Московский государственный технический университет
имени Н. Э. Баумана
Факультет Информатика и системы управления
Кафедра Компьютерные системы и сети

«УТВЕРЖДАЮ»
Заведующий кафедрой ИУ-6
_____ Сюзев В.В.

Р.С. Самарев

Программирование веб-приложений с использованием Ruby on Rails

Методические указания по выполнению практикумов № 6, 7
по дисциплине Языки Интернет-программирования

Москва 2014

Практикум № 6. Формирование и обработка RSS-XML.

Цель работы

Освоить принципы формирования и обработки XML-данных на примере веб-приложения, построенного с использованием Ruby on Rails.

Задание

Модифицировать код Практикума 5 таким образом, чтобы по запросу с указанными параметрами выдавался результат в формате RSS 2.0.

1. Сформировать элементы RSS как строки матрицы, вычисляемой в Практикуме 5. Заполнить недостающие поля произвольным образом. Использовать в качестве шаблона прилагаемый пример.
2. Проверить, что браузер может отображать формируемый поток как подписку. Проверить корректность формируемой XML (исходный код страницы в браузере).
3. Проверить формирование RSS и сохранить в файл для отладки второго приложения.
4. Написать функциональный тест, проверяющий формат выдаваемых данных при запросе RSS.

Разработать второе приложение, являющееся посредником между клиентом и первым приложением, задачей которого является преобразование RSS в HTML или передача в неизменном виде браузеру для отображения браузером. Приложение должно запускаться с указанием номера порта TCP, отличным от номера порта первого приложения (например rails server -p 3001)!

1. Подготовить каркас приложения, а также форму формирования запроса, форму отображения результата и соответствующие действия контролера.
2. Написать код серверного преобразования RSS (узлов XML) в массив с помощью разбора XML в модели DOM. Проверить работоспособность на сохраненном ранее тестовом файле с RSS. Использовать парсер nokogiri. Для использования Nokogiri::XML необходимо включить в Gemfile строку gem 'nokogiri', а в коде контроллера добавить require 'nokogiri'.
3. Подключить запрос RSS с первого приложения (с помощью метода open из 'open-uri') и проверить работу приложений в связке.
4. Написать функциональный тест, проверяющий что при различных входных данных результат генерируемой страницы различен.
5. Доработать код контроллера приложения для выдачи браузеру RSS-потока в неизменном виде (трансляция без изменений).
6. Проверить, что браузер получает RSS в неизменном виде.

Итоговая форма ввода параметра должна содержать кнопки или селектор, позволяющие проверить выдачу rss без изменения и результат разбора в виде таблицы.

Примеры кода:

Контроллер:

```
class RssController < ApplicationController
  def view
    @items = Array.new(10){rand(10)}
    respond_to do |format|
      format.rss
    end
  end
end
```

Шаблон view.rss.erb

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
  <channel>
   <title>RSS Title</title>
   <description>This is an example of an RSS feed</description>
   <link><%= link_to "RSS feed", params.merge(:format => :rss), :class => "feed_link" %></link>
   <pubDate><%= DateTime.now %></pubDate>

   <% @items.each do |item| %>
   <item>
     <title>Example entry</title>
     <description>Here is some text containing an interesting description. item: <%= item.to_s
%></description>
     <link><%= link_to "RSS feed", params.merge(:format => :rss), :class => "feed_link" %></link>
     <pubDate><%= DateTime.now %></pubDate>
   </item>
   <% end %>

  </channel>
</rss>
```

Требования к отчету

Результаты должны быть представлены в виде двух файлов:

- отчет в формате pdf/odt/doc;
- архив rails-приложения (zip, tgz,7z).

Практикум № 7. Добавление модели. ORM. Разработка БД, подключение, хранение и поиск данных.

Цель работы

Освоить принципы применения средств объектно-реляционного преобразования в составе Ruby on Rails.

Задание

Модифицировать код Практикума 5 таким образом, чтобы запросы, которые были ранее выполнены, сохранялись в БД и при следующем запросе не требовали повтора вычислений.

- Сформировать модель в соответствии с потребностями хранения данных. Входные параметры являются ключами, по которым извлекается результат.
- Выполнить создание БД и миграцию соответствующими запросами rake.
- Написать тест на добавление и поиск данных с помощью модели. Проверить выполнение теста.
- Модифицировать код приложения таким образом, чтобы результат вычислений преобразовывался в строковый или бинарный формат (на выбор: json, xml, и пр.). Проверить через отладочную печать в консоль, что преобразование выполняется корректно.
- Вставить код для сохранения данных в БД и запрос на поиск предыдущего результата вычислений.
- Добавить действие в контроллер, позволяющее определить, что хранится в БД через сериализацию в XML.
- Проверить, что при выполнении запроса, данные добавляются в БД.
- При помощи консоли сообщений Webrick определить, производится ли поиск результата предыдущего запроса в БД и не повторяются ли одни и те же вычисления.
- Модифицировать модель таким образом, чтобы добавление записей с одинаковыми параметрами было невозможно.
- Реализовать тест модели, проверяющий невозможность повторного добавления одних и тех же результатов вычислений.
- Реализовать функциональный тест, проверяющий, что результаты вычислений различны при различных входных параметрах.
- Проверить маршруты приложения с помощью rake routes и убрать лишние. Обеспечить доступ при обращении по адресу /.

Требования к отчету

Результаты должны быть представлены в виде двух файлов:

- отчет в формате pdf/odt/doc;
- архив rails-приложения (zip, tgz,7z).

Отчет должен содержать:

- ФИО, номер группы и текст задания;
- перечень и содержимое файлов, которые были изменены в процессе создания приложения.
- XML-распечатку содержимого БД (ограничить несколькими записями так, чтобы результат поместился на 1-2 страницах).
- Примеры SQL-кода добавления и извлечения данных из БД из отладочной консоли сервера Webrick.

Литература

1. Материалы лекций по курсу Языки Интернет программирования.
2. Оби Фернандес. Путь Rails. Подробное руководство по созданию приложений в среде Ruby on Rails. -М. Символ-Плюс, 2009 г.
3. Гибкая разработка веб-приложений в среде Rails. 4-е издание Сэм Руби, Дэйв Томас, Дэвид Хэнссон. Серия: Для профессионалов.- Питер: 2013.- 464 стр.