Introduction to Git For Groups
Getting the changes made by others
Viewing old versions of your work
Working on multiple lines of development
A single person working

git clone https://gitlab.doc.ic.ac.uk/lab1314_summer/arm11_<group_no>.git

Your home directory
Adding changes to the index

- `git add`
- `git rm`
- `git mv`
- `git status`
- `git diff`
- `git diff --cached`
>git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   Added.txt
    modified:   Hello.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   Hello.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    NotAdded.txt
git status, diff and diff --cached

>git diff --cached Hello.txt
diff --git a/Hello.txt b/Hello.txt
index cd08755..a92b91b 100644
--- a/Hello.txt
+++ b/Hello.txt
@@ -1 +1,3 @@
    Hello world!
+This is my brand new story that I've added...

>git diff Hello.txt
diff --git a/Hello.txt b/Hello.txt
index a92b91b..0a7ef26 100644
--- a/Hello.txt
+++ b/Hello.txt
@@ -1,3 +1,5 @@
    Hello world!
+This is my brand new story that I've added...
+
+And this is something I've written in Hello.txt but not added.
Committing Changes

```
master
```

```
git add
```

```
git commit
```

```
toto
```

```
git add
```
Committing and pushing

>git commit -m "Added Hello."
[master 68c9ae4] Added Hello.
  2 files changed, 3 insertions(+)
  create mode 100644 Added.txt

>git push origin master
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 590 bytes | 0 bytes/s, done.
Total 7 (delta 0), reused 0 (delta 0)
To ../tmp_remote.git/
  * master -> master
Committing and pushing
Two people working at separate times
Two people working at separate times

```
git clone <url>
<edit>
git add <file>
git commit
git push origin master
```
Two people working at separate times

```
master
```

```
git fetch
  origin/master
  master
```

```
git merge origin/master
  origin/master
  master
```

```
master
```
Two people working at the same time (different files)
Two people working at the same time (different files)

```
git clone <url>
<edit file one>
git add
git commit

master
```

```
master
origin/master

git push origin master
```

```
git clone <url>
<edit file two>
git add
git commit
```

```
master
origin/master
```
Two people working at the same time (different files)

```
git fetch
origin/master
master

git merge origin/master
```
Two people working at the same time (different files)

- `git push origin master`
- `git fetch`
- `git merge origin/master`
Two people working at the same time (same files)
Two people working at the same time (same files)

>git merge origin/master
Auto-merging Hello.txt
CONFLICT (content): Merge conflict in Hello.txt
Automatic merge failed; fix conflicts and then commit the result.
Two people working at the same time (same files)

Hello world!
This is the start of a great program!!!

I like zebras.

Hello world!
I hope you're listening.
This is the start of a great program!!!

I like elephants.

Hello world!
I hope you're listening.
This is the start of a great program!!!

I like giraffes.

<<<<<<<<<<< HEAD
I like elephants.
=======
I like giraffes.
>>>>>>origin/master
Two people working at the same time (same files)

Hello world!
I hope you're listening.
This is the start of a great program!!

<<<<<<< HEAD
I like elephants.
=======
I like giraffes.
>>>>>>> origin/master

<edit Hello.txt to resolve>
git add Hello.txt
git commit

Hello world!
I hope you're listening.
This is the start of a great program!!

I like elephants and giraffes.
Two people working at the same time (same files)

Merge remote-tracking branch 'origin/master'

Conflicts:
  Hello.txt
#
# It looks like you may be committing a merge.
# If this is not correct, please remove the file
# .git/MERGE_HEAD
# and try again.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Your branch and 'origin/master' have diverged,
# and have 1 and 1 different commit each, respectively.
# (use "git pull" to merge the remote branch into yours)
#
# All conflicts fixed but you are still merging.
#
# Changes to be committed:
# modified:   Hello.txt
Viewing / browsing old versions of your work

gitk --all &
Viewing / browsing old versions of your work

```bash
$ git log spec/spec.tex

commit ffba57269dcc1a12cfc929e3a4f6322931cf0f1c
Author: Dr. Tristan Allwood <tora@doc.ic.ac.uk>
Date:   Tue May 20 12:31:59 2014 +0100

    More updates to the spec.

commit 93734d0101efaa6b0dfb347587284eea844b36bb
Author: Dr. Tristan Allwood <tora@doc.ic.ac.uk>
Date:   Mon May 12 15:38:50 2014 +0100

    Spec pass up to working and submission.

commit e4d606286b3791b01c3636040f541502f9af196a
Merge: 1976787 1f0d7c6
Author: Dr. Tristan Allwood <tora@doc.ic.ac.uk>
Date:   Mon May 12 11:08:18 2014 +0100

    Merge remote-tracking branch 'origin/master' into wip-2013

Conflicts:
...
Viewing / browsing old versions of your work

