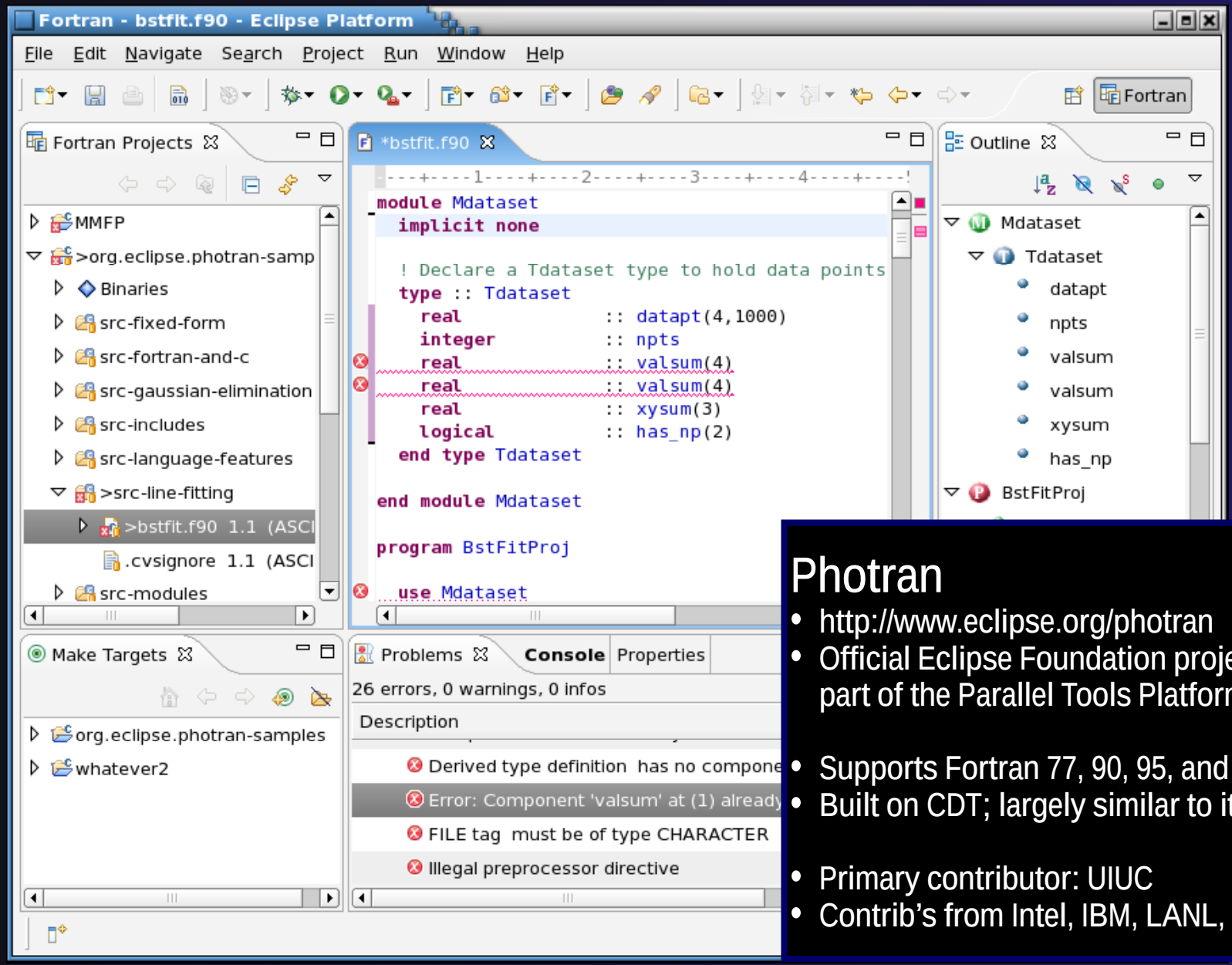
# Module 6: Fortran

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


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


…which became the name of their Fortran IDE.

