Node.js

Dr. Özcan Kasal

İçindekiler

1

Node.js

Bu Bölümde

Bu bölümde, size Node.js platformunu genel hatlarıyla tanıtmaya çalışacağız. Bloklamayan I/O (dosya sistemine yazma ve okuma) işlemlerinin, socket yönetimi ve diğer tüm işlemlerin Node.js platformu ve JavaScript ile asenkron program akışı modeli içerisinde doğal bir şekilde yapılabilmesinin vazgeçilmez rahatlığından bahsedeceğiz.

GİRİŞ

Yeni bir dil öğrenmek istiyorsanız ya da yeni bir platform kullanmak durumundaysanız bu yolda en önemli adım öğreneceğiniz dili ya da platformu, genel çalışma prensiplerini, diğer dillerden ya da platformlardan ayrılan yönlerini öğrenmek olabilir. Bu detaylara hakim olduğunuz zaman, bu dilin ya da platformun sizin amaçlarınız için uygun olup olmadığına karar verebilirsiniz.

Node.js özellikle web için dizayn edilmiş olan ve internet tarayıcıları tarafından desteklenen JavaScript dilinin sunucu tarafında kullanılabilmesine olanak sağlamıştır. Böylece sunucu ve istemci için kodlama farklılıklarını ortadan kaldırmıştır. Tek programlama dili kullanarak tüm ihtiyaçların giderilmesinin önünü açmıştır. Veri tipi olarak **JSON** benzeri bir yapı kullanan MongoDB'nin de vitrinlerde yerini almasıyla Node.js ve dolayısıyla JavaScript gücünü tümden artırmıştır. **WebSocket** gibi günümüz web teknolojilerini kullanabilen uygulamaların tek dil çerçevesinde geliştirilmesini sağlamıştır.

JavaScript sayesinde elde edilen geliştirme kolaylığının yanısıra Node.js tek yürütmeli (single threaded) mimari yapı içerisinde **asenkron** program akışını kullanmaktadır. Bu ise geliştiricilere geliştirdikleri yazılımlarda yüksek performans ve ölçeklendirilebilme olanaklarını sunmuştur. Bunlara ek olarak Node.js ile birlikte gelen paket yönetim sistemi NPM ile geliştiriciler ihtiyaç duydukları paketleri bir havuzdan elde ederek kolayca uygulamalarına ekleyebilmektedirler.

NODE.JS NEDİR?

Node.js sunucu tarafında JavaScript kodlama dilini kulanarak web uygulamaları geliştirmek için kullanılan açık kaynaklı bir yürütme çevresi platformudur. Node.js için geliştirici ekip ***nodejs.org*** adresindeki proje sayfasında şu tanımlamayı yapmakta:

Node.js platformu Chrome V8 JavaScript motoru üzerine inşa edilmiş bir Javascript yürütme platformudur. Node.js platformunun olay-güdümlü, bloklamayan bir I/O modeli kullanması platformu hafif ve verimli bir hale getirmektedir. Node.js platformunun paket ekosistemi olan NPM, dünyadaki en geniş açık kaynak ekosistemidir.

Web programcılığı ile az ya da çok ilgilenmiş olan geliştiriciler bir noktada JavaScript ile karşılaşmışlardır. İnternet tarayıcıları üzerinde bir sanal makina içinde yürütülen JavaScript kodları açılan pencere çerçevesi içinde ulaşılabilir durumdadır. Genellikle JavaScript dilini html belgelerindeki **DOM** elemanlarını değiştirmek ya da **AJAX** istekleri göndermek için kullanmışızdır. Aslına bakarsanız JavaScript dili tıpkı diğer programlama dilleri gibi geniş özelliklere sahip

bir programlama dilidir. Google Chrome için geliştirilen V8 JavaScript sanal makinası sayesinde JavaScript kodlarını tarayıcı çerçevesi dışında yüksek hızlarda yürütmek mümkün hale gelmiştir. Bu sayede Node.js platformu dosya sistemine ulaşabilmekte, modül yapısı sayesinde ölçeklendirilebilir web uygulamaları geliştirmeye uygun olarak HTTP sunucusu görevi de görebilmektedir.

