**Marmara Üniversitesi**

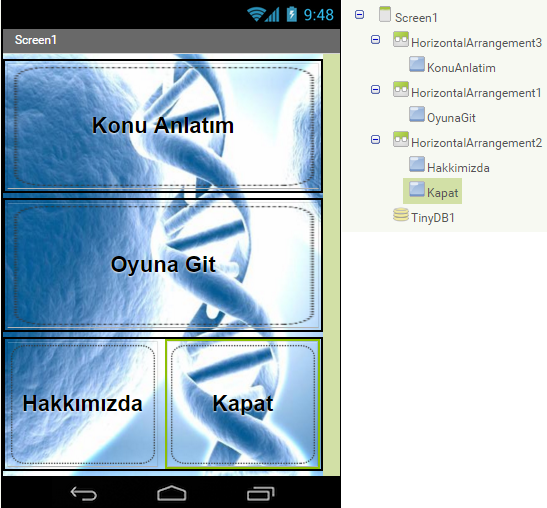
16

**HÜCRE VE YAPISI**

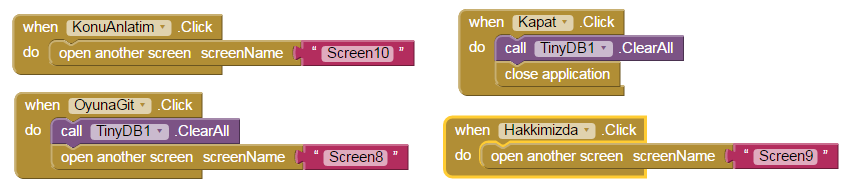
**PROGRAMLAMA II**

**Mert TOKAMA 100212027**

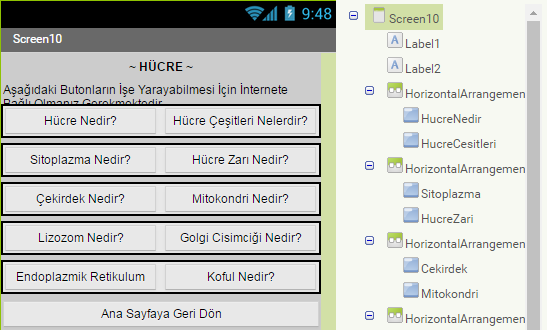
**Anasayfa = Screen1**



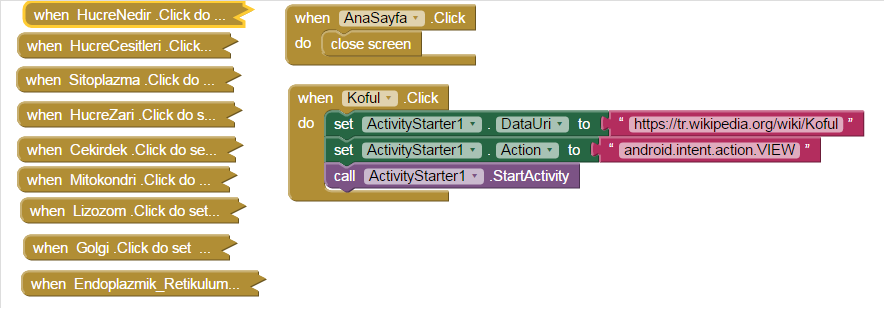
Anasayfada Kapat, Hakkımızda, Oyuna Git ve Konu Anlatım butonları var. Sayfa HorizontalArrangement ile 3 parçaya böldüm. Böylece daha düzgün bir tasarım oldu. Butonların üzerine .PNG uzantılı arka planı transparan, kenarları oval olan dikdörtgenler resim olarak ekledim. Veri Tabanı ekledim. Bunlar Desingner bölümünde olanlar.



Anasayfanın Blocks kısmıda ise yukarıdakiler var. Burada Tüm butonlara tıklanınca yapılacaklar var. İlgili ekranları açma 3 ünde, Kapat butonunda uygulamayı komple kapatma işlemleri gerçekleşecektir. Ayrıca Kapat ve Oyuna Git butonlarına tıklandığında veri tabanını temizleme, sıfırlama işlemi var.

