Advanced Programming C#

Lecture 1

dr inż. Małgorzata Janik
malgorzata.janik@pw.edu.pl

Organizational issues

- **Lecture + laboratories + project:**
  - dr inż. Małgorzata Janik
    Zakład Fizyki Jądrowej
    pok. 117D, Gmach Fizyki

    malgorzata.janik@pw.edu.pl
- **Time:**
  - Wednesday, 12:15-13:45
- **Webpage:**
  - www.if.pw.edu.pl/~majanik/wiki
- **Office hours,** 117D GF**:**
  - Please write to me on the chat of MS Teams to schedule a meeting

Organizational issues

- **Final grades:**
  - Laboratories: 60% of the grade
  - Project: 40% of the grade
- **Laboratories:**
  - 14 classes: 1 instructional, 10 graded, 3 project-related
  - used software: Visual Studio Community
  - classes duration: 90 minutes (no break)
- **Projects:**
  - Project presentation on 6th, 10th and 14th classes

Conditions to pass the classes (1)

- **Laboratories:**
  - 10 classes of diversified level (**0-6 pkt each**)
  - you can use any printed materials, your own programs, as well as resources available on the Internet*
  - The task will be explained during the class, afterwards you have **1 week** to finish the program:
    - finished program must be presented in the next class
    - during the class (1.5 h) you can ask me any questions related to the current task & present the program from the previous class

*) it is forbidden to use mailboxs, messangers, social networks or programs written by other students, as well as phones, tablets etc. to communicate with others.

- **Absences:**
  - max 2 absences (not presenting the program) are allowed (**0 pkt**)
  - unless absence is justified, number of points will be reduced if the deadline is missed

Conditions to pass the classes (2)

- **Project:**
  - grading: **0-40 pkt** for the project
  - During the semester there will be **2 intermediate stages,** when the current status of the project should be presented
  - Each intermediate stage: **0-10 pkt**
  - Final project (should be shown in the last class): **0-20 pkt**
  - To pass the subject **>50% of the points from the project** should be acquired (minimal project requirements should be completed)

Project proposals

- Simulation of several simple physics experiments
- Simulation of the interaction of the radiation with matter
- Main building path finder: application showing the shortest path between two rooms in the Warsaw University of Technology Main Building
- Network Messenger
- Simple RPG game
- Simple platform game

Project proposals

- Simu
- Simu matter
- Main shortest path betwe echnology
  Main
- Netw
- Simp
- Simp

