OpenSees Thermomechanical & Project SIF Builder

(A development initiative by OpenSees Research Group at UoE)

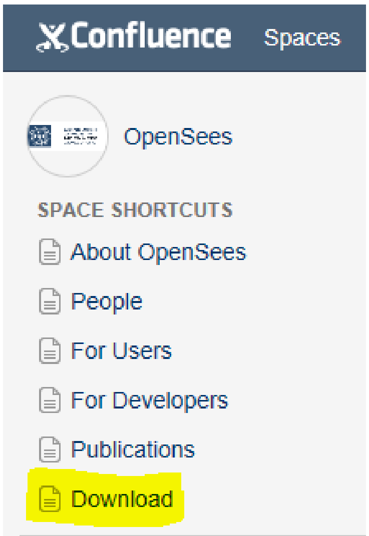
Brief Overview

- **Open System for Earthquake Engineering Simulations** a.k.a **OpenSees** is a software framework for building finite element applications in structural and geotechnical systems.
- OpenSees developers group at the University of Edinburgh is working on Thermo-Mechanical version of OpenSees to facilitate the analysis of Structures in Fire.
- Similar to the OpenSees source codes, the thermo-mechanical analysis codes are written using the Object Oriented Language, **C++**.
- SIF Builder stands for Structures in Fire Builder, which is a project being developed by the research group to analyse the structures under real fire scenarios.

Downloading OpenSees Source Code

Step 1: Click the following link or Copy and Paste it on your browser
https://www.wiki.ed.ac.uk/display/opensees/UoE+OpenSees

Step 2: On the **left pane**, select the option **Download**

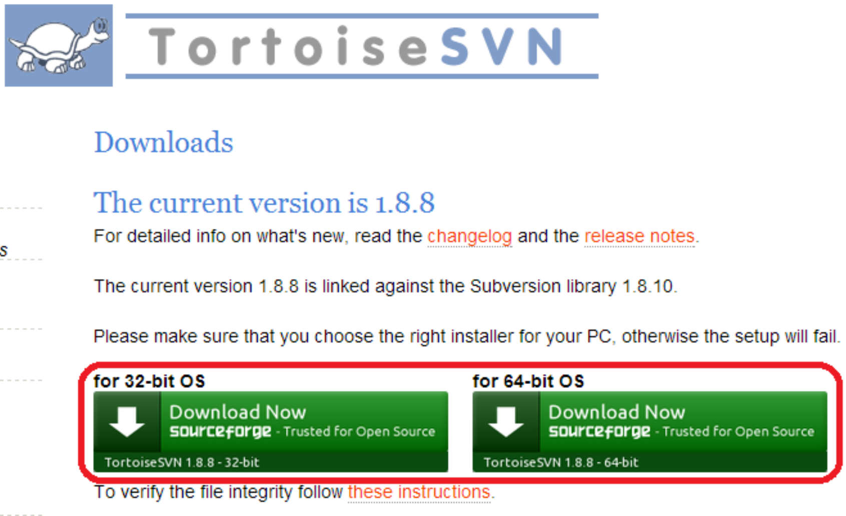


*Note: OpenSees Thermal source codes are stored using **Apache Subversion (SVN)** Software and hence requires a SVN client to download.*

by,
Praveen Kamath
University of Edinburgh
Praveen.Kamath@ed.ac.uk

Step 3: Download the version control / source control software TortoiseSVN, using the link given below *(based on Apache™ Subversion (SVN)®)*, depending on whether your version of windows is 32 bit / 64 bit.

http://tortoisesvn.net/downloads.html



TortoiseSVN

Info

About
About TortoiseSVN

Screenshots
Screenshots of various dialogs

Testimonials
What users say about TortoiseSVN

News Archive
News archive

Support

FAQ
Frequently asked questions

Downloads

The current version is 1.8.8

For detailed info on what's new, read the changelog and the release notes.

The current version 1.8.8 is linked against the Subversion library 1.8.10.

Please make sure that you choose the right installer for your PC, otherwise the setup will fail.

for 32-bit OS

Download Now sourceforge - Trusted for Open Source
TortoiseSVN 1.8.8 - 32-bit

for 64-bit OS

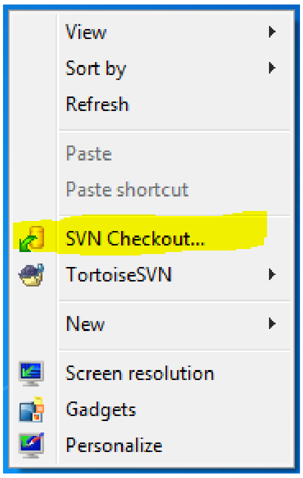
Download Now sourceforge - Trusted for Open Source
TortoiseSVN 1.8.8 - 64-bit

To verify the file integrity follow these instructions.

Note: Clicking download now will take you to the sourceforge.net page and your download should begin automatically. If not, click on the direct link or its mirror to initiate the download.

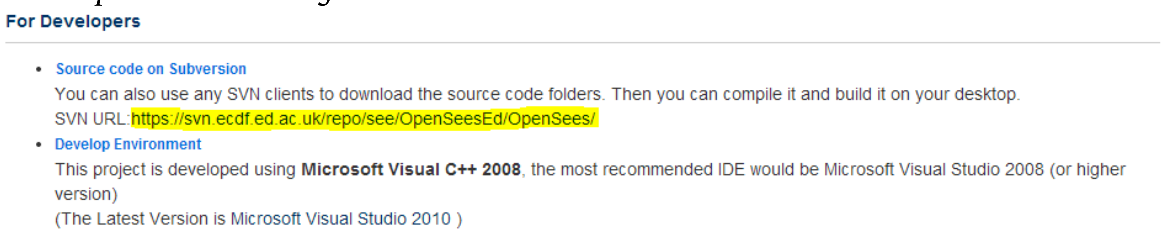
Step 4: Run the downloaded file and install TortoiseSVN™.