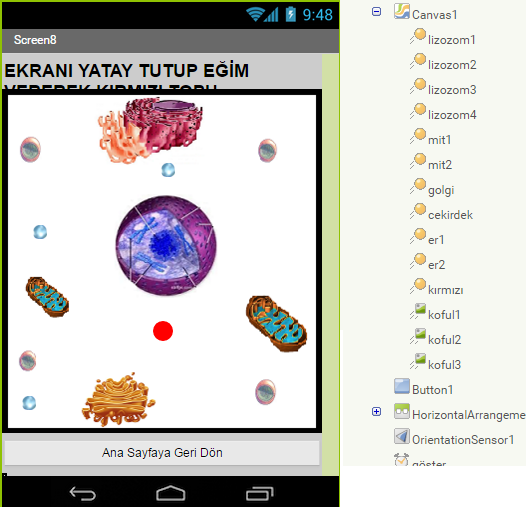
**Konu Anlatım = Screen10**

Hücre ile ilgili terimleri HorizontalArrangement ile tasarımsal olarak düzenledim. Her bir HorizontalArrangement içinde bir sağda bir solda olmak üzere iki buton var.

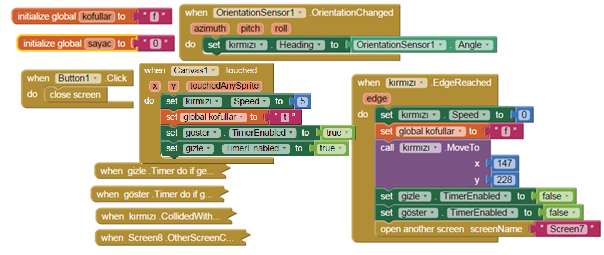


Burada ise bütün butonların tıklandığında yapılacak işlemleri bulunuyor. En alttaki Anasayfa butonunda var olan “Close Screen” açık olan Screen’i kapatıyor. Böylece altta açık olan Anasayda yani Screen1’e geri dönmüş oluyor. Diğer butonların ise içerisinde hemen hemen aynı işlemler gerçekleşiyor. Örnek olarak koful butonuna bakarsak Design bölümünde eklediğimiz “Activity” nin kodlarından 3 işlem alıyoruz. Bunlar “Action” yani android tarafından gerçekleştirilecek işlem, “DataUrl” yani gidelecek sayfa ve “Call StartActivity”dir. Bunların 3’ü yazıldığında butonumuza tıklandığında istenilen URL’ye gidiyor. Bunu <https://www.youtube.com/watch?v=hZP--5OEGB4> adresinden yardım alarak yaptım.

