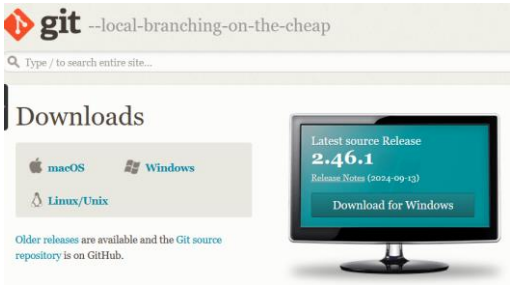
Höhere technische Bundeslehranstalt Wien 3
Rennweg 89b, A-1030 Wien
Höhere Abteilung für Informationstechnologie
Fachschule für Informationstechnik

Labor-Protokoll
SEW

Name	**Luka Pacar**
4-stellige Login-Nummer	**1188**
Klasse	**4CN**
Datum der Übung	17.09.2024
Datum der Abgabe	18.09.2024
Übungsnummer	00
Auftraggeber	ZAI
Thema der Übung	**git manual**

Inhalt

1 Git Installation

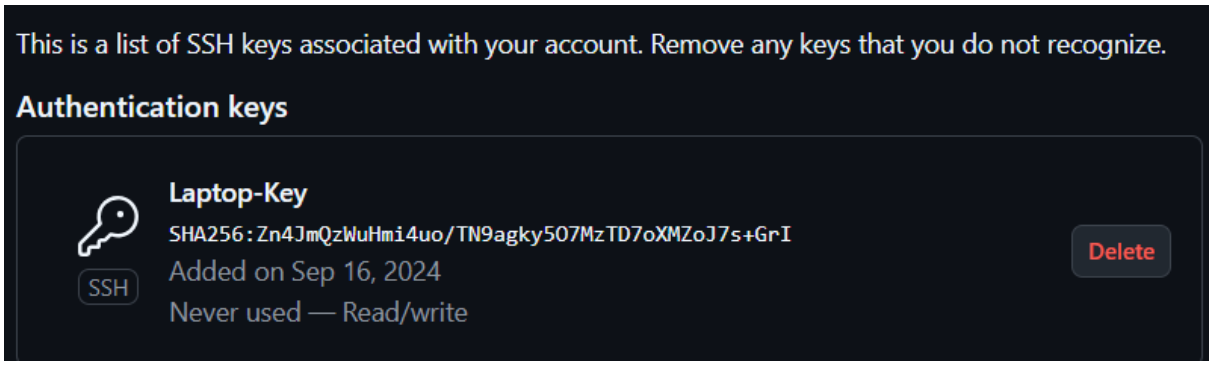


Erste Konfiguration:

```
PS C:\Users\Luka Pacar> git config --global user.name "Luka Pacar"
PS C:\Users\Luka Pacar> git config --global user.email "1188@htl.rennweg.at"

PS C:\Users\Luka Pacar> git config --global push.default simple
```

SSH Key erstellen und in Github einbinden:

```
PS C:\Users\Luka Pacar> ssh-keygen.exe
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\Luka Pacar/.ssh/id_rsa):
Created directory 'C:\\Users\\Luka Pacar/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\Luka Pacar/.ssh/id_rsa
Your public key has been saved in C:\Users\Luka Pacar/.ssh/id_rsa.pub
The key fingerprint is:
```



2 Git Übungen – Git Katas

2.1 Basic-Commit

1. Use `git status` to see which branch you are on.

```
mits\exercise> git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
mits\exercise>
```

2. What does `git log` look like?

```
mits\exercise> git log
fatal: your current branch 'master' does not have any commits yet
```

3. Create a file

4. What does the output from `git status` look like now?

```
mits\exercise> git status
On branch master

No commits yet

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

5. `add` the file to the staging area

```
mits\exercise> git add .\test.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