Step 5: Go to the folder where you want the source codes to be downloaded and **Right Click anywhere on the screen and select SVN Checkout…**



Step 6: In the checkout window, enter the following URL under the **URL of Repository** field. Choose the checkout depth ***Fully Recursive.***
https://svn.ecdf.ed.ac.uk/repo/see/OpenSeesEd/OpenSees/

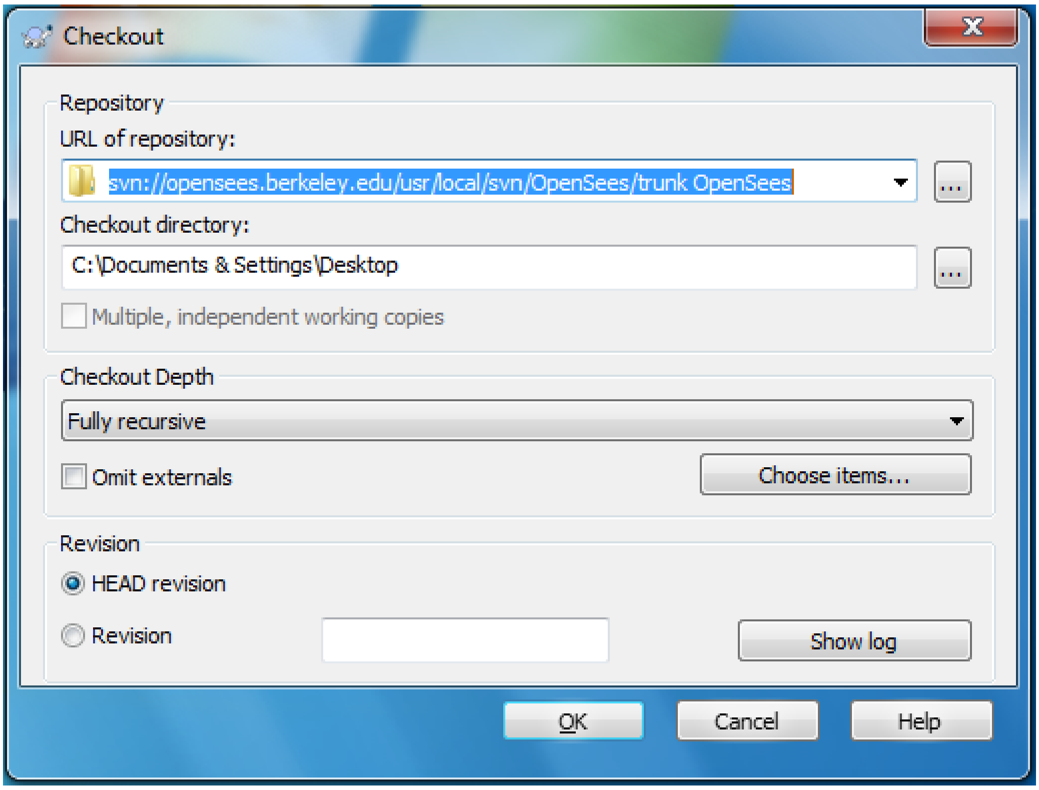
***Link on OpenSees Edinburgh site*



For Developers

- Source code on Subversion
  You can also use any SVN clients to download the source code folders. Then you can compile it and build it on your desktop.
  SVN URL:https://svn.ecdf.ed.ac.uk/repo/see/OpenSeesEd/OpenSees/
- Develop Environment
  This project is developed using **Microsoft Visual C++ 2008**, the most recommended IDE would be Microsoft Visual Studio 2008 (or higher version)
  (The Latest Version is Microsoft Visual Studio 2010)

***SVN Checkout Window*



Checkout

Repository
URL of repository: svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk OpenSees
Checkout directory: C:\Documents & Settings\Desktop
□ Multiple, independent working copies

Checkout Depth
Fully recursive
□ Omit externals

Revision
● HEAD revision
○ Revision

Note: Choose the checkout directory as desired by altering the field under ***Checkout Directory:***

Step 7: Wait until the source code downloads and the checkout action shows **Completed!**

OpenSees Thermomechanical & Project SIF Builder

(A development initiative by OpenSees Research Group at UoE)

OpenSees Source Codes

Brief Overview

The source codes are written in the Object oriented language, C++. The source codes for OpenSees were developed using Microsoft Visual Studio 2008. The source codes can also be compiled using higher versions Microsoft Visual Studio 2010, 2012 and 2013. It is advisable to stick to version 2008 / 2010 to avoid compatibility issues during the conversion of the solution (*.sln) files.

