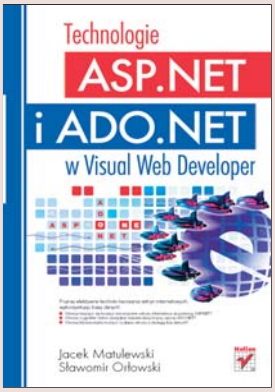
Technologie ASP.NET i ADO.NET w Visual Web Developer

Autor: Jacek Matulewski, Sławomir Orłowski
ISBN: 978-83-246-0738-9
Format: B5, stron: 296



Firma bez własnej witryn internetowych to obecnie rzadkość, a wiele aplikacji sieciowych korzysta z baz danych. Za pośrednictwem stron internetowych klienci mogą na przykład zamawiać usługi i kupować produkty oraz przeglądać katalogi z ofertą sklepu, a firmy mogą między innymi zarządzać magazynem czy danymi klientów. Technologia ASP.NET umożliwia błyskawiczne tworzenie rozbudowanych i nowoczesnych witryn internetowych, a dzięki ADO.NET można szybko dodać do nich obsługę baz danych. Bezpłatne środowisko Visual Web Developer Express pozwala każdemu przekonać się o ogromnych możliwościach tych narzędzi.

Książka „Technologie ASP.NET i ADO.NET w Visual Web Developer" zawiera wszechstronny opis technik i narzędzi do tworzenia zaawansowanych witryn internetowych za pomocą ASP.NET i ADO.NET. Dzięki niej nauczysz się korzystać z tych technologii w środowisku Visual Web Developer oraz używać podstawowych języków do programowania stron ASP.NET. Dowiesz się, jak wygodnie zarządzać bazami danych w aplikacjach sieciowych i jak sprawnie umieścić gotową witrynę na serwerze. Przykładowe studia przypadku pokazują, jak zastosować omawiane technologie w praktyce, a w części poświęconej AJAX-owi zobaczysz, jak przyspieszyć działanie swych witryn.

- Wprowadzenie do środowiska Visual Web Developer
- Korzystanie z języków C# i XML w projektach ASP.NET
- Zapewnianie spójnego stylu za pomocą wzorców stron
- Zarządzanie sesjami i plikami cookie
- Korzystanie z ADO.NET do obsługi baz SQL Server 2005, Access i XML
- Tworzenie i używanie usług sieciowych
- Budowanie kompletnych rozwiązań sieciowych
- Przyspieszanie witryn za pomocą AJAX-a
- Dodawanie do witryny kontrolek AJAX-a

Wypróbuj ASP.NET wraz z ADO.NET i przekonaj się, że tworzenie rozbudowanych witryn internetowych z obsługą baz danych nie musi być trudne.

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

Spis treści

Rozdział 4.

Mechanizmy spajające witrynę ASP.NET

Jacek Matulewski

Wzorzec strony (master page)

W przypadku witryny zawierającej kilka stron warto posłużyć się wzorcem. **Wzorzec** (ang. *master page*) to zwykła strona ASP.NET zapisana do pliku z rozszerzeniem *.master*. Wyróżnia ją jednak to, że zawiera komponenty `ContentPlaceHolder`, które rezerwują miejsce do wypełnienia przez strony korzystające ze wzorca. Poza tym wzorzec może zawierać zwykłe elementy HTML, jak i komponenty ASP.NET.

Wzorzec służy jako szablon pozostałych stron projektu. Tworząc nowe strony projektu, możemy wskazać ich wzorzec, a wówczas w widoku projektowania, zamiast edytować całą stronę, będziemy edytować jedynie te miejsca, które we wzorcu zostały zarezerwowane komponentami `ContentPlaceHolder`.

W najprostszym przypadku można posłużyć się wzorcem do ujednolicenia nagłówków i stopek wszystkich stron witryny — wówczas wzorzec zawiera tylko jeden komponent `ContentPlaceHolder`. I właśnie na takim przykładzie nauczymy się teraz tworzenia wzorców i korzystania z nich.

Przygotowywanie projektu

Nie można, a przynajmniej nie jest to proste, przypisać strony master od istniejącej witryny ASP.NET. Dlatego utworzymy zupełnie nowy projekt — Czytelnik odpocznie wreszcie od *Kolorów* — w którym pierwszą czynnością będzie usunięcie domyślnie utworzonej strony *Default.aspx*.

