

Formularze

Interfejsy służące użytkownikom do wprowadzania danych wejściowych do witryny, celem ich przetworzenia (np. formularz rejestracji wprowadzający nowego użytkownika do bazy, formularz logowania wprowadzający login i hasło, których poprawność następnie sprawdzamy w bazie danych).

```
<form action="login.php" method="post">
```

obszar formularza, w którym pojawią się wybrane kontrolki wymiany danych

```
</form>
```

Cały obszar formularza zamykamy w znacznikach <form>, zaś pomiędzy tymi tagami, układamy z predefiniowanych kontrolek (niczym z klocków lego) dowolny potrzebny w witrynie interfejs.

```
<form></form>
<label></label>
<fieldset></fieldset>
<legend></legend>
<input type="text">
<input type="password">
<input type="number">
<input type="search">
<input type="checkbox">
<input type="radio">
<input type="tel">
<input type="email">
```

```
<input type="date">
<input type="month">
<input type="week">
<input type="time">
<input type="color">
<select></select>
<select multiple></select>
<textarea></textarea>
<input type="submit">
<input type="button">
<button></button>
```

<form></form>

(ang. form = formularz) obszar na stronie internetowej, zawierający różnorodne elementy interfejsu, służące do komunikacji / wymiany danych z użytkownikiem witryny

```
<form action="login.php" method="post">
```

dane zostaną wysłane do pliku login.php

metoda przesyłania danych: "get" albo "post"

```
</form>
```

action - który skrypt PHP ma przetworzyć dane, które przyjdą z naszego formularza

method - jakiej metody HTTP użyć do przesyłania danych formularza (**get** czy **post**)

`<form></form>`

`enctype="application/x-www-form-urlencoded"`

Wartość domyślna, przyjmowana jeśli atrybutu nie określono w tagu `<form>`. Wszystkie znaki zostają zakodowane metodą urlencode - spacje zamieniają się na (+), zaś wszystkie wartości, które nie są alfanumeryczne zostaną zastąpione procentem (%), po którym pojawią się dwie cyfry zapisane szesnastkowo, oznaczające dany znak. Więcej szczegółów: bit.ly/url-encode

`enctype="multipart/form-data"`

Wartość stosowana dla plików (input typu file) - znaki nie zostają zakodowane w urlencode (ważne, aby plik dotarł na serwer w niezmienionej postaci).

`enctype="text/plain"`

Ustawienie tego MIME-type oznacza, iż przesyłane znaki reprezentują tekst możliwy do przeczytania przez człowieka i bezpośredniego wyświetlenia przez przeglądarkę – nie ma kodowania znaków specjalnych w urlencode

Wybór jednokrotny

Możliwy jest wybór tylko jednej opcji z wybranego zbioru (np. wybór rozmiaru t-shirta, nadwozia samochodu etc.)

Opcja pierwsza	▼
Opcja druga	
Opcja trzecia	

- ☒ Opcja pierwsza
- ☐ Opcja druga
- ☐ Opcja trzecia

Zasada ogólna

We współczesnych formularzach każda kontrolka formularza powinna posiadać "gorącą" (aktywującą element) etykietę.

```
<form action="login.php" method="post">  
  <label>Podaj login: <input type="text"></label>  
  
</form>
```

Wyjątkiem są przyciski, których aktywacja odbywa się po prostu przez kliknięcie ich, zaś napis "związany" z przyciskiem już przecież znajduje się w środku buttona.

Dwa możliwe zapisy etykiety

Metoda pierwsza z użyciem atrybutów: for, id o tej samej wartości

```
<label for="pole">Podaj hasło:</label>  
<input type="password" id="pole">
```

Metoda druga - umieszczenie kontrolki formularza pomiędzy znacznikami <label></label>

```
<label>Podaj login: <input type="text"></label>
```


Input type email (1/2)

Element formularza dedykowany wprowadzaniu adresu poczty elektronicznej.