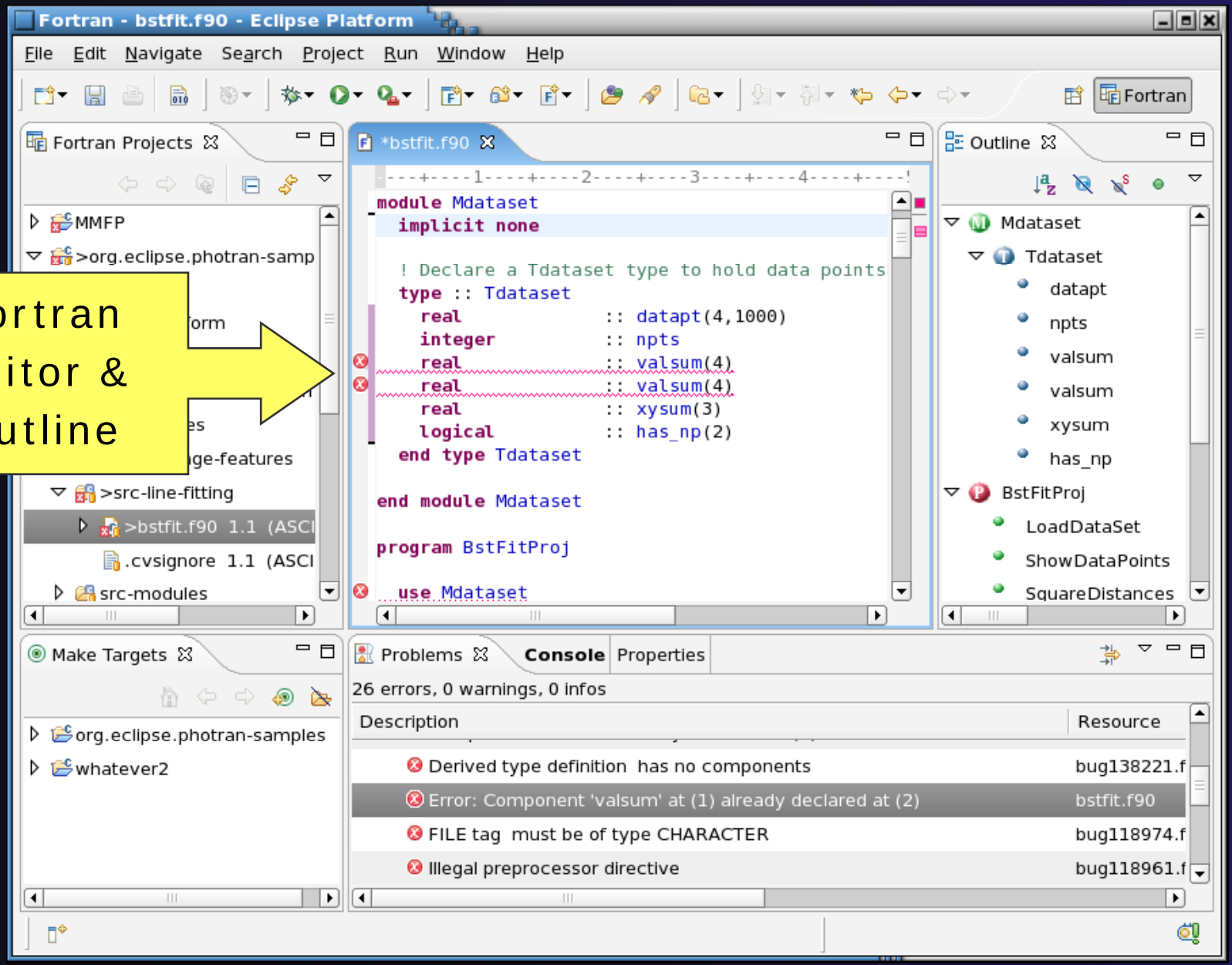



## Photran

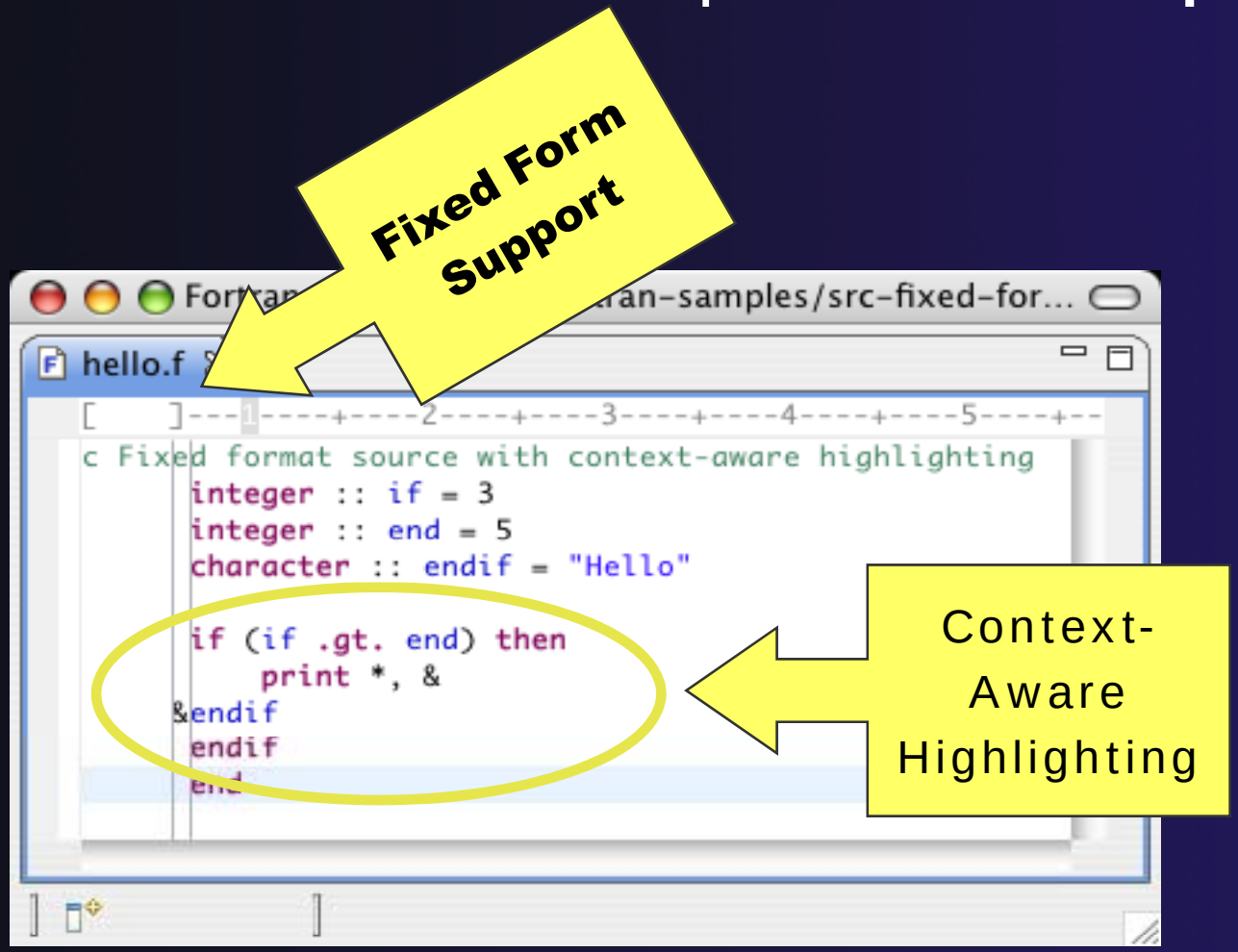
- http://www.eclipse.org/photran
- Official Eclipse Foundation project; part of the Parallel Tools Platform (PTP)

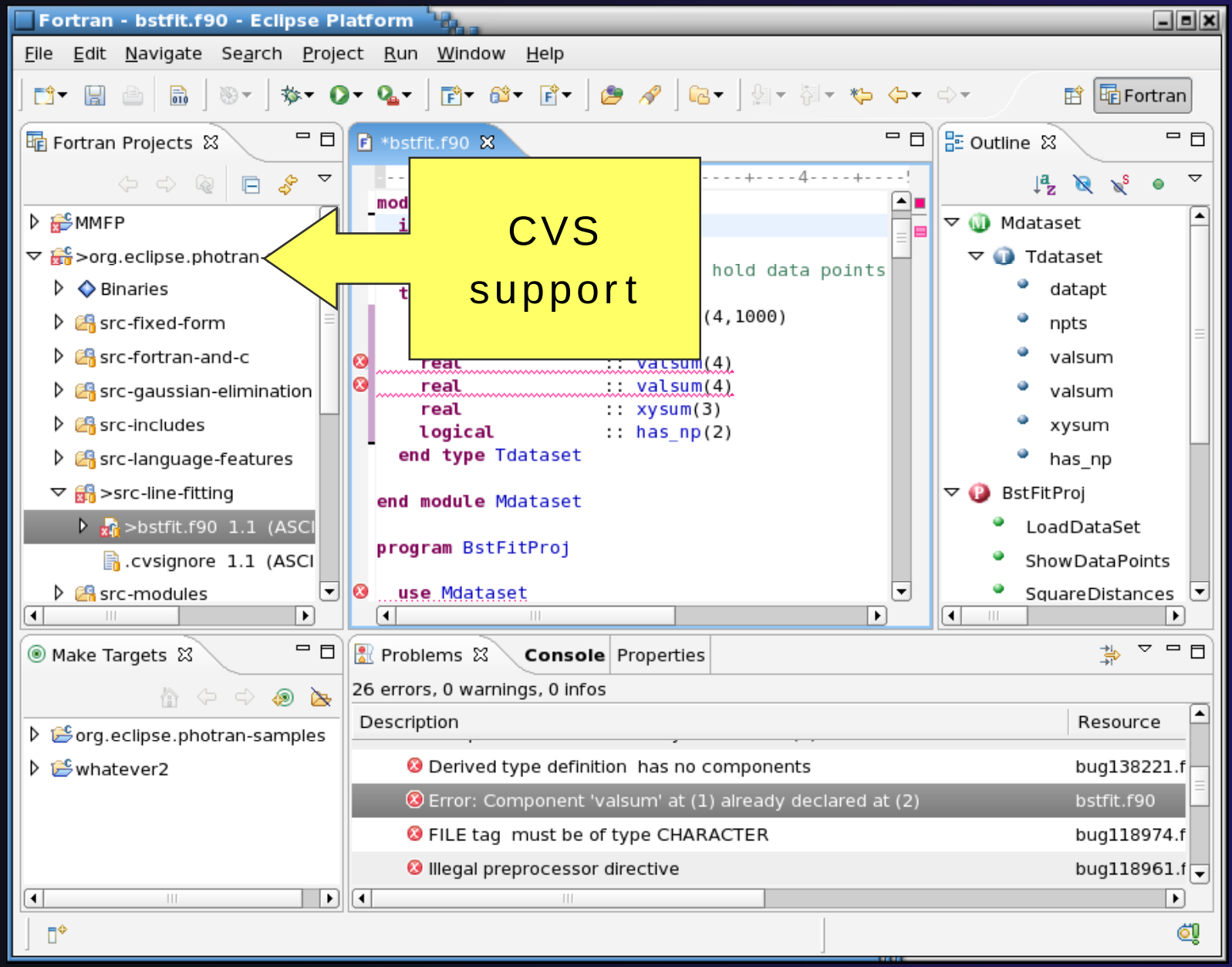
- Supports Fortran 77, 90, 95, and 2003
- Built on CDT; largely similar to it

