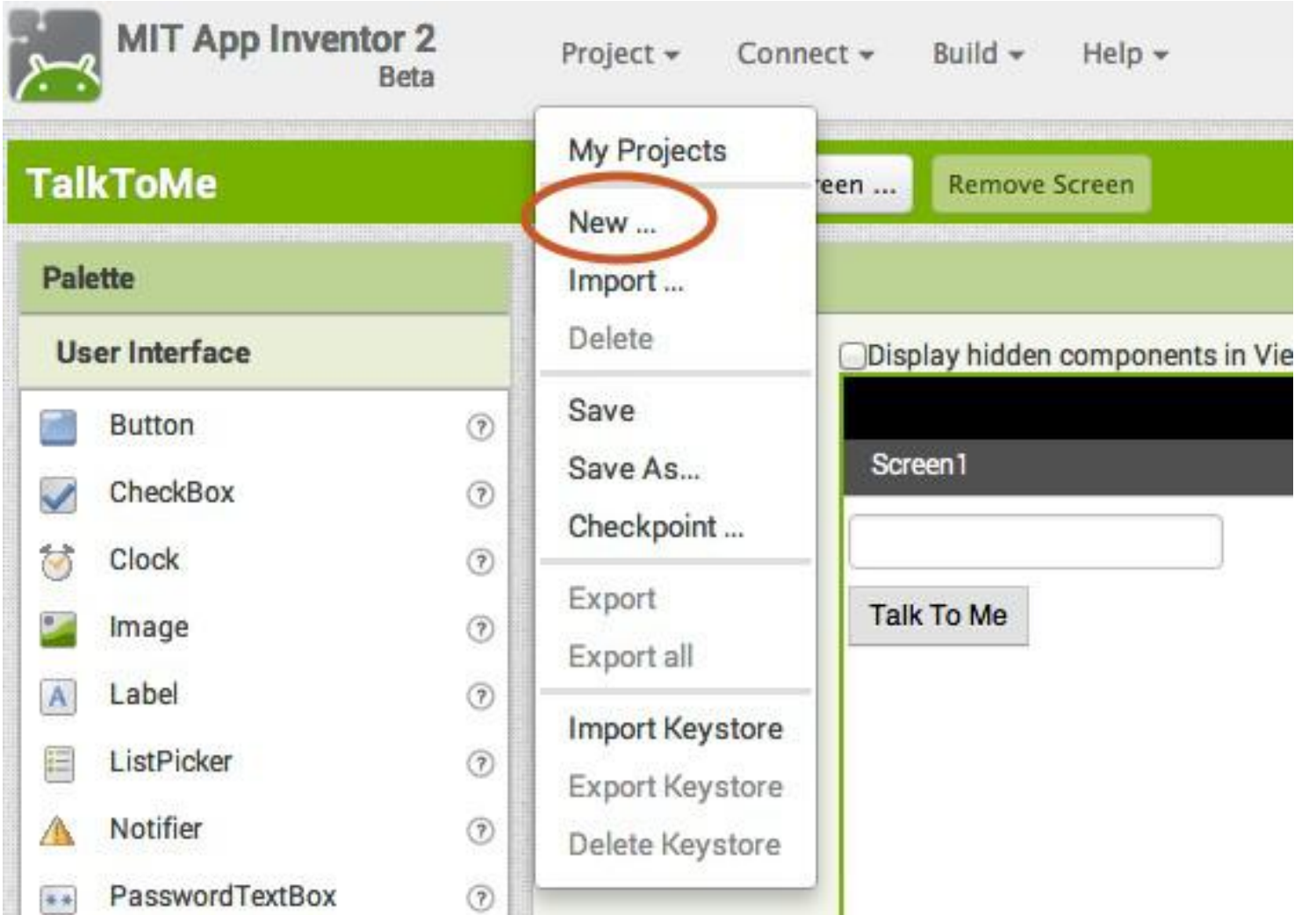




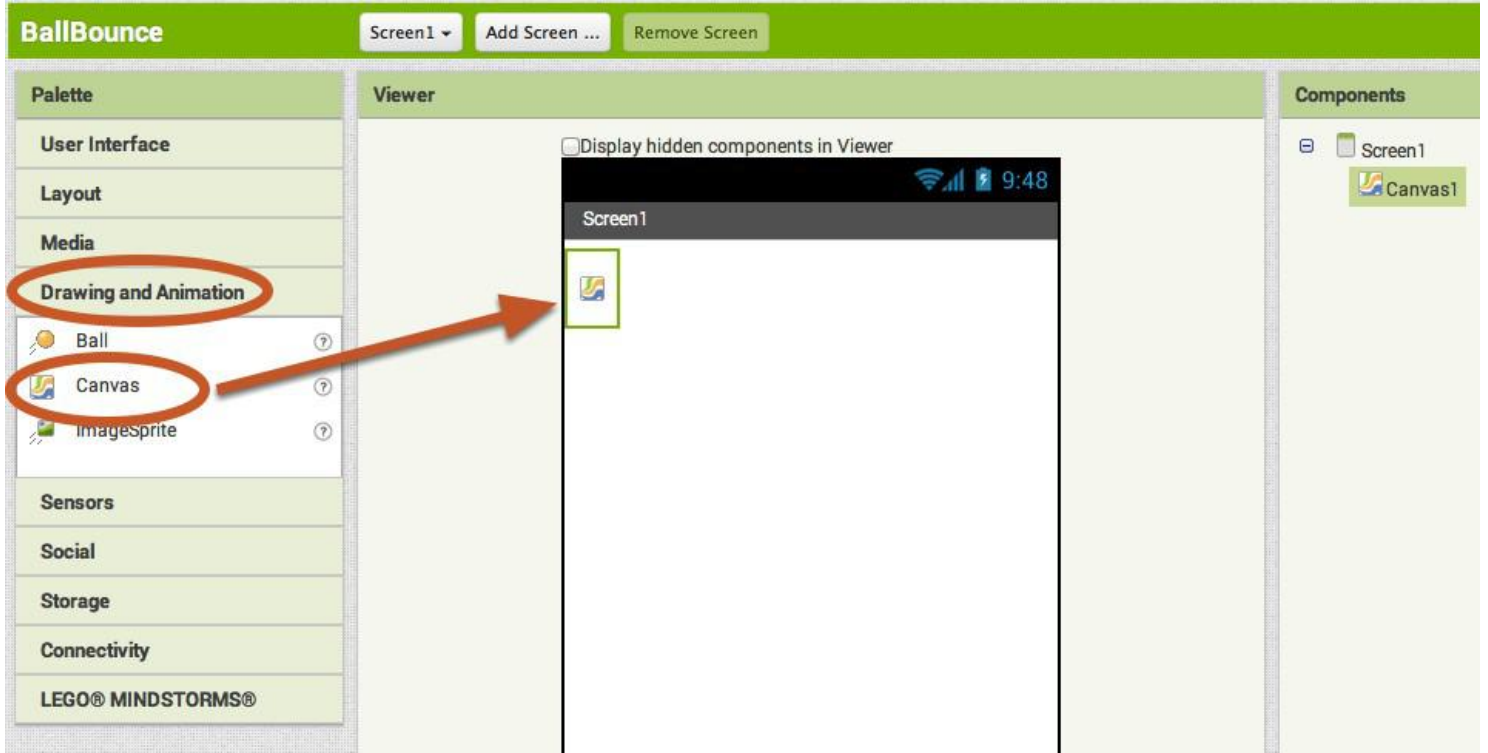
## BallBounce: Basit bir oyun uygulaması

Bu uygulamada Canvas mantığını öğrenerek o canvas'ta bir top'un (ball bileşeni) hareket etmesini sağlarız.

### Yeni bir proje oluşturup isimlendirin



**Drawing and Animation** bölümünden, **Canvas** bileşenini görüntüleyici (viewer) bölümüne sürükleyip bırakın.





## Ekranı kaydırılmaz yapalım ki hareket etmesin

Özellikle Canvas kullanırken "**Scrollable**" ayarını (kutucuktaki işareti kaldırarak) pasif hale getirerek ekranı hareket etmeyecek hale getirebiliriz.