Kontrolka akceptuje:

- pustą wartość – o ile nie jest obecny atrybut `required`
- poprawny składniowo adres pocztowy – sprawdzane jest odpowiednie tzw. wyrażenie regularne, nie zaś "autentyczność" samego adresu
- poprawne składniowo adresy pocztowe – potrzebny jest dodatkowy atrybut `multiple`, który pozwala wprowadzić kilka adresów rozdzielonych przecinkiem, na przykład: `adres1@domena.pl, adres2@domena.pl`

Input type email (2/2)

W przeglądarkach, które nie obsługują pola typu email (np. IE<10) nastąpi tzw. **fallback** (cofnięcie się, downgrade) do zwykłego inputa tekstowego (a więc pozbawionego prewalidacji).

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu **name**) inputa typu email.

PHP

```
$email = $_POST['adres'];  
echo $email;
```

Walidacja po stronie przeglądarki - zasada ogólna

Przeglądarkowa prewalidacja formularza nie może nigdy stanowić substytutu właściwej sanityzacji wejścia po stronie serwera! Nie możemy ufać front-endowym ograniczeniom nałożonym w interfejsie, który tak łatwo przecież zmodyfikować po stronie klienta.

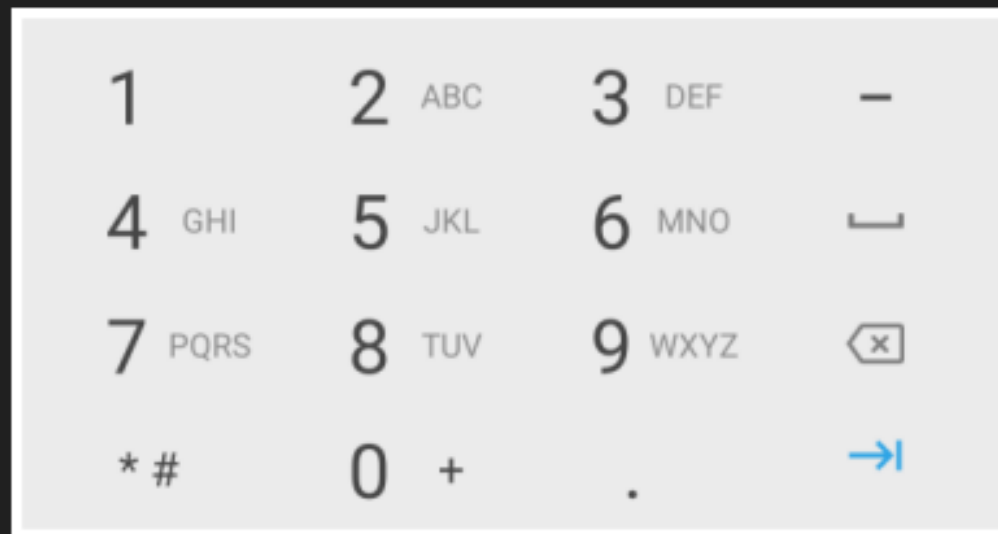
Inna rzecz, że dane do skryptu (u nas [order.php](#)) można przysłać bezpośrednim żądaniem na serwer, czyli w ogóle pomijając część front-endową naszej witryny!

Input type tel (1/2)

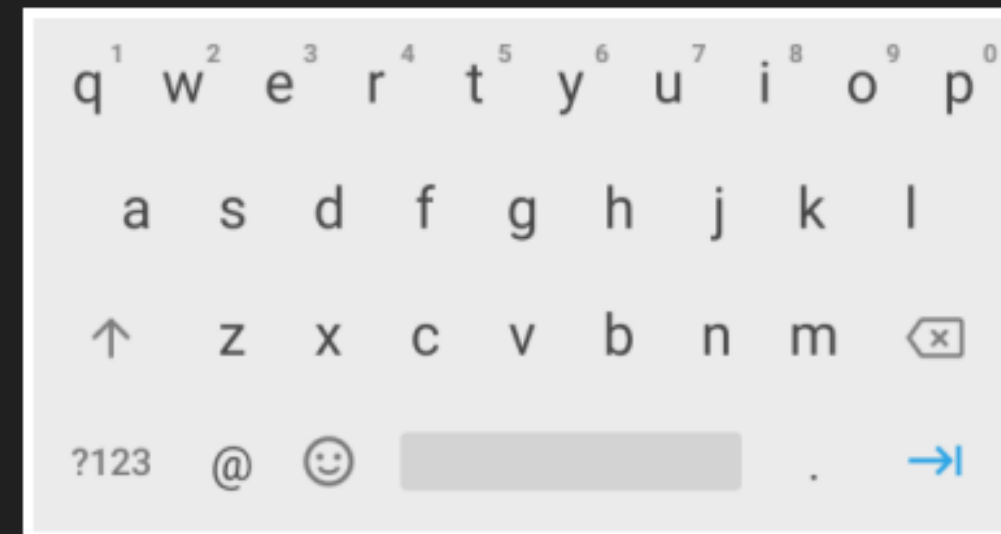
Kontrolka formularza dedykowana wprowadzaniu numeru telefonu, lecz co ciekawe - pozbawiona jest sprawdzania jego poprawności (z powodu zbyt dużej różnorodności form zapisu numerów telefonicznych na świecie).

