

プロジェクト演習 テーマD

第7回

担当：CS学部 講師 伏見卓恭
連絡先：fushimity@edu.teu.ac.jp

授業の流れ

- 第1回: 実験環境の構築 / Python, Gitの復習 / CUIゲームの開発
- 第2回: Tkinterの基礎
- 第3回: TkinterによるGUIゲーム開発
- 第4回: PyGameの基礎
- 第5回: PyGameによるゲーム開発
- 第6回: ゲーム開発演習
- 第7回: 成果発表

本日のお品書き

【先週】

- 3限：実装するゲームの相談
- 45限：個別にゲーム実装（残りは宿題）

【本日】

- 3限：グループ内討論・グループ成果物の作成
- 4限：3グループ合同発表会
- 5限：発表会でのコメントに基づき修正

←おおよそ16時まで

←おおよそ17時30分まで

←おおよそ18時45分まで

グループ内討論時の注意点

- メンバー全員のコードを見比べ、グループとしての成果物を作成する
- メンバーのコードをマージする際、誰のコードかわかるようにコード内コメントとREADMEに名前を明記すること
- グループとしての成果物に対するREADMEを作成する
- README作成時の分担者の名前を入れること
- 発表の準備をする（役割分担 / 発表時間 / 発表内容 / 質疑応答）
- 発表者には加点する（1人でも複数名でもOK）

メンバーのコードの共有・比較(1/3)

- まずは普段通り、グループ内メンバーのリモートリポジトリを参照し、コードとREADMEの最終版について説明し合う
 - 必要に応じてゲームのデモプレイをする
 - Issueによるコメントは不要
 - グループの成果物として中核をなすコードを決定する
 - 中核をなすコードの作成者を代表者とする
 - その他のコードに対して、中核をなすコードにマージできそうな部分を検討する
- ※グループとしての成果物に対する貢献度が決まる
(中核コード：2 / その他マージコード：1 / マージされない：0)

メンバーのコードの共有・比較(2/3)

代表者

1. 代表者のアカウントに公開リポジトリ「ProjExD_pub」を作成する
2. 新しいフォルダ「ProjExD_pub」を作成する
3. ローカルリポジトリを初期化する
`git init`
4. ブランチ名をmainに変更する
`git branch -m main`
5. 公開リポジトリをgit(ローカル)に登録する
`git remote add public https://github.com/<代表者アカウント名>/ProjExD_pub.git`
6. 最終版のコードとREADMEを「ProjExD_pub」フォルダにコピーする(画像など必要なものすべて)
7. ステージング, コミットする
`git add *`
`git commit -m "名前 最終版"`
8. 公開リポジトリのmainブランチにプッシュする
`git push public main`

その他

2. 新しいフォルダ「ProjExD_pub」を作成する
3. ローカルリポジトリを初期化する
`git init`
4. ブランチ名をmainに変更する
`git branch -m main`
5. 公開リポジトリをgit(ローカル)に登録する
`git remote add public https://github.com/<代表者アカウント名>/ProjExD_pub.git`

メンバーのコードの共有・比較(3/3)

その他

9. 公開リポジトリのmainブランチをプルする

```
git pull public main
```

10. 学籍番号ブランチを作成し, スイッチする

```
git branch 学籍番号  
git switch 学籍番号
```

11. プルした中核をなすコードにマージできそうな部分を追記する (コード内に名前をコメント表記)

12. ステージング, コミットする

```
git add *  
git commit -m "名前 追加1"
```

13. 公開リポジトリの学籍番号ブランチにプッシュする

```
git push public 学籍番号
```

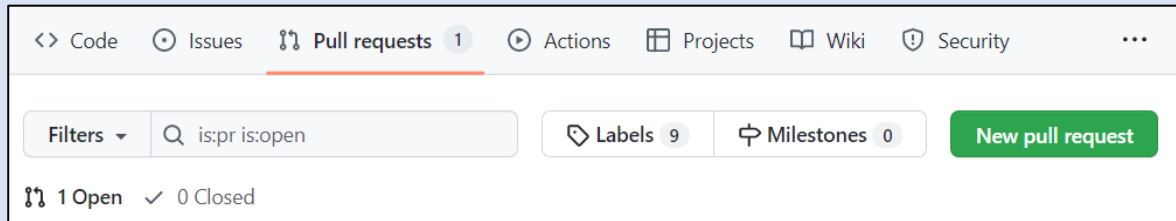
失敗する場合は, 「git push public ローカルブランチ名:リモートブランチ名」

