Node.js'in Rolü

- Node.js, sunucu tarafı betikleme için kullanılan açık kaynaklı bir dildir ve V8 motoru üzerinde çalışır.
- JavaScript, istemci tarafı işlevselliği için yaygın olarak kullanılırken, Node.js sunucu bileşeni olarak aynı dili kullanır.

Express.js Hakkında

- Express.js, Node.js uygulamaları geliştirmek için yüksek derecede yapılandırılabilir bir çerçevedir ve alt düzey API'leri HTTP yardımcı yöntemleri ve middleware kullanarak soyutlar.
- Express.js, uygulama geliştirmeyi hızlandıran özellikler sunar, örneğin, kamuya açık varlıklar, şablonlar/görünümler ve yönlendirme.

Node.js Uygulama Geliştirme Süreci

- Kullanıcı arayüzünde bir seçenek seçildiğinde, bu işlem JavaScript kodunu tetikler ve istemci tarafında iş mantığını uygular.
- JavaScript uygulaması, HTTP üzerinden bir web hizmeti çağrısı yapar ve REST web hizmeti, HTTP isteğini alarak yanıt verir.

Modül Nedir?

- Modüller, belirli bir amaca hizmet eden, ilişkili ve kapsüllenmiş JavaScript kodunu içeren dosyalardır.
- Modüller, tek bir dosya veya birden fazla dosya ve klasörün bir koleksiyonu olabilir.

Modül Spesifikasyonları

- Modül spesifikasyonları, JavaScript kodunda paketler oluşturmak için kullanılan standartlar ve kurallardır.
- Node.js uygulamaları için en yaygın modül spesifikasyonları CommonJS ve ES modülleridir.

Import ve Require İfadeleri

- CommonJS modülleri, modülleri içe aktarmak için "require()" ifadesini kullanırken, ES modülleri "import()" fonksiyonunu kullanır.
- "Require" ifadesi dosyanın herhangi bir yerinde çağrılabilirken, "import" ifadesi yalnızca dosyanın başında çağrılabilir.

Senkron ve Asenkron Yükleme

- "Require" ifadeleri senkron, yani modüller sırayla yüklenir; "import" ifadeleri ise asenkron, yani modüller aynı anda işlenebilir.
- Büyük ölçekli uygulamalarda "import" ifadeleri daha hızlı çalışır.

Node.js ve Sunucu Tarafı JavaScript

- Node.js, JavaScript'i sunucu tarafında çalıştırmak için kullanılan bir programlama çerçevesidir.
- Sunucu tarafı JavaScript, istemciden gelen web hizmeti isteklerini işleyip yönlendirmek için kullanılır.

JavaScript'in Temel Özellikleri

- JavaScript, dinamik web uygulamaları oluşturmak için kullanılan bir yorumlanmış dildir.
- Modern web tarayıcıları JavaScript'i destekler ve bu dil, HTML ve CSS ile birlikte etkileşimli web uygulamaları geliştirmek için kullanılır.

İstemci Tarafı ve Sunucu Tarafı JavaScript Arasındaki Farklar

- İstemci tarafı JavaScript, kullanıcı arayüzünü HTML ve CSS ile oluşturur ve tarayıcıda çalışır.
- Sunucu tarafı JavaScript, Node.js ile çalışır ve istemciden gelen istekleri sunucu üzerinde işler.

Node.js'in Temel Özellikleri

- Node.js, JavaScript kullanan bir sunucu tarafı programlama çerçevesidir.
- Tek iş parçacıklı bir uygulama ortamıdır ve girdi/çıktı (I/O) işlemlerini olaylar aracılığıyla yönetir.

Asenkron I/O İşlemleri

- Node.js, asenkron I/O işlemleri için geri çağırma (callback) fonksiyonları kullanır. Bu, işlemlerin bloklanmadan tamamlanmasını sağlar.
- Geri çağırma fonksiyonları, işlemler tamamlandığında sonuçları işlemek için yazılır.

HTTP Modülü ile Web Sunucusu Oluşturma

- HTTP modülü kullanarak, HTTP isteklerini dinleyen ve yanıt mesajları döndüren bir uygulama geliştirebilirsiniz.
- HTTP.createServer fonksiyonu ile bir web sunucusu örneği oluşturulur ve bu sunucu bir değişkende saklanır.
- Sunucunun belirli bir portta dinlemesi için HTTP.listen fonksiyonu kullanılır; örneğin, 8080 portunu dinlemek için bu fonksiyon çağrılır.

