AEM Gems - Dispatcher New Features and Best Practices

Dominique Pfister | Senior Computer Scientist

Prerequisites

- Basics of HTTP protocol
- Familiarity with `dispatcher.any` configuration file

- More info on how the dispatcher caching works: https://github.com/cqsupport/webinar-dispatchercache
- Previous GEM session: Dispatcher Caching and Optimizations: https://docs.adobe.com/ddc/en/gems/dispatcher-caching---new-features-and-optimizations.html

What's Covered

- Refresher: what does the dispatcher cache?

- New dispatcher features available in version 4.2.2
  - Response header caching
  - TTL-based invalidation and invalidation throttling
  - Regular expressions in `/filter` and `/cache` section
  - Various small things

- Apache specific features
  - Use Apache `Defines`
  - Dump parsed configuration
  - Per host configuration

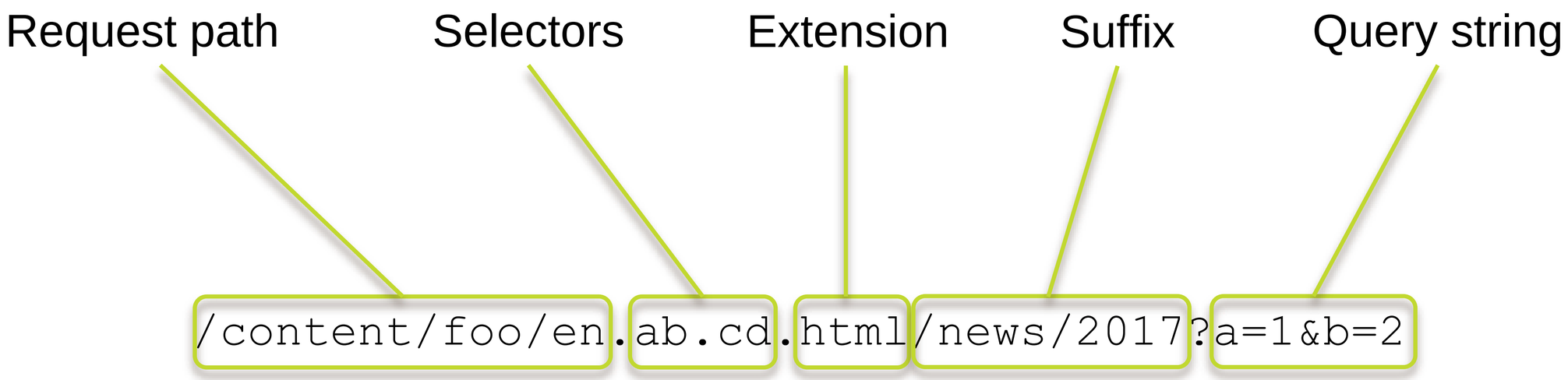
Planning

- Next dispatcher release 4.2.3 in December 2017

- Adobe will drop support for Apache 2.2 after December 2017, and recommends to upgrade to Apache 2.4.
- In case you stay with Apache 2.2 beyond December 2017, it is recommended to use a Linux distribution that commits to Apache 2.2 security updates.
- The latest dispatcher release of 2017 for Apache 2.2 will still be available for download until end of 2018.

Refresher: what does the dispatcher cache?

URL Decomposition



What does the dispatcher cache?

#1

The URL must be allowed by the `/cache` rules and `/filter` sections of `dispatcher.any`

What does the dispatcher cache?

#2

The URL must have a file **extension**

`/content/foo.html`	cached
`/content/foo`	not cached

Adobe

What does the dispatcher cache?

#3

The URL must not contain any query string parameter

`/content/foo.html`	cached
`/content/foo.html?queryparam=1`	not cached

What does the dispatcher cache?

#4

If the URL has a suffix, then that suffix must have a file extension

`/content/foo.html/suffix/path.html`	cached
`/content/foo.html/suffix/path`	not cached

What does the dispatcher cache?

#5

The HTTP method must be GET

`GET /foo.html`	cached	
`HEAD /foo.html`	not cached	delivered
`POST /foo.html`	not cached	

What does the dispatcher cache?

