Ajax, jQuery, ASP.NET MVC, ASP.NET Web API

Do realizacji ćwiczeń potrzebne jest zintegrowane środowisko programistyczne Microsoft Visual Studio 2015.

Celem ćwiczeń jest wykorzystanie techniki Ajax do komunikacji z usługą zaimplementowaną w ASP.NET Web API z poziomu strony HTML w przeglądarce. Usługa będzie obliczać wyniki dodawania, odejmowania, mnożenia i dzielenia dla podanych dwóch liczb całkowitych.

1. Utwórz nowy projekt (Project) aplikacji webowej ASP.NET.

   a) Uruchom narzędzie Microsoft Visual Studio.

   b) Z menu głównego wybierz File→New→Project. Wybierz szablon ASP.NET Web Application z grupy szablonów dla języka Visual C#. Zaakceptuj zaproponowany katalog lub zmień go na inny gdy nie masz prawa zapisu w proponowanym katalogu. Jako nazwę projektu możesz podać np. „WebApiAjax". Kliknij przycisk OK.

   c) W kolejnym kroku kreatora projektu z listy szablonów ASP.NET wybierz Web API. Pozostałe ustawienia w tym kroku pozostaw domyślne. Kliknij przycisk OK.

2. Z poziomu węzła Models wywołaj operację Add i dodaj w tym folderze nową klasę. Nazwij ją „CalcResult". Klasa ta będzie reprezentować strukturę z wynikami obliczeń, którą Web API będzie serializować do formatów XML i JSON. Jako ciało klasy wklej poniższy kod:

```
public class CalcResult
{
    public int Sum { get; set; }
    public int Difference { get; set; }
    public int Product { get; set; }
    public int Quotient { get; set; }
}
```

2. Z poziomu węzła Controllers projektu wywołaj operację Add→Controller. Wybierz szablon Web API 2 Controller - Empty i kliknij Add. Jako nazwę kontrolera podaj „MathController".

3. W klasie kontrolera dodaj poniższy kod akcji.

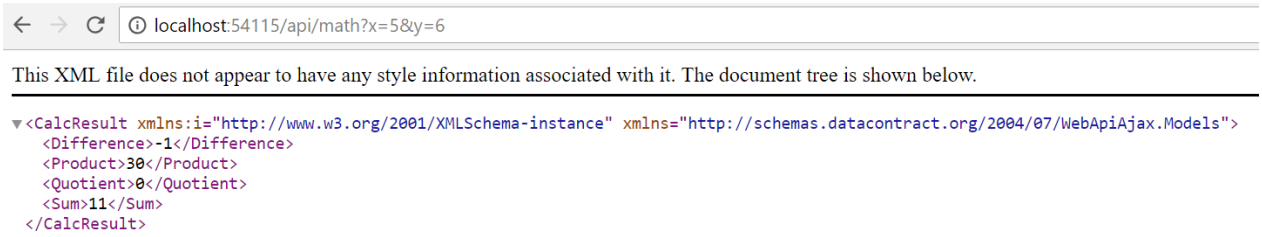
```
// GET api/math
[HttpGet]
public CalcResult Calculate(int x, int y)
{
    return new CalcResult() { Sum = x+y, Difference = x-y,
                              Product = x*y, Quotient = x/y };
}
```

Wynikiem działania metody jest obiekt przygotowanej wcześniej klasy CalcResult. Jego serializacją do formatu XML lub JSON, w zależności od preferencji klienta, zajmie się framework ASP.NET Web API. Web API obsłuży również przekazanie parametrów zawartych w żądaniu HTTP jako parametrów metody kontrolera. Ponieważ nie

zdefiniowaliśmy reguły routingu przewidującej zawarcie liczb na których mają być przeprowadzone obliczenia jako parametrów ścieżkowych adresu URI, będziemy przekazawali te wartości jako parametry w query string w ramach domyślnej reguły routingu. (Web API automatycznie pobiera parametry z obu tych źródeł i wiąże z parametrami metody kontrolera na podstawie zgodności nazw.)

