Конспект RubySchool.us [5]

Урок 31

Как работает HTTP

> Изучи ссылку: How exacty HTTP protocol works? - Stack Overflow https://stackoverflow.com/questions/20918321/how-exacty-http-protocol-works и what happens when you type in a URL in browser - Stack Overflow https://stackoverflow.com/questions/2092527/what-happens-when-you-type-in-a-url-in-browser

Порт - это абстракция операционной системы, или абстракция протокола.

```
1. Resolve domain if not an IP (DNS query)
2. Open port 80 by default if not SSL and not overridden by a colon (http: //host:port/)
3. Send request (#1) for http: //host/uri/here?other=stuff&too
4. Receive response (#2)
5. Close
```

Порты:

- 80 - http
- 443 - https

Запрос:

```
GET /uri/here?other=stuff&too HTTP/1.1
Host: host
Other: Headers, too.  Such as cookies
Header: Value
```

Ответ:

```
HTTP/1.1 200 OK
Other: Headers, too.  Such as cookies
Header: Value

<html>Actual HTTP payload is here, could be HTML data, downloaded file data, etc.
```

> Изучи ссылку: Сетевая модель OSI — Википедия https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI

what happens when you type in a URL in browser

```
1. browser checks cache; if requested object is in cache and is fresh, skip to #9
2. browser asks OS for server's IP address
3. OS makes a DNS lookup and replies the IP address to the browser
4. browser opens a TCP connection to server (this step is much more complex with HTTPS)
5. browser sends the HTTP request through TCP connection
6. browser receives HTTP response and may close the TCP connection, or reuse it for another request
7. browser checks if the response is a redirect or a conditional response (3xx result status codes), authorization request (401), error (4xx and 5xx), etc.; these are handled differently from normal responses (2xx)
8. if cacheable, response is stored in cache
9. browser decodes response (e.g. if it's gzipped)
10. browser determines what to do with response (e.g. is it a HTML page, is it an image, is it a sound clip?)
11. browser renders response, or offers a download dialog for unrecognized types
```

> what happens when you type in a URL in browser - Stack Overflow

stateless (без состояния)

HTTP - это протокол без состояния, т.к. сервер не прицепляет никакого специального значения.

> Изучить ссылку: Why is it said that "HTTP is a stateless protocol"? - Stack Overflow https://stackoverflow.com/questions/13200152/why-is-it-said-that-http-is-a-stateless-protocol

> Протокол без сохранения состояния — Википедия https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%B1%D0%B5%D0%B7_%D1%81%D0%BE%D1%85%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F_%D1%81%D0%BE%D1%81%D1%82%D0%BE%D1%8

Если сервер разрывает соединение, как же сделать так, чтобы работала авторизация и мы были уверены, что это тот самый пользователь. Для этого существует специальный механизм.

> Программа charles-proxy - анализ трафика

> url unescape online (search query)

> Изучи ссылку: RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1 https://tools.ietf.org/html/rfc2616

gem devise - для авторизации

> GitHub - plataformatec/devise: Flexible authentication solution for Rails with Warden.

> Аутентификация в Rails-приложениях с помощью Devise. Часть 1: базовая настройка / Хабр

Vanilla.js == JS

> Vanilla JS - http://vanilla-js.com/

Функция в js

```
function foo()
{
  alert('Hello!');
}
```

В JS функция вызывается по хэндлеру/событию (обработчик):

```
<button onclick="foo()">Click me!</button>

<script>
  function foo() {
    alert("Hello!");
  }
</script>
```

Переменные:

```
<button onclick="foo()">Click me!</button>

<script>
  var x = 1;

  function foo() {
    alert(x);
    x = x+1;
  }
</script>
```

Урок 32

Разбор вопросов:

> Изучить: 15 Questions to Ask During a Ruby Interview · GitHub https://gist.github.com/krdprog/64a463de21fe77a8946019fde6662d67

- классы содержат данные, имеют методы, которые взаимодействуют с этими данными и используются для того, чтобы создавать объекты на основе этих классов
- объект - это экземпляр класса. для некоторых - это коренной класс руби Object
- модуль - механизм, который служит для namespaces (пространства имён) module/end - Namespace::Class.method. Ещё, модули предоставляют механизм для множественного наследования с помощью миксинов. module/end - extend
- три уровня доступа для модулей и классов: public - по умолчанию, protected, private - методы доступные только внутри самого класса

