



UPPSALA
UNIVERSITET

. NET-programmering

Metoder



Synlighet

- **public**
 - synlig där deklarerande namnutrymme är känt
- **private**
 - synlig i deklarerande klass
- **internal**
 - synlig i deklarerande assembly
- **protected**
 - synlig i deklarerande klass och dess underklasser
- **protected internal**
 - synlig i deklarerande assembly samt i underklasser till deklarerande klass som ligger i andra assemblies



UPPSALA
UNIVERSITET

Metoder

```
class MathWizard
{
    public int Square(int x)
    {
        return x * x;
    }
}
```



Statiska metoder

- Statiska metoder är associerade med klassen, inte enskilda objekt
- Använd för att initiera statiska fält

```
class Rectangle
{
    static Color defaultColor;

    public static void ResetColor()
    {
        defaultColor = Color.white;
    }
}
```



- Värdeparametrar
 - "Call by value"
 - Formell parameter är kopia av verklig parameter
- Referensparametrar
 - "Call by reference"
 - Den verkliga parameterns adress skickas
- Out-parametrar
 - Som referensparametrar, men har ej tilldelats något värde innan de skickas
 - Måste tilldelas värde i metoden
 - Får inte användas före tilldelningen

Parametrar

```
void Inc(int x) { x++; }
```

```
void F() {  
    int val = 3;  
    Inc(val);  
}
```

```
void Inc(ref int x) { x++; }
```

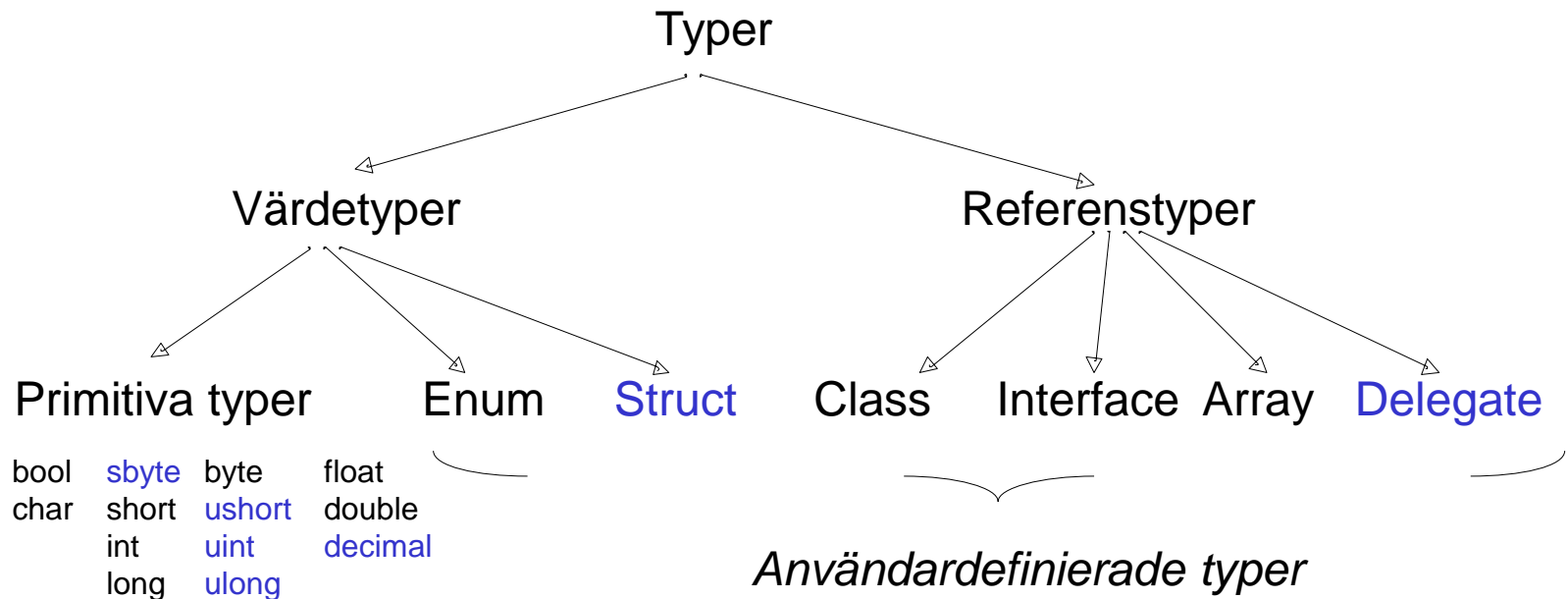
```
void F() {  
    int val = 3;  
    Inc(ref val);  
}
```

```
void Set(out int x) { x = 3; }
```

```
void F() {  
    int val;  
    Set(out val);  
}
```