14. Compare & pull request をクリックし, 追加した差分を代表者に伝える
→ プルリクされた内容を精査し, グループとしての最終成果物として採用するか検討

メンバーのコードのマージ(1/2)

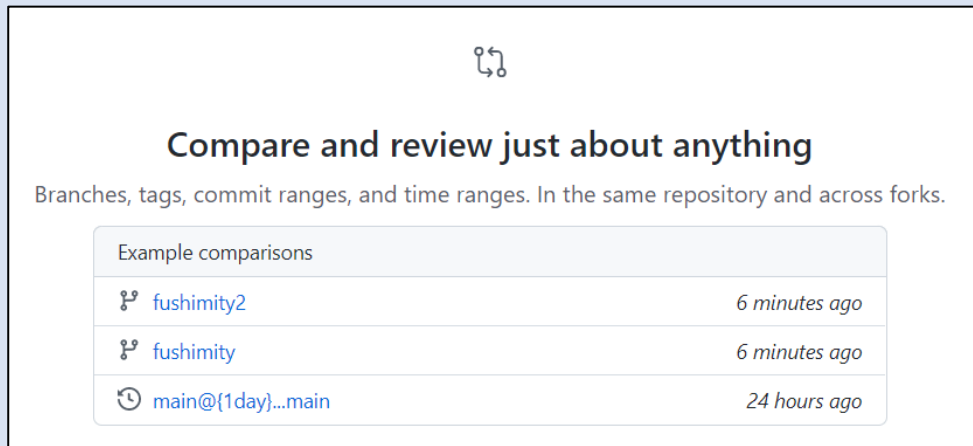
その他

1. New pull request






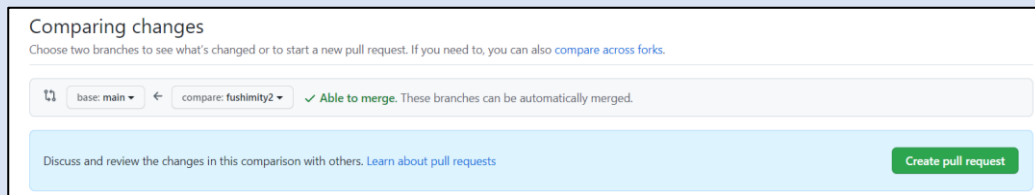
The screenshot shows the top navigation bar of a GitHub repository. The 'Pull requests' tab is selected and highlighted with a red underline, showing a count of 1. Other tabs include Code, Issues, Actions, Projects, Wiki, and Security. Below the navigation bar, there is a search bar with the text 'is:pr is:open', a 'Filters' dropdown, and buttons for 'Labels 9' and 'Milestones 0'. A green button labeled 'New pull request' is on the right. At the bottom of this section, it says '1 Open' and '0 Closed'.

2. Compare and review just about anything で自身の学籍番号のブランチを選択する



The screenshot shows the 'Compare and review just about anything' page on GitHub. It features a title 'Compare and review just about anything' and a subtitle 'Branches, tags, commit ranges, and time ranges. In the same repository and across forks.' Below this is a section titled 'Example comparisons' with a table of comparisons:

Example comparisons	
 fushimity2	6 minutes ago
 fushimity	6 minutes ago
 main@{1day}...main	24 hours ago

