

# Node Js Ödevi

## 2<sup>nd</sup> Week

# Farhan Ahmad

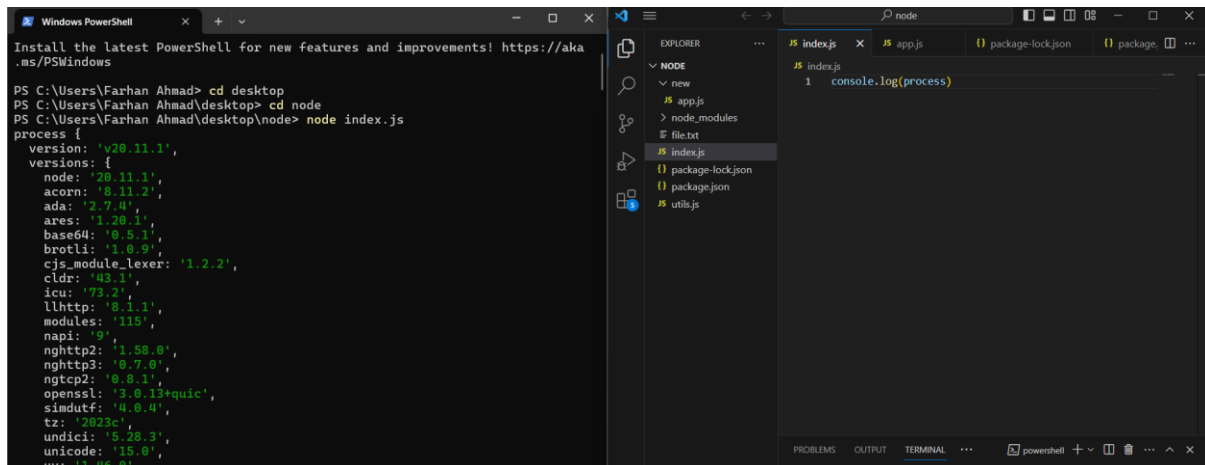
**20360859096**

## Node jste komut satırı argumanlarını yazdırma:

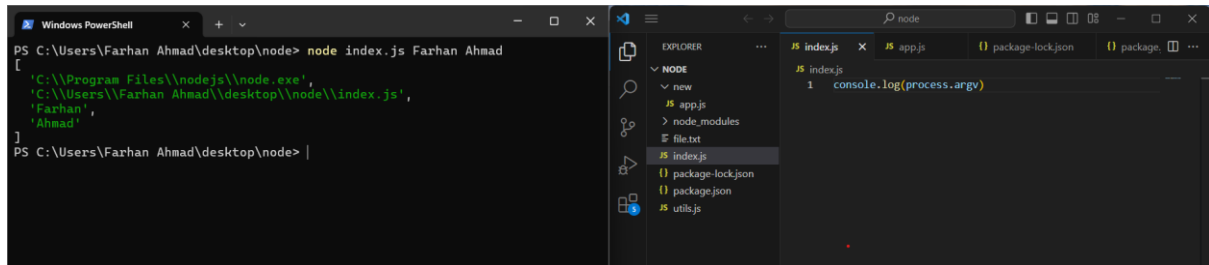
Node js **süreç(Process)** aslında çalışan bir programın bir örneğidir.

Node.js'de bir script çalıştırdığımızda Node.js'de, bu kodun yürütülmesini yönetmek için yeni bir process başlatılır. Node js te bir komut satiri degerini tutmak icin **process.argv** kullanilir.

```
console.log(process)
```



```
console.log(process.argv)
```