6. How does `git status` look now?

```
mits\exercise> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   test.txt

PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

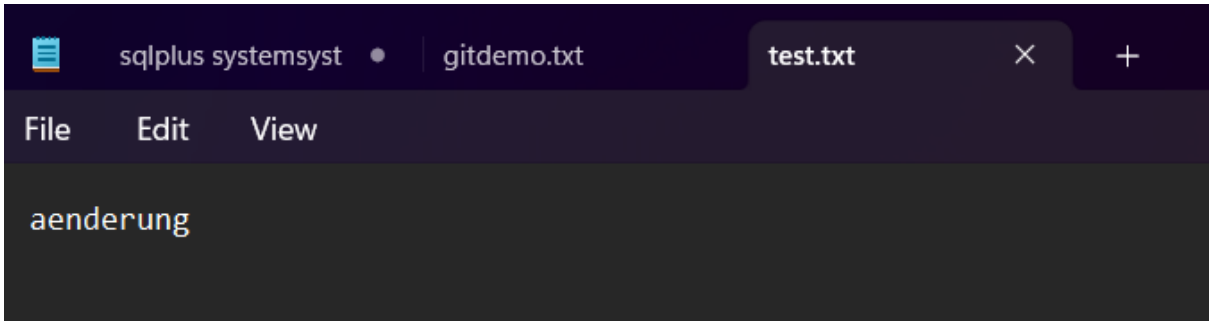
7. `commit` the file to the repository

```
mits\exercise> git commit -m "UE00: Basic-Commit1"
[master (root-commit) a28f287] UE00: Basic-Commit1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

8. How does `git status` look now?

```
mits\exercise> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

9. Change the content of the file you created earlier



10. What does `git status` look like now?

```
mits\exercise> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

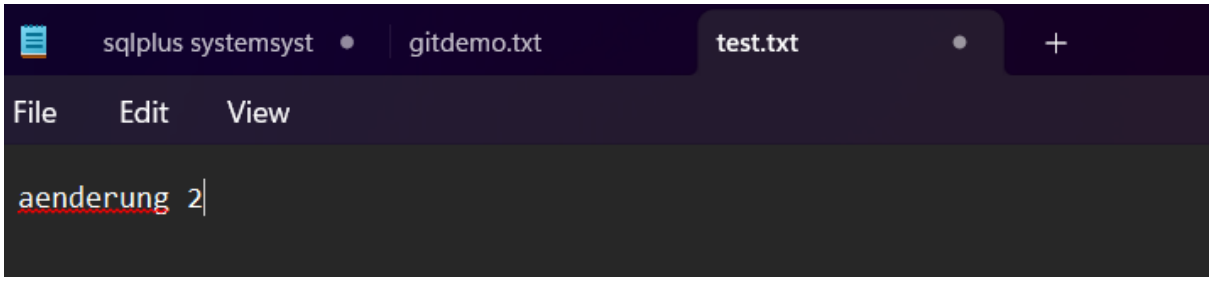
11. `add` the file change

```
mits\exercise> git add test.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

12. What does `git status` look like now?

```
mits\exercise> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   test.txt
```

13. Change the file again



14. Make a `commit`

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
mits\exercise> git commit -m "UE00: Basic-Commit2"
[master 3dca400] UE00: Basic-Commit2
 1 file changed, 1 insertion(+)
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

15. What does the `status` look like now? The `log`?

```
mits\exercise> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

16. Add and commit the newest change

```
mits\exercise> git commit -m "UE00: Basic-Commit"
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-com
```

2.2 Basic-Staging

1. What's the content of `file.txt`?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-staging\exercise> cat .\file.txt
1
```

2. Overwrite the content in `file.txt`: `echo 2 > file.txt` to change the state of your file in the working directory (or `sc file.txt '2'` in PowerShell)

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
ging\exercise> sc file.txt '2'
```

3. What does `git diff` tell you?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
ging\exercise> git diff
diff --git a/file.txt b/file.txt
index d00491f..0cfbf08 100644
--- a/file.txt
+++ b/file.txt
@@ -1 +1 @@
-1
+2
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

4. What does `git diff --staged` tell you? why is this blank?

```
ging\exercise> git diff --staged
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

Because i have not added something to the stage yet.

5. Run `git add file.txt` to stage your changes from the working directory.

```
ing\exercise> git add file.txt
2    .Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

6. What does `git diff` tell you?

```
ging\exercise> git diff
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

7. What does `git diff --staged` tell you?

