

# "Аяксификация" корзины:

- поместим информацию о корзине на боковую панель каталога
- применим AJAX-технологии для обновления корзины на боковой панели без повторного отображения всей страницы

# Шаг Г1: перемещение корзины

При помощи **шаблонных фрагментов** перенесём обслуживание корзины в макет, отображающий общий каталог. **Фрагмент** – часть представления, размещённая в собственном файле.

Перепишем представление корзины:

```
app/views/store/add_to_cart.rhtml
```

```
<div class="cart-title">Your Cart</div>
```

```
<table>
```

```
<%= render(:partial => "cart_item", :collection => @cart.items) %>
```

```
<tr class="total-line">
```

```
<td colspan="2">Total</td>
```

```
<td class="total-cell"><%= number_to_currency(@cart.total_price) %></td>
```

```
</tr>
```

```
</table>
```

```
<%= button_to "Empty cart", :action => :empty_cart %>
```

Фрагмент сохраняем в файле `_cart_item.rhtml` каталога `app/views/store`

***app/views/store/\_cart\_item.rhtml***

```
<tr>
```

```
  <td><%= cart_item.quantity %>&times;</td>
```

```
  <td><%= h(cart_item.title) %></td>
```

```
  <td class="item-price"><%= number_to_currency(cart_item.price) %></td>
```

```
</tr>
```

Т.о. мы убрали отображение корзины.

При вызове фрагмента мы можем использовать метод **render**. Параметр `:object` захватывает объект, который присвоен локальной переменной с тем же именем, что и у фрагмента. Поэтому в макете можем воспользоваться **ВЫЗОВОМ**

`<% render(:partial => "cart" , :object => @cart) %>` и в шаблоне `_cart.rhtml` можем сослаться на корзину с помощью переменной `cart`.

Для этого:

1) создадим шаблон `_cart.rhtml`

***app/views/store/\_cart.rhtml***

```
<div class="cart-title">Your Cart</div>
```

```
<table>
```

```
<%= render(:partial => "cart_item", :collection => cart.items) %>
```

```
<tr class="total-line">
```

```
<td colspan="2">Total</td>
```

```
<td class="total-cell"><%= number_to_currency(cart.total_price) %></td>
```

```
</tr>
```

```
</table>
```

```
<%= button_to "Empty cart", :action => :empty_cart %>
```

2) внесём изменения в макет store и включим в его боковую панель новый фрагмент

### ***app/views/layouts/store.rhtml***

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
  <title>Pragprog Books Online Store</title>
  <%= stylesheet_link_tag "depot", :media => "all" %>
</head>
<body id="store">
  <div id="banner">
    <%= image_tag("logo.png") %>
    <%= @page_title || "Pragmatic Bookshelf" %>
  </div>
  <div id="columns">
    <div id="side">

    <div id="cart">
      <%= render(:partial => "cart", :object => @cart) %>
    </div>
```

```
<a href="http://www....">Home</a><br />
  <a href="http://www..../faq">Questions</a><br />
  <a href="http://www..../news">News</a><br />
  <a href="http://www..../contact">Contact</a><br />
</div>
<div id="main">
```

```
<% if flash[:notice] -%>
  <div id="notice"><%= flash[:notice] %></div>
<% end -%>
```

```
  <%= yield :layout %>
</div>
</div>
</body>
</html>
```

3) внесём изменения в контроллер магазина store (в действии контроллера index нужно определить @cart)

*app/controllers/store\_controller.rb*

```
def index
```

```
  @products = Product.find_products_for_sale
```

```
  @cart = find_cart
```

```
end
```

Изменим работу кнопки «Добавить в корзину» - вместо отображения страницы корзины ей нужно будет обновить главную страницу каталога. В конце действия `add_to_cart` браузер перенаправляется обратно на каталог.

