Open System for Earthquake Engineering Simulation
Pacific Earthquake Engineering Research Center

Discovering OpenSees: Surfing the Waves of OpenSees

Adding your Code to OpenSees

Frank McKenna
fmckenna@ce.berkeley.edu

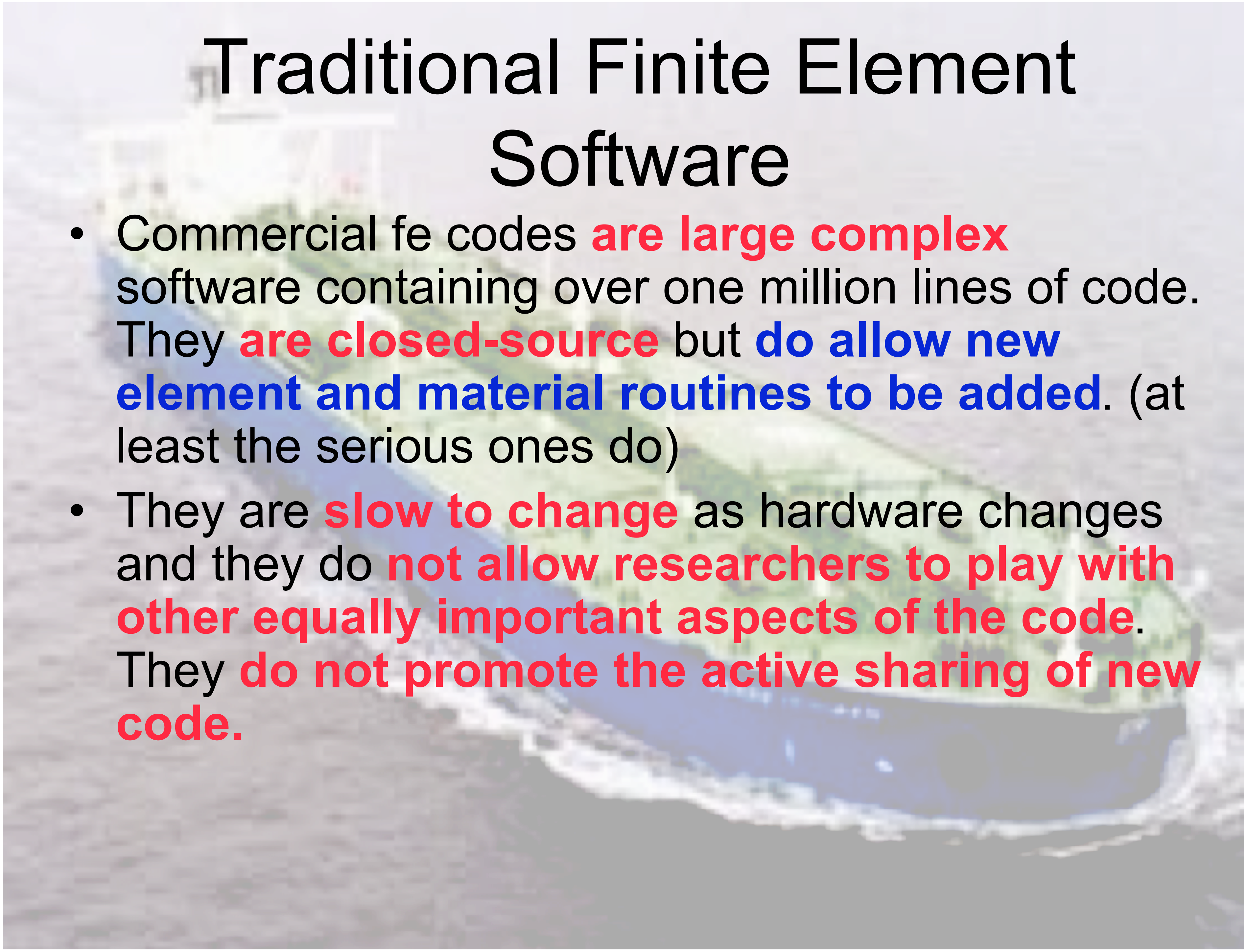
http://opensees.berkeley.edu/AddingYourCode.pdf

Sponsored by:
NEES through NEEScomm
Pacific Earthquake Engineering Research Center
National Science Foundation

Outline

- Introduction
- Adding a New Material to OpenSees.exe
- Adding a New Integrator to OpenSees.exe
- Summary & Conclusions

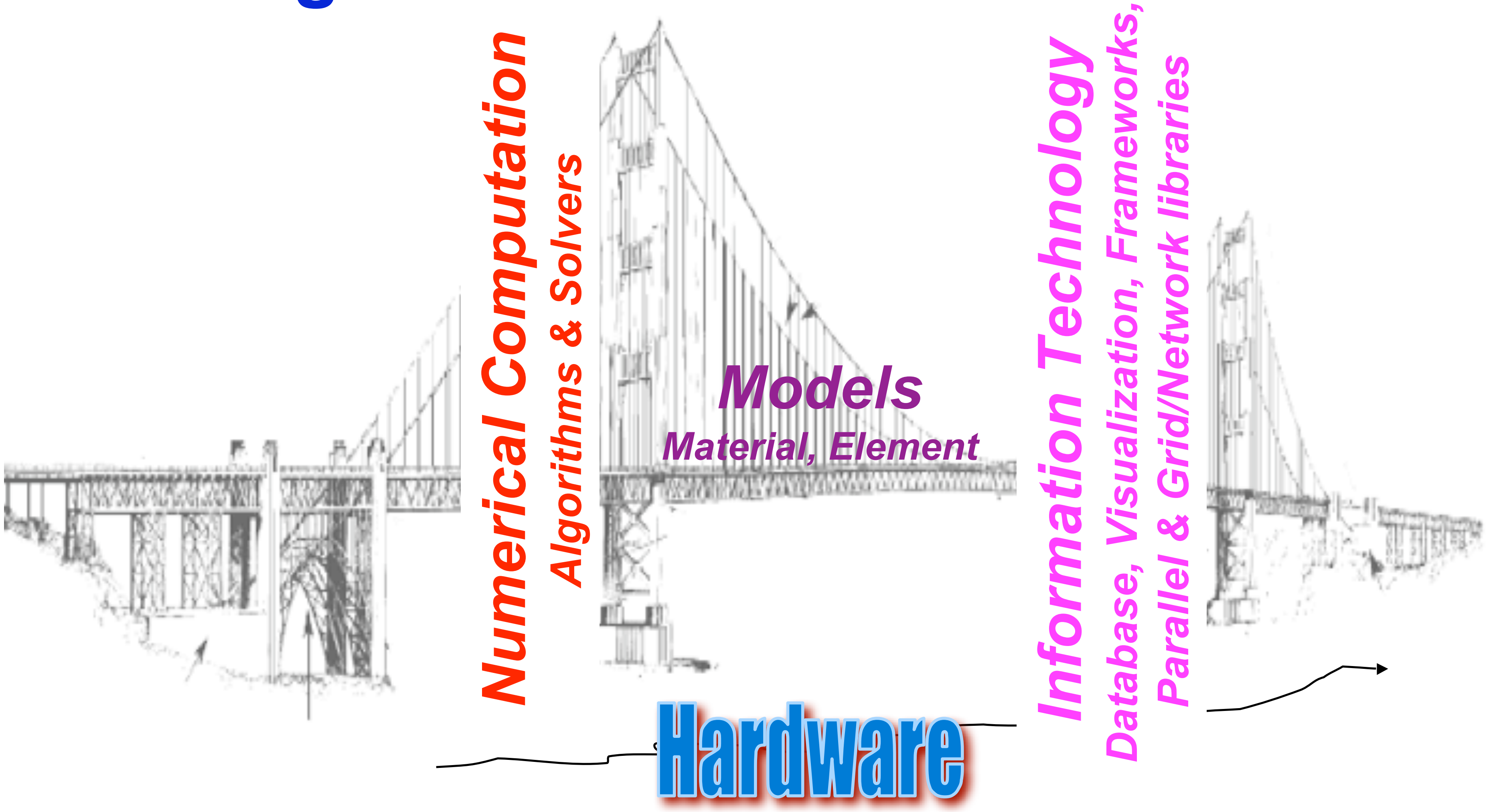
NOTE: I HOPE NOT TO GET BOGGED DOWN IN C++ ISSUES .. THIS IS NOT A WEBINAR ON HOW TO PROGRAM!



Traditional Finite Element Software

- Commercial fe codes **are large complex** software containing over one million lines of code. They **are closed-source** but **do allow new element and material routines to be added**. (at least the serious ones do)
- They are **slow to change** as hardware changes and they do **not allow researchers to play with other equally important aspects of the code**. They **do not promote the active sharing of new code.**

Building Blocks for Modern Simulation



OpenSees Goals

•To use **modern software techniques** to evolve an **extensible** finite element software platform for earthquake engineering that would encompass both **structural & geotechnical engineering** and be able to change with the **rapidly changing hardware resources**.

•To provide a common analytical research framework for **researchers to educate students & share new knowledge**.

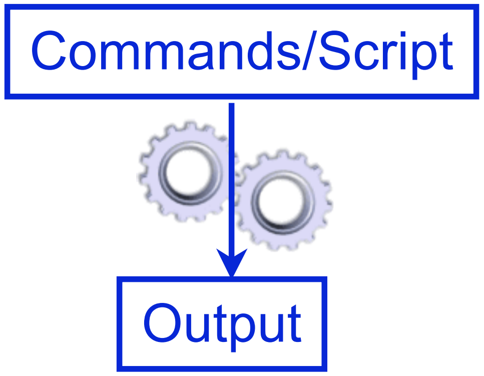
•To foster a mechanism whereby new research could be **disseminated quickly to industry for testing and adoption**.

OpenSees Classes

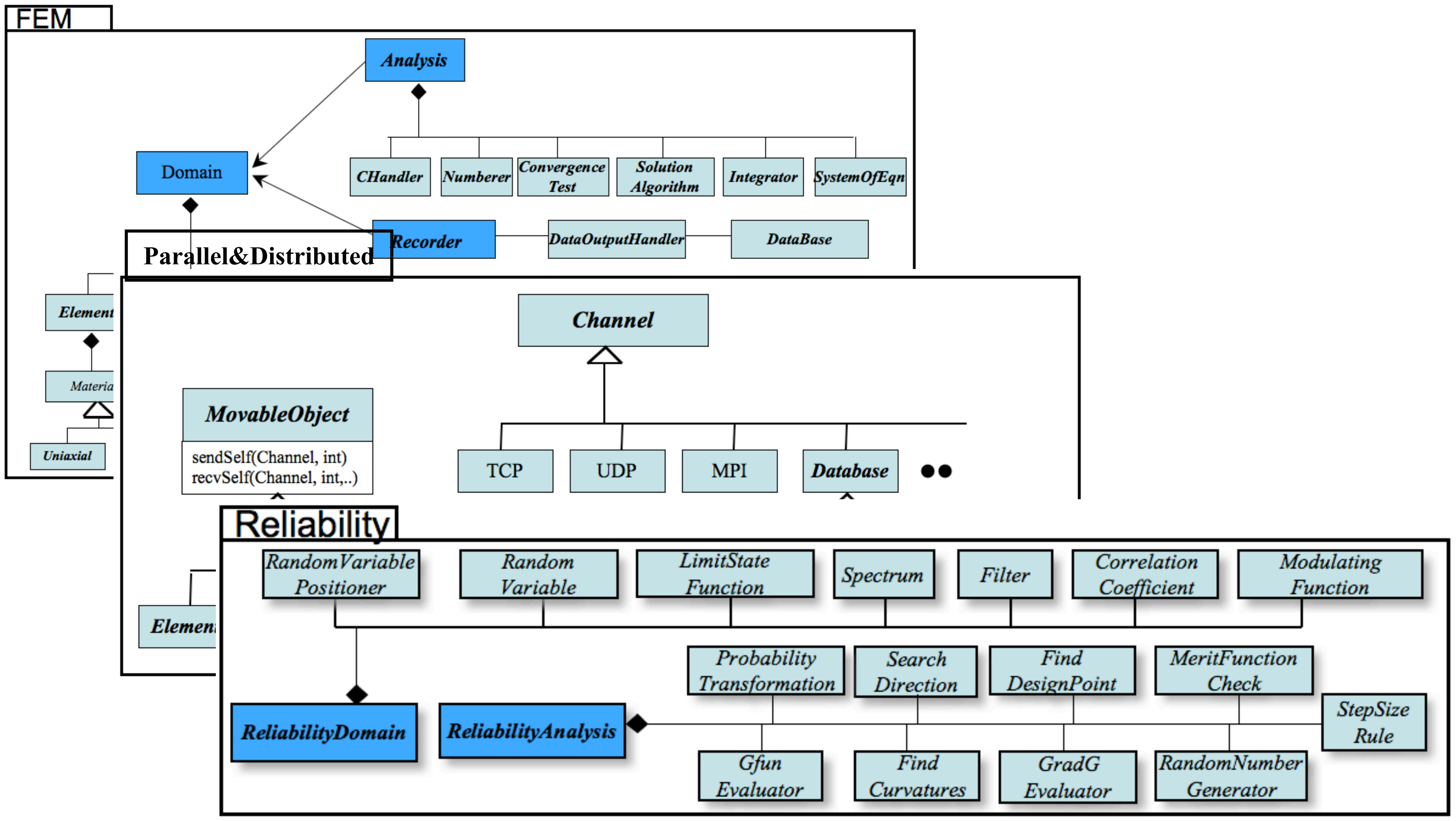
•To achieve these goals we developed an **open-source framework** for the development of **sequential and parallel** finite element applications. This framework we call **OpenSees**.

•The framework contains many classes provided by ourselves and **many many others** that allows you to build finite element applications.

One example of which is the interpreted finite element application, **OpenSees.exe**.



