

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

産業技術大学院大学 (AIIT)

第 1 章モダンなソフトウェア開発の道具達

1 連絡事項

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とは

5 ＜演習課題（準備作業）＞

連絡事項

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<演習課題
(準備作業)>

資料等の入手先

- GitHub の下記リポジトリにまとめておきます
 - [ychubachi/enpit](#)
- 資料は随時 update するので、適宜、最新版をダウンロードしてください

Twitter のハッシュタグ

- Twitter のハッシュタグは #enpit_aiit を使ってください
- まとめサイトなど作ってくれると嬉しいです
 - 昨年の例 -> enPiT BizApp AIIT ビジネスアプリケーション演習 1 日目 - Togetter まとめ

1 連絡事項

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とは

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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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 連絡事項

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とは

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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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 連絡事項

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とは

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

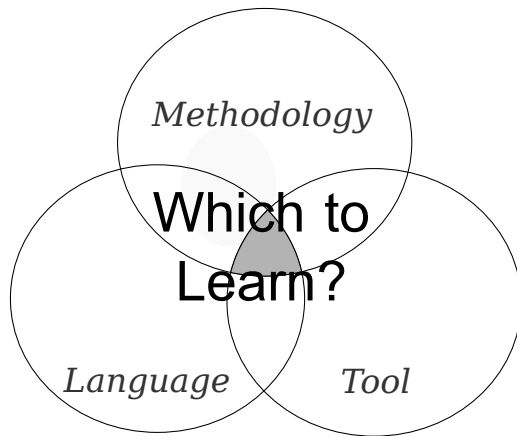


Figure: The Framework-Language-Tool framework.

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

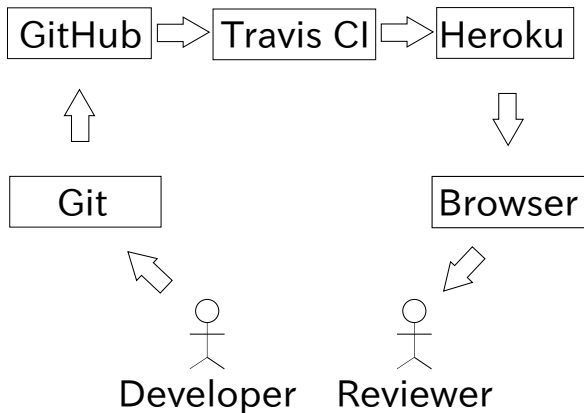


Figure: The modern tools for Scrum developments.

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

enPiT 仮想化環境

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<演習課題
(準備作業)>

インストール済みの言語と道具

- エディタ (Emacs/Vim)
- Ruby の実行環境
- GitHub, Heroku, Travis CI と連携するための各種コマンド (github-connect.sh, hub, heroku, travis)
- PostgreSQL のクライアント・サーバーと DB
- 各種設定ファイル (.bash_profile, .gemrc, .gitconfig)
- その他

仮想化環境の構築用リポジトリ (参考)

- [ychubachi/vagrant_enpit](#)

1 連絡事項

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とは

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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフトウェア開発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフトウェア開発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフ
トウェア開
発とは

<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

連絡事項

授業の全体像

授業の方法

モダンなソフトウェア開発とは

<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 Git/GitHub の基本操作

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

リモートリポジトリへの公開

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，`--setupstream` オプションを指定

コマンド

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

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
```


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 GitHub を用いた開発の流れ

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 による協同作業

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.

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>

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
```

Bundle install

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

- Gemfile の中身に基づき、必要な Gem（ライブラリ）をダウンロードする
 - Gemfile.lock というファイルができる
 - このファイルも commit の対象に含める

コマンド

```
bundle install
```

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 -u 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 Ruby on Rails アプリの開発

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

22 Ruby on Rails アプリの生成と実行

23 Controller/View の作成

24 Heroku にデプロイする

25 <演習課題>

RoR を使った 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Ruby on Rails (RoR) とは？

- Web アプリケーションを作成するためのフレームワーク

参考文献

- Ruby on Rails

Heroku で動かす方法

ビジネスアプリ
ケーション
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Getting Started

- [Getting Started with Rails 4.x on Heroku | Heroku Dev Center](#)

DB について

- Database は PostgreSQL を使用する
 - RoR 標準の sqlite は使わない

PostgreSQL に DB を作成

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

開発で利用する DB

`rails_enpit_development` 開発作業中に利用

`rails_enpit_test` テスト用に利用

`rails_enpit_production` 本番環境で利用（ローカルには作成しない）

コマンド

```
createdb rails_enpit_development
```

```
createdb rails_enpit_test
```

rails_enpit 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

内容

- rails は予め、仮想化環境にインストールしてある
- rails new コマンドを用いて、RoR アプリの雛形を作成する

コマンド

```
rails new ~/rails_enpit --database=postgresql
cd ~/rails_enpit
git init
git create
git add .
git commit -m 'Generate a new rails app'
git push -u origin master
```


