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

Beth Tibbitts, IBM
tibbitts@us.ibm.com

Jay Alameda, NCSA
jalameda@ncsa.uiuc.edu

Jeff Overbey, NCSA
jeffreyoverbey@acm.org

Wyatt Spear, U. Oregon
wspear@cs.uoregon.edu

Galen Arnold, NCSA
arnoldg@ncsa.uiuc.edu

Alan Humphrey, SCI
ahumphrey@sci.utah.edu

July 16, 2012

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 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

Tutorial Outline

Time (Tentative!)	Module	Topics	Presenter
8:00-8:30	1. Eclipse Installation 2. Introduction/Overview	✦ Installation of Eclipse and PTP ✦ Eclipse architecture & organization overview	Beth, Jeff
8:30-10:00	3. Eclipse basics (1.5 hr))	✦ Eclipse basics; Creating a new project from CVS; Local, remote, and synchronized projects ✦ Creating a synchronized project from CVS ✦ Creating a sync project directly from source code ✦ Editor features; MPI Features	Beth Beth Jeff Beth
10:00-10:15	BREAK		
10:30-12:00	4. Build/run (1.5 hr)	✦ Building w/Makefile ✦ Target configurations and launching a parallel app ✦ Including Modules (Build Environment Mgmt)	Jay
12:00 – 1:00	Lunch		
1:00-3:00	4. Debugging (1 hr) 5. Fortran (30 min) 6. Adv/Misc (30 min)	✦ Debugging an MPI program ✦ Fortran ✦ Adv features - including XSEDE feature	Beth Jeff Jeff
3:00-3:15	BREAK		
3:15-4:45	5. Performance Tuning & Analysis Tools	✦ TAU, External Tools FrameWork (:40) ✦ Gprof/Gcov (:40) ✦ GEM (:10)	Wyatt Galen Alan
4:45-5:00	6. Other Tools, Wrapup	✦ PTP Future plans, Other Tools, website, mailing lists, getting involved, Tutorial feedback	Beth

Final Slides, Installation Instructions

- Please go to http://wiki.eclipse.org/PTP/tutorials/XSEDE12 for slides and installation instructions

Installation

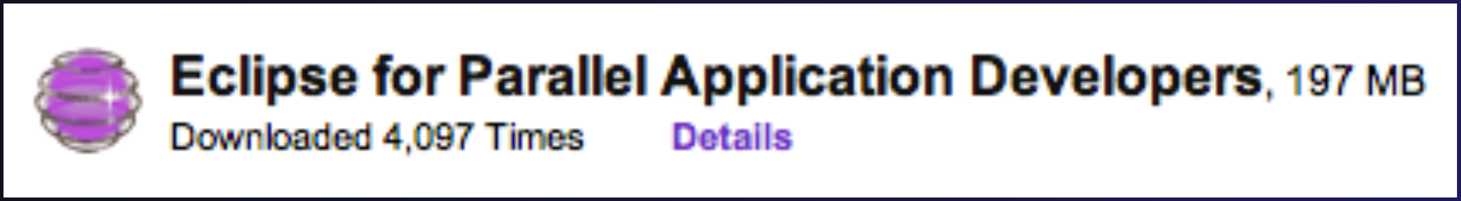
- Objective
  - To learn how to install Eclipse and PTP
- Contents
  - System Prerequisites
  - Eclipse Download and Installation of “Eclipse for Parallel Application Developers”
  - Installation Confirmation
  - Updating the PTP within your Eclipse to the latest release

System Prerequisites

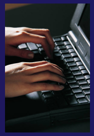
- Local system (running Eclipse)
  - Linux (just about any version)
  - MacOSX (10.5 Leopard or higher)
  - Windows (XP on)
- Java: Eclipse requires Sun or IBM Java
  - Only need Java runtime environment (JRE)
  - Java 1.6 or higher
    - Java 1.6 is the same as JRE 6.0
  - The GNU Java Compiler (GCJ), which comes standard on Linux, will not work!
  - OpenJDK, distributed with some Linux distributions, has not been tested by us but should work.
  - See http://wiki.eclipse.org/PTP/installjava

Eclipse Packages

- The current version of Eclipse (4.2) is also known as "Juno"
- Eclipse is available in a number of different packages for different kinds of development
  - http://eclipse.org/downloads
- For PTP, we recommend the all-in-one download:
  - Eclipse for Parallel Application Developers



Eclipse for Parallel Application Developers, 197 MB
Downloaded 4,097 Times Details

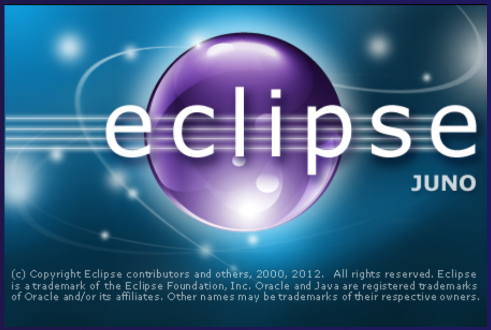


Exercise

- Download the "Eclipse for Parallel Application Developers" package to your laptop
  - Your tutorial instructions will provide the location of the package
  - Make sure you match the architecture with that of your laptop
- If your machine is Linux or Mac OS X, untar the file
  - On Mac OS X you can just double-click in the Finder
- If your machine is Windows, unzip the file
- This creates an **eclipse** folder containing the executable as well as other support files and folders

Starting Eclipse

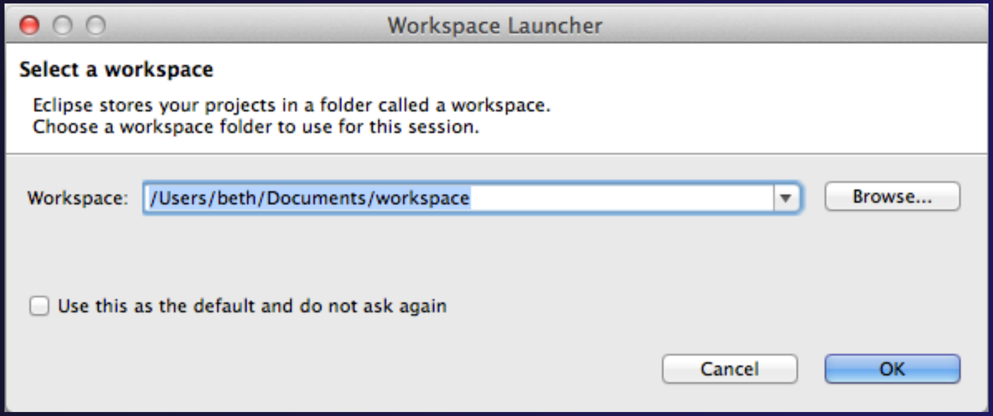
- **Linux**
  - From a terminal window, enter "<eclipse_installation_path>/eclipse/eclipse &"
- **Mac OS X**
  - From finder, open the **eclipse** folder where you installed
  - Double-click on the **Eclipse** application
  - Or from a terminal window
- **Windows**
  - Open the **eclipse** folder
  - Double-click on the **eclipse** executable


eclipse
JUNO
(c) Copyright Eclipse contributors and others, 2000, 2012. All rights reserved. Eclipse is a trademark of the Eclipse Foundation, Inc. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Specifying A Workspace

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

The prompt can be turned off



Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

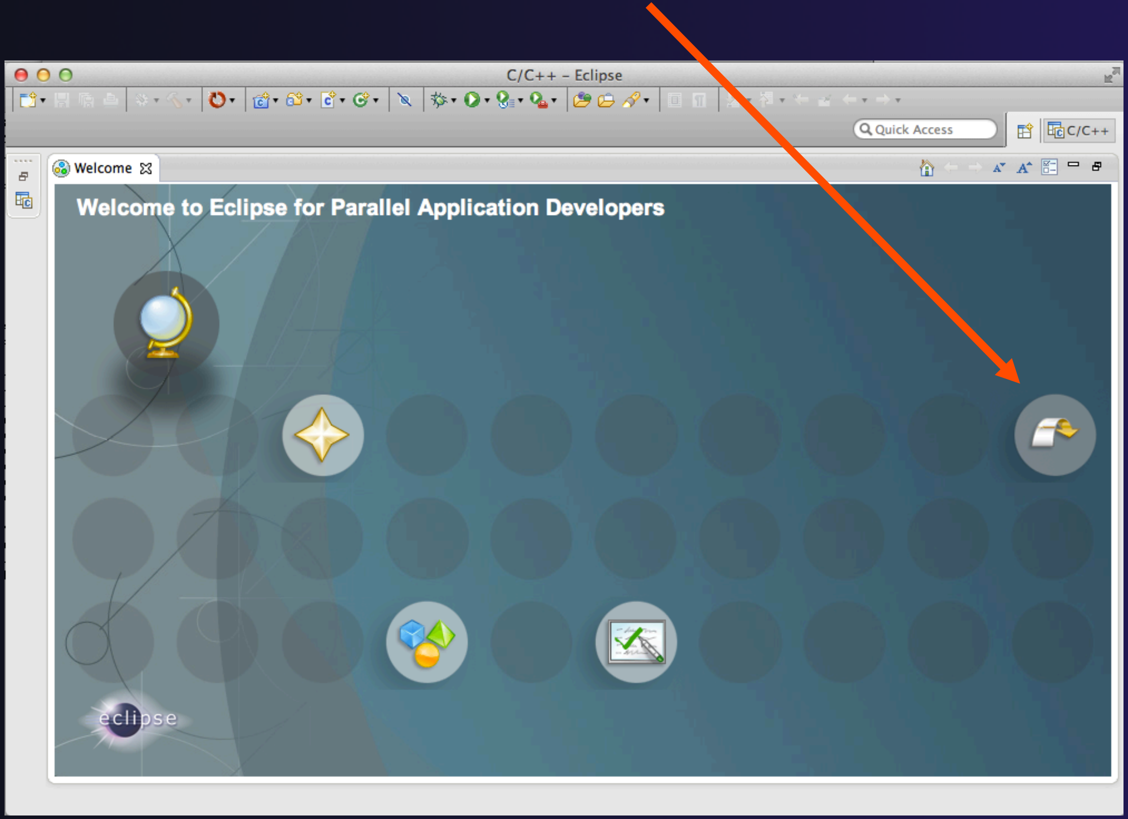
Workspace: /Users/beth/Documents/workspace

☐ Use this as the default and do not ask again

Eclipse Welcome Page

- Displayed when Eclipse is run for the first time

Select "Go to the workbench"

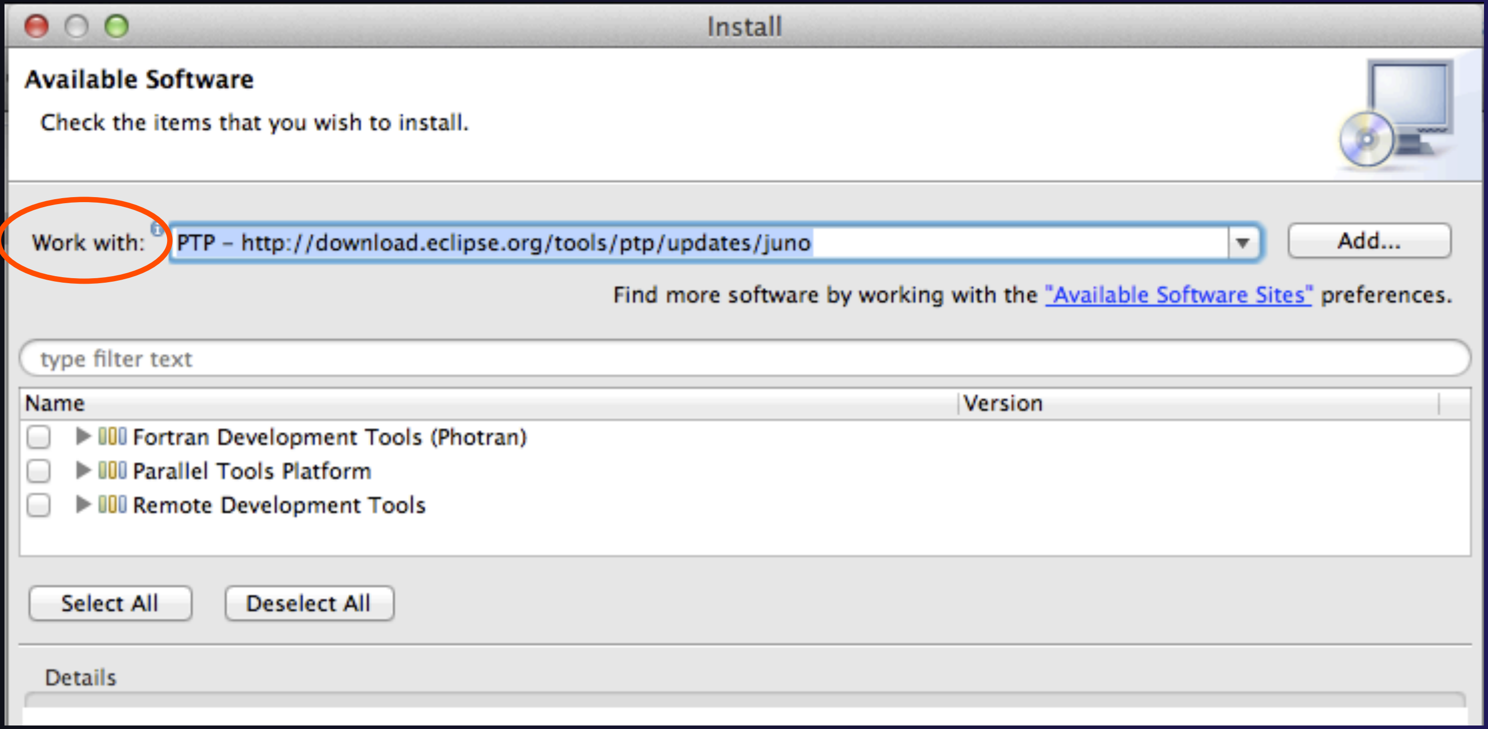


Checking for PTP Updates

- From time-to-time there may be newer PTP releases than the Indigo release
  - Juno and "Parallel package" updates are released only in Sept and February
- PTP maintains its own update site with the most recent release
  - Bug fix releases can be more frequent than Indigo's and what is within the parallel package
- You must enable (and install from) the PTP-specific update site before the updates will be found

Updating PTP

- Now select **Help>Install New Software…**
  - In the **Work With:** dropdown box, select the PTP update site you confirmed already:



Install

Available Software

Check the items that you wish to install.

Work with: PTP – http://download.eclipse.org/tools/ptp/updates/juno

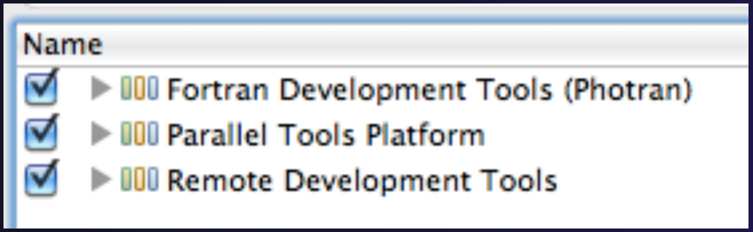
Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
Fortran Development Tools (Photran)	
Parallel Tools Platform	
Remote Development Tools	

Updating PTP (2)

- Easiest option is to check everything - which updates existing features and adds a few more

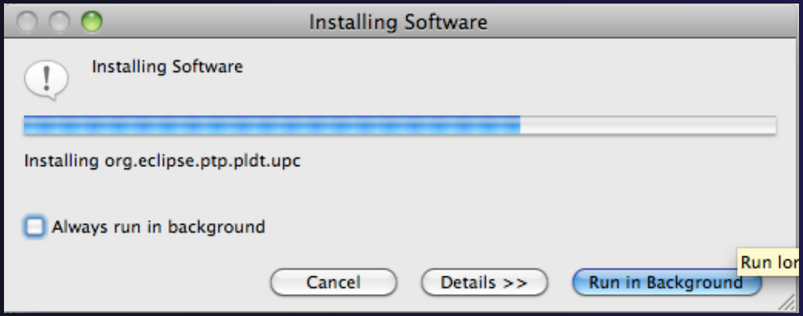


Name
- ✓ Fortran Development Tools (Photran)
- ✓ Parallel Tools Platform
- ✓ Remote Development Tools

- Select **Next** to continue updating PTP
- Select **Next** to confirm features to install

Updating PTP (3)

- Accept the License agreement and select **Finish**



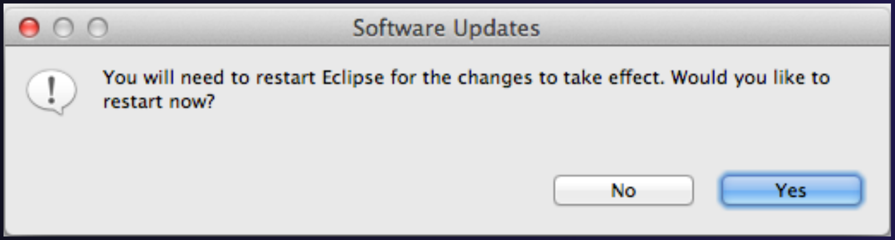
Installing Software

Installing Software

Installing org.eclipse.ptp.pldt.upc

Always run in background

- Select **Yes** when prompted to restart Eclipse

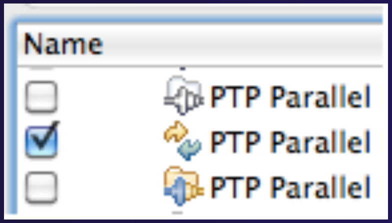


Software Updates

You will need to restart Eclipse for the changes to take effect. Would you like to restart now?

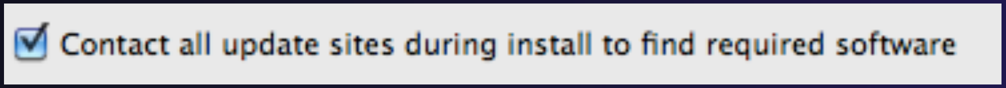
Updating Individual Features

- It’s also possible to update features without adding any new features
  - Open each feature and check the ones you want to update
  - Icons indicate: Grey plug: already installed
    Double arrow: can be updated
    Color plug: Not installed yet



Name
☐ PTP Parallel
☑ PTP Parallel
☐ PTP Parallel

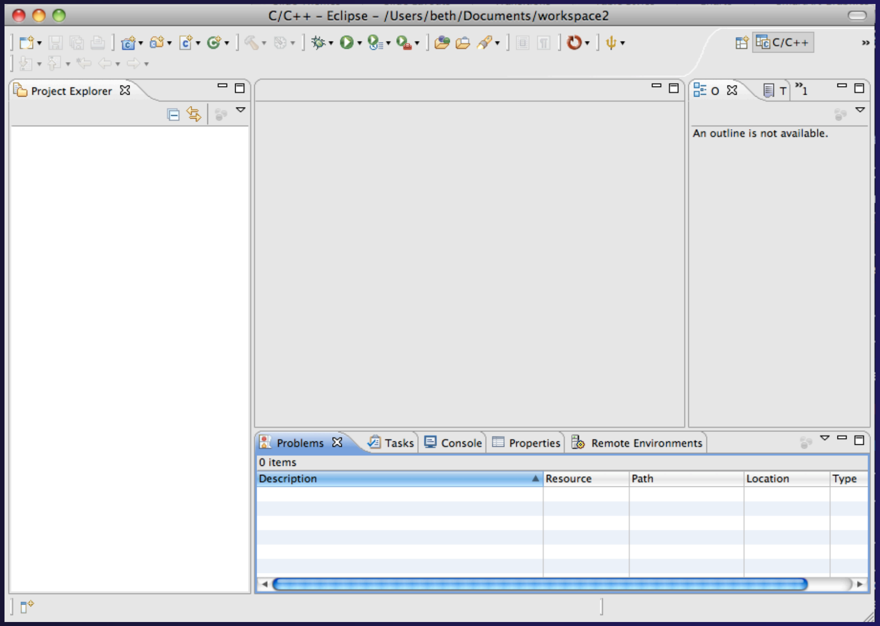
- Note: if network is slow, consider unchecking



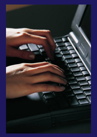
☑ Contact all update sites during install to find required software

Restart after Install

- If any new top-level features are installed, they will be shown on the welcome screen
- We only updated PTP, so we land back at C/C++ Perspective



- **Help>About** or **Eclipse > About Eclipse** … will indicate the release of PTP installed
- Further **Help>Check for Updates** will find future updates on the PTP Update site



Exercise

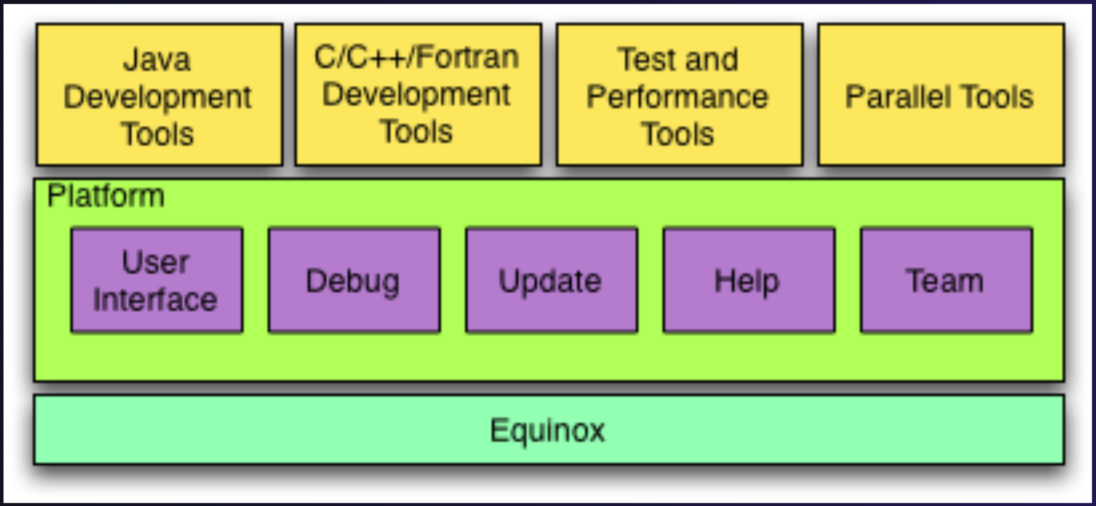
1. Launch Eclipse and select the default workspace
2. Configure Eclipse to check for PTP updates
3. Update all PTP features to the latest level
4. Restart Eclipse once the installation is completed

Introduction

- Objective
  - To introduce the Eclipse platform and PTP
- Contents
  - New and Improved Features
  - 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

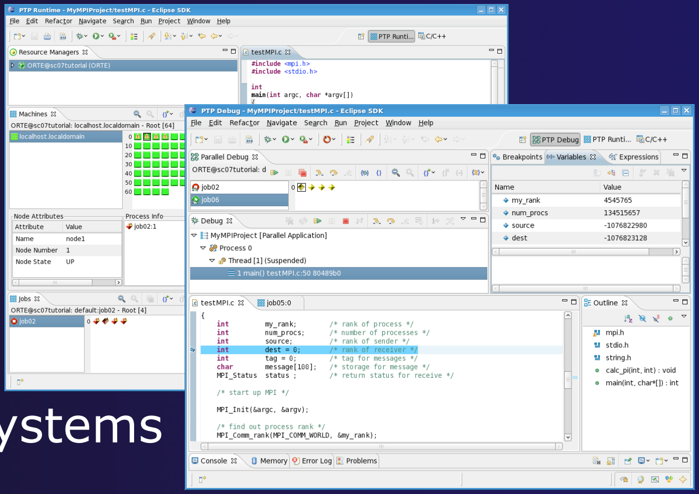


Eclipse Features

- Full development lifecycle support
- Revision control integration (CVS, SVN, Git)
- Project dependency management
- Incremental building
- Content assistance
- Context sensitive help
- Language sensitive searching
- Multi-language support
- Debugging

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[])
```

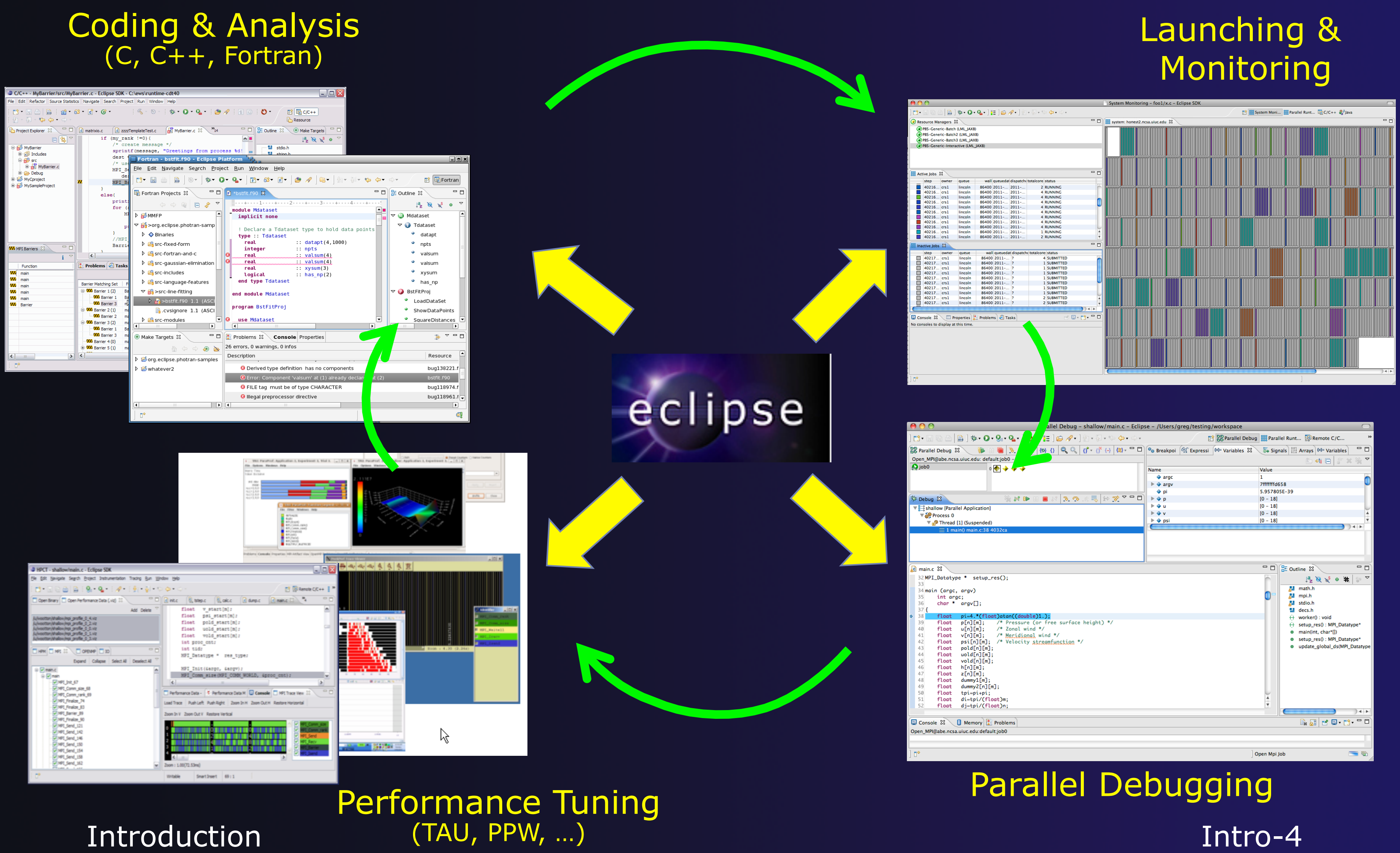
Attribute	Value
Name	node1
Node Number	1
Node State	UP

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

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

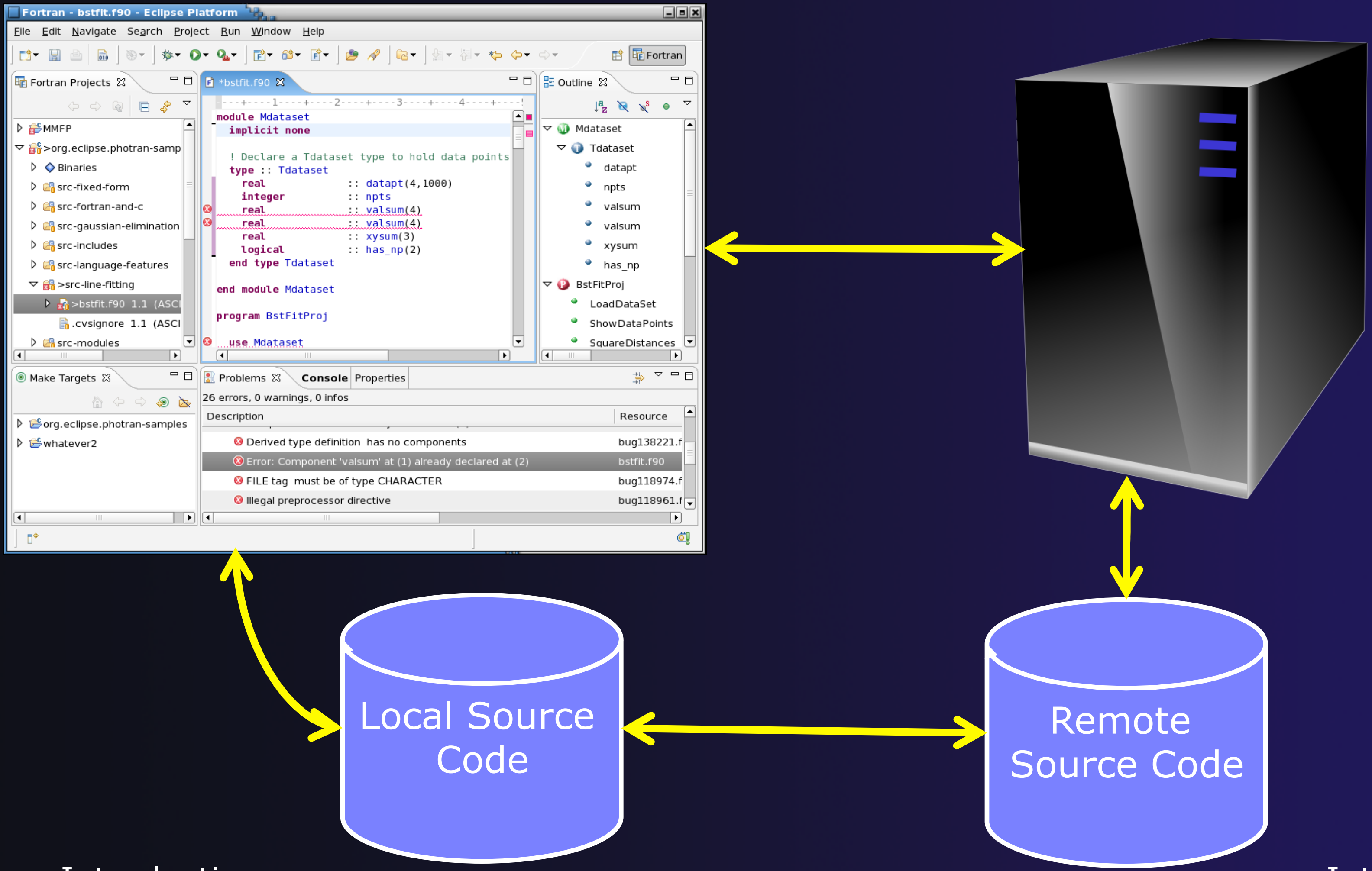
```
{
    int         my_rank;        /* rank of process */
    int         num_procs;      /* number of processes */
    int         source;         /* rank of sender */
    int         dest = 0;       /* rank of receiver */
    int         tag = 0;        /* tag for messages */
    char        message[100];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

Eclipse PTP Family of Tools



How Eclipse is Used

Editing/Compiling



```
*bstfit.f90
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
```

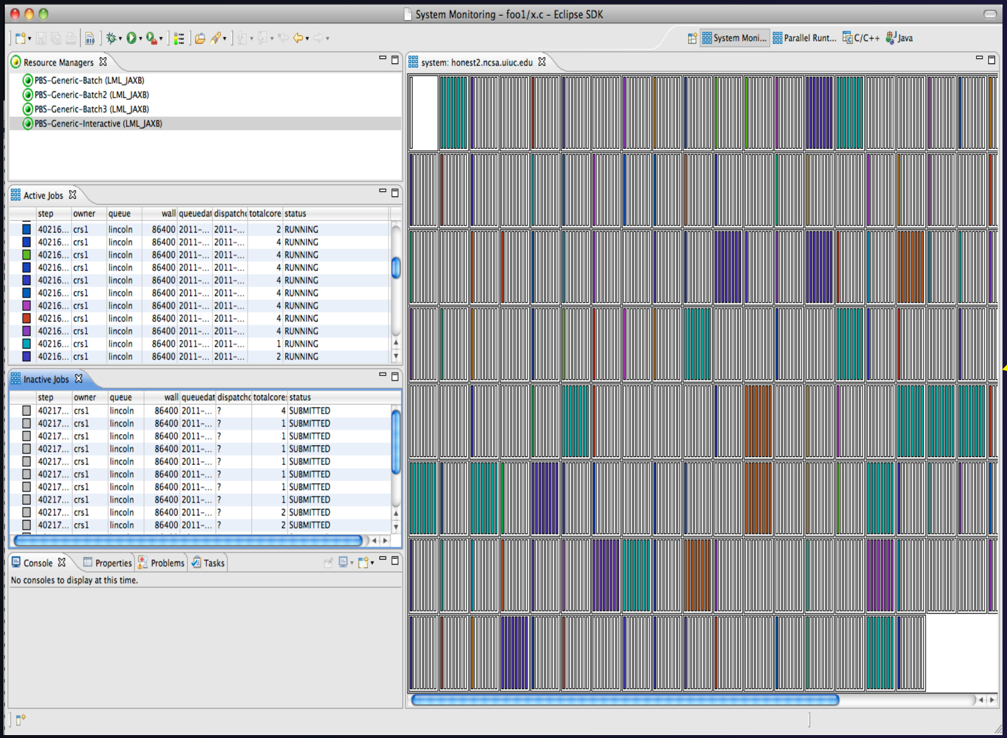
Problems | Console | Properties

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

How Eclipse is Used

Launching/Monitoring



System Monitoring – foo1/x.c – Eclipse SDK

Resource Managers

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

Active Jobs

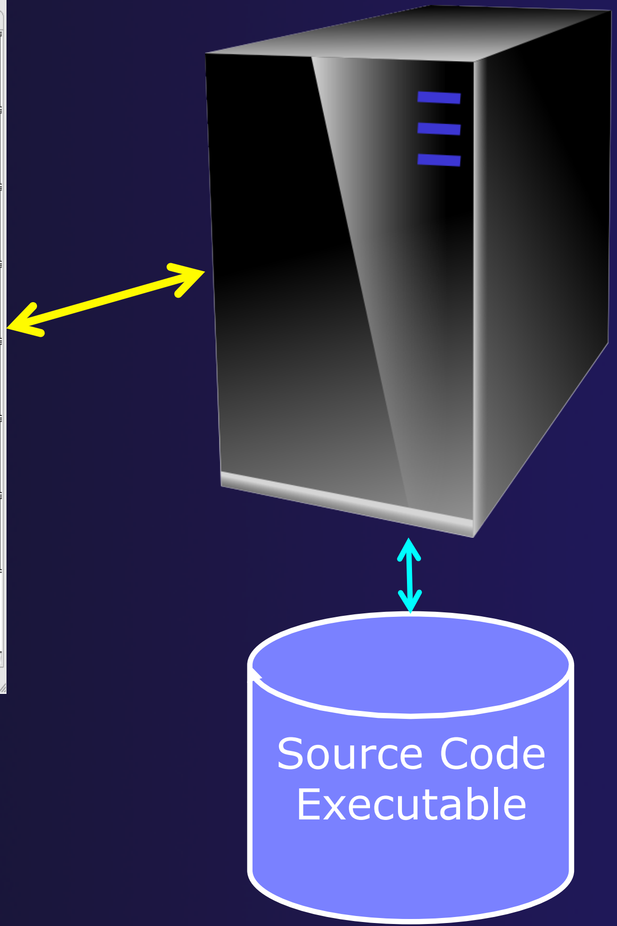
	step	owner	queue	wall	queuedat	dispatchc	totalcore	status
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	2	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟩	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	crs1	lincoln	86400	2011-...	2011-...	4	RUNNING
🟦	40216...	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	dispatchc	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

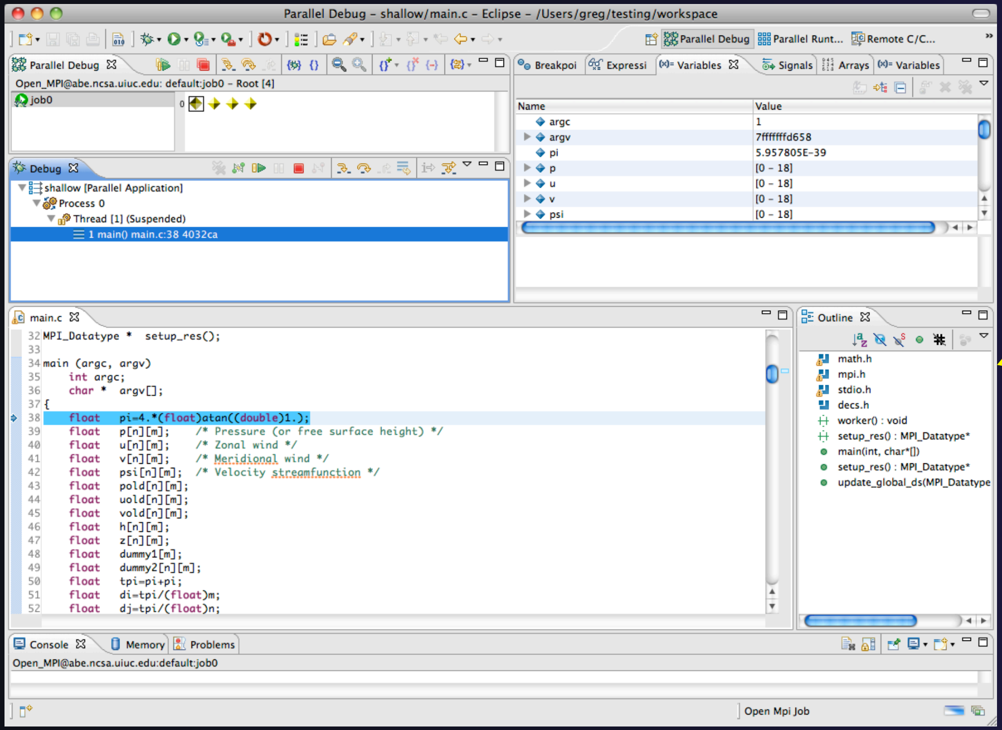
Console | Properties | Problems | Tasks

No consoles to display at this time.



How Eclipse is Used

Debugging



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

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

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

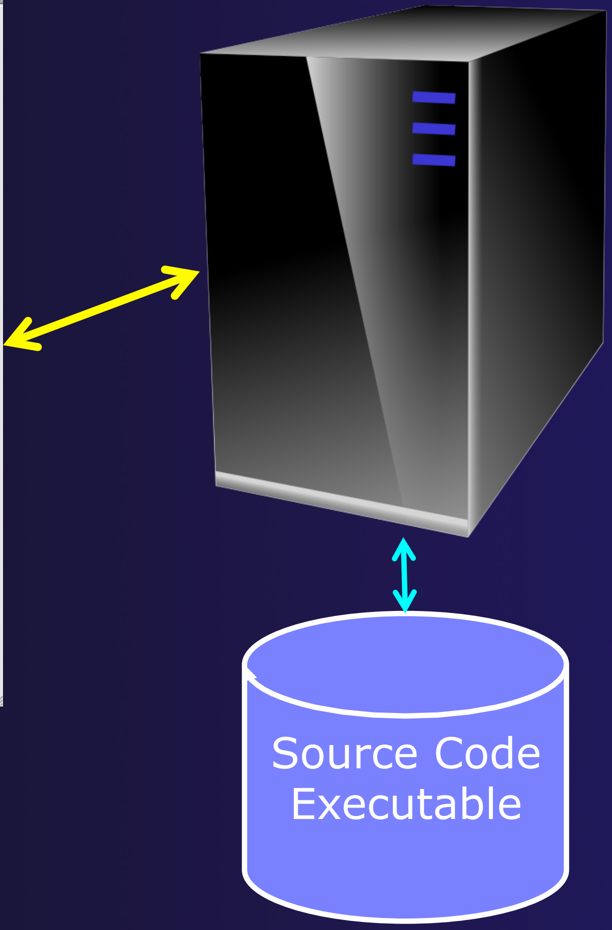
Name	Value
argc	1
argv	7fffffffd658
pi	5.957805E-39
p	[0 – 18]
u	[0 – 18]
v	[0 – 18]
psi	[0 – 18]

```c
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;
```

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

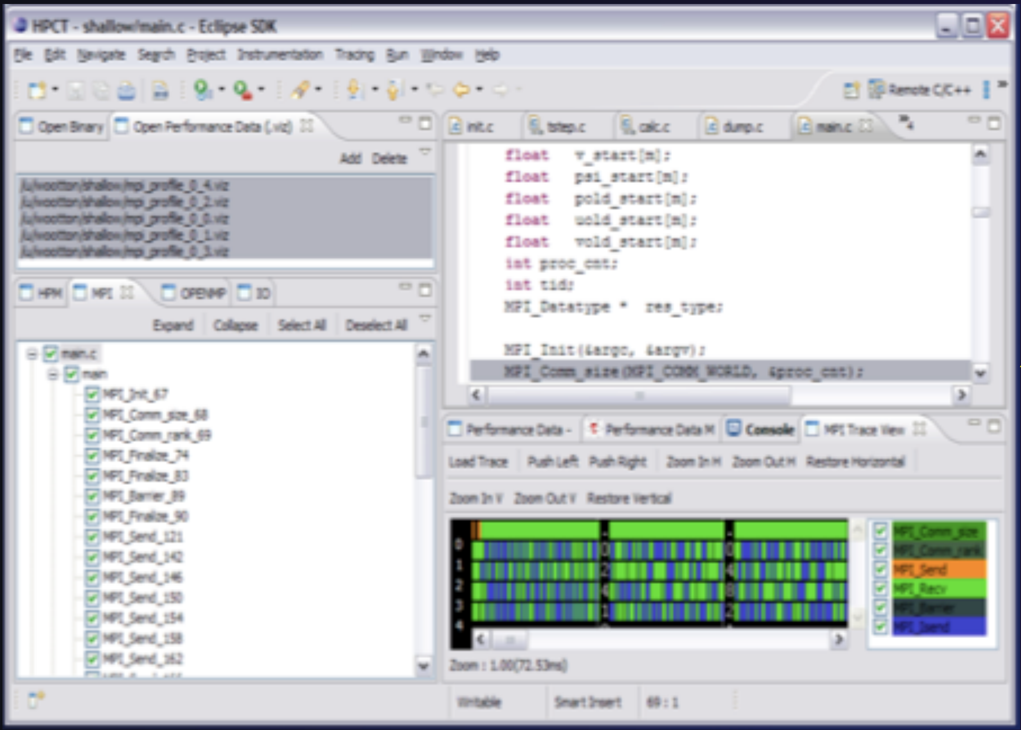
Open_MPI@abe.ncsa.uiuc.edu:default:job0

Open Mpi Job



How Eclipse is Used

Performance Tuning


HPCT - shallow/main.c - Eclipse SDK
File Edit Navigate Search Project Instrumentation Tracing Run Window Help
Remote C/C++
Open Binary
Open Performance Data (.viz)
Add Delete
HPM MPI OPENMP IO
Expand Collapse Select All Deselect All
init.c tstep.c calc.c dump.c main.c
Performance Data - Performance Data M Console MPI Trace View
Load Trace Push Left Push Right Zoom In H Zoom Out H Restore Horizontal
Zoom In V Zoom Out V Restore Vertical
Writable Smart Insert

```
/u/wootton/shallow/mpi_profile_0_4.viz
/u/wootton/shallow/mpi_profile_0_2.viz
/u/wootton/shallow/mpi_profile_0_0.viz
/u/wootton/shallow/mpi_profile_0_1.viz
/u/wootton/shallow/mpi_profile_0_3.viz
```

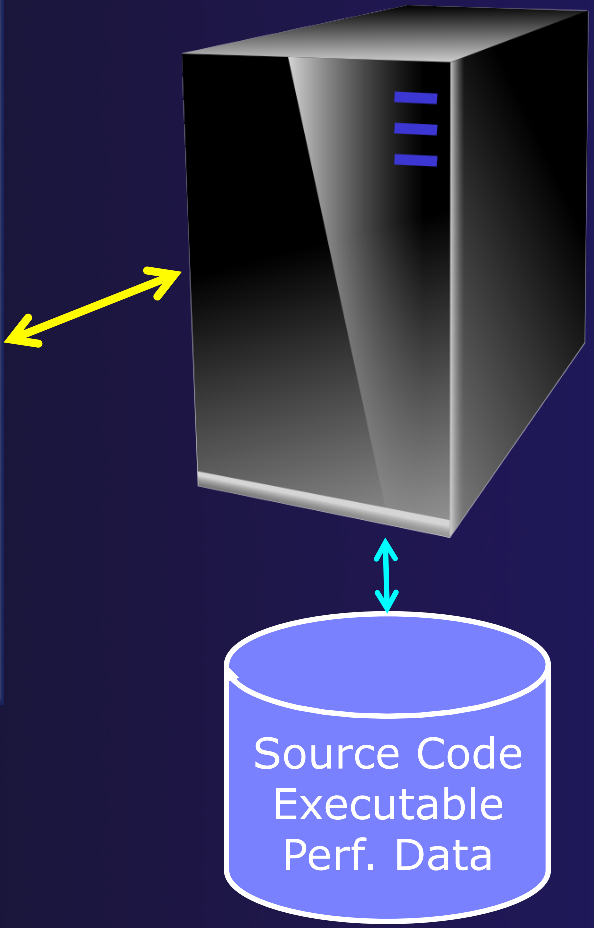
```
main.c
  main
    MPI_Init_67
    MPI_Comm_size_68
    MPI_Comm_rank_69
    MPI_Finalize_74
    MPI_Finalize_83
    MPI_Barrier_89
    MPI_Finalize_90
    MPI_Send_121
    MPI_Send_142
    MPI_Send_146
    MPI_Send_150
    MPI_Send_154
    MPI_Send_158
    MPI_Send_162
```

```
    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);
```

✓ MPI_Comm_size
✓ MPI_Comm_rank
✓ MPI_Send
✓ MPI_Recv
✓ MPI_Barrier
✓ MPI_Isend

Zoom : 1.00(72.53ms)

69 : 1

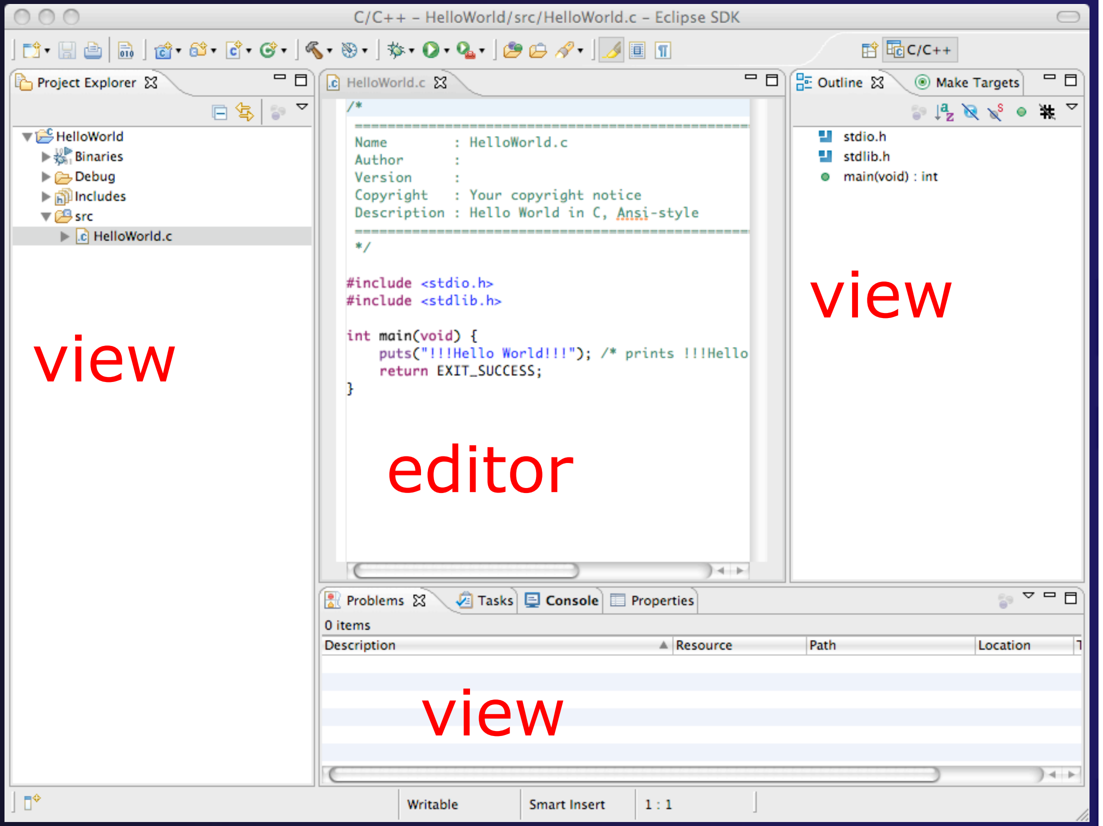


Eclipse Basics

- Objective
  - Learn about basic Eclipse workbench concepts: projects,
  - Learn about projects: local, synchronized, remote
- Contents
  - Workbench components: Perspectives, Views, Editors
  - Local, remote, and synchronized projects
  - Learn how to create and manage a C project
  - Learn about Eclipse editing features

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

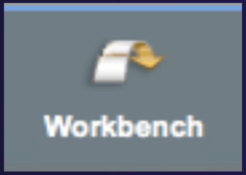


```
/*
 ============================================================
 Name        : HelloWorld.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================
 */

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

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

perspective

Perspectives

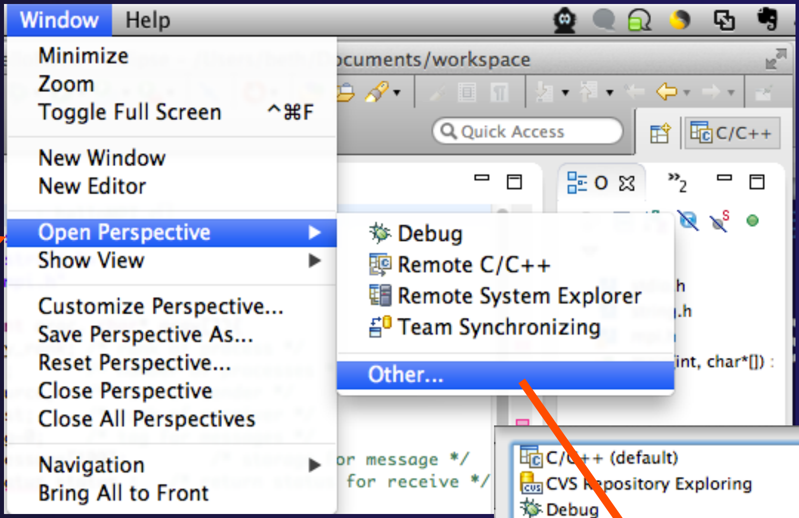
- Perspectives define the layout of views and editors in the workbench
- They are *task oriented*, i.e. they contain specific views for doing certain tasks:
  - **C/C++ Perspective** for manipulating compiled code
  - **Debug Perspective** for debugging applications
  - **System Monitoring Perspective**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

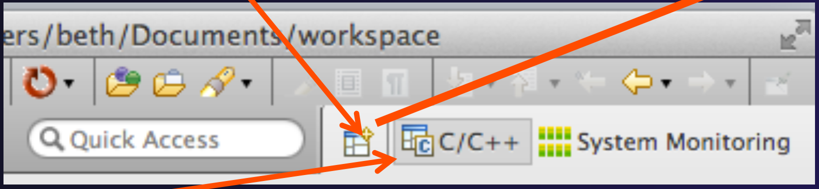


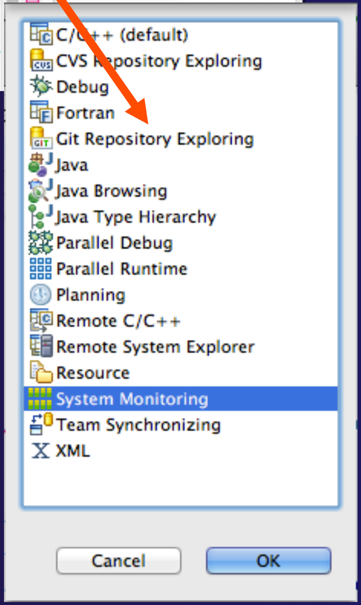
Switching Perspectives

- Three ways of changing perspectives

  1. Choose the **Window>Open Perspective** menu option Then choose **Other…**
  2. Click on the **Open Perspective** button in the upper right corner of screen (hover over it to see names)
  3. Click on a perspective shortcut button

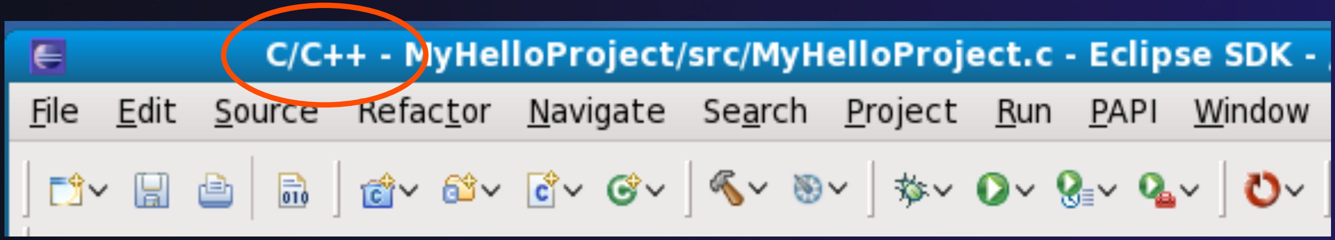




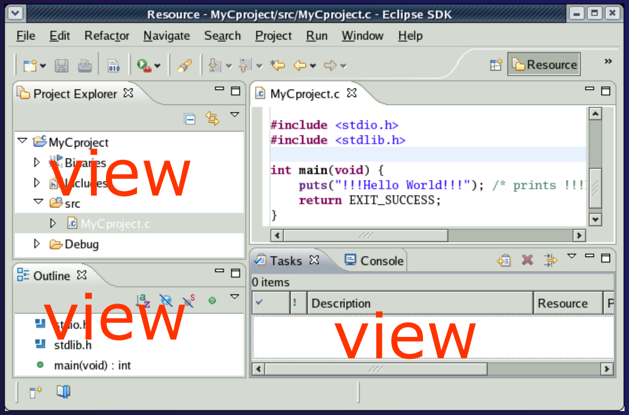


Which Perspective?

- The current perspective is displayed in the title bar

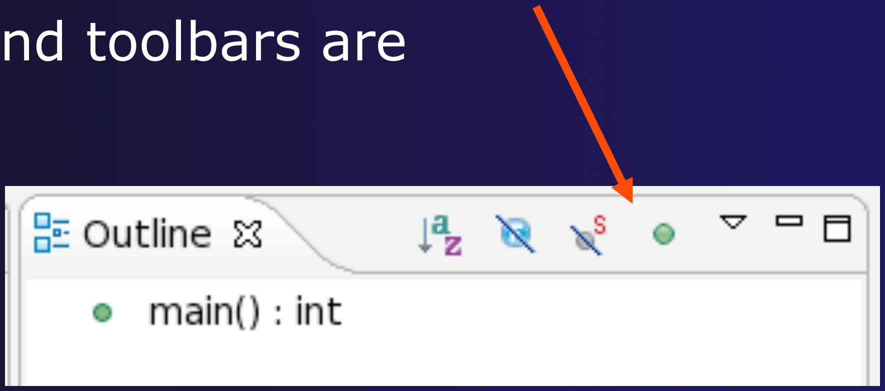

C/C++ - MyHelloProject/src/MyHelloProject.c - Eclipse SDK -

Views



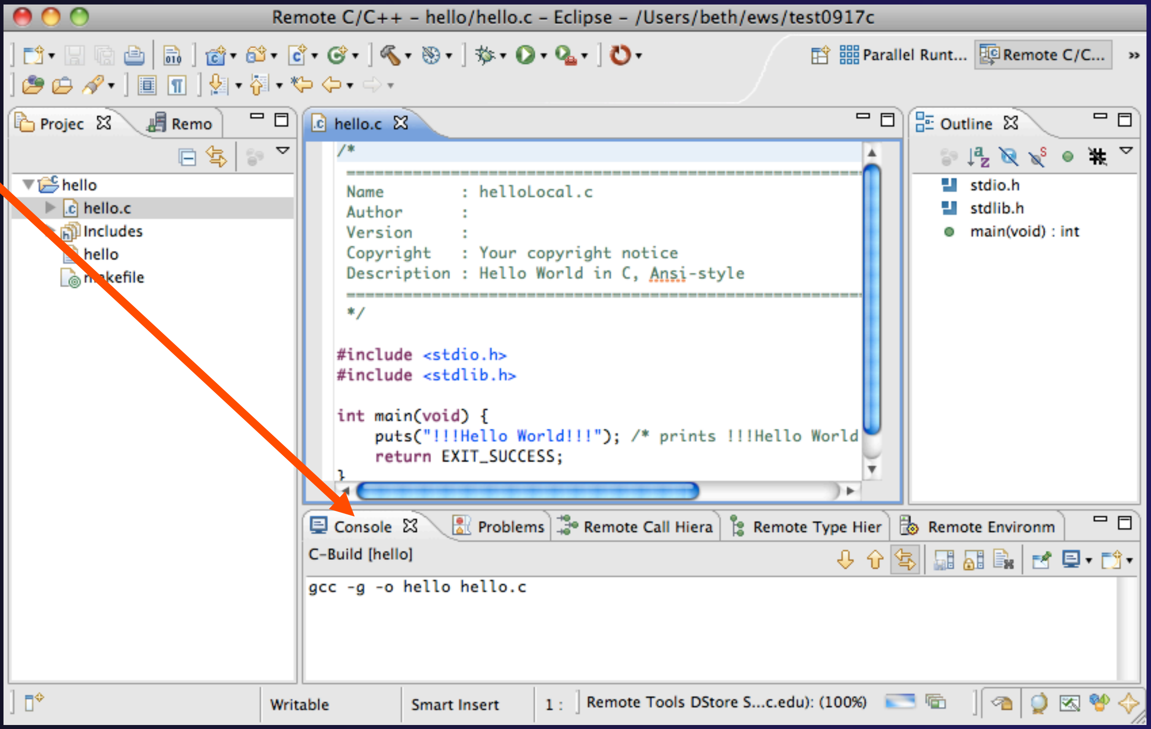
```
#include <stdio.h>
#include <stdlib.h>

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

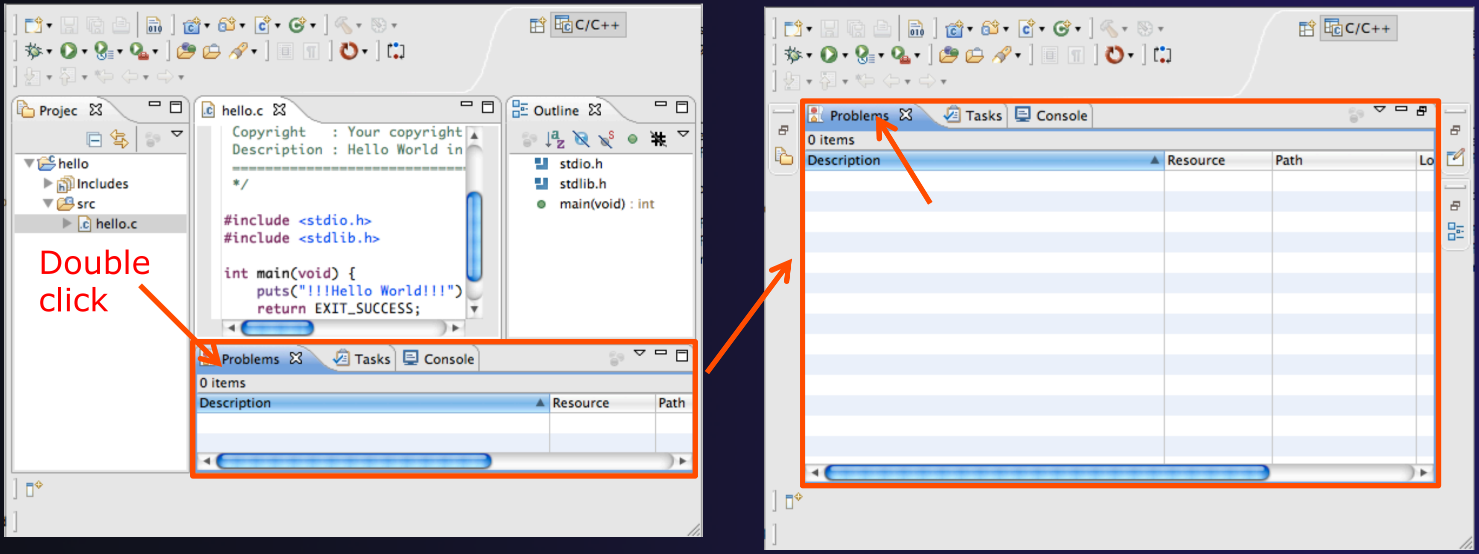


```
/*
 ============================================================
 Name        : helloLocal.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================
 */

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

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

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

Expand a View

- Double-click on a view/editor’s tab to fill the workbench with its content;
- Repeat to return to original size

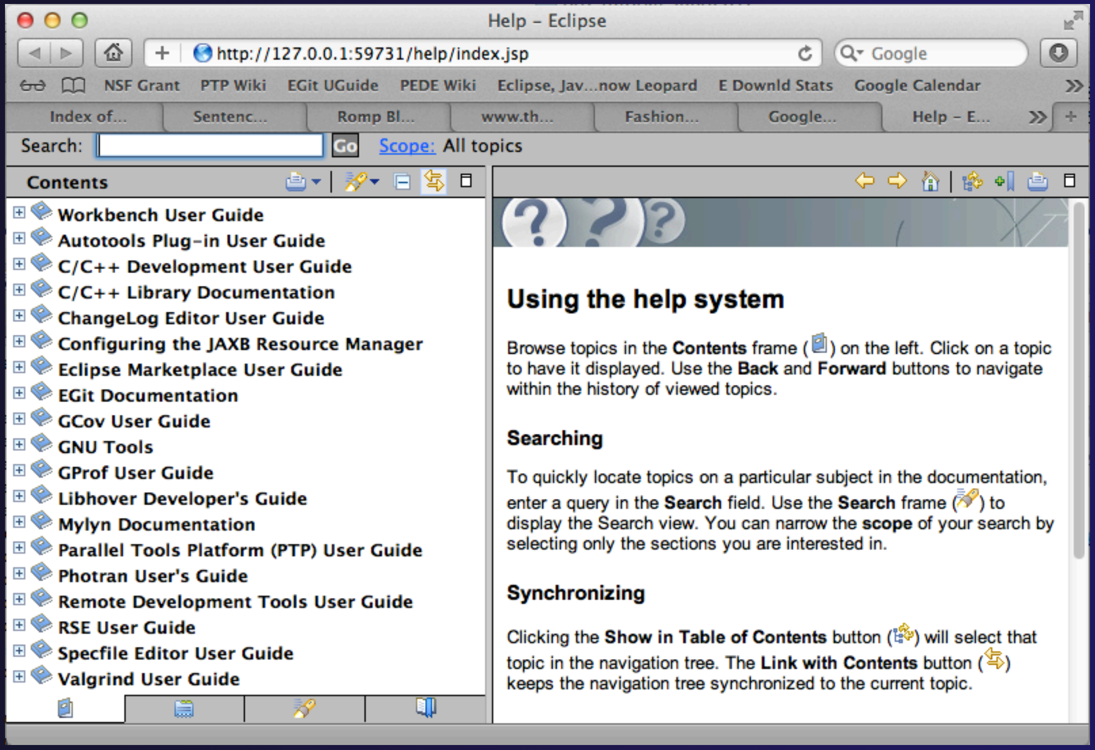


```
 Copyright   : Your copyright
 Description : Hello World in
 ================================
 */

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

int main(void) {
    puts("!!!Hello World!!!")
    return EXIT_SUCCESS;
```

- Window > Reset Perspective returns everything to original positions

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.)


Help – Eclipse
http://127.0.0.1:59731/help/index.jsp
Google
NSF Grant
PTP Wiki
EGit UGuide
PEDE Wiki
Eclipse, Jav...now Leopard
E Downld Stats
Google Calendar
Index of...
Sentenc...
Romp Bl...
www.th...
Fashion...
Google...
Help – E...
Search:
Go
Scope: All topics
Contents
Workbench User Guide
Autotools Plug-in User Guide
C/C++ Development User Guide
C/C++ Library Documentation
ChangeLog Editor User Guide
Configuring the JAXB Resource Manager
Eclipse Marketplace User Guide
EGit Documentation
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
Specfile Editor User Guide
Valgrind User Guide

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.

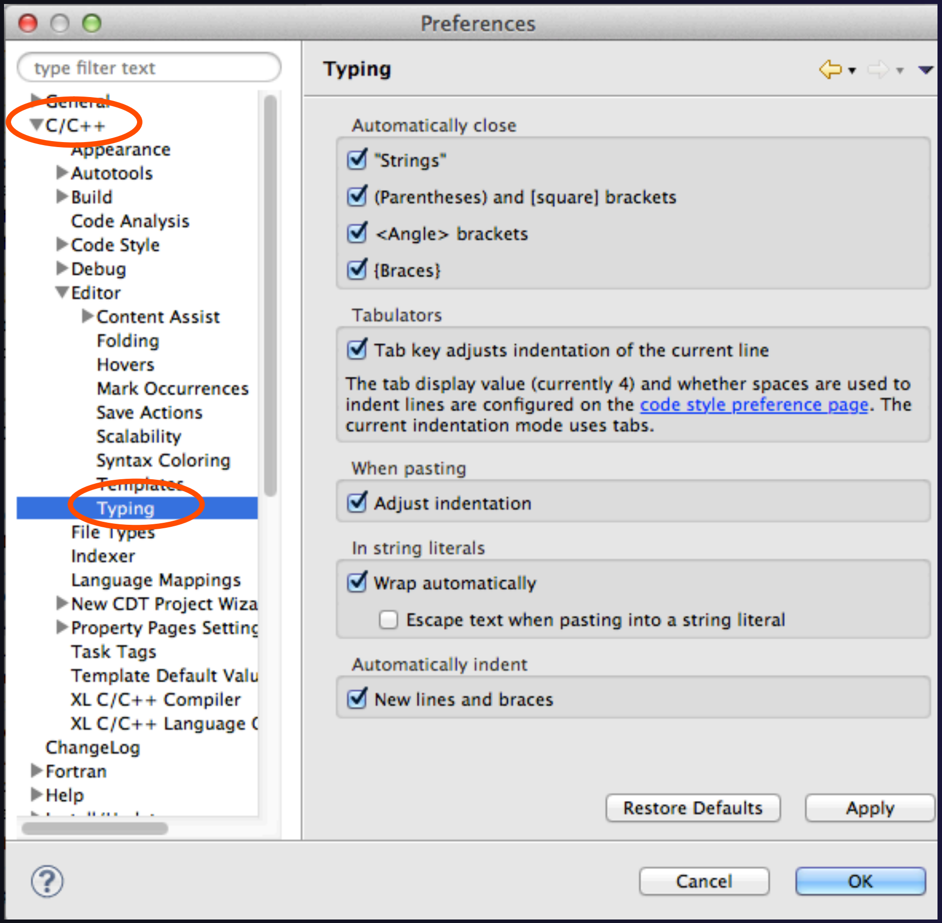
Searching

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.

Eclipse 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

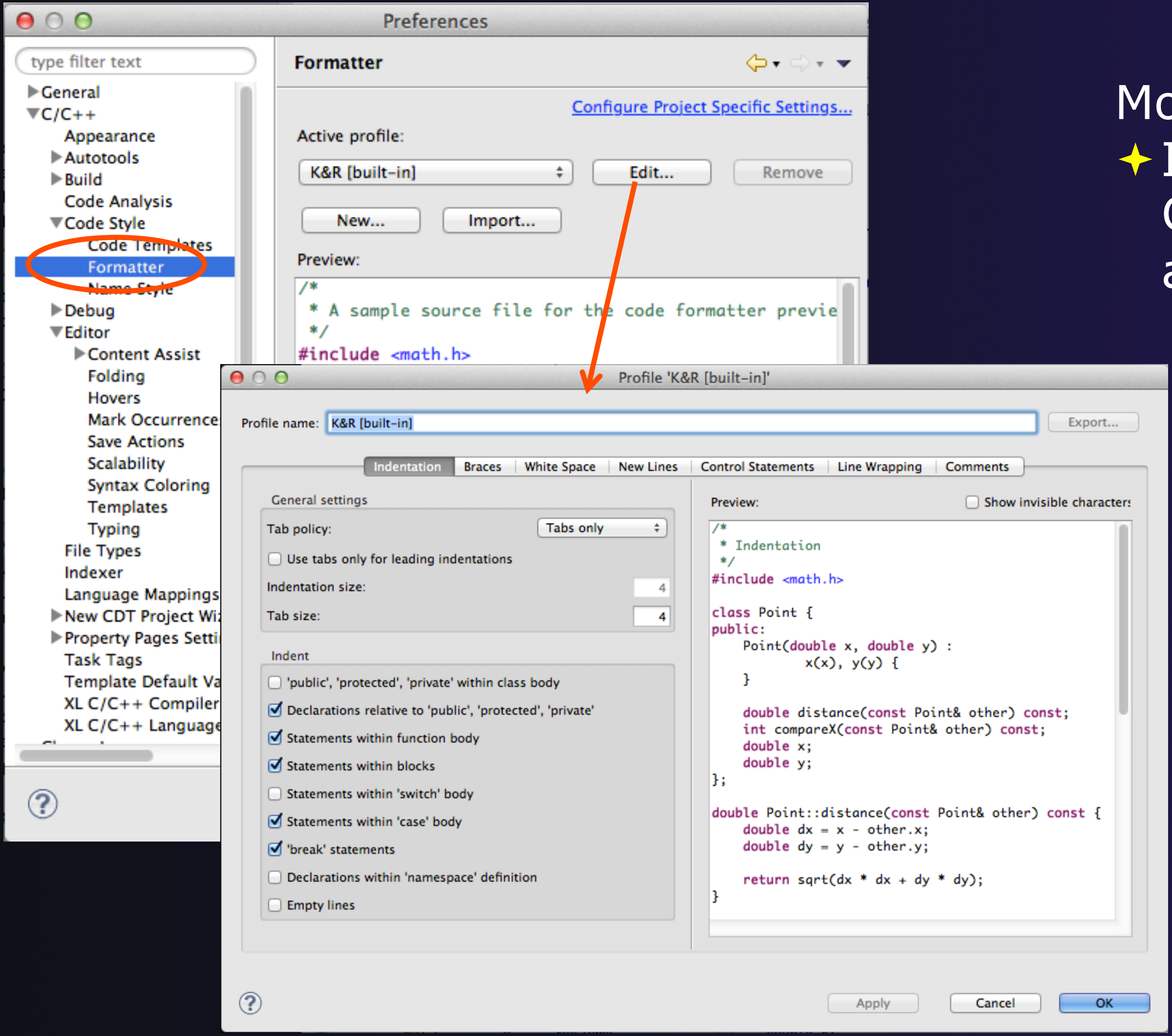
- [✓] Wrap automatically
- [ ] Escape text when pasting into a string literal

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 Example



Active profile: K&R [built-in]

Preview:

```
/*
 * A sample source file for the code formatter previe
 */
#include <math.h>
```

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

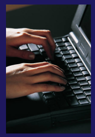
- ☐ 'public', 'protected', 'private' within class body
- ☑ Declarations relative to 'public', 'protected', 'private'
- ☑ Statements within function body
- ☑ Statements within blocks
- ☐ Statements within 'switch' body
- ☑ Statements within 'case' body
- ☑ 'break' statements
- ☐ Declarations within 'namespace'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

Preview: ☐ Show invisible characters

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

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

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

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

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

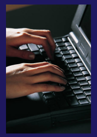
More C/C++ preferences:

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



Exercise

1. Change to a different perspective
2. Experiment with moving and resizing views
   - Move a view from a stack to beside another view
   - Expand a view to maximize it; return to original size
3. Save the perspective
4. Reset the perspective
5. Open Eclipse preferences
6. Search for "Launching"
7. Make sure the "Build (if required) before launching" setting is *disabled*



Optional Exercise

Best performed *after* learning about projects, CVS, and editors

1. Use source code formatting to format a source file, or a region of a source file
   - Use Source>Format menu
2. In Eclipse Preferences, change the C/C++ source code style formatter, e.g.
   - Change the indentation from 4 to 6
   - Make line wrapping not take effect until a line has a maximum line width of 120, instead of the default 80
   - Save a (new) profile with these settings
   - Format a source file with these settings
3. Revert the file back to the original – experiment with
   - Replace with HEAD, replace with previous from local history, or reformat using original style

Eclipse CVS

- Objective
  - Learn how to use a source code repository with Eclipse
- Contents
  - Checking out project in CVS
  - Converting to a Synchronized Project
  - Handling changes

Project Creation
Alternative #2

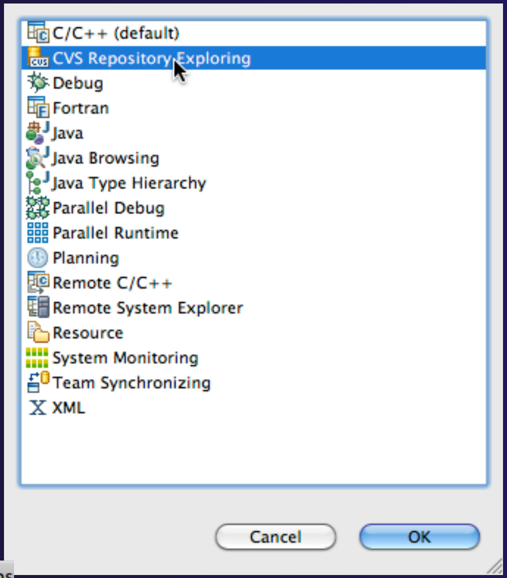
Checking out the project

Using a Source Code Repository
Introduction to Team Features

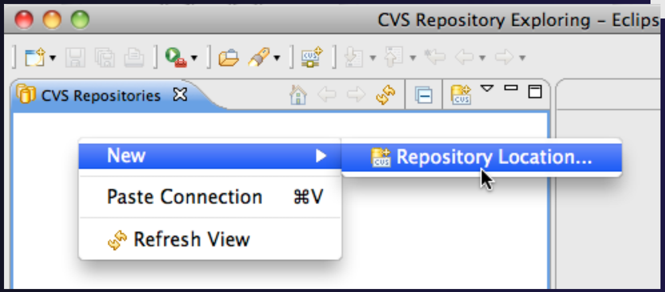
In this scenario, we will check out code from a CVS source code repository, creating a local project, and then convert it to a synchronized project on a remote host.

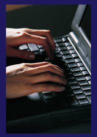
Importing a Project from CVS

- Switch to **CVS Repository Exploring** perspective
  - Window > Open Perspective > Other…
  - Select **CVS Repository Exploring**
  - Select **OK**



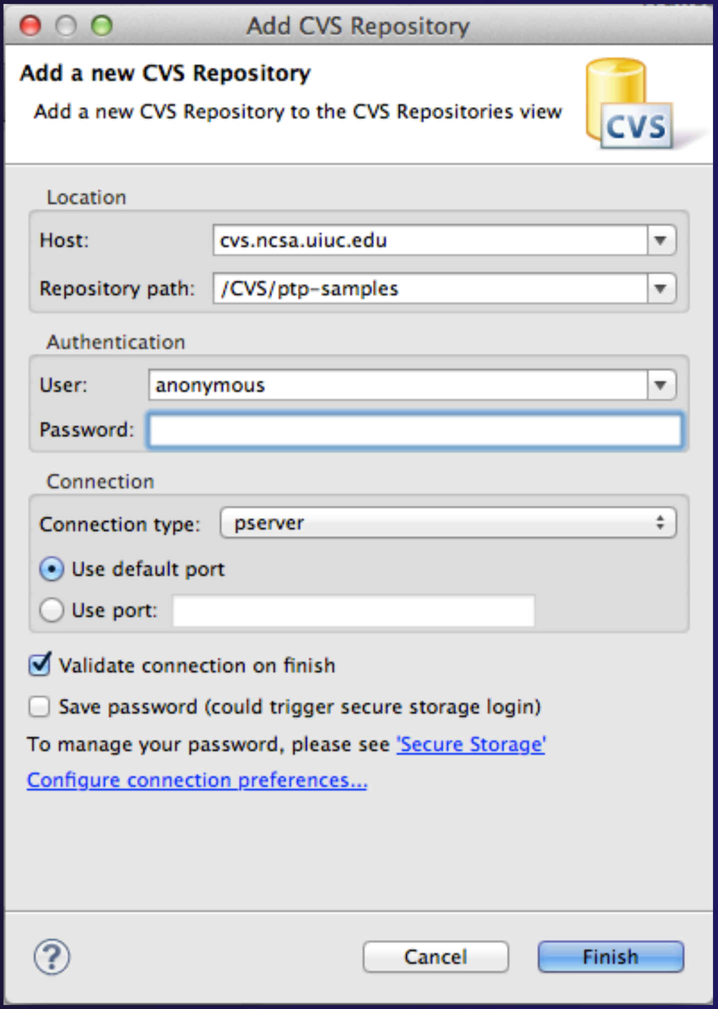
- Right click in **CVS Repositories** view and select **New>Repository Location…**





Add CVS Repository

- Enter **Host:** cvs.ncsa.uiuc.edu
- **Repository path:** /CVS/ptp-samples
- For anonymous access:
  - **User**: anonymous
  - No password is required
  - **Connection type:** pserver (default)
- For authorized access:
  - **User**: your userid
  - **Password**: your password
  - **Connection type:** change to **extssh**
- Select **Finish**



Add a new CVS Repository

Add a new CVS Repository to the CVS Repositories view

Location

Host:	cvs.ncsa.uiuc.edu
Repository path:	/CVS/ptp-samples

Authentication

User:	anonymous
Password:	

Connection

Connection type: pserver

● Use default port

○ Use port:

[✓] Validate connection on finish

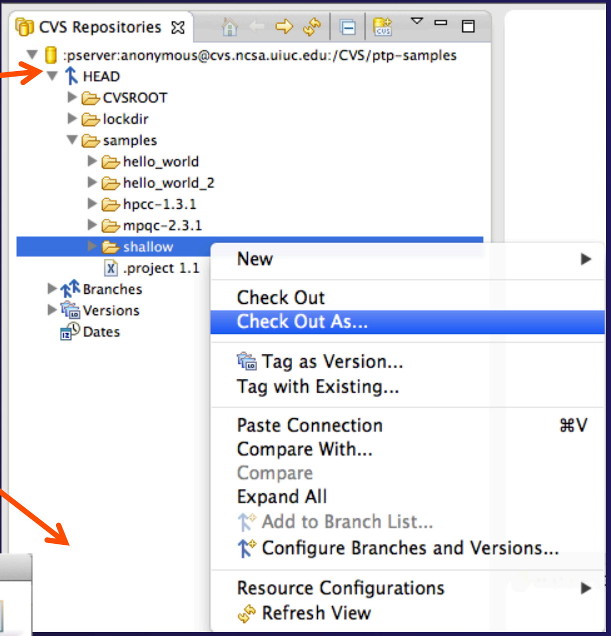
[ ] Save password (could trigger secure storage login)

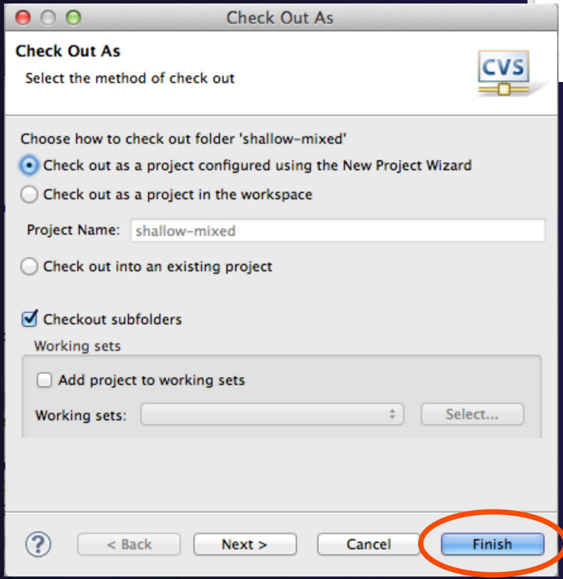
To manage your password, please see 'Secure Storage'

Configure connection preferences...

Checking out the Project

- Expand the repository location
- Expand **HEAD**
- Expand **samples**
- Right click on **shallow** and select **Check Out As...**
- On **Check Out As** dialog, select **Finish**




Check Out As

Check Out As

Select the method of check out

Choose how to check out folder 'shallow-mixed'

- [✓] Check out as a project configured using the New Project Wizard
- Check out as a project in the workspace

Project Name: shallow-mixed

- Check out into an existing project

[✓] Checkout subfolders

Working sets

[ ] Add project to working sets

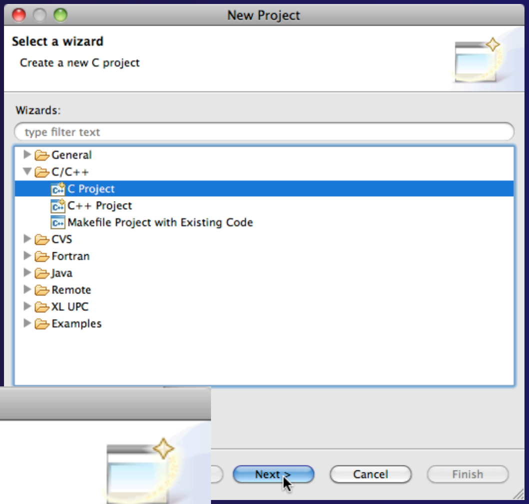
Working sets:

The default of "Check out as a project configured using the New Project Wizard" is what we want

New Project Wizard

As project is checked out from CVS, the **New Project** Wizard helps you configure the Eclipse information to be added to the project

- Expand **C/C++**
- Select **C Project** and click on **Next>**
- Enter 'shallow' as **Project Name**
- Under **Project type,** expand **Makefile project**
  - scroll to the bottom
- Select **Empty Project**
- Select **Remote Linux GCC Toolchain**
  - You may need to uncheck "Show project types and toolchains only if they are supported on the platform"
  - In general, choose a remote toolchain that matches the compiler you intend to use on the remote system
- Click on **Finish**


New Project

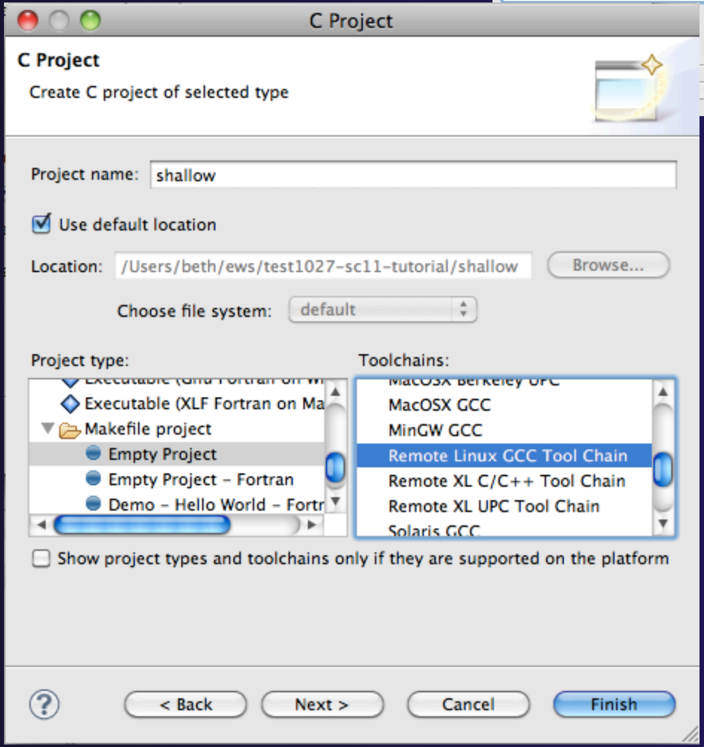
Select a wizard

Create a new C project

Wizards:

type filter text

- General
- C/C++
  - C Project
  - C++ Project
  - Makefile Project with Existing Code
- CVS
- Fortran
- Java
- Remote
- XL UPC
- Examples


C Project

C Project

Create C project of selected type

Project name: shallow

☑ Use default location

Location: /Users/beth/ews/test1027-sc11-tutorial/shallow

Choose file system: default

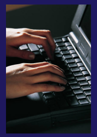
Project type:

- Executable (Gnu Fortran on Wi
- Executable (XLF Fortran on Ma
- Makefile project
  - Empty Project
  - Empty Project - Fortran
  - Demo - Hello World - Fortr

Toolchains:

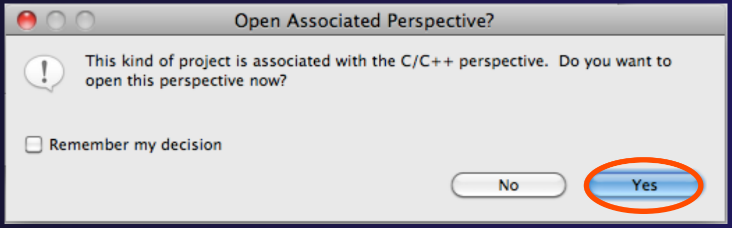
- MacOSX Berkeley UPC
- MacOSX GCC
- MinGW GCC
- Remote Linux GCC Tool Chain
- Remote XL C/C++ Tool Chain
- Remote XL UPC Tool Chain
- Solaris GCC

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



Project successfully checked out

- Switch to the C/C++ Perspective when prompted after checking our the code
- You should now see the “shallow” project in your workspace

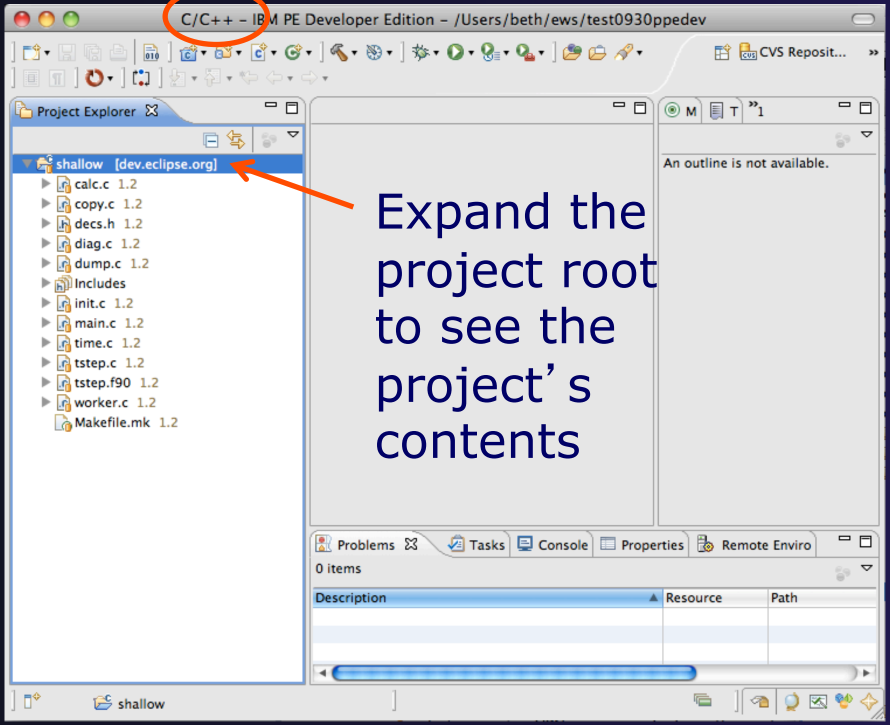


Open Associated Perspective?

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

☐ Remember my decision

No [✓ Yes]



C/C++ – IBM PE Developer Edition – /Users/beth/ews/test0930ppedev

Expand the project root to see the project’s contents

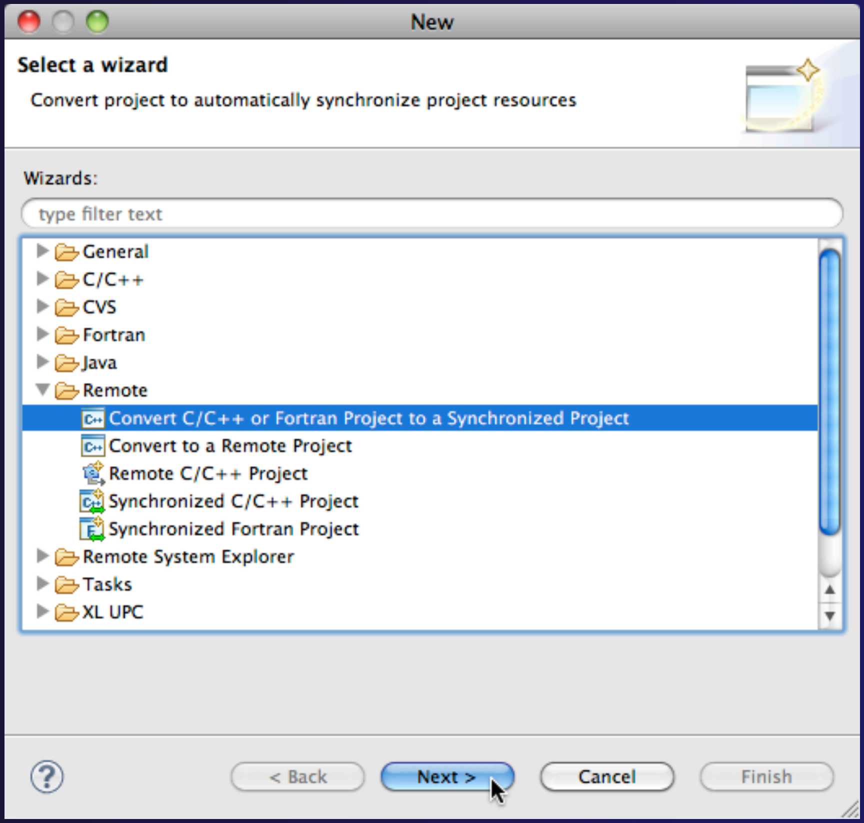
Convert to a Synchronized Project

Synchronizing the Project

- Because we will be running on a remote system, we must also build on that system
- Source files must be available to build
- We will use a synchronized project to do this
  - Only needs to be done once for each project
- Files are synchronized automatically when they are saved
- A full synchronize is also performed prior to a build

Converting To Synchronized

- Select **File>New>Other…**
- Open the Remote folder
- Select **Convert C/C++ or Fortran Project to a Synchronized Project**
- Click **Next>**


New

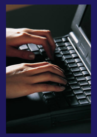
Select a wizard

Convert project to automatically synchronize project resources

Wizards:

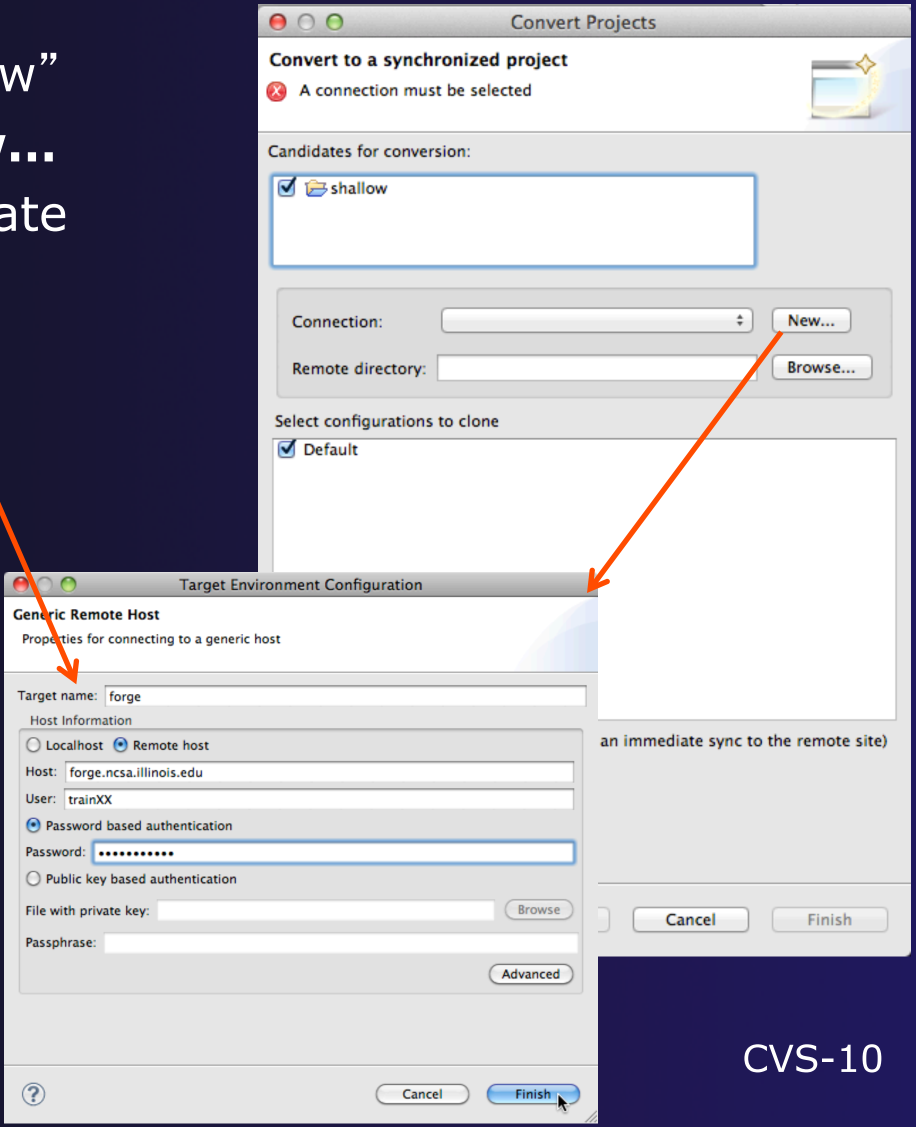
type filter text

- General
- C/C++
- CVS
- Fortran
- Java
- Remote
  - Convert C/C++ or Fortran Project to a Synchronized Project
  - Convert to a Remote Project
  - Remote C/C++ Project
  - Synchronized C/C++ Project
  - Synchronized Fortran Project
- Remote System Explorer
- Tasks
- XL UPC



Convert Projects Wizard

- Select checkbox next to “shallow”
- For **Connection:**, click on **New…**
- The tutorial instructor will indicate what to enter for:
  - **Target name**
  - **Host** name of remote system
  - **User** ID
  - **Password**
- Click **Finish** to close it
- See the connection name in the **Convert Projects** wizard for **Connection**



Convert Projects

Convert to a synchronized project

A connection must be selected

Candidates for conversion:

☑ shallow

Connection: | New…

Remote directory: | Browse…

Select configurations to clone

☑ Default

an immediate sync to the remote site)

Cancel | Finish

Target Environment Configuration

Generic Remote Host

Properties for connecting to a generic host

Target name: forge

Host Information

Localhost [✓ Remote host]

Host: forge.ncsa.illinois.edu

User: trainXX

[✓ Password based authentication]

Password: ••••••••••••

Public key based authentication

File with private key: | Browse

Passphrase:

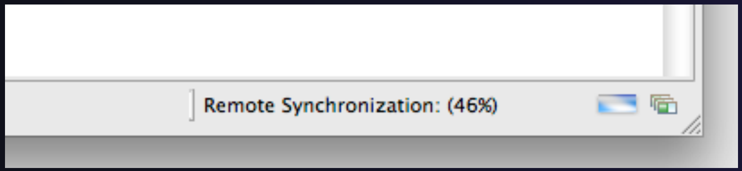
Advanced

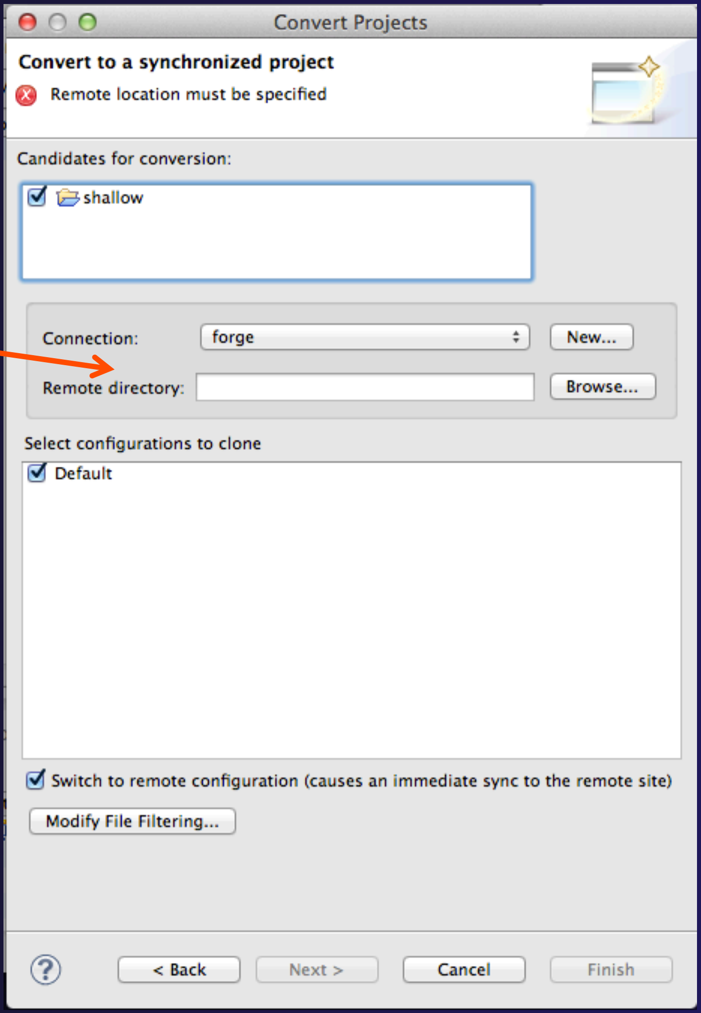
Cancel | Finish

Convert Projects Wizard (2)

Back in the **Convert Projects** dialog, we specify where the remote files will be stored

- Enter a directory name in the **Remote Directory** field: select **Browse…**
  - Sample: /u/ac/trainXX/shallow
  - Project files will be copied under this directory
- Click **Finish**
- The project should synchronize automatically


Remote Synchronization: (46%)


Convert Projects

Convert to a synchronized project

Remote location must be specified

Candidates for conversion:

☑ shallow

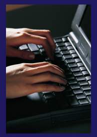
Connection: forge

Remote directory:

Select configurations to clone

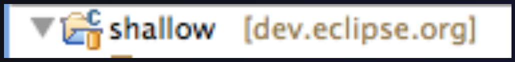
☑ Default

☑ Switch to remote configuration (causes an immediate sync to the remote site)

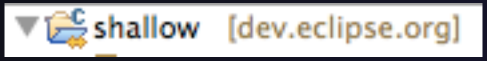


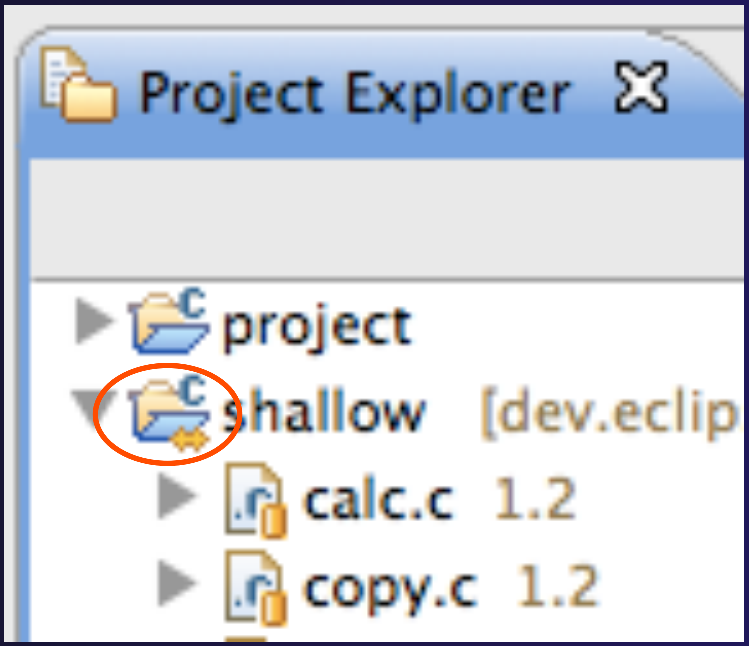
Synchronized Project

- Back in the Project Explorer, decorator on project icon indicates synchronized project
- Double-+ icon
- Before sync



- After sync





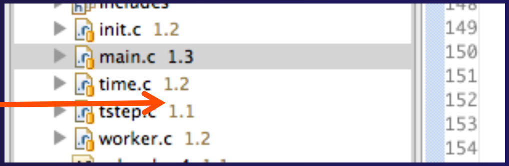
Team Features

“Team” Features

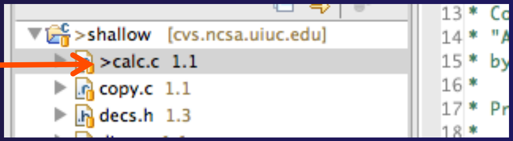
- Eclipse supports integration with multiple version control systems (VCS)
  - CVS, SVN, Git, and others
  - Collectively known as “Team” services
- Many features are common across VCS
  - Compare/merge
  - History
  - Check-in/check-out
- Some differences
  - Version numbers
  - Branching

CVS Features

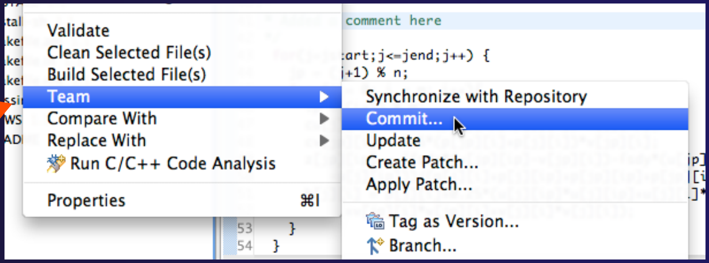
- Shows version numbers next to each resource



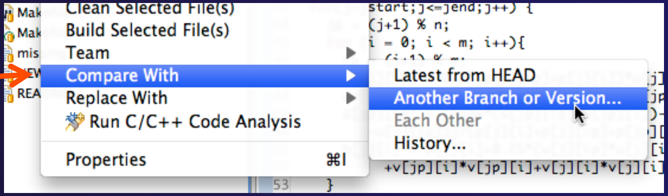
- Marks resources that have changed
  - Can also change color (preference option)



- Context menu for Team operations



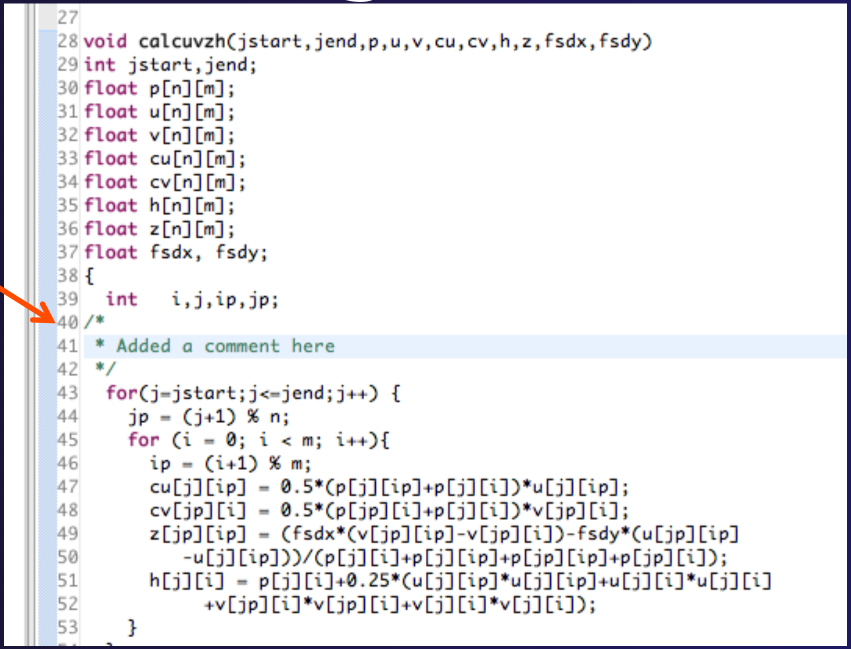
- Compare to latest, another branch, or history



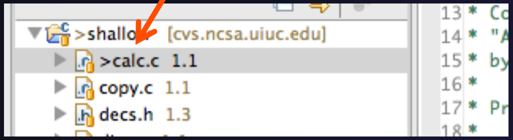
- Synchronize whole project (or any selected resources)

How to tell that you've changed something

- Open "calc.c"
- Add comment at line 40
- Save file
- File will be marked ">" to indicate that it has been modified

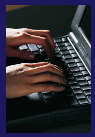


```
27
28 void calcuvzh(jstart,jend,p,u,v,cu,cv,h,z,fsdx,fsdy)
29 int jstart,jend;
30 float p[n][m];
31 float u[n][m];
32 float v[n][m];
33 float cu[n][m];
34 float cv[n][m];
35 float h[n][m];
36 float z[n][m];
37 float fsdx, fsdy;
38 {
39   int   i,j,ip,jp;
40 /*
41  * Added a comment here
42  */
43   for(j=jstart;j<=jend;j++) {
44     jp = (j+1) % n;
45     for (i = 0; i < m; i++){
46       ip = (i+1) % m;
47       cu[j][ip] = 0.5*(p[j][ip]+p[j][i])*u[j][ip];
48       cv[jp][i] = 0.5*(p[jp][i]+p[j][i])*v[jp][i];
49       z[jp][ip] = (fsdx*(v[jp][ip]-v[jp][i])-fsdy*(u[jp][ip]
50          -u[j][ip]))/(p[j][i]+p[j][ip]+p[jp][ip]+p[jp][i]);
51       h[j][i] = p[j][i]+0.25*(u[j][ip]*u[j][ip]+u[j][i]*u[j][i]
52            +v[jp][i]*v[jp][i]+v[j][i]*v[j][i]);
53     }
```

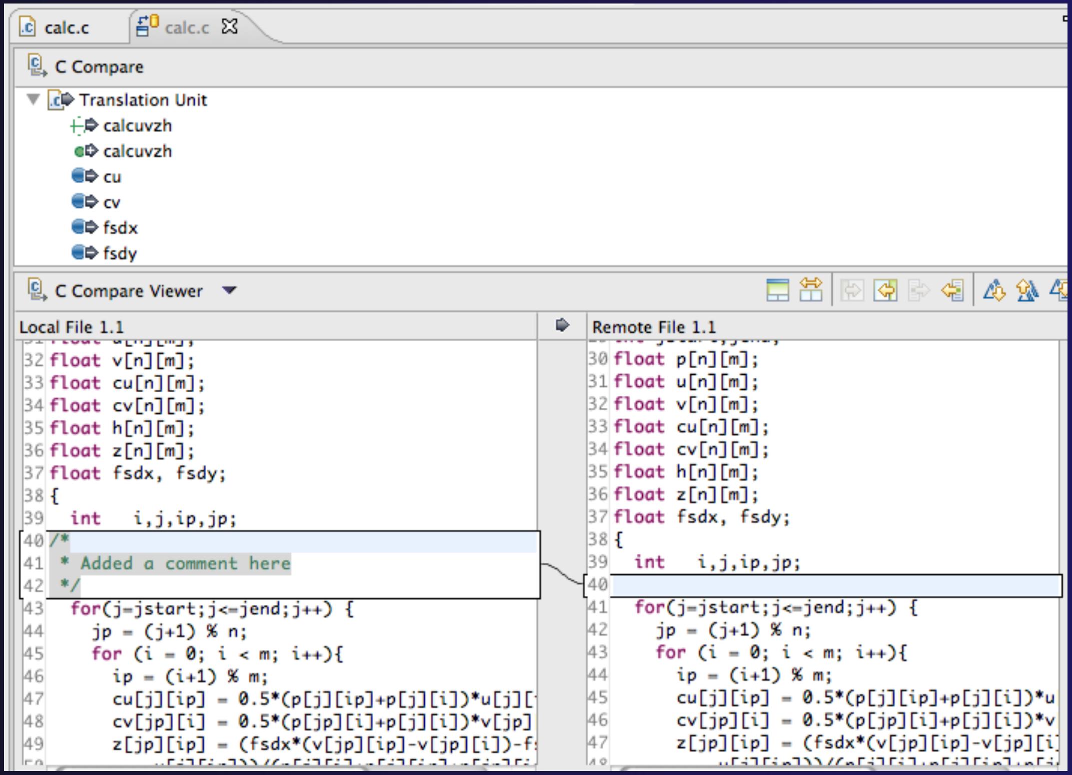


```
>shallo  [cvs.ncsa.uiuc.edu]
  >calc.c  1.1
  copy.c  1.1
  decs.h  1.3
```

Comparing your version with what's in the repository



- Right-click on "calc.c" and select **Compare With>Latest from HEAD**
- Compare editor will open showing differences between local (changed) file and the original
- Buttons allow changes to be merged from right to left
- Can also navigate between changes using buttons



Local File 1.1

```
float v[n][m];
float cu[n][m];
float cv[n][m];
float h[n][m];
float z[n][m];
float fsdx, fsdy;
{
  int   i,j,ip,jp;
/*
 * Added a comment here
 */
  for(j=jstart;j<=jend;j++) {
    jp = (j+1) % n;
    for (i = 0; i < m; i++){
      ip = (i+1) % m;
      cu[j][ip] = 0.5*(p[j][ip]+p[j][i])*u[j][
      cv[jp][i] = 0.5*(p[jp][i]+p[j][i])*v[jp]
      z[jp][ip] = (fsdx*(v[jp][ip]-v[jp][i])-f
```

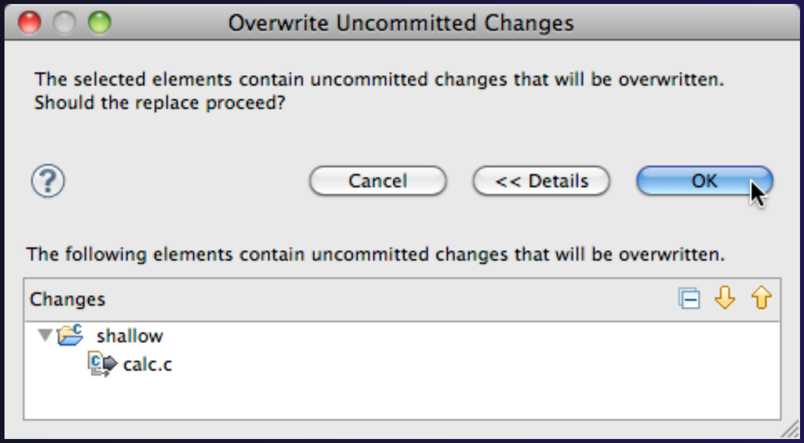
Remote File 1.1

```
float p[n][m];
float u[n][m];
float v[n][m];
float cu[n][m];
float cv[n][m];
float h[n][m];
float z[n][m];
float fsdx, fsdy;
{
  int   i,j,ip,jp;

  for(j=jstart;j<=jend;j++) {
    jp = (j+1) % n;
    for (i = 0; i < m; i++){
      ip = (i+1) % m;
      cu[j][ip] = 0.5*(p[j][ip]+p[j][i])*u
      cv[jp][i] = 0.5*(p[jp][i]+p[j][i])*v
      z[jp][ip] = (fsdx*(v[jp][ip]-v[jp][i]
```

Revert To The Latest Version

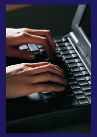
To replace your project contents to the current contents of the project in the src code repo,

- Right-click on the “shallow” project and select **Replace With>Latest from HEAD**
- Review the resources that will be replaced, then click **OK**



The selected elements contain uncommitted changes that will be overwritten. Should the replace proceed?

The following elements contain uncommitted changes that will be overwritten.



Exercise

- Check out the *shallow* project from CVS and convert it to a synchronized project as described in this module

Optional Exercise

1. Name every person who modified the Makefile
2. Identify which parts of the Makefile changed since revision 1.3

Hint: *Right-click the Makefile and select* ***Team > Show History****. Both of these can be done from the History view.*

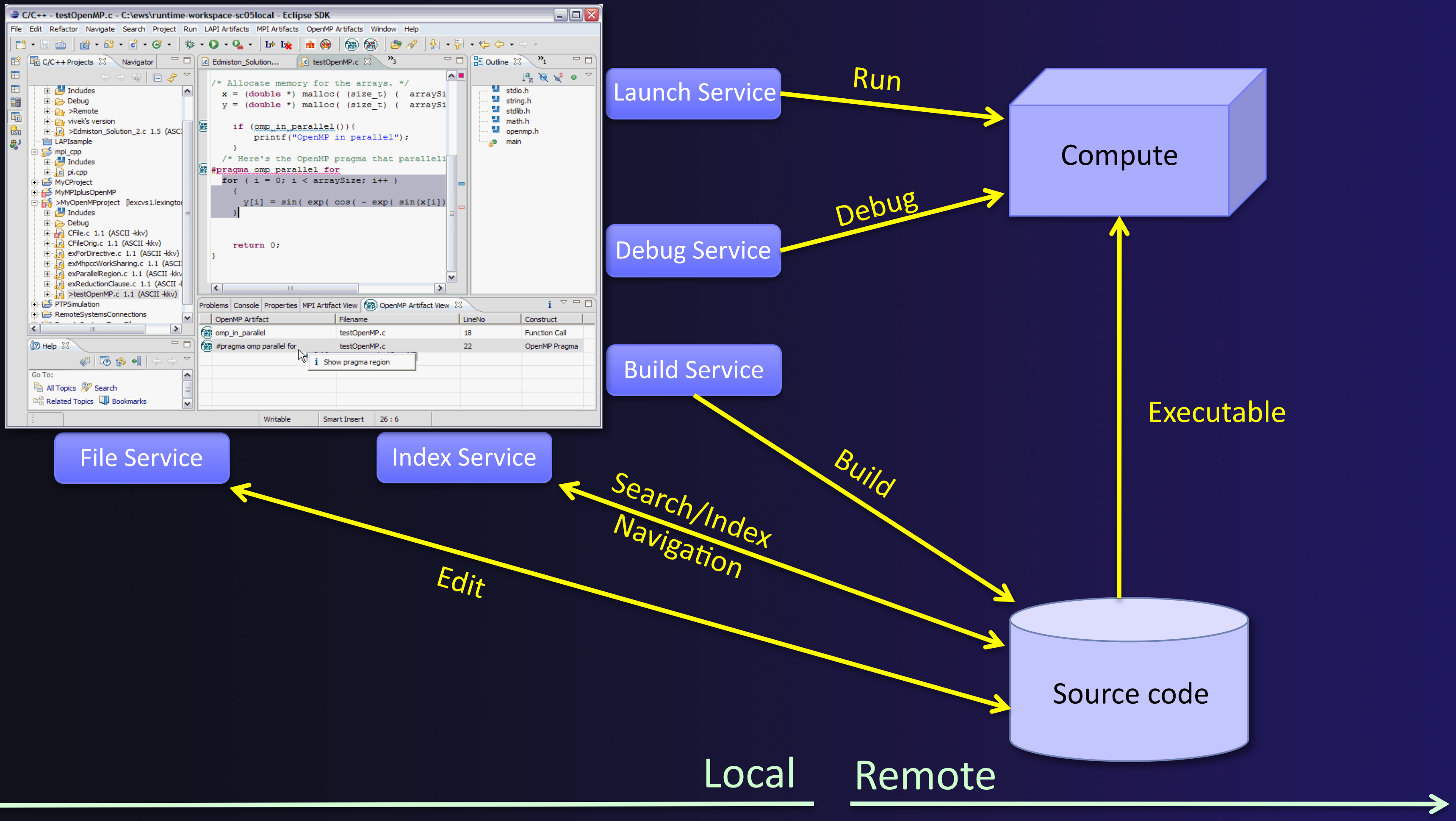
Creating a Synchronized Project

- Objective
  - Learn how to create and use synchronized projects
  - Learn how to create a sync project directly from source that already exists on a remote machine
- Contents
  - Eclipse project types
  - Creating a synchronized project
  - Using synchronize filters
  - Converting an existing project to synchronized

Project Types

- Local
  - Source is located on local machine, builds happen locally
- Remote
  - Source is located on remote machine(s), build and launch takes place on remote machine(s)
- Synchronized
  - Source is local, then synchronized with remote machine(s) (or vice-versa)
  - Building and launching happens remotely (can also happen locally)

Remote Projects

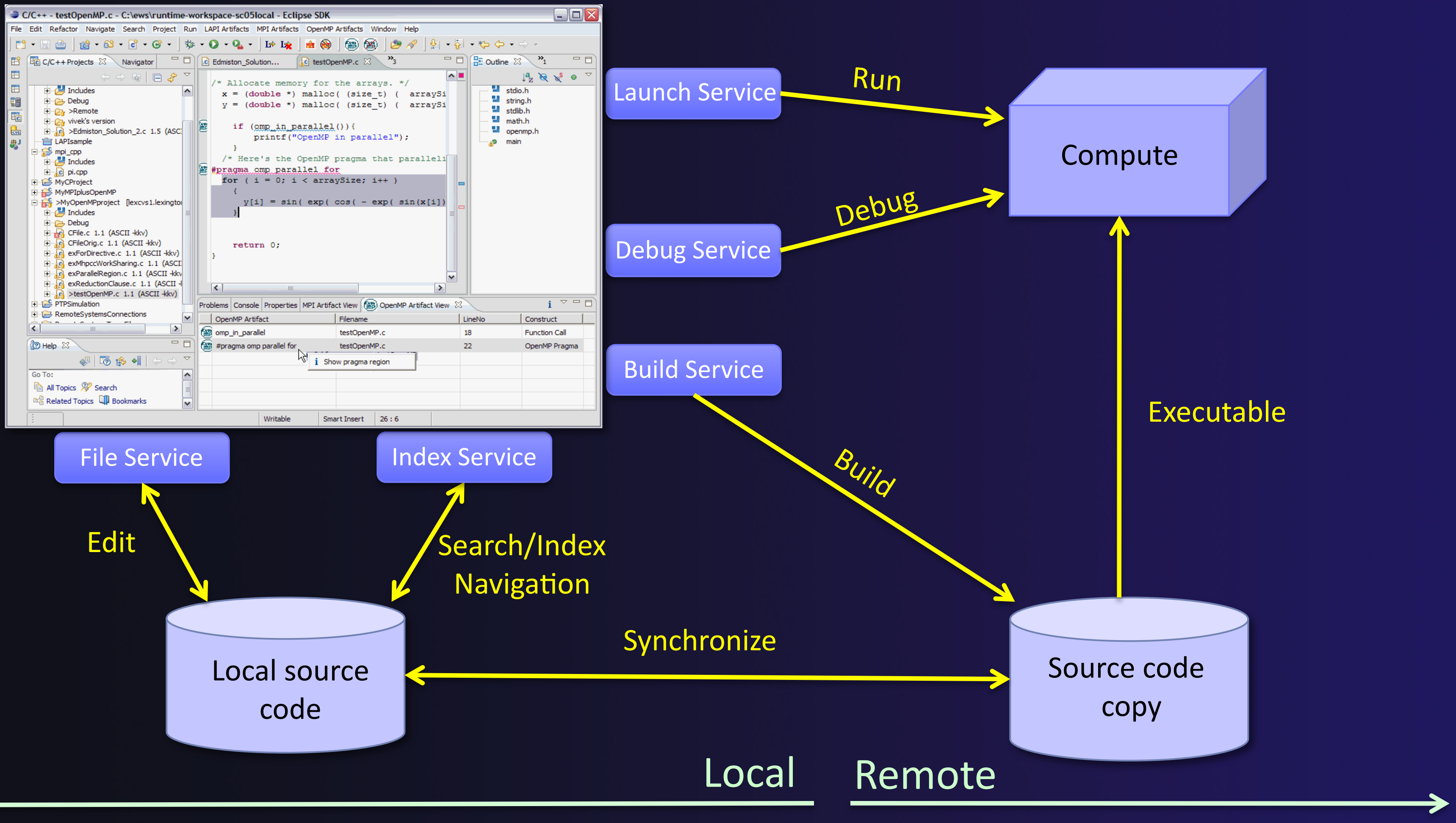


```
/* Allocate memory for the arrays. */
  x = (double *) malloc( (size_t) (  arraySi
  y = (double *) malloc( (size_t) (  arraySi

    if (omp_in_parallel()){
        printf("OpenMP in parallel");
    }
  /* Here's the OpenMP pragma that paralleli
#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	Construct
omp_in_parallel	testOpenMP.c	18	Function Call
#pragma omp parallel for	testOpenMP.c	22	OpenMP Pragma

Synchronized Projects

Synchronized Projects



C/C++ - testOpenMP.c - C:\ews\runtime-workspace-sc05local - Eclipse SDK

```
/* Allocate memory for the arrays. */
  x = (double *) malloc( (size_t) (  arraySi
  y = (double *) malloc( (size_t) (  arraySi

    if (omp_in_parallel()){
        printf("OpenMP in parallel");
    }
  /* Here's the OpenMP pragma that paralleli
#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	Construct
omp_in_parallel	testOpenMP.c	18	Function Call
#pragma omp parallel for	testOpenMP.c	22	OpenMP Pragma

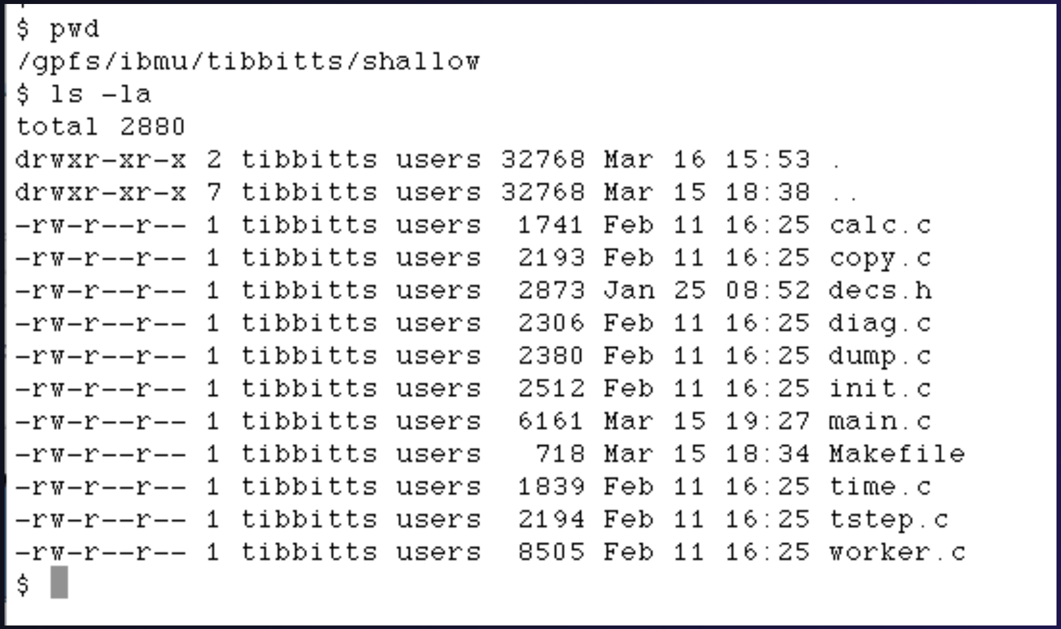
C, C++, and Fortran Projects

Build types

- Makefile-based
  - Project contains its own makefile (or makefiles) for building the application – or other build command
- Managed
  - Eclipse manages the build process, no makefile required

Source Code for project

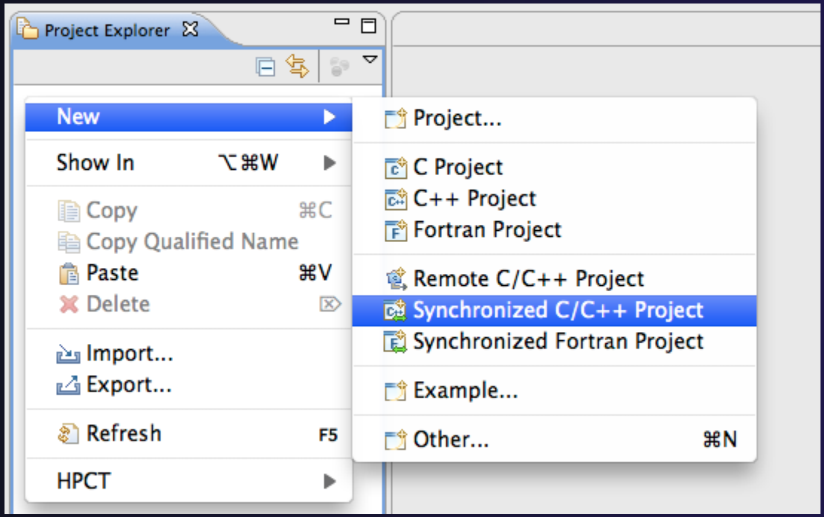
- Source code exists on remote target



```
$ pwd
/gpfs/ibmu/tibbitts/shallow
$ ls -la
total 2880
drwxr-xr-x 2 tibbitts users 32768 Mar 16 15:53 .
drwxr-xr-x 7 tibbitts users 32768 Mar 15 18:38 ..
-rw-r--r-- 1 tibbitts users  1741 Feb 11 16:25 calc.c
-rw-r--r-- 1 tibbitts users  2193 Feb 11 16:25 copy.c
-rw-r--r-- 1 tibbitts users  2873 Jan 25 08:52 decs.h
-rw-r--r-- 1 tibbitts users  2306 Feb 11 16:25 diag.c
-rw-r--r-- 1 tibbitts users  2380 Feb 11 16:25 dump.c
-rw-r--r-- 1 tibbitts users  2512 Feb 11 16:25 init.c
-rw-r--r-- 1 tibbitts users  6161 Mar 15 19:27 main.c
-rw-r--r-- 1 tibbitts users   718 Mar 15 18:34 Makefile
-rw-r--r-- 1 tibbitts users  1839 Feb 11 16:25 time.c
-rw-r--r-- 1 tibbitts users  2194 Feb 11 16:25 tstep.c
-rw-r--r-- 1 tibbitts users  8505 Feb 11 16:25 worker.c
$ 
```

Create Synchronized Project

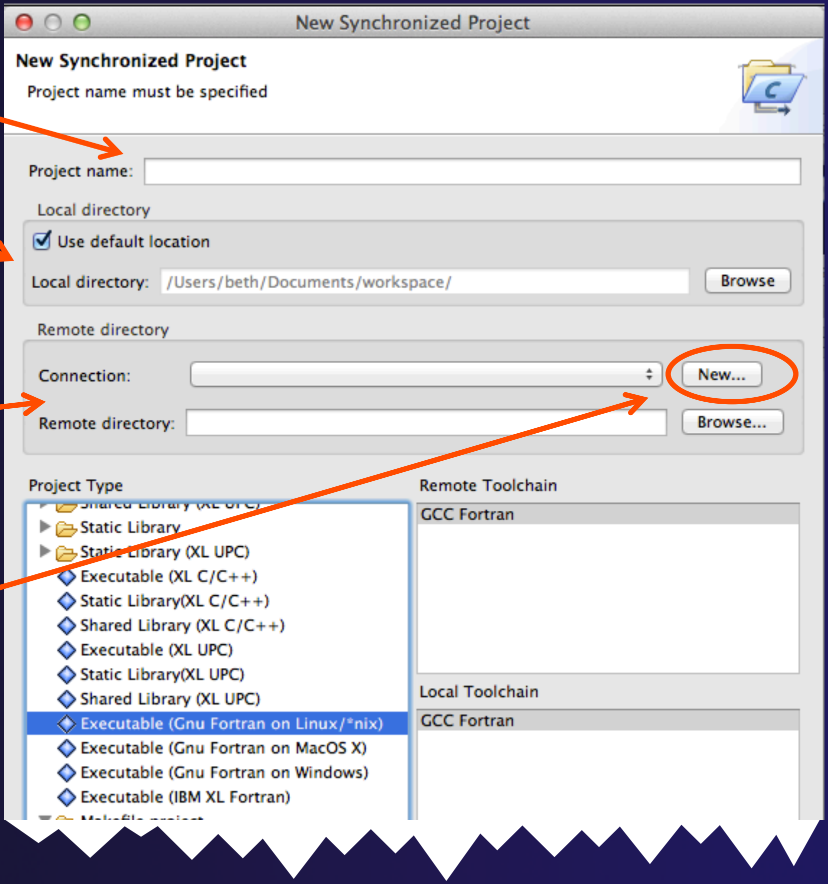
- In the Project Explorer, right click then choose
  - **New>Synchronized C/C++ Project** if your project is C/C++ only



  - **New>Synchronized Fortran Project** if your project contains Fortran files
  - This adds a Fortran nature so you can access Fortran properties, etc.

New Synchronized Project Wizard

- Enter the **Project Name**
  - E.g. "shallow"
- The **Local Directory** specifies where the local files are located
  - Leave as default
- The **Remote Directory** specifies where the remote files are located
  - Select a connection to the remote machine, or click on **New...** to create a new one
    (See next slide)
  - Browse for the directory on the remote machine


New Synchronized Project

New Synchronized Project

Project name must be specified

Project name:

Local directory

☑ Use default location

Local directory: /Users/beth/Documents/workspace/

Remote directory

Connection:

Remote directory:

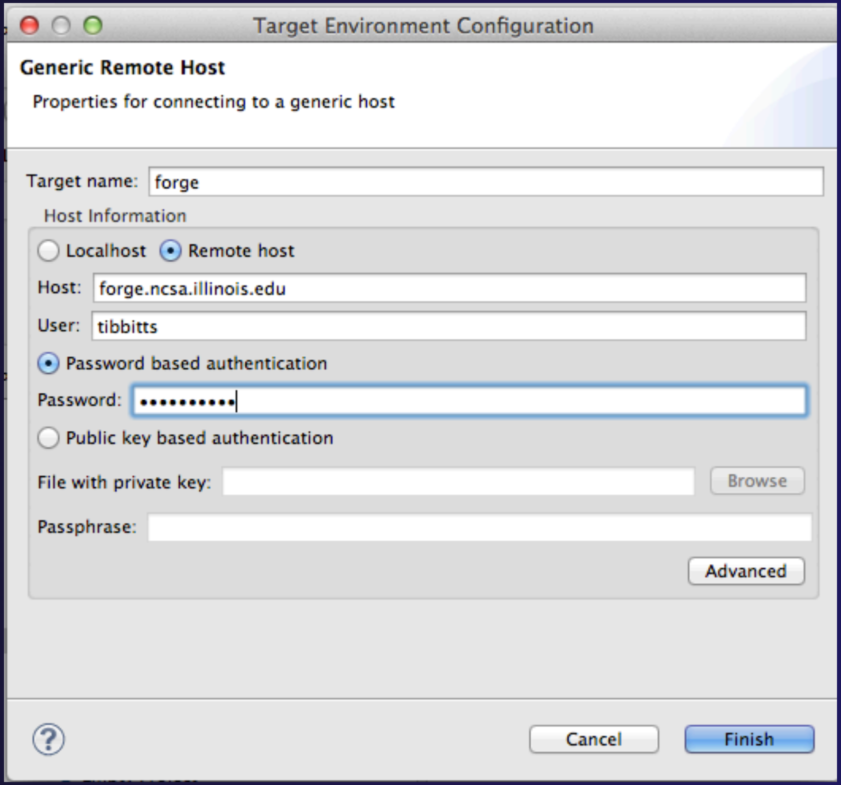
Project Type
- Static Library
- Static Library (XL UPC)
- Executable (XL C/C++)
- Static Library(XL C/C++)
- Shared Library (XL C/C++)
- Executable (XL UPC)
- Static Library(XL UPC)
- Shared Library (XL UPC)
- Executable (Gnu Fortran on Linux/*nix)
- Executable (Gnu Fortran on MacOS X)
- Executable (Gnu Fortran on Windows)
- Executable (IBM XL Fortran)

Remote Toolchain: GCC Fortran

Local Toolchain: GCC Fortran

Creating a Connection

- In the **Target Environment Configuration** dialog
  - Enter a **Target name** for the remote host
  - Enter host name, user name, and user password or other credentials
  - Select **Finish**



Generic Remote Host

Properties for connecting to a generic host

Target name: forge

Host Information

Localhost [✓ Remote host]

Host: forge.ncsa.illinois.edu

User: tibbitts

[✓ Password based authentication]

Password: ••••••••••

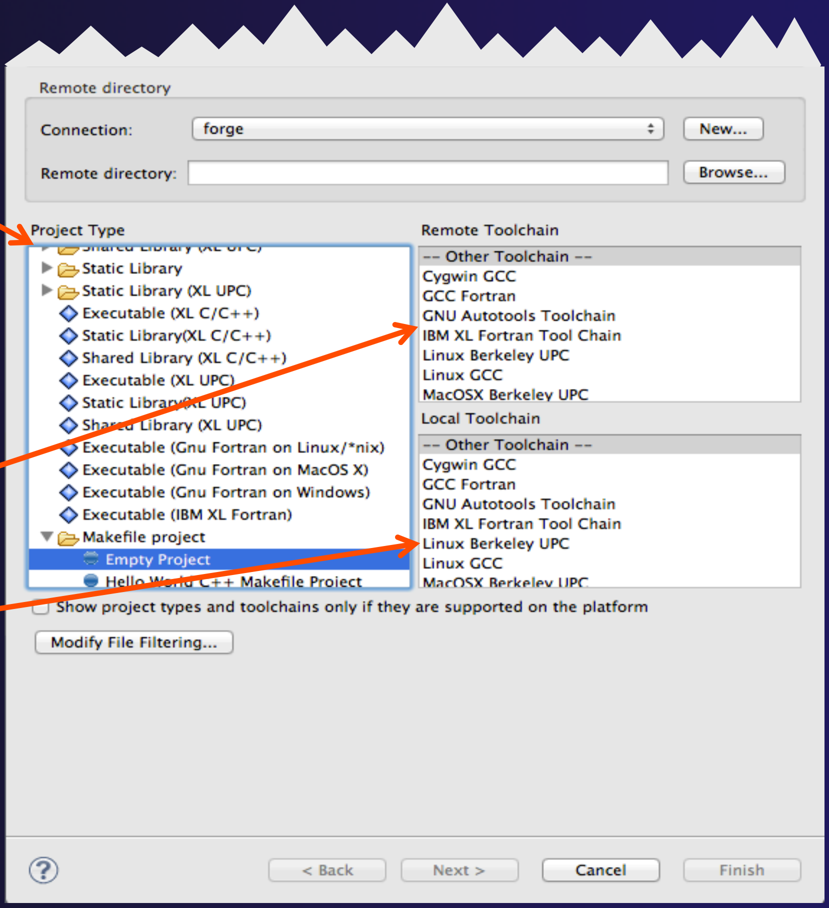
Public key based authentication

File with private key:

Passphrase:

Project Type & Toolchain

- Choose the **Project Type**
  - If you are synchronizing with an existing project, use **Makefile Project>Empty Project**
  - Otherwise, choose the type of project you want to create
- Choose the toolchain for the remote build
  - Use a toolchain that most closely matches the remote system
- Choose a toolchain for the local build
  - This is used for advanced editing/ searching
- Use **Modify File Filtering…** if required (see later slide)
- Click **Finish** to create the project


Remote directory

Connection: forge

Remote directory:

Project Type
- Shared Library (XL UPC)
- Static Library
- Static Library (XL UPC)
- Executable (XL C/C++)
- Static Library(XL C/C++)
- Shared Library (XL C/C++)
- Executable (XL UPC)
- Static Library(XL UPC)
- Shared Library (XL UPC)
- Executable (Gnu Fortran on Linux/*nix)
- Executable (Gnu Fortran on MacOS X)
- Executable (Gnu Fortran on Windows)
- Executable (IBM XL Fortran)
- Makefile project
  - Empty Project
  - Hello World C++ Makefile Project

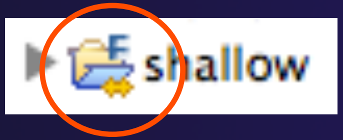
Remote Toolchain
- -- Other Toolchain --
- Cygwin GCC
- GCC Fortran
- GNU Autotools Toolchain
- IBM XL Fortran Tool Chain
- Linux Berkeley UPC
- Linux GCC
- MacOSX Berkeley UPC

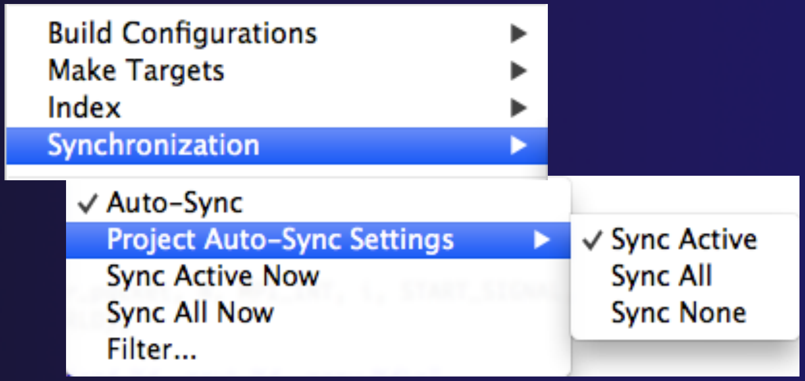
Local Toolchain
- -- Other Toolchain --
- Cygwin GCC
- GCC Fortran
- GNU Autotools Toolchain
- IBM XL Fortran Tool Chain
- Linux Berkeley UPC
- Linux GCC
- MacOSX Berkeley UPC

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

Synchronized Project

- Synchronized projects are indicated with a "synchronized" icon
- Right click on project to access **Synchronization** menu
  - Select **Auto-Sync** to enable/disable automatic syncing
  - **Project Auto-Sync Settings** are used to determine which configurations are synchronized (**Active** only, **All** or **None**)
  - **Sync Active/All Now** to manually synchronize
  - **Filter…** to manage synchronization filters



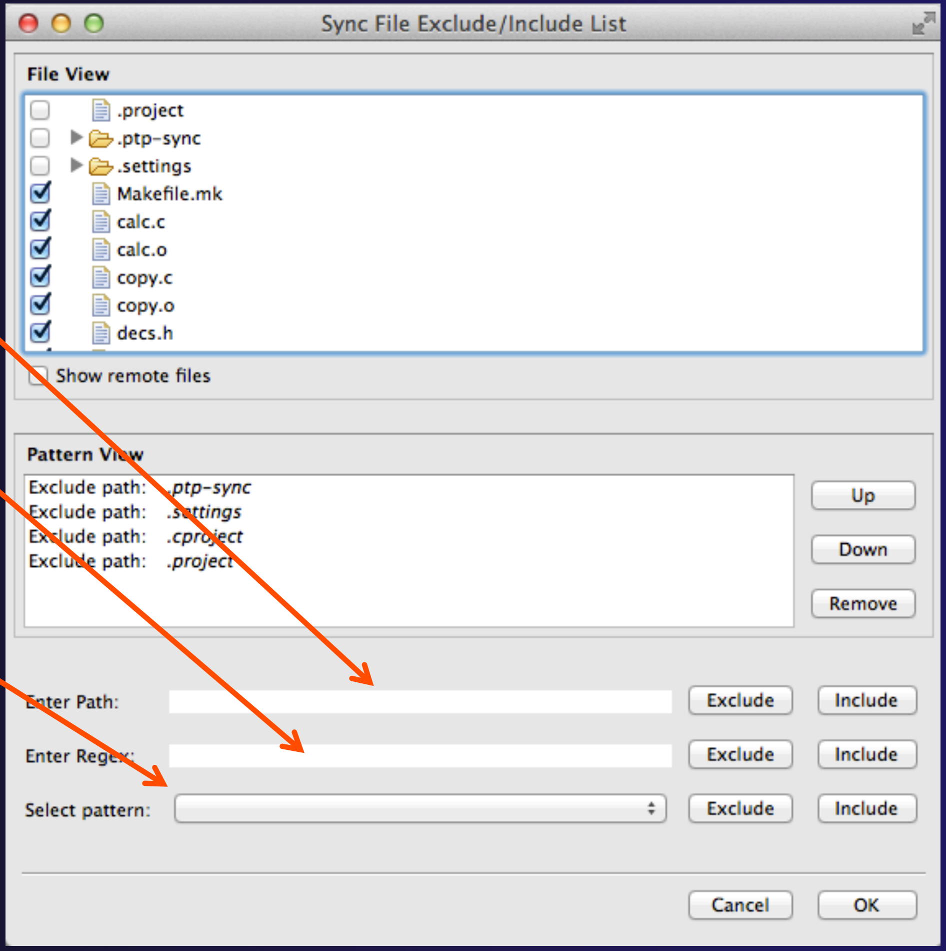


Synchronize Filters

- If not all files in the remote project should be synchronized, a filter can be set up
  - For example, it may not be desirable to synchronize binary files, or large data files
- Filters can be created at the same time as the project is created
  - Click on the **Modify File Filtering...** button in the New Project wizard
- Filters can be added later
  - Right click on the project and select **Synchronization>Filter...**

Synchronize Filter Dialog

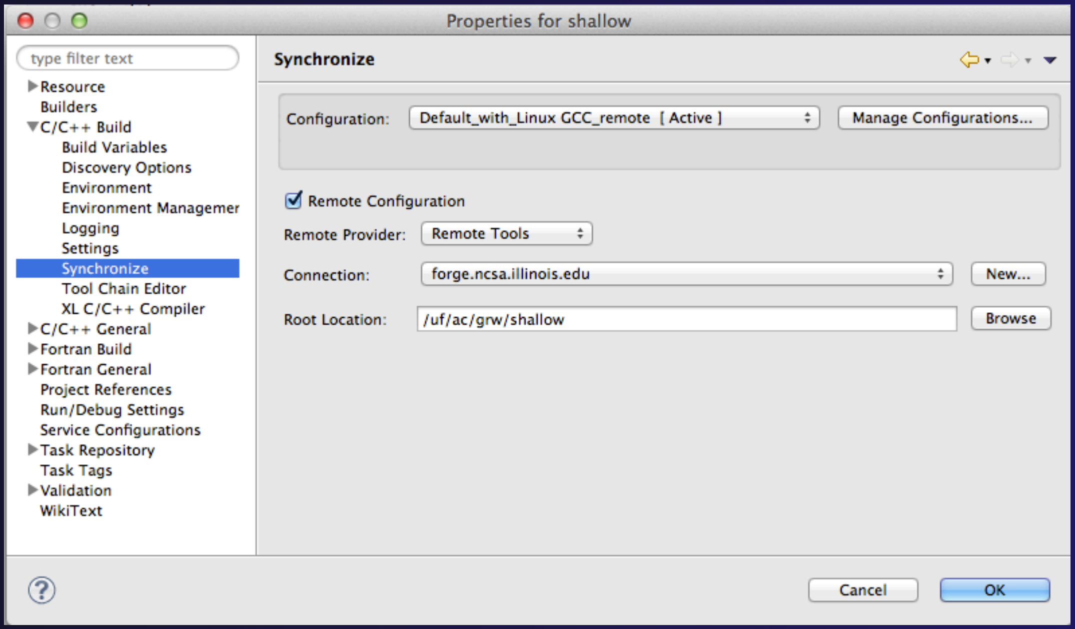
- Files can be filtered individually by selecting/unselecting them in the **File View**
- Include or exclude files based on paths
- Include or exclude files based on regular expressions
- Include or exclude files based on predefined patterns, such as "Binary files"



Exclude path: .ptp-sync
Exclude path: .settings
Exclude path: .cproject
Exclude path: .project

Synchronized Project Properties

- Synchronized project properties can be configured manually
- Open the project properties, then select **C/C++ Build>Synchronize**
- Each configuration is associated with a remote connection and a root directory
- Can be changed manually, but only if you know what you are doing!



Configuration: Default_with_Linux GCC_remote [Active]

☑ Remote Configuration

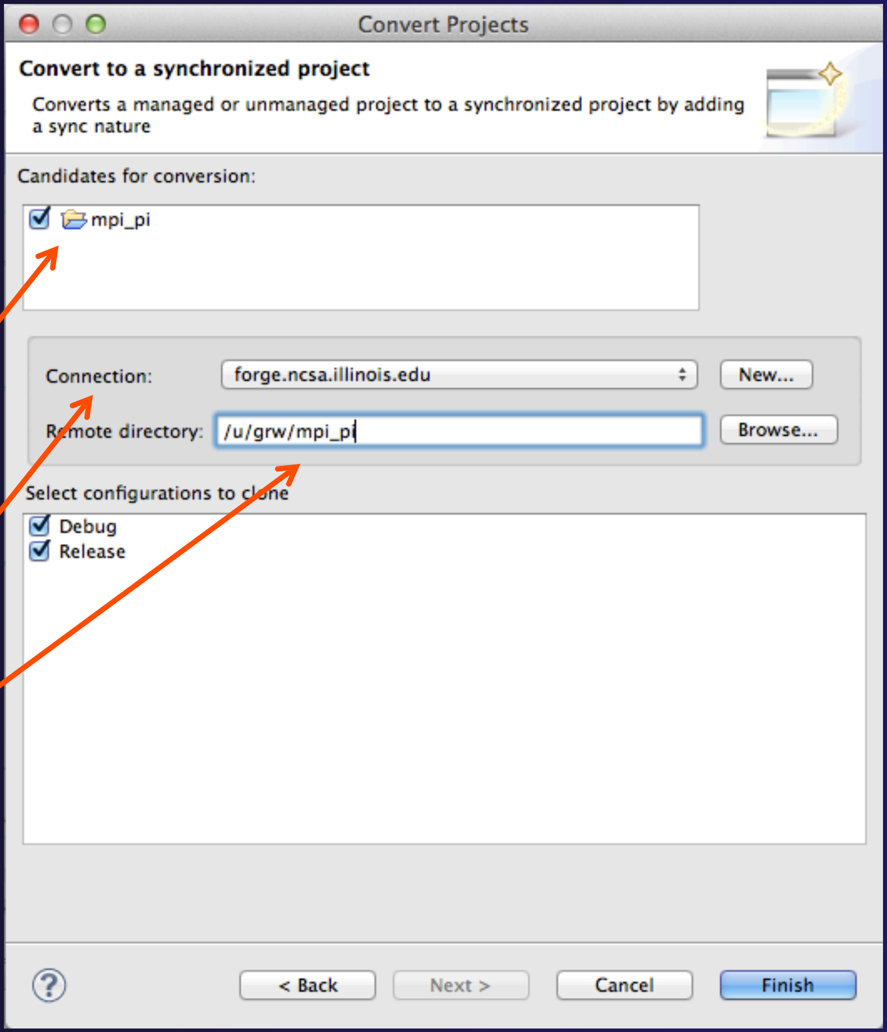
Remote Provider: Remote Tools

Connection: forge.ncsa.illinois.edu

Root Location: /uf/ac/grw/shallow

Convert Project to Synchronized

- If you have an existing project in your workspace, you can convert it to a synchronized project
- Select **File>New>Other…**
- Open the **Remote** folder
- Select **Convert C/C++ or Fortran Project to a Synchronized Project**
- Click **Next>**
- Select the project to convert from the **Candidates for conversion**
- Select a connection (or create a new one)
- Browse for or enter a directory on the remote system
  - This should normally be an empty directory
- Click **Finish**



Convert Projects

Convert to a synchronized project

Converts a managed or unmanaged project to a synchronized project by adding a sync nature

Candidates for conversion:

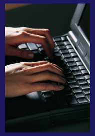
☑ mpi_pi

Connection: forge.ncsa.illinois.edu

Remote directory: /u/grw/mpi_pi

Select configurations to clone

☑ Debug
☑ Release



Exercise

- Create a synchronized project
  - Your login information and source directory will be provided by the tutorial instructor
- Observe that the project files are copied to your workspace
- Open a file in an editor and add a comment
- Observe that the file is synchronized when you save the file

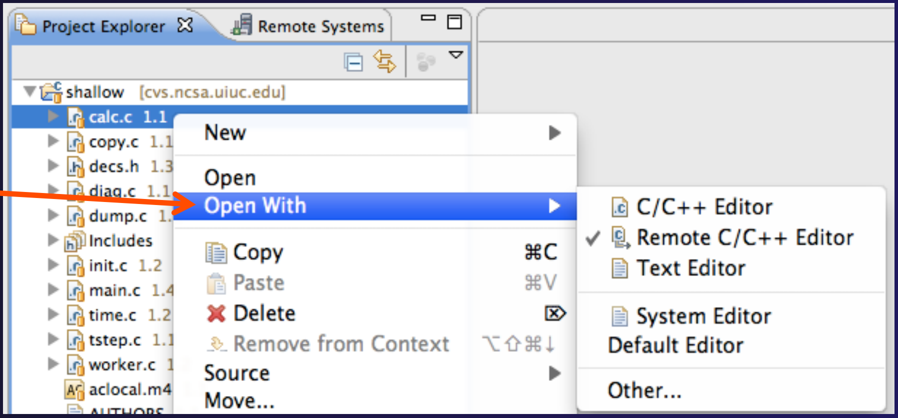
Optional Exercise

Editor Features

- Objective
  - Learn about Eclipse editor features
- Contents
  - Saving
  - Editor markers
  - Code analysis
  - Content assistance and templates

Editors

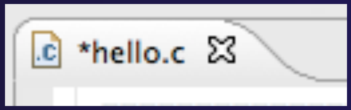
- An editor for a resource (e.g. a file) opens when you double-click on a resource
- The type of editor depends on the type of the resource
  - .c files are opened with the C/C++ editor by default
  - You can use **Open With** to use another editor
  - In this case the default editor is fine (double-click)



  - 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

Saving File in Editor

- When you change a file in the editor, an asterisk on the editor's title bar indicates unsaved changes

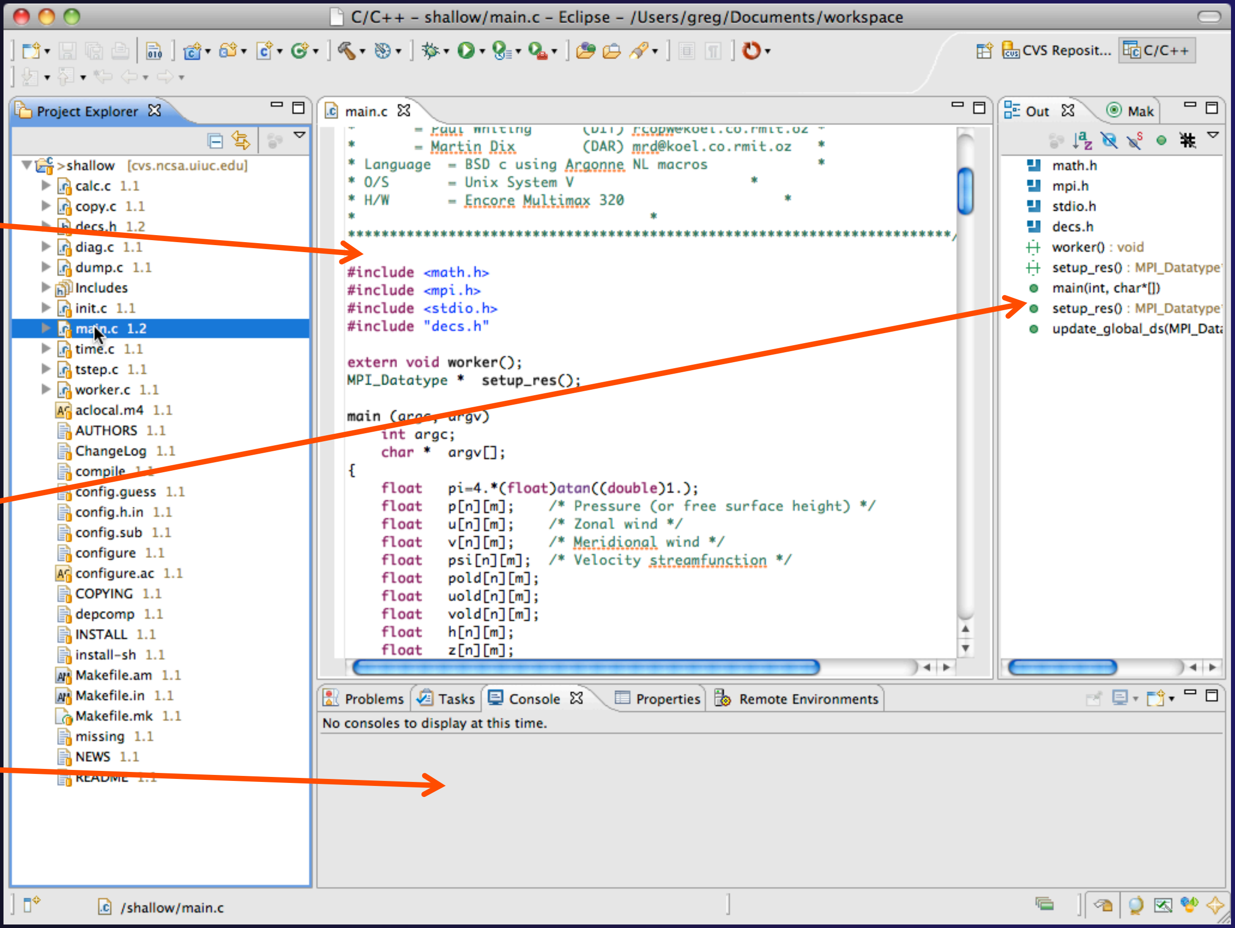


- Save the changes by using Command/Ctrl-S or **File>Save**

- Undo last change using **Command/Ctrl Z**

Editor and Outline View

- Double-click on source file
- Editor will open in main view
- Outline view is shown for file in editor
- Console shows results of build, local runs, etc.



C/C++ – shallow/main.c – Eclipse – /Users/greg/Documents/workspace

Project Explorer

```
shallow [cvs.ncsa.uiuc.edu]
  calc.c 1.1
  copy.c 1.1
  decs.h 1.2
  diag.c 1.1
  dump.c 1.1
  Includes
  init.c 1.1
  main.c 1.2
  time.c 1.1
  tstep.c 1.1
  worker.c 1.1
  aclocal.m4 1.1
  AUTHORS 1.1
  ChangeLog 1.1
  compile 1.1
  config.guess 1.1
  config.h.in 1.1
  config.sub 1.1
  configure 1.1
  configure.ac 1.1
  COPYING 1.1
  depcomp 1.1
  INSTALL 1.1
  install-sh 1.1
  Makefile.am 1.1
  Makefile.in 1.1
  Makefile.mk 1.1
  missing 1.1
  NEWS 1.1
  README 1.1
```

main.c

```
*        - Paul Whiting        (DIT) rcopw@koel.co.rmit.oz *
*        - Martin Dix          (DAR) mrd@koel.co.rmit.oz   *
* Language  - BSD c using Argonne NL macros                *
* O/S       - Unix System V                        *
* H/W       - Encore Multimax 320                      *
*                                *
*************************************************************************/

#include <math.h>
#include <mpi.h>
#include <stdio.h>
#include "decs.h"

extern void worker();
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];
```

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_Data
```

Problems | Tasks | Console | Properties | Remote Environments

No consoles to display at this time.