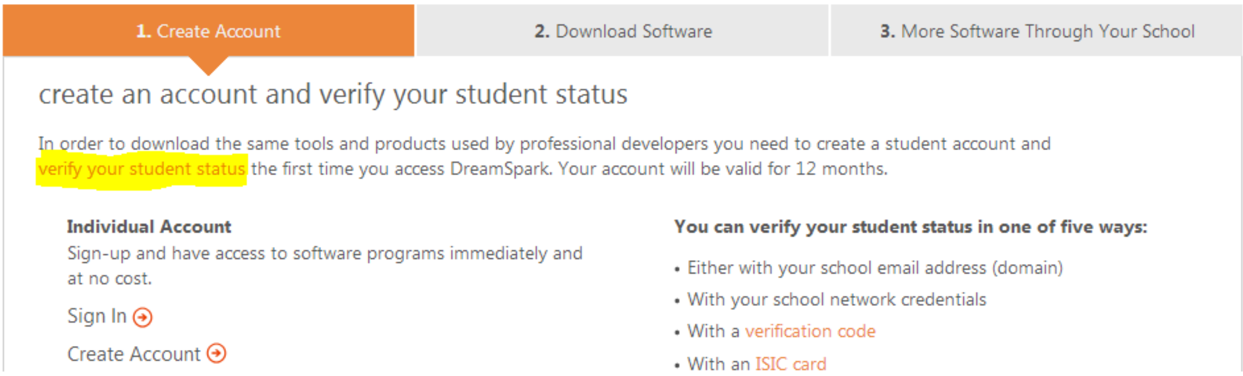
Attention!:
Before beginning the project, make sure to install Microsoft Visual Studio and Tcl/Tk.

Downloading & Installing the Absolute Essentials

Microsoft Visual Studio and Tcl/Tk

All version of Microsoft Visual Studio Professional may be downloaded for FREE using **Microsoft Dreamspark**.

Step 1: Before signing up for an individual account, **verify your student status using your school email Address.**
Note: It is encouraged to merge the dreamspark account with Microsoft Outlook account



create an account and verify your student status

In order to download the same tools and products used by professional developers you need to create a student account and verify your student status the first time you access DreamSpark. Your account will be valid for 12 months.

Individual Account
Sign-up and have access to software programs immediately and at no cost.

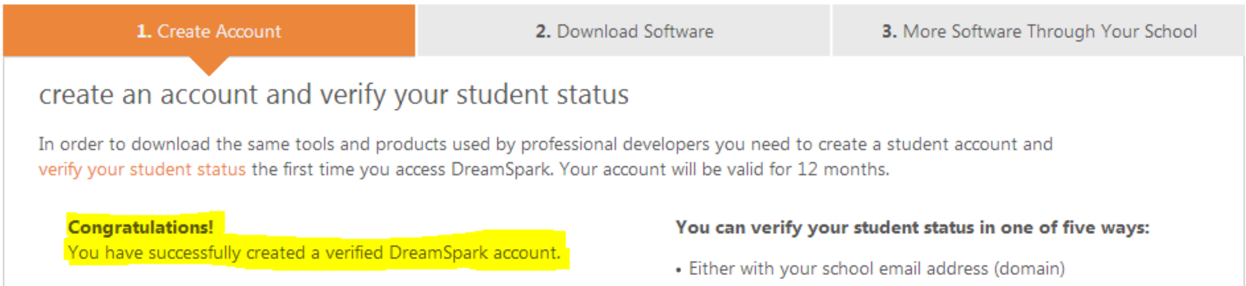
Sign In

Create Account

You can verify your student status in one of five ways:

- Either with your school email address (domain)
- With your school network credentials
- With a verification code
- With an ISIC card

Step 2: Upon completing the verification, you should receive the following greeting.



1. Create Account | 2. Download Software | 3. More Software Through Your School

create an account and verify your student status

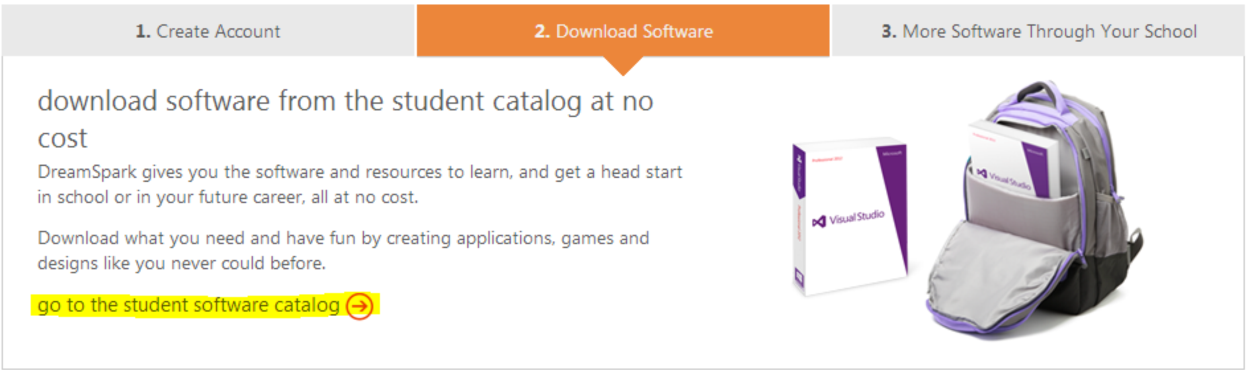
In order to download the same tools and products used by professional developers you need to create a student account and verify your student status the first time you access DreamSpark. Your account will be valid for 12 months.

Congratulations!
You have successfully created a verified DreamSpark account.