1. Tworzymy nowy projekt:
   a) z menu *File* wybieramy *New Web Site…*,
   b) zaznaczamy pozycję *ASP.NET Web Site*,
   c) w rozwijanej liście *Location* wybieramy *File System* (wartość domyślna),
   d) a w rozwijanej liście *Language — Visual C#*,
   e) klikamy *OK*.
2. Z projektu usuwamy stronę *Default.aspx*:
   a) zaznaczamy ją w oknie projektu (podokno o nazwie *Solution Explorer*),
   b) rozwijamy menu kontekstowe
   c) i wybieramy z niego polecenie *Delete*;
   d) pojawi się pytanie o potwierdzenie, w którym należy kliknąć przycisk *OK*.

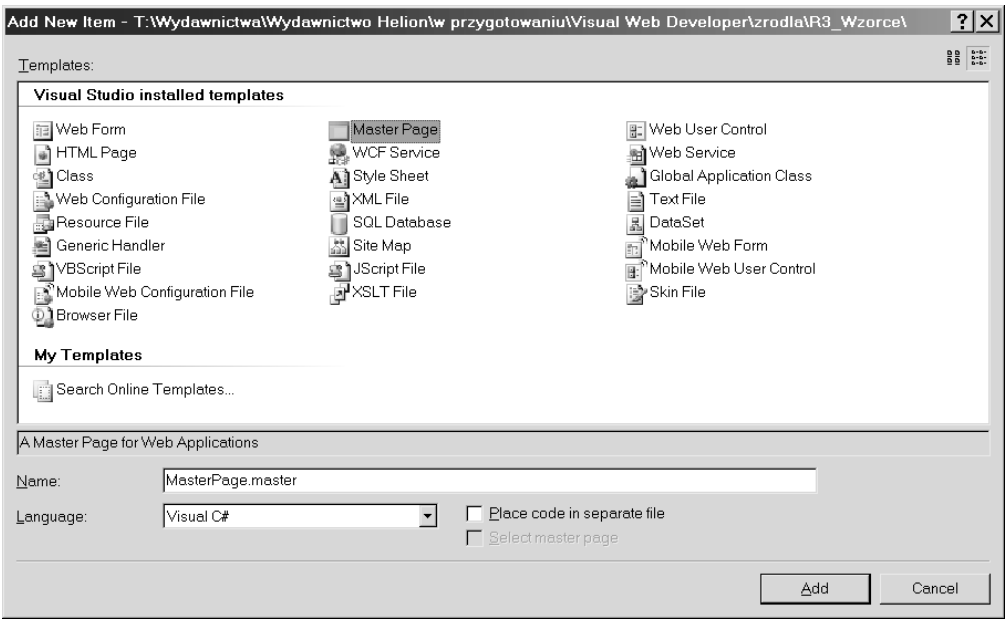
To usunie plik strony nie tylko z projektu, ale także z dysku. W tym miejscu umieścimy nową wersję strony o nazwie *Default.aspx*, ale korzystającą ze wzorca. Wcześniej musimy oczywiście przygotować wzorzec. Nie będziemy w tym zbyt wymyślni — zdefiniujemy prosty nagłówek oraz stopkę strony i zadowolimy się jednym komponentem `ContentPlaceHolder`.


Uwaga

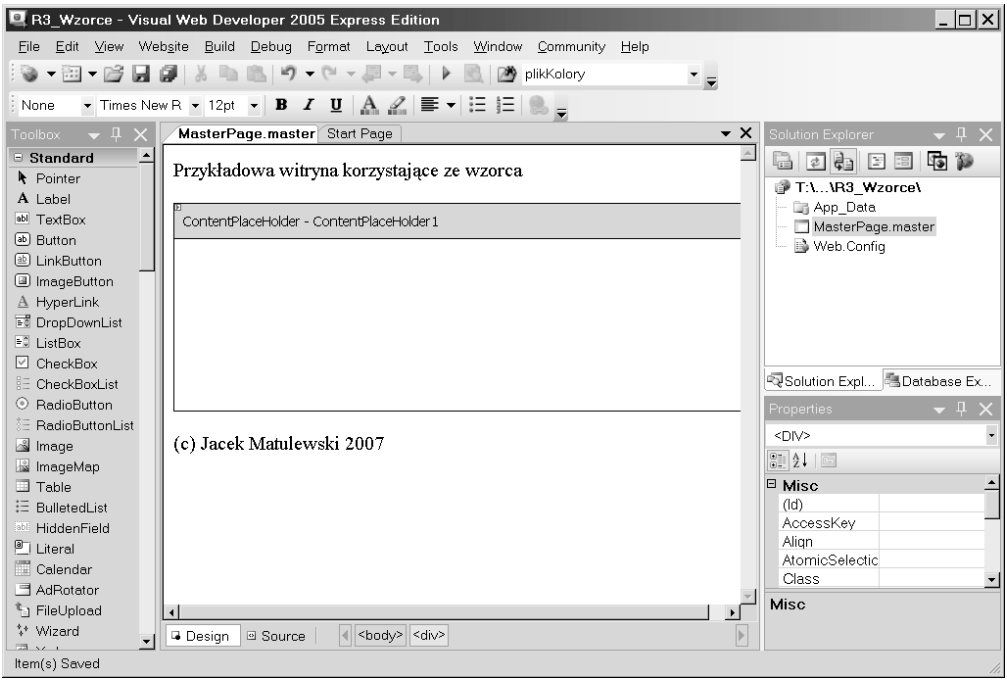
Nie należy mylić usuwania pliku (także z dysku), a więc polecenia *Delete*, z usuwaniem pliku z projektu, tj. z poleceniem *Exclude From Project*.