Gemfile の変更

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

変更する内容

- Gemfile に Rails 内部で動作する JavaScript の実行環境を設定する
- 当該箇所のコメントを外す
- 変更を commit しておく

変更前

```
# gem 'therubyracer', platforms: :ruby
```

変更後

```
gem 'therubyracer', platforms: :ruby
```

Bundle install 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にディ
プロイする

<演習課題>

bundle install

- Gemfile を読み込み、必要な gem をインストールする
- rails new をした際にも、bundle install は実行されている
- 今回は therubyracer と、それが依存している gem でまだインストールしていないものをインストール
- インストールする先は ~/.rbenv 以下の特定のディレクトリ

コマンド

```
bundle install  
git commit -a -m 'Run bundle install'
```

Rails server の起動

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Rails server を起動

- この段階で、アプリケーションを起動できるようになっている
- Host OS の Web ブラウザで、`http://localhost:3000` にアクセスして確認
- 端末にもログが表示される
- 確認したら、端末で `Ctrl-C` を押してサーバを停止する

コマンド

```
rails server
```

22 Ruby on Rails アプリの生成と実行

23 Controller/View の作成

24 Heroku にデプロイする

25 <演習課題>

Hello World を表示する Controller

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Controller とは？

- MVC 構造でいう Controller
- HTTP のリクエストを処理し，View に引き渡す
- `rails generate controller` コマンドで作成する

コマンド

```
rails generate controller welcome
```

View 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

View とは？

- HTML 等で結果をレンダリングして表示する
- `app/views/welcome/index.html.erb` を作成する
- `erb` で作成するのが一般的で、内部で Ruby コードを動作させることができる

`index.html.erb`

```
<h2>Hello World</h2>
```

```
<p>
```

```
  The time is now: <%= Time.now %>
```

```
</p>
```

root となる route の設定

ビジネスアプリ
リケーション
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Route とは？

- HTTP のリクエスト（URL）とコントローラを紐付ける設定
- ここでは root へのリクエスト（GET /）を welcome コントローラの index メソッドに紐付ける
- rake routes で確認する

config/routes.rb の当該箇所をアンコメント

```
root 'welcome#index'
```

Controller と View 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

動作確認の方法

- 再度, rails server でアプリを起動する
- Web ブラウザで `http://localhost:3000/` を開いて確認する

コマンド

```
rails server
```


ここまですをコミットしておく

ビジネスア
プリケーシ
ン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

ここまでの内容

- ここまでの作業で、controller と view を 1 つ備える RoR アプリができた
- 作業が一区切りしたので、commit する（commit はひとかたまりの作業に対して行う）

コマンド

```
git add .  
git commit -m 'Create welcome controller and view'
```

22 Ruby on Rails アプリの生成と実行

23 Controller/View の作成

24 Heroku にデプロイする

25 <演習課題>

Gemfile 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

Heroku 用 Gem

- Gemfile に rails_12factor を追加する
- Ruby のバージョンも指定しておく
- Gemfile を変更したら必ず bundle install すること

Gemfile に追加する内容

```
gem 'rails_12factor', group: :production
ruby '2.1.2'
```

Git にコミット

ビジネスアプリ
ケーション
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

コミットする必要性

- Heroku にコードを送るには, git を用いる
- ローカルで最新版を commit しておく必要がある
- ついでに GitHub にも push しておく

コマンド

```
git commit -a -m 'Set up for Heroku'  
git push # origin master -> GitHub が省略されている
```

Heroku アプリの作成とディプロイ

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にディ
プロイする

<演習課題>

作成とディプロイ

- heroku コマンドを利用してアプリを作成する
- heroku create で表示された URL を開く
- git push でディプロイすると，Heroku からのログが流れてくる

コマンド

```
heroku create  
git push heroku master
```

22 Ruby on Rails アプリの生成と実行

23 Controller/View の作成

24 Heroku にデプロイする

25 <演習課題>

RoR 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

Ruby on Rails
アプリの生成
と実行

Controller/View
の作成

Heroku にデ
プロイする

<演習課題>

演習課題

- ここまでの説明に従い、Heroku で動作する RoR アプリを完成させなさい

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 7 章 DB を使うアプリの開発と継続的統合

26 DB と Scaffold の作成

27 RoR アプリのテスト

28 Travis CI との連携

29 <演習課題>

Scaffold

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

Scaffold とは

- scaffold - Google 検索
- RoR では、MVC の雛形を作る
 - CRUD 処理が全て実装される
- 多くのコードが自動生成されるので、branch を切っておくと良い
 - 動作が確認できたら branch をマージ
 - うまく行かなかったら branch ごと削除すれば良い

コマンド

```
git branch books
git checkout books
rails generate scaffold book title:string author:string
```