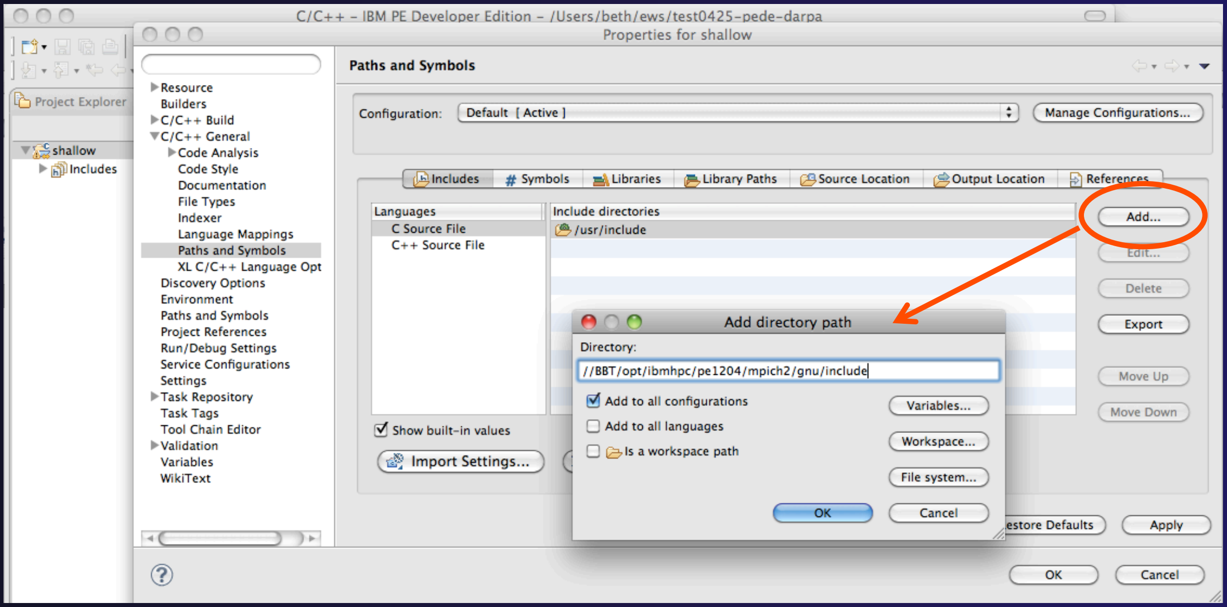
/shallow/main.c

Include Paths

- In order for editor and build features to work properly, Eclipse needs to know where your include files are located
- The build environment on the remote host knows your include files etc., but we must tell Eclipse so that indexing, search, completion, etc. will know where things are

- Open Project Properties
- Expand C/C++ General
- Select Paths and Symbols
- Select **Add…**
- A UNC-style path specifies //<connection>/<path>
- Example //forge/usr/mpi/gcc/openmpi-1.4.2/include

- Select **OK, OK**
- Will ask to rebuild..
- Select **Yes**


C/C++ - IBM PE Developer Edition - /Users/beth/ews/test0425-pede-darpa

Properties for shallow

Paths and Symbols

Configuration: Default [Active]

Languages: C Source File, C++ Source File

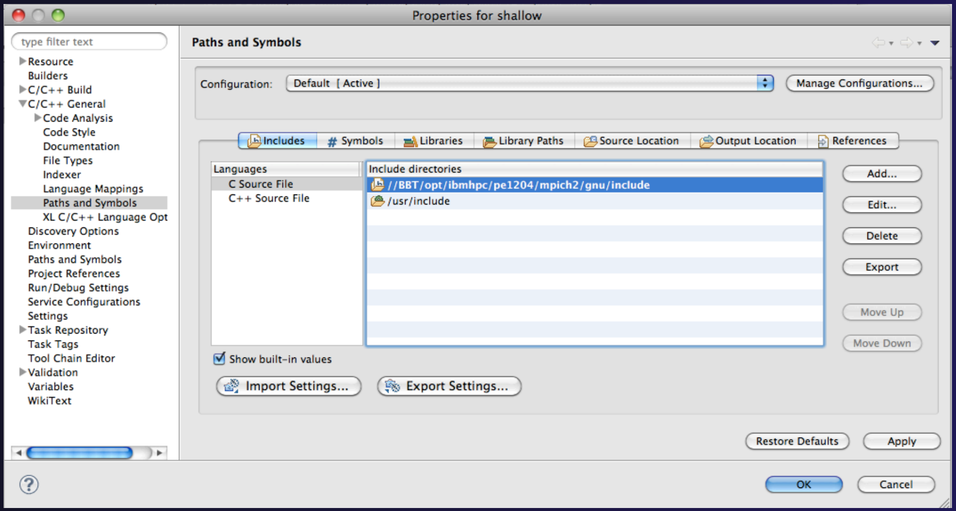
Include directories: /usr/include

Add directory path

Directory: //BBT/opt/ibmhpc/pe1204/mpich2/gnu/include

☑ Add to all configurations
☐ Add to all languages
☐ Is a workspace path

☑ Show built-in values


Properties for shallow

Paths and Symbols

Configuration: Default [Active]

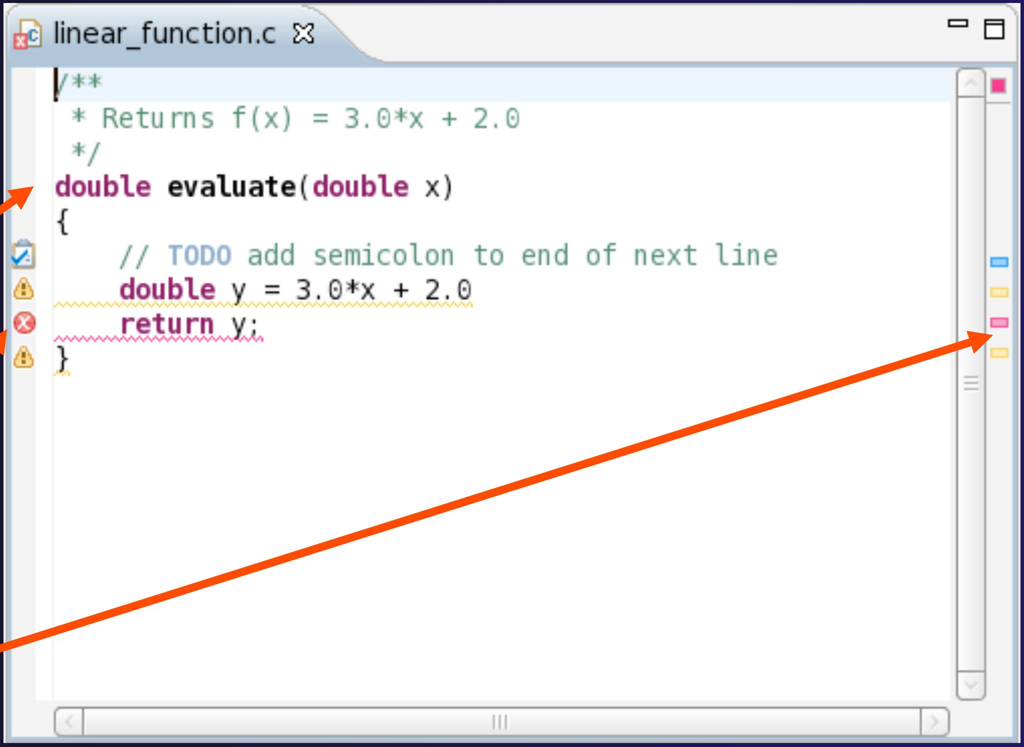
Languages: C Source File, C++ Source File

Include directories:
//BBT/opt/ibmhpc/pe1204/mpich2/gnu/include
/usr/include

☑ Show built-in values

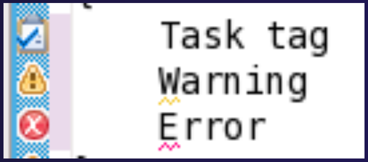
Source Code Editors & Markers

- 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:



Code Analysis (Codan)

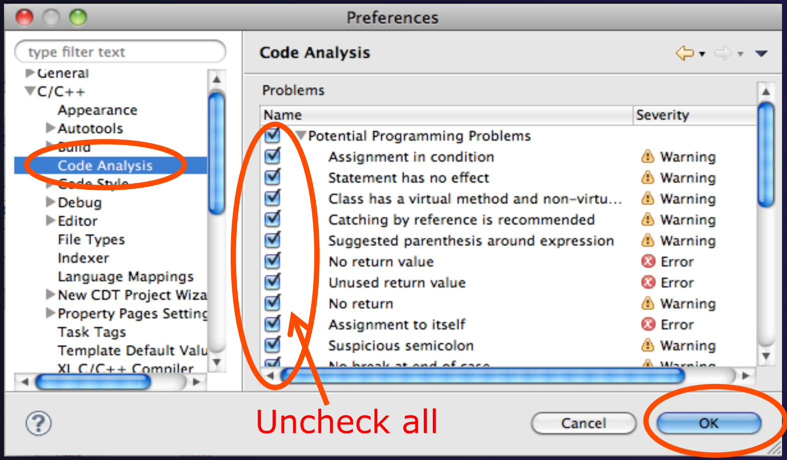
- If you see bug icons in the editor marker bar, they are likely suggestions from Codan
  - If you set up include paths properly, you may not see any
- Code checkers can flag possible errors, even if code is technically correct
- To turn them off, use Preferences

Window > Preferences or Mac: Eclipse > Preferences

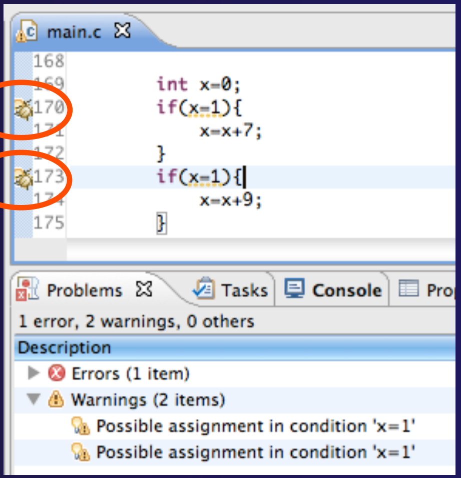
C/C++ > Code Analysis

and uncheck all problems

- Select OK to close Preferences



Name	Severity
☑ Potential Programming Problems	
☑ Assignment in condition	Warning
☑ Statement has no effect	Warning
☑ Class has a virtual method and non-virtu…	Warning
☑ Catching by reference is recommended	Warning
☑ Suggested parenthesis around expression	Warning
☑ No return value	Error
☑ Unused return value	Error
☑ No return	Warning
☑ Assignment to itself	Error
☑ Suspicious semicolon	Warning
☑ No break at end of case	Warning



```
        int x=0;
        if(x=1){
            x=x+7;
        }
        if(x=1){
            x=x+9;
        }
```

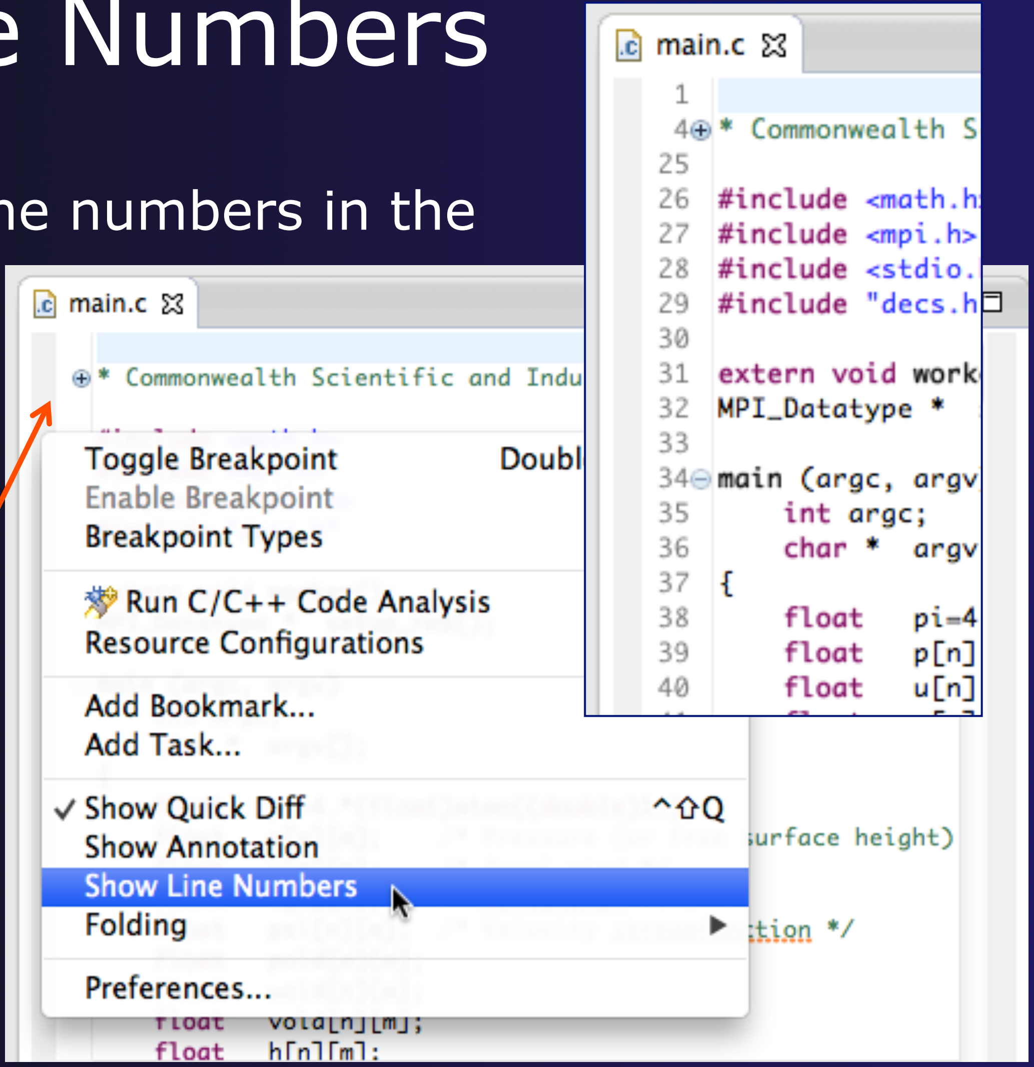
1 error, 2 warnings, 0 others

Description
- Errors (1 item)
- Warnings (2 items)
  - Possible assignment in condition 'x=1'
  - Possible assignment in condition 'x=1'

- If icons don't disappear: Right mouse on Project > Run C/C++ Code Analysis

Line Numbers

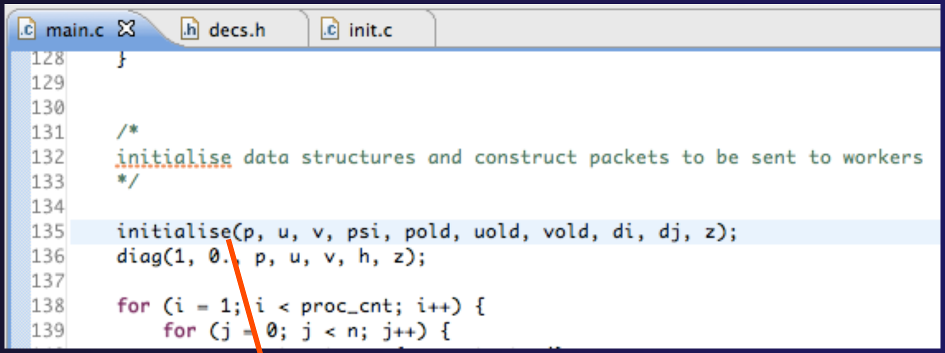
- Text editors can show line numbers in the left column
- To turn on line numbering:
  - Right-mouse click in the editor marker bar
  - Click on **Show Line Numbers**



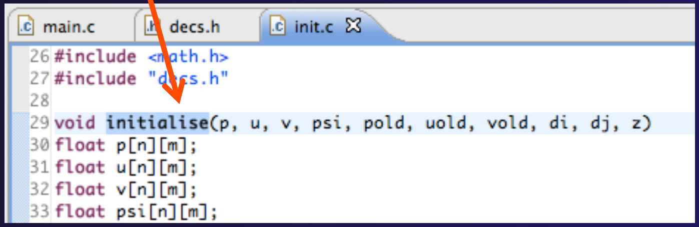
```
1
4⊕* Commonwealth S
25
26 #include <math.h
27 #include <mpi.h>
28 #include <stdio.
29 #include "decs.h
30
31 extern void work
32 MPI_Datatype *
33
34⊖main (argc, argv
35     int argc;
36     char *  argv
37 {
38     float   pi=4
39     float   p[n]
40     float   u[n]
```

Navigating to Other Files

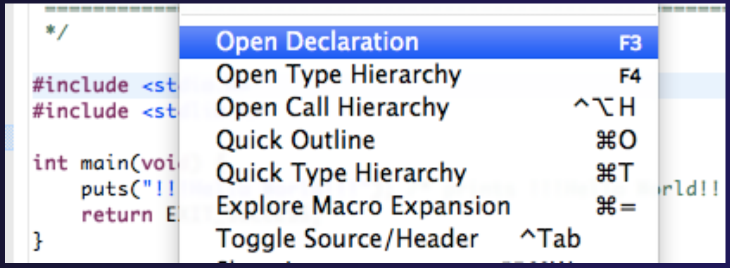
- On demand hyperlink
  - In main.c line 135:
  - Hold down Command/Ctrl key e.g. on call to `initialise`
  - Click on `initialise` to navigate to its definition in the header file (Exact key combination depends on your OS)
  - E.g. Command/Ctrl and click on initialise
- Open declaration
  - Right-click and select **Open Declaration** will also open the file in which the element is declared
  - E.g. in main.c line 29 right-click on decs.h and select **Open Declaration**



```
128     }
129
130
131     /*
132     initialise data structures and construct packets to be sent to workers
133     */
134
135     initialise(p, u, v, psi, pold, uold, vold, di, dj, z);
136     diag(1, 0., p, u, v, h, z);
137
138     for (i = 1; i < proc_cnt; i++) {
139         for (j = 0; j < n; j++) {
```



```
26 #include <math.h>
27 #include "decs.h"
28
29 void initialise(p, u, v, psi, pold, uold, vold, di, dj, z)
30 float p[n][m];
31 float u[n][m];
32 float v[n][m];
33 float psi[n][m];
```

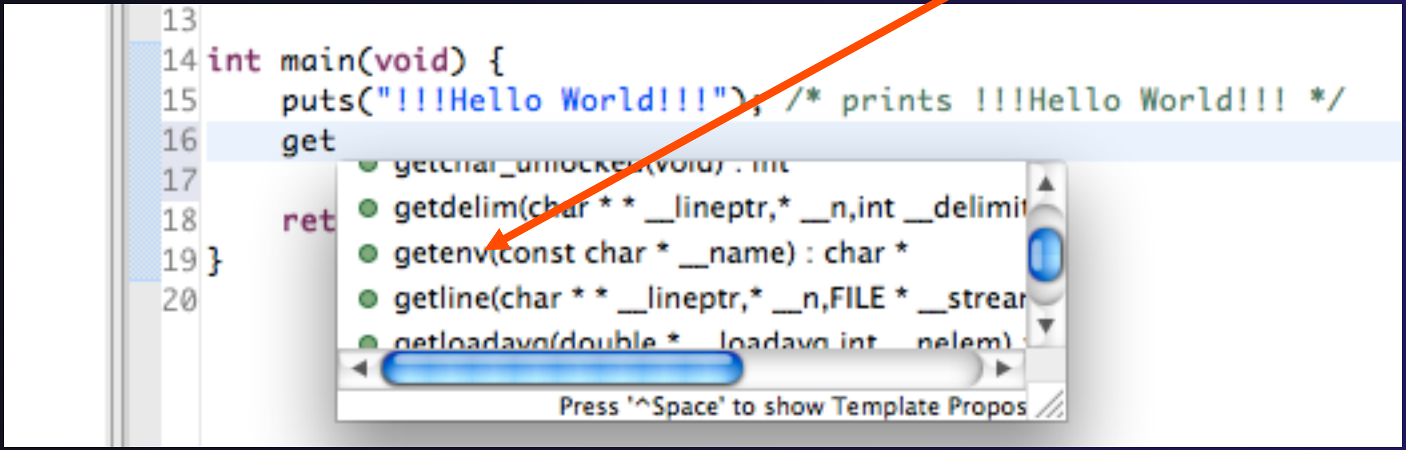


```
*/

#include <st
#include <st

int main(voi
    puts("!!                    rld!!
    return E
}
```

Note: may need to left-click before right-click works

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

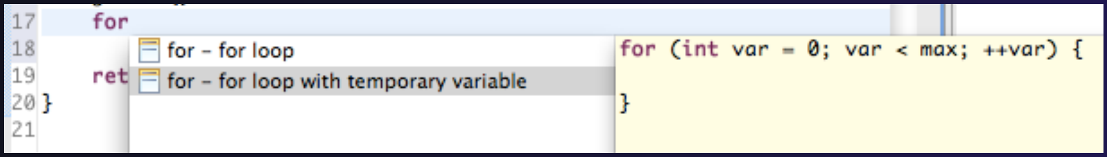


```
int main(void) {
    puts("!!!Hello World!!!"); /* prints !!!Hello World!!! */
    get

    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

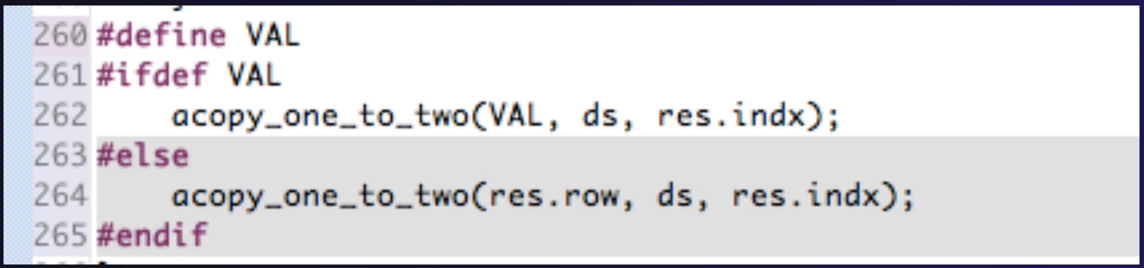


```
for (int var = 0; var < max; ++var) {

}
```

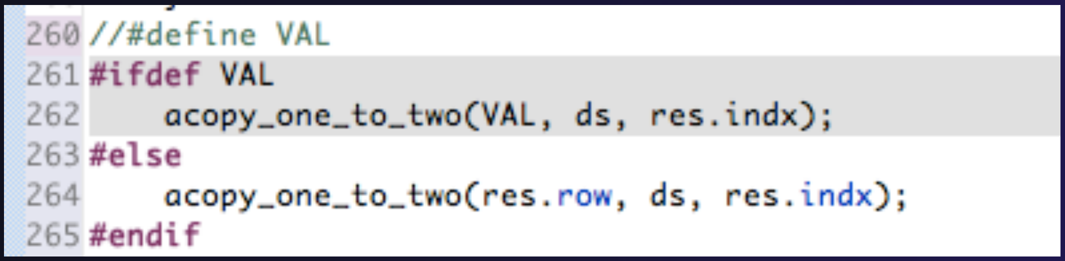
More info on code templates later

Inactive code

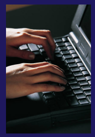
✦ Inactive code will appear grayed out in the CDT editor



```
#define VAL
#ifdef VAL
    acopy_one_to_two(VAL, ds, res.indx);
#else
    acopy_one_to_two(res.row, ds, res.indx);
#endif
```

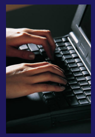


```
//#define VAL
#ifdef VAL
    acopy_one_to_two(VAL, ds, res.indx);
#else
    acopy_one_to_two(res.row, ds, res.indx);
#endif
```



Exercise

1. Open an editor by double clicking on a source file in the **Project Explorer**
2. Use the **Outline View** to navigate to a different line in the editor
3. Back in main.c, turn on line numbering
4. In main.c, ctrl-click on line 99, master_packet, should navigate to its definition in the file
5. In worker.c, line 132, hover over variable p to see info



Optional Exercise

1. Type “for”, then activate content assist
   - Select the **for loop with temporary variable** template, insert it, then modify the template variable
   - Surround the code you just inserted with “#if 0” and “#endif” and observe that it is marked as inactive
   - Save the file
2. What do these keys do in the editor?
   - Ctrl+L; Ctrl+Shift+P (do it near some brackets)
   - Ctrl+Shift+/;
   - Ctrl+Shift+Y and Ctrl+Shift+X (do it on a word or variable name e.g.)
   - Alt+Down; Alt+Up
3. To make sure you didn’t do any damage,
   - Select any source files you changed and do rightmouse > replace with .. Latest from HEAD
   - Observe that your changes are gone.

MPI Programming

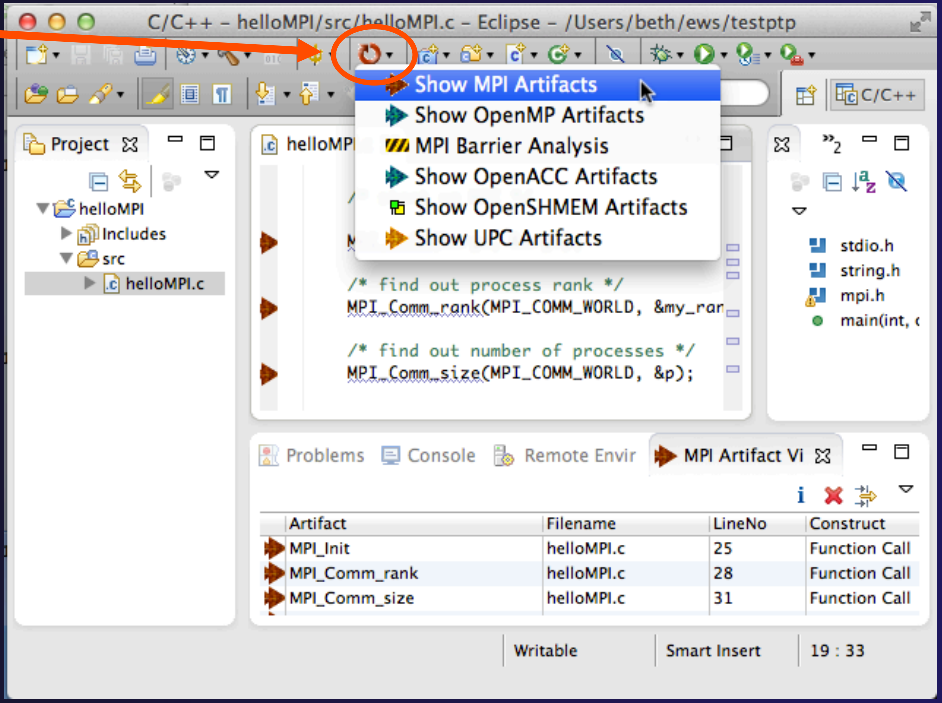
- Objective
  - Learn about MPI features for your source files
- Contents
  - Using Editor features for MPI
  - MPI Help features
  - Finding MPI Artifacts
  - MPI New Project Wizards
  - MPI Barrier Analysis

MPI-Specific Features

- PTP's Parallel Language Development Tools (PLDT) has several features specifically for developing MPI code
  - Show MPI Artifacts
  - Code completion / Content Assist
  - Context Sensitive Help for MPI
  - Hover Help
  - MPI Templates in the editor
  - MPI Barrier Analysis
- PLDT has similar features for OpenMP, UPC, OpenSHMEM, OpenACC

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
- Select **Show MPI Artifacts**
- Works on local and remote files

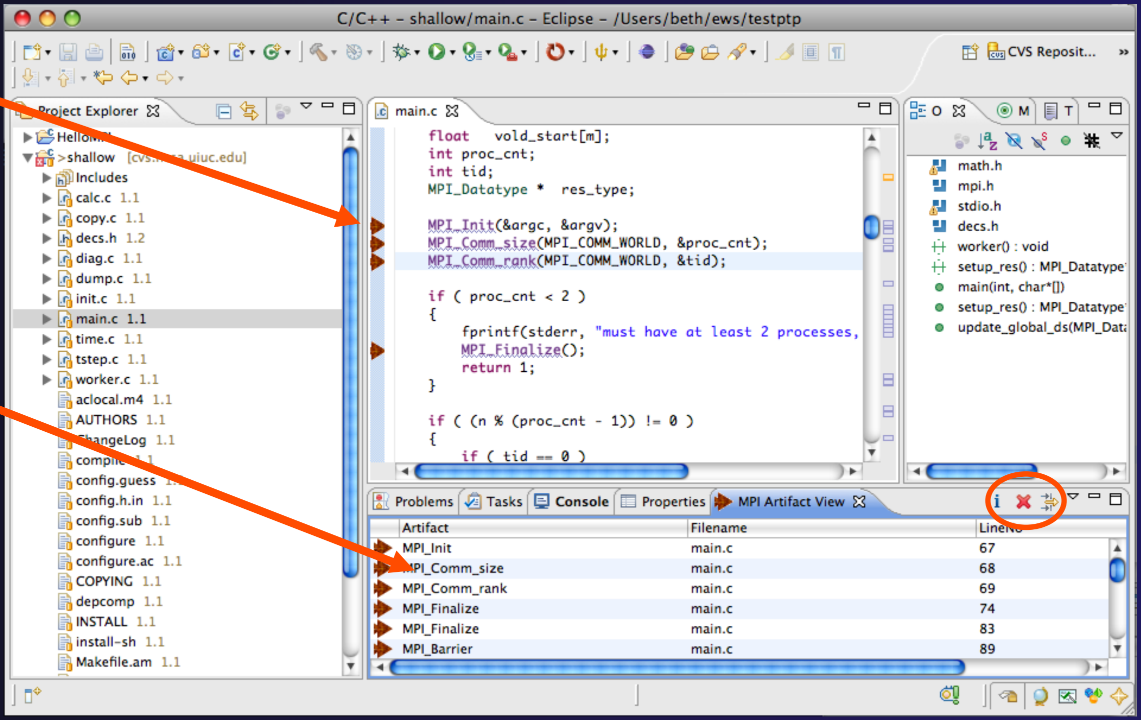


```
/* find out process rank */
MPI_Comm_rank(MPI_COMM_WORLD, &my_rar

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

Artifact	Filename	LineNo	Construct
MPI_Init	helloMPI.c	25	Function Call
MPI_Comm_rank	helloMPI.c	28	Function Call
MPI_Comm_size	helloMPI.c	31	Function Call

MPI Artifact View

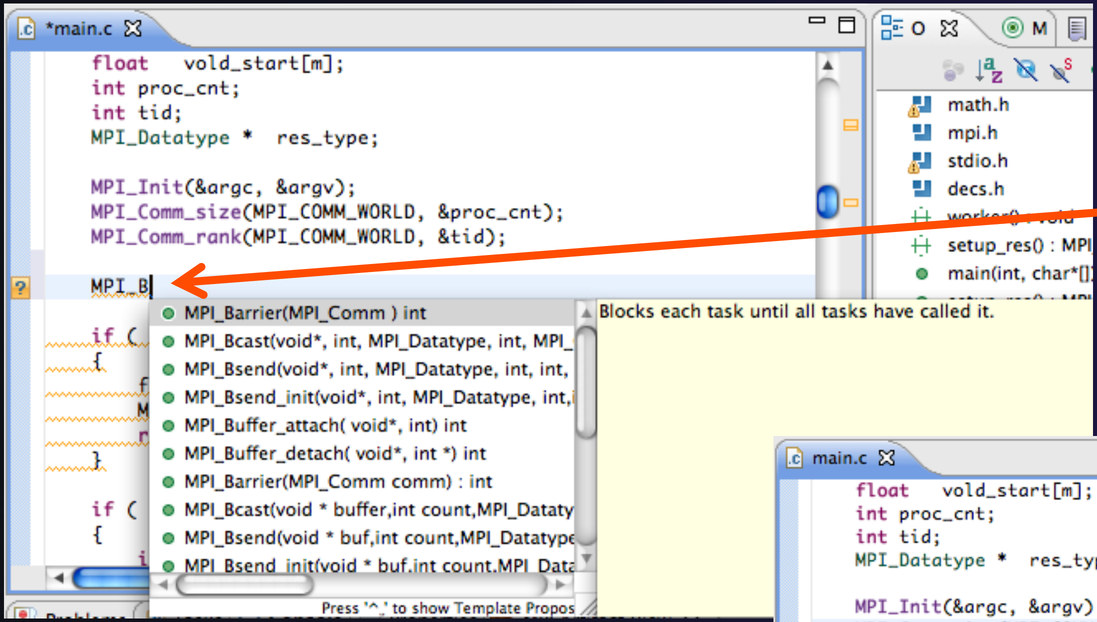
- Markers indicate the location of artifacts in editor
- The **MPI Artifact View** lists the type and location of each artifact
- Navigate to source code line by double-clicking on the artifact
- Run the analysis on another file (or entire project!) and its markers will be added to the view
- Click on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



```
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

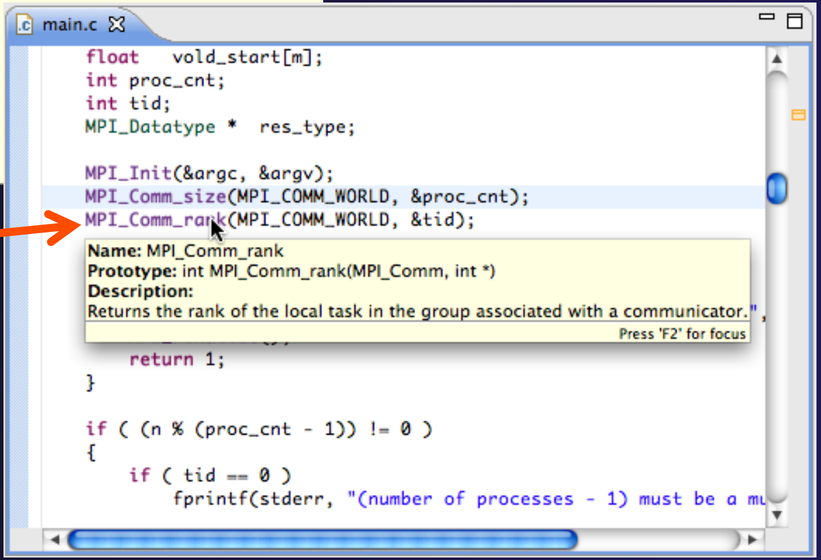


```
float   vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype *  res_type;

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

MPI_B
```

```
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
```

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>



```
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);
```

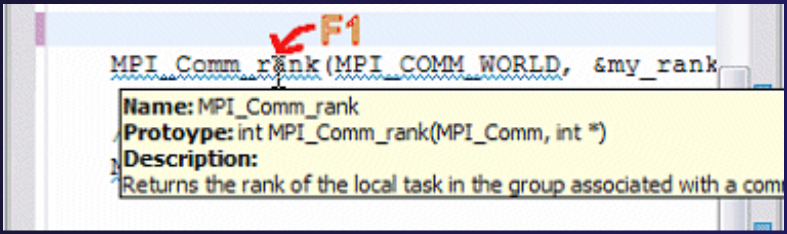
Name: MPI_Comm_rank
Prototype: int MPI_Comm_rank(MPI_Comm, int *)
Description:
Returns the rank of the local task in the group associated with a communicator.

```
    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

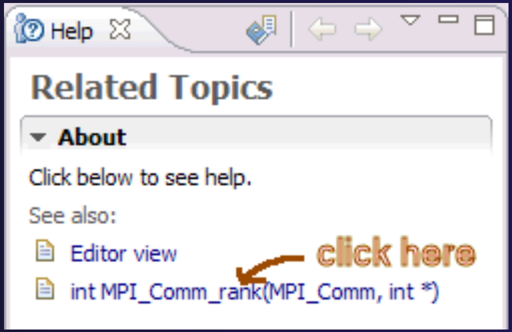
Context Sensitive Help

- 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



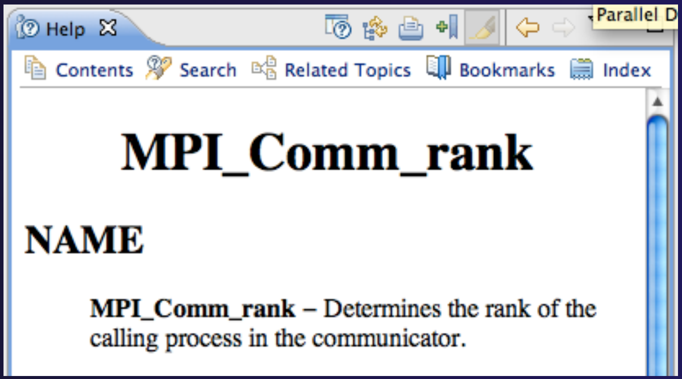
```
MPI_Comm_rank(MPI_COMM_WORLD, &my_rank
```

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



Click below to see help.
See also:
Editor view
int MPI_Comm_rank(MPI_Comm, int *)

Some special info has been added for MPI APIs



MPI_Comm_rank

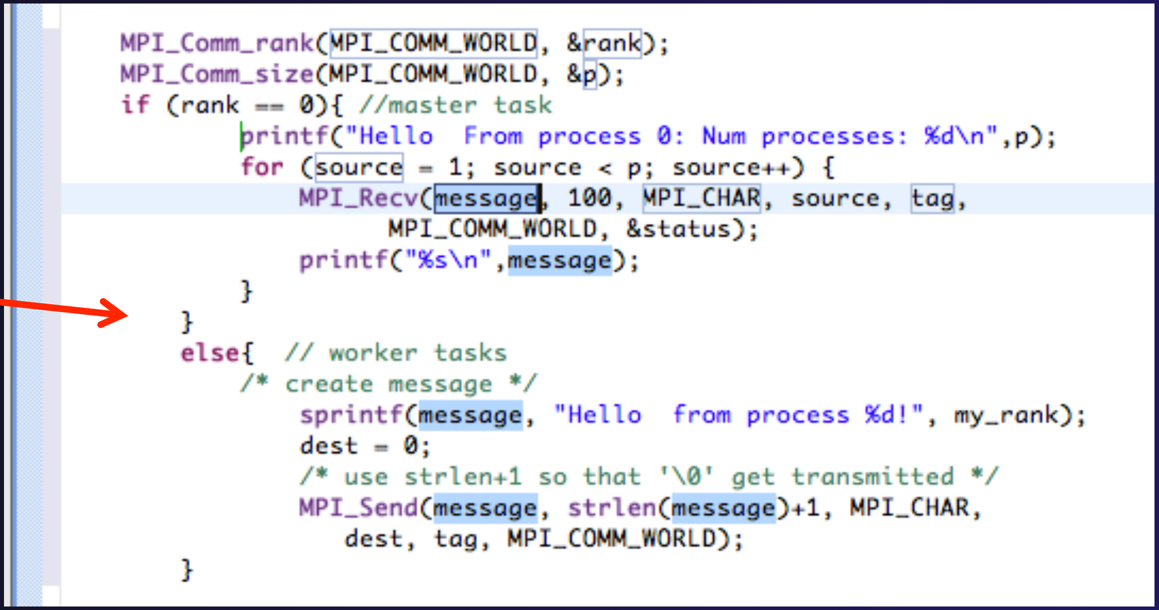
NAME

MPI_Comm_rank – Determines the rank of the calling process in the communicator.

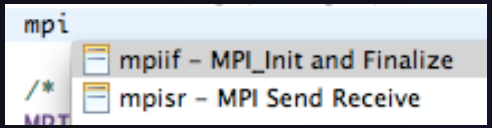
MPI Templates

- 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



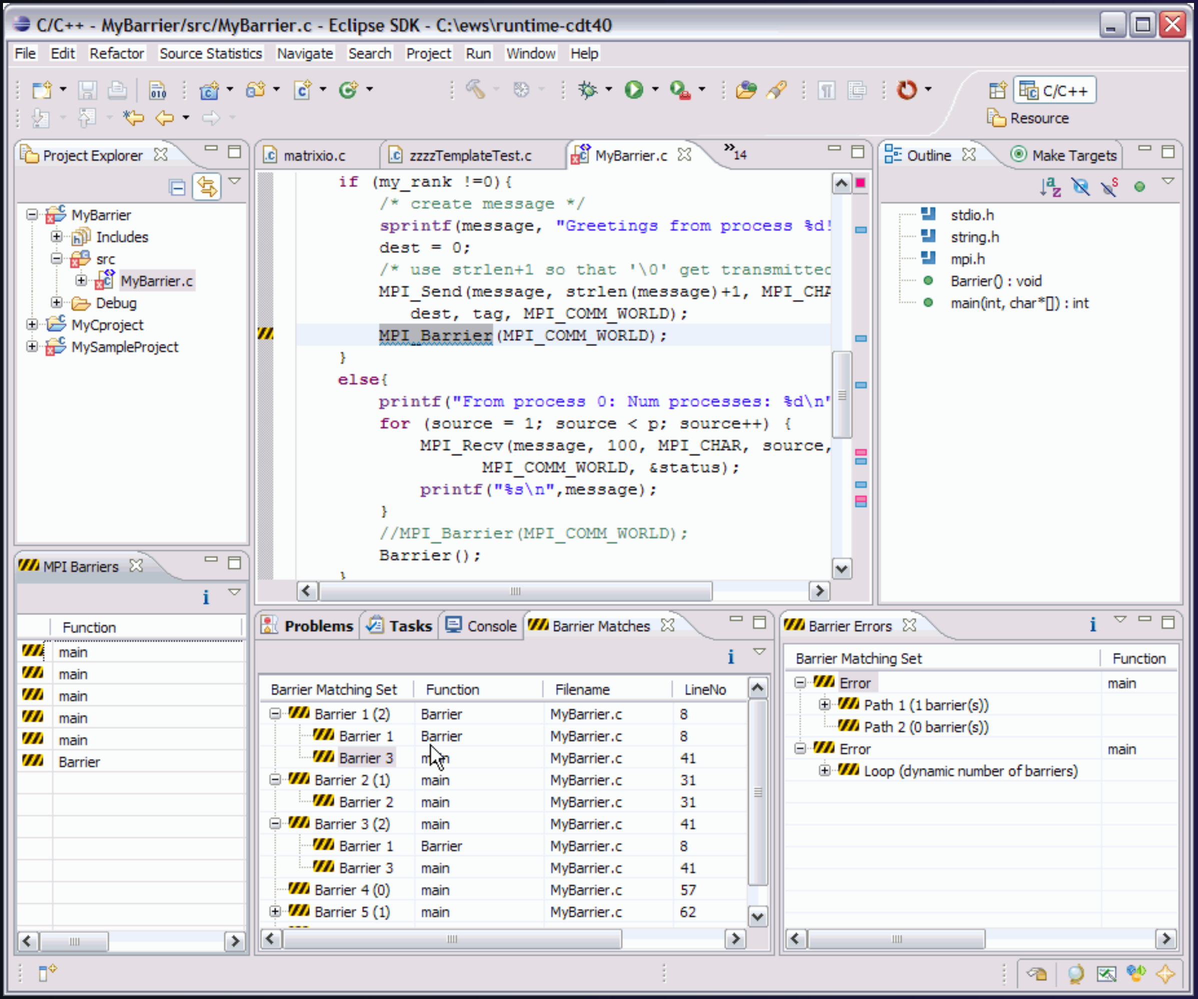
```
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &p);
if (rank == 0){ //master task
        printf("Hello  From process 0: Num processes: %d\n",p);
        for (source = 1; source < p; source++) {
            MPI_Recv(message, 100, MPI_CHAR, source, tag,
                  MPI_COMM_WORLD, &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);
    }
```



```
mpi
  mpiif – MPI_Init and Finalize
  mpisr – MPI Send Receive
```

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

MPI Barrier Analysis



C/C++ - MyBarrier/src/MyBarrier.c - Eclipse SDK - C:\ews\runtime-cdt40

Project Explorer: MyBarrier, Includes, src, MyBarrier.c, Debug, MyCproject, MySampleProject

```
    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();
```

Outline: stdio.h, string.h, mpi.h, Barrier() : void, main(int, char*[]) : int

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)	

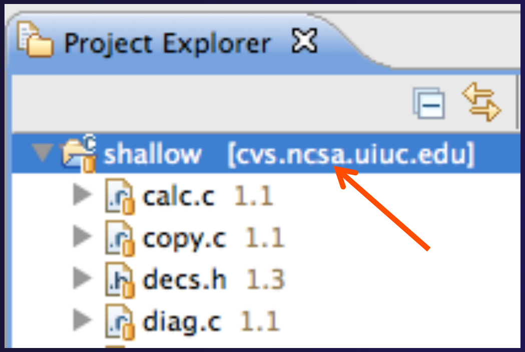
- Verify barrier synchronization in C/MPI programs
- For verified programs, lists barrier statements that synchronize together (match)
- For synchronization errors, reports counter example that illustrates and explains the error

Local files only

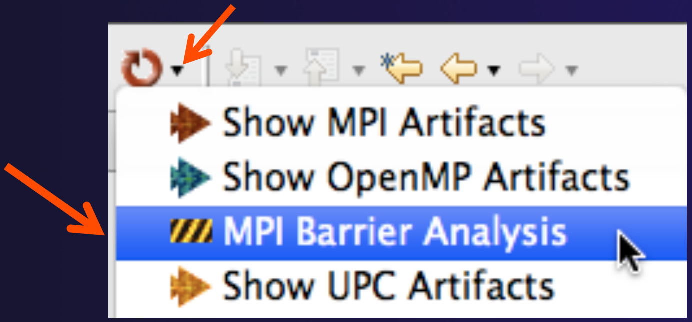
MPI Barrier Analysis (2)

Run the Analysis:

- In the Project Explorer, select the project (or directory, or file) to analyze

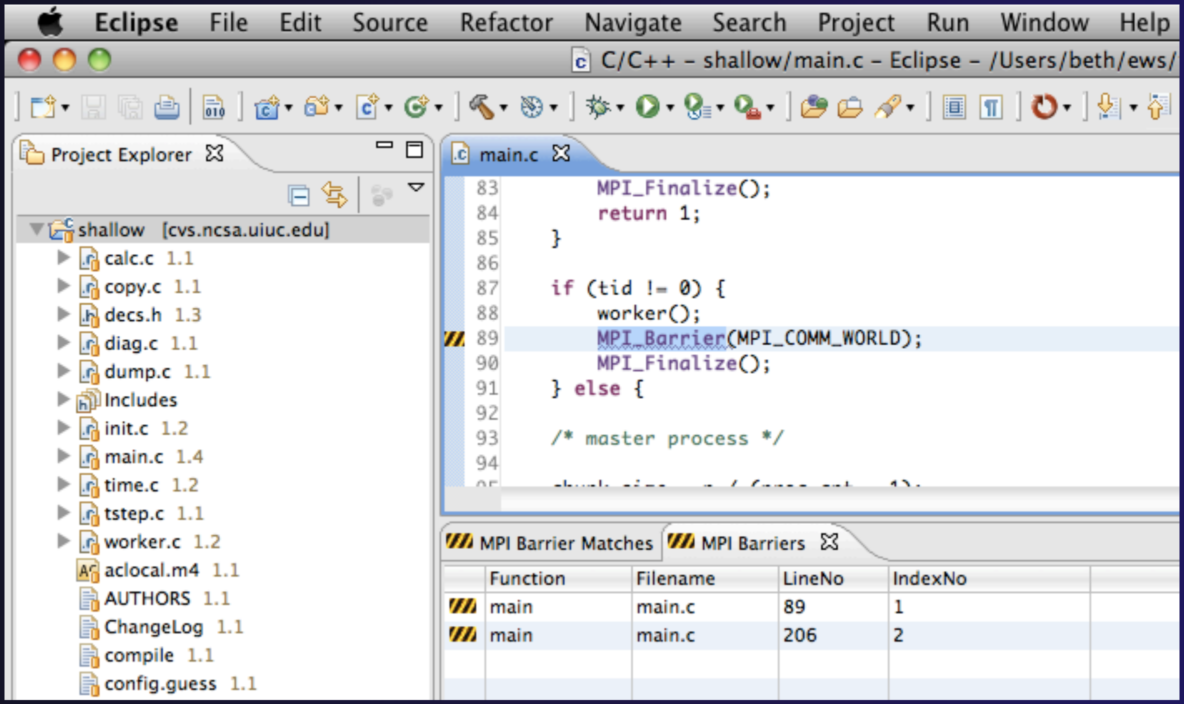


- Select the MPI Barrier Analysis action in the pull-down menu



MPI Barrier Analysis (3)

- No Barrier Errors are found (no pop-up indicating error)
- Two barriers are found



Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
C/C++ – shallow/main.c – Eclipse – /Users/beth/ews/
Project Explorer
main.c

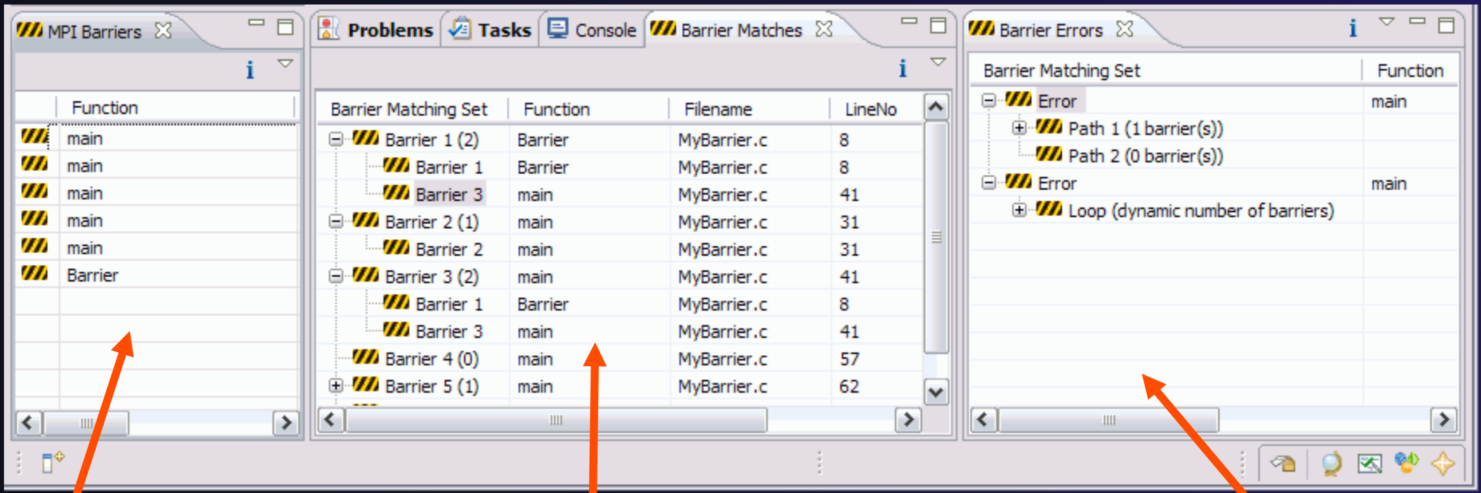
- shallow [cvs.ncsa.uiuc.edu]
  - calc.c 1.1
  - copy.c 1.1
  - decs.h 1.3
  - diag.c 1.1
  - dump.c 1.1
  - Includes
  - init.c 1.2
  - main.c 1.4
  - time.c 1.2
  - tstep.c 1.1
  - worker.c 1.2
  - aclocal.m4 1.1
  - AUTHORS 1.1
  - ChangeLog 1.1
  - compile 1.1
  - config.guess 1.1

```
83          MPI_Finalize();
84          return 1;
85      }
86
87      if (tid != 0) {
88          worker();
89          MPI_Barrier(MPI_COMM_WORLD);
90          MPI_Finalize();
91      } else {
92
93      /* master process */
94
```

MPI Barrier Matches | MPI Barriers

Function	Filename	LineNo	IndexNo
main	main.c	89	1
main	main.c	206	2

MPI Barrier Analysis Views



MPI Barriers

	Function
	main
	main
	main
	main
	main
	Barrier

Problems Tasks Console 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

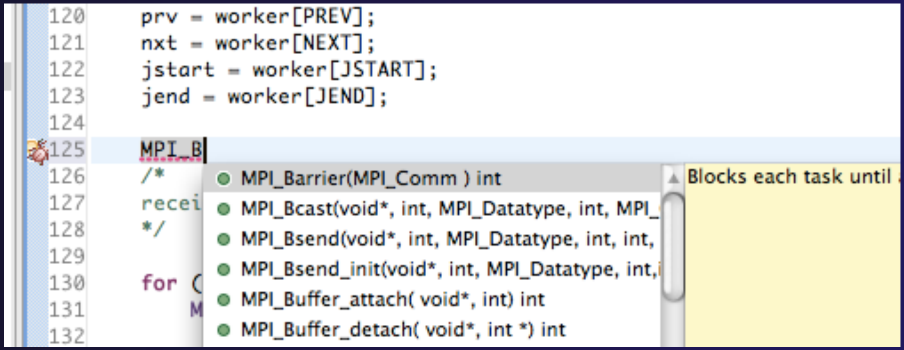
Groups barriers that match together in a barrier set – all processes must go through a barrier in the set to prevent a deadlock

Barrier Errors view

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

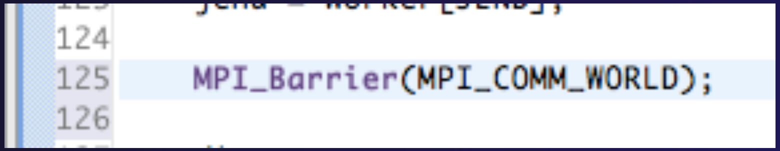
Barrier Errors

- Let's cause a barrier mismatch error
- Open worker.c in the editor by double-clicking on it in Project Explorer
- At about line 125, enter a barrier:
  - Type MPI_B
  - Hit Ctl-space
  - Select MPI_Barrier
  - Add communicator arg MPI_COMM_WORLD and closing semicolon



```
120     prv = worker[PREV];
121     nxt = worker[NEXT];
122     jstart = worker[JSTART];
123     jend = worker[JEND];
124
125     MPI_B
126     /*
127     recei
128     */
129
130     for (
131         M
132
```

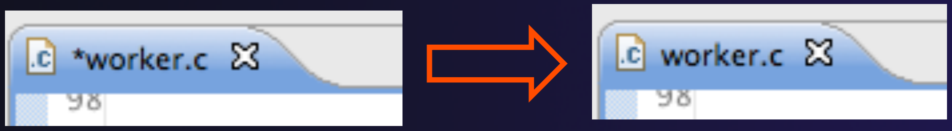
MPI_Barrier(MPI_Comm) int
MPI_Bcast(void*, int, MPI_Datatype, int, MPI_
MPI_Bsend(void*, int, MPI_Datatype, int, int,
MPI_Bsend_init(void*, int, MPI_Datatype, int,
MPI_Buffer_attach(void*, int) int
MPI_Buffer_detach(void*, int *) int
Blocks each task until



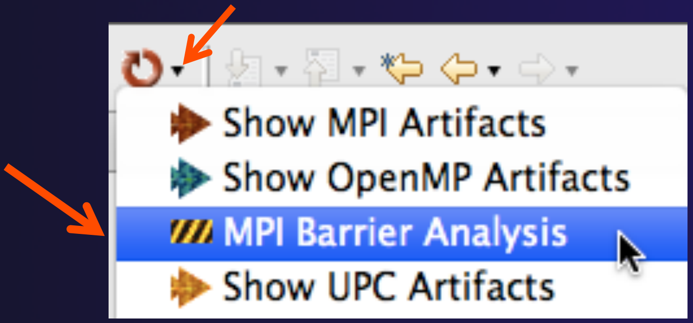
```
124
125     MPI_Barrier(MPI_COMM_WORLD);
126
```

Barrier Errors (2)

- Save the file
  - Ctl-S (Mac Command-S) or File > Save
  - Tab should lose asterisk indicating file saved

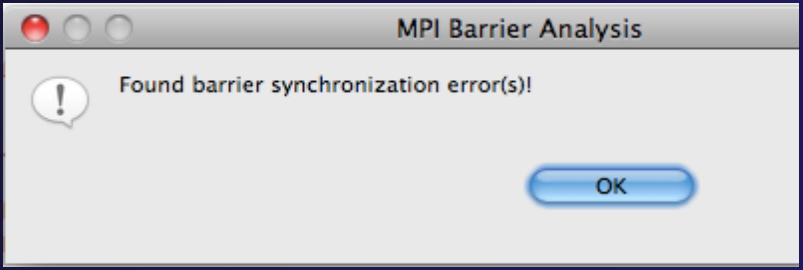


- Run barrier analysis on shallow project again
  - Select shallow project in Project Explorer first



Barrier Errors (3)

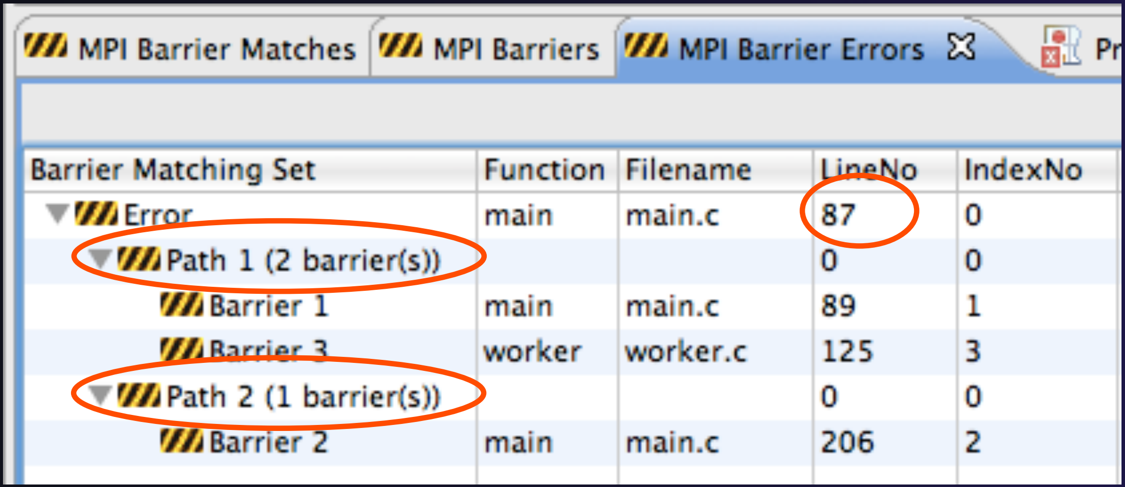
- Barrier Error is found
- Hit OK to dismiss dialog



MPI Barrier Analysis

Found barrier synchronization error(s)!

- Code diverges on line 87
  - One path has 2 barriers, other has 1

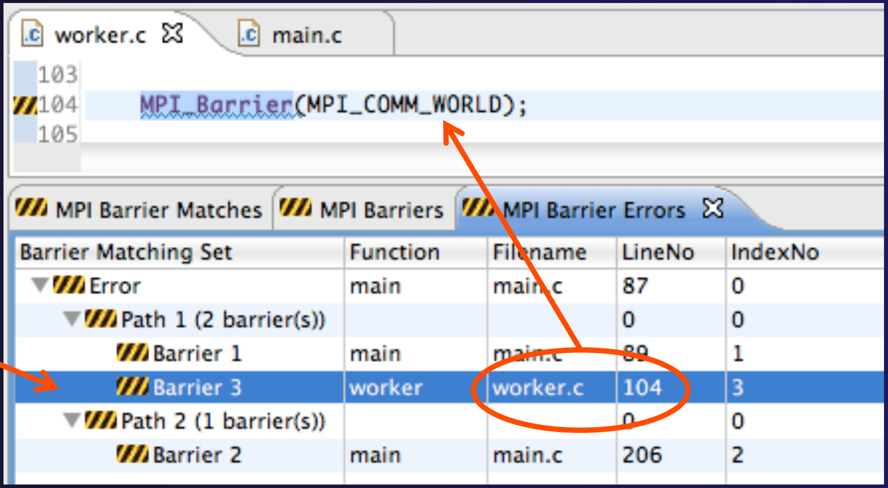


Barrier Matching Set	Function	Filename	LineNo	IndexNo
Error	main	main.c	87	0
Path 1 (2 barrier(s))			0	0
Barrier 1	main	main.c	89	1
Barrier 3	worker	worker.c	125	3
Path 2 (1 barrier(s))			0	0
Barrier 2	main	main.c	206	2

Double-click on a row in Barrier Errors view to find the line it references in the code

Fix Barrier Error

- Fix the Barrier Error before continuing
- Double-click on the barrier in worker.c to quickly navigate to it



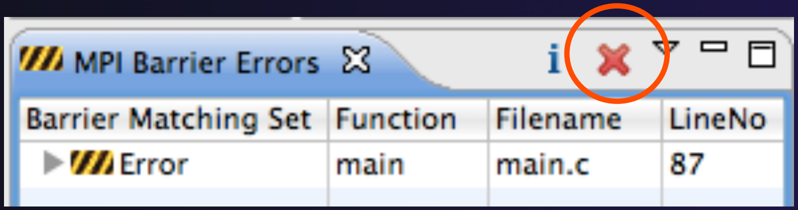
```
MPI_Barrier(MPI_COMM_WORLD);
```

Barrier Matching Set	Function	Filename	LineNo	IndexNo
Error	main	main.c	87	0
Path 1 (2 barrier(s))			0	0
Barrier 1	main	main.c	89	1
Barrier 3	worker	worker.c	104	3
Path 2 (1 barrier(s))			0	0
Barrier 2	main	main.c	206	2

- Remove the line and save the file
- Re-run the barrier analysis to check that it has been fixed

Remove Barrier Markers

- Run Barrier Analysis again to remove the error
- Remove the Barrier Markers via the "X" in one of the MPI Barrier views

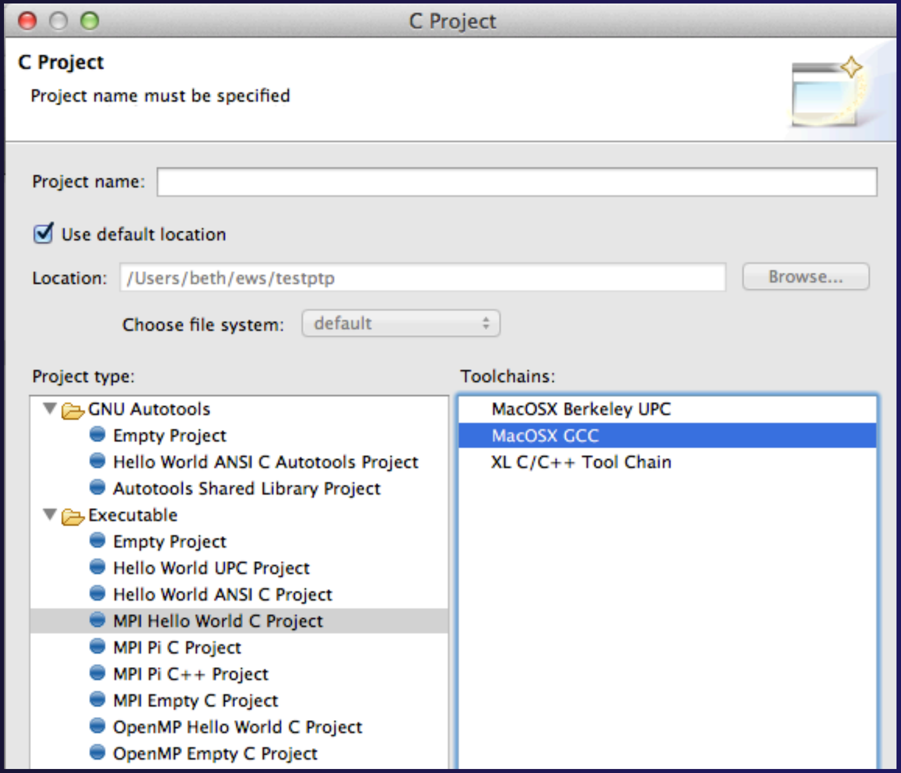


Barrier Matching Set	Function	Filename	LineNo
Error	main	main.c	87

MPI New Project Wizards

- Quick way to make a simple MPI project
- File > New > C Project

- "MPI Hello World" is good for trying out Eclipse for MPI


C Project

C Project
Project name must be specified

Project name:

[✓] Use default location

Location: /Users/beth/ews/testptp

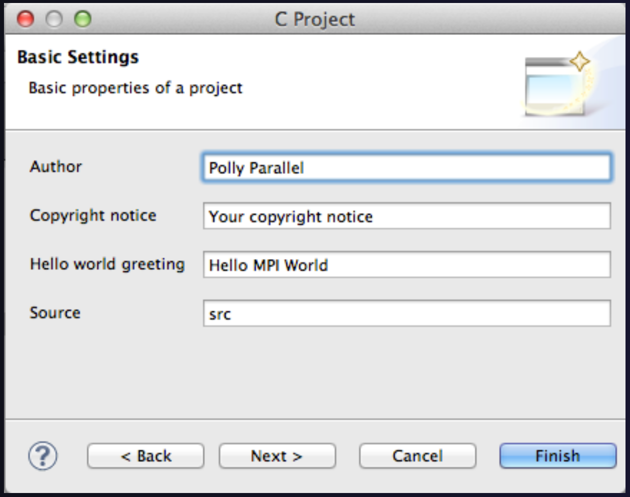
Choose file system: default

Project type:
- GNU Autotools
  - Empty Project
  - Hello World ANSI C Autotools Project
  - Autotools Shared Library Project
- Executable
  - Empty Project
  - Hello World UPC 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

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

MPI New Project Wizards (2)

- Next> and fill in (optional) Basic Settings



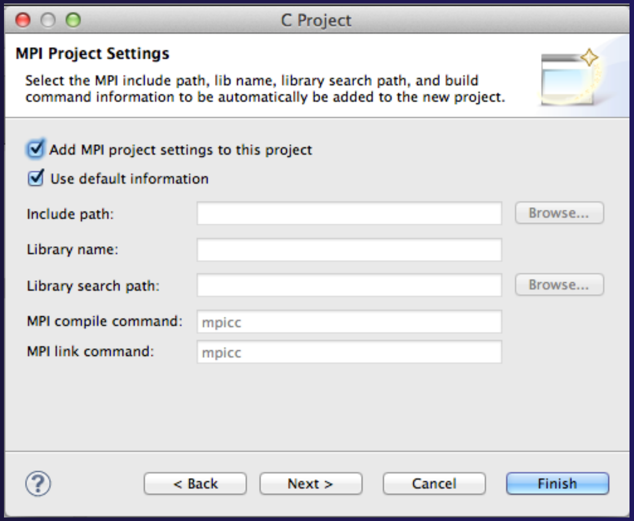
C Project

Basic Settings

Basic properties of a project

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

- Next> and fill in MPI Project Settings
- Include path set in MPI Preferences can be added to project



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.

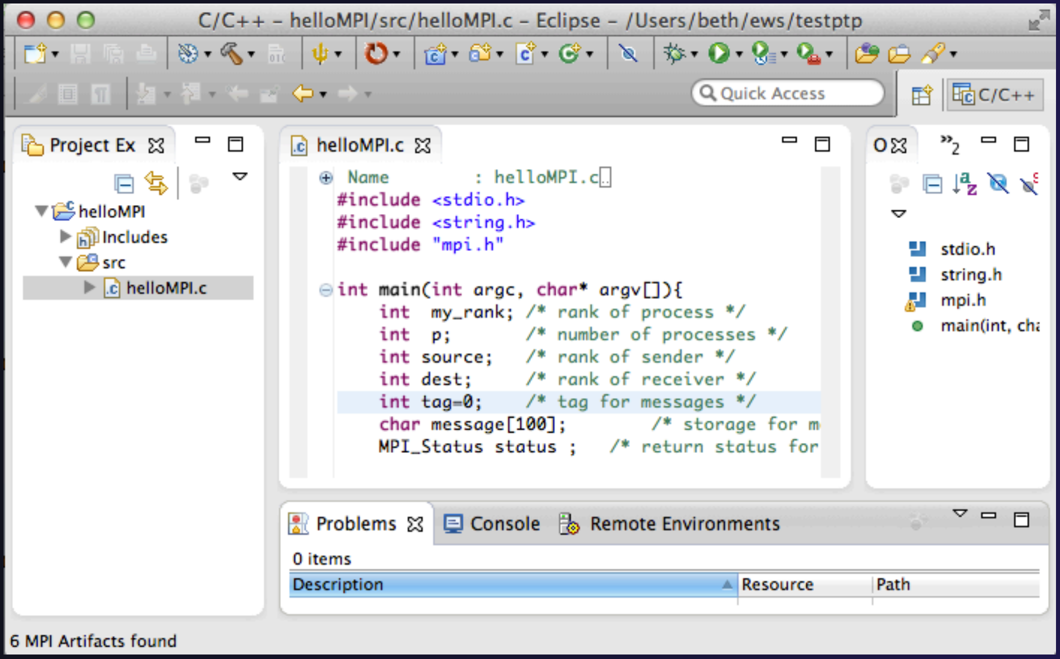
✓ Add MPI project settings to this project

✓ Use default information

Include path:	
Library name:	
Library search path:	
MPI compile command:	mpicc
MPI link command:	mpicc

MPI New Project Wizards (3)

- Select **Finish** and "MPI Hello World" project is created


C/C++ – helloMPI/src/helloMPI.c – Eclipse – /Users/beth/ews/testptp

```
   Name        : helloMPI.c
#include <stdio.h>
#include <string.h>
#include "mpi.h"

int main(int argc, char* argv[]){
    int  my_rank; /* rank of process */
    int  p;       /* number of processes */
    int source;   /* rank of sender */
    int dest;     /* rank of receiver */
    int tag=0;    /* tag for messages */
    char message[100];        /* storage for m
    MPI_Status status ;   /* return status for
```

Problems | Console | Remote Environments

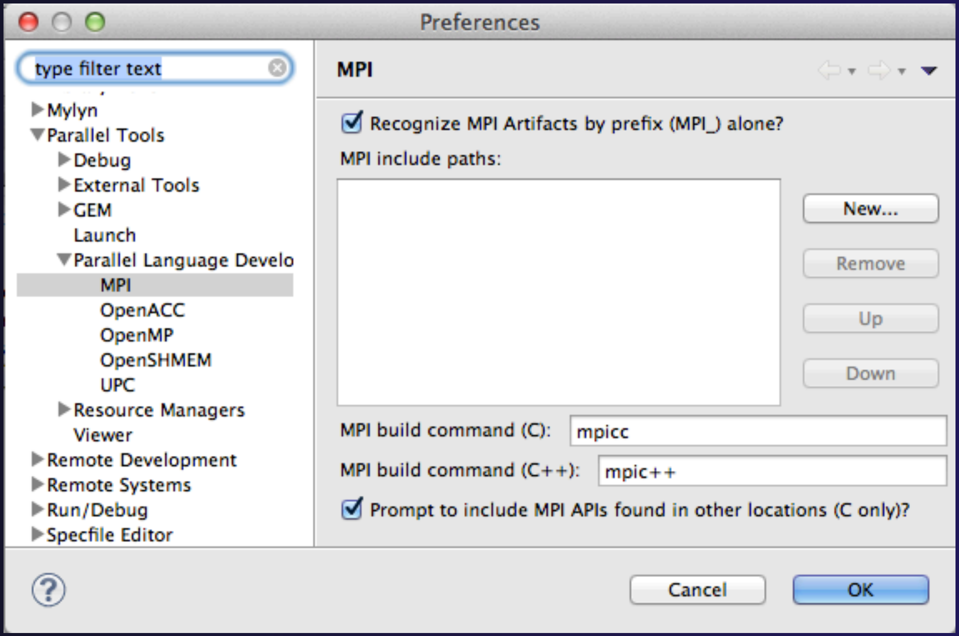
0 items

Description	Resource	Path

6 MPI Artifacts found

MPI Preferences

- ✦ Settings for MPI New Project wizards
- ✦ MPI Include paths, if set in MPI Preferences, are added in MPI New Project Wizard



Preferences

MPI

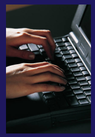
☑ Recognize MPI Artifacts by prefix (MPI_) alone?

