

Aliaksandra_Bacharova

2) Pass practical course <https://www.katacoda.com/courses/git>


1)

The screenshot shows the 'Congratulations!' page for the first scenario on Katacoda. The page has a blue header with the O'Reilly Katacoda logo, 'KATACODA OVERVIEW & SOLUTIONS', and a 'TRY O'REILLY' button with a 'LOG IN >' link. The main content area is blue and features the text 'Congratulations! You've completed the scenario!' and a 'Scenario Rating' of five stars. Below this, a paragraph explains that the scenario has explained how to initialise a repository and commit files, and that it has been added to the user's scrapbook. There are three 'Share Your Success' buttons (LinkedIn, Twitter, Facebook) and two buttons: 'DOWNLOAD SCENARIO' and 'NEXT SCENARIO'. A dark terminal window is visible on the right side of the page.

2)

The screenshot shows the 'Congratulations!' page for the second scenario on Katacoda. The page has a blue header with the O'Reilly Katacoda logo, 'KATACODA OVERVIEW & SOLUTIONS', and a 'CLAIM YOUR PROFILE' button with a 'LOG OUT >' link. The main content area is blue and features the text 'Congratulations! You've completed the scenario!' and a 'Scenario Rating' of five stars. Below this, a paragraph explains that the exercise has demonstrated how to view changes and commit them to the repository, and that it has been added to the user's scrapbook. There are three 'Share Your Success' buttons (LinkedIn, Twitter, Facebook) and two buttons: 'RESTART SCENARIO' and 'NEXT SCENARIO'. A dark terminal window is visible on the right side of the page, showing some code and terminal output.

3)

 KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE [LOG OUT >](#)

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

This scenario demonstrated how you can push / pull changes between different repositories. By continually using `git push` and `git pull` you can ensure everyone has access to the latest version of the code base.


This scenario has been added to your scrapbook where you can review the examples and commands you executed.

[Share Your Success](#) [Share Your Success](#) [Share Your Success](#)

[RESTART SCENARIO](#) [NEXT SCENARIO](#)

```
[1;36mHEAD -> [ESC][mESC[1;32mmaster[ESC][mESC[
```

4)

 KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE [LOG OUT >](#)

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★


This scenario demonstrated how you can reset and revert changes you've made and how to go back to a previous state.

This scenario has been added to your scrapbook where you can review the examples and commands you executed.

[Share Your Success](#) [Share Your Success](#) [Share Your Success](#)

[RESTART SCENARIO](#) [NEXT SCENARIO](#)

5)

**O'REILLY**
katacoda

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE [LOG OUT >](#)

×

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★


In this scenario we've explored the different ways of handling merges. We've seen how to use `git fetch` and `git merge` to pull remote changes, how to resolve conflicts with other commits and finally how to keep our git log and commits clean using `git rebase`. Merging is an important part of Git, a topic we'll explore in more depth in future scenarios.

This scenario has been added to your scrapbook where you can review the examples and commands you executed.

[Share Your Success](#) [Share Your Success](#) [Share Your Success](#)

[RESTART SCENARIO](#) [NEXT SCENARIO](#)

6)

**O'REILLY**
katacoda

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE [LOG OUT >](#)

×

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

In this scenario we've explored how you can work with branches which are ideal for prototyping and experiments as they can be quickly created and thrown away.

This scenario has been added to your scrapbook where you can review the examples and commands you executed.

[Share Your Success](#) [Share Your Success](#) [Share Your Success](#)

[RESTART SCENARIO](#) [NEXT SCENARIO](#)

7)

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE
LOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

With the comprehensive history of your source code within the Git repository allows you to go back in time and identify when issues occurred, such as bugs or performance issues. Commands like `git diff` allow you to quickly compare changes, while `git bisect` helps you search to identify the cause.

This scenario has been added to your scrapbook where you can review the examples and commands you executed.

