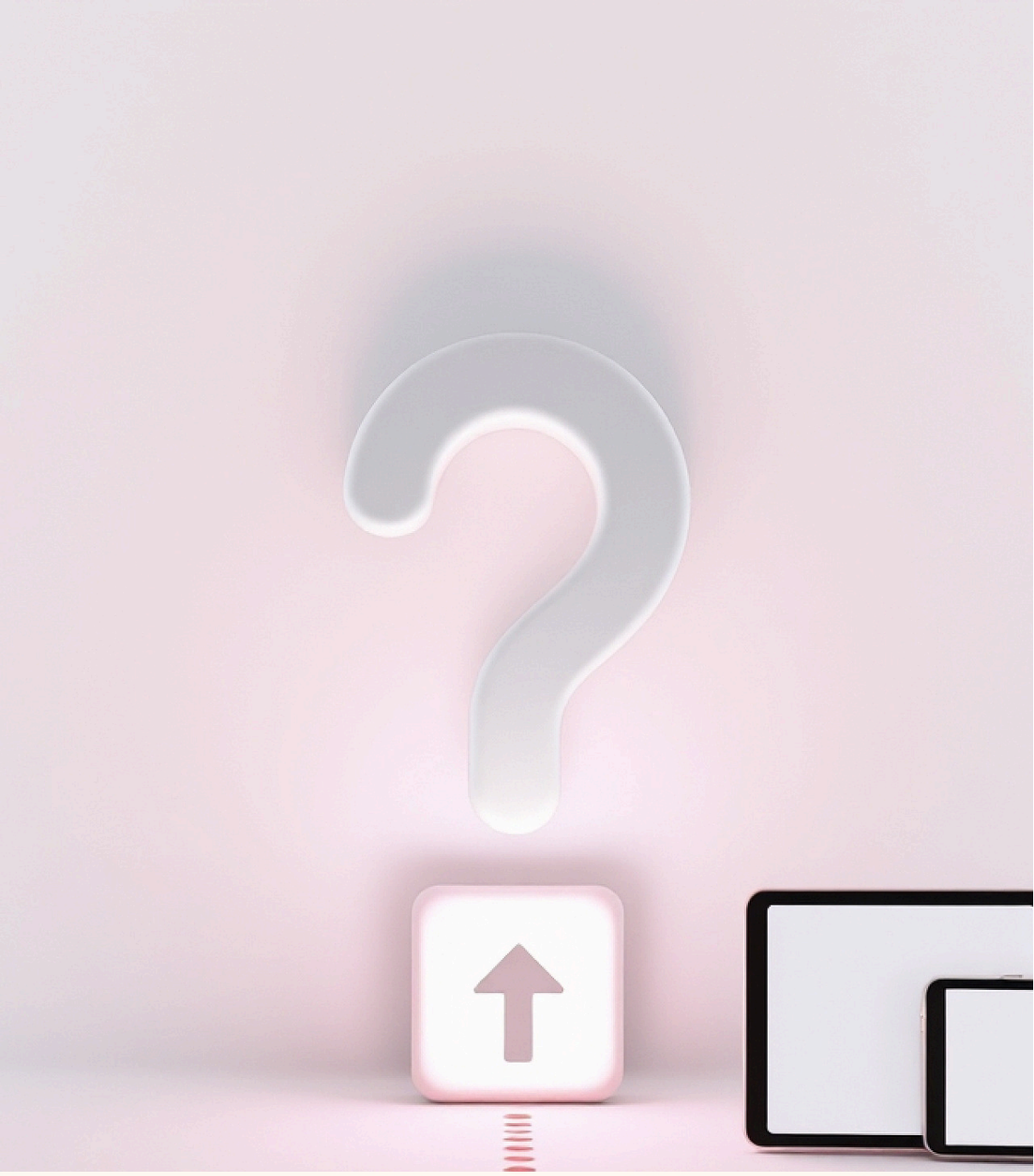
Bir oyundaki karakter klavyeyle nasıl hareket ettirilir?



www.yzokul.com.tr

Bir oyundaki karakter klavyeyle nasıl hareket ettirilir?



