

<ASSET PIPELINE/>

Presented by David Estes (@davydotcom) #asset-pipeline



What is Asset-Pipeline?

A library for both rapid development iteration and optimized production handling of static and not-so-static assets.

What will we cover?

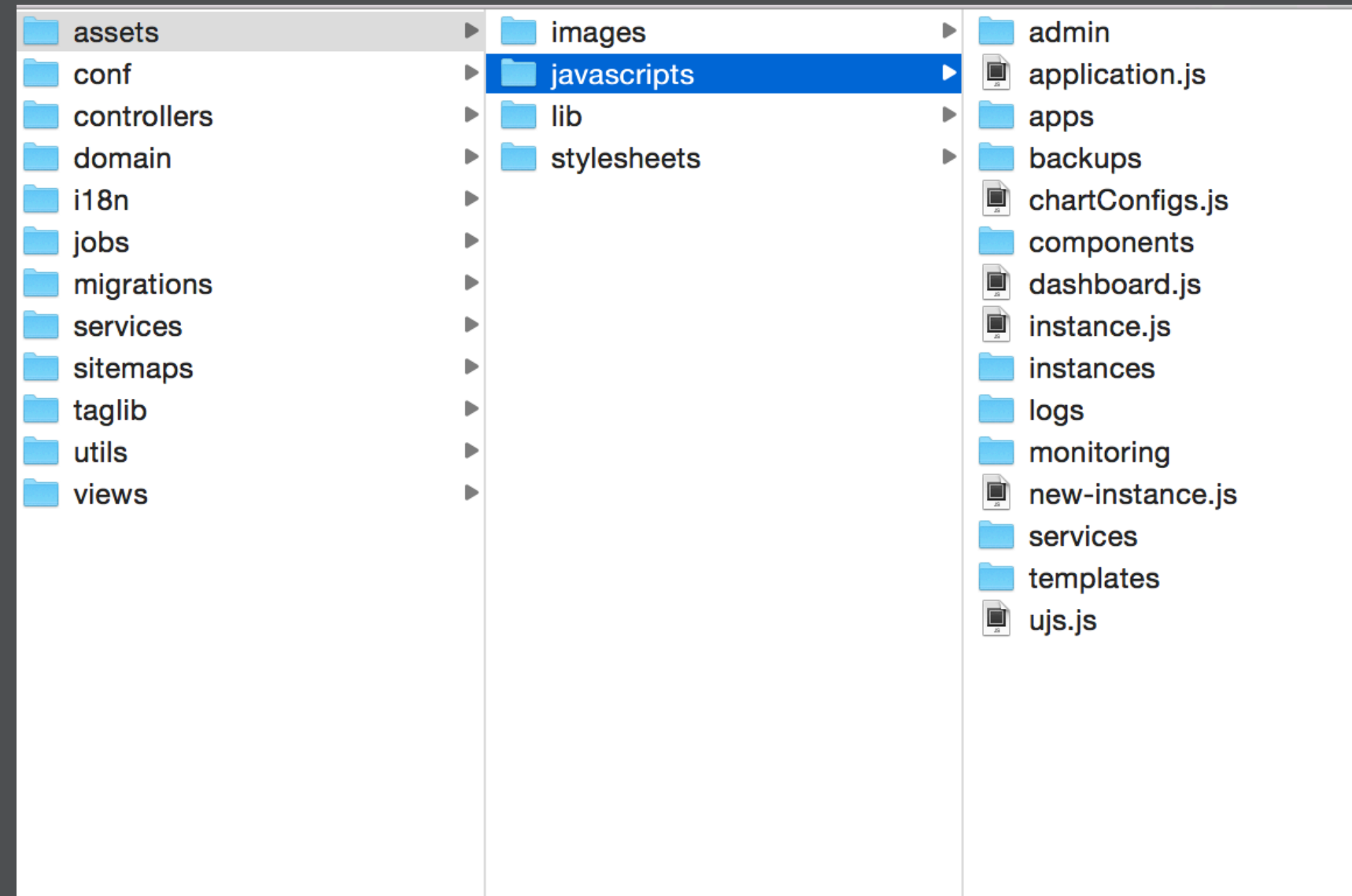
- Development cycle: On the fly processing and transpiling of assets
- Popular extension plugins
- Production tuning with Grails
- Digesting and cache headers
- Other frameworks

Whats new in Version 2.x

- 2-10x Faster than the 1.x.x Series
- Resolvers
- Easier Extensibility
- Framework Decoupling (Works with Grails3, Ratpack, Spring Boot, Gradle, etc.)
- Improved Relative Path recalculations