Shooter
Movement
Shooting
Death
Opponents AI

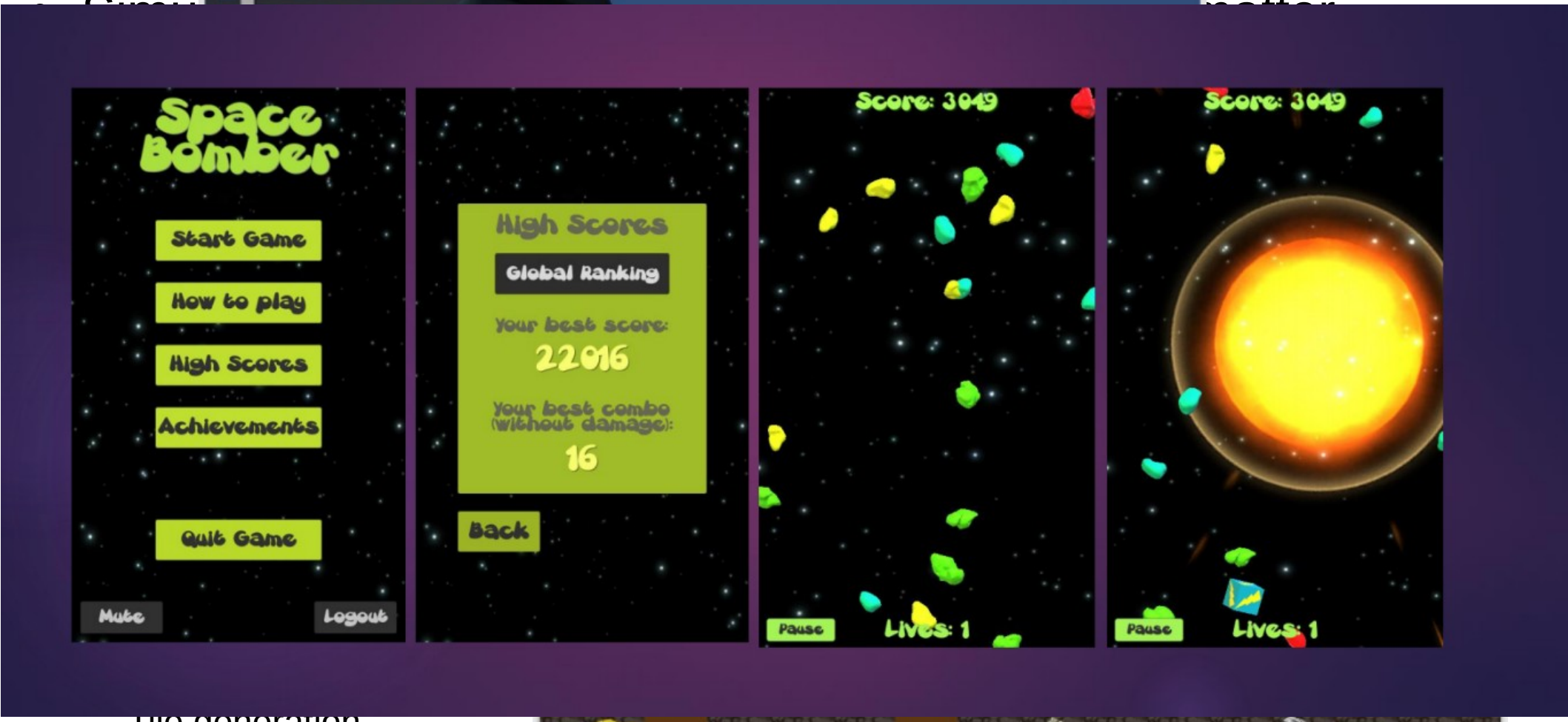
Project proposals

- Simu
- Simu matter
- Main shortest path betwe
  Main
- Netw
- Simp
- Simp

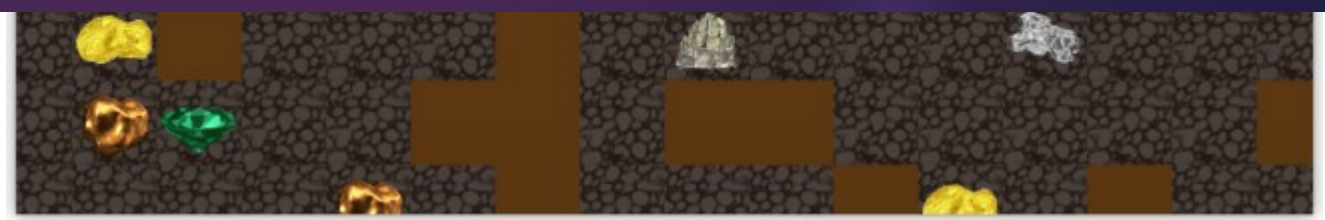
MarsMiner
Tile generation
Selling Minerals
Town Buildings
Buttons, Mouse Support

Project proposals

- Simu

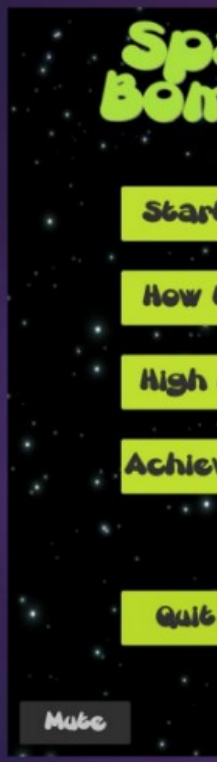


Tile generation
Selling Minerals
Town Buildings
Buttons, Mouse Support



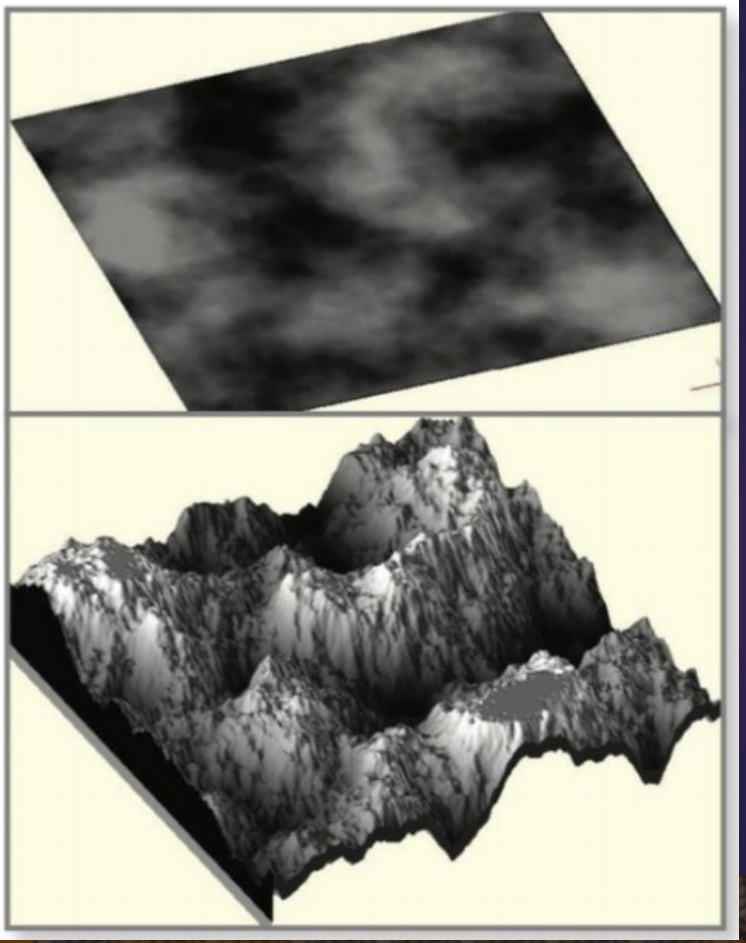
Project proposals

- Simu



Topic

Convert BMP file to STL file



Tile ge
Selling Minerals
Town Buildings
Buttons, Mouse Support

Conditions to pass the classes (3)

- **Grading:**
  - Maximal number of points: **100**
    - laboratories: **10*6 = 60**
    - project: **2*10+20 = 40**
- **To pass the subject (% of the total number of points):**
  - >50% - 3 (50,5 pkt. – 60,0 pkt.)
  - >60% - 3,5 (60,5 pkt. – 70,0 pkt.)
  - >70% - 4 (70,0 pkt. – 80,0 pkt.)
  - >80% - 4,5 (80,5 pkt. – 90,0 pkt.)
  - >90% - 5 (90,5 pkt. – 100,0 pkt.)
- **Warning! To pass the subject you have to deliver the project (>50% points)**

Literature

- English:

  1. Joseph Albahari, Ben Albahari, C# 6.0 in a Nutshell, 2016.
  2. Ian Griffiths, Programming C# 5.0, O'Reilly Media, 2012.

