JS字符串补全方法padStart()和padEnd()简介

这篇文章发布于 2018年07月24日，星期二，00:29，归类于 JS API。 阅读 7653 次, 今日 15 次 16 条评论

一、关于字符串补全

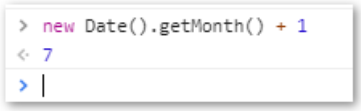
在JS中，字符串补全是常用操作，用的比较多的就是时间或者日期前面的补`0`。

例如，日期，我们多采用4-2-2的表示形式，例如：

```
2018-07-23
```

当我们使用时间戳进行月份获取的时候，是没有前面的`0`的，例如：

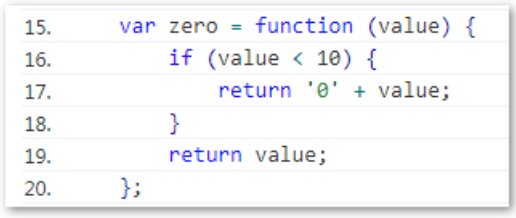
```
var month = new Date().getMonth() + 1;    // 结果是7
```



此时，就需要进行补全，通常做法是这样：

```
if (month < 10) {
    month = '0' + month;
}
```

甚至会专门定义一个补`'0'`方法，例如此日期转时间戳微码中的自定义的`zero`方法。



```
15.     var zero = function (value) {
16.         if (value < 10) {
17.             return '0' + value;
18.         }
19.         return value;
20.     };
```

然而，随着JS字符串补全方法`padStart()`和`padEnd()`的出现，类似场景使用就简单多了！

二、关于padStart

`padStart`可以在字符串的开头进行字符补全。

语法如下：

```
str.padStart(targetLength [, padString])
```

其中：

targetLength （可选）

`targetLength`指目标字符串长度。

然后，根据我的测试，`targetLength`参数缺省也不会报错，原本的字符串原封不动返回，不过代码没有任何意义，因此，基本上没有使用的理由。

还有，`targetLength`参数的类型可以是数值类型或者弱数值类型。在JavaScript中，`1 == '1'`，`1`是数值，`'1'`虽然本质上是字符串，但也可以看成是弱数值。在`padStart()`方法中，数值类型或者弱数值类型都是可以。例如：

```
'zhangxinxu'.padStart('5');
```

因此，我们实际使用的时候，没必要对`targetLength`参数进行强制的类型转换。

最后，如果`targetLength`设置的长度比字符串本身还要小，则原本的字符串原封不动返回，例如：

```
'zhangxinxu'.padStart(5);
// 结果还是'zhangxinxu'
```

padString （可选）

`padString`表示用来补全长度的字符串。然而，虽然语义上是字符串，但是根据我的测试，任意类型的值都是可以的。无论是Chrome浏览器还是Firefox浏览器，都会尝试将这个参数转换成字符串进行补全。例如下面几个例子：

```
'zhangxinxu'.padStart(15, false);
// 结果是'falsezhangxinxu'
```

```
'zhangxinxu'.padStart(15, null);
// 结果是'nullnzhangxinxu'
```

```
'zhangxinxu'.padStart(15, []);
// 结果是'zhangxinxu'，因为[]转换成字符串是空字符串
```

```
'zhangxinxu'.padStart(15, {});
// 结果是'[objezhangxinxu'，只显示了'[object Object]'前5个字符
```

`padString`参数默认值是普通空格`' '`（U+0020），也就是敲Space空格键出现的空格。

从上面几个案例可以看出，如果补全字符串长度不足，则不断循环补全；如果长度超出，则从左侧开始依次补全，没有补到的字符串直接就忽略。

此方法返回值是补全后的字符串。

回到一开始的日期补`'0'`功能，有了`padStart()`方法，我们代码可以简化成下面这一行：

```
var month = String(new Date().getMonth() + 1).padStart(2, '0');    // 结果是'07'
```

