

# Programlama, p5.js ve ilk sketch

Önce nasıl düşüneceğimizi konuşacağız. Sonra tarayıcıda ilk tuvalimizi açacağız.

Algoritma nedir?

Neden p5.js?

Web Editor

İlk sketch

## SORU

# Bir robota ay demlemeyi nasıl anlatırsınız?

Robot hiçbir şeyi kendiliğinden bilmiyor. Ne söylerseniz onu, söylediğiniz sırayla yapıyor.

İki dakika: yanınızdaki arkadaşınızla adımları yazın.



# Adım adım, sırayla, eksiksiz

- 1 Suyu kaynat.
- 2 Demliğe çay koy.
- 3 Kaynar suyu ekle.
- 4 Demlenmesini bekle.
- 5 Servis et.

## Bir adımı atlarsanız?

Çay olmaz. Sırayı bozarsanız da olmaz.

## Bilgisayar da böyle

Ne yazdıysanız onu yapar. Eksik bıraktığınız adımı kendisi tamamlamaz.

# Programlama ve algoritma

## Programlama

Bilgisayara belirli bir görevi yaptırmak için verilen, adım adım yazılmış talimatlar bütünüdür.

## Algoritma

Bir problemi çözmek ya da bir hedefe ulaşmak için izlenen **sonlu**, **açık** ve **sıralı** adımlar dizisidir.

### Sonlu

Bir yerde biter.

### Açık

Yoruma yer bırakmaz.

### Sıralı

Sıra değişirse sonuç değişir.

Çay tarifi bir algoritmadır. Onu bir dilde yazdığınızda program olur.

# “Algoritma” sözcüğü nereden geliyor?



Temsilî portre. Yaşadığı dönemden kalma bir resmi yok.

## el-Harezmi

- 9. yüzyılda yaşamış matematikçi (Al-Khwarizmi).
- “Algoritma” sözcüğü onun adından türedi.
- Zamanla anlamı genişledi: **belirli kurallara göre çözüm yöntemi.**

SORU

**“Basit bir oyun yap.”  
Bu bir talimat mı?**

Bilgisayar bu cümleyle ne yapabilir? Hangi bilgi eksik?



# Büyük işi küçük adımlara böl

## ~~Basit bir oyun yap~~

- 1 Ekranda bir daire çiz.
- 2 Daireyi sağa hareket ettir.
- 3 Daire kenara değince yönünü tersine çevir.

### Algoritmik düşünce

Büyük bir problemi daha küçük ve yönetilebilir alt problemlere ayırma becerisidir.

Oyun programlamanın temeli bu iki iş:  
**parçalamak** ve **sıralamak**.

# Algoritmayı bilgisayara hangi dille anlatacağız?

## JavaScript

Programlama dilimiz. Web tarayıcısında çalışır, ek bir şey kurmak gerekmez. Günümüzün en yaygın dillerinden biri.

## p5.js

JavaScript'in görsel kütüphanesi. Şekil çizmek, renk vermek, fareyi okumak ve animasyon yapmak için hazır komutlar sunar.

## Programlama dili

Algoritmaları bilgisayara aktarmak için kullanılan biçimsel sistemdir.

# Yaratıcı kodlama

## Yaratıcı kodlama (creative coding)

Yazılımın teknik disiplinini görsel sanatların ifade biçimleriyle birleştiren, estetik ve etkileşimli dijital eserler üretmeyi hedefleyen yaklaşımdır.

### Klasik yol

Değişken, döngü, koşul... İlk haftalarda sonuç yalnızca konsolda bir satır yazı. Motivasyon düşebilir.

### p5.js ile

Aynı kavramlar renkli şekillere, hareket eden nesnelere ve oyun mekaniklerine dönüşür.

# p5.js nereden geldi?

**2001**

Processing MIT'de Ben Fry ve Casey Reas geliştirdi. Amaç: programlamayı sanatçılar ve tasarımcılar için erişilebilir kılmak.

**2014**

p5.js Lauren McCarthy önderliğinde doğdu. Processing'in felsefesini JavaScript ve HTML5 ile tarayıcıya taşıdı.

# Neden p5.js ile öğreniyoruz?



Kurulum yokWeb Editor tarayıcıda açılır; hemen yazıp çalıştırırsınız.



Anında görüntü `draw()` sürekli yenilenir. Kodu değiştirince sonucu hemen görürsünüz.



SketchProgram bir “taslak”tır. Programlama, üzerinde oynanan bir eskiz gibi düşünülür.



