Kostenloses eBook

LERNEN

github

Free unaffiliated eBook created from
Stack Overflow contributors.

#github

Inhaltsverzeichnis

Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: github

It is an unofficial and free github ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official github.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Kapitel 1: Erste Schritte mit Github

Bemerkungen

In diesem Abschnitt erhalten Sie einen Überblick darüber, was github ist und warum ein Entwickler es verwenden möchte.

Es sollte auch alle großen Themen in github erwähnen und auf die verwandten Themen verweisen. Da die Dokumentation für github neu ist, müssen Sie möglicherweise erste Versionen dieser verwandten Themen erstellen.

Examples

Installation oder Setup

GitHub ist eine riesige Sammlung von Git-Repositories. Mit anderen Worten, Sie können sich GitHub als eine Sammlung vieler Projekte vorstellen!

Ein Profil erstellen

- Besuchen Sie die Hauptseite von GitHub Here
- Wählen Sie einen Benutzernamen aus, geben Sie Ihre E-Mail-Adresse ein, wählen Sie ein sicheres Passwort und los geht's!

Nützliche Hilfsmittel

Für Git / GitHub-Anfänger kann es zunächst verwirrend sein, wie die Versionskontrolle funktioniert. Es gibt eine GUI-Version von GitHub, die Sie herunterladen und verwenden können. GitHub Desktop ist genau dieses Werkzeug.

Erstellen Sie Ihr erstes Repository

Sie können sich ein Repository als Projekt vorstellen. Sie können ein Repository online oder offline erstellen. Folgen Sie den unteren Schritten:

Online

1. Loggen Sie sich zuerst ein und gehen Sie zu Ihrem Profil.
2. Navigieren Sie zur Registerkarte "Repositorys" oben auf der Seite
3. Drücke den grünen "Neu" -Button und schon kann es losgehen!

Offline

1. Laden Sie git herunter und installieren Sie es (wählen Sie das Betriebssystem, das Sie ausführen)
2. Nach dem Herunterladen und der Installation können Sie entweder das Befehlszeilentool verwenden oder einen GUI-Client herunterladen.
3. Erstellen Sie nach der Installation ein Konto bei github
4. Klicken Sie oben rechts auf das + und wählen Sie entweder ein neues Repository erstellen oder ein vorhandenes auf importieren.
5. Wenn Sie einen neuen Namen auswählen, geben Sie den Namen des Repositorys ein und wählen Sie entweder "public" oder "private".
6. Klicken Sie auf: Repository erstellen

Hinweis: Private Repositorys stehen nicht kostenlos zur Verfügung.

README-Datei

Wenn Ihr Projekt nicht über README.md verfügt, kann GitHub README.rdoc analysieren, um Details anzuzeigen. Wenn beides vorhanden ist, wird README.md verwendet, wobei rdoc ignoriert wird.

Eine README-Datei kann enthalten

Projekttitel

Beschreiben Sie kurz Ihr Projekt. Sie können auch die Website, Projektabzeichen, Community- und Kontaktinformationen (z. B. E-Mail, soziale Website) des Projekts angeben.

Herunterladen

Link für ausführbare Dateien (ausführbare oder minimierte Dateien oder Installationsdateien). Es kann auch Links zu früheren Versionen geben.

Installation

Wie kann Ihre Arbeit genutzt werden? Es kann die Voraussetzungen, Einstellungen, Bibliotheken von Drittanbietern, Verwendung, Vorsichtsmaßnahmen usw. enthalten.

Demonstration

Es kann Codebeispiele, GIF-Dateien, Videolinks oder sogar Screenshots enthalten.

Autoren

Autorennamen, Kontaktinformationen usw.

Danksagungen

Liste der Personen oder der Gemeinschaft, die während des gesamten Projekts geholfen und inspiriert wurden

Mitwirken

Anweisungen zum Beitrag (dh Hinzufügen von Funktionen, Melden von Fehlern, Senden von Patches) zum Projekt. Kann auch Dokumentationslink enthalten.

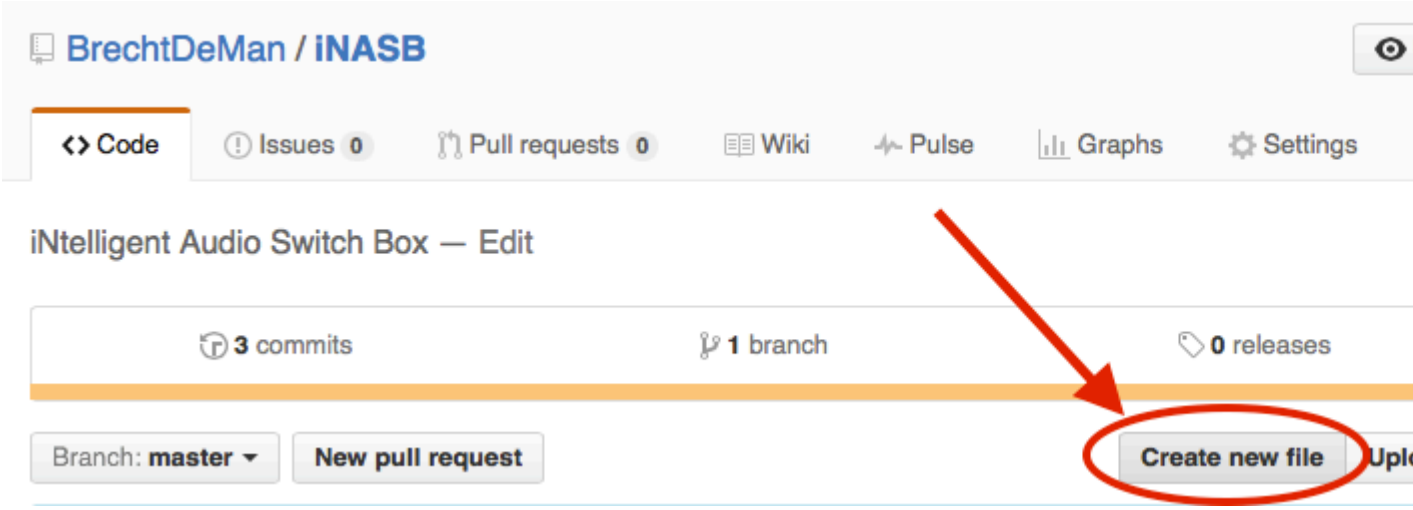
Lizenz

Geben Sie ein kurzes Intro über Ihre Lizenz. Sie können auch einen Link zur Lizenzseite angeben.

Lizenzdatei

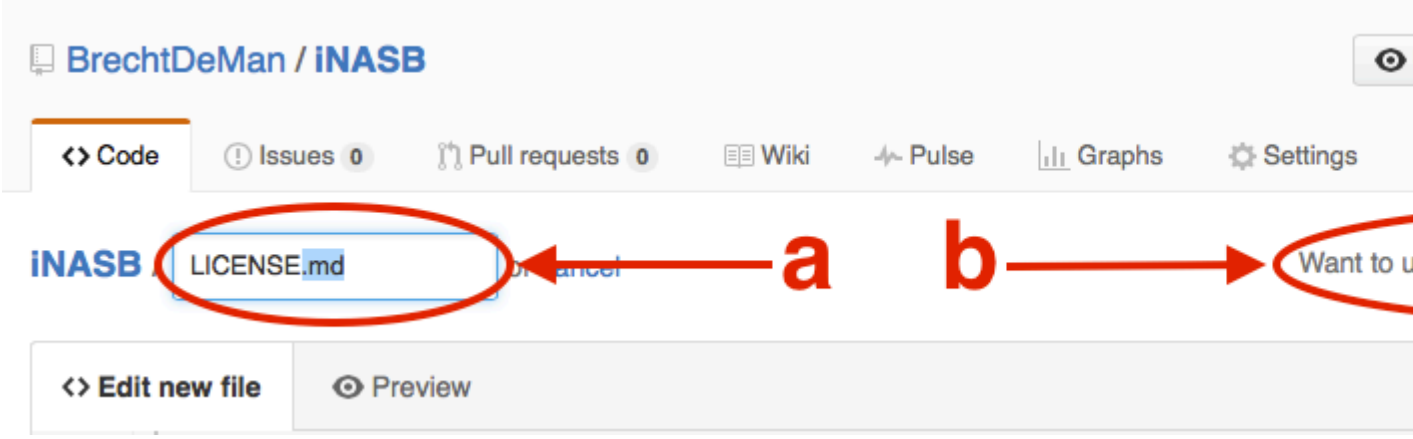
GitHub hilft Ihnen, schnell eine Lizenz zu Ihrem Repository hinzuzufügen, als Alternative zum Hinzufügen einer eigenen Text- / Markdown-Datei.

