Hotwire

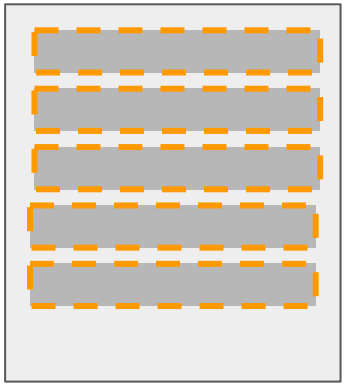
by Jackson Pires

Conhecendo o Turbo Frame

Conhecendo o Turbo Frame

Com o Turbo Drive vimos que ele se encarrega de substituir o header e o body da página, tornando assim a navegação mais fluida, no entanto, ele substitui TODO o body. Agora com o Turbo Frame vamos conseguir atualizar apenas uma parte específica da página. Para isso comece usando o helper `turbo_frame_tag` em conjunto com o helper `dom_id`, conforme a seguir:

Conhecendo o Turbo Frame



```
# user_courses/_user_course.html.erb
<%= turbo_frame_tag dom_id(user_course) do %>
    <p><%= "Usuário: #{user_course.user.name}"%> </p>
  <p><%= "Email: #{user_course.user.email}"%> </p>
  <p><%= "Curso:#{user_course.course.title}"%> </p>
  <p><%= "Url:#{user_course.course.video_url}"%> </p>

  <p>Qtd. de likes: <strong><%= user_course.likes %></strong></p>
  <p><%= Time.now.strftime("%H:%M:%S") %></p>

  <%= button_to 'Like!', like_user_course_path(user_course), method: :post %>
<% end %>
```

Conhecendo o Turbo Frame

- O `dom_id` é um helper que extrai o ID de um objeto para ser usado como identificador do elemento html.
- Verifique a requisição `fetch` e perceba uma header `Turbo-Frame` vai na requisição. É esse header que informa pro turbo qual parte atualizar, no entanto perceba que neste momento a requisição tem um redirect e traz todos os frames. Ajustaremos isso no controller renderizando apenas o que precisamos.

Conhecendo o Turbo Frame

```ruby
# user_courses_controller.rb
def like
  user_course = UserCourse.find(params[:id]).increment!(:likes)
  render user_course
end
```

Conhecendo o Turbo Frame

- Analisando mais uma vez agora o `fetch`, percebemos que apenas o frame específico foi retornado para o update na página.

Saindo de frames

Saindo de frames

- Às vezes precisaremos evitar o comportamento padrão do turbo frame e deixar apenas o Turbo Drive fazer seu papel, para isso, podemos "sair" de um frame propositalmente.
- Para nosso exemplo, vamos adicionar dois links para acesso à ação `show` do usuário e curso.

Saindo de frames

```
<%= turbo_frame_tag dom_id(user_course) do %>
  <p><%= "Usuário: #{user_course.user.name}"%><%= link_to '(ver)',
user_path(user_course.user)%></p>
  <p><%= "Email: #{user_course.user.email}"%> </p>
  <p><%= "Curso:#{user_course.course.title}"%><%= link_to '(ver)',
course_path(user_course.course) %></p>
  <p><%= "Url:#{user_course.course.video_url}"%> </p>

  <p>Qtd. de likes: <strong><%= user_course.likes %></strong></p>
  <p><%= Time.now.strftime("%H:%M:%S") %></p>

  <%= button_to 'Like!', like_user_course_path(user_course), method:
:post %>
<% end %>
```

Saindo de frames

- Perceba que ao acessarmos o link o erro é gerado (inclusive no terminal do dev tools do navegador), indicando que o problema é que não temos um retorno de turbo frame esperado. E de fato, esperamos apenas que o link acessado seja redirecionado em nossa página. Então, para conseguirmos evitar a ação padrão do turbo frame, podemos usar um `data-attribute` especial.

Saindo de frames

```
<%= turbo_frame_tag dom_id(user_course) do %>
  <p><%= "Usuário: #{user_course.user.name}"%> <%= link_to '(ver)',
user_path(user_course.user), data: { turbo_frame: "_top" } %></p>
  <p><%= "Email: #{user_course.user.email}"%> </p>
  <p><%= "Curso:#{user_course.course.title}"%> <%= link_to '(ver)',
course_path(user_course.course), data: { turbo_frame: "_top" } %></p>
  <p><%= "Url:#{user_course.course.video_url}"%> </p>

  <p>Qtd. de likes: <strong><%= user_course.likes %></strong></p>
  <p><%= Time.now.strftime("%H:%M:%S") %></p>

  <%= button_to 'Like!', like_user_course_path(user_course), method:
:post %>
<% end %>
```