



# VFabrika

## Web İçerik Geliştirme Platformu

### VFABRİKA PROGRAMI KULLANIM KILAVUZU

---

Bu doküman VFabrika programının nasıl kullanılacağını anlatır. VFabrika, Sebit Eğitim ve Bilgi Teknolojileri A.Ş. bünyesinde üretilen bir Windows uygulamasıdır.

## İçindekiler

Giriş .....	5
Açılış Ekranı .....	6
Yeni Proje Oluşturma .....	6
VFabrika Sürüm Bilgisi ve Güncellemeler.....	8
VFabrika Geliştirme Ortamı .....	9
Dizayn ve Bloklar.....	9
Timeline (Zaman Çizelgesi) .....	10
Proje Öğeleri ve Hazır Materyaller .....	13
Sahne Nesneleri.....	13
Özellikler ve Hizalama.....	14
Araç Çubuğu.....	14
Interaction (Etkileşim).....	15
BÖLÜM 1 - VFABRIKA İLE KODLAMAYA GİRİŞ.....	16
Değişkenler .....	16
Variable (Değişken) nedir? .....	16
Veri Tipleri.....	16
Değişken Kapsama Alanı.....	17
Get – Set Erişimcileri.....	17
Operatörler.....	19
Operatör nedir? .....	19
Aritmetik Operatörler .....	19
Mantıksal Operatörler .....	19
Karşılaştırma Operatörleri .....	21
Karar Kontrol Yapıları .....	24
If Then (Eğer ise) .....	24
If Then Else (Eğer ise Değilse) .....	24
If Then Else Return (Eğer ise Değilse Döndür) .....	25
Switch Case (Değişken Durum).....	25
Döngüler (Loops) .....	28
Foreach Döngüsü (Her Eleman İçin Döngüsü) .....	28

While Döngüsü (Sürekli Yap Döngüsü) .....	29
Do While Döngüsü (Yap ve Doğruysa Sürekli Yap Döngüsü) .....	29
Break (Durdur) .....	30
Continue (Devam Et).....	30
Listeler .....	33
Create List (Liste Oluşturma) .....	33
Create From Comma Text (Virgülle Ayrılmış Metinden Liste Oluştur).....	34
Add Item (Öge Ekleme).....	34
Add Item Into (Sıraya Öge Ekle).....	34
Remove Item (Öge Çıkarma) .....	35
Remove Item At (Pozisyonadaki Ögeyi Sil) .....	35
Length (Liste Uzunluğu) .....	35
Contains (Mevcut) .....	35
Get Item At (Pozisyonadaki Ögeyi Oku) .....	36
Index Of (Pozisyon Bul).....	36
Set Item At (Pozisyonadaki Ögeyi Değiştir) .....	36
Shuffle List (Listeyi Karıştır) .....	36
Prosedürler .....	38
Prosedür Nedir? .....	38
Procedure (Prosedür Oluşturma) .....	38
Call (Çağırma).....	38
Return Result (Sonuç Çağırma).....	39
Call for Result (Sonuç Döndüren Prosedür Çağırma) .....	39
String (Metin) .....	41
Compare Text (Karşılaştır) .....	41
Contains (İçeriyor mu) .....	41
Is Empty (Boş mu) .....	42
Join (Ekle).....	42
Index of Text (Metnin Konumunu Bul) .....	42
Length (Uzunluk).....	42
Replace All (Hepsini Değiştir).....	42

Segment (Böl) .....	43
Starts At (Başlangıç Pozisyonunu Bul) .....	43
Trim (Boşlukları Temizle) .....	43
To Text (Metne Çevir) .....	43
Concat (Birleştir) .....	44
Split (Ayır) .....	44
<b>BÖLÜM 2 - FORM NESNELERİ</b> .....	<b>45</b>
Button (Buton).....	45
Image Button (Resim Butonu).....	46
Radio Button (Seçenek Butonu) .....	47
CheckBox (Kontrol Butonu) .....	47
Dropdown List (Açılan Kutu) .....	48
Textbox (Metin Kutusu).....	49
Slider (Sürgü) .....	50
<b>BÖLÜM 3 – MEDYA NESNELERİ</b> .....	<b>51</b>
Image (Resim).....	51
Sound (Ses) .....	52
Video .....	52
<b>BÖLÜM 4 – TASARIM NESNELERİ</b> .....	<b>54</b>
Canvas (Resim Kâğıdı).....	54
Content (İçerik).....	58
Table (Tablo).....	59
Highlight (Vurgu) .....	61
Primitive Objects (Temel Nesneler) .....	61
Smart Objects (Akıllı Nesneler) .....	62
Parallax Scroll (Paralaks Kaydırma).....	62
Polygon (Çokgen).....	63
<b>BÖLÜM 5 – ARAÇLAR</b> .....	<b>65</b>
Delay (Erteleyici) .....	65
Timer (Zamanlayıcı) .....	65
Trigger (Tetikleyici).....	66

BÖLÜM 6 – SEBİT FRAMEWORK .....	67
Sebit Framework .....	67
Feedbacks (Bildirimler) .....	68
Questions (Sorular) .....	69
Skip Intro (Tanıtımı Geç) .....	70
Animation Popup (Animasyon Penceresi) .....	71
Hypothesis (Hipotezler) .....	72
BÖLÜM 7 – HAZIR KÜTÜPHANE FONKSİYONLARI .....	74
Math (Matematik) .....	74
Math Advanced (İleri Matematik) .....	75
Trigonometry (Trigonometri) .....	76
Json Data Set .....	79
BÖLÜM 8 – DİĞER ARAÇLAR .....	81
General .....	81

## Giriş

VFabrika, öğretmenlerin farklı becerilere sahip öğrenci gruplarıyla birlikte, kolaylıkla özgün etkileşimli içerik üretmelerini sağlayan “HTML5 İçerik Geliştirme Platformu”dur.

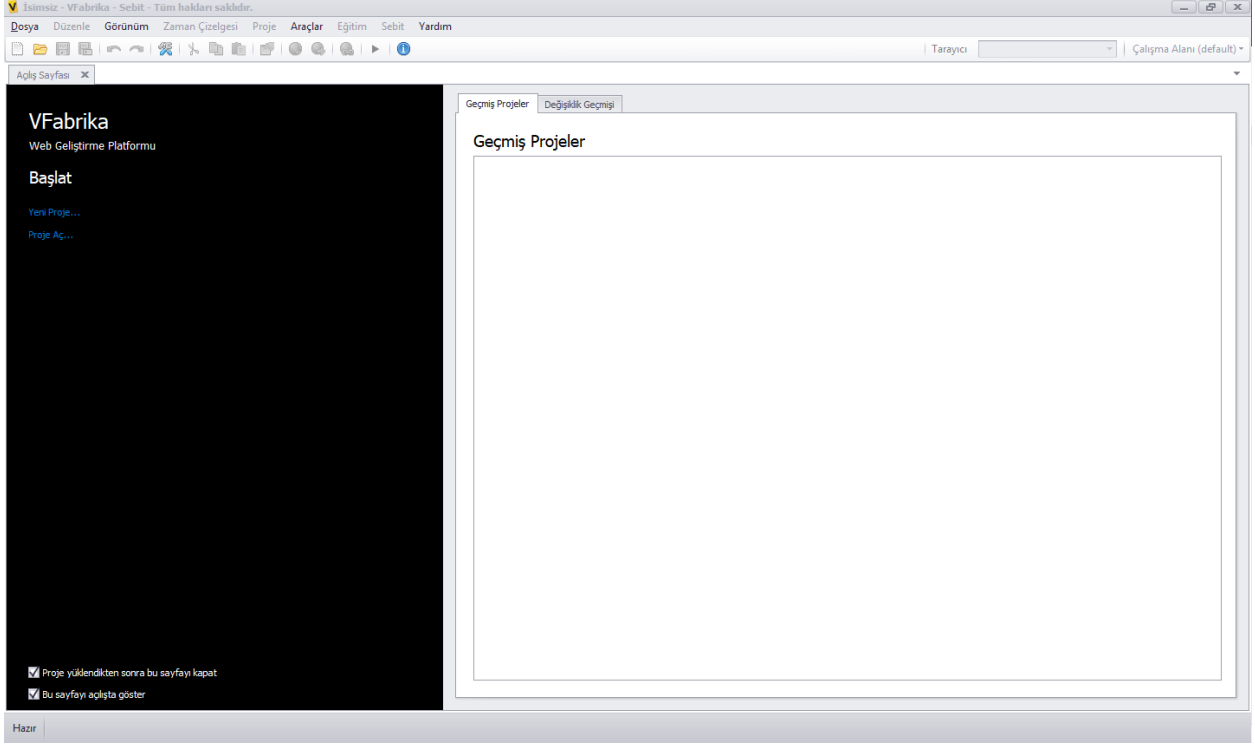
Öğretmenlerin ve öğrenci gruplarının alıştırmaya, deney, canlandırma, eğitsel oyun gibi öğrenme materyallerini kod yazmadan, herhangi bir yardım almadan, nitelikli bir şekilde üretmeleri için tasarlanmıştır.

VFabrika’yla öğretmenlerin liderliğindeki öğrenciler, üretilecek e-içerik için planladıkları çalışmaları, kendi yetenek ve ilgileri doğrultusunda tüm süreçlere katkıda bulunarak iş birliği içerisinde tamamlarlar.

VFabrika, üretim sürecinde öğretmenlere ve öğrencilere sistematik kurgu, tasarım, planlama, algoritmik düşünme ve iş birliği içinde çalışma becerileri kazandırır.

## Açılış Ekranı

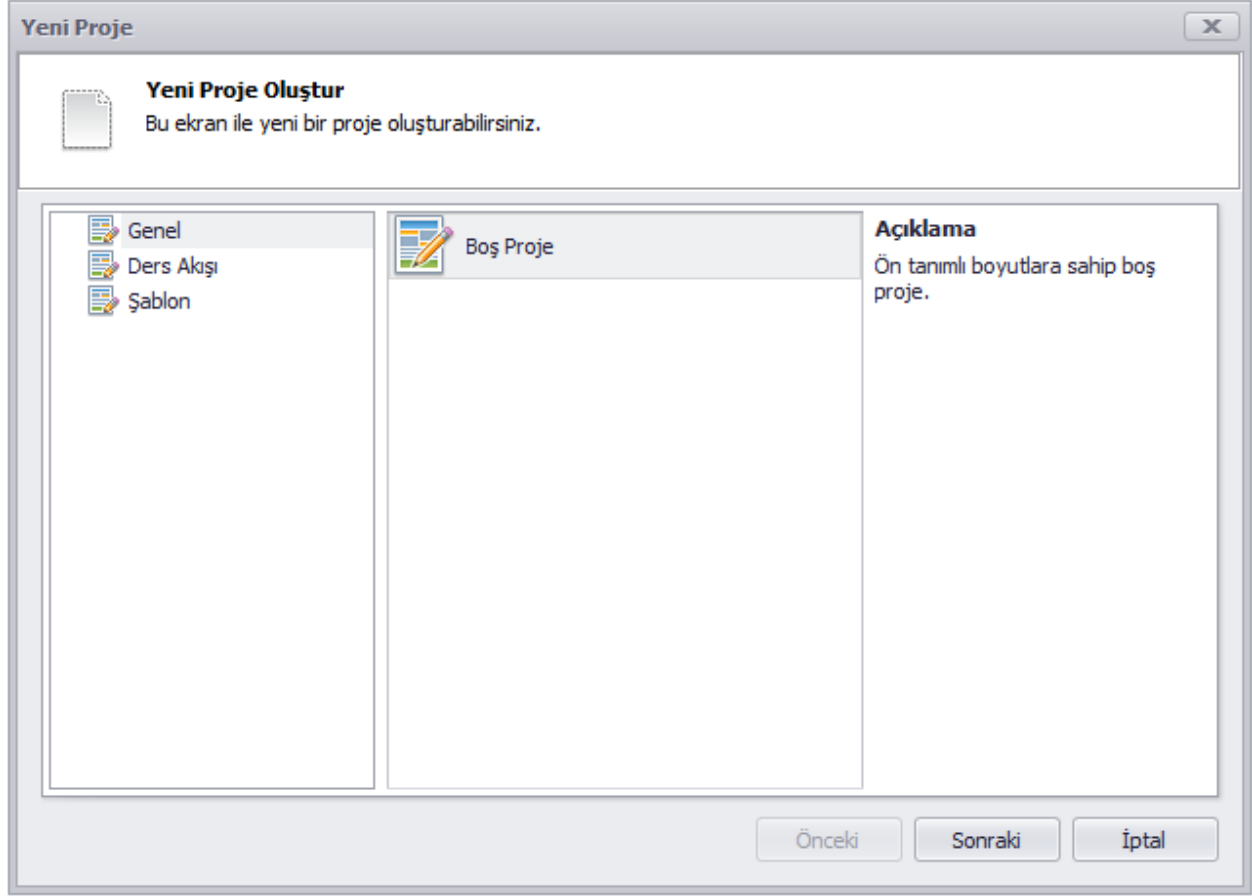
VFabrika'yı başlattığımızda karşımıza ilk olarak “Geçmiş Projeler” ve “Değişiklik Geçmişi” sekmeleri gelir. Geçmiş projeler sekmesinde, daha önce çalıştığımız projeler listelenir. Bu listeden bir projeye tıklayarak kaldığımız yerden devam edebiliriz. Değişiklik Geçmişi penceresinde de VFabrika güncellemeleriyle yapılan değişikliklerin neler olduğu kısaca belirtilir.



Şekil 1: Açılış Ekranı

## Yeni Proje Oluşturma

Yeni bir proje oluşturmak için, Açılış Ekranında “Yeni Proje...” linkine tıklayarak veya Dosya > Yeni Proje adımını takip etmemiz yeterli olur. Bunlara ek olarak, “CTRL + N” kısayolu ile de yeni bir proje oluşturabiliriz.

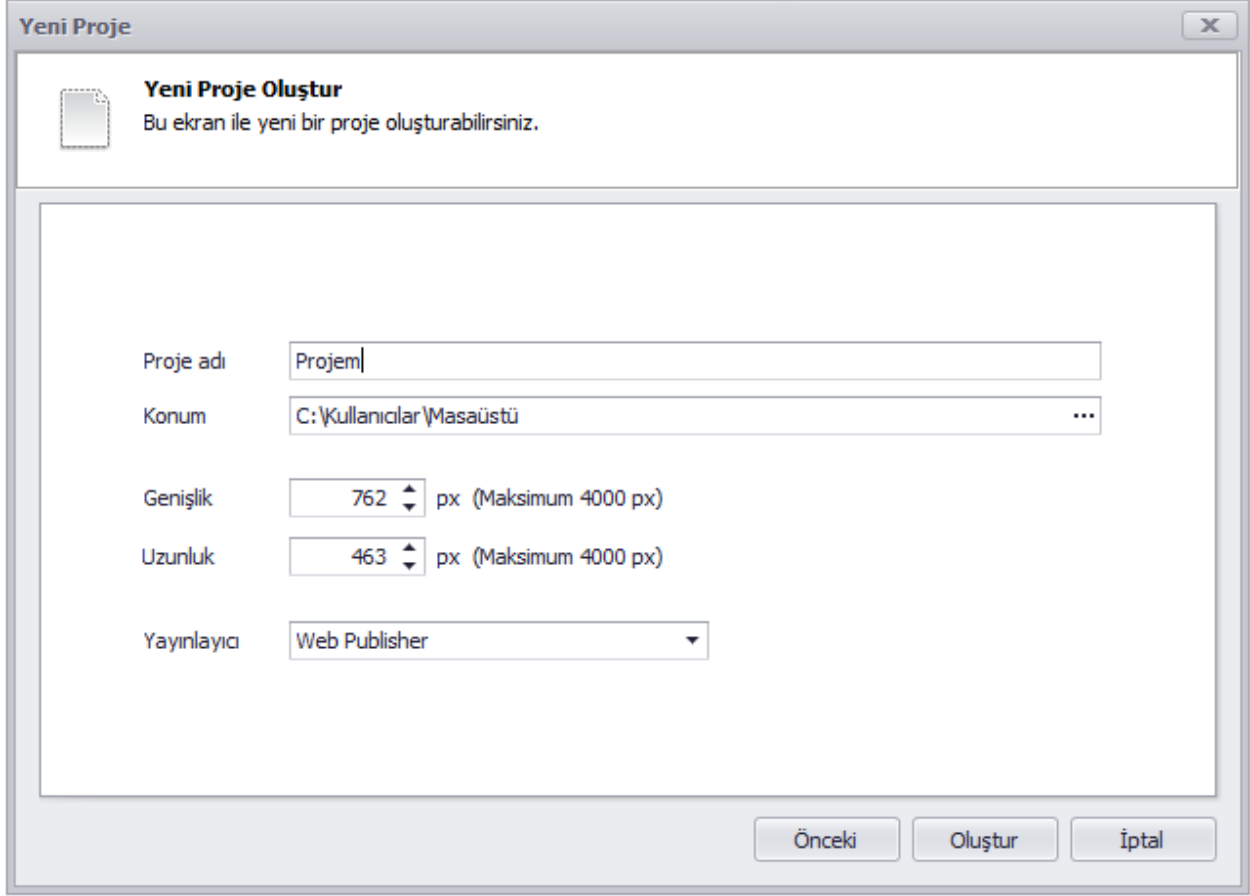


Şekil 2: Yeni Proje Penceresi

Açılan “Yeni Proje” penceresinde Boş bir genel proje seçebileceğimiz gibi ön tanımlı projelerden birini de seçebiliriz. Ders Akışı projelerinde İnteraktif Etkinlik veya Etkileşimli Materyal yer alır. İnteraktif Etkinlik projesinde tüm ekranda açılan interaktif bir proje oluşturabiliriz. Etkileşimli Materyal projesinde öğrenme adımı içerisinde gömülü çalışan etkileşimli materyal projesi oluşturabiliriz. Şablonda ise ön tanımlı bir Sürükle/Bırak projesi oluşturulabilir.

Proje şeklimizi seçip sonraki adıma geçtiğimizde karşımıza oluşturacağımız proje ile ilgili küçük değişiklikler yapabileceğimiz bir ekran gelir. Bu ekranda projenin adını, konumunu, boyutlarını ve yayınlayıcısını belirleyerek yeni projemizi oluşturabiliriz.





Şekil 3: Yeni Proje Penceresi

## VFabrika Sürüm Bilgisi ve Güncellemeler



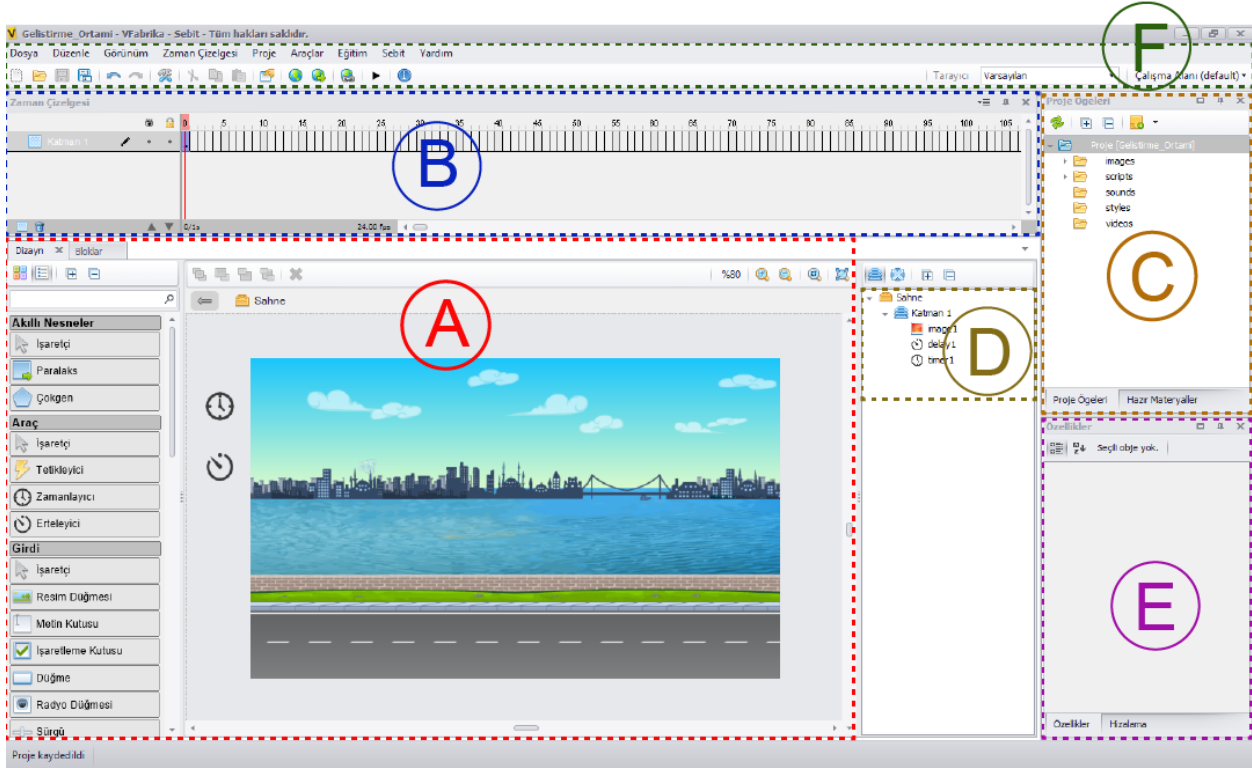
Şekil 4:Hakkında Penceresi

Kullandığımız VFabrika sürüm bilgisini Yardım > Hakkında yolunu izleyerek karşımıza çıkan pencereden öğrenebiliriz. Bu pencerede VFabrika sürümünü, Donanım Anahtarını ve Lisans bilgilerini öğrenebiliriz.

VFabrika'nın sürüm güncellemesini kontrol etmek için de Yardım > Güncellemeleri Kontrol Et yolunu kullanabiliriz.

## VFabrika Geliştirme Ortamı

VFabrika’da projelerimizi geliştirme ortamında oluştururuz. VFabrika geliştirme ortamı çok kullanışlı ve anlaşılabilir şekilde tasarlanmıştır. Temel olarak 6 bölümden oluşmaktadır. Bunlar; Dizayn ve Bloklar, Zaman Çizelgesi, Proje Öğeleri ve Hazır Materyaller, Sahne Nesneleri, Özellikler ve Hizalama, Araç Çubuğu. Bu bölümler sayesinde VFabrika’yı daha verimli kullanabilirsiniz.



