В.В. Подбельский, О.В. Максименкова

Программирование на C#

Семинар 1

Знакомство с IDE
Консольный Ввод-Вывод

Где Искать Материалы Курса?

Материалы семинаров находятся в SmartLMS

Материалы дублируются на Google-диск
https://drive.google.com/drive/folders/1fFRagLpDDj_jGj82riAXpUEpe3cqTv0A?usp=sharing

Работа в Компьютерном Классе

- Рекомендуется создать папку **на диске D:** и назвать её своей фамилией (используйте только латинские символы, иначе возможны проблемы при работе внутри IDE);
- В папке следует сохранять результаты работы на практическом занятии – при желании Вы можете выгрузить материал на своё хранилище данных.

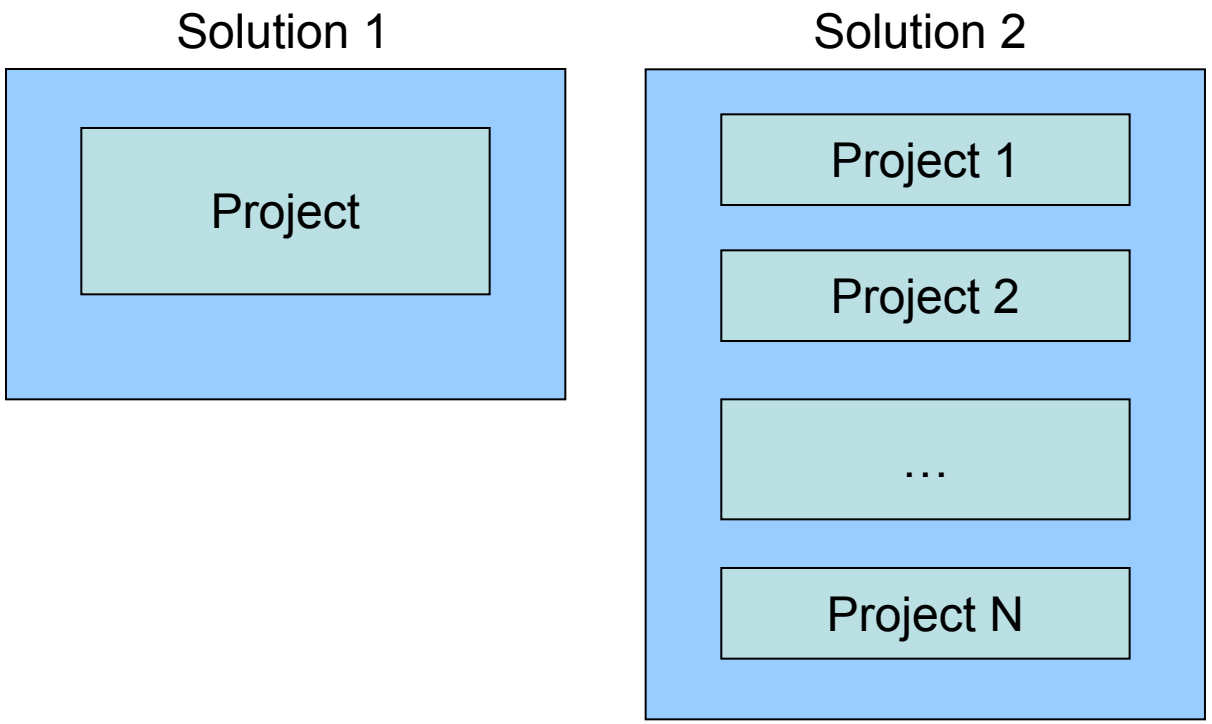
Важно: после выключения компьютера все папки с рабочего стола и диска C:\ автоматически удаляются – после этого восстановление не является возможным!

Проекты и Решения

- Для организации C# программ вводятся две основные логические единицы – решения и проекты;
- Решение является контейнером для набора проектов, каждый из которых может определять запускаемое приложение, библиотеку классов, набор тестов и т. д.
- Пример:

Решение [*Solution*]

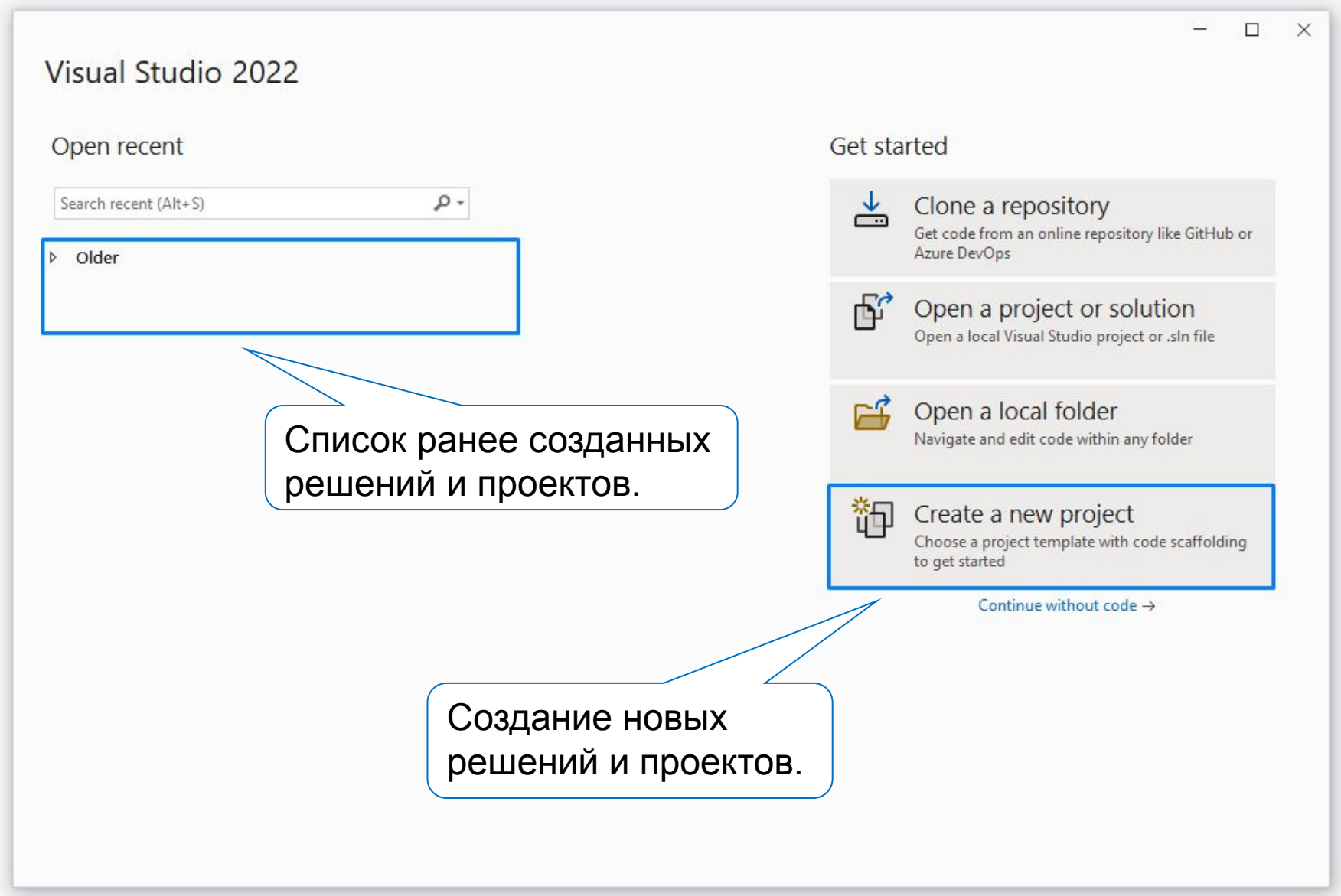
- Проект1 [*Project1*]
- Проект2 [*Project2*]
- …
- ПроектN [*ProjectN*]



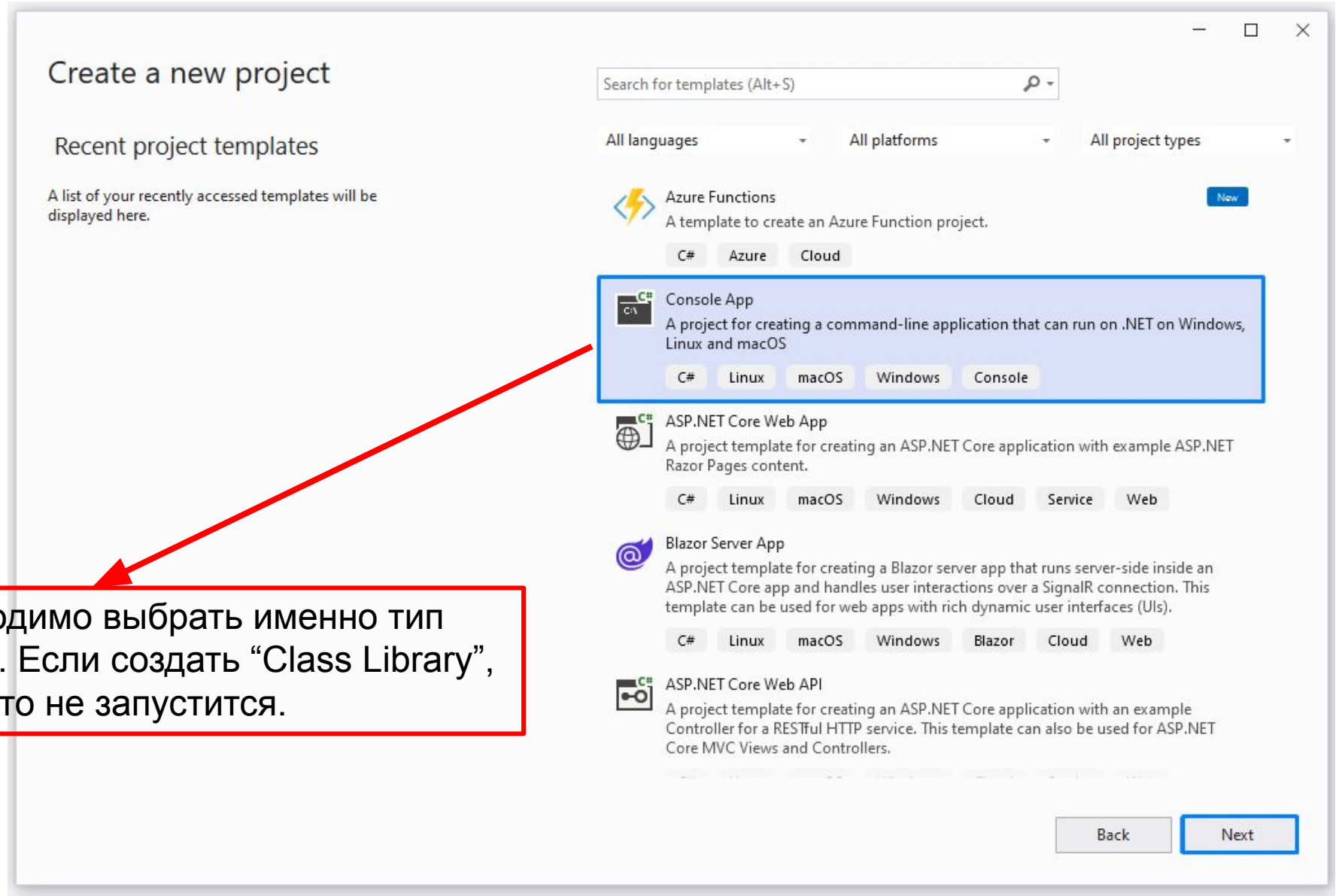
Последовательность Действий

- Запустить Microsoft Visual Studio 2022 (далее – VS 2022);
- Ознакомиться с инструкцией по созданию первого решения можно по ссылке: https://visualstudio.microsoft.com/vs/getting-started/