Typhierarki



Blåmarkerade typer saknar motsvarighet i Java



UPPSALA
UNIVERSITET

C# - Värde typ

```
private void DoStuff(int a)
{
    a = 2;
}

int x = 5;
DoStuff(x);

Console.WriteLine("x == {0}", x);
```



UPPSALA
UNIVERSITET

Övning

```
class Program
{
    static void DoStuff(int a)
    {
        a = 2;
    }

    static void Main(string[] args)
    {
        int x = 5;
        DoStuff(x);
        Console.WriteLine("x == {0}", x);

        Console.ReadKey();
    }
}
```




UPPSALA
UNIVERSITET

C# - Referenstyp

```
private void DoStuff(int[] a)
{
    a[0] = 2;
}

int[] x = {5};
DoStuff(x);

Console.WriteLine("x == {0}", x[0]);
```



UPPSALA
UNIVERSITET

C# - Referenstyp

```
private void DoStuff(int[] a)
{
    a = new int[]{42};
}

int[] x = {5};
DoStuff(x);

Console.WriteLine("x == {0}", x[0]);
```



UPPSALA
UNIVERSITET

C# - ref (Värdetyp)

```
private void DoStuff(ref int a)
{
    a = 2;
}

int x = 5;
DoStuff(ref x);

Console.WriteLine("x == {0}", x);
```



UPPSALA
UNIVERSITET

C# - ref (Referenstyp)

```
private void DoStuff(ref int[] a)
{
    a = new int[]{42};
}

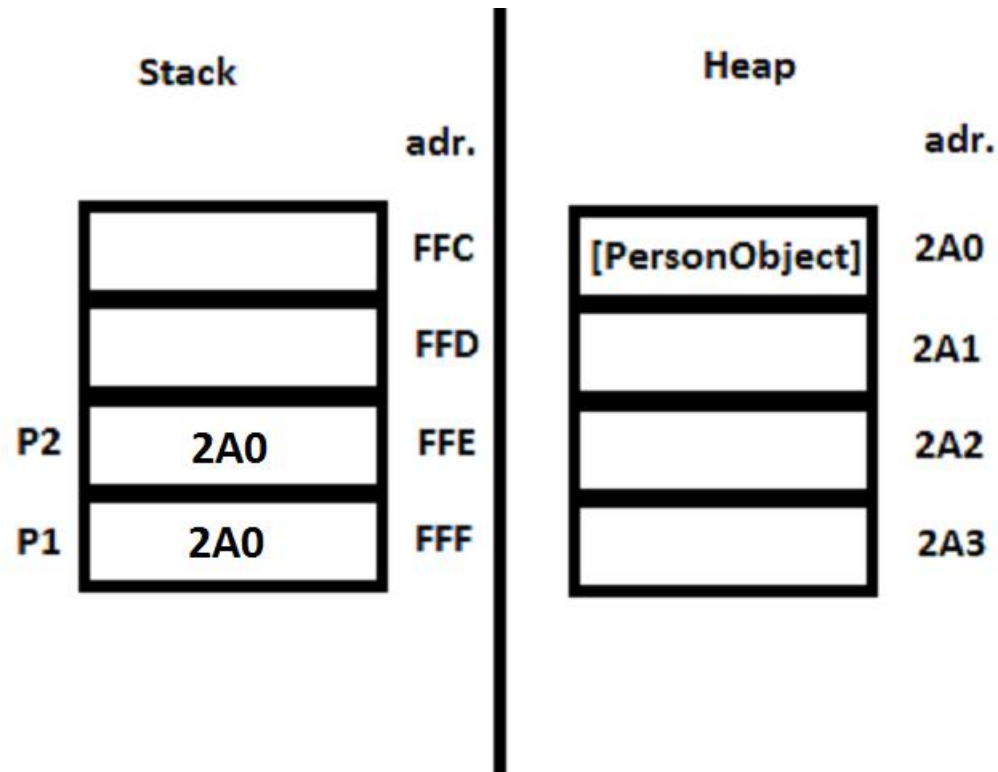
int[] x = {5};
DoStuff(ref x);

Console.WriteLine("x == {0}", x[0]);
```