Şekil 5: Geliştirme Ortamı

*Varsayılan Geliştirme Ortamı: A. Dizayn ve Bloklar B. Timeline (Zaman Çizelgesi) C. Proje Öğeleri ve Hazır Materyaller D. Sahne Nesneleri E. Özellikler ve Hizalama F. Araç Çubuğu*

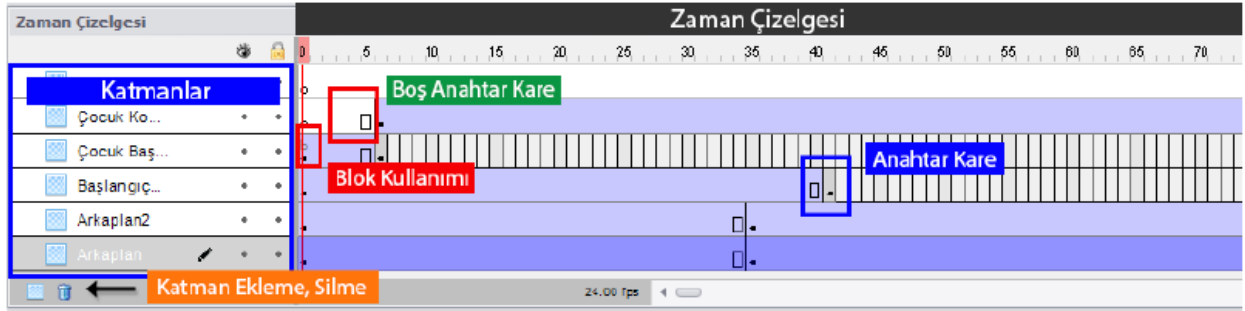
### Dizayn ve Bloklar

Dizayn çalışma alanı, sahne nesnelerini projeye yerleştirebilmemizi ve görsel ayarlama yapmamızı sağlar. Dizayn araç çubuğundaki nesneleri sürükleyip bırakarak sahneye ekleyebiliriz. Sahne sınırlarının dışında olan görseller projeyi çalıştırdığımızda ekranda görünmez.

Blok çalışma alanında projemizi kodlayabilir, sahneye eklediğimiz nesnelerin özelliklerini bloklar yardımıyla değiştirebilir ve çeşitli algoritmalar oluşturabiliriz.

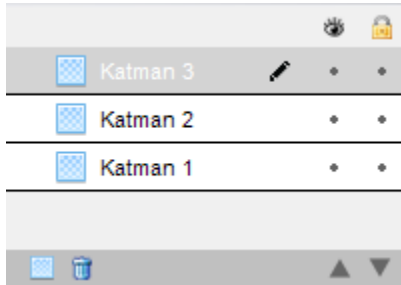
## Timeline (Zaman Çizelgesi)

VFabrika’da Timeline (Zaman çizelgesi); frame (kare) ve layer’lardan (katman) oluşan ve bunları kontrol eden bir yapıdır. Animasyonları kareler aracılığı ile zaman çizelgesinde oluşturabiliriz. VFabrika’da süre uzunlukları karelere bölünür ve kare sayısına göre animasyonların ne kadar hızlı veya yavaş olacağını ayarlanabilir. Projede eğer blok kullanılmışsa kullanılan katmanın zaman çizelgesine “b” işareti gelir. Bu işaret, o katmanda blokların da olduğunu gösterir.



Şekil 6: Zaman Çizelgesi

## Layer (Katman)



Şekil 7: Katman

Katman, projedeki nesneleri gruplandırmamızı sağlayan ve animasyonları daha kolay getiren bir zaman çizelgesi bileşenidir. Katmanlarda sıralama önemlidir. En üstte yer alan katman sahnenin en önünde bulunurken en alttaki katman da sahnenin en gerisinde bulunur.

Katman panelinden kolaylıkla yeni katman ekleyebilir, var olan katmanı silebilir, katmanların yerini değiştirebiliriz. Ayrıca katmanları kilitleyebilir veya sahnedeki görünürliğini kapatabiliriz.

Bir katmanı kilitlediğimizde, katmandaki nesneleri de kilitlemiş oluruz. Böylece o katmandaki nesneleri sahnede seçemeyiz veya yerini değiştiremeyiz. Örneğin, projemizde bir arka plan olsun ve onu da “arka plan” katmanına koymuş olalım. Arka planı ekledikten ve ayarladıktan sonra çalışmamız bitince onu kilitlemek bizim için faydalı olabilir. Böylece yerini değiştirmek, silmek gibi yanlışlıkla yapılabilecek hatalardan da kaçınmış oluruz.

Kilit düğmesinin yanında bulunan göze tıklayarak seçtiğimiz bir katmanı gizleyebiliriz. Gizlediğimiz katmanda bulunan nesneler sahnede görünmez.

### Frame (Kare)

VFabrika’da süre uzunlukları kare olarak oluşturulur. Varsayılan olarak 1 saniyedeki kare sayısı 24’tür. Ancak bunu değiştirmek mümkündür. Bunun için, araç çubuğundaki Proje > Özellikler yoluyla karşımıza çıkan pencereden yararlanabiliriz. Animasyonları da bu karelerle oluşturabiliriz. Kare eklemek için F5 kısa yolunu kullanabiliriz. Sahnedeki tüm nesneler eklediğimiz karede de aynı şekilde yer alır.

### Keyframe (Anahtar Kare)

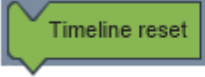
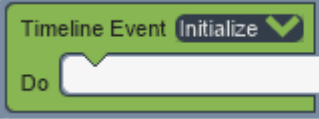
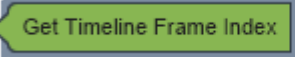
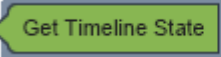
Anahtar kare, sahnede veya zaman çizelgesinde bir değişikliğin meydana geleceği karelerdir. Örneğin, yolda giden bir araba animasyonu yapmak istediğimizde, arabanın başlangıç noktası ilk kare olurken arabanın sahnede en son bulunacağı 20. kareye bir anahtar kare ekleriz ve sahnemizdeki araba nesnesini de en son duracağı yere taşırız. Ardından, 1 ile 20. kare arasında herhangi bir yerde sağ tıklayarak “Hareket Oluştur” dediğimizde arabanın belirtilen doğrultuda hareket ettiğini göreceğiz. Anahtar kare eklemek için F6 kısayolunu kullanabiliriz.

### Blank Keyframe (Boş Anahtar Kare)

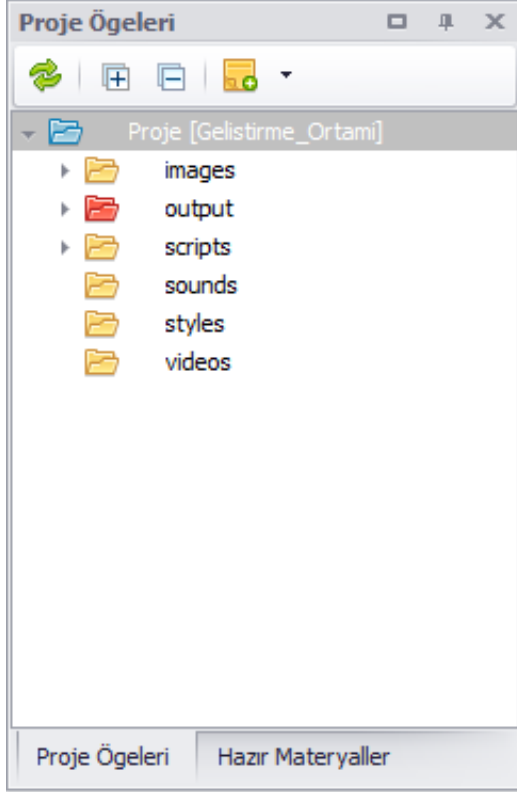
Boş anahtar kare, eklediğimiz karede sahneyi boş bırakmak için kullanılır. Anahtar kare eklediğimizde, sahnedeki tüm nesneler aynı şekilde eklendiği kareye taşınır fakat boş anahtar karede sahne hiçbir nesne yer almaz. Boş anahtar kare eklemek için F7 kısayolunu kullanabiliriz.

### Timeline (Zaman Çizelgesi) Blokları

 <b>Timeline Player Control</b>	Timeline Player Control (Zaman Çizelgesi Kontrolü) bloğu, zaman çizelgesini başlatma veya durdurma komutları vermemizi sağlar.
 <b>Timeline Player Control Go To</b>	Timeline Player Control Go To (Zaman Çizelgesi Kontrol ve Kare Değişimi) bloğu, zaman çizelgesinde belirtilen kareye gidilmesini ve o karedeyken oynatma veya durdurma komutlarını vermemizi sağlar.
 <b>Timeline Set Fps</b>	Timeline Set Fps (Saniyedeki Kare Sayısını Değiştir) bloğu, zaman çizelgesinde saniyedeki kare sayısını değiştirmemizi sağlar.

 <b>Timeline Reset</b>	<p>Timeline Reset (Zaman Çizelgesini Sıfırla) bloğu, zaman çizelgesini sıfırlamamızı sağlar.</p>
 <b>Timeline Event</b>	<p>Timeline Event (Zaman Çizelgesi Olayı) bloğu, zaman çizelgesi başladığında (initialize), kare değiştiğinde (frame change) veya durum değiştiğinde (state change) neler yapılacağını belirlememizi sağlar.</p>
 <b>Get Frame Index</b>	<p>Get Frame Index (Kare İndeksini Al) bloğu, zaman çizelgesinde hangi karede yer aldığımızı öğrenmemizi sağlar.</p>
 <b>Get Timeline State</b>	<p>Get Timeline State (Zaman Çizelgesi Durumunu Al) bloğu, zaman çizelgesi oynuyorsa True (Doğru) oynamıyorsa False (Yanlış) değerini döndürür.</p>

## Proje Öğeleri ve Hazır Materyaller



Şekil 8: Proje Öğeleri ve Hazır Materyaller

Proje öğeleri paneli aracılığıyla projenin dosya sistemine ulaşabiliriz. Projemizde yer alan tüm klasörler burada sıralanır. Varsayılan olarak images, output, scripts, sounds, styles, videos klasörleri yer alır. Ekle butonu ile yeni komut dizisi, klasör veya yeni stil ekleyebilir.

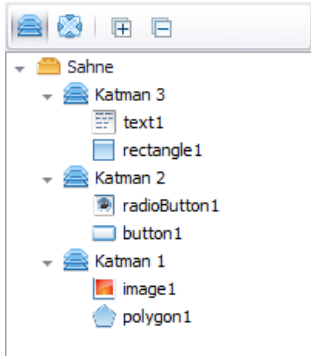
**Dikkat** edilmesi gereken nokta; output klasörünün içine dosya veya klasör eklenmemesi gerektiğidir. Bunun sebebi, publish (F12) alındığında output klasörünün içindeki her şey sıfırlanmakta ve baştan oluşturulmaktadır. Dolayısıyla, output klasörüne koyacağımız dosya veya klasörler publish aldığımız an silinecektir. Output klasörünün **kırmızı** renkte olma sebebi de budur.

Dışarıdan bir dosya eklediğimizde proje öğeleri panelinde görünmesi için yenile butonuna basılmalıdır.

Hazır Materyaller bölümünden de proje için kullanabileceğimiz resimlere ulaşabiliriz. Bunun için,

arama kısmındaki metin kutusuna anahtar kelimeleri yazdıktan sonra çıkan sonuçlardan dilediğimizi sürükleyip sahneye ekleyebiliriz.

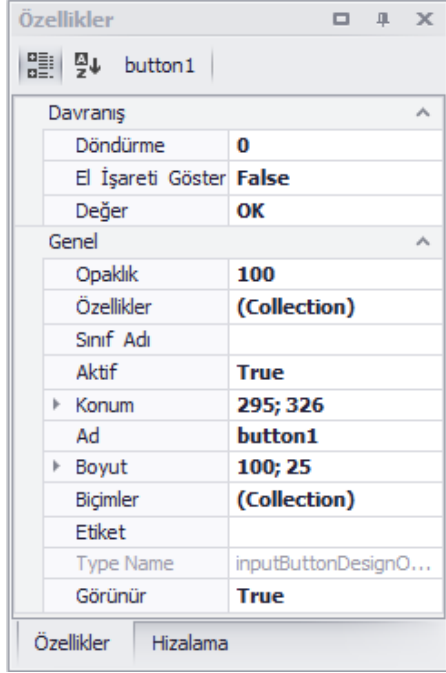
## Sahne Nesneleri



Şekil 9: Sahne Nesneleri

Sahne kullanılan nesneleri hiyerarşik yapısına göre bu panelde bulabiliriz. Katman sırası değiştiğinde bu panelde de katman sırası değişmektedir. Kullanılan tüm nesnelere ulaşabilir ve hiyerarşik sırasını bu panelle değiştirebiliriz. Sahne bir nesne seçildiğinde, eğer nesne görünür durumdaysa ekranda seçili hâle gelir ve nesnenin içinde bulunduğu Layer ve Keyframe otomatik olarak seçilir.

## Özellikler ve Hizalama



Şekil 10: Özellikler ve Hizalama

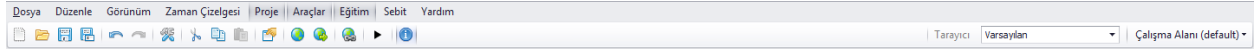
VFabrika’da her nesnenin kendine ait birtakım özellikleri vardır. Bu özelliklere ulaşmak için, sahne panelinden veya sahnedeyken nesneye tıklamamız yeterlidir. Böylece o nesneye ait özellikleri bu panelde bulabilir ve değiştirebiliriz.

Genel özellikler başlığı her nesnede bulunan; opaklık, konum, ad, boyut, görünür, aktif gibi özellikleri içerir.

**NOT:** Biçimler kısmıyla, seçili nesne için CSS özellikleri ekleyebiliriz.

Hizalama panelinde hizala, dağıt, boyut eşitle gibi özellikler bulunur. Bu özelliklerle seçili nesnenin hizalama işlemlerini yapabiliriz. Ayrıca, sahneye göre hizala seçeneği ile otomatik olarak hizalanmasını da sağlayabiliriz.

## Araç Çubuğu



Şekil 11:Araç Çubuğu


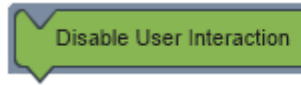

VFabrika programında kullanılan temel araçları barındıran paneldir. Panelde şu araçlar bulunur: Yeni Proje, Proje Aç, Projeyi Kaydet, Projeyi Farklı Kaydet, Geri Al, İleri Al, Tercihler, Kopyala, Kes, Yapıştır, Proje Ayarları, Yayınla, Yayınla ve Çalıştır, Yayın Ayarları, Zaman Çizelgesini Oynat, Hakkında.

Tarayıcı kısmıyla projeyi çalıştırdığımız zaman hangi tarayıcının kullanılacağını seçebiliriz. Varsayılan olarak seçiliyse bu durumda bilgisayarın varsayılan tarayıcısı kullanılır.


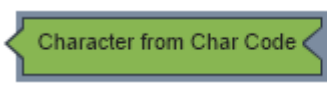
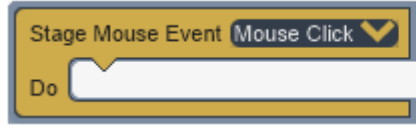

En sağda yer alan Çalışma Alanı bölümünde programın arayüzünü düzenleyebilir, düzenlediğimiz arayüzü kaydedebilir ve daha önce kaydettiğimiz arayüzleri yükleyebiliriz. Çalışma Alanı menüsünden “default” seçeneğine tıkladığımızda, program arayüzünü varsayılan hâline geri döndürür.

## Interaction (Etkileşim)

VFabrika’da oluşturduğumuz içerikler yalnızca tasarım ve sunumdan meydana geliyorsa bu projeler statik (durağan) olarak nitelendirilir. Statik projelerde kullanıcı projeyi kullanırken yalnızca izleyicidir. Herhangi bir şekilde projeye veya içeriğine müdahale edemez. Ancak, etkileşim nesnelerini kullanarak oluşturacağımız projeyi dinamik hâle getirebiliriz. Böylelikle kullanıcı ile etkileşimli bir proje oluşturulabilir.

 <b>Enable User Interaction</b>	Enable User Interaction (Kullanıcı Etkileşimini Etkinleştir) bloğu, kullanıcı etkileşiminin aktif hale getirilmesini sağlar.
 <b>Disable User Interaction</b>	Disable User Interaction (Kullanıcı Etkileşimini Devre Dışı Bırak) bloğu, kullanıcı etkileşimi aktif ise devre dışı bırakarak etkileşimi kapatır.
 <b>On Input Key Event</b>	On Input Key Event (Tuş Olayı) bloğu, kullanıcı klavyeden bir tuşa bastığında neler yapılacağını belirlememizi sağlar.

**NOT:** Tuş olayında Key Up, Key Down, Key Press olmak üzere üç seçenek vardır. Key Up, tuşa basma işlemi sonlandığında; Key Down, tuş basılı iken ve Key Press ise tuşa basıldığında çalışır.

 <b>Open Html Link</b>	Open Html Link (Html Link Aç) bloğu, girilen web sayfası linkinin açılmasını sağlar.
 <b>Character from Char Code</b>	Character from Char Code (Char Code’dan Karakter Getir) bloğuna karakter kodunu eklediğimizde, eklenen karakter kodunun aktif olması sağlanır.
 <b>On Stage Mouse Event</b>	On Stage Mouse Event (Sahne Fare Olayı) bloğu, fareye tıklandığında neler yapılacağını belirlememizi sağlar.
 <b>Is Key Pressed</b>	Is Key Pressed (Tuşa Basıldı mı) bloğu, eklenen karakter kodunun basılıp basılmadığını kontrol eder.



## BÖLÜM 1 - VFABRİKA İLE KODLAMAYA GİRİŞ

### Değişkenler

#### Variable (Değişken) nedir?

Verilerin, bilgisayarın hafızasında tutulduğu alana verilen isimdir. Değişkenleri bir kutu gibi düşünebiliriz. İçine bir değer ekleyebilir, değeri değiştirebilir veya kaldırabiliriz. Değişkenler için bilgisayarın geçici hafızasında (RAM) bir yer ayrılır. Değişkenlere atadığımız isimler sayesinde onları hafızadan çağırabilir ve üzerinde işlemler yapabiliriz.

VFabrika’da bir değişken tanımlayalım;

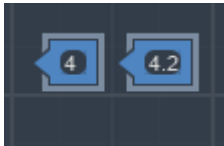


Şekil 12

Bilgisayar hafızasında *degiskenim* adlı bir kutu oluşturduk ve içine de 5 değerini koymuş olduk. Değişkenin değerine ulaşmak için ona atadığımız isim kullanılır.

#### Veri Tipleri

VFabrika’da temel olarak üç çeşit veri tipi vardır. Bunlar; sayı, metin ve boolean.



Şekil 13

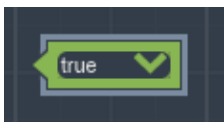
**Sayı** veri tipinde, tam sayı ve ondalık sayı değerleri girilebilir.

Tam sayı değerine ondalık bir değer giremeyiz. Ondalık girdiğimiz değer, otomatik olarak tam sayıya dönüşür.



Şekil 14

**String (Metin)** veri tipinde metin, cümle, harf değerleri girilebilir.



Şekil 15

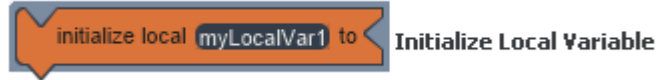
**Boolean** ikili veri tipidir ve yalnızca iki değer bulunur. Bunlar doğru ve yanlıştır.

Bazı programlama dillerinde değişken tanımlamadan önce veri tipini belirtmek zorunludur. VFabrika’da değişken tanımlarken veri tipini belirtmek gerekmemekte, program bunu otomatik olarak algılayabilmektedir.

## Değişken Kapsama Alanı

Kapsama alanı, tanımlanan değişkenin programın hangi bölümünde çalışabileceğini belirler. VFabrika iki farklı kapsama alanına sahiptir. Bunlar Yerel (Local) ve Genel (Global) kapsama alanlarıdır.

**Local (Yerel) Kapsama Alanı:** Yerel tanımlanmış değişkenler yalnız kullanıldığı blok veya fonksiyon içinde tanınabilir ve kullanılabilirler. Programın tamamında kullanılamazlar.



Şekil 16

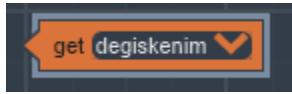
**Global (Genel) Kapsama Alanı:** Genel tanımlanan değişkenler programın her yerinde tanınabilir ve kullanılabilirler.



Şekil 17

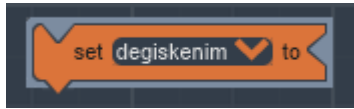
## Get – Set Erişimcileri

Get (Değişken Oku) ve Set (Değişken Ayarla) erişimcileri, değişken tanımladıktan sonra değişkene ulaşmamızı ve üzerinde değişiklik yapmamızı sağlar. Dolayısıyla, değişken ile ilgili tüm işlemleri bu erişimciler aracılığıyla yapabilmekteyiz.



Şekil 18

**Get** erişimcisi ile tanımlanmış olan değişkeni çağırabiliriz. Böylelikle değişkene atanmış değere ulaşabiliriz.

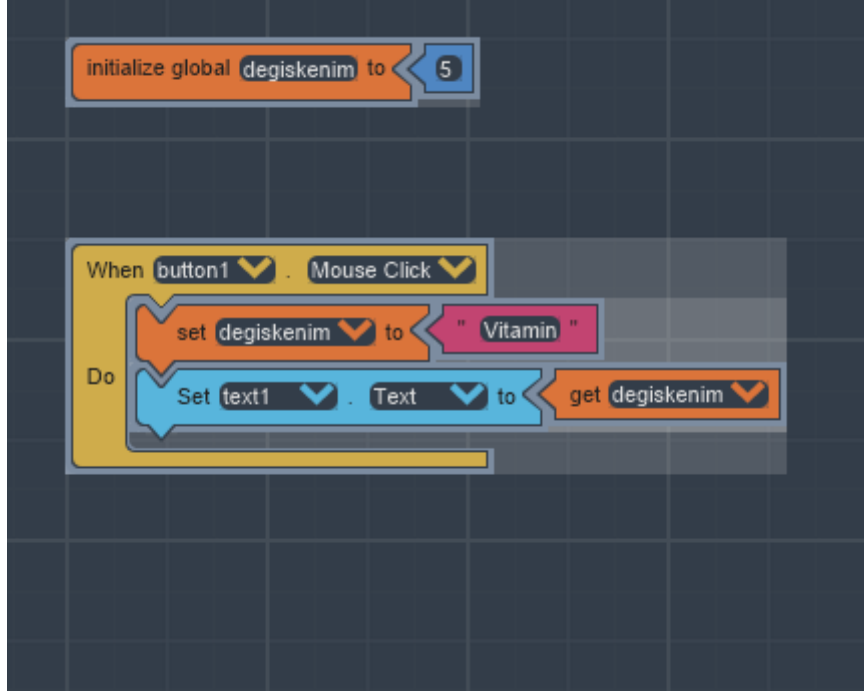


Şekil 19

**Set** erişimcisi ile tanımlanmış olan değişkenin değerini ve veri tipini değiştirebiliriz. Örneğin; başlangıçta 5 olarak tanımlanmış bir değişkenin değerini 10 yapabiliriz.

**Uygulama 1:** Tam sayı değerine sahip genel bir değişken tanımlayıp butona bastığımızda değerini değiştiren bir uygulama yapalım.

*\*Buton ve metin kutusunun özellikleri ilerleyen bölümlerde daha detaylı anlatılacaktır.*



Şekil 20

- 1- *degiskenim* isimli bir değişken tanımlayarak değerini 5 olarak belirledik.
- 2- Butona tıklandığında, değişkenimizin değerini set erişimcisini kullanarak *Vitamin* olarak değiştirdik.
- 3- Get erişimcisi ile değişkenimizi çağırdık ve metin kutusuna değişkenimizin yeni değerini yazdırdık.

Buton

Vitamin

Şekil 21

Şekil 10'da görüldüğü üzere, butona tıkladığımızda ekrana 5 değerini değil, *Vitamin* değerini yazdırdık.

