

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

2014-09-22

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業) >

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第 1 章 ガイダンス

1 自己紹介

2 授業の全体像

3 授業の方法

4 モダンなソフトウェア開発

5 <演習課題（準備作業）>

自己紹介

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

名前

- 中鉢欣秀（ちゅうばちよしひで）

出身地

- 宮城県仙台市

肩書

- 産業技術大学院大学産業技術研究科
情報アーキテクチャ専攻准教授

連絡先

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業) >

E-Mail yc@aiit...

Facebook [ychubachi](#)

Twitter [ychubachi](#) (あんまり使ってない)

Skype [ychubachi](#) (あんまり使ってない)

学歴

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

| | | |
|--------|------|---|
| 1991 年 | 4 月 | 慶應義塾大学環境情報学部入学 |
| 1995 年 | 10 月 | 同大大学院政策・メディア研究科 修士課程入学 |
| 1997 年 | 10 月 | 同大大学院政策・メディア研究科 後期博士課程入学 |
| 2004 年 | 10 月 | 同大大学院政策・メディア研究科 後期博士課程卒業 学位：博士（政策・メディア） |

職歴

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

- | | | |
|--------|------|---|
| 1997 年 | 10 月 | 合資会社ニューメリック設立 社長就任 |
| 2005 年 | 4 月 | 独立行政法人科学技術振興機構 PD 級研究員 (長岡技術科学大学) |
| 2006 年 | 4 月 | 産業技術大学院大学産業技術研究科 情報アーキテクチャ専攻准教授 |

起業経験

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

社名

- 合資会社ニューメリック

設立

- 1997 年

資本金

- 18 万円

起業の背景

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

設立当時の状況

- Windows 95 が普及（初期状態でインターネットは使えなかった）
- 後輩のやっていたベンチャーの仕事を手伝って面白かった

会社設立の理由

- 「やってみたかった」 から
- 少しプログラムがかければ仕事はいくらでもあった
- 後輩にそそのかされた・笑

起業から学んだこと

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

- 実プロジェクトの経験
- 使える技術
- お金は簡単には儲からない

教育における関心事

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

情報技術産業の変化

- 情報技術のマーケットが変化
- ユーザ・ベンダ型モデルの終焉

モダンなソフトウェア開発者

- 新しいサービスの企画から、ソフトウェアの実装まで何でもこなせる開発者
- このような人材の育成方法

1 自己紹介

2 授業の全体像

3 授業の方法

4 モダンなソフトウェア開発

5 <演習課題（準備作業）>

学習目標と目的

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

目標

- ビジネスアプリケーションを構築するための基礎力
- 分散型 PBL を実施する上で必要となる知識やツールの使い方
- これら活用するための自己組織的なチームワーク

目的

- 分散ソフトウェア開発のための道具を学ぶ
 - 開発環境 (Ruby), VCS とリモトリポジトリ (GitHub)
 - テスト自動化, 継続的インテグレーション, PaaS

