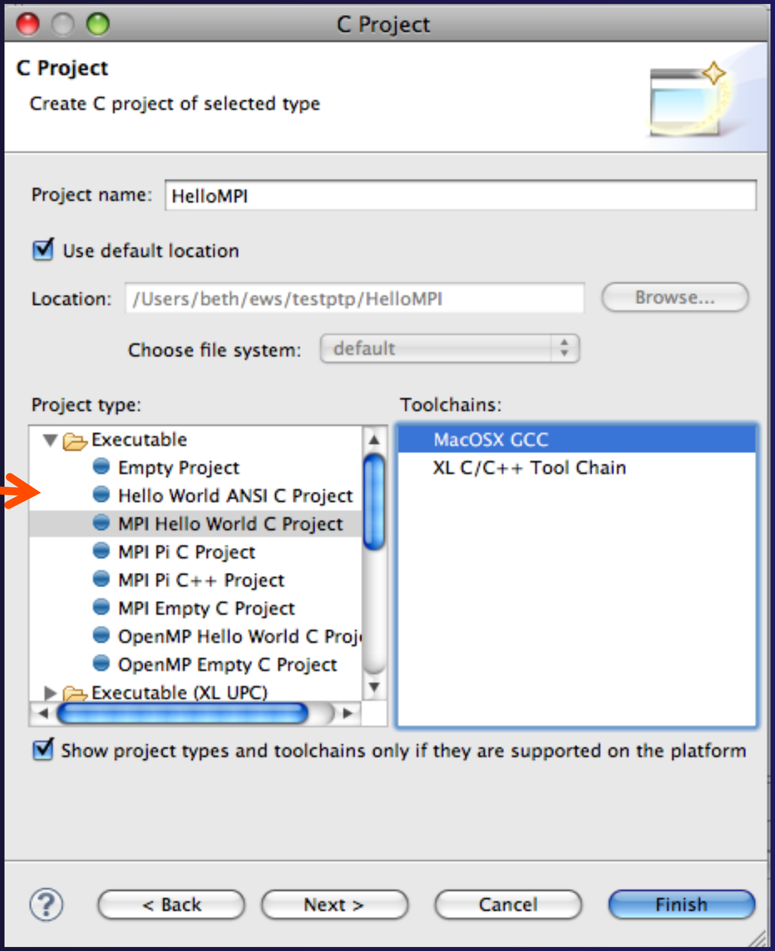
Creating Local Project

- The next slide shows you how to create a local MPI project.
- If you do not have MPI on your local machine, you can't build or run.
- *But* you should be able to demonstrate the MPI features in PTP's PLDT regardless.
- Several PLDT MPI features pertain to developing code – just using the local editor, etc.
- Most PLDT features *do* work on remote projects.

Create local MPI Project

Using a Managed Build Project – for a quick sample *local* MPI project

- ✦**File > New > C Project**
- ✦Give Project a name, e.g. HelloMPI
- ✦Confirm Toolchain
- ✦Select **MPI Hello World C Project**


C Project

C Project

Create C project of selected type

Project name: HelloMPI

[✓] Use default location

Location: /Users/beth/ews/testptp/HelloMPI

Choose file system: default

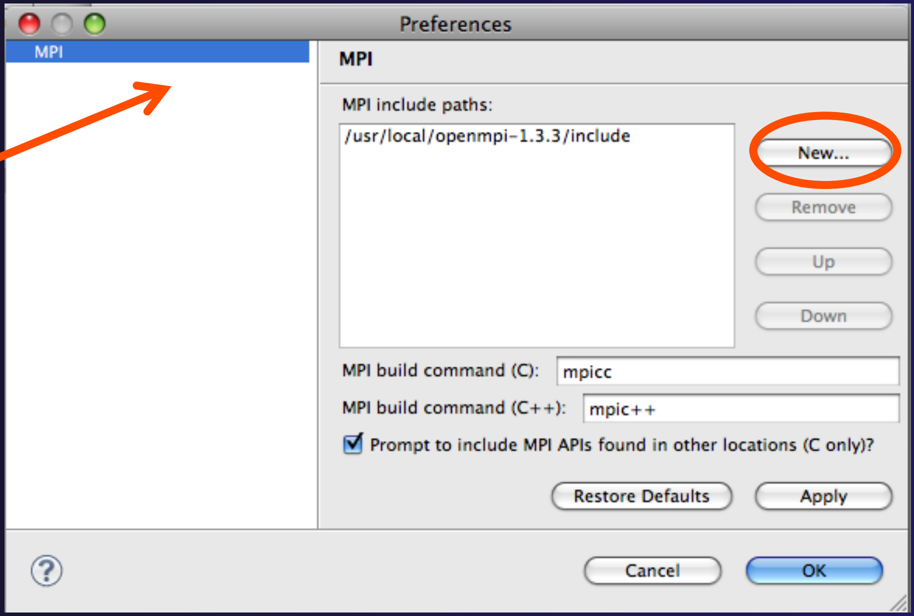
Project type:
- Executable
  - Empty Project
  - Hello World ANSI C Project
  - MPI Hello World C Project
  - MPI Pi C Project
  - MPI Pi C++ Project
  - MPI Empty C Project
  - OpenMP Hello World C Proj
  - OpenMP Empty C Project
- Executable (XL UPC)

Toolchains:
- MacOSX GCC
- XL C/C++ Tool Chain

[✓] Show project types and toolchains only if they are supported on the platform

Set MPI Preferences

- When creating a local MPI project with the wizard, you need to set MPI Preferences (once)
- This assures the include paths, etc. will be set for new MPI projects – for building, and for Eclipse assistance features for MPI.
- Select **Yes** to set the MPI preferences.

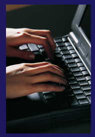


MPI include paths:
/usr/local/openmpi-1.3.3/include

MPI build command (C): mpicc
MPI build command (C++): mpic++
[✓] Prompt to include MPI APIs found in other locations (C only)?

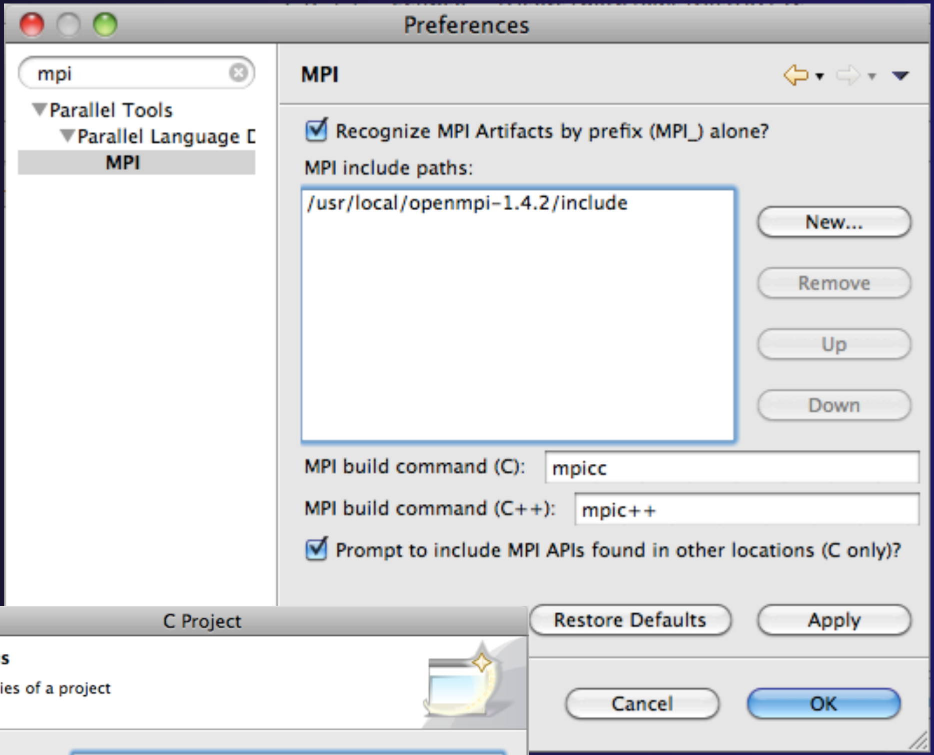
No MPI?

Note: if you do not have MPI on your local machine, you can use just an MPI header file (mpi.h) so you play with the PTP MPI development features without building or running on your local machine.



Set MPI Preferences (2)

- On the MPI Preferences page, add a new MPI include path.
- New … and point to the *directory* containing your MPI header file (mpi.h)
- Select **OK**
- Back on New Project Wizard page, select **Next>** and fill in Author name, etc.



☑ Recognize MPI Artifacts by prefix (MPI_) alone?

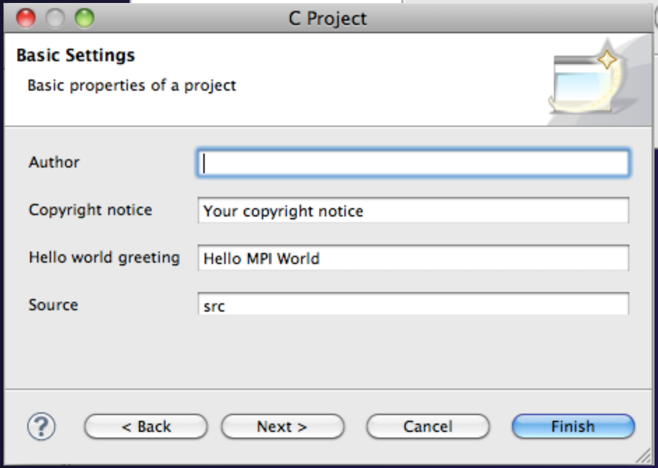
MPI include paths:

/usr/local/openmpi-1.4.2/include

MPI build command (C): mpicc

MPI build command (C++): mpic++

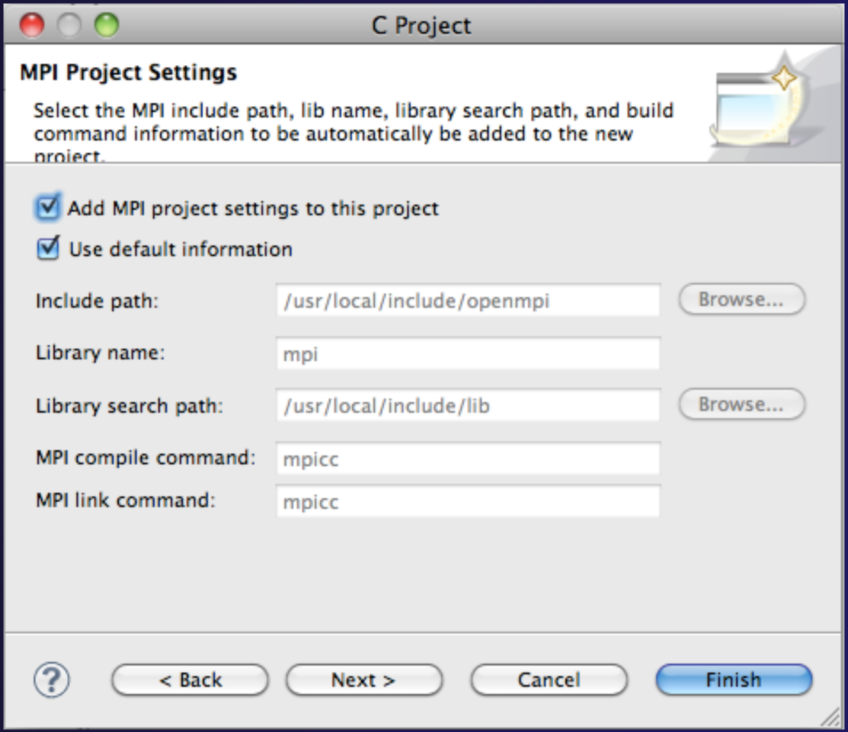
☑ Prompt to include MPI APIs found in other locations (C only)?



Author	
Copyright notice	Your copyright notice
Hello world greeting	Hello MPI World
Source	src

Review MPI Project Settings

- On the next wizard page, review the MPI project settings based on the information you have provided.
- Make changes if you wish.
- The defaults should be fine.
- Click **Finish.**
- You will be prompted to switch perspectives



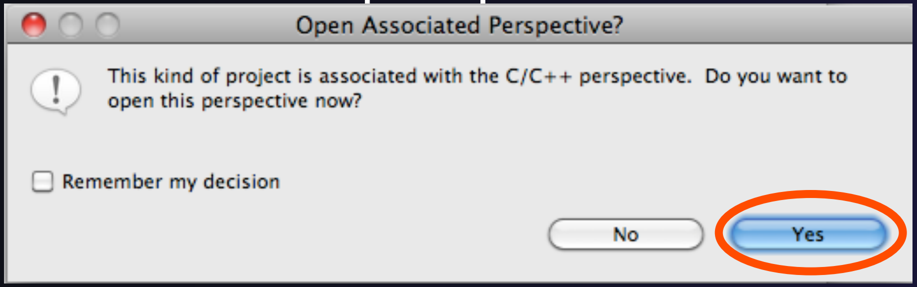
C Project

MPI Project Settings

Select the MPI include path, lib name, library search path, and build command information to be automatically be added to the new project.

- [✓] Add MPI project settings to this project
- [✓] Use default information

Include path:	/usr/local/include/openmpi
Library name:	mpi
Library search path:	/usr/local/include/lib
MPI compile command:	mpicc
MPI link command:	mpicc



Open Associated Perspective?

This kind of project is associated with the C/C++ perspective. Do you want to open this perspective now?

