**Formularze**

1. **Definicja formularza:**

*<FORM>* - początek definicji formularza

*</FORM>* - koniec definicji formularza

Dostępne parametry:

NAME=nazwa - określa nazwę formularza/

ACTION=url - określa adres programu, który ma przejąć dane z formularza

METHOD=POST/GET - sposób przesyłania danych

1. **Dostępne pola dla formularzy:**
2. **Pole wprowadzania danych**

*<INPUT TYPE=TEXT>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wstępna wartość pola

SIZE=liczba - szerokość pola

MAXLENGTH=liczba - maksymalna liczba znaków

1. **Pole wyboru jednokrotnego**

*<INPUT TYPE=RADIO>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wstępna wartość pola

CHECKED - zaznaczone domyślnie

1. **Pole wyboru wielokrotnego**

*<INPUT TYPE=CHECKBOX>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wstępna wartość pola

CHECKED - zaznaczone domyślnie

1. **Przycisk wysyłania danych**

*<INPUT TYPE=SUBMIT>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wyświetlana wartość

1. **Przycisk resetowania danych**

*<INPUT TYPE=RESET>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wyświetlana wartość

1. **Przycisk ogólnego zastosowania**

*<INPUT TYPE=BUTTON>*

Dostępne parametry:

NAME=nazwa - określa nazwę pola

VALUE=wartość - wyświetlana wartość

1. **Pole wprowadzania tekstów**

*<TEXTAREA>* zawartość *</TEXTAREA>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

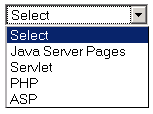
COLS=liczba - ilość kolumn

ROWS=liczba - ilość znaków w wierszu

WRAP=off/soft/hard - sposób przejścia do następnej linii (zawijania tekstu w polu)

1. **Lista rozwijana (drop down list)**

*<SELECT>* opcje *</SELECT>*



Dostępne parametry:

NAME=nazwa - określa nazwę pola

SIZE=liczba - ilość wierszy

MULTIPLE - możliwość wielokrotnego wyboru

1. **Opcje dla listy rozwijanej (dodajemy wewnątrz *<SELECT></SELECT>*)**

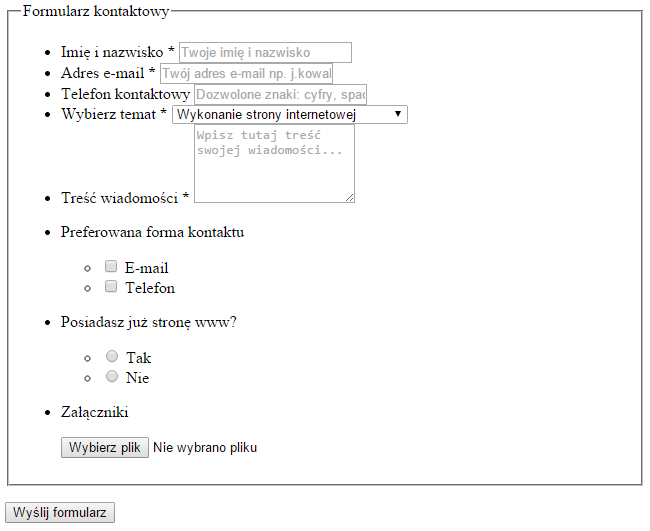
*<OPTION>*tekst wyświetlany*</OPTION>*

Dostępne parametry:

VALUE=wartość - określa wartość opcji

SELECTED - opcja wybrana

1. **Grupowanie pól formularza** - przy pomocy znacznika ***<fieldset> </fieldset>*** można agregować pola formularza w jeden byt, zostaje on zgrupowany i otoczony ramk.
2. **Pola/typy pól dla formularza, dostępne od HTML5** - od wersji HTML5 dostępne są typy pól, które pozwalają zapewnić dodatkową walidację (sprawdzanie) zawartości pola wprowadzoną przez użytkownika, pod kątem jej poprawności względem typu.
   1. ***<INPUT TYPE=”email”>*** *-* waliduje zawartość inputu tak, aby wartość wpisana odpowiadała rzeczywistemu adresowi e-mail
   2. ***<INPUT TYPE=”number”>***- waliduje zawartość inputu poprzez wymuszenie wprowadzenia liczby (możemy dodatkowo użyć parametrów **min/max**, aby określić dodatkowo przedział)
   3. ***<INPUT TYPE=”date”>***- waliduje zawartość inputu poprzez wymuszenie wprowadzenia daty
   4. ***<INPUT TYPE=”time”>***- waliduje zawartość inputu poprzez wymuszenie wprowadzenia czasu
   5. ***<INPUT TYPE=”color”>*** - otwiera użytkownikowi aplet do wyboru koloru
   6. ***<INPUT TYPE=”range”>***- waliduje zawartość inputu poprzez wymuszenie wprowadzenia liczby z podanego zakresu (aby określić zakres, należy użyć parametrów **min/max**, aby określić dodatkowo przedział)
3. Atrybut ***required*** - dzięki niemu możemy wymóc na użytkowniku wypełnienie pola. Tworząc więc pole *<input type="email" required>* upewniamy się, że użytkownik poda nam adres e-mail i nie zostawi pola pustego.
4. Nowy element ***datalist*** daje nam możliwość stworzenia podpowiedzi do wypełnianego pola. Dzięki temu możemy łatwo zasugerować użytkownikowi jedną z najczęściej wybieranych opcji.
5. Przykładowy formularz korzystający z powyższych rodzajów pól:



1. Aby przetworzyć dane z formularza możemy użyć skryptów napisanych np. w języku PHP. W tym celu do znacznika *<FORM>* naszego formularza należy dodać parametr ***action=”adres”***, gdzie w miejsce słowa “adres” wpisujemy adres skryptu, do którego dane mają zostać przekazane.
2. **Kodowanie application/x-www-form-urlencoded:**

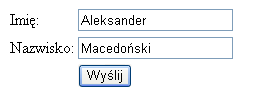
Formularz jest przedstawiany w oknie przeglądarki w postaci szeregu kontrolek. Układ graficzny kontrolek nie wpływa na sposób zakodowania danych. Dane wprowadzone do formularza są kodowane przez przeglądarkę. O sposobie kodowania decyduje atrybut enctype elementu FORM. Domyślnym kodowaniem formularzy jest application/x-www-form-urlencoded. Kodowanie to polega na utworzeniu par:

***nazwakontrolki=wartosc***

i połączeniu ich separatorem &. Wszystkie znaki specjalne występujące w nazwach lub wartościach kontrolek zostają przedstawione w postaci kodu szesnastkowego poprzedzonego znakiem procentu. Na przykład spacja jest zamieniana na napis %20 (kod ASCII znaku spacja - w systemie dziesiętnym - jest równy 32; liczba 32 w systemie szesnastkowym wynosi 20HEX).

Nazwy zmiennych są pobierane z kodu HTML formularza. Każda kontrolka posiada atrybut name. Atrybut ten ustala nazwę zmiennej. W formularzu przedstawionym poniżej występują dwie kontrolki o nazwach ***imie*** oraz ***nazwisko***. Po wprowadzeniu do formularza danych **Aleksander Macedoński**, otrzymamy zakodowany napis:

**imie=Aleksander&nazwisko=Macedo%F1ski**



Pierwszy krok interakcji użytkownika z aplikacją internetową polega na wprowadzeniu danych do formularza. Następnie, po naciśnięciu przycisku Wyślij, przeglądarka koduje wprowadzone przez użytkownika dane, po czym używając wspomnianego parametru ***action*** przesyła je we wskazane miejsce.

1. **Przesyłanie danych pochodzących z formularza protokołem HTTP**

Wszystkie transakcje WWW, a zatem także wysyłanie zawartości formularza, są realizowane przy użyciu protokołu HTTP. Protokół ten definiuje cztery metody przekazywania danych. Metodami tymi są POST, GET HEAD oraz PUT.

