

# Source Control

Comp-361 : Source Control  
Lecture 6

Alexandre Denault  
Computer Science  
McGill University  
Winter 2008

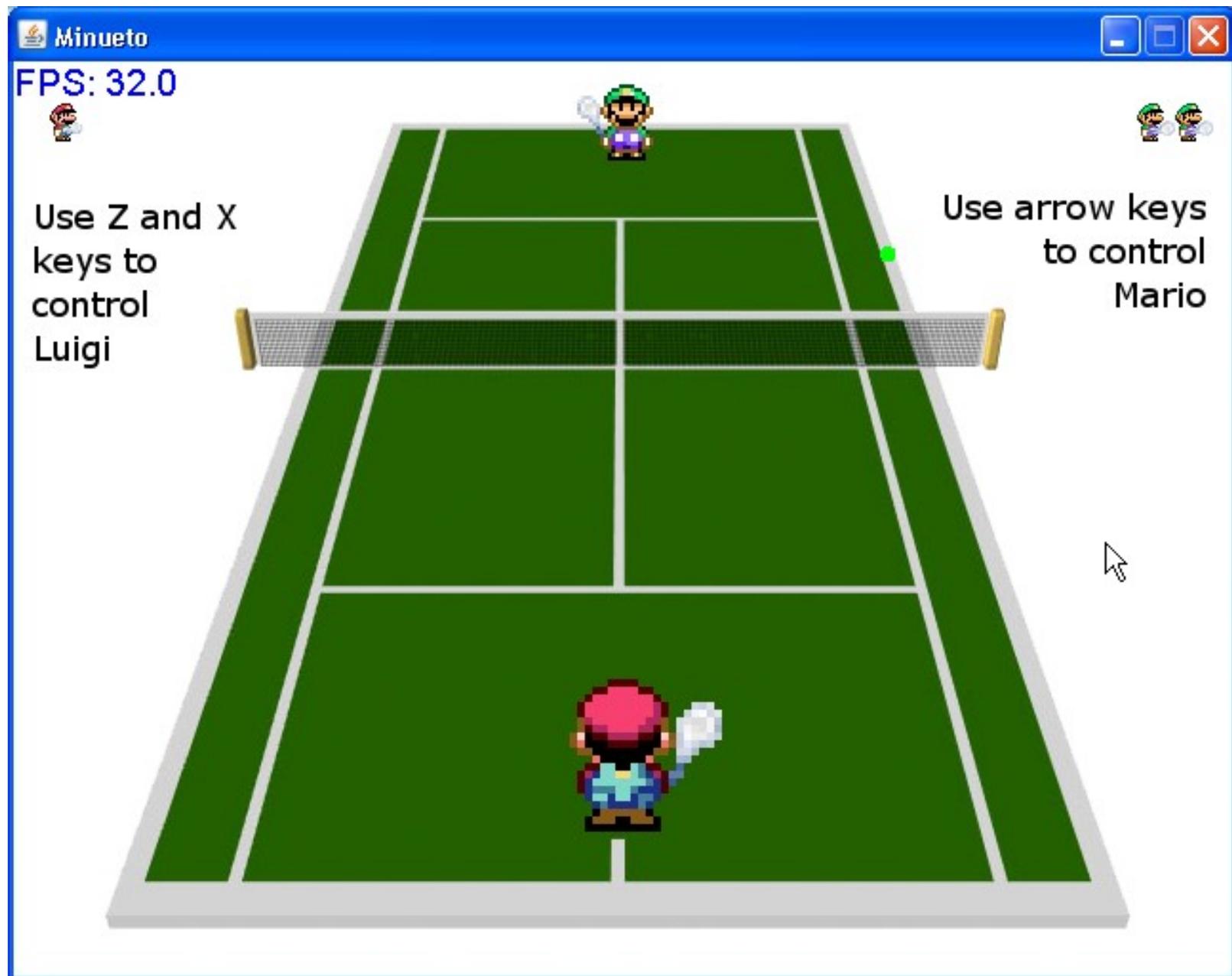
# Chicken Pox

- A student in the class was diagnose with Chicken Pox.
  - ◆ If you didn't get it as a child, or not vaccinated, time to get vaccinated.
  - ◆ If you suffer from a form of immune-deficiency, time to see a doctor.
- Your official McGill email box contains more information.