- [ ] Remember my decision

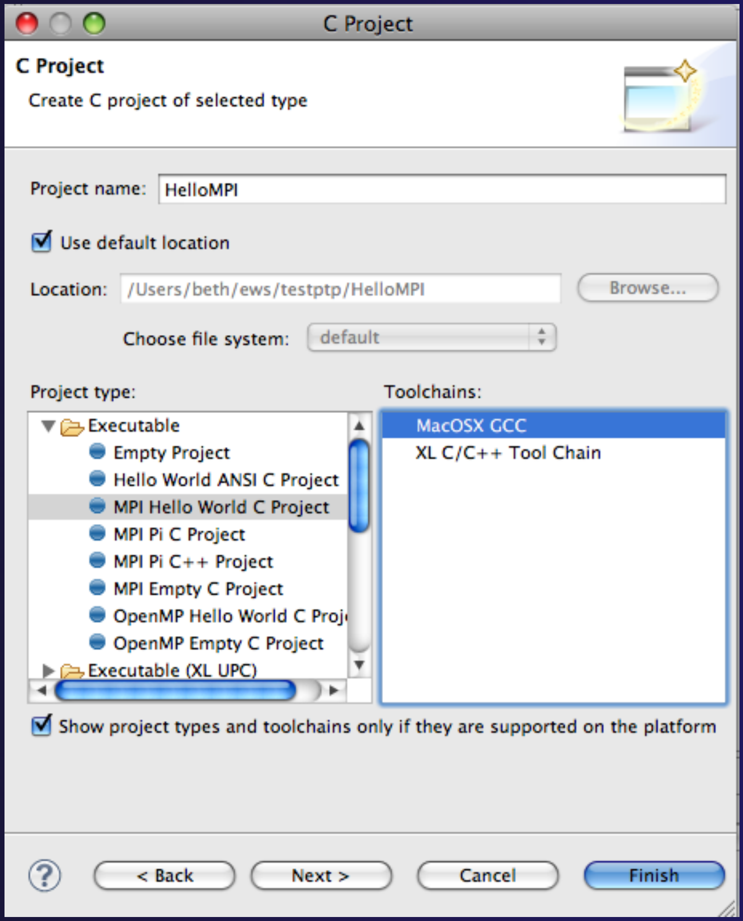
No [✓ Yes]

Create MPI Project

Recap:

- File > New > C Project
- Give Project a name, e.g. HelloMPI
- Select Toolchain
- Select MPI Hello World C Project
- Set MPI Prefs, if first time
- Click Finish

- Note: if it doesn't build on your machine, you can still continue with this exercise



C Project

C Project

Create C project of selected type

Project name: HelloMPI

[✓] Use default location

Location: /Users/beth/ews/testptp/HelloMPI

Choose file system: default

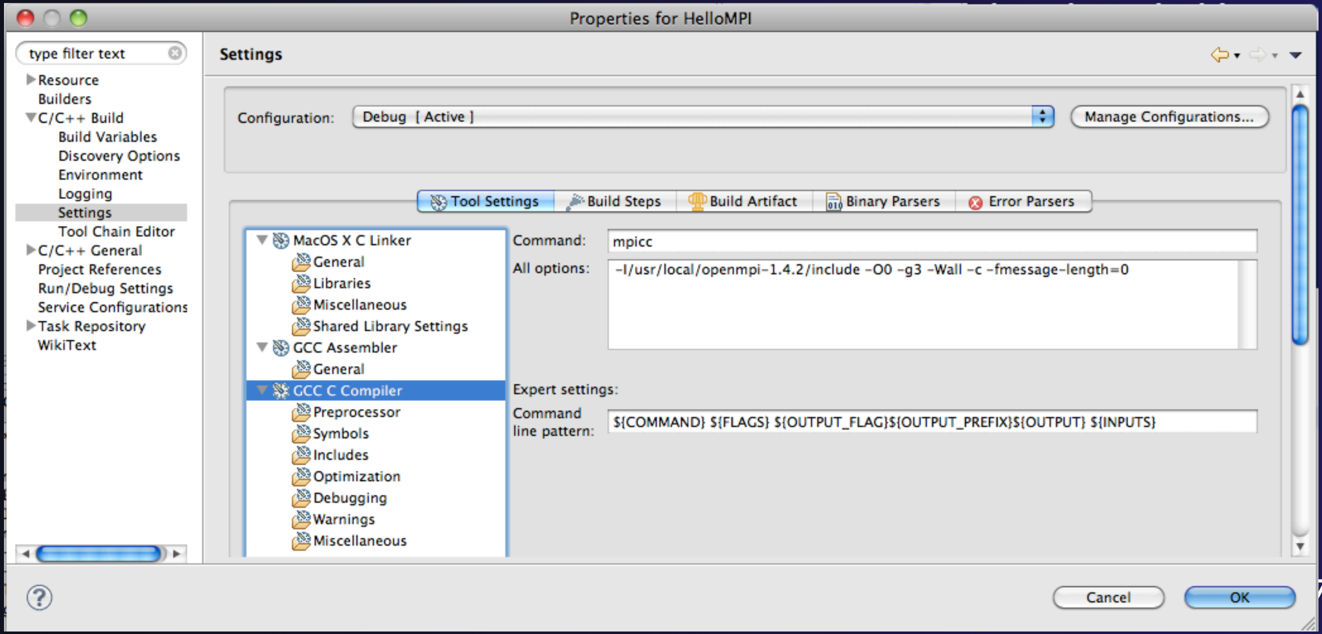
Project type:
- Executable
  - Empty Project
  - Hello World ANSI C Project
  - MPI Hello World C Project
  - MPI Pi C Project
  - MPI Pi C++ Project
  - MPI Empty C Project
  - OpenMP Hello World C Proj
  - OpenMP Empty C Project
- Executable (XL UPC)

Toolchains:
- MacOSX GCC
- XL C/C++ Tool Chain

[✓] Show project types and toolchains only if they are supported on the platform

Project Properties: Managed Build Project

- Right-click on project in Project Explorer view and select **Properties**
- Project Properties for Managed Build project
  - Compiler, Linker, etc. settings set automatically without a Makefile



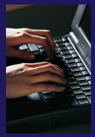
Configuration: Debug [Active]

Command: mpicc

All options: -I/usr/local/openmpi-1.4.2/include -O0 -g3 -Wall -c -fmessage-length=0

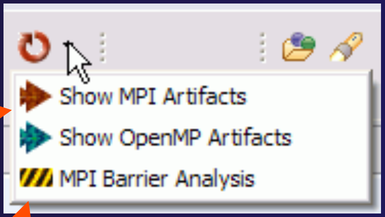
Expert settings:

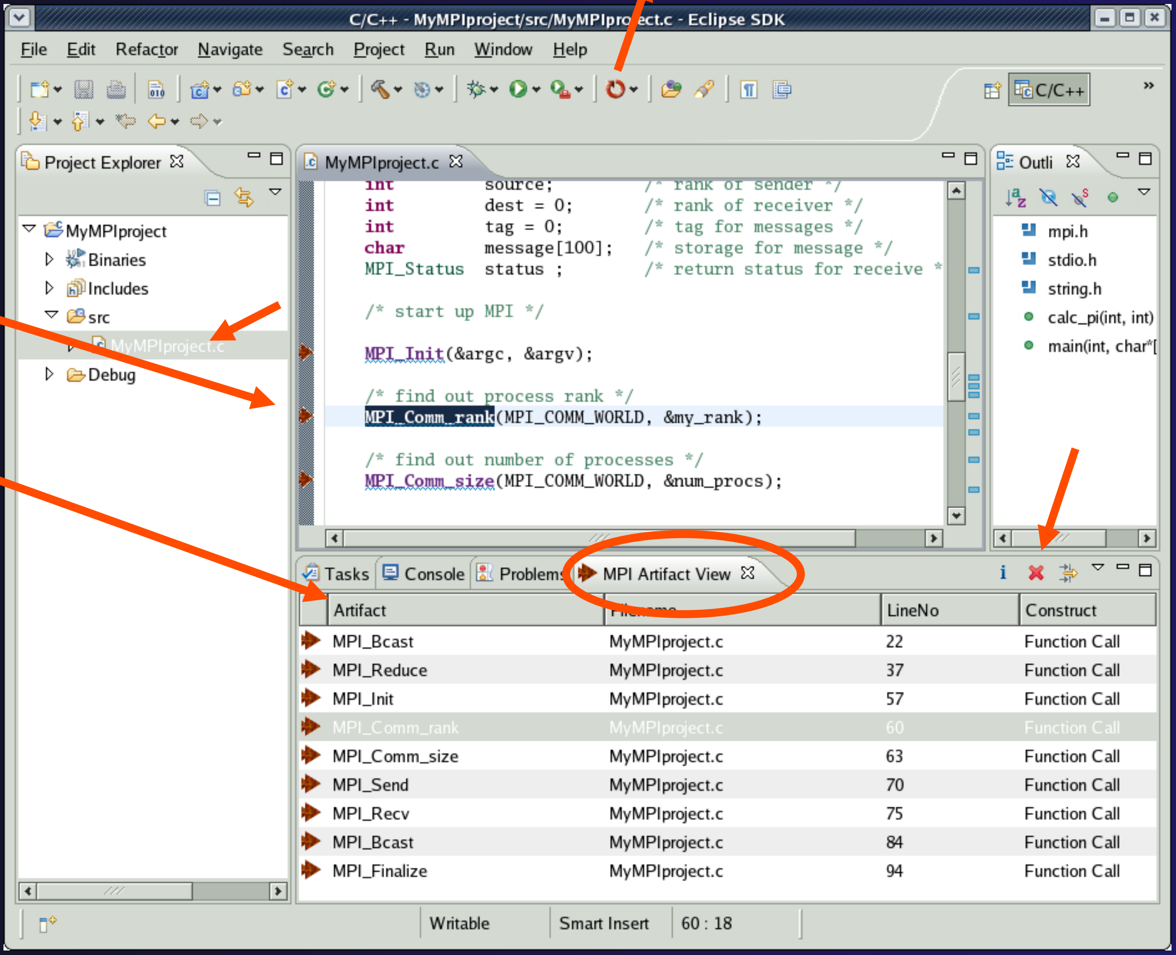
Command line pattern: ${COMMAND} ${FLAGS} ${OUTPUT_FLAG}${OUTPUT_PREFIX}${OUTPUT} ${INPUTS}



Show MPI Artifacts

- Select source file in Project Explorer; Select **Show MPI Artifacts** in PLDT menu
- Markers indicate the location of artifacts in editor
- In **MPI Artifact View** sort by any column (click on col. heading)
- Navigate to source code line by double-clicking on the artifact
- Run the analysis on another file and its markers will be added to the view
- Remove markers via



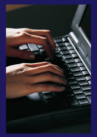


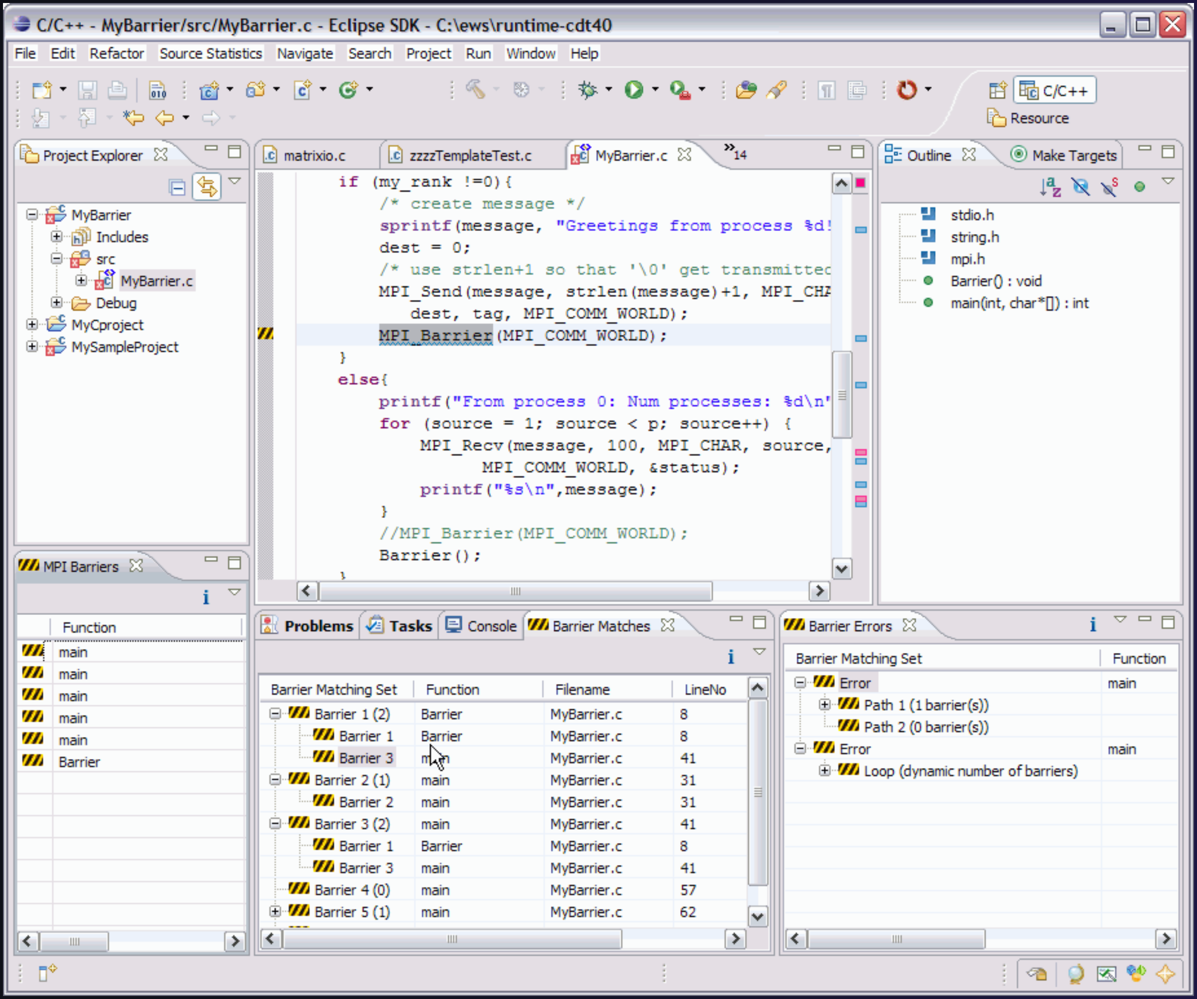
```
int         source;        /* rank of sender */
int         dest = 0;      /* rank of receiver */
int         tag = 0;       /* tag for messages */
char        message[100];  /* storage for message */
MPI_Status  status ;       /* return status for receive *

/* start up MPI */

MPI_Init(&argc, &argv);

/* find out process rank */
MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);

/* find out number of processes */
MPI_Comm_size(MPI_COMM_WORLD, &num_procs);
```

Artifact	Filename	LineNo	Construct
MPI_Bcast	MyMPIproject.c	22	Function Call
MPI_Reduce	MyMPIproject.c	37	Function Call
MPI_Init	MyMPIproject.c	57	Function Call
MPI_Comm_rank	MyMPIproject.c	60	Function Call
MPI_Comm_size	MyMPIproject.c	63	Function Call
MPI_Send	MyMPIproject.c	70	Function Call
MPI_Recv	MyMPIproject.c	75	Function Call
MPI_Bcast	MyMPIproject.c	84	Function Call
MPI_Finalize	MyMPIproject.c	94	Function Call