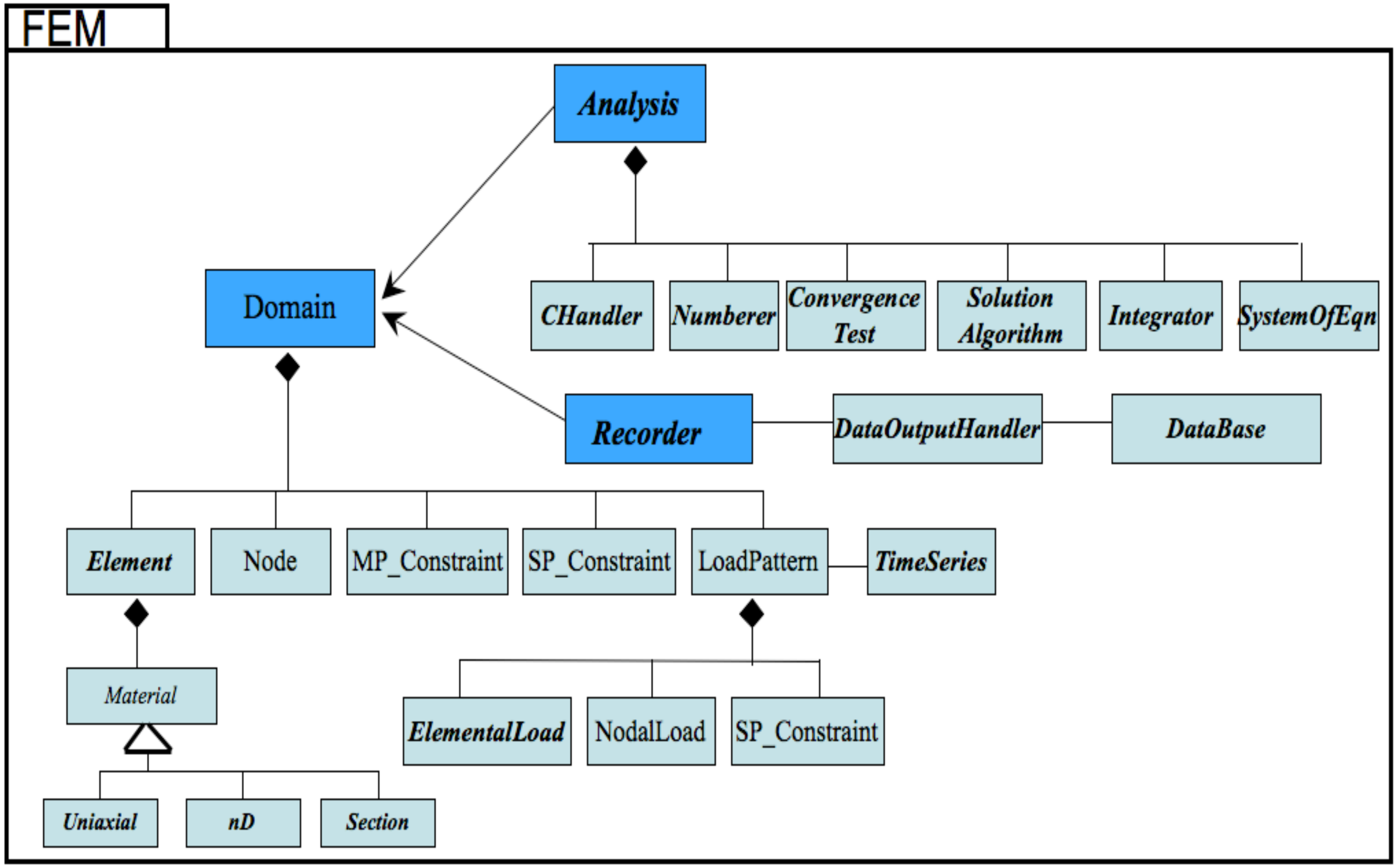
OpenSees Abstract Classes



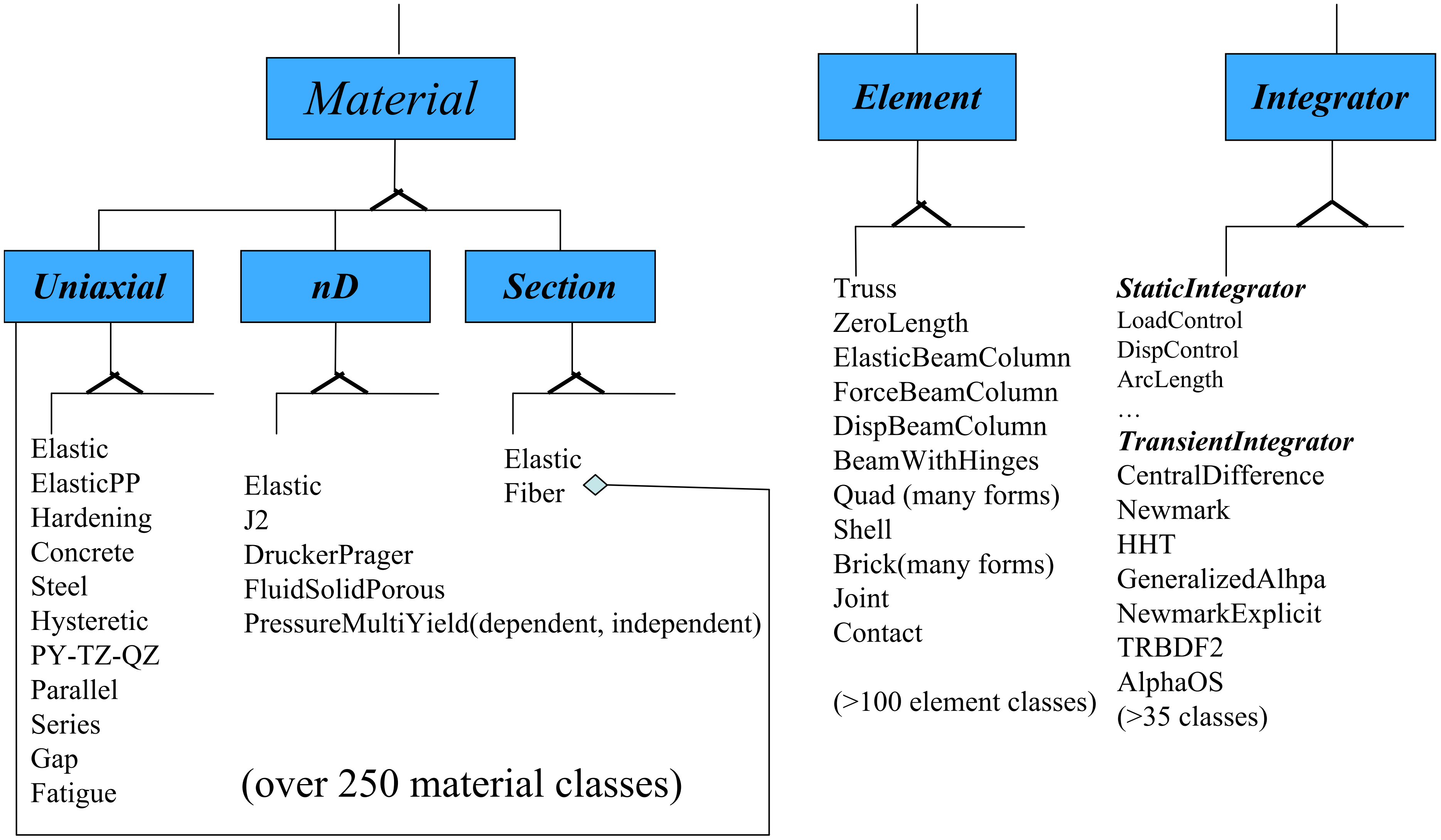
Currently over 1000 classes (modules)!

THE **ADVANTAGE** OF A **SOFTWARE FRAMEWORK** SUCH AS OPENSEES IS THAT **YOU DON'T HAVE TO UNDERSTAND ALL OF IT** TO BUILD APPLICATIONS OR MAKE CONTRIBUTIONS **YOU JUST NEED TO UNDERSTAND THAT PART THAT CONCERNS YOU**

OPENSEES FEM ABSTRACT CLASSES

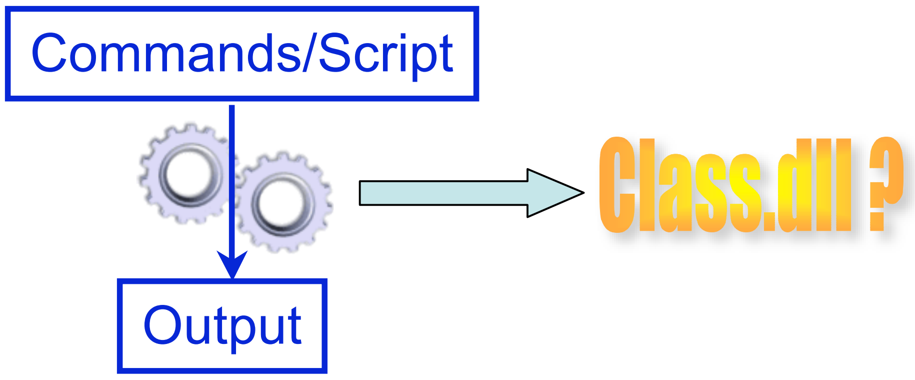


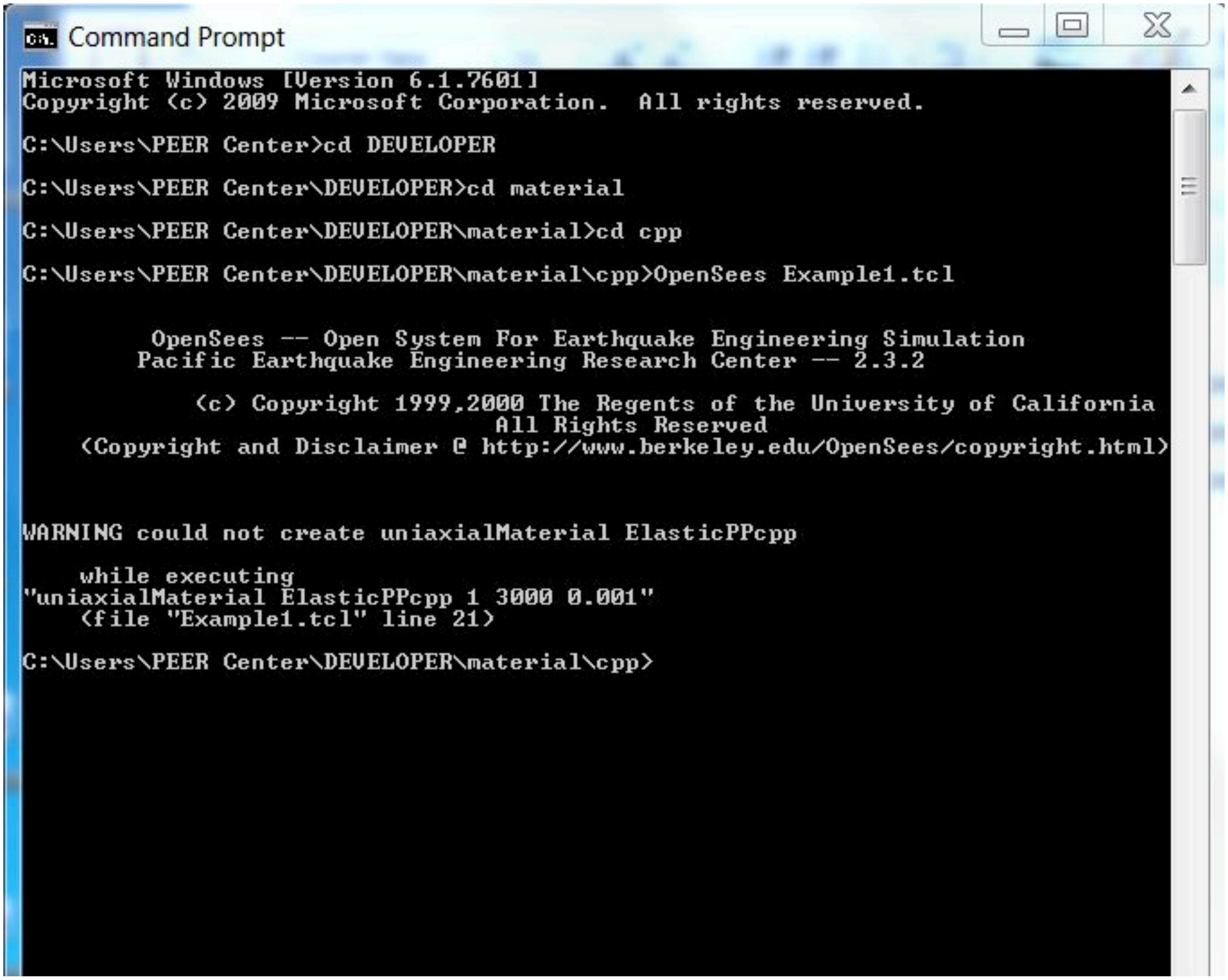
Framework Contains both Abstract Classes & Concrete Subclasses



Unknown Class Type

When OpenSees.exe is running and it comes across a class type it knows nothing about **BEFORE GIVING AN ERROR IT WILL TRY AND LOAD A LIBRARY OF THAT CLASSES NAME**. If it cannot find a **library of the appropriate name** or the **procedure of the appropriate name** in the library it will **FAIL**.





```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\PEER Center>cd DEVELOPER

C:\Users\PEER Center\DEVELOPER>cd material

C:\Users\PEER Center\DEVELOPER\material>cd cpp

C:\Users\PEER Center\DEVELOPER\material\cpp>OpenSees Example1.tcl


         OpenSees -- Open System For Earthquake Engineering Simulation
        Pacific Earthquake Engineering Research Center -- 2.3.2

            (c) Copyright 1999,2000 The Regents of the University of California
                                 All Rights Reserved
    (Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)



WARNING could not create uniaxialMaterial ElasticPPcpp

    while executing
"uniaxialMaterial ElasticPPcpp 1 3000 0.001"
    (file "Example1.tcl" line 21)

C:\Users\PEER Center\DEVELOPER\material\cpp>
```

To Add a New Class you must:

1) Provide Code that meets the Interface of the appropriate super-class
2) you must BUILD THE LIBRARY
3) make it accessible to the program.

WHILE C++ IS THE GLUE LANGUAGE THAT HOLDS OPENSEES TOGETHER

YOUR CODE DOES NOT HAVE TO BE WRITTEN IN C++.

C and FORTRAN OPTIONS ARE ALSO AVAILABLE

UniaxialMaterial Interface

```
class UniaxialMaterial : public Material
{
  public:
    UniaxialMaterial(int tag, int classTag);
    virtual ~UniaxialMaterial();

    virtual int setTrialStrain(double strain, double strainRate = 0.0) = 0;
    virtual double getStrain(void) = 0;
    virtual double getStress(void) = 0;
    virtual double getTangent(void) = 0;
    virtual double getInitialtangent(void) =0;

    virtual int commitState(void) = 0;
    virtual int revertToLastCommit(void) = 0;
    virtual int revertToStart(void) = 0;

    virtual UniaxialMaterial *getCopy(void) = 0;

    virtual Response *setResponse(const char **argv, int argc,OPS_Stream &theOutput);
    virtual int getResponse(int responseID, Information &info);
    virtual void Print(OPS_Stream &s, int flag =0);

     virtual int sendSelf(int commitTag, Channel &theChannel)=0;
     virtual int recvSelf(int commitTag, Chanel &theChannel, FEM_ObjectBroker &theBroker)=0;
    // THERE ARE SOME OTHERS .. BUT THESE ARE PURE VIRTUAL ONES THAT MUST BE PROVIDED
  protected:
  private:
};
```

Must be overridden by subclass, "pure virtual"

Can be overridden by subclass

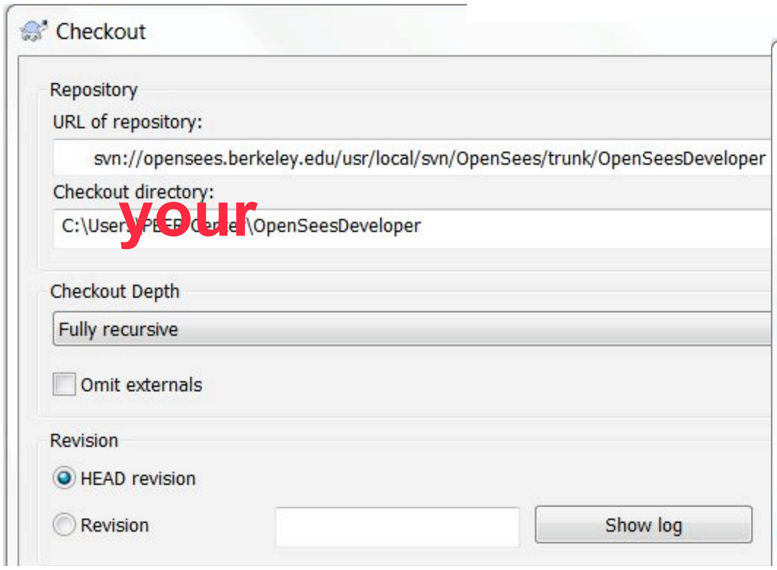
Adding New Code

For those new to Programming **NEVER EVER NEVER START WITH AN EMPTY FILE** .. TAKE SOMETHING SIMILAR THAT WORKS AND MAKE CHANGES TO THAT FILE.

We provide C++, C and Fortran examples on-line using svn

svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk/DEVELOPER

TortoiseSVN for Windows Users



Checkout

Repository

URL of repository: svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk/OpenSeesDeveloper

