

Systemutveckling

Ramverk

25 HP

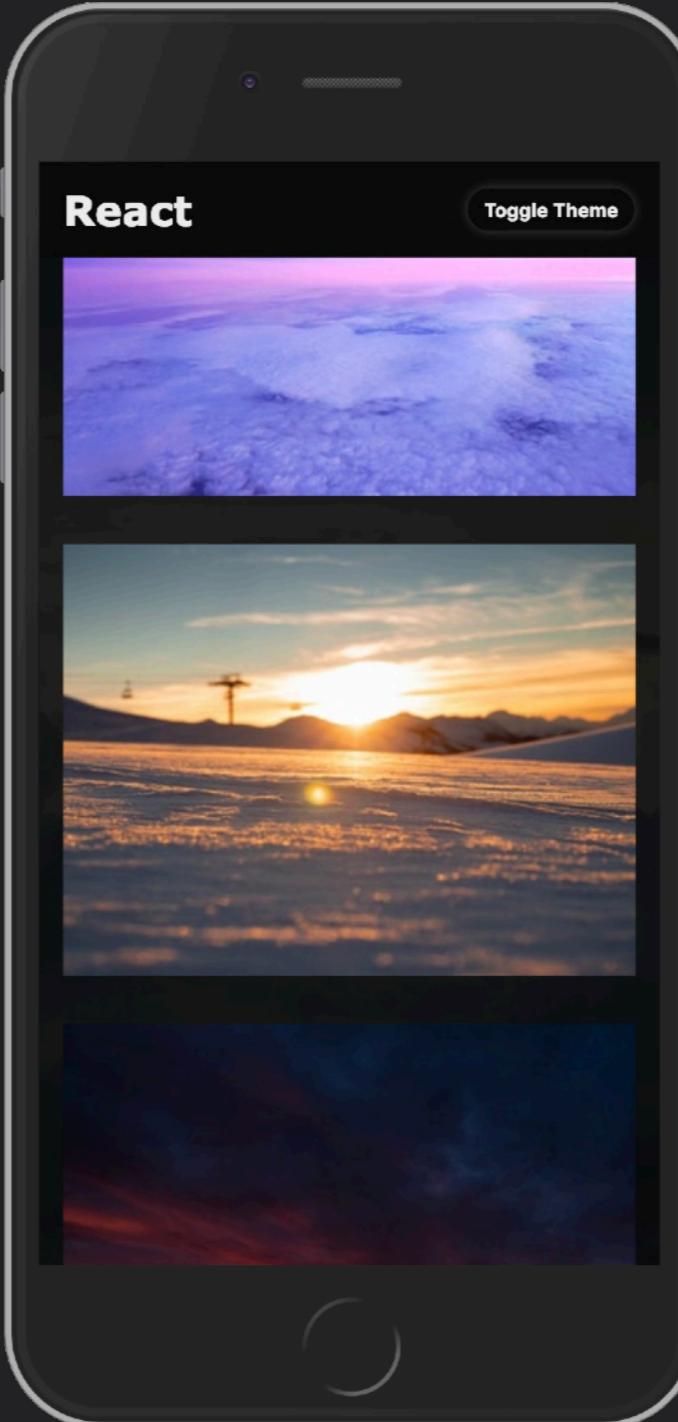
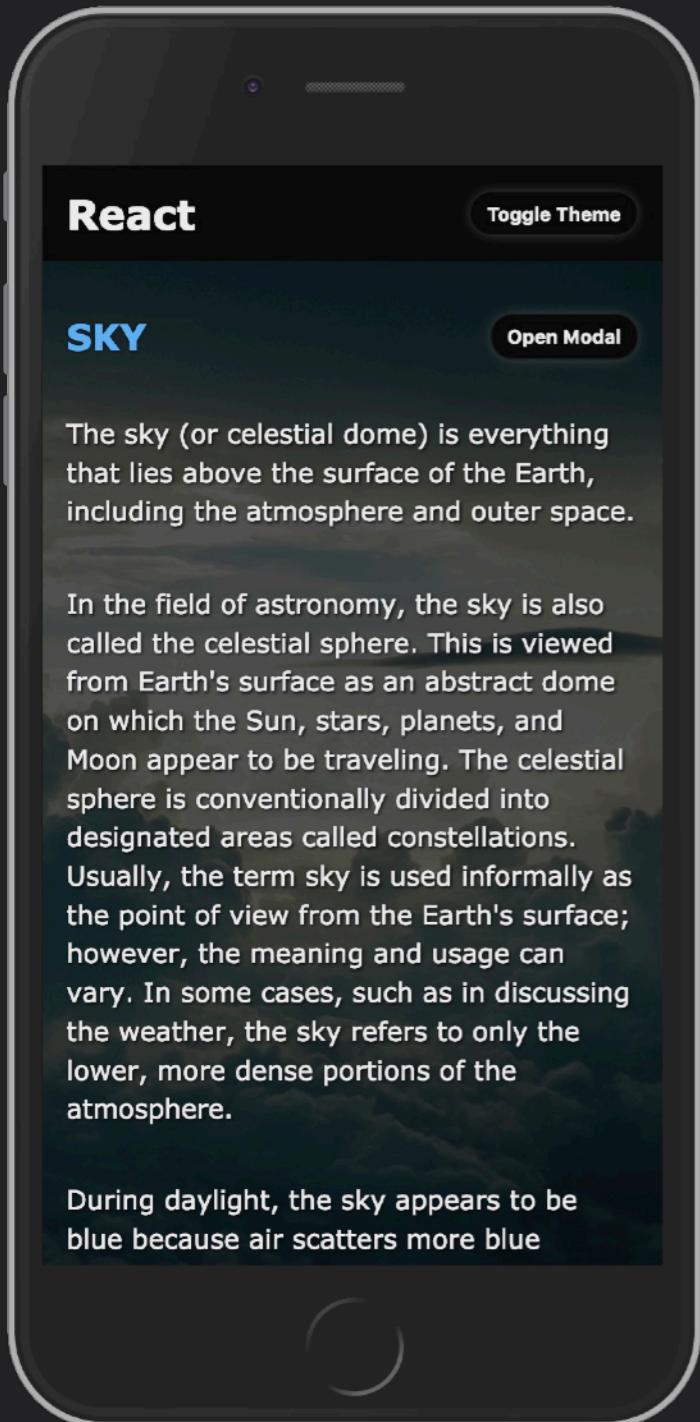
Kursupplägget