兼容性

IE14以其已下浏览器都不支持，考虑到现在还是windows 7天下，PC端对外项目还不能直接用；移动端

，UC浏览器，QQ浏览器也不支持。但是，也不是不能使用，加一个Polyfill就好了，这个后面会展示。

三、关于padEnd

`padEnd`可以在字符串的后面进行字符补全，语法参数等都和`padStart`类似。

语法：

```
str.padEnd(targetLength [, padString])
```

其中：

targetLength （可选）

`targetLength`指补全后字符串的长度。

然后，根据我的测试，`targetLength`参数如果不设置，不会报错，直接返回原始字符串，不过这样代码就没有任何意义，因此，实际开发，此参数肯定是需要设置的。

同样的，`targetLength`参数的类型可以是数值类型或者弱数值类型。例如下面2个用法都是可以的：

```
'zhangxinxu'.padEnd('15');
'zhangxinxu'.padEnd(15);
```

因此，我们实际写代码的时候，没必要强制`targetLength`参数为数值。

最后，如果`targetLength`设置的长度比字符串本身还要小，则原字符串原封不动返回，例如：

```
'zhangxinxu'.padEnd(5);
// 结果还是'zhangxinxu'
```

如果`targetLength`是一些乱七八糟的数值类型，也是返回原始字符串。例如：

```
'zhangxinxu'.padEnd(false);
// 结果还是'zhangxinxu'
```

但是有意义吗？没意义，所以不要这么用。

padString （可选）

`padString`表示用来补全长度的字符串。虽然语义上是字符串，实际上这个参数可以是各种类型。例如下面几个例子：

```
'zhangxinxu'.padEnd(15, false);
// 结果是'zhangxinxufalse'
```

```
'zhangxinxu'.padEnd(15, null);
// 结果是'zhangxinxunulln'
```

```
'zhangxinxu'.padEnd(15, []);
// 结果是'zhangxinxu'，因为[]转换成字符串是空字符串
```

```
'zhangxinxu'.padEnd(15, {});
// 结果是'zhangxinxu[obje'，只显示了'[object Object]'前5个字符
```

从上面几个案例可以看出，如果补全字符串长度不足，则从左往右不断循环补全；如果长度超出可以补全的字符长度，则从左侧尽可能补全，补不到的没办法，只能忽略，例如`'zhangxinxu'.padEnd(15, {})`等同于执行`'zhangxinxu'.padEnd(15, '[object Object]')`，最多只能补5个字符，因此，只能补`'[object Object]'`前5个字符，于是最后结果是：`'zhangxinxu[obje'`。

`padString`参数如果不设置，则会使用普通空格`' '`（U+0020）代替，也就是Space空格键敲出来的那个空格。

举一个在后面补全字符串案例

在JS前端我们处理时间戳的时候单位都是`ms`毫秒，但是，后端同学返回的时间戳则不一样是毫秒，可能只有10位，以`s`秒为单位。所以，我们在前端处理这个时间戳的时候，保险起见，要先做一个13位的补全，保证单位是毫秒。使用示意：

```
timestamp = +String(timestamp).padEnd(13, '0');
```