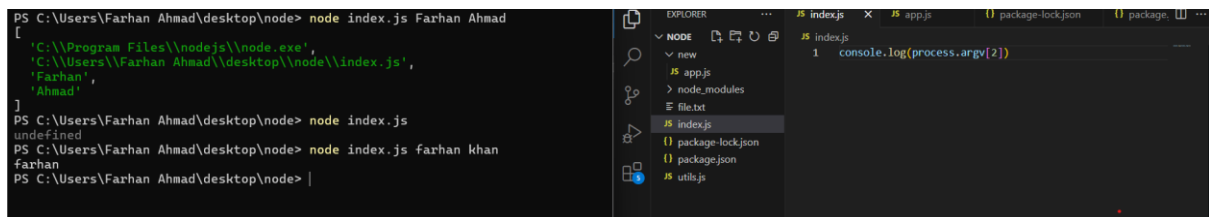


```
PS C:\Users\Farhan Ahmad\desktop\node> node index.js Farhan Ahmad
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\Farhan Ahmad\\desktop\\node\\index.js',
  'Farhan',
  'Ahmad'
]
PS C:\Users\Farhan Ahmad\desktop\node> |
```

Skripti yeniden çalıştırınca, process.argv dizisi güncellenir. Yeni dizi üç elemana sahiptir: Node yürütücüsünün yolunu, dosyanın yolunu ve argümanı içerir.

```
console.log(process.argv[2])
```

bu kodu kullandıktan sonra 2. Indexteki veriyi sadece gösteriyor.



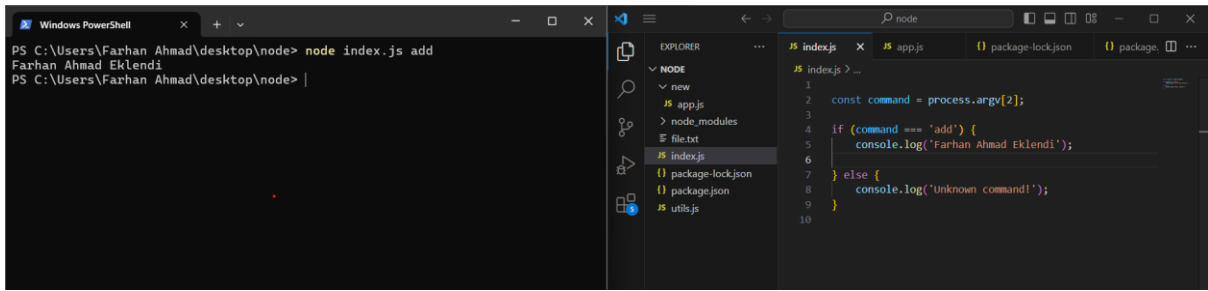
```
PS C:\Users\Farhan Ahmad\desktop\node> node index.js Farhan Ahmad
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\Farhan Ahmad\\desktop\\node\\index.js',
  'Farhan',
  'Ahmad'
]
PS C:\Users\Farhan Ahmad\desktop\node> node index.js
undefined
PS C:\Users\Farhan Ahmad\desktop\node> node index.js farhan khan
farhan
PS C:\Users\Farhan Ahmad\desktop\node> |
```

## Üç temel argümanları

- Add
- Remove
- List

"add" argümanı, kullanıcının bir not eklemek istediğini belirtir. Bu durumda, add argümanını kullanarak yeni bir not eklemek için aşağıdaki gibi bir kod kullanılabilir:

```
const command = process.argv[2];
if (command === 'add') {
  console.log('Farhan Ahmad Eklendi');
} else {
  console.log('Unknown command!');
}
```



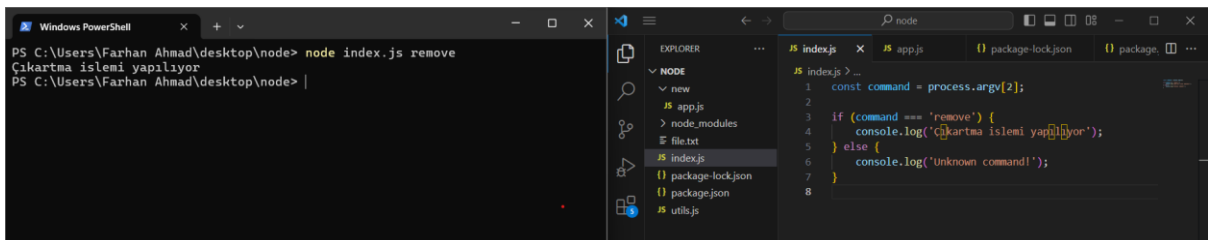
```
PS C:\Users\Farhan Ahmad\desktop\node> node index.js add
Farhan Ahmad Eklendi
PS C:\Users\Farhan Ahmad\desktop\node> |
```

```
1
2 const command = process.argv[2];
3
4 if (command === 'add') {
5   console.log('Farhan Ahmad Eklendi');
6 }
7 } else {
8   console.log('Unknown command!');
9 }
10
```