Checkout directory: C:\Users\PEER Center\OpenSeesDeveloper

your

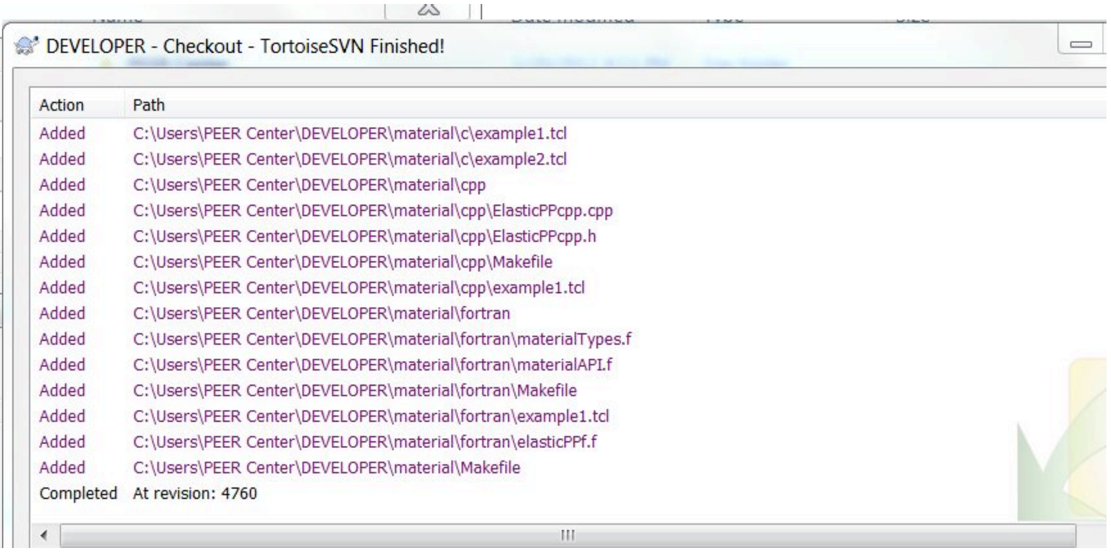
Checkout Depth: Fully recursive

[ ] Omit externals

Revision

(●) HEAD revision

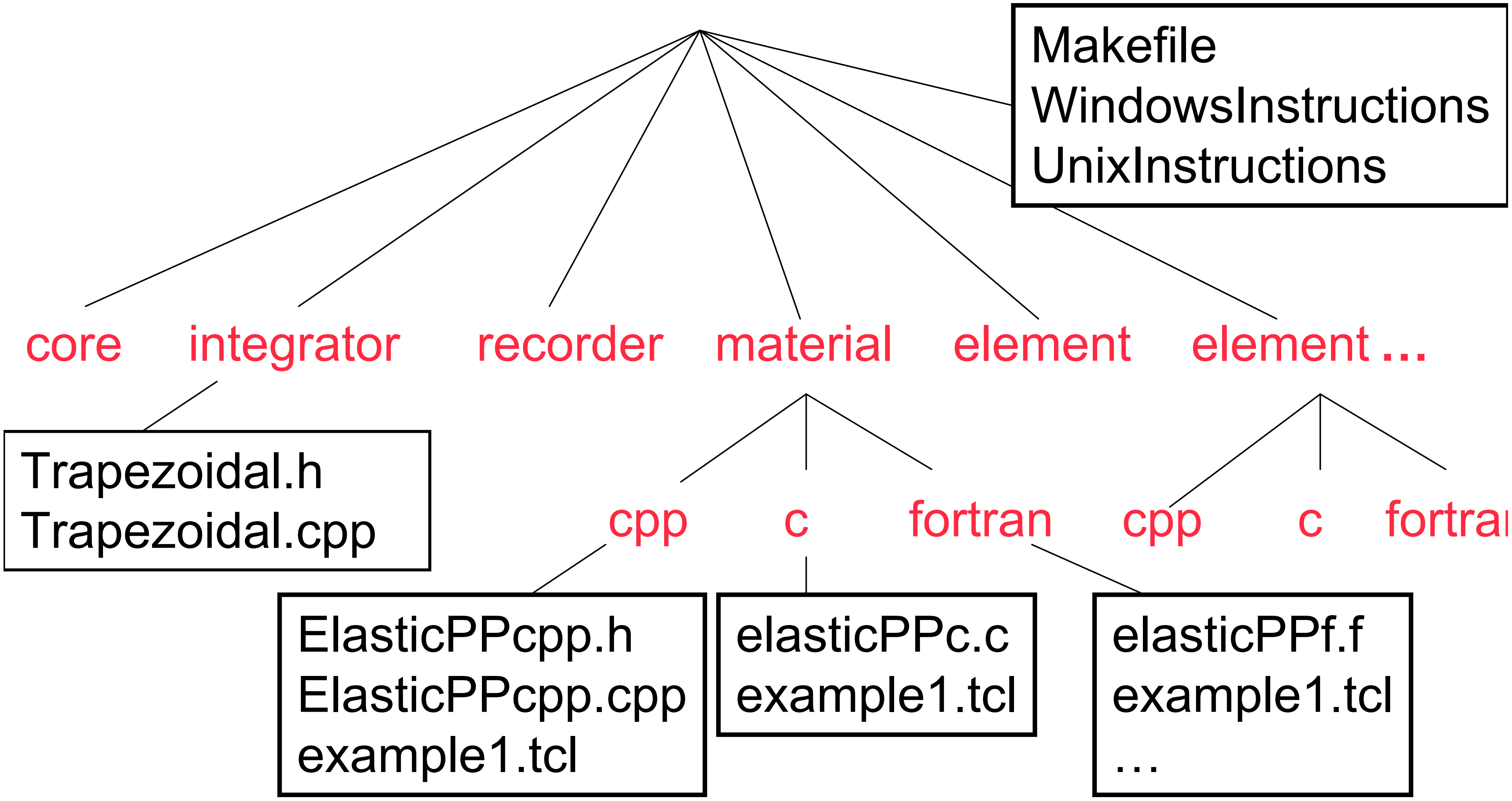
( ) Revision

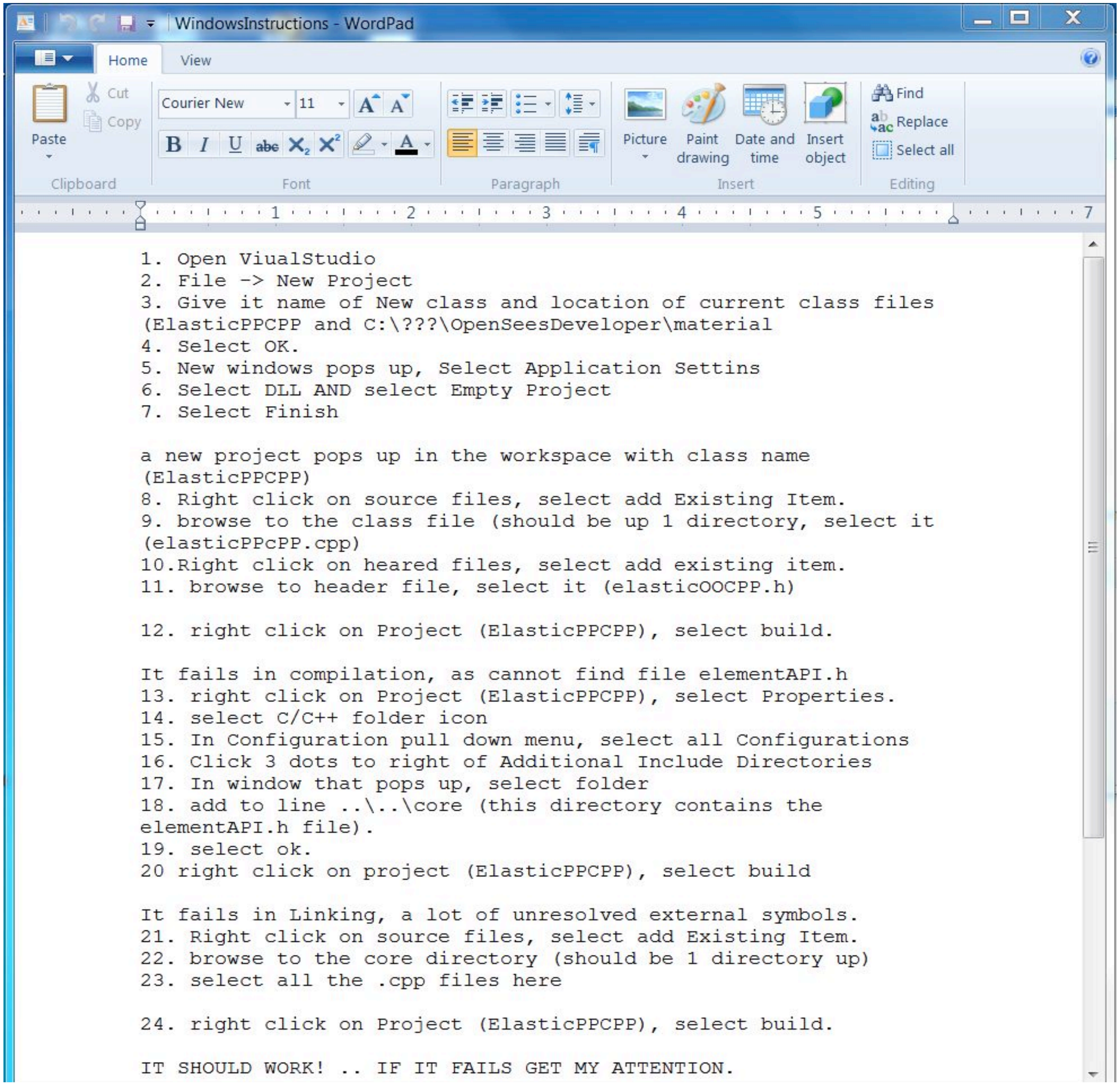


DEVELOPER - Checkout - TortoiseSVN Finished!

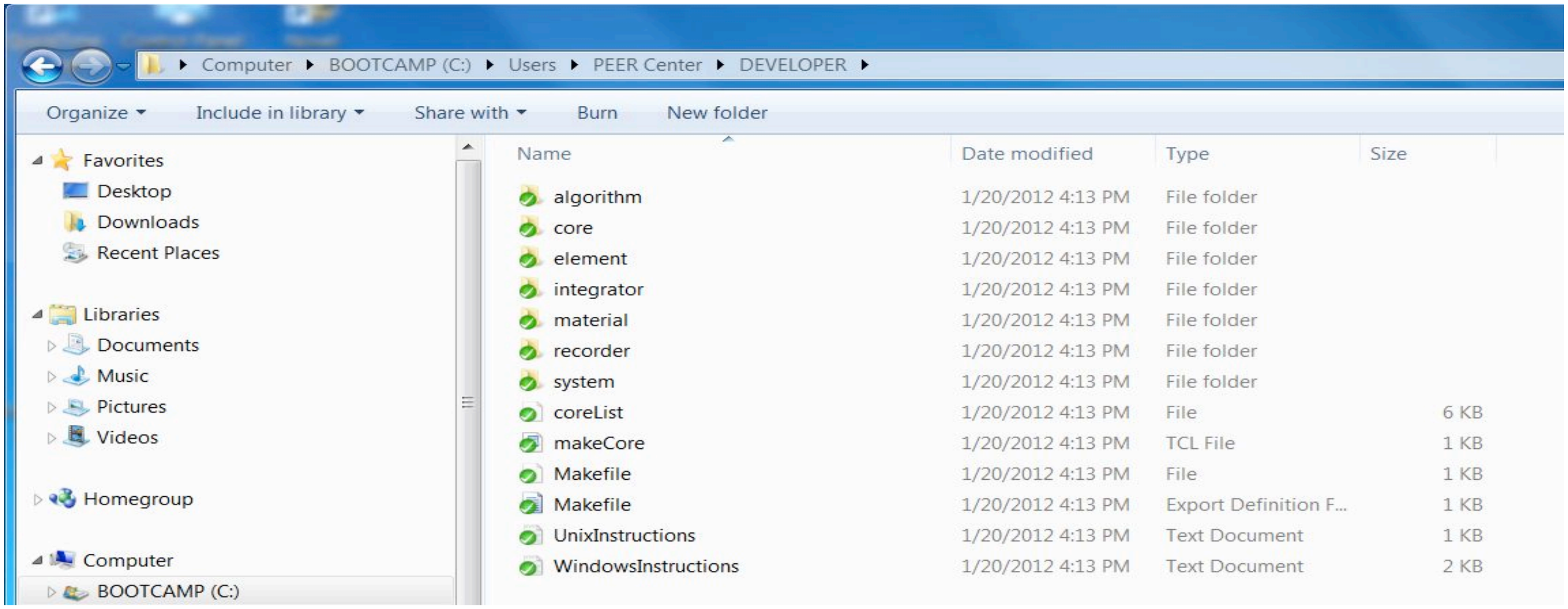
```
Action     Path
Added      C:\Users\PEER Center\DEVELOPER\material\c\example1.tcl
Added      C:\Users\PEER Center\DEVELOPER\material\c\example2.tcl
Added      C:\Users\PEER Center\DEVELOPER\material\cpp
Added      C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp.cpp
Added      C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp.h
Added      C:\Users\PEER Center\DEVELOPER\material\cpp\Makefile
Added      C:\Users\PEER Center\DEVELOPER\material\cpp\example1.tcl
Added      C:\Users\PEER Center\DEVELOPER\material\fortran
Added      C:\Users\PEER Center\DEVELOPER\material\fortran\materialTypes.f
Added      C:\Users\PEER Center\DEVELOPER\material\fortran\materialAPI.f
Added      C:\Users\PEER Center\DEVELOPER\material\fortran\Makefile
Added      C:\Users\PEER Center\DEVELOPER\material\fortran\example1.tcl
Added      C:\Users\PEER Center\DEVELOPER\material\fortran\elasticPPf.f
Added      C:\Users\PEER Center\DEVELOPER\material\Makefile
Completed  At revision: 4760
```

Source Code Tree in DEVELOPER

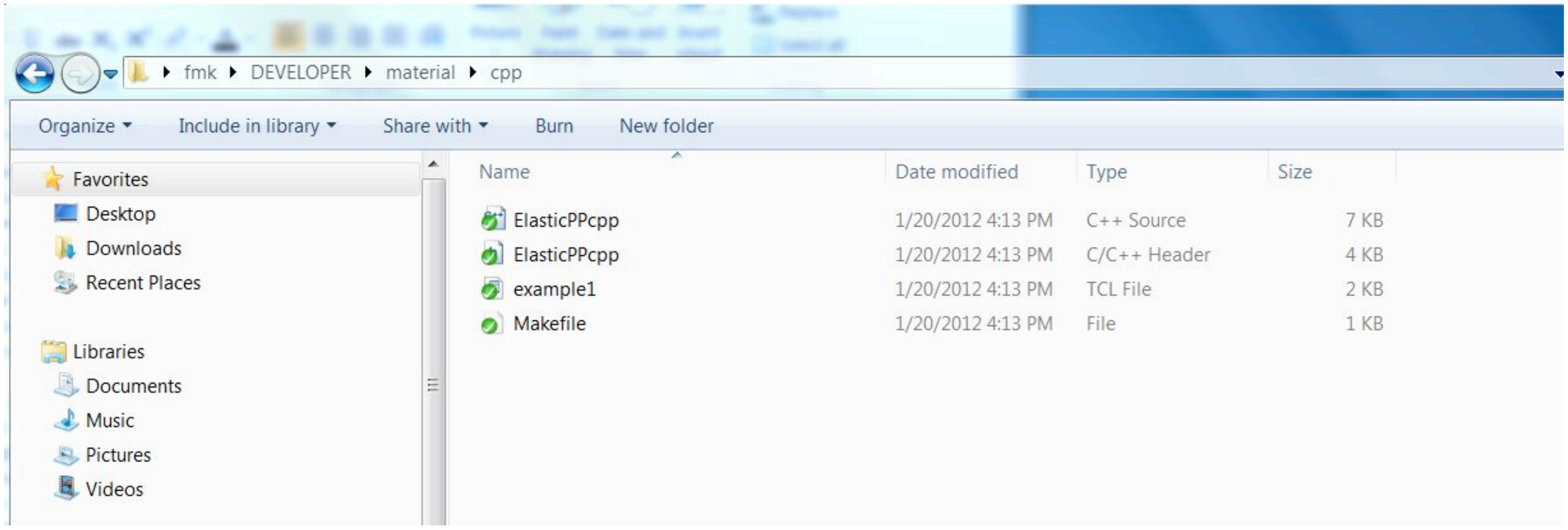




```
1. Open ViualStudio
2. File -> New Project
3. Give it name of New class and location of current class files
(ElasticPPCPP and C:\???\OpenSeesDeveloper\material
4. Select OK.
5. New windows pops up, Select Application Settins
6. Select DLL AND select Empty Project
7. Select Finish

a new project pops up in the workspace with class name
(ElasticPPCPP)
8. Right click on source files, select add Existing Item.
9. browse to the class file (should be up 1 directory, select it
(elasticPPcPP.cpp)
10.Right click on heared files, select add existing item.
11. browse to header file, select it (elasticOOCPP.h)

12. right click on Project (ElasticPPCPP), select build.

It fails in compilation, as cannot find file elementAPI.h
13. right click on Project (ElasticPPCPP), select Properties.
14. select C/C++ folder icon
15. In Configuration pull down menu, select all Configurations
16. Click 3 dots to right of Additional Include Directories
17. In window that pops up, select folder
18. add to line ..\..\core (this directory contains the
elementAPI.h file).
19. select ok.
20 right click on project (ElasticPPCPP), select build

It fails in Linking, a lot of unresolved external symbols.
21. Right click on source files, select add Existing Item.
22. browse to the core directory (should be 1 directory up)
23. select all the .cpp files here

24. right click on Project (ElasticPPCPP), select build.

IT SHOULD WORK! .. IF IT FAILS GET MY ATTENTION.
```



Name	Date modified	Type	Size
algorithm	1/20/2012 4:13 PM	File folder	
core	1/20/2012 4:13 PM	File folder	
element	1/20/2012 4:13 PM	File folder	
integrator	1/20/2012 4:13 PM	File folder	
material	1/20/2012 4:13 PM	File folder	
recorder	1/20/2012 4:13 PM	File folder	
system	1/20/2012 4:13 PM	File folder	
coreList	1/20/2012 4:13 PM	File	6 KB
makeCore	1/20/2012 4:13 PM	TCL File	1 KB
Makefile	1/20/2012 4:13 PM	File	1 KB
Makefile	1/20/2012 4:13 PM	Export Definition F...	1 KB
UnixInstructions	1/20/2012 4:13 PM	Text Document	1 KB
WindowsInstructions	1/20/2012 4:13 PM	Text Document	2 KB



Name	Date modified	Type	Size
ElasticPPcpp	1/20/2012 4:13 PM	C++ Source	7 KB
ElasticPPcpp	1/20/2012 4:13 PM	C/C++ Header	4 KB
example1	1/20/2012 4:13 PM	TCL File	2 KB
Makefile	1/20/2012 4:13 PM	File	1 KB

Interface (.h file)

Material Name

material input properties

```
class ElasticPPcpp: public UniaxialMaterial {
      public:
        ElasticPPcpp(int tag, double e, double eyp);
        ElasticPPcpp();
       ~ ElasticPPcpp();
        int setTrialStrain(double strain, double strainRate=0.0);
        double getStrain(void);
        double getStress(void);
       double getTangent(void);
        double getInitialTangent(void);

        int commitState(void);
        int revertToLastCommit(void);
        int revertToStart(void);
       UniaxialMaterial *getCopy(void);
       int sendSelf(int commitTag, Channel &theChannel);
       int recvSelf(int commitTag, Channel &theChannel, FEM_ObjectBroker &theBroker);
       void Print(OPS_Stream &s, int flag = 0);
   private:
        double fyp, fyn; // pos & neg yield stress
        double ezero, E,ep; // init strain, elastic mod
        double ep; // plastic strain at last commit
        double trialStrain, trialStress, trialTangent;
        double commitStrain, commitStress, commitTangent;
};
```

data unique to material includes: Material parameters, & State variables

Implementation (.cpp file)

```
ElasticPPcpp::ElasticPPcpp(int tag, double e, double eyp)
:UniaxialMaterial(tag, 0),
 ezero(0), E(e), ep(0.0) trialStrain(0.0),trialStress(0.0),trialTangent(e),
 commitStreain(0.0),commitStress(0.0),commitTangent(e)
{
   fyp=E*eyp;
   fyn = -fyp;
}
ElasticPPcpp::ElasticPPcpp()
:UniaxialMaterial(tag, 0)
 fyp(0),fyn(0),ezero(0), E(0),ep(0),
 trialStrain(0.0),trialStress(0.0),trialTangent(0),
 commitStrain(0.0),commitStress(0.0),commitTangent(e){
 }
ElasticPPcpp::~ElasticPPcpp
{
   // does nothing .. No memory to clean up
}
UniaxialMaterial *ElasticPPcpp::getCopy(void)
{
 ElasticPPcpp *theCopy = new ElasticPPcpp(this->getTag(), E, fyp/E);
  return theCopy;
};
```

Hardest Method to Write

Implemetation

```
ElasticPPcpp::setTrialStrain(double strain, double strainRate)
{
     if (fabs(trialStrain - strain) < DBL_EPSILON)
       return 0;
    trialStrain = strain;
    double sigtrial;    // trial stress
    double f;           // yield function
    // compute trial stress
    sigtrial = E * ( trialStrain - ezero - ep );
    // evaluate yield function
    if ( sigtrial >= 0.0 )
        f =  sigtrial - fyp;
    else
        f = -sigtrial + fyn;
    double fYieldSurface = - E * DBL_EPSILON;
    if ( f <= fYieldSurface ) {

      // elastic
      trialStress = sigtrial;
      trialTangent = E;

    } else {

      // plastic
      if ( sigtrial > 0.0 ) {
        trialStress = fyp;
      } else {
        trialStress = fyn;
      }

      trialTangent = 0.0;
    }


     return  0;
}
```