# Pong 36-Hour Challenge

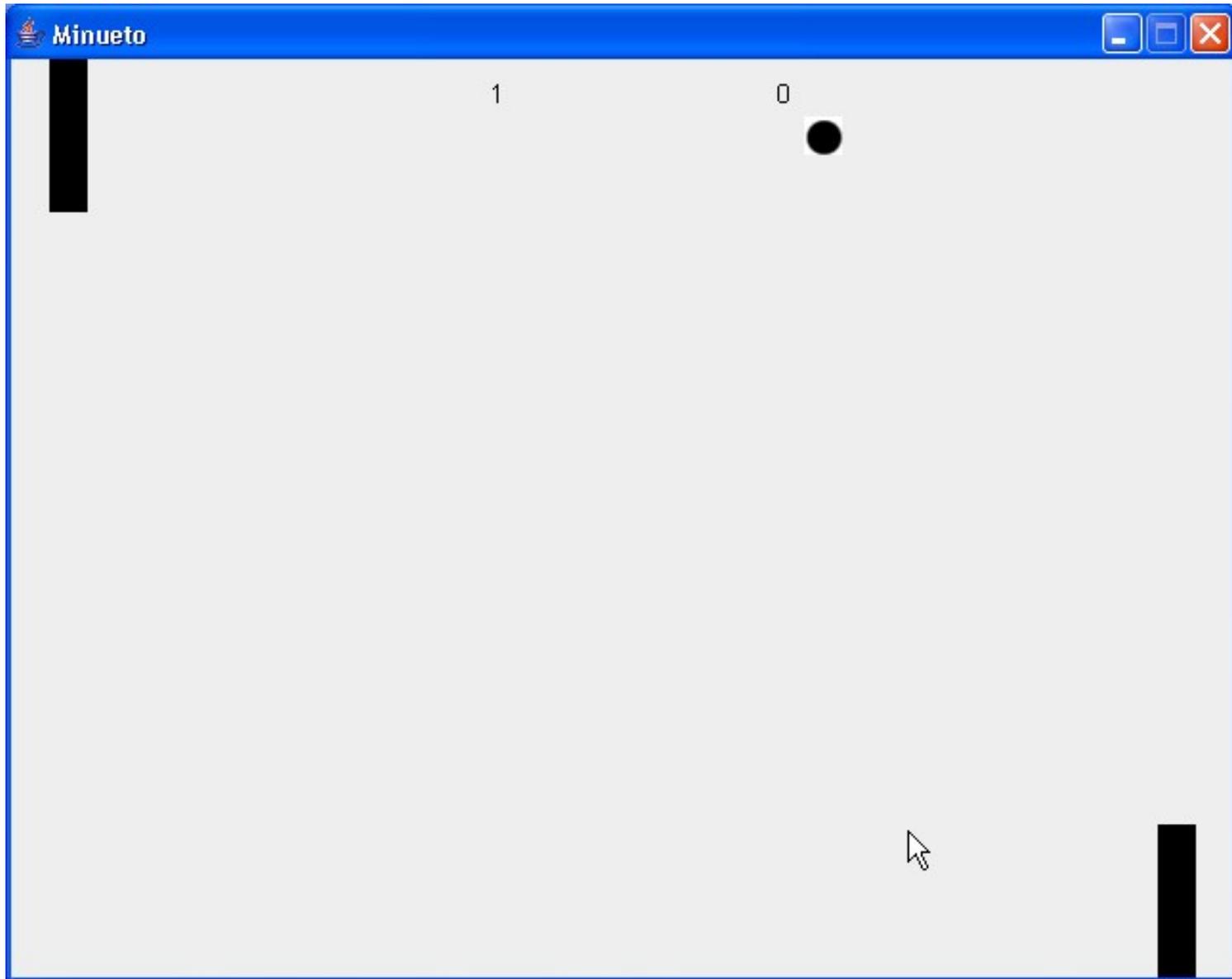
- Gabriel Lemonde-Labrecque
- Luke Bayly
- Marc-Olivier Dozois Lyrette
- Mitch Shum-lok
- Robert Rolnick
- Winston Lin

# Winner : Mario Pong



Gabriel Lemonde-Labrecque

# Participation : VoteE



Luke Bayly

# Question

Who knows about source control systems?

