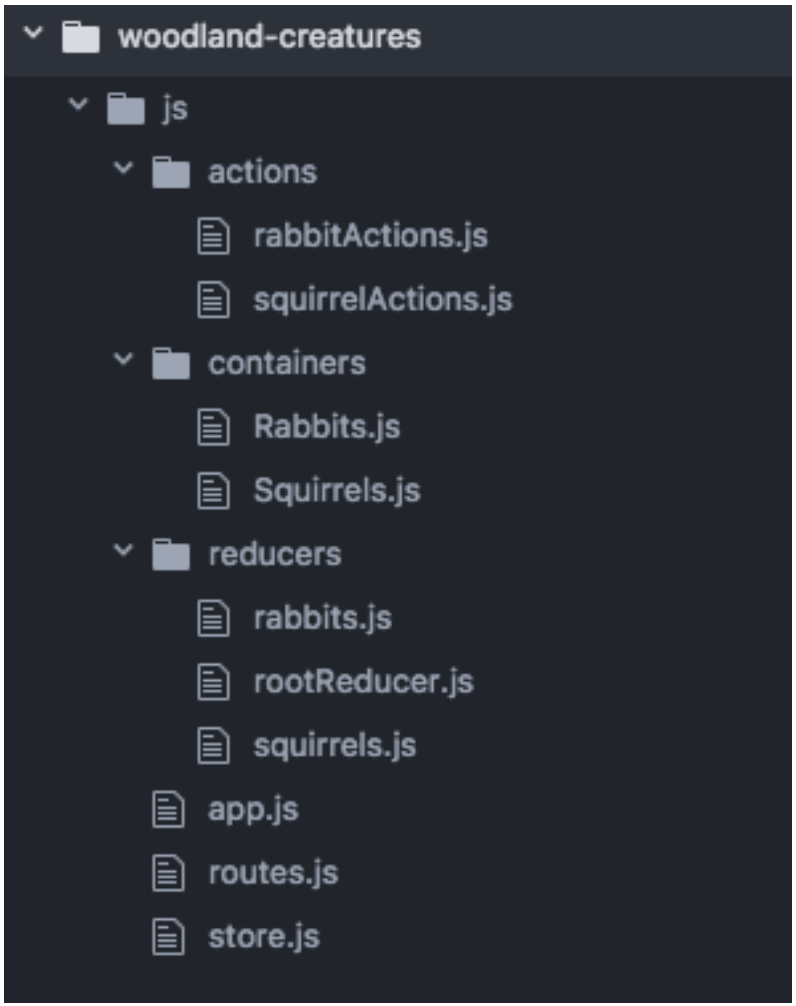
Developing Large Scale Apps

File structure

- File-Type-First
- Feature First
- Shared files

File-Type-First



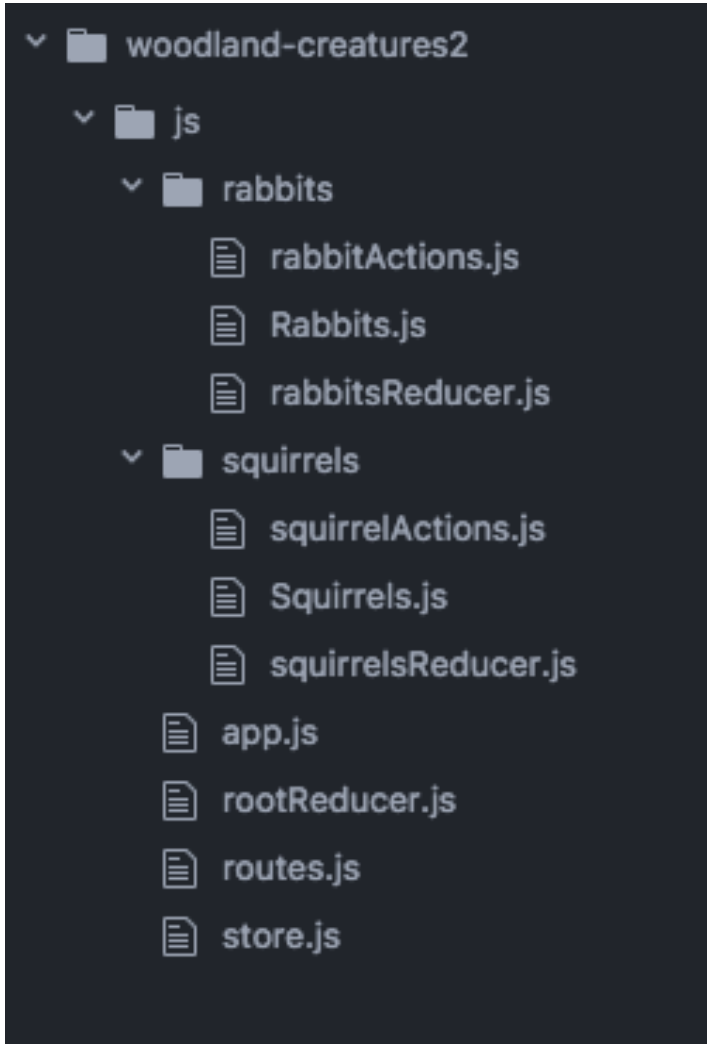
Pros

- Easy to get started; an empty project could have structure
- Easy to locate a file by its function
- Files imported into a single file can be in the same folder e.g. combining reducers

Cons

- Code splitting is difficult
- Editing a feature means I have to search for files across many folders
- Ejecting a component is cumbersome

Feature First



Pros

- Entry file(s) are obvious at feature/app level
- Easy to locate all files related to a feature
- Code splitting becomes easier
- Moving components into groups or out of the app is easier
- We can put may other related files: CSS modules, unit tests, etc.

Cons

- Combining reducers is more cumbersome
- Where do we put base/app level files i.e. where would we put actions, reducers, constants that don't fall under features

Shared Files

Code Splitting

- Split per route
  - Each distinct route should have a separate entry container component; not a single container used in multiple routes
- Separate module bundles
  - Initial load - login/registration (security)
  - Main app
  - Modules less used by general user base like an admin console
- Separate vendor bundles
  - Cached in browsers for longer durations

Other Topics

- All navigation between pages must be route driven
- Redux-Saga
  - Simplify async code
  - Readable, testable async flows
  - avoid callback hell
  - helps maintain pure reducers i.e. only return an object not a Promise or an object
- Accessibility
- Linting

The End