DB の Migrate

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

migrate とは

- Database のスキーマ定義の更新
- Scaffold を追加したり，属性を追加したりした際に行う

コマンド

```
rake db:migrate
```

route の確認

ビジネスアプリ
리케이션
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

route

- ルーティングの設定を確認しよう

コマンド

```
rake routes
```

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

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

動作確認の方法

- Web ブラウザで `http://localhost:3000/books` を開く
- CRUD 処理が完成していることを確かめる

コマンド

```
rails server
```

完成したコードをマージ

ビジネスアプリ
리케이션
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

ブランチをマージ

- 動作確認できたので、books branch をマージする
- 不要になったブランチは、git branch -d で削除する

コマンド

```
git add .  
git commit -m 'Generate books scaffold'  
git checkout master  
git merge books  
git branch -d books
```

Heroku にデプロイ

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

デプロイ

- ここまでのアプリをデプロイする
- heroku にある db も migrate する
- Web ブラウザで動作確認する

コマンド

```
git push heroku master  
heroku run rake db:migrate
```

Scaffold の作成を取り消す場合（参考）

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

取り消す操作

- migration を取り消す
- branch に一旦コミットして、master ブランチに移動
- branch を削除

コマンド

```
rake db:rollback  
git add .  
git commit -m 'Rollback'  
git checkout master  
git branch -D books
```


PostgreSQL クライアントのコマンド（参考）

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

■ psql で DB にログイン

Backslash コマンド	説明
	DB の一覧
c	DB に接続
d	リレーションの一覧
q	終了

26 DB と Scaffold の作成

27 RoR アプリのテスト

28 Travis CI との連携

29 <演習課題>

テストについて

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

ガイド

- A Guide to Testing Rails Applications —Ruby on Rails Guides

テストの実行

ビジネスアプリケーション
演習

中鉢欣秀・上田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

テストコード

- Scaffold はテストコードも作成してくれる
- テスト用の DB (rails_enpit_test) が更新される

コマンド

```
rake test
```

26 DB と Scaffold の作成

27 RoR アプリのテスト

28 Travis CI との連携

29 <演習課題>

Travis CI

ビジネスアプ
リケーション
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

— Travis CI: Building a Ruby Project

Travis の初期化

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

内容

- Travis にログインして初期化を行う
- init すると .travis.yml ができる

コマンド

```
travis login --auto      # GitHub のログイン情報でログイン
travis init               # 質問には全て Enter を押す
```

Heroku との連携

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

Heroku との連携

- Travis CI から Heroku への接続を設定する
- master 以外のブランチで実行すると、そのブランチのみ Heroku に送る（ようだ）
 - Travis CI: Heroku Deployment

コマンド

```
travis setup heroku
```


Travis で動かす Ruby のバージョン設定

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

設定ファイルの変更

■ Ruby のバージョン

.travis.yml (抜粋)

```
language: ruby
rvm:
- 2.1.2
```

Travis 用 DB 設定ファイル

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

Travis でのテスト DB

— テスト DB 用の設定ファイルを追加する

```
config/database.yml.travis
```

```
test:
  adapter: postgresql
  database: travis_ci_test
  username: postgres
```

Travis 上の DB 設定

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

設定ファイルの変更（追加）

- PostgreSQL のバージョン
- DB の作成
- Travis CI: Using PostgreSQL on Travis CI

.travis.yml（抜粋）

```
addons:  
  postgresql: "9.3"  
before_script:  
  - psql -c 'create database travis_ci_test;' -U postgres  
  - cp config/database.yml.travis config/database.yml
```

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

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

説明

- ここまでの設定で，GitHub に push されたコードは，Travis CI でテストされ，テストが通ったコミットが Heroku に送られるようになった
- Web ブラウザで Travis CI を開いて確認する

コマンド

```
git add .  
git commit -m 'Configure Travis CI'  
git push
```

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

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

Travis のログを閲覧

- Web ブラウザで Travis CI の画面を開く
- ログを読む

Heroku への Deploy

- テストが通れば、自動で Heroku に配備される
- 配備できたら Web ブラウザでアプリのページを開いて確認する

26 DB と Scaffold の作成

27 RoR アプリのテスト

28 Travis CI との連携

29 <演習課題>

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

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

演習課題

- books コントローラを備えたアプリを Heroku に配備しなさい

リンクの追加

ビジネスア
プリケーシ
ョン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

演習課題

- welcome コントローラの view から, books コントローラの view へのリンクを追加しなさい

Scaffold の追加

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

演習課題

- Scaffold を追加しなさい
- DB の migration を行い，動作確認しなさい
- うまく動作したら Heroku に配備しなさい

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

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

演習課題

- Travis 経由で Heroku へ deploy できるようにする

Status Image

ビジネスア
プリケーショ
ン
演習

中鉢欣秀・上
田隆一

DB と Scaffold
の作成

RoR アプリの
テスト

Travis CI と
の連携

<演習課題>

演習課題

- README.md を編集し，Travis のテスト状況を表示する Status Image を追加する
- Travis CI: Status Images