MPI include paths:

MPI build command (C): mpicc

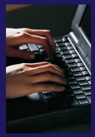
MPI build command (C++): mpic++

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



Exercise

1. Find MPI artifacts in ‘shallow’ project
   - Locate all the MPI communication (send/receive) calls
2. Use content assist to add an api call
   - E.g., Type MPI_S, hit ctl-space
3. Use hover help
4. Use a template to add an MPI code template
   - On a new line, type mpisr and ctl-space…



Optional Exercise

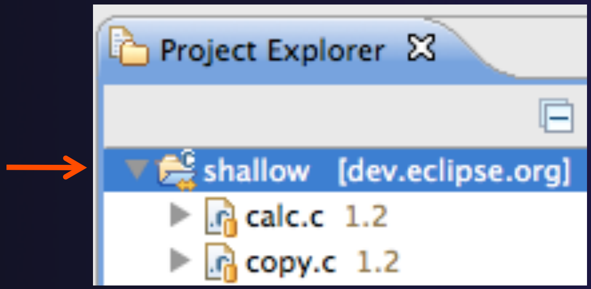
1. Insert an MPI_Barrier function call into one of your source files using content assist
   - E.g. Line 125 of worker.c
2. Save the file
3. Run Barrier Analysis on the project
4. Locate the source of the barrier error and remove the statement
5. Re-run barrier analysis to observe that the problem has been fixed

Building a Project

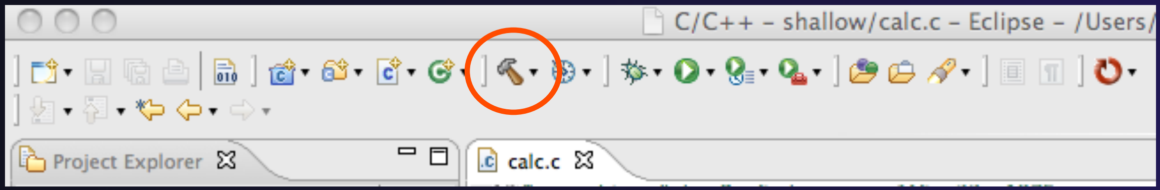
- Objective
  - Learn how to build an MPI program on a remote system
- Contents
  - How to change build settings
  - How to start a build and view build output
  - How to clean and rebuild a project
  - How to create build targets

Starting the Build

- Select the project in Project Explorer

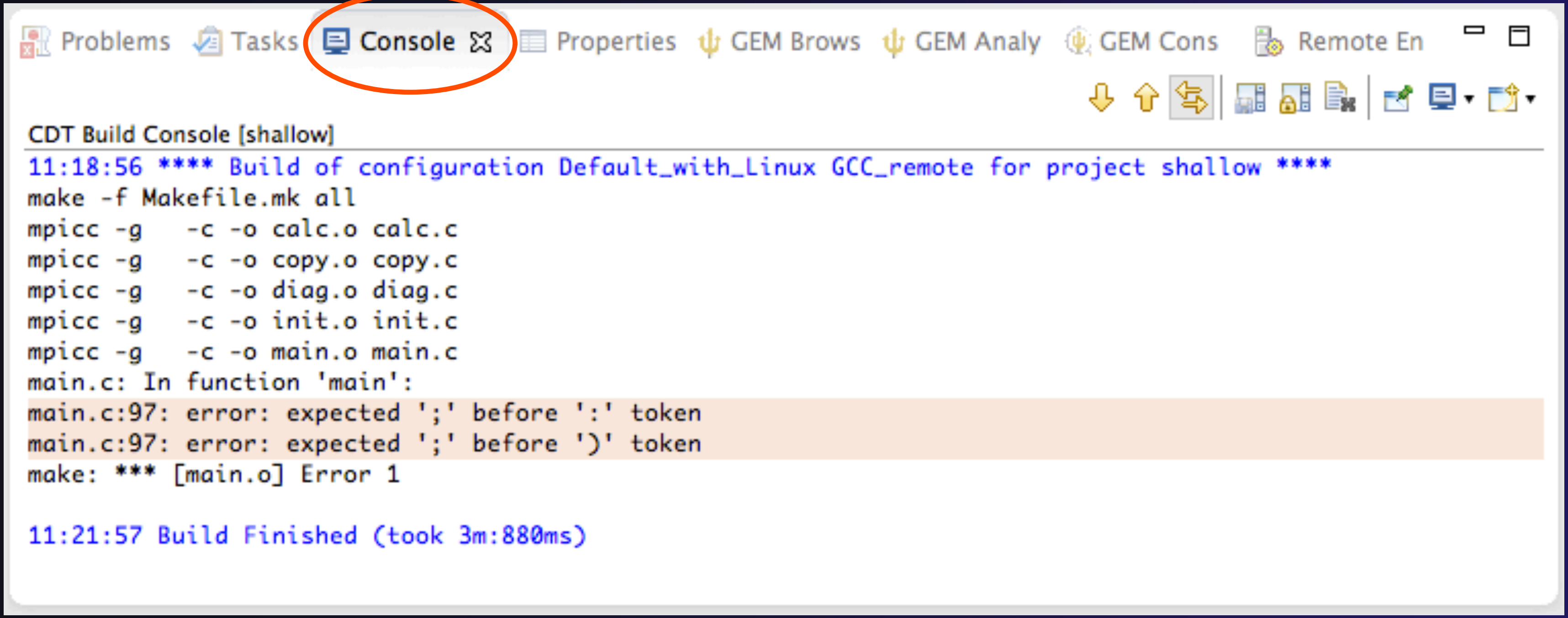


- Click on the hammer button in toolbar to run a build using the active build configuration



Viewing the Build Output

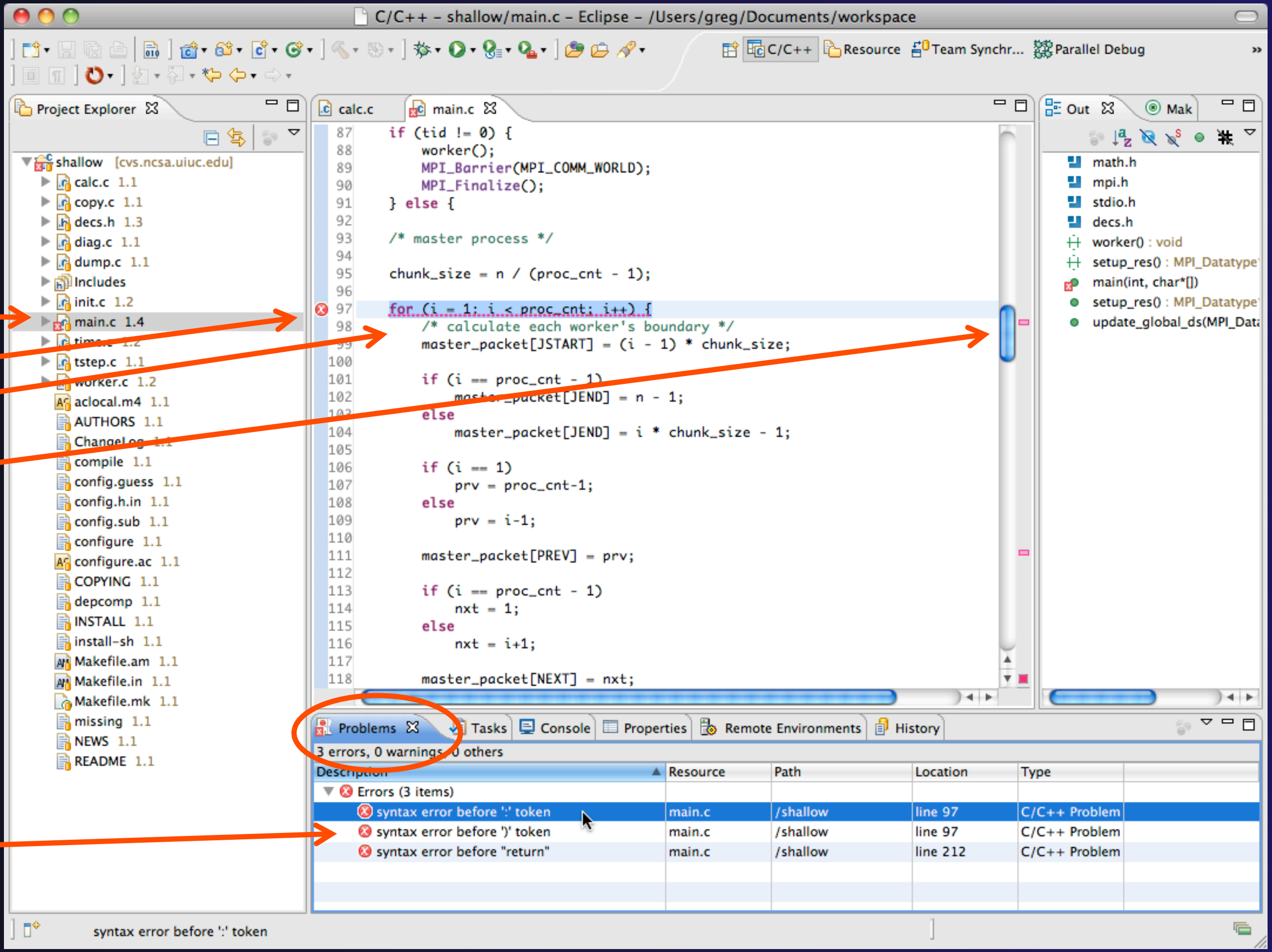
✦ Build output will be visible in console



```
CDT Build Console [shallow]
11:18:56 **** Build of configuration Default_with_Linux GCC_remote for project shallow ****
make -f Makefile.mk all
mpicc -g   -c -o calc.o calc.c
mpicc -g   -c -o copy.o copy.c
mpicc -g   -c -o diag.o diag.c
mpicc -g   -c -o init.o init.c
mpicc -g   -c -o main.o main.c
main.c: In function 'main':
main.c:97: error: expected ';' before ':' token
main.c:97: error: expected ';' before ')' token
make: *** [main.o] Error 1

11:21:57 Build Finished (took 3m:880ms)
```

Build Problems

- Build problems will be shown in a variety of ways
  - Marker on file
  - Marker on editor line
  - Line is highlighted
  - Marker on overview ruler
  - Listed in the **Problems view**
- Double-click on line in **Problems view** to go to location of error in the editor



```
 87    if (tid != 0) {
 88        worker();
 89        MPI_Barrier(MPI_COMM_WORLD);
 90        MPI_Finalize();
 91    } else {
 92
 93    /* master process */
 94
 95    chunk_size = n / (proc_cnt - 1);
 96
 97    for (i = 1: i < proc_cnt; i++) {
 98        /* calculate each worker's boundary */
 99        master_packet[JSTART] = (i - 1) * chunk_size;
100
101        if (i == proc_cnt - 1)
102            master_packet[JEND] = n - 1;
103        else
104            master_packet[JEND] = i * chunk_size - 1;
105
106        if (i == 1)
107            prv = proc_cnt-1;
108        else
109            prv = i-1;
110
111        master_packet[PREV] = prv;
112
113        if (i == proc_cnt - 1)
114            nxt = 1;
115        else
116            nxt = i+1;
117
118        master_packet[NEXT] = nxt;
```

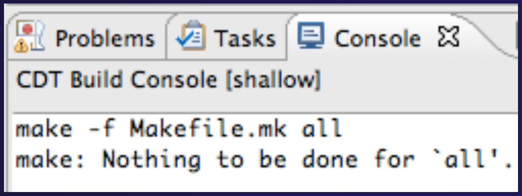
3 errors, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (3 items)				
syntax error before ':' token	main.c	/shallow	line 97	C/C++ Problem
syntax error before ')' token	main.c	/shallow	line 97	C/C++ Problem
syntax error before "return"	main.c	/shallow	line 212	C/C++ Problem

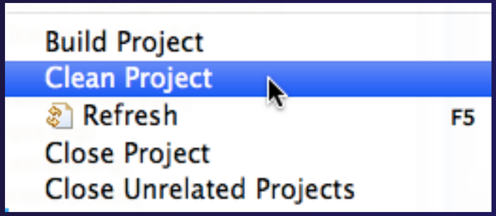
syntax error before ':' token

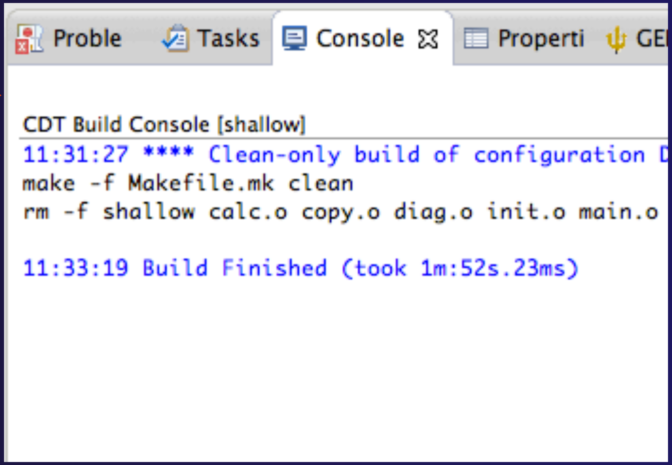
Forcing a Rebuild

- If no changes have been made, `make` doesn't think a build is needed
- In **Project Explorer**, right click on project
  - Select **Clean Project**
  - Choose to clean all projects, or select specific projects to clean
- Build console will display results
- Rebuild project by clicking on build button again



```
make -f Makefile.mk all
make: Nothing to be done for `all'.
```

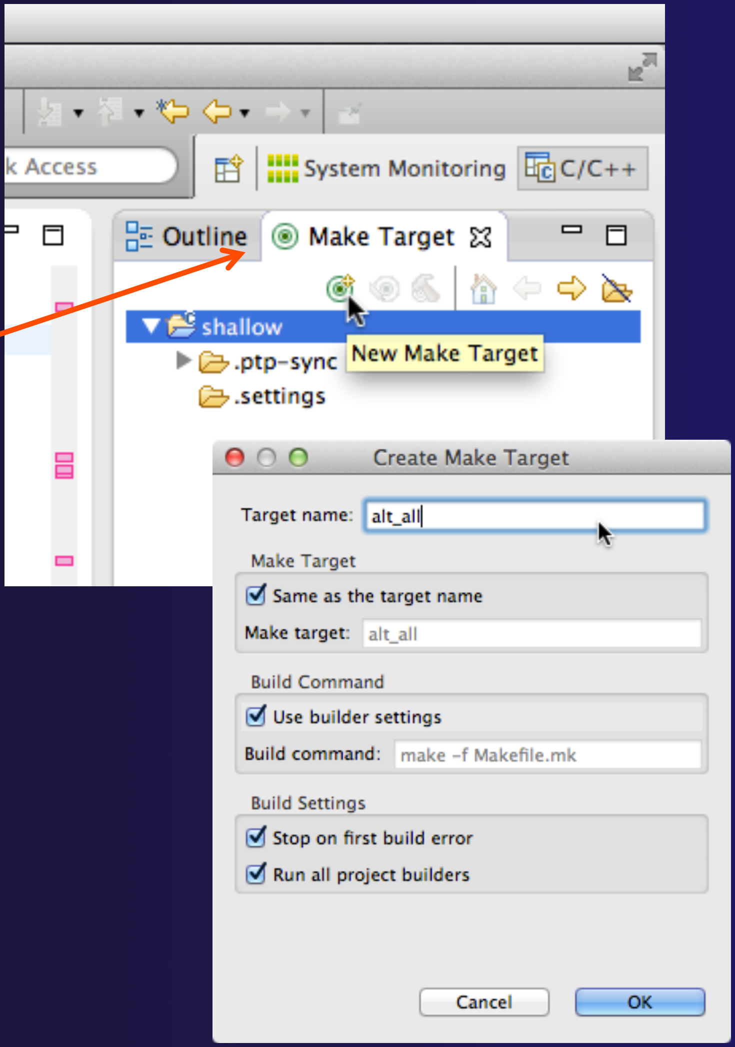




```
11:31:27 **** Clean-only build of configuration D
make -f Makefile.mk clean
rm -f shallow calc.o copy.o diag.o init.o main.o

11:33:19 Build Finished (took 1m:52s.23ms)
```

Creating Make Targets

- By default
  - The build button will run “make all”
  - Cleaning a project will run “make clean”
- Sometimes, other build targets are required
- Open **Make Targets** view
- Select project and click on **New Make Target** button
- Enter new target name
- Modify build command if desired
- New target will appear in view
- Double click on target to activate



Target name: alt_all

Make Target

☑ Same as the target name

Make target: alt_all

Build Command

☑ Use builder settings

Build command: make -f Makefile.mk

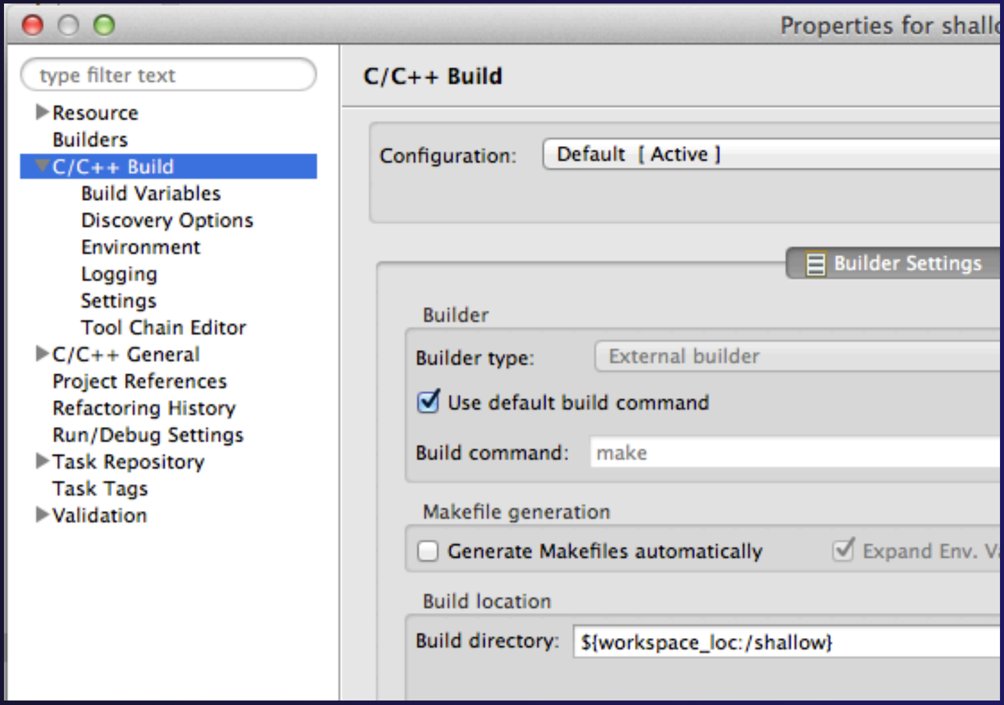
Build Settings

☑ Stop on first build error

☑ Run all project builders

Build Configuration

- The build configuration is specified in the project properties
- Open the properties by right-clicking on the project name in the **Project Explorer** view and selecting **Properties** (bottom of the context menu list)
- **C/C++ Build**
  - Configure the build command
  - Default is “make” but this can be changed to anything
- **C/C++ Build > Settings**
  - Binary and Error parser selection
  - Tool Chain settings (managed projects only)
- **C/C++ Build > Environment**
  - Modify/add environment variables passed to build
- **C/C++ Build > Logging**
  - Enable/disable build logging



Configuration: Default [Active]

Builder type: External builder

☑ Use default build command

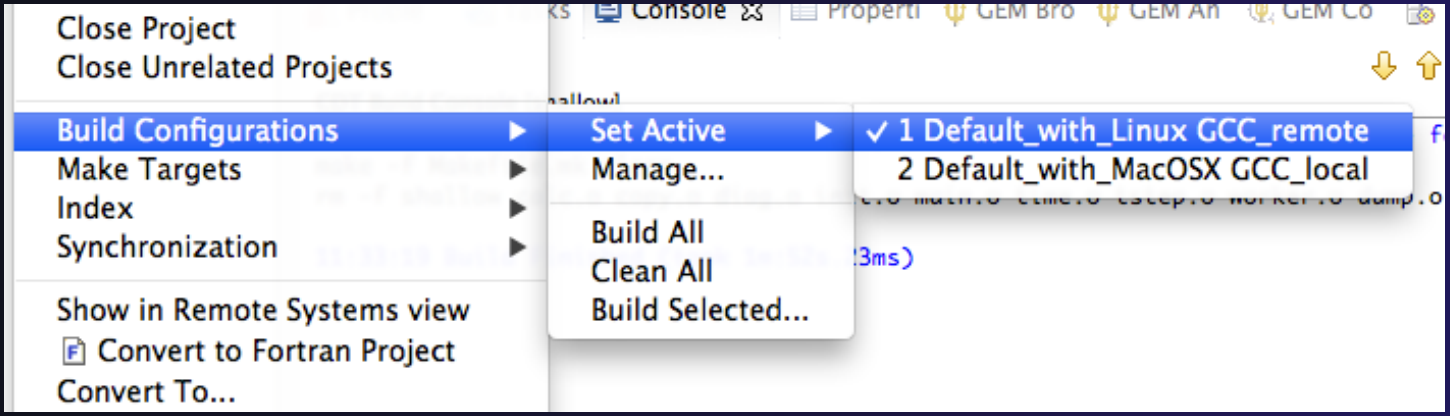
Build command: make

☐ Generate Makefiles automatically ☑ Expand Env. V

Build directory: ${workspace_loc:/shallow}

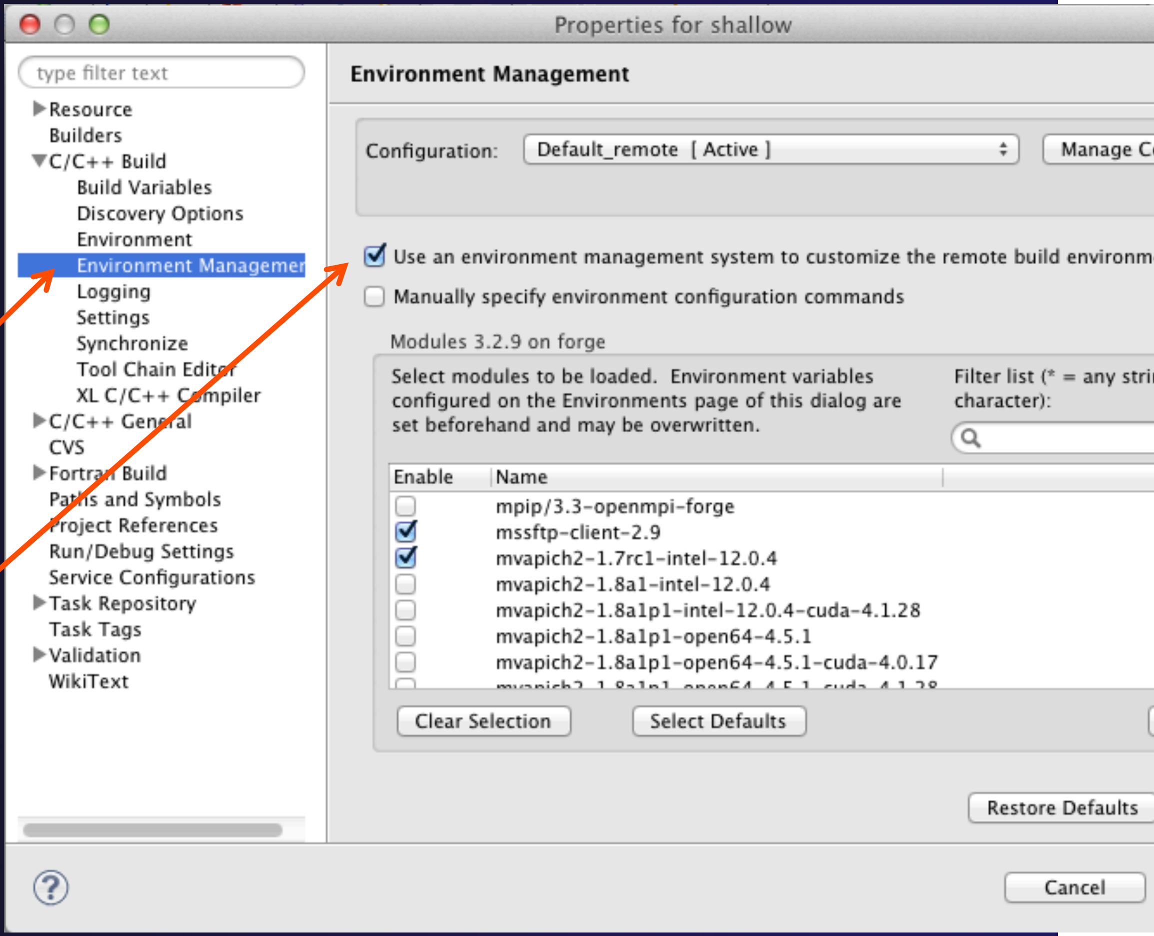
Selecting Build Configuration

- Multiple build configurations may be available
  - Remote and local build configuration
  - Build configurations for different architectures
- The active build configuration is set from the **Build Configurations** project context menu
  - Right click on project, then select the build configuration from the **Build Configurations > Set Active** menu



Configuring Build Modules

- If the remote system has Modules installed, a custom set of modules can be configured for building C/C++ projects
- In the project properties, navigate to **C/C++ Build > Environment Management**
- Check **Use an environment management system to customize the remote build environment**



Configuration: Default_remote [Active]

☑ Use an environment management system to customize the remote build environm
☐ Manually specify environment configuration commands

Modules 3.2.9 on forge

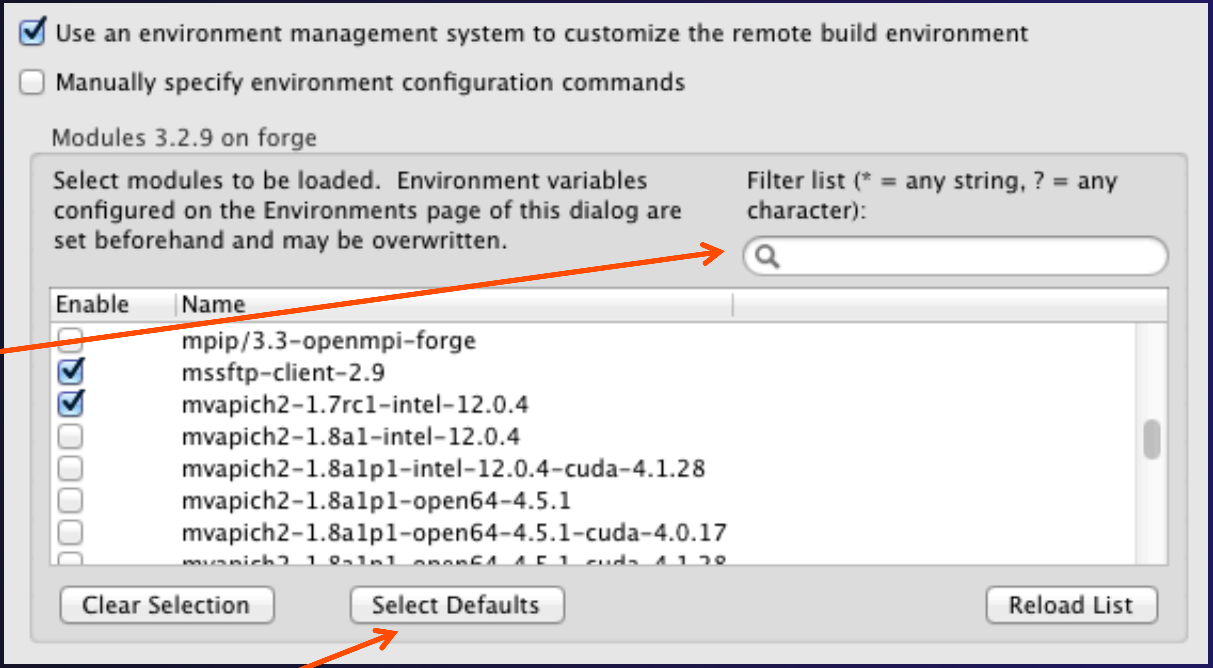
Select modules to be loaded. Environment variables configured on the Environments page of this dialog are set beforehand and may be overwritten.

Filter list (* = any stri character):

Enable	Name
☐	mpip/3.3-openmpi-forge
☑	mssftp-client-2.9
☑	mvapich2-1.7rc1-intel-12.0.4
☐	mvapich2-1.8a1-intel-12.0.4
☐	mvapich2-1.8a1p1-intel-12.0.4-cuda-4.1.28
☐	mvapich2-1.8a1p1-open64-4.5.1
☐	mvapich2-1.8a1p1-open64-4.5.1-cuda-4.0.17

Configuring Build Modules

- Select modules from the list
- Use the **Filter list** field to quickly find modules with a given name



☑ Use an environment management system to customize the remote build environment

☐ Manually specify environment configuration commands

Modules 3.2.9 on forge

Select modules to be loaded. Environment variables configured on the Environments page of this dialog are set beforehand and may be overwritten.

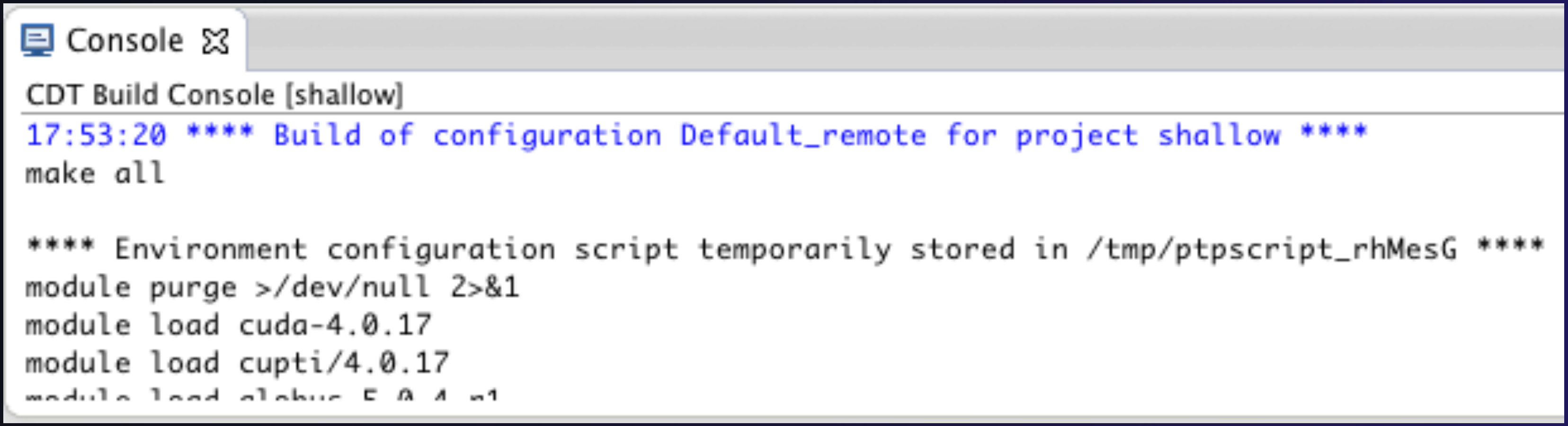
Filter list (* = any string, ? = any character):

Enable	Name
☐	mpip/3.3-openmpi-forge
☑	mssftp-client-2.9
☑	mvapich2-1.7rc1-intel-12.0.4
☐	mvapich2-1.8a1-intel-12.0.4
☐	mvapich2-1.8a1p1-intel-12.0.4-cuda-4.1.28
☐	mvapich2-1.8a1p1-open64-4.5.1
☐	mvapich2-1.8a1p1-open64-4.5.1-cuda-4.0.17

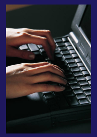
- Click **Select Defaults** to check only those modules that are present in a new Bash login shell

Configuring Build Modules

- To build the project, Eclipse will
  - Open a new Bash login shell
  - Execute *module purge*
  - Execute *module load* for each selected module
  - Run *make*
- Module commands are displayed in the Console view during build
- Beware of modules that must be loaded in a particular order, or that contain common paths like /bin or /usr/bin

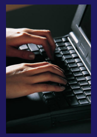


```
CDT Build Console [shallow]
17:53:20 **** Build of configuration Default_remote for project shallow ****
make all

**** Environment configuration script temporarily stored in /tmp/ptpscript_rhMesG ****
module purge >/dev/null 2>&1
module load cuda-4.0.17
module load cupti/4.0.17
```



Exercise

1. Start with your ‘shallow’ project
2. Build the project
3. Edit a source file and introduce a compile error
   - In main.c, line 97, change ‘;’ to ‘:’
   - Save, rebuild, and watch the Console view
   - Use the Problems view to locate the error
   - Locate the error in the source code by double clicking on the error in the **Problems** view
   - Fix the error
4. Rebuild the project and verify there are no build errors



Optional Exercise

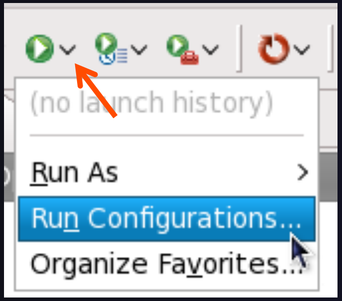
1. Open the Makefile in Eclipse. Note the line starting with "tags:" – this defines a make target named **tags**.
2. Open the Outline view while the Makefile is open. What icon is used to denote make targets in the Outline?
3. Right-click the **tags** entry in the Outline view. Add a Make Target for **tags**.
4. Open the Make Targets view, and build the **tags** target.

5. Rename Makefile to Makefile.mk
6. Attempt to build the project; it will fail
7. In the project properties (under the C/C++ Build category), change the build command to: **`make –f Makefile.mk`**
8. Build the project; it should succeed

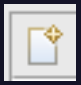
Running an Application

- Objective
  - Learn how to run an MPI program on a remote system
- Contents
  - Creating a run configuration
  - Configuring the application run
  - Monitoring the system and jobs
  - Controlling jobs
  - Obtaining job output

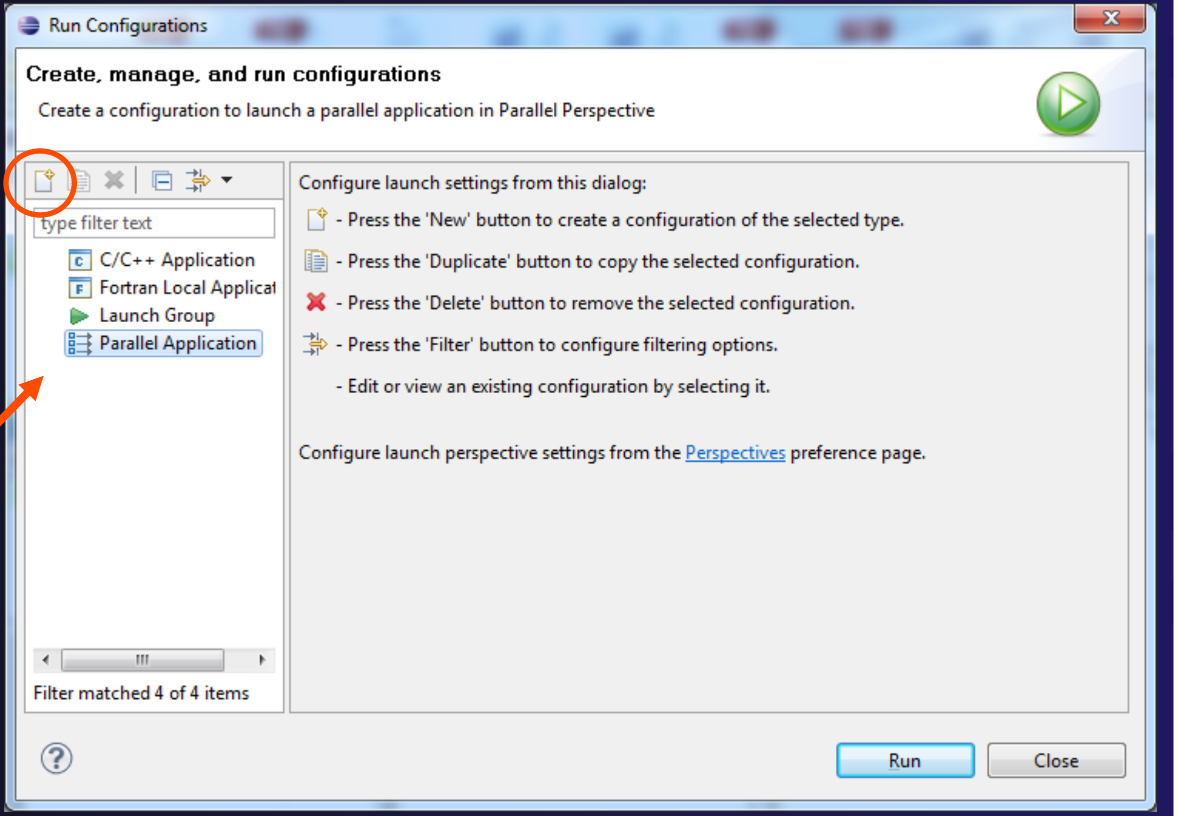
Creating a Run Configuration



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



Or, just double-click on **Parallel Application** to create a new one



Create, manage, and run configurations

Create a configuration to launch a parallel application in Parallel Perspective

Configure launch settings from this dialog:

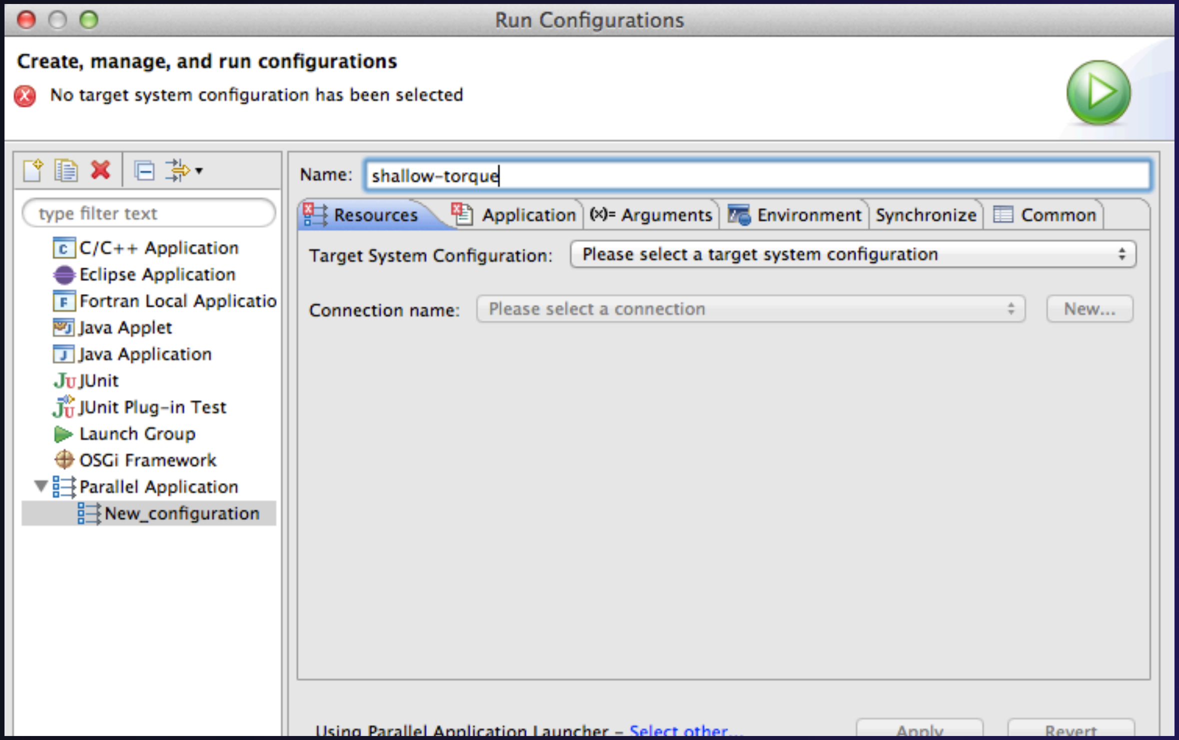
- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration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

Configure launch perspective settings from the Perspectives preference page.

Note: We use "Launch Configuration" as a generic term to refer to either a "Run Configuration" or a "Debug Configuration", which is used for debugging.

Set Run Configuration Name

- Enter a name for this run configuration
  - E.g. "shallow-torque"
- This allows you to easily re-run the same application


Run Configurations

Create, manage, and run configurations

No target system configuration has been selected

type filter text

C/C++ Application
Eclipse Application
Fortran Local Applicatio
Java Applet
Java Application
JUnit
JUnit Plug-in Test
Launch Group
OSGi Framework
Parallel Application
New_configuration

Name: shallow-torque

Resources | Application | Arguments | Environment | Synchronize | Common

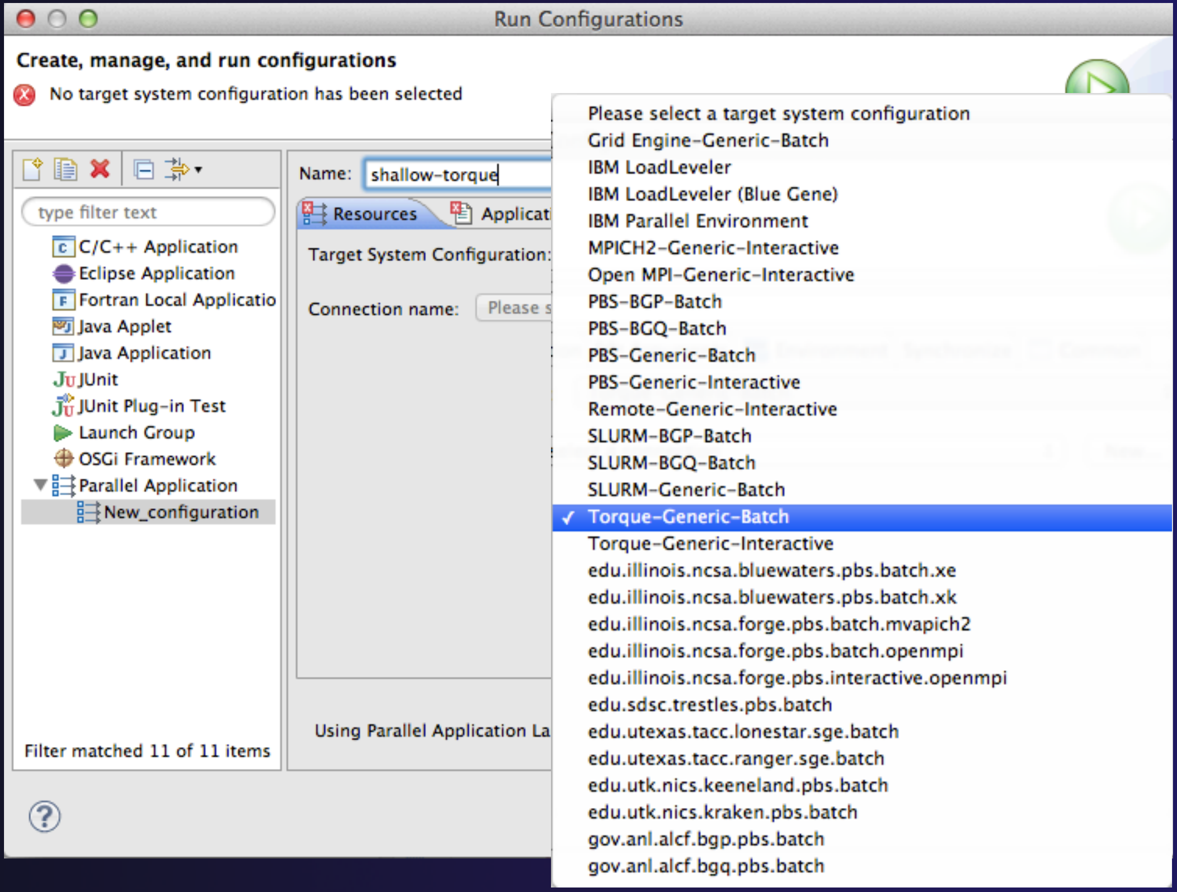
Target System Configuration: Please select a target system configuration

Connection name: Please select a connection   New...

Using Parallel Application Launcher – Select other...   Apply   Revert

Configuring the Target System

- In **Resources** tab, select a **Target System Configuration** that corresponds to your target system
  - The tutorial instructor will indicate what Target System Configuration to select
- Target system configurations can be *generic* or can be specific to a particular system
- Use the specific configuration if available, or the generic configuration that most closely matches your system


Run Configurations

Create, manage, and run configurations

No target system configuration has been selected

Name: shallow-torque

Resources | Applicat

Target System Configuration:

Connection name: Please s

Using Parallel Application La

type filter text

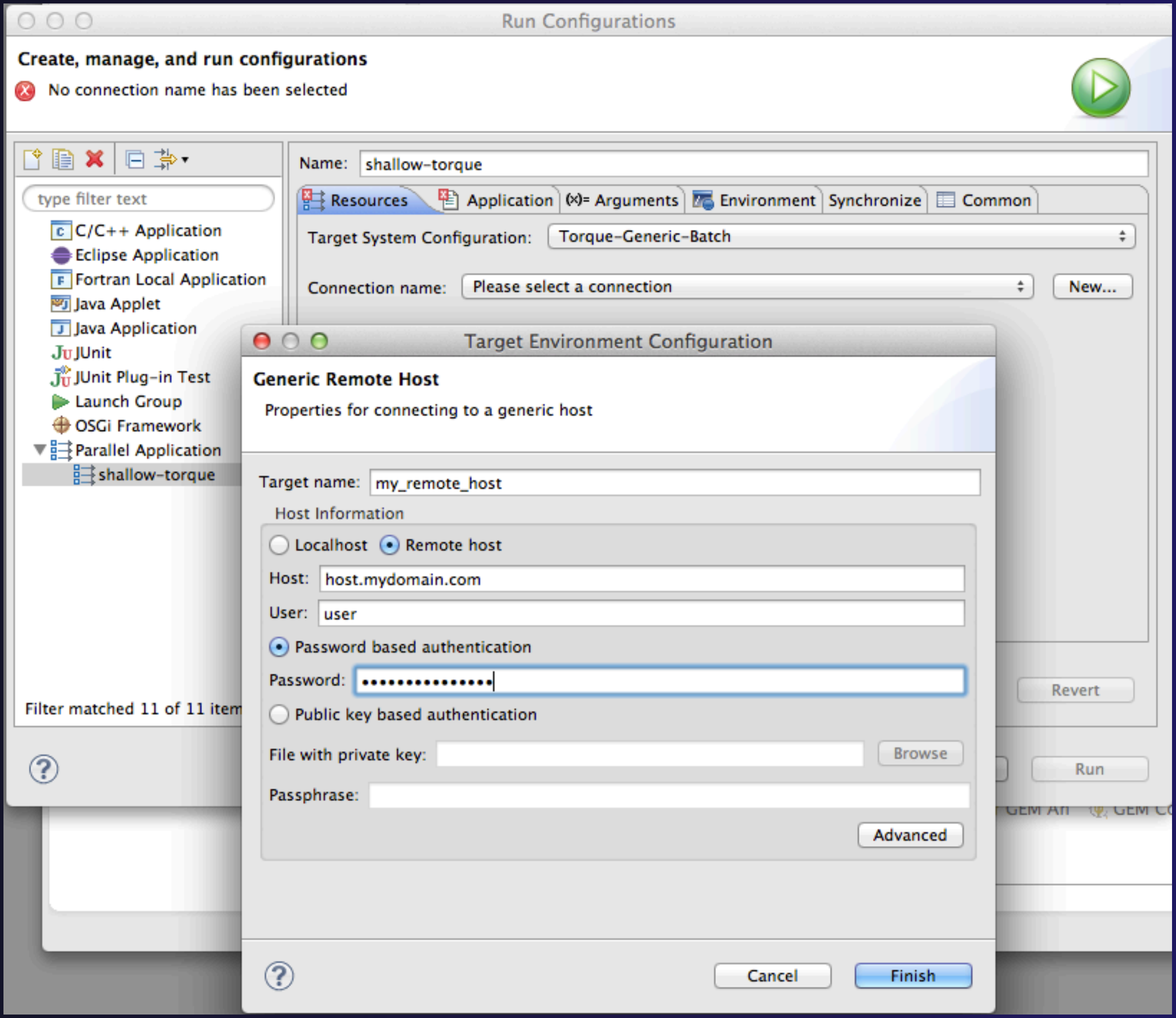
C/C++ Application
Eclipse Application
Fortran Local Applicatio
Java Applet
Java Application
JUnit
JUnit Plug-in Test
Launch Group
OSGi Framework
Parallel Application
New_configuration

Filter matched 11 of 11 items

Please select a target system configuration
Grid Engine-Generic-Batch
IBM LoadLeveler
IBM LoadLeveler (Blue Gene)
IBM Parallel Environment
MPICH2-Generic-Interactive
Open MPI-Generic-Interactive
PBS-BGP-Batch
PBS-BGQ-Batch
PBS-Generic-Batch
PBS-Generic-Interactive
Remote-Generic-Interactive
SLURM-BGP-Batch
SLURM-BGQ-Batch
SLURM-Generic-Batch
✓ Torque-Generic-Batch
Torque-Generic-Interactive
edu.illinois.ncsa.bluewaters.pbs.batch.xe
edu.illinois.ncsa.bluewaters.pbs.batch.xk
edu.illinois.ncsa.forge.pbs.batch.mvapich2
edu.illinois.ncsa.forge.pbs.batch.openmpi
edu.illinois.ncsa.forge.pbs.interactive.openmpi
edu.sdsc.trestles.pbs.batch
edu.utexas.tacc.lonestar.sge.batch
edu.utexas.tacc.ranger.sge.batch
edu.utk.nics.keeneland.pbs.batch
edu.utk.nics.kraken.pbs.batch
gov.anl.alcf.bgp.pbs.batch
gov.anl.alcf.bgq.pbs.batch

Configure the Connection

- Choose a connection to use to communicate with the target system
- If no connection has been configured, click on the **New** button to create a new one
  - Fill in connection information, then click ok
- The new connection should appear in the dropdown list


Run Configurations

Create, manage, and run configurations

No connection name has been selected

Name: shallow-torque

Target System Configuration: Torque-Generic-Batch

Connection name: Please select a connection

Target Environment Configuration

Generic Remote Host

Properties for connecting to a generic host

Target name: my_remote_host

Host Information

Localhost [✓ Remote host]

Host: host.mydomain.com

User: user

[✓ Password based authentication]

Password: •••••••••••••••

Public key based authentication

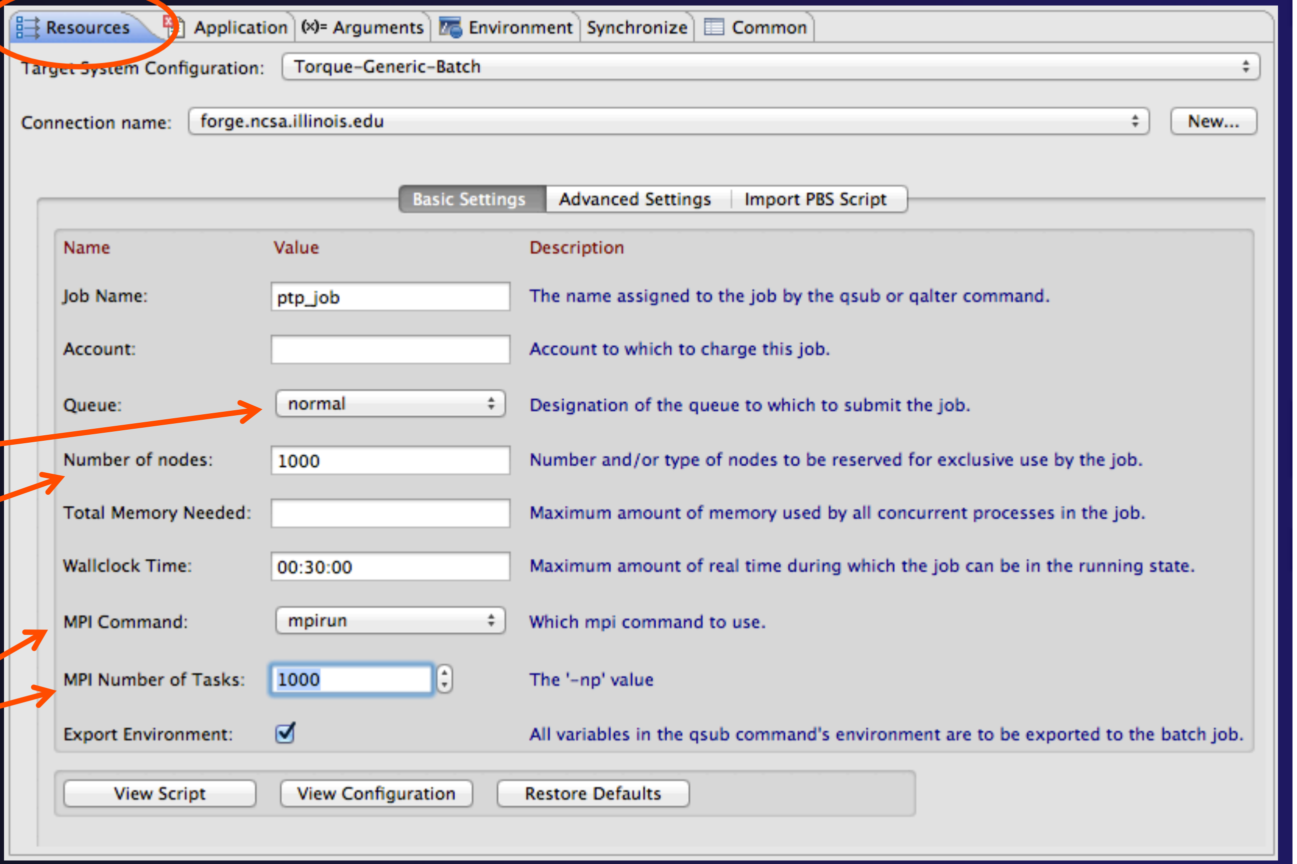
File with private key:

Passphrase:

Filter matched 11 of 11 items

Resources Tab

- The content of the **Resources** tab will vary depending on the target system configuration selected
- This example shows the TORQUE configuration
- For TORQUE, you will normally need to select the *Queue* and the *Number of nodes*
- For parallel jobs, choose the *MPI Command* and the *Number of Tasks*



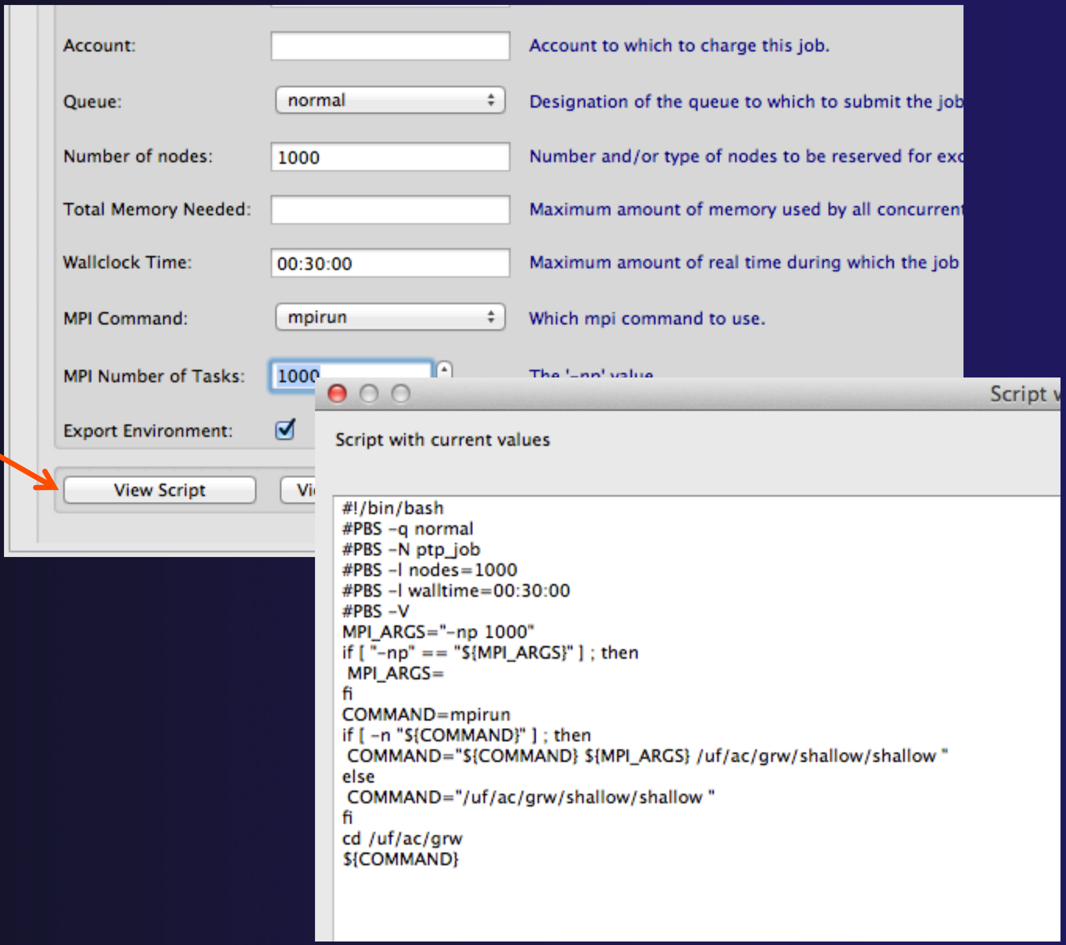
Target System Configuration: Torque-Generic-Batch

Connection name: forge.ncsa.illinois.edu

Name	Value	Description
Job Name:	ptp_job	The name assigned to the job by the qsub or qalter command.
Account:		Account to which to charge this job.
Queue:	normal	Designation of the queue to which to submit the job.
Number of nodes:	1000	Number and/or type of nodes to be reserved for exclusive use by the job.
Total Memory Needed:		Maximum amount of memory used by all concurrent processes in the job.
Wallclock Time:	00:30:00	Maximum amount of real time during which the job can be in the running state.
MPI Command:	mpirun	Which mpi command to use.
MPI Number of Tasks:	1000	The '-np' value
Export Environment:	✓	All variables in the qsub command's environment are to be exported to the batch job.

Viewing the Job Script

- Some target configurations will provide a **View Script** button
- Click on this to view the job script that will be submitted to the job scheduler
- Batch scheduler configurations should also provide a means of importing a batch script



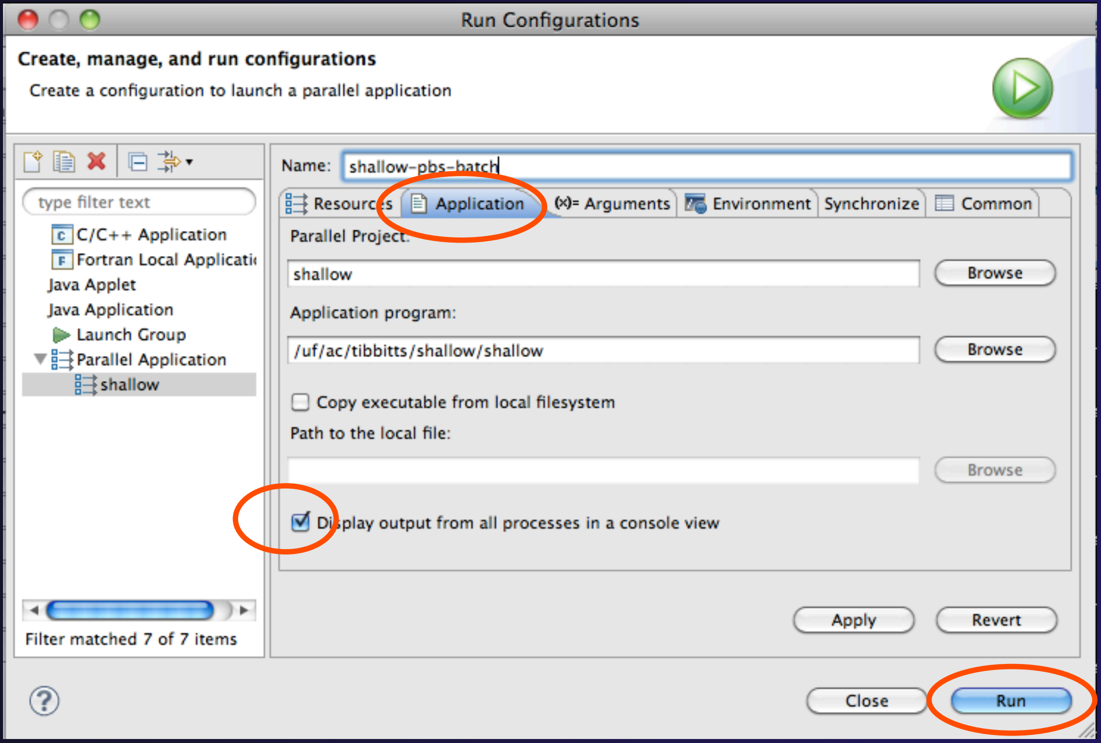
Account: | Account to which to charge this job.
Queue: normal | Designation of the queue to which to submit the job
Number of nodes: 1000 | Number and/or type of nodes to be reserved for exe
Total Memory Needed: | Maximum amount of memory used by all concurrent
Wallclock Time: 00:30:00 | Maximum amount of real time during which the job
MPI Command: mpirun | Which mpi command to use.
MPI Number of Tasks: 1000 | The '-np' value
Export Environment: ✓

Script with current values

```
#!/bin/bash
#PBS -q normal
#PBS -N ptp_job
#PBS -l nodes=1000
#PBS -l walltime=00:30:00
#PBS -V
MPI_ARGS="-np 1000"
if [ "-np" == "${MPI_ARGS}" ] ; then
 MPI_ARGS=
fi
COMMAND=mpirun
if [ -n "${COMMAND}" ] ; then
 COMMAND="${COMMAND} ${MPI_ARGS} /uf/ac/grw/shallow/shallow "
else
 COMMAND="/uf/ac/grw/shallow/shallow "
fi
cd /uf/ac/grw
${COMMAND}
```

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**



Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application

Name: shallow-pbs-batch

Parallel Project: shallow

Application program: /uf/ac/tibbitts/shallow/shallow

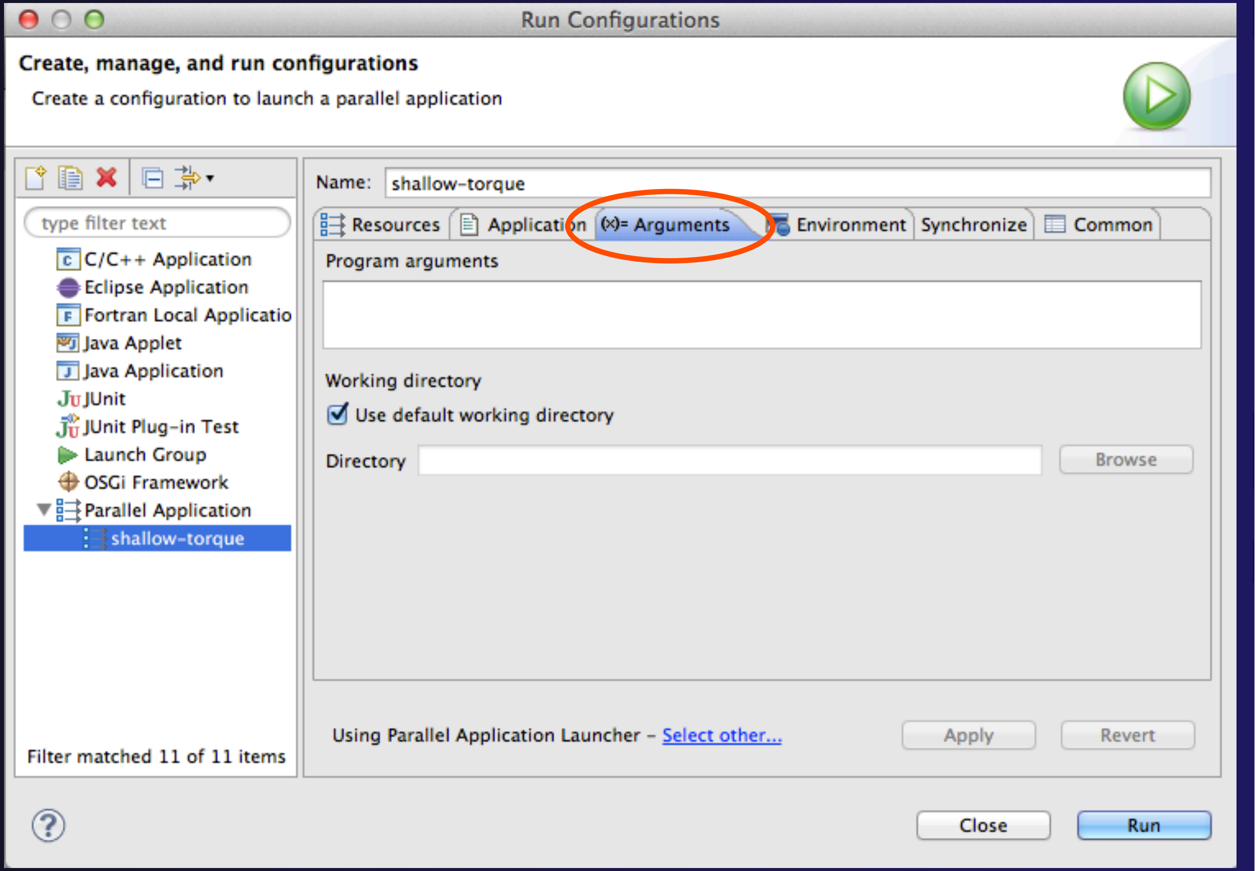
Copy executable from local filesystem

Path to the local file:

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

Arguments Tab (Optional)

- The **Arguments** tab lets you supply command-line arguments to the application
- You can also change the default working directory when the application executes


Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application

Name: shallow-torque

Program arguments

Working directory

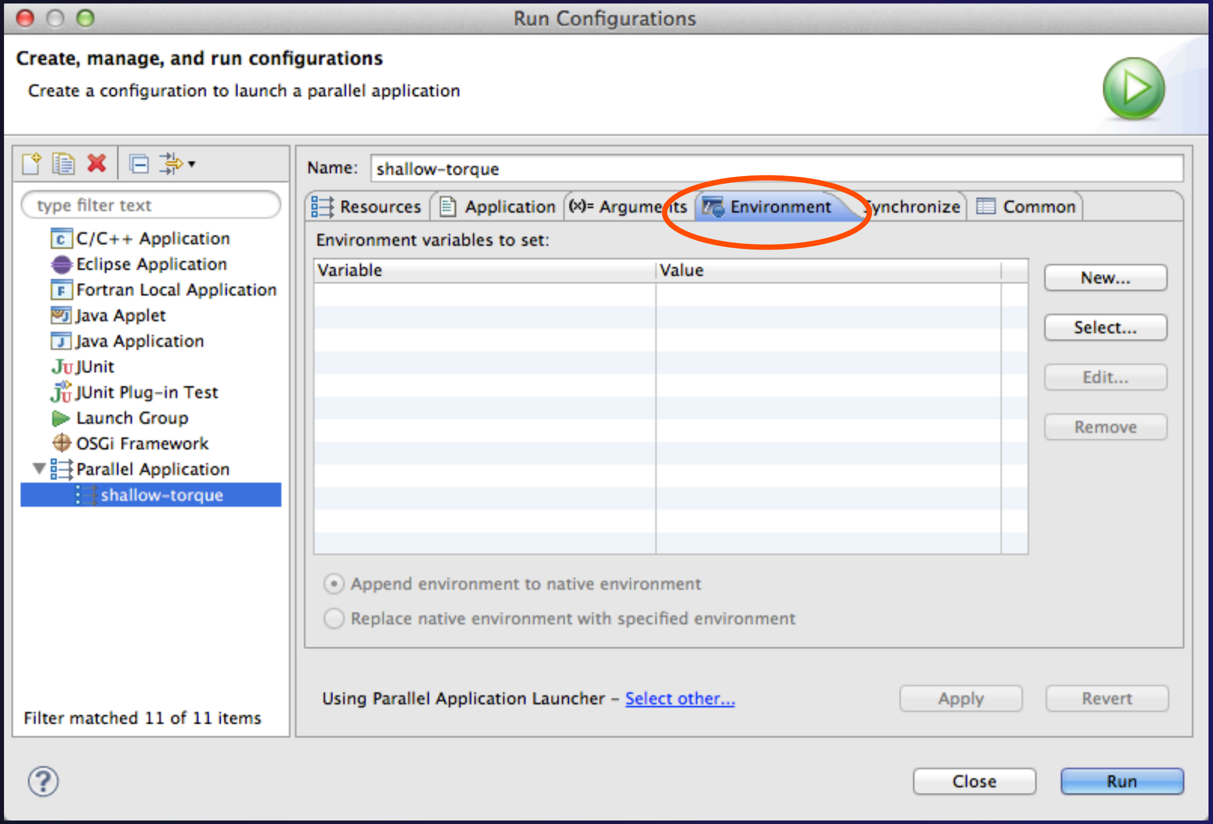
[✓] Use default working directory

Directory

Using Parallel Application Launcher – Select other...

Environment Tab (Optional)

- The **Environment** tab lets you set environment variables that are passed to the job submission command
- This is independent of the Environment Management (module/softenv) support described in a separate module



Run Configurations

Create, manage, and run configurations

Create a configuration to launch a parallel application

type filter text

- C/C++ Application
- Eclipse Application
- Fortran Local Application
- Java Applet
- Java Application
- JUnit
- JUnit Plug-in Test
- Launch Group
- OSGi Framework
- Parallel Application
  - shallow-torque

Filter matched 11 of 11 items

Name: shallow-torque

Resources | Application | (x)= Arguments | Environment | Synchronize | Common

Environment variables to set:

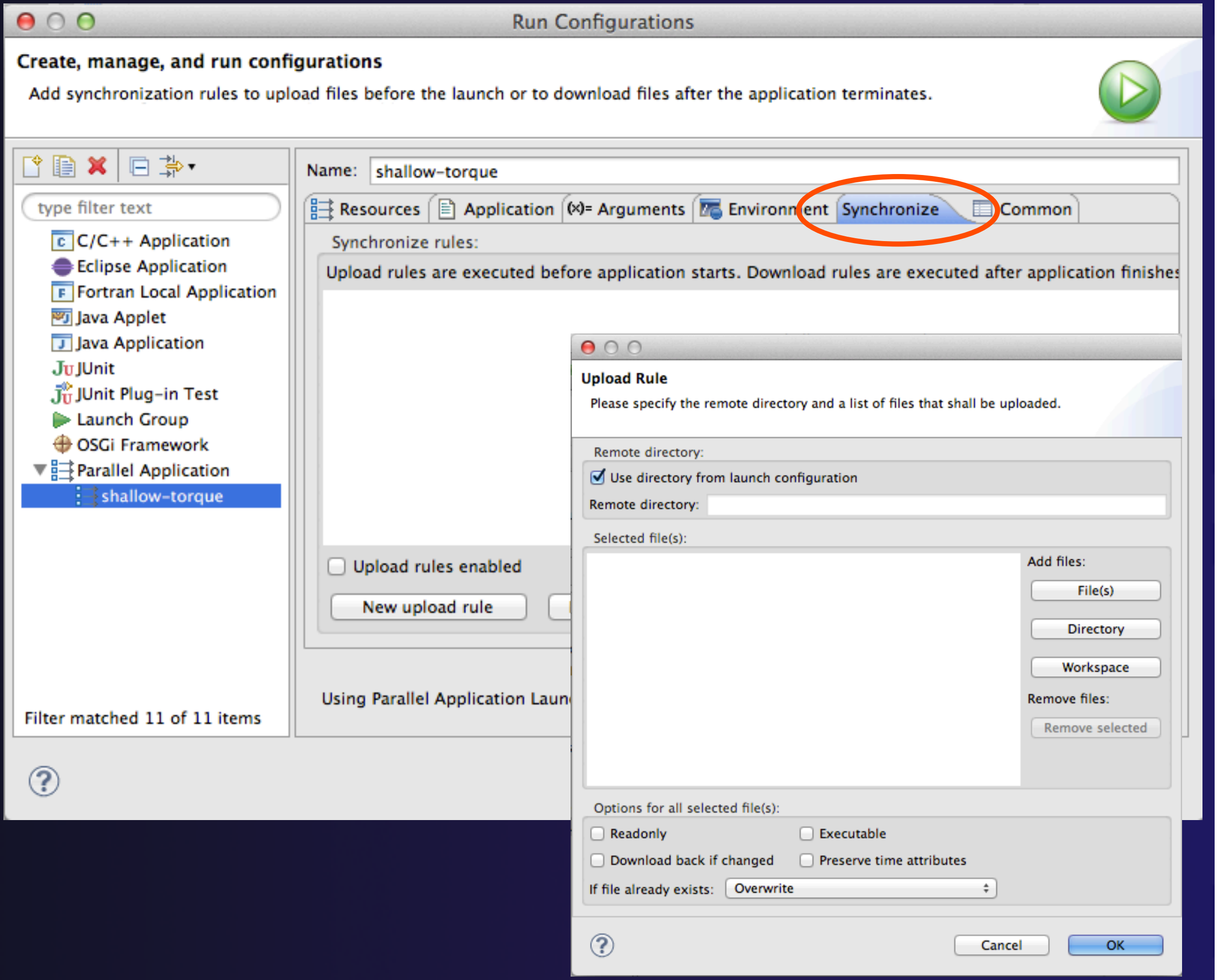
Variable	Value

(●) Append environment to native environment
( ) Replace native environment with specified environment

Using Parallel Application Launcher – Select other...

Synchronize Tab (Optional)

- The **Synchronize** tab lets you specify upload/ download rules that are execute prior to, and after the job execution
- Click on the **New upload/download rule** buttons to define rules
- The rule defines which file will be uploaded/ downloaded and where it will be put
- Can be used in conjunction with program arguments to supply input data to the application


Run Configurations

Create, manage, and run configurations

Add synchronization rules to upload files before the launch or to download files after the application terminates.

