



Akademia C# 3

Od Podstaw





Joanna Piątek



V rok Informatyki na W-4



Programista .NET w firmie Comarch



Członek koła EKA.NET od 3 lat

Email: joan.piat@wp.pl



0 Agenda

- 🔍 Kolekcje
- 🔍 Klasa i jej elementy
- 🔍 Tworzymy obiekty
- 🔍 CitizenFormatter
- 🔍 „static”



1 Wprowadzenie do kolekcji

Możliwości:

- Zgromadzenie wielu elementów o tym samym typie
- Dostęp do elementów i operacje na nich

Podstawowe rodzaje:

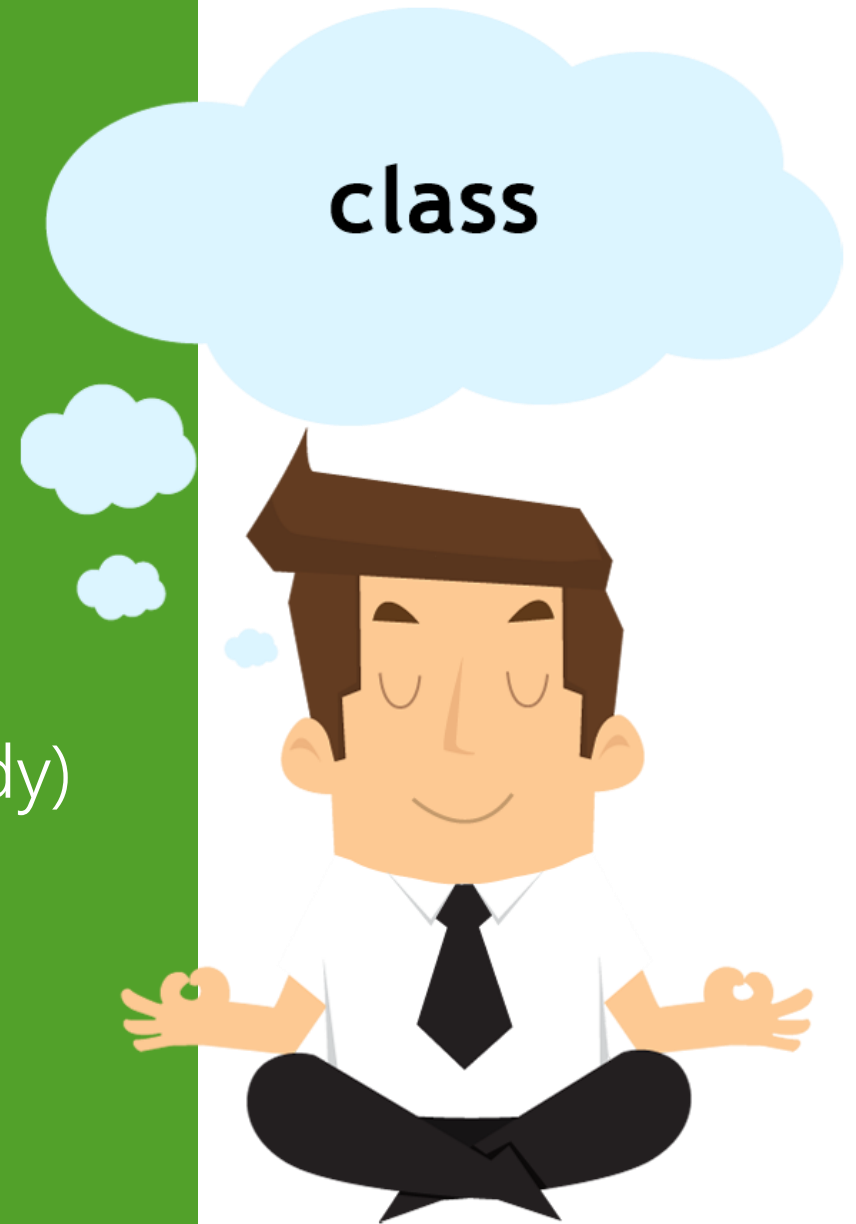
- Tablica – rozmiar deklarowany w momencie utworzenia
- Lista – rośnie i rośnie

