Laborationsrapport

Moment 4 / Frontend ramverk
DT162G, Javascriptbaserad webbutveckling

Författare: Elias Eriksson, eler2006@student.miun.se
Termin, år: VT, 2021

Mittuniversitetet
MID SWEDEN UNIVERSITY

Campus Härnösand Universitetsbacken 1, SE-871 88. **Campus Sundsvall** Holmgatan 10, SE-851 70 Sundsvall.
Campus Östersund Kunskapens väg 8, SE-831 25 Östersund.
Phone: +46 (0)771 97 50 00, Fax: +46 (0)771 97 50 01.

Innehållsförteckning

1 Inledning

I denna uppgift kommer JavaScript ramverken Angular, Vue och React att jämföras. I jämförelsen kommer det att ingå vad ramverken i huvudsak fokuserar på och hur de skiljer sig från de andra två samt hur de har presterat i bloggen DailyJS årliga jämförelse av JavaScript ramverk. Ett kod exempel kommer också att finnas med om React för att dyka lite djupare i hur React fungerar.

2 Angular

Angular utvecklat av Google och är det ramverk av dessa tre som har kvar på användningen av TypeScript. Eftersom att det är utvecklar utav Google så finns det ingen oro om att projektet inte blir tillräckligt finansierat. Angular är även känt som AngularJS som är versionerna 1.x utav Angular. AngularJS kommer att nå EoL 2022 och all AngularJS kod rekommenderas att uppgraderas till Angular innan detta datum. [2][3][4]

Angular är ett fullt utrustat ramverk och ska därmed inte behöva använda sig av tredje parts verktyg för att kunna utföra standard operationer för utveckling av UI och navigation mellan olika sidor. [2]

Angular är ett ramverk med starka åsikter om hur något ska utföras. Så om någonting ska göras i Angular så finns det ofta ett rätt sätt att göra det på. [2]Detta gör att det blir lätt att utveckla det som ska göras så länge utvecklarna av Angular hade det i åtanke men om det man ska göra går utanför ramverkets utvecklares syn så kan det bli drygare att implementera.

Vid DailyJS prestanda jämförelse med Lighthouse för mobil så får Angular 82 poäng. Vilket är betydligt bättre än föregångaren AngularJS som får 63 poäng. Vid DailyJS filstorleks så är Angular sidan 141 KB vilket också är bättre än AngularJS sida som är 317KB.[1]

3 Vue

Vue är ett ramverk som inte har ett stort företag bakom sig utan är istället utvecklat utav organisationen Vue och startades utan Evan You. Eftersom att Vue inte har ett specifikt stort företag bakom sig så är de beroende av supportrar och donationer och donationer. Det uppmuntrar företag som använder Vue att donera till dom årligen för att kunna ge support åt något företaget använder. [5][6]

Vue är inspirerat av Angular och är precis som Angular ett fullt utrustat ramverk. Till skillnad från både Angular och React är det inte ett krav att använda Vue över en hel webbplats. Vue kan därför användas på en specifik del av en sida där Vues funktionalitet behövs och det gör det lätt att migrera över en kod bas som inte använder något front-end ramverk till Vue då det kan göras lite i taget. [2] [13]

Vue är inte ett stort ramverk och är jämfört med Angular och React väldigt litet i storleken. I DailyJS jämförelse så blir deras demo sida byggd i Vue endast 71KB. Vue är också snabbt i jämförelse och får i DailyJS jämförelse 86 poäng i deras test med Lighthouse för mobil. [1]

Den nuvarande huvudversionen av Vue är Vue 3. En stor ändring från Vue 2 till Vue 3 är att Vue 3 inte längre har stöd för Internet Explorer.[7]

4 React

React precis som Angular har ett stort företag bakom sig. Facebook startade utvecklingen av React och vart tillgängligt för alla år 2013. [2]

Till skillnad från Angular så är det inte ett krav att använda TypeScript och kommer det är inte heller ett fullt utrustat ramverk. React är marknadsfört som ett ramverk som är till för att bygga UIs och kommer därför inte med inbyggda funktioner för att skapa sökvägar mellan sidor. Om det är önskvärt att ha tillgång till sådana verktyg så finns det tredje parts bibliotek för det. Populära tillval kan vara React Router. Eftersom att React inte är ett full utrustat ramverk så finns det inte alltid så starka åsikter om hur ett visst problem ska lösas.[2]

React är ett komponent baserat ramverk där varje komponent kan ha statiska ”properties”, ett internt ”state” och ska kunna generera JSX. JSX är en form av en blandning mellan HTML och JavaScript. Komponentens properties och state kan användas för att beskriva den JSX som ska genereras och resultatet skrivs sedan ut som vanligt HTML till DOM. När en komponents state ändras med funktionen setState kommer komponenten att ritas om och kommer till slut att uppdatera det som tidigare genererats till DOM.[8][11][12]

En stor fördel med React är att det finns två olika versioner av React. ReactJS finns för att utveckla mot webb och React Native för att utveckla mot mobil. De olika versionerna skiljer sig litegrann men koden är i stor del väldigt lik vilket gör det lätt för en utvecklare att byta från ReactJS till React Native. [10]