The screenshot shows the MIT App Inventor interface for the 'BallBounce' app. The interface is divided into four main panels: Palette, Viewer, Components, and Properties. The Palette panel on the left shows various components like Ball, Canvas, and ImageSprite. The Viewer panel in the center shows a preview of the app with a black ball on a white canvas. The Components panel on the right shows a tree view with 'Screen1' containing 'Canvas1' and 'Ball1'. The Properties panel on the far right shows settings for 'Screen1', including 'Scrollable' which is currently unchecked. Two callout boxes highlight the 'Scrollable' checkbox in the Properties panel and the 'Scrollable' checkbox in the Components panel, indicating that unchecking it will prevent the screen from scrolling.



## Canvas yükseklik ve genişliğini ekranı kaplayacak hale getirelim

Dİğer örneklerde gördüğümüz gibi bunun için "Fill Parent" ayarını hem genişlik hem yükseklik için uygulayacağız.

The screenshot shows the MIT App Inventor interface for a project named "BallBounce". The "Components" panel on the right shows "Screen1" selected, with "Canvas1" highlighted. A circled "1" is next to the "Canvas1" component. The "Properties" panel on the right shows the "Canvas1" properties. A circled "2" is next to the "Canvas1" property. The "Height" property is set to "Fill parent". A dialog box is open over the "Height" property, showing the "Automatic" option selected, with "Fill parent" and "pixels" options also visible. An arrow points to the "Fill parent" option in the dialog box.



## Ball bileşenini eklemek

Bu kez Canvas'ın içine Ball bileşenini de ekleyelim (#1). Eğer topun daha büyük görünmesini istiyorsanız Radius (çap) ayarını değiştirebiliriz.

The screenshot displays the MIT App Inventor interface with four main panels: Palette, Viewer, Components, and Properties. In the Palette panel, the 'Drawing and Animation' section is expanded, and the 'Ball' component is circled in red with a '1' in a black circle. An orange arrow points from the 'Ball' component to a black circle on the Canvas in the Viewer panel. In the Components panel, 'Ball1' is listed under 'Canvas1'. In the Properties panel, the 'Radius' property is circled in red with a '2' in a black circle, and its value is set to 15. Other properties like 'Enabled', 'Heading', 'Interval', 'PaintColor', 'Speed', 'Visible', 'X', 'Y', and 'Z' are also visible.

## Open the Blocks Editor.





## Şimdi bloklar bölümüne geçelim.

The screenshot shows the MIT App Inventor interface for a project named "BallBounce". The "Blocks" panel on the left lists various categories like Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. Under "Screen1", there is a "Ball1" component highlighted with a red circle. The "Viewer" panel on the right displays several event-driven code blocks for "Ball1":

- when Ball1 .CollidedWith**: Includes an "other" block and a "do" field.
- when Ball1 .Dragged**: Includes input fields for startX, startY, prevX, prevY, currentX, and currentY, and a "do" field.
- when Ball1 .EdgeReached**: Includes an "edge" block and a "do" field.
- when Ball1 .Flung**: Includes input fields for x, y, speed, heading, xvel, and yvel, and a "do" field.
- when Ball1 .No longer Colliding With**: (Partially visible)

## Flung tetikleyicisini sürükleyin

Bu tetikleyici kullanıcı parmağını topun üstüne koyup fırlatma işlemi yaptığında tetiklenecektir.





## Topun yönünü ve hızını ayarlayın

Bunun için **set Ball1.Heading** and **set Ball1.Speed** bloklarını kullanacağız.