Definiowanie wzorca

1. W oknie projektu zaznaczamy główną gałąź reprezentującą projekt całej witryny (a nie katalog *App_Data*).
2. Z menu *File* wybieramy *New File…*.
3. W oknie *Add New Item* (rysunek 4.1) zaznaczamy pozycję *Master Page*.
4. Klikamy *Add*. Do projektu dodany zostanie nowy plik *MasterPage.master*. W edytorze zobaczymy jego kod.
5. Przejdźmy od razu do widoku projektowania nowej strony (zakładka *Design* w dole okna). Zobaczymy na niej tylko jeden komponent klasy `ContentPlaceHolder` — to jest miejsce, które będzie zapełniane przez strony korzystające z naszego wzorca. Jeżeli chcemy dodać ich więcej, to komponent ten znajdziemy na zakładce *Standard*.
6. Umieśćmy powyżej i poniżej tego komponentu jakiś tekst pełniący rolę nagłówka i stopki stron naszej witryny. Przykład widoczny jest na rysunku 4.2.



Rysunek 4.1. *Polecenie dodawania pliku jest czułe na zaznaczoną pozycję w oknie projektu — aby zobaczyć wszystkie możliwe rodzaje plików, należy zaznaczyć pozycję odpowiadającą całemu projektowi*



Rysunek 4.2. *Osadzanie komponentu ContentPlaceHolder we wzorcu witryny*

Strona korzystająca ze wzorca

Nasz wzorzec jest prosty, żeby nie powiedzieć prymitywny, ale nie o estetykę teraz chodzi, a o ideę. Stworzymy zatem przykładowe strony, które będą z tego wzorca korzystać.

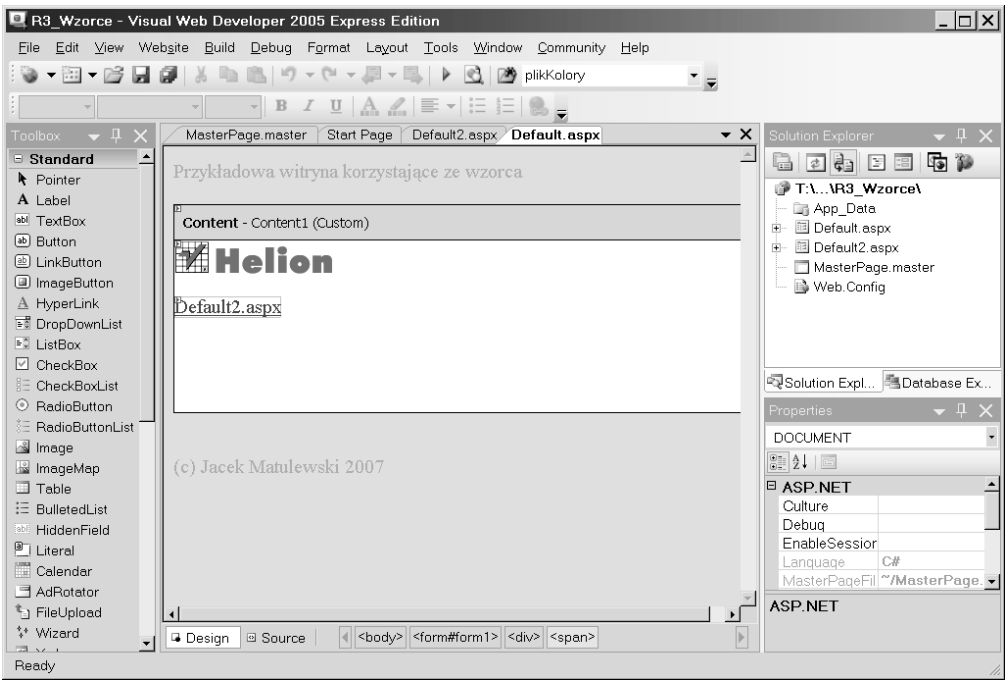
1. Z menu *File* wybieramy ponownie pozycję *New File*.
2. Tym razem zaznaczamy ikonę *Web Form*.
3. Koniecznie musimy zaznaczyć pole opcji *Select master page*. Tylko w momencie tworzenia strony można wskazać jej wzorzec.
4. Zalecam również zaznaczenie opcji *Place code in separate file*, dzięki której ewentualne metody zdarzeniowe będą umieszczone w osobnym pliku.
5. Musimy wskazać jeszcze nazwę pliku strony — domyślnie jest to *Default.aspx* — oraz język użyty do programowania metod zdarzeniowych. Jak już się pewnie Czytelnik zorientował, w tej książce zalecamy używanie C#.
6. Wreszcie klikamy *Add*.
7. Natychmiast pojawi się okno *Select a Master Page*. Wskazujemy w nim stronę *MasterPage.master* i klikamy *OK*.

