Temat: Lokalna konfiguracja serwera WWW – plik .htaccess

Określona konfiguracja serwera WWW na którym prowadzony jest hosting naszej aplikacji internetowej jest dostępna z poziomu pliku konfiguracyjnego **httpd.conf**. Dostęp do niego ma jednak tylko administrator serwera WWW. Istnieje jednak zmiana wielu parametrów serwera w obrębie katalogu głównego naszego hostingu i wszystkich jego podkatalogów. Umożliwia to specjalny plik konfiguracyjny o nazwie **.htaccess**.

Plik .htaccess jest charakterystycznym elementem serwera WWW, który pozwala skonfigurować wiele jego parametrów. Plik ten działa w obrębie katalogu, w którym został umieszczony oraz w podkatalogach (chyba, że w podkatalogu umieszczono kolejny plik .htaccess). W ramach jednego konta może funkcjonować wiele niezależnych plików .htaccess, z których każdy może definiować inną akcję.

Kiedy można korzystać z .htaccess?

Mechanizm **.htaccess** jest domyślnie włączony na serwerze Apache większości hostingów w sieci. Za jego włączenie odpowiada dyrektywa **AllowOverride All** w pliku konfiguryjnym **httpd.conf** serwera. Na przykład w naszej pracowni będzie to wpis:

```
    ServerName localhost:80
    ...
    DocumentRoot "C:/xampp/htdocs"
    <Directory "C:/xampp/htdocs">
    ...
    AllowOverride All
    Order allow,deny
    Allow from all
</Directory>
</VirtualHost>
```

Administrator serwera może jednak z jakiejś przyczyny w pliku konfiguracyjnym Apache'a zamiast dyrektywy **AllowOverride All** wstawić dyrektywę:

```
AllowOverride None
```

Wówczas mechanizm **.htaccess** zostanie dla tego katalogu (i jego podkatalogów) wyłączony.

Co należy wiedzieć o pliku .htaccess na początek?

Zbiór **.htaccess** powinien mieć uprawnienia 644 (nadane przez polecenie **chmod**). Pozwoli to na dostęp do pliku przez serwer, ale uniemożliwi jego zmianę z poziomu przeglądarki (chodzi rzecz jasna o systemy Linux, Unix, FreeBSD itd.):

```
chmod 644 .htaccess
```

Zanim przystąpimy do pierwszych zapisów w pliku, warto wspomnieć, że komentarze w pliku **.htaccess** wstawia się poprzedzając je znakiem **#**. Plik **.htaccess** i współpracujące z nimk pliki (np. **.htpasswd**) powinny być zapisane w zwykłym formacie ANSI.

Ograniczenie dostępu do witryny

Katalog główny witryny (i każdy inny) funkcjonujący w ramach serwera WWW może zostać całkowicie zablokowany za pomocą pliku **.htaccess** z zwartością:

```
Order allow,deny
Deny from all
```

Wówczas serwer WWW automatycznie wygeneruje błąd 403. Jeśli chcielibyśmy udostępnić naszą stronę tylko osobom o wybranym adresie IP, to wystarczy zamienić polecenia w pliku na:

```
Order allow,deny
Allow from adresIP
```

Na przykład:

```
Order allow,deny
Allow from 127.0.0.1
```

Banowanie określonych IP, czyli ograniczanie dostępu do całej strony robimy tak:

```
Order Allow,Deny
Allow from all
Deny from adresIP1
Deny from adresIP2
...
```

Dopuszczalny jest zapis w postaci:

```
Deny from 200.200.100
```

co oznaca, że wszystkie adresy zaczynające się od podanych wartości będą banowane. Możliwy jest także zapis:

```
Deny from nazaDomeny
```

Na przykład:

```
Deny from .zst.edu.pl
```

Ćwiczenie 81.

Przetestuj kolejno opisane wyżej polecenia w pliku ***.htaccess*** *na witrynie z archiwum* ***cwiczenie81.zip****. Plik* ***.htaccess*** *umieść w katalogu głównym localhosta (htdocs).*

Ograniczenie dostępu do witryny poprzez hasło

Aby ograniczyć dostęp do katalogu/strony WWW tylko dla osób znających hasło, należy w zabezpieczanym katalogu umieścić plik **.htaccess** zwierający:

```
AuthName "Tekst komunikatu "
AuthType Basic
AuthUserFile katalog/.htpasswd  # np. C:\xampp\htdocs\4tiX\haslo\.htpasswd
Require valid-user
```

Gdzie **AuthName** może być dowolnym tekstem, który zostanie wyświetlony przy próbie wywołania zabezpieczonego katalogu. Natomiast w linii **AuthUserFile** należy podać pełną ścieżkę do pliku **.htpasswd**.

Następną wymaganą czynnością jest utworzenie pliku **.htpasswd**, który będzie zawierał nazwy użytkowników oraz ich hasła dostępu. Zawartość pliku **.htpasswd** tworzy się według poniższego schematu:

```
nazwaUzytkownika:hasłoDostępu
```

Gdzie **hasłoDostępu** jest odpowiednio zakodowane algorytmem **crypt()**.

Plik .htpasswd powinien znajdować się poza katalogiem głównym, czyli tam gdzie zwykły użytkownik nie będzie miał dostępu. Hasło do pliku .htpasswd można zakodować pod http://www.htaccessredirect.net/htpasswd-generator.

Ćwiczenie 82.

Zabezpiecz hasłem dostęp do witryny z archiwum ***cwiczenie82.zip****. Plik* ***.htaccess*** *umieść w katalogu głównym witryny, zaś plik z hasłem w katalogu* ***dostep****.*

Definiowanie nietypowej strony startowej

Operacja definiowania nowego pliku startowego polega na wpisaniu odpowiedniego polecenia do pliku **.htaccess**. Składnia polecenia:

DirectoryIndex nazwaStronyX nazwaStronyY

Na przykład:

DirectoryIndex start.html index.php

Ćwiczenie 83.

*Na przykładzie witryny z archiwum **cwiczenie83.zip** ustaw jako stronę startową witryny stronę **start.php**.*

Definiowanie strony błędu

Zdarza się, że przeglądając strony WWW trafiasz na nieistniejące dokumenty (np. klikając na uszkodzony lub nieaktualny link). Wyświetlany jest wtedy standardowy komunikat błędu nr 404 – informujący o braku strony. Polecenie:

ErrorDocument 404 adresURL

Na przykład:

ErrorDocument 404 http://localhost/4tiX/blad/404.html

Ćwiczenie 84.

*Na przykładzie witryny z archiwum **cwiczenie84.zip** sprawdź obsługę błędu 404.*

Ustawienia listowania zawartości katalogu

Listowanie to operacja, która powoduje wyświetlenie zawartości wybranego katalogu i jego podkatalogów. W przypadku jeśli listowanie będzie wyłączone, to podczas próby wywołania zawartości katalogu zostanie wyświetlony błąd 403. I tak polecenie:

Options +Indexes

włącza listowanie katalogu (i podkatalogów), zaś polecenie wyłączające ma postać:

Options -Indexes

Ćwiczenie 85.

Na przykładzie witryny z poprzedniego ćwiczenia sprawdź polecenia listowania zawartości katalogu strony WWW.