前提知識と到達目標

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

前提とする知識

- 情報系の学部レベルで基礎的な知識を持っていること

最低到達目標

- 授業で取り上げる各種ツールの基本的な使い方を身につける

上位到達目標

- 授業で取り上げる各種ツールの高度な使い方に習熟する。

授業の形態

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業) >

対面授業

- 担当教員による講義・演習

個人演習

- 個人によるソフトウェア開発

グループ演習

- グループによるソフトウェア開発

1 自己紹介

2 授業の全体像

3 授業の方法

4 モダンなソフトウェア開発

5 <演習課題（準備作業）>

講義・演習・課題

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

講義

- ツールの説明
- ツールの使い方

演習

- 個人でツールを使えるようになる
- グループでツールを使えるようになる

成績評価

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

課題

- 個人でソフトウェアを作る
- グループでソフトウェアを作る

評価の方法

- 課題提出と実技試験

評価の観点

- 分散 PBL で役に立つ知識が習得できたかどうか

1 自己紹介

2 授業の全体像

3 授業の方法

4 モダンなソフトウェア開発

5 <演習課題（準備作業）>

ソフトウェア開発のための方法・言語・道具

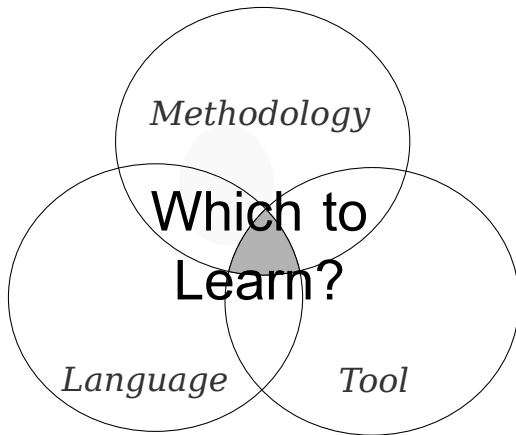


Figure: The Framework-Language-Tool framework.

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業)>

授業で取り上げる範囲

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

取り上げること

- 方法を支えるための道具
- 良い道具には設計概念として方法論が組み込まれている
- 道具はプログラミング言語を問わない

取り扱わないこと

- 方法論そのものについてはアジャイル開発特論で学ぶ
- 言語の備えるエコシステムについては必要な範囲で学ぶ

Scrum するための道具

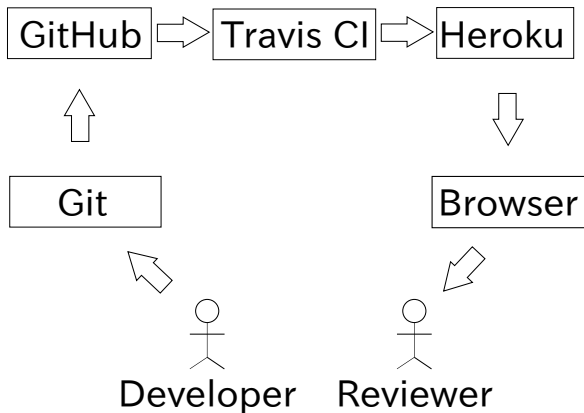


Figure: The modern tools for Scrum developments.

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業)>

モダンな開発環境の全体像

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業) >

仮想化技術 (Virtualization)

- Windows や Mac で Linux 上での Web アプリケーション開発を学ぶことができる
- Heroku や Travis CI 等のクラウドでの実行や検査環境として用いられている

ソーシャルコーディング (Social Coding)

- Linux のソースコードの VCS として用いられている Git を学ぶ
- Git は GitHub と連携することで OSS 型のチーム開発ができる

1 自己紹介

2 授業の全体像

3 授業の方法

4 モダンなソフトウェア開発

5 ＜演習課題（準備作業）＞

クラウドのアカウント作成

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業) >

GitHub

- [Join GitHub · GitHub]

Heroku

- [Heroku - Sign up]

Travis CI

- [Travis CI]
 - Travis CI は、GitHub のアカウントでログインできる

enPiT 仮想化環境のアップデート

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業) >

作業内容

- enPiT 仮想化環境（vagrant の box）を更新しておく

コマンド

```
cd ~/enpit  
vagrant destroy  
vagrant box update
```

Port Forward の設定

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業) >

説明

- Guest OS で実行するサーバに、Host OS から Web ブラウザでアクセスできるようにしておく
- 任意のエディタで Vagrantfile を変更

変更前

```
# config.vm.network "forwarded_port", guest: 80, host: 8080
```

変更後

```
config.vm.network "forwarded_port", guest: 3000, host: 3000  
config.vm.network "forwarded_port", guest: 4567, host: 4567
```

enPiT 仮想化環境にログイン

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業) >

作業内容

- 前の操作に引き続き、仮想化環境に SSH 接続する

コマンド

```
vagrant up  
vagrant ssh
```

github-connect スクリプト

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフ
トウェア開発

<演習課題
(準備作業)>

URL

- [github-connect.sh]

git config を代行

- GitHub にログインし，名前と email を読み込んで git に設定

SSH の鍵生成と登録

- SSH 鍵を作成し，公開鍵を GitHub に登録してくれる

github-connect.sh の実行

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフト
ウェア開発

<演習課題
(準備作業) >

作業内容

- スクリプトを起動し、設定を行う
- GitHub のログイン名とパスワードを聞かれるので、入力する
- rsa key pair のパスフレーズは入力しなくて構わない