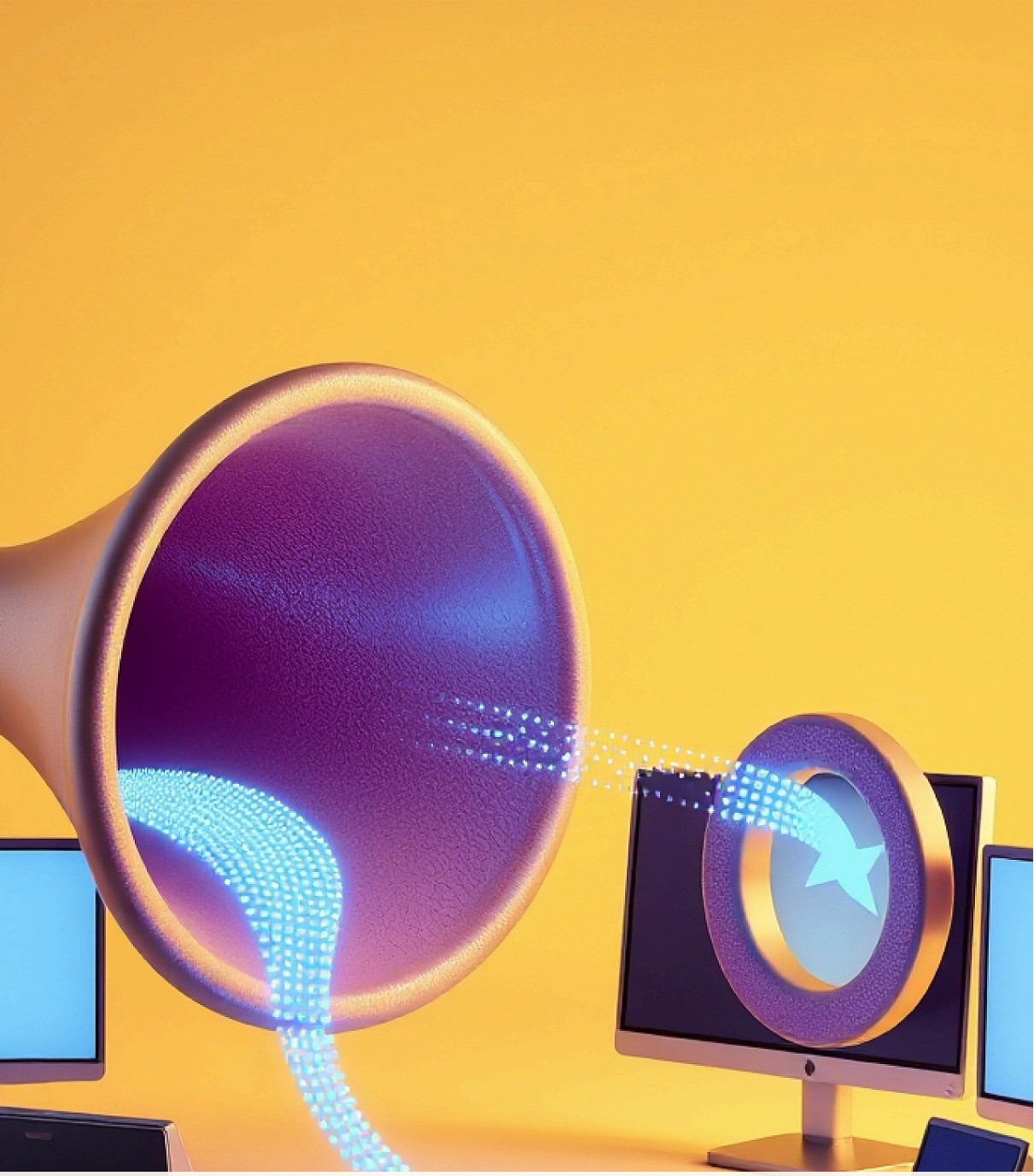
Bir oyun karakteri klavyeyle nasıl hareket ettirilir? Kısa yanıtı, programın hangi tuşlara basıldığını denetlemesi ve buna göre karakterin konumunu değiştirmesidir. Program klavyeden sinyaller alır, hangi tuşun basılı olduğuna karar verir ve karakterin konum sayılarını günceller. Bu kitap her parçayı sıfırdan başlayarak adım adım açıklar.