The 'assets' folder

- First subfolder is for organizational use. (i.e. images, fonts, stylesheets, javascripts)
- This is never used in the request path and is essentially flattened out



While the file is stored in `assets/javascripts/application.js`, the request is simply made at: `/assets/application.js`

APPLICATION STATUS

Environment: development
App profile: web
App version: 0.1
Grails version: 3.0.3
Groovy version: 2.4.3
JVM version: 1.8.0_45
Reloading active: true

ARTEFACTS

Controllers: 0
Domains: 0
Services: 3
Tag Libraries: 14

INSTALLED PLUGINS

restResponder - 3.0.3
scaffolding - 3.0.3
core - 3.0.3
eventBus - 3.0.3
dataSource - 3.0.3

Welcome to Grails

Congratulations, you have successfully started your first Grails application! At the moment this is the default page, feel free to modify it to either redirect to a controller or display whatever content you may choose. Below is a list of controllers that are currently deployed in this application, click on each to execute its default action:

Available Controllers:

| | | | | | | |
|-------------------------------------|------------|--------------|-------------------|--------------|-------------|--|
| jquery-2.1.3.js?compile=false | 200 | sc... | (index)... | 24... | 99... | |
| application.css?compile=false | 200 | st... | (index)... | 67... | 19... | |
| application.js?compile=false | 200 | sc... | (index)... | 85... | 9 ms | |
| grails_logo.png | 200 | png | (index)... | 10... | 12... | |
| spinner.gif | 200 | gif | (index):1 | 2.... | 5 ms | |
| in-page-script.js | 200 | sc... | conten... | (fr... | 2 ms | |

9 requests | 275 KB transferred | Finish: 335 ms | DOMContentLoaded: 333 ms | Load: 352 ms

×

Headers

Preview

Response

Cookies

Timing

▼ General

Remote Address: [::1]:8080

Request URL: http://localhost:8080/assets/application.js?compile=false

Request Method: GET

Status Code: ● 200 OK

▼ Response Headers

view source

Cache-Control: no-cache, no-store, must-revalidate

Content-Length: 599

Content-Type: application/javascript

Date: Sat, 18 Jul 2015 22:11:34 GMT

Expires: Thu, 01 Jan 1970 00:00:00 GMT

Pragma: no-cache

Resolvers

The means with which the asset-pipeline resolves files requested as well as imported. There are several base resolver types:

- Filesystem
- Jar
- Classpath

NOTE: Resolvers are only used in development mode

Grails Resolvers

Resolvers are registered for the main grails-app/assets folder and each plugin.

- * **Binary Plugins:** META-INF/assets. (pre-flattened)
- * **Source Plugins:** \$pluginPath/grails-app/assets
- * **Binary Dependencies:** META-INF/static and META-INF/resources

Gradle Resolvers

Customizable but defaults to the `src/assets` folder.

```
assets {  
    sourceDir = 'src/assets'  
    compileDir = "$buildDir/assets"  
}
```

Bundling

- Directives
- Encoding

What are Directives

File annotations that allow for the inclusion of other files, as well as the assignment of file options.

- `//= require jquery`
- `//= require_tree .`
- `//= require_self`
- `//= encoding UTF8`

Bundling Javascript

Sample application.js file taken from rails:

```
// This is a manifest file that'll be compiled into application.js.
//
// Any JavaScript file within this directory can be referenced here using a relative path.
//
// You're free to add application-wide JavaScript to this file, but it's generally better
// to create separate JavaScript files as needed.
//
//= require jquery-2.1.3.js
//= require_tree .
//= require_self

if (typeof jQuery !== 'undefined') {
  (function($) {
    $('#spinner').ajaxStart(function() {
      $(this).fadeIn();
    }).ajaxStop(function() {
      $(this).fadeOut();
    });
  })(jQuery);
}
```

Bundling Stylesheets

Sample application.css taken from rails:

```
/*
 * This is a manifest file that'll be compiled into application.css, which will include all the files
 * listed below.
 *
 * Any CSS file within this directory can be referenced here using a relative path.
 *
 * You're free to add application-wide styles to this file and they'll appear at the top of the
 * compiled file, but it's generally better to create a new file per style scope.
 *
 *= require main
 *= require mobile
 *= require_self
 */
```

Encoding

Source encoding can be specified within each individual javascript file

```
//=encoding UTF-8
```

Output encoding can be customized with param:

```
http://localhost:8080/assets/application.js?encoding=utf-8
```

