

Fare ve ilk oyunumuz: Daire Tıklama

Değişkenleri, koşulları ve fareyi birleştirip oynanabilir bir oyun yazacağız.

mouseX, mouseY

text ve hizalama

dist ile isabet

mousePressed

Adım adım oyun

SORU

Oyun, oyuncunun ne yaptığını nereden bilir?

Şimdiye kadar sketch'lerimiz kendi kendine çalıştı. Bir oyunda ise kararlar oyuncuya bağlı: fare nerede, tıkladı mı?



p5.js fareyi bizim yerimize izler

Ad	Tür	Ne verir?	Örnek
<code>mouseX</code>	number	İmlecin x konumu (0 ... width - 1)	500 genişlikte tuvalin ortasında ≈ 250
<code>mouseY</code>	number	İmlecin y konumu (0 ... height - 1)	Üst kenarda 0
<code>mouseIsPressed</code>	boolean	Fare tuşu şu an basılı mı?	Basılıyken <code>true</code>
<code>mousePressed()</code>	fonksiyon	Tuşa basıldığı an çalışır	Her basışta bir kez
<code>mouseReleased()</code>	fonksiyon	Tuş bırakıldığı an çalışır	Her bırakışta bir kez

Olay fonksiyonu

`mousePressed()` ve `mouseReleased()`, tıpkı `setup()` ve `draw()` gibi özeldir: olay gerçekleşince p5.js bu addaki fonksiyonu kendisi çağırır.

Yalnızca fareyle oynanan oyunlar

Fruit Ninja 2010

Parmağın ekranda bıraktığı iz meyveye değiyor mu? Oyun her frame parmağın konumunu okur.

osu! 2007

Müzikle beliren dairelere doğru anda tıklanır. Bugün yazacağımız oyunun ritimli akrabası.

Nişan antrenmanı

E-spor oyuncularını maçtan önce ısınır: beliren hedefe en hızlı tıklayan kazanır.

Üçünün de sorduğu iki soru

İmleç şu an nerede? Tıklama, hedefin içine mi düştü? Bu derste ikisini de yanıtlayacağız.

Daire fareyi takip etsin

sketch.js

Kopyala

Düzenle

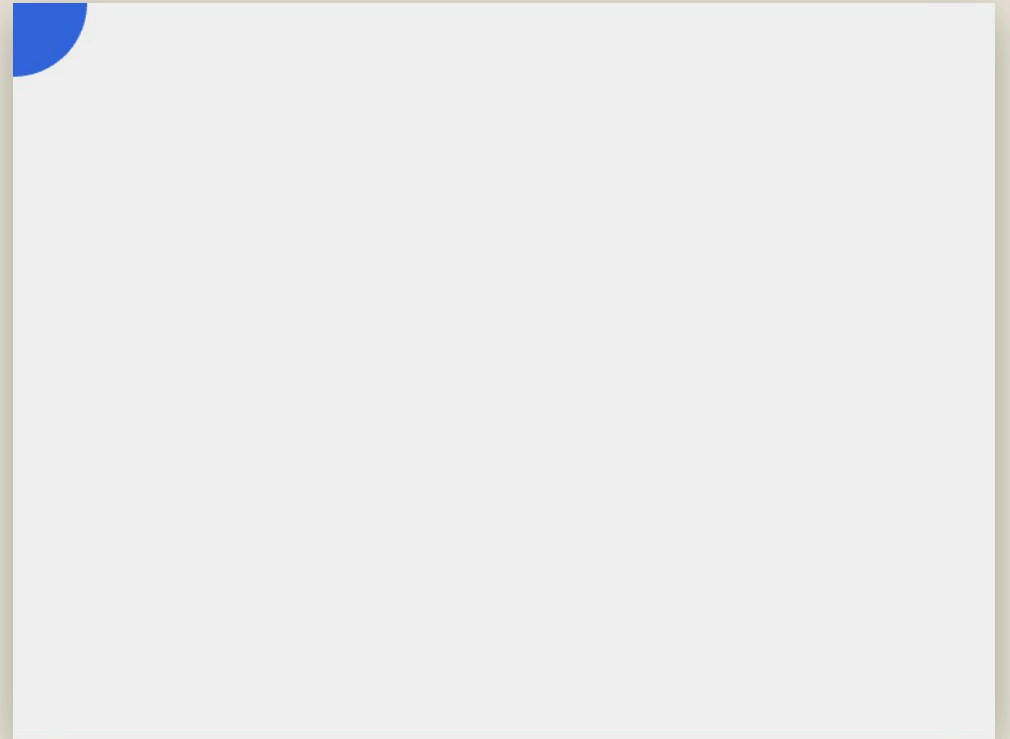
Büyüt

```
1 function setup() {  
2   createCanvas(400, 300);  
3 }  
4  
5 function draw() {  
6   background(240);  
7  
8   if (mouseIsPressed) {  
9     fill(230, 70, 40); // basılıyken kırmızı  
10  } else {  
11    fill(50, 100, 220); // değilken mavi  
12  }  
13  
14  noStroke();  
15  ellipse(mouseX, mouseY, 60, 60);  
16 }
```

