

setup, draw ve ilk animasyon

Şekil çizmeyi biliyoruz. Bugün onları hareket ettireceğiz.

setup bir kez

draw her frame

width ve height

frameCount ve %

SORU

Bir çizgi filmde karakter gerçekten hareket ediyor mu?

Ekranda tek tek duran resimler var. Peki hareketi nasıl görüyoruz?



Hareket, art arda gösterilen resimlerdir

- Her resim bir öncekinden biraz farklıdır.
- Resimler yeterince hızlı değişince göz bunu hareket olarak görür.
- p5.js de aynısını yapar: tuvali saniyede yaklaşık **60 kez** yeniden çizer.

Frame

`draw()` fonksiyonunun bir kez çalışması; ekranın bir kez güncellenmesi.

FPS (frames per second)

Saniyedeki frame sayısı. 60 FPS ise saniyede 60 frame üretilir.

Sketch'in omurgası: iki fonksiyon

setup()

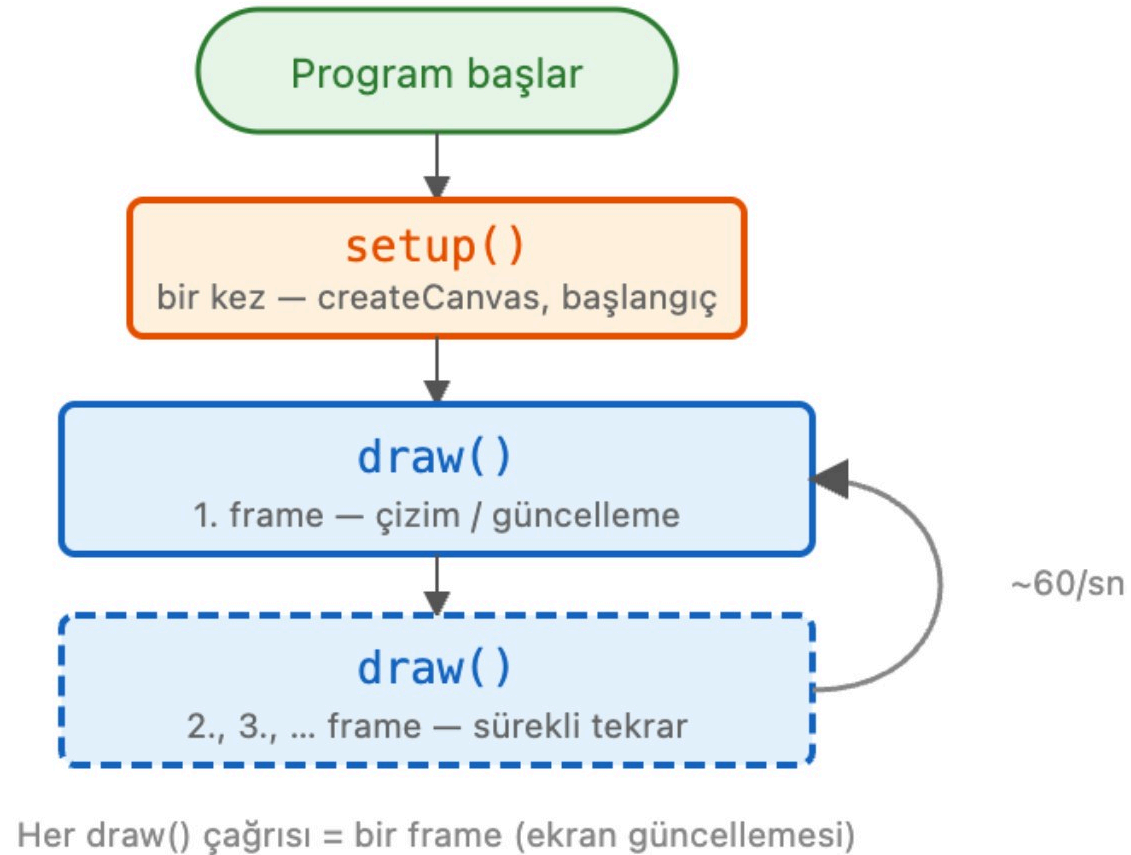
Program başladığında **bir kez** çalışır. Tuvali oluşturmak, başlangıç değerlerini vermek gibi hazırlık işleri burada yapılır.

draw()

Hazırlık bittikten sonra **sürekli tekrarlanır**. Varsayılan hız saniyede yaklaşık 60 kездir. `frameRate()` ile bu değeri değiştirilebilir.