Zwróć uwagę na dodany atrybut informujący o tym, że metoda kontrolera będzie wywoływana w odpowiedzi na żądania GET. Oznaczenie to jest konieczne, gdyż w naszym przypadku z nazwy metody wywnioskowana byłaby inna metoda protokołu HTTP. Czy wiesz która?

4. Uruchom aplikację i poczekaj na otwarcie strony startowej w przeglądarce. Popraw adres na wywołujący zdefiniowaną usługę Web API, przekazując dowolne parametry całkowitoliczbowe jako elementy query string. W efekcie przeglądarka powinna wyświetlić dokument XML zawierający wyniki obliczeń, zgodnie z przykładem na poniższym zrzucie ekranu (w Twoim przypadku konkretne wartości, jak i numer portu na którym dostępna jest aplikacja, mogą być inne).



5. Uruchom ponownie stronę startową aplikacji. Wyświetl źródło strony w przeglądarce. Zauważ że do strony została automatycznie dołączona biblioteka jQuery.

6. Strona startowa jest obsługiwana przez framework ASP.NET MVC i została utworzona przez kreator projektu aplikacji webowej. Odszukaj w projekcie:

a) Kod kontrolera i metodę akcji obsługującą nawigację do strony startowej.
b) Kod widoku strony startowej
c) Szablon strony na którym oparta jest strona startowa

7. W kodzie szablonu strony odszukaj linię odpowiedzialną za dołączenie biblioteki jQuery i przenieś ją z sekcji <BODY> dokumentu HTML do jego sekcji <HEAD>, tak aby można było korzystać z jQuery w skryptach zawartych w ciele dokumentu. Odszukaj plik z biblioteką jQuery w strukturze projektu.

8. Przejdź do edycji strony widoku startowego aplikacji (Home\Index.cshtml). Zastąp całą dotychczasową zawartość tego pliku poniższym kodem.

```
<script type="text/javascript">
$(document).ready(function() {
  $("#calc").click(function () {
    var x = ...;
    var y = ...;

    $.ajax(...);
  });
});
</script>
<div>
    <h1>Calculations</h1>
    <form>
        x = <input type="text" id="x" />
        y = <input type="text" id="y" />
        <input type="button" id="calc" value="Calculate" />
    </form>
    x + y = <span id="sum"></span><br />
    x - y = <span id="difference"></span><br />
    x * y = <span id="product"></span><br />
    x / y = <span id="quotient"></span><br />
</div>
```

9. W miejscu wielokropków samodzielnie dopisz kod realizujący następujące zadania:

a) odczyt wartości wprowadzonych do pól formularza i ich zapisanie w przygotowanych zmiennych
b) wysłanie żądania Ajax i umieszczenie odebranych wyników obliczeń w przygotowanych elementach <SPAN> (jako adres URI dla żądania podaj „/api/math”).

10. Przetestuj działanie aplikacji dla różnych par wartości całkowitoliczbowych.

Zadanie do samodzielnego wykonania

1. Zmień etykietę przycisku na „Call API (jQuery)”.

2. Dodaj obok istniejącego przycisku drugi przycisk z etykietą „Call API (vanilla JS)”.

3. Samodzielnie zaimplementuj obsługę dodanego przycisku tak aby jego działanie funkcjonalnie było takie samo jak pierwszego przycisku, ale aby implementacja była w czystym JavaScripcie (bez jQuery).

- do wysłania żądania do API wykorzystaj bezpośrednio obiekt XMLHttpRequest

- pamiętaj o ustawieniu nagłówka dla wysyłanego żądania, tak aby serwer odpowiedział danymi w formacie JSON, a nie (domyślnym) XML

- do odczytania parametrów z pól formularza i modyfikacji elementów przygotowanych do wyświetlania wyników obliczeń wykorzystaj interfejs DOM HTML-a z poziomu czystego JavaScriptu