Name: shallow-torque

Synchronize rules:

Upload rules are executed before application starts. Download rules are executed after application finishes

Upload rules enabled

Using Parallel Application Laun

Filter matched 11 of 11 items

Upload Rule

Please specify the remote directory and a list of files that shall be uploaded.

Remote directory:

[✓] Use directory from launch configuration

Remote directory:

Selected file(s):

Options for all selected file(s):

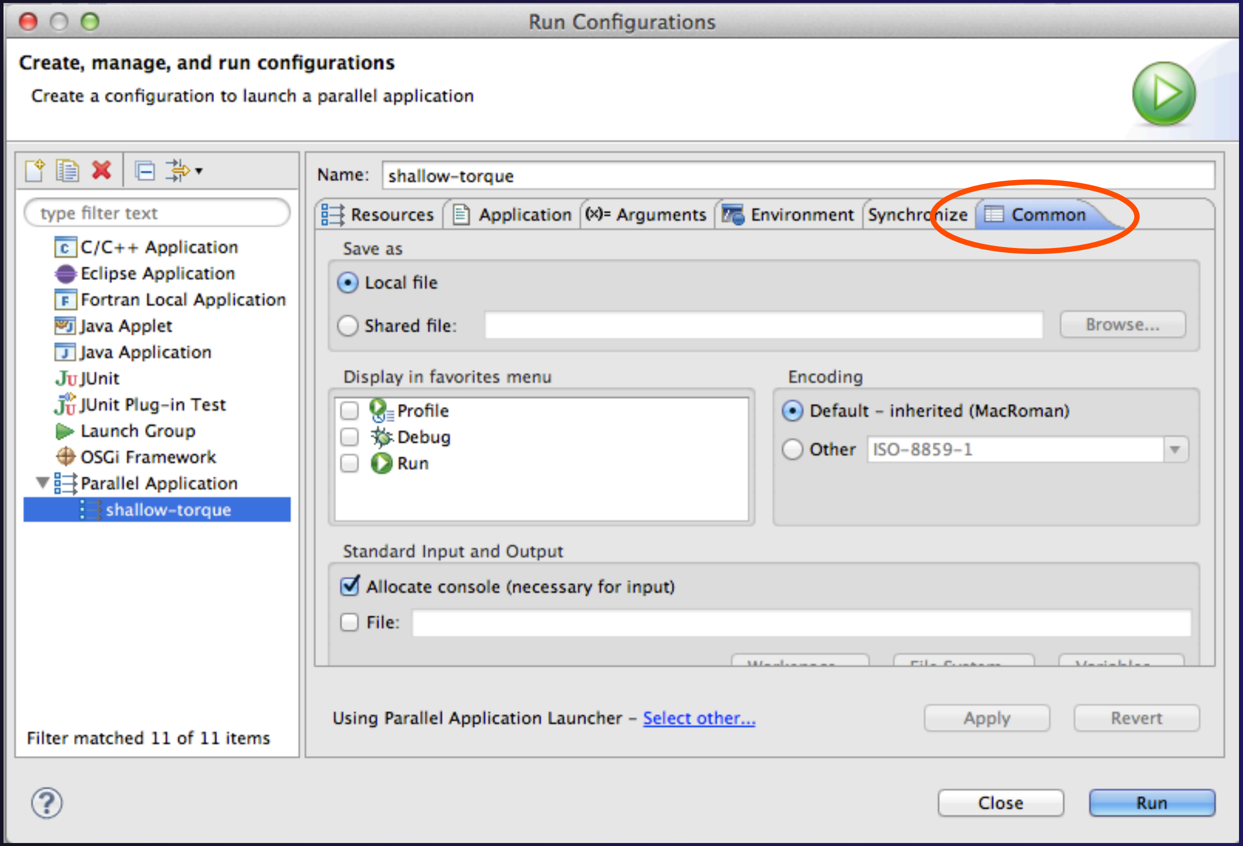
[ ] Readonly [ ] Executable

[ ] Download back if changed [ ] Preserve time attributes

If file already exists: Overwrite

Common Tab (Optional)

- The **Common** tab is available for most launch configuration types (not just Parallel Application)
- Allows the launch configuration to be exported to an external file
- Can add the launch configuration to the favorites menu, which is available on the main Eclipse toolbar
- Select **Run** to launch the job



Name: shallow-torque

Save as
- [✓ Local file]
- Shared file:

Display in favorites menu
- ☐ Profile
- ☐ Debug
- ☐ Run

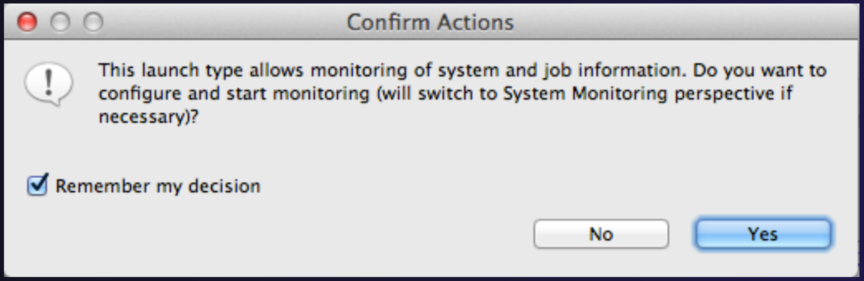
Encoding
- [✓ Default - inherited (MacRoman)]
- Other ISO-8859-1

Standard Input and Output
- ☑ Allocate console (necessary for input)
- ☐ File:

Using Parallel Application Launcher - Select other...

Run

- Select **Run** to launch the job
- You may be asked to switch to the System Monitoring Perspective



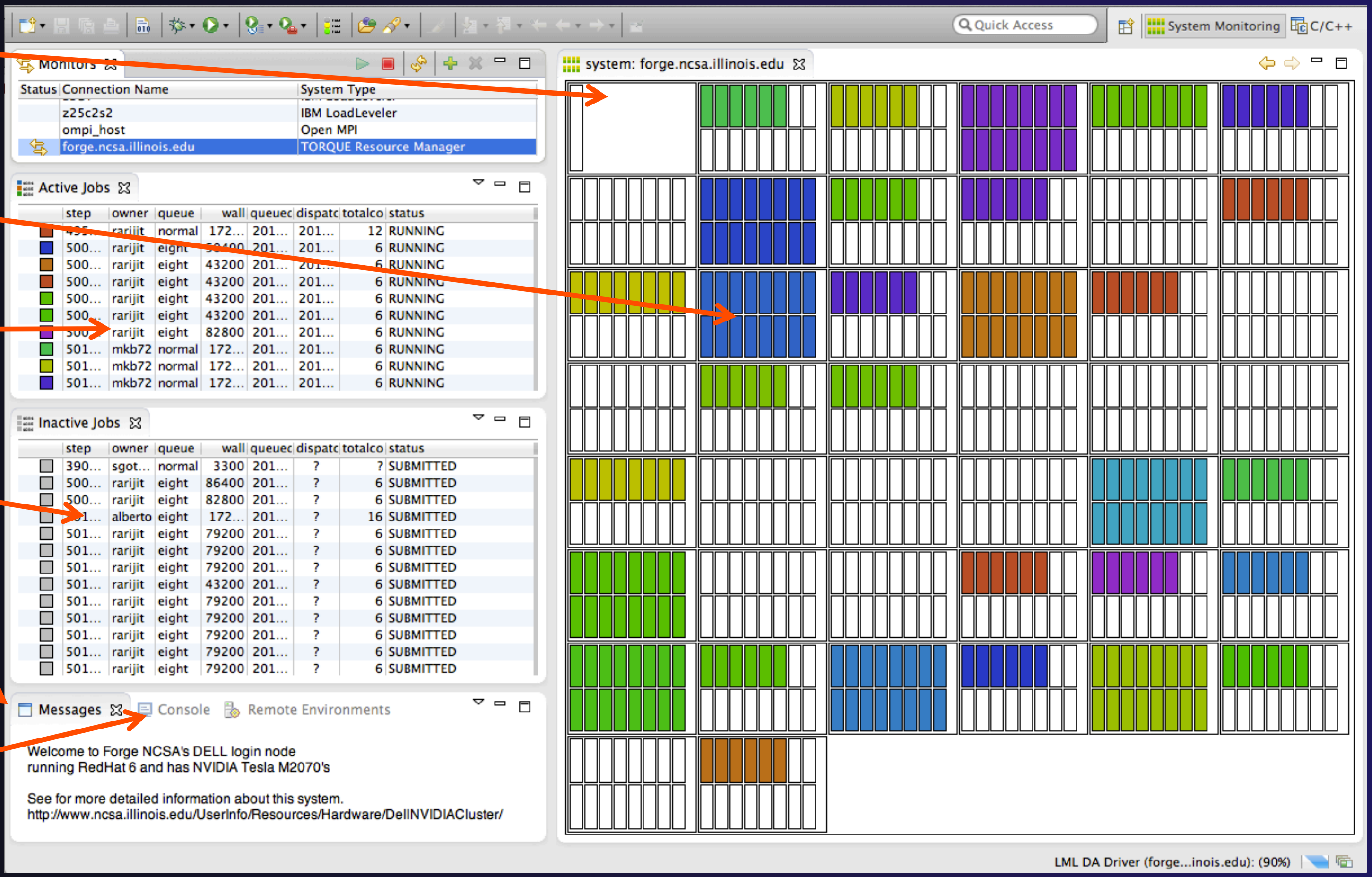
This launch type allows monitoring of system and job information. Do you want to configure and start monitoring (will switch to System Monitoring perspective if necessary)?

✓ Remember my decision

- Select **Remember my decision** so you won't be asked again
- Select **Yes** to switch and launch the job

System Monitoring Perspective

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



Status	Connection Name	System Type
	z25c2s2	IBM LoadLeveler
	ompi_host	Open MPI
	forge.ncsa.illinois.edu	TORQUE Resource Manager

Active Jobs

	step	owner	queue	wall	queuec	dispatc	totalco	status
🟥	[illegible]	rarijit	normal	172...	201...	201...	12	RUNNING
🟦	500...	rarijit	eight	50400	201...	201...	6	RUNNING
🟧	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟥	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟩	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟩	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟪	500...	rarijit	eight	82800	201...	201...	6	RUNNING
🟩	501...	mkb72	normal	172...	201...	201...	6	RUNNING
🟨	501...	mkb72	normal	172...	201...	201...	6	RUNNING
🟪	501...	mkb72	normal	172...	201...	201...	6	RUNNING

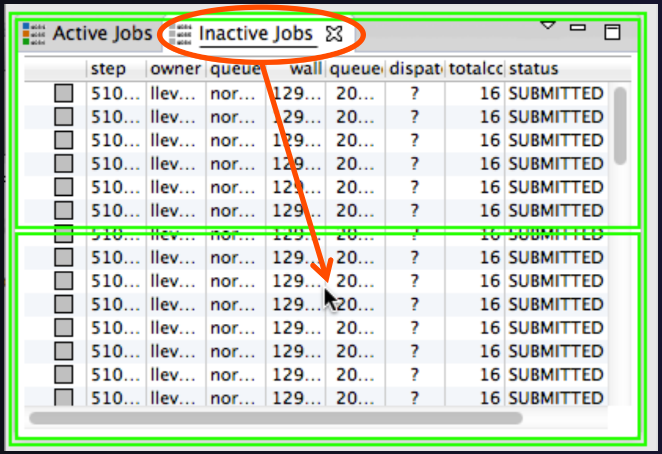
Inactive Jobs

	step	owner	queue	wall	queuec	dispatc	totalco	status
⬜	390...	sgot...	normal	3300	201...	?	?	SUBMITTED
⬜	500...	rarijit	eight	86400	201...	?	6	SUBMITTED
⬜	500...	rarijit	eight	82800	201...	?	6	SUBMITTED
⬜	[illegible]1...	alberto	eight	172...	201...	?	16	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	43200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED

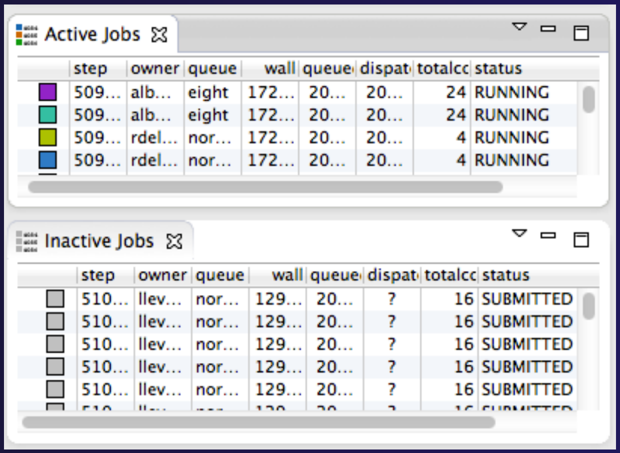
```
Welcome to Forge NCSA's DELL login node
running RedHat 6 and has NVIDIA Tesla M2070's

See for more detailed information about this system.
http://www.ncsa.illinois.edu/UserInfo/Resources/Hardware/DellNVIDIACluster/
```

Moving views

- The System Monitoring Perspective overlaps the **Active Jobs** and **Inactive Jobs** views
- To split them apart and see both at once, *drag* the tab for the **Inactive Jobs** view to the lower half of its area, and let go of mouse



	step	owner	queue	wall	queue	dispat	totalcc	status
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED



Active Jobs

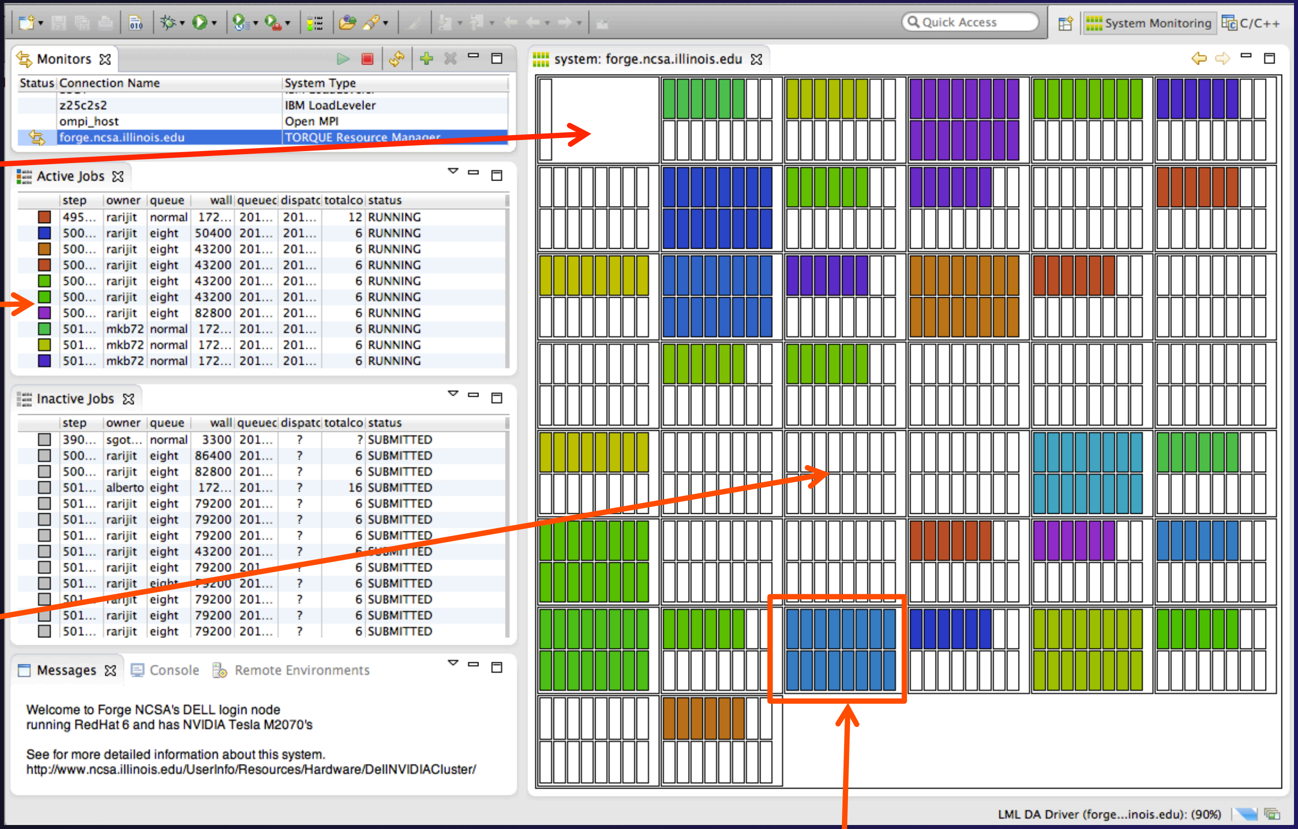
	step	owner	queue	wall	queue	dispat	totalcc	status
🟪	509...	alb...	eight	172...	20...	20...	24	RUNNING
🟩	509...	alb...	eight	172...	20...	20...	24	RUNNING
🟨	509...	rdel...	nor...	172...	20...	20...	4	RUNNING
🟦	509...	rdel...	nor...	172...	20...	20...	4	RUNNING

Inactive Jobs

	step	owner	queue	wall	queue	dispat	totalcc	status
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED
☐	510...	llev...	nor...	129...	20...	?	16	SUBMITTED

System Monitoring

- **System** view, with abstraction of system configuration
- Hold mouse button down on a job in **Active Jobs** view to see where it is running in **System** view
- Hover over node in **System** view to see job running on node in **Active Jobs** view


Quick Access
System Monitoring
C/C++
system: forge.ncsa.illinois.edu
LML DA Driver (forge...inois.edu): (90%)

Monitors

Status	Connection Name	System Type
	z25c2s2	IBM LoadLeveler
	ompi_host	Open MPI
	forge.ncsa.illinois.edu	TORQUE Resource Manager

Active Jobs

	step	owner	queue	wall	queuec	dispatc	totalco	status
🟫	495...	rarijit	normal	172...	201...	201...	12	RUNNING
🟦	500...	rarijit	eight	50400	201...	201...	6	RUNNING
🟧	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟥	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟩	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟩	500...	rarijit	eight	43200	201...	201...	6	RUNNING
🟪	500...	rarijit	eight	82800	201...	201...	6	RUNNING
🟩	501...	mkb72	normal	172...	201...	201...	6	RUNNING
🟨	501...	mkb72	normal	172...	201...	201...	6	RUNNING
🟦	501...	mkb72	normal	172...	201...	201...	6	RUNNING

Inactive Jobs

	step	owner	queue	wall	queuec	dispatc	totalco	status
⬜	390...	sgot...	normal	3300	201...	?	?	SUBMITTED
⬜	500...	rarijit	eight	86400	201...	?	6	SUBMITTED
⬜	500...	rarijit	eight	82800	201...	?	6	SUBMITTED
⬜	501...	alberto	eight	172...	201...	?	16	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	43200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED

Messages | Console | Remote Environments

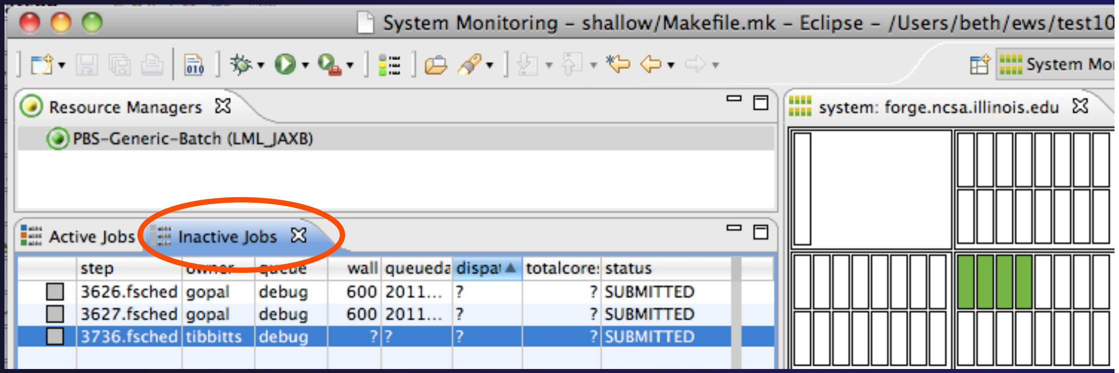
Welcome to Forge NCSA's DELL login node
running RedHat 6 and has NVIDIA Tesla M2070's

See for more detailed information about this system.
http://www.ncsa.illinois.edu/UserInfo/Resources/Hardware/DellNVIDIACluster/

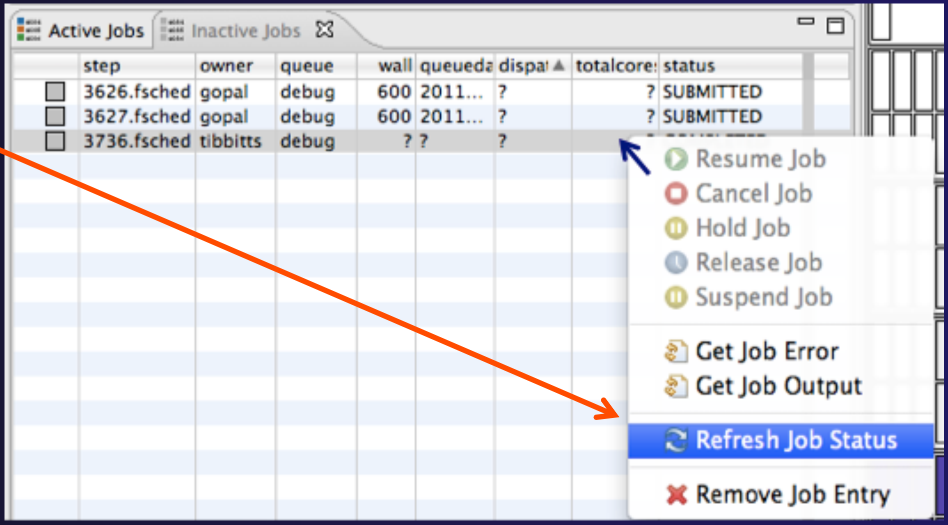
One node with
16 cores

Job Monitoring

- Job initially appears in **Inactive Jobs** view
- Moves to the **Active Jobs** view when execution begings
- Returns to **Inactive Jobs** view on completion
- Status refreshes automatically every 60 sec
- Can force refresh with menu


System Monitoring – shallow/Makefile.mk – Eclipse – /Users/beth/ews/test10
System Mo
Resource Managers
PBS-Generic-Batch (LML_JAXB)
system: forge.ncsa.illinois.edu
Active Jobs
Inactive Jobs

	step	owner	queue	wall	queueda	dispa ▲	totalcore	status
☐	3626.fsched	gopal	debug	600	2011...	?	?	SUBMITTED
☐	3627.fsched	gopal	debug	600	2011...	?	?	SUBMITTED
☐	3736.fsched	tibbitts	debug	?	?	?	?	SUBMITTED

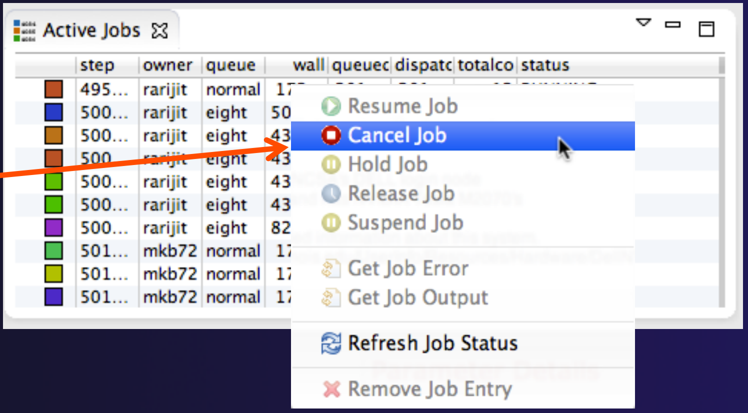


	step	owner	queue	wall	queueda	dispa ▲	totalcore	status
☐	3626.fsched	gopal	debug	600	2011...	?	?	SUBMITTED
☐	3627.fsched	gopal	debug	600	2011...	?	?	SUBMITTED
☐	3736.fsched	tibbitts	debug	?	?	?		

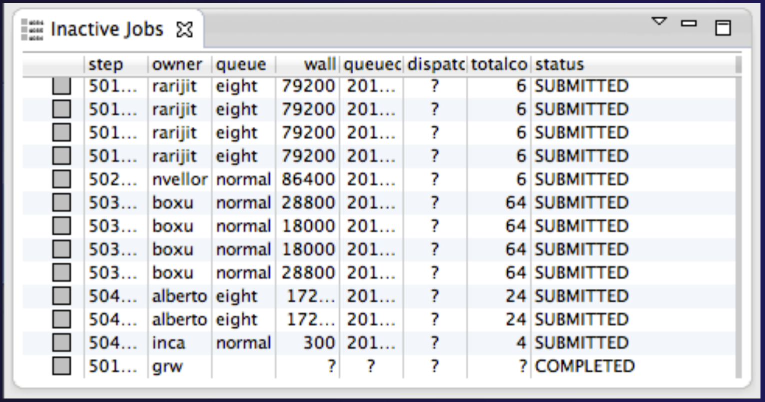
Controlling Jobs

- Right click on a job to open context menu
- Actions will be enabled IFF
  - The job belongs to you
  - The action is available on the target system
  - The job is in the correct state for the action

- When job has COMPLETED, it will remain in the **Inactive Jobs** view



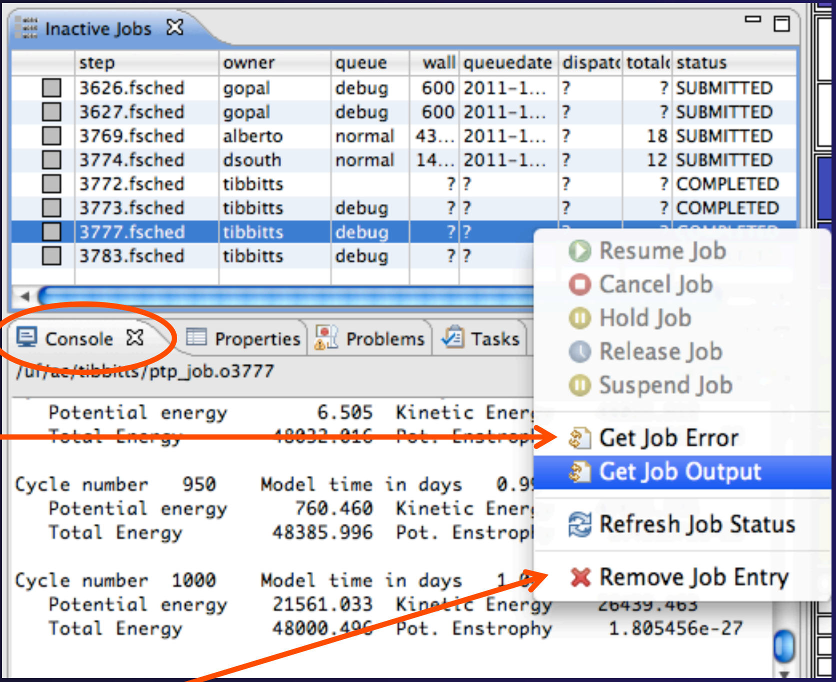
	step	owner	queue	wall	queuec	dispatc	totalco	status
🟥	495...	rarijit	normal	17				
🟦	500...	rarijit	eight	50				
🟧	500...	rarijit	eight	43				
🟥	500	rarijit	eight	43				
🟩	500...	rarijit	eight	43				
🟩	500...	rarijit	eight	43				
🟪	500...	rarijit	eight	82				
🟩	501...	mkb72	normal	17				
🟨	501...	mkb72	normal	17				
🟪	501...	mkb72	normal	17				



	step	owner	queue	wall	queuec	dispatc	totalco	status
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	501...	rarijit	eight	79200	201...	?	6	SUBMITTED
⬜	502...	nvellor	normal	86400	201...	?	6	SUBMITTED
⬜	503...	boxu	normal	28800	201...	?	64	SUBMITTED
⬜	503...	boxu	normal	18000	201...	?	64	SUBMITTED
⬜	503...	boxu	normal	18000	201...	?	64	SUBMITTED
⬜	503...	boxu	normal	28800	201...	?	64	SUBMITTED
⬜	504...	alberto	eight	172...	201...	?	24	SUBMITTED
⬜	504...	alberto	eight	172...	201...	?	24	SUBMITTED
⬜	504...	inca	normal	300	201...	?	4	SUBMITTED
⬜	501...	grw		?	?	?	?	COMPLETED

Obtaining Job Output

- After status changes to COMPLETED, the output is available
  - Right-click on the job
  - Select **Get Job Output** to display output sent to standard output
  - Select **Get Job Error** to retrieve output sent to standard error
- Output/Error info shows in Console View
- Jobs can be removed by selecting **Remove Job Entry**

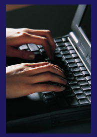


Inactive Jobs

	step	owner	queue	wall	queuedate	dispatc	totalc	status
☐	3626.fsched	gopal	debug	600	2011-1...	?	?	SUBMITTED
☐	3627.fsched	gopal	debug	600	2011-1...	?	?	SUBMITTED
☐	3769.fsched	alberto	normal	43...	2011-1...	?	18	SUBMITTED
☐	3774.fsched	dsouth	normal	14...	2011-1...	?	12	SUBMITTED
☐	3772.fsched	tibbitts		?	?	?	?	COMPLETED
☐	3773.fsched	tibbitts	debug	?	?	?	?	COMPLETED
☐	3777.fsched	tibbitts	debug	?	?	?	?	COMPLETED
☐	3783.fsched	tibbitts	debug	?	?			

/u/ac/tibbitts/ptp_job.o3777

```
   Potential energy        6.505   Kinetic Ener
   Total Energy        48032.016   Pot. Enstroph

Cycle number    950    Model time in days    0.9
   Potential energy      760.460   Kinetic Ener
   Total Energy        48385.996   Pot. Enstropl

Cycle number   1000    Model time in days    1.0
   Potential energy    21561.033   Kinetic Energy      26439.463
   Total Energy        48000.496   Pot. Enstrophy      1.805456e-27
```



Exercise

- Start with your ‘shallow’ project
- Create a run configuration
- Complete the Resources tab
- Select the executable in the Application tab
- Submit the job
- Check the job is visible in the Inactive Jobs view, moves to the Active Jobs view when it starts running, then moves back to the Inactive Jobs view when completed
- Remove the job from the Inactive Jobs view

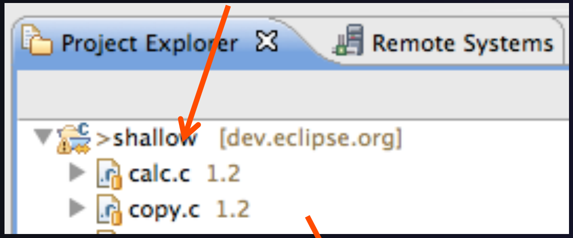
Fortran

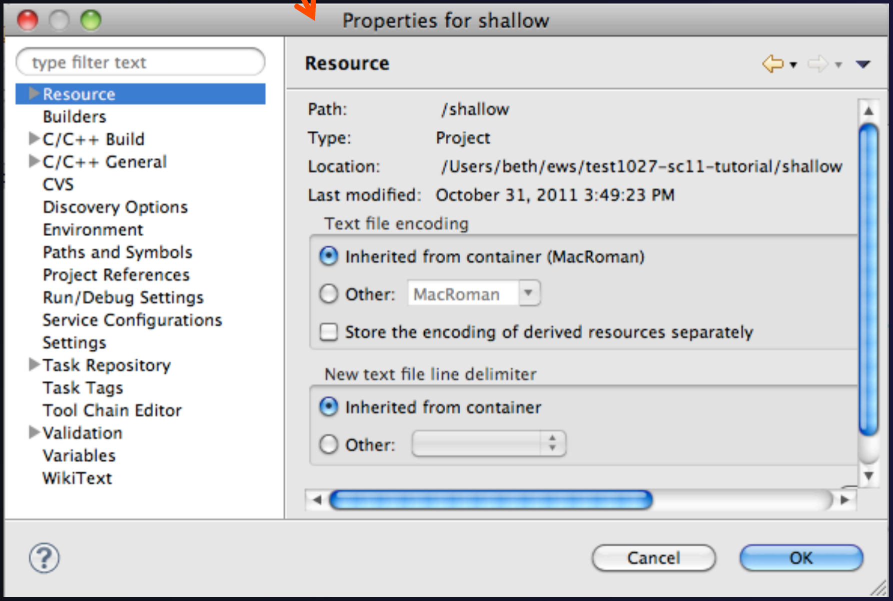
- Objectives
  - Learn how to create and convert Fortran projects
  - Learn to use Fortran-specific editing features
  - Learn about Fortran-specific properties/preferences
- Contents
  - Fortran projects
  - Using the Fortran editor
  - Fortran project properties and workbench preferences
- Prerequisites
  - Basics (for exercises)

Configuring Fortran Projects

Project Properties

- Right-click Project
- Select **Properties…**




Properties for shallow

Resource

Path: /shallow
Type: Project
Location: /Users/beth/ews/test1027-sc11-tutorial/shallow
Last modified: October 31, 2011 3:49:23 PM

Text file encoding
[✓ Inherited from container (MacRoman)]
Other: MacRoman
Store the encoding of derived resources separately

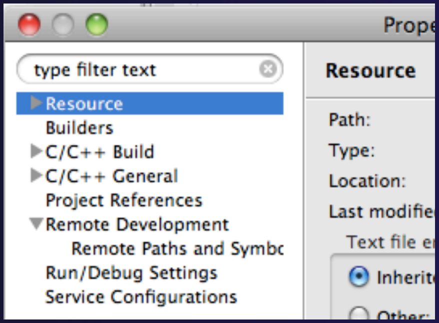
New text file line delimiter
[✓ Inherited from container]
Other:

- *Project properties* are settings that can be changed for each project
  - Contrast with *workspace preferences,* which are the same regardless of what project is being edited
    - e.g., editor colors
    - Set in **Window▶ Preferences** (on Mac, **Eclipse▶ Preferences**)
    - Careful! Dialog is very similar

Converting to a Fortran Project

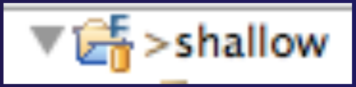
- Are there categories labeled **Fortran General** and **Fortran Build** in the project properties?

No Fortran categories →



Do this once

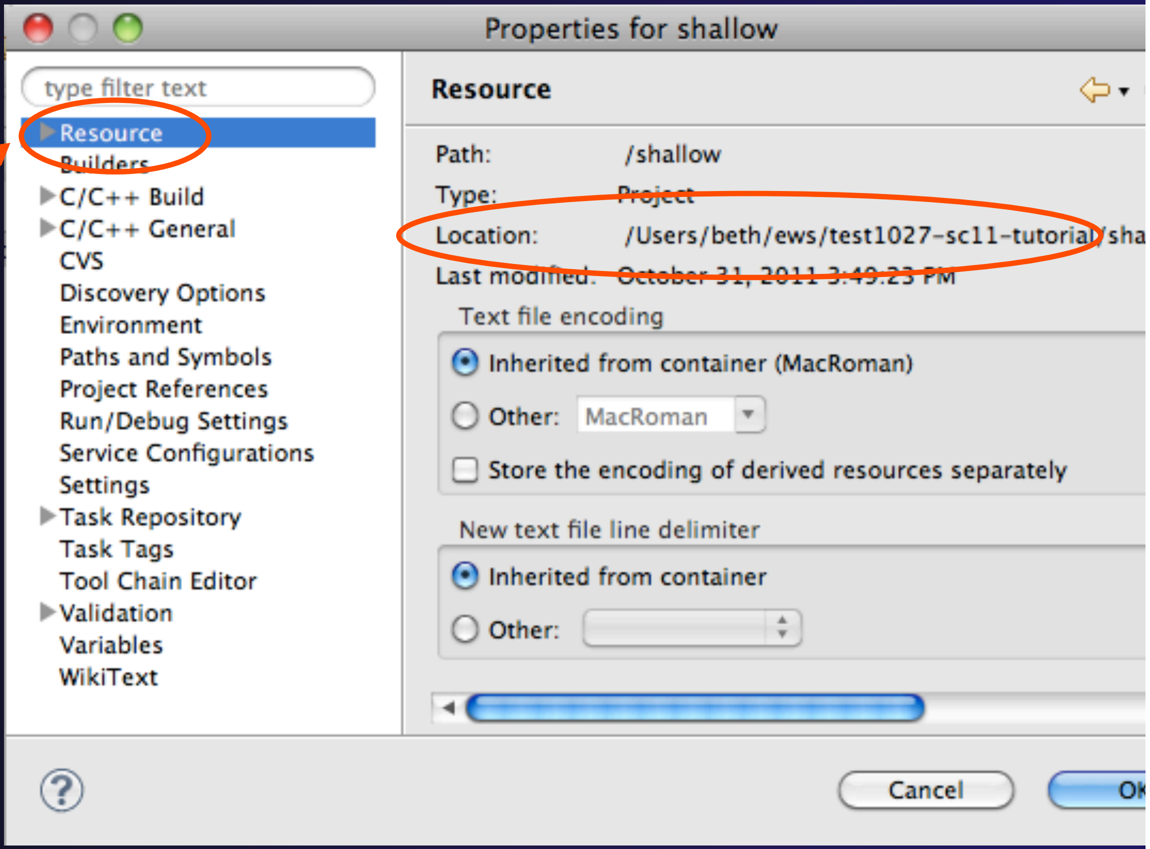
- If not, the project is not a Fortran Project
  - Switch to the Fortran Perspective
  - In the Fortran Projects view, right-click on the project, and click **Convert to Fortran Project**
  - Don’t worry; it’s still a C/C++ project, too



- *Every* Fortran project is also a C/C++ Project

Project Location

- How to tell where a project resides?
- In the project properties dialog, select the **Resource** category



Properties for shallow

type filter text

- Resource
- Builders
- C/C++ Build
- C/C++ General
- CVS
- Discovery Options
- Environment
- Paths and Symbols
- Project References
- Run/Debug Settings
- Service Configurations
- Settings
- Task Repository
- Task Tags
- Tool Chain Editor
- Validation
- Variables
- WikiText

Resource

Path:	/shallow
Type:	Project
Location:	/Users/beth/ews/test1027-sc11-tutorial/sha
Last modified:	October 31, 2011 3:49:23 PM

Text file encoding

[✓ Inherited from container (MacRoman)]
Other: MacRoman
Store the encoding of derived resources separately

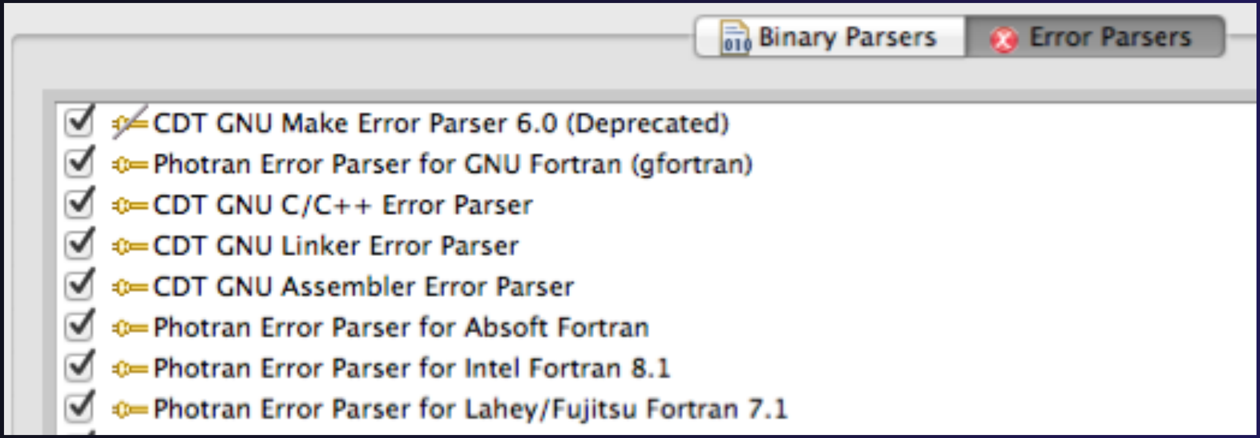
New text file line delimiter

[✓ Inherited from container]
Other:

Error Parsers

- Are compiler errors not appearing in the Problems view?
  - Make sure the correct *error parser* is enabled
  - In the project properties, navigate to **C++ Build▶Settings** or **Fortran Build▶Settings**
  - Switch to the **Error Parsers** tab
  - Check the error parser(s) for your compiler(s)

Do this once



- ✓ CDT GNU Make Error Parser 6.0 (Deprecated)
- ✓ Photran Error Parser for GNU Fortran (gfortran)
- ✓ CDT GNU C/C++ Error Parser
- ✓ CDT GNU Linker Error Parser
- ✓ CDT GNU Assembler Error Parser
- ✓ Photran Error Parser for Absoft Fortran
- ✓ Photran Error Parser for Intel Fortran 8.1
- ✓ Photran Error Parser for Lahey/Fujitsu Fortran 7.1

Fortran Source Form Settings

- Fortran files are either *free form* or *fixed form;* some Fortran files are *preprocessed* (#define, #ifdef, etc.)
  - Source form determined by filename extension
  - Defaults are similar to most Fortran compilers:

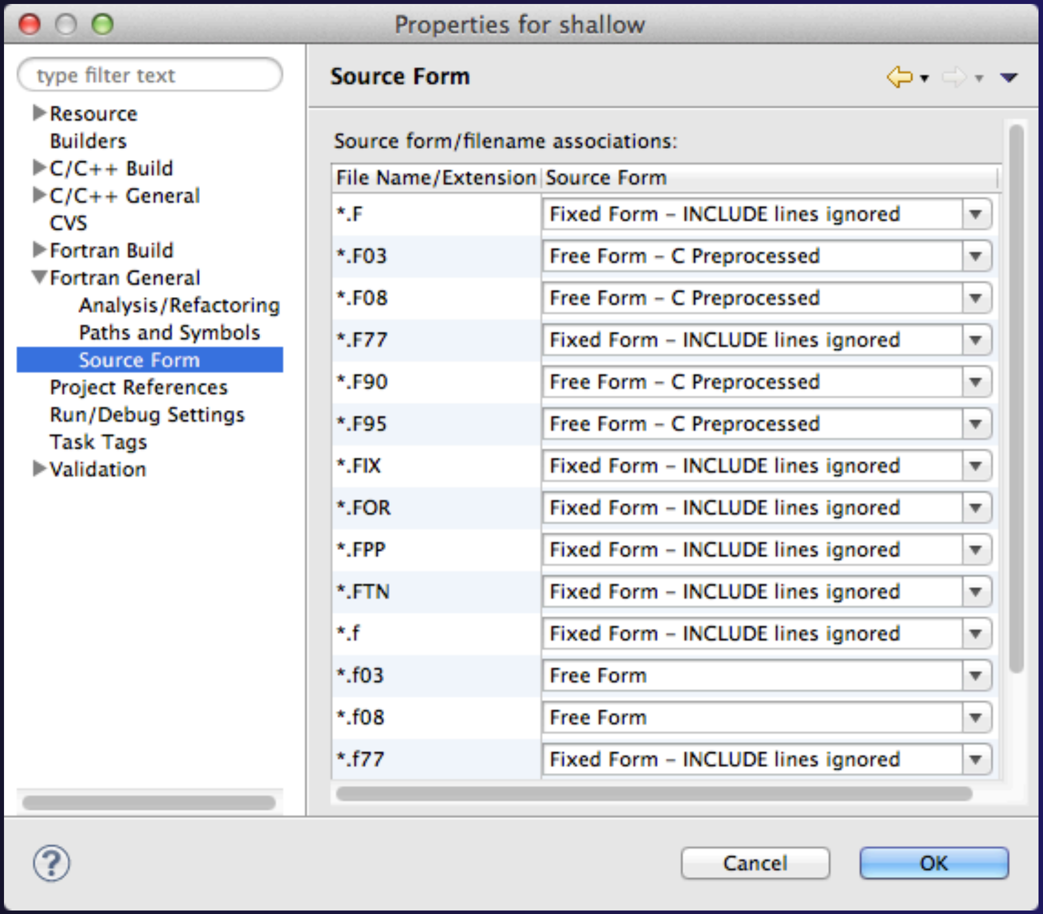
Fixed form:	.f	.fix	.for	.fpp	.ftn	.f77	
Free form:	.f08	.f03	.f95	.f90		< unpreprocessed	
	.F08		.F03	.F95	.F90		< preprocessed

- Many features *will not work* if filename extensions are associated with the wrong source form (outline view, content assist, search, refactorings, etc.)

Fortran Source Form Settings

Do this once

- In the project properties, select **Fortran General ▶ Source Form**
- Select source form for each filename extension
- Click **OK**



Source Form

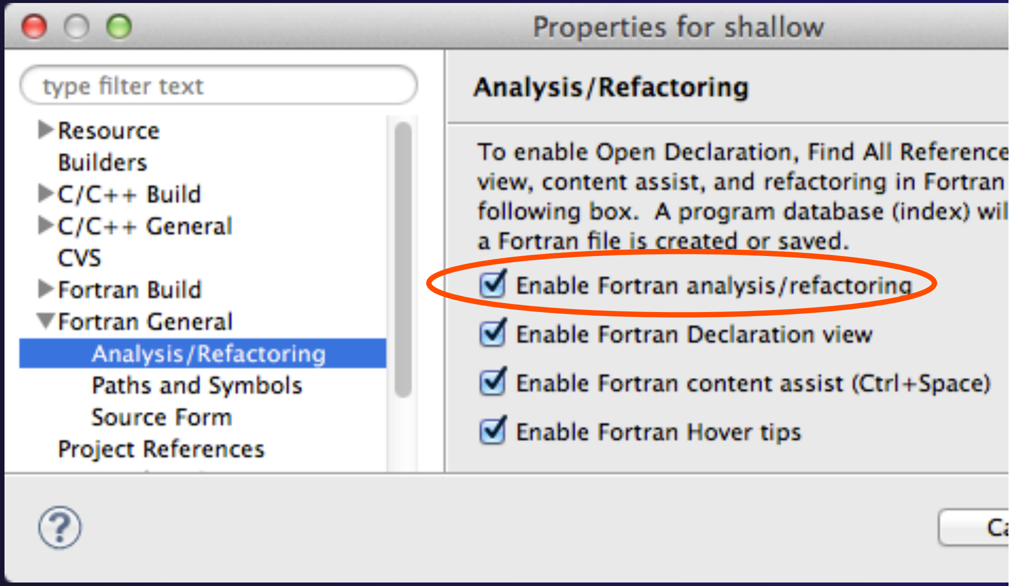
Source form/filename associations:

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

Enabling Fortran Advanced Features

- Some Fortran features are *disabled* by default
- Must be explicitly enabled
  - In the project properties dialog, select **Fortran General ▶ Analysis/Refactoring**
  - Click **Enable Analysis/ Refactoring**
  - Close and re-open any Fortran editors
- This turns on the "Photran Indexer"
  - Turn it off if it's slow

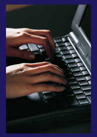
Do this once



Analysis/Refactoring

To enable Open Declaration, Find All Reference
view, content assist, and refactoring in Fortran
following box. A program database (index) wil
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



Exercise

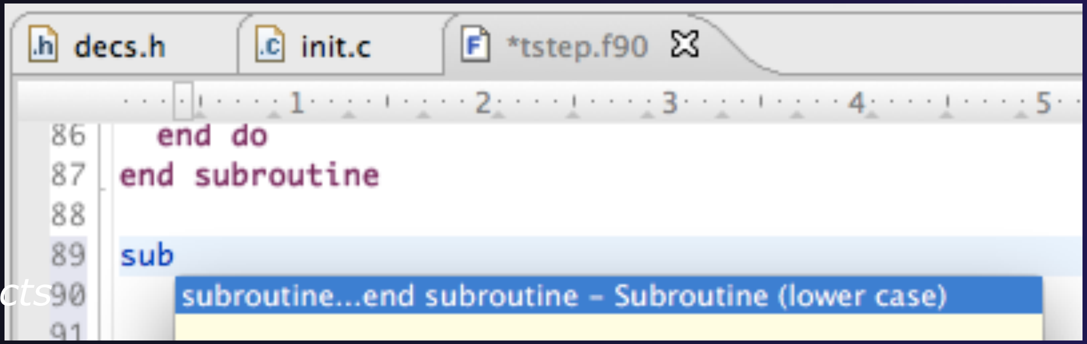
1. Convert shallow to a Fortran project
2. Make sure errors from the GNU Fortran compiler will be recognized
3. Make sure *.f90 files are treated as “Free Form” which is unpreprocessed
4. Make sure search and refactoring will work in Fortran

Advanced Editing

Code Templates

Code Templates
(C/C++ and Fortran)

- Auto-complete common code patterns
  - For loops/do loops, if constructs, etc.
  - Also MPI code templates
- Included with content assist proposals (when **Ctrl-Space** is pressed)
  - E.g., after the last line in tstep.f90, type "sub" and press **Ctrl-Space**
  - Press **Enter** to insert the template



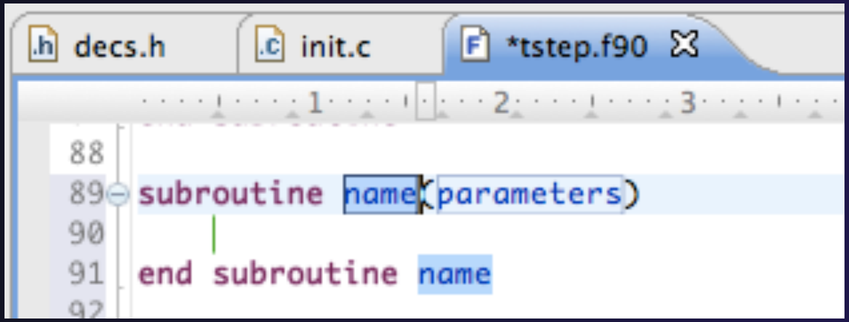
```
  end do
end subroutine

sub
```

```
subroutine...end subroutine - Subroutine (lower case)
```

Code Templates (2)

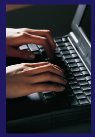
(C/C++ and Fortran)

- After pressing enter to insert the code template, completion fields are highlighted



```
subroutine name(parameters)

end subroutine name
```

- Press **Tab** to move between completion fields
- Changing one instance of a field changes all occurrences

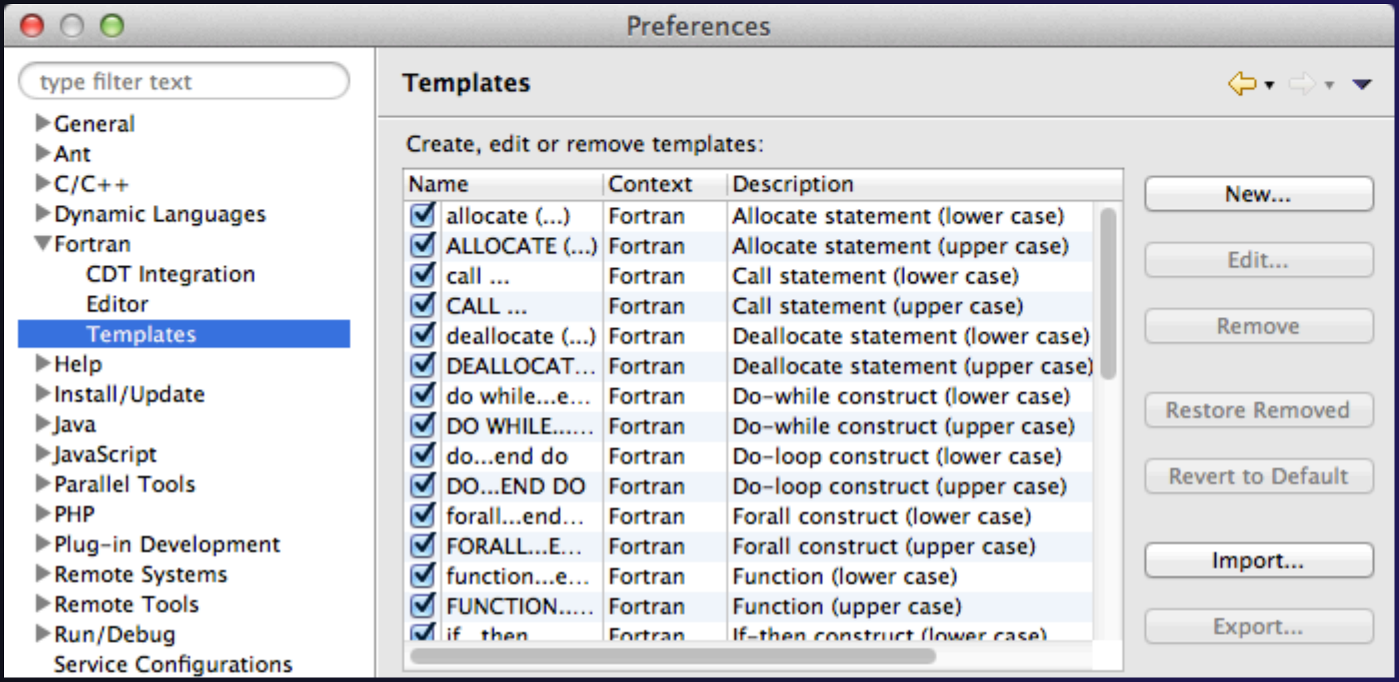


Exercise

- Open tstep.f90 and retype the last loop nest
  - Use the code template to complete the do-loops
  - Use content assist to complete variable names

Custom Code Templates
(Fortran)

- Customize code templates in **Window ▶ Preferences ▶ Fortran ▶ Templates**



Create, edit or remove templates:

Name	Context	Description
✓ allocate (...)	Fortran	Allocate statement (lower case)
✓ ALLOCATE (...)	Fortran	Allocate statement (upper case)
✓ call ...	Fortran	Call statement (lower case)
✓ CALL ...	Fortran	Call statement (upper case)
✓ deallocate (...)	Fortran	Deallocate statement (lower case)
✓ DEALLOCAT...	Fortran	Deallocate statement (upper case)
✓ do while...e...	Fortran	Do-while construct (lower case)
✓ DO WHILE......	Fortran	Do-while construct (upper case)
✓ do...end do	Fortran	Do-loop construct (lower case)
✓ DO...END DO	Fortran	Do-loop construct (upper case)
✓ forall...end...	Fortran	Forall construct (lower case)
✓ FORALL...E...	Fortran	Forall construct (upper case)
✓ function...e...	Fortran	Function (lower case)
✓ FUNCTION.....	Fortran	Function (upper case)
✓ if...then	Fortran	If-then construct (lower case)

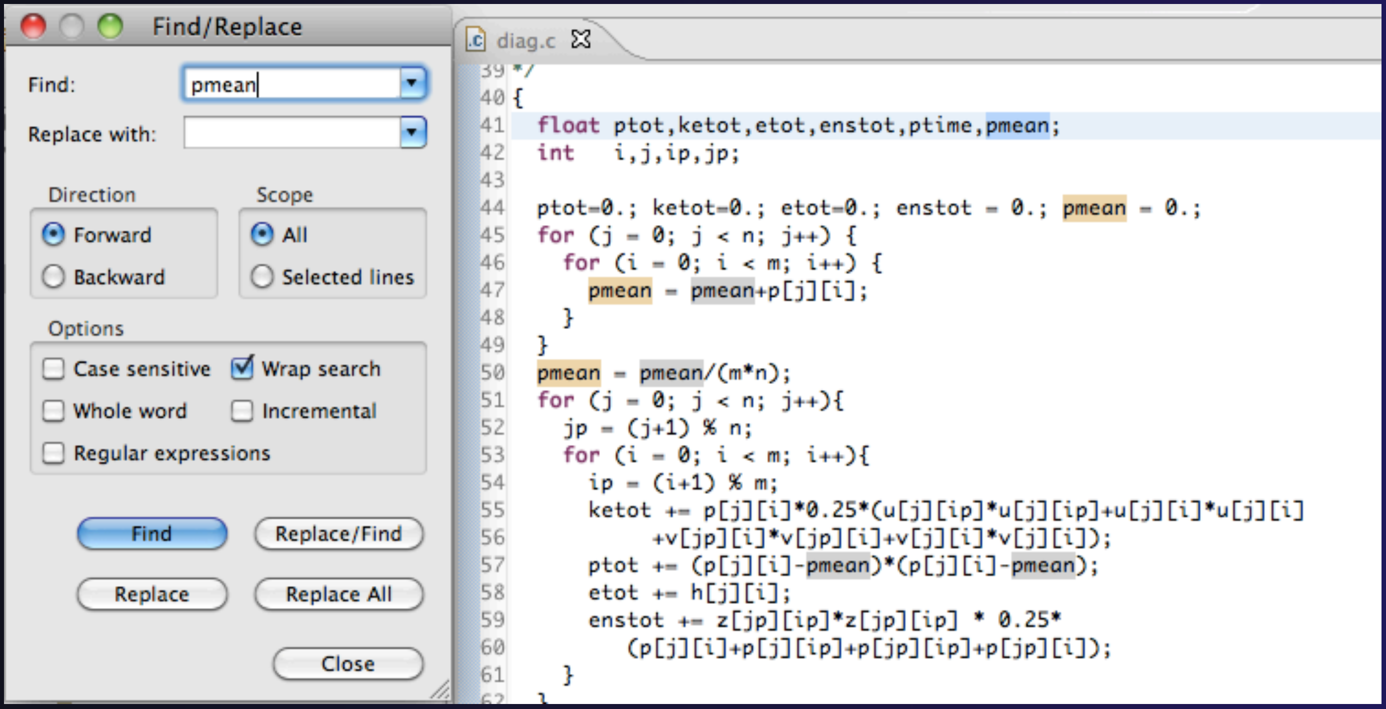
- Can import/export templates to XML files

Search & Refactoring

- Objectives
  - Develop proficiency using Eclipse’s textual and language-based search and navigation capabilities
  - Introduce common automated refactorings
- Contents
  - Searching
  - Refactoring and Transformation
- Prerequisites
  - Basics
  - Fortran

Find/Replace within Editor

- Simple Find within editor buffer
- Ctrl-F (Mac: Command-F)



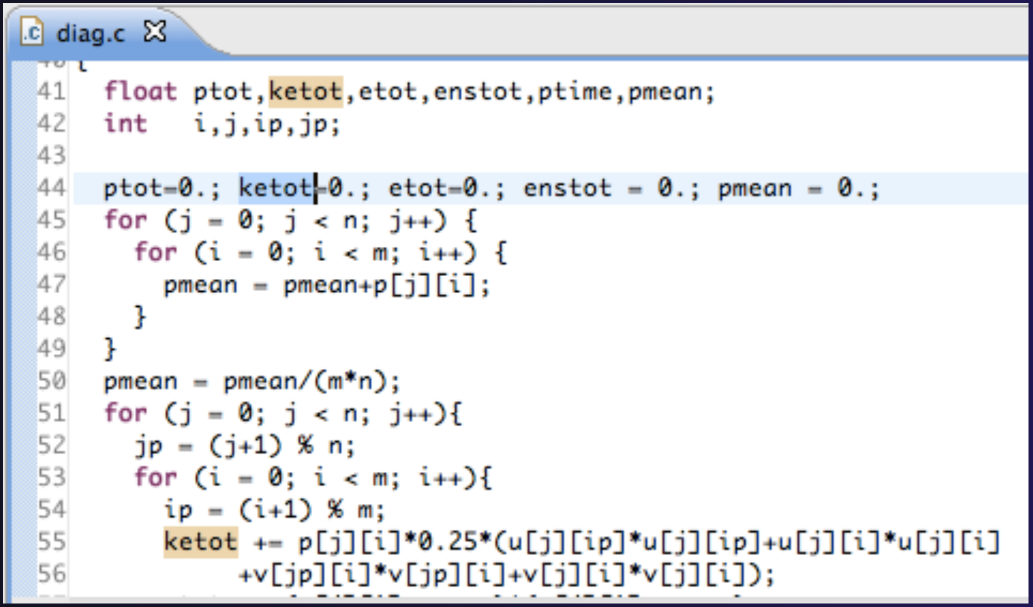
Find: pmean
Replace with:
Direction: [✓ Forward] Backward
Scope: [✓ All] Selected lines
Options: Case sensitive, [✓ Wrap search], Whole word, Incremental, Regular expressions

```
*/
{
  float ptot,ketot,etot,enstot,ptime,pmean;
  int   i,j,ip,jp;

  ptot=0.; ketot=0.; etot=0.; enstot = 0.; pmean = 0.;
  for (j = 0; j < n; j++) {
    for (i = 0; i < m; i++) {
      pmean = pmean+p[j][i];
    }
  }
  pmean = pmean/(m*n);
  for (j = 0; j < n; j++){
    jp = (j+1) % n;
    for (i = 0; i < m; i++){
      ip = (i+1) % m;
      ketot += p[j][i]*0.25*(u[j][ip]*u[j][ip]+u[j][i]*u[j][i]
           +v[jp][i]*v[jp][i]+v[j][i]*v[j][i]);
      ptot += (p[j][i]-pmean)*(p[j][i]-pmean);
      etot += h[j][i];
      enstot += z[jp][ip]*z[jp][ip] * 0.25*
         (p[j][i]+p[j][ip]+p[jp][ip]+p[jp][i]);
    }
  }
```

Mark Occurrences

(C/C++ Only)

- 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 (Mac: Alt-Cmd-O)

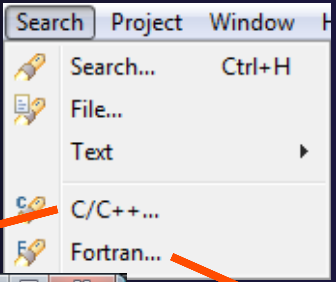


```
  float ptot,ketot,etot,enstot,ptime,pmean;
  int   i,j,ip,jp;

  ptot=0.; ketot=0.; etot=0.; enstot = 0.; pmean = 0.;
  for (j = 0; j < n; j++) {
    for (i = 0; i < m; i++) {
      pmean = pmean+p[j][i];
    }
  }
  pmean = pmean/(m*n);
  for (j = 0; j < n; j++){
    jp = (j+1) % n;
    for (i = 0; i < m; i++){
      ip = (i+1) % m;
      ketot += p[j][i]*0.25*(u[j][ip]*u[j][ip]+u[j][i]*u[j][i]
            +v[jp][i]*v[jp][i]+v[j][i]*v[j][i]);
```

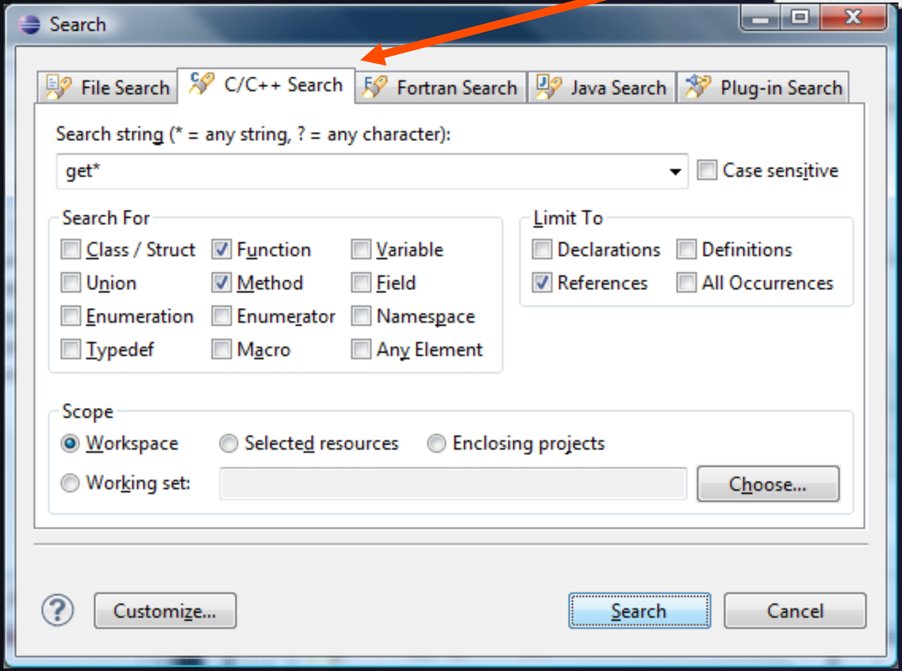
Language-Based Searching

(C/C++ and Fortran)

- "Knows" what things can be declared in each language (functions, variables, classes, modules, etc.)



- E.g., search for every call to a function whose name starts with "get"
- Search can be project- or workspace-wide



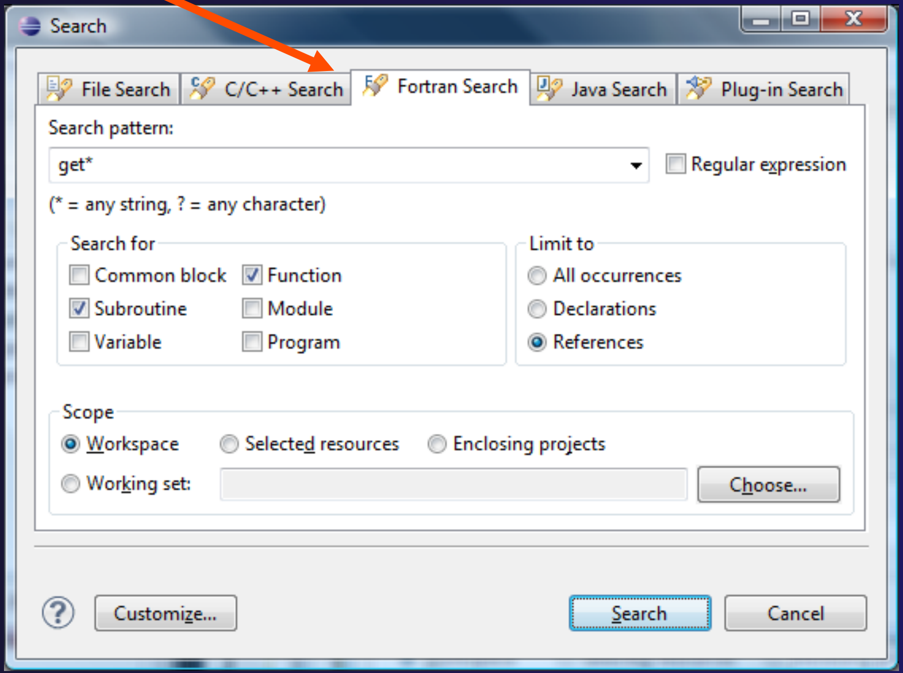
Search string (* = any string, ? = any character): get*

☐ 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: Choose...



Search pattern: get*

☐ Regular expression

(* = any string, ? = any character)

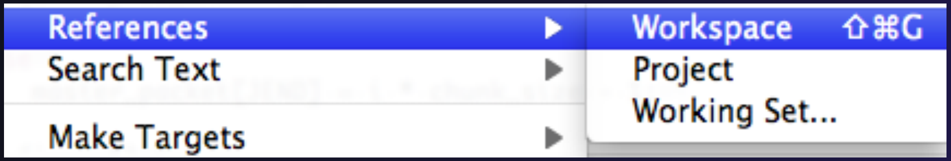
Search for: ☐ Common block, ✓ Function, ✓ Subroutine, ☐ Module, ☐ Variable, ☐ Program

Limit to: All occurrences Declarations [✓ References]

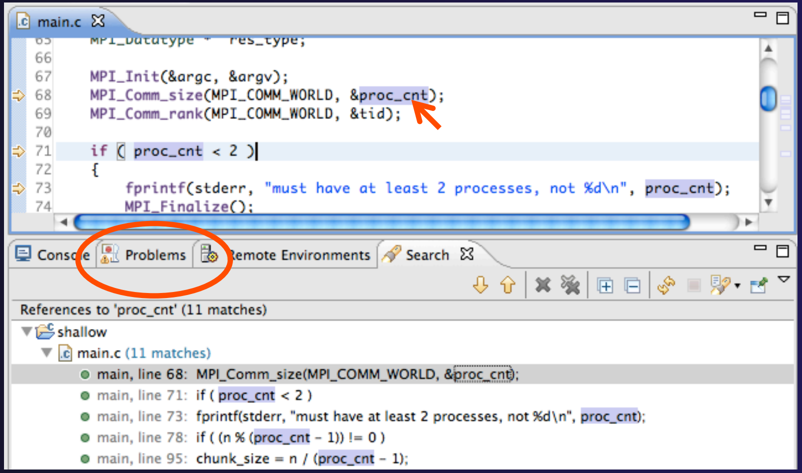
Scope: [✓ Workspace] Selected resources Enclosing projects Working set: Choose...

Find References
(C/C++ and Fortran)

- Finds all of the places where a variable, function, etc., is used
  - Right-click on an identifier in the editor
  - Click **References▶Workspace** or **References▶Project**



  - **Search** view shows matches



```
    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();
```

```
References to 'proc_cnt' (11 matches)
shallow
  main.c (11 matches)
    main, line 68:  MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
    main, line 71:  if ( proc_cnt < 2 )
    main, line 73:  fprintf(stderr, "must have at least 2 processes, not %d\n", proc_cnt);
    main, line 78:  if ( (n % (proc_cnt - 1)) != 0 )
    main, line 95:  chunk_size = n / (proc_cnt - 1);
```

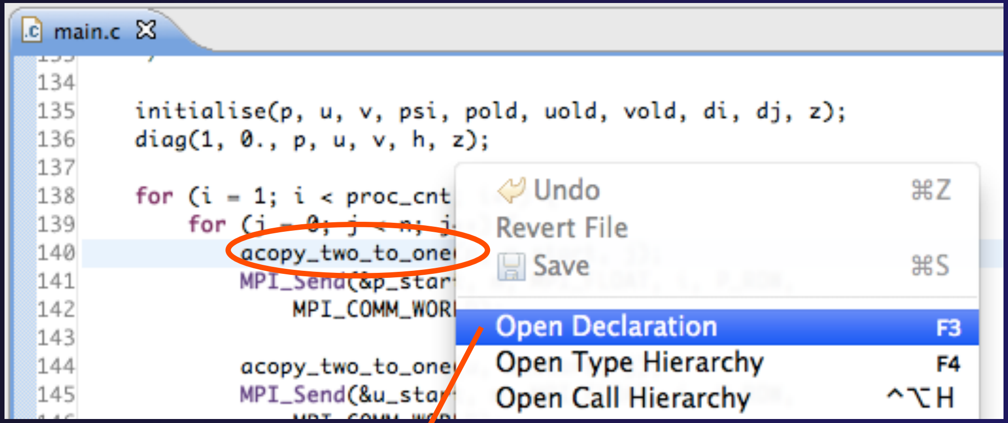
Open Declaration

(C/C++ and Fortran)

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

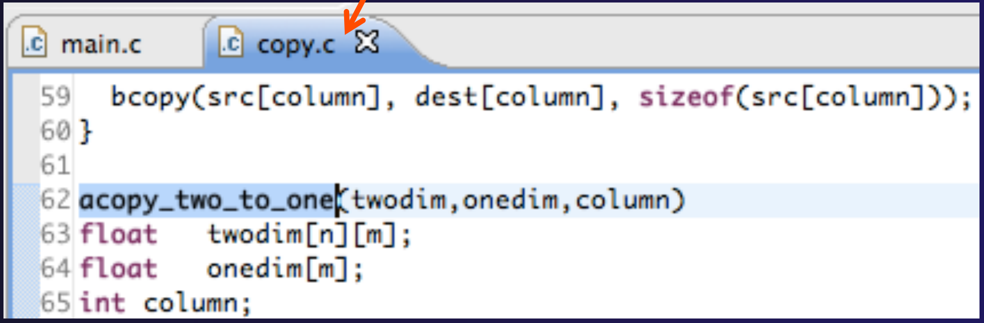
- Left-click to select identifier
- Right-click on identifier
- Click **Open Declaration**

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

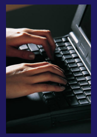


```
    initialise(p, u, v, psi, pold, uold, vold, di, dj, z);
    diag(1, 0., p, u, v, h, z);

    for (i = 1; i < proc_cnt
      for (j = 0; j < n; j
          acopy_two_to_one
          MPI_Send(&p_star
              MPI_COMM_WOR

          acopy_two_to_one
          MPI_Send(&u_sta
```

Goes to its declaration in copy.c



```
  bcopy(src[column], dest[column], sizeof(src[column]));
}

acopy_two_to_one(twodim,onedim,column)
float    twodim[n][m];
float    onedim[m];
int column;
```



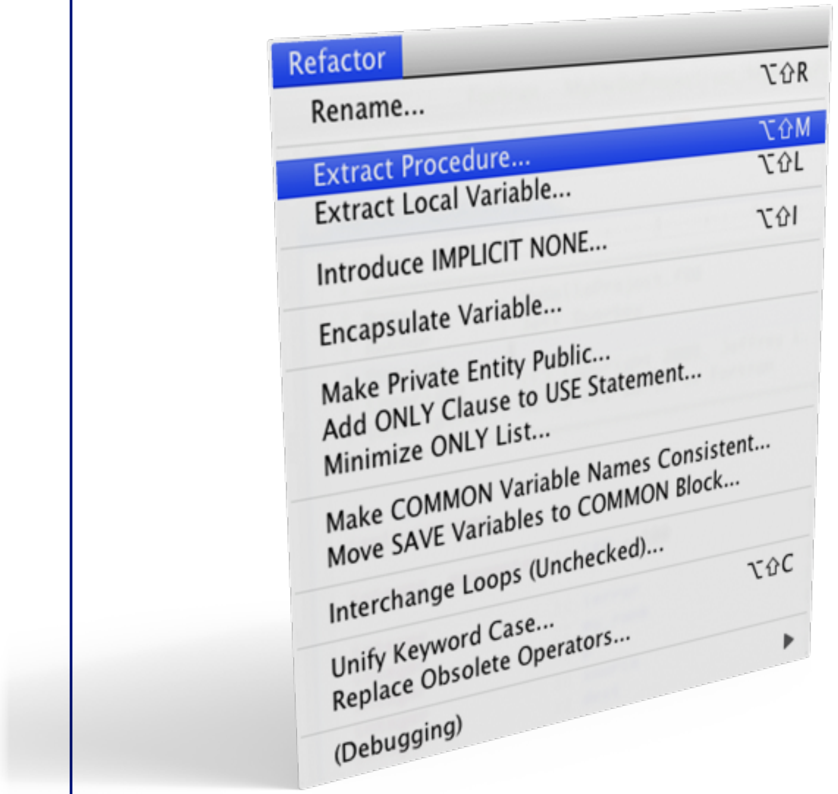
Search – Try It!