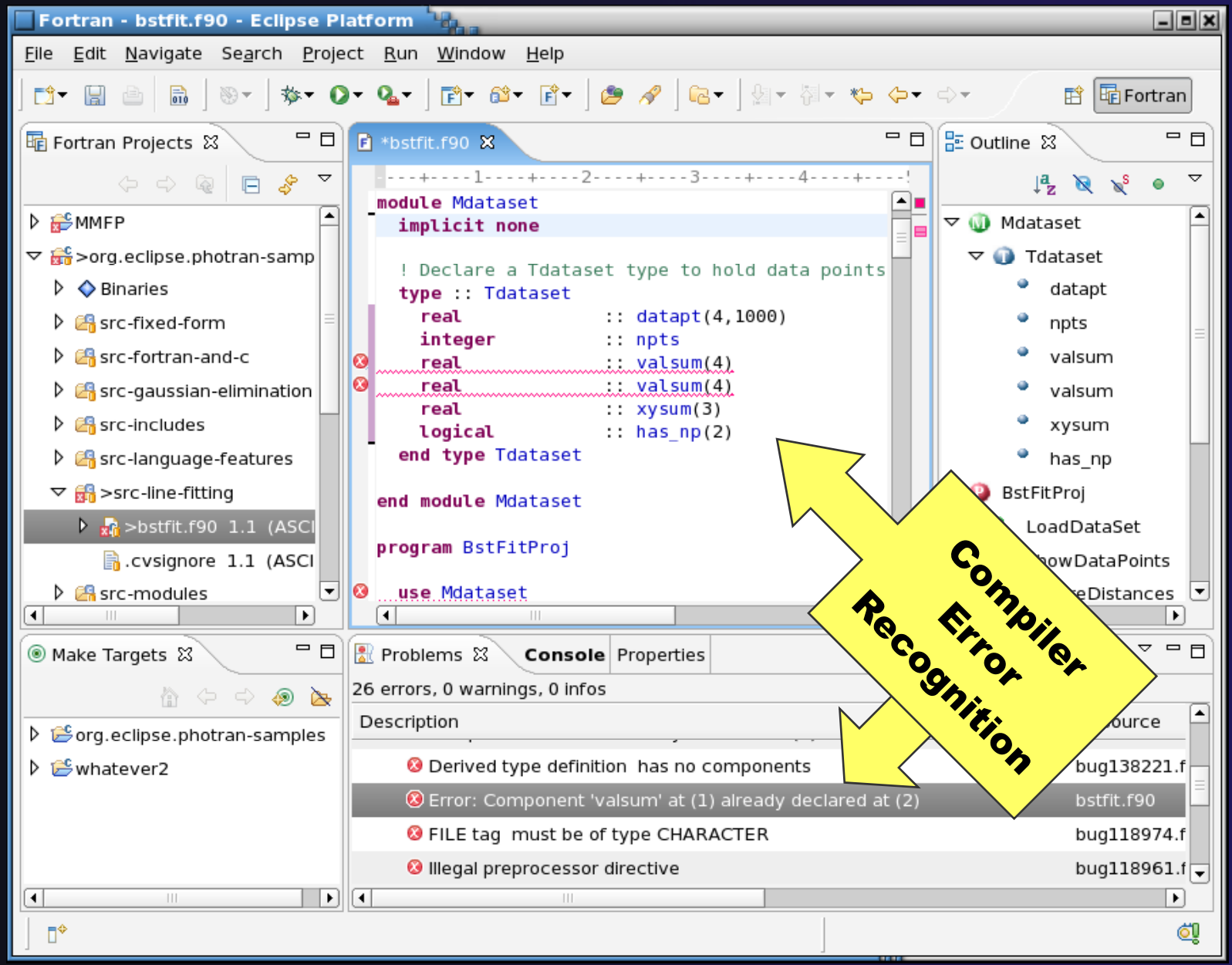
- Primary contributor: UIUC
- Contrib's from Intel, IBM, LANL, & others

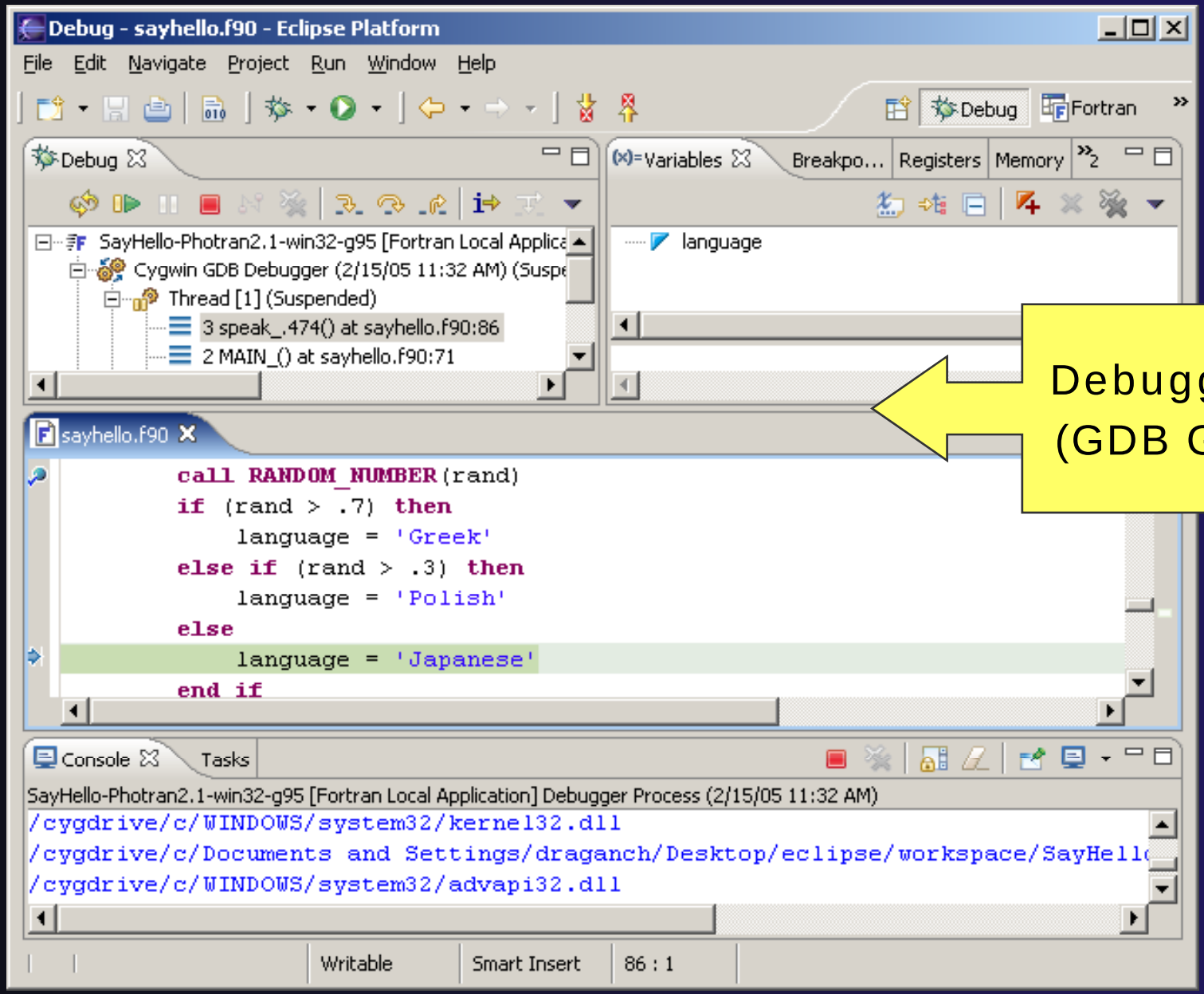



Fortran Editor & Outline


Fixed Form Support
hello.f
c Fixed format source with context-aware highlighting
integer :: if = 3
integer :: end = 5
character :: endif = "Hello"
if (if .gt. end) then
print *, &
&endif
endif
Context-Aware Highlighting


Fortran - bstfit.f90 - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Fortran
Fortran Projects
MMFP
>org.eclipse.photran
Binaries
src-fixed-form
src-fortran-and-c
src-gaussian-elimination
src-includes
src-language-features
>src-line-fitting
>bstfit.f90 1.1 (ASCI
.cvsignore 1.1 (ASCI
src-modules
CVS support
*bstfit.f90
hold data points
(4,1000)
real :: valsum(4)
real :: valsum(4)
real :: xysum(3)
logical :: has_np(2)
end type Tdataset
end module Mdataset
program BstFitProj
use Mdataset
Outline
Mdataset
Tdataset
datapt
npts
valsum
valsum
xysum
has_np
BstFitProj
LoadDataSet
ShowDataPoints
SquareDistances
Make Targets
org.eclipse.photran-samples
whatever2
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

