Projektowanie aplikacji ASP.NET
Zestaw 2

Podstawy ASP.NET

15-10-2019

Liczba punktów do zdobycia: **6/16**
Zestaw ważny do: 29-10-2019

1. (**1p**) Zaimplementować i skonfigurować w potoku przetwarzania moduł 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 zawierało niepustą treść)

2. (**1p**) Nauczyć się korzystać z debuggera warstwy HTTP, Fiddlera (`http://www.telerik.com/fiddler`).

   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 wyklikać w Fiddlerze żądanie, które do naszej aplikacji wyśle Fiddler, a nie przeglądarka; zakładka Composer!).

   Czym różni się wysyłanie żądania GET od POST po stronie Fiddlera? Jak odróżnić takie żądania po stronie kodu 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**) Nauczyć się korzystać z debuggera warstwy HTTP, 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 Proxy/Options, Proxy/Intercept).

   Pokazać jak za pomocą Burpa wysyłać dowolne żądania typu GET i żądania typu POST do własnej strony ASP.NET (czyli jak wyklikać w Burp żądanie, które do naszej aplikacji wyśle Burp, a nie przeglądarka; zakładka Repeater!).

   Uwaga. Podobnie jak w poprzednim zadaniu, można to zademonstrować na module z zadania pierwszego.

4. (**1p**) Wyjaśnić różnicę różnymi sposobami przekierowywania żądań:

   (a) hyperlink po stronie przeglądarki

   (b) POST między stronami

   (c) `Response.Redirect` na serwerze

   (d) `Server.Transfer` na serwerze

   Wskazówka: Redirecting Users to Another Web Forms Page.

5. (**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 użytkownik aplikacji powinien mieć możliwość wpisania imienia i nazwiska, daty, nazwy zajęć i numeru zestawu oraz kompletu wyników kolejnych deklarowanych 10 zadań z odpowiednią liczbą punktów. Program powinien kontrolować poprawność wpisywanych danych.

   Po zaakceptowaniu formularza zgłoszenia, użytkownik powinien w przeglądarce zobaczyć formularz wydruku: pasek zgłoszenia w postaci możliwej do natychmiastowego wydrukowania.

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

   (a) `Response.Redirect` z argumentami (strona wyjściowa) oraz `Request.QueryString` (strona docelowa)

   (b) `PostBackUrl` na przycisku (strona wyjściowa) oraz `PreviousPage` (strona docelowa)

   (c) kontener serwerowy `Session`

   Jeden punkt za każdą zaprezentowaną metodę.

   Wskazówka: How to: Pass Values Between ASP.NET Web Forms Pages.

Wiktor Zychla