MPI Barrier Analysis

Local files only





```
    if (my_rank !=0){
        /* create message */
        sprintf(message, "Greetings from process %d!
        dest = 0;
        /* use strlen+1 so that '\0' get transmittec
        MPI_Send(message, strlen(message)+1, MPI_CHA
            dest, tag, MPI_COMM_WORLD);
        MPI_Barrier(MPI_COMM_WORLD);
    }
    else{
        printf("From process 0: Num processes: %d\n"
        for (source = 1; source < p; source++) {
            MPI_Recv(message, 100, MPI_CHAR, source,
                    MPI_COMM_WORLD, &status);
            printf("%s\n",message);
        }
        //MPI_Barrier(MPI_COMM_WORLD);
        Barrier();
```

Function
main
main
main
main
main
Barrier

Barrier Matching Set	Function	Filename	LineNo
Barrier 1 (2)	Barrier	MyBarrier.c	8
Barrier 1	Barrier	MyBarrier.c	8
Barrier 3	main	MyBarrier.c	41
Barrier 2 (1)	main	MyBarrier.c	31
Barrier 2	main	MyBarrier.c	31
Barrier 3 (2)	main	MyBarrier.c	41
Barrier 1	Barrier	MyBarrier.c	8
Barrier 3	main	MyBarrier.c	41
Barrier 4 (0)	main	MyBarrier.c	57
Barrier 5 (1)	main	MyBarrier.c	62

Barrier Matching Set	Function
Error	main
Path 1 (1 barrier(s))	
Path 2 (0 barrier(s))	
Error	main
Loop (dynamic number of barriers)	

Verify barrier synchronization in C/MPI programs

Interprocedural static analysis outputs:

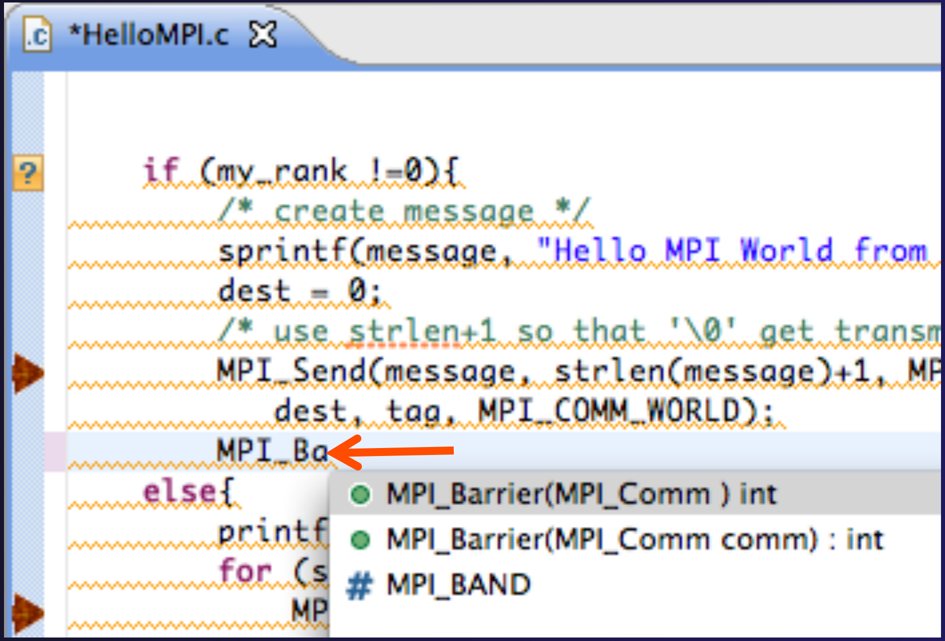
✦For verified programs, lists barrier statements that synchronize together (match)

✦ For synchronization errors, reports counter example that illustrates and explains the error

MPI Barrier Analysis – Try it

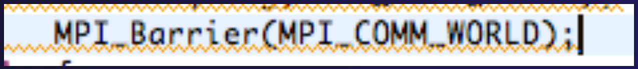
Add some barriers:

- Inside the sample if (rank…) add a barrier:
- Use Content Assist to help you type
- Type: MPI_ and press Ctrl-space. See completion alternatives. Keep typing until you see MPI_Barrier and hit enter.
- For args, start typing MPI_Comm_ etc. and it will also complete MPI_COMM_WORLD
- Add the same barrier statement at the end of the **else** as well.



*HelloMPI.c

```
    if (my_rank !=0){
        /* create message */
        sprintf(message, "Hello MPI World from
        dest = 0;
        /* use strlen+1 so that '\0' get transm
        MPI_Send(message, strlen(message)+1, MP
            dest, tag, MPI_COMM_WORLD);
        MPI_Ba
    else{
        printf
        for (s
            MP
```



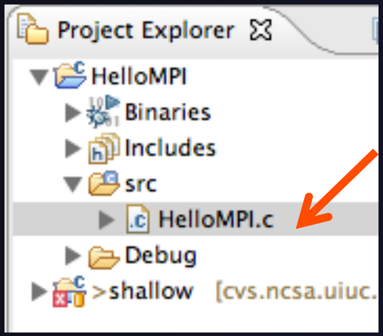
```
MPI_Barrier(MPI_COMM_WORLD);
```

Resulting statement

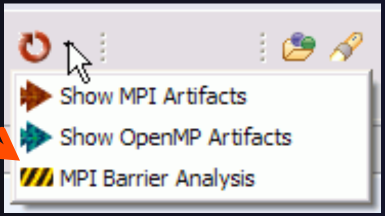
MPI Barrier Analysis – Try it (2)

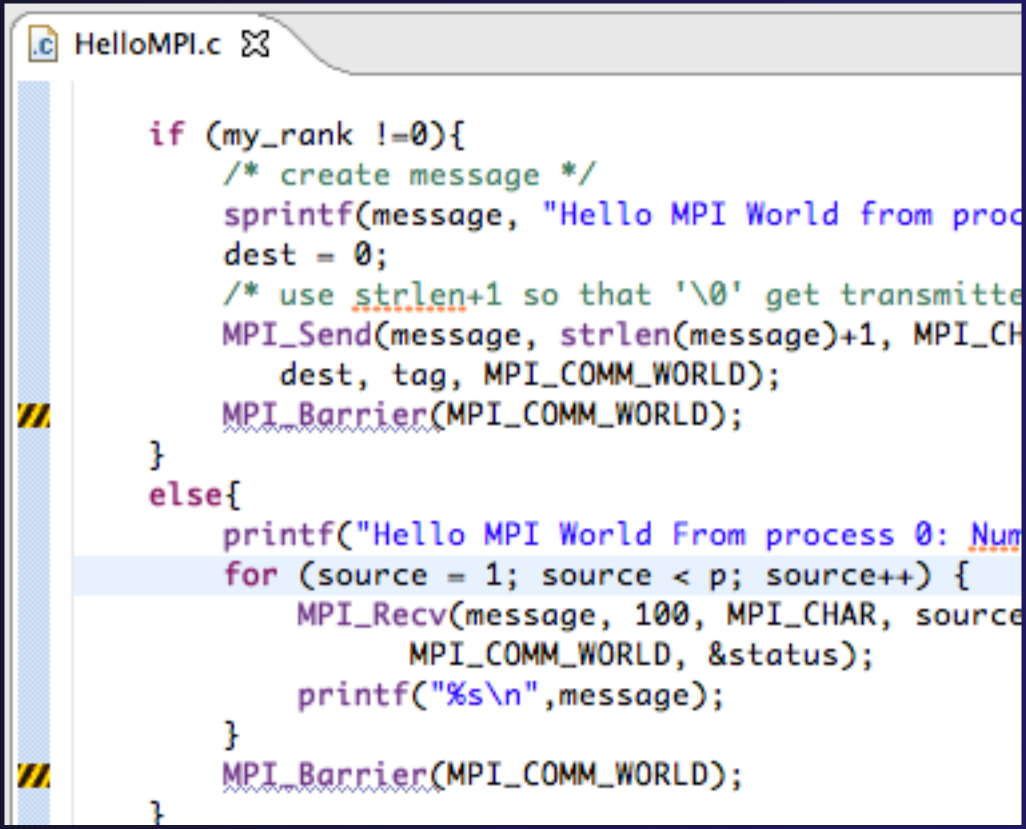
Run the Analysis:

- In the Project Explorer, Select the source file (or directory, or project) of file(s) to analyze



- Select the MPI Barrier Analysis action in the menu

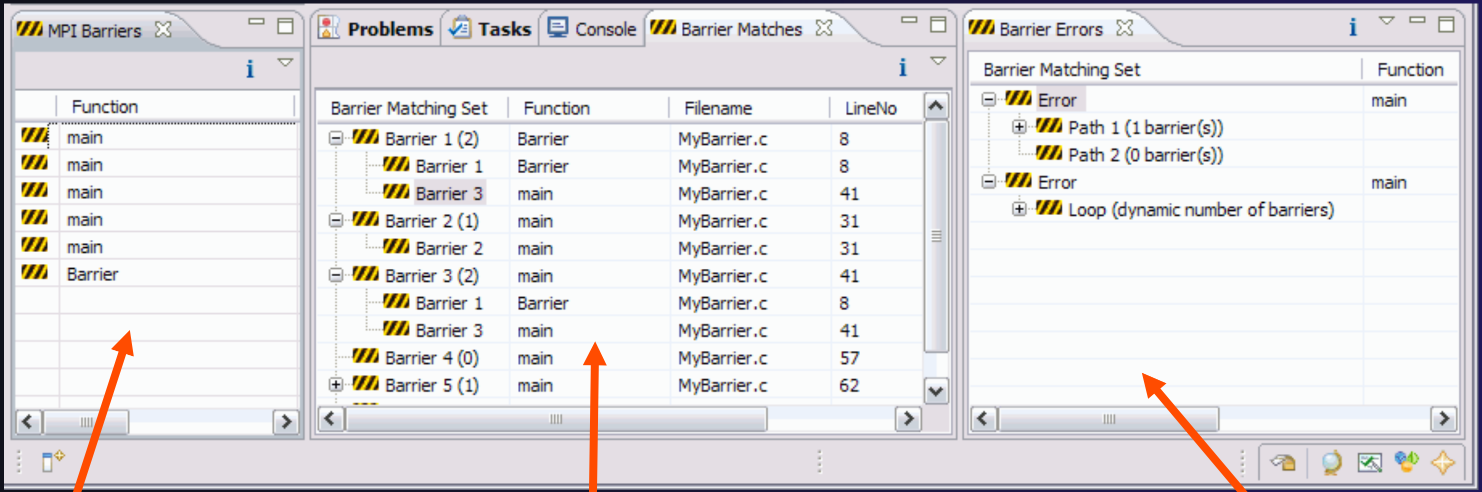




HelloMPI.c

```
if (my_rank !=0){
    /* create message */
    sprintf(message, "Hello MPI World from proc
    dest = 0;
    /* use strlen+1 so that '\0' get transmitte
    MPI_Send(message, strlen(message)+1, MPI_CH
        dest, tag, MPI_COMM_WORLD);
    MPI_Barrier(MPI_COMM_WORLD);
}
else{
    printf("Hello MPI World From process 0: Num
    for (source = 1; source < p; source++) {
        MPI_Recv(message, 100, MPI_CHAR, source
            MPI_COMM_WORLD, &status);
        printf("%s\n",message);
    }
    MPI_Barrier(MPI_COMM_WORLD);
}
```

MPI Barrier Analysis - views



MPI Barriers

Function
main
main
main
main
main
Barrier

Barrier Matches

Barrier Matching Set	Function	Filename	LineNo
Barrier 1 (2)	Barrier	MyBarrier.c	8
Barrier 1	Barrier	MyBarrier.c	8
Barrier 3	main	MyBarrier.c	41
Barrier 2 (1)	main	MyBarrier.c	31
Barrier 2	main	MyBarrier.c	31
Barrier 3 (2)	main	MyBarrier.c	41
Barrier 1	Barrier	MyBarrier.c	8
Barrier 3	main	MyBarrier.c	41
Barrier 4 (0)	main	MyBarrier.c	57
Barrier 5 (1)	main	MyBarrier.c	62

Barrier Errors

Barrier Matching Set	Function
Error	main
Path 1 (1 barrier(s))	
Path 2 (0 barrier(s))	
Error	main
Loop (dynamic number of barriers)	

MPI Barriers view

Simply lists the barriers

Like MPI Artifacts view, double-click to navigate to source code line (all 3 views)

Barrier Matches view

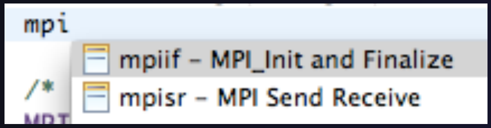
Groups barriers that match together in a barrier set – all processes must go through a barrier in the set to prevent a deadlock

Barrier Errors view

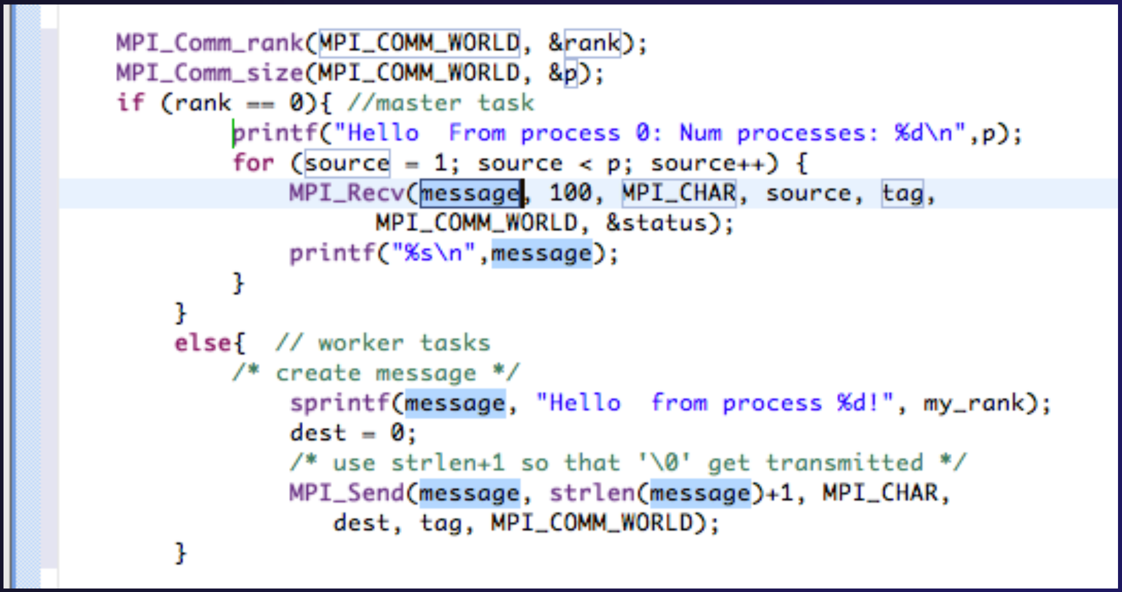
If there are errors, a counter-example shows paths with mismatched number of barriers