Jednak główną zaletą inputa typu tel jest wyświetlenie na urządzeniach mobilnych (smartfony, tablety) uproszczonej klawiatury do obsługi tego pola:

Klawiatura inputa tel



Klawiatura inputa text



Input type tel (2/2)

W przeglądarkach, które nie obsługują pola typu tel (np. IE<10) nastąpi fallback do zwykłego inputa tekstowego.

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) inputa typu tel:

PHP

```
$numer_telefonu = $_POST['telefon'];  
echo $numer_telefonu;
```

Input type date (1/2)

Kontrolka formularza dedykowana wprowadzaniu daty kalendarzowej, wyposażona w wygodny interfejs wyboru daty (różnie wyglądający w poszczególnych przeglądarkach).

Element akceptuje:

- pustą wartość – o ile nie jest obecny atrybut **required**
- poprawną datę – dodatkowo można ustawić atrybutami **min** oraz **max** akceptowany przedział dat (data spoza przedziału nie przejdzie prewalidacji)

Uwaga! Pole typu date nie jest wspierane w przeglądarkach IE oraz Safari – nastąpi w nich fallback do zwykłego inputa tekstowego.

Input type date (2/2)

Warto także zwrócić uwagę na inną postać zapisu daty (warstwa użytkownika vs. warstwa programisty):

- W przeglądarce wygląda to tak: 14.02.2019
- Po stronie serwera w PHP jest tak: 2019-02-14

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) inputa typu date.

PHP

```
$data_dostawy = $_POST['dzien'];  
echo $data_dostawy;
```


Input type time (1/2)

Kontrolka formularza dedykowana wprowadzaniu czasu – godzina, minuty oraz ewentualnie sekundy (przy obecności atrybutu **step**), wyposażona w wygodny interfejs i prewalidację wprowadzonego czasu. Element nie pozwala wpisywać liter, reaguje tylko na cyfry.

Input akceptuje:

- **pustą wartość** – o ile nie jest obecny atrybut **required**
- **poprawny czas** – w formacie 24-godzinny albo 12-godzinny (AM/PM) zależnie od konfiguracji naszego systemu operacyjnego. Zapis 12-godzinny czasu: **AM** (łac. ante meridiem) - przed południem, czyli od północy do godziny 12:00 oraz **PM** (łac. post meridiem) - po południu, czyli od godziny 12:00 do północy

Input type time (2/2)

Uwaga! Pole typu time nie jest wspierane w przeglądarkach IE oraz Safari – nastąpi w nich fallback do zwykłego inputa tekstowego.

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) inputa typu time.

PHP

```
$czas_dostawy = $_POST['czas'];  
echo $czas_dostawy;
```

Warto także zdawać sobie sprawę, że odebrana wartość będzie zawsze w formacie 24-godzinnym (to czy pojawi się liczba sekund zależy od obecności atrybutu `step`).

Pole wieloliniowe textarea (1/2)

Element formularza dedykowany wprowadzaniu dłuższych, kilkulinowych wypowiedzi, na przykład komentarzy czy prywatnych wiadomości. Pole textarea nie posiada paska narzędziowego z funkcjami edycji tekstu, pozwala jedynie na wpisanie więcej niż jednej linii tekstu (choć oczywiście może ono także pozostać puste, jeśli nie wymuszamy wypełnienia atrybutem **required**).