> ООП с примерами (часть 1) / Хабр ООП с примерами (часть 2) / Хабр
>
> GitHub - ro31337/first-visit-js: Tiny jQuery plugin to display a message to the user on the first visit to a page

localStorage

```
window.localStorage
```

> HTML5 Web Storage https://www.w3schools.com/html/html5_webstorage.asp
>
> Почему не стоит использовать LocalStorage / Хабр
>
> LocalStorage на пальцах
>
> window.localStorage - Web APIs | MDN

```
function foo() {
  var x = window.localStorage.getItem('score');

  window.localStorage.setItem('score', 555);

  alert(x);
}
```

```
function foo() {
  var x = window.localStorage.getItem('score'); // это как x = hh['score'] в ruby

  // x * 1 - чтобы преобразовать строку в число
  x = x * 1 + 1;

  window.localStorage.setItem('score', x); // hh['score'] = x

  alert(x);
}
```

> Looping through localStorage in HTML5 and JavaScript - Stack Overflow https://stackoverflow.com/questions/3138564/looping-through-localstorage-in-html5-and-javascript

ActiveRecord

> ActiveRecord::ConnectionAdapters::TableDefinition
>
> ActiveRecord::ConnectionAdapters::SchemaStatements

- :primary_key
- :text
- :integer
- :float
- :decimal
- :datetime
- :timestamp
- :time
- :date
- :binary
- :boolean

PizzaShop

- цены рекомендуется хранить в базе в минимальных величинах

```
# + to app.rb

require 'sinatra'
require 'sinatra/reloader'
require 'sinatra/activerecord'

set :database, "sqlite3:database.db"

class Product < ActiveRecord::Base
end

get '/' do
  erb :index
end
```

```
rake db:create_migration NAME=create_products
```

```
# + to db/migrate/9279387982_create_products.rb

class CreateProducts < ActiveRecord::Migration[5.2]
  def change
    create_table :products do |t|
      t.string :title
      t.text :description
      t.decimal :price
      t.decimal :size
      t.boolean :is_spicy
      t.boolean :is_veg
      t.boolean :is_best_offer
      t.string :path_to_image

      t.timestamps
    end
  end
end
```

Seed database. Наполним вручную базу данных

```
rake db:create_migration NAME=add_products
```

```
# + to db/migrate/786238472_add_products.rb

class AddProducts < ActiveRecord::Migration[5.2]
  def change
    Product.create :title => 'Гавайская',
      :description => 'Это гавайская пицца',
      :price => 350,
      :size => 30,
      :is_spicy => false,
      :is_veg => false,
      :is_best_offer => false,
      :path_to_image => '/images/01.jpg'

    Product.create :title => 'Пепперони',
      :description => 'Это пицца Пепперони',
      :price => 450,
      :size => 30,
      :is_spicy => false,
      :is_veg => false,
      :is_best_offer => true,
      :path_to_image => '/images/02.jpg'

    Product.create :title => 'Вегетарианская',
      :description => 'Это вегетарианская пицца',
      :price => 400,
      :size => 30,
      :is_spicy => false,
      :is_veg => true,
      :is_best_offer => false,
      :path_to_image => '/images/03.jpg'
  end
end
```

И, сделаем:

```
rake db:migrate
```

Домашнее задание:

- сделать страницу вывода продуктов

Урок 33

Разбор вопросов (2):

> Изучить: 15 Questions to Ask During a Ruby Interview · GitHub https://gist.github.com/krdprog/64a463de21fe77a8946019fde6662d67

- три способа вызвать метод в руби:

```
# 1
object = Object.new
puts object.object_id

# 2
puts object.send(:object_id)

# 3
puts object.method(:object_id).call
```

- оператор ||=

> operators - What does ||= (or-equals) mean in Ruby? - Stack Overflow
>
> ruby - Что делает оператор «|| =» в рубине?

- Что такое self? self всегда указывает на текущий объект. Может быть вызван без создания объекта.

```
class WhatIsSelf
  def test
    puts "At the instance level, self is #{self}"
  end

  def self.test
    puts "At the class level, self is #{self}"
  end
end

WhatIsSelf.test
 #=> At the class level, self is WhatIsSelf

WhatIsSelf.new.test
 #=> At the instance level, self is #<WhatIsSelf:0x28190>
```

> 15 Questions to Ask During a Ruby Interview · GitHub

- Что такое Proc? Процедура. Три типа:

1. анонимные методы (функции без имени)
2. lambda
3. блок

```
# wants a proc, a lambda, AND a block
def three_ways(proc, lambda, &block)
  proc.call
  lambda.call
  yield # like block.call
  puts "#{proc.inspect} #{lambda.inspect} #{block.inspect}"
end

anonymous = Proc.new { puts "I'm a Proc for sure." }
nameless  = lambda { puts "But what about me?" }

three_ways(anonymous, nameless) do
  puts "I'm a block, but could it be???"
end
 #=> I'm a Proc for sure.
 #=> But what about me?
 #=> I'm a block, but could it be???
 #=> #<Proc:0x00089d64> #<Proc:0x00089c74> #<Proc:0x00089b34>
```

Продолжение PizzaShop

- нам надо создать корзину, но чтобы при закрытии браузера данные сохранялись.