● Çıktı

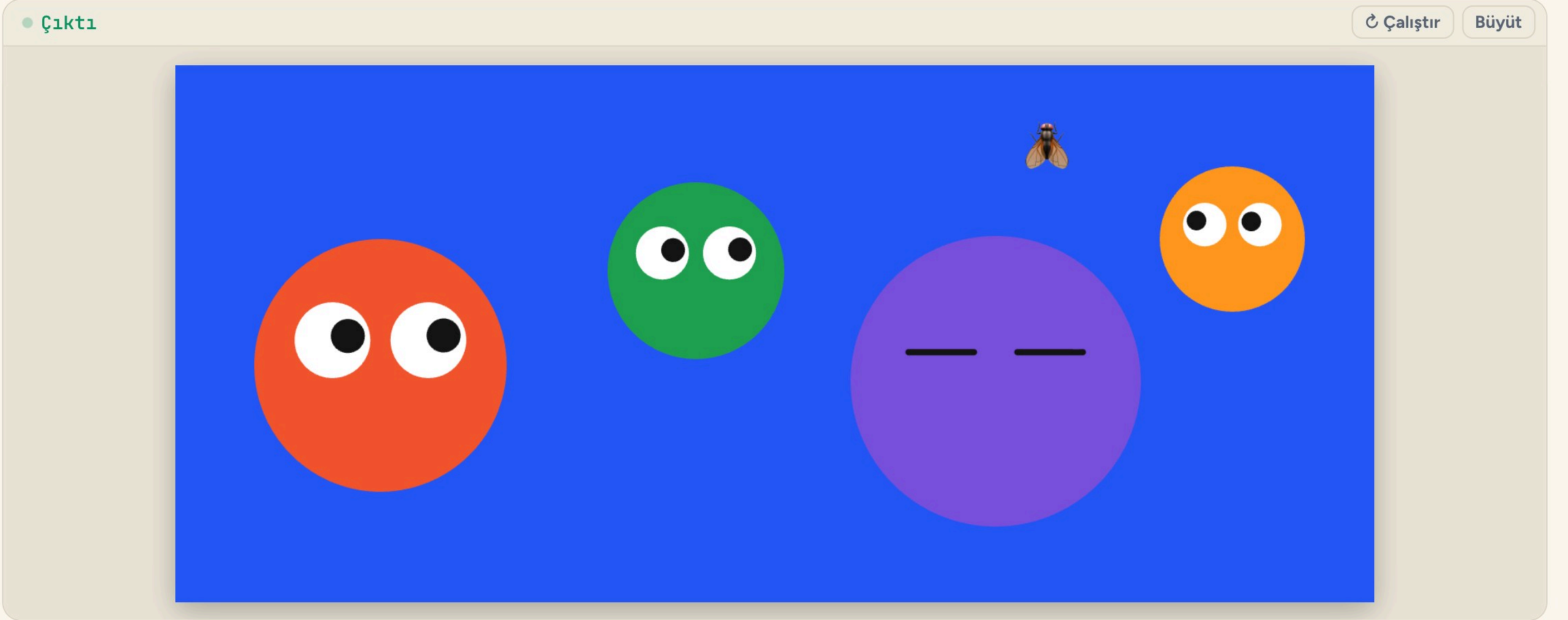
Çalıştır

Büyüt



Fareyi tuvalde gezdirin, sonra tuşu basılı tutun. `mouseX` ve `mouseY` her frame güncellenir.

mouseX ve mouseY ile: gözler sineği izler



Fareyi tuvalde gezdirin. Dört yaratık da aynı iki sayıyı okuyor: `mouseX` ve `mouseY`. Göz bebeğinin hesabını şimdi anlamanız gerekmiyor.

Ekrana yazı yazmak

Komut	Ne yapar?
<code>text(metin, x, y)</code>	(x, y) noktasını referans alarak metni yazar.
<code>textSize(boyut)</code>	Yazı boyutunu piksel olarak ayarlar.
<code>textAlign(hizaX, hizaY)</code>	(x, y) noktasının, metnin hangi köşesine ya da kenarına denk geleceğini seçer.

Yatay

`LEFT` sol kenar, `CENTER` orta, `RIGHT` sağ kenar.

Dikey

`TOP` üst, `CENTER` orta, `BOTTOM` alt.

`textAlign` koordinatı değiştirmez; o noktanın metne **nereden** bağlanacağını değiştirir.

Aynı araçlar, iki farklı hizalama

sketch.js

Kopyala

Düzenle

Büyüt

```
1 function setup() {  
2   createCanvas(400, 300);  
3 }  
4  
5 function draw() {  
6   background(240);  
7   fill(0);  
8  
9   textSize(24);  
10  textAlign(LEFT, TOP); // (15, 15) = metnin sol-üst köşesi  
11  text("Skor: 0", 15, 15);  
12  
13  textSize(32);  
14  textAlign(CENTER, CENTER); // nokta = metnin ortası  
15  text("Kazandınız!", width / 2, height / 2);  
16 }
```

Çıktı

Çalıştır

Büyüt

Skor: 0

Kazandınız!

Köşe yazıları için `textAlign(LEFT, TOP)`; ekran ortası mesajı için `textAlign(CENTER, CENTER)` ve `width / 2, height / 2`.

SORU

Fare dairenin içinde mi? Bunu program nasıl anlar?

Elimizde dört sayı var: farenin x ve y 'si, dairenin merkezi ve yarıçapı.
Hangi soruyu sormalıyız?

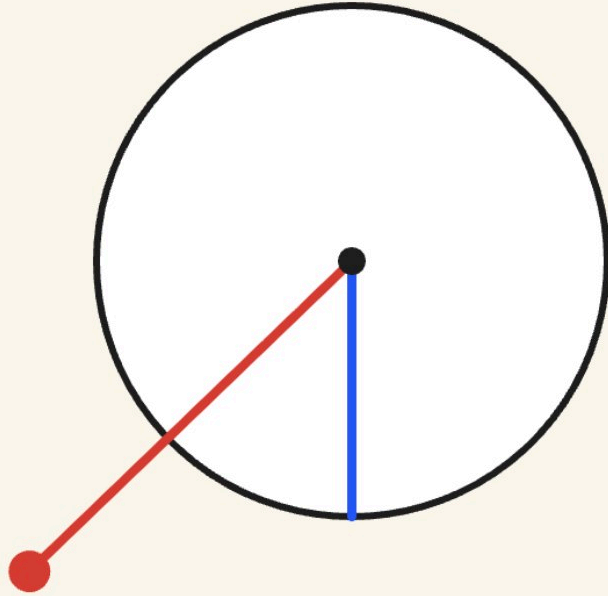


Mesafe yarıçaptan küçükse içeridedir

• Çıktı

Çalıştır

Büyüt

`mesafe = 193``yarıçap = 110``mesafe < yarıçap``false``fare dairenin dışında`

Fareyi dairenin içine ve dışına götürün. Renkli çizgi merkezle fare arasındaki mesafe, mavi çizgi yarıçaptır; soru yalnızca hangisinin daha uzun olduğudur.