Po utworzeniu strony znajdziemy się w edytorze kodu. Przejdźmy niezwłocznie do widoku projektowania. Zobaczymy w nim stronę wzorca, ale poza obszarem wyznaczonym przez komponent `ContentPlaceHolder` jest ona niedostępna do edycji (rysunek 4.3). Miejscem, w którym możemy umieszczać nasze komponenty, jest wyłącznie miejsce zarezerwowane wcześniej we wzorcu.

Wypełnijmy miejsce przeznaczone na stronę jakąś przykładową zawartością. Umieśćmy w nim np. komponent `HyperLink`. Za pomocą okna *Properties* do jego właściwości `ImageUrl` przypiszmy adres *http://helion.pl//img/logo162_35.gif*, natomiast do właściwości `NavigateUrl` adres *http://helion.pl*. Na podglądzie powinniśmy zobaczyć natychmiast logo Wydawnictwa Helion (por. rysunek 4.3). Możemy stworzyć teraz kolejne strony korzystające z tego samego wzorca, który ujednolici ich wygląd. Wzorzec poprawia zatem spójność całej witryny. Nie do przecenienia jest fakt, że stopkę i nagłówek wszystkich stron witryny kontrolujemy z jednego pliku, zatem jeżeli chcemy coś do nich dodać lub je zmienić, wystarczy edytować tylko plik wzorca.

Site map i komponent SiteMapPath

Pozostańmy przy tym samym projekcie, a nawet dorzućmy do niego jeszcze kilka stron. Jeżeli witryna ma więcej stron (mowa o kilkunastu, kilkudziesięciu), warto pomyśleć o *site map* — mapie witryny. W ASP.NET przygotowanie takiej mapy polega na utworzeniu pliku XML o nazwie *Web.sitemap*, w którym znajduje się zhierarchizowana grupa elementów `siteMapNode`. W atrybutach każdego z nich wskazujemy adres strony z witryny, jej tytuł i ewentualnie opis. Struktura znaczników ma odzwierciedlać logiczną strukturę strony, wskazując strony nadrzędne i ich podstrony. Ilość stopni hierarchii jest w zasadzie dowolna.



Rysunek 4.3. *Edycja stron korzystających ze wzorca ogranicza się do obszarów wyznaczonych na wzorcu przez komponent ContentPlaceHolder*

W naszym projekcie jest tylko kilka stron (załóżmy, że trzy: *Default.aspx*, *Default2.aspx* i *Default3.aspx*), ale i my zdefiniujemy plik *Web.sitemap*. Przyjmijmy, że *Default.aspx* jest stroną tytułową jakiegoś działu witryny o nazwie „Łącza do ważnych stron", a *Default2.aspx* i *Default3.aspx* są zwykłymi stronami tego działu. Z menu *File* wybieramy pozycję *New File...* i w oknie *Add New Item* wskazujemy pozycję *Site Map*. Powstanie plik *Web.sitemap*. Uzupełniamy go według wzoru z listingu 4.1.