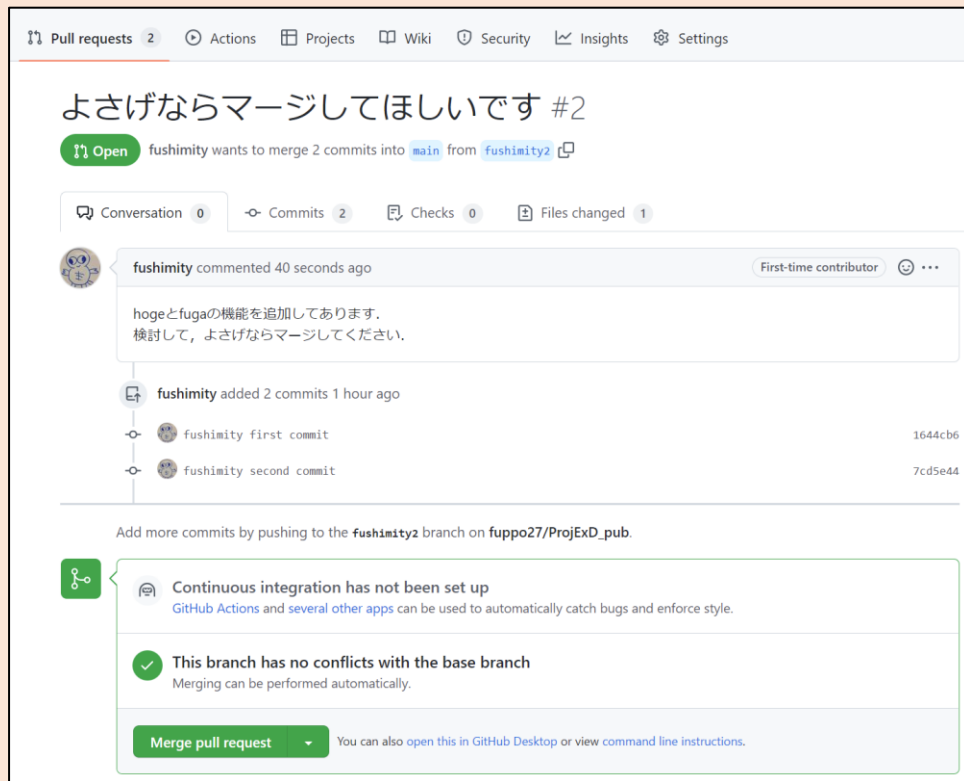


The screenshot shows the 'Comparing changes' section of the GitHub interface. It has a title 'Comparing changes' and a subtitle 'Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).' Below this is a form with two dropdown menus: 'base: main' and 'compare: fushimity2'. To the right of the 'compare' dropdown, it says 'Able to merge. These branches can be automatically merged.' At the bottom, there is a light blue box with the text 'Discuss and review the changes in this comparison with others. [Learn about pull requests](#)' and a green button labeled 'Create pull request'.

メンバーのコードのマージ(2/2)

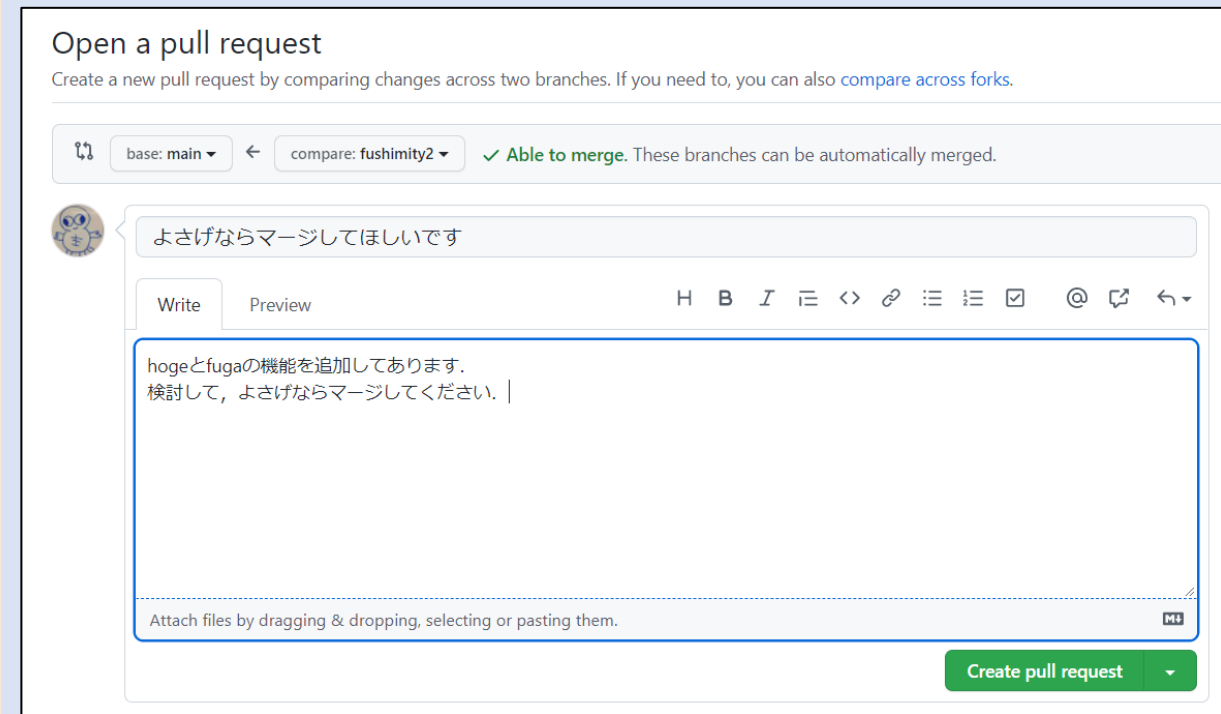
代表者

4. Merge pull request → Confirm merge




その他

3. コメントを書く → Create pull request



チェック項目：グループ成果物(5点満点)

