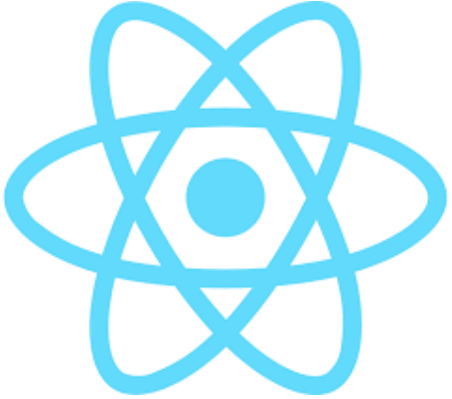


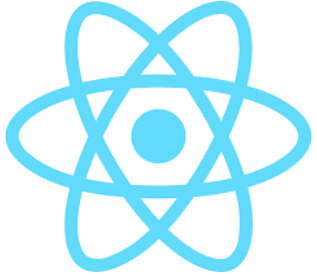
State (Durum) Yapısı

Onur KULABAŞ

onurkulabas@yahoo.com

React Training

Istanbul, 2022



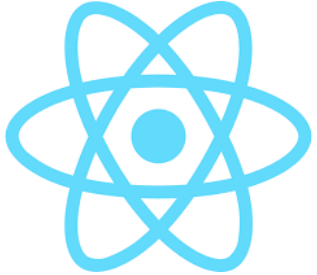
useState Kullanımı

React içinde kullanılacak olan değişken değerleri, yapılacak olan işlemler veya çağrılacak olan servisler için State mekanizması dikkate alınmalıdır.

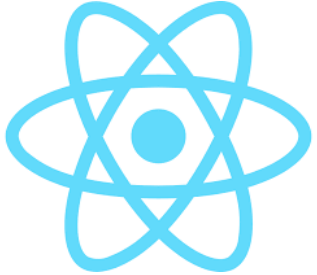
Öncelikle aşağıdaki satır ilgili sayfanın en üstüne eklenmeli ve **useState hook**'u kullanılmalıdır.

```
import React, {useEffect,useState} from 'react';
```

Hooks



React'ın kendi dokümanında **Hooklar** için şu tanım yer almaktadır; *Hooklar React 16.8'de yeni bir ektir.* ***Bir sınıf yazmadan*** *state ve diğer React özelliklerini kullanmanıza izin verir.(React.js Docs).* Hook yapısı class componentleri yerine JavaScript'deki **fonksiyonel** yapıyı kullanmamızı sağladı. (Facebook class componentlerin de kullanımda olacağını ve terkedilmeyeceğini belirtiyor)



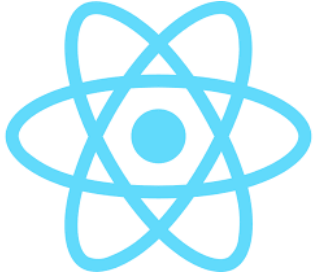
Hooks

Bir çok hook çeşidi bulunmakla beraber en çok kullanılanlar şunlardır;

- useState
- useEffect
- useContext
- useRef
- useCallback
- useReducer
- useDebugValue
- useLayoutEffect

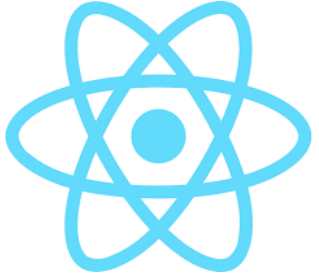
Bunların dışında ihtiyacımıza göre kendi hook'larımızı da yazabileceğiz.

Değişkene Değer Atamak



```
const [degisken1, setDegisken1] = useState("test123");
const [result, setResult] = useState();
setResult('Success');
```

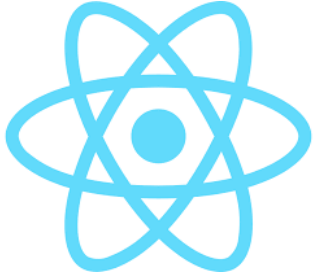
useState hook'u değişkenleri global bir şekilde tutar, bu nedenle kaybolmazlar. Yukarıdaki örnekte degisken1 adlı değişkenin değerini test123 olarak verdik. Result adlı değişkenin ise ilk değeri verilmemiştir. Buna karşın **setResult** adlı bileşenin **result** değişkeninin durumunu yani değerini **değiştirebilmesi** sağlanmıştır. En alt satırda `setResult('Success');` kullanımı ile result değerini Success olarak vermiş olduk.



async - await

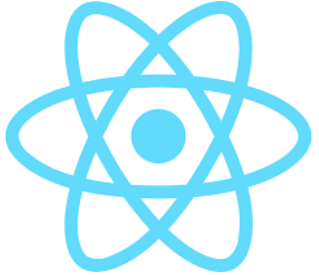
React içindeki node paketleri varsayılan olarak kod içindeki tüm satırları ve komutları **paralel şekilde** çalıştırmak üzere kurgulandığı için alışageldik şeklin dışında sonuçlar alınması olasıdır. Normal koşullarda sırayla çalışmasını beklediğimiz satırlar paralel olarak çalışınca mantıksal kurguda bozulmalar meydana gelebilmektedir. Paralel çalışmanın da gerekli olabileceği durumlar olmakla beraber standart kullanım için **fonksiyonlarda async** anahtar kelimesi kullanılarak paralel yerine seri çalıştırma sağlanmalı ve **service call'larda await** anahtar kelimesi de yerleştirilmelidir. Eğer fonksiyon **async** yani asenkron **değilse await** anahtar kelimesi de **kullanılamaz**.

async – await Örnek Kullanım



```
async function getServiceName()
{
  const response = await axios.get('service_name');
```

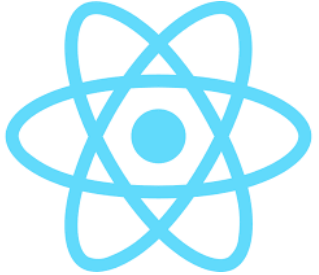
Yukarıdaki örnekte function'ın başına async eklenerek paralel yerine seri çalışması sağlanmıştır. Axios.get satırı önünde yer alan await komutu da işlemin function içinde ilgili sıra geldiğinde yapılmasını sağlamıştır. Axios ile ilgili detaylı bilgi 11. modülde verilecektir.



useEffect

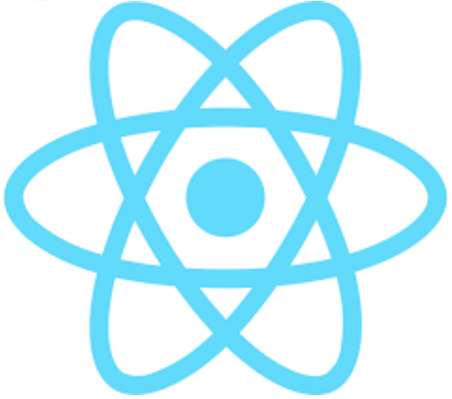
Daha önce belirttiğimiz üzere Hook'lar, React state ve yaşam döngüsü özelliklerine fonksiyonel bileşenleri kullanarak bağlamaya yarayan fonksiyonlardır. **useEffect**, React class'larındaki **componentDidMount**, **componentDidUpdate**, ve **componentWillUnmount** ile aynı işleve sahiptir fakat tek bir API içerisinde birleştirilmiştir. Yeni bir bileşen eklenmesi ve DOM'a **eklenmesi** sürecinde **componentDidMount**, component'in **güncellenmesi** sürecinde **componentDidUpdate** ve en son aşamada component DOM'dan **çıkarıldığında componentWillUnmount** metodları çalışmaktaydı. Bu sürece **Life Cycle (Yaşam Döngüsü)** adı verilmekteydi. Şimdi bunları useEffect ile nasıl yapacağımızı inceleyelim.

useEffect



```
useEffect
(
  () =>
  {
        console.log("Merhaba, useEffect sayfa yüklendiği anda çalışıyor!");
  },
  []
);
```

Bu örnekte useÊffect'i **2. parametre** olarak **boş bir array** kullanım şekli **[]** kullanmak suretiyle component render olduktan hemen sonra çalışacak şekilde (**componentDidMount** gibi) yazdık Component yüklendiği anda yapılması gereken kontroller (örneğin, kullanıcı oturum açmış mı, açmamışsa sayfayı login'e yönlendir veya sayfa yüklendiği anda bir servis veya array'den gelen bilgileri oku ve ekrana bas gibi..) bu sırada yapılmalıdır. İkinci parametreyi yani **boş array kısmını yazmasaydık**, state değişimlerinde (**componentDidUpdate** gibi) çalışacaktı.



Teşekkürler ☺

Onur KULABAŞ

onurkulabas@yahoo.com

http://tr.linkedin.com/in/onurkulabas