`dist()` : iki nokta arasındaki mesafe

`dist(x1, y1, x2, y2)`

p5.js'in yerleşik fonksiyonu. İki nokta arasındaki düz mesafeyi (Öklid mesafesi) sayı olarak verir.

- Matematikteki karşılığı: $\sqrt{(x2-x1)*(x2-x1) + (y2-y1)*(y2-y1)}$. `sqrt` de yerleşiktir; isterseniz kendiniz yazabilirsiniz.
- İleride iki dairenin birbirine değip değmediğini de böyle anlayacağız: mesafe ile yarıçapların toplamını karşılaştırarak.

Fare daireye girince renk deęişsin

sketch.js

Kopyala

Düzenle

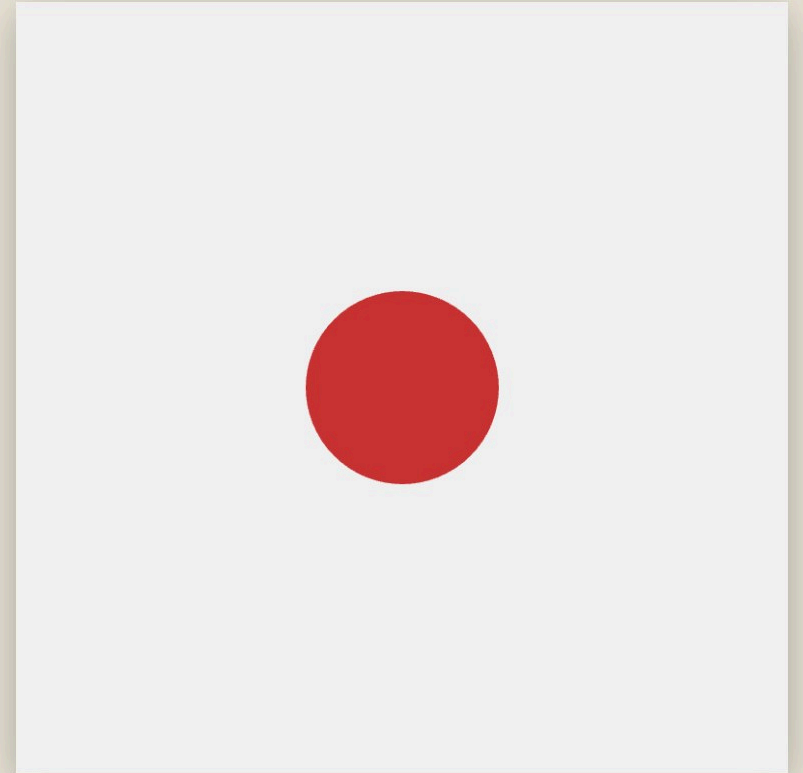
Büyüt

```
1 function setup() {  
2   createCanvas(400, 400);  
3 }  
4  
5 function draw() {  
6   background(240);  
7  
8   let merkezX = 200;  
9   let merkezY = 200;  
10  let yaricap = 50;  
11  
12  let mesafe = dist(mouseX, mouseY, merkezX, merkezY);  
13  
14  if (mesafe < yaricap) {  
15    fill(50, 200, 80); // Yeşil – fare dairenin içinde (tam kenar hariç)  
16  } else {  
17    fill(200, 50, 50); // Kırmızı – fare uzakta  
18  }  
19  
20  noStroke();  
21  ellipse(merkezX, merkezY, yaricap * 2, yaricap * 2);  
22 }
```

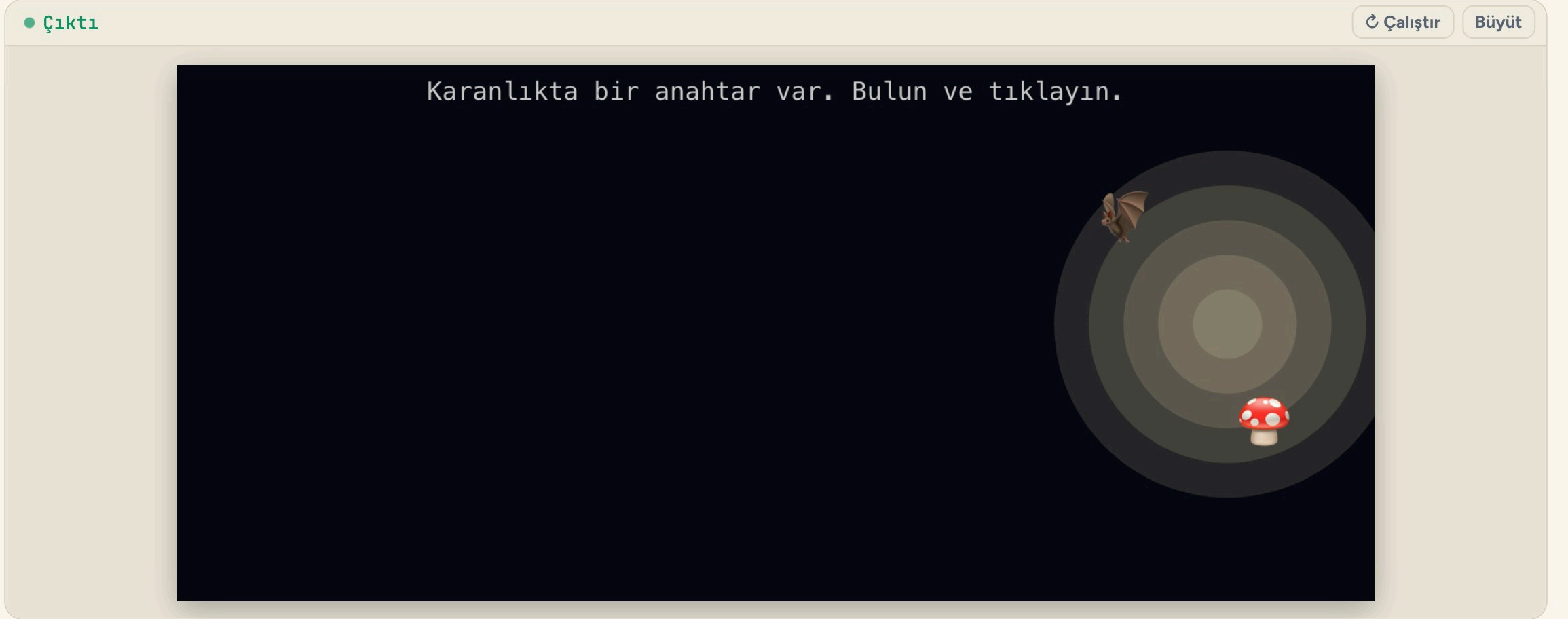
● Çıktı

Çalıştır

Büyüt



Aynı test, başka oyun: el feneri



Bir nesne yalnızca `dist(fare, nesne) < 110` ise çiziliyor. Korku oyunlarındaki el feneri, strateji oyunlarındaki “savaş sisi” aynı sorunun cevabına bakar.

SORU

Tıklayınca skor 1 artsın. Bunu `draw` içine yazarsak?

sketch.js

Kopyala

Büyüt

```
if (mouseIsPressed) {  
  skor = skor + 1;  
}
```

Bir kez tıkladınız. Skor kaç olur?

Tek tıklama, onlarca puan

sketch.js

Kopyala

Düzenle

Büyüt

```
1 let drawdaSayilan = 0;
2 let olaydaSayilan = 0;
3
4 function setup() {
5   createCanvas(400, 240);
6 }
7
8 function draw() {
9   if (mouseIsPressed) {
10     drawdaSayilan = drawdaSayilan + 1;
11   }
12   background(240);
13   fill(0);
14   textSize(22);
15   text("draw + mouseIsPressed: " + drawdaSayilan, 20, 90);
16   text("mousePressed(): " + olaydaSayilan, 20, 150);
17 }
18
19 function mousePressed() {
20   olaydaSayilan = olaydaSayilan + 1;
21 }
```

Çıktı

Çalıştır

Büyüt

draw + mouseIsPressed: 0

mousePressed(): 0

Tuvale birkaç kez tıklayın. Sonra tuşu basılı tutun: hangi sayaç fırlıyor?

Dünyanın en cömert oyunu



BİR TIK, KIRK PUAN

Fare düğmesi bir saniyenin üçte ikisi kadar basılı kaldı.

Neden?

`draw` saniyede 60 kez çalışır. Parmağınız düğmedeyken her frame “basılı mı?” sorusunun cevabı evettir.

Gerçek oyunlarda da görülür: bir düğmeye bir kez bastığınızda menüde üç satır birden atlanıyorsa sebep büyük ihtimalle budur.

Basılı tutmak ile basma anı aynı şey değil

Yöntem	Ne zaman?	Skor için
<code>mouseIsPressed</code>	Tuş basılı tutulduğu her frame (saniyede ~60)	Tek tıklama birçok kez sayılır
<code>mousePressed()</code>	Tuşa basıldığı tek an	Skor bir kez artar

Kural

Bir kez sayılması gereken olaylar `mousePressed()` içine yazılır.

mousePressed() ile tek tıklamayı yakalamak

sketch.js

Kopyala

Düzenle

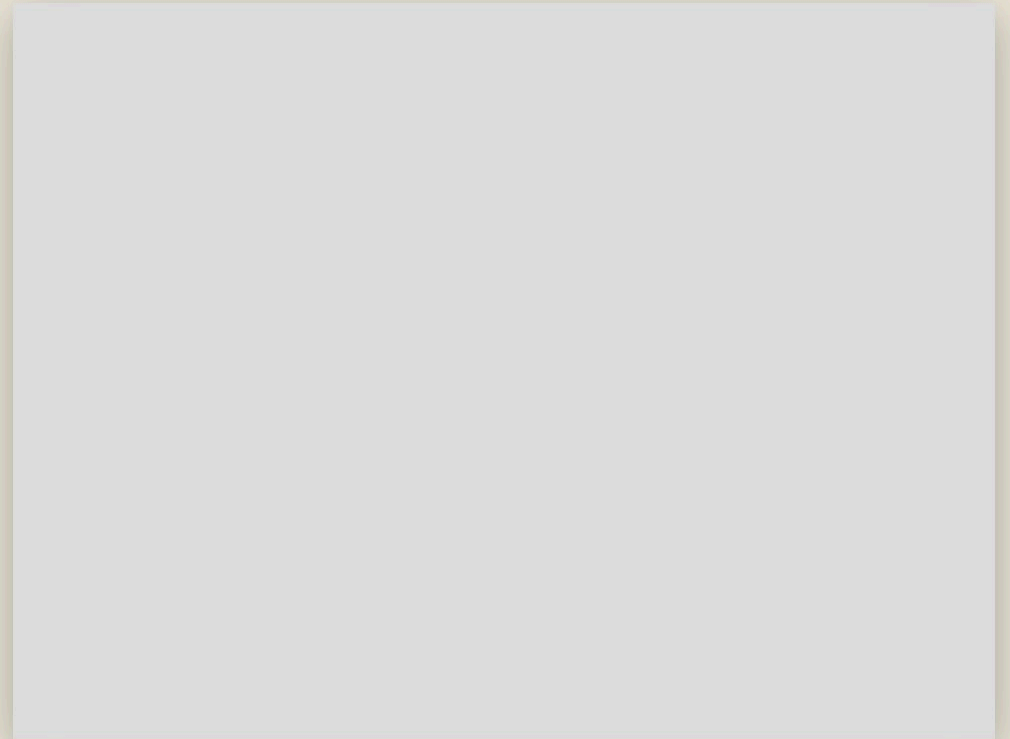
Büyüt

```
1 let tiklandi = false;
2
3 function setup() {
4   createCanvas(400, 300);
5 }
6
7 function mousePressed() {
8   tiklandi = true;
9 }
10
11 function draw() {
12   background(220);
13   if (tiklandi) {
14     fill(0, 180, 0);
15     textSize(20);
16     textAlign(LEFT, TOP);
17     text("Tiklandi!", 150, 140);
18   }
19 }
```

