

Laboratorio de Computación IV

Clase 22

Andrés Fortier

Edición anidada de relaciones 1:N

```
$ bin/rails g model phone phone_type:string number:string  
profile:references  
$ bin/rake db:migrate
```

```
/app/models/phone.rb
```

```
class Phone < ActiveRecord::Base  
  belongs_to :profile  
end
```

```
/app/models/profile.rb
```

```
class Profile < ActiveRecord::Base  
  belongs_to :user  
  has_many :phones, :dependent => :destroy  
  accepts_nested_attributes_for :phones  
end
```

Edición anidada de relaciones 1:N

- Comencemos por crear un teléfono falso para ver la edición

```
/app/controllers/profiles_controller.rb
```

```
class ProfilesController < ApplicationController
  ...
  def edit
    @profile = current_user.profile
    authorize @profile
    if @profile.phones.empty?
      @profile.phones << Phone.new(
        phone_type: 'Mobile',
        number: '(011) 154789345634')
    end
  end
  ...
end
```

Edición anidada de relaciones 1:N

- Tenemos que permitir los atributos

```
/app/controllers/profiles_controller.rb
```

```
def profile_params
  params
    .require(:profile)
    .permit(
      :first_name,
      :last_name,
      :date_of_birth,
      phones_attributes: [
        :id,
        :phone_type,
        :number
      ])
end
...
end
```

Edición anidada de relaciones 1:N

- Modifiquemos la vista de edición

```
/app/views/profiles/edit.html.erb
```

```
...  
<div class="panel panel-default">  
  <div class="panel-heading">  
    <h3 class="panel-title">Phones</h3>  
  </div>  
  <div class="panel-body">  
    <%= f.fields_for :phones do | form_builder | %>  
      <%= render "phone_fields", :f => form_builder %>  
    <% end %>  
  </div>  
</div>  
<p>  
  <%= f.submit 'Update', :class => 'btn btn-primary' %>  
</p>  
<% end %>
```

Edición anidada de relaciones 1:N

- Y agreguemos el *partial* `_phone_fields`

```
/app/views/profiles/_phone_fields.html.erb
```

```
<div class="form-inline">
  <div class="form-group">
    <%= f.label :phone_type %>
    <%= f.select(:phone_type,
                ['Mobile', 'Work', 'Home'],
                {},
                {:class => 'form-control'}) %>
  </div>
  <div class="form-group">
    <%= f.label :number %>
    <%= f.text_field :number, class: 'form-control' %>
  </div>
</div>
```

Edición anidada de relaciones 1:N

- Prueben la edición.
- Nos falta agregar y eliminar en forma dinámica.

Edición anidada de relaciones 1:N

- Veamos como eliminar
 - ``_destroy`` como flag para que AR lo elimine.

```
/app/models/profile.rb
```

```
class Profile < ActiveRecord::Base
  belongs_to :user
  has_many :phones, :dependent => :destroy
  accepts_nested_attributes_for :phones,
    :allow_destroy => true
end
```


Edición anidada de relaciones 1:N

```
/app/controllers/profiles_controller.rb
```

```
def profile_params
  params
    .require(:profile)
    .permit(
      :first_name,
      :last_name,
      :date_of_birth,
      phones_attributes: [
        :id,
        :phone_type,
        :number,
        :_destroy
      ]
    )
end

...
end
```

Edición anidada de relaciones 1:N

```
/app/views/profiles/_phone_fields.html.erb
```

```
...  
<div class="form-group">  
  <%= f.label :number %>  
  <%= f.text_field :number, class: 'form-control' %>  
</div>  
<div class="form-group">  
  <%= f.check_box :_destroy %>  
  <%= f.label :_destroy, "Remove" %>  
</div>  
</div>
```

Edición anidada de relaciones 1:N

- Prueben la edición.
- Vuelvan al perfil usando el link “My Profile”.
- Modificaciones en la eliminación
 - En lugar de utilizar un checkbox, utilizar un link o botón.
 - Asignar el valor de un campo `hidden`.
 - Esconder los componentes de la página usando javascript.