兼容性

本字符串API属于ES6新增方法，IE14以其已下浏览器都不支持，部分国产移动端浏览器也不支持。目前对外项目使用还需要附上Polyfill代码。

四、Polyfill代码

以下Polyfill代码取自polyfill项目中的string.polyfill.js，其中使用依赖的`repeat()`也是ES6新增方法，因此，完成Polyfill代码如下，注释部分我做了简单的翻译，代码部分简化了些许逻辑，同时，修复了一个**bug**，下面代码红色高亮部分就是修复内容：

```
// https://github.com/uxitten/polyfill/blob/master/string.polyfill.js
// repeat()方法的polyfill
if (!String.prototype.repeat) {
    String.prototype.repeat = function (count) {
        'use strict';
        if (this == null) {
            throw new TypeError('can\'t convert ' + this + ' to object');
        }
        var str = '' + this;
        count = +count;
        if (count != count) {
            count = 0;
        }
        if (count < 0) {
            throw new RangeError('repeat count must be non-negative');
        }
        if (count == Infinity) {
            throw new RangeError('repeat count must be less than infinity');
        }
        count = Math.floor(count);
        if (str.length == 0 || count == 0) {
            return '';
        }
        if (str.length * count >= 1 << 28) {
            throw new RangeError('repeat count must not overflow maximum string size'
);
        }
```

```
        var rpt = '';
        for (; ;) {
            if ((count & 1) == 1) {
                rpt += str;
            }
            count >>>= 1;
            if (count == 0) {
                break;
            }
            str += str;
        }
        return rpt;
    }
}
// padStart()方法的polyfill
if (!String.prototype.padStart) {
    String.prototype.padStart = function (targetLength, padString) {
        // 截断数字或将非数字转换为0
        targetLength = targetLength>>0;
        padString = String((typeof padString !== 'undefined' ? padString : ' '));
        if (this.length > targetLength || padString === '') {
            return String(this);
        }
        targetLength = targetLength-this.length;
        if (targetLength > padString.length) {
            // 添加到初始值以确保长度足够
            padString += padString.repeat(targetLength / padString.length);
        }
        return padString.slice(0, targetLength) + String(this);
    };
}
// padEnd()方法的polyfill
if (!String.prototype.padEnd) {
    String.prototype.padEnd = function (targetLength, padString) {
        // 转数值或者非数值转换成0
        targetLength = targetLength >> 0;
        padString = String((typeof padString !== 'undefined' ? padString : ' '));
        if (this.length > targetLength || padString === '') {
            return String(this);
        }
        targetLength = targetLength - this.length;
        if (targetLength > padString.length) {
            // 添加到初始值以确保长度足够
            padString += padString.repeat(targetLength / padString.length);
        }
        return String(this) + padString.slice(0, targetLength);
    };
}
```

以上polyfill代码需要放在调用`padStart()`/`padEnd()`方法的代码的前面，只要在合适的位置就这么一粘贴，然后，无论什么版本浏览器浏览器，哪怕IE6，IE8，我们也可以放心使用`padStart()`或者`padEnd()`方法了。

polyfill代码下的demo案例

您可以狠狠地点击这里：padStart和padEnd方法polyfill测试demo

原polyfill方法的一个bug就是通过这个测试demo测出来的，下面是修正后的polyfill代码在IE9浏览

器的测试结果截图：

五、结语

标题虽然是简介，但是本文内容实际上已经非常深入细节了。`padStart()`和`padEnd()`两个方法参数容错性非常强，非常有JS的特色，我很喜欢。但是，也带来另外的问题，就是，如果我们的参数值因为各种原因，最后并不是我们期望的参数值，则很可能会产生非常隐蔽的bug，因为不会报错啊，运行还是正常的。所以，容错性强，也算是有利有弊。

ES6对字符串还扩展了很多其他方法，除了本文提到的`repeat()`方法，还有诸如`normalize()`，`trimStart()`，`trimEnd()`等API方法。这些之后再一个一个深入细节。

话说，我现在才开始深入学习ES6的基础知识应该还不算晚吧，哈哈！

（本篇完） // 想要打赏？点击这里。有话要说？点击这里。

« 深入理解CSS的width:auto

了解woff2字体及转换 »

猜你喜欢

- JavaScript实现http地址自动检测并添加URL链接
- 了解HTML/HTML5中的download属性
- 原来浏览器原生支持JS Base64编码解码
- ES6 JavaScript Promise的感性认知
- HTMLUnknownElement与HTML5自定义元素的故事
- 简单了解ES6/ES2015 Symbol() 方法
- 万岁，浏览器原生支持ES6 export和import模块啦！
- 看，for..in和for..of在那里吵架！

标签： ES6, padEnd, padStart, polyfill, repeat, String

发表评论（目前16条评论）

名称 (必须)

邮件地址(不会被公开) (必须)

网站

提交评论

1. **pakyang说道：**
2018年08月28日 15:22

String.prototype.repeat中的 if(count !== count) ？ ？ ？ 没懂

回复

> **yerled说道：**
> 2018年09月25日 19:33
>
> NaN
>
> 回复

2. **mfk说道：**
2018年07月27日 10:46

没必要用那一大坨的polyfill吧，直接for循环拼不行吗?