コマンド

```
github-connect.sh
```

Git と GitHub の設定確認

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

自己紹介

授業の全体像

授業の方法

モダンなソフトウェア開発

<演習課題
(準備作業) >

Git の設定確認

```
git config --list
```

GitHub の設定確認

- ブラウザで GitHub の SSH Key ページを開く

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第2章 ローカルリポジトリの操作

6 ローカルリポジトリ

7 リモートリポジトリ

8 Git と GitHub の基本操作

9 <演習課題>

Git のローカルリポジトリの作成

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

ローカルリポジトリ

- ソースコードや各種のファイルを保存し，開発に利用する
- 「my_enpit」というディレクトリを作成し，初期化する

コマンド

```
mkdir ~/my_enpit  
cd ~/my_enpit  
git init
```

Git の設定ディレクトリ

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

隠しフォルダ「.git」

- Git ソースコードの履歴情報や、各種の設定を Git が保存するディレクトリ
- このフォルダは通常、Git を経由しないで変更することはない

確認方法

```
ls -a  
find .git
```

6 ローカルリポジトリ

7 リモートリポジトリ

8 Git と GitHub の基本操作

9 <演習課題>

Hub コマンド

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

ローカルリ
ポジトリ

リモートリ
ポジトリ

Git と GitHub
の基本操作

<演習課題>

enPiT 環境の Hub コマンド

- `github/hub`

Git への GitHub 操作機能追加

- 通常の Git の機能に加えて、GitHub 用のコマンドが利用できる
- エイリアス設定しており、コマンド名は「git」のまま

確認方法

```
git version  
alias git
```

Hub コマンドによるリモートリポジトリの作成

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

作業内容

- コマンドライン操作で, GitHub にリポジトリを作成する
- Hub コマンドの機能である `git create` を利用
- 初回既動時にはパスワードが聞かれる

コマンド

```
git create
```

リポジトリの確認方法

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

確認方法

- Web ブラウザで GitHub を開き、「my_enpit 」ができていることを確認

コマンドラインで確認

```
git remote -vv
```

6 ローカルリポジトリ

7 リモートリポジトリ

8 Git と GitHub の基本操作

9 <演習課題>

Git の操作方法

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

マニュアル等

- [Git - Documentation](#)

commit ログの書き方

- [Writing good commit messages · erlang/otp Wiki](#)

ステータスの確認

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

ローカルリポジトリ

リモートリポジトリ

Git と GitHub
の基本操作

<演習課題>

リポジトリの状態を確認する

- `git status` は、頻繁に利用するコマンド
- リポジトリの状態を確認することができる
- この表示の読み方を理解することが重要

コマンド

```
git status
```

ファイルの追加とステータスの確認

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

ローカルリポジトリ

リモートリポジトリ

Git と GitHub
の基本操作

<演習課題>

作業内容

- テキストエディタで README.md を作成
- ステータスの変化を見る

コマンド

```
emacs README.md  
git status
```

Add/Commit の方法

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

ローカルリ
ポジトリ

リモートリ
ポジトリ

Git と GitHub
の基本操作

<演習課題>

ステージングエリアを利用する場合

- `git add README.mb`
- `git commit -m 'First commit'`

ステージングエリアを省略する場合

- `git commit -a -m 'First commit'`
 - トラックされていないファイルは commit しないので注意

Log の閲覧

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

コミットログ

- ソースコードに加えた変更の履歴を，commit を単位として閲覧できる

コマンド

```
git log
```

Push の方法

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

push とは？

- ローカルで作成した commit を，リモートのリポジトリにアップロードすること
- origin とは，リモートのリポジトリの内部的な名前
- upstream とは，ブランチ（後述）が紐づいているリポジトリのこと
- 最初にそのブランチを push するときは，`--set-upstream` オプションを指定

コマンド

```
git push --set-upstream origin master
```

コミットのログを詳細に書く方法

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

ローカルリポジトリ

リモートリポジトリ

Git と GitHub
の基本操作

<演習課題>

エディタを使ったログの記述

- コミットのログや、Pull Request の記述を、より詳しく書くことができる
- `commit` や `pull_request` から `-m` オプションを外すと、エディタが立ち上がる
 - エディタは `emacs` を起動するようになっている
 - `C-x C-s` で保存、`C-x C-c` で終了

