

Klasser och objekt

Grundläggande grunder

Upphovsrätt för detta verk

Detta verk är framtaget i anslutning till kursen Inledande programmering med C# vid Linnéuniversitetet.

Du får använda detta verk så här:

Allt innehåll i detta verk av Mats Looch, förutom Linnéuniversitetets logotyp och symbol samt fotografier och figurer, är licensierad under:



Creative Commons Erkännande-IckeKommersiell-DelaLika 2.5 Sverige licens.

<http://creativecommons.org/licenses/by-nc-sa/2.5/se/>

Det betyder att du i icke-kommersiella syften får:

- kopiera hela eller delar av innehållet
- sprida hela eller delar av innehållet
- visa hela eller delar av innehållet offentligt och digitalt
- konvertera innehållet till annat format
- du får även göra om innehållet

Om du förändrar innehållet så ta inte med Linnéuniversitetets logotyp och symbol samt fotografier och figurer i din nya version!

Vid all användning måste du ange källan: "Linnéuniversitetet – Inledande programmering med C#" och en länk till <https://coursepress.lnu.se/kurs/inledande-programmering-med-csharp> och till Creative Common-licensen här ovan.

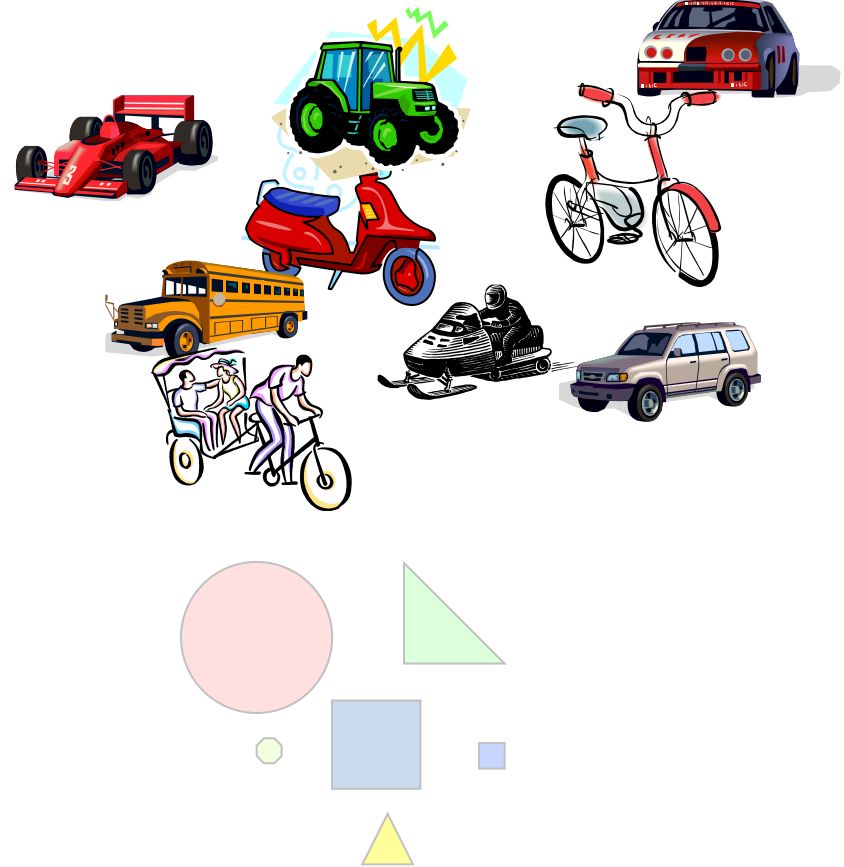
Vad är ett objekt?

Vilka typer av objekt har vi?



Vad är ett objekt?

Vilka typer av objekt har vi?



