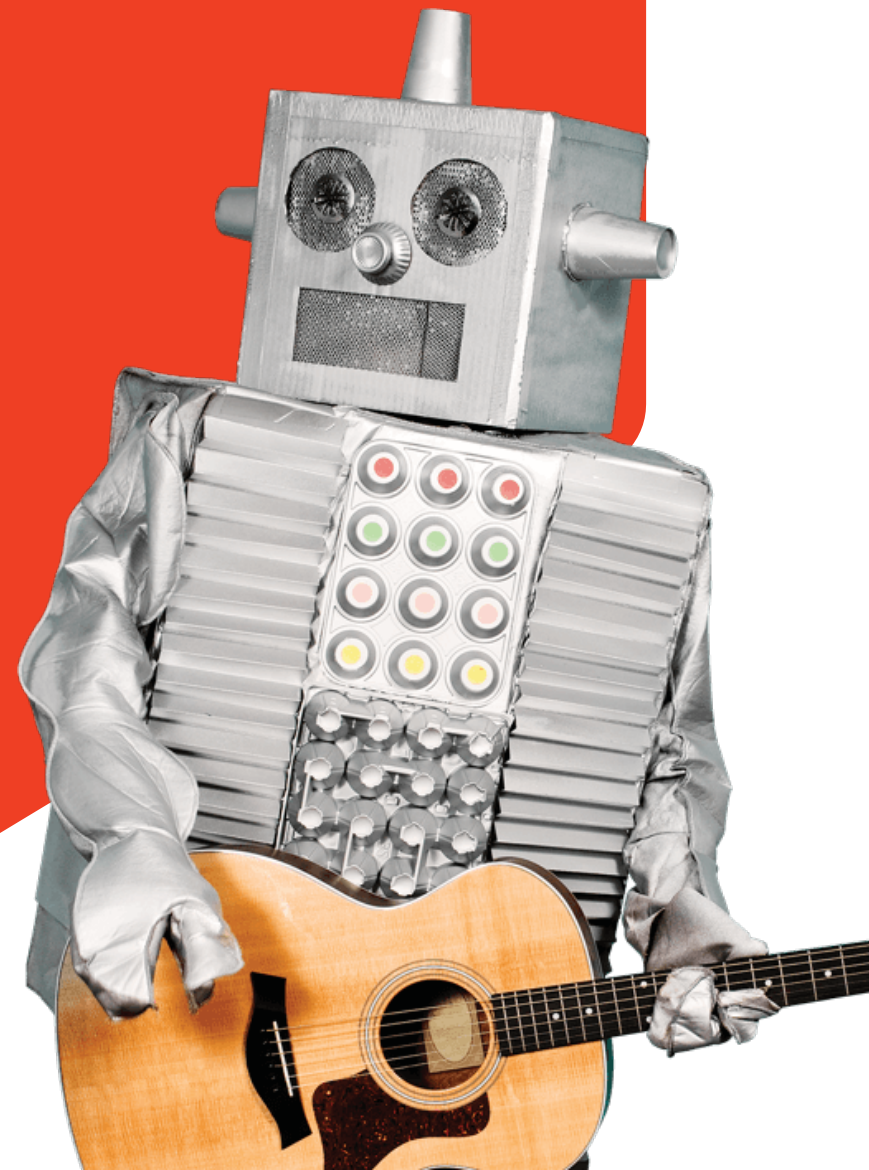
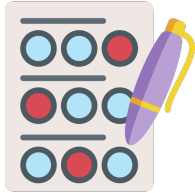


ハンズオンAの確認





クイズの時間!



GitとGitHubはどのように関係していますか？

- ヒント：「hub」とは、活動またはネットワークの中心点です。



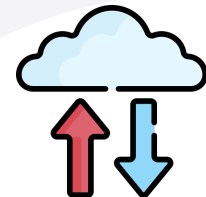
「git add」と「git commit」のどちらを初めに使いますか？

- ヒント：「commit」とは、特定の行動（結婚など）を約束することです。



「git push」コマンドは何をしますか？

- ヒント：「git push」コマンドは「git fetch」コマンドの逆の行動を行います。



GitHub 例 - Apple

アップル、「Password Manager Resources」をオープンソースで公開

- japan.zdnet.com/article/35154931/
- Appleが、パスワードマネージャーなどのアプリの開発者向けに、強力なパスワードを生成できるよう支援するための一連のツールとリソースを無償公開している。
<https://github.com/apple/password-manager-resources>

The screenshot shows the GitHub repository page for 'apple/password-manager-resources'. At the top, it displays the repository name, a 'Watch' button with 62 notifications, a 'Star' button with 1.9k stars, and a 'Fork' button with 126 forks. Below this, there are tabs for 'Code', 'Issues' (23), 'Pull requests' (16), 'Security' (0), and 'Insights'. A description states: 'A place for creators and users of password managers to collaborate on resources to make password management better.' Below the description, statistics show 108 commits, 2 branches, 0 packages, 0 releases, 47 contributors, and the MIT license. A table lists recent commits by user 'ameya-pandilwar', including files like '.github', 'quirks', 'tools', '.gitignore', 'CODE_OF_CONDUCT.md', 'CONTRIBUTING.md', 'LICENSE.md', and 'README.md'. The bottom section shows the 'README.md' content, which includes a 'Welcome!' message and a description of the project's purpose: 'The Password Manager Resources project exists so creators of password managers can collaborate on resources to make password management better for users. Resources currently consist of data, or "quirks", as well as code.' It also defines 'Quirk' as a term from web browser development.

apple / password-manager-resources

Watch 62 Star 1.9k Fork 126

Code Issues 23 Pull requests 16 Security 0 Insights

A place for creators and users of password managers to collaborate on resources to make password management better.

108 commits 2 branches 0 packages 0 releases 47 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

File	Commit Message	Time Ago
.github	Add a GitHub action to lint JSONs (#130)	6 hours ago
quirks	Include rule for github.com (#86)	21 minutes ago
tools	Initial commit.	last month
.gitignore	Initial commit.	last month
CODE_OF_CONDUCT.md	Initial commit.	last month
CONTRIBUTING.md	Add a CONTRIBUTING.md file. (#39)	3 days ago
LICENSE.md	Add a CONTRIBUTING.md file. (#39)	3 days ago
README.md	Create websites-that-append-2fa-to-password.json (#128)	36 minutes ago