Помежуточная проверка работает ли кнопка:

```
<p><button onclick="alert('hello')">Добавить в корзину</button></p>
```

Number 2:

js:

```
function add_to_cart() {
  alert('hello all!');
}
```

html:

```
<p><button onclick="add_to_cart()">Добавить в корзину</button></p>
```

> Плохие программисты беспокоятся о коде. Хорошие программисты беспокоятся о структурах данных и их взаимодействии (Л. Торвальдс)

```
# + to app.rb

require 'sinatra'
require 'sinatra/reloader'
require 'sinatra/activerecord'

set :database, "sqlite3:database.db"

class Product < ActiveRecord::Base
end

get '/' do
  erb :index
end

get '/products' do
  @products = Product.all
  erb :products
end
```

```
# + to views/products.erb

<h1>Наша продукция:</h1>

<table cellpadding="10" cellspacing="0" border="1">
<% @products.each do |product| %>
  <tr>
    <td>
      <h2><%= product.title %></h2>
      <p><strong>Описание:</strong> <%= product.description %></p>
    </td>

    <td><img src="<%= product.path_to_image %>" alt="<%= product.title %>"></td>

    <td>
      <p><strong>Цена:</strong> <%= product.price %> руб.</p>
      <p><strong>Размер:</strong> <%= product.size %> см</p>
      <p><button onclick="add_to_cart(<%= product.id %>)">Добавить в корзину</button></p>
    </td>
  </tr>
<% end %>
</table>
```

Структура данных для нашей корзины заказа в PizzaShop - хеш:

- key - id продукта
- value - количество

Перед написанием на js напишем на ruby:

```
order = {}

loop do
  print 'Введите id продукта: '
  id_product = gets.strip

  print "Сколько штук вы хотите заказать: "
  amount_now = gets.strip.to_i

  amount = order[id_product].to_i
  amount += amount_now

  order[id_product] = amount

  puts order.inspect
end
```

> GitHub - krdprog/order-counter-ruby: Order counter (Ruby) https://github.com/krdprog/order-counter-ruby/

На сервер мы будем передавать строку.

```
1 = 5, 2 = 12, 3 = 0
```

Можно использовать json.

Теперь напишем это на JS:

```
// + to public/js/main.js

function add_to_cart(id)
{
  var key = 'product_' + id;

  var x = window.localStorage.getItem(key);
  x = x * 1 + 1;
  window.localStorage.setItem(key, x);
}
```

Смотрим на наличие ошибок в консоли Firefox веб-инспектора.

И, смотрим в хранилище веб-инспектора. Или в консоли веб-инспектора набрать:

```
localStorage
```

Очистить:

```
localStorage.clear();
```

> Когда нужно делать рефакторинг кода. Ответ: всегда.

Сделаем счётчик товаров в корзине:

- ламерский способ - создать переменную counter, но это не будет учитывать наш localStorage

Нам надо пройтись по хешу и посчитать продукты, которые добавили в корзину:

```
# + to app-order-calc.rb

# calculate total number of items in cart
total = 0

order.each do |key, value|
  total += value
end

puts "Total: #{total}"
```

> javascript - Listing localstorage - Stack Overflow https://stackoverflow.com/questions/2841029/listing-localstorage#2841042
>
> javascript - enumerating localStorage properties - Stack Overflow https://stackoverflow.com/questions/27946563/enumerating-localstorage-properties

Домашнее задание: - найти про enumerate localStorage - найти как пройтись по каждому элементу хеша localStorage - написать JS функцию подсчёта общего количества заказанных продуктов в корзине.