Föreläsning			Innehåll (FM)	Övningar (EM)	Inlämning
1	mån 25 mars	D	Introduktion Typescript (Basic Types, Type Inferrence, Variable Declarations, Iterators and Generators)	TypeScripts Hemsida TypeScript in 5 minutes + övning	
2	ons 27 mars	V	TypeScript forts. (Functions, Classes, Interfaces, Modules)		
3	fre 29 mars	V	Introduktion React (SPA, Virtuell DOM, Kap. 1-6)	Reacts Hemsida Main Concepts inklusive övningar i CodePen Kodövning (RP)	Inlämning 1 ges ut
4	mån 1 apr.	D	React Playground (1-initial-setup + 2-layout)		
5	ons 3 apr.	V	React forts. (Kap. 7-12)		
6	fre 5 apr.	V	Create React App TS Intro	Övning "Todo App"	Handledning
7	tis 9 apr.	D	React Playground (3-navigation-with-state) React Playground (4-navigation-with-routes)	Kodövning (RP)	
8	ons 10 apr.	V	React Playground (5-code-splitting) React Playground (6-error-boundary)	Kodövning (RP)	
9	fre 12 apr.	V	React Playground (7-portals)	Muntlig Presentation Kodövning (RP)	Inlämning 1 in Inlämning 2 ut
10	mån 15 apr.	D	React Playground (8-code-split-app-start)	Kodövning (RP)	
11	tis 16 apr.	D	React Playground (9-api-lib-axios)	Kodövning (RP)	Handledning
12	tis 23 apr.	D	React Playground (10-context)	Kodövning (RP)	
13	tors 25 apr.	V	Tentaplugg		Inlämning 2 lämnas in
14	fre 26 apr.	V	TENTAMEN		

