PROGRAMLAMA DİLLERİ MODÜLÜ (JAVASCRIPT)

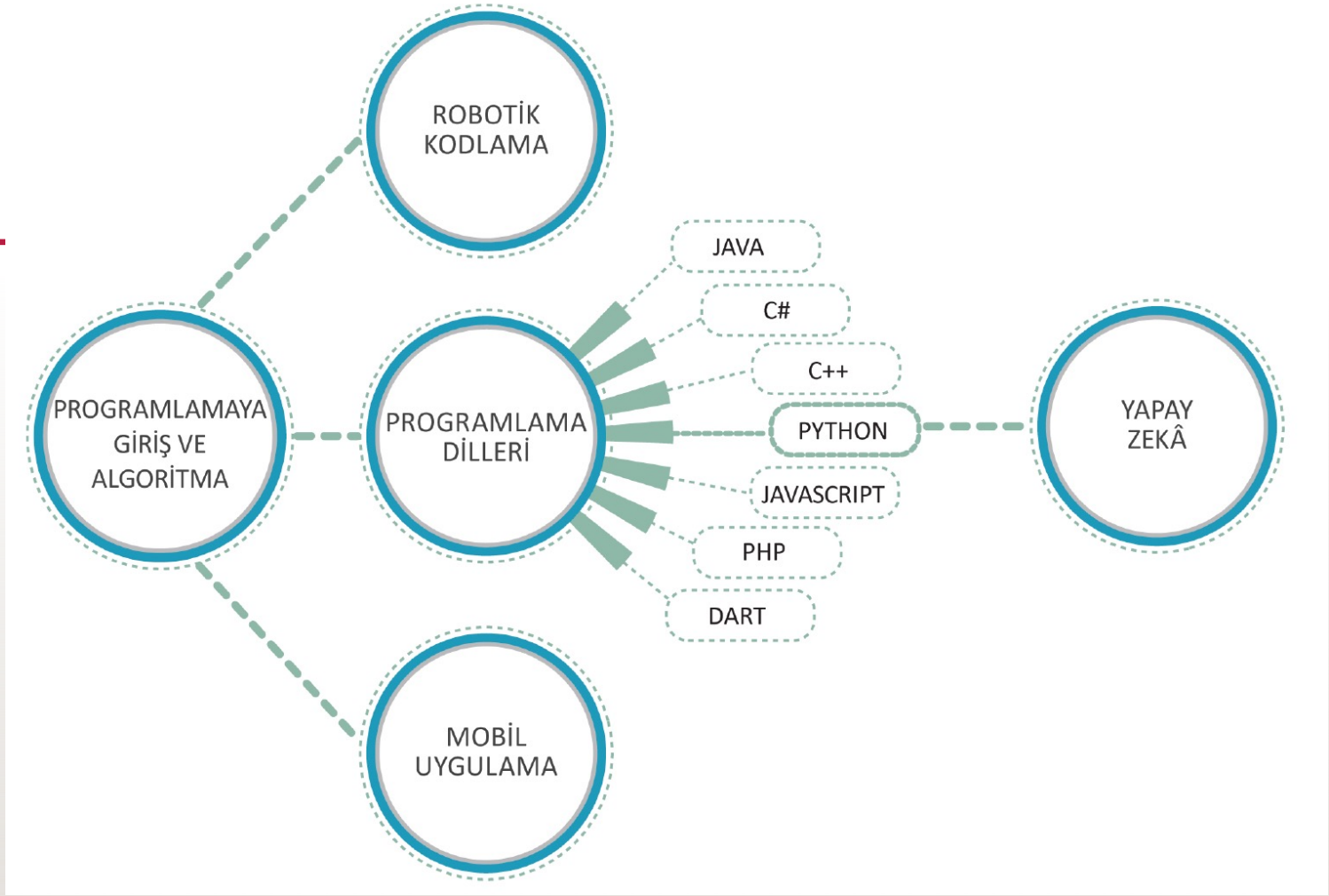
HAZ: ONUR ALTUNTAŞ

DERS SEÇİM KRİTERİ

Programlama dilleri modülünde; Java, C++, C#, Dart, PHP, Javascript, Python programlama dillerinden birisi seçilir.

Programlamaya giriş ve algoritma modülü alınmadan; robotik kodlama, mobil uygulama geliştirme ve programlama dilleri modülleri seçilemez. Programlama dilleri modülü alınmadan yapay zekâ uygulamaları modülü seçilemez.

DERS SEÇİM KRİTERİ ŞEMASI



ÜNİTE, KAZANIM SAYISI VE SÜRE TABLOSU
JAVASCRIPT ÜNİTE DERS SAATLERİ

MODÜL	ÜNİTE	KAZA-NIM SAYISI	DERS SAATİ	YÜZDE ORANI (%)
	1. Ünite: Javascript Temelleri	8	18	25
	2. Ünite: Javascript Fonksiyonları ve Nesne Tabanlı Programlama	11	18	25
	3. Ünite: Javascript ile Web Geliştirme	14	36	50

STRATEJİ

ÖĞREN!

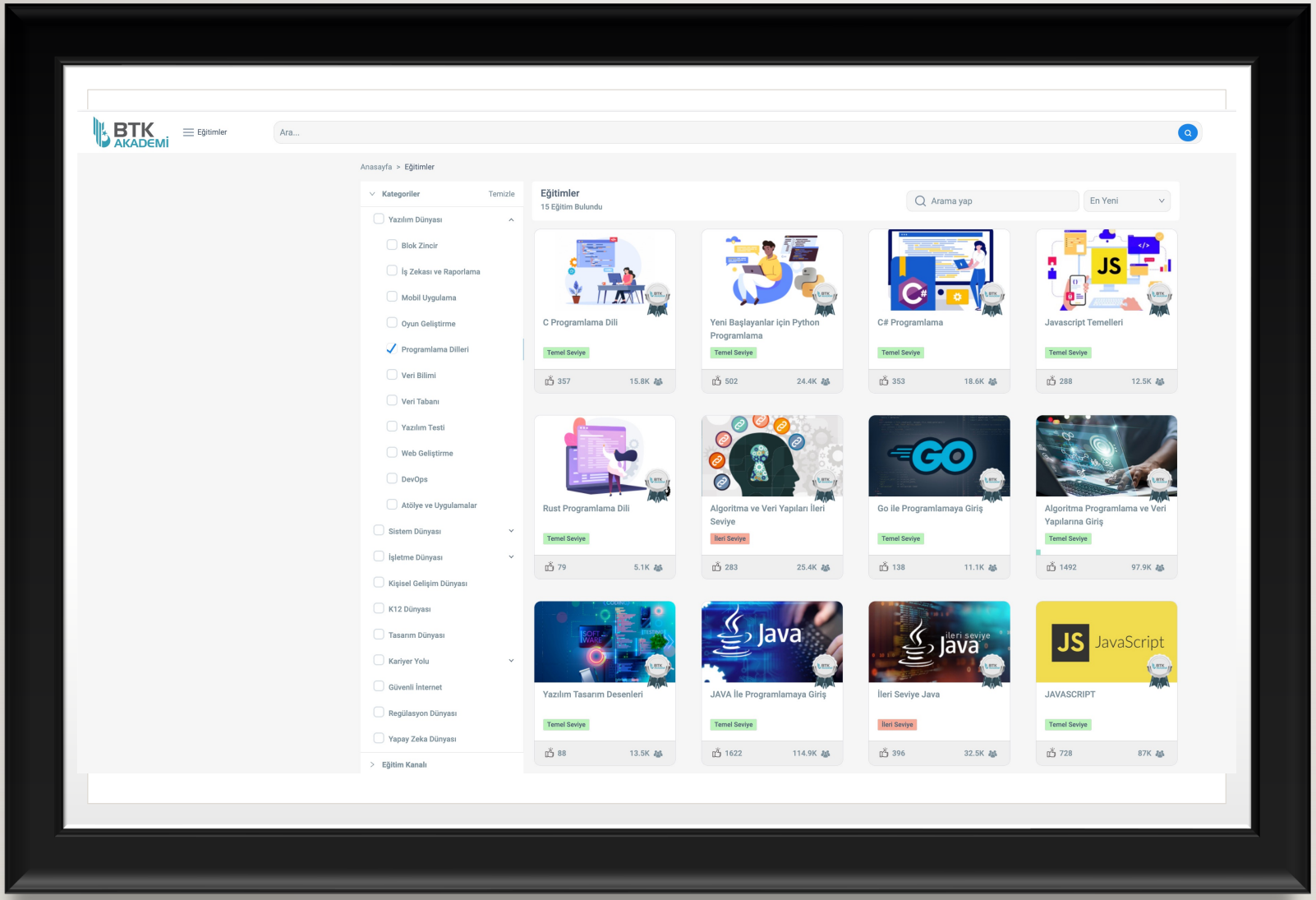
ÖĞRET!

UYGULAT!

YÖNTEM

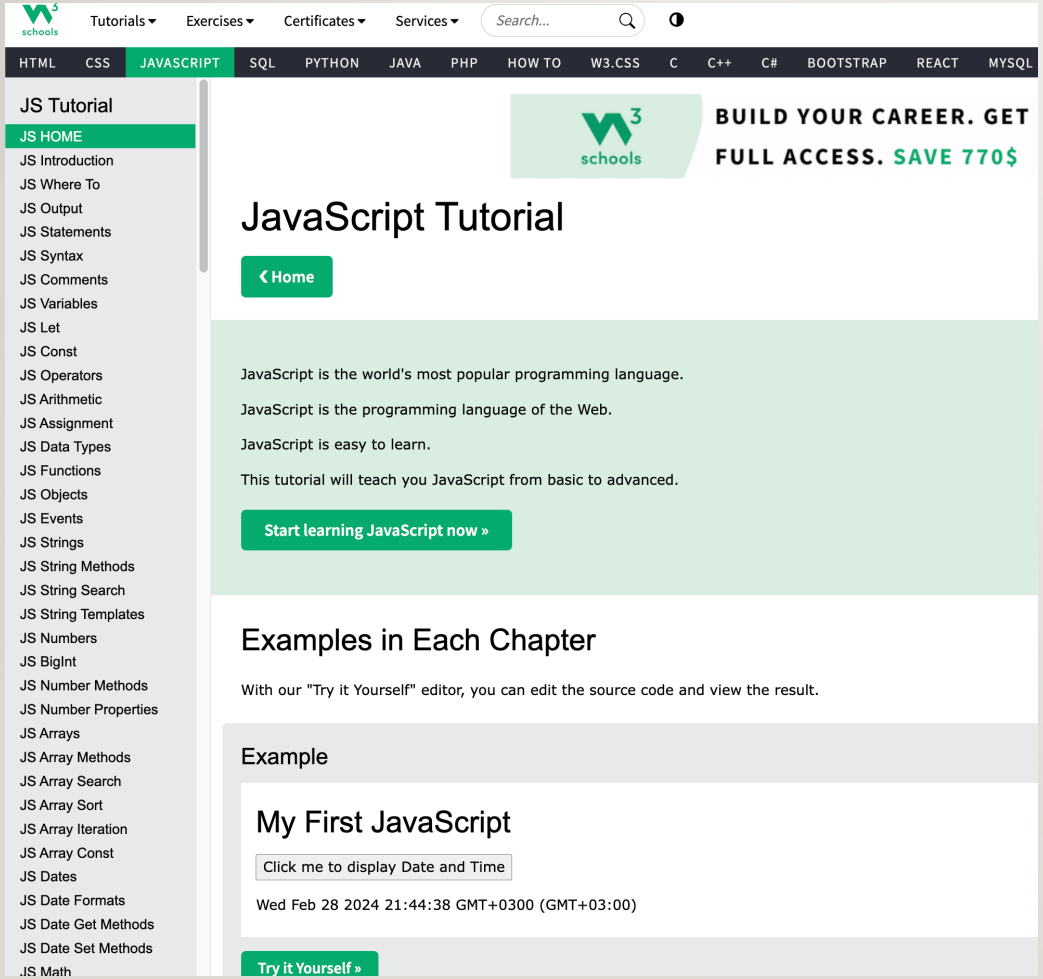
EZBER DEĞİL SÜREKLİ UYGULAMA!

***HEDEF SENARYO İÇİN GEREKLİ* KODU BULMAK VE YAZMAK**



ÖĞRENME KAYNAKLARI

w3schools.com



ÖĞRENME KAYNAKLARI

JAVASCRIPT MODÜLÜNÜN AMACI

- Javascript programlama dili temellerini, fonksiyonları, nesne tabanlı programlamayı, asenkron programlamayı, DOM ve Html'i birlikte kullanmayı, modüler kodlama işlemlerini öğrenerek, web geliştirme ve programlama
- becerilerinin geliştirilmesi amaçlanmaktadır.

JAVASCRIPT

ÜNİTE - KAZANIM VE AÇIKLAMALARI

1. ÜNİTE: JAVASCRIPT PROGRAMLAMA DİLİ TEMELLERİ

Ünite Açıklaması

- Bu ünitede; web geliştirme süreçlerinin işleyişi, Javascript programlama dilinin web sitelerindeki rolü, Javascript programlama dilinin terim ve kavramları üzerinde durulur. Javascript programlama dilinde değişkenler, veri tipleri, koşullu ifadeler, karşılaştırma operatörleri ve mantıksal operatörler kavramları anlatılır.

1. ÜNİTE: JAVASCRIPT PROGRAMLAMA DİLİ TEMELLERİ

Değerler

- Sabır (Kazanım 1.1., 1.2., 1.3., 1.6.)
- Özdenetim (Kazanım 1.4., 1.5., 1.6.)
- Sorumluluk (Kazanım 1.4., 1.6., 1.7., 1.8.)
- Yardımseverlik (Kazanım 1.6.)

Alan Becerileri

- Soyutlama Becerisi (Kazanım 1.1., 1.2., 1.3.)
- Mantıksal Düşünme Becerisi (Kazanım 1.4., 1.5., 1.6., 1.7.)
- Algoritmik Düşünme Becerisi (Kazanım 1.4., 1.5., 1.6., 1.7., 1.8
- Kodlama, Programlama Becerisi (Kazanım 1.1., 1.2., 1.3., 1.4., 1.5., 1.6., 1.7., 1.8.)

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Proje Planlaması

- Bu aşamada, web sitesinin amacını, hedef kitlesini ve ana özelliklerini belirleriz.
- "Ne yapmak istiyoruz?", "Kim için yapmak istiyoruz?" ve "Bunu nasıl yapacağız?" sorularına cevap ararız.
- Aynı zamanda, projenin zaman çizelgesi ve bütçesi gibi önemli detayları da planlarız.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Tasarım

- Planlama aşamasından sonra sıra, web sitemizin nasıl görüneceğine karar vermeye gelir. Bu aşama, bir evin iç dekorasyonunu tasarlamak gibidir. Web sitesinin genel görünümü, renkler, düzen, menüler ve butonlar gibi unsurlar bu aşamada tasarlanır.

- Tasarımdan önce web sitelerini inceletin.

- Kağıt üzerinde tasarım yaptırın.

- **Video Dersler**: YouTube veya diğer platformlardaki kaliteli web geliştirme derslerini kullanın. Görsel öğrenme, öğrencilerin konseptleri daha iyi anlamasına yardımcı olur.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Kodlama

- Bu aşama, bir evin inşa edilmesi gibidir.
- Web geliştiricileri, HTML, CSS, JavaScript gibi dilleri kullanarak siteyi kodlarlar.
- Eğer site dinamikse (kullanıcı etkileşimine göre içeriği değişen), PHP, Python, JavaScript'in bir kütüphanesi olan Node.js gibi sunucu tarafı teknolojileri de kullanılır.
- Veri saklamak için veritabanları devreye girer.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Canlı Kodlama Uygulamaları: w3school, CodePen, JSFiddle gibi canlı kodlama platformlarını kullanarak öğrencilere gerçek zamanlı olarak kodun nasıl çalıştığını gösterin.

Gerçek Örnekleri Kullanın

- Öğrencilerin günlük hayatta kullandıkları web sitelerini örnek olarak gösterin ve bu sitelerin hangi teknolojilerle yapıldığını tartışın.
- Gerçek dünya problemlerini çözümlemek için web teknolojilerini nasıl kullanabileceklerini gösterin.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Test

Gerçek hayattan örneklerle, testin atlanmaması gereken kritik bir adım olduğunu vurgulayın.

- Bir ürünü piyasaya sürmeden önce nasıl test ediyorsak, web sitesini de yayınlanmadan önce test etmemiz gerekir.
- Bu aşama, yazılım hatalarını, bağlantı sorunlarını ve diğer teknik aksaklıkları bulmak için yapılır.
- Ayrıca, web sitesinin farklı tarayıcılarda ve cihazlarda düzgün çalışıp çalışmadığı kontrol edilir.

Etkileşimli Öğrenme

- Test sürecini daha etkileşimli ve eğlenceli hale getirmek için oyunlaştırma tekniklerinden yararlanın.
- Örneğin herhangi bir web sitesini inceleterek en çok hatayı bulan veya en yaratıcı test senaryosunu yazan öğrencilere ödüller verin.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Alan adı ve hosting kavramlarını açıklayın

- **Temel Kavramlar**: Bir web sunucusunun ne olduğunu ve bir alan adının (domain) internet üzerindeki bir adres gibi nasıl çalıştığını açıklayın.

- **Alan adı ve Hosting**: Alan adlarının nasıl satın alınabileceği ve bir hosting hizmetiyle nasıl ilişkilendirileceği konusunda temel bilgiler verin.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Dağıtım

- Bu aşama, üretilen bir ürünün raflara yerleştirilmesi gibidir.
- Web sitesi, bir web sunucusuna yüklenir ve artık herkes tarafından erişilebilir hale gelir.
- Ancak işimiz burada bitmez; web sitesinin düzenli olarak güncellenmesi, performansının izlenmesi ve olası sorunların çözülmesi gerekir.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

A) WEB GELİŞTİRME SÜREÇLERİNİN TEMEL ADIMLARI OLAN, PROJE PLANLAMASI, TASARIM, KODLAMA, TEST VE DAĞITIM SÜREÇLERİ AÇIKLANIR.

Basit Hosting Servisleri

Ücretsiz Hosting Servislerini Tanıtın: Öğrencilere, GitHub Pages, Netlify, Vercel gibi ücretsiz ve kullanımı kolay web hosting servislerini tanıtın.

Pratik Yapma: Bir web projesini bu platformlara nasıl yükleyecekleri konusunda adım adım rehberlik edin.

Örneğin, bir GitHub hesabı oluşturup, bir projeyi GitHub Pages üzerinden yayınlanabileceğini anlatın.

1.1. WEB GELİŞTİRME SÜREÇLERİNİN İŞLEYİŞ ŞEKİLLERİNİ ANLAR.

B) SÜREÇLER ARASINDA BAĞLANTILAR KURULARAK TASARIMIN KODLAMAYA DÖNÜŞME AŞAMASI VURGULANIR.

Tasarımdan Kodlamaya Geçiş: Öğrencilere, tasarlanan sayfanın statik bir ön yüz web sayfasına (HTML ve CSS kullanarak) nasıl dönüştürüleceğini adım adım gösterin. Örneğin, bir başlık nasıl kodlanır.

Projeye Dayalı Öğrenme

- **Küçük Grup Projeleri**: Öğrencileri küçük gruplara ayırın ve her gruba bir tasarım verin. Grupların bu tasarımı kullanarak basit bir web sayfası oluşturmalarını isteyin. Bu, tasarımın kodlamaya nasıl dönüştürüleceğini pratik yaparak öğrenmelerini sağlar.

1.2. JAVASCRİPT PROGRAMLAMA DİLİNİN WEB SİTELERİNDEKİ ROLÜNÜ KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNİN WEB SİTELERİNDEKİ KULLANIM AMACI AÇIKLANIR. KULLANICI ETKİLEŞİMİNİ ARTIRMAK VE SAYFALARI DİNAMİK HÂLE GETİRMEK İÇİN KULLANILDIĞI VURGULANIR.

JavaScript'in Temel Kavramlarını Tanıtın

- JavaScript'in ne olduğunu ve web geliştirmedeki rolünü açıklayın. HTML ve CSS ile birlikte çalışarak nasıl daha interaktif kullanıcı deneyimleri oluşturduğunu vurgulayın.

Interaktif Örneklerle Öğretin

- Basit JavaScript kodları yazarak, bir butona tıklama gibi kullanıcı etkileşimlerini nasıl yönetebileceklerini gösterin.

https://www.w3schools.com/js/tryit.asp?filename=tryjs_myfirst

1.3. JAVASCRİPT PROGRAMLAMA DİLİNİN TERİM VE KAVRAMLARINI TANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDE KULLANILAN TEMEL TERİMLER VE KAVRAMLAR AÇIKLANIR

JS 101

- **Basit Tanımlar**: JavaScript'in temel terimlerini ve kavramlarını tanımlayarak başlayın. Değişkenler, fonksiyonlar, diziler, döngüler, koşullu ifadeler, nesneler ve diziler

1.3. JAVASCRİPT PROGRAMLAMA DİLİNİN TERİM VE KAVRAMLARINI TANIR.

B) DEĞİŞKEN, FONKSİYON, DİZİ GİBİ KAVRAMLARIN JAVASCRİPT PROGRAMLAMA DİLİNDEKİ ÖNEMİ VURGULANIR.