README.md

Password Manager Resources

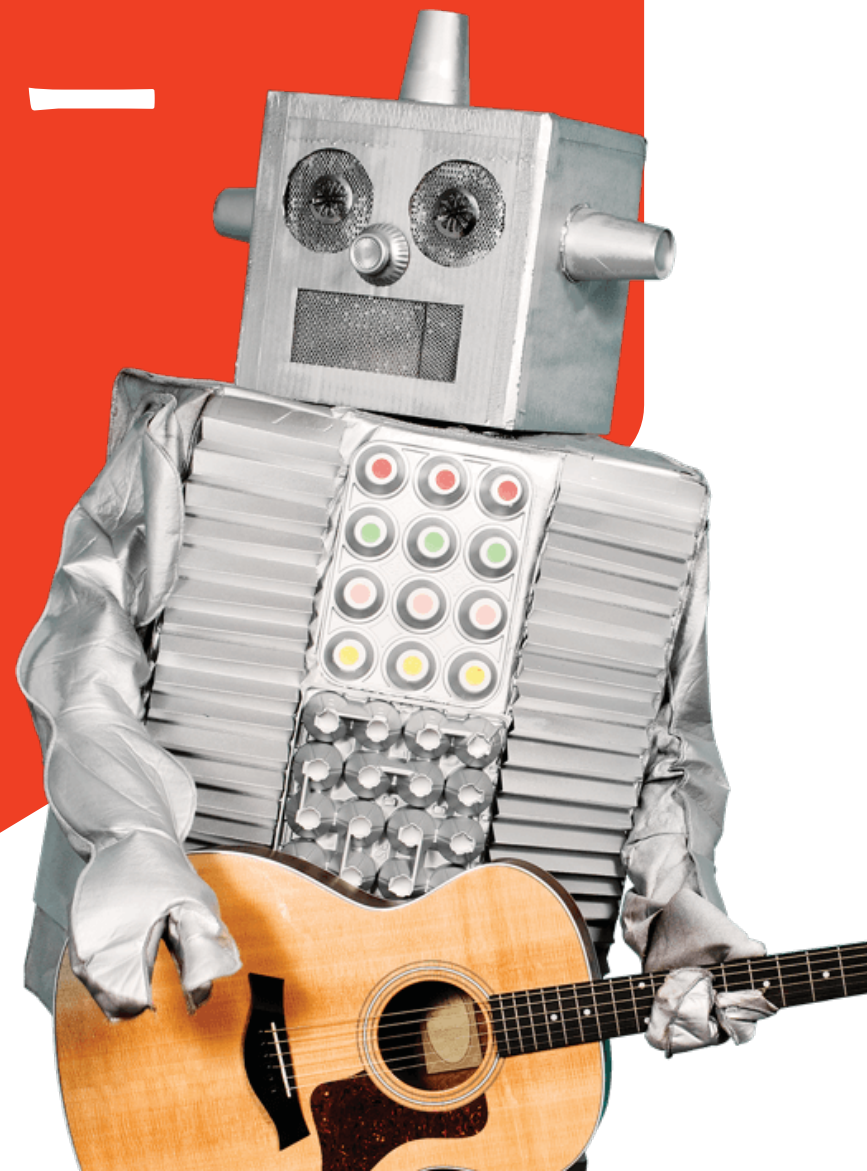
Welcome!

The Password Manager Resources project exists so creators of password managers can collaborate on resources to make password management better for users. Resources currently consist of data, or "quirks", as well as code.

"Quirk" is a term from web browser development that refers to a website-specific, hard-coded behavior to work around an issue with a website that can't be fixed in a principled, universal way. In this project, it has the same meaning. Although ideally, the industry will work to eliminate the need for all of the quirks in this project, there's value in customizing behaviors to ensure better user experience. The current quirks are:

ブランチの作成とマージ！

Hands-on B



新しいブランチを作成する

```
$ git checkout -b develop  
Switched to a new branch  
'develop'
```

```
$ git branch  
* develop  
master
```

リポジトリにはデフォルトで**master**ブランチが存在します。

まず、**develop**という名前のブランチを作成します

git checkout -b <branch name>

- リポジトリに新しいブランチを作成して、そのブランチに移動するコマンド

git branch

- ブランチ一覧を表示するコマンド
- アスタリスク (*) がついているブランチが現在のブランチです

develop ブランチへの変更

```
$ touch develop_file.md
$ git add develop_file.md
$ git commit -m "develop only"
[develop f946eb0] File only in develop branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 develop_file.md
$ git status
On branch develop
nothing to commit, working tree clean
$ git push -u origin develop
```