ToplulukThe Coding Train, Patt Vira, OpenProcessing, GitHub. Başkasının kodunu okuyarak öğrenmek yaygın.

# p5.js bir oyun motoru değildir

## Unity ya da Godot gibi değil

Hazır fizik, sahne yönetimi ve varlık (asset) sistemi yok. Bu derste oyun mekaniklerini **sıfırdan** kodlayacağız. Böylece arkadaki mantığı adım adım göreceğiz.

## Ek kütüphaneler var, ama...

- `p5.play` : sprite, animasyon, basit çarpışma.
- `Matter.js` : yerçekimi ve sert cisim çarpışması.

Prototipi hızlandırırlar ama “sihirli kutu” gibi davranırlar. Burada kullanmıyoruz.

# Üretken sanat ve Genuary

## Üretken sanat (generative art)

Görüntü, kural ve rastgelelikle kodla üretilir. Her çalıştırmada sonuç biraz farklı olabilir.

## Genuary

Her ocak ayında düzenlenir. Katılımcılar günlük istemlere (prompt) göre kod sanatı üretir; örneklerin çoğu p5.js ile yapılır.

**Fikir toplamak için:** OpenProcessing, p5.js Web Editor galerisi, The Coding Train, Patt Vira.

Bu ders oyun mekaniğine odaklanıyor. Üretken sanat zorunlu değil; merak edenler için açık bir kapı.