Share Your Success

Share Your Success

Share Your Success

RESTART SCENARIO

NEXT SCENARIO

```

n, pharetra a.</li>
eu vulputate magna eros eu erat.</li>
i, condimentum sed.</li>

lor sit amet, consectetur adipiscing elit.
unt mauris eu risus.</li>
tor dapibus neque.</li>
uis dui placerat ornare. Pellentesque odio
us, neque id cursus faucibus, tortor neque
dapibus neque
et est et sapien ullamcorper pharetra. Vest
ermentum dolor</li>

s, neque id cursus faucibus, tortor neque e
apibus neque
t est et sapien ullamcorper pharetra. Vesti

```

8)

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE
LOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

The ability to re-write history is useful to keep your history of the repository clean and accurate. This will help in future to indicate reasons for change or to debug problems.

This scenario has been added to your scrapbook where you can review the examples and commands you executed.

Recommendation

You should only rebase commits that have not been shared with other people via push. Rebasing commits causes their commit-ids to change which can result in losing future commits.

Share Your Success

Share Your Success

Share Your Success

RESTART SCENARIO

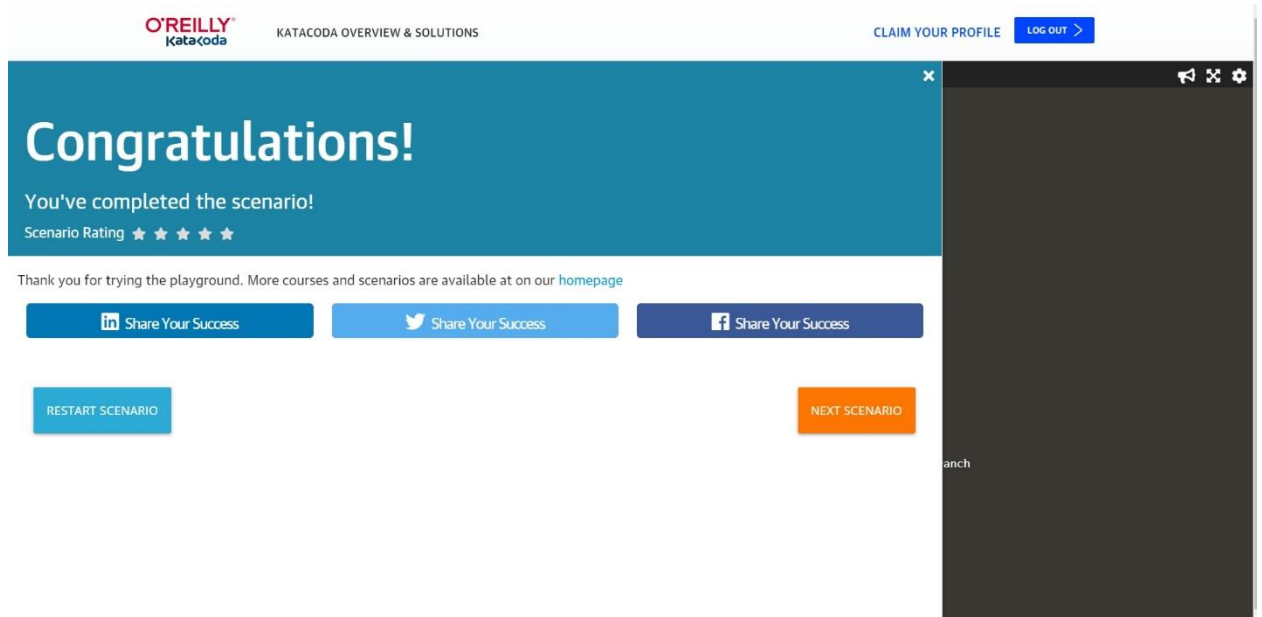
NEXT SCENARIO

```

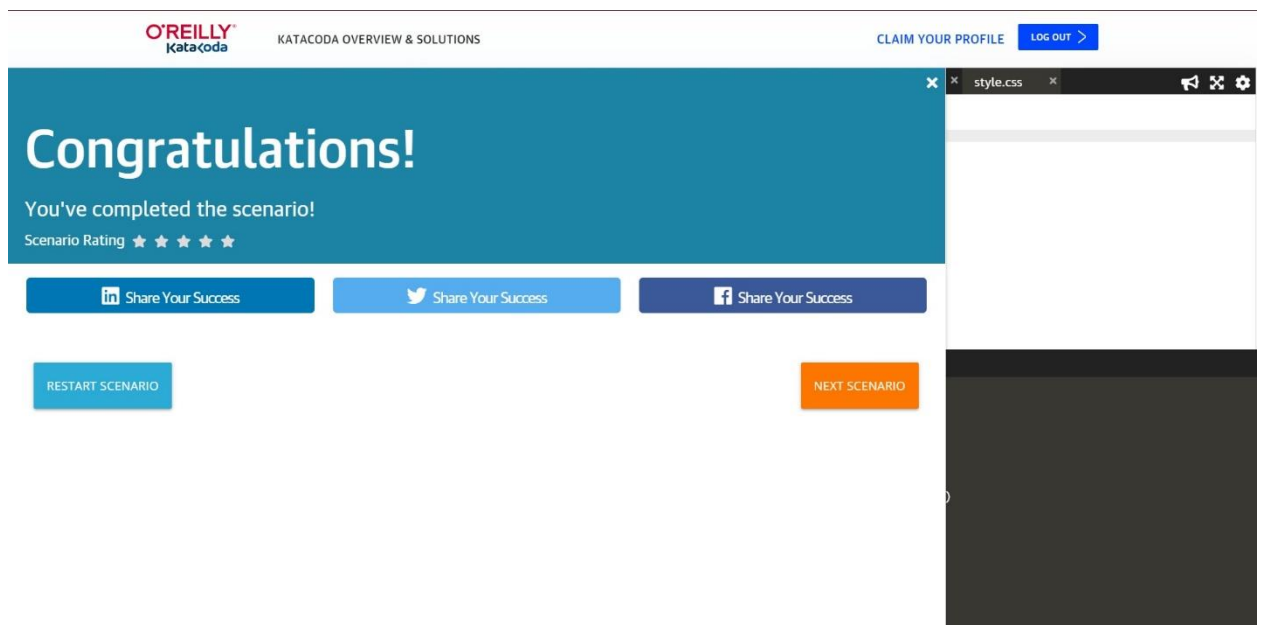
SC[33m]ESC[m File 4

