

Zaawansowane metody programowania

Formularze

Znaczniki HTML

Formularz : struktura formularza

```
<form>
|-- <???>
    |-- <fieldset> zbiór pól
    |   |-- <legend> opis zbioru pól
    |   |-- <input> pole danych
    |-- <fieldset> zbiór pól
    |   |-- <legend> opis zbioru pól
    |   |-- <input> pole danych
    |-- <input type="submit"> przycisk
```

Wymagany jest jeden element blokowy (np. *div*, *table*, *fieldset*) obejmujący całą zawartość formularza. Aby formularz można było wysłać do serwera, niezbędny jest element typu *submit*. Formularz powinien też zawierać pola danych, np. typu *input*.

Znaczniki HTML

Formularz : struktura formularza

`<form>`

| -- `<fieldset>` zbiór pól

| -- `<legend>` opis zbioru pól

| -- `<input type="text">` pole edycyjne

| -- `<input type="submit">` przycisk

W najprostszym przypadku wystarczy jeden element *fieldset*, opcjonalnie uzupełniony przez *legend*, pola danych (np. *input*) oraz przycisk wysyłania formularza *submit*.

Znaczniki HTML

Formularz: form

- Wstawia do dokumentu formularz

```
<form method="POST" action="/adres/url/">
```

...

```
</form>
```

- Znacznik zamykający – obowiązkowy
- Atrybuty:
 - id, class, style, title,
 - method – sposób wysłania do serwera (POST albo GET),
 - action – URL skryptu, do którego zostanie wysłany formularz,
 - enctype – rodzaj zawartości formularza,
 - accept – lista typów zawartości dla dołączanych plików
 - onsubmit – skrypt (np. JavaScript) wykonywany przy wysyłaniu
 - onreset – skrypt wykonywany po zresetowaniu

Znaczniki HTML

Formularz: atrybuty action i method

- action – formularze typowo są obsługiwane po stronie serwera, dlatego w atrybucie *action* trzeba podać URL skryptu (lub innego typu dynamicznej strony WWW), do którego zostanie wysłany formularz;
- method – sposób wysłania formularza do serwera; formularz może być wysłany przez HTTP przy pomocy żądania POST albo GET – różnica polega na sposobie przekazywania parametrów:
 - POST – w nagłówkach HTTP
 - GET – w adresie WWW, zatem są widoczne dla użytkownika, np. .../forum?login=admin&passwd=kernel21
a także zapisywane w historii przeglądarki (bezpieczeństwo!)

Wybór metody zależy od sytuacji: obie mają wady i zalety.

Znaczniki HTML

Formularz: atrybuty enctype i accept

- enctype – rodzaj zawartości formularza
 - jeżeli formularz nie zawiera dołączonych plików:
"application/x-www-form-urlencoded"
(jest to wartość domyślna – można pominąć)
 - jeżeli formularz zawiera dołączone pliki:
"multipart/form-data"
- accept – dotyczy plików dołączanych do formularza; jest to lista typów zawartości, jakie są akceptowane przez docelowy skrypt (wskazany atrybutem *action*) – przeglądarka może wykorzystać ten atrybut podczas wyświetlania okna wyboru pliku

Znaczniki HTML

Zbiór pól: fieldset

- Reprezentuje zbiór powiązanych ze sobą elementów formularza; wyświetlany jako owalna ramka, z opcjonalnym podpisem w postaci elementu legend:

```
<form>  
  <fieldset>  
    <legend>Tytuł</legend>  
    <input ...>  
  </fieldset>  
</form>
```

- Znacznik zamykający – obowiązkowy
- Atrybuty:
id, class, style, title,

Znaczniki HTML

Opis zbioru pól: legend

- Wyświetla opis dla elementu fieldset :

```
<fieldset>
  <legend>Tytuł</legend>
  ...
</fieldset>
```

- Znacznik zamykający – obowiązkowy
- Atrybuty:
 - id, class, style, title,
 - accesskey – klawisz szybkiego dostępu (skrót klawiaturowy)

Znaczniki HTML

Pola danych formularzy HTML:

- Pola tekstowe, jedno- i wielolinijkowe
- Przyciski
 - przycisk *submit* – wysyła zawartość formularza do serwera,
 - przycisk *reset* – przywraca wartości początkowe danych,
 - inne przyciski – działanie zależne od skojarzonego skryptu, wykonywanego po stronie przeglądarki
- Pola wyboru
- Przyciski radiowe
- Listy rozwijane
- Pola wyboru pliku
- Pola ukryte
- Obiekty osadzone

Znaczniki HTML

Pole danych: input

- Reprezentuje pole danych:

`<input type="..." name="...">`

- Znacznik zamykający – zabroniony

- Atrybuty:

id, class, style, title,

type – typ pola (np. text, checkbox, radio, submit, ...)

name – nazwa, pod jaką zawartość będzie wysłana do serwera

value – wartość początkowa (znaczenie zależnie od typu)

checked – pole zaznaczone (dla typu checkbox i radio)

disabled – niedostępne dla użytkownika

readonly – pole tylko do odczytu

required – pole wymagane

size – rozmiar pola (znaczenie zależnie od typu)

maxlength – maksymalna długość tekstu (dla text i password)

...

Znaczniki HTML

Pole danych: input

- Reprezentuje pole danych:

`<input type="..." name="...">`

- Znacznik zamykający – opcjonalny

- Atrybuty:

...

src – URL pliku z obrazkiem (dla typu image)

tabindex – kolejność aktywacji klawiszem tabulatora

accesskey – klawisz szybkiego dostępu (skrót klawiaturowy

– niestety sposób aktywacji zależy od systemu operacyjnego)

onfocus – skrypt wykonywany po uzyskaniu "fokusu"

onblur – skrypt wykonywany po utracie "fokusu"

onselect – skrypt wykonywany zmianie zaznaczenia tekstu

onchange – skrypt wykonywany modyfikacji tekstu

Znaczniki HTML

Pole danych: atrybuty type

- Określa typ kontrolki, jaka ma zostać utworzona:
 - text – pole edycyjne, jednolinijkowe pole tekstowe
 - password – jak *text*, ale wpisywane znaki są maskowane
 - checkbox – pole wyboru (opcje niezależne od siebie)
 - radio – przycisk radiowy (opcje nawzajem wykluczające się)
 - submit – przycisk wysyłania formularza do serwera
 - reset – przycisk przywracający domyślne wartości kontrolek
 - file – kontrolka wyboru pliku (pole edycyjne + przycisk)
 - hidden – pole ukryte
 - image – graficzny odpowiednik *submit*
 - button – przycisk skryptu wykonywanego przez przeglądarkę

Możliwość stosowania oraz znaczenie pozostałych atrybutów częściowo zależy od typu kontrolki

- HTML5 wprowadził wiele specjalnych kontrolek tekstowych (email, date, time, ...) – implementacja zależna od przeglądarki

Znaczniki HTML

Pole danych: atrybuty name i value

- name – dla kontrolek, które wysyłają dane do serwera (text, password, submit, checkbox, radio, file, hidden) – określa nazwę, pod jaką zawartość kontrolki (*value*) zostanie wysłana. Brak atrybutu name powoduje, że zawartość kontrolki w ogóle nie jest wysłana;
- value – znaczenie zależne od typu kontrolki:
 - text, password – wartość początkowa wyświetlana w kontrolce (przywracana po kliknięciu przycisku reset)
 - checkbox, radio – wartość wysłana o ile kontrolka jest zaznaczona (prze braku value wysyłana jest wart. "on"); jeżeli kontrolka nie jest zaznaczona, nic nie jest wysyłane
 - hidden – wartość niewidoczna, wysyłana do serwera
 - submit, reset, button – opis wyświetlany na przycisku

