C# DA DELEGATLAR VA HODISALARNING ROLI: VOQEALARGA ASOSLANGAN DASTURLASHGA CHUQUR KIRISH

Farmonov Sherzodbek Raxmonjonovich
Farg’ona davlat unversiteti amaliy matematika va informatika kafedrasi katta o‘qituvchisi
farmonovsh@gmail.com
Hakimov Muhammadqodir Tolibjon o’g’li
Farg‘ona davlat universiteti 2-kurs talabasi
muhammadqodirxakimov775@gmail.com

ARTICLE INFO
Qabul qilindi: 10- December 2023 yil
Ma’qullandi: 15- December 2023 yil
Nashr qilindi: 21-December 2023 yil
KEY WORDS
C#, Delegatlar, Hodisalar, Voqea-hodisalarga asoslangan dasturlash, Dasturiy ta'minot, Paradigma, Dinamik o'zaro ta'sir, Publish-subscribe, Tushunchalar, Sezgir ilovalar, O'zgaruvchilar, Event , EventHandler, Main, CLR, .NET Framework, Dokumentatsiya

ABSTRACT
Taqdim etilgan maqolada C# delegatlar va hodisalarni dasturiy ta'minot ilovalarida dinamik o'zaro ta'sir va javob berish uchun qanday ishlatishini ko'rsatadi. Izoh delegatlar kontseptsiyasini o'rganadi va ularning C# da voqealarga asoslangan dasturlash uchun asosiy konstruktsiyalar sifatidagi rolini ta'kidlaydi. Delegatlar dinamik ravishda usullarni o'tkazish va chaqirishni osonlashtiradigan, turdagi xavfsiz, havola mexanizmlari sifatida tavsiflanadi.

C# dasturlash tilidagi delegatlar va hodisalar kontseptsiyasini ta'riflanadi. Maqola voqealarga asoslangan dasturlashning asosiy paradigmasi va uning C# dasturlashda ahamiyati haqida gaplashish bilan boshlanadi. Delegatlar va hodisalar, dinamik o'zaro ta'sir va javob berish uchun foydalaniladigan asosiy konstruktsiyalar sifatida ta'kidlangan. Delegatlar kontseptsiyasini o'rganish va ularning C# dasturlashdagi ro'li haqida tushunchalar beriladi. Delegatlar dinamik ravishda usullarni o'tkazish va chaqirishni osonlashtirish, xavfsizlikni ta'minlashdir Maqola o'z ichiga C# dasturlashda voqealarga asoslangan modelda nashriyotlar va obunachilar o'rtasida aloqa o'rnatishda delegatlar va hodisalarining muhimligini ajratib o'tilgan. Ular C# dagi voqealarni o'rganish va ularni xatti-harakatlardagi o'zgarishlarni qamrab oluvchi yuqori darajadagi abstraktsiyalar sifatida ta'riflangan.

Maqola C# da delegat va hodisalardan foydalanishni tushunishni mustahkamlovchi kod misollari orqali ko'rsatadi. Ular bilan yaratilgan kodlar delegat yaratish, voqealarni e'lon qilish, voqealarni boshlash va voqealarga obuna bo'lishni o'rgatadi.

Voqealarga asoslangan dasturlash dasturiy ta'minotni ishlab chiqishda asosiy paradigma bo'lib, ilovalarga foydalanuvchi o'zaro ta'siri va tizim hodisalariga dinamik javob berishga imkon beradi. C# dasturlash sohasida delegatlar va voqealar rolini tushunish sezgir va interaktiv ilovalarni yaratish uchun juda muhimdir. Ushbu maqola C# da voqea-hodisalarga asoslangan dasturlashga chuqur kirishni taqdim etadi , bu kontekstdagi delegatlar va voqealarning ahamiyatini ta'kidlaydi.