I DailyJS prestanda jämförelse med Lighthouse på mobil fick React + Redux 67 poäng oh deras demosida de byggt var 193 KB.[1]

Reacts senaste huvudversion är React 17.[9]

5 React Exempel – Rulla tärning

Ett React projekt startas genom att köra npx kommandot

npx npx create-react-app site --template typescript

En funktion för att simulera tärningsrullningar skapas först. Denna Funktion antar att tärningens sida inte kan vara 0 och ger summan av vad antalet tärningar slog.

```
const rollDice = (numberOfDice: number, sides: number): number => {
    let sum = 0;
    for (let i = 0; i < numberOfDice; i++) {
        sum += Math.floor(Math.random() * (sides - 1 + 1) + 1);
    }
    return sum;
}
```

En enkel komponent som bara innehåller statisk HTML för sidans header. Fördelen med att göra det till en komponent är att komponenten kan användas på andra sidor för att skapa exakt samma header.

```
class Header extends React.Component {
    render = (): JSX.Element => {
        return (
            <div className="content-wrapper header">
                <header className="content">
                    <h1>Den fantastiska React sidan!</h1>
                </header>
            </div>
        );
    }
}
```

Precis som för headern så skapas en komponent för footern. Denna komponent är inte helt statisk som Header komponenten. Denna komponent utnyttjar properties för att dynamiskt kunna ändra repo om det skulle behövas. Eftersom att Projektet är skrivet i TypeScript ser vi att propertyn heter repoLink och är en string och vi ser sen i render metoden att den används för att skapa länken i <a>.

```
class Footer extends React.Component<{ repoLink: string }> {
   render = (): JSX.Element => {
      return (
         <div className="content-wrapper footer">
            <footer className="content">
               <p>Elias Eriksson ©  | <a target="_blank" rel="noreferrer"
href={this.props.repoLink}>Repo</a></p>
            </footer>
         </div>
      );
   }
}
```

Submit input komponenten nedan precis som i Footer komponenten använder en property för att bestämma vad som ska stå på knappen.

```
class SubmitInput extends React.Component<{ label: string }> {
    render = (): JSX.Element => {
        return (
            <input type="submit" value={this.props.label}/>
        );
    }
}
```

Nummer input komponenten precis som submit input komponenten använder sig av properties men också utav state. Via TypeScript ser vi att komponentens properties är en label och ett standard nummer. Sen ser vi också att state variabeln ska ha ha ett number attribut. Eftersom att komponenten har ett state så måste state initialiseras vilket görs i konstruktorn. State attributet number initialiseras till standard numret om det finns annars 0.

```
class NumberInput extends React.Component<{ label: string, defaultNumber?:
number }, { number: number }> {
    constructor(props: { label: string, defaultNumber?: number }) {
        super(props);
        this.state = {
            number: props["defaultNumber"] ?? 0
        }
    }

    onChange = (event: ChangeEvent<HTMLInputElement>) => {
        this.setState(() => ({
            number: parseInt(event.target.value)
        }));
    }

    render = (): JSX.Element => {
        return (
            <label>
                {this.props.label}
                <input onChange={e => {
                    this.onChange(e)
                }} type="number" value={this.state.number}/>
            </label>
        );
    }
}
```

DiceRollForm är är en komponent som hanterar tärningsrullandet. Komponenten har två referenser till sina input fält. Dessa krävs för att kunna läsa av input fältens state och se vad fälten för närvarande innehåller. Formet kommer också såklart att ha en submit knapp men en referens till denna knapp behövs inte då knappen kommer att trigga formet att skickas. DiceRollForm har även två metoder som beskrivs nedan.

```
class DiceRollForm extends React.Component<{}, { result?: number }> {
    private readonly numberOfDice: React.RefObject<NumberInput>;
    private readonly diceMaxValue: React.RefObject<NumberInput>;

    constructor(props: {}) {
        super(props);
        this.numberOfDice = React.createRef<NumberInput>();
        this.diceMaxValue = React.createRef<NumberInput>();
        this.state = {
            result: undefined
        }
    }
}
```

När DiceRollForm blir skickat fångas eventet och förhindrar att sidan laddas om. Värdet på båda formets input fält läses från dess komponenters state och om värdena är större än 0 så körs funktionen för att rulla tärning. Resultatet från tärningsrullandet sparas till DiceRollForms state.

```
onSubmit = (event: FormEvent) => {
    event.preventDefault();

    const numberOfDice = this.numberOfDice.current?.state.number ?? 0;
    const numberOfSides = this.diceMaxValue.current?.state.number ?? 0;
    if (numberOfDice > 0 && numberOfSides > 0) {
        this.setState(() => ({
            result: rollDice(
                numberOfDice,
                numberOfSides
            )
        }));
    }
}
```