Node.js Kullanim Alanlari

Eğer sunucu ile kullanıcı arasında sürekli bir bağlantı gerektiren ve bu bağlantı üzerinde yoğun olarak gerçek zamanlı veri transferi gerçekleştiren bir uygulama tasarlamak istiyorsanız Node.js sizin için uygun bir platform olabilir. Node.js ve Socket.io kombinasyonu ile kullanıcıya gerçek zamanlı olarak güncelleme bilgileri veren bir uygulama geliştirebilirsiniz. **Ruby on Rails** gibi programlama dillerini kullanarak yazılan uygulamalar sunucu üzerinde büyük yük ortaya çıkarabilirler. Çünkü her bir aktif kullanıcı için ayrı bir sunucu süreci başlatılacaktır. Node.js ise her bir kullanıcı için ayrı bir sunucu süreci başlatmaya gerek duymadan tek bir süreç üzerinde tüm işlemleri olay-döngüsü modeli içerisinde yürütebilir.

Örneğin bir sohbet uygulaması tasarlamak istiyorsanız Node.js uygun bir platform olacaktır. Benzer şekilde online oyun servisleri, ortak çalışma platformları gibi alanlarda Node.js üstün performansı ile göz dolduracaktır. İnternet üzerinde konuyla ilgili biraz araştırma yaparsanız göreceksiniz ki milyonlarca kullanıcısı bulunan, internet üzerinden satış gerçekleştiren birçok firma Node.js platformunu kullanmaktadır.

Sosyal medya olarak adlandıracağımız, kullanıcıların yoğun etkileşim içinde bulunduğu, resim, video, ses dosyalarının yoğun olarak paylaşıldığı sistemlerde Node.js yüksek performans, düşük sistem gereksinimleri ve geliştirme maliyeti sağlayabilir.

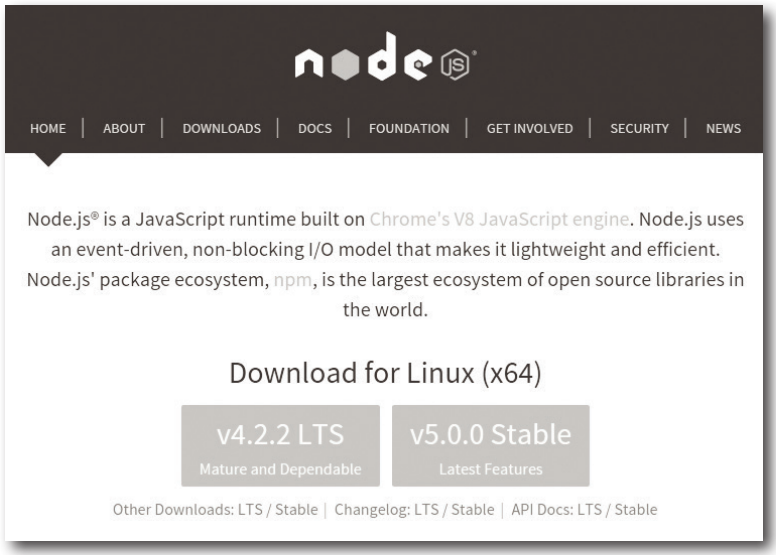
Node.js Avantajları ve Dezavantajları

Öncelikle Node.js ile uygulama geliştirilirken sunucu ve istemci tarafındaki kodlar için tümüyle JavaScript kullanılması, veri transferi, form doğrulama, veritabanı işlemleri (MongoDB kullanılarak) ve DOM öğelerinin etkileşimli özelliklerine kadar aynı dil kullanılması geliştirici için hayatı oldukça kolaylaştırmaktadır.

Sunucu yapılandırması **Apache** veya **Tomcat** ile karşılaştırılamayacak kadar basittir. Birkaç satırlık kod ile tam anlamıyla fonksiyonel bir http sunucusu yaratabilirsiniz.

NODE.JS KURULUMU

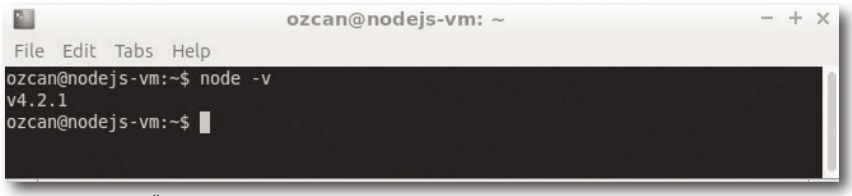
Node.js platformu çeşitli işletim sistemleri için geliştirilmiş kurulum programlarını da kullanıcılara sunmaktadır. Node.js platformunu kurmak için öncelikle platformun internet sayfası olan ***http://nodejs.org*** adresine ulaşmamız gereklidir. Resim 2.1'de görüldüğü gibi platformun anasayfasında Node.js'in iki sürümü hızlıca indirilebilir durumdadır.