You can verify your student status in one of five ways:

- Either with your school email address (domain)

Step 3: Click on **Download Software** tab and then click on go to the **student software catalog**



1. Create Account | 2. Download Software | 3. More Software Through Your School

download software from the student catalog at no cost

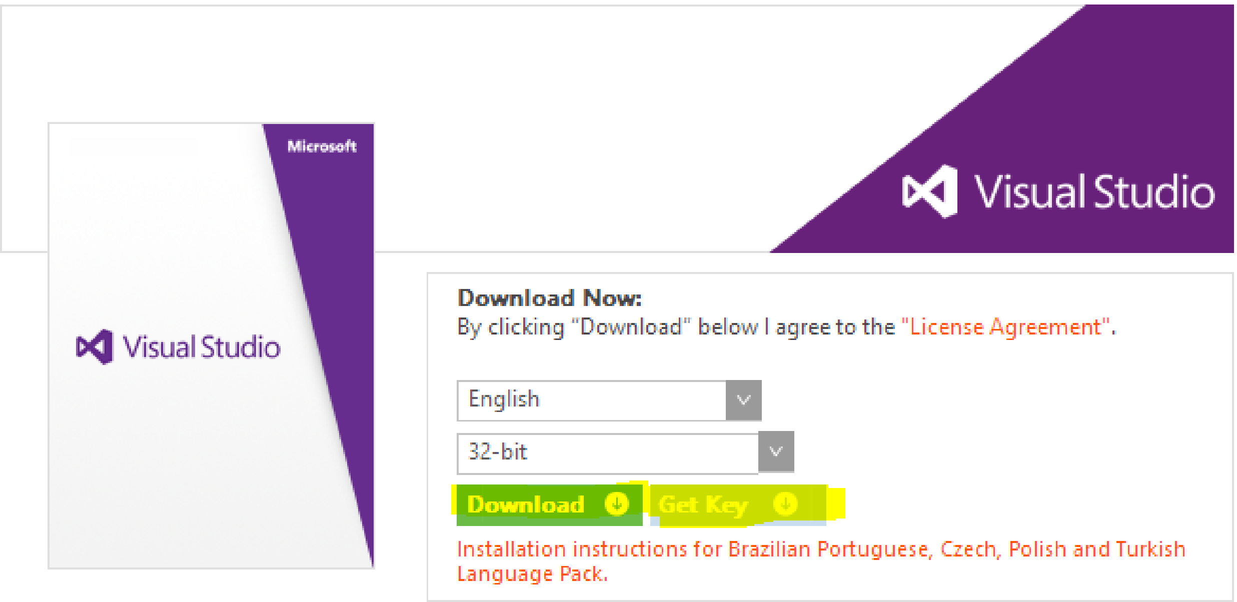
DreamSpark gives you the software and resources to learn, and get a head start in school or in your future career, all at no cost.

Download what you need and have fun by creating applications, games and designs like you never could before.

go to the student software catalog ➔

Step 4: Browse and select for the desired version of **Microsoft Visual Studio Professional** under **Developer & Designer Tools**
Note: Choose ver 2008 / ver 2010.

Step 5: Click on **Get Key** button and note down the 16 digit **product key** before the download



Download Now:

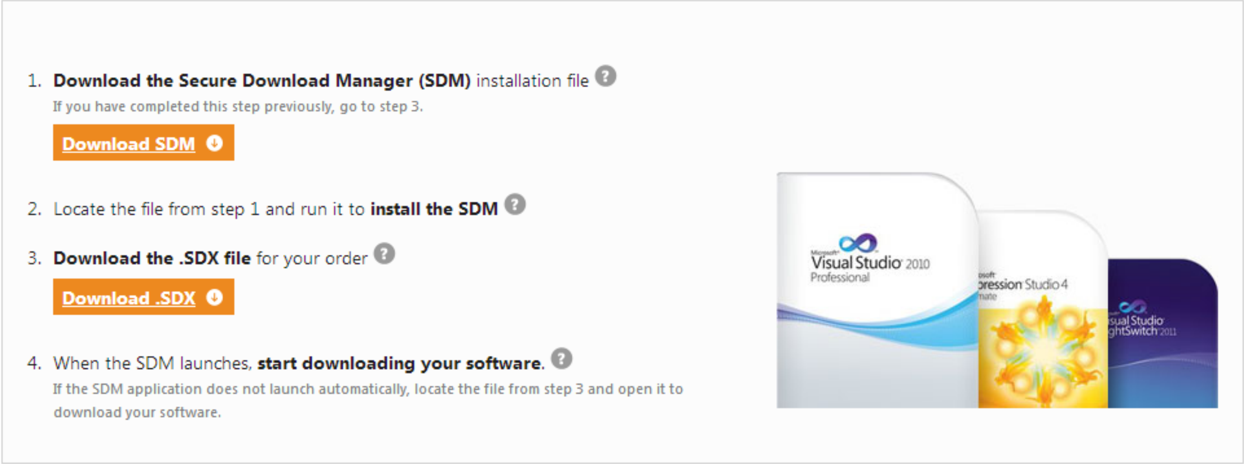
By clicking "Download" below I agree to the "License Agreement".

English

32-bit

Microsoft
Visual Studio
Visual Studio
Download
Get Key
Installation instructions for Brazilian Portuguese, Czech, Polish and Turkish Language Pack.