1. Find every call to `MPI_Recv` in Shallow.
2. In worker.c, on line 42, there is a declaration `float p[n][m]`.
   a) What is `m` (local? global? function parameter?)
   b) Where is `m` defined?
   c) How many times is `m` used in the project?
3. Find every C function in Shallow whose name contains the word `time`

Refactoring and Transformation

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 7.0: 31 refactorings

Refactoring Caveats

- Photran can only refactor free form code that is *not* preprocessed
  - Determined by Source Form settings
    (recall from earlier that these are configured in **Project Properties: Fortran General▶Source Form**)

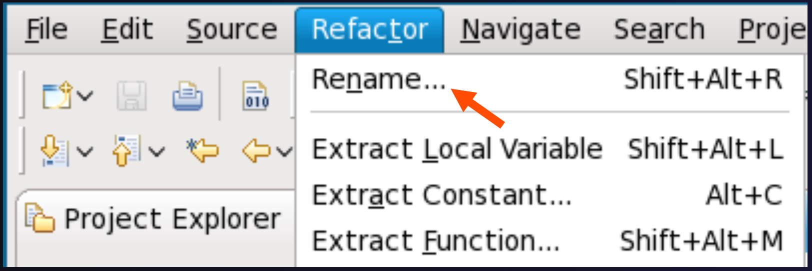
✓	**Free Form, Unpreprocessed:**			.f08	.f03	.f95	.f90
✖	**Free Form, Preprocessed:**			.F08	.F03	.F95	.F90
✖	**Fixed Form:**	.f	.fix	.for	.fpp	.ftn	.f77

- Refactor menu will be empty if
  - Refactoring not enabled in project properties
    (recall from earlier that it is enabled in **Project Properties: Fortran General▶Analysis/Refactoring**)
  - The file in the active editor is fixed form
  - The file in the active editor is preprocessed

Rename Refactoring

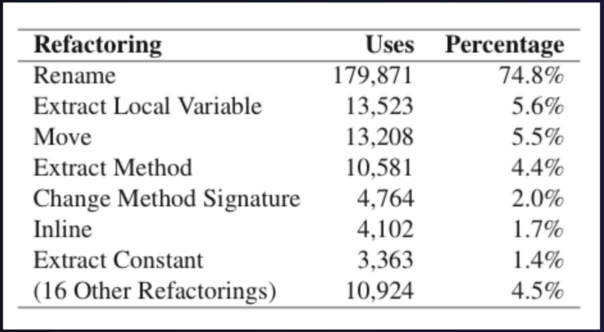
(also available in Fortran)

- 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.)


File
Edit
Source
Refactor
Navigate
Search
Proje
Rename... Shift+Alt+R
Extract Local Variable Shift+Alt+L
Extract Constant... Alt+C
Extract Function... Shift+Alt+M
Project Explorer

- Switch to C/C++ Perspective
- Open a source file
- In the editor, click on a variable or function name
- Select menu item **Refactor▶Rename**
  - Or use context menu
- Enter new name

In Java (Murphy-Hill et al., ICSE 2008):

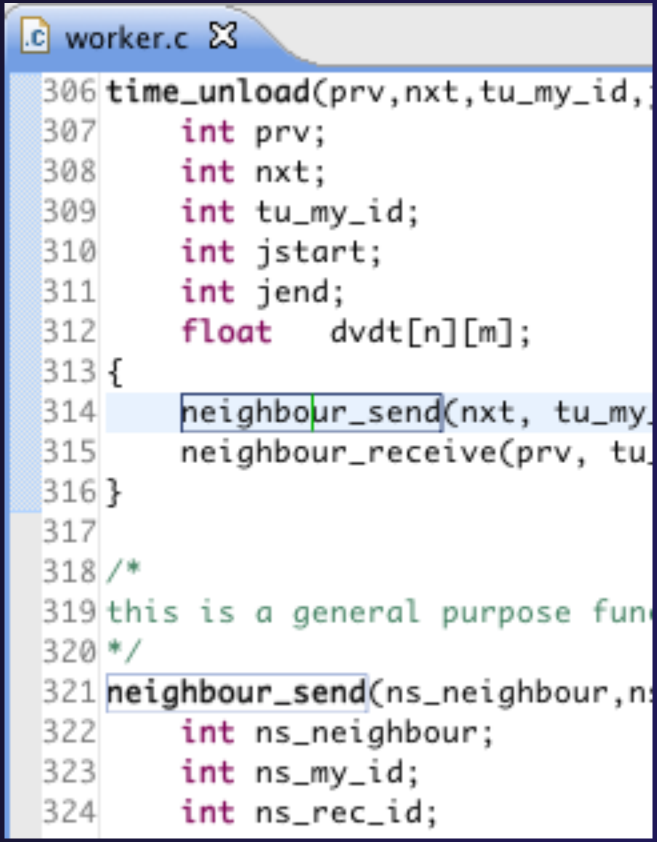


Refactoring	Uses	Percentage
Rename	179,871	74.8%
Extract Local Variable	13,523	5.6%
Move	13,208	5.5%
Extract Method	10,581	4.4%
Change Method Signature	4,764	2.0%
Inline	4,102	1.7%
Extract Constant	3,363	1.4%
(16 Other Refactorings)	10,924	4.5%

Rename in File

(C/C++ Only)

- Position the caret over an identifier.
- Press **Ctrl-1** (**Command-1** on Mac).
- Enter a new name. Changes are propagated within the file as you type.

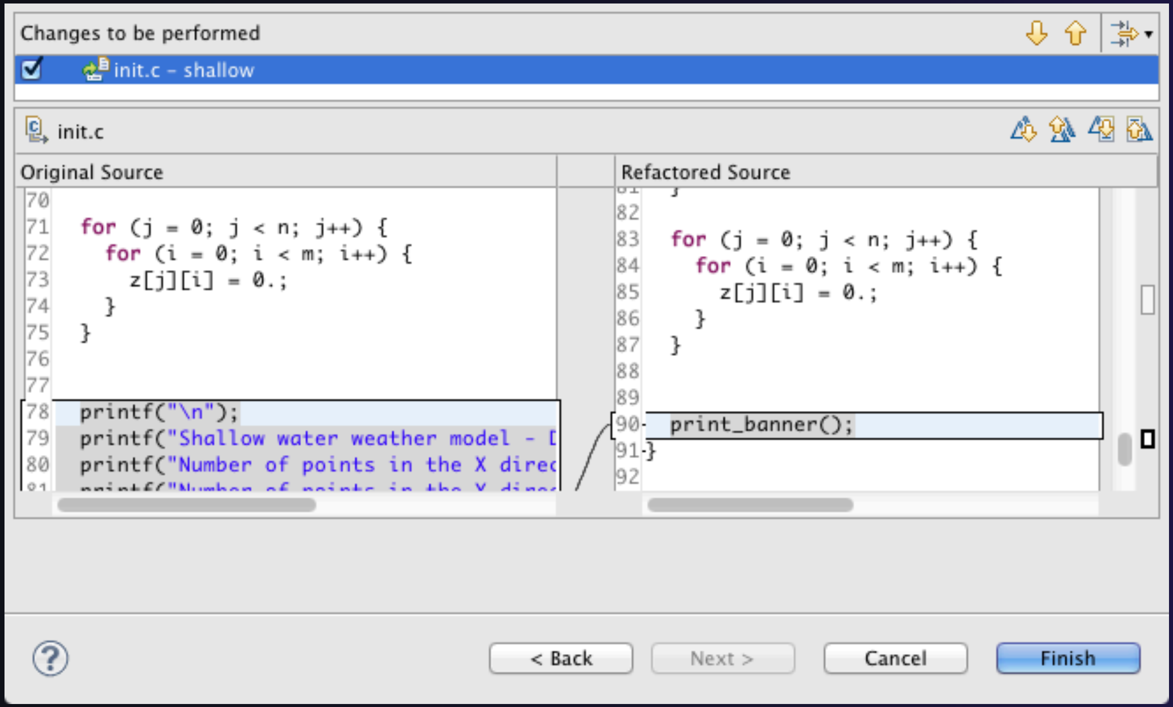


```
time_unload(prv,nxt,tu_my_id,
    int prv;
    int nxt;
    int tu_my_id;
    int jstart;
    int jend;
    float   dvdt[n][m];
{
    neighbour_send(nxt, tu_my
    neighbour_receive(prv, tu
}

/*
this is a general purpose fun
*/
neighbour_send(ns_neighbour,n
    int ns_neighbour;
    int ns_my_id;
    int ns_rec_id;
```

Extract Function Refactoring

(also available in Fortran - “Extract Procedure”)

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



Original Source:

```
  for (j = 0; j < n; j++) {
    for (i = 0; i < m; i++) {
      z[j][i] = 0.;
    }
  }


  printf("\n");
  printf("Shallow water weather model - [
  printf("Number of points in the X direc
```

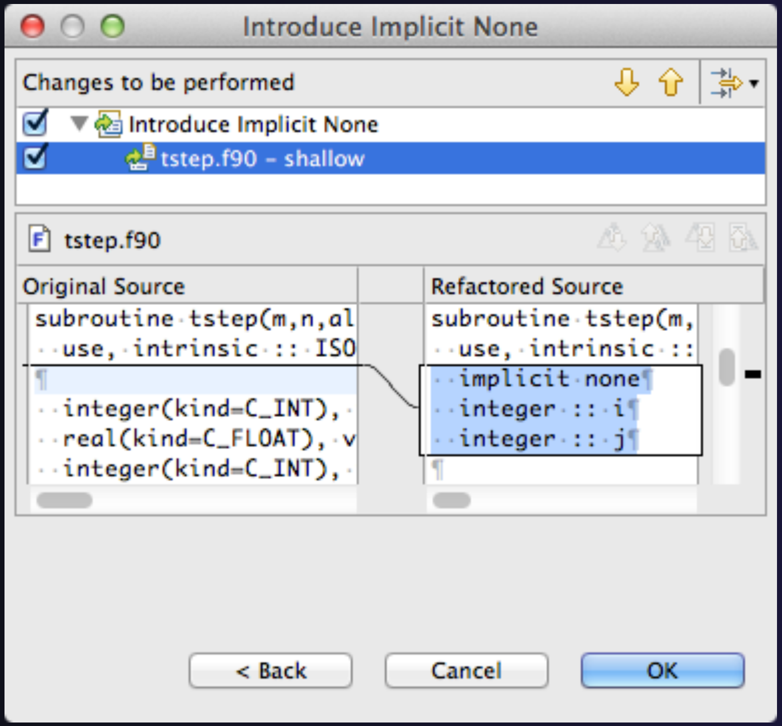
Refactored Source:

```
  for (j = 0; j < n; j++) {
    for (i = 0; i < m; i++) {
      z[j][i] = 0.;
    }
  }


  print_banner();
}
```

- Select a sequence of statements
- Select menu item **Refactor ▶ Extract Function…**
- 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:

```
subroutine tstep(m,n,al
  use, intrinsic :: ISO

  integer(kind=C_INT), 
  real(kind=C_FLOAT), v
  integer(kind=C_INT), 
```

Refactored Source:

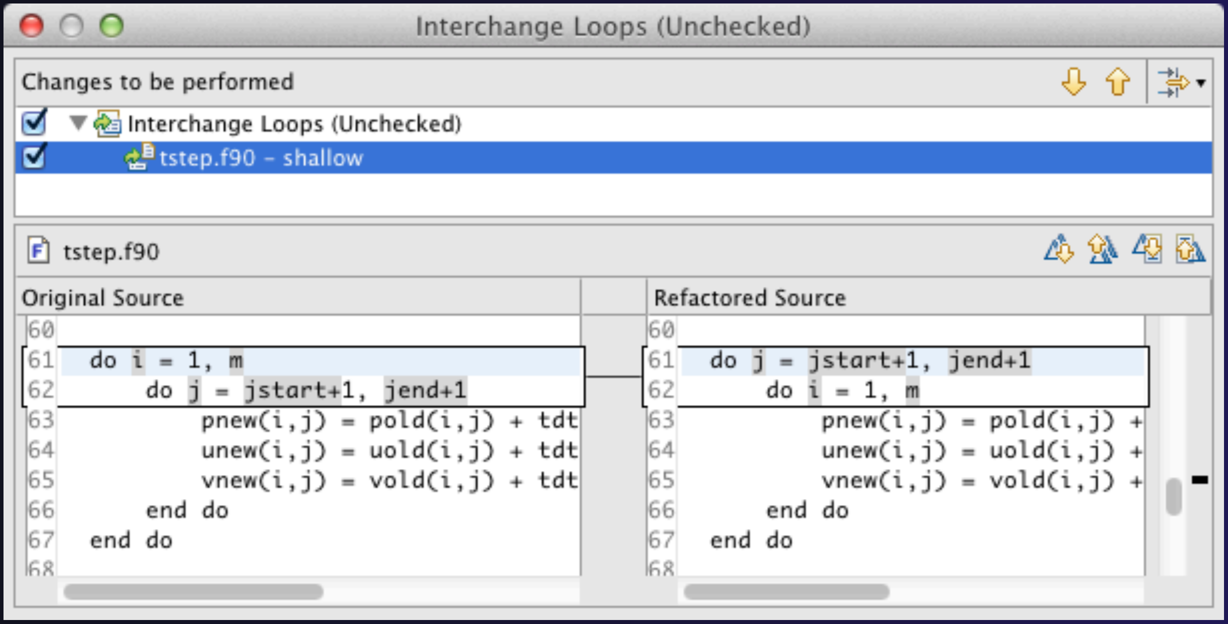
```
subroutine tstep(m,
  use, intrinsic ::
  implicit none
  integer :: i
  integer :: j
```

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

Loop Transformations

(Fortran only)

- **Interchange Loops** **CAUTION**: No check for behavior preservation
  - Swaps the loop headers in a two-loop nest
  - Select the loop nest, click menu item **Refactor ▸ Do Loop ▸ Interchange Loops (Unchecked)…**



Original Source:

```
    do i = 1, m
        do j = jstart+1, jend+1
            pnew(i,j) = pold(i,j) + tdt
            unew(i,j) = uold(i,j) + tdt
            vnew(i,j) = vold(i,j) + tdt
        end do
    end do
```

Refactored Source:

```
    do j = jstart+1, jend+1
        do i = 1, m
            pnew(i,j) = pold(i,j) +
            unew(i,j) = uold(i,j) +
            vnew(i,j) = vold(i,j) +
        end do
    end do
```

Old version traverses matrices in row-major order

New version traverses in column-major order (better cache performance)

Loop Transformations

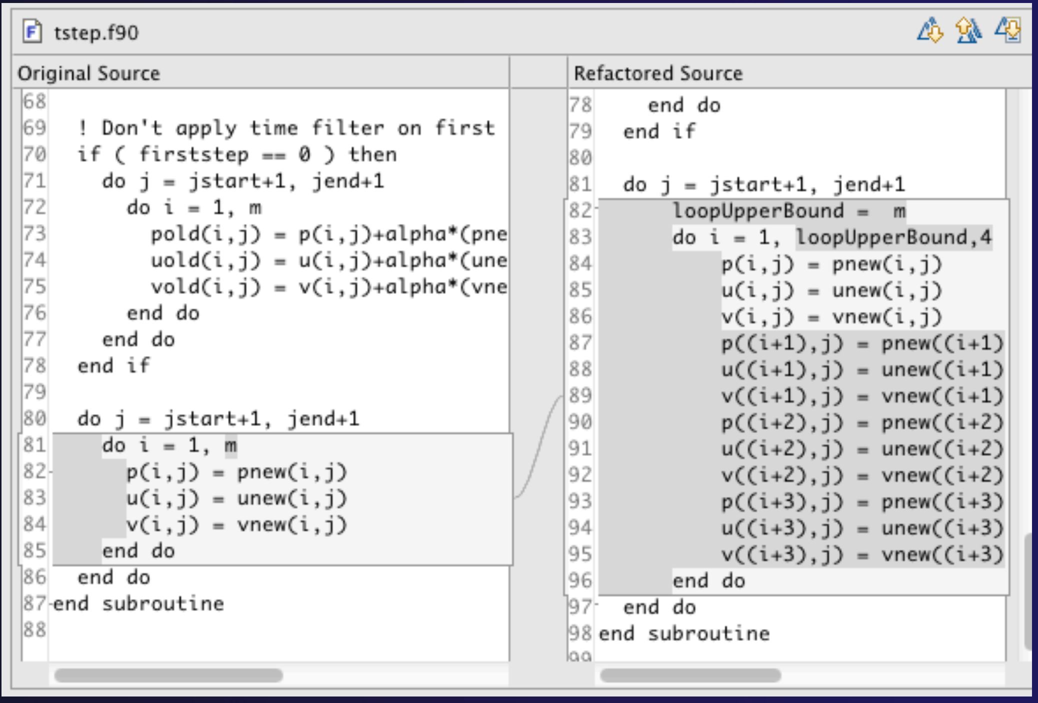
(Fortran only)

- **Unroll Loop**
- Select a loop, click **Refactor ▶ Do Loop ▶ Unroll Loop…**

```
do i = 1, 10
  print *, 10*i
end do
```

Unroll 4×

```
do i = 1, 10, 4
  print *, 10*i
  print *, 10*(i+1)
  print *, 10*(i+2)
  print *, 10*(i+3)
end do
```

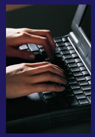


Original Source

```

  ! Don't apply time filter on first
  if ( firststep == 0 ) then
    do j = jstart+1, jend+1
      do i = 1, m
        pold(i,j) = p(i,j)+alpha*(pne
        uold(i,j) = u(i,j)+alpha*(une
        vold(i,j) = v(i,j)+alpha*(vne
      end do
    end do
  end if

  do j = jstart+1, jend+1
    do i = 1, m
      p(i,j) = pnew(i,j)
      u(i,j) = unew(i,j)
      v(i,j) = vnew(i,j)
    end do
  end do
end subroutine
```

Refactored Source

```
    end do
  end if

  do j = jstart+1, jend+1
      loopUpperBound =  m
      do i = 1, loopUpperBound,4
          p(i,j) = pnew(i,j)
          u(i,j) = unew(i,j)
          v(i,j) = vnew(i,j)
          p((i+1),j) = pnew((i+1)
          u((i+1),j) = unew((i+1)
          v((i+1),j) = vnew((i+1)
          p((i+2),j) = pnew((i+2)
          u((i+2),j) = unew((i+2)
          v((i+2),j) = vnew((i+2)
          p((i+3),j) = pnew((i+3)
          u((i+3),j) = unew((i+3)
          v((i+3),j) = vnew((i+3)
      end do
  end do
end subroutine
```



Refactoring & Transformation – Try It!

In tstep.f90…

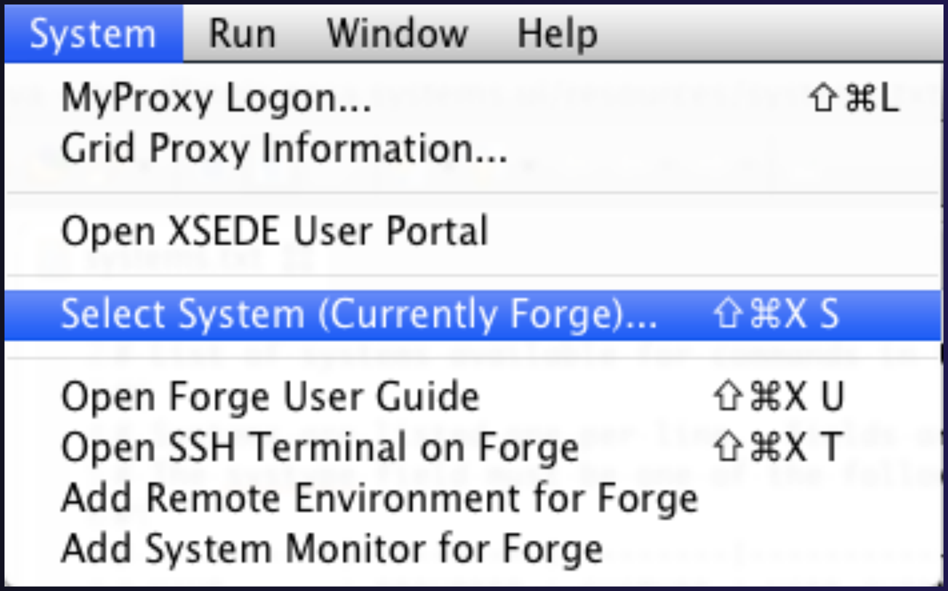
1. In init.c, extract the `printf` statements at the bottom of the file into a new function called `print_banner`
2. In worker.c, change the spellings of `neighbour_send` and `neighbour_receive` to American English
3. In tstep.f90, make the (Fortran) `tstep` subroutine IMPLICIT NONE

NCSA/XSEDE Features

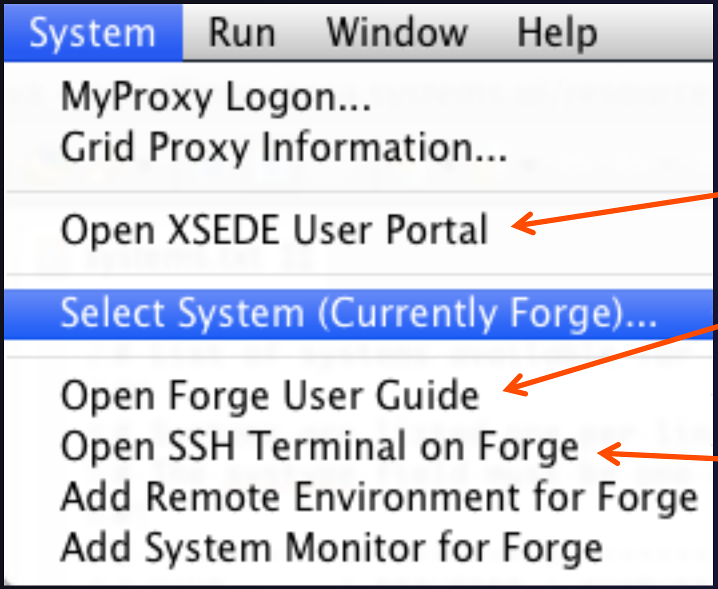
- Objectives
  - Install NCSA's GSI auth and XSEDE support plug-ins
  - Become familiar with the System menu
- Contents
  - Capabilities
  - Installation
- Prerequisites
  - (none)

Additional Plug-ins from NCSA

- NCSA publishes additional plug-ins can be added onto an existing PTP installation
- Contribute a **System** menu to the menu bar with XSEDE- and NCSA-specific commands


System
Run
Window
Help
MyProxy Logon... ⇧⌘L
Grid Proxy Information...
Open XSEDE User Portal
Select System (Currently Forge)... ⇧⌘X S
Open Forge User Guide ⇧⌘X U
Open SSH Terminal on Forge ⇧⌘X T
Add Remote Environment for Forge
Add System Monitor for Forge

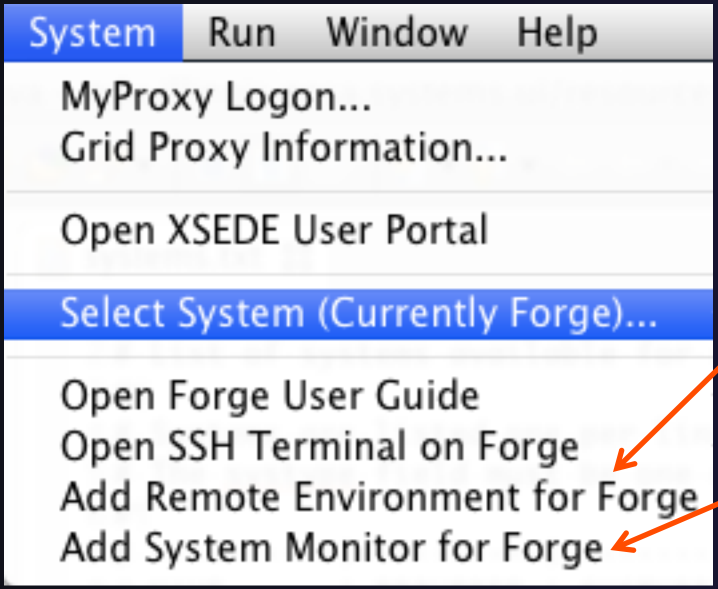
System Menu


System
Run
Window
Help
MyProxy Logon...
Grid Proxy Information...
Open XSEDE User Portal
Select System (Currently Forge)...
Open Forge User Guide
Open SSH Terminal on Forge
Add Remote Environment for Forge
Add System Monitor for Forge

- Open Web content in Eclipse:
  - **Open XSEDE User Portal**
  - **Open User Guide** for a machine
- Open an SSH terminal (as an Eclipse view)

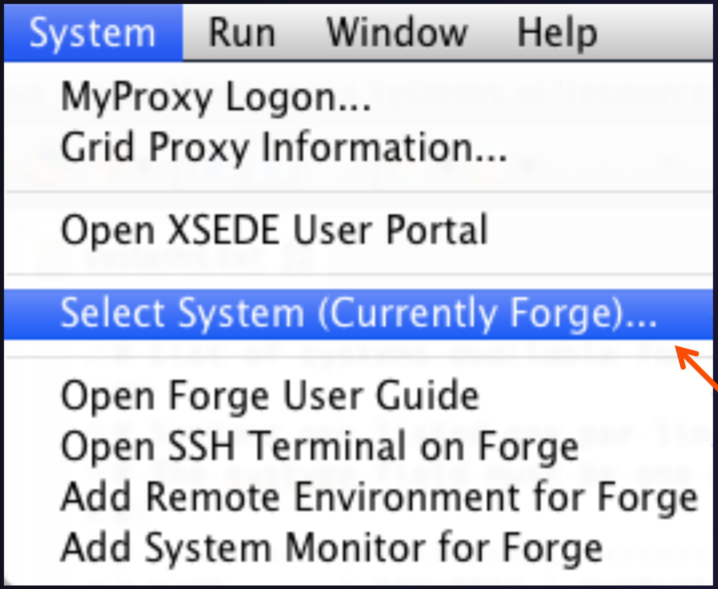
Eclipse-integrated SSH terminals are provided by the Remote System Explorer (RSE), one of the features that is included in the Eclipse for Parallel Application Developers package.

System Menu


System
Run
Window
Help
MyProxy Logon...
Grid Proxy Information...
Open XSEDE User Portal
Select System (Currently Forge)...
Open Forge User Guide
Open SSH Terminal on Forge
Add Remote Environment for Forge
Add System Monitor for Forge

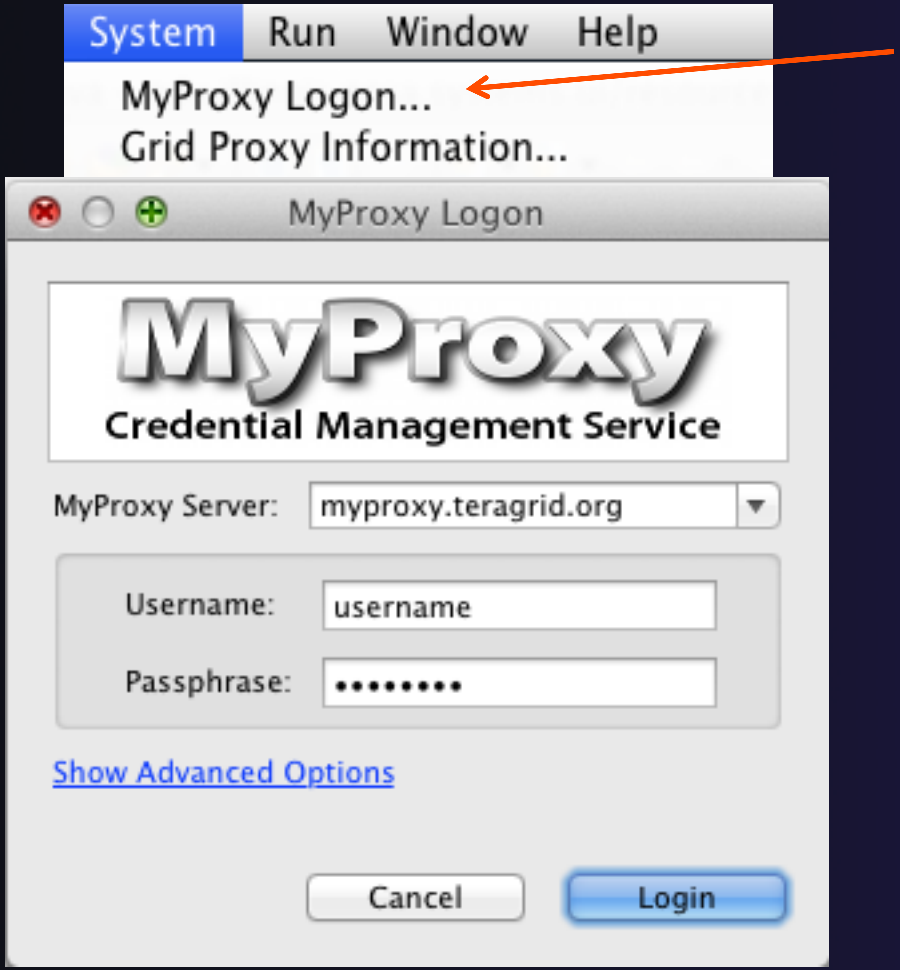
- Shortcuts for common PTP tasks:
  - **Add Remote Environment** adds a Remote Tools connection for a particular machine
  - **Add System Monitor** opens the System Monitoring perspective and begins monitoring a particular machine

System Menu


System
Run
Window
Help
MyProxy Logon...
Grid Proxy Information...
Open XSEDE User Portal
Select System (Currently Forge)...
Open Forge User Guide
Open SSH Terminal on Forge
Add Remote Environment for Forge
Add System Monitor for Forge

- The plug-in is preconfigured with information about XSEDE and NCSA resources
- The bottom four commands generally prompt for a system
- **Select System** can be used to eliminate this prompt, so these commands always act on a particular system

MyProxy Logon



MyProxy Server: myproxy.teragrid.org

Username: username

Passphrase: ••••••••

- **MyProxy Logon** allows you to authenticate with a MyProxy server
  - Often **myproxy.teragrid.org**
- It stores a "credential," which is usually valid for 12 hours
- During these 12 hours, SSH connections to XSEDE resources will not require a password; they can use the stored credential
  - However, you **must** enter the correct username for that machine!

Installation

1. Click **Help > Install New Software**
2. Click **Add** to open the Add Repository dialog
3. In the **Location** field, enter

   `http://forecaster.ncsa.uiuc.edu/updates/juno`

   and then click **OK** to close the Add dialog
4. Select the following:
   - GSI Authentication and MyProxy Logon Support
   - NCSA and XSEDE System Support
5. Click **Next** and complete the installation

Parallel Debugging

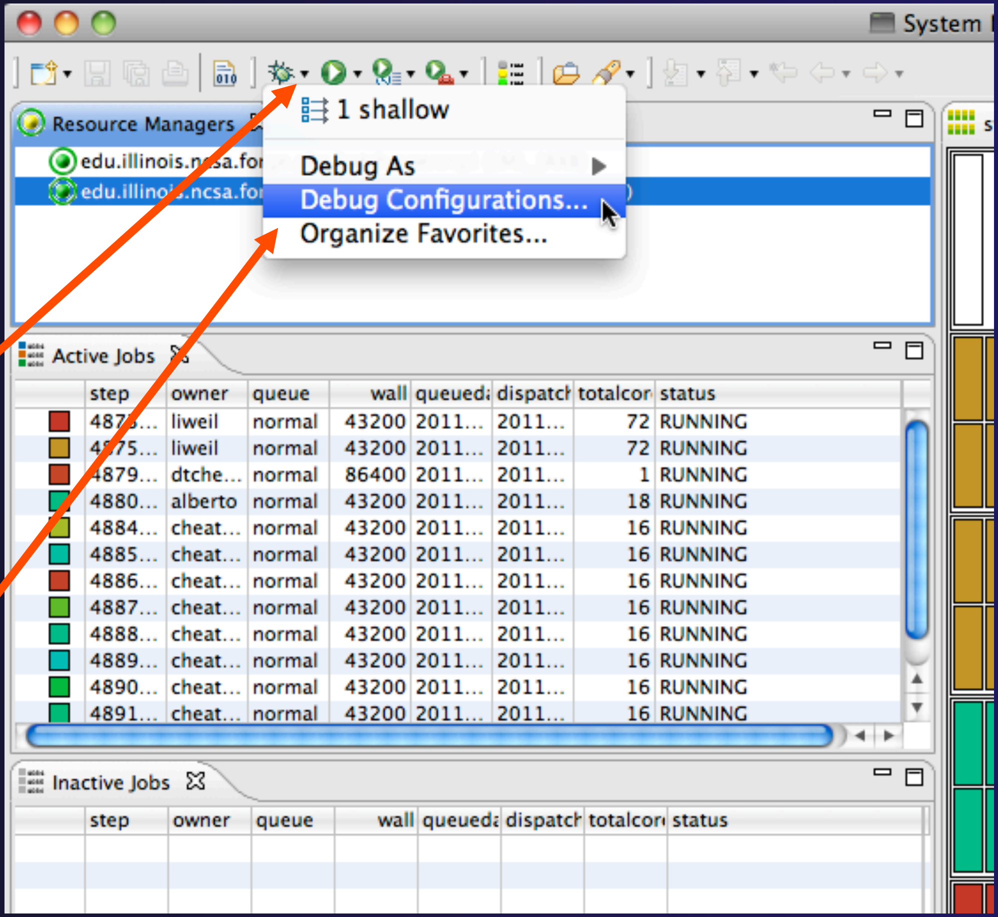
- Objective
  - Learn the basics of debugging parallel programs
- Contents
  - Launching a debug session
  - The Parallel Debug Perspective
  - Controlling sets of processes
  - Controlling individual processes
  - Parallel Breakpoints
  - Terminating processes

Debugging Setup

- Debugging requires interactive access to the application
- Can use any of the -*Interactive* target configurations
  - *Torque-Generic-Interactive*
  - *PBS-Generic-Interactive*
  - *OpenMPI-Generic-Interactive*

Create a Debug Configuration

- A debug configuration is essentially the same as a run configuration (like we used in the *Running an Application* module)
- It is possible to re-use an 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



Active Jobs

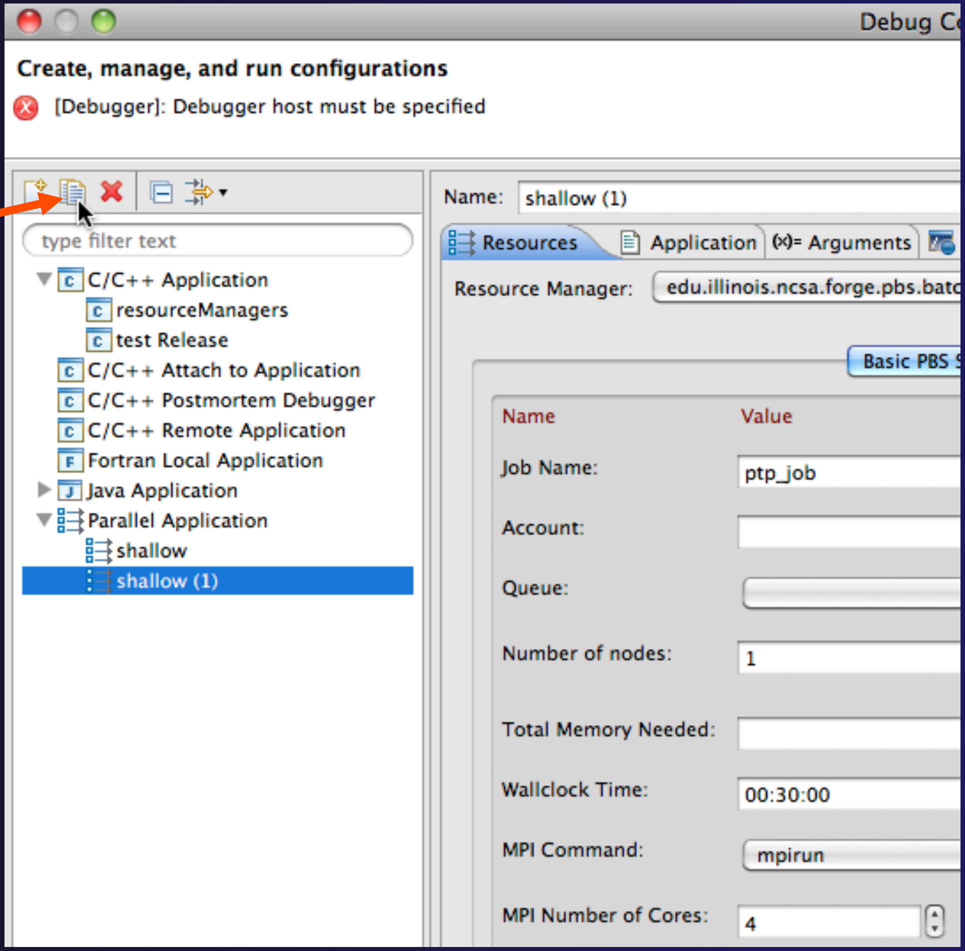
	step	owner	queue	wall	queued:	dispatch	totalcor	status
🟥	487[illegible]...	liweil	normal	43200	2011...	2011...	72	RUNNING
🟧	4875...	liweil	normal	43200	2011...	2011...	72	RUNNING
🟥	4879...	dtche...	normal	86400	2011...	2011...	1	RUNNING
🟩	4880...	alberto	normal	43200	2011...	2011...	18	RUNNING
🟨	4884...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4885...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟥	4886...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4887...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4888...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4889...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4890...	cheat...	normal	43200	2011...	2011...	16	RUNNING
🟩	4891...	cheat...	normal	43200	2011...	2011...	16	RUNNING

Inactive Jobs

	step	owner	queue	wall	queued:	dispatch	totalcor	status

Copy the Existing Configuration

- Select the existing configuration
- Click on the **copy** button to create a duplicate configuration



Debug C

Create, manage, and run configurations

[Debugger]: Debugger host must be specified

type filter text

- C/C++ Application
  - resourceManagers
  - test Release
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- Fortran Local Application
- Java Application
- Parallel Application
  - shallow
  - shallow (1)

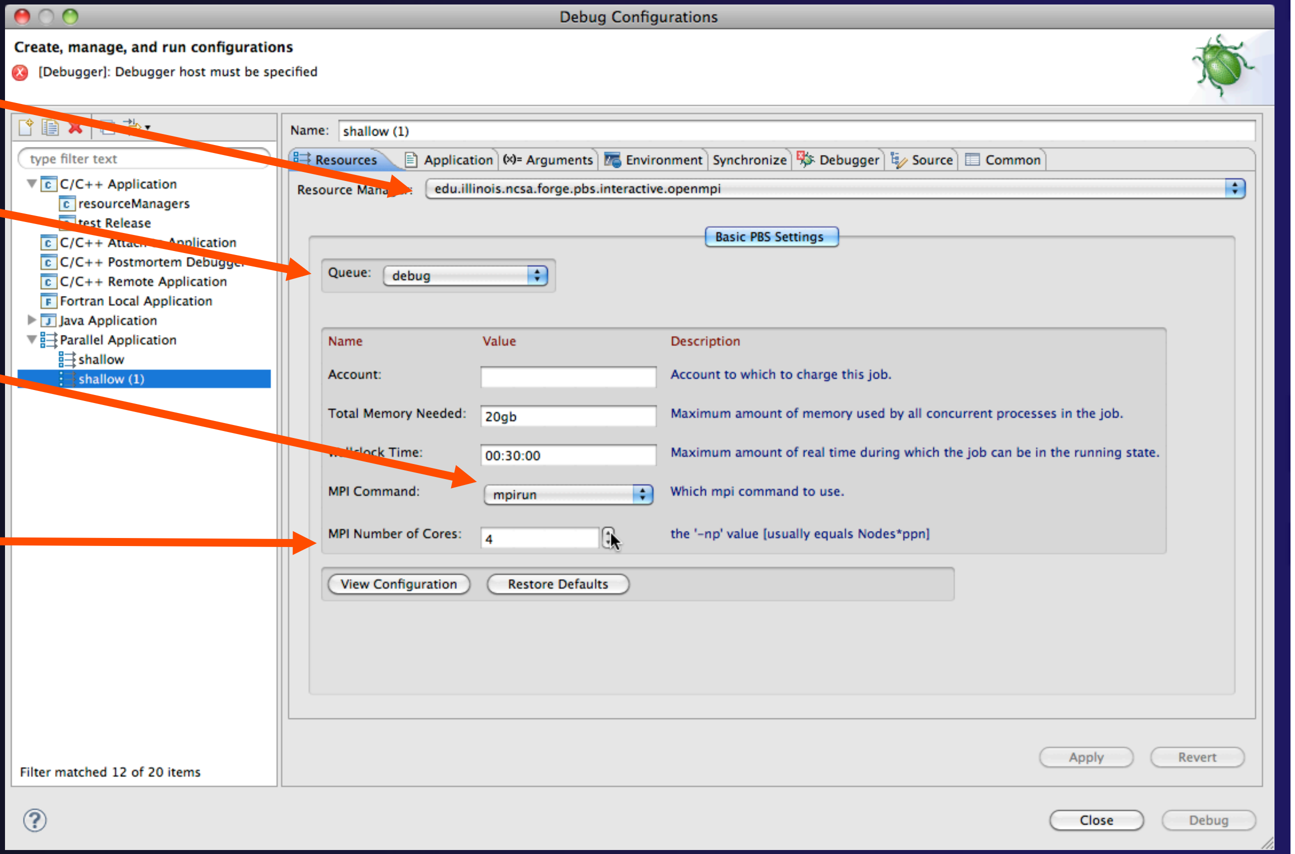
Name: shallow (1)

Resource Manager: edu.illinois.ncsa.forge.pbs.batc

Name	Value
Job Name:	ptp_job
Account:	
Queue:	
Number of nodes:	1
Total Memory Needed:	
Wallclock Time:	00:30:00
MPI Command:	mpirun
MPI Number of Cores:	4

Configure the Resource Tab

- Select the new resource manager
- Choose the **debug** queue
- Choose the **mpirun** command
- Select the number of cores (in this case use 4)



Debug Configurations

Create, manage, and run configurations

[Debugger]: Debugger host must be specified

Name: shallow (1)

Resource Manager: edu.illinois.ncsa.forge.pbs.interactive.openmpi

Basic PBS Settings

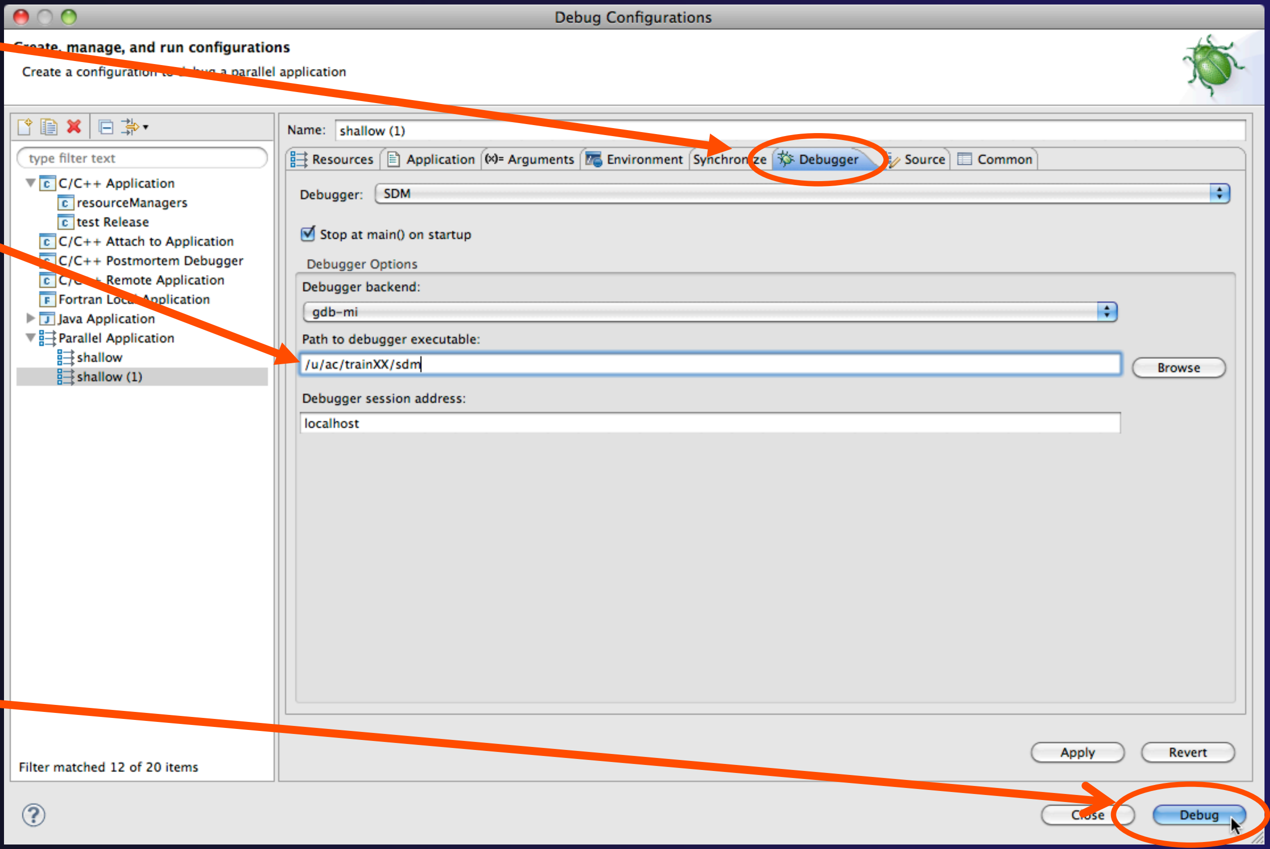
Queue: debug

Name	Value	Description
Account:		Account to which to charge this job.
Total Memory Needed:	20gb	Maximum amount of memory used by all concurrent processes in the job.
Wallclock Time:	00:30:00	Maximum amount of real time during which the job can be in the running state.
MPI Command:	mpirun	Which mpi command to use.
MPI Number of Cores:	4	the '-np' value [usually equals Nodes*ppn]

Filter matched 12 of 20 items

Configure the Debug Tab

- Select **Debugger** tab
- Set the debugger path to the **sdm** in your home directory
- 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 debug a parallel application

Name: shallow (1)

Debugger: SDM

☑ Stop at main() on startup

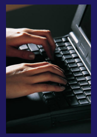
Debugger Options

Debugger backend: gdb-mi

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

Debugger session address: localhost

Filter matched 12 of 20 items

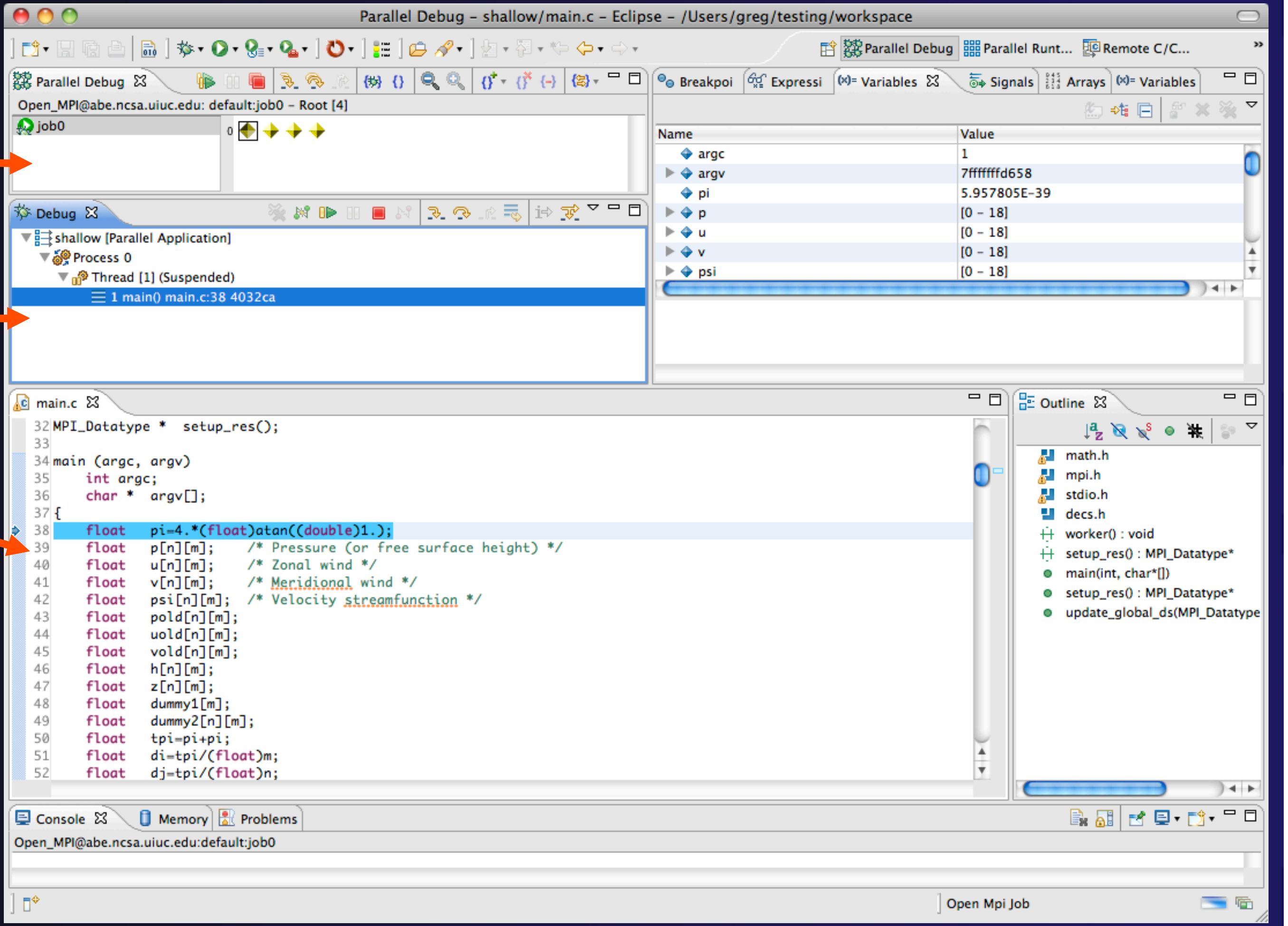


Exercise

- Open the debug configuration dialog
- Copy an existing configuration
  - Or create a new configuration (if you don't already have one)
- Select an *–Interactive* target configuration
- Configure the **Debug** tab
- Launch the debugger

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



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

job0 0

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

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

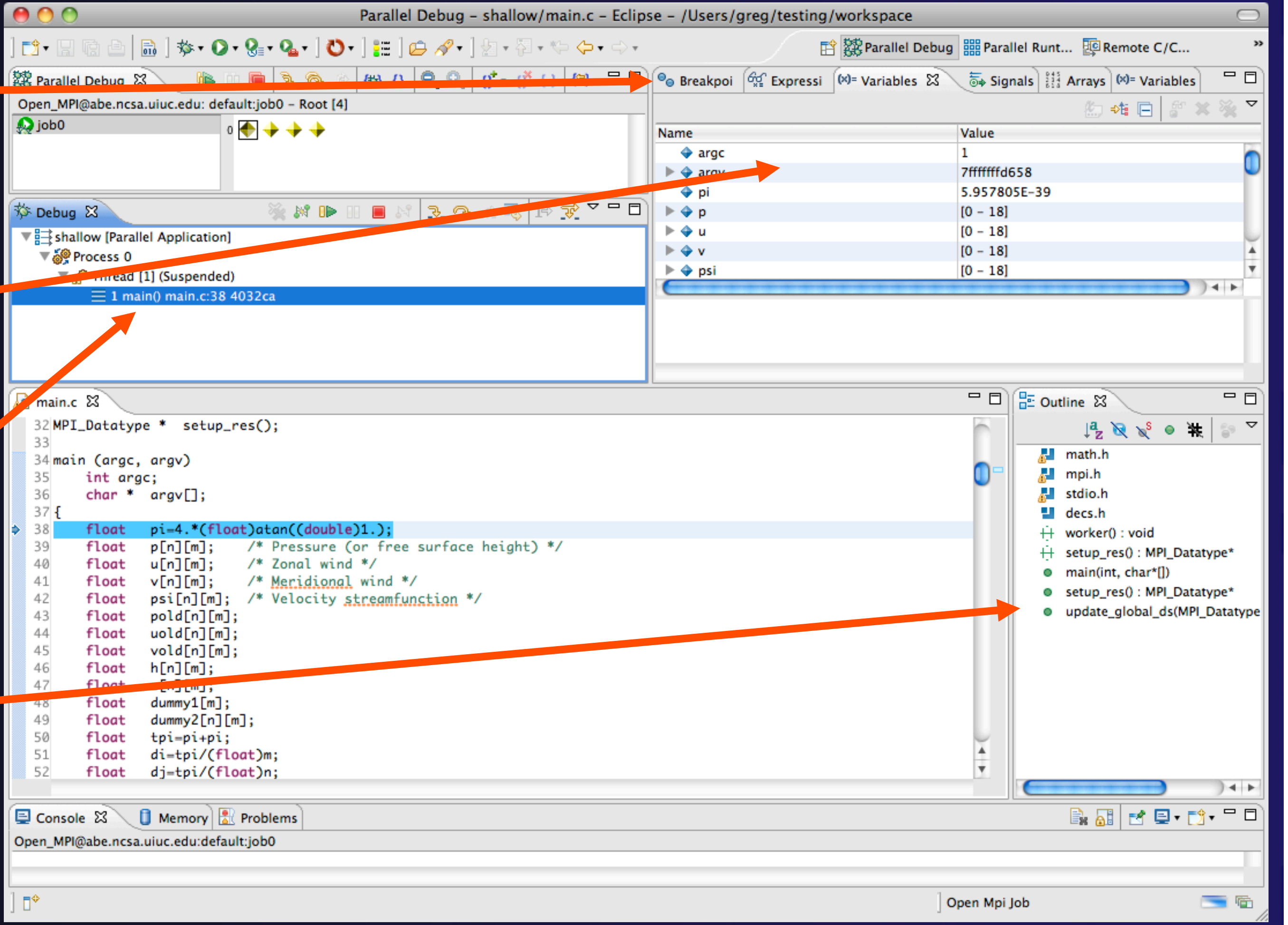
```
MPI_Datatype *  setup_res();

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
```

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

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]

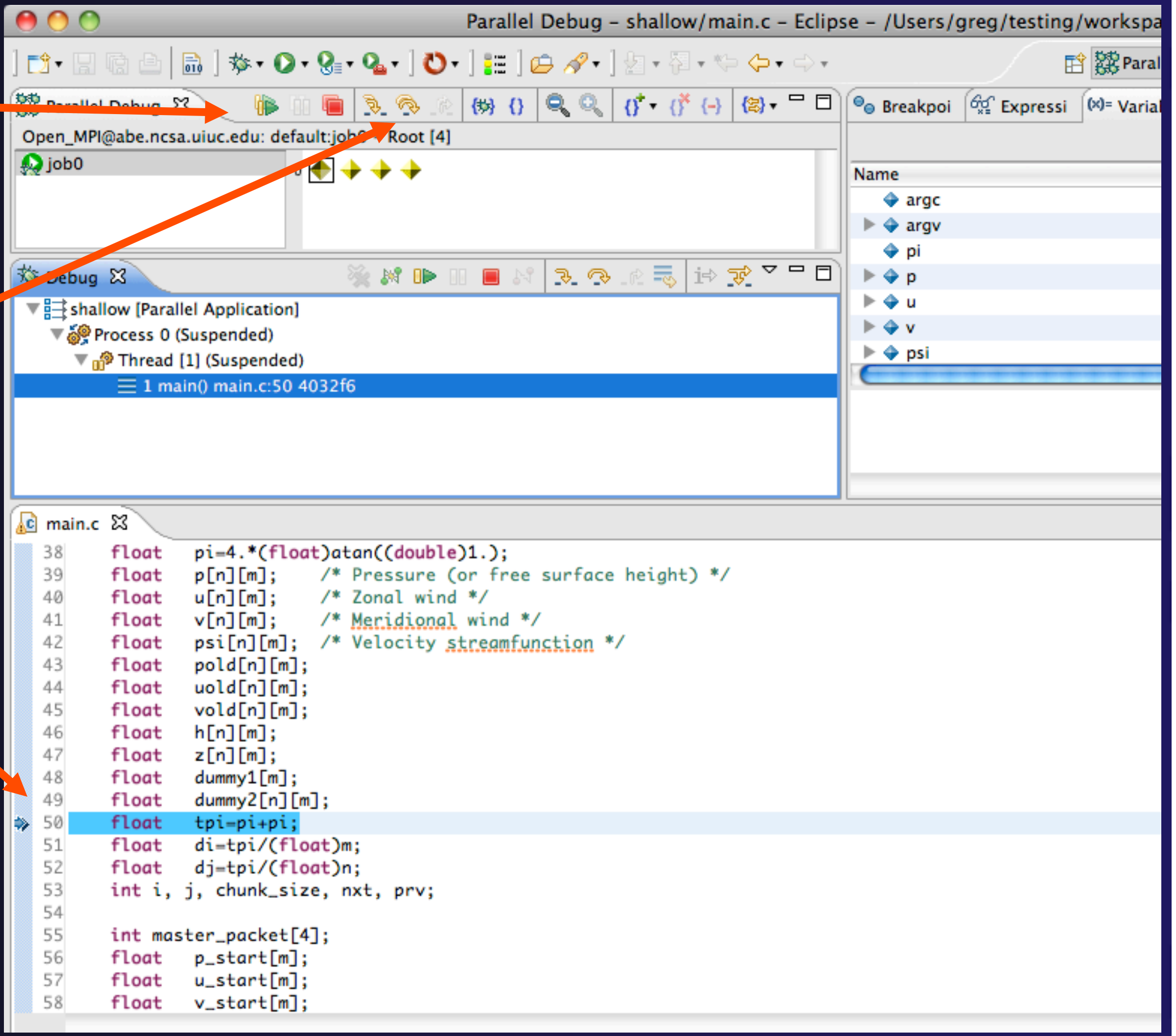
```
MPI_Datatype *   setup_res();

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   [illegible][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

Stepping All Processes

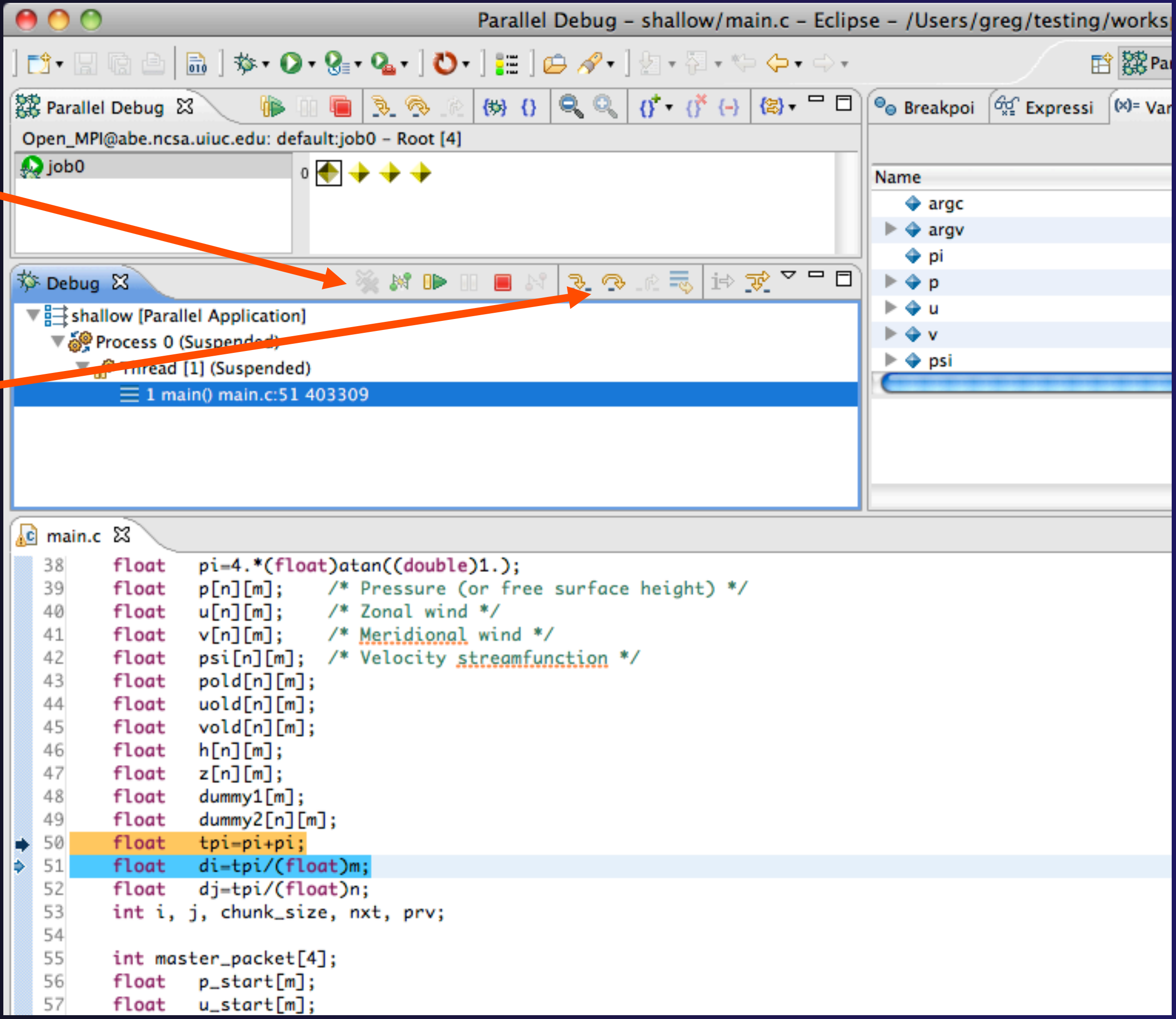
- The buttons in the **Parallel Debug View** control groups of processes
- Click on the **Step Over** button
- Observe that all process icons change to green, then back to yellow
- Notice that the current line marker has moved to the next source line


Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspa
Paral
Parallel Debug
Breakpoi
Expressi
Varial
Open_MPI@abe.ncsa.uiuc.edu: default:job0
Root [4]
job0
Name
argc
argv
pi
p
u
v
psi
Debug
shallow [Parallel Application]
Process 0 (Suspended)
Thread [1] (Suspended)
1 main() main.c:50 4032f6
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

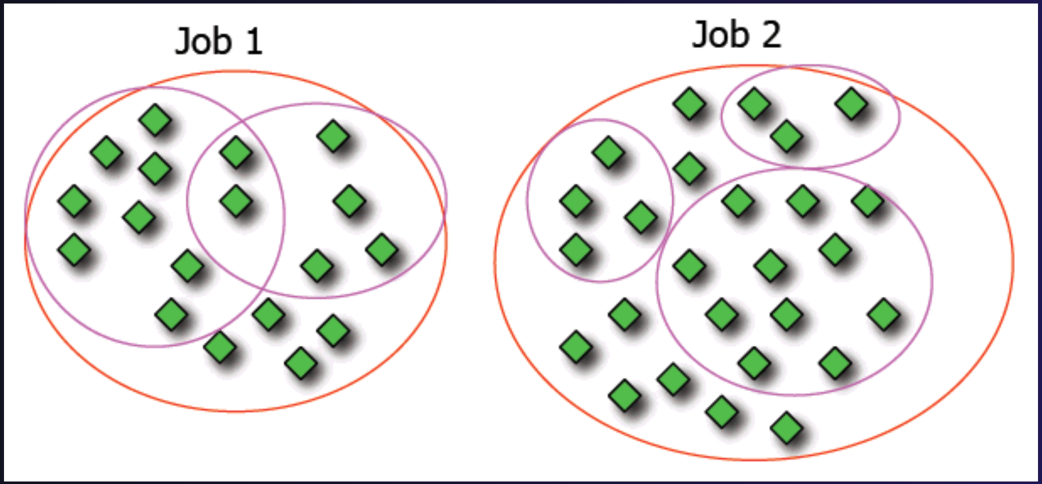
- The buttons in the **Debug view** are used to control an individual process, in this case process 0
- Click the **Step Over** button
- You will now see two current line markers, the first shows the position of process 0, the second shows the positions of processes 1-3


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

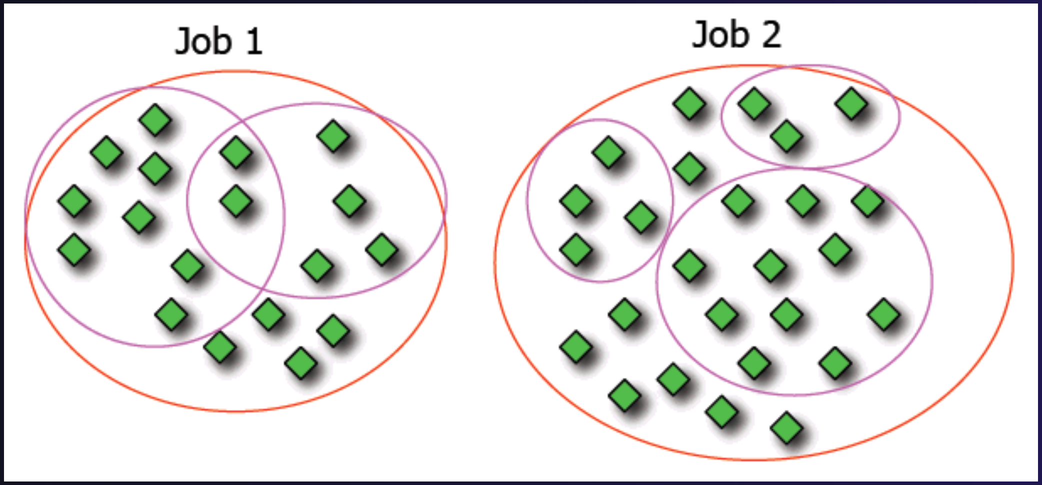
Process Sets (1)

- 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



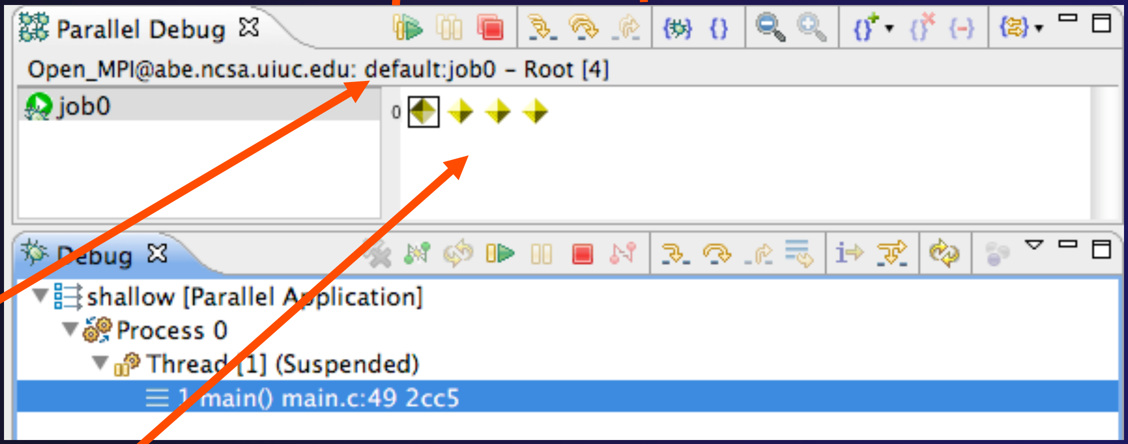
Process Sets (2)

- 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



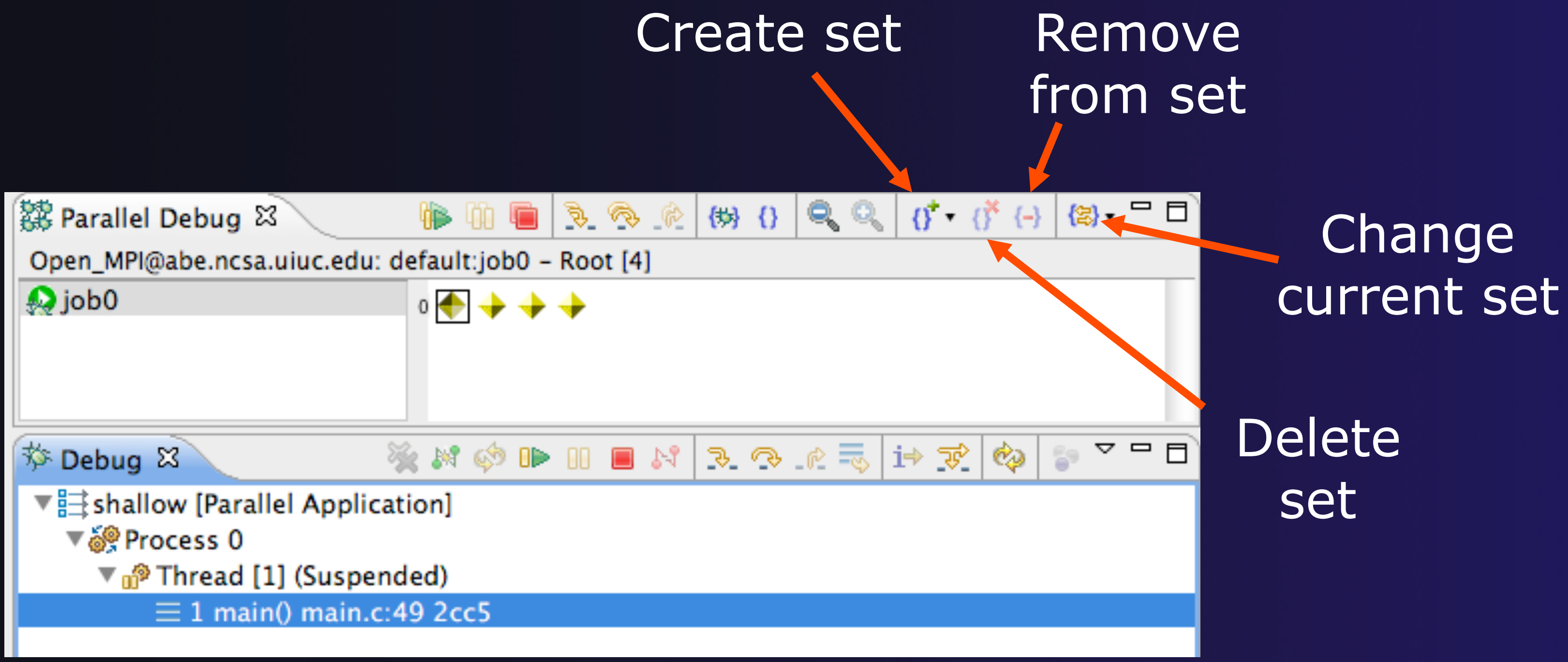
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

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



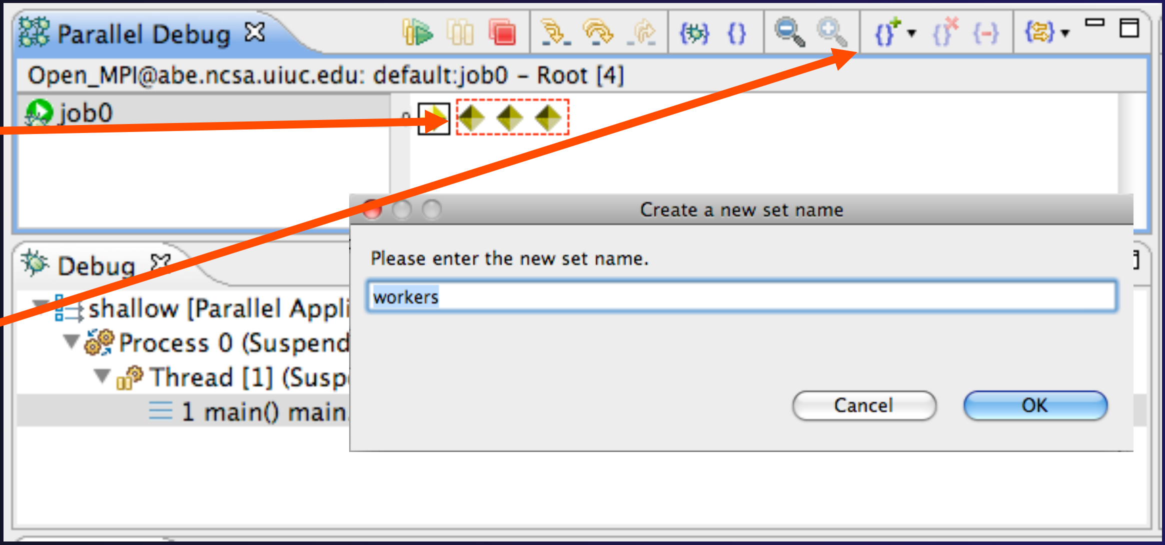
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – Root [4]

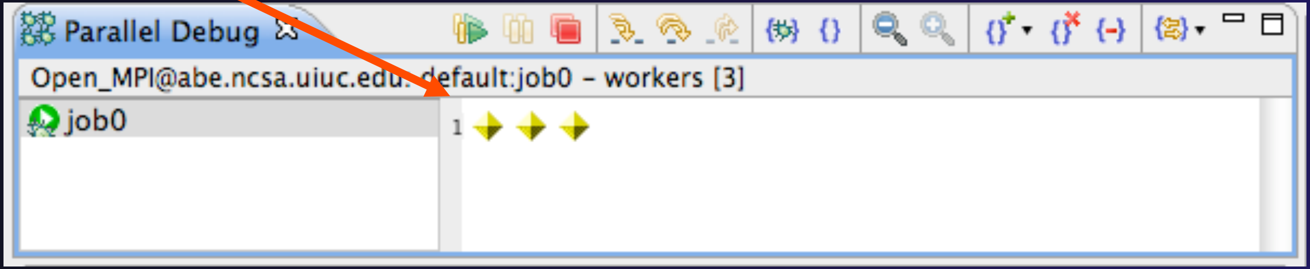
job0

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

Creating A New Process Set

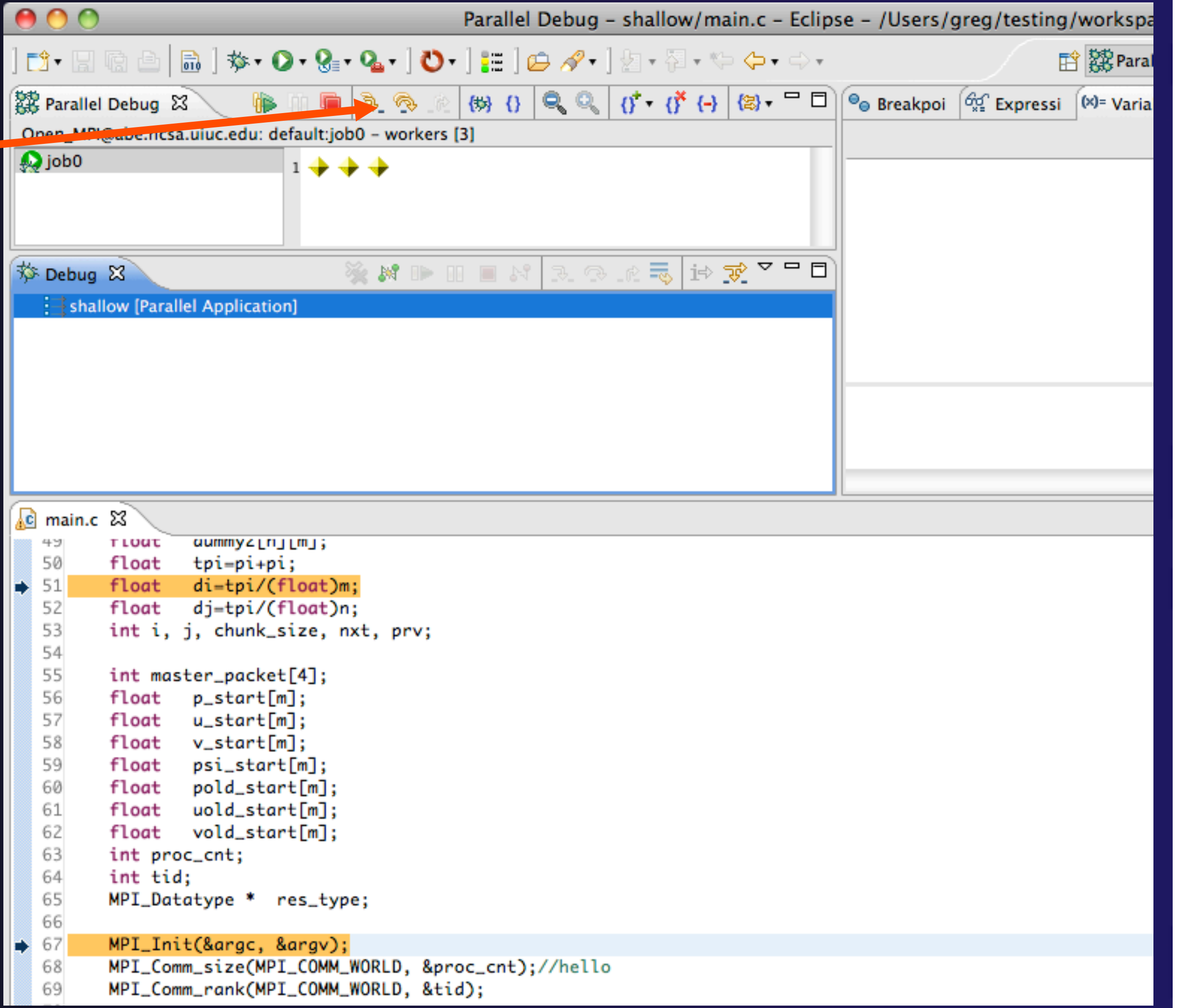
- Select the processes you want in the set by clicking and dragging, in this case, the last three
- Click on the **Create Set** button
- Enter a name for the set, in this case **workers**, and click **OK**
- You will see the view change to display only the selected processes





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
Breakpoi
Expressi
Varia
Debug
shallow [Parallel Application]
main.c

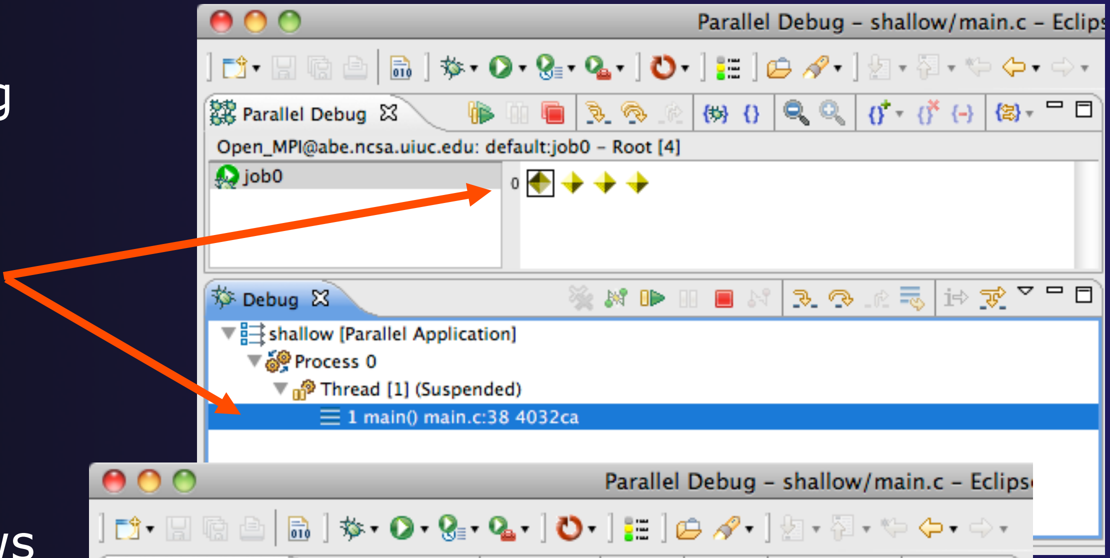
```
    float   tpi=pi+pi;
    float   di=tpi/(float)m;
    float   dj=tpi/(float)n;
    int i, j, chunk_size, nxt, prv;