The screenshot shows the MIT App Inventor interface. On the left is the 'Blocks' pane, and on the right is the 'Viewer' pane. In the 'Blocks' pane, the 'Ball1' component is selected and circled in orange. In the 'Viewer' pane, a sequence of blocks is shown for the 'Ball1' component. The 'set Ball1 . Heading to' block and the 'set Ball1 . Speed to' block are circled in orange. A callout box with a red arrow points to the 'set Ball1 . Heading to' block, containing the text: 'Scroll down to the green "setter" blocks for the Ball's Heading and Speed'.



```
when Ball1 .Flung
  x y speed heading xvel yvel
do
  set Ball1 . Speed to
  set Ball1 . Heading to
```

### Hız ve yönü kullanıcının yaptığı fırlatma hareketinden alacak kodları ekleyelim

Bunun için fare imlecini **when Ball1.Flung** üzerinde tutup menü açılana kadar tıklamadan bekleyin. Açılan menüden get.speed bloğunu sürükleyip set Ball1.speedTo bloğu ile birleştirin.

```
when Ball1 .Flung
  x y speed heading xvel yvel
do
  set Ball1 . Speed to
  set Ball1 . Heading to
```

1 Hold mouse over speed without clicking

2 Grab "get Speed" block





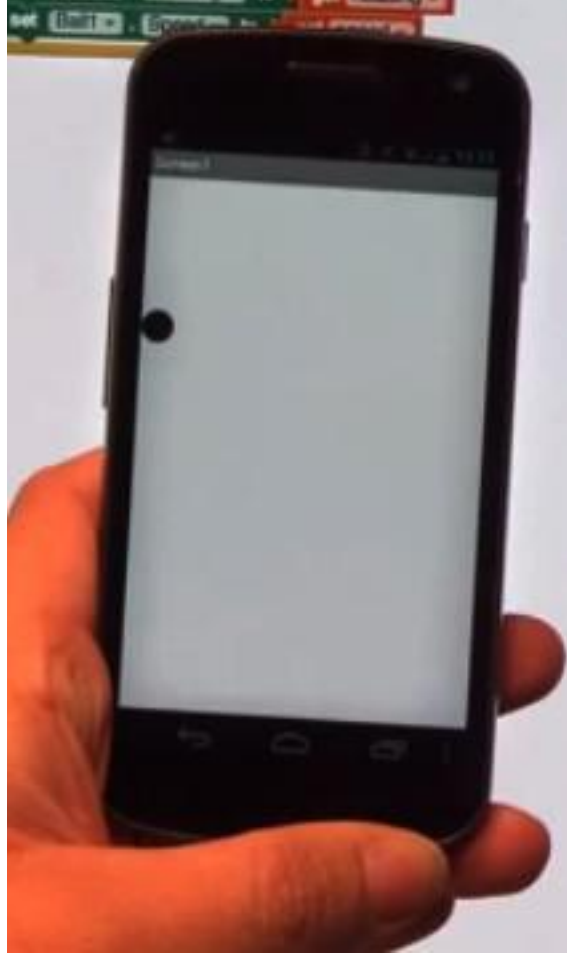
Aynısını yön için yapın

```
when Ball1 .Flung
  x y speed heading xvel yvel
do
  set Ball1 . Speed to get speed
  set Ball1 . Heading to get heading
```



## Şimdi test edin

Test ettiğinizde topun köşlere çarpınca yapışıp kaldığını farkedeceksiniz.



## Neden ekran kenarına yapıştı?

Ekranın kenarlardan sekmesi için Bounce yani sekme bloğunu kullanacağız. Bunu tetikleyecek işlem içinse kenara dokunduğunda başlayacak bir tetikleyiciye ihtiyacımız olacak. Bunun için "When Edge Reached" bloğunu sürükleyip blok bölümüne bırakalım.

