

Node.js ile Başlarken

Tam yığın uygulaması:

Tam yığın uygulaması aşağıdaki bileşenleri içerir:

İstemci Tarafı:

Kullanıcıya yönelik web sitesi ve mobil uygulama.

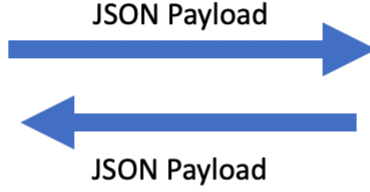
Sunucu Tarafı:

İstemci tarafından gelen her türlü isteği işler ve yanıtları istemciye geri gönderir. Günümüzde, bulut web sunucusunu, uygulama sunucusunu ve veritabanını barındırır.



Client

- HTML
- CSS
- Javascript
- React
- Etc.,



Server

- Node.js
- Java
- Python
- Etc.,

Açık Kaynak ve Çoklu Platform:

JavaScript, HTML sayfalarının istemci tarafında doğrulanması için ideal bir seçimdir çünkü çok yönlüdür ve çoklu platform uyumluluğuna sahiptir. Kullanım kolaylığının fark edilmesiyle, JavaScript sunucu tarafı kodlaması için de uyarlanmıştır ve bu da Node.js'i ortaya çıkarmıştır. Node.js, bir çalışma zamanı ortamı olarak işlev görür ve ayrıca kullanım için özel lisans gerektirmez. Ayrıca, açık kaynak doğası sayesinde Node.js ile kullanılmak üzere birçok paket ve kütüphane geliştirilmiştir. Dahası, Node.js kodu, Linux, Windows ve Mac OSX gibi çeşitli işletim sistemlerinde sorunsuz bir şekilde çalışabilir.

V8 Motoru:

Yazdığınız her bir kod parçası, işlenmesi ve makine tarafından okunabilir bir forma dönüştürülmesi gereken bir süreçten geçmelidir. Node.js alanında, JavaScript kodu Google V8 motoru kullanılarak yürütülür. Yüksek performansı ile tanınan V8, Google tarafından geliştirilen açık kaynak bir motordur ve tüm Google Chrome tarayıcılarına entegre edilmiştir. Microsoft Edge gibi modern tarayıcılar JavaScript için V8 motorunu kullanırken, Node.js sunucu tarafında V8'i kullanır.

Olay Tabanlı, Asenkron, Engellemeyen, Tek İşlemci:

Sunucu süreçleri “tek iş parçacıklı” veya “çoklu iş parçacıklı” olarak kategorize edilebilir. Tek iş parçacıklı bir ortamda, yalnızca bir komut belirli bir anda işlenirken, çoklu iş parçacıklı bir ortamda birden fazla komut aynı anda işlenebilir. Tek iş parçacıklı olmasına rağmen, Node.js asenkron ve engellemeyen doğası sayesinde performans açısından öne çıkar. Bu, bir süreç yürütülürken programın tamamlanmasını beklemesi gerekmeyi anlamına gelir. Node.js olay tabanlıdır; bu, ağdan okuma veya bir veritabanına veya dosya sistemine erişim gibi bir girdi/çıkı (I/O) işlemi gerçekleştirdiğinde bir olayın tetiklendiği anlamına gelir. İş parçasını engellemek ve yanıt beklerken işlemci zamanını tüketmek yerine, Node.js yanıt alındığında veya ilgili olay gerçekleştiğinde işlemleri yeniden başlatır. Bu engellemeyen davranış, sunucunun yanıt verebilir durumda kalmasını ve çoklu görevleri eşzamanlı olarak yönetmesini sağlar, tıpkı çoklu iş parçacıklı bir ortam gibi.

JSON Yüklü

JSON, “anahtar-değer çiftleri” olarak biçimlendirilmiş JavaScript Nesne Notasyonu’nu ifade eder. Yük, istemci ile sunucu arasında iletilen veriyi temsil eder. İstemci sunucuya veri göndermesi gerektiğinde, bunu aşağıdaki örnekte gösterildiği gibi bir JSON nesnesi şeklinde yapar:

```
{
  "name": "John",
  "age": "24",
  "email": "johnparker@gmail.com"
}
```

Yukarıdaki nesne bir JSON'dur. Bu veri istek parçası olarak gönderildiğinde, değerleri request.name gibi ifadelerle kolayca çıkarılabilir.

Benzer şekilde, yanıt da JSON formatında gönderilir. Aşağıda bir örnek verilmiştir:

```
{
  "status": "ok",
  "message": "Successfully added"
}
```

Express Framework

Node.js, bir sunucu oluřturmak iin paketler saėlarken, Express framework' API'ler ve u noktalar oluřturma srecini basitleřtirir. Bir API u noktası, istemciden sunucuya yapılan bir isteėin belirli bir giriř noktasıdır.

Sonraki adımlar

Bu kursta, Express erevesini kullanarak Node.js ile sunucu tarafı kodlamayı ėreneceksiniz.



Skills Network