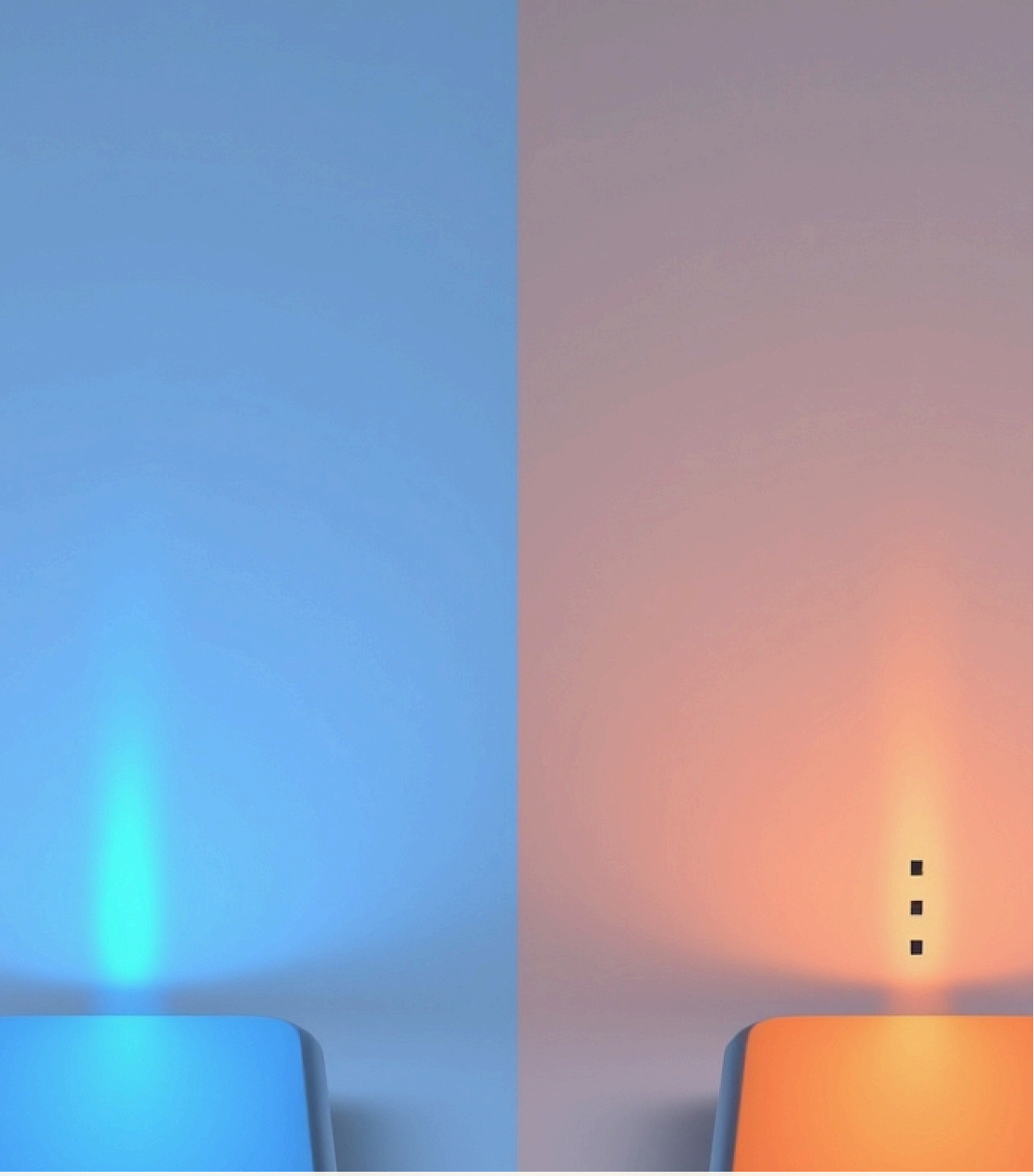
Klavyedeki her tuş basıldığında bilgisayara bir sinyal gönderir. Program bu sinyalleri girdi olarak alabilir. Bu sinyallere klavye girdisi denir. Bir tuşa bastığında bilgisayar, belirli bir tuşun basıldığına dair bir mesaj alır. Program bu mesajları kullanarak oyuncunun ne yapmak istediğini anlar.



