
GÖRSEL PROGRAMLAMA
Olay tabanlı programlama ve basit bileşenler

Olay Tabanlı Programlama

- Event driven programming
- Ana döngü
- Kullanıcıdan alınan giriş
- Callback

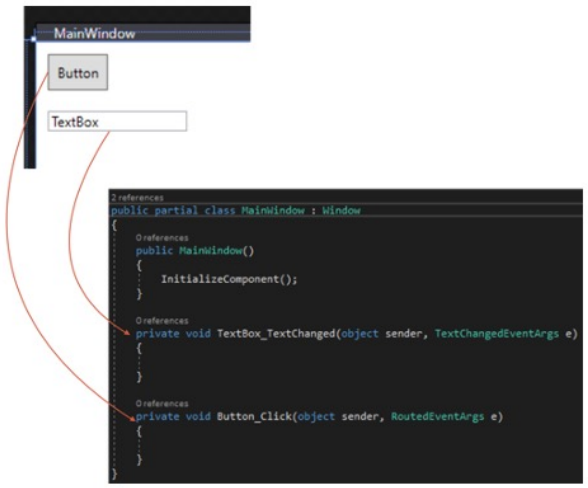
Olay tabanlı programlama aşağıda verilen bileşenlerden oluşur.
Olayları dinleyen bir ana döngü (main loop - main thread): Kullanıcı uygulamayı çalıştırdığında çalışmaya başlar. Kullanıcının etkileşim takibi ve arayüzün güncellenmesi işlevini üstlenir.
Kullanıcıdan tetiklenen bir olaya göre metotların çalışması: Kullanıcının gerçekleştirdiği herhangi bir girdide veya arayüz ile olan etkileşiminde belirli metotlar çağırılır.
Çağırımların kontrolü (callback): Çağırılan metotların işlemleri gerçekleştirilir ve sonucunda varsa ui değişiklikleri main thread üzerinde gerçekleştirilir.
UI üzerindeki bileşenlerin kontrolü main thread üzerinde olduğu için sonraki haftalarda bahsedilecek thread uygulamalarında ana thread üzerindeki UI bileşenlerine erişimde bazı kısıtlar bulunmaktadır.

Olay Tabanlı Programlama

```
public static string Decode(MWIStream Stream)
{
    string[] D1 = (" " + Stream.D1).Split(" ");
    string[] D2 = Stream.D2.Split(" ");
    string[] S3 = Stream.S3.Split(" ");
    int S1Counter, S2Counter, S3Counter, CodeWord;
    StringBuilder Output = new StringBuilder();

    S1Counter = 0;
    S2Counter = 0;
    S3Counter = 0;
    bool[] BV = Stream.BV.SelectMany(GetBits).ToArray();

    for (int i = 0; i < BV.Length - Stream.RedundantBitLength; i++)
    {
        if (!BV[i])
        {
            CodeWord = Stream.S1[S1Counter++];
            if (CodeWord == 0)
                Output.Append(S3[S3Counter++]+ " ");
            else
                Output.Append(D1[CodeWord]);
        }
        else if (BV[i])
        {
            byte[] parca = new byte[2];
            parca[0] = Stream.S2[S2Counter++];
            parca[1] = Stream.S2[S2Counter++];
            CodeWord = BitConverter.ToInt16(parca, 0);
            Output.Append(D2[CodeWord]);
        }
```



