Chapter 28: Higher Order Components

Higher Order Components ("HOC" in short) is a react application design pattern that is used to enhance components with reusable code. They enable to add functionality and behaviors to existing component classes.

A HOC is a pure javascript function that accepts a component as it's argument and returns a new component with the extended functionality.

Section 28.1: Higher Order Component that checks for authentication

Let's say we have a component that should only be displayed if the user is logged in.

So we create a HOC that checks for the authentication on each render():

AuthenticatedComponent.js

```
import React from "react";

export function requireAuthentication(Component) {
    return class AuthenticatedComponent extends React.Component {

        /**
         * Check if the user is authenticated, this.props.isAuthenticated
         * has to be set from your application logic (or use react-redux to retrieve it from global
state).
         */
        isAuthenticated() {
            return this.props.isAuthenticated;
        }

        /**
         * Render
         */
        render() {
            const loginErrorMessage = (
                <div>
                    Please <a href="/login">login</a> in order to view this part of the
application.
                </div>
            );

            return (
                <div>
                    { this.isAuthenticated === true ? <Component {...this.props} /> :
loginErrorMessage }
                </div>
            );
        }
    };
}

export default requireAuthentication;
```

We then just use this Higher Order Component in our components that should be hidden from anonymous users:

MyPrivateComponent.js

```
import React from "react";
import {requireAuthentication} from "./AuthenticatedComponent";

export class MyPrivateComponent extends React.Component {
    /**
     * Render
     */
    render() {
        return (
            <div>
                My secret search, that is only viewable by authenticated users.
            </div>
        );
    }
}

// Now wrap MyPrivateComponent with the requireAuthentication function
export default requireAuthentication(MyPrivateComponent);
```

This example is described in more detail here.

Section 28.2: Simple Higher Order Component

Let's say we want to console.log each time the component mounts:

hocLogger.js

```
export default function hocLogger(Component) {
  return class extends React.Component {
    componentDidMount() {
      console.log('Hey, we are mounted!');
    }
    render() {
      return <Component {...this.props} />;
    }
  }
}
```

Use this HOC in your code:

MyLoggedComponent.js

```
import React from "react";
import {hocLogger} from "./hocLogger";

export class MyLoggedComponent extends React.Component {
    render() {
        return (
            <div>
                This component gets logged to console on each mount.
            </div>
        );
    }
}

// Now wrap MyLoggedComponent with the hocLogger function
export default hocLogger(MyLoggedComponent);
```