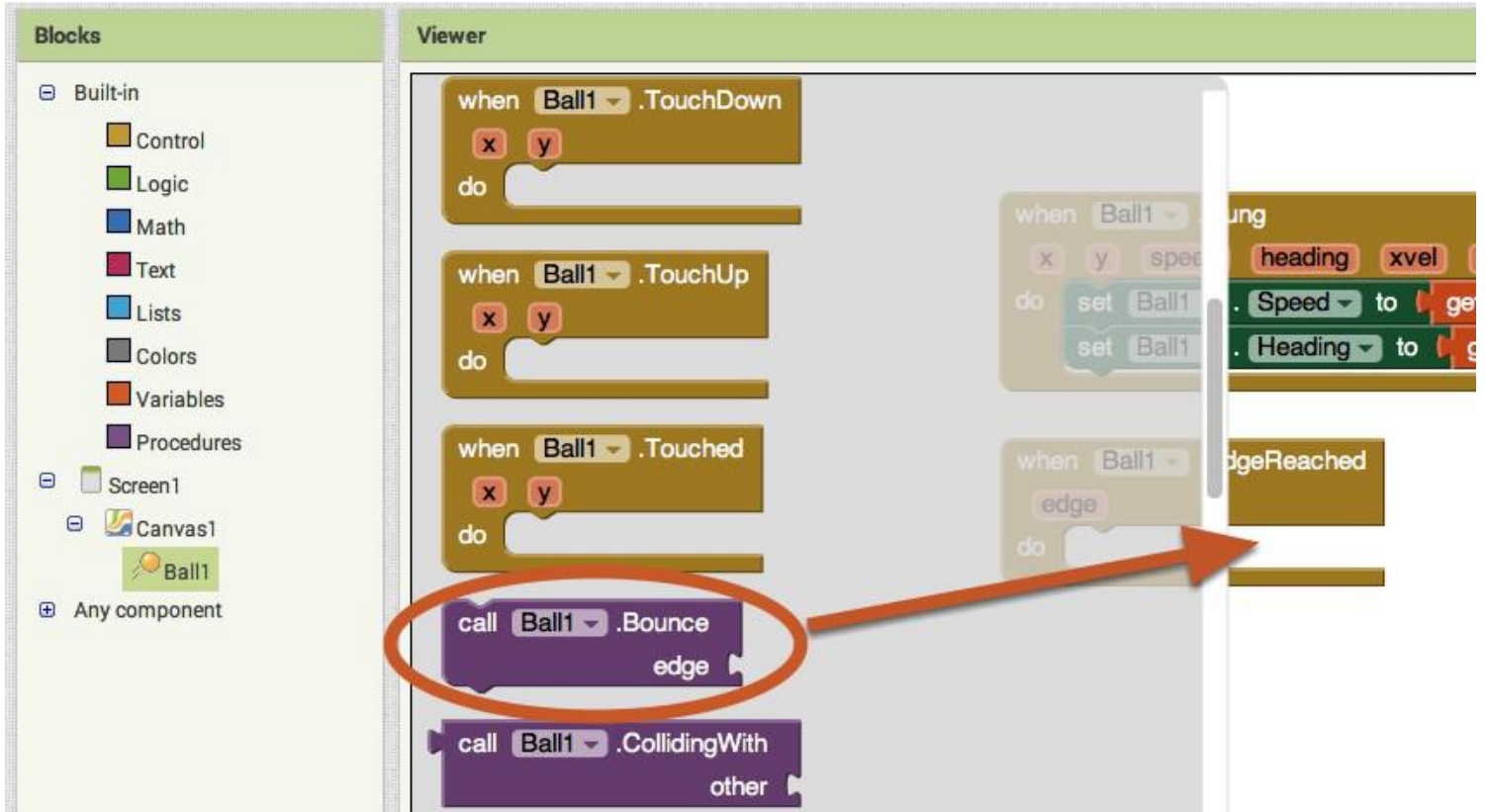


## Add an Edge Reached Event

Go into the Ball1 drawer and pull out a **when Ball1.EdgeReached do** event.

The screenshot shows the MIT App Inventor interface. On the left is the 'Blocks' drawer, and on the right is the 'Viewer' workspace. In the 'Blocks' drawer, the 'Ball1' component is selected and circled in red. In the 'Viewer' workspace, several event blocks are visible. The 'when Ball1.EdgeReached do' block is highlighted with a red circle, and a red arrow points to it from the right. Other visible blocks include 'when Ball1.CollidedWith', 'when Ball1.Dragged', 'when Ball1.Moving', and 'when Ball1.NoLongerCollidingWith'.

