

9

Learning resources:



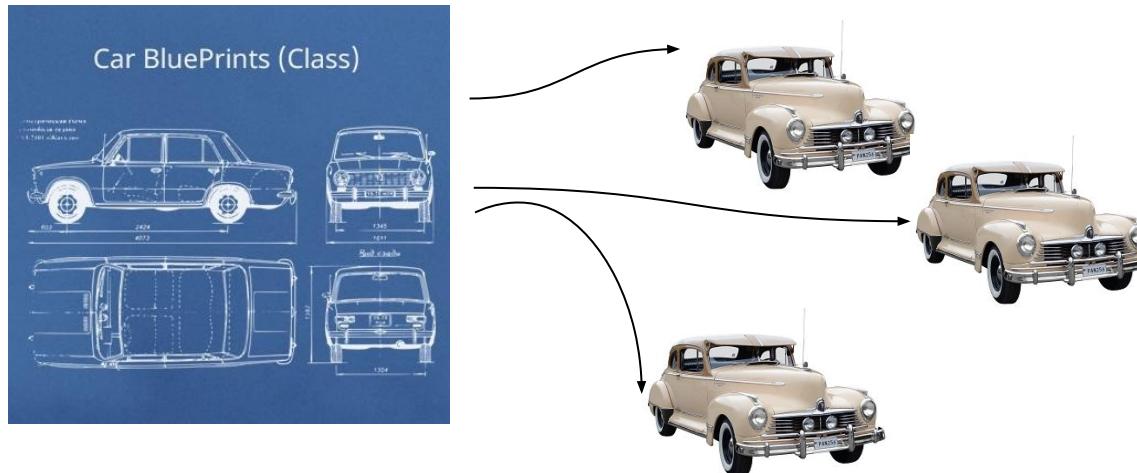
<https://www.sololearn.com/Play/CSharp/>

<https://www.tutorialspoint.com/csharp/>

<http://www.tutorialsteacher.com/csharp/csharp-tutorials>

Pakartokime - Kas yra klasė?

- Klasė – tai „brėžinys“, apibūdinantis kokius duomenis turės, ir kaip galės elgtis objektas.
- Klasė – tai atskiras, mūsų sukurtas objekto tipas.
- Objektas – tai atskiras klasės atvejis.



Pakartokime - Kas sudaro klasę?

Access modifier

Constructor

this points to self

Parameter (variable)

Class variable

Local variable

```
class Car {  
    public string Name;  
    public int Year;  
    private bool IsInsured;  
  
    public Car(string Name, int Year, bool IsInsured) {  
        this.Name = Name;  
        this.Year = Year;  
        this.IsInsured = IsInsured;  
    }  
  
    private void StartEngine() {  
        int maxSpeed = 220;  
    }  
  
    public void Drive() {  
    }  
}
```

Fields

Members

Methods

Pradedam daryt žaidimą!

OPEN FILE : "C# raimundas-klases-praktika.pdf"

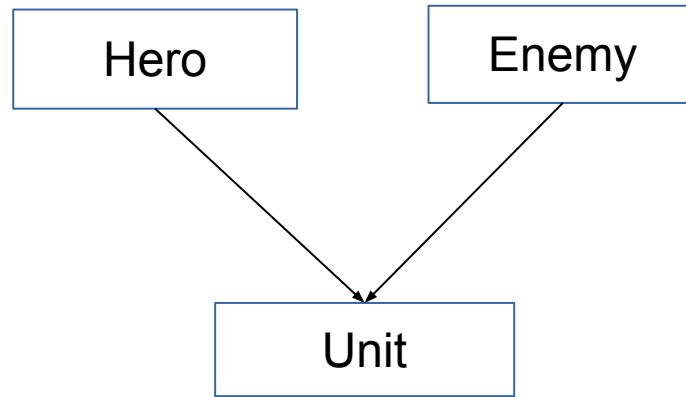
Inheritance

Dažnai mes susiduriame su objektais, kurie turi panašumų. PVZ:

```
class Hero {  
    private int x;  
    private int y;  
    private string name;  
  
    public Hero(int x, int y, string name) {  
        this.x = x;  
        this.y = y;  
        this.name = name;  
    }  
  
    public void MoveRight() {  
        x++;  
    }  
  
    public void MoveLeft() {  
        x--;  
    }  
  
    public void PrintInfo() {  
        Console.WriteLine($" Hero {name} is at {x}{y}");  
    }  
}  
  
class Enemy {  
    private int id;  
    private int x;  
    private int y;  
    private string name;  
  
    public Enemy(int id, int x, int y, string name) {  
        this.id = id;  
        this.x = x;  
        this.y = y;  
        this.name = name;  
    }  
  
    public void MoveDown() {  
        y++;  
    }  
  
    public void PrintInfo() {  
        Console.WriteLine($" Enemy {name} is at {x}{y}");  
    }  
  
    public int GetId() {  
        return id;  
    }  
}
```

Inheritance

Būtų gerai tuos panašumus parašyti vieną kartą, ir pasakyti, kad abu objektai juos turi.



