GitHub:

Release the Open Source Kraken!

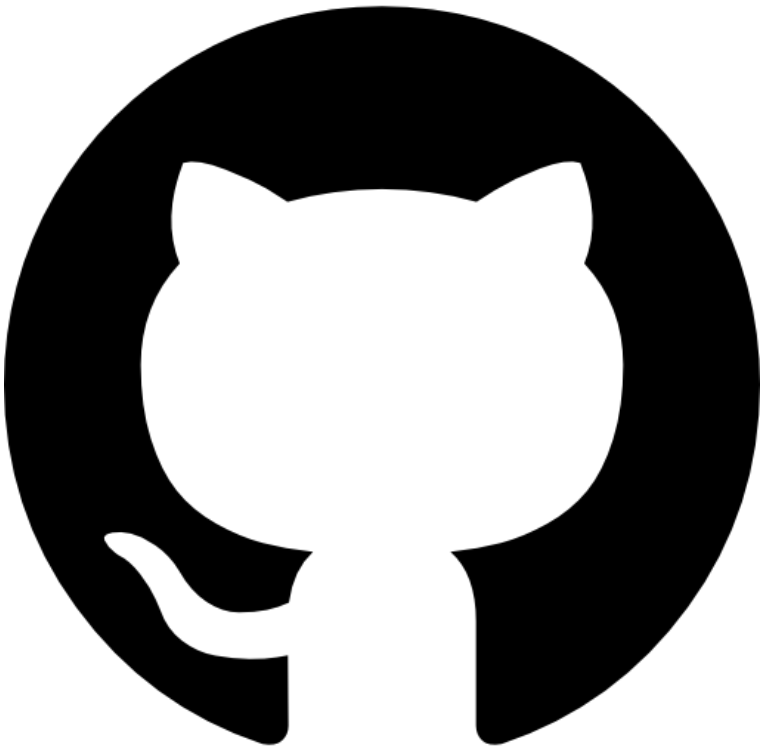
Created with Artificial Intelligence

Compiled by Thiago Michaelsen

GitHub

Release the Open Source Kraken!

This ebook will guide you through the essential aspects of GitHub, from basic commands to advanced project management strategies. We'll use real-world examples to illustrate each concept, ensuring you can confidently navigate the world of version control.



Disclaimer

This ebook was written with the assistance of artificial intelligence. It has not been reviewed or edited by a human author. While the AI has been trained on a vast dataset, it is still under development and may produce inaccurate, incomplete, or outdated information.

Please be aware that:

- Some content may be incorrect or factually flawed.
- The information provided may not be up-to-date.
- This ebook should not be considered a definitive source of information.

We strongly encourage you to:

- Verify any information presented in this ebook through reputable sources.
- Consult with qualified professionals for advice or guidance on any specific topic discussed.

By using this ebook, you acknowledge and agree that:

- The authors and publishers are not responsible for any errors, omissions, or inaccuracies in the content.
- The information provided is for general informational purposes only and does not constitute professional advice.

We appreciate your understanding and hope this ebook provides you with useful information.

01

SETTING UP YOUR GITHUB ACCOUNT

Your Gateway to Collaboration: Setting Up Your GitHub Account

Setting Up Your GitHub Account

Before you dive into the technicalities, you need a GitHub account. This is where you'll store your code, track changes, and collaborate with others.

1. Head to https://github.com and create a free account.

2. Choose a username that reflects your identity or project preferences.

3. Set up a strong password for security.

02

GIT BASICS: THE COMMAND LINE INTERFACE

Speaking the Language of Git: Essential Commands

Git Basics: The Command Line Interface

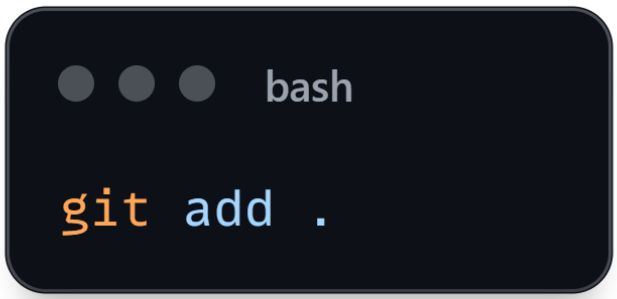
Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

1. git init : Initializes a Git repository in your project directory.

Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

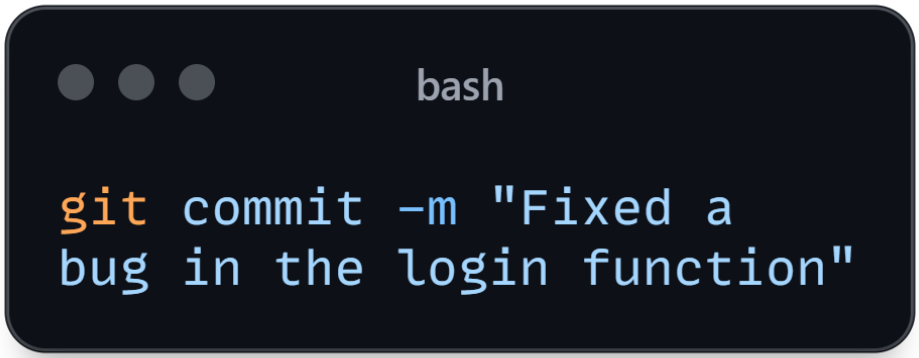
2. git add . : Stages all changes in your working directory for commit.



Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

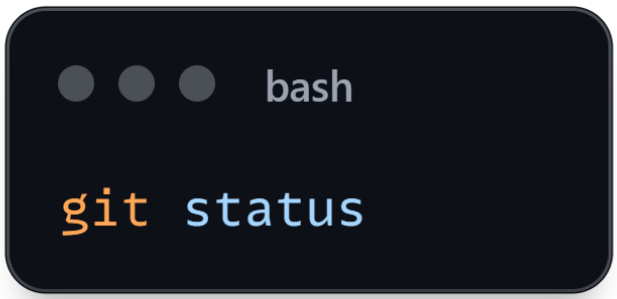
3. git commit -m "Commit message": Records changes with a descriptive message.



Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

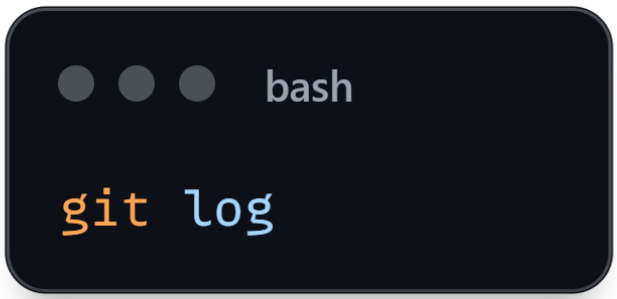
4. git status: Checks the current status of your repository, showing modified or untracked files.



Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

5. git log: Displays a history of commits made to the repository.



Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

6. git branch: Creates, lists, or switches between branches.

bash

```bash
git branch new-feature
# Creates a new branch

git checkout new-feature
# Switches to the new-feature branch
```

Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

7. git merge <branch-name>: Combines changes from another branch into your current branch.

```bash
git merge main
# Merges changes from the 'main'
# branch into the current branch
```

Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

8. git push origin <branch-name>: Uploads local changes to a remote repository on GitHub.

```bash
git push origin main
# Pushes the 'main' branch to
# the 'origin' remote repository
```

Git Basics: The Command Line Interface

Git is the version control system that powers GitHub. Familiarize yourself with these fundamental commands to manage your code:

9. git pull origin <branch-name>: Downloads changes from the remote repository and integrates them into your local branch.

```bash
git pull origin main
# Pulls the latest changes
# from the 'main' branch
# on the remote repository
```

03

WORKING WITH REMOTE REPOSITORIES

Beyond Your Local Space: Collaborating with Remote Repositories

Working with Remote Repositories

Remote repositories on GitHub allow you to share your code, collaborate with others, and back up your work.

1. git clone <repository URL>: Creates a local copy of a remote repository.

```bash
git clone https://github.com/username/repository-name.git
```

Working with Remote Repositories

Remote repositories on GitHub allow you to share your code, collaborate with others, and back up your work.

2. git remote add origin <repository URL>: Adds a remote repository to your local project.

```bash
git remote add origin https://github.com/username/repository-name.git
```

Working with Remote Repositories

Remote repositories on GitHub allow you to share your code, collaborate with others, and back up your work.

