# **React From The Ground Up**

lärare: <http://davidtkatz.com/#/about>

kod: <https://github.com/15Dkatz/ReactJS_VideoGuides>

Plattform: Udemy [1]

## Intro

React är toppar listan, och har det gjort ett tag nu, på GitHubs hemsida. React ligger just nu på sjätte plats och visar tydligt att det är väl andvändt idag. React skapades… har idag växt till en av de popluäraste sätt att bygga en webbapplikation. och därmed bra är biblioteket bra underhållet. [2]

React är skapat och underhålls utav Facebook och deras utvecklare. Reacts bibliotek har blivit mycket populärt kanske endel eftersom det funkar så bra med Facebook. Men även stora företag som Netflix, Airbnb och Walmart stödjs utav React. [1]

### React, ES6 & Redux

React är…

Själva Reactkoden skrivs och körs som JavaScript och koden följer ES6, den sjätte versionen av JavaScript som kom först i Juni 2015 [1].

Tillsammans med React kör man ofta ett annat bibliotek vid namn **Redux.** Redux tillåter oss att bygga applikationer som kan växa. Redux är en bra kombination med React som gör att sättet att jobba blir tightare. [1]

#### Golden React Rules

* Never mutate state directly.

In order to change your state data, you can’t manipulate it like you would with normal variables.

For example, something like `x++` or `counter += 1` would be fine in any other case. But in order to update state, you must use the **setState()**function and have new instances of data. [1]

### Verktyg – att installera React

Man kommer behöva att installera en del verktyg och program för att kunna köra React och Redux tillsammans på sin plattform. Man behöver helt enkelt förbereda för att så smidigt som möjligt kunna konfigurera och jobba med Reactprojekt. Tack och lov finns det många nya sätt som hjälper oss med detta.

#### Node.js

<https://nodejs.org/en/>

Behövs endast installeras på din dator. Använd den senaste rekommenderade versionen för ett stabilt resultat. Skriv kommandot node -v i konsolen för att se om det funkar och du har rätt version. Skriv npm -v i terminalen för att kolla vilken version. Det ska vara en version som är högre än 3 för att kunna köras med Node. Npm kommer med Node, vanligtvis. Detta program hanterar de olika modulerna och paketen i Node.

#### Atom

<https://atom.io/>

Något att skriva kod i. I detta projekt kommer skrivas i Atom.

Efter installationen av Atom, om den är på en Windows plattform, så kan man använda atom . i konsolen för att starta upp atom i den projekt mappen man står för nuvarande i.

#### create-react-app

För att installera ett enkelt sätt att skapa ett React projekt.

<https://github.com/facebook/create-react-app>

npm install -g create-react-app. För att initialitiera ett React projekt.

create-react-app ”my-name”

Ger en tre mappar

node\_modules – Med all React kod och alla andra packet

public – index fil i html. Första som visas.

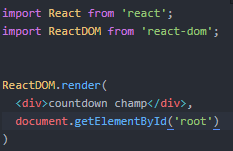
Src – Egna javascript och css filer. Where we HOST all the React code.

det är I den man kommer jobba mest.

För att sedan starta

Gå in i projekt mappen och skriv npm start.

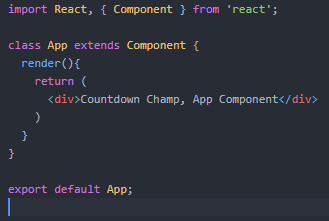
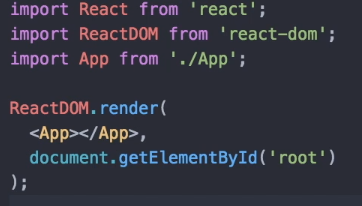
### Starta Projekt – Redigera din applikation

En bra sak att börja efter du har skapat din React applikation med kommandot kan vara att ta bort all förinställd Reactkod. Detta genom att helt enkelt ta bort scr mappen. Detta för att enkelt kunna börja på ny kula genom att skapa en ny scr mapp, dom nu är tom på innehåll. I denna mappen skapar du sedan det fil som kommer köras först av all Reactkod, nämligen index.js. Den filen hämtar vart på index.html (den enda html-sidan) som Reactkoden ska skrivas och renderas.

¨

#### Skapa en React komponent

I React jobbar man med att skapa sina egna komponenter som man sedan bygger upp hela sitt projekt med.

* Start by create App.jsx
  + JS with the javascript extension
  + Importera React och Component som vi ska använda.
  + Skapa en class med egenskaperna av en React komponent.
    - Som renderar och returnerar html.
  + Exportera sedan komponenten.
  + Importera sedan App i index.js
  + Och lägg till komponenten i DOM renderingen.

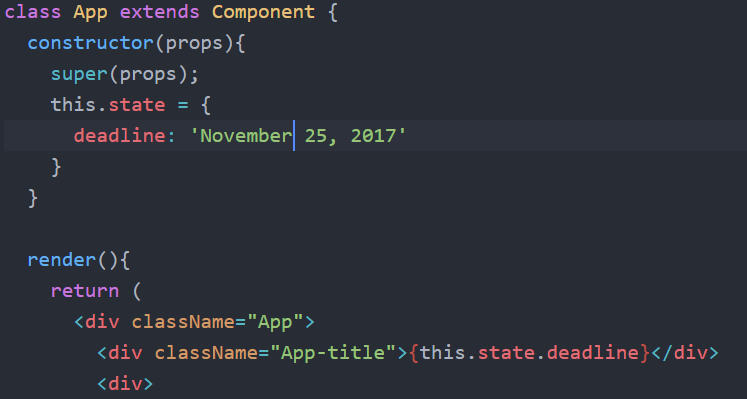
#### Strukturen I en React applikation

När man jobbar med React är det viktigt att veta vilka slags filer som ska ligga på vilka slags platser. För att Reactkoden ska kunna fungera beövs en viss struktur på mappstrukturen av projektet och det måste innehålla vissa filer.