1. Klicken Sie in Ihrem Repository auf "Neue Datei erstellen".

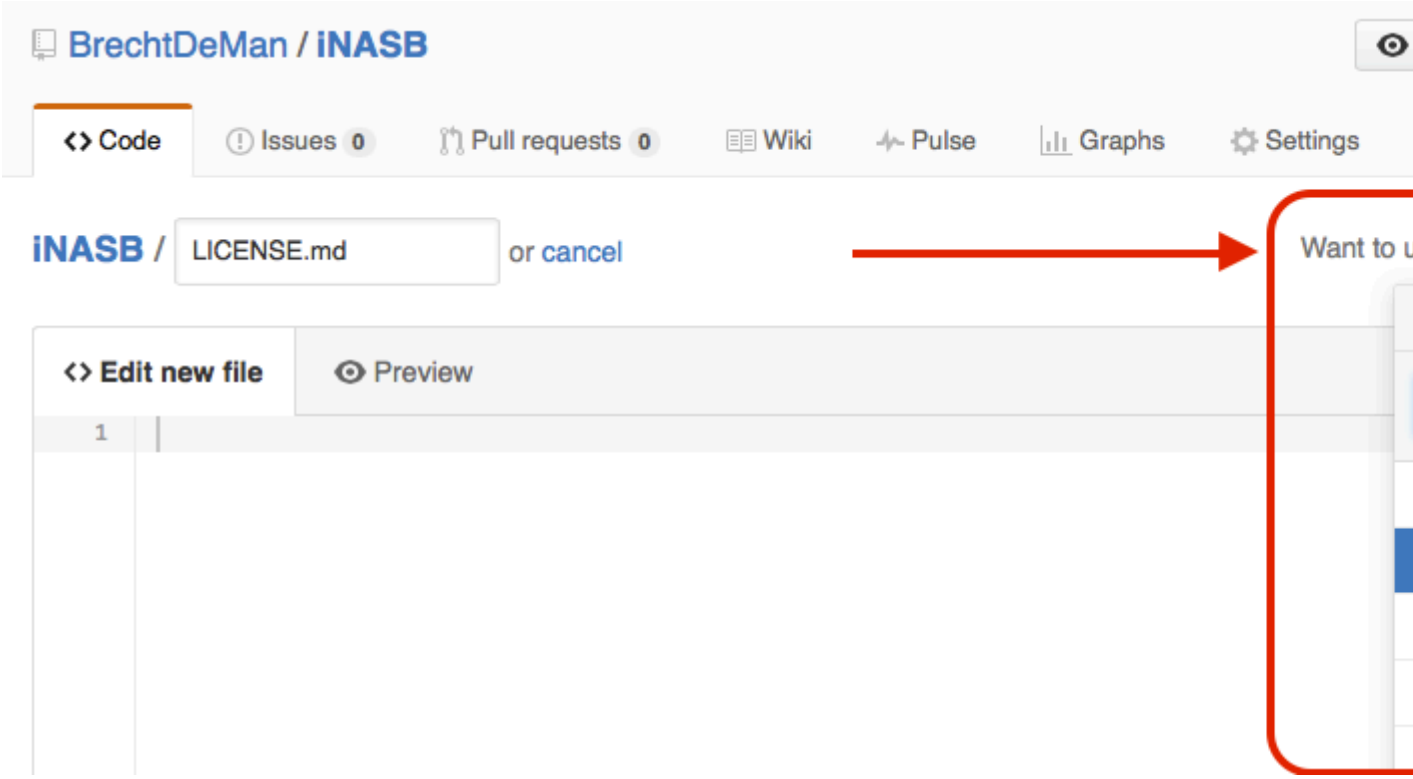


2. Auf der nächsten Seite:

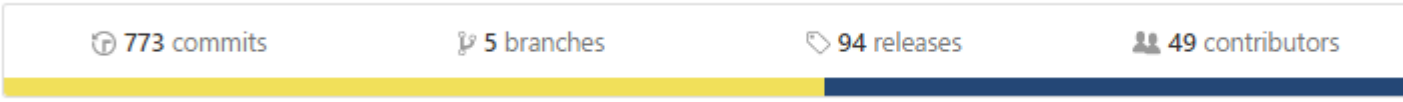
    1. `LICENSE.md` `LICENSE.txt` `LICENSE.md` oder `LICENSE.txt` als Dateinamen der neuen Datei ein.
    2. *Möchten Sie eine neue Vorlage verwenden?* ein Dialog erscheint.



3. Wählen Sie Ihre bevorzugte Lizenz.



4. Die Lizenz, die Sie in den Repository-Details sehen konnten:



From Q & A - Wie fügt man einem vorhandenen Github-Projekt eine Lizenz hinzu?

GitHub Flavored Markdown

GitHub erweitert die Markdown- Syntax um neue nützliche Funktionen.

Header

```
# Header1
## Header2
### Header3
#### Header4
##### Header5
###### Header6
H1
===
H2
---
```

Header1

Header2

Header3

Header4

Header5

Header6

H1

H2

Betonung

```
*Italic1* _Italic2_
**Bold1** __Bold2__
***Bold_Italic***
~~Strikethrough~~
```

Italic1 Italic2
Bold1 Bold2
Bold_Italic
~~Strikethrough~~

Horizontale Linie

```
---
***
___
```

Liste

```
unordered list:

* item-1
  * sub-item-1
  * sub-item-2
- item-2
  - sub-item-3
  - sub-item-4
+ item-3
  + sub-item-5
  + sub-item-6


ordered list:

1. item-1
  1. sub-item-1
  2. sub-item-2
2. item-2
  1. sub-item-3
  2. sub-item-4
3. item-3
```

unordered list:

- item-1
  - sub-item-1
  - sub-item-2
- item-2
  - sub-item-3
  - sub-item-4
- item-3
  - sub-item-5
  - sub-item-6

ordered list:

1. item-1
   i. sub-item-1
   ii. sub-item-2
2. item-2
   i. sub-item-3
   ii. sub-item-4
3. item-3

Tabelle

```
Table Header-1 | Table Header-2 | Table Header-3
:--- | :---: | ---:
Table Data-1 | Table Data-2 | Table Data-3
TD-4 | Td-5 | TD-6
Table Data-7 | Table Data-8 | Table Data-9
```

Table Header-1	Table Header-2	Table Header-3
Table Data-1	Table Data-2	Table Data-3
TD-4	Td-5	TD-6
Table Data-7	Table Data-8	Table Data-9

Code

````
inline code- `int i=0`

block code-
``` C
for(int i=0; i<10; i++){
    printf("Hallow World! \n");
}
```
````

inline code- `int i=0`

block code-

```c
for(int i=0; i<10; i++){
    printf("Hallow World! \n");
}
```

Zitat

```
> Stay hungry; stay foolish.
>> Quality is better than quantity.
>>> Life is not fair; get used to it.
```

> Stay hungry; stay foolish.
>
> > Quality is better than quantity.
> >
> > > Life is not fair; get used to it.

Verknüpfung

```
https://github.com
[GitHub](https://github.com)
[GitHub](https://github.com "github website")
[GitHub][1]

[1]: https://github.com
```

https://github.com
GitHub
GitHub
GitHub

Bild

```
![GitHub Logo](https://assets-cdn.github.com/images/icons/emoji/octocat.png "GitHub")
```

Aufgabenlisten

```
- [x] completed item
- [ ] incomplete item
```

- [x] completed item
- [ ] incomplete item

Emoji

```
:octocat: :+1: :book: :ghost: :bulb: :imp:
```

🐙👍📖👻💡😈

Für alle GitHub-Emojies besuchen Sie das Emoji Cheat Sheet .

SHA-Referenzen

Jeder Verweis auf einen SHA1-Hash eines Commits wird in einen Link zum Commit selbst auf GitHub umgewandelt:

```
e7909ea4fbbb162db3f7f543d43c30684a3fb745f
```

Write | Preview

e7909ea

Pull-Request und Issue-Referenzen

Jeder Verweis auf eine Pull-Anfrage oder ein Problem wird automatisch mit dieser Pull-Anfrage

oder Ausgabe verknüpft.

Sie können dies tun, indem Sie ein `#` vor die Issue / Pull Request-Nummer setzen.

Erste Schritte mit Github online lesen: https://riptutorial.com/de/github/topic/1214/erste-schritte-mit-github

Kapitel 2: Aktualisieren Sie ein gegabeltes Repository

Bemerkungen

- GitHub Help: Konfigurieren einer Fernbedienung für eine Abzweigung
- GitHub Help: Synchronisieren einer Verzweigung
- beliebte ans in StackOverFlow

Examples

Konfigurieren Sie eine Fernbedienung für Ihre Gabel und synchronisieren Sie dann Ihre Gabel (Master-Zweig).

1. Konfigurieren Sie eine Fernbedienung für meine Gabel

```
$ cd my_local_repo

$ git remote add upstream https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY.git
    # Specify a new remote upstream repository that will be synced with the fork

$ git remote -v
    # Verify the new upstream repository specified for my fork
```

2. Synchronisiere meine Gabel lokal

```
$ cd my_local_repo

$ git fetch upstream
    # Fetch the branches and their respective commits from the upstream repository
    # Commits to master will be stored in a local branch, upstream/master

$ git checkout master

$ git merge upstream/master
    # Merge the changes from upstream/master into your local master branch
    # brings your fork's master branch into sync with the upstream repo
```

3. Synchronisiere meine Gabel mit Github

```
$ git push origin master
```

Aktualisieren Sie ein gegabeltes Repository online lesen:
https://riptutorial.com/de/github/topic/3758/aktualisieren-sie-ein-gegabeltes-repository

Kapitel 3: Anzeigen der GitHub-Timeline / -Feeds auf Ihrer Website

Examples

Anzeigen der GitHub-Timeline / -Feeds auf Ihrer Website

In diesem Dokument wird erläutert, wie Sie Ihre GitHub-Feeds / Timeline auf Ihrer Website anzeigen.

Beispiel: Ein Live-Beispiel ist verfügbar unter:

> https://newtonjoshua.com

GitHub-Timeline:

GitHub stellt die öffentliche Timeline für jeden Benutzer im Atom-Format bereit.

Sie können Ihre Zeitleiste anzeigen unter:

> https://github.com/ {{GitHub_username}}. atom

siehe: https://developer.github.com/v3/activity/feeds

Google Feed-API:

Mit der Feed-API können Sie jeden öffentlichen Atom-, RSS- oder Medien-RSS-Feed nur mit JavaScript herunterladen, sodass Sie Feeds mit nur wenigen JavaScript-Zeilen mit Ihrem Inhalt und anderen APIs kombinieren können. Dadurch können Sie Feeds schnell in Ihre Website integrieren.

Siehe: https://developers.google.com/feed/v1/devguide

Laden der JavaScript-API: Um die Feed-API zu verwenden, fügen Sie das folgende Skript in die Kopfzeile Ihrer Webseite ein.

