Refactoring: Subversion

CPSC 501: Advanced Programming Techniques
Fall 2020

Jonathan Hudson, Ph.D
Instructor
Department of Computer Science
University of Calgary

Tuesday, August 4, 2020

UNIVERSITY OF CALGARY

Subversion … rise and fall

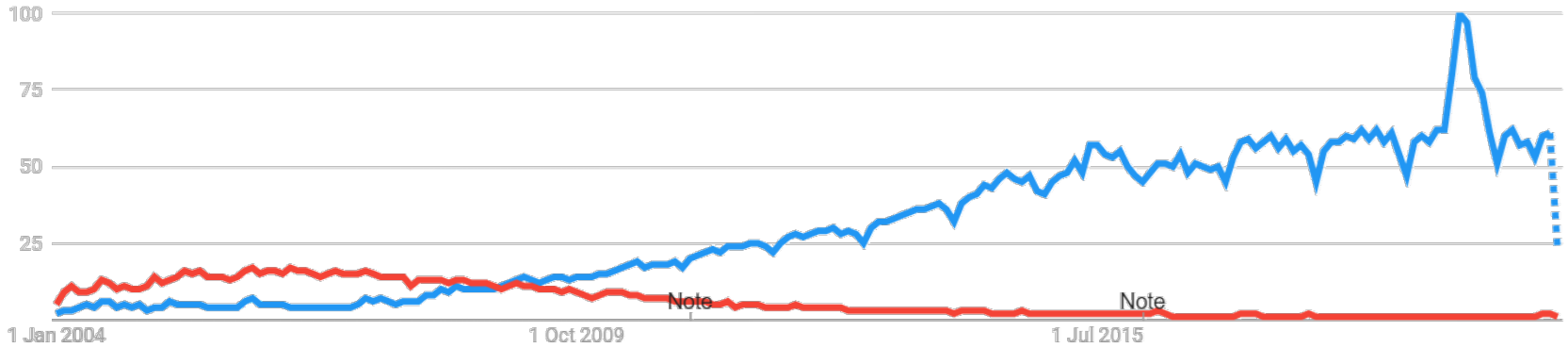
(and why we'll breeze through it)

Subversion … rise and fall

- Initial release 2000 (still update - > 1.14 May 2020)
- Open source (Apache Foundation)
- Popular with sourceforge! (Basically what github now is)
  - 2012 bought (along with Slashdot [RIP]) by Dice.com
  - Bundleware practices increased (some cases of malware)
  - Developers fled sit to others like Github
- Some more popular tools included TortoiseSVN

Subversion … rise and fall

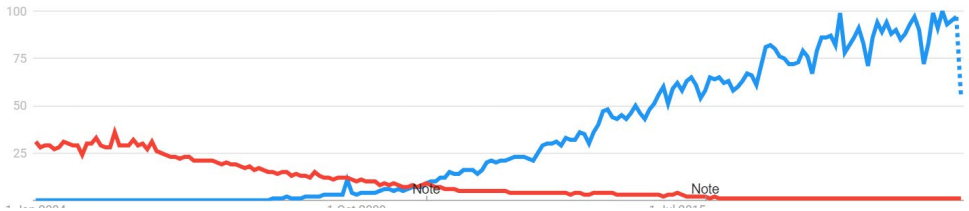
- Initial release 2000 (still update - > 1.14 May 2020)
- Open source (Apache Foundation)
- Popular with sourceforge! (Basically what github now is)
  - 2012 bought (along with Slashdot [RIP]) by Dice.com
  - Bundleware practices increased (some cases of malware)
  - Developers fled sit to others like Github
- Some more popular tools included TortoiseSVN
- Git (Blue) vs SVN (google trends)



