adaptTo()

APACHE SLING & FRIENDS TECH MEETUP
10-12 SEPTEMBER 2018

Integrating a Modern Frontend Setup with AEM
Natalia Venditto, Netcentric

A modern frontend setup

How do we understand a modern setup?

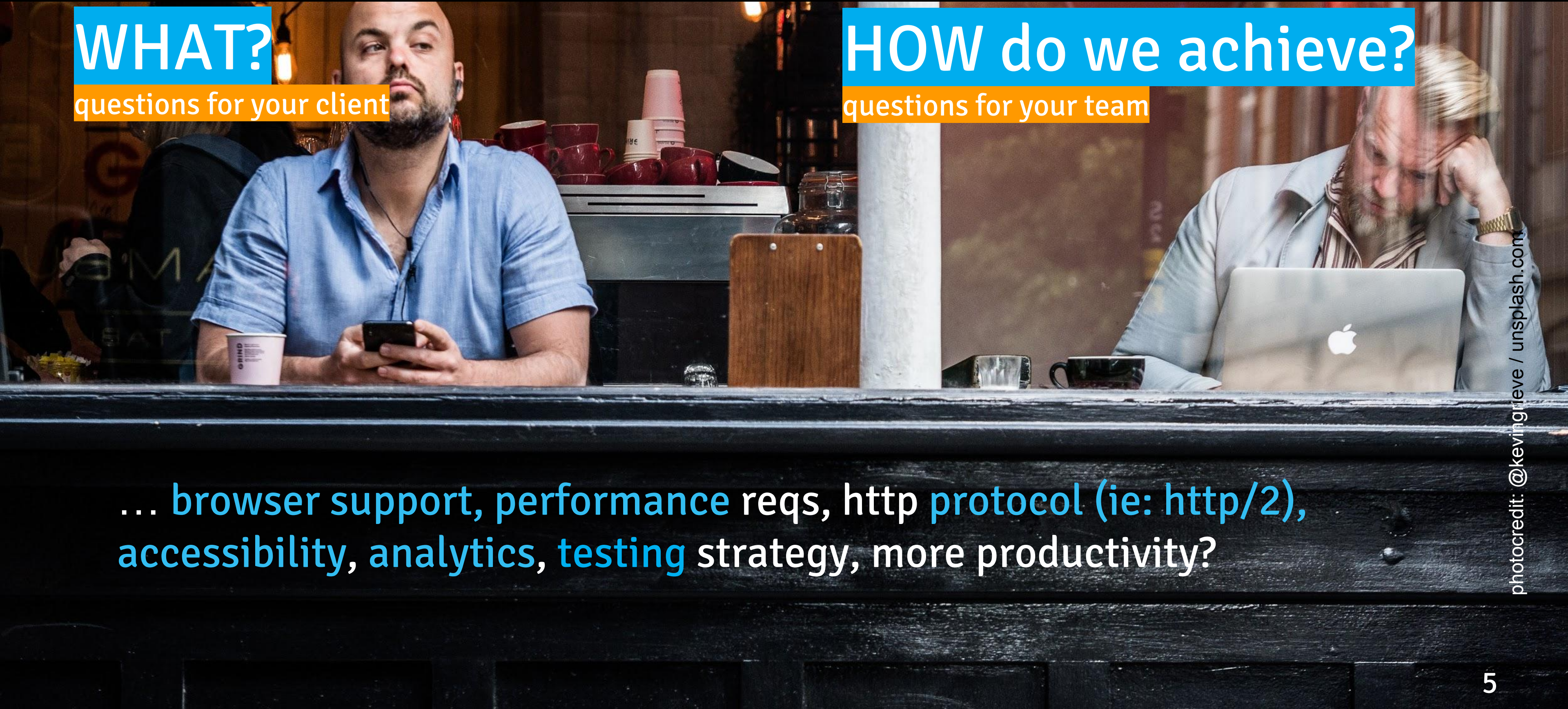
- A modern setup
    - ...does not necessarily include the latest tool in the market
    - …it's standard enough that it can be easily maintained
    - …does not force everyone in the team to a high knowledge ramp up investment
    - …it implements industry (current) best practices
    - ...requires a strategy

adaptTo()

Defining a frontend strategy!

photocredit: @rawpixel / unsplash.com

The `What?` and `How?` questions



WHAT?

questions for your client

HOW do we achieve?

questions for your team

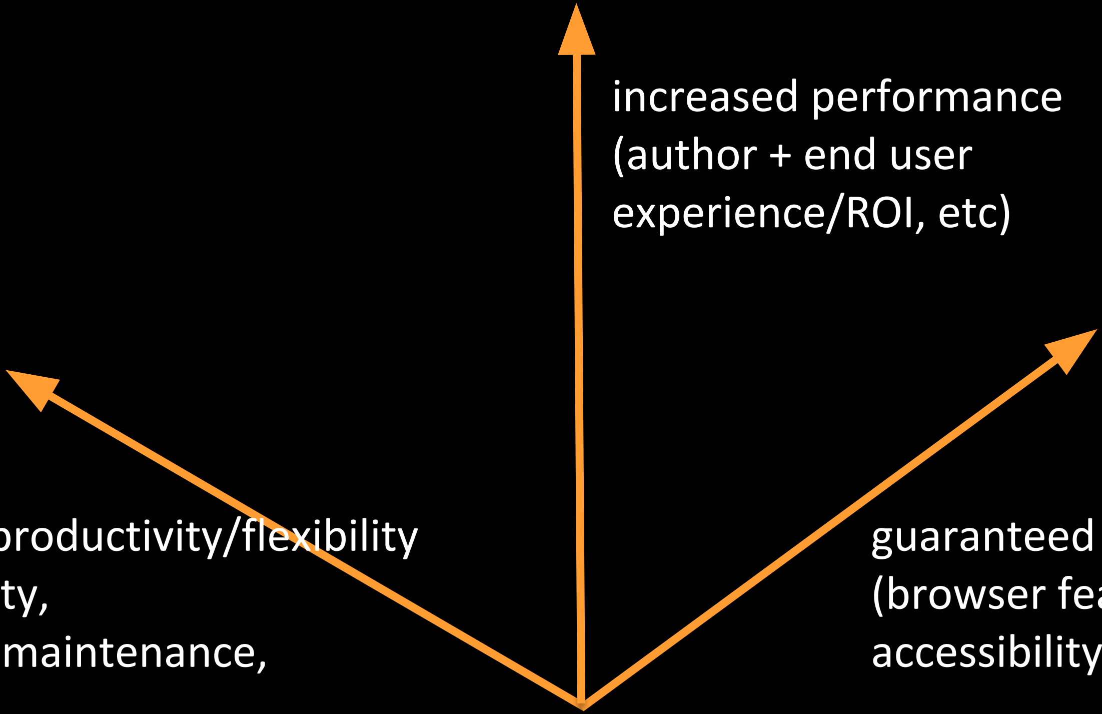
… browser support, performance reqs, http protocol (ie: http/2), accessibility, analytics, testing strategy, more productivity?

