A short tutorial on Git

Serveshe Muralidharan

4 March 2014
This Tutorial

• What is Source Control
• Distributed source control with Git
• Git in Assignment 5
• Using Gitlab
• Using git in Eclipse
• More about Assignment 5
This Tutorial

• What is Source Control
• Distributed source control with Git
  • Git in Assignment 5
  • Using Gitlab
  • Using git in Eclipse
• More about Assignment 5
What is Source Control

- Tracks changes to files
- Maintains history of changes
- Maintains snapshots
- Enables collaboration

Main operations
- **add:** mark files on the hard drive to be tracked in the repository
- **commit:** commit to the repository the changes in files on the hard drive
- ...
Distributed Source Control

• Allows multiple repositories
  – each one is a copy of the main repository (usually)

• All repositories can be synchronized
  – clone: creates a local copy of the main repo
  – push the changes from the local repository to the main repo
  – pull the changes from the main repo to the local one
Distributed Source Control with Git

Git a popular program to use distributed source control

Operations:
- add
- commit
- push
- pull
- ...
This Tutorial

• What is Source Control
• Distributed source control with Git
• Git in Assignment 5
• Using Gitlab
• Using git in Eclipse
• More about Assignment 5
Git in Assignment 5

• Teams of 2
  – by Friday, 7 March: **form teams**
  – if you haven’t formed team by then you will be assigned a partner

• Each team will use one git repository for the assignment (**details to follow**)

• Marking:
  – Correctness (common for both team members)
  – Performance (common for both team members)
  – contribution to the git repository (for each student)
The output has been generated by [gitinspector](https://code.google.com/p/gitinspector/), the statistical analysis tool for git repositories.

The following historical commit information, by author, was found in the repository.

<table>
<thead>
<tr>
<th>Author</th>
<th>Commits</th>
<th>Insertions</th>
<th>Deletions</th>
<th>% of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>288</td>
<td>7721</td>
<td>4617</td>
<td>39.19</td>
</tr>
<tr>
<td>James Johnson</td>
<td>135</td>
<td>8910</td>
<td>2422</td>
<td>35.99</td>
</tr>
<tr>
<td>Robert Brown</td>
<td>71</td>
<td>2564</td>
<td>1352</td>
<td>12.44</td>
</tr>
<tr>
<td>Michael Davids</td>
<td>134</td>
<td>2943</td>
<td>954</td>
<td>12.38</td>
</tr>
</tbody>
</table>

Below are the number of rows from each author that have survived and are still intact in the current revision.

<table>
<thead>
<tr>
<th>Author</th>
<th>Rows</th>
<th>% in comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>3533</td>
<td>22.02</td>
</tr>
<tr>
<td>James Johnson</td>
<td>6113</td>
<td>52.15</td>
</tr>
<tr>
<td>Robert Brown</td>
<td>1123</td>
<td>21.19</td>
</tr>
<tr>
<td>Michael Davids</td>
<td>1464</td>
<td>20.15</td>
</tr>
</tbody>
</table>

The following history timeline has been gathered from the repository.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td></td>
</tr>
<tr>
<td>James Johnson</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td></td>
</tr>
<tr>
<td>Robert Brown</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Michael Davids</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Modified Rows:</td>
<td>1522</td>
<td>3832</td>
<td>7553</td>
<td>6143</td>
<td>5833</td>
<td>5123</td>
<td>1477</td>
</tr>
</tbody>
</table>

The extensions below were found in the repository history (extensions used during statistical analysis are marked):

xml java pdf txt css

https://code.google.com/p/gitinspector/
Git in Assignment 5

- Make sure you commit code only from your user name
- Code changes, amount of code contributed, type of changes, etc., would be used to calculate individual member's marks
- Not necessary for you to use advanced git features
- You will not loose or gain points for using or not using features of git
This Tutorial

• What is Source Control
• Distributed source control with Git
• Git in Assignment 5
• Using Gitlab
• Using git in Eclipse
• More about Assignment 5
Using Gitlab with eclipse

1. First time (one user does this)
   - Create a remote repo in gitlab
   - Create a local repo
   - Add files to local repo
   - Commit to local
   - Push (local \(\rightarrow\) global)

2. Every other time (same user, on the same computer)
   - Pull global to local
   - Change files on disk
   - Add more files to local repo
   - Commit (local)
   - Push (local \(\rightarrow\) global)

3. Every other time (every user, on different computer)
   - Clone the remote repository
   - Change / add files
   - Commit
   - Push
First time (one user does this)

- Create a remote repo in git lab
- Create a local repo
- Add files to local repo
- Commit to local
- Push (local → global)
1. Goto [www.gitlab.scss.tcd.ie](http://www.gitlab.scss.tcd.ie)
2. Log in using scss username / password
3. In the dashboard click on the '+' symbol
4. Specify a project name
5. Select visibility level to be 'Private'
6. Click on 'Create Project'
7. Click on 'http' button
8. Make a note of the url highlighted in yellow
9. Select 'settings' tab on the top, followed 'members' on the left
10. Select New project member
11. Search for your team mates name in the 'people field'
12. All users with scss account should be visible
13. Select Project Access level to what you desire, ex: Team Leader – Master and the rest as Developer.
14. After adding all of your team mates, add the following members of CS2012 staff as Reporters.