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

Laden Sie anschließend die Feed-API mit google.load (Modul, Version, Paket).

```
<script type="text/javascript">
  google.load("feeds", "1");
</script>
```

Angeben der Feed-URL: Sie können google.feeds.Feed () wie folgt aufrufen:

```
var feed = new google.feeds.Feed("https://github.com/{{GitHub_UserName}}.atom");
```

Laden eines Feeds: .load (Rückruf) lädt den im Konstruktor angegebenen Feed von den Servern von Google herunter und ruft den angegebenen Rückruf auf, wenn der Download abgeschlossen ist.

```
<script type="text/javascript">

    function initialize() {
      feed.load(function(result) {
        if (!result.error) {
          var container = document.getElementById("feed");
          result.feed.entries.forEach(function (feed) {
            var feedTitle= feed.title;
            var feedLink = feed.link;
            var feedDate = formatDate(feed.publishedDate);
            var feedContent = formatContent(feed.content);

           // display the feed in your website
          });
        }
      });
    }
    google.setOnLoadCallback(initialize);

    </script>
```

Aufruf des onLoad-Handlers: setOnLoadCallback (callback) ist eine statische Funktion, die die angegebene Handlerfunktion registriert, sobald die Seite mit diesem Aufruf geladen wird. Callback ist eine erforderliche Funktion, die aufgerufen wird, wenn das enthaltende Dokument geladen wird und die API verwendet werden kann

```
<script type="text/javascript">
    google.setOnLoadCallback(initialize);
 </script>
```

Festlegen der Anzahl der Feedeinträge **:** .setNumEntries (num) setzt die Anzahl der Feedeinträge, die von diesem Feed geladen werden, auf num. Standardmäßig lädt die Feed-Klasse vier Einträge.

```
var feed = new google.feeds.Feed("https://github.com/{{GitHub_UserName}}.atom");
feed.setNumEntries(500);
```

Jetzt können Sie Ihre GitHub-Feeds / Timeline auf Ihrer Website formatieren und anzeigen.

Anzeigen der GitHub-Timeline / -Feeds auf Ihrer Website online lesen:
https://riptutorial.com/de/github/topic/7479/anzeigen-der-github-timeline----feeds-auf-ihrer-website

Kapitel 4: Gist verwenden

Einführung

Gists sind eine großartige Möglichkeit, Ihre Arbeit zu teilen. Sie können einzelne Dateien, Teile von Dateien oder vollständige Anwendungen freigeben. Sie können auf Gists unter https://gist.github.com zugreifen.

Bei jedem gist handelt es sich um ein Git-Repository, das heißt, es kann gabelig gemacht und geklont werden. Der Haupteditor wird von CodeMirror unterstützt.

Es gibt zwei Arten von Gists: öffentliche Gänge und geheime Gists.

Wenn Sie beim Erstellen Ihres Gists nicht bei GitHub angemeldet sind, wird es außerdem ein anonymer Gist sein.

Bemerkungen

Gists sind eine großartige Möglichkeit, Ihre Arbeit zu teilen. Sie können einzelne Dateien, Teile von Dateien oder vollständige Anwendungen freigeben.

Es gibt zwei Arten von Gists: öffentliche Gänge und geheime Gists. Wenn Sie beim Erstellen Ihres Gists nicht bei GitHub angemeldet sind, wird es außerdem ein anonymer Gist sein.

Öffentliche Gists

Öffentliche Gists werden in Discover angezeigt, in denen Benutzer neue Gists durchsuchen können, wenn sie erstellt werden. Sie sind auch durchsuchbar. Sie können sie also verwenden, wenn andere Personen Ihre Arbeit finden und anzeigen möchten.

Geheime Gists

Geheime Gists werden nicht in Discover angezeigt und können nicht durchsucht werden. Verwenden Sie sie, um eine Idee zu notieren, die Ihnen im Traum begegnet ist, erstellen Sie eine To-Do-Liste oder bereiten Sie einen Code oder eine Prosa vor, die nicht für die Welt bereit ist.

Sie können so viele geheime Gists erstellen, wie Sie möchten.

Anonyme Gists

Wenn Sie eine Giste erstellen, ohne sich bei GitHub anzumelden, wird diese anonyme Giste. Anonyme Gists können öffentlich oder geheim sein. Wenden Sie sich an GitHub-Support oder Ihren Site-Administrator, um eine anonyme Liste auf GitHub.com oder GitHub Enterprise zu löschen. Bitte geben Sie die URL des Gists an, den Sie löschen möchten.

Examples

Public Gist

Ein öffentlicher Kern kann *fast* alles sein.

Ein einfaches Beispiel für eine Javascript-Funktion:

```
function randomInt(min, max) {
return Math.floor((max - min + 1) * Math.random()) + min;
}
```

Secret Gist

Ein geheimes Element sollte für alles verwendet werden, das Sie nicht öffentlich auf GitHub veröffentlichen möchten. Geheime Gists können verwendet werden, wenn Sie nicht möchten, dass private Schlüssel für die Öffentlichkeit oder allgemein für privaten Code zugänglich sind.

Ein einfaches Beispiel für JSON-Code, der für einen geheimen Kern besser geeignet wäre:

```
{
"id": AKIAIOSFODNN7EXAMPLE,
"secret": wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
}
```

Gist verwenden online lesen: https://riptutorial.com/de/github/topic/7978/gist-verwenden

Kapitel 5: GitHub Buttons verwenden

Einführung

Was sind GitHub-Buttons? GitHub-Schaltflächen sind Schaltflächen, die Sie Ihrer Website hinzufügen können, um Benutzer zu einem beliebigen Repository umzuleiten, das Sie mögen!

Bemerkungen

Credits:

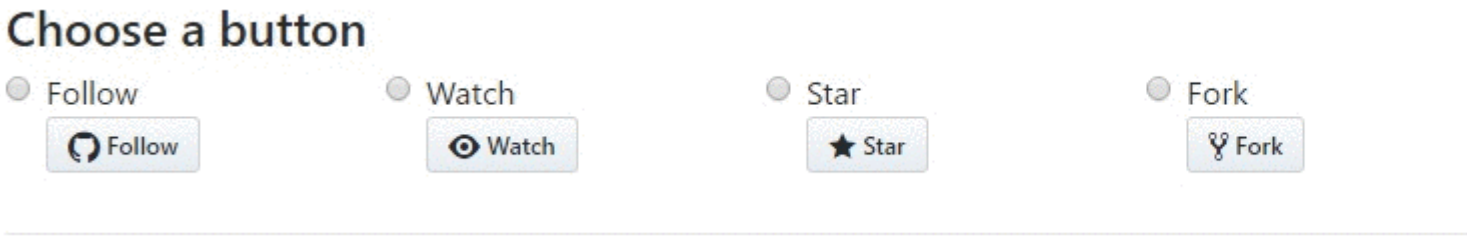
- Mit Recordit aufgenommene GIF- Bilder
- Mit Snipping Tool aufgenommene statische Bilder
- Der in vollständigen Tutorials verwendete Code-Editor war codepen.io

Examples

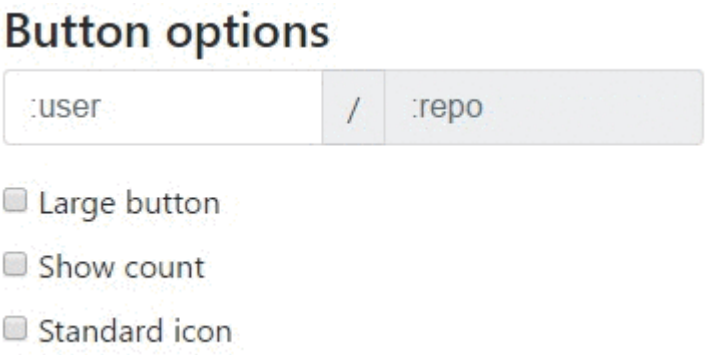
Folgen Schaltfläche

Eine Follow-Schaltfläche ist eine Schaltfläche, die auf eine GitHub-Benutzerseite verweist und den Benutzer auffordert, dem Benutzer zu folgen. So erstellen Sie eine:

1. Gehen Sie auf github: Buttons
2. Klicken Sie auf "Folgen".



3. Tragen Sie Ihren GitHub-Benutzernamen in das Feld ": user" ein.



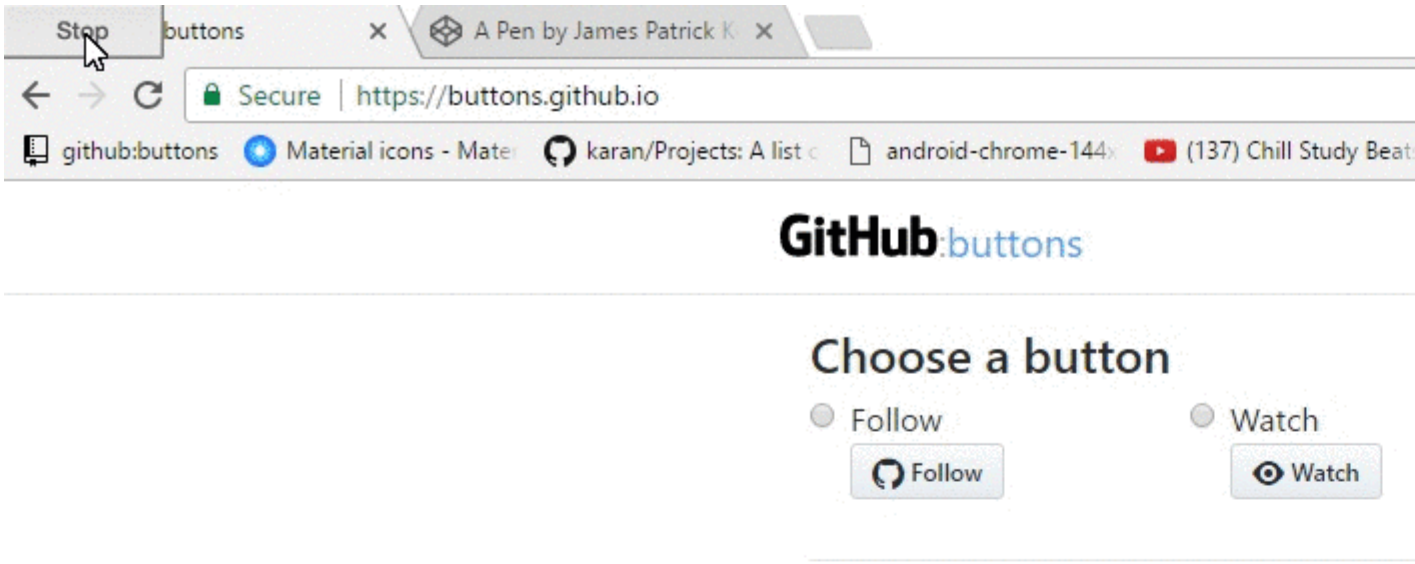
4. Passen Sie die Schaltfläche mit den Feldern "Große Schaltfläche", "Anzahl anzeigen" und

