

ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej

PWr

Spotkanie 5

Aplikacje webowe na platformie .NET

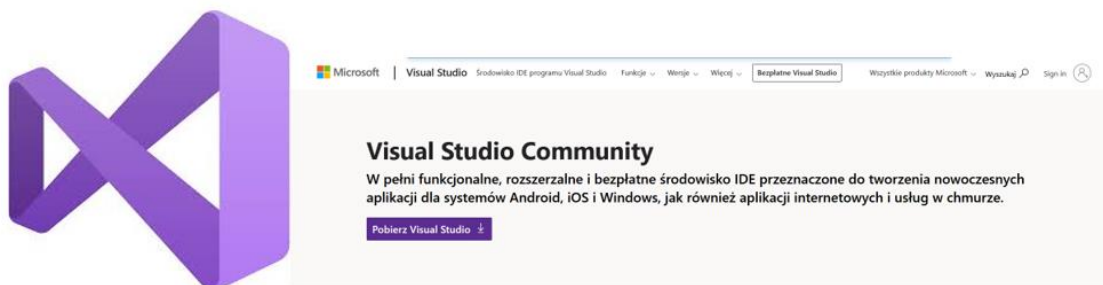
Laboratorium – **Lista 5**

Wstęp.

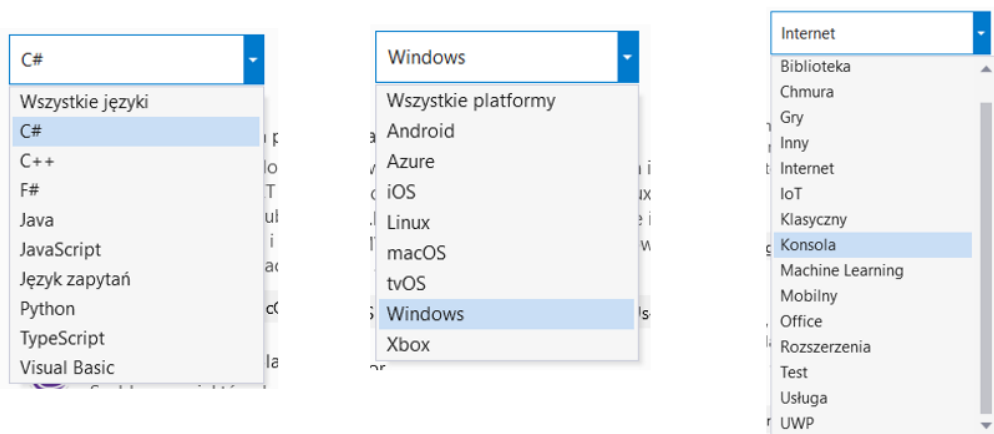
Instalacja i przetestowanie środowiska do tworzenia programów konsolowych i webowych z wykorzystaniem języka C#. Jest to lista techniczna, bez punktów za jej wykonanie.

