

ハイプロ 2021/11/19



...

開発を安全かつ爆速で進める

藤島 拓大

自己紹介



:tingtt



:tiTakung

フルスタックエンジニアになりたい...

名前: 藤島 拓大

得意:

TypeScript, docker, git

やってる/やりたい:

VPN構築, 自宅サーバ, DNS, CI/CD

話すこと

- gitとは
- 実践フロー
- Tipsいろいろ



gitとは



バージョン管理ツール

Gra...	Description	Date	Author	Com...
●	[doc] APIのmysql接続用envファイルについて	19 Oct 20...	tingtt	45a6f240
●	[doc] クエリ発行メソッドのUsage	19 Oct 20...	tingtt	d81dfc84
●	[add] mysql接続とクエリ発行用のメソッド	19 Oct 20...	tingtt	dd71e361
●	[npm] install 'serverless-mysql'	18 Oct 20...	tingtt	475326b6

- タスクや機能でバージョンを区切ってコミットする
- Diff(差分)で管理される

Diff

- 直前のコミットから
の変更差分

```
// クエリ発行
const selectQueryResult: any = await query(
  "SELECT id, name, parent_id, theme_color, pinned, `order`, hidden FROM tags WH
  [user_id, tag_id]

// クエリ結果のチェック
if (!Array.isArray(selectQueryResult)) {
  throw new Error("Error: Query returned unsupported response")
}
if (selectQueryResult.length === 0) {
  res.status(404).json({ message: "Tag not found" })
  return
}

const embedResult: any = await Promise.all(
  selectQueryResult.map(async (value: any) => {
    // クエリ発行
    const sub_tags: any = await query(
      "SELECT id, name, theme_color, pinned, `order`, hidden FROM tags WHERE use
      [user_id, value.id]
    )
    // クエリ結果のチェック
    if (!Array.isArray(sub_tags)) {
      throw new Error("Error: Query returned unsupported response")
    }
    if (sub_tags.length !== 0) {
      value.sub_tags = sub_tags
    }
    return value
  })
)
res.status(200).json(embedResult[0])
return
```

```
// クエリ発行
const selectQueryResult: any = await query(
  `SELECT
    parent.id, parent.name, parent.theme_color, parent.pinned, parent.`order`,
    CONCAT('[', TRIM(TRAILING ',' FROM GROUP_CONCAT('${"id":', child.id, ', "na
  FROM
    tags parent
  LEFT JOIN tags child ON parent.id = child.parent_id
  WHERE parent.user_id = ? AND child.user_id = ? AND parent.id = ?
  GROUP BY parent.id;',
  [user_id, user_id, tag_id]

// クエリ結果のチェック
if (!Array.isArray(selectQueryResult)) {
  throw new Error("Error: Query returned unsupported response")
}
if (selectQueryResult.length === 0) {
  res.status(404).json({ message: "Tag not found" })
  return
}

const parsedSelectQueryResult = selectQueryResult.map((row) => {
  row.tags = JSON.parse(row.tags)
  return row
})

res.status(200).json(parsedSelectQueryResult[0] as Tag)
return
```

バージョン管理の利点

- 何を作ったのか
- コードが何のために書かれたのか

が把握できる。

gitの基本はこんな感じ

複数人で作る場合



リモートリポジトリ

チーム開発で活用

- GitHub
- GitLab
- BitBucket

コミットを共有できる

オンライン上でマージもできる



実践フロー

※一例



ブランチを切る

- 機能やissue(タスク的なやつ)でログを枝分かれさせる
- 何の機能ためのコミットなのか



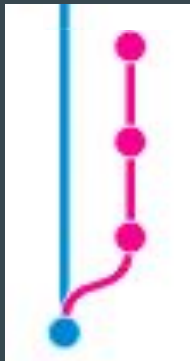
コミット

●	[doc] APIのmysql接続用envファイルについて	19 Oct 20...	tingtt	45a6f240
●	[doc] クエリ発行メソッドのUsage	19 Oct 20...	tingtt	d81dfc84
●	[add] mysql接続とクエリ発行用のメソッド	19 Oct 20...	tingtt	dd71e361
●	[npm] install 'serverless-mysql'	18 Oct 20...	tingtt	475326b6