Znaczniki HTML

Pole danych: atrybuty checked, disabled, readonly, required

- checked – tylko kontrolki typu checkbox oraz radio – jeżeli występuje, oznacza, że początkowo kontrolka jest zaznaczona;
 - checkbox – każdy może być zaznaczony lub nie, niezależnie od pozostałych
 - radio – powinien być zaznaczony dokładnie jeden z każdej grupy przycisków (grupy są określone przez atr. name)
- disabled – jeżeli występuje, oznacza że kontrolka jest niedostępna dla użytkownika; stan ten można zmienić za pomocą skryptu wykonywanego po stronie przeglądarki; Zawartość kontrolki niedostępnej nie jest przesyłana na serwer
- readonly – jeżeli występuje, oznacza że kontrolka nie może być zmieniana przez użytkownika; Zawartość kontrolki jest przesyłana na serwer
- required – jeżeli występuje, to wypełnienie kontrolki jest wymagane, bez tego przeglądarka nie wyśle formularza

Znaczniki HTML

Pole danych: atrybuty size i maxlength

- size – tylko kontrolki typu text, password, file – szerokość pola przeznaczonego na wpisanie tekstu (wyrażona w znakach)
- maxlength – tylko kontrolki typu text, password – maksymalna ilość znaków, jakie można wpisać; Może mieć wartość większą, niż size – wówczas tekst będzie przewijany podczas wpisywania

Znaczniki HTML

Pole danych: atrybuty tabindex i accesskey

- `tabindex` – kolejność aktywacji klawiszem tabulatora; domyślnie odpowiada kolejności występowania kontrolek w dokumencie;
- `accesskey` – klawisz szybkiego dostępu (skrót klawiaturowy); sposób aktywacji zależy od platformy – np. w Windows trzeba użyć kombinacji `[Alt]+skrót`;
działanie jest zależne od typu kontrolki:
 - `text`, `password`, `file` – aktywacja
 - `checkbox`, `radio` – zmiana zaznaczenia

Znaczenie `accesskey` jest ograniczone ze względu na trudność we wskazaniu skrótu (podkreślenie może być mylone z łączem)

Znaczniki HTML

Etykieta: label

- Reprezentuje etykietę przypisaną do pola danych formularza:

```
<input type="..." id="ID" name="...">  
<label for="ID">Opis komponentu</label>
```

- Znacznik zamykający – wymagany
- Atrybuty:
 - id, class, style, title,
 - for – id elementu opisywanego przez etykietę
 - accesskey – klawisz szybkiego dostępu (skrót klawiaturowy)
 - onfocus – skrypt wykonywany po uzyskaniu "fokusu"
 - onblur – skrypt wykonywany po utracie "fokusu"
- Działanie: aktywacja etykiety (skrót klawiaturowy lub kliknięcie) jest równoważne aktywacji pola danych;
W przypadku kontroltek typu checkbox i radio – obowiązkowe!

Znaczniki HTML

Pole edycyjne: input type="text"

- Jednolinijkowe pole edycyjne do wpisania tekstu:

```
<input type="text" name="t1"  
value="tu wpisz..." size="25" maxlength="30" >
```

- Ważniejsze atrybuty:
 - id – identyfikator, kojarzący pole z etykietą,
 - name – nazwa, pod jaką zawartość będzie wysłana do serwera
 - value – początkowa wartość
 - size – rozmiar pola
 - maxlength – maksymalna długość wpisanego tekstu
- Etykieta - opcjonalnie

Znaczniki HTML

Pole edycyjne hasła: input type="password"

- Jednolinijkowe pole edycyjne do wpisania hasła:

```
<input type="password" name="p1"
      size="25" maxlength="30" >
```

- Ważniejsze atrybuty:
 - id – identyfikator, kojarzący pole z etykietą,
 - name – nazwa, pod jaką zawartość będzie wysłana do serwera
 - size – rozmiar pola
 - maxlength – maksymalna długość wpisanego tekstu
- Etykieta - opcjonalnie

Znaczniki HTML

Pole wyboru: input type="checkbox"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="checkbox" id="ch1" name="ch1"
      value="1" checked >
```

```
<label for="ch1">Opis opcji 1</label>
```

```
<input type="checkbox" id="ch2" name="ch2"
      value="1">
```

```
<label for="ch2">Opis opcji 2</label>
```

- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera

value – wartość wysyłana, jeżeli pole jest zaznaczone

(domyślnie "on"; jeżeli nie zaznaczone, nic nie jest wysyłane)

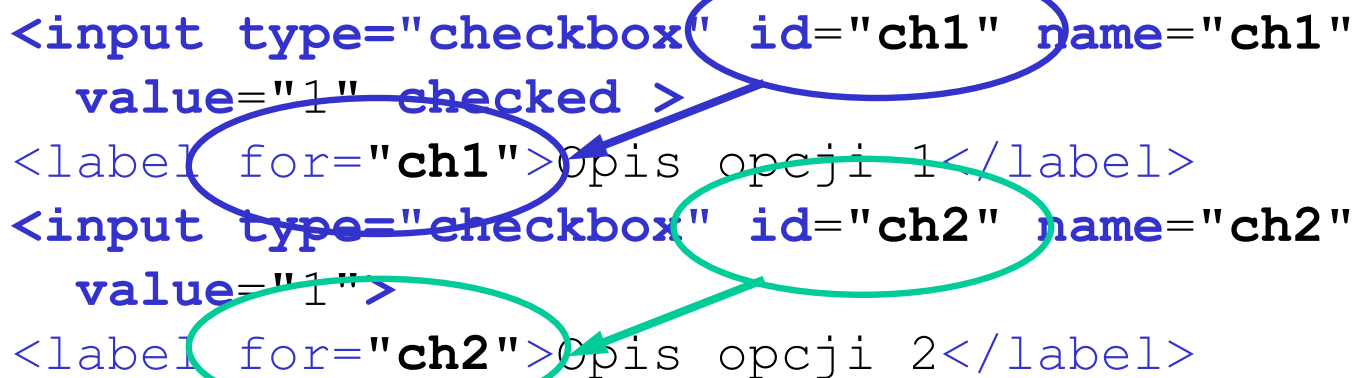
checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Pole wyboru: input type="checkbox"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="checkbox" id="ch1" name="ch1"
      value="1" checked >
<label for="ch1">Opis opcji 1</label>
<input type="checkbox" id="ch2" name="ch2"
      value="1">
<label for="ch2">Opis opcji 2</label>
```



- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera

value – wartość wysyłana, jeżeli pole jest zaznaczone

(domyślnie "on"; jeżeli nie zaznaczone, nic nie jest wysyłane)

checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Pole wyboru: input type="checkbox"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="checkbox" id="ch1" name="ch1" value="1" checked >
```

```
<label for="ch1">Opis opcji 1</label>
```

```
<input type="checkbox" id="ch2" name="ch2" value="1">
```

```
<label for="ch2">Opis opcji 2</label>
```

- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera

value – wartość wysyłana, jeżeli pole jest zaznaczone

(domyślnie "on"; jeżeli nie zaznaczone, nic nie jest wysyłane)

checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Przycisk radiowy: input type="radio"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="radio" id="r1" name="rg1"
      value="1" checked >
```

```
<label for="r1">Opis opcji 1</label>
```

```
<input type="radio" id="r2" name="rg1"
      value="2">
```

```
<label for="r2">Opis opcji 2</label>
```

- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera;

ważne! atrybut name dla grupy opcji musi być ten sam

value – wartość wysyłana, jeżeli pole jest zaznaczone

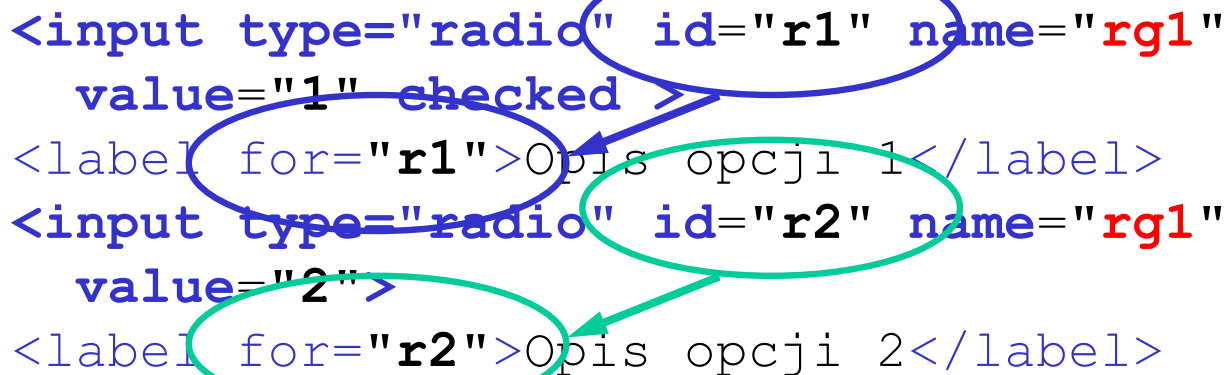
checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Przycisk radiowy: input type="radio"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="radio" id="r1" name="rg1"
      value="1" checked >
<label for="r1">Opis opcji 1</label>
<input type="radio" id="r2" name="rg1"
      value="2">
<label for="r2">Opis opcji 2</label>
```



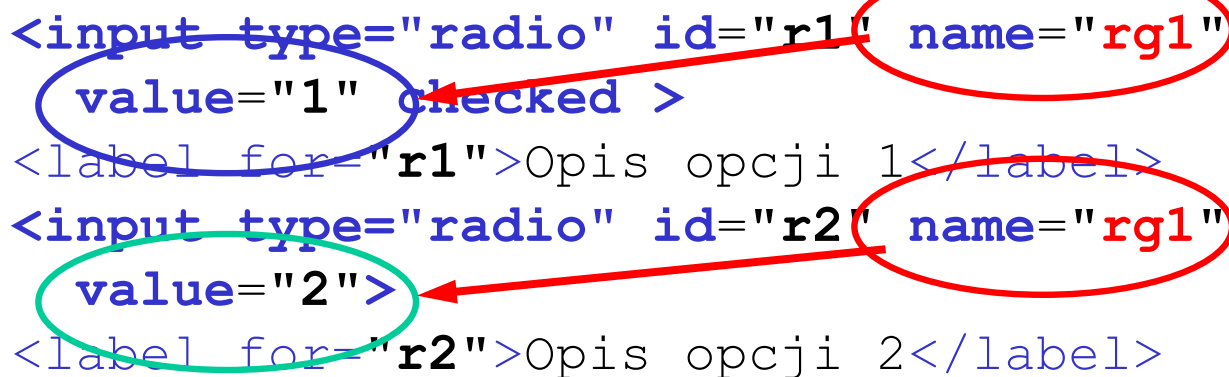
- Ważniejsze atrybuty:
 - id – identyfikator, kojarzący pole z etykietą,
 - name – nazwa, pod jaką zawartość będzie wysłana do serwera;
ważne! atrybut name dla grupy opcji musi być ten sam
 - value – wartość wysyłana, jeżeli pole jest zaznaczone
 - checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Przycisk radiowy: input type="radio"

- Pole edycyjne opcji niezależnych od siebie:

```
<input type="radio" id="r1" name="rg1" value="1" checked >  
<label for="r1">Opis opcji 1</label>  
<input type="radio" id="r2" name="rg1" value="2">  
<label for="r2">Opis opcji 2</label>
```



- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera;

ważne! atrybut name dla grupy opcji musi być ten sam

value – wartość wysyłana, jeżeli pole jest zaznaczone

checked – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Pole wysyłania pliku: input type="file"

- Pole edycyjne i przycisk wyboru pliku:

```
<input type="file" name="f1"  
size="25">
```

- Ważniejsze atrybuty:

id – identyfikator, kojarzący pole z etykietą,

name – nazwa, pod jaką zawartość będzie wysłana do serwera

size – rozmiar pola

- W zwykły sposób (para name - value) wysyłana jest nazwa pliku, sam plik jako załącznik żądania HTTP;

Przesyłanie pliku jest możliwe tylko metodą POST

Znaczniki HTML

Przycisk submit: `input type="submit"`

- Przycisk wysyłania formularza:

```
<input type="submit" name="s1" value="Wyślij">
```

- Ważniejsze atrybuty:

value – napis wyświetlany na przycisku w przeglądarce

name – jeżeli jest, to jest też wysyłana na serwer – można utworzyć kilka przycisków i sprawdzić który został użyty

Przycisk reset: `input type="reset"`

- Przycisk czyszczenia formularza:

```
<input type="reset" value="Wyczyść">
```

- Ważniejsze atrybuty:

value – napis wyświetlany na przycisku w przeglądarce

Znaczniki HTML

Pole tekstowe: textarea

- Tworzy wielolinijkowe pole tekstowe:

```
<textarea rows="..." cols="..." name="..."> ...  
</textarea>
```

- Znacznik zamykający – obowiązkowy
- Atrybuty:
 - id, class, style, title,
 - name – nazwa, pod jaką zawartość będzie wysłana do serwera
 - rows, cols – liczba widocznych wierszy i kolumn
 - disabled – niedostępne dla użytkownika
 - readonly – pole tylko do odczytu
 - onfocus – skrypt wykonywany po uzyskaniu "fokusu"
 - onblur – skrypt wykonywany po utracie "fokusu"
 - onselect – skrypt wykonywany zmianie zaznaczenia tekstu
 - onchange – skrypt wykonywany po modyfikacji zaznaczenia

Znaczniki HTML

Lista wyboru: struktura listy

```
<select>
| -- <option> pozycja listy
| -- <option> pozycja listy
| -- <optgroup> nazwa grupy pozycji
|     | -- <option>
|     | -- <option>
| -- <optgroup>
|     | -- <option>
|     | -- <option>
```

Lista wyboru może być wyświetlana jako lista zwykła lub lista rozwijana. Opcjonalne elementy *optgroup* tworzą hierarchiczne menu (jednak nie mogą być zagnieżdżone); nadają nazwę grupom pozycji listy, lecz same nie mogą być wybierane.

Znaczniki HTML

Lista wyboru: select

- Tworzy zwykłą lub rozwijaną listę wyboru:

```
<select size="..." name="..."> ...  
</select>
```

- Znacznik zamykający – obowiązkowy
- Atrybuty:
 - id, class, style, title,
 - name – nazwa, pod jaką zawartość będzie wysłana do serwera
 - size – rozmiar pola (znaczenie zależnie od typu)
 - multiple – możliwość wyboru wielu pozycji na liście
 - disabled – niedostępne dla użytkownika
 - tabindex – kolejność aktywacji klawiszem tabulatora
 - onfocus – skrypt wykonywany po uzyskaniu "fokusu"
 - onblur – skrypt wykonywany po utracie "fokusu"
 - onchange – skrypt wykonywany po modyfikacji zaznaczenia

Znaczniki HTML

Lista wyboru: atrybuty name, size, multiple

- name – nazwa, pod jaką zawartość kontrolki zostanie wysłana do serwera (wysyłana jest wartość określona przez atrybut *value* wybranego elementu *option*)
- multiple – jeżeli występuje, oznacza możliwość zaznaczenia więcej niż jednej pozycji na liście (np. w systemie Windows trzeba klikając wciskać klawisz [CTRL]). Lista wielokrotnego wyboru jest zawsze wyświetlana jako zwykła.
- size – określa sposób wyświetlania i rozmiar listy;
Jeżeli nie występuje atrybut *multiple* i rozmiar ma wartość 1 (wartość domyślna), to wyświetlana jest lista rozwijana;
Z kolei jeżeli rozmiar jest większy niż 1, to wyświetlana jest lista zwykła, zaś atrybut *size* określa liczbę widocznych na raz linijek

Znaczniki HTML

Element listy: option

- Reprezentuje pozycję na liście wyboru:

```
<option value="..." selected>...</option>
```

- Znacznik zamykający – opcjonalny
- Atrybuty:
 - id, class, style, title,
 - value – wartość wysłana do serwera, o ile pozycja jest wybrana
 - selected – pozycja początkowo oznaczona jako wybrana
 - disabled – pozycja listy niedostępna dla użytkownika

Znaczniki HTML

Grupa elementów listy: optgroup

- Umożliwia tworzenie hierarchicznego menu:

```
<optgroup label="...">  
  <option value="..." selected>...  
</optgroup>
```

- Znacznik zamykający – wymagany
- Atrybuty:
 - id, class, style, title,
 - label – etykieta dla grupy opcji
 - disabled – grupa niedostępna dla użytkownika

Znaczniki HTML

Lista wyboru: przykład 1

- Lista rozwijana

```
<select size="1" name="li1">  
  <option value="1" selected>opcja 1  
  <option value="2">opcja 2  
  <option value="3">opcja 3  
  <option value="4">opcja 4  
</select>
```

- Ważniejsze atrybuty:

name – nazwa, pod jaką zawartość będzie wysłana do serwera;
value – wartość wysyłana, jeżeli pole jest zaznaczone
selected – jeżeli wystąpi, pole jest zaznaczone

Znaczniki HTML

Lista wyboru: przykład 2

- Lista przewijana, z układem hierarchicznym

```
<select size="6" name="li2">  
  <optgroup label="grupa 1">  
    <option value="1a" selected>opcja 1a  
  </optgroup>  
  <optgroup label="grupa 2">  
    <option value="2a">opcja 2a  
  </optgroup>  
</select>
```

- Ważniejsze atrybuty:

name – nazwa, pod jaką zawartość będzie wysłana do serwera;

value – wartość wysyłana, jeżeli pole jest zaznaczone

selected – jeżeli wystąpi, pole jest zaznaczone

Formularze

Sposób przesyłania danych formularza do serwera zależy od wybranej metody HTTP:

- Metoda GET

```
<form method="GET" action="plik.php"  
  enctype="application/x-www-form-urlencoded">  
  ...  
</form>
```

- Metoda POST

```
<form method="POST" action="plik.php"  
  enctype="multipart/form-data">  
  ...  
</form>
```

Formularze

Metoda GET

- Działanie

dane z formularza są przesyłane w żądaniu HTTP:

```
GET /action.php?name=value&name=value HTTP/1.1  
Host: www.sth.com
```

Przeglądarka wyświetla dane formularza w pasku adresu:

[www.sth.com/action?**name=value&name=value**](http://www.sth.com/action?name=value&name=value)

- Cechy:

- dane żądania są widoczne i zapisywane w historii/ulubionych
- spacje i znaki spoza ASCII są kodowane ("+", "%HH")
- nie można przesyłać plików na serwer

Formularze

Metoda POST

- Działanie

dane z formularza są przesyłane w nagłówkach HTTP:

```
POST /action.php HTTP/1.1
```

```
Host: www.sth.com
```

```
name=value&name=value
```

Przeglądarka nie wyświetla danych formularza w pasku adresu:

www.sth.com/action

- Cechy:

- dane żądania nie są widoczne ani zapisywane w historii
(ale dane nie są szyfrowane; bezpieczeństwo -> HTTPS)
- można przesyłać pliki na serwer

Formularze

Odczyt danych z formularza

- Dane z formularza są odczytywane z żądania (met. GET) albo z nagłówka HTTP (met. POST) i dostępne w skrypcie, na kilka możliwych sposobów:
 - zmienne \$name
 - tablice \$HTTP_GET_VARS albo \$HTTP_POST_VARS
 - tablice \$_GET albo \$_POST
 - tablica \$_REQUEST

Należy używać wyłącznie tablic \$_GET lub \$_POST

Niezależnie od użytej metody i typu kontrolki formularza, wszystkie wartości są przekazywane jako łańcuchy tekstowe.

Formularze

Odczyt danych z formularza: tablice `$_GET` i `$_POST`:

- Tablice `$_GET` oraz `$_POST` są superglobalnymi tablicami asocjacyjnymi, gdzie klucze pochodzą od atrybutu **name** kontrolek formularza
- Nie jest wymagana zmiana konfiguracji PHP
- Są superglobalne – dostępne w każdym miejscu skryptu

HTML:

```
<input type="checkbox" name="opcja" value="1">  
<input type="text" name="tekst">
```

PHP:

```
$opcja = $_POST ["opcja"] ?? 0;  
$tekst = $_POST ["tekst"] ?? "";
```

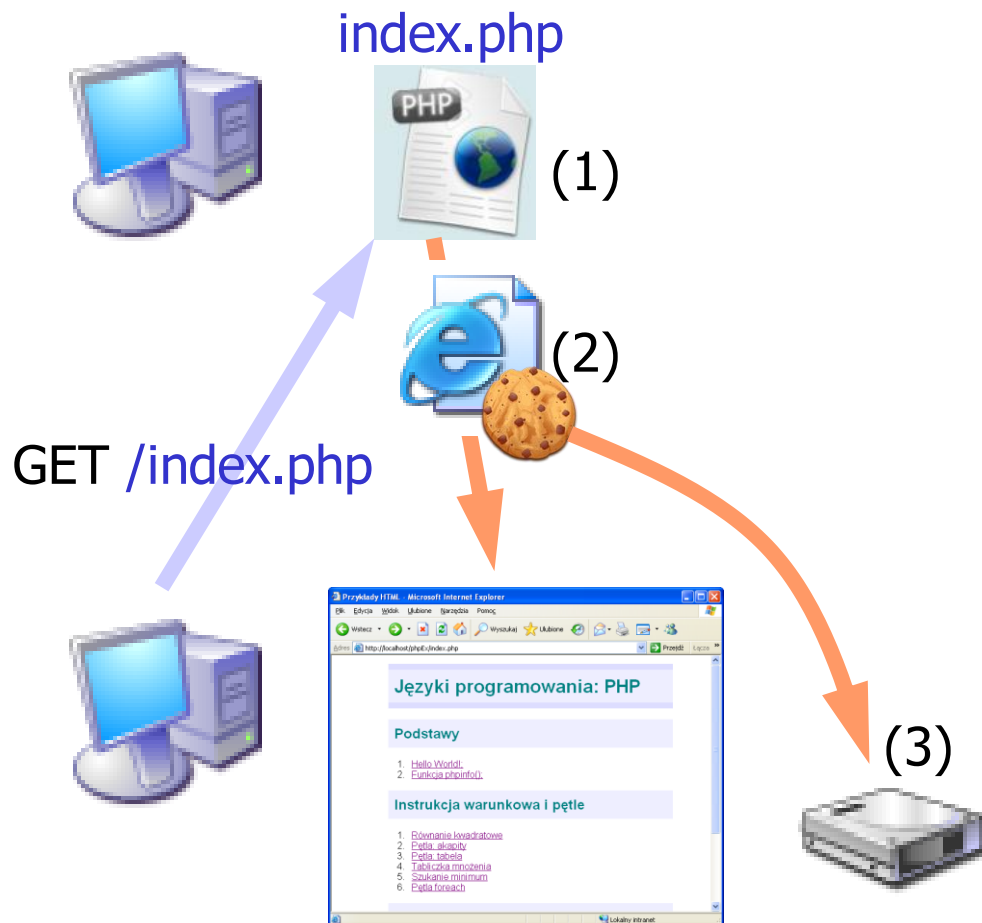
Ciasteczka

Wykorzystanie plików cookie (ciasteczek):

- Protokół HTTP jest bezstanowy – nie są zachowywane żadne informacje o zrealizowanej transakcji HTTP
- Śledzenie stanu nie może być zrealizowane wyłącznie przez serwer (nie ma jednoznacznej identyfikacji klienta)
- Istnieje mechanizm, który pozwala serwerowi zapisać dane po stronie klienta i potem je odczytywać – są to pliki cookie (ciasteczka); Można ich użyć do śledzenia historii transakcji; Ciasteczka nie są w pełni bezpieczne – nie należy ich używać do zapisywania informacji poufnych

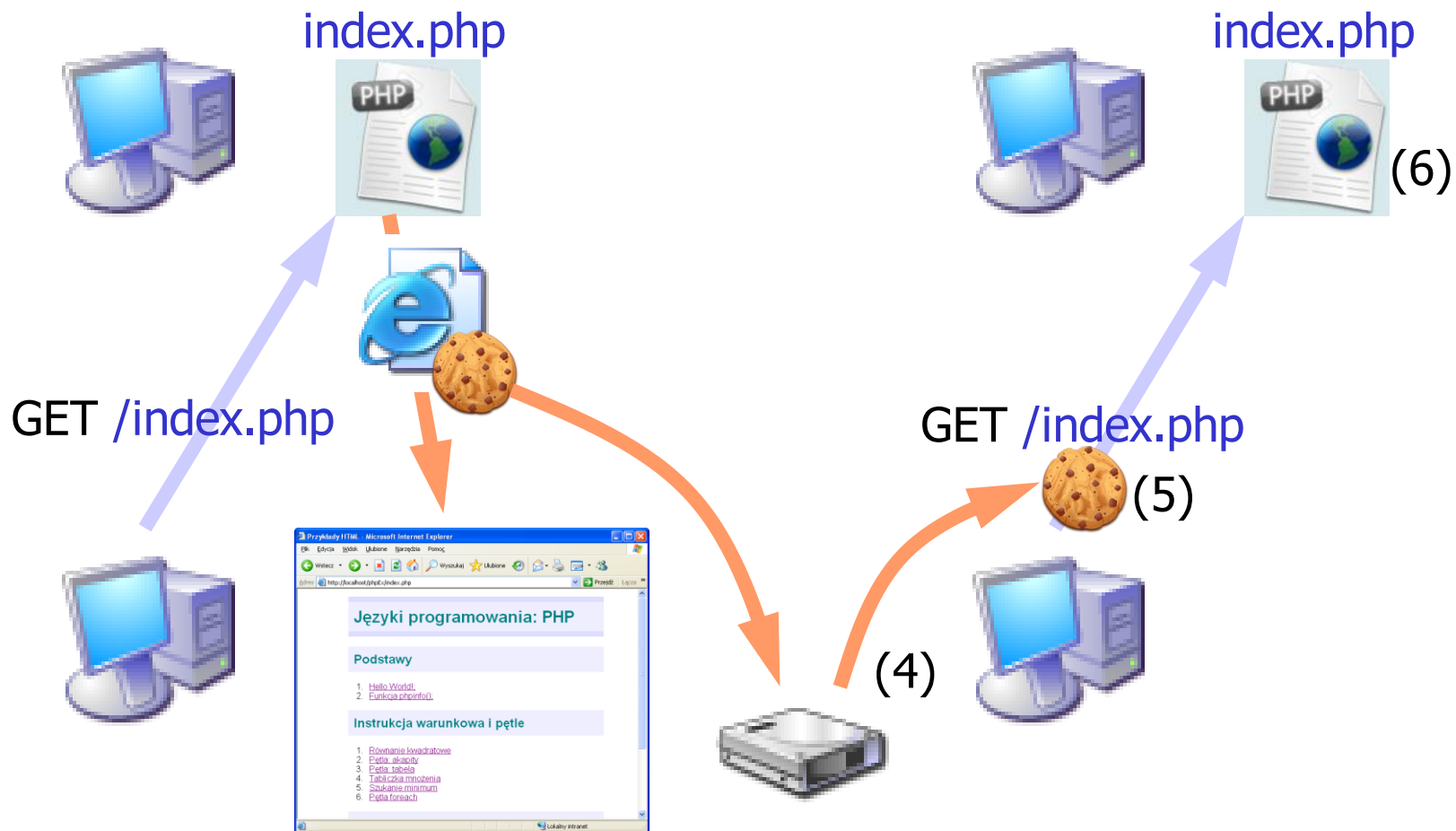
Ciasteczka

(1) Skrypt PHP generuje ciasteczko, które jest (2) przesyłane do przeglądarki, a następnie (3) zapisywane na dysku



Ciasteczka

Przy kolejnym żądaniu (4) ciasteczko jest odczytywane,
(5) przesyłane do serwera i (6) dostępne w skrypcie PHP



Ciasteczka

Obsługa ciasteczek w PHP:

- Tworzenie ciasteczka

Do tworzenia ciasteczka służy funkcja *setCookie*:

```
setCookie (nazwa, wartość, ...);
```

Uwaga! Ze względu na mechanizm przesyłania ciasteczek, funkcja *setCookie* musi być użyta zanim choć jeden znak zostanie umieszczony w strumieniu wyjściowym HTML

```
<?php
    setCookie ("wizyta", $numer);
?>
!DOCTYPE ...
```

Przed znacznikiem `<?php` nie może być choćby spacji; przed użyciem *setCookie* nie można użyć funkcji *echo* ani *print*

Ciasteczka

Obsługa ciasteczek w PHP:

- Funkcja *setCookie*:

```
setCookie (nazwa, wartość, termin,  
          ścieżka, domena, protokół);
```

Argumenty:

- nazwa – nazwa pliku cookie
- wartość – wartość zapisywana w pliku cookie
- termin – termin ważności;
- ścieżka – ścieżka zasobów, dla których plik cookie jest ważny
- domena – domena, dla której plik cookie jest ważny
- protokół – jeżeli "1", to cookie dotyczy tylko protokołu HTTPS

Ciasteczka

Obsługa ciasteczek w PHP:

- Ważniejsze argumenty *setCookie*:

Termin – liczba sekund od 01.01.1970;

- `time()` + sekundy – ważny przez określony czas od teraz
- `time()` - 1 – usunięcie pliku cookie
- 0 – tzw. plik cookie sesji, zapisywany w RAM zamiast dysku

Ścieżka – ścieżka w ramach witryny;

- "/" – cała witryna
- "/press" – tylko folder "/press" i jego podfoldery

domyślnie – folder z którego pochodzi skrypt ustawiający cookie

```
setCookie (nazwa, wartość, time()+300, "/");
```

Ciasteczka

Obsługa ciasteczek w PHP:

- Odczytanie ciasteczka

Przeglądarka automatycznie przesyła ciasteczka w nagłówkach żądania HTTP. Parser PHP odczytuje ciasteczka i udostępnia je w skrypcie. Istnieje kilka metod odczytania wartości ciasteczek:

- zmienne \$name
- tablica \$HTTP_COOKIE_VARS
- tablice \$_COOKIE
- tablica \$_REQUEST

Ze względów bezpieczeństwa zalecane jest korzystanie z tablicy \$_COOKIE

Ciasteczka

Odczytanie ciasteczka: tablica `$_COOKIE`:

- Tablica `$_COOKIE` jest superglobalną tablicą asocjacyjną, gdzie nazwy kluczy pochodzą od nazw ciasteczek

