Life Cycle Methods

```
// 1. Life Cycle Methods
  // 1. constructor()
  // Oluşturucu, yerel uygulamaya tepki vermeye başladığında ilk kez her zaman
çağrılır, çoğunlukla yerel uygulama sınıfı tepkilerinde Eyaletler oluşturmak için
kullanılır.
  constructor(){
    super();
    console.log("Constructor Called.");
  }

  // 2. componentWillMount()
  // Bu işlev, constructor () çağrıldıktan hemen sonra çağrılır. Bu, çoğunlukla
yerel uygulamalara tepki vermekten kaynaklanan eşzamansız işlevleri veya web
çağrılarını çağırmak için kullanılır.
  componentWillMount(){
    console.log("ComponentWillMount() Called.");
  }

  // 3. render()
  // Render işlevi, bir sınıfa ait en önemli işlevlerden biridir; çünkü bu, View
veya herhangi bir grafik temsilini döndürme bloğunu kullanarak ekranda oluşturmak
için kullanılır.
  // Sınıfımızda render işlevi kullanılmadan, tepki veren yerel uygulamalarda
kararlı Görünümler veya Ekranlar oluşturamayız.
  render()
  {
    console.log("Render Called");
    return(
      <View style = { styles.MainContainer }>
        <Text style={{fontSize: 20}}>React.Component</Text>
      </View>
    );
  }


  // 4. componentDidMount()
  // çağrıldıktan sonra kendisini çağrılan componentDidMount işlevi, uygulama
başladığında ilk kez JSON verilerini ayrıştırmak için Web Çağrısı'nı çağırmak için
kullanılır.
  componentDidMount(){
    console.log("ComponentDidMount() Called.");
  }



  // 2. Updating methods:
  // Güncelleme yöntemleri, doğal olarak tepki veren Sahne veya Durum değerini
güncellemek veya değiştirmek için kullanılır.
```

```
  // Önceden oluşturulmuş bir bileşen yeniden oluşturulduğunda bu bileşenler
otomatik olarak çağrılır.

  // 1. componentWillReceiveProps()
  // Bu fonksiyon, bileşen dozumuzdan önce yeni sahne ile bir şey çağrılabilirdi,
Bir sonraki propayı argüman olarak gönderirdik.
  componentWillReceiveProps(nextProps) {
    this.setState({
      value: nextProps.myProp + "hello"
    });
  }

  // 2. shouldComponentUpdate()
  // ShouldComponentUpdate () işlevi, ekran veya üst bileşenin yeniden oluşturulma
işleminden önce her defasında çağrılır.
  // Bu fonksiyonda false ileterek yeniden görüntülemeyi durdurabilirsiniz
  shouldComponentUpdate(nextProp, nextState) {
    console.log(nextProp, nextState);
    console.log(this.props, this.state);
    return false;
  }

  // 3. componentWillUpdate()
  // Bu işlev, yeniden oluşturma işleminden önce ve güncelleme için yeni durum veya
sahne alındığında ve doz, this.setState ({}) yöntemine izin verilmediğinde
çağrılır.
  componentWillUpdate(nextProp, nextState) {
    console.log('componentWillUpdate Called', nextProp, nextState);
  }

  // 4. componentWillUnmount() -> Öldürme Yöntemleri
  // Bu işlev, bileşen DOM kaldırıldıktan veya yok edildikten hemen sonra çağrılır.
Kullanıcılar, çalışan tüm zamanlayıcıları temizleyebilir, ağ isteklerini
durdurabilir ve uygulamada önceden depolanmış olan değeri temizleyebilir.
  componentWillUnmount() {
    this.value= this.value.destroy();
  }

  // 4. Error handling method : Hata mesajları alma
  // ComponentDidCatch () yöntemi, hata işleme yönteminin bir parçasıdır.
  // JavaScript kodu arasında hata bulmak ve onlarla doğru mesaj veya argüman
ileterek bunları işlemek için kullanılır.
  //Herhangi bir hatayı kullanmamıza ya da herhangi bir hatayı işlemek için bir
yöntem kullanmamıza yardımcı olacaktır.
  componentDidCatch(error, info){
    //Handle error.
  }
```