photocredit: @kevingrieve / unsplash.com

The `Which?` questions

Which JS	Which CSS	Which Build Tools/ Setup	Which AEM features?
ECMAScript Spec Frameworks State Manag. 3rd party/APIS	Which Preprocessor? (SASS, LESS, PostCSS)	Bundlers Task Runners Linters/Coverage Loaders/Plugins	What version of AEM? What new features? Clientlibs strategy

The win/win results matrix



increased performance
(author + end user
experience/ROI, etc)

enhanced productivity/flexibility
(code quality,
testability, maintenance,
scalability)

guaranteed innovation/creativity
(browser feature support,
accessibility)

adaptTo()

Study case: directory structure

adaptTo()

Directory structure

```
. ~
|-- [PROJECT-ID]-3rd-party/
|-- [PROJECT-ID]-complete-package/
|-- [PROJECT-ID]-components/[PROJECT-ID]-components-package/
    |-- src/main/jcr_root/apps/[PROJECT-ID]/
        |-- commons/
        |-- components/
            |-- component1/
            |-- component2/
            |-- // as many components
        |-- pages/
|-- [PROJECT-ID]-configuration/
|-- [PROJECT-ID]-demo-content/
.pom.xml
```

Directory structure

```
|-- components/component1/
        |-- clientlibs/
            |-- author/
            |-- publish/
                 |-- .content.xml
                 |-- component1.publish.entry.js
                 |-- component1.publish.entry.scss
                 |-- css.txt
                 |-- js.txt
        |-- .content.xml
        |-- component1.html
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jcr:root
xmlns:cq="http://www.day.com/jcr/cq/1.0"
xmlns:jcr="http://www.jcp.org/jcr/1.0"
jcr:primaryType="cq:ClientLibraryFolder"
categories="[PROJECT-ID.publish.component1
]"
jsProcessor="[min:gcc;obfuscate=true]"
cssProcessor="[min:gcc]"
/>
```

adaptTo()

Study case: tools and configuration

adaptTo()

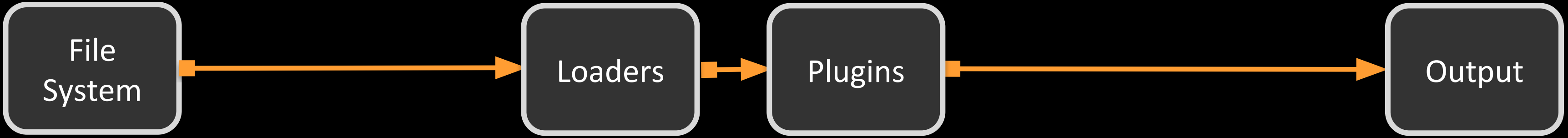
Tools and configuration

node JS
Node js Development

npm

webpack

Tools and configuration



- configure webpack to traverse our component folders, collect
  [name-of-component].entry.js
  [name-of-component].entry.scss
  files
  (webpack allows us to configure entry points per path or other params)
- configure webpack to read files according to loaders, and perform additional operations
  (transpilation, tree-shaking and DCE, etc)
- configure webpack to output files in designated targets, so that we are able to minify them,
  aggregate them and compress them (in AEM with gcc)

Setup Tools

```
nvm install --lts
cd [PROJECT-ID]-components/[PROJECT-ID]-components-package
mkdir frontend
cd frontend
npm init -y // generates package.json
npm install webpack --save-dev // maintain as dev
dependency

//successive installations
 npm install modulename --save ||
 --save-dev
```

Setup Tools

BABEL

Transpiling of ES (to JS ES5)

test: /\.css$/,

Transpiling of CSS pre processed code

Linting

Browser Support

Setup configuration

```
. ~
|-- [PROJECT-ID]-components/[PROJECT-ID]-components-package/
    |-- frontend/
        |-- .browserslistrc
        |-- .eslintignore
        |-- .eslintcache
        |-- .eslintrc
        |-- .stylelintcache
        |-- .stylintrc
        |-- .gitignore
        |-- .npmrc
        |-- package.json
.pom.xml
```

```
last 2 version
> 1%
maintained node versions
not dead
```

Setup Tools

BABEL

Transpiling of ES (to JS ES5)

test: /\.css$/,

Transpiling of CSS pre processed code

ESLint

Linting

Browser Support

Minification Aggregation of JS and CSS

Directory structure

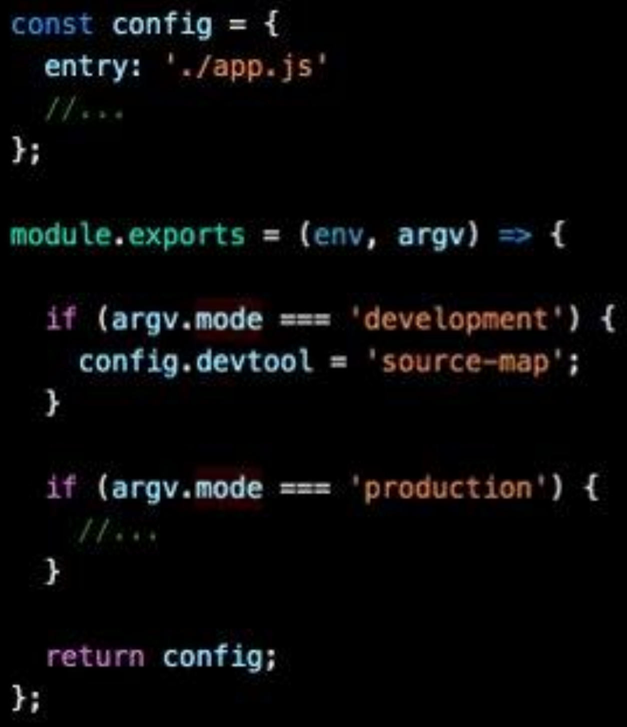
```
|-- components/component1/
        |-- clientlibs/
            |-- author/
            |-- publish/
                 |-- .content.xml
                 |-- component1.publish.entry.js
                 |-- component1.publish.entry.scss
                 |-- css.txt
                 |-- js.txt
       |-- .content.xml
       |-- component1.html
```

```
<?xml version="1.0" encoding="UTF-8"?>
<jcr:root
xmlns:cq="http://www.day.com/jcr/cq/1.0"
xmlns:jcr="http://www.jcp.org/jcr/1.0"
jcr:primaryType="cq:ClientLibraryFolder"
categories="[PROJECT-ID.publish.component1
]"
jsProcessor="[min:gcc;obfuscate=true]"
cssProcessor="min:gcc]"
/>
```

