Webpack’d Plone

asko.soukka@iki.fi

Example: Customer orders a themed site

- Customer wants attractive site
- Customer orders attractive theme from ad agency

- Customer gets an attractive theme
- Developer gets a list of unexpected feature requirements

- It might be easier to write disposable features bound with theme than a generic configurable components

Example: Customer really wants a carousel

Implement carousel as a reusable package

Configure carousel in policy package

Theme carousel in theme package

Package and buildout

npm install slick-carousel

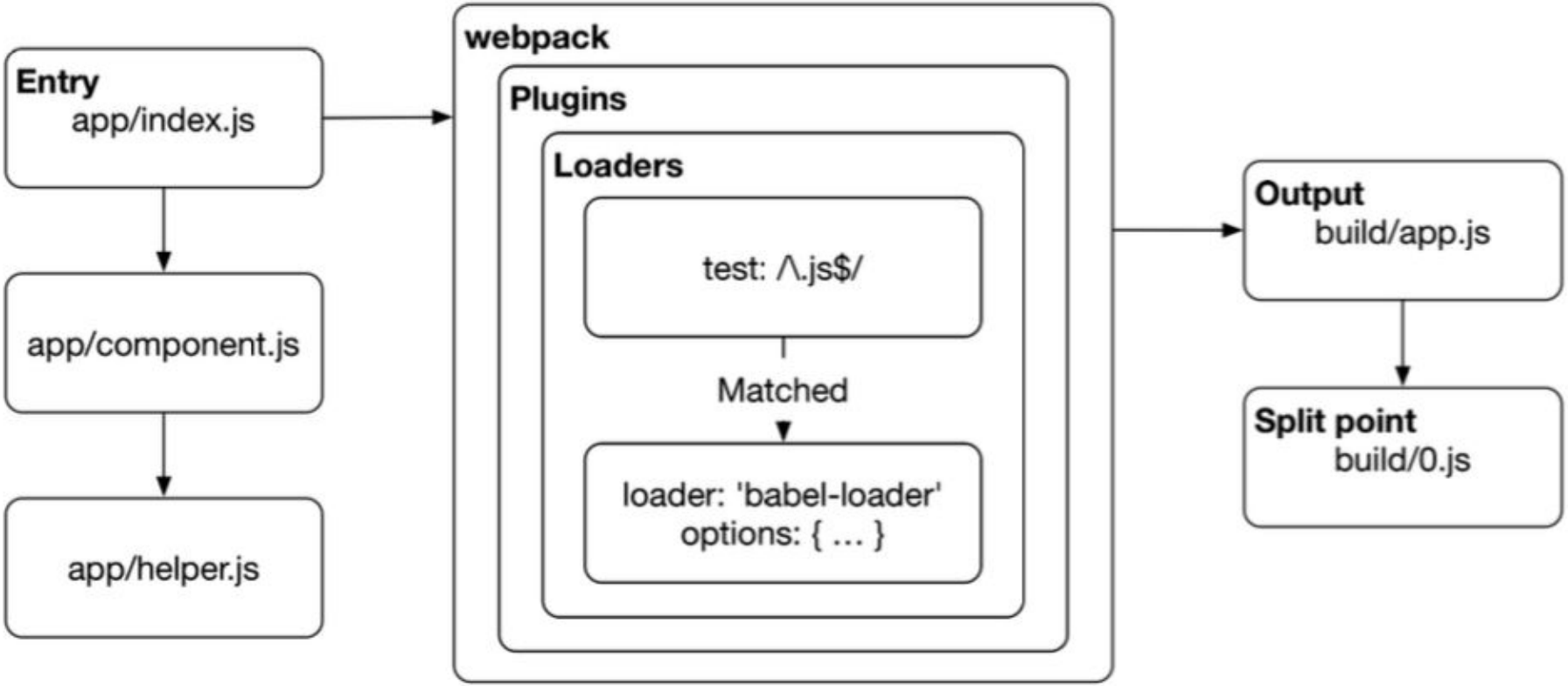
Add ZPT template fragment to generate HTML

Add JavaScript to init the carousel

Upload as a zip-file

Webpack is a bundler


BUNDLE
ALL THE THINGS
memegenerator.net



Webpack's execution process

survivejs.com

Bundling Plone Themes

Goals

- bundle all Plone front-end resources with theme

Wins:

- everything can be customized / overridden
- new JS libraries can be integrated
- new version can be deployed in a minute
- static resources can be served outside Plone
- more fun: no need to apologize, but just do it

Issues:

- reusable packaging is hard

1st gen : plonetheme.webpack

Steps:

- npm install all dependencies (from npmjs or github)
- map Plone require.js paths to filesystem using aliases
- shim with loaders and plugins like it’s 2014 again

Issues:

- huge webpack configuration
- hard to track dependency versions

2nd gen : plonetheme-webpack-plugin 0.x
plonetheme.webpacktemplate 1.x

Steps:

- webpack plugin to resolve Plone require.js paths to **virtual file system**
- default webpack configurations to use with webpack-merge

Issues:

- opaque webpack default configurations
- magic plugin and invisible virtual file system files
- edge cases with webpack resolving and virtual file system
- build requires running Plone site with all required resources

3rd gen : plonetheme-webpack-plugin 1.x plonetheme.webpacktemplate master

Steps:

- webpack plugin to resolve Plone require.js paths to **`./plone`**
- default webpack configurations to use with webpack-merge

Issues:

- opaque webpack default configurations
- first build requires running Plone site with all required resources

Happy path

```
git clone git@github.com:collective/plonetheme.webpacktemplate

mrbob plonetheme.webpacktemplate

cd [project]

make watch

# or when errors:

make watch_plone
make watch_theme

# open http://localhost:8080/Plone and enable your theme
```

customize

override

extend

Opinionated theme contents

- barceloneta.less
- default.js
- default.less
- index.html
- logged-in.js
- logged-in.less
- manifest.cfg
- preview.png
- rules.xml
- theme.less
- webpack.xml

Overriding LESS files (`less-variables.js`)

```
const PLONE = new PlonePlugin({
  ...
  variables: {
    'mockup-patterns-filemanager': '\'new/path/pattern.less\''
  }
  ...
});
```

Overriding JS modules (`config.js`)

```
const common = {
  ...
  resolve: {
    alias: {
      'mockup-patterns-filemanager': 'new/path/pattern.js',
      'mockup-patterns-filemanager-url': 'new/path'
    }
  }
  ...
};
```

collective.themesitesetup

collective.themefragments

plonetheme-upload

Examples

- https://github.com/collective/plonetheme.webpacktemplate
- https://github.com/plone/plone.app.theming/tree/datakurre-webpack
- https://github.com/jyukopla/reclas
- https://github.com/datakurre/plonetheme.plonereact

Questions?