MPI Templates

- Allows quick entry of common patterns in MPI programming
- Example: MPI send-receive
- Enter: mpisr <ctrl-space>
- Expands to the code shown at right
- Highlighted variable names can all be changed at once
- Type mpi <ctrl-space> <ctrl-space> to see all templates



```
mpi
    mpiif - MPI_Init and Finalize
/*  mpisr - MPI Send Receive
MPI
```



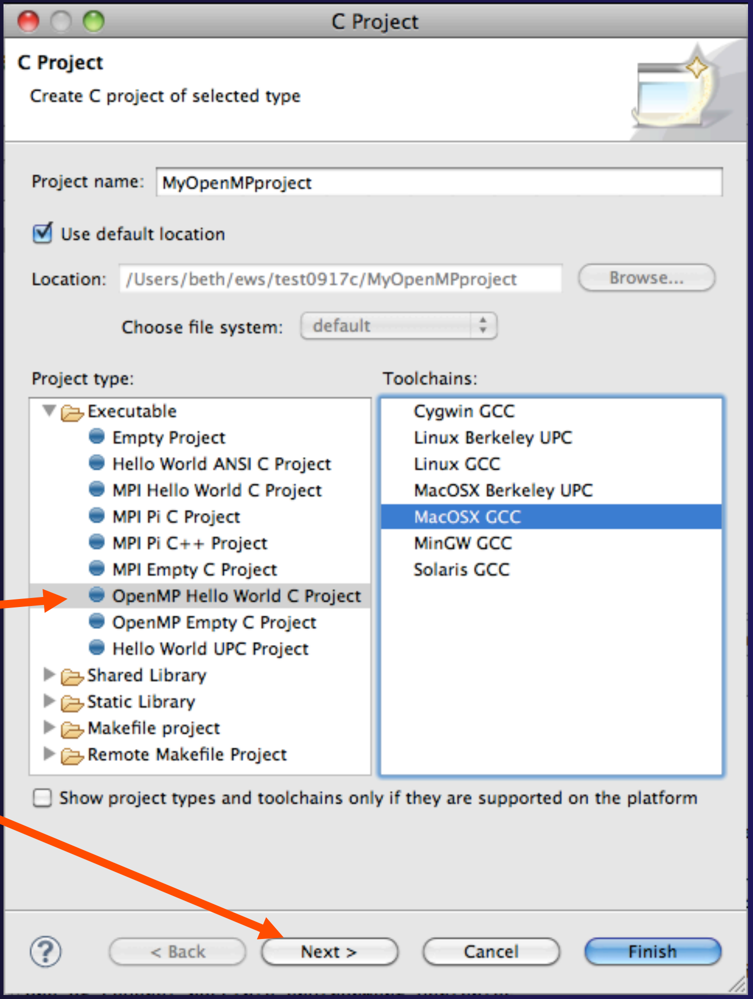
```
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &p);
if (rank == 0){ //master task
        printf("Hello  From process 0: Num processes: %d\n",p);
        for (source = 1; source < p; source++) {
            MPI_Recv(message, 100, MPI_CHAR, source, tag,
                  MPI_COMM_WORLD, &status);
            printf("%s\n",message);
        }
    }
    else{  // worker tasks
        /* create message */
            sprintf(message, "Hello  from process %d!", my_rank);
            dest = 0;
            /* use strlen+1 so that '\0' get transmitted */
            MPI_Send(message, strlen(message)+1, MPI_CHAR,
               dest, tag, MPI_COMM_WORLD);
    }
```

- Eclipse preferences: add more!
  - C/C++ > Editor > Templates
- Extend to other common patterns

OpenMP Managed Build Project

Local files only

- This will need OpenMP preferences (e.g. include file location) set up as well
- Create a new OpenMP project
  - **File▶New▶C Project**
  - Name the project e.g. 'MyOpenMPproject'
  - Select Toolchain
  - Select **OpenMP Hello World C Project**
  - Select **Next,** then fill in other info like MPI project



C Project

C Project

Create C project of selected type

Project name: MyOpenMPproject

☑ Use default location

Location: /Users/beth/ews/test0917c/MyOpenMPproject

Choose file system: default

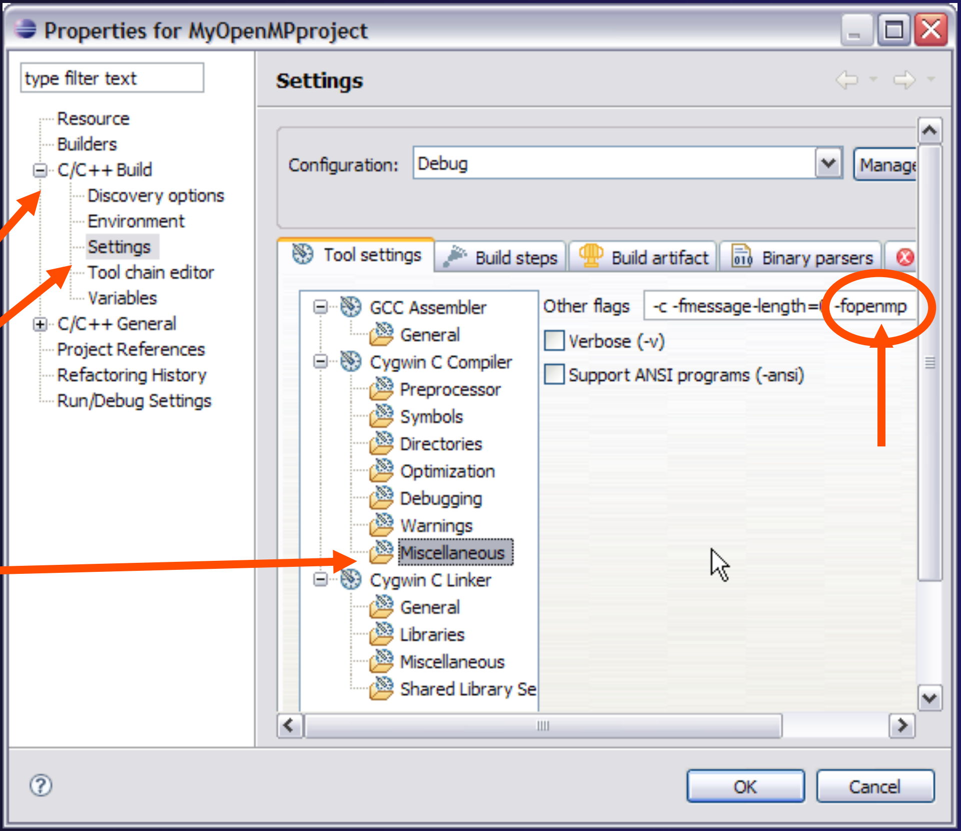
Project type:
- Executable
  - Empty Project
  - Hello World ANSI C Project
  - MPI Hello World C Project
  - MPI Pi C Project
  - MPI Pi C++ Project
  - MPI Empty C Project
  - OpenMP Hello World C Project
  - OpenMP Empty C Project
  - Hello World UPC Project
- Shared Library
- Static Library
- Makefile project
- Remote Makefile Project

Toolchains:
- Cygwin GCC
- Linux Berkeley UPC
- Linux GCC
- MacOSX Berkeley UPC
- MacOSX GCC
- MinGW GCC
- Solaris GCC

☐ Show project types and toolchains only if they are supported on the platform

Setting OpenMP Special Build Options

- OpenMP typically requires special compiler options.
  - Open the project properties
  - Expand **C/C++ Build**
  - Select **Settings**
  - Select **C Compiler**
    - In Miscellaneous, add option(s). -fopenmp
- Click **OK**; Project should attempt to build


Properties for MyOpenMPproject

Settings

Configuration: Debug

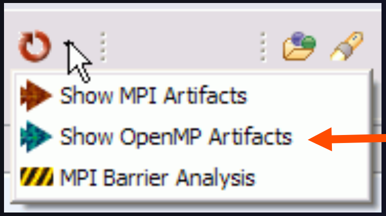
Other flags: -c -fmessage-length=0 -fopenmp

Verbose (-v) [ ]

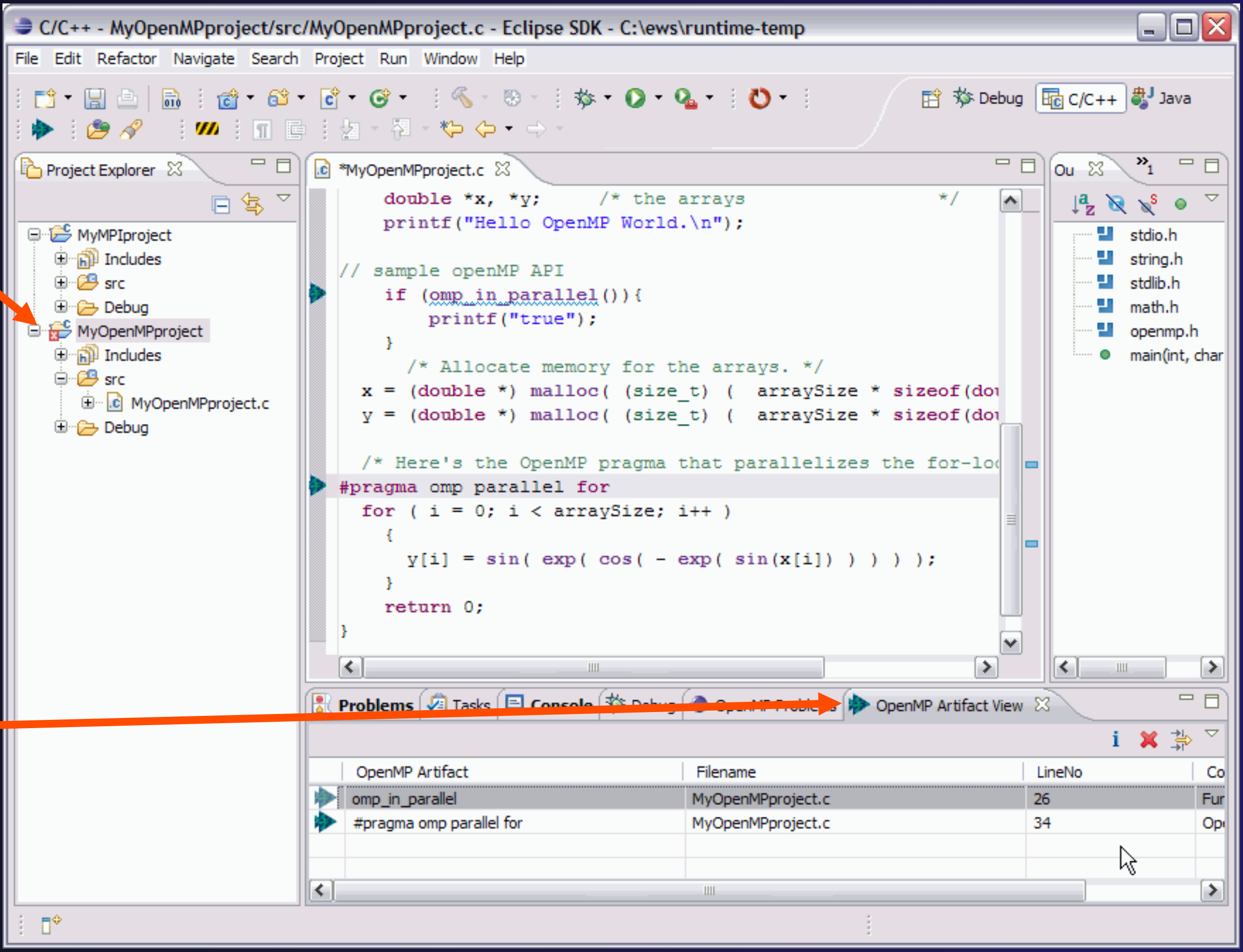
Support ANSI programs (-ansi) [ ]