var stringProto=String.prototype;

stringProto.padStart=stringProto.padStart||function(targetLength, padString){
targetLength=+targetLength||0;
var currentLength=this.length;
if(curretnLength>=targetLength){
return this;
}
padString=(padString===undefined)?” ”
:(padString+”); //转型成String
var padStringLength=padString.length();
var result=this;
while(currentLength<targetLength){
result=padString+result;
currentLength+=result.length;
}
if(currentLength==targetLength){
return result;
}
return result.substr(-targetLength); //取右边targetLength位
}

padEnd类似。只修改：
（1）while中的那个result=padString+result;为result+=padString;
（2）最后的 return result.substr(-targetLength); 改为 return result.substr(0,targetLength);

两代码中有很多重复部分，可以提取公共部分为一个function。

回复

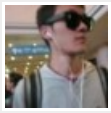
> **mfk说道：**
> 2018年07月27日 10:48
>
> currentLength+=result.length;
> 写错了。应该是
> currentLength+=padStringLength;
>
> 回复

3. 堂说道：
2018年07月25日 16:13

不知道配babel的话还需要加polyfill吗

回复

4. **Halbert**说道：
2018年07月25日 11:10



毫秒的例子，有很大限制吧？
new Date(parseInt(String(+new Date('1970-01-01T08:00:00.001')).padEnd(13, '0')))

回复

5. 和尚爱敲钟说道：
2018年07月24日 17:25

IE完全不支持啊。。。。

回复

> 张 鑫旭说道：
> 2018年07月24日 23:30
>
> 
>
> 可以polyfill~
>
> 回复

6. 王念一说道：
2018年07月24日 16:16

『因此，基本上没有不使用的理由』
不应该是『基本上没有使用的理由』吗…
另外我觉得补位字符串可以有两种首尾对齐模式，比如：
首对齐→"wassup".padStartFromBegin(7, "ab") == "awassup"
尾对齐→"wassup".padStartFromBegin(7, "ab") == "bwassup"

回复

> 张 鑫旭说道：
> 2018年07月24日 23:31
>
> 
>
> 好的，感谢反馈！
>
> 回复

7. 日期补0可以这样子说道：
2018年07月24日 12:23

('0'+(new Date().getMonth()+1)).slice(-2)
偶然遇到的一个方法

回复

8. **Qsir**说道：
2018年07月24日 11:07

张老师，我在自己的小站qixiansen.xyz上推荐你的《css世界》这本书，已经引用了网站链接，在此告知您一声！

回复

张 鑫旭说道：
2018年07月24日 11:12

好的。

回复

你个马屁精说道：
2018年08月6日 14:08

你走是个马屁精

回复

9. 广建说道：
2018年07月24日 10:12

已经关注大神好久

回复

10. **busyhe**说道：
2018年07月24日 00:57

大神如此努力，现在怎可睡觉

回复

最新文章

今日热门

今年热议

» 《CSS世界》女主角诚寻靠谱一起奋斗之人 (76)
» 不借助Echarts等图形框架原生JS快速实现折线图效果 (64)
» 看，for..in和for..of在那里吵架！ (60)
» 是时候好好安利下LuLu UI框架了！ (47)
» 原来浏览器原生支持JS Base64编码解码 (35)
» 妙法攻略：渐变虚框及边框滚动动画的纯CSS实现 (33)
» 炫酷H5中序列图片视频化播放的高性能实现 (31)
» CSS scroll-behavior和JS scrollIntoView让页面滚动平滑 (30)
» windows系统下批量删除OS X系统.DS_Store文件 (26)
» 写给自己看的display: flex布局教程 (26)

猜你喜欢

- JavaScript实现http地址自动检测并添加URL链接
- 了解HTML/HTML5中的download属性
- 原来浏览器原生支持JS Base64编码解码
- ES6 JavaScript Promise的感性认知
- HTMLUnknownElement与HTML5自定义元素的故事
- 简单了解ES6/ES2015 Symbol() 方法
- 万岁，浏览器原生支持ES6 export和import模块啦！
- 看，for..in和for..of在那里吵架！