```
<?php
    $zmienna = $_COOKIE['nazwa'];
    ...
?>
```

Jeżeli ciasteczka o podanej nazwie nie ma, zwracane jest NULL.

- Do sprawdzenia czy ciasteczko istnieje można użyć funkcji *isset*:

```
<?php
    if (isset($_COOKIE['nazwa']))
    ...
?>
```

Ciasteczka

Obsługa ciasteczek w PHP:

- Odczytanie ciasteczka

Jeżeli wartość ciasteczka ma być zmieniona, należy to zrobić jeszcze przed umieszczeniem choćby jednego znaku w HTML:

```
<?php
    setCookie (nazwa, wartość);
    $zmienna = $_COOKIE['nazwa'];
?>
```

Uwaga! Użycie funkcji *setCookie* nie zmienia wartości ciasteczka zapisanej w tablicy `$_COOKIE`:

Sesje

Wykorzystanie plików cookie (ciasteczek):

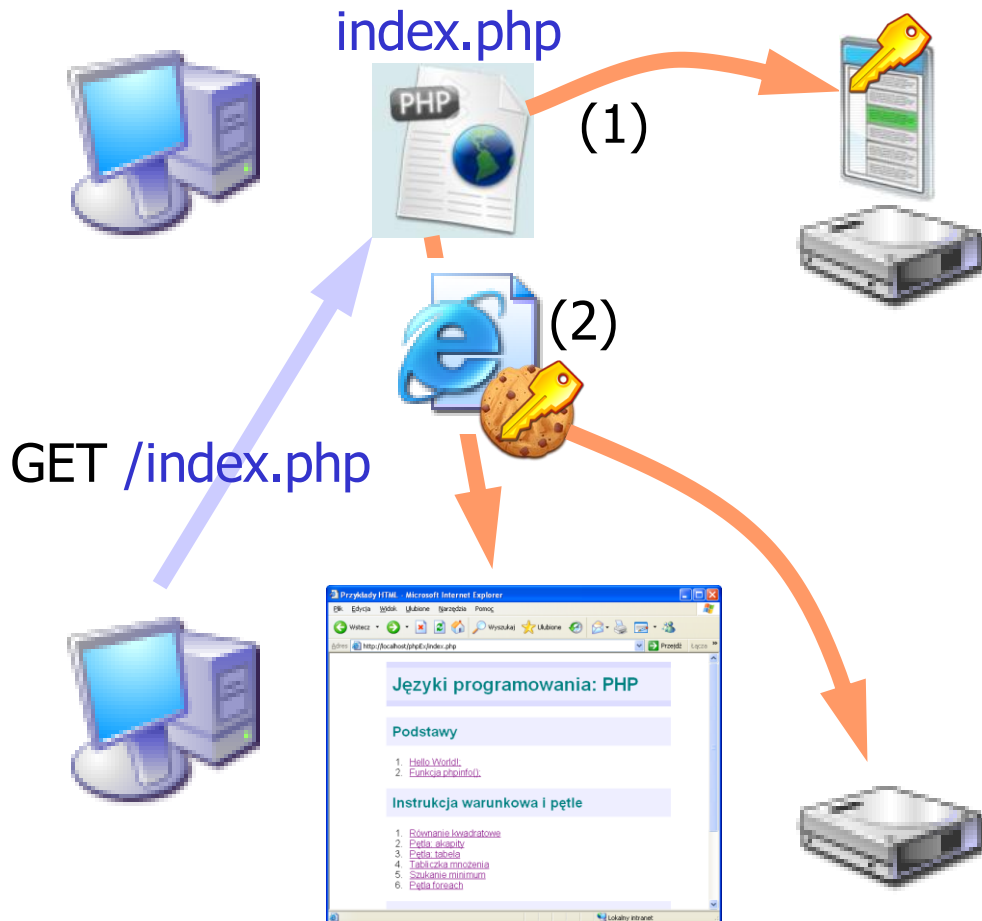
- Ciasteczka nie są w pełni bezpieczne – nie należy ich używać do zapisywania informacji poufnych
- Większe bezpieczeństwo zapewnia mechanizm sesji – dane (w tym poufne) są zapamiętywane po stronie serwera, zaś ciasteczko przechowuje tylko identyfikator sesji;

Sesje również nie są w pełni bezpieczne – dopóki dane (np. login i hasło, identyfikator sesji itp.) są przesyłane w postaci jawnej, można je przechwycić, czemu mechanizm sesji nie może zapobiegać

- Pełniejsze bezpieczeństwo zapewnia połączenie mechanizmu sesji, szyfrowania połączenia przez protokół HTTPS oraz działań zapobiegających atakom XSS (*Cross-Site Scripting*), a w tym konfiguracja PHP, odrzucanie obcych identyfikatorów sesji itd.

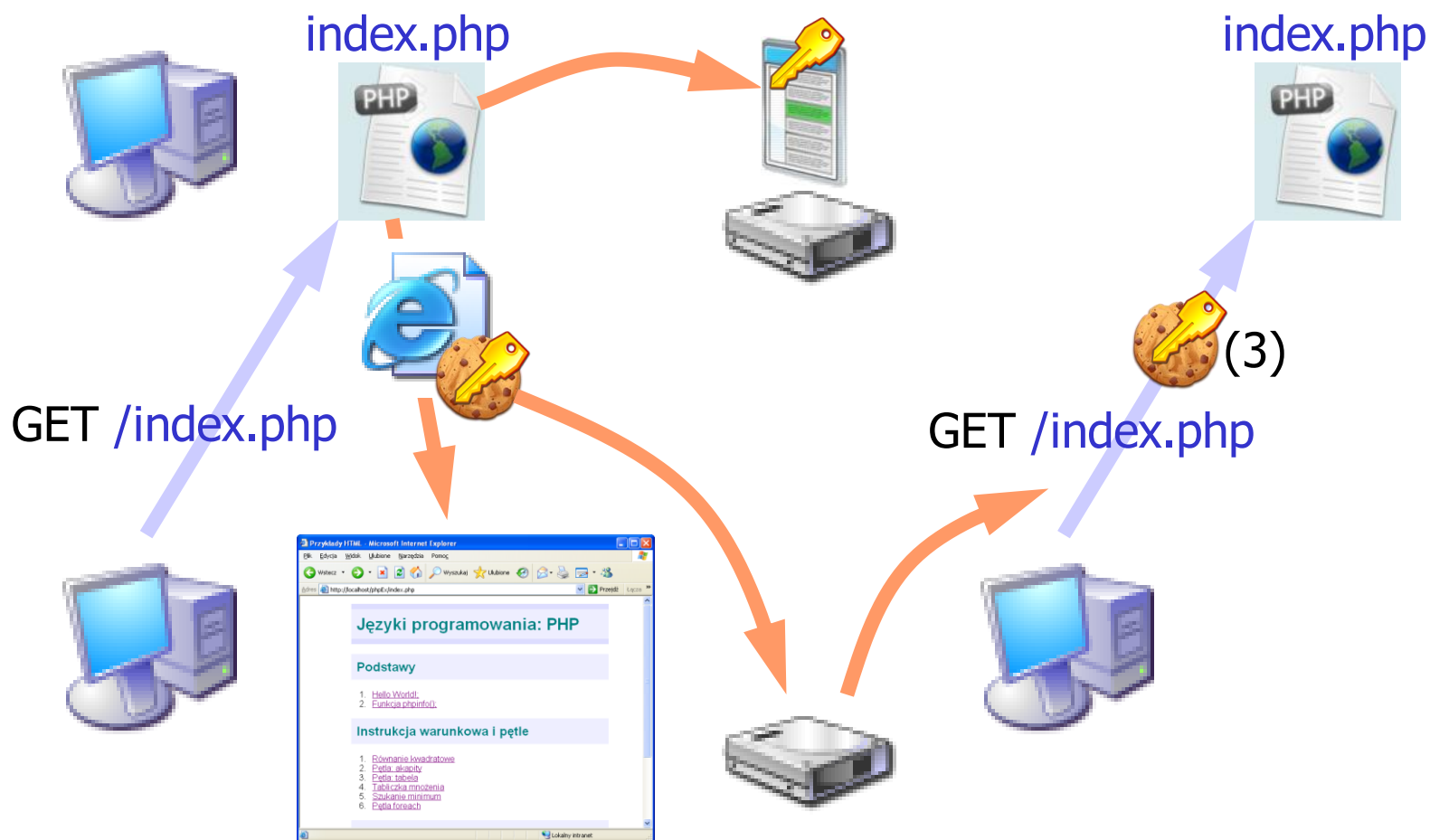
Sesje

PHP (1) zapisuje dane sesji na dysku i generuje identyfikator, który (2) przesyła do przeglądarki jako plik cookie sesji



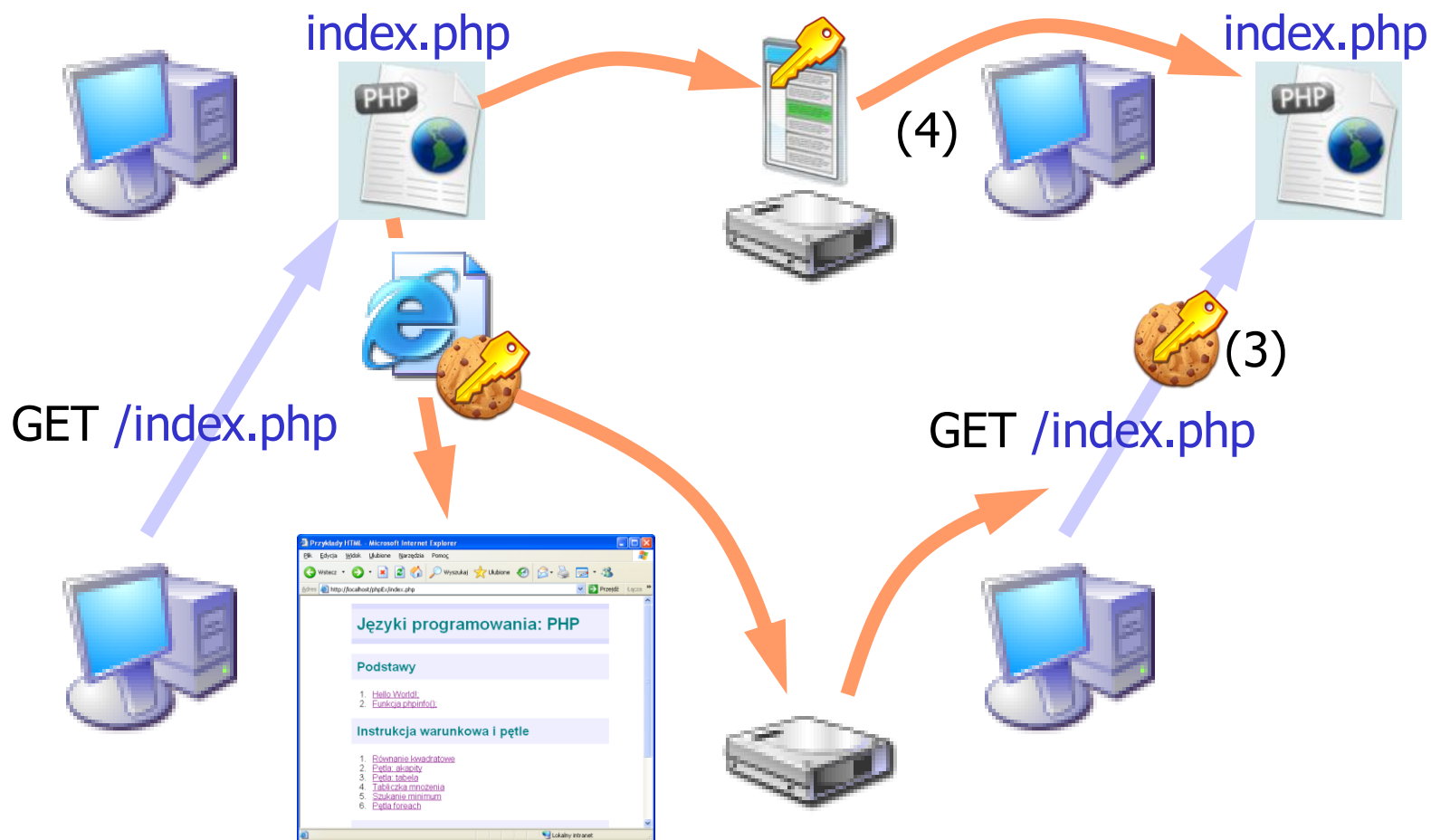
Sesje

Przy kolejnym żądaniu (3) przeglądarka odsyła cookie, ...



Sesje

Przy kolejnym żądaniu (3) przeglądarka odsyła cookie, a serwer (4) odnajduje dane na podst. identyfikatora sesji



Sesje

Obsługa sesji w PHP:

- Tworzenie sesji

Do tworzenia sesji służy funkcja *session_start*:

```
session_start ( ) ;
```

Uwaga! Ze względu na mechanizm przesyłania ciasteczek, funkcja *session_start* musi być użyta zanim choć jeden znak zostanie umieszczony w strumieniu wyjściowym HTML

```
<?php
    session_start ( ) ;
?>
!DOCTYPE ...
```

Przed znacznikiem `<?php` nie może być choćby spacji; przed użyciem *session_start* nie można użyć funkcji *echo* ani *print*

Sesje

Obsługa sesji w PHP:

- Funkcje dotyczące sesji:

```
session_name (nazwa);  
session_name ();
```

Ustawia nazwę sesji (domyślnie zazwyczaj PHPSESSID) lub ją odczytuje (jeżeli pominąć argument *nazwa*);

Chcąc używać własnej nazwy sesji , należy użyć *session_name* zawsze przed wywołaniem *session_start*

```
session_get_cookie_params ();
```

Podaje parametry pliku cookie sesji, w postaci tablicy, np.:

[lifetime]=>0, [path]=>"/", [domain]=>"", [secure]=>""

```
session_id ();
```

Podaje aktualny identyfikator sesji; jest to wygenerowany losowo ciąg znaków ASCII (32 znaki)

Sesje

Obsługa sesji w PHP:

- Zapis i odczyt stanu sesji

PHP udostępnia stan sesji w superglobalnej tablicy `$_SESSION`

Kiedy funkcja *session_start* jest użyta pierwszy raz (dla danego połączenia z klientem) – tworzona jest początkowo pusta tablica `$_SESSION`. Skrypt może w niej zapisać dowolne dane:

```
$_SESSION["userId"]=$userId;
```

Kiedy funkcja *session_start* jest użyta kolejny raz, PHP odtwarza tablicę `$_SESSION` i można odczytać zapisane uprzednio dane:

```
if (isset($_SESSION["userId"]))  
    $userId = $_SESSION["userId"];
```

Dostęp do danych sesji ma każdy skrypt w domenie

Sesje

Obsługa sesji w PHP:

- Niszczenie sesji

Przechwycenie identyfikatora sesji daje dostęp do usług na prawach uprawnionego użytkownika, dlatego niepotrzebne sesje powinny być niszczone:

```
<?php
    session_destroy();
    setcookie (session_name(), "", time()-1000);
    $_SESSION = array();
?>
```

Kolejne wykonanie skryptu z użyciem *session_start* powoduje utworzenie nowej sesji (z nowym identyfikatorem)

PHP po jakimś czasie niszczy nieużywane sesje, ale nie można na tym polegać – haker może wykorzystać nie zamkniętą sesję.

Sesje

Wykorzystanie sesji:

- Śledzenie poczynąń użytkownika
Sesja może przechować dane o działaniach i decyzjach użytkownika, np. sklep internetowy – ostatnio oglądane produkty, stan koszyka itp.
- Pamiętanie uprawnień użytkownika
Sesja może przechować dane o uprawnieniach użytkownika, np. forum internetowe – użytkownik zalogowany lub nie, nazwa konta, rodzaj konta (root/admin/moderator/user) itp.
- Pamiętanie danych z formularzy
Sesja może przechować dane wprowadzane przez użytkownika do formularzy, np. sklep internetowy – podczas kilkietapowego procesu składania zamówienia można cofać się i poprawiać dane bez konieczności ich ponownego wpisywania, można sprawdzić czy formularz nie został wyświetlony na "skróty" itp.

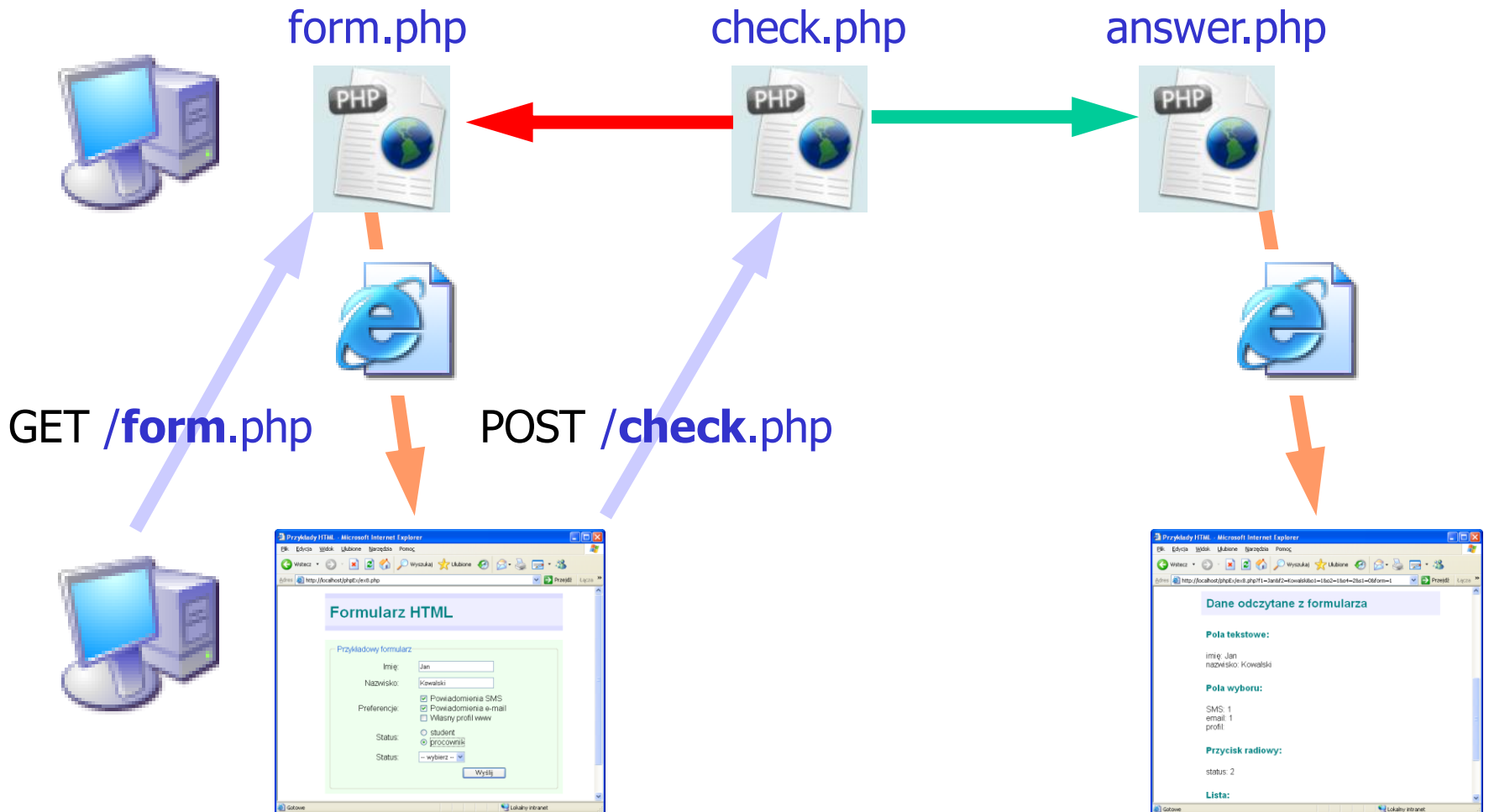
Przekierowania

Wykorzystanie przekierowań:

- Wykonanie pewnych działań (weryfikacja danych, zapis lub odczyt danych w bazie danych, wykonanie obliczeń itp.) bez wyświetlania dokumentu HTML i następnie skierowanie użytkownika na jedną z kilku stron, zależnie od rezultatów wykonanych działań, np.:
 - rejestracja użytkownika: po sprawdzeniu poprawności danych użytkownik jest przełączany do strony potwierdzającej rejestrację docelowej lub ponownie do formularza rejestracji
 - logowanie: po weryfikacji nazwy konta i hasła, użytkownik jest przełączany do strony docelowej lub ponownie do strony logowania (albo po n-tej próbie blokowany)
 - wylogowanie: skrypt wykonuje czynności wylogowania po czym użytkownik jest przełączany do strony głównej

Przekierowania

Mechanizm przekierowania



Przekierowania

Obsługa przekierowania w PHP

- Realizacja przekierowania

Przekierowanie realizuje funkcja *header*, która służy do dołączania dowolnych nagłówków do odpowiedzi HTTP. Przekierowanie dokonuje nagłówek HTTP "Location":

```
header ("Location: adres");
```

Uwaga! Ze względu na mechanizm działania funkcji *header*, musi ona być użyta zanim choć jeden znak zostanie umieszczony w strumieniu wyjściowym HTML

```
<?php
    if (warunek)
        header ("Location: adres");
?>
```

Przed znacznikiem <?php nie może być choćby spacji; przed użyciem *header* nie można użyć funkcji *echo* ani *print*

Przekierowania

Obsługa przekierowania w PHP

- Realizacja przekierowania

W nagłówku "Location" najlepiej jest podać pełen adres zasobu internetowego. Do jego utworzenia można wykorzystać dane zapisane w tablicy `$_SERVER`:

```
<?php
    if (warunek)
    {
        $address =
            "http://" . $_SERVER["HTTP_HOST"]
                . "/ścieżka/docelowy.php";
        header ("Location: $address");
    }
?>
```

Przekierowania

Obsługa przekierowania w PHP

- Realizacja przekierowania

... ale można też podać adres względny: "/" – strona główna,
"./" – bieżący folder lub "/folder/plik.php" – wybrana podstrona

```
<?php
    if (warunek)
    {
        header ("Location: /login.php");
    }
?>
```