Saas Архитектура и знакомство с Rails

- Сеть является клиентской/серверной архитектурой

- Она ориентирована на вопрос/ответ

основанный на ASCII-based протокол вопроса-ответа для передачи информации в сеть

- HTTP вопрос содержит:

– метод (GET, POST, etc.)"

– Универсальный идентификатор ресурса (URI)"

– HTTP версию протокола, понимаемую клиентом

– хедеры— дополнительная информация касаемо вопроса передачи

HTTP статус коды:

- HTTP ответ от сервера

2xx — все хорошо

– Пртокольная версия & код статуса =>"

3xx — resource moved

– Response headers"

4xx — access problem

– Response body"

5xx — server error

Создание динамического контента

Раньше большинство веб страниц были старыми текстовыми файлами

- Наиболее интересные Web 1.0/екоммерческие сайты действительно используют программы для создания страниц.
- Первоначально: шаблоны с встроенного кода "фрагменты"
- В конце концов, код стал "хвостом, который вилял собакой " и переехал из веб-сервера"

Сайты, которые являются программами (Saas)

- Как ты:

– составляешь универсальный идентификатор ресурса, чтобы корректировать программу и функцию?

– передаешь аргументы

– выполняешь программу на сервере?

– справляешься с хранением?

– работаешь с кукис?

– с ошибками?

– составляешь ответ для пользователя?

- Шаблоны поодерживают эти частые задачи

Сопоставьте понятия:

(a) presentation tier, (b) logic tier,
(c) persistence tier

☐ (a) Apache web server (b) Rack+Rails
(c) Relational database

☐ (a) Firefox (b) Apache web server
(c) PostgreSQL

☐ (a) Microsoft Internet Information Server
(b) Rack+Rails (c) Apache web server

☐ (a) Firefox (b) Microsoft Internet
Information Server (c) MySQL

Шаблон MVC дизайна

- Цель: разделить организауию данных от UI & презентации

путем представления контроллера !

– помогать действиям пользователя, запрашивающего доступ к данным.

– привести данные по рендерингу по виду

- Веб-приложения могут показаться "безусловно" MVC дизайном, но и другие варианты могут иметь место

Какое утверждение про Model-View-Controller не верно:

* " В SaaS приложениях в сети, действия контроллера и
и view contents передаются с использованием HTTP."
* Все MVC приложения имеют две части "клиентскую" (например, веб-браузер) и «облачную» часть . "
* Model-View-Controller один из нескольких возможных путей организовать Saas приложение.
* Peer-to-peer приложения могут быть организованы как Model-
View-Controller.

Помнить Vs Хранить

- Как отправить объект на хранение

Пример: Кино и обзоры

- Основные операции с объектом: CRUD (Create,

Read, Update, Delete)"

- ActiveRecord: каждая модель знает как

CRUD себя, используя обычные механизмы"

Rails модели хранения данных в реляционных Базах данных

- Каждый тип модели получает свою собственную таблицу базы данных!
  - У всех рядов в таблице одинаковые структуры
  - 1 строка в таблице == один экземпляр модели
  - Каждая колонка хранит значение атрибута модели
  - Каждая строка имеет уникальное значение для первичного ключа (по Конвенции, в Rails это целое и называется ID) "

I

Альтернатива: DataMappers

- Data Mapper ассоциирует отдельный картопостроитель с каждой моделью

– Идея: сохранитькартопостроение независимым от отдельного склада информации - работать с большим количеством баз.

– Используется Google AppEngine"

– Con: can’t exploit

RDBMS features to

simplify complex

queries & relationships"

- Вернемся к этому, когда подойдем к вопросу о ассоциациях

Какое утверждение неверно о модели в Model-View-Controller:

☐" CRUD действия пожходят только для моделей работающих в базе данных, которые поддерживают ActiveRecord."

☐ Часть работы модели заключается в том, чтобы конвертирвать внутреннюю память и хранить репрезентации объектов.

☐ Хотя данные модели показываются в Виде, прямые действия модели происходят с помощью контролллеров.

☐ Хотя DataMapper не использует роджственные базы, это возможный вариант применения модели.

Маршруты

- В MVC, каждое действие юзера может быть также осуществлено контроллером
– Ruby "методом, который занимается данным действием
- Карты маршрута <HTTP method, URI> to controller action"
- "

Route! Action!

GET /movies/3 Show info about movie whose ID=3"

POST /movies Create new movie from attached form data"

PUT /movies/5 Update movie ID 5 from attached form data"

DELETE /movies/5 Delete movie whose ID=5"

Краткое знакомство с ситемой маршрутизации rails

- отправка <method,URI> чтобы исправить действия контроллера
- Осуществляет вспомогательные методы, которые генерирует

<method,URI> пара с заланными действиями контроллера

- анализирует параметры запроса из URI как форму

представления в удобный hash

- Встроенные ярлыки для генерации CRUD маршрутов

GET/movies/3/editHTTP/1.0

- подходит маршрут:"

GET /movies/:id/edit {:action=>"edit", :controller=>"movies"}"

- разбор параметров шаблона: params[:id] = "3"
- отправка для редактирования метода в movies_controller.rb
- Чтобы включить URI в генерированный вид, который будет поддерживать форму

чтобы обновлять действия контроллера [:id]==3,

call helper:

update_movie_path(3) # => PUT /movies/3!

REST (Representational State Transfer)

- Идея. указывать дял автономных вопросов ресурсы для выполнения операции и что нужно сделать.
  – Roy Fielding’s PhD thesis, 2000"
  – Wikipedia: “описание фичи, сделанное после ее реализации”!
- Сервис (in the SOA sense) , чьи операции похожи с RESTful service"
- Ideally, RESTful URIs name the operations"

Какое утверждение неверно о
Rails RESTful routes и ресурсов на какие они ссылаются:

- " Ресурс может быть существующим контентом или просьбой заменить что то"
- Каждый маршрут должен вызывать действия конроллера "
- Один распространненый набор RESTful действий это
CRUD actions на моделях. "
- Маршрут всегда содержит один или более параметров, как: :id, чтобы показать ресурс

Путешествие по приложению Rails приложению

1. Маршруты (in routes.rb) карт входящих URL's для действий контроллеров и извлесения функциональных параметров

- Маршрут шаблонных параметров(eg :id), плюс все, что идет вслед за"?" in URL,

ставится в квадратные скобки и доступно для действий контроллера

2. Действия контроллера устанавливают переменные, которые показыват вид

1. Подзаголовки и названия видов совпадают с названиями действий контроллера

- Действие контроллера влияет на вид

Почему каждое действие с
SaaS app вызывает изменения?

☐Из-за соглашения о конфигурации

☐Потому что HTTP это протокол вопрос-ответ

☐ Пртому что Model-View-Controller , имеет в виду что каждое действие оказывает влияние на вид

☐ Все из вариантов