```
module Performance
  require "fileutils"
  require "jammit"
  require "memcached"

  #It uses Apache configuration to do compreession of some type of html, css,
js and xml.
  #It creates cache with duration of 1 month to images and others static
files.
  def configs_apache
    if File.exists?("#{Rails.root}/public/.htaccess")  then
      File.open("#{Rails.root}/public/.htaccess", "a") do |file_apache|
        file_apache.puts "<IfModule mod_deflate.c>\n\tAddOutputFilterByType
DEFLATE text/html text/plain text/xml text/css
application/x-javascript\n</IfModule>\n\n"
        file_apache.puts "ExpiresActive On\n<FilesMatch
'\\.(ico|jpg|jpeg|png|gif|js|css)'>\n\tExpiresDefault 'access plus 1
month'\n</FilesMatch>"
        file_apache.puts "<Directory
'#{Rails.root}/public/assets'>\n\tExpiresDefault 'access plus 1
month'\n</Directory>"
      end
    else
      File.open("#{Rails.root}/public/.htaccess", "w") do |file_apache|
        file_apache.puts "<IfModule mod_deflate.c>\n\tAddOutputFilterByType
DEFLATE text/html text/plain text/xml text/css
application/x-javascript\n</IfModule>\n\n"
        file_apache.puts "ExpiresActive On\n<FilesMatch
'\\.(ico|jpg|jpeg|png|gif|js|css)'>\n\tExpiresDefault 'access plus 1
month'\n</FilesMatch>"
        file_apache.puts "<Directory
'#{Rails.root}/public/assets'>\n\tExpiresDefault 'access plus 1
month'\n</Directory>"
      end
    end
  end

  #This procedure joins .css and .js files. Minimize your data. It creates
helpers into ApplicationHelper to load .js and .css files
  #into views and insert .css to load first and .js after
  def join_jscss
    make_datas
    Dir.chdir("#{Rails.root}/public/")
#    system("jammit")
    Jammit.package!
```

```
    p "Making procedures into app helper"
    dir_helper =  "#{Rails.root}/app/helpers/"
    File.open("#{dir_helper}application_helper.rb","a") do |read_helper|
      read_reader.readlines().each do |line|
        if line =~ /ApplicationHelper/ then
          read_helper.puts "module ApplicationHelper\n\n\tdef
stylesheets(*files)
          \n\tcontent_for(:stylesheets)  { stylesheet_link_tag(*files)
}\nend\n\n
          def javascripts(*files)\n\tcontent_for(:javascripts)  {
javascripts_link_tag(*files)
          }\nend"
        else
          read_helper.puts "#{line}"
        end
      end
    end

    dir_view_app =  "#{Rails.root}/app/views/layouts/"
    Dir.chdir("#{dir_view_app}")
    #Copia o conteudo do arquivo application.html.erb e cria a melhoria dos
links de js e css agrupados
    temp = File.new("app.html.erb","w")
    if File.exists?("application.html.erb") then
      File.open("application.html.erb", "r") do |file_reader|
        file_reader.readlines().each do |line|
          if line =~ /<\/title>/
            temp.puts "\t\t\t<title><%= title %><\/title>\n\t\t\t<%=
csrf_meta_tag %>\n\t\t\t<%= include_stylesheets :workspace, :media => 'all'
%>\n\t\t\t<%= yield stylesheets %>\n"
          else
            temp.puts "#{line}"
          end
          temp.puts "\n\t\t<%= include_javascripts :workspace %>\n\t\t<%=
yield javascripts %>\n" if line =~ /<\/body>/
        end
        temp.close
      end
    else
      puts "Arquivo application.html.erb Inexistente"
    end
    p "Renaming your application.html.erb to old_application.html.erb"
    FileUtils.mv("application.html.erb", "old_application.html.erb")
    FileUtils.mv("app.html.erb", "application.html.erb")
  end
```

```
  def make_datas
    Dir.mkdir("#{Rails.root}/public/config") if !Dir.exists?
    Dir.chdir("#{Rails.root}/public/config")
    File.new("assets.yml", "w") do |file|
      file.puts("embed_assets: on\njavascripts:\n\tworkspace:\n\t\t -
public/javascripts/*.js")
      file.puts("\nstylesheets:\n\tworkspace:\n\t\t -
public/stylesheets/*.css")
    end
  end

  #Make cache in controller and action, when happen the operations destroy,
create and update.