## Operatörler

### Operatör nedir?

Operatörler, programlama dillerinde matematiksel, mantıksal ve ilişkisel işlemleri yapmamızı sağlayan, aynı zamanda tek başına kullanılamayan sembol ya da karakterlere denir.

Operatörlerin işlem yapabilmesi için gerekli olan değişken veya sabit değerlere ise **operand** denir. Örneğin,  $4 + 5$  işlemini ele alırsak, burada rakamlara “operand” toplama işareti ise “operatör” denir. VFabrika’da yer alan operatör tipleri şunlardır:

- Aritmetik Operatörler
- Mantıksal Operatörler
- Karşılaştırma Operatörleri

### Aritmetik Operatörler

Matematiksel işlemler aritmetik operatörler vasıtası ile yapılır. VFabrika’da dört temel aritmetik operatör bulunmaktadır.



Şekil 22: Toplama



Şekil 23: Çıkarma



Şekil 25: Çarpma



Şekil 24: Bölme

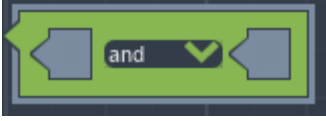
Bunlara ek olarak, diğer programlama dillerinde karakter olarak bulunan Mod alma (%) VFabrika’da karakter olarak değil yazı olarak bulunmaktadır.



Şekil 26: Mod Alma

### Mantıksal Operatörler

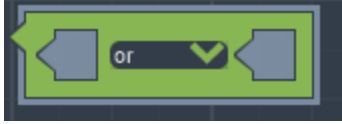
Belirlenen şartları kontrol eden ve true (doğru) ya da false (yanlış) sonuçları çıkaran operatörlere denir. Mantıksal operatörlerde ve (&&), veya (||), değil (!) bulunur. VFabrika’da bu operatörler sembol olarak değil, yazı olarak yer alır.



Şekil 27: Ve Operatörü

And (Ve) operatöründe iki şartın da doğru olup olmadığı kontrol edilir. İki operand da doğru ise sonuç doğru olarak çıkar.

1. Operand	2. Operand	Sonuç
Doğru	Doğru	Doğru
Doğru	Yanlış	Yanlış
Yanlış	Doğru	Yanlış
Yanlış	Yanlış	Yanlış



Şekil 28: Veya Operatörü

Or (Veya) operatöründe iki şarttan birinin doğru olup olmadığı kontrol edilir. İki operand'dan en az biri doğru ise sonuç doğru olarak çıkar.

1. Operand	2. Operand	Sonuç
Doğru	Doğru	Doğru
Doğru	Yanlış	Doğru
Yanlış	Doğru	Doğru
Yanlış	Yanlış	Yanlış

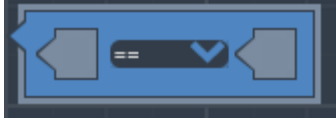


Şekil 29: Değil Operatörü

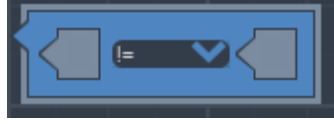
Not (Değil) operatöründe ise bir tane operand alınır ve o operandın tersi sonuç olarak çıkar. Alınan operand doğru ise yanlış, yanlış ise doğru sonucu çıkar.

## Karşılaştırma Operatörleri

İsminden de anlaşılacağı üzere, karşılaştırma operatörleri iki operandı karşılaştırmak için kullanılır. Karşılaştırma sonucu da doğru veya yanlış olarak çıkar. Karşılaştırma operatörleri şunlardır;

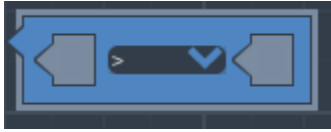


Şekil 30: Eşitse

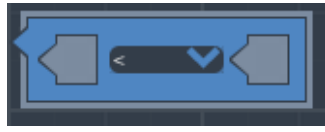


Şekil 31: Eşit Değilse

Eşitse ve eşit değilse iki değerin birbirine eşit olup olmadığını kontrol eder. Eğer şart sağlanıyorsa doğru sağlanmıyorsa yanlış sonucunu çıkarır.



Şekil 33: Büyükse

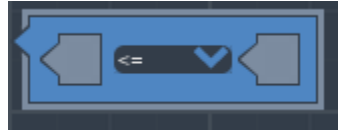


Şekil 32: Küçükse

Büyükse ve küçükse iki değerin birbirinden büyük veya küçük olup olmadığını kontrol eder.



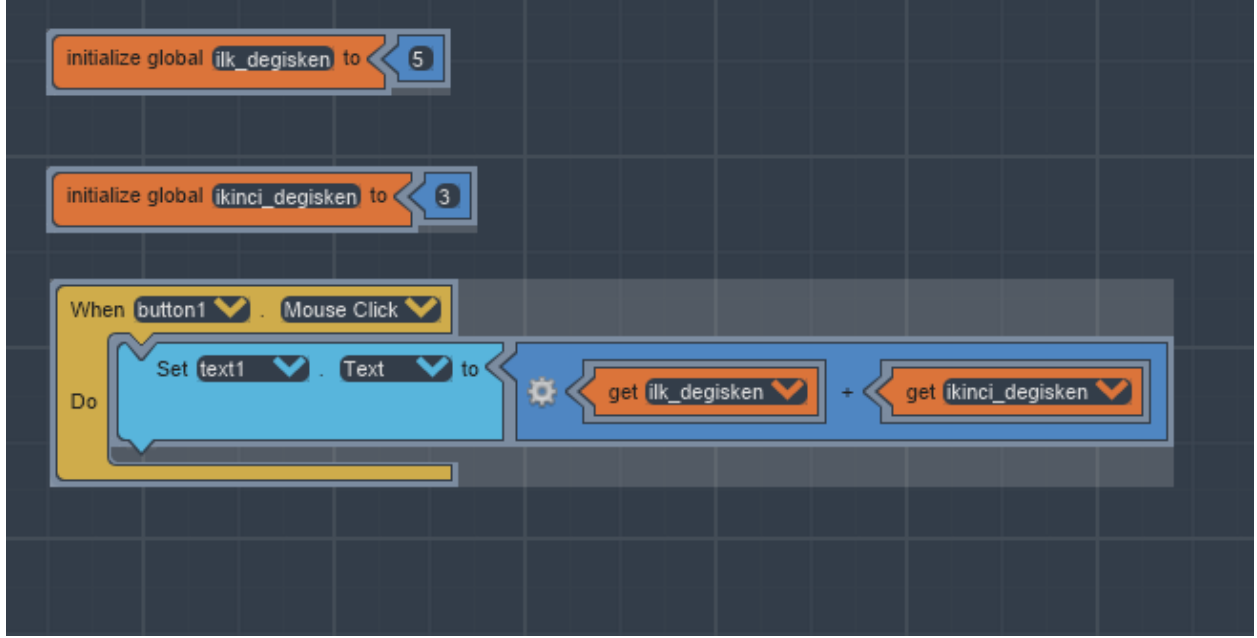
Şekil 34: Büyük Eşitse



Şekil 35: Küçük Eşitse

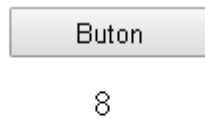
Büyük eşitse ve küçük eşitse iki değerinden birbirinden büyük veya birbirine eşit olup olmadığını kontrol eder. Eğer şart sağlanıyorsa doğru sağlanmıyorsa yanlış sonucunu çıkarır.

**Uygulama 1:** İki farklı değişken tanımlayıp ardından operatörlerle işlemler yapalım. Butona basıldığında da metin kutusuna sonucu yazdıralım.



Şekil 36

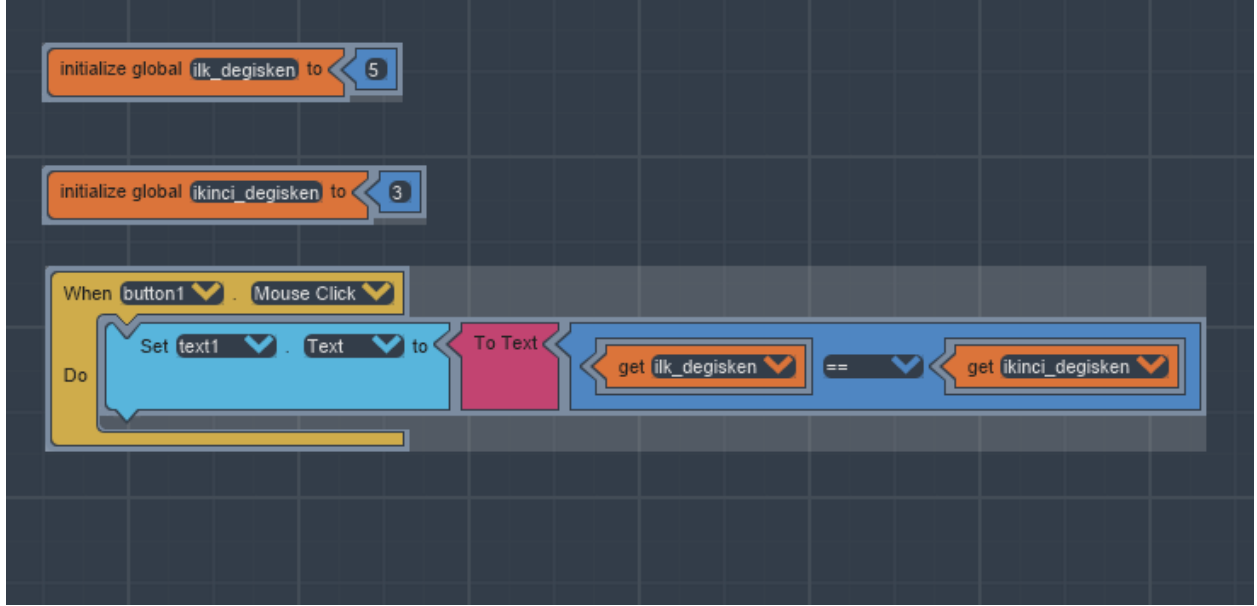
- 1- İki farklı değişken tanımladık.
- 2- Aritmetik operatörlerde gördüğümüz toplama operatörünü bloğumuza ekledik.
- 3- Get erişimcisi ile değişkenlerimizin değerlerini aldık ve toplama işlemine yerleştirdik.
- 4- Son olarak da ekrandaki text'imize sonucu yazdırdık.



Çıktıda da görüldüğü gibi, metin kutusuna 5 ile 3'ün toplamı yazıldı.

Şekil 37

Şimdi de uygulamada tanımladığımız değişkenlerin eşit olup olmadığına bakalım.

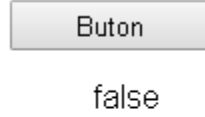


Şekil 38

- 1- İki farklı değişken tanımladık.
- 2- Karşılaştırma operatörlerinde gördüğümüz eşitse operatörünü bloğumuza ekledik.
- 3- Get erişimcisi ile değişkenlerimizin değerlerini aldık ve eşitse bloğuna yerleştirdik.
- 4- To Text (Metne Çevir) bloğunu da sonucu metin olarak görmek istediğimiz için bloğumuza ekledik.
- 5- Son olarak da ekrana sonucu yazdırdık.

**NOT:** Metne Çevir bloğunu eklemesek VFabrika yine değişkenlerin birbirine eşit olup olmadığını kontrol eder fakat bunu text1 kutusuna yazamaz. Programlama dillerinde doğru 1 yanlış da 0 ile temsil edilir. Böyle bir karşılaştırma durumunda da sonuç ya 1 ya 0 olarak çıkar fakat 1 ve 0'ı ekrana yazdıramayız. Bunun yerine To Text (Metne Çevir) bloğunu kullanırız. Böylelikle program gelen sonucun ne olduğuna bakar ve onu metne çevirir. Bizim uygulamamızda da eşitse operatörünün sonucu 0 olarak çıkar ve To Text bloğu bunu false (yanlış) olarak metne çevirir.





Çıktıda da görüldüğü gibi, butona bastığımızda metin kutusunda false (yanlış) değerini yazdırdı.

Şekil 39

## Karar Kontrol Yapıları

Programlamada, tıpkı gerçek hayatta olduğu gibi, karşımıza koşul ve ihtimaller çıkabilir. Bu tür durumlarda ne yapılacağına önceden karar vermek gerekir. Karar kontrol yapıları, program içinde şartlara bağlı olarak hangi kod bloklarının çalıştırılıp çalıştırılmayacağını uygulayan yapılardır. VFabrika’da bulunan karar kontrol yapıları şunlardır: If Then (Eğer ise) If Then Else (Eğer ise Değilse), If Then Else Return (Eğer ise Değilse Döndür) ve Switch Case (Değişken Durum).

### If Then (Eğer ise)

If Then en temel karar kontrol yapısıdır. Program içinde bir koşul komutu oluşturmamızı ve o şartın sağlanıp sağlanmadığını kontrol etmemizi sağlar. If kısmında şart, then kısmında da şart sağlandığında ne yapılacağı belirtilir.

Blok üzerindeki çarka tıkladığımızda istenildiği kadar else if (değilse eğer) veya else (değilse) eklenebilir.



Şekil 40: If Then

### If Then Else (Eğer ise Değilse)

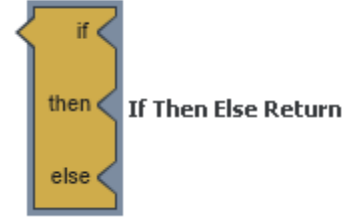


Şekil 41: If Then Else

If Then Else oluşturulan şartın sağlanıp sağlanmadığını kontrol eder. If Then bloğundan farklı olarak koşul sağlanmadığında da bir komut oluşturabiliriz. Koşul sağlandığında then, sağlanmadığında else kısmında belirtilen bloklar çalışır.

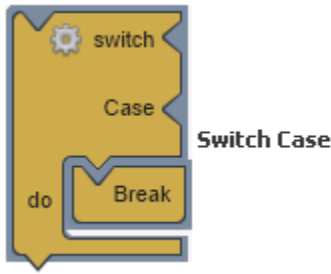
## If Then Else Return (Eğer ise Değilse Döndür)

If Then Else Return, belirlenen şartın sağlanıp sağlanmadığını kontrol eder ve eğer şart sağlanmıyorsa belirtilen bloğu döndürür.



Şekil 43: If Then Else Return

## Switch Case (Değişken Durum)



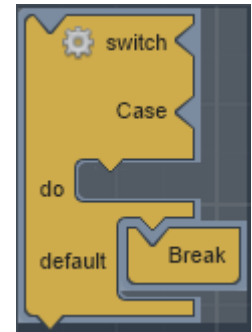
Şekil 44: Switch

Switch Case de aynı If Then gibi karar kontrol yapısıdır fakat farklılıkları vardır. Switch Case’de koşul tanımlamak yerine bir değişkenin durumuna bakarak karar verilir. Örneğin, 1 ile 7 arasında bir sayı değerine sahip değişken tanımlayıp bunun hangi güne karşılık geldiğini gösteren bir program yaptığımızda, her seferinde şartımızı ayrı ayrı yazmak yerine Switch Case’de switch kısmına 1 ile 7 arasında değere sahip olan değişkenimizi koyarız, case (durum) kısmında 1, 2, 3 vs. şeklinde değerlerimizi koyup do kısmında da günleri belirtebiliriz. Böylelikle değişkenimiz 3 ise

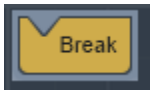
Çarşamba, 5 ise program bize Cuma’yı gösterir.

Bloğu eklediğimizde, varsayılan olarak bir tane case gelir fakat bloğun üzerinde bulunan çarka tıklayarak istediğimiz kadar case ekleyebiliriz.

**Default:** Switch Case’de değişkenimiz durumların hiçbiri ile eşleşmediğinde ne yapılmasına gerektiğine default ile karar veririz. Switch Case bloğumuzda varsayılan olarak default gelmemektedir. Ancak çarka tıkladığımızda kolaylıkla ekleyebiliriz.



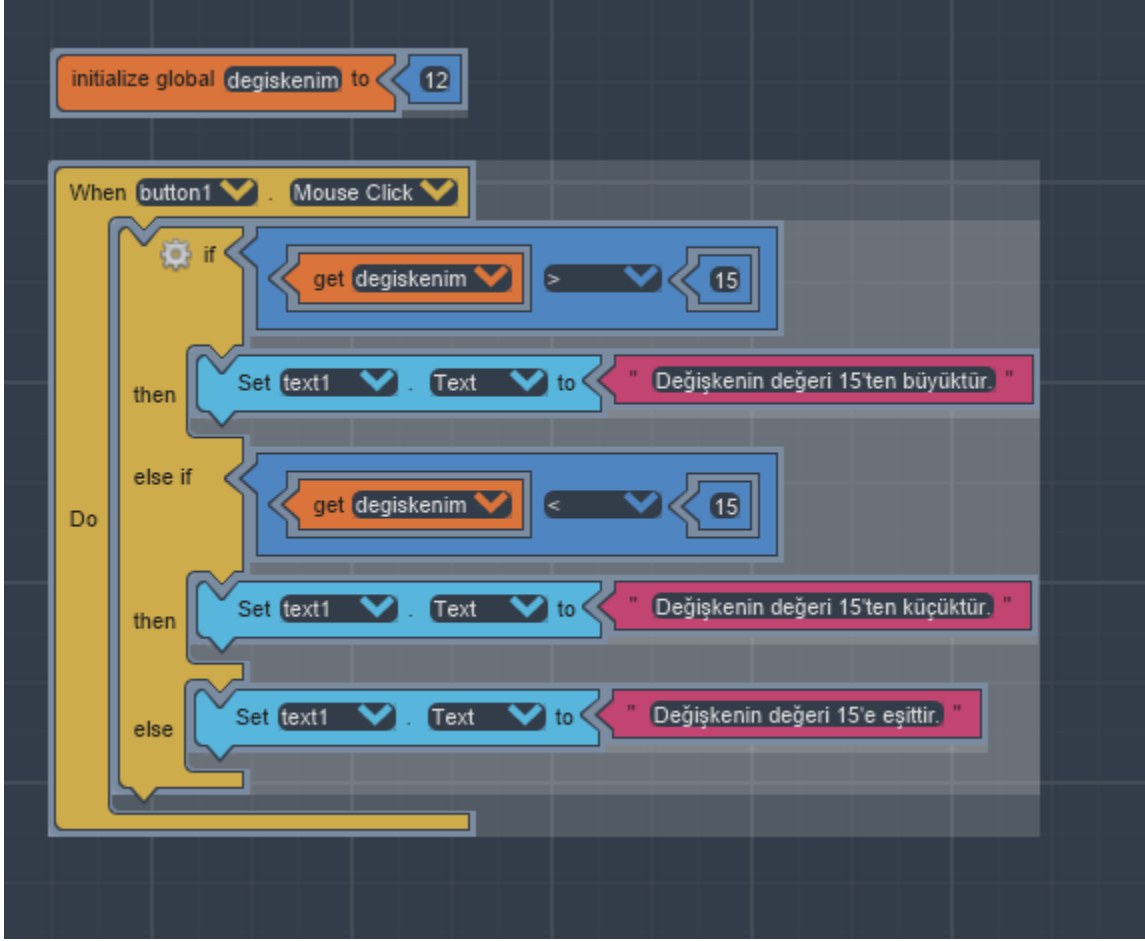
Şekil 45: Default



Şekil 46: Break

**Break (Durdur):** Değişkenimiz belirtilen durumlardan biriyle eşleştiğinde diğer durumlara bakılmasını istemiyorsak durdur bloğunu kullanırız. Böylelikle değişken Switch Case’den çıkar. Aksi durumda değişken şartın sağlandığı her bloğu kontrol eder.

**Uygulama:** Bir değişken tanımlayalım ve bu değişkenin 15'ten büyük mü küçük mü yoksa 15'e eşit mi olduğunu kontrol edelim.



Şekil 47

- 1- Değeri 12 olan *degiskenim* adlı bir değişken tanımladık.
- 2- Butona tıkladığımız zaman ekrana yazmasını istediğimiz için Mouse Click bloğunu ekledik.
- 3- Kıyaslamayı yapabilmek için If Then kontrol bloğunu ekledik.
- 4- Şartlarımızı oluşturduk, eğer büyük veya küçük değil ise eşit olduğundan ötürü son olasılığımızı da yerleştirdik.
- 5- Ekrandaki Text'e yazdırmak istediğimiz için Set Text bloğunu ekledik ve ekrana yazdırdık.

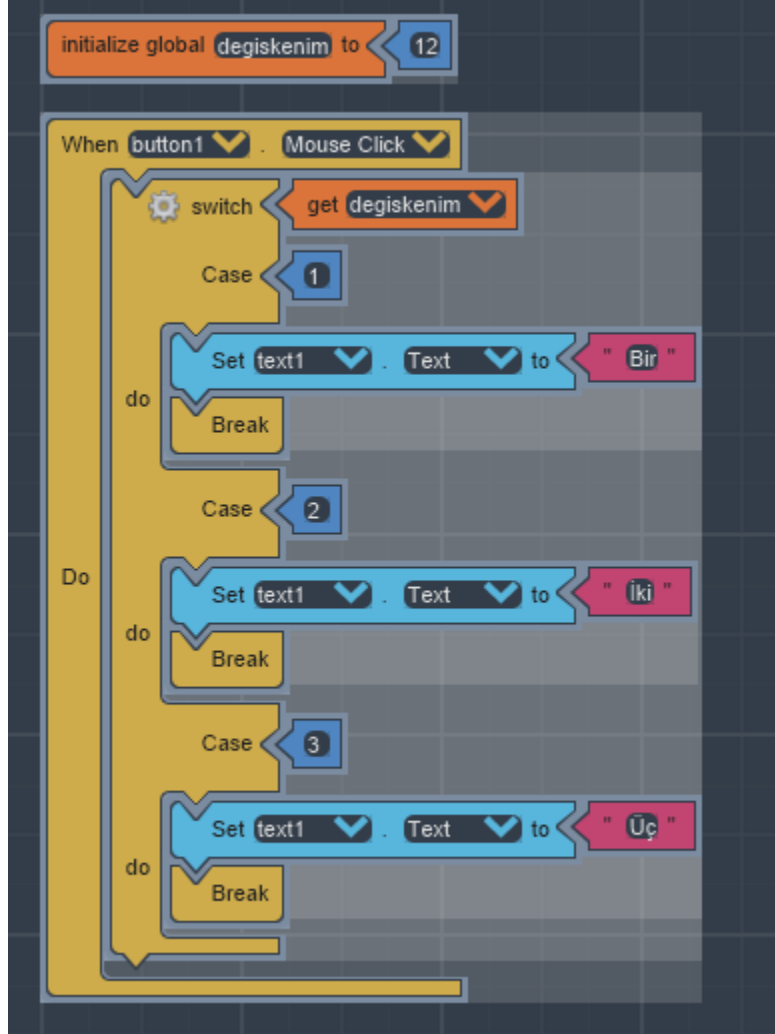
Buton

Değişkenin değeri  
15'ten küçüktür.

Çıktıda da görüldüğü gibi, değişkenimize 12 değerini verdik ve if then karar kontrol yapısıyla kontrol ettirdikten sonra çıkan sonucu ekrana yazdırdık.

Şekil 48

**Uygulama:** Bu uygulamamızda da switch case'i kullanarak 1-3 arası bir değişken tanımlayarak onu yazı ile ekrana yazdıralım.