Setup configuration

```
. ~
|-- [PROJECT-ID]-3rd-party/
|-- [PROJECT-ID]-complete-package/
|-- [PROJECT-ID]-components/[PROJECT-ID]-components-package/
    |-- frontend/
        |-- build/config/
            |-- webpack.config.dev.js
            |-- webpack.config.prod.js
            |-- whatever other configs (ie: unit testing, e2e...)
        |-- package.json
|-- [PROJECT-ID]-configuration/
|-- [PROJECT-ID]-demo-content/
.pom.xml
```

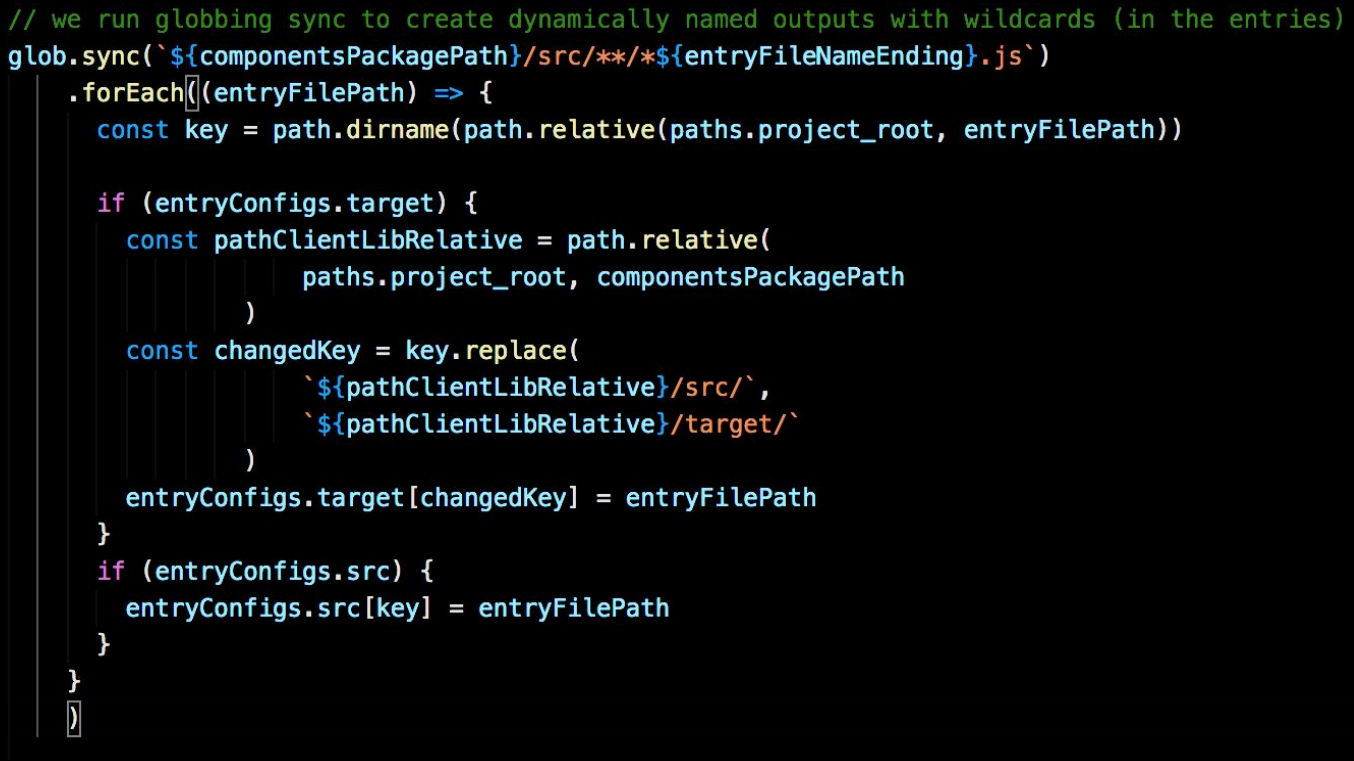
Setup configuration



```
const config = {
  entry: './app.js'
  //...
};

module.exports = (env, argv) => {

  if (argv.mode === 'development') {
    config.devtool = 'source-map';
  }

  if (argv.mode === 'production') {
    //...
  }

  return config;
};
```

- for our frontend code build, we will favor webpack 4 optimized "mode" configuration, over OSGI environment definition, in order to leverage Webpacks caching and performance

Setup configuration



```
// we run globbing sync to create dynamically named outputs with wildcards (in the entries)
glob.sync(`${componentsPackagePath}/src/**/*${entryFileNameEnding}.js`)
  .forEach((entryFilePath) => {
   const key = path.dirname(path.relative(paths.project_root, entryFilePath))

   if (entryConfigs.target) {
    const pathClientLibRelative = path.relative(
          paths.project_root, componentsPackagePath
        )
    const changedKey = key.replace(
          `${pathClientLibRelative}/src/`,
          `${pathClientLibRelative}/target/`
        )
    entryConfigs.target[changedKey] = entryFilePath
   }
   if (entryConfigs.src) {
    entryConfigs.src[key] = entryFilePath
   }
  }
  )
```

adaptTo()