```
ging\exercise> git diff --staged
diff --git a/file.txt b/file.txt
index d00491f..0cfbf08 100644
--- a/file.txt
+++ b/file.txt
@@ -1 +1 @@
-1
+2
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

8. Overwrite the content in `file.txt`: `echo 3 > file.txt` to change the state of your file in the working directory (or `sc file.txt '3'` in PowerShell).

```
ging\exercise> sc file.txt '3'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

9. What does `git diff` tell you?

```
ging\exercise> git diff
diff --git a/file.txt b/file.txt
index 0cfbf08..00750ed 100644
--- a/file.txt
+++ b/file.txt
@@ -1 +1 @@
-2
+3
```

10. What does `git diff --staged` tell you?

```
ging\exercise> git diff --staged
diff --git a/file.txt b/file.txt
index d00491f..0cfbf08 100644
--- a/file.txt
+++ b/file.txt
@@ -1 +1 @@
-1
+2
```

11. Explain what is happening

git diff -> shows the changes when file input got changed to '3',
git diff –staged -> Still shows the change that was pushed to the staging area when the file still had the input '3'.

12. Run `git status` and observe that `file.txt` are present twice in the output.

```
ging\exercise> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file.txt

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

PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

13. Run `git restore --staged file.txt` to unstage the change

```
ging\exercise> git restore --staged file.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-st
```

14. What does `git status` tell you now?

```
ging\exercise> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

15. Stage the change and make a commit

```
ging\exercise> git commit -m "UE00: Basic-Staging1"
[master 4c56898] UE00: Basic-Staging1
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-s
```

16. What does the log look like?

```
ging\exercise> git log
commit 4c5689851ee8595eaedf7ae9c4d2fac47cb54f9f (HEAD -> master)
Author: git-katas trainer bot <git-katas@example.com>
Date:   Mon Sep 16 16:13:17 2024 +0200

    UE00: Basic-Staging1

commit 9efc768829057dcef8135adc63ee870d604d7c3a
Author: git-katas trainer bot <git-katas@example.com>
Date:   Mon Sep 16 16:02:19 2024 +0200

    1
```

17. Overwrite the content in `file.txt`: `echo 4 > file.txt` (or `sc file.txt '4'` in PowerShell)

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
ging\exercise> sc file.txt '4'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

18. What is the content of `file.txt`?

```
ging\exercise> cat .\file.txt
4
```

19. What does `git status` tell us?

```
ging\exercise> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

20. Run `git restore file.txt`

```
ging\exercise> git restore file.txt
Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
```

21. What is the content of `file.txt`?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-mas
ging\exercise> cat .\file.txt
3
```

22. What does `git status` tell us?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-sta
ging\exercise> git status
On branch master
nothing to commit, working tree clean
```

2.3 Basic-Branching

1. Use `git branch` to see the two branches that are relevant for this exercise

```
nching\exercise> git branch
* master
  second-branch
```

2. What branch are you on?

*** master branch**

3. Use `git branch mybranch` to create a new branch called _*mybranch*_

```
nching\exercise> git branch mybranch
     \Users\Luka Pacar\Downloads\git-ka
```

3. Use `git branch` again to see the new branch created.

```
nching\exercise> git branch
* master
  mybranch
  second-branch
```

4. Use `git switch mybranch` to switch to your new branch.

```
nching\exercise> git switch mybranch
Switched to branch 'mybranch'
```

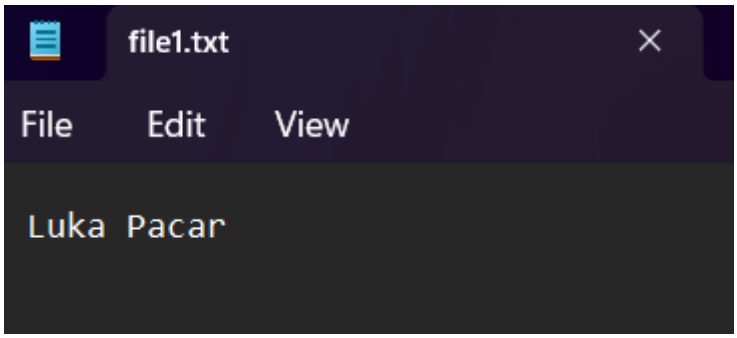
5. How does the output from `git status` change when you switch between the _*master*_ and the new branch that you have created?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> git switch mybranch
Switched to branch 'mybranch'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> git status
On branch mybranch
nothing to commit, working tree clean
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> git switch master
Switched to branch 'master'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> git status
On branch master
nothing to commit, working tree clean
```

6. How does the workspace change when you change between the two branches?