C# da delegatlar:

Delegatlar C# da voqealarga asoslangan dasturlash uchun asos bo'lib xizmat qiladi. Delegat usullarga havolalarni ifodalovchi tur bo'lib, bu usullarni parametr sifatida o'tkazish va dinamik ravishda chaqirish imkonini beradi. Delegatlar voqealarga asoslangan modelda noshirlar va obunachilar o'rtasida aloqa o'rnatishda hal qiluvchi rol o'ynaydi.

C# da delegatlar “delegate” kalit so'zi yordamida e'lon qilinadi . Ular qaytish turi va parametrlarini belgilovchi usul imzosini qamrab oladi. Delegatlar bilvosita usullarni chaqirishni osonlashtirib, usul ko'rsatkichlarini yaratishga imkon beradi.

C# da voqealar:

Voqealar delegatlar ustidan yuqori darajadagi mavhumlikni ta'minlaydi va nashriyot va obunachi munosabatlarini qamrab oladi. Hodisa - ob'ektlar o'rtasidagi aloqa mexanizmi, muayyan harakat yoki holat o'zgarishining paydo bo'lishi haqida signal beradi. Tadbirlar delegatlarga asoslangan bo'lib , noshirning ichki amalga oshirishning yaxlitligini himoya qiluvchi bilvosita qatlam sifatida ishlaydi.

C# da hodisani e'lon qilish uchun `event` kalit so'zi delegat turi bilan birgalikda ishlatiladi. Voqealar bir nechta ishlov beruvchilar (obunachilar) tomonidan obuna bo'lishi mumkin, bu ajratilgan va kengaytiriladigan arxitekturaga imkon beradi.

Voqealarga asoslangan dasturlash ish jarayoni:

1. Delegatning tuzilishi :

```
public delegate void MyEventHandler(object sender, EventArgs e);
```

2. Hodisa deklaratsiyasi :

```
public event MyEventHandler MyEvent;
```

3. Noshir tomonidan amalga oshirilishi :

```
public class HodisaNashriyoti
{
  public event MyEventHandler                                    MyEvent;

  protected virtual void OnMyEvent()
  {
    MyEvent?.Invoke(this, EventArgs.Empty);
  }

  public void PerformAction()
  {

    OnMyEvent();
  }
}
```

4. Abonentni amalga oshirish :

```
public class EventSubscriber
{
  public void Subscribe(HodisaNashriyoti                          Nashriyotchi)
  {
    publisher.MyEvent +=                                          HandleEvent;
```

```
  }

  private void HandleEvent(object sender, EventArgs e)
  {
    // Handle the event
  }
}
```

C# da delegatlar va hodisalar publish -subscribe naqshini amalga oshirish uchun ishlatiladi , bunda ob'ekt (nashriyotchi) boshqa ob'ektlarni (obunachilarni) muayyan hodisalar yoki harakatlar haqida xabardor qilishi mumkin. Vakillar ham, tadbirlar ham voqealarni boshqarish bilan bog'liq bo'lsa-da, ular bir oz boshqacha maqsadlarga xizmat qiladi.

Delegate C# da ma'lum bir imzoga ega usullarga havolalarni ifodalovchi tur. U usullarni birinchi darajali ob'ektlar sifatida inkapsulyatsiya qilish va o'tkazish usulini taqdim etadi. Delegatlar C va C++ dagi funksiya ko'rsatkichlariga o'xshaydi. Delegatlar delegate kalit so'zi yordamida e'lon qilinishi va mos imzoga ega bo'lgan usul bilan yaratilishi mumkin . Instantsiya qilingandan so'ng, delegat oddiy usul kabi chaqirilishi mumkin. Delegatlar, birinchi navbatda, qayta qo'ng'iroq qilish mexanizmlarini yaratish uchun ishlatiladi , bu usulga ma'lum bir harakat sodir bo'lganda chaqiriladigan boshqa usulni belgilashga imkon beradi. Mana delegat deklaratsiyasi va foydalanishga misol:

```
using System;
delegate void MyDelegate(string message);
class TestClasss
{
  public void MyMethod(string message)
  {
    Console.WriteLine(message);
  }
}
class Program
{
  static void Main()
  {
    TestClasss obj = new TestClasss();
    MyDelegate del = new MyDelegate(obj.MyMethod);
    del("Hello, world!");
  }
}
```