Show OpenMP Artifacts

- Select source file, folder, or project
- Run analysis



- See artifacts in **OpenMP Artifact view**

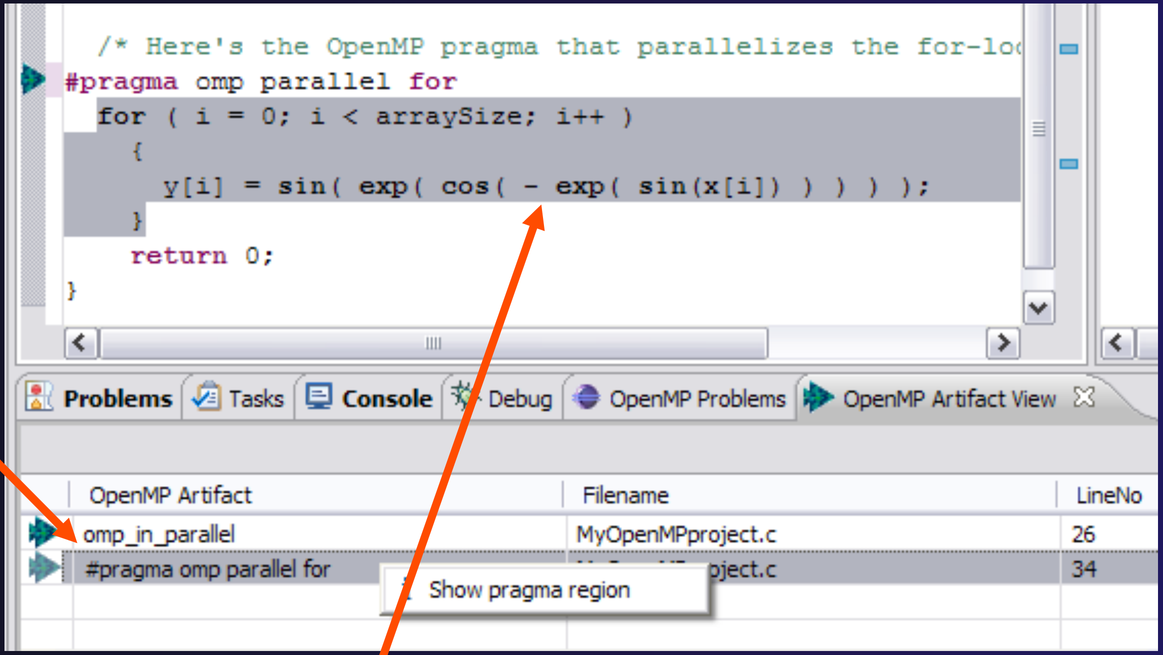

C/C++ - MyOpenMPproject/src/MyOpenMPproject.c - Eclipse SDK - C:\ews\runtime-temp
File Edit Refactor Navigate Search Project Run Window Help
Debug C/C++ Java
Project Explorer
MyMPIproject
Includes
src
Debug
MyOpenMPproject
Includes
src
MyOpenMPproject.c
Debug
*MyOpenMPproject.c
Ou
stdio.h
string.h
stdlib.h
math.h
openmp.h
main(int, char
Problems Tasks Console Debug OpenMP Problems OpenMP Artifact View

```
    double *x, *y;      /* the arrays                    */
    printf("Hello OpenMP World.\n");

// sample openMP API
    if (omp_in_parallel()){
        printf("true");
    }
      /* Allocate memory for the arrays. */
  x = (double *) malloc( (size_t) (  arraySize * sizeof(dou
  y = (double *) malloc( (size_t) (  arraySize * sizeof(dou

  /* Here's the OpenMP pragma that parallelizes the for-loo
#pragma omp parallel for
  for ( i = 0; i < arraySize; i++ )
    {
      y[i] = sin( exp( cos( - exp( sin(x[i]) ) ) ) );
    }
    return 0;
}
```

OpenMP Artifact	Filename	LineNo	Co
omp_in_parallel	MyOpenMPproject.c	26	Fur
#pragma omp parallel for	MyOpenMPproject.c	34	Op

Show Pragma Region

- Run OpenMP analysis
- Right click on pragma in artifact view
- Select **Show pragma region**
- See highlighted region in C editor



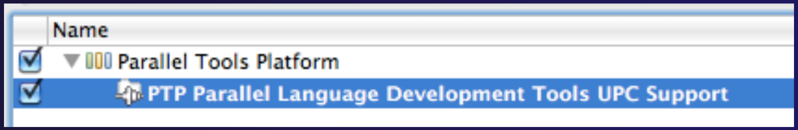
```
  /* Here's the OpenMP pragma that parallelizes the for-loo
#pragma omp parallel for
  for ( i = 0; i < arraySize; i++ )
    {
      y[i] = sin( exp( cos( - exp( sin(x[i]) ) ) ) );
    }
    return 0;
}
```

OpenMP Artifact	Filename	LineNo
omp_in_parallel	MyOpenMPproject.c	26
#pragma omp parallel for	...oject.c	34

UPC

UPC Features Installation

- If you installed PTP PLDT UPC feature, you *should* have CDT UPC feature too



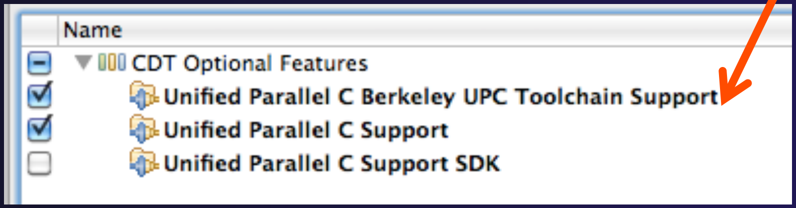
Name

☑ Parallel Tools Platform

☑ PTP Parallel Language Development Tools UPC Support

- See Also: http://wiki.eclipse.org/PTP/other_tools_setup#Using_UPC_features
- You can also install UPC features from the CDT-specific update site
  - Enable it in update manager
  - Help, Install New Software, Click **available Software Sites** link
  - Check the CDT site: http://download.eclipse.org/tools/cdt/releases/helios
  - Click OK to return to Install dialog
  - In **Work with:** select the CDT site you enabled
  - Check UPC features
  - Finish install and restart

BUPC toolchain only on CDT site



Name

▣ CDT Optional Features

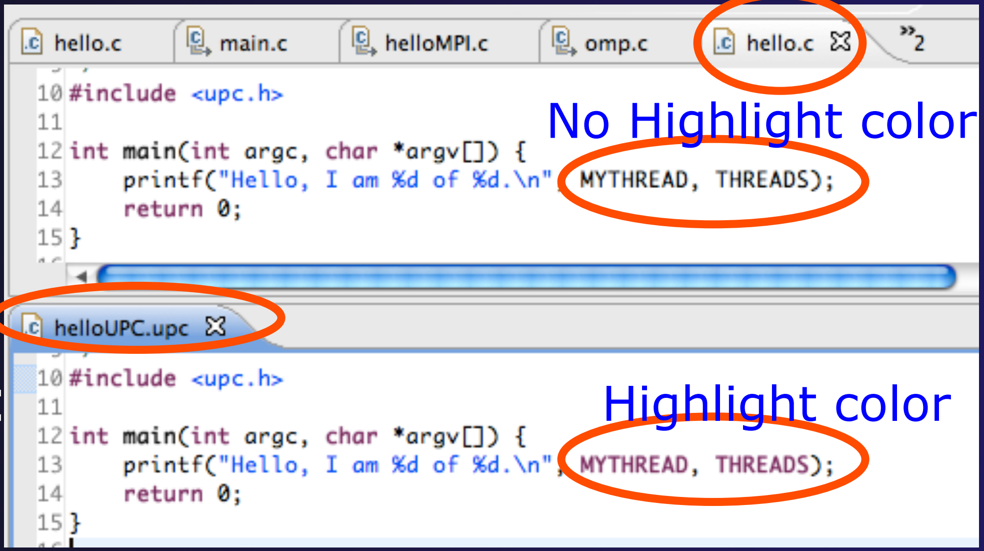
☑ Unified Parallel C Berkeley UPC Toolchain Support

☑ Unified Parallel C Support

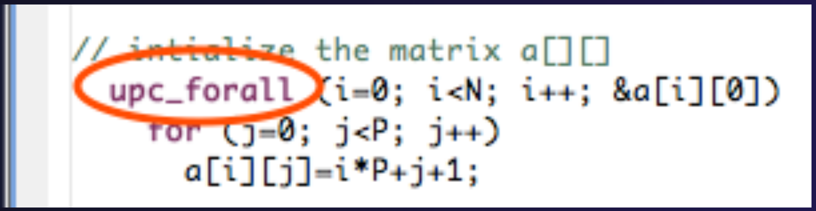
☐ Unified Parallel C Support SDK

UPC syntax in .c files

- ✦ UPC syntax is recognized by the parser in *.upc files
- ✦ Copy helloUPC.upc to hello.c to see the difference



```
#include <upc.h>

int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, THREADS);
    return 0;
}
```

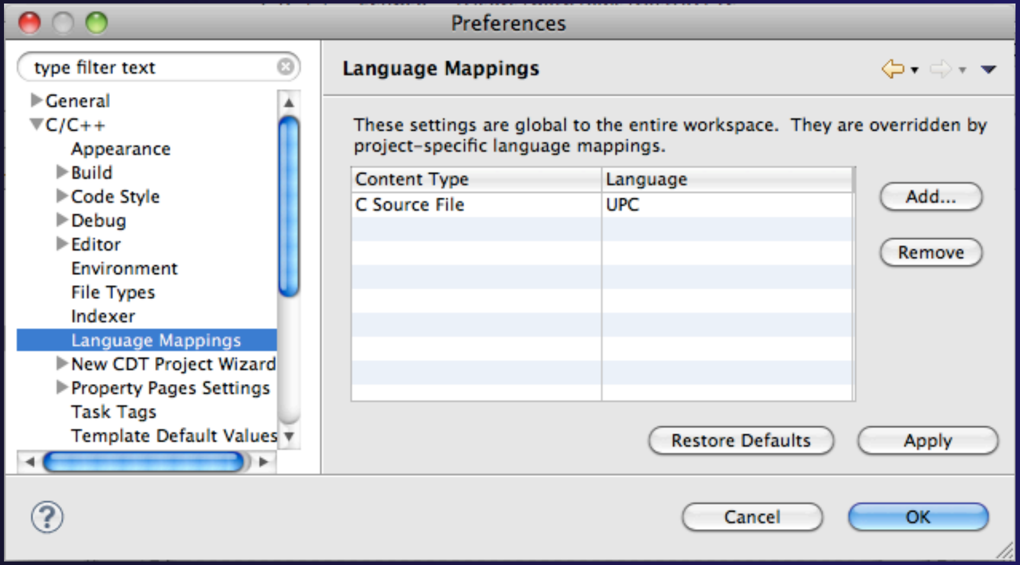


```
// initialize the matrix a[][]
upc_forall (i=0; i<N; i++; &a[i][0])
   for (j=0; j<P; j++)
      a[i][j]=i*P+j+1;
```

Keywords as well as new syntax are recognized

UPC syntax in .c files (2)

- To enable UPC syntax in *.c files, we will change the language mappings
- Preferences, C/C++, Language Mappings
- Click the **Add...** button to add a Language mapping.
- For Content Type, **C Source File**
- For Language, select **UPC**
- Click **OK, OK**



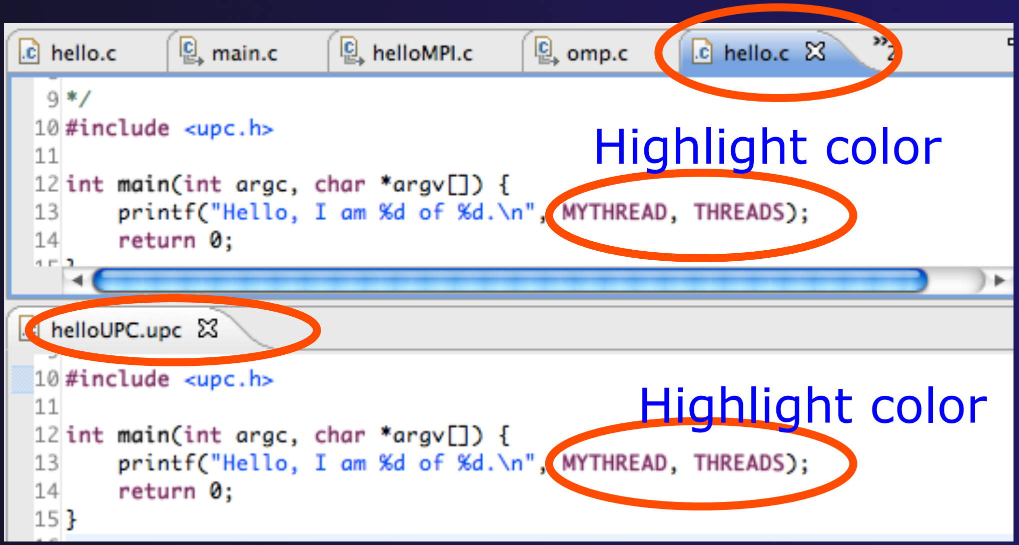
Language Mappings

These settings are global to the entire workspace. They are overridden by project-specific language mappings.

Content Type	Language
C Source File	UPC

UPC syntax in .c files (3)

- Now UPC syntax is recognized in both types of files
- You may need to close and re-open a file to see the change.
- Note: in Project Properties, you can do this for just individual projects.

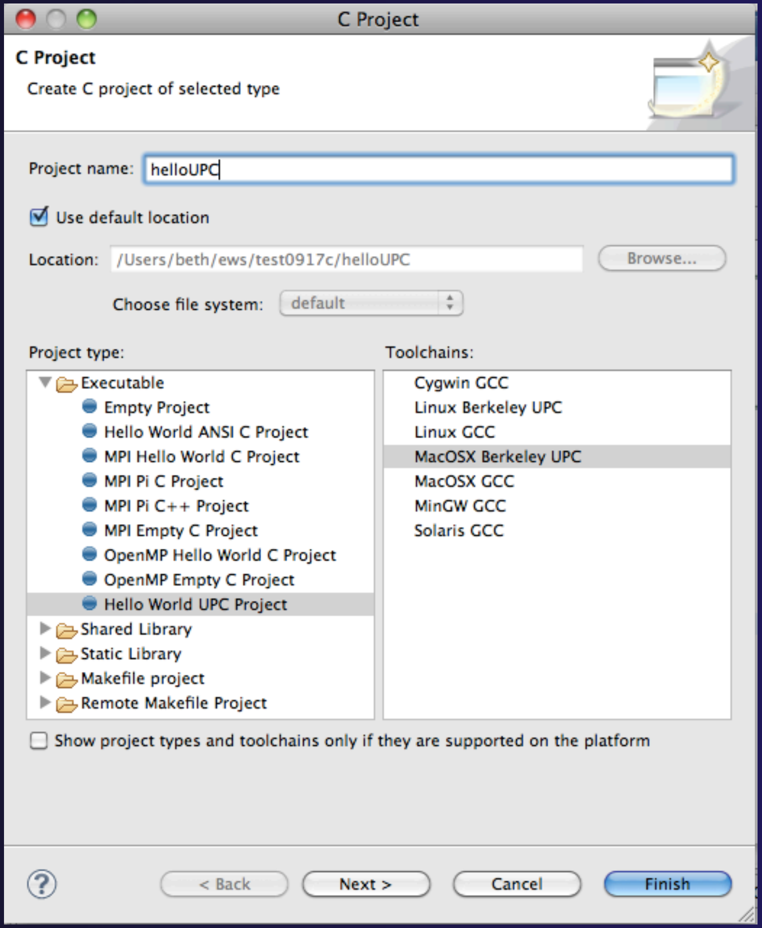


```
*/
#include <upc.h>

int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, THREADS);
    return 0;
```

```
#include <upc.h>

int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, THREADS);
    return 0;
}
```

Berkeley UPC toolchain

- Local projects only
- File > New > C project
- Hello World UPC project
- Select toolchain (if you don't have the toolchain, it just won't build.)
- Next, Next, Finish


C Project

C Project

Create C project of selected type

Project name: helloUPC

☑ Use default location

Location: /Users/beth/ews/test0917c/helloUPC Browse...

