بسم اللّه الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

متد های رشته ای / String methods در جاوا اسکریپت

مدرس: مهندس افشین رفوآ

آدرس آموزشگاه. تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 561 - واحد 7
88146323 - 88446780 - 88146330

متد های رشته ای / String methods در جاوا اسکریپت

متدهای رشته ای در کار با رشته ها به شما کمک می کنند.

یافتن یک رشته داخل رشته ی دیگر

متد indexOf() اولین مکان رخداد نوشته ی معین را در رشته برمی گرداند، در واقع تابع بیان شده شماره / اندیس مکان قرار گیری اولین نمونه یک حرف یا کلمه را در یک متغیر متنی باز می گرداند.

مثال

```
var str = "Please locate where 'locate' occurs!";
var pos = str.indexOf("locate");
```

متد lastIndexOf() نیز شماره ی آخرین رخداد / اندیس یا شماره ی مکان قرار گیری آخرین نمونه ی یک متن معین را در یک رشته باز می گرداند.

مثال

```
var str = "Please locate where 'locate' occurs!";
var pos = str.lastIndexOf("locate");
```

هر دو متد نام برده در صورت یافت نشدن متن مورد نظر، -1 را برمی گردانند.

توجه

جاوا شمارش موقعیت ها را از صفر آغاز می کند، بدین معنا که 0 اولین موقعیت در یک رشته محسوب می شود و 1 دومین و غیره....

هر دو متد بیان شده یک پارامتر دوم به عنوان مکان شروع جستجو می پذیرند.

متد lastIndexOf()، بجای آغاز جستجو از ابتدای رشته این کار را از انتهای آن آغاز می کند.

جستجو برای یک رشته داخل رشته ی دیگر

تابع search() در یک رشته به دنبال مقدار مشخص شده گشته و موقعیت یا مکان مقدار مورد نظر را در صورت یافتن آن، برمی گرداند، به عبارت دیگر از این متد برای جستجو یک حرف یا کلمه خاص در یک متغیر متنی استفاده می شود.

مثال

```
var str = "Please locate where 'locate' occurs!";
var pos = str.search("locate");
```

توابع indexOf() و search() هر دو یکسان بوده و کاربردی مشابه دارند.

هر دو آرگومان ها (پارامتر ها) یکسان پذیرفته و به دنبال آن مقادیر یکسان باز می گردانند.

با اینکه هر دو متد یکسان هستند، تابع search() (powerful search values) را می تواند بگیرد.

استخراج (بدست آوردن) بخش هایی از یک رشته

برای این منظور سه متد مختلف وجود دارد که به شرح زیر می باشند.

slice(start، end)

substring(start، end)

substr(start، length)

متد slice ()

این تابع بخشی از یک رشته را برش داده و استخراج می کند، سپس بخش استخراج شده را در رشته ی جدید باز می گرداند (به منظور جستجو یک حرف یا کلمه خاص در یک متغیر متنی استفاده می شود).

متد بالا دو پارامتر می گیرد : اندیس آغازین (اولین مکان قرار گیری)، اندیس پایانی (آخرین مکان قرار گیری).

مثال زیر بخشی از یک رشته که از موقعیت 7 تا 13 را به خود تخصیص داده (ادامه دارد) برش داده و استخراج می گرداند.

مثال

```
var str = "Apple، Banana، Kiwi";
var res = str.slice(7،13);
```

نتیجه

Banana

چنانچه پارامتر داده شده به تابع منفی بود، در آن صورت شمارش مکان (موقعیت قرارگیری نمونن) از پایان یا انتهای رشته آغاز می گردد.

مثال

```
var str = "Apple، Banana، Kiwi";
var res = str.slice(-12،-6);
```

نتیجه

Banana

در صورت حذف کردن پارامتر دوم، متد باقی رشته را برش داده و استخراج می گرداند.

مثال

```
var res = str.slice(7);
```

مثال

```
var res = str.slice(-12);
```

متد () substring

از تابع ذکر شده جهت برش تعداد مشخصی از کاراکترهای یک متغیر متنی بین دو نقطه معین استفاده می شود.

متد **substring()** درست مشابه slice() است، با این تفاوت که substring() قادر به پذیرفتن ایندکس منفی نیست.

مثال

```
var str = "Apple، Banana، Kiwi";
var res = str.substring(7،13);
```

نتیجه

Banana

اگر دومین پارامتر را حذف کنید، متد مزبور بقیه ی رشته ی مورد نظر را برمی گرداند.

متد substr()

از این تابع به منظور برش تعداد معینی از کاراکترهای یک متغیر متنی استفاده می شود.

substr() مشابه متد **slice()** عمل می کند، با این تفاوت که دومین پارامتر **طول** (تعداد کاراکترهای) بخش استخراج شده از متغیر متنی را مشخص می کند.

مثال

```
var str = "Apple، Banana، Kiwi";
var res = str.substr(7،6);
```

نتیجه

Banana

چنانچه اولین پارامتر منفی بود، شمارش (موقعیت) مکان قرار گیری از انتهای رشته آغاز می شود.