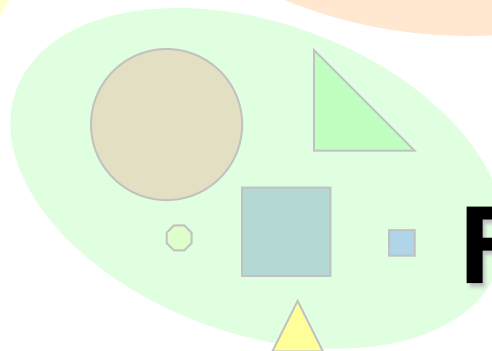
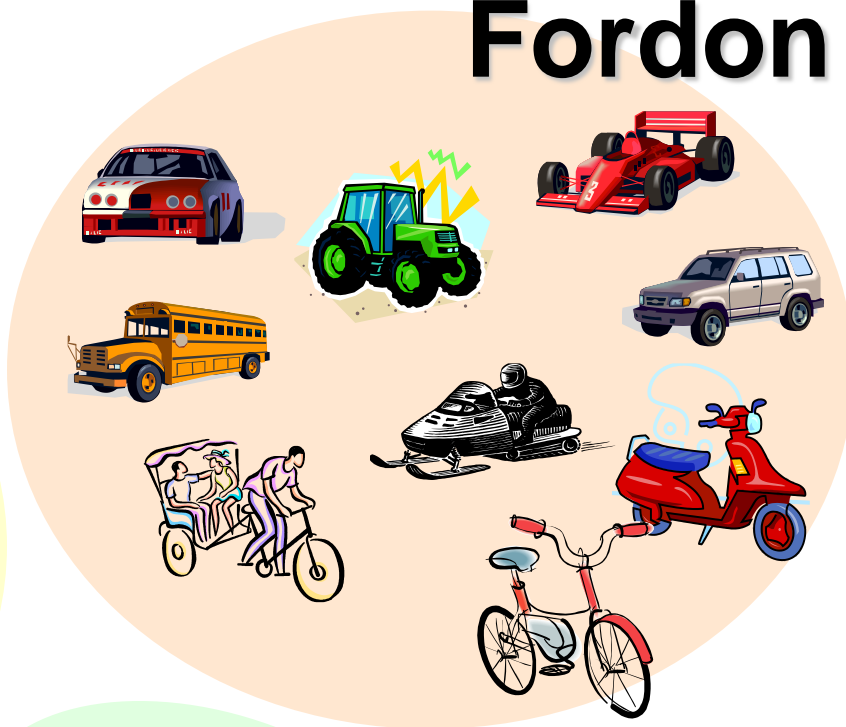
Klassificering av objekt

