Языки Интернет-программирования

Лекция 9. Асинхронный обмен данными

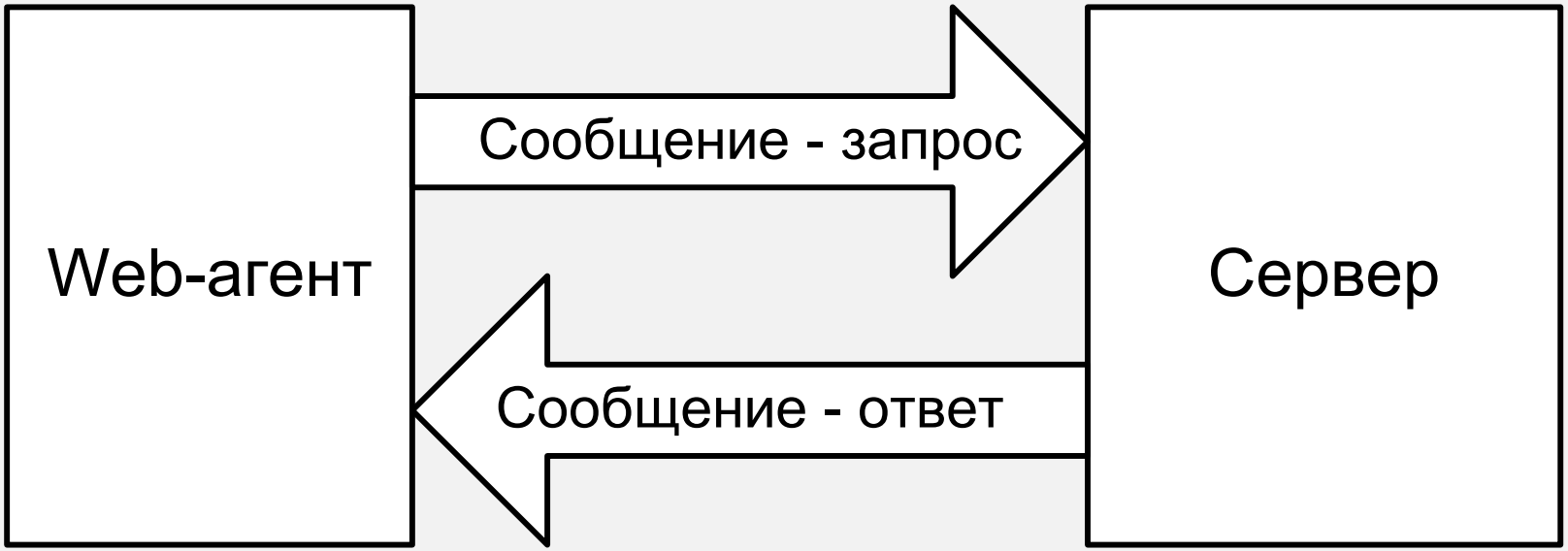
- Синхронная / асинхронная (AJAX) передача данных
- Форматы и обработка данных в AJAX (JSON, XML, JS)
- AJAX средствами Ruby on Rails

