CS257
Ant, JUnit, & Tigris

Presented by
Paul Nguyen & Denis Gefter

Overview

- Ant
- JUnit
- Project Build Files & Test Cases
- Tigris Project Site

Ant

- Apache Ant is a Java-based build tool. In theory, it is kind of like *make*, without *make*'s wrinkles. *--Apache Software Foundation*
- Why another make tool?
  - Ant is better than make
  - Java based, so is OS Independent, make relies on OS Shell commands
  - Ant uses XML format; no more ugly make tab problems!

Structure of an Ant Build File

- My First Ant Build File (build1.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<project default=“hello">
   <target name="hello" >
      <antstructure output="ant.dtd"/>
   </target>
</project>
```

Run It!

- ant -file build1.xml

```
C:\>ant -file build1.xml
Buildfile: build1.xml
hello:
BUILD SUCCESSFUL
Total time: 1 second
```

- A file named “ant.dtd” is produced in the current directory. Let’s look at it!

Ant Build File DTD Defines…

- Elements for (amongst others…)
  - Project - document root element
  - Target - can have dependencies
  - Tasks - executed by Ant
  - Properties – a variable in the Ant environment

- Note: The DTD is useful to see what Ant currently recognizes, but can not be used to validate. Developers can extend ant with custom tasks.

Ant Project Element

```
<!ELEMENT project (target | property | taskdef | typedef | %types;)*>
<!ATTLIST project
        name    CDATA #IMPLIED
        default CDATA #REQUIRED
        basedir CDATA #IMPLIED>
```

Ant Target Element

```
<!ELEMENT target (%tasks; | %types;)*>
<!ATTLIST target
        id          ID    #IMPLIED
        name        CDATA #REQUIRED
        if          CDATA #IMPLIED
        unless      CDATA #IMPLIED
        depends     CDATA #IMPLIED
        description CDATA #IMPLIED>
```

Ant Tasks

<!ENTITY % tasks "propertyfile | vsscheckin | sql | cvspass | p4reopen | csc | dirname | wlrun | p4label | p4revert | replaceregexp | get | jjtree | sleep | jarlib-display | dependset | zip | patch | jspc | style | test | tstamp | unwar | vsshistory | icontract | cvschangelog | p4submit | ccmcheckin | p4change | bzip2 | p4delete | vssadd | javadoc | translate | signjar | vajload | jarlib-available | WsdlToDotnet | buildnumber | jpcovmerge | ejbjar | war | rename | sequential | serverdeploy | property | move | copydir | cccheckin | wljspc | fixcrlf | sosget | pathconvert | record | p4sync | exec | p4edit | manifest | maudit | antlr | netrexxc | jpcovreport | execon | ccmcheckout | ant | xmlvalidate | xslt | iplanet-ejbc | ccmcheckintask | gzip | native2ascii | starteam | ear | input | rmic | checksum | mail | loadfile | vsscheckout | stylebook | soscheckin | mimemail | stlabel | gunzip | concat | cab | touch | parallel | splash | antcall | cccheckout | typedef | p4have | filter | xmlproperty | copy | antstructure | ccmcreatetask | rpm | delete | replace | mmetrics | waitfor | untar | loadproperties | available | echoproperties | vajexport | stcheckout | bunzip2 | copyfile | vsscreate | ejbc | unjar | wsdltodotnet | mkdir | condition | cvs | tempfile | junitreport | taskdef | echo | ccupdate | java | renameext | vsslabel | basename | javadoc2 | vsscp | tar | vajimport | setproxy | wlstop | p4counter | ilasm | soscheckout | apply | ccuncheckout | jarlib-resolve | cvstagdiff | jlink | javacc | jarlib-manifest | pvcs | chmod | jar | sound | mparse | blgenclient | uptodate | genkey | javah | ccmreconfigure | fail | unzip | javac | p4add | soslabel | jpcoverage | depend | vssget | deltree | ddcreator">

- NOTE: Look at Ant's Users Manual for information on how to use each task. The generated DTD also defines each of these elements.

Ant Properties

```
<!ELEMENT property (classpath)*>
<!ATTLIST property
        id ID #IMPLIED
        refid IDREF #IMPLIED
        name CDATA #IMPLIED
        classpath CDATA #IMPLIED
        file CDATA #IMPLIED
        userproperty %boolean; #IMPLIED
        resource CDATA #IMPLIED
        environment CDATA #IMPLIED
        prefix CDATA #IMPLIED
        taskname CDATA #IMPLIED
        value CDATA #IMPLIED
        location CDATA #IMPLIED
        description CDATA #IMPLIED>
```

Ant Properties (Cont.)

- Is a variable with a “name” and “value”
- Can be set in build file using the <property> task.
- Can be set outside of Ant
- Can be a System Property from Java i.e. System.getProperties()
- Referenced in task attributes using ${*prop*} in build file.
  Example:
  <javac srcdir="${src}" destdir="${build}"/>

Built-in Ant Properties

- **basedir** the absolute path of the project's basedir (as set with the basedir attribute of <project>).
- **ant.file** the absolute path of the buildfile.
- **ant.version** the version of Ant
- **ant.project.name** the name of the project that is currently executing; it is set in the name attribute of <project>.
- **ant.java.version** the JVM version Ant detected; currently it can hold the values "1.1", "1.2", "1.3" and "1.4".

Hello World Ant!

- Hello Ant Build File (build.xml)

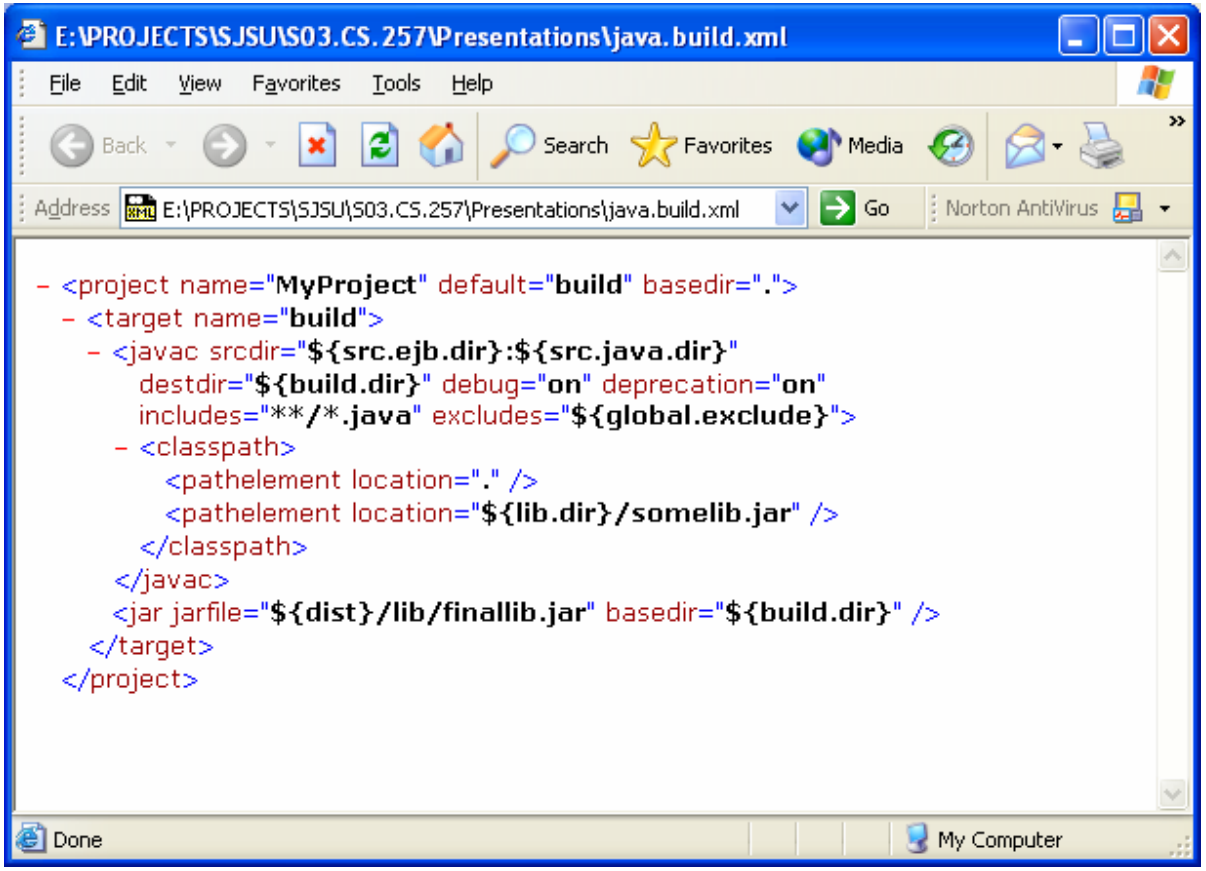
```
<?xml version="1.0" encoding="UTF-8"?>
<project default="build">
    <target name="build" >
     <echo message="Hello, world"/>
    </target>
</project>
```

Run It!

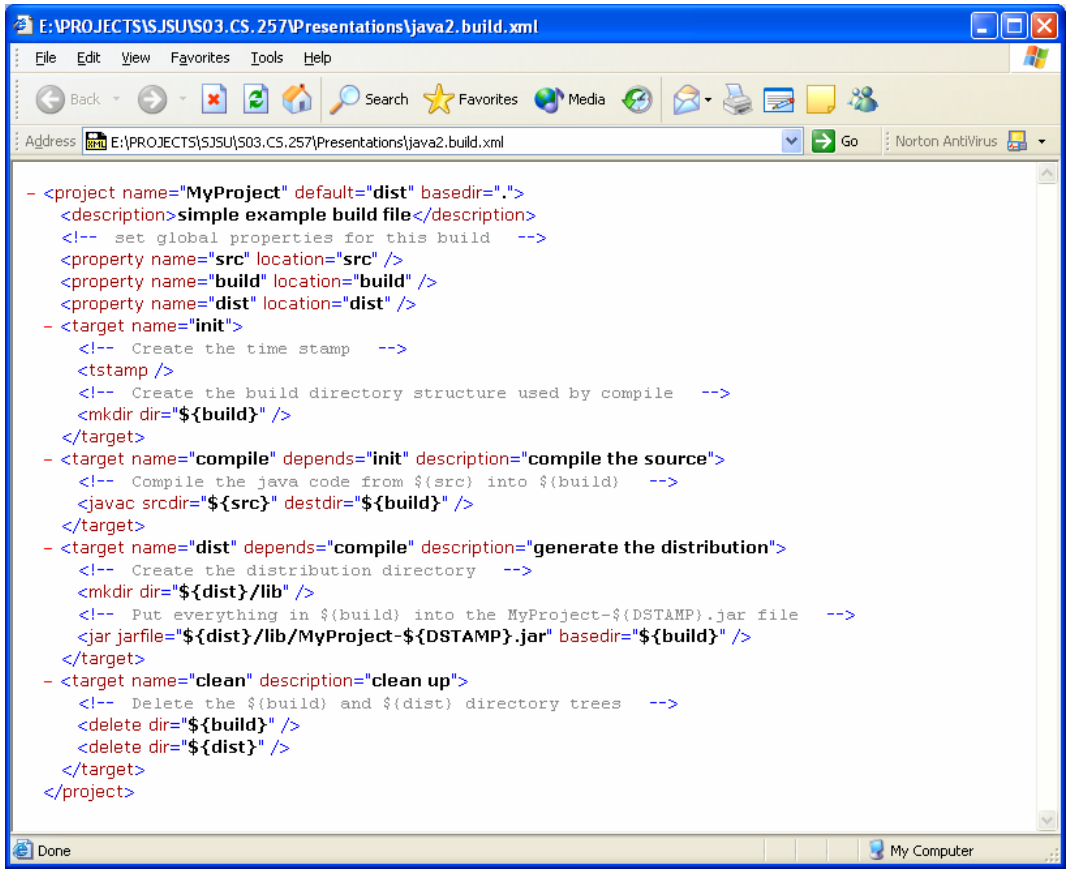
- build.xml file is the default
  - Just run command> ant