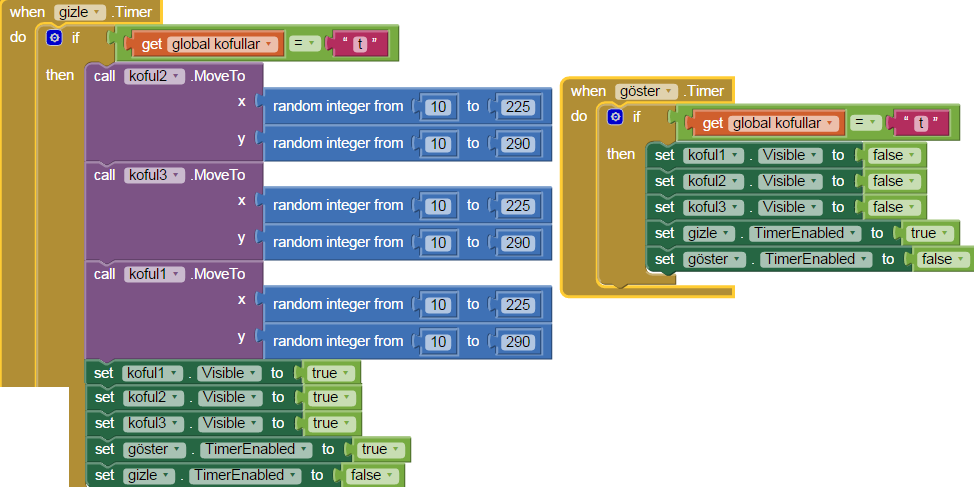
**Oyun Oynama Ekranı = Screen8**



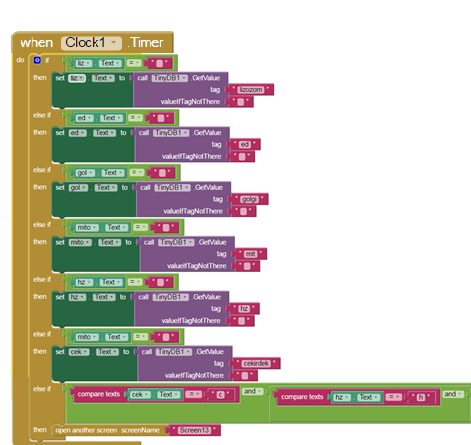
Bu en karışık Screen. Burada anlatması zor ama özet olarak anlatabilirim. Canvas hareket alanı eklendi. Bu alanın arkasına bir adet kendi hazırladığım içerisinde çeşitli hücre organelleri olan bir resim dosyası arkaplan olarak eklendi. Bu resimdeki organellerin üzerine tam denk gelecek şekilde çizim ve animasyon bölümünden alınan topları ekledim. Bu topları kendi topumla çarpışmaları sonrası olacakları belirlemek için kullandım. Hepsinin PaintColor özelliğini None yaparak görünmez yani saydam olmalarını sağladım. 3 tane zamana göre Visible özelliği açılıp kapanan ve belirli alanda zamana göre moveto özelliği gösteren kofullar olarak kullanacağım, yine çizim ve animasyon kısmında bulunan ImageSprite ekledim. Topun hareket özelliği OrientationSensor ekledim bir tane. Zaman ile ilgili ayarlar olduğu için 2 tane Clock var. Veri Tabanı yani “TinyDB” burada da eklenmiş durumda. Bu oyun oynama sayfasını daha detaylı olarak aşağıda anlatıcam.



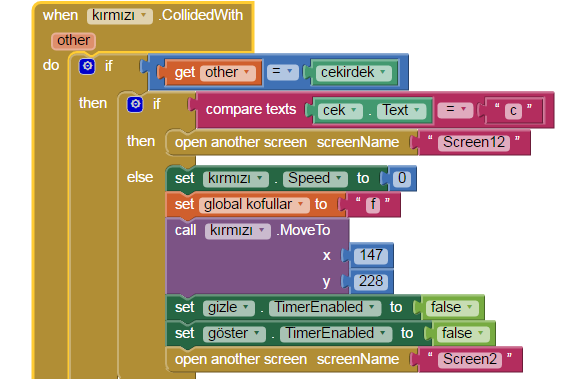
Parça parça anlatmak zorundayım burayı kodlar sığmadığı için. Burada Expand Block özelliği açık olanları inceleyerek başlayalım. 2 adet değişken tanımladık bunlara başlangıç değerleri atadık. When OrientationChanged i OrientationSensor özelliklerinden alıp içerisine kırmızı topun Heading’ine OrientationSensor’ün Angle değerini yazarak topumuzun hareketini sağlıyoruz. Bunu nasıl yapacağımı internetten bulmuştum bu kadarını. Button bir bizim sayfamızı kapatarak arkada açık olan anasayfaya dönmemizi sağlıyor. When EdgeReached özelliği Canvas alanının kenarları için kullandığımız bir özellik. Bunu kırmızı topun özelliklerinden çektim. Burada kırmızı top Canvas alanının kenarlarına değdiğinde ne olacağını tanımlıyoruz. Burada topun Speed değerini sıfırlayıp, konumunu başlangıç konumuna MoveTo ile atayıp, Timer’ları Disable yaptım. Değişkenin içerisine benim için false anlamına gelen f harfi yazdım. Ardından yeni bir Screen açtım. Bu Screen Canvas alanının kenarlarını yani hücre zarını simgeliyor. Bu yüzden Screen7’de Hücre Zarı bölümü var. Canvas özelliklerinden When Touched i aldım, içerisine değişkenimizi t yani true yazdım. Topumuza tekrar Speed değeri verdim. Bu işlemleri çok kez When kırmızı CollidedWith özelliğinde de yaptığımızı söylemem gerek. Bunun sebebi kofullar değişkenini yani rasgele ekranda beliren kofulların ve Timer’ın ve topumuzun durması başlangıç konumuna gelmesini istedim. **Genel olarak istediğim şey Screen8’in oyun sırasında sürekli ön ya da arka planda açık olması. Bunun için Screen8 yani oyun arka plandayken oyunun durması gerek. Oyunu durdurmak için oyun arka plana geçtiğinde yani yeni Screen açıldığında bütün hareketleri durdurmak, oyun tekrar ön plana geldiğinde ve Canvas alanına tıklandığında Timer dahil bütün hareketlerin tekrar başlaması.**