Setup configuration

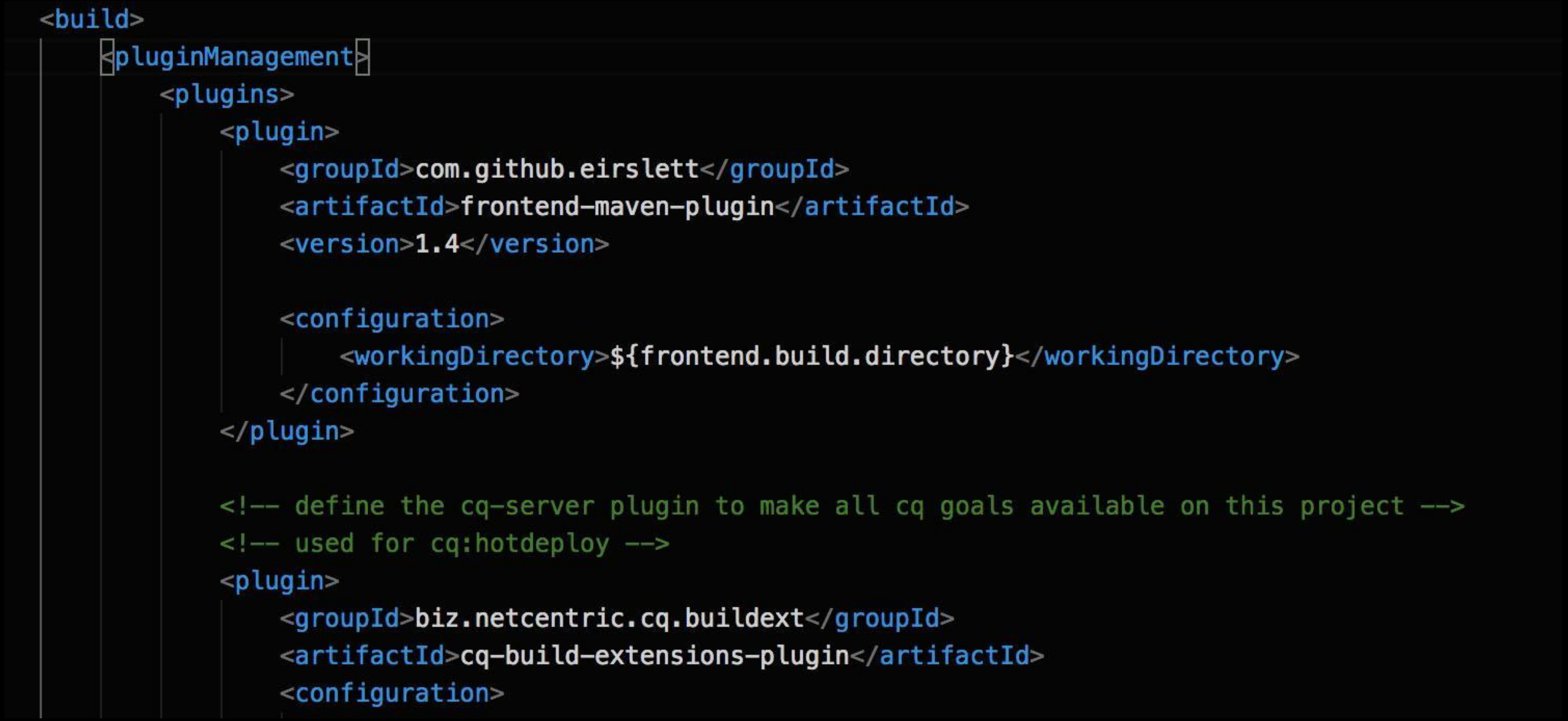
```
module.exports = {
  //...
  optimization: {
    splitChunks: {
      chunks: 'async',
      minSize: 30000,
      maxSize: 0,
      minChunks: 1,
      maxAsyncRequests: 5,
      maxInitialRequests: 3,
      automaticNameDelimiter: '~',
      name: true,
      cacheGroups: {
        vendors: {
          test: /[\\/]node_modules[\\/]/,
          priority: -10
        },
        default: {
          minChunks: 2,
          priority: -20,
          reuseExistingChunk: true
        }
      }
    }
  }
};
```

may need to update this (default) config for SplitChunkPlugin

Setup configuration

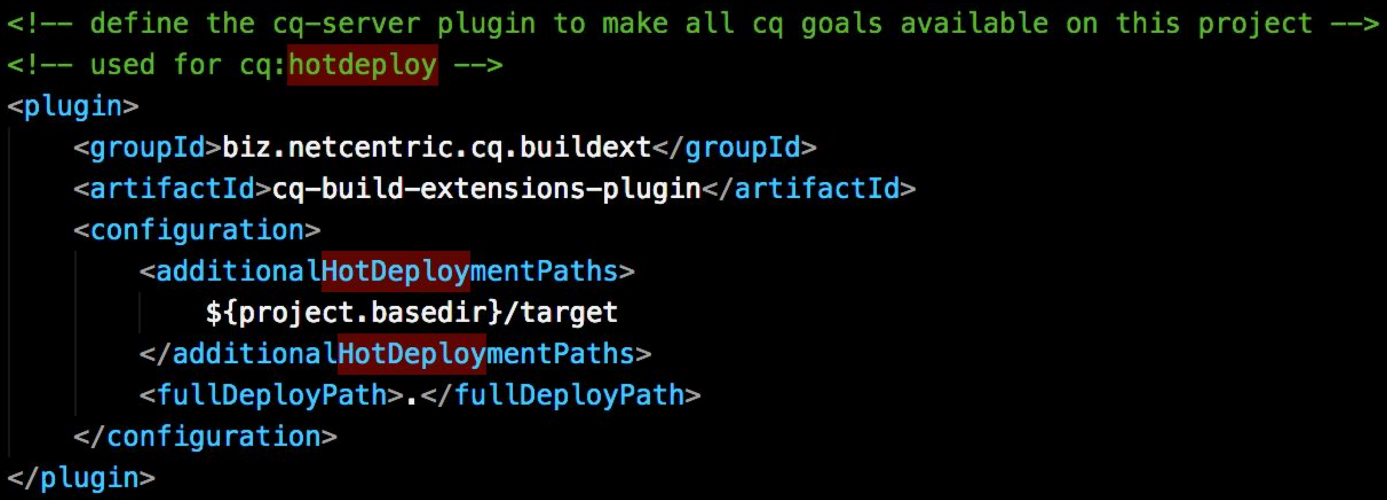
- edit the corresponding POM.xml, to make use of the frontend plugin for maven* -> https://mvnrepository.com/artifact/com.github.eirslett/frontend-maven-plugin?repo=redhat-ga

Maven frontend plugin



```
<build>
    <pluginManagement>
        <plugins>
            <plugin>
                <groupId>com.github.eirslett</groupId>
                <artifactId>frontend-maven-plugin</artifactId>
                <version>1.4</version>

                <configuration>
                    <workingDirectory>${frontend.build.directory}</workingDirectory>
                </configuration>
            </plugin>

            <!-- define the cq-server plugin to make all cq goals available on this project -->
            <!-- used for cq:hotdeploy -->
            <plugin>
                <groupId>biz.netcentric.cq.buildext</groupId>
                <artifactId>cq-build-extensions-plugin</artifactId>
                <configuration>
```

Syncing with AEM



```
<!-- define the cq-server plugin to make all cq goals available on this project -->
<!-- used for cq:hotdeploy -->
<plugin>
    <groupId>biz.netcentric.cq.buildext</groupId>
    <artifactId>cq-build-extensions-plugin</artifactId>
    <configuration>
        <additionalHotDeploymentPaths>
            ${project.basedir}/target
        </additionalHotDeploymentPaths>
        <fullDeployPath>.</fullDeployPath>
    </configuration>
</plugin>
```

webpack

```
webpack --watch
```

Setup configuration

```
|-- components/component1/
        |-- clientlibs/
            |-- author/
                 |-- .content.xml
                 |-- component1.author.entry.js
                 |-- component1.author.scss
                 |-- css.txt
                 |-- js.txt
                 |-- component1_dialog.js
            |-- publish/
        |-- _cq_dialog/
        |-- .content.xml
        |-- component1.html
```

adaptTo()