МГТУ им. Н.Э. Баумана, доц. каф. ИУ-6,
к.т.н. Самарев Роман Станиславович

samarev@acm.org

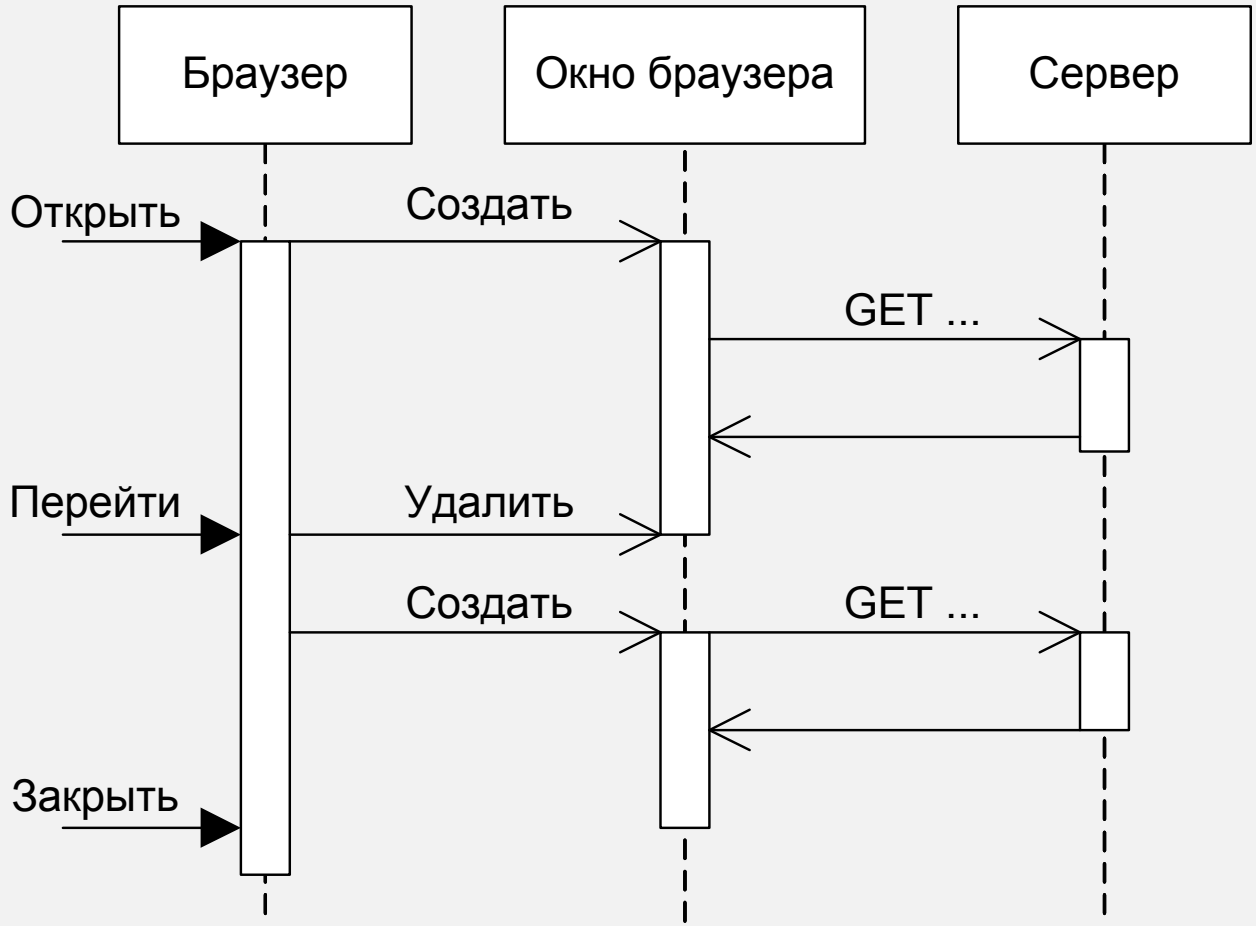
Взаимодействие

- HTTP, FTP, SOAP, NNTP, SMTP, POP3…



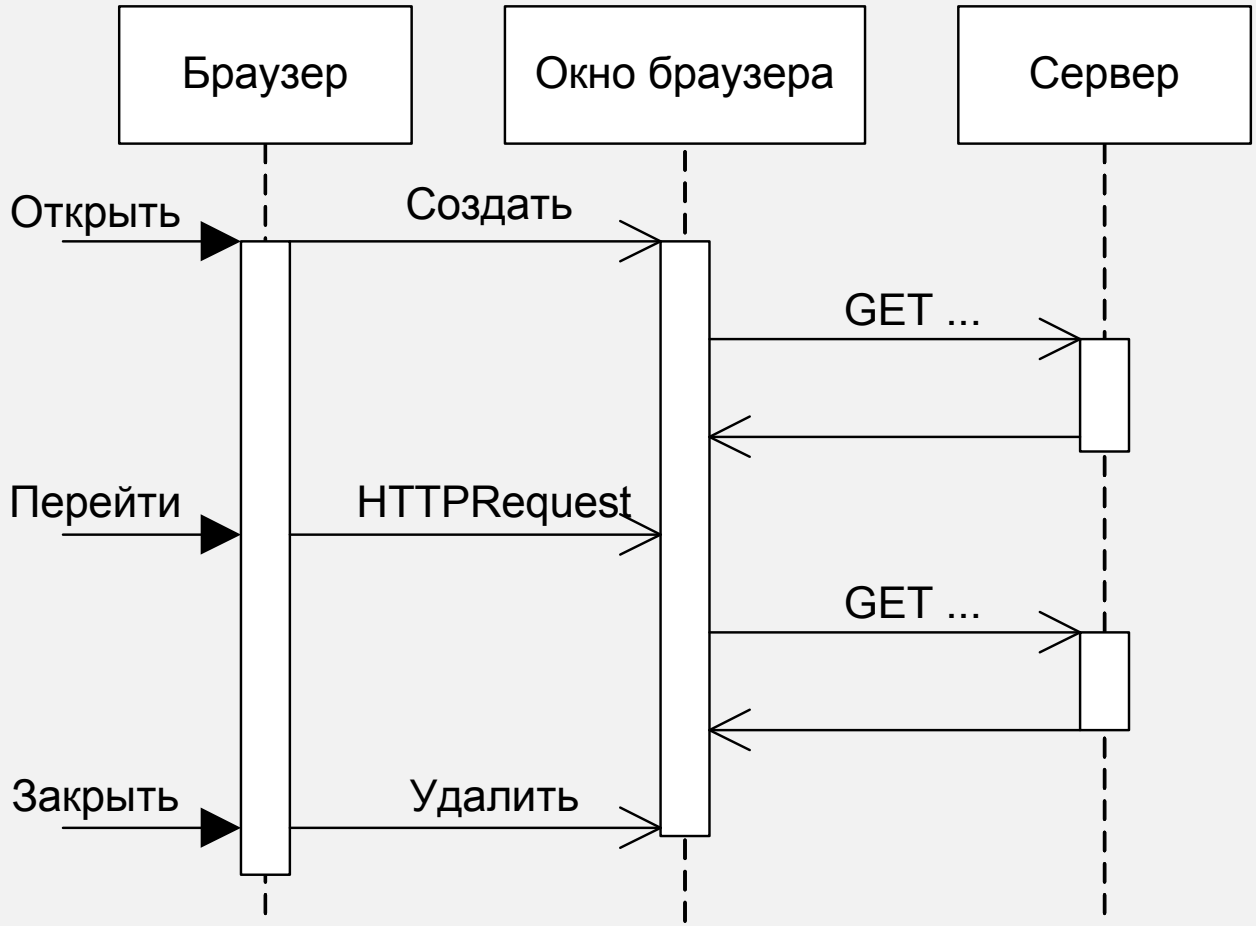
Принципы **синхронного** обмена данными

30.10.18



Принципы **асинхронного** обмена данными

30.10.18



AJAX
Asynchronous JavaScript and XML

30.10.18

- Технология, которая включает:
  - **HTML** (или XHTML) + **CSS** для представления
  - The Document Object Model (**DOM**) для динамического отображения и взаимодействия с данными
  - **XML** для обмена данными, **XSLT** для манипулирования данными
  - Объект **XMLHttpRequest** для асинхронного взаимодействия
  - **JavaScript** для связывание выше указанного

XML
eXtensible Markup Language

30.10.18

- derived from SGML (ISO 8879)
- 1998 - XML 1.0 Specification
- 2004 - XML 1.1

- http://www.w3.org/XML/

XML. Терминология

30.10.18

- **well-formed**
  - Содержит только допустимые Unicode-символы.
  - Нет спец. символов "<", "&" кроме тэгов и спец. разметки.
  - Корректные пары тегов или самозакрывающиеся тэги.
  - Полное соответствие регистра в именах тэгов. Имена тэгов не могут содержать !"#$%&'()*+,/;<=>? @[\]^`{|}~ и пробел, не могут начинаться с ‘-’, ‘.’, ‘,’ или цифры.
  - Только один корневой элемент.

- **valid XML** = well-formed + DTD / XML Schema

Well-formed XML

30.10.18

- **example.xml**

```
<?xml version="1.0" encoding="UTF-8"?>

<people_list>
    <person>
        <name>Fred Bloggs</name>
        <birthdate>27/11/2008</birthdate>
        <gender>Male</gender>
</person>
</people_list>
```

- http://ru.wikipedia.org/wiki/DTD

Valid XML

30.10.18

- **example.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE people_list SYSTEM "example.dtd">
<people_list>
    <person>
        <name>Fred Bloggs</name>
        <birthdate>27/11/2008</birthdate>
        <gender>Male</gender>
</person>
</people_list>
```

- http://ru.wikipedia.org/wiki/DTD

Valid XML
DTD-спецификация

30.10.18

- **example.dtd**

```
