Day 13---日期对象和万年历

一、了解日期和时间的一些概念

1.1 GMT时间

GMT(Greenwich mean time 格林尼治标准时间，格林威治标准时间，格林尼治平均时间)：是指位于英国伦敦郊区的皇家格林尼治天文台的标准时间，因为本初子午线被定义在通过那里的经线。自1924年2月5日开始，格林尼治天文台每隔一小时会向全世界发放调时信息。世界上发生的重大时间都是以格林尼治时间时间为标准的。

比如：我们用的北京时间，因为北京是东八区，所以我们的事件比GMT事件早八个小时。我们这里是8点早上，格林威治还是夜里0点呢。

1.2 UTC时间

UTC(Coordinated Universal Time 协调世界时间)　协调世界时是以原子时秒长为基础，在时刻上尽量接近于世界时的一种时间计量系统。

原子时与以往的计时系统不同，它非常精确并且不以某地的平均太阳时为基准，但是遇有地球自转速度不均匀，原子时与世界时之间的时差便日积月累，因 此，UTC 会在一段时期后加上正或负的闰秒来补偿。因此协调世界时与国际原子时(TAI) 之间会出现若干整数秒的差别。位于巴黎的国际地球自转事务中央局(ERS)负责决定何时加入闰秒。

二、创建日期对象(120)

> JavaScript中的 Date 类型是在早期 Java 中的 java.util.Date 类基础上构建的。为此， Date类型使用自 UTC 1970 年 1 月 1 日午夜（零时）开始经过的毫秒数来保存日期。在使用这种数据存储格式的条件下， Date 类型保存的日期能够精确到 1970 年 1月 1 日之前或之后的 285 616 年。

2.1 获取当前日期字符串形式

> 直接调用Date()函数(注意首字母大写)。返回的是表示当前日期和时间的字符串。传参数无效，所以不需要传入参数。
>
> 注意：得到的仅仅是个String类型的字符串

```
<body>
    <script type="text/javascript">
        var date = Date();
        alert(date);
    </script>
</body>
```

2.2 创建日期对象

2.2.1 创建表示当前表示当前日期的对象

> new Date()

```
<body>
    <script type="text/javascript">
        //使用关键字new 调用Date()构造函数，创建的对象表示当前日期
        var now = new Date();
        alert(now);
    </script>
</body>
```

2.2.2 创建指定字符串日期对象

> new Date("字符串格式日期")
>
> 支持格式1：英文月 日,年 时:分:秒 例如：May 23, 2016 09:00:00
>
> 支持格式2：年-月-日 例如：2016-11-11
>
> 支持格式3：年/月/日 例如：2016/11/11

```
<body>
    <script type="text/javascript">
        var d1 = new Date("May 31, 2016 09:00:00");
        console.log(d1);    //Tue May 31 2016 09:00:00 GMT+0800 (中国标准时间)
        var d2 = new Date("2016-11-12");
        console.log(d2);    //Sat Nov 12 2016 08:00:00 GMT+0800 (中国标准时间)
        var d3 = new Date("2016/11/12");
        console.log(d3);    //Sat Nov 12 2016 08:00:00 GMT+0800 (中国标准时间)
    </script>
</body>
```

2.2.3 创建指定毫秒值的日期对象

> new Date(指定的毫秒值)
>
> 这个毫秒值是指的从1970年1月1日0:0:0开始算的毫秒值

```
<body>
    <script type="text/javascript">
        var d1 = new Date(1646464646646);
        console.log(d1);    // Sat Mar 05 2022 15:17:26 GMT+0800 (中国标准时间)
        var d2 = new Date(1000 * 60 * 60 * 72);
        console.log(d2);    //Sun Jan 04 1970 08:00:00 GMT+0800 (中国标准时间)
    </script>
</body>
```

2.2.4 通过参数传入年月日时分秒来创建日期对象

> newDate(年, 月[, 日, 时, 分, 秒])
>
> 注意：
>
> - 年月必须有，日时分秒可以省略。(1日，0分，0秒)
> - 月份的范围是0-11. 0表示1月份。
> - 如果设置的月日分秒超过了范围，则自动进行增减操作。例如：如果一个月只有31天，你如果传入32，则自动调整为下个月的1号。

```
<body>
    <script type="text/javascript">
        var d1 = new Date(2016, 11, 31);
        console.log(d1);    //Sat Dec 31 2016 00:00:00 GMT+0800 (中国标准时间)
        var d2 = new Date(2016, 11, 33);
        console.log(d2);    //Mon Jan 02 2017 00:00:00 GMT+0800 (中国标准时间)
    </script>
</body>
```