# Question

Who has use a source control system in a previous project?

# Question

Who has done a project where they find that the source control system was useful?

# Source Control

It's not enough to set it up, you need to use it!

– Emmanuel

# Question

Have you ever been working on a source file and wished you could retrieve a previous version of a file?

# Question

Have you ever worked on a team project and had difficult sharing files with your partner?

- Source Control is about the management of revision
  - ◆ changes are noted with a revision number

# Why?

- For sharing purposes (team work)
- For tracking / auditing purposes (accountability)
- For debugging purposes (history)

- The code is located in one central location
  - ◆ Code repository
  - ◆ Always contains the latest official version
- Each developer acquires his copy of the code
  - ◆ Local development
- To share changes, he must commit them to the repository.
  - ◆ Each change is assigned a revision number

# Can't avoid it

- If you work in industry, you will use a source control system.

# Team Overlap

- Source control allows large groups of developers to work on the same project
  - ◆ Minimizes the risks of overlapping changes.
- Each developer can work on his local copy
  - ◆ Doesn't affecting other developers.
  - ◆ Only commits once changes are stable.

# Question

What happens if several developers want to work on a separate experimental version of an application?

# Question

How can I record important revisions?

# Trunk / Branch / Tags

A source tree is separated into three categories: the trunk, branches and tags (tree analogy).

- The trunk is the main copy of your code.
- Branches are separate copies of your main code.
- Tags are snapshots of the trunk or branches.

- CVS is the Concurrent Versions System, was created in the mid 1980's.
  - It was recreated as a follow up to an earlier version system called Revision Control System (RCS).
    - ◆ RCS was great for individual files, bad for large projects.
- 

- Subversion (a.k.a. SVN) was developed as a modern day replacement to CVS.
- Subversion has many key features:
  - ♦ Commits are truly atomic (can't have problem with 2 people committing at the same time).
  - ♦ You can now move or rename files.
  - ♦ Strong integration with Apache.
  - ♦ Etc ...

# To set up a SVN at School

- You will need to collect the CS user names of each team members.
- Send a request to [help@cs.mcgill.ca](mailto:help@cs.mcgill.ca) for SVN directory for your Comp-361 project along with the collected user names.

# Creating a repository

- To create a repository, you simply need to use the `svnadmin` command.

```
svnadmin create /xtra/2008/cs361/team1
```

- ◆ This creates an svn directory in `/xtra/2008/cs361/team1`
- The next step would be to set up a trunk/branch/tag structure.
  - ◆ But you don't need it.

# URL of repository

- To use a repository, you need its location (URL)
- The URL depends on which access method you use.

