

Laborator 1

Crearea unui proiect nou

Visual Studio 2019

Open recent

Search recent (Alt+S)

Older

Get started



Clone a repository

Get code from an online repository like GitHub or Azure DevOps



Open a project or solution

Open a local Visual Studio project or .sln file



Open a local folder

Navigate and edit code within any folder



Create a new project

Choose a project template with code scaffolding to get started

[Continue without code →](#)

Create a new project

Recent project templates

A list of your recently accessed templates will be displayed here.

Search for templates (Alt+S)

All languages

All platforms

All project types



Blank solution

Create an empty solution containing no projects

Other



Console App (.NET Framework)

A project for creating a command-line application

C#

Windows

Console



Windows Forms App (.NET Core)

A project for creating an application with a Windows Forms (WinForms) user interface

C#

Windows

Desktop



ASP.NET Web Application (.NET Framework)

Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

C#

Windows

Cloud

Web



Class Library (.NET Framework)

A project for creating a C# class library (.dll)

C#

Windows

Library

Class Library (.NET Core)

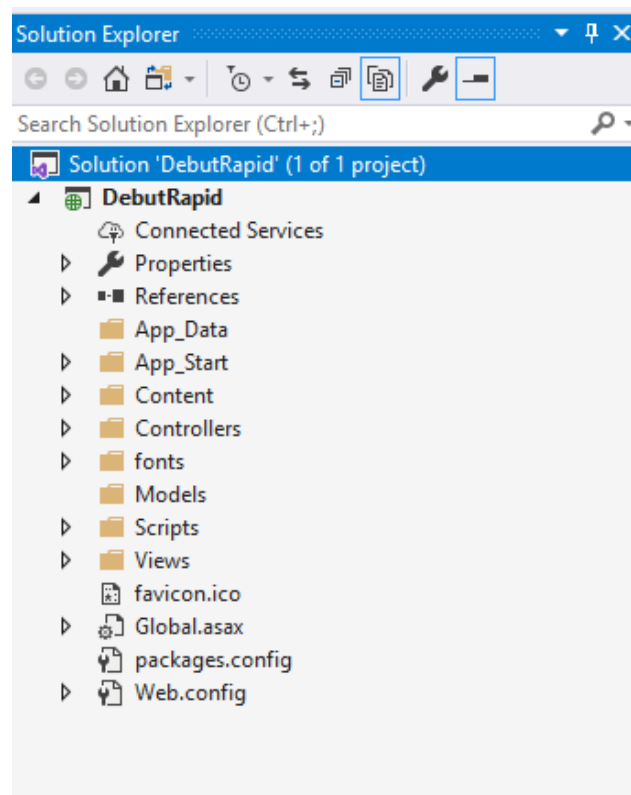
Back

Next

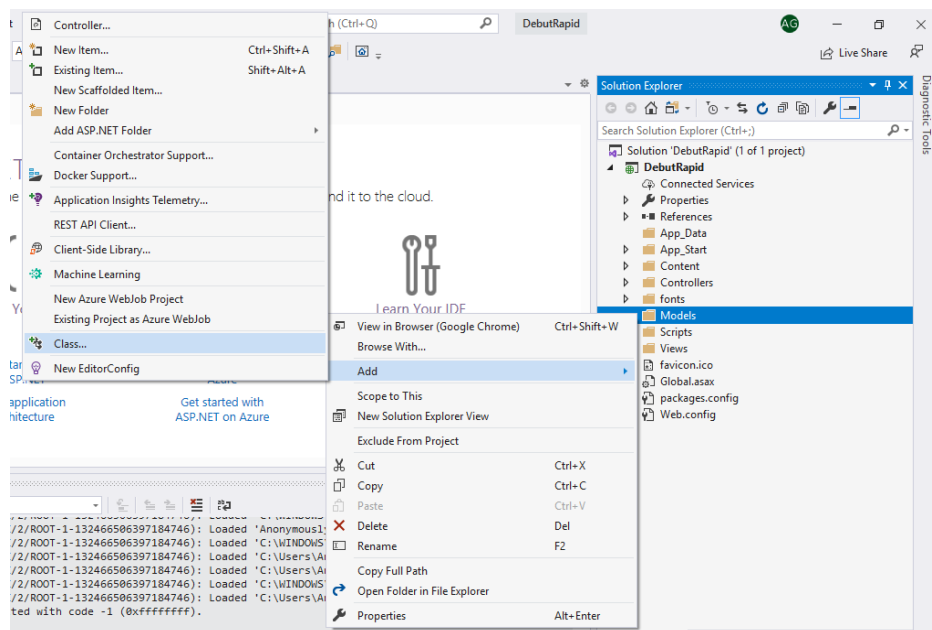
Denumirea proiectului

Denumiti proiectul **DebutRapid**.