Visual Studio 2019/2022 Community

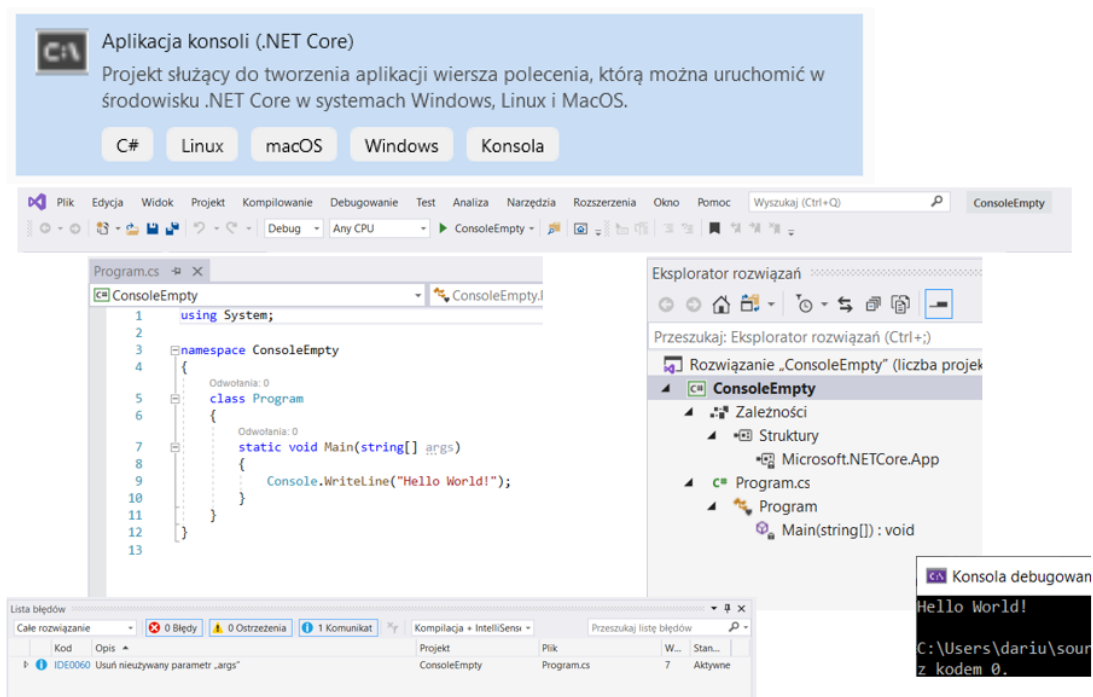
W ramach laboratoriów z tego przedmiotu można używać np. Visual Studio Code doinstalowując sobie odpowiednie wtyczki. Jednak lepszym rozwiązaniem jest przejście na środowisko specjalnie przygotowane do tego celu czyli Visual Studio 2019/2022 Community. Instalację należy zacząć od adresu <https://visualstudio.microsoft.com/pl/vs/community/> gdzie po wybraniu używanego systemu operacyjnego można łatwo i szybko wykonać cały proces dla wersji najnowszej, 2022. Starsze wersje można znaleźć pod adresem <https://visualstudio.microsoft.com/pl/vs/older-downloads/>.



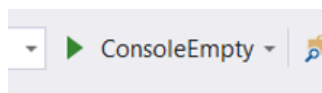
Po poprawnej instalacji należy wybrać odpowiedni typ projektu, gdyż środowisko jest przygotowane do bardzo szerokiej gamy projektów. W przypadku tego kursu ustawienia powinny być jak poniżej (język C#, system Windows lub inny używany, Konsola lub Internet:




Dla poznawania możliwości języka C# używać będziemy typu aplikacji „Konsola”, a gdy zaczniemy tworzyć aplikacje internetowe, to będzie to „Internet”. W przypadku wersji konsolowej powinno się otrzymać projekt wyglądający i działający jak poniżej:



Na powyższym zrzucie z ekranu uruchamianie wykonuje się poprzez wciśnięcie przycisku z zielonym trójkątem i napisem „ConsoleEmpty” (który jest nazwą solucji/projektu).



Gdy wybrany zostanie projekt aplikacji internetowej:

 Aplikacja internetowa platformy ASP.NET Core

Szablony projektów do tworzenia aplikacji internetowych i internetowych interfejsów API platformy ASP.NET Core dla systemów Windows, Linux i macOS przy użyciu platformy .NET Core lub .NET Framework. Twórz aplikacje internetowe z użyciem platform Razor Pages i MVC albo aplikacje jednostronicowe przy użyciu platformy Angular, React lub React + Redux.


[C#](#) [Linux](#) [macOS](#) [Windows](#) [Chmura](#) [Usługa](#) [Internet](#)


jest wiele możliwości dalszych kroków. Na początku skupimy się na projektach we wzorcu MVC, więc powinno się wybrać odpowiednie opcje zgodnie ze zrzutami poniżej:


Create a new ASP.NET Core web application


.NET Core


ASP.NET Core 5.0


**ASP.NET Core Empty**
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.

**ASP.NET Core Web API**
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.

**ASP.NET Core Web App**
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.

**ASP.NET Core Web App (Model-View-Controller)**
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

**ASP.NET Core with Angular**
A project template for creating an ASP.NET Core application with Angular

**ASP.NET Core with React.js**

[Get additional project templates](#)