Değişkenler

- **Tanım**: Değişkenlerin bir değeri saklamak için kullanılan etiketler olduğunu açıklayın. Gerçek hayattan bir örnek vererek başlayabilirsiniz, örneğin bir kutuya koyulan bir nesneyi düşünün, bu kutunun üzerine yazılan isim değişken adıdır ve içindeki nesne değişkenin değeridir.
- **Kod Örneği**: Basit bir değişken tanımlama örneği gösterin:

```
let kutununIcerigi = "elma";
```

1.3. JAVASCRİPT PROGRAMLAMA DİLİNİN TERİM VE KAVRAMLARINI TANIR.

B) DEĞİŞKEN, FONKSİYON, DİZİ GİBİ KAVRAMLARIN JAVASCRİPT PROGRAMLAMA DİLİNDEKİ ÖNEMİ VURGULANIR.

Fonksiyonlar

- **Tanım**: Fonksiyonların, belirli bir görevi yerine getiren kod blokları olduğunu açıklayın

Kod Örneği: Basit bir fonksiyon örneği yazın ve açıklayın:

Butona basınca «Merhaba» yazısı ekrana gelsin

https://onuraltuntas61.w3spaces-preview.com/function.html

1.3. JAVASCRİPT PROGRAMLAMA DİLİNİN TERİM VE KAVRAMLARINI TANIR.

B) DEĞİŞKEN, FONKSİYON, DİZİ GİBİ KAVRAMLARIN JAVASCRİPT PROGRAMLAMA DİLİNDEKİ ÖNEMİ VURGULANIR.

Diziler

- **Tanım**: Dizilerin, birden çok değeri tek bir değişkende saklamak için kullanıldığını açıklayın. Bir kitaplıkta bulunan kitaplar gibi, her bir kitabın (elemanın) kendine ait bir yeri (indeksi) olduğunu ve bu kitapların (elemanların) bir arada tutulduğunu anlatabilirsiniz.

```
let meyveler = ["elma", "muz", "çilek"];
```

- **Kod Örneği**: Basit bir dizi tanımlayın ve elemanlara nasıl erişileceğini gösterin:

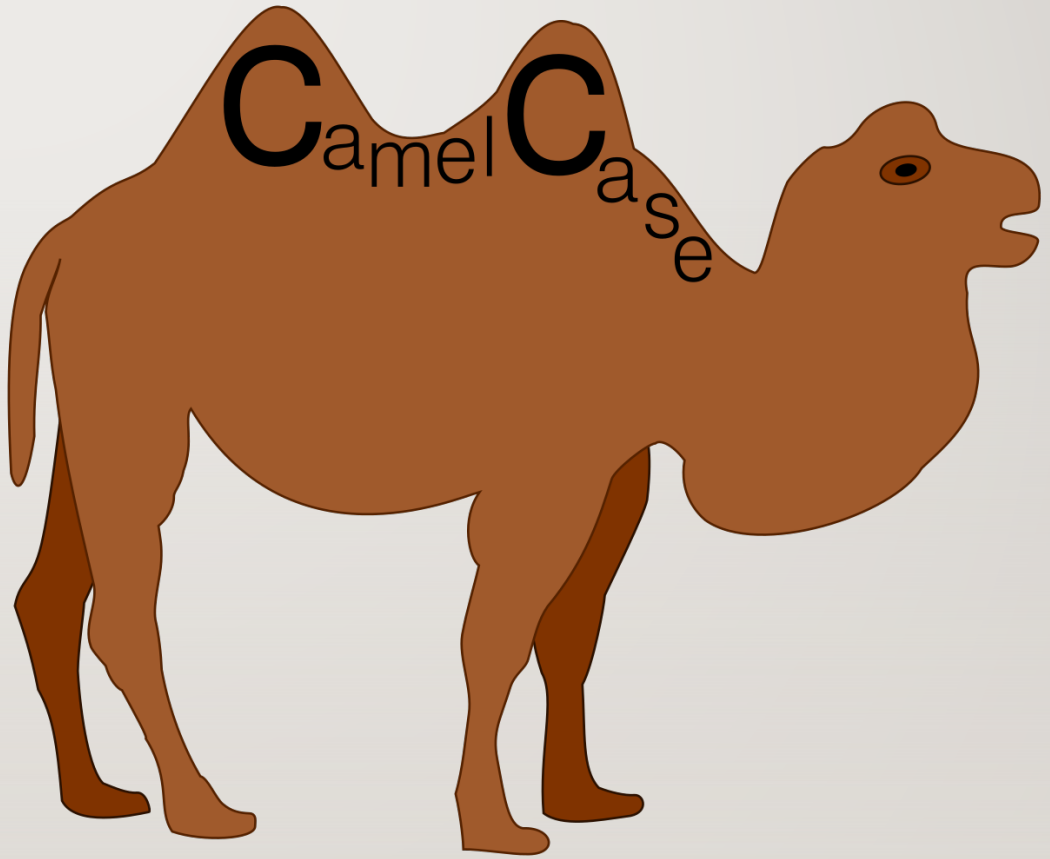
https://onuraltuntas61.w3spaces-preview.com/dizi.html

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

A) DEĞİŞKEN TANIMLAMA VE DEĞİŞKENLERİN KULLANIM ŞEKLİ AÇIKLANIR.

İsimlendirme Kuralları ve İpuçları

- Değişken isimleri sayı ile başlayamaz.
- Boşluk veya özel karakter içeremez (ancak _ ve $ kullanılabilir).
- Anlamlı isimler seçmek kodunuzu daha okunabilir hale getirir.
- JavaScript'te genellikle camelCase kullanılır: İlk kelime küçük harfle başlar, sonraki kelimelerin ilk harfi büyük olur (ornekDegisken).



1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

B) DEĞİŞKENLERİN TÜRLERİ AÇIKLANIR VE KULLANILIR.

JavaScript'te değişken tanımlamak için üç kelime kullanabiliriz: var, let, const.

- **var**: Bu, eskiden beri kullanılan bir yöntem. Ancak biraz kafa karıştırıcı olabilir, bu yüzden genellikle let ve const kullanmayı öneririz.
- **let**: Değişkenin değerinin değişebileceği durumlar için kullanılır.
- **const**: Değişkenin değerinin hiç değişmeyeceği, yani sabit kalacağı durumlar için kullanılır.

```
let okulNumaran = 123;

const okulAdi = "Lise Adı";
```

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

C) DEĞİŞKENLERİN YAŞAM DÖNGÜSÜ ANLATILIR. GLOBAL VE LOKAL DEĞİŞKENLER ARASINDAKİ FARKLAR AÇIKLANIR.

Değişkenlerin Yaşam Döngüsü: Bir Hikaye Olarak

1. **Doğum (Tanımlama):** Bir değişken, ona bir değer atandığında veya ilk kez tanımlandığında "doğar". Bu, değişkenin hikayesinin başlangıcıdır.

Örnek:

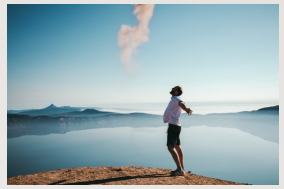
let kitap = "Matematik";

*Burada kitap adında bir değişken "doğar" ve "Matematik" değerine sahip olur.

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

C) DEĞİŞKENLERİN YAŞAM DÖNGÜSÜ ANLATILIR. GLOBAL VE LOKAL DEĞİŞKENLER ARASINDAKİ FARKLAR AÇIKLANIR.

Değişkenlerin Yaşam Döngüsü: Bir Hikaye Olarak



1. **Hayat (Kullanım):** Değişken tanımlandıktan sonra, kodun çeşitli yerlerinde kullanılabilir. Bu, değişkenin "yaşadığı" zamandır.

Örnek:

console.log(kitap);

*Burada kitap değişkeni kullanılır, yani "yaşar".

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

C) DEĞİŞKENLERİN YAŞAM DÖNGÜSÜ ANLATILIR. GLOBAL VE LOKAL DEĞİŞKENLER ARASINDAKİ FARKLAR AÇIKLANIR.

Değişkenlerin Yaşam Döngüsü: Bir Hikaye Olarak

1. **Son (Erişilemezlik):** Bir değişkenin kapsamı dışına çıkıldığında, artık erişilemez hale gelir ve yaşam döngüsü sona erer.

Örnek:

Bir fonksiyon içinde tanımlanan let ders = "Biyoloji"; değişkeni, fonksiyonun dışında erişilemez hale gelir.

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

C) DEĞİŞKENLERİN YAŞAM DÖNGÜSÜ ANLATILIR. GLOBAL VE LOKAL DEĞİŞKENLER ARASINDAKİ FARKLAR AÇIKLANIR.

Global ve Lokal Değişkenler: Bir

- **Global Değişkenler:** Tüm evde tanınan ve herkesin erişebileceği bir eşya gibidir. Programın her yerinden erişilebilirler.
  - Örnek: Evin dış kapısı gibi, salondaki televizyon herkes tarafından görülebilir ve kullanılabilir.

- **Lokal Değişkenler:** Belirli bir oda (fonksiyon veya blok) ile sınırlı olan eşyalar gibidir. Yalnızca o odada erişilebilirler.
  - Örnek: Bir oyun odasındaki özel bir oyun gibi, yalnızca o odada olanlar tarafından oynanabilir.

1.4. DEĞİŞKENLERİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

C) DEĞİŞKENLERİN YAŞAM DÖNGÜSÜ ANLATILIR. GLOBAL VE LOKAL DEĞİŞKENLER ARASINDAKİ FARKLAR AÇIKLANIR.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    DEĞİŞKENLER
</body>

<script>

let evSahibi = "Ayşe"; // Global değişken, tüm evde (programda) bilinir.

function parti(){
  let misafir = "Ali"; // Lokal değişken, yalnızca bu fonksiyon (oda) içinde bilinir.
  console.log(evSahibi); // Çalışır, çünkü ev sahibi her yerde bilinir.
  console.log(misafir); // Çalışır, çünkü misafir şu anda bu odada (fonksiyon).
}

console.log(evSahibi); // Çalışır, global değişken.
console.log(misafir); // Hata verir, misafir yalnızca partide (fonksiyonda) tanınır.

</script>
</html>
```

https://onuraltuntas61.w3spaces-preview.com/degisken.html

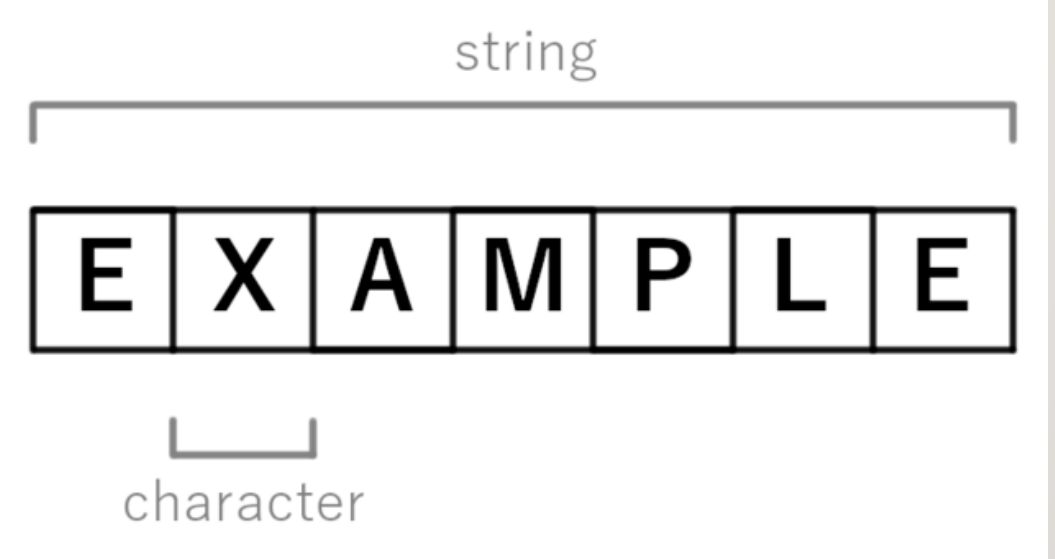
1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ VERİ TİPLERİ (STRİNG, NUMBER, BOOLEAN, OBJECT, VB.) AÇIKLANIR.

1. String (Metin)
Metin veya kelimeleri ifade eder. Tırnak işaretleri (" " veya ' ') arasına yazılır.

Örnek: Birinin adı, bir kitap başlığı veya bir cümle. "Merhaba, Dünya!" veya 'Ali’.

Günlük Hayat Örneği: İsminiz bir metindir. Arkadaşınıza yazdığınız bir not da metindir.



1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ VERİ TİPLERİ (STRİNG, NUMBER, BOOLEAN, OBJECT, VB.) AÇIKLANIR.

2. Number (Sayı)

Nasıl Anlatılır: Sayısal değerleri ifade eder. Tam sayılar ve ondalık sayılar olabilir.
Örnek: 5, 3.14, -123.

Günlük Hayat Örneği: Yaşınız, evinizin numarası veya bir elma sayısı birer sayıdır.

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

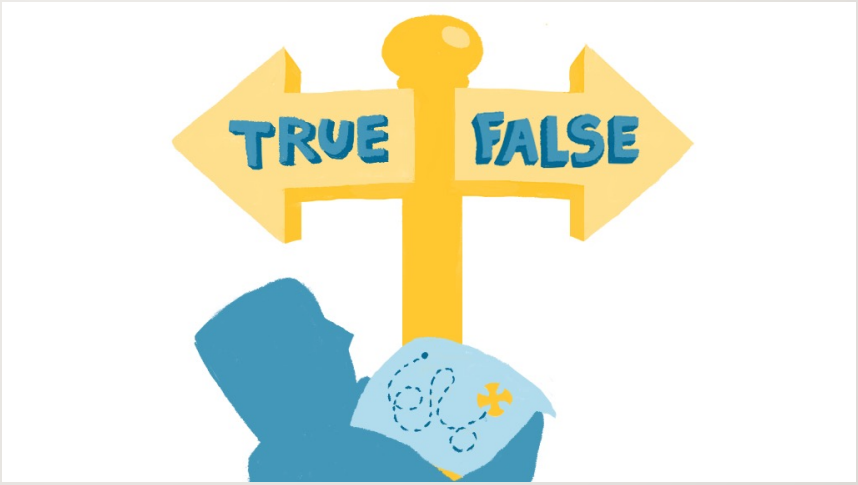
A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ VERİ TİPLERİ (STRİNG, NUMBER, BOOLEAN, OBJECT, VB.) AÇIKLANIR.

3. Boolean (Mantıksal)

Nasıl Anlatılır: Yalnızca iki değer alabilir: true (doğru) veya false (yanlış).

Örnek: Bir ışığın açık olup olmadığı, bir cevabın doğru veya yanlış olması.

Günlük Hayat Örneği: "Bugün hava güneşli mi?" sorusuna vereceğiniz "evet" veya "hayır" cevabı, aslında birer boolean değeridir



1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ VERİ TİPLERİ (STRİNG, NUMBER, BOOLEAN, OBJECT, VB.) AÇIKLANIR.

4. Object (Nesne)

Nasıl Anlatılır: Birden fazla değeri bir arada tutmaya yarar. Özellikleri (properties) ve metodları (functions) olabilir.

Örnek: Bir kişi nesnesi, isim, yaş ve adres gibi özelliklere sahip olabilir.

Günlük Hayat Örneği: Bir okul çantası düşünün. Bu çanta, kitaplar, defterler ve kalemler gibi çeşitli eşyaları içinde barındırır. Her bir eşya, çantanın bir özelliği gibidir.

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ VERİ TİPLERİ (STRİNG, NUMBER, BOOLEAN, OBJECT, VB.) AÇIKLANIR.

5. Array (Dizi)

Nasıl Anlatılır: Birden çok değeri sıralı bir şekilde tutar. Her bir elemanına indeks numarası ile erişilebilir.

Örnek: let notlar = [90, 80, 85]; Bu, üç notun bir listesidir.

Günlük Hayat Örneği: Bir alışveriş listesi, bir diziye benzer. Listede, almanız gereken her şey sıralı bir şekilde yer alır.

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

B) VERİ DÖNÜŞÜMÜ VE TİP DÖNÜŞÜMÜNÜN NASIL YAPILACAĞI AÇIKLANIR.

Manuel Tip Dönüşümü

Bazen verileri bizim manuel olarak dönüştürmemiz gerekir. Bu, veri türlerini bilinçli olarak birbirine çevirmek için kullanılır.

Sayıdan Metne Dönüşüm:

```
let sayi = 5;
let metin = sayi.toString();                    // "5 "
```

Metinden Sayıya Dönüşüm:

```
let metin = "123";
let sayi = parseInt(metin);                     // 123
```

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

B) VERİ DÖNÜŞÜMÜ VE TİP DÖNÜŞÜMÜNÜN NASIL YAPILACAĞI AÇIKLANIR.

Boolean Dönüşümü

Verileri "doğru" (true) ya da "yanlış" (false) boolean değerlerine dönüştürmek de mümkündür.

```
let deger = 1;
let booleanDeger = Boolean(deger);        // true
```

***Bu konuları anlatırken, örneklerin üzerinden gitmek ve öğrencilere kendi örneklerini denemeleri için zaman tanımak, kavramların daha iyi anlaşılmasını sağlar.**

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

B) VERİ DÖNÜŞÜMÜ VE TİP DÖNÜŞÜMÜNÜN NASIL YAPILACAĞI AÇIKLANIR.

Neden Tip Dönüşümü Yapılır?

1. Kullanıcı Girdisini İşlemek

- Kullanıcılardan alınan girdiler genellikle metin (string) formundadır. Ancak bu girdilerle matematiksel işlemler yapmak istiyorsak, bu verileri sayısal bir tipe dönüştürmemiz gerekir.

Örnek: Bir web sitesinde iki sayıyı toplamak için kullanıcıdan girdi istiyorsunuz. Kullanıcı bu sayıları bir form aracılığıyla girer. Formdan alınan bu girdiler metin şeklindedir. Toplama işlemi yapabilmek için bu metinleri sayılara (number) dönüştürmeniz gerekir.

https://onuraltuntas61.w3spacespreview.com/veri_donusum.html

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

B) VERİ DÖNÜŞÜMÜ VE TİP DÖNÜŞÜMÜNÜN NASIL YAPILACAĞI AÇIKLANIR.

Neden Tip Dönüşümü Yapılır?

Kullanıcıya Bilgi Sunmak

- Bazen, işlediğiniz sayısal verileri kullanıcıya daha anlaşılır bir formatta sunmak isteyebilirsiniz. Bu durumda, sayıları metne (string) dönüştürebilirsiniz.

Örnek: Kullanıcının yaşıyla ilgili bir hesaplama yapıyorsunuz ve sonucu bir cümle içinde göstermek istiyorsunuz.

https://onuraltuntas61.w3spaces-preview.com/veri_donusumu_sayi_string.html

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

B) VERİ DÖNÜŞÜMÜ VE TİP DÖNÜŞÜMÜNÜN NASIL YAPILACAĞI AÇIKLANIR.

Mantıksal Kontroller Yapmak

Bazen, bir değerin varlığını veya belirli bir koşulu kontrol etmek için boolean tipe dönüşüm yapabilirsiniz.

Örnek: Bir formda kullanıcıdan alınan bir girdinin dolu olup olmadığını kontrol etmek.

https://onuraltuntas61.w3spaces-preview.com/veri_donusumu_boolean.html

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

C) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NULL VE UNDEFİNED GİBİ ÖZEL DEĞERLER VURGULANIR.

6. Undefined ve Null

Nasıl Anlatılır:
•undefined, bir değişkenin değeri atanmamışsa bu değere sahip olur.
•null, bir değişkenin hiçbir değere sahip olmadığını belirtmek için kullanılır.

Örnek: Bir değişkene değer atamadan önce undefined, bilinçli olarak boş bırakılmak istenen bir değişkene null atanır.

Günlük Hayat Örneği: Bir sınavdan aldığınız not henüz belli değilse, bu "undefined" olarak düşünülebilir. Eğer bir sınav yapılmadıysa ve bu yüzden bir not yoksa, bu "null" olarak düşünülebilir.

Boş kavanoz örneği verin.

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

C) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NULL VE UNDEFİNED GİBİ ÖZEL DEĞERLER VURGULANIR.

undefined: Değer Atanmamış Değişken:

```
let kutu;
console.log(kutu);
```

// çıktı: undefined

null

null, bir değişkenin değerinin bilinçli olarak "hiçbir şey" veya "boş" olarak belirlendiği durumları ifade eder. Programcı tarafından manuel olarak atanır ve bir değişkenin hiçbir değere sahip olmadığını gösterir.