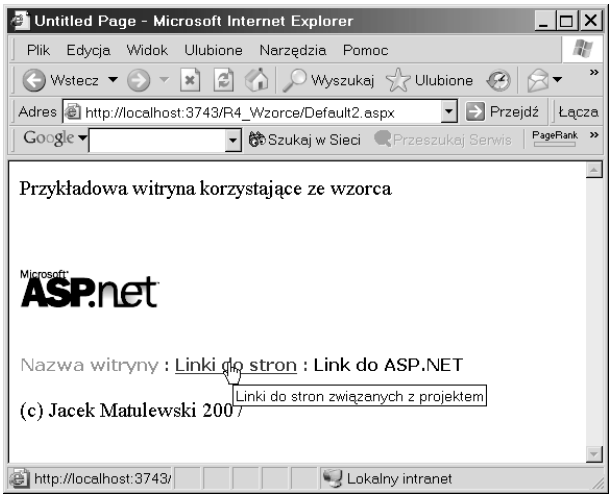
Listing 4.1. *Plik Web.sitemap to plik XML opisujący logiczną strukturę witryny ASP.NET na potrzeby komponentów nawigacyjnych*

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="" title="Nazwa witryny"  description="Opis witryny">
    <siteMapNode url="Default.aspx" title="Linki do stron"  description="Linki do
    stron związanych z projektem" />
      <siteMapNode url="Default2.aspx" title="Link do ASP.NET"  description="Linki
      do oficjalnej strony ASP.NET" />
      <siteMapNode url="Default3.aspx" title="Link do MSDN"  description="Link do
      strony dokumentacji MSDN2" />
    </siteMapNode>
  <siteMapNode url="" title=""  description="" />
  </siteMapNode>
</siteMap>
```

> W elemencie `siteMap` może być tylko jeden element `siteMapNode`, więc ewentualną rozbudowę powyższej struktury należy zacząć od trzeciego poziomu zagnieżdżenia elementów XML.

Plik *Web.sitemap* może być źródłem danych dla komponentów umieszczanych na stronach, które pozwalać będą internaucie na zorientowanie się w pozycji oglądanej strony w strukturze całej witryny i szybkie przejście do innych jej stron. Na początek przyjrzyjmy się komponentowi `SiteMapPath`.

1. Przejdźmy do widoku projektowania wzorca *MasterPage.master*.
2. Umieśćmy na nim komponent `SiteMapPath` z zakładki *Navigation*. Komponent ten pokazuje ścieżkę aktualnej strony w strukturze zdefiniowanej w pliku *Web.sitemap* (rysunek 4.4).



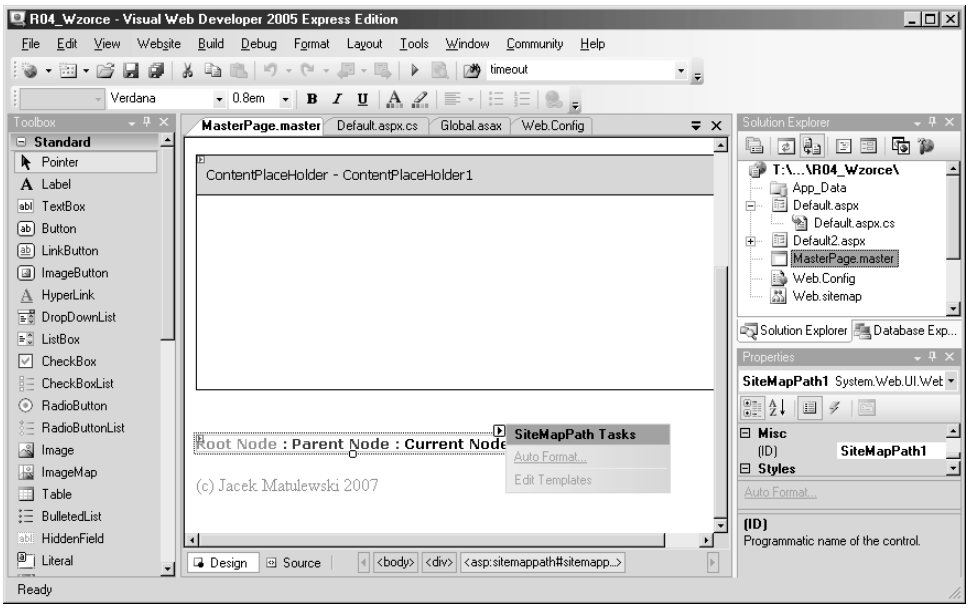
Rysunek 4.4. *Do działania komponentu SiteMapPath konieczna jest obecność pliku Web.sitemap. Zdefiniowane w nim opisy pojawiają się w okienkach podpowiedzi*

3. Zwróćmy uwagę na mały trójkącik widoczny po prawej stronie górnej krawędzi nowego komponentu, jeżeli ten jest zaznaczony. Jeżeli go klikniemy, pojawi się lista typowych zadań dotyczących tego komponentu (rysunek 4.5). W tym przypadku składa się ona z dwóch pozycji *Auto Format* oraz *Edit Templates*. Pierwsze polecenie służy do niemal automatycznego konfigurowania wyglądu komponentu. Wybierzmy np. szablon *Colorful*.

Menu strony

Innym zastosowaniem mapy witryny jest automatycznie tworzone menu i drzewo zawierające strony uwzględnione w tym pliku. Zacznijmy od menu. Dodamy je do wzorca — tym razem ponad komponentem rezerwującym miejsce dla stron.

1. Przechodzimy do widoku projektowania wzorca *MasterPage.master*.
2. Ponad komponentem `ContentPlaceHolder` umieszczamy komponent `Menu` z zakładki *Navigation*.



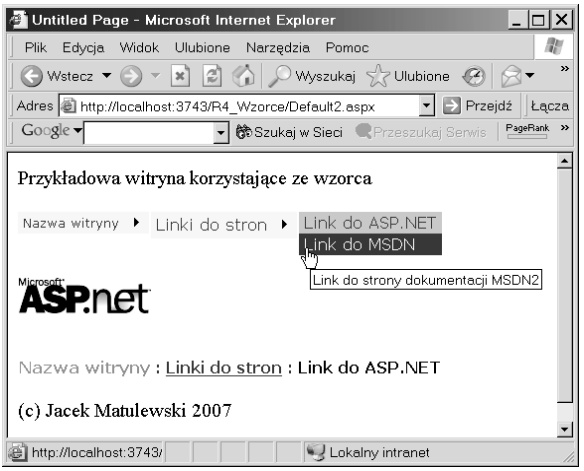
Rysunek 4.5. *Podręczna lista zadań to zbiór najczęściej wykorzystywanych kreatorów związanych z komponentem*

3. W liście podręcznej z rozwijanej listy *Choose Data Source* wybieramy *<New data source…>*.

4. Pojawi się kreator *Data Source Configuration Wizard* pozwalający na wybór źródła danych, na podstawie których utworzone zostanie menu. Może ono zostać zbudowane w oparciu o dowolny plik XML lub nasz gotowy plik *Web.sitemap*. Wybieramy oczywiście tę drugą możliwość, zaznaczając ikonę *Site Map*. Klikamy *OK*. Utworzony zostanie komponent `SiteMapDataSource1`.