<!ELEMENT people_list (person*)>
<!ELEMENT person (name, birthdate?, gender?,
   socialsecuritynumber?)>
<!ELEMENT name (#PCDATA) >
<!ELEMENT birthdate (#PCDATA) >
<!ELEMENT gender (#PCDATA) >
```

XML Schema
http://www.w3.org/XML/Schema

30.10.18

- **country.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<страна xmlns="http://www.bmstu.ru"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.bmstu.ru/country.xsd">
    <название>Франция</название>
    <население>59.7</население>
</страна>
```

XML Schema
http://www.w3.org/XML/Schema

30.10.18

- **country.xsd**

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="страна" type="страна"/>
  <xs:complexType name="страна">
    <xs:sequence>
      <xs:element name="название" type="xs:string"/>
      <xs:element name="население" type="xs:decimal"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

- http://ru.wikipedia.org/wiki/XML_Schema

Модели разбора XML

30.10.18

- Document Object Model (DOM)
  - W3C
  - Весь документ загружается сразу
  - Возможно позиционирование по предкам/потомкам
  - Есть языки запросов XPath, XQuery
- SAX (Simple API for XML)
  - Обработка по мере появления данных
  - Не хранит документ в целом
  - Нет возможности перемещения по элементам

XML и Javascript

30.10.18

```
txt="<note>" + "<to>Tove</to>" +
  "<from>Jani</from>" +
  "<heading>Reminder</heading>"+
  "<body>Don't forget me this
     weekend!</body>"+
  "</note>";

if (window.DOMParser) {
  parser=new DOMParser();
  xmlDoc=
     parser.parseFromString(txt,"text/xml");
} else { // Internet Explorer
   xmlDoc=new
     ActiveXObject("Microsoft.XMLDOM");
   xmlDoc.async=false;
   xmlDoc.loadXML(txt);
}
```

```
document.getElementById("to").innerHTML=
     xmlDoc.getElementsByTagName("to")[0].
          childNodes[0].nodeValue;

document.getElementById("from").innerHTML=
     xmlDoc.getElementsByTagName("from")[0].
          childNodes[0].nodeValue;

document.getElementById("message").innerHTML=
     xmlDoc.getElementsByTagName("body")[0].
          childNodes[0].nodeValue;
```