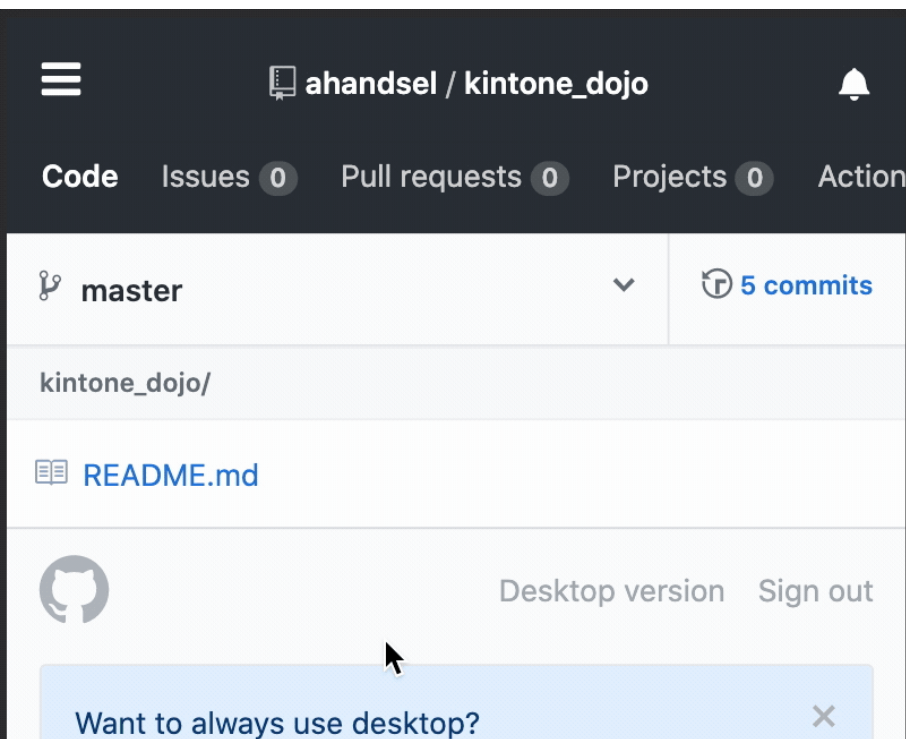
develop ブランチ上でファイルを作成します。

git add と **git commit** を実行して、ローカルリポジトリに保存します。

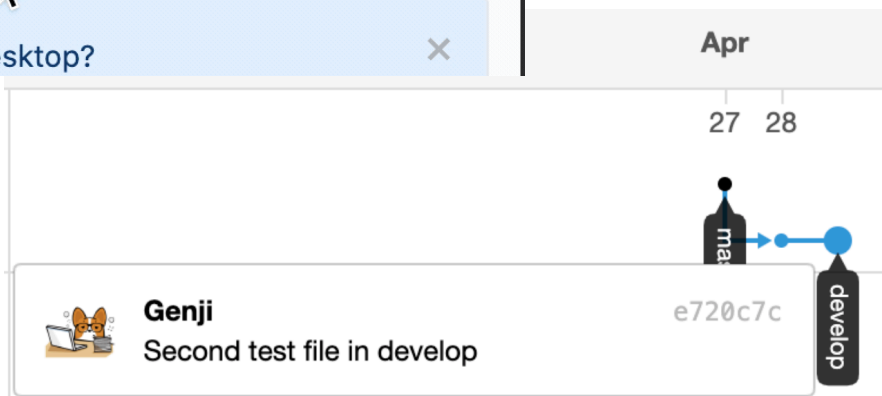
git status を実行して、変更が保存されたことを確認します。

git push を実行して、GitHub リポジトリに変更をプッシュします。

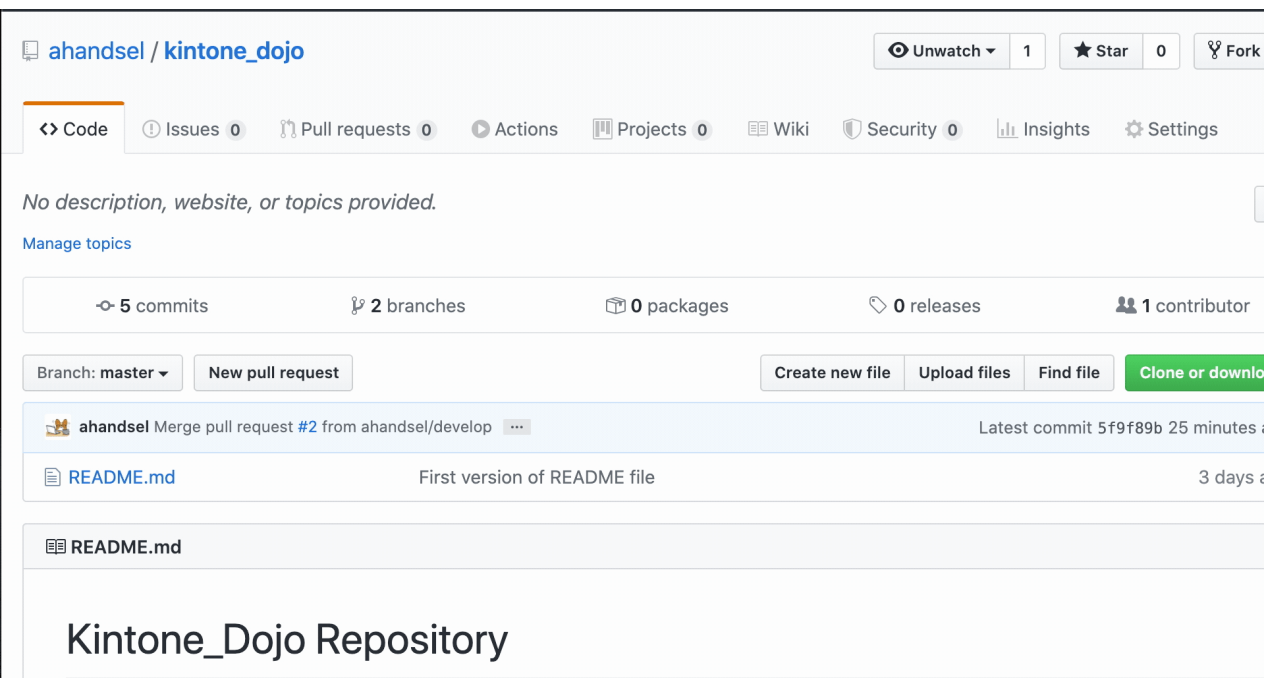
GitHubでの変更の確認



- **develop** ブランチでのみ新しいファイルが追加されました
- **develop** ブランチに別のファイルを追加します
- 次に、**Network graph**を表示します
- <https://github.com/USER/REPO/network>

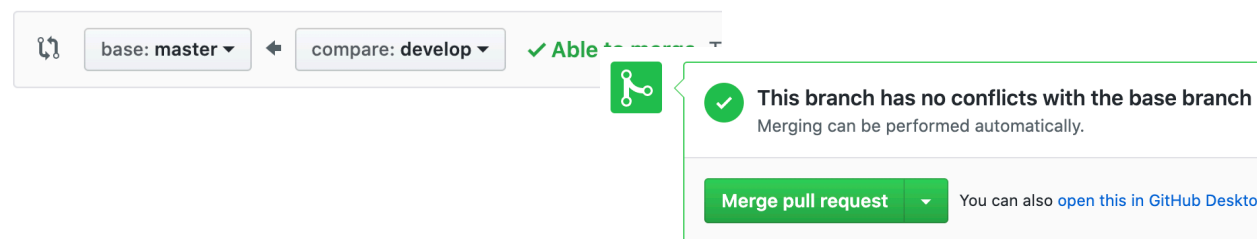


プルリクエストの作成とマージ



Open a pull request

Create a new pull request by comparing changes across two branches.



- **Pull Requests**で、実際にファイルが変更される前に、他のユーザーの変更などを確認できます。
 - コードレビューなどに使われます。
- **develop**ブランチを**master**ブランチにマージするために、GitHubで**Pull request**を作成します。
- 変更を確認し、Pull Requestをマージします。
- **master**ブランチに2つの新しいファイルが表示されました！

GitHub repoからLocal repoへの更新

```
$ git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 4 commits & can be fast-forwarded
(use "git pull" to update your local branch)

$ git pull origin master
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), 631 bytes | 210.00 KiB/s, done.
From https://github.com/ahandsel/kintone_dojo
* branch      master    -> FETCH_HEAD
   5f9f89b..1438ca5 master -> origin/master
Updating d775d42..1438ca5
Fast-forward
 2nd_file.md   | 0
 develop_file.md | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 2nd_file.md
 create mode 100644 develop_file.md

$ git branch -d develop
Deleted branch develop (was c6e6c83).

$ git branch
* master
```

```
$ git checkout master
$ git pull origin master
$ git branch -d develop
$ git branch
```