Taglibs

- Javascript
- Stylesheets
- Images
- Deferred scripts

Sample Grails Layout

```
1  <!doctype html>
2  <html lang="en" class="no-js">
3      <head>
4          <asset:stylesheet src="application.css"/>
5          <asset:javascript src="application.js"/>
6      </head>
7      <body>
8          <div id="grailsLogo" role="banner"><a href="http://grails.org"><asset:image src="grails_logo.png"
9              alt="Grails"/></a></div>
10         <g:layoutBody/>
11         <div class="footer" role="contentinfo"></div>
12         <div id="spinner" class="spinner" style="display:none;"><g:message code="spinner.alt" default="
13             Loading&hellip;" /></div>
14     </body>
15 </html>
```

NOTE: No Need to specify assets/ folder in path

Debug Mode

In Development the taglib will include each file individually rather than bundled. This makes it easier to debug your javascript in real time

APPLICATION STATUS

Environment: development
App profile: web
App version: 0.1
Grails version: 3.0.3
Groovy version: 2.4.3
JVM version: 1.8.0_45
Reloading active: true







ARTEFACTS

Controllers: 0
Domains: 0
Services: 3

Welcome to Grails

Congratulations, you have successfully started your first Grails application! At the moment this is the default page, feel free to modify it to either redirect to a controller or display whatever content you may choose. Below is a list of controllers that are currently deployed in this application, click on each to execute its default action:

Available Controllers:

| Name | St... | Type | Initiator | Size | Time | Timeline |
|---|-------|-------|----------------------------|--------|-------|---|
|  jquery-2.1.3.js?compile=false | 200 | sc... | (index)... | 24... | 32... |  |
|  application.js?compile=false | 200 | sc... | (index)... | 85... | 11... |  |
|  in-page-script.js | 200 | sc... | conten... | (fr... | 2 ms |  |

Deferred Scripts

Added to facilitate easier transition from the Resources plugin but not recommended.

```
<asset:script type="text/javascript">  
  console.log("Hello World");  
</asset:script>  
<asset:script type="text/javascript">  
  console.log("Hello World 2");  
</asset:script>
```

Now to render the output of these scripts simply use the following:

```
<asset:deferredScripts/>
```

Scoping Javascript to a Page

Use a class or data-page attribute on your body tag to selectively trigger javascript. This allows you to seperate your javascript from your view and keep it all bundled in one file.

On the fly

- Why not watched?
- Cache
- Advantages over Gulp or Grunt

Why not Watched Files?

- Slow
- Delay between saving and being able to refresh your browser
- Gets exponentially worse the bigger the project
- Doesn't stack well with other watch type transpilers in a project

Instant Gratification

When a file is changed, you can refresh your browser and see your changes instantly. Allows for a faster iteration.

Development Runtime Cache

Processed files are cached in memory, and given a cache dependency tree. Asset-Pipeline serves from this cache unless the file has changed since the last cache.

Gulp and Grunt

- Not just a build tool
- Tightly integrated with your framework for a more seamless workflow
- No watched resources in development
- Less development dependencies

CSS Goodies

- Recalculated relative urls
- Minification

LESS

```
compile 'com.bertramlabs.plugins:less-asset-pipeline:x.x.x'
```

- Standard vs. Less4j
- Use less4j
- Clean debugging output in your logs
- Use imports instead of directives

CoffeeScript

```
compile 'com.bertramlabs.plugins:coffee-asset-pipeline:x.x.x'
```

- Each file is isolated scoped
- Sourcemaps coming soon
- Different directive pattern `#= require blah`
- Uses nodejs coffee plugin if detected for faster compile

Handlebars

compile 'com.bertramlabs.plugins:handlebars-asset-pipeline:x.x.x'

- Bundle all templates in one js file (template cache)
- Configurable template prefix to control your template names by path
- Embedded handlebars library, or override by including your own

SASS Compass

```
compile 'com.bertramlabs.plugins:sass-asset-pipeline:x.x.x'
```

- Powerful and elegant
- Compass baked in (some restrictions)

SASS Customizations

This plugin uses `jruby-container` to isolate your Jruby gemsets and prevent corruption by your environment.