コマンド

```
git commit
git pull_request
```

6 ローカルリポジトリ

7 リモートリポジトリ

8 Git と GitHub の基本操作

9 <演習課題>

Init/Status/Add の練習

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

演習課題

- 1 解説した手順に従い、my_enpit リポジトリを作成
- 2 git status コマンドを実行
- 3 README.md ファイルを作成しなさい
- 4 git status コマンドを実行し、変化を見なさい
- 5 commit しなさい。ログを必ず書くこと
- 6 git status コマンドを実行し、変化を見なさい

Commit/Log/Push の練習

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

ローカルリポジトリ

リモートリポジトリ

Git と GitHub
の基本操作

<演習課題>

演習課題

- 1 README.md を修正して commit してください
- 2 新しいファイルを作成して commit してください
- 3 作業が完了したら，push してください（--set-upstream が必要）
- 4 コミットが push されていることを Web ブラウザで確認してください
- 5 作成したファイルを削除して commit して push してください
- 6 エディタを使って，詳細なログを書きなさい
- 7 その他，自由に commit の作業を試しなさい

ここまでの課題の提出

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ローカルリポ
ジトリ

リモートリポ
ジトリ

Git と GitHub
の基本操作

<演習課題>

提出物

- 下記のことを提出してください
 - GitHub と Heroku アカウント
 - 作成した my_enpit リポジトリの URL

提出先

- [enPiT 演習アカウント (2014)]

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第3章 ブランチとリモートリポジトリ

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

10 GitHub Flow

11 ブランチの操作

12 リモートのブランチ

13 Pull Request

14 <演習課題>

GitHub Flow (1)

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

GitHub Flow

ブランチ
の操作

リモートのブランチ

Pull Request

<演習課題>

- 1 思い立ったらブランチ作成
 - 新しい機能追加や、アイデアを試す
- 2 ブランチにコミットを追加
 - 変更点をコミットとして作成
 - コミットのログは、他人が読んでわかるように書く
- 3 Pull Request を開く
 - コミットについて、意見交換ができる
 - 作業途中で Pull Request を出しても構わない

GitHub Flow (2)

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

GitHub Flow

ブランチ
の操作

リモートのブランチ

Pull Request

<演習課題>

1 議論とレビュー

- レビューをしたり，質疑応答をしたりする

2 マージしてディプロイ

- master ブランチにマージする（自動でディプロイ）
- マージの前にテストしたいときは，ローカルで試す

参考文献

- [Understanding the GitHub Flow](#) ・ [GitHub Guides](#)

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

10 GitHub Flow

11 ブランチの操作

12 リモートのブランチ

13 Pull Request

14 <演習課題>

branch の作成

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

ブランチとは？

- リポジトリには master ブランチがある
- 新しい作業を行う場合、必ず branch を切る

コマンド

```
git branch new_branch  
git branch -vv
```

branch の checkout

ビジネスアプリ
ケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

branch を切り替える

- checkout してブランチを切り替える
- ブランチを commit することができる
- 切り替える前に、ブランチでの作業は commit しておく（stash も可）

コマンド

```
git checkout new_branch
```

< 編集作業 >

```
git commit -a -m 'Create a new branch'
```

他の branch を merge する

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

merge とは

- ブランチで作業した内容（commit）を、他のブランチに統合すること
- new_branch での作業を master に統合する場合、最初に master を checkout する

コマンド操作

```
git checkout master  
git merge new_branch
```

Conflict（競合）とその解消

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

Conflict とは

- branch で行う作業がかち合った場合、発生する
- merge する際、conflict が生じた場合、エラーになる

解消方法

- エディタ等で編集を行い、解消する

参考文献

- Resolving a merge conflict from the command line ・GitHub Help

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

10 GitHub Flow

11 ブランチの操作

12 リモートのブランチ

13 Pull Request

14 <演習課題>

Branch の Push

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

リモートへの Push

- Branch を GitHub に Push することができる
- master ブランチを Push した際と同様, upstream を指定する
- Push できたかどうかを Web ブラウザで確認する