Урок 34

x ||= y

Это сокращённо:

```
x = x || y
```

Что означает:

```
x || y
```

|| - логическое ИЛИ

```
if x == 1 || x == 2
```

Срабатывает первое условие, если не срабатывает, берёт второе условие:

```
nil || 4 #=> 4
false || 2 #=> 2
123 || 2 #=> 123

x = x || 4
```

```
x = false
x = x || 2
puts x
#=> 2

x = true
x = x || 2
puts x
#=> true

x = 5
x = x || 2 # это мы можем заменить на x ||= 2
puts x
#=> 5

x = 5
x ||= 2
puts x
#=> 5
```

x ||= y используется для установки значения по умолчанию

Продолжаем PizzaShop:

Напишем подсчёт количества товаров в корзине на js:

```
// + to public/js/main.js

function cart_get_number_of_items()
{
  var cnt = 0;

  for (var i = 0; i < window.localStorage.length; i++)
  {
    var key = window.localStorage.key(i); // получаем ключ
    var value = window.localStorage.getItem(key); // получаем значение

    if(key.indexOf('product_') == 0)
    {
      cnt = cnt + value * 1;
    }
  }

  return cnt;
}
```

Проверим в консоли firefox:

```
cart_get_number_of_items()
```

```
# + to views/layout.erb
<form action="/cart" method="POST">
  <input type="submit" value="Корзина (0 шт.)">
</form>
```

ИТОГО:

```
# + to views/layout.erb

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title></title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>

<p><a href="/">Главная страница</a> | <a href="/products">Наша продукция</a>

  <form action="/cart" method="POST">
    <input type="hidden" name="orders" id="orders_input">
    <input type="submit" id="orders_button">
  </form>
</p>

<%= yield %>


  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="/js/main.js"></script>

  <script>
    $(function() {
      update_orders_input();
      update_orders_button();
    });
  </script>

</body>
</html>
```

```
// + to public/js/main.js

function add_to_cart(id)
{
  var key = 'product_' + id;

  var x = window.localStorage.getItem(key);
  x = x * 1 + 1;
  window.localStorage.setItem(key, x);

  update_orders_input();
  update_orders_button();
}


function cart_get_number_of_items()
{
  var cnt = 0;

  for (var i = 0; i < window.localStorage.length; i++)
  {
    var key = window.localStorage.key(i); // получаем ключ
    var value = window.localStorage.getItem(key); // получаем значение

    if(key.indexOf('product_') == 0)
    {
      cnt = cnt + value * 1;
    }
  }

  return cnt;
}


function update_orders_input()
{
  orders = cart_get_orders();
  $('#orders_input').val(orders);
}


function cart_get_orders()
{
  var orders = '';

  for (var i = 0; i < window.localStorage.length; i++)
  {
    var key = window.localStorage.key(i); // получаем ключ
    var value = window.localStorage.getItem(key); // получаем значение

    if(key.indexOf('product_') == 0)
    {
      orders = orders + key + '=' + value + ',';
    }
  }

  return orders;
}


function update_orders_button()
{
  var text = 'Корзина (' + cart_get_number_of_items() + ' шт.)';
  $('#orders_button').val(text);
}
```

Домашнее задание:

- на странице /cart вывести в виде таблицы список продуктов в корзине и их количество
- на странице /cart сделать так, чтобы форма сабмитилась по адресу /order и чтобы в базу данных заносился заказ: имя, телефон, адрес доставки, список купленных товаров (в виде текстового поля).

Ruby on Rails

Я устанавливаю ruby через RVM

Посмотрим доступные версии Rails:

```
gem search '^rails$' --all
```

Чтобы установить конкретную версию, введите (вместо rails_version - номер версии):

```
gem install rails -v rails_version
```

С помощью gemset-ов можно использовать вместе разные версии Rails и Ruby. Это делается с помощью команды gem.

```
 rvm gemset create gemset_name # create a gemset
rvm ruby_version@gemset_name  # specify Ruby version and our new gemset
```

Gemset-ы позволяют создавать полнофункциональные окружения для gem-ов, а также настраивать неограниченное количество окружений для каждой версии Ruby.

> GitHub - DefactoSoftware/Hours: Time registration that doesn't suck

Rails-приложение может запускаться в 3 типах окружения:

- development
- test
- production

Создадим новое рейлс-приложение:

```
rails new blog
```

Последовательность запуска rails (для справки):

```
boot.rb -> rails -> environment.rb -> development.rb (test.rb or production.rb)
```

Запуск приложения:

```
rails s
```

Если не запустится, установи nodejs:

```
sudo apt-get install nodejs
```

Если всё ок, приложение запустится и можно открывать в браузере: http://localhost:3000/

Обновлять bundle можно командой:

```
bundle update
```

MVC

Model, View, Controller

Controller - отвечает за работу с какой-либо сущностью

Создадим контроллер:

```
rails generate controller home index
```

Найдём файл /app/controllers/home_controller.rb

Найдём файл /app/views/home/index.html.erb

Откроем в браузере: http://localhost:3000/home/index

> Обычно /home/index создаётся для главной страницы сайта Надо открыть /config/routes.rb и прописать:

```
Rails.application.routes.draw do
  get '/' => 'home#index'
end
```

