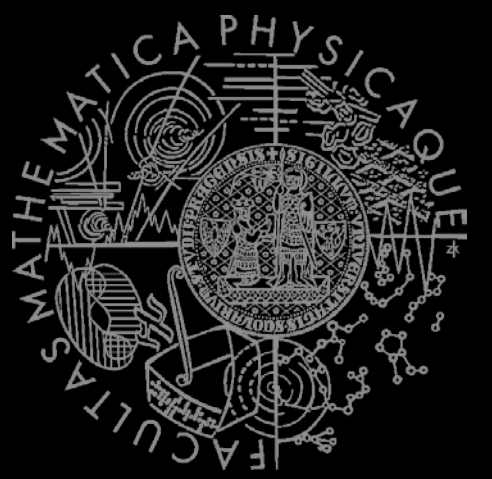
Faculty of Mathematics and Physics
Charles University in Prague
3rd March 2016




C# Made Easy!

# Programming II

## Workshop 2 – Gentle C# + OOP Introduction

# Workshop 2
## Outline

1. Test
2. C#, OOP, UML
3. Homework

# Test 02

## Quick Warm up!

**Find the test here (no-ads):**

https://goo.gl/qO9uhA

**Permanent link:**

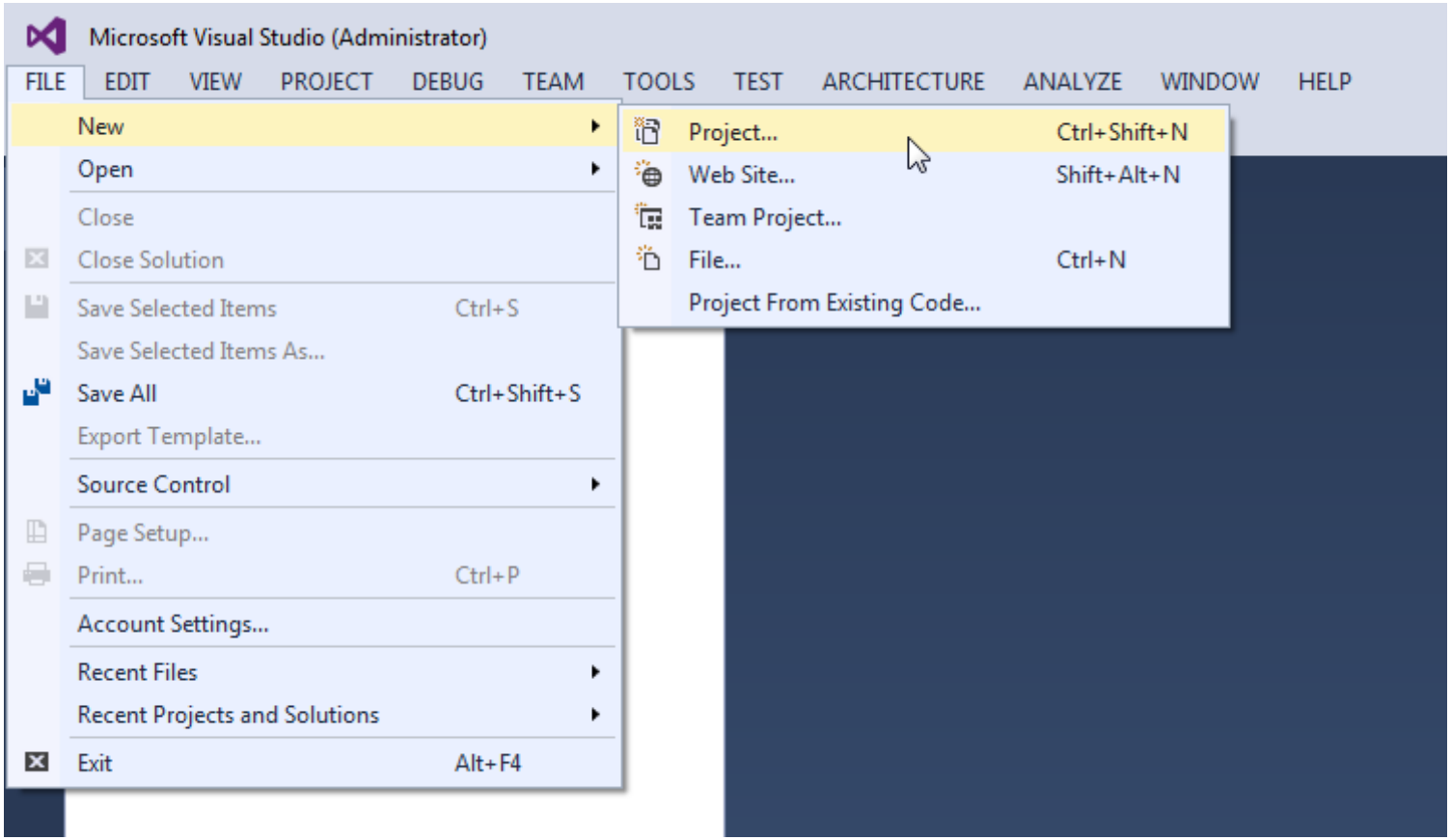
https://docs.google.com/forms/d/1it-GI5EdCcQNzDyhNkJdk-Ehx0iGSMdgVc_dGQVMGXM/viewform