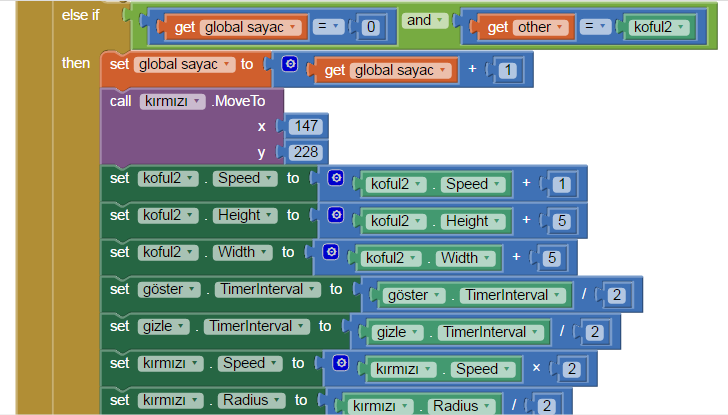


Burada Timer’lar sadece oyundaki kofullar için kullanılıyor. Bir Timer Kofulları gizlemek, diğeri göstermek için kullanılıyor. Gösterilirken kofullar Canvas alanı içerisinde rasgele bir X,Y konumuna MoveTo özelliğiyle gönderdim. Bir Timer 1,5 saniye, diğeri ise 3 saniyede bir çalışmaktadır. Canvas alanına sadece tıkladığımızda kofullar istenilenleri yapsın diye değişken tanımlamıştık. Canvas alanına tıklandığında bu değişken değeri t oluyordu. Burada Timer’lar Canvas alanına dokunulduğunda yani değişken t değeri olduğunda çalışıyorlar.

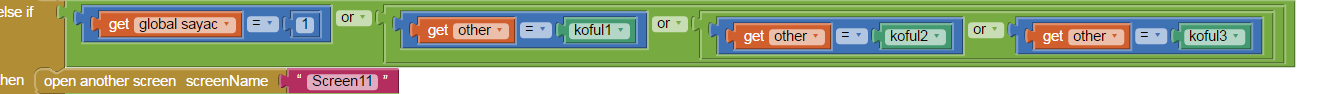


Yukarıda uygulamada bulunduğun haricinde bir saniyenin 4’te 1’inde Screen8’de yani oyun ekranında olacaklar var. Her bir organelin kendi Screen sayfası olduğunu söylemem gerek. Kırmızı topumuz bir başka topa yani organele değdiğinde açılıyordu bu sayfalar. Bu sayfalarda bir soru soruyoruz. Çarptığı organelin resmini ve birkaç özelliğini verip ismini soruyoruz. Cevabı doğruysa veri tabanına bir şeyler yazdırıyoruz. İşte yukarıdaki kod bloğunda veri tabanına yazılanlarla ilgili bir bölüm var. Örnek vererek anlatmak gerekirse, Kırmızı topumuz çekirdeğe değerse Çekirdeğin sayfası açılacak. Bu sayfada eğer biz çekirdeğe çarptığımızı doğru bir şekilde bilebilirsek, Veri tabanında cekirdek isimli bir Tag oluşturuluyor ve bu Tag’ın içerisine ‘c’ harfini yazıyor. Bizde Yukarıdaki kod bloğunda bunu kontrol ediyoruz. Eğer cekirdek Tag’ına c yazıldıysa bunu bir oluşturduğumuz çekirdek label’ine yazdırıyor. Bunu bütün organeller için yaptırıyoruz. Ve labelde eğer daha önceden bir şey yazıldıysa karışıklık olmasın diye tekrar bir şey yazdırmıyor. Bu kontrolü sağlamak için IF’in yardımını alıyor. Yine bu kod bloğunda eğer bütün organellerin içerisine birşeyler yazıldıysa yani **bütün organel isimleri bilindiyse** kazandınız sayfasına yani Screen13 sayfasına yönlendiriyor.



When kırmızı CollidedWith özelliğinde bolca tekrar parçaları var. Sadece top isimleri değişik olan birçok copy paste olduğu için örnekleri sadece anlatıcam. Burada Kırmızı topum sayfam çekirdek topuna dokunursa olacaklar var. Eğer veri tabanından değer çekip içerisine yazdırdığımız cek Label’inin içerisinde c yazıyorsa bu çekirdeğin daha önce kullanıcı tarafından isminin bilindiği anlamına geliyordu. Bu durumda bunu daha önce bildiğini anlatan Screen12 sayfasına yönlendiriyoruz. Eğer daha önce bilinmediyse ise topun hızını 0 yapıp, başlangıç noktasına topumuzu döndürüp, hareketleri kapatıp, çekirdek ile ilgili Screen’i açıyoruz. Burasının yaptığı işlemler bu kadar. Tabi Yeni Screen açtığımız için kofullar değişkenimizi false yani f yapmayı unutmuyoruz. Tüm bunları aynen bütün organeller için tekrarlıyoruz. 

