View All Pages

🖥

Git Intro

Branching:

In Git, you can create branches of your code for trying out new features or experimenting with your code without the risk of breaking the functionality of the main code (in the master branch). If you want to keep the changes of the new branch, you can merge the code in that branch back into the master branch.

To create a new branch:

```
$ git branch branchname
```

To move into a new branch to start working on that branch:

```
$ git checkout branchname
```

Once you finish working on that branch and want to move to the master branch and merge the changes:

```
$ git checkout master
$ git merge branchname
```

After the merge, any changes made in the new branch will be present in your master branch. Here, you may also run into merge conflicts and have to manually resolve them and re-commit the change.

Checklist:

Congratulations! Now, you should be able to:

- Create a new repository using Git for your project
- Enter commits to keep track of changes to your code
- Pull changes from a remote repository
- Push changes to a remote repository
- Create new branches to experiment with your code and implement new features

Other Useful Commands

There are many other useful commands you may need for working with git. Some are listed below. Please check the links under "Useful Guides & Resources" for more information, tutorials, and documentation for Git.

If you want to check with files have been changed since the last commit, or if you want to see new files or files staged for commit, you can use

```
$ git status
```

to check the status of your repository.

To see the changes in the files you've made since your last commit line by line, use the following command:
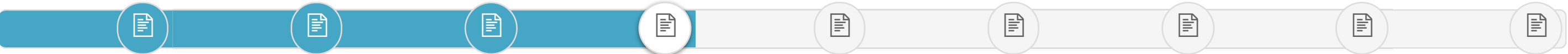
```
$ git diff
```

To view the commit history use:

```
$ git log
```

Useful Guides & Resources:

- The Official Git Homepage
- Interactive Git Tutorial
- GitHub Introduction & GitHub Help Page
- Atlassian Git Tutorials and Training
- Basic Git Commands

This page was generated by GitHub Pages.