```
C:\ ant
Buildfile: build.xml

build:
    [echo] Hello, world

BUILD SUCCESSFUL
Total time: 0 seconds
```

A Simple Java Example



Example From Ant User's Manual



More on Ant… (not covered)

- Data Types
- User Defined Tasks
- Master Build File (ant calling ant)

JUnit

- JUnit is a regression testing framework written by Erich Gamma and Kent Beck. -- *www.junit.org*

JUnit Features

- Assertions for testing expected results
- Test fixtures for sharing common test data
- Test suites for easily organizing and running tests
- Graphical and textual test runners

JUnit Concepts

- A TestCase is a subclass of the TestCase class.
- A TestCase can define any number of public testXXX() methods.
- Use assert() methods to check results (see: **junit.framework.Assert**)

JUnit Concepts (Cont.)

- Fixtures are common objects used by TestCase's public testXXX() methods
  - Initialize using setup()
  - Release using tearDown()
- TestCases can be combined into TestSuite to yield a single passed/failed status

JUnit TestCase Example

```
public class SimpleTest extends TestCase {
        protected int fValue1;
        protected int fValue2;

        protected void setUp() {
                fValue1= 2;
                fValue2= 3;
        }
        public static Test suite() {
                return new TestSuite(SimpleTest.class);
        }
        public void testAdd() {
                double result= fValue1 + fValue2;
                // forced failure result == 5
                assertTrue(result == 6);
        }
        public void testDivideByZero() {
                int zero= 0;
                int result= 8/zero;
        }
        public void testEquals() {
                assertEquals(12, 12);
                assertEquals(12L, 12L);
                assertEquals(new Long(12), new Long(12));