**Time for the test:**

10 min

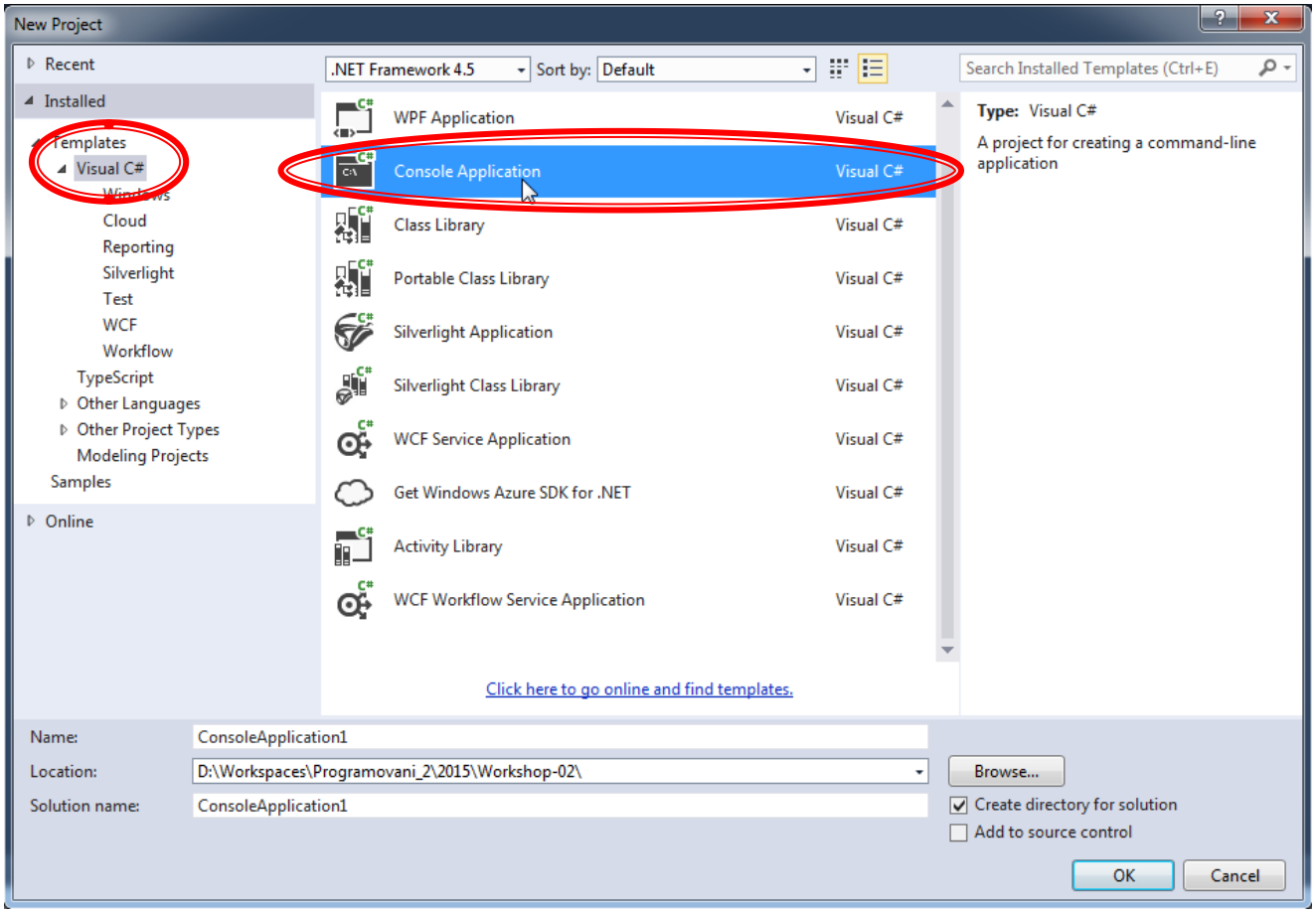
# C#
# Visual Studio 2015

## 1. Creating new Solution and Project



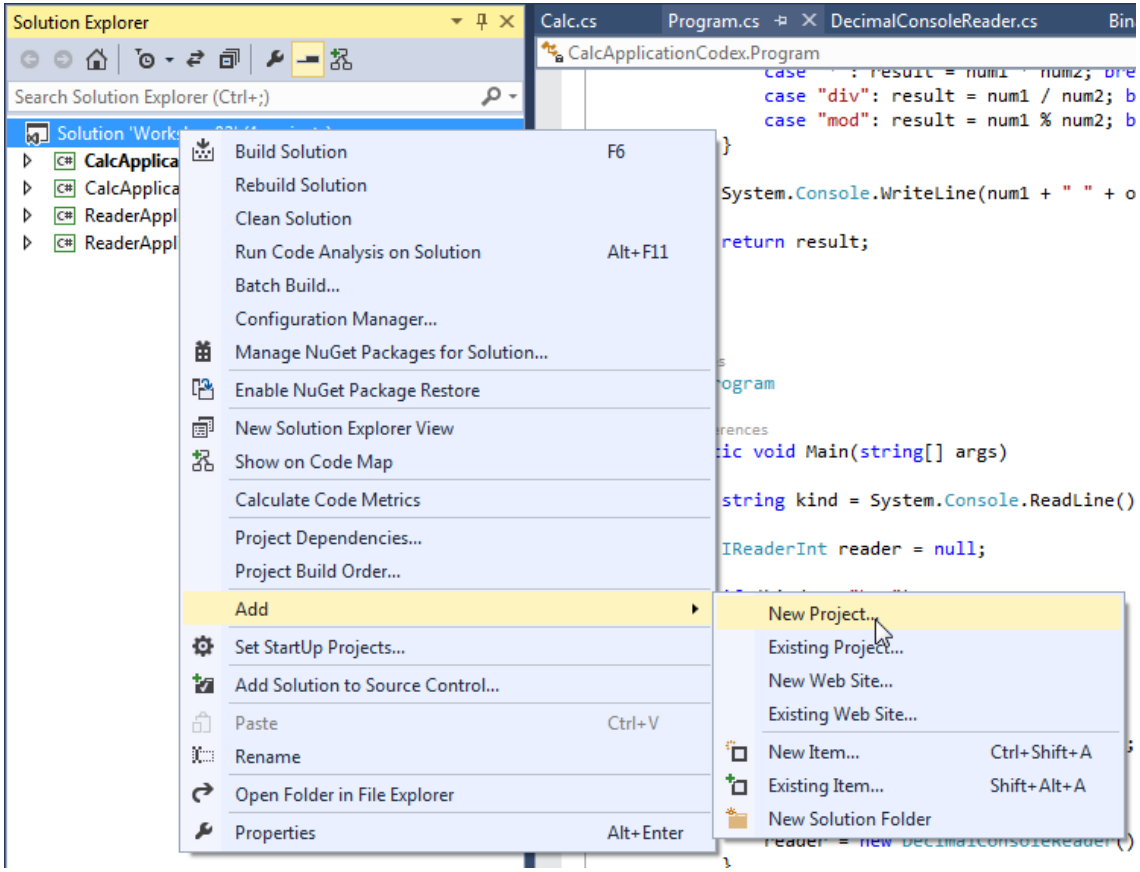
# C#
# Visual Studio 2015

## 1. Creating new Solution and Project



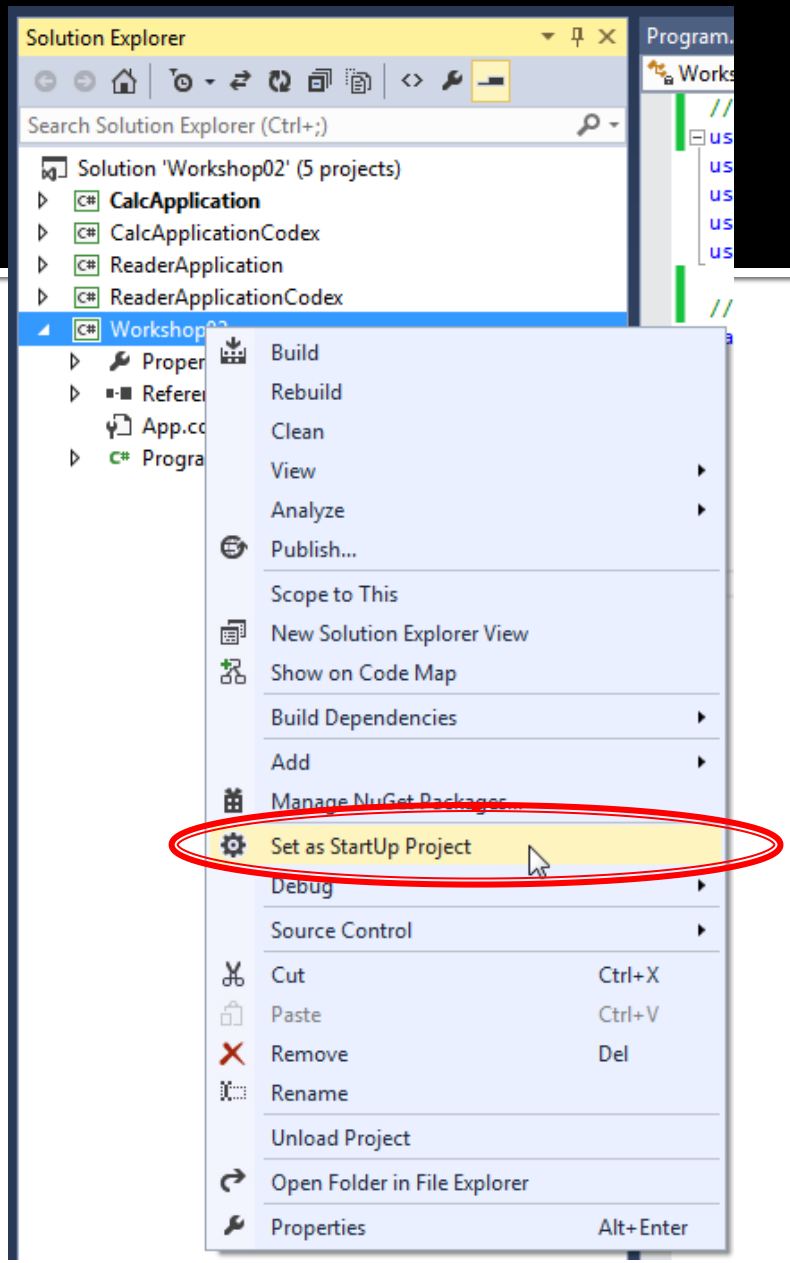
# C#
# Visual Studio 2015

## 2. Adding new Project into an existing Solution



# C#
# Visual Studio 2015

3. Changing “main” project



# C#
# Visual Studio 2015

## 4. Hot-keys overview

20. 2. 2014

**Hotkeys MSVC# 2010 Expres / 2010 Ultimate / 2012 Ultimate**

- Ctrl+Shift+N: nový projekt
- F5, Ctrl+F5: spustit (debug, bez-ladění)
- Shift+F5: ukončit
- F9: breakpoint zapnout/vypnout
- F10, F11: krokování

- F12: volání funkce -> definice
- F7, Shift+F7: zdrojový kód, designer / F7
- Ctrl+K,R: nalézt všechny výskyty / Shift+F12