Edición anidada de relaciones 1:N

```
/app/views/profiles/_phone_fields.html.erb
```

```
...  
<div class="form-group">  
  <%= f.label :number %>  
  <%= f.text_field :number, class: 'form-control' %>  
</div>  
<div class="form-group">  
  <%= f.hidden_field :_destroy %>  
  <a href="#" onclick="removePhone(this)">Remove</a>  
</div>  
</div>
```

Edición anidada de relaciones 1:N

- Vamos a crear un nuevo archivo para nuestro código Javascript

```
/app/assets/javascripts/phones.js  
function removePhone(linkNode) {  
  console.log(linkNode);  
}
```

```
/app/assets/javascripts/phones.js  
function removePhone(linkNode) {  
  console.log($(linkNode));  
}
```

Edición anidada de relaciones 1:N

```
/app/assets/javascripts/phones.js  
function removePhone(linkNode) {  
    console.log($(linkNode).parent());  
}
```

```
/app/assets/javascripts/phones.js  
function removePhone(linkNode) {  
    var link = $(linkNode);  
    var myFormGroup = link.parent();  
    console.log(myFormGroup.find("input[type=hidden]"));  
}
```

Edición anidada de relaciones 1:N

```
/app/assets/javascripts/phones.js  
function removePhone(linkNode) {  
  var link = $(linkNode);  
  var myFormGroup = link.parent();  
  myFormGroup.find("input[type=hidden]").val("true");  
}
```

- Clickeen en “Remove” y hagan “Update”
 - Vayan a “My Profile” para re-generar el teléfono

Edición anidada de relaciones 1:N

- Nos falta esconder los componentes de ese teléfono
 - Nota: esconder vs eliminar

```
/app/assets/javascripts/phones.js
```

```
function removePhone(linkNode) {  
  var link = $(linkNode);  
  var myFormGroup = link.parent();  
  myFormGroup.find("input[type=hidden]").val("true");  
  myFormGroup.parent().hide();  
}
```


Edición anidada de relaciones 1:N



- Ahora debemos crear nuevas entradas dinámicamente.
- Eliminen todos los teléfonos del perfil.
- Hagamos una pequeña modificación en el controller.

Edición anidada de relaciones 1:N

```
/app/controllers/profiles_controller.rb
class ProfilesController < ApplicationController
  ...
  def edit
    @profile = current_user.profile
    authorize @profile
    if @profile.phones.empty?
      @profile.phones << Phone.new(phone_type: 'Mobile',
number: '11')
      @profile.phones << Phone.new(phone_type: 'Mobile',
number: '22')
      @profile.phones << Phone.new(phone_type: 'Mobile',
number: '33')
    end
  end
  ...
end
```

Edición anidada de relaciones 1:N

- Inspeccionen el html

```
▼ <div class="panel panel-default">
  ▶ <div class="panel-heading">...</div>
  ▼ <div class="panel-body">
    ::before
    ▼ <div id="phones">
      ▶ <div class="form-inline" style="margin: 10px;">...</div>
        <input id="profile_phones_attributes_0_id" name="profile[phones_attributes][0][id]"
      ▶ <div class="form-inline" style="margin: 10px;">...</div>
        <input id="profile_phones_attributes_1_id" name="profile[phones_attributes][1][id]"
      ▶ <div class="form-inline" style="margin: 10px;">...</div>
        <input id="profile_phones_attributes_2_id" name="profile[phones_attributes][2][id]"
      </div>
```

- Y vayan a la consola a ver el update

Edición anidada de relaciones 1:N

- Comencemos por agregar un formulario vacío

```
/app/controllers/profiles_controller.rb
class ProfilesController < ApplicationController
  ...
  def edit
    @profile = current_user.profile
    authorize @profile
  end
  ...
end
```

Edición anidada de relaciones 1:N

```
/app/views/profiles/edit.html.erb
```

```
...  
<div id="phones">  
  <%= f.fields_for :phones do | form_builder | %>  
    <%= render "phone_fields", :f => form_builder %>  
  <% end %>  
</div>  
  
<% new_phone = Phone.new %>  
<%= f.fields_for(:phones, new_phone) do | form_builder |  
  %>  
  <%= render("phone_fields", :f => form_builder) %>  
<% end %>
```

- Por cada update agregamos un teléfono vacío.