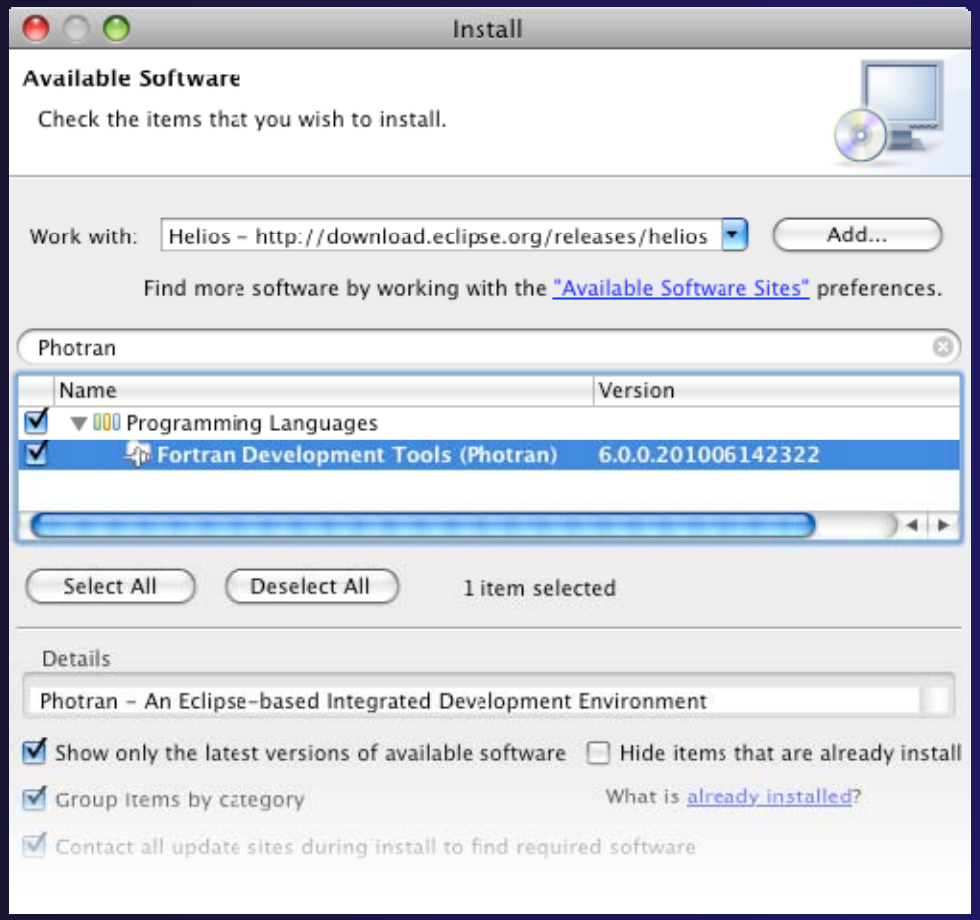

Fortran - bstfit.f90 - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Fortran
Fortran Projects
MMFP
>org.eclipse.photran-samp
Binaries
src-fixed-form
src-fortran-and-c
src-gaussian-elimination
src-includes
src-language-features
>src-line-fitting
>bstfit.f90 1.1 (ASCI
.cvsignore 1.1 (ASCI
src-modules
*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
Outline
Mdataset
Tdataset
datapt
npts
valsum
valsum
xysum
has_np
BstFitProj
LoadDataSet
owDataPoints
eDistances
Compiler Error Recognition
Make Targets
org.eclipse.photran-samples
whatever2
Problems
Console
Properties
26 errors, 0 warnings, 0 infos
Description
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


Debug - sayhello.f90 - Eclipse Platform
File Edit Navigate Project Run Window Help
Debug
Fortran
Debug
SayHello-Photran2.1-win32-g95 [Fortran Local Applica
Cygwin GDB Debugger (2/15/05 11:32 AM) (Suspe
Thread [1] (Suspended)
3 speak_.474() at sayhello.f90:86
2 MAIN_() at sayhello.f90:71
Variables
Breakpo...
Registers
Memory
language
sayhello.f90
call RANDOM_NUMBER(rand)
if (rand > .7) then
language = 'Greek'
else if (rand > .3) then
language = 'Polish'
else
language = 'Japanese'
end if
Console
Tasks
SayHello-Photran2.1-win32-g95 [Fortran Local Application] Debugger Process (2/15/05 11:32 AM)
/cygdrive/c/WINDOWS/system32/kernel32.dll
/cygdrive/c/Documents and Settings/draganch/Desktop/eclipse/workspace/SayHell
/cygdrive/c/WINDOWS/system32/advapi32.dll
Writable
Smart Insert
86 : 1
Debugging
(GDB GUI)

# Installing Photran

http://wiki.eclipse.org/PTP/photran/documentation/photran6#Installation_Procedure

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



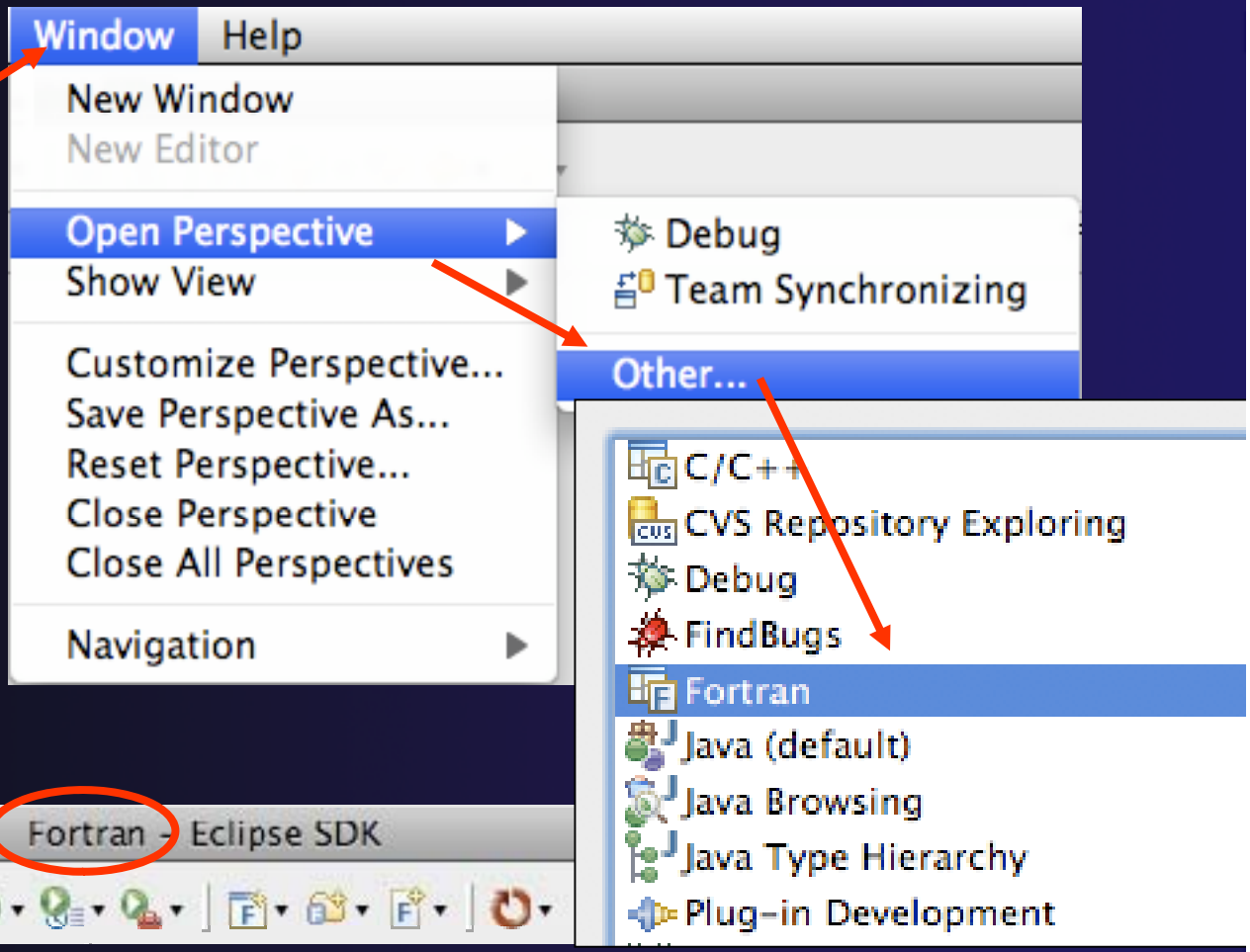
# Using Photran

- It's just like using CDT...
  - Similar New Project wizards
  - Similar build procedure
  - Similar launch/debug procedure
- ...but not exactly
  - Remote development not supported
  - Configuring fixed vs. free form file extensions
  - Different editor features
  - Different advanced features (Module 7)

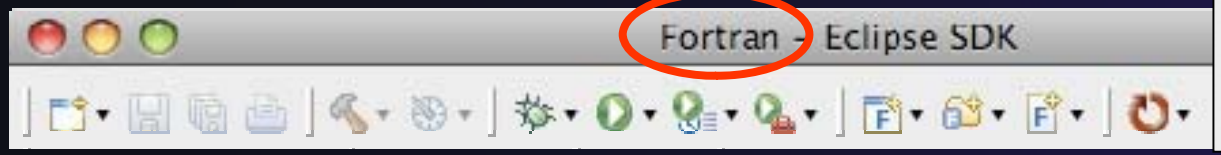
# Switch to Fortran C/C++ Perspective

(same as for C/C++)