3. git remote -v: Lists all remote repositories connected to your local project.

```bash
git remote -v
```

Working with Remote Repositories

Remote repositories on GitHub allow you to share your code, collaborate with others, and back up your work.

4. git push origin <branch-name>: Pushes your local changes to the remote repository.

```bash
git push origin main
# Pushes the 'main' branch
# to the remote repository.
```

Working with Remote Repositories

Remote repositories on GitHub allow you to share your code, collaborate with others, and back up your work.

5. git pull origin <branch-name>: Retrieves changes from the remote repository and integrates them into your local branch.

```bash
git pull origin main
# Pulls the latest changes
# from the 'main' branch
# on the remote repository
```

04

BRANCHING FOR FEATURE DEVELOPMENT

Isolating Your Code: The Power of Branching

Branching for Feature Development

Branching allows you to develop features independently without affecting the main codebase.

1. git branch <branch-name>: Create a new branch for your feature development.

```bash
git branch new-feature
# Creates a branch
# named 'new-feature'
```

Branching for Feature Development

Branching allows you to develop features independently without affecting the main codebase.

2. git checkout <branch-name>: Switch to the new branch.

```bash
git checkout new-feature
# Switches to the
# 'new-feature' branch
```

Branching for Feature Development

Branching allows you to develop features independently without affecting the main codebase.

3. git merge <branch-name>: Integrate changes from a branch into the current branch.

bash

```
git merge new-feature
# Merges changes
# from the 'new-feature' branch
# into the current branch
```

Branching for Feature Development

Branching allows you to develop features independently without affecting the main codebase.

4. git branch -d <branch-name>: Deletes a branch after it's been merged.

```bash
git branch -d new-feature
# Deletes the 'new-feature' branch
```

05

PULL REQUESTS: COLLABORATING SEAMLESSLY

Collaboration Made Easy: Working with Pull Requests

GitHub Pull Requests: Collaborating Seamlessly

Pull requests are the heart of collaboration on GitHub. They allow you to propose changes and review code before merging it into the main branch.

1. **Create a Pull Request:** Navigate to the "Pull requests" tab on your repository and click "New pull request." Select the branches you want to merge.

2. **Write a Descriptive Title and Description:** Clearly explain the purpose and changes in your pull request.

3. **Review and Discuss:** Collaborators can review your code, suggest changes, and discuss the implementation.

4. **Merge the Pull Request:** Once all reviews and comments are addressed, the pull request can be merged into the target branch.

06

PROJECT MANAGEMENT WITH GITHUB ISSUES

Keeping Track of Everything: Utilizing GitHub Issues

Project Management with GitHub Issues

GitHub Issues provide a powerful framework for managing tasks, bugs, and features within your project.

1. **Create an Issue:** Navigate to the "Issues" tab on your repository and click "New issue.“

2. **Assign Labels:** Use labels to categorize issues based on type (bug, feature, enhancement), priority, and area of the project.

3. **Assign Milestones:** Group issues into milestones to track progress on specific releases or projects.

4. **Assign Users:** Assign issues to specific users who are responsible for addressing them.

5. **Add Comments and Updates:** Leave comments to discuss issues, provide updates, and collaborate with team members.

07

GITHUB ACTIONS: AUTOMATING YOUR WORKFLOW

Streamline Your Process:
Leveraging GitHub Actions

GitHub Actions: Automating Your Workflow

GitHub Actions automate repetitive tasks, such as testing, building, and deploying your code.

1. **Create a Workflow File:** Create a .yml file in the .github/workflows directory of your repository.

2. **Define Triggers:** Specify when your workflow should run (e.g., on push, pull request, schedule).

3. **Configure Jobs:** Break down your workflow into jobs, which represent independent tasks.

4. **Define Steps:** Each job consists of steps that run specific commands or actions.