  - Ctrl+W, S: Solution Explorer / Ctrl+Alt+R
  - Ctrl+W, C: Class View / Ctrl+Shift+C
  - Ctrl+W, X: Toolbox / Ctrl+Alt+X
  - Ctrl+W, T: ToDo-list / Ctrl+\, T
- Ctrl+D:
  - Ctrl+D, C: Call Stack / Ctrl+Alt+C / Alt+F7
  - Ctrl+D, W: Watches / Ctrl+Alt+W, 1..4

……..

# C#
## Basics

Code wrapped within classes, which are wrapped within namespaces.

Specific method as application "entry point", which is invoked as the first by .NET

```
// Which namespace are available
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

// Which namespace we're creating classes/interfaces/structs for
namespace Workshop02
{
    // Class definition
    0 references
    class Program
    {
        // Application entry point
        0 references
        static void Main(String[] args)
        {
            // Pascal: writeln(...);
            System.Console.WriteLine("You!");
            // Pascal: sort-of readln(...);
            System.Console.ReadLine();
        }
    }
}
```

# C#
## Declaring variables

"Standard" types and their nullable counterparts.

Why is there no "string?" … ?

string is already a class!

```
int     wholeNumber = 20;
float   floatNumber = 0.5f;
char    character   = 'A';
string  stringVar   = "Ahoj";
bool    trueFalse   = false;

int?    wholeNumberN = null;
float?  floatNumberN = null;
char?   characterN   = null;
bool?   trueFalseN   = null;
```

# C#
# Reading/Writing from/to console

Sort of writeln() but different readln()…

```
// Pascal: writeln(...);
System.Console.WriteLine("You!");
// Pascal: sort-of readln(...);
string line = System.Console.ReadLine();

// Converting String to int/float, accessing respective chars
int    wholeNumber = int.Parse(line);
float  floatNumber = float.Parse(line);
char   character   = line.Length > 0 ? line[0] : '?';
```

# C#
## Functions

```
<return type> <function name>(<parameters>) {
   <function body>
   return <value of return type>;
}
```

```
0 references
int readNumber_1()
{
    string line = System.Console.ReadLine();
    int number  = int.Parse(line);
    return number;
}

0 references
int readNumber_2()
{
    return int.Parse(System.Console.ReadLine());
}
```

# C#

## Task 2.1.1

- Create a simple number reader
- Use functions only

This simple reader should be able to read numbers in binary, decimal and hexadecimal numeral systems. Hexadecimal system may mix lower and upper case symbols for respective digits (e.g. 'aAbB' is a valid number).