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




- What Perspective am in in? See Title Bar

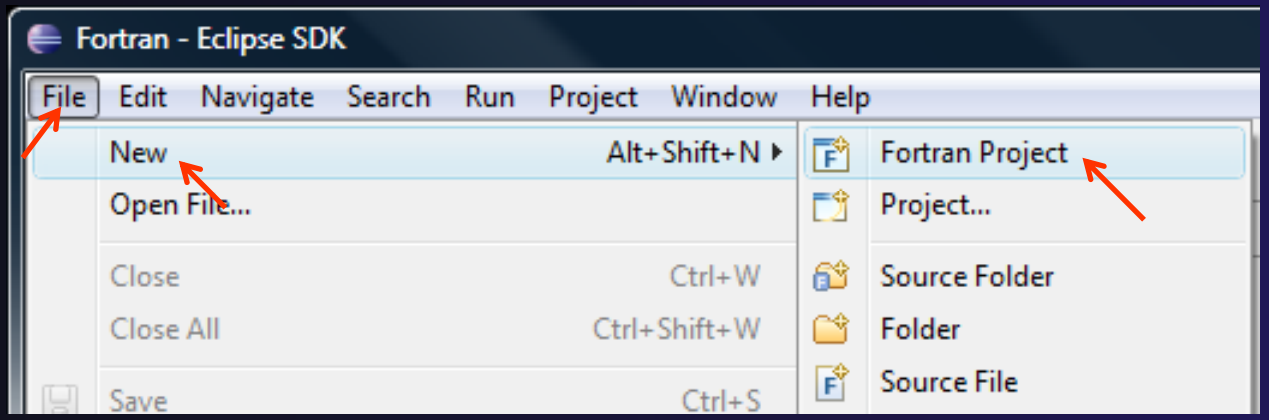


# Creating a Fortran Application

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

Steps:

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

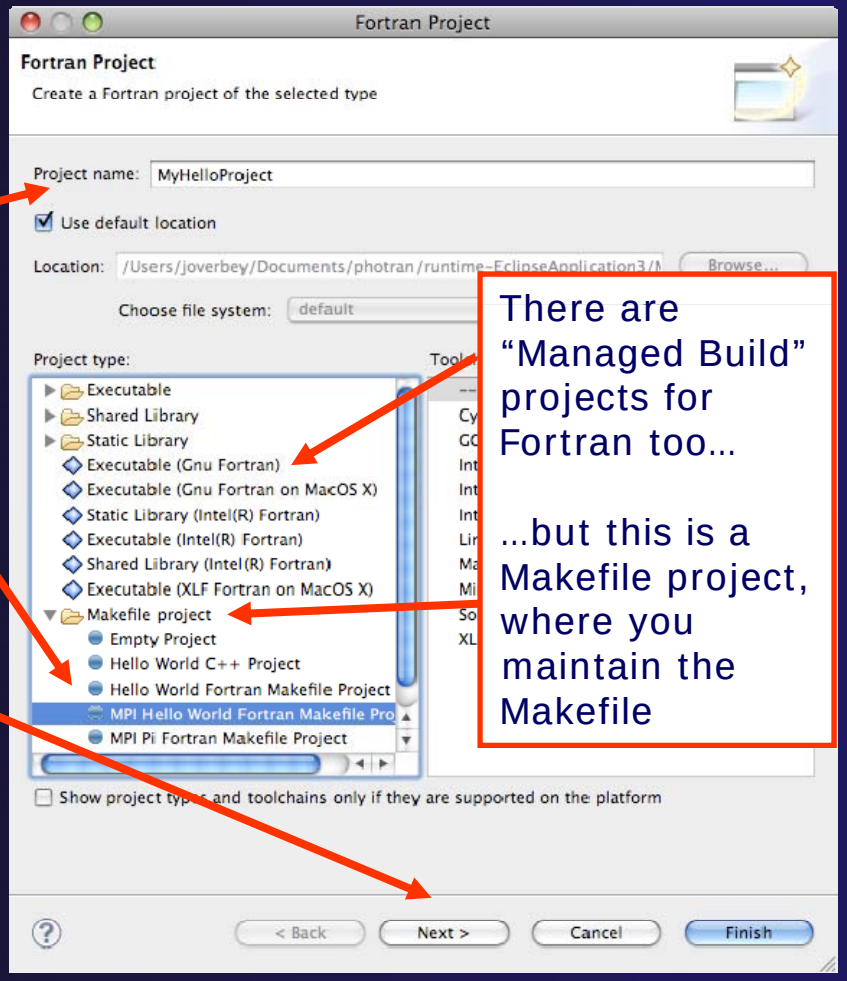


# New Fortran Project Wizard

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

Create a new MPI project

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



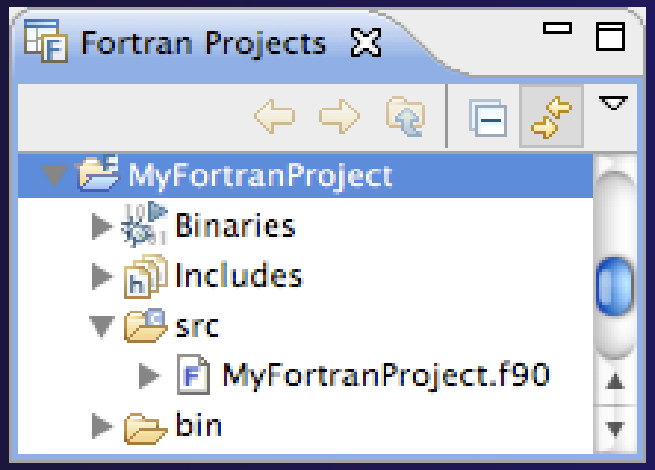

There are "Managed Build" projects for Fortran too…

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

# Fortran Projects View

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

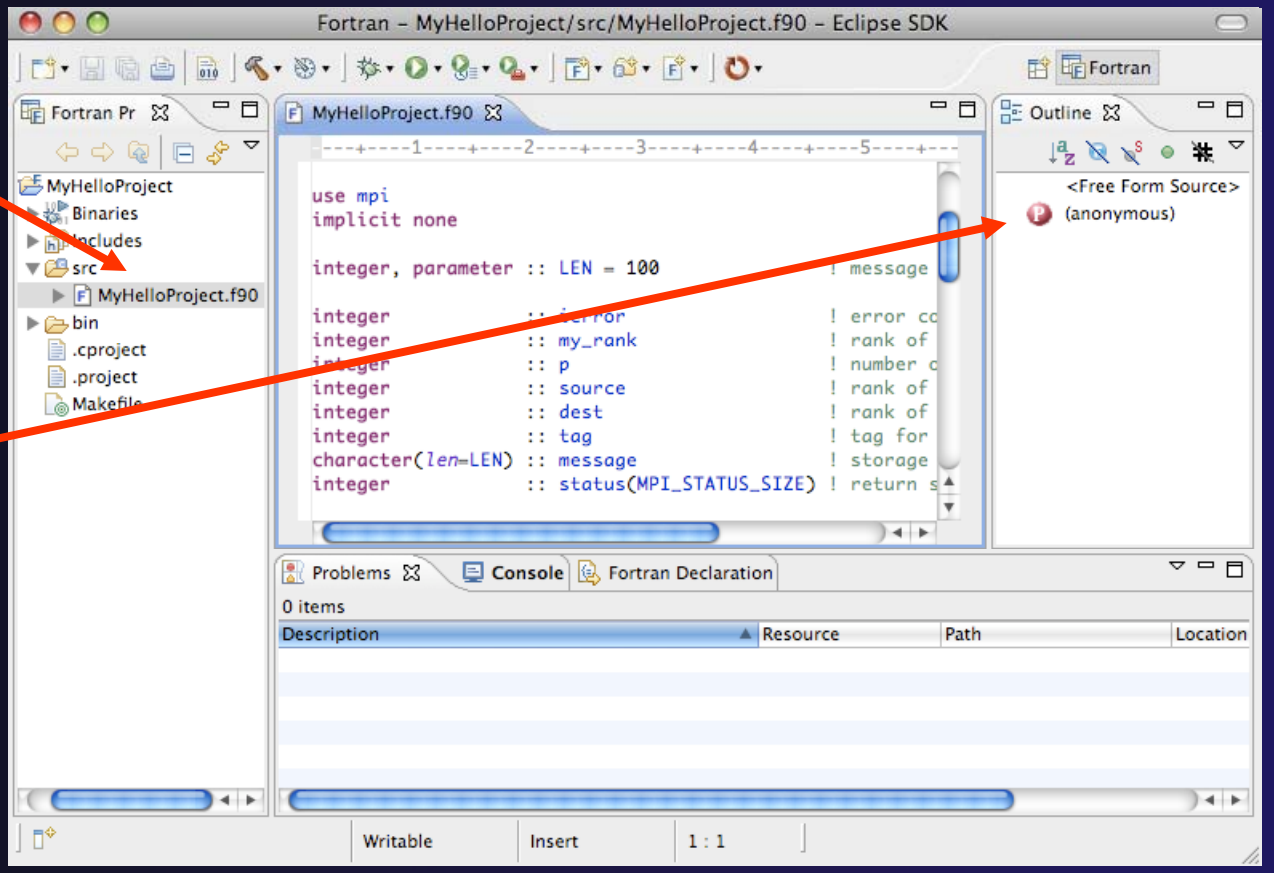
- Represents user's data
- It is a set of user defined resources
  - Files
  - Folders
  - Projects
    - Collections of files and folders
    - Plus meta-data
- Resources are visible in the Fortran Projects View