Şekil 49

- 1- Değeri 3 olan *degiskenim* adlı bir değişken tanımladık.
- 2- Butona tıkladığımız zaman ekrana yazmasını istediğimiz için Mouse Click bloğunu ekledik.
- 3- Değişkenimizi kontrol edebilmek için switch case bloğu ekledik ve çarka tıklayarak 2 tane daha durum yarattık.
- 4- Switch bölümüne değişkenimizi yerleştirerek kontrol edilebilmesini sağladık.
- 5- Case bölümlerine de olabilecek ihtimalleri yerleştirdik ve ekrana yazdırdık.



Çıktıda da görüldüğü gibi, değişkenimize 3 değerini verdik ve switch case karar kontrol yapısıyla kontrol ettirdikten sonra çıkan sonucu ekrana yazdırdık.

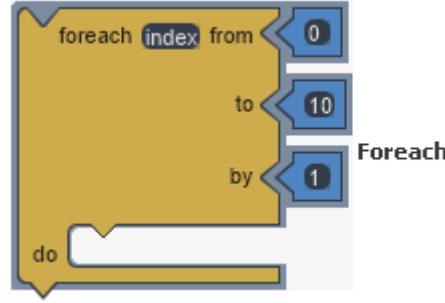
Şekil 50

## Döngüler (Loops)

Döngüler bir kod bloğunun birden çok kez tekrar edilmesini sağlayan yapılara verilen addır. Daha hızlı ve daha kullanışlı kod blokları oluşturmamızı sağlar. Örneğin, 1'den 5'e kadar olan sayıları ekrana yazdırmak istediğimizde bu sayıları tek tek yazmamız gerekirken döngüleri kullanarak bu sayıları tek seferde ekrana yazdırabiliriz. VFabrika'da bulunan döngü türleri şunlardır; Foreach (Her Eleman İçin), While (Sürekli Yap) ve Do While (Yap ve Doğruysa Sürekli Yap) döngüsü.

### Foreach Döngüsü (Her Eleman İçin Döngüsü)

Bu döngüde koşul sağlanana dek bloklar çalıştırılır. Üç temel özelliği vardır; from (başlangıç), to (bitiş) ve by (artış). From kısmında döngünün hangi değerden başlayacağını, to kısmında şartı, by kısmında da başlangıç değerinin her döngüde kaç arttırılacağını belirtiriz.



Şekil 51: Foreach

Örneğin, varsayılan olarak gelen blokta from 0, to 10, by 1'dir. Buna göre, döngü 0'dan başlar, değerin 10'dan büyük olup olmadığını kontrol eder, büyük değilse 1 arttırarak do kısmındaki blokları çalıştırır. Değer 10'dan büyük olduğunda şart sağlanmış olur ve döngüden çıkılır.

*index* isimli varsayılan olarak gelen değer bizim bu döngüdeki değişkenimizdir. Bu değişken local (yerel) bir değişken olmakla birlikte do kısmında bu değişkenimizi kullanabiliriz. Foreach döngüsü *index*'i döngü içerisinde arttırabilir fakat başka bir değişken kullandığımızda eğer biz arttırmadıysak döngü tarafından arttırılmaz.

### While Döngüsü (Sürekli Yap Döngüsü)



Şekil 52: While

Foreach döngüsü gibi bu döngü de bir şarta bağlı olarak tekrarlama işlemlerini yapar. While kısmında belirtilen şart, do kısmında ise şart sağlandığı sürece yapılacak olan bloklar yer alır.

Döngüyü bitirecek bir şart olmazsa döngü sürekli olarak kendini tekrar eder ve sonsuz bir döngüye girebilir.

### Do While Döngüsü (Yap ve Doğruysa Sürekli Yap Döngüsü)

Foreach ve While döngülerinde önce şart kontrol edilir, eğer şart sağlanıyorsa döngü içindeki bloklar çalıştırılırdı. Do while döngüsünün en önemli farkı da do kısmında belirtilen blokların ilk olarak şart kontrol edilmeden uygulanmasıdır. Daha sonra while kısmındaki şart kontrol edilerek tekrarlama işlemi devam eder ya da döngü biter.



Şekil 53: Do While

## Break (Durdur)



Şekil 54: Break

Break bloğuna Karar Kontrol Yapılarında Switch Case başlığında değinmiştik. Orada olduğu gibi döngülerde de aynı şekilde kullanılmaktadır. Belirtilen bir şart sağlandığında döngüden çıkılmasını istiyorsak Break bloğunu kullanırız.

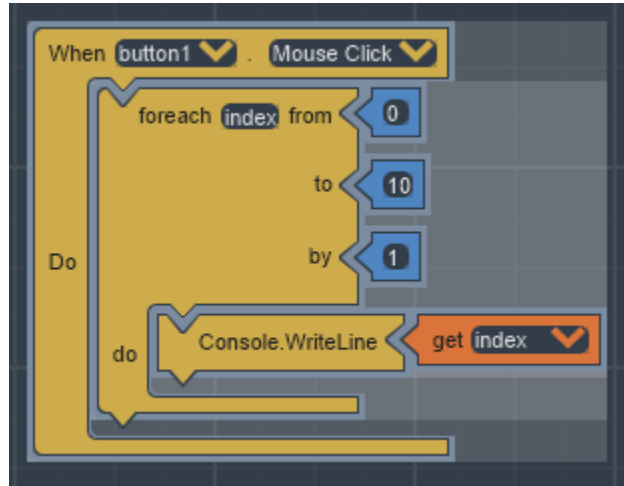
## Continue (Devam Et)



Şekil 55: Continue

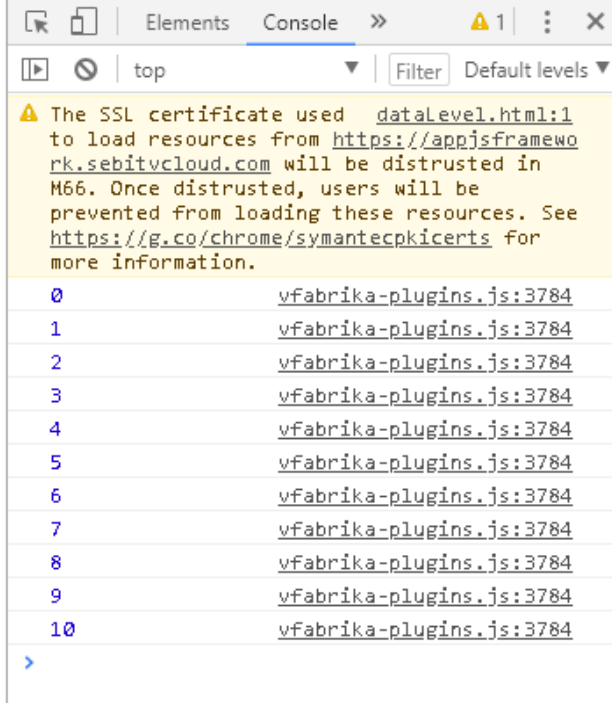
Continue bloğu kullanıldığında döngüden çıkmaz fakat döngüde bir sonraki tekrara geçilir. Örneğin, 1'den 10'a kadar olan sayıları yazan bir programda "4'e eşit olduğunda" gibi bir şart koyup Devam Et bloğunu eklersek 4 atlanarak direkt 5'e geçilir. Continue bloğundan sonraki bloklar atlanarak döngü 1 atlanır.

**Uygulama:** Foreach döngüsünü kullanarak 0'dan 11'e kadar olan sayıları konsola yazdıralım.



Şekil 56

- 1- Butona tıkladığımızda yazdırmasını istediğimiz için Mouse Click bloğunu ekledik.
- 2- Foreach döngüsünde *index* bizim local değişkenimizdir. Bu sebeple *index* isimli değişkenimiz 0'dan başlar, 10'a kadar 1 artarak devam eder fakat 10 da işleme dâhil edilir.
- 3- Local değişkenimizi konsola yazdırmak istediğimiz için Console.WriteLine bloğunu kullandık ve ona da get erişimcisi ile *index* isimli değişkenimizi yerleştirdik.



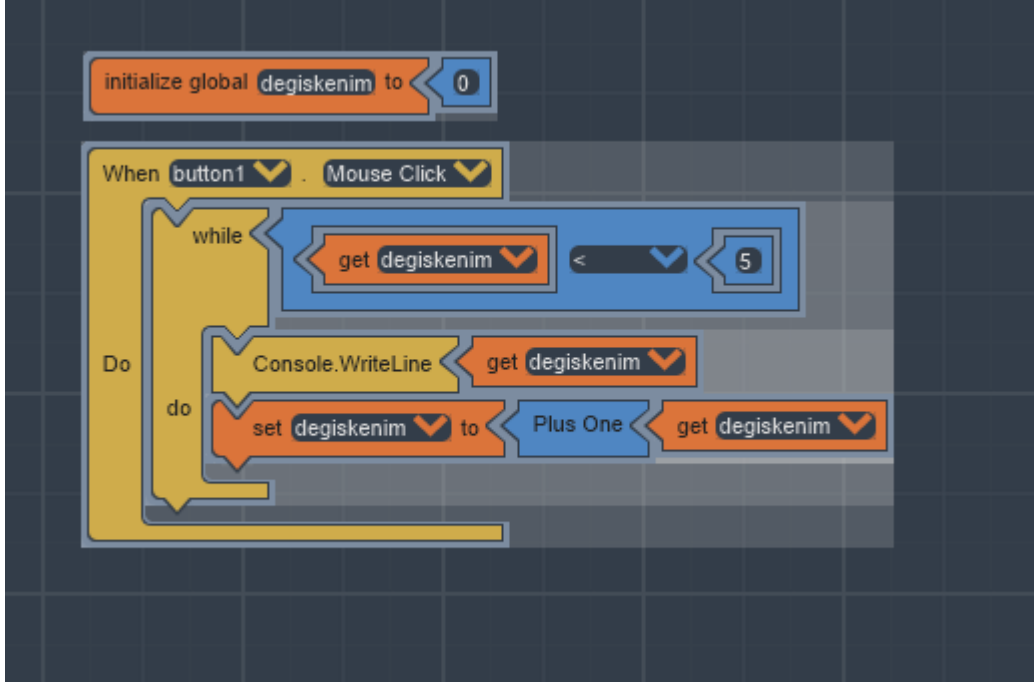
Çıktıda da görüleceği üzere 0'dan 11'e kadar olan sayıları konsola yazdırdık.

Şekil 57

**NOT:** Konsol, metin tabanlı arayüz programıdır. Konsolda grafikler yer almaz. Birçok programlama dili öğretilirken konsol üzerinden öğretilir. Bunun sebebi de önce programlama ve algoritmanın öğretilmesidir. Ayrıca, konsollar kodları daha hızlı şekilde derleyebilir. VFabrika'da uygulamalarımızı yaparken varsayılan tarayıcıyı kullanırız. Her tarayıcı kendi konsoluna sahiptir ve konsol üzerinden de oluşturduğumuz blokları çalıştırabiliriz. `Consol.WriteLine` bloğu da konsola yazdırmamızı sağlar.

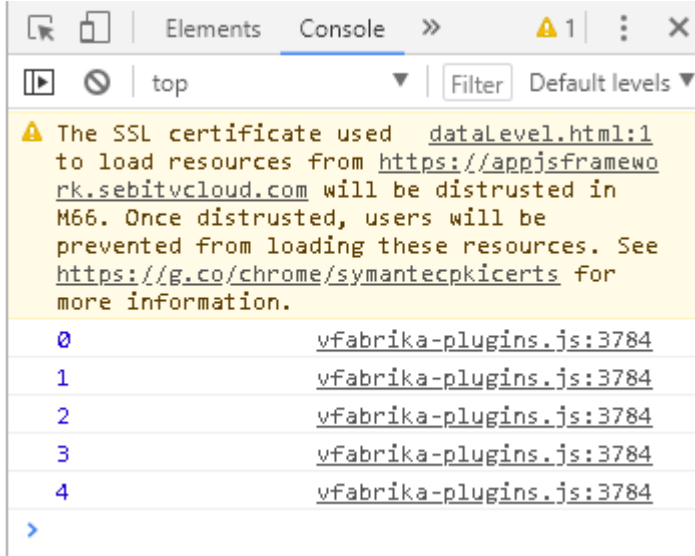
**Uygulama:** Bu uygulamamızda da değeri sıfır (0) olan bir değişken tanımlayıp while döngüsü ile 5'e kadar konsola yazdıralım.





Şekil 58

- 1- Değeri 0 olan *degiskenim* isimli global bir değişken tanımladık.
- 2- Butona tıkladığımız zaman ekrana yazmasını istediğimiz için Mouse Click bloğunu ekledik.
- 3- While döngüsünde get erişimcisi ile değişkenimizin değerini karşılaştırma operatörünü kullanarak 5'ten büyük olup olmadığını kontrol ettirdik.
- 4- Şartı sağladığı sürece konsola değişkenimizi yazdırdık.
- 5- Yine şartı sağladığı sürece; Get erişimcisi değişkenimizin değerini aldık, Plus one bloğu ile değişkenimizi 1 arttırdık. Değişkenimizin değerini değiştirmek istediğimiz için set erişimcisini kullandık.



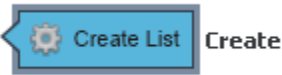
Şekil 59

Çıktıda da görüldüğü gibi, 0'dan 5'e kadar olan sayılar konsola yazdırıldı. Blok sırasında önce değeri yazıp ardından arttırma işlemi yaptığımız için 0'dan başlayıp 5'e kadar yazdırdı fakat 5'e geldiğinde şart (5'ten küçük olduğu sürece) sağlanmadığı için 5'i yazdırmadan döngüden çıktı.

## Listeler

Şu ana kadar gördüğümüz değişkenlerde her zaman tek bir değeri tuttuk. Listeler bizim aynı değışkende birden çok değeri tutmamızı sağlayan yapılardır. Değişkenleri boş bir kutu olarak düşündüğümüzde; listeler, bölmeleri olan bir kutu olarak düşünülebilir. Örneğin; bir kişinin adını, soyadını ve oturduğu şehri listeler yardımıyla tek bir değışkende tutabiliriz. Listelerin en önemli özelliklerinden biri sıralı olmasıdır. Biz de listenin içindeki değerlere ulaşmak için bu sırayı kullanırız. Listelerin sıralı olmasındaki bir diğer önemli nokta da listelerde sıranın 0'dan başlamasıdır. Bizim listedeki ilk değeri sıfırıncı index'te bulunur.

### Create List (Liste Oluşturma)

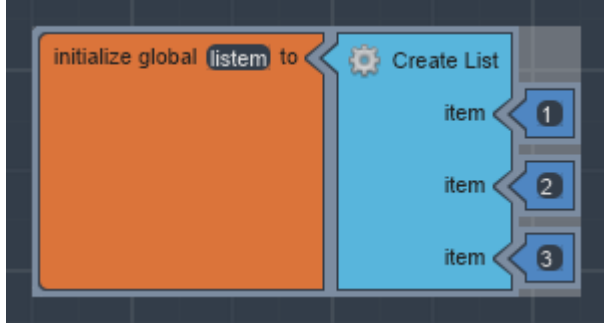


Şekil 60: Create

Bir liste oluşturmak için Create List bloğunu kullanırız. Çarka tıklayarak istediğimiz kadar item (öge) ekleyebiliriz.

Liste oluştururken dikkat edilmesi gereken nokta, değışken kapsama alanıdır. Listeler local (yerel) değışkenlerde tanımlanamaz. Yalnızca global (genel) değışkenlerde tanımlanabilir.

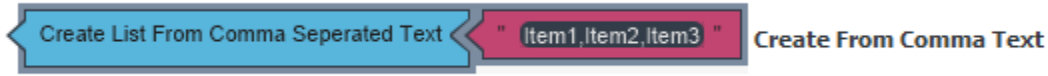
**Uygulama:** 1, 2 ve 3 değerlerine sahip global bir liste oluşturalım.



Şekil 61

Liste oluşturmak da değişken oluşturmakla benzerdir. Listede birden çok değer tutabildiğimiz için, çarka tıklayarak ekleyeceğimiz değer kadar item ekleyebiliriz.

### Create From Comma Text (Virgülle Ayrılmış Metinden Liste Oluştur)



Şekil 62: Create From Comma

Metin içeriğine sahip öğeleri daha kolay şekilde eklememizi sağlar. Liste oluştururken eklemek istediğimiz verileri tek tek bloğa eklemek durumunda kalırken, bu blok ile beraber daha kısa zamanda öğeleri ekleyebiliriz. Ekleyeceğimiz verileri virgül ile ayırmamız yeterlidir.

### Add Item (Öge Ekleme)



Şekil 63: Add Item

Listeye ekleyeceğimiz öğeleri her zaman en başında tanımlamak zorunda değiliz. Listelerde add item bloklar içinde de öge ekleyebiliriz. List bölümüne hangi listeye eklemek istediğimizi, item kısmına da eklemek istediğimiz öğeyi yerleştirebiliriz. Eklenen öge listede en son sıraya eklenir.

### Add Item Into (Sıraya Öge Ekle)



Şekil 64: Add Item Into

Listeye öge eklediğimizde listenin son sırasına eklenir fakat Add Item Into bloğu ile istediğimiz sıraya istediğimiz öğeyi ekleyebiliriz. Örneğin başlangıç pozisyonuna öge eklemek istersek Into bölümünde 0 değerini koyarız ve öğeyi ekleriz. Böylelikle var olan öğelerin de sırası değişir. 1.

sırada olan öge 2. sıraya, 2. sırada olan öge de 3. sıraya gelir.

### Remove Item (Öge Çıkarma)



Şekil 65: Remove Item

Listelere bloklar içinde öge ekleyebildiğimiz gibi öge de çıkarabiliriz. Bunun için Remove Item bloğu kullanılır. List bölümüne hangi listeden çıkarmak istediğimizi, item kısmında da çıkarmak istediğimiz ögeyi yerleştirebiliriz.

### Remove Item At (Pozisyondaki Ögeyi Sil)



Şekil 66: Remove Item At

İstediğimiz ögeyi çıkarabildiğimiz gibi, index'ine göre de ögeleri listeden çıkarabiliriz. Bunun için Remove Item At bloğu kullanılır. List kısmına hangi listeden çıkarılacağı, index bölümüne de hangi sıradaki ögenin çıkarılacağı eklenir.

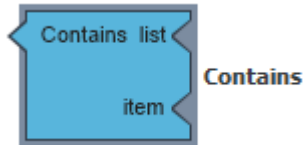
### Length (Liste Uzunluğu)



Şekil 67: Length

Length bloğu, listedeki öge sayısına ulaşmamızı sağlar. Tam sayı döndürür.

### Contains (Mevcut)



Şekil 68: Contains

Program içinde hangi ögenin bulunup bulunmadığını kontrol eden Contains (Mevcut) bloğu, yalnızca true (doğru) ya da false (yanlış) değerlerini gösterir.

### Get Item At (Pozisyondaki Ögeyi Oku)



Şekil 69: Get Item At

Get Item At bloğu, girilen index sayısına bağlı olarak listede o index'teki değeri getiren bloklardır.

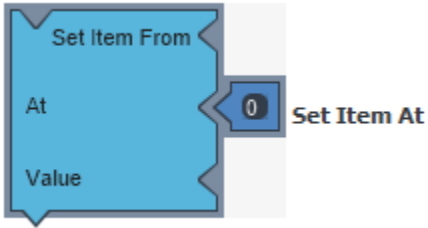
### Index Of (Pozisyon Bul)



Şekil 70: Index Of

Listenin içinde pozisyonunu bulmak istediğimiz değerler olduğunda Index Of bloğunu kullanırız. List kısmında listeyi, Index Of'a da değeri yerleştiririz. Bu işlemin ardından VFabrika bize ögenin pozisyonu döndürür.

### Set Item At (Pozisyondaki Ögeyi Değiştir)



Şekil 71: Set Item At

Program içinde listedeki değerleri değiştirmemiz gerekebilir. Bu tür durumlarda Set Item At bloğunu kullanırız. From kısmında hangi listeye bakacağımızı, At kısmında index'i ve Value kısmında da yeni değeri yerleştirebiliriz. Bu blok yardımıyla, örneğin, 3. index'teki ögenin değerini kolaylıkla değiştirebiliriz.

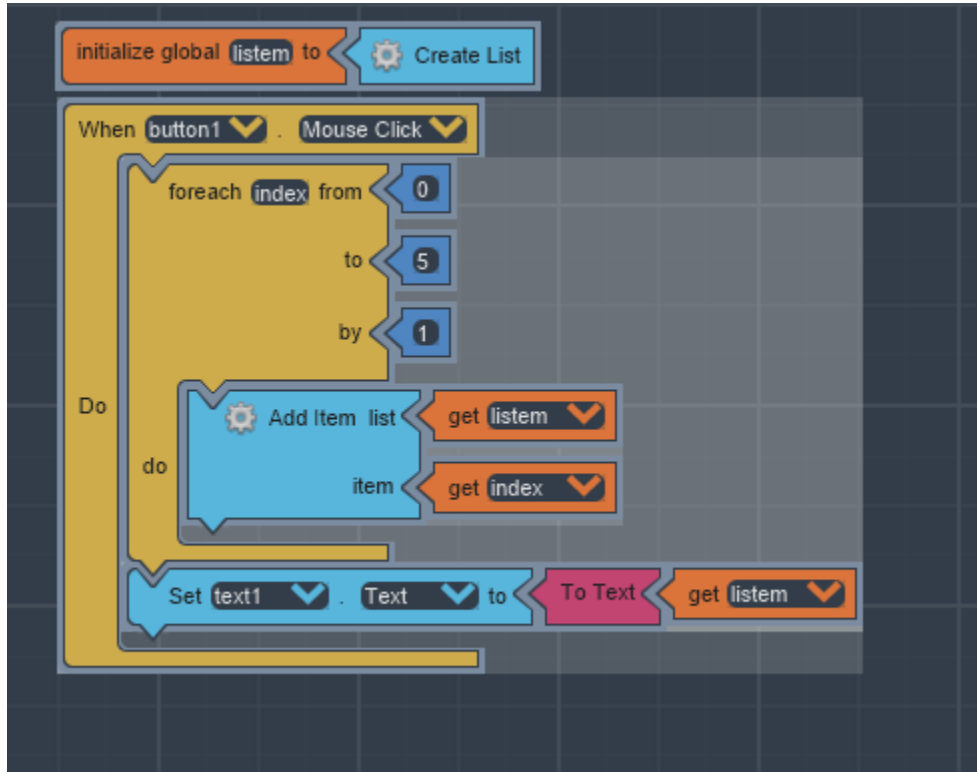
### Shuffle List (Listeyi Karıştır)



Şekil 72: Shuffle List

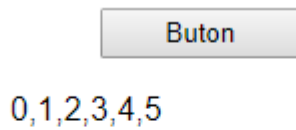
Listelerde ögeler her zaman sıralı hâlde bulunur fakat bu sırayı değiştirmek gerekebilir. Shuffle List, listedeki pozisyon sırasını kolayca değiştirmemizi sağlayan blok tipidir. Liste oluşturulurken veya blok yapıları içinde kullanılabilir.

**Uygulama:** *listem* isimli bir liste oluşturalım ve 0'dan 5'e kadar olan tam sayıları foreach döngüsü kullanarak listeye ekleyelim ve ekrana yazdıralım.