- http://www.w3schools.com/xml/xml_dom.asp

jQuery.parseXML

30.10.18

```
var xml = "<rss version='2.0'><channel><title>RSS
    Title</title></channel></rss>",
    xmlDoc = $.parseXML( xml ),
    $xml = $( xmlDoc ),
    $title = $xml.find( "title" );

/* append "RSS Title" to #someElement */
$( "#someElement" ).append( $title.text() );

/* change the title to "XML Title" */
$title.text( "XML Title" );

/* append "XML Title" to #anotherElement */
$( "#anotherElement" ).append( $title.text() );
```

- http://api.jquery.com/jQuery.parseXML/

JSON
JavaScript Object Notation

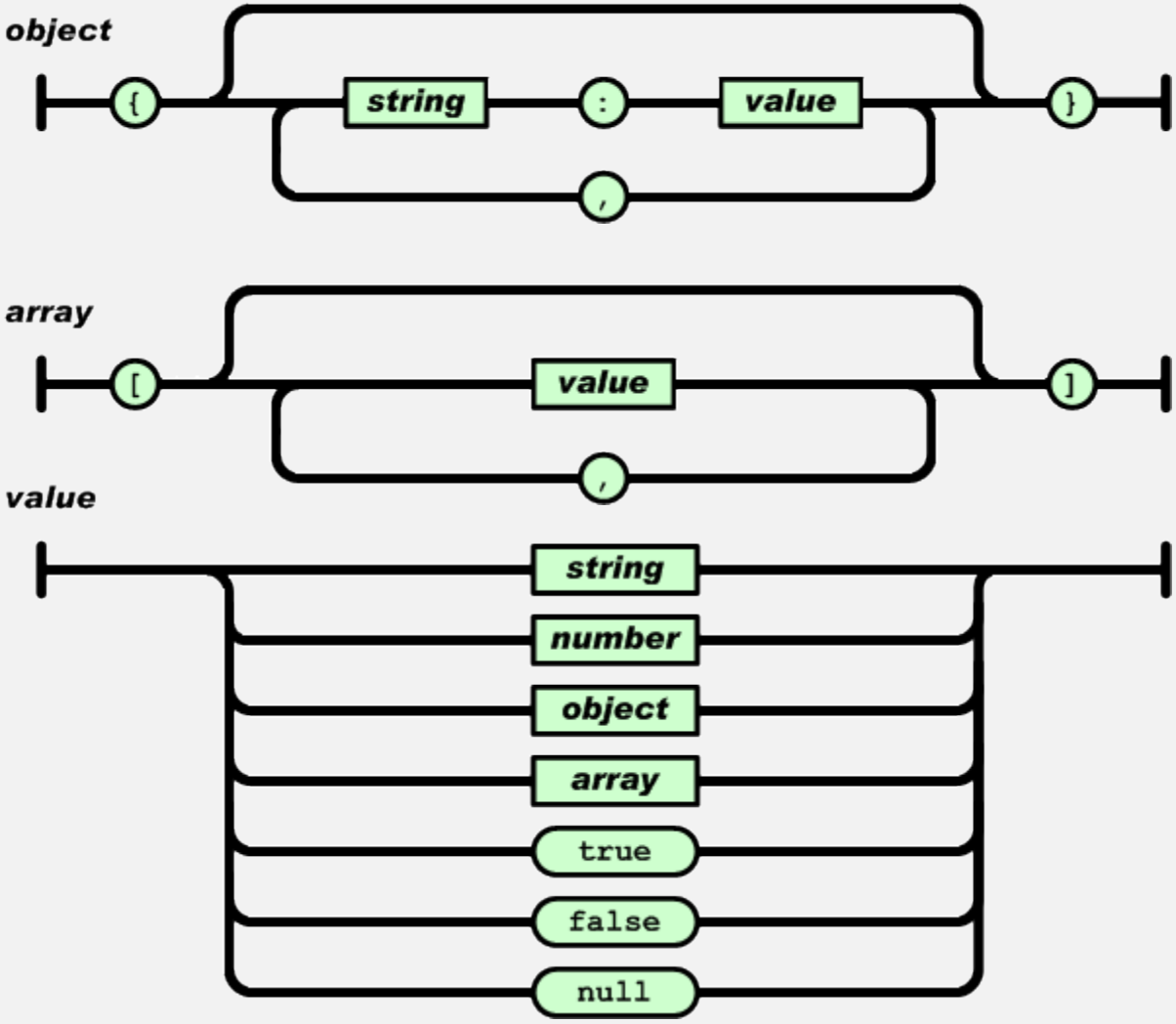
30.10.18

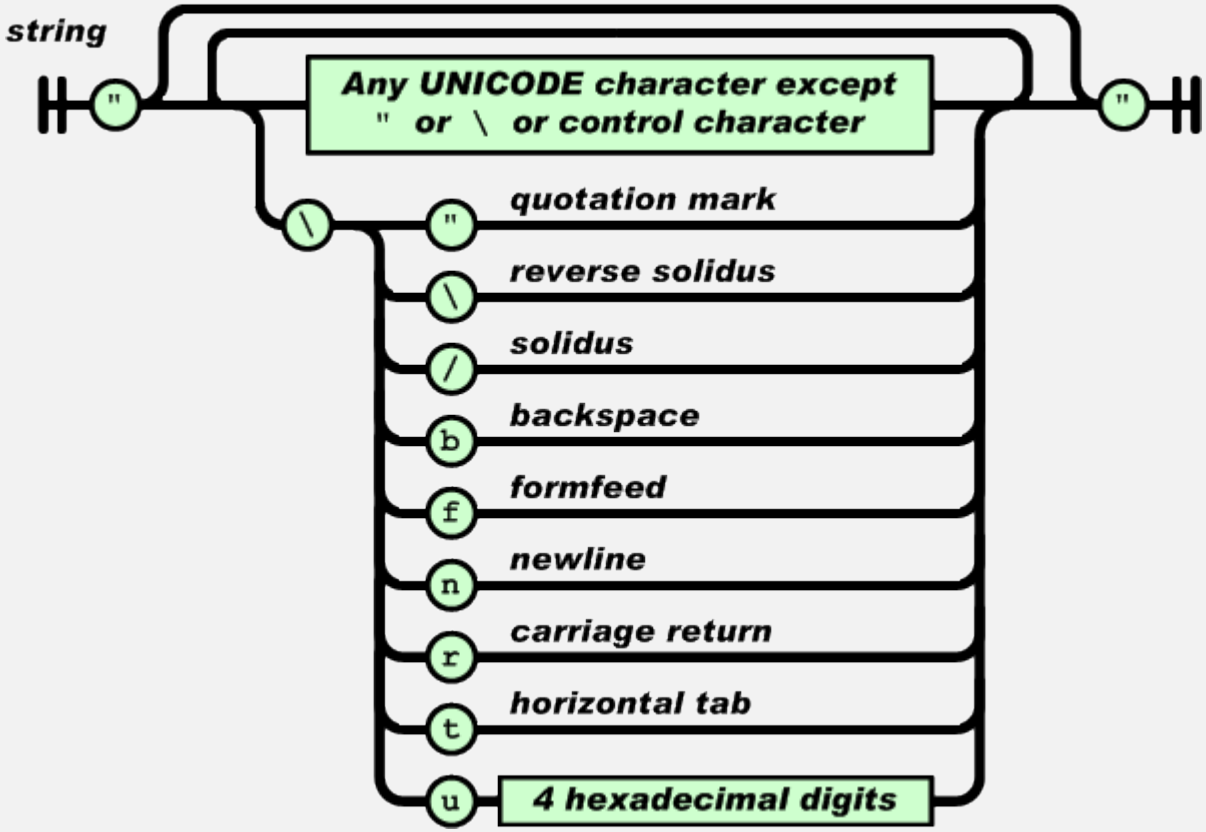
- Подможество языка Javascript

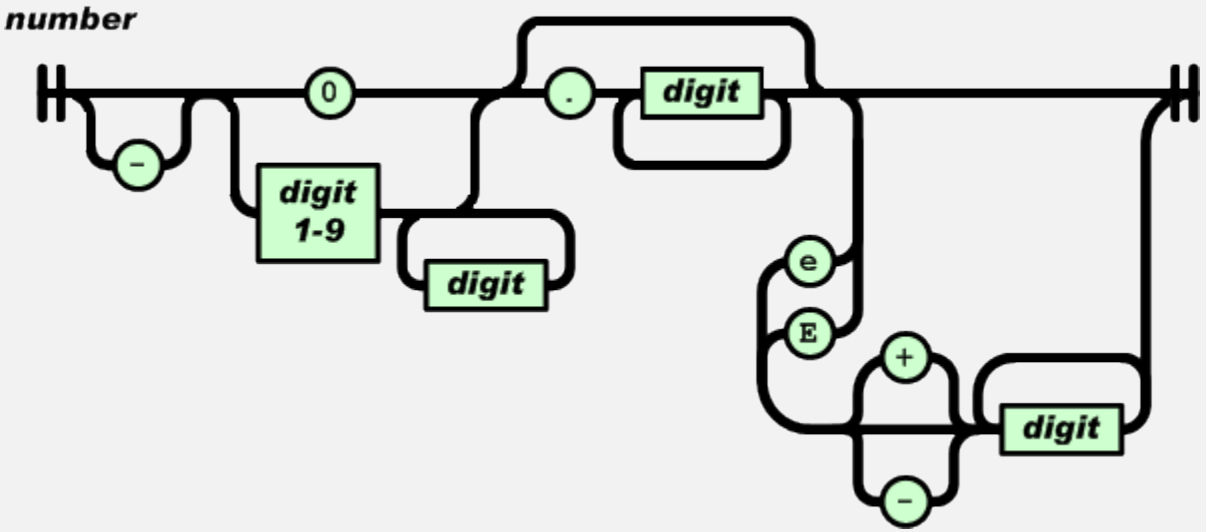
```
{
    "id": 1,
    "name": "Foo",
    "price": 123,
    "tags": [ "Bar", "Eek" ],
    "stock": {
        "warehouse": 300,
        "retail": 20
    }
}
```