コマンド

```
git push --set-upstream origin new_branch
```

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

10 GitHub Flow

11 ブランチの操作

12 リモートのブランチ

13 Pull Request

14 <演習課題>

Pull Request の作成

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

Pull Request とは？

- push した branch での作業の統合（merge）を依頼する
- hub コマンドの pull-request で発行できる

コマンド

```
git pull-request -m 'Update a new branch'
```


Pull Request の merge

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

GitHub Flow

ブランチ
の操作

リモートのブランチ

Pull Request

<演習課題>

Pull Request をレビューする

- Web ブラウザで Pull Request を確認する

ブラウザで merge

- 問題なければ merge ボタンを押す

コマンドラインで merge する場合

```
git merge pull_request_URL
```

Branch の Pull

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

Branch を Pull するとは

- リモートで行われた変更を適用すること
- 内部的には fetch でダウンロードしてから merge する

コマンド

```
git checkout master  
git pull
```

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

10 GitHub Flow

11 ブランチの操作

12 リモートのブランチ

13 Pull Request

14 <演習課題>

branch の操作（ローカル）

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

演習課題

- 1 my_enpit リポジトリでブランチを作成しなさい（new_branch）
- 2 checkout で new_branch に移動する
- 3 ファイルを編集し commit する
- 4 master ブランチに移動してファイルの内容が「編集されていないこと」を確認しなさい
- 5 merge して、変更を適用しなさい

競合の発生と解消

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

GitHub Flow

ブランチ
の操作

リモートのブランチ

Pull Request

<演習課題>

演習課題

- 1 new_branch でファイルを編集して、commit する
- 2 master に移動し、ファイルの同じ箇所を編集して、commit する
- 3 master に new_branch を merge して、コンフリクトを発生させる
- 4 エディタで競合箇所を修正して commit する

リモートの branch の操作

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

GitHub Flow

ブランチ
の操作

リモートのブ
ランチ

Pull Request

<演習課題>

演習課題

- 1 新しいブランチを作成して、remote に push する
- 2 Pull Request を送る
- 3 ブラウザで、Pull Request をマージする
- 4 master ブランチに移動して、pull することで、更新する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第 4 章 GitHub を使った協同作業

15 他の人の開発状況を見る

16 開発に参加する

17 GitHub の他の機能

18 <演習課題>

リモートのリポジトリを Clone

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

他の人の開発状況を見る

開発に参加する

GitHub の他の機能

<演習課題>

Clone とは

- GitHub で公開されているリポジトリはだれでも複製（clone）できる
- ソースコードはローカルにコピーされ、閲覧やコンパイルなどができるようになる
- アクセス権限がない場合は、push できない

コマンド

```
git clone octocat/Spoon-Knife
```

Pull Request をチェックアウト

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

Pull Request のチェックアウト

- 誰かが作成した Pull Request の内容を，ブランチとしてローカルにコピーする
- 試しに動作させたり，コードをチェックするときなどに利用

コマンド

```
git checkout https://github.com/octocat/Spoon-Knife/pull/3166
```

15 他の人の開発状況を見る

16 開発に参加する

17 GitHub の他の機能

18 <演習課題>

オリジナルのリポジトリを Fork する

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

Fork とは

- Clone したリポジトリを，自分のアカウントが所持するリポジトリとして GitHub 上で複製する
- remote の値は，オリジナルのリポジトリが origin ，自分のリポジトリは自分の GitHub ユーザ名になる

コマンド

```
git fork  
git remote -vv
```

ブランチを作成し自分のリポジトリに push

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

オリジナルの改変等

- 新しい機能追加等を行う場合、ブランチを作成する
- ブランチは、自分のリポジトリに push する

コマンド

```
git branch my_branch
git checkout my_branch
<編集>
git commit -a -m 'Update'
git push -u ychubachi my_branch
```

Fork した元に Pull Request を送る

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

他の人の開発状況を見る

開発に参加する

GitHub の他の機能

<演習課題>

コードのレビューやマージを依頼する

- 新しい機能ができたら、オリジナルに Pull Request を送り、レビューやマージをしてもらう

コマンド

```
git pull_request -m 'Pull Request'
```

15 他の人の開発状況を見る

16 開発に参加する

17 GitHub の他の機能

18 <演習課題>

Issue/Wiki

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

Issue

- 課題管理 (ITS: Issue Tracking System)
- コミットのメッセージで close できる
 - Closing issues via commit messages ・GitHub Help

