Integrate Rails into an Existing IIS Web infrastructure using Apache and FastCGI

This article will walk you through the steps of installing Ruby, Gems, Rails, and other important libraries on a Windows 2003 server with IIS.

Microsoft's Internet Information Server is a popular proprietary web and application server. Those who have attempted to run Rails applications on IIS have had mixed results at best.

Revision 1.0 – May 4, 2006

Assumptions

This article assumes that you have a working Rails application to test, that you are familiar with how IIS works, and that you have the MySQL Database installed on the local machine. IIS should be running on TCP Port 80.

Shopping List

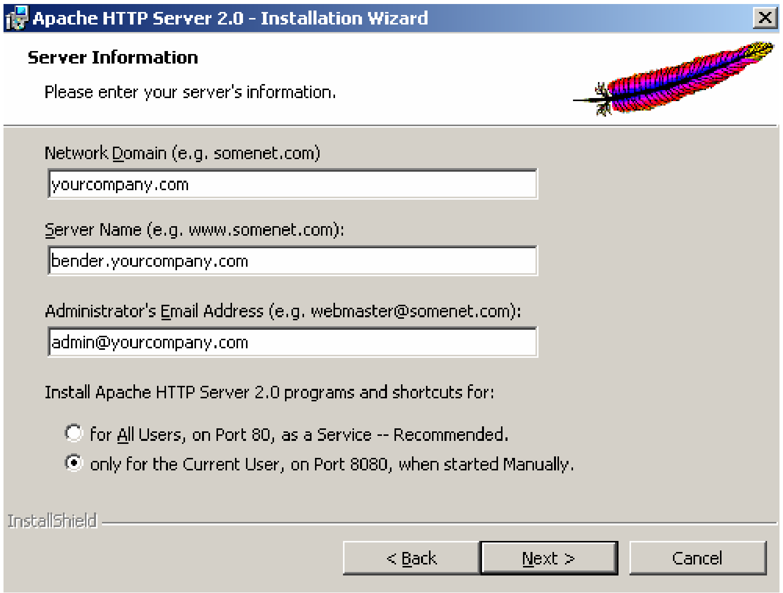
In order to make this work, you'll need to download the full version of ISAPI Rewrite from a company called Helicon. You can obtain a free 30 day unlimited trial from their web site but they charge $70 per server (or less if you buy more than once license) if you want to use it in production. While there are free rewrite plugins available, this is the only one I know of that provides proxy capabilities for IIS.

Download and Install Apache

This article will use the 2.0x branch of Apache. At the time of writing, the 2.2x branch is available but there are issues with FastCGI.

1. Download Apache for Windows from here:
   http://apache.cs.utah.edu/httpd/binaries/win32/apache_2.0.55-win32-x86-no_ssl.msi

2. Log in as a member of the **Administrators** group and run the installer.



Apache HTTP Server 2.0 - Installation Wizard

Server Information

Please enter your server's information.

Network Domain (e.g. somenet.com)
yourcompany.com

Server Name (e.g. www.somenet.com):
bender.yourcompany.com

Administrator's Email Address (e.g. webmaster@somenet.com):
admin@yourcompany.com

Install Apache HTTP Server 2.0 programs and shortcuts for:

○ for All Users, on Port 80, as a Service -- Recommended.
● only for the Current User, on Port 8080, when started Manually.

3. Be sure to select "only for the current user on port 8080". We'll install it as a service later but we'll need to make a lot of configuration changes.

4. Choose **Custom Install** and change the installation folder to **c:\apache** and choose **Next**.

5. The installer will take a minute or so, and you should see a console window appear for some configuration. After this, the installation should complete.

Start Apache as a Service.

1. Open a command prompt and navigate to the folder **c:\apache\apache2\bin**
2. Type **apache –k install** to install Apache as a service.
3. If the Windows Firewall asks you that you need to allow access to Apache, you need to allow it.
4. The service should now appear as a Windows service and should also appear in the Apache monitor.
5. Apache is now running as a service, but it is not started yet. That's fine because we've got a lot more work to do.

Installing Rails

Installing Ruby, Rubygems, and RDoc

1. Download the One-Click Ruby Installer. You can use **Ruby 1.8.2** or **Ruby 1.8.4**.
2. Install the software to the default location **c:\ruby** and accept all defaults.

Install Rails

1. Open a command prompt
2. Install Rails (**gem install rails –include-dependencies**)
3. Install RedCloth (**gem install redcloth**)

Install Rmagick

This section is left in because it may eventually work again. Right now, this version of Rmagick **does not work on Windows with Ruby 1.8.4.**

1. Download the special Windows version of Rmagick from
   http://rubyforge.org/frs/download.php/6276/RMagick-1.9.2-IM-6.2.4-6-win32.zip
2. Unzip this to a temporary location
3. Open a command prompt and navigate into the extracted location
4. Install the gem
   gem install Rmagick-1.9.2-IM-6.2.4-6-win32.gem
5. Run the postinstall.rb script
   ruby postinstall.rb

Configure Apache for Rails Hosting

There are some important steps that need to be performed in order for Apache to start serving out Rails applications efficiently.

Install Ruby for Apache

1. Download Ruby for Apache from http://rubyforge.org/frs/download.php/5256/RubyForApache-1.3.1.exe

2. Run the installer

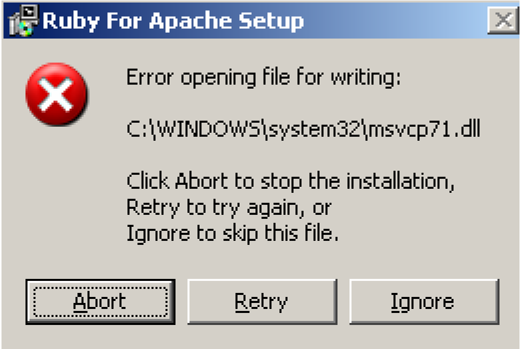
3. For the **Destination Folder**, choose the default

4. For the **Apache Directory**, enter **C:\apache\apache2**

5. For the **Ruby Directory**, enter **C:\Ruby**

6. **only install mod_fastcgi**. Do not install **mysql.so** or **mod_ruby**!!!

7. If you see this error:



Error opening file for writing:

C:\WINDOWS\system32\msvcp71.dll

Click Abort to stop the installation,
Retry to try again, or
Ignore to skip this file.

you can safely choose **ignore.**

Download and install mod_fastcgi for Apache 2

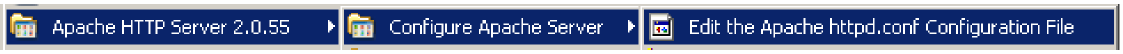
1. Download from http://www.fastcgi.com/dist/mod_fastcgi-2.4.2-AP20.dll

2. Copy the file to **c:\apache\apache2\modules**

3. Rename the file to **mod_fastcgi.dll**

Configure Apache

1. Locate the file c:\apache\apache2\conf\httpd.conf and open it in a text editor. For convenience, the Apache installer may have placed an entry in your Start menu for this. You may have to choose to associate this file with an editor. Choose Notepad or a similar text editor.



Apache HTTP Server 2.0.55
Configure Apache Server
Edit the Apache httpd.conf Configuration File

2. Locate the line
   LoadModule rewrite_module modules/mod_rewrite.so
   and remove the # character to uncomment the line

3. Add the line
   LoadModule fastcgi_module modules/mod_fastcgi.dll
   to the end of the LoadModules section

4. Save the file. **Don't close it!**

5. Start Apache by using the Apache Control or by starting the service from the **Services** panel in Windows.

6. Navigate to http://localhost:8080 and ensure that you do have a server running!

7. Shut down the Apache service

Serving a Rails Application

Preparing a Rails application

1. Make a folder on your server to hold your rails application.
   ```
   c:\rails
   ```

2. Copy a working Rails application into that folder
   ```
   c:\rails\test
   ```

3. Ensure the Rails application works by testing it with WEBrick. Make sure that the database configuration for production is correct
   ```
   cd\rails\test
   ruby script/server –e production
   ```

Aliases and Rails

To prepare our server to play nicely with IIS, we will mount each rails application as a subfolder, or alias, in Apache. This will allow us to use one Apache installation to host several Rails applications.

For our **test** application, we'll mount that to an alias of '**test**' and we should have a url of http://localhost:8080/test/ when we're done.

We'll eventually configure IIS to send requests for http://localhost/test/ to this new address.

Configure Apache for a Rails App

1. In the **httpd.conf** file, go to the very end of the file.

2. Add an **alias** for our new application. The alias should point to the **public** folder of the Rails application.

   ```
   Alias /test "c:/rails/test/public"
   ```

3. Add the **FastCGI** server. This should point to the **dispatch.fcgi** file in the **public** folder of the Rails application.

   ```
   FastCgiServer c:/rails/test/public/dispatch.fcgi -idle-timeout 120  -initial-env
   RAILS_ENV=production -processes 2
   ```

4. Finally, add the **Directory** information so Apache can serve the files appropriately.
   ```
   <Directory c:/rails/test/public>
     Options ExecCGI FollowSymlinks
     AllowOverride All
   </Directory>
   ```
5. Save the file. **Don't close it yet!**

6. **Start Apache**. If you receive errors, then you need to correct them before you can continue. Apache must be started for you to continue.

You can review the complete httpd.conf file in the appendix.

Modify the Rails .htaccess file

Because we're using an Alias to mount this application to Apache, we have to make a minor change to the way the rewrite rules work for the Rails application. Additionally, we need to tell our Rails application to use FastCGI.

Each Rails application has a file called .htaccess that is located in the **public** folder of the Rails structure. This file contains important information about how the application is accessed through Apache.

1. Open this file in a text editor
2. Locate the line that says
   # RewriteBase /myrailsapp
3. Change it to
   RewriteBase /test
4. Locate the line
   RewriteRule ^(.*)$ dispatch.cgi [QSA,L]
5. Change it to
   RewriteRule ^(.*)$ dispatch.fcgi [QSA,L]
6. Save and close the file

You can see the complete .htaccess file in the appendix.

Testing the Application

Open a browser and point to **http://localhost:8080/test/** Your Rails application should appear.

IIS Integration

Installl ISAPI Rewrite

Visit http://www.isapirewrite.com/ and download the trial version of the ISAPI Rewrite plugin.

- Direct download is http://www.isapirewrite.com/download/isapi_rwf_x86_0060.msi
- Launch the installation program and accept all of the default settings.
- You will be prompted to restart IIS and you should allow this.
- If you experience trouble with the installation, you'll need to refer to the developers of this product.

Fixing the ISAPIRewrite association issue

After installing, it may be necessary to "fix" the association of this filter.

- Open a command prompt
- Navigate to **C:\Program Files\Helicon\ISAPI_Rewrite**
- Launch the ProxyCfg.vbs script
    - Proxycfg.vbs –r or cscript proxycfg.vbs –r
- Restart IIS

Configure ISAPI Rewrite

The last step is to modfy the httpd.ini file which resides in **C:\Program Files\Helicon\ISAPI_Rewrite** (you can review the complete file in the appendix.)

Add this line to the bottom of the file.

```
# Proxy requests to Apache on 8080.

# FOR TEST APPLICATION
RewriteProxy /test(.*) http\://localhost:8080/test$1 [I,U]
```

Save the file and restart IIS.

Testing the setup

If all worked well, you can now pull up your Rails application via IIS by navigating to http://localhost/test/

Unfortunately, it's not going to look very good. Read on to find out why.

Reverse Proxy and URLs

The big problem we're faced with now is that the URLs that Rails creates internally, such as stylesheet links, url_for links and other links don't work as we expect… instead, they direct users around the proxy. This is bad because it exposes the proxied server.

IIS has no method to handle reverse proxying. A reverse proxy rewrites the content served from the backend to mask the fact that the request was filtered through a proxy.

Thankfully, there's a way around this… using a simple Rails plugin that modifieds the way Rails creates its URLs. We're going to make Rails prepend our external URL to any URLs it creates through the system. This will force all user requests to come back through the IIS proxy.

Installing the proxy plugin

Execute the command

```
ruby script/plugin install http://svn.napcsweb.com/public/reverse_proxy_fix
```

from within your application's root folder. The plugin should install and then ask you for the base url. Enter **http://localhost/app1** and pres 'enter'. If all goes well, the configuration file will be written. If the configuration file can't be modified, you can navigate to **vendor/plugins/reverse_proxy_fix** and change it yourself.

If the installation fails, you can build the plugin yourself if you follow the next section.

Creating the proxy plugin

If you don't have Subversion installed, you can follow these steps to get the plugin configured properly.

- Create a new Rails plugin called "reverse_proxy_fix"
  ruby/script generate reverse_proxy_fix
- Navigate to your application's vendor/plugins/reverse_proxy_fix folder and edit the init.rb file
  - Add the following code to the file
    Require 'reverse_proxy_fix"
- Edit **vendor/plugins/iis_proxy_fix/lib/reverse_proxy_fix.rb** and replace the contents with the code located in the appendix.
- Modify the first line to match your application's url…
- **BASE_URL = 'http://localhost/app1'**
- Finally, restart your Rails application by shutting down WEBrick and restarting it
- If all went as expected, any internal links in your application should now be corrected and routed back through the proxy.

Wrapping Up

You now know how to run a Rails application on Apache and have it integrated into your IIS environnment. However there are a few concerns.