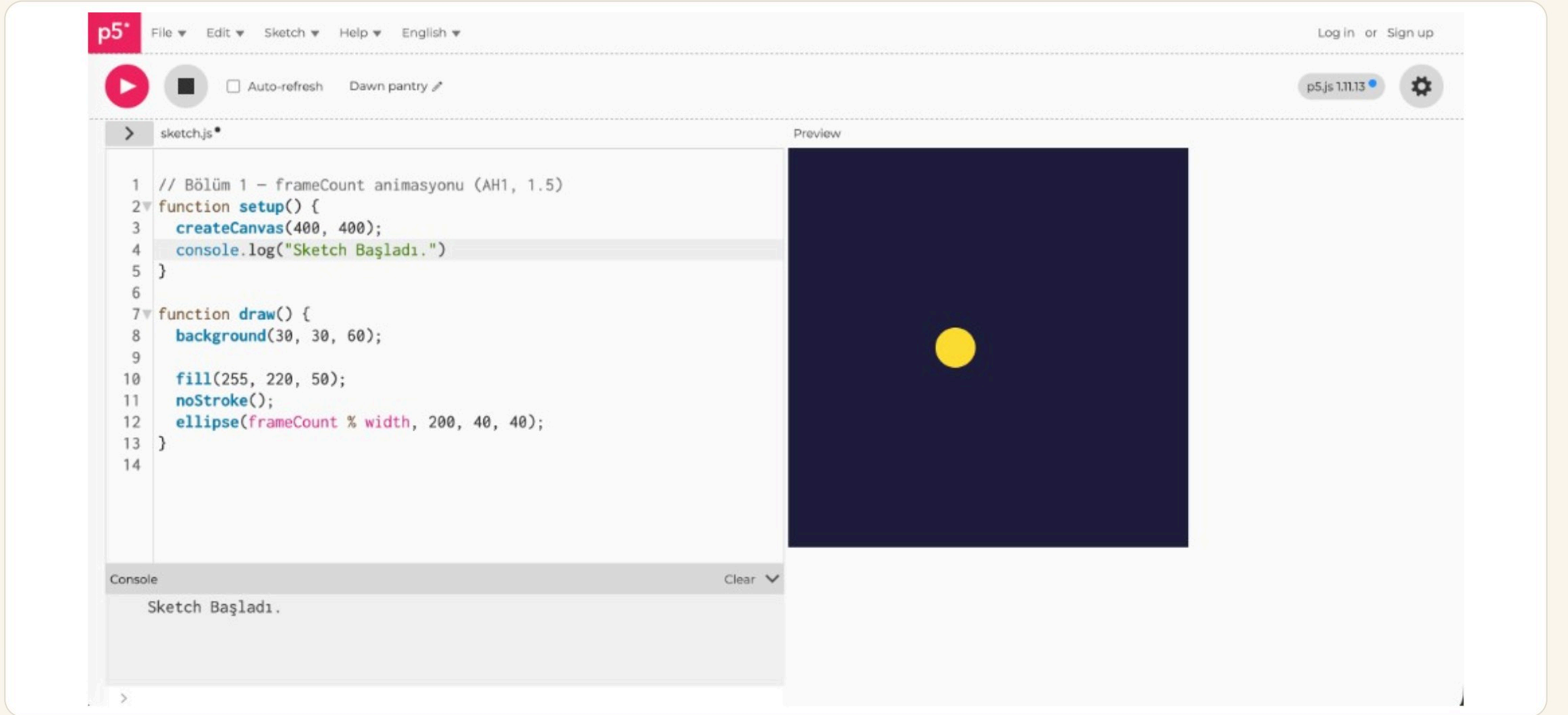
# p5.js Web Editor

- 1 `editor.p5js.org` adresine girin. Hazır bir şablon açılır.
- 2 ► **Play** ile çalıştırın, ■ ile durdurun.
- 3 Hesap açın. Projeleriniz bulutta saklanır.
- 4 Kaydet: **File** ► **Save** ya da `Ctrl + S` (Mac: `Cmd + S`).

## İlk bölümler için yeterli

Ayrı bir yazılım indirmiyoruz. Örneklerin hepsi Web Editor'da çalışacak şekilde anlatılıyor.

# Editörü tanıyalım



Solda dosyalar ve kod · üstte Play / Stop · sağda önizleme (tuval) · altta konsol

# Editörün dört bölümü

1

Dosya paneli Solda.  
Projedeki dosyalar:

```
sketch.js ,  
index.html ,  
style.css .
```

2

Kod paneli JavaScript  
/ p5.js kodunu buraya  
yazarız. Satır  
numaraları burada.

3

Önizleme Sağda. Kod  
çalışınca oluşan  
görsel çıktı, yani tuval.

4

Konsol Altta. Hata  
mesajları ve  
`console.log()`  
çıktıları burada  
görünür.

# Aynı kod kendi bilgisayarınızda da çalışır

- 1 VS Code (ya da sevdiğiniz bir editör) kurun.
- 2 Boş bir klasöre iki dosya koyun: `index.html` ve `sketch.js`.
- 3 `index.html` `p5.js`'i ve `sketch.js` 'i yükler. `setup` / `draw` `sketch.js` 'e yazılır.
- 4 Klasörü yerel sunucuyla açın: VS Code'da **Live Server** eklentisi.

## Çift tıklamayın

`index.html` 'e çift tıklamak bazı tarayıcılarda kısıtlara takılır. Yerel sunucu kullanın.

## Ne zaman yerel?

Proje birkaç dosyaya yayılınca ya da değişiklikleri Git ile saklamak isteyince.

# index.html iskeleti

index.html

Kopyala

Büyüt

```
1 <!DOCTYPE html>
2 <html lang="tr">
3 <head>
4   <meta charset="utf-8"/>
5   <script src="https://cdn.jsdelivr.net/npm/p5@1.11.3/lib/p5.min.js"></script>
6 </head>
7 <body>
8   <script src="sketch.js"></script>
9 </body>
10 </html>
```

## CDN (içerik dağıtım ağı)

Kütüphane dosyasını indirmeden, internetteki bir adresten tarayıcıya yükleme yoludur. 5. satır bunu yapar.

- Bilgisayarınızda p5.js kopyası olmasa da sketch çalışır.
- İnternet bağlantısı gerekir.
- Çevrimdışı çalışmak için `p5js.org/download` adresinden indirip yerel yoldan bağlayın.