Wiki

- GitHub のリポジトリに Wiki を作る
 - About GitHub Wikis ・GitHub Help

GitHub

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

他の人の開発状況を見る

開発に参加する

GitHub の他の機能

<演習課題>

GitHub Pages

- 特殊なブランチを作成すると、Web ページが構築できる
 - GitHub Pages

Git blame

- だれがどの作業をしたかわかる（誰がバグを仕込んだのかも）
 - Using git blame to trace changes in a file •GitHub Help

15 他の人の開発状況を見る

16 開発に参加する

17 GitHub の他の機能

18 <演習課題>

our_enpit にファイルを追加する

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

演習課題

- 1 ychubich/our_enpit を clone して fork する
- 2 新しいブランチを作成し，新規にファイルを追加する
 - 内容は任意（自己紹介など）
 - Markdown で書いてください（拡張子は.md）
- 3 コミットを作成し，pull request を送信する
- 4 教員がマージ作業を行います

既存のファイルを変更する

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

演習課題

- 1 README.md を改変して、pull request を送信する
- 2 GitHub の Pull Request 一覧を確認する
- 3 おそらくコンフリクトが発生するので、GitHub の指示に従い競合を解消する

隣の人との協同作業

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

演習課題

- 1 新しくリポジトリを作成する（名称は任意）
- 2 互いに、隣の席の人にリポジトリ名を教え、fork してもらい Pull Request を送ってもらう
- 3 マージしてあげる
- 4 2～3 を繰り返し、協同作業を行ってみよう

Issue/Wiki の利用

ビジネスア
プリケーション
演習

中鉢欣秀・上
田隆一

他の人の開発
状況を見る

開発に参
加する

GitHub の他
の機能

<演習課題>

演習課題

- GitHub の Issue の機能を使ってみなさい
- commit のログで Issue をクローズさせてみなさい
- Wiki を作ってください

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第 5 章 Sinatra アプリの開発

19 Sinatra アプリケーションの作成

20 Heroku でアプリケーションを動かす

21 <演習課題>

Sinatra を使った簡単な Web アプリケーション

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

Sinatra とは？

- Web アプリケーションを作成する DSL
- Rails に比べ軽量で，学習曲線が緩やか

参考文献

- Sinatra

Sinatra アプリ用リポジトリを作成する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

内容

- Sinatra アプリを作成するため、新しいリポジトリを作る

コマンド

```
mkdir ~/sinatra_enpit  
cd ~/sinatra_enpit  
git init  
git create
```

Sinatra アプリを作成する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

コマンド

```
emacs hello.rb  
git add hello.rb  
git commit -m 'Create hello.rb'
```

コード: hello.rb

```
require 'sinatra'  
  
get '/' do  
  "Hello World!"  
end
```

Sinatra アプリを起動する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

起動の方法

- hello.rb を ruby で動かせば、サーバが立ち上がる
- vagrant の port forward を利用するためのオプションを追加する
 - ruby - Unable to access Sinatra app on host machine with Vagrant forwarded ports - Stack Overflow

コマンド

```
ruby hello.rb -o 0.0.0.0
```

Sinatra アプリの動作確認

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

動作確認の方法

- Host OS の Web ブラウザで, `http://localhost:4567` にアクセスする.

19 Sinatra アプリケーションの作成

20 Heroku でアプリケーションを動かす

21 <演習課題>

コマンドラインで Heroku にログインする

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーション
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

内容

- enPiT 環境には heroku コマンドをインストールしてある
- heroku コマンドを用いて, Heroku にログインできる
- 以後の作業は Heroku コマンドを利用する

コマンド

```
heroku login
```

heroku に SSH の公開鍵を設定する

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

Sinatra アプリケーション
の作成

Heroku でアプリケーション
を動かす

<演習課題>

内容

- Heroku も git のリモートリポジトリである
- ここに公開鍵でアクセスできるようにする

コマンド

```
heroku keys:add
```

確認

```
heroku keys
```


Heroku で動作できる Sinatra アプリ

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

内容

- Heroku で動作できる Sinatra アプリと設定ファイルの例 Deploying Rack-based Apps | Heroku Dev Center
- 例を見ながら、エディタを用いて、次の3つのファイルを作成する
 - `hello.rb` Ruby による Web アプリ本体（作成済み）
 - `config.ru` Web アプリサーバ（Rack）の設定
 - `Gemfile` アプリで利用するライブラリ（Gem）

