141-Setting up Logout to Work With New Routes video

- there is an issue with this code, when we try to logout, it won't let us logout, we don't get redirected to the home page like before
- and the reason is because of the new Route component that we created
- so in routes.js we are going to be adding a prop with the name privacy with values of unauth or auth
-

```
export const routes = (
  <Router history={browserHistory}>
      <Route path="/" component = {Login} privacy="unauth" onEnter = {onEnterPublicPage}/>
      <Route  path="/signup" component={Signup} privacy="unauth"  onEnter={onEnterPublicPage}/>
      <Route path="/dashboard" component={Dashboard} privacy="auth"  onEnter=
{onEnterPrivatePage}/>
      <Route path="/dashboard/:id" component={Dashboard} privacy="auth"  onEnter={onEnterNotePag
e}/>
      <Route path="*" component={NotFound}/>
  </Router>
);
```

- React Router allows you to nests instances of Route
- so will be creatind a new Route component and we will put all of our Routes there

```
export const routes = (
  <Router history={browserHistory}>
    <Route>
      <Route path="/" component = {Login} privacy="unauth" onEnter = {onEnterPublicPage}/>
      <Route  path="/signup" component={Signup} privacy="unauth"  onEnter={onEnterPublicPage}/>
      <Route path="/dashboard" component={Dashboard} privacy="auth"  onEnter=
{onEnterPrivatePage}/>
      <Route path="/dashboard/:id" component={Dashboard} privacy="auth"  onEnter={onEnterNotePag
e}/>
      <Route path="*" component={NotFound}/>
    </Route>
```

```
  </Router>
);
```

- in this new Route we will have two props onEnter and onChange

```
export const routes = (
  <Router history={browserHistory}>
    <Route onEnter onChange >
      <Route path="/" component = {Login} privacy="unauth" onEnter = {onEnterPublicPage}/>
      <Route  path="/signup" component={Signup} privacy="unauth"  onEnter={onEnterPublicPage}/>
      <Route path="/dashboard" component={Dashboard} privacy="auth"  onEnter=
{onEnterPrivatePage}/>
      <Route path="/dashboard/:id" component={Dashboard} privacy="auth"  onEnter={onEnterNotePag
e}/>
      <Route path="*" component={NotFound}/>
    </Route>
  </Router>
);
```

- now we are going to create two function for those two new props

```
export const globalOnChange = ()=>{
};


export const globalOnEnter= ()=>{
};


export const routes = (
  <Router history={browserHistory}>
    <Route onEnter={globalOnEnter} onChange={globalOnChange}>
      <Route path="/" component = {Login} privacy="unauth" onEnter = {onEnterPublicPage}/>
      <Route  path="/signup" component={Signup} privacy="unauth"  onEnter={onEnterPublicPage}/>
      <Route path="/dashboard" component={Dashboard} privacy="auth"  onEnter=
{onEnterPrivatePage}/>
```

```
      <Route path="/dashboard/:id" component={Dashboard} privacy="auth"  onEnter={onEnterNotePag
e}/>
      <Route path="*" component={NotFound}/>
    </Route>
  </Router>
);
```

- now lets console log it so we can see what is going on on the console

```
export const globalOnChange = ()=>{
  console.log('globalOnChange');
};

export const globalOnEnter= ()=>{
  console.log('globalOnEnter');
  debugger;
};
```

- when we check the console, we only get the globalOnEnter string, but when we click on the link in our app, the globalOnChange will appear
- now lets define some arguments in our functions
- lets drop a debugger in our code to see what is going on in our program

```
export const globalOnChange = (prevState, nextState)=>{
  console.log('globalOnChange');
};

export const globalOnEnter= (nextState)=>{
  console.log('globalOnEnter');
  debugger;
};
```

- and this is what we see in the console when we type nextState
-

```
nextState

{routes: Array(2), params: {…}, location: {…}}

location : {pathname: "/", search: "", hash: "", state: undefined, action: "POP", …}

params:{}

routes:(2) [{…}, {…}]

__proto__

:Object
```

- and now lets explore routes
- in our routes we see that we have the privacy prop that we created

```
{routes: Array(2), params: {…}, location: {…}}

location:{pathname: "/", search: "", hash: "", state: undefined, action: "POP", …}

params:{}

routes:Array(2)

    0:{childRoutes: Array(5), onEnter: ƒ, onChange: ƒ}

    1:{path: "/", privacy: "unauth", component: ƒ, onEnter: ƒ}

    length:2

    __proto__:Array(0)

__proto__:Object
```

- so we are going to be grabing the last route in the routes Array to have access to the privacy prop to determine what type of page we are on
- so in our globalOnEnter lets create a const that will have this value

```
export const globalOnEnter= (nextState)=>{

  const lastRoute = nextState.routes[nextState.routes.length - 1];

  console.log('globalOnEnter');

  debugger;

};
```

- and now we can do a Session.set ,and the we can provide the key value pair

```
export const globalOnEnter= (nextState)=>{

  const lastRoute = nextState.routes[nextState.routes.length - 1];
```

```
  Session.set('currentPagePrivacy', lastRoute.privacy);
};
```

- now in the other function globalOnChange lets just call the globalOnEnter that takes one parameter

```
export const globalOnChange = (prevState, nextState)=>{
  globalOnEnter(nextState);
  console.log('globalOnChange');
};

export const globalOnEnter= (nextState)=>{
  const lastRoute = nextState.routes[nextState.routes.length - 1];
  Session.set('currentPagePrivacy', lastRoute.privacy);
};
```

- we are going to be making some changes to main.js
- on Tracker.autorun we are going to be fetching the value of the Session.set we created called currentPagePrivacy

```
import { Meteor } from 'meteor/meteor';
import ReactDom from 'react-dom';
import { Tracker } from 'meteor/tracker';
import { Session } from 'meteor/session';
import { browserHistory } from 'react-router';

import { routes, onAuthChange } from '../imports/routes/routes';
import '../imports/startup/simple-schema-configuration.js';

Tracker.autorun(()=>{
  const isAuthenticated = !!Meteor.userId();
  const currentPagePrivacy = Session.get('currentPagePrivacy');
  onAuthChange(isAuthenticated);
});
```

- now lets dump this to the screen by using console.log

```
Tracker.autorun(()=>{
  const isAuthenticated = !!Meteor.userId();
  const currentPagePrivacy = Session.get('currentPagePrivacy');
  console.log('currentPagePrivacy ',currentPagePrivacy);
  onAuthChange(isAuthenticated);
});
```

- and this is what it shows on the console

```
currentPagePrivacy  undefined
currentPagePrivacy  unauth
```

- **and now when we login...**

```
currentPagePrivacy  auth
```

-