使用Tengine+Lua+GraphicsMagick实现图片自动裁剪缩放—爱旅行项目

  随着互联网的快速发展、需求频繁变更、内容数量的俱增、时间的持续增长，图片数量也会越来越多。在实际需求中，会出现在若干个页面或同一个页面不同位置，展示同一条信息以及其缩略图。此时若使用CSS 控制图片显示的大小，对于一些与该位置不成比例的图片，缩小后就会出现图片变形，并且也不可能让UI对所有的图片进行PS，特别是当一张图需要多种比例尺寸，这样会产生巨大的工作量。基于这种情况就产生了强烈的自动化裁剪、缩放图片的需求，来适应不同规格的缩略图。那么针对作为一个旅游电商类网站：爱旅行项目，页面中有大量的图片加载，图片也是非常重要的一项页面内容，并且随之移动端的加入，考虑页面的响应速度，图片的加载速度，以及节省用户流量等等系统性能方面，都需要页面初始加载不再使用原图，而是根据不同比例需求等比裁剪的小图。所以要实现该非功能性需求，我们的解决方案是使用Tengine+Lua+GraphicsMagick来实现图片自动裁剪缩放。

补充说明：Nginx虽然有自带的 image filter module也可实现此功能，但是存有弊端，所以我们不推荐。

1. image filter module 使用的是GD，但是GD 性能、效率以及处理后的图片质量都不如 GraphicsMagick
2. image filter module没法真正生成裁剪/缩放后的图片，而是通过Nginx直接输出的，这样每次请求或缓存过期后都需要重新裁剪/缩放，这样无疑会增加Nginx负担。

## 1.1 软件准备

1. Tengine
2. 淘宝的Nginx，集成了lua模块，若要是使用官方的Nginx，默认安装没有lua模块，自己安装配置较为麻烦，所以我们选择Tengine。
3. 安装包：tengine-master.zip
4. 下载地址：<https://github.com/alibaba/tengine>
5. Lua
6. 脚本语言，Nginx通过它可以进行一些命令行操作。
7. 安装包：lua-5.3.1.tar.gz、LuaJIT-2.0.4.tar.gz（Lua依赖包）
8. 下载地址：<http://www.lua.org/ftp/>

http://luajit.org/download.html

1. GraphicsMagick
2. 图片处理工具，简单理解就是没有UI的PS。
3. 安装包：GraphicsMagick-1.3.18.tar.gz
4. 下载地址：https://sourceforge.net/projects/graphicsmagick/files/graphicsmagick/
5. 依赖包和类库，可yum安装
6. libjpeg、libjpeg-devel
7. libpng、libpng-devel
8. giflib、giflib-devel
9. freetype、freetype-devel

## 1.2 安装与配置

### 1.2.1 安装

安装过程不再赘述，安装后的路径：

1. /usr/local/Tengine
2. /usr/local/GraphicsMagick

安装过程强调两点：

1. 安装顺序： 1、GraphicsMagick 2、Lua 3、Tengine
2. 安装Tengine时，进入Tengine源码目录，编译安装命令如下:

./configure --prefix=/usr/local/Tengine --dso-path=/usr/local/Tengine/modules --with-http\_realip\_module --with-http\_gzip\_static\_module --with-http\_stub\_status\_module --with-http\_concat\_module --with-http\_lua\_module --http-proxy-temp-path=/var/tmp/Tengine/proxy\_temp --http-fastcgi-temp-path=/var/tmp/Tengine/fastcgi\_temp --http-uwsgi-temp-path=/var/tmp/Tengine/uwsgi\_temp --http-scgi-temp-path=/var/tmp/Tengine/cgi\_temp --http-client-body-temp-path=/var/tmp/Tengine/client\_body\_temp --http-log-path=/var/log/Tengine/access.log --error-log-path=/var/log/Tengine/error.log

make

make install

### 1.2.2 配置

1. 负责裁剪的Lua脚本（/usr/local/Tengine/lua/ImageResizer.lua）

local command = "/usr/local/GraphicsMagick/bin/gm convert " .. ngx.var.request\_filepath .. " -resize " .. ngx.var.width .. "x" .. ngx.var.height .. " +profile \"\*\" " .. ngx.var.request\_filepath .. "\_" .. ngx.var.width .. "x" .. ngx.var.height .. "." .. ngx.var.ext;

os.execute(command);

ngx.exec(ngx.var.request\_uri);

1. Nginx配置

user root; # 裁剪图片需要root权限

server {

listen 80;

server\_name img.itrip.project.bdqn.cn;

root /data/itrip/uploadimg;

location / {

root /data/itrip/uploadimg; # 站点根目录

expires 1h; # 缓存时间

add\_header Cache-Control max-age=3600; # 缓存时间

access\_log /var/log/Tengine/host\_access.log;

}

#如果 url 格式如：xxxx.gif\_数字x数字.gif

location ~\* ^(.+\.(jpg|jpeg|gif|png))\_(\d+)x(\d+)\.(jpg|jpeg|gif|png)$ {

root /data1/uploadimg; #这里必须设置，否则根目录，即 $document\_root 会是 Nginx 默认的 Nginx Root/html，在 Lua 中会得不到期望的值

if (!-f $request\_filename) { #如果文件不存在时才需要裁剪

add\_header X-Powered-By 'Lua GraphicsMagick'; #此HTTP Header无实际意义，用于测试

add\_header file-path $request\_filename; #此 HTTP Header无实际意义，用于测试

lua\_code\_cache on; #在编写外部 Lua脚本时，设置为off Nginx不会缓存 Lua，方便调试

set $request\_filepath /data1/uploadimg$1; #设置原始图片路径，如：/document\_root/1.gif

set $width $3; # 设置裁剪/缩放的宽度

set $height $4; # 设置裁剪/缩放的高度

set $ext $5; # 图片文件格式后缀

content\_by\_lua\_file /usr/local/Tengine/lua/ImageResizer.lua; #加载外部 Lua 文件

}

}

…}

## 1.3 项目运用

在进行爱旅行项目实际业务操作时，就可以实现按需自动裁剪，系统中的所有的上传图片，都只须上传原始图即可，有效的省去切图的工作量。具体示例如下：

目前服务器上有一张原始图（8-3566787986543-7654678.jpg），如下图1所示：

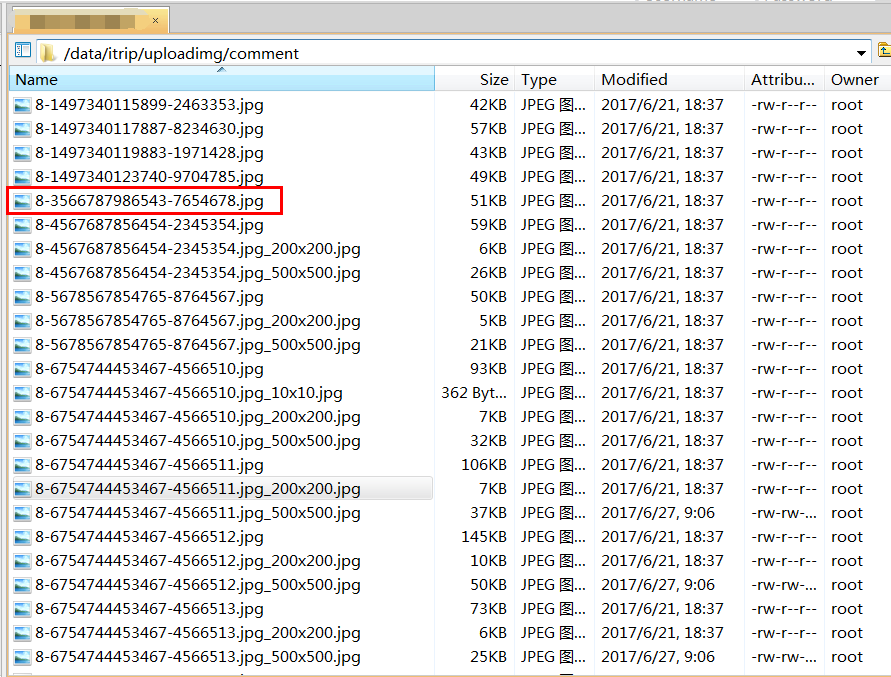


图1

访问该原始图(8-3566787986543-7654678.jpg)的URL为：http://img.itrip.project.bdqn.cn/comment/8-3566787986543-7654678.jpg ，如图2所示：

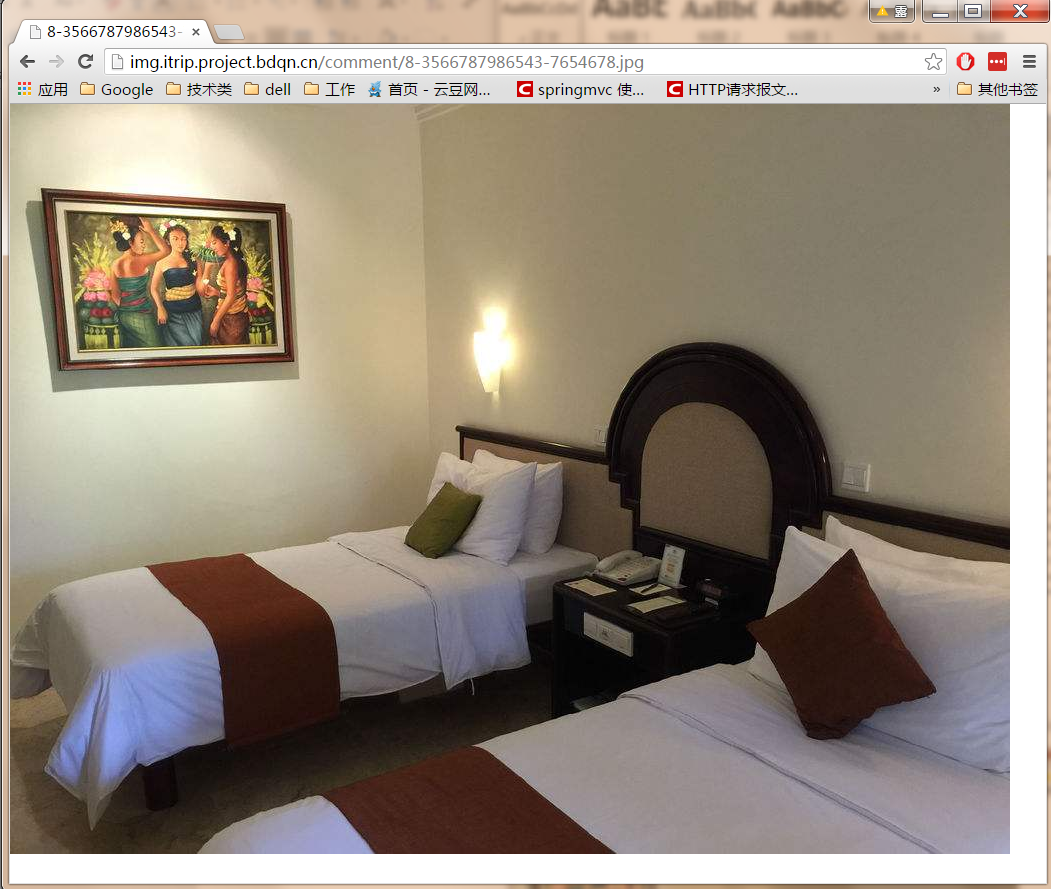


图2

若根据需求，页面需要展现该图片的小图（300x300），那么前端只需访问：http://192.168.9.44/comment/8-3566787986543-7654678.jpg\_300x300.jpg（如图3所示）即可，即：根据用户的访问请求，Lua会自动进行图片裁剪，生成相应尺寸的小图。

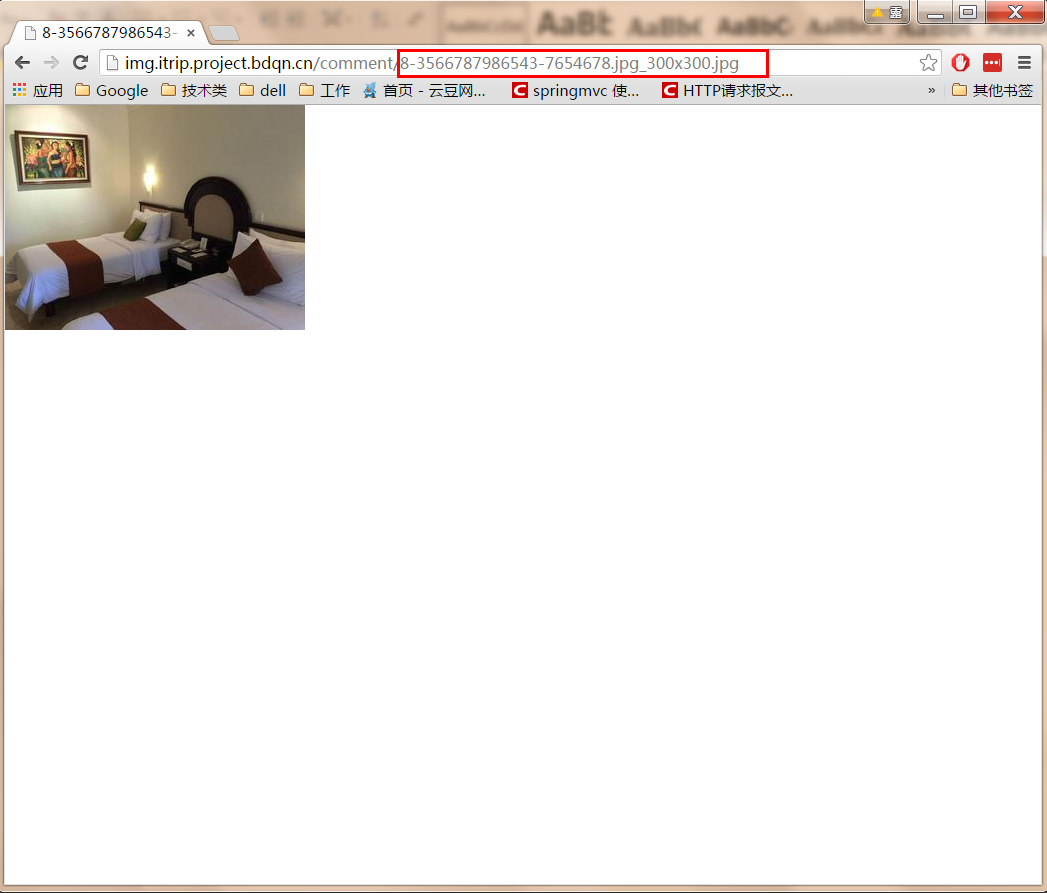


图3

服务器上自动生成8-3566787986543-7654678.jpg\_300x300.jpg，如下图4所示：

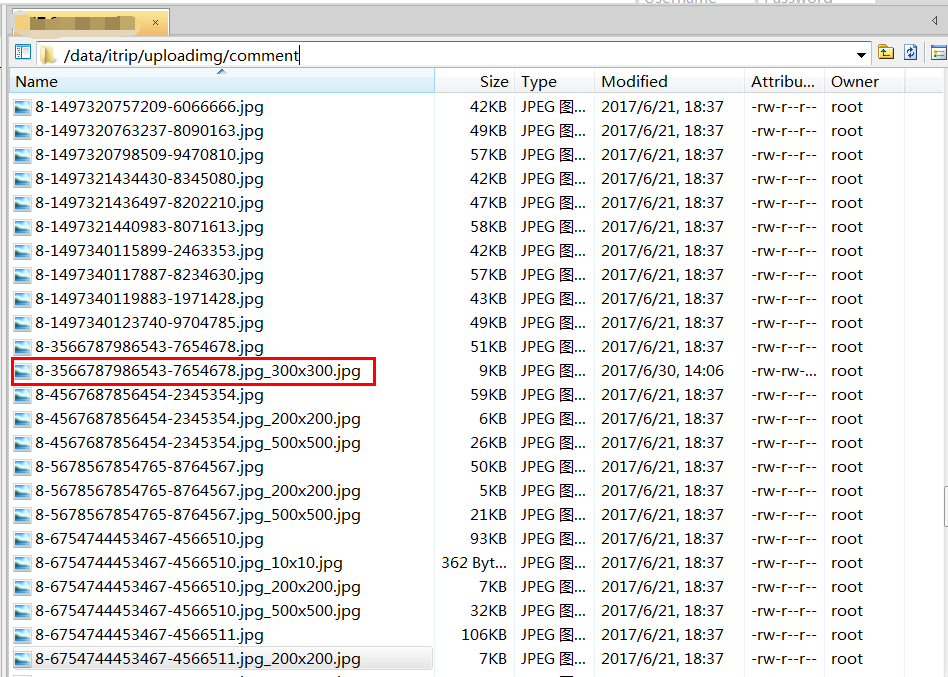


图4