

자바스크립트 코어

Core - Javascript

# 문장

Statements





바보무

Loop

# 자바스크립트의 반복문은 3가지! 반복 수행되어야 할 일을 처리하는 명령문



계속 달리기만 하면  
되는거지요? 헉헉...

헉..헉...헉...

while문

# 자바스크립트 첫번째 반복문 while

(조건이 참인) ~하는 동안 반복수행

죄를 뉘우칠 때까지  
계속 기도하세요.



무얼 봐...

# 조건이 참(true)인 동안(while) { ... }을 수행하라

```
var count = 0; //초기 값
```

```
while ( count <= 15 )
```

조건이 항상 참이어서 무한반복 된다!

```
document.write("<p>" + count + "은 15보다 작습니다</p>");
```

```
count++;
```

반복되는 동안 설정 값에 변화를 준다.

```
}
```

```
function cleanWhiteSpace(element) {  
  element = element || document;  
  var cur = element.firstChild;  
  while ( cur != null ) {  
    if( cur.nodeType === 3 && !/S/.test(cur.nodeValue) ) {  
      element.removeChild(cur);  
    } else if( cur.nodeType === 1) {  
      cleanWhiteSpace(cur);  
    }  
    cur = cur.nextSibling;  
  }  
}
```

요소가 비어있지 않으면 반복 수행

!

오! 함수와 문장의 조합이 이리도  
강력해지는군요!!!





# do while문



자바스크립트 두번째 반복문 do, while  
1회 우선 실행 후, (조건이 참인) ~하는 동안 반복수행

우선 한대 때리죠!  
그래도 죄를 누우치지 않으면  
계속 때릴테요.



# 조건과 상관없이 먼저 한번 실행(do)하고나서, 주어진 조건(while)을 확인한 후 참이면 실행하라

```
var count = 22; //초기 값
```

```
do {
```

조건과 상관없이 한번은 무조건 실행

```
    document.write("<p>count는 " + count + "입니다.</p>");
```

```
    count++;
```

```
} while ( count <= 15 )
```

조건이 거짓이므로 더이상 실행되지 않는다.

```
function prev(element) {  
  if(element.previousElementSibling) {  
    element = element.previousElementSibling;  
  } else {  
    do {  
      element = element.previousSibling;  
    } while (element && element.nodeType !== 1);  
  }  
  return element;  
};
```



오! 함수와 문장의 조합이 이리도  
강력해지는군요!!!



```
function next(element) {  
  if(element.nextElementSibling) {  
    element = element.nextElementSibling;  
  } else {  
    do {  
      element = element.nextSibling;  
    } while (element && element.nodeType !== 1);  
  }  
  return element;  
};
```



오! 함수와 문장의 조합이 이리도  
강력해지는군요!!!



```
function firstChild(element) {  
    el = element.firstChild || element.firstChild;  
    return (el && el.nodeType !== 1) ? next(el) : el;  
}
```

```
function lastChild(element) {  
    el = element.lastElementChild || element.lastChild;  
    return (el && el.nodeType !== 1) ? prev(el) : el;  
}
```



오! 함수와 문장의 조합이 이리도  
강력해지는군요!!!





# for문



# 자바스크립트 세번째 반복문 for 가장 일반적으로 많이 사용되는 반복수행문

앞서 기도하고,  
때렸던 모든 일을 앞으로  
평생 기억하시오!



그만 바라봐!



# for문은 편리하게 사용할 수 있는 반복문

초기값

조건 테스트

변화값


```
for ( var count = 0; count <= 15; count++ ) {  
    document.write("<p>count는 ' +count + '입니다.</p>");  
}
```

## for문의 실행 흐름 순서

```
for ( var count = 0; count <= 15; count++ ) {  
    document.write("<p>count는 " + count + "입니다.</p>");  
}
```

1. 초기값
2. 조건(참이면 계속, 거짓이면 그만!)
3. { } 문장 수행
4. 변수값 변화

# continue는 조건이 참이면 건너뛸 수 있다

```
var Apple = ['apple', 'fruits', true, 2010];  
var elements_in_Apple = [ ];  
for ( var i = 0; i < Apple.length; i++ ) {  
    if (typeof Apple[i] == 'boolean') continue;   
    elements_in_Apple.push(Apple[i]);  
    document.write(elements_in_Apple[i]+'<br />');  
}
```

# break는 조건이 참이면 더이상 실행 안한다

```
var Apple = ['apple','fruits',2010,true];  
var elements_in_Apple = [ ];  
for ( var i = 0; i < Apple.length; i++ ) {  
    if (typeof Apple[i] == 'boolean') break;  
    elements_in_Apple.push(Apple[i]);  
    document.write(elements_in_Apple[i]+'<br />');  
}
```

```
function parent(element, count) {  
    count = count || 1;  
    for ( var i=0; i<count; i++ ) {  
        if(element != null) {  
            element = element.parentNode;  
        }  
    }  
    return element;  
}
```



오! 함수와 문장의 조합이 이리도  
강력해지는군요!!!



