

Statements

Method Summary

void [break](#)()
void [const](#)()
void [continue](#)()
void [do while](#)()
loop
void [for](#)()
loop
void [for each in](#)()
loop
void [if](#)()
void [if else](#)()
void [label](#)()
void Provides a statement with an identifier that you can refer to using a break or continue statement.
void [switch](#)()
void [try catch](#)()
void [try catch finally](#)()
void [var](#)()
void [while](#)()
loop

Method Details

[break](#)
void **break**()
Replace with description

Returns
void

Sample

```
break
```

[const](#)
void **const**()
Replace with description

Returns
void

Sample

```
const #;
```

[continue](#)
void **continue**()
Replace with description

Returns
void

Sample

```
continue
```

[do while](#)
void **do while**()
loop
Returns
void

Sample

```
do
{
}
while ( # )
```

for

void **for**()

loop

Returns

void

Sample

```
for ( var i = 0 ; i < # ; i++ )
{
}
```

for each in

void **for each in**()

loop

Returns

void

Sample

```
for ( var item in obj )
{
}
```

if

void **if**()

Replace with description

Returns

void

Sample

```
if ( # )
{
}
```

if else

void **if else**()

Replace with description

Returns

void

Sample

```
if ( # )
{
}
else
{
}
```

label

void **label**()

Provides a statement with an identifier that you can refer to using a break or continue statement.

For example, you can use a label to identify a loop, and then use the break or continue statements to indicate

whether a program should interrupt the loop or continue its execution.

Returns

void

Sample

```
var i = 0, j;  
outer_loop: while (i < 10) {  
    i++;  
    j = 0;  
    while (j < 10) {  
        j++;  
        if (j > i) continue outer_loop;  
        application.output("i=" + i + ", j=" + j);  
    }  
}
```

switch

void **switch()**

Replace with description

Returns

void

Sample

```
switch( # )  
{  
    case:  
    default:  
}
```

try catch

void **try catch()**

Replace with description

Returns

void

Sample

```
try  
{  
    #  
}  
catch(#)  
{  
    #  
}
```

try catch finally

void **try catch finally()**

Replace with description

Returns

void

Sample

```
try  
{  
    #  
}  
catch(#)  
{  
    #  
} finally  
{  
    #  
}
```

var

void **var**()

Replace with description

Returns

void

Sample

```
var #;
```

while

void **while**()

loop

Returns

void

Sample

```
while ( # )  
{  
    #  
}
```