Implementation

```
double
ElasticPPcpp::getStrain(void)
{
  return trialStrain;
}

double
ElasticPPcpp::getStress(void)
{
  return trialStress;
}

double
ElasticPPcpp::getTangent(void)
{
  return trialTangent;
}

int
ElasticPPcpp::revertToLastCommit(void)
{
  trialStrain = commitStrain;
  trialTangent = commitTangent;
  trialStress = commitStress;

  return 0;
}
```

OpenSees Interpreter (.cpp file)

Interpreter looking for this function in lib

```
OPS_Export void *
OPS_ElasticPPcpp()
{
  if (numElasticPPcpp == 0) {
    opserr << "ElasticPPcpp unaxia
    numElasticPPcpp =1;
  }
  // Pointer to a uniaxial material that will be returned
  UniaxialMaterial *theMaterial = 0;
  int    iData[1];
  double dData[2];
  int numData;

  numData = 1;
  if (OPS_GetIntInput(&numData, iData) != 0) {
    opserr << "WARNING invalid uniaxialMaterial ElasticPP tag" << endln;
    return 0;
  }
  numData = 2;
  if (OPS_GetDoubleInput(&numData, dData) != 0) {
    opserr << "WARNING invalid E & ep\n";
    return 0;
  }

  theMaterial = new ElasticPPcpp(iData[0], dData[0], dData[1]);

  return theMaterial;
}
```

You can give yourself some KUDOS, e.g. please reference … if you used this

parse the script for three material parameters

Function returns new material

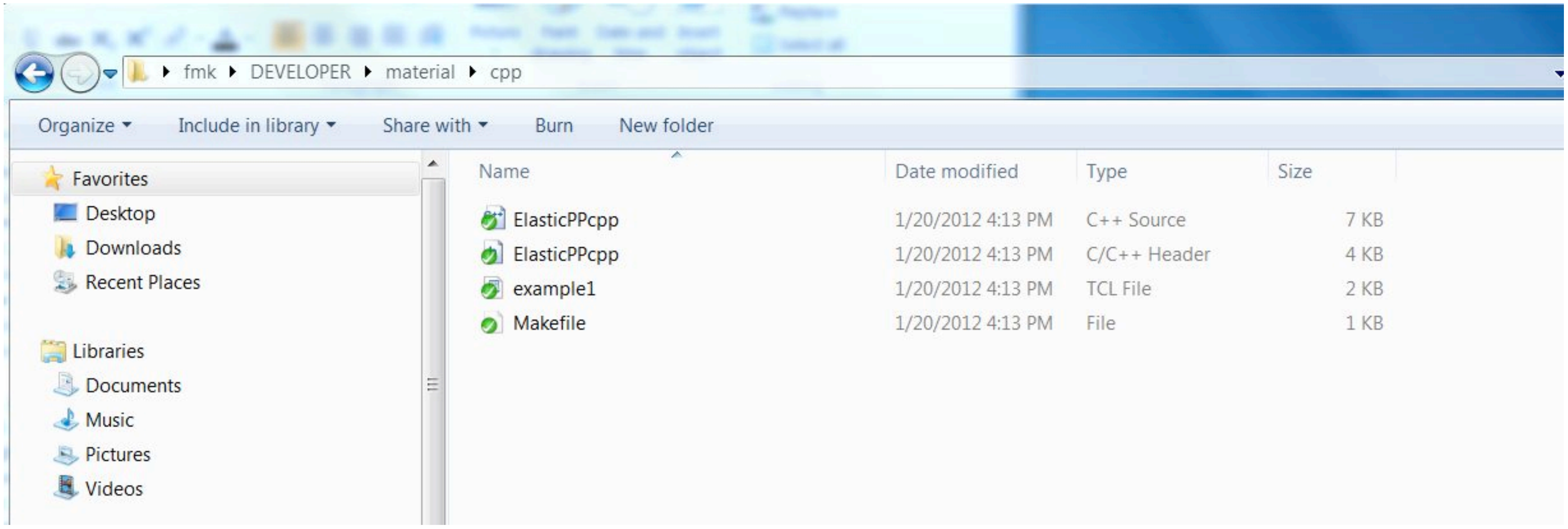
C & Fortran Procedural Languages Can Also Be Used

```
OPS_Export void
elasticPPc (matObj *thisObj,
          modelState *model,
          double *strain,
          double *tang,
          double *stress,
          int *isw,
          int *result)
{
  *result = 0;

  if (*isw == ISW_INIT) {

    double dData[2];
    int    iData[1];

 /* get the input data  - tag? E? e
    int numData = 1;
    OPS_GetIntInput(&numData,
    numData = 2;
    OPS_GetDoubleInput(&numD

    /* Allocate the element state */
```

```
 SUBROUTINE ELASTICPPF(matObj,model,strain,tang,stress,is

!DEC$ IF DEFINED (_DLL)
!DEC$ ATTRIBUTES DLLEXPORT :: ELASTICPPF
!DEC$ END IF
      use materialTypes
      use materialAPI
      implicit none
      IF (isw.eq.ISW_INIT) THEN

c      get the input data  - tag? E? eyp?

        numData = 1
        iPtr=>iData;
        err = OPS_GetIntInput(numData, iPtr)
        numData = 2
        dPtr=>dData;
        err = OPS_GetDoubleInput(numData, dPtr)

c      Allocate the element state
        matObj%tag = idata(1)
        matObj%nparam = 2
```

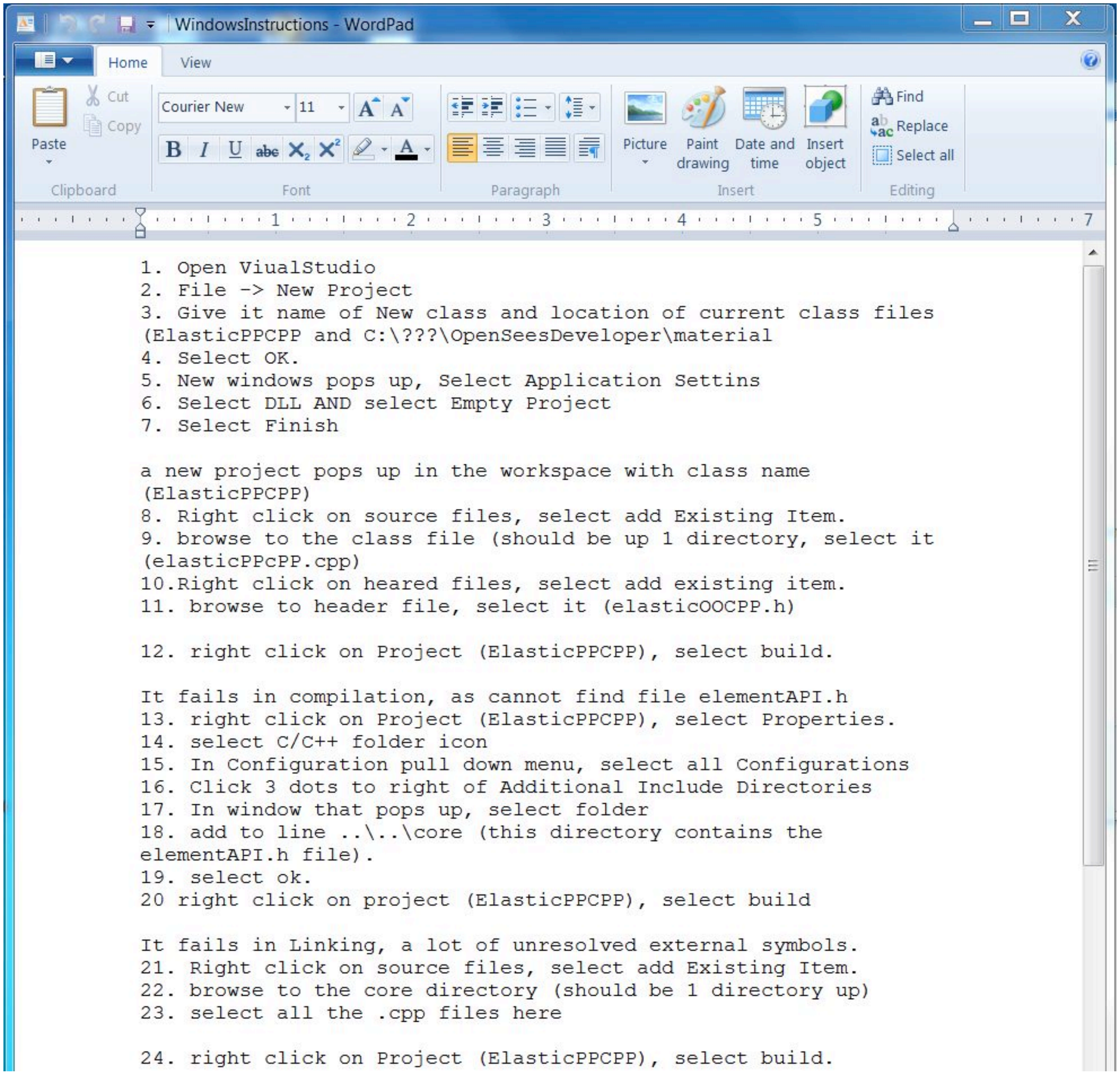
What Follows are the Steps required to Build ElasticPPcpp.dll on a Windows Machine with Visual Studio 2010 Installed

We Will Build the ElasticPPcpp.dll



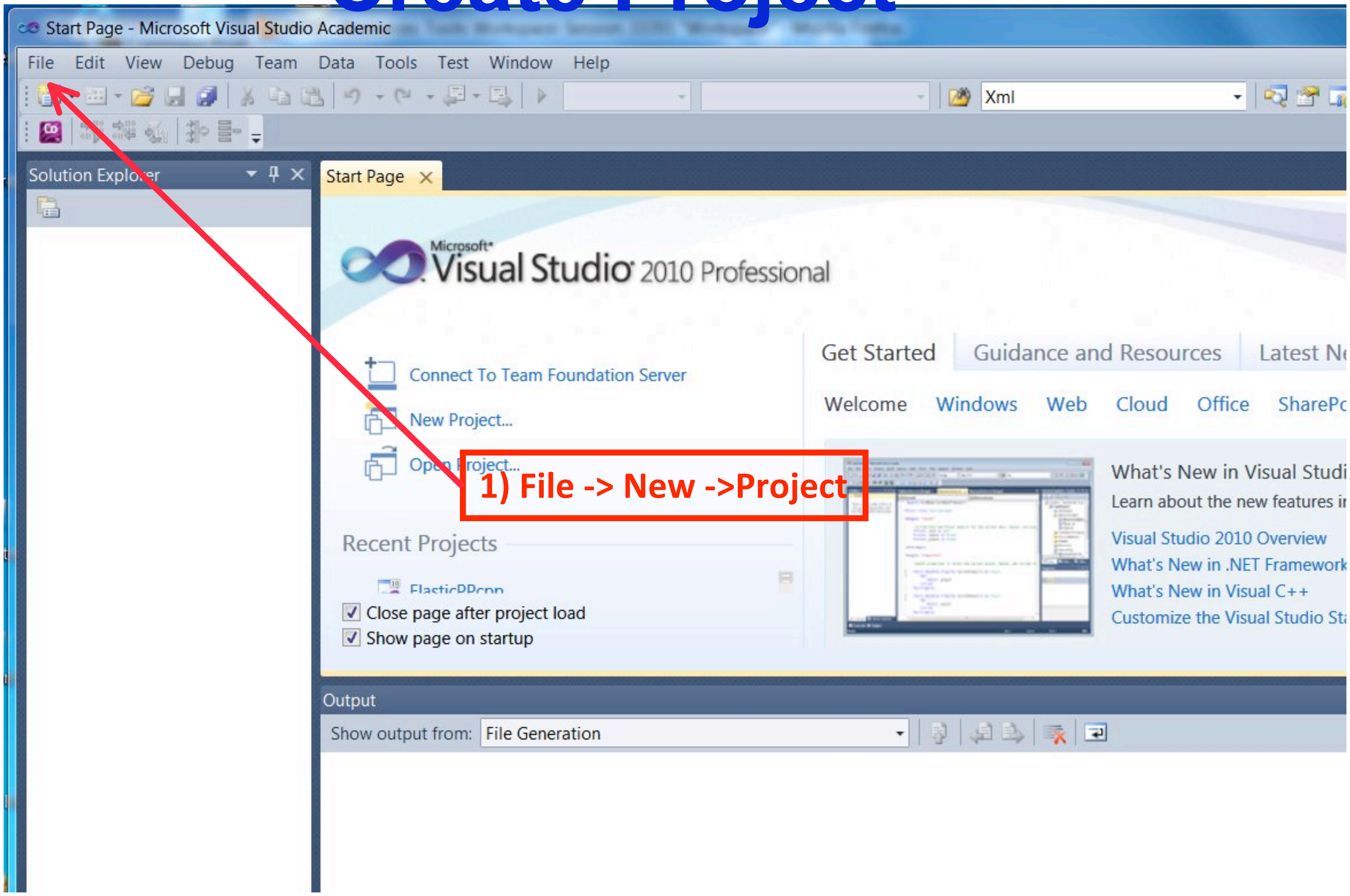
Name	Date modified	Type	Size
ElasticPPcpp	1/20/2012 4:13 PM	C++ Source	7 KB
ElasticPPcpp	1/20/2012 4:13 PM	C/C++ Header	4 KB
example1	1/20/2012 4:13 PM	TCL File	2 KB
Makefile	1/20/2012 4:13 PM	File	1 KB

Source code and example are in /DEVELOPER/material/cpp

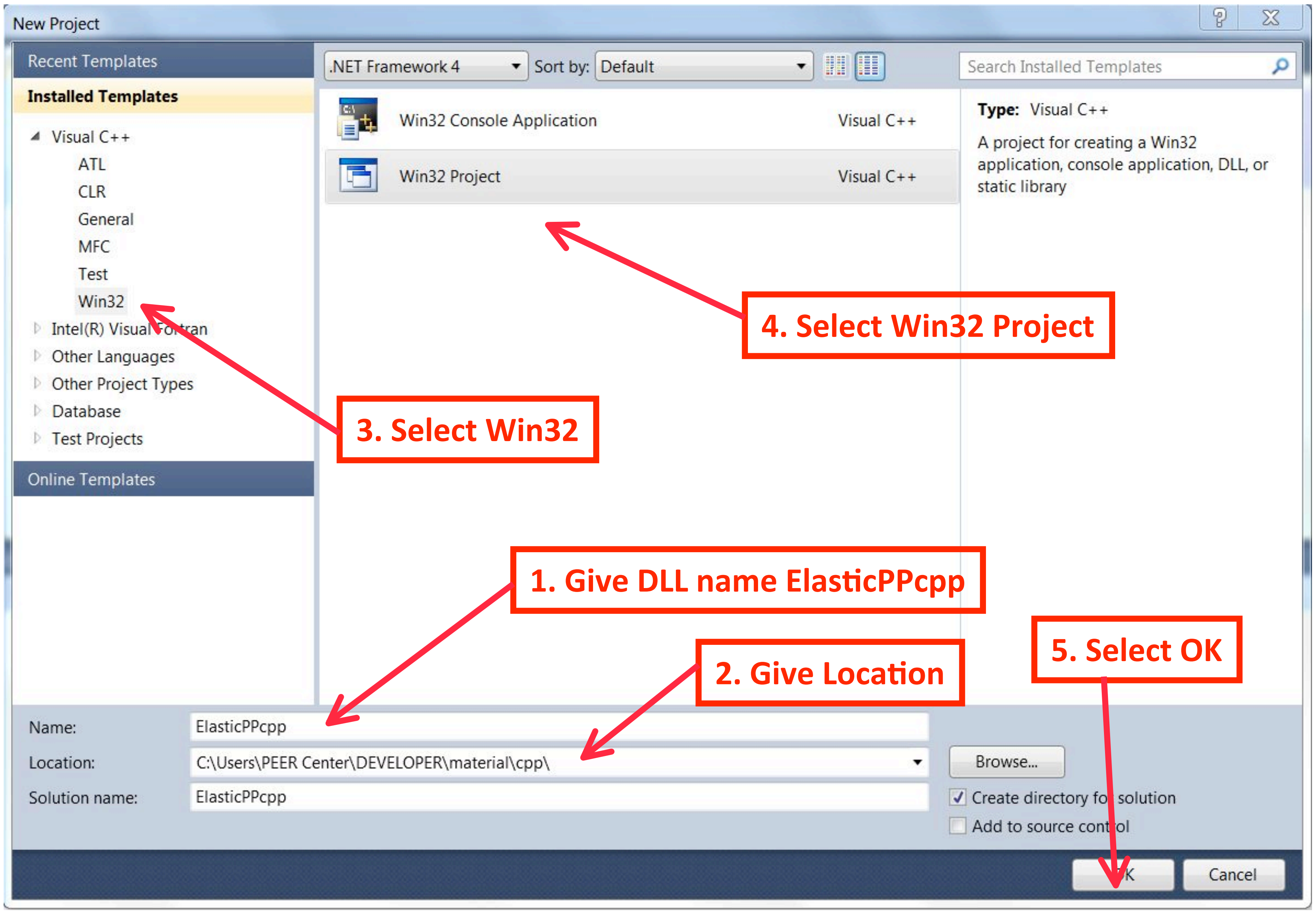


```
1. Open ViualStudio
2. File -> New Project
3. Give it name of New class and location of current class files
(ElasticPPCPP and C:\???\OpenSeesDeveloper\material
4. Select OK.
5. New windows pops up, Select Application Settins
6. Select DLL AND select Empty Project
7. Select Finish

a new project pops up in the workspace with class name
(ElasticPPCPP)
8. Right click on source files, select add Existing Item.
9. browse to the class file (should be up 1 directory, select it
(elasticPPcPP.cpp)
10.Right click on heared files, select add existing item.
11. browse to header file, select it (elasticOOCPP.h)

12. right click on Project (ElasticPPCPP), select build.

It fails in compilation, as cannot find file elementAPI.h
13. right click on Project (ElasticPPCPP), select Properties.
14. select C/C++ folder icon
15. In Configuration pull down menu, select all Configurations
16. Click 3 dots to right of Additional Include Directories
17. In window that pops up, select folder
18. add to line ..\..\core (this directory contains the
elementAPI.h file).
19. select ok.
20 right click on project (ElasticPPCPP), select build

It fails in Linking, a lot of unresolved external symbols.
21. Right click on source files, select add Existing Item.
22. browse to the core directory (should be 1 directory up)
23. select all the .cpp files here

24. right click on Project (ElasticPPCPP), select build.
```