```
nching\exercise> git switch master
Already on 'master'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> dir


    Directory: C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-mas
    ter\basic-branching\exercise


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         9/16/2024   4:19 PM              7 dummy.txt


PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> git switch mybranch
Switched to branch 'mybranch'
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-bra
nching\exercise> dir


    Directory: C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-mas
    ter\basic-branching\exercise


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         9/16/2024   4:19 PM              7 dummy.txt
```

8. Make sure you are on your _*mybranch*_ branch before you continue.
9. Create a file called `file1.txt` with your name.



10. `Add` the file and `commit` with this change.

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-
nching\exercise> git add .\file1.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-
nching\exercise> git commit -m "Basic-Branching1"
[mybranch dc5d56f] Basic-Branching1
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-
```

11. Use `git log --oneline --graph` to see your branch pointing to the new commit.

```
nching\exercise> git log --oneline --graph
* dc5d56f (HEAD -> mybranch) Basic-Branching1
* 14f03d8 (second-branch, master) dummy commit
```

12. Switch back to the branch called _*master*_.

```
nching\exercise> git switch master
Switched to branch 'master'
```

13. Use `git log --oneline --graph` and notice how the commit you made on the _*mybranch*_ branch is missing on the _*master*_ branch.

```
nching\exercise> git log --oneline --graph
* 14f03d8 (HEAD -> master, second-branch) dummy commit
```

14. Make a new file called `file2.txt` and commit that file.

15. Use `git log --oneline --graph --all` to see your branch pointing to the new commit, and that the two branches now have different commits on them.

```
nching\exercise> git add file2.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-
nching\exercise> git commit -m "Basic-Branching-on-master1"
[master 806d91a] Basic-Branching-on-master1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file2.txt
```

17. Switch to your branch _*mybranch*_.

```
nching\exercise> git switch mybranch
Switched to branch 'mybranch'
```

18. What happened to your working directory? Can you see your `file2.txt`?

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basi
nching\exercise> dir


    Directory: C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas
    ter\basic-branching\exercise


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         9/16/2024   4:19 PM              7 dummy.txt
-a----         9/16/2024   4:35 PM             10 file1.txt
```

19. Use `git diff mybranch master` to see the difference between the two branches.

```
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\basic-br
nching\exercise> git diff mybranch master
diff --git a/file1.txt b/file1.txt
deleted file mode 100644
index 58f31ff..0000000
--- a/file1.txt
+++ /dev/null
@@ -1 +0,0 @@
-Luka Pacar
\ No newline at end of file
diff --git a/file2.txt b/file2.txt
new file mode 100644
index 0000000..e69de29
```

2.4 Fast-Forward-Merge

1. Create a (feature)branch called `feature/uppercase` (yes, `feature/uppercase` is a perfectly legal branch name, and a common convention).

```
exercise> git branch feature/uppercase
```

2. Switch to this branch

```
git switch feature/uppercase
```

3. What is the output of `git status`?

```
exercise> git status
On branch feature/uppercase
nothing to commit, working tree clean
```

4. Edit the greeting.txt to contain an uppercase greeting

5. Add `greeting.txt` files to staging area and commit

```
> git add .\greeting.txt
```

6. What is the output of `git branch`?

```
exercise> git branch
* feature/uppercase
  master
```

7. What is the output of `git log --oneline --graph --all`

```
exercise> git log --oneline --graph --all
* 643e3f8 (HEAD -> feature/uppercase, master) Add content to greeting.txt
* 6726e69 Add file greeting.txt
PS C:\Users\Luka Pacar\Downloads\git-katas-master\git-katas-master\ff-merge\
```

Remember: You want to update the master branch so it also has all the changes currently on the feature branch. The command 'git merge [branch name]' takes one branch as argument from which it takes changes. The branch pointed to by HEAD (currently checked out branch) is then updated to also include these changes.

8. Switch to the `master` branch