Edición anidada de relaciones 1:N

- Tenemos que
 - Agregar con javascript nuevos formularios.
 - Manejar los ids de los componentes para que no se pisen.

```
/app/assets/javascripts/phones.js
```

```
...  
function addPhone($parent) {  
    $parent.append("<h1>Hola</h1>");  
}
```

Edición anidada de relaciones 1:N

```
/app/views/profiles/edit.html.erb
```

```
...  
<div id="phones">  
  <%= f.fields_for :phones do | form_builder | %>  
    <%= render "phone_fields", :f => form_builder %>  
  <% end %>  
</div>  
  
<a href="#" onclick="addPhone($(' #phones '))">Add</a>
```

Edición anidada de relaciones 1:N



- Idea
 - Convertir el resultado del render del partial en un string.
 - Pasarlo como parámetro a la función `addPhone`

Edición anidada de relaciones 1:N

```
/app/assets/javascripts/phones.js
```

```
...  
function addPhone($parent, formHTML) {  
  $parent.append(formHTML);  
}
```

Edición anidada de relaciones 1:N

```
/app/views/profiles/edit.html.erb
```

```
...  
<div id="phones">  
  <%= f.fields_for :phones do | form_builder | %>  
    <%= render "phone_fields", :f => form_builder %>  
  <% end %>  
</div>  
  
<%  
  new_phone = Phone.new  
  fields = f.fields_for(:phones, new_phone) do |fb|  
    render("phone_fields", :f => fb)  
  end  
  js = escape_javascript(fields)  
  fn = html_escape("addPhone($(' #phones' ), \"#{js}\")")  
  concat(raw("<a href=\"#{\" onclick=\"#{fn}\">Add</a>"))  
  %>
```

Edición anidada de relaciones 1:N



- Prueben agregar 3 números
 - Parece funcionar pero sólo agrega uno.
 - ¿Porqué?

Edición anidada de relaciones 1:N

```
/app/views/profiles/edit.html.erb
```

```
...
```

```
<%  
  new_phone = Phone.new  
  fields = f.fields_for(:phones,  
    new_phone,  
    :child_index => "id_placeholder") do |fb|  
    render("phone_fields", :f => fb)  
  end  
  js = escape_javascript(fields)  
  fn = html_escape("addPhone($(' #phones' ), \"#{js}\")")  
  concat(raw("<a href=\"#{\" onclick=\"#{fn}\">Add</a>"))  
  %>
```

Edición anidada de relaciones 1:N

```
/app/assets/javascripts/phones.js
```

```
...  
function addPhone($parent, formHTML) {  
    var new_id = new Date().getTime();  
    var regexp = new RegExp("id_placeholder", "g");  
    var content = formHTML.replace(regexp, new_id)  
    $parent.append(content);  
}
```

- Con esto completamos el edit.

Edición anidada de relaciones 1:N

- Yapa: usar iconos

```
/app/views/profiles/_phone_fields.html.erb
```

```
...  
<div class="form-group">  
  <%= f.hidden_field :_destroy %>  
  <a href="#" class="btn btn-warning"  
onclick="removePhone(this)">  
    <span class="glyphicon glyphicon-minus"></span>  
  </a>  
</div>  
</div>
```

Edición anidada de relaciones 1:N

```
/app/views/profiles/edit.html.erb
```

```
...

<%
  new_phone = Phone.new
  fields = f.fields_for(:phones,
    new_phone,
    :child_index => "id_placeholder") do |fb|
    render("phone_fields", :f => fb)
  end
  js = escape_javascript(fields)
  fn = html_escape("addPhone($(' #phones' ), \"#{js}\")")
    concat(raw("<a href=\"#{\" class=\"btn btn-primary\"
onclick=\"#{fn}\"><span class=\"glyphicon glyphicon-
plus\"></span></a>"))
%>
```