第九章 拖放

内容安排

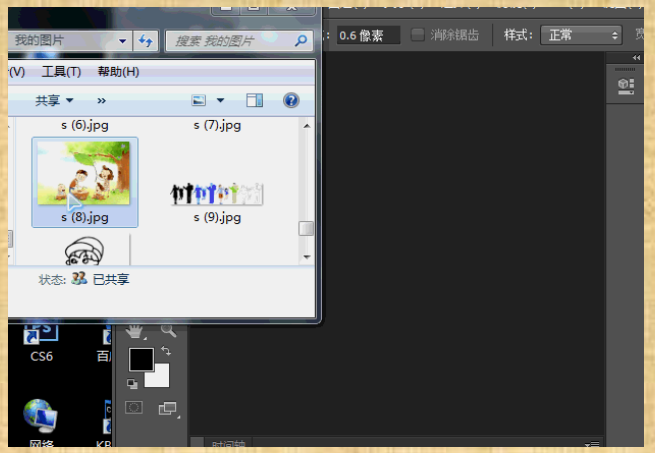
- 9.1 拖放概述
- 9.2 传递拖拽数据
- 9.3 HTML5拖放的实例
- 9.4 小结

9.1 拖放概述

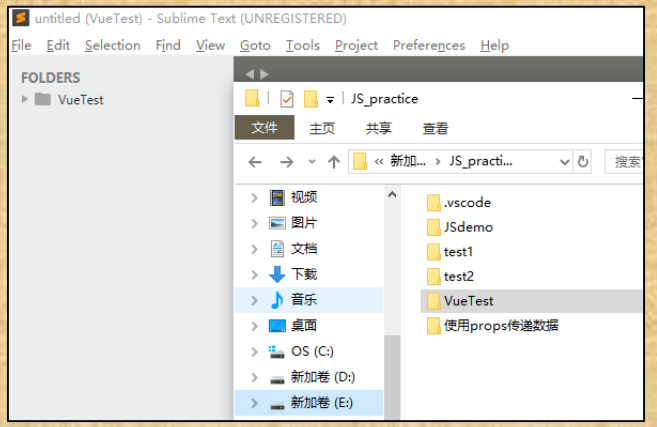
- 什么是拖放？
- 设置元素为可拖放
- 拖放事件

什么是拖放？

拖放可以分为两个动作，即拖拽（drag）和放开（drop）。拖拽就是移动鼠标到指定对象，按下左键，然后拖动对象；放开就是放开鼠标左键，放下对象。



将图片拖放到PS软件里



将文件拖放到sublime text

设置元素为可拖放

首先要定义使网页中的元素可以被拖放，可以通过将元素的draggable属性设置为true实现此功能。

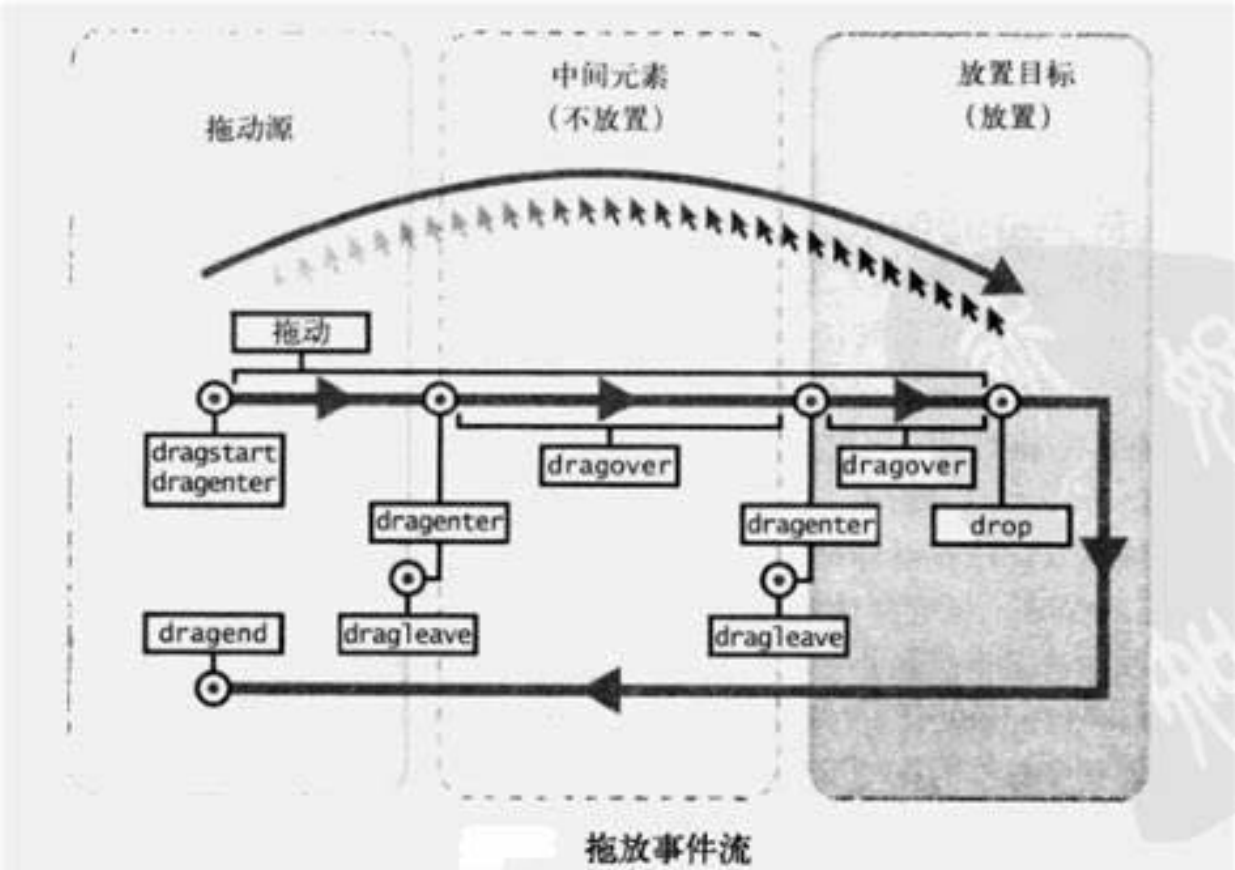
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>设置元素可拖放</title>
</head>
<body>
    <img src="figures/bird.jpg" alt="飞翔的小鸟" draggable="true">
</body>
</html>
```

拖放事件

事 件	说 明	作 用 对 象
dragstart	源对象开始拖放，开始移动时事件触发	源对象
drag	源对象拖放过程中，移动被拖拽对象时触发	源对象
dragenter	源对象进入过程对象范围内，被拖拽对象进入过程对象时被触发	过程对象
dragleave	源对象离开过程对象的范围，被拖拽对象离开目标对象时触发	过程对象
dragover	源对象在过程对象范围内移动，被拖拽对象在过程对象内移动时触发	过程对象
drop	用户释放鼠标时就会触发	目标对象
dragend	源对象拖放结束，整个拖放操作结束时触发。	源对象

拖放事件

拖放事件流程：



拖放事件流

拖放事件的处理函数

- 在定义元素时，可以指定拖放事件的处理函数。

例如，在网页中定义一个可拖放的图片，并指定其dragstart事件的处理函数为drag(event)代码如下：

```
<img src="figures/bird.jpg" alt="飞翔的小鸟"
draggable="true" ondragstart="drag(event)">
```

拖放事件的处理函数

- drag(event)函数的格式如下：

```
<script type="text/javascript">
    function drag(event){
        // 处理dragstart事件的代码
    }
</script>
```

每个拖放事件的处理函数都有一个Event对象作为参数。Event 对象代表事件的状态，比如发生事件中的元素、键盘按键的状态、鼠标的位置、鼠标按钮的状态。