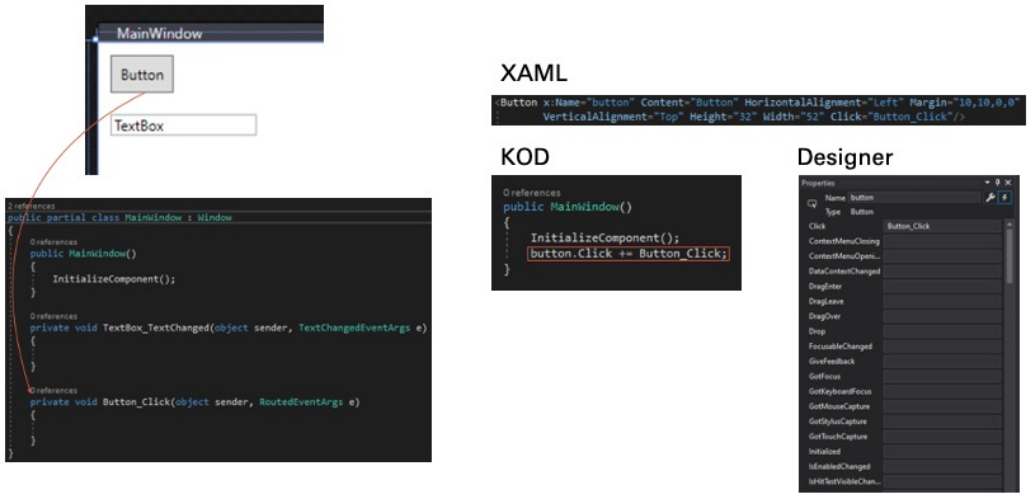
Prosedürel programlamada uygulama main metodundan başlayıp satır sırasıyla kodları çalıştırmaya devam eder. Fonksiyon çağırımında da bu fonksiyona yönlenir ve dönüşünde kalınan satırdan devam edilir. Olay tabanlı programlamada ise ana döngünün çalışmaya başlamasından itibaren kullanıcıdan bir girdi beklenir. Bu girdinin türüne göre gereken metot çağırılır. Örneğin bir tuşa basıldığında yalnızca Button_Click olayı çalışır ve TextBox'ın içine bir veri girildiğinde TextBox_TextChanged çalışır. Event'ların altında çalışan kod yine sıralı bir şekilde icra edilir.

Olay Tabanlı Programlama

- Event handler
- Arayüze bağlama
- Event args

Event handler herhangi bir bileşenin tetiklenen olayını bağlamak için kullanılır.
Olaylar gerçekleştirilirken bu olayların detayları (tuş basma olayında hangi tuşa basıldığı, fare tıklama olayında hangi tuşa tıklandığı gibi) event args sayesinde elde edilir.
Tasarım ekranına eklenen bileşenlerin kodda olaylara bağlanması için birden fazla yol bulunur

Olay Tabanlı Programlama



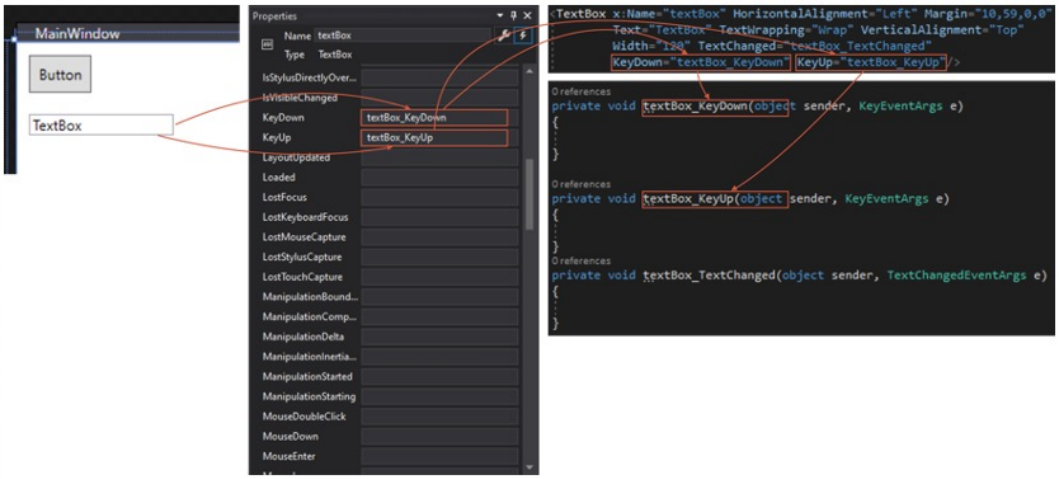
Elle event eklenebildiği gibi çift tıklama ile de eklenebilir. Çift tıklama ile olay XAML koduna da eklenir. Çift tıklama ile bileşenin varsayılan olayı eklenmektedir. Diğer olaylar için yine xaml, properties penceresi ya da kod ile ekleme kullanılmalıdır. Kod üzerinden eklenen event dinamik bileşen eklemede de kullanılmaktadır.