- わかりやすい区切りでコミットしていく
- しようと思えば後からコミットをまとめて1つのコミットを作り直したりできるので細かいほうが吉
(依存関係があるコードの変更を2つ以上のコミットに分割するのはやめたほうが良い)

プッシュ

コミットをリモートリポジトリにアップロードする。



mysql_module

Commits on Oct 19, 2021

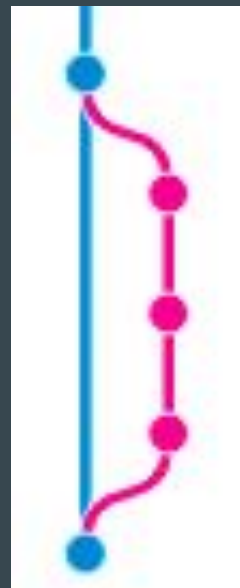
- [doc] APIのmysql接続用envファイルについて
tingtt committed 25 days ago
45a6f24
- [doc] クエリ発行メソッドのUsage
tingtt committed 25 days ago
d81dfc8
- [add] mysql接続とクエリ発行用のメソッド
tingtt committed 25 days ago
dd71e36

Commits on Oct 18, 2021

- [npm] install 'serverless-mysql'
tingtt committed 26 days ago
475326b

※ 画像はGitHubのコミットログ画面

マージ

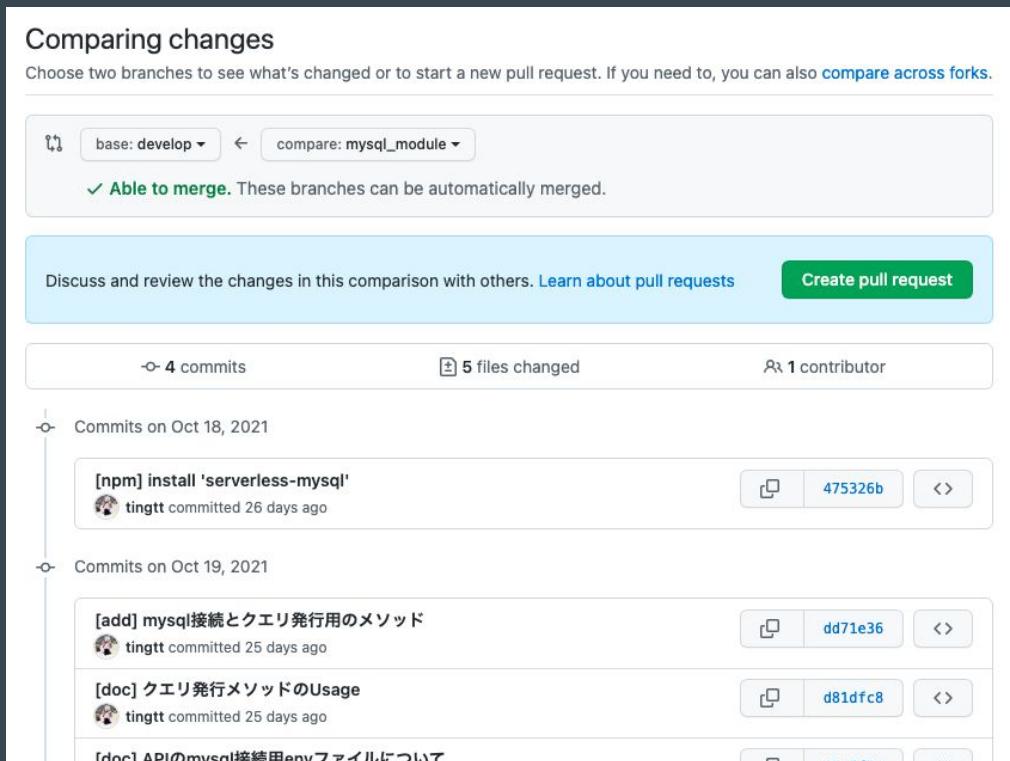


プルリクエスト / マージリクエスト

変更を反映させて良いか
GitHub, GitLab, BitBucket
上で確認する。

依存確認を含めたコードレビュー
も行う。


特定のメンバーをレビュー担当
に任せたりできる。
(アサイン)