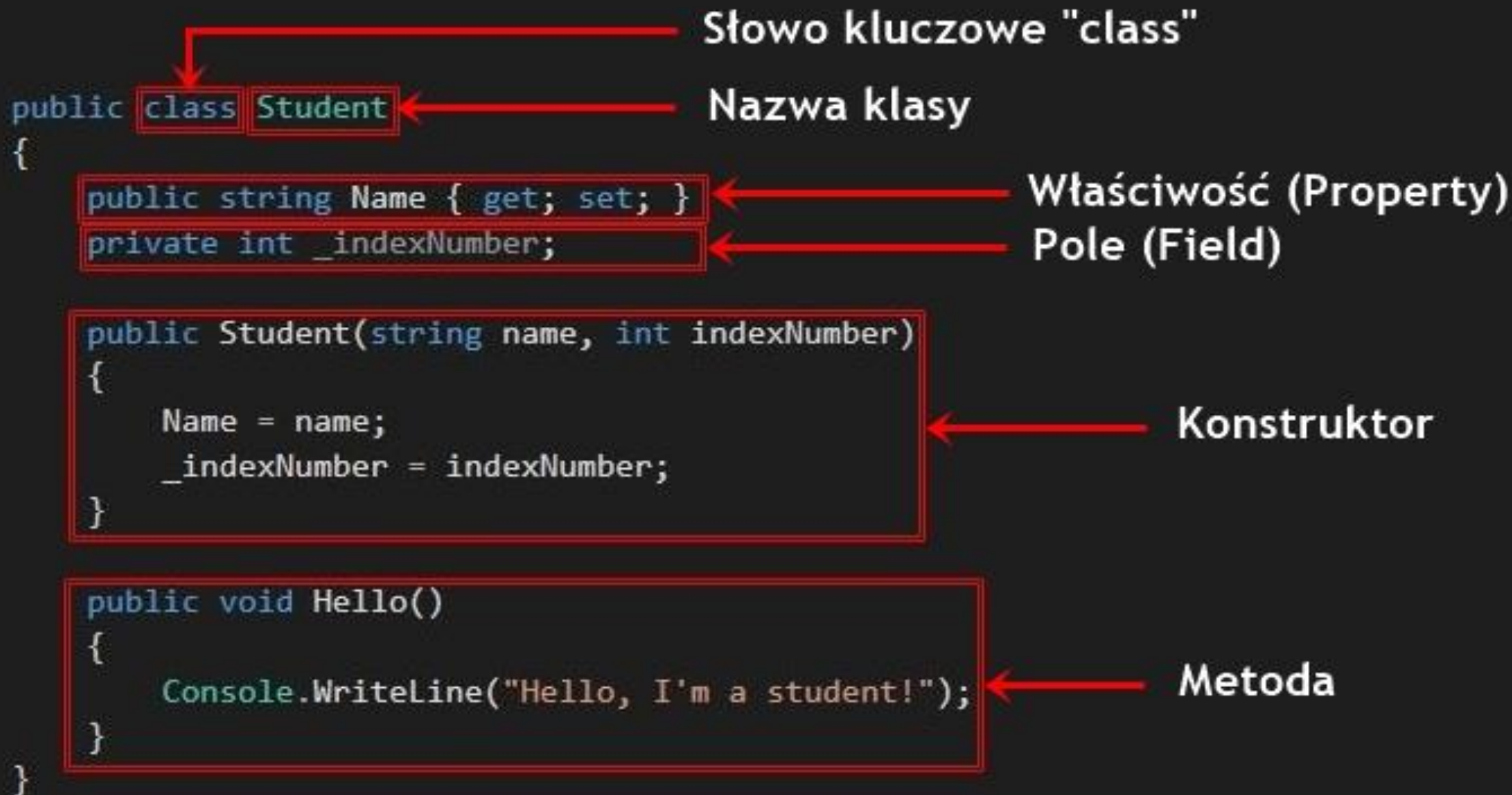


DEMO – Introduce Collections

2 Co to jest klasa?

- Typ danych
- Zawsze ma konstruktor
- Może mieć właściwości
- Może mieć określone zachowania (metody)
- Deklaracja klasy ->





The diagram illustrates the components of a C# class definition for a `Student` class. Red arrows point from descriptive labels to specific parts of the code:

- Słowo kluczowe "class"**: Points to the `class` keyword.
- Nazwa klasy**: Points to the `Student` class name.
- Właściwość (Property)**: Points to the `public string Name { get; set; }` property declaration.
- Pole (Field)**: Points to the `private int _indexNumber;` field declaration.
- Konstruktor**: Points to the `public Student(string name, int indexNumber)` constructor method.
- Metoda**: Points to the `public void Hello()` method.

```
public class Student
{
    public string Name { get; set; }
    private int _indexNumber;