“**remove**” komutu girildiğinde Silme(Removing) işlemi yapar. Mesela aşağıdaki kodda remove işlemi yaptığımızda "Removing note!" mesajını konsola yazdıracak ve daha sonra not silme işlemi için gerekli işlemleri gerçekleştirebilir. Eğer kullanıcı tanımlanmayan bir komut girerse, "Unknown command!" mesajını verecektir.

```
const command = process.argv[2];

if (command === 'remove') {
  console.log('Çıkartma işlemi yapılıyor');
} else {
  console.log('Unknown command!');
}
```



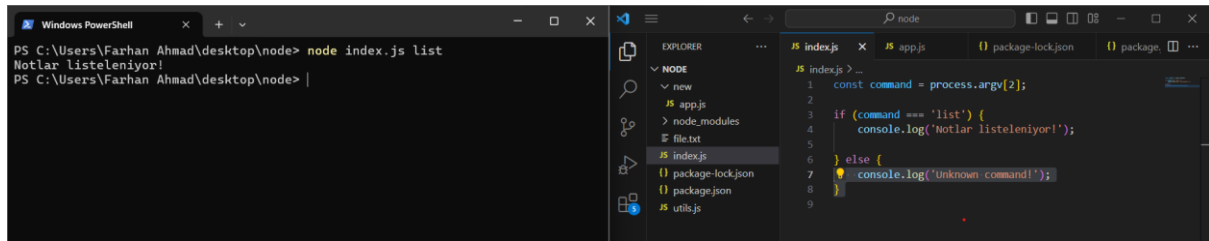
```
PS C:\Users\Farhan Ahmad\desktop\node> node index.js remove
Çıkartma işlemi yapılıyor
PS C:\Users\Farhan Ahmad\desktop\node> |
```

```
1 const command = process.argv[2];
2
3 if (command === 'remove') {
4   console.log('Çıkartma işlemi yapılıyor');
5 } else {
6   console.log('Unknown command!');
7 }
8
```

“**List komutu**” argümanı, kullanıcının mevcut notları listelemek istediğini belirtir. Aşağıdaki gibi bir kod kullanarak list argümanını kullanarak mevcut notları nasıl listeleyebileceğinizi göstermektedir:

```
const command = process.argv[2];

if (command === 'list') {
  console.log('Notlar listeleniyor!');
} else {
  console.log('Unknown command!');
}
```

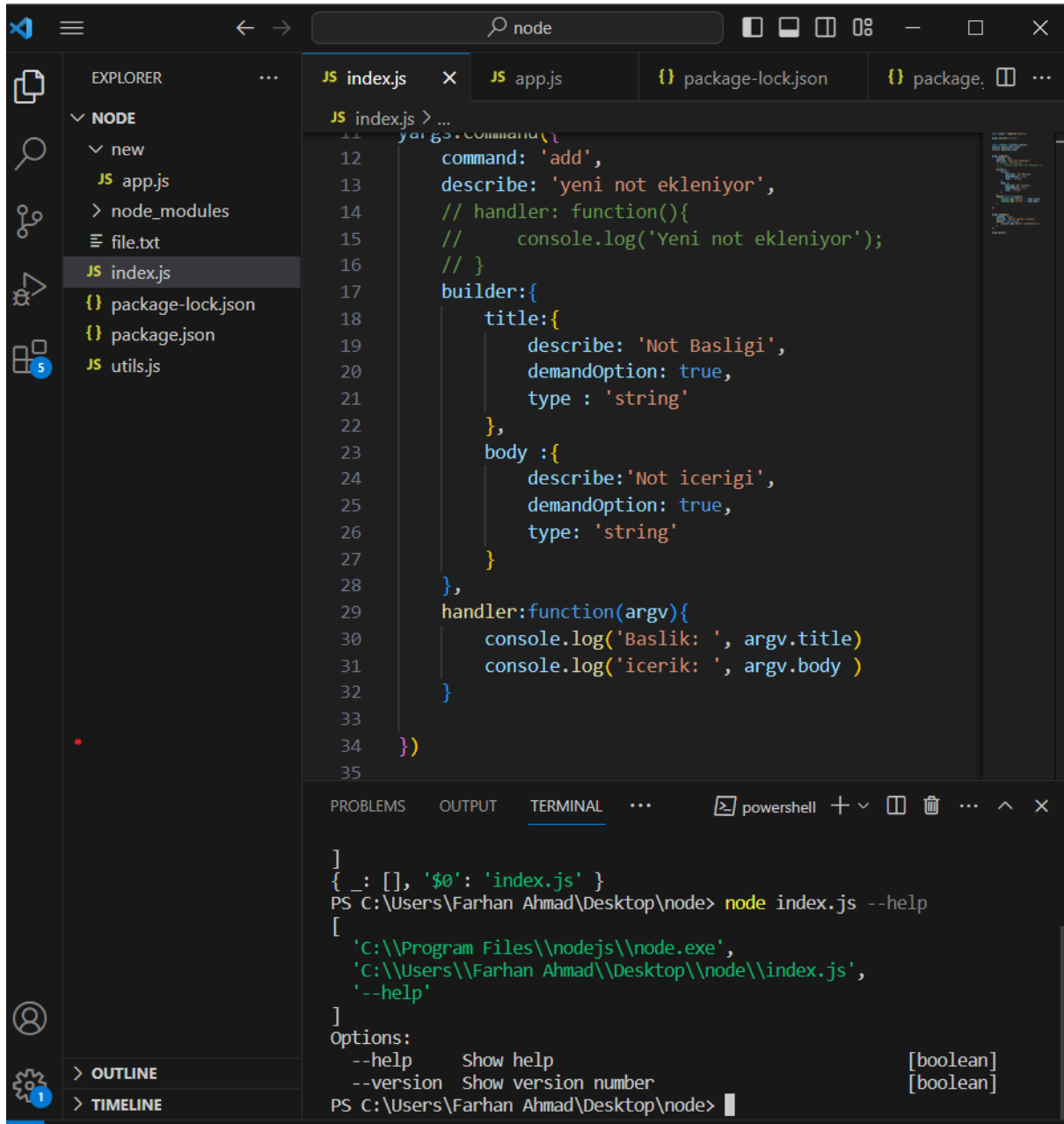


## Yargs Kullanımı:

**“yargs”** Node.js için bir komut satırı argümanı işleme kütüphanesidir. **yargs** kullanarak komut satırı argümanlarını daha kolay ve yapılandırılmış bir şekilde işlenmektedir.

**yargs** modülüne eklemek için:

npm i yargs



```
11 yargs.command({
12   command: 'add',
13   describe: 'yeni not ekleniyor',
14   // handler: function(){
15   //   console.log('Yeni not ekleniyor');
16   // }
17   builder:{
18     title:{
19       describe: 'Not Basligi',
20       demandOption: true,
21       type : 'string'
22     },
23     body :{
24       describe:'Not icerigi',
25       demandOption: true,
26       type: 'string'
27     }
28   },
29   handler:function(argv){
30     console.log('Baslik: ', argv.title)
31     console.log('icerik: ', argv.body )
32   }
33 }
34 })
35
```

```
]
{ _: [], '$0': 'index.js' }
PS C:\Users\Farhan Ahmad\Desktop\node> node index.js --help
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\Farhan Ahmad\\Desktop\\node\\index.js',
  '--help'
]
Options:
  --help    Show help                                [boolean]
  --version Show version number                       [boolean]
PS C:\Users\Farhan Ahmad\Desktop\node>
```

