**ToLocalString 方法**



[qq\_42453950](https://blog.csdn.net/qq_42453950)https://csdnimg.cn/release/blogv2/dist/pc/img/newCurrentTime2.png于 2019-03-08 10:06:14 发布https://csdnimg.cn/release/blogv2/dist/pc/img/articleReadEyes2.png3538https://csdnimg.cn/release/blogv2/dist/pc/img/tobarCollect2.png 收藏 9

toLocalString 方法是用于返回格式化对象后的字符串，该字符串格式因不同语言而不同。可以通过传参决定返回的语言与具体的表现，某些场景下相当有用，语法如下：

object.toLocaleString([locales [, options]]);

locales 参数用于指定格式化对象时使用的语言环境，默认为当前环境的语言，可以不传。该参数具体可选的值可以参考这里，一般而言使用 en 或 zh 即可应付绝大多数情况。例子如下：

const date = new Date();

date.toLocaleString('zh'); // 2018-4-4 15:08:38

date.toLocaleString('en'); // 4/4/2018, 3:08:38 PM

顺带一提，此参数大小写不敏感，已经在浏览器与 Node 验证过。

options 参数为输出样式的配置项，根据 object 类型不同会有不同选项，下文会仔细解释这个参数。但需要注意的是如果不传 locales 参数，那么 options 参数是不会生效的，其实上面的语法其实已经显现出这点。

最后则是兼容性问题，具体如下图：

使用参数的情况下兼容性稍差，这比较可惜，但总体而言还是比较乐观的。

**Number.prototype.toLocaleString**

先介绍 toLocaleString 在数字类型上的使用。面试偶尔会问到如何格式化数字，使整数部分每三位加一个逗号，这时不妨：

const num = 2333333;

num.toLocaleString(); // 2,333,333

其实是不需要任何正则的~打完出题者的脸后，我们一起看看 toLocaleString 在数字类型的 options 参数有好用属性，以方便我们  
  
~~偷懒~~  
使用。注意，本文不是翻译文档，因而只会介绍一些常用的属性，更具体的选项请查阅MDN相关文档。

style 表示格式化时使用的样式，默认值是 decimal 也就是纯数字，也可为 percent 百分比显示与 currency货币显示。值为 currency 时必须同时指定 options 中的 currency 属性，否则报错。具体例子如下：

const num = 2333333;

num.toLocaleString('zh', { style: 'decimal' }); //2,333,333

num.toLocaleString('zh', { style: 'percent' }); //233,333,300%

num.toLocaleString('zh', { style: 'currency' }); //报错

接下来的两个属性是 style 设为 currency 时才有用的，它们分别是 currency 与 currencyDisplay ，前者指定对应的货币，如 USD 、 EUR 与 CNY 等，实测也是不区分大小写的。后者是货币符号的展示样式，默认值是 symbol ，即对应的符号，如 CNY 是 ￥。该属性的值也可以是 code 与 name ，只是用得比较少，看看例子就好了~具体如下：

const num = 2333333;

num.toLocaleString('zh', { style: 'currency', currency: 'CNY' }); //￥2,333,333.00

num.toLocaleString('zh', { style: 'currency', currency: 'cny', currencyDisplay: 'code' }); //CNY2,333,333.00

num.toLocaleString('zh', { style: 'currency', currency: 'cny', currencyDisplay: 'name' }); //2,333,333.00人民币

最后是两组相当强大的属性，某些场景下能带来极大的便利。第一组是 minimumIntegerDigits 、 minimumFractionDigits 与 maximumFractionDigits ，用于指定整数最少位数与小数的最少和最多位数，不够则用0去凑。简单说，自动补0！具体例子如下：

let num = 2333.3;

num.toLocaleString('zh', { minimumIntegerDigits: 5 }); //02,333.3

//如果不想有分隔符，可以指定useGrouping为false

num.toLocaleString('zh', { minimumIntegerDigits: 5, useGrouping: false }); //02333.3

num.toLocaleString('zh', { minimumFractionDigits: 2, useGrouping: false }); //2333.30

num = 66666.666

num.toLocaleString('zh', { maximumFractionDigits: 2, useGrouping: true }); //66,666.67