● Çıktı

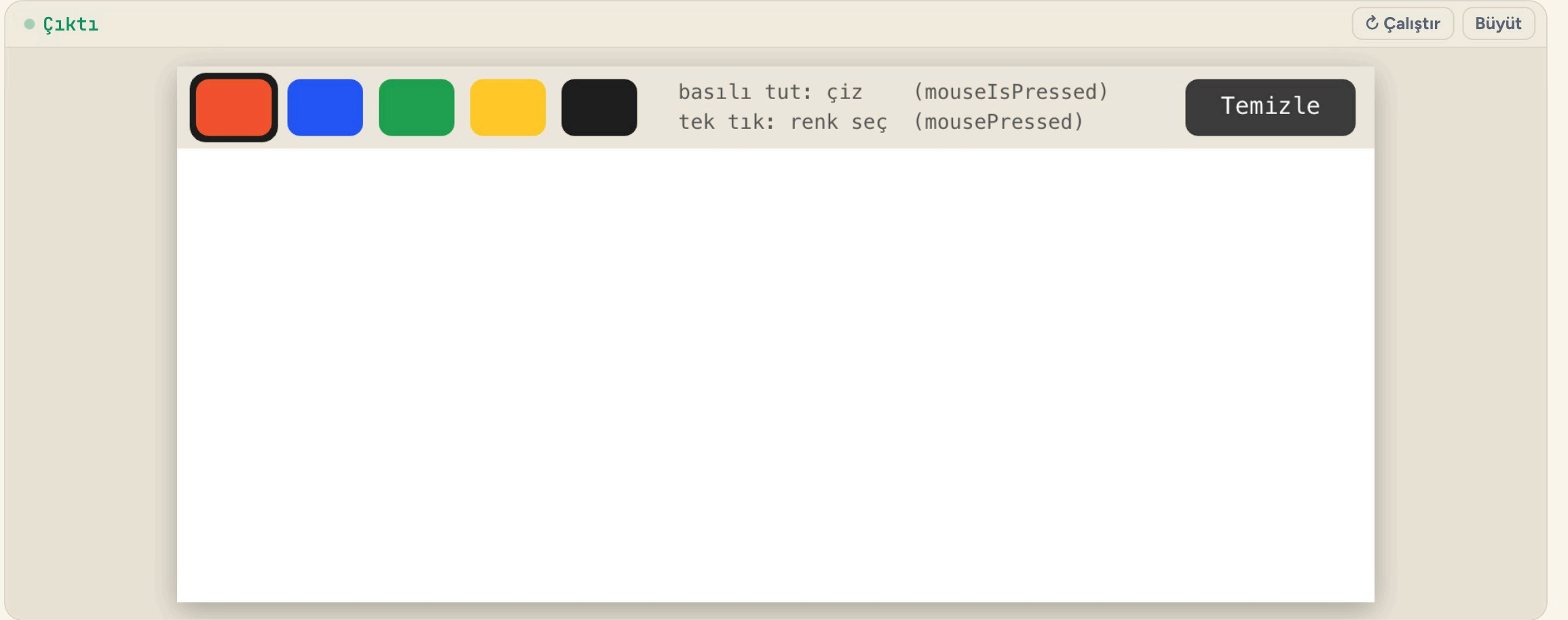
Çalıştır

Büyüt



`tiklandi` bir **bayrak**: tıklama olunca `true` oluyor, `draw` ona bakıp yazıyı gösteriyor.

İkisi bir arada: boya programı



Çizmek süren bir iş: basılı olduğu her frame çizgi uzar. Renk seçmek tek seferlik bir olay: tıklama anında bir kez olur. Bu sketch'te `background` yalnızca `setup`'ta ve "Temizle"de çağrılıyor; çizgiler o yüzden kalıyor.

Daire Tıklama: kurallar

1. Ekranda rastgele bir konumda hedef daire görünür.
2. Oyuncu daireye tıklarsa skor 1 artar; daire yeni bir rastgele konuma taşınır.
3. Daireye isabet etmezse skor değişmez.
4. Skor 10'a ulaştınca oyun kazanılır.
5. Yeni tur için sketch yeniden çalıştırılır; ayrı bir menü yok.

Neye ihtiyacımız var?

Yalnızca değişkenler ve `if / else`.
Döngü ya da dizi gerekmiyor.

Tıklama anında kararlar

- 1 `mousePressed()` tetiklenir.
- 2 Oyun bittiyse hiçbir şey yapma; çık.
- 3 Fare ile hedef arasındaki mesafeyi hesapla.
- 4 Mesafe yarıçaptan küçük değilse: ıskaladı, skor değişmez.
- 5 Küçükse: skoru 1 artır.
- 6 Skor hedefe ulaştıysa oyunu bitir; ulaşmadıysa hedefi yeni bir konuma taşı.

Kutuları hazırla

sketch.js

Kopyala

Büyüt

```
1 let hedefX;           // Hedef dairenin X konumu
2 let hedefY;           // Hedef dairenin Y konumu
3 let hedefYaricap = 40;
4 let skor = 0;
5 const HEDEF_SKOR = 10;
6 let oyunBitti = false;
```

- Hepsi en üstte, `setup` ve `draw` 'ın dışında: oyun boyunca yaşayacaklar.
- `hedefX` ve `hedefY` henüz boş; değerlerini `setup` verecek.
- `HEDEF_SKOR` değişmeyecek: `const` .
- `oyunBitti` bir bayrak: başta `false` .

setup() : tuval ve ilk hedef

sketch.js

Kopyala

Büyüt

```
1 function setup() {  
2   createCanvas(500, 400);  
3   // İlk hedef konumu – random(min, max) bu adımda tanıtıldı  
4   hedefX = random(hedefYaricap, width - hedefYaricap);  
5   hedefY = random(hedefYaricap, height - hedefYaricap);  
6 }
```

random(min, max)

`min` ile `max` arasında rastgele, ondalıklı bir sayı verir; `max` dahil değildir. `random(10, 50)` yaklaşık 10 ile 50 arasında bir değer üretir.

Tek parametrelili `random(max)` 0 ile max arasında, parametresiz `random()` 0 ile 1 arasında değer verir.

SORU

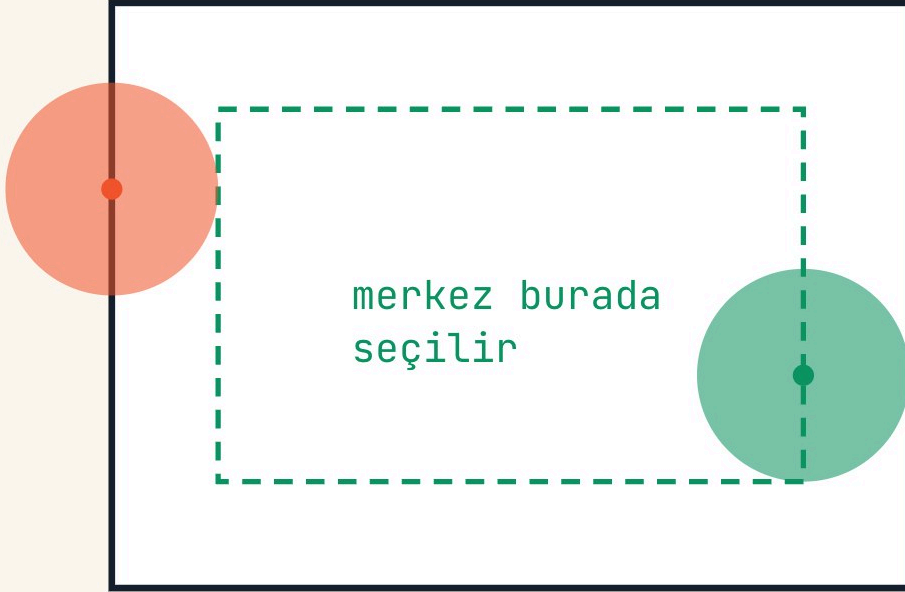
Neden `random(0, width)` yazmadık?

Daha basit görünüyor. Hedefin merkezi tam 0'a ya da tam `width` 'e denk gelirse ne olur?



Daire tuvalden taşar

merkez $x = 0$



ellipse merkezden çizer. Merkez kenara çok yaklaşırsa dairenin bir kısmı dışarıda kalır; oyuncu yarım bir hedefe tıklamaya çalışır.

Çözüm: bir yarıçap içeride kal

x için alt sınır `hedefYaricap`, üst sınır `width - hedefYaricap`. y için aynısı `height` ile.

draw() : çizim ve oyun durumu

sketch.js

Kopyala

Büyüt

```
1 function draw() {
2   background(25, 30, 50); // Koyu arka plan
3
4   if (!oyunBitti) {
5     // Hedef daire
6     fill(255, 180, 0);
7     noStroke();
8     ellipse(hedefX, hedefY, hedefYaricap * 2, hedefYaricap * 2);
9   } else {
10    // Kazanma ekranı
11    fill(255);
12    textSize(32);
13    textAlign(CENTER, CENTER);
14    text("Kazandınız!", width / 2, height / 2);
15  }
16
17  // Skor göstergesi (her durumda)
18  fill(255);
19  textSize(20);
20  textAlign(LEFT, TOP);
21  text("Skor: " + skor, 15, 15);
22 }
```

- `!oyunBitti` : “oyun bitmediyse”. Hedef daire yalnızca o zaman çizilir.
- Oyun bittiyse dairenin yerine kazanma yazısı gelir.
- Skor göstergesi `if` ’in dışında: her durumda çizilir.
- İki yazı için iki ayrı `textAlign` : ortada ve sol üstte.

mousePressed() : tıklama kontrolü

sketch.js

Kopyala

Büyüt

```
1 function mousePressed() {  
2   if (oyunBitti) {  
3     return; // Oyun bittiyse aşağıdaki satırlar çalışmaz  
4   }  
5  
6   let mesafe = dist(mouseX, mouseY, hedefX, hedefY);  
7  
8   if (mesafe < hedefYaricap) {  
9     skor = skor + 1;  
10  
11    if (skor >= HEDEF_SKOR) {  
12      oyunBitti = true;  
13    } else {  
14      hedefX = random(hedefYaricap, width - hedefYaricap);  
15      hedefY = random(hedefYaricap, height - hedefYaricap);  
16    }  
17  }  
18 }
```

return

Bulunulan fonksiyonun geri kalanını çalıştırmadan çıkmasını sağlar.

- Oyun bittiyse tıklamalar yok sayılır.
- İsabet varsa skor 1 artar.
- Hedef skora ulaşıldıysa oyun biter; ulaşılmadıysa hedef yeni konuma taşınır.

