LEKCJA 9 – Konwencje pisania

(Dobre praktyki programowania)

Konwencje programowania, są to ogólnie przyjęte zasady pisania kodu. Nie mają one wpływu na działanie programu. Stosowanie ich zwiększa jednak czytelność i ułatwia programistą czytanie cudzego kodu. Konwencji programowania jest bardzo dużo. W tej lekcji poznamy tylko kilka z nich. Warto od razu wyrobić sobie nawyk ich stosowania, aby ułatwić sobie pracę w przyszłości.

Konwencje nazywania elementów programu

PascalCase

Wygląd

Wszystkie wyrazy nazwy piszemy razem, bez żadnych przerw. Pierwsza litera każdego wyrazu jest wielka a kolejne małe.

Np.: CaloriesCalculator, HelloWorld, addYears.

Zastosowanie

Nazywanie klas, plików, przestrzeni nazw (*namespace*), metod.

camelCase

Wygląd

Wszystkie wyrazy nazwy piszemy razem, bez żadnych przerw. Pierwsza litera prawie każdego wyrazu jest wielka a kolejne małe. Wyjątek stanowi pierwszy wyraz, który w całości jest pisany małymi literami.

Np.: addedYears, displayScreen, numberOfCircles.

Zastosowanie

Nazywanie lokalnych zmiennych i prywatnych elementów programu.

UPPER_CASE

Wygląd

Nazwę piszemy w całości wielkimi literami (wszystkie litery). Wyrazy oddzielamy od siebie przy pomocy podkreślnika („_”).

Np.: PI_NUMBER, AGE_OF_CONSENT, NUMBER_OF_THREADS.

Zastosowanie

Nazywanie stałych.

Konwencja zapisu nawiasów klamrowych

Nawiasy klamrowe są częstym elementem kodu w języku C#. Definiują zasięgi przestrzeni nazw, klas, metod oraz instrukcji warunkowych i pętli. W odróżnieniu od większości instrukcji, po nawiasie klamrowym nie stawia się średnika. W języku C# przyjęło się, że nawias klamrowy stawiamy w kolejnej linii niż instrukcja której dotyczy. Visual Studio domyślnie pomaga w utrzymaniu tej konwencji.

Np.:

```
namespace HelloWorld
{
        wnętrzePrzestrzeniNazw;
}
```

```
public class Program
{
        wnętrzeKlasy;
}
```

```
public static void Main(string[] args)
{
        instrukcjeMetody;
}
```

```
if(a>b)
{
        instrukcjeWykonywaneWPrzypadkuSpełnieniaWarunku;
}
```

```
while(a>b)
{
        instrukcjeWykonywaneWPrzypadkuSpełnieniaWarunku;
        a--;
}
```

Konwencja tworzenia nowych metod

Pomiędzy kolejnymi metodami umieszcza się maksymalnie jedną linię odstępu (pustą linię). Stosowanie większych odstępów zmniejsza czytelność kodu. Powoduje jego niepotrzebne wydłużenie. To z kolei utrudnia sprawną nawigację po kodzie.

Konwencja pisania modyfikatorów dostępu

Zaleca się pisanie modyfikatora dostępu za każdym razem, nawet kiedy nie jest on wymagany. Standardowym i domyślnym modyfikatorem dostępu dla większości elementów języka C# jest *private* lub *internal*. W niektórych przypadkach można więc je pominąć, bez wpływu na funkcjonowanie programu. Zaleca się jednak zapisywanie ich w postaci jawnej, aby zwiększyć przejrzystość kodu. Związane jest to ze stosowaniem wielu słów kluczowych przed nazwami metod (takich jak *static*, czy typ zwracany przez metodę). Zapisanie modyfikatorów dostępu w postaci jawnej ułatwia przeglądanie kodu w poszukiwaniu metod o określonym typie dostępu.

Konwencja tworzenia klas

Każda klasa powinna znajdować się w oddzielnym pliku (każdy plik powinien zawierać tylko jedną klasę). Język C# umożliwia umieszczenie wielu klas w jednym pliku. W miarę rozrastania się programu zmniejsza to jednak jego czytelność i utrudnia nawigację. Visual Studio umożliwia łatwe przeniesienie utworzonych już klas do nowych plików, przy pomocy Szybkich akcji.

Konwencja stosowania typów w nazwach

W odróżnieniu od niektórych innych języków programowania, w języku C# w większości przypadków nazwy elementu **NIE** rozpoczyna się od jego typu (ani skrótu od nazwy typu elementu). Od tej zasady występuje jeden wyjątek, a mianowicie interfejs. Nazwy interfejsów zawsze rozpoczyna się od wielkiej litery „i" („I", jak *Interface*).

Np.:

```
public interface IProgram
{
}
```