```
exercise> git switch master
M       greeting.txt
Switched to branch 'master'
```

9. Use `cat` to see the contents of the greetings

```
exercise> cat .\greeting.txt
Hello
```

10. Diff the branches

```
exercise> git diff feature/uppercase master
```

11. Merge the branches

```
git merge feature/uppercase
```

12. Use `cat` to see the contents of the greetings

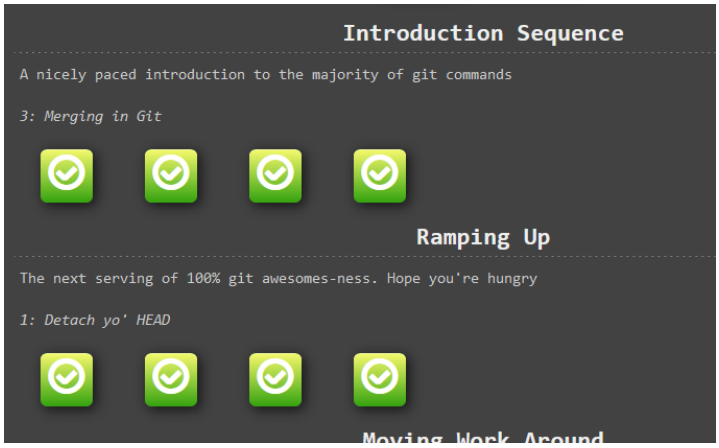
```
exercise> cat .\greeting.txt
Hello
```

13. Delete the uppercase branch