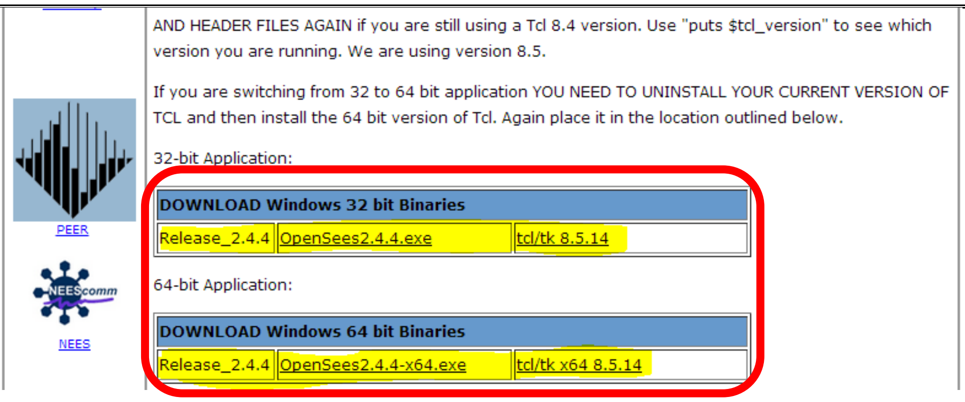
Step 6: Click on **Download** and follow the onscreen instructions to download the software



1. **Download the Secure Download Manager (SDM)** installation file
   If you have completed this step previously, go to step 3.
   Download SDM
2. Locate the file from step 1 and run it to **install the SDM**
3. **Download the .SDX file** for your order
   Download .SDX
4. When the SDM launches, **start downloading your software.**
   If the SDM application does not launch automatically, locate the file from step 3 and open it to download your software.

Step 7: Run the downloaded file to **Install Microsoft Visual Studio Professional.** Note: Keep complete install as the default option.

Step 8: Download tcl/tk X.X.XX or tcl/tk x64 X.X.XX depending on whether your version of windows is 32 bit / 64 bit.



AND HEADER FILES AGAIN if you are still using a Tcl 8.4 version. Use "puts $tcl_version" to see which version you are running. We are using version 8.5.

If you are switching from 32 to 64 bit application YOU NEED TO UNINSTALL YOUR CURRENT VERSION OF TCL and then install the 64 bit version of Tcl. Again place it in the location outlined below.

32-bit Application:

DOWNLOAD Windows 32 bit Binaries		
Release_2.4.4	OpenSees2.4.4.exe	tcl/tk 8.5.14

64-bit Application:

DOWNLOAD Windows 64 bit Binaries		
Release_2.4.4	OpenSees2.4.4-x64.exe	tcl/tk x64 8.5.14

Step 9: Double click on the downloaded version of tcl/tk

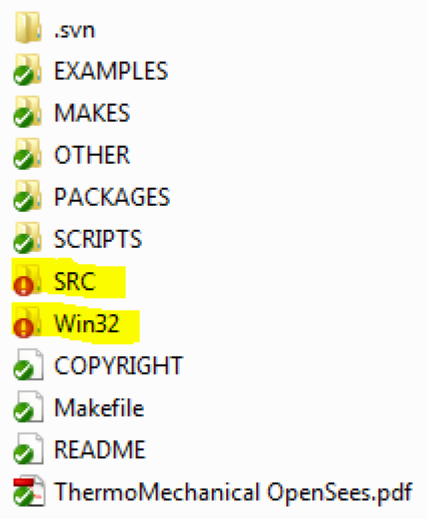
Note: In most of the windows 7 PCs, right click on the file and click ***Run as Administrator***

Step 10: Select Next *(at bottom right)* > **I accept the license agreement and select the Tcl folder for installation**

Note: Make sure you Install Tcl in ***C:\Program Files\Tcl*** *Folder. This folder does not exist by default and hence be created. Also, by default Tcl tends to install in C:\Tcl. This path needs to be changed to C:\Program Files\Tcl.*

Working with the Source Codes

- The source codes downloaded using the subversion client contains the following files and folders.



.svn
EXAMPLES
MAKES
OTHER
PACKAGES
SCRIPTS
SRC
Win32
COPYRIGHT
Makefile
README
ThermoMechanical OpenSees.pdf

- The two folders of interest are **SRC** and **Win32.**
- The source C++ source and header files may be found in the sub folders of the SRC folder whereas, the solution file for the OpenSees Project is located in **Win32** Folder

Step 1: Open the folder containing the **OpenSees project solution file**

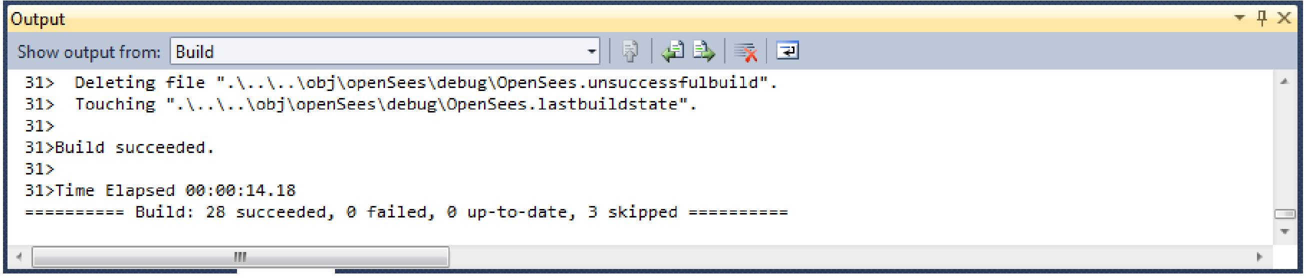
OpenSees/Win32/opensees.sln

Step 2: Double click / run the file **opensees.sln**
Note: For VS versions higher than 2008, the project files needs to be converted into new format. Follow the onscreen instructions and wait until the project files load and appear in the solutions explorer. This may take a few minutes. Also, choose to create a backup of the old solution when prompted.