※ 画像はGitHubのプルリクエスト発行画面


マージ


変更を反映





**Continuous integration has not been set up**
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

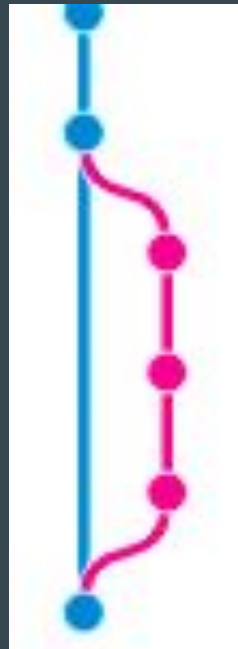
**This branch has no conflicts with the base branch**
Merging can be performed automatically.

Merge pull request  You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

 Merged

  **tingtt** merged commit **2c4f999** into **develop** on 26 Jan

Revert



ピンクブランチのコミットを青ブランチにマージした場合のログ

merge PINK into BLUE.

フェッチ / プル

フェッチ:

リモートリポジトリの変更をローカルに取得

プル:

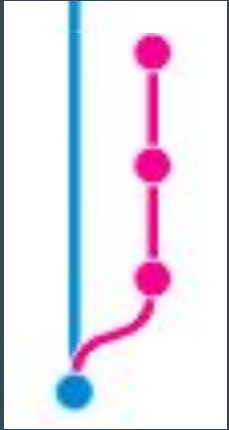
マージの一種

リモートリポジトリとローカルのリポジトリの間でマージする

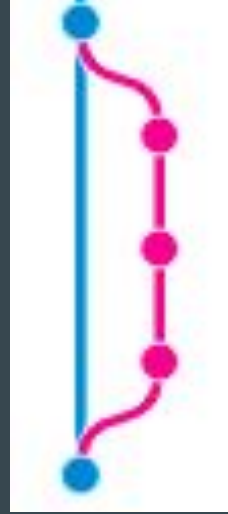
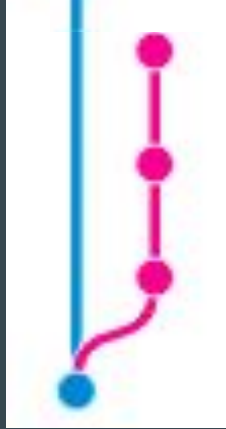
マージリクエスト / マージ

リモート

コミット



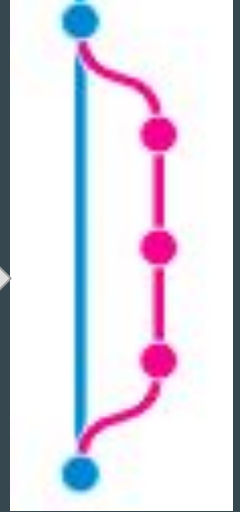
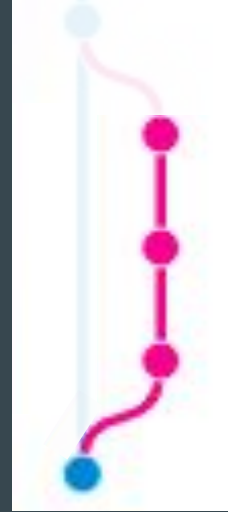
プッシュ



フェッチ



プル



ローカル

繰り返し

- ブランチを切る
- コミット × *
- プッシュ
- マージリクエスト(プルリクエスト)
 - レビュー担当者はレビュー
 - 修正が必要なら修正してコミットをプッシュ
 - マージされたらプル
- ブランチを切る

```
git checkout -b <feature_1>
```

```
git commit -m "..."
```

```
git push <remote>
```

```
git checkout main && git pull
```

```
git checkout -b <feature_2>
```