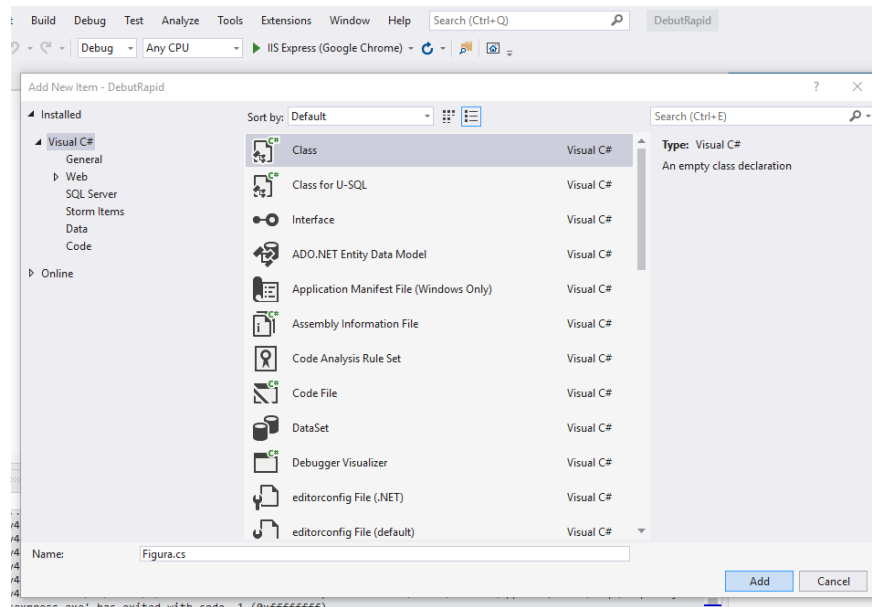
Puteți explora proiectul nou-creat în Solution Explorer.



Crearea unui model



Numim modelul **Figura.cs** si apasam pe butonul Add (clasa se va denumi automat Figura).



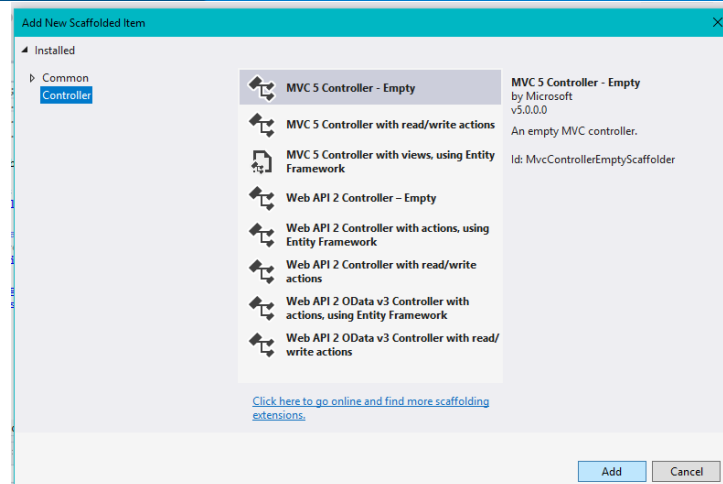
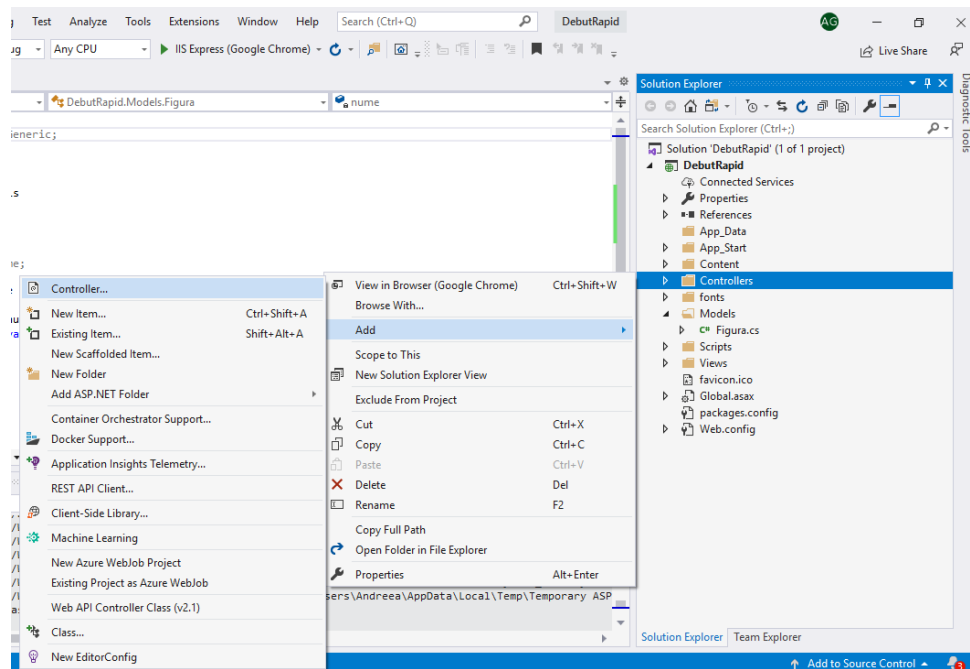
Introducem urmatorul cod in interiorul clasei **Figura**.

```
public class Figura
{
    private string nume;
    public string Nume
    {
        get { return nume; }
        set { nume = value; }
    }
}
```

