How To Compare Branches on GitHub?

Last Updated : 26 Aug, 2024

One important aspect of branch management is comparing branches to review differences in code, commits, or pull requests. Whether you're working on new features, resolving conflicts, or reviewing code, comparing branches is a common task in GitHub.

In this article, we will cover different ways to compare branches on GitHub and explain the tools GitHub provides for branch comparison.

Table of Content

Why Compare Branches on GitHub?

- **Code Reviews:** Before merging a feature branch, it's important to compare it with the main branch to review the differences.
- **Conflict Resolution:** Identifying differences between branches helps resolve conflicts when merging.
- **Progress Tracking:** Comparing branches shows what's been changed, added, or removed, helping you track the development process.
- **Releasing New Versions:** Compare release branches with the main branch to see what's included in a release.

GitHub provides built-in tools that simplify the process of comparing branches, making it easy to visualize changes and make informed decisions during

development.

Method 1: Using the GitHub Branch Comparison Tool

GitHub has a dedicated branch comparison feature that allows you to compare any two branches in a repository.

Steps to Compare Branches Using GitHub's Comparison Tool:

Step 1: Navigate to Your Repository

Open your repository on GitHub.

Step 2: Go to the Compare Page

- In the repository, click on the "Insights" tab, then select "Compare."
- Alternatively, you can manually navigate to the comparison URL by appending /compare to your repository's URL:

```
https://github.com/username/repository/compare
```

Step 3: Select the Base and Compare Branches

- The base branch is the branch you are comparing against (usually main or master).
- The compare branch is the branch you want to see the differences from (e.g., feature-branch).
- GitHub will display the differences between the two branches, including commits, files changed, and lines added/removed.

Step 4: Review the Differences

You can now see a detailed view of the changes, including:

- A list of commits that are unique to the compare branch.
- The files that have been added, modified, or deleted.
- Line-by-line code differences.

Method 2: Comparing Branches in Pull Requests

Pull requests (PRs) are GitHub's primary tool for managing and merging changes between branches. When you open a pull request, GitHub automatically provides a branch comparison between the base branch and the feature branch.

Steps to Compare Branches in a Pull Request

Step 1: Open a Pull Request

In your repository, click the "Pull Requests" tab, then click "New Pull Request."

Step 2: Select Base and Compare Branches

- Select the base branch (e.g., main) and the compare branch (e.g., feature-branch).
- GitHub will display a comparison similar to the one in the branch comparison tool.

Step 3: Review the Changes

- GitHub highlights the changes in files, shows commit history, and provides options for leaving comments and approving the changes.
- You can review these changes before merging the branches.

Method 3: Using Git Locally to Compare Branches

While GitHub's web interface is powerful, you can also compare branches using Git commands in your local environment.

Steps to Compare Branches Locally

Step 1: Open Your Terminal or Command Line

Navigate to your repository in your terminal.

Step 2: Use the git diff Command

Run the following command to compare two branches:

```
git diff branch1..branch2
```

For example, to compare the main branch with a feature-branch

```
git diff main..feature-branch
```

This shows the changes between the branches at the file level.

Step 3: View Commit Differences

If you want to see commit differences, use:

```
git log main..feature-branch
```

This command lists the commits that are present in the feature-branch but not in main.

While comparing branches locally gives you more control over the comparison process, GitHub's web interface is often more user-friendly and provides a visual overview of differences.

Understanding Branch Comparison Results

When comparing branches on GitHub, you'll see several key elements:

- **Commits:** A list of commits that are unique to the compare branch.
- **Files Changed:** The specific files that have been added, modified, or deleted.
- **Diff View:** Line-by-line changes in the code, showing what has been added, removed, or updated.

GitHub uses color coding in the diff view:

- **Green Lines:** Lines that have been added.
- **Red Lines:** Lines that have been removed.
- **Grey Lines:** Lines that have been unchanged or context lines.

This makes it easier to review the impact of changes before merging branches.

Best Practices for Branch Comparison

- **Perform Regular Comparisons:** Regularly compare branches, especially when working on long-term feature branches, to stay aligned with the main branch and avoid large conflicts.
- **Use Descriptive Branch Names:** Clearly named branches (e.g., feature/add-login) help you easily identify what is being compared.
- **Review Pull Requests Thoroughly:** When comparing branches as part of a pull request, take the time to review changes for code quality, security, and alignment with project goals.

- **Rebase Before Comparison:** If your branch is outdated, consider rebasing it onto the main branch before comparing. This ensures a cleaner diff view with fewer conflicts.
- **Keep Comparisons Small:** Smaller, frequent branch comparisons are easier to review and merge than large, infrequent ones.

Common Use Cases for Branch Comparison

- **Feature Development:** Comparing a feature branch against the main branch helps track progress and prepare for merging.
- **Bug Fixes:** When fixing a bug, comparing branches can help verify that the fix is isolated and does not introduce regressions.
- **Release Management:** Before releasing a new version, comparing the release branch against the main branch helps verify what changes will be included.
- **Conflict Resolution:** During complex merges, branch comparison helps identify and resolve conflicts efficiently.

Next Article

How To Compare Branches on GitHub?

Similar Reads

Non-linear Components

In electrical circuits, Non-linear Components are electronic devices that need an external power source to operate actively. Non-Linear Components are those that are changed with respect to the voltage and current...

11 min read

JavaScript Tutorial

JavaScript is a programming language used to create dynamic content for websites. It is a lightweight, cross-platform, and single-threaded programming language. It's an interpreted language that executes code line b...

11 min read

Web Development

Web development is the process of creating, building, and maintaining websites and web applications. It involves everything from web design to programming and database management. Web development is...

5 min read

Spring Boot Tutorial

Spring Boot is a Java framework that makes it easier to create and run Java applications. It simplifies the configuration and setup process, allowing developers to focus more on writing code for their applications. Th...

10 min read

Class Diagram | Unified Modeling Language (UML)

A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them. It helps everyone involved in a projectâ€”like...

12 min read

React Interview Questions and Answers

React is an efficient, flexible, and open-sourceÂ JavaScriptÂ library that allows developers to create simple, fast, and scalable web applications. Jordan Walke, a software engineer who was working for Facebook,...

15+ min read

Steady State Response

In this article, we are going to discuss the steady-state response. We will see what is steady state response in Time domain analysis. We will then discuss some of the standard test signals used in finding the response o...

9 min read

JavaScript Interview Questions and Answers

JavaScript (JS) is the most popular lightweight, scripting, and interpreted programming language. JavaScript is well-known as a scripting language for web pages, mobile apps, web servers, and many other platforms. Bo...

15+ min read

Backpropagation in Neural Network

Back Propagation is also known as "Backward Propagation of Errors" is a method used to train neural network . Its goal is to reduce the difference between the modelâ€™s predicted output and the actual output by...

9 min read

React Tutorial

React is a JavaScript Library known for front-end development (or user interface). It is popular due to its component-based architecture, Single Page Applications (SPAs), and Virtual DOM for building web...

8 min read

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:

A-143, 7th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305)

Registered Address:

K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tkinter
Python Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

Inteview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects