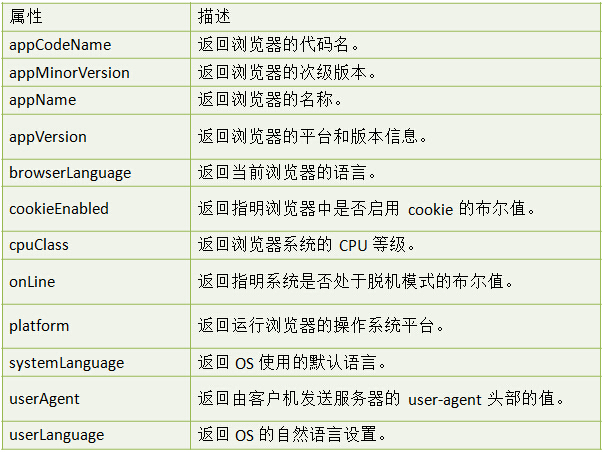
2020年6月26日星期五

* location对象：
  + location 用于获取或设置窗体的 URL，并且可以用于解析 URL。
  + 使用语法：location.[属性|方法]或 window.location.[属性|方法]



* + 属性描述
    - hash设置或返回从井号 (#) 开始的 URL（锚）。
    - host设置或返回主机名和当前 URL 的端口号。
    - hostname设置或返回当前 URL 的主机名。
    - href设置或返回完整的 URL。
    - pathname设置或返回当前 URL 的路径部分。
    - port设置或返回当前 URL 的端口号。
    - protocol设置或返回当前 URL 的协议。
    - search设置或返回从问号 (?) 开始的 URL（查询部分）。
  + 方法描述
  + assign()加载新的文档。
  + reload()重新加载当前文档。
  + replace()用新的文档替换当前文档。
* navigator对象



* screen 对象
  + screen 就是你的屏幕，而这个对象里就包含了很多关于你的屏幕的信息，比如高度、宽度、颜色深度等。
  + screen.height 可以返回屏幕分辨率的高。
  + screen.width 返回屏幕分辨率的宽。(单位都以像素计)
  + screen.availWidth 属性返回访问者屏幕的宽度，以像素计，减去界面特性，比如任务栏。
  + screen.availHeight 属性返回访问者屏幕的高度，以像素计，减去界面特性，比如任务栏。
* DOM（Document Object Model）文档对象模型
  + - 三种常见的 DOM 节点:
    - 1. 元素节点：上图中 <html>、<body>、<p> 等都是元素节点，即标签。
    - 2. 文本节点：向用户展示的内容，如 <li>...</li> 中的 JavaScript、DOM、CSS 等文本。
    - 3. 属性节点：元素属性，如 <a> 标签的链接属性href="http://www.jisuanke.com”。
* 节点返回文本。
  + nodeValue 可读可写，这是对元素节点不能写。一般只用于设置文本节点的值。</td></tr></tbody></table>''')
  + 属性说明
    - nodeName节点名称，相当于 tagName 属性节点返回属性名，文本节点返回 #text。nodeName，是只读的。
    - nodeType节点的类型 ，返回值：1，元素节点；2，属性节点；3，文本节点。nodeType 是只读的。
    - nodeValue节点的值，返回一个字符串，指示这个节点的值 。元素节点返回 null，属性节点返回属性值，文本节点返回文本。nodeValue 可读可写，这是对元素节点不能写。一般只用于设置文本 节点的值。
  + 一、nodeName 属性：节点的名称，是只读的。
    - 1. 元素节点的 nodeName 与标签名相同
    - 2. 属性节点的 nodeName 是属性的名 称
    - 3. 文本节点的 nodeName 永远是#text
    - 4. 文档节点的 nodeName 永远是#document
  + 二、nodeValue 属性：节点的值
    - 1. 元素节点的 nodeValue 是 undefined 或 null
    - 2. 文本节点的 nodeValue 是文本自身
    - 3. 属性节点的 nodeValue 是属性的值
  + 三、nodeType 属性：节点的类型，是只读的。以下常用的几种结点类型:
    - 元素类型：元素、属性、文本、注释、文档
    - 节点类型：1、2、3、8、9
* 元素内容
  + InnerHTML：获取或替换元素的内容
  + InnerHTML：获取或替换元素的文本内容
* 元素样式
  + style 可以直接改变元素的样式，比如宽高，颜色，背景等，
    - Object.style.property = new style;
  + className 只会改变元素的类（即 class），具体的样式写在了类之中。
    - Object.className = classname;
* 节点
  + childNodes
    - 节点之间的空白符是一个文本节点
  + firstChild
  + lastChild
  + parentNode
  + nextSibling、previousSibling
    - 两个属性获取的是节点，所以我们可以先判断节点 nodeType 是否为 1，如不为元素节点，跳过。
* removeChild
  + 需要注意的是：把删除的子节点赋值给 rm，这个子节点不在 DOM 树中，但是还存在内存中，如果要完全删除对象，需给 rm 赋 null 值。
* fnode.insertBefore(newnode, node);返回插入的节点
* fnode.replaceChild(newnode, oldnode);返回oldnode