Step 3: Wait until the project loads and **Ready** appears on the bottom left of the screen. All the projects appear in the solution explorer to the left of the VS screen.
*Note: Check if the main project **OpenSees** is boldfaced. If not, right click on the project and select the option **Set as StartUp Project.***
*Also, check if the list contains the project **SIFBuilder**.*

Step 4: To run the solution, click on **Build** and select the option **Build Solution** or simply press the **F7** on the keyboard and wait until the solution is built. This process may take several minutes.

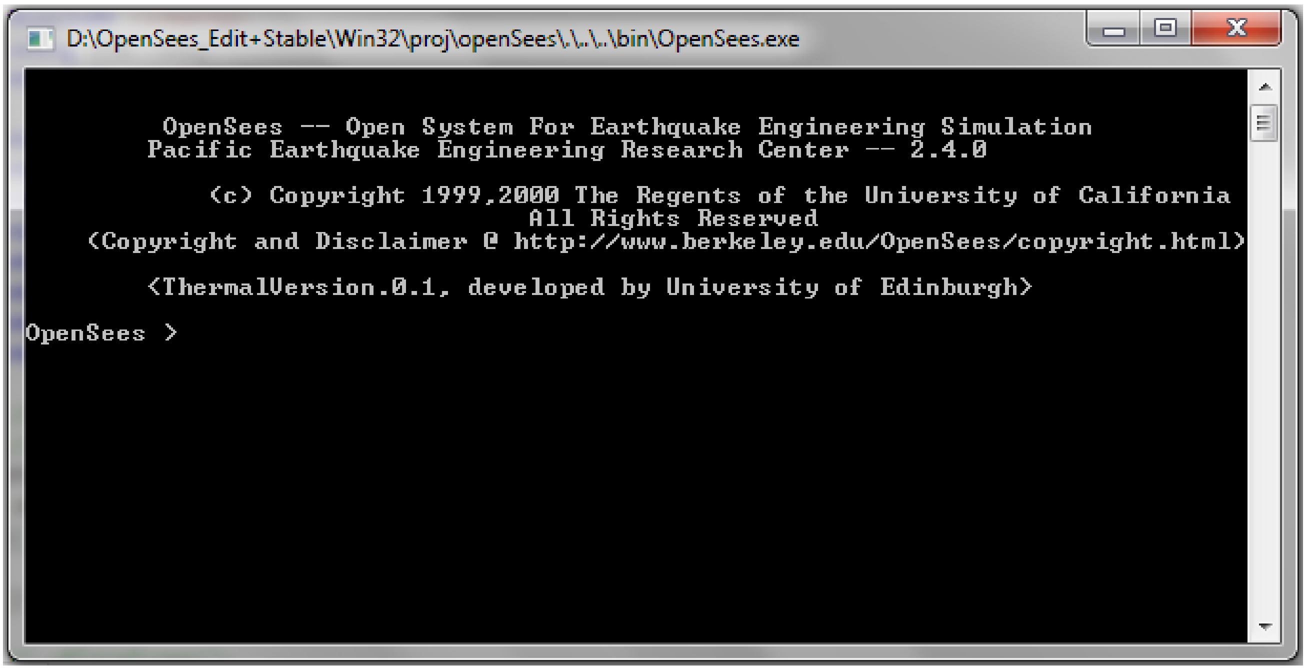
Successful build



```
31>  Deleting file ".\..\..\obj\openSees\debug\OpenSees.unsuccessfulbuild".
31>  Touching ".\..\..\obj\openSees\debug\OpenSees.lastbuildstate".
31>
31>Build succeeded.
31>
31>Time Elapsed 00:00:14.18
========== Build: 28 succeeded, 0 failed, 0 up-to-date, 3 skipped ==========
```

Unsuccessful build

Step 5: Click on **Debug** and select **Start Debugging** or simply press **F5** and wait until **OpenSees command window** opens.


D:\OpenSees_Edit+Stable\Win32\proj\openSees\.\..\..\bin\OpenSees.exe

```
         OpenSees -- Open System For Earthquake Engineering Simulation
        Pacific Earthquake Engineering Research Center -- 2.4.0

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

        (ThermalVersion.0.1, developed by University of Edinburgh)

OpenSees >
```

OpenSees Thermomechanical & Project SIF Builder

(A development initiative by OpenSees Research Group at UoE)

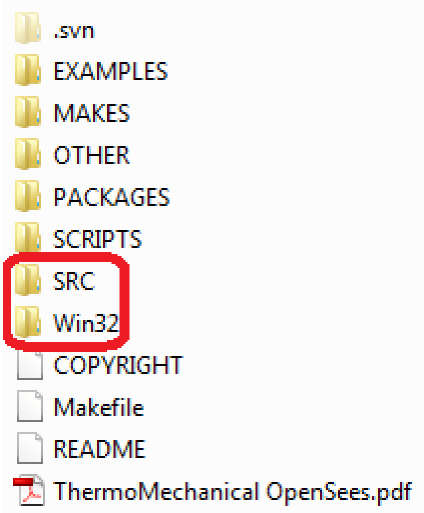
OpenSees PEER NEES NEEScomm

Structures in Fire (SIF) Builder

SIF Builder is a project in OpenSees amongst the list of other projects such as ***actor, analysis, api, . . . material, matrix, model builder, . . ., tcl, unittest, unity***. The following steps show the procedure to add a new **project** and link it with the existing source codes.

