Projektowanie aplikacji ASP.NET
Zestaw 1

Podstawy ASP.NET

08-10-2019

Liczba punktów do zdobycia: **10/10**
Zestaw ważny do: 22-10-2019

1. (**1p**) Upewnić się, że na lokalnej maszynie jest zainstalowany serwer **Internetowe Usługi Informacyjne**. Jeśli serwer był doinstalowywany, upewnić się że na liście **Ograniczenia ISAPI i CGI** podsystem ASP.NET w wersji 4.0 ma status **Dozwolone**.

2. (**2p**)

   - za pomocą przystawki **Menedżer internetowych usług informacyjnych** na serwerze IIS utworzyć nową witrynę na nagłówku hosta `ap1.myserver.com`
   - w lokalnej mapie hostów (`Windows/System32/Drivers/etc/hosts`) nagłówek hosta zmapować na lokalną maszynę (127.0.0.1)
   - dla witryny wybrać jakąś lokalizację na dysku (np. `C:\inetpub\ap1`)
   - upewnić się, że tożsamość w ramach której IIS odczytuje pliki statyczne (użytkownik IUSR lub grupa IIS_IUSRS) ma uprawnienia do odczytu zawartości foldera
   - na witrynie umieścić plik statyczny `index.html` z przykładową zawartością
   - dowolną przeglądarką internetową nawigować do `http://ap1.myserver.com`, upewnić się że serwer poprawnie odpowiada na żądanie i strona pokazuje się w przeglądarce

3. (**1p**) Pokazać, że jedną i tę samą witrynę można udostępnić na wielu różnych nagłówkach hostów. Do czego mogłaby przydać się taka możliwość?

4. (**1p**) W konfiguracji witryny na serwerze aplikacji umieć wskazać miejsce definiowania filtrów kojarzących rozszerzenia zasobów z logiką ich przetwarzania. Co się stanie jeśli na serwerze aplikacji umieścimy zasób o rozszerzeniu, na które nie mapuje się żaden filtr, a następnie spróbujemy do serwera wysłać żądanie o wydanie tego zasobu?

   Zademonstrować to na przykładzie eksperymentu z zasobem o rozszerzeniu np. ***.foo**: utworzyć w aplikacji zasób o takim rozszerzeniu i spróbować go pobrać z poziomu przeglądarki.

5. (**1p**) W Visual Studio utworzyć projekt witryny ASP.NET z jedną stroną `*.aspx` (Web Form), strona może mieć zawartość wyłącznie statyczną (nie musi mieć żadnych interaktywnych elementów) (wygodnie jest startować od ustawienia *Empty project* i samodzielnie dodawać zawartość, w przeciwnym razie z szablonu zostanie utworzony projekt który zawiera całkiem dużo domyślnych elementów).

   Uruchomić aplikację w Visual Studio na serwerze deweloperskim (IIS Express) (ustawienia projektu) na wybranym przez siebie numerze portu.

Czym różni się serwer deweloperski (IIS Express) od serwera produkcyjnego (IIS)?

6. (**1p**) Zbudowaną w poprzednim zadaniu aplikację nauczyć się przenosić na serwer IIS. Formalnie - użyć opcji **Publish** do publikacji zawartości. Który rodzaj publikacji (FTP, HTTP, File system...) jest najwygodniejszy w sytuacji gdy zarówno serwer deweloperski jak i produkcyjny znajdują się na tej samej maszynie? A jak radzić sobie z publikacją na zdalny serwer produkcyjny?

7. (**1p**) Do domyślnego formularza strony `*.aspx` dodać dwa elementy interaktywne, pole tekstowe (**asp:TextBox**) oraz przycisk powodujący odesłanie formularza na serwer (**asp:Button**).

   Przy uruchamianiu aplikacji serwer przetłumaczy te tagi *serwerowe* na elementy DOM zrozumiałe dla przeglądarki. Jakie elementy DOM odpowiadają tym konkretnym tagom serwerowym? Jak to podejrzeć w przeglądarce? A jak na serwerze podejrzeć wynik kompilacji strony `*.aspx` na kod C# wykonujący się podczas każdego żądania?

8. (**2p**) Do strony `*.aspx` dodać kod wykonujący się po stronie serwera (może być w zdarzeniu `Page_Load`). W tym kodzie wykonującym się po stronie serwera spróbować otworzyć do odczytu wskazany plik tekstowy z dysku serwera (**StreamReader sr = ...**), umieszczony na dysku w jakiejś innej lokalizacji niż pliki witryny (np. `C:\temp`).

   Zawartość pliku wypisać na stronie (można to zrobić na wiele sposobów, np. zapisując zawartość do kontrolki **asp:Label** czy bezpośrednio przez tag `<#= #>`).

   Wykonać eksperyment polegający na uruchomieniu aplikacji w Visual Studio na serwerze IIS Express. Plik powinien odczytać się poprawnie - IIS Express uruchamia się na tożsamości tego użytkownika który zalogowany jest do systemu.

   Ale kod wykonujący się na serwerze produkcyjnym IIS ma przez system operacyjny przypisaną tożsamość (właściciela) i nie jest nim zalogowany użytkownik (serwer produkcyjny działa nawet wtedy kiedy do systemu nie jest zalogowany **żaden** użytkownik). Nauczyć się więc wskazywać tożsamość dla procesu puli aplikacji w ustawieniach witryny na przystawce IIS.

   Wykonać eksperyment, który udowodni, że kod wykonujący się po stronie serwera jest ograniczony uprawnieniami puli aplikacji.

   Do eksperymentu przygotować dwie różne witryny w różnych fizycznych lokalizacjach na dysku. Dla każdej z witryny utworzyć osobną pulę aplikacji. Jako tożsamości pul aplikacji utworzyć i przypisać dwa **różne** konta w systemie operacyjnym (te konta można nazwać na przykład `ap1` i `ap2`).

   Pokazać, że próba otwarcia do odczytu pliku w kodzie serwerowym wymaga nadania jawnie w systemie operacyjnym uprawnień do pliku (lub foldera zawierającego plik) dla konta określającego tożsamość puli aplikacji. Pokazać co się dzieje w przypadku niewystarczających uprawnień (konto puli aplikacji nie ma uprawnień dostępu do pliku).

   Ten eksperyment pokazuje, że administrator serwera ma pełną kontrolę nad tym do jakich zewnętrznych zasobów (pliki, bazy danych itp.) ma dostęp kod witryny uruchamiany na serwerze. Administrator powinien z tej możliwości korzystać w sposób świadomy.

   Dlaczego nie jest dobrą praktyką przełączanie tożsamości puli aplikacji na LocalSystem?

Wiktor Zychla