Moduł Mod_rewrite

Mod_rewrite to wyspecjalizowany dodatek serwera Apache umożliwiający wykonanie "przezroczystego" dla użytkownika przekierowania na inny adres URL. Ma wiele zastosowań, począwszy od prostej zmiany nieaktualnych adresów na nowe, aż do zmiany linków zawierających zmienne wysyłane w standardzie GET na prostsze, dzięki czemu są one znacznie przyjaźniejsze zarówno osobom odwiedzającym serwis, jak również i wyszukiwarkom internetowym. Moduł zostaje włączony przez dyrektywę:

```
RewriteEngine on
```

1. Proste przekierowania

Załóżmy, że zmieniliśmy nazwę pliku z **index.php** na **newindex.php**. Ponieważ serwer WWW obsługujący witrynę jest ukierunkowany na plik **index.php**, domyślnie to stara wersja pliku startowego bylaby wczytywana przez serwer WWW. Jednak tworząc odpowiednią regułę pozwolimy na korzystanie także ze starego adresu:

```
RewriteEngine On
RewriteRule ^index\.php$ newindex.php
```

W przekierowaniu, w dyrektywie **RewriteRules** użyto trzech znaków specjalnych:

- znaku **^** oznaczającego początek ścieżki do domyślnego pliku startowego w katalogu w którym znajduje się plik **.htaccess**;
- znaku dolara **$**, oznaczającego koniec analizowanej ścieżki pliku;
- znaku ukośnika **** mówiącego, że następujący po nim znak nie jest znakiem specjalnym wyrażenia regularnego, a normalnym znakiem (w tym przypadku separatora nazy pliku od jego rozszerzenia).

Ćwiczenie 86.

*Na przykładzie witryny z archiwum **cwiczenie86.zip** utwórz przekierowanie z pliku **index.php** na **newindex.php**.*

2. Przekierowanie domeny na inny katalog

Załóżmy, że dysponujemy jakąś subdomeną i musimy przekierować ją na odpowiedni katalog na serwerze, inny niż ten wynikający z konfiguracji domeny w jej panelu administracyjnym czy zapisany w pliku konfiguracyjnym Apache. Kod takiego przekierowania może być nastepujący:

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^(www\.)?domena$ [NC]
```

```
RewriteCond %{REQUEST_FILENAME} !/katalog/
RewriteRule ^(.*)$ /katalog/$1 [L]
```

Na przykład:

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^(www\.)?localhost$ [NC]
RewriteCond %{REQUEST_FILENAME} !/nowy/
RewriteRule ^(.*)$ /nowy/$1 [L]
```

Gdzie użyte zmienne i stałe oznaczają odpowiednio:

- **HTTP_HOST** – adres hosta, który jest wywoływany,
- **REQUEST_FILNAME** – pełna ścieżka do pliku,
- **%** – poprzedza wartość zmiennej;
- **?** – zero lub jedno wystąpienie;
- **!** – negacja
- **[NC]** – ignorowanie wielkości liter;
- **[L]** – dyrektywa jest wczytana jako ostatnia (kolejne po niej są ignorowane).

Ćwiczenie 87.

*Na przykładzie witryny z archiwum **cwiczenie87.zip** w pliku **.htaccess** w katalogu głównym witryny ustaw przekierowanie domeny na katalog o nazwie **nowy**. Pozostaw w pliku dyrektywę obsługi błędu 404.*

3. Przekierowanie 301

Przekierowanie 301 jest to sposób na przekierowanie użytkownika z jednego adresu URL na inny (np. znajdujący się na zewnętrznym serwerze lub znajdujący się na tym samym serwerze, ale w innej lokalizacji). Jest to najlepsze rozwiązanie z punktu widzenia pozycjonowania danej strony WWW w wyszukiwarkach. Kod 301 oznacza „Moved Permanently", czyli trwale przeniesiony. Aby wykonać przekierowanie typu 301 należy w katalogu (do którego obecnie przekierowana jest domena) umieścić plik **.htaccess** o składni wzorowanej na poniższej:

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^(www\.)?localhost [NC]
RewriteRule (.*) http://www.zst.edu.pl/$1 [R=301,L]
```

Ćwiczenie 88.

*Na przykładzie witryny z archiwum **cwiczenie88.zip** w pliku **.htaccess** ustaw przekierowanie dowolnej domeny z localhost na domenę **www.zst.edu.pl**.*