Resim 2.1 Node.js Anasayfası

Burada Node.js platformunun sürümleri hakkında bir parantez açmak gerekiyor. Şu anda Node.js platformunun uzun süre desteklenen sürümü 4.2.2 (Long Term Support ya da kısaca LTS). Bunun yanında son kararlı sürümü 5.0.0. Kitapta Node.js geliştirme konularını ele alırken 4.2.2 sürümünü göz önüne alacağız. 5.0.0 sürümü her ne kadar kararlı bir sürüm olsa da ve bazı yeni özellikleri beraberinde getirse de Node.js geliştiricilerinin bu sürüme vereceği destek sekiz ay ile sınırlı. Halbu ki LTS sürümlerine verilen destek 18 ay sürmekte. Bu anlamda LTS sürümleri Node.js platformunu yeni tanımaya başlayan geliştiriciler için uygun bir seçim olacaktır.

Şimdi Node.js platformunun çeşitli işletim sistemleri için kurulumunun nasıl yapılacağı konusunu ele alalım. Node.js Anasayfası üzerinde **DOWNLOADS** kısmına giriş yaptığımızda Resim 2.2.'de görebileceğimiz gibi çeşitli platformlar için kurulum dosyalarına erişebiliriz.

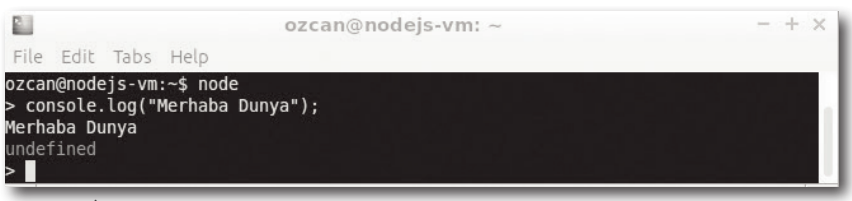


Resim 2.14 Konsol Üzerinde Node.js Sürüm Numarasını Görüntüleme

Node.js platformunun kurulumunu anlatırken bahsettiğimiz gibi Node.js konsol arayüzüne erişmek için konsol üzerinde `node` komutunu giriyoruz. Konsol arayüzüne ulaştıktan sonra isterseniz ilk programı yazalım. Konsol ekranına javaScript ile çıktı sağlamak için kullandığımız komut şöyledir.

```
console.log("Konsolda yer alacak ifade");
```

Şimdi konsol üzerinde bu komutu kullanarak ilk programımız yazalım. Konsol üzerinde **Merhaba Dünya** yazdırmak için Resim 2.15'te görülen kodu çalıştırdığımızda ekrana istediğimiz ifade yazdırılacaktır.



Resim 2.15 İlk Node.js Programı

DİKKAT	Dikkat ederseniz ilk programımızdaki ifadede **Türkçe** karakter kullanmaktan kaçındık. Kullandığınız konsol programı UTF-8 karakter kodlamasını kullanıyorsa bir probleme yol açmayabilir. Eğer sisteminiz **UTF-8** dil kodlaması için doğru şekilde yapılandırılmadıysa Türkçe karakterler bir yürütme hatasına yol açabilir. Bunun için kitaptaki konsol için kodlama örneklerinde **Türkçe** karakter kullanmaktan kaçınacağız.

Node.js Betiklerini Yürütme

Konsol üzerinde JavaScript kodlarını nasıl yürüteceğimizi bir önceki başlıkta ele almıştık. Node.js geliştirme çalışmalarının büyük çoğunluğu dosya sisteminize kayıtlı `.js` uzantılı JavaScript dosyalarınızı yönetme şeklinde gerçekleşeceği için konsol üzerinde bu dosyaların nasıl çağrılacağının, içeriklerindeki kodların nasıl yürütüleceğinin bilinmesi oldukça önemli. Dosya sisteminde

4

NPM ile Paket Yönetimi

Bu Bölümde

Bu bölümde

Öncelikle Node.js platformunun kullandığı en önemli araçlardan biri olan JSON veri transferi formatı üzerinde kısaca duracağız. NPM paket yönetim sistemini kullanarak yerel ve global modüllerin yüklenmesini, kaldırılmasını ve uygulamalarda kullanılmasını göreceğiz.

Giriş

Node.js platformu gücünü akıllıca tasarlanmış modül yapısının yanısıra NPM paket yönetim sisteminden almaktadır. NPM ile geliştiriciler yazdıkları modülleri diğer geliştiricilerle paylaşabilmektedir. NPM bu modüllerin taranabilmesine, bağımlılıkların otomatik olarak çözümlenebilmesine olanak vermiştir. JSON veri transferi formatını kullanan NPM ile geliştiriciler için proje geliştirme ve paylaşma konusunda bir standart oluşturulmuştur diyebiliriz.