Стартовый Экран VS 2022

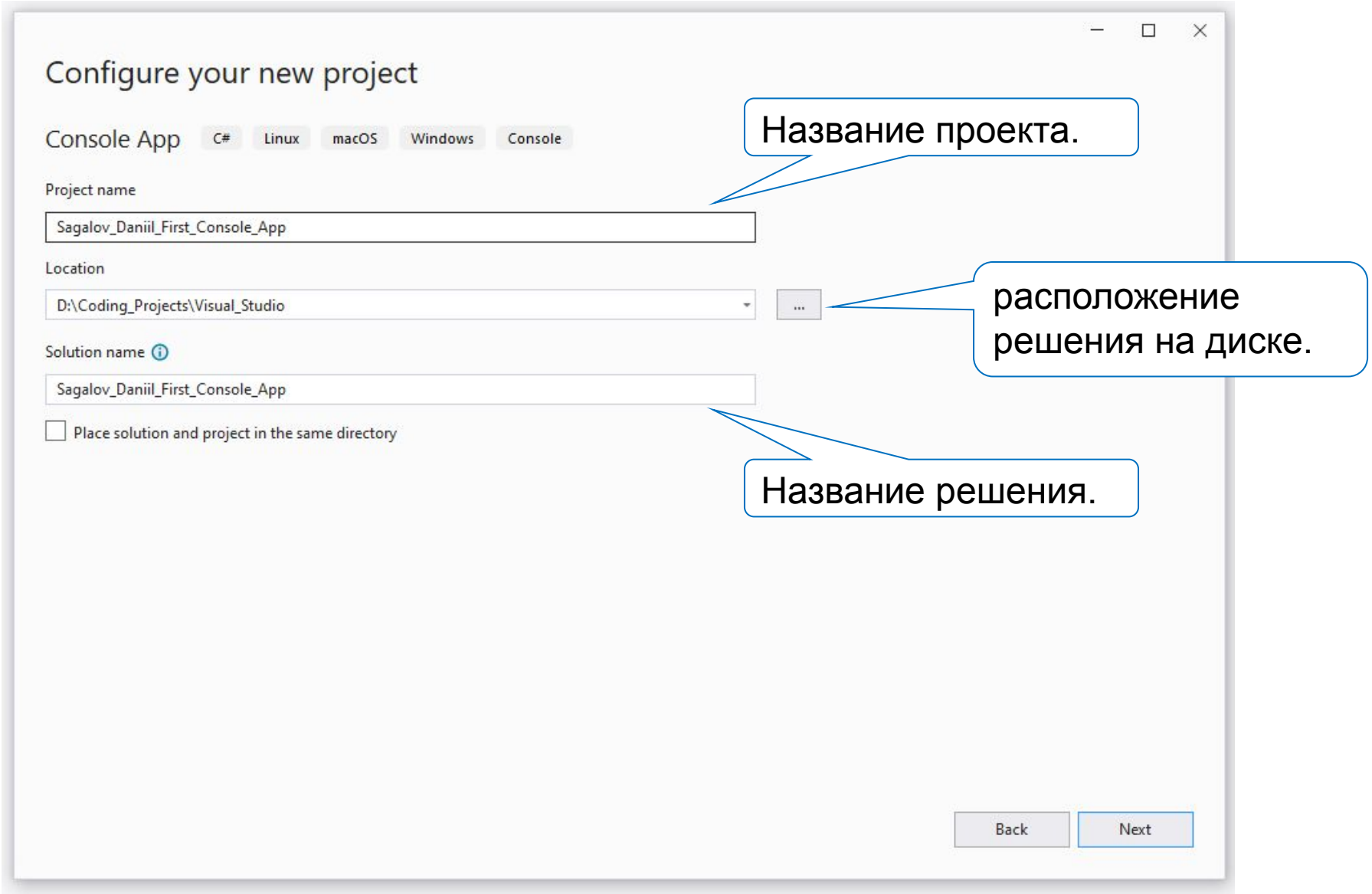


Выбор Шаблона Решения

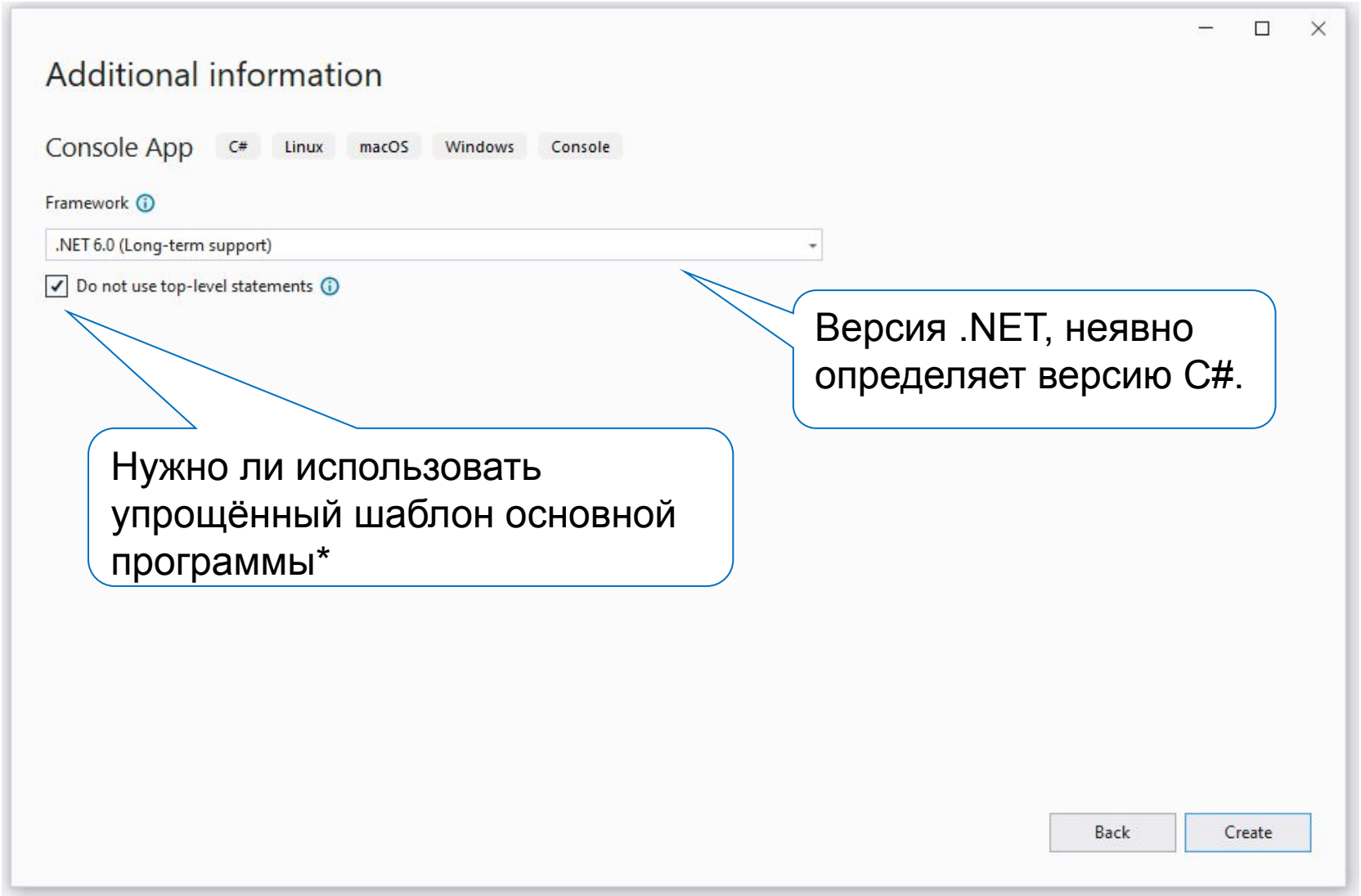


Важно: необходимо выбрать именно тип “**Console App**”. Если создать “Class Library”, то проект просто не запустится.

