**获取当前本机电脑的时间:获取的是一个标准的时间格式的数据**

如果要求以时间为重要参考依据的不能使用本机的时间(例如：秒杀)

var nowTime = new Date();

Sun Jun 28 2015 12:19:41 GMT+0800 (中国标准时间)

JS中提供了一系列的操作标准时间格式数据的方法：

**将指定时间格式的字符串变成我们想要的格式(如果timeStr没有传，默认获取的是当前自己电脑的时间)**

var nowTime = **new Date();**

var year = nowTime.**getFullYear();**//获取四位年

var month = nowTime.**getMonth()** + 1;//0-11代表1-12月

var day = nowTime.**getDate();**

var weekStr = "日一二三四五六", week = "星期" + weekStr.charAt(nowTime**.getDay()**);

var hours = nowTime.**getHours();**

var minutes = nowTime.**getMinutes();**

var seconds = nowTime**.getSeconds();**

var mSeconds = nowTime**.getMilliseconds();**

var str = "2015/6/28 12:49:30";

如果想向上面一样用那些方法操作，必须先把字符串变成标准的时间格式数据

要求时间格式的字符串要符合一定的规则，中间用-分隔的在IE6~8下不兼容，想要兼容需要把-替换成/

var time = new Date(str);

console.log(time);

var week = nowTime.getDay();//0-6代表周日-周六

var weekStr = "日一二三四五六";//索引0-6正好和我们week相同

var str = "星期" + weekStr.charAt(week);

**getTime:获取时间距离1970-1-1 00:00:00的毫秒差**

console.log(time.getTime());//1435467651652

当前时间-->距离1970年毫秒差 n

目标时间-->距离1970年毫秒差 m

m-n:目标时间和当前时间之间相差的毫秒数

**定时器：**设置一个定时器，规定一个等待时间，当到达指定时间后执行我们相关的操作

两种类型的定时器:

[interval]设置的等待的时间

[function]到达时间后执行的方法

**window.setInterval([function],[interval]);** 设置一个定时器，到达指定时间后，执行[function]，执行完成，定时器并没有停止，以后每隔这么长的时间都重新执行一下[function]

**window.setTimeout([function],[interval]);**设置一个定时器，到达指定的时间后，执行[function],执行完成，定时器自动停止

**window.setInterval([function],[interval])**

设置一个定时器，到达指定的时间(interval 毫秒)，执行function，但是定时器并不停止，以后每隔这么长时间，都从新执行一遍function

**window.setTimeout([function],[interval])**

设置一个定时器，到达指定的时间(interval 毫秒)，执行我们的操作(function)，定时器就停止运行了

定时器是有返回值的，返回值是一个数字，代表当前是第几个定时器

使用setTimeout来模拟setInterval的持久化执行

原理：设置一个定时器到达时间执行方法，方法执行完成，我们在重新设置一个定时器，到达时间重新执行这个方法...--->递归模式:自己执行完成在重新调用自己执行

关于定时器的异步加载原理

**同步：**上一件事情没有完成，下一件事情是不能处理的，需要等上一件完成才能处理

**异步：**上一件事情即使没有完成，下一件事情已经开始执行了

**所有的定时器都是异步的**：在定时器等待期间，下面JS代码需要继续执行的

在定时器中等待时间设为0，并不是立即执行，每一个浏览器都有一个默认的最小等待时间 [参考：谷歌下是5-6毫秒 IE下10毫秒左右]

**JS是单线程的，**有一个任务在线程上工作的时候，下一个任务可以等着，但是不能工作，在定时器中，设置一个定时器，在等待过程中，代码继续执行，但是只有等下面的当代码都执行完成(不管你定时器有没有到时间)，才会返回头看定时器到时间了吗，到了时间则执行