Responsive Image as Service (RIaS)

Abhishek Tiwari

Citation: *A. Tiwari*, "Responsive Image as Service (RIaS)", Abhishek Tiwari, 2013. doi:10.59350/39xms-4p508

Published on: December 19, 2013

A Responsive Image as Service (RIaS) offers on-the-fly responsive image generation and delivery using REST APIs or REST like image paths.

RIaS is a key component of any responsive image solution. RIaS acts as image proxy. To generate responsive images it uses master images typically stored in an image repository.

RIaS can be deployed as standalone service as well as embedded service. Embedded service is normally part of core application and runs on same domain. Standalone RIaS runs on a different domain or sub-domain. A standalone RIaS is highly recommended.

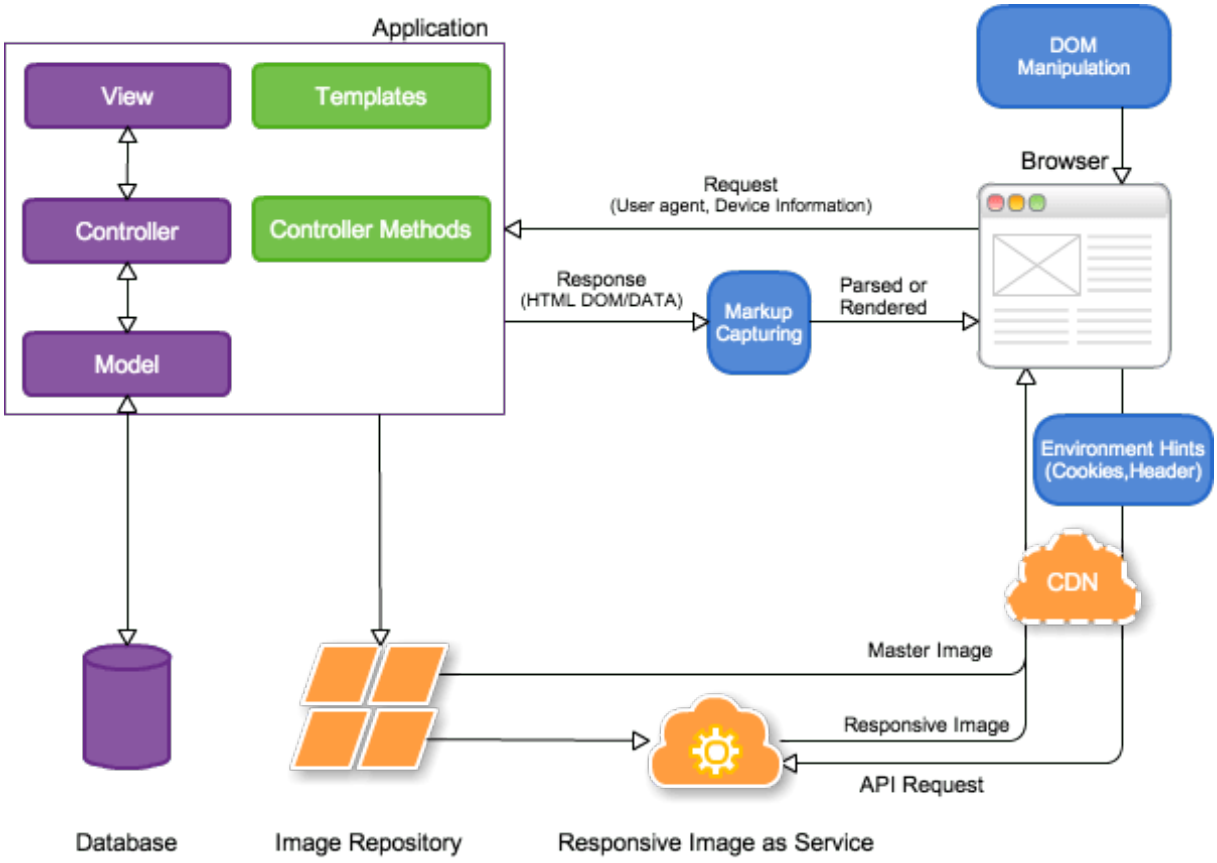

Figure 1: Responsive Image as Service (RIaS) is a key component of any responsive image solution

REST API

Normally, a RIaS API can use Path parameters, Query parameters, or both to enable dynamic image manipulation. As a rule of thumb, most of RIaS APIs require source file path as Path parameter. Depending on RIaS solution, optional parameters describing re-size, crop, format, device-pixel-ratio, breakpoints etc can be passed either as Query or Path parameters.