    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

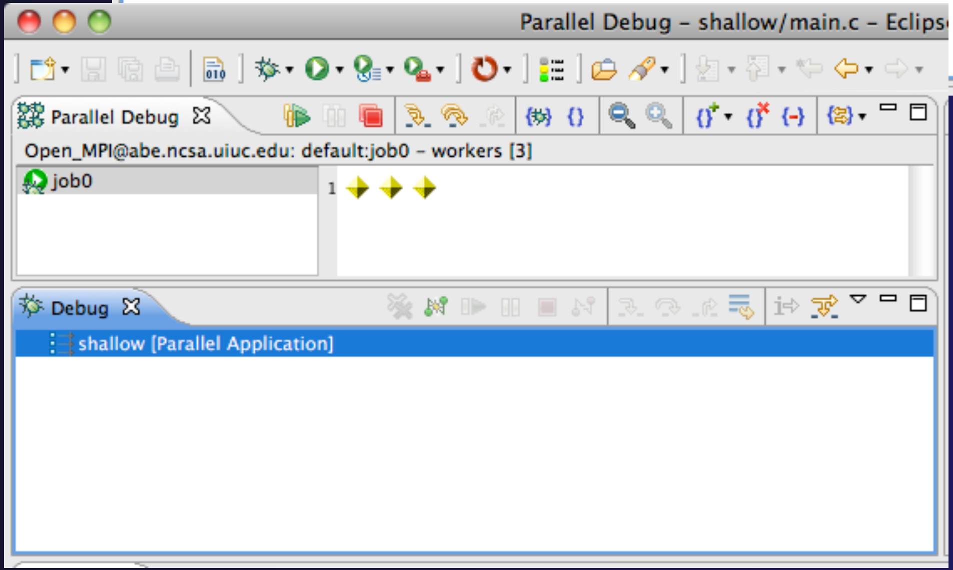


Parallel Debug – shallow/main.c – Eclips

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



Parallel Debug – shallow/main.c – Eclips

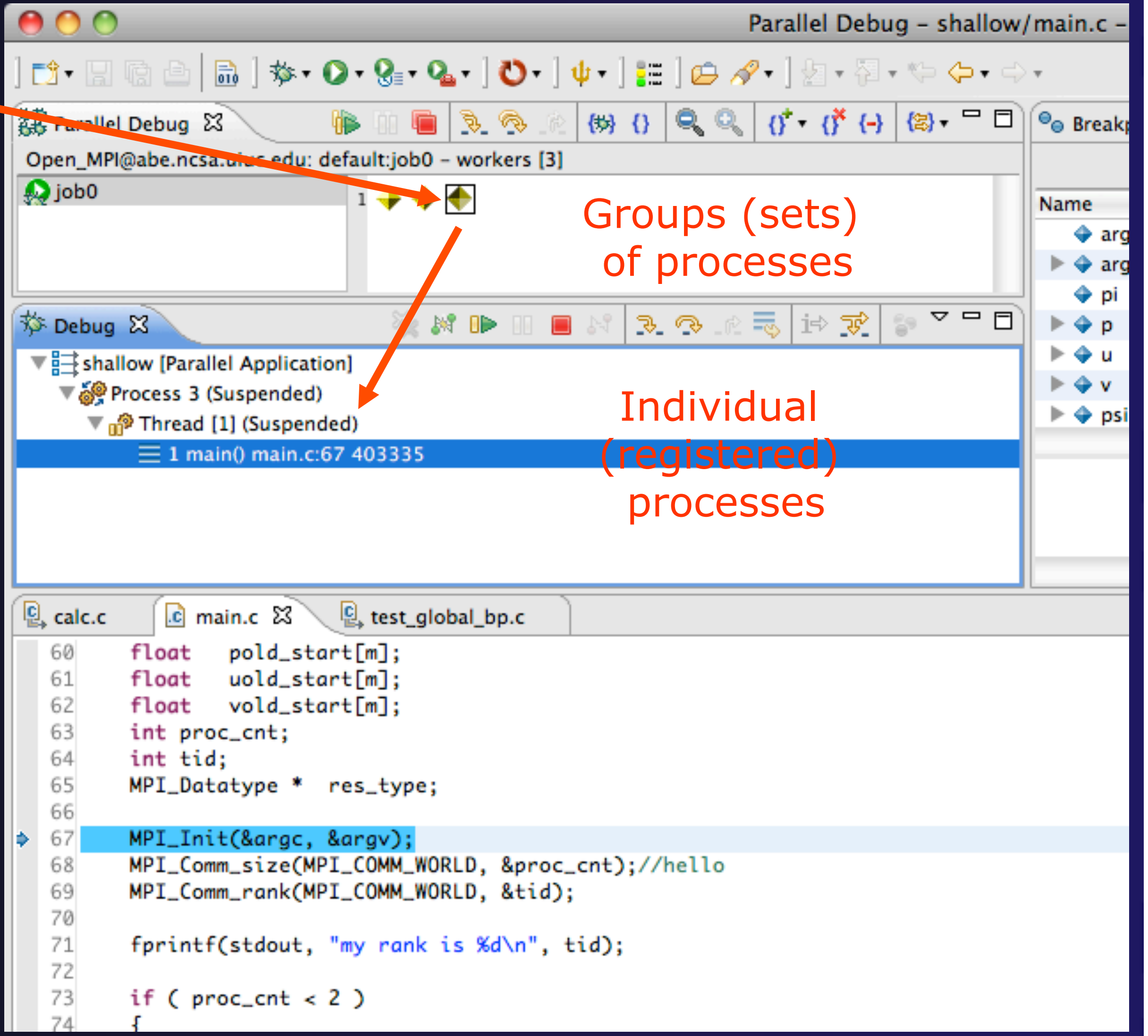
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]

job0

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
Groups (sets) of processes
Debug
shallow [Parallel Application]
Process 3 (Suspended)
Thread [1] (Suspended)
1 main() main.c:67 403335
Individual (registered) processes
Breakp
Name
arg
arg
pi
p
u
v
psi
calc.c
main.c
test_global_bp.c

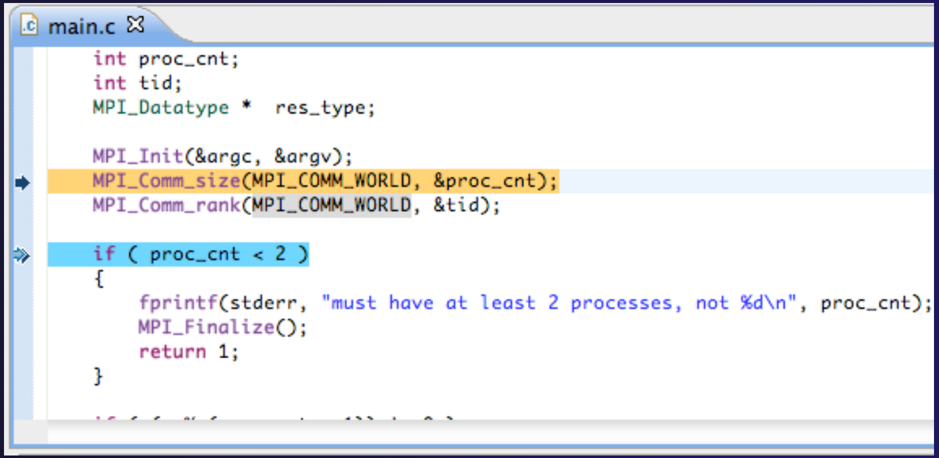
```
    float   pold_start[m];
    float   uold_start[m];
    float   vold_start[m];
    int proc_cnt;
    int tid;
    MPI_Datatype *  res_type;

    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



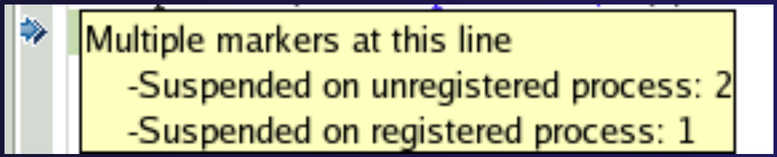
main.c

```
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

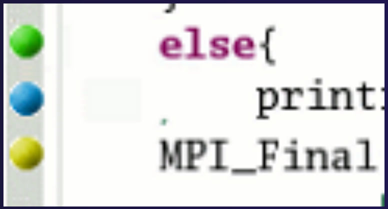
Un-registered process marker



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

Breakpoints

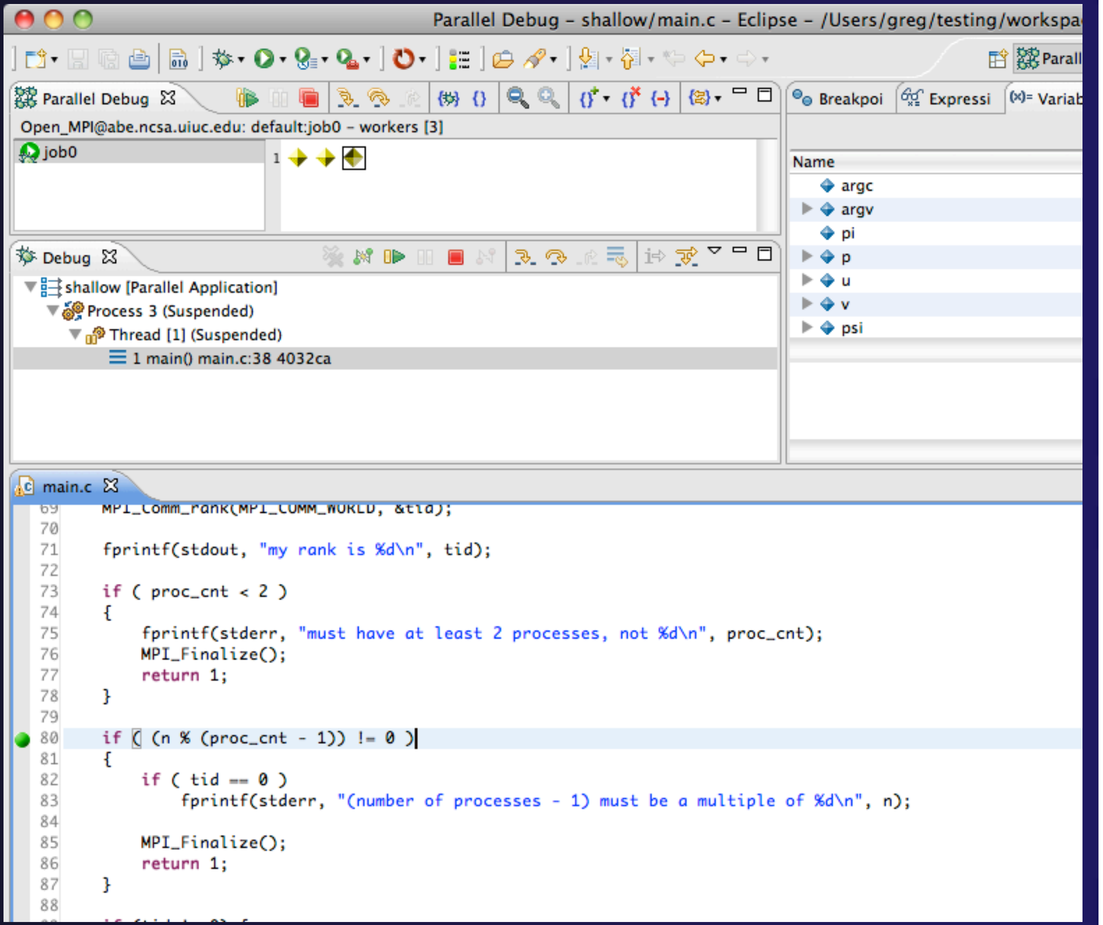
- Apply only to processes in the particular set that is active in the **Parallel Debug view** when the breakpoint is created
- Breakpoints are colored depending on the active process set and the set the breakpoint applies to:
  - Green indicates the breakpoint set is the same as the active set.
  - Blue 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

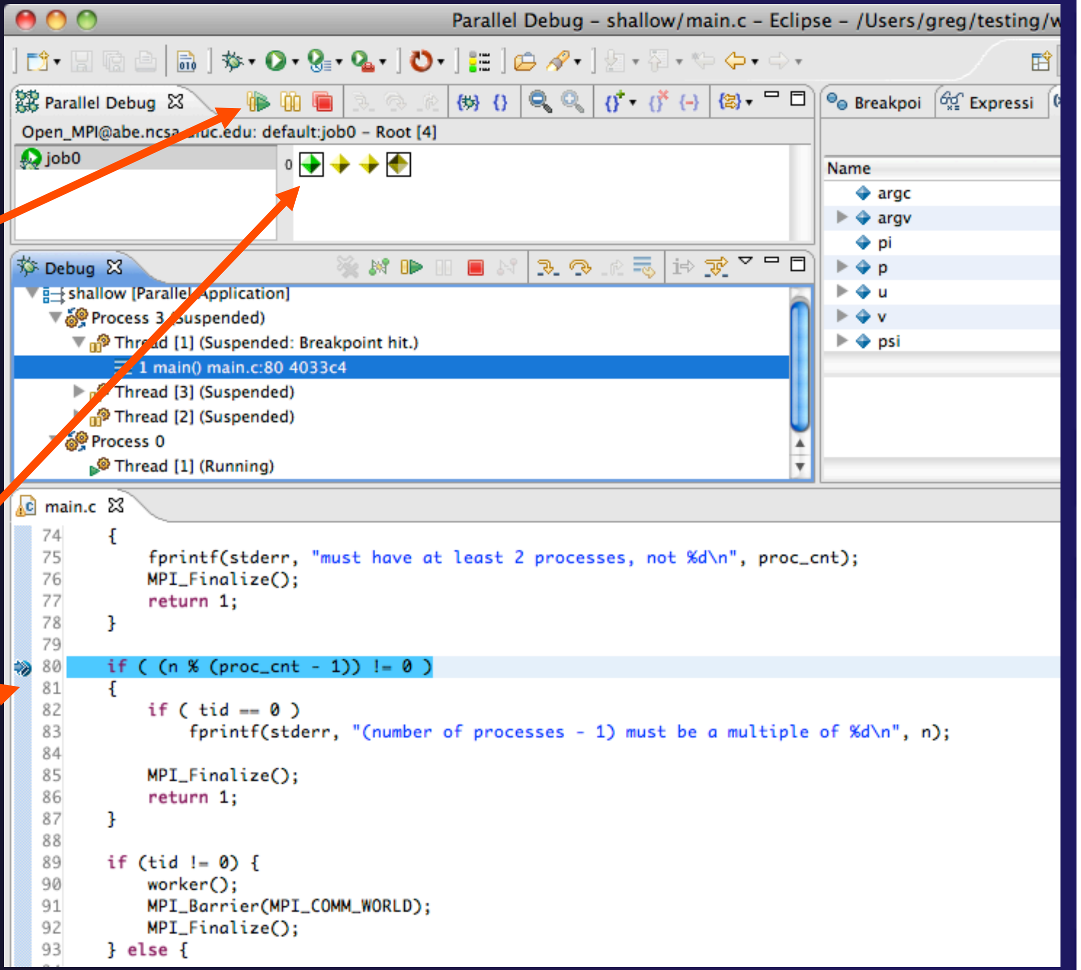
- Select the process set that the breakpoint should apply to, in this case, the **workers** set
- Double-click on the left edge of an editor window, at the line on which you want to set the breakpoint, or right click and use the **Parallel Breakpoint▶Toggle Breakpoint** context menu
- The breakpoint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


Parallel Debug – shallow/main.c – Eclipse – /Users/greg/testing/workspa
Parall
Parallel Debug
Breakpoi
Expressi
Variab
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]
job0
1
Name
argc
argv
pi
p
u
v
psi
Debug
shallow [Parallel Application]
Process 3 (Suspended)
Thread [1] (Suspended)
1 main() main.c:38 4032ca
main.c

```
    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
Parallel Debug
Breakpoi
Expressi
Debug
main.c

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

job0

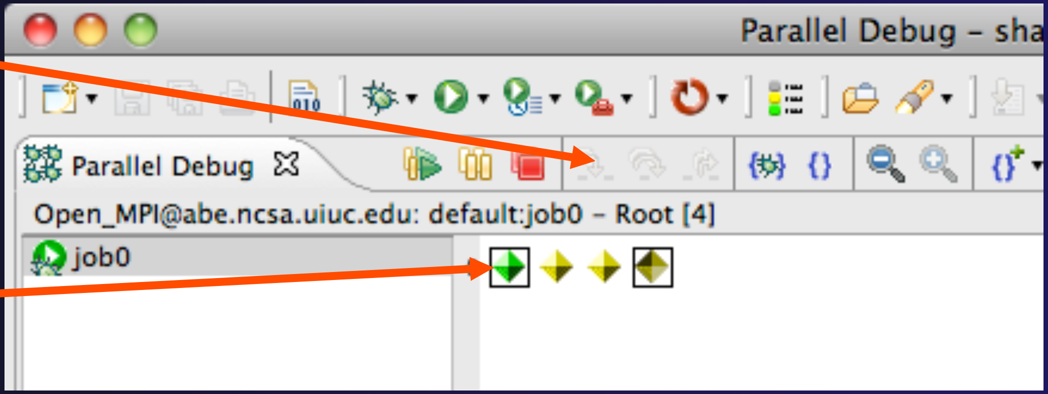
Name
argc
argv
pi
p
u
v
psi

```
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



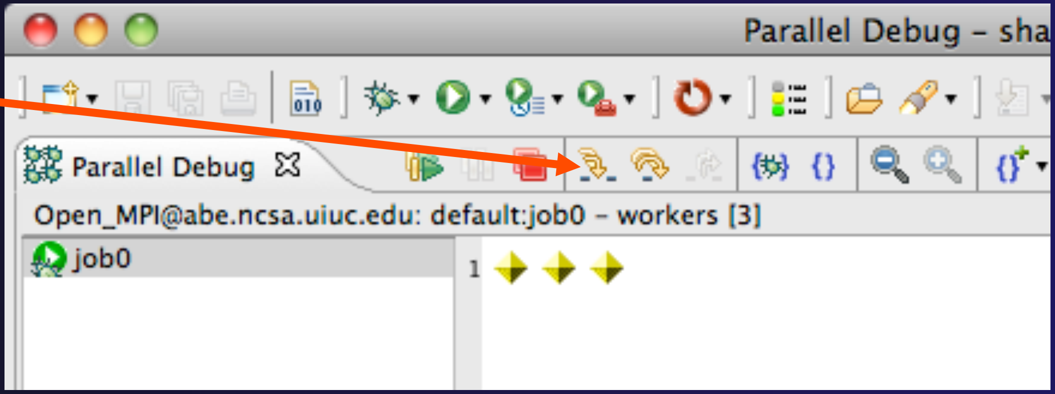
Parallel Debug – sha

Parallel Debug

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

job0

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



Parallel Debug – sha

Parallel Debug

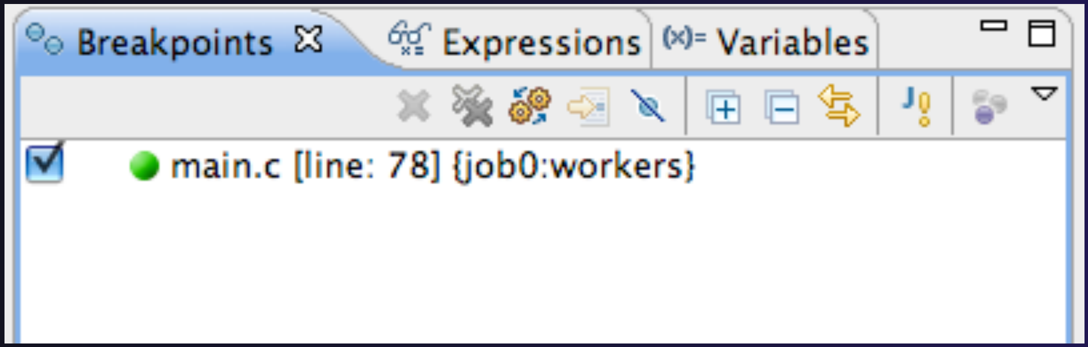
Open_MPI@abe.ncsa.uiuc.edu: default:job0 – workers [3]

job0

1

Breakpoint Information

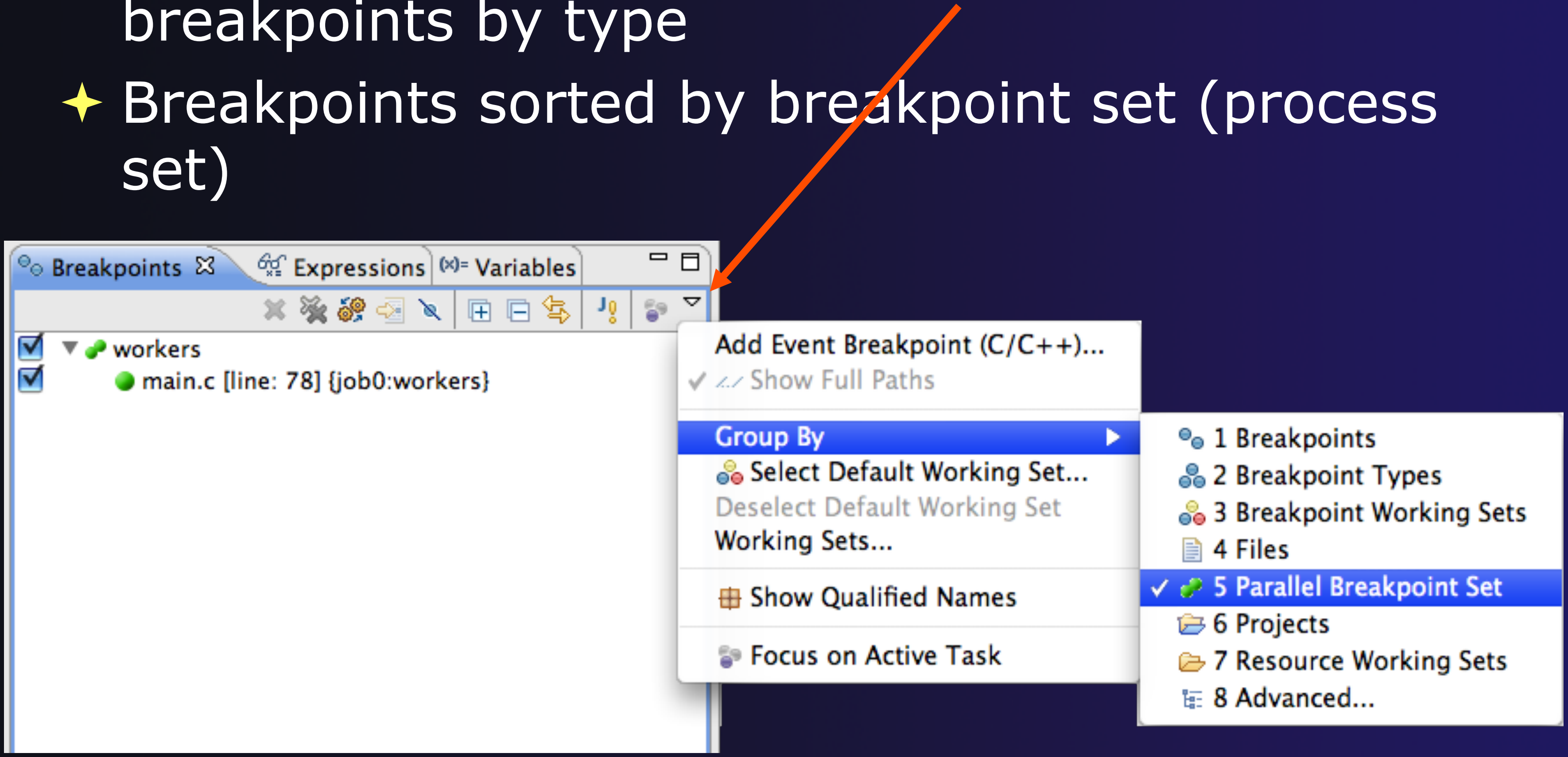
- Hover over breakpoint icon
  - Will show the sets this breakpoint applies to
- Select **Breakpoints** view
  - Will show all breakpoints in all projects



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

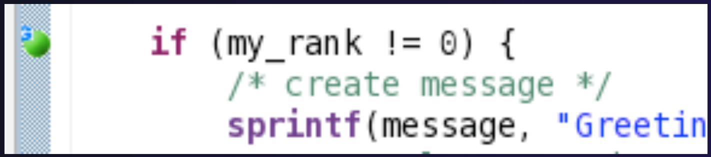
Breakpoints View

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



Global Breakpoints

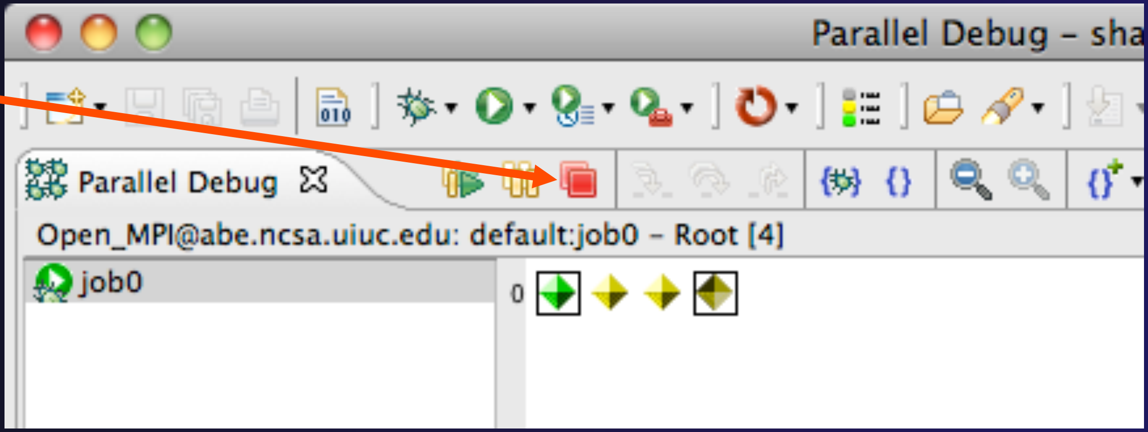
- Apply to all processes and all jobs
- Used for gaining control at debugger startup
- To create a global breakpoint
  - First make sure that no jobs are selected (click in white part of jobs view if necessary)
  - Double-click on the left edge of an editor window
  - Note that if a job is selected,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



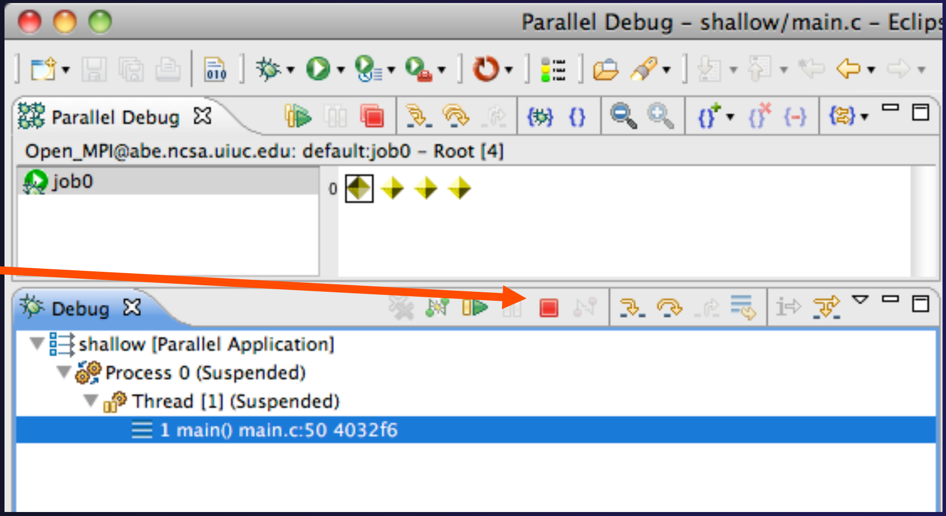
```
    if (my_rank != 0) {
        /* create message */
        sprintf(message, "Greetin
```

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


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

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

Performance Tuning and Analysis Tools

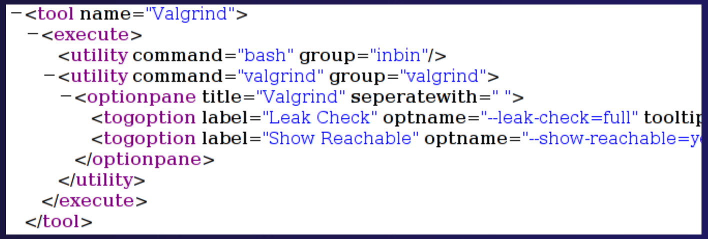
- Objective
  - Become familiar with tools integrated with PTP, to help enhance performance of parallel applications
- Contents
  - Overview of ETFw and Performance Tools
  - Maybe one slide on each?
  - More detail in separate modules

PTP/External Tools Framework

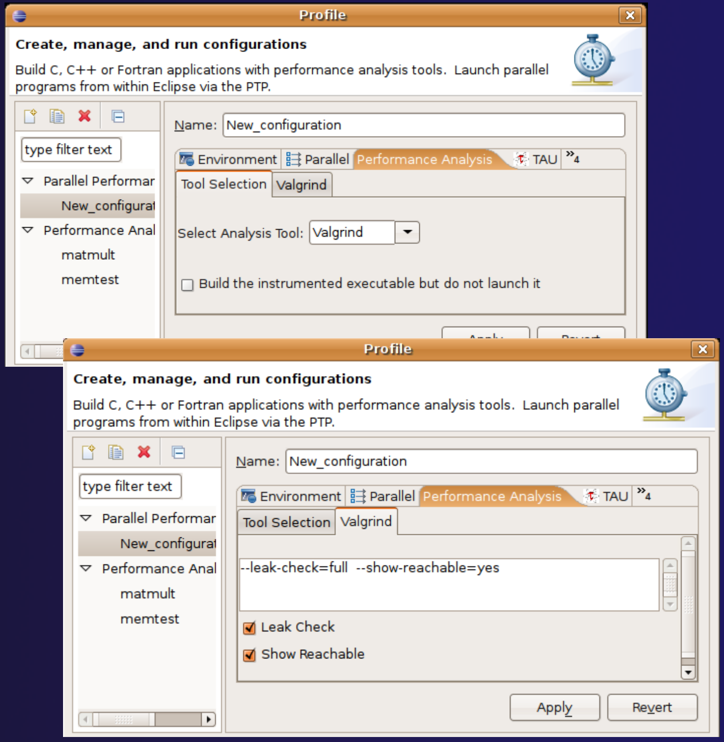
formerly “Performance Tools Framework”

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
    - One-click ‘launch’ functionality
    - Support for development tools such as TAU, PPW and others.
    - Adding new tools is much easier than developing a full Eclipse plug-in



```
−<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" toolti
          <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

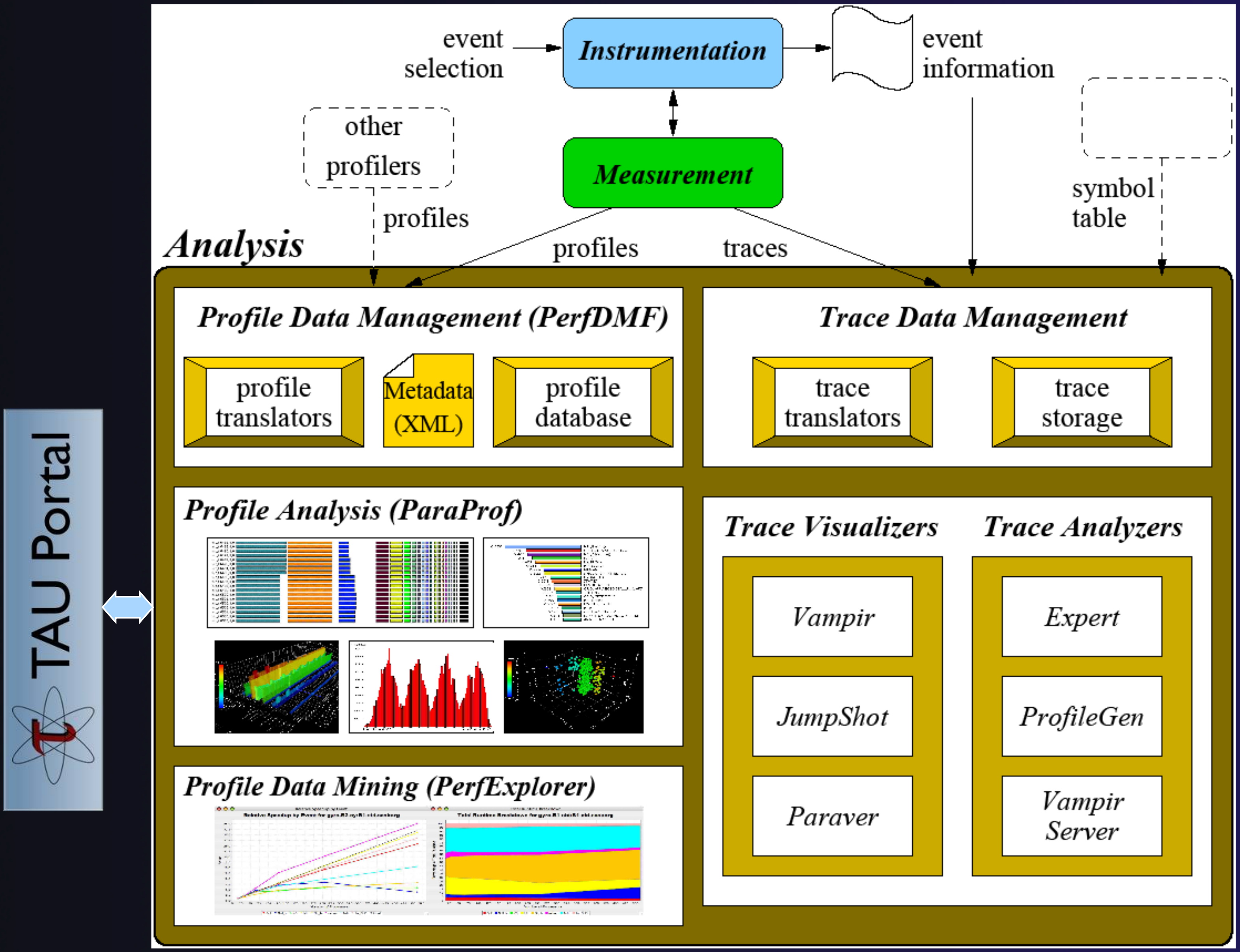
Performance Tuning and Analysis Tools - TAU

- Objective
  - Become familiar with tools integrated with PTP, to help enhance performance of parallel applications
- Contents
  - Performance Tuning and external tools:
    - PTP External Tools Framework (ETFw), TAU
    - Hands-on exercise using TAU with PTP

TAU: Tuning and Analysis Utilities

- TAU is a performance evaluation tool
- It supports parallel profiling and tracing
- Profiling shows you how much (total) time was spent in each routine
- Tracing shows you *when* the events take place in each process along a timeline
- TAU uses a package called PDT for automatic instrumentation of the source code
- Profiling and tracing can measure time as well as hardware performance counters from your CPU (or GPU!)
- TAU can automatically instrument your source code (routines, loops, I/O, memory, phases, etc.)
- TAU runs on all HPC platforms and it is free (BSD style license)
- TAU has instrumentation, measurement and analysis tools
  - paraprof is TAU’s 3D profile browser

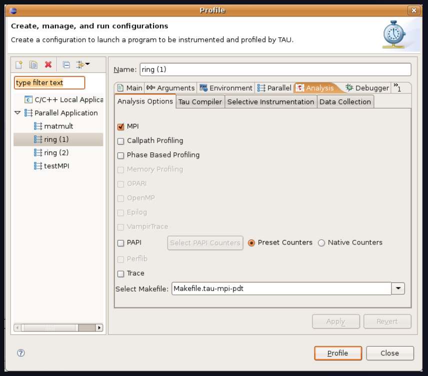
TAU Performance System Architecture



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
- Full GUI support for the TAU command line interface
- Performance analysis integrated with development environment



Profile

Create, manage, and run configurations

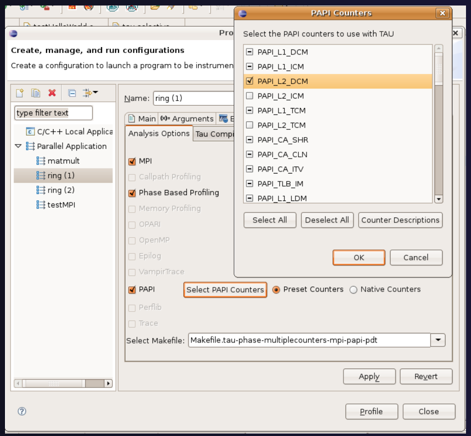
Create a configuration to launch a program to be instrumented and profiled by TAU.

Name: ring (1)

Analysis Options:

- [✓] MPI
- [ ] Callpath Profiling
- [ ] Phase Based Profiling
- [ ] PAPI — Preset Counters [✓] / Native Counters
- [ ] Trace

Select Makefile: Makefile.tau-mpi-pdt



PAPI Counters

Select the PAPI counters to use with TAU

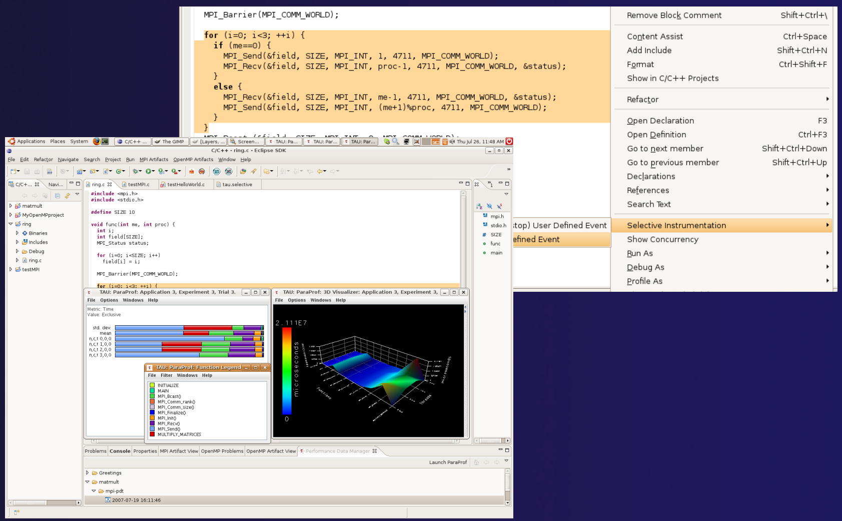
- PAPI_L1_DCM
- PAPI_L1_ICM
- [✓] PAPI_L2_DCM
- PAPI_L2_ICM
- PAPI_L1_TCM
- PAPI_L2_TCM
- PAPI_CA_SHR
- PAPI_CA_CLN
- PAPI_CA_ITV
- PAPI_TLB_IM
- PAPI_L1_LDM

Name: ring (1)

Analysis Options:

- [✓] MPI
- [✓] Phase Based Profiling
- [✓] PAPI — Preset Counters [✓] / Native Counters

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

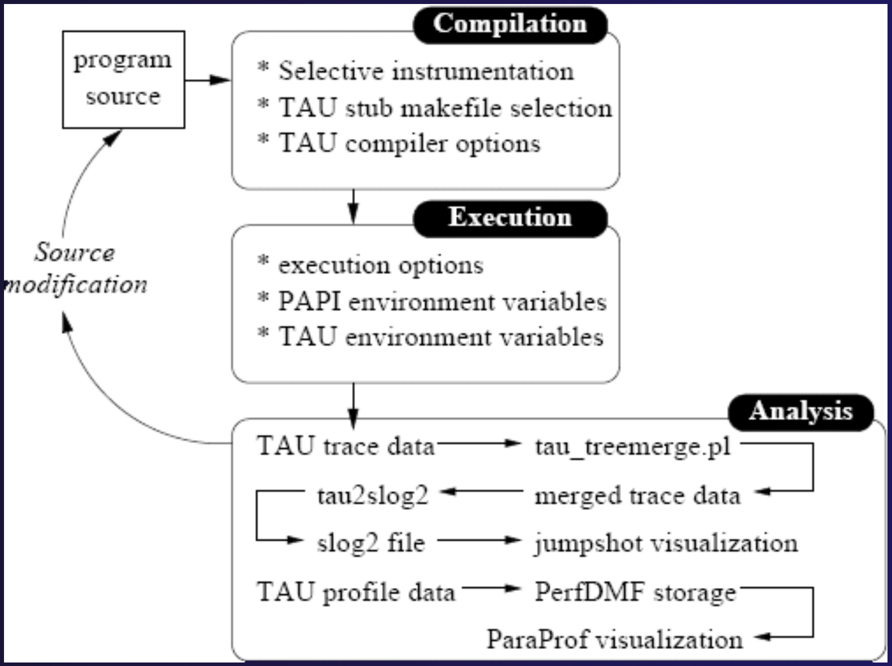


```
MPI_Barrier(MPI_COMM_WORLD);

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

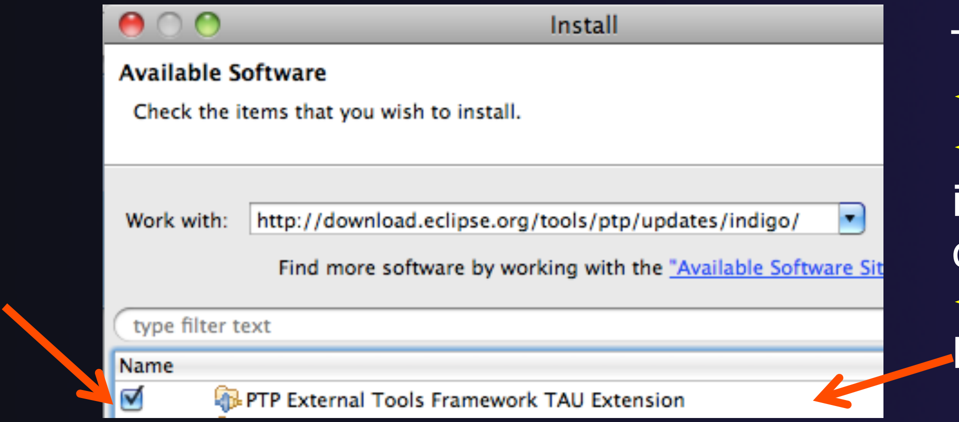
TAU Integration with PTP

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



TAU PTP Installation

- This tutorial assumes that the TAU extensions for PTP are installed – they are not included in the "Eclipse IDE for Parallel Application Developers"
- The installation section (Module 1) shows how to install TAU and other features from the PTP update site – be sure TAU was selected


Install

Available Software

Check the items that you wish to install.

Work with: http://download.eclipse.org/tools/ptp/updates/indigo/

Find more software by working with the "Available Software Sit

type filter text

Name

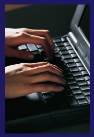
☑ PTP External Tools Framework TAU Extension

To confirm:
- Help>Install New Software…
- Select the link "What is already installed" at the bottom of the dialog
- You should see the TAU Extension

TAU/ETFw Hands-On(0) Assumptions

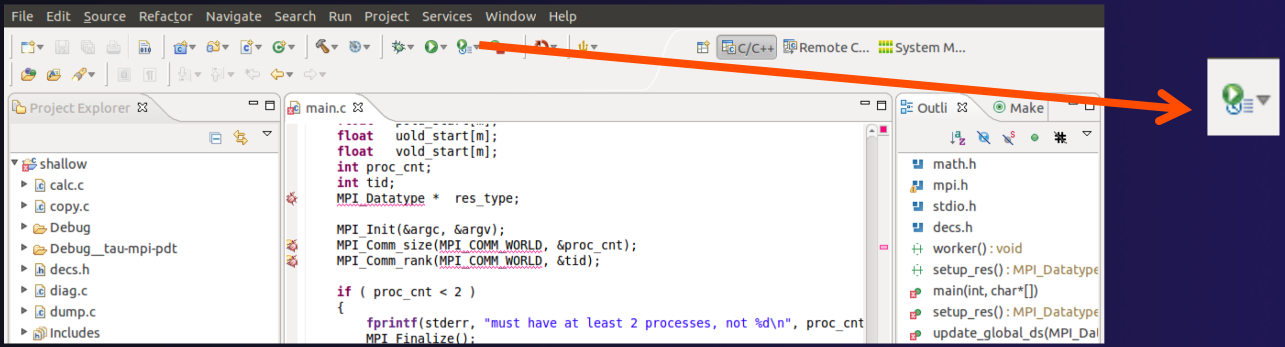
- Obtain and install TAU*
  - Download at tau.uoregon.edu
  - The website includes setup and user guides
- Set up the $PATH on the remote machine*
  - For TAU you should be able to run ‘which pprof’ on a remote login and see a result from your TAU bin directory
  - On forge.ncsa.illinois.edu this is accomplished by placing ‘module load tau’ in the .modules file in the home directory
- Include ‘eclipse.inc’ in the makefile*
  - Create an empty eclipse.inc file in the same directory as the makefile
  - Place ‘include eclipse.inc’ in the makefile after regular compiler definitions
  - ETFw will modify eclipse.inc to set CC/CXX/FC variables

* SC tutorial: this has been done for you



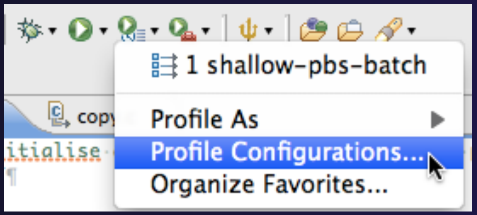
TAU/ETFw Hands-On(1) Begin Profile Configuration

- The ETFw uses the same run configurations and resource managers as debugging/launching
- Click on the 'Run' menu or the right side of the Profile button

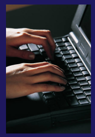


```
float   uold_start[m];
float   vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype *  res_type;

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

- From the dropdown menu select 'Profile configurations…'

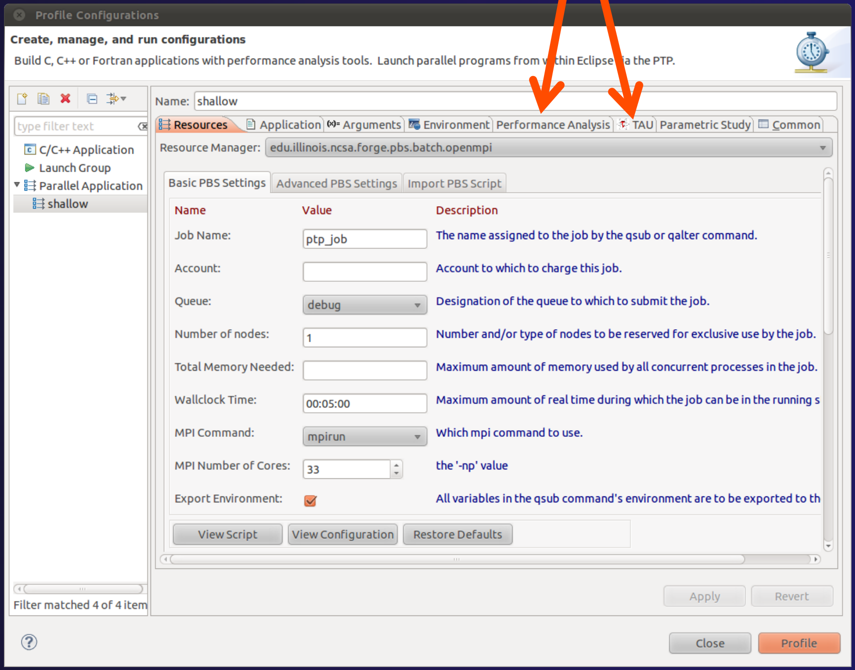


TAU/ETFw Hands-On(2) Select Configuration



- Select the shallow configuration prepared earlier
- The Resource and Application configuration tabs require little or no modification
  - We are using the same resource manager (edu.illinois.ncsa.forge.pbs.batch.openmpi) and PBS settings
  - Since we are using a makefile project the application will be rebuilt in and run from the previously selected location

Performance Analysis and TAU tabs are present in the **Profile Configurations** dialog



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

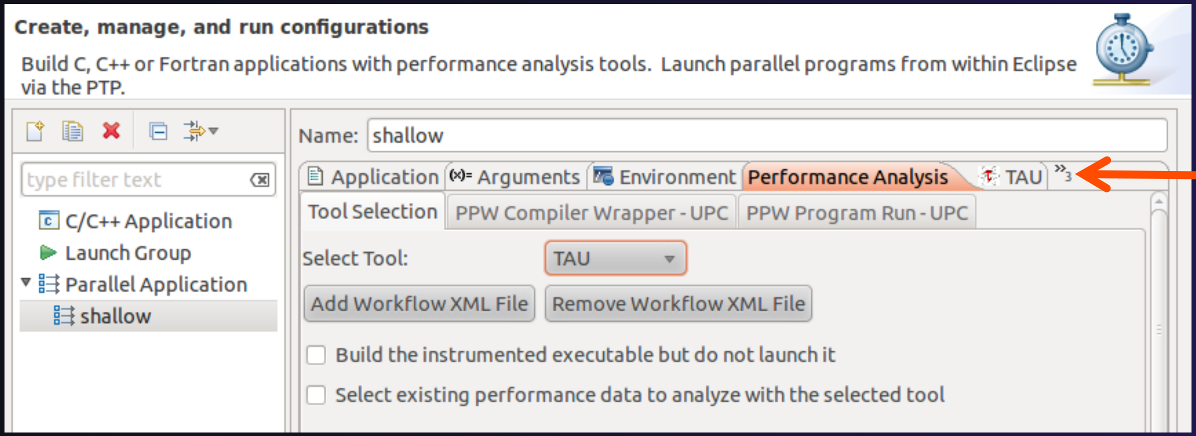
Name: shallow

Resource Manager: edu.illinois.ncsa.forge.pbs.batch.openmpi

Name	Value	Description
Job Name:	ptp_job	The name assigned to the job by the qsub or qalter command.
Account:		Account to which to charge this job.
Queue:	debug	Designation of the queue to which to submit the job.
Number of nodes:	1	Number and/or type of nodes to be reserved for exclusive use by the job.
Total Memory Needed:		Maximum amount of memory used by all concurrent processes in the job.
Wallclock Time:	00:05:00	Maximum amount of real time during which the job can be in the running s
MPI Command:	mpirun	Which mpi command to use.
MPI Number of Cores:	33	the '-np' value
Export Environment:	✓	All variables in the qsub command's environment are to be exported to th

TAU/ETFw Hands-On (3) Select Tool/Workflow

- Select the **Performance Analysis** tab and choose the TAU tool set in the 'Select Tool' dropdown box
  - Other tools may be available, either installed as plug-ins or loaded from workflow definition XML files



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: shallow

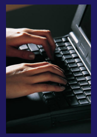
Select Tool: TAU

☐ Build the instrumented executable but do not launch it

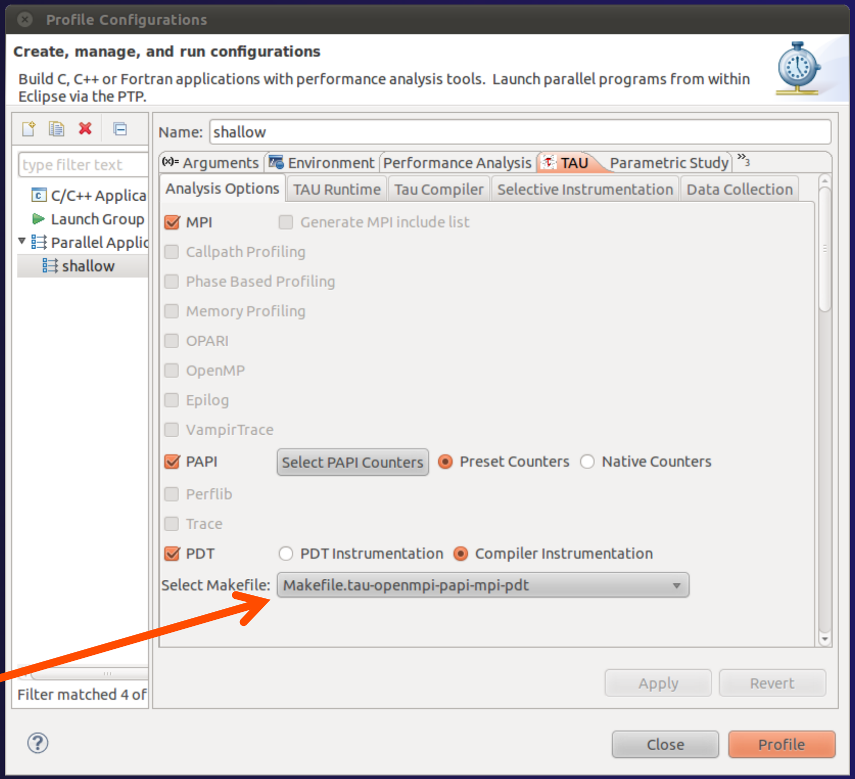
☐ Select existing performance data to analyze with the selected tool

Tabs may be hidden if the window is too small

TAU/ETFw Hands-On (4) Select TAU Configuration



- Select the **TAU** tab
- Choose the TAU stub makefile:
  - All TAU configurations in remote installation are available
  - Check MPI, PAPI and PDT checkboxes to filter listed makefiles
  - Make your selection in the **Select Makefile:** dropdown box
  - Select Makefile.tau-openmpi-papi-mpi-pdt



Profile Configurations

Create, manage, and run configurations

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

type filter text

- C/C++ Applica
- Launch Group
- Parallel Applic
  - shallow

Filter matched 4 of

Name: shallow

Tabs: Arguments | Environment | Performance Analysis | TAU | Parametric Study | »3

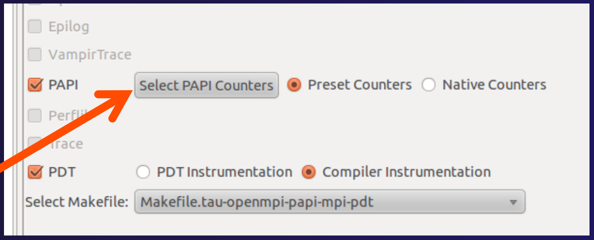
Analysis Options | TAU Runtime | Tau Compiler | Selective Instrumentation | Data Collection

- [✓] MPI — [ ] Generate MPI include list
- [ ] Callpath Profiling
- [ ] Phase Based Profiling
- [ ] Memory Profiling
- [ ] OPARI
- [ ] OpenMP
- [ ] Epilog
- [ ] VampirTrace
- [✓] PAPI — Select PAPI Counters — [✓ Preset Counters] Native Counters
- [ ] Perflib
- [ ] Trace
- [✓] PDT — PDT Instrumentation [✓ Compiler Instrumentation]

Select Makefile: Makefile.tau-openmpi-papi-mpi-pdt

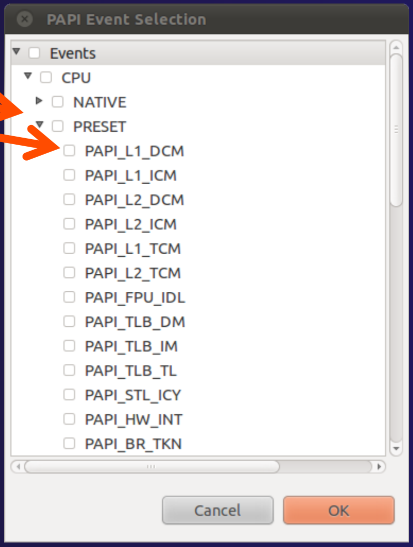
TAU/ETFw Hands-On (5) Choose PAPI Hardware Counters

- When a PAPI-enabled TAU configuration is selected the PAPI Counter tool becomes available
  - Select the 'Select PAPI Counters' button to open the tool
  - Open the PRESET subtree
  - Select PAPI_L1_DCM (Data cache misses)
  - Scroll down to select PAPI_FP_INS (Floating point instructions)
  - Invalid selections are automatically excluded
  - Select **OK**



- [ ] Epilog
- [ ] VampirTrace
- [x] PAPI — Select PAPI Counters — [● Preset Counters] ○ Native Counters
- [ ] Perfl...
- [ ] Trace
- [x] PDT — ○ PDT Instrumentation [● Compiler Instrumentation]

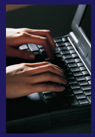
Select Makefile: Makefile.tau-openmpi-papi-mpi-pdt



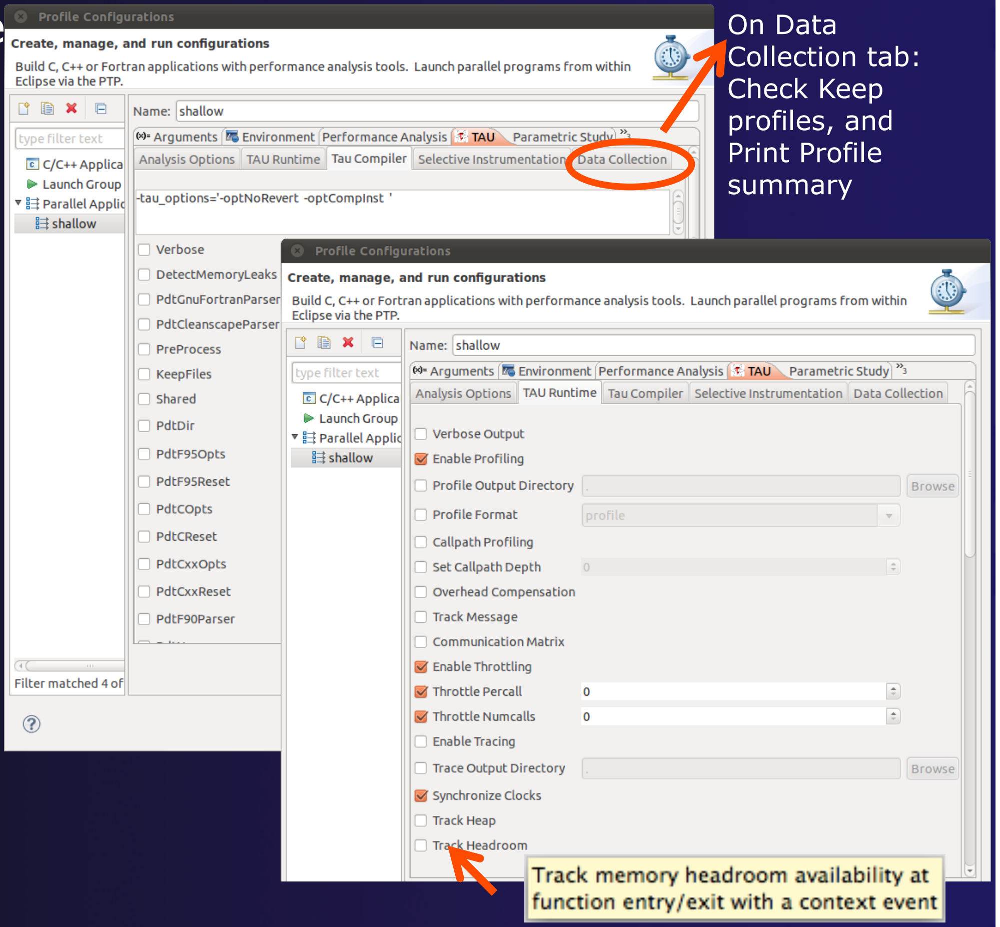
PAPI Event Selection

- Events
  - CPU
    - NATIVE
    - PRESET
      - PAPI_L1_DCM
      - PAPI_L1_ICM
      - PAPI_L2_DCM
      - PAPI_L2_ICM
      - PAPI_L1_TCM
      - PAPI_L2_TCM
      - PAPI_FPU_IDL
      - PAPI_TLB_DM
      - PAPI_TLB_IM
      - PAPI_TLB_TL
      - PAPI_STL_ICY
      - PAPI_HW_INT
      - PAPI_BR_TKN

TAU/ETFw Hands-On (6) Compiler and Runtime Options



- Other tab settings are described here but no changes are required…
- TAU Compiler Options
  - Set arguments to TAU compiler scripts
  - Control instrumentation and compilation behavior
- TAU Runtime options
  - Set environment variables used by TAU
  - Control data collection behavior
- All options included context sensitive help



Profile Configurations

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: shallow

Arguments | Environment | Performance Analysis | TAU | Parametric Study

Analysis Options | TAU Runtime | Tau Compiler | Selective Instrumentation | Data Collection

-tau_options='-optNoRevert -optCompInst '

- [ ] Verbose
- [ ] DetectMemoryLeaks
- [ ] PdtGnuFortranParser
- [ ] PdtCleanscapeParser
- [ ] PreProcess
- [ ] KeepFiles
- [ ] Shared
- [ ] PdtDir
- [ ] PdtF95Opts
- [ ] PdtF95Reset
- [ ] PdtCOpts
- [ ] PdtCReset
- [ ] PdtCxxOpts
- [ ] PdtCxxReset
- [ ] PdtF90Parser

Filter matched 4 of

Profile Configurations

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: shallow

Arguments | Environment | Performance Analysis | TAU | Parametric Study

Analysis Options | TAU Runtime | Tau Compiler | Selective Instrumentation | Data Collection

- [ ] Verbose Output
- [x] Enable Profiling
- [ ] Profile Output Directory
- [ ] Profile Format: profile
- [ ] Callpath Profiling
- [ ] Set Callpath Depth: 0
- [ ] Overhead Compensation
- [ ] Track Message
- [ ] Communication Matrix
- [x] Enable Throttling
- [x] Throttle Percall: 0
- [x] Throttle Numcalls: 0
- [ ] Enable Tracing
- [ ] Trace Output Directory
- [x] Synchronize Clocks
- [ ] Track Heap
- [ ] Track Headroom

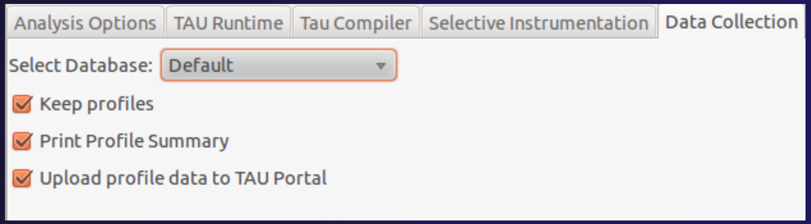
Track memory headroom availability at function entry/exit with a context event

On Data Collection tab: Check Keep profiles, and Print Profile summary

Hover help

TAU/ETFw Hands-On (7)

- If local PerfDMF databases are available you may select one to hold profile output
- A text summary may be printed to the console
- Profiles may be uploaded to the TAU Portal for viewing online
  - tau.nic.uoregon.edu
- Profiles may be copied to your workspace

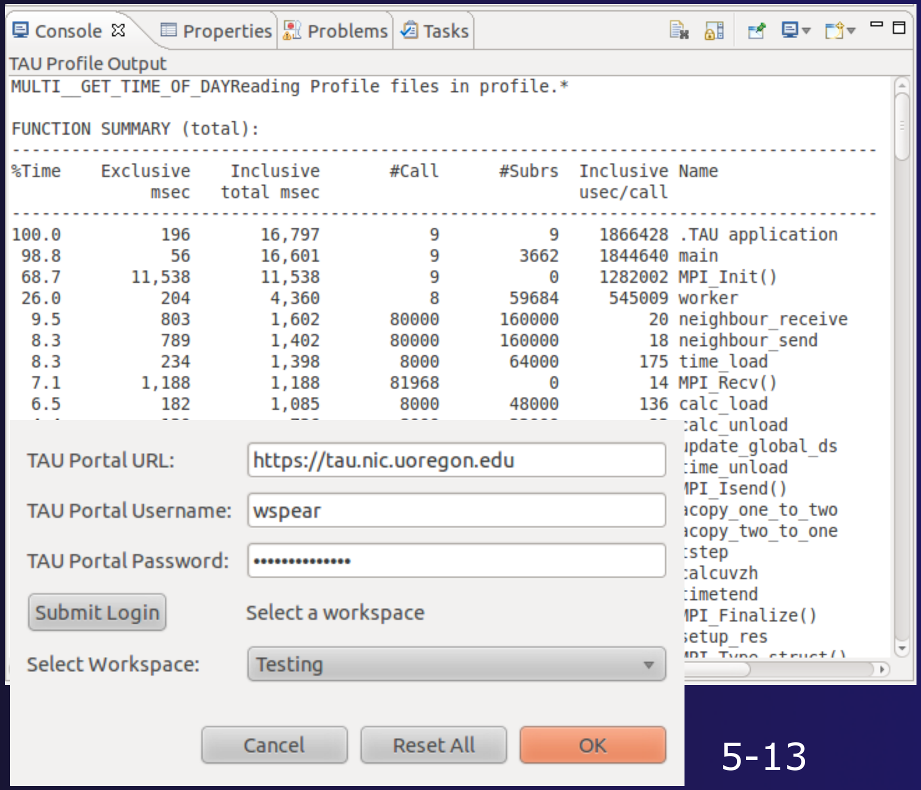


Select Database: Default

✓ Keep profiles

✓ Print Profile Summary

✓ Upload profile data to TAU Portal



TAU Profile Output

```
MULTI__GET_TIME_OF_DAYReading Profile files in profile.*

FUNCTION SUMMARY (total):
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0          196       16,797           9           9    1866428 .TAU application
 98.8           56       16,601           9        3662    1844640 main
 68.7       11,538       11,538           9           0    1282002 MPI_Init()
 26.0          204        4,360           8       59684     545009 worker
  9.5          803        1,602       80000      160000         20 neighbour_receive
  8.3          789        1,402       80000      160000         18 neighbour_send
  8.3          234        1,398        8000       64000        175 time_load
  7.1        1,188        1,188       81968           0         14 MPI_Recv()
  6.5          182        1,085        8000       48000        136 calc_load
                                                                   :alc_unload
                                                                   ipdate_global_ds
                                                                   :ime_unload
                                                                   MPI_Isend()
                                                                   acopy_one_to_two
                                                                   acopy_two_to_one
                                                                   :step
                                                                   :alcuvzh
                                                                   :imetend
                                                                   MPI_Finalize()
                                                                   setup_res
```

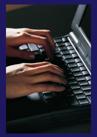
TAU Portal URL: https://tau.nic.uoregon.edu

TAU Portal Username: wspear

TAU Portal Password: ••••••••••••••

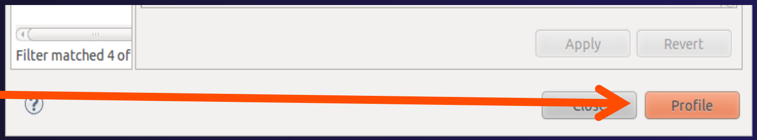
Select a workspace

Select Workspace: Testing

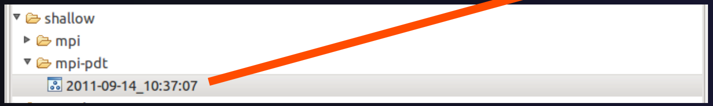


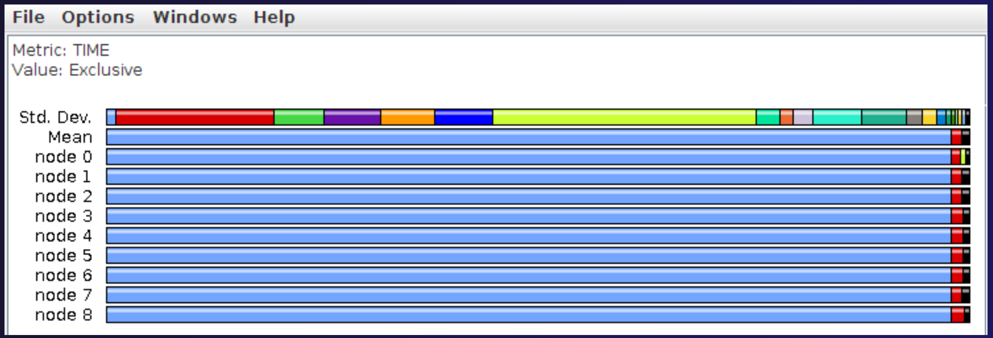
TAU/ETFW Hands-On (8)

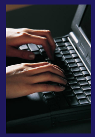
- Once your TAU launch is configured select 'Profile'



- Notice that the project rebuilds with TAU compiler commands
- The project will execute normally but TAU profiles will be generated
- TAU profiles will be processed as specified in the launch configuration.
- If you have a local PerfDMF database the run will show up in the Performance Data Management view
  - Double click the new entry to view in ParaProf
  - Right click on a function bar and select **Show Source Code** for source callback to Eclipse

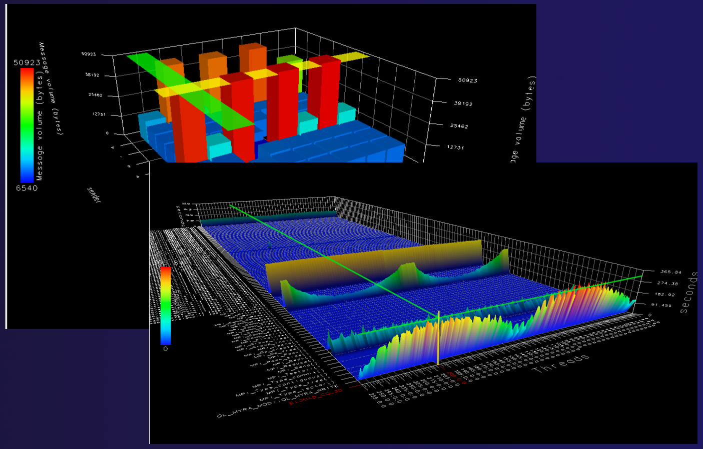






TAU/ETFW Hands-On (9)

- Use ParaProf for profile visualization to identify performance hotspots
  - Inefficient sequential computation
  - Communication overhead
  - IO/Memory bottlenecks
  - Load imbalance
  - Suboptimal cache performance



- Compare multiple trials in PerfExplorer to identify performance regressions and scaling issues
- To use ParaProf, install TAU from tau.uoregon.edu or use Java webstart from tau.uoregon.edu/paraprof

GEM -Graphical Explorer of MPI Programs

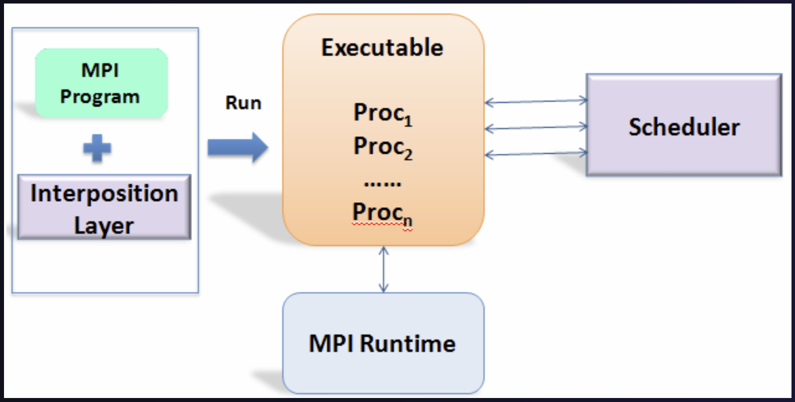
- Objective
  - Become familiar with GEM and how it can be used in the development life-cycle to detect common MPI problems, such as **deadlocks** and **functionally irrelevant barriers**
- Contents
  - Overview of GEM
  - GEM installation and features
- Prerequisites
  - Eclipse IDE and PTP core
  - In-situ Partial Order (ISP), the verification engine

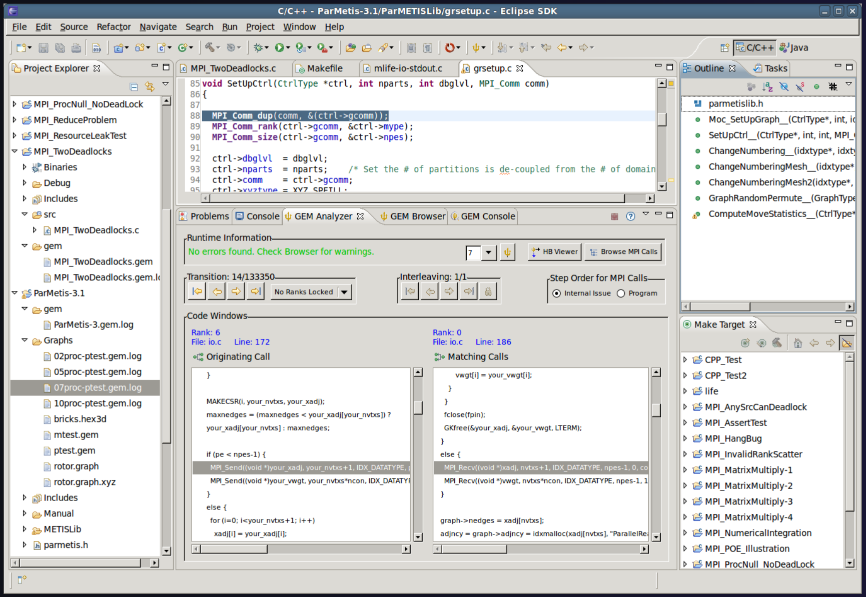
http://www.cs.utah.edu/fv/GEM

GEM
Graphical Explorer of MPI Programs

- Dynamic verification for MPI C/C++ that detects:
  - Deadlocks
  - MPI object leaks (communicators, requests, etc)
  - Functionally irrelevant barriers
  - Local assertion violations
  - MPI Send/Recv Type Mismatches
- Offers rigorous coverage guarantees
  - Complete nondeterministic coverage for MPI (MPI_ANY_SOURCE)
  - Determines relevant interleavings, replaying as necessary
  - Examines communication / synchronization behaviors

GEM - Overview





```
MPI_ProcNull_NoDeadLock
MPI_ReduceProblem
MPI_ResourceLeakTest
MPI_TwoDeadlocks
  Binaries
  Debug
  Includes
  src
    MPI_TwoDeadlocks.c
  gem
    MPI_TwoDeadlocks.gem
    MPI_TwoDeadlocks.gem.lo
ParMetis-3.1
  gem
    ParMetis-3.gem.log
  Graphs
    02proc-ptest.gem.log
    05proc-ptest.gem.log
    07proc-ptest.gem.log
    10proc-ptest.gem.log
    bricks.hex3d
    mtest.gem
    ptest.gem
    rotor.graph
    rotor.graph.xyz
  Includes
  Manual
  METISLib
  parmetis.h
```

```
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;
  ctrl->xyztype = XYZ_SPFILL;
```

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

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, p
   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
Matching Calls

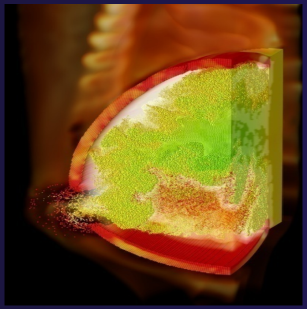
```
      vwgt[i] = your_vwgt[i];
    }
  }
  fclose(fpin);
  GKfree(&your_xadj, &your_vwgt, LTERM);
 }
 else {
  MPI_Recv((void *)xadj, nvtxs+1, IDX_DATATYPE, npes-1, 0, co
  MPI_Recv((void *)vwgt, nvtxs*ncon, IDX_DATATYPE, npes-1, 1
 }

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

```
parmetislib.h
Moc_SetUpGraph__(CtrlType*, int, i
SetUpCtrl__(CtrlType*, int, int, MPI_
ChangeNumbering__(idxtype*, idxt
ChangeNumberingMesh__(idxtype*
ChangeNumberingMesh2(idxtype*,
GraphRandomPermute__(GraphType
ComputeMoveStatistics__(CtrlType*
```

```
CPP_Test
CPP_Test2
life
MPI_AnySrcCanDeadlock
MPI_AssertTest
MPI_HangBug
MPI_InvalidRankScatter
MPI_MatrixMultiply-1
MPI_MatrixMultiply-2
MPI_MatrixMultiply-3
MPI_MatrixMultiply-4
MPI_NumericalIntegration
MPI_POE_Illustration
MPI_ProcNull_NoDeadLock
```

GEM

- Front-end for In-situ Partial Order (ISP) developed at University of Utah
- Contributes "push-button" C/C++ MPI verification and analysis to the development cycle
- Automatically instruments and runs user code, displaying post verification results
- Variety of views & tools to facilitate debugging and MPI runtime understanding


(Image courtesy of Steve Parker, U of Utah)

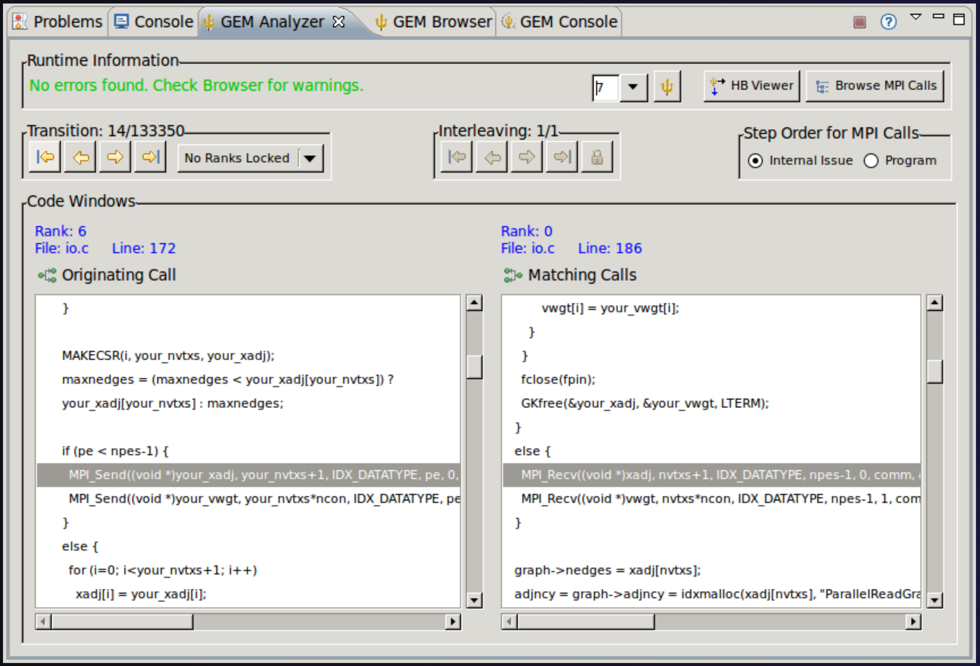
GEM – Views & Tools

Analyzer View

Highlights bugs, and facilitates post-verification review / debugging

Browser View

Groups and localizes MPI problems. Maps errors to source code in Eclipse editor



No errors found. Check Browser for warnings.

