GitHub (und git)

Einführung

Heiko Fanieng, 12.05.2024

GitHub

1. Einführung / Beschreibung

- GitHub ist ein webbasierter Dienst, der Entwicklern hilft, ihren Code zu speichern und zu verwalten sowie Änderungen an ihrem Code zu verfolgen und zu kontrollieren. Es basiert auf Git, einem Open-Source-Versionskontrollsystem, das 2005 von Linus Torvalds entwickelt wurde.
- Im Gegensatz zu anderen Dienstleistern zur Verwaltung quelloffener Software steht auf GitHub nicht das Projekt als Sammlung von Quellcode im Zentrum, sondern der Nutzer mit seinen Quelltext-Datenbanken, den sogenannten Repositories.
- GitHub ist besonders benutzerfreundlich, was es auch Anfängern ermöglicht, die Vorteile von Git zu nutzen. Seit Dezember 2018 gehört das Unternehmen GitHub, Inc., das seinen Sitz in San Francisco in den USA hat, zu Microsoft.

GitHub

1. Einführung / Vorteile

- Versionskontrolle und Zusammenarbeit: GitHub basiert auf Git, einem Open-Source-Versionskontrollsystem, das es Entwicklern ermöglicht, Änderungen an ihrem Code zu verfolgen und zu verwalten. Es erleichtert die Zusammenarbeit, indem es Teams ermöglicht, sicher an verschiedenen Teilen eines Projekts zu arbeiten, ohne das Gesamtprojekt zu beeinträchtigen.
- Benutzerfreundlichkeit: Die Benutzeroberfläche von GitHub ist benutzerfreundlich gestaltet, was es auch Anfängern ermöglicht, die Vorteile von Git zu nutzen.
- Community und Open-Source-Projekte: GitHub ist auch eine Plattform für Open-Source-Projekte. Entwickler können ihre Projekte hosten und mit anderen Entwicklern zusammenarbeiten. Es ermöglicht auch die Vernetzung und den Austausch zwischen Entwicklern.
- Unternehmenslösungen: GitHub bietet auch gehostete private Code-Repositories und andere geschäftsorientierte Pläne an, die es Unternehmen erleichtern, Teammitglieder und Sicherheit zu verwalten.

Zusammenfassend lässt sich sagen, dass GitHub ein unverzichtbares Werkzeug für Entwickler und Teams ist, die effizient zusammenarbeiten und ihren Code verwalten möchten.

GitHub

2. Grundlagen

- Repository (Verzeichnis) : Das grundlegende Element von GitHub, ein Ort um Code und Dateien einschließlich des Versionsverlaufes zu speichern. Einem Repository können mehrere Projektmitarbeiter zugewiesen werden und entweder öffentlich oder privat sein.
- Commit: Repräsentiert das Abbild (Snapshot) des Projektes zu einem bestimmten Zeitpunkt. Enthält Informationen über Änderungen seit dem letzten commit.
- Branch (Verzweigung): Unabhängige Entwicklungslinie. Man kann an neuen Funktionen oder Fehlerbehebungen arbeiten ohne die Arbeit der anderen Teammitglieder zu stören.

GitHub

3. Zusammenarbeit

- Pull Request: Vorgang bei dem Entwickler Änderungen aus dem Fork in das original Repository übertragen.
- Code Review: Gemeinsames Betrachten des Codes mit anderen Entwicklern. Code-Reviews ermöglichen es Entwicklern, zusammen an einem Projekt zu arbeiten, den Code der anderen zu überprüfen und die Gesamtqualität der Codebasis zu verbessern.
- Merge-Konflikt: Tritt in GitHub auf, wenn zwei oder mehr Branches konkurrierende Änderungen an denselben Dateien innerhalb eines Repositorys vorgenommen haben. Dies geschieht in der Regel, wenn Personen unterschiedliche Änderungen an derselben Zeile derselben Datei vornehmen, oder wenn eine Person eine Datei bearbeitet und eine andere Person dieselbe Datei löscht.

GitHub

4. Fortgeschrittene Funktionen

- GitHub Actions: Plattform für Continuous Integration und Continuous Delivery (CI/CD). Die Actions sind nicht nur auf DevOps beschränkt.
- GitHub Pages: Hosting-Dienst für statische Webseiten, welcher HTML, CSS und JavaScript Dateien aus dem Repository bezieht. Damit kann man direkt im Repository eine Webseite über das Projekt, das Team etc. bereitstellen.
- GitHub Packages: Dienst zum Hosten und Verwalten von kompletten Softwarepaketen, welcher vollständig in GitHub integriert ist.