This is example Image API using Path parameter and Query parameter.

```
http://image.my-site.com/static/media/myimage.jpg?width=500&height=400
```

This is equivalent Image API call using only Path parameters.

```
http://image.my-site.com/500/400/static/media/myimage.jpg
```

Cookies and Device Detection

Rather than REST API, some RIaS solutions relies solely on cookies or device detection for scaling images. Other solutions follow a mixed approach - REST API with cookies.

On first page load a client-side JavaScript will set a cookie. Environment hints (EH) like viewport, device-pixel-ratio, breakpoints, orientation are stored in cookie as properties. If cookie domain is scoped correctly, RIaS can read the cookie and scale images according to cookie properties. Overall workflow is described in this image.

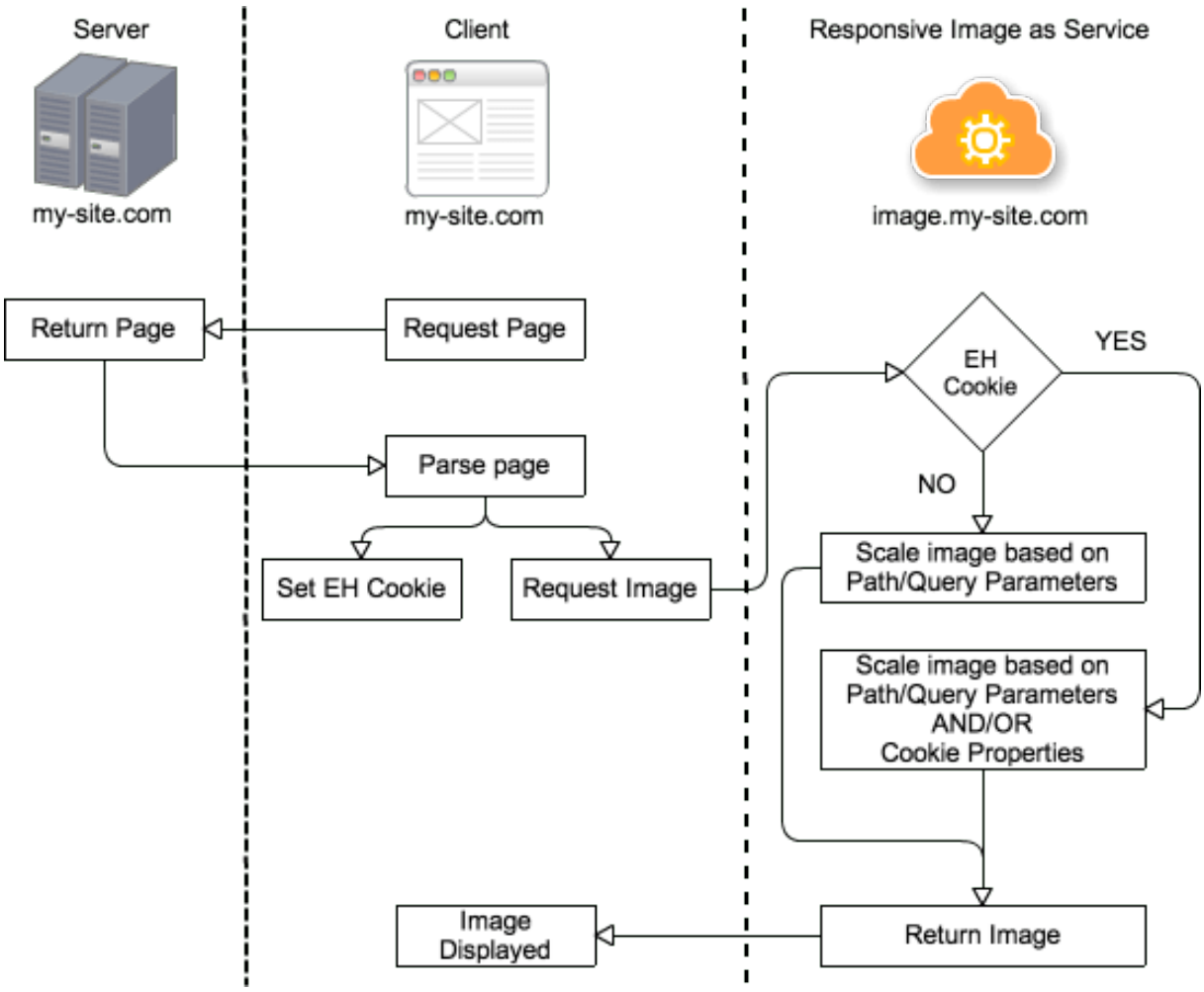

Figure 2: Cookies based image scaling.