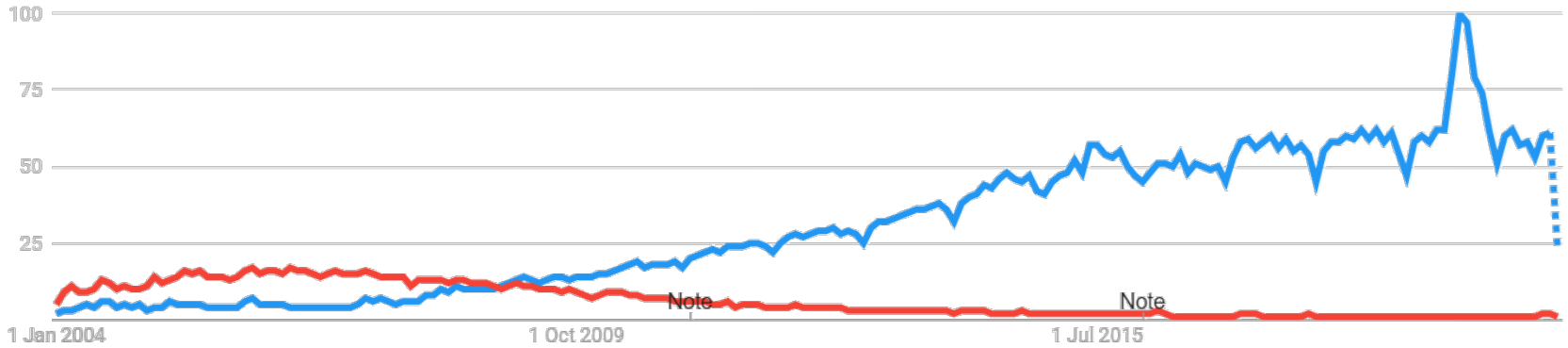
UNIVERSITY OF CALGARY

Subversion … rise and fall

- Initial release 2000 (still update - > 1.14 May 2020)
- Open source (Apache Foundation)
- Popular with sourceforge! (Basically what github now is)
  - 2012 bought (along with Slashdot [RIP]) by Dice.com
  - Bundleware practices increased (some cases of malware)
  - Developers fled sit to others like Github



- Some more popular tools included TortoiseSVN
- Git (Blue) vs SVN (google trends)



UNIVERSITY OF CALGARY

This is how you do it

Working with SVN

- Short for Subversion
  - Download and manuals available at: subversion.apache.org
- Use **svn --version** at the command line to check if installed
  - Also: **svnadmin --version**
- There are several GUI frontends to SVN
  - Many are not free
  - TortoiseSVN (will install SVN with it for you as well)
- Generally integrated in many IDEs

SVN: Creating a Repository

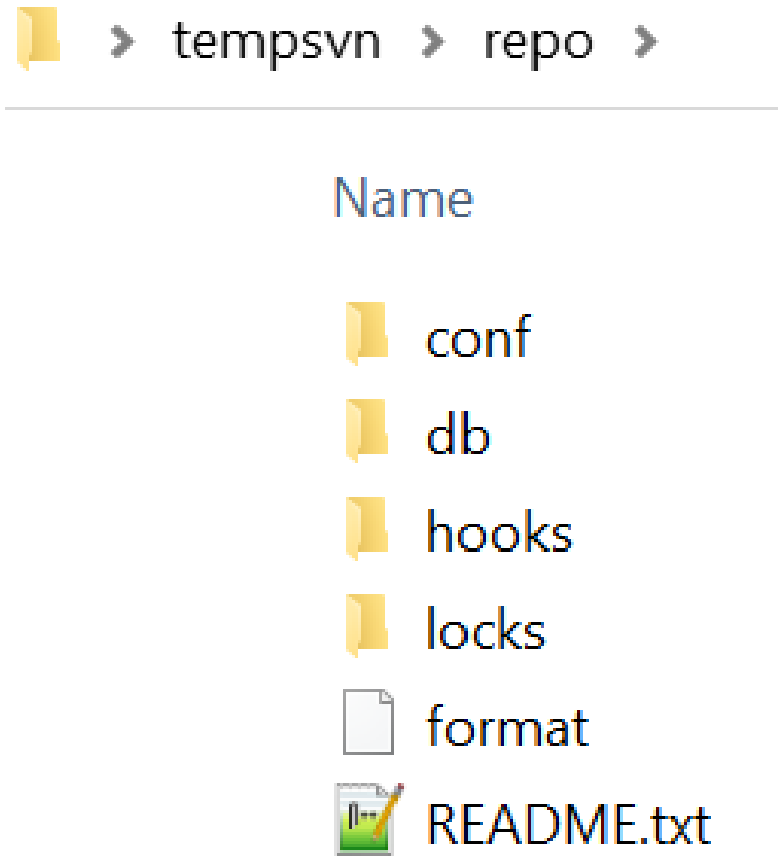
- Initial set up:
  - Create a repository in your account on the local file system.