React Playground

En app skapad med Typescript & React

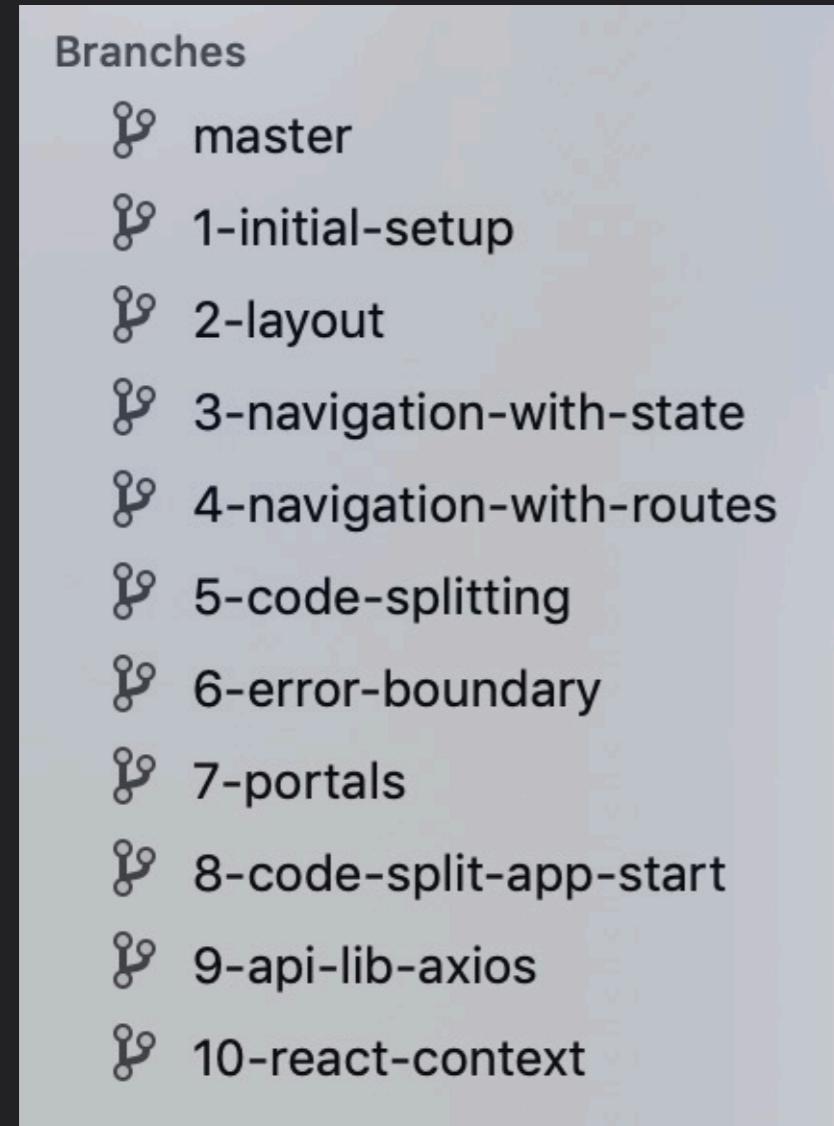
React Playground



Branches

- ⚡ master
- ⚡ 1-initial-setup
- ⚡ 2-layout
- ⚡ 3-navigation-with-state
- ⚡ 4-navigation-with-routes
- ⚡ 5-code-splitting
- ⚡ 6-error-boundary
- ⚡ 7-portals
- ⚡ 8-code-split-app-start
- ⚡ 9-api-lib-axios
- ⚡ 10-react-context