    public Student(string name, int indexNumber)
    {
        Name = name;
        _indexNumber = indexNumber;
    }

    public void Hello()
    {
        Console.WriteLine("Hello, I'm a student!");
    }
}
```

Konstruktory

Definiowane:

- Może przyjmować parametry
- Można zdefiniować ciało

```
public Student(string name, int indexNumber)
{
    Name = name;
    _indexNumber = indexNumber;
}

public Student()
{
    Name = "Heniu";
    _indexNumber = 123123;
}
```

Domyślny:

- Kiedy nie zdefiniujemy żadnego, tworzony automatycznie
- Nie widać go w kodzie, ale można wywołać
- Nie przyjmuje parametrów
- Ciało jest puste

```
public Student()
{
}
```


Modyfikatory dostępu

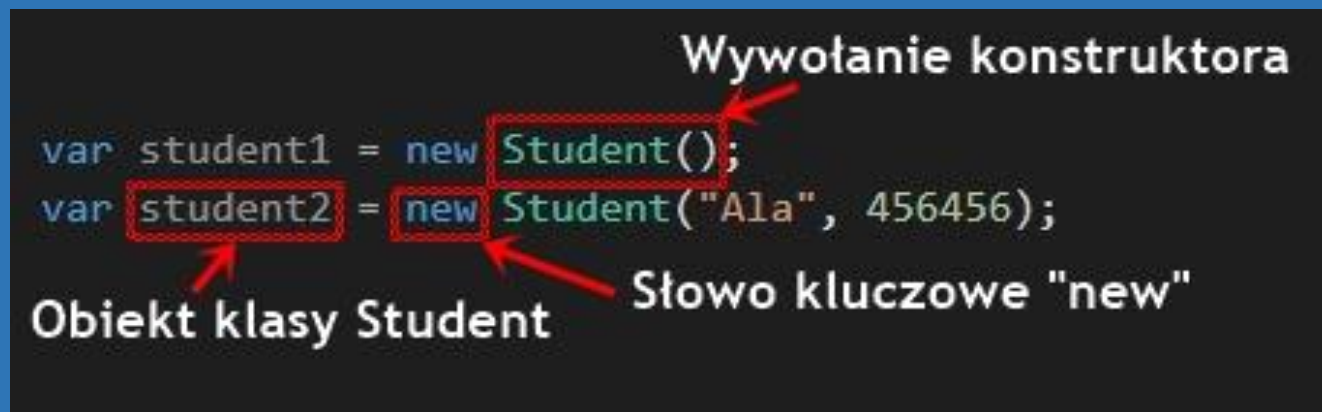
- Public – dostępny zawsze
- Internal – dostępny w ramach projektu VS
- Protected – dostępny wewnątrz klasy i dla klas potomnych
- Private – dostępny tylko wewnątrz klasy