```
let bosDeger = null;
console.log(bosDeger);
```

// çıktı: null

1.5. JAVASCRİPT PROGRAMLAMA DİLİ İÇERİSİNDE KULLANILAN FARKLI VERİ TİPLERİNİ KAVRAR.

C) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NULL VE UNDEFİNED GİBİ ÖZEL DEĞERLER VURGULANIR.

Karşılaştırma

null ve undefined, JavaScript'te "değer yok" anlamına gelir, ancak:

•undefined genellikle JavaScript tarafından bir değişkene otomatik olarak atanır.

•null ise programcı tarafından bir değişkene bilinçli olarak atanır ve bir değişkenin kasıtlı olarak boş olduğunu gösterir.
Bu iki özel değerin farkını anlamak, özellikle koşullu ifadeler yazarken veya veri türlerini kontrol ederken önemlidir.

1.6. PROBLEMLERDE KARŞILAŞILAN FARKLI SENARYOLARDA KOŞULLU İFADELERİ KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİ İLE İF-ELSE İFADELERİ VE SWİTCH-CASE YAPILARININ KULLANIMI AÇIKLANIR.

if-else ve switch-case yapıları bir programın farklı koşullara göre farklı yollar izlemesini sağlar. Yani bir nevi yol ayrımı gibidirler.

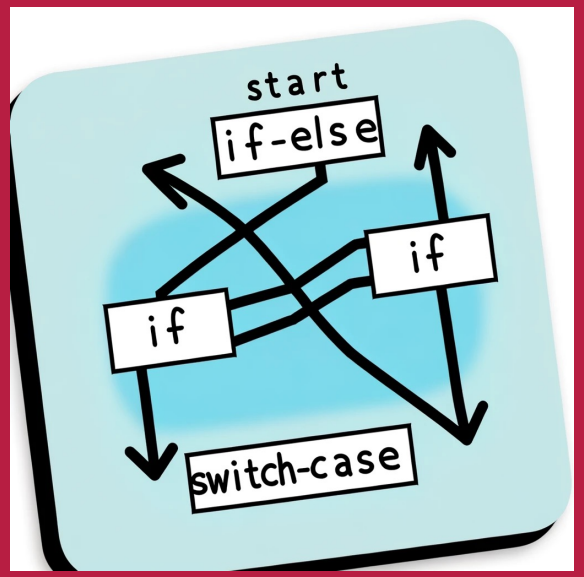
if-else Yapısı

Diyelim ki bir arkadaşınıza film izlemeyi teklif ediyorsunuz. Eğer arkadaşınız "Evet" derse, sinemaya gideceksiniz; "Hayır" derse, evde oyun oynayacaksınız. Bu karar verme süreci, if-else yapısına benzer.

```
if (arkadasinCevabi === "Evet") {
  console.log("Sinemaya gidelim.");
} else {
  console.log("Evde oyun oynayalım.");
}
```

1.6. PROBLEMLERDE KARŞILAŞILAN FARKLI SENARYOLARDA KOŞULLU İFADELERİ KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİ İLE İF-ELSE İFADELERİ VE SWİTCH-CASE YAPILARININ KULLANIMI AÇIKLANIR.



switch-case Yapısı

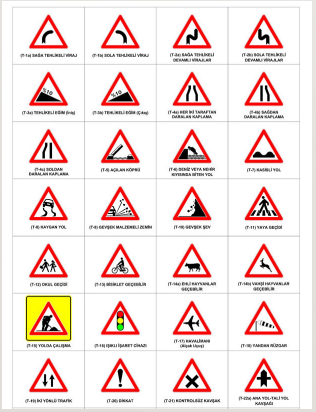
Şimdi de farklı bir senaryo düşünelim. Bir yemek menüsünden seçim yapacaksınız. "Pizza" seçerseniz, pizza siparişi verilecek; "Hamburger" seçerseniz, hamburger siparişi verilecek; "Salata" seçerseniz, salata siparişi verilecek.

```
let secim = "Pizza";

switch (secim) {
  case "Pizza":
    console.log("Pizza siparişi verildi.");
    break;
  case "Hamburger":
    console.log("Hamburger siparişi verildi.");
    break;
  case "Salata":
    console.log("Salata siparişi verildi.");
    break;
  default:
    console.log("Geçerli bir seçim yapmadınız.");
}
```

1.7. KARŞILAŞTIRMA OPERATÖRLERİNİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDE KARŞILAŞTIRMA OPERATÖRLERİ (==, ===, !=, <, >, VB.) KULLANILARAK DEĞİŞKENLERİN KARŞILAŞTIRMASI AÇIKLANIR.



ÖĞRENCİLERE SORALIM. NEDEN İŞARETLERE İHTİYAÇ DUYARIZ?

Günlük Hayattaki Örnekler:

1. **Trafik İşaretleri:**
2. **Okulda Kullanılan İşaretler:**
3. **Uyarı İşaretleri:**

1.7. KARŞILAŞTIRMA OPERATÖRLERİNİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDE KARŞILAŞTIRMA OPERATÖRLERİ (==, ===, !=, <, >, VB.) KULLANILARAK DEĞİŞKENLERİN KARŞILAŞTIRMASI AÇIKLANIR.

- PROGRAMLAMA DİLLERİ İŞARETLERLE İLETİŞİM KURAR

TÜRKÇE

EĞER GİRİLEN YAŞ 18'DEN BÜYÜKSE EKRANA "EHLİYET ALABİLİR" YAZ.

JAVASCRIPT

```
if (age > 18) {
console.log("EHLİYET ALABİLİR")
}
```

==

===

1.7. KARŞILAŞTIRMA OPERATÖRLERİNİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDE KARŞILAŞTIRMA OPERATÖRLERİ (==, ===, !=, <, >, VB.) KULLANILARAK DEĞİŞKENLERİN KARŞILAŞTIRMASI AÇIKLANIR.

- JAVASCRİPT KARŞILAŞTIRMA OPARATÖRLERİ

Eşittir (== ve ===)

- Diyelim ki karşılıklı iki grup var her grupta ikişer BALIK var.
- Bu balıkların sayıları 2'şer tanedir yani sayı olarak eşittir ancak bir grup japon balığı diğer grup köpek balığıdır yani tür olarak eşit değildir .

7 < 12

!=

1.7. KARŞILAŞTIRMA OPERATÖRLERİNİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDE KARŞILAŞTIRMA OPERATÖRLERİ (==, ===, !=, <, >, VB.) KULLANILARAK DEĞİŞKENLERİN KARŞILAŞTIRMASI AÇIKLANIR.

- JAVASCRİPT KARŞILAŞTIRMA OPARATÖRLERİ

Büyüktür (>) ve Küçüktür (<)

- >: Bir değerin diğerinden büyük olup olmadığını kontrol eder.
- <: Bir değerin diğerinden küçük olup olmadığını kontrol eder.

1.7. KARŞILAŞTIRMA OPERATÖRLERİNİ JAVASCRİPT PROGRAMLAMA DİLİ İLE KULLANIR

B) KARŞILAŞTIRMA OPERATÖRLERİ İLE KOŞULLU İFADELERİN BİRLİKTE KULLANILMASIYLA İLGİLİ ÖRNEKLER YAPILIR.

- https://onuraltuntas61.w3spaces-preview.com/_/error/unauthorized.html

1.8. BİRDEN FAZLA KOŞUL DURUMLARINI BİRLEŞTİRMEK İÇİN MANTIKSAL OPERATÖRLER KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ MANTIKSAL OPERATÖRLER (&&, II, !, VB.) AÇIKLANIR.

&&

ve

||

veya

!

değil

1.8. BİRDEN FAZLA KOŞUL DURUMLARINI BİRLEŞTİRMEK İÇİN MANTIKSAL OPERATÖRLER KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ MANTIKSAL OPERATÖRLER (&&, II, !, VB.) AÇIKLANIR.

&& - ve

ZEYNEP VE MURAT TAHTAYA KALKSIN

HER İKİSİ DE TAHTAYA KALKAR

1.8. BİRDEN FAZLA KOŞUL DURUMLARINI BİRLEŞTİRMEK İÇİN MANTIKSAL OPERATÖRLER KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ MANTIKSAL OPERATÖRLER (&&, II, !, VB.) AÇIKLANIR.

|| - veya

ZEYNEP VEYA MURAT TAHTAYA KALKSIN

İÇLERİNDEN BİR TANESİ TAHTAYA KALKMASI YETERLİDİR

1.8. BİRDEN FAZLA KOŞUL DURUMLARINI BİRLEŞTİRMEK İÇİN MANTIKSAL OPERATÖRLER KULLANIR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ MANTIKSAL OPERATÖRLER (&&, II, !, VB.) AÇIKLANIR.

! - DEĞİL

ZEYNEP TAHTAYA KALKSIN

ZEYNEP YERİNE MURAT TAHTAYA KALKARSA ÖĞRETMEN» SEN DEĞİL ZEYNEP GELSİN» DİYE*CEKTİR*.

1.8. BİRDEN FAZLA KOŞUL DURUMLARINI BİRLEŞTİRMEK İÇİN MANTIKSAL OPERATÖRLER KULLANIR.

B) MANTIKSAL OPERATÖRLERİ KULLANARAK BİRDEN FAZLA KOŞUL BİRLEŞTİRİLİR VE KARMAŞIK İFADELER OLUŞTURULUR.

1. Kullanıcının üye olması gerekir (uyeMi).
2. Kullanıcının ya 18 yaşından büyük olması gerekir (yas > 18) ya da özel izin verilmiş olması gerekir (ozelIzin).
3. Engel durumu var mı? (engelDurumu)

```
let uyeMi = true;
let yas = 17;
let ozelIzin = true;
let engelDurumu = false;

if (uyeMi && (yas > 18 || ozelIzin) && !engelDurumu) {
    console.log("İçeriği görüntüleyebilirsiniz.");
} else {
    console.log("İçeriği görüntüleyemezsiniz.");
}
```

2. ÜNİTE: JAVASCRIPT PROGRAMLAMA DİLİ FONKSİYONLARI VE NESNE TABANLI PROGRAMLAMA

Ünite Açıklaması

- Bu ünitede; Javascript programlama dilinde fonksiyonlarının genel tanımı, fonksiyonların düzenlenmesi ve yönetilmesi, fonksiyonlarda kullanılan parametreler, fonksiyonların değişkenlere atanabilmesi üzerinde durulur. Javascript programlama dilinde diziler ve döngüler anlatılarak dizi elemanlarını sıralama ve filtreleme olaylarından bahsedilir. Nesne ve sınıf kavramları anlatılarak constructor'lara özellikler ekleme, nesneler arası etkileşim kurma işlemleri uygulamalı olarak yapılır.

2. ÜNİTE: JAVASCRIPT PROGRAMLAMA DİLİ FONKSİYONLARI VE NESNE TABANLI PROGRAMLAMA

Değerler

- Sabır (Kazanım 1.1., 1.2., 1.3., 1.6.)
- Özdenetim (Kazanım 1.4., 1.5., 1.6.)
- Sorumluluk (Kazanım 1.4., 1.6., 1.7., 1.8.)
- Yardımseverlik (Kazanım 1.6.)

Alan Becerileri

- Soyutlama Becerisi (Kazanım 1.1., 1.2., 1.3.)
- Mantıksal Düşünme Becerisi (Kazanım 1.4., 1.5., 1.6., 1.7.)
- Algoritmik Düşünme Becerisi (Kazanım 1.4., 1.5., 1.6., 1.7., 1.8
- Kodlama, Programlama Becerisi (Kazanım 1.1., 1.2., 1.3., 1.4., 1.5., 1.6., 1.7., 1.8.)

2.1. JAVASCRİPT PROGRAMLAMA DİLİ İLE FONKSİYON TANIMLAR.

A) JAVASCRİPT PROGRAMLAMA DİLİ İLE FONKSİYONLARIN NASIL TANIMLANDIĞI VE KULLANILDIĞI AÇIKLANIR.

Fonksiyon Nedir?

Fonksiyon Tanımlama (Kod düzeni)

Parametreler ve Argümanlar

Fonksiyon Çağrısı

Fonksiyonlardan Değer Döndürme

```
function carpma(sayi1, sayi2) {
    return sayi1 * sayi2;
}

let sonuc = carpma(4, 3);
console.log(sonuc); // Çıktı: 12
```

2.1. JAVASCRİPT PROGRAMLAMA DİLİ İLE FONKSİYON TANIMLAR.

B) FONKSİYONLAR KULLANILARAK TEKRARLAYAN İŞLEMLERİN KOLAYLAŞTIRILDIĞI VURGULANIR.

Fonksiyon olmasaydı bütün işaretleri düzeltmek zorunda kalacaktık

```
a * b
c * d
e + f
g  / j
k – m
n + p
```

Fonksiyonların Avantajları

1. **Yeniden Kullanılabilirlik:**
2. **Düzen ve Okunabilirlik:**
3. **Hata Ayıklama:**

Fonksiyon tanımında sadece bir değişiklik yeter(+ yerine *)

```
function carpma(sayi1, sayi2) {
    return sayi1  + sayi2;
}

let sonuc = carpma(4, 3);
console.log(sonuc); // Çıktı: 12
```

2.2. FONKSİYONLARI ÇAĞIRARAK KODUN DÜZENLENMESİNİ KAVRAR.

A) FONKSİYONLARIN ÇAĞIRILARAK PROGRAM İÇERİSİNDEKİ KULLANIM ŞEKLİ AÇIKLANIR.

- Fonksiyonları, bir video oyunundaki "güçlendirme" veya "özel yetenek" butonu gibi düşünebiliriz.
- Oyunda, bu özel yetenekleri kullanmak için belirli bir butona basmanız gerekir, ve her basışınızda, karakteriniz özel bir hareket yapar.
- Programlamada fonksiyon çağrısı da benzer şekilde işler; fonksiyonun adını yazarak ve parantezleri kullanarak "o butona basarsınız" ve kodunuzda o fonksiyonun tanımladığı işlemi gerçekleştirirsiniz.

FONKSİYON TANIMLAMA

```
function selamVer() {
    console.log("Merhaba!");
}
```

FONKSİYONU ÇAĞIRMAK

```
selamVer();
```

2.2. FONKSİYONLARI ÇAĞIRARAK KODUN DÜZENLENMESİNİ KAVRAR.

B) FONKSİYONLARIN KODUN DAHA DÜZENLİ VE OKUNABİLİR OLMASI KONUSUNDA SAĞLADIĞI KATKI VURGULANIR.

```
function selamVer(isim) {
    console.log("Merhaba, " + isim + "!");
}
```

Fonksiyon Kullanmadan

```
console.log("Merhaba, nasılsın Ayşe?");
console.log("Merhaba, nasılsın Ali?");
console.log("Merhaba, nasılsın Zeynep?");
console.log("Merhaba, nasılsın Mehmet?");
```

Fonksiyon Kullanarak

```
selamVer("Ayşe");
selamVer("Ali");
selamVer("Zeynep");
selamVer("Mehmet");
```

2.3. FONKSİYONLARI ÇAĞIRARAK KODUN YÖNETİLMESİNİ KAVRAR.

A) FONKSİYONLARIN RETURN DEĞERİNİ YAKALAMASI VE BU DEĞERİN KULLANILMASI AÇIKLANIR.

TÜM YİYECEKLER RETURN EDİLMİŞ AMA DAHA KULLANILMADI. İÇİNE PARA ATINCA KULLANILMIŞ OLACAK

Fonksiyonlardan Değer Döndürmek

return anahtar kelimesinin kullanıldığı aktarılır

```
function carpma(sayi1, sayi2) {
    return sayi1 * sayi2;
}

let sonuc = carpma(4, 3);
console.log(sonuc); // Çıktı: 12
```

2.3. FONKSİYONLARI ÇAĞIRARAK KODUN YÖNETİLMESİNİ KAVRAR.

B) FONKSİYONLARIN FARKLI DOSYALAR İÇERİSİNDE TUTULARAK KOD KARMAŞIKLIĞININ AZALTILABİLECEĞİ VURGULANIR

BU OKUL DEFTERİNE HEM MATEMATİK HEM FİZİK HEM KİMYA HEM BİYOLOJİ HEM TARİH HEM COĞRAFYA HEM EDEBİYAT NOTLARINA YAZSAK NASIL OLURDU?

2.3. FONKSİYONLARI ÇAĞIRARAK KODUN YÖNETİLMESİNİ KAVRAR.

B) FONKSİYONLARIN FARKLI DOSYALAR İÇERİSİNDE TUTULARAK KOD KARMAŞIKLIĞININ AZALTILABİLECEĞİ VURGULANIR

BÜTÜN DERSLERİ AYRI BİR DEFTERE NOT ALMAK DÜZEN AÇISINDAN SON DERECE YARARLIDIR

2.3. FONKSİYONLARI ÇAĞIRARAK KODUN YÖNETİLMESİNİ KAVRAR.

B) FONKSİYONLARIN FARKLI DOSYALAR İÇERİSİNDE TUTULARAK KOD KARMAŞIKLIĞININ AZALTILABİLECEĞİ VURGULANIR

Programlama Dünyasında Fonksiyonlar ve Dosyalar

Diyelim ki bir oyun programlıyorsun ve bu oyunda oyuncunun sağlığını artıran iksirler, hasar veren tuzaklar ve puan kazandıran altınlar var. Oyuncunun sağlığıyla ilgili işlemleri yapan fonksiyonları bir dosyada (örneğin saglik.js), tuzaklarla ilgili işlemleri başka bir dosyada (örneğin tuzaklar.js), ve puan sistemiyle ilgili işlemleri başka bir dosyada (örneğin puan.js) tutabilirsin.

saglik.js

tuzaklar.js

puan.js

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

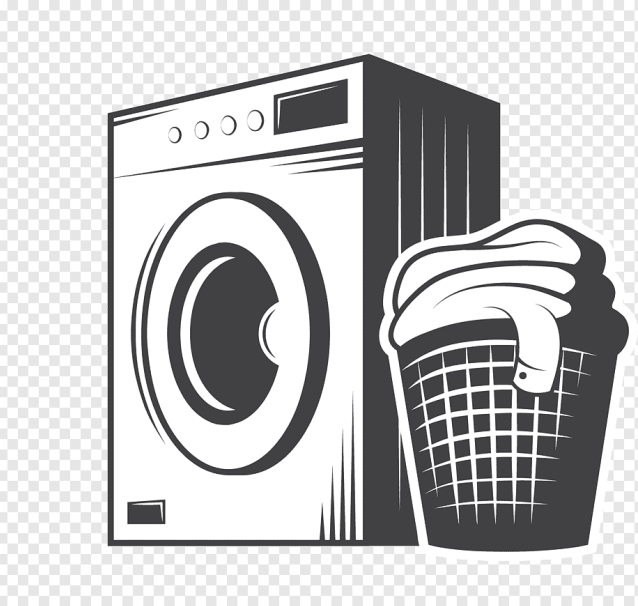
A) FONKSİYONLARA PARAMETRE EKLEME YÖNTEMİ VE BU PARAMETRELERİN KULLANILMA ŞEKİLLERİ AÇIKLANIR.

Diyelim ki çamaşır makineniz var ve bu çamaşır makinesinin temel işlevi çamaşırları yıkamaktır. Programlama diliyle konuşacak olursak, çamaşır makinesi bir "fonksiyon" gibidir.

Bu fonksiyonun adı "çamaşırYıkama" olsun.

•Çamaşır makinesini çalıştırmak için, içine koyacağınız çamaşırlar (parametreler) ve belki de yıkama programı gibi ek bilgiler gereklidir.

•Yani bu fonksiyonun çalışması için bazı girdilere ihtiyacı vardır.



2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

A) FONKSİYONLARA PARAMETRE EKLEME YÖNTEMİ VE BU PARAMETRELERİN KULLANILMA ŞEKİLLERİ AÇIKLANIR.

Parametrelerin Kullanılması:

•Çamaşır makinesini çalıştırırken seçtiğiniz yıkama programı (örneğin, hızlı yıkama, ağır kirli, hassas), fonksiyon içinde bu parametrelere göre farklı işlemler yapılmasını sağlar.

•Çamaşır makinesine koyduğunuz her bir çamaşır tipi de (örneğin, renkliler, beyazlar, hassas kıyafetler) birer "parametre"dir.

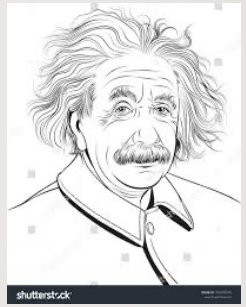
•Eğer fonksiyonumuz birden fazla parametre alıyorsa, bunları virgülle ayırarak sıralayabiliriz.

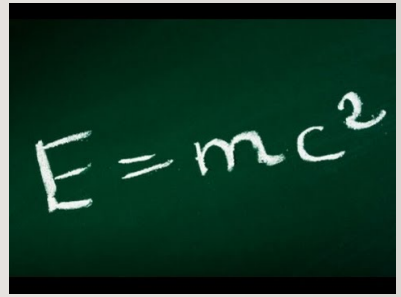
çamaşırYıkama("renkliler", "hızlıYıkama");

Parametreler

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

A) FONKSİYONLARA PARAMETRE EKLEME YÖNTEMİ VE BU PARAMETRELERİN KULLANILMA ŞEKİLLERİ AÇIKLANIR.





UYGULAMA ÖRNEĞİ

```
function enerjiHesapla(kutle) {
    const c = 299792458; // Işık hızı, metre/saniye cinsinden
    return kutle * c * c;
}
```

```
let sonuc = enerjiHesapla(1);

console.log("1 kilogram kütlenin eşdeğer enerjisi: " + sonuc + " joule");
```

Çıktı: 1 kilogram kütlenin eşdeğer enerjisi: 89875517873681760 joule

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

B) PARAMETRELERİN FONKSİYONLARIN KULLANIMINDA ESNEKLİĞİ ARTIRDIĞI VURGULANIR.

---

1. Tekrar Kullanılabilirlik

2. Kodun Okunabilirliği ve Temizliği

3. Esneklik ve Genelleme

Sayılarla Çalışmak

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

C) FONKSİYONLARIN FARKLI TİPTEKİ PARAMETRELERİ (SAYI, DİZİ, NESNE, VB.) İŞLEYEBİLMESİ AÇIKLANIR.

