giganci programowania

Materiały powtórzeniowe

Dziedziczenie

Witaj w materiałach powtórzeniowych przygotowanych specjalnie dla Ciebie. Znajdziesz tutaj powtórkę materiału z ostatniej lekcji oraz zadania do wykonania dla chętnych. Z pewnością sobie poradzisz. Powodzenia!

Cel lekcji

Celem lekcji jest pogłębienie wiedzy w tematyce programowania obiektowego. Przedstawione zostaną zagadnienia takie jak dziedziczenie, nadpisywanie metod oraz elementy statyczne w programowaniu.

Konstruktor

Konstruktor jest to specjalna metoda, która wykonuje się w chwili tworzenia obiektu. Jej zadaniem jest inicjalizacja pól obiektu. Musi nazywać się identycznie jak klasa. Nic nie może zwracać, musi być publiczny. Konstruktor wykorzystujemy zazwyczaj do ustawienia pewnych startowych właściwości w klasie oraz do przypisania przekazanych parametrów do własności klasy.

```
public class Pojazd
{
	public int Moc;
	public string Kolor;
	public string Marka;

	public Pojazd(int moc, string kolor, string marka)
	{
		Moc = moc;
		Kolor = kolor;
		Marka = marka;
	}
}
```

Dziedziczenie to kluczowy mechanizm obiektowości. Dziedziczenie pozwala na powielanie funkcjonalności wobec różnych klas w ten sposób nie musimy pisać ciągle tego samego kodu.

Jako przykład weźmy ssaki. Ssaki to duża kategoria zwierząt, w której zawiera się też człowiek. Wszystkie ssaki mają pewne cechy wspólne jak na przykład sposób karmienia potomstwa. Jednak każdy odrębny ssak ma swoje cechy specjalne.
Drugi przykład - Klasa pojazdy opisuje cechy wspólne klas jak samochod czy autobus, ponieważ są one też pojazdami. samochod i autobus jednak mają swoje cechy specjalne. Ustalanie cech wspólnych poszczególnych klas daje sens tworzenia klas bazowych. Nie chciałbyś dla każdego pojazdu czy ssaka powielać tych samych cech, które występują. Jest to esencja modelowania klasy. Zarządzenie taką aplikacją jest sporo łatwiejsze, ponieważ w wypadku dodania nowej metody dla wszystkich ssaków czy pojazdów nie trzeba byłoby robić kopiuj/wklej dla każdej klasy po kolei.

Klasa pochodna dziedziczy po klasie bazowej. W wyniku tej operacji wszystkie pola, metody, właściwości nieoznaczone modyfikatorem **private** są dostępne dla klasy pochodnej. Klasa nie może dziedziczyć po wielu klasach bazowych, czyli w C# dziedziczenie może zajść tylko do jednej klasy bazowej.
Każda klasa domyślnie dziedziczy z klasy ***Object***, która jest podstawową klasą w C#.

```
public class SamochodOsobowy : Pojazd // określenie dziedziczenia
{
	public SamochodOsobowy(int moc, string kolor, string marka) : base(moc, kolor, marka)
	{
	}

	public void UruchomSilnik()
	{
	   Console.WriteLine("Silnik samochodu uruchomiony...");
	}
}
```

Jeżeli klasa bazowa posiada konstruktor z parametrami to wszystkie klasy, które z niej dziedziczą muszę posiadać konstruktor, który wywoła ten z klasy bazowej i przekaże do niej wartości. Robimy to za pomocą składni***: base****(argument1, argument2)* tak jak pokazano na powyższym przykładzie.

Dodatkowa misja Programisty:

Programowanie obiektowe i związane z nim dziedziczenie wraz z pozostałymi elementami nie są prostym zagadnieniem. W ramach zadania domowego należy powtórzyć wszystkie poznane elementy i zastanowić się czy na pewno wszystko rozumiem. Wymyślić sobie jakąś klasę bazową, dopisać kilka klas pochodnych, stworzyć konstruktor, dodać własności i metody.

Przygotować sobie listę z pytaniami do nauczyciela co jest dla mnie niejasne, czego jeszcze nie rozumiem.