現在、GitHubリポジトリはローカルリポジトリよりもファイルが最新になっています。

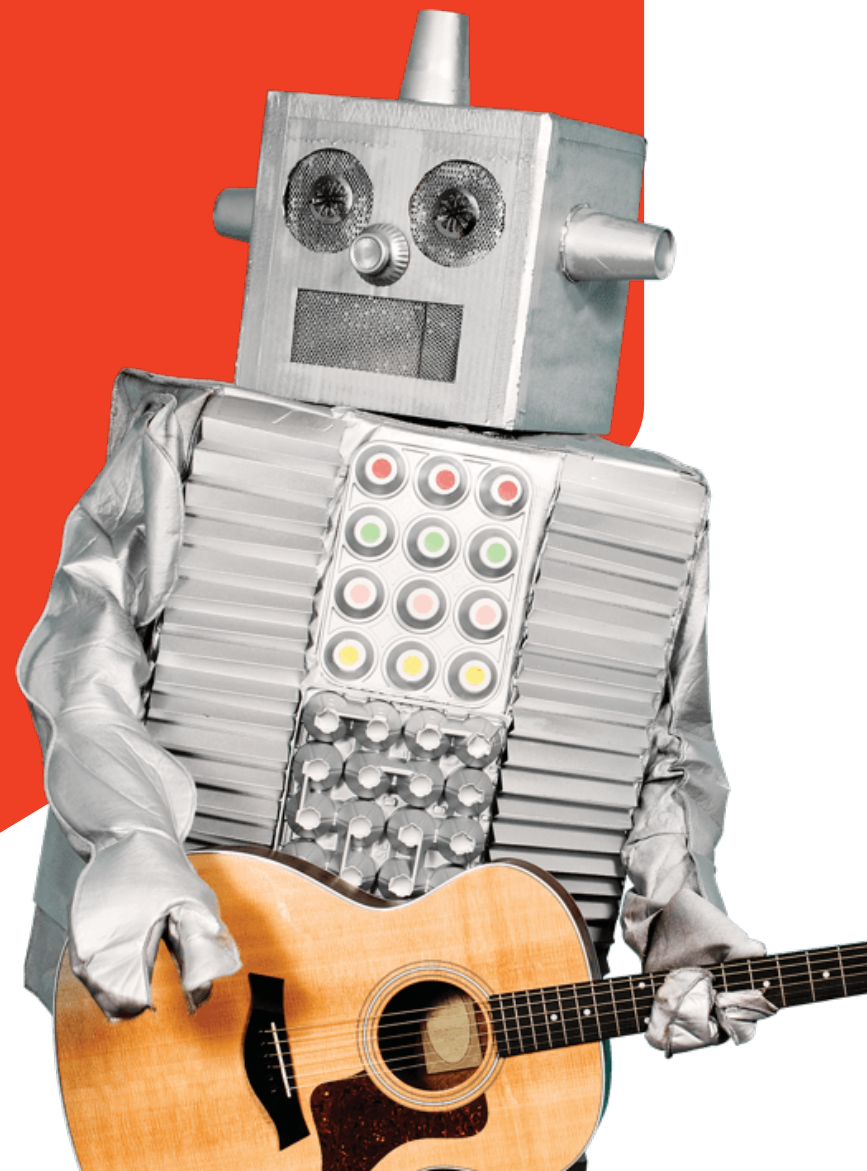
- **git pull** コマンドを使用して対応します

git pull origin master

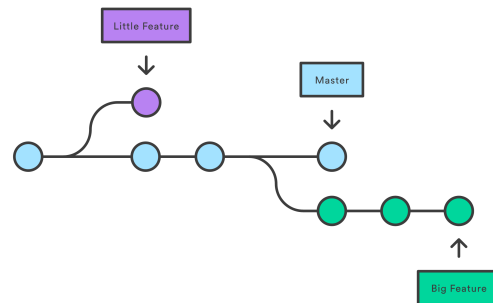
- 最新のリポジトリのバージョンをGitHubからローカルにpullします

ブランチとは？

ハンズオンB の概要



Git ブランチ



ブランチとは？

- コミットの動くポインタ。
- 複数のタイムラインで管理し、メインラインに影響を与えずに
となく変更することができます

git checkout

- ブランチを切り替えるコマンド。

git branch -d <branch-name>

- ブランチを削除するコマンド。

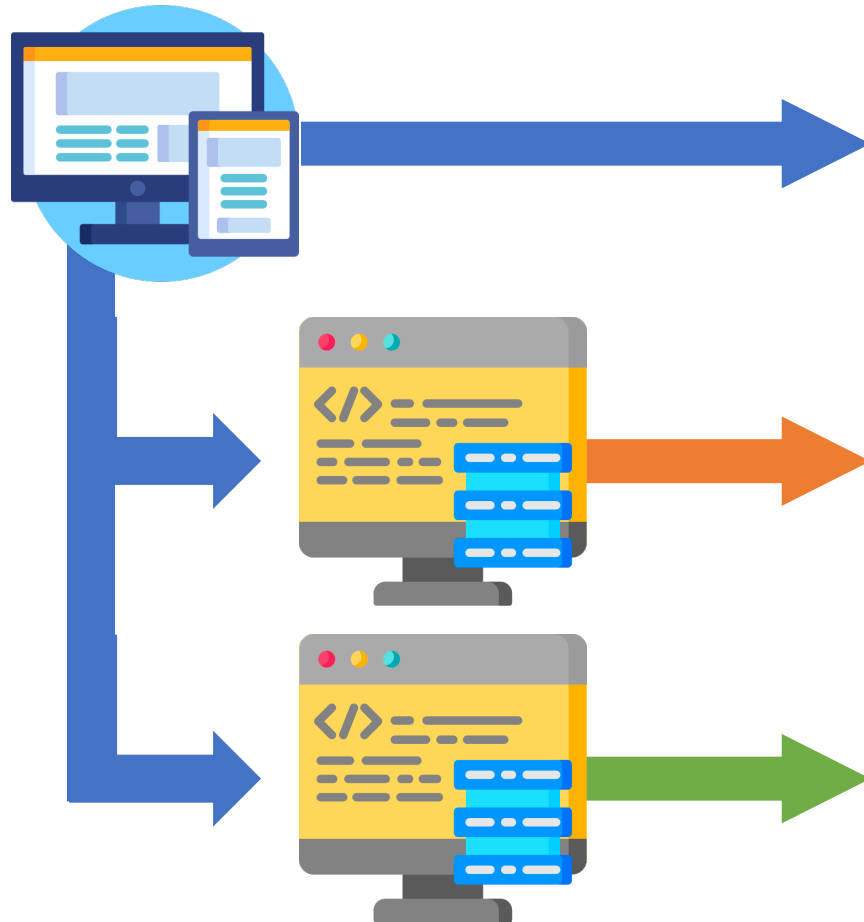
なぜブランチを使うのですか？

- 安定した版、開発版、実験版などを
を離隔するため。

例：このGitHubスライド

- 各ハンズオンとコンセプトセクションを
ブランチにすることができます。
- それぞれを同時に開発できます。

ブランチ・ウェブサイト



masterブランチには、Webサイトを実行するコードが存在します。

- もしmasterブランチで変更が加わると、ユーザーに影響を与えてしまいます！！

2人の開発者が同時にWebサイトを変更したい場合、3つのブランチを作成します

- **master** → ライブコード
- **feature_A** → 開発者Aが実装する
- **feature_B** → 開発者Bが実装する

開発が完了したら、ブランチをマージします！

Git Push vs Pull - Teamwork



git push

- 「アップロード」コマンド
- 「プッシュ」は、ターゲットリポジトリに変更を強制します。
 - [あなたのコード]-プッシュ->[ターゲット]
- 「プッシュリクエスト」は、変更をプッシュするように要求するターゲットリポジトリです。



git pull

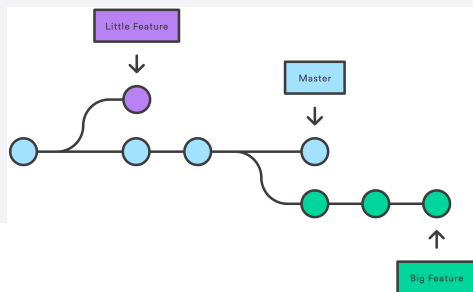
- 「ダウンロード」コマンド
- 「プル」はターゲットリポジトリから変更を取得します
 - [あなたのコード]-プル-[ターゲット]
- 「プルリクエスト」とは、変更を取得するためにターゲットリポジトリをリクエストすることです。



Hands-on C Review

git checkout -b develop

ブランチを切り替えるコマンド。



ブランチを使う理由

コードの開発、テスト、公開バージョンなどを分離する

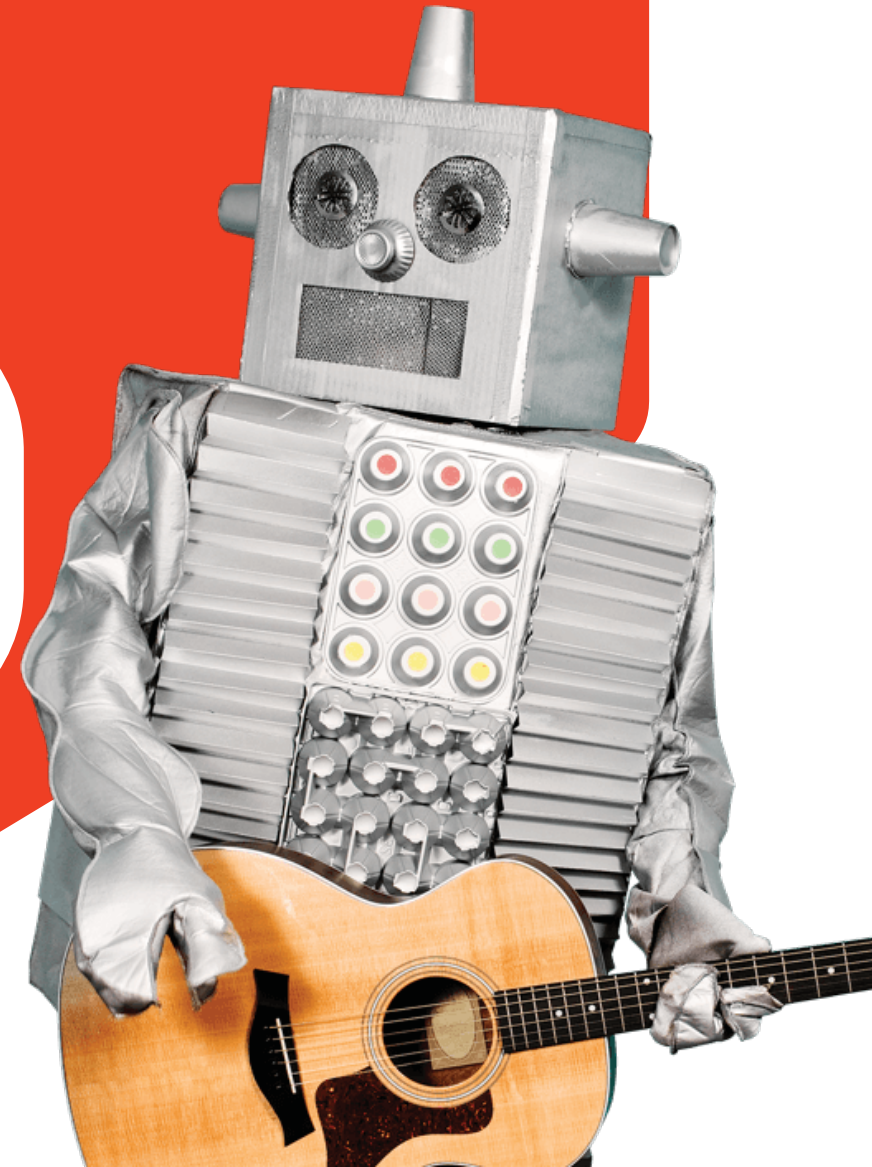
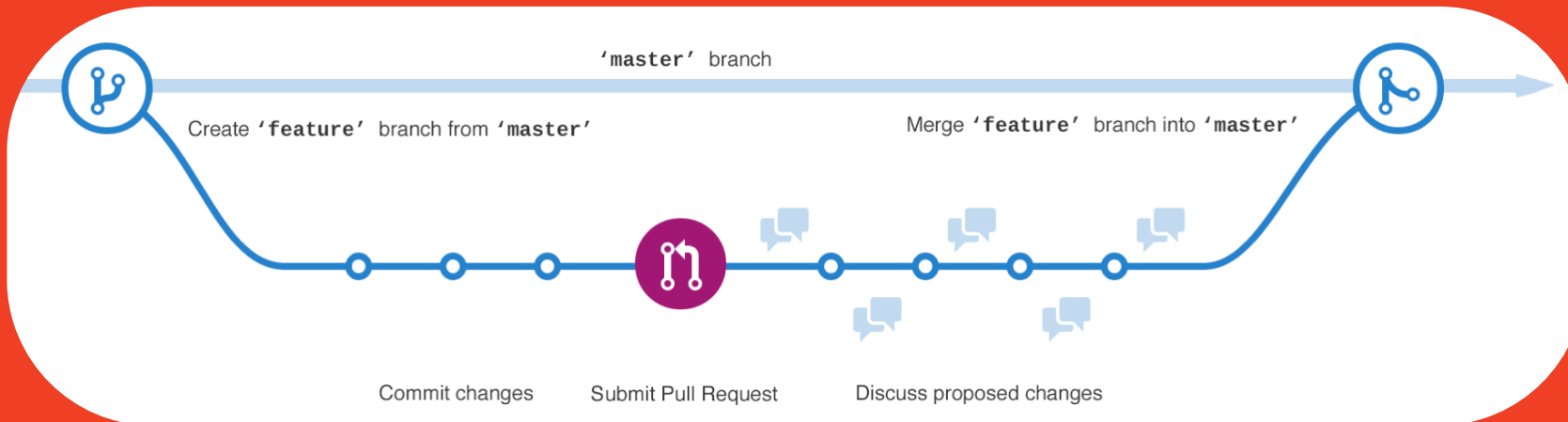


Pull Requests と git pull

「プルリクエスト」とは、変更を取得するためにターゲットリポジトリをリクエストすることです。

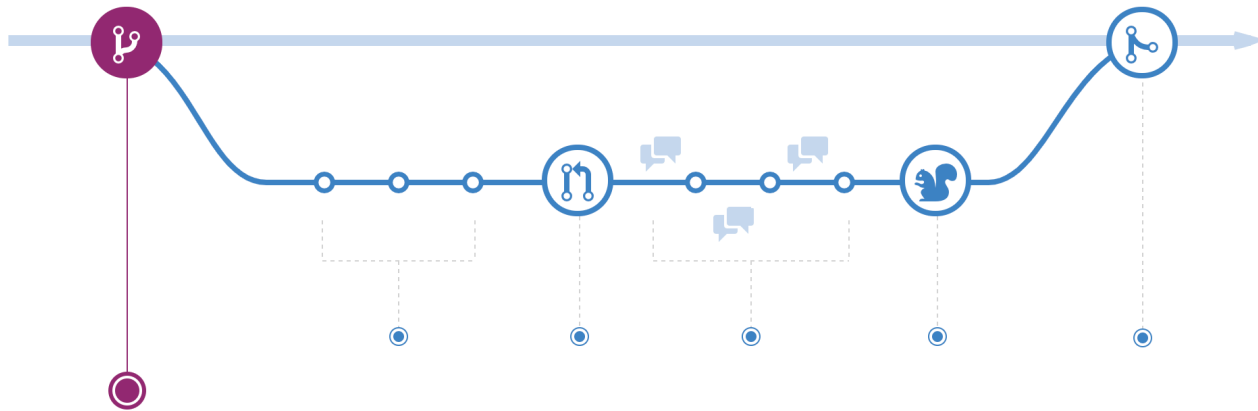


GitHub Workflow



<https://guides.github.com/introduction/flow/>

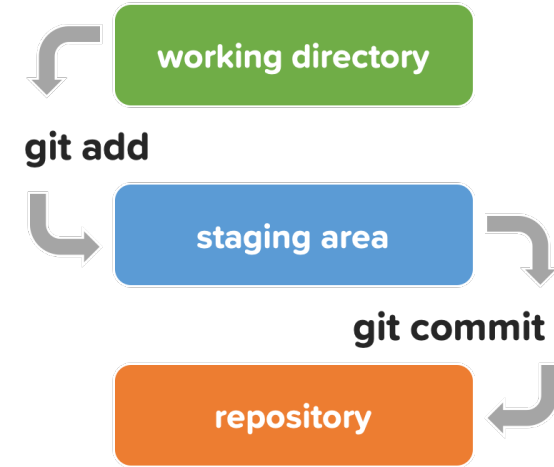
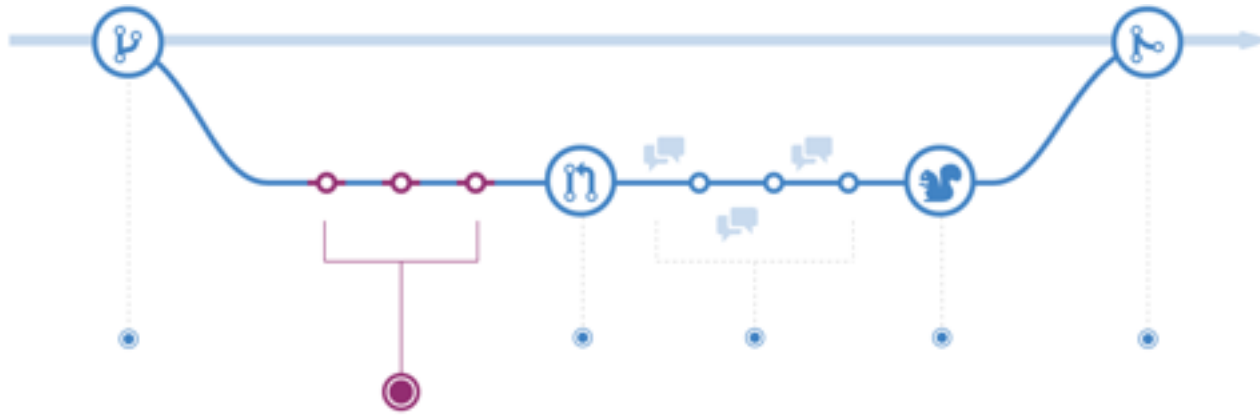
#1 - Create a Branch



```
$ git checkout -b develop  
$ git branch
```

- webサイトの本番コードは master ブランチに保管します。
 - masterでの変更はユーザーにも影響を与えてしまいます！
- もし新しいページを作成したい場合、まず新しく feature ブランチを作成します。
 - featureブランチで新しいページの開発を行います。

#2 - Making a Commit



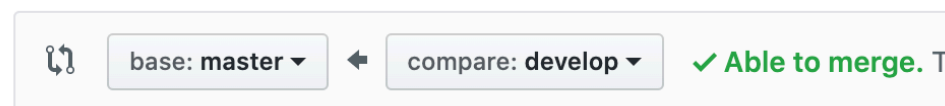
- feature ブランチでコードを実装します。
- コード実装が完了したら、commitを作成します。
- Commitを作成することで変更履歴を確認できます。
- コミットにより、ロールバックと参照が可能になります。

#3 - Open a Pull Request



Open a pull request

Create a new pull request by comparing changes across two branches.



- 実装内容を他の人と共有する準備ができたなら、 Pull Requestを作成します。
- Pull Requestを作成することでコードレビューの準備を行います。
- GitHubの[プルリクエスト]タブでコメントを作成します。

#4 - Test

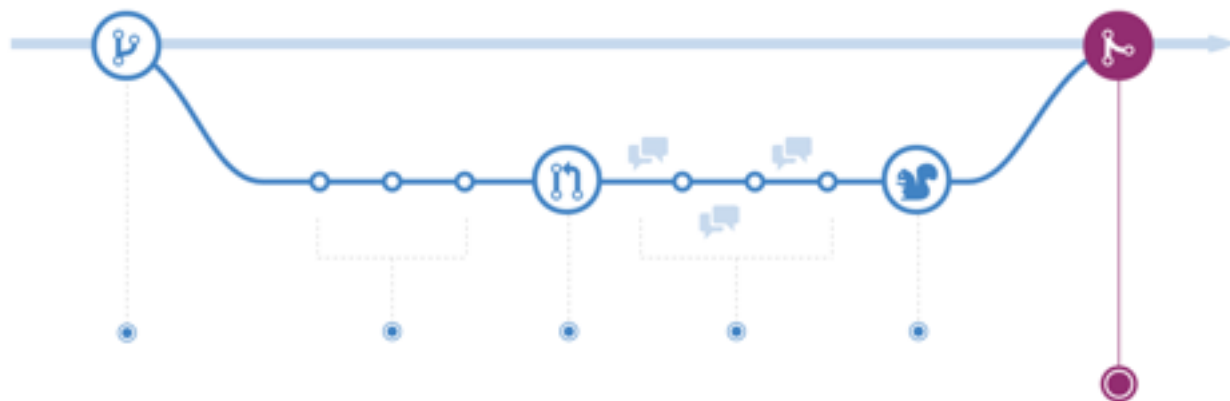


新しいテーマを追加
Zendeskの標準テーマを使用するか、独自に作成したテーマをインポートします



- コードレビューの後、コードをテスト環境にデプロイして問題なく機能することを確認します。
- 例：Webサイトのレイアウトや動作が期待どおりかどうかをZendesk Themes版などでテストします。

#5 - Merge to Master!



This branch has no conflicts with the base branch

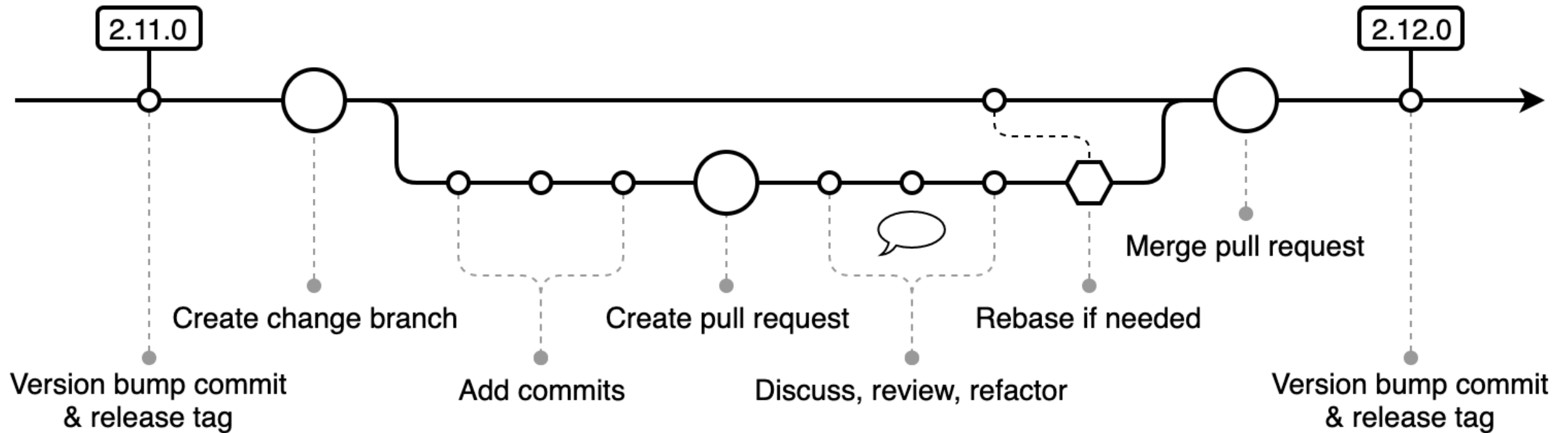
Merging can be performed automatically.

Merge pull request

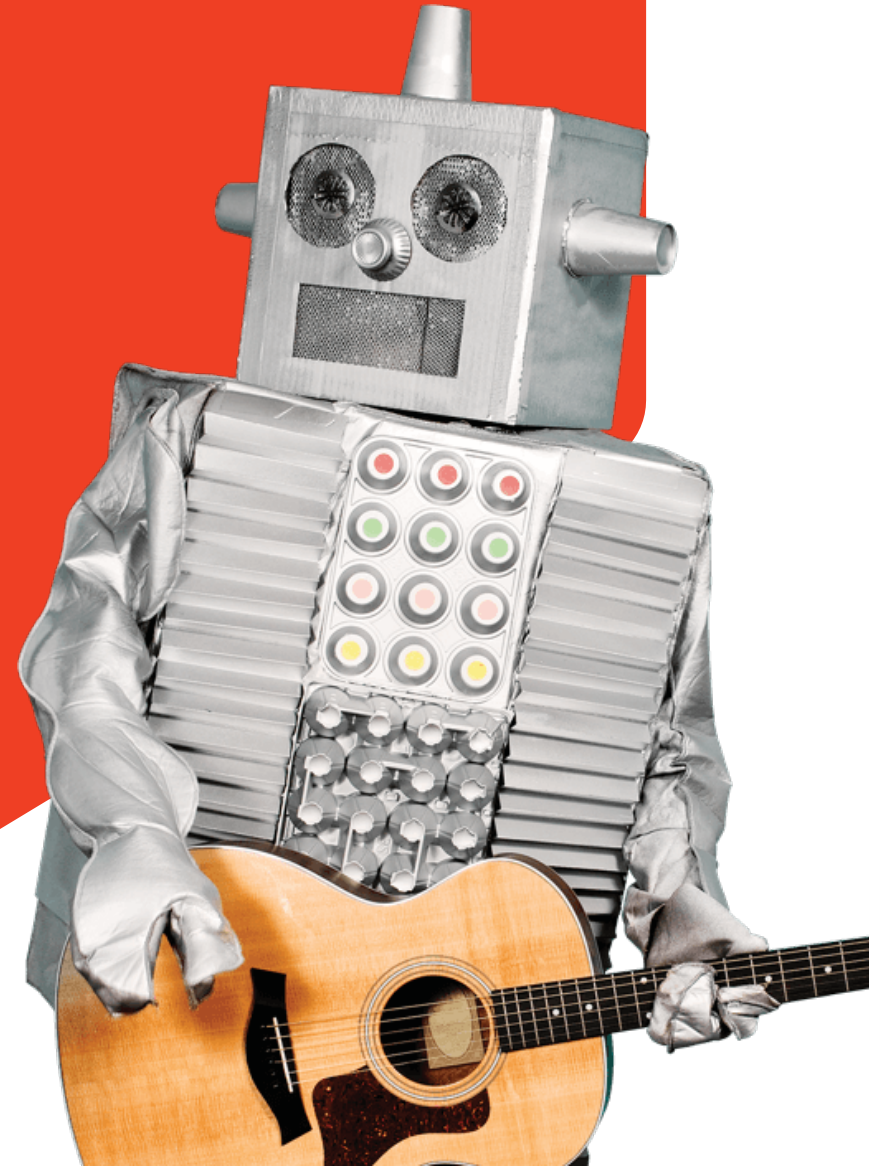
You can also [open this in GitHub Desktop](#)

- コードを master ブランチにマージ!
 - これで実装内容が有効になり、ユーザーもwebサイトの変更を確認できます。
- 統合すると、 Pull Request は参照点として機能します。
- 例：新しいロゴのプルリクエストは、ロゴ変更の「前」と「後」を示す参照点になります。


Overview of GitHub Workflow

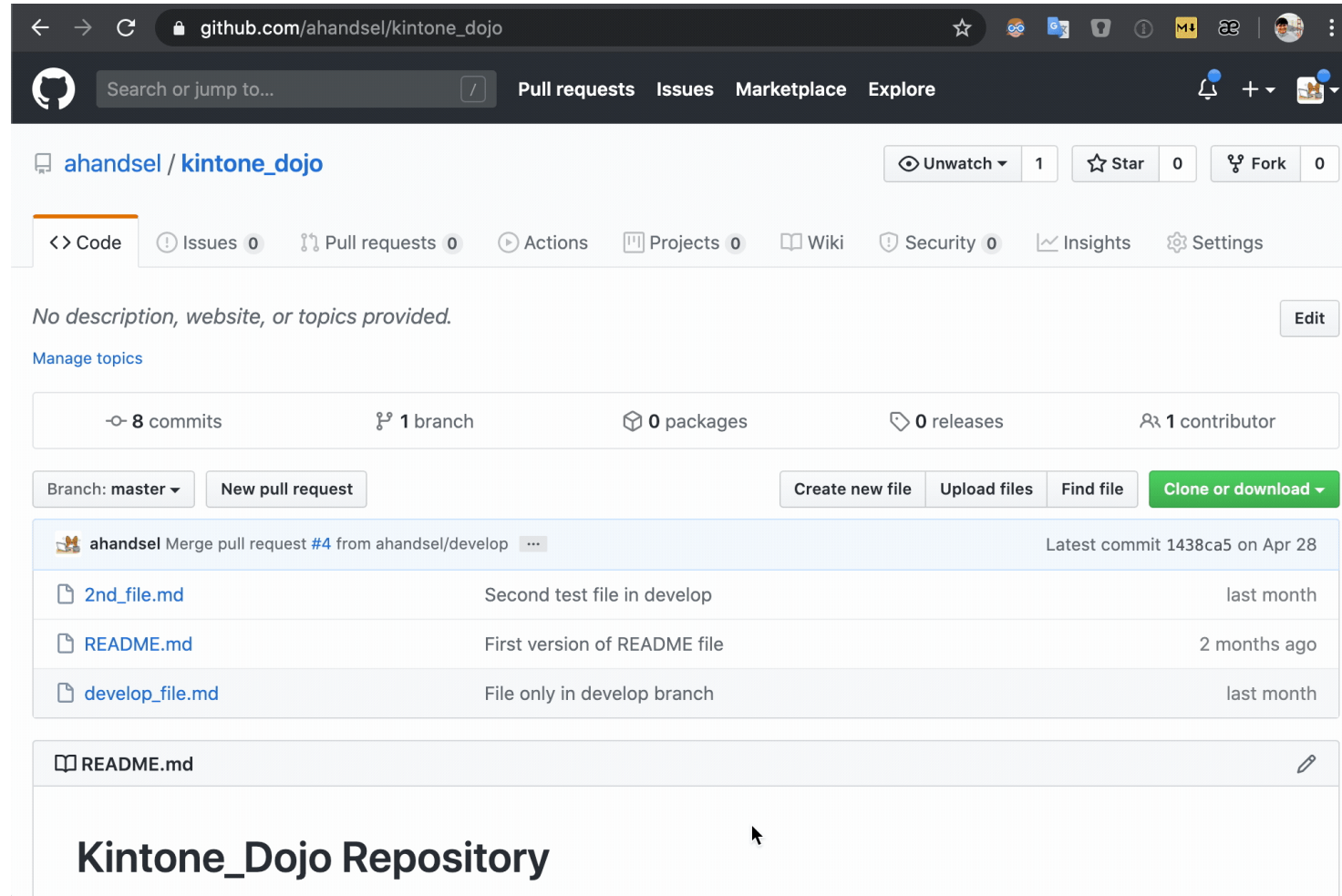


Kintone Dojo Workflow



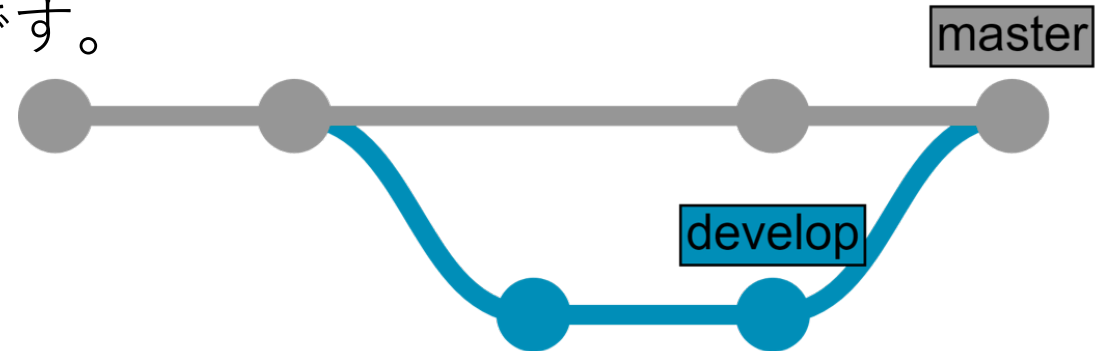
Setup

- GithubのKintone_Dojoリポジトリに移動します。
 - 例 : github.com/ahandsel/kintone_dojo
- [ Settings]
- > [Manage access]
- > [Invite a collaborator]
- Sohei Miyakura (miyass)
- Genji Fujimori (ahandsel)



Workflow

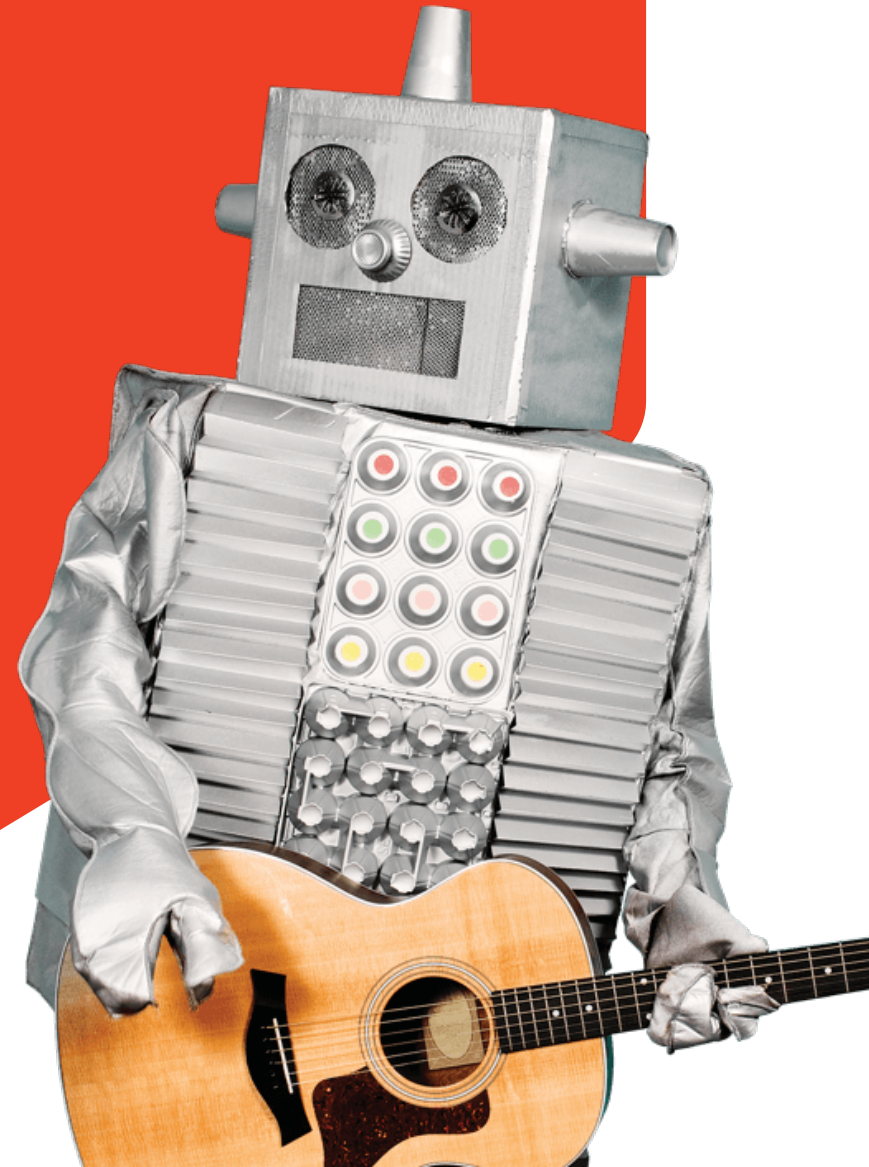
- **master** ブランチ
 - 最終的なコードとメモが格納される場所です。
 - つまり、コードレビューに合格したファイルが保存される場所。
- **develop** ブランチ
 - 開発中のコードが格納される場所です。



Workflow

- #1 - 割り当てごとにフォルダを作成します。 (masterブランチ上)
- #2 - **develop-task-#** ブランチを作成する
 - # = タスク番号
 - 例：develop-task-01はタスク番号1用です。
 - 進行中のHTMLおよびJSファイルをここに保存します。
 - Markdown (.md) ファイルとして関連するメモを含めます。
- #3 - コードレビューが必要なときにプルリクエストを行う
 - Sohei や Genjiをタグする。
- #4 - レビューが成功した後
 - develop ブランチをmasterにマージ！

GitHub Website Overview



GitHub Repository

<> Code

! Issues 0

Pull requests 0

Actions

Projects 0

Wiki

Security 0

Insights

Settings

- GitHub Repository - Code

- コードやドキュメントなど、プロジェクトに関連するすべてのファイルを保持するコンテナです。
- GitHubに保存されているすべてのファイルは、gitでバージョン管理されています
- リポジトリはしばしば「リポ」と短縮されて呼ばれます

- README.md

- リポジトリの下部に表示される、プロジェクトの紹介などを表示するためのファイルです

GitHub Issues & Pull Request



The screenshot shows the GitHub repository navigation bar. The 'Issues' tab is selected and highlighted with an orange underline. Other tabs include Code, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Each tab has a small icon and a count (0) next to it.

- Issues

- ユーザーがリポジトリの内容に関して議論する場所
- 課題をユーザーに割り当て、ラベルを追加して読みやすくすることができます



The screenshot shows the GitHub repository navigation bar. The 'Pull requests' tab is selected and highlighted with an orange underline. Other tabs include Code, Issues, Actions, Projects, Wiki, Security, Insights, and Settings. Each tab has a small icon and a count (0) next to it.

- Pull Request

- ユーザーがリポジトリに変更を加えたいときに使われます
- 例：README.md ファイルを新しく追加したい時

GitHubの概要



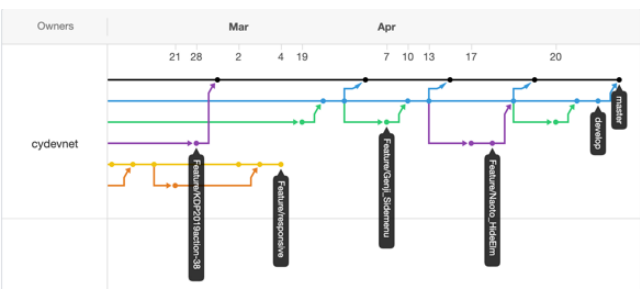
- Project boards : KANBAN形式でのタスクボードです



- Wiki : 関連するプロジェクトドキュメントの作成と保存ができます



- Insight : リポジトリの分析ツール :
 - **Network** グラフ : コミットとブランチをタイムラインで視覚化します
 - **Pulse** : 進行中、あるいは完了したタスクを表示します



GitHubのパーツ

Branch

- コードの代替タイムライン
- 例：マスター、開発、機能/ xxx

Commit

- ファイルの変更をリポジトリに保存する

Pull Request

- 提案している変更を他の人と共有する

Merge Pull Request

- 実際にブランチ（マスターなど）を変更して更新し、