```
function topla(sayi1, sayi2) {
    return sayi1 + sayi2;
}

console.log(topla(3,8))
```

Çıktı: 11

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

C) FONKSİYONLARIN FARKLI TİPTEKİ PARAMETRELERİ (SAYI, DİZİ, NESNE, VB.) İŞLEYEBİLMESİ AÇIKLANIR.

Dizilerle Çalışmak

```
function diziToplami(sayilar) {
    let toplam = 0;
    for (let sayi of sayilar) {
        toplam += sayi;
    }
    return toplam;
}

numbers =[ 3, 5, 8, 9,16, 33]

Console.log(diziToplami(numbers))
```

Çıktı: 74

Nesnelerle Çalışmak

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

C) FONKSİYONLARIN FARKLI TİPTEKİ PARAMETRELERİ (SAYI, DİZİ, NESNE, VB.) İŞLEYEBİLMESİ AÇIKLANIR.

```
class Kisi {
    constructor(ad, yas) {
        this.ad = ad;
        this.yas = yas;
    }
}
```

```
let kisi1 = new Kisi("Ayşe", 30);

let kisi2 = new Kisi("Mehmet", 25);
```

```
function selamla(kisi) {
    return `Merhaba, benim adım ${kisi.ad} ve yaşım ${kisi.yas}.`;
}

console.log(selamla(kisi2))
```

Çıktı: Merhaba, benim adım Mehmet ve yaşım 25.

Fonksiyonlarla Çalışmak
Evet, fonksiyonlar başka fonksiyonlarla da çalışabilir!

2.4. FONKSİYONLARA PARAMETRE EKLEMEYİ KAVRAR.

C) FONKSİYONLARIN FARKLI TİPTEKİ PARAMETRELERİ (SAYI, DİZİ, NESNE, VB.) İŞLEYEBİLMESİ AÇIKLANIR.

```
function islemYap(islem, deger) {
    return islem(deger);
}

function ikiEkle(sayi) {
    return sayi + 2;
}

console.log(islemYap(ikiEkle, 5));
```

Çıktı: 7

2.5. FONKSİYONLARIN DEĞİŞKENLERE ATANARAK KULLANILABİLECEĞİNİ ANLAR.

A) FONKSİYONLARIN BİRER DEĞERİ OLDUĞUNU VE BUNUN DEĞİŞKENLERE ATANARAK BAŞKA YERDEN ÇAĞRILABİLECEĞİ AÇIKLANIR.

Fonksiyonların Değişkenlere Atanması

```
const selamVer = function() {
    console.log("Merhaba!");
};
```

Fonksiyonların Başka Yerlerden Çağrılması

```
selamVer();
```

Çıktı: Merhaba!

Fonksiyonların Değişkenlere Atanması

Anonim fonksiyonlar, adı olmayan ve genellikle bir kez kullanılmak üzere tasarlanmış fonksiyonlardır.

2.5. FONKSİYONLARIN DEĞİŞKENLERE ATANARAK KULLANILABİLECEĞİNİ ANLAR.

B) FONKSİYONLARIN ANONİM (İSİMSİZ) OLARAK KULLANILABİLECEĞİ VURGULANIR.

TEK BAŞINA BİR ANLAM İFADE ETMEZLER

```
function() {
    console.log("Merhaba, nasılsın?");
};
```

Örneğin, bir HTML elemanına tıklama olayı eklerken doğrudan kullanılabilir:

https://onuraltuntas61.w3spaces.com/isimsiz_fonksiyon.html

2.5. FONKSİYONLARIN DEĞİŞKENLERE ATANARAK KULLANILABİLECEĞİNİ ANLAR.

C) FONKSİYONLARIN RETURN DEĞERİNİN DEĞİŞKENLERE ATANABİLME DURUMU AÇIKLANIR.

Uygulama

```
function kare(sayi1) {
    return sayi1 ** 2;
}

let sonuc = kare(12);

console.log(sonuc); // Çıktı: 144
```

Kare fonksiyonunu sonuç değişkenine atadık.

2.6. JAVASCRİPT PROGRAMLAMA DİLİ İLE DİZİLERİ TANIMLAR.

B) YENİ DİZİ OLUŞTURMA, DİZİLERE ELEMAN EKLEME VE ÇIKARMA İŞLEMLERİNİN NASIL YAPILACAĞI AÇIKLANIR.

Yeni Dizi Oluşturma

```
let dizi = []; // Boş bir dizi
let meyveler = ["Elma", "Muz", "Çilek"];
```

Dizilere Eleman Ekleme

```
meyveler.push("Portakal");
console.log(meyveler);

Çıktı:  ["Elma", "Muz", "Çilek", "Portakal"]

meyveler.unshift("Kivi");
console.log(meyveler);

Çıktı: ["Kivi", "Elma", "Muz", "Çilek", "Portakal"]
```

Dizilere Eleman Çıkarma

["Kivi", "Elma", "Muz", "Çilek", "Portakal"]

Dizilere Eleman Çıkarma

2.6. JAVASCRİPT PROGRAMLAMA DİLİ İLE DİZİLERİ TANIMLAR.

B) YENİ DİZİ OLUŞTURMA, DİZİLERE ELEMAN EKLEME VE ÇIKARMA İŞLEMLERİNİN NASIL YAPILACAĞI AÇIKLANIR.

```
meyveler.pop();
console.log(meyveler);
```

Çıktı: ["Kivi", "Elma", "Muz", "Çilek"]

```
meyveler.shift("Kivi");
console.log(meyveler);
```

Çıktı: ["Elma", "Muz", "Çilek"]

2.7. JAVASCRİPT PROGRAMLAMA DİLİNDE DÖNGÜLERİN KULLANIMINI KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ DÖNGÜ YAPILARI (FOR, WHİLE, DO-WHİLE) AÇIKLANIR.

For Döngüsü

KİLİT KAVRAMLAR

- İŞLEM TEKRARI
- BAŞLANGIÇ DEĞERİ
- BİTİŞ DEĞERİ
- ARTIŞ DEĞERİ

```
for (let sayac = 0; sayac < 6; sayac++) {
    console.log(sayac);
}
```

Çıktı:

```
0
1
2
3
4
5
```

2.7. JAVASCRİPT PROGRAMLAMA DİLİNDE DÖNGÜLERİN KULLANIMINI KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ DÖNGÜ YAPILARI (FOR, WHİLE, DO-WHİLE) AÇIKLANIR.

while Döngüsü

KİLİT KAVRAMLAR

- İŞLEM TEKRARI
- KOŞUL İFADESİ
- KOŞULU BİTİRECEK İFADE
- SONSUZ DÖNGÜ

```
let sayac = 0;
while (sayac < 8) {
    console.log(sayac);
    sayac++;
}
```

Çıktı:

```
0
1
2
3
4
5
6
7
```

2.7. JAVASCRİPT PROGRAMLAMA DİLİNDE DÖNGÜLERİN KULLANIMINI KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ DÖNGÜ YAPILARI (FOR, WHİLE, DO-WHİLE) AÇIKLANIR.

do while Döngüsü

KİLİT KAVRAMLAR

- İŞLEM TEKRARI
- KOŞUL İFADESİ
- KOŞULUN SIRASI (SONDA)
- EN AZ BİR KERE İŞLEMİN GERÇEKLEŞMESİ
- KOŞULU BİTİRECEK İFADE
- SONSUZ DÖNGÜ

```
let sayac = 0;
do {
    console.log(sayac);
    sayac++;
} while (sayac < 8);
```

Çıktı:

```
0
1
2
3
4
5
6
7
```

2.7. JAVASCRİPT PROGRAMLAMA DİLİNDE DÖNGÜLERİN KULLANIMINI KAVRAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ DÖNGÜ YAPILARI (FOR, WHİLE, DO-WHİLE) AÇIKLANIR.

while Döngüsü

```
let sayac = 0;
while (sayac < 5) {
    console.log(sayac);
    sayac++;
}
```

Çıktı:

```
0
1
2
3
4
```

do while Döngüsü

```
let sayac = 0;
do {
    console.log(sayac);
    sayac++;
} while (sayac < -8);
```

Çıktı:

```
0
```

*Koşul ifadesi gerçekleşmemesine rağmen işlem en az bir kere tekrar etti

2.7. JAVASCRIPT PROGRAMLAMA DILINDE DÖNGÜLERIN KULLANIMINI KAVRAR.

B) DÖNGÜLERIN KULLANILDIĞI SENARYO ÖRNEKLERI VERILIR.

DÖNGÜ SENARYO ÖRNEKLERİ

- 15'den 85’e kadar olan sayıları ekrana yazdırın.
- Bir dizi içerisindeki tüm elemanların toplamını hesaplayan bir program yazın.
- Kullanıcıdan sürekli olarak sayı girmesini isteyen ve girilen sayıların toplamı 100'ü geçtiğinde
programın sonlanmasını sağlayan bir program yazın.
- Bir sayının faktöriyelini hesaplayan bir program yazın.
- Kullanıcıdan en az bir kere sayı girmesini isteyen ve girilen sayılar pozitif olduğu sürece bu işlemi tekrarlayan bir program yazın.

2.8. DİZİLER İÇERİSİNDE GEZİNMEYİ KAVRAR.

A) DİZİLER İÇERİSİNDE DÖNGÜLER KULLANILARAK NASIL GEZİLECEĞİ AÇIKLANIR.

for döngüsü

```
const elemanlar = [1, 2, 3, 4, 5];
for (let i = 0; i < elemanlar.length; i++) {
    console.log(elemanlar[i]);
}
```

forEach Metodu

```
const elemanlar = [1, 2, 3, 4, 5];
elemanlar.forEach(function(eleman) {
    console.log(eleman);
});
```

for .. of döngüsü

```
const elemanlar = [1, 2, 3, 4, 5];
for (const eleman of elemanlar) {
    console.log(eleman);
}
```

2.8. DİZİLER İÇERİSİNDE GEZİNMEYİ KAVRAR.

B) DİZİLERİN UZUNLUĞUNUN KONTROL EDİLME DURUMU AÇIKLANIR.

Dizilerin uzunluğunu bulma

```
const elemanlar = [1, 2, 3, 4, 5];

console.log(elemanlar.length);      // Çıktı: 5
```

2.9. DİZİLER İÇERİSİNDE İŞLEM YAPMAYI KAVRAR.

A) DİZİLER İÇERİSİNDE DÖNGÜLER KULLANILARAK DİZİ ELEMANLARI ÜZERİNDE NASIL DEĞİŞİKLİK YAPILACAĞI AÇIKLANIR.

Örnek uygulama: Dizi üzerindeki her sayıyı 2 ile çarpmak

```
let sayilar = [1, 2, 3, 4, 5];

for (let i = 0; i < sayilar.length; i++) {
    sayilar[i] = sayilar[i] * 2
}

console.log(sayilar); // Çıktı: [2, 4, 6, 8, 10]
```

2.9. DİZİLER İÇERİSİNDE İŞLEM YAPMAYI KAVRAR.

A) DİZİLER İÇERİSİNDE DÖNGÜLER KULLANILARAK DİZİ ELEMANLARI ÜZERİNDE NASIL DEĞİŞİKLİK YAPILACAĞI AÇIKLANIR.

map metodu

```
const sayilar = [1, 2, 3, 4, 5];

const yeniSayilar = sayilar.map(deger => deger * 2);

console.log(yeniSayilar); // Çıktı: [2, 4, 6, 8, 10]
```

*Bu yöntem, orijinal diziyi değiştirmez; her eleman üzerinde belirtilen işlemi uygular ve işlemin sonucunu yeni bir dizi olarak döndürür.

2.9. DİZİLER İÇERİSİNDE İŞLEM YAPMAYI KAVRAR.

B) DİZİ ELEMANLARINI SIRALAMA VE FİLTRELEME İŞLEMLERİ AÇIKLANIR.

Sayısal değerler için küçükten büyüğe sıralama

```
const sayilar = [3, 1, 4, 1, 5, 9, 2, 6];

sayilar.sort((a, b) => a - b);

console.log(sayilar); // Çıktı: [1, 1, 2, 3, 4, 5, 6, 9]
```

Sayısal değerler için büyükten küçüğe sıralama

```
const sayilar = [3, 1, 4, 1, 5, 9, 2, 6];

sayilar.sort((a, b) => b - a);

console.log(sayilar); // Çıktı: [9, 6, 5, 4, 3, 2, 1]
```

2.9. DİZİLER İÇERİSİNDE İŞLEM YAPMAYI KAVRAR.

B) DİZİ ELEMANLARINI SIRALAMA VE FİLTRELEME İŞLEMLERİ AÇIKLANIR.

string değerler için küçükten büyüğe sıralama

```
const kelimeler = ['elma', 'armut', 'muz', 'kiraz’];

kelimeler.sort();

console.log(kelimeler); // Çıktı: ['armut', 'elma', 'kiraz', 'muz']
```

Dizi elemanlarını filtreleme

```
const sayilar = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

const ciftSayilar = sayilar.filter(sayi => sayi % 2 === 0);

console.log(ciftSayilar); // Çıktı: [2, 4, 6, 8, 10]
```

2.10. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNE VE SINIF KAVRAMLARINI TANIMLAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNELERİN VE SINIFLARIN TANIMLANMASI AÇIKLANIR.

Nesneler

Bir nesne, gerçek hayatta herhangi bir şeyi temsil edebilir. Örneğin, bir kalem, bir kitap, bir telefon ya da hatta bir öğrenci. Bu nesnelerin her birinin özellikleri (renk, boyut, marka gibi) ve yapabilecekleri işlevler (yazmak, aramak yapmak gibi) vardır.

2.10. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNE VE SINIF KAVRAMLARINI TANIMLAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNELERİN VE SINIFLARIN TANIMLANMASI AÇIKLANIR.

Nesne oluşumu:

JavaScript'te bir nesne, süslü parantezler {} kullanılarak oluşturulur ve içinde çeşitli özellikler ve fonksiyonlar barındırabilir. Özellikler, nesnenin sahip olduğu verilerdir (adı, yaşı gibi), fonksiyonlar ise nesnenin yapabileceği işlemleri ifade eder.

```
let ogrenci = {
    isim: "Ayşe",
    yas: 16,
    okul: "Lise",
    dersCalis: function() {
        console.log(this.isim + " ders çalışıyor.");
    }
};

// Nesnenin özelliklerine erişim
console.log(ogrenci.isim); // Çıktı: Ayşe

// Nesnenin fonksiyonunu çağırma
ogrenci.dersCalis(); // Çıktı: Ayşe ders çalışıyor.
```

2.10. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNE VE SINIF KAVRAMLARINI TANIMLAR.

A) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ NESNELERİN VE SINIFLARIN TANIMLANMASI AÇIKLANIR.

B) NESNELERİN ÖZELLİKLERİ VE METOTLARININ KULLANILMA ŞEKLİ AÇIKLANIR.

2.11. CONSTRUCTOR FONKSİYONUYLA NESNE OLUŞTURMAYI KAVRAR.

A) CONSTRUCTOR FONKSİYONLARININ TANIMLANARAK NESNE OLUŞTURULMASI AÇIKLANIR.

B) CONSTRUCTOR FONKSİYONLARI İLE NESNELERE ÖZELLİK EKLEME DURUMU AÇIKLANIR.

C) CONSTRUCTOR FONKSİYONLARI İLE NESNELERİ ÖZELLEŞTİRME DURUMU AÇIKLANIR.

*__constructor__ metodu, sınıftan her yeni nesne oluşturulduğunda çağrılır ve o nesnenin özelliklerini başlatır.

Sınıflar

Sınıflar, nesneleri oluşturmak için kullanılan bir tür "şablon" ya da "kalıptır". JavaScript'te bir sınıf tanımlamak için **class** anahtar kelimesi kullanılır. Sınıf içinde öğrencilerin (veya herhangi bir nesnenin) sahip olacağı ortak özellikler ve fonksiyonlar tanımlanır. Sonrasında, bu sınıftan yeni nesneler oluşturulabilir.

```
class Ogrenci {
    constructor(isim, yas, okul) {
        this.isim = isim;
        this.yas = yas;
        this.okul = okul;
    }

    dersCalis() {
        console.log(this.isim + " ders çalışıyor.");
    }
}

// Sınıftan bir nesne oluşturma
let ogrenci1 = new Ogrenci("Mehmet", 17, "Lise");

// Oluşturulan nesnenin özelliklerine erişim ve fonksiyonunu çağırma
console.log(ogrenci1.isim); // Çıktı: Mehmet

ogrenci1.dersCalis(); // Çıktı: Mehmet ders çalışıyor.
```

2.12. NESNELERE ÖZELLİK VE METOTLAR EKLEYEREK DİĞER NESNELERLE ETKİLEŞİM KURMAYI KAVRAR.

A) NESNELER ARASI İLETİŞİM VE ETKİLEŞİM SAĞLAMA YÖNTEMLERİ AÇIKLANIR.

B) NESNELER ARASINDA VERİ PAYLAŞMA VE METOT ÇAĞIRMA İŞLEMLERİ YAPILIR.

```
class Kullanici {
    constructor(isim) {
        this.isim = isim;
    }

    selamla() {
        console.log(`Merhaba, ben ${this.isim}!`);
    }
}

class Mesaj {
    constructor(gonderen, icerik) {
        this.gonderen = gonderen;
        this.icerik = icerik;
    }

    mesajGoster() {
        this.gonderen.selamla();
        console.log(`Mesaj: ${this.icerik}`);
    }
}

let kullanici = new Kullanici("Ayşe");
let mesaj = new Mesaj(kullanici, "Nasılsın?");
mesaj.mesajGoster();