Choose file system: default

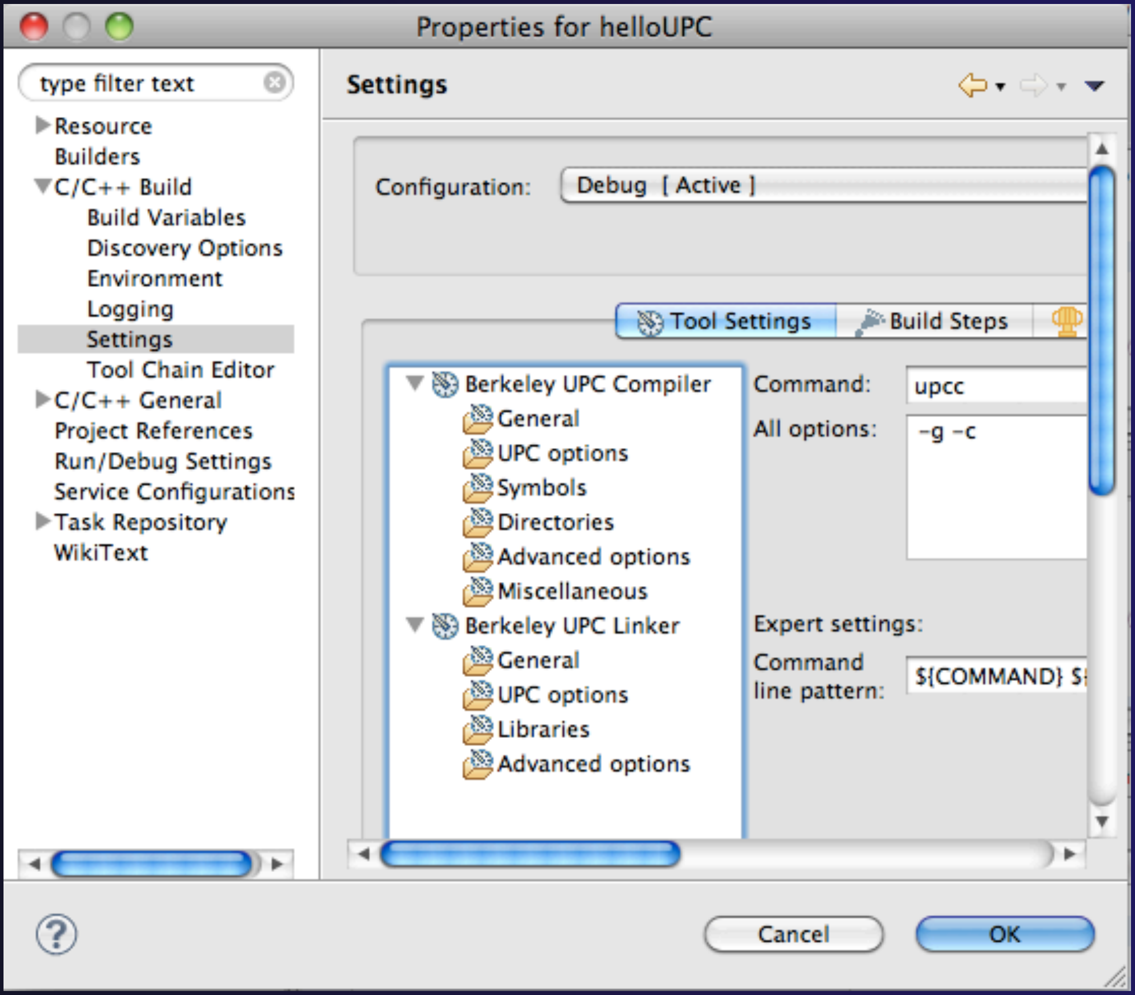
Project type:
- Executable
  - Empty Project
  - Hello World ANSI C Project
  - MPI Hello World C Project
  - MPI Pi C Project
  - MPI Pi C++ Project
  - MPI Empty C Project
  - OpenMP Hello World C Project
  - OpenMP Empty C Project
  - Hello World UPC Project
- Shared Library
- Static Library
- Makefile project
- Remote Makefile Project

Toolchains:
- Cygwin GCC
- Linux Berkeley UPC
- Linux GCC
- MacOSX Berkeley UPC
- MacOSX GCC
- MinGW GCC
- Solaris GCC

☐ Show project types and toolchains only if they are supported on the platform

BUPC toolchain

- Bring up Project Properties to see details of BUPC toolchain:
- Project, right mouse, Properties



Configuration: Debug [Active]

Command: upcc

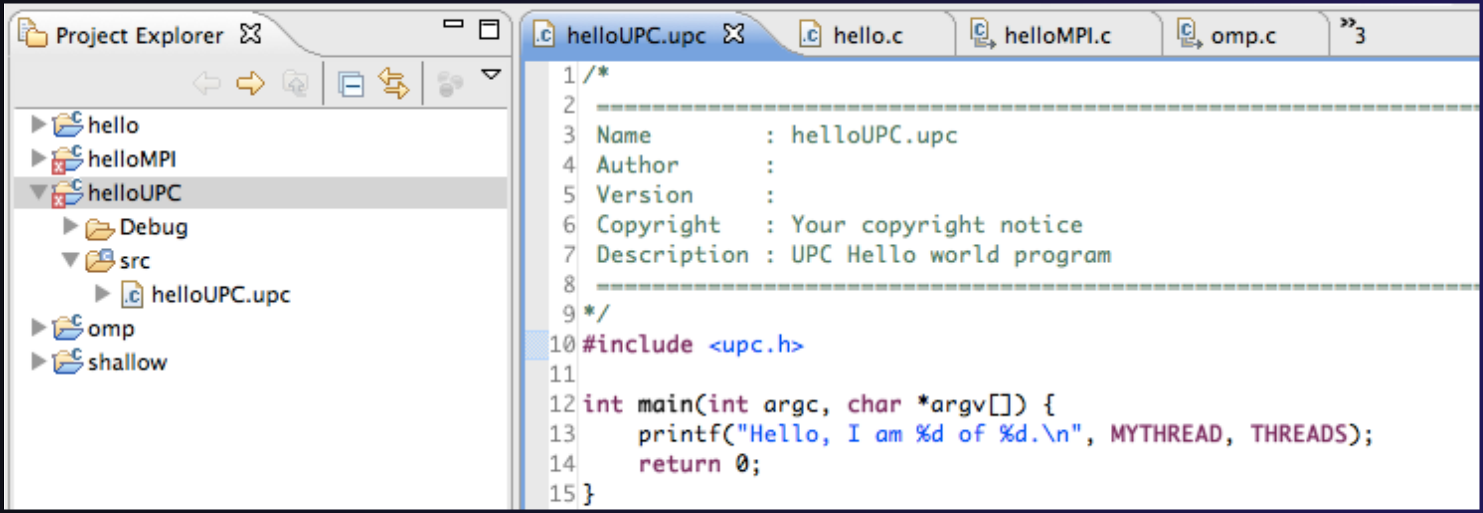
All options: -g -c

Expert settings:

Command line pattern: ${COMMAND} $

Hello World UPC project

- Hello (Berkeley) World UPC project
- Note UPC syntax highlighting
- Toolchain has been modified for UPC



```
/*
 ============================================================================
 Name        : helloUPC.upc
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : UPC Hello world program
 ============================================================================
*/
#include <upc.h>

int main(int argc, char *argv[]) {
    printf("Hello, I am %d of %d.\n", MYTHREAD, THREADS);
    return 0;
}
```

UPC on abe.ncsa.uiuc.edu

- BUPC is located at:
  - /usr/apps/mpi/upc/berkeley_upc
- To run from cmd line on abe:
  - setenv PATH /usr/apps/mpi/upc/berkeley_upc/bin:${PATH}

TO RUN FROM PTP/ECLIPSE:

- In your home dir on abe: use ‘helloUPC’ to make a remote proj
- Set Remote Paths and Symbols to include:
  - /usr/apps/mpi/upc/berkeley_upc/opt/include/upcr_preinclude
- To run: use a Generic Remote Launch for Resource Manager
- Run config:
  - Application program: /usr/apps/mpi/upc/berkeley_upc/bin/upcrun
  - Arguments tab: -q -n 4 ~/helloUPC/helloUPC

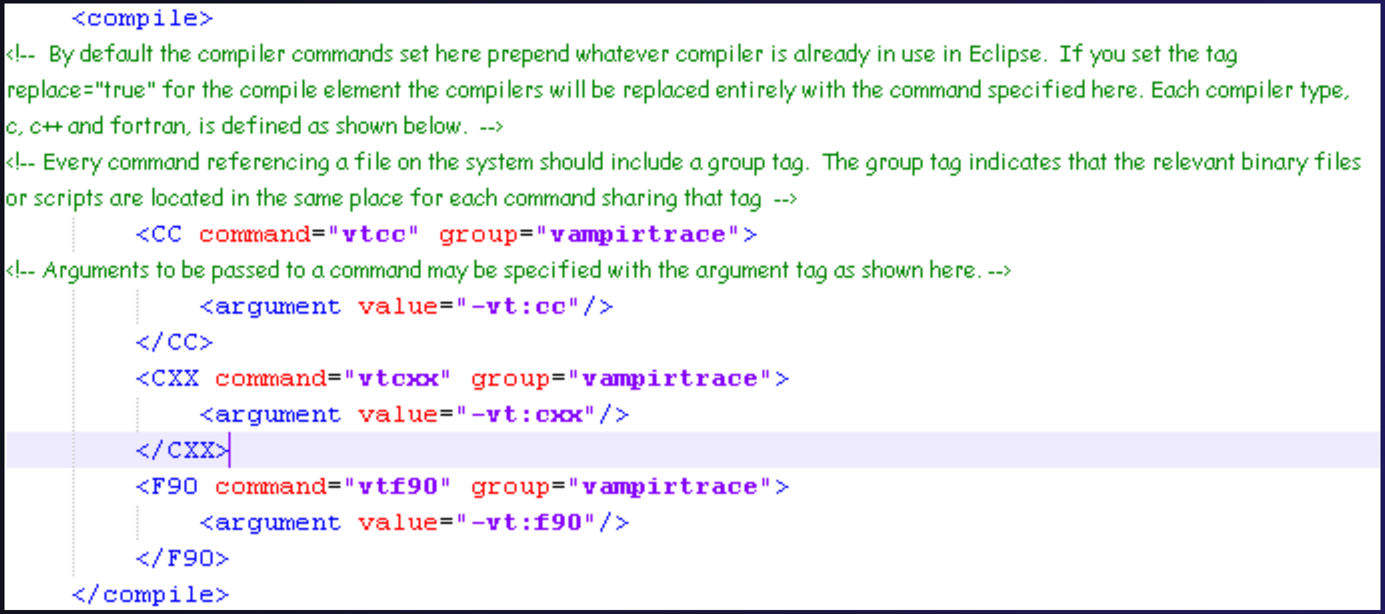
External Tools Framework ETFw Motivation

- There are numerous command-line oriented development tools employed in HPC
- These can be complicated or time consuming to use
- IDE integration for individual development tools is slow and inconsistent
- We want all our development tools in one place with one interface
- We want our development tools to work together

ETFw: Development Tool Workflows

- Variations on ‘Compile, Execute, Analyze-Results’ are common to most software development
- These steps may be tedious and time consuming, especially over multiple iterations
- By defining both tool interfaces and behavior in an XML document these steps can be simplified and automated

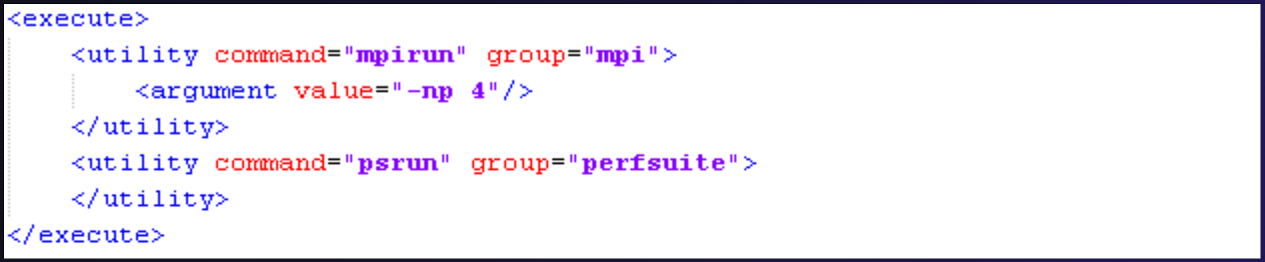
ETFw: The Build Phase



```
    <compile>
<!-- By default the compiler commands set here prepend whatever compiler is already in use in Eclipse.  If you set the tag
replace="true" for the compile element the compilers will be replaced entirely with the command specified here. Each compiler type,
c, c++ and fortran, is defined as shown below. -->
<!-- Every command referencing a file on the system should include a group tag.  The group tag indicates that the relevant binary files
or scripts are located in the same place for each command sharing that tag -->
        <CC command="vtcc" group="vampirtrace">
<!-- Arguments to be passed to a command may be specified with the argument tag as shown here. -->
            <argument value="-vt:cc"/>
        </CC>
        <CXX command="vtcxx" group="vampirtrace">
            <argument value="-vt:cxx"/>
        </CXX>
        <F90 command="vtf90" group="vampirtrace">
            <argument value="-vt:f90"/>
        </F90>
    </compile>
```

- Set compilers and arguments for each language
- Define UI for compiler/compiler-wrapper configuration

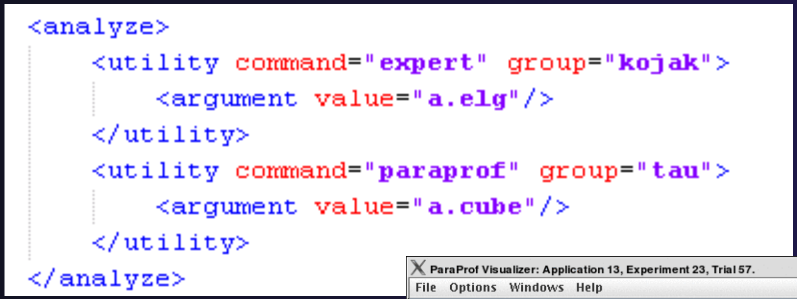
ETFw: The Execution Phase



```xml
<execute>
    <utility command="mpirun" group="mpi">
        <argument value="-np 4"/>
    </utility>
    <utility command="psrun" group="perfsuite">
    </utility>
</execute>
```