Ważne atrybuty:

- Ilość wierszy **rows** oraz ilość znaków **cols**
- Ograniczenia ilości znaków w polu – **minlength** oraz **maxlength**

Aby pozbawić użytkownika możliwości zwymiarowania rozmiaru pola, możemy w CSS dopisać do kontrolki właściwość **resize: none;**

Pole wieloliniowe textarea (2/2)

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) pola wieloliniowego:

PHP

```
$komentarz = $_POST['komentarz'];  
echo $komentarz;
```

Edytory WYSIWYG (ang. `what you see is what you get`) wspierane JavaScriptem, bogatsze w opcje formatowania tekstu, często wypierają klasyczne pole textarea. Przykłady: [CKEditor](#), [TinyMCE](#), [Summernote](#), [Froala Editor](#)

Input type file (1/3)

Kontrolka formularza dedykowana przesyłaniu na serwer plików. Po dodaniu do formularza tego typu inputa, warto ustawić otwierającemu znacznikowi form atrybut: `enctype="multipart/form-data"`. Dzięki temu dane wejściowe nie zostaną zakodowane w urlencode (ważne jest, aby plik dotarł na serwer w niezmienionej postaci).

Pole to jest trudne do obsłużenia tak front-endowo, jak i back-endowo.

Trudność po stronie klienta polega na tym, iż do ostylewania kontrolki najczęściej używamy dość określonych metod – samego inputa ukrywa się poprzez dopisanie `display:none` albo `position:absolute`, zaś rolę wizualizacji kontrolki przejmuje jej odpowiednio ostylewana etykieta `label` (widoczna w miejscu pola typu `file`).

Trudność po stronie serwera polega zaś na tym, iż należy wyjątkowo dokładnie sanityzować plik przesłany przez użytkownika – pozwolenie na umieszczenie na naszym serwerze pliku to ryzykowna pod kątem bezpieczeństwa decyzja.

Input type file (2/3)

Odebranie wartości w PHP następuje z użyciem tablicy `$_FILES`, szufladka ma indeks taki sam jak nazwa (wartość atrybutu `name`) inputa typu file, jednak tablica ta jest dwuwymiarowa. Indeks drugiej szufladki to `wybrany atrybut pliku`, np. `tmp_name`, `type`, `size`.

PHP

```
if(isset($_FILES['obraz']))  
{  
    // skrypt sanityzujący  
    // plik wejściowy  
}
```


Input type file (3/3)

Ważne atrybuty:

- **accept** – wpływamy na filtr pokazujący rodzaj (rozszerzenie) pliku w oknie, w którym następuje wybór konkretnego pliku
- **multiple** – pozwala wskazać w oknie wyboru kilka plików

Pole typu file może i jest nieco archaiczne, ale przez to obsługiwane w praktycznie każdej przeglądarce (hurra!)

Input type color

Kontrolka formularza dedykowana pobraniu od użytkownika koloru, wyposażona w interfejs jego wskazania. Wartość zakodowana jest szesnastkowo, w klasycznym formacie #RRGGBB (składowe Red, Green, Blue). Odebranie wartości w PHP następuje standardowo z użyciem tablicy \$_POST, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu **name**) pola typu color:

PHP

```
$kolor = $_POST['kolor'];  
echo $kolor;
```

Uwaga! Pole typu color nie jest wspierane w przeglądarkach IE oraz Safari na MacOS – nastąpi w nich fallback do zwykłego inputa tekstowego, przez co nie pojawi się interfejs wskazania koloru (a nie każdy użytkownik potrafi ręcznie wpisać szesnastkowy kod koloru). Stąd często takie pole zastępowane jest color pickerem obsługiwany przez JavaScript (na przykład jscolor.com).

Znacznik progress (1/2)

Kontrolka formularza obrazująca dla użytkownika stopień postępu jakiegoś procesu (na przykład uploadu pliku czy procentowego stopnia ukończenia testu).