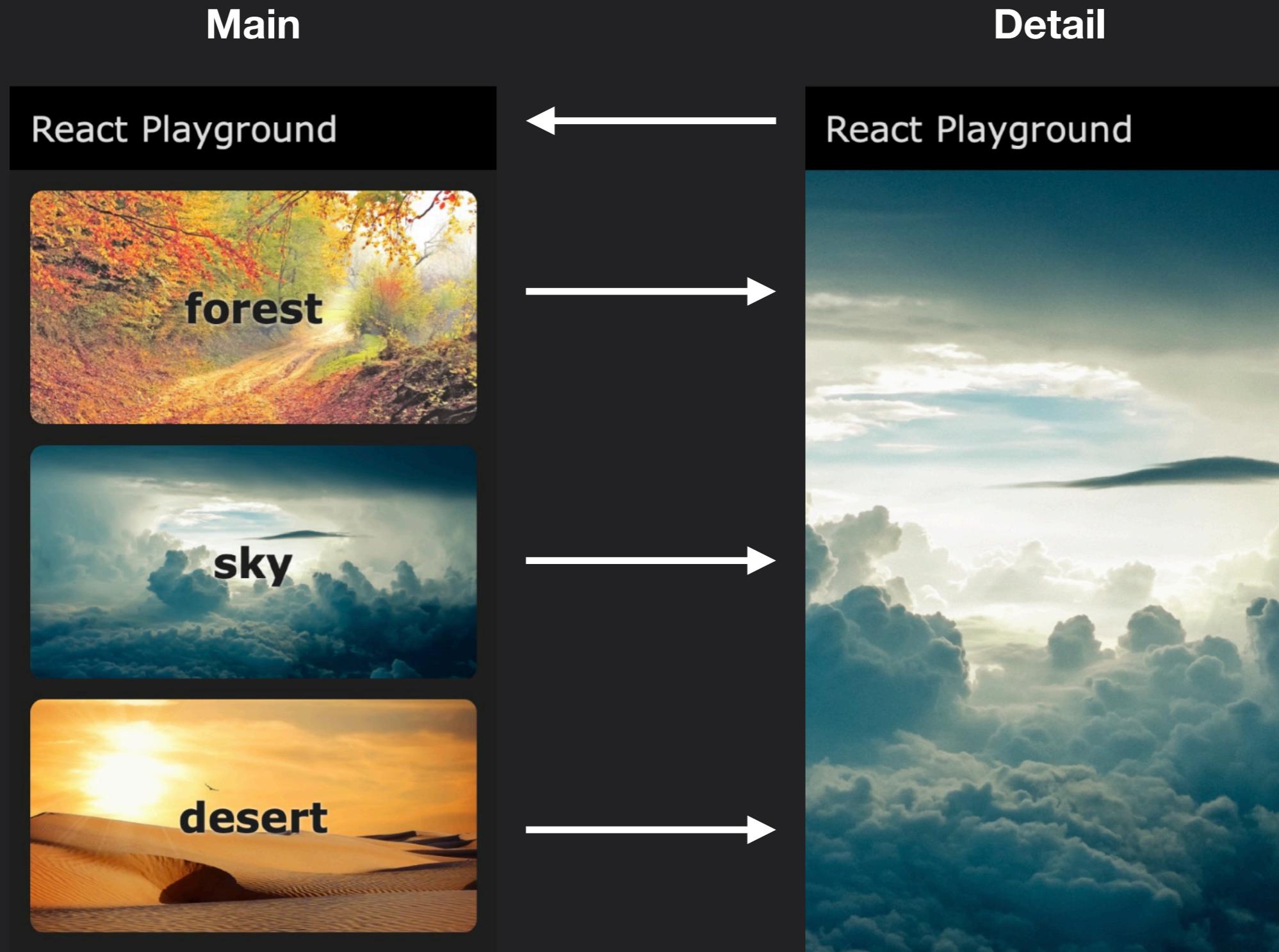
React Playground



Innan vi börjar

- 1. Starta det ni gjorde i förra veckan och se så att allt fungerar.**
 - **npm run app**
- 2. Hämta 2-layout koden från #slack och utgå ifrån den.**
 - **npm install**
 - **npm run app**