DEMO – Klasa Citizen

3 Co to jest obiekt?

- Instancja klasy
- Ma takie właściwości i metody, jakie zostały zdefiniowane w klasie
- Można utworzyć wiele obiektów tej samej klasy
- Tworzenie obiektu przez konstruktor:



The diagram shows two lines of C# code on a dark background. The first line is `var student1 = new Student();` and the second line is `var student2 = new Student("Ala", 456456);`. Annotations with red arrows point to specific parts: "Wywołanie konstruktora" points to the `Student()` call in the first line; "Obiekt klasy Student" points to the `student2` variable in the second line; and "Słowo kluczowe 'new'" points to the `new` keyword in the second line.

```
var student1 = new Student();  
var student2 = new Student("Ala", 456456);
```

Wywołanie konstruktora

Obiekt klasy Student

Słowo kluczowe "new"

DEMO – Obiekt Citizen

4 Task 1. - CitizenFormatter

- Stwórz nową klasę – CitizenFormatter
- Stwórz w niej metodę „Format”, która:
 - Jako argument przyjmuje obiekt typu Citizen,
 - Zwraca stringa, który jest zdaniem opisującym dostarczony argument



CitizenFormatter – przykładowa implementacja

```
public class CitizenFormatter
{
    public string Format(Citizen citizen)
    {
        return "Obywatel nazywa się " + citizen.Name
            + ", zarabia " + citizen.Income + " zł miesięcznie, a jego stawka podatkowa wynosi "
            + citizen.TaxRate*100 + "%.";
    }
}
```

5

Metody statyczne - „static”

Cechy:

- Można ich używać nie tworząc obiektu klasy, do której należy metoda

```
var formattedCitizen = CitizenFormatter.Format(citizen);
```

- Nie można używać na rzecz stworzonego obiektu

```
var formatter = new CitizenFormatter();  
var formattedCitizen = formatter.Format(citizen);
```

Kiedy używać:

- Kiedy potrzebujemy metody, która nie operuje na strukturach klasy, tylko na dostarczonych argumentach

Zadanie domowe 1. – Lista obywateli

- W klasie Main stwórz metodę, która (w kolejności):
 1. Tworzy nową listę obywateli
 2. Wyświetla tę listę w konsoli lub oznajmia, że lista jest pusta
 3. Pyta, czy dodać nowego obywatela – jeśli tak, pyta o imię i dodaje go do listy
 4. Pyta, czy zakończyć program – jeśli nie, wraca do p. 2.
- Podpowiedź – warto wykorzystać elementy:
 - Pętla do-while, CitizenFormatter, wypisywanie komunikatów na konsolę i sczytywanie z niej znaków

Zadanie domowe 2. - Punkt

- Stwórz klasę *Point* do obsługi punktów na płaszczyźnie. Klasa ma zawierać:
 - 2 pola – współrzędne x i y (jaki typ zmiennej wykorzystasz dla każdej współrzędnej?)
 - Konstruktor, przyjmujący 2 argumenty – współrzędne punktu
 - Metody:
 - *Move*, która przesuwa punkt w osi x i y o podane wartości (jaki typ argumentów wybierzesz dla tej metody?)
 - *Show*, która wypisuje na konsolę aktualne współrzędne



Pytania?

Źródła

1. <https://msdn.microsoft.com/pl-pl/library/ms173109.aspx>
2. [https://msdn.microsoft.com/pl-pl/library/ebca9ah3\(v=vs.90\).aspx](https://msdn.microsoft.com/pl-pl/library/ebca9ah3(v=vs.90).aspx)
3. <http://www.freepik.com/>
4. <http://www.1designshop.com/stock-photos/business-clip-art/>



Dziękuję za uwagę!