Integrate Rails into an Existing IIS Web infrastructure using Apache and FastCGI

First, this is not going to be the best solution if you want to run multiple applications. Apache and FastCGI tends to become **really slow** on startup when you restart Apache if you have several applications served by the same server. You can distribute the applications on many backend servers thanks to ISAPI Rewrite and the plugin described in this document.

Also, in order to restart a production Rails application, you need to shut down Apache which will kill off all of your other applications as well.

Appendix

Sample Apache httpd.conf file

```
ServerRoot "C:/apache/Apache2"

PidFile logs/httpd.pid
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15

<IfModule mpm_winnt.c>
        ThreadsPerChild 250
        MaxRequestsPerChild  0
</IfModule>

Listen 80

LoadModule access_module modules/mod_access.so
LoadModule actions_module modules/mod_actions.so
LoadModule alias_module modules/mod_alias.so
LoadModule asis_module modules/mod_asis.so
LoadModule auth_module modules/mod_auth.so
LoadModule autoindex_module modules/mod_autoindex.so
LoadModule cgi_module modules/mod_cgi.so
LoadModule dir_module modules/mod_dir.so
LoadModule env_module modules/mod_env.so
LoadModule imap_module modules/mod_imap.so
LoadModule include_module modules/mod_include.so
LoadModule isapi_module modules/mod_isapi.so
LoadModule log_config_module modules/mod_log_config.so
LoadModule mime_module modules/mod_mime.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule setenvif_module modules/mod_setenvif.so
LoadModule userdir_module modules/mod_userdir.so
#LoadModule ssl_module modules/mod_ssl.so


ServerAdmin myserver.mydomain.com

ServerName myserver.mydomain.com:80

UseCanonicalName Off

DocumentRoot "C:/apache/Apache2/htdocs"

<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

<Directory "C:/apache/Apache2/htdocs">
    Options Indexes FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

DirectoryIndex index.html index.html.var
AccessFileName .htaccess

<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
</FilesMatch>

TypesConfig conf/mime.types
DefaultType text/plain

<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>

HostnameLookups Off

ErrorLog logs/error.log
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog logs/access.log common

ServerTokens Full
ServerSignature On

Alias /icons/ "C:/apache/Apache2/icons/"

<Directory "C:/apache/Apache2/icons">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

AliasMatch ^/manual(?:/(?:de|en|es|fr|ja|ko|ru))?(/.*)?$ "C:/apache/Apache2/manual$1"

<Directory "C:/apache/Apache2/manual">
    Options Indexes
    AllowOverride None
    Order allow,deny
    Allow from all

    <Files *.html>
        SetHandler type-map
    </Files>

   SetEnvIf Request_URI ^/manual/(de|en|es|fr|ja|ko|ru)/ prefer-language=$1
   RedirectMatch 301 ^/manual(?:/(de|en|es|fr|ja|ko|ru)){2,}(/.*)?$ /manual/$1$2
</Directory>

ScriptAlias /cgi-bin/ "C:/apache/Apache2/cgi-bin/"

<Directory "C:/apache/Apache2/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>


IndexOptions FancyIndexing VersionSort

AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*

AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

DefaultIcon /icons/unknown.gif

ReadmeName README.html
HeaderName HEADER.html

IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t

AddLanguage ca .ca
AddLanguage cs .cz .cs
AddLanguage da .dk
AddLanguage de .de
AddLanguage el .el
AddLanguage en .en
AddLanguage eo .eo
AddLanguage es .es
AddLanguage et .et
AddLanguage fr .fr
AddLanguage he .he
AddLanguage hr .hr
AddLanguage it .it
AddLanguage ja .ja
AddLanguage ko .ko
AddLanguage ltz .ltz
AddLanguage nl .nl
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
AddLanguage pt .pt
AddLanguage pt-BR .pt-br
AddLanguage ru .ru
AddLanguage sv .sv
AddLanguage zh-CN .zh-cn
AddLanguage zh-TW .zh-tw

LanguagePriority en ca cs da de el eo es et fr he hr it ja ko ltz nl nn no pl pt pt-BR ru
sv zh-CN zh-TW

ForceLanguagePriority Prefer Fallback

AddCharset ISO-8859-1  .iso8859-1 .latin1
AddCharset ISO-8859-2  .iso8859-2 .latin2 .cen
AddCharset ISO-8859-3  .iso8859-3 .latin3
AddCharset ISO-8859-4  .iso8859-4 .latin4
AddCharset ISO-8859-5  .iso8859-5 .latin5 .cyr .iso-ru
AddCharset ISO-8859-6  .iso8859-6 .latin6 .arb
AddCharset ISO-8859-7  .iso8859-7 .latin7 .grk
AddCharset ISO-8859-8  .iso8859-8 .latin8 .heb
AddCharset ISO-8859-9  .iso8859-9 .latin9 .trk
AddCharset ISO-2022-JP .iso2022-jp .jis
AddCharset ISO-2022-KR .iso2022-kr .kis
AddCharset ISO-2022-CN .iso2022-cn .cis
AddCharset Big5        .Big5       .big5
AddCharset WINDOWS-1251 .cp-1251    .win-1251
AddCharset CP866       .cp866
AddCharset KOI8-r      .koi8-r .koi8-ru
AddCharset KOI8-ru     .koi8-uk .ua
AddCharset ISO-10646-UCS-2 .ucs2
AddCharset ISO-10646-UCS-4 .ucs4
AddCharset UTF-8       .utf8
AddCharset GB2312      .gb2312 .gb
AddCharset utf-7       .utf7
AddCharset utf-8       .utf8
AddCharset big5        .big5 .b5
AddCharset EUC-TW      .euc-tw
AddCharset EUC-JP      .euc-jp
AddCharset EUC-KR      .euc-kr
AddCharset shift_jis   .sjis

AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz

AddHandler type-map var

BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-carefully
BrowserMatch "MS FrontPage" redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[0123]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully
BrowserMatch "^XML Spy" redirect-carefully
BrowserMatch "^Dreamweaver-WebDAV-SCM1" redirect-carefully

Alias /test "c:/rails/test/public"
FastCgiServer c:/rails/test/public/dispatch.fcgi -idle-timeout 120  -initial-env
RAILS_ENV=production -processes 2
<Directory c:/rails/test/public>
  Options ExecCGI FollowSymlinks
  AllowOverride All
</Directory>
```