Two **MAIN** folders of interest for development of **SIF Builder** are,

1. **OpenSees** > **OpenSees2.4.0** > **SRC** > **SIFBuilder**
2. **OpenSees** > **OpenSees2.4.0** > **Win32** > **projects** > **SIFBuilder**

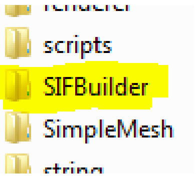


```
.svn
EXAMPLES
MAKES
OTHER
PACKAGES
SCRIPTS
SRC
Win32
COPYRIGHT
Makefile
README
ThermoMechanical OpenSees.pdf
```

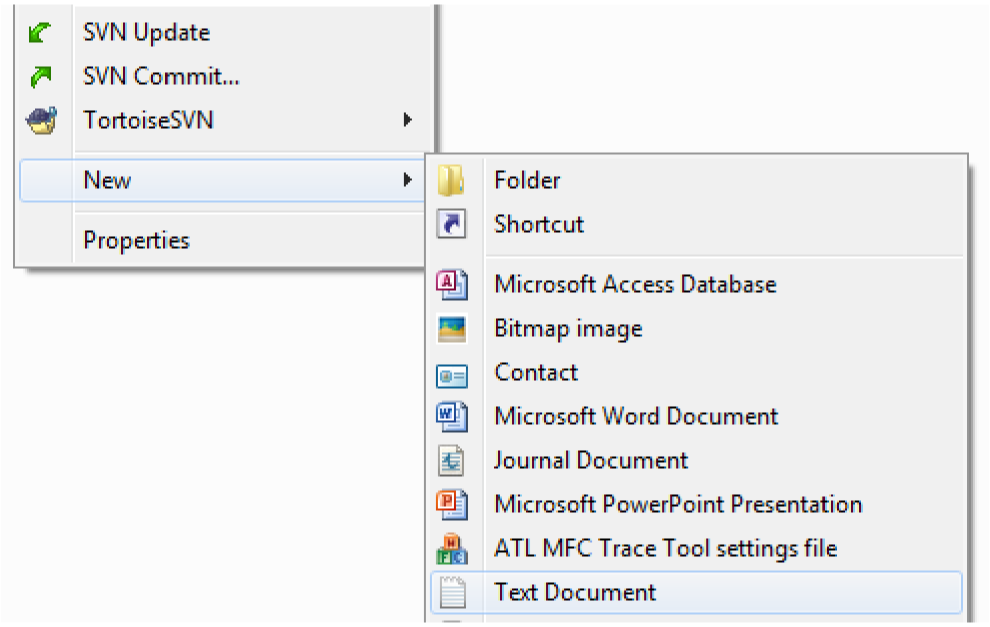
Note: For all development activities, source and header files are added in the former of the two folders.

Adding source codes to the project SIFBuilder

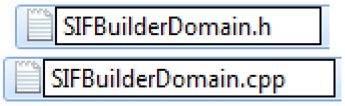
Step 1: Go to the folder OpenSees > OpenSees2.4.0 > SRC > SIFBuilder



Step 2: Create two files using: Right Click > new > text document

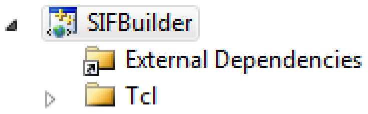


Step 3: Rename one of the files with a *.**h** extension and another with *.**cpp**, which represents **HEADER** and **SOURCE** files respectively.
For example, *SIFBuilderDomain.h* and *SIFBuilderDomain.cpp*



Add these files to the SIFBuilder project folder in Visual C++

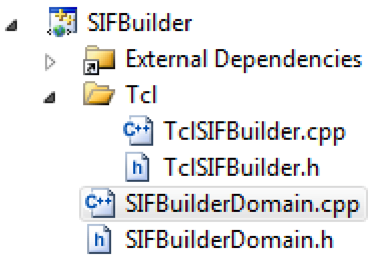
Step 4: Click on the project folder SIFBuilder on the Solution Explorer to expand it.



SIFBuilder
External Dependencies
Tcl

Step 5: Add the created source and header files.
Right click on **SIF Builder** > **Add** > **Existing Item…** > Browse to **OpenSees** » **OpenSees2.4.0** » **SRC** » **SIFBuilder** > select the *.**h** and *.**cpp** file > Click **Add**

Step 6: If *.h file is added to the Tcl folder, **drag and drop** it into the SIFBuilder.



SIFBuilder
External Dependencies
Tcl
TclSIFBuilder.cpp
TclSIFBuilder.h
SIFBuilderDomain.cpp
SIFBuilderDomain.h

Note: *Whenever you add the source and header files to the project, the header file automatically locates itself in the* ***Tcl*** *filter. Drag and drop it to the* ***SIFBuilder*** *filter.*

Working with the development of SIFBuilder

SIFBuilder follows the philosophies of the existing project Model Builder.

Two main set of source and header files are:

1. SIFBuilderDomain.h and SIFBuilderDomain.cpp
2. TclSIFBuilder.h and TclSIFBuilder.cpp

SIFBuilderDomain (header and source) contains the data and procedures of every header and source files added to the project whereas **TclSIFBuilder** (header and source) contains the procedure for tcl commands created to facilitate the user input.

Adding a piece of code to the project…

Whenever you add a piece of code, remember to link it to SIFBuilderDomain, since it holds the current state of the model. Follow the example to add your own piece of code.