Shamollatish teshiklari delegatlar ustiga qurilgan va inkapsulyatsiya va himoya qatlamini ta'minlaydi. Voqea - ob'ektda ma'lum bir harakat yoki holat o'zgarishi sodir bo'lganda obunachilarni (voqea ishlov beruvchilarini) xabardor qilish mexanizmi. Voqealar faqat sinf yoki struct ichida e'lon qilinishi mumkin va ular tashqi obunachilar uchun hodisa ishlov beruvchilarini biriktirish va ajratish uchun ochiq bo'lishi mumkin. Voqealar odatda delegat turi bilan birga voqea kalit so'zi yordamida e'lon qilinadi . Hodisa deklaratsiyasi ko'p tarmoqli delegat yaratadi, bu hodisaga bir nechta voqea ishlov beruvchilarini biriktirish imkonini

beradi.

Mana voqea deklaratsiyasi va foydalanishga misol:

```
using System;
class MyEventClass
{
  public event EventHandler MyEvent;
  public void TriggerEvent()
  {
    OnMyEvent();
  }
  protected virtual void OnMyEvent()
  {
    MyEvent?.Invoke(this, EventArgs.Empty);
  }
}
class Program
{

  static void Main()
  {
    MyEventClass obj = new MyEventClass();
    obj.MyEvent += HandleEvent;
    obj.TriggerEvent();
  }
  static void HandleEvent(object sender, EventArgs e)
  {
    Console.WriteLine("Event handled Call!");
  }
}
```

Ushbu misolda **MyEventClass MyEvent** deb nomlangan hodisani e'lon qiladi **EventHandler** turi . TriggerEvent usuli **OnMyEventni chaqirish** orqali voqeani chaqiradi usul, bu o'z navbatida hodisaga biriktirilgan voqea ishlov beruvchilarini chaqiradi. Main usuli **HandleEvent** ni biriktirish orqali hodisaga obuna bo'ladi usuli voqea ishlovchisi sifatida, keyin voqea ishga tushirilganda chaqiriladi .

Event mavhumlik va inkapsulyatsiya darajasini ta'minlaydi, bu voqeani e'lon qiluvchi sinfga obunachilar hodisa ishlov beruvchilarini qanday biriktirish yoki ajratish mumkinligini nazorat qilish imkonini beradi. Bu tashvishlarni ajratishni saqlashga yordam beradi va komponentlar o'rtasida bo'sh ulanishni ta'minlaydi.

Xulosa:*Xulosa qilib aytganda, delegatlar va hodisalarning rolini tushunish C# da voqealarga asoslangan dasturlashni o'zlashtirish uchun juda muhimdir. Delegatlar usulga havolalar mexanizmini taqdim etadilar, voqealar esa noshir va obunachi munosabatlarini qamrab oladi, ajratilgan va kengaytiriladigan kodni rivojlantiradi. Ushbu kontseptsiyalardan foydalangan holda, ishlab chiquvchilar foydalanuvchilarning o'zaro ta'siri va tizim hodisalariga samarali*

javob beradigan sezgir va interaktiv ilovalarni yaratishi mumkin.

Foydalanilgan Adabiyotlar:

1. Albahari, J., Albahari, B., & Rice, O. (2017). C# 7.0 in a Nutshell: The Definitive Reference. O'Reilly Media.
2. Liberty, J., & MacDonald, B. (2009). Learning C# 3.0: Master the fundamentals of C# 3.0. O'Reilly Media.
3. Troelsen, A. (2012). Pro C# 5.0 and the .NET 4.5 Framework. Apress.
4. MSDN Documentation on Events in C#: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/event
5. MSDN Documentation on Delegates in C#: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/delegates/
6. Richter, J. (2012). CLR via C# (4th Edition). Microsoft Press.
7. Skeet, J. (2019). C# in Depth. Manning Publications.