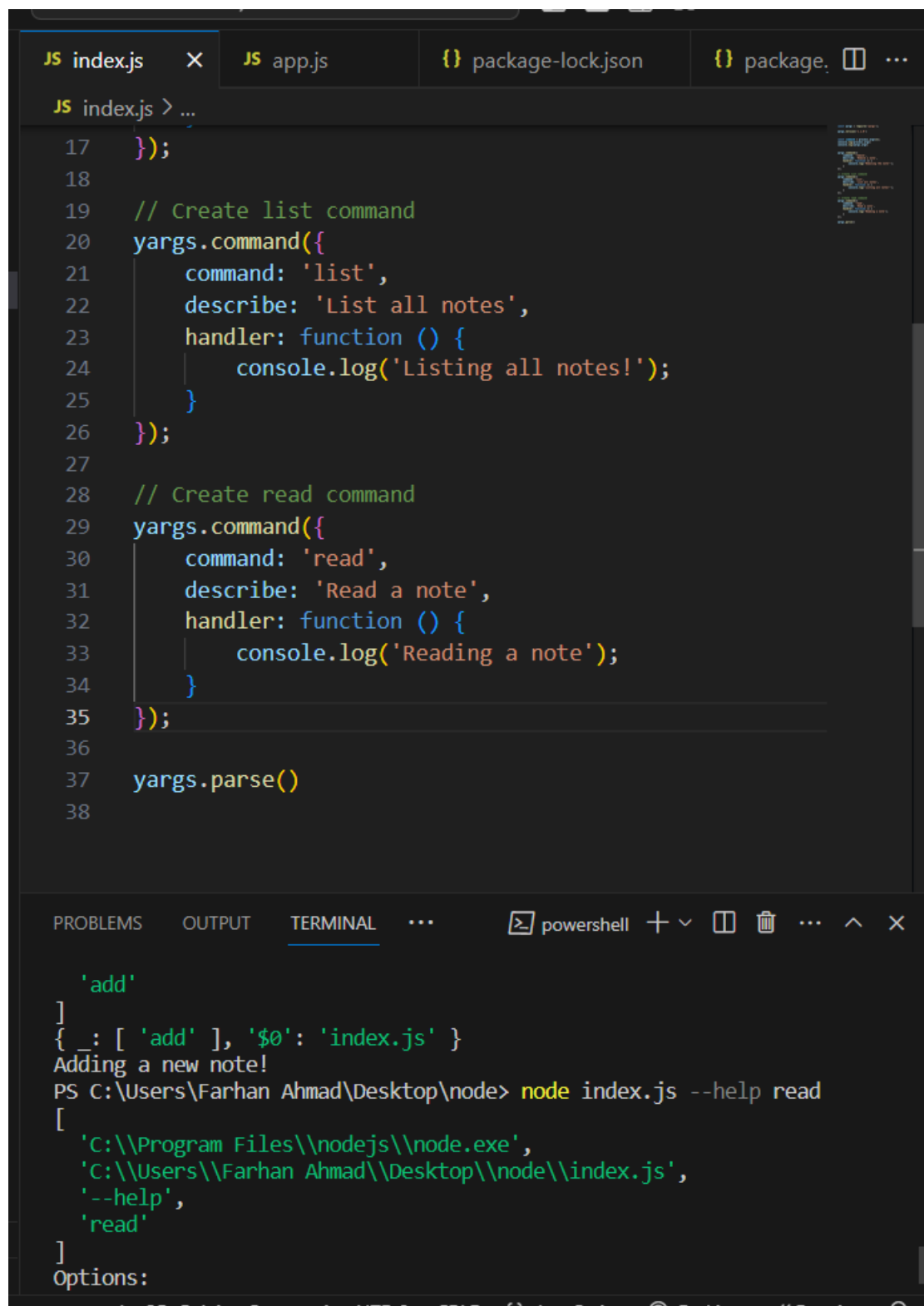
Handler, kullanıcı belirli bir giriş yaptığıında yürütülen işlemdir emretmektedir.

```
JS index.js  X  JS app.js  {} package-lock.json  {} package.  .
JS index.js > ...
1  const yargs = require('yargs');
2
3  yargs.version('1.1.0')
4
5
6  const command = process.argv[2];
7  console.log(process.argv)
8  console.log(yargs.argv)
9
10
11  yargs.command({
12    command: 'add',
13    describe: 'Add a new note',
14    handler: function () {
15      console.log('Adding a new note!');
16    }
17  });
18
19  yargs.parse()
20
```

Bu kodu uygulamanıza ekledikten sonra, mevcut seçenekleri ve komutları görmek **için node index.js --help** komutunu çalıştırabilmek lazım. Add komutunun yardım bilgilerini özel olarak görmek için **node index.js add --help** komutunu çalıştırmaktadır. Add komutunu gerçekten çalıştırmak için **node index.js add** komutunu çalıştırırsınız.

## Remove, list, read command:

Aşağıdaki kodla, çıkartma listeleme ve okuma komutları kullanılabilir.



The image shows a Visual Studio Code editor with a dark theme. The top panel displays four tabs: 'index.js', 'app.js', 'package-lock.json', and 'package.json'. The 'index.js' tab is active, showing a JavaScript file with the following code:

```
17  });
18
19  // Create list command
20  yargs.command({
21    command: 'list',
22    describe: 'List all notes',
23    handler: function () {
24      console.log('Listing all notes!');
25    }
26  });
27
28  // Create read command
29  yargs.command({
30    command: 'read',
31    describe: 'Read a note',
32    handler: function () {
33      console.log('Reading a note');
34    }
35  });
36
37  yargs.parse()
38
```

The bottom panel shows the 'TERMINAL' tab with a PowerShell prompt. The terminal output is as follows:

```
'add'
]
{ _: [ 'add' ], '$0': 'index.js' }
Adding a new note!
PS C:\Users\Farhan Ahmad\Desktop\node> node index.js --help read
[
  'C:\\Program Files\\nodejs\\node.exe',
  'C:\\Users\\Farhan Ahmad\\Desktop\\node\\index.js',
  '--help',
  'read'
]
Options:
```

## JavaScript'te JSON kullanarak veri depolamak ve geri almak:

**Proje Oluşturma:** Öncelikle, projeniz için yeni bir klasör oluşturun ve içine bir script ekemek.

**Not Objesi Tanımlama:** Script dosyasında, bir notu temsil eden bir obje tanımlamak

```
const book = {  
  title: 'my title',  
  author: 'can conomo'  
};
```

**JSON Dönüşümü:** Objeyi JSON dizesine dönüştürmek için `JSON.stringify()` kullanmak:

```
const bookJSON = JSON.stringify(book);  
console.log(bookJSON);
```

**Dosyaya Yazma:** `fs` modülünü kullanarak JSON dizesini bir dosyaya yazın:

```
const fs = require('fs');  
fs.writeFileSync('1-json.json', bookJSON);
```

**Dosyadan Okuma:** Dosyadan JSON dizesini okuyun ve tekrar bir objeye dönüştürmek için `JSON.parse()` kullanmak:



```
const dataBuffer = fs.readFileSync('1-json.json');  
const dataJSON = dataBuffer.toString();  
const data = JSON.parse(dataJSON);  
console.log(data.title);
```

Çalıştırma: Script'i çalıştırarak (node 1-json.js), çıktıyı gözlemlemek.

Bu örnek, JavaScript'te JSON kullanarak veri depolama ve geri almayı göstermektedir.