Tam kod

sketch.js

Kopyala

Büyüt

```
1 let hedefX;
2 let hedefY;
3 let hedefYaricap = 40;
4 let skor = 0;
5 const HEDEF_SKOR = 10;
6 let oyunBitti = false;
7
8 function setup() {
9   createCanvas(500, 400);
10   hedefX = random(hedefYaricap, width - hedefYaricap);
11   hedefY = random(hedefYaricap, height - hedefYaricap);
12 }
13
14 function draw() {
15   background(25, 30, 50);
16
17   if (!oyunBitti) {
18     fill(255, 180, 0);
19     noStroke();
20     ellipse(hedefX, hedefY, hedefYaricap * 2, hedefYaricap * 2);
21   } else {
22     fill(100, 255, 100);
23     textSize(32);
24     textAlign(CENTER, CENTER);
25     text("Tebrikler! 10 puana ulaştınız.", width / 2, height / 2);
26   }
27
```

```
28   fill(255);
29   textSize(20);
30   textAlign(LEFT, TOP);
31   text("Skor: " + skor + " / " + HEDEF_SKOR, 15, 15);
32 }
33
34 function mousePressed() {
35   if (oyunBitti) {
36     return;
37   }
38
39   let mesafe = dist(mouseX, mouseY, hedefX, hedefY);
40
41   if (mesafe < hedefYaricap) {
42     skor = skor + 1;
43     if (skor >= HEDEF_SKOR) {
44       oyunBitti = true;
45     } else {
46       hedefX = random(hedefYaricap, width - hedefYaricap);
47       hedefY = random(hedefYaricap, height - hedefYaricap);
48     }
49   }
50 }
```

Turuncu daireye tıklayın: 10 isabet

sketch.js · satır 34-50

Kopyala

Büyüt

```
34 function mousePressed() {  
35   if (oyunBitti) {  
36     return;  
37   }  
38  
39   let mesafe = dist(mouseX, mouseY, hedefX, hedefY);  
40  
41   if (mesafe < hedefYaricap) {  
42     skor = skor + 1;  
43     if (skor >= HEDEF_SKOR) {  
44       oyunBitti = true;  
45     } else {  
46       hedefX = random(hedefYaricap, width - hedefYaricap);  
47       hedefY = random(hedefYaricap, height - hedefYaricap);  
48     }  
49   }  
50 }
```

Çıktı

Çalıştır

Büyüt

Skor: 0 / 10



Bu oyunda olmayanlar

Klavye

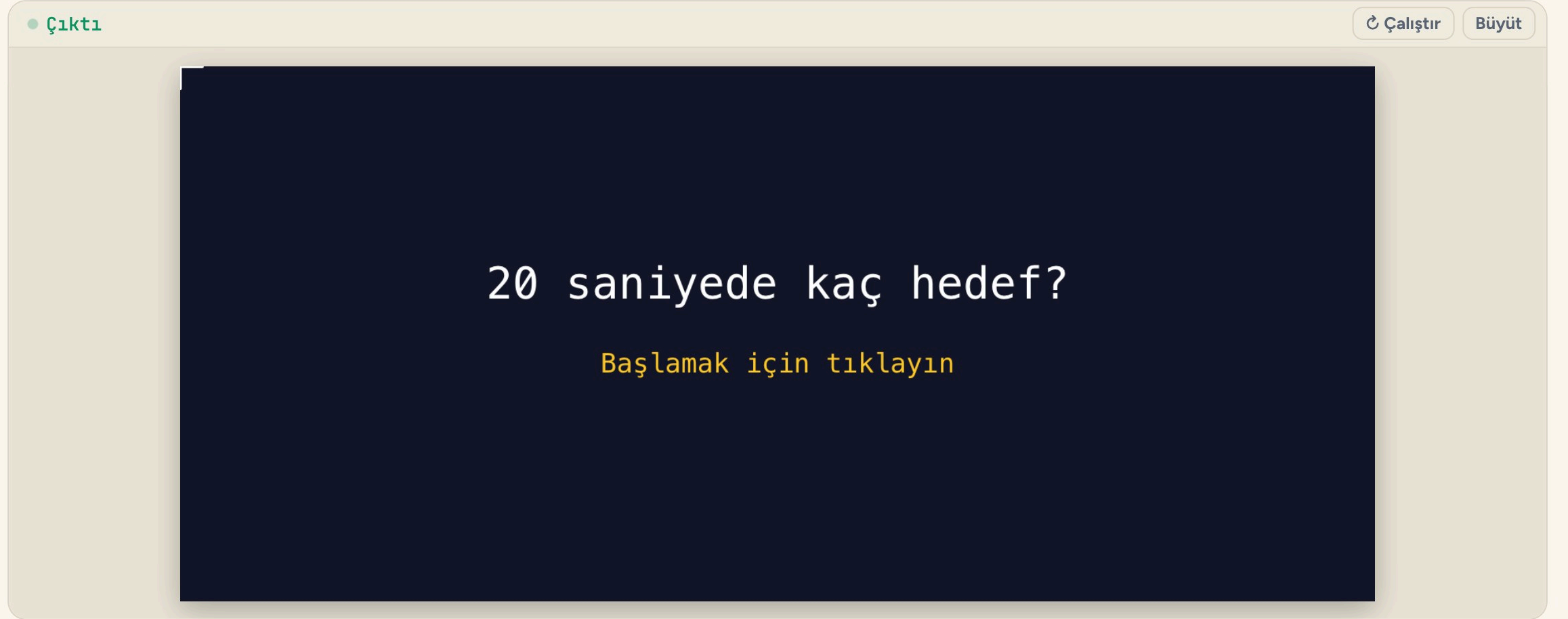
`keyIsDown` ve `keyPressed()` kullanmadık.
Klavye girdisini ilerleyen bölümlerde ele alacağız.

Birden fazla hedef

Aynı anda birçok daire için dizilere ihtiyaç var. O da sonraki bölümlerin konusu.

Bu oyun, bir oyunun en temel döngüsünü gösteriyor: **girdi → karar → geri bildirim.**

Bir adım ötesi: süre, küçülen hedef, ıska



Çekirdek aynı: `dist` ile isabet testi ve `mousePressed`. Üstüne üç küçük fikir eklendi. Bunları “Oyunu zorlaştırın” etkinliğinde kendiniz yazmayı deneyeceksiniz.