- http://json.org
- http://en.wikipedia.org/wiki/JSON

JSON - http://json.org
JavaScript Object Notation




string
"
Any UNICODE character except
" or \ or control character
"
\
"
quotation mark
\
reverse solidus
/
solidus
b
backspace
f
formfeed
n
newline
r
carriage return
t
horizontal tab
u
4 hexadecimal digits


number
0
digit
1-9
digit
.
digit
e
E
+
-
digit

JSON Schema (draft…)

30.10.18

```
{
    "description":"A person",
    "type":"object",

    "properties":{
      "name":{"type":"string"},
      "age" :{
          "type":"integer",
          "maximum":125
      }
    }
  }
```

- http://json-schema.org/latest/json-schema-core.html

JSON и Javascript

30.10.18

- ECMA-262
  - Объект JSON
    - JSON.stringify(obj) - объект в строку JSON
    - JSON.parse(str) – строку JSON в объект

- Разбор через eval на примере JSON.parse

```
JSON.parse = JSON.parse || function (str) {
    if (str === "") str = """";
    eval("var p=" + str + ";");
    return p;
};
```

Асинхронные запросы в Javascript

30.10.18

- Объект XMLHttpRequest

```
xmlhttp=new XMLHttpRequest();
```

- Старые Internet Explorer (IE5, IE6) используют ActiveX Object:

```
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
```

Асинхронный запрос JSON

30.10.18

```
var my_JSON_object = {};
var http_request = new XMLHttpRequest();
http_request.open("GET", url, true);
http_request.onreadystatechange = function () {
    // этот участок кода будет вызван только после отправки данных
    var done = 4, ok = 200;
    if (http_request.readyState == done && http_request.status == ok) {
        my_JSON_object = JSON.parse(http_request.responseText);
        ... // вызываем обработку для my_JSON_object
    }
};
http_request.send(null);
```

- http://en.wikipedia.org/wiki/JSON

Ruby on Rails

30.10.18

- Встроенное преобразование в JSON и YAML
- ActiveSupport
- ActiveRecord

Сериализация объектов Ruby Hash в JSON

30.10.18

- Преобразование Hash в JSON

```
b = { "data" =>
            { "string" => "somestring",
              "number" => "123" } }
puts b.to_json
```

- Обратное преобразование

```
JSON.parse('...')
```

- Поддержка типов String, Number, Array, Hash

- http://www.skorks.com/2010/04/serializing-and-deserializing-objects-with-ruby/

Сериализация объектов в JSON через ActiveSupport

30.10.18

- ActiveSupport::JSON
  - decode
  - encode

- Поддержка типов: String, Symbol, Date, Time, DateTime, Enumerable, Array, Hash, FalseClass, TrueClass, NilClass, Numeric, Float, Integer, Regexp, Object

- http://www.simonecarletti.com/blog/2010/04/inside-ruby-on-rails-serializing-ruby-objects-with-json/

Сериализация объектов в JSON

30.10.18

- ActiveModel::Serializers::JSON

```
user = User.find(1)
user.as_json
# => { "user": {"id": 1, "name": "Yota Mizumi", "age": 26,
                "created_at": "2016/08/01", "awesome": true} }

ActiveRecord::Base.include_root_in_json = false
user.as_json
# => {"id": 1, "name": "Yota Mizumi", "age": 26,
      "created_at": "2016/08/01", "awesome": true}
```

Генерация произвольных данных

30.10.18

- xml_str = "<document></document>“
- json_str = "{'fname':'Victor', " +
  " 'lname':'Frankenstein'}"
- json_prepro_str = "{'head':**tr**('Quick Setup'),"+

  "subhead':'**ZVMODELVZ**'}"

Ruby on Rails
Формирование JSON

30.10.18

```
def index
  @users = User.all

  respond_to do |format|
    format.html # index.html.erb
    format.json { render json: @users }
    format.xml { render xml: @users }
    format.any { render inline: 'Use json or xml!' }
  end
end
```

Использование шаблона

30.10.18

- erb
- http://builder.rubyforge.org/
- http://xtemplate.sourceforge.net/
- …

Формирование XML в ActiveModel

30.10.18

- ActiveModel::Serializers::Xml

```
User.find_by_login('obie').to_xml
```

```
User.find_by_login('obie').
    to_xml(:only => [:email, :login])
```

```
User.find_by_login('obie').
    to_xml(:only => [:email, :login], :skip_instruct => true)
```

Переопределение метода to_xml

30.10.18

```
class User < ActiveRecord::Base
...
  def to_xml(options = {})
    xml = options[:builder] || Builder::XmlMarkup.new(options)
    xml.instruct! unless options[:skip_instruct]
    xml.user do
    xml.tag!(:email, email)
    end
  end
...
end
```

Builder::XmlMarkup

30.10.18

```
require 'builder'

favorites = {
  'candy' => 'Neccos', 'novel' => 'Empire of the Sun', 'holiday' => 'Easter'
}

xml = Builder::XmlMarkup.new( :target => $stdout, :indent => 2 )

xml.instruct! :xml, :version => "1.1", :encoding => "US-ASCII"

xml.favorites do
 favorites.each do | name, choice |
  xml.favorite( choice, :item => name )
 end
end
```

Ruby XML
Парсеры

30.10.18

- Nokogiri (использует libxml, libxslt)
  - http://nokogiri.org/
  - gem install nokogiri
  -
- REXML – медленный, но встроенный
- Ox
- OGA
- libxml-ruby – интерфейс к библиотеке на C
- ...