5. Nam pozostaje tylko sformatować wygląd menu. Proponuję również tym razem wybrać szablon *Colorful* (rysunek 4.6).

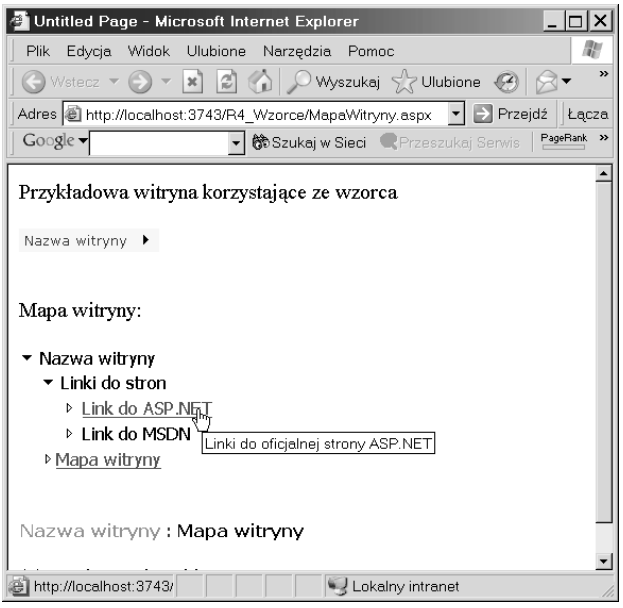
Rysunek 4.6. *Użycie tego samego stylu do formatowania menu i ścieżki pokazującej pozycję w strukturze witryny to zalążek estetycznego i spójnego wizerunku wszystkich stron witryny*



Drzewo pokazujące strukturę stron w witrynie

Na zakładce *Navigation* jest jeszcze jeden komponent, na który też warto zwrócić uwagę. Jest to drzewo `TreeView`, które prezentuje strukturę witryny. Nadaje się bardziej na osobną stronę niż do umieszczenia w nagłówku lub stopce stron.

1. Z menu *File* wybieramy *New File*.
2. W oknie *Add New Item* zaznaczamy *Web Form*, wybierając opcję *Select master page*, i zmieniamy nazwę pliku na *MapaWitryny.aspx*.
3. Klikamy *Add*. W nowym oknie wybieramy wzorzec i klikamy *OK*.
4. Przechodzimy do widoku projektowania nowej strony.
5. Na dostępnym obszarze umieszczamy komponent `TreeView` z zakładki *Navigation*.
6. Postępując identycznie jak w przypadku menu, tworzymy źródło danych korzystające z mapy witryny (niestety, nie można użyć gotowego komponentu `SiteMapDataSource1` widocznego w obszarze wzorca).
7. Formatujemy drzewo, wybierając z podręcznej listy zadań pozycję *Auto Format...*. Proponuję użycie stylu *Arrows 2* (rysunek 4.7).



Rysunek 4.7. *Automatycznie generowana mapa witryny*

8. Nową stronę warto dopisać do mapy witryny, tworząc węzeł równorzędny do *Default.aspx* (por. źródła dołączone do książki).

Poza zwykłą nawigacją, jaką umożliwia komponent `TreeView`, można definiować metody zdarzeniowe związane z kliknięciem różnych pozycji drzewa (zdarzenie `SelectedNodeChanged`).

Dwa słowa na temat kaskadowych arkuszy stylów

Wiemy już, że w projektach ASP.NET możemy w znacznym stopniu, w zasadzie nawet całkowicie, odseparować kod C# od szablonu HTML strony. W ten sposób oddzielony zostaje kod odpowiedzialny za statyczny wygląd stron witryny od metod określających ich dynamikę. Do tych dwóch etapów projektowania dochodzi trzeci, w którym za pomocą kaskadowych arkuszy stylów określamy estetyczny aspekt witryny. Podobnie jak w przypadku kodu C#, także arkusze stylu mogą być całkowicie odseparowane w plikach *.css*, a przez to ich rozwój, podobnie jak kodu C#, może być z łatwością powierzony innym osobom niż rozwój kodu z plików *.aspx*.

W kilku poniższych przykładach przedstawię podstawowe narzędzia służące do budowania kaskadowych arkuszy stylów. Nieco dodatkowych przykładów znajdzie Czytelnik także w rozdziale 10.

Tworzenie arkuszy stylów

Kaskadowe arkusze stylów (ang. *cascade style sheet*) to kolejne obok wzorca narzędzie ujednolicenia stron witryny, a jednocześnie zcentralizowania kontroli nad ich wyglądem. I w tym przypadku wsparcie ze strony VWD jest godne pochwały.

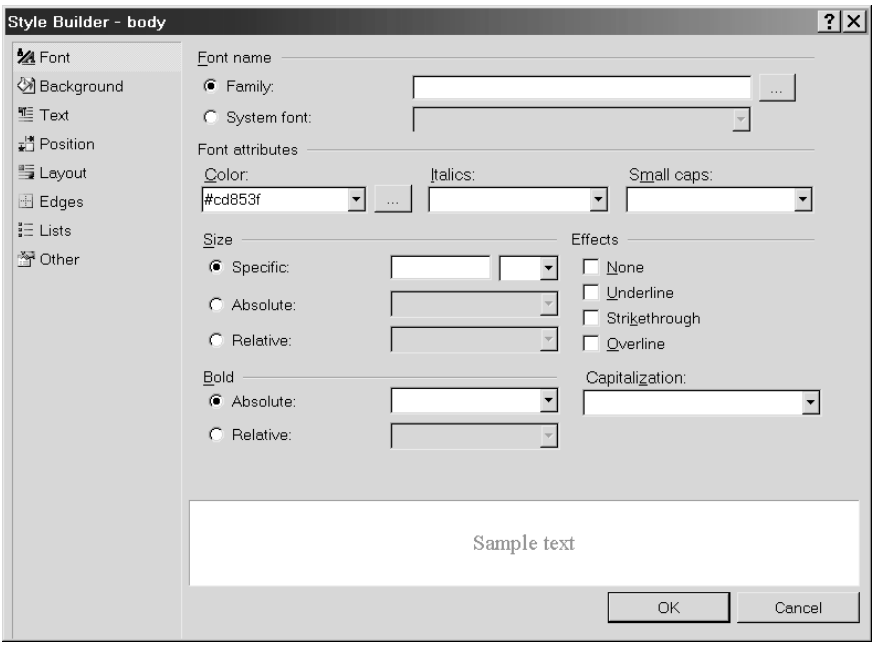
1. Z menu *File* wybieramy *New File…*.
2. W oknie *Add New Item* zaznaczamy pozycję *Style Sheet*, jeżeli odczuwamy taką potrzebę, zmieniamy nazwę pliku i klikamy *OK*.

W edytorze zobaczymy niemal pusty plik, który zawiera jedynie tekst:

```
body {
}
```

To zalążek stylu związanego ze zwykłym tekstem umieszczonym na stronie (tekst między znacznikami `BODY` w kodzie HTML). Na szczęście nie musimy znać się na formacie arkuszy stylów, bo VWD zawiera proste narzędzie pozwalające na ich definiowanie. Zacznijmy od rozbudowania reguły formatowania dotyczącej prostego tekstu.

1. W edytorze ustawiamy kursor (edycji, nie myszy) między nawiasami istniejącej reguły stylu.
2. Klikamy na pasku narzędzi przycisk *Build Style…*.
3. W oknie *Style Builder* (rysunek 4.8) możemy wybrać format i kolor czcionki, tła, list i innych elementów umieszczonych na stronie. My ograniczymy się do sformatowania czcionki, dlatego klikamy pozycję *Font* na liście zakładek widocznej z lewej strony okna.



Rysunek 4.8. *Definiowanie reguły stylu dla znacznika body*

4. W części zatytułowanej *Font attributes* klikamy przycisk z trzema kropkami pozwalający na swobodny wybór koloru:

   a) na zakładce *Named Colors* odnajdujemy kolor *Peru* (jeden z brązowych), który będzie dobrze współgrał ze stylem formatowania wybranym w menu i innych komponentach nawigacyjnych;

   b) odznaczamy pole opcji *Use color names*, aby użyć określenia koloru poprzez składowe zamiast nazwy[1], i klikamy *OK*, aby zamknąć kreator stylu.

5. Następnie korzystając z ikony *Add Style Rule* na pasku narzędzi, tworzony nowy styl dla znacznika `A` (tj. dla umieszczonych na stronie łączy):

   a) tym razem wybierzmy kolor *SaddleBrown* (pamiętajmy o wyłączeniu opcji *Use color names*);

   b) w oknie *Build Style* w części *Effects* zaznaczamy pole *None*.

6. Po tym zdefiniujmy jeszcze jedną regułę formatowania dla `A:hover` (łącze po najechaniu na niego kursorem), w którym kolor ustalamy na *Orange*, a w części *Effects* (por. rysunek 4.8) włączamy opcję *Underline*.

Po tych czynnościach plik kaskadowego arkusza stylów (plik z rozszerzeniem *.css*) powinien wyglądać jak na listingu 4.2:

[1] Bardziej egzotyczne nazwy kolorów mogą być niezrozumiałe dla starszych przeglądarek.

Listing 4.2. *Zawartość pliku kaskadowych arkuszy stylów*

```
body
{
    color: #cd853f;
}
A
{
    color: #8b4513;
    text-decoration: none;
}
A:hover
{
    color: #ffa500;
    text-decoration: underline;
}
```

Stosowanie arkuszy stylów

Czas, aby arkusz wykorzystać do formatowania stron naszej przykładowej witryny:

1. Przejdźmy na stronę *Default.aspx*.
2. W widoku projektowania dodajmy do niej prosty tekst (wpisując go w widoku projektowania w polu `Content`) oraz komponent `HyperLink` ze skonfigurowaną właściwością `NavigateUrl` i etykietą (właściwość `Text`).
3. Teraz przejdźmy do widoku projektowania pliku wzorca *MastepPage.master*.
4. Przeciągnijmy z okna projektu (*Solution Explorer*) utworzony plik *.css*. Do kodu strony dodany zostanie element `<link href="StyleSheet.css" rel="stylesheet" type="text/css" />`, dzięki któremu wzorzec i wszystkie używające go strony będą korzystały z arkusza i zdefiniowanych w nim stylów.

W podglądzie wzorca i podglądzie stron, które z niego korzystają, zobaczymy zmianę — tekst zmieni kolor na brązowy, łącza na jasnobrązowy. Ponadto łącza pozbawione zostały podkreślenia. Jeżeli obejrzymy stronę w przeglądarce, to zobaczymy, że kolor łączy zmienia się na pomarańczowy, jeżeli najechać na nie myszą, oraz że pojawia się wówczas pod nimi podkreślenie.


Wskazówka

Jeżeli witryna nie ma wzorca, arkusz należy dodać do każdej strony osobno. To samo dotyczy stron w naszej witrynie, które nie korzystają ze wzorca.

Można również edytować indywidualny styl poszczególnych komponentów na stronach. W ich menu kontekstowym znajduje się pozycja *Style...*, która uruchamia okno *Style Builder* z rysunku 4.8 lub, jeżeli chcemy użyć istniejących klas stylu, w oknie *Properties* odnajdujemy właściwość `CssClass` i tam wpisujemy nazwę klasy zdefiniowanej w arkuszu stylu. Klasy można definiować w pliku *.css*, dodając regułę i zaznaczając opcję *Class name* (zob. rozdział 10.).