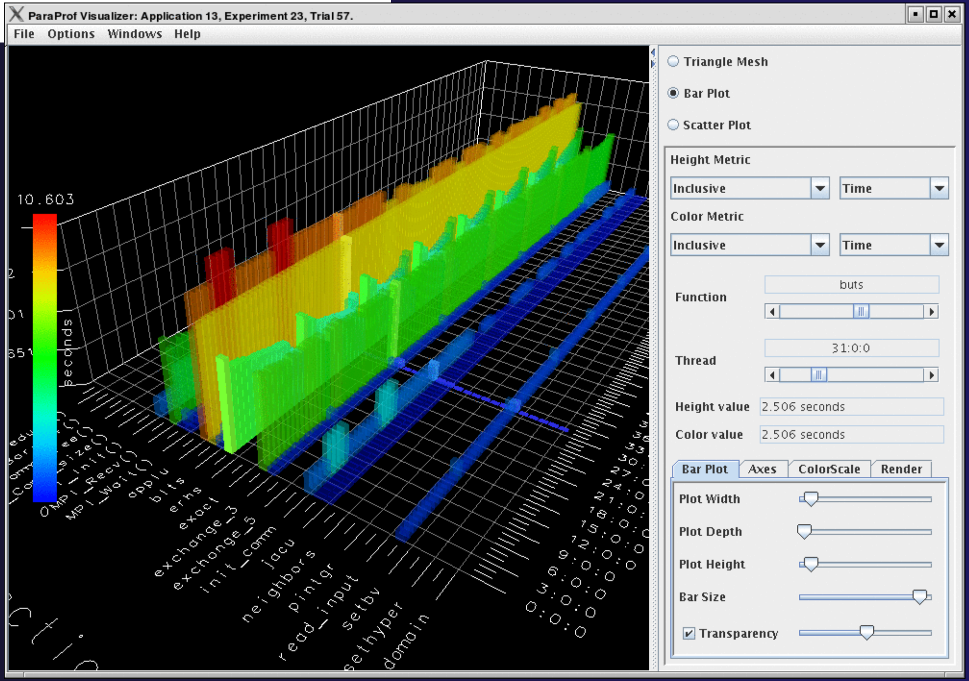
- Specify composed execution tools such as Perfsuite or Valgrind
- Set launch environment variables
- Define variables and tool options in XML or provide a UI in the IDE
- Integrates with PTP parallel launch environment

ETFw: The Analysis/Post-Processing Phase



```
<analyze>
    <utility command="expert" group="kojak">
        <argument value="a.elg"/>
    </utility>
    <utility command="paraprof" group="tau">
        <argument value="a.cube"/>
    </utility>
</analyze>
```

- Sequentially run tools on program output
- Launch external visualization tools

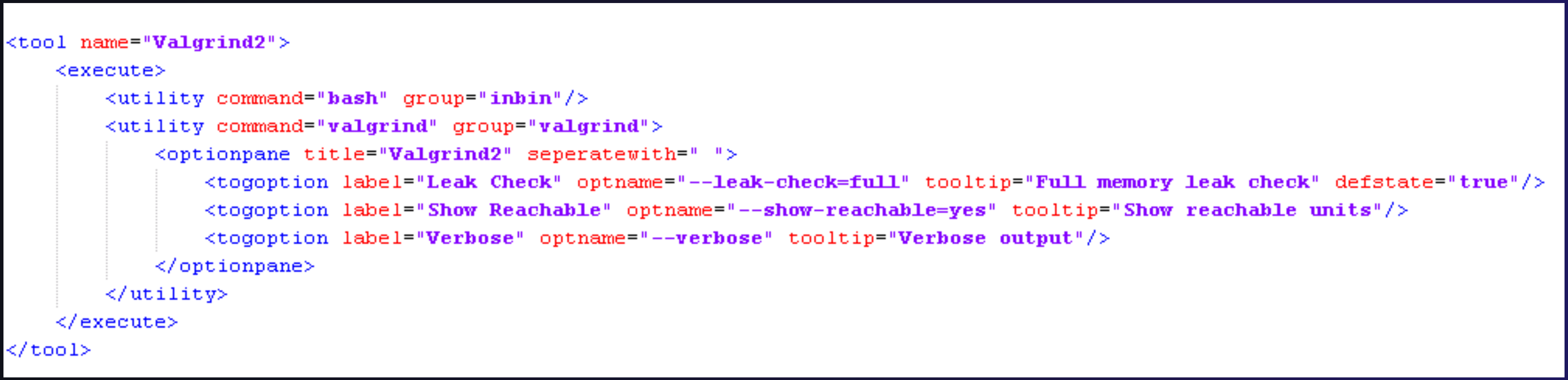


ParaProf Visualizer: Application 13, Experiment 23, Trial 57.

Triangle Mesh [✓ Bar Plot] Scatter Plot

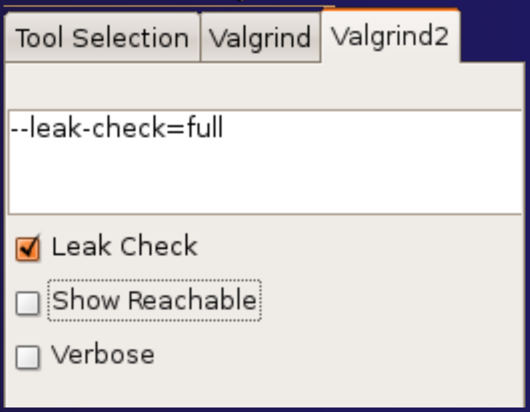
Height Metric: Inclusive, Time
Color Metric: Inclusive, Time
Function: buts
Thread: 31:0:0
Height value: 2.506 seconds
Color value: 2.506 seconds
[✓ Transparency]

ETFw: XML-Defined UI Components



```
<tool name="Valgrind2">
    <execute>
        <utility command="bash" group="inbin"/>
        <utility command="valgrind" group="valgrind">
            <optionpane title="Valgrind2" seperatewith=" ">
                <togoption label="Leak Check" optname="--leak-check=full" tooltip="Full memory leak check" defstate="true"/>
                <togoption label="Show Reachable" optname="--show-reachable=yes" tooltip="Show reachable units"/>
                <togoption label="Verbose" optname="--verbose" tooltip="Verbose output"/>
            </optionpane>
        </utility>
    </execute>
</tool>
```

- Each pane constructs a set of options sent to a tool or a set of environment variables
- Numerous options for converting a command line interface into an intelligent GUI without Eclipse coding

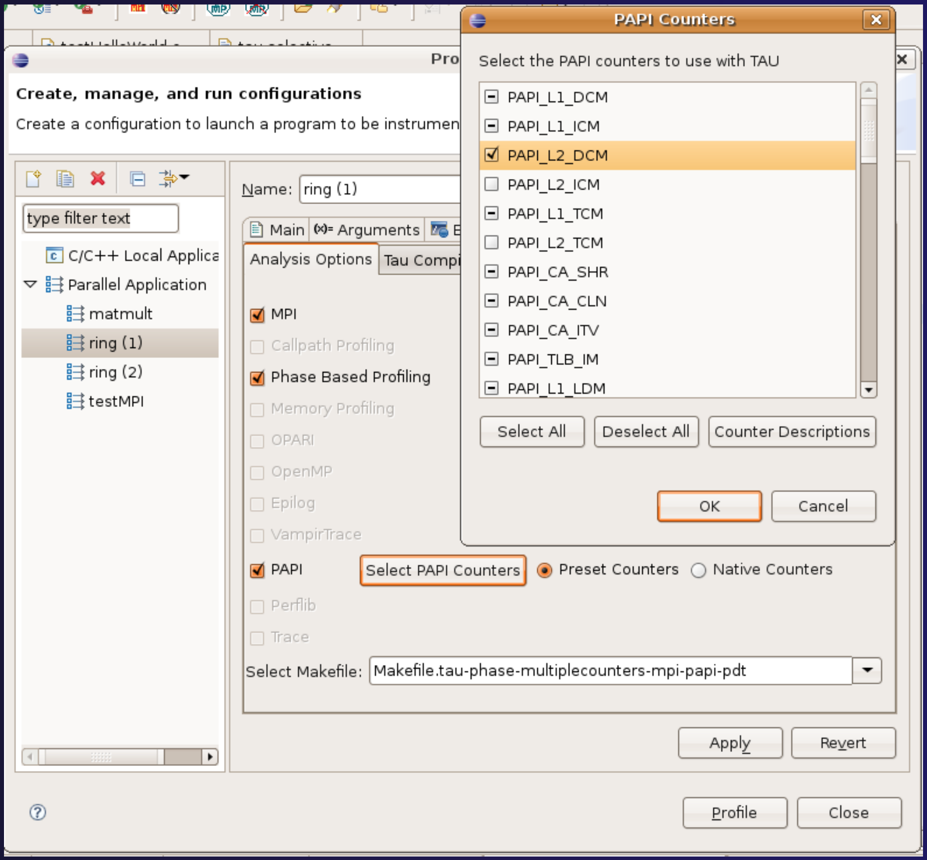


```
--leak-check=full
```

- [✓] Leak Check
- [ ] Show Reachable
- [ ] Verbose

ETFw: Advanced Components

- Extension points allow integration with UIs and workflow behavior too complex to define in XML
- Logical and iterative workflows for successive executions and parametric studies



Create, manage, and run configurations

Create a configuration to launch a program to be instrumen

type filter text

- C/C++ Local Applica
- Parallel Application
  - matmult
  - ring (1)
  - ring (2)
  - testMPI

Name: ring (1)

Main | Arguments | Analysis Options | Tau Compi

- [✓] MPI
- [ ] Callpath Profiling
- [✓] Phase Based Profiling
- [ ] Memory Profiling
- [ ] OPARI
- [ ] OpenMP
- [ ] Epilog
- [ ] VampirTrace
- [✓] PAPI — Select PAPI Counters — [✓ Preset Counters] Native Counters
- [ ] Perflib
- [ ] Trace

Select Makefile: Makefile.tau-phase-multiplecounters-mpi-papi-pdt

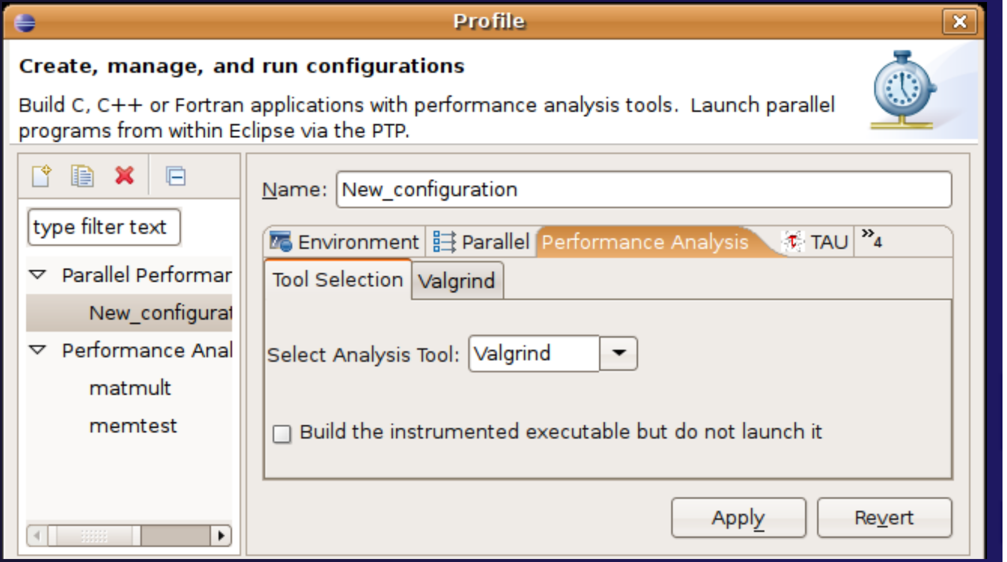
PAPI Counters

Select the PAPI counters to use with TAU

- PAPI_L1_DCM
- PAPI_L1_ICM
- [✓] PAPI_L2_DCM
- [ ] PAPI_L2_ICM
- PAPI_L1_TCM
- [ ] PAPI_L2_TCM
- PAPI_CA_SHR
- PAPI_CA_CLN
- PAPI_CA_ITV
- PAPI_TLB_IM
- PAPI_L1_LDM

ETFw: Using Workflows

- New workflows are added to the ETFw launch configuration system
- Multiple workflow configurations can be defined and saved for different use cases
- XML Workflow definitions can be saved and reused in different environments


Profile

Create, manage, and run configurations

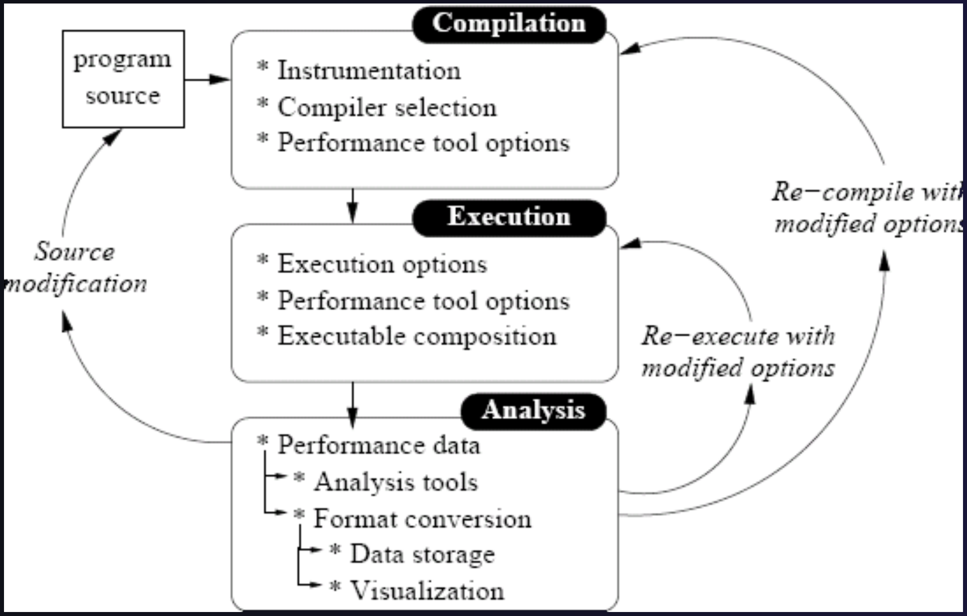
Build C, C++ or Fortran applications with performance analysis tools. Launch parallel programs from within Eclipse via the PTP.