Şekil 73

- 1- Global bir *listem* isimli liste oluşturduk.
- 2- Butona tıkladığımız zaman ekrana yazmasını istediğimiz için Mouse Click bloğunu ekledik.
- 3- Foreach döngüsü ekledik 0'dan 5'e kadar 1 artacak şekilde düzenledik.
- 4- Add Item List ile oluşturduğumuz listeye 0'dan 5'e kadar olan değerleri ekledik. Index değişkeni yalnızca foreach döngüsü içinde kullanıldığından dolayı da onu yerleştirdik.
- 5- Set Text ile de listemizi ekrandaki Metin kutusuna yazdırdık.



Çıktı da görüldüğü üzere, butona tıkladığımızda listemize öğeleri ekledik ve ekrana da yazdırdık.

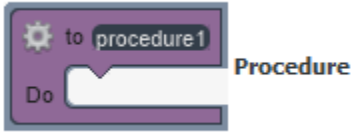
Şekil 74

## Prosedürler

### Prosedür Nedir?

Her zaman az bir kod bloğu ile çalışmayabiliriz. Böyle bir durumda aynı blokları tekrar ve tekrar yerleştirmek hem programı yorar hem de kullanıcının zamanını çalar. Prosedürler, kod bloklarını küçük parçalara bölmemizi ve daha kolay şekilde yönetmemizi sağlayan yapılardır. Programlamanın temel yapı taşlarından biri olan prosedürler, programın çeşitli yerlerinde tekrar tekrar kullanılabilir ve değiştirilebilir.

### Procedure (Prosedür Oluşturma)



Şekil 75: Procedure

Prosedür oluşturmak için Procedure bloğunu kullanırız. Prosedüre isim verebilir ve Do bölümüne de prosedürün yapmasını istediğimiz kod bloklarını yerleştirebiliriz. Çarka tıkladığımızda prosedüre istediğimiz kadar input (girdi) girebiliriz.

### Call (Çağırma)

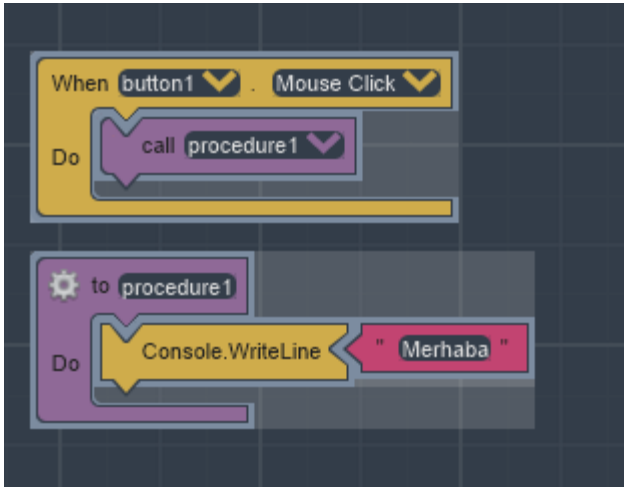


Şekil 76: Call

Prosedür oluşturduktan sonraki diğer adım, oluşturduğumuz prosedürü çağırma. Oluşturulan prosedür çağırılmadığı takdirde programa herhangi bir etkisi olmaz. Bloкта bulunan açılır pencere ile çağırılmasını istediğimiz prosedürü seçebiliriz. Bunun yanında, programın istenilen yerinde ve istenildiği kadar prosedür çağırılabilir.

Bu da prosedürlerin en büyük avantajlarından biridir.

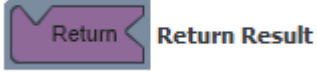
**Uygulama:** Konsola “merhaba” yazdıran bir prosedür oluşturalım ve onun nasıl çağırıldığına bakalım.



Şekil 77

Görüldüğü üzere Procedure bloğuna istediğimiz bloğu/blokları ekledikten sonra Call bloğunu da prosedürün çağırılacağı yere yerleştiriyoruz. Böylelikle bir işlemi her seferinde tekrar tekrar yapmak zorunda kalmıyoruz.

## Return Result (Sonuç Çağırma)



Şekil 78: Return Result

Prosedürün değer döndürmesini istediğimiz durumlarda Return Result bloğu kullanılır. Prosedür bloğunda genellikle en altta bulunan Return Result bloğu opsiyoneldir. Örneğin, programın belirli yerlerinde 2 farklı sayının toplamını istiyorsak burada prosedürün içinde sayıları toplayıp Return Result bloğu ile de

sonucu gönderebiliriz.

## Call for Result (Sonuç Döndüren Prosedür Çağırma)

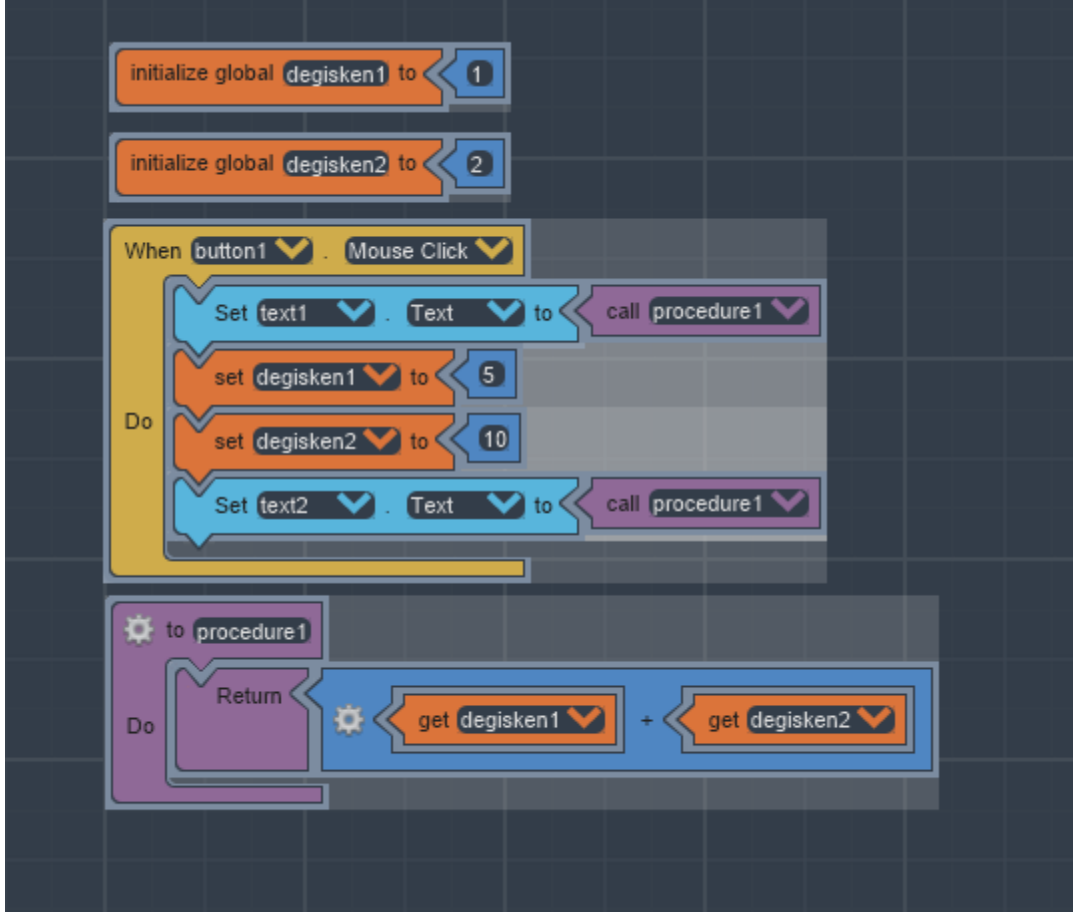


Şekil 79: Call For Result

Değer döndüren prosedürler oluşturduğumuzda, döndürülen değer blok içinde kullanılması gerekebilir. Bu tür durumlarda Call for Result bloğunu kullanırız.

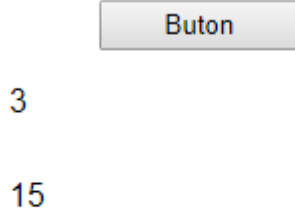
**Uygulama:** İki değişkenin prosedür içinde toplama işlemini yapalım ve elde edilen sonucu yazdıralım.





Şekil 80

- 1- *degisken1* ve *degisken2* diye iki farklı değişken tanımladık.
- 2- Butona basıldığında değişkenlerin ilk tanımladığımız değerleri ile prosedürden toplama işleminin sonucu çağırdık ve text1'e yazdırdık.
- 3- *degisken1* ve *degisken2*'nin değerlerini değiştirdik.
- 4- Prosedürden değişkenlerin yeni değerleri ile toplama işleminin sonucu çağırdık ve text2'e yazdırdık.



Şekil 81

Program içinde var olan değişkenlerin değerini değiştirmemiz gerekebilir. Toplama işlemi prosedür içinde tanımlanmamış olsaydı değişkenlerin değeri her değiştiğinde toplama işlemini tekrar ve tekrar yapmak zorunda kalacaktık.

## String (Metin)

Değişkenler konusu anlatılırken String ile bir veri tipi olarak karşılaştık. Şimdi de bu konuyu daha ayrıntılı olarak inceleyeceğiz.

String, tek bir öge gibi davranan karakterler dizisidir. String ile karakterler üzerinde oynamalar yapılabilir ve karakterleri saklayabiliriz.



Şekil 82: Value

String değerini VFabrika'da Value (Değer) bloğu ile birlikte gireriz. Bu bloğa harfler, semboller ve sayılar girilebilir fakat hepsi karakter olarak işlenir. Örneğin, bloğun içine 15 yazdığımızda sayısal değer olarak değil karakter olarak işlenir. Aritmetik işlemler yapılmaz.

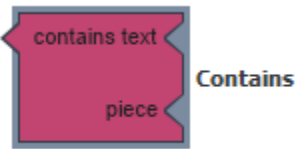
## Compare Text (Karşılaştır)



Şekil 83: Compare

Girilen iki karakter dizisini karşılaştıran blok yapısıdır. Açılır pencerede; eşittir ve eşit değildir olmak üzere iki farklı seçenek vardır.

## Contains (İçeriyor mu)



Şekil 84: Contains

Bir string içinde istenilen karakterin olup olmadığını kontrol eden blok yapısıdır. Text kısmında kontrol edilecek string, piece bölümünde ise aranacak karakter eklenir.

## Is Empty (Boş mu)



Şekil 85: Is Empty

Eklenen string'in karakter içerip içermediğini kontrol eden blok yapısıdır. String karakter içeriyorsa True (Doğru), içermiyorsa False (Yanlış) değerini döndürür.

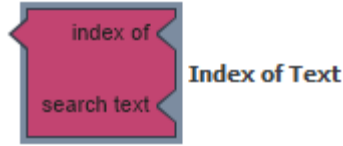
## Join (Ekle)



Şekil 86: Join

Eklenen string'leri birleştirerek tek bir string hâline getiren blok yapısıdır. Üzerinde bulunan çarka tıklanarak istenildiği kadar bölme açılabilir. Birleştirme sırası yukardan başlayarak aşağıya doğru ilerler.

## Index of Text (Metnin Konumunu Bul)



Şekil 87: Index of Text

Aranılan karakter veya karakterin string'in içinde kaçınıcı pozisyonda olduğunu gösteren blok yapısıdır. Stringlerde pozisyon sırası 0'dan başlar. Index of Text bloğu bulunan ilk pozisyonu döndürür. Örneğin, "mehmet" isminde "m" karakteri olup olmadığını kontrol ettirdiğimizde bize 0 değerini döndürecek.

Aranılan karakteri string içinde aramaya başlar ve ilk bulduğunda pozisyonu döndürür. String'in geri kalanına bakmaz. Aranılan karakteri string içerisinde bulamazsa -1 değerini döndürür.

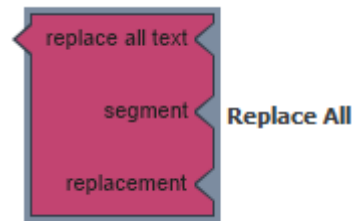
## Length (Uzunluk)



Şekil 88: Length

String'deki karakter sayısını veren blok yapısıdır.

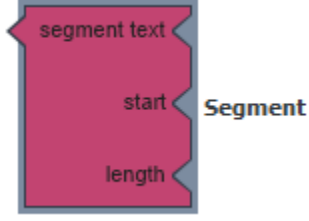
## Replace All (Hepsini Değiştir)



Şekil 89: Replace All

String içinde bulunan kelime veya cümleyi değiştirmeyi sağlayan blok yapısıdır. Text kısmına değiştirecek olan string eklenir, segment kısmına hangi bölümün değiştirileceği belirtilir ve replacement bölümünde de hangi string ile değiştireceğini ekleriz. Örneğin; "Ahmet kodlamayı seviyor." diye bir string'e sahip olalım. Segment kısmına "Ahmet" ekleyip replacement kısmına da "Ali" ismini eklersek yeni string'imiz "Ali kodlamayı seviyor." olacaktır.

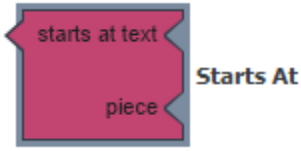
## Segment (Böl)



Şekil 90: Segment

String’de belirtilen değerler arasındaki karakterleri almamızı sağlayan blok yapısıdır. Start ve Length kısımlarına başlangıç ve bitiş değerlerini ekleriz. Örneğin; “Programlama” diye bir string’imiz olsun. Start değerine 4 ve length değerine de 6 değerlerini eklersek çıktı olarak “ram” değerini alırız.

## Starts At (Başlangıç Pozisyonunu Bul)



Şekil 91: Starts At

Aranılan karakter veya karakterin string’in içinde kaçınıcı sıradan başladığını gösteren blok yapısıdır. Stringlerde pozisyon sırası 0’dan başlar. Eğer tek bir karakter aranıyorsa o karakteri ilk bulunduğu değeri döndürür. Ancak, bir karakter topluluğu veya bir kelimeyi arıyorsa ilk karakterlerinin başlangıç sırasını döndürür.

## Trim (Boşlukları Temizle)



Şekil 92: Trim

Trim bloğu string’in başında veya sonunda olan boşlukları kaldırır.

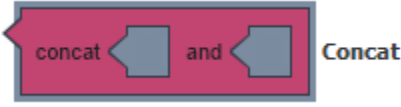
## To Text (Metne Çevir)



Şekil 93: To Text

Kimi durumlarda bazı değerleri metin değerine çevirmemiz gerekebilir. Örneğin, karşılaştırma operatörlerinde iki değişkeni karşılaştırıp eşit olup olmadığını bir metin kutusuna yazdırmak istediğimizde bunu yapamayız çünkü bu karşılaşmadan çıkan sonuç 1 (true) veya 0 (false) olur ve program da bunu algılamayabilir. Bu tür durumlarda To Text bloğu kullanılır ve çıkan sonucun metin kutusuna metin olarak yazdırmamızı sağlar.

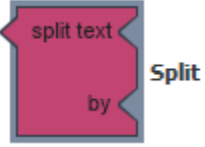
## Concat (Birleştir)



Şekil 94: Concat

Concat, eklenen iki string'in tek bir string hâline çevrilmesini sağlayan blok yapısıdır.

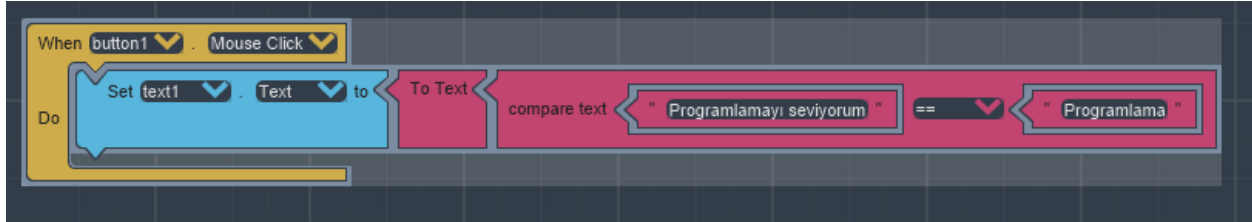
## Split (Ayrır)



Şekil 95: Split

Bir string'in istenildiği yerden parçalara ayrılmasını sağlayan blok yapısıdır. String ayrıldıktan sonra bir liste gibi olur. Girilen karakter veya karakterler string'in her yerinden kaldırılır. Örneğin; "Vitamin" kelimesinden "i" karakterini çıkardığımızda elde edeceğimiz çıktı üç elemanlı bir liste olacaktır. Elemanları da "V", "tam", "n" olur.

**Uygulama:** Verilen iki string'i karşılaştıralım ve sonucu ekrana yazdıralım.



Şekil 96

- 1- Butona tıkladığımızda yazdırmasını istediğimiz için Mouse Click bloğunu ekledik.
- 2- "Programlamayı seviyorum" ve "Programlama" değerlerine sahip iki string'i Compare Text bloğuna ekledik.
- 3- Sonuç 1 veya 0 olarak çıkacağından sonucun metne çevrilmesi gerekiyor, bu sebeple To Text bloğunu kullandık.

Buton

false

Şekil 97

## BÖLÜM 2 - FORM NESNELERİ

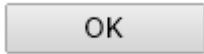
İnternet sitelerini ziyaret ettiğimizde bizden çoğu zaman kullanıcı girişi yapmamızı ister. Bunun için de önümüze kullanıcı adımızı ve şifremizin yazılacağı iki kutu ve yazdıktan sonra göndereceğimiz bir buton bulunur.

The image shows a login form with a light gray background. At the top, the text 'KULLANICI ADI' is written in bold. Below it is a white rectangular input field. Further down, the text 'ŞİFRE' is written in bold. Below it is another white rectangular input field. At the bottom right of the form is a dark gray button with the word 'GİRİŞ' in white capital letters.

Şekil 98

Bu tür kullanıcı ile web sayfası arasındaki veri alışverişini sağlayan yapılar HTML’de **form** olarak adlandırılmaktadır. Formlar kullanıcıdan veri girişi almamızı sağlar. “Input (Girdi)” en önemli form etiketidir. Bu etiket JavaScript kodu yazarken kullanılmak zorunda fakat VFabrika’da hazır olarak form nesnelere eklenebilmektedir. Input ile beraber birçok form nesnesini kullanabiliriz. Button (Buton), Image Button (Resim Butonu), Radio Button (Seçenek Butonu), Chechkbox (Kontrol Butonu), Dropdown List (Açılan Kutu), Textbox (Metin Kutusu), Slider (Kaydırıcı) VFabrika’da kullanılan form nesneleridir.

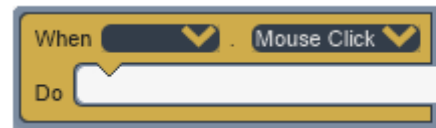
### Button (Buton)



Şekil 99

Buton, sahneye tıklanabilir bir düğme eklememizi sağlar. Böylelikle oluşturduğumuz blok veya blok yapılarını tetikleyebiliriz. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye buton ekleyebiliriz. Bloklar’a geçtiğimizde, bu nesneye özel blokları Button

penceresinde bulabiliriz.



On Mouse Event

On mouse event (Fare Olayı) bloğu ile Button’a tıklandığında hangi işlemleri yapacağımızı belirleriz.

**NOT:**

Şekil 101



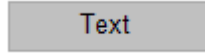
Şekil 100

**Set Property (Özellik Ayarla):** Sahneye eklediğimiz nesnelere tıkladığımızda, sağ altta Properties (Özellikler) isimli bir pencere görebiliriz. Bu pencerede tıkladığımız nesnenin özelliklerini değiştirebiliriz. Dizayn bölümünün dışında bloklar yardımı ile de nesnelerin özelliklerine ulaşabiliriz. Set Property (Özellik Ayarla), bloğun özelliğini dilediğimiz şekilde ayarlamamızı sağlayan kod bloğudur. Set kısmındaki ilk açılır pencereden istediğimiz bloğu seçip ikinci açılır pencereden de ayarlamak istediğimiz özelliği seçebiliriz.

**Get Property (Özellik Oku):** Nesnelerin özelliklerinin okunmasını sağlayan kod bloğudur. Örneğin, bir butonun yüksekliğini, font büyüklüğünü Get Property (Özellik Oku) yardımıyla rahatlıkla öğrenebiliriz.

Set Property (Özellik Ayarla) ve Get Property (Özellik Oku) blokları tüm form nesnelerinde yer alır ve hepsinde de o nesneye özel olarak bulunur. Örneğin; bloklar bölümünde, Button penceresindeki bir Set Property bloğunu eklediğimizde bu bloğu başka herhangi bir nesne için kullanamayız.

### Image Button (Resim Butonu)



Şekil 102

Resim butonu tıpkı button gibi sahneye tıklanabilir bir düğme eklememizi sağlayan form nesnesidir. Butondan farklı olarak standart bir görüntü yerine resim ekleyebilir ve öyle kullanabiliriz. Bunun için de sahnemize eklediğimiz resim butonuna tıklayarak özellikler penceresine gidip “Durum Resim

Url’leri” başlığından resmi seçip ekleyebiliriz.

Ayrıca, resim butonuna tek bir resim ekleyebildiğimiz gibi Sprite (Resim Grubu) da ekleyebiliriz.


	<p>On Mouse Event (Fare Olayı) ile resim butonuna tıklandığında yapılmasını istediğimiz blokları çalıştırabiliriz.</p>
	<p>Text Align (Metin Konumu) bloğu ile resim butonunda bulunan metnin pozisyonunu ayarlayabiliriz.</p>

## Radio Button (Seenek Butonu)



Birok seenek iinden tek bir seeneėin seilmesine imkân veren form nesnesidir. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye istenildiėi kadar radio buton ekleyebiliriz.

Şekil 103

	<p>Radio Button'larda en önemli özelliklerden biri seilip seilmediėinin kontrol edilmesidir. Bloklar'da On Check Event (Seilme Olayı) bloėunu bu kontrolü sağlamak için kullanabiliriz. Onun dıřında da Get Property (Özellik Oku) bloėu ile Radio Button'ın seilip seilmediėini öğrenebiliriz.</p>
---	--

**NOT:** Radio button denmesinin sebebi, eski radyolardaki düėmelere benzeyiři ve iřleyiř řeklinin aynı olmasıdır. Eski radyolarda da bir düėmeye basıldıėında eėer basılı bir düėme varsa iptal olurdu ve yalnızca tek bir düėme basılı olarak kalırdı.


## CheckBox (Kontrol Butonu)



Seenekler iinden birden fazla seeneėin seilmesine olanak saėlayan form nesnesidir. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye ekleyebiliriz.

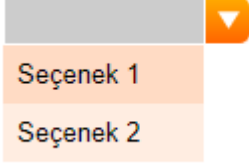
Şekil 104

Radio Button ile en önemli farkı, radio button yalnızca tek bir seeneėin seilmesine imkân verirken checkbox ile birden fazla seeneėin seilebilmesidir.

	<p>Radio Button'da olduėu gibi, checkbox nesnesinde de seilip seilmediėini blok'larda bulunan Checkbox bařlıėı altında On Check Event (İřaretlenme Olayı) veya Get Property (Özellik Ayarla) blokları yardımıyla yapabiliriz.</p>
---	---



## Dropdown List (Açılan Kutu)



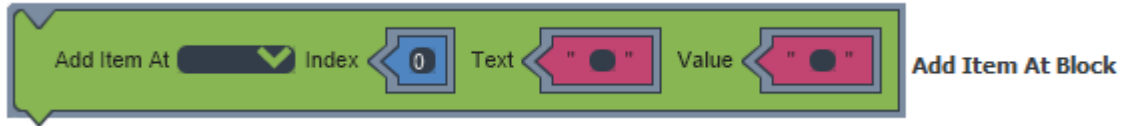
Şekil 105

Açılır kapanır bir menü oluşturmamızı ve içinde çeşitli seçenekler barındırmamızı sağlayan form nesnesidir. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye ekleyebiliriz. Açılan kutunun içine öğe eklemesini, dizayn bölümünden veya bloklar aracılığı ile yapabiliriz. Dizayn bölümünden açılır listeye tıkladığımızda özellik menüsü çıkacaktır. Orada “Nesneler” kısmındaki üç noktaya tıklayarak seçeneklerimizi

ekleyebiliriz. Bloklar ile de bunu yapabiliriz.