#6

The HTTP response status must be 200 OK

`HTTP/1.1 200 OK`	cached
`HTTP/1.1 500 Internal Server Error`	not cached
`HTTP/1.1 404 Not Found`	not cached

What does the dispatcher cache?

#7

The HTTP response must not contain a header that forbids caching

`Cache-Control: no-cache`	**not cached**
`Pragma: no-cache`	not cached
`Dispatcher: no-cache`	not cached

What does the dispatcher cache?

Suffix Exception #1

Children of cached resources will not be cached

`/content/foo.html`	already cached
`/content/foo.html/suffix/path.html`	not cached

What does the dispatcher cache?

Suffix Exception #2

Parents of cached resources will be cached

~~`/content/foo.html/suffix/path.html`~~	deleted
`/content/foo.html`	**cached**

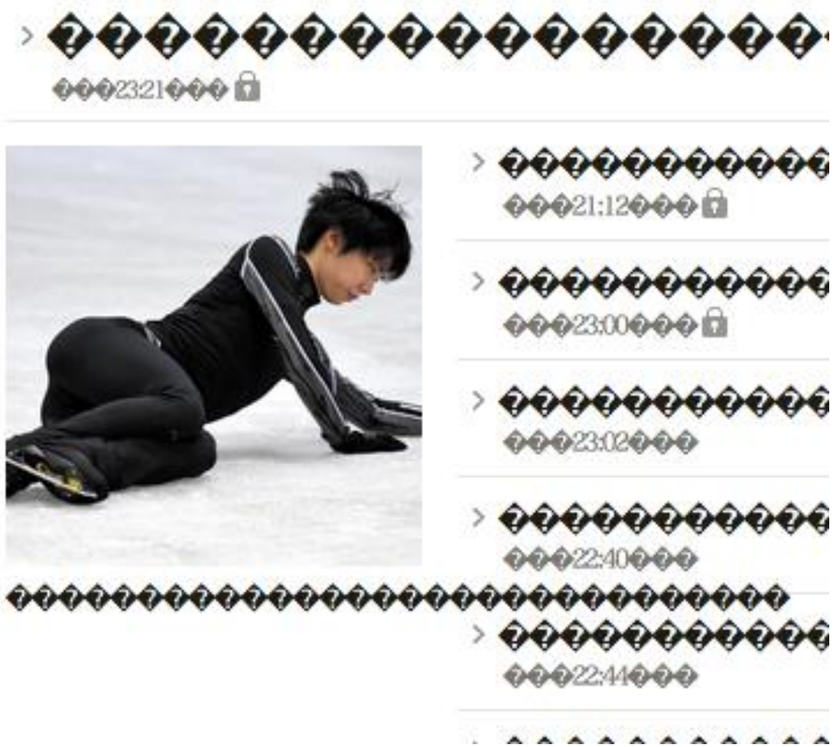
Response Header Caching

Evolution Bureau

Response Header Caching - Why?

- HTML page encodings might get lost

```
Content-Type: text/html;charset=shift_jis
```



Response Header Caching - Why?

- Some headers are indispensable for security reasons, e.g.:

```
Content-Disposition: attachment; filename=f.txt
```

- With that header, content will not automatically be displayed inline in the browser
- Abusing JSONP with Rosetta Flash:
  https://miki.it/blog/2014/7/8/abusing-jsonp-with-rosetta-flash/
- Dispatcher configuration in distribution comes with preconfigured set of headers

TTL-based invalidation and invalidation throttling

TTL-based invalidation

- With “statfile” invalidation, and the following configuration:

```
/invalidate {
    /0001 { /type "deny" /glob "*" }
    /0002 { /type "allow" /glob "*.html" }
}
```

every flush request will invalidate **all** HTML pages

- Now, some pages or sections in your content might stay valid for some time, *regardless* of other modifications

TTL-based invalidation (continued)

- On the server side, you can set response headers to mark such pages:

```
Cache-Control: max-age=<time>
Expires: <date>
```

- And enable TTL on the dispatcher:

```
/enableTTL “1”
```

- The dispatcher will look for those response headers and automatically re-fetch a cached item, when it expires.
- That setting takes precedence over “statfile” invalidation ⚠