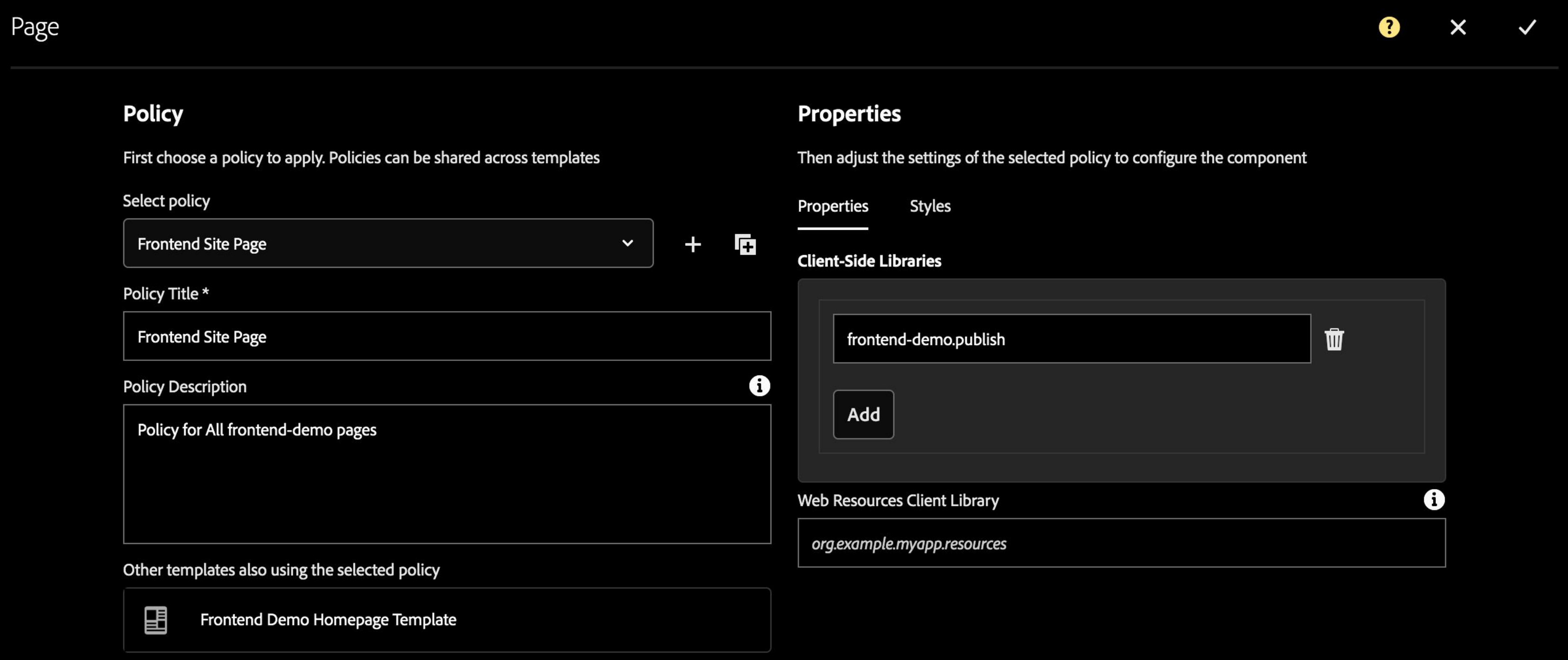
Setup configuration

```
. ~
|-- [PROJECT-ID]-components/[PROJECT-ID]-components-package/
    |-- frontend/
        |-- .browserslistrc
        |-- .eslintignore
        |-- .eslintcache
        |-- .eslintrc
        |-- .stylelintcache
        |-- .stylintrc
        |-- .gitignore
        |-- .npmrc
        |-- package.json
.pom.xml
```

```
*.bundle.js
node_modules
*_dialog.js
build/shared/variables/*.js
node/*
build/config
gulpfile.js
```

Study case: shipping the code (to the browser)