    cd tempsvn

    mkdir repo

    svnadmin create repo
  - Do not directly change the files in this directory
    - These are all svn repository management files
    - Always use **svn** or **svnadmin** commands
  - Note the repository URL

    C:/tempsvn/repo



SVN: Import some initial files

- Create your project
  - Decide on a project name (e.g. "panther")
  - Create some initial source code files in a temporary directory

```
mkdir temp
cd temp
echo public class Person{} > Person.java
echo public class Student extends Person{} > Student.java
```

  - Import the files into the repository

```
svn import . "file:///C:/tempsvn/repo/panther/trunk" –m "initial import"
```

  - We can remove the **temp** directory and files now
    - They are recorded in the repository

SVN: Import some initial files

svn import . "file:///C:/tempsvn/repo/panther/trunk" -m "initial import"

- **svn** -> program that does subversion commands
- **import** -> the import command
- **.** -> everything in our temp directory (i.e. Student.java and Person.java)
- **"file:///C:/tempsvn/repo/panther/trunk"** -> the repository directory
  - we'll start a new **trunk** for a **panther** project with these two files
  - This often a url "https://svn.example.com/repos/repo/**panther**/trunk"
- **-m** -> a flag to send in a message attached to the import
- **"initial import"** -> an import message

SVN: Workflow

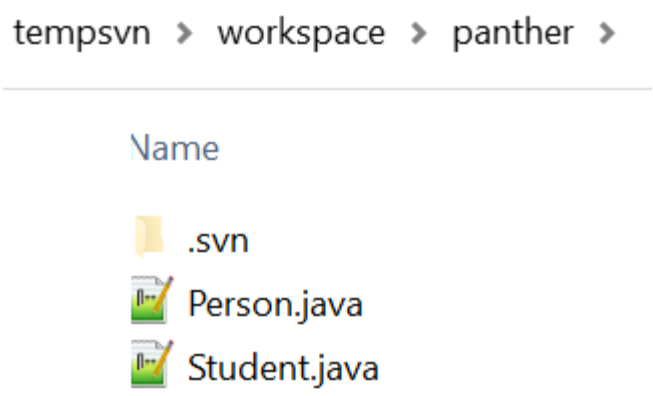
- Typical daily workflow:
  - Create a workspace directory

    mkdir workspace

    cd workspace
  - Check out the project to a folder called **panther** in **workspace**

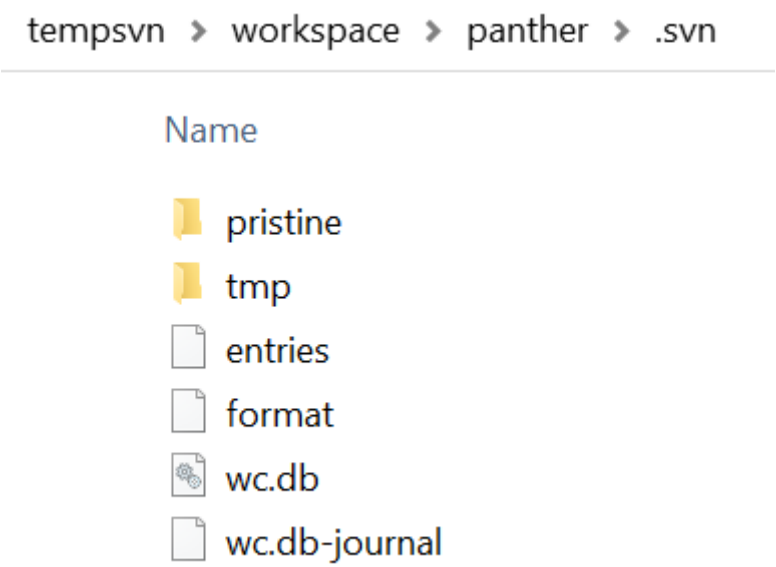
    svn co file:///tempsvn/repo/panther/trunk panther
  - Change into the project subdirectory

