

React web-appiksen perusanatomia

Aino Haikala, 2022

Alussa oli HTML

```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="utf-8" />
5      <title>Joulukalenteri</title>
6    </head>
7
8    <body data-theme="dark">
9      <div id="threejs-container"></div>
10    </body>
11  </html>
```

.... Ja CSSja Javascript

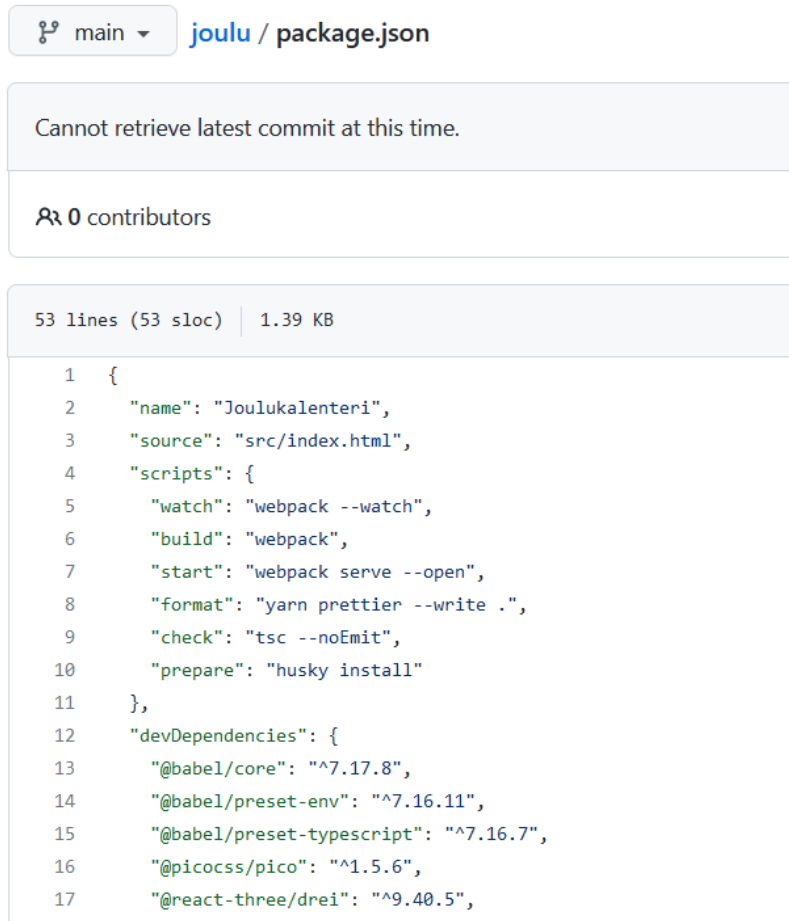
- <https://github.com/ainohai/joulu/tree/main/dist>
- ... mutta miksi meidän src-kansio ei näytä ollenkaan tältä?

Webpack

```
module.exports = {
  entry: './src/index.tsx',
  mode: 'development',
  devtool: 'inline-source-map',
  devServer: {
    static: './dist',
  },
  module: {
    rules: [
      {
        test: /\.tsx?$/,
        use: 'ts-loader',
        exclude: /node_modules/,
      },
      {
        test: /\.s[ac]ss$/i,
        use: ['style-loader', 'css-loader', 'sass-loader'],
      },
    ],
  },
}
```

- ”Kaikki” modernit webbisoftat on käännetty jollain tavoin ennen kuin ne päätyy selaimelle.
 - Ts => js, minimointi, eri selainten tuki, tyylitiedostojen prosessointi....
- .ts = typescript tiedosto, .tsx = typescript + jsx, .scss = sass
- <https://github.com/ainohai/joulu/blob/main/webpack.config.js> , tsconfig sisältää typescriptiin liittyvät konffit.
- (muitakin vaihtoehtoja on kuin webpack)

Yarn (npm) hoitaa paketinhallinnan



The screenshot shows a GitHub repository page for 'joulu / package.json'. At the top, there's a search bar with 'main' selected and a dropdown arrow. Below it, a message states 'Cannot retrieve latest commit at this time.' and '0 contributors'. The file size is '53 lines (53 sloc)' and '1.39 KB'. The code is a JSON file for a package named 'Joulukalenteri'. It includes scripts for watch, build, start, format, check, and prepare. It also lists devDependencies for @babel/core, @babel/preset-env, @babel/preset-typescript, @picocss/pico, and @react-three/drei.

```
1 {
2   "name": "Joulukalenteri",
3   "source": "src/index.html",
4   "scripts": {
5     "watch": "webpack --watch",
6     "build": "webpack",
7     "start": "webpack serve --open",
8     "format": "yarn prettier --write .",
9     "check": "tsc --noEmit",
10    "prepare": "husky install"
11  },
12  "devDependencies": {
13    "@babel/core": "^7.17.8",
14    "@babel/preset-env": "^7.16.11",
15    "@babel/preset-typescript": "^7.16.7",
16    "@picocss/pico": "^1.5.6",
17    "@react-three/drei": "^9.40.5",
```

- Vastaa mavenia.
- Mahdollistaa, että voi ladata puolet internetistä koneelle
 - Lokaalisti asennetut paketit tallentuvat node-modules-kansioon.
 - yarn.lock –tiedostoon tallentuvat käytetyt versiot.

