# OOP in C#

Kristian Kristensen & Jakob Andersen
Microsoft Student Partners

## Agenda

- Motivation
- Basic Syntax
- Advanced Features
- Development Environment
- Future versions

## Motivation

- **Easy to learn C# after Java and vice versa**
- Similar architecture
    - Java Runtime ~ Common Language Runtime
    - Java Bytecode ~ Intermediate Language
- Similar syntax
    - Borrows from C/C++ and from each other
- Similar typesystem
- **Keep C# in mind for future projects but keep your focus on Java now to avoid confusion**

## Basic Syntax

- Many constructs use same syntax
  - if/else construct
  - for,while and do loop construct
  - Zerobased arrays with indexers( myArray[3] )
  - Dot-operator is always used (no -> or :: operators)
  - And many more...

## Difference in syntax

- Example of a small difference in syntax: foreach/for

**C#:**
```
int[] intlist = {1,2,3,4,5};
foreach(int i in intlist){
    System.Console.Write(i);
}
```

**Java(from 1.5):**
```
int[] intlist = {1,2,3,4,5};
for(int i : intlist){
    System.out.print(i);
}
```

## Difference in syntax

- Inheritance is done using ”:”

**C#:**
```
public class A : ISerializable, MinKlasse{
        .....
}
```

**Java:**
```
public class a extends MinKlasse implements ISerializable{
        .....
}
```

## Access modifiers

| **C#** | **Java** |
|---|---|
| private | private |
| public | public |
| internal | protected |
| protected (like in C++) | N/A |
| internal protected | N/A |

## Polymorphism

- Classes are not virtual by default like in Java

```
public class A{
        public virtual void foo(){
                ....
        }
}

public class B : A{
        //Hides implementation on A
        public override void foo(){
                ....
        }
}

public class C : B{
        public new void foo(){
                ....
        }
}

C bar = new C();
bar.foo(); //Calls C's foo()
((B)bar).foo(); //Calls B's foo()
((A)bar).foo(); //Calls B's foo()
```

## Exception handling

- Exceptions must be caught in Java not in C#
- Java supports throws keyword for methods. Not neccesary in C# because exceptioncatching is mandatory

## String formatting

- Java uses C-printf syntax

```
formatter.format(”%s er det samme som %<s hvilket ikke er %s”, ”to”, ”en”);
```

- C# uses custom syntax:

```
string.Format(”{0} er det samme som {0} hvilket ikke er {1}”, ”to”, ”en”);
```

- Større fleksibilitet da man kan bruge alle parametre på et givet tidspunkt

## Keywords: out and ref

- Pass arguments by reference
- Output parameters

```
public void Swap(ref int x, ref int y){
        int z = x;
        x = y;
        y = z;

}

public void Fill(out int x, out string y){
        x = 10;
        y = ”foo”;
}
```

```
int x = 7;
int y = 14;

this.Swap(ref x, ref y);

x == 14; //true
y == 7; //true
```

```
int x;
string y;
this.Fill(out x, out y);
x == 10; //true
y == ”foo”; //true
```

## Constructors

- Optional parameters in constructors:

```
public MyClass() : this(true){}
public MyClass(bool foo){ ... }
```

- Call to base class using base keyword:

```
public MyClass(string foo, bool bar) : base(bar){
            ...
}
```

# Properties

**Java:**

```
public class Car{
        private string _make;

        public string getMake(){
                return _make;
        }

        public void setMake(string make){
                _make = make;
        }

}
```

Usage:

```
Car c = new Car();
c.setMake(”Ford”);
if(c.getMake() == ”Ford”){
        ...
}
```

**C#:**

```
public class Car{
        private string _make;
        public string Make{
                get{ return _make; }
                set{ _make = value; }
        }
}
```

Usage:

```
Car c = new Car();
c.Make = ”Ford”;
if(c.Make == ”Ford”){
        ...
}
```

## Operator overloading

- Not supported in Java, operators used on custom objects

```
public class Score{
    int value;
    public Score (int score) {
        value = score;
    }
    public static bool operator == (Score x, Score y) {
        return x.value == y.value;
    }
    public static bool operator != (Score x, Score y) {
        return x.value != y.value;
    }
}
```

```
Score a = new Score(5);
Score b = new Score(5);
a == b; //true
a != b; //false
```

## Development Environment

- Visual Studio.NET is the defacto standard
  - Alternative editor of choice with .NET SDK
  - Open source IDE alternative SharpDevelop
- Java has plenty of environments
  - Eclipse
  - JBuilder
  - ...

## Future versions

- C# 2.0 Released 7. November
- C# 3.0 incorporates data access

## Questions?

Feel free to contact us:
kk@cs.aau.dk – Kristian Kristensen SW7
jta@cs.aau.dk – Jakob Andersen DAT5