Input:
('bin' | 'dec' | 'hex')
<number in given system>

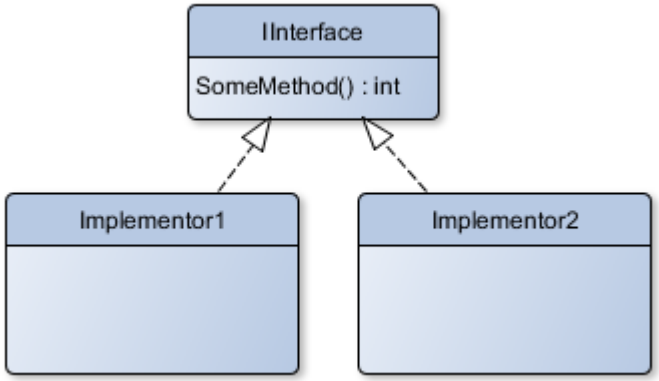
Output:
<number in decimal system>

Examples:

| No. | Input | Output |
|---|---|---|
| 1 | bin<br>111 | 7 |
| 2 | dec<br>111 | 111 |
| 3 | hex<br>111 | 273 |
| 4 | hex<br>Ff | 255 |

# C#
## Interfaces, Classes

Hiding concrete implementation behind common interface.




UML

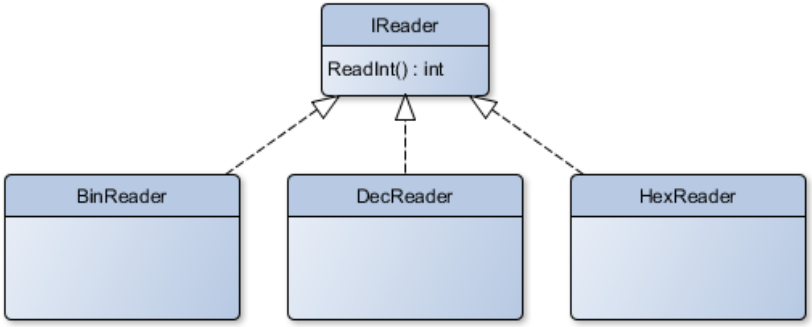
```
2 references
interface IInterface
{
    0 references
    int SomeMethod();
}

0 references
class Implementor1 : IInterface
{
    0 references
    int SomeMethod()
    {
        return 0;
    }
}
0 references
class Implementor2 : IInterface
{
    0 references
    int SomeMethod()
    {
        return 1;
    }
}
```

```
static void Main(String[] args)
{
    IInterface iface;

    iface = new Implementor1();
    System.Console.WriteLine(
        iface.SomeMethod()
    );

    iface = new Implementor2();
    System.Console.WriteLine(
        iface.SomeMethod()
    );
}
```

# C#

## Task 2.1.2

- Create common `IReader` interface
- Provide different implementations




This simple reader should be able to read numbers in binary, decimal and hexadecimal numeral systems. Hexadecimal system may mix lower and upper case symbols for respective digits (e.g. 'aAbB' is a valid number).

Input:
('bin' | 'dec' | 'hex')
<number in given system>

Output:
<number in decimal system>

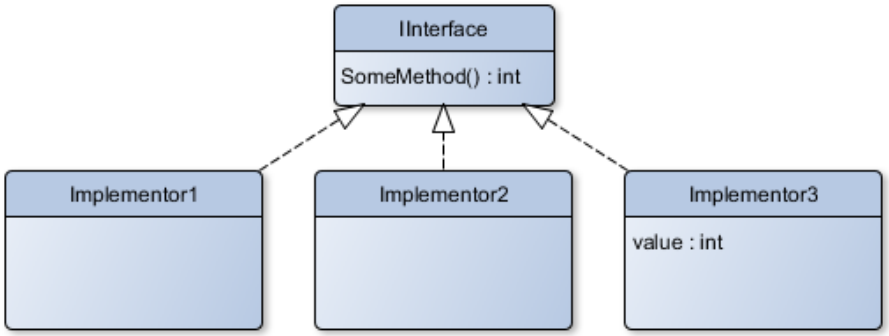
Examples:

| No. | Input | Output |
|---|---|---|
| 1 | bin<br>111 | 7 |
| 2 | dec<br>111 | 111 |
| 3 | hex<br>111 | 273 |
| 4 | hex<br>Ff | 255 |