```
exercise> git branch -d feature/uppercase
Deleted branch feature/uppercase (was 643e3f8).
```

3 Git-Branching

3.1 main

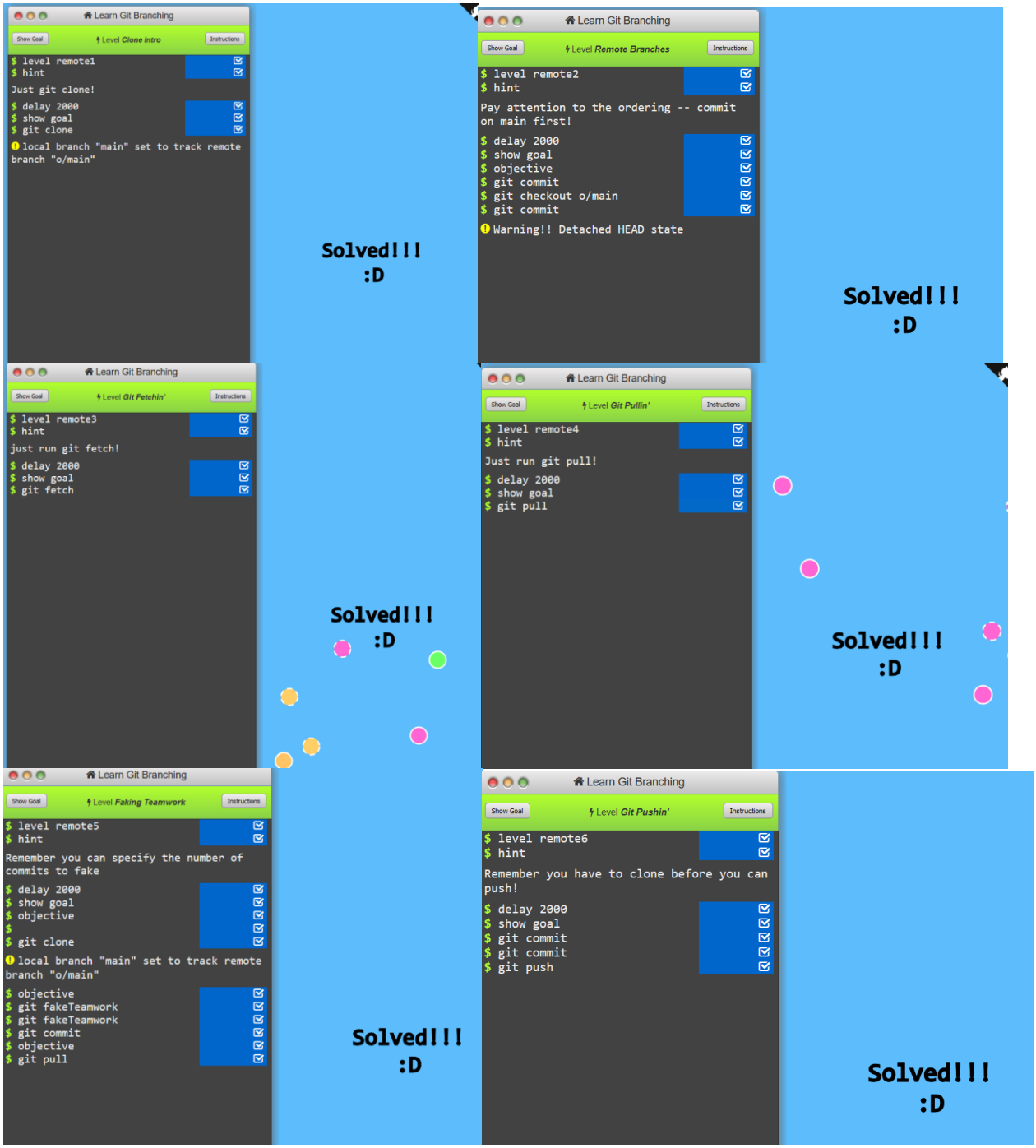


3.1.1 Zeile 1

3.1.2 Zeile 2


Learn Git Branching
Level Detach yo' HEAD
$ level rampup1
$ hint
Use the label (hash) on the commit for help!
$ delay 2000
$ show goal
$ objective
$ git checkout C4
Solved!!!
:D
Learn Git Branching
Level Relative Refs (^)
$ level rampup2
$ hint
Remember the Caret (^) operator!
$ delay 2000
$ show goal
$ git checkout bugFix
$ git checkout HEAD^
Solved!!!
:D
Learn Git Branching
Level Relative Refs #2 (~)
$ level rampup3
$ hint
You'll need to use at least one direct reference (hash) to complete this level
$ delay 2000
$ show goal
$ git branch -f main C6
$ git branch -f bugFix Head~3
The ref Head~3 does not exist or is unknown
$ git branch -f bugFix HEAD~3
Commit C2 does not have a ~3
$ git branch -f bugFix HEAD~2
$ git checkout HEAD^
Solved!!!
:D
Learn Git Branching
Level Reversing Changes in Git
level rampup4
hint
Notice that revert and reset take different arguments.
delay 2000
show goal
git branch -f local Head^
The ref Head^ does not exist or is unknown
git branch -f local HEAD^
git checkout pushed
git revert
expect at least 1 argument(s) for with git revert
git revert pushed
C0
C1
C2
C2'
main
local
pushed*
Solved!!!
:D

3.2 remote


Learn Git Branching
Level Clone Intro
$ level remote1
$ hint
Just git clone!
$ delay 2000
$ show goal
$ git clone
local branch "main" set to track remote branch "o/main"
Solved!!!
:D
Learn Git Branching
Level Remote Branches
$ level remote2
$ hint
Pay attention to the ordering -- commit on main first!
$ delay 2000
$ show goal
$ objective
$ git commit
$ git checkout o/main
$ git commit
Warning!! Detached HEAD state
Solved!!!
:D
Learn Git Branching
Level Git Fetchin'
$ level remote3
$ hint
just run git fetch!
$ delay 2000
$ show goal
$ git fetch
Solved!!!
:D
Learn Git Branching
Level Git Pullin'
$ level remote4
$ hint
Just run git pull!
$ delay 2000
$ show goal
$ git pull
Solved!!!
:D
Learn Git Branching
Level Faking Teamwork
$ level remote5
$ hint
Remember you can specify the number of commits to fake
$ delay 2000
$ show goal
$ objective
$
$ git clone
local branch "main" set to track remote branch "o/main"
$ objective
$ git fakeTeamwork
$ git fakeTeamwork
$ git commit
$ objective
$ git pull
Solved!!!
:D
Learn Git Branching
Level Git Pushin'
$ level remote6
$ hint
Remember you have to clone before you can push!
$ delay 2000
$ show goal
$ git commit
$ git commit
$ git push
Solved!!!
:D