> Изучить: Rails Routing from the Outside In — Ruby on Rails Guides http://guides.rubyonrails.org/routing.html

Урок 35

Разбор вопросов на интервью

> 15 Questions to Ask During a Ruby Interview · GitHub

- Что такое юнит-тесты. Юнит-тесты предназначены для тестирования отдельных модулей программы (к частям, которые не делимы).

```
require "test/unit"

class Brokened
  def uh_oh
    "I needs fixing"
  end
end

class BrokenedTest < Test::Unit::TestCase
  def test_uh_oh
    actual = Brokened.new
    assert_equal("I'm all better!", actual.uh_oh)
  end
end
 #=> Started
 #=> F
 #=> Finished in 0.663831 seconds.
 #=>
 #=>   1) Failure:
 #=> test_uh_oh:11
 #=> <"I'm all better!"> expected but was
 #=> <"I needs fixing">.
 #=>
 #=> 1 tests, 1 assertions, 1 failures, 0 errors
```

> Вызываем метод и проверяем, что результат совпадает с эталоном

test coverage - покрытие тестами

Интеграционное тестирование - тестирование сайта в браузере

Как много надо покрывать кода тестами?

Юнит тестирование нужно, чтобы при возрастании сложности приложения функциональность большого приложения сохранялась.

- Статическая и динамическая типизация

```
// Java
public boolean isEmpty(String s) {
  return s.length() == 0;
}
```

```
# ruby
def empty?(s)
  return s.size == 0
end
```

В руби меньше кода и он более гибкий.

> oDesk and Elance Tests 2015
>
> Ruby on Rails Test 2015 - oDesk

Продолжаем PizzaShop

Нам надо разбить строку результата заказа в Ruby

```
"product_1=9,product_3=2,"
```

Создадим отдельную руби-программу:

```
# pizza-shop-other.rb

orders = "product_1=9,product_3=2,"

s1 = orders.split(/,/)

s1.each do |x|
  s2 = x.split(/=/)
  s3 = s2[0].split(/_/)

  key = s3[1]
  value = s2[1]

  puts "Priduct Id: #{key}, number of items: #{value}"
end
```

Это не эффективный способ, лучше использовать json (позволяет поддерживать формат данных любой сложности).

Перепишем в метод:

```
# + to app.rb

post '/cart' do
  orders_input = params[:orders]

  @items = parse_orders_input orders_input

  @items.each do |item|
    # id, cnt
    item[0] = Product.find(item[0])
  end

  erb :cart
end

# Parse orders line:
def parse_orders_input orders_input

  s1 = orders_input.split(/,/)

  arr = []

  s1.each do |x|
    s2 = x.split(/=/)
    s3 = s2[0].split(/_/)

    id = s3[1]
    cnt = s2[1]

    arr2 = [id, cnt]
    arr.push arr2
  end

  return arr
end
```

```
# + to /views/cart.erb

<h2>Заказанные товары:</h2>

<table border="1" cellspacing="0" cellpadding="10">
  <tr>
    <th>Товар</th>
    <th>Цена</th>
    <th>Количество</th>
  </tr>

  <% total_qty = 0 %>
  <% total_price = 0 %>

  <% @items.each do |item| %>
  <tr>
    <td><%= item[0].title %></td>
    <td><%= item[0].price %> руб.</td>
    <td><%= item[1] %></td>
  </tr>

  <% total_qty += item[1].to_i %>
  <% total_price += item[0].price * total_qty %>
  <% end %>

  <tr style="background: yellow;">
    <td><strong>Сумма:</strong></td>
    <td><%= total_price %> руб.</td>
    <td><%= total_qty %></td>
  </tr>

</table>
```

Отношение между сущностями:

- один ко многим
- многие к одному
- многие ко многим
- один к одному

Доделаем PizzaShop не лучшим, но простым путём. Создадим форму и отправим заказ на сервер (сохраним в базе данных).

> GitHub - krdprog/PizzaShop-rubyschool: Pizza Shop (rubyschool project). Ruby, Sinatra, ActiveRecord, JS, localStorage https://github.com/krdprog/PizzaShop-rubyschool

Домашнее задание:

- сделать модель Order с полями из формы cart.erb, не забыть про миграцию
- добавить post-обработчик /place_order в котором получать данные из страницы и сохранять в базу данных
- выводить на экран сообщение "Заказ принят"
- выводить на экран страницу со всеми принятыми заказами

Продолжение конспекта: Урок 36-40 - https://github.com/krdprog/rubyschool-notes/blob/master/rubyschool-notes-06.md

Содержание конспекта: