

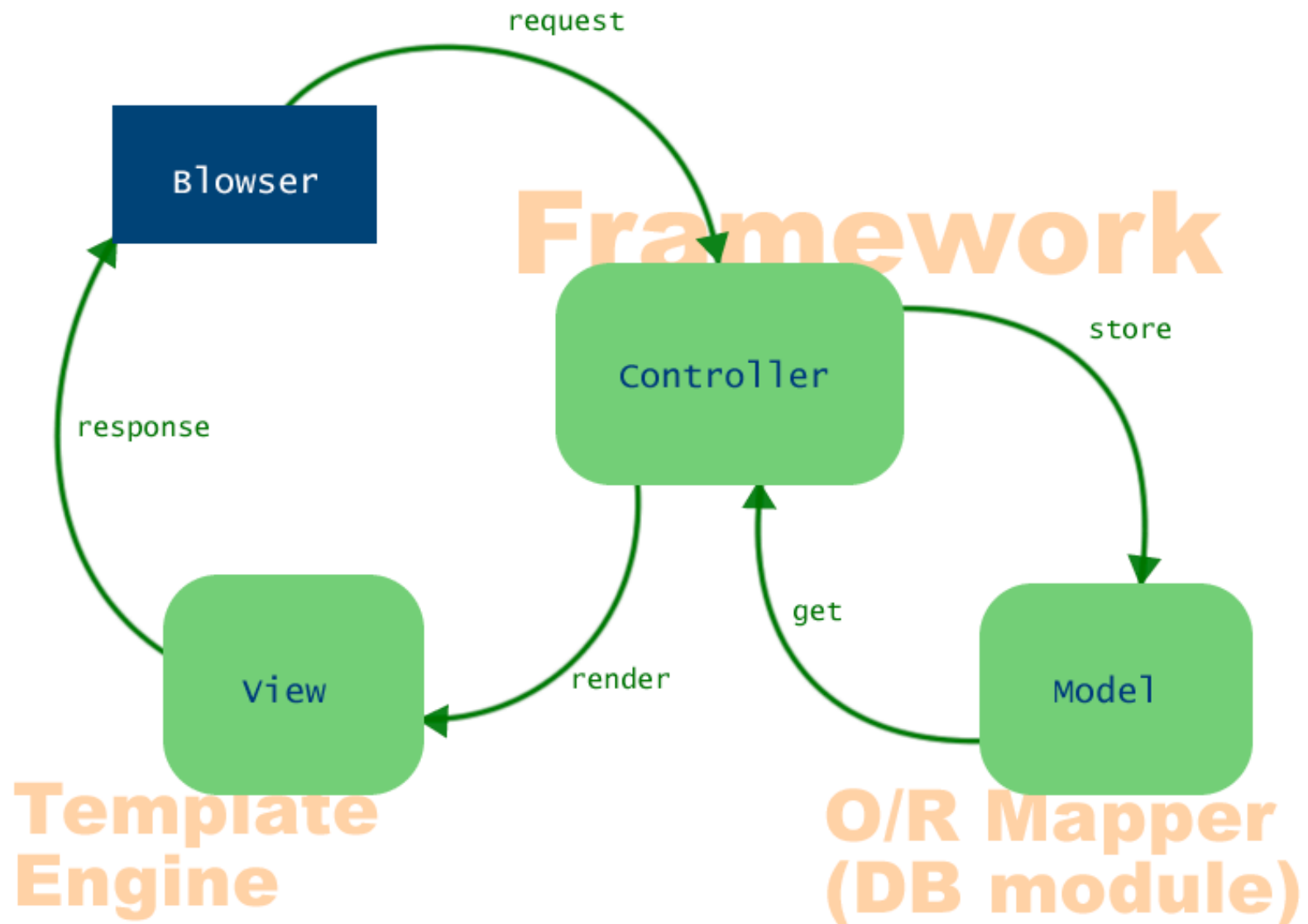
about Catalyst

daisuke komatsu <komatsu@plucore.jp>

Web Application Framework とは

- CGI や Web アプリケーションで、よく行う処理を定型化して、効率よく開発するための土台となるモジュールやクラス。
- リクエストを解析して、特定のサブルーチン（メソッド）を呼び出したりとか

WAF についておさらい



Perl では

- Catalyst
- Maypole
- Sledge
- CGI::Application
- TripletaiL
- Soozy
- Jifty
- etc.

