PROPS

Conceptually, components are like JavaScript functions. They accept inputs (called "props") and return React elements describing what should appear on the screen.

Alla komponenter har tillgång till `props`

`props` kommer att bli din bästa vän

Alla värden som skickas ner blir till props

```
this.props
```

```
//App.js
render(){
  return(
    <div>
      <Header propName="propValue" />
    </div>
  );
}
```

Liknande som att man sätter ett attribut på ett HTML-element

Här skickar vi vidare värdet `propValue` för att användas i Header

Det vi döper vår prop till är vad den kommer heta i vår komponent

```
//Header.js
render(){
  return(
      <h1> { this.props.propName === "propValue" } <h1/>
  );
}
```

Blir tillgängligt via `this.props`

Spread props

Vi kan skicka ner värdena en för sig:

Vi kan även använda JSX-spread

Fungerar "typ" som ES6-spread

```
render(){
  let obj = {
    prop1: "value1",
    prop2: "value2"
  }
  return(
    <div>
      <List {...obj} />
    </div>
  );
```

```
render(){
  return(
    <div>
      <List prop1="value1" prop2="value" />
    </div>
  );
}
```

Conditional Rendering

Oftast vill man rendera olika innehåll baserat på olika värden

Hur? `IF/ELSE`

Innan `return` kan vi göra beräkningar

```
render(){
  if(???){
    //Do stuff, return element?
  }
  return(
    <div>


    </div>
  );
}
```

```
render(){
  if(false){
    return null
  }else{
    return(
      <div>
        Hello!
      </div>
    );
  }
```

Ternary Operator

```
render(){
  let text = this.props.text ? this.props.text : 'Default'
  return(
    <div>
      { text }
    </div>
  );
}
```

Användbara grejer

ES6 Destructuring

ES6 Spread and Rest syntax

Ternary Operator