"Standardsymbol" an:

5. Fügen Sie diesen Code in den `<head>` oder vor dem Ende des `<body>` Ihres Codes ein:

```
<a class="github-button" href="https://github.com/hubot" aria-label="Follow @hubot on
GitHub">Follow @hubot</a>
```

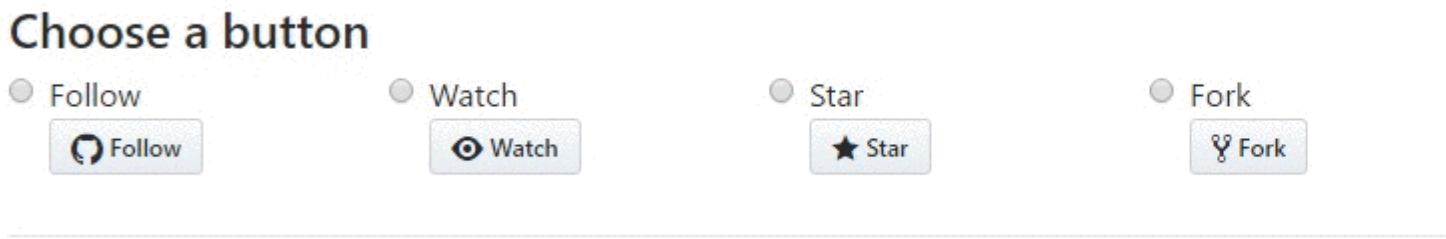
6. Platzieren Sie den benutzerdefinierten Button-Rendering-Code in Ihrem Code.


Stop
buttons
A Pen by James Patrick K
Secure
https://buttons.github.io
github:buttons
Material icons - Mate
karan/Projects: A list
android-chrome-144
(137) Chill Study Beat
GitHub:buttons
Choose a button
Follow
Follow
Watch
Watch

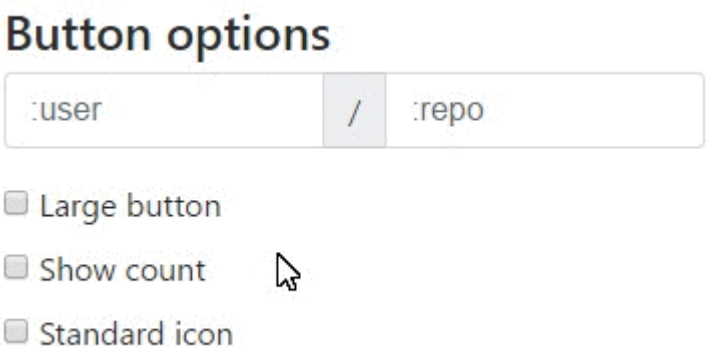
- **Starten Sie** ein Repository
- **Fork** ein Repository
- **Laden Sie** ein Repository **herunter**
- Listen Sie ein **Problem** mit einem Repository auf

So erstellen Sie einige:

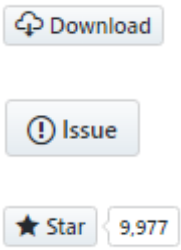
1. Gehen Sie auf github: Buttons
2. Klicken Sie auf den Schaltflächentyp, den Sie erstellen möchten (Watch, Star, Fork, Download oder Issue).



3. Geben Sie Ihren GitHub-Benutzernamen in das Feld ": user" und Ihr Repository in das Feld ": repo" ein.



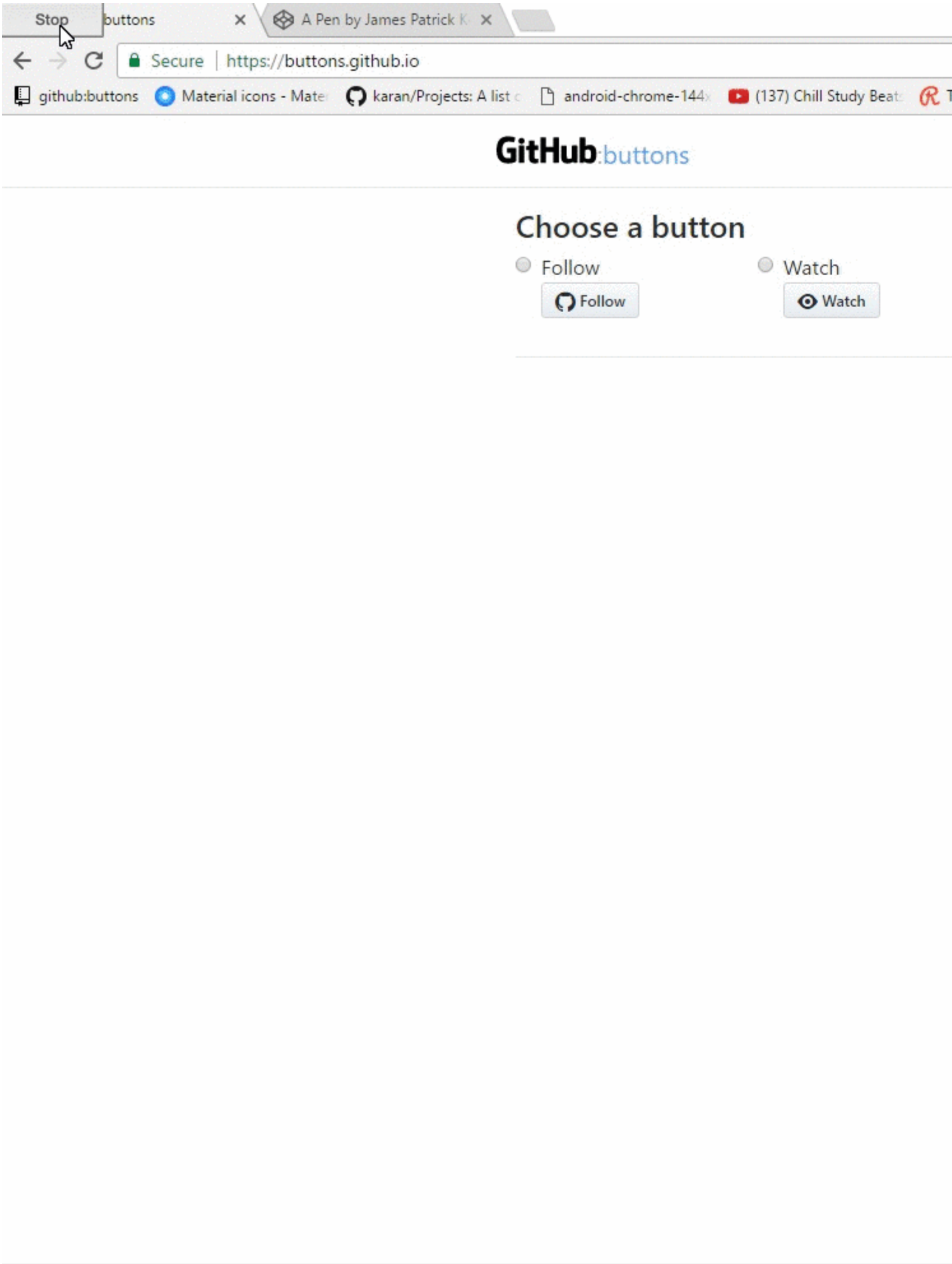
4. Passen Sie die Schaltfläche mit den Feldern "Große Schaltfläche", "Anzahl anzeigen" und "Standardsymbol" an:



5. Fügen Sie diesen Code in den `<head>` oder vor dem Ende des `<body>` Ihres Codes ein:

```
<a class="github-button" href="https://github.com/hubot" aria-label="Follow @hubot on
GitHub">Follow @hubot</a>
```

6. Platzieren Sie den benutzerdefinierten Button-Rendering-Code in Ihrem Code.


Stop
buttons
A Pen by James Patrick K
Secure
https://buttons.github.io
github:buttons
Material icons - Mate
karan/Projects: A list
android-chrome-144x
(137) Chill Study Beat
GitHub:buttons
Choose a button
Follow
Watch
Follow
Watch
1J2GxbWa2o (1).gif
1J2GxbWa2o.gif
ss6fW8XoSL (1).gif

https://riptutorial.com/de/github/topic/10585/github-buttons-verwenden

Kapitel 6: GitHub Desktop

Einführung

Wie installiere und arbeite ich mit GitHub Desktop?

GitHub Desktop ist - wie der Name schon sagt - eine Desktopumgebung für Windows und MacOS, die die Hauptfunktionen von Git umfasst, wie Klonen, Pushing, Pulling (Synchronisieren in GitHub Desktop), ...