- Can customize a list of additional gem dependencies (such as `bourbon`)
- Always evaluates a `config.rb` file if detected in the same path as the SASS file being compiled

SASS Limitations

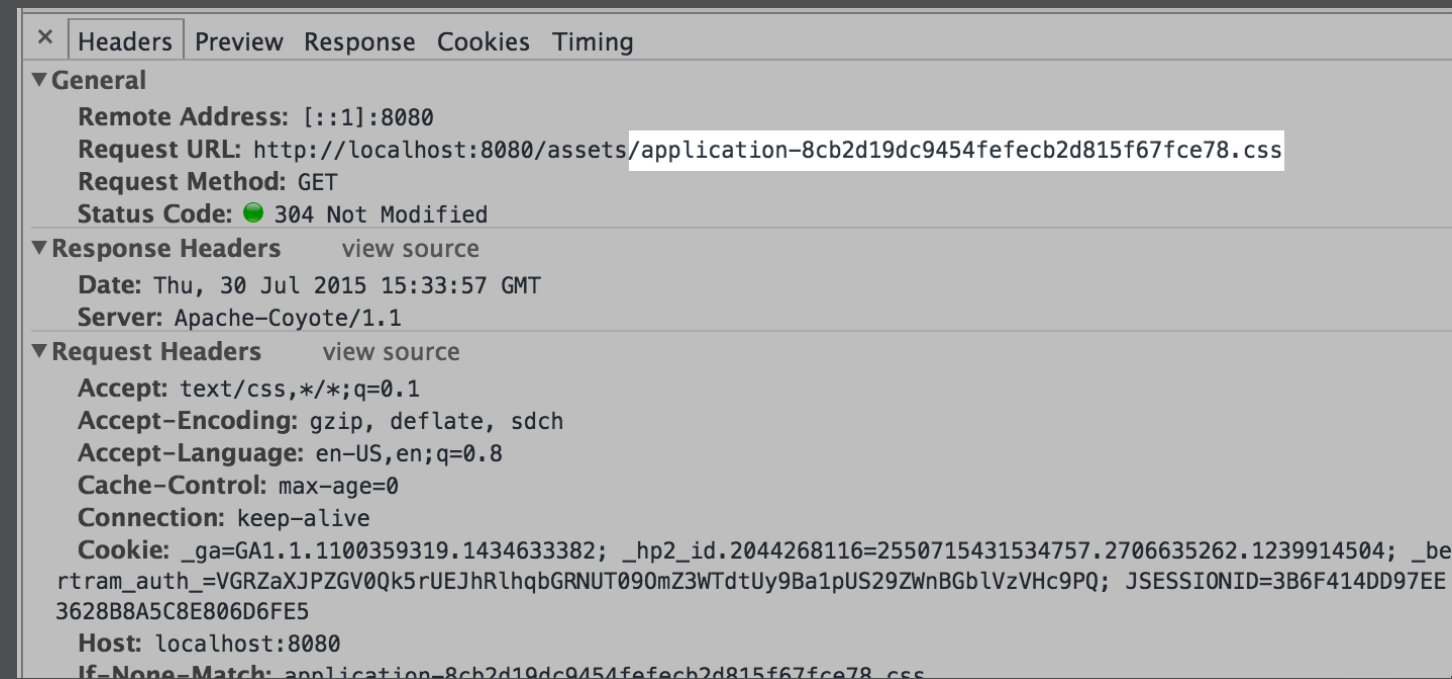
- Compass requires some monkey patching. SASS itself does not.
- Sprite Generation support still being refactored
- Truby is a hog atm

Production!

- Digests
- Etags
- CDNs
- External storagepath
- Custom request urls
- Minification
- More SPEED!

Digests

MD5 Suffix appended to the end of a filename. Used for cache busting consistently across browsers and proxies. i.e.



Packaging

- Assets are packaged in your container's assets folder along with a manifest.properties file.
- Manifest allows for both diff level compiling and proper etags when requesting by non digested name
- We don't force digest only request handling.

Excludes / Includes

- Partial files are automatically excluded `**/_*.*`
- Common patterns
 - Uses GLOB patterns i.e. `**/*.js` or `**/*.less`
- Common mistakes
 - Don't factor in your organizational assets subfolders
- Use Bower Installer

Minification

- Closure compiler uses Rhino's AST parser (not runtime)
- Much more efficient than uglifyjs
- Sourcemaps
- Angular JS hates to be small (but it can be done)

Other Plugins

- `cdn-asset-pipeline`
- `jsx-asset-pipeline`
- `angular-templates-asset-pipeline`
- `ember-asset-pipeline`
- `more...`

Additional Resources

- **Documentation:** <http://bertramdev.github.com/asset-pipeline>
- **Github:** <http://github.com/bertramdev/asset-pipeline-core>
- **Angular Asset-Pipeline:** @craigburke

Questions?