Loading styles via Content Policies



Page

Policy

First choose a policy to apply. Policies can be shared across templates

Select policy

Frontend Site Page

Policy Title *

Frontend Site Page

Policy Description

Policy for All frontend-demo pages

Other templates also using the selected policy

Frontend Demo Homepage Template

Properties

Then adjust the settings of the selected policy to configure the component

Properties | Styles

Client-Side Libraries

frontend-demo.publish

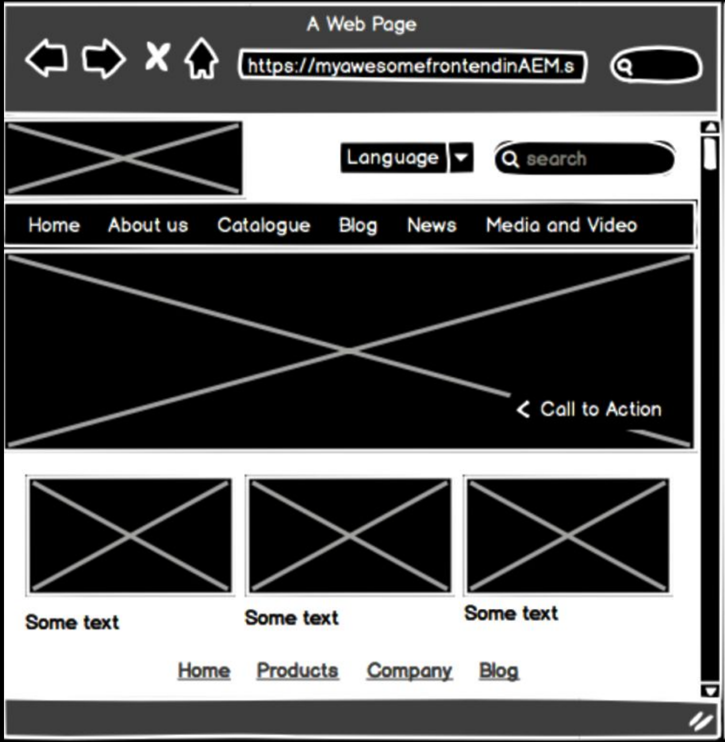
Add

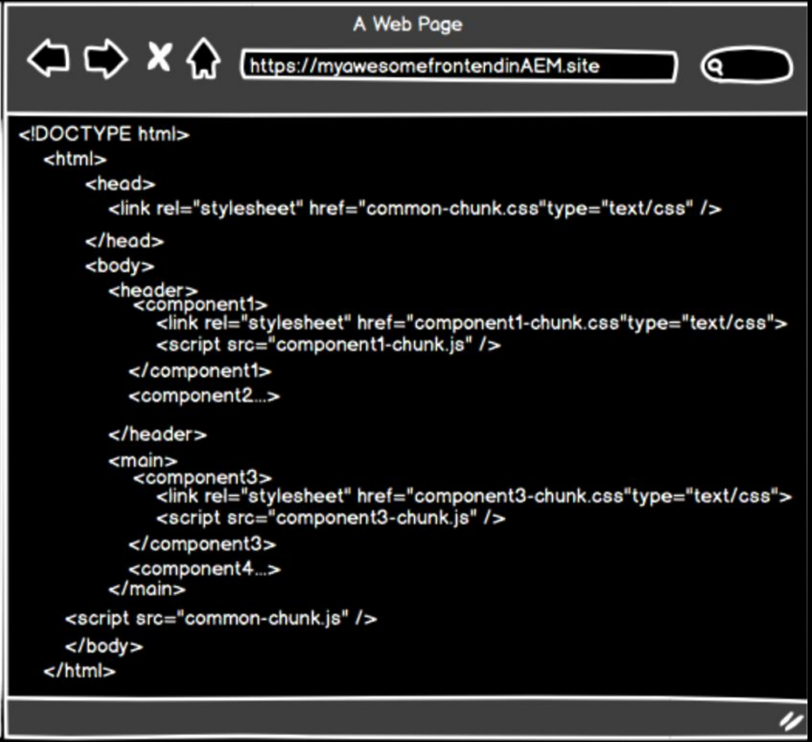
Web Resources Client Library

org.example.myapp.resources

adaptTo()

Loading techniques





```
<!DOCTYPE html>
  <html>
      <head>
        <link rel="stylesheet" href="common-chunk.css"type="text/css" />
      </head>
      <body>
        <header>
          <component1>
            <link rel="stylesheet" href="component1-chunk.css"type="text/css">
            <script src="component1-chunk.js" />
          </component1>
          <component2...>
        </header>
        <main>
          <component3>
            <link rel="stylesheet" href="component3-chunk.css"type="text/css">
            <script src="component3-chunk.js" />
          </component3>
          <component4...>
        </main>
    <script src="common-chunk.js" />
    </body>
  </html>
```

Loading: Templating (Sightly, Handlebars...)

```
|-- components/component1/
        |-- clientlibs/
            |-- author/
            |-- publish/
            |-- additional-styles
                |-- .content.xml
                |-- component1.add-sty
                |-- component1.add-sty
                |-- css.txt
                |-- js.txt
        |-- .content.xml
        |-- component1.html
```

```
<!--/* component1.author */-->
<sly data-sly-test="${wcmmode.edit}"
data-sly-use.clientLib="${'/libs/granite/sightly/tem
plates/clientlib.html'}"
data-sly-call="${clientLib.all @
categories='[PROJECT-ID].component1.author'}" />


<!--/* component1.publish */-->
<sly
data-sly-use.clientLib="${'/libs/granite/sightly/tem
plates/clientlib.html'}"
data-sly-call="${clientLib.all @
categories='[PROJECT-ID].component1.publish'}" />
```

adaptTo()

Script loading attributes

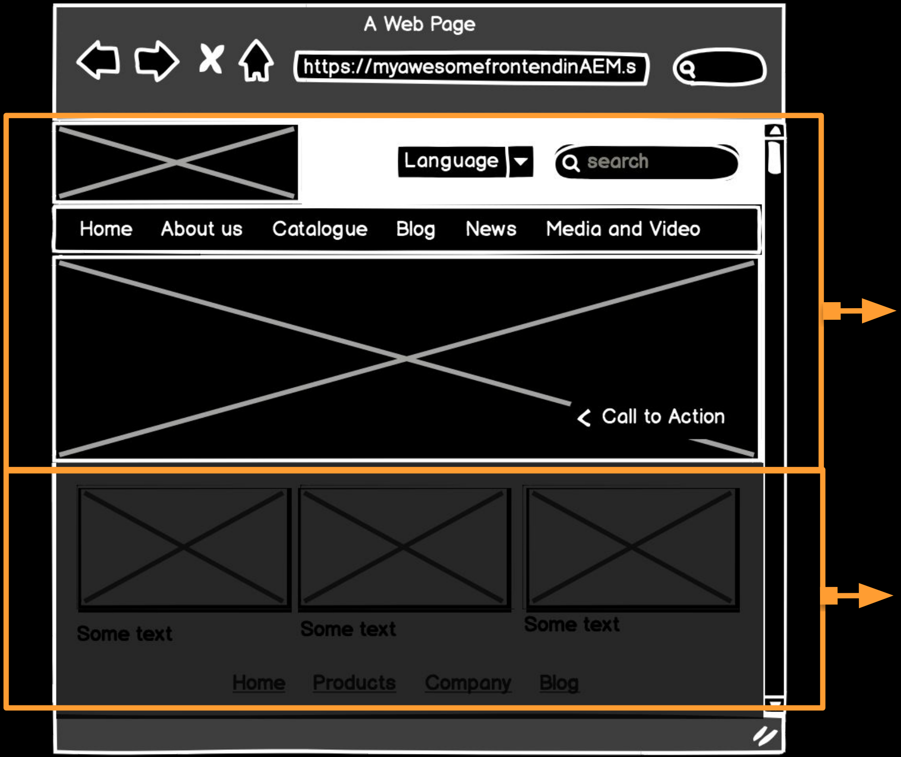
```
<script src="myproject.component1.js" defer></script>
<script src="myproject.component1.js" async></script>

<script src="myproject.component1.js" preload></script>
<script ...
```

adaptTo()

Lazy Loading

adaptTo()

Inline load + Lazy Loading strategy



hero area/above the fold

lazy loaded chunks
(and resources, such as images)

Intersection Observers API

```
// test for browser support
if (!'IntersectionObserver' in window &&
    !'IntersectionObserverEntry' in window &&
    !'intersectionRatio' in
window.IntersectionObserverEntry.prototype) {
    // load polyfill now
}
```