Add Item Block (Öge Ekle Bloğu) ile açılan kutuya öğe ekleyebiliriz.



Add Item At Block (Ögeyi İndekse Ekle Bloğu) ile de ekleyeceğimiz ögenin hangi sırada yer alacağını belirtebiliriz.

Öge ekleyebildiğimiz gibi var olan öğeleri de kaldırabiliriz. Bunu hem özellikler menüsünden hem de bloklar ile yapabiliriz.



Remove Item Block (Öge Çıkar Bloğu) en sondaki öğeyi siler.

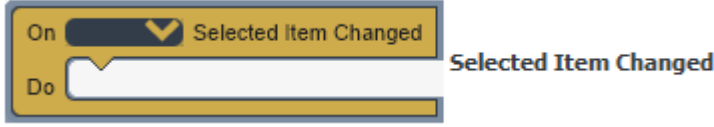


Remove Item At Block (Ögeyi İndeksten Çıkar Bloğu) girdiğimiz index sayısına göre o sırada bulunan seçeneği siler.



Clear (Temizle) bloğu, seçilmiş olan bir öğe varsa bu seçimi geri alır ve listeyi varsayılan hâline geri getirir.

Hangi seçeneğin seçildiğini öğrenebilmek için Radio Button ve CheckBox nesnelerinde de olduğu gibi Get Property (Özellik Ayarla) bloğunu kullanabiliriz. Bu blokta bulunan *Item*, *Text*, *Value*, *Index* değerlerinden birini seçerek ona göre de kontrolü sağlayabiliriz.

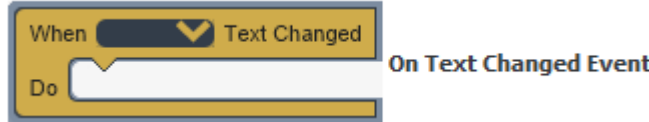



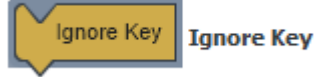
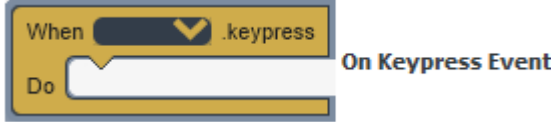
	<p>Selected Item Changed (Seçilen Nesne Değişimi Olayı) ile seçenekleri seçtiğimizde neler yapılacağını belirleyebiliriz. Bu bloğa koyduğumuz bloklar seçenek değişir değişmez yapılıdır.</p>
---	---


## Textbox (Metin Kutusu)



Şekil 106

Kullanıcıdan metin girdisi almamızı sağlayan form nesnesidir. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye ekleyebiliriz. Metin Kutusuna özellikler menüsü ve bloklar aracılığı ile birtakım özellikler ekleyebiliriz.

	<p>On Text Changed Event (Metin Değişme Olayı) ile metin değiştiği zaman neler yapılacağını belirleyebiliriz.</p>
	<p>Highlight (Vurgu) bloğu ile vurgu rengini değiştirebiliriz.</p>
	<p>Check Validation (Doğrulamayı Kontrol Et) bloğu metin kutusunun doldurulmasının gerekli olduğu durumlarda kullanılır.</p>
	<p>Tooltip Position (İpucu pozisyonu) ile yazılan ipucunun metin kutusunun hangi tarafında olacağını belirtebiliriz.</p>
	<p>Ignore Key (Tuşu Yoksay) bloğu klavye ile veri girişini engeller.</p>
	<p>On Keypress Event (Tuşa Basılma Olayı) bloğu metin kutusunda klavyeden bir tuşa basıldığında neler yapılacağını belirlememizi sağlar.</p>

 <p>When [ ] . [ ] Do [ ]</p> <p>On Focus Event</p>	<p>On Focus Event (Odaklanma Olayı) bloğu metin kutusuna tıklandığında neler olacağını belirlememizi sağlar. Got Focus tıklandığında, Lost Focus ise tıklanma kaldırıldığında çalışır.</p>
--	--

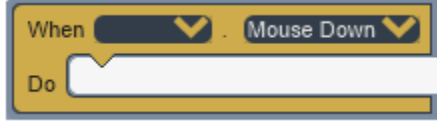

## Slider (Sürgü)



Şekil 107

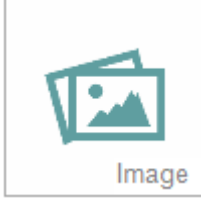
Slider nesnesi, sürgüyü kaydığımız zaman oluşturulan blok yapılarının çalışmasını sağlayan form nesnesidir. Dizayn bölümünde iken Input (Girdi) penceresinden sürükleyip bırak yöntemiyle sahneye ekleyebiliriz. Slider nesnesinde varsayılan olarak 100 bölme

bulunur. Bu da sürgüyü çok fazla kaydırabileceğimiz anlamında gelir. Bu sayıyı slider kullanma amacımıza göre azaltabilir veya arttırabiliriz. Dizayn bölümünde slider'a tıkladığımızda çıkan davranış özelliklerinden "En Büyük Değer", "En Küçük Değer", "Adım Büyüklüğü" değerleriyle değiştirebiliriz.

 <p>When [ ] . Mouse Down [ ] Do [ ]</p> <p>On Mouse Event</p>	<p>On Mouse Event (Fare Olayı) bloğu ile slider'a tıkladığında yapılmasını istediğimiz blokları çalıştırabiliriz.</p>
 <p>When [ ] . Value Changed [ ] Do [ ]</p> <p>On Value Change Event</p>	<p>On Value Change Event (Değer Değişme Olayı) bloğu ile de sürgü her kaydırıldığında, yani sürgünün değeri değiştiğinde, yapılmasını istediğimiz blokları çalıştırır. Slider'ın en büyük değerini, en küçük değerini ve adım büyüklüğünü davranış özelliklerinden değiştirebildiğimiz gibi Set Property bloğu ile de değiştirebiliriz.</p>

## BÖLÜM 3 – MEDYA NESNELERİ

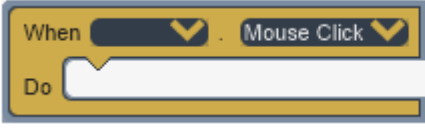


### Image (Resim)




Şekil 108

Yaptığımız projenin görünümünü değiştirmek veya yeni görseller eklemek için resim nesnesini kullanabiliriz. Dizayn bölümünde iken Medya penceresinden sürükle bırak yöntemiyle sahneye resim ekleyebiliriz. Sahneye bıraktığımız anda bizden eklemek istediğimiz resmin adresini isteyecektir, resmin konumunu seçtikten sonra resmimiz sahneye eklenir.

Ayrıca, resim nesnesinde Sprite Image (Resim Grubu) özelliğini kullanabiliriz. Sprite Image ile basit gif animasyonları hazırlanabilir.

	<p>On Mouse Event (Fare Olayı) bloğu ile eklenilen resme tıklandığında hangi işlemleri yapacağımızı belirleriz.</p>
	<p>On Drag Event (Sürükle) bloğu ile resmi sürüklediğimizde yapacağımız işlemleri belirleriz.</p>
	<p>On Drag and Drop Event (Sürükle Bırak) bloğu ile de resmi bıraktığımızda yapılacak işlemleri belirleyebiliriz.</p>

**NOT:** Bir resmi veya bir nesneyi sürükle bırak yapabilmek için, o nesne ile bırakılacağı yerin aynı Content (İçerik) içinde olması gerekmektedir. Content (İçerik) nesnesi Bölüm 4'te detaylı şekilde anlatılacaktır. Bunun yanında, resmin sürüklenabilir ve bırakılabilir özelliklerinin açık olması gerekmektedir. Özellikler penceresinden veya Set Property (Özellik Ayarla) bloğu ile bunları yapabiliriz.

	<p>On Animation Control Event (Animasyon Olayı) bloğu ile oluşturduğumuz resim animasyonlarını kontrol edebiliriz. Örneğin, animasyon başladığında, bittiğinde, durduğunda veya frame'i</p>
---	---



	değiştiğinde neler yapılacağı belirtilebilir.
	Oluşturulan animasyonu oynatmayı veya durdurmayı sağlayan blok yapısıdır.
	Animation GoTo (Animasyon Karesi Ayarla) bloğu oluşturulan animasyonun frame'ini değiştirip aynı zamanda oynatma veya durdurma işlemlerini yapar.

## Sound (Ses)



Projeye ses eklememizi sağlayan medya nesnesidir. Eklediğimiz sesleri kolaylıkla kontrol edebiliriz. Dizayn bölümünde iken Medya penceresinden sürükleyip bırak yöntemiyle sahneye eklediğimizde karşımıza çıkan pencereden ses dosyasının konumunu girerek ekleyebiliriz.

Şekil 109




	On Player Control Event (Oynatıcı Kontrolü Olayı) bloğu ile ses başladığında, durduğunda veya bittiğinde neler yapılacağını belirtebiliriz.
	Eklediğimiz ses nesnesinin oynatılmasını, durdurulmasını veya duraklatılmasını Player Control (Oynatıcı Kontrolü) bloğu ile yapabiliriz.

## Video



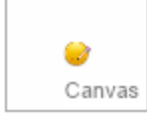
Şekil 110

Video nesnesi ile projemize videolar ekleyebiliriz. Video nesnesini sahneye eklediğimizde bizden poster ve video adresinin konumunu isteyen bir pencere çıkar, bu pencereden dosyamızın konumunu verebilir ve aynı zamanda videonun özelliklerinden olan “Tekrarla”, “Kontroller” ve “Otomatik Oynat” seçeneklerini de seçebiliriz.

 <p><b>On Player Control Event</b></p>	<p>On Player Control Event (Oynatıcı Kontrol Olayı) bloğu video nesnesini birçok şekilde kontrol etmemizi sağlayan blok yapısıdır. Video başladığında, durduğunda, bittiğinde, yüklendikten sonra ve oynarken hangi blokların çalıştırılacağını belirtebiliriz.</p>
 <p><b>Player Control</b></p>	<p>Eklediğimiz video nesnesinin oynatılmasını, durdurulmasını veya baştan başlatılmasını Player Control (Oynatıcı Kontrolü) bloğu ile yapabiliriz.</p>
 <p><b>Player GoTo</b></p>	<p>Player GoTo (Oynatıcıyı Götür) bloğu, istenilen saniyeye gidip videoyu oynatmayı veya durdurmayı sağlar.</p>

## BÖLÜM 4 – TASARIM NESNELERİ

### Canvas (Resim Kâğıdı)



Canvas (Resim Kâğıdı) sahnemize çizim yapmamızı sağlayan tasarım nesnesidir. Canvas nesnesinde çizimler bloklar yardımı ile yapılır. Dolayısıyla, sahneye eklenen canvas nesnesi bloklar ile yaptığımız çizimleri tutar.

Şekil 111

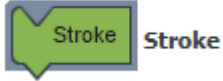
	<p>Context (Ortam) Çizim yapılacak canvas nesnesini seçmemizi ve aynı zamanda eklenen çizim bloklarının da döndürülmesini sağlayan blok yapısıdır.</p>
	<p>Begin Path (Yola Başla) ve Close Path (Yolu Bitir) blokları çizimin başlangıç ve bitiş yolunu belirtmemizi sağlar.</p>
	<p>Move To (Taşı) bloğu çizimin nereden başlayacağını belirtir fakat herhangi bir çizim oluşturmaz.</p>
	<p>Line To (Çizgi Çiz) bloğu noktaları belirterek çizgi çizmemizi sağlar.</p>
<p>Rect (Kare) bloğu girdiğimiz değerlere göre bir kare oluşturmamızı sağlar fakat sahnede görünmez.</p>	
<p>Stroke Rect (Boş Kare Çiz) içi boş bir kare çizmemizi sağlar.</p>	



Fill Rect (Dolu Kare Çiz) kare oluştururken içinin de dolu olmasını istediğimiz durumlarda kullanılabilir.



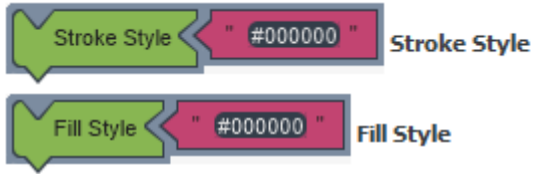
Clear Rect (Alanı Temizle) bloğu ile girilen koordinat, yükseklik ve genişliğe göre alanı temizler.



Stroke(Çizgi) bloğu ile oluşturduğumuz karede eğer çizgiler yoksa çizgi eklememizi sağlar. Örneğin; Rect (Kare) bloğu eklediğimizde sahnede bir kare görünmediğinden bahsetmiştik, stroke bloğu ile rect bloğuna çizgi ekleyebiliriz.



Fill (Doldur) bloğu ile oluşturulan karelerin içini doldurabiliriz.



Stroke Style (Çizgi Stili) ve Fill Style (Dolgu Stili) blokları ile çizgi ve dolgulara renk verebiliriz. Renkleri kelime (red, green, black gibi) ile ya da RGB renk kodları ile belirtebiliriz.

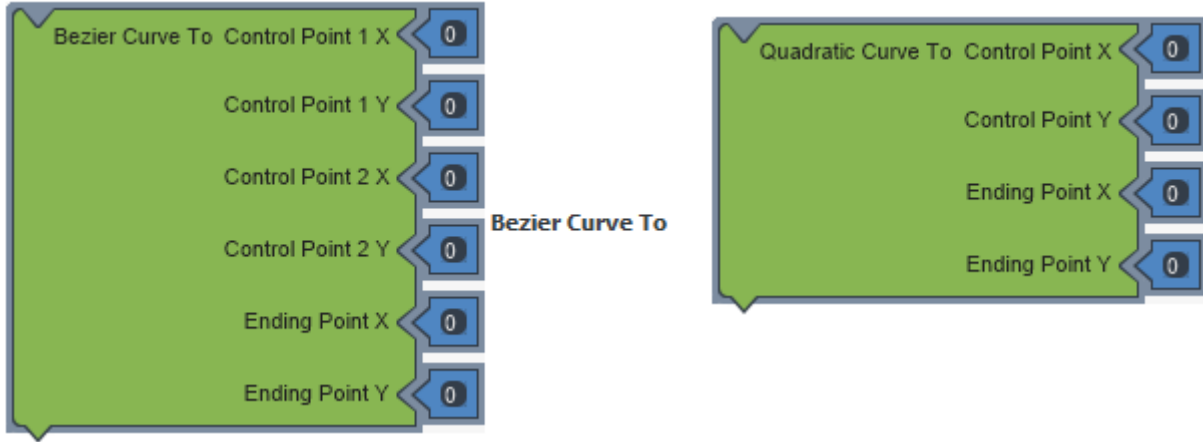


Line Width (Çizgi Kalınlığı) bloğu ile çizgilerin kalınlığını değiştirebiliriz.

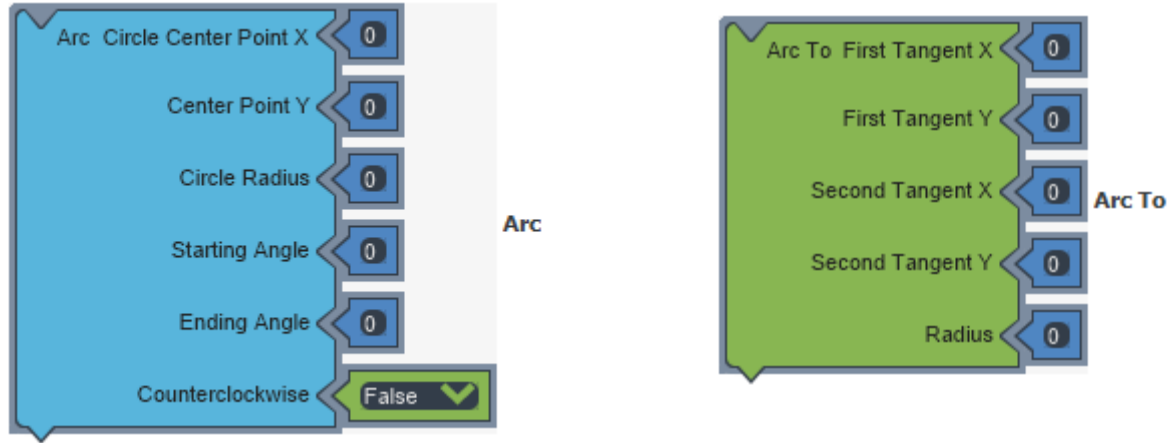


Line Cap (Çizgi Ucu) bloğu çizilen çizgilerin ucunun nasıl olacağını belirler. Kare, yuvarlak ve kapaksız şeklinde üç farklı dizayn verebiliriz.

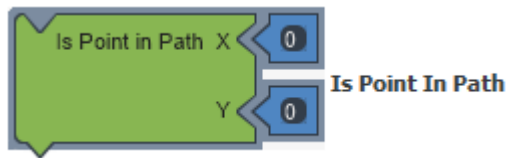




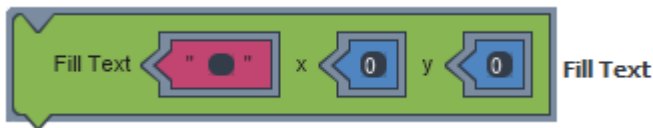
Quadratic Curve (Kuatratik Yay) ve Bezier Curve (Yönlü Bezier Yayı) girilen parametrelere göre bezier ve kuadratık yaylar çizmemizi sağlayan bloklardır.



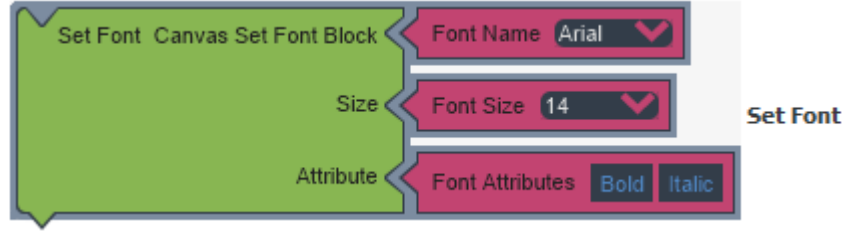
Arc (Yay) ve Arc To (Yönlü Yay) da girilen parametrelere göre yay oluşturmamızı sağlayan bloklardır.



Is Point in Path (Nokta Yolun İçinde Mi) bloğu, koordinatlarını girdiğimiz noktanın, oluşturduğumuz yolda ise true (doğru), değilse false (yanlış) değerini döndürür.



Fill Text (Dolgu Metni) bloğu, girilen pozisyona göre oraya renkli dolgulu metin girilmesini sağlar.



Set Font (Yazı Tipini Ayarla) bloğu ile canvas içindeki metinlerin stilini büyüklüğünü ve özelliklerini ayarlayabiliriz. Tek bir blokta yapabildiğimiz gibi ayrı ayrı bloklarda da yapabiliriz.

<p>Set Font Name (Yazı Tipini Ayarla)</p>	Set Font Name (Yazı Tipini Ayarla)
<p>Set Font Size (Yazı Tipi Boyutunu Ayarla)</p>	Set Font Size (Yazı Tipi Boyutunu Ayarla)
<p>Set Font Attribute (Yazı Tipini Ayarla)</p>	Set Font Attribute (Yazı Tipini Ayarla)

## Content (İçerik)



Şekil 112

Content (İçerik), sahneye eklenen nesneleri gruplandırmamızı sağlar. Dolayısıyla bir content nesnesinin özelliğini değiştirdiğimizde, örneğin gizlediğimizde, içindeki tüm nesnelere uygulamış oluruz. Buna ek olarak, Drag-Drop (Sürükle Bırak) yapılacak nesnelerin de aynı content içinde bulunmaları gerekmekte. Eğer drag yapılacak obje bir content'in içinde ise drop edileceği yer de aynı content'te olmalı.

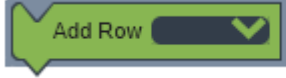


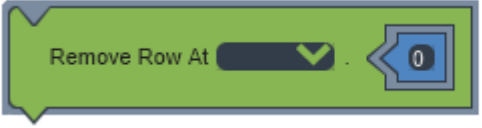
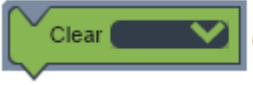


	On Drag Drop Event (Sürükle Bırak Olayı) bloğu ile nesne bırakıldığında neler yapılacağını belirleyebiliriz.
	On Drag Event (Sürükle Olayı) bloğu ile nesne sürüklenmeye başladığında neler yapılacağını belirleyebiliriz. Sürüklenme sürecinde dört farklı seçenek vardır. Bunlar; Drag Begin (Sürüklemeye Başladığında), Drag Move (Sürükleme Hareketinde), Drag Ending (Sürükleme Biterken), Drag End (Sürükleme Bittiğinde).
	Get Drag Object (Sürüklenen Nesneyi Oku) bloğu sürüklenen nesnenin tüm özelliklerini (adını, pozisyonunu, görünürlüğünü vs.) öğrenmemizi sağlar.
	Get Drag Object Name (Sürüklenen Nesnenin Adını Oku) bloğu Get Drag Object'ten farklı olarak sürüklenen nesnenin yalnızca adını öğrenmemizi sağlar.
	Get Drop Object (Bırakılan Nesneyi Oku) bloğu, bırakılan nesnenin tüm özelliklerini (adını, pozisyonunu, görünürlüğünü vs.) öğrenmemizi sağlar.
	Get Drop Object Name (Bırakılan Nesnenin Adını Oku) bloğu, Get Drop Object'ten farklı olarak bırakılan nesnenin yalnızca adını öğrenmemizi sağlar.

## Table (Tablo)



Şekil 113

Tablo nesnesi, sahnemize yeni tablolar eklememizi ve içinde de istediğimiz verilerin listelenmesini sağlar. Tabloyu dilediğimiz gibi biçimlendirebiliriz, böylelikle tasarımını da belirleyebiliriz. Dizayn bölümünde bulunan özellikler penceresinden tabloya sütun ekleyebilir, satır yüksekliklerini ayarlayabilir ve diğer özelliklerini değiştirebiliriz.

	Add Row (Satır Ekle) bloğu ile tabloya bir satır ekleyebiliriz.
	Add Row At (Pozisyona Satır Ekle) bloğu, tabloda girilen konuma satır ekler.
	Remove Row (Satır Sil) bloğu ile tabloda en altta bulunan satırı silebiliriz.
	Remove Row At (Pozisyonadaki Satırı Sil) bloğu, tabloda istenilen pozisyonadaki satırı siler.
	Clear (Temizle) bloğu; tablodaki satır, sütun ve verileri siler.
	
	

Set Column By Index (Pozisyonadaki Sütun Metnini Ayarla) bloğu, girilen pozisyonadaki metni değiştirmemizi, metin yoksa da eklememizi sağlar. Tablodaki son satıra ekler.