  #The controller and action are parameters 0 and 1 RESPECTIVAMENTE
  def cache_page_server(controller, action)
    File.open("#{Rails.root}/app/models/#{controller.downcase}_sweeper.rb",
"w") do |ob|
      ob.puts "class #{controller.capitalize}Sweeper <
ActionController::Caching::Sweeper"
      ob.puts "\tobserve #{controller.capitalize!}"
      ob.puts "\tdef expire_cached_content(#{controller.downcase})\t"
      ob.puts "expire_page :controller => '#{controller.pluralize}', :action
=> '#{action.downcase}'"
      ob.puts "\texpire_fragment(%r{#{controller.pluralize}/.*})\nend "
      ob.puts "alias_method :after_save, :expire_cached_content"
      ob.puts "alias_method :after_destroy, :expire_cached_content\nend"
    end

File.open("#{Rails.root}/app/controllers/#{controller.pluralize}_controller.
b", "a") do |file|
      file.readlines().each do |line|
        if line =~ /class #{controller.pluralize}Controller <
ApplicationController/ then
          file.puts("class #{controller.pluralize}Controller <
ApplicationController\n\tcaches_page :#{action.downcase}")
          file.puts("\n\tcache_sweeper :#{controller.downcase}_sweeper, :only
=> [:create, :update, :destroy]")
        else
          file.puts "#{line}"
        end
      end
    end
  end
```

```
  #Configure a static server to use to download static files such as images,
videos and sounds.
  def config_static_server(server)
    File.open("#{Rails.root}/config/environments/production.rb","a") do
|file|
      file.readlines().each do |line|
        if line =~ /# config.action_controller.asset_host =
'http:\/\/assets.example.com'/ then
          file.puts("\n\tconfig.action_controller.asset_host =
'http:\/\/assets%d.#{server}'\n")
        else
          file.puts "#{line}"
        end
      end
    end
  end

  def copy_tasks

FileUtils.mv("files/performance.rake","#{Rails.root}/lib/tasks/performance.r
ke")
  end

  def require_memory
    temp = File.new("#{Rails.root}/app.rb","w")
    File.open("#{Rails.root}/config/application.rb","r") do |file_require|
      file_require.readlines().each do |line|
        if line =~ /require 'rails\/all'/
          temp.puts "require 'rails\/all'"
          temp.puts "\nrequire 'memcached'\n"
        else
          temp.puts "#{line}"
        end
        if line=~ /class Application < Rails::Application/
          temp.puts "class Application <
Rails::Application\n\tconfig.session_store = :mem_cache_store"
        end
      end
    end
    temp.close
    File.mv("#{Rails.root}/config/application.rb",
"#{Rails.root}/config/old_application.rb")
    File.mv("#{Rails.root}/app.rb","#{Rails.root}/config/application.rb")
  end
```

```
  #Memory configuration using memcached
  def memory(size)
    if size.nil?
      system("memcached -d -m 512 -p 11211")
    else
      system("memcached -d -m #{size} -p 11211")
    end
    require_memory
    temp1 = File.new("#{Rails.root}/controller.rb","w")
    File.open("#{Rails.root}/app/controllers/application_controller.rb","r")
do |file_controller|
      file_controller.readlines().each do |line|
        if line =~ /class ApplicationController < ActionController::Base/
          temp1.puts "class ApplicationController < ActionController::Base"
          temp1.puts "\n\tsession :cache =>
MemCache.new('localhost:11211')\n"
        else
          temp1.puts "#{line}"
        end
      end
    end
    File.mv("#{Rails.root}/config/application.rb",
"#{Rails.root}/config/old_application.rb")
    File.mv("#{Rails.root}/app.rb","#{Rails.root}/config/application.rb")
    temp1.close
  end

  #Procedure to make balance memory in 2 servers
def multi_memory(s1,s2)
  app_dir = "#{Rails.root}/app/controllers/application_controller.rb"
  if s1.present? && s2.nil?
    require_memory
    temp1 = File.new("#{Rails.root}/controller.rb","w")
    File.open("#{app_dir}","r") do |file_controller|
      file_controller.readlines().each do |line|
        if line =~ /class ApplicationController < ActionController::Base/
          temp1.puts "class ApplicationController < ActionController::Base"
          temp1.puts "\n\tsession :cache => MemCache.new('#{s1}:11211')\n"
        else
          temp1.puts "#{line}"
        end
      end
    end
  elsif s1.present? && s2.present?
    require_memory
```

```
    File.open("#{app_dir}","r") do |file_controller|
      file_controller.readlines().each do |line|
        if line =~ /class ApplicationController < ActionController::Base/
          temp1.puts "class ApplicationController <
ActionController::Base\n\t"
          temp1.puts "session :cache => MemCache.new('#{s1}:11211',
'#{s2}:11211')\n"
        else
          temp1.puts "#{line}"
        end
      end
    end
  elsif s1.nil? && s2.nil?
    memory(512)
  elsif s1.nil? && s2.present?
    require_memory
    temp1 = File.new("#{Rails.root}/controller.rb","w")
    File.open("#{app_dir}","r") do |file_controller|
      file_controller.readlines().each do |line|
        if line =~ /class ApplicationController < ActionController::Base/
          temp1.puts "class ApplicationController < ActionController::Base"
          temp1.puts "\n\tsession :cache => MemCache.new('#{s2}:11211')\n"
        else
          temp1.puts "#{line}"
        end
      end
    end
  temp1.close
  else
    puts "Não foi possível configurar o servidor memcached."
    exit
  end
  File.mv("#{app_dir}",
"#{Rails.root}/app/controllers/old_application_controller.rb")
  File.mv("#{Rails.root}/controller.rb","#{app_dir}")
end

  def run
    join_jscss
    configs_apache
    memory(512)
    cache_page_server
    config_static_server
  end
end
```