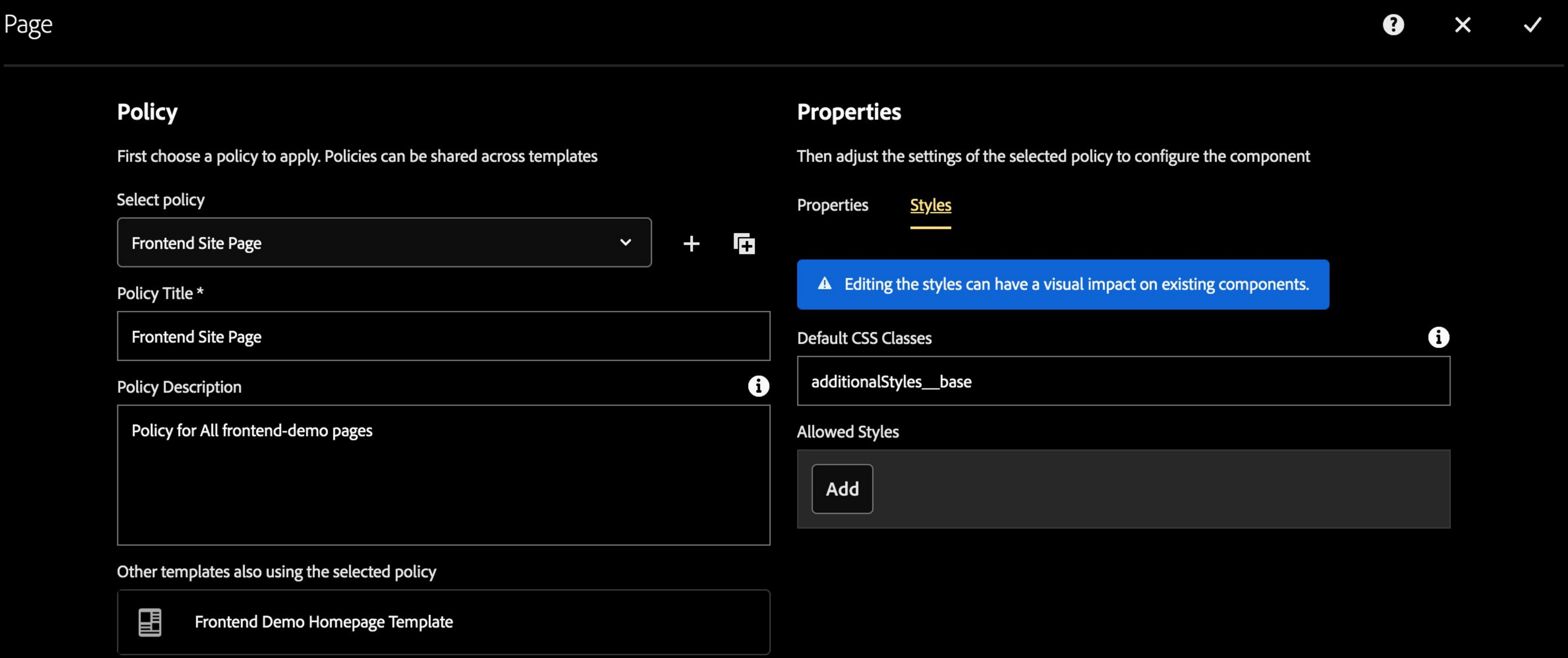
LazyLoading and SEO

- SSR (server side rendering techniques)
- Content Fragments
- Experience Fragments

adaptTo()

Demo Time (added backup)

Loading styles via Content Policies



Page

Policy

First choose a policy to apply. Policies can be shared across templates

Select policy

Frontend Site Page

Policy Title *

Frontend Site Page

Policy Description

Policy for All frontend-demo pages

Other templates also using the selected policy

Frontend Demo Homepage Template

Properties

Then adjust the settings of the selected policy to configure the component

Properties | Styles

Editing the styles can have a visual impact on existing components.

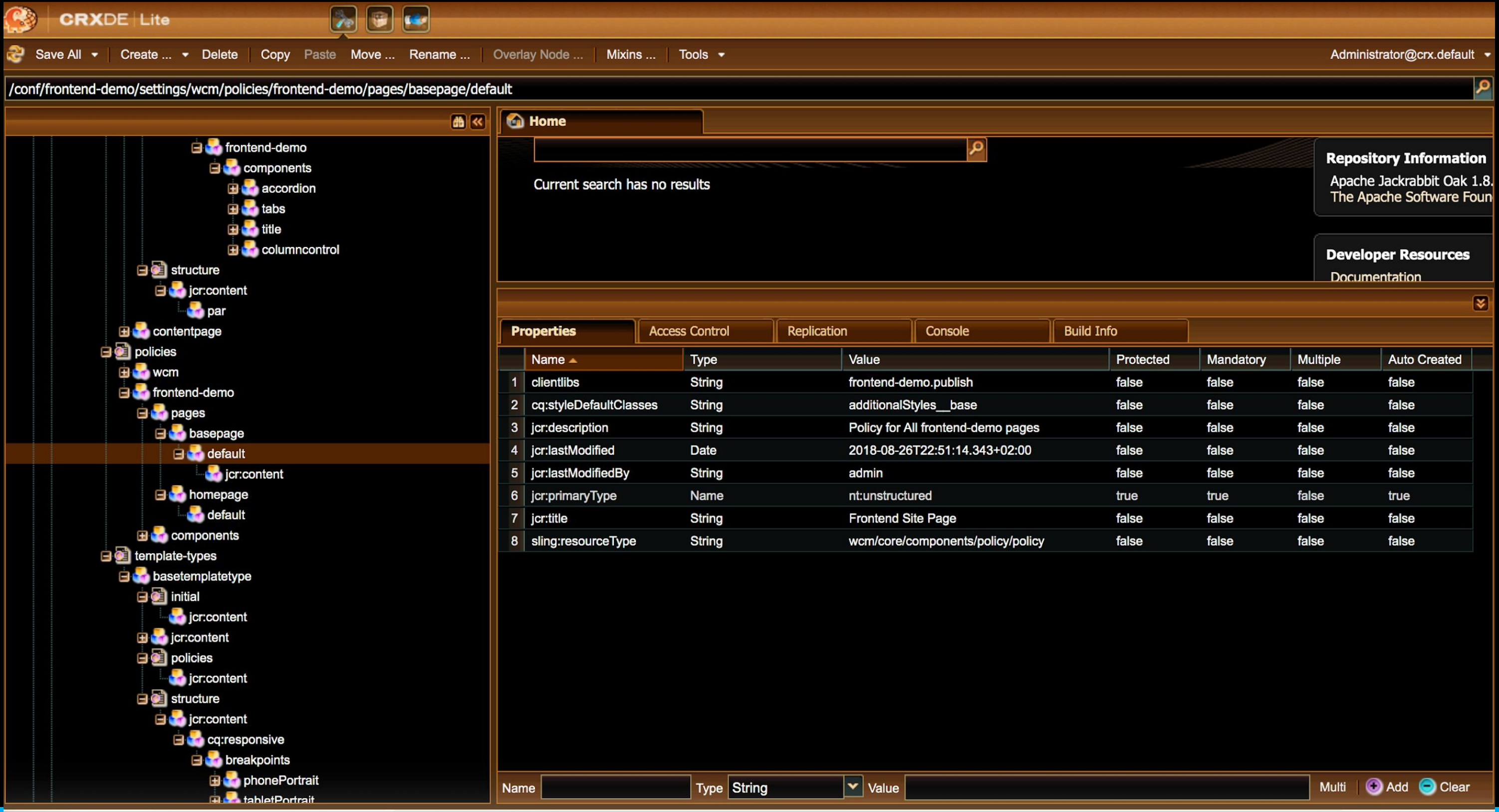
Default CSS Classes

additionalStyles__base

Allowed Styles

Add

Loading styles via Content Policies



/conf/frontend-demo/settings/wcm/policies/frontend-demo/pages/basepage/default

```
frontend-demo
  components
    accordion
    tabs
    title
    columncontrol
structure
  jcr:content
    par
contentpage
policies
  wcm
  frontend-demo
    pages
      basepage
        default
          jcr:content
      homepage
        default
    components
template-types
  basetemplatetype
    initial
      jcr:content
    jcr:content
    policies
      jcr:content
    structure
      jcr:content
        cq:responsive
          breakpoints
            phonePortrait
            tabletPortrait
```