Set Column By Name (Sütun Metnini Ayarla) bloğu da sütunun ismini girerek veri eklememizi veya değiştirmemizi sağlar. Tablodaki son satıra ekler.

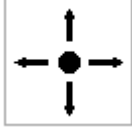


Set Column At Row (Pozisyonadaki Sütun Metnini Ayarla) bloğu ile girilen satır ve pozisyona istenilen veriyi eklememizi sağlar.

**NOT:** Satırların (Row) pozisyonu 1'den başlıyor fakat sütunların (Column) pozisyonu 0'dan başlıyor.

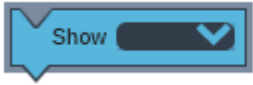

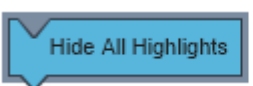
	<p>Scroll To Bottom Row (Son Satıra Kaydır) bloğu, tablonun aşağıdaki satırlarına gitmemizi sağlar.</p>
	<p>Set Footer Text (Alt Başlığı Ayarla) bloğu, alt başlığının metnini değiştirmemizi sağlar.</p>
	<p>Set Selected Cell (Seçili Hücreyi Ayarla) bloğu ile satır ve sütunu girilen hücreyi işaretleyebiliriz.</p>
	<p>Get Selected Column Index (Seçili Sütun Pozisyonu) bloğu, seçili olan bir hücre varsa onun sütununu öğrenmemizi sağlar.</p>
	<p>Get Selected Row Index (Seçili Satır Pozisyonu) bloğu, seçili olan bir hücre varsa onun satır bilgisini öğrenmemizi sağlar.</p>
	<p>Clear Selection (Seçimi Temizle) bloğu, seçili olan bir hücre varsa ondaki seçimi kaldırmamızı sağlar.</p>

## Highlight (Vurgu)



Şekil 114

Highlight (Vurgu) nesnesi, sahnede dikkat çekilmesini istediğimiz yer veya nesnelerde kullanılabilir. Örneğin, metin kutusuna veri girilmesini istediğimizde vurgu ekleyerek kullanıcının dikkatini metin kutusuna çekebiliriz. VFabrika'da birçok farklı vurgu türü vardır; dikkat çekilmesi istenen nesneye göre vurgu türünü seçebiliriz. Vurgu türlerine dizayn bölümündeki özellikler penceresinden ulaşabiliriz.

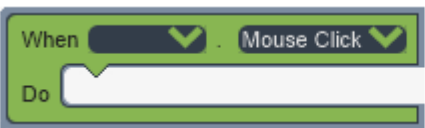




 Show	Show (Göster) bloğu, vurgu nesnesinin sahnede gösterilmesi sağlar.
 Hide	Hide (Gizle) bloğu, vurgu nesnesinin sahnede gizlenmesini ve gösterilmemesini sağlar.
 Hide All Highlights	Hide All Highlights (Tüm Vurguları Gizle) bloğu ile Hide (Gizle) nesnesinden farklı olarak sahnede tüm vurgu nesnelerini gizleyebiliriz.

## Primitive Objects (Temel Nesneler)



Şekil 115

Primitive Objects (Temel Nesneler) sahneye geometrik şekiller eklememizi sağlar. Bu şekiller; elips, dikdörtgen, yuvarlatılmış dikdörtgen ve üçgendir. Temel nesnelerdeki geometrik şekiller statiklerdir.

 On Mouse Event	On Mouse Event (Fare Olayı) bloğu ile temel nesnelere tıklandığında neler yapılacağını belirleyebiliriz.
 Get Fill Color	Get Fill Color (Dolgu Rengini Oku) bloğu, nesnelerin içini dolduran dolgu renklerini okumamızı sağlar.
 Set Fill Color	Set Fill Color (Dolgu Rengini Ayarla) bloğu, dolguların rengini seçmemizi sağlar.
 Get Stroke Color	Get Stroke Color (Çizgi Rengini Oku) bloğu, temel nesnelerin çizgilerinin hangi renk olduğunu öğrenmemizi sağlar.
 Set Stroke Color	Set Stroke Color (Çizgi Rengini Ayarla) bloğu, temel nesnelerin çizgi rengini düzenler.

	Get Stroke Thickness (Çizgi Kalınlığını Oku) ile temel nesnelerin sahip oldukları çizgi kalınlıklarını öğrenebiliriz.
	Set Stroke Thickness (Çizgi Kalınlığını Ayarla) bloğu ile temel nesnelerin çizgi kalınlıklarını istediğimiz değerde yapabiliriz.
	Get Corner Radius (Köşe Yarıçapını Oku) bloğu ile temel geometrik şekillerin köşe yarıçaplarını öğrenebiliriz.
	Set Corner Radius (Köşe Yarıçapını Ayarla) bloğu ile temel geometrik şekillerin köşe yarıçaplarını ayarlayabiliriz.

## Smart Objects (Akıllı Nesneler)

VFabrika’da iki tür akıllı nesne vardır: Parallax Scroll (Paralaks Kaydırma) ve Polygon (Çokgen).

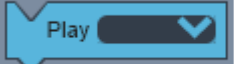



### Parallax Scroll (Paralaks Kaydırma)

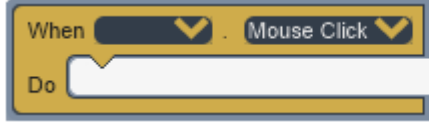


Şekil 116

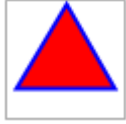
Parallax Scroll (Paralaks Kaydırma) belirli bir hızda akan resimler oluşturmamızı sağlayan akıllı nesnedir. Örneğin; sürekli hareket eden bir yol ve etrafındaki ağaçları paralaks vasıtasıyla kolaylıkla yapabiliriz. Bunun için, yol resmini ve ağaç resmini paralaksın içine atıp başlatmamız yeterlidir.

Sahnemizdeki paralaks nesnesine tıkladığımızda özellikler penceresinde “Kayan Görseller” yazısının yanında bulunan üç noktaya tıkladığımızda kaymasını istediğimiz resimleri ekleyebiliriz.

	Play (Oynat) bloğu ile paralaksı başlatabiliriz.
	Stop (Durdur) bloğu ile kayan paralaks resimlerini durdurabiliriz.
	Set Speed (Hızı Ayarla) bloğu ile paralaksın hızını ayarlayabiliriz.
	Set Direction (Yönü Ayarla) bloğu ile paralaksın kayacağı yönü seçebiliriz. Paralaks nesnesi varsayılan olarak Sol tarafa doğru kayar.









	<p>On Mouse Event (Fare Olayı) bloğunda, paralaks nesnesine tıklandığında neler yapılacağını belirleyebiliriz.</p>
---	--

## Polygon (Çokgen)



Polygon (Çokgen), sahneye geometrik şekiller eklememizi sağlayan akıllı nesnedir. Primitive Objects'ın (Temel Nesneler) statik olduğu bir önceki başlıkta belirtilmişti. Çokgenler dinamiktir. Dolayısıyla köşe sayıları ve başlangıç açıları değiştirilebilir.

Şekil 117

	<p>Get Edge Count (Köşe Sayısını Oku) bloğu, çokgenin kaç köşeye sahip olduğunu gösterir.</p>
	<p>Set Edge Count (Köşe Sayısını Ayarla) bloğu, çokgenin köşe sayısını değiştirmemizi sağlar. Böylelikle üçgen, beşgen, yedigen gibi geometrik şekiller oluşturabiliriz.</p>
	<p>Get Starting Angle (Başlangıç Açısını Oku) bloğu ile çokgenin başlangıç açısını öğrenebiliriz.</p>
	<p>Set Start Angle (Başlangıç Açısını Ayarla) bloğu ile çokgenin başlangıç açısını değiştirebiliriz.</p>
	<p>Get Fill Color (Dolgu Rengini Oku) bloğu, çokgenlerin içini dolduran dolgu renklerini okumamızı sağlar.</p>
	<p>Set Fill Color (Dolgu Rengini Ayarla) bloğu, dolguların rengini seçmemizi sağlar.</p>
	<p>Get Stroke Color (Çizgi Rengini Oku) bloğu, çokgenlerin çizgi renklerini okur.</p>
	<p>Set Stroke Color (Çizgi Rengini Ayarla) bloğu, çokgenlerin çizgi rengini düzenler.</p>



 <b>Get Stroke Thickness</b>	Get Stroke Thickness (Çizgi Kalınlığını Oku) ile çokgenlerin çizgi kalınlıklarını öğrenebiliriz.
 <b>Set Stroke Thickness</b>	Set Stroke Thickness (Çizgi Kalınlığını Ayarla) bloğu ile çokgenlerin çizgi kalınlıklarını ayarlayabiliriz.
 <b>On Mouse Event</b>	On Mouse Event (Fare Olayı) bloğunda, çokgene tıklandığında neler yapılacağını belirleyebiliriz.

## BÖLÜM 5 – ARAÇLAR

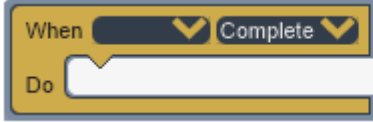

### Delay (Erteleyici)



Şekil 118

Belirlenen bir süre sonunda bazı kod bloklarının çalıştırılması gerekebilir. Böyle bir durumda Delay (Erteleyici) nesnesini kullanırız. Örneğin, 5 saniye sonra ekranda video oynatılmasını istiyorsak delay nesnesi ile bunu kolaylıkla yapabiliriz. Delay nesnesinde en önemli unsurlardan biri Interval (Süre) özelliğidir. Özellikler penceresinden Interval değerini ayarlayabiliriz. Bir diğer önemli özellik ise otomatik başlatma özelliğidir. Varsayılan olarak False (Yanlış) gelmektedir. Delay nesnesini, proje başlarken çalıştırmak istiyorsak True (Doğru) yapmalıyız.

**NOT:** Delay nesnesinde varsayılan süre 1000 milisaniyedir bu da 1 saniyeye denk gelmektedir.

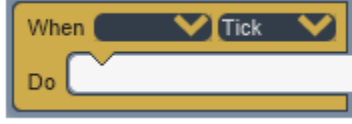

 <p>On Delay Event</p>	<p>On Delay Event (Erteleyici Olayı) bloğu ile delay nesnesi tamamlandığında, başladığında veya durduğunda neler yapılacağını belirleyebiliriz.</p>
 <p>Delay Control</p>	<p>Delay Control (Erteleyici Kontrolü) bloğu ile delay nesnesini başlatabilir veya durdurabiliriz.</p>

### Timer (Zamanlayıcı)



Şekil 119

Projeye zamanlayıcı eklememizi sağlayan araçtır. Delay nesnesinde olduğu gibi timer'da da Interval (süre) ve otomatik başlatma özellikleri önemlidir. Interval zamanlayıcının hızını belirtir. Örneğin, interval değeri 3000 milisaniye olan bir zamanlayıcı 3 saniyede bir işler.


 <p>On Timer Event</p>	<p>On Timer Event (Zamanlayıcı Olayı) bloğu ile timer nesnesi başladığında, durduğunda veya süre her işlediğinde neler yapılacağını belirleyebiliriz.</p>
 <p>Timer Control</p>	<p>Timer Control (Zamanlayıcı Kontrolü) bloğu ile timer nesnesini başlatabilir veya durdurabiliriz.</p>

## Trigger (Tetikleyici)



Trigger nesnesi, bulunduğu anahtar karede çalışmaya başlar. Projede trigger bulunan anahtar kareye gidildiğinde, trigger bloğundaki kod blokları tetiklenir ve çalışırlar.

Şekil 120

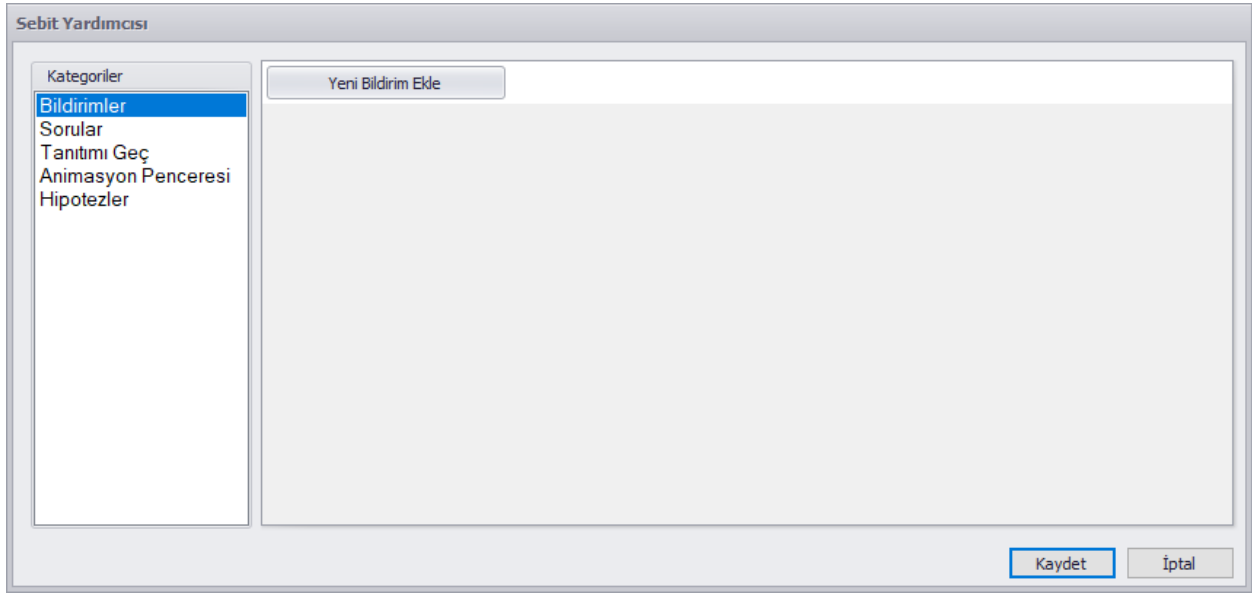
 <p>On Invoke Event</p>	<p>On Invoke Event (Tetiklenme Olayı) bloğu, sahneye yerleştirilen trigger tetiklendiğinde neler yapılacağı belirlememizi sağlar.</p>
--	---

## BÖLÜM 6 – SEBİT FRAMEWORK

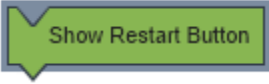
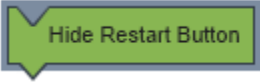

### Sebit Framework


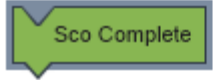
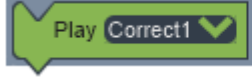

Sebit Framework, özellikle eğitsel içerikler geliştiren kullanıcılar için oldukça faydalı blokları barındırır. Örneğin, soru doğru cevaplandığında doğru sesi dinletebilmek, soru gösterme veya aktivite yapıldığında küçük bir tamamlandı animasyonu göstermek bu bloklar aracılığıyla projeye eklenebilir. Böylelikle projemizdeki görsel ve işitsel etkileşimleri arttırabiliriz.

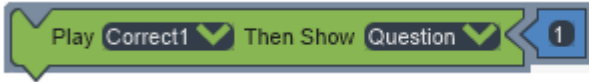
Sebit Framework'ü kullanabilmek için araç çubuğunda bulunan Sebit > Yardımcı adımlarını izlediğimizde karşımıza *Sebit Yardımcısı* isimli pencere çıkacaktır. Bu pencere aracılığı ile projemize bildirim, soru, tanıtım, animasyon penceresi ve hipotez ekleyebiliriz.



Şekil 121

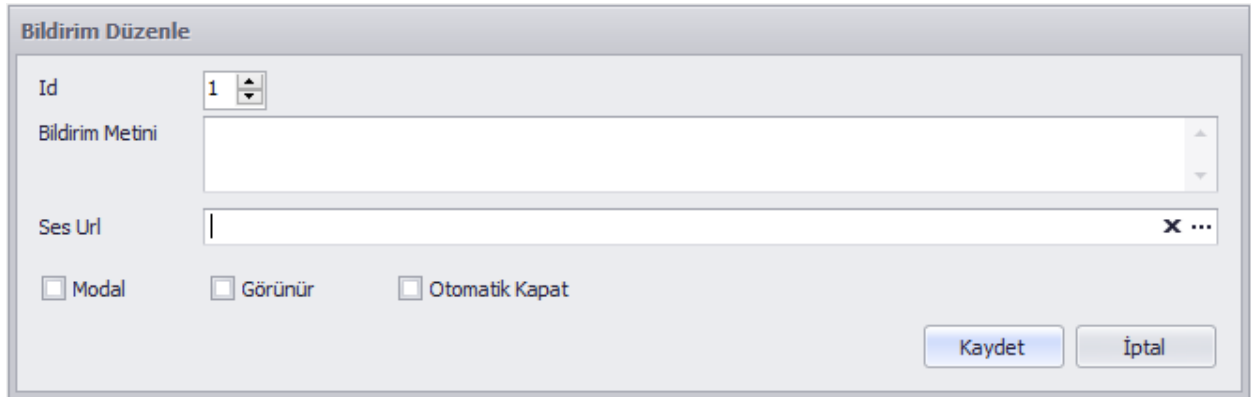
 <b>Show Restart Button</b>	Show Restart Button (Yeniden Başlat Düğmesini Göster) bloğu ile ekranımızın sol alt köşesine yeniden başlat butonu ekleyebiliriz.
 <b>Hide Restart Button</b>	Hide Restart Button (Yeniden Başlat Düğmesini Gizle) bloğu ile ekranda eğer bir yeniden başlat düğmesi varsa onu gizleyebiliriz.
 <b>On Restart Event</b>	On Restart Event (Yeniden Başlat Olayı) bloğu, sahnemizdeki yeniden başlat butonuna tıklandığında neler yapılacağını belirlememizi sağlar.

 <b>Show Logo</b>	Show Logo (Logoyu Göster) bloğu, sahnenin sol üst tarafına bir <i>Vitamin</i> logosu eklememizi sağlar.
 <b>Sco Complete</b>	Sco Complete (Sco Tamamlandığında) bloğu, ekranda <i>tamamlandı</i> animasyonu oynatmamızı sağlar.
 <b>Play Notify Sound Solo</b>	Play Notify Sound Solo (Uyarı Sesini Solo Çal) bloğu yalnızca sesin çalınmasını sağlar.
 <b>Notify Sound Hide Event</b>	Notify Sound Hide Event (Uyarı Sesi Bitiş Olayı) bloğu ile uyarı sesi bittiğinde neler yapılacağını belirleyebiliriz.

 <b>Play Notify Sound</b>	Play Notify Sound (Uyarı Sesi Çal) bloğu, istenilen bir doğru veya yanlış sesi çaldıktan sonra soru veya geri bildirim gösterilmesini sağlar. Üç tane doğru sesi ve üç tane de yanlış sesi içerir.
--	--

## Feedbacks (Bildirimler)

*Sebit Yardımcısı* penceresinde Bildirimler başlığında *Yeni Bildirim Ekle* butonuna tıkladığımızda karşımıza *Bildirim Düzenle* penceresi gelecektir.



**Bildirim Düzenle**

Id: 1

Bildirim Metni:

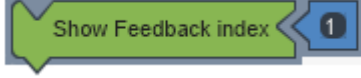
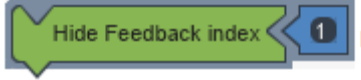
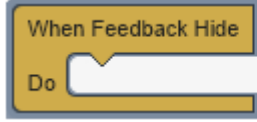
Ses Url:  X ...

☐ Modal ☐ Görünür ☐ Otomatik Kapat

Kaydet İptal

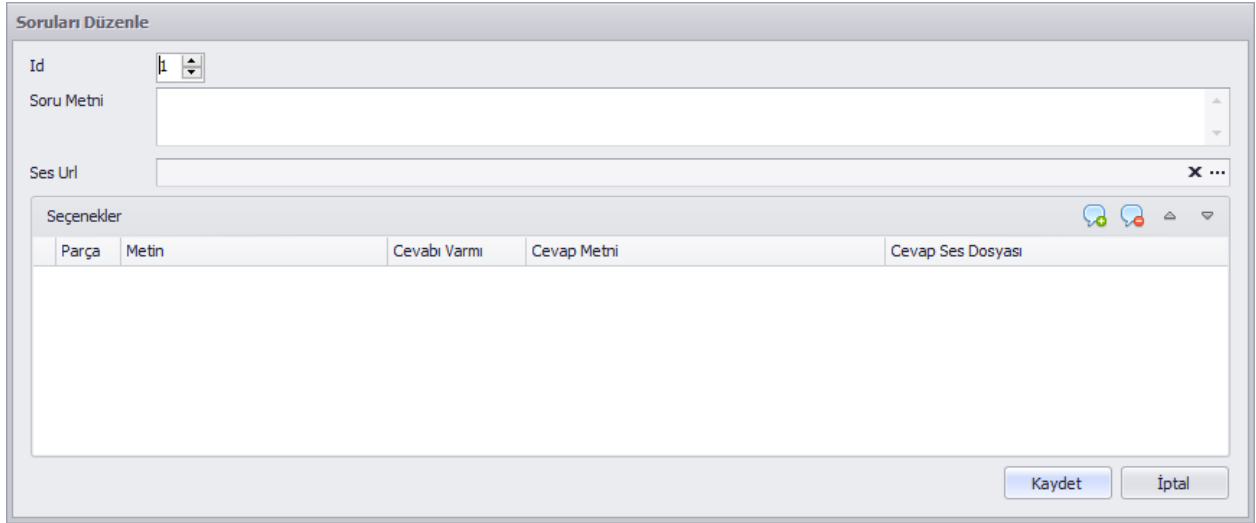
Şekil 122

Bu pencereden eklenecek bildirimi ayarlayabiliriz. Burada Id numarasına dikkat edilmesi gerekiyor. Bloklar ile bildirim ulaşmak istediğimizde bu Id numarasını kullanırız. Bildirimi oluştururken Modal, Görünür, Otomatik Kapat özelliklerini ayarlayabilir ayrıca bildirim ses de ekleyebiliriz.

	Show Feedback (Geri Bildirim Göster) bloğu, eklediğimiz bildirimin sahnede gösterilmesini sağlar. Index kısmında hangi bildirimin gösterileceği belirtilir.
	Hide Feedback (Geri Bildirim Gizle) bloğu, eklediğimiz bildirimin sahnede gösterilmemesini sağlar.
	On Feedback Hide Event (Geri Bildirim Gizlenme Olayı) bloğu ile bildirim sahnede gizlendiğinde neler yapılacağını belirtebiliriz.

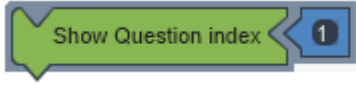
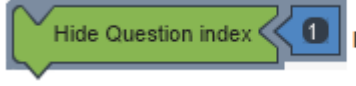
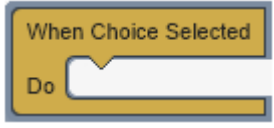
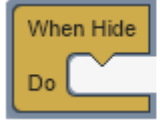
## Questions (Sorular)

*Sebit Yardımcısı* penceresinde Sorular başlığında *Yeni Soru Ekle* butonuna tıkladığımızda karşımıza *Soruları Düzenle* penceresi gelecektir.



