Varnish cache buster add-on for Statamic

Abhishek Tiwari

Citation: *A. Tiwari*, ”Varnish cache buster add-on for Statamic”, Abhishek Tiwari, 2014. doi:10.59350/18kwt-y0f02

Published on: September 26, 2014

Few months back I managed to get Statamic working with Varnish. Setting up Varnish for Statamic was quite easy and my website performance improved drastically (you can see load and performance testing benchmarks here). One of the biggest challenges was Varnish cache invalidation on content update. Every time, Statamic content was updated I have to invalidate cache manually using curl requests.

Varnish cache buster add-on

To avoid cache invalidation troubles, I decided to write a simple Varnish cache buster add-on for Statamic. This add-on requires two new Statamic settings.

```
purge_type: BAN
varnish_hook: true
```

This add-on utilises the control panel publish hook and PHP curl functionality . When you save and publish content via admin, it results into call to control panel publish hook which in-turn runs add-on's own custom hook. Custom hook gets full URL for the page and execute appropriate curl request.

```
<?php

// Goes in _add-on/varnish_cache_buster/hooks.varnish_cache_buster.php

class Hooks_varnish_cache_buster extends Hooks {

  public $meta = array(
    'name'       => 'Cache Buster for Varnish',
    'version'    => '1.0.0',
    'author'     => 'Abhishek Tiwari',
    'author_url' => 'http://abhishek-tiwari.com'
  );

  protected $file_updated;

  public function control_panel__publish($data) {
    if (Config::get('varnish_hook') == true) {
      $this->file_updated = $data['file'];
      $this->runHook('invalidate', 'replace', $this->file_updated, $this->
          file_updated);
    }
    return $data;
  }

  public function varnish_cache_buster__invalidate($path) {
    $this->invalidateVarnish($path);
    return $path;
  }
  private function invalidateVarnish($path) {
    $this->curlThisURL($this->getFullURL($path));
    return;
  }

  private function curlThisURL($url) {
    $curl    = curl_init();
    curl_setopt($curl, CURLOPT_URL, $url);
    curl_setopt($curl, CURLOPT_CUSTOMREQUEST, Config::get('purge_type'));
    curl_exec($curl);
    curl_close($curl);
    return;
  }

  private function getFullURL($path) {
    $path  = Path::clean(Path::resolve(str_replace('/'.Config::
        getContentRoot(), '', $path)));
    $path  = str_replace('/page.'.Config::getContentType(), '', $path);
    $path  = str_replace('.'.Config::getContentType(), '', $path);
    $url   = Config::getSiteURL().$path;
    return $url;
  }
}
```

Prerequisite

One key prerequisite for this add-on is Varnish setup with specific BAN configuration as described here.

```
  # Verify the ACL for an incoming ban request and handle it.
  if (req.request == "BAN") {
    if (!client.ip ~ purge) {
      # Not from an allowed IP? Then die with an error.
      error 405 "This IP is not allowed to send PURGE requests.";
    }
    ban("req.http.host == " +req.http.host+" && req.url ~ "+req.url+"$");
    error 200 "Ban added";
  }
```

Soon, I will be pushing a more complete version of add-on to my Github repository. For now this mini add-on will get you started.