Kendi cümlelerinizle açıklayın

⌚ 10 dk

1. Skorun bir basışta birden fazla artmaması için `mousePressed` ile `draw` içindeki `mouseIsPressed` arasındaki farkı anlatın. Yanlış tasarımda ne olurdu?
2. Hedef konumunda neden `random(hedefYaricap, width)` ya da `random(0, width)` yazmadık? Bu ikisinden birini kullansaydık görünüm ve tıklanabilirlik nasıl değişirdi? Editörde deneyip gözleyin.

Oyunu zorlaştırın

 25 dk

1. Dairenin dışına tıklanınca skor **3 puan azalsın**.
2. Skor sıfırın altına düşerse oyun bitsin ve ekranda “Üzgünüm, oyunu kaybettiniz” yazsın.
3. Kazanma ve kaybetme mesajlarını skora bakarak ayırt edin.

İpucu

`mousePressed` içindeki `mesafe < hedefYaricap` dalına bir `else` ekleyin. `draw` içinde iki mesajı skor ile ayırın.

Zorlu görevler

 30 dk

Küçülen hedef

Her isabette yarıçap 3 piksel küçülsün ama 15'in altına inmesin.

İsabet oranı

Her tıklamayı sayın. Sol üstte "7 / 12" gibi isabet / deneme gösterin. Oyun bitince tıklamalar sayılmasın.

Süre

Oyuncunun 15 saniyesi olsun. Kalan süreyi ekrana yazın; süre bitince oyun bitsin. İpucu:

```
frameCount / 60.
```

Yapay zekâyla: nokta elipsin içinde mi?

 20-25 dk

1. Yapay zekâdan, (mouseX, mouseY) noktasının bir dairenin içinde olup olmadığını denetleyen kısa bir kod ve formülün terim terim açıklamasını isteyin.
2. İlk yanıt çoğu zaman yüzeysel kalır. Soruyu ilerletin: “Aynısını `dist` kullanmadan `sqrt` ile yaz; Pisagor bağıntısıyla ilişkisini göster.”
3. Kendi fonksiyonunu tanımlarsa geri çevirin: henüz fonksiyon yazmayı görmedik.
4. Aldığınız kodu `dist(mouseX, mouseY, hedefX, hedefY) < hedefYaricap` ifadesiyle karşılaştırın.

Yapay zekâyla: dokunmatik ekran

 15-20 dk

1. Sorun: “p5.js sketch’imi dokunmatik ekranda kullanacağım. `mousePressed` / `mouseX` yerine dokunmayı nasıl algılarım?”
2. Yanıttaki adları (örneğin `touches` , `touchStarted`) p5.js referansından doğrulayın.
3. Sketch’inizi telefonda ya da tablette nasıl açacağınızı sorun ve önerilen yollardan birini deneyin.

`dist(0, 0, 3, 4)` kaç döndürür?

A 7

✓ 5

C 12

D 25

Pisagor: $3^2 + 4^2 = 25$, karekökü 5. `dist` yatay ve dikey farkları toplamaz.

Oyuncu tam dairenin kenarına tıkladı: mesafe = 40, yarıçap = 40.

sketch.js

Kopyala

Büyüt

```
if (mesafe < hedefYaricap) {  
    skor = skor + 1;  
}
```

☐ A İsabet sayılır

☒ B Iska sayılır

40 < 40 yanlıştır. Kenarı da isabet saymak isteseydik <= yazardık. Hangisini istediğinize siz karar verirsiniz.

mousePressed 'in başındaki **return** 'lü **if** 'i silersek?

- A** Hiçbir şey değişmez.
- B** Oyun hiç bitmez.
- ☒ **C** Oyun bittikten sonra da, eski hedefin olduğu yere tıklanınca skor artmaya devam eder.
- D** Program hata verir.

Daire artık çizilmiyor ama `hedefX` ve `hedefY` hâlâ duruyor. İsabet testi çalışır; skor 11, 12... olur.

Merak edenler için kaynaklar

JEFFREY THOMPSON

Collision Detection: Point / Circle

"Nokta dairenin içinde mi?" testinin çizimli, etkileşimli anlatımı.

[jeffreythompson.org > collision-detection > point-circle](http://jeffreythompson.org/collision-detection/point-circle)

THE CODING TRAIN

Variables: mouseX, mouseY

Fare konumunu okumak ve çizime bağlamak.

[thecodingtrain.com > 2-variables > 1-mouse](http://thecodingtrain.com/2-variables/1-mouse)

THE CODING TRAIN

random()

Hedefi rastgele bir yere koymak için.

[thecodingtrain.com > 2-variables > 4-random](http://thecodingtrain.com/2-variables/4-random)

HAPPY CODING

Input

Fare ve klavye olayları üzerine yazılı anlatım (İngilizce).

[happycoding.io > tutorials > p5js > input](http://happycoding.io/tutorials/p5js/input)

Bölüm 1'i bitirdik

- ✓ `mouseX`, `mouseY` imlecin konumunu verir; her frame güncellenir.
- ✓ `mouseIsPressed` basılı tutulduğu her frame doğrudur; `mousePressed()` basma anında bir kez çalışır.
- ✓ `text`, `textSize`, `textAlign` ile ekrana yazı yazılır.
- ✓ `dist` < yarıçap: nokta dairenin içinde.
- ✓ `random(min, max)` ile hedef, kenardan bir yarıçap içeride seçilir.
- ✓ `return` fonksiyondan erken çıkar; `oyunBitti` bayrağı oyunu durdurur.

SIRADAKİ BÖLÜM

Bölüm 2: Döngüler ve fonksiyonlar

Aynı satırı yüz kez yazmadan yüz şekil çizmek.

[Bölüm 2'ye geç →](#)

[Tüm sunumlar](#)