1. Ta bort dubbel taggen från skapade komponenter i ReactDOM renderringen
   1. <App></App> == <App/>
2. Add CSS
   1. App.css in src
   2. Importera in i jsx filen som ska ha egenskaperna.
      1. Används className istället för class i jsx renderingen av den fil som använder den skapade cssen.

### State

* You can think of state as the data that is pertinent to the application.
  + De uppgifter som är relevanta för applikationen.
  + Varje komponent har sitt egna lokala state.
    - With respect to the global state of the entire app.
      * Ex a school witch has a state of the number of teatchers, number of students, classrooms etc.
        + But each student also has a own state, a number of textbooks and pens etc.
* För att lägga till state till componenten lägger man till en constructor.
  + Countructor(props){super(props)}
    - I constructorna ska man deklarera det state till komponenten.
      * I React är state alltid ett objekt.
      * Sedan hämtar man sitt objekt i renderingen via {this.state.valt state.}



#### Uppdatera State

* För att uppdatera vårat state dynamiskt i våran applikation kan man ta fördel av en specifik metod i React.
  + Denna metod kallas för setState.
  + Man måste kalla på this.setState genom koden för att ändra eller uppdatera på lokala klassers data.
    - För skapar man en valfri metod som ska uppdatera state.
      * ”You must never mutate or change state directly”
        + Meaning the component will not read render if we mutate directly like this.state.deadline = ‘vad vi byter till.’
        + Måste ske genom setState() 🡪 En metod
* För att kalla på en skapad metod i vår knapp använder man onClick i renderingen.
  + I onClicken skapar man en ES6 funktion där man lägger in this.metoden
    - ES6 så kommer den köras en gång och rätt.

Kolla count-down-champ\_v1



### Props

### Lifecycle methods

### Adding bootstrap to our App

## Källförteckning

[1] – Udemy Course

[2] – https://github.com/trending/javascript

## Tillhörande material

2.16 Create Clock Component

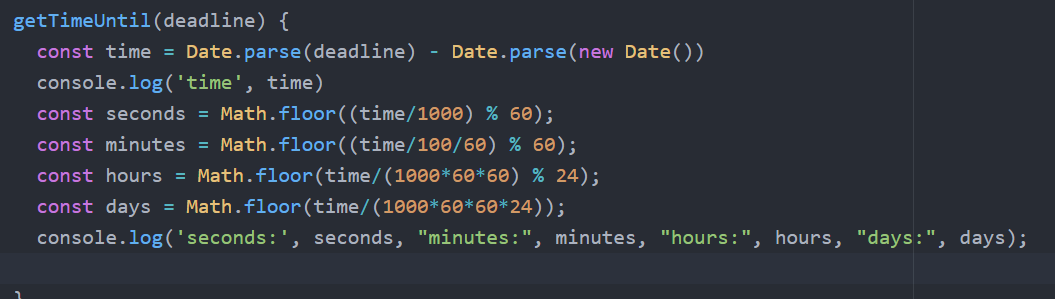
Eftersom våra fyra tids divvar ser ungefär likadan ut är det perfekt att skapa en komponent för dessa.

1. Skapa en ny komponent i scr mappen
   1. En ny jsx komponent.
   2. Exportera och importera i jsx filen där den ska ligga. Glöm inte lägga till den i renderingen som en komponenet.

2.17 Props

* Props refers to some data within the application.
* With props you pass data or state to child components from parent components
  + - Meaning o can specify a deadline within the component in the rendering method.

2.19

1. In JS we don’t use var when declaring variables. We use const and let.
   1. Must often use const for varables that never need updating.
   2. And let for variables that do need updating.

2.20 Lifecycle methods

1. componentWillMount()
   1. React function.
   2. This one runs before the component has completely rendered to the component.
   3. Ser till så att komponentens state updateras typ. Kolla upp detta lite mer.
   4. Men detta är något som ingår i en React komponent.
2. componentDidMount()
   1. This one runs after the component has completely

2.22 Adding bootstrap to our App

1. <https://react-bootstrap.github.io/>
   1. <https://react-bootstrap.github.io/getting-started/introduction>
2. npm install react-bootstrap –save
   1. in project folder 🡪 To install React Bootstrap in project.
3. <!-- Latest compiled and minified CSS to our html -->

**<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/cs**s/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

**Countdown Champ**

Congratulations on finishing countdown champ! You just built your first app with React JS. Here’s an overview of the important concepts that you just learned:

* **React Components** - re-usable and independent pieces of React code that comprise the User Interface.
* **ES6 importing and exporting -**a new syntax for sharing code between separate files. Used in cases like `import React from ‘react’`
* **State -**the pertinent data to an application. Each component has its local state as long as you declare add the constructor to a Component and declare its state object.
* **Updating State -**When updating state, make sure to **never** mutate state directly. Doing so will lead to fatal errors in your application. Instead, re-declare new instances of state arrays or objects and use the setState() function to update state.
* **Props -**similar to state, except this data inherits from parent component specifying pieces of state as properties.
* **LifeCycle Methods -**events in React components that trigger in cases such as rendering on or off the screen, or during state updates. One example is the `componentDidMount()` lifeCycle hook.

Now, how about testing yourself and tackling a challenge or two?

**Challenges:**

1. Make a separate component that works like a stopwatch. The user inputs the amount of time for the clock to run, and it starts counting down to 0.
2. Add an alert message that pops up when the timer counts down to 0.
3. Try running `npm build` for a build version and host the code on your website or any other host!