// Çıktı:
// Merhaba, ben Ayşe!
// Mesaj: Nasılsın?
```

3. ÜNİTE:

JAVASCRIPT PROGRAMLAMA DİLİ İLE WEB GELİŞTİRME

Ünite Açıklaması

- Bu ünitede; Javascript programlama dilinde DOM kavramı, DOM'un web sitelerindeki rolü, DOM aracılığı ile
- HTML elemanlarına erişimden bahsedilir. Javascript programlama dilinde eventler kullanılarak kullanıcı ile
- etkileşim kurma, asenkron programlamanın web sitelerindeki önemi üzerinde durulur. Jquery ve Ajax kütüphaneleri
- ile etkileşim kurma, fetch işlemleri ile asenkron veri alışverişi yapma işlemleri anlatılır. Javascript
- programlama dili kodlarını modüler hâle getirerek kullanmanın öneminden, NodeJS ve CommonJS modülleri
- kullanılarak arkaplanda yapılabilecek işlemlerden, NPM aracılığı ile Javascript programlama dili içerisine harici
- kütüphaneler yüklenebileceğinden bahsedilir. Javascript programlama dilinin popüler framework'lerinden
- React, Angular ve Vue kullanılarak web siteleri oluşturabilme, React Native framework'ü ile mobil uygulamalar
- oluşturabilme işlemleri anlatılacaktır. Ecmascript kullanarak Javascript programlama dilinin güncel standartlarına
- uygun şekilde çalışılabileceğinden bahsedilecektir.

3. ÜNİTE:

JAVASCRIPT PROGRAMLAMA DİLİ İLE WEB GELİŞTİRME

Değerler

- Sabır (Kazanım 3.1., 3.2., 3.3., 3.5.)
- Özdenetim (Kazanım 3.4., 3.5., 3.6., 3.7., 3.14.)
- Sorumluluk (Kazanım 3.4., 3.5., 3.7., 3.14.)
- Yardımseverlik (Kazanım 3.11., 3.12., 3.13.)
- Alan Becerileri
- Soyutlama Becerisi (Kazanım 3.1., 3.2., 3.3.)
- Mantıksal Düşünme Becerisi (Kazanım 3.4., 3.5., 3.6., 3.7.)
- Algoritmik Düşünme Becerisi (Kazanım 3.4., 3.5., 3.6., 3.7.)
- Kodlama, Programlama Becerisi (Kazanım 3.4., 3.5., 3.6., 3.7., 3.8., 3.9., 3.10., 3.11.)
- Genelleme Becerisi (Kazanım 3.11., 3.12., 3.13., 3.14.)
- Test Etme ve Hata Ayıklama Becerisi (Kazanım 3.14.)

3.1. DOM KAVRAMINI TANIMLAR.

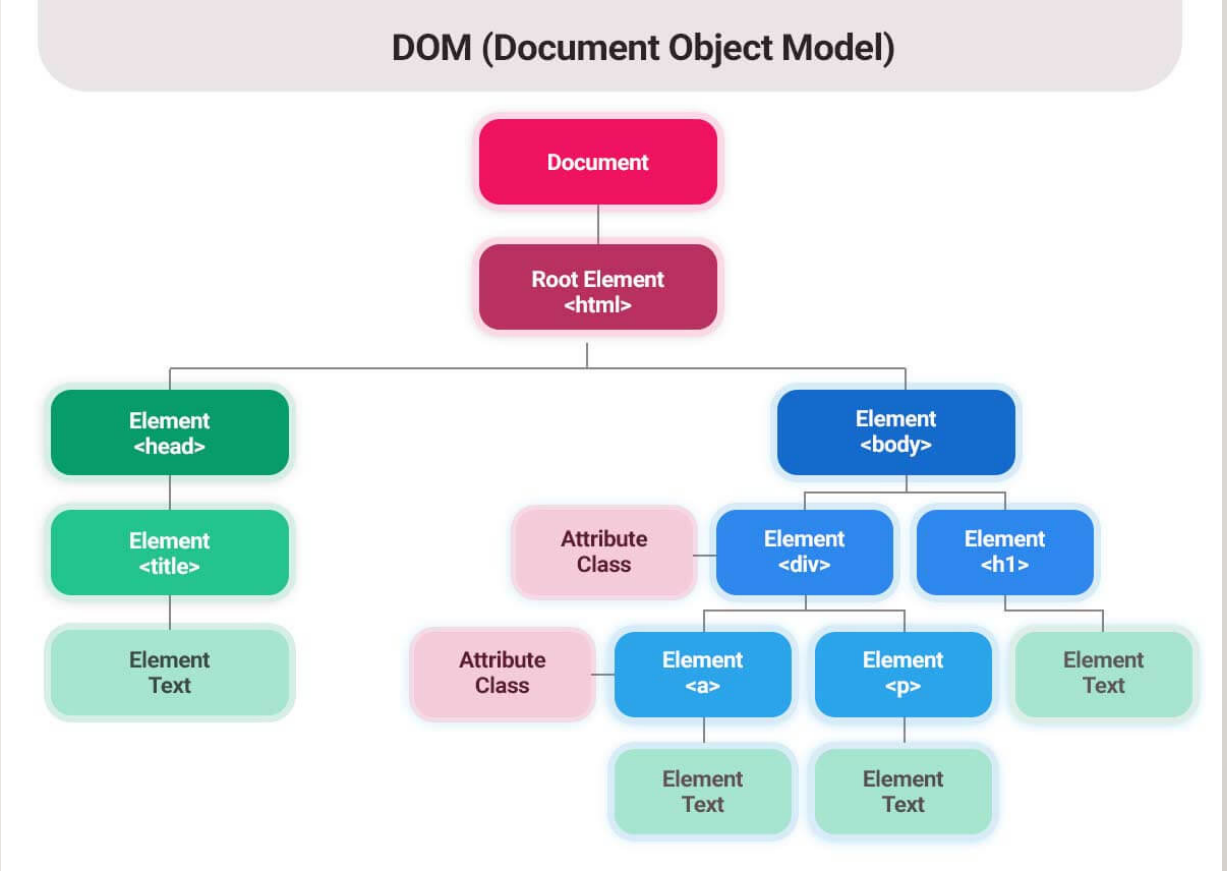
A) DOM (DOCUMENT OBJECT MODEL) TERİMİNİN NE ANLAMA GELDİĞİ AÇIKLANIR.

B) WEB SAYFALARININ AĞAÇ YAPISI OLARAK TEMSİL EDİLMESİ VE BU YAPININ KULLANILMA ŞEKLİ AÇIKLANIR.

C) DOM AĞACININ HER DÜĞÜMÜNÜN BİRER HTML ELEMANINI TEMSİL ETMESİ VURGULANIR.

DOM

İnternet tarayıcınızda gördüğünüz her web sayfasını bir ağaç gibi düşünebilirsiniz. Bu ağaçta dallar, yapraklar ve belki de bazı meyveler var. Web sayfası ağacında ise, dallar ve yapraklar web sayfasının çeşitli bölümlerini temsil eder: başlıklar, paragraflar, resimler, bağlantılar ve daha fazlası.



3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

DOM

- **Sayfa İçeriğinin Dinamik Değişimi: M**etin, görüntüler ve diğer HTML elemanlarının içeriğini güncelleyebilmenizi
- **Etkileşimli Kullanıcı Deneyimleri:** Form verilerini işleme, hata mesajlarını görüntüleme ve sayfa layout'unu kullanıcı tercihlerine göre değiştirme
- **Etkinlik Yönetimi: B**uton tıklamaları, klavye girişleri veya fare hareketleri
- **Sayfa Yapısını Programatik Olarak Değiştirme: D**inamik menüler, dialog kutuları
- **Performans ve Erişilebilirlik: S**ayfanın yüklenme süresi azaltılabilir, veya klavye erişilebilirliği için gerekli olan özellikler

3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

DOM

DOM, JavaScript ve diğer programlama dillerinin, web sayfasının içeriğini ve yapısını okumasına, değiştirmesine, eklemesine veya silmesine olanak tanır.

Yani bir web sayfasında bir butona tıkladığınızda bir menünün açılması, bir form doldurup gönderdiğinizde bir uyarı mesajı almanız gibi etkileşimleri DOM sayesinde gerçekleştirebiliriz.

Örneğin, bir web sayfasında bir paragrafın metnini değiştirmek istediğinizi düşünün. JavaScript ve DOM kullanarak, doğrudan o paragrafa ulaşabilir ve metnini istediğiniz gibi güncelleyebilirsiniz.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ANASAYFA</title>
</head>
<body>

<h1 id="benimParagrafim"> BU METİN DEĞİŞECEK</h1>

<button onclick="metniDegistir()">Metni Değiştir</button>

<script>

function metniDegistir(){
document.getElementById("benimParagrafim").textContent = "Merhaba, dünya!";
};

</script>

</body>

</html>
```

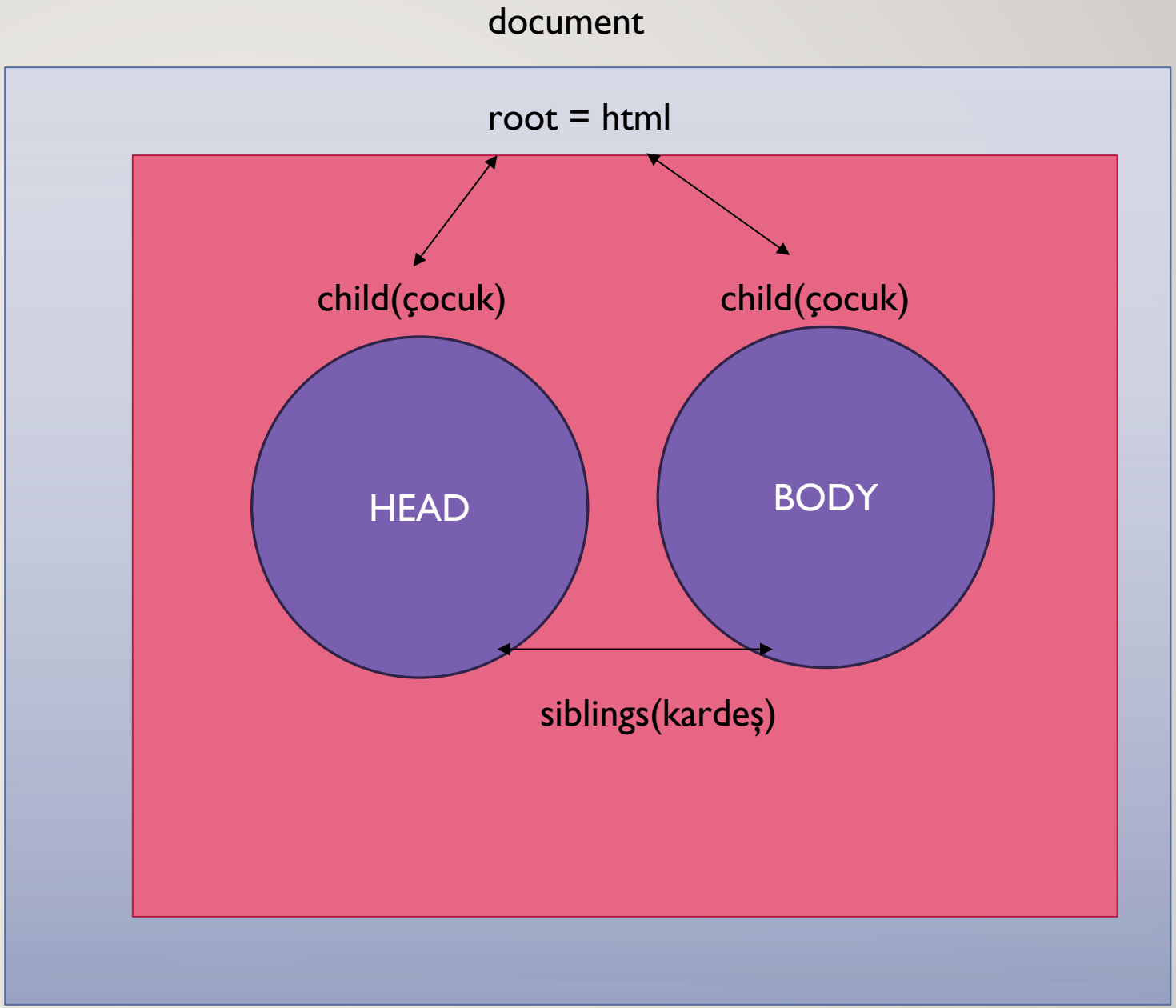
https://onuraltuntas61.w3spaces.com/dom_buton_metni_degistir.html

3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

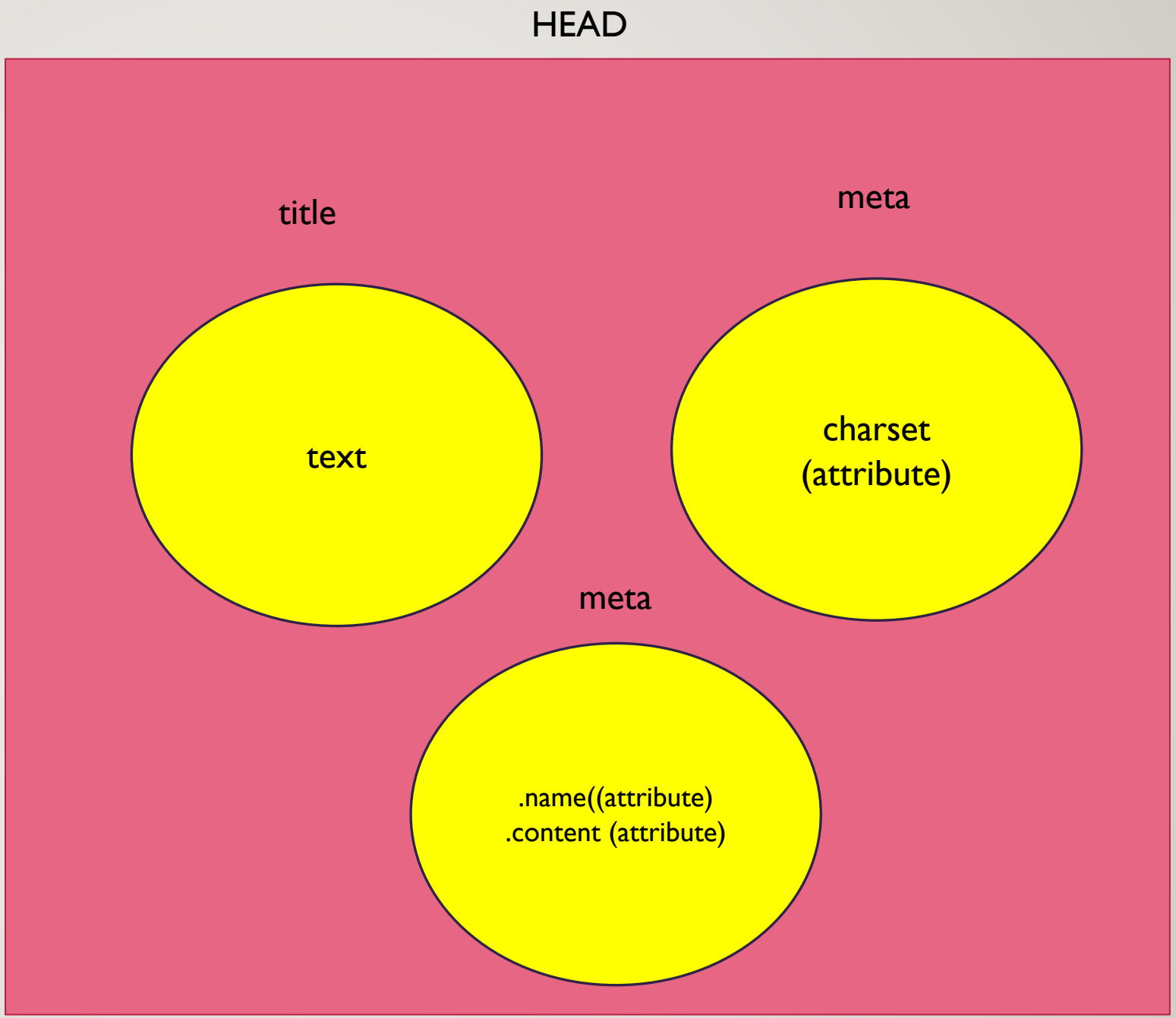


3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.



3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

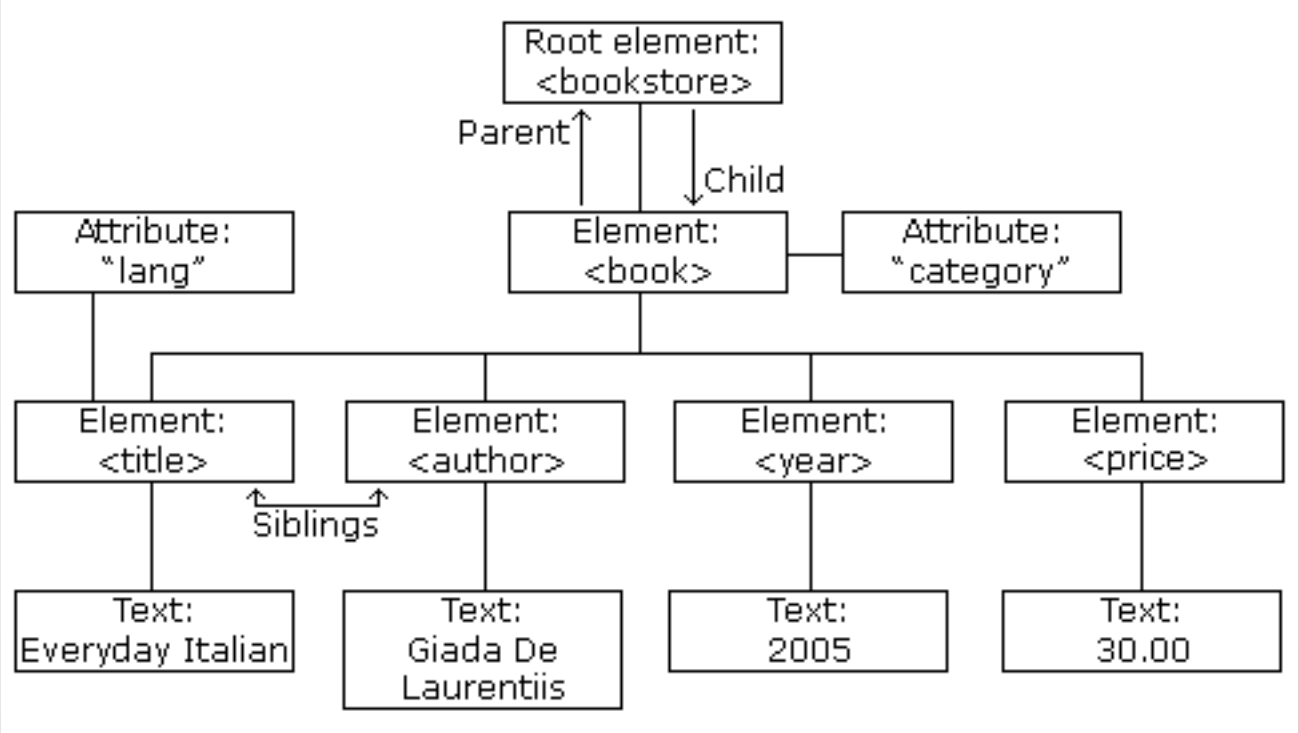
A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

KISACA

DOM OLMASAYDI JAVASCRİPT VE HTML İLETİŞİME GEÇEMEYECEKTİ!



3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

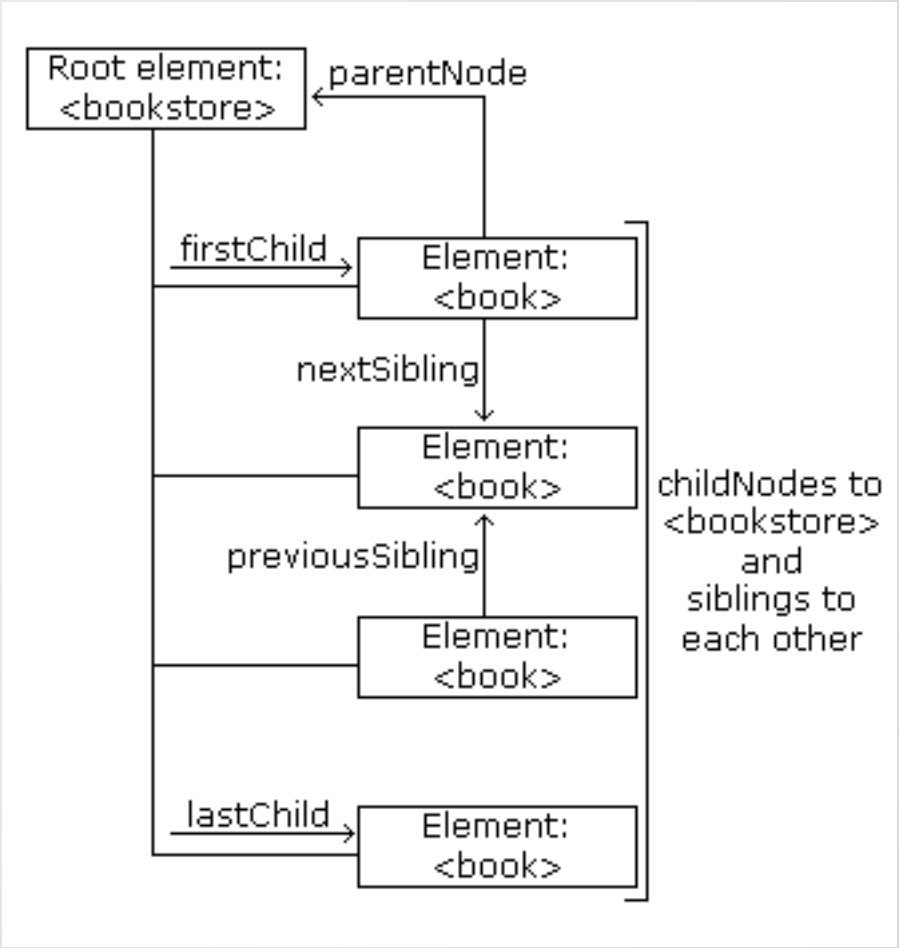
A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

KISACA

DOM OLMASAYDI JAVASCRİPT VE HTML İLETİŞİME GEÇEMEYECEKTİ!



3.2. DOM'UN WEB SAYFALARINDAKİ ROLÜNÜ KAVRAR.

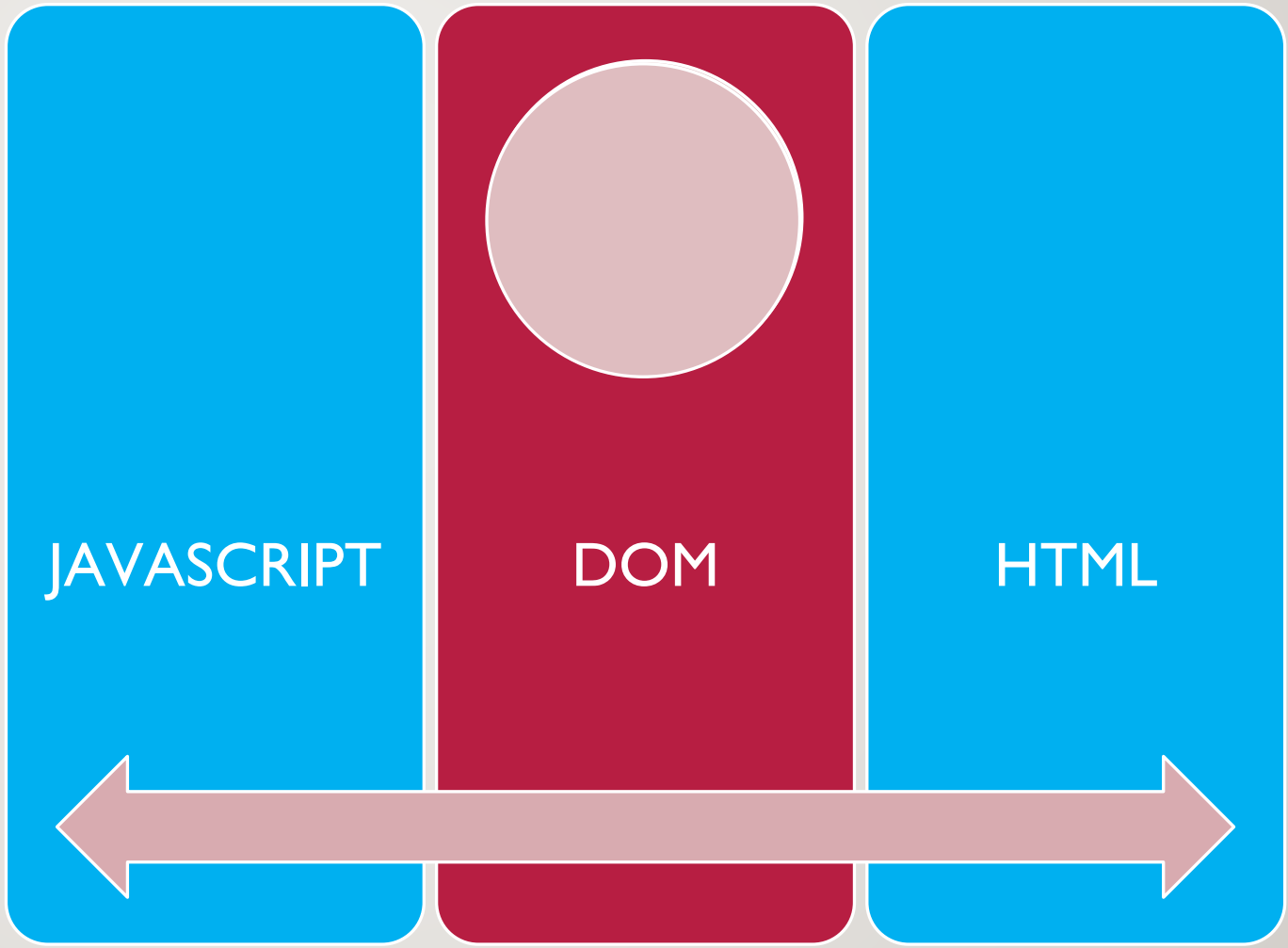
A) DOM'UN WEB SAYFALARINDAKİ İÇERİĞİ, YAPISI VE GÖRÜNÜMÜ AÇIKLANIR.