(1) Vasileios Koutavas
(2) Servesh Muralidharan
(3) Aravind Vasudevan
(4) Shixiong Xu
(5) Yu Xu

Important: To be able to access and mark your project
First time (one user does this)

- Create a remote repo in git lab
- Create a local repo
- Add files to local repo
- Commit to local
- Push (local → global)
1. You can choose your workspace directory in 'U' Drive, if you don’t want to keep setting up eclipse on every system or alternatively you can use your personal machine.
2. Choose import wizard from file menu.
3. Select 'Git' followed by 'Projects from Git'
4. Choose the 'URI' option
5. Provide the link you saved from the gitlab
6. Provide your scss username and password.
7. Since this is an empty repository, Eclipse will warn you about it, select next.
8. Select a directory where you want to create the local repository
9. The master password warning is not required to get rid of it click on 'yes' and select cancel on the next step.
10. Click cancel to ignore master password creation
11. Since its an empty repository, you need to use the New project wizard
12. Select 'Java Project' as the type
13. Provide project name, note it **should be the same name** that you used in the gitlab server.
14. The rest of options leave it at default and click 'Finish'.
15. Eclipse will take a few sec and create the project
16. Right click on **project name** in project explorer window then select **Team** followed by **Commit**
17. Type in your commit message
18. Select 'All the files' in your project
19. Click on commit and push
20. Eclipse displays the message showing the successful commit and the creation of master branch
First time (one user does this)

- Create a remote repo in git lab
- Create a local repo
- Add files to local repo
- Commit to local
- Push (local $\rightarrow$ global)
1. Add a file to the java project as usual
First time (one user does this)

- Create a remote repo in git lab
- Create a local repo
- Add files to local repo
- Commit to local
- Push (local → global)
2. Right click on project name in project explorer window then select Team followed by Commit
3. Commit and push the changes, but make sure to add the newly created files.
4. Notice a new node being created for the commit of the file
Every other time (same user, on the same computer)

- Pull global to local
- Change files on disk
- Add more files to local repo
- Commit (local)
- Push (local → global)
1. To fetch or push from the repository, select the project name followed by Team and then choose either 'Fetch from upstream' or 'Pull' for pull
Every other time (same user, on the same computer)

- Pull global to local
- Change files on disk
- Add more files to local repo
- Commit (local)
- Push (local $\rightarrow$ global)
1. Right click on **project name** in project explorer window then select **Team** followed by **Commit**
2. Commit and push the changes, but make sure to add the newly created files
Every other time (every user, on different computer)

- Clone the remote repository
- Change / add files
- Commit
- Push
1. You can choose your workspace directory in 'U' Drive, if you don’t want to keep setting up eclipse on every system or alternatively you can use your personal machine.
2. Choose import wizard from file menu.
3. Select 'Git' followed by 'Projects from Git'
4. Choose the 'URI' option
5. Provide the link you saved from the gitlab
6. Provide your scss username and password.
7. When you reach the Branch selection window you should find the already created master branch.

8. If you get permission error or unable to access the repository ensure that you have the relevant access privileges when the team lead creates them.

9. Alternatively you can log into your gitlab account and should notice the project you have been added under
10. Specify the location where the local repository should be created
11. Make sure you choose 'Import existing projects'.
12. Finally click on finish to complete the import of the existing project.
Every other time (every user, on different computer)

- Clone the remote repository
- Change / add files
- Commit
- Push
1. Add a file to the java project as usual
Every other time (every user, on different computer)

- Clone the remote repository
- Change / add files
- Commit
- Push
1. Right click on **project name** in project explorer window then select **Team** followed by **Commit**
2. Commit and push the changes, but make sure to add the newly created files.
3. Notice a new node being created for the commit of the file
“Conflict resolution”

• Both partners edit the same lines of code → conflict
  – Git will refuse to push code onto remote repository
  – Prompts for resolution

• Resolution: merge conflicts manually
  – In Eclipse: http://youtu.be/ZK20jVt7XEc
  – Commit/push again
Avoiding conflicts

- Simple solution
  - Team work
  - Decide on a function for each team member
  - Only one team member modifies a particular function at a time
  - Commit and push code every time you have reasonable set of changes
  - Always pull the code from the server before you modify existing code
- but it’s not tragic!
This Tutorial

- What is Source Control
- Distributed source control with Git
- Git in Assignment 5
- Using Gitlab
- Using git in Eclipse
- More about Assignment 5
Specifics for Assignment 5

• Deadline for setting up your team members, **Friday, March 7th**.

• Report to us if you don’t have a team.

• Deadline for setting up a bare repository and each team member committing and pushing a change is end of day **Tuesday, March 11th**.

• You would need to submit the team names, members and the git URL as a text document to the submission server.