```
app/controllers/store_controller.rb
```

```
def add_to_cart
  begin
    product = Product.find(params[:id])
  rescue ActiveRecord::RecordNotFound
    logger.error("Attempt to access invalid product #{params[:id]}")
    redirect_to_index("Invalid product")
  else
    @cart = find_cart
    @cart.add_product(product)
    redirect_to_index
  end
end
```

Чтобы этот код работал, нужно изменить определение метода `redirect_to_index` и сделать параметр сообщения необязательным.

```
app/controllers/store_controller.rb
```

```
def redirect_to_index(msg = nil)
  flash[:notice] = msg if msg
  redirect_to :action => :index
end
```

Шаблон `add_to_cart.rhtml` можно удалить.



# Шаг Г2: корзина, использующая технологию AJAX

Сначала изменим страницу каталога, заставив её посылать AJAX-запрос на серверное приложение и заставив приложение ответить HTML-фрагментом, содержащим обновлённую корзину.

Заменяем метод `button_to` на следующий код:

***app/views/store/index.rhtml***

```
<% form_remote_tag :url => { :action => :add_to_cart, :id => product } do %>
  <%= submit_tag "Add to Cart" %>
<% end %>
```

С помощью параметра `url` мы сообщаем `form_remote_tag` способ вызова серверного приложения. Код между `do` и `end` – тело формы.

Нужно приспособить браузер к отправке AJAX-запроса нашему приложению.

```
app/views/layout/store.rhtml
```

```
<html>
<head>
  <title>Pragprog Books Online Store</title>
  <%= stylesheet_link_tag "depot", :media => "all" %>
  <%= javascript_include_tag :defaults %>
</head>
```

Приложение также должно возвращать ответ.

Действием `add_to_cart` прекращаем перенаправление на отображение каталога

```
app/controllers/store_controller.rb
```

```
def add_to_cart
  begin
    product = Product.find(params[:id])
  rescue ActiveRecord::RecordNotFound
    logger.error("Attempt to access invalid product #{params[:id]}")
    redirect_to_index("Invalid product")
  else
    @cart = find_cart
    @cart.add_product(product)
  end
end
```

Когда метод `add_to_cart` завершит обработку AJAX-запроса, Rails для отображения страницы станет искать шаблон `add_to_cart`, а мы его удалили. Создадим шаблон rjs-шаблон (`.rjs` шаблоны – способ передачи браузеру кода JavaScript, который целиком выполняется созданием кода на стороне сервера).

```
app/views/store/add_to_cart.rjs
```

```
page.replace_html("cart", :partial => "cart", :object => @cart)
```

Переменная `page` – экземпляр JavaScript-генератора (Rails-класса, который создаёт код JavaScript на сервере. Чтобы он выполнялся браузером). Этот шаблон представляет HTML-код для отображения корзины и предписывает браузеру заменить содержимое контейнера `<div>` с параметром `id="cart"` этим HTML-кодом.

Т.о. мы создали AJAX-приложение.

# Возможные проблемы

- Вы удалили старый файл `add_to_cart.rhtml`?
- Вы не забыли включить библиотеки JavaScript в макет store (используя `javascript_include_tag`)?
- Пора сделать полную перезагрузку страницы?
- Вы получали какие-нибудь сообщения об ошибках? Посмотрите файл `development` в каталоге `logs`
- Если в регистрационном файле нет входящих запросов на выполнение действия `add_to_cart`, то браузер не создаёт AJAX-запросов. У браузера может быть отключено выполнение JavaScript (посмотреть сгенерированный HTML-код)
- Остановить и снова запустить приложение
- Поместить в макет заголовок  
`<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">`,  
чтобы переключить IE в стандартный режим, который лучше работает с AJAX-содержимым.

# Вопросы

- Что такое шаблонный фрагмент?
- Как работает метод `render`?
- Каким образом мы ссылаемся на текущий элемент корзины в коде шаблонного фрагмента?
- Какие возможности предоставляет технология AJAX?
- Как приспособить браузер к отправке AJAX-запроса в адрес нашего приложения?
- Для чего нужны шаблоны `.rjs`?
- Что позволяет делать метод `replace_html`?