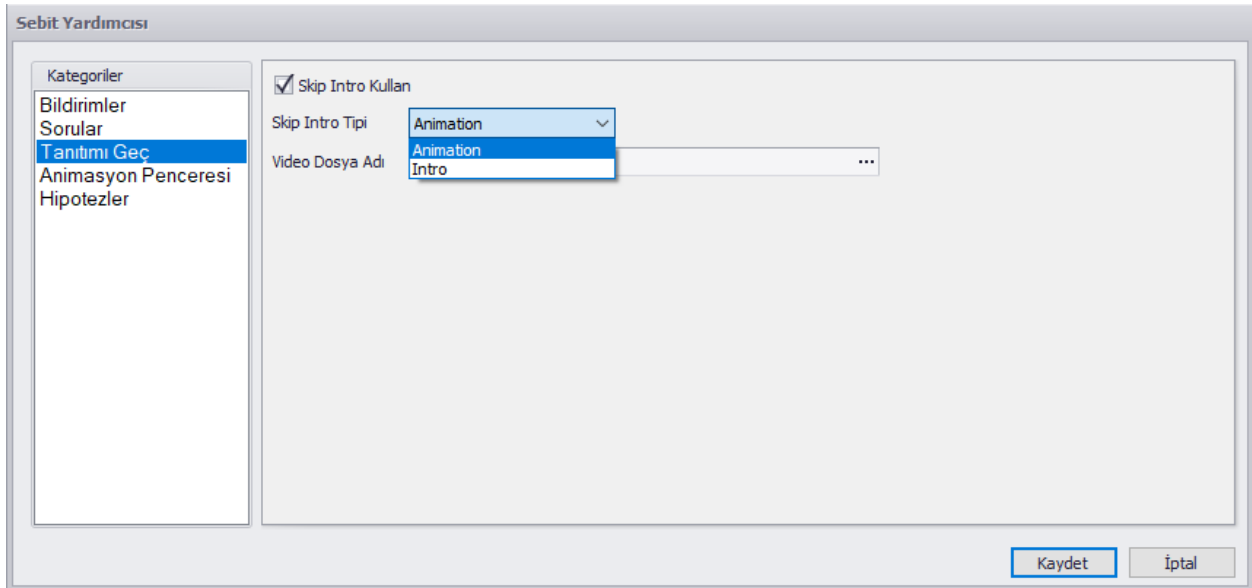
Şekil 123

Bu pencere aracılığı ile soru ekleyebiliriz. Bloklar ile soruya ulaşmak istediğimizde Id numarası kullanırız. Seçenekler penceresinde istenilen kadar seçenek soruya ekleyebiliriz. Soruya ses ekleyebildiğimiz gibi doğru cevaba da ses ekleyebiliriz.

 <b>Show Question</b>	Show Question (Soruyu Göster) bloğu, oluşturduğumuz sorunun sahnede gösterilmesini sağlar. Index kısmında hangi sorunun gösterileceği belirtilir.
 <b>Hide Question</b>	Hide Question (Soruyu Gizle) bloğu, oluşturduğumuz sorunun sahnede gizlenmesini sağlar.
 <b>On Question Choice Event</b>	On Question Choice Event (Soru Seçenek Olayı) bloğu ile seçenekler seçildiğinde neler yapılacağını belirleyebiliriz.
 <b>On Question Hide Event</b>	On Question Hide Event (Soru Gizleme Olayı) bloğu ile soru gizlendiğinde neler yapılacağını belirtebiliriz.

### Skip Intro (Tanıtımı Geç)

*Sebit Yardımcısı* penceresinde Tanıtımı Geç başlığına tıkladığımızda projemize kolaylıkla bir tanıtım videosu ekleyebiliriz. Eklediğimiz tanıtım projemizden önce sahneye gelir.



Şekil 124

## Animation Popup (Animasyon Penceresi)



Şekil 125

Animasyon Penceresi projemize video ve animasyonlar eklememizi sağlar. Dizayn bölümünde Sebit Framework başlığından sürükleyip bırak yöntemiyle sahnemize ekleyebiliriz. Animasyonu, *Sebit Yardımcısı* aracılığı ile ekleriz fakat öncesinde animasyon penceresinin sahneye eklenmiş olması gerekmektedir.

*Sebit Yardımcısı* penceresinde Animasyon Penceresi başlığında *Yeni Animasyon Penceresi Ekle* butonuna tıkladığımızda karşımıza *Animation Popup Düzenle* penceresi gelecektir.

Şekil 126

Açılan pencere aracılığıyla projemize video ekleyebiliriz. Eklediğimiz videoya Id numarası ile bloklardan ulaşabiliriz. Videoya ayrıca Modal, Kontroller, Otomatik Kapat/Oynat, Kapat Butonu, Çerçeve özelliklerini ekleyebilir ve genişlik ile yüksekliğini de belirleyebiliriz.

	<p>Show Animation Popup (Pencere Animasyonu Göster) bloğu, eklediğimiz animasyonun sahnede gösterilmesini sağlar. Index kısmında hangi animasyonun gösterileceği belirtilir.</p>
	<p>Hide Animation Popup (Pencere Animasyonunu Gizle) bloğu, eklediğimiz animasyonun sahnede gizlenmesini sağlar.</p>





On Animation Popup Event

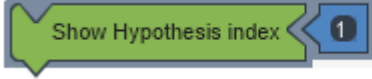
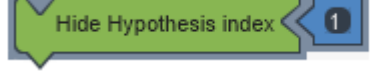
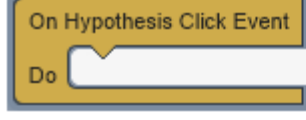
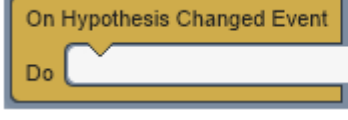
On Animation Popup Event (Pencere Animasyon Olayı) bloğu ile eklediğimiz animasyon sonlandığında (Video Ended) veya kapatıldığında (Video Close) neler yapılacağını belirleyebiliriz.

## Hypothesis (Hipotezler)

*Sebit Yardımcısı* penceresinde Hipotezler başlığında *Yeni Hipotez Ekle* butonuna tıkladığımızda karşımıza *Hipotezi Düzenle* penceresi gelecektir.

Şekil 127










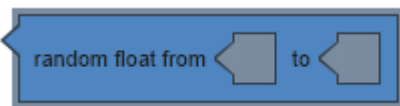
Hipotez Düzenle penceresi ile projemize hipotez ekleyebiliriz. Eklediğimiz hipoteze Id numarası ile bloklardan ulaşabiliriz.

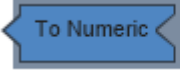
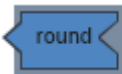

 <b>Show Hypothesis</b>	<p>Show Hypothesis (Hipotezi Göster) bloğu, eklediğimiz hipotezin sahnede gösterilmesini sağlar. Index kısmında hipotezin Id numarasını belirtilir.</p>
 <b>Hide</b>	<p>Hide Hypothesis (Hipotezi Gizle) bloğu, eklediğimiz hipotezin sahnede gizlenmesini sağlar.</p>
 <b>On Hypothesis Click Event</b>	<p>On Hypothesis Click Event (Hipotez Tıklama Olayı) bloğu ile hipoteze tıklanıldığında neler yapılacağını belirleyebiliriz.</p>
 <b>On Changed Event</b>	<p>On Hypothesis Changed Event (Hipotez Değişme Olayı) bloğu ile hipotez değiştiğinde neler yapılacağını belirleyebiliriz.</p>

## BÖLÜM 7 – HAZIR KÜTÜPHANE FONKSİYONLARI

### Math (Matematik)










Matematik kütüphanesi, matematiksel işlemlerde kullandığımız blokları barındırır. Aritmetik işlemlerin yanı sıra, karşılaştırma işlemleri de bu kütüphanede bulunur.

 <b>Value</b>	Value (Değer) bloğu, tam sayı değeri eklememizi sağlar.
 <b>Float Value</b>	Float Value (Ondalık Değer) bloğu, ondalıklı bir sayı değeri eklememizi sağlar.
 <b>Sum</b>	Sum (Topla) bloğu, iki veya daha fazla sayının toplamını döndürür.
 <b>Subtract</b>	Subtract (Çıkar) bloğu, iki sayının farkını döndürür.
 <b>Multiply</b>	Multiply (Çarp) bloğu, iki veya daha fazla sayının çarpımını döndürür.
 <b>Divide</b>	Divide (Bölme) bloğu, iki sayının bölümünü döndürür.
 <b>Power</b>	Power (Üssü) bloğu, ilk eklenen sayının ikinci eklenen sayı kadar üssünü döndürür.
 <b>Equality</b>	Equality (Eşitlik); iki sayı arasındaki eşit, eşit değil, büyük eşit, küçük eşit, büyüktür ve küçüktür ifadelerinin sonucunu döndürür.
 <b>Fixed Decimal Places</b>	Fixed Decimal Places (Sabit Ondalık Değer) bloğu, ondalık değere sahip bir sayının virgülden sonra kaç hane olduğunu döndürür.
 <b>Random</b>	Random (Rastgele) bloğu, eklenen iki değer arasından rastgele bir ondalık değer döndürür.

 <b>To Numeric</b>	To Numeric (Sayıya Çevir), eklenen bloğu sayıya çevirir.
 <b>Round</b>	Round (Yuvarla) bloğu, eklenen değeri matematiksel değerine göre yuvarlar.
 <b>Min Max</b>	Min Max (Minimum Maksimum) bloğu, eklenen iki değer arasındaki en büyük veya en küçük değeri döndürür.

### Math Advanced (İleri Matematik)

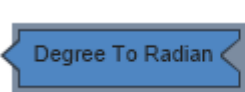
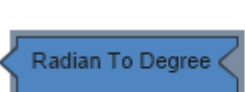
Daha ileri matematiksel işlemler için İleri Matematik blokları kullanılabilir. Örneğin; logaritma, mutlak değer, sinüs, kosinüs gibi bloklar burada yer alır.







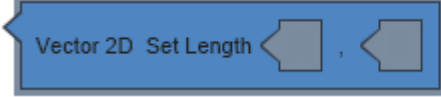
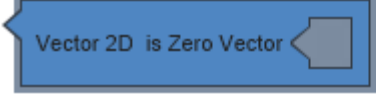
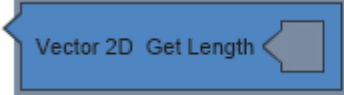
 <b>Absolute</b>	Absolute (Mutlak Değer) bloğu, eklenen sayının mutlak değerini döndürür.
 <b>Atan</b>	Atan bloğu, eklenen sayının arctanjantını derece cinsinden döndürür.
 <b>Atan2</b>	Atan2 bloğu, eklenen iki sayının arctanjantını döndürür.
 <b>Asin</b>	Asin bloğu eklenen sayının arcsinüsünü derece cinsinden döndürür.
 <b>Ceil</b>	Ceil (Tavan) bloğu, eklenen sayıdan büyük veya ona eşit olan en küçük tam sayıyı döndürür.
 <b>Cos</b>	Cos bloğu, eklenen sayının kosinüsünü derece cinsinden döndürür.
 <b>Exp</b>	Exp bloğu, eklenen sayıyı eksponansiyel sayısının (2,71828...) üssü olarak döndürür.
 <b>Floor</b>	Floor (Taban) bloğu, eklenen sayıdan küçük veya ona eşit olan en büyük tam sayıyı döndürür.
 <b>Log Name</b>	Log (Logaritma), eklenen sayının logaritmasını döndürür.

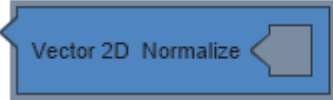
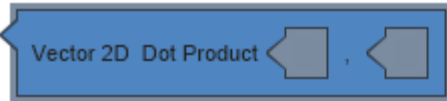

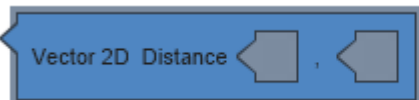



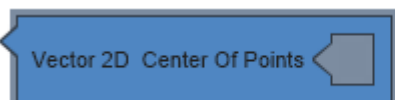
 Sin	Sin (Sinüs), eklenen sayının sinüs değerini derece cinsinden döndürür.
 Sqrt	Sqrt (Karekök), eklenen sayının karekök değerini döndürür.
 Tan	Tan bloğu, eklenen sayının tanjant değerini derece cinsinden döndürür.
 Constant	Constant (Sabit), bazı önemli sayıların (örn: e, pi) sabit değerlerini döndürür.
 Minimum In List	Minimum in List bloğu, listedeki en küçük değeri döndürür.
 Maximum In List	Maximum in List bloğu, listedeki en büyük değeri döndürür.
 Mod 10 of : Modulus	Modulus (Mod) bloğu, eklenen sayının modunu döndürür. Hangi sayıya göre modunun alınacağını da belirleyebiliriz.
 IsNaN	Is NaN (Sayı Değil mi) bloğu, eğer eklenen nesne bir sayı ise True (Doğru) değerini döndürür aksi hâlde (Yanlış) değerini döndürür.
 Plus One	Plus One (Bir Artır) bloğu, eklenen değeri bir artırır.
 Minus One	Minus One (Bir Azalt) bloğu, eklenen değeri bir azaltır.

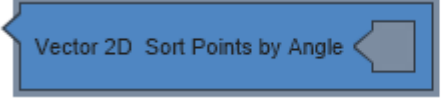
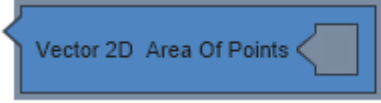
## Trigonometry (Trigonometri)

İleri Matematiğe ek olarak, VFabrika'da trigonometrik işlemleri yapmamızı sağlayan bloklar da yer alır.

 Degree To Radian	Degree To Radian (Dereceyi Radyana Çevir) bloğu, eklenen değeri radyana çevirir.
 Radian To Degree	Radian To Degree (Radyanı Dereceye Çevir) bloğu, eklenen değeri dereceye çevirir.

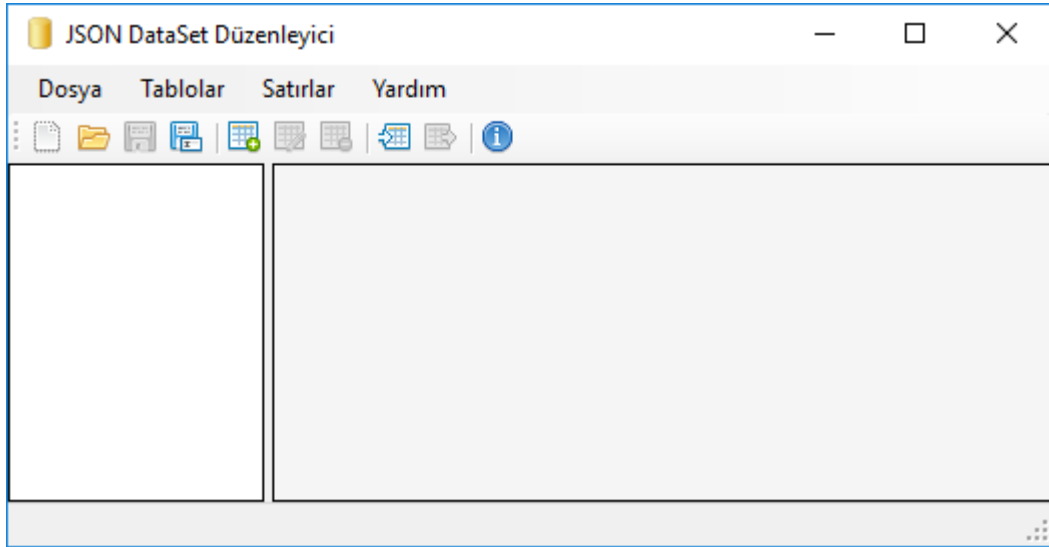
	<p>Vector 2D Create (2B Vektör Oluştur) bloğu, girilen X ve Y değerlerine göre bir vektör oluşturur.</p>
	<p>Vector 2D Add (2B Vektör Topla) bloğu, iki vektörün toplamını döndürür.</p>
	<p>Vector 2D Subtract (2B Vektör Çıkart) bloğu, iki vektörün farkını döndürür.</p>
	<p>Vector 2D Multiply (2B Vektör Çarp) bloğu, vektörün eklenen sayı ile olan çarpımını döndürür.</p>
	<p>Vector 2D Divide (2B Vektör Böl) bloğu, vektörün eklenen sayı ile bölümünü döndürür.</p>
	<p>Vector 2D Inverse (Tersine Çevir) bloğu, vektörü tam tersi yönüne çevirir. Örneğin, Vektör X: 10, Y: 20 ise tersine çevir bloğu ile X: -10, Y: -20 olur.</p>
	<p>Vector 2D Set Length (Uzunluğu Değiştir) bloğu, vektörün eklenen sayı kadar uzunluğunu değiştirir. Vektör aynı yönde fakat farklı büyüklükte olur.</p>
	<p>Vector 2D is Zero Vector (Boş 2B Vektör mü?) bloğu, eklenen vektörün sıfır olup olmadığını kontrol eder. Vektör eğer sıfır ise True (Doğru), değilse False (Yanlış) değerini döndürür.</p>
	<p>Vector 2D Get Length (Uzunluk Hesapla) bloğu, vektörün uzunluğunu döndürür.</p>

 <b>Vector 2D Normalize</b>	<p>Vector 2D Normalize (Birim Vektöre Çevir) bloğu, vektörü birim vektörüne dönüştürür. Vektörün uzunluğu 1 olur ve yönü değişmez.</p>
 <b>Vector 2D Dot Product</b>	<p>Vector 2D Dot Product (Nokta Çarpımı) bloğu, eklenen iki vektörün nokta çarpımını döndürür. Nokta çarpımı, karşılıklı bileşenlerin çarpımları toplamına eşittir.</p>
 <b>Vector 2D Cross Product</b>	<p>Vector 2D Cross Product (Çapraz Çarpım) bloğu, eklenen iki vektörün çapraz çarpımını döndürür.</p>
 <b>Vector 2D Distance</b>	<p>Vector 2D Distance (Uzaklık) bloğu, eklenen iki vektör arasındaki uzaklığı döndürür.</p>
 <b>Vector 2D Rotate Degree</b>	<p>Vector 2D Rotate Degree (Açıyla Döndür) bloğu, eklenen vektörü belirtilen açı doğrultusunda döndürür.</p>
 <b>Vector 2D Angle</b>	<p>Vector 2D Angle (2B Vektör) bloğu, 180 derece işaretli veya işaretsiz iki vektör arasındaki açıyı döndürür.</p>
 <b>Vector 2D Equality</b>	<p>Vector 2D Equality (2B Vektörel Eşitlik) bloğu, eklenen iki vektörün eşit olup olmadığını kontrol eder. Vektörler birbirine eşitse True (Doğru), değilse False (Yanlış) değerini döndürür.</p>
 <b>Vector 2D Center Of Points</b>	<p>Vector 2D Center of Points (Noktanın Ağırlık Merkezini Hesapla) bloğu, vektörlerin ağırlık merkezini döndürür. Örneğin, bir listede 4 tane</p>

	vektör varsa bu vektörlerin ağırlık merkezini verir.
	Vector 2D Sort Points by Angle (Noktaları Açıyla Göre Sırala) bloğu, eklenen listede yer alan vektörleri X ekseninden başlayarak sıralar.
	Vector 2D Area of Points (Noktaların Oluşturduğu Alanı Hesapla) bloğu, eklenen listede yer alan vektörlerin toplam alanını döndürür.

## Json Data Set

Json bir veri değişim formatıdır. Daha küçük boyutlarda veri saklamak ve alıp göndermek için kullanılır. VFabrika’da bir JSON DataSet Düzenleyici bulunmaktadır. Araçlar > Json veri seti editörü yolu ile ulaşılabilir.



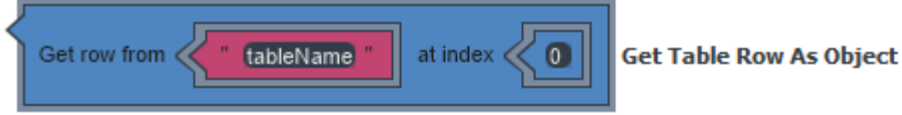
Şekil 128

Buradan bir tablo oluşturabilir ve onu da bir veri tabanı gibi kullanabiliriz. Bloklar yardımıyla oluşturduğumuz tablodan veri çekebiliriz. Ancak, veri ekleme veya değiştirme işlemlerini yapamayız.





Get Value (Değer Oku) bloğu, oluşturduğumuz tablonun adını, satırını, alan adını girerek istediğimiz veriye ulaşmamızı sağlar.








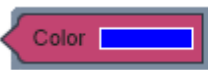

Get Table Row As Object (Tablo Satırını Nesne Olarak Al) bloğu, tablo adını ve satırını girdiğimiz verileri bir nesne olarak bize getirir.





Get Table Row Count (Tablo Satır Sayısı) bloğu, oluşturduğumuz tablodaki satır sayısına ulaşmamızı sağlar.

## BÖLÜM 8 – DİĞER ARAÇLAR

### General

 <b>Create Object</b>	<p>Create Object (Nesne Oluştur) bloğu VFabrika'da nesne oluşturmamızı sağlar. Nesneyi oluşturmak için değişken tanımladığımızda değer kısmına Create Object bloğunu ekleriz.</p>
 <b>Get Object Property Value</b>	<p>Get Object Property Value (Nesne Özellik Değeri Oku) bloğu, oluşturduğumuz nesne ve özelliğini okumamızı sağlar.</p>
 <b>Set Object Property Value</b>	<p>Set Object Property Value (Nesne Özellik Değeri Ata) bloğu, oluşturulan nesneye bir özellik oluşturabilir ve ona da bir değer atamamızı sağlar.</p>
 <b>Set Depth</b>	<p>Set Depth (Derinliği Ayarla) bloğu, nesnenin derinliğini ayarlamamızı sağlar.</p>
 <b>Get Depth</b>	<p>Get Depth (Derinliği Oku) bloğu, nesne derinliğine ulaşmamızı sağlar.</p>
 <b>Color Value</b>	<p>Color Value (Renk Değeri) bloğu ile VFabrika'da kullandığımız nesnelere renk verebilir ve renklerin hexadecimal (onaltılı) kodlarına ulaşabiliriz.</p>
 <b>Set Property</b>	<p>Set Property (Özellik Yaz) bloğunda, ilk açılır pencerede sahnedeki tüm nesneleri görebilir ve ikinci açılır pencerede de seçtiğimiz nesnenin tüm özellikleri listelenir. Burada istediğimiz nesnenin özelliğini ayarlayabiliriz.</p>

	<p>Get Property (Özellik Oku) bloğu ile tüm eklediğimiz nesnelerin özelliklerine ulaşabiliriz.</p>
	<p>Get Reference (Referans Oku) bloğu, sahnedeki nesnelere ulaşmamızı sağlar. Get Property (Özellik Oku) bloğunda da Get Reference bloğu ile nesneye ulaşabiliyoruz.</p>
	<p>Get Reference Property (Referans Özelliği Oku) bloğu, nesnelerin özelliklerine ulaşmamızı sağlar. Get Property (Özellik Oku) bloğunda Get Reference Property bloğu ile nesnenin özelliğine ulaşabiliriz.</p>
	<p>Get Attribute (Nitelik Oku) bloğu ile niteliğini okumak istediğimiz nesneyi seçtikten sonra yazdığımız niteliğe ulaşabiliriz.</p>
	<p>Set Attribute (Nitelik Yaz) bloğu ile niteliğini ayarlayacağımız nesneyi seçtikten sonra nesnenin eski ve yeni niteliğini yazarak nesnenin niteliğini değiştirebiliriz.</p>