

NodeJS ile Web Programlama Hafta-3 Rapor

Bu dersimizde 2. hafta yaptığımız kod üzerinden ilerledik. Not ekleme, silme gibi işlemleri başka dosya ile birleştirerek kullandık.

notes.js adlı dosyada ilk olarak fonksiyonlar oluşturduk. İlk fonksiyonumuz addNote adında notları dosyaya kaydedeceğimiz bir fonksiyondur. Fonksiyonumuz parametre olarak da title ve body alıyordu. Fonksiyon için export etme işlemini yaptık. Önceden export formunu şu şekilde kullanırdık :

```
76 module.exports = addNote;
```

Fakat bizim birden fazla export edecek fonksiyonumuz olma durumunda teker teker yazmak yerine farklı bir kullanımımız daha var. Onu da getNotes ve addNotes fonksiyonları üzerinde göstermek gerekirse de şu şekilde:

```
69 module.exports = {  
70   //sağ ile eşleşmek zorunda, sol farklı isim alabilir  
71   getNotes: getNotes,  
72   addNote: addNote,  
73 };  
74
```

Burada ise sağ ve sol tarafta da aynı isim verildiği görülüyor. Sağ tarafta bulunan o dosyada kullanılan ismi olurken sol tarafta bulunan ise başka bir dosyaya aktarma durumunda o dosyada kullanılacak isim olarak kullanılır. Karışıklılık olma durumu olmaması adına da sağ ve sol taraftaki isimleri aynı yazmak tercih edilir.

Bu dosyayı geçen hafta yazdığımız dosyaya ekleme kısmında ise require ile app.js dosyasına notes.js dosyasını dahil ederiz.

```
1  const yargs = require("yargs");
2  const notes = require("./notess"); //nokta bulunduğu konum demek oluyor
3  //require içersinde obje geliyor bu yuzden gecer isim vermek gerekiyor
4  yargs.version("1.1.0");
5
```

Bu kısımda da './dosyaadi' şeklinde dosyayı dahil ediyoruz. Burada kullanılan nokta, bulunduğu konum demek oluyor. İsim verme kısmında da gelen şey bir obje olduğundan spesifik bir isim yerine daha genel isim vermek tercih edilir. Bunu da dosyamızın ismi olan notes ile yaptık.

Artık add içerisine girebiliriz. Öncelikle app.js içerisinde add bölümündeki handler yani işlemlerin yapıldığı yerde bir değişikliğe gittik. O da şu şekilde oldu:

```
6  yargs.command({
7    command: "add",
8    describe: "yeni not ekler",
9    builder: {
10     title: {
11       describe: "Not basligi",
12       demandOption: true,
13       type: "string",
14     },
15   },
16   body: {
17     describe: "Not icerigi",
18     demandOption: true,
19     type: "string",
20   },
21   handler: function (argv) {
22     notes.addNote(argv.title, argv.body);
23   },
24 });
```

notes ile notes.js dosyamızın içine girmeyi sağladık ve oradan da addNote fonksiyonuna parametre gönderdik. Parametre olarak da terminalden gönderdiğimiz argümanların title ve body kısmını gönderdik.

addNotes fonksiyonunda gönderilen parametreyi JSON formatında dosyaya kaydetmeyi ve array döndürmesini amaç ediyoruz. Bunun içinde dosyaya kaydetme işlemini başka işlemlerde kullanma ihtimaline göre başka fonksiyon içerisinde yazıp addNote fonksiyonunda çağıracağız. Bu fonksiyonumuz da loadNotes. loadNotes fonksiyonumuzda ilk olarak not bilgisini getirecek ve eski notlara bakıp ona göre ekleme işlemi yapacak. Aynı olma durumunda ekleme gerçekleşmeyecek.

Kod üzerinden anlatmak gerekirse:

```
1  const fs = require("fs");
2
3  const addNote = function (title, body) {
4      const notes = loadNotes();
5  };
6  const getNotes = function () {};
7
8  const loadNotes = function () {
9      // not bilgisi getirecek
10     // eski notlara bakıp ona göre ekleme yapacağız
11     const dataBuffer = fs.readFileSync("notes.json"); //dosyadan okuma işlemi
12     const dataJSON = dataBuffer.toString(); //okudugu veriyi json formatına
13     //parse the string and return
14     return JSON.parse(dataJSON); //Json formatına dönüşür
15 };
16
```

Fs modülünü require ile projemize ekliyoruz. loadNotes fonksiyonunda readFileSync ile notes.json adında dosyayı okuma işlemi yapıyor ve okunan bilgileri dataBuffer değişkenine atıyoruz. Ardından JSON formatında olan bilgileri toString() metodu ile stringe çeviriyoruz son olarak da return olarak Json formatı olarak array döndürüyoruz. Bu fonksiyonu da addNote fonksiyonunda çağırıyoruz.

loadNotes fonksiyonunda dosya olup olmama durumunda hata ile karşılaşmamak için try catch kullanırız:

```
8  const loadNotes = function () {
9      try {
10         //dosya olmama durumunda hata almamak için try catch bloguna alırız
11         // not bilgisi getirecek
12         // eski notlara bakıp ona göre ekleme yapacağız
13         const dataBuffer = fs.readFileSync("notes.json"); //dosyadan okuma işlemi
14         const dataJSON = dataBuffer.toString(); //okudugu veriyi json formatına çeviriyor
15         //parse the string and return
16         return JSON.parse(dataJSON); //JSON formatına dönüşür
17     } catch (e) {
18         return [];
19     }
20 };
```

Catch e düşme durumunda return[] ile boş bir array döndürmüş olduk.

addNote fonksiyonu içerisinde

```
const notes = loadNotes();
```

ile çağırma işlemi yapmıştık. Şimdi de notes arrayini push ederek eleman eklemeyi yapacağız ve arrayi yazdıracağız.

```
const addNote = function (title, body) {
    const notes = loadNotes();
    notes.push({
        title: title,
        body: body,
    }); //arraye eleman eklemek
    console.log(notes);};
```

Çıktı olarak da şunu alır:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='içerik'
[ { title: 'baslik', body: 'içerik' } ]
PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> []
```

Ardından dosyaya kopyalama işlemi için saveNotes adında fonksiyon oluşturduk. Dosyaya parametre olarak array gönderdik.

```
const dataJSON = JSON.stringify(notes);
```

kullanarak JSON formatında aynı zamanda stirnge çevirmeyi sağlıyor. Dosya yazmada ise

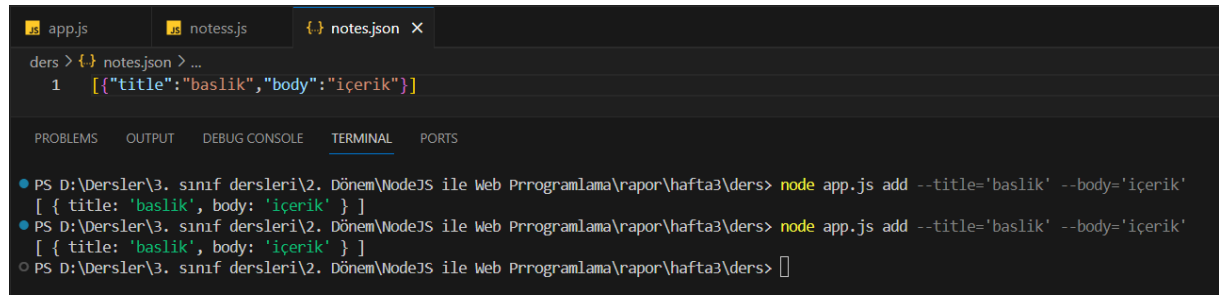
```
fs.writeFileSync("notes.json", dataJSON);
```

writeFileSync komutu kullandık ve notes.json dosyasına dataJSON göndermiş olduk.

SaveNotes fonksiyonumuzu addNote fonksiyonuna şu şekilde dahil edip

```
const addNote = function (title, body) {
  const notes = loadNotes();
  notes.push({
    title: title,
    body: body,
  }); //arraye eleman eklemek
  console.log(notes);
  saveNotes(notes);
};
```

Kodu çalıştırma durumunda çıktımız:



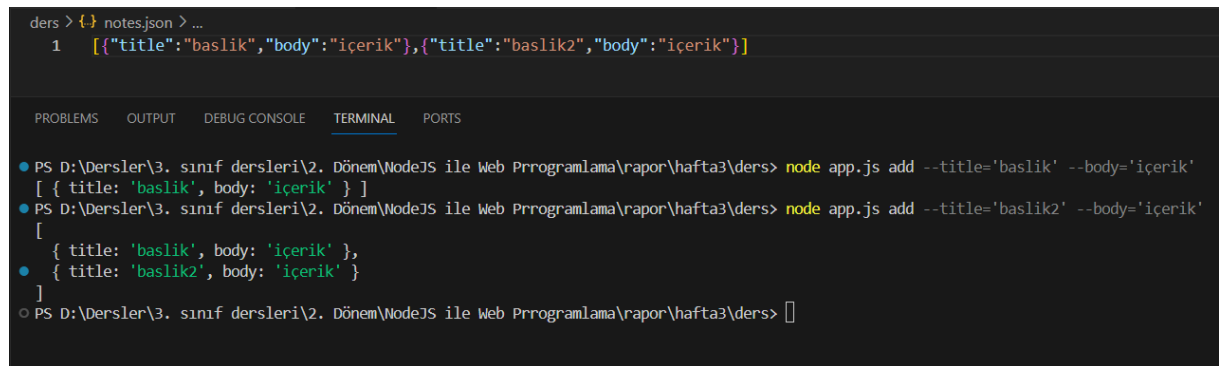
```
ders > {} notes.json > ...
1 [{"title":"baslik","body":"icerik"}]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='icerik' [{ title: 'baslik', body: 'icerik' }]
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='icerik' [{ title: 'baslik', body: 'icerik' }]
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders>

Olur ve notes.json dosyasına girdiğimiz argümanlar yazılır.

Kodu bir daha çalıştırma durumuna bakarsak:



```
ders > {} notes.json > ...
1 [{"title":"baslik","body":"icerik"}, {"title":"baslik2","body":"icerik"}]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='icerik' [{ title: 'baslik', body: 'icerik' }]
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik2' --body='icerik' [{ title: 'baslik', body: 'icerik' }, { title: 'baslik2', body: 'icerik' }]
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders>

Sonradan yazılan başlık sona eklendi yani append işlemi yapmış oldu.

Aynı başlık yapma durumunda kabul etmeme şartını kodumuza uyarladık. Onda da kıyas işlemi ile yaptık. Bunu da addNote fonksiyonunda filter komutu kullanarak yaptık.

Filter komutuna fonksiyon attık ve parametre olarak note attık. Bu parametre note argümanı olucak. Kıyas olarak da return üzerinden argümanın title si bizim bulunan title lara eşit olup olmama durumuna bakıcak. Tek bir satırlık kod olsa da tüm title ları tarıyor olucak eşit olma durumunda da return olarak döndürecek.

Bu tüm filter komutunu duplicateNotes adında değişkene atadık ve ardından if ile kıyas yaptık. duplicateNotesin değeri

olma durumunda eşleşme olmaması durumu olucak ve not eklenebilecek.Else durumunda ise not eklenmeyecek ve uyarı mesajı verecek. Kod üzerinden bakmak gerekirse:

```
const addNote = function (title, body) {  
  //notları dosyaya kaydedicez  
  const notes = loadNotes(); //array  
  dondurcek  
  const duplicateNotes =  
notes.filter(function (note) {  
    //aynı başlık olma durumunda kosula  
sokmak ıcın bu fonk yazdık.  
    //note.title: notesin ıcindeki  
elemanlardan bır tanesi  
    return note.title === title;  
  });  
  if (duplicateNotes.length === 0) {  
    //hiçbir eşleşme yok,yeni not eklenebilir  
    notes.push({  
      title: title,  
      body: body,  
    }); //arraye eleman eklemek  
    console.log(notes); // arrayin  
içindekileri consola yazdırma  
    saveNotes(notes);  
  } else {  
    //o başlık daha once alınmıs  
    console.log("Bu başlık daha once  
kullanıldı.Not eklenemiyor!!!");  
  }  
};
```

Çıktı olarak da :

```
20 const saveNotes = function (notes) {  
  PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js add --title='baslik2' --body='içerik'  
  Bu başlık daha önce kullanıldı,Not eklenmiyor!!!  
○ PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> []
```

Bundan sonra da remove komutuna baktık.

app.js dosyasında remove olan kısımda birkaç değişiklik yaptık. Şunun gibi :

```
yargs.command({  
  command: "remove",  
  describe: "secilen notu siler",  
  builder: {  
    title: {  
      describe: "Not basligi",  
      demandOption: true,  
      type: "string",  
    },  
  },  
  handler: function (argv) {  
    notes.removeNote(argv.title);  
  },  
});
```

Silme işleminde title üzerinden yapacağımız için sadece title ı ekledik. Yine argümandan alınan değeri notes üzeerinden removeNote fonksiyonuna gönderdik.

Notess.js dosyasında ilk olarak removeNote fonksiyonunu yapıp export ettik.

Fonksiyonda notları loadNotes fonksiyonu ile çektik çünkü not olmama ihtimaline karşın bir kıyas yapmalıydık. Yine kıyas yapma işlemi filter ile sağladık. Filter içerisinde olan fonksiyonu note parametresi ile gönderdik. Argüman olan note dan title a ulaşp bulunan title ları kıyasladık ve bu fonksiyonda eşit olmama durumuna göre return olarak döndürdük. Çünkü açıklamak gerekirse dizimizin elemanları geri dönecek yani eşit olamayan her eleman dönmüş olacak ama eşit olma durumunda yazmamalı çünkü o başlık altındaki not silinmeli. Ardından filter komutunu notesToKeep değişkenine atadık. Ardından filter çıkışında if ile koşul alacağız. Eğer başlangıçtaki not sayımız (notes.length) filterden geçen not sayısından(notesToKeep) büyükse demek ki not silinmiş ve oraya uygun bir çıktı döndürebiliriz. Ardından notesToKeep yani filter edilmiş notları saveNotes ile dosyaya kaydedebiliriz.

Else düşme durumunda yani büyük olmama durumunda eşittir yani silme işlemi olmamıştır. Bu durumda da yine konsola çıktı vereceğiz. Koda ve çıktıya bakalım:

```
const removeNote = function (title) {
  const notes = loadNotes(); //notları aldık
  const notesToKeep = notes.filter(function (note) {
    return note.title !== title;
  });
  if (notes.length > notesToKeep.length) {
    console.log("Note remove");
    saveNotes(notesToKeep);
  } else {
    console.log("Silmek istediginiz not bulunamamıştır.");
  }
};
```

Çıktı olarak da önce olmayan sonra olan bir başlık ismi girdim :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js remove --title='baslik5'
Silme istediginiz not bulunamamıştır.
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node app.js remove --title='baslik'
Note remove
○ PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> 
```

Aynı işlemi not çıktısı olarak bulma durumunda kırmızı bulmama durumunda ise yeşil arka planda yazdırmaya çalıştık onu da chalk modülü ile sağladık. Öncelikle require ile dahil ettik.

```
const chalk = require("chalk");
```

Chalk modülünü terminal üzerinden

<<npm i chalk@4 >>şeklinde indirdik

Chalk kullanımı olarak da:

```
if (notes.length > notesToKeep.length) {
  //başlangıctaki not sayısı sonrakinden
  büyükse silinmiştir aksi halde silme yoktur
  console.log(chalk.green.inverse("Note
remove")); //inverse terse çekiyor renkleri
  ondeki ile arkadaki
  saveNotes(notesToKeep);
} else {
  console.log(chalk.red.inverse("Silme
istediginiz not bulunamamıştır."));
}
```

inverse arka plan ve yazı rengini terse çevirmeye yazıyor.

Çıktısı olarak da:

```
Note remove
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node app.js remove --title='baslik'
Silme istediğiniz not bulunamamıştır.
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node app.js remove --title='baslik2'
Note remove
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> []
```

Aldık. Aynı işlemi add üzerinden de yaptık. Not ekleme durumu yeşil;bulamama durumu ise kırmızı.

```
console.log(chalk.green.inverse("yeni not eklendi")); // arrayin içindekileri consola yazdırma
    saveNotes(notes);
} else {
    //o başlık daha önce alınmış
    console.log(
        chalk.red.inverse("Bu başlık daha önce kullanıldı.Not eklenemiyor!!!")
    );
}
```

Çıktısına da bakarsak:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='icerik'
yeni not eklendi
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node app.js add --title='baslik' --body='icerik'
Bu başlık daha önce kullanıldı.Not eklenemiyor!!!
• PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> []
```

Arrow function – JS ES6 Feature

Arrow_deneme.js dosyasına geçtik.

```
ders > JS arrow_deneme.js > ...
1  const square = function (x) {
2    |   return x * x;
3    |   };
4
5  console.log(square(4));
6

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
16
○ PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> 
```

Yukarıdaki kodda klasik bir kare alma fonksiyonu yazdık.

Ardından bunu arrow function ok dosyası adı verilen arrow formatında yapacağız. Bu daha modern bir yöntem ok kullanıldığı için de bu şekil isimlendirilmiş.

```
9  // Arrow function - Js ES6 feature
10 const square = (x) => {
11   |   return x * x;
12   |   };
13 console.log(square(4));
14

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
16
● PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
16
● PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
16
○ PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> 
```

Yukarıdaki işlemlerde de yazılabilir fakat eğer bizim yapacağımız işlem if else bulundurmuyorsa, kompleksizse ve tek satırda yazılabiliyorsa tek satırda da yazabiliriz. O da şu şekilde:

```
16 const square = (x) => x * x;
17 console.log(square(4));
18

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
16
○ PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> 
```

Başka bir örnekte bakmak gerekirse bir event oluşturalım. Bir doğum günü partisi listesi tutsun.

```
21 const myEvent = {
22   name: "xx bebek doğum gunu partisi",
23   printGuestList: function () {
24     console.log(this.name + "için katılımcı listesi");
25   },
26 };
27
28 myEvent.printGuestList();
29
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js 16
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js 16
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
xx bebek doğum gunu partisi için katılımcı listesi
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders>

Bu yazılım bizim klasik halimizdir. Yine bunu arrow halini yazarsak :

```
28 const myEvent = {
29   name: "xx bebek doğum gunu partisi",
30   printGuestList: () => {
31     console.log(this.name + "için katılımcı listesi");
32   },
33 };
34
35 myEvent.printGuestList();
36
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js 16
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js 16
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
xx bebek doğum gunu partisi için katılımcı listesi
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
undefined için katılımcı listesi
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders>

Bu şekil yapabiliydik fakat çıktıda da görüldüğü gibi this.name için hata veriyor. Çünkü arrow yazmada kendi değişkenine erişemiyor ve undefined çıktısı veriyor.

Bunu çözmek için oku kaldırıp yazabiliriz :

```
28  const myEvent = {
29      name: "xx bebek doğum gunu partisi",
30      printGuestList() {
31          console.log(this.name + "için katılımcı listesi");
32      },
33  };
34
35  myEvent.printGuestList();
36
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
xx bebek doğum gunu partisi için katılımcı listesi
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders>

Görüldüğü üzere bu şekilde çalıştı. Şimdi bir de kişi listesi ekleyelim burada çalışma durumuna bakalım.

```
28  const myEvent = {
29      name: "xx bebek doğum gunu partisi",
30      guestList: ["Gizem", "Kadir", "Özgül"],
31      printGuestList() {
32          console.log(this.name + "için katılımcı listesi");
33          //this diyerek objeyi refore ettik
34          this.guestList.forEach(function (guest) {
35              //guest listedeki her elemanın ismi
36              console.log(guest + ", " + this.name + "ne katılıyor");
37              //yazılacak her sey array in her elemanı için teker teker çalışacak sebebi de foreach komutu
38          });
39      }
40  };
41
42  myEvent.printGuestList();
43
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders> node arrow_deneme.js
xx bebek doğum gunu partisi için katılımcı listesi
- Gizem, undefinedne katılıyor
Kadir, undefinedne katılıyor
Özgül, undefinedne katılıyor
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Programlama\rapor\hafta3\ders>

Bu sefer de name e ulaşamıyoruz. Bu sorunu

const that = this

yazarak çözebiliriz. this yazan yere that yazıp çalışmasını sağlayabiliriz. Ama bu çözüm önerilmez. Bu yüzden başja bir yöntemle çözeceğiz.

Bu yöntemde de foreach kullanarak katılımcı listesini teker teker tarayacağız ve string toplama şeklinde yazdıracağız o da şu şekilde olacak:

```

28 const myEvent = {
29   name: "xx bebek doğum gunu partisi",
30   guestlist: ["Gizem", "Kadir", "Özgül"],
31   printGuestList() {
32     console.log(this.name + " için katılımcı listesi");
33     //this diyerek objeyi refore ettik
34     this.guestlist.forEach((guest) => {
35       //guest listedeki her elemanın ismi
36       console.log(guest + ", " + this.name + "ne katılıyor");
37       //yazılacak her sey array in her elemanı için teker teker çalışacak sebebi de foreach komutu
38     });

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders> node arrow_deneme.js
xx bebek doğum gunu partisi için katılımcı listesi
Gizem, xx bebek doğum gunu partisine katılıyor
Kadir, xx bebek doğum gunu partisine katılıyor
Özgül, xx bebek doğum gunu partisine katılıyor
- PS D:\Dersler\3. sınıf dersleri\2. Dönem\NodeJS ile Web Prrogramlama\rapor\hafta3\ders>

Son olarak istenilen her şart sağlandı. 36.satırda bulunan guest i 34. Satırda foreach içine parametre olarak atadık ve anlamı da orada bulunan kişi listesi. Haliyle kodumuz kişi listesini tarayacak ve her kişi için ayrı ayrı çalışıp bize çıktı verecek.