Çalışma sırası

setup() ve draw() çalışma sırası



Hazırlık tek seferlik; `draw` sketch durana kadar tekrar eder.

Sıra hiç değişmez

- 1 Program başlar.
- 2 `setup()` bir kez çalışır: `createCanvas()` ve başlangıç ayarları.
- 3 `draw()` ilk kez çalışır. Bu 1. frame.
- 4 `draw()` yeniden çalışır. Bu 2. frame.
- 5 3. frame, 4. frame... Sketch durdurulana kadar.

	Ne sıklıkla?	Tipik işler
<code>setup()</code>	Başta bir kez	<code>createCanvas()</code> , başlangıç değerleri, <code>frameRate()</code>
<code>draw()</code>	Her frame	<code>background()</code> , çizim, konum güncelleme

Bu, oyun döngüsünün ilk hâli

`setup()` oyun dünyasını kurar

Tuval, başlangıç konumları, skorun sıfırlanması.

`draw()` her frame'de üç iş yapar

1. Girdiyi okur.
2. Durumu günceller.
3. Ekranı yeniden çizer.

`draw()`, oyun döngüsünün (game loop) kalbidir. Bölüm 2'nin sonunda bu döngüye ayrıntılı döneceğiz.

Tuvalin boyutu: `width` ve `height`

Değişken	Anlamı	<code>createCanvas(500, 400)</code> sonrası
<code>width</code>	Tuval genişliği (piksel)	<code>500</code>
<code>height</code>	Tuval yüksekliği (piksel)	<code>400</code>

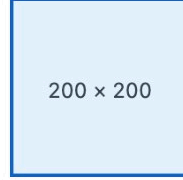
- Bu iki değişkeni p5.js kendisi sağlar; biz tanımlamayız.
- Kenara hizalarken ve ortalarken kullanılır.
- Yatay orta nokta: `width / 2`

Farklı createCanvas , farklı tuval

createCanvas — dört farklı tuval boyutu

Ölçek: 1 birim çizim $\approx$ 0,5 ekran pikseli (görelî karşılaştırma)

`createCanvas(200, 200)`



Kare
width=200 · height=200

`createCanvas(400, 200)`



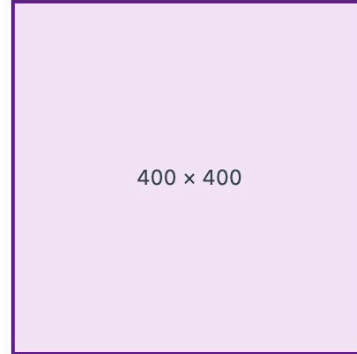
Yatay dikdörtgen
width=400 · height=200

`createCanvas(200, 400)`



Dikey dikdörtgen
width=200 · height=400

`createCanvas(400, 400)`



Büyük kare
width=400 · height=400

Not

`createCanvas`(genişlik,
yükseklik) sonrası
`width` ve `height`
bu sayılara eşitlenir.

Önizleme panelinde tuval
bu oranlarda görünür.

Kitap örneklerinde sıkça
`400 × 400`
kullanılır.

Sol taraftaki dört kutu
görelî boyutları gösterir;
piksel-ekran ölçeği
küçültülmüştür.

`createCanvas(200, 200)`, `(400, 200)`, `(200, 400)` ve `(400, 400)` : önce genişlik, sonra yükseklik.

Sayı yazmak yerine `width / 2`

sketch.js

Kopyala

Düzenle

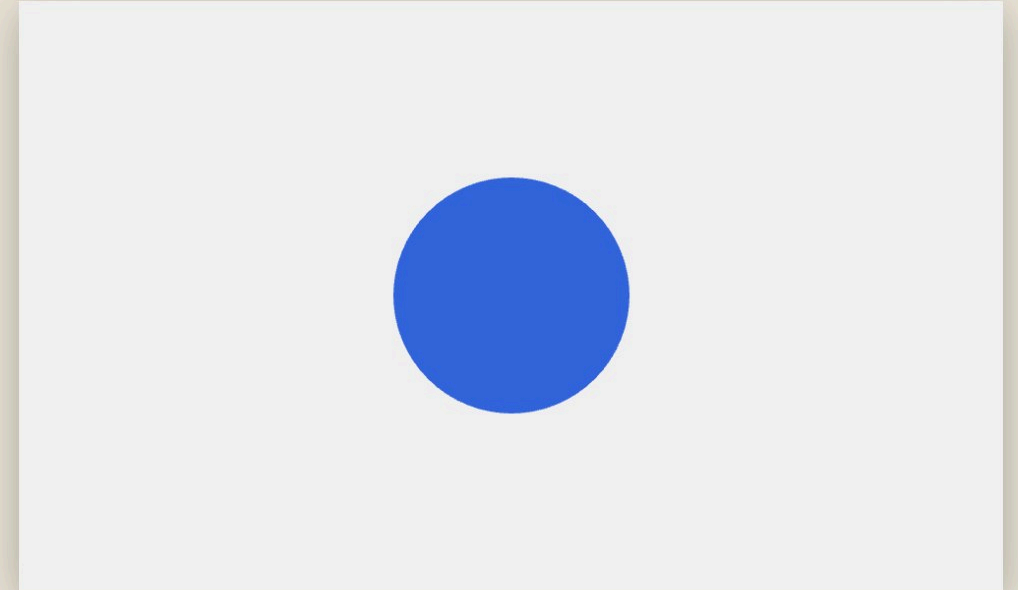
Büyüt

```
1 function setup() {  
2   createCanvas(500, 300);  
3 }  
4  
5 function draw() {  
6   background(240);  
7  
8   fill(50, 100, 220);  
9   noStroke();  
10  ellipse(width / 2, height / 2, 120, 120);  
11 }
```

● Çıktı

Çalıştır

Büyüt



“Düzenle” ile tuvali `createCanvas(300, 300)` yapın. Daire hâlâ ortada mı? `ellipse(250, 150, ...)` yazsaydık ne olurdu?

frameCount : kaçınıcı frame'deyiz?

sketch.js

Kopyala

Düzenle

Büyüt

```
1 function setup() {  
2   createCanvas(400, 200);  
3 }  
4  
5 function draw() {  
6   background(240);  
7   fill(0);  
8   textSize(64);  
9   text(frameCount, 40, 125);  
10 }
```

Çıktı

Çalıştır

Büyüt

109

frameCount

p5.js'in yerleşik değişkenidir. `draw()` her çalıştığında 1 artan bir sayaçtır: ilk çağrıda 1, ikincide 2 olur.

SORU

Sürekli artan bir sayımız var. Onu bir dairenin x konumuna yazarsak?

sketch.js

Kopyala

Büyüt

```
ellipse(frameCount, 200, 40, 40);
```

Daire ne yapar? Tuvalin sağ kenarına gelince ne olur?

Daire sağa kayar... ve bir daha gelmez

sketch.js

Kopyala

Düzenle

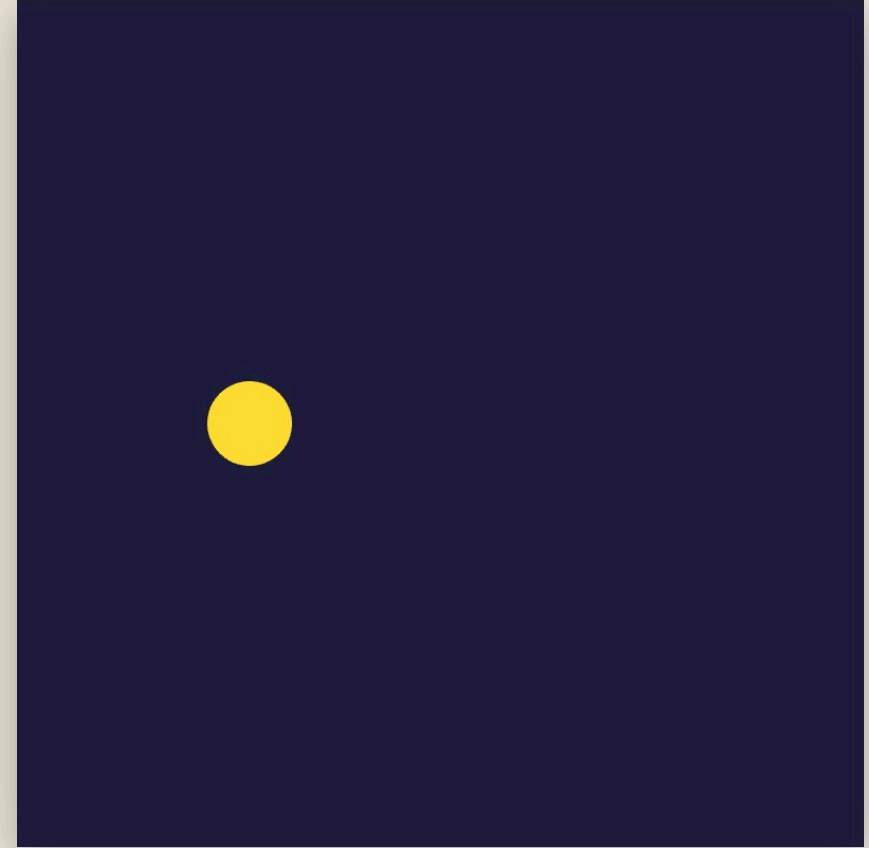
Büyüt

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

Çıktı

Çalıştır

Büyüt



Her frame x bir piksel artıyor. 400'ü geçince daire tuvalin dışında kalıyor. Onu geri getirmenin bir yolu lazım.

Modulo: bölümden kalan

Modulo operatörü %

Bölümden **kalanı** verir. Örneğin $7 \% 3$ ifadesinin sonucu 1'dir.

Sayı büyüdükçe kalan hep aynı aralıkta döner. Saatin 12'den sonra yeniden 1'e dönmesi gibi.

n	n % 4
0	0
1	1
2	2
3	3
4	0
5	1
6	2
7	3
8	0

frameCount % width : sona gelince başa dön

sketch.js

Kopyala

Düzenle

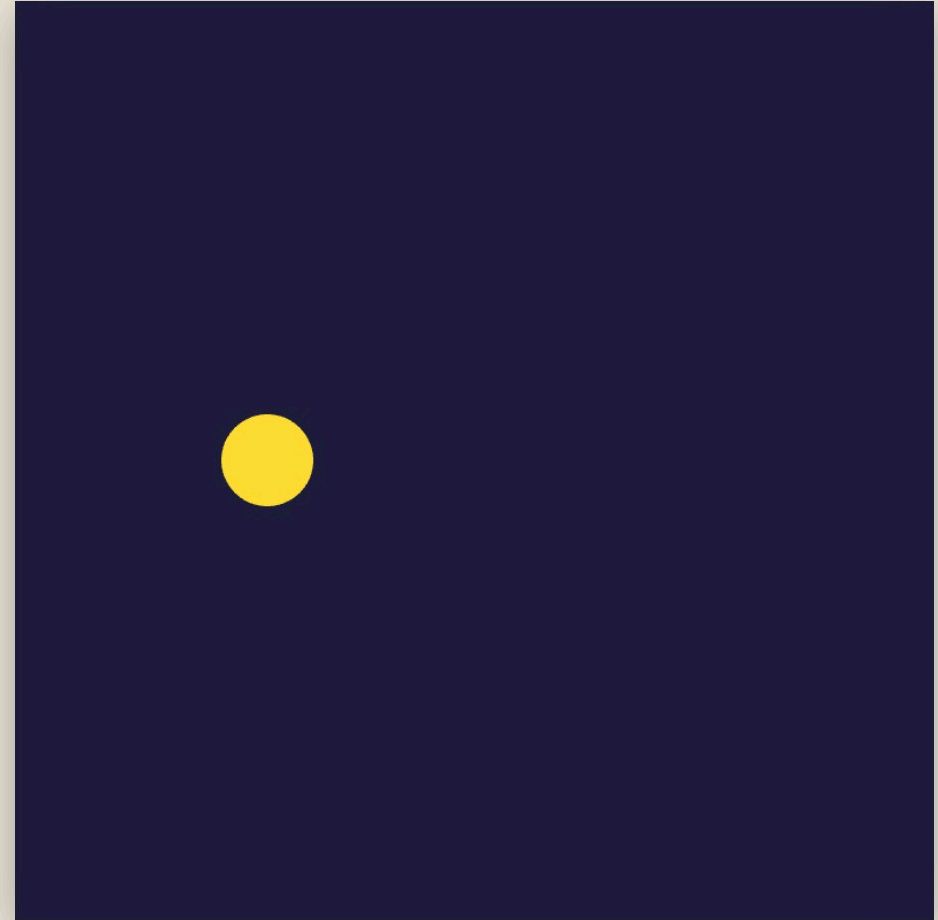
Büyüt

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

● Çıktı

Çalıştır

Büyüt



`frameCount % width` sayacı 0 ile `width - 1` arasında döndürür. Daire sağdan çıkınca soldan yeniden girer.

SORU

**background satırını draw 'dan
silersek ne olur?**

Daire yine hareket eder mi? Ekranda başka ne görürüz?



Eski frame'ler silinmez, daire iz bırakır

sketch.js

Kopyala

Düzenle

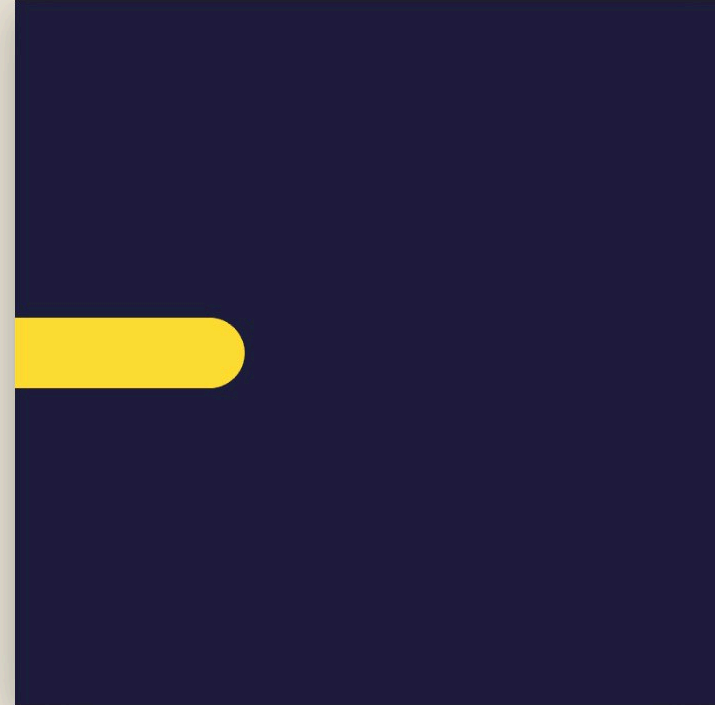
Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3   background(30, 30, 60); // yalnızca bir kez  
4 }  
5  
6 function draw() {  
7   fill(255, 220, 50);  
8   noStroke();  
9   ellipse(frameCount % width, 200, 40, 40);  
10 }
```

Çıktı

Çalıştır

Büyüt



background neden draw içinde?

Her frame'in başında sahneyi temizler. Temizlenmezse yeni çizim eskilerin üstüne biner.

draw 'ı durdurmak: noLoop() ve loop()

noLoop()

draw() tekrarını durdurur. Hareket olmayan çizimlerde setup içine yazılır; draw yalnızca bir kez çalışır.

loop()

Durmuş draw() tekrarını yeniden başlatır.

Geçen dersteki dört kombinasyon ve yedi adım örneklerinde noLoop() vardı: çizim değişmediği için tekrar etmeye gerek yoktu.

Her komutu ezberlemek gerekmez: referans sayfası

- Her dilin ve kütüphanenin resmî bir **referans sayfası** (dokümantasyon) vardır.
- Fonksiyonun adı, parametreleri, kısa açıklaması ve çalışan örneği orada bulunur.
- Asıl beceri, ihtiyacınız olan komutu nerede arayacağınızı bilmektir.
- p5.js için adres: `p5js.org/reference`

Bakılacak fonksiyon	Ne işe yarar?
<code>square()</code>	Kare
<code>triangle()</code>	Üç köşeli şekil
<code>quad()</code>	Dört köşeli şekil
<code>arc()</code>	Daire yayı
<code>noLoop()</code> , <code>loop()</code>	<code>draw</code> 'ı durdur, başlat

Oyun örneklerimiz çoğunlukla `rect`, `ellipse`, `line` ve `point` ile ilerleyecek. Diğerlerini referanstan tanımanız yeterli.

Referans sayfasını okuyun

🕒 25-35 dk

1. `noLoop`, `loop`, `triangle`, `quad`, `square`, `ellipse` ve `arc` sayfalarını açın. Her biri için bir cümlelik not alın.
2. Karşılaştırın: `rect` ile `square` farkı ne? `triangle` ve `quad` kaç köşe ister? `ellipse` ile `arc` arasındaki fark ne?
3. Hâlâ anlamadığınız bir nokta varsa yapay zekâya sorun. Önce referansı okuduğunuzu söyleyin.
4. Yapay zekânın verdiği örnekleri editörde deneyin. Çelişki görürseniz resmî sayfayı esas alın.

İlk sketch'i çalıştırırken

1. `rect()` ile `ellipse()` arasındaki konumlandırma farkı neden hata kaynağı olur?
2. `draw()` 'ın sürekli çalışması oyun geliştirmede ne anlama gelir?
3. `background()` çoğu animasyonda neden her frame yeniden çağrılır?
4. `fill(255, 0, 0)` ile `fill(128)` arasındaki fark ne? Gri ton ne zaman yeter?
5. Alfa 0, 120 ve 255 iken aynı renk nasıl görünür? Oyun arayüzünde ne işe yarar?

Günlük bir rutin: setup mı, draw mı?

 10 dk

1. Kendi günlük bir rutininizi (örneğin çay demlemek) 4–6 adımlık bir algoritma olarak yazın.
2. Hangi adımlar `setup()` 'a benziyor: bir kez yapılıyor mu?
3. Hangi adımlar `draw()` 'a benziyor: tekrar tekrar mı yapılıyor?
4. Kararınızı birer cümleyle gerekçelendirin.

Animasyonu siz yönetin

 20 dk

Kayan daire örneğinden başlayın. Her adımda yalnızca bir satır değişecek.

1. Daire soldan sağa değil, **yukarıdan aşağı** aksın.
2. Daire iki kat hızlı gitsin.
3. Daire **sağdan sola** gitsin.
4. Daire yerinde dursun ama çapı 0'dan 200'e büyüsün, sonra başa dönsün.
5. Zorlu: daire çapraz gitsin ve iz bıraksın.

250. frame'de dairenin x konumu kaç?

sketch.js

Kopyala

Büyüt

