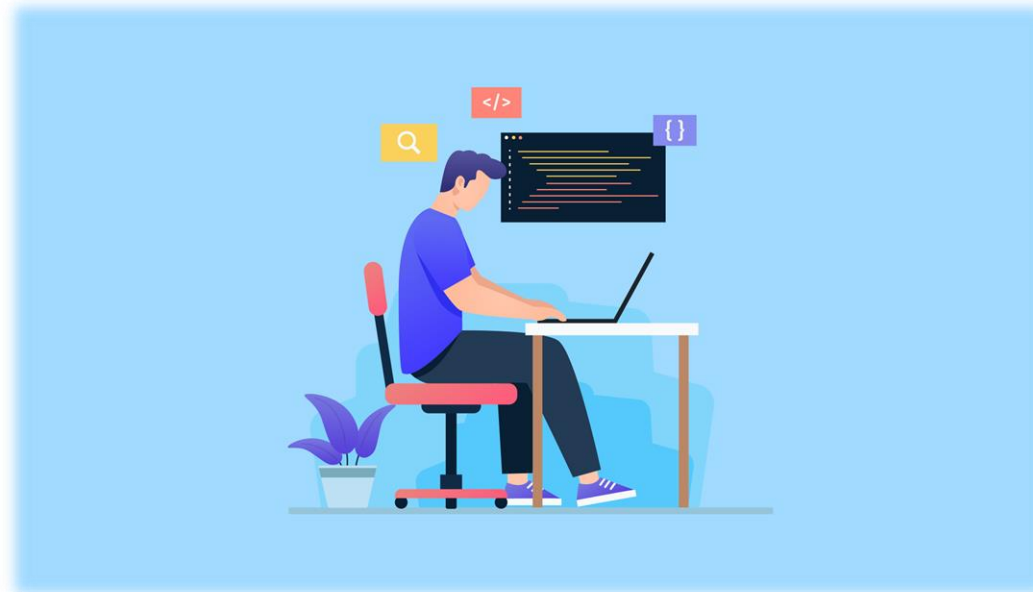


Programuje z użyciem wyjątków



Co to jest wyjątek w kodzie?

Idea obsługi wyjątków polega na tym, że **funkcja, która napotkała problem, z którym nie potrafi sobie poradzić zgłasza wyjątek**. Wyjątek jest przesyłany do miejsca wywołania funkcji. Tam może być wyłapany i obsłużony lub może być przesłany dalej (wyżej).



Blok Try.....Catch

Try

{

//kod który może wywołać błąd podczas działania programu

}

catch

{

//strefa przechwycenia wyjątku z bloku try

}

Przykład

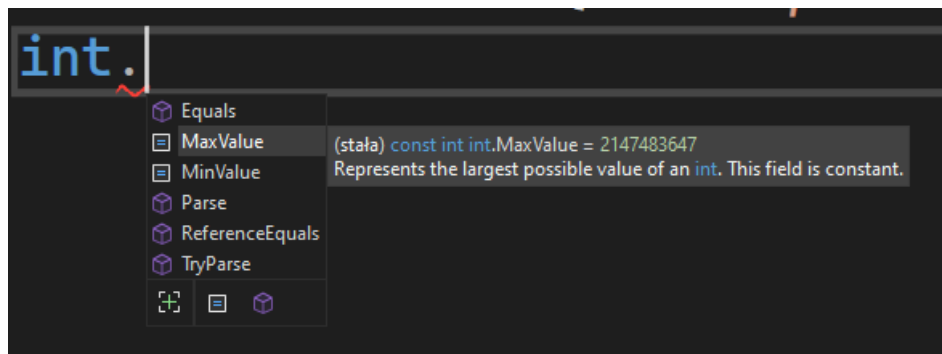
```
Console.WriteLine("Podaj dwie liczby");
try
{
    string x = Console.ReadLine();
    int xx = int.Parse(x);
    string y = Console.ReadLine();
    int yy = int.Parse(y);
    Console.WriteLine(xx/yy);

}
catch (FormatException fEx)
{
    Console.WriteLine(fEx.Message);
}
catch (OverflowException OverEx)
{
    Console.WriteLine(OverEx.Message);
}
catch (ArithmeticException ArgEx)
{
    Console.WriteLine(ArgEx.Message);
}
catch (Exception Ex)
{
    Console.WriteLine(„Źle!");
}
```

Użycie bloku „Finally”

```
int aa = 15;
try
{
    string a = Console.ReadLine();
    aa = int.Parse(a);
}
catch
{
    Console.WriteLine("Nie zmieniłeś zmiennej- Błąd");
}
finally
{
    Console.WriteLine(aa);
}
```

Słowo „unchecked”



```
int max = int.MaxValue;  
unchecked  
{  
    Console.WriteLine(max);  
    Console.WriteLine(++max);  
    Console.WriteLine(max + 21400000000);  
}
```

```
2147483647  
-2147483648  
-7483648
```

Słowo „checked”


```
int min = int.MinValue;  
int sprawdz = checked(min - 1);
```

Nieobsługiwany wyjątek

System.OverflowException: „Arithmetic operation resulted in an overflow.”

[Pokaż stos wywołań](#) | [Wyświetl szczegóły](#) | [Kopiuj szczegóły](#) | [Rozpocznij sesję rozszerzenia Live Share](#)

▸ [Ustawienia wyjątków](#)

 **Sprawdzaj przepełnienie arytmetyczne** ?

☐ Zgłaszaj wyjątki, gdy arytmetyczna liczba całkowita generuje wartości z poza zakresu.

Switch...case...throw

```
public static string DzieńTygodnia(int liczba)
{
    switch (liczba)
    {
        case 1:
            return "poniedziałek";
            break;
        case 2:
            return "wtorek";
            break;
        case 3:
            return "środa";
            break;
        case 4:
            return "czwartek";
            break;
        case 5:
            return "piątek";
            break;
        case 6:
            throw new System.Exception(„6 dzień tygodnia ja rozpalony
                                     jak pochodnia");

            break;
        case 7:
            throw new EndOfStreamException
            („Boży dzień, do kościoła czas");
            break;
        default:
            throw new ArgumentOutOfRangeException
            ("Liczba musi być z przedziału od 1 do 7");
    }
}
```

```
try
{
    DzieńTygodnia(int.Parse(Console.ReadLine()));
}
catch (Exception ex)
{
    Console.WriteLine(ex.Message);
};
```


Dziękuję za uwagę