# for in 문

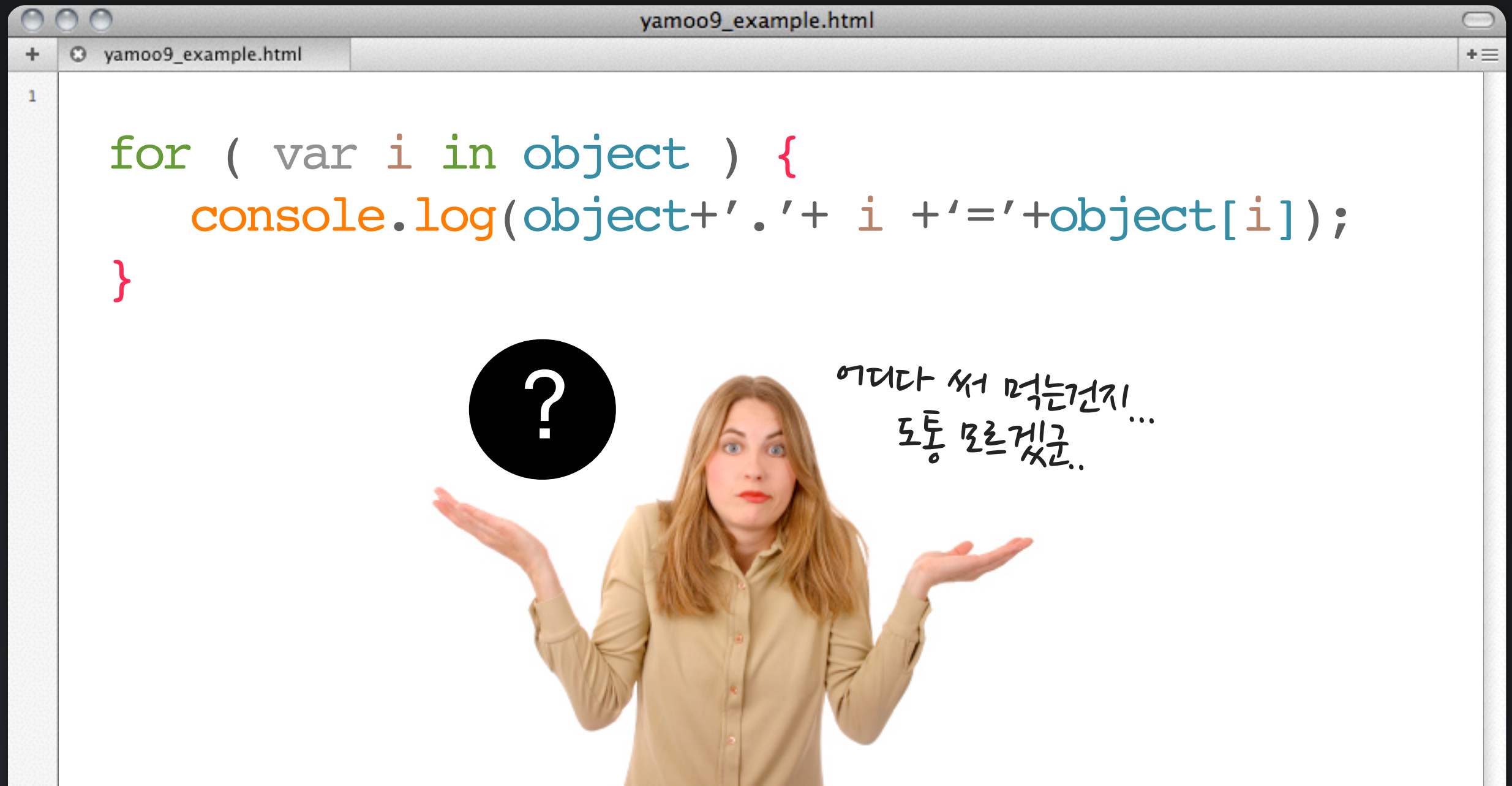


내 안에  
너 있다!



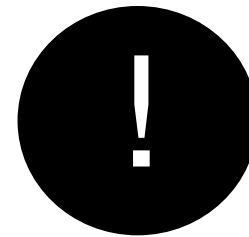


# for~in문은 객체의 내부 속성을 순환하여 처리.





```
function getEnabledStyles(element) {  
    if(typeof element !== 'object') return false;  
    var style_list = [], styles;  
    for(i in element.style) {  
        var styles = element.style[i];  
        if(styles !== '' && styles !== null &&  
typeof styles !== 'function') {  
            style_list.push(i);  
        };  
    };  
    return style_list;  
};
```



오! 함수와 문장의 조합이 이리도  
강력해지는 군요!!!



while, for statement	description
while(condition) {...}	(조건)이 참인 동안 {...} 코드블럭 실행
do {...} while(condition)	조건과 상관없이 한번은 반드시 실행 (조건)이 참인 동안 {...} 코드블럭 실행
for(초기값;조건;변화량) {...}	(조건)이 참인 동안 {...} 코드블럭 실행
continue	continue를 만나면 아래 코드 생략, 조건 확인후 다시 실행