What I used: I basically used Nokogiri and RSpec, since Nokogiri is the most known XML parsing gem for Ruby and it is known for its performance (http://www.rubyinside.com/ruby-xml-performance-benchmarks-1641.html)

What could be improved: I could use JRuby to get real threads and make the parsing/html creation processed in parallel

Ideal solution: I could use Java or Scala, and NIO2 to read a chunk. For each batch in a chunk, send it a LinkedBlockingQueue that is consumed by threads (num of cpu's) and then the thread finds taxonomies and write the html file.

Improvement for this ideal solution: Change the LinkedBlockingQueue to a Akka Actors that receives the chunk and process.

Some small projects that I worked:

· https://github.com/rodrigodealer/authentication - Small study case for Elixir lang

· https://github.com/rodrigodealer/scala_terminal_twitter - Small terminal twitter case written in Scala, this is a very nice project.

It has ansible provisioning and it's also very well tested

· https://github.com/rodrigodealer/user-graph - It's a fork from a project that I worked in a Globo's hackathon, we created an application that runs with a Neo4j Graph Database and adds/consumes user interests.

My stack overflow profile: http://stackoverflow.com/users/1731689/rodrigo-oliveira