`file:///directory`

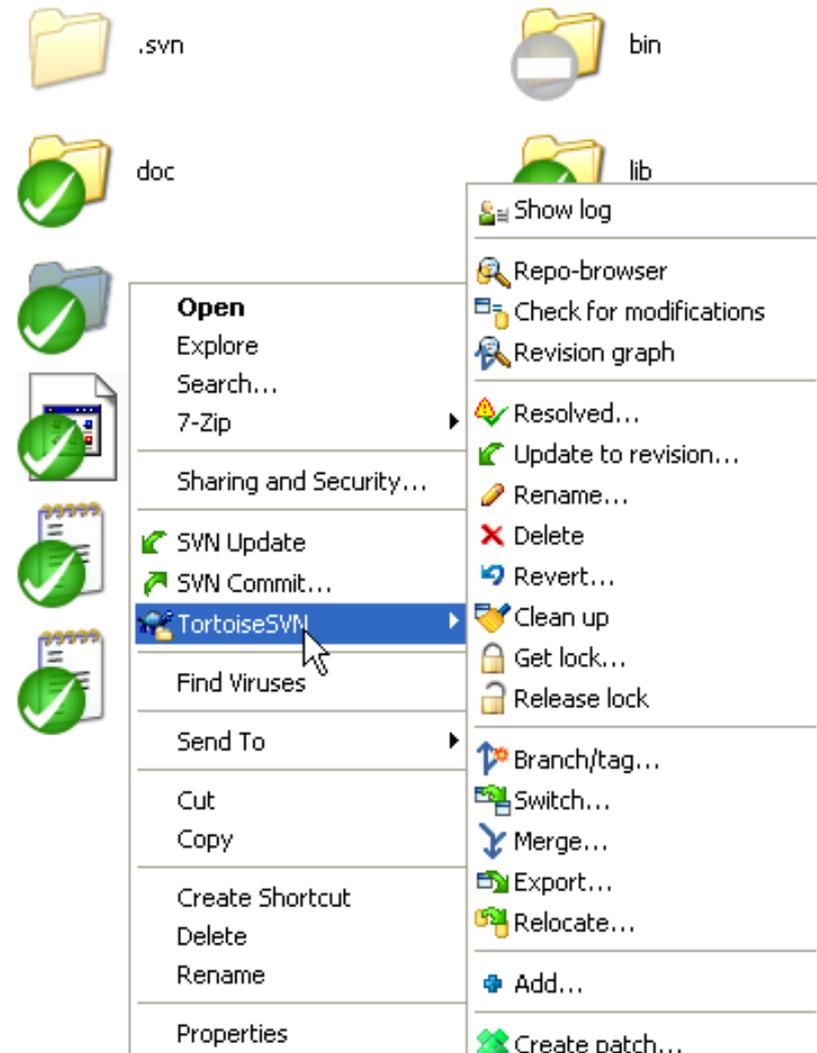
`svn+ssh://username@server/directory`

`http://server/xtra/directory`

`svn+ssh://bob@svn.cs.mcgill.ca/xtra/2008/cs361/team1`

# SVN Client

- All OS: command line svn
- Windows : SVN Tortoise
- Eclipse : Subclipse
- NetBeans : built-in



SVN Tortoise

# svn command

- The svn command is an all purposes tool. It contains all the necessary functionality to
  - ♦ **checkout** code from a repository
  - ♦ **adding** files to a repository
  - ♦ **update** a local repository
  - ♦ merge two revisions
  - ♦ compare two revisions
  - ♦ **commit** code to a repository
  - ♦ Etc.

## ■ You type in the *svn help* command to see

```
usage: svn <subcommand> [options] [args]
```

```
Subversion command-line client, version 1.2.3.
```

```
Type 'svn help <subcommand>' for help on a specific subcommand.
```

Most subcommands take file and/or directory arguments, recursing on the directories. If no arguments are supplied to such a command, it recurses on the current directory (inclusive) by default.

Available subcommands:

```
add
```

```
blame (praise, annotate, ann)
```

```
cat
```

```
checkout (co)
```

```
cleanup
```

```
commit (ci)
```

```
copy (cp)
```

```
...
```

# Checking Out

```
svn checkout URL [PATH]
```

- To modify code in a repository, you need to check out a local copy of the code.

```
svn checkout  
  svn+ssh://adenau@svn.cs.mcgill.ca/xtra/mammoth  
  /trunk mammoth-trunk
```

```
svn add FILES
```

- To add a file to a repository, you need to first place it in your checkout directory (in the correct location).
- Then call the `svn add` command.
- The file will be added next time you commit your changes.

# Committing

```
svn commit [PATH]
```

- Once you've tested your changes, you can commit them to the repository.
- When committing, you will be asked to supply a short message.
- This short message should explain what you are committing:
  - ◆ Changes you did
  - ◆ Reasons for the change
  - ◆ Bugs you fixed (including bug id if available)

# Updating

```
svn update [PATH]
```

- Other people are continuously contributing to the svn repository.
- To update your code with their latest changes, just use the `svn update` command.
- If somebody changed lines in a file that you also changed, a conflict occurs.
  - ♦ The file is going to be tagged as in a conflicted state.
  - ♦ Before you can commit your changes, you need to resolve the conflict.