Create Project


Start Page - Microsoft Visual Studio Academic
File Edit View Debug Team Data Tools Test Window Help
Xml
Solution Explorer
Start Page
Microsoft Visual Studio 2010 Professional
Connect To Team Foundation Server
New Project...
Open Project...
Recent Projects
ElasticPPcpp
Close page after project load
Show page on startup
Get Started
Guidance and Resources
Latest Ne
Welcome
Windows
Web
Cloud
Office
SharePo
What's New in Visual Studi
Learn about the new features in
Visual Studio 2010 Overview
What's New in .NET Framework
What's New in Visual C++
Customize the Visual Studio Sta
Output
Show output from:
File Generation

1) File -> New ->Project



Type: Visual C++

A project for creating a Win32 application, console application, DLL, or static library

Name: ElasticPPcpp

Location: C:\Users\PEER Center\DEVELOPER\material\cpp\

Solution name: ElasticPPcpp

[✓] Create directory for solution

[ ] Add to source control

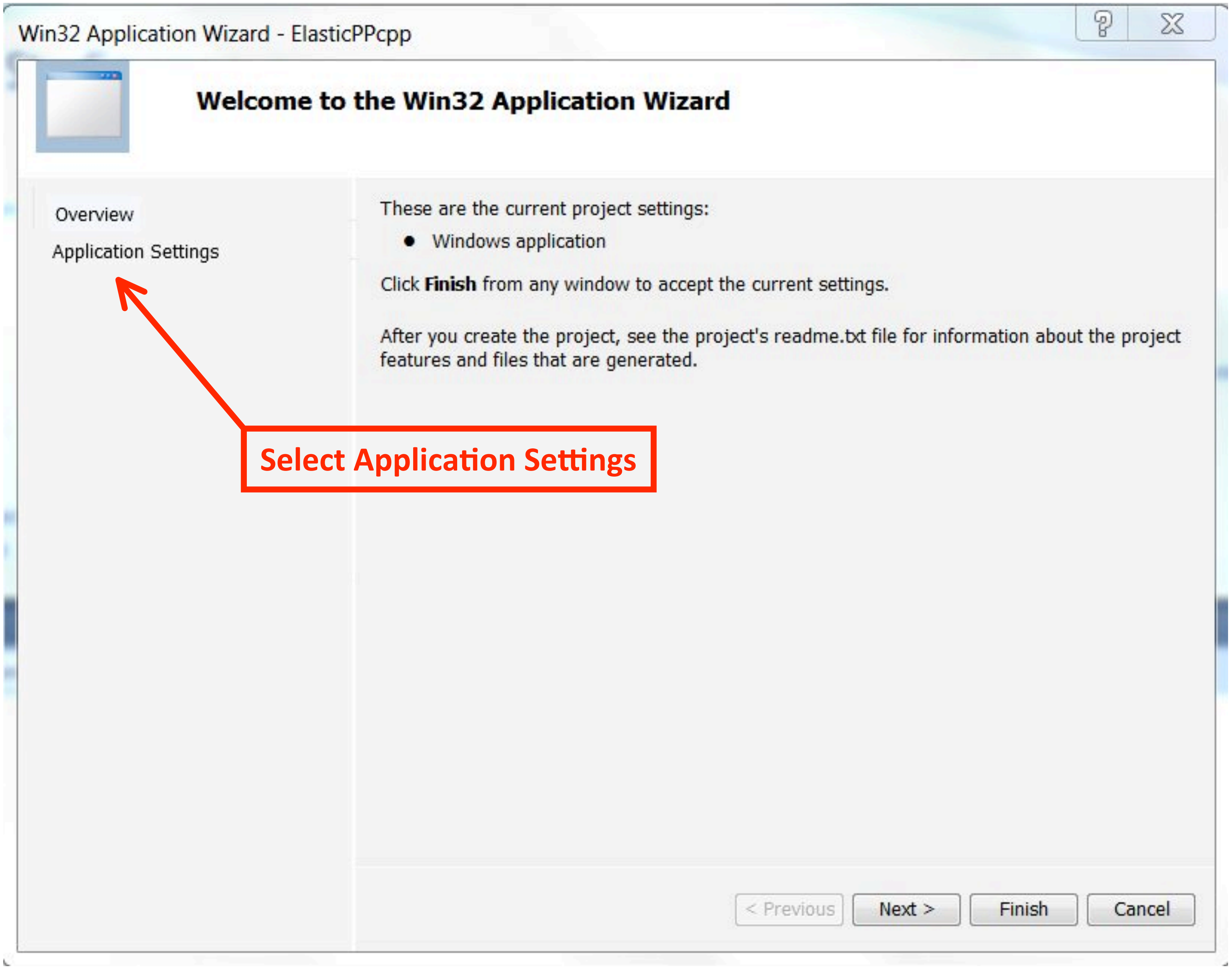
1. Give DLL name ElasticPPcpp

2. Give Location

3. Select Win32

4. Select Win32 Project

5. Select OK



Win32 Application Wizard - ElasticPPcpp

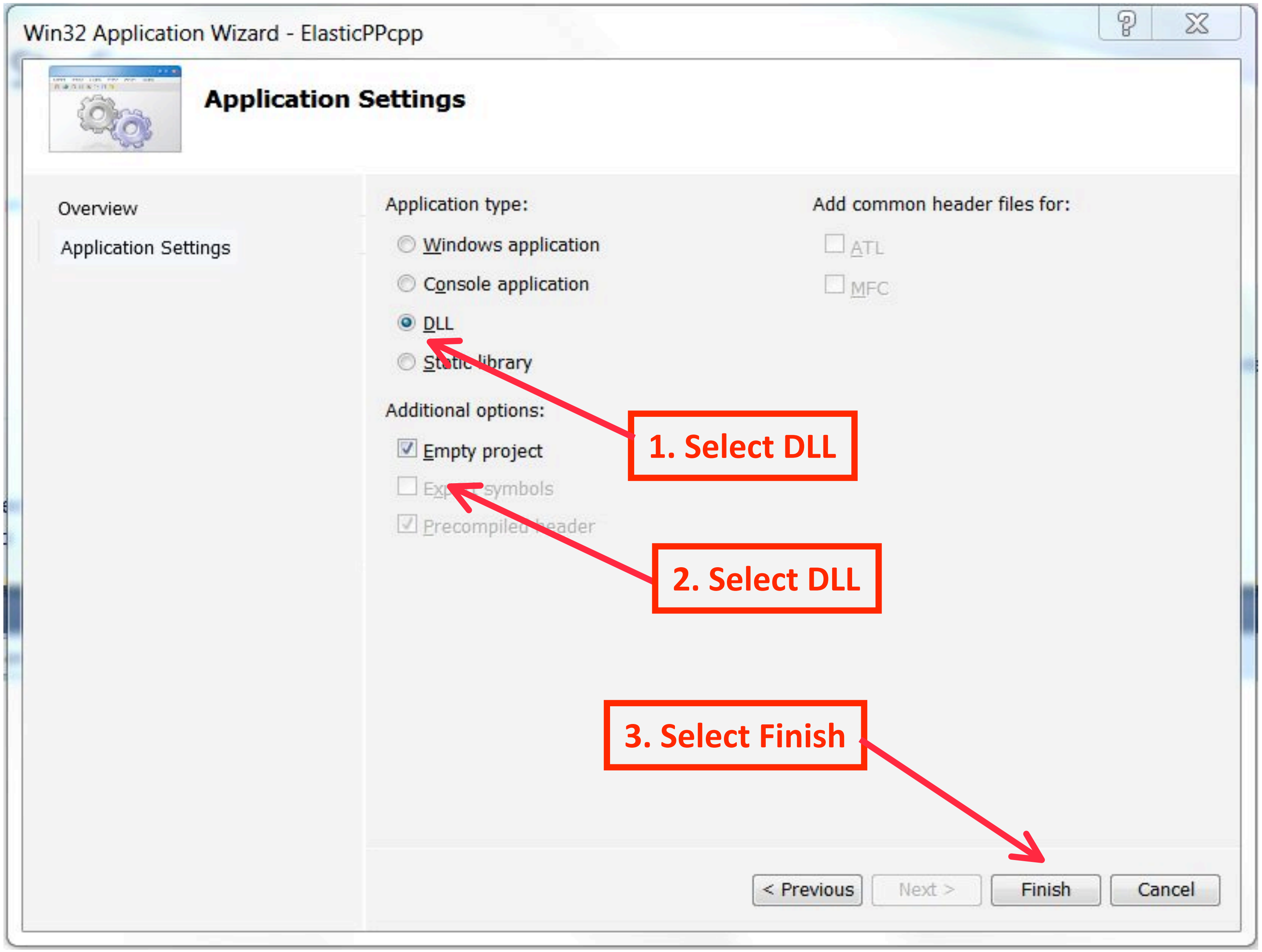
Welcome to the Win32 Application Wizard

These are the current project settings:

- Windows application

Click **Finish** from any window to accept the current settings.

After you create the project, see the project's readme.txt file for information about the project features and files that are generated.


Win32 Application Wizard - ElasticPPcpp

Application Settings

Application type:

- Windows application
- Console application
- [✓ DLL]
- Static library

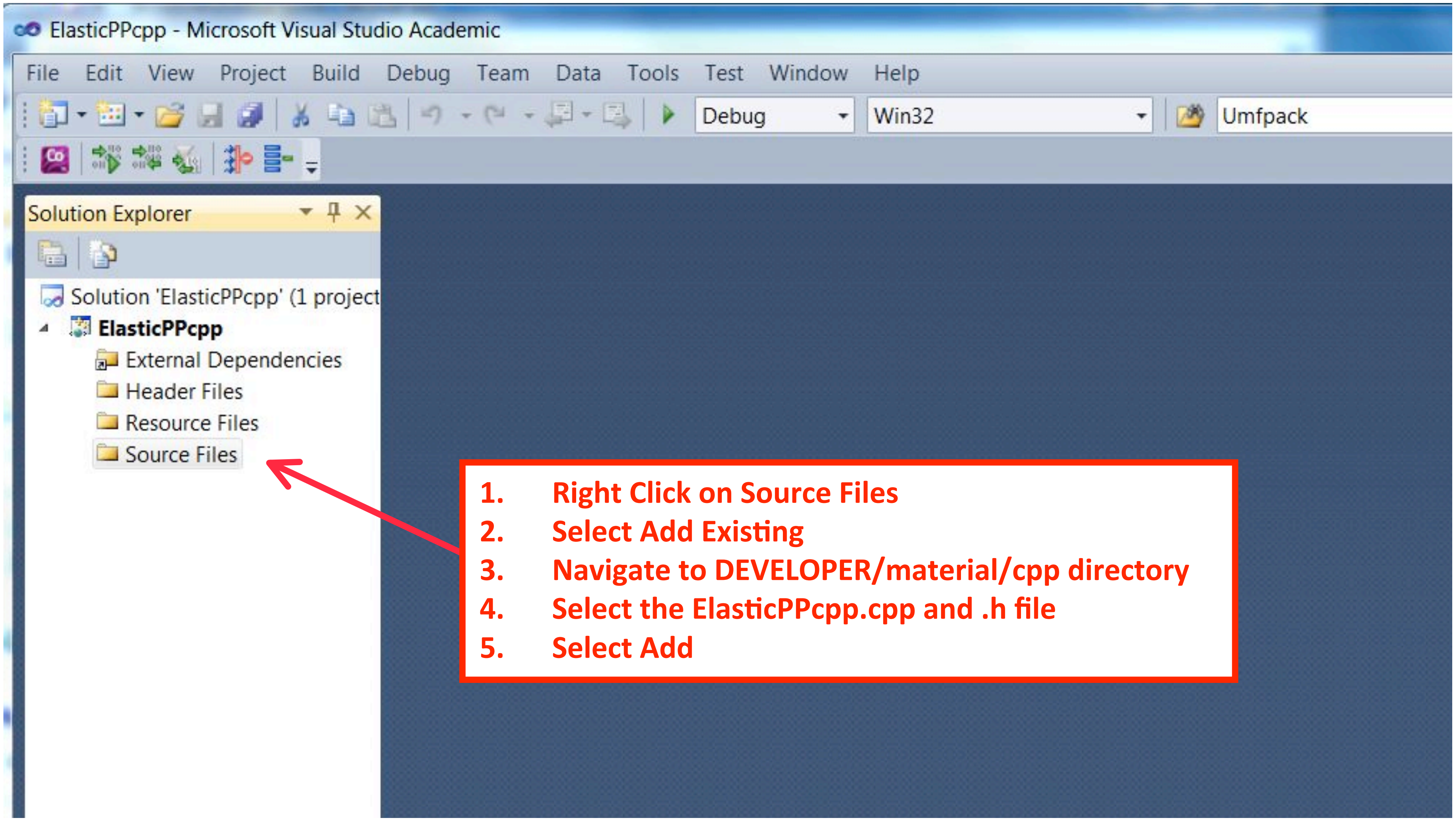
Additional options:

- [✓ Empty project]
- Export symbols
- [✓ Precompiled header]

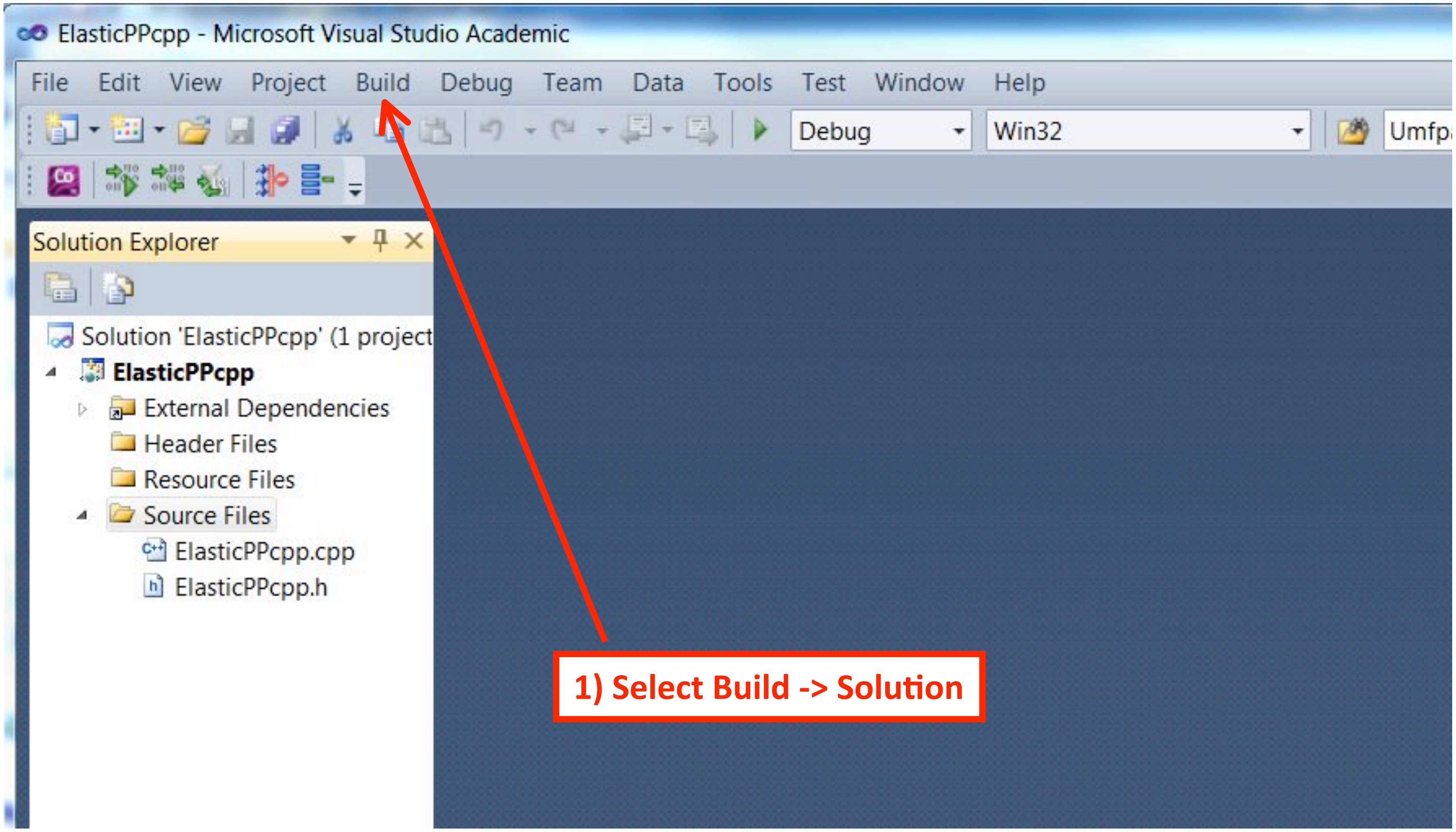
Add common header files for:

- ATL
- MFC

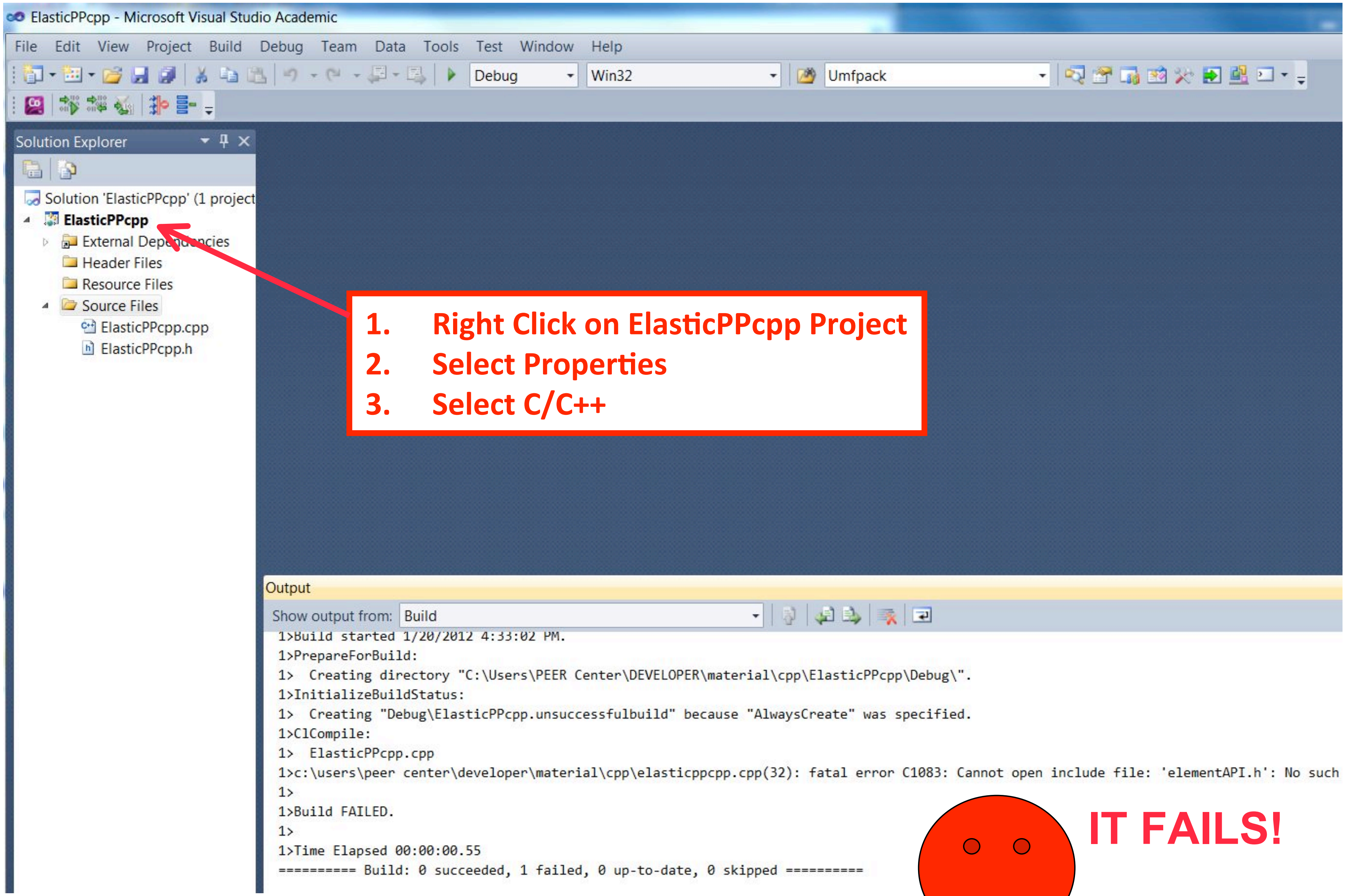
Add Files To Project


ElasticPPcpp - Microsoft Visual Studio Academic
File Edit View Project Build Debug Team Data Tools Test Window Help
Debug
Win32
Umfpack
Solution Explorer
Solution 'ElasticPPcpp' (1 project
ElasticPPcpp
External Dependencies
Header Files
Resource Files
Source Files

1. Right Click on Source Files
2. Select Add Existing
3. Navigate to DEVELOPER/material/cpp directory
4. Select the ElasticPPcpp.cpp and .h file
5. Select Add

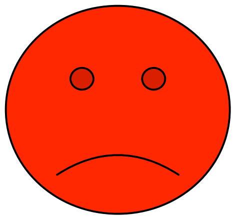


1) Select Build -> Solution

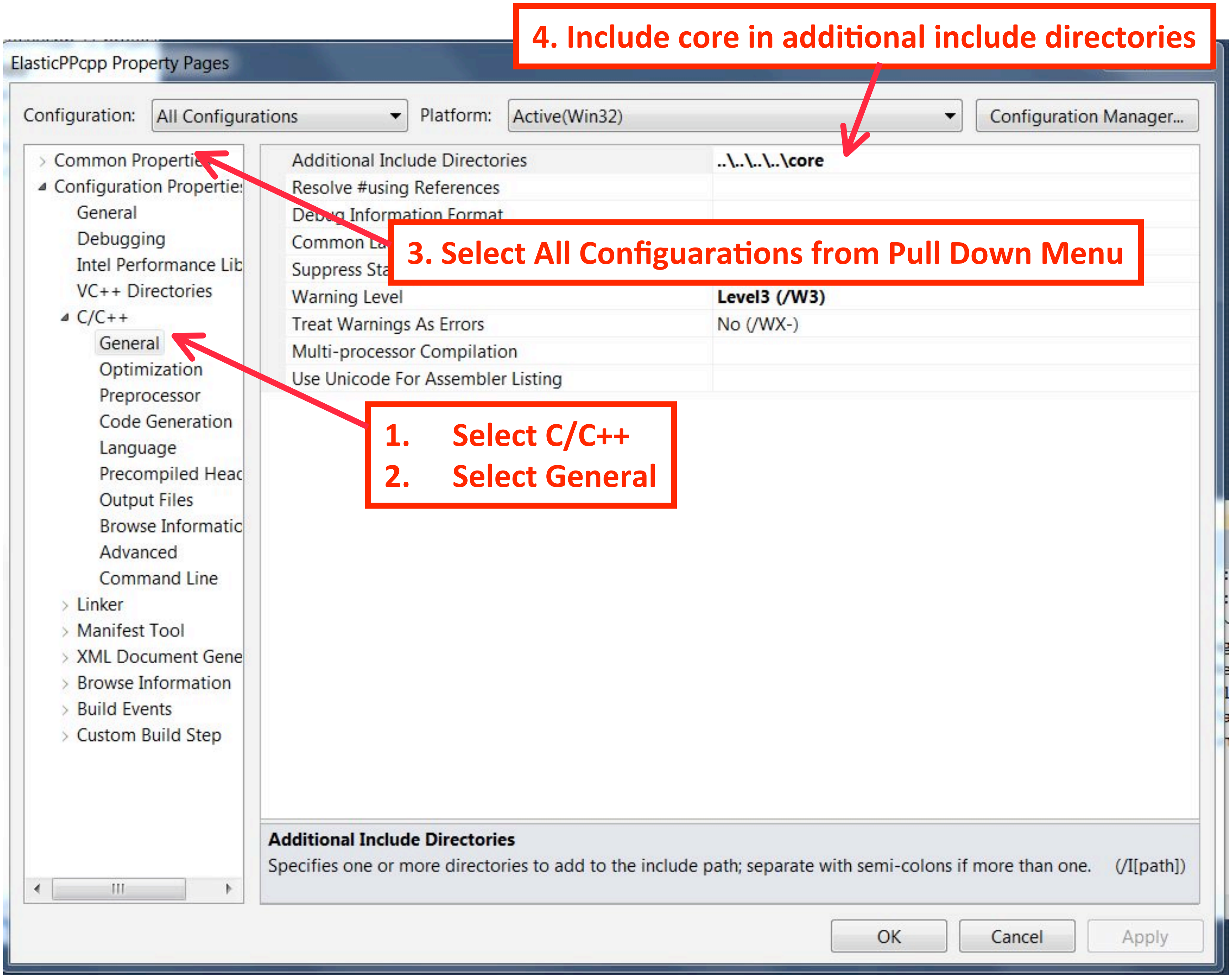


1. Right Click on ElasticPPcpp Project
2. Select Properties
3. Select C/C++

```
1>Build started 1/20/2012 4:33:02 PM.
1>PrepareForBuild:
1>  Creating directory "C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp\Debug\".
1>InitializeBuildStatus:
1>  Creating "Debug\ElasticPPcpp.unsuccessfulbuild" because "AlwaysCreate" was specified.
1>ClCompile:
1>  ElasticPPcpp.cpp
1>c:\users\peer center\developer\material\cpp\elasticppcpp.cpp(32): fatal error C1083: Cannot open include file: 'elementAPI.h': No such
1>
1>Build FAILED.
1>
1>Time Elapsed 00:00:00.55
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```



IT FAILS!


4. Include core in additional include directories

3. Select All Configuarations from Pull Down Menu

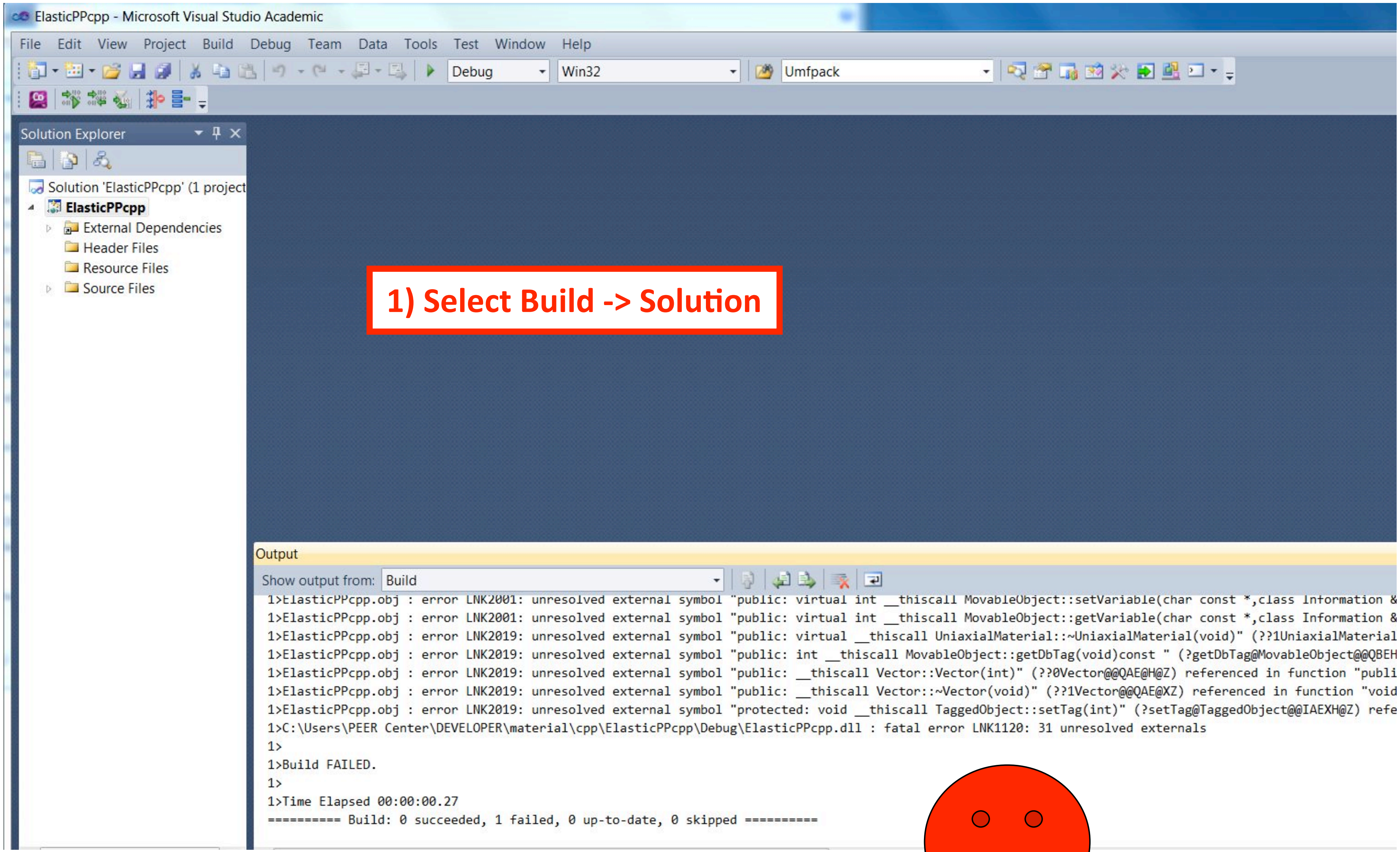
1. Select C/C++
2. Select General

ElasticPPcpp Property Pages

Configuration: All Configurations
Platform: Active(Win32)

Additional Include Directories	..\..\..\..\core
Resolve #using References	
Debug Information Format	
Common Language	
Suppress Startup	
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
Multi-processor Compilation	
Use Unicode For Assembler Listing	

Additional Include Directories
Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])

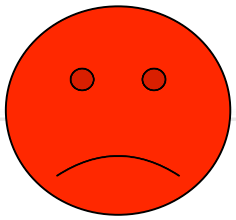

ElasticPPcpp - Microsoft Visual Studio Academic
File Edit View Project Build Debug Team Data Tools Test Window Help
Solution Explorer
Solution 'ElasticPPcpp' (1 project
ElasticPPcpp
External Dependencies
Header Files
Resource Files
Source Files
Output

Debug | Win32 | Umfpack

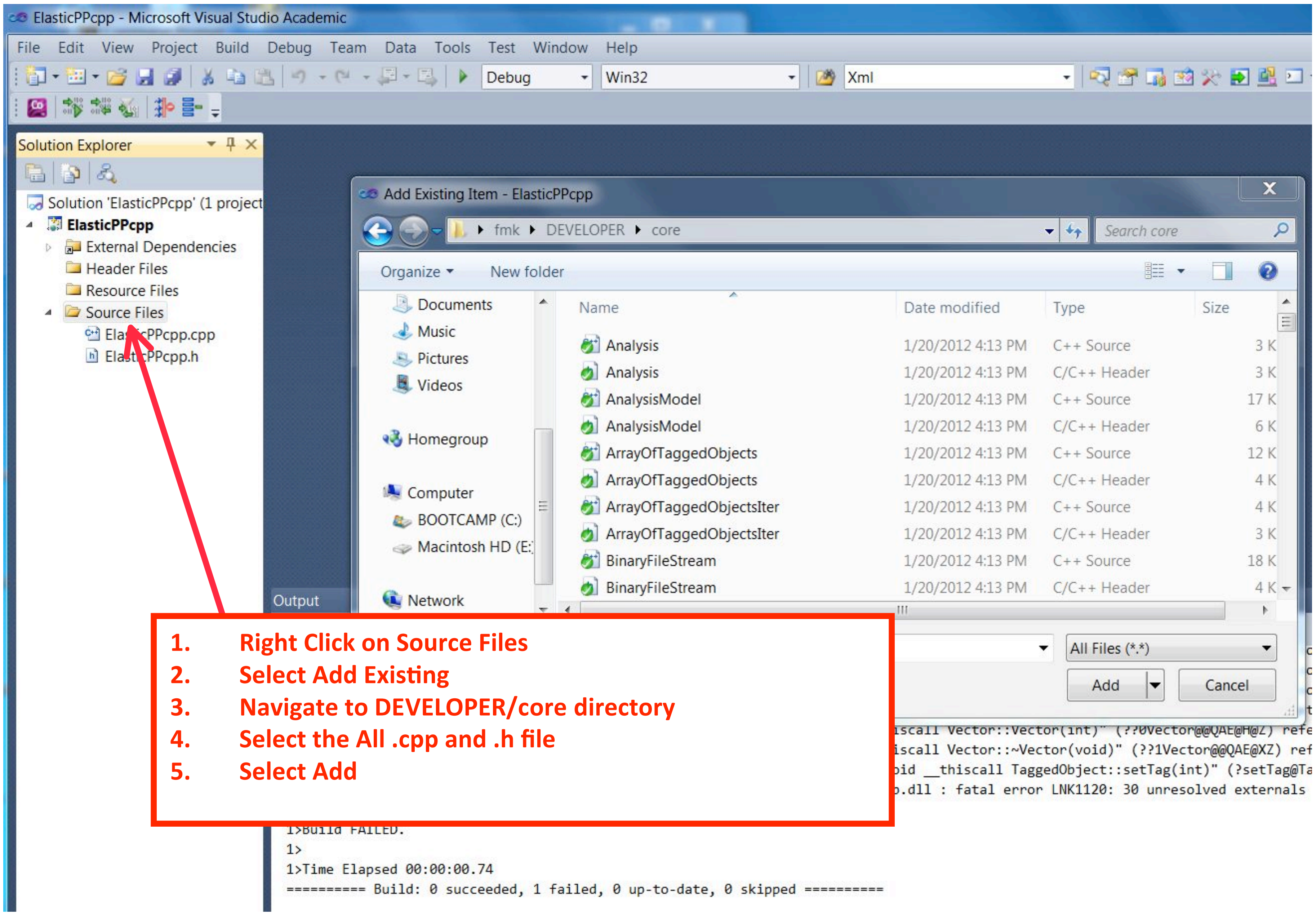
Show output from: Build

1) Select Build -> Solution

```
1>ElasticPPcpp.obj : error LNK2001: unresolved external symbol "public: virtual int __thiscall MovableObject::setVariable(char const *,class Information &
1>ElasticPPcpp.obj : error LNK2001: unresolved external symbol "public: virtual int __thiscall MovableObject::getVariable(char const *,class Information &
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: virtual __thiscall UniaxialMaterial::~UniaxialMaterial(void)" (??1UniaxialMaterial
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: int __thiscall MovableObject::getDbTag(void)const " (?getDbTag@MovableObject@@QBEH
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: __thiscall Vector::Vector(int)" (??0Vector@@QAE@H@Z) referenced in function "publi
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: __thiscall Vector::~Vector(void)" (??1Vector@@QAE@XZ) referenced in function "void
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "protected: void __thiscall TaggedObject::setTag(int)" (?setTag@TaggedObject@@IAEXH@Z) refe
1>C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp\Debug\ElasticPPcpp.dll : fatal error LNK1120: 31 unresolved externals
1>
1>Build FAILED.
1>
1>Time Elapsed 00:00:00.27
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```



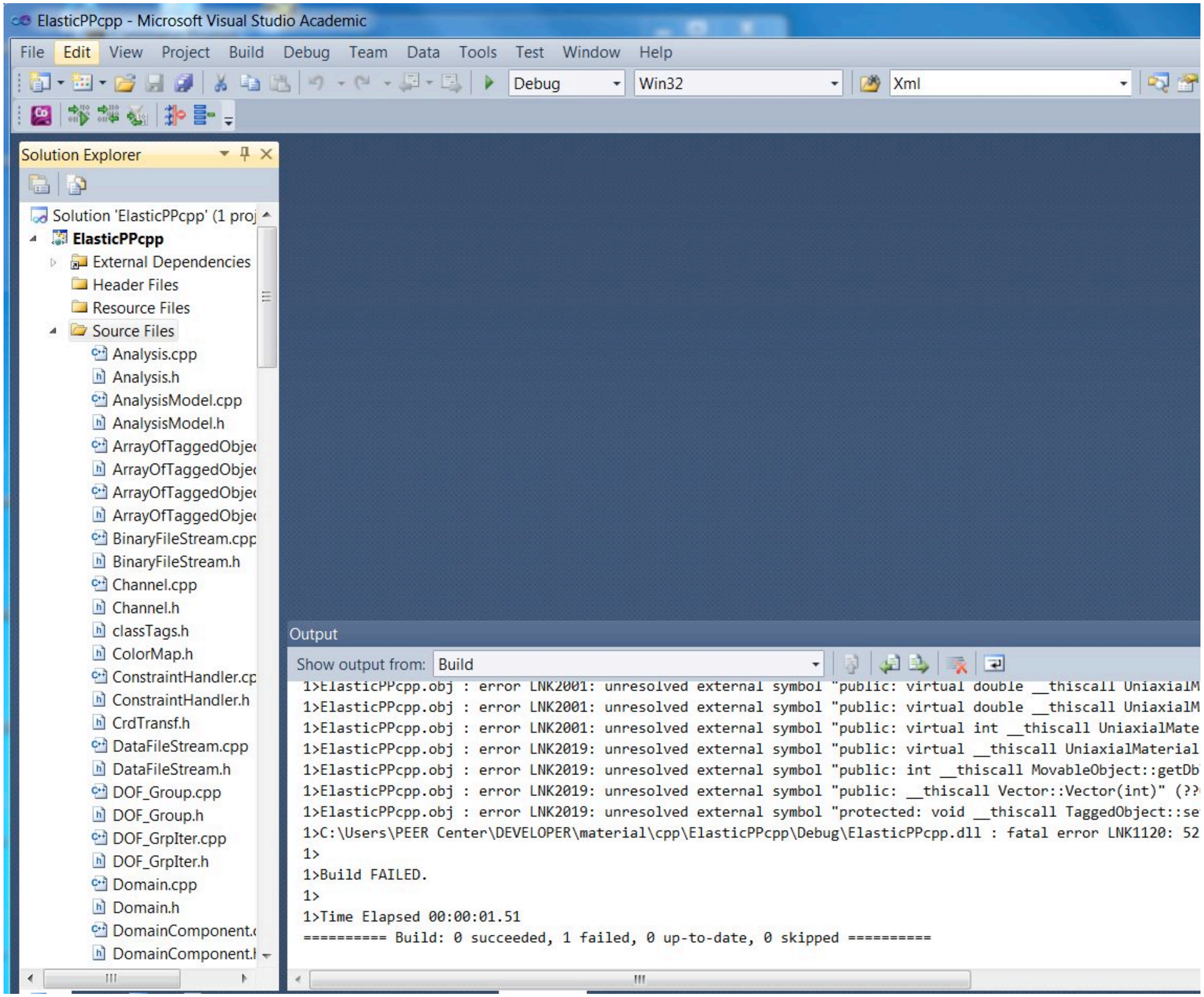
IT FAILS!



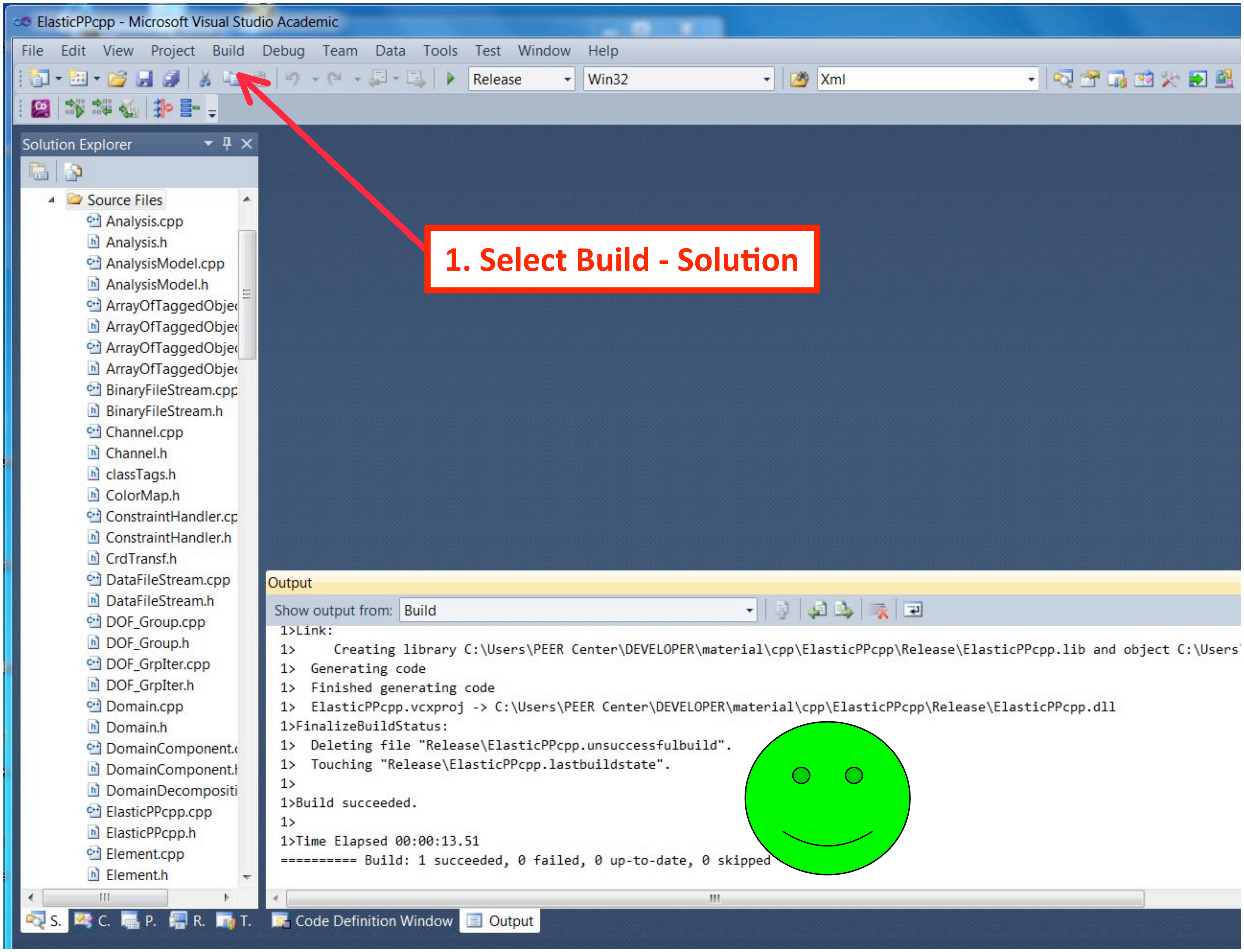
Name	Date modified	Type	Size
Analysis	1/20/2012 4:13 PM	C++ Source	3 K
Analysis	1/20/2012 4:13 PM	C/C++ Header	3 K
AnalysisModel	1/20/2012 4:13 PM	C++ Source	17 K
AnalysisModel	1/20/2012 4:13 PM	C/C++ Header	6 K
ArrayOfTaggedObjects	1/20/2012 4:13 PM	C++ Source	12 K
ArrayOfTaggedObjects	1/20/2012 4:13 PM	C/C++ Header	4 K
ArrayOfTaggedObjectsIter	1/20/2012 4:13 PM	C++ Source	4 K
ArrayOfTaggedObjectsIter	1/20/2012 4:13 PM	C/C++ Header	3 K
BinaryFileStream	1/20/2012 4:13 PM	C++ Source	18 K
BinaryFileStream	1/20/2012 4:13 PM	C/C++ Header	4 K

```
iscall Vector::Vector(int)" (??0Vector@@QAE@H@Z) refe
iscall Vector::~Vector(void)" (??1Vector@@QAE@XZ) ref
oid __thiscall TaggedObject::setTag(int)" (?setTag@Ta
o.dll : fatal error LNK1120: 30 unresolved externals
1>Build FAILED.
1>
1>Time Elapsed 00:00:00.74
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```

1. Right Click on Source Files
2. Select Add Existing
3. Navigate to DEVELOPER/core directory
4. Select the All .cpp and .h file
5. Select Add



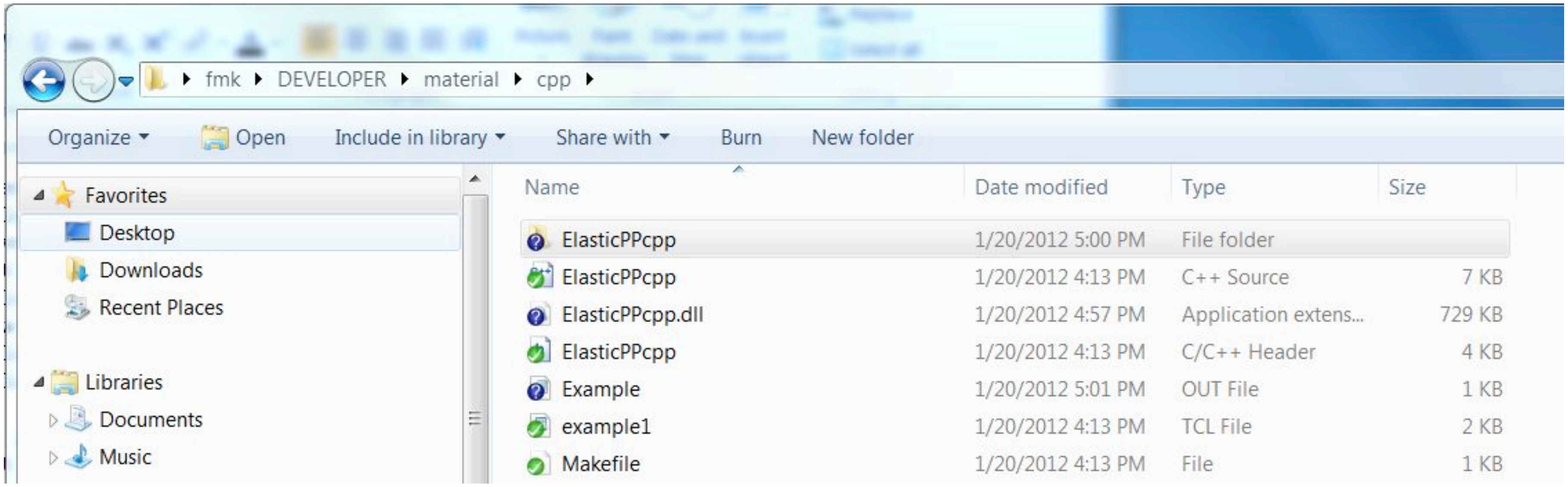
```
1>ElasticPPcpp.obj : error LNK2001: unresolved external symbol "public: virtual double __thiscall UniaxialM
1>ElasticPPcpp.obj : error LNK2001: unresolved external symbol "public: virtual double __thiscall UniaxialM
1>ElasticPPcpp.obj : error LNK2001: unresolved external symbol "public: virtual int __thiscall UniaxialMate
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: virtual __thiscall UniaxialMaterial
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: int __thiscall MovableObject::getDb
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "public: __thiscall Vector::Vector(int)" (??
1>ElasticPPcpp.obj : error LNK2019: unresolved external symbol "protected: void __thiscall TaggedObject::se
1>C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp\Debug\ElasticPPcpp.dll : fatal error LNK1120: 52
1>
1>Build FAILED.
1>
1>Time Elapsed 00:00:01.51
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```



1. Select Build - Solution

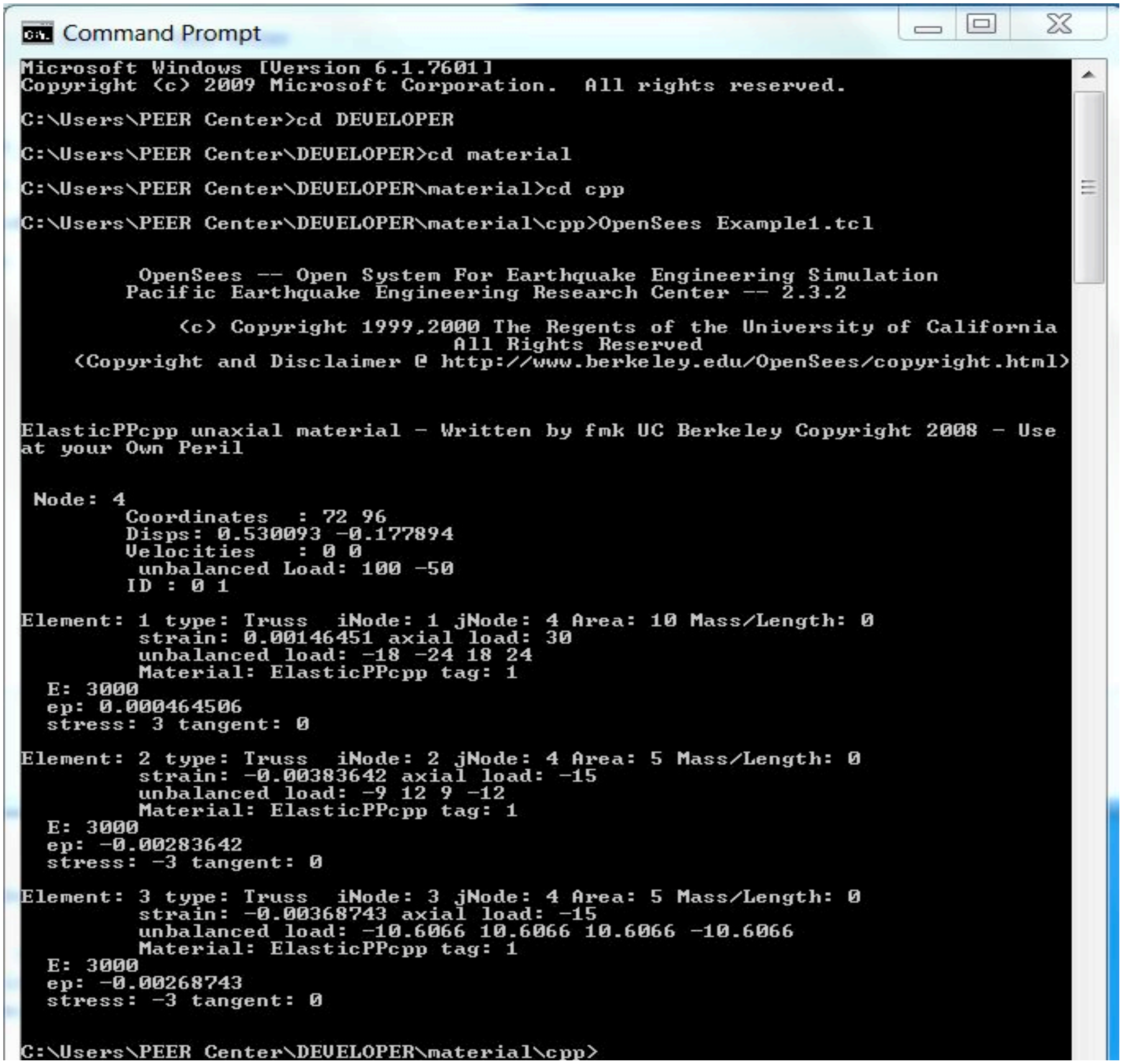
```
1>Link:
1>     Creating library C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp\Release\ElasticPPcpp.lib and object C:\Users
1>  Generating code
1>  Finished generating code
1>  ElasticPPcpp.vcxproj -> C:\Users\PEER Center\DEVELOPER\material\cpp\ElasticPPcpp\Release\ElasticPPcpp.dll
1>FinalizeBuildStatus:
1>  Deleting file "Release\ElasticPPcpp.unsuccessfulbuild".
1>  Touching "Release\ElasticPPcpp.lastbuildstate".
1>
1>Build succeeded.
1>
1>Time Elapsed 00:00:13.51
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped
```

Copy ElasticPPcpp.dll from location into current directory



Name	Date modified	Type	Size
ElasticPPcpp	1/20/2012 5:00 PM	File folder	
ElasticPPcpp	1/20/2012 4:13 PM	C++ Source	7 KB
ElasticPPcpp.dll	1/20/2012 4:57 PM	Application extens...	729 KB
ElasticPPcpp	1/20/2012 4:13 PM	C/C++ Header	4 KB
Example	1/20/2012 5:01 PM	OUT File	1 KB
example1	1/20/2012 4:13 PM	TCL File	2 KB
Makefile	1/20/2012 4:13 PM	File	1 KB

Now run Example



Command Prompt

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\PEER Center>cd DEVELOPER

C:\Users\PEER Center\DEVELOPER>cd material

C:\Users\PEER Center\DEVELOPER\material>cd cpp

C:\Users\PEER Center\DEVELOPER\material\cpp>OpenSees Example1.tcl


         OpenSees -- Open System For Earthquake Engineering Simulation
        Pacific Earthquake Engineering Research Center -- 2.3.2