**SORU**

# Play'e basınca ekranda ne göreceğiz?

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   /* Tuval boyutunu bir kez ayarlar.  
3     Bu blok, birden fazla satırı kapsayan yorum örneğidir. */  
4   createCanvas(400, 400); // 400x400 piksel tuval oluştur  
5 }  
6  
7 function draw() {  
8   background(220); // Açık gri arka plan  
9 }
```

Çalıştırmadan önce tahmin edin. Kaç satır gerçekten bir iş yapıyor?

# İlk sketch: gri bir kare

sketch.js

Kopyala

Düzenle

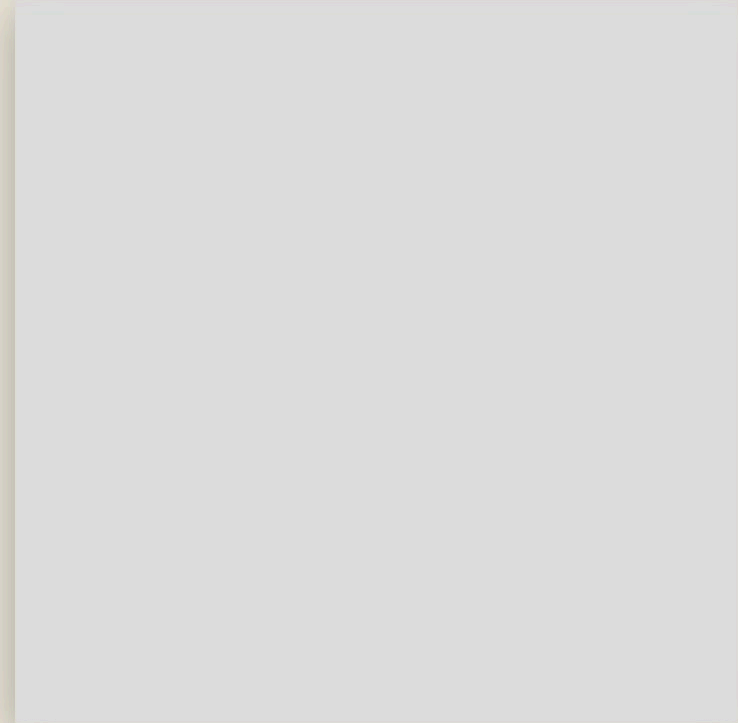
Büyüt

```
1 function setup() {  
2   /* Tuval boyutunu bir kez ayarlar.  
3     Bu blok, birden fazla satırı kapsayan yorum örneğidir. */  
4   createCanvas(400, 400); // 400x400 piksel tuval oluştur  
5 }  
6  
7 function draw() {  
8   background(220); // Açık gri arka plan  
9 }
```

● Çıktı

Çalıştır

Büyüt



`createCanvas(400, 400)` 400 piksel genişliğinde, 400 piksel yüksekliğinde bir çizim alanı (canvas) açar.

`background(220)` arka planı boyar. Sayı 0 (siyah) ile 255 (beyaz) arasındadır.

# Yorum satırları

## Yorum (comment)

Kodun yanına ya da üstüne yazılan, programın çalışmasını **etkilemeyen** açıklama metnidir.

- `//` satırın sonuna kadar sürer.
- `/* ... */` birden fazla satırı kapsar.
- Okuyana ve gelecekteki size ne yapıldığını hatırlatır.
- Bu derste çoğunlukla `//` kullanacağız.

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   /* Tuval boyutunu bir kez ayarlar.  
3     Bu blok, birden fazla satırı kapsayan yorum örneğidir. */  
4   createCanvas(400, 400); // 400x400 piksel tuval oluştur  
5 }  
6  
7 function draw() {  
8   background(220); // Açık gri arka plan  
9 }
```

# Her sketch'te iki fonksiyon var

## setup()

Program başlarken **bir kez** çalışır. Tuvali burada kurarız.

## draw()

Ardından **tekrar tekrar** çalışır. Çizimi buraya yazarız.

Her p5.js projesine **sketch** (taslak) denir. Bir sketch en az bu iki fonksiyonu içerir.

İkisinin ayrıntısını sonraki derste, animasyonla birlikte göreceğiz.

# console.log() : konsola yaz

sketch.js

Kopyala

Düzenle

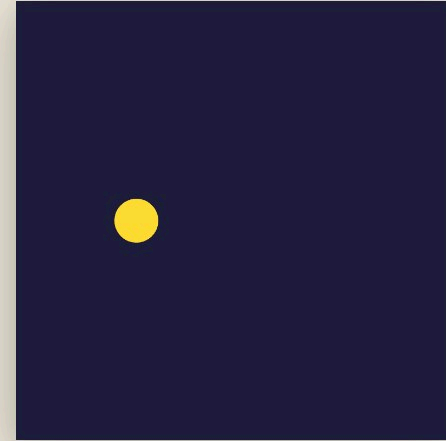
Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3   console.log("Sketch başladı.");  
4 }  
5  
6 function draw() {  
7   background(30, 30, 60);  
8   fill(255, 220, 50);  
9   noStroke();  
10  ellipse(frameCount % width, 200, 40, 40);  
11 }
```

Çıktı

Çalıştır

Büyüt



Sketch başladı.

Tuvalde **görünmez**; alttaki konsola yazar.

Hata ayıklamak ve bir değeri gözlemek için kullanılır.

Birden fazla değer alır:

```
console.log("sayac:", sayac)
```

# İlk günün üç tuzağı

## Parantezler

( ) ve { } her zaman eşleşmeli. Eksik parantez en sık yapılan sözdizimi hatası.

## Noktalı virgül

JavaScript'te çoğu yerde zorunlu değil. Yine de belirsizliği azaltır; örneklerde hep koyuyoruz.

## Büyük / küçük harf

`createCanvas` ile `createcanvas` aynı şey değil. Komutları olduğu gibi yazın.

## SORU

# Bu kodda iki hata var. Bulun.

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3  
4 function draw() {  
5   background(220;  
6 }
```

# İkisi de eksik parantez

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3 }  
4  
5 function draw() {  
6   background(220);  
7 }
```

## 3. satır

setup 'ın gövdesi kapanmamıştı: `}` eksikti.

## 6. satır

background(220) açılmış ama kapanmamıştı: `)` eksikti.

Konsol hatanın satırını söyler. Önce konsola bakmayı alışkanlık edinin.

# Zıplayan topu adımlara bölün

 15-20 dk

1. “Ekranada zıplayan bir top” hedefini kendi cümlelerinizle **5–7 adıma** bölün. Kod yok, yalnızca sıralı liste.
2. Aynı hedefi bir yapay zekâya adım adım yazdırın. Onun listesini kendinizinkiyle karşılaştırın.
3. Eksik ya da sırası yanlış adımları düzeltin.

## Kural

Yapay zekâdan henüz kod istemeyin. Yalnızca sıralı talimat listesi isteyin.

# Editörde ilk denemeler

 10 dk

1. Tuvali  $600 \times 300$  yapın.
2. Arka planı sırayla siyah, beyaz ve koyu gri yapın. Hangi sayıları kullandınız?
3. `setup` içinde adınızı konsola yazdırın.
4. Kendi cümleinizle bir `//` yorumu ve bir `/* */` yorumu ekleyin. Çıktı değişti mi?
5. Bir parantezi bilerek silin. Konsol ne diyor? Hatayı düzeltin.

# Hangisi algoritma sayılmaz?

- A** 1'den 10'a kadar say; her sayıyı ekrana yaz.
- B** Suyu kaynat, çayı koy, beş dakika bekle, servis et.
- C** Bir sayı tut. Sıfırdan büyükse 1 çıkar ve yinele; sıfırsa dur.
- ☒ Güzel görünen bir renk seç; daha güzelini bulana kadar denemeye devam et.

"Güzel" açık değil, "daha güzelini bulana kadar" sonlu değil. Algoritma sonlu, açık ve sıralı olmalı.

# Bir saniye sonra konsolda ne olur?

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   createCanvas(200, 200);  
3   console.log("A");  
4 }  
5  
6 function draw() {  
7   background(0);  
8   console.log("B");  
9 }
```

**A** Bir A, bir B



Bir A, ardından yaklaşık 60 tane B

**C** 60 A ve 60 B

**D** Hiçbir şey; tuval siyah olur

setup bir kez çalışır. draw saniyede yaklaşık 60 kez yinelenir.

## index.html'den CDN satırını silerseniz?

- ☐ A Tuval beyaz açılır.
- ☒ B Tuval hiç oluşmaz; p5.js yüklenmediği için `setup` ve `draw` 'ı çağıran olmaz.
- ☐ C Sketch daha hızlı çalışır.
- ☐ D Hiçbir şey değişmez; p5.js tarayıcıda zaten vardır.

p5.js tarayıcının parçası değildir. `createCanvas`, `background` ve `draw` döngüsü o dosyadan gelir.

# Bugün ne öğrendik?

- ✓ **Algoritma:** sonlu, açık ve sıralı adımlar. Program, bu adımların bir dilde yazılmış hâli.
- ✓ **Algoritmik düşünce:** büyük işi küçük adımlara bölmek ve sıralamak.
- ✓ **p5.js:** kurulum yok, sonuç hemen ekranda. Ama oyun motoru değil; mekanikleri biz yazacağız.
- ✓ **Web Editor:** dosyalar, kod, önizleme, konsol.
- ✓ **İlk sketch:** `setup` + `draw`, `createCanvas`, `background`.
- ✓ **Yorum** `//` ve `/* */` çalışmayı etkilemez; `console.log` konsola yazar.

SIRADAKİ DERS

# Şekil tuvalde nereye çizilir?

Koordinat sistemi, temel şekiller ve renkler.

Sonraki sunum →

Tüm sunumlar