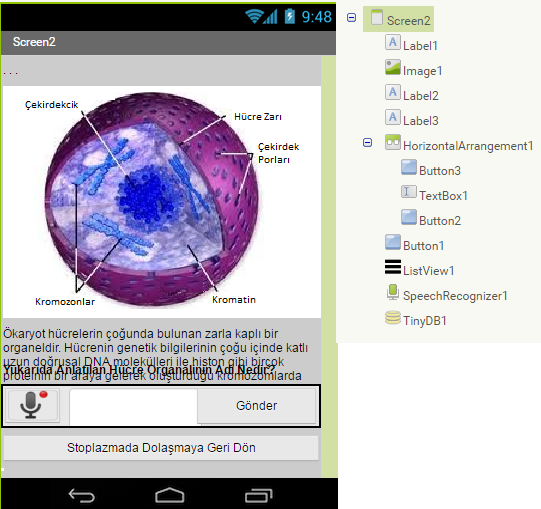
Burada kofullara kırmızı topumuz dokunduğunda olacaklar var. Oyunda başaltım kofullarıda bulunmakta. Bu kofullar topumuza 1 kere dokunursa eğer, topumuzun boyutunu yarıya indiriyor. Topumuzu başlangıç konumuna gönderiyor. Topumuzun hızını 2 katına çıkartıyor. Dokunan kofulun ise yarıçapı artıyor, hızı artıyor ve bütün kofullarının kaybolup tekrar ortaya çıkış süreleri yarıya iniyor. Tüm bu şartlar başta tanımladığımız sayac değişkeni 0 iken bir koful dokunuyorsa oluyor çünkü 2. Kez dokunursa bir koful kırmızı topumuza oyunu kaybediyoruz ve kaybetme ekranında boşaltım kofulu ile ilgili bir bilgilendirme çıkıyor. Bu yüzden kaç kere kofula dokunduğumuzu bilmemiz lazım. Bu yüzden her dokunulduğunda sayaca +1 ekliyoruz.



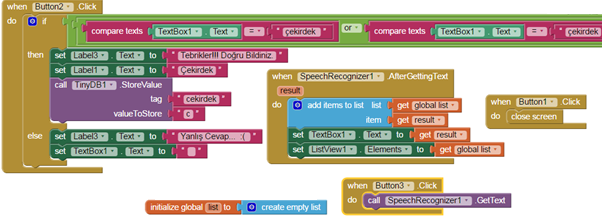
Yukarıda kod bloğunda ise az önce anlattığımız kofulların değmesi durumunda sayac 1 olduktan sonra ikinci değmesi durumunda yani sayac 1 iken değmesi durumunda yapılacaklar var. Kaybettiniz sayfası olan ve boşaltıcı koful ile ilgili bilgi veren sayfaya yönlendiriliyor.

**Hücre Çekirdeği Sayfası = Screen2**

Burada anlatılanlar Screen2, Screen7 ve arasındaki bütün Screen sayfalarında aynıdır. Çünkü sadece organel resimleri ve bilgileri değişmektedir. Geri kalan her şey birebir aynıdır. App Inventor’un Çanta özelliği kullanılarak kodlar Screen’ler arasında taşınmıştır.

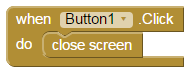


Yukarıda Screen2 sayfasının Design görünüşü bulunmaktadır. Burada en üstteki Label’da doğru bilindiği takdirde organelin adı yazıcak. Daha sonra organelin resmi ve hakkında ufak kısa bir bilgi verilmiştir. Ardından başka bir Label’de soru yazılmıştır. Bu Label’da aynı zamanda soru yanlış bilinirse Yanlış, doğru bilinirse Doğru yazmaktadır. Onun altında 2 Buton ve aralarında bir TextBox var. Soldaki butona ses simgesini yükledim. Sağdakinde ise cevabın doğru yada yanlış olduğu sorgulanıyor. En aşağıda ise oyuna geri dönme butonu var. Tabi ki veri tabanı ekledim buraya da. Bir adet ses kayıt amaçlı SpeechRecognizer, ve sonuçların yazıldığı bir ListView ekledim.