    cd panther



SVN: .svn

- Hidden folder in file system
- You may have to show hidden files to see it
- Tracks info about repo locally
- Do not delete/alter this directory if you plan to do anything after the checkout action that you want to affect the repo correctly
- If you plan to archive your project in a zip file somewhere as a final item (remove the directory)



SVN: Status, Diff

- Make changes to the files
  - Use **status** to give the current state of the files

    svn status *.java
    - Should indicate they are “locally modified” (M) if any changes down
  - Use **diff** to show the differences between the local copy and repository version

    svn diff Person.java

SVN: Commit, Log, Update

- Commit the changes to the repository

  svn commit -m “Did rename method refactoring”

- Use **log** to see the history of a file

  svn log Person.java

- Use **update** to refresh the files and directories in the workspace
  - If there is a newer revision of your file in the repo you will not be allowed to commit until you **update** and self merge in your change to the file
  - Necessary if a multi-person project (or if you deleted local copies)

    svn update

SVN: Add

- Use **add** to add new directories to the repository

```
mkdir package
svn add package
```

Use **add** to add new files to the repository

```
cd package
# create and edit file Math.java
svn add Math.java
svn commit -m "new math stuff"
```

SVN: Delete and Move

- Use **delete** to remove directories and/or files from the repository

  svn delete package

  svn commit -m “deleted package”

- Use **mv** to move and/or rename files and directories

  svn mv Person.java Main.java

  svn commit -m “renamed file”

- **move**, **mv**, and **ren** are aliases for **rename**

SVN: Revert changes before committed

- Use **revert** to reverse unwanted changes done to **working copy** files and/or directories

  # mistakenly edit Student.java

  svn revert Student.java

- Use **-R** to apply this to an entire directory or the entire project

  svn revert -R mydirectory

  # Or in current directory

  svn revert -R .

SVN: Revert committed revisions

- To revert a **committed** revision:
  - Do a reverse merge from the latest revision to an earlier revision
    svn merge -r 5:4 .
  - Commit the change
    svn commit -m "reverted to r4"
  - **Note: this creates a revision 6, identical to 4**
    - **i.e. SVN never throws anything away**

SVN: Release tagging

- To create a release tag from a working copy:
  - Make sure the project is up to date

    svn update

  - If not done already, create a “tags” subdirectory for the project in the repository

    svn mkdir file:///C:/tempsvn/repo/panther/tags -m “Created tags subdirectory”

  - Use copy to create a tagged version in the newly created subdirectory in the repository

    svn copy . file:///C:/tempsvn/repo/panther/tags/Rev1 -m “Created Rev1 tag”

  - Later, you can use this tag to checkout this set of files

    svn co file:/// C:/tempsvn/repo/panther/tags/Rev1 panther

SVN: Notes

- Many IDEs have support for subversion
- Will be able to connect a project to a repository file/url and generally see in GUI if a file has been modified or not during coding
- You can then commit a bunch of changed/added files through GUI with a message
- Generally other behaviours such as rename, delete, tagging, reversion all supported

- TortoiseSVN is a tool that integrates in file system explorer (adds right click svn menu)

- **svn is centralized version control, if you aren't in contact with repo you can't version control changes.**
  - Leads to big commits in communication challenged situations

Onward to …
git.

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~hudsonj/

UNIVERSITY OF CALGARY