                assertEquals("Size", 12, 13);
                assertEquals("Capacity", 12.0, 11.99, 0.0);
        }
        public static void main (String[] args) {
                junit.textui.TestRunner.run(suite());
        }
}
```

Web Resources

- http://ant.apache.org/
- http://www.junit.org

SM Build and Testing

- Presentation by Denis Gefter
  - Directory Structure of CS257 Project
  - Ant Build Files
  - Running the Test Cases against the Reference Implementation

Tigris.org

- Tigris.org is a mid-sized open source community focused on building better tools for collaborative software development.
- Tigris.org has a policy of allowing university class projects to be hosted. These “student” projects are an exception to their typical rule of approving only open source projects.

Student Project Category

- “This category is for students to host their class projects. Projects can be open or proprietary based on the instructor's rules for the course. After the end of the course, projects will be archived and removed, or made into open source projects.” -- *tigris.org*

Public vs Private Projects

- Projects on Tigris.org can be Private or Public.
  - Private projects require membership to view and contribute.
  - Public projects are visible to the general public.

Tigris.org for CS 257

- Each team should create their own Private project.
- We are awaiting approval for the course shared project.
- To be named: sjsu-cs257 (http://sjsu-cs257.tigris.org/)

  --still need to finalize a few details with Johnny, but should be coming soon!

Naming Convention?

- We currently have one team project on Tigris.org named:

  sjsu-cs257-team1

- Could use this.name++ or make up your own.

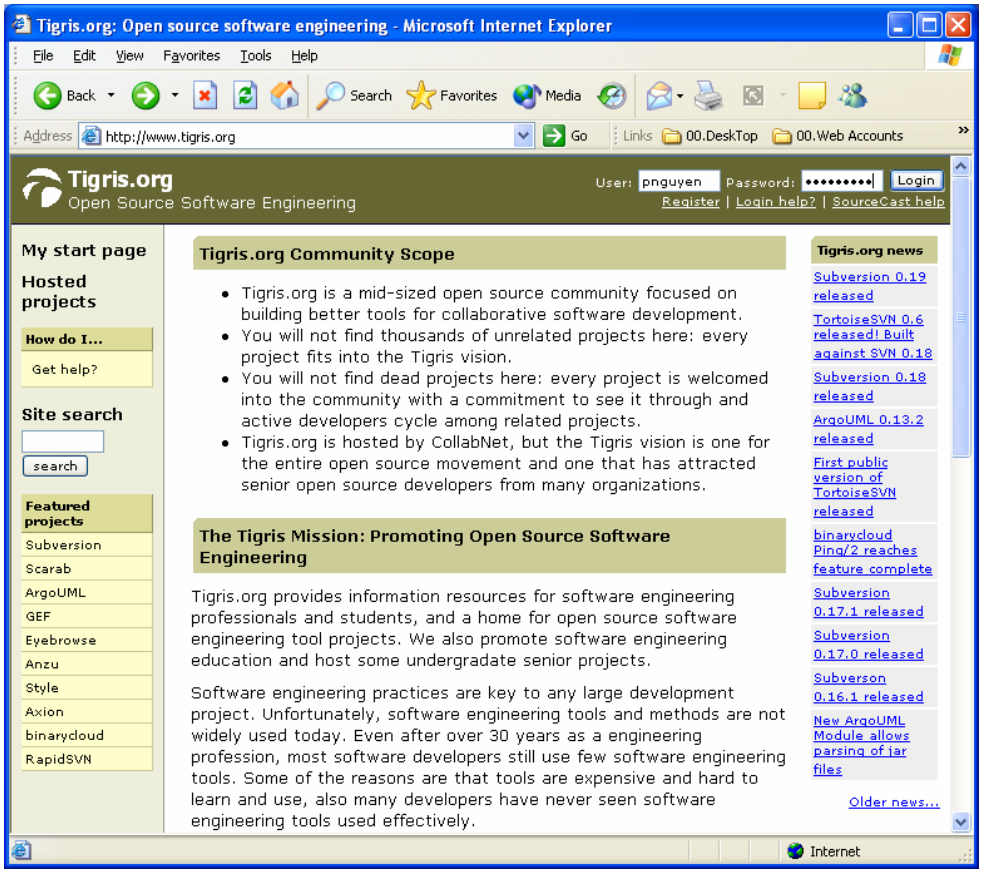
Hosted Tools

- Each Project has the following web based tools:
  - Membership
  - Mailing lists
  - Source code (CVSWeb)
  - Issue tracking
  - File sharing
  - News

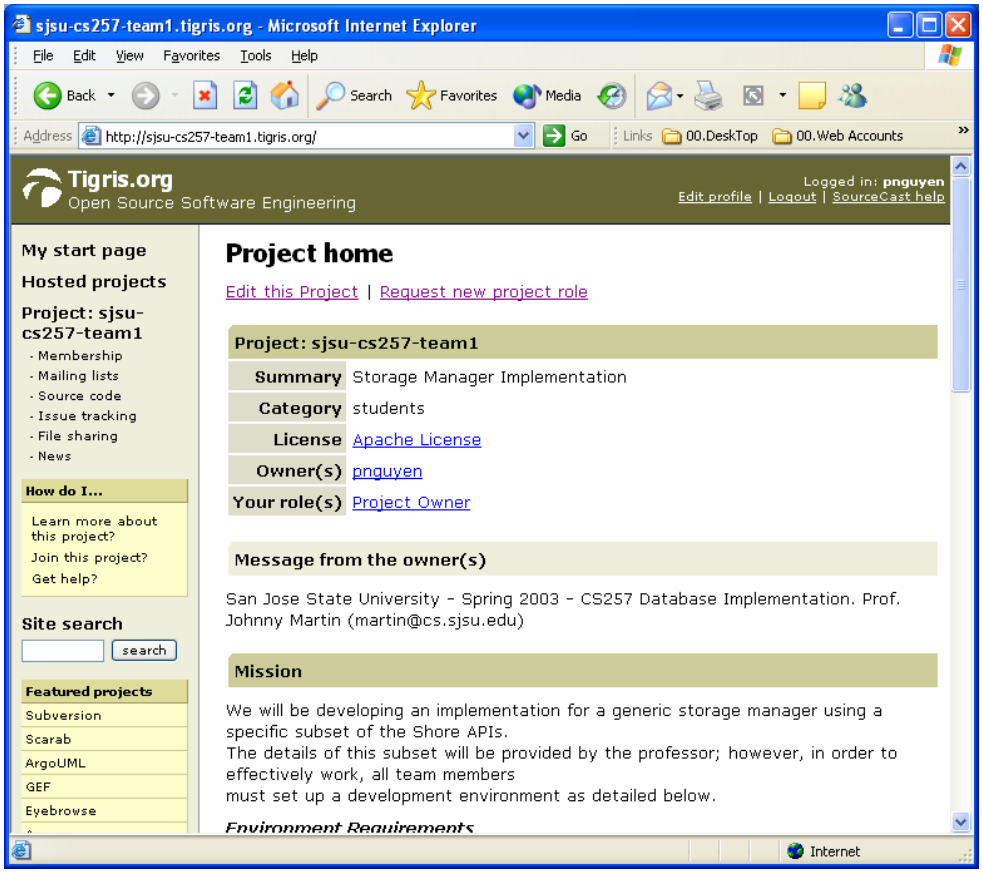
Private CVS Repository