Node.js Paketleri

- Node.js paketleri, bir veya daha fazla modülden oluşur ve her paketin bir package.json dosyası bulunur. Bu dosya, modül hakkında detayları tanımlar.
- Eğer bir modülün package.json dosyası yoksa, Node.js varsayılan olarak ana sınıfın index.js adında olduğunu varsayar.

Modül İçe Aktarma ve Dışa Aktarma

- Node.js modüllerini içe aktarmak için require fonksiyonu kullanılır. Bu fonksiyon, içe aktarılan modülü temsil eden bir nesne oluşturur.
- Her Node.js modülünün varsayılan bir exports nesnesi vardır. Bir fonksiyonu veya değeri modül dışına aktarmak için, exports nesnesine bir özellik eklenir.

Modül Kullanımı

- require fonksiyonu ile bir modül içe aktarıldığında, bu fonksiyon bir JavaScript nesnesi döner. Örneğin, today değişkeni, today modülünün bir örneğidir.
- Modülün özelliklerine erişmek için, içe aktarılan nesne üzerinden ilgili özellik çağrılır. Örneğin, today.dayOfWeek ifadesi, today modülünden dışa aktarılan bir özelliği temsil eder.

Paket Yöneticisi Nedir?

- Paket yöneticisi, modüller ve bağımlılıkları yönetmek için kullanılan bir araçtır. Yazılımın doğru çalışabilmesi için gerekli bağımlılıkları otomatik olarak bulur, yükler ve günceller.
- NPM, Node.js için varsayılan paket yöneticisidir ve iki ana işlevi vardır: komut satırı arayüzü sağlamak ve çevrimiçi bir JavaScript paketleri deposu olarak işlev görmek.

package.json Dosyası

- Her NPM paketi, projenin kök dizininde bulunan bir "package.json" dosyasına ihtiyaç duyar. Bu dosya, projenin kimlik bilgilerini ve bağımlılıklarını tanımlar.
- package.json dosyası, anahtar-değer çiftleri şeklinde proje adı ve versiyon numarası gibi bilgileri içerir.

Yerel ve Küresel Yükleme

- Yerel yükleme, yalnızca belirli bir uygulama için gerekli olan paketlerin yüklenmesini sağlar. Bu, uygulamanın bulunduğu dizinde yapılır ve varsayılan davranıştır.
- Küresel yükleme, yüklenen paketin makinedeki tüm uygulamalar tarafından erişilebilir olmasını sağlar. Ancak, farklı projelerde farklı versiyonlar kullanılıyorsa, bu durum uyumsuzluk yaratabilir.
- Arka uç teknolojileri, çeşitli sunucu türlerini ve programlama dilleri, çerçeveler ve diğer donanımlar gibi destekleyici altyapıları içerir.
- Node.js JavaScript'in sunucu tarafı bileşenidir.
- require ifadesi, uygulama kodunun herhangi bir yerinden çağrılabilir, dinamik olarak bağlanır ve senkronizedir. Import ifadesi bir dosyanın başında çağrılmalı, statik olarak bağlanmalı ve aynı zamanda modüller yürütülmeden önce yüklendiğinden senkronizedir.

- İstemci tarafı JavaScript, ön uç kullanıcı arabirimi öğelerini işlemek için kullanılır, ve sunucu tarafı JavaScript, farklı türde sunuculara ve web uygulamalarına erişimi sağlamak için kullanılır.
- Sunucu tarafı JavaScript ile Node.js uygulamaları istemciden gelen web hizmeti isteklerini işler ve yönlendirir.
- Modülünüzü içe aktaran Node.js uygulamaları için bir işlevi veya değeri kullanılabilir hale getirmek için, örtük dışa aktarma nesnesine bir özellik ekleyin.
- Çekirdek modüller minimum işlevsellik içerir, yerel modüller uygulamanız için oluşturduğunuz modüllerdir ve Node.js topluluğu üçüncü taraf modüller oluşturur.
- Yerel yükleme, yalnızca yüklenen dizin içindeki uygulamanın pakete erişebileceği anlamına gelirken, genel yükleme, makinedeki herhangi bir uygulamanın pakete erişebileceği anlamına gelir.