TTL-based invalidation (continued)

- You can set mentioned headers manually or use the **Dispatcher TTL** component.
- This component allows to set a `max-age` or `expiry` for all pages with a path matching a definable pattern, e.g.:

```
filter.pattern="[/content/site/services/(.*)]”
max.age="60"
```

- More info: https://adobe-consulting-services.github.io/acs-aem-commons/features/dispatcher-ttl/index.html

Invalidation throttling

- In a setup where activations happen in batches, the cache can keep getting invalidated ("thrashing"). This will increase load on the backend.

- In such a situation it is advisable to throttle invalidations in the dispatcher:

```
/gracePeriod <seconds>
```

- With that setting, the dispatcher will keep stale cache entries for that number of seconds

Regular expressions in `/filter` and `/cache` section

Evolution Bureau

Regular expressions in `/filter` section

- Using globs alone, some rules can be quite hard to define
- In the old days, when you had to match the request line, to e.g. allow images:

```
/filter {
    /0001 { /type "deny" /glob "*" }
    /0002 { /type "allow" /glob "* *.gif *" }
    /0003 { /type "allow" /glob "* *.png *" }
    /0004 { /type "allow" /glob "* *.jpg *" }
    /0005 { /type "allow" /glob "* *.jpeg *" }
}
```

Of course, above example is bad for security reasons ⚠

Regular expressions in `/filter` section (continued)

- With the "structured" approach, introduced in version 4.1.5:

```
/filter {
    /0001 { /type "deny" /glob "*" }
    /0002 { /type "allow" /extension "gif" }
    /0003 { /type "allow" /extension "png" }
    /0004 { /type "allow" /extension "jpg" }
    /0005 { /type "allow" /extension "jpeg" }
}
```

That is safe, but still, 4 lines, really?

Regular expressions in `/filter` section (continued)

- Enter regular expressions:

```
/filter {
    /0001 { /type "deny" /glob "*" }
    /0002 { /type "allow" /extension '(gif|png|jpe?g)' }
}
```

Planned addition to `/cache` section in 4.2.3

- Requests with a suffix can lead to directories in the cache, that need to be removed again to cache the parent resource. If we want to forbid that, we might define:

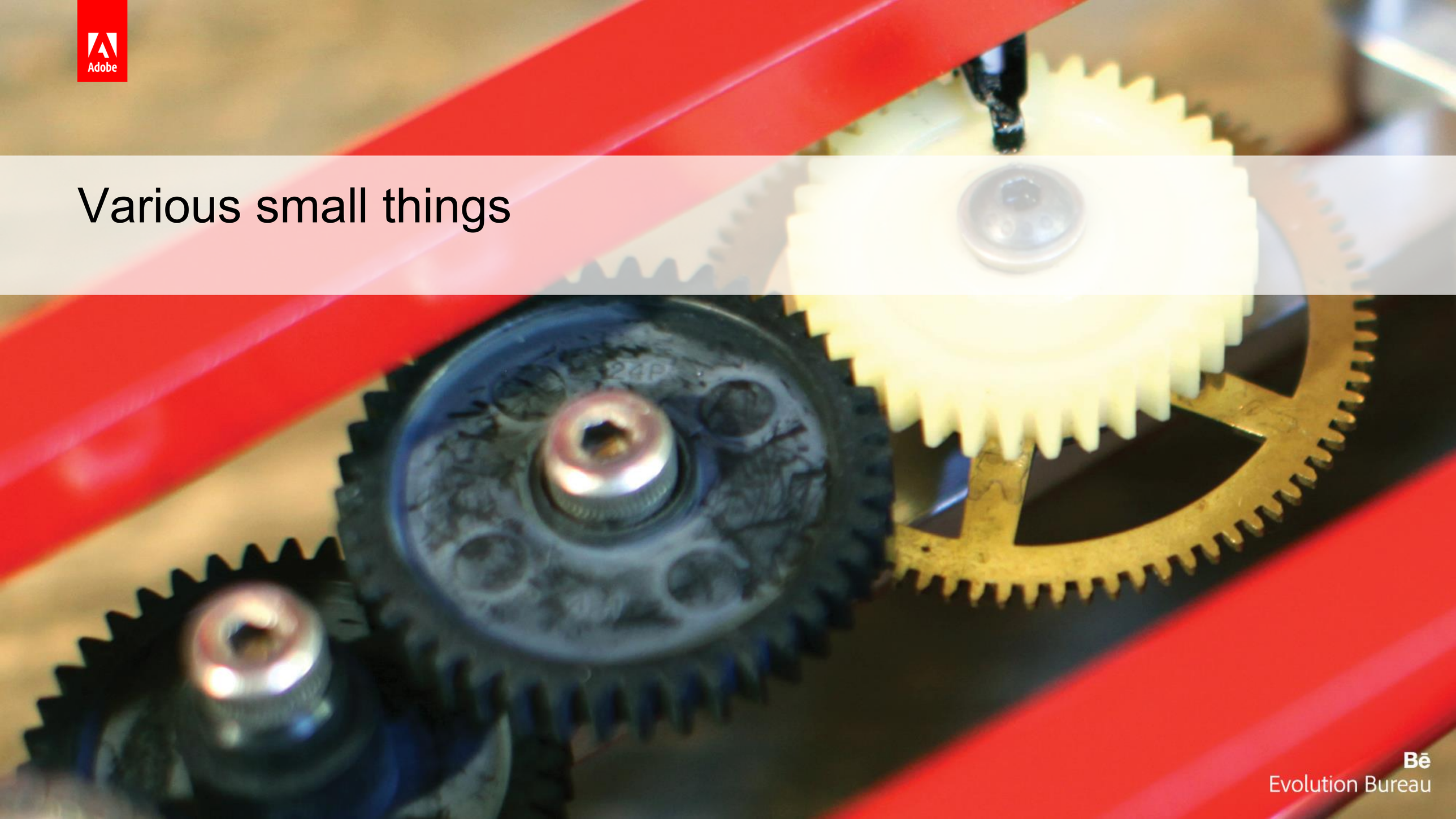
```
/rules {
    /0001 { /type "allow" /glob "*.html" }
    /0002 { /type "deny" /glob "*.html/*" }
}
```

- Structured elements in cache rules will make this simpler:

```
/rules {
    /0001 { /type "allow" /extension "html" /suffix "" }
}
```

- This will also simplify selector white listing.

Various small things



Bē
Evolution Bureau

Various small things

- From time to time, dispatcher log will display the cache hit ratio:

```
[Fri Nov 10 14:26:16 2017] [I] [pid 1488] "GET /content/we-retail/us/en.html" - - 0ms [website/-]
[Fri Nov 10 14:26:16 2017] [I] [pid 1488] Current cache hit ratio: 99.95 %
```

- If cache hit ratio is lower than expected, look at your cache rules!

- And it will show what farm/backend it selected:

```
[Fri Nov 10 14:45:28 2017] [I] [pid 2464] "GET /" 302 - 4ms [website/publish]
[Fri Nov 10 14:45:28 2017] [I] [pid 2464] "GET /index.html" 302 - 9ms [website/publish]
[Fri Nov 10 14:45:28 2017] [I] [pid 2464] "GET /content/we-retail.html" 302 - 39ms [website/publish]
[Fri Nov 10 14:45:28 2017] [I] [pid 2464] "GET /content/we-retail/us/en.html" - - 1ms [website/-]
```

- This can help in troubleshooting if requests go to an unexpected farm

Various small things (continued)

- Keeping connections to backends alive is now the default.
- This reduces DNS lookups, time for initial TCP/IP handshake, and is particularly valuable in secure setups where the added SSL handshake consumes even more time.

- Previously, when a POST to some backend failed, the client received a gateway error (502), because the request body was no longer available.
- Now, the dispatcher will store away small request bodies, so it can retry.

Various small things (continued)

- Wildcard includes are now sorted, so if we have:

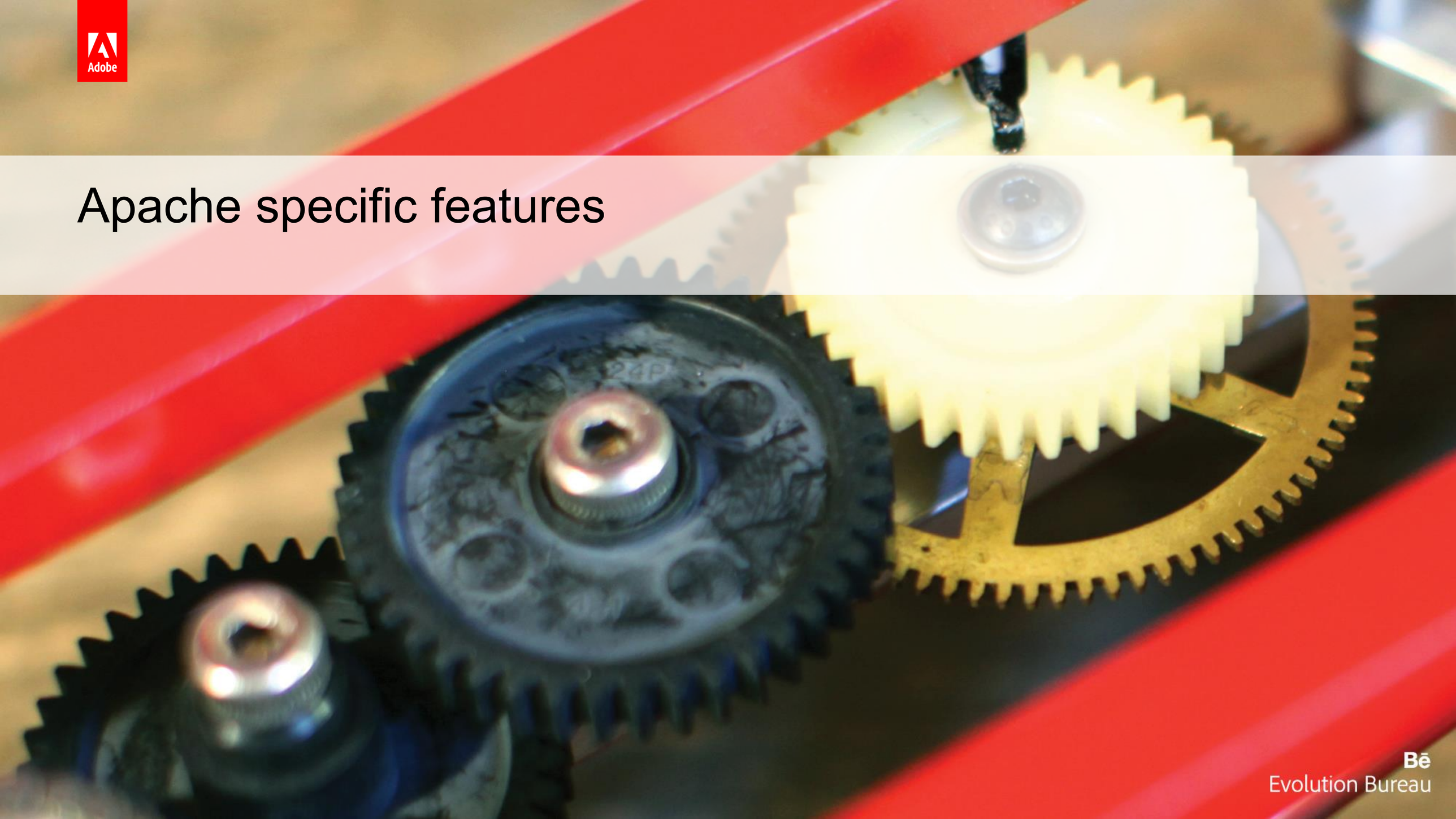
```
/farms {
    /www { ... }
    /docs { ... }
    /kb { ... }
}
```

- We can put each farm definition in a separate file prefixed by its **order**:

```
$ ls farms
00-www.any
01-docs.any
02-kb.any
```

```
/farms {
    $include "farms/*.any"
}
```

