JavaScript – część 2

Liczenie w pamięci nie jest mocną stroną wielu osób. Bez większego problemu można jednak na stronie WWW... stworzyć prosty kalkulator. W tej części warsztatu poświęconego skryptowi Java zobaczymy, jak dobrze uzupełniają się JavaScript i HTML.

W pierwszej części warsztatu JavaScript (CHIP 5/97) pokazaliśmy, jak wstawić skrypt Javy do dokumentu HTML. Nauczyliśmy się również, w jaki sposób przy pomocy metody document.write() wyświetlić tekst na stronie WWW oraz jak posłużyć się obiektem Date(), aby umieścić na stronie bieżącą datę i czas systemowy naszego komputera. Dziś ciąg dalszy odpowiedzi na niektóre pytania, na przykład:

W jaki sposób wyświetlić rysunki? Można tego dokonać przy pomocy metody write(), która rozpoznaje i interpretuje etykiety HTML. Wystarczy podać znaną z języka HTML etykietę określającą rysunek:

```
Document.write(„img src=
”rysunek.gif”);
```

Ponadto można wykorzystywać wszystkie etykiety służące do formatowania i definiowania tekstu lub grafiki.

Jaka jest różnica pomiędzy write() i writeln()? Obie metody wyświetlają tekst w taki sam sposób. Różnica polega na tym, że writeln() dodaje do tekstu znak końca linii. W przypadku tekstu niesformatowanego znak ten jest ignorowany przez przeglądarkę, jeśli jednak jest on poprzedzony etykietą <pre>, przeglądarka potraktuje go jako rozkaz rozpoczęcia nowego wiersza.

Co zrobić z przeglądarkami, które nie obsługują JavaScript? Niestety, do tekstu pierwszej części niniejszego cyklu wkradł się błąd. Użycie mylnie podanej sekwencji <!.....> nie rozwiązuje tego problemu. W celu ukrycia kodu JavaScript przed przeglądarkami, które go nie obsługują, należy oznaczyć cały skrypt jako komentarz HTML. Blok komentarza otwiera etykieta:

```
<!—komentarz,
```

zaś zamyka go linia:

```
//—>.
```

Znacznie więcej możliwości w tym zakresie oferuje JavaScript. Interesującym przykładem jest wbudowany w stronę WWW kalkulator, który wykonuje operacje na wprowadzanych liczbach. Pokażemy, jak zaprogramować prosty kalkulator, z którego będzie mógł skorzystać każdy odwiedzający naszą stronę gość.

Workshop w skrócie

CHIP 5/97: podstawy JavaScript, wyświetlanie tekstu na stronie WWW, funkcja daty
CHIP 6/97: funkcje, zmienne, formularze, funkcja eval()
CHIP 7/97: obliczenia naukowe z wykorzystaniem obiektu math
CHIP 8/97: praca z ramkami
CHIP 9/97: wykorzystanie cookies, rozważania nad przyszłością JavaScript

Najpierw formularz HTML

Autorzy stron WWW używają zwykle formularzy po to, by pobrać dane od użytkownika, a następnie przesłać je do serwera, gdzie zostaną przetworzone przy użyciu specjalnych programów. Dla przeciętnego internauty formularze te wydają się bezużyteczne, ponieważ HTML nie posiada żadnych wbudowanych funkcji do obróbki danych, a niewiele osób ma w domu własny serwer internetowy.

Kalkulator z polami do wpisywania wyrażenia arytmetycznego i wyświetlania wyników oraz przyciskiem „Oblicz” stworzymy na bazie formularza (patrz ramka „Krótki opis formularzy HTML”). Wartość wprowadzonego równania matematycznego obliczy funkcja eval() języka JavaScript.

Działanie programu jest następujące: użytkownik wpisuje w polu wejściowym wyrażenie typu „3 * (43 + 2)”. Po kliknięciu myszką na przycisku „Oblicz” w polu wynikowym powinna pojawić się wartość „135”. Za chwilę dowiemy się jak to działa.

Metoda eval()

Oferowana przez JavaScript metoda eval() oblicza wartość przesłanego do niej tekstowego wyrażenia matematycznego, na przykład „3 * (43 + 2)”. Przypomnijmy z pierwszej części: metoda jest funkcją przypisaną do określonego obiektu. W przypadku eval() wygląda to trochę inaczej – można ją określić jako metodę obiektu „aplikacja”, czyli po prostu przeglądarki WWW.

Aby przekazać funkcji eval() zawartość pola wejściowego, należy umieścić w nawiasie jego nazwę. Nie wystarczy jednak zapis: eval(Wyrażenie). Trzeba jeszcze – zgodnie z zasadami języka obiektowego – określić, do jakiego nadrzędnego obiektu należy pole „Wyrażenie”. W tym przypadku należy więc podać dodatkowo nazwę formularza „Kalkulator”. Przekaza-

▶ 17

Krótki opis formularzy HTML

Mianem formularza określa się obszar strony WWW, na którym znajdują się pola do wpisywania tekstu i/lub przyciski, pozwalające użytkownikowi na wprowadzanie danych.