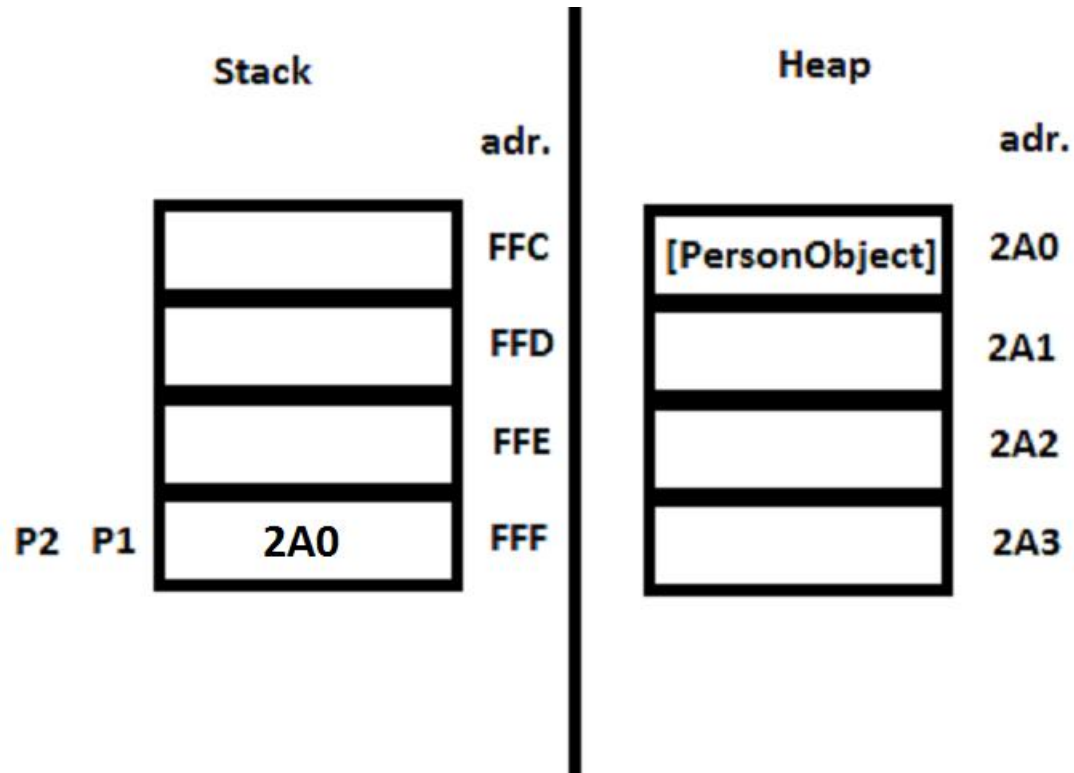
C# - ref

Utan ref:





Med ref:





C# - out

- Används på samma sätt som ref
- Kräver att parameteren får ett värde i metoden
- Kräver inte att parameteren har ett värde
- Kan användas ifall man behöver flera returvärden



UPPSALA
UNIVERSITET

C# - out

```
private void Divide(int x, int y,  
                    out int q, out int r)  
{  
    q = x / y;  
    r = x % y;  
}  
  
int a;  
int b;  
Divide(11, 3, out a, out b);  
  
Console.WriteLine("11/3 == {0} + {1}/3", a, b);
```



Varierande antal parametrar

- Sista parameteren kan vara en array
- Nyckelordet **params**
- Metoden tar valfritt antal parametrar

```
void Add(out int sum, params int[] val)
{
    sum = 0;
    foreach (int i in val) sum += i;
}
```

- Kan till exempel anropas så här

```
Add(out sum, 1, 5, 8, 4, 12, 7);
```



Kan inte
användas
med **ref**
och **out**

Överlagrade metoder

- Kan ha samma namn förutsatt att de har
 - olika antal parametrar
 - olika typer av parametrar
 - olika sätt att skicka parametrar (värde, ref/out)
- Kompilatorn måste kunna välja rätt metod entydigt, annars kompileringsfel
- Får inte skilja enbart på returtyp, params, ref/out



Return

- Funktionsmetoder måste avslutas med return och något värde
 - `return listOfStudents;`
- Void-metoder kan avslutas med return utan värde
 - `return;`
- Main()-metoden kan returnera en integer

```
return 0;    //inga fel  
return 1;    //avslut pga. fel
```