- http://pascalbetz.github.io/ruby/benchmark/2016/02/12/xml-anyone/
- http://www.ohler.com/dev/xml_with_ruby/xml_with_ruby.html

Nokogiri::XML::Reader

30.10.18

- Быстрый алгоритм разбора
- Пригоден только для чтения

```
reader = Nokogiri::XML::Reader(<<-eoxml)
  <x xmlns:tenderlove='http://tenderlovemaking.com/'>
    <tenderlove:foo awesome='true'>snuggles!</tenderlove:foo>
  </x>
eoxml

reader.each do |node|
  # node is an instance of Nokogiri::XML::Reader
  puts node.name
end
```

- http://www.rubydoc.info/github/sparklemotion/nokogiri/Nokogiri/XML/Reader

Nokogiri::XML

30.10.18

- Медленный разбор
- Позволяет модифицировать элементы

```
doc = Nokogiri::XML <<-EOXML
  <root>
    <car xmlns:part="http://general-motors.com/">
    <part:tire>Michelin Model XGV</part:tire>
  </car>
  <bicycle xmlns:part="http://schwinn.com/">
    <part:tire>I'm a bicycle tire!</part:tire>
  </bicycle>
  </root>
EOXML
```

- doc.xpath('//tire').to_s # => ""

```
doc.xpath('//part:tire', 'part' => 'http://general-motors.com/').to_s
    # => "<part:tire>Michelin Model XGV</part:tire>“
…
```

- http://www.nokogiri.org/tutorials/parsing_an_html_xml_document.html

Nokogiri::HTML

30.10.18

```
page = Nokogiri::HTML(open(PAGE_URL))
news_links = page.css('a').select do |link|
   link['data-category'] == "news“
end
news_links.each { |link| puts link['href'] }

#=>   http://reddit.com
#=>   http://www.nytimes.com

puts news_links.class   #=>   Array
```

- http://ruby.bastardsbook.com/chapters/html-parsing/

Ненавязчивый (Unobtrusive) JavaScript

30.10.18

- Предположим, что необходимо окрасить текст с помощью JS

```
<a href="#" onclick="this.style.backgroundColor='#990000'">Paint it red</a>
<a href="#"
    onclick="this.style.backgroundColor='#009900';this.style.color='#FFFFFF';">
    Paint it green</a>
```

- Возможное улучшение (код на CoffeeScript)

```
paintIt = (element, backgroundColor, textColor) ->
  element.style.backgroundColor = backgroundColor
  if textColor?
    element.style.color = textColor
```

```
<a href="#" onclick="paintIt(this, '#990000')">Paint it red</a>
<a href="#" onclick="paintIt(this, '#009900', '#FFFFFF')">Paint it green</a>
<a href="#" onclick="paintIt(this, '#000099', '#FFFFFF')">Paint it blue</a>
```

Разделение HTML и Javasctipt

30.10.18

- Код на CoffeeScript

```
paintIt = (element, backgroundColor, textColor) ->
  element.style.backgroundColor = backgroundColor
  if textColor?
    element.style.color = textColor

$ ->
  $("a[data-background-color]").click ->
    backgroundColor = $(this).data("background-color")
    textColor = $(this).data("text-color")
    paintIt(this, backgroundColor, textColor)
```

- Разметка

```
<a href="#" data-background-color="#990000">Paint it red</a>
<a href="#" data-background-color="#009900" data-text-color="#FFFFFF">Paint it green</a>
<a href="#" data-background-color="#000099" data-text-color="#FFFFFF">Paint it blue</a>
```

AJAX в Ruby on Rails

30.10.18

- Ajax-запрос на CoffeeScript с использованием встроенной jQuery

```
$.ajax(url: "/test").done (html) ->
  $("#results").append html
```

- http://edgeguides.rubyonrails.org/working_with_javascript_in_rails.html
- http://rusrails.ru/working-with-javascript-in-rails

Встроенные «хелперы»
Rails "Ajax helpers" = JavaScript + Ruby

30.10.18

- Шаблон erb генерирует следующий код

```
<%= form_for(@post, remote: true) do |f| %>
 ...
<% end %>
```

```
<form accept-charset="UTF-8" action="/posts" class="new_post" data-remote="true"
      id="new_post" method="post">
 ...
</form>
```

- Явная регистрация асинхронных событий (CoffeeScript, jQuery)

```
$(document).ready ->
    $("#new_post").on("ajax:success", (event) ->
        [data, status, xhr] = event.detail
        $("#new_post").append xhr.responseText
  ).on "ajax:error", (event) ->
        $("#new_post").append "<p>ERROR</p>"
```