```
> git blame spec/spec.tex

<table>
<thead>
<tr>
<th>Commit</th>
<th>Author</th>
<th>Date</th>
<th>Time</th>
<th>Zone</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>61313227</td>
<td>Dr. Tristan Allwood</td>
<td>2013-05-19</td>
<td>12:52:55</td>
<td>+0100</td>
<td>1) \documentclass[a4paper,twoside]{article}</td>
</tr>
<tr>
<td>cd6b4b2d</td>
<td>Will Jones</td>
<td>2012-05-14</td>
<td>12:09:02</td>
<td>+0100</td>
<td>2)</td>
</tr>
<tr>
<td>61313227</td>
<td>Dr. Tristan Allwood</td>
<td>2013-05-19</td>
<td>12:52:55</td>
<td>+0100</td>
<td>3) \usepackage{fullpage}</td>
</tr>
<tr>
<td>c05c8bed</td>
<td>Will Jones</td>
<td>2012-04-18</td>
<td>17:36:15</td>
<td>+0100</td>
<td>4) \usepackage{bytefield}</td>
</tr>
<tr>
<td>dfe12a56</td>
<td>Will Jones</td>
<td>2012-05-14</td>
<td>14:40:51</td>
<td>+0100</td>
<td>5) \usepackage{graphicx}</td>
</tr>
<tr>
<td>0155868c</td>
<td>Dr. Tristan Allwood</td>
<td>2012-05-17</td>
<td>19:04:45</td>
<td>+0100</td>
<td>6) \usepackage{hyperref}</td>
</tr>
<tr>
<td>5a27d821</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:18:02</td>
<td>+0100</td>
<td>7) \usepackage{alltt}</td>
</tr>
<tr>
<td>a3b7a457</td>
<td>Dr. Tristan Allwood</td>
<td>2013-05-17</td>
<td>17:20:28</td>
<td>+0100</td>
<td>8) \usepackage{amsmath}</td>
</tr>
<tr>
<td>a3b7a457</td>
<td>Dr. Tristan Allwood</td>
<td>2013-05-17</td>
<td>17:20:28</td>
<td>+0100</td>
<td>9) \usepackage{tabulary}</td>
</tr>
<tr>
<td>d87cdb9f</td>
<td>Maria</td>
<td>2013-05-13</td>
<td>17:42:20</td>
<td>+0100</td>
<td>10) \usepackage{float}</td>
</tr>
<tr>
<td>73bb5cc4</td>
<td>Dr. Tristan Allwood</td>
<td>2013-05-19</td>
<td>19:34:44</td>
<td>+0100</td>
<td>11) \usepackage{comment}</td>
</tr>
<tr>
<td>d87cdb9f</td>
<td>Maria</td>
<td>2013-05-13</td>
<td>17:42:20</td>
<td>+0100</td>
<td>12) \restylefloat{table}</td>
</tr>
<tr>
<td>cd6b4b2d</td>
<td>Will Jones</td>
<td>2012-05-14</td>
<td>12:09:02</td>
<td>+0100</td>
<td>13)</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>14) \hypersetup{</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>15) colorlinks=true,</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>16) linkcolor=blue,</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>17) citecolor=blue,</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>18) filecolor=blue,</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>19) urlcolor=blue</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>20) }</td>
</tr>
<tr>
<td>07215a58</td>
<td>Will Jones</td>
<td>2012-05-18</td>
<td>12:31:43</td>
<td>+0100</td>
<td>21)</td>
</tr>
<tr>
<td>cd6b4b2d</td>
<td>Will Jones</td>
<td>2012-05-14</td>
<td>12:09:02</td>
<td>+0100</td>
<td>22) % Macros for printing code</td>
</tr>
<tr>
<td>cd6b4b2d</td>
<td>Will Jones</td>
<td>2012-05-14</td>
<td>12:09:02</td>
<td>+0100</td>
<td>23)</td>
</tr>
</tbody>
</table>
```
Multiple Branches of Development
Multiple Branches of Development
Multiple branches of development
Multiple branches of development

```bash
git checkout -b my_awesome_fire_truck
<work and commit on the fire truck branch>
```
Multiple branches of development

```bash
git checkout master
```

```
master
```
Multiple branches of development

```
git fetch

origin/my_awesome_fire_truck

git checkout my_awesome_fire_truck

my_awesome_fire_truck

master

origin/master

```

```
git push origin my_awesome_fire_truck

```
Multiple branches of development

```
git checkout master

my_awesome_fire_truck

master

my_awesome_fire_truck

master
```

```
git merge my_awesome_firetruck
<merge conflict>
<put out fires>
git add

git commit
```
3.11.1 Program not running

This state indicates that Deeva has just been started from the command line and program is not yet running. In this state, user can inspect various files and set breakpoints accordingly.

3.11.2 Program is running

When user runs the program, Deeva is in the running state until it starts waiting for input or reaches a breakpoint or terminates. This state indicates that Deeva is just executed from the command line and program has not been run yet. In this state, user can inspect various files and set breakpoints accordingly.

3.11.3 Program is running

When user runs the program, Deeva is in the running state until it reaches a breakpoint, it might also be waiting for input, or the program is terminated. Running state could be hardly seen sometimes as program reaches paused state very quickly.

3.11.4 Program paused

Once Deeva reaches a breakpoint, it is in a paused state. During this state, user may choose various actions e.g. continue the program, execute the next line or inspect the current state of the stack and heap.
Lots of things we've seen

git clone       git fetch
git status     git merge
git add        git log
git rm         gitk --all
git mv         gitg
git diff       git blame

git commit     git checkout -b
git push
Lots of other things to investigate

git checkout  git help <command>
git cherry-pick

git rebase

git stash

git reset

git branch

git remote