Šitas metodas vadinasi Inheritance!

Galima sakyti: Hero IS-A Unit. Enemy IS-A Unit

Inheritance

Unit is **Base, parent, Superclass** of Hero class.

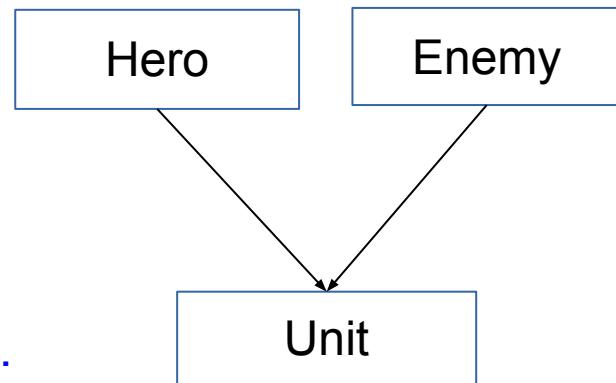
Unit is **Base, parent, Superclass** of Enemy class.

Hero is **derived from; child of; Subclass of** Unit class.

Enemy is **derived from; child of; Subclass of** Unit class.

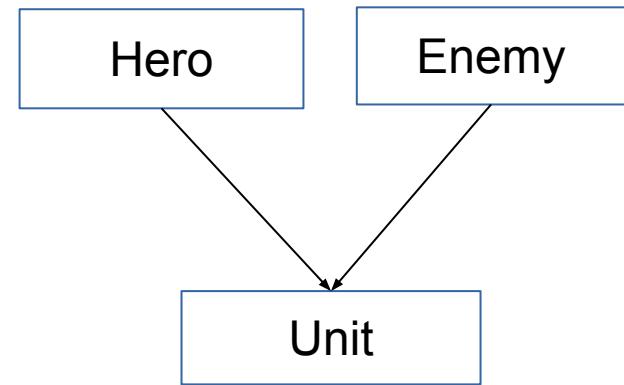
Hero **extends** Unit class.

Enemy **extends** Unit class.



Inheritance - sintaksė

```
class <base_class> {  
    ...  
}  
  
class <derived_class> : <base_class>  
{  
    ...  
}
```



Derived class turi viską ką turi Base class!
(bet gali keisti tik **public** ir **protected** laukus, negali keisti **private**!)

Inheritance - konstruktoriaus sintaksė

```
class Parent {  
    public Parent(int parentNr) {  
    }  
}  
  
class Child : Parent {  
    public Child(int parentNr, string childName) : base(parentNr) {  
    }  
}
```

Protected

protected yra skirtas inheritance objektams.

Child mato **protected** ir **public** Parent klasės laukus.

Child nemato **private** Parent klasės laukų.

private – tik klasė gali naudoti.

protected – klasė ir jos vaikai gali naudoti.

public- klasė, jos vaikai, ir visi kiti objektais.

base

- Child klasė gali pasiekti Parent klases public ir protected kintamuosius.
- Kai vardas sutampa – reikia naudoti base raktažodį.
- Base naudojamas pasiekti

```
class Parent {  
    protected string name;  
    public Parent(int nr) {  
    }  
}  
  
class Child : Parent {  
    private string name;  
    public Child(int nr, string name) : base(nr) {  
        this.name = name;  
        base.name = this.name;  
    }  
}
```

Užduotys

- Sukurkite klasę Gyvunas su: private protected ir public kintamaisiais (po vieną).
- Sukurkite klasę Paukstis su: private protected ir public kintamaisiais (po vieną).
- Sukurkite Gyvunas konstruktorių, su 3 kintamaisiais.
- Sukurkite Paukstis konstruktorių su 6 kintamaisiais , 3 iš jų nusiųskite parent klasei.
- Main >> Sukurkite objektą – Gyvunas.
- Main >> Sukurkite objektą – Paukstis.

```
class Parent {  
    public Parent(int nr) {  
    }  
}  
  
class Child : Parent {  
    public Child(int nr, string name) : base(nr) {  
    }  
}
```

- Main >> pabandykite atspausdinti Gyvunas 3 kintamuosius.
- Main >> pabankykite atspausdinti Paukstis 6 kintamuosius.
- Gyvunas.PrintData(); >> pabandykite atspausdinti 3 kintamuosius.
- Paukstis.PrintData(); >> pabandykite atspausdinti 6 kintamuosius.

Links:

https://www.tutorialspoint.com/csharp/csharp_inheritance.htm