- Each Project also has a CVS repository only accessible to project members—you must login.
- Access is via “PSERVER”, SSH Tunnel recommended, but not required.

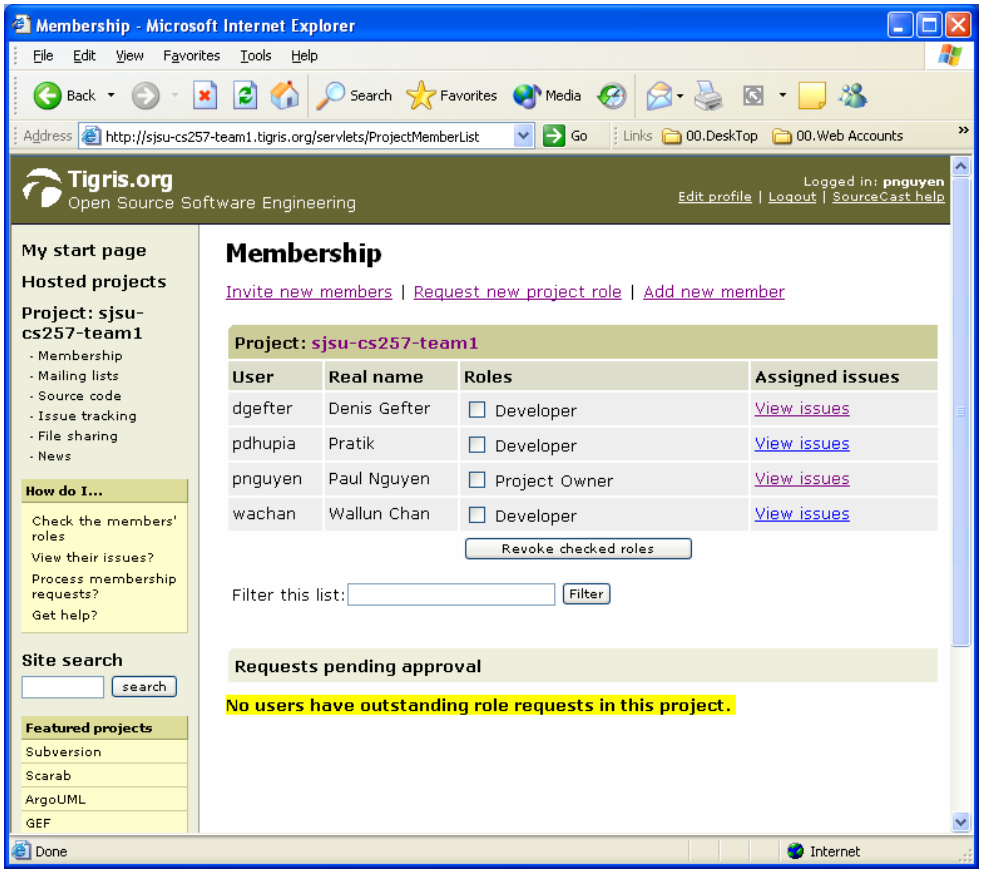
Tigris.org


Tigris.org: Open source software engineering - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.tigris.org
Go Links 00.DeskTop 00.Web Accounts
Tigris.org
Open Source Software Engineering
User: pnguyen Password: Login
Register | Login help? | SourceCast help
My start page
Hosted projects
How do I...
Get help?
Site search
search
Featured projects
Subversion
Scarab
ArgoUML
GEF
Eyebrowse
Anzu
Style
Axion
binarycloud
RapidSVN
Tigris.org Community Scope
Tigris.org is a mid-sized open source community focused on building better tools for collaborative software development.
You will not find thousands of unrelated projects here: every project fits into the Tigris vision.
You will not find dead projects here: every project is welcomed into the community with a commitment to see it through and active developers cycle among related projects.