Особенности регистрации событий

30.10.18

- Rails < 5.1 with jquery-ujs

```
function handleAjaxSuccess(event, data, status, xhr) { … }
$(document).on('page:load', function {
        $("#new_post").on('ajax:success', handleAjaxSuccess)
}
```

- Rails ≥ 5.1

```
handleAjaxSuccess = function(event) {
    [data, status, xhr] = event.detail;
    ....
}


document.addEventListener('DOMContentLoaded', function(){
        document.querySelector("#new_post").addEventListener(
                'ajax:success', handleAjaxSuccess)
})
```

- http://guides.rubyonrails.org/working_with_javascript_in_rails.html

Дополнительные элементы

30.10.18

- <%= link_to "a post", @post, remote: true %>

```
<a href="/posts/1" data-remote="true">a post</a>
```

```
$ ->
  $("a[data-remote]").on "ajax:success", (event) ->
    alert "The post was deleted."
```

- <%= button_to "A post", @post, remote: true %>

```
<form action="/posts/1" class="button_to" data-remote="true"
    method="post">
  <div><input type="submit" value="A post"></div>
</form>
```

Встраивание AJAX в Rails приложение (JS как ответ)

30.10.18

- Создаём приложение

rails new ajax_test

- Создаём контроллер

rails g controller User index create

- Создаём модель

rails g model User name

UserController

30.10.18

```
class UserController < ApplicationController
  def index;     @users = User.all;     @user = User.new;   end

  def create
    @user = User.new user_params
    respond_to do |format|
      if @user.save
        format.html { redirect_to @user, notice: 'User was successfully created.' }
        format.js   {}  # response with javascript action
        format.json { render json: @user, status: :created, location: @user }
      else
        format.html { render action: “index" }
        format.json { render json: @user.errors, status: :unprocessable_entity }
      end
    end
  end
private
  def user_params; params.require(:user).permit(:name) end
end
```

Представления

30.10.18

- app/views/user/index.html.erb

```
<b>Users</b>
<ul id="users">
<% @users.each do |user| %>
  <%= render 'user', user: user %>
<% end %>
</ul> <br>
<%==== # Indirect async handler registration + Accept: text/javascript %>
<%= form_for(@user, :url => { :action => "create" }, remote: true) do |f| %>
  <%= f.label :name %><br>
  <%= f.text_field :name %>
  <%= f.submit %>
<% end %>
```

- app/views/user/create.js.erb

```
$("<%= escape_javascript(render 'user', user: @user) %>").appendTo("#users");
```

- app/views/user/_user.html.erb

```
<li><%= user.name %></li>
```

Изменение маршрутов

30.10.18

- config/routes.rb

```
get "user/index"
get "user/create"
post "user/create", as: 'users'
root "user#index"
```

- Запускаем приложение и проверяем результат

```
rake db:migrate
rails server
```

Встраивание AJAX в Rails приложение (AJAX обработчик)

30.10.18

- app/assets/javascripts/user.js

```
handleAjaxSuccess = function(event) {
    [data, status, xhr] = event.detail;
    var list = document.querySelector("#users");
    var li = document.createElement("li");
    li.textContent = data.name;
    list.appendChild(li);
}
document.addEventListener('DOMContentLoaded', function(){
        document.querySelector("#new_user").addEventListener(
            'ajax:success', handleAjaxSuccess)

})
```

Представления

30.10.18

- app/views/user/index.html.erb

```
<b>Users</b>
<ul id="users">
<% @users.each do |user| %>
  <%= render 'user', user: user %>
<% end %>
</ul> <br>


<%= form_for(@user, :url => { :action => "create" },
      html: {:'data-type' => 'json'} # явно указываем требуемый формат
      remote: true) do |f| %>
  <%= f.label :name %><br>
  <%= f.text_field :name %>
  <%= f.submit %>
<% end %>
```

- app/views/user/_user.html.erb

```
<li><%= user.name %></li>
```

Turbolinks

30.10.18

- Активен, если в Gemfile прописано gem turbolinks
- Работает в браузерах с поддержкой PushState
- Подменяет полную перезагрузку заменой содержимого страницы
- Автоматически присоединяется к <a>
- Исключение
  No turbolinks here.

- Событие окончания загрузки страницы без Turbolinks

```
$(document).ready ->
  alert "page has loaded!"
```

- Событие окончания загрузки страницы с Turbolinks

```
$(document).on "page:change", ->
  alert "page has loaded!"
```

- https://github.com/turbolinks/turbolinks

Литература

30.10.18

- https://www.w3schools.com/js/js_ajax_intro.asp
- https://www.w3schools.com/xml/xml_whatis.asp
- http://edgeguides.rubyonrails.org/working_with_javascript_in_rails.html
- http://rusrails.ru/working-with-javascript-in-rails

30.10.18

Система контроля версий Git

30.10.18

- Позволяет не потерять когда-либо сделанные изменения
- Удобна для выделения изменений в процессе выполнения лабораторных работ
  - git init
  - git add
  - git commit
  - git diff

- https://ru.wikibooks.org/wiki/Ruby_on_Rails/Введение