Tips




コミットメッセージ

悪い例:

そもそも雑、何のコミットなのか分からない



挨拶機能
修正
新機能
あああ



greeting feature
fix
new feature
aaaa

良い例:

- Update greeting text
- Ignore the build file
- Add greeting function
- Add `go.mod`

※ 英語でメッセージを書く場合に接続詞を使って行の詰め込むとわかりにくくなるので注意が必要

日本語と違って漢字が無いいため難しい

- 挨拶文を変更

- `.gitignore`でビルド後のファイルを除外

- 挨拶機能を作成

- `go.mod`を作成

↓ 動詞を英語で頭につけるのすすめ

- [update] 挨拶文

- [git] ビルド後のファイルをignore

- [add] 挨拶

- [add] `go.mod`

コミットをやり直したい

```
$ git commit --amend
```

一つ前のコミットを修正できる。
コミットメッセージを打ち直したり、
変更をステージしてコミットに含めることができる。

```
$ git reset --soft HEAD^
```

コミットを取り消す。
HEADの後の'^'('~'でも可)の数で戻すコミットの数を増やせる。
--softは変更を保持したまま取り消す用のオプション
--hardでコミットも変更自体も取り消せる。

一時退避

```
$ git stash
```

変更を退避できる。

-uオプションで未追跡のファイルも退避する。

```
$ git stash pop
```

退避した変更を取り出す。

複数stashがある場合は番号で指定して取り出せる。

rebase

```
$ git rebase <branch>
```

Assume the following history exists and the current branch is "topic":

```
      A---B---C topic
     /
D---E---F---G master
```

From this point, the result of either of the following commands:

```
git rebase master
git rebase master topic
```

would be:

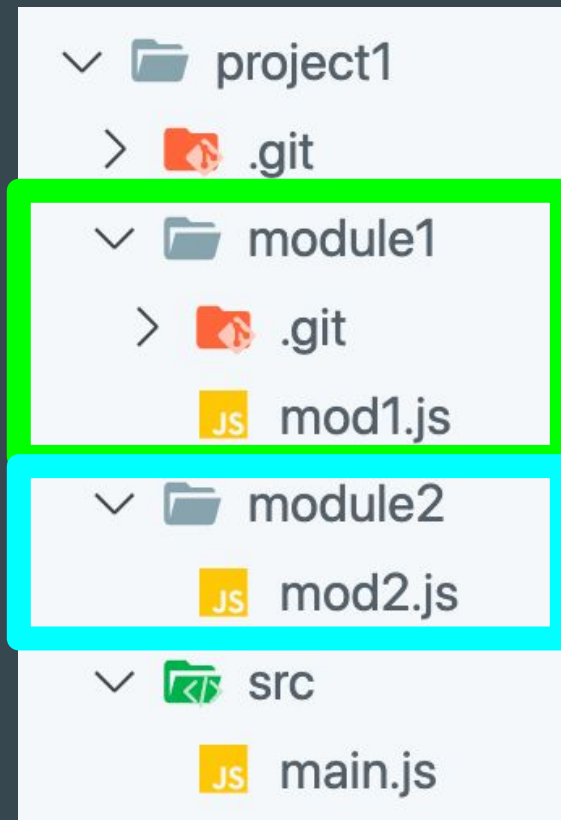
```
      A'--B'--C' topic
     /
D---E---F---G master
```

ローカルでマージ(あまりやらない)

マージリクエストなどを発行せずに、ローカルでマージすることもできる。

- 個人開発
- フォークして開発している場合
- 細かいブランチを切っていてそれらを統合してからプッシュしたい場合
(自分が切ったブランチ同士)

submoduleとsubtree



submodule :

- ``.git/`` がある。
- そのディレクトリが1つのリポジトリ

subtree :

- ``.git/`` がない。
- ディレクトリに対してなにか管理を行うわけではない。

その他小技

ブランチを切りながらブランチ切り替え

```
$ git checkout -b <new_branch>
```

コミットしていない変更を削除

```
$ git checkout .
```

ステージしてない変更をaddしてコミット

```
$ git commit -a
```