# C#
## Constructor

Enforcing correct initialization of the object.

```
class Implementor3 : IInterface
{
    private int value;

    2 references
    public Implementor3(int value)
    {
        this.value = value;
    }

    0 references
    int SomeMethod()
    {
        return value;
    }
}
```

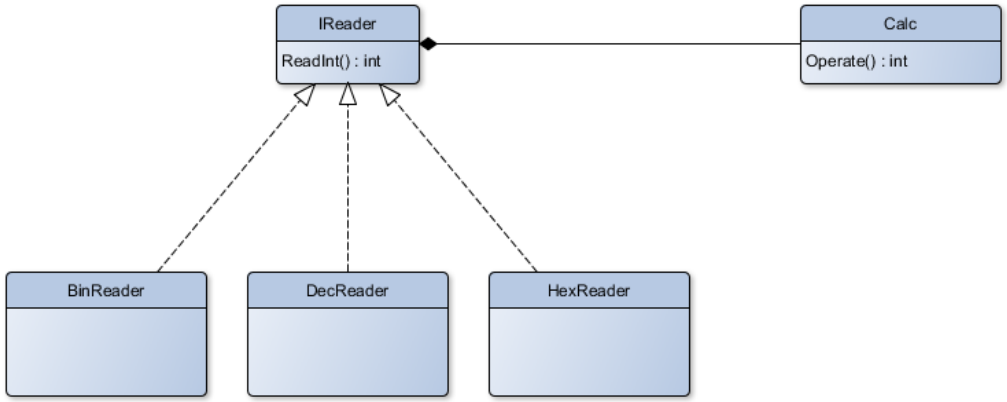



UML

```
static void Main(String[] args)
{
    IInterface iface;

    iface = new Implementor3(10);
    System.Console.WriteLine(
        iface.SomeMethod()
    );

    iface = new Implementor3(20);
    System.Console.WriteLine(
        iface.SomeMethod()
    );
}
```

# C#
## Task 2.2

- Create `Calc` class that will internally use some `IReader`




UML - Composition

This Calc is simple extension to the previous Reader. You should create another class, that should be initialized with a correct reader (according to the input) and implement +,-,*,div,mod operations. Again you should be able to read numbers in binary, decimal and hexadecimal numeral systems. Hexadecimal system may mix lower and upper case symbols for respective digits (e.g. 'aAbB' is a valid number).

Input:
('bin' | 'dec' | 'hex')
<number in given system>
('+' | '-' | '*' | 'div' | 'mod')
<number in given system>

Output:
<first number in decimal system> <given operator> <second number in decimal system> = <result>

**WARNING:** The output **MUST HAVE** single whitespace around operator and aroun '='. E.g.: "10 + 10 = 20" is correct, "10+10=20" is incorrect.

Examples:

| No. | Input | Output |
|---|---|---|
| 1 | bin<br>111<br>-<br>101 | 7 - 5 = 2 |
| 2 | dec<br>111<br>+<br>1 | 111 + 1 = 112 |
| 3 | hex<br>111<br>*<br>11 | 273 * 17 = 4641 |
| 4 | hex<br>eE<br>-<br>Ff | eE - fF = -17 |

# Assignment 2
## Send me an email if you haven't already

- If you have not sent me an email during Assignment 1, DO IT NOW!
- Email: gemrot@gamedev.cuni.cz
- Subject: **Programming II – 2015 – Assignment 02**
- Content:
  - **Your name**
  - **Your CUNI number**
  - **Your CoDex nick**
  - **I will add you into a CodEx group**

# Assignment 2
## Code tasks 2.1, 2.2

- Create implementations for **tasks 2.1, 2.2** (if you haven't already) and submit them to **CodEx**; each task worths 5 points
- Deadline: **9.3.2015 23:59**

# Questions?

## I sense a soul in search of answers…

- Sadly, I do not own the patent for perfection (and will never do)
- In case of doubts about the assignment or some other problems don't hesitate to contact me!
  - Jakub Gemrot
    - gemrot@gamedev.cuni.cz