Etykieta <form>

Etykiety <form> i </form> oznaczają odpowiednio początek i koniec formularza. Jeżeli na stronie znajduje się kilka formularzy, należy każdemu z nich nadać nazwę za pomocą właściwości name.

Etykieta <input>

<input> pozwala definiować pola do wprowadzania tekstu i przyciski w formularzu. Do najważniejszych właściwości etykiety <input> należą type, name, value i size.

Właściwość <type>

Type określa rodzaj pola formularza. Dla pola pozwalającego wprowadzić jedną linię tekstu należy mu nadać wartość „text". Jeśli type nadamy wartość „checkbox", wówczas utworzone zostanie pole kontrolne, które będzie można zaznaczać i odznaczać klikając na nim myszką. Z kolei wartość „button" powoduje stworzenie przycisku, z którym można powiązać określone wydarzenie. Możliwe są też inne wartości type. Kompletna lista znajduje się na serwerze WWW CHIP-a.

Właściwość name przypisuje polu nazwę, za pośrednictwem której można odwoływać się do tego pola. Właściwość value wstawia do pola typu text predefiniowany tekst. Jeśli pole jest typu button, wówczas value określa napis na przycisku. Właściwość size definiuje szerokość pola typu text.

Pole mieszczące 20 znaków i noszące nazwę „Wyrażenie" należy zdefiniować w następujący sposób:

```
<input type="text" name=
"Wyrażenie" size=20>
```

Przycisk z napisem „Oblicz" uzyskuje się wpisując:

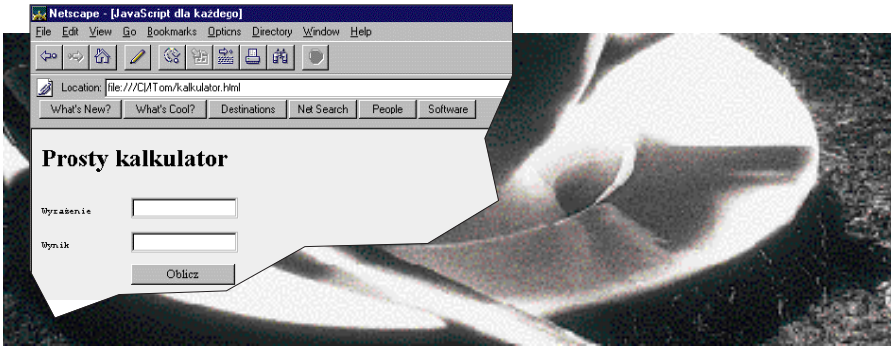
```
<input type="button" name=
"blabla" value="Oblicz">
```

Szerokość przycisku zależna jest od długości napisu. Właściwość size nie ma w tym przypadku żadnego znaczenia. Formularz symulujący prosty kalkulator można uzyskać w następujący sposób:

```
<form name="Kalkulator">
<pre>
Wyrażenie <input type="text"
name="Wyrażenie" size=20>
Wynik <input type="text" na-
me="Wynik" size=20>
<input type="button" name=
"blabla" value=" Oblicz ">
<pre>
</form>
```

Równomiernie rozłożenie poszczególnych obiektów formularza na stronie może ułatwić etykieta <pre>. Powoduje ona, że występujący po niej tekst traktowany jest jako tekst sformatowany. Oznacza to, że przeglądarka uwzględnia spacje, tabulatory, puste linie i znaki końca linii.

W sformatowanym tekście można posłużyć się znakami tabulacji, by pola wejściowe i wyjściowe umiejscowione były jedno pod drugim, niezależnie od długości nazw je poprzedzających. W linii, w której definiowany jest przycisk „Oblicz" tabulator jest pierwszym znakiem.



Może się przydać: za pomocą formularza HTML i kilku poleceń JavaScript można zaprogramować prosty kalkulator

ne metodzie eval() wyrażenia wpisanego w polu wejściowym formularza „Kalkulator" musi zatem mieć postać:

```
eval(Kalkulator.Wyrażenie.value)
```

Wynik operacji powinien pojawić się w polu wynikowym (Wynik). Oznacza to, że obliczona wartość musi być przekazana właściwości value (wartość) pola Wynik. Dokonuje się tego poleceniem: **Kalkulator.Wynik.value = eval(Kalkuator.Wyrażenie.value).**

Obsługa zdarzeń

Przeliczenie zadanego wyrażenia i przekazanie jego wartości do pola wynikowego powinno nastąpić po wykonaniu odpowiedniej czynności – w naszym przypadku po kliknięciu myszką na przycisku „Oblicz".