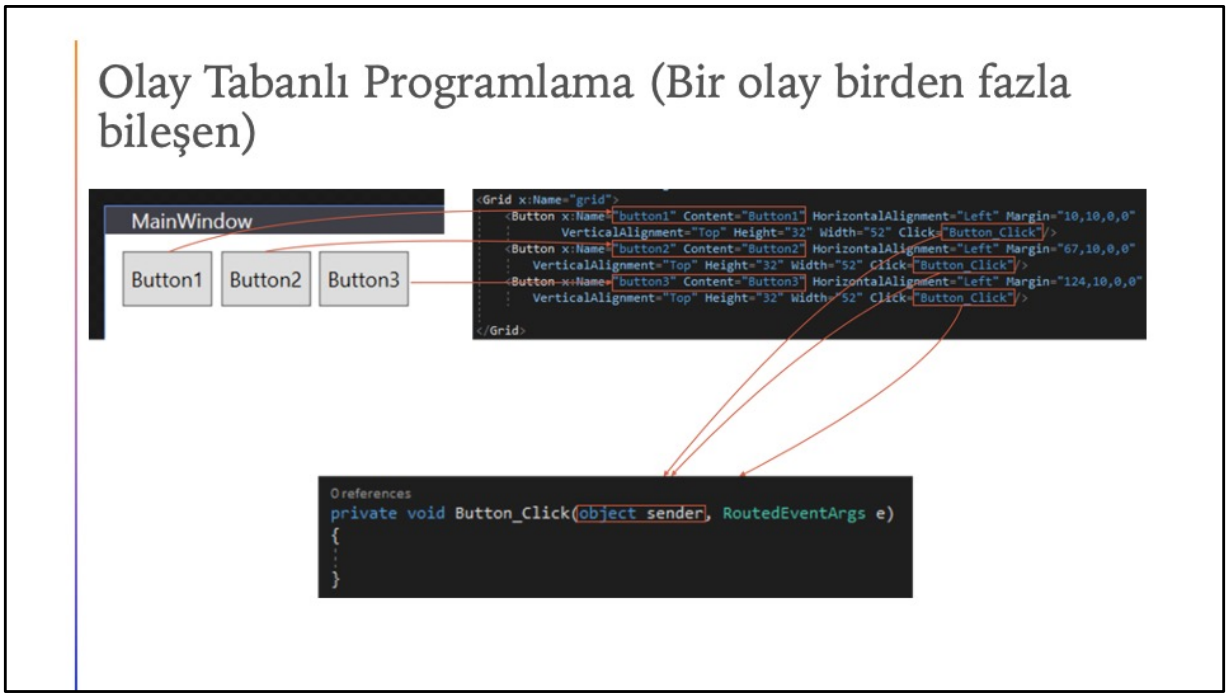
Olay Tabanlı Programlama

- Bir bileşen - Birden fazla event
- Bir event - Birden fazla bileşen

Bir bileşene birden fazla event bağlanabilir. Bu durumda bir bileşenin üzerinde gerçekleştirilen birden fazla olay için farklı senaryolar üretilebilir. Örneğin bir tuşun üzerine geldiğinde rengi değişsin ve tıklandığında ise hesaplama işlemi gerçekleştirmek istiyorsanız bu durumda tuşun üzerine gelindiğinde (mouseenter) ve tıklandığında (click) çalışacak iki farklı olay oluşturmanız gerekmektedir.
Bir event'ı birden fazla bileşen kullanabilir. Eğer birden fazla ui elemanınız aynı işlemi yapıyorsa (örneğin bir hesap makinesindeki tüm sayı tuşları yalnızca sayı girdisi yapmaktadır) bu durumda hepsinin bir event'ı çağırması sağlanabilir. Böylece kod tekrarından da kurtulmak amaçlanmış olur.

Olay Tabanlı Programlama (Bir bileşen birden fazla olay)