دومین پارامتر نمی تواند منفی باشد زیرا که طول رشته را تعیین می کند.

اگر دومین پارامتر را حذف کنید، substr() بقیه ی رشته را برش داده و استخراج می کند.

جایگزینی در محتوای یک رشته

تابع **replace ()** یک مقدار را در رشته جایگرین مقدار دیگری می کند.(جهت جایگزینی یک حرف یا کلمه خاص در یک متغیر متنی و جایگزینی آن با یک مقدار جدید بکار می رود)

مثال :

```
str = "Please visit Microsoft!";
var n = str.replace("Microsoft","W3Schools");
```

متد ذکر شده همچنین می تواند یک عبارت باقاعده (regular expression) به عنوان search value بگیرد.

تبدیل و نمایش نوشته یا متن رشته با حروف کوچک و بزرگ

toUpperCase() : از این تابع به منظور نمایش متن یک متغیر رشته ای با حروف بزرگ استفاده می شود.

مثال

```
var text1 = "Hello World!";       // String
var text2 = text1.toUpperCase();  // text2 is text1 converted to upper
```

toLowerCase() : جهت نمایش نوشته و متن یک متغیر رشته ای با حروف کوچک بکار می رود.

مثال

```
var text1 = "Hello World!";       // String
var text2 = text1.toLowerCase();  // text2 is text1 converted to lower
```

متد concat()

از تابع concat () برای چسباندن یا متصل کردن و همچنین اضافه کردن دو یا چند متغیر رشته ای به هم کمک گرفته می شود.

مثال

```
var text1 = "Hello";
var text2 = "World";
text3 = text1.concat(" ",text2);
```

متد concat() را می توان بجای عملگر " + " برای متصل کردن دو یا چند رشته بکار برد. دو خط کد زیر هر دو یک کار را انجام می دهند.

مثال

```
var text = "Hello" + " " + "World!";
var text = "Hello".concat(" "،"World!");
```

نکته

کلیه ی متدهای رشته ای (string method) در نتیجه یک رشته ی جدید برمی گردانند و رشته ی اصلی را به هیچ عنوان اصلاح نمی کنند (تغییر نمی دهند). به عبارتی دیگر رشته ها تغییر ناپذیر هستند، از این رو نمی توان آن ها را تغییر داد و فقط می توان آن ها را جایگزین کرد.

استخراج (بدست آوردن) و نمایش کاراکترهای یک رشته

دو متد وجود دارد که با استفاده از آن ها می توان کاراکترهای یک رشته را استخراج کرد(و نمایش داد)

charAt(position) : به منظور نمایش مقدار یک کاراکتر مورد نظر در یک متغیر رشته ای مورد استفاده قرار می گیرد

charCodeAt(position) : جهت نمایش کد اسکی یک کاراکتر مورد نظر در یک متغیر رشته ای بکار می رود

متد charAt()

این متد برای برگرداندن مقدار یک کاراکتر در متغیر رشته ای بکار گرفته می شود.

مثال

```
 var str = "HELLO WORLD";
str.charAt(0);            // returns H
```

متد charCodeAt()

این تابع در واقع Unicode کاراکتر مورد نظر را در یک متغیر رشته ای برمی گردانند.

مثال

```
var str = "HELLO WORLD";
```

```
str.charCodeAt(0);          // returns 72
```

دسترسی به یک رشته مانند آرایه ناامن می باشد. ممکن است با کدهایی مانند مثال زیر برخورد کرده باشید.

```
var str = "HELLO WORLD";

str[0];                     // returns H
```

این روش دسترسی به رشته نا امن بوده و ممکن است نتایج غیرقابل پیشبینی برگرداند.

در کلیه ی مرورگرها کار نخواهد کرد (IE5، IE6، IE7)

باعث می شود رشته ها مانند آرایه دیده شوند (درحالی که اصلا نباید چنین باشد)

str[0] = "H" اگرچه باعث صادر شدن خطا یا error نمی شود در عین حال کارساز نخواهد بود

بدین ترتیب اگر مایلید یک رشته را مثل (به عنوان) آرایه مورد استفاده قرار دهید یا بخوانید، باید حتما آن را به آرایه تبدیل کنید.

تبدیل رشته به یک آرایه

تبدیل یک رشته به آرایه با استفاده از متد split() امکان پذیر می باشد.

مثال

```
var txt = "a‹b‹c‹d‹e";   // String
txt.split("‹");          // Split on commas
txt.split(" ");          // Split on spaces
txt.split("|");          // Split on pipe
```

حال اگر تفکیک کننده (separator) را حذف کنید، آرایه ی برمی گرداند که از ایندکس 0 آغاز می شود.

در صورتی که تفکیک گر ("") باشد، آرایه ی برگردانده شده آرایه ی متشکل از تک کاراکترها خواهد بود.

مثال

```
var txt = "Hello";       // String
txt.split("");           // Split in characters
```