Props

Ustun Ozgur

2015-06-13 Cumartesi

Props

- properties (özellikler) için kısaltma
- Hem div, p gibi hazır bileşenler, hem de MerhabaDünya gibi özel bileşenler, props alabilir
- Key-value (anahtar-değer) özellikleri
- Eşittir işareti
- MerhabaDunya'ya `{ad: "ustun"}` geçirmek için
- `<MerhabaDunya ad="Ustun"/>`

Props

- Render metodunda ve diğer yaşam döngüsü metodlarında erişilebilir
- `this.props` olarak erişilir
- `ad` alanına ulaşmak için: `this.props.ad`

```
var MerhabaDunya = React.createClass({
  render: function () {
    return <div>{"Merhaba " + this.props.ad}</div>;
  }});
```

Daha Karmaşık Bir Örnek

- Hedef: Birden fazla insana merhaba de
- İnsanlar dizisi: `["Ali", "Ahmet", "Ayse"]`

```
var Insanlar = React.createClass({
  render: function () {
    var insanlar = ["Ali", "Ahmet", "Ayse"];
    var icerik = [];

    for (var i = 0; i < insanlar.length; i++) {
      icerik.push(<MerhabaDunya ad={insanlar[i]}/>);
    }
    return <div>{içerik}</div>;
}});
```

Daha Karmaşık Bir Örnek

- *Key* özelliği
- Her çocuk elemanı eşsiz yapmak için
- Performans artışı

```
var Insanlar = React.createClass({
  render: function () {
    var insanlar = ["Ali", "Ahmet", "Ayse"];
    var icerik = [];

    for (var i = 0; i < insanlar.length; i++) {
      icerik.push(<MerhabaDunya key={insanlar[ı]} ad={insanl
    }
    return <div>{icerik}</div>;
}});
```

Daha Karmaşık Bir Örnek: Map Kullanarak

```
var Insanlar = React.createClass({
  render: function () {
    var insanlar = ["Ali", "Ayşe", "Ahmet"];

    var icerik = insanlar.map(function (person) {
      return <MerhabaDunya key={person} ad={person}/>;
    });

    return <div>{içerik}</div>;
}});
```

Devam

- Ara değişkenlere ihtiyacımız yok
- Doğrudan bir JSX ifadesi koyabiliriz
- Key olarak indeks sayısı kullanabiliriz.

```
var Insanlar = React.createClass({
  render: function () {
    var insanlar = ["Ali", "Ayse", "Ahmet"];

    return <div>
      {insanlar.map(function (insan, i) {
        return <MerhabaDunya key={i} ad={insan}/>; })}
    </div>;
}});
```

Alıştırma:

- Todo uygulamasını, yapılacaklar listesi bir string listesi olarak, props şeklinde girildiği şekilde degistirin.
- Örnek: ToDos bileşenine şu şekilde bir liste geçirilmelidir:
- `["Domates al", "Patates al"]`
- Listeyi değiştirin, HTML çıktısının değiştiğini görüntüleyin.
- Render metodunu değiştirerek todo listesinin alfabetik sıralı halini görüntüleyin.
- Listenin en altında aynı listenin orijinal sırası olacak şekilde bir değişiklik yapın.

Olaylar (Events) ve Fonksiyonların Props olarak kullanımı

- Herhangi bir JavaScript ifadesi props olarak kullanılabilir.
- JS'te fonksiyonlar birinci sınıf değerlerdir: Fonksiyonları props olarak geçirebiliriz.
- Olay yönetimi için onClick, onBlur gibi adlarda props'larımız vardır.
- Bu propslar için fonksiyon geçirin.

Bir onClick fonksiyonu bağlamak

```
var MerhabaDunya = React.createClass({
  onClick: function () {
    console.log("Merhaba " + this.props.ad);
  },
  render: function () {
    return <div onClick={this.onClick}>
      {"Merhaba " + this.props.ad}
    </div>;
  }
});
```

İnsan Listesi

- İnsan listesi ve MerhabaDunya çocuk bileşenleri
- event handler'ın İnsanlar bileşeninde olduğunu varsayın, MerhabaDünya'da değil
- MerhabaDunya bileşenine bir onClick özelliği geçirilecek
- Tepeden ne inerse o çalıştırılacak

```
var MerhabaDunya = React.createClass({
  render: function () {
    return <div onClick={this.props.onClick}>
      {"Merhaba " + this.props.ad}
    </div>;
  }
});
```

Karmaşıklık Ana Bileşende

```
var Insanlar = React.createClass({
  onClick: function (ad) {
    console.log("Merhaba " + ad);
  },
  render: function () {
    var people = ["Ali", "Ahmet", "Ayse"];

    return <div>
      {insanlar.map(function (person, i) {
        return <MerhabaDunya
        key={i}
        onClick={this.onClick.bind(this, insan)}
        ad={insan}/>;
      }.bind(this))}
    </div>;
  }});
```

Ne Değişti?

- onClick handler'ı ad parametresi alacak şekilde değiştirdik.

```
onClick: function (ad) {
```

- MerhabaDunya'ya geçirilen her onClick handler'ı, ad parametresini şu anki insan adına bind edecek (bağlamak) şekilde değiştirdik

```
onClick={this.onClick.bind(this, insan)}
```

Bind metodu

- ES5 ile geldi
- Bind = Bağlamak
- "creates a new function that, when called, has its this keyword set to the provided value, with a given sequence of arguments preceding any provided when the new function is called."
- İlk amaç: `this` değerini bağlamak
- İkinci amaç: argümanları bağlayıp partial (yarım) bir fonksiyon oluşturmak

Örnek: 5 ile Toplama Yapan Fonksiyon

```
function add(a, b) { return a + b; }
```

- a'yı 5'e sabitlemek istiyoruz.
- `this`'in değeri önemli değil, çünkü kullanılmıyor
- `this` değerini null'a eşitle

```
var add5 = add.bind(null, 5)
console.log(add5(3)); // 8
```

Alıştırma:

- `console.log` metodunu `l` adında bir fonksiyona nasıl eşitlersiniz?
- Basit bir `var l = console.log` çözümü doğru mu? Doğru çözüm ne? solution?

Önceki örneğe tekrar bakış

```
var Insanlar = React.createClass({
  onClick: function (ad) {
    console.log("Merhaba " + ad);
  },
  render: function () {
    var insanlar = ["Ali", "Ahmet", "Mehmet"];
    return <div>
      {insanlar.map(function (insan, i) {
        return <MerhabaDunya
        key={i}
        onClick={this.onClick.bind(this, insan)}
        ad={insan}/>;
      }.bind(this))}
    </div>;
  }});
```

Daha Uzun Hali

```
var Insanlar = React.createClass({
  onClick: function (ad) {
    console.log("Merhaba " + ad);
  },
  render: function () {
    var people = ["Ali", "Ahmet", "Mehmet"];

    return <div>
    {insanlar.map(function (insan, i) {
      var boundFunction = this.onClick.bind(this, insan);
      return <MerhabaDunya
      onClick={boundFunction}
       key={i}
      ad={insan}/>;
    }.bind(this))}
    </div>;
```

Diğer Alternatif

- Şu anki `this` değerini başka bir değişkende sakla, örneğin `that`

```
var Insanlar = React.createClass({
  onClick: function (ad) {
    console.log("Merhaba " + ad);
  },
  render: function () {
    var people = ["Ali", "Ayse", "Ahmet"];
    var that = this;
    return <div>
      {insanlar.map(function (insan, i) {
        return <MerhabaDunya
        key={i}
        onClick={that.onClick.bind(that, insan)}
        ad={insan}/>; })}
    </div>;
  }});
```

_.partial kullanarak alternatif

```
var Insanlar = React.createClass({
  onClick: function (name) {
    console.log("Merhaba " + ad);
  },
  render: function () {
    var people = ["Ali", "Ayse", "Ahmet"];
    var that = this;
    return <div>
      {insanlar.map(function (ad) {
        return <MerhabaDunya
        onClick={_.partial(that.onClick, insan)}
        ad={insan}/>; })}
    </div>;
  }});
```

getDefaultProps

- Render gibi bir bileşen metodu
- Varsayılan özellik değerleri

Örnek: Varsayılan Selamlı MerhabaDunya Bileşeni

```
var MerhabaDunya = React.createClass({
  getDefaultProps: function () {
    return {selam: 'Merhaba'}
  },
  render: function () {
    return <div>
      {this.props.selam} {this.props.ad}
    </div>;
  }
});

var Insanlar = React.createClass({
  render: function () {
    return <div>
      <MerhabaDunya ad="Ali"/>
      <MerhabaDunya selam="Hello" ad="John"/>
```