Burada ilk olarak bir List oluşturdum ses özelliği için. Ardından SpeechRecognizer’in GetText’ini Buton3’e tıklandığında çağırdım. Böylece ses simgesi yüklediğim butona tıklandığında çalışacak hale geldi. SpeechRecognizer’in When AfterGettingText özelliğinide <http://appinventorschool.blogspot.com.tr/2015/04/sample-applications-media-components.html> sitesinden bakarak yaptım. Son olarak yazılan yazının doğru mu yanlış mı olduğunu sorgulayan Text kısmında “Gönder” yazan Buton2 Click Eventi var. Burada eğer TextBox’a yazılan yazı benim belirlediğim değerlerden biri değilse (Çekirdek,çekirdek,Çekirdek , çekirdek ) girilen yazının yanlış olduğunu soru sorulan Label’a yazıp, TextBox’ın Text’ini siliyor. Eğer giriken ya da söylenen yazı doğru ise yani çekirdek ise Label1’e yani başlığa çekirdek yazıyor, veritabanında ‘çekirdek’ isimli bir Tag açıp içerisine ‘c’ yazıyor. Bunun sebebini anlatmıştık. Böylece kullanıcının oyun sırasında çekirdeği daha önceden doğru cevaplayıp cevaplamadığını kontrol edebiliyoruz. Veri tabanına yazma işlemini Design tarafında eklediğimiz TinyDB’nin özelliklerinden call StoreValue ile yapıyoruz.

**Hakkımızda = Screen9**

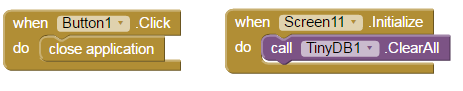
****

En basit sayfamız burası. Bir adet Image, bir adet Label ve bir adet Buton bulunmaktadır. Butonun işlevi de var olan sayfayı kapatmaktır. Böylece Anasayfaya dönülmüş olunur.

**Kaybettiniz Sayfası = Screen11**

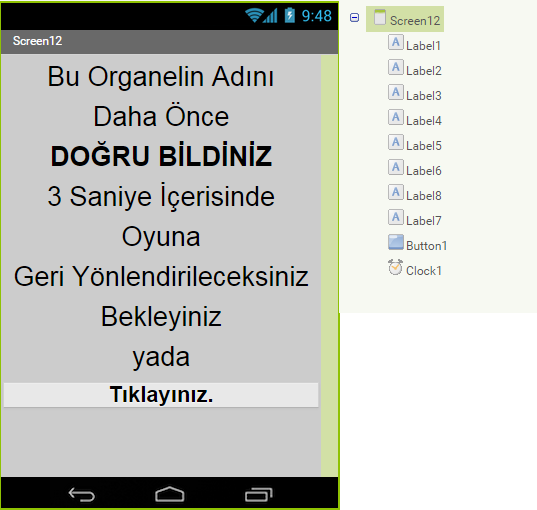


Bu sayfa oyun sırasında etrafta gezinen kofullara 2 defa değerseniz açılan sayfa. Sayfada 2 Label, 1 Image, birde TinyDB yani veritabanı bulunmaktadır. Bu sayfa açıldığında oyunu kaybetmiş olduğunuzdan veri tabanı temizlenir ve oyun kapatılır.

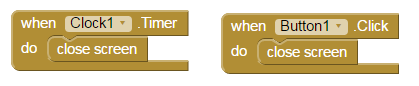


Button1’e tıklandığında uygulamayı kapatırken Screen11 açıldığı anda tüm veri tabanı silinmektedir.

**Daha Önce Bildiniz Sayfası = Screen12**



Bu sayfa daha önce ismini doğru bildiğiniz bir organelin üzerine tekrar geldiğiniz karşınıza çıkan sayfadır. Yapısı çok basittir. Alt alta yazabilmek için 7 adet Label kullanılmıştır. Oyuna dönmek için bir Buton ve Butona basmadan dönmek için bir adet Clock vardır.

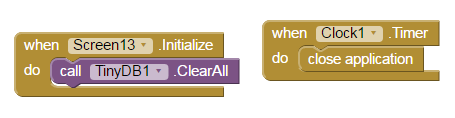


Burada görüldüğü üzere Clock’ta kurulu olan zaman dolduğunda ya da Buton’a tıklandığında Screen kapanıyor. Böylece oyuna geri dönebiliyoruz. Clock zamanı 3 saniyeye kurulu durumdadır.

**Kazandınız Sayfası = Screen13**



Bu sayfa oyunda tüm organellerin adını bildiğinizde açılan sayfadır. Bir adet TinyDB yani veritabanı ve bir adet Clock vardır.



Sayfa açıldığında veri tabanı temizlenir ve saat zamanı dolduğunda uygulama kapatılır.