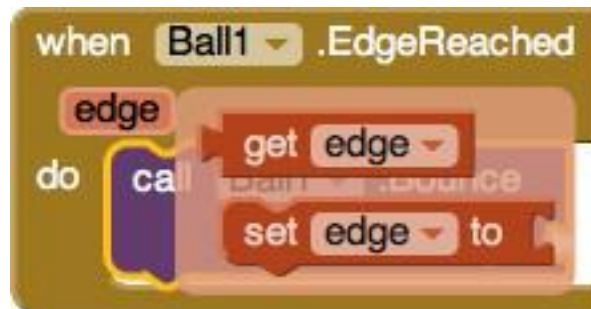
Go back into the Ball1 drawer and pull out a Ball.Bounce block.



The screenshot shows the MIT App Inventor interface. On the left is the 'Blocks' pane with categories like Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. On the right is the 'Viewer' pane showing a code block for 'when Ball1.Touched' with a 'do' loop containing a 'call Ball1.Bounce' block. An orange circle highlights the 'call Ball1.Bounce' block, and an orange arrow points from it to the 'EdgeReached' block in the Viewer pane.

Şimdi kenar değerini ekleyeceğiz

Bounce bloğu bir kenar değerine ihtiyaç duymaktadır. Bunun için yine fareyi edge parametresi üzerinde bekleterek get edge bloğunu sürükleyip Bounce bloğu ile birleştireceğiz.



The screenshot shows a close-up of the 'when Ball1.EdgeReached' code block. The 'do' loop contains a 'get edge' block, a 'call Ball1.Bounce' block, and a 'set edge to' block.



Blokların son hali aşağıdaki gibi olmalıdır

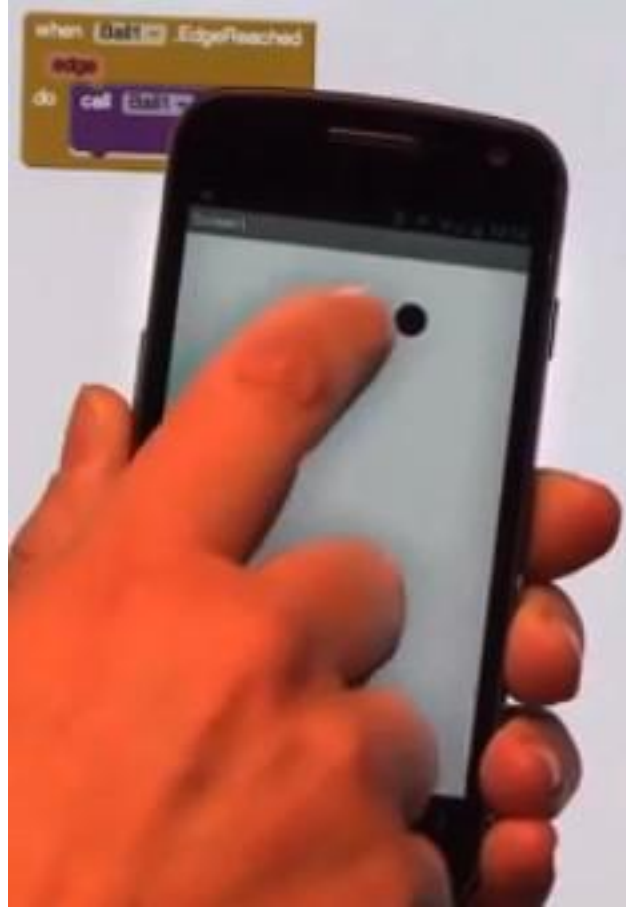
```
when Ball1 .Flung
  x y speed heading xvel yvel
do
  set Ball1 . Speed to get speed
  set Ball1 . Heading to get heading

when Ball1 .EdgeReached
  edge
do
  call Ball1 .Bounce
  edge get edge
```



## Test eout!

Test edelim.



## Bu uygulamayı geliřtrimenin birok yolu var!

Bazı fikirler ařađıda... ama sınırlar size ait

- Topun rengini hıza bađlı olarak deđiřtirin.
- Topun arpma sayısını ekrana yazdırma
- Topun arpacağı bařka engeller