Do powiązania metody eval() z przyciskiem wykorzystuje się tzw. program obsługi zdarzeń (event handler), który kontroluje wszystkie wydarzenia zachodzące na stronie WWW. W zasadzie każdy obiekt posiada osobny program obsługi zdarzeń. Zdarzeniem jest na przykład kliknięcie myszką, a także pisanie lub modyfikacja tekstu w formularzu. Zdarzeniem jest również załadowanie strony.

Dzięki handlerom program napisany w JavaScript może reagować na określone zdarzenia. Jeśli na przykład chcemy sprawdzić, czy w pewnym polu tekstowym nie wprowadzono lub nie zmieniono tekstu, możemy posłużyć się handlerem onChange(). Kliknięcie na przycisku można wykryć przy pomocy onClick().

Zdefiniujmy uruchamiający wyliczenie wartości wyrażenia przycisk. Należy w tym celu umieścić w programie linie określające jego wygląd oraz funkcję onClick():

```
<input type="button"
 name="blabla"
 value=" Oblicz "
 onClick="Kalkulator.Wynik.value
= eval(Kalkulator.Wyrażenie.value)">;
```

Zdjęcie powyżej przedstawia kompletny listing kodu źródłowego strony WWW, który zawiera opisane elementy.

Dziwić może tak dużo spacji w określonym polem value opisie przycisku „Oblicz". Zostały one dodane jedynie ze względów kosmetycznych – dzięki nim przycisk ma szerokość równą szerokości pola tekstowego ponad nim. Liczbę spacji w nazwie trzeba dobrać doświadczalnie, metodą prób i błędów.

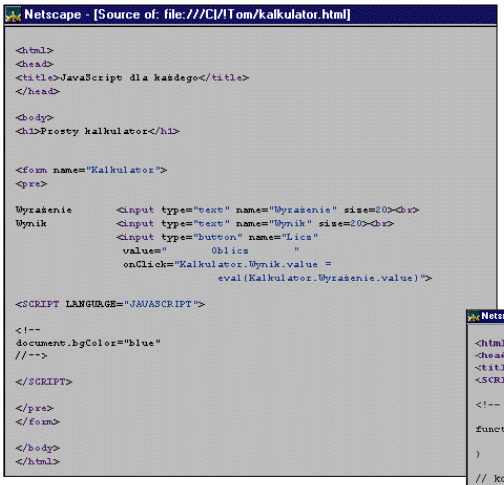
Własne funkcje ułatwiają programowanie

W celu uczynienia struktury programu bardziej przejrzystą, stosuje się technikę programowania znaną w niemal wszystkich językach – funkcje definiowane przez użytkownika. Dzięki nim sekwencję instrukcji obliczających wartość można zmieścić w jednej linii programu.

JavaScript posiada szereg wbudowanych funkcji, nazywanych w terminologii języków obiektowo zorientowanych metodami. Jedną z nich jest przedstawiona wcześniej metoda eval(), inną write(), która została opisana w pierwszej części kursu JavaScript. Programista JavaScript może również definiować własne funkcje. Definicja ma postać:

```
function nazwa_funkcji (argu-
ment1, argument2,...) {blok poleceń};
```

W nawiasie okrągłym podaje się (opcjonalnie) argumenty funkcji, zaś



Gotowy kalkulator: aby stworzyć formularz wystarczy kilka poleceń HTML. Komendy JavaScript onClick() i eval() przeliczają wyrażenie matematyczne po kliknieciu myszką

Eleganckie rozwiązanie: funkcję korzystającą z metody eval() umieszczono w nagłówku strony HTML. Słowo kluczowe this przekazuje handlerowi onClick() wskazanie bieżącego obiektu (Kalkulator)

w nawiasie klamrowym – realizowane przez nią polecenia. Zgodnie z konwencją przyjętą w JavaScript, małe i wielkie litery w nazwie są rozróżniane. Dopuszczalne są litery, cyfry oraz znak podkreślenia, przy czym nazwa musi rozpoczynać się literą. Funkcję można więc nazwać „Oblicz_1”, ale nazwy „1_Oblicz” i „_Oblicz1” są niepoprawne.