0: 不可
1: 可
2: 優

- コードの一貫性 [0 or 1]
 - グループメンバーのコードをマージした場合
 - 複数サイトの記事を参考にした場合
- グループ成果物（コード）に貢献しているか [0 -- 2]
 - 中核をなすコード：2
 - その他マージされたコード：1
 - マージされていない：0
- README作成に貢献しているか [0 or 1]
- コメントに適切に対応しているか [0 or 1]  ← 発表会后

提出物：グループ成果物

- ファイル名：C0B21XXX_kadai07.pdf

- 内容：

1. [グループ成果物]グループとしてのコードとその説明を書いた**README**

2. [Issue]合同発表会でもらったコメント一覧（詳細は不要）

3. [最終成果物]合同発表会でのコメントを受けて修正したコード
（1. との差分を表示すること）

← 発表会后

上記の順番で全てのPDFを1つにマージしてください

グループ成果物であるが、提出物は各自で作成・提出すること

合同発表会時の注意点

- 3グループ合同で行う
- グループ成果物について説明する（最大15分）
 - 発表者は1名でも複数名でもよい
 - 発表者（一言など説明など点数稼ぎの発表は除く）には加点する
- ゲームの遊び方のデモンストレーションをする
- 他グループの人は、
 - 口頭とIssueでコメントを行う（最大15分）
 - コメント者（点数稼ぎの意味のないコメントは除く）には加点する
※加点上限あり

合同発表会の流れ

- 座席を移動する
- 担当TASAの進行に従い、グループごとに発表する
 - ZOOM-BORで画面共有する
 - 公開リポジトリのURLを宣言する
 - 他グループ者は、公開リポジトリにアクセスする
 - ゲームの説明、デモなどを行う（発表者は学籍番号、名前を宣言する）
 - コメントを受け付ける（コメント者は学籍番号、名前を宣言する）

チェック項目：プレゼン(10点満点)

※提出物はない。以下の項目を担当TASAがチェックする。

- 発表者には加点する [0 or 3]

- 発表時間は適切か [0 or 1]

- 発表内容は適切か [0 -- 2]

- 質疑応答は適切か [0 -- 2]

- 役割分担は適切か [0 -- 2]

- 発表・質疑応答・デモなど

} 発表者以外のメンバーにも、同じ点数を加算する

コメントを受けて、コードを修正する

1. まず、代表者のリポジトリにあるグループとしての成果物をローカルのmainブランチにプルする

```
git pull public main
```

2. コメントIssueごとに、対応担当者を決める

3. コメントIssueごとに、ローカルブランチを作る

```
git branch comment4 → git switch comment4
```

← Issueコメント#4の例

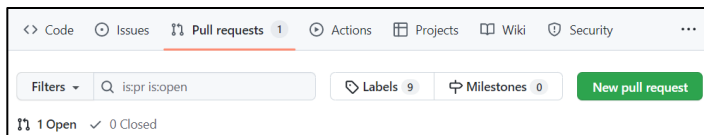
4. ブランチでコードを修正し、コミットする (Issue番号必須)

```
git add hoge.py → git commit -m "コメント対応 #4"
```