```
createCanvas(100, 100);  
// ...  
ellipse(frameCount % width, 50, 20, 20);
```

☐ A 250☐ B 100☒ C 50☐ D 0

250'yi 100'e bölünce kalan 50. Daire tuvali iki kez geçmiş, üçüncü turun ortasında.

Bu sketch çalışınca ekranda ne görünür?

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3 }  
4  
5 function draw() {  
6   fill(255, 0, 0);  
7   ellipse(200, 200, 100, 100);  
8   background(220);  
9 }
```

- ☐ A Gri zeminde kırmızı daire
- ☒ B Yalnızca gri zemin
- ☐ C Yanıp sönen bir daire
- ☐ D Hata verir

`background` en son çalışıyor ve daireyi her frame'de örtüyor. Sıra önemli.

`createCanvas(400, 400)` satırını `draw` 'ın içine taşırsanız?

- ☐ A Hiçbir şey değişmez.
- ☐ B Tuval her frame büyür.
- ☒ C Tuval saniyede 60 kez yeniden kurulur: gereksiz iş, yavaş ve hataya açık.
- ☐ D Animasyon iki kat hızlanır.

Bir kez yapılacak hazırlık `setup` 'ın işidir. `draw` 'a yazılan her şey her frame tekrarlanır.

Bugün ne öğrendik?

- ✓ `setup()` başta bir kez; `draw()` saniyede yaklaşık 60 kez çalışır.
- ✓ Frame: `draw()` 'ın bir turu. FPS: saniyedeki frame sayısı.
- ✓ `width` ve `height` tuvalin boyutunu tutar; ortalamak için `width / 2`.
- ✓ `frameCount` her frame 1 artar; `%` ile başa döndürülür.
- ✓ `background` her frame sahneyi temizler; yoksa şekiller iz bırakır.
- ✓ Bilmediğiniz komut için ilk adres: `p5js.org/reference`.

SIRADAKİ DERS

Program bir şeyi nasıl hatırlar?

Değişkenler, veri türleri ve operatörler.

Sonraki sunum →

← Önceki