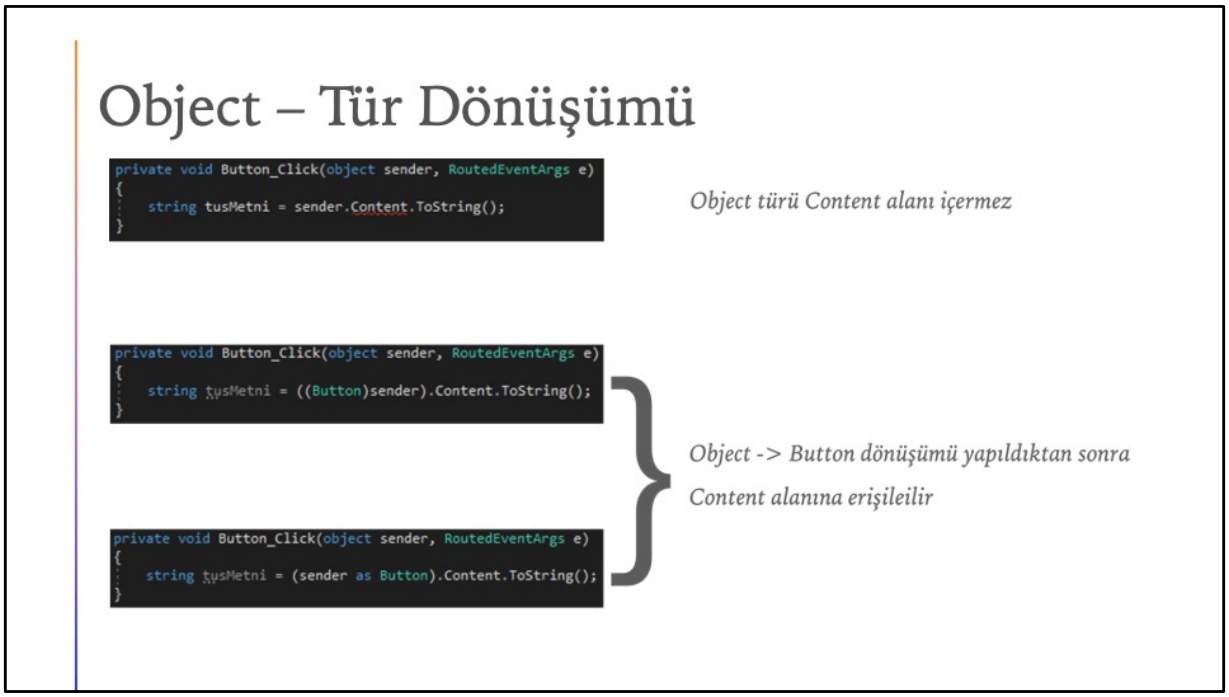


Birden fazla component bir event kullanıyorsa, çağıran component'i tespit etmek amacıyla sender kullanılır. Sender her türden arayüz elemanı olabileceği için object olarak tanımlanmıştır. Kod içerisinde daha sonra bu elemanın özelliklerine erişebilmek için object türünden dönüşüm gerçekleştirilmesi gerekmektedir.

Object – Tüm sınıfların temel sınıfı

- Sender -> Object
- Object
  - Equals()
  - Finalize()
  - GetHashCode()
  - GetType()
  - Clone()
  - ReferenceEquals()
  - ToString()

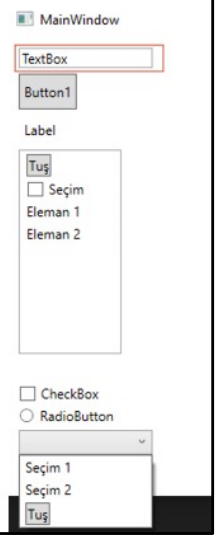
sender parametresinin object türünden olmasının sebebi C#'ta tüm sınıfarın object'ten türetilmiş olmasıdır. Bu sebeple her sınıf mutlaka object'in içerdiği listedeki metotları içermektedir.



object türünde bir nesneyi türünü bildiğimiz bir nesneye dönüştürmek için parantezler arasında dönüşüm yapabileceğimiz gibi as anahtar kelimesini de kullanabiliriz.

Temel Bileşenler (Textbox)

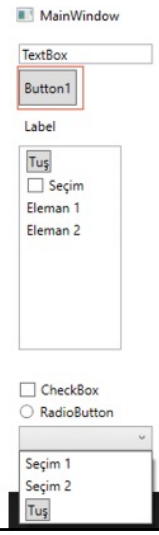
- Özellik
  - Text
- Olay
  - KeyDown
  - KeyPress
  - TextChanged



Textbox ekrandan girdi almak amacıyla kullanılan en temel bileşenlerden biridir. İçerisindeki metne erişmek için .text özelliği kullanılır. En çok kullanılan olayları ise TextChanged -> textbox içerisinde bir değişiklik olduğunda tetiklenir.
KeyPress -> textbox üzerine odaklanılmışken (focus) bir tuşa basılması ile tetiklenir.
KeyDown -> textbox üzerinde bir tuşa basıldığında, bu tuş bırakılmadan tetiklenir. Hangi tuşa basıldığının eventarg'ı kullanılır.