B) KULLANICI ARAYÜZÜ İLE ETKİLEŞİM KURMA, İÇERİK EKLEME VEYA DEĞİŞTİRME İŞLEMLERİNDE DOM'UN ÖNEMİ VURGULANIR.

C) DOM'UN WEB GELİŞTİRME SÜRECİNDE ALDIĞI ROL VURGULANIR.

KISACA

DOM OLMASAYDI JAVASCRİPT VE HTML İLETİŞİME GEÇEMEYECEKTİ!



3.3. DOM ARACILIĞI İLE HTML ELEMANLARINA NASIL ERİŞİLECEĞİNİ KAVRAR.

A) DOM ÜZERİNDEN HTML ELEMANLARINA ERİŞME YÖNTEMİ AÇIKLANIR.

B) ELEMANLAR SEÇİLEREK ÜZERİNDE DEĞİŞİKLİK YAPMA YÖNTEMLERİ AÇIKLANIR.

•**getElementById()**: Sayfadaki benzersiz bir ID'ye sahip bir elementi bulmak için kullanılır. Örneğin, document.getElementById('benzersizId') şeklinde kullanılır.

•**getElementsByClassName()**: Belirli bir sınıfa (class) sahip tüm elementleri bir liste olarak döndürür. Örneğin, document.getElementsByClassName('sinifAdi’).

•**getElementsByTagName()**: Belirli bir etiket adına (örneğin, <p>,) sahip tüm elementleri bir liste olarak döndürür. Örneğin, document.getElementsByTagName('p’).

•**querySelector()**: CSS seçicileri kullanarak bir elementi seçmek için kullanılır. Bu, daha esnek bir yöntemdir ve document.querySelector('.sinifAdi') veya document.querySelector('#benzersizId') şeklinde kullanılabilir.

•**querySelectorAll()**: querySelector() gibi çalışır ancak belirtilen seçiciyle eşleşen tüm elementleri bir liste olarak döndürür.

https://onuraltuntas61.w3spaces.com/dom_nesne_.html

3.4. KULLANICILAR İLE ETKİLEŞİM KURMAYI KAVRAR.

A) KULLANICI ETKİLEŞİMİ OLUŞTURMAK İÇİN JAVASCRİPT PROGRAMLAMA DİLİ EVENTLERİNİN (OLAYLARININ) KULLANILMA ŞEKLİ AÇIKLANIR.

B) EVENT DİNLEYİCİLERİ EKLENEREK SAYFA ÜZERİNDE TEPKİ VERME İŞLEMLERİ AÇIKLANIR.

EVENT(OLAY) NEDİR?

Bir web sayfasındaki kullanıcı etkileşimleridir

1.Buton Tıklama:

1. click: Bir butona veya link gibi tıklanabilir herhangi bir öğeye tıklandığında tetiklenir.

2.Fare Olayları:

1. mouseover: Fare imlecini bir HTML öğesi üzerine getirdiğinde tetiklenir.
2. mouseout: Fare imlecini bir HTML öğesinin üzerinden çıkardığında tetiklenir.
3. mousedown: Bir öğeye tıklama işlemi başladığında (fare tuşu basılı tutulduğunda) tetiklenir.
4. mouseup: Bir öğede tıklama işlemi bittiğinde (fare tuşu serbest bırakıldığında) tetiklenir.
5. mousemove: Fare imleci bir öğenin üzerinde hareket ettirildiğinde tetiklenir.

3.Klavye Olayları:

1. keydown: Bir tuşa basıldığında tetiklenir.
2. keyup: Bir tuş basılı tutulduktan sonra serbest bırakıldığında tetiklenir.
3. keypress: Bir tuşa basıldığında ve tuşa basılı tutulduğu sürece (eski tarayıcılarda) tetiklenir.

3.4. KULLANICILAR İLE ETKİLEŞİM KURMAYI KAVRAR.

A) KULLANICI ETKİLEŞİMİ OLUŞTURMAK İÇİN JAVASCRİPT PROGRAMLAMA DİLİ EVENTLERİNİN (OLAYLARININ) KULLANILMA ŞEKLİ AÇIKLANIR.

B) EVENT DİNLEYİCİLERİ EKLENEREK SAYFA ÜZERİNDE TEPKİ VERME İŞLEMLERİ AÇIKLANIR.

EVENT(OLAY) NEDİR?

Bir web sayfasındaki kullanıcı etkileşimleridir.

Form Olayları:

1. submit: Bir form gönderildiğinde tetiklenir.
2. change: Bir form elemanı (örneğin, metin kutusu, açılır menü vb.) değiştirildiğinde tetiklenir.
3. focus: Bir form elemanı odaklandığında (seçildiğinde) tetiklenir.
4. blur: Bir form elemanı odak dışı bırakıldığında tetiklenir.

Dokunmatik Olaylar:

1. touchstart: Kullanıcı bir dokunmatik cihazda bir öğeye dokunduğunda tetiklenir.
2. touchmove: Bir öğe üzerinde bir dokunma hareket ederken tetiklenir.
3. touchend: Bir dokunma işlemi sona erdiğinde tetiklenir.

Pencere Olayları:

1. load: Bir sayfa tamamen yüklendiğinde tetiklenir.
2. resize: Pencere boyutlandırıldığında tetiklenir.
3. scroll: Kullanıcı bir sayfada kaydırma yaptığında tetiklenir.
4. unload: Bir sayfa kapatıldığında veya kullanıcı başka bir sayfaya navigasyon yaptığında tetiklenir (modern tarayıcılarda genellikle kullanım dışı).

Medya Olayları:

1. play: Bir medya oynatılmaya başladığında tetiklenir.
2. pause: Bir medya duraklatıldığında tetiklenir.
3. ended: Bir medyanın oynatılması sona erdiğinde tetiklenir.

.

3.4. KULLANICILAR İLE ETKİLEŞİM KURMAYI KAVRAR.

A) KULLANICI ETKİLEŞİMİ OLUŞTURMAK İÇİN JAVASCRİPT PROGRAMLAMA DİLİ EVENTLERİNİN (OLAYLARININ) KULLANILMA ŞEKLİ AÇIKLANIR.

B) EVENT DİNLEYİCİLERİ EKLENEREK SAYFA ÜZERİNDE TEPKİ VERME İŞLEMLERİ AÇIKLANIR.

EVENT(OLAY DİNLEYİCİSİ) NEDİR?

Bir olay gerçekleştiğinde belirli bir fonksiyonu çalıştırmaktır. Genellikle **addEventListener** fonksiyonu kullanılır.

.

```
document.getElementById(‘myBtn').addEventListener('click', function() {
  document.body.style.backgroundColor = 'yellow';
});
```

https://onuraltuntas61.w3spaces.com/addEventListener.html

3.5. ASENKRON PROGRAMLAMANIN WEB UYGULAMALARINDAKİ ÖNEMİNİ KAVRAR.

A) ASENKRON PROGRAMLAMA AÇIKLANARAK BUNUN WEB UYGULAMALARINDAKİ ÖNEMİ VURGULANIR.

B) ASENKRON PROGRAMLAMANIN SAYFA HIZI VE KULLANICI DENEYİMİNE SAĞLADIĞI KATKI VURGULANIR.

Asenkron programlama, bir web uygulamasının birden fazla işi aynı anda yapabilmesini sağlayan bir programlama tekniğidir.

Diyelim ki bir pizza dükkanısın. Bir müşteri pizza siparişi verdiğinde, pizzanın hazırlanması biraz zaman alır. Eğer senkron (yani asenkron olmayan) bir dükkân işletiyorsan, sipariş hazır olana kadar başka bir iş yapamazsın. Yani başka müşterilere bakamaz, telefonlara cevap veremezsin. Bütün işler, bir sipariş bitene kadar durur. Bu, verimsiz ve zaman kaybıdır.

Ancak asenkron bir dükkan işletiyorsan, bir siparişi aldığında pizza fırına girer ve sen o sırada başka siparişleri almaya, telefonlara bakmaya ya da diğer müşterilere yardımcı olmaya devam edebilirsin. Pizza fırında pişerken senin diğer işleri yapmana engel değildir. Böylece, birçok işi aynı anda yürütebilir ve zamanı verimli kullanabilirsin.

Web uygulamaları için de durum benzerdir. Örneğin, bir kullanıcı bir web sayfasında bir dosya indirmek istediğinde, indirme işlemi başladığında kullanıcının beklemesine gerek yoktur. Kullanıcı indirme işlemi arka planda devam ederken sayfada başka işlemler yapabilir. Bu, özellikle yavaş internet bağlantıları veya büyük dosyalar söz konusu olduğunda kullanıcı deneyimini büyük ölçüde iyileştirir.

3.5. ASENKRON PROGRAMLAMANIN WEB UYGULAMALARINDAKİ ÖNEMİNİ KAVRAR.

C) CALLBACK FONKSİYONLAR VE ASYNC/AWAİT GİBİ ASENKRON PROGRAMLAMA TEKNİKLERİ AÇIKLANIR.

Callback Fonksiyonlar:
Callback fonksiyonlar, bir işlemin tamamlanmasının ardından çağrılan fonksiyonlardır.

Genellikle uzun süren işlemler için kullanılırlar.

Örnek:
Bir veritabanından veri almak veya bir dosya okumak gibi işlemler bittiğinde, belirli bir kodun çalışmasını istiyorsanız callback fonksiyonları kullanılır.

```
function veriAl(callback) {
  // Farz edelim ki bu fonksiyon veritabanından veri alıyor
  setTimeout(() => { // setTimeout asenkron bir fonksiyondur.
    const veri = 'Veritabanından alınan bilgi';
    callback(veri); // Veri alındıktan sonra callback fonksiyonu
çağrılır
  }, 2000); // 2 saniye bekleme simülasyonu
}

veriAl(function(alinanVeri) {
  console.log(alinanVeri); // Veritabanından alınan bilgi
});
```

3.5. ASENKRON PROGRAMLAMANIN WEB UYGULAMALARINDAKİ ÖNEMİNİ KAVRAR.

C) CALLBACK FONKSİYONLAR VE ASYNC/AWAİT GİBİ ASENKRON PROGRAMLAMA TEKNİKLERİ AÇIKLANIR.

Promises Fonksiyonlar:
Karmaşık asenkron işlemler için "promises" kullanılır.

Örnek:
Promises, bir işlemin tamamlanmasının ardından bir değer döndüren ya da bir hata fırlatan nesnelerdir.

```
function veriAl() {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      const veri = 'Veritabanından alınan bilgi';
      resolve(veri); // İşlem başarılıysa resolve çağrılır
      // Eğer bir hata oluşursa reject ile hata yönetilebilir
    }, 2000);
  });
}

veriAl().then(alinanVeri => {
  console.log(alinanVeri); // Veritabanından alınan bilgi
}).catch(hata => {
  console.error(hata); // Bir hata olursa burada ele alınır
});
```

3.5. ASENKRON PROGRAMLAMANIN WEB UYGULAMALARINDAKİ ÖNEMİNİ KAVRAR.

C) CALLBACK FONKSİYONLAR VE ASYNC/AWAİT GİBİ ASENKRON PROGRAMLAMA TEKNİKLERİ AÇIKLANIR.

async/await Fonksiyonlar:

Promises kullanımını daha da basitleştiren ve kodunuzu senkron gibi görünmesini sağlayan modern bir Javascript özelliğidir. **async** fonksiyonlar otomatik olarak bir Promise döndürür ve **await** anahtar kelimesi, bir Promise'in sonucunun gelmesini bekler.

```
async function veriAlVeIsle() {
  try {
    const alinanVeri = await veriAl(); // veriAl Promise'ini bekler
    console.log(alinanVeri); // Veritabanından alınan bilgi
  } catch (hata) {
    console.error(hata); // Bir hata olursa burada ele alınır
  }
}

veriAlVeIsle();
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

A) JQUERY VE AJAX KAVRAMLARI AÇIKLANIR.

jQuery, internet sitelerini daha eğlenceli ve kolay kullanılır hale getirmek için tasarlanmış bir Javascript kütüphanesidir.

Web sayfasındaki elementlerle etkileşim, DOM (Document Object Model) manipülasyonu, animasyon, event handling (olay yönetimi) ve Ajax işlemleri için kullanılır.

Örneğin, bir butona tıklandığında bir resmin belirmesi ya da bir menünün açılıp kapanması gibi işlemleri kolay bir şekilde yapmanızı sağlar. Farklı internet tarayıcıları bazen aynı şeyleri farklı şekillerde yapar, ama jQuery bu farklılıkları düzeltir,

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

A) JQUERY VE AJAX KAVRAMLARI AÇIKLANIR.

Ajax, bir internet sitesinde bir şeylere tıklayıp yeni bilgiler istediğinizde, tüm sayfanın yeniden yüklenmesine gerek kalmadan sadece ihtiyacınız olan bilginin güncellenmesini sağlar. Örneğin, sosyal medyada yeni mesajlarınızı kontrol ederken, sayfayı yeniden yüklemek yerine, yeni mesajlar otomatik olarak sayfaya eklenir. Bu, internet sitelerini daha hızlı ve akıcı hale getirir.

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

A) JQUERY VE AJAX KAVRAMLARI AÇIKLANIR.

jQuery ve Ajax birlikte kullanıldığında, internet siteleri sadece daha hızlı ve kullanışlı olmakla kalmaz, aynı zamanda daha interaktif hale gelir. Bu, ziyaretçilere daha iyi bir deneyim sunar çünkü onlar beklerken tüm sayfanın yeniden yüklenmesini beklemek zorunda kalmazlar.

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

Jquery kullanmak için ilk olarak kütüphanesi yüklenir:

Query kütüphanesini web sayfanıza eklemenin birkaç yolu vardır. En yaygın yöntemlerden ikisi, bir CDN (Content Delivery Network) kullanarak veya kütüphaneyi doğrudan indirip projenize dahil etmektir. Google veya Microsoft gibi sağlayıcılardan bir CDN linki kullanabilirsiniz.

CDN Kullanarak jQuery Ekleme

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery Örneği</title>
    <!-- jQuery CDN'si -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
<body>

    <!-- Sayfanızın içeriği buraya gelecek -->

</body>
</html>
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

Jquery kullanmak için ilk olarak kütüphanesi yüklenir:

Kendi Hostinginizde jQuery'yi Barındırarak Ekleme

Eğer internet bağlantınızın hızı önemliyse veya özel bir jQuery sürümünü kullanmak istiyorsanız, jQuery'yi doğrudan indirip kendi sunucunuzda barındırabilirsiniz.

1.jQuery'yi jQuery'nin resmi web sitesinden indirin.
2.İndirdiğiniz jquery.min.js dosyasını web projenizin dosya yapısına ekleyin.
3.HTML dosyanızda, dosyayı <script> etiketi ile dahil edin.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery Örneği</title>
    <!-- Yerel jQuery Dosyası -->
    <script src="path/to/your/jquery.min.js"></script>
</head>
<body>

    <!-- Sayfanızın içeriği buraya gelecek -->

</body>
</html>
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

1. Eleman Seçimi ve İçerik Değiştirme

```
<p id="paragraf">Bu bir paragraf.</p>
```

Normal javascript kodları ile;

```
document.getElementById('paragraf').innerText= "Yeni içerik!"
```

Jquery kodları ile;

```
$('#paragraf').html('<strong>Yeni içerik!</strong>');
```

https://onuraltuntas61.w3spaces.com/jquery-icerik_degistirr.html

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

2. Eleman Ekleme

```
// Paragrafa yeni bir içerik ekleyelim
$('#paragraf').append(' ve burası eklenen metin.');
```

3. CSS İşlemleri

```
// Paragrafın yazı rengini değiştirelim
$('#paragraf').css('color', 'blue');
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

4. addClass() kullanımı:

Öncelikle bir CSS sınıfımız olsun:

```
.yaziStili {
    font-size: 50px;
    color: red;
}
```

Jquery ile bu sınıfı ekleyelim:

```
$('#paragraf').addClass('yaziStili');
```

https://onuraltuntas61.w3spaces.com/ADDCLASS.html

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

5. attr() kullanımı:

Bir bağlantı elemanımız olsun:

```
<a id="link" href="https://ornek.com">Örnek Site</a>
```

Bu bağlantının href özelliğini değiştirelim:

```
$('#link').attr('href', 'https://baskaornek.com');
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

B) JQUERY İLE HTML ELEMANLARININ MANİPÜLE EDİLME YÖNTEMLERİ AÇIKLANIR.

5. Olay yönetimi

on() kullanımı:

Bir butonumuz olsun:

```
<button id="buton">Tıkla</button>
```

Bu butona tıklama olayı ekleyelim:

```
$('#buton').on('click', function() {
    alert('Butona tıklandı!');
});
```

https://onuraltuntas61.w3spaces.com/jquery_buton.html

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

C) AJAX KULLANILARAK SUNUCU İLE VERİ ALIŞVERİŞİ YAPMA İŞLEMİ AÇIKLANIR.

Temel Ajax İşlem Adımları:

1.XMLHttpRequest (XHR) Nesnesi Oluşturmak: JavaScript'te, sunucu ile asenkron iletişim kurmak için XMLHttpRequest nesnesi kullanılır.

1.Bir Callback Fonksiyonu Tanımlamak: Sunucudan cevap geldiğinde ne yapılacağını belirleyen bir fonksiyon tanımlamak gereklidir.

2.İstek Göndermek: Oluşturulan XMLHttpRequest nesnesi üzerinden sunucuya bir istek gönderilir. Bu istek, veri gönderme veya sunucudan veri alma işlemi olabilir.

3.Yanıtı İşlemek: Sunucu, isteğe cevap olarak veri gönderir. Client tarafında, tanımlanan callback fonksiyonu bu veriyi yakalar ve işler. Örneğin, sunucudan alınan verilerle bir listeyi güncelleyebilir veya bir form gönderildikten sonra kullanıcıya bir bildirim gösterebilir.

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

C) AJAX KULLANILARAK SUNUCU İLE VERİ ALIŞVERİŞİ YAPMA İŞLEMİ AÇIKLANIR.

Temel Ajax İşlem Adımları:

```
$.ajax({
    url: 'sunucu-dosyasi.php', // İstek yapılacak sunucu dosyasının URL'si
    type: 'GET', // İstek türü, örneğin GET veya POST
    dataType: 'json', // Alınacak veri türü, örneğin json, xml, html
    success: function(cevap) {
        // Sunucudan başarılı bir yanıt geldiğinde yapılacak işlemler
        console.log("Sunucudan alınan veri: ", cevap);
    },
    error: function(xhr, status, error) {
        // İstek sırasında bir hata oluştuğunda yapılacak işlemler
        console.error("Bir hata oluştu: ", error);
    }
});
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

C) AJAX KULLANILARAK SUNUCU İLE VERİ ALIŞVERİŞİ YAPMA İŞLEMİ AÇIKLANIR.

Adım 1: PHP Dosyası Oluşturma (sunucu-dosyasi.php)
Bu PHP dosyası, basit bir JSON nesnesi döner. Dosyanızı sunucunuza yerleştirin:

```
<?php
// Header ayarını JSON tipinde yanıt dönecek şekilde ayarlıyoruz
header('Content-Type: application/json');

// Basit bir JSON nesnesi oluşturuyoruz
$response = array(
    'mesaj' => 'Merhaba, bu mesaj sunucudan geldi!'
);

// JSON nesnesini echo ile yazdırıyoruz
echo json_encode($response);
?>
```

3.6. JQUERY VE AJAX KÜTÜPHANELERİ İLE ETKİLEŞİMLİ ŞEKİLDE ÇALIŞMAYI KAVRAR.

C) AJAX KULLANILARAK SUNUCU İLE VERİ ALIŞVERİŞİ YAPMA İŞLEMİ AÇIKLANIR.