- eller indelning av objekt i grupper



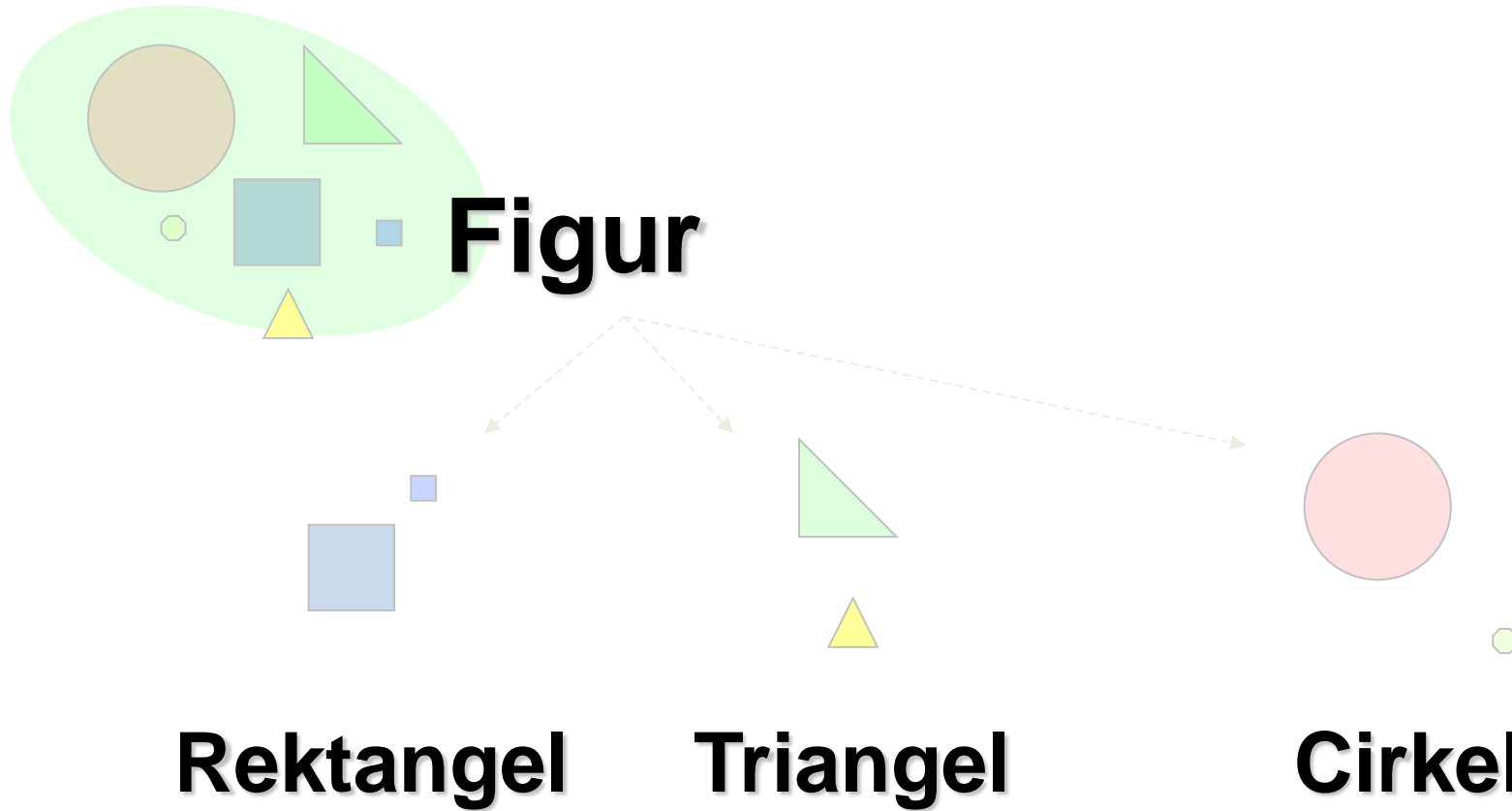
Verktyg

Fordon

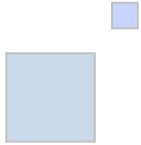


Figur

Klassificering av figurer



Vad utmärker en rektangel?

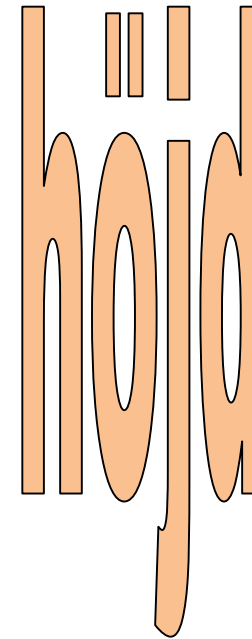


Rektangel

b r e d d a

attribut

färg



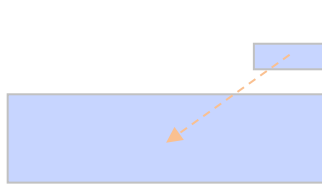
Vad en rektangel har och "kan"

✓ En rektangel **har**:

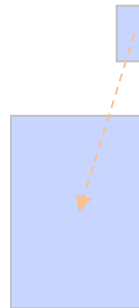
- bredd
- höjd
- färg

} **attribut**

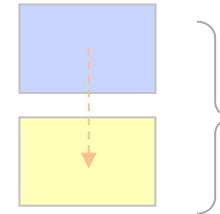
✓ En rektangel **kan**:



- ändra bredd



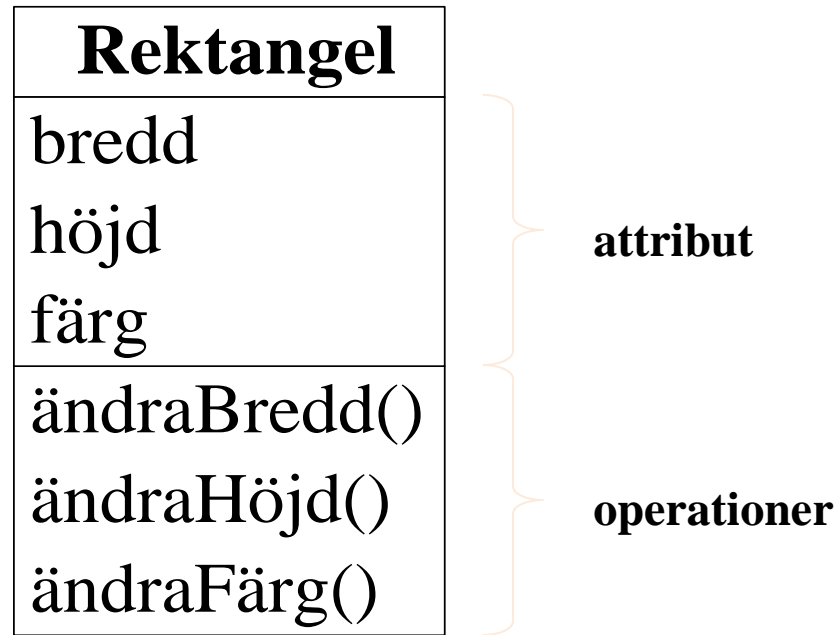
- ändra höjd



- ändra färg

} sätt att ändra på
kännetecken
- **operationer**

Rektangeln som klass



- ✓ Ett klassdiagram beskriver vilka attribut och operationer en klass har.

Klassdefinition i C#

Rektangel
bredd
höjd
färg
ändraBredd()
ändraHöjd()
ändraFärg()

attribut

operationer

```
class Rectangle
{
    private Color _color;
    private int _height;
    private int _width;

    public Rectangle(int width, int height, Color color)
    {
        _width = width;
        _height = height;
        _color = color;
    }

    public Color Color
    {
        get { return _color; }
        set { _color = value; }
    }

    public int Height
    {
        get { return _height; }
        set { _height = value; }
    }

    public int Width
    {
        get { return _width; }
        set { _width = value; }
    }

    public void Display()
    {
        Console.WriteLine("(0), (1), (2)");
        _width, _height, _color.ToString();
    }
}
```