- Polish:

  1. Joseph Albahari, Ben Albahari, C# 6.0 w pigułce, Helion 2016
  2. Ian Griffiths - "C# 5.0. Programowanie", Helion, 2013.
  3. Andrew Troelsen - "Język C# 2010 i platforma .NET 4", PWN, 2011.
  4. Jon Skeet - "C# od podszewki", Helion, 2012.
  5. Jesse Liberty - “Programowanie C#”, Helion 2012

Programme

1. Introduction to the C# programming language and Visual Studio software.
2. Principles of C# programming language, basic information on the .NET platform. Windows Forms.
3. Classes, inheritance, virtual methods.
4. Interfaces, instruction foreach, yield iterators.
5. Standard library classes (collections, streams and files).
6. Delegations, lambda expressions.
7. Events, exceptions.
8. LINQ technology.

Programme

1. Introduction to the C# programming language and Visual Studio software. Principles of C# programming language.
2. Windows Forms.
3. Windows Presentation Foundation (WPF).
4. Web Forms: ASP.NET.
5. Databases: AOD.NET. → [Online?]
6. PROJECT I
7. Classes, inheritance, virtual methods.
8. Delegations, lambda expressions.
9 . Events, exceptions.
10. LINQ technology.
11. PROJECT II
12. LINQ to SQL
13. Multithreading.
9. Writing code with Chat GPT
14. PROJECT II

Introduction to the C# language and Visual Studio software

C#

- C# (pronounced "C sharp") is a programming language that is designed for building a variety of applications that run on the .NET Framework.

.NET Framework

.NET Framework (pronounced **dot net**) is a software framework developed by Microsoft.

.NET Framework includes 2 parts:

- a **large class library** known as **Framework Class Library** (FCL) and provides language interoperability (each language can use code written in other languages) **across several programming languages (C#, C++, F#, Visual Basic, and a few dozen others).**

- programs written for .NET Framework execute in a software environment known as **Common Language Runtime** (CLR), an **application virtual machine** that provides services such as security, memory management, and exception handling.

Why C#?

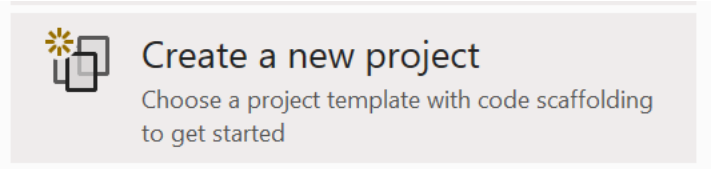
- Simple and easy to learn
- Curly-brace syntax of C# will be instantly recognizable to anyone familiar with C, C++ or Java → easy for people previously programming in any of those languages
- C# syntax simplifies many of the complexities of C++ and provides powerful features such as nullable value types, enumerations, delegates, lambda expressions and direct memory access, which are not found in Java.
- C# supports generic methods and types, which provide increased type safety and performance, and iterators, which enable implementers of collection classes to define custom iteration behaviors that are simple to use by client code.
- Language-Integrated Query (LINQ) expressions make the strongly-typed query a first-class language construct.

Hands on!