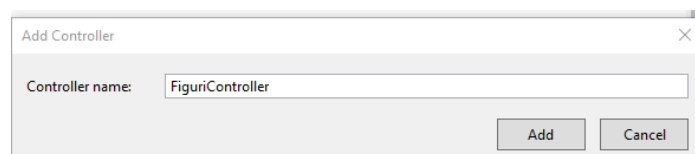
Primul rând introduce o variabilă privată (ca în C++). Restul codului introduce o proprietate ce îmbracă variabila într-o formă public accesibilă. Acest tip de cod este destul de răspândit – el admite și următoarea prescurtare:

```
public string Nume { get; set; }
```

Crearea unui Controller



Denumim controller-ul **FiguriController**.



Adăugăm directiva

`using DebutRapid.Models;`

sub restul directivelor `using`, ca să putem accesa direct, de pildă, clasa **Figura**.

Copiem metoda `Index` (ce va fi la final accesibilă via URL-ul `/Figuri/`) și redenumim copia în `Prima` (ce va fi accesibilă via URL-ul `/Figuri/Prima`).

Modificăm codul metodei **Prima** în următorul:

```
Figura f = new Figura();
f.Nume = "cerc";
return View(f);
```

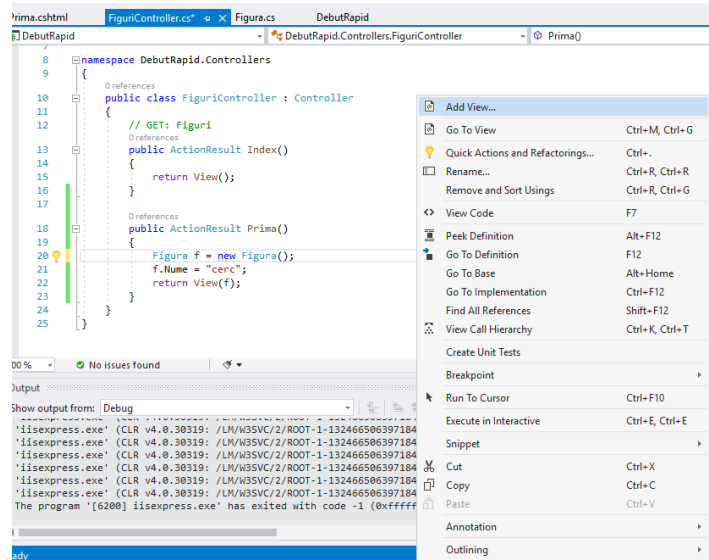
Metoda **Prima** va arata astfel:

```
public ActionResult Prima()
{
    Figura f = new Figura();
    f.Nume = "cerc";
    return View(f);
}
```

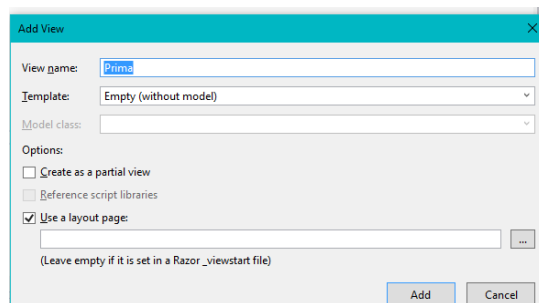
După cum se vede, metoda apelează acum view-ul corespunzător controllerului cu argumentul *f*. Acest view trebuie însă creat.

Crearea unui View

Click-dreapta pe numele metodei **Prima** și selectați **Add View**.



Apăsăm **Add**. Observăm că viewul a fost creat în **Views\Figuri**.



Adăugăm la începutul fișierului directiva

```
@model DebutRapid.Models.Figura
```

Astfel, compilatorul va ști ce fel de obiect este transmis viewului ca argument.

Înlocuim conținutul headerului de tip h2 cu

```
@Model.Nume
```

*Acest cod Razor (cod C# în HTML) va afișa valoarea proprietății **Nume** a obiectului transmis.*

View-ul **Prima** ca arata astfel:

```
@model DebutRapid.Models.Figura
```

```
@{  
    ViewBag.Title = "Prima";  
}
```

```
@Model.Nume
```

Având viewul la vedere, apăsăm Ctrl+F5 pentru a compila și afișa pagina ce se obține apelând metoda corespunzătoare din controller.