Name: New_configuration

Select Analysis Tool: Valgrind

☐ Build the instrumented executable but do not launch it

ETFw: General Purpose Workflow



- Automated
- Generalized
- Quick performance analysis and other development tool integration
- Exposes tool capabilities to the user

ETFw: Continuing Development

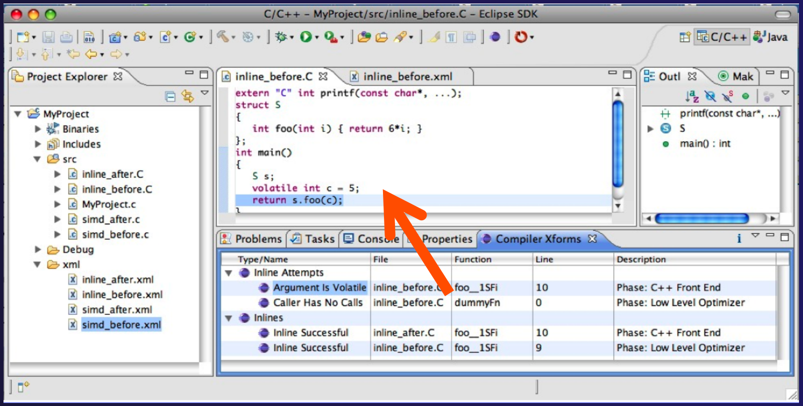
Plans:

- Integration with PTP Remote Development Tools
- Additional options for GUI definition
- Generalization of TAU specific features such as hardware counter selection and performance data storage

- Contact: Wyatt Spear

ETFw Feedback view

- Many existing tools provide information that can be mapped to source code lines
  - Compiler errors, warnings, suggestions
  - Performance tool findings
- ETFw feedback view provided to aid construction of these views
  - Currently geared toward data provided by tools in XML files
- Original ETFw facilities aid the CALL of external tools from PTP
  - Feedback view aids the exposition of results to the user

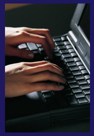


```
extern "C" int printf(const char*, ...);
struct S
{
   int foo(int i) { return 6*i; }
};
int main()
{
   S s;
   volatile int c = 5;
   return s.foo(c);
```

Type/Name	File	Function	Line	Description
Inline Attempts				
Argument Is Volatile	inline_before.C	foo__1SFi	10	Phase: C++ Front End
Caller Has No Calls	inline_before.C	dummyFn	0	Phase: Low Level Optimizer
Inlines				
Inline Successful	inline_after.C	foo__1SFi	10	Phase: C++ Front End
Inline Successful	inline_before.C	foo__1SFi	9	Phase: Low Level Optimizer

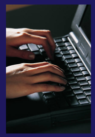
Examples:

- Compiler optimization report
- Performance tool data
- Refactoring tool uses “advice” from external files



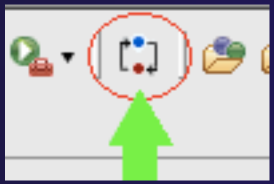
Feedback Sample

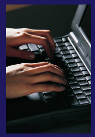
- Download a sample implementation of the feedback view:
- Complete instructions here: http://wiki.eclipse.org/PTP/ETFw/feedback
- And on following slide…



Feedback Sample – (1) Install

- Download the plugin jar file
- http://download.eclipse.org/tools/ptp/misc/feedback/org.eclipse.ptp.etfw.feedback.sample_1.0.0.201010280927.jar
- Save it in your eclipse/dropins directory
  - This is a "quick and dirty" type of installation
  - Eclipse knows to look here when it starts, and it installs whatever it finds here
- Then restart eclipse
  - You should see the feedback icon





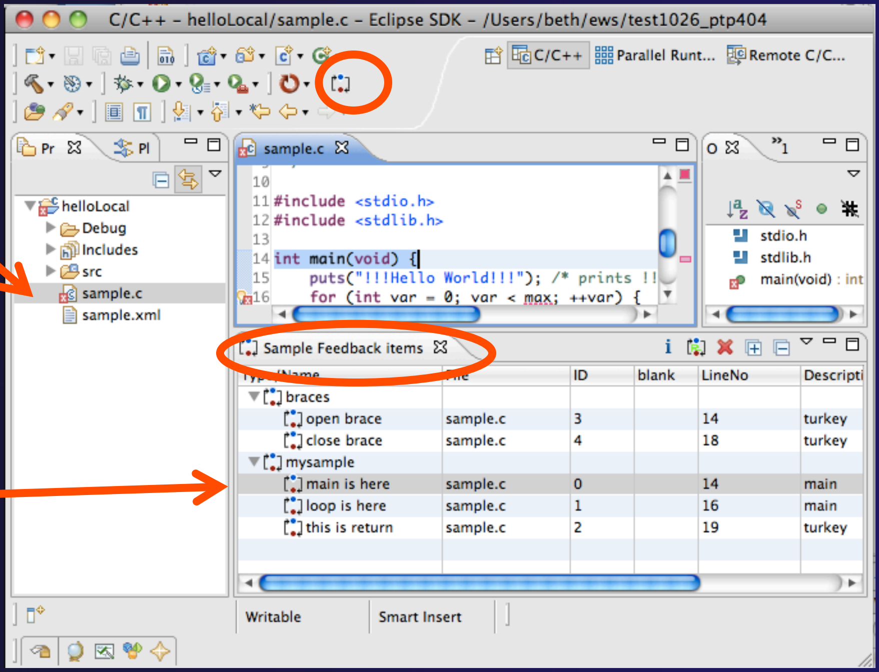
Feedback Sample – (2) data files

- You have the Feedback sample plug-in installed
- Now you need some sample files for it to process
  - sample.c and sample.xml
  - They are hidden in the plug-in!
  - Let's take it apart to find them
  - Unzip the jar file; they are in the data/ directory
    - Alternate instructions on the wiki page
  - Put them in a (local) eclipse project

Feedback Sample – (3) Try it

- ✦ You have the Feedback sample plug-in installed
- ✦ You have an xml file that it can parse, and the source file that it refers to.

1. Select xml file
2. Click feedback button
3. See Sample Feedback view
4. Double-click in view to navigate to source code lines



C/C++ – helloLocal/sample.c – Eclipse SDK – /Users/beth/ews/test1026_ptp404

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    puts("!!!Hello World!!!"); /* prints !!
    for (int var = 0; var < max; ++var) {
```

Sample Feedback items

Type/Name	File	ID	blank	LineNo	Descripti
braces					
open brace	sample.c	3		14	turkey
close brace	sample.c	4		18	turkey
mysample					
main is here	sample.c	0		14	main
loop is here	sample.c	1		16	main
this is return	sample.c	2		19	turkey

END

Module 8: Other Tools and Wrap-up

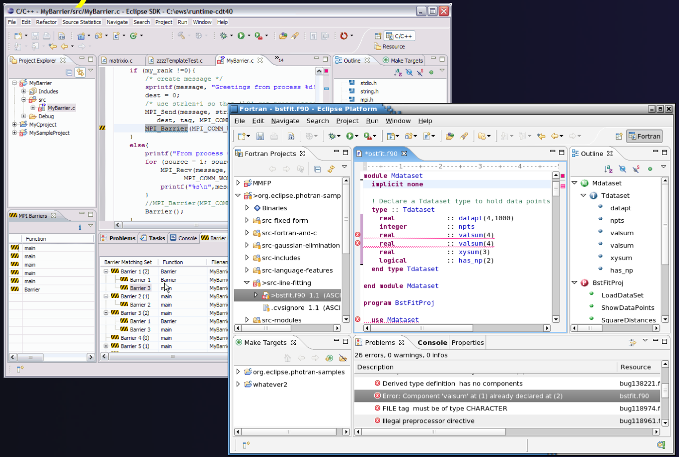
- Objective
  - How to find more information on PTP
  - Learn about other tools related to PTP
  - See PTP upcoming features
- Contents
  - Links to other tools, including performance tools
  - Planned features for new versions of PTP
  - Additional documentation
  - How to get involved

NCSA HPC Workbench

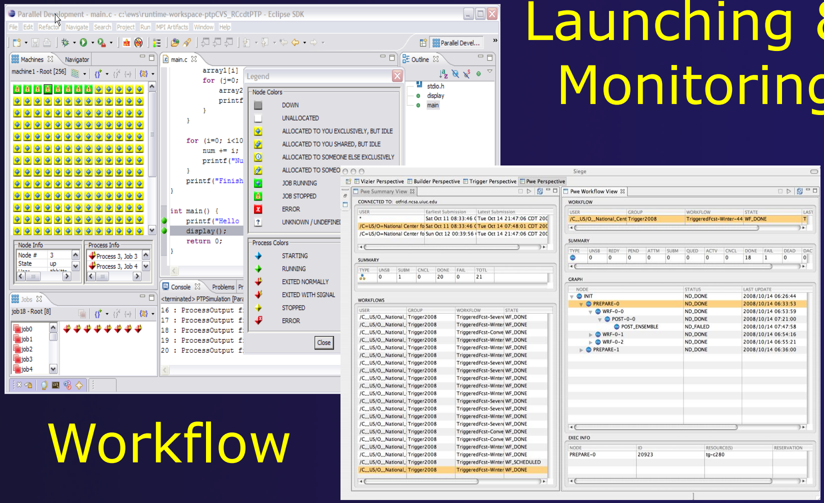
- Tools for NCSA Blue Waters
  - http://www.ncsa.illinois.edu/BlueWaters/
  - Sustained Petaflop system
- Based on Eclipse and PTP
- Includes some related tools
  - Performance tools
  - Scalable debugger
  - Workflow tools (https://wiki.ncsa.uiuc.edu/display/MRDPUB/MRD+Public+Space+Home+Page)
- Part of the enhanced computational environment described at:
  http://www.ncsa.illinois.edu/BlueWaters/ece.html

NCSA HPC Workbench

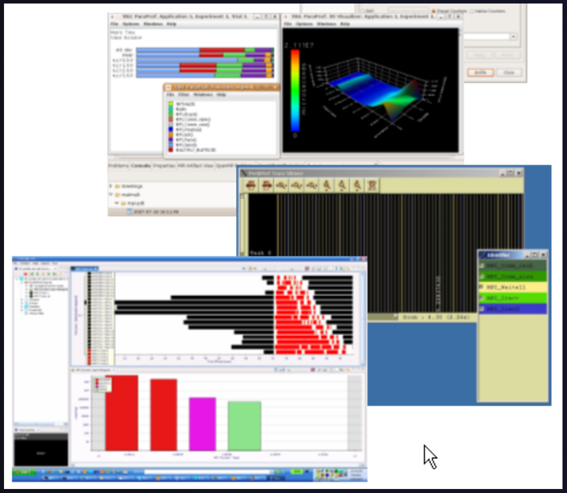
Coding & Analysis (CDT, PLDT, Photran)



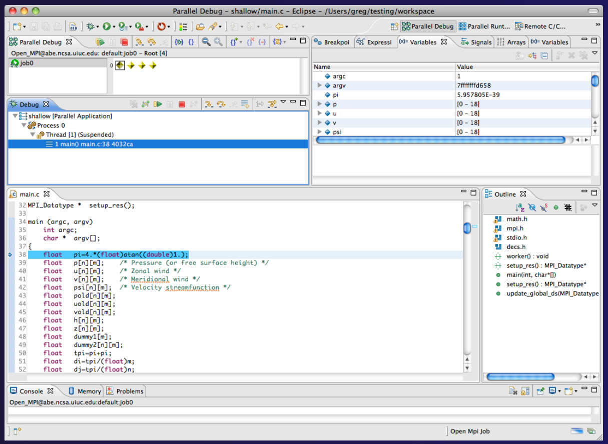
PTP Launching & Monitoring



Workflow



Performance Tuning (HPC toolkit, HPCS toolkit, RENCI, …)



Scalable Debugger

Planned PTP Future Work

- Scalability improvements
  - UI to support 1M processes
  - Optimized communication protocol
  - Very large application support
- Resource Managers
  - More implementations of configurable resource managers
- Synchronized project improvements
  - Conversion wizard
  - Resolving merge conflicts
- Enhancements to the debugger
  - Stability enhancements
  - Transition to Scalable Communication Infrastructure (SCI)

Useful Eclipse Tools

- Linux Tools (autotools, valgrind, Oprofile, Gprof)
  - http://eclipse.org/linuxtools
- Python
  - http://pydev.org
- Ruby
  - http://www.aptana.com/products/radrails
- Perl
  - http://www.epic-ide.org
- Git
  - http://www.eclipse.org/egit
- VI bindings
  - Vrapper (open source) - http://vrapper.sourceforge.net
  - viPlugin (commercial) - http://www.viplugin.com

Online Information

- Information about PTP
  - Main web site for downloads, documentation, etc.
    - http://eclipse.org/ptp
  - Developers' (*and users*) wiki for designs, planning, meetings, etc.
    - http://wiki.eclipse.org/PTP
  - Articles and other documents
    - http://wiki.eclipse.org/PTP/articles
- Information about Photran
  - Main web site for downloads, documentation, etc.
    - http://eclipse.org/photran
  - User's manuals
    - http://wiki.eclipse.org/PTP/photran/documentation

Mailing Lists

- PTP Mailing lists
  - Major announcements (new releases, etc.) - low volume
    - http://dev.eclipse.org/mailman/listinfo/ptp-announce
  - User discussion and queries - medium volume
    - http://dev.eclipse.org/mailman/listinfo/ptp-user
  - Developer discussions - high volume
    - http://dev.eclipse.org/mailman/listinfo/ptp-dev
- Photran Mailing lists
  - User discussion and queries
    - http://dev.eclipse.org/mailman/listinfo/photran
  - Developer discussions –
    - http://dev.eclipse.org/mailman/listinfo/photran-dev

Getting Involved

- See http://eclipse.org/ptp
- Read the developer documentation on the wiki
- Join the mailing lists
- Attend the monthly developer meetings
  - Teleconference Monthly
  - Each second Tuesday, 1:00 pm ET
  - Details on the PTP wiki
- Attend the montly user meetings
  - Teleconference Monthly
  - Each 4th Wednesday, 2:00 pm ET

PTP will only succeed with your participation!

PTP Tutorial Feedback

- Please complete feedback form
- Your feedback is valuable!

Thanks for attending
We hope you found it useful