GitHub Actions: Automating Your Workflow

Example Workflow File:

bash

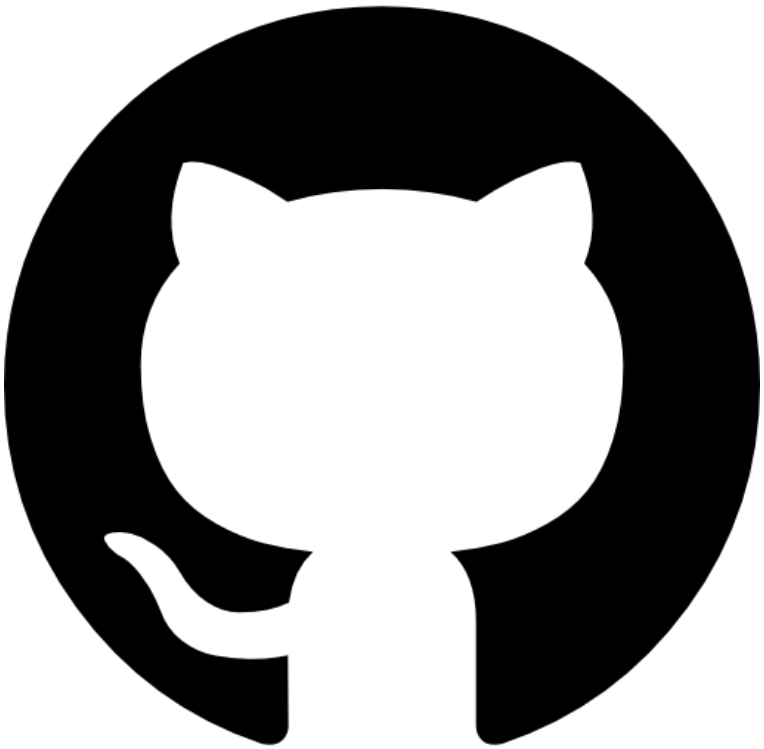
```yaml
name: CI
on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Install dependencies
        run: npm install
      - name: Run tests
        run: npm test
```

CONCLUSION

Conclusion

This ebook provided a comprehensive overview of GitHub, equipping you with the knowledge and skills to navigate version control, collaborate effectively, and manage projects efficiently.

As you continue your GitHub journey, explore advanced features like forks, releases, and project boards to further enhance your workflow and productivity.

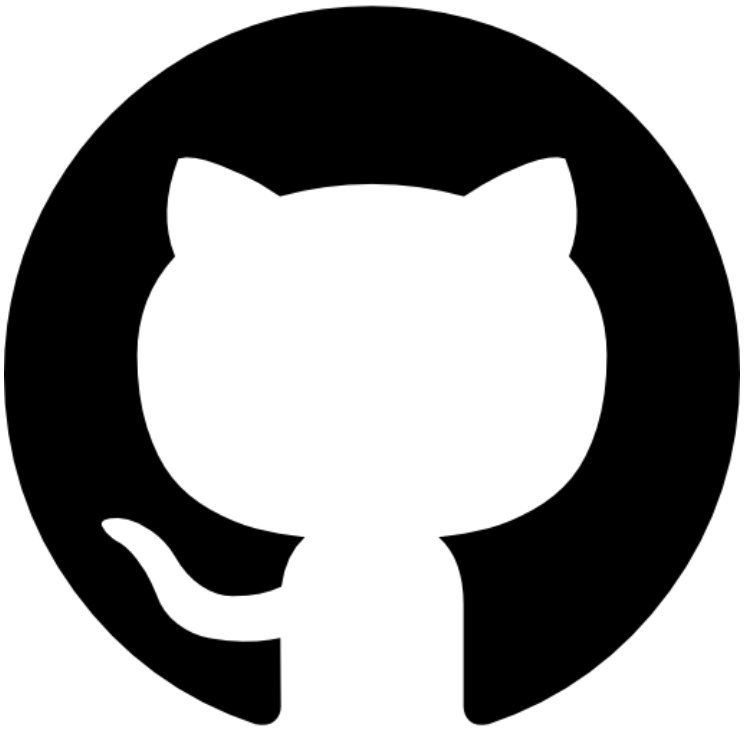


ACKNOWLEDGEMENTS

Acknowledgements

This content has been generated for didactic construction purposes, following DIO's Santander 2024 - AI Fundamentals for Devs bootcamp. There has been no validation of the content to demonstrate the generative capacity of an AI.

The step-by-step instructions for this project can be found on my GitHub, as well as some other works. You can find me clicking on the following icon.



TMMichaelsen