# Resolving

```
svn resolved FILE
```

- Once both piece of code have been merge, the svn resolve command must be used to indicate the new state of the file.

# Conflict Avoidances

- To minimize the risk of conflicts, some companies have established “manual” locking scheme.

# Conflict Avoidances

- To minimize the risk of conflicts, some companies have established “manual” locking scheme.
- One of the most memorable is the stuffed toy locking system.
  - ♦ Only the person with the stuffed toy on his desk can commit his code to repository.
  - ♦ A programmer can “acquire” the toy by getting it from its designated storage.
  - ♦ Once he is finished committing his code, he must return the toy to its designated storage.
- Although this solution solves some problems of simultaneous commits, it
  - ♦ shares a lot of problems with file locking.
  - ♦ does not prevent conflicts from occurring, just reduces the chances.

```
svn status [PATH]
```

- For a given path, svn status will give the svn state of each file.
  - ♦ 'A' Added
  - ♦ 'C' Conflicted
  - ♦ 'D' Deleted
  - ♦ 'G' Merged
  - ♦ 'I' Ignored
  - ♦ 'M' Modified
  - ♦ 'R' Replaced
  - ♦ '?' item is not under version control
  - ♦ '!' item is missing
- More information about the output can be found by using `svn help status`.

# Read a tutorial



"svn tutorial"

Search

[Advanced Search](#)  
[Preferences](#)

Search:  the web  pages from Canada

Web

Results 1 - 10 of about 6,790 for "svn tutorial". (0.11 seconds)

## [SVN Tutorial fo Unix](#)

12 Jul 2007 ... **SVN Tutorial** fo Unix. This tutorial is meant to be read linearly so that it introduces the important notions gently. ...

[artis.inrialpes.fr/~Xavier.Decoret/resources/svn/index.html](http://artis.inrialpes.fr/~Xavier.Decoret/resources/svn/index.html) - 29k - [Cached](#) - [Similar pages](#) - 

## [SVN Tutorial](#)

**SVN Tutorial**. This page's menu: Research · Publications · Contact · Tutorials - --- **SVN Tutorial**. Subversion Tutorial. Subversion is an open source revision ...

[www.cs.ubc.ca/~vailen/svn\\_howto.htm](http://www.cs.ubc.ca/~vailen/svn_howto.htm) - 9k - [Cached](#) - [Similar pages](#) - 

## [SVN Tutorial](#)

[svnbook.red-bean.com/nightly/en/svn.intro.quickstart.html](http://svnbook.red-bean.com/nightly/en/svn.intro.quickstart.html) - [Similar pages](#) - 

## [svn command line tutorial for beginners 1 » Linux by Examples](#)

½ 5 votos. April 17th, 2007 mysurface Posted in Developer, svn, svnadmin, svnlook | Hits: 51664 |. svn can be known as subversion, it is a version control ...

[linux.byexamples.com/archives/255/svn-command-line-tutorial-for-beginners-1/](http://linux.byexamples.com/archives/255/svn-command-line-tutorial-for-beginners-1/) - 54k - [Cached](#) - [Similar pages](#) - 

## [SVN Tutorial](#)

**SVN Tutorial** Sean Russell CIO Germano Software 60252 Rimfire Rd Bend OR 97702

- SourceSafe is the previous version control package solution from Microsoft, distributed with Visual Studio.
  - ♦ purely file locking mechanism.
  - ♦ tight integration with Visual Studio
  - ♦ works well for small teams
  - ♦ does not scale well for large teams

# Visual Studio Team Foundation Server

- New solution from Microsoft for larger teams
  - ◆ source control
  - ◆ data collection
  - ◆ reporting
  - ◆ project tracking

- Perforce is the industry solution for revision control.
- It has an impressive client list
  - ◆ Activision, ATI, Cisco, EA, Ericsons, IBM, SCEA, etc
- Perforce supports several operating system and can integrate itself with several application.
  - ◆ Visual Studio / Eclipse / Xcode
  - ◆ Photoshop
  - ◆ 3DS Max, Maya
  - ◆ MS Office

# This Weekend

- Continue trying out technologies.
- Start thinking about data structures.
- Meetings start Monday.
  - ◆ McConnell 322