In case of cookie based solution, first page view can be visually different from subsequent page views. This can happen because on first page view preloader may have already started downloading images before cookie was set.

Although for a given image cookies can be different for different user agents, image URLs will be same (not unique). Hence CDN or proxy caching may not work with cookie based RIaS. This defeats the purpose. To disable the caching by proxy or CDN, RIaS is required to set `Cache-Control` header **`private`**. This enables images to be cached in client-browser but not on proxy or CDN.

When setting cookie, cookie `domain` should be scoped properly. If JavaScript sets cookie `domain` as `my-site.com` then cookie can actually be read by the RIaS on any sub-domain `*.my-site.com` .

How to choose a RIaS?

- Does service support RESTful or REST like Image APIs? How we can integrate without vendor lock-in issues?
- Does service store master images or require sync with other image repository? Otherwise, can it directly access and use images stored in existing image repository such as DAM, High-speed blob storage (like Amazon S3 or Azure Blob Storage). Also how service will connect with private image repositories using authentication bridge?
- Does service support image resize, crop, re-compress, reformat the image file on the fly?
- Does service support detection of environmental conditions such as device type, device resolution, network connectivity? Otherwise, does service expect environment hints via cookies or custom headers?
- Does service require any specific server-side or client-side library?
- Does service respond to content negotiations headers such as `Accept` to server WebP?
- Does service integrates seamlessly with any CDN? Or, does it restrict to use a particular CDN?
- Does service expect to forward cookies and query string from CDN?
- Does service provide appropriate control in terms of cache flushing and expiry? For instance, is it possible to set the minimum TTL or to flush cache in API call when we request a responsive image?
- Does service render responsive images using external images source such as Flickr etc?
- Does service offer a good response time in general as well under high-load conditions? As responsive images are generated on the fly, ability to generate a large number of responsive images at beginning or during high-load conditions will be critical. Otherwise some kind of warm-up approach coupled with caching will be required.
- Does service supports both HTTP/HTTPs connection?

Solutions

3rd party solutions

There are many 3rd party cloud hosted RIaS solutions. Most of these 3rd party RIaS solutions allow access to responsive images via RESTful or REST like Image APIs.

One of the biggest challenge with these services is fragmentation around the REST convention. This creates platform and vendor lock-in issues. Again this can be avoided using a decoupled implementation approach.

List of services

Adobe Scene7

Adobe Scene7 is one of the most sophisticated dynamic media (both image and video) solutions in the market. It can deliver optimised and personalised media assets for web, mobile, social, email, and print. In fact Scene7 is more than just RIaS but it is costly. For enterprise customer with large investment in DAM solutions, Adobe Scene7 is highly recommended.

```
http://crc.scene7.com/is/image/demo/bedroom.tif?wid=800&hei=300&scl=02
```

RESRC.IT

ReSRC delivers responsive images on-demand, direct from the cloud. ReSRC can be used as either a responsive image as service or a complete responsive image solution with help of ReSRC's JavaScript plugin.

```
http://app.resrc.it/S=W800M,H800M,PD1.3/C=W800,H450,X50,Y0/http://www.your
    -site.co/image.jpg
```

Using JavaScript plugin, ReSRC can support the popular Mobile First (LQIP) approach, HiDPI (retina) devices and most appropriate responsive image in single request.

Mobify

Mobify's image resizing backend (`ir0.mobify.com`) can manipulate the width, height, file format, and quality of any image directly using following API request format. Mobify can't crop or create different aspect ratios from the original image.

```
http://ir0.mobify.com/<format><quality>/<maximum width>/<maximum height>/<
    url>
http://ir0.mobify.com/c<hours>/<format><quality>/<maximum width>/<maximum
    height>/<url>
```

```
http://ir0.mobify.com/c4/webp50/800/http://cdn.mobify.com/mobifyjs/
    examples/assets/images/forest.jpg
```

All requests resizing backend are cached by Mobify's CDN.

CDN

Many CDN vendors offer support for dynamic image manipulation including network connectivity based optimisation. Normally standard quality images are embedded in page using `img` tag point-

ing to CDN. After this all magic happens on CDN side. CDN will generate all common variants based on how people view images on the web[1] and server them as appropriate.

An image variant differs in size, quality or format. Depending on user-agent and network conditions, you will get different variants of an image. Nonetheless image URL remains same.

