02

Atom e Content negotiation via URI

Até agora, aprendemos a renderizar páginas em HTML, XML ou JSON. Com isso, podemos ter diferentes aplicações consumindo nossos dados, sem muito esforço.

Dê agora uma olhada no nosso `JobsController`. Veja por exemplo, o método `index`: nele já nos preocupamos em renderizar as informações em HTML ou JSON. O método `respond_to` é quem se responsabiliza por descobrir se o cliente quer a resposta em HTML ou JSON. Chamamos isso de **Content Negotiation**: a negociação que é feita com o cliente, para saber o tipo da resposta que ela espera.

```
respond_to do |format|
  format.html
  format.json { render json: @jobs }
end
```

Se formos no browser e acessarmos `/jobs`, teremos a resposta em HTML. Mas, se mudarmos a URL, e forçarmos um JSON com `/jobs?format=json`, ele nos devolve em JSON.

No exemplo acima, "forçamos" o JSON, mas o Rails é esperto, e observa também o header do HTTP, que é a maneira elegante de informar o tipo da resposta esperada de uma requisição.

Fazemos uma requisição pelo terminal: `curl http://localhost:3000/jobs (http://localhost:3000/jobs)`. Como não falamos nada sobre o tipo de retorno que esperamos, ele nos devolve em HTML, que é o padrão de negociação de conteúdo do Rails.

Mas se falarmos no cabeçalho HTTP que só aceitamos respostas em JSON, ele nos devolverá em JSON. Para isso adicionamos o header "Accept: application/json", fazendo então `curl -H "Accept: application/json" http://localhost:3000/jobs (http://localhost:3000/jobs)`.

Outro formato bem comum para disponibilizar informações na Web, é através de RSS ou Atom. Ou seja, um "feed" de informações.

Para isso, basta adicionarmos mais um formato:

```
respond_to do |format|
  format.html
  format.json { render json: @jobs }
  format.atom { }
end
```

O que faremos agora é usar a maneira tradicional de renderizar uma action no Rails: passando para um ERB. Só que aqui usaremos um builder para gerar esse conteúdo. Vamos criar um arquivo `/views/jobs/index.atom.builder`. Dentro desse builder, colocaremos as informações para a geração do feed. Todo feed tem uma língua, título e uma data de última atualização. Isso, em Ruby, fica:

```
atom_feed language: 'en-US' do |feed|
  feed.title 'Jobs Board'
```

```
    feed.updated @last_updated
  end
```

A variável `@last_updated` será preenchida pelo método no controller. Ela conterá a data do último emprego inserido. Dessa maneira, um cliente dessa API consegue saber se houve atualização ou não.

De volta ao controller, vamos passar os dados para esse template:

```
  respond_to do |format|
    format.html
    format.json { render json: @jobs }
    format.atom {
      if @jobs.first
        @last_updated = @jobs.first.updated_at
      else
        @last_updated = Time.now
    }
  end
```

Se acessarmos agora `/jobs?format=atom` , ele nos devolve a resposta em Atom. Ótimo, o primeiro passo já foi dado. Precisamos agora colocar os jobs nesse atom. Para isso, faremos um loop na lista de jobs, e para cada job, inseriremos uma `entry` . Ela tem informações relevantes para o feed, como URL, título, conteúdo, data de atualização, autor.

Vamos ao código:

```
  atom_feed language: 'en-US' do |feed|
    feed.title 'Jobs Board'
    feed.updated @last_updated

    @jobs.each do |job|
      feed.entry(job) do |entry|
        entry.url url_for(job)
        entry.title job.title
        entry.content job.description
        entry.updated entry.updated_at.strftime("%Y-%m-%dT%H:%M:%SZ")
        entry.author do |author|
          author.name job.company.name
        end
      end
    end
  end
```

Agora, se olharmos o feed, ele está completo:

```
  <?xml version="1.0" encoding="UTF-8"?>
  <feed xml:lang="en-US" xmlns="http://www.w3.org/2005/Atom">
    <id>tag:localhost,2005:/jobs?format=atom</id>
    <link rel="alternate" type="text/html" href="http://localhost:3000"/>
    <link rel="self" type="application/atom+xml" href="http://localhost:3000/jobs?format=atom"/>
    <title>Jobs Board</title>
    <updated>2013-03-01T13:30:25Z</updated>
```

```
  <entry>
    <id>tag:localhost,2005:Job/4</id>
    <published>2013-03-01T12:40:41Z</published>
    <updated>2013-03-01T12:40:41Z</updated>
    <link rel="alternate" type="text/html" href="http://localhost:3000/jobs/4-Teacher"/>
    <url>/jobs/4-Teacher</url>
    <title>Teacher</title>
    <content>Teach people</content>
    <updated>2013-03-01T12:40:41Z</updated>
    <author>
      <name>Caelum</name>
    </author>
  </entry>
  <entry>
    <id>tag:localhost,2005:Job/3</id>
    <published>2013-03-01T12:40:35Z</published>
    <updated>2013-03-01T12:40:35Z</updated>
    <link rel="alternate" type="text/html" href="http://localhost:3000/jobs/3-Support"/>
    <url>/jobs/3-Support</url>
    <title>Support</title>
    <content>Help people</content>
    <updated>2013-03-01T12:40:35Z</updated>
    <author>
      <name>Caelum</name>
    </author>
  </entry>
  <entry>
    <id>tag:localhost,2005:Job/2</id>
    <published>2013-03-01T12:39:50Z</published>
    <updated>2013-03-01T12:39:50Z</updated>
    <link rel="alternate" type="text/html" href="http://localhost:3000/jobs/2-Front-end"/>
    <url>/jobs/2-Front-end</url>
    <title>Front-end</title>
    <content>Need a front-end developer</content>
    <updated>2013-03-01T12:39:50Z</updated>
    <author>
      <name>Plataformatec</name>
    </author>
  </entry>
  <entry>
    <id>tag:localhost,2005:Job/1</id>
    <published>2013-03-01T12:38:06Z</published>
    <updated>2013-03-01T12:38:06Z</updated>
    <link rel="alternate" type="text/html" href="http://localhost:3000/jobs/1-Developer"/>
    <url>/jobs/1-Developer</url>
    <title>Developer</title>
    <content>Good guy.</content>
    <updated>2013-03-01T12:38:06Z</updated>
    <author>
      <name>Plataformatec</name>
    </author>
  </entry>
</feed>
```

Precisamos agora avisar os outros sistemas que temos a lista disponibilizada através de um feed! Para isso, voltamos ao nosso index.html.erb, e acrescentamos o seguinte código:

```
<%= auto_discovery_link_tag :atom, jobs_url %>
```

Essa tag inclui na página uma informação que é lida pelo browser, e que o avisa de que existe uma versão dessa mesma página, mas disponibilizada em outro media type.

Veja, se olharmos o código-fonte no browser, vemos a tag:

```
<link href="http://localhost:3000/jobs" rel="alternate" title="ATOM" type="application/atom+xml"
```

Dessa forma, esse feed pode ser importado em ferramentas como Google Reader ou similares.