Demo notes

- Checkout branch: “lecture8-router” & goto Lecture8 directory
- Checkout revision 88ea66c Initial commit
- Futher mentioned „checkout: “ are followed with brief variant of a commit named with „Router – „ prefix
- Headlines below represent name of related slide in presentation
- Works with PascalCase (e.g. About) mark a component within solution

Client-side Routing (before HTML5)

- checkout: manual fragment routing example
- Have a look at App, Menu, Content components
- See Loader (so it can be seen when page really reloads)
- Upper navigation does not work at all
- Check Tab and Panel components
- The causeRerender – to avoid other atrocities –> single component, “manual” re-render

Client-side Routing (after HTML5)

- checkout: manual path routing example
- Use new tab in chrome
- Upper navigation does not work until refresh
- See history (state) in browser
- Try rewriting URL to a different string
- Use new tab in chrome (again)
- Replace pushState with replaceState
- Add historyApiFallback to webpack dev server settings and restart it (npm run start)

Routers

- checkout: revert to "some layout"
- Install packages react-router-dom and its @types
- Wrap everyting in App’s rendered with BrowserRouter
- Nothing changes, app should work as before

Links

- Replace links for Home and About in Menu: a href –> Link to
- See that URL change for Home and About
- See that clicking on Routing link still reloads
- See that Profile link does not work
- Set webpack dev server setting again (historyApiFallback: true)

Dynamic routing

- Add two Route components to Container (remind that it now works through component tree)
    - One for path="/", use render={ All roads lead to Home. }
        - Avoid Exact first to show that both will be rendered
    - One for path="/About", ue render={ I have a bad feeling About this. }

Route component

- checkout: add basic route and navigation
- See (custom) Navigation\NavLink
    - match is null if path does not match
    - see react-router's NavLink
- Compare render vs component vs children

Inclusive rendering

- checkout: showcase inclusive routing
- Click though all link parts
- Matches work via "/" not substrings (see inCONclusive part of the link)
- Have a look all usages of exact

Exclusive rendering

- checkout: showcase exclusive routing
- Click though all link parts
- Remove exact's first
- Click though all link parts
- Reorder by longest route
- Click though all link parts
- Revert changes - exact "beats" order

Parameter

- checkout: showcase route parameters
- See main component (rendered in Content.tsx)
- See RouteContains
    - Explore types – why is there any in RouteProps?
- See main component (again, but concentrate on individual RouteContainses props)
    - click through
- See Article
    - Explore types
    - Click on the "sample"
    - Click on the "other"
- Parameter validation in Recursive routing

Recursive routing

- checkout: showcase recursive routing
- Click through the book (aka old adventure play books)
- Static routing limitation: route would have to be parsed manually by a component)
- Explore the parameter
    - See that not-number won't hit
    - See that non-existent number will (console error –> "badly written" component)
- See BookPage component
    - Understand the route starts to exist with new "page turn"

Redirect component

- checkout: declarative routing
- See around – click on:
    - click on Muffin of color
    - click on Tasteful cake
    - click on Home
    - click on No, I love cakes
    - click on Graceful cake
    - click on White muffin
- Static routing hell: app-wide guard
- See main component state
    - The First switch either renders the error or one of the Lovers (with button in it)
    - Based on Freak flag, first or second redirect is rendered
        - both "normal" (/cakes and /muffins) routes just show path
    - Muffins redirect uses parameter passing (show redirect in FreaknessError as well)
    - Cakes redirect uses location state – parameter stored in history
        - See type in FreaknessError
        - referrer could be just the pathname string, but it can store whole objects allowing redirect back to original with full route knowledge including parameters and state

withRouter & redux - withRouter

- checkout: make components pure (basic)
- See that routes don't work
- Wrap Menu and Content to a Route
    - so Router props are provided to the PureComponent (show in dev tools)
- See routes don't work – Loader does not re-render
- wrap Loader to Route render
- to bring Router props closer to individual components, withRouter HoC can be used
- checkout: refactor pure components using withRouter

withRouter & redux - redux

- checkout: add redux and counter
- See intended functionality
    - Look at BasicRoutes
        - react-router does not (intend to) provide propTypes
    - See increaseCounter action
    - See reducer
        - eventually checkout unknown
- Write container for BasicRoutes component
    - Create file. Types are provided in ./src/@types/redux.d.ts
    - Start writing mapStateToProps: MapStateToProps, generic requires StateProps
    - Go to BasicRoutes component, split props interface –StateProps and –DispatchProps and introduce union type
    - Go back to container, finish mapStateToProps
    - Write mapDispatch to props, without dispatch function
        - Just return ({ onButtonClick: () => increaseCounter() }))
    - Assign connect to const BasicRoutesConnected
        - Don't forget about type: React.ComponentType
    - Go to Content, remove import of BasicRoutes and props from the component invocation
        - Use the new container instead
    - Verify it increases counter, but routes don't work
        - well, they start working once the button is clicked
- The container acts somewhat similar to PureComponents
    - It only triggers re-rendering when necessary (~ mapsStateToProps or ownProps (below) change)
    - Since withRouter is **behind** the connect, HoC does not know about any props that have changed
- Move withRouter to the container
    - Remove propTypes (partially) and RouteComponentProps from BasicRoutes component
        - Remove withRouter from BasicRoutes component (import at top as well)
    - Switch to container and create routered variant:
        - BasicRoutesWithRouter: React.ComponentType = withRouter(…)
    - Declare type BasicRoutesOwnProps = RouteComponentProps and fix typings
    - Verify it works
- Rerendering went wild again, while it is fixed in two following commits, we won't explore it in presentation
- See connected-react-router package for more efficient integration with Redux