Apache specific features



Use Apache `Defines`

- You can `Define` variables in your Apache configuration:

```
Define SITE mysite
Define RUN_MODE author
```

- And then use those variables in your dispatcher configuration:

```
/cache {
    /docroot “/mnt/var/html/${SITE}”
    $include “/etc/dispatcher/${RUN_MODE}-rules.any”
}
```

- This will help reusing scripts in your environment and avoid copy-paste errors

Dump parsed configuration

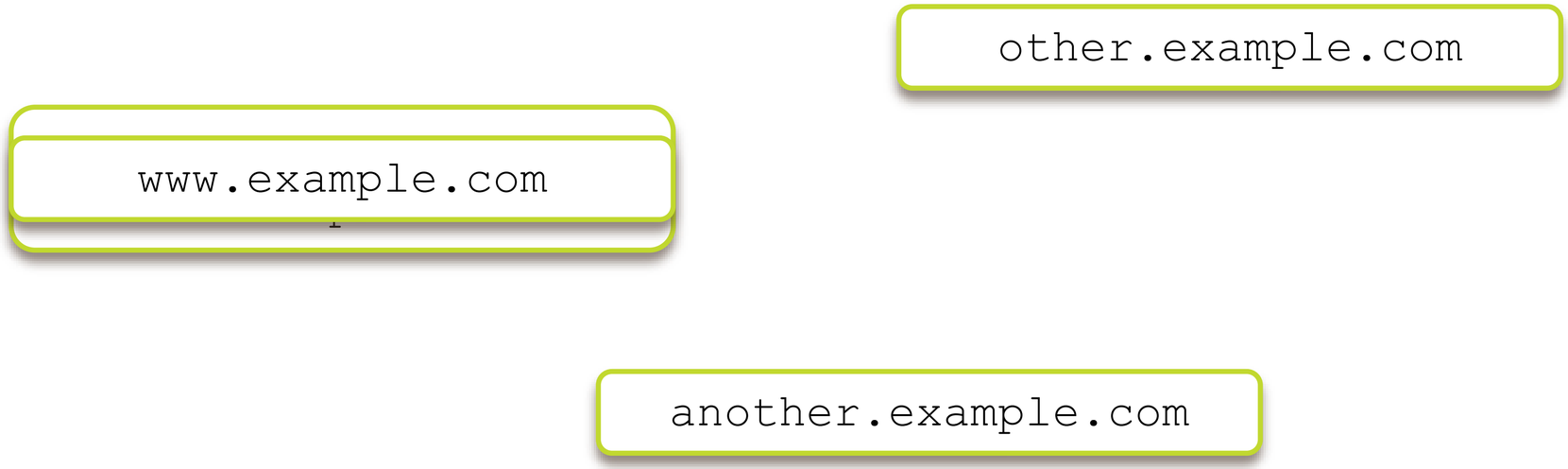
- Pass `-D DUMP_ANY` to the syntax check option `-t` on the command line:

kneipix:~/Applications/apache (bash)

```
[kneipix apache] bin/apachectl -t -D DUMP_ANY
# Dispatcher configuration: (/Users/dpfister/Applications/apache/conf/dispatcher.any)
/farms {
  /website {
    /virtualhosts {
      "*"
    }
    /clientheaders {
      "*"
    }
    /renders {
      /publish {
        /hostname "kneipix"
        /port "4503"
      }
    }
    /filter {
      /0001 {
        /type "deny"
        /glob "*"
      }
```

Per host configuration

- If you have a lot of farms in your `dispatcher.any`, a single file becomes unmaintainable. You could split into multiple files and `$include` them.
- Still, at some point, adding a new include or redefining matching `virtualhosts` becomes error-prone:



Per host configuration (continued)

- Instead of selecting the right farm based on the host header, let the `<VirtualHost>` definition point to the right dispatcher configuration:

```
<VirtualHost *:443>
   ServerName www.example.com
   ServerAlias *.example.com
   DispatcherConfig sites/default.any
</VirtualHost>

<VirtualHost *:443>
   ServerName other.example.com
   DispatcherConfig sites/other.any
</VirtualHost>
```

Demo

MAKE IT AN EXPERIENCE