React Playground - Navigation



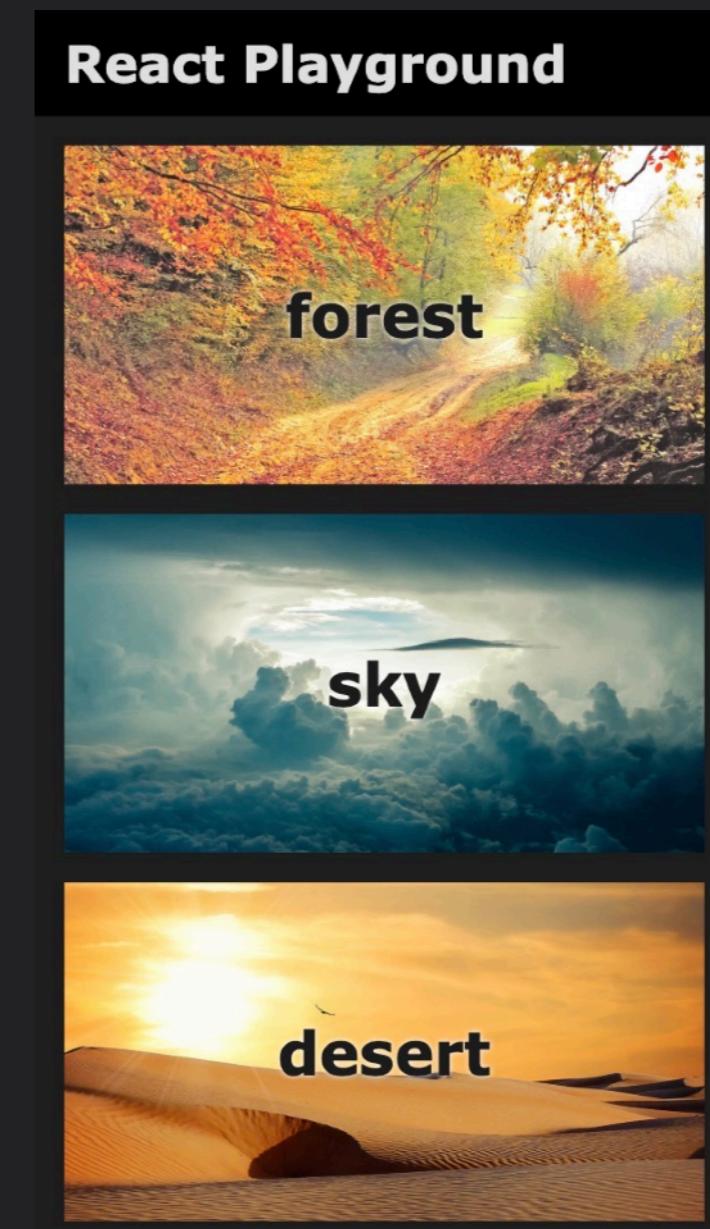
Navigation - State

Hur man navigerar på sidan med hjälp av state

Navigation - Using State

Bygga ut appens struktur

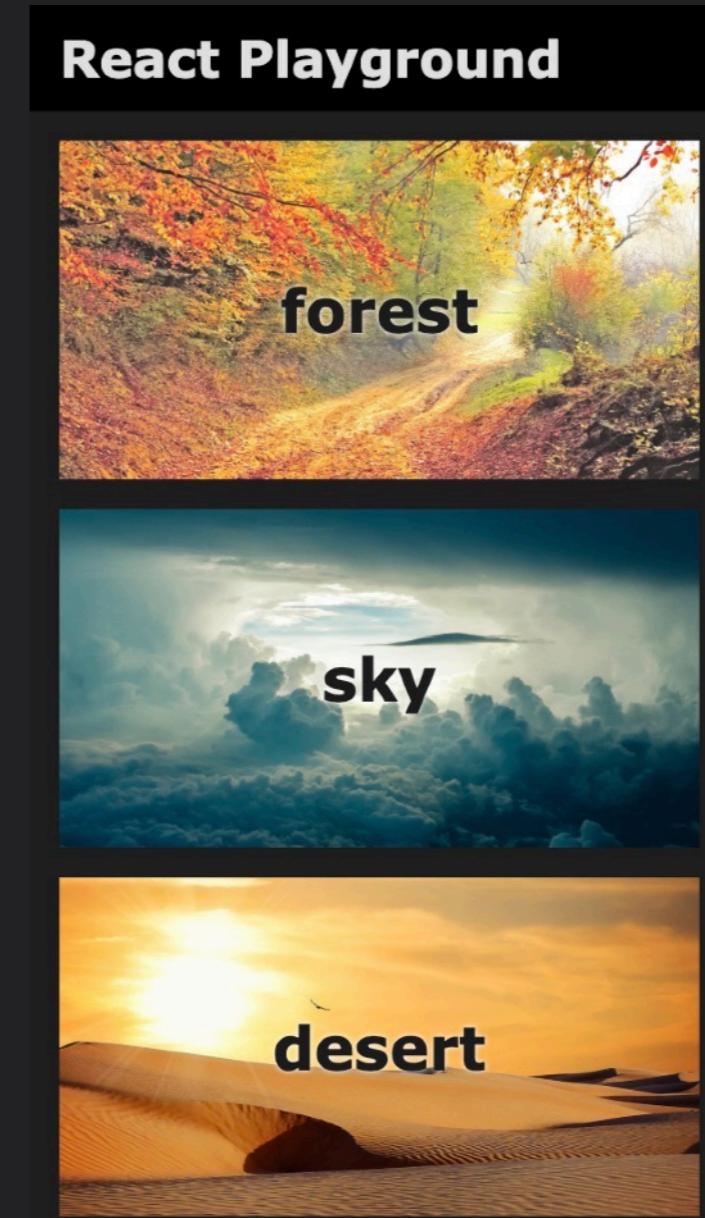
```
<App>  
  <Layout>  
    <Navbar/>  
    <Content>  
      <SectionItem/>  
      <SectionItem/>  
      <SectionItem/>  
    </Content>  
  </Layout>  
</App>
```