.htaccess file for the rails application

```
AddHandler fastcgi-script .fcgi
AddHandler cgi-script .cgi
Options +FollowSymLinks +ExecCGI

RewriteEngine On
RewriteBase /test
RewriteRule ^$ index.html [QSA]
RewriteRule ^([^.]+)$ $1.html [QSA]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ dispatch.fcgi [QSA,L]

ErrorDocument 500 "<h2>Application error</h2>Rails application failed to start properly"
```

ISAPI Rewrite httpd.ini file

```
[ISAPI_Rewrite]
# 3600 = 1 hour
CacheClockRate 3600

RepeatLimit 32

# Block external access to the httpd.ini and httpd.parse.errors files
RewriteRule /httpd(?:\.ini|\.parse\.errors).* / [F,I,O]
# Block external access to the Helper ISAPI Extension
RewriteRule .*\.isrwhlp / [F,I,O]

# Proxy requests to Apache on 8080.

# TEST APPLICATION
RewriteProxy /test(.*) http\://localhost:8080/test$1 [I,U]
```

plugin/reverse_proxy_fix/lib/reverse_proxy_fix.rb

```
# Copyright © 2006 Brian Hogan
#
# Permission is hereby granted, free of charge, to any person obtaining
# a copy of this software and associated documentation files (the
# "Software"), to deal in the Software without restriction, including
# without limitation the rights to use, copy, modify, merge, publish,
# distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so, subject to
# the following conditions:
#
# The above copyright notice and this permission notice shall be
# included in all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
# EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
# MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
# NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
# LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
# WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

BASE_URL = ‘’
module ActionController

  protected
  # Configure the prefix on the url only if we're running in production mode
  # Throws an exception if the BASE_URL constant has not been configured in
  # config.rb
  def self.check_mode_and_base
        if RAILS_ENV == 'production'
                return true
        else
                return false
        end
  end

  # Set the asset host for CSS, JS, and image files if we're in production
  # mode and the base_path has been configured.
  if check_mode_and_base
        ActionController::Base.asset_host = BASE_URL
  end

  # Modifies the original UrlRewriter class, altering how the URLs are created.
  class UrlRewriter

        alias old_rewrite_url rewrite_url

        # Prepends the BASE_URL to all of the URL requests created by the
        # URL rewriter in Rails, stripping off the host, port, etc to ensure that
        # the new URL is exactly what you expect.
        #
        # This method calls check_mode_and_base to ensure that the URL fixing only occurs
        # in production mode and that the BASE_URL variable in config.rb is set.
        def rewrite_url(path, options)
                url = old_rewrite_url(path, options)
                url = url.gsub(@request.protocol + @request.host_with_port, '')
                if ActionController::check_mode_and_base
                        url = BASE_URL + url
                end
                url

        end
  end

end
```