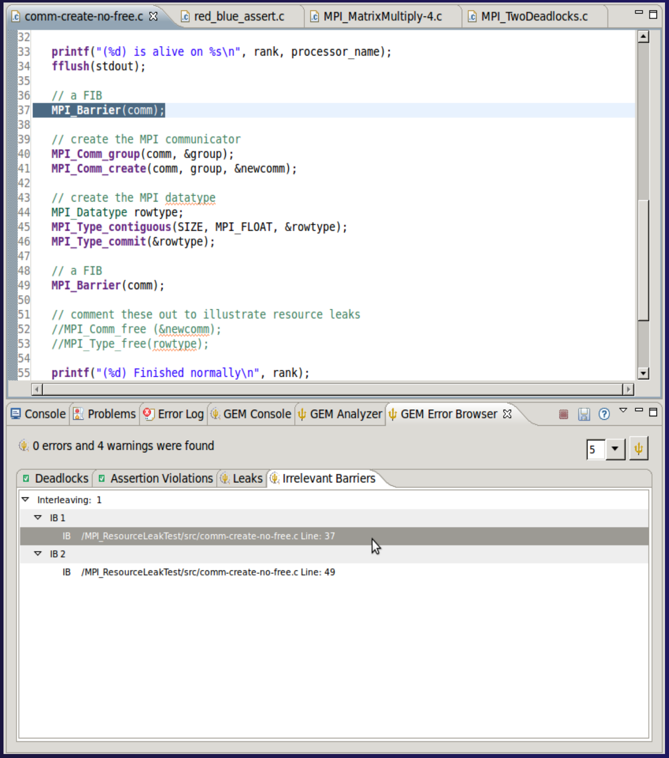
Step Order for MPI Calls: [✓ Internal Issue] Program

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, 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

```
    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
```



```
printf("(%d) is alive on %s\n", rank, processor_name);
fflush(stdout);

// a FIB
MPI_Barrier(comm);

// create the MPI communicator
MPI_Comm_group(comm, &group);
MPI_Comm_create(comm, group, &newcomm);

// create the MPI datatype
MPI_Datatype rowtype;
MPI_Type_contiguous(SIZE, MPI_FLOAT, &rowtype);
MPI_Type_commit(&rowtype);

// a FIB
MPI_Barrier(comm);

// comment these out to illustrate resource leaks
//MPI_Comm_free (&newcomm);
//MPI_Type_free(rowtype);

printf("(%d) Finished normally\n", rank);
```

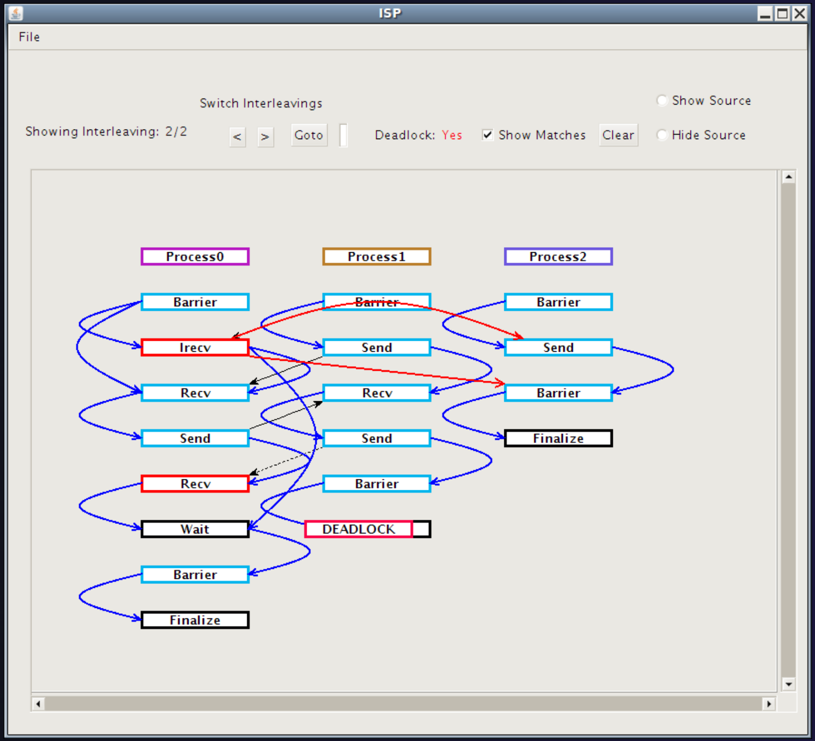
0 errors and 4 warnings were found

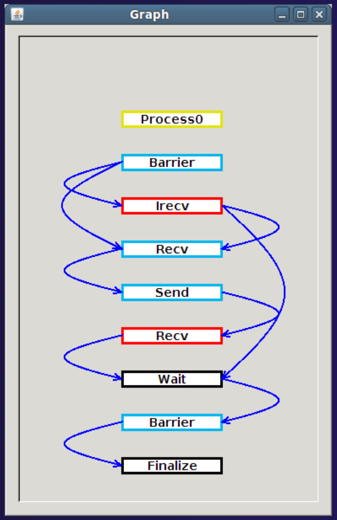
```
Interleaving: 1
  IB 1
    IB  /MPI_ResourceLeakTest/src/comm-create-no-free.c Line: 37
  IB 2
    IB  /MPI_ResourceLeakTest/src/comm-create-no-free.c Line: 49
```

GEM – Views & Tools (cont.)

Happens-Before Viewer

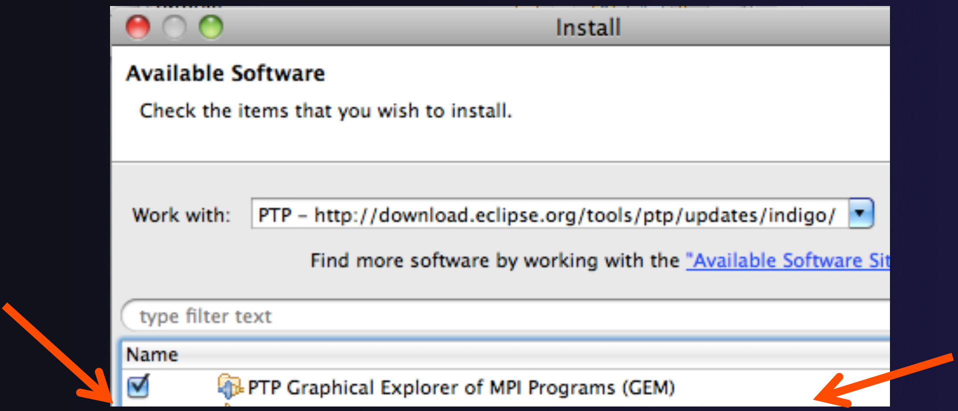
Shows required orderings and communication matches
(currently an external tool)





GEM PTP Installation

- GEM is **NOT** included in the package:
  "Eclipse for Parallel Application Developers"
- Gem is installed as an individual PTP feature. Refer to the individual feature installation section of this tutorial. Be sure GEM is selected.



Install

Available Software

Check the items that you wish to install.

Work with: PTP – http://download.eclipse.org/tools/ptp/updates/indigo/

Find more software by working with the "Available Software Sit

type filter text

Name

☑ PTP Graphical Explorer of MPI Programs (GEM)

To confirm:

- Help>Install New Software
- Select PTP download site
- Check the GEM Extension

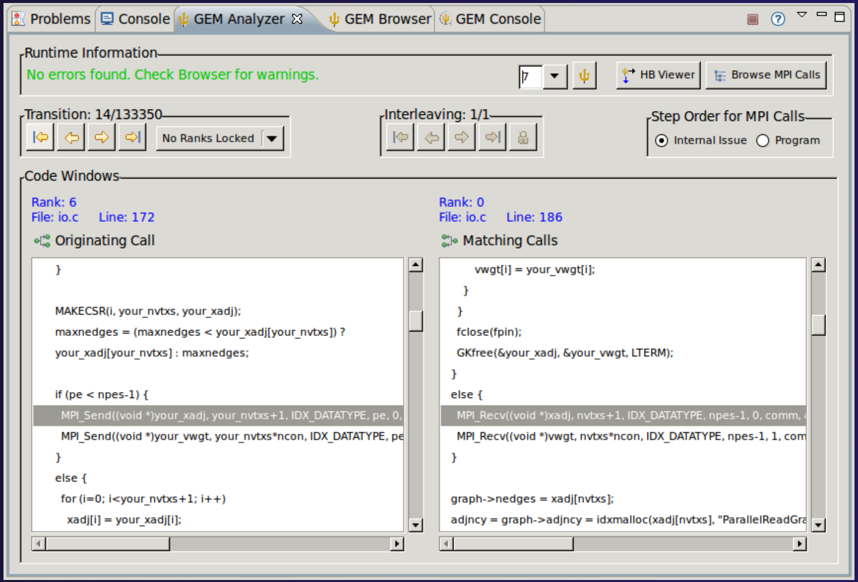
ISP Installation

- **ISP itself must be installed prior to using GEM**
  - Download ISP at http://www.cs.utah.edu/fv/ISP

- Untar ***isp-0.3.0.tar.gz*** into a tmp directory:
  - Configure and install
    - ./configure –-prefix=<ISP install directory>
    - make
    - make install
  - Libraries and necessary scripts are installed

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



No errors found. Check Browser for warnings.

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

Rank: 6
File: io.c Line: 172

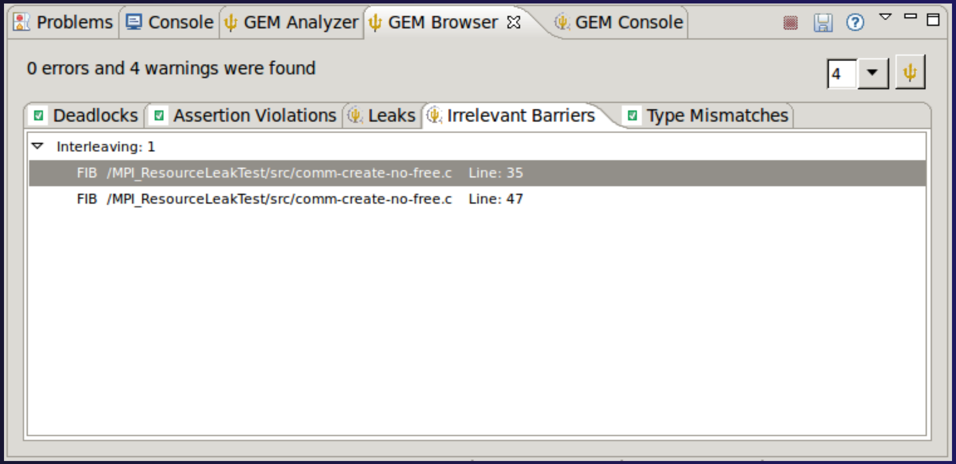
```
    }

    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, 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

```
        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

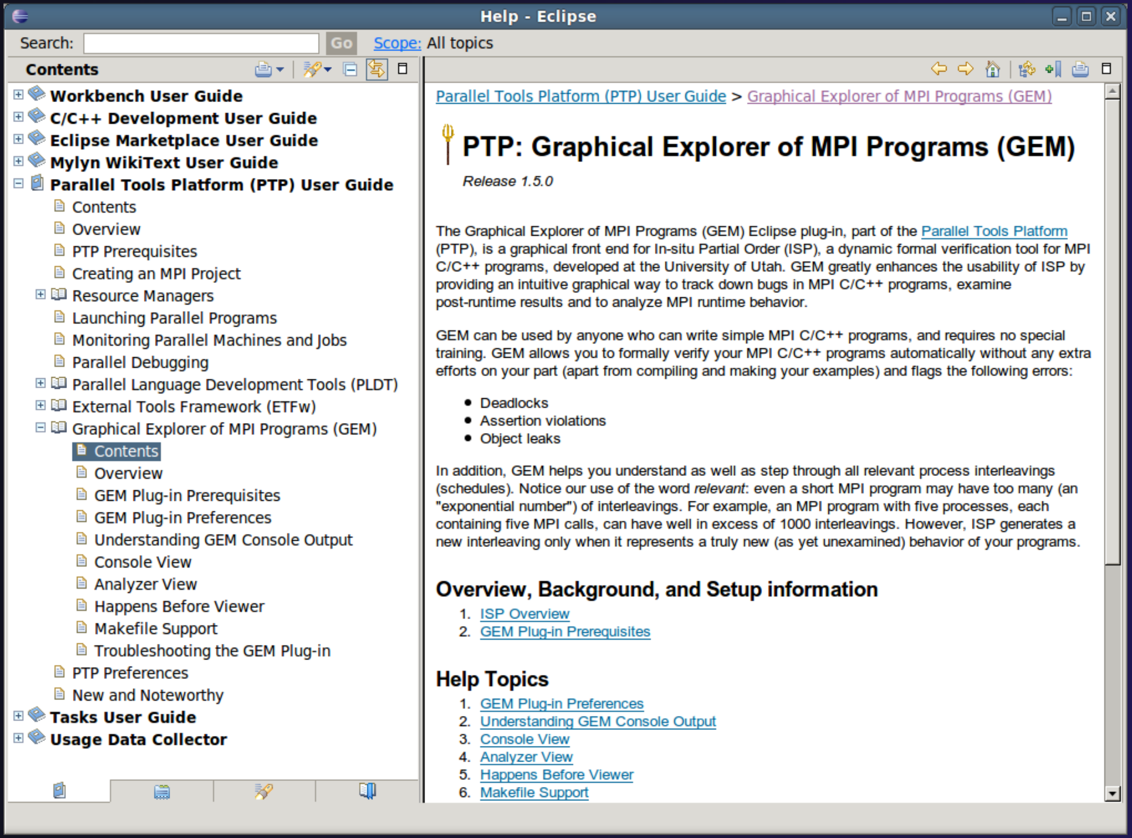


0 errors and 4 warnings were found

```
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



- 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

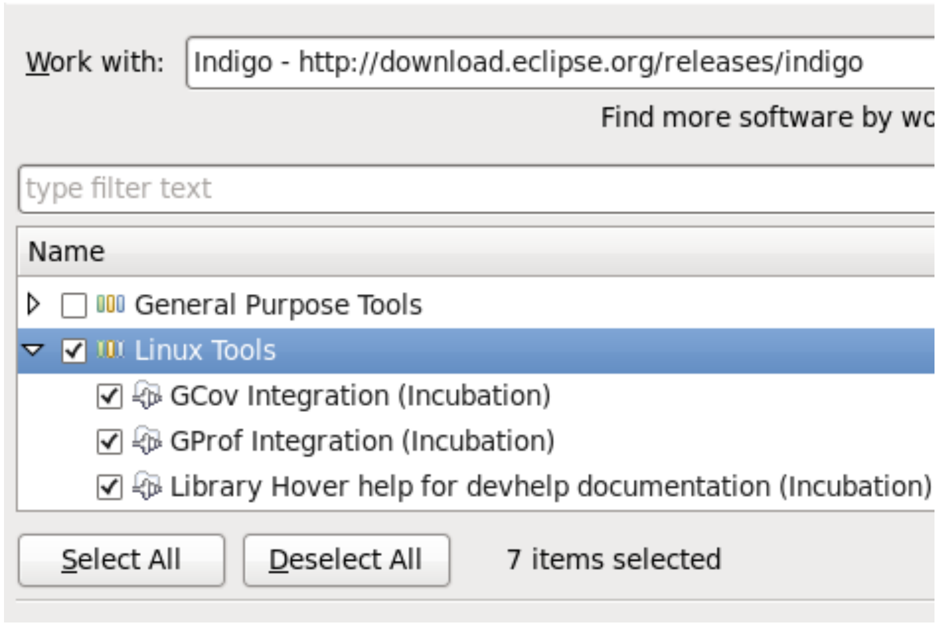
Help Topics

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

Gcov and gprof support in linux tools



Work with: Indigo - http://download.eclipse.org/releases/indigo

Find more software by wo

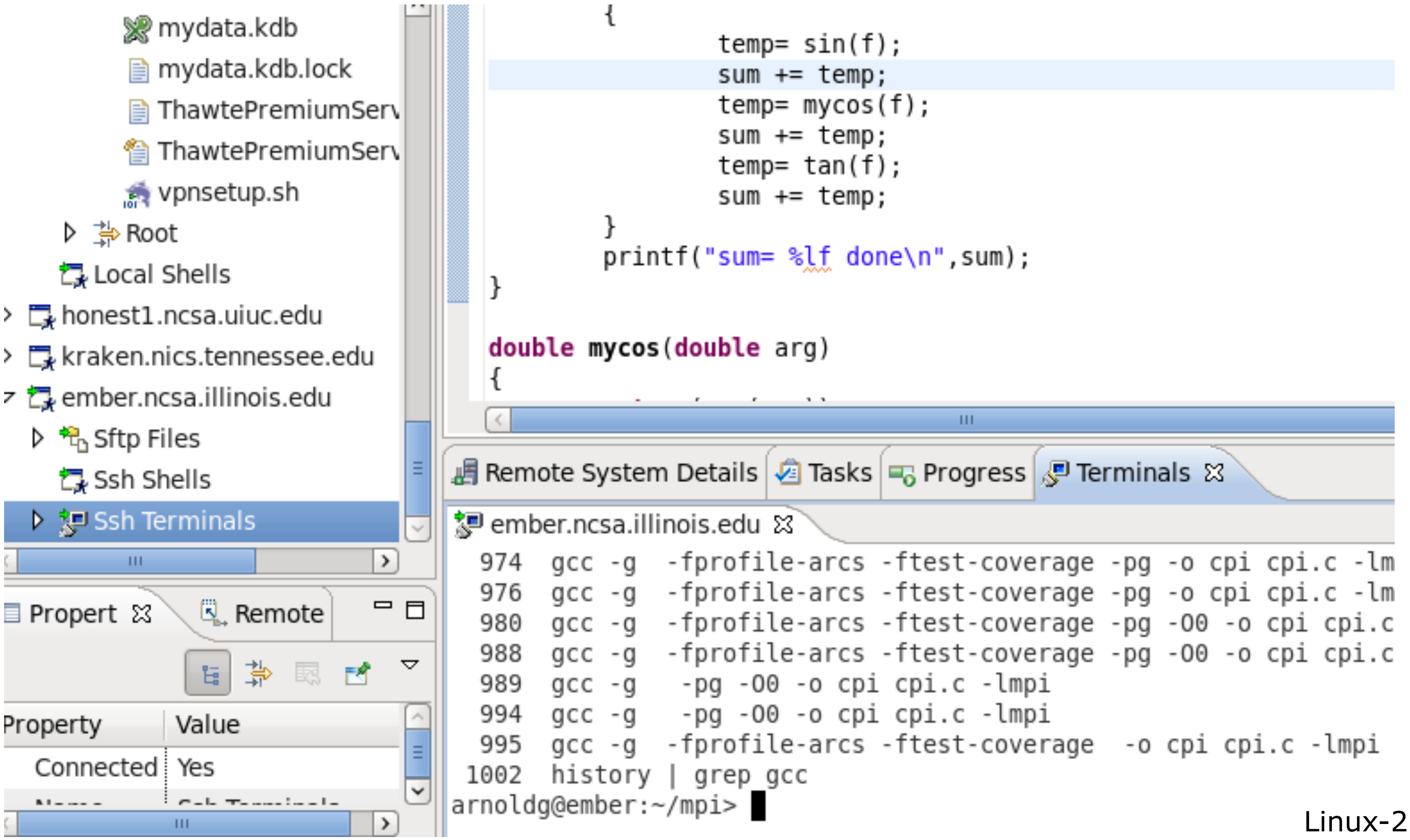
type filter text

Name
☐ General Purpose Tools
✓ Linux Tools
✓ GCov Integration (Incubation)
✓ GProf Integration (Incubation)
✓ Library Hover help for devhelp documentation (Incubation)

7 items selected

- gcov
- gprof
- Linux tools

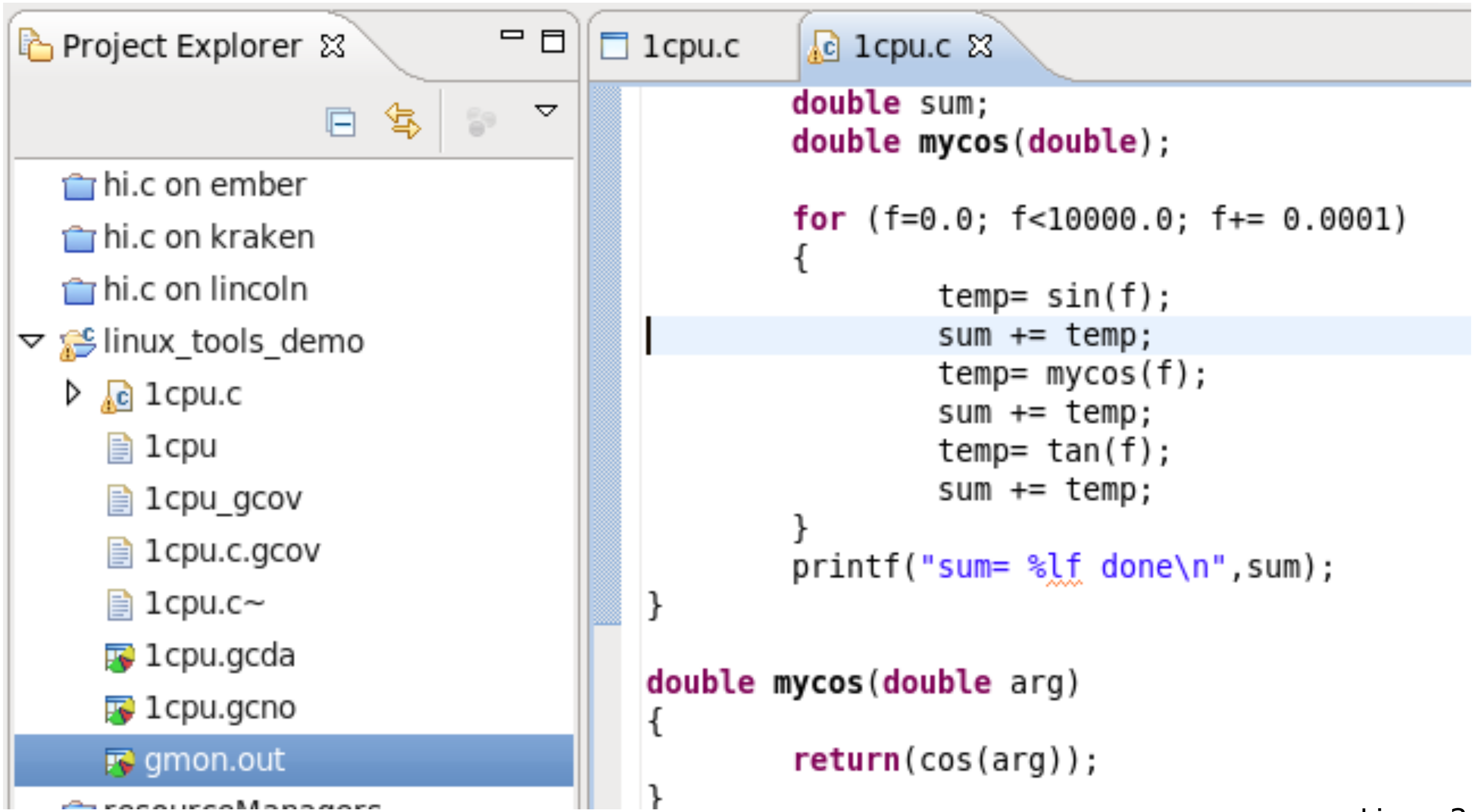
Gprof and gcov setup (local or remote)



```c
        {
                temp= sin(f);
                sum += temp;
                temp= mycos(f);
                sum += temp;
                temp= tan(f);
                sum += temp;
        }
        printf("sum= %lf done\n",sum);
}

double mycos(double arg)
{
```

```
  974  gcc -g  -fprofile-arcs -ftest-coverage -pg -o cpi cpi.c -lm
  976  gcc -g  -fprofile-arcs -ftest-coverage -pg -o cpi cpi.c -lm
  980  gcc -g  -fprofile-arcs -ftest-coverage -pg -O0 -o cpi cpi.c
  988  gcc -g  -fprofile-arcs -ftest-coverage -pg -O0 -o cpi cpi.c
  989  gcc -g   -pg -O0 -o cpi cpi.c -lmpi
  994  gcc -g   -pg -O0 -o cpi cpi.c -lmpi
  995  gcc -g  -fprofile-arcs -ftest-coverage  -o cpi cpi.c -lmpi
 1002  history | grep gcc
arnoldg@ember:~/mpi> █
```

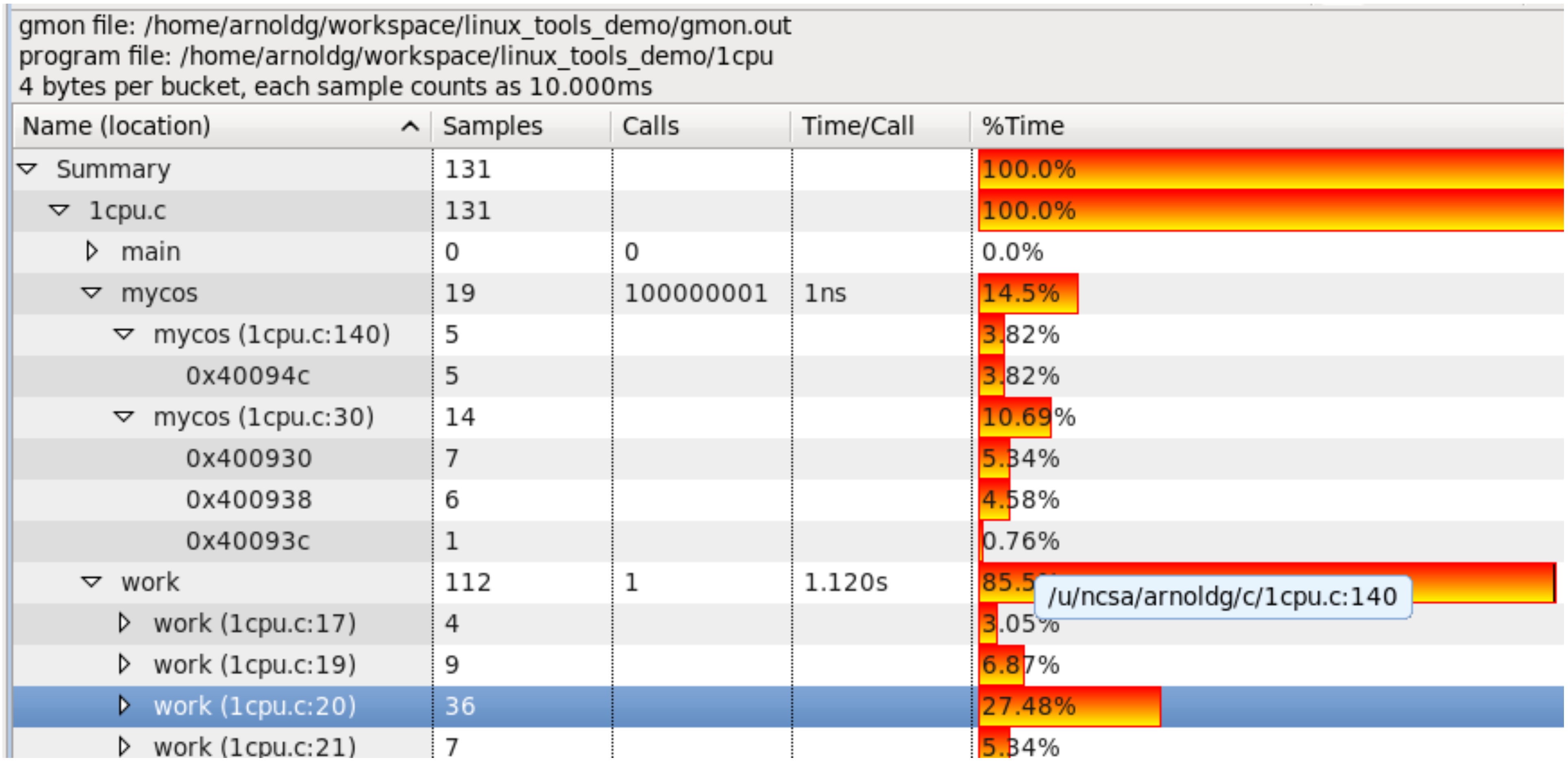
Run code, inspect gprof output



```
hi.c on ember
hi.c on kraken
hi.c on lincoln
linux_tools_demo
  1cpu.c
  1cpu
  1cpu_gcov
  1cpu.c.gcov
  1cpu.c~
  1cpu.gcda
  1cpu.gcno
  gmon.out
```

```
        double sum;
        double mycos(double);

        for (f=0.0; f<10000.0; f+= 0.0001)
        {
                temp= sin(f);
                sum += temp;
                temp= mycos(f);
                sum += temp;
                temp= tan(f);
                sum += temp;
        }
        printf("sum= %lf done\n",sum);
}

double mycos(double arg)
{
        return(cos(arg));
}
```

Gprof tab

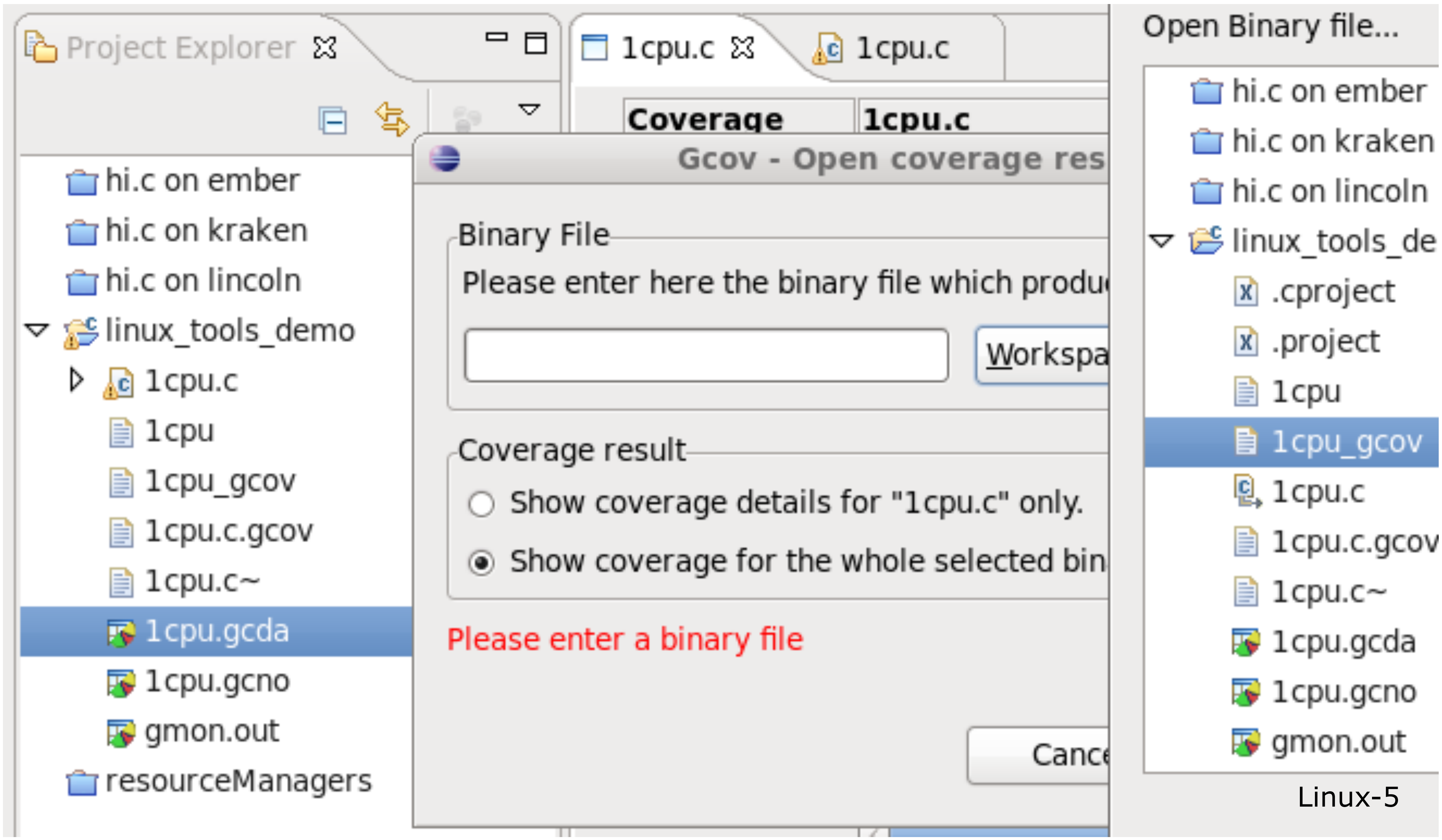


gmon file: /home/arnoldg/workspace/linux_tools_demo/gmon.out
program file: /home/arnoldg/workspace/linux_tools_demo/1cpu
4 bytes per bucket, each sample counts as 10.000ms

Name (location)	Samples	Calls	Time/Call	%Time
Summary	131			100.0%
1cpu.c	131			100.0%
main	0	0		0.0%
mycos	19	100000001	1ns	14.5%
mycos (1cpu.c:140)	5			3.82%
0x40094c	5			3.82%
mycos (1cpu.c:30)	14			10.69%
0x400930	7			5.34%
0x400938	6			4.58%
0x40093c	1			0.76%
work	112	1	1.120s	85.5
work (1cpu.c:17)	4			3.05%
work (1cpu.c:19)	9			6.87%
work (1cpu.c:20)	36			27.48%
work (1cpu.c:21)	7			5.34%

/u/ncsa/arnoldg/c/1cpu.c:140

Gcov code coverage



Project Explorer

hi.c on ember
hi.c on kraken
hi.c on lincoln
linux_tools_demo
1cpu.c
1cpu
1cpu_gcov
1cpu.c.gcov
1cpu.c~
1cpu.gcda
1cpu.gcno
gmon.out
resourceManagers

1cpu.c 1cpu.c

Coverage 1cpu.c

Gcov - Open coverage res

Binary File

Please enter here the binary file which produ

Workspa

Coverage result

Show coverage details for "1cpu.c" only.

[✓ Show coverage for the whole selected bin]

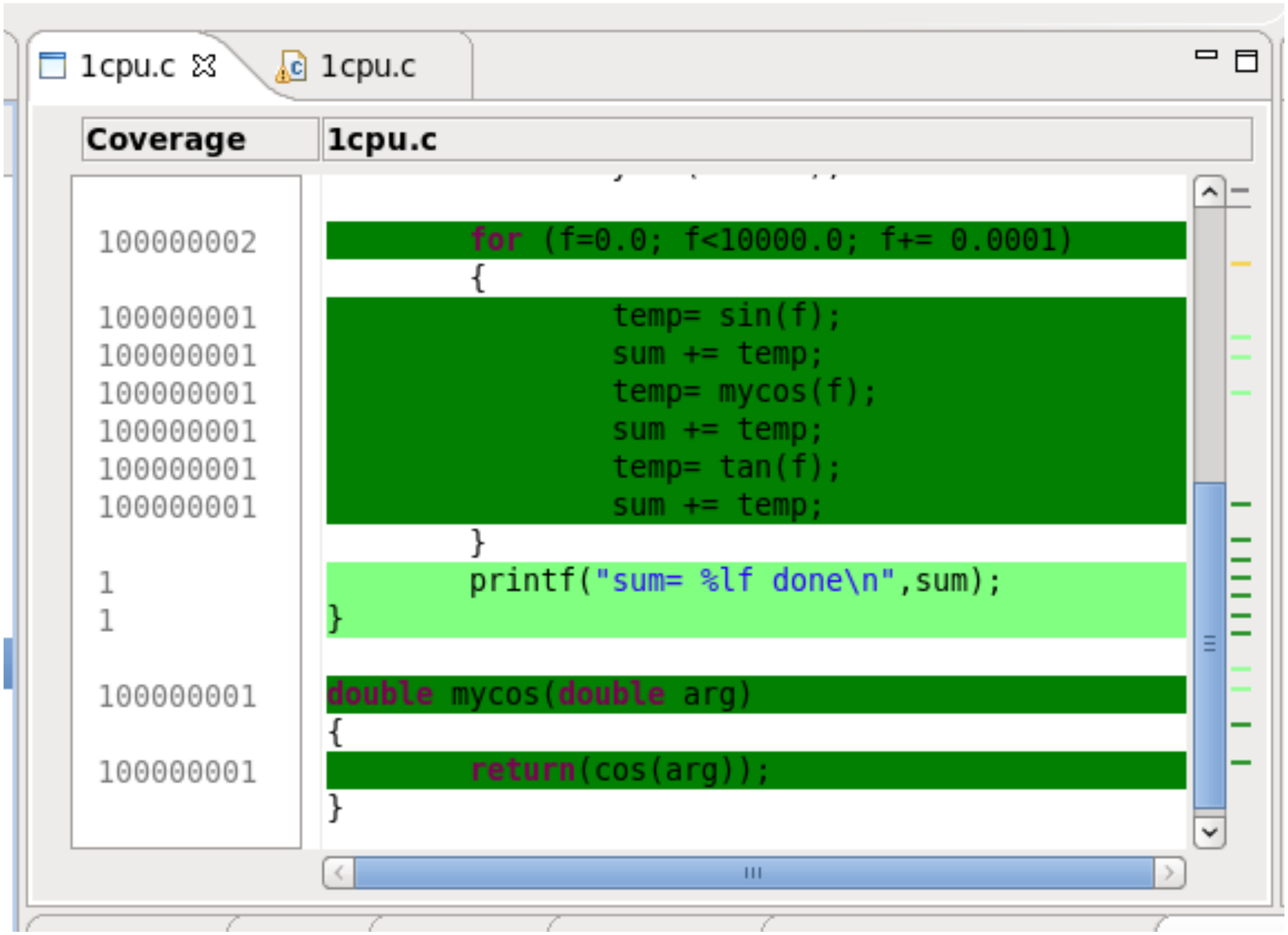
Please enter a binary file

Cance

Open Binary file...

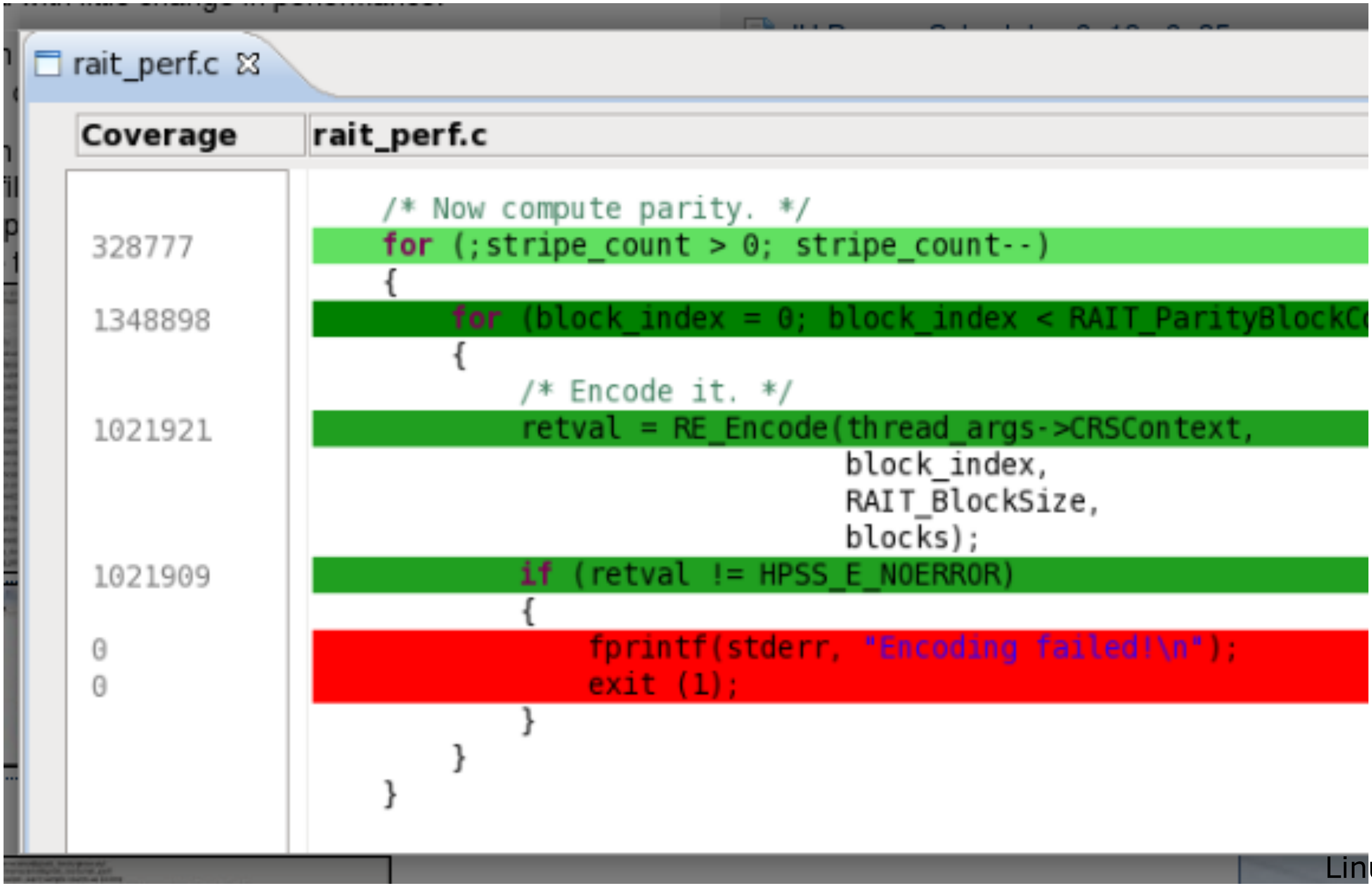
hi.c on ember
hi.c on kraken
hi.c on lincoln
linux_tools_de
.cproject
.project
1cpu
1cpu_gcov
1cpu.c
1cpu.c.gcov
1cpu.c~
1cpu.gcda
1cpu.gcno
gmon.out

Run code, inspect gcov display



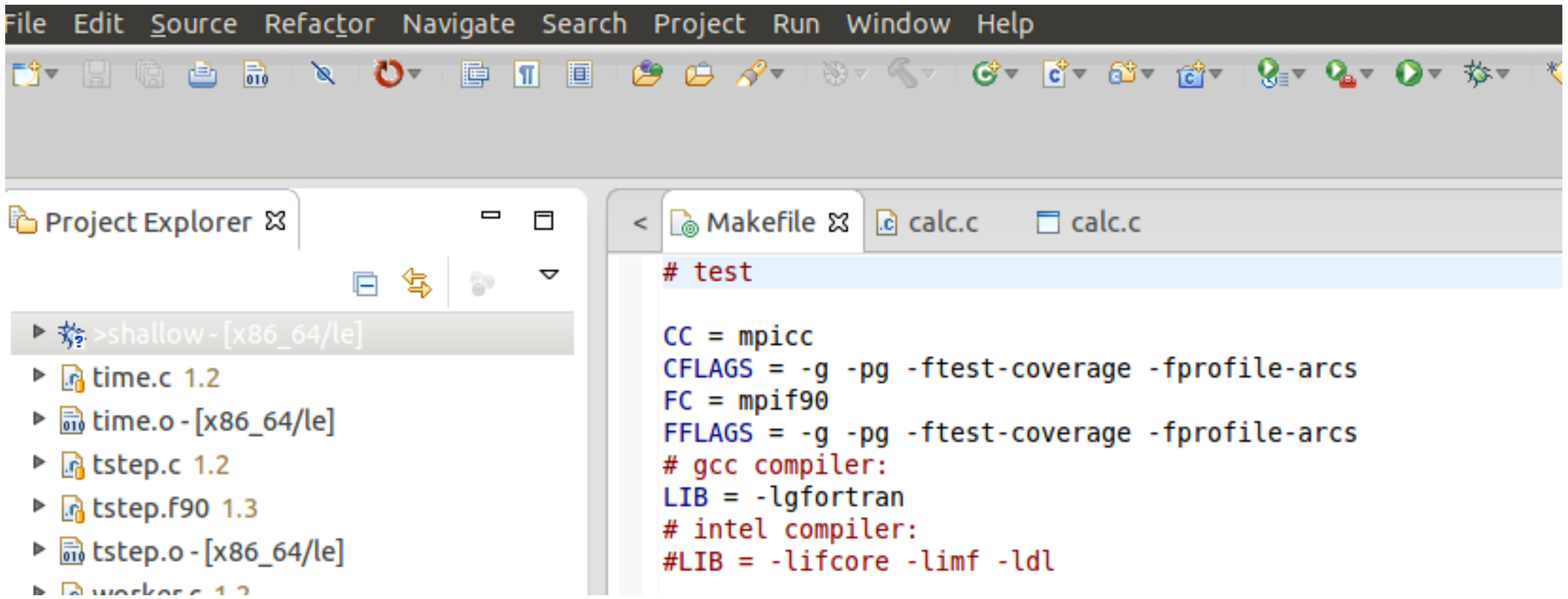
```
100000002        for (f=0.0; f<10000.0; f+= 0.0001)
                 {
100000001                temp= sin(f);
100000001                sum += temp;
100000001                temp= mycos(f);
100000001                sum += temp;
100000001                temp= tan(f);
100000001                sum += temp;
                 }
1                printf("sum= %lf done\n",sum);
1        }

100000001    double mycos(double arg)
             {
100000001            return(cos(arg));
             }
```

Gcov with a production code, unexecuted region



```
Coverage   rait_perf.c

                 /* Now compute parity. */
328777           for (;stripe_count > 0; stripe_count--)
                 {
1348898              for (block_index = 0; block_index < RAIT_ParityBlockC
                     {
                         /* Encode it. */
1021921                  retval = RE_Encode(thread_args->CRSContext,
                                                 block_index,
                                                 RAIT_BlockSize,
                                                 blocks);
1021909                  if (retval != HPSS_E_NOERROR)
                         {
0                            fprintf(stderr, "Encoding failed!\n");
0                            exit (1);
                         }
                     }
                 }
```

Gmon with shallow project, MPI

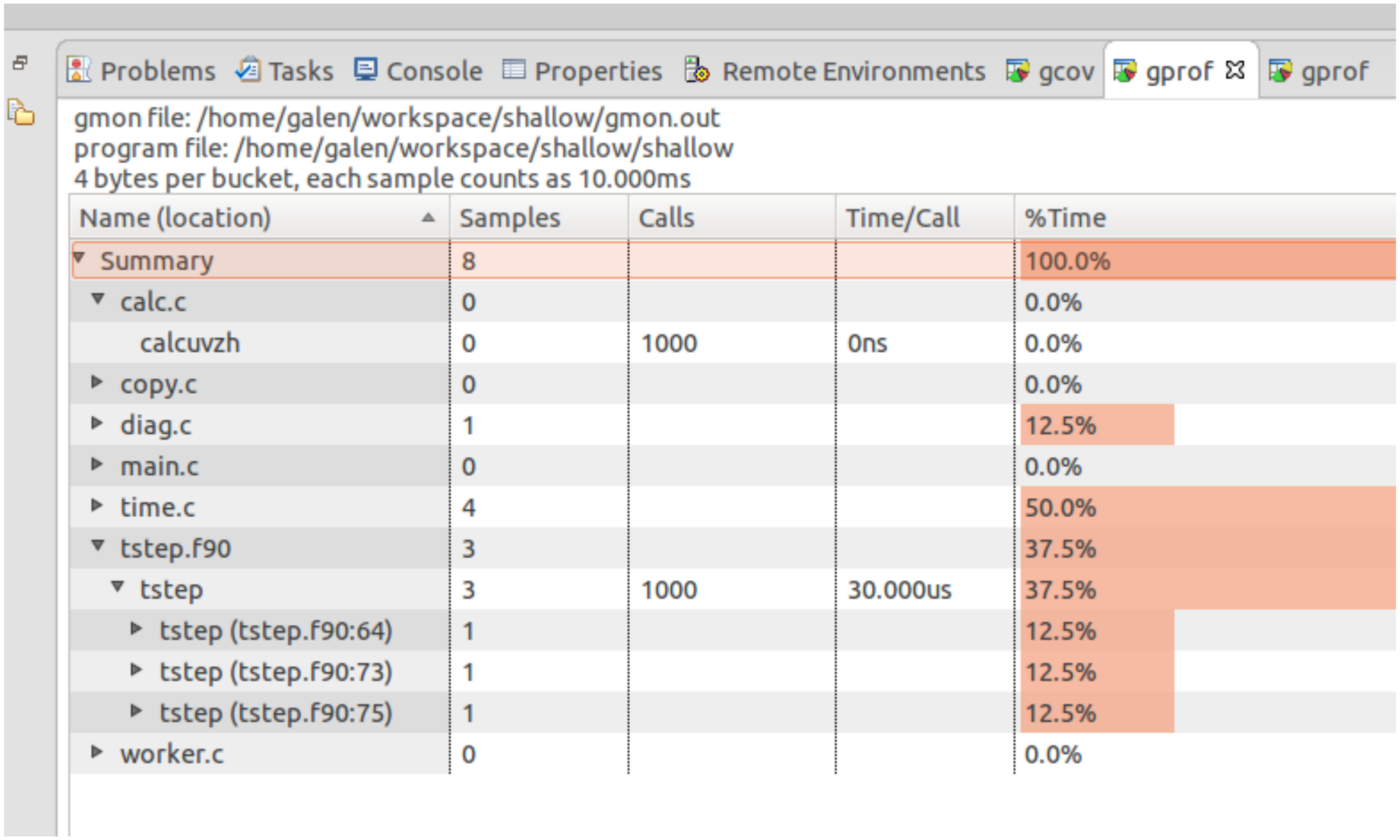


```
# test

CC = mpicc
CFLAGS = -g -pg -ftest-coverage -fprofile-arcs
FC = mpif90
FFLAGS = -g -pg -ftest-coverage -fprofile-arcs
# gcc compiler:
LIB = -lgfortran
# intel compiler:
#LIB = -lifcore -limf -ldl
```

Setup the MPI run configuration with the Environment variable GMON_OUT_PREFIX defined with a name for your individual MPI rank gmon outputs. By default gmon.out is used but MPI doesn't do that well and you end up with a profile that's missing most of the information, so by using GMON_OUT_PREFIX, each MPI rank adds its process id to its gmon output filename.

Gmon with shallow project, 1 rank

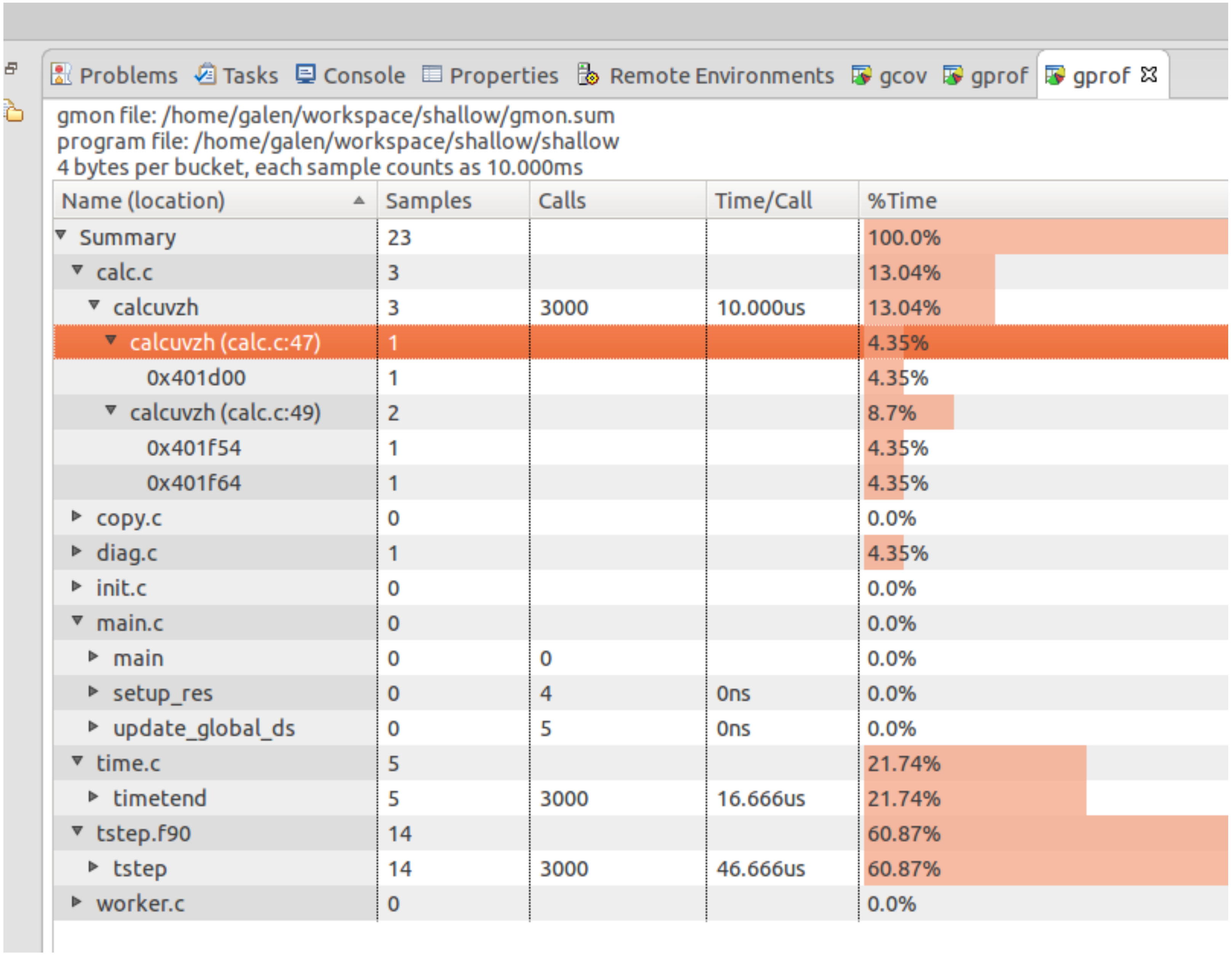
The gmon output files can be combined in a summary with the gprof -s command as shown. It's interesting to compare the summary gmon output to that from one of the ranks (copy a rank's gmon_* to gmon.out to easily view it with Eclipse linuxtools).



```
gmon file: /home/galen/workspace/shallow/gmon.out
program file: /home/galen/workspace/shallow/shallow
4 bytes per bucket, each sample counts as 10.000ms
```

Name (location)	Samples	Calls	Time/Call	%Time
▼ Summary	8			100.0%
▼ calc.c	0			0.0%
calcuvzh	0	1000	0ns	0.0%
▶ copy.c	0			0.0%
▶ diag.c	1			12.5%
▶ main.c	0			0.0%
▶ time.c	4			50.0%
▼ tstep.f90	3			37.5%
▼ tstep	3	1000	30.000us	37.5%
▶ tstep (tstep.f90:64)	1			12.5%
▶ tstep (tstep.f90:73)	1			12.5%
▶ tstep (tstep.f90:75)	1			12.5%
▶ worker.c	0			0.0%

Gmon with shallow project, summary

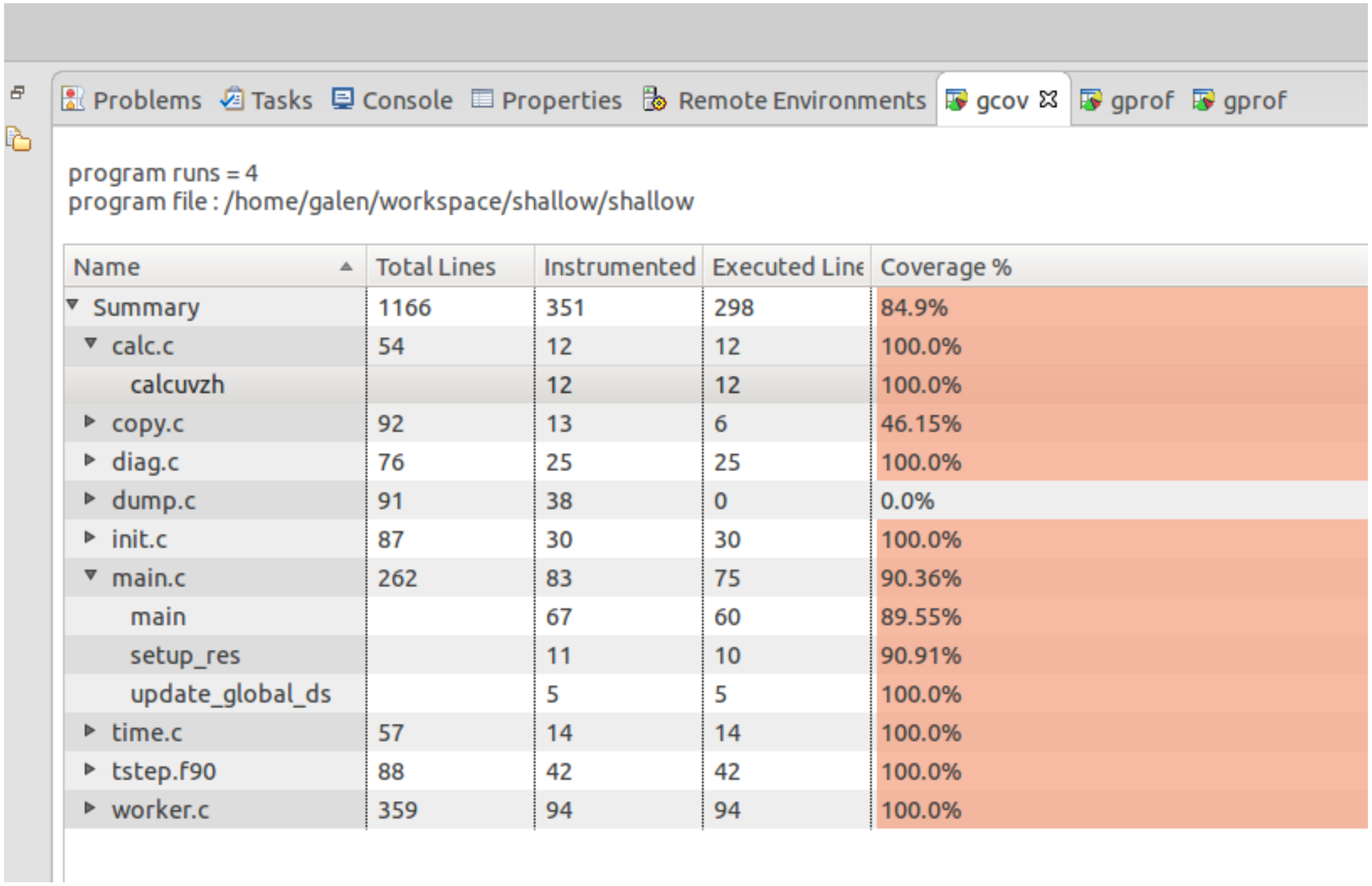


gmon file: /home/galen/workspace/shallow/gmon.sum
program file: /home/galen/workspace/shallow/shallow
4 bytes per bucket, each sample counts as 10.000ms

Name (location)	Samples	Calls	Time/Call	%Time
▼ Summary	23			100.0%
▼ calc.c	3			13.04%
▼ calcuvzh	3	3000	10.000us	13.04%
▼ calcuvzh (calc.c:47)	1			4.35%
0x401d00	1			4.35%
▼ calcuvzh (calc.c:49)	2			8.7%
0x401f54	1			4.35%
0x401f64	1			4.35%
▶ copy.c	0			0.0%
▶ diag.c	1			4.35%
▶ init.c	0			0.0%
▼ main.c	0			0.0%
▶ main	0	0		0.0%
▶ setup_res	0	4	0ns	0.0%
▶ update_global_ds	0	5	0ns	0.0%
▼ time.c	5			21.74%
▶ timetend	5	3000	16.666us	21.74%
▼ tstep.f90	14			60.87%
▶ tstep	14	3000	46.666us	60.87%
▶ worker.c	0			0.0%

Gcov with shallow project

The gcov view is simlar to the gprof view but keep in mind that you're looking at code coverage and not necessarily performance or timing information (though there is a relationship...code not executed is performing quite well !). Also note that multiple executions will accumulate values in the gcov output files until they are removed or truncated to zero-length (2nd run to demonstrate this).

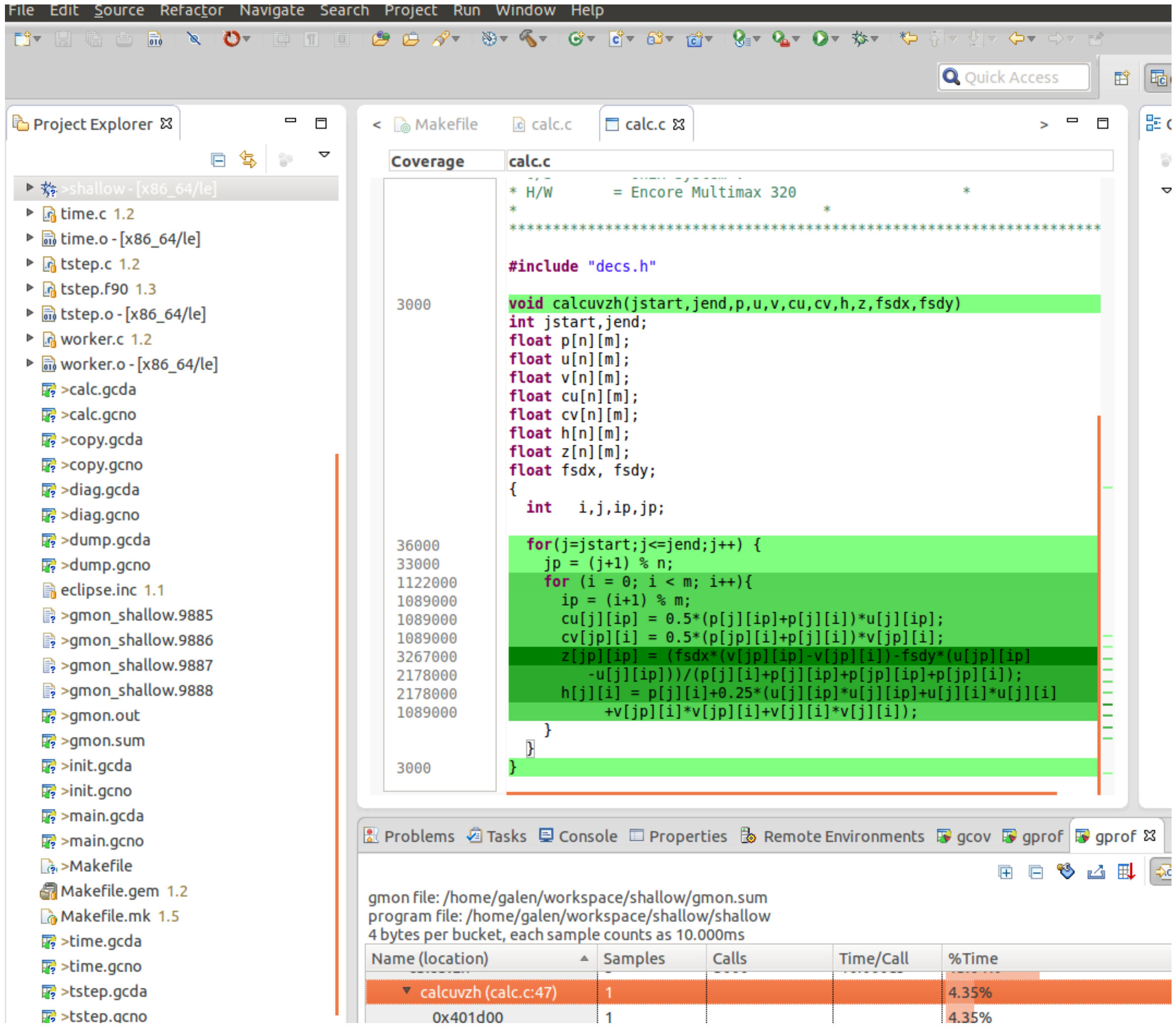


program runs = 4
program file : /home/galen/workspace/shallow/shallow

Name	Total Lines	Instrumented	Executed Line	Coverage %
Summary	1166	351	298	84.9%
calc.c	54	12	12	100.0%
calcuvzh		12	12	100.0%
copy.c	92	13	6	46.15%
diag.c	76	25	25	100.0%
dump.c	91	38	0	0.0%
init.c	87	30	30	100.0%
main.c	262	83	75	90.36%
main		67	60	89.55%
setup_res		11	10	90.91%
update_global_ds		5	5	100.0%
time.c	57	14	14	100.0%
tstep.f90	88	42	42	100.0%
worker.c	359	94	94	100.0%

Gcov with shallow project, cdt integration

Selecting (double click) a source code line from either the gcov or gprof view and you'll see the file and routine highlighted in the cdt c/c++ perspective. Also notice the support for the .f90 file and its routines.


File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Project Explorer
Makefile
calc.c
calc.c
Problems Tasks Console Properties Remote Environments gcov gprof gprof

```
>shallow - [x86_64/le]
time.c 1.2
time.o - [x86_64/le]
tstep.c 1.2
tstep.f90 1.3
tstep.o - [x86_64/le]
worker.c 1.2
worker.o - [x86_64/le]
>calc.gcda
>calc.gcno
>copy.gcda
>copy.gcno
>diag.gcda
>diag.gcno
>dump.gcda
>dump.gcno
eclipse.inc 1.1
>gmon_shallow.9885
>gmon_shallow.9886
>gmon_shallow.9887
>gmon_shallow.9888
>gmon.out
>gmon.sum
>init.gcda
>init.gcno
>main.gcda
>main.gcno
>Makefile
Makefile.gem 1.2
Makefile.mk 1.5
>time.gcda
>time.gcno
>tstep.gcda
>tstep.gcno
```

```
Coverage   calc.c
           * H/W      = Encore Multimax 320           *
           *                             *
           *******************************************************************

           #include "decs.h"

3000       void calcuvzh(jstart,jend,p,u,v,cu,cv,h,z,fsdx,fsdy)
           int jstart,jend;
           float p[n][m];
           float u[n][m];
           float v[n][m];
           float cu[n][m];
           float cv[n][m];
           float h[n][m];
           float z[n][m];
           float fsdx, fsdy;
           {
             int   i,j,ip,jp;

36000        for(j=jstart;j<=jend;j++) {
33000          jp = (j+1) % n;
1122000        for (i = 0; i < m; i++){
1089000          ip = (i+1) % m;
1089000          cu[j][ip] = 0.5*(p[j][ip]+p[j][i])*u[j][ip];
1089000          cv[jp][i] = 0.5*(p[jp][i]+p[j][i])*v[jp][i];
3267000          z[jp][ip] = (fsdx*(v[jp][ip]-v[jp][i])-fsdy*(u[jp][ip]
2178000             -u[j][ip]))/(p[j][i]+p[j][ip]+p[jp][ip]+p[jp][i]);
2178000          h[j][i] = p[j][i]+0.25*(u[j][ip]*u[j][ip]+u[j][i]*u[j][i]
1089000               +v[jp][i]*v[jp][i]+v[j][i]*v[j][i]);
               }
             }
3000       }
```

gmon file: /home/galen/workspace/shallow/gmon.sum
program file: /home/galen/workspace/shallow/shallow
4 bytes per bucket, each sample counts as 10.000ms

Name (location)	Samples	Calls	Time/Call	%Time
calcuvzh (calc.c:47)	1			4.35%
0x401d00	1			4.35%

Linux-12

Tutorial 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

Planned PTP Future Work

- Scalability improvements
  - UI to support 1M processes
  - Very large application support
- Usability improvements
  - New wizard to improve setup experience
  - Ability to share configuration information
- Target Systems
  - Support for additional resource managers and target systems

Useful Eclipse Tools

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

Online Information

- Information about PTP
  - PTP online help
    - http://help.eclipse.org
  - Main web site for downloads, documentation, etc.
    - http://eclipse.org/ptp
  - Wiki for designs, planning, meetings, etc.
    - http://wiki.eclipse.org/PTP
- Information about Photran
  - Main web site for downloads, documentation, etc.
    - http://eclipse.org/photran

Mailing Lists

- User Mailing Lists
  - PTP
    - http://dev.eclipse.org/mailman/listinfo/ptp-user
  - Photran
    - http://dev.eclipse.org/mailman/listinfo/photran
  - Major announcements (new releases, etc.) - low volume
    - http://dev.eclipse.org/mailman/listinfo/ptp-announce
- Developer Mailing Lists
  - Developer discussions - higher volume
    - http://dev.eclipse.org/mailman/listinfo/ptp-dev

Getting Involved

- See http://eclipse.org/ptp
- Read the developer documentation on the wiki
  - http://wiki.eclipse.org/PTP
- Join the mailing lists
- Attend the monthly developer meetings
  - Conf Call Monthly: Second Tuesday, 1:00 pm ET
  - Details on the PTP wiki
- Attend the monthly user meetings
  - Teleconf Monthly: 4th Wednesday, 1:00 pm ET
  - Details on the PTP wiki

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