JavaScript Strings

- description : JavaScript Strings
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-04-23

The source of this article

JavaScript Strings
JavaScript string은 텍스트를 저장하고 조작하는데 사용됩니다.

JavaScript Strings

JavaScript 문자열은 따옴표 안에 쓰여진 0 개 이상의 문자입니다.

```
var x = "John Doe";
```

작은 따옴표나 큰 따옴표를 사용할 수 있습니다.

```
var carName1 = "Volvo XC60";  // Double quotes
var carName2 = 'Volvo XC60';  // Single quotes
```

문자열을 둘러싼 따옴표와 일치하지 않는 한, 문자열 안에 따옴표를 사용할 수 있습니다.

```
var answer1 = "It's alright";
var answer2 = "He is called 'Johnny'";
var answer3 = 'He is called "Johnny"';
```

String Length

문자열의 길이를 찾으려면, 내장 `length` 속성을 사용하십시오.

```
var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
var sln = txt.length;
```

Escape Character

문자열은 따옴표로 묶어야 하므로, JavaScript는 아래 예제의 문자열을 오해합니다.

```
var x = "We are the so-called "Vikings" from the north.";
```

상기 예제에서 문자열은 “We are the so-called”까지 잘립니다.

이 문제를 피하는 해결책은 백슬래시 이스케이프 문자(backslash escape character)를 사용하는 것입니다.

백슬래시 (\) 이스케이프 문자는 특수 문자를 문자열 문자로 바꿉니다.

code	**Result**	**Description**
\'	'	Single quote (작은 따옴표)
\"	"	Double quote (큰 따옴표)
\\	\	Backslash (백슬래시, 역슬래시)

\" 시퀀스(sequence)는 문자열에 큰 따옴표를 삽입합니다.

```
var x = "We are the so-called \"Vikings\" from the north.";
```

\' 시퀀스(sequence)는 문자열에 작은 따옴표를 삽입합니다.

```
var x = 'It\'s alright.';
```

\\ 시퀀스(sequence)는 문자열에 백슬래시(역슬래시, backslash)를 삽입합니다.

```
var x = "The character \\ is called backslash.";
```

JavaScript에서 유효한 다른 6개의 이스케이프 시퀀스
\\^ Code ^ Result ^

\b	백스페이스
\f	폼 피드
\n	줄 바꿈
\r	캐리지 리턴
\t	수평 탭(태블레이터)
\v	수직 탭(테블레이터)

위의 6 개의 이스케이프 문자는 원래 타자기, 텔레타이프 및 팩스기를 제어하기 위해 설계되었습니다.

HTML에서는 의미가 없습니다.

Breaking Long Code Line

가독성을 높이기 위해 프로그래머는 종종 80 자보다 긴 코드 줄을 피하는 것을 좋아합니다.

JavaScript 스테이트먼트가 한 줄에 맞지 않는 경우, 줄바꿈하는 가장 좋은 위치는 연산자 뒤 입니다.

```
document.getElementById("demo").innerHTML =
"Hello Dolly!";
```

백슬래시 하나로 **텍스트 문자열 내의** 코드 줄을 줄바꿈할 수도 있습니다.

```
document.getElementById("demo").innerHTML = "Hello \
Dolly!";
```

''%%\%%'' 메서드는 선호되는 방법이 아닙니다. 보편적인 지원이 없을 수도 있습니다.
일부 브라우저는 \ 문자 뒤에 공백을 허용하지 않습니다.

문자열을 줄바꿈하는 더 안전한 방법은 문자열 추가를 사용하는 것입니다.

```
document.getElementById("demo").innerHTML = "Hello " +
"Dolly!";
```

백슬래시로 코드 줄을 줄바꿈할 수 없습니다.

```
document.getElementById("demo").innerHTML = \
"Hello Dolly!";
```

Strings Can be Objects

일반적으로, JavaScript 문자열은 리터럴(literals)에서 생성된 원시값(primitive values)입니다:

```
var firstName = "John";
```

그러나 new 키워드를 사용하여 문자열을 오브젝트로 정의할 수도 있습니다.

```
var firstName = new String("John");
```

```
var x = "John";
var y = new String("John");
```

```
// typeof x will return string
// typeof y will return object
```

문자열을 오브젝트로 만들지 마십시오. 실행 속도가 느려집니다.
''new'' 키워드는 코드를 복잡하게 만듭니다. 이로 인해 예상치 못한 결과가 발생할 수 있습니다.

== 연산자를 사용할 경우, 동일한 문자열은 동일합니다.

```
var x = "John";
var y = new String("John");

// (x == y) is true because x and y have equal values
```

=== 연산자를 사용할 때, === 연산자는 데이터 유형과 값이 모두 같을 것으로 예상하기 때문에
같은 값이 같지 않을 수 있고,

```
var x = "John";
var y = new String("John");

// (x === y) is false because x and y have different types (string and
object)
```

또는 더 나쁠 수 있습니다. 오브젝트는 비교할 수 없습니다.

```
var x = new String("John");
var y = new String("John");

// (x == y) is false because x and y are objects
```

```
var x = new String("John");
var y = new String("John");

// (x === y) is false because x and y are objects
```

''%%(x == y)%%''와 ''%%(x === y)%%''의 차이점에 유의하십시오.
또한 두 %%JavaScript%% 오브젝트를 비교하면 항상 ''false''가 반환됩니다.

오션, Javascript, Strings

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다.** - 2023.12

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:javascript:javascript_note:js_strings&rev=1619142041

Last update: **2022/03/10 19:52**