W stosunku do formularzy zastosowanie znajdują dwie spośród nich: **GET** oraz **POST**:

1. W metodzie GET dane są dołączone do adresu URL i przyjmują postać:

**ADRES?imie=Aleksander&nazwisko=Macedo%F1ski**

(Gdzie **ADRES** jest wartością parametru ***action***)

1. W metodzie POST dane z formularza są dołączone na końcu zapytania HTTP (za wszystkimi nagłówkami).
2. Metodę przekazywania danych formularza ustalamy atrybutem ***method*** znacznika *<FORM>*:

***<FORM action="ADRES" method="GET">***

1. **Odbieranie danych pochodzących z formularza w skrypcie PHP:**

Tablice **$\_GET**, **$\_POST** oraz **$\_REQUEST** zawierające przetworzone dane pochodzące z formularza i dostępne wewnątrz skryptu php są zmiennymi superglobalnymi. Oznacza to, że są one widoczne wewnątrz wszystkich funkcji i metod bez konieczności stosowania słowa kluczowego global. Tablica $\_GET zawiera dane przekazane do skryptu metodą GET. Tablica $\_POST zawiera dane przekazane do skryptu metodą POST. Natomiast tablica $\_REQUEST zawiera dane pochodzące z ciasteczek, sesji, oraz przekazane metodami POST lub GET.

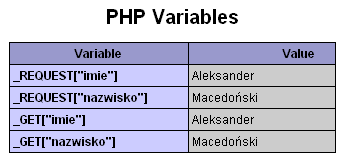
Dane pochodzące z formularzy przekazywanych metodą GET są dostępne w skrypcie php w tablicy $\_GET. Jeśli formularz jest przekazany metodą POST, to należy użyć tablicy $\_POST.

Wszystkie trzy wymienione tablice są tablicami asocjacyjnymi. Indeksem w powyższych tablicach może być napis. Indeksami w tablicach $\_POST i $\_GET są nazwy kontrolek formularza.

Jeśli powyższy formularz przekażemy za pomocą metody GET, odczytać dane z odpowiedniej tablicy możemy np. w sposób: **$\_GET['imie']** lub **$\_GET['nazwisko']**

1. **Jakie dane zostały przesłane do skryptu:**

Wszystkie informacje na temat danych pochodzących z formularza i dostępnych wewnątrz skryptu zwraca funkcja **phpinfo()**. Wynikiem wykonania funkcji w skrypcie, do którego zostali byśmy przekierowani z powyższego formularza przedstawiałby się następująco:



Drugim sposobem sprawdzenia danych przekazanych do skryptu jest użycie jednej z funkcji **var\_dump(), var\_export(), print\_r()** poznanych wcześniej.

1. **Nagłówki transakcji HTTP:**

Droga, jaką odbywają dane wprowadzone do formularza jest następująca:

* użytkownik wypełnia formularz, po czym naciska przycisk Wyślij,
* przeglądarka koduje informacje zawarte w formularzu, a następnie wysyła zapytanie HTTP do serwera,
* aplikacja działająca na serwerze odbiera zapytanie HTTP, zapytanie jest przekazywane przez kolejne warstwy oprogramowania: stos protokołów TCP/IP przekazuje zapytanie do procesu Apache, Apache uruchamia maszynę PHP i przekazuje jej zapytanie, zaś maszyna PHP przetwarza zapytanie, uruchamia skrypt i przekazuje do skryptu tablice $\_GET, $\_POST, itd.
* skrypt przetwarza dane, produkuje wynikowy kod HTML,
* kod zostaje wysłany w odpowiedzi HTTP do przeglądarki.