Temel Bileşenler (Button)

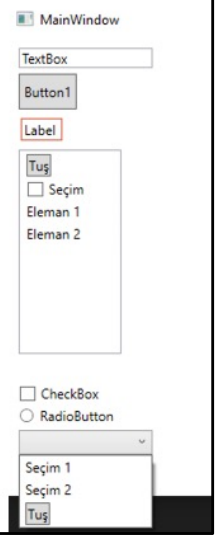
- Özellik
  - Content
- Olay
  - Click
  - MouseDown
  - MouseUp



Herhangi bir işlemi gerçekleştirmek için tıklamak adına button kullanılır.
İçeriğine erişmek adına content kullanılır. Text kullanılmamasının sebebi, button içerisinde istenilen bir başka component'in koyulabilmesidir.
Click -> tuşa tıklanıldığında tetiklenir
MouseDown -> tuşa fare ile tıklandığında bırakılmadan tetiklenir.
MouseUp -> tuşa fare ile tıklanıp bırakıldığında tetiklenir.

Temel Bileşenler (Label)

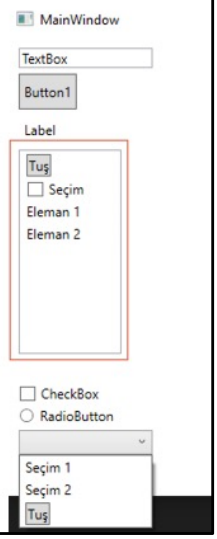
- Özellik
  - Content
- Olay
  - -



Ekranda genellikle sabit olan metinlerin gösterilmesi için kullanılır ve content ile içeriği değiştirilebilir. Temel tüm event'lara sahip olsa da genellikle etkileşim için kullanılmazlar.

Temel Bileşenler (Listbox)

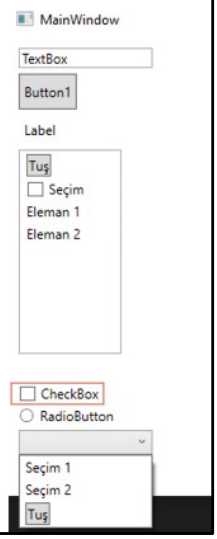
- Özellik
  - Items (ItemSource)
  - SelectedIndex
  - SelectedItem
- Olay
  - SelectionChanged



Liste halindeki verilerin gösteriminde kullanılır. Items özelliği ile ekleme, silme, değiştirme gibi işlemler yapılabildiği gibi, direkt olarak bir dizi, liste gibi sınıfların içeriği itemSource ile listelenebilir.
SelectedIndex özelliği ile seçilen index değerine ulaşılabilir ve SelectedItem özelliği ile seçilen nesneye erişilebilir.
Varsayılan olayı SelectionChanged listede yapılan seçim değişikliğinde tetiklenir.

Temel Bileşenler (Checkbox)

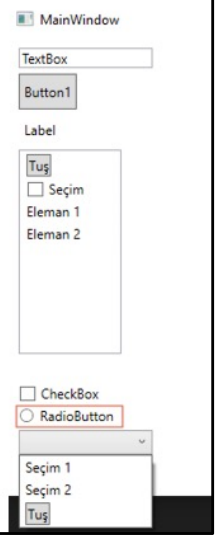
- Özellik
  - Content
  - IsChecked
- Olay
  - Checked



Çoklu seçimler için kullanılan bir bileşendir. Birden fazla seçim yapılacakken kullanılır. Content ile içeriği ya da metni değiştirilebilirken ischecked özelliği ile seçili olup olmadığı belirlenebilir.
Seçildiğinde tetiklenen checked olayı bulunmaktadır.

Temel Bileşenler (Radiobutton)

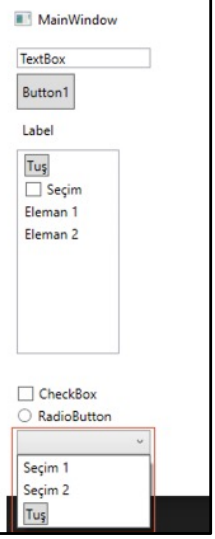
- Özellik
  - Content
  - IsChecked
- Olay
  - Checked



Checkboxtan farklı olarak aynı container içerisinde yalnızca bir radiobutton seçili olabilmektedir.

Temel Bileşenler (Combobox)

- Özellik
  - Items (ItemSource)
  - SelectedItem
  - SelectedIndex
- Olay
  - SelectionChanged



Listbox'a benzer bir yapıdadır ve seçildiğinde listenin açıldığı bir bileşendir. Listboxta olduğu gibi eleman olarak istenen bileşenler eklenebilmektedir.