GitHub

5. Die besten Einsatzmöglichkeiten / best practices

- Wie man effektiv mit GitHub arbeitet / how to work effectively with GitHub
- Gemeinsame Fehler und wie man sie vermeidet / common errors and how to avoid them

GitHub

6. Fazit / conclusion

GitHub bietet eine Vielzahl von Vorteilen, die es zu einer idealen Plattform für die Zusammenarbeit in Projektteams machen:

1. Effiziente Teamarbeit: GitHub ermöglicht es Entwicklern, effizient und intelligent zusammenzuarbeiten. Es fördert die Teamarbeit und kreative Problemlösungen durch seine intuitive Benutzeroberfläche und fortschrittliche Funktionen.

2. Codeverwaltung: Es schafft ein verteiltes System zur Codeverwaltung und erstellt Snapshots (Abbilder)des aktuellen Zustands eines Codes. Dies ermöglicht eine einfache Zusammenarbeit, da Entwickler eine neue Version der Software herunterladen, Änderungen vornehmen und die neueste Version hochladen können.

3. Problemverfolgung und Code-Reviews: Mit GitHub können Entwickler Code hosten, Issues verfolgen, Pull-Anfragen erstellen und Code-Reviews durchführen.

4. Sicherheit und Integrität des Codes: Versionskontrollsysteme wie Git helfen dabei, die Integrität und Sicherheit des sich ständig weiterentwickelnden Codes aufrechtzuerhalten, indem sie Änderungen schützen.

Zusammenfassend lässt sich sagen, dass **GitHub eine leistungsstarke Plattform für die Zusammenarbeit in Projektteams ist.** Es bietet eine Reihe von Tools und Funktionen, die die Effizienz und Produktivität von Entwicklerteams erheblich steigern können.

GitHub

Konzeptionelle Unterschiede zwischen Repository und Project

Repository: Ein Repository ist das grundlegendste Element von GitHub1 Es ist am einfachsten als Projektordner vorstellbar. Ein Repository enthält alle Projektdateien (einschließlich Dokumentation) und speichert den Revisionsverlauf jeder Datei Repositorys können mehrere Mitarbeiter haben und entweder öffentlich oder privat sein.

Project: GitHub hat eine Funktion namens Projekte eingeführt. Projekttafeln auf GitHub helfen, die Arbeit zu organisieren und zu priorisieren. Man kann Projekttafeln für bestimmte Feature-Arbeiten, umfassende Roadmaps oder sogar Release-Checklisten erstellen.

Zum Beispiel, wenn man mehrere unabhängige Prototyp-Anwendungen hat, könnte man für jede Anwendung ein separates Repository erstellen. Innerhalb dieser Repositorys kann man dann Projekte erstellen, um die Entwicklungsaufgaben zu verwalten und zu organisieren.

GitHub Packages

Vollintegrierter Dienst zum Hosten und Verwalten von Softwarepaketen

Schlüsselfunktionen:

1. Sicherheit: Sie können Pakete sicher innerhalb Ihrer Organisation oder mit der gesamten Welt veröffentlichen und konsumieren.
2. Integration: GitHub Packages kombiniert Ihren Quellcode und Ihre Pakete an einem Ort, um integriertes Berechtigungsmanagement und Abrechnung zu bieten..
3. Unterstützung für mehrere Paketmanager: Es bietet Unterstützung für mehrere beliebte Paketmanager wie npm, RubyGems, Maven, Gradle, NuGet, Docker und mehr..
4. Automatisierung: Sie können Ihre Pakete mit GitHub Actions automatisieren.
5. Einfache Veröffentlichung: Verwenden Sie branchen- und gemeinschaftsstandard Paketmanager mit nativen Tooling-Befehlen. Authentifizieren und veröffentlichen Sie dann direkt auf GitHub.
6. Vertrauenswürdige Quellen: Verstehen und installieren Sie Paketinhalte sicher. Erhalten Sie Pakete direkt von der Community auf GitHub und verwenden Sie nur das, was für Ihre Organisation genehmigt wurde.