20.2.3 设计工具提示

Tooltip（工具提示）是一种比较实用的 JavaScript 应用。当为一个元素（一般为超链接 a 元素）定义 title 属性时，会在鼠标经过是显示提示信息，这些提示能够详细描绘经过对象的包含信息，这对于超链接（特别是图像式超链接）非常有用。同时，搜索引擎也喜欢检索这些信息。

设计思路：使用 DOM 技术获取 title（或其他属性）中的提示信息，然后把这些属性删除，再利用 JavaScript 脚本动态生成一个浮动的层，在层中显示这些提示信息，最后利用 Even 事件对象的鼠标指针坐标属性进行定位。如果结合 CSS 技术，可以把这写浮动的层设计成不同样式，以此达到个性化设计的要求。

【示例 1】本示例不涉及到结构层和表现层的设计，为了化繁为简，这里先就一个简单的案例来探索 Tooltip 脚本的实现过程。

【操作步骤】

第 1 步，启动 Dreamweaver，新建文档，保存为 test.html。在文档中设计如下的超链接，其提示信息设置为“title="提示信息"”。下面尝试把这个提示信息提取出来，然后删除该属性，最后使用一个新创建的 div 元素动态显示它的位置。并借助 CSS 美化一下该 div 元素。

```
<a href="#" title="提示信息" target="_blank">超链接文本</a>
```

第 2 步，在脚本中获取超链接元素 a，以及该标签设置的 title 属性值。代码如下：

```
var a = document.getElementsByTagName("a")[0];
//获取 a 元素的引用
var tit   = a.getAttribute("title");
//获取 title 属性值，并存储到一个变量中
```

第 3 步，获取 title 属性值之后，删除该属性，避免它干扰设计。

```
a.removeAttribute("title");                                    //移出 title 属性
```

第 4 步，创建一个 div 元素和一个文本节点，把 titile 属性值赋予给文本节点，然后把文本节点增加到 div 元素内，设置 div 元素样式为绝对定位，并增加一个 class 属性和值，以便于在 CSS 样式表中对该 div 元素进行更个性的控制。

```
var div = document.createElement("div");
//创建 div 元素节点
var   txt = document.createTextNode(tit);
//创建文本节点，并显示 title 属性值
div.style.position = "absolute";
//为 div 元素定义一个行内样式：绝对定位
div.setAttribute("class", "title");
//为 div 元素增加一个类样式，兼容非 IE
div.setAttribute("className", "title");
//为 div 元素增加一个类样式，兼容 IE
div.appendChild(txt);
//获取 title 属性值，并存
```

第 5 步，为超链接 a 元素绑定鼠标经过和鼠标移出的事件处理函数。设计当鼠标指针移过超链接文本时，把创建的 div 元素节点增加到该 a 元素中，而当鼠标指针移出超链接文本时，把 a 元素中增加的 div 节点删除掉。

```
a.onmouseover = function(){
//鼠标经过事件处理函数
    a.appendChild(div);
//把 div 元素增加到 a 元素中
}
a.onmouseout = function(){
//鼠标移出事件处理函数
    a.removeChild(div);
//把 a 元素中的 div 元素删除
}
```

第 6 步，定义鼠标移动事件处理函数，实际实时跟踪鼠标指针的坐标，并利用该坐标来定位创建的 div 元素的显示位置，以实现它始终显示在鼠标指针的右下角。

第 7 步，最后在 CSS 样式表中为 title 类定义类样式，至此整个提示框效果就设计完成了。在浏览器中预览，则显示效果如图 E20.2、图 E20.3 所示。

```
<style type="text/css">
.title {/* 提示框类样式 */
        padding:4px 8px;              /* 增加内侧补白 */
    border:solid 2px red;             /* 设计边框样式 */
    background:blue;                  /* 定义背景为蓝色 */
    color:#fff;                       /* 定义字体为白色 */
    text-decoration:none;             /* 清除受 a 元素影响而产生的下划线 */
}
</style>
```

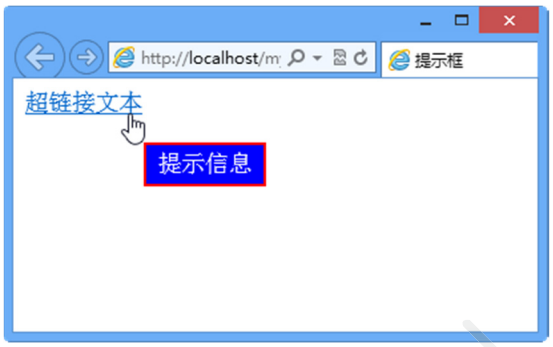

图 E20.2 指定元素的提示框演示效果

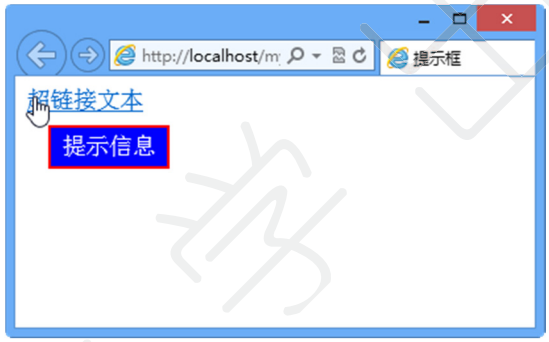

图 E20.3 指定元素的提示框演示效果

【示例 2】示例 1 仅就页面中某个具体的超链接来定义提示框，但是在实际设计中是无法预测到页面中到底有多少超链接，为此需要使用遍历 a 元素节点集合技术来实现动态为页面中所有超链接设计提示框，演示如图 E20.4、图 E20.5 所示。

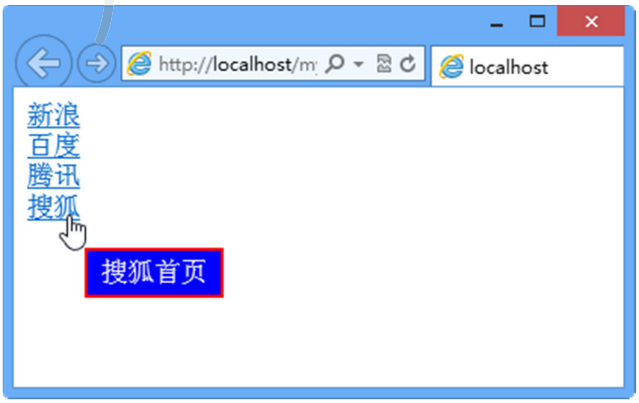

图 E20.4 为页面中所有超链接设计提示框演示效果

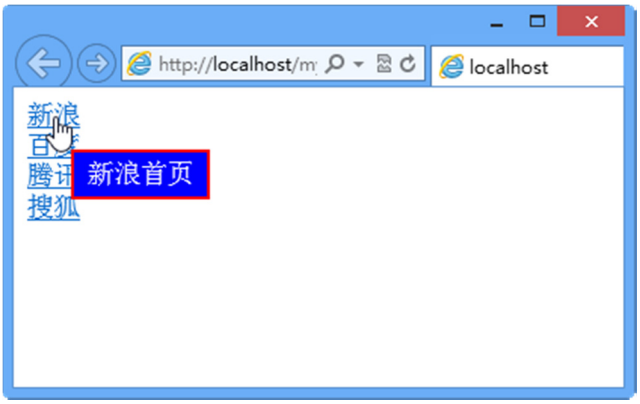

图 E20.5 为页面中所有超链接设计提示框演示效果

实现上述演示效果，本示例依然采用上节示例的设计思路，只不过中间增加了 for 循环结构来遍历文档中所有的超链接元素 a。

【操作步骤】

第 1 步，复制 test.html，另存为 test2.html。在页面初始化事件中绑定一个事件处理函数。然后获取页面内所有 a 元素的引用。

```
window.onload = function(){
    var a = document.getElementsByTagName("a");
//创建 div 元素节点
}
```

第 2 步，在该事件处理函数内，完成上节示例的操作

```
for(var i = 0; i < a.length; i ++ ){        //遍历页面内所有 a 元素
    tit = a[i].getAttribute("title");
//获取 a 元素的 title 属性值
    if(tit)   a[i].removeAttribute("title");
//如果属性值存在，则删除该属性
    var div = document.createElement("div");
//创建 div 元素节点
    var txt = document.createTextNode(tit);
//创建并把提示信息赋予给文本节点
    div.setAttribute("class", "title");
//为 div 元素增加类属性，兼容 FF
    div.setAttribute("className", "title");
//为 div 元素增加类属性，兼容 IE
    div.style.position = "absolute";
//绝对定位 div 元素
    div.appendChild(txt);
//把文本节点增加到 div 元素
}
```

第 3 步，设计鼠标经过和移出的事件处理函数，以实现增加和删除 div 到 a 元素。考虑到在函数体内定义闭包是无法与外界进行数据交流的，为此我们在这里主动为闭包函数传递外部动态参数。

```
a[i].onmouseover = (function(i,div){    //鼠标经过时的事件处理函数
    return function(){                  //返回处理函数
        a[i].appendChild(div);          //为 a 元素增加 div 元素
    }
})(i,div);
//为闭包函数传递参数，i 表示循环变量值，div 表示引用
a[i].onmouseout = (function(i,div){     //鼠标移出时的事件处理函数
    return function(){                  //返回处理函数
        a[i].removeChild(div);          //为 a 元素移出 div 元素
    }
})(i,div);
```

第 4 步，设计鼠标指针移动的事件处理函数。为了能够兼容不同主流浏览器，以及考虑浏览器窗口可能会出现滚动条，所以这里使用多个条件结构进行判断来设置指针的坐标值，读者需要了解 Event 对象的属性，详细内容可以参阅上一章讲解。

```
a[i].onmousemove = (function(div,e){//第 1 个参数表示定位元素引用，第 2 个参数表示事件参数
    return function(e) {//闭包内返回函数体
        var posx = 0, posy = 0;
//定义两个局部变量，用来存储鼠标指针的坐标
        //判断当前浏览器，如果为 IE，则使用 window.event 获取鼠标指针
        if(e == null) e = window.event;
        //判断是否支持 pageX 或 pageY 事件属性，如果支持表示浏览器符支持 DOM 2.0（如 FF 等），此时可以使用这两个属性获取鼠标指针在窗口中的坐标
        if(e.pageX || e.pageY){
            posx = e.pageX;
            posy = e.pageY;
        }
        else if(e.clientX || e.clientY){
//如果不支持 pageX 或 pageY 事件属性，则使用 clientX 或 clientY
//如果支持 document.documentElement.scrollTop 属性，则使用该属性值计算指针坐标
            if(document.documentElement.scrollTop){
                posx = e.clientX + document.documentElement.scrollLeft;
                posy = e.clientY + document.documentElement.scrollTop;
            }else{//否则使用传统的方法来计算指针的坐标位置
                posx = e.clientX + document.body.scrollLeft;
                posy = e.clientY + document.body.scrollTop;
            }
        }
        div.style.top = (posy + 20) + "px";
//把鼠标指针的 y 坐标作为定位值赋予给 div
        div.style.left = (posx + 10) + "px";
//把鼠标指针的 x 坐标作为定位值赋予给 div
    }
})(div);
```

【提示】

documentElement 在 DOM 2.0 中表示 html 元素，因此要获取当前页面的滚动条纵坐标位置，可以使用如下方法：

```
document.documentElement.scrollTop;
```

如果浏览器不支持 documentElement 对象，则可以使用如下方法获取滚动条纵坐标位置：

```
document.body.scrollTop;
```

这里 body 对象表示 body 元素，而 document 表示文档对象，scrollTop 属性表示滚动条的纵坐标。在标准 W3C 下，document.body.scrollTop 始终为 0，需要使用 document.documentElement.scrollTop 来获取滚动条坐标。IE 浏览器从 5.5 版本开始不再支持 document.body.scrollTop 和 document.body.scrollLeft 方法，所以在编程中一般使用上面方法来进行判断。