React I

Martin Ledvinka

martin.ledvinka@fel.cvut.cz

Winter Term 2016

Contents

Deployment and build

Deployment

How is the web content served by our application?

- The configuration is in `WebAppConfig`,
- Resource handlers map URLs to locations of static content (JS, CSS),
    - More efficient access,
- We enable default servlet configuration,
- Add `MultipartResolver` to support file upload,
- Configure Jackson message converter for serialization/deserialization of request data,
- `index.html` is in the root of war and is served by default by the application server.

Tools Used in React Apps

- NodeJS,
  - $\approx$ JVM,
  - JavaScript application server, runtime for executing JavaScript applications,
- npm,
  - $\approx$ Maven,
  - Used for dependency management in JS,
  - And for execution of build tools.

Build Process

The following applies to both ear-rt and ear-setup.
Section *scripts* in `package.json`.

1. `npm install` – downloads all JS dependencies declared in `package.json`,
2. `npm run build` – builds production version of the JS UI,
3. `npm start` – starts file watcher which rebuilds development version of the JS UI when changes in source code are detected. Used during development and debugging.

Build Tools

- Section *scripts* in `package.json`.

Browserify

- Recursive analysis of module imports in JS code,
- Builds a bundle of all the discovered modules,
- Bundle is then served in HTML page in a single `script` tag.

Babel

- JS compiler, enables use of the latest JS syntax in older-browser compatible way,
- Used as transformation step in *Browserify* (*babelify*).

Build Tools II

Watchify

- Watches for changes in source files and reruns *Browserify* when a change is detected.

Uglify

- JS code minification,
- Reduces resulting bundle size.

clean-css

- CSS optimization and minification,
- Reduces CSS file size.

CommonJS Modules

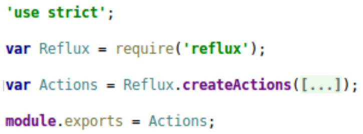
CommonJS Modules

CommonJS

- Specification of modules in JS,
- No module specification built in JS until ES6,
- Implemented by NodeJS - function `require`,
- And object `module.exports`,
- Works for:
    - External libraries,
    - Local files.

Module example

Module



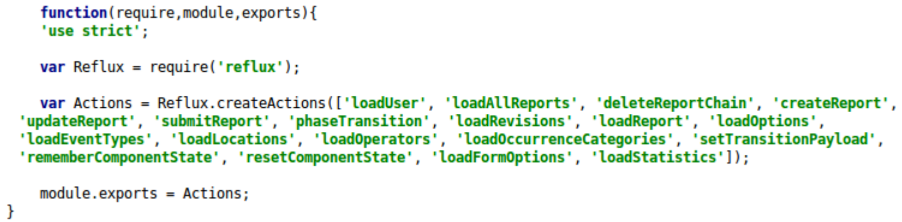
```
'use strict';

var Reflux = require('reflux');

var Actions = Reflux.createActions([...]);

module.exports = Actions;
```

is translated into



```
    function(require,module,exports){
    'use strict';

    var Reflux = require('reflux');

    var Actions = Reflux.createActions(['loadUser', 'loadAllReports', 'deleteReportChain', 'createReport',
  'updateReport', 'submitReport', 'phaseTransition', 'loadRevisions', 'loadReport', 'loadOptions',
  'loadEventTypes', 'loadLocations', 'loadOperators', 'loadOccurrenceCategories', 'setTransitionPayload',
  'rememberComponentState', 'resetComponentState', 'loadFormOptions', 'loadStatistics']);

    module.exports = Actions;
}
```

ES6 Modules

- EcmaScript 6 comes with its own definition of modules,
- `import`, `export` keywords,
- Not compatible with CommonJS,
- Still not supported by browsers and NodeJS,
- Babel translates into CommonJS `require` and `module.exports`.

For Interested

Compare the source codes of ear-rt and the compiled `bundle.js` *to see the difference.*

Tasks

Tasks

Working with ear-rt, the latest version from git.

1. Create a user profile view, where the user can view and edits his/her information,
    - First name,
    - Last name,
    - Username (read-only),
    - Password (with confirmation field),
2. Make sure the user data are validated,
    - All fields are non-empty,
    - Password matches its confirm,
3. Make the user profile view accessible from the top-right menu.

Resources

- https://blog.risingstack.com/node-js-at-scale-module-system-commonjs-require/,
- https://hackernoon.com/node-js-tc-39-and-modules-a1118aecf95e.