コマンド

```
emacs config.ru  
emacs Gemfile
```

アプリを GitHub に push する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーション
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

内容

- Heroku で動かす前に、commit が必要
- ついでに、GitHub にコードを push しておく
 - この場合の push 先は origin master

コマンド

```
git add .  
git commit -m 'Add configuration files for Heroku'  
git push origin master
```

Heroku にアプリを作る

ビジネスアプリ
ケーション
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーション
の作成

Heroku でアプリ
ケーション
を動かす

<演習課題>

アプリを作る

- Heroku が自動生成した URL が表示されるので，メモする
- `git remote -v` で heroku という名前の remote が追加されたことが分かる
- Web ブラウザで Heroku の管理画面を開くと，アプリができていることが確認できる

コマンド

```
heroku create  
git remote -v
```

Heroku にアプリを配備する

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーション
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

配備する方法

- Heroku のリモートリポジトリに push する
- Web ブラウザでアプリの URL を開き，動作を確認する

コマンド

```
git push heroku master
```

19 Sinatra アプリケーションの作成

20 Heroku でアプリケーションを動かす

21 <演習課題>

Sinatra アプリの作成

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーシ
ョン
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

演習課題

- Sinatra アプリを作成して、Heroku で動作させなさい
- Sinatra の DSL について調べ、機能を追加しなさい
- コミットのログは詳細に記述し、どんな作業を行ったかが他の人にも分かるようにしなさい
- 完成したコードは GitHub にも push しなさい

Sinatra アプリの共同開発

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Sinatra アプリ
ケーション
の作成

Heroku でア
プリケーシ
ョン
を動かす

<演習課題>

演習課題

- 隣の席の人と協同で Sinatra アプリを開発しなさい
- 一方が GitHub のリポジトリを作成し、もう一人が Fork する
- 最初に、どんな機能をもたせるかを相談しなさい
 - メンバーのスキルに合わせて、できるだけ簡単なもの
 - データベースは使わない
- ブランチを作成し、Pull Request を送る

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

第 6 章 Web API を活用したサービス構築

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ご利用ガイド

楽天 API SDK

Sinatra との組
み合わせ

22 ご利用ガイド

23 楽天 API SDK

24 Sinatra との組み合わせ

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

ご利用ガイド

楽天 API SDK

Sinatra との組
み合わせ

■ 楽天ウェブサービス: ご利用ガイド

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ご利用ガイド

楽天 API SDK

Sinatra との組
み合わせ

22 ご利用ガイド

23 楽天 API SDK

24 Sinatra との組み合わせ

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ご利用ガイド

楽天 API SDK

Sinatra との組
み合わせ

■ rakuten-ws/rws-ruby-sdk

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

ご利用ガイド

楽天 API SDK

Sinatra との組
み合わせ

22 ご利用ガイド

23 楽天 API SDK

24 Sinatra との組み合わせ

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

WIP [講義] Ruby on Rails の開発

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

25 Memo

26 Ruby on Rails の全体像

27 Scaffold の作成と動作

28 RSpec によるテスト

29 Travis CI

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

■ Gemfile の therubyracer をアンコメントする

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

25 Memo

26 Ruby on Rails の全体像

27 Scaffold の作成と動作

28 RSpec によるテスト

29 Travis CI

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

25 Memo

26 Ruby on Rails の全体像

27 Scaffold の作成と動作

28 RSpec によるテスト

29 Travis CI

Heroku への deploy

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

Scaffold による Model の拡張

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

25 Memo

26 Ruby on Rails の全体像

27 Scaffold の作成と動作

28 RSpec によるテスト

29 Travis CI

RSpec の実行

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

25 Memo

26 Ruby on Rails の全体像

27 Scaffold の作成と動作

28 RSpec によるテスト

29 Travis CI

Travis CI の Web 管理画面

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

GitHub と Travis CI 連携

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

Travis 経由での Heroku への deploy

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Memo

Ruby on Rails
の全体像

Scaffold の作
成と動作

RSpec による
テスト

Travis CI

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

< 演習 > Ruby on Rails を用いた開発演習 (1)

