Instrukcja warunkowa if

Instrukcji warunkowej używamy, gdy w zależności od tego, czy warunek zawarty w instrukcji jest prawdziwy lub fałszywy, wykonane zostają inne instrukcje.

Instrukcja if występuje w kilku wersjach, najprostsza z nich ma schematyczną postać:

1. **Instrukcja If …**

```
if (warunek)
{
 //instrukcje do wykonania
}
```

Każda pojedyncza instrukcja w bloku pomiędzy znakami nawiasu klamrowego musi się kończyć znakiem średnika, natomiast za instrukcją if średnik może, ale nie musi występować (z reguły jest pomijany).

W sytuacji, kiedy w bloku instrukcji if miałaby się znaleźć tylko jedna instrukcja dopuszczalne jest pominięcie znaków nawiasu klamrowego, jednak obligatoryjny jest wtedy średnik kończący:

```
if (warunek)
{
  instrukcja;
}
```

```
if (warunek)
  instrukcja;

lub

if (warunek) instrukcja;
```

Do skonstruowania warunku wykorzystywane są operatory relacyjne

```
… if (liczba < 0)
… if (liczba >= 0){
```

2. **Instrukcja If … Else…**

```
if (warunek){
  //instrukcje do wykonania, kiedy warunek jest prawdziwy
}
else{
  //instrukcje do wykonania, kiedy warunek jest fałszywy
}
```

3. **Instrukcja if...else if**

Trzecia wersja instrukcji if pozwala na badanie wielu warunków. Otóż po bloku if może wystąpić wiele dodatkowych bloków else if. Schematyczna postać takiej konstrukcji to:

```
if (warunek1){
  instrukcje1;
}
else if (warunek2){
  instrukcje2;
}
else if (warunek3){
```

```
instrukcje3;
}
...
```

Zagnieżdżanie instrukcji warunkowych

Zarówno w bloku if, jak i w bloku else mogą wystąpić dowolne instrukcje JavaScript. Oznacza to, że można tam umieścić kolejne instrukcje if, a zatem, że mogą być one zagnieżdżane.

```
if (warunek1)
{
   if (warunek2)
    {
      instrukcje1;
     }
   else
    {
      instrukcje2;
     }
 }
else
{
  if (warunek3)
   {
     instrukcje3;
   }
  else
     {
       instrukcje4;
     }
}
```

Wyrażenia warunkowe

Wyrażenia warunkowe, stosowane w instrukcji if, mogą być bardziej złożone, mogą się składać z wielu członów połączonych operatorami logicznymi.

```
if(liczba > 0)
{
  if(liczba < 10)
{
    document.write("Liczba
jest większa od 0 i
mniejsza od 10.");
  }
}
```

```
if((liczba > 0) &&
(liczba < 10)){

  document.write("Liczba
jest większa od 0 i
mniejsza od 10.");

}
```

Operatory relacyjne **>** i **<** mają większy priorytet (są silniejsze) niż operator iloczynu logicznego **&&.** Wynika z tego, że można pominąć nawiasy w wyrażeniu warunkowym, które równie dobrze może mieć postać:

```
liczba > 0 && liczba < 10
```

Jest to w pełni dopuszczalne, aczkolwiek niezalecane, gdyż zaciemnia kod i utrudnia jego analizę, szczególnie w przypadku bardziej złożonych wyrażeń.

```
(liczba > 0) && (liczba != 5) && (liczba != 10)) || (liczba == -8)
```

```
if(((liczba > 0) && (liczba != 5) && (liczba != 10)) || (liczba == -8))
```