Название Проекта и Решения

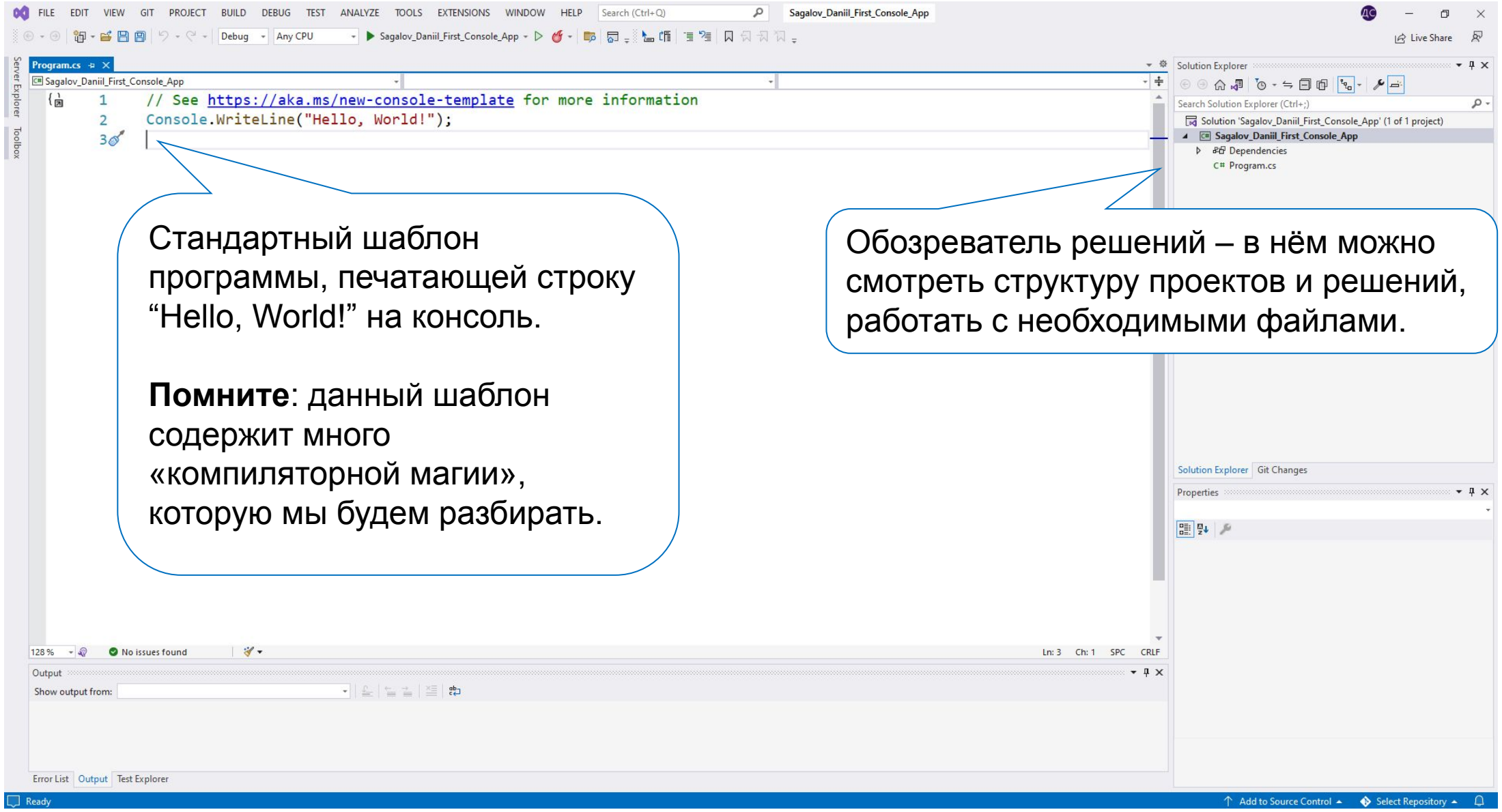


Выбор Версии и Создание Решения



*Различия между стандартным и упрощённым шаблонами будут рассматриваться далее

Первая Программа в Редакторе VS 2022



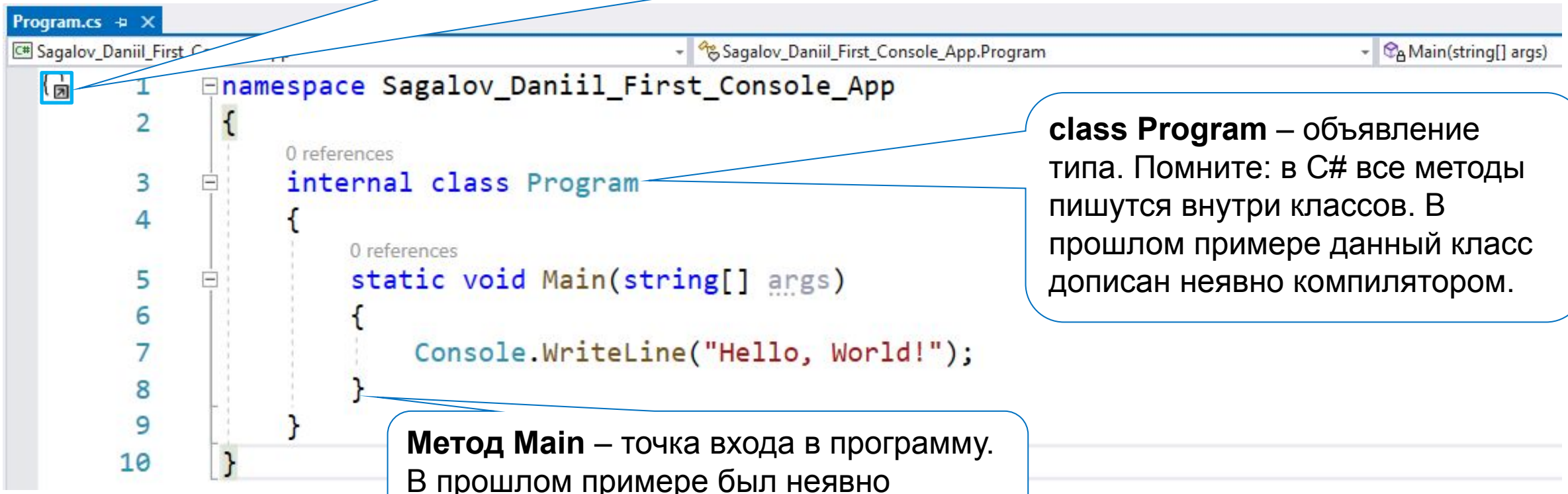
Стандартный шаблон программы, печатающей строку “Hello, World!” на консоль.

Помните: данный шаблон содержит много «компиляторной магии», которую мы будем разбирать.

Обозреватель решений – в нём можно смотреть структуру проектов и решений, работать с необходимыми файлами.

«Классическая» Первая Программа в VS 2022

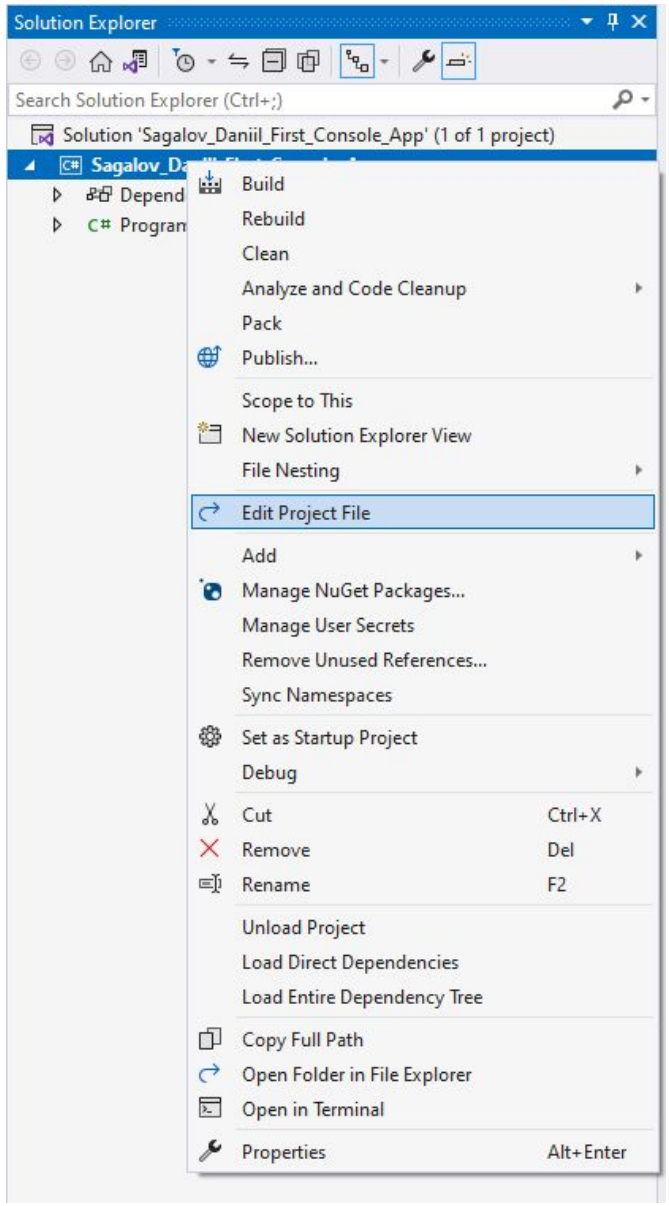
Список неявно импортированных пространств имён – позволяет использовать обращение к Console сокращённо (без явного написания System.Console каждый раз).



class Program – объявление типа. Помните: в C# все методы пишутся внутри классов. В прошлом примере данный класс дописан неявно компилятором.

Метод Main – точка входа в программу. В прошлом примере был неявно дописан компилятором за нас.

Файл с Настройками Проекта в VS 2022



В данном файле можно увидеть тип проекта (например, исполняемый exe-файл), отключить неявные using.

Sagalov_Danii...ole_App.csproj | Program.cs

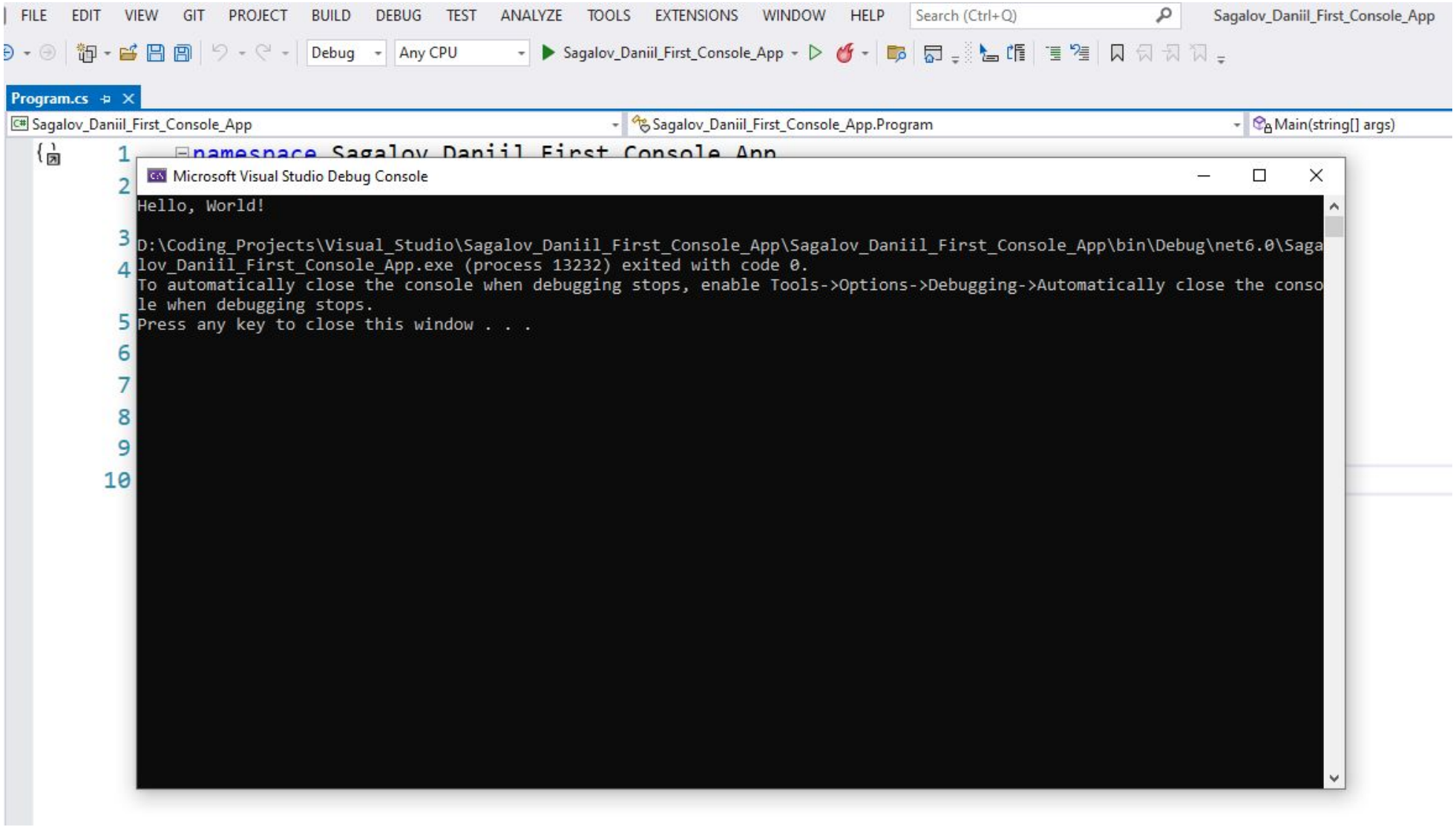
```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net6.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
  </PropertyGroup>

</Project>
```

Запуск Проекта в VS 2022

- Комбинация клавиш Ctrl + F5 (Windows);
- Вкладка с именем активного проекта → ЛКМ;
- Частично прозрачная стрелка (запуск без отладки).