Der Hauptzweck des Desktop-Clients besteht darin, eine einfachere Arbeitsweise mit git (und GitHub) zu ermöglichen. Im Hintergrund werden dieselben Befehle verwendet, die die meisten Benutzer von der Befehlszeile aus verwenden würden.

Examples

Installation und Einrichtung

Die Installation ist recht einfach, da hier separate Installationsprogramme für MacOS- und Windows-Maschinen verfügbar sind . Derzeit stehen zwei Versionen zum Download zur Verfügung: eine Beta und eine Stable.

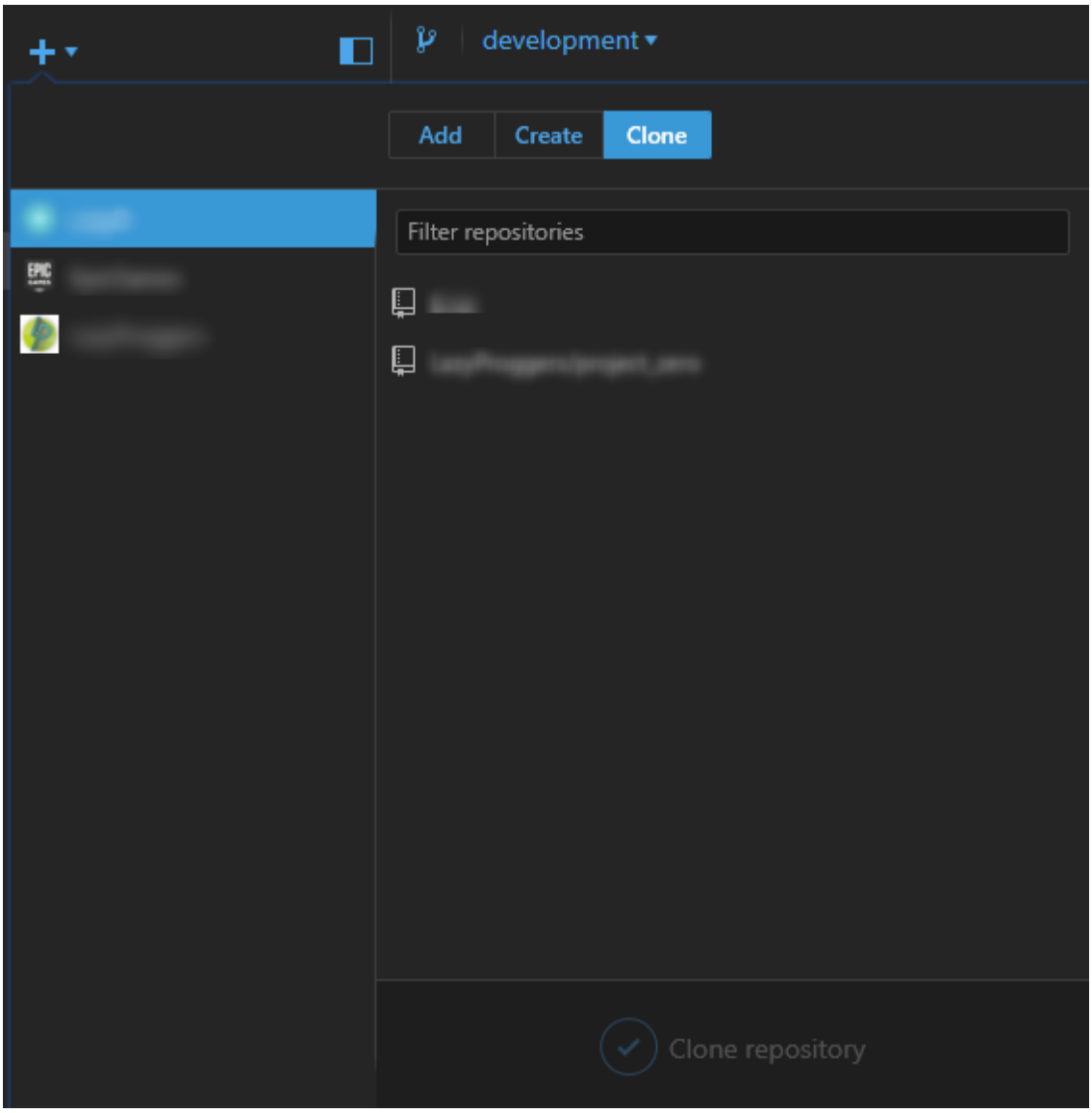
Das Setup wird gestartet, nachdem Sie das Programm heruntergeladen haben, und Sie müssen sich mit Ihren GitHub-Anmeldeinformationen anmelden. Dies ist wirklich der einzige Schritt, denn danach können Sie ein Repository erstellen oder eines klonen.

> Hinweis: Während der Installation werden nicht nur GitHub Desktop, sondern auch Git installiert. Sie müssen es also nicht separat installieren.

Repository klonen

Wie bei GitHub Desktop ist der Großteil der Arbeit ziemlich einfach: Sie wählen "Repository klonen" (in der stabilen Version oben links) und es gibt einige Repositorys (eigene und Repos von jedem Unternehmen, in dem Sie sich befinden) empfohlen. Alternativ können Sie einen Link in ein anderes Repository einfügen, das Sie möglicherweise klonen möchten.

> Hinweis: In der neueren Version (Beta) gibt es keine (nicht Jet?) Empfehlungen.



Verzweigung

Sie können oben links einen Zweig auswählen. Wenn Sie den rechten Zweig ausgewählt haben, müssen Sie den Sync-Button (oben rechts) drücken, der jetzt wie `git checkout BRANCHNAME` .

> In der älteren Version können Sie zwei verschiedene Zweige gleichzeitig anzeigen und die Pushs vergleichen. Außerdem können Sie eine Zeitleiste Ihres Projekts anzeigen (siehe unten).

Einen neuen Zweig erstellen

Sie können einen neuen Zweig erstellen, indem Sie auf das Zweigsymbol (alter Client) klicken oder unter `File --> New Branch` .

> Beachten Sie, dass Sie auswählen können, welchen Zweig der neue Zweig als Basis

verwenden soll, indem Sie auf den Zweignamen klicken.

Push und Pull (oder: die Sync-Taste)

Ziehen (Sync)

Wie in der Befehlszeile müssen Sie gelegentlich den aktuellen Status des Repositorys abrufen. In GitHub Desktop wird dieser Vorgang mit dem `sync` Button in der oberen rechten Ecke aufgerufen.

drücken

Wenn Sie lokale Änderungen vorgenommen haben und diese übernehmen möchten, schreiben Sie etwas in das Textfeld für die Zusammenfassung. Dann drücken Sie `Commit to YOURCURRENTBRANCH` . Jetzt müssen Sie die Sync-Taste drücken, und Sie werden gedrückt.

> Hinweis: Sie können Emoticons, Erwähnungen und Verweise für andere Commits oder Ausgaben direkt aus dem Textfeld verwenden.

So kann die **Sync-** Taste zum `Push` , `Pull` oder `Checkout` .

GitHub Desktop online lesen: https://riptutorial.com/de/github/topic/10023/github-desktop

Kapitel 7: GitHub sichern

Examples

Klonen aller Repositorys für einen Benutzernamen

Führen Sie den folgenden Befehl aus, wobei Sie den Benutzernamen durch den Benutzernamen ersetzen, um alle GitHub-Repositorys für diesen Benutzer in das aktuelle Verzeichnis zu kopieren.

```
curl "https://api.github.com/users/username/repos?page=1&per_page=100" | grep -e 'git_url*' |
cut -d \" -f 4 | xargs -L1 git clone
```

Dadurch werden nur die ersten 100 Repositorys geklont.

GitHub sichern online lesen: https://riptutorial.com/de/github/topic/3760/github-sichern

Kapitel 8: GitHub-Seiten

Examples

Verwenden des automatischen Seitengenerators für ein Repository

1. Gehen Sie zur GitHub-Website
2. Öffnen Sie Ihr Repository
3. Klicken Sie auf Einstellungen
4. Klicken Sie unter GitHub Pages auf "Launch Automatic Page Generator".
5. Folge den Anweisungen

Verwenden von Git zum Erstellen von Seiten von Grund auf

1. Erstellen Sie ein neues Repository oder klonen Sie ein vorhandenes Repository.
2. Erstellen Sie einen neuen Zweig namens `gh-pages` ohne Historie

```
$ git checkout --orphan gh-pages

# ensure you are in the correct directory then,
# remove all files from the old working tree
$ git rm -rf
```

3. Fügen Sie dem Stamm des Repositorys eine index.html-Datei hinzu.

```
$ echo "Hello World" > index.html
$ git add index.html
$ git commit -a -m "First pages commit"
```

4. Drücken Sie zu Github.

```
$ git push origin gh-pages
```

Sie können jetzt Ihre neue Github Pages-Site unter `http(s)://<username>.github.io/<projectname>`

Erstellen einer benutzerdefinierten URL für Ihre GitHub-Seite

Sie benötigen einen Domainnamen von einem Registrar .

Erstellen Sie im `gh-pages` Zweig Ihres Projekt-Repositorys oder im Hauptzweig Ihres `username.github.io` Repositorys eine CNAME-Datei mit dem Inhalt `www.yourdomain.com` - der *kanonischen* Domäne .

Richten Sie auf der Domain-Konfigurationsseite Ihres Registrars Ihre Domain auf Ihre GitHub-Website. Richten Sie zwei CNAME-Einträge ein (einen für die Wurzelspitze (@) und einen für das WWW). Beide sollten auf `username.github.io` oder `username.github.io/repository` verweisen. Wenn Ihr DNS-Anbieter keine ALIAS-Einträge auf der Stammspitze (@) unterstützt, erstellen Sie einfach

A-Einträge, die auf 192.30.252.153 und 192.30.252.154 zeigen.

