Zasięg możemy określić jako miejsca, w których zmienna jest ważna i można się do niej bezpośrednio odwoływać. W dotychczas prezentowanych przykładach wiele zmiennych miało tzw. zasięg globalny, czyli można było z nich bezpośrednio korzystać w każdym miejscu skryptu. O tym, że tak jest w istocie, przekona nas skrypt zaprezentowany na listingu 2.31.

Listing 2.31. Globalny zasięg zmiennej

```
<script type=&quot;text/javascript&quot;>

var liczba = 1;

function func()

{

  document.write(&quot;Wartość zmiennej liczba w funkcji func to &quot;);

  document.write(liczba + &quot;<br >&quot;);

}

func();

document.write(&quot;Wartość zmiennej liczba poza funkcją func to &quot;);

document.write(liczba + &quot;<br >&quot;);

</script>
```

Kiedy go uruchomimy, w przeglądarce zobaczymy dwa napisy podające wartość zmiennej liczba. Warto zwrócić uwagę, że na przykład w bardzo popularnym PHP jest inaczej. W PHP zmienna globalna nie jest bezpośrednio dostępna wewnątrz funkcji i aby się do niej odwołać, trzeba stosować dodatkowe konstrukcje programistyczne. W języku JavaScript, podobnie jak w wielu klasycznych językach programowania (jak C i C++), nie ma z tym problemu — zmienne globalne są dostępne w całym skrypcie.

Inaczej jest, jeśli zmienna zostanie zdefiniowana wewnątrz funkcji za pomocą słowa var — nazywamy ją wtedy zmienną lokalną funkcji. Zasięg takiej zmiennej jest ograniczony tylko do wnętrza funkcji, w której została zdefiniowana, a w pozostałych miejscach skryptu nie można się do niej odwoływać, co ilustruje kod widoczny na listingu 2.32.

Listing 2.32. Zasięg zmiennej lokalnej

```
<script type=&quot;text/javascript&quot;>

function func()

{

  var liczba = 1;

  document.write(&quot;Wartość zmiennej liczba w funkcji func to &quot;);

  document.write(liczba + &quot;<br >&quot;);
```

```
}

func();

document.write(&quot;Wartość zmiennej liczba poza funkcją func  to &quot;);

document.write(liczba + &quot;<br  >&quot;);

</script>
```

Tym razem po uruchomieniu kodu w drugim napisie nie pojawi się nazwa zmiennej (rysunek 2.7).

Rysunek 2.7. Efekt braku dostępu do zmiennej lokalnej

Jeśli używamy przeglądarki udostępniającej konsolę JavaScript (np. Firefox, Nestcape Navigator, Opera), po jej uruchomieniu zobaczymy komunikat o błędzie, informujący o niezdefiniowanej zmiennej. Widać więc wyraźnie, że zmienne lokalne faktycznie dostępne są jedynie w obrębie funkcji, w której zostały zdefiniowane.

Co się jednak stanie w sytuacji, kiedy wewnątrz funkcji zdefiniujemy zmienną o takiej samej nazwie jak istniejąca zmienna globalna? Taka sytuacja jest jak najbardziej możliwa. Zmienna lokalna przesłoni wtedy globalną, a zatem wewnątrz funkcji będziemy się wtedy mogli odwoływać do zmiennej lokalnej, a na zewnątrz do globalnej, co ilustruje skrypt z listingu 2.33.

Listing 2.33. Przesłanianie zmiennych

```
<script  type=&quot;text/javascript&quot;>

var liczba = 10;

function func()

{

  var liczba = 20;

  document.write(&quot;Wartość zmiennej liczba w funkcji func  to &quot;);

  document.write(liczba + &quot;<br >&quot;);

}

func();

document.write(&quot;Wartość zmiennej liczba poza funkcją func  to &quot;);

document.write(liczba + &quot;<br  >&quot;);

</script>
```

Po jego uruchomieniu zobaczymy dwa napisy. Pierwszy będzie wynikiem wywołania funkcji func i będzie informował, że stan zmiennej liczba to 20. Jest tak dlatego, że w funkcji func mamy do czynienia ze zmienną

lokalną tej funkcji, która jest dostępna tylko wewnątrz niej. Drugi napis poinformuje nas, że zmienna liczba ma wartość 10 — oczywiście ta informacja będzie odnosiła się do zmiennej globalnej (zdefiniowanej poza funkcją).

Zwróćmy jednak szczególną uwagę na podaną wyżej definicję zmiennej lokalnej. Musi ona spełniać dwa warunki.

1.

Jej definicja musi znajdować się wewnątrz funkcji.

2.

W definicji musi zostać użyte słowo var.

To oznacza, że jeżeli w funkcji w pierwszym odwołaniu do jakiejś zmiennej nie użyjemy słowa var, to zostanie to potraktowane jak odwołanie do zmiennej globalnej. Co więcej, jeżeli taka zmienna jeszcze nie istnieje, to zostanie utworzona! Ilustruje to fragment kodu widoczny na listingu 2.34 (ważne wiersze zostały ponumerowane).

Listing 2.34. Zmienna globalna tworzona wewnątrz funkcji

```
<script type=&quot;text/javascript&quot;>

var liczba1 = 10;//1

function func()

{

  liczba1 = 20;//3

  liczba2 = 30;//4

}

func();//2

liczba2 = 50;//5

</script>
```

Instrukcja oznaczona numerem 1 tworzy zmienną liczba1 i przypisuje jej wartość 10. Zostało użyte słowo var, ale zmienna znajduje się poza funkcjami (czyli jest deklarowana w zasięgu globalnym), jest zatem zmienną globalną. Instrukcja oznaczona jako 2 wywołuje funkcję o nazwie func. W tejże funkcji znajduje się instrukcja 3, która modyfikuje wartość istniejącej globalnej zmiennej liczba1, oraz instrukcja 4, która modyfikuje wartość globalnej zmiennej o nazwie liczba2. Ponieważ zmienna liczba2 nie została wcześniej utworzona, powstanie w momencie wykonania tej instrukcji i będzie dostępna nawet po zakończeniu działania funkcji (jest przecież zmienną globalną). Można się o tym przekonać, wykonując instrukcję 5, która modyfikuje wartość globalnej zmiennej liczba2 (utworzonej w funkcji func).

Wystarczy dodać do tego kodu instrukcje document.write wyświetlające zawartości zmiennych na każdym etapie wykonania kodu, aby przekonać się, że powyższy opis jest prawdziwy.