# Editor and Outline View

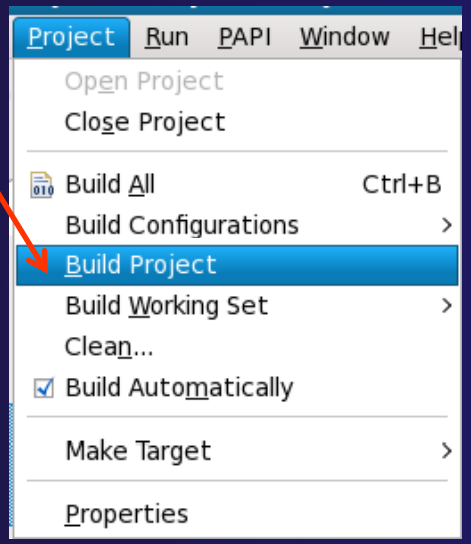
(similar to C/C++)

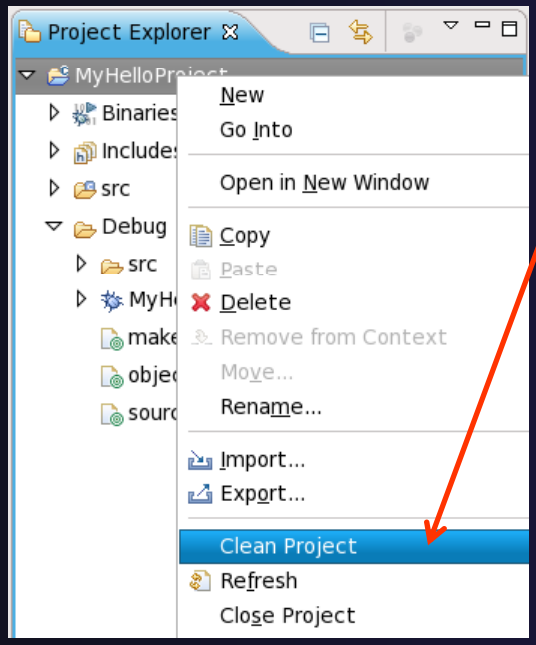
- Double-click on source file to open Fortran editor
- Outline view is shown for file in editor



# Build
(same as C/C++)

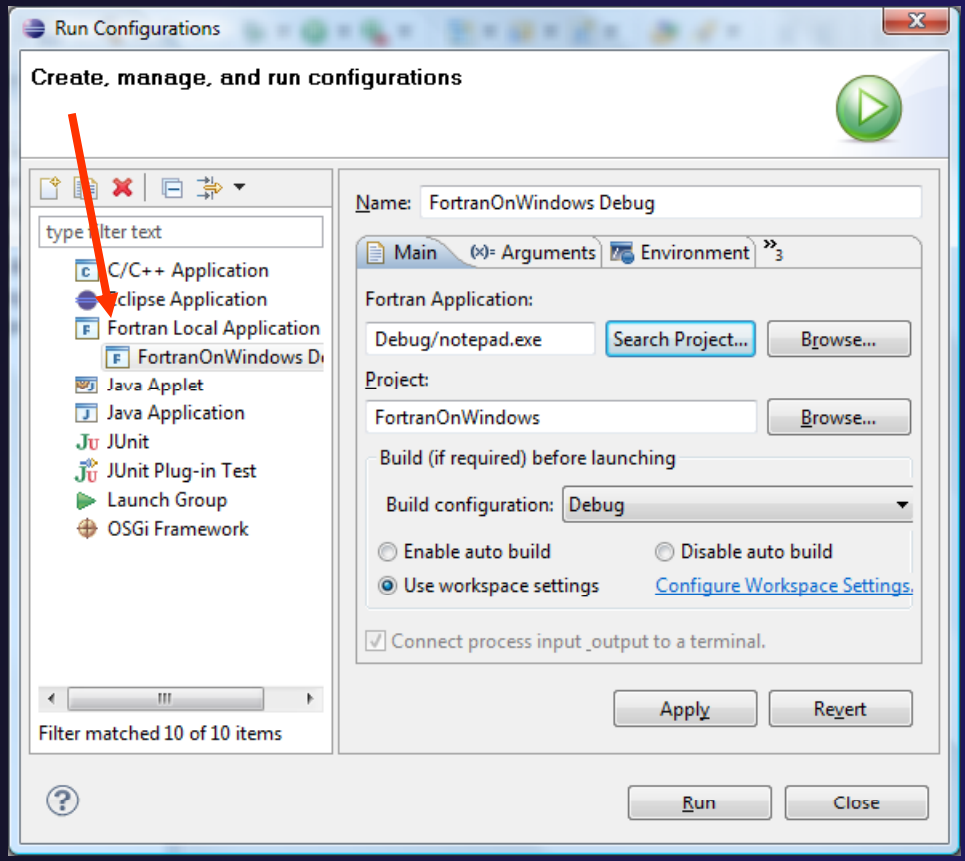
- Your program should build when created.
- To rebuild, many ways include:
  - Select project, Hit hammer icon in toolbar
  - Select project, **Project ▶ Build Project**
  - Right mouse on project, **Clean Project**



# Et Cetera

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



# Et Cetera

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

# Diagnosing Common Problems

(also true for C/C++)

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

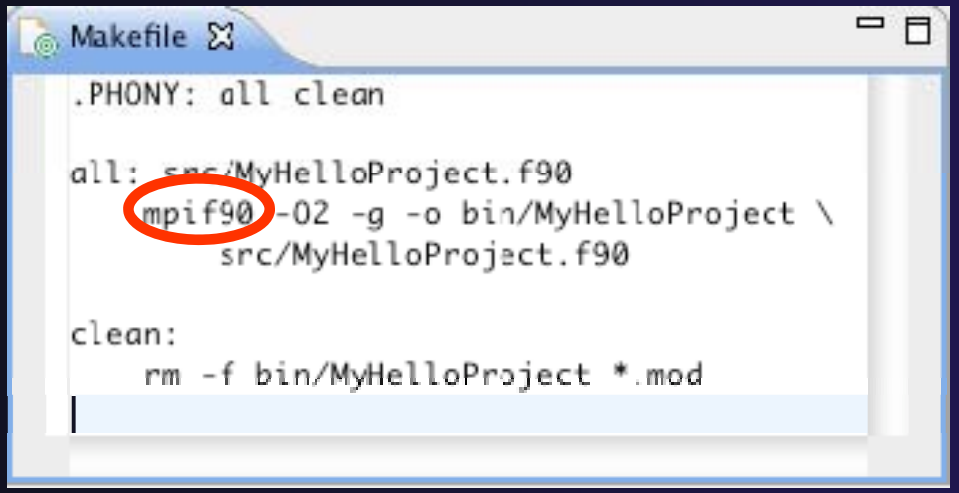
- Right-click on the project in the Fortran Projects view, and choose **Properties**
- Expand **Fortran Build▶Settings**
- Switch to the **Error Parsers** tab
- Are Photran's error parsers checked? If not, click **Check all**
- Click **OK** and re-build