SIF_Member.cpp

```
#include <SIFMember.h> //include line for header file

SIFMember::SIFMember(int tag, double jt1, double jt2, double
gamma):TaggedObject(tag),Jt(0), Gam(0) //constructor
{
            Jt = new Vector(2); //defining vector for storing joint ID
            (*Jt)(0) = jt1; //pointer to joint 1
            (*Jt)(1) = jt2; //pointer to joint 2
            Gam = new Vector(1); // defining vector for storing skew angle
            (*Gam)(0) = gamma; //pointer to skew angle
}

SIFMember::~SIFMember() //destructor
{
      //
}
```

SIF_Member.h

```
#ifndef SIFMember_h //code to define a macro-name (if !defined)
#define SIFMember_h //code to define a pre-processor macro

#include <TaggedObject.h> //pre-processor directive for TaggedObject header file
#include <MovableObject.h> //pre-processor directive for MovableObject header
file
#include <Vector.h> //pre-processor directive for Vector header file
#include <ID.h> //pre-processor directive for ID header file

class ID; //class for ID
class Vector; //class for Vector

class SIFMember: public TaggedObject //constructor for the class SIFMember
{
public:
      SIFMember(int tag, double jt1, double jt2, double gamma);


      virtual ~SIFMember (); //destructor for the class SIFMember
    //virtual int GetEntityTypeTag();
      //int cTag;


      virtual void  Print(OPS_Stream&, int = 0) {return;}; //printing output
information to the stream

private:
    Vector* Jt; //defining vector for joint
      Vector* Gam; //defining vector for skew angle
};
#endif
```

Changes to be made after adding a piece of code…

After adding a piece of code *(for ex. SIFMember,* necessary changes should be made in the SIFBuilderDomain source and header files.

SIFBuilderDomain.h

1. Include the pre-processor directive for the code added
   eg., #include <SIFMember.h>

2. Include the class of the new code.
   eg., *class SIFMember*

3. Add a constructor with a pointer to tag the class added.
   eg.,
   ```
   int addSIFMember (SIFMember &theMember);
       SIFMember* getSIFMember (int tag);
   ```

4. To the constructor, add a private class to assign a tagged object storage pointer
   eg., *TaggedObjectStorage* the SIFMembers;*

SIFBuilderDomain.cpp

1. To the SIFBuilderDomain constructor, add the array of tagged objects.
   eg., *theSIFMembers = new ArrayOfTaggedObjects(500);*

2. Add a constructor to the newly added code and add an entity and a pointer to access it.
   eg.,
   ```
   int
       SIFBuilderDomain : : addSIFMember (SIFMember &theSIFMember)
           {
           bool result = theSIFMembers -> addComponent (&theSIFMember);
             if (result == true)
                   return 0;
             else {
                   opserr << "SIFBuilderDomain : : addSIFMember () – failed to add
           SIFMember: " << the SIFMember;
             return -1;
                 }
           }
     SIFMember*
    SIFBuilderDomain : : getSIFMember (int Tag)
   {
   TaggedObject *mc = theSIFMembers -> getComponentPtr(tag);
           if (mc == 0);
           return 0;
           SIFMember *result = (SIFMember *)mc;
           return result;
   }
   ```

TclSIFBuilder.h

1. Include the pre-processor directive for the class
   eg., *#include <SIFMember.h>*

2. Add a line in the **TclSIFBuilder** class to access the entity added
   eg., *SIFMember *getSIFMember (int tag);*

TclSIFBuilder.cpp

1. Include the pre-processor directive for the class
   eg., *#include <SIFMember.h>*

2. Add the command for user input
   eg., *int TclSIFBuilderCommand_addSIFMaterial(ClientData clientData, Tcl_Interp *interp, int argc, TCL_Char **argv);*

3. Create the command in the TclSIFBuilder Constructor
   eg., *Tcl_CreateCommand(interp, “AddSIFMaterial”, (Tcl_CmdProc*)TclSIFBuilderCommand_addSIFMaterial,(ClientData)NULL, NULL);*

4. Create a delete command in the destructor for TclSIFBuilder.
   eg., *Tcl_DeleteCommand(theInterp, “AddSIFMaterial”);*

5. Write the procedure for newly created command
   eg.,

```
int
TclSIFBuilderCommand_addSIFMember(ClientData clientData, Tcl_Interp *interp, int
argc,
                        TCL_Char **argv)
{

        if (theSIFDomain == 0) {
          opserr << "WARNING no active SIFBuilder Domain - Storey\n";
        return TCL_ERROR;
  }

        SIFMember* theSIFMember=0;
        int SIFMemberTag = 0;
		int SIFMemberTypeTag = 0;
	int count = 1;

		if(theSIFMember!=0){
			theSIFDomain->addSIFMember(*theSIFMember);
		}
		else
			opserr<<"WARNING: TclSIFBuilderModule fail to add SIFMember: 
	"<<argv[1]<<endln;

		return TCL_OK;

	}
```

After making these changes in the header and source files of SIFBuilderDomain and TclSIFBuilder, rebuild the solution and compile the SIFBuilder with OpenSees.

Useful Links:

OpenSees Webpage

http://opensees.berkeley.edu/

OpenSees Examples Manual

http://opensees.berkeley.edu/OpenSees/manuals/ExamplesManual/HTML/

OpenSees Command Language Manual

http://opensees.berkeley.edu/OpenSees/manuals/usermanual/