

Props & Children

Jens Palmqvist

Props

En React-komponent liknar mycket en vanlig funktion och på samma sätt som en funktion tar emot argument / parametrar kan man skicka data till en React-komponent med hjälp av props

Exempel - imageUrl som prop

```
//Komponentfunktionen tar emot props
```

```
const BildKomponent = (props) => <img src={props.imageUrl} />
```

```
<!-- Man skickar in props genom att lägga till attribut i komponenttaggen -->
```

```
<BildKomponent imageUrl="bild.png" />
```

Övning - Title med props

- Bygg om Title-komponenten så att den tar emot en prop som heter *text*
- Det som kommer in i text-propen ska sedan skrivas ut i H1:an

Implementation

```
<Title text="Kaffeappen V2">
```

Resulterad HTML

```
<h1>Kaffeappen V2</h1>
```

Olika typer av props

- Props kan vara av olika typer, t.ex.
 - Strängar
 - Nummer
 - Objekt
 - Arrayer
 - Funktioner

Exempel - flera props av olika typer

```
const KomponentMedOlikaProps = (props) => {  
  console.log(props.enText, props.ettNummer, props.enLista, props.ettObjekt);  
  props.enFunktion(); //Funktionen kan man anropa  
  return (<p>Jag har många props men visar inte upp nån av dem</p>)  
}
```

```
<KomponentMedOlikaProps  
  enText="Hej på dig"  
  ettNummer={3}  
  enLista=[1, 3, 4]  
  ettObjekt={{ foo: "bar" }} //Obs! Dubbla hakparanteser  
  enFunktion={() => console.log("Jag skickades in i den här komponenten!")} />
```

Övning - bryt ut props

```
const BildText = () => {  
  const animalType = "fågel";  
  const speed = 20; //kilometer i timmen;  
  
  return (<p>Bilderna visar en {animalType}.  
    Den kan flyga i {speed} km/h vilket  
    motsvarar {speed / 3.6} m/s.</p>);  
}
```

- Utgå ifrån koden ovan och bryt ut props så att komponenten kan användas enligt nedan

```
<BildText animalType="katt" speed={30} />
```

Deconstructa props-objektet

- För att det ska bli tydligare vilka props en komponent tar emot brukar man deconstructa propsobjektet direkt i parameterlistan

Deconstructa props - exempel

```
//Utan destructing  
const KomponentMedOlikaProps = (props) => {  
  console.log(props.enText, props.ettNummer, props.enLista, props.ettObjekt);  
  props.enFunktion(); //Funktionen kan man anropa  
  return (<p>Jag har många props men visar inte upp någon av dem</p>)  
}
```

```
//Med destructing  
const KomponentMedOlikaProps = ({ enText, ettNummer, enLista, ettObjekt, enFunktion }) => {  
  console.log(enText, ettNummer, enLista, ettObjekt);  
  enFunktion(); //Funktionen kan man anropa  
  return (<p>Jag har många props men visar inte upp någon av dem</p>)  
}
```

Övning - destructa props

- Utgå ifrån BildText-komponenten från förra övningen
- Destrukta props-objektet direkt i parameterlistan
- Uppdatera alla ställen där du använder props.^{***} i komponenten

Children

- Det finns en speciell prop som heter *children*
- Den innehåller automatiskt allt som är mellan taggarna när använder en komponent

```
const EnKomponentMedChildren = (props) => <p>{props.children}</p>
```

```
<EnKomponentMedChildren>Allt som står här kommer hamna i  
  props.children i komponentens kod</EnKomponentMedChildren>
```

Children, forts.

- Children kan också innehålla andra element och komponenter:

```
const MinSuperLista = ({ children }) => (  
  <ul className="superStyling">  
    { children }  
  </ul>  
)
```

```
<MinSuperLista>  
  <li>En rad</li>  
  <li>Två rader</li>  
  <li>Tre rader</li>  
</MinSuperLista>
```

Övning - Title med children

- Gör om Title-komponenten så att texten kommer in som children istället för en prop

Implementation

```
<Title>Kaffeappen V3</Title>
```

Resulterad HTML

```
<h1>Kaffeappen V3</h1>
```