Navigation - Using State

Bygga ut appens struktur

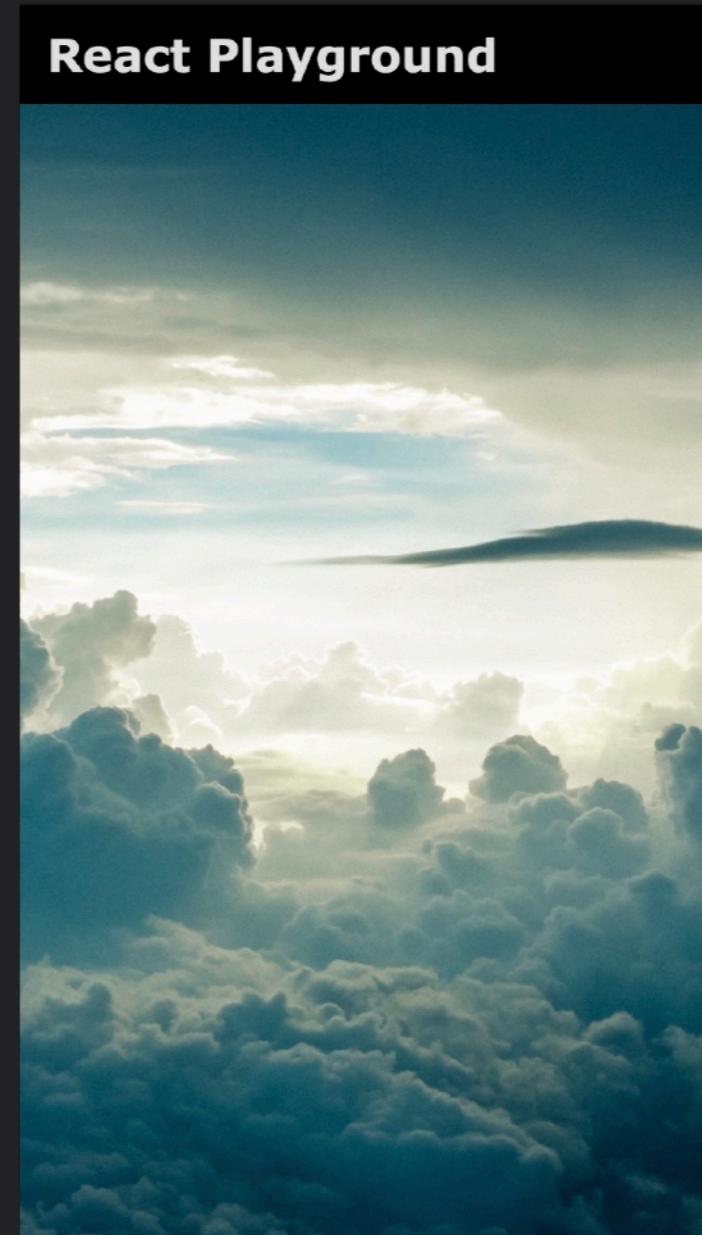
```
<App>  
  <Layout>  
    <Navbar/>  
    <ViewContainer>  
      <MainView>  
        <SectionItem/>  
      <MainView>  
        <DetailView/>  
    </ViewContainer>  
  </Layout>  
</App>
```



Navigation - Using State

Bygga ut appens struktur

```
<App>
  <Layout>
    <Navbar/>
    <ViewContainer>
      <MainView>
        <SectionItem/>
      <MainView>
        <DetailView/>
      </ViewContainer>
    </Layout>
  </App>
```



Navigation - Using State

Skapa statet

```
<App>  
  <Layout> ←  
    <Navbar/>  
    <ViewContainer>  
      <MainView>  
        <SectionItem/>  
      <MainView>  
        <DetailView/>  
    </ViewContainer>  
  </Layout>  
</App>
```