            (c) Copyright 1999,2000 The Regents of the University of California
                                 All Rights Reserved
    (Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)



ElasticPPcpp unaxial material - Written by fmk UC Berkeley Copyright 2008 - Use
at your Own Peril


 Node: 4
        Coordinates  : 72 96
        Disps: 0.530093 -0.177894
        Velocities   : 0 0
         unbalanced Load: 100 -50
        ID : 0 1

Element: 1 type: Truss  iNode: 1 jNode: 4 Area: 10 Mass/Length: 0
         strain: 0.00146451 axial load: 30
         unbalanced load: -18 -24 18 24
         Material: ElasticPPcpp tag: 1
  E: 3000
  ep: 0.000464506
  stress: 3 tangent: 0

Element: 2 type: Truss  iNode: 2 jNode: 4 Area: 5 Mass/Length: 0
         strain: -0.00383642 axial load: -15
         unbalanced load: -9 12 9 -12
         Material: ElasticPPcpp tag: 1
  E: 3000
  ep: -0.00283642
  stress: -3 tangent: 0

Element: 3 type: Truss  iNode: 3 jNode: 4 Area: 5 Mass/Length: 0
         strain: -0.00368743 axial load: -15
         unbalanced load: -10.6066 10.6066 10.6066 -10.6066
         Material: ElasticPPcpp tag: 1
  E: 3000
  ep: -0.00268743
  stress: -3 tangent: 0


C:\Users\PEER Center\DEVELOPER\material\cpp>
```

What Follows are the Steps required to Build ElasticPPcpp.so on a Linux Machine with gcc installed

NOTE: We Will use NEEShub for this demonstration (http://nees.org)

3 Simple Steps!

1. Download code

```
svn co svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk/OpenSees/Developer Developer
```

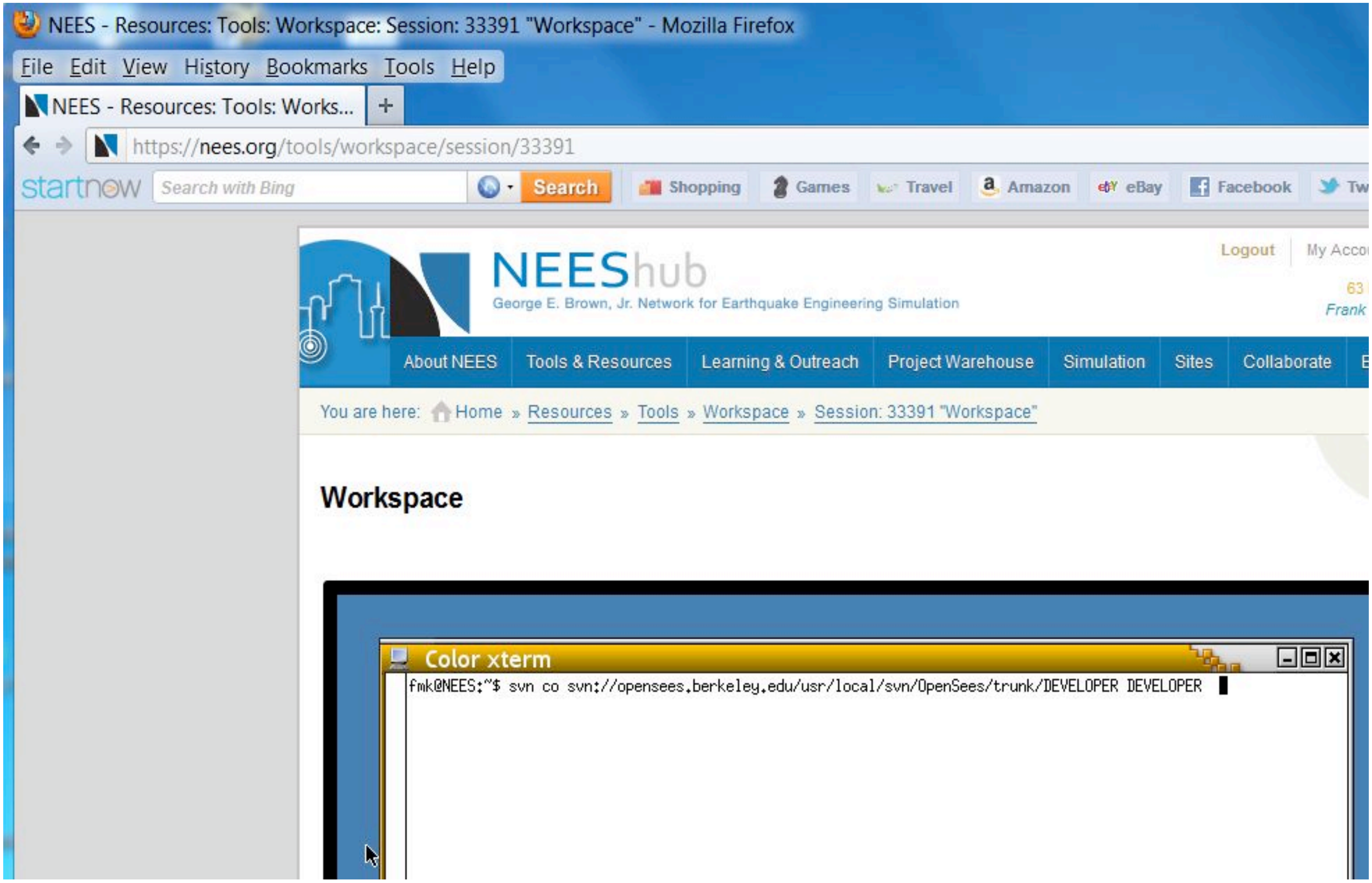
2. cd DEVELOPER/material/cpp
3. type make

if you type ls you should see the .so and you

Can test it using >OpenSees example1.tcl

NOTE: mac users must have xcode installed and must open DEVELOPER/Makefile.def and switch comments on lines 1 and 2 before step 3.

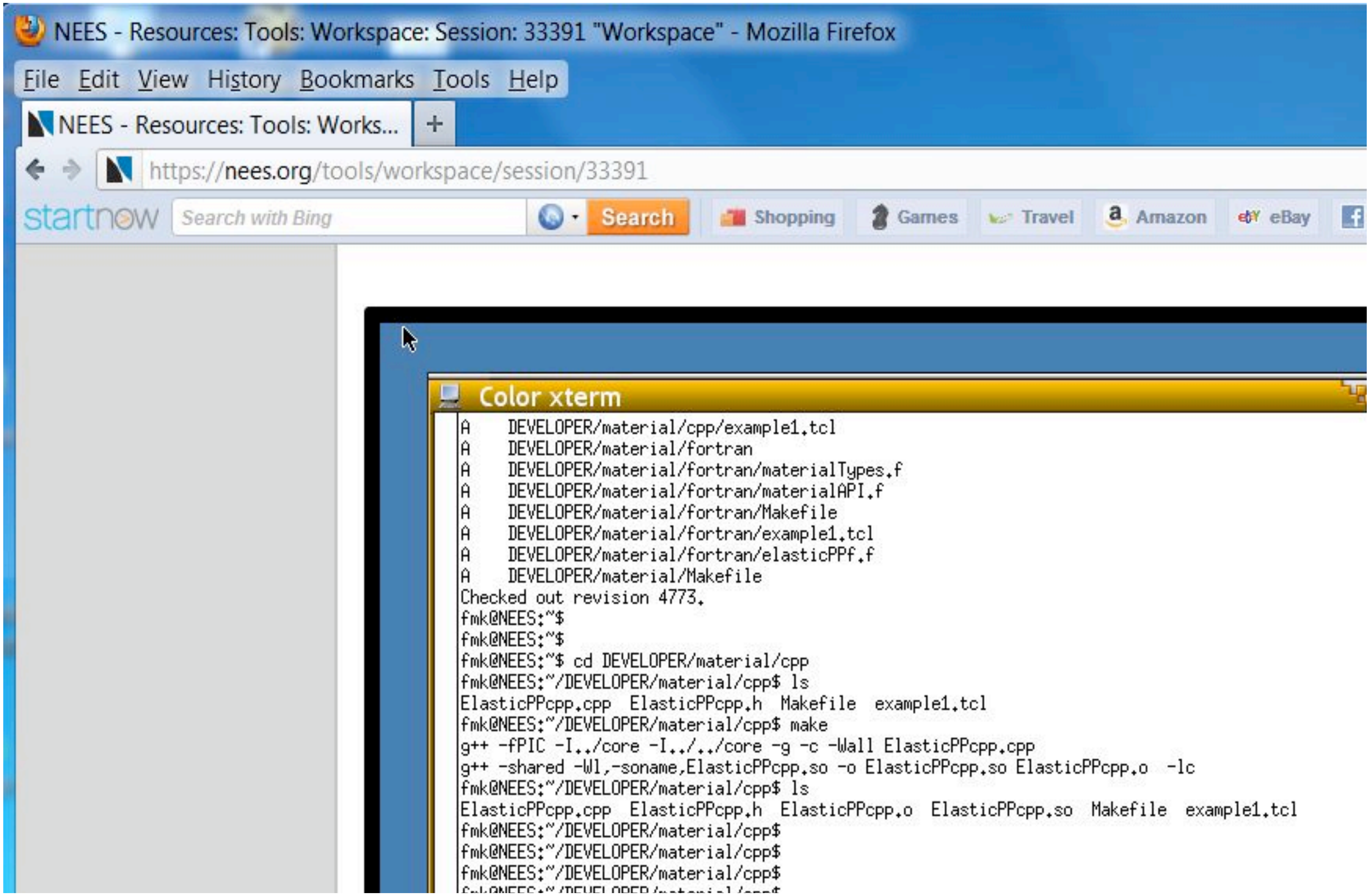
1: Download Source Code



NEES - Resources: Tools: Workspace: Session: 33391 "Workspace" - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://nees.org/tools/workspace/session/33391
NEEShub
George E. Brown, Jr. Network for Earthquake Engineering Simulation
About NEES
Tools & Resources
Learning & Outreach
Project Warehouse
Simulation
Sites
Collaborate
You are here: Home » Resources » Tools » Workspace » Session: 33391 "Workspace"
Workspace
Color xterm

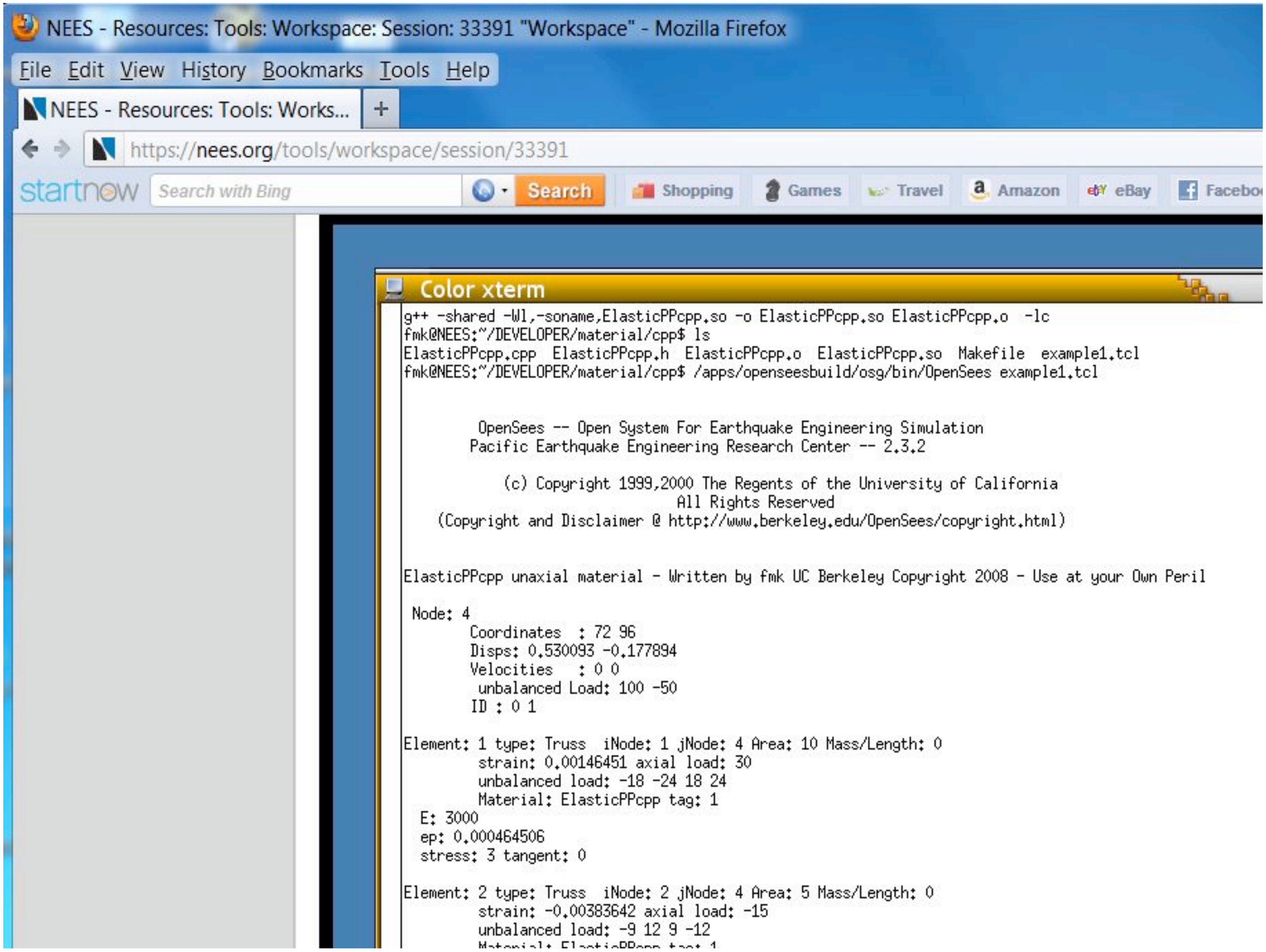
```
fmk@NEES:~$ svn co svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk/DEVELOPER DEVELOPER
```

2/3: cd to Directory & Type make


NEES - Resources: Tools: Workspace: Session: 33391 "Workspace" - Mozilla Firefox
File Edit View History Bookmarks Tools Help
NEES - Resources: Tools: Works...
https://nees.org/tools/workspace/session/33391
startnow
Search with Bing
Search
Shopping
Games
Travel
Amazon
eBay
Color xterm

```
A    DEVELOPER/material/cpp/example1.tcl
A    DEVELOPER/material/fortran
A    DEVELOPER/material/fortran/materialTypes.f
A    DEVELOPER/material/fortran/materialAPI.f
A    DEVELOPER/material/fortran/Makefile
A    DEVELOPER/material/fortran/example1.tcl
A    DEVELOPER/material/fortran/elasticPPf.f
A    DEVELOPER/material/Makefile
Checked out revision 4773.
fmk@NEES:~$
fmk@NEES:~$
fmk@NEES:~$ cd DEVELOPER/material/cpp
fmk@NEES:~/DEVELOPER/material/cpp$ ls
ElasticPPcpp.cpp  ElasticPPcpp.h  Makefile  example1.tcl
fmk@NEES:~/DEVELOPER/material/cpp$ make
g++ -fPIC -I../core -I../../core -g -c -Wall ElasticPPcpp.cpp
g++ -shared -Wl,-soname,ElasticPPcpp.so -o ElasticPPcpp.so ElasticPPcpp.o  -lc
fmk@NEES:~/DEVELOPER/material/cpp$ ls
ElasticPPcpp.cpp  ElasticPPcpp.h  ElasticPPcpp.o  ElasticPPcpp.so  Makefile  example1.tcl
fmk@NEES:~/DEVELOPER/material/cpp$
fmk@NEES:~/DEVELOPER/material/cpp$
fmk@NEES:~/DEVELOPER/material/cpp$
```

Run It



NEES - Resources: Tools: Workspace: Session: 33391 "Workspace" - Mozilla Firefox
File Edit View History Bookmarks Tools Help
NEES - Resources: Tools: Works...
https://nees.org/tools/workspace/session/33391
startnow
Search with Bing
Search
Shopping
Games
Travel
Amazon
eBay
Facebo
Color xterm

```
g++ -shared -Wl,-soname,ElasticPPcpp.so -o ElasticPPcpp.so ElasticPPcpp.o  -lc
fmk@NEES:~/DEVELOPER/material/cpp$ ls
ElasticPPcpp.cpp  ElasticPPcpp.h  ElasticPPcpp.o  ElasticPPcpp.so  Makefile  example1.tcl
fmk@NEES:~/DEVELOPER/material/cpp$ /apps/openseesbuild/osg/bin/OpenSees example1.tcl


         OpenSees -- Open System For Earthquake Engineering Simulation
        Pacific Earthquake Engineering Research Center -- 2.3.2

            (c) Copyright 1999,2000 The Regents of the University of California
                                 All Rights Reserved
    (Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)


ElasticPPcpp unaxial material - Written by fmk UC Berkeley Copyright 2008 - Use at your Own Peril

 Node: 4
        Coordinates  : 72 96
        Disps: 0.530093 -0.177894
        Velocities   : 0 0
         unbalanced Load: 100 -50
        ID : 0 1

Element: 1 type: Truss  iNode: 1 jNode: 4 Area: 10 Mass/Length: 0
         strain: 0.00146451 axial load: 30
         unbalanced load: -18 -24 18 24
         Material: ElasticPPcpp tag: 1
  E: 3000
  ep: 0.000464506
  stress: 3 tangent: 0

Element: 2 type: Truss  iNode: 2 jNode: 4 Area: 5 Mass/Length: 0
         strain: -0.00383642 axial load: -15
         unbalanced load: -9 12 9 -12
```

NEXT SEMINAR

- Feb 2012, Topic **EITHER** be High Performance Computing and OpenSees or How To Model Soil-Structure Interaction.

- We will again be using NEEShub for the demonstration. So get an account if you don’t have one! **It’s free to everyone**.