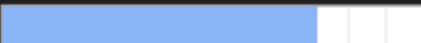
Ważne atrybuty:


- **max** – wartość największa, czyli postęp maksymalny (ukończony)
- **value** – wartość aktualna
- **id** – identyfikator pozwalający uchwycić element w JS (ważny, gdyż często skrypt JS będzie podmieniać **value** progressbara, reagując na postęp procesu)


Znacznik progress jest obsługiwany w wielu przeglądarkach (wyjątkiem jest klasycznie IE w wersji poniżej 10). Czasami jednak ten wskaźnik postępu zastępowany jest wersją animowaną, wspieraną przez JS - przykłady: [ProgressBar.js](#), [LoadingBar.js](#)

Znacznik progress (2/2)

Wygląd domyślny paska postępu zależy od tego na jakim systemie operacyjnym / w jakiej przeglądarce się znajdujemy:

Opera 58.0 Stopień realizacji zamówienia 

Edge 17.17 Stopień realizacji zamówienia 

Firefox 65.0 Stopień realizacji zamówienia 

Wartości w PHP nie odbieramy, gdyż ten element służy nam, developerom do przekazania wartości użytkownikowi (nie odwrotnie).

Znacznik meter

Kontrolka formularza obrazująca wartość jakiejś wielkości, ale w odniesieniu do jej możliwego przedziału (zakresu, zasięgu). Na przykład: wartość zużycia baterii w porównaniu do jej pełnego naładowania, ilość paliwa w baku, wzięwszy pod uwagę pojemność całego zbiornika na paliwo w samochodzie. Ważne atrybuty:

- **min, max** – wartość najmniejsza / największa w podanym zasięgu (przedziale)
- **title** – tytuł dostępny przy hoverze (może zawierać odpowiedź co do jednostki, w której wyrażono naszą wielkość fizyczną)
- **low, high, optimum** – wartości uznawane kolejno za poziom: niski, wysoki i optymalny – te atrybuty pomagają dobrać elementowi kolory wypełnienia (czerwony oznacza wartość bliską minimum/low, żółta poniżej optymalnej, a zielona optymalną / wysoką)
- **value** – aktualna wartość

Wartości w PHP nie odbieramy, gdyż ten element służy nam, developerom do przekazania wartości użytkownikowi (nie odwrotnie).

Input type url

Element formularza dedykowany wprowadzaniu linka w formacie URL (ang. Uniform Resource Locator). Kontrolka akceptuje:

- pustą wartość – o ile nie jest obecny atrybut `required`
- poprawny składniowo link URL – sprawdzane jest odpowiednie tzw. **wyrażenie regularne**, nie zaś „autentyczność” samego adresu. W praktyce warto pamiętać, że link w formacie URL musi być wyposażony w **oznaczenie usługi (protokołu)**, po czym następuje `://` Przykład → <http://pasja-informatyki.pl>
- Ograniczenie ilości znaków w polu – `minlength` oraz `maxlength`

W przeglądarkach, które nie obsługują pola typu email (np. IE<10) nastąpi fallback do zwykłego inputa tekstowego (a więc pozbawionego walidacji zasobu URL).

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) pola wieloliniowego.

Input type hidden

Kontrolka ukryta (niewidoczna) w warstwie prezentacji formularza użytkownikowi, jawna staje się dopiero w kodzie źródłowym witryny. Jako element ukryty nie musi posiadać etykiety label. Służy do przesyłania na serwer danych, których użytkownik wypełniający formularz nie musi / nie potrzebuje widzieć (np. generowany przez PHP token uwierzytelniający usera w sesji).

Odebranie wartości w PHP następuje standardowo z użyciem tablicy `$_POST`, szufladka ma indeks taki sam, jak nazwa (wartość atrybutu `name`) pola typu hidden.

PHP

```
$token = $_POST['token'];  
if ($token != $token_w_sesji)  
{  
    // hmm, to podejrzane!  
}
```

Input type reset

Przycisk, który usuwa dane już wprowadzone do formularza oraz przywraca wybory domyślne (najprościej mówiąc: resetuje formularz do stanu pierwotnego).

Nie należy jednak mieć na uwadze możliwość pomyłki użytkownika – jeśli przycisk resetujący umieścimy blisko przycisku wysyłającego formularz, to może zdarzyć się, że ktoś wykasuje wprowadzone dane z intencją ich wysłania. Należy więc odpowiednio rozplanować interfejs, aby do tak frustrującej sytuacji w witrynie nie dochodziło.