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

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

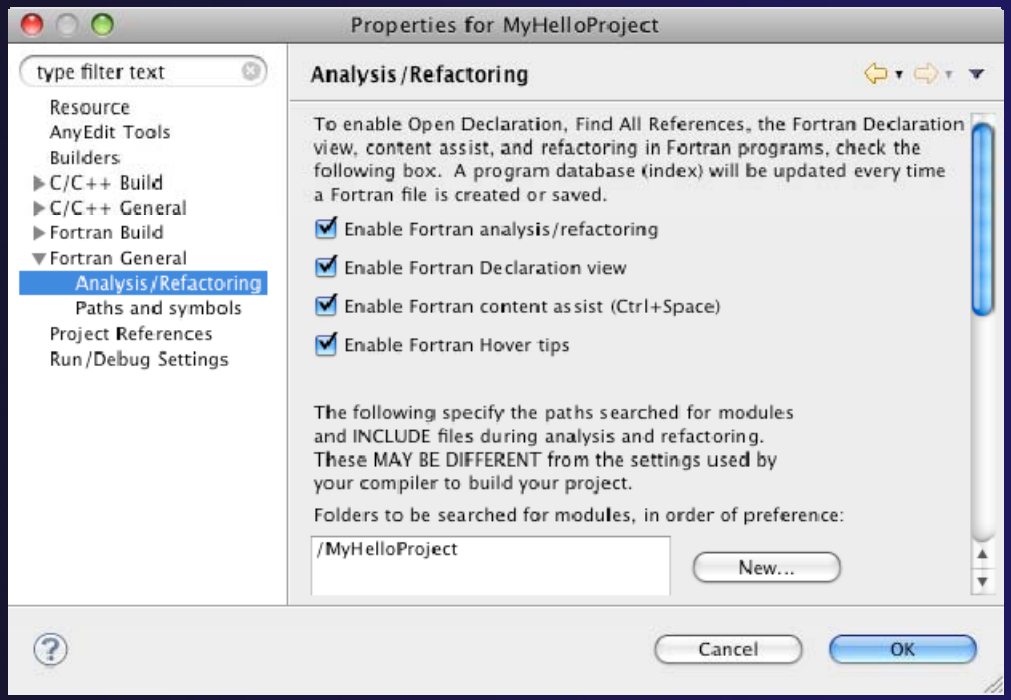
# Differences (1): MPI Project Wizard

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



# Differences (2): Content Assist

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



# Differences (3): Source Form

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

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

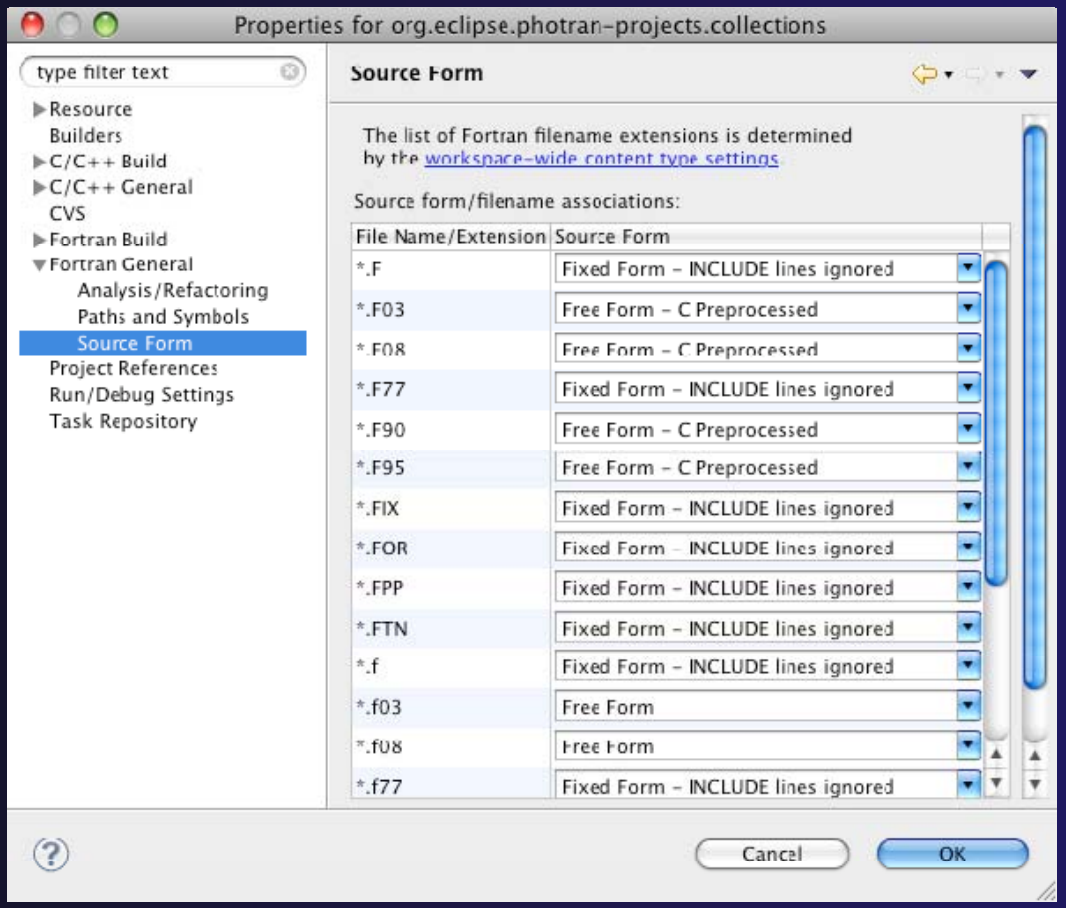
- Many features *will not work* if filename extensions are associated incorrectly

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

# Differences (3): Source Form

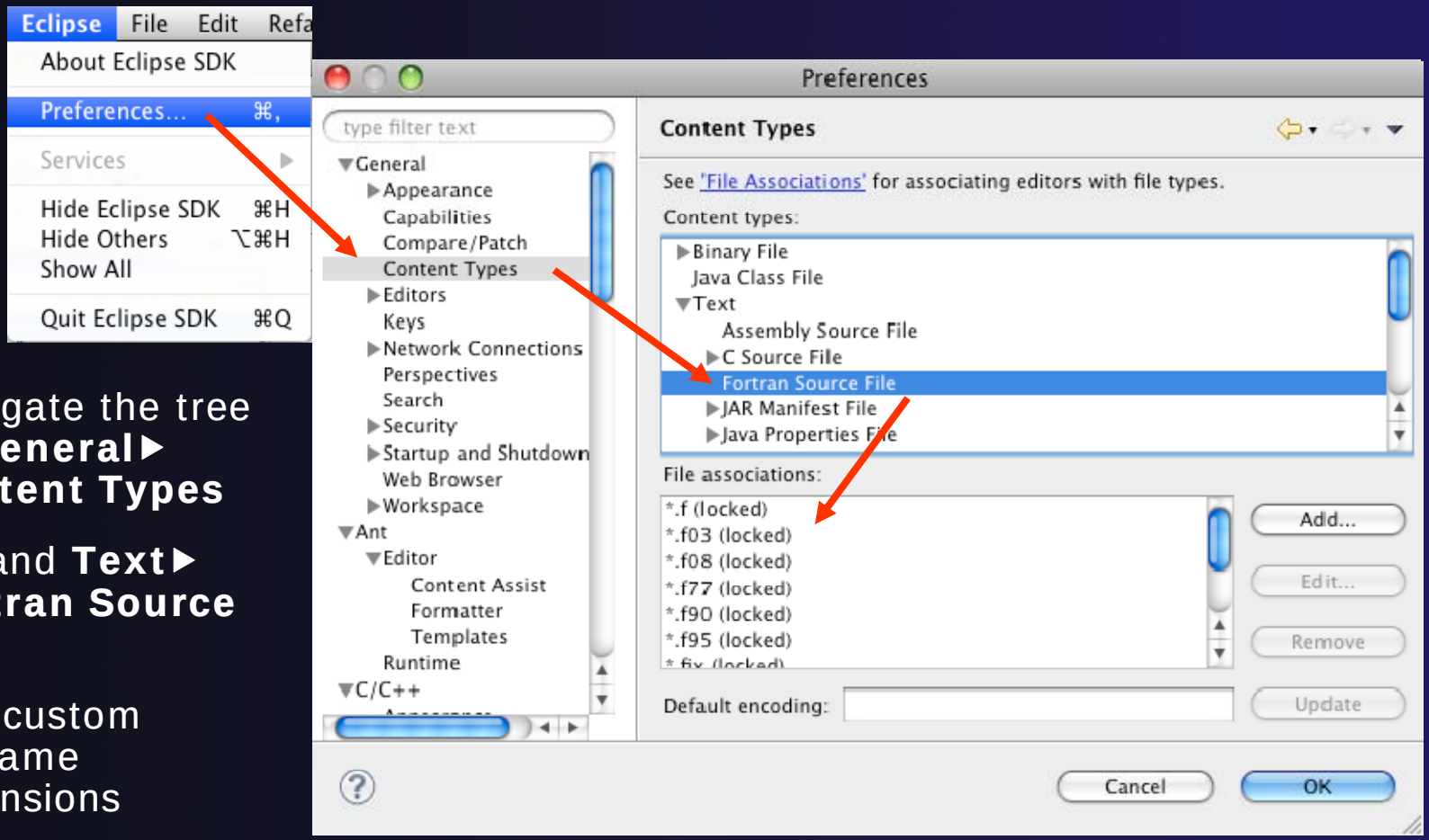
**Set free / fixed form associations in the project properties**