Результат Запуска Программы в VS 2022

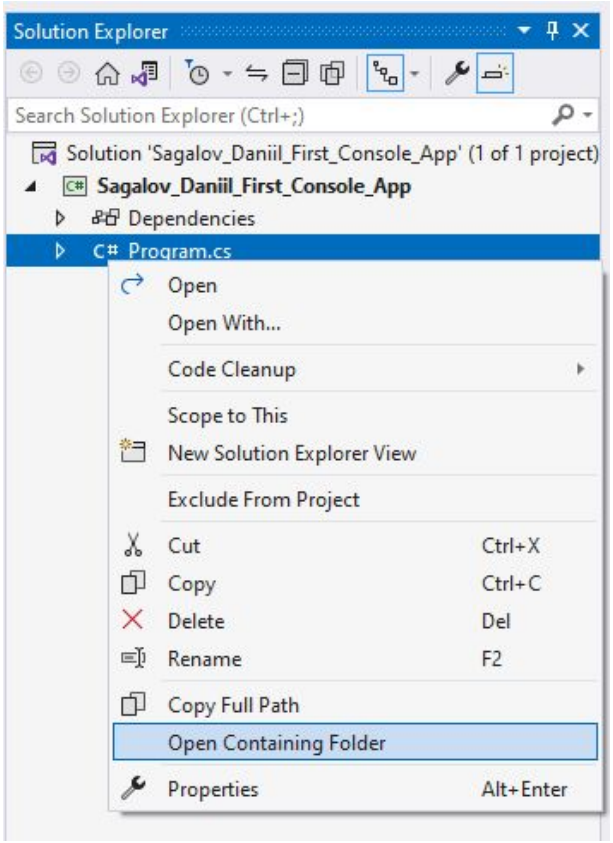
VS 2022 запускает программу в отдельном консольном окне.

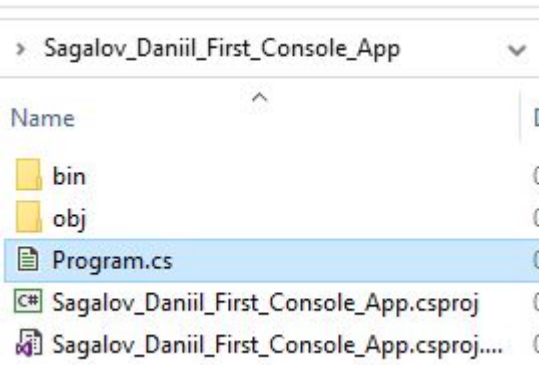


Запуск Программы вне VS 2022

Простой способ запустить программу вне VS 2022 – ПКМ по файлу с исходным кодом → с помощью “**Open containing folder**” перейти в папку, содержащую файл с исходным кодом, затем спуститься в **bin** → **Профиль сборки (по умолчанию Debug)** → **Версия .NET** → **ЛКМ по .exe файлу**.

Важно: программа мгновенно завершится, т. к. не ожидает никаких действий со стороны пользователя!





Модификация Программы в Vs 2022

```
namespace Sagalov_Daniil_First_Console_App
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
            Console.WriteLine("Press any key to exit...");
            Console.ReadKey();
        }
    }
}
```

Теперь программа ожидает нажатия любой клавиши, поэтому не закроется автоматически.

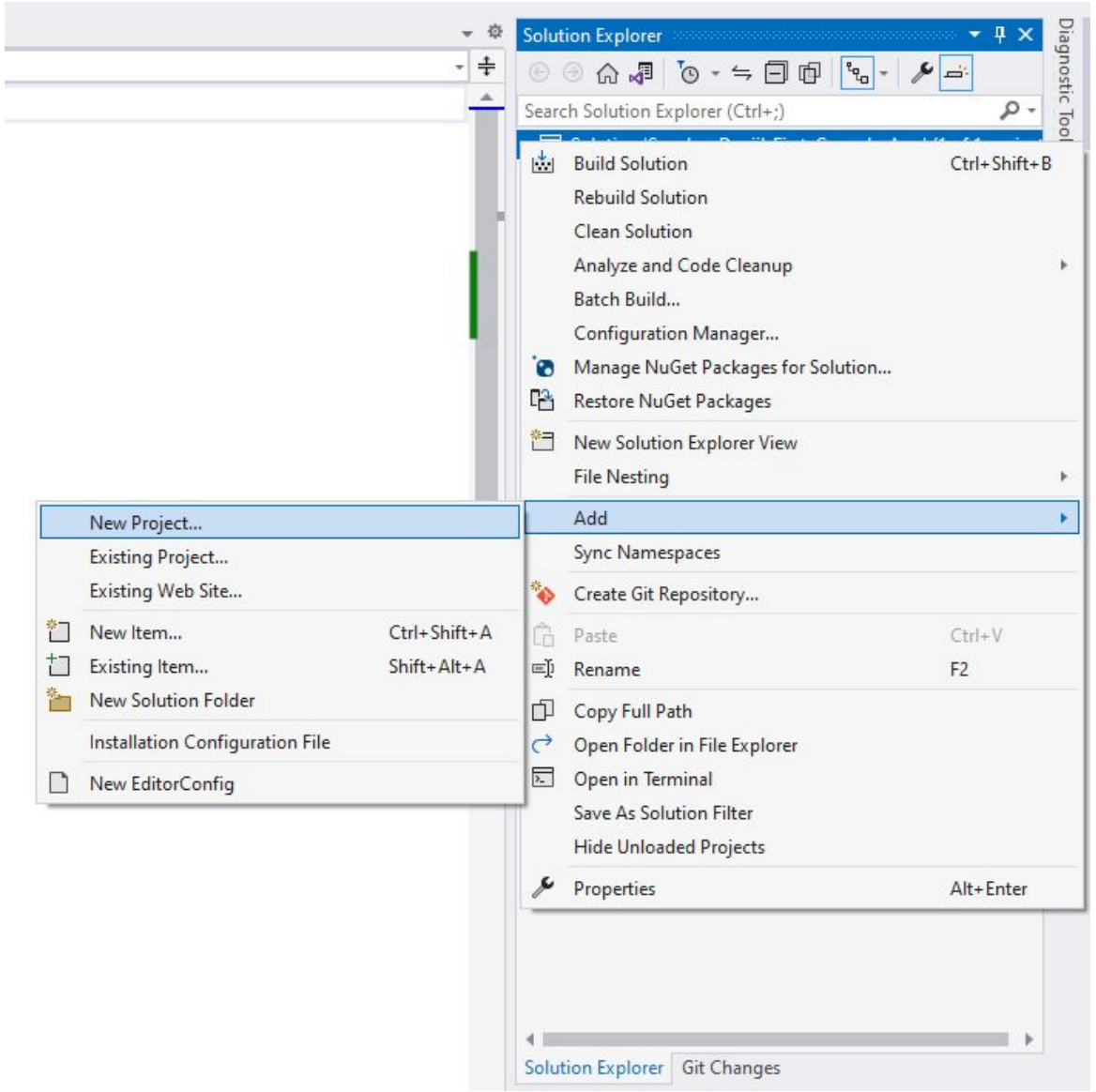
Попробуйте запустить данную программу вне VS 2022 – теперь она не закроется мгновенно.

Создание Второго Проекта в VS 2022

Удобно иметь несколько запускаемых приложений в одном решении.

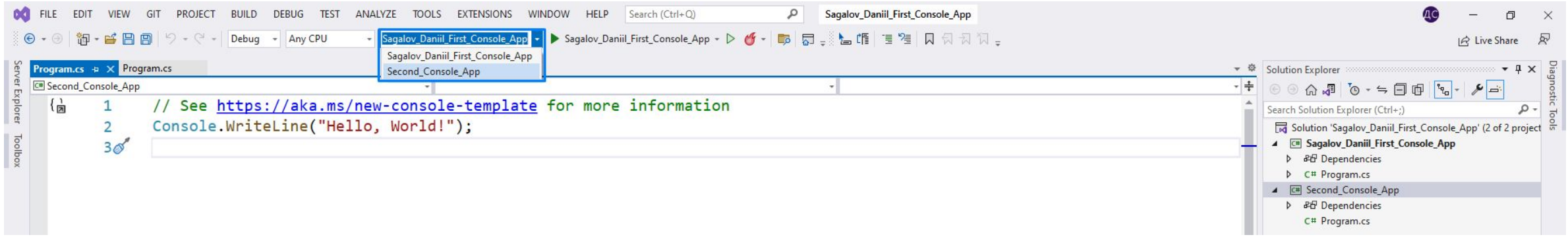
Для этого:

ПКМ по названию <u>решения</u> (не проекта!) → Add → New Project.



Выбор Запускаемого Проекта в VS 2022

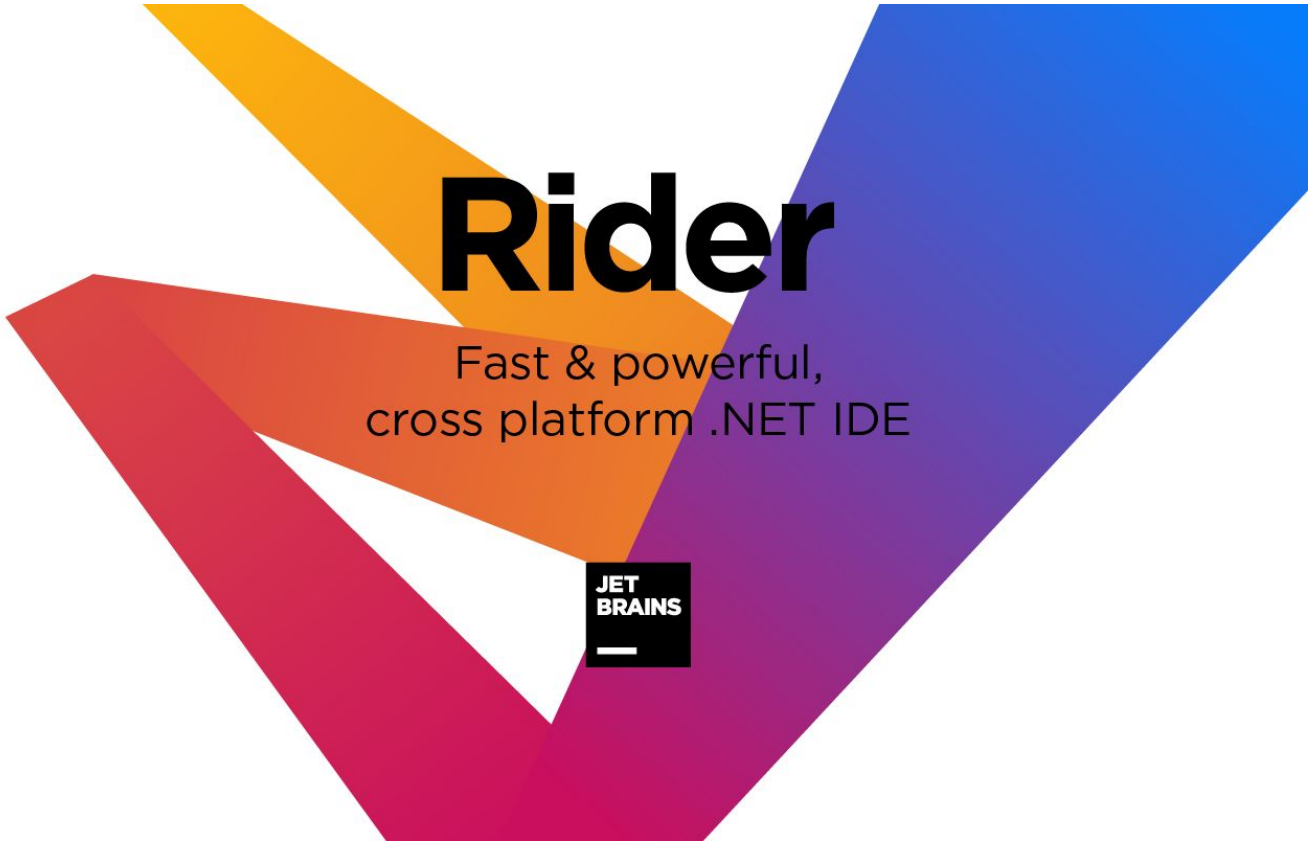
VS 2022 позволяет выбрать запускаемый проект с помощью вкладки выбора запускаемого проекта, расположенной левее кнопки запуска программы:



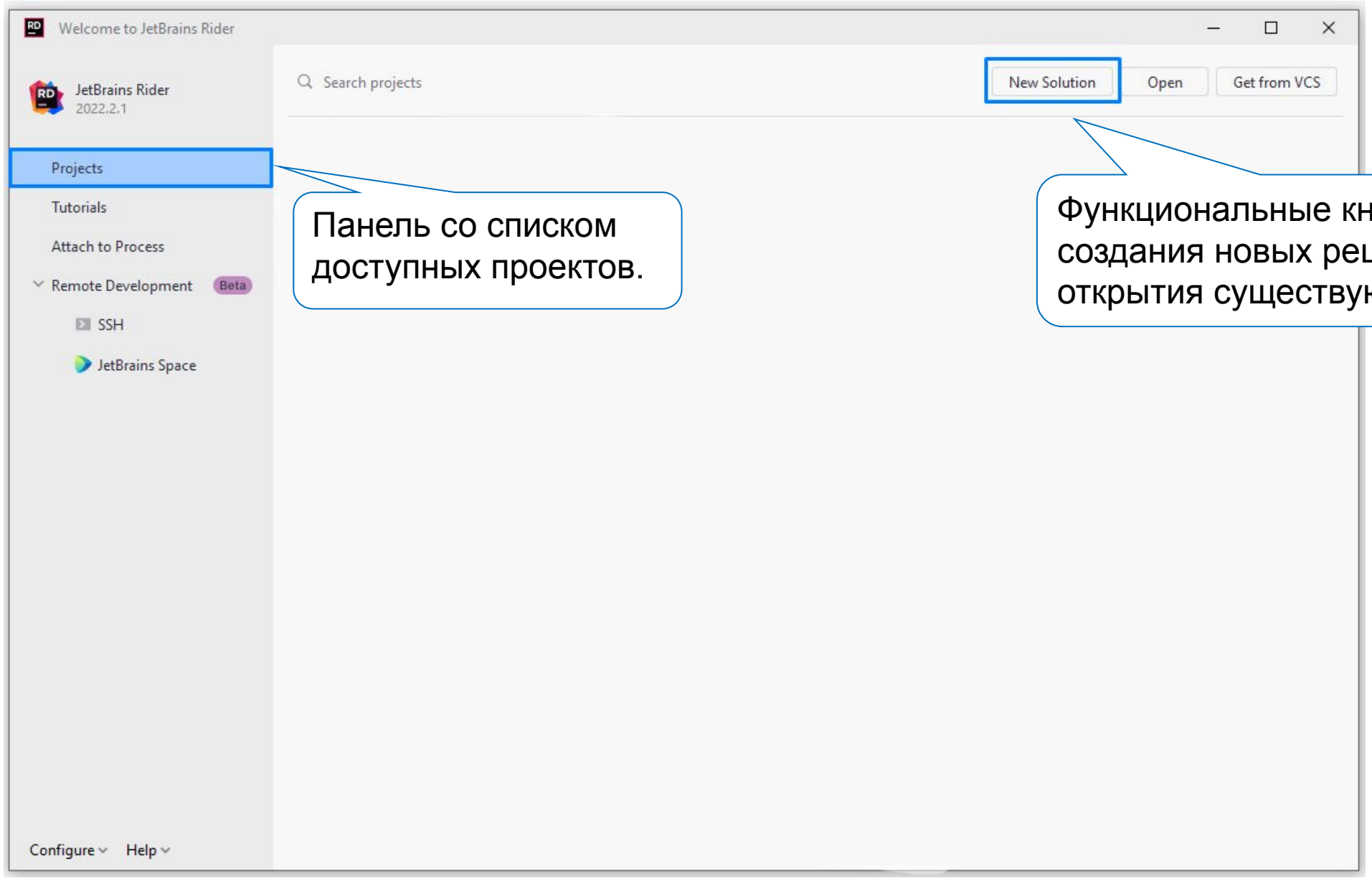
На заметку: Visual Studio выделяет текущий выбранный для запуска проект жирным шрифтом в обозревателе решений.

Последовательность Действий

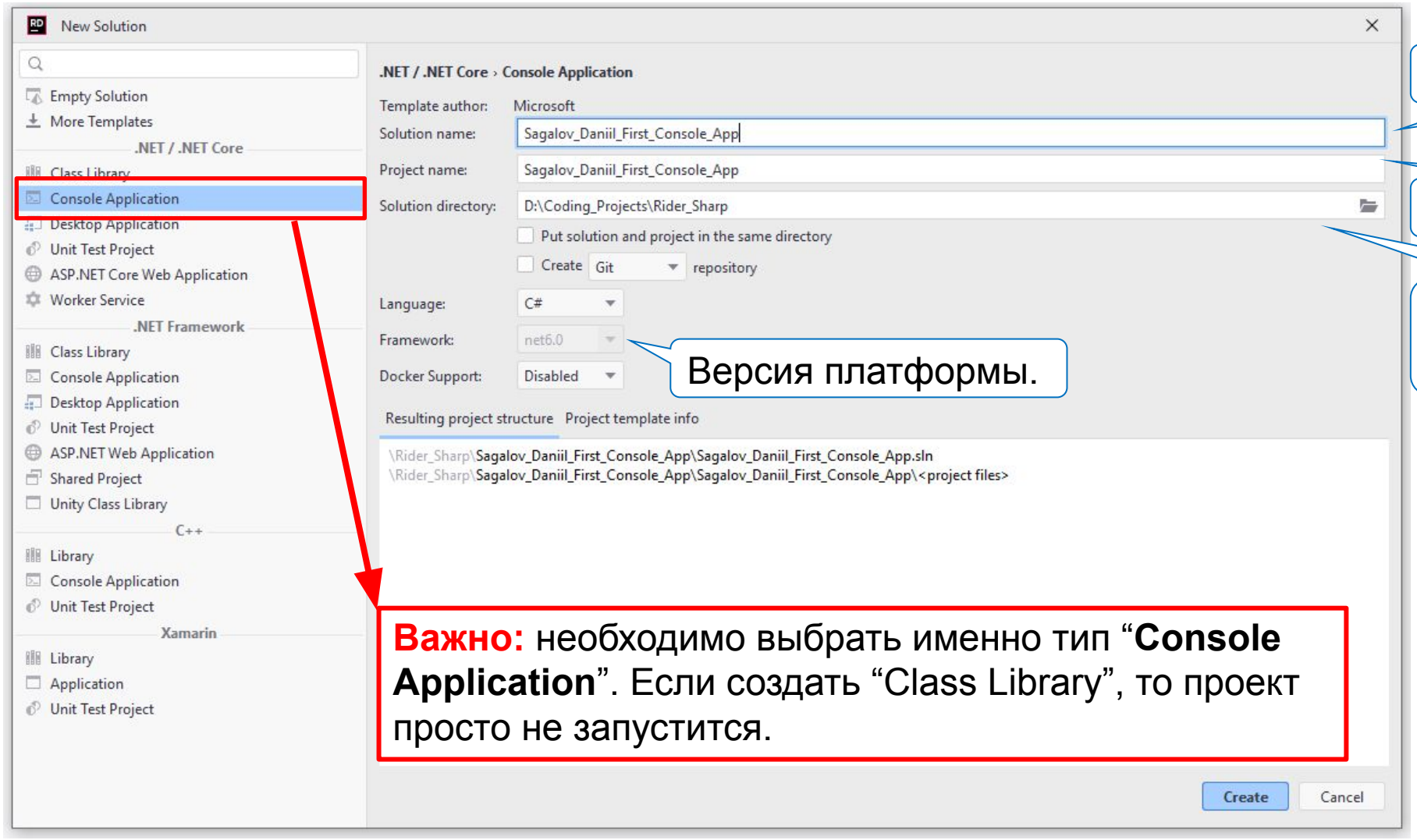
- Запустить JetBrains Rider;
- Ознакомиться с инструкцией по созданию первого решения можно по ссылке: https://www.jetbrains.com/help/rider/Creating_and_Opening_Projects_and_Solutions.html



Стартовый Экран Rider



Создание Решения в Rider



Название решения.

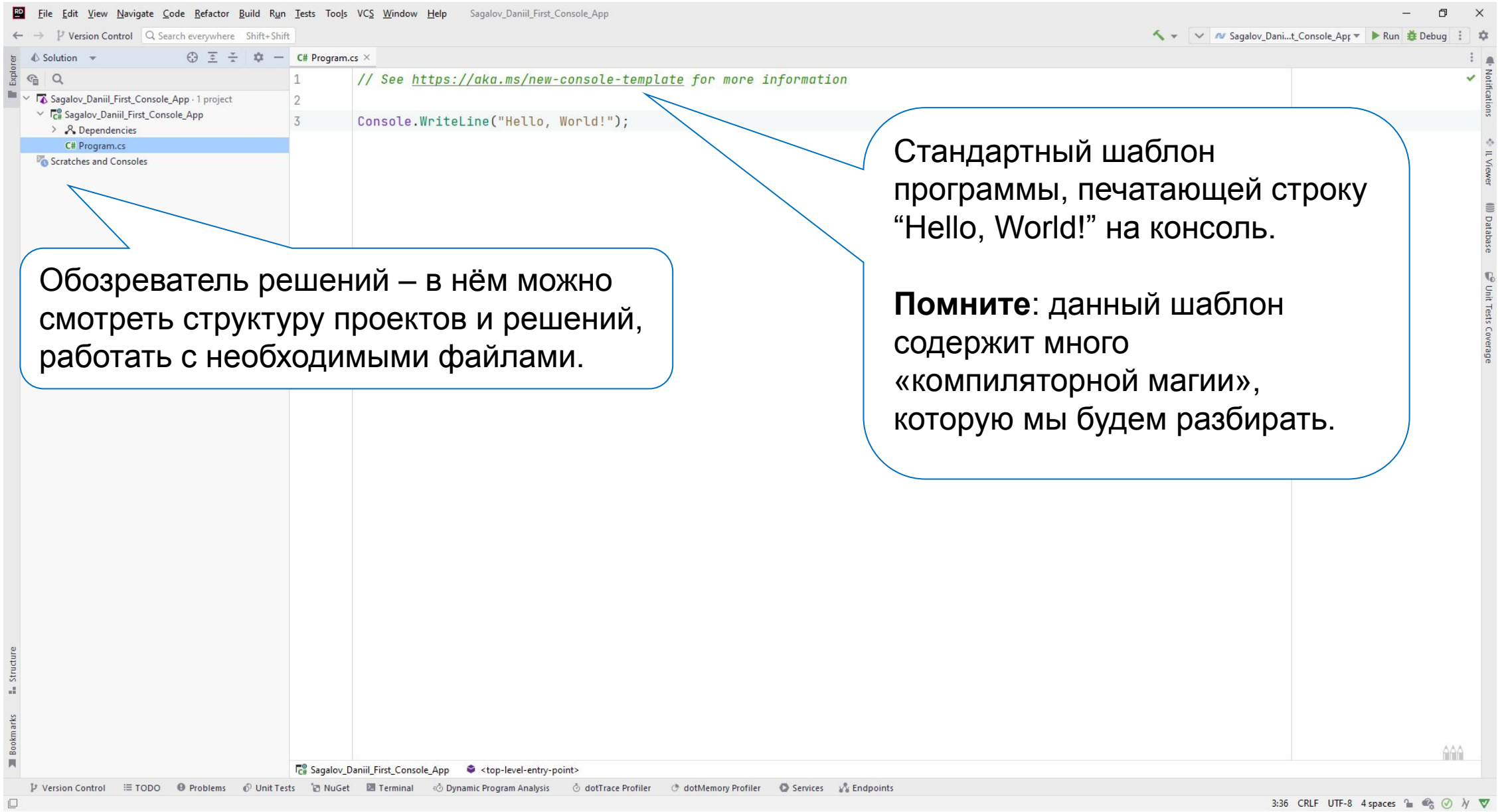
Название проекта.