Adım 2: HTML ve jQuery Kullanarak Veriyi Alıp Gösterme

Şimdi, kullanıcıya sunucudan gelen mesajı gösterecek bir HTML sayfası oluşturalım. Bu sayfada jQuery kullanarak sunucu-dosyasi.php adresinden veriyi alıp, bir <p> etiketine yazdıracağız.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery Ajax Örneği</title>
    <!-- jQuery Kütüphanesi -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
<body>

<p id="mesaj">Mesaj burada görünecek...</p>

<script>
$(document).ready(function(){
    $.ajax({
        url: 'sunucu-dosyasi.php', // PHP dosyasının URL'si
        type: 'GET', // İstek türü
        dataType: 'json', // Alınacak veri türü
        success: function(data) {
            // Sunucudan gelen yanıtı #mesaj id'li <p> etiketine yazdır
            $('#mesaj').text(data.mesaj);
        },
        error: function(xhr, status, error) {
            // Hata durumunda hata mesajını yazdır
            console.error("Bir hata oluştu: ", error);
        }
    });
});
</script>

</body>
</html>
```

3.7. FETCH İŞLEMLERİNİ KULLANARAK ASENKRON ÇALIŞMAYI KAVRAR.

A) FETCH API KULLANILARAK SUNUCUDAN VERİLERİN NASIL ALINABİLECEĞİ AÇIKLANIR.

B) JSON FORMATINDA ÇEKİLEN VERİLERİN NASIL İŞLENECEĞİ VE KULLANILACAĞI AÇIKLANIR.

Fetch API, modern JavaScript'te sunucu ile asenkron veri alışverişi yapmak için kullanılan güçlü bir araçtır. XMLHttpRequest'a göre daha esnek ve güçlüdür.

Adım 1: Sunucu Dosyası Oluşturma (sunucu-dosyasi.php)

```php
<?php
// Basit bir JSON verisi oluşturuyoruz
$veri = [
    'mesaj' => 'Merhaba, bu bir test mesajıdır!',
    'tarih' => date('Y-m-d H:i:s')
];

// JSON verisini döndürüyoruz
header('Content-Type: application/json');
echo json_encode($veri);
?>
```

3.7. FETCH İŞLEMLERİNİ KULLANARAK ASENKRON ÇALIŞMAYI KAVRAR.

A) FETCH API KULLANILARAK SUNUCUDAN VERİLERİN NASIL ALINABİLECEĞİ AÇIKLANIR.

B) JSON FORMATINDA ÇEKİLEN VERİLERİN NASIL İŞLENECEĞİ VE KULLANILACAĞI AÇIKLANIR.

Fetch API, modern JavaScript'te sunucu ile asenkron veri alışverişi yapmak için kullanılan güçlü bir araçtır. XMLHttpRequest'a göre daha esnek ve güçlüdür.

Adım 2: Fetch API Kullanarak Veri Almak ve Göstermek

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Fetch API Örneği</title>
</head>
<body>

<p id="mesaj">Mesaj burada görünecek...</p>

<script>
// Fetch API kullanarak sunucu dosyasından veri alıyoruz
fetch('sunucu-dosyasi.php')
    .then(response => response.json()) // Gelen yanıtı JSON'a çeviriyoruz
    .then(data => {
        // JSON'dan alınan veriyi kullanarak <p> etiketini güncelliyoruz
        document.getElementById('mesaj').textContent = data.mesaj + " - " + data.tarih;
    })
    .catch(error => console.error('Bir hata oluştu:', error)); // Hata yakalama
</script>

</body>
</html>
```

3.7. FETCH İŞLEMLERİNİ KULLANARAK ASENKRON ÇALIŞMAYI KAVRAR.

C) FETCH İŞLEMLERİNİN WEB UYGULAMARINDAKİ KULLANILMA ŞEKİLLERİ ÖRNEKLERLE AÇIKLANIR.

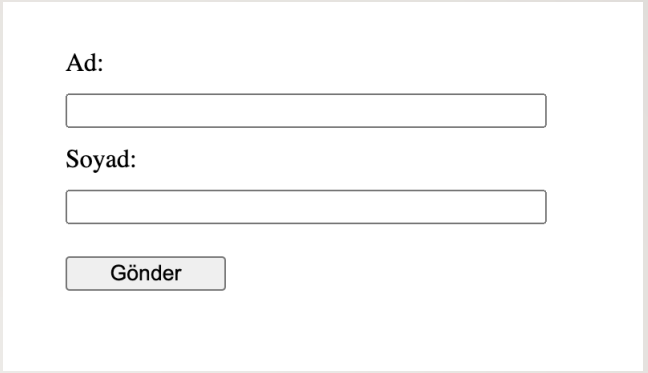
FETCH KULLANIM ŞEKİLLERİ

1. **Bir API'den Veri Çekme**
2. **Form Verilerini Gönderme**
3. **Dosya Yükleme**
4. **Kaynakların Önbelleğe Alınması**

3.7. FETCH İŞLEMLERİNİ KULLANARAK ASENKRON ÇALIŞMAYI KAVRAR.

C) FETCH İŞLEMLERİNİN WEB UYGULAMARINDAKİ KULLANILMA ŞEKİLLERİ ÖRNEKLERLE AÇIKLANIR.

Form Verilerini Gönderme




```
<form id="ogrenciFormu">

    <label for="ad">Ad:</label>
    <input type="text" id="ad" name="ad" required>

    <label for="soyad">Soyad:</label>
    <input type="text" id="soyad" name="soyad" required>

    <button type="button" onclick="formuIsle()">Gönder</button>

</form>
```

3.7. FETCH İŞLEMLERİNİ KULLANARAK ASENKRON ÇALIŞMAYI KAVRAR.

C) FETCH İŞLEMLERİNİN WEB UYGULAMARINDAKİ KULLANILMA ŞEKİLLERİ ÖRNEKLERLE AÇIKLANIR.

Form Verilerini Gönderme

```
<script>

    function formuIsle() {
        var formData = {
        ad: document.getElementById('ad').value,
        soyad: document.getElementById('soyad').value
        };

        fetch('sunucu.php', {
            method: 'POST’,
            headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify(formData),
        })

        .then(response => response.json())
        .then(data => console.log('Başarılı:', data))
        .catch(error => console.error('Hata:', error));
    }

</script>
```

3.8. JAVASCRİPT PROGRAMLAMA DİLİ KODLARINI DAHA MODÜLER HÂLE GETİREREK KULLANMANIN ÖNEMİNİ KAVRAR.

A) KODU PARÇALARA BÖLEREK MODÜLER ÇALIŞMANIN ÖNEMİ VURGULANIR

Modüler çalışmanın önemi şu noktalarda vurgulanabilir:

1.Düzen ve Organizasyon: Her modül, sadece belli işler için sorumludur. Bu, tıpkı okul çantanızı düzenler gibi, her şeyin yerli yerinde ve aradığınızı kolayca bulabileceğiniz anlamına gelir.

2.Hata Ayıklama Kolaylığı: Bir sorun olduğunda, tüm kodu değil, sadece sorunlu parçayı (modülü) kontrol etmek gerekir. Bu, tıpkı bir sınavda yanlış cevapladığınız bir soruyu bulup düzeltmek gibi daha az zaman alır ve daha az strese neden olur.

3.8. JAVASCRİPT PROGRAMLAMA DİLİ KODLARINI DAHA MODÜLER HÂLE GETİREREK KULLANMANIN ÖNEMİNİ KAVRAR.

A) KODU PARÇALARA BÖLEREK MODÜLER ÇALIŞMANIN ÖNEMİ VURGULANIR

Modüler çalışmanın önemi şu noktalarda vurgulanabilir:

1.Yeniden Kullanılabilirlik: Bir modülü, tıpkı okul projelerinizde yaptığınız bir araştırmayı farklı dersler için kullanabilir gibi, farklı projelerde kullanabilirsiniz. Bu, zamandan tasarruf sağlar ve çabalarınızın değerini artırır.

2.Ekip Çalışması: Bir grup projesinde herkesin kendi bölümü üzerinde çalışması gibi, modüller üzerinde birden fazla geliştirici bağımsız olarak çalışabilir. Bu da iş bölümünü ve işbirliğini kolaylaştırır.

3.8. JAVASCRİPT PROGRAMLAMA DİLİ KODLARINI DAHA MODÜLER HÂLE GETİREREK KULLANMANIN ÖNEMİNİ KAVRAR.

B) JAVASCRİPT PROGRAMLAMA DİLİ MODÜLLERİNİ OLUŞTURUP KULLANMA KONUSUNDA ÖRNEK ÇALIŞMALAR YAPILIR.

Örnek 1: Bir Matematik Modülü Oluşturma
math.js adında bir dosya oluşturarak başlayalım.

```
// math.js
function topla(x, y) {
  return x + y;
}

function cikar(x, y) {
  return x - y;
}

export { topla, cikar };
```

3.8. JAVASCRİPT PROGRAMLAMA DİLİ KODLARINI DAHA MODÜLER HÂLE GETİREREK KULLANMANIN ÖNEMİNİ KAVRAR.

B) JAVASCRİPT PROGRAMLAMA DİLİ MODÜLLERİNİ OLUŞTURUP KULLANMA KONUSUNDA ÖRNEK ÇALIŞMALAR YAPILIR.

Örnek 2: Matematik Modülünü Kullanma

app.js adında başka bir dosya oluşturalım ve math.js modülünü bu dosyada kullanalım.

```
// app.js
import { topla, cikar } from './math.js';

console.log(topla(5, 3)); // 8
console.log(cikar(5, 3)); // 2
```

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

A) NODEJS İLE SUNUCU TARAFINDA JAVASCRİPT PROGRAMLAMA DİLİNİN GELİŞTİRİLME İŞLEMİ AÇIKLANIR.

Node.js, JavaScript'i sadece internet tarayıcıları yerine her türlü ortamda çalıştırmak için kullanılan bir platformdur. Genellikle web sunucuları oluşturmak için kullanılır.

Diyelim ki, siz bir parti **MASTER ŞEF**'siniz ve diğer çalışanlara nasıl yemek yapacakları konusunda ne yapacaklarını söyleyen bir talimatnameniz var. Node.js, internet dünyasındaki bu talimatnamedir. İnternet üzerindeki bilgisayarlar arasında iletişim kurarak veri alışverişinde bulunur ve herkese ne yapacaklarını söyler. Bu veriler genellikle web sayfaları, kullanıcı bilgileri veya başka türden veriler olabilir.

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

A) NODEJS İLE SUNUCU TARAFINDA JAVASCRİPT PROGRAMLAMA DİLİNİN GELİŞTİRİLME İŞLEMİ AÇIKLANIR.

Node.js ile çalışırken, JavaScript kullanarak komutlar yazarız.

Bu komutlar, sunucunun ne yapacağını belirler.

Örnek: Bir kullanıcının web sayfasına girmesi isteği geldiğinde, Node.js bu isteği alır ve kullanıcıya gösterilecek web sayfasını geri gönderir.

Node.js ile JavaScript'i kullanarak, internet üzerinde çalışan bir bilgisayara (yani bir sunucuya) komutlar verebiliriz.

Bu sayede, web sitelerinin arka planında çalışan birçok işlemi yönetebiliriz.

Bu işlemler tarayıcıda görünmeyebilir ama internetin sorunsuz çalışması için çok önemlidir.

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

A) NODEJS İLE SUNUCU TARAFINDA JAVASCRİPT PROGRAMLAMA DİLİNİN GELİŞTİRİLME İŞLEMİ AÇIKLANIR.

NODE.JS UYGULAMA ÖRNEĞİ

1.İlk olarak, bilgisayarınızda Node.js yüklü olduğundan emin olun.
2.Bir metin düzenleyici açın ve aşağıdaki kodu yazın. Bu kod, basit bir web sunucusu oluşturur:

```
// server.js dosyası
const http = require('http'); // Node.js'in HTTP modülünü dahil ediyoruz.

// Sunucuyu oluşturuyoruz.
const server = http.createServer((request, response) => {
  response.statusCode = 200; // HTTP durum kodu 200 OK.
  response.setHeader('Content-Type', 'text/plain'); // İçerik tipini düz metin olarak ayarlıyoruz.
  response.end('Merhaba Lise!'); // Kullanıcıya gönderilecek yanıt.
});

// Sunucuyu dinlemek için bir port belirliyoruz. Örneğin 3000.
const port = 3000;
server.listen(port, () => {
  console.log(`Sunucu port ${port} üzerinde çalışıyor.`);
});
```

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

A) NODEJS İLE SUNUCU TARAFINDA JAVASCRİPT PROGRAMLAMA DİLİNİN GELİŞTİRİLME İŞLEMİ AÇIKLANIR.

NODE.JS UYGULAMA ÖRNEĞİ

3.Bu dosyayı server.js adıyla kaydedin.
4.Komut istemcisini açın ve bu dosyanın bulunduğu klasöre gidin.
5.node server.js komutunu yazarak sunucuyu başlatın.
6.Web tarayıcınıza gidin ve adres çubuğuna localhost:3000 yazıp enter'a basın.
7.Tarayıcınızda "Merhaba Lise!" mesajını görmelisiniz.

Bu örnek, Node.js kullanarak bir sunucu oluşturmanın en temel şeklidir. Sunucu, kullanıcı tarayıcıdan bir istek gönderdiğinde, yani localhost:3000 adresine gittiğinde, bu isteği alır ve cevap olarak kullanıcıya 'Merhaba Lise!' metnini gönderir. Bu işlem, internet üzerindeki bilgi alışverişinin çok temel bir örneğidir.

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

B) NODEJS İLE DOSYA VE AĞ İŞLEMLERİ GİBİ ARKAPLAN İŞLEMLERİNİN GERÇEKLEŞTİRİLEBİLECEĞİ VURGULANIR.

Dosya İşlemleri:
Node.js, JavaScript ile dosya okuma, yazma, düzenleme ve silme gibi işlemleri yapmanıza olanak tanır.

Örneğin, bir sınav sonucunu bir dosyaya kaydedebilir veya bir ders planını dosyadan okuyabilirsiniz. Node.js'in 'fs' (file system) modülü bu tür işlemleri kolaylaştırır.

ÖRNEK:

```
const fs = require('fs'); // fs modülünü dahil ediyoruz.

fs.writeFile('ornek.txt', 'Merhaba', (hata) => {
  if (hata) throw hata;
  console.log('Dosyaya başarıyla yazıldı!');
});
```

3.9. NODEJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

B) NODEJS İLE DOSYA VE AĞ İŞLEMLERİ GİBİ ARKAPLAN İŞLEMLERİNİN GERÇEKLEŞTİRİLEBİLECEĞİ VURGULANIR.

Ağ İşlemleri:
Node.js ile, internet üzerinde bilgi alıp gönderebilirsiniz. Bir web sitesinden veri çekebilir, e-posta gönderebilir veya bir chat uygulaması yapabilirsiniz. Bunlar için Node.js'in ağ ile ilgili modüllerini kullanırsınız, örneğin 'http', 'https', 'net' gibi

Diğer Arkaplan İşlemleri:
Node.js, veritabanı işlemleri, oturum yönetimi ve kullanıcı doğrulama gibi arkaplan işlemlerini de yapabilir. Veritabanlarına bağlanabilir, kullanıcı bilgilerini güvende tutabilir ve web uygulamanızın güvenliğini sağlayabilirsiniz.

3.10. COMMONJS MODÜLÜNÜ KULLANARAK ARKAPLANDA İŞLEMLER YAPILABİLECEĞİNİ KAVRAR.

COMMONJS MODÜL SİSTEMİ KULLANILARAK DOSYALARIN MODÜLER HÂLE GETİRİLEBİLECEĞİ VURGULANIR.

CommonJS, Node.js'de kullanılan bir modül sistemidir. Bunu bir okul dolabı gibi düşünebilirsiniz; her ders için farklı kitaplarınız var ve sadece ihtiyacınız olan dersin kitabını alırsınız.

Örneğin, bir matematik işlevlerini içeren bir matematik.js adında bir dosyanız olsun. Bu dosya içinde bir toplama işlevi yazdınız:

```
// matematik.js

function topla(a, b) {
  return a + b;
}

module.exports = topla; // topla fonksiyonunu dışa aktarıyoruz.
```

Başka bir dosyada bu topla fonksiyonunu kullanmak istediğimizde, şöyle yaparız:

```
// app.js

const topla = require('./matematik'); // matematik.js'den topla fonksiyonunu içe aktarıyoruz.

console.log(topla(2, 3)); // 5
```

3.11. NPM ARACILIĞI İLE HARİCİ KÜTÜPHANELER KULLANARAK MODÜL EKLEME İŞLEMİNİ KAVRAR.

A) NPM (NODE PACKAGE MANAGER) ARACILIĞIYLA HARİCİ JAVASCRİPT KÜTÜPHANELERİNİN PROJEYE EKLENME ŞEKLİ AÇIKLANIR

NPM NEDİR?

Npm (Node Package Manager), JavaScript için bir paket yöneticisidir ve genellikle Node.js projelerinde harici kütüphaneleri projeye eklemek için kullanılır.

Projeye bir kütüphane eklemek için genellikle aşağıdaki adımlar izlenir:

1. Npm'in Yüklü Olması

Npm, Node.js ile birlikte gelir. Dolayısıyla, npm'i kullanabilmek için öncelikle bilgisayarınıza Node.js'in yüklü olması gerekir. Node.js ve npm'in yüklü olup olmadığını kontrol etmek için terminal veya komut istemcisinde aşağıdaki komutları çalıştırabilirsiniz:

```
node -v
npm -v
```

3.11. NPM ARACILIĞI İLE HARİCİ KÜTÜPHANELER KULLANARAK MODÜL EKLEME İŞLEMİNİ KAVRAR.

A) NPM (NODE PACKAGE MANAGER) ARACILIĞIYLA HARİCİ JAVASCRİPT KÜTÜPHANELERİNİN PROJEYE EKLENME ŞEKLİ AÇIKLANIR

B) HARİCİ KÜTÜPHANELERİN PROJEYE DAHİL EDİLMESİ İŞLEMLERİ AÇIKLANIR

2. Proje Dizinine Geçiş

Terminal veya komut istemcisini açın ve npm kullanarak kütüphane eklemek istediğiniz Node.js projesinin dizinine geçin. Bu, cd komutu ile yapılabilir.

```
cd yol/proje-dizini
```

3. package.json Dosyasının Oluşturulması

Her Node.js projesinde, projenin bağımlılıklarını ve konfigürasyonlarını içeren bir package.json dosyası bulunur. Eğer bu dosya mevcut değilse, aşağıdaki komut ile oluşturulabilir:

```
npm init
```

Bu komut, projenin ismi, versiyonu, açıklaması gibi bilgileri sormak için bir dizi soru yöneltecektir. Tüm soruları yanıtladıktan sonra package.json dosyası oluşturulur.

3.11. NPM ARACILIĞI İLE HARİCİ KÜTÜPHANELER KULLANARAK MODÜL EKLEME İŞLEMİNİ KAVRAR.

A) NPM (NODE PACKAGE MANAGER) ARACILIĞIYLA HARİCİ JAVASCRİPT KÜTÜPHANELERİNİN PROJEYE EKLENME ŞEKLİ AÇIKLANIR

B) HARİCİ KÜTÜPHANELERİN PROJEYE DAHİL EDİLMESİ İŞLEMLERİ AÇIKLANIR

4. Kütüphanenin Eklenmesi

Örnek olarak express kütüphanesini ekleyelim

```
npm install express
```

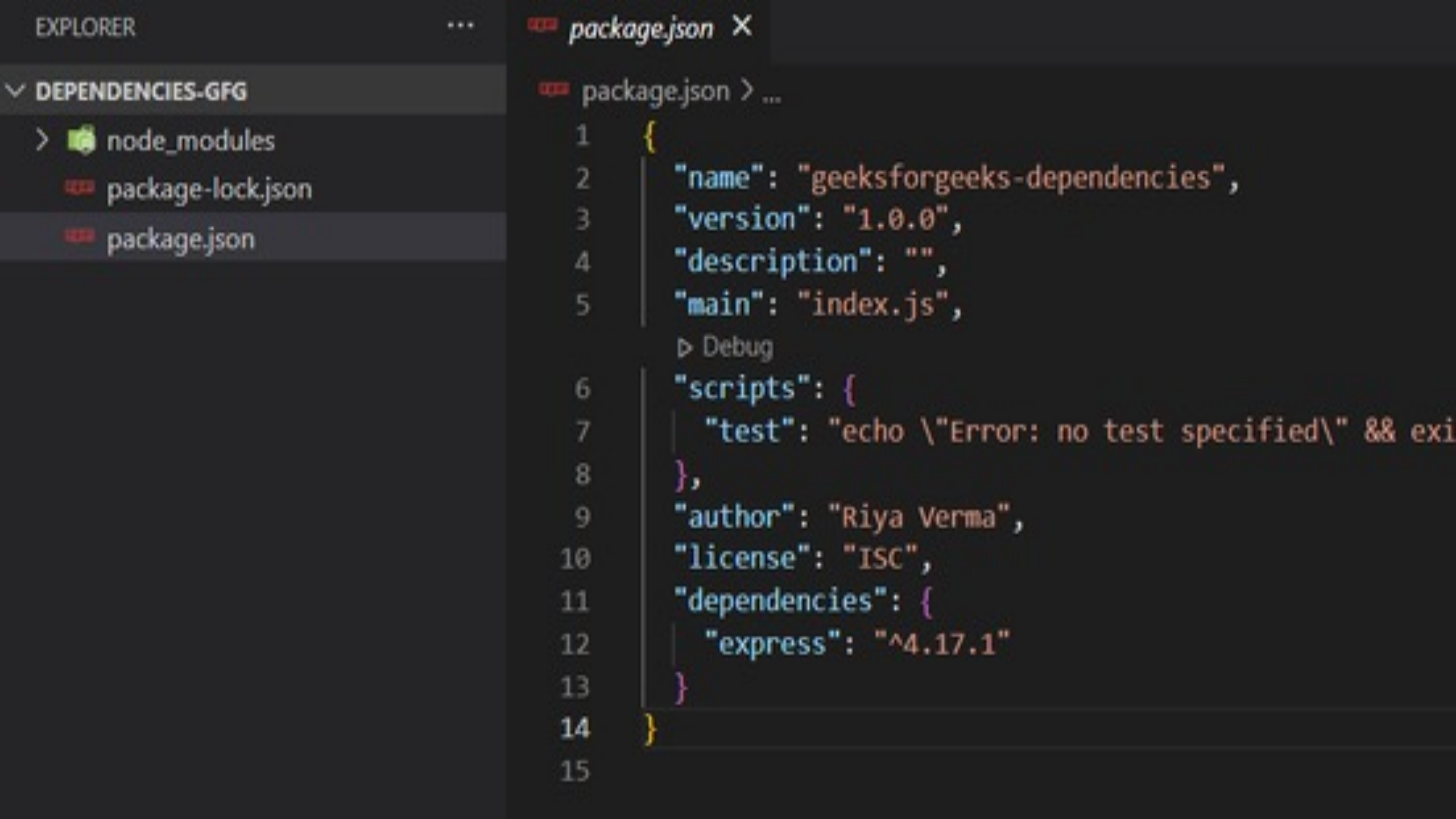
Bu komut, belirtilen kütüphaneyi indirir ve projenin **node_modules** dizinine ekler. Ayrıca, kütüphaneyi **package.json** dosyasındaki **dependencies** bölümüne ekler.

