情報システム演習

第６回基礎課題１

学籍番号：0312012151

氏名：村上潤紀

・フレームワークのメリット、デメリット

-メリット

生産性の向上。昨今のソフトウェア開発では同じプロジェクトに複数人がかかわり、プログラムを共有して開発してゆく。このときそれぞれが違った書き方をしていては誰かの書いたプログラムの理解から始まることになり生産性が落ちてしまう。ここで同じ書き方をあらかじめ指定しておけばグループ内でのプログラムの認識が同じになり生産性が向上する。

品質の向上。プログラムを作る際にやはり大事なことは「バグ」である。フレームワークを用いると基本的に関数の入出力が決まっていて、個人のコーディングの負担を極力減らすことにより品質の安定と向上が見込まれ、コーディング規約や命名規則もある程度統一しているのでそれに合わせたコーディングを行えば保守性も向上する。

-デメリット

制限が多い。メリットの要であるコーディング規約はきれいであればあるほどプログラムを書く際に制限が増えてくる。ルールを守ることで見やすいプログラムを書くことができるが書いてるプログラマーは自由にプログラミングすることができなくなる。さらにグループで決めたコーディング規約にのっとることで油断が生まれプログラムをあまり理解しないで書いてしまう傾向がある。だからこそ一人ひとりプログラムの大まかな流れを理解しておく必要性がある。

・フレームワーク

-symfony

Mojaviから派生し、Ruby用フレームワークRuby on Railの影響を色濃く受けて作られたフレームワーク。シンプルなテンプレートとヘルパーがあり、様々な環境に対応。

-codeigniter

他と比較して軽量で高速。シンプルな設定で簡単だが拡張性もある。コマンドラインが不要。コーディング規約が最小限。シンプルが売り。scaffolding機能というデータベースの基本操作に必要なモジュールを自動生成することができとても便利。

-マイクロフレームワーク「ちいたん」

世界最軽量でCakePHPを使用するほど大きなプロジェクトではなく、でも多少手のかかる物の場合に使えるフレームワークがほしいというときに使えるフレームワーク。

・Ruby on Rails

-特徴

Ruby on Railsは「同じことを繰り返さない」、「設定より規約」に重きを置いたRubyのフレームワーク。この理念により定義などの作業は一回だけでよい。またコンポーネントの統合は手動での設定を必要とせず自動で規約に従い行われる。

-MVCモデル

・Model

　データベース駆動のMVC WebアプリケーションではModelはRDBMSのテーブルを表すクラスを意味する。 RailsではActive Recordを通じてModelクラスを扱う。通常プログラマはActiveRecord::Baseクラスのサブクラスを作る必要がある。そうすることでRDBMSのどのテーブルを使うべきか、どういったカラムを持つべきかを自動的に決定してくれる。

・View

　MVCではViewは表示のためのロジックであり、コントローラクラスからのデータをどのように表示するかを規定している。 WebアプリケーションではHTML内に若干のコードを埋め込むことで実現される。

・Controller

MVCではControllerはRailsのAction Packには含まれるアプリケーションコントローラクラスによって扱われる。 WebベースMVCアプリケーションではWebブラウザを操作するユーザによりコントローラのメソッドが起動される。

参考URL：<http://el.jibun.atmarkit.co.jp/happy/2010/01/19-9042.html>

　　URL：<http://antenasites.com/2013/07/php-framework/>

URL：<http://sterfield.co.jp/programmer/codeigniter%E3%81%AE%E7%89%B9%E5%BE%B4.html>

URL：<http://www.presentation.bz/24>

URL：<http://maskana-soft.com/rails/pro/body/1>

URL：<http://ja.wikipedia.org/wiki/Ruby_on_Rails>

URL：<http://cakephp.seesaa.net/article/123192243.html>