расположение решения на диске.

Версия платформы.

Важно: необходимо выбрать именно тип “**Console Application**”. Если создать “Class Library”, то проект просто не запустится.

Первая Программа в Редакторе Rider



Обозреватель решений – в нём можно смотреть структуру проектов и решений, работать с необходимыми файлами.

Стандартный шаблон программы, печатающей строку “Hello, World!” на консоль.

Помните: данный шаблон содержит много «компиляторной магии», которую мы будем разбирать.

«Классическая» Первая Программа в Rider

C# Program.cs

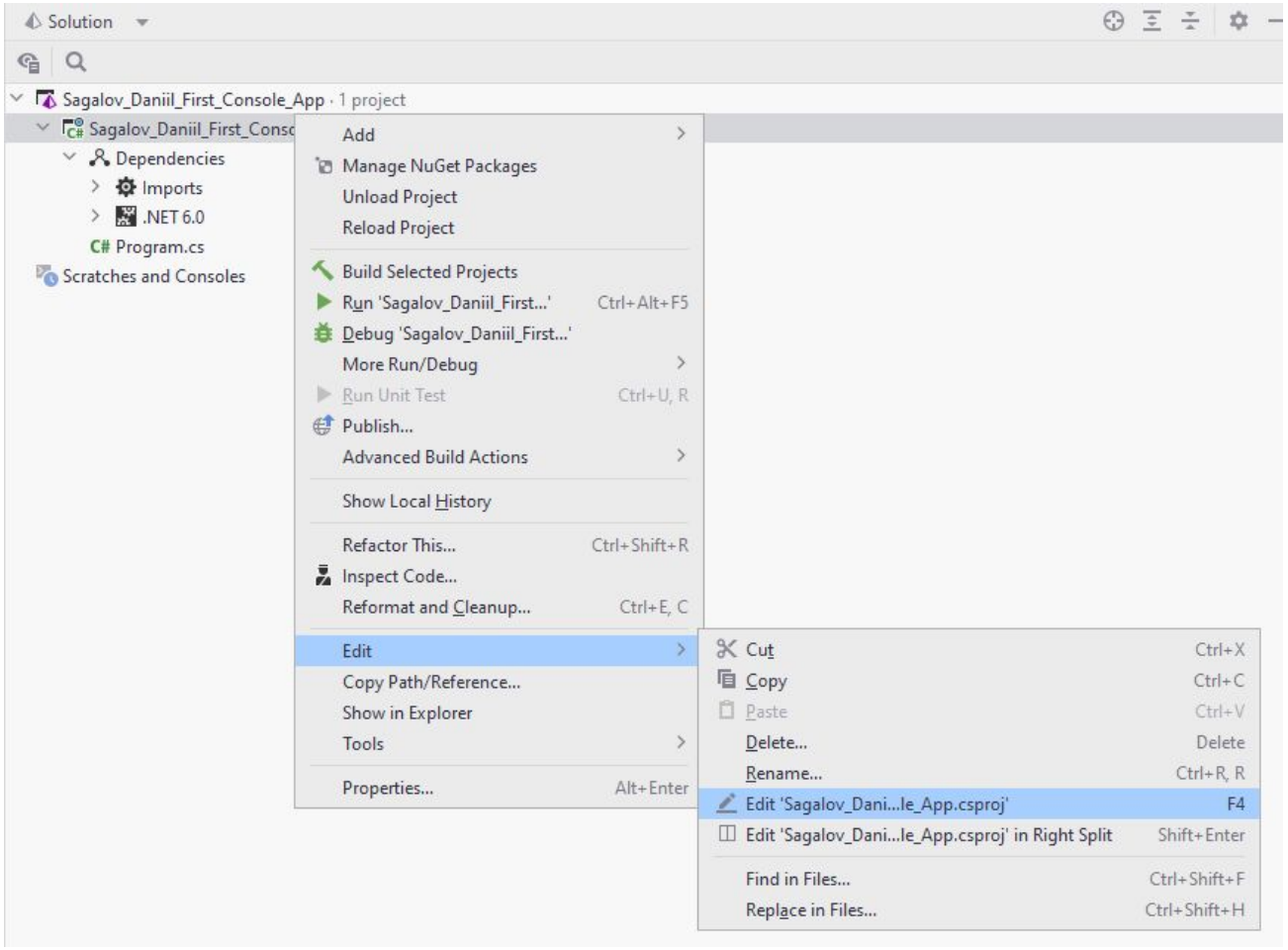
```
1   using System;
2
3   namespace Sagalov_Daniil_First_Console_App
4   {
5       class Program
6       {
7           static void Main(string[] args)
8           {
9               Console.WriteLine("Hello, World!");
10          }
11      }
12  }
```

using System; – позволяет использовать обращение к Console сокращённо (без явного написания System.Console каждый раз).

class Program – объявление типа. Помните: в C# все методы пишутся внутри классов. В прошлом примере данный класс дописан неявно компилятором.

Метод Main – точка входа в программу. В прошлом примере был неявно дописан компилятором за нас.

Файл с Настройками Проекта в Rider



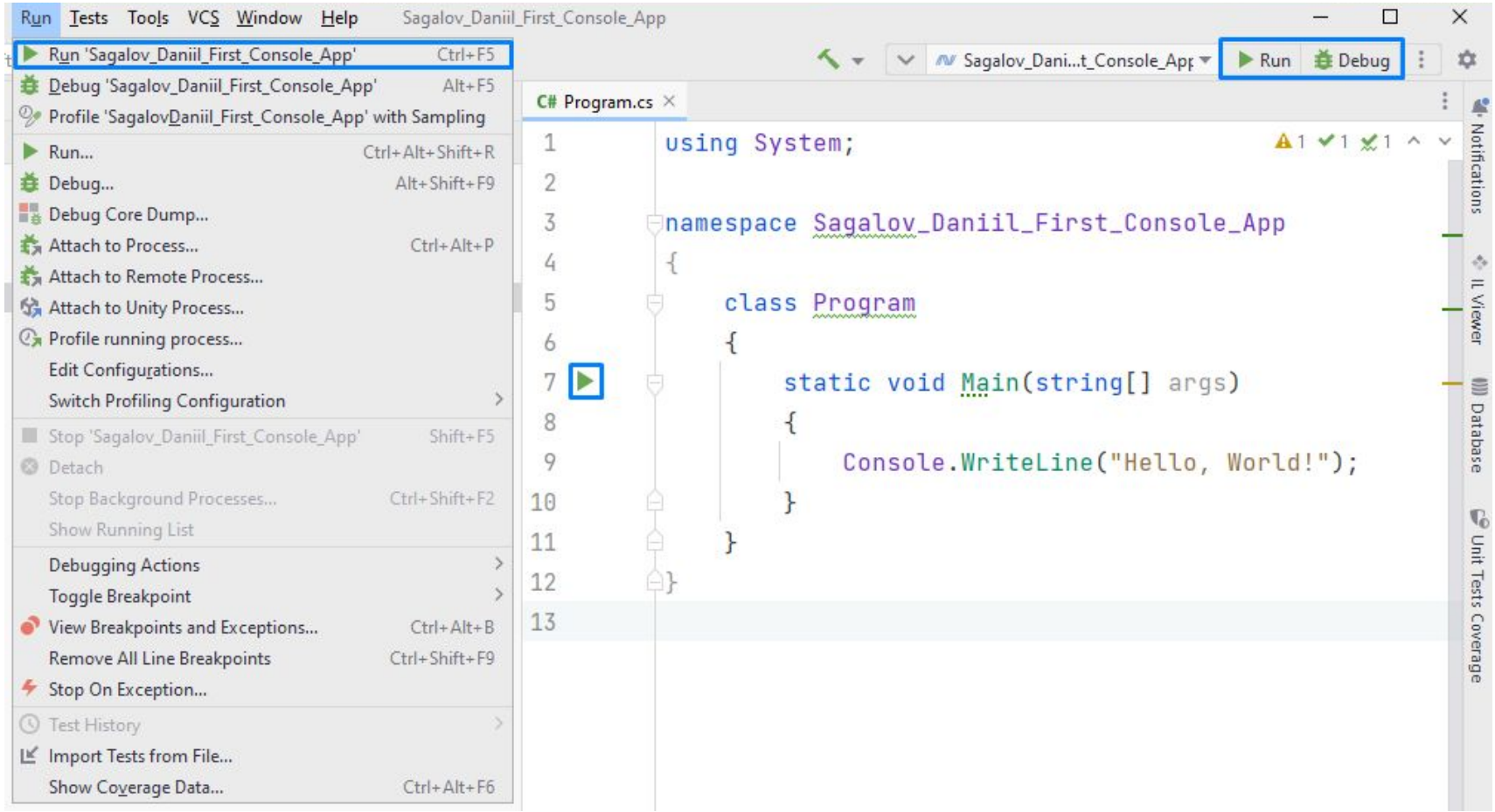
В данном файле можно увидеть тип проекта (например, исполняемый exe-файл), отключить неявные using.

Sagalov_Daniil_First_Console_App.csproj

```
1   <Project Sdk="Microsoft.NET.Sdk">
2
3       <PropertyGroup>
4           <OutputType>Exe</OutputType>
5           <TargetFramework>net6.0</TargetFramework>
6           <ImplicitUsings>enable</ImplicitUsings>
7           <Nullable>enable</Nullable>
8       </PropertyGroup>
9
10  </Project>
11
```