Dodatkowe atrybuty kontrolek

Istnieje kilka globalnych, dodatkowych atrybutów, które można nadać wielu elementom formularza. Przypomnijmy najważniejsze z nich:

- **required** – wartość elementu formularza staje się wymagana (czyli bez jej podania nie nastąpi submit)
- **disabled** – element formularza staje się nieaktywny (jest wyszarzony)
- **readonly** – elementowi formularza nie można zmienić wartości

Pola input typu radio

Wszystkie inputy radio, które stanowią wybór jednokrotny, powinny mieć ustawiony atrybut name o takiej samej wartości

```
<form action="login.php" method="post">  
  <label><input type="radio" name="plec" value="m">Mężczyzna</label>  
  <label><input type="radio" name="plec" value="k">Kobieta</label>  
  
</form>
```

Ustawienie wartości domyślnej w kodzie formularza zrealizujemy dodatkowym atrybutem checked

```
<label><input type="radio" name="nadwozie" checked>Hatchback</label>
```

Odebranie wartości pól radio w PHP

W praktyce, z właściwości name korzystamy w PHP do uchwycenia wartości kontrolki formularza, zaś atrybut id kontrolki służy do uchwycenia jej w JavaScript lub do poprawnego działania etykiet z atrybutem for

```
<div><label><input type="radio" value="1" name="nadwozie" checked> Hatchback </label></div>  
<div><label><input type="radio" value="2" name="nadwozie"> Kabriolet </label></div>
```

PHP – odebranie wartości (value) 1 lub 2

```
$wartosc = $_POST['nadwozie'];  
echo $wartosc;      wartością w zmiennej będzie: 1 lub 2
```

Znaczniki: <fieldset> i <legend>

Z pomocą podwójnego znacznika <fieldset></fieldset> możemy utworzyć zgrupowanie kontrolek formularza otoczone ramką, opatrzone dodatkowo legendą (opisem), utworzoną znacznikiem <legend></legend>

```
<fieldset>
```

```
  <legend> Rodzaj nadwozia </legend>
```

```
  <div><label><input type="radio" name="nadwozie"> Hatchback </label></div>
```

```
  <div><label><input type="radio" name="nadwozie"> Kabriolet </label></div>
```

```
</fieldset>
```

Rodzaj nadwozia

- ☐ Hatchback
- ☐ Kabriolet

Zasada ogólna

We współczesnych formularzach, zamiast znaczników `
`, do rozmieszczania kontrolek formularza najlepiej użyć elementów blokowych, w razie potrzeby mających nadany `display:inline-block` lub `float:left` (lub jak się dowiemy później: `display:flex` lub `display:inline-flex`)

```
<div>
  <label>
    <input type="radio" name="nadwozie" checked> Hatchback
  </label>
</div>
<div>
  <label>
    <input type="radio" name="nadwozie"> Kabriolet
  </label>
</div>
```

Lista wyboru (wybór jednokrotny)

```
<label for="silnik"> Rodzaj silnika </label>
<select id="silnik" name="silnik">
  <option value="w"> wysokoprężny </option>
  <option value="b" selected> benzynowy </option>
  <option value="e"> elektryczny </option>
  <option value="h"> hybrydowy </option>
</select>
```

PHP – odebranie wartości (value) - tutaj: w, b, e lub h

```
$wartosc = $_POST['silnik'];
echo $wartosc;    wartością w zmiennej będzie: w, b, e lub h
```

Wybór wielokrotny

Możliwy jest wybór wielu opcji z wybranego zbioru (np. wybór konfiguracji danego modelu samochodu lub składników zamawianej pizzy).



