デザインプロジェクトD班 運用マニュアル

ソースコード

```
git clone https://github.com/delihiros/design-project.git
cd design-project
lein ring server ;; ただしXがある場合。
```

アプリ起動に必要なもの

- Leiningen
- Java
- Mysql

アプリの起動の仕方

以下のコマンドによってサーバーが起動する。

```
lein ring server
```

bgでの実行の場合

```
lein trampoline run &
```

アプリの管理ユーザー・テストユーザー等のユーザー名とパスワード

|ユーザー|ID|password| |-------:|-:|-------:| |大学(admin)|admin|admin| |修了生|graduate|graduate| |学生|student|student| |イベントのみ参加者|participant|participant|

その他

ソースはこちら

Clojureでの開発について

発表の際Clojureで開発したことによる違いについて聞かれましたが、ここでその回答とさせて頂きます。

JVMとleiningen

本チームはWindows,Linux,MacOSXと、多様な環境での開発が必要だった。 そのためそれぞれの設定や環境を緩和するようなシステムが必要だった。JVMはこれに対してのよい解決案だった。 またClojureのツールであるleiningenは、ライブラリのバージョン管理からブートまで面倒を見てくれるツールであり、 おかげでWindows上にLinuxをブートさせるといった環境セットアップや、ライブラリやイン

ストール手段の違いに時間を取られることがなかった。

マクロ

ルーティング周り(src/design-project/handler.clj)はすべてマクロで書かれている。 これによって直感的なシンタックスが提供され、高速な開発が行えた。

また、モデル側ではデータのバリデータとしてマクロを書くことで、きわめて簡単に大量の正確なバリデータを生成することができた。

データベースへの接続などもマクロになっており、これにより冗長な「接続して、これこれの処理をして、接続を解除する」といった頻出の処理を消し、ミスをなくした。

関数

高階関数が使えることで、関数にある機能を付け足した関数を作りたいなどといったことが、もとの関数に手を加えずに可能となった。 そのため、認証と通常のルーティングを論理的に切り離し、ルーティング関数から認証を必要とするルーティング関数を生成することにより、見つけにくいバグの発生を防いだ。 また条件が式なのでそのままフィルタ制作の際に流用できた。

短く正確に、本質だけにこだわって書けたと考えている。