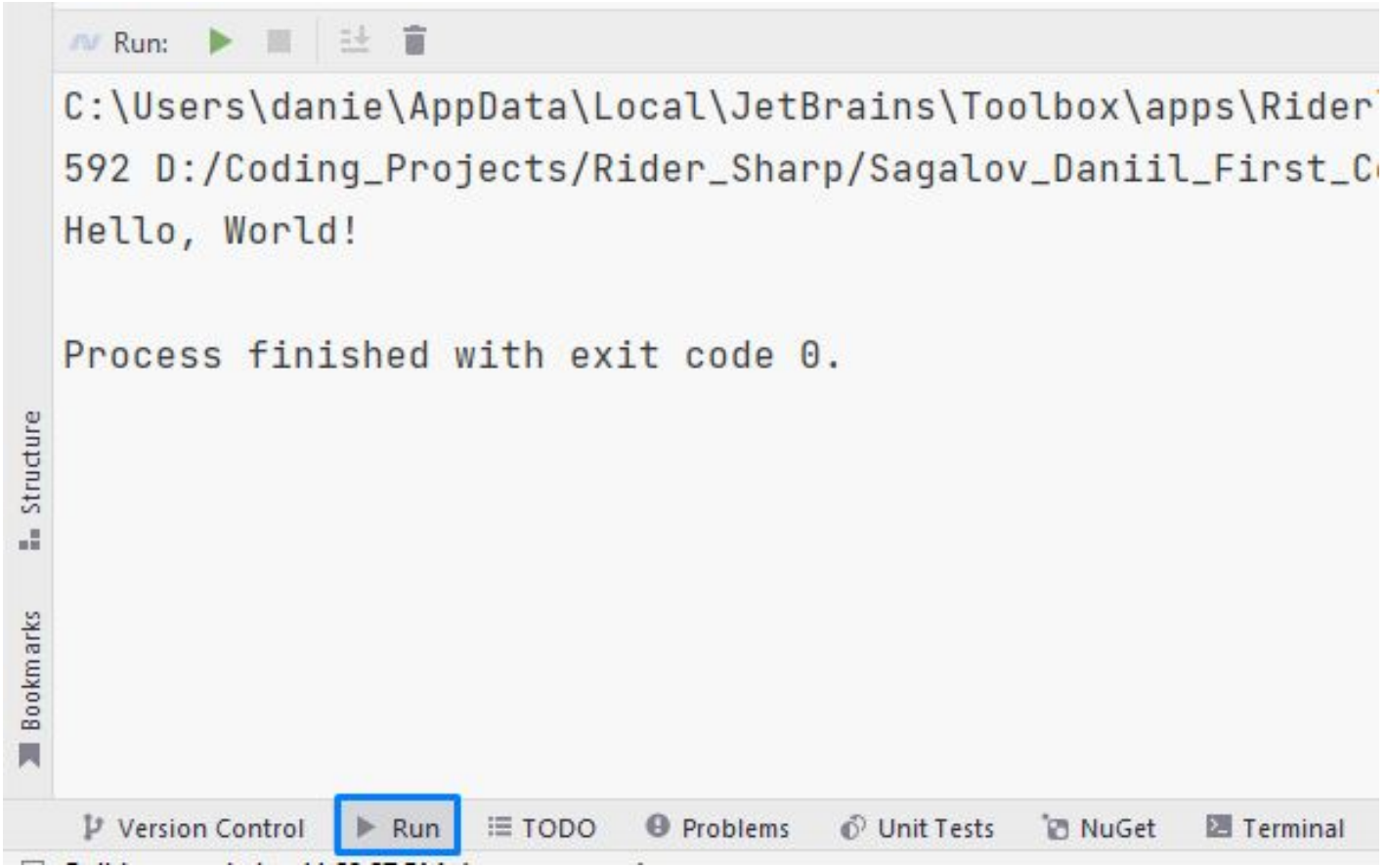
Запуск Проекта в Rider (Множеством Способов)

- Комбинация клавиш Ctrl + F5 (Windows);
- Вкладка Run → Выбрать подходящий вариант;
- Кнопки Run или Debug в правой верхней части экрана;
- Значок запуска рядом с основным методом запускаемого приложения.



Результат Запуска Программы в Rider

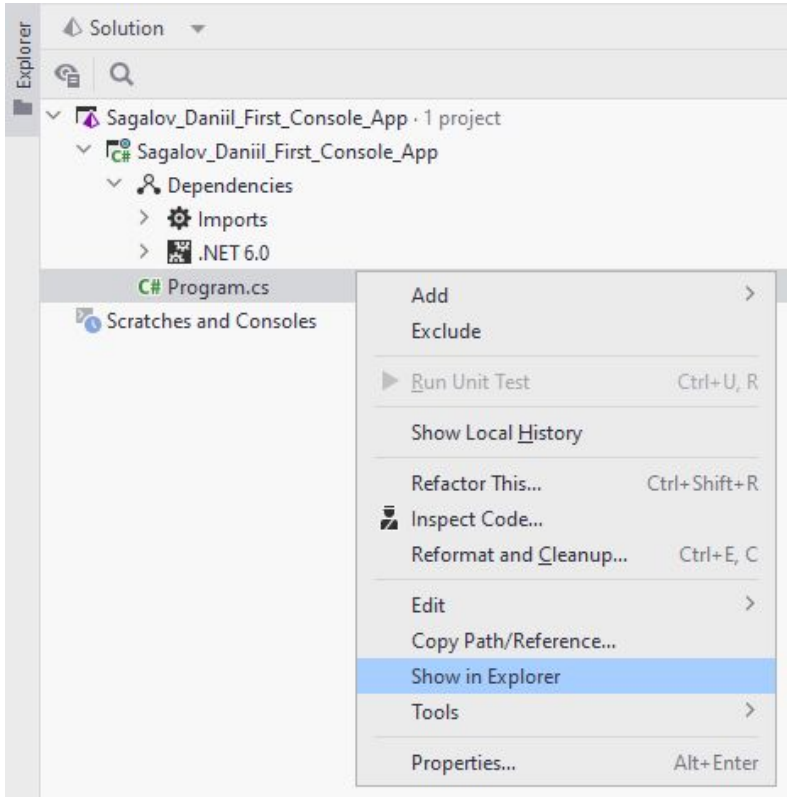
Rider визуализирует результаты работы программы в нижней части экрана во вкладке Run.

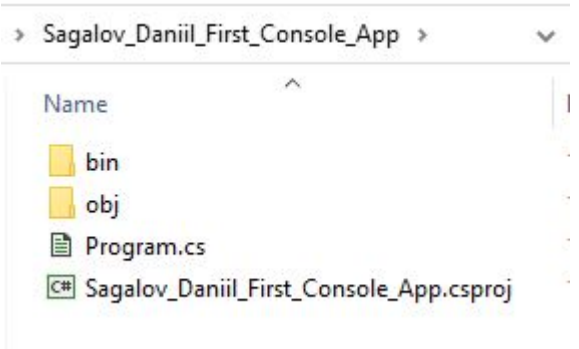


Запуск Программы вне Rider

Простой способ запустить программу вне Rider – ПКМ по файлу с исходным кодом → перейти с помощью “**Show in Explorer**” в папку, содержащую файл с исходным кодом, затем спуститься в **bin** → **Профиль сборки (по умолчанию Debug)** → **Версия .NET** → **ЛКМ по .exe файлу**.

Важно: программа мгновенно завершится, т. к. не ожидает никаких действий со стороны пользователя!





Модификация Программы

```
using System;

namespace Sagalov_Daniil_First_Console_App
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
            Console.WriteLine("Press any key to exit...");
            Console.ReadKey();
        }
    }
}
```

Теперь программа ожидает нажатия любой клавиши, поэтому не закроется автоматически.

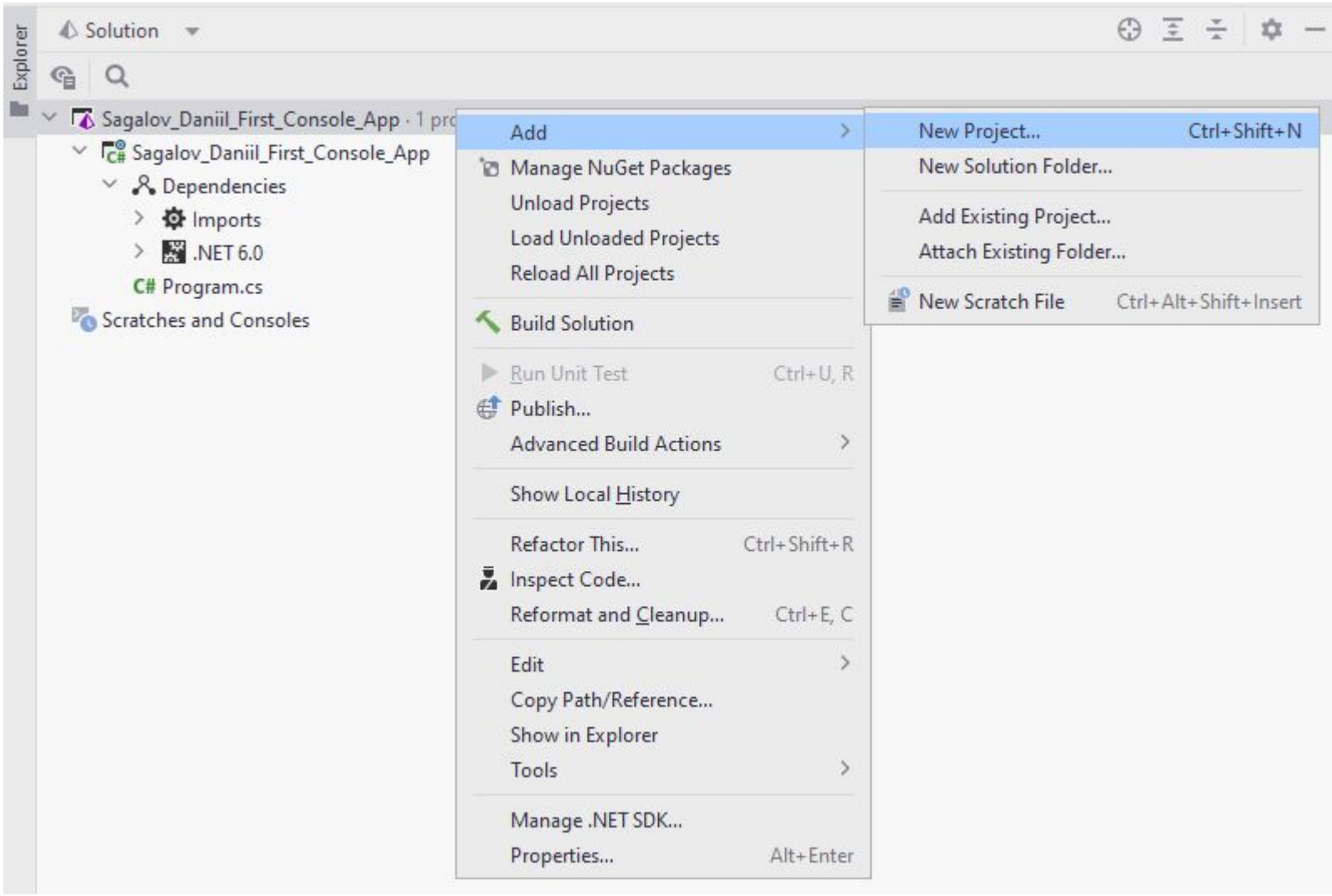
Попробуйте запустить данную программу вне Rider – теперь она не закроется мгновенно.

Создание Второго Проекта в Rider

Удобно иметь несколько запускаемых приложений в одном решении.

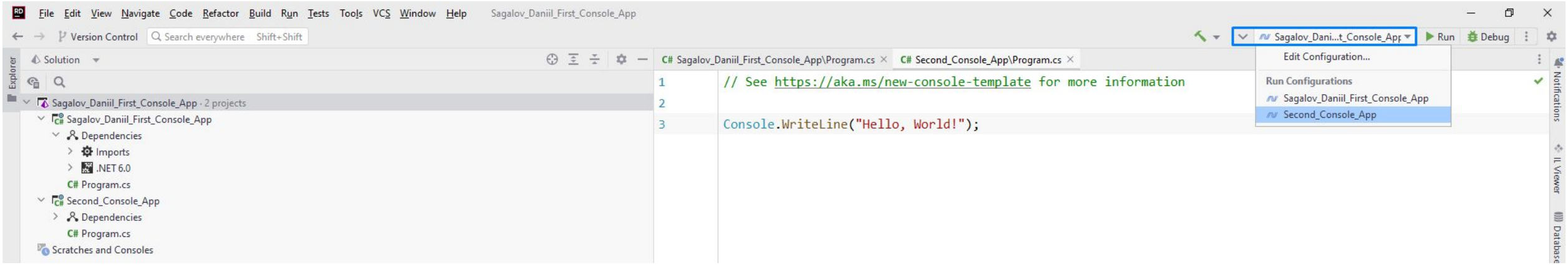
Для этого:

ПКМ по названию <u>решения</u> (не проекта!) → Add → New Project.



Выбор Запускаемого Проекта в Rider

Rider позволяет выбрать запускаемый проект с помощью меню выбора конфигураций в правом верхнем углу интерфейса:



С помощью опции “Edit Configurations” можно изменить настройки запуска проекта.