Opcja pierwsza



Opcja druga



Opcja trzecia

Opcja pierwsza



Opcja druga

Opcja trzecia



Pola input typu checkbox

W przeciwieństwie do pól radio, możliwe jest zaznaczenie / odznaczenie dowolnej ilości pól (no, chyba że jakieś pole jest nieaktywne - wówczas nie można zmienić jego stanu)

```
<div>
  <label>
    <input type="checkbox" name="wyp[]" value="1" disabled checked> klimatyzacja
  </label>
</div>
<div>
  <label>
    <input type="checkbox" name="wyp[]" value="2" checked> systemy: ESP i ABS
  </label>
</div>
```

pole nieaktywne pole zaznaczone

Odebranie wartości checkboxów w PHP

PHP - odebranie wartości (wartość odbierzemy tylko dla pól aktywnych)

```
$wartosci = $_POST['wyp'];  
  
for ($i=0; $i<count($wartosci); $i++)  
{  
    echo $wartosci[$i]."<br>";  
}  
  
// to samo wykonane na pętli foreach  
  
$wartosci = $_POST['wyp'];  
foreach ($wartosci as $opcja)  
{  
    echo $opcja."<br>";  
}
```

W HTML ustawiono dla wszystkich zgrupowanych checkboxów atrybut:

```
name="wyp[]"
```

(przesyłanie w tablicy)

Lista multiple (wybór wielokrotny)

```
<label for="choinka"> Jakie choinki zapachowe? </label>
<select id="choinka" name="choinka[]" multiple size="4">
  <option value="k">kokosowa</option>
  <option value="c" selected>cytrynowa</option>
  <option value="t">truskawkowa</option>
  <option value="b">brzoskwiniowa</option>
</select>
```

PHP – odebranie wartości (value) - tutaj: k, c, t, b

```
$wartosci = $_POST['choinka'];
foreach ($wartosci as $opcja)
{
    echo $opcja."<br>";
}
```

Pola tekstowe jednoliniowe

Pole tekstowe, maskowane pole z hasłem, pole wyszukiwarki:

```
<label> Imię <input type="text" placeholder="Imię" name="imie"></label>  
<label> Hasło <input type="password" name="haslo"></label>  
<label> Szukaj <input type="search" name="frazza"></label>
```

PHP – odebranie wartości z pola:

```
$imie = $_POST['imie'];    $haslo = $_POST['haslo'];  
$frazza = $_POST['frazza'];    echo $imie." " ".$haslo." " ".$frazza;
```

Atrybut placeholder pola tekstowego

Stosowanie tego typu podpowiedzi to "gorący" temat wśród web-developerów (niektórzy uważają, że placeholderzy psują całkowicie user-experience wielu formularzy). Na pewno nie należy nadużywać placeholderów, a już szczególnie nie mogą one zastąpić etykiet

Input typu number

Pole nie akceptuje znaków nieużywanych w liczbach dziesiętnych. Możliwe jest także zastosowanie zapisu naukowego – na przykład 3.5e2 to inaczej liczba 350 (przecinek przesuwamy o dwa miejsca w prawo zgodnie z deklaracją „e2”)

```
<label>
  Ile kluczyków
  <input type="number" name="kluczyk" step="0.1">
</label>
```

Atrybut step oznacza krok zmiany wartości liczby (domyślna wartość to 1). Możemy też użyć atrybutu required - wówczas wymusimy wypełnienie pola. Odebranie wartości w PHP:

```
$liczba = $_POST['kluczyk']; echo $liczba;
```

Rodzaje przycisków w formularzach

Przycisk `input type submit` - służy w formularzu do dokonania tzw. submita, czyli podsumowania całego formularza

```
<input type="submit" value="Zamawiam">  
<button>Zamawiam</div>
```

spowodują submit formularza

```
<input type="button" value="Zamawiam">  
<button type="button">Zamawiam</div>
```

nie spowodują submita formularza

Przycisk `input type button` - definiuje "zwykły" przycisk – w sensie: jego kliknięcie nie submituje całego formularza (po kliknięciu nie zostaje podjęta próba wysłania danych wybraną metodą na serwer)

Przycisk `button` - zamiast pojedynczego znacznika `<input>`, możemy zamiennie zapisać podwójny znacznik `<button></button>`. A to sprawia, że pomiędzy tagami można np. umieścić obrazek ``, czyli niejako uatrakcyjnić wizualnie ten element. Kliknięcie elementu `<button>`, powoduje podsumowanie formularza (chyba że obecny jest dodatkowo atrybut `type="button"`)