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



# Differences (3): Source Form

**Add new filename extensions in workspace preferences**




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

# Differences (4): Remote Support

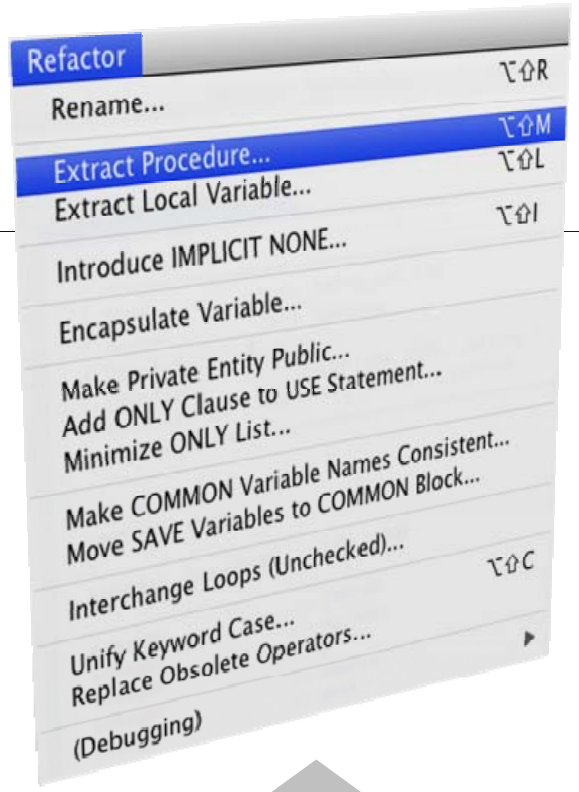
- Remote Fortran projects are not supported
  - Basic features will work
    (editor, Outline view, etc.)
  - Advanced features should not be enabled
    (content assist, search, refactoring, etc.)

# For More Information

- **Photran online documentation**
  linked from http://www.eclipse.org/photran
  - **User's Guide**
    General introduction, basic features
  - **Advanced Features Guide**
    Features requiring analysis/refactoring to be enabled
- **Online tutorial:** Compiling and running the Parallel Ocean Program using Photran and PTP
  linked from http://wiki.eclipse.org/PTP/photran/tutorials

# Refactoring

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

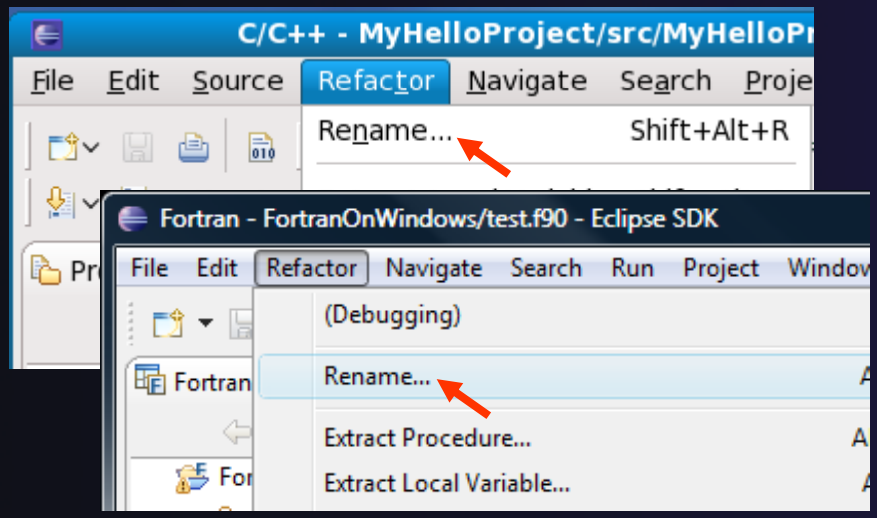



- Refactoring is the research motivation for Photran @ Illinois
  - Illinois is a leader in refactoring research
  - "Refactoring" was coined in our group (Opdyke & Johnson, 1990)
  - We had the first dissertation… (Opdyke, 1992)
  - …and built the first refactoring tool… (Roberts, Brant, & Johnson,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

# Rename Refactoring

(also available in C/C++)

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

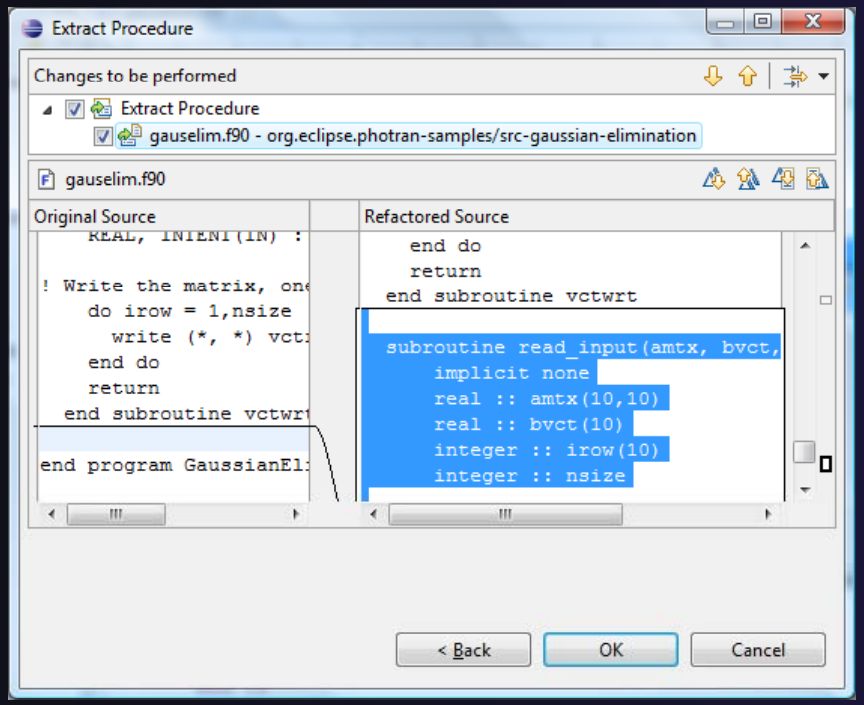



- Select **Fortran Perspective**
- Open a source file
- Click in editor view on declaration of a variable
- Select menu item **Refactor▶Rename**
  - Or use context menu
- Enter new name

# Extract Procedure Refactoring

(also available in C/C++ - “Extract Function”)

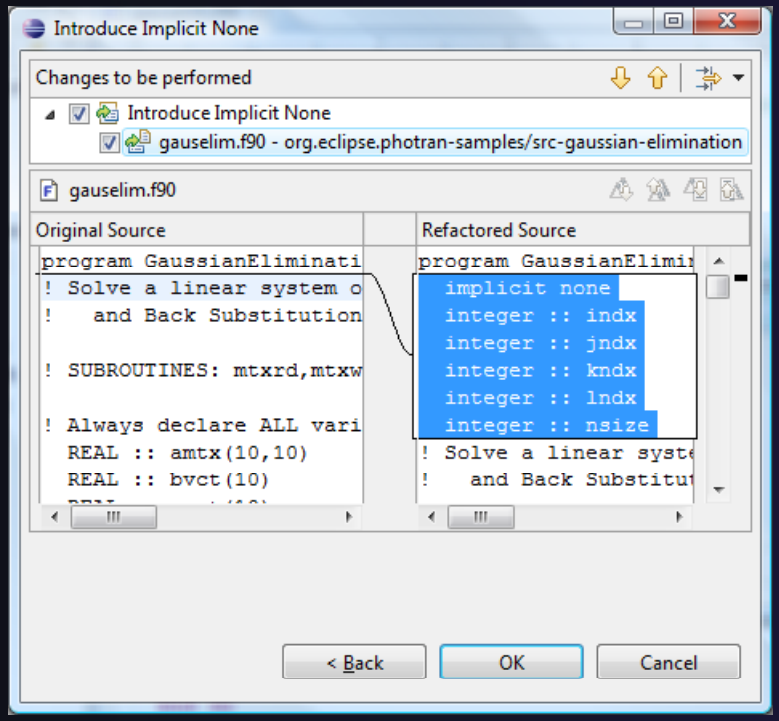
- Moves statements into a new subroutine, replacing the statements with a call to that subroutine
- Local variables are passed as arguments




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

# Introduce IMPLICIT NONE Refactoring

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




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