На заметку: для одного проекта можно создать несколько конфигураций.

Ввод Строк с Консоли

```
using System;

class Program
{
    static void Main()
    {
        string userInput = Console.ReadLine();
        Console.WriteLine(userInput);
    }
}
```

Метод Console.ReadLine() считает строку с консоли (до перехода на новую строку) и вернёт её как значение типа string.

Вывод Строк на Консоль. Простое Форматирование

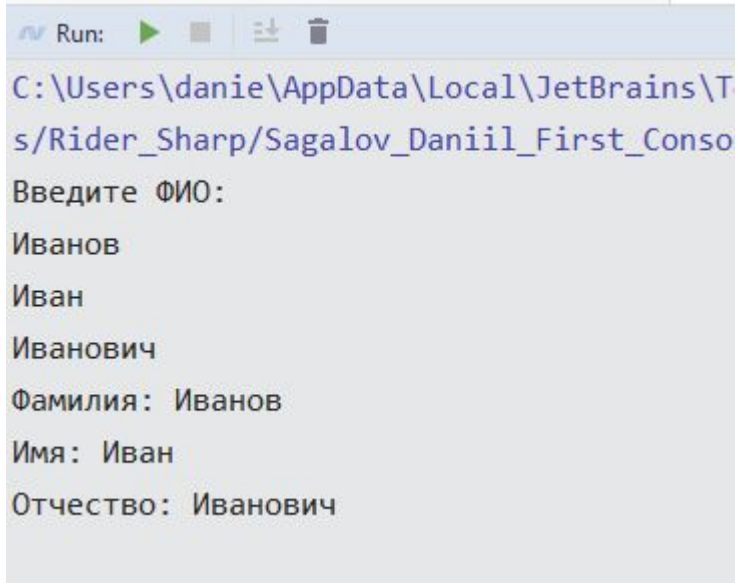
Console.Write() не добавляет переход на новую строку, в отличие от Console.WriteLine().

```
using System;

class Program
{
    static void Main()
    {
        Console.Write("Enter text: ");
        // Рекомендуется описывать переменные максимально близко
        // к месту первого использования в коде.
        string userInput = Console.ReadLine();
        Console.WriteLine("Your input: " + userInput);
        Console.WriteLine("Your input: {0}", userInput);
        Console.WriteLine($"Your input: {userInput}");
    }
}
```

Проект 3: Ввод Строк

1) Изменить код программы второго проекта так, чтобы программа запрашивала имя пользователя и здоровалась с ним по этому имени.

Пример результата:



2) Добавить в решение третий проект с именем Task_03. Программа последовательно запрашивает у пользователя имя, фамилию и отчество. Сохраняет данные в три разные переменные и выводит на экран на отдельных строках в виде анкетных данных.

Ввод:
Иванов
Иван
Иванович

Пример вывода:
Фамилия: Иванов
Имя: Иван
Отчество: Иванович

Проект 4: Арифметические Выражения

1. Создайте в решении проект с именем Task_04;
2. В методе `Main()` инициалируйте переменную `result`:
   ```
   string result = "5 / 3 = " + 5 / 3;
   ```
3. Добавьте вывода значения переменной `result;`
4. Запустите программу (`Ctrl+F5`), добавьте строчку кода, изменяющую значение переменной `result:`
   ```
   result = "5.0 / 3.0 = " + 5.0 / 3.0;
   ```
5. Повторите шаги 3 и 4;
6. Добавьте ещё одну строчку кода, изменяющую значение переменной `result`:
   ```
   result = 5 / 3;
   ```
7. Повторите шаги 3 и 4;
8. Измените строку, добавленную на шаге 7 на:
   ```
   result = 5 / 3 + " - это 5/3";
   ```
9. Выполните шаг 4;
10. Добавьте команду, приостанавливающую выполнение программы до нажатия `Enter;`
11. Скомпилируйте программу;
12. Вспомните, как найти и запустить исполняемый файл вне Rider.

Форматирование Чисел с Помощью ToString()

С помощью вызова ToString() можно форматировать числа:

<выражение>.ToString(“SR”),

где **выражение**:

- Арифметическая константа;
- Переменная встроенного арифметического типа;
- Арифметическое выражение в круглых скобках.

S - спецификатор формата (**D**, **d**, **F**, **f**, **E**, **e**, **G**, **g**,...)

R – натуральное число (количество цифр в изображении числа или его дробной части)

Подробнее о форматировании чисел:
https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-numeric-format-strings

Проект 5: Форматирование с Помощью ToString()

Создайте в решении проект с именем Task_05 и посмотрите результат выполнения следующей программы:

Вывод:
2/3 = 0
2.0/3 = 0.6667
Press any key to exit...

```
using System;

class FirstSeminar
{
    static void Main()
    {
        Console.WriteLine("2/3 = " + (2 / 3).ToString("D"));
        Console.WriteLine("2.0/3 = " + (2.0 / 3).ToString("F4"));
        Console.WriteLine("Press any key to exit...");
        Console.ReadKey();
    }
}
```

Проект 6: Форматирование Таблицы с Числами

Создайте в решении проект с именем Task_06 и, воспользовавшись возможностями метода ToString(), сформируйте и выведите на экран таблицу:

```
*-----------*--------*----------------------*
|*Выражение*|*Формат*|**** Изображение ****|
|-----------|--------|----------------------|
| (5.0/3.0) |   F    | 1,67                 |
| (5.0/3.0) |   F4   | 1,6667               |
| (5.0/3.0) |   E    | 1,666667E+000        |
| (5.0/3.0) |   E5   | 1,66667E+000         |
| (5.0/3.0) |   G    | 1,66666666666667     |
| (5.0/3.0) |   G3   | 1,67                 |
| (5.0/3e10)|   G3   | 1,67E-10             |
|(5.0/3e-10)|   G    | 16666666666,6667     |
| (5.0/3e20)|   G    | 1,66666666666667E-10|
*-----------*--------*----------------------*
```

Форматирование Таблицы с Числами – Результат

```
using System;
class Program {
    static void Main() {
        string top = "|*Выражение*|*Формат*|**** Изображение ****|";
        Console.WriteLine(top);
        string line = "|-----------|--------|--------------------|";
        Console.WriteLine(line);
        string str = "| (5.0/3.0) |  F     | " + (5.0 / 3.0).ToString("F");
        Console.WriteLine(str);
        str = "| (5.0/3.0) |  F4    | " + (5.0 / 3.0).ToString("F4");
        Console.WriteLine(str);
        str = "| (5.0/3.0) |  E     | " + (5.0 / 3.0).ToString("E");
        Console.WriteLine(str);
        str = "| (5.0/3.0) |  E5    | " + (5.0 / 3.0).ToString("E5");
        Console.WriteLine(str);
        str = "| (5.0/3.0) |  G     | " + (5.0 / 3.0).ToString("G");
        Console.WriteLine(str);
        str = "| (5.0/3.0) |  G3    | " + (5.0 / 3.0).ToString("G3");
        Console.WriteLine(str);
        str = "| (5.0/3e10)|  G3    | " + (5.0 / 3e10).ToString("G3");
        Console.WriteLine(str);
        str = "|(5.0/3e-10)|  G     | " + (5.0 / 3e-10).ToString("G");
        Console.WriteLine(str);
        str = "| (5.0/3e20)|  G     | " + (5.0 / 3e10).ToString("G");
        Console.WriteLine(str);
        line = "*-----------*--------*--------------------*";
        Console.WriteLine(line);
    }
}
```

Проект 7: Преобразование Строк в Целые Числа

Создайте в решении проект с именем Task_07:

```
using System;

class Program
{
    static void Main()
    {
        int first;                                          // Объявляем целочисленные
переменные.
        Console.Write("Enter an integer: ");
        string userInput = Console.ReadLine();              // Чтение с клавиатуры строки.
        first = int.Parse(userInput);                       // Преобразование в тип int.

        Console.Write("Enter an integer: ");
        userInput = Console.ReadLine();
        int.TryParse(userInput, out int second);
        Console.WriteLine("Result: " + first + second);   // Вывод.
    }
}
```

Попробуйте:

1. Ввести вместо первого числа строку;
2. Ввести вместо второго числа строку;
3. Заключить **`firstInt + secondInt`** в круглые скобки.

Методы Преобразования Строк – Parse и TryParse

```
int parseResult = int.Parse(str);
```

<тип> <тип>.Parse(<строка>)

Преобразует строковое представление числа в эквивалентное ему значение типа <тип>. В случае, если строковое представление не может быть преобразовано к соответствующему типу, возникает ошибка выполнения – исключение.

```
bool parsedSuccessfully = int.TryParse(str, out int parseResult);
```

bool <тип>.TryParse(<строка>, out <переменная>)

Преобразует строковое представление числа в эквивалентное ему значение типа <тип> и присваивает значение переменной типа <тип>. В случае, если строковое представление не может быть преобразовано к соответствующему типу, метод возвращает значение false, иначе возвращает true.

Проект 8: Операции над Целыми Числами

Создайте в решении проект с именем Task_08. Введите два целых числа типа int **(left, right)** и выведите результаты выполнения следующих операций:

```
(left - right)
(left * right)
(left / right)
(left % right)
(left << right)
(left >> right)
```

Проект 9: Операция Приведения Типов

(<тип>)(выражение)
http://msdn.microsoft.com/ru-ru/library/ms173104.aspx

```
double myPi = 3.1418281828;
int piIntegralPart = (int)myPi;
Console.Write(piIntegralPart);
```

3

```
char letter = 'A';
int letterCode = letter;
Console.Write(letterCode);
```

65

```
int value = 98;
char letter = value;
Console.Write(letter);
```

Ошибка компиляции!!!

Создайте в решении проект с именем Task_09. Введите два вещественных числа типа double. Выведите сумму их дробных частей.

Проекты 10-11

В проекте с именем Task_10 напишите программу, решающую следующую задачу. Программа запрашивает у пользователя три любых слова, после чего печатает их в одну строку, разделяя вместо пробела символом восклицательный знак.

Ввод:
Небо
Солнце
Море

Вывод:
Небо!Солнце!Море

В проекте с именем Task11 напишите программу, решающую следующую задачу. Программа запрашивает у пользователя три любых слова, после чего печатает их в столбик, обрамляя каждое слово слева и справа знаками минус.

Ввод:
Небо
Солнце
Море

Ввод:
-Небо-
-Солнце-
-Море-

Задачи для самостоятельного решения

Создайте решение с именем **<Имя>_<Фамилия>_HW_1** и добавьте в него 3 проекта:

- **Task01:** осуществить вывод на экран строки “Hello, World!”;

- **Task02:** В текстовом виде выведите в консольное окно изображение вашей фамилии, составленное из выбранных вами символов (‘*’, ‘x’, проч.);

- **ASCIIDecoder:** Пользователем вводится корректное число в диапазоне **symbolCode** [**32; 127**]. Выведите на экран изображение символа, представленное в таблице кодов **ASCII** кодом **symbolCode**. При выполнении задания используйте только методы преобразования строк в целочисленные типы и операцию приведения типов.

Задачи для самостоятельного решения

Добавьте в созданное ранее решение ДЗ (см. прошлый слайд) ещё 2 проекта:

- **Task04:** Давайте вспомним физику! Получите от пользователя значения напряжения **U** и сопротивления **R** и вычислите
  1) силу тока:
  $\mathbf{I = U / R}$
  2) потребляемую мощность электрической цепи:
  $\mathbf{P = U^2 / R}$

- **Task05:** На основе ввода пользователем вещественных длин двух катетов, вычислите и выведите на экран длину гипотенузы.