



DevOps Lab

VERSION CONTROL SYSTEMS: GIT

Home task

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

CONFIDENTIAL | Effective Date: 25-Jan-19

TASK

O'REILLY
katakoda

LEARN CREATE FOR LEAD GEN FOR EVENTS FOR DEVREL FOR TEAMS

CLAIM YOUR PROFILE

LOG OUT >

Scenario 1 - Committing Files

Learn how to initialise a repository and start committing files.

Repeat Scenario

Scenario 2 - Committing Changes

Learn how to compare and commit changes.

Repeat Scenario

Scenario 3 - Working Remotely

Learn how to share your changes with others and access other people's changes.

Repeat Scenario

Scenario 4 - Undoing Changes

Learn how to undo changes when required.

Repeat Scenario

Scenario 5 - Fixing Merge Conflicts

Learn how to fix merge conflicts then they occur.

Repeat Scenario

Scenario 6 - Experiments Using Branches

Learn how to create branches of master for experimenting and prototyping ideas.

Repeat Scenario

Scenario 7 - Finding Bugs

Learn how to find commits related to bugs and issues with code.

Repeat Scenario

Scenario 8 - Being Picky With Git

Learn how to pick certain commits and changes from other repositories.

Repeat Scenario

Scenario 9 - Re-writing History

Learn how to re-write history when required.

Repeat Scenario



Playground

Use Git in a safe playground environment.

Explore Playground



Your Content Here

Add your own content to Katakoda and share your experience or product with the community

Create Content

O'REILLY
katakoda

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE

LOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★ ★ ★ ★ ★

This scenario has explained how you can initialise a repository and then commit files to it. In the next scenario we'll investigate how to compare and commit changes to these files. In future scenarios we'll cover how to share these changes with other people.

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

DOWNLOAD SCENARIO

NEXT SCENARIO

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


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

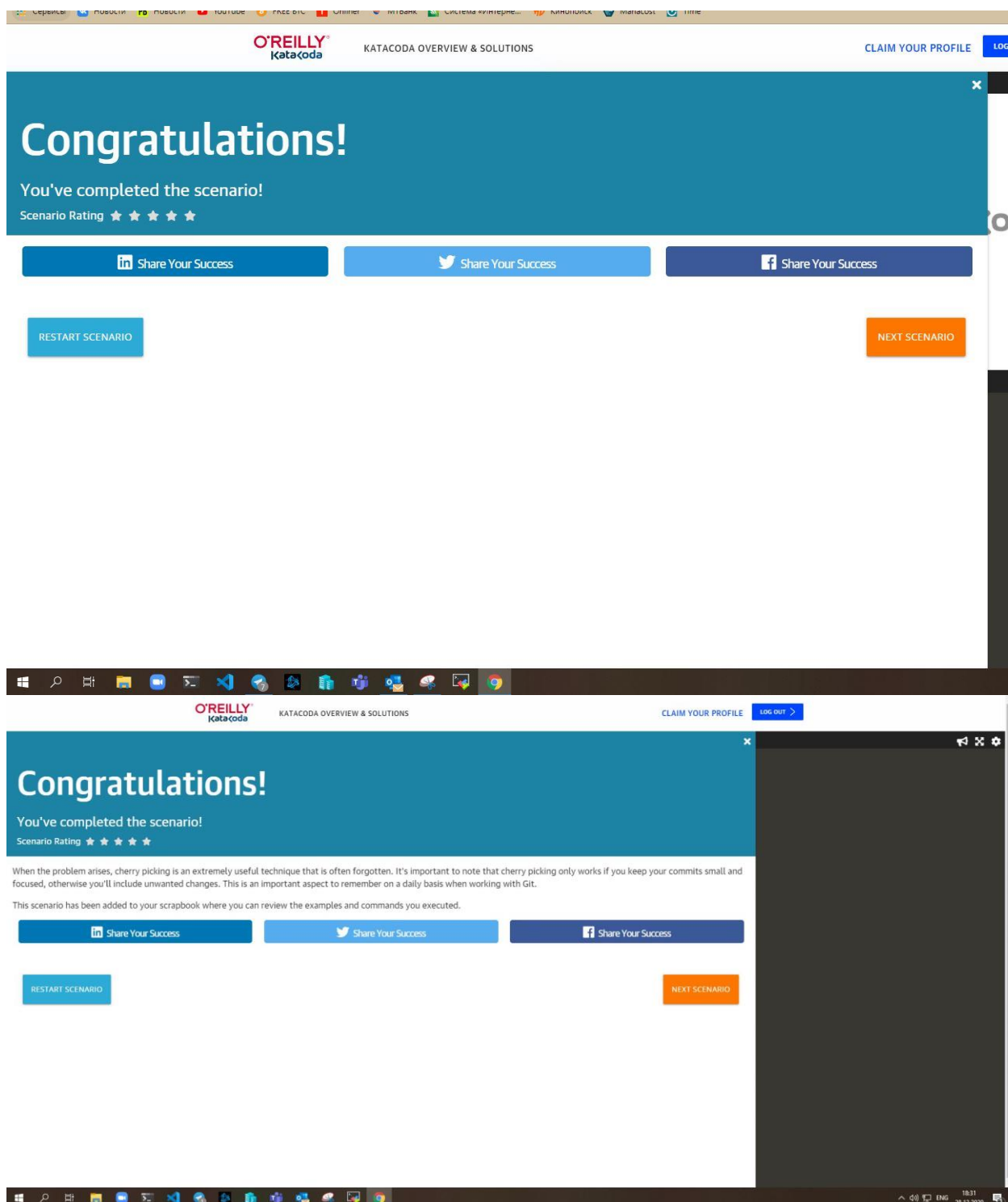
 Share Your Success


 Share Your Success

 Share Your Success

RESTART SCENARIO

NEXT SCENARIO




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


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


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


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](#)

Bug #55


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](#)

"Commit To Revert"

O'REILLY
Kata-coda

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILELOG 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

Windows taskbar

Terminal window showing git commands and output:

```
ESC[1;32mmasterESC[mESC[33mESC[mESC[1;31morigin/maste
```

O'REILLY
Kata-coda

KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILELOG OUT >

Congratulations!

You've completed the scenario!

Scenario Rating ★★★★★

This exercise has demonstrated how you can view your changes and commit them to the repository. In the next scenario we'll investigate how to share these changes with other people.

This exercise has been added to your scrapbook where you can review the example and code you written at any point.

Share Your Success

Share Your Success

Share Your Success

RESTART SCENARIO

NEXT SCENARIO

Windows taskbar

Terminal window showing git commands and output:

```
ESC[1;32mmasterESC[mESC[33mESC[m
```

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