5. 公開リポジトリのブランチにプッシュする

```
git push public comment4
```

6. プルリクする



代表者およびメンバー全員で
マージしてよいか検討して、
OKなら代表者がmainにマージする

提出物：グループ成果物

- ファイル名：C0B21XXX_kadai07.pdf

- 内容：

1. [グループ成果物]グループとしてのコードとその説明を書いたREADME

2. [Issue]合同発表会でもらったコメント一覧（詳細は不要）

3. [最終成果物]合同発表会でのコメントを受けて修正したコード
（1. との差分を表示すること）

← 発表会后

上記の順番で全てのPDFを1つにマージしてください

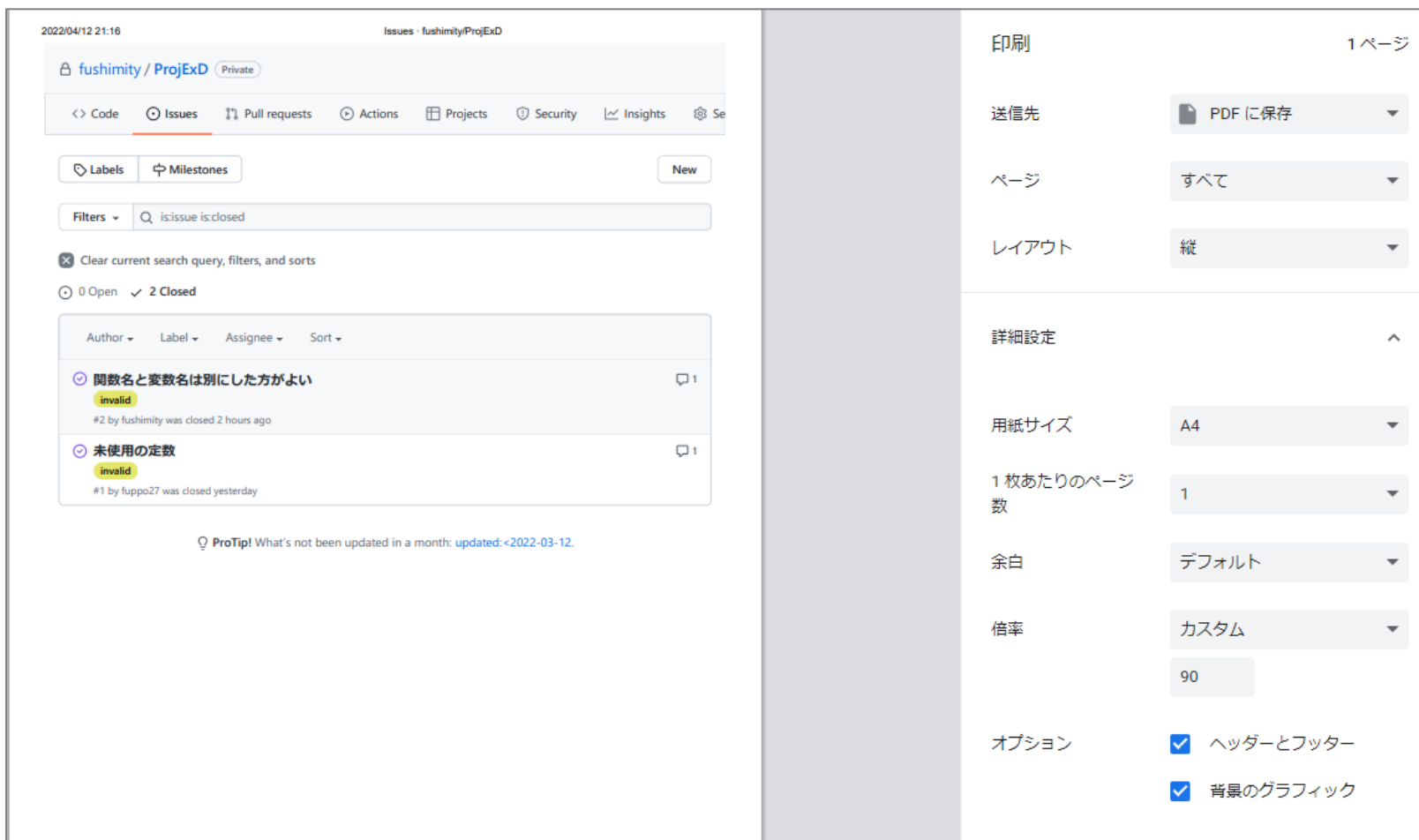
グループ成果物であるが、提出物は各自で作成・提出すること

提出，チェックの流れ

1. 受講生：提出物ができたらMoodleにアップロードする
2. 受講生：課題チェック依頼のスプレッドシートにて，
担当TASAの列に学籍番号を入力する
3. TASA：Moodleにアップされた課題をチェックする
4. TASA：チェックが済んだら，
 - Moodleに点数を入力する
 - スプレッドシートの学籍番号の色を赤くする【チェック完了の合図】
5. 受講生：自分の学籍番号の色が変わったら，帰る

【再掲】 ChromeでPDFとして保存する方法

1. 該当ページを表示させた状態で「Ctrl+P」
2. 以下のように設定し、「保存」をクリックする



←送信先：PDFに保存

←ページ：すべて

←レイアウト：縦

←用紙サイズ：A4

←余白：デフォルト

←倍率：90

←両方チェック

【再掲】 各PDFを単一ファイルにする方法

1. ChromeでPDFとして保存する
 2. 以下のURLから各PDFをマージする
 3. ファイル名を「C0B21XXX_kadai07.pdf」として保存する
- オンラインでPDFをマージするサービスの例：
 - https://www.ilovepdf.com/ja/merge_pdf
 - <https://chrome.google.com/webstore/detail/merge-pdf/ehbfcoenegfhpnnmkoaimmmlhikfccli/related?hl=ja>