Akamai

Akamai's Adaptive Image Compression service can dynamically resize, crop, re-compress, reformat the image file on the fly. It can also detect browser support for advanced image formats (WebP and JPEG XR) and serve appropriate image variant.

CloudFlare

CloudFlare offers similar features Mirage and Polish optimise image on the fly. Mirage can detect screen size, resolution and connection speed to deliver the best image for the device. On the other hand, Polish can perform both "lossless" and "lossy" image optimisation to reduce image sizes on-average by 35%.

SENCHA.IO SRC

Sencha.io Src offers really powerful REST API to manipulate images on the fly.

```
http://src[shard].sencha.io
       [/flush]
       [/data]
       [/format[quality]]
       [/orientation]
       [/width[/height]]
       /url
```

```
http://src.sencha.io/jpg50/800/400/http://sencha.com/files/u.jpg
```

Apart from core image manipulation functionality it also provides ability to perform complex formulaic operations.

For environment hints Sencha.io Src uses a client-side measurement library to detect the browser's screen dimensions and set them in a cookie (on `src.sencha.io` domain).

Formulaic operations can be combined with client-side measurement. For instance, following `img` markup using client-side measurement `sw` (`screen.width`) and then deduct 16 pixels,

[1] How does Akamai crop or resize images so fast for Facebook?

```
<img
  src='http://src.sencha.io/sw-16/http://sencha.com/files/u.jpg'
  alt='Client-measurement, reduced'
/>
```

Sencha.io Src also supports domain sharding and cache flushing.

PicarIS

PicarIS offers online image manipulation using URL based API. PicarIS can do more than just re-sizing, cropping, scaling, re-formating , etc. In terms of features there is a lot of overlap with Scene7 like on-demand adding text, changing colour or texture.

```
http://www.picarisplatform.com/picaris/getimage.ashx?ft=1&fn=sfeer3&w=800&
    h=400&q=50&sr=2
```

Pixtulate

Pixtulate offers a mature API for on-demand re-sizing, cropping, scaling, re-formating using their own global CDN.

```
http://demo.api.pixtulate.com/demo/large-4.jpg?tlc=-450,300&w=400&h=100&
    dpr=1.75
```

Image Optimizer

Image Optimizer from WhateverWeb scale images for different layouts and devices dynamically, using breakpoints or API parameters.

For dynamic scaling it relies on client-side environment hints.

```
http://img.demo.wew.io/vpw_320/bp_n/pc/w_30/m_46/n_98/http://demo.wew.io/
    demo_img.jpg
```

Self-hosted solutions

Adaptive Images

Adaptive Images is yet another responsive image service. Adaptive Images is based on PHP backend and requires environment hints using cookies. Adaptive Images is relies on cookies for re-sizing parameters.

ImageResizer

ImageResizer offers simple and effortless way to create responsive images[2]. Apart from core image manipulation ImageResizer can perform focal point detection and auto-cropping. ImageResizer is designed for Windows environment and runs in standalone mode under IIS.

ImageResizer can also be used in embedded mode which means in-process inside an existing ASP.NET application. All known ASP.NET content management systems support ImageResizer in embedded mode.

thumbor

thumbor enables smart on-demand crop, resizing and flipping of images. thumbor is a Python solution and it deploys state-of-the-art focal point detection (face as well as feature detection) algorithms for better cropping and resizing.

IIPImage Server

IIPImage can generate responsive images on-demand using its core image resize API. IIPImage API can also perform image processing if required.

```
http://merovingio.c2rmf.cnrs.fr/fcgi-bin/iipsrv.fcgi?FIF=PIA03883.pyr.tif&
    WID=800&RGN=0.1,0.6,0.9,0.55&QLT=60&CVT=jpeg
```

Server Side Responsive Images

Server Side Responsive Images is quite similar to Adaptive Images. Rather than cookies it relies on device detection to generate responsive images. For device detection it uses WURFL database and Java library.

Bespoke Solution

Creating a bespoke RIaS solution is quite easy.

1. Pick a Imaging Library. Between ImageMagic, GD, ImageResizer and PIL[3], ImageMagic is most widely supported.
2. Pick a micro-framework like Sinatra, Express or Flask to implement API interface.
3. Pick a RESTful Image API convention, preferably RESTful Image API Specification.

[2] Effortless responsive images using ImageResizer
[3] What is the best image manipulation library?

4. Implement the bespoke RIaS and deploy it on AWS. Make provision for auto-scaling and load-balancing.

OR just wait for my next post.