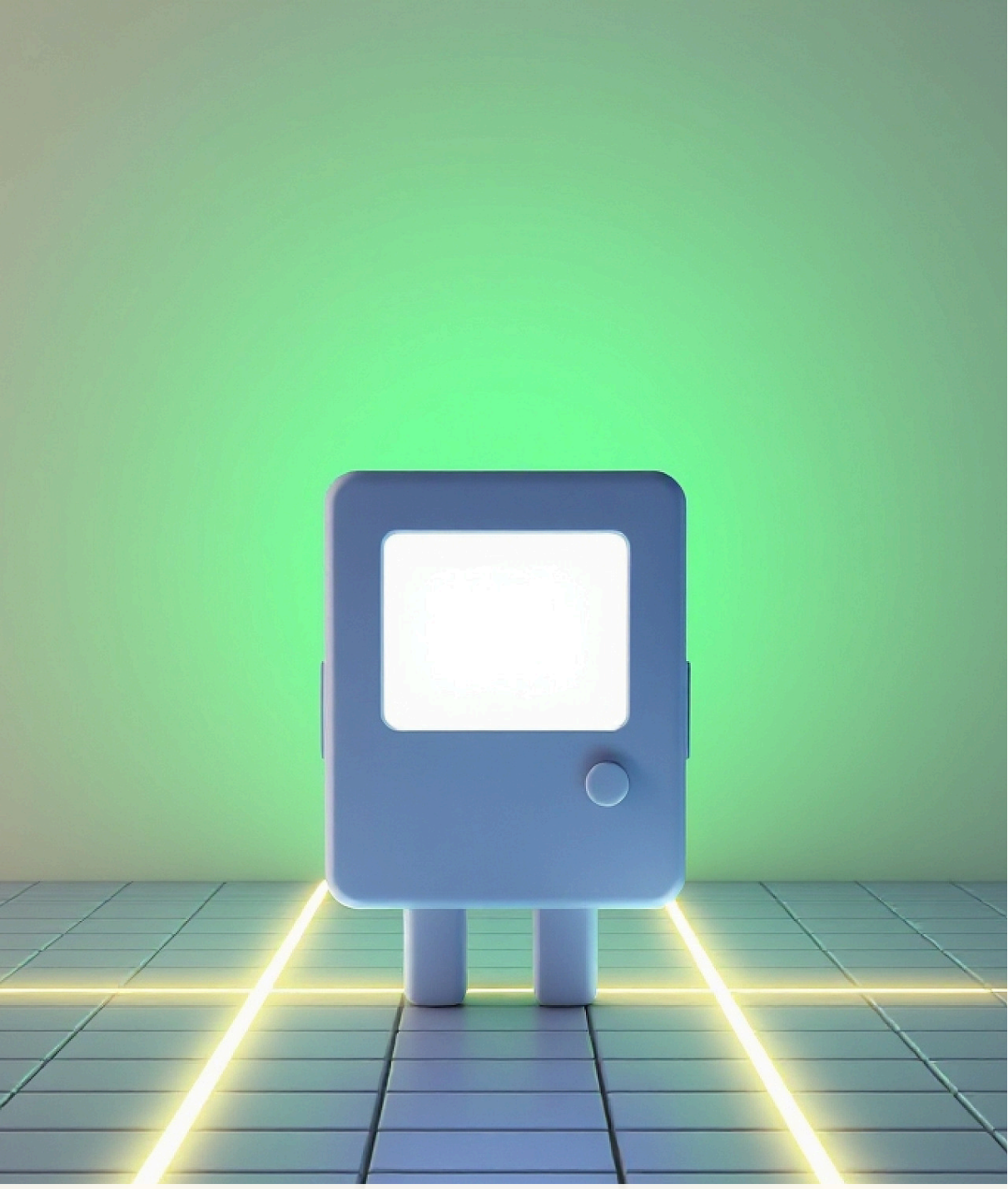
Sağ ok veya A harfi gibi her tuşun kendi adı veya kodu vardır. Program bu ada bakarak hangi belirli tuşa basıldığını anlayabilir. Örneğin sağ ok tuşunun kodu ArrowRight olabilir. Bir tuş olayı geldiğinde program, hangi tuş olduğunu anlamak için kodu denetler. Bu, programın farklı tuşlara farklı yanıt vermesini sağlar.



Program, yani bir tuşun basıldığını veya bırakıldığını duymayı bekleyerek klavye olaylarını denetler. Bu denetim sayesinde program oyuncunun hareket etmek istediğini anlar. Program bu olayları saniyede birçok kez çalışan ana döngüsünde dinler. Bir olay gerçekleştiğinde program hangi tuşun değiştiğini ve basıldığını mı yoksa bırakıldığını mı kaydeder.