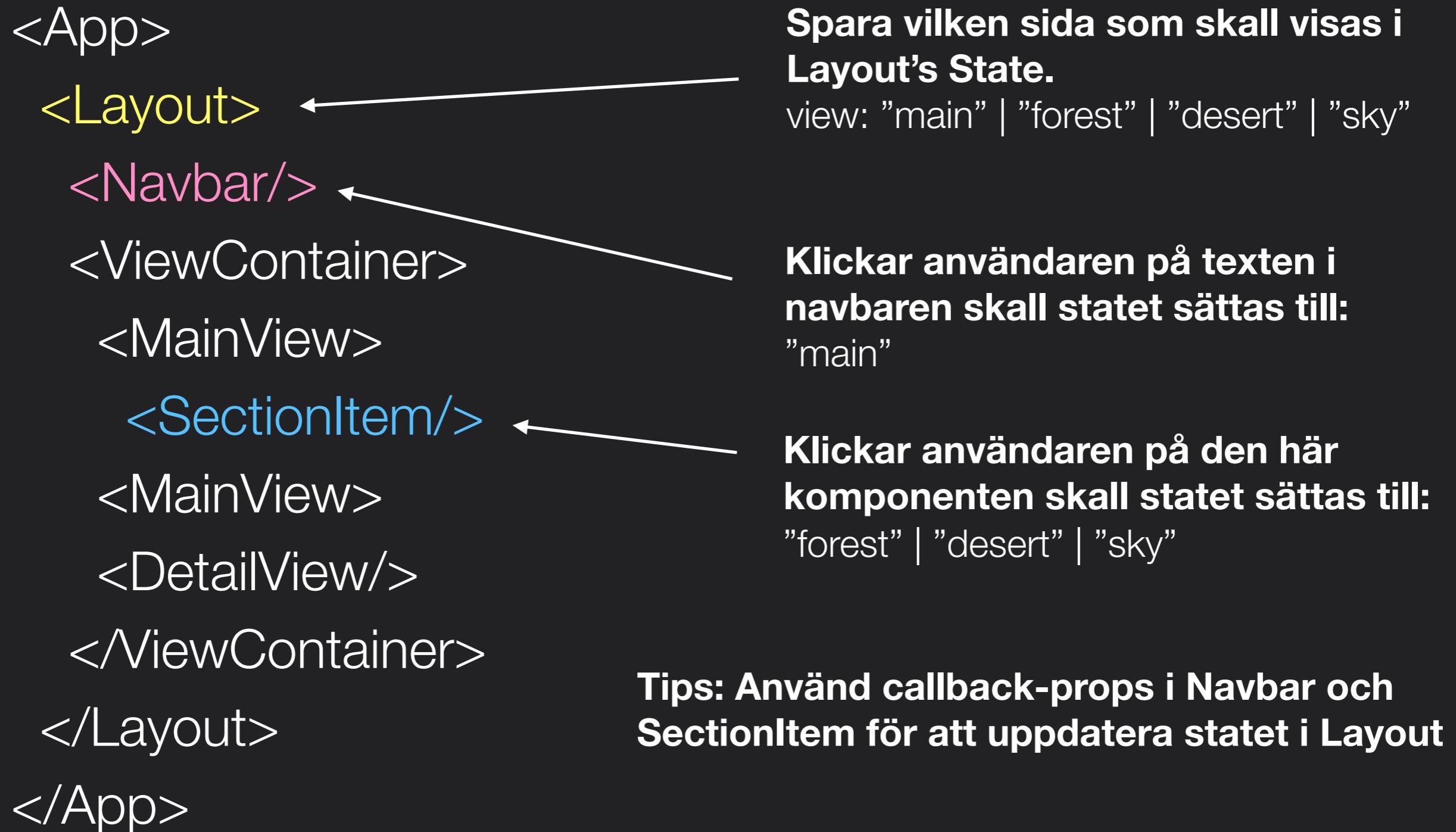
Spara vilken sida som skall visas i Layout's State.

view: "main" | "forest" | "desert" | "sky"

Varför i Layout och inte i ViewContainer?