JSON

Farklı noktalarla veri alışverişi yapacak bir sistemin tasarımı yapılırken bir veri formatı ve transfer için protokol belirlenmelidir. JSON hafif, metin temelli bir veri transfer formatıdır. İnsan tarafından okunabilir olması, platformdan bağımsız olması, XML formatına göre çok daha az yer kaplaması gibi özellikleriyle zaman içinde geliştiriciler arasında popüler hale gelmiştir. Temel olarak JavaScript dilinin bir alt kümesi kullanılarak ortaya çıktığı için özellikle JavaScript dili tarafından doğal olarak çözümlenebilmektedir.

JSON veri formatı iki tür yapı üzerine inşa edilmiştir.

1. İsim/değer çifti koleksiyonu
2. Sıralı değer listesi

Örnek olarak alttaki koda bakalım. Süslü parantezler içerisinde üç adet isim/değer çifti bulunmakta. Bunlar **virgül** ile ayrılmış ve isim ile değer arasına **iki nokta** üst üste konulmuştur.

```
{
  "ad" : "Özcan",
  "soyad" : "Kasal",
  "dogumTarihi" : 1981
}
```

Yukarıdaki örneği biraz daha geliştirelim. E-posta adreslerini saklayabilmek için bir **isim/değer** çifti daha ekledik. Burada isim olarak **eposta** kullanıldığını görebilirsiniz. Değer kısmında ise `[ ve ]` parantezleri arasında **virgülle** ayrılmış sıralı bir liste olduğunu görebilirsiniz.

```
{
     "ad" : "Özcan",
     "soyad" : "Kasal",
     "dogumTarihi" : 1981,
     "eposta" : [
          "kasal@ornekadres1.com",
          "kasal@ornekadres2.com",
          "kasal@ornekadres3.com"
     ]
}
```

Genel olarak JSON veri yapısı süslü parantezler içinde virgülle ayrılmış isim/değer çiftlerinden oluşur. Burada isim mutlaka **çift tırnaklar** arasına alınmış metin olmalıdır. Değer ise; `String` (alfanumerik), `Number` (numerik), `Boolean` (Doğru/Yanlış), Array (dizi), `Object` (nesne) ya da `Null` (boş) veri formatlarından biri olmalıdır. `dizi` formatındaki değerler köşeli parantezlerin arasına alınır. JSON veri formatı JavaScript dilinin bir alt kümesinden türetildiği için JSON nesnelerinin JavaScript nesnelerine çevrilmesi ya da tersi oldukça basit bir işlemdir. Yukarıdaki örnekte bulunan ifadeyi bir `string` değişkenine aşağıdaki gibi atayalım.

```
var text = '{' +
     '"ad" : "Özcan",' +
     '"soyad" : "Kasal",' +
     '"dogumTarihi" : 1981,' +
     '"eposta" : [' +
          '"kasal@ornekadres1.com",' +
          '"kasal@ornekadres2.com",' +
          '"kasal@ornekadres3.com"' +
  ']' +
'}';
```

`text` değişkeninden yola çıkarak bir JavaScript nesnesi elde etmek için `JSON.parse()` yöntemini kullanabiliriz.

```
var obj = JSON.parse(text);
```

Böylece geçerli bir JavaScript nesnesi elde etmiş oluruz. Nesnenin özelliklerine alışılmış şekilde ulaşabiliriz.

7

Express ile Web Yönetimi

Bu Bölümde

Bu bölümde

Express paketini kullanarak web sunucusu yaratmayı öğreneceğiz. Express ile yarattığımız sunucu için yapılandırma ayarlarını yapmayı, rota düzenlemeyi, HTTP isteklerini yönetmek için Request ve Response nesnelerini kullanmayı öğreneceğiz. Bunların yanısıra Jade şablon motoru ile HTML şablonu oluşturmayı ve Express uygulamalarında kullanımını öğreneceğiz.

Giriş

Önceki bölümde Node.js platformunu kullanarak HTTP isteklerini nasıl yönetebileceğimiz konusuna bir giriş yaptık. Node.js platformunun çekirdek modüllerini kullanarak HTTP isteklerini yönetmeye çalışmak geliştiriciler için tekerleği yeniden icat etmek gibidir diyebiliriz. Hızlı şekilde web uygulamalarının geliştirilebileceği, HTTP ile ilgili temel konuların mümkün olduğunca az kod yazarak halledilebileceği çeşitli paketler mevcuttur.