Bir tuşa bir kez basılabilir veya basılı tutulabilir. Program, bir tuşun hâlâ basılı olup olmadığını anlamak için o anki durumunu denetleyebilir. Buna tuş basışı algılama veya tuş durumunu denetleme denir. Tuş basılıysa program karakteri hareket ettirmeye devam eder. Tuş bırakılırsa program hareketi durdurur.



Ekrandaki karakterin yatay ve dikey sayı gibi konum değerleri vardır. Karakteri hareket ettirmek bu sayıları değiştirmek demektir. Yatay konum solu ve sağı, dikey konum yukarıyı ve aşağıyı denetler. Program bu sayıları saklar ve karakteri ekranda doğru yere çizmek için kullanır.



Sağ tuşa basılırsa program karakterin yatay konumunu artırır. Bu, karakterin ekranda sağa hareket etmesini sağlar. Örneğin yatay konum 100 ise ve program 5 eklerse karakter 105 konumuna hareket eder. Program, tuş basılı olduğu sürece her güncellemede bunu yapar.



Diğer tuşlar konumu başka şekillerde değiştirir. Sol tuş yatay konumu azaltarak karakteri sola hareket ettirir. Yukarı tuşu dikey konumu azaltır, aşağı tuşu ise ekran koordinatlarının nasıl çalıştığına bağlı olarak onu artırır. Her yön tuşu bir konum değerini kendi şekilde değiştirir, böylece karakter her yöne hareket edebilir.



Program denetimlerini bir döngü içinde saniyede birçok kez tekrarlar. Döngünün her turunda tuşları denetler ve karakterin konumunu günceller. Bu döngüye oyun döngüsü veya güncelleme döngüsü denir. Döngünün saniyede çalışma sayısı kare hızıdır. Daha yüksek kare hızı, saniyede daha çok güncelleme demektir.



Her karede konuma sabit bir miktar eklemek hareket hızını değiştirebilir. Kare hızı değişirse karakter beklenenden daha hızlı veya daha yavaş hareket eder. Örneğin saniyede 60 karede her kareye 5 piksel eklemek saniyede 300 piksel verir. Saniyede 30 karede aynı adım yalnızca saniyede 150 piksel verir. Bu, sabit adım sorunudur.