Typescript ei toimi selaimissa

```
export type CardText = {  
  text: string;  
  size?: 'big' | 'normal';  
};  
  
export type SingleCard = CardText[];  
  
export type CardsOfDayType = {  
  [day: number]: SingleCard[];  
};  
  
export type CardState = {  
  cardsOfTheDay: SingleCard[];  
  visibleCardIndex: number;  
  days: string[];  
  nextCard: () => void;  
  anotherJoke: (index: string) => void;  
};
```

- JavaScriptin superskripti
- Joitakin nostoja:
 - Jos muuttuja voi olla undefined | null, se pitää ilmaista muuttujan tyypissä.
 - Muuttuja voi olla tyyppiä any.
 - Short-handejä, kuten ?? tai ?
 - () => void

Sitten itse asiaan...

```
import * as React from 'react';
import { createRoot } from 'react-dom/client';
import WinterCanvas from './components/canvas';
import Cards from './components/cards';
import TestNavigation from './components/testNavigation';
import './styles.scss';

export default function App() {
  return (
    <>
      <WinterCanvas />
      <Cards />
      <TestNavigation />
    </>
  );
}

createRoot(document.getElementById('threejs-container')).render(<App />);
```

<- Importoidaan tyylit

<- Funktio, joka "renderöi" UI- komponentit

<- tässä käytössä JSX. Sitä käytetään tyypillisesti (muttei välttämättä) ilmaisemaan UI-komponentin rakennetta.

<- Luo react-applikaation #threejs-containeriin

React-komponentit muodostavat käyttöliittymän

```
export default function Cards() {  
  const cards = useCardStore((state) => state.cardsOfTheDay);  
  
  return (  
    <main className="container">  
      <div className="centeredCard">  
        {cards.map((card, index) => (  
          <Card  
            card={card}  
            index={index}  
            totalNumOfCards={cards.length}  
            key={index}  
          />  
        ))}  
      </div>  
    </main>  
  );  
}
```

- Se mitä näkyy käyttöliittymällä rakentuu react-funktioilla.
- Komponenttien tilan muuttuessa ne renderöidään automaattisesti uudestaan.
- Muutamia react-käsitteitä:
 - Funktionaalinen komponentti vs luokkakomponentti
 - Props vs state
 - Hooks

Mistä tila tulee?

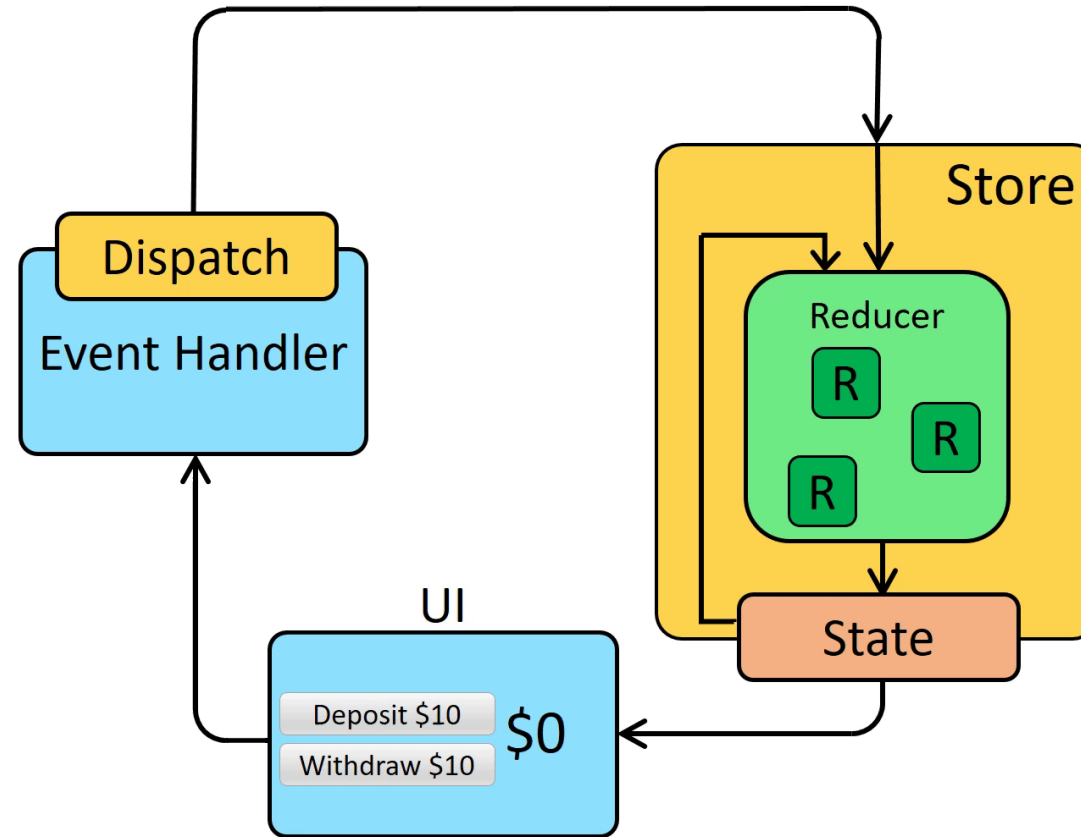
```
import create from 'zustand';
import { CardState } from '../types/card';
import { createJoke, getDays } from './jokes';

const today = '1';

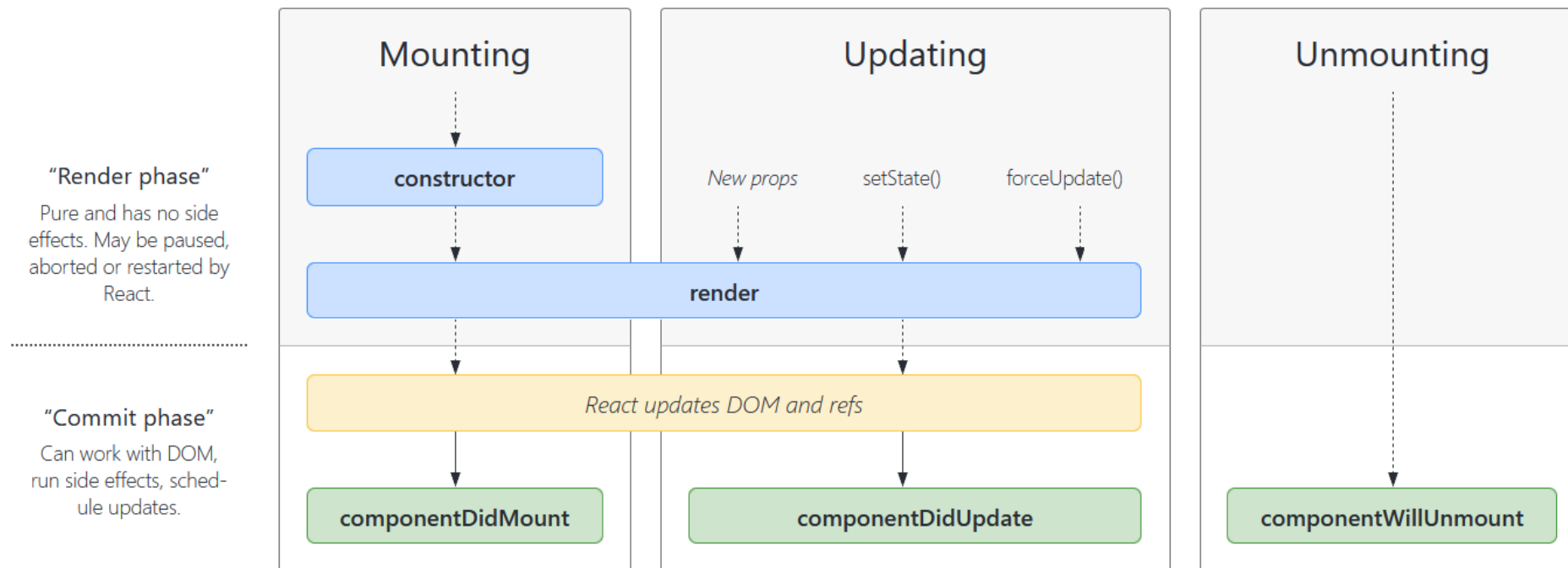
export const useCardStore = create<CardState>()((set) => ({
  cardsOfTheDay: createJoke(today),
  visibleCardIndex: 0,
  days: getDays(),
  nextCard: () =>
    set((state) => ({
      visibleCardIndex:
        state.visibleCardIndex < state.cardsOfTheDay.length - 1
          ? state.visibleCardIndex + 1
          : state.visibleCardIndex,
    })),
  anotherJoke: (index: string) =>
    set((state) => ({ cardsOfTheDay: createJoke(index), visibleCardIndex: 0 })),
})));
```

- Komponentin propsejen kautta komponentin vanhemmalta.
- Reactin state ja context.
- Erillisestä kirjastosta, joka pitää kirjaa koko appiksen tilasta.
 - **Redux**, recoil, zustand....
- useCardStore –hookit.

Redux-kirjaston kaavio tilanmuutoksista



Reactin renderöinti ei tarkoita, että selaimessa renderöitäisiin mitään!



- <https://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/>

Mitä muuta tuotanto-appiksessa olisi

- Perusrakenne on aika samantapainen, mutta koodia on paljon enemmän.
- Perusappiksessa on tyypillisesti ainakin myös:
 - Rajapintakutsuja, esim. axios (kts jokes.ts)
 - Router, jonka avulla navigoidaan appiksen eri sivuilla.
 - Eli käyttäjälle näyttää, että sijainti muuttuu. Käytännössä voi olla vain muutos siinä, mitä näytetään.
 - Kieleistykset
- Testit!
 - Yksikkötestit : Jest. Integraatiotestit (riippuu mitä niillä tarkoitetaan)
 - E2e-testit : Cypress, robot-framework