6. Kütüphaneyi Kaydetme

Eğer kütüphaneyi sadece geliştirme aşamasında kullanacaksanız, --save-dev seçeneği ile kütüphaneyi devDependencies altında kaydedebilirsiniz:

```
npm install <kutuphane-adi> --save-dev
```

Bu adımlar, npm aracılığıyla JavaScript projenize harici kütüphaneleri nasıl ekleyeceğinizi açıklar. Bu işlemler, projenizin geliştirme sürecini kolaylaştırır ve ihtiyacınız olan kütüphanelere hızlı bir şekilde erişim sağlar.


EXPLORER
DEPENDENCIES-GFG
node_modules
package-lock.json
package.json
package.json
package.json > ...
{
"name": "geeksforgeeks-dependencies",
"version": "1.0.0",
"description": "",
"main": "index.js",
Debug
"scripts": {
"test": "echo \"Error: no test specified\" && exi
},
"author": "Riya Verma",
"license": "ISC",
"dependencies": {
"express": "^4.17.1"
}
}

3.11. NPM ARACILIĞI İLE HARİCİ KÜTÜPHANELER KULLANARAK MODÜL EKLEME İŞLEMİNİ KAVRAR.

C) HARİCİ KÜTÜPHANELERİN PROJE İÇERİSİNDE GÜNCELLENMESİ GİBİ YÖNETİM İŞLEMLERİ AÇIKLANIR.

Tek Bir Kütüphanenin Güncellenmesi:

```
npm update <kutuphane-adi>
```

Tüm Kütüphanelerin Güncellenmesi:

```
npm update
```

Bu komut, package.json dosyasındaki dependencies ve devDependencies altında listelenen tüm kütüphaneleri günceller.

Güvenlik Güncellemeleri:

```
npm audit
```

Güvenlik açıklarını otomatik olarak düzeltmek için:

```
npm audit fix
```

3.11. NPM ARACILIĞI İLE HARİCİ KÜTÜPHANELER KULLANARAK MODÜL EKLEME İŞLEMİNİ KAVRAR.

C) HARİCİ KÜTÜPHANELERİN PROJE İÇERİSİNDE GÜNCELLENMESİ GİBİ YÖNETİM İŞLEMLERİ AÇIKLANIR.

Kütüphanelerin Kaldırılması

```
npm uninstall <kutuphane-adi>
```

BAZI NODE.JS KÜTÜPHANELERİ

1.Express.js:

Express, Node.js için kullanılan bir web uygulama çatısıdır. Web ve mobil uygulamalar geliştirmek içinkullanılır. . API'lar oluşturmanın yanı sıra, web uygulamaları için hızlı ve kolay bir yol sunar.

2. **React (React.js veya ReactJS):**

React, kullanıcı arayüzleri (UI) oluşturmak için kullanılan bir JavaScript kütüphanesidir. Facebook tarafından geliştirilmiştir ve genellikle web uygulamalarında frontend geliştirme için kullanılır.

3. **Socket.IO:**

Gerçek zamanlı web uygulamaları geliştirmek için idealdir. Örneğin, çevrimiçi sohbet uygulamaları veya gerçek zamanlı veri akışı gerektiren uygulamalar Socket.IO kullanabilir.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

A) YAYGIN JAVASCRİPT PROGRAMLAMA DİLİ FRAMEWORK'LERİ AÇIKLANIR VE BU FRAMEWORK'LERİN ÖNEMİ VURGULANIR.

JavaScript, web sitelerini canlı ve etkileşimli hale getiren bir programlama dilidir. Framework'ler ise, bu dil ile çalışırken işleri kolaylaştıran bir nevi hazır araç kutularıdır.

BAZI JAVASCRIPT FRAMEWORKLERİ

1. React

•**Nedir?** Bir çeşit yapboz kutusu gibi düşünebilirsin. Web sitenin farklı parçalarını (örneğin bir arama çubuğu, bir liste veya bir buton) React ile kolayca oluşturabilir ve bunları tekrar tekrar kullanabilirsin.

•**Neden Önemli?** Çünkü bu, aynı parçaları farklı yerlerde kullanarak zaman kazanmanı sağlar. Ayrıca, herkesin sevdiği Facebook ve Instagram gibi siteler de React kullanıyor.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

A) YAYGIN JAVASCRİPT PROGRAMLAMA DİLİ FRAMEWORK'LERİ AÇIKLANIR VE BU FRAMEWORK'LERİN ÖNEMİ VURGULANIR.

BAZI JAVASCRIPT FRAMEWORKLERİ

2. Angular

•**Nedir?** Angular, bir web sitesini baştan sona kurman için gereken her şeyi içeren büyük bir araç kutusudur. Hem sitenin görünümünü hem de çalışma şeklini kontrol edebilirsin.

•**Neden Önemli?** Çünkü büyük projelerde, her şeyin düzenli ve bir arada olması gerektiğinde Angular çok yardımcı olur. Google gibi dev bir şirket tarafından destekleniyor olması da cabası.

3. Vue.js

•**Nedir?** Vue.js, React gibi, web sitenin parçalarını oluşturmak için kullanılan bir araç. Ancak daha basit ve daha esnek olduğu söylenir. Öğrenmesi kolay ve hızlı başlamak isteyenler için mükemmeldir.

•**Neden Önemli?** Küçük ve orta ölçekli projeler için idealdir ama gerektiğinde büyük projelerde de kullanılabilir. İnsanlar onu seviyor çünkü kolay ve eğlenceli.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

B) FRAMEWORK'LERİN KULLANILMA ŞEKLİ VE WEB UYGULAMALARINI GELİŞTİRİRKEN ÜSTLENDİĞİ ROL AÇIKLANIR.

ÖRNEK BİR UYGULAMA GÖSTERME

Express.js ile Basit Bir Web Sunucusu Oluşturma:

1- Öncelikle NODE.JS paketini bilgisayara yükle

2- Bir Node.js projesi oluştur. Bunun için bir klasör/dizin oluştur.

3. Proje dizinine git ve ilk ayarları yap

```
cd my-express-app
npm init -y
```

npm init -y komutu, varsayılan değerlerle bir package.json dosyası oluşturur.

4. Express.js'i Projene Ekle:

```
npm install express
```

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

B) FRAMEWORK'LERİN KULLANILMA ŞEKLİ VE WEB UYGULAMALARINI GELİŞTİRİRKEN ÜSTLENDİĞİ ROL AÇIKLANIR.

ÖRNEK BİR UYGULAMA GÖSTERME

5. Web Sunucusu Kodunu Yaz:
Projende index.js adında bir dosya oluştur ve aşağıdaki kodu içine yapıştır:

```
const express = require('express');
const app = express();
const port = 3000;

app.get('/', (req, res) => {
  res.send('Merhaba Express.js!');
});

app.listen(port, () => {
  console.log(`Sunucu http://localhost:${port} adresinde
çalışıyor.`);
});
```

Bu kod, 3000 numaralı portta çalışan basit bir web sunucusu oluşturur. Bu sunucu, tarayıcıdan http://localhost:3000 adresine bir istek yapıldığında "Merhaba Express.js!" mesajını döndürür.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

B) FRAMEWORK'LERİN KULLANILMA ŞEKLİ VE WEB UYGULAMALARINI GELİŞTİRİRKEN ÜSTLENDİĞİ ROL AÇIKLANIR.

ÖRNEK BİR UYGULAMA GÖSTERME

6. Sunucuyu Çalıştır:
Terminalde aşağıdaki komutu kullanarak sunucunu çalıştır:

```
node index.js
```

Artık tarayıcını açıp http://localhost:3000 adresine gittiğinde, sunucunun döndürdüğü "Merhaba Express.js!" mesajını görebilirsin.

Bu basit örnek, bir framework'ün web uygulaması geliştirmede nasıl kullanılabileceğini ve ne kadar kolay ve hızlı olduğunu gösterir. Framework'ler, web uygulamaları geliştirmek için sağlam bir temel ve yapı sağlar, böylece geliştiricilerin daha verimli çalışmalarına yardımcı olur.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

C) FRAMEWORK'LERİN AVANTAJ VE DEZAVANTAJLARI BELİRTİLİR.

Avantajları:

1.Hızlı Geliştirme: Çoğu framework, sık kullanılan özellikler için hazır kod blokları sunar, bu da geliştirme sürecini hızlandırır.

2.Güvenlik: İyi tasarlanmış framework'ler, güvenlik açıklarına karşı koruma sağlar ve güncel güvenlik uygulamalarını içerir.

3.Bakım Kolaylığı: Kodun modüler yapısı sayesinde, uygulama üzerinde yapılan değişikliklerin yönetimi ve bakımı daha kolaydır.

4.En İyi Uygulamalar: Framework'ler, endüstri standartları ve en iyi uygulamalara uygun şekilde geliştirilmiştir, bu da kod kalitesini artırır.

5.Topluluk Desteği: Popüler framework'ler genellikle büyük ve aktif topluluklara sahiptir, bu da sorunlarla karşılaştığınızda yardım alabileceğiniz anlamına gelir.

3.12. JAVASCRİPT PROGRAMLAMA DİLİNDEKİ YAYGIN FRAMEWORK'LERİN WEB GELİŞTİRMEDEKİ ROLÜNÜ KAVRAR.

C) FRAMEWORK'LERİN AVANTAJ VE DEZAVANTAJLARI BELİRTİLİR.

Dezavantajları:

1.Öğrenme Eğrisi: Her framework'ün kendine has özellikleri ve kullanım şekilleri vardır, bu da yeni başlayanlar için öğrenme sürecini zorlaştırabilir.

2.Esneklik Sınırlamaları: Bazı framework'ler, belirli bir yapıyı veya geliştirme modelini zorunlu kılar, bu da projenin gereksinimlerine göre esneklik sağlamada sınırlamalar yaratabilir.

3.Performans Sorunları: Framework'ler ekstra kod ve işlevsellik getirebilir, bu da bazen gereksiz yere kaynak tüketimi ve performans düşüklüğüne yol açabilir.

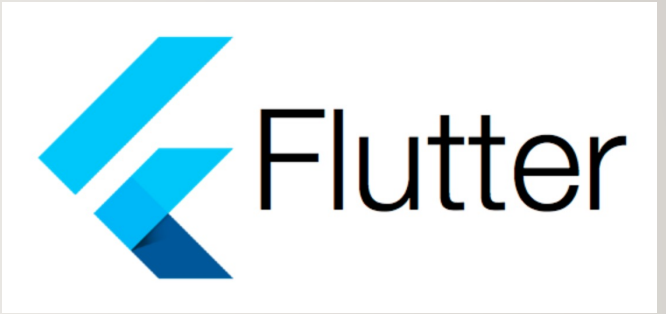
4.Bağımlılık: Bir framework'e fazla bağımlı olmak, ileride framework güncellendiğinde veya desteklenmediğinde problemlere yol açabilir.

5.Aşırı Genellemeler: Bazı framework'ler genel kullanım senaryolarını hedef alır ve bu da özelleştirilmiş ihtiyaçlar için fazladan ayarlama gerektirebilir.

3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

A) AÇIK KAYNAK KODLU MOBİL UYGULAMA GELİŞTİRME FRAMEWORKLERİNİN NE OLDUĞU VE NASIL ÇALIŞTIĞI AÇIKLANIR.

Popüler Açık Kaynak Kodlu Mobil Uygulama Geliştirme Framework'leri:



3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

A) AÇIK KAYNAK KODLU MOBİL UYGULAMA GELİŞTİRME FRAMEWORKLERİNİN NE OLDUĞU VE NASIL ÇALIŞTIĞI AÇIKLANIR.

Popüler Açık Kaynak Kodlu Mobil Uygulama Geliştirme Framework'leri:

1.React Native: Facebook tarafından geliştirilen React Native, JavaScript kullanarak hem iOS hem de Android için yerel benzeri uygulamalar oluşturmanıza olanak tanır.

2.Flutter: Google tarafından geliştirilen Flutter, Dart programlama dili kullanarak yüksek performanslı ve görsel olarak çekici mobil uygulamalar oluşturmak için kullanılır.

1.Apache Cordova (önceden PhoneGap olarak bilinir): Web teknolojileri (HTML, CSS ve JavaScript) kullanarak mobil uygulamalar geliştirmek için bir framework'tür.

3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

B) WEB GELİŞTİRME BİLGİSİNİN MOBİL UYGULAMA GELİŞTİRMEDE KULLANILAN FRAMEWORK'LERE UYARLANARAK MOBİL UYGULAMA GELİŞTİRME SÜRECİNE SAĞLADIĞI KATKI VURGULANIR.

Daha Hızlı Geliştirme Süreci

Maliyet Etkinliği

Geniş Erişim ve Esneklik

Güçlü Topluluk ve Kaynaklar

3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

C) MOBİL UYGULAMA GELİŞTİRME SÜREÇLERİ AÇIKLANIR.

1. Fikir ve Araştırma

Fikir: Süreç, genellikle bir problemi çözme veya kullanıcılara değer sunma fikriyle başlar.
Pazar Araştırması: Benzer uygulamalar, hedef kitle ve pazar ihtiyaçları araştırılır.

2. Planlama

Uygulama Kapsamı: Uygulamanın özellikleri, işlevleri ve kullanıcı akışı belirlenir.

Teknoloji Seçimi: Uygulamanın yerel (iOS/Android için özel), hibrit veya platformlar arası (birden fazla platformda çalışabilen) olup olmadığına karar verilir.

3. Tasarım

Kullanıcı Arayüzü (UI) Tasarımı: Uygulamanın görsel tasarımı, kullanıcı arayüzü bileşenleri ve kullanıcı deneyimi (UX) düşünülerek yapılır.

Prototipleme: Uygulamanın nasıl çalışacağını gösteren temel, etkileşimli prototipler oluşturulur.

3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

C) MOBİL UYGULAMA GELİŞTİRME SÜREÇLERİ AÇIKLANIR.

4. Geliştirme

Kodlama: Seçilen teknoloji stack'ine göre uygulamanın ön yüzü (frontend) ve arka yüzü (backend) kodlanır.

Ara Bağlantılar: Uygulama, gerekli harici servislerle (API'lar, veritabanları vb.) entegre edilir.

5. Test Etme

Birim Testleri: Kodun her parçasının beklenen şekilde çalıştığından emin olmak için testler yapılır.

Kullanılabilirlik Testleri: Gerçek kullanıcılar uygulamayı test ederek geri bildirim sağlar.

Performans Testleri: Uygulamanın farklı cihazlarda ve ağ koşullarında nasıl performans gösterdiğini ölçmek için testler yapılır.

3.13. FARKLI AÇIK KAYNAK KODLU WEB GELİŞTİRME FRAMEWORK'LERİNİN MOBİL UYGULAMA GELİŞTİRMEDE NASIL KULLANILACAĞINI KAVRAR.

C) MOBİL UYGULAMA GELİŞTİRME SÜREÇLERİ AÇIKLANIR.

6. Dağıtım

•**Mağazalara Yükleme:** Uygulama, Apple App Store, Google Play Store gibi uygulama mağazalarına yüklenir.

•**Pazarlama ve Promosyon:** Uygulamanın hedef kitleye ulaşması için pazarlama stratejileri uygulanır.

7. Bakım ve Güncellemeler

•**Geri Bildirim Toplama:** Kullanıcılardan gelen geri bildirimler analiz edilir.

•**Güncellemeler:** Uygulama, yeni özellikler eklenerek veya hatalar düzeltilerek düzenli olarak güncellenir.

3.14. GELİŞMEKTE OLAN ECMASCRİPT İLE JAVASCRİPT PROGRAMLAMA DİLİNİN GÜNCEL STANDARTLARA UYUMLU ŞEKİLDE ÇALIŞABİLECEĞİNİ KAVRAR.

A) ECMASCRİPT'İN JAVASCRİPT PROGRAMLAMA DİLİNİN GÜNCELLEME VE YENİ ÖZELLİKLERİNİ İÇERMESİ DURUMU AÇIKLANIR.

ECMAScript

EcmaScript Nedir? *European Computer Manufacturers Association*

- ECMA, bilgisayar donanımları, yazılım dilleri ve iletişim teknolojileri için standartlar geliştirmeyi amaçlayan, kar amacı gütmeyen bir kuruluştur.
- Kullandığımız QWERTY klavye düzeni de ECMA'nın oluşturduğu önemli standartlardan bir tanesi
- Disket ve C yazılım diline kadar bir çok yazılım ve donanım için standartları belirlemiş ve bunların uluslararası kabul gören standartlarını da oluşturmuşlardır.
- JavaScript için 1997'de geliştirilen ilk standartta belirtilen ECMAScript ise JavaScript standartlarını belirledikleri dilin adı aslında.
- En son 12. sürümü Haziran 2021'de yayınlandı.

3.14. GELİŞMEKTE OLAN ECMASCRİPT İLE JAVASCRİPT PROGRAMLAMA DİLİNİN GÜNCEL STANDARTLARA UYUMLU ŞEKİLDE ÇALIŞABİLECEĞİNİ KAVRAR.

B) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ KODUN GÜNCEL STANDARTLARA UYUMLU HÂLE GETİRMENİN ÖNEMİ VURGULANIR

EcmaScript Neden Önemli?

•Syntax iyileştirmeleri, yeni fonksiyonlar ve düzeltmeler gibi, JavaScript'i daha eğlenceli ve kullanışlı hale getirir.

•**Daha Az Hata:** Güncellemeler, programların daha düzgün çalışmasını sağlar ve eski hataları giderir.

•**Herkes İçin Daha İyi:** Yeni özellikler, daha hızlı çalışan web siteleri ya da daha güzel animasyonlar ortaya çıkarabilir.

SON GÜNCELLEMELERİ VE GELİŞMELERİ TAKİP EDECEĞİNİZ RESMİ SİTE

https://ecma-international.org/technical-committees/tc39/?tab=general

3.14. GELİŞMEKTE OLAN ECMASCRİPT İLE JAVASCRİPT PROGRAMLAMA DİLİNİN GÜNCEL STANDARTLARA UYUMLU ŞEKİLDE ÇALIŞABİLECEĞİNİ KAVRAR.

B) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ KODUN GÜNCEL STANDARTLARA UYUMLU HÂLE GETİRMENİN ÖNEMİ VURGULANIR

EcmaScript Neden Önemli?

•Syntax iyileştirmeleri, yeni fonksiyonlar ve düzeltmeler gibi, JavaScript'i daha eğlenceli ve kullanışlı hale getirir.

•**Daha Az Hata:** Güncellemeler, programların daha düzgün çalışmasını sağlar ve eski hataları giderir.

•**Herkes İçin Daha İyi:** Yeni özellikler, daha hızlı çalışan web siteleri ya da daha güzel animasyonlar ortaya çıkarabilir.

SON GÜNCELLEMELERİ VE GELİŞMELERİ TAKİP EDECEĞİNİZ RESMİ SİTE

https://ecma-international.org/technical-committees/tc39/?tab=general

3.14. GELİŞMEKTE OLAN ECMASCRİPT İLE JAVASCRİPT PROGRAMLAMA DİLİNİN GÜNCEL STANDARTLARA UYUMLU ŞEKİLDE ÇALIŞABİLECEĞİNİ KAVRAR.

B) JAVASCRİPT PROGRAMLAMA DİLİNDEKİ KODUN GÜNCEL STANDARTLARA UYUMLU HÂLE GETİRMENİN ÖNEMİ VURGULANIR

EcmaScript Örneği

ES 12 ile gelen bir yenilik - 2021

replaceAll() yöntemine basit bir ekleme gerçekleşti

```
let example = 'BAHATTİN';

example = example.replaceAll('T','D');

console.log(example)
```

ÇIKTI: 'BAHADDİN'

SON GÜNCELLEMELERİ VE GELİŞMELERİ TAKİP EDECEĞİNİZ RESMİ SİTE

https://ecma-international.org/technical-committees/tc39/?tab=general

TEŞEKKÜRLER!