从此之后，补0与控制位数再也不愁~

另一组是 minimumSignificantDigits 与 maximumSignificantDigits ，用于控制有效数字位数，只要设置了这一组属性，第一组属性全部忽略不算，具体如下：

const num = 1234.5;

num.toLocaleString('zh', { minimumSignificantDigits: 6, useGrouping: false }); //1234.50

num.toLocaleString('zh', { maximumSignificantDigits: 4, useGrouping: false }); //1235

注意， maximumFractionDigits 与 maximumSignificantDigits 均是四舍五入，使用时需要注意。数字类型的 toLocaleString 介绍就告一段落，下面让我们看看日期类型的 toLocaleString 的 options 有什么好用的属性。

**Date.prototype.toLocaleString**

与数字类型不同，日期类型的 locales 对输出的影响十分之大（其实数字类型影响也大，只是一般用不到），因而应该根据实际情况选择合适的语言环境。一般而言，日期显示是有要求的，样式方面需要统一。但如果是内部项目或者是 pm 允许，那么使用 toLocaleString 格式化日期就相当好了。与数字类型一样，只介绍常用属性，详细的属性介绍请查阅MDN文档。

hour12 表示是使用十二小时制还是二十四小时制，默认值视 locales 而定。例子如下：

const date = new Date();

date.toLocaleString('zh', { hour12: true }); //2018/4/4 下午6:57:36

date.toLocaleString('zh', { hour12: false }); //2018/4/4 18:57:36

之后就是格式化年月日时分秒星期等选项了，MDN 文档说必须按照一定的分组设置属性，实际使用中发现每个属性单独使用并不报错，因而按属性的值分开介绍会比较好理解。

具体的属性一共有 9 个，分别是 weekday 、 era 、 year 、 month 、 day 、 hour 、 minute 、 second与 timeZoneName 。具体的意思，看单词估计就能秒懂，不作过多解释。然而需要留意的是他们的可选值。先讨论 weekday 与 era ，它们均可以取值为 narrow 、 short 或 long ，简单说就是能有多短多短，缩写与正常表现，具体表现如下：

const date = new Date();

date.toLocaleString('en', { weekday: 'narrow', era: 'narrow' }); //W A

date.toLocaleString('en', { weekday: 'short', era: 'short' }); //Wed AD

date.toLocaleString('en', { weekday: 'long', era: 'long' }); //Wednesday Anno Domini

跟着是 timeZoneName 属性，这个属性只有 short 或 long 两个值，表现如下：

const date = new Date();

date.toLocaleString('zh', { timeZoneName: 'short' }); //2018/4/5 GMT+8 下午7:18:26

date.toLocaleString('zh', { timeZoneName: 'long' }); //2018/4/5 中国标准时间 下午7:18:26

剩下的属性，均可以取值为 numeric 与 2-digit ，简单说就是否仅用两位数字表示，看码说话：

const date = new Date();

date.toLocaleString('zh', { year: 'numeric', month: 'numeric', day: 'numeric', hour: 'numeric', minute: 'numeric', second: 'numeric', }); //2018/4/5 下午7:30:17

date.toLocaleString('zh', { year: '2-digit', month: '2-digit', day: '2-digit', hour: '2-digit', minute: '2-digit', second: '2-digit' }); //18/04/05 下午7:30:17

（比较奇怪的是 hour 、 minute 与 second 三个属性，无论设置为何值，表现都是一样的，希望有大佬告知原因，我换成 en 环境也是一样的。）

最后是 month 这个属性，语言对月份有不同的展现，除去 numeric 与 2-digit 外，它额外多三个属性，分别是 narrow 、 short 与 long 。展示如下：

const date = new Date();

date.toLocaleString('en', { month: 'narrow' }); //A

date.toLocaleString('en', { month: 'short' }); //Apr

date.toLocaleString('en', { month: 'long' }); //April

**小结**

至此，关于 toLocaleString 的介绍就暂告一段落了。可以看到，在某些场景下需要格式化对象成字符串时，可以发挥极大的作用，不再需要自己苦兮兮地写函数做转换。尽管这个 API 稍微冷门一点，但还是很有意思的。