Ressourcen

GitHub-Anweisungen für eine benutzerdefinierte Domäne

Fragen und Antworten zum Stapelüberlauf: "Benutzerdefinierte Domäne für GitHub-Projektseiten"

Audrey Watters - Mit GitHub ein Webprojekt betreiben: Wie und warum

Alex Cican - Wie ich meine Websites auf Dropbox und GitHub verschoben habe

Baumhaus - Verwenden von GitHub-Seiten zum Hosten Ihrer Website

GitHub-Seiten online lesen: https://riptutorial.com/de/github/topic/3759/github-seiten

Kapitel 9: Laden Sie eine einzelne Datei aus dem GitHub-Repository herunter

Examples

aus einem öffentlichen Repository über die Befehlszeile und die Umbenennungsdatei

In diesem Beispiel wird die Datei Node.gitignore aus dem GitHub-Gititore-Repository entnommen, in Ihr aktuelles Arbeitsverzeichnis heruntergeladen und in .gitignore umbenannt. Dies sind alles typische Aktionen für jemanden, der ein neues node.js-Projekt startet.

```
$ curl http://github.com/github/gitignore/raw/master/Node.gitignore -o .gitignore
```

Finden Sie die URL der Datei, die Sie herunterladen möchten

1. Navigieren Sie zur gewünschten Datei in einem Repository
2. Klicken Sie auf die Schaltfläche "Roh"
3. Kopieren Sie die URL aus der Adressleiste

Siehe das folgende Beispiel aus GitHubs Gititore-Repository:
http://github.com/github/gitignore/raw/master/Node.gitignore

Sie können schnell eine URL erkennen, die zum Herunterladen einer einzelnen Datei oder zum Herunterladen der HTML-Seite funktioniert. Suchen Sie nach dem Unterverzeichnis / raw / vor dem Zweignamen.

Laden Sie eine einzelne Datei aus dem GitHub-Repository herunter online lesen:
https://riptutorial.com/de/github/topic/10898/laden-sie-eine-einzelne-datei-aus-dem-github-repository-herunter

Kapitel 10: Mit Gitflow arbeiten

Syntax

- git flow <Unterbefehl>
- git flow init
- Git Flow [Feature | Release | Hotfix] [Start | Ende]

Parameter

Unterbefehl	Einzelheiten
drin	Initialisieren Sie ein neues Git-Repo mit Unterstützung für das Verzweigungsmodell.
Merkmal	Verwalten Sie Ihre Funktionszweige.
Veröffentlichung	Verwalten Sie Ihre Release-Zweige.
Hotfix	Verwalten Sie Ihre Hotfix-Zweige.

Bemerkungen

- Gitflow-Konzept vom Autor
- Zweigmodell Bild

Examples

Betrieb an 5 gemeinsamen Niederlassungen vor Ort

Einer der häufigsten Anwendungsfälle von Gitflow

1. Repo *initialisieren* und Zweige definieren

```
$ git flow init
    # if you use default setup, you'll define six types of branches:
    #
    # main branches (lives forever)
    #
    #   1. master:  for production releases
    #   2. develop: for "next release" development
    #
    # supporting branches
    #
    #   3. feature: for a product feature
    #   4. release: for preparation of a new production release
```

```
    #   5. hotfix:  for resolving critical bug of production version
    #   6. support
    #
    # also, two main branches are created: master, develop
```

2. *Starten* und *beenden Sie* eine Funktion

```
$ git flow feature start my_feature
    # create branch 'feature/my_feature' based on the 'develop'

    # made development and commits...

$ git flow feature finish my_feature
    # merge 'feature/my_feature' back to the 'develop'
    # delete 'feature/my_feature'
```

3. *Starten* und *beenden Sie* eine Freigabe

```
$ git flow release start my_release
    # create branch 'release/my_release' based on the 'develop'

    # made bug fixes...

$ git flow release finish my_release
    # merge branch 'release/my_release' to the 'master' and add tag
    # merge branch 'release/my_release' back to the 'develop'
    # delete 'release/my_release'
```

4. *Starten* und *Beenden* Hotfix

```
$ git flow hotfix start my_hotfix
    # create branch 'hotfix/my_hotfix' based on the 'master'

    # made some hotfixes...

$ git flow hotfix finish my_hotfix
    # merge branch 'hotfix/my_hotfix' back to the 'master' and add tag
    # merge branch 'hotfix/my_hotfix' to the 'develop'
    # delete 'hotfix/my_hotfix'
```

Mit Gitflow arbeiten online lesen: https://riptutorial.com/de/github/topic/6231/mit-gitflow-arbeiten

Kapitel 11: Probleme

Examples

Problem erstellen

1. Gehen Sie auf die GitHub-Seite für das Projekt, in dem Sie ein Problem erstellen möchten.
2. Klicken Sie auf ***Issues*** .
3. Klicken Sie oben rechts auf ***Neue Ausgabe*** .
4. Geben Sie den Titel der Ausgabe ein.
5. Geben Sie den Hauptteil des Problems ein (einschließlich Protokolle, Code-Snippets usw.).
6. *Optional:* Um das Problem vor dem Senden anzuzeigen, klicken Sie auf Vorschau.
7. Klicken Sie auf ***Neue Ausgabe senden*** .