Authentication

No Authentication
[Change](#)

Advanced

☒ Configure for HTTPS

☐ Enable Docker Support
(Requires [Docker Desktop](#))

Linux

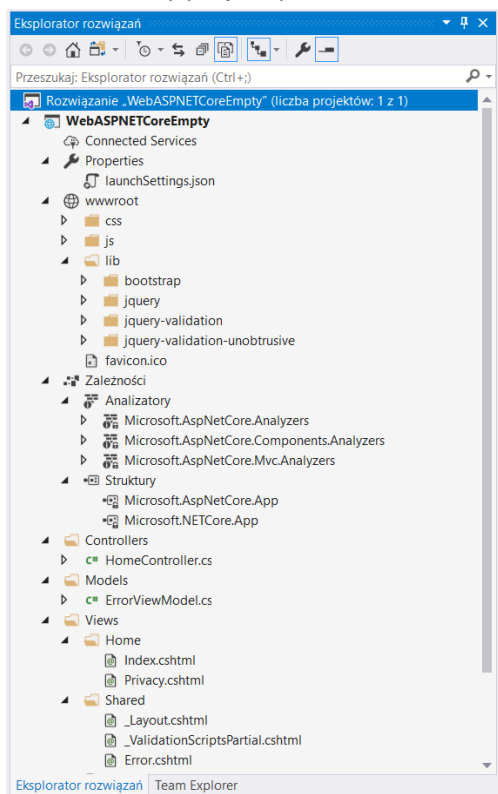
☐ Enable [Bazor](#) runtime compilation

Author: Microsoft
Source: Templates 5.0.11

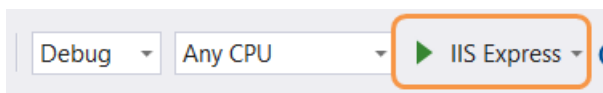
Back

Create

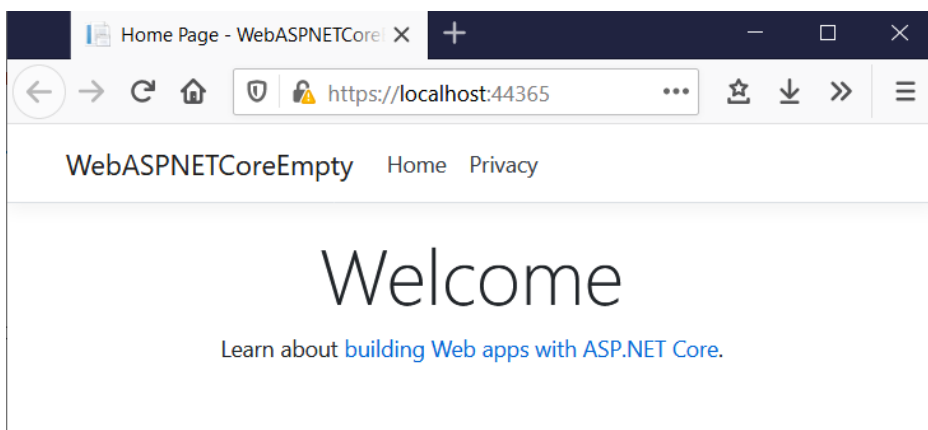
Tak stworzony projekt posiada wiele folderów i powinien wyglądać jak poniżej:



A po uruchomieniu za pomocą przycisku z zielonym trójkątem:

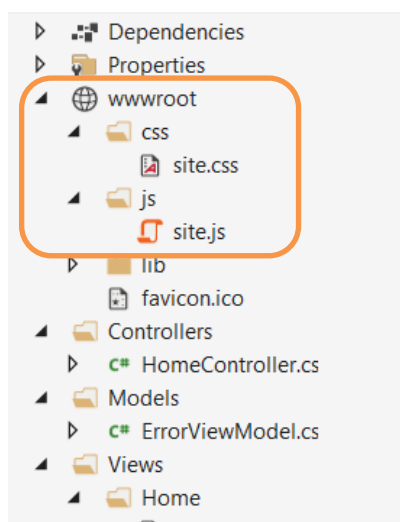


(wówczas uruchomi serwer IIS Express a następnie domyślna przeglądarka z właściwym adresem) strona WWW powinna wyglądać jak poniżej:



Wstawianie poprzedniej listy zadań do nowego projektu webowego ASP .NET

Po stworzeniu projektu webowego (ze wzorcem MVC) zlokalizować pliki i foldery jak na rysunku 5.1:

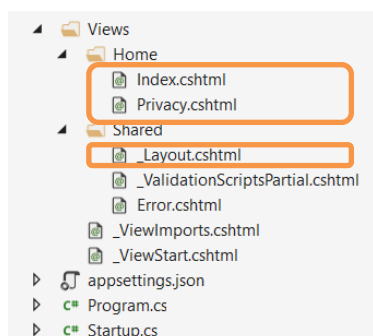


Rys 5. 1

Pliki CSS najlepiej wstawić do folderu `wwwroot/css`, natomiast pliki JavaScript do folderu `wwwroot/js`. Można też przekopiować kod po prostu do istniejących plików `site.css` lub `site.js`.

Jeśli użyjemy nowych plików należy zlokalizować plik `Views/Shared/_Layout.cshtml` (Rys 5.2) i **dopisać link** do stylów CSS na początku pliku (Rys 5.3), natomiast link do pliku JS na końcu tego pliku (Rys 5.4).

Do treści plików HTML wykorzystać należy `Views/Home/Index.cshtml` oraz ewentualnie `Views/Home/Privacy.cshtml`. Początkową zawartość pliku `Index.cshtml` (Rys. 5.5) należy zamienić na zawartość elementu `<body>` dla zadań z poprzedniej listy zadań, czyli np. jak na (Rys 5.6).



Rys 5. 2

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>@ViewData["Title"] - WebAppMVCMinimum</title>
  <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="~/css/site.css" />
</head>
<body>

```

Rys 5.3

```

  </body>
</html>
</script>
<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.js" asp-append-version="true"></script>
@RenderSection("Scripts", required: false)
</body>
</html>

```

Rys 5.4

```

@{
    ViewData["Title"] = "Home Page";
}

<div class="text-center">
  <h1 class="display-4">Welcome</h1>
  <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NET Core</a>.</p>
</div>

```

Rys 5.5

```

@{
    ViewData["Title"] = "Home Page";
}

<table id="myTable"></table>

```

Rys 5.6

List zadań

1. Napisać aplikację konsolową, która pobiera trzy wartości a, b, c współczynników równania kwadratowego $ax^2+bx+c=0$, a następnie wywołuje trzyargumentową metodę, która rozwiązuje to równanie i „jakoś” zwraca wynik (0,1 lub 2 liczby rzeczywiste). Wziąć pod uwagę również **możliwe wartości 0** współczynników a, b, c . Program główny po otrzymaniu wyników wypisuje je na ekran. Gdy jest jedno rozwiązanie wypisuje je używając **formatowania złożonego**. Przy wypisywaniu dwóch rozwiązań używa **interpolacji łańcucha znaków**. Reprezentacja rozwiązania ma być do maksymalnie 5 cyfr znaczących. Preferowany jest zapis bez wykładnika (jeśli jest to możliwe).
2. Napisać aplikację konsolową, w której użytkownik podaje na początku liczbę $n \geq 1$, a następnie n liczb rozdzielonych białymi znakami (spacja/tabulacja/enter). Po wczytaniu ostatniej liczby wypisać **drugą co do wartości** największą wartość spośród tych n liczb. W zadaniu NIE WOLNO użyć tablic, ani żadnych innych kolekcji do pamiętania kolejnych liczb, tylko kilka zmiennych .
Przykładowo dla danych:
7
3 4 2 6 1 6
odpowiedź jest 4.
Gdy wszystkie wartości są równe wypisać „brak rozwiązania”.
3. Stworzyć aplikację webową (we wzorcu MVC, jak na wykładzie). Wstawić rozwiązanie poprzedniej listy w odpowiednie miejsca powstałego projektu i uruchomić go.

Data I: Spotkanie 6 (0 punktów)

Data II: Spotkanie 7 (0 punktów)

Data III: Spotkanie 8 (0 punktów)