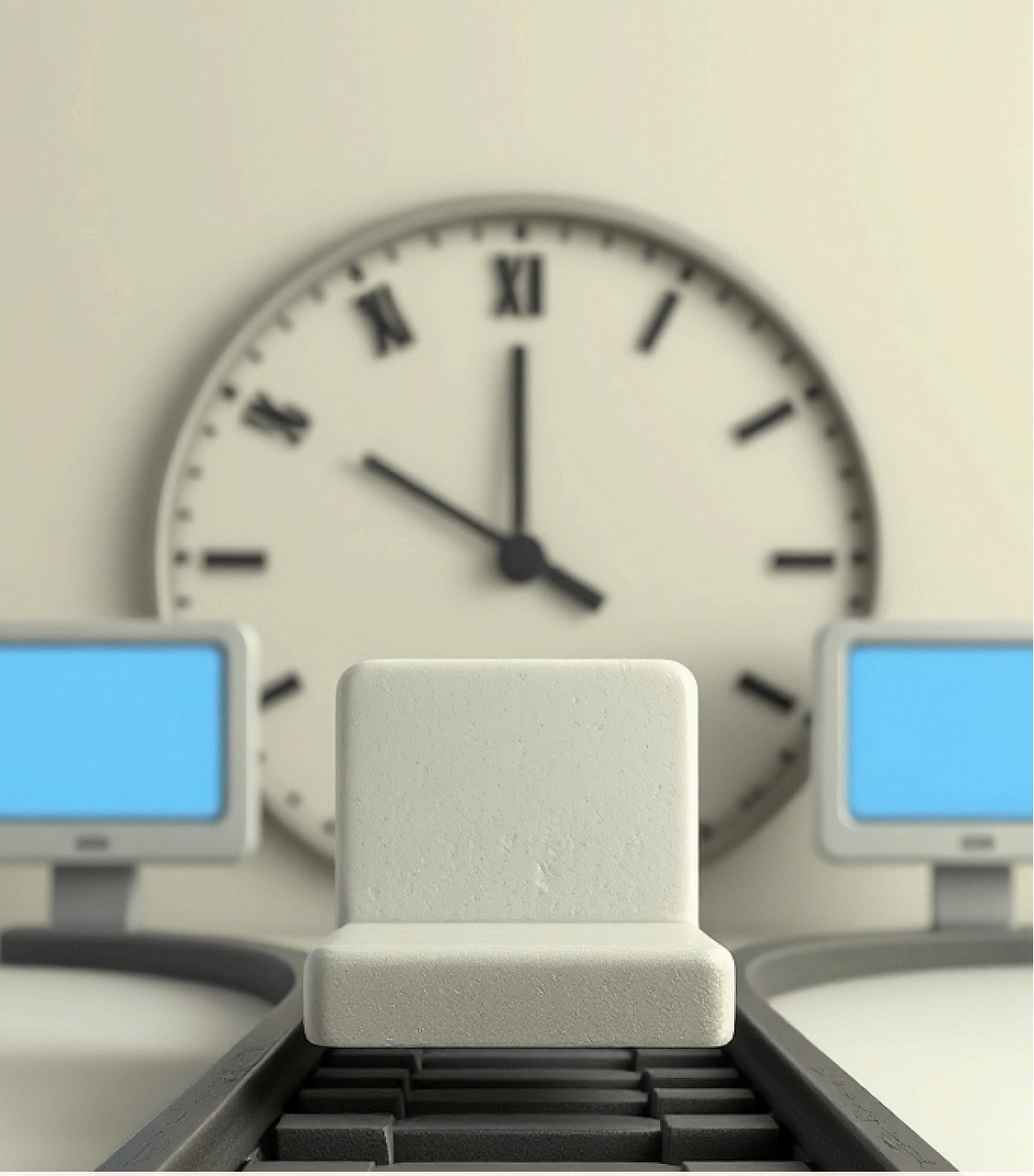
Program son güncellemeden bu yana ne kadar zaman geçtiğini ölçebilir. Bu geçen zaman, programa hareketin ne kadar süre uygulanması gerektiğini söyler. Geçen zaman genellikle saniye veya milisaniye cinsinden ölçülür. Geçen zamanı bilerek program hareketi yalnızca kare sayısına değil, gerçek zamana uyacak şekilde ayarlayabilir.



Hareket miktarı, hız ile geçen zamanın çarpımına eşittir. Bu, hareketi kareleri saymak yerine gerçek zamana bağlar. Örneğin hız saniyede 200 piksel ve geçen zaman 0,016 saniye ise hareket 200 çarpı 0,016, yani 3,2 pikseldir. Program her güncellemede bu miktarı konuma ekler.



Hareketi geçen zamana bağlamak, karakterin hızını farklı cihazlarda aynı tutar. Hızlı bir cihaz da yavaş bir cihaz da karakteri aynı gerçek dünya hızında hareket ettirir. Kare hızı düşse bile geçen zaman artar, böylece hareket miktarı bunu telafi etmek için artar. Bu, oyunun tutarlı hissettirmesini sağlar.



Yaygın bir hata, her karede sabit bir adım hareket ettirmek ve zamanı yok saymaktır. Bu, hızı kare hızına bağımlı kılar; geçen zaman yönteminin çözdüğü sorun da budur. Her kareye her zaman 5 piksel eklersen karakter hızlı bir cihazda daha hızlı, yavaş bir cihazda daha yavaş hareket eder. Bundan kaçınmak için her zaman geçen zamanı kullan.



Bir başka hata, bir tuşa basılıp basılmadığını denetlemeden konumu değiştirmektir. Program önce tuş durumunu denetlemeli, konumu yalnızca bir tuş basılıyken değiştirmelidir. Denetlemeden karakteri hareket ettirirsen hiçbir tuşa basılmasa bile sürekli hareket eder. Hareket ettirmeden önce her zaman tuş durumunu denetle.



Program tuş basışlarını veya tuş durumlarını denetler ve karakterin konum değerlerini değiştirir. Kare hızı değiştiğinde hızın tutarlı kalması için hareketi geçen zamana bağlamayı unutma. Ne zaman hareket edeceğini bilmek için tuş basışı algılamayı kullan ve doğru miktarı elde etmek için hızı geçen zamanla çarp. Bir oyun karakteri klavyeyle işte böyle hareket eder.