Probleme online lesen: https://riptutorial.com/de/github/topic/3757/probleme

Kapitel 12: Pull-Anfragen

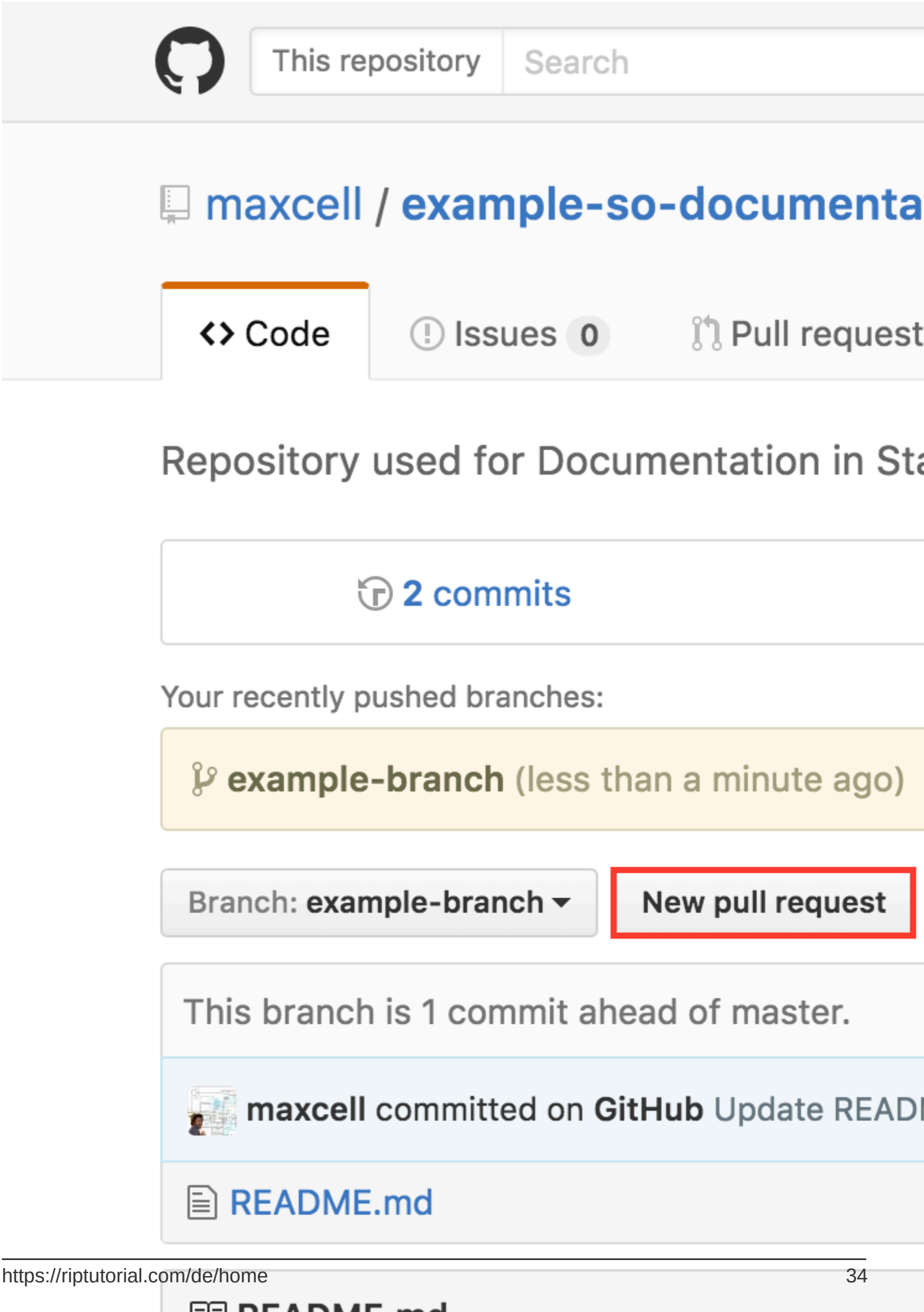
Examples

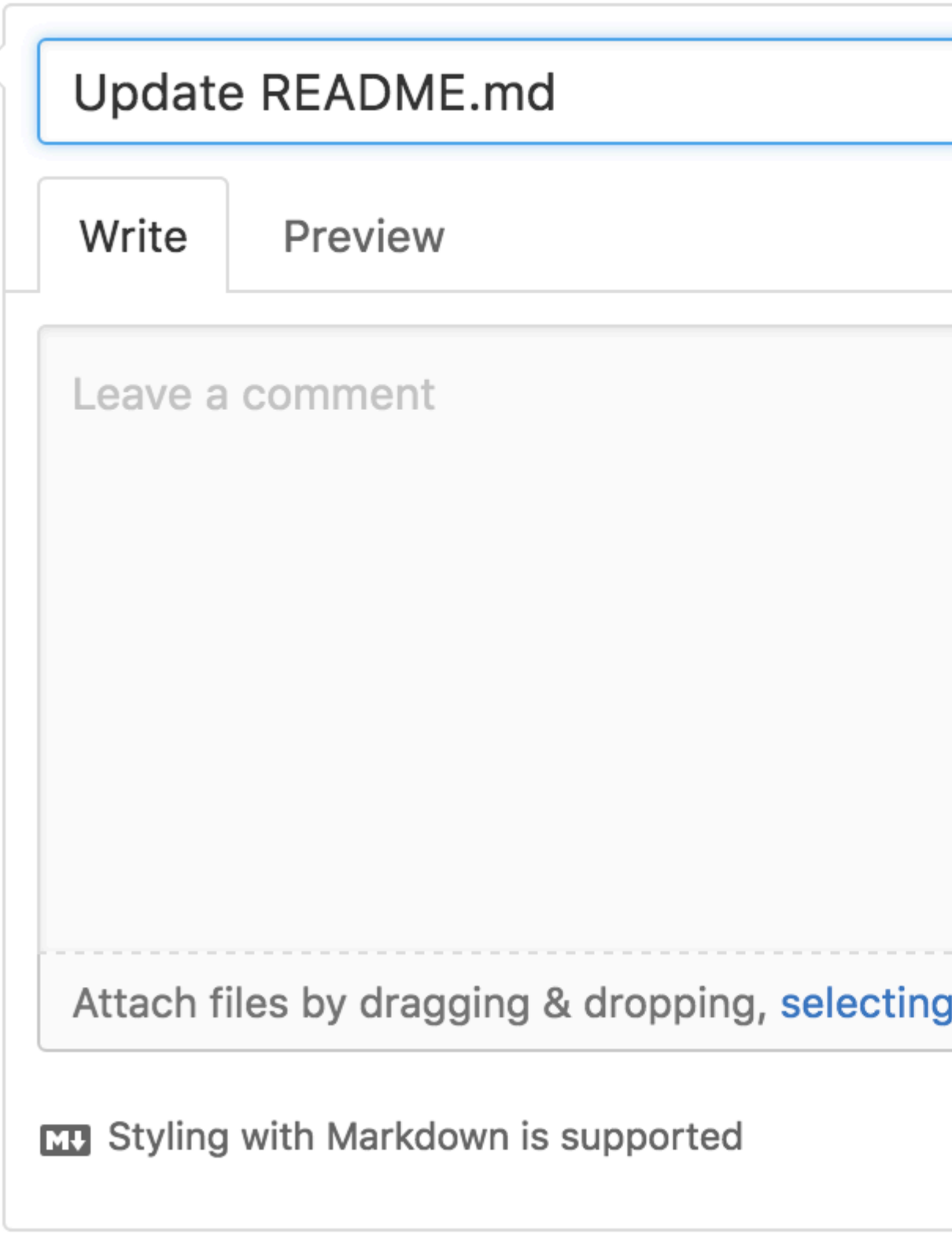
Pull-Request öffnen

Neuer Pull Request

Wann immer Sie einen Pull-Request erstellen möchten (beispielsweise durch eine kürzlich vorgenommene Änderung. Sie können dies jedoch auch mit einer älteren Änderung tun!), Sie können GitHub einen Großteil der Arbeit abnehmen und drücken Sie die grüne **Compare & Pull Request-** Schaltfläche (*NICHT MIT KLON ODER DOWNLOAD ZU KONFUSIONIEREN*) innerhalb des Benachrichtigungsfelds, das erwähnt, dass Sie gerade innerhalb eines Zweiges gedrückt haben.

Andernfalls verwenden Sie die Schaltfläche " **Neue Pull-Anforderung"** neben Ihrem Zweig.


This repository
Search
maxcell / example-so-documenta
Code
Issues 0
Pull request
Repository used for Documentation in Sta
2 commits
Your recently pushed branches:
example-branch (less than a minute ago)
Branch: example-branch
New pull request
This branch is 1 commit ahead of master.
maxcell committed on GitHub Update READ
README.md



Pull-Anfragen online lesen: https://riptutorial.com/de/github/topic/5761/pull-anfragen

Kapitel 13: Repository aus GitHub klonen

Syntax

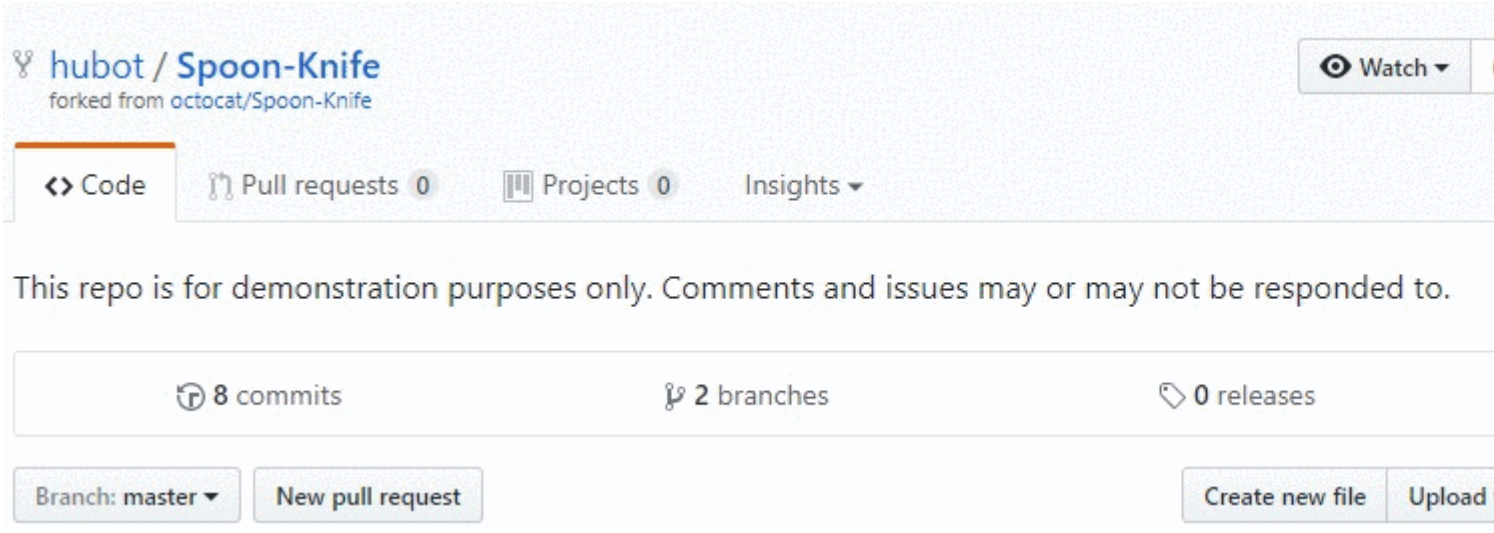
- git clone github.com/benutzername/repository

Examples

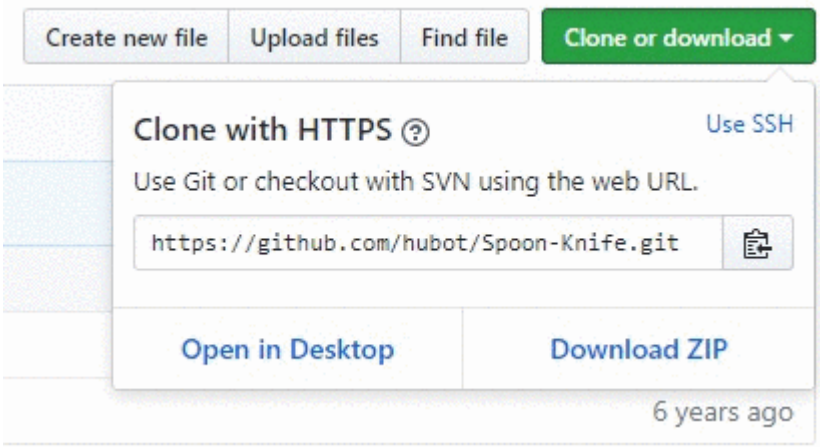
Klonen Sie ein Repository

1. Gehen Sie zu dem Repository, das Sie klonen möchten (etwa: https://github.com/ *username* / *repo*).

2. Klicken Sie rechts auf die grüne Schaltfläche mit dem Namen *clone oder download*



3. Ein kleines Fenster wird angezeigt. Kopieren Sie die URL (etwa: https://github.com/ *username* / *repo* .git).



4. Öffnen Sie ein Terminalfenster auf der Maschine, auf der Sie das Projekt klonen möchten
5. Navigieren Sie von der Befehlszeile zu dem Ort, an dem Sie das Projekt klonen möchten
6. Geben Sie den Befehl ein: git clone <copied_url_from_step_3>
7. Drücken Sie Enter
8. Etwas wie das Folgende wird erscheinen:

   Klonen in `<repo_name>` ...

   Remote: Objekte zählen: 10, fertig.

   Remote: Komprimieren von Objekten: 100% (8/8), fertig.

   entfernen: Gesamt 10 (Delta 1), wiederverwendet 10 (Delta 1)

   Auspacken von Objekten: 100% (10/10), fertig.

Repository aus GitHub klonen online lesen: https://riptutorial.com/de/github/topic/3761/repository-aus-github-klonen

Kapitel 14: Vertrauliche Daten oder große Dateien entfernen

Einführung

Wenn Sie vertrauliche Daten, z. B. ein Kennwort oder einen SSH-Schlüssel, in ein Git-Repository einbinden, können Sie sie aus dem Verlauf entfernen. Um unerwünschte Dateien vollständig aus dem Repository-Verlauf zu entfernen, können Sie entweder den Befehl git filter-branch oder den BFG Repo-Cleaner verwenden.