fält

konstruktör


egenskaper


metoder


Rectangle

Class


Fields


 _color: Color


 _height: int

 _width: int


Properties


 Color { get; set; } : Color

 Height { get; set; } : int

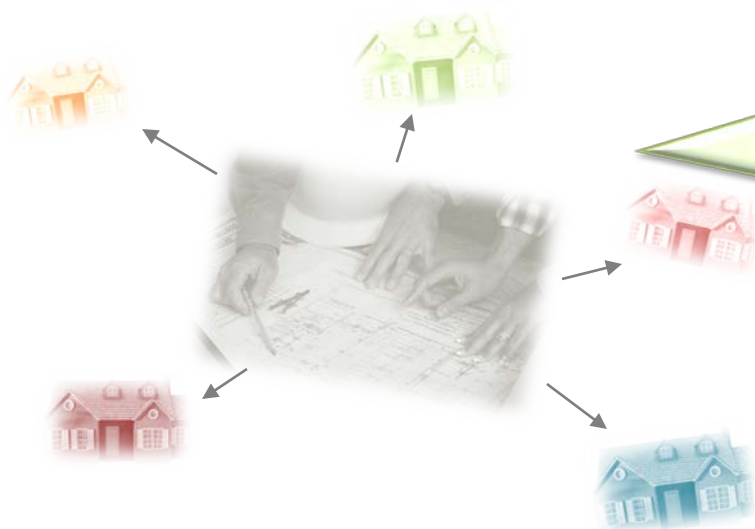
 Width { get; set; } : int

Methods

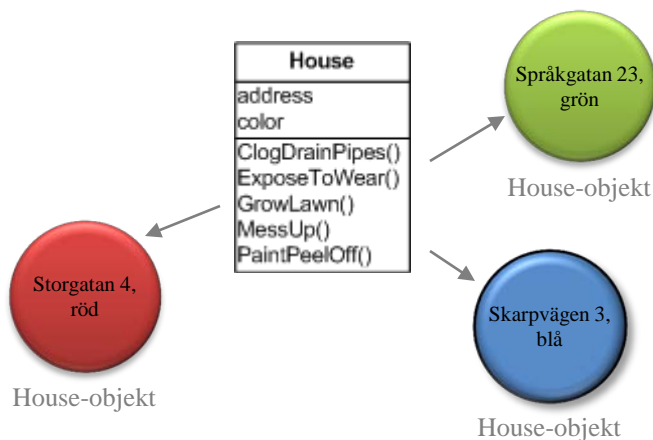
 Display() : void

 Rectangle(int width, int height, Color color)

Du använder en klass för att skapa objekt




När du definierar en klass, definierar du dess medlemmar precis på samma sätt som en ritning definierar hur ett hus kommer att se ut.



Du kan använda samma ritning för att bygga hur många hus som helst, och på samma sätt kan du använda en klass för skapa hur många objekt som helst.

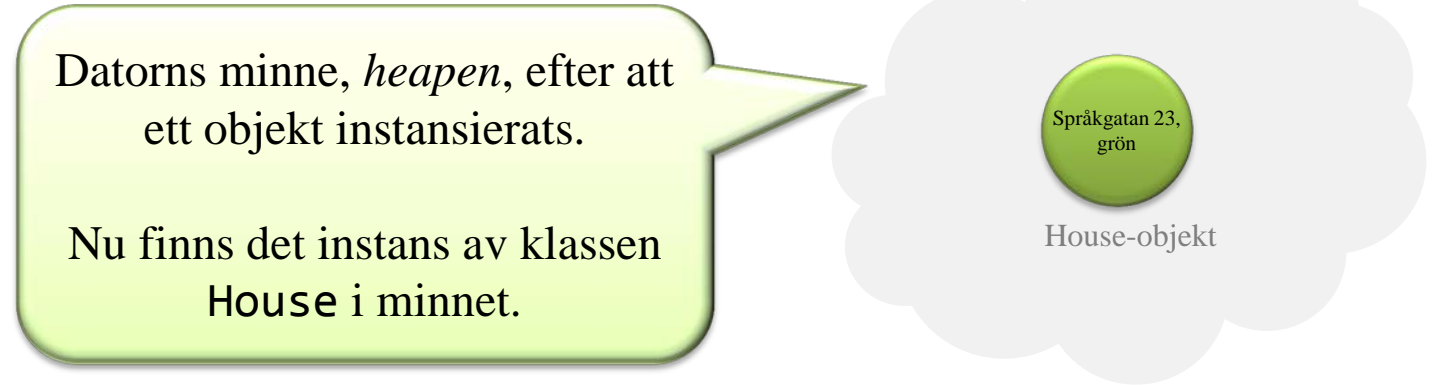
När du skapar ett nytt objekt av en klass...

...kallas det att du skapar en **instans** av den klassen.



Datorns minne,
heapen, innan ett
objekt instansierats.

```
House myHouse = new House();
```



Datorns minne, *heapen*, efter att
ett objekt instansierats.

Nu finns det instans av klassen
House i minnet.

Språkgatan 23,
grön

House-objekt