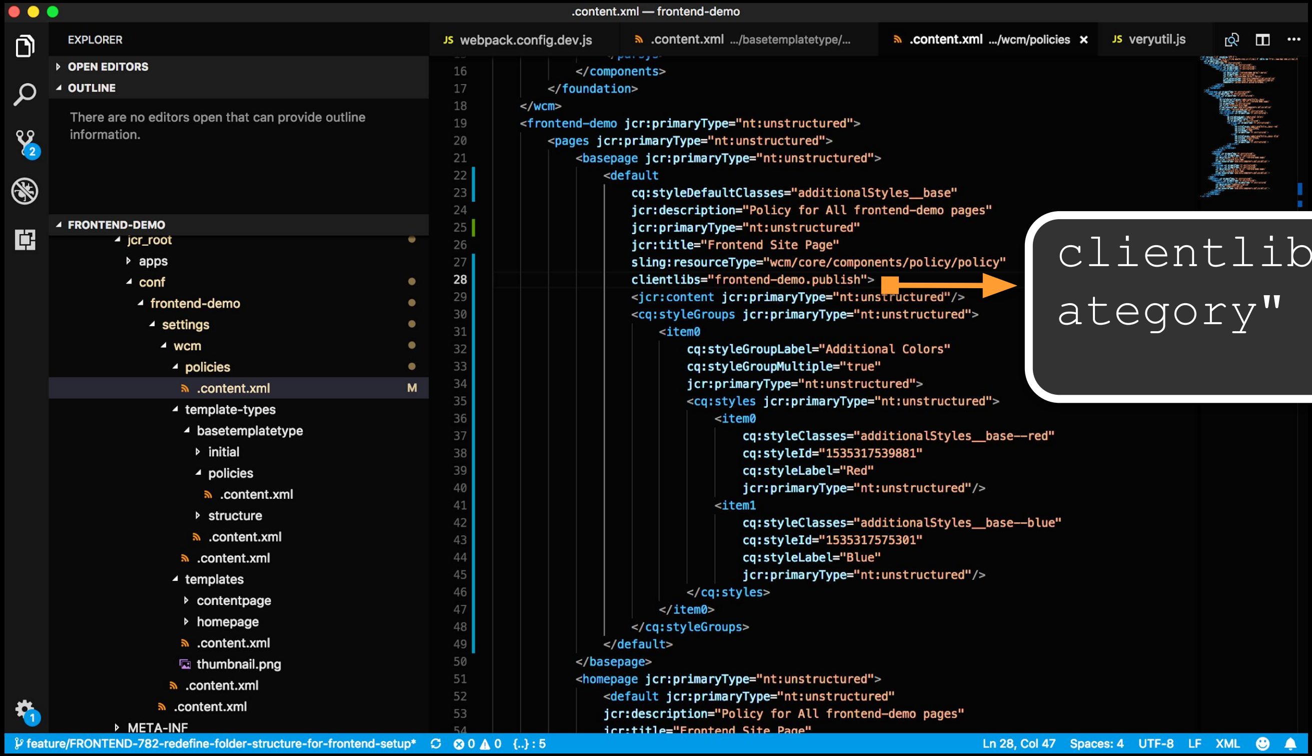
Current search has no results

Repository Information
Apache Jackrabbit Oak 1.8.
The Apache Software Foun

Developer Resources
Documentation

	Name	Type	Value	Protected	Mandatory	Multiple	Auto Created
1	clientlibs	String	frontend-demo.publish	false	false	false	false
2	cq:styleDefaultClasses	String	additionalStyles__base	false	false	false	false
3	jcr:description	String	Policy for All frontend-demo pages	false	false	false	false
4	jcr:lastModified	Date	2018-08-26T22:51:14.343+02:00	false	false	false	false
5	jcr:lastModifiedBy	String	admin	false	false	false	false
6	jcr:primaryType	Name	nt:unstructured	true	true	false	true
7	jcr:title	String	Frontend Site Page	false	false	false	false
8	sling:resourceType	String	wcm/core/components/policy/policy	false	false	false	false

Loading styles via Content Policies



.content.xml — frontend-demo

```
            </components>
        </foundation>
    </wcm>
    <frontend-demo jcr:primaryType="nt:unstructured">
        <pages jcr:primaryType="nt:unstructured">
            <basepage jcr:primaryType="nt:unstructured">
                <default
                    cq:styleDefaultClasses="additionalStyles__base"
                    jcr:description="Policy for All frontend-demo pages"
                    jcr:primaryType="nt:unstructured"
                    jcr:title="Frontend Site Page"
                    sling:resourceType="wcm/core/components/policy/policy"
                    clientlibs="frontend-demo.publish">
                    <jcr:content jcr:primaryType="nt:unstructured"/>
                    <cq:styleGroups jcr:primaryType="nt:unstructured">
                        <item0
                            cq:styleGroupLabel="Additional Colors"
                            cq:styleGroupMultiple="true"
                            jcr:primaryType="nt:unstructured">
                            <cq:styles jcr:primaryType="nt:unstructured">
                                <item0
                                    cq:styleClasses="additionalStyles__base--red"
                                    cq:styleId="1535317539881"
                                    cq:styleLabel="Red"
                                    jcr:primaryType="nt:unstructured"/>
                                <item1
                                    cq:styleClasses="additionalStyles__base--blue"
                                    cq:styleId="1535317575301"
                                    cq:styleLabel="Blue"
                                    jcr:primaryType="nt:unstructured"/>
                            </cq:styles>
                        </item0>
                    </cq:styleGroups>
                </default>
            </basepage>
            <homepage jcr:primaryType="nt:unstructured">
                <default jcr:primaryType="nt:unstructured"
                jcr:description="Policy for All frontend-demo pages"
```

```
feature/FRONTEND-782-redefine-folder-structure-for-frontend-setup*   ⊗ 0 ⚠ 0   {..}: 5        Ln 28, Col 47   Spaces: 4   UTF-8   LF   XML
```

```
clientlibs="some.category"
```

adaptTo()

Thank you!