**HTML5**

1. Specyfikacja **HTML5** wprowadza nowe znaczniki pozwalające w łatwy i intuicyjny sposób budować szkielet strony, który – przez zmniejszenie ilości kodu – jest czytelniejszy i łatwiejszy w utrzymaniu, pozwala poza tym odróżnić elementy strony internetowej, dzięki czemu wiadomo, gdzie jest treść właściwa, gdzie jest menu, a gdzie znajdują się potencjalnie reklamy.
2. **Podstawowa struktura dokumentu HTML:**
3. *<!doctype html> -* określa standard dokumentu w jakim został napisany
4. *<html></html>* - znacznik, który obejmuje wewnątrz cały dokument HTML, informując przeglądarkę, gdzie znajduje się właściwy kod strony w języku HTML
5. *<head></head>*  - część nagłówkowa dokumentu HTML, w której określa się metadane np. autora, nazwę, słowa kluczowe i wiele innych
6. *<body></body>* - właściwa część dokumentu HTML, która definiuje wygląd strony w przeglądarce
7. *<meta charset="UTF-8" >* - określa kodowanie znaków na stronie
8. *<title>Tytuł strony...</title>* - określa tytuł strony, który będzie widniał na belce przeglądarki/karty

W sekcji *<body>* możemy używać wszelakiej maści znaczników html, które formatują nam dane na stronie. Szerszą listę przydatnych znacznikówchociażby. do formatowania tekstu można znaleźć np. pod adresem: <https://technikinformatyk.pl/kursy/kurs/html-css/lekcja/html-lista-znacznikow/>

1. Podstawowy dokument HTML, który używa powyższych znaczników definiujących go, może wyglądać następująco:

*<!doctype html>*

*<html>*

*<head>*

*<meta charset="UTF-8" />*

*<title>Tytuł strony...</title>*

*</head>*

*<body>*

*<p>Hello <b>World</b></p>*

*</body>*

*</html>*

1. Od wersji 5 języka HTML wprowadzona została charakterystyczna budowa, złożona z wprowadzonych razem z tym standardem znaczników:
   1. ***header*** – część nagłówkowa strony
   2. ***nav*** – pojemnik na elementy nawigacyjne
   3. ***section*** – tematyczna grupa treści
   4. ***article*** – treść
   5. ***aside*** – dodatkowe informacje
   6. ***footer*** – stopka

Przykładowa strona zbudowana z dostępnych w HTML5 może mieć taką strukturę:



**ZADANIA**

1. Prosty kalkulator:
2. stwórz formularz z miejscem na wpisanie 2 liczb oraz wyborem działania (dodawanie, odejmowanie, mnożenie, dzielenie)
3. stwórz skrypt PHP, który obsłuży dane z formularza (na podstawie wybranego działania policzy i wyświetli wynik w przeglądarce)
4. Formularz rezerwacji hotelu:
5. stwórz formularz, który będzie pozwalał: podać z listy rozwijanej ilość osób (1-4), których dotyczy rezerwacja, wpisać dane osoby rezerwującej pobyt np. imię, nazwisko, adres, dane karty kredytowej, e-mail, podać datę pobytu, czy godzinę przyjazdu itd. (pamiętając o odpowiedniej walidacji pól - typach), zaznaczyć czy jest potrzeba dostawienia łóżka dla dziecka, z listy wybrać odpowiednie udogodnienia np. klimatyzacja i popielniczka dla palacza (pamiętaj określić które pola są wymagane)
6. stwórz skrypt PHP, który odbierze powyższe dane i w ładny i przejrzysty sposób wyświetli podsumowanie rezerwacji (użyć do wyświetlenia szablonu HTML
7. Dla zadania nr 2 dodaj krok, w którym w zależności od liczby osób wyświetli się formularz, który pozwoli uzupełnić podstawowe dane tych osób w zgrupowanych formularzach i doda tę informację do podsumowania rezerwacji.
8. Czy dana liczba jest liczbą pierwszą?
9. stwórz formularz z miejscem na wpisanie liczby
10. stwórz skrypt PHP, który przyjmie liczbę z formularza (sprawdzi czy to na pewno liczba całkowita dodatnia), a następnie wywoła funkcję, sprawdzającą czy liczba jest liczbą pierwszą
11. w swoim programie umieść zmienną, która policzy wszystkie iteracje pętli, potrzebne do wykonania obliczeń. Spróbuj tak zmodyfikować program, by było potrzeba jak najmniej iteracji (przy zachowaniu prawidłowego działania).