Bemerkungen

1. Sagen Sie Ihren Mitarbeitern, dass sie alle Zweige, die sie aus Ihrem alten (verfälschten) Repository-Verlauf erstellt haben, neu abstimmen und nicht zusammenführen sollen. Durch ein Merge-Commit könnten Sie die verdorbene Geschichte, die Sie gerade erledigt haben, ganz oder teilweise wieder einführen.
2. Nachdem einige Zeit vergangen ist und Sie zuversichtlich sind, dass git filter-branch keine unbeabsichtigten Nebenwirkungen hatte, können Sie alle Objekte in Ihrem lokalen Repository mit den folgenden Befehlen (mit Git 1.8.5 oder neuer) dereferenzieren lassen.

   git for-each-ref --format = 'delete% (refname)' refs / original | git update-ref --stdin

   git reflog verfällt --expire = now --all

   git gc --prune = jetzt

Examples

Filterzweig verwenden

```
git filter-branch --force --index-filter \
'git rm --cached --ignore-unmatch PATH-TO-YOUR-FILE-WITH-SENSITIVE-DATA' \
--prune-empty --tag-name-filter cat -- --all
```

Fügen Sie Ihre Datei mit vertraulichen Daten zu .gitignore hinzu, um sicherzustellen, dass Sie sie nicht erneut versehentlich festlegen.

```
echo "YOUR-FILE-WITH-SENSITIVE-DATA" >> .gitignore
git add .gitignore
git commit -m "Add YOUR-FILE-WITH-SENSITIVE-DATA to .gitignore"
```

Schieben Sie Ihr lokales Repo auf GitHub

```
git push origin --force --all
```

Um die vertrauliche Datei aus Ihren getaggten Releases zu entfernen, müssen Sie auch einen Push-Push gegen Ihre Git-Tags durchführen:

```
git push origin --force --tags
```

Verwendung des BFG Repo Cleaners

BFG Repo-Reiniger ist eine Alternative zum Git-Filter-Zweig. Es kann verwendet werden, um vertrauliche Daten oder große Dateien zu entfernen, die falsch wie Binaries aus der Quelle kompiliert wurden. Es ist in Scala geschrieben.

Projekt-Website: BFG Repo Cleaner

Bedarf

Die Java-Laufzeitumgebung (Java 7 oder höher - BFG v1.12.3 war die letzte Version, die Java 6 unterstützt). Die Scala-Bibliothek und alle anderen Abhängigkeiten werden in das herunterladbare Glas gefaltet.

Entfernen Sie Dateien mit vertraulichen Daten

```
bfg --delete-files YOUR-FILE-WITH-SENSITIVE-DATA
```

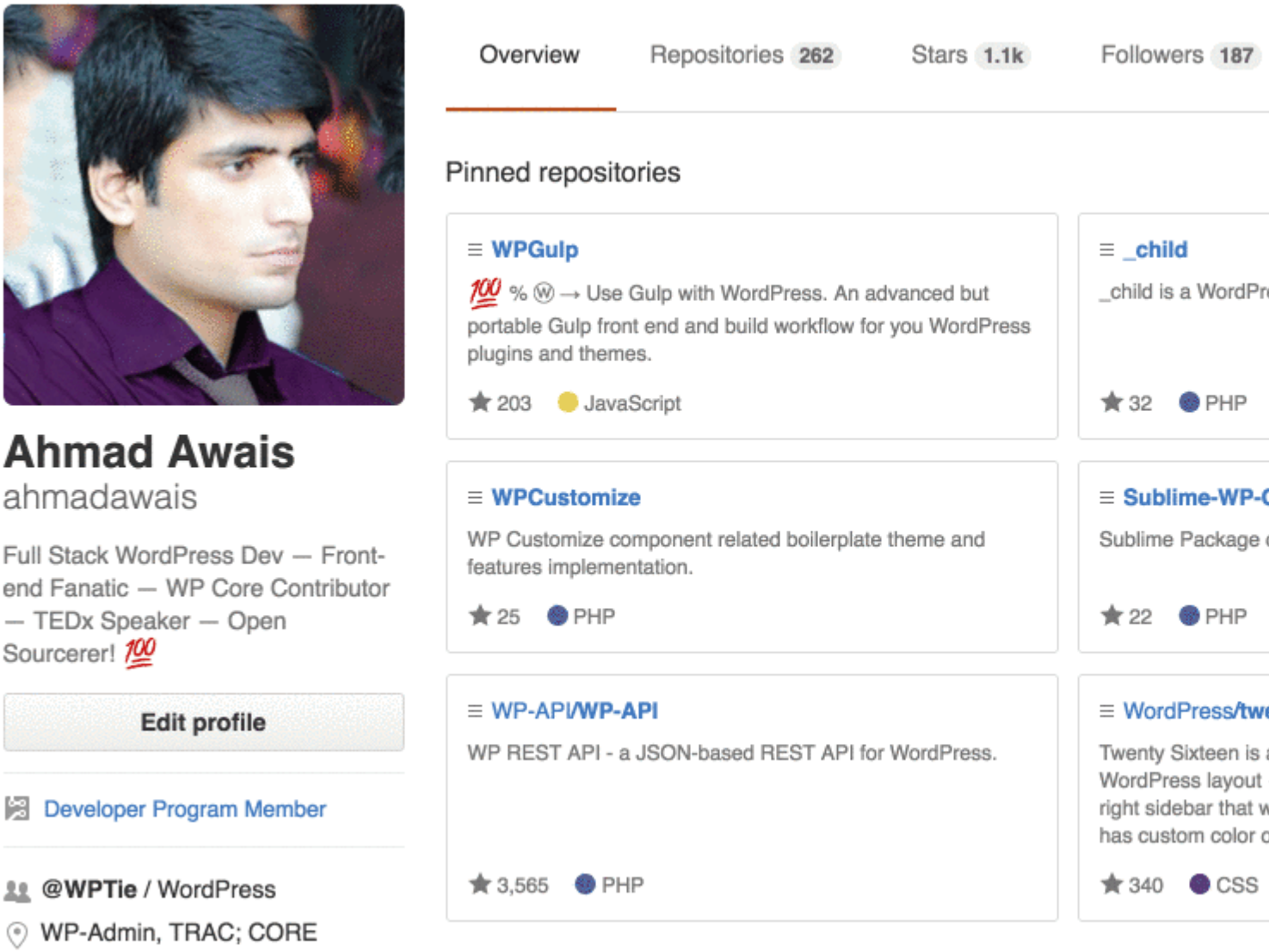
Vertrauliche Daten oder große Dateien entfernen online lesen:
https://riptutorial.com/de/github/topic/8170/vertrauliche-daten-oder-gro-e-dateien-entfernen

Kapitel 15: Wie erstelle ich benutzerdefinierte GitHub Labels?

Examples

Erstellen Sie benutzerdefinierte GitHub-Labels!

Hier ist ein kurzes GIF, um den Prozess so einfach wie möglich zu gestalten.



Beschriftungen können auf Probleme und Pull-Anfragen angewendet werden, um die Priorität, Kategorie oder andere nützliche Informationen anzuzeigen.

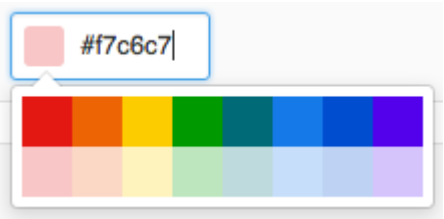
Navigieren Sie auf GitHub zur Hauptseite des Repositorys.

1. Klicken Sie unter Ihrem Repository-Namen auf Probleme oder Pull-Anforderungen.
2. Klicken Sie auf die Schaltfläche Beschriftungen neben dem Suchfeld auf Beschriftungen.

3. Klicken Sie auf Neues Label, um ein neues Label zu erstellen, oder klicken Sie auf Bearbeiten, um ein vorhandenes Label zu bearbeiten.

4. Geben Sie in das Textfeld Ihren neuen Labelnamen ein.

5. Wählen Sie aus der Farbleiste eine Farbe für das Etikett aus. Sie können diese Farbe anpassen, indem Sie die Hexadezimalzahl über der Farbleiste bearbeiten.



6. Klicken Sie auf Etikett erstellen, um das neue Etikett zu speichern.

Ich hoffe, es hilft. Wenn es stimmt, stimmt es.

Wie erstelle ich benutzerdefinierte GitHub Labels? online lesen:
https://riptutorial.com/de/github/topic/7159/wie-erstelle-ich-benutzerdefinierte-github-labels-

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit Github	BadAllOff, BrechtDeMan, Community, H. Pauwelyn, Hamzawey, Hugo, intboolstring, Kronos, Mateusz Piotrowski, Minhas Kamal, Nicholas Qiao, rpadovani
2	Aktualisieren Sie ein gegabeltes Repository	Derek Liu
3	Anzeigen der GitHub-Timeline / -Feeds auf Ihrer Website	Hugo, Newton Joshua
4	Gist verwenden	Kronos, tehp
5	GitHub Buttons verwenden	James Kerrane
6	GitHub Desktop	creyD
7	GitHub sichern	geek1011
8	GitHub-Seiten	BrechtDeMan, geek1011, Mono
9	Laden Sie eine einzelne Datei aus dem GitHub-Repository herunter	ownsourcing dev training
10	Mit Gitflow arbeiten	Derek Liu
11	Probleme	geek1011, Hamzawey, SuperBiasedMan
12	Pull-Anfragen	Maxcell
13	Repository aus GitHub klonen	demonplus, geek1011, Hamzawey, James Kerrane, Mateusz Piotrowski
14	Vertrauliche Daten oder große Dateien entfernen	Gautam Krishna R, Kronos
15	Wie erstelle ich	Ahmad Awais

	benutzerdefinierte GitHub Labels?	