Express.js projesi Node.js platformunun http modülü ile ilgili işlemleri kolay kullanılabilir bir hale getirmek için geliştirilmiş bir projedir. http modülünün sağladığı işlevlerle sınırlı kalmayarak sunucu rotalarını yönetme, HTTP yanıt mesajlarını düzenleme/yönlendirme, çerezleri yönetme gibi konularda ek işlevler sağlamaktadır.

Express Nedir?

Node.js platformu için geliştirilmiş olan paketler içerisinde bugüne kadar belki de en popüler olan paket **Express** paketidir. Node.js ile geliştirilmiş web uygulamalarının çoğunluğunda Express kullanıldığını kolayca söyleyebiliriz. Peki Express'i bu kadar popüler yapan ne ki? Önceki bölümde gördüğümüz gibi Node.js platformu ile HTTP sunucusu oluşturmak birkaç satırda mümkün iken niye bir çerçeve program kullanalım?

Bu sorulara çeşitli dozda yanıtlar verilebilir. Özetle şunları söylemek sanırım yeterli olur. Günümüzde web uygulamalarının uyması zorunlu olmasa da tercih edilen çeşitli kalıplar mevcut. Uygulamanız mümkün olduğunca MVC yapısına uygun olmalı. Rota yönlendirmeleriniz REST ilkeleri çerçevesinde yapılmalı. Tabi bu tip standartlara uygun bir web uygulamasını yalnızca Node.js platformunun çekirdek modüllerini kullanarak geliştirmek gereksiz zaman israfı olabilir. Ayrıca zaten keşfedilmiş ve geliştiricilere sunulmuş olan araçları kullanmak yerine bunları her seferinde yeniden yaratmak genel prensipler açısında çok uygun olmayabilir.

Madalyonun diğer tarafında ise bu tip çerçeve programların geliştiricilere getirdiği kısıtlar bulunmakta. Geliştiriciler için kullanılan araçların sağladığı esneklik oldukça önemli bir konu. Bazen hayatını ne kadar kolaylaştıracak olsa da önüne konulan kısıtlar nedeniyle geliştiriciler uzun yolu tercih edebilirler.

Basitçe anlatmak gerekirse `extends` komutu ile kullanılacak şablon sayfaya eklenmekte. Bu şablon yine **view** klasöründe bulunan **layout.jade** dosyasıdır. Jade şablon motoruna bu bölümün devamında ayrıntılı olarak değineceğiz. Üstteki kodun devamını özetlemek gerekirse `block content` komutu ile `layout.jade` şablonunda yer alan `content` kısmına referans verilmektedir. Devamında yer alan kodlar `layou.jade` şablonunun `content` kısmında görüntülenecektir. **routes/index** dosyasında tanımlanan `title` değişkeni sayfaya yine `title` adıyla aktarılmıştır. Dikkat ederseniz tarayıcımızla ana sayfayı açtığımızda `rota` dosyasında tanımlandığı gibi `title` değişkeni ile ekrana `Express` yazdırılmıştır.

JADE İLE HTML ŞABLONU OLUŞTURMA

Jade aslına bakarsanız html oluşturabilen bir dil olarak nitelendirilebilir. Express paketi için ön tanımlı şablon motoru olarak kullanılmaktadır. Node.js platformunun dışında Jade şablon motoru php, scala, ruby, python ve java dilleri ile de kullanılabilmektedir. Jade ile sayfaya geçilen veri yorumlanarak sayfa içerisinde istenildiği şekillerde görüntülenmesi sağlanabilmektedir.

Genel anlamda şablon motorlarını kullanmamızın sebeplerinden biri sayfalarımızı dinamik olarak oluşturabilmektir. Sayfanın görüntülenme şekliyle ilgili bir değişiklik yapmak istersek sadece şablonlarımızda değişiklik yaparak diğer sayfalarımıza bu değişikliklerin yansımasını sağlayabiliriz.

Jade şablon motoru boşlukları ve satır girintilerini anlamlandırarak kendi dilinin bir parçası halinde kullanabilmektedir. Jade için satır başlarında bulunan kelimeler HTML etiketleri olarak yorumlanırlar. **Örneğin;** satır başında `div` kelimesi varsa bunun html karşılığı `<div></div>` olarak yorumlanmaktadır. Yani Jade kullandığımızda etiket açmak ve uygun noktalarda bu etiketleri kapatmak gibi bir zorunluluğumuz yoktur. Örnek vermek gerekirse;

```
div
    h1 Merhaba Express
    p Bu ilk Jade kullanimi
```

satırları Jade tarafından yorumlanarak HTML diline şöyle çevrilmektedir.

```
<div>
    <h1>Merhaba Express</h1>
    <p>Bu ilk Jade kullanimi</p>
</div>
```