Chcąc wykorzystać metodę eval() w funkcji o nazwie Oblicz(), jako argument należy jej przekazać wartość wpisaną w polu wejściowym. W tym celu podczas definiowania funkcji określa się zmienną powiązaną z obiektem (patrz ramka „Zmienne”). Odpowiedni fragment programu ma postać:

```
function Oblicz(Obiekt)
{Obiekt.Wynik.value=
eval(Obiekt.Wyrażenie.value)};
```

Teraz trzeba jeszcze poinformować program obsługi zdarzeń onClick o nazwie funkcji, która zawiera pole wejściowe i wynikowe – w tym przypadku jest to formularz „Kalkulator”:

```
onClick=”oblicz(Kalkulator)”
```

Po kliknięciu myszką nazwa formularza (jako zmienna Obiekt) zostanie przekazana funkcji Oblicz(). Funkcja może wówczas przeprowadzić wszystkie wymagane operacje. Pobiera ona wyrażenie arytmetyczne z pola wejściowego, oblicza jego wartość przy pomocy metody eval() i wyświetla wynik w polu wynikowym.

Ostatnie pociągnięcia pędzla

W ostatecznej wersji programu uwagę zwracają dwa szczegóły: umieszczenie funkcji w nagłówku strony HTML i parametr this.form w wywołaniu funkcji Oblicz().

Poprawne umiejscowienie funkcji w tekście HTML wymaga wiedzy o tym, w jaki sposób przeglądarka przetwarza kod źródłowy. Wszystkie polecenia odczytywane są kolejno i od razu wykonywane. Oznacza to, że interpreter musi znać funkcję zdefiniowaną przez użytkownika zanim zostanie ona wczytana. Z tego względu zaleca się definiować funkcje w nagłówku strony WWW, a w każdym razie przed jej pierwszym wywołaniem. W przeciwnym przypadku przeglądarka wyświetli komunikat o błędzie.

Eleganckim rozwiązaniem jest umieszczenie wykorzystującej metodę eval() funkcji w nagłówku strony HTML. Dzięki słowu kluczowemu this program obsługi zdarzeń onClick przekazuje informacje do obsługiwanych obiektów.

Słowo this jest jednym z kluczowych słów języka JavaScript. Zawsze wskazuje ono na obiekt, który wywoływany jest przez metodę korzystającą z this. W naszym przykładzie this wskazuje na formularz i z tego względu musi być połączony ze zmienną form:

```
onClick=”Oblicz(this.form)”;
```

Przykład kalkulatora pokazuje nie tylko zastosowanie funkcji, formularzy, handlera zdarzeń onClick i metody eval(), ale przede wszystkim obrazuje możliwość wzajemnego uzupełniania się języków HTML i JavaScript. Dalsze przykłady współdziałania obu języków oraz więcej informacji na temat formularzy, pól tekstowych i programów obsługi zdarzeń będzie można znaleźć w kolejnej części warsztatu JavaScript. Pełna dokumentacja poleceń JavaScript dostępna jest także na serwerze WWW redakcji (http://www.chip.pl).

W następnym odcinku pokażemy, jak przy pomocy obiektu matematycznego można stworzyć rozbudowany kalkulator i umieścić go na stronie WWW.

oprac. Tomasz Czarnecki (jp)

Zmienne

W JavaScript występują cztery typy danych: liczby, łańcuchy tekstowe, wartości logiczne i wartość null.

Liczby mogą w zasadzie przyjmować dowolne wartości. Dopuszczalne są trzy systemy notacji:
System dziesiętny (podstawa 10) – powszechnie stosowane liczby całkowite bez zer na początku, na przykład 726
System ósemkowy (podstawa 8) – liczby całkowite z zerami na początku, na przykład 056
System szesnastkowy (podstawa 16) – liczby całkowite, przed którymi dodaje się przedrostek 0x lub 0X, na przykład 0x2F lub 0X2F.

Łańcuchy tekstowe są zmiennymi tekstowymi. Podawane są w cudzysłowie i mogą zawierać dowolne znaki alfanumeryczne.

Zmienne logiczne mogą w JavaScript przyjmować jedną z dwóch wartości: „true” lub „false”

Null daje w wyniku wartość zerową. Może być pustym ciągiem tekstowym, takim jak „”, albo liczbą o wartości 0. W przypadku null nie występuje żadna wartość, a zmienna tego typu nie jest definiowana. JavaScript zwraca wartość null, jeśli zostanie wciśnięty przycisk „Cancel”.

Deklarować zmienne? Przed pierwszym odwołaniem się do zmiennej należy ją zadeklarować, czyli poinformować system o jej istnieniu. Służy temu etykieta var:

```
var zmienna.
```

Jednocześnie można przypisać zmiennej wartość, np.:

```
var zmienna „Halo”.
```

Nie trzeba podawać typu zmiennej. JavaScript ustali go automatycznie na podstawie przypisanej jej wartości.
Deklaracja zmiennej za pomocą etykiety var nie jest konieczna. System sam rozpozna zmienną podczas pierwszego odwołania się do niej. Istnieje jednak wówczas niebezpieczeństwo utraty kontroli nad wprowadzonymi zmiennymi.

Uwaga

Dodatkowe informacje na temat języka JavaScript można znaleźć na CHIP-CD w dziale **Software | JavaScript**