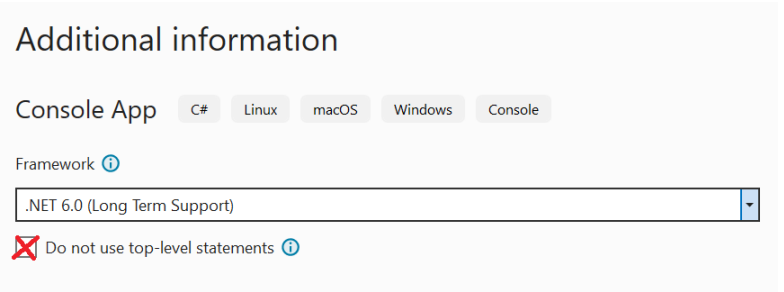
First console application

- Open Visual Studio
- File → New → Project
- Console Application

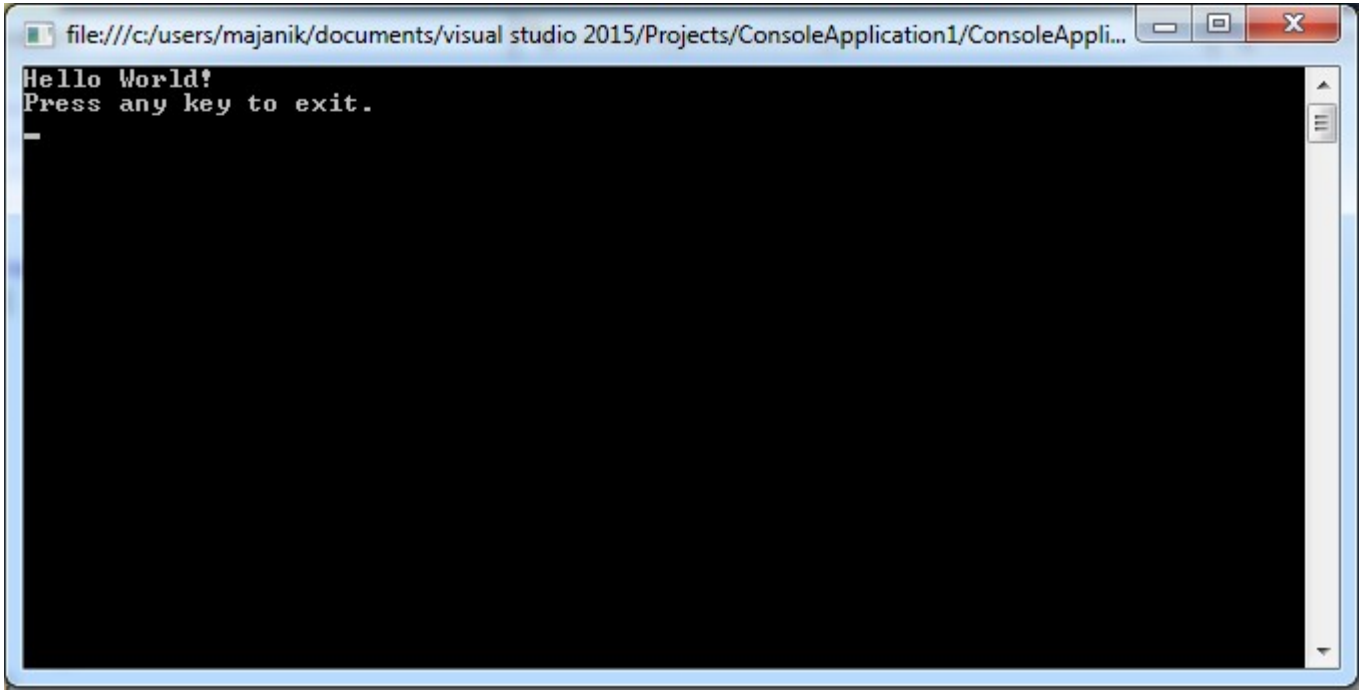
  (without ".NET Framework")



Console Application - Printing




```
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            System.Console.WriteLine("Hello World!");

            // Keep the console window open in debug mode.
            System.Console.WriteLine("Press any key to exit.");
            System.Console.ReadKey();
        }
    }
}
```



Console Application - Variables

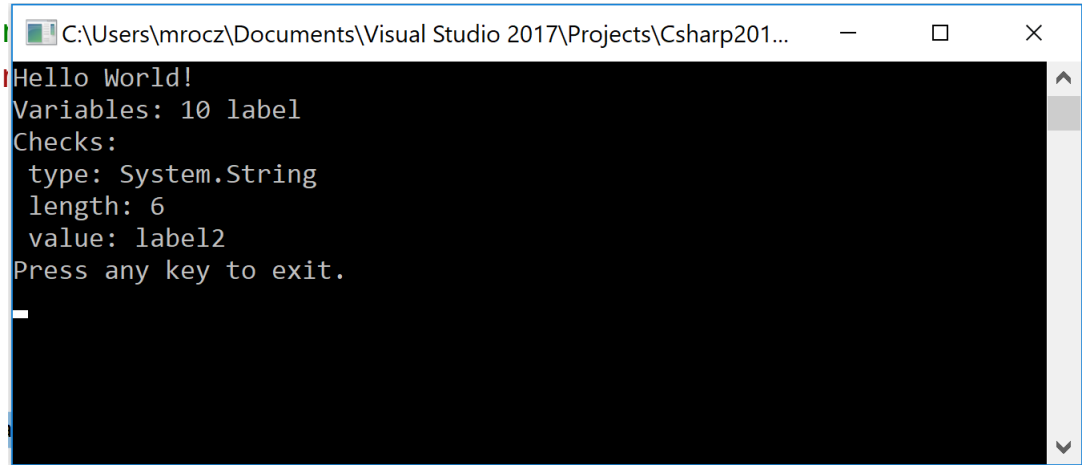
```
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            System.Console.WriteLine("Hello World!");

            int a = 10;
            string b = "label";
            System.Console.WriteLine("Variables: {0} {1}", a, b);
            System.Console.WriteLine($"Variables: {a} {b}");


            var c = "label2";
            // var d; // NOT POSSIBLE
            // Keep the console window open in debug mode.
            System.Console.WriteLine("Press any key to exit.");
            System.Console.ReadKey();
        }
    }
}
```

Console Application - Task

```
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            System.Console.WriteLine("Hello World!");

           int a = 10;
            string b = "label";
            System.Console.WriteLine("Variables: {0} {1}", a, b);
            System.Console.WriteLine($"Variables: {a} {b}");

            var c = "label2";

            // TASK
            // Print: type of c, length of c and value of c

            // Keep the console window open in
            System.Console.WriteLine("Press an
            System.Console.ReadKey();
        }
    }
}
```