plucore でよく使う WAF

- Catalyst + TT + DBIC
 - 規模がある程度大きい場合。
 - 専用サーバ + 特権
- CGI::Application + TT (+ DBIC)
 - CGI アプリ
 - 置けば動く
- Sledge + TT + CDBI
 - 一部外注さん

plucore でよく使う Catalyst 構成

- **TT (Template-Toolkit)**
 - 高機能テンプレートエンジン
 - 独自のタグで渡された変数のループ処理も可能
 - テンプレート内でメソッド (サブルーチン) が呼び出せる
 - DBIC の `resultset` を渡してテンプレート内で展開とかも可能
- **DBIC (DBIx::Class)**
 - O/R マッパー (オブジェクトとデータベースをマッピングする)
 - 遅延評価により必要になるところ以外では DB とのやりとりが極力発生しない

Catalyst のメリットデメリット

- メリット
 - ある程度の規模でも素早く開発
 - 機能一覧から DB 設計と URI 設計すればあまり迷わず開発できる
 - 多くの Plugin で大体やりたいことがサポートされてる
 - Test 用ミニサーバが付いてる
 - 起動してほげほげするだけならプログラマじゃなくてもできる
 - ヘルパースクリプトによる開発支援
- デメリット
 - 動作環境を選ぶ (専用サーバとか)

個人的雑感

- Catalyst は URI / Method マッパー
- 特定の URI とメソッドを結びつけられるため、そのことだけ考えて開発できる。

個人的雑感

- Catalyst は URI / Method マッパー
- 特定の URI とメソッドを結びつけられるため、そのことだけ考えて開発できる。

例)

`/user/create`



`MyApp::Controller::User#create`

`/key/create`



`MyApp::Controller::Key#create`

URI / メソッド / DB / テンプレート

- 商品詳細ページが必要だ
- URI は `/article/23` みたいにしよう
- 呼び出されるメソッドは
`MyApp::Contoroller::Article#detail` で
- `article` テーブルに商品の説明を貯めよう
- `MC::Article#detail` では `DBIC::Article` をよべばおk
- テンプレートは `root/src/article/detail.tt` で決まり



URI / メソッド / DB / テンプレート

- 商品詳細ページが必要だ
- URI は `/article/23` みたいにしよう
- 呼び出されるメソッドは
`MyApp::Contoroller::Article#detail` で
- `article` テーブルに商品の説明を貯めよう
- `MC::Article#detail` では `DBIC::Article` をよべばおk
- テンプレートは `root/src/article/detail.tt` で決まり



DB の設計と URI の設計ができれば、
プログラムの中核とテンプレートが決まる。

追加も楽ちん

- URI 決めて
- 必要だったら DB にテーブル作って
- ほげほげするコントローラ書いて
- テンプレート書いたらおしまい

Plam::Controller::*

URI	Class(module)	method
/	Plam::Controller::Root	index
/logout		logout
/user	Plam::Controller::User	index
/user/create		create
/user/*		detail
/user*/update_trac		update_trac
/user*/delete_trac		delete_trac
/user*/update_mailaddress		update_mailaddress
/user*/add_trac		add_trac
/user*/update_password		update_password
/entry	Plam::Controller::Entry	index
/entry/detail/*		detail
/entry/user/*		user
/tomorrow	Plam::Controller::Tomorrow	index
/tomorrow/form		form
/tomorrow/send		send

Catalyst Plugins

- **Catalyst::Plugin::Authentication**
 - 認証用フレームワーク
- **Catalyst::Plugin::Authentication::Role**
 - 権限付与（admin とか user とか。設定は自由。）
- **Catalyst::Plugin::Session**
 - セッション管理
- **Catalyst::Plugin::Email::Japanese**
 - メール送信（雛形+日本語処理）
- **Catalyst::Plugin::FillInForm**
 - フォームへの自動入力
 - 入力エラー時などに今まで入力した値とか

なんで使い回せないの？

- vkgtaro が下手だから
- Catalyst のアプリケーションは、クラス名にアプリ名が付いてるから
 - **Makit::Controller::User::Company** とか
 - 実際には lib/Makit/Controller/User/Company.pm
 - **Plam::Contoroller::Tomorrow** とか
 - 実際には lib/Plam/Contoroller/Tomorrow.pm
- DB の Table 構成が違うから
 - でも実際には一部使い回してます
 - user / user_role / role / session テーブル

でも実際には一部使い回しています

- コピペだけど（ダサイ
- ユーザ認証、ユーザ管理とか plugin 化すればもっとうまくいきそう。
- 今後のノウハウの蓄積とコードの蓄積でプラグイン化が進めばよく使われる機能とかは実装できるのではないかと思う。

Lazy-People.org

- lazy-people.org では一緒になにか作ろう
と思っている人を募集しています。
- <http://redmine.lazy-programmer.com/>
- IRC: [@irc.lazy-people.org](irc://irc.lazy-people.org/#project)
- Thanks!

ご質問？