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

< 演習 > Ruby on Rails を用いた開発演習 (2)

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

[講義] ミニプロジェクト

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

< 演習 > ミニプロジェクト演習 (1)

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

< 演習 > ミニプロジェクト演習 (2)

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

memo

30 バージョン管理の概念

シナリオ

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

バージョン管
理の概念

HTML による Web ページ

index.html を作りブラウザで開く

バージョン管理の基礎知識

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

バージョン管
理の概念

diff

patch

sha1

演習課題

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

バージョン管
理の概念

演習課題

- あなたがよく知っている「歴史上の有名人」を一人取り上げる
- その人を紹介する Web ページを作成する
- HTML を作成する（リンクや画像の埋め込みにもチャレンジ）
- git でバージョン管理
- GitHub に push する

GitHub に push

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

バージョン管
理の概念

コマンド

```
git commit -a -m 'First commit'  
git push -u origin msater
```

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DONE

chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようになる
こと

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

ビジネスアプリケーション演習

中鉢欣秀・上田隆一

産業技術大学院大学 (AIIT)

Tasks

DONE

chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようにな
ること

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 **DONE** chocolatey のインストールを kazam でキャプチャする

32 **TODO** 英語の原典を読めるようになること

33 **TODO** よくある間違い cd しないで git init するとか。

34 **TODO** OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 **TODO** アンケートを作成する

36 **TODO** .bash_profile から .bashrc を読み込む（カラー化）

DONE
chocolatey の
インストールを
kazam でキャ
プチャする

TODO 英語の
原典を読める
ようになるこ
と

TODO よくあ
る間違い cd し
ないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 DONE chocolatey のインストールを kazam でキャプチャする

32 TODO 英語の原典を読めるようになること

33 TODO よくある間違い cd しなくて git init するとか。

34 TODO OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 TODO アンケートを作成する

36 TODO .bash_profile から .bashrc を読み込む（カラー化）

DONE
chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようにな
ること

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 **DONE** chocolatey のインストールを kazam でキャプチャする

32 **TODO** 英語の原典を読めるようになること

33 **TODO** よくある間違い cd しないで git init するとか。

34 **TODO** OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 **TODO** アンケートを作成する

36 **TODO** .bash_profile から .bashrc を読み込む（カラー化）

DONE
chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようにな
ること

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 **DONE** chocolatey のインストールを kazam でキャプチャする

32 **TODO** 英語の原典を読めるようになること

33 **TODO** よくある間違い cd しないで git init するとか。

34 **TODO** OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 **TODO** アンケートを作成する

36 **TODO** .bash_profile から .bashrc を読み込む（カラー化）

DONE
chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようにな
ること

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 DONE chocolatey のインストールを kazam でキャプチャする

32 TODO 英語の原典を読めるようになること

33 TODO よくある間違い cd しないで git init するとか。

34 TODO OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 TODO アンケートを作成する

36 TODO .bash_profile から .bashrc を読み込む（カラー化）

■ 調査の目的

■ モダンなソフトウェア開発の理解度（これは 2 回やる）

- git について 90%（業務でのソフトウェア開発に利用できる）, 70%, 50%, 30%, 10%（ほとんど知らない・使ったことはない）

- PBL のために、事前学習が役に立ったか（これは PBL 後）事前学習をした人とそうでない人とで、PBL の満足感、達成感が違うか円滑に PBL をすすめることができたか

■ 方法論

あなたは BizApp 演習の内容を学習しましたか？

1 授業を履修した

1 ビデオを視聴した

1 学習していない

2 道具

DONE
chocolatey の
インストール
を kazam でキ
ャプチャする

TODO 英語の
原典を読める
ようにな
ること

TODO よくあ
る間違い cd
しないで git
init するとか。

TODO OS を
インストール
し、手順書を
参照しながら
長々とコマン
ドを打つ、と

31 DONE chocolatey のインストールを kazam でキャプチャする

32 TODO 英語の原典を読めるようになること

33 TODO よくある間違い cd しないで git init するとか。

34 TODO OS をインストールし、手順書を参照しながら長々とコマンドを打つ、ということが不要になった。

35 TODO アンケートを作成する

36 TODO .bash_profile から.bashrc を読み込む（カラー化）