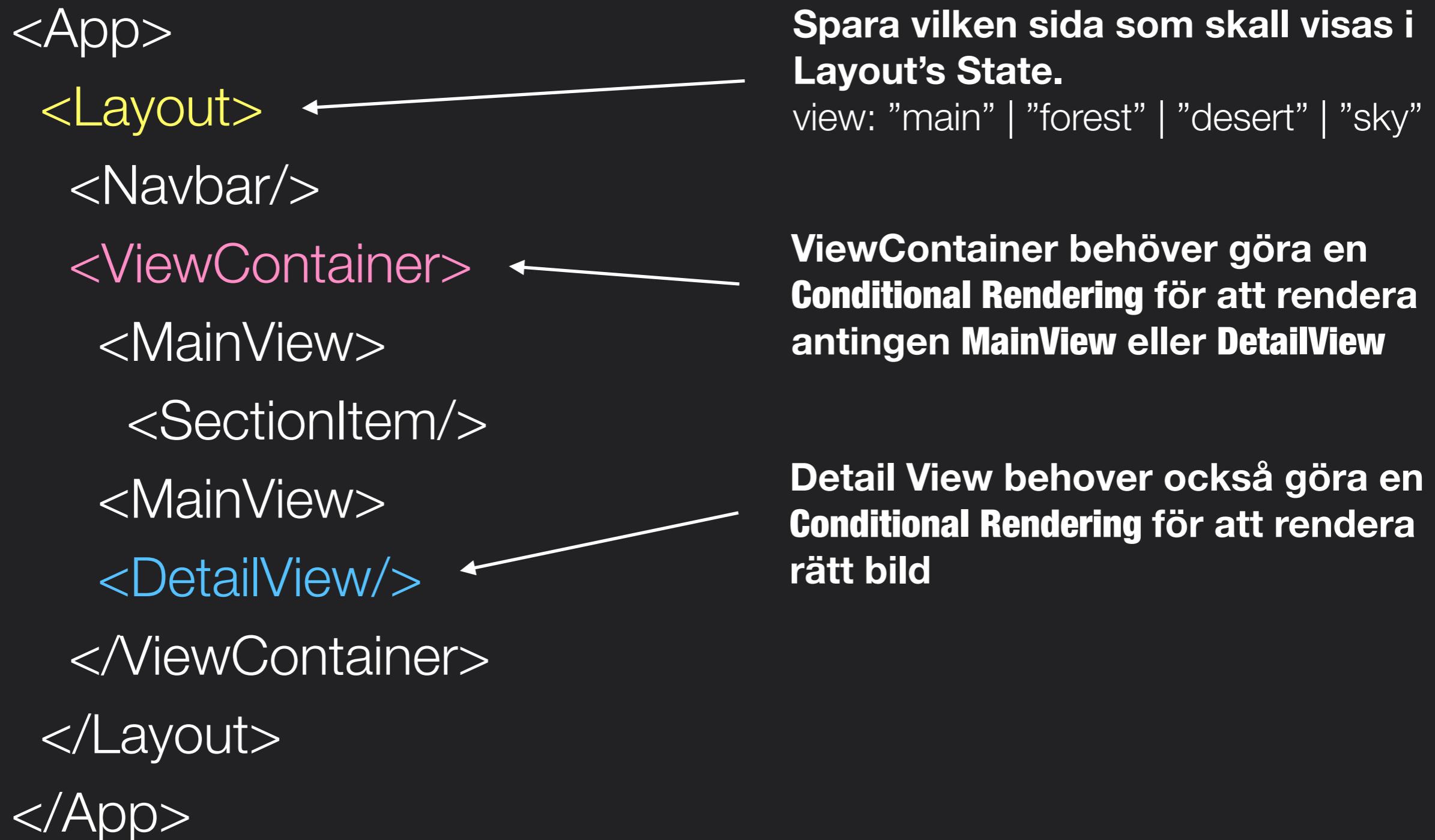
Navigation - Using State

Uppdatera statet



Navigation - Using State

Reagera på statet



Navigation - Routes

Hur man navigerar på sidan med hjälp av React Router

Navigation - React Router

React Router är ett JS lib som låter oss skapa sökvägar i vår react applikation. Sedan version 4 går det att skriva sökvägarna direkt i komponenterna - **Dynamic Routing**.

För att skapa en länkning använder vi komponenten **<Link to="/">** istället för html kodens ****.

En sökväg skapas med **Route** komponenten och ser ut så här: **<Route path="/" component={MainView}>**

Om en komponent behöver få tillgång till urlen, antingen den aktuella sökvägen eller parametrar, måste vi wrappa våran komponent i funktionen **withRouter**.

Navigation - React Router

För att förstå mer om hur React Router bör användas och hur man kan tänka kring dynamiska sökvägar, rekommenderas att ni läser lite kring deras filosofi:

<https://reacttraining.com/react-router/web/guides/philosophy>

Navigation - React Router

1. Installera react-router-dom

Det första ni behöver göra för att lägga navigera med routes är att installera **react-router-dom** via npm.

Kom även ihåg att installera **@types/react-router-dom**

För mer information följ länken nedan.

<https://reacttraining.com/react-router/web/guides/quick-start>

Navigation - React Router

2. Importera BrowserRouter

Sedan långt upp i vår appstruktur behöver läggat ill komponenten **BrowserRouter** som importeras ifrån react-router-dom.

Ett lämplig ställe för detta är i vår **<App>** komponent.

För mer information följ länken nedan.

<https://reacttraining.com/react-router/web/guides/quick-start>

Navigation - React Router

3. Uppdatera url via knapparna

Byt ut nuvarandra navigationselement i header texten samt SectionItems till **<Link>**'s och se så att urlen uppdateras.

För mer information följ länken nedan.

<https://reacttraining.com/react-router/web/guides/quick-start>

Navigation - React Router

4. Ersätt State med Routes

Ta bort koden som sätter statet i Layout och lägg istället till **<Route/>** komponenter i ViewContainer komponenten.

För mer information följ länken nedan.

<https://reacttraining.com/react-router/web/guides/quick-start>

Navigation - React Router

5. En sista förändring i package.json

```
"live-server": "live-server --entry-file=index.html --no-browser",
```

Så att live server kan uppdatera sidan när vi är inne på sub-URL'er.

OBS: Kom ihåg att starta om utvecklingsservern - **npm run app**

Läsanvisningar

React Router Web Guides

<https://reacttraining.com/react-router/>

React Docs

<https://reactjs.org/docs/getting-started.html>

TypeScript Handbook

<https://www.typescriptlang.org/docs/handbook>

Nästa lektion

React Playground
(Code Splitting & Error Boundary)

Tack