三、日期继承的3个方法(121)

3.1 toString()方法

> toString()方法一般返回带有时区信息的日期和时间。
>
> 备注：我们直接输出一个日期对象的时候，就相当于调用了toString()方法再输出

3.2 toLocalString() 方法

> toLocalString()返回符合本地习惯的日期和时间格式

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.toLocaleString()); //  2016/11/9 上午11:28:01    注意：不同的浏览器有差异
    </script>
</body>
```

3.3 valueOf()方法

> valueOf()方法不返回字符串，而是返回的代表这个时间的毫秒值。
>
> 一般用用来比较两个日期的大小，就知道谁在前谁在后了。
>
> 日期早的小于日期晚的。
>
> 例如：2016年11年11日是小于2016年11月12日的

```
<body>
    <script type="text/javascript">
        var d1 = new Date("2016-11-20");
        var d2 = new Date("2016-11-22");
        console.log(d1 < d2); //true
        console.log(d1 > d2); //false
        console.log(d1 - d2);  // 间隔的毫秒值：-172800000
    </script>
</body>
```

四、Date对象的常用方法(122)

4.1 getTime()和setTime(毫秒值)

> getTime()返回表示日期的毫秒值，与valueOf()一样
>
> setTime(毫秒值) 以毫秒数设置日期，会改变整个日期

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getTime());     //1478667185681
        date.setTime(979797979879879);
        console.log(date.toLocaleString());//33018/7/25 下午5:31:19
    </script>
</body>
```

4.2 getFullYear()和setFullYear(年份)

> getFullYear() 取得4位数的年份
>
> setFullYear()设置日期的年份，必须是4为数字的年份,否则将来获取的年份是不对的。

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getFullYear());
        date.setFullYear(2018);
        console.log(date.toLocaleString());
    </script>
</body>
```

4.3 getMonth()和setMonth(月份)

> getMonth()取得月份。 范围：0-11
>
> setMonth(月份) 设置月份

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getMonth());    //如果是11月份则返回 10
        date.setMonth(2);    // 设置月份为3月
        console.log(date.toLocaleString());
    </script>
</body>
```

4.4 getDate()和setDate(日)

> getDate() 获取一个月中的第几天(1-31)
>
> setDate(日) 设置一个月中的第几天

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getDate());
        date.setDate(11);
        console.log(date.toLocaleString());
    </script>
</body
```

4.5 getDay()

> getDay() 获取是星期几 0代表星期日 6代表星期6
>
> 注意：没有setDay()的方法

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getDay());
    </script>
</body>
```

4.6 getHours()和setHours(小时)

> getHours()　获取小时数(0-23)
>
> setHours(小时)　设置小时数　如果设置的值超过23则更改月份中的日期，小于0则减少日期数

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getHours());
        date.setHours(10);
        console.log(date.toLocaleString());
    </script>
</body>
```

4.7 getMinutes() 和setMinutes(分钟)

> getMinutes() 获取分钟数(0-59)
>
> setMinutes(分钟) 设置分钟数 如果超过60或小于0则自动增加小时

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getMinutes());
        date.setMinutes(62);
        console.log(date.toLocaleString());
    </script>
</body>
```

4.8 getSeconds()和setSeconds(秒)

> getSeconds() 获取秒数 0-59
>
> setSeconds(秒数) 设置秒数 如果超过60或小于0，则自动增减分钟数

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getSeconds());
        date.setSeconds(62);
        console.log(date.toLocaleString());
    </script>
</body>
```

4.9 getMilliSeconds()和setMilliSeconds(毫秒值)

> getMilliSeconds() 获取毫秒值 0-999
>
> setMilliSeconds(毫秒值) 设置毫秒值

```
<body>
    <script type="text/javascript">
        var date = new Date();
        console.log(date.getMilliseconds());
        date.setMilliseconds(999);
        console.log(date.toLocaleString());
    </script>
</body>
```

五、 日期格式化方法

1. toDateString() ——以特定于实现的格式显示星期几、月、日和年；
2. toTimeString() ——以特定于实现的格式显示时、分、秒和时区；
3. toLocaleDateString() ——以特定于地区的格式显示星期几、月、日和年；
4. toLocaleTimeString() ——以特定于实现的格式显示时、分、秒；
5. toUTCString() ——以特定于实现的格式完整的 UTC 日期。