Tigris.org is hosted by CollabNet, but the Tigris vision is one for the entire open source movement and one that has attracted senior open source developers from many organizations.
The Tigris Mission: Promoting Open Source Software Engineering
Tigris.org provides information resources for software engineering professionals and students, and a home for open source software engineering tool projects. We also promote software engineering education and host some undergradate senior projects.
Software engineering practices are key to any large development project. Unfortunately, software engineering tools and methods are not widely used today. Even after over 30 years as a engineering profession, most software developers still use few software engineering tools. Some of the reasons are that tools are expensive and hard to learn and use, also many developers have never seen software engineering tools used effectively.
Tigris.org news
Subversion 0.19 released
TortoiseSVN 0.6 released! Built against SVN 0.18
Subversion 0.18 released
ArgoUML 0.13.2 released
First public version of TortoiseSVN released
binarycloud Ping/2 reaches feature complete
Subversion 0.17.1 released
Subversion 0.17.0 released
Subverson 0.16.1 released
New ArgoUML Module allows parsing of jar files
Older news...
Internet

Project Home Page



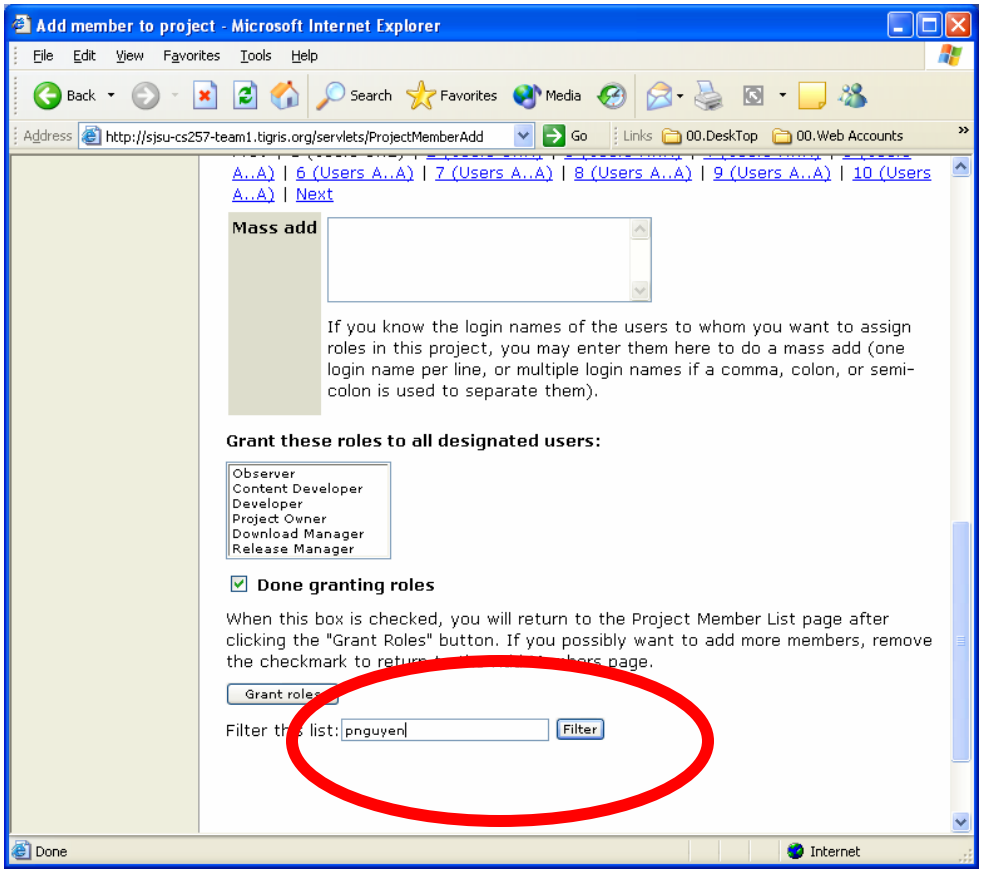
Project Members



Add new member - Find

NOTE:
To add a new member, the project owner finds and selects the tigris account to add.

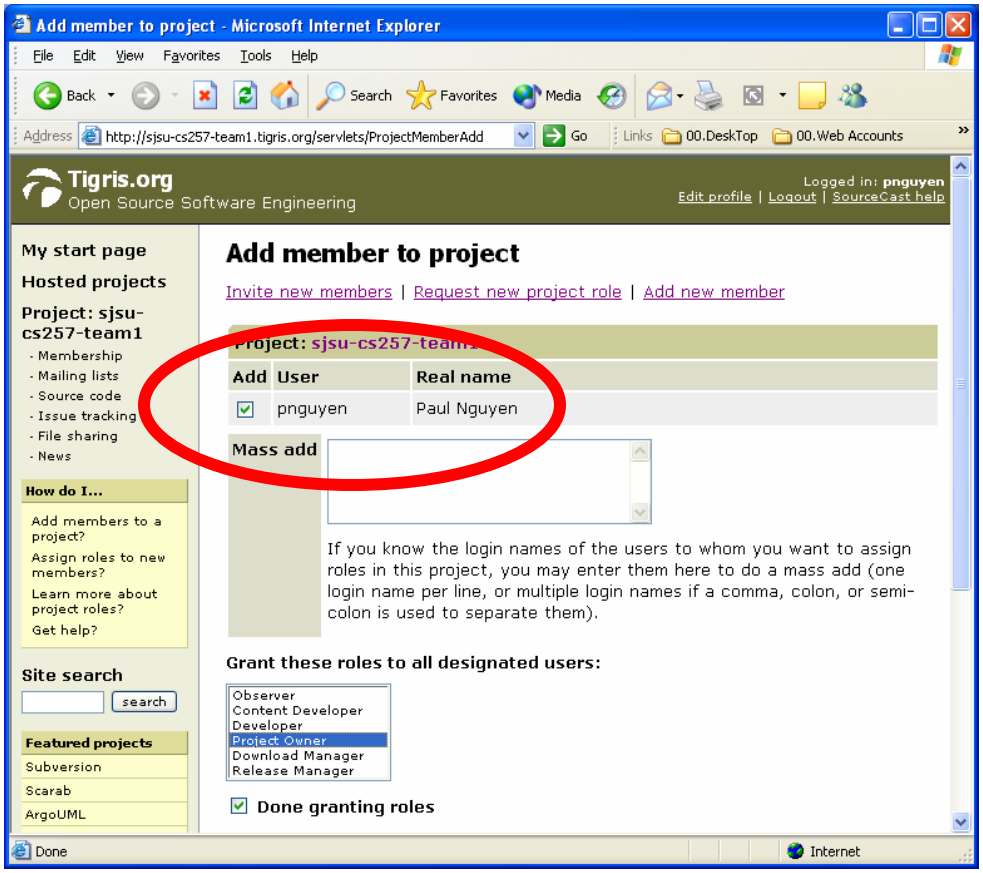
So, team members must first create and account on Tigris and send the username to the project owner



Before you can add members, the project must be approved.

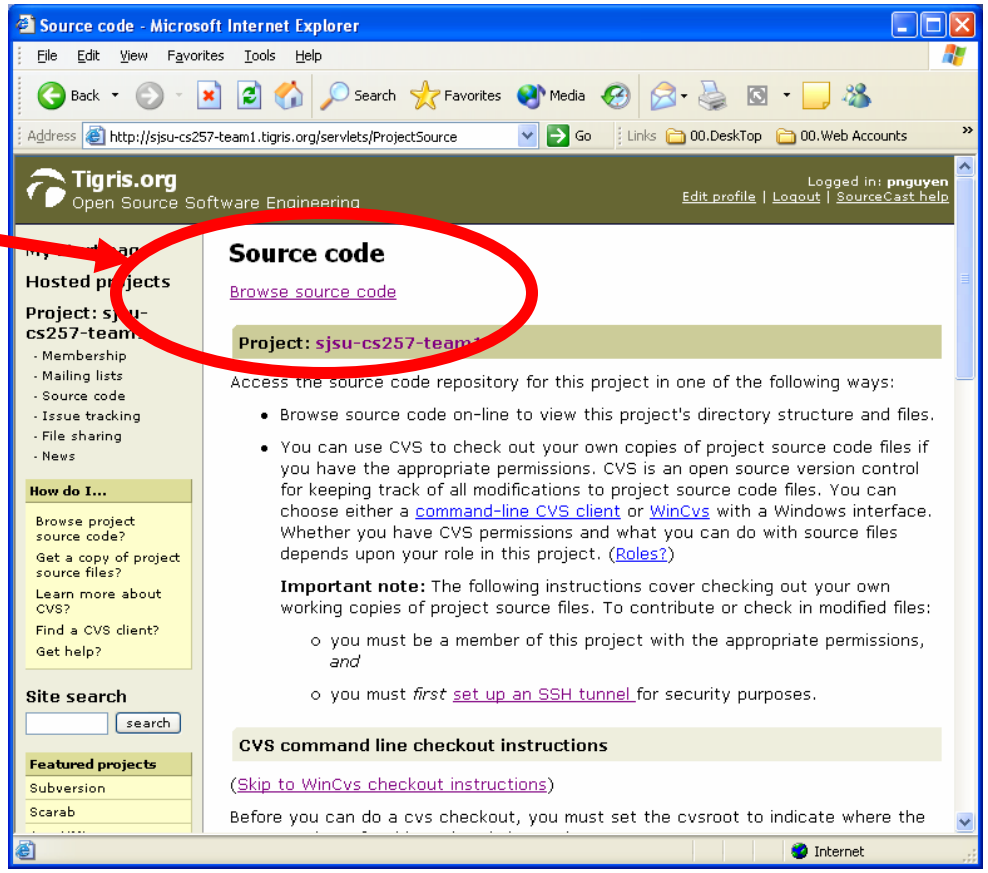
Send an email to: jrobbins@tigris.org

Add New Member - Select

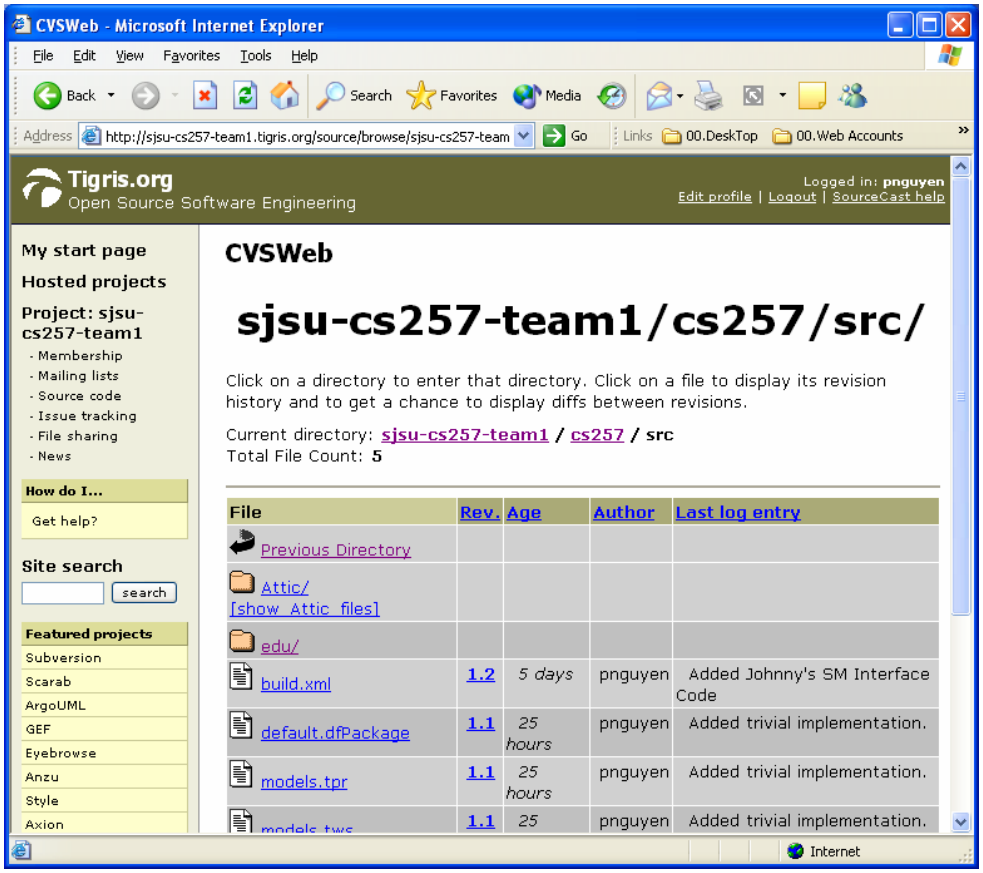


Browsing CVS Files

CLICK HERE



Browsing CVS Files



Connecting to CVS

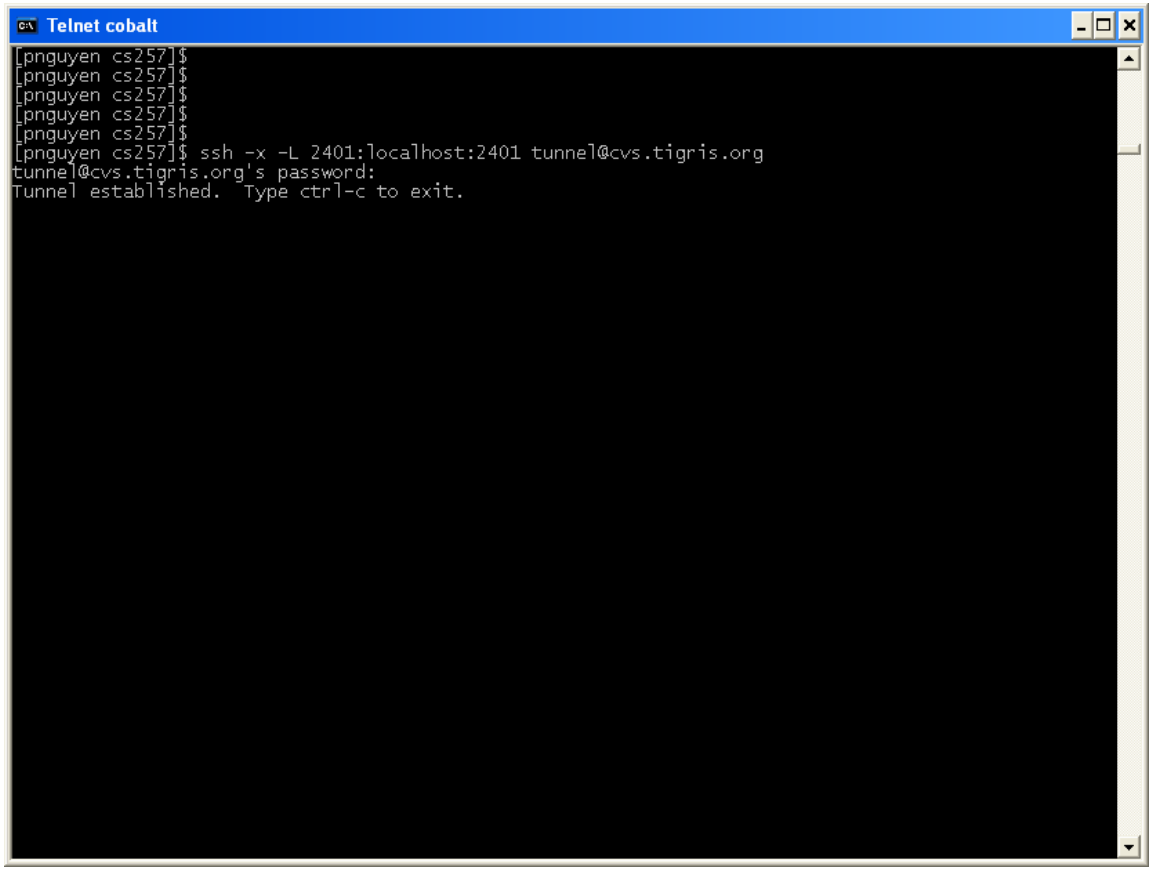
- Create a SSH Tunnel.