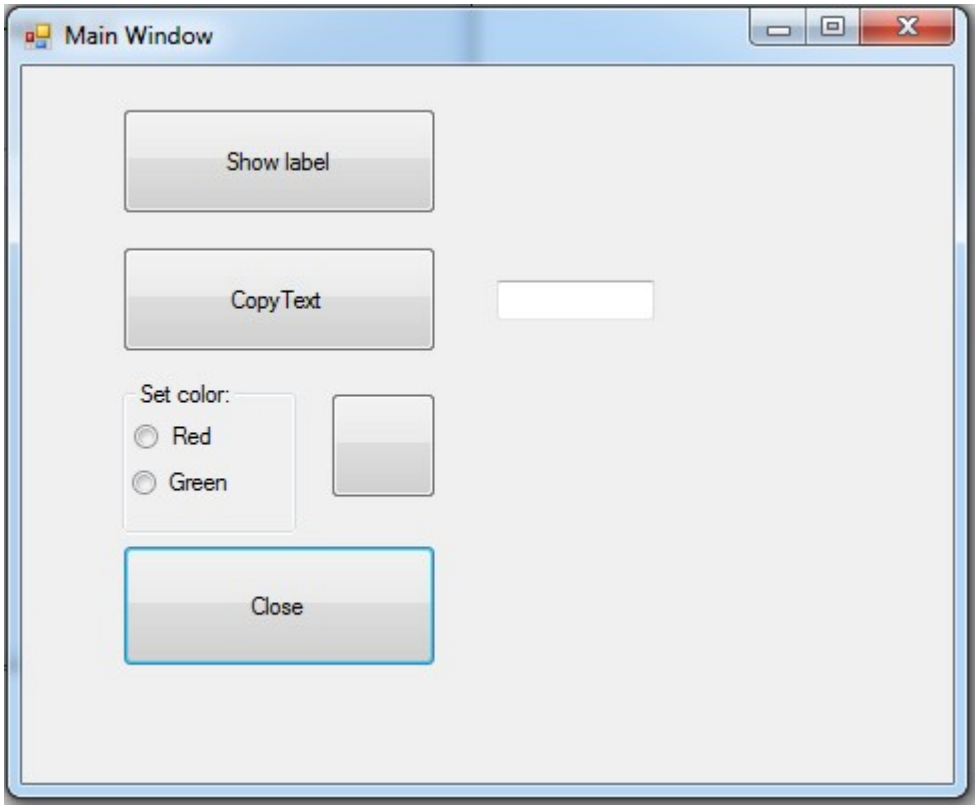
Type “c.” and wait for the list of possible methods and properties appear.

Browse through them and try to find the ones requested.



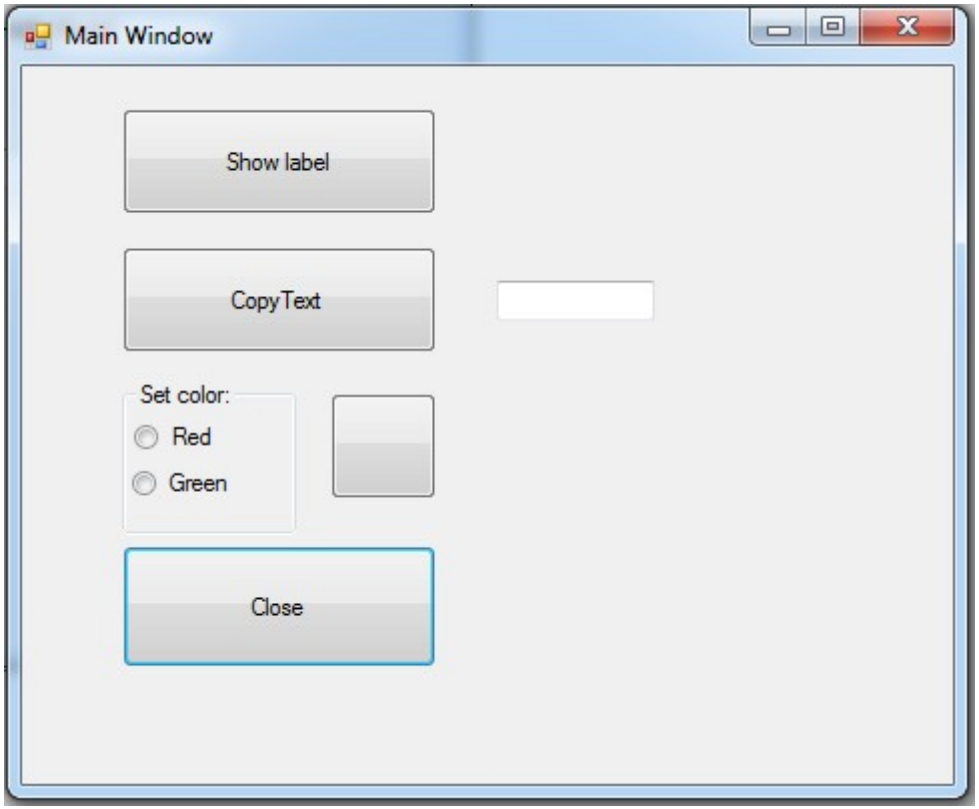
Planned application

Initial window:

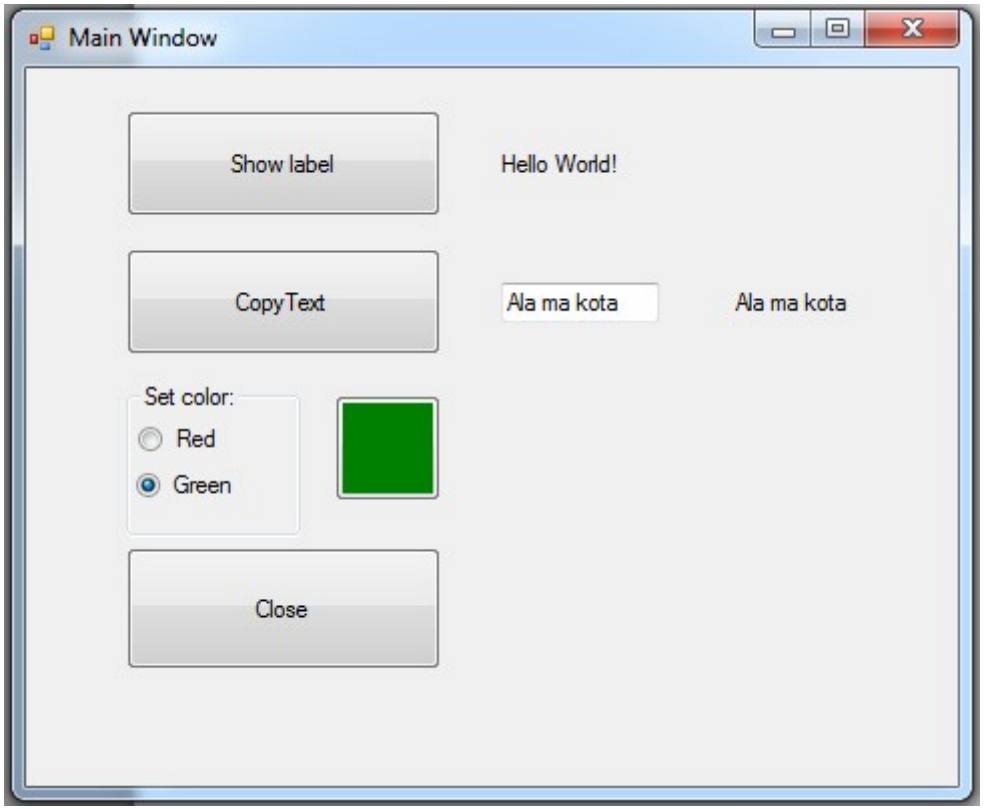


Planned application

Initial window:



Used functionalities:

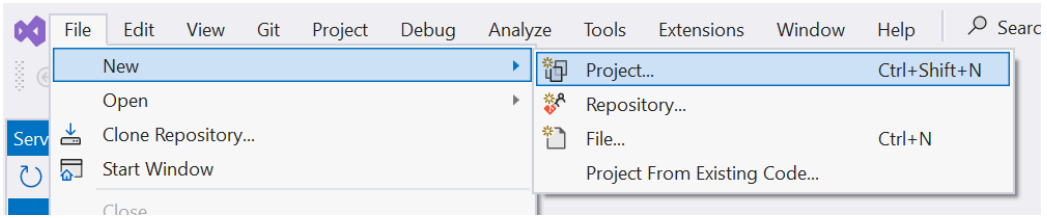


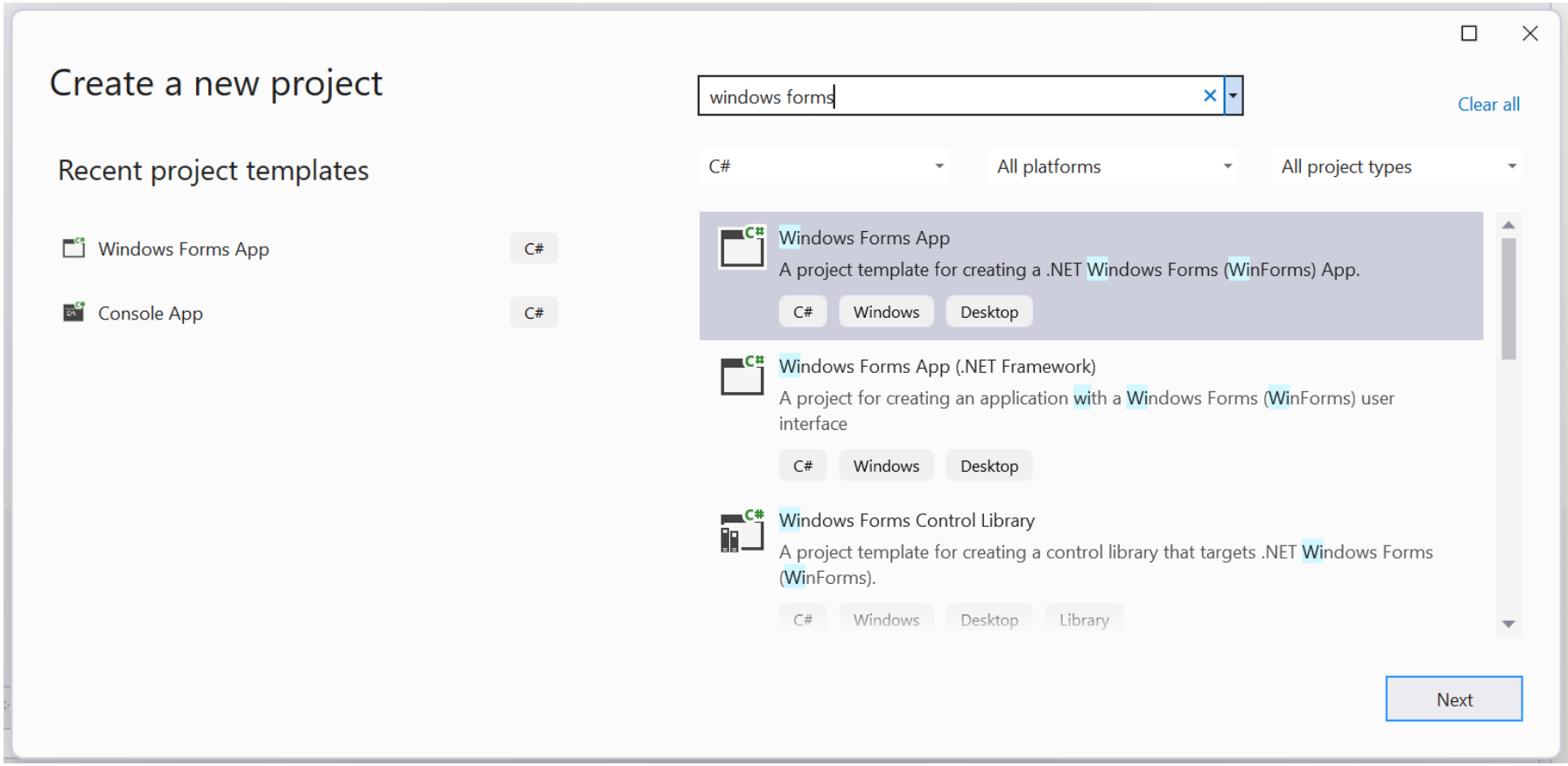
Create new project

New…

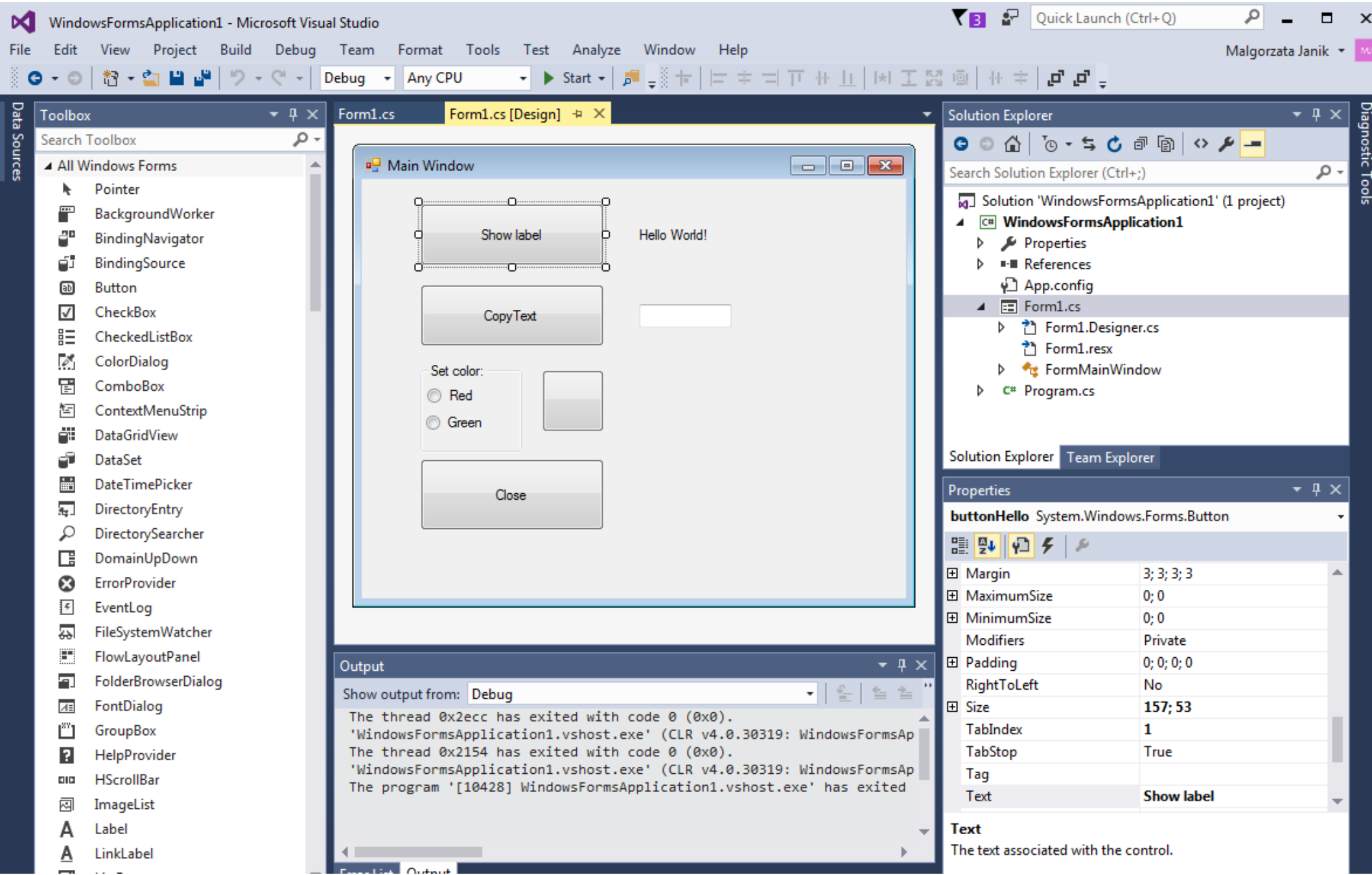
Project

Windows Forms App

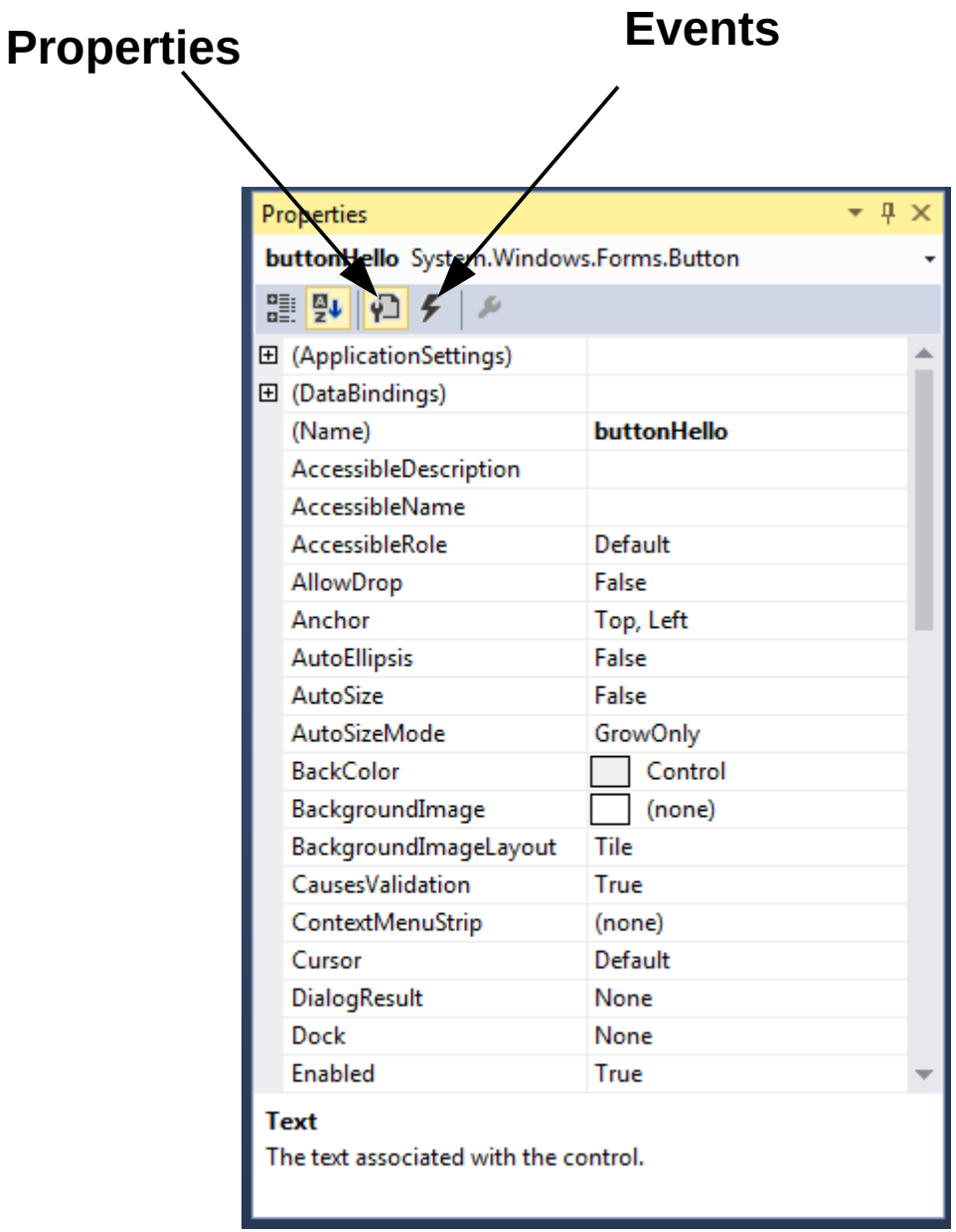




Create new project and synchronize it with repository

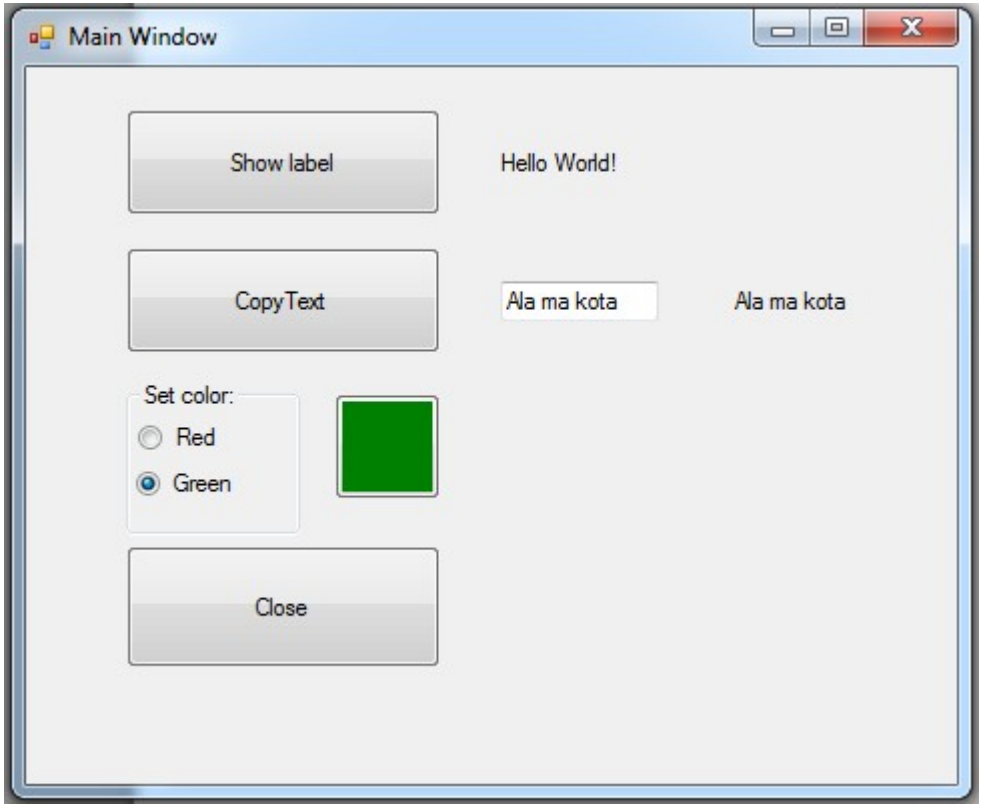


Properties and Events



Build your application

Used functionalities:



Remember to:
- give readable names to all controls
- commit changes after each part

Names:
- Always change default names!
- Each team can have its own naming convention.
- Common thing: names are readable!

This classes:
- Always keep the control name + readable part.

e.g. formMainWindow
labelHelloWorld

THE END