DiceRollForms render funktion När formet renderas så kontrolleras först om Formet state för rullningsresultatet har ändrats från undefined. Om det har det så kommer även en <p> att renderas med resultatet. I denna kod ser vi även hur ett element binds till en utav referenserna skapade i konstruktorn fungerar. Ett element binds till en referens med JSX attributet ref.

```
render = (): JSX.Element => {
  if (this.state.result !== undefined) {
    return (
      <div>
        <form onSubmit={event => this.onSubmit(event)}>
          <NumberInput label={"Antal tärningar:"}
                       defaultNumber={2}
                       ref={this.numberOfDice}/>
          <NumberInput label={"Antal sidor på vardera tärning:"}
                       defaultNumber={6}
                       ref={this.diceMaxValue}/>
          <SubmitInput label={"Rulla tärningar"}/>
        </form>
        <p>Du rullade {this.numberOfDice.current?.state.number} tärningar
          med {this.diceMaxValue.current?.state.number} sidor och fick summan
{this.state.result}</p>
      </div>
    );
  }
  return (
    <div>
      <form onSubmit={event => this.onSubmit(event)}>
        <NumberInput label={"Antal tärningar:"} defaultNumber={2}
ref={this.numberOfDice}/>
        <NumberInput label={"Antal sidor på vardera tärning:"} defaultNumber={6}
ref={this.diceMaxValue}/>
        <SubmitInput label={"Rulla tärningar"}/>
      </form>
    </div>
  );
}
```

Main komponenten nedan lägger ihop allting i en main tagg och hela sidan exporteras sedan.

```
class Main extends React.Component {
  render = (): JSX.Element => {
    return (
      <div className="content-wrapper main">
        <main className="content">
          <DiceRollForm/>
        </main>
      </div>
    );
  }
}

export default (
  <React.StrictMode>
    <div className="window">
      <Header/>
      <Main/>
    </div>
    <Footer

repoLink="https://github.com/EliasEriksson/miun/tree/master/webbutveckling/
JavascriptbaseradWebbutveckling/Moment4"
    />
  </React.StrictMode>
);
```

I index.js filen importeras de hopslagna komponenterna under namnet App och React lägger komponenten under HTML elementet med id root i HTML filen.

```
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

ReactDOM.render(
  App,
  document.getElementById('root')
);

reportWebVitals();
```

Hela kodexemplet bifogas med uppgiften och den färdiga webbplatsen kan ses på https://miun.eliaseriksson.io/jsweb/moment4/.

6 Slutsatser

Det finns för och nackdelar med alla tre olika ramverk och beroende på vad som eftersöks med ett ramverk så kan det hända att ett specifikt ramverk är föredraget över det andra eller det tredje.

Angular har fördelen med att det är kommer med allt som behövs och det kan vara lättare att hitta exakt hur ett specifikt problem ska lösas då ramverket har starka åsikter om hur problem ska lösas.

Om istället prestanda är av större intresse är Vue kanske mer relevant då Vue hade överlägset bäst poäng i DailyJS tester med lägst filstorlek på 71 KB jämfört med Angular och React som hade 141 KB respektive 193 KB. Samt att Vue presterade best enligt DailyJS Lighthouse test för mobil med en poäng på 86 jämfört med Angular och React som hade 82 respektive 67. Med testerna är det lätt att se att Vue presterar bäst med Angular väldigt snarlik när det kommer till hastighet och React hamnar lite bakom de andra två.

React presterade sämst utav dessa tre men fördelarna med React över Angular är att det inte är lika satt i sten hur vissa problem löses och det kanske är önskvärt för vissa utvecklare och React till skillnad från både Angular och Vue har React utvecklare lätt att hoppa till mobilutveckling om det är önskvärt senare.

7 Källförteckning

[1] DailyJS ”RealWorld Comparison” https://medium.com/dailyjs/a-realworld-comparison-of-front-end-frameworks-2020-4e50655fe4c1

[2] John Hannah ”The Ultimate Guide to JavaScript Frameworks” https://jsreport.io/the-ultimate-guide-to-javascript-frameworks/

[3] Mark Thompson ”Finding a Path Forward with AngularJS” https://blog.angular.io/finding-a-path-forward-with-angularjs-7e186fdd4429

[4] AngularJS ”https://docs.angularjs.org/misc/version-support-status” https://docs.angularjs.org/misc/version-support-status

[5] Vue ”Meet the Team” https://v3.vuejs.org/community/team.html

[6] Vue ”Sponsor Vue.js Development” https://v3.vuejs.org/support-vuejs/#recurring-pledges

[7] Vue ”vue-next README” https://github.com/vuejs/vue-next

[8] React ”Introducing JSX” https://reactjs.org/docs/introducing-jsx.html

[9] React ”React CHANGELOG” https://github.com/facebook/react/blob/main/CHANGELOG.md#1700-october-20-2020

[10] Joshjnunez ”Transitioning from ReactJS to React Native (Web to Mobile Development)” https://medium.com/@joshjnunez09/transitioning-from-reactjs-to-react-native-web-to-mobile-development-85b86425e26 .

[11] React ”Components and Props” https://reactjs.org/docs/components-and-props.html

[12] React ”State and Lifecycle” https://reactjs.org/docs/state-and-lifecycle.html

[13] Vue ”Comparison with Other Frameworks” https://vuejs.org/v2/guide/comparison.html