Use this command on Linux and Windows (Cygwin):

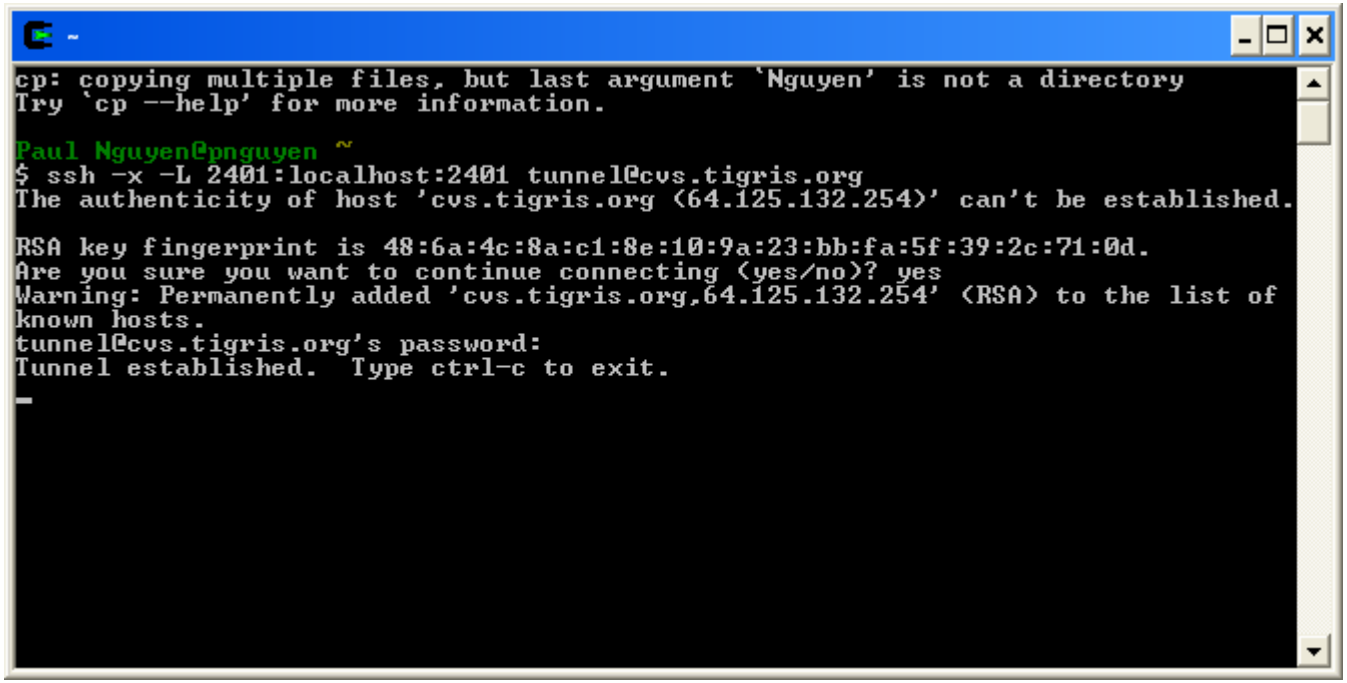
ssh -x -L 2401:localhost:2401 tunnel@cvs.tigris.org

It will ask for a password. Type in: “tunnel”

Linux SSH Tunnel



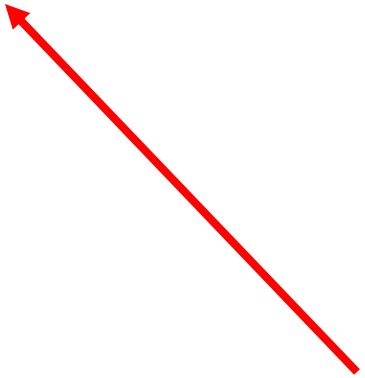
Windows SSH Tunnel



Login to CVS & Check Out

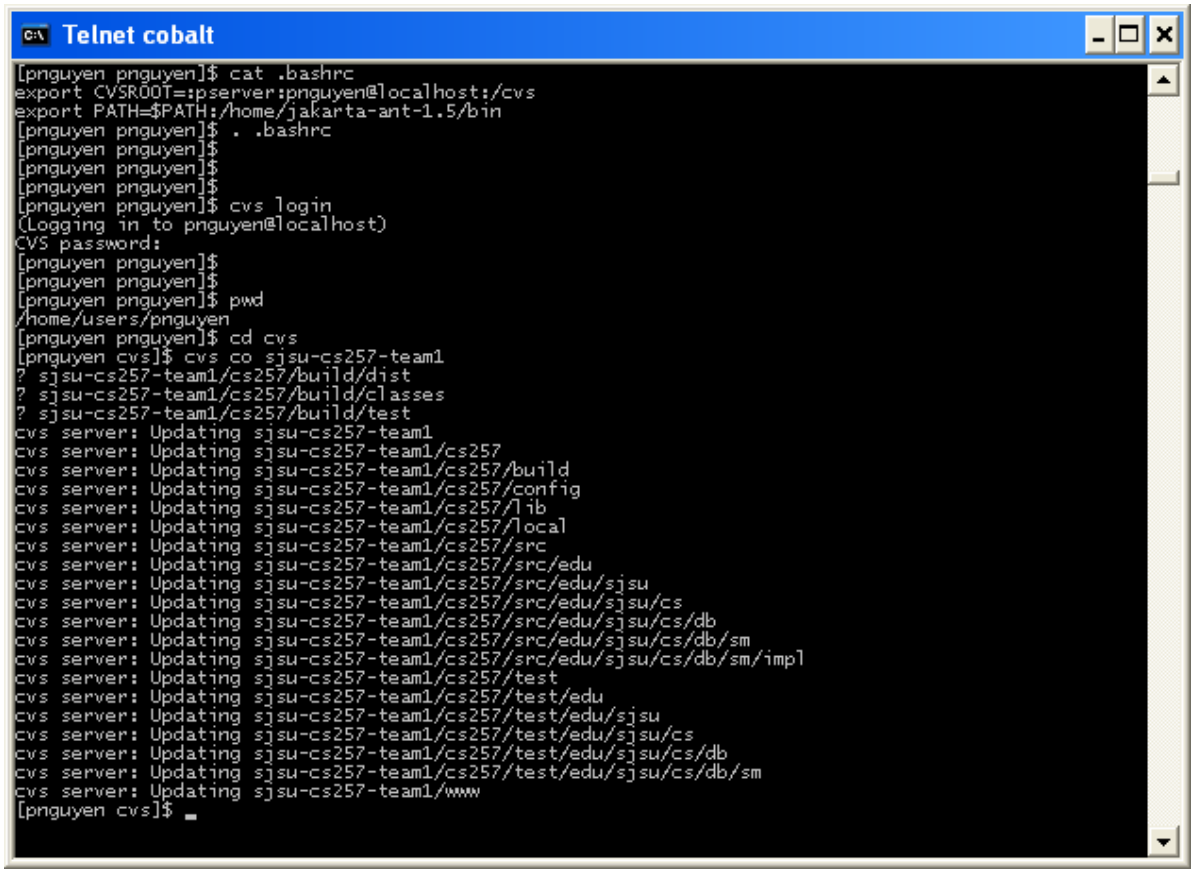
export CVSROOT=:pserver:pnguyen@localhost:/cvs

cvs login
cvs co *sjsu-cs257-team1*

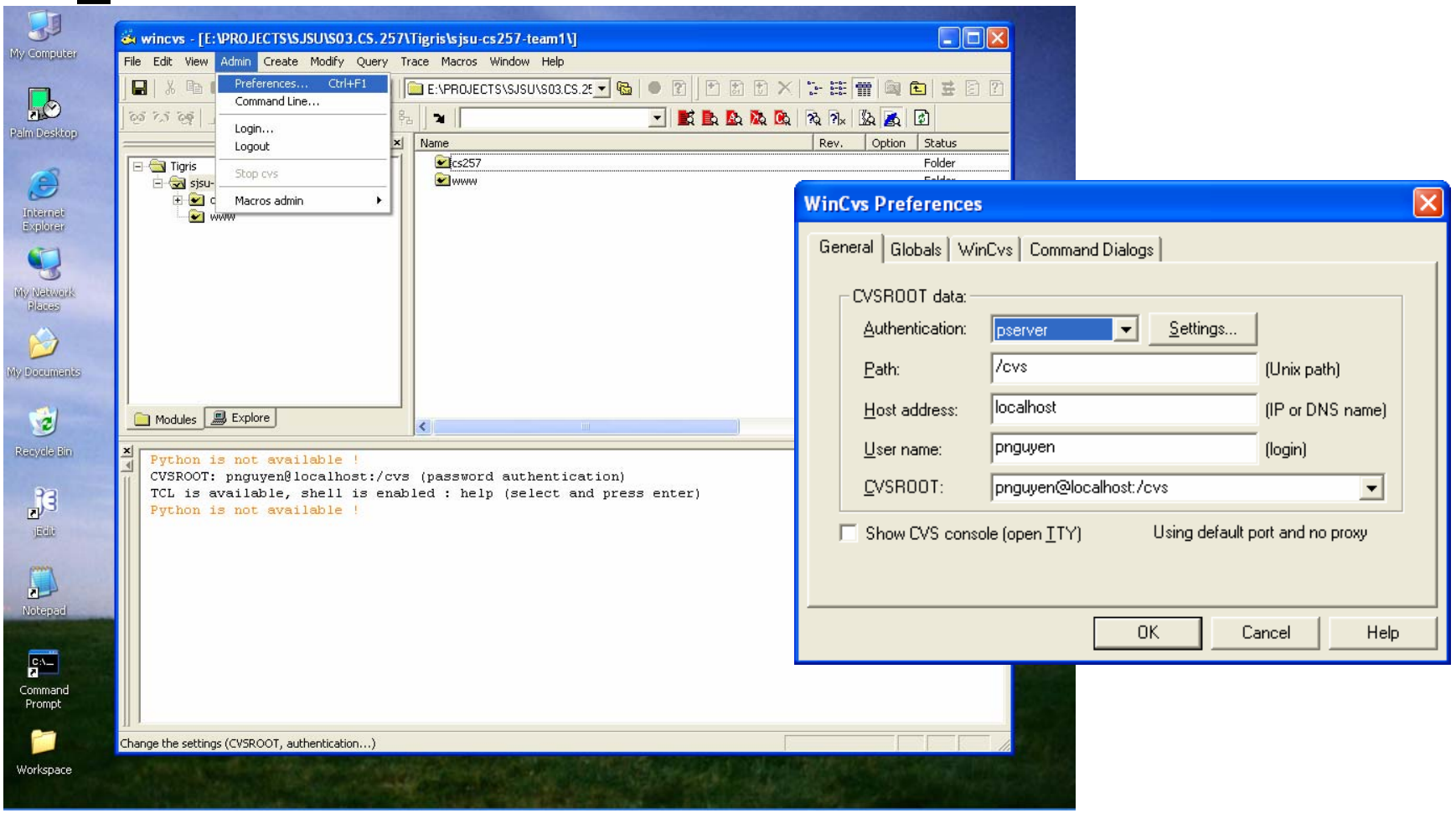


Use the name of your project!

Linux CVS Login & Check Out

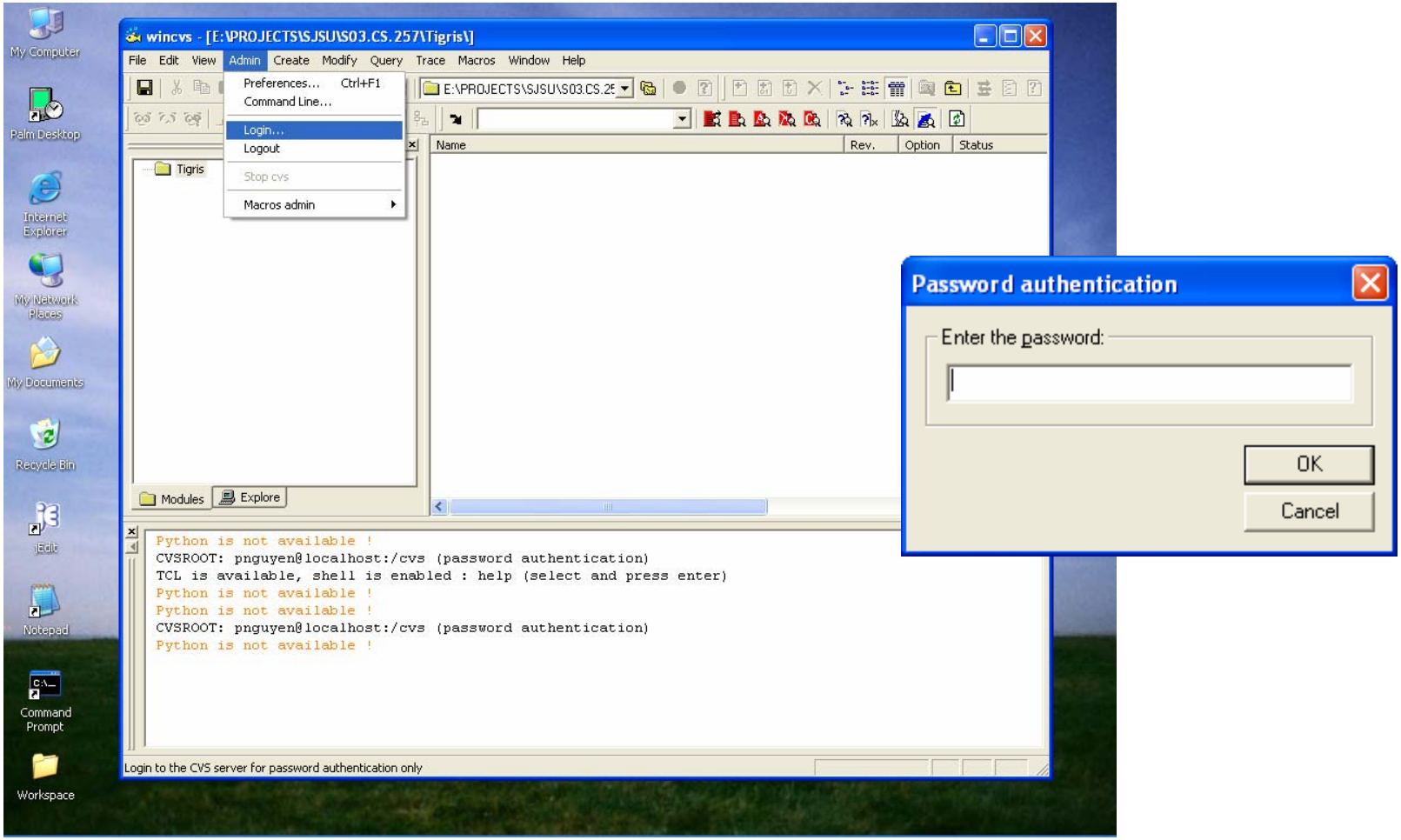


Windows CVS – Set CVSROOT

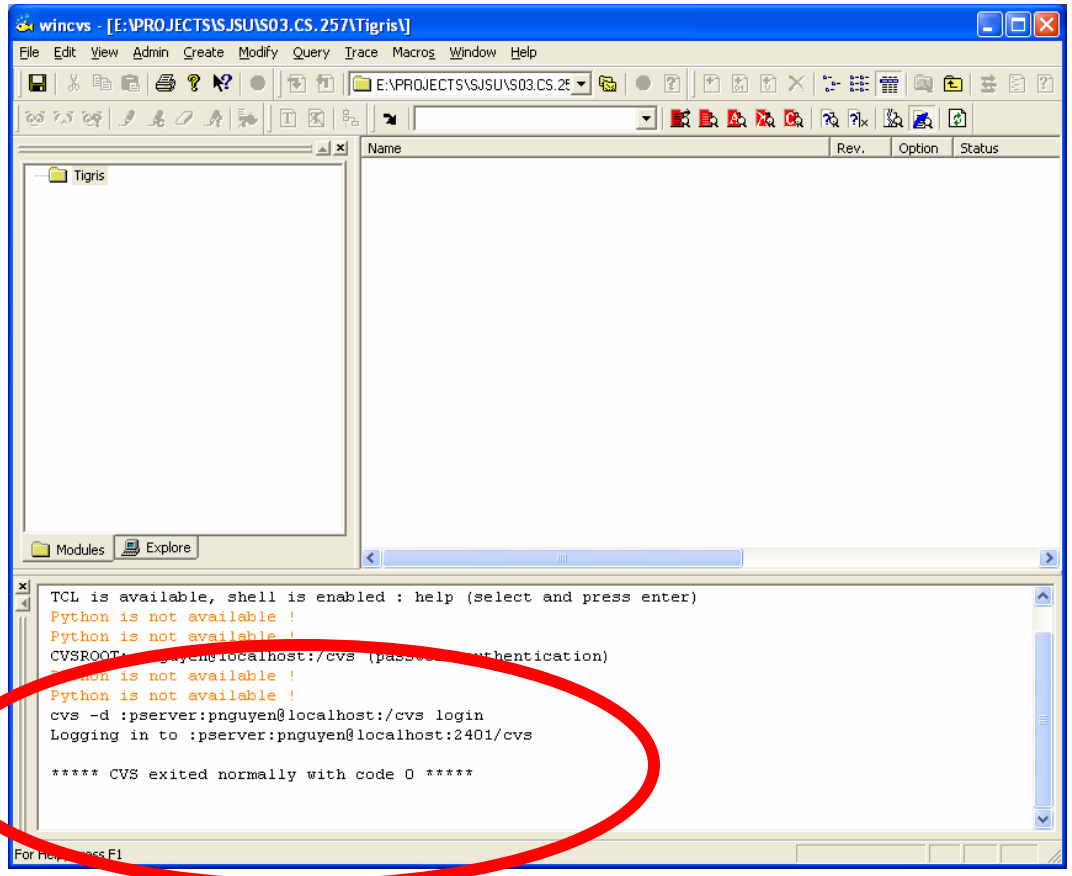


Using WinCVS (a GUI for Windows)

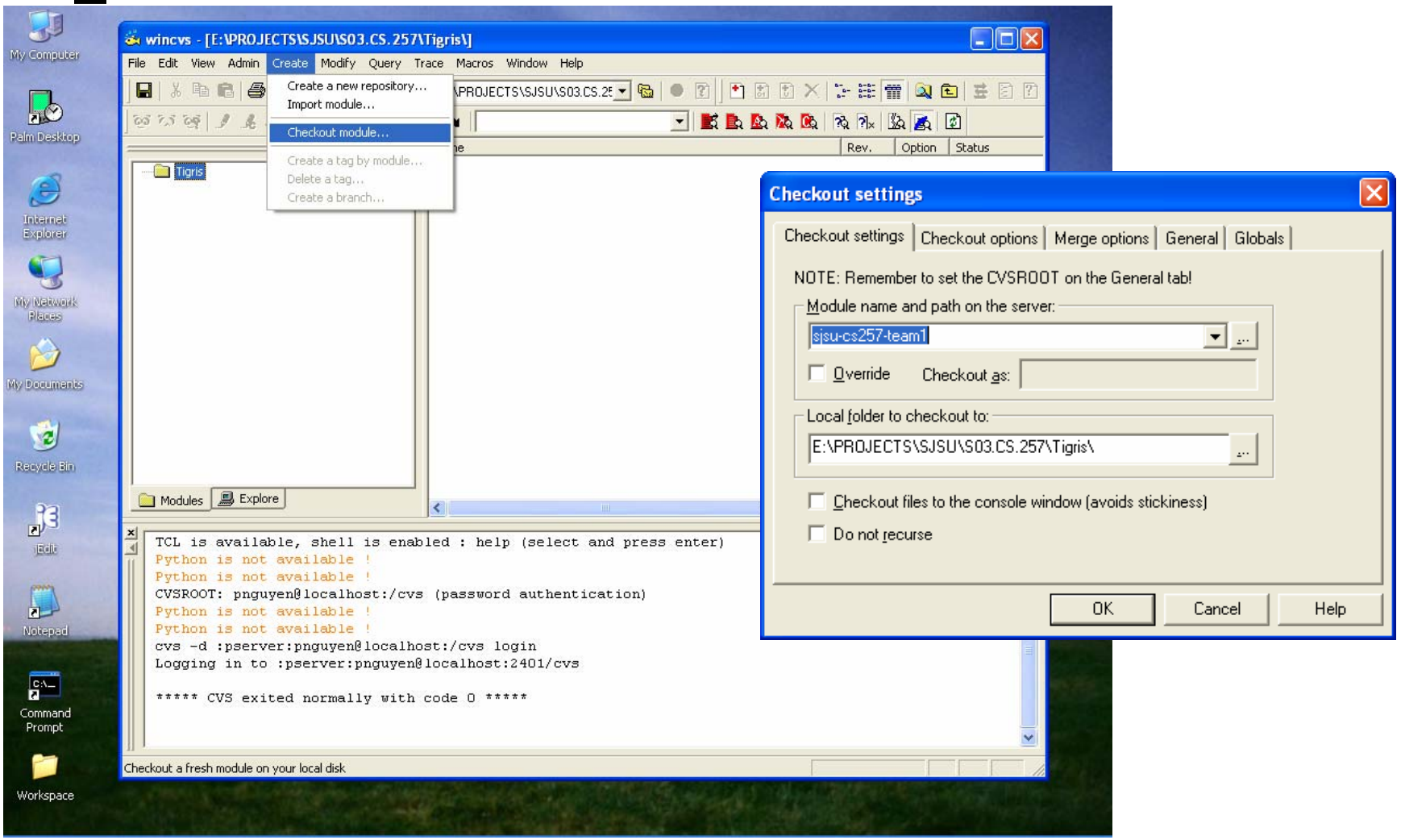
Windows CVS – Login



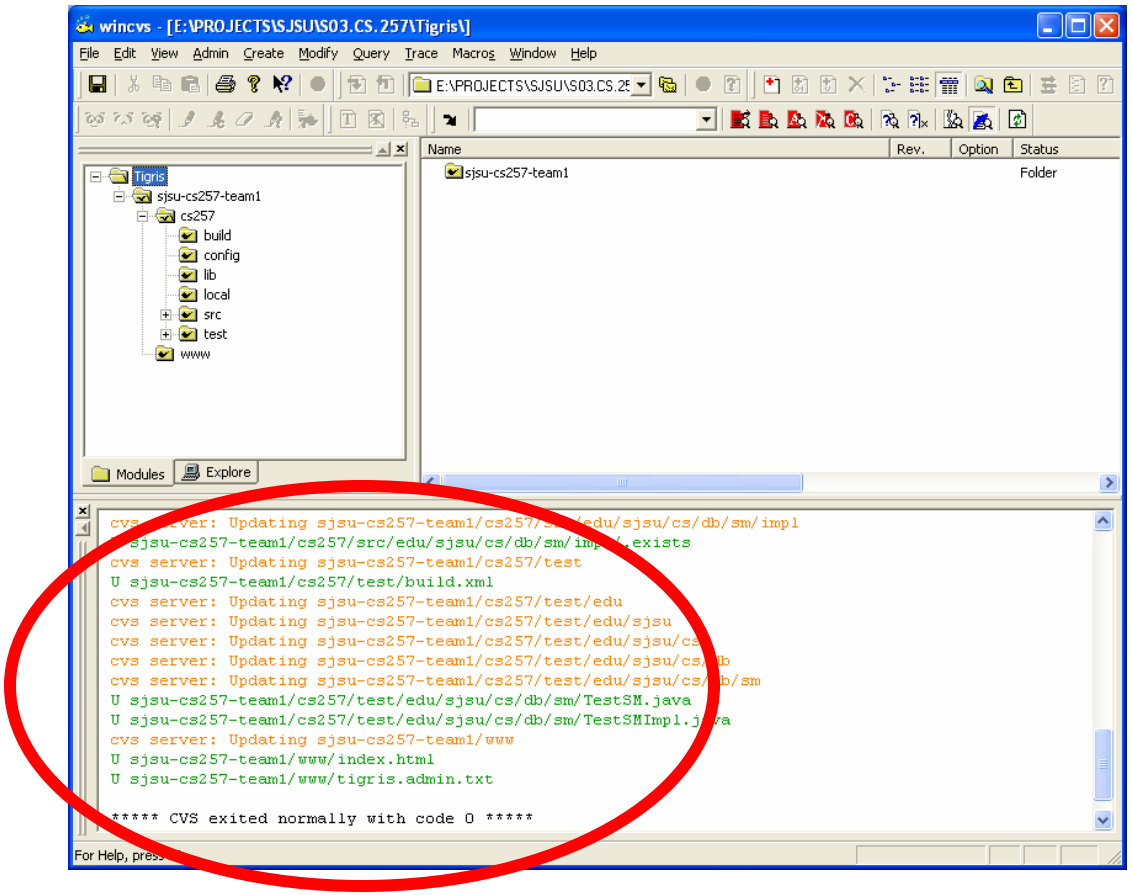
Windows CVS – Login OK



Windows CVS – Check Out



Windows CVS – Check Out Messages



Web Resources

- http://www.cygwin.com/
- http://www.wincvs.org/
- http://www.cvshome.org/project/www/docs/ddSSHGuideCygwin.html