Projektowanie aplikacji ASP.NET
Zestaw 2

Podstawy ASP.NET dla .NET Framework

2024-10-15

Liczba punktów do zdobycia: **6/16**
Zestaw ważny do: 2024-10-29

1. (**1p**) Zaimplementować i skonfigurować w potoku przetwarzania handler HTTP, który zwróci użytkownikowi prostą stronę-echo, zawierającą informację o

   - pełnym adresie bieżącego żądania
   - wszystkich nagłówkach HTTP, które zawierało żądanie
   - rodzaju żądania (HTTP verb)
   - treści żądania (jeśli żądanie było typu POST i zawierało niepustą treść)

2. (**2p**) Nauczyć się korzystać z debuggera warstwy HTTP, Fiddlera (`http://www.telerik.com/fiddler`) lub Burpa (`https://portswigger.net/burp/`).

   Pokazać jak przechwytywać ruch z przeglądarki do internetu - jak podglądać zawartości żądań i odpowiedzi i jak stawiać pułapki i za ich pomocą **modyfikować** żądania i/lub odpowiedzi (zakładka Inspectors).

   Pokazać jak za pomocą Fiddlera wysyłać dowolne żądania typu GET i żądania typu POST do własnej strony ASP.NET (czyli jak przygotować w Fiddlerze żądanie, które do naszej aplikacji wyśle Fiddler, a nie przeglądarka; zakładka Composer!).

   (Uwaga! Przygotowanie samodzielnie żądania "od zera" wymaga nieco wprawy, znacznie łatwiej jest wysłać jakieś żądanie z przeglądarki, a potem z listy żądań "przeciągnąć" je do zakładki Composer i wtedy je modyfikować)

   Czym różni się wysyłanie żądania GET od POST z punktu widzenia Fiddlera? Jak odróżnić żądania GET od POST po stronie kodu C# w aplikacji ASP.NET? Pokazać jak w kodzie aplikacji ASP.NET po stronie serwera odczytać parametry przesłane przez GET a jak te przesłane przez POST.

   Uwaga. Można to zadanie pokazać razem z poprzednim - żądania z Fiddlera będą przechwytywane przez moduł na serwerze, a do Fiddlera (tu: klienta) trafi echo żądania.

3. (**1p**) Znać i umieć wyjaśnić różnicę różnymi sposobami przekierowywania żądań:

   (a) hyperlink po stronie przeglądarki

   (b) POST między stronami (strona X pobrana z serwera zawiera formularz wykonujący POST do strony Y)

   (c) `Response.Redirect` na serwerze

(d) `Server.Transfer` na serwerze

Wskazówka: *Redirecting Users to Another Web Forms Page*:
https://msdn.microsoft.com/pl-pl/library/x3x8t37x.aspx.

4. (**2p**) Przygotować aplikację ASP.NET WebForms, która pozwala wprowadzić i wydrukować standardowy "pasek zgłoszenia zadań".

   Aplikacja ma składać się z dwóch stron `*.aspx`: formularza zgłoszenia i formularza wydruku.

   Na formularzu zgłoszenia (`start.aspx` użytkownik aplikacji powinien mieć możliwość wpisania imienia i nazwiska, daty, nazwy zajęć i numeru zestawu oraz wyników kolejnych deklarowanych 10 zadań z odpowiednią liczbą punktów.

   Program powinien kontrolować poprawność wpisywanych danych (na przykład - nie pozwolić wpisać tekstu do pola z liczbą punktów) ale nie musi wymagać wypełnienia wszystkich pól (można na przykład przyjąć konwencję że brak wpisu to deklarowane 0 punktów).

   Jeśli dane nie są prawidłowe - użytkownik powinien otrzymać stosowny komunikat i możliwość korekty.

   Po zaakceptowaniu formularza zgłoszenia, aplikacja powinna przekierować użytkownika do strony "formularza wydruku" (`print.aspx`) gdzie przedstawiony pasek zgłoszenia w postaci możliwej do wydrukowania (czyli pozbawiony "kontrolek" a zamiast tego ułożony w postaci tabelki HTML)

   Pokazać jak użyć **różnych** metod przekazywania danych między stronami:

   (a) `Response.Redirect` z parametrami w pasku adresowym (ze strony wyjściowej), odczyt `Request.QueryString` (na stronie docelowej)

   (b) kontener serwerowy `Session`

   Wskazówka: *How to: Pass Values Between ASP.NET Web Forms Pages*:
   https://msdn.microsoft.com/en-us/library/6c3yckfw.aspx.

Wiktor Zychla