```


9)



3) Pass practical scenario <https://www.katacoda.com/aossama/scenarios/git-scm-lab-101>



4) Pass practical scenario <https://www.katacoda.com/aossama/scenarios/git-scm-lab-102>



KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILELOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

The most important takeaways from this lab are:

- `git clone` is used to create a copy of a target repo
- `git remote` is used to create, view, and delete connections to other repositories
- `git push` is used to propagate changes on the local repository to remote repository
- `git fetch` is used to download objects and refs from another repository
- `git pull` is used to fetch from and integrate with another repository or a local branch

Share Your Success

Share Your Success

Share Your Success


RESTART SCENARIO

NEXT SCENARIO

Editor

the working tree.

5) Pass practical scenario <https://www.katacoda.com/aossama/scenarios/git-scm-lab-201>



KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILELOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

The most important takeaways from this lab are:

- `git checkout` can be used to create branches, switch branches, and checkout remote branches
- `git branch` commands primary functions are to create, list, rename and delete branches
- `git tag` is used to create semantic version number identifier tags that correspond to software release cycles
- `git merge` is used to combine multiple sequences of commits into one unified history
- `git rebase`
- `git reset`

Share Your Success

Share Your Success

Share Your Success

RESTART SCENARIO

NEXT SCENARIO

Editor

': Could not resolve host: git.itworx.cloud

6) Pass practical scenario <https://www.katacoda.com/aossama/scenarios/git-scm-lab-202>

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

Now that you have an understanding of the projects you will use throughout this course, let's get started!



Share Your Success



Share Your Success



Share Your Success

RESTART SCENARIO

NEXT SCENARIO