String.prototype.substr()

The **`substr()`** method returns the characters in a string beginning at the specified location through the specified number of characters.

Syntax

```
str.substr(start[, length])
```

Parameters

`start`

Location at which to begin extracting characters. If a negative number is given, it is treated as `strLength + start` where `strLength` is the length of the string (for example, if `start` is -3 it is treated as `strLength - 3`.)

`length`

Optional. The number of characters to extract.

Description

`start` is a character index. The index of the first character is 0, and the index of the last character is 1 less than the length of the string. `substr()` begins extracting characters at `start` and collects `length` characters (unless it reaches the end of the string first, in which case it will return fewer).

If `start` is positive and is greater than or equal to the length of the string, `substr()` returns an empty string.

If `start` is negative, `substr()` uses it as a character index from the end of the string. If `start` is negative and `abs(start)` is larger than the length of the string, `substr()` uses 0 as the start index. Note: the described handling of negative values of the `start` argument is not supported by Microsoft JScript.

If `length` is 0 or negative, `substr()` returns an empty string. If `length` is omitted, `substr()` extracts characters to the end of the string.

Examples

Using `substr()`

```
var str = 'abcdefghij';

console.log('(1, 2): '   + str.substr(1, 2));   // '(1, 2): bc'
console.log('(-3, 2): '  + str.substr(-3, 2));  // '(-3, 2): hi'
console.log('(-3): '     + str.substr(-3));     // '(-3): hij'
console.log('(1): '      + str.substr(1));      // '(1): bcdefghij'
console.log('(-20, 2): ' + str.substr(-20, 2)); // '(-20, 2): ab'
console.log('(20, 2): '  + str.substr(20, 2));  // '(20, 2): '
```

Polyfill

Microsoft's JScript does not support negative values for the start index. If you wish to make use of this feature, you can use the following compatibility code to work around this bug:

```
// only run when the substr() function is broken
if ('ab'.substr(-1) != 'b') {
  /**
   *  Get the substring of a string
   *  @param  {integer}  start   where to start the substring
   *  @param  {integer}  length  how many characters to return
   *  @return {string}
   */
```

```
  String.prototype.substr = function(substr) {
    return function(start, length) {
      // call the original method
      return substr.call(this,
        // did we get a negative start, calculate how much it is from the beginning of the string
        // adjust the start parameter for negative value
        start < 0 ? this.length + start : start,
        length)
    }
  }(String.prototype.substr);
}
```

Specifications

Specification	Status	Comment
ECMAScript 3rd Edition (ECMA-262)	Standard	Defined in the (informative) Compatibility Annex B. Implemented in JavaScript 1.0.
ECMAScript 5.1 (ECMA-262) The definition of 'String.prototype.substr' in that specification.	Standard	Defined in the (informative) Compatibility Annex B
ECMAScript 2015 (6th Edition, ECMA-262) The definition of 'String.prototype.substr' in that specification.	Standard	Defined in the (normative) Annex B for Additional ECMAScript Features for Web Browsers

Browser compatibility

Desktop Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari
Basic support	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)

See also

- `String.prototype.slice()`
- `String.prototype.substring()`