9.2 传递拖拽数据

- data Transfer对象的属性
- data Transfer对象的方法

data Transfer对象的属性

dropEffect属性

取值	说　明
copy	显示copy光标
link	显示link光标
move	显示move光标
none	默认值，即没有指定光标

data Transfer对象的属性

effectAllowed属性

取值	说　明
copy	允许执行复制操作
link	将源对象链接到目的地
move	将源对象移动到目的地
copyLink	可以是copy或link，取决于目标对象的缺省值
copyMove	可以是copy或move，取决于目标对象的缺省值
linkMove	可以是link或move，取决于目标对象的缺省值
all	允许所有数据传输操作

data Transfer对象的方法

getData()方法

getData()方法用于从dataTransfer对象中以指定的格式获取数据，语法如下：

```
sretrievedata = object.getdata(sdataformat)
```

参数sdataformat是指定数据格式的字符串，可以是下面的值：

- Text，以文本格式获取数据。
- URL，以URL格式获取数据。

getData()方法的返回值是从dataTransfer对象中获取的数据。

data Transfer对象的方法

setData()方法

setData ()方法用于以指定的格式设置dataTransfer对象中的数据，语法如下：

```
bsuccess = object.setdata(sdataformat, sdata)
```

参数sdataformat是指定数据格式的字符串，可以是下面的值：

- Text，以文本格式保存数据。
- URL，以URL格式保存数据。

参数sdata是指定要设置的数据的字符串。

如果设置数据成功，则setData ()方法返回True；否则返回False。

9.3 HTML5拖放的实例

- 拖放HTML元素
- 拖放外部图片文件

拖放HTML元素

本示例将实现HTML页面中<img>元素的拖放。

HTML代码如下：

```
<body>
    <p>请把图片拖放到矩形中：</p>
    <!-- 用于接收被拖动的img元素 -->
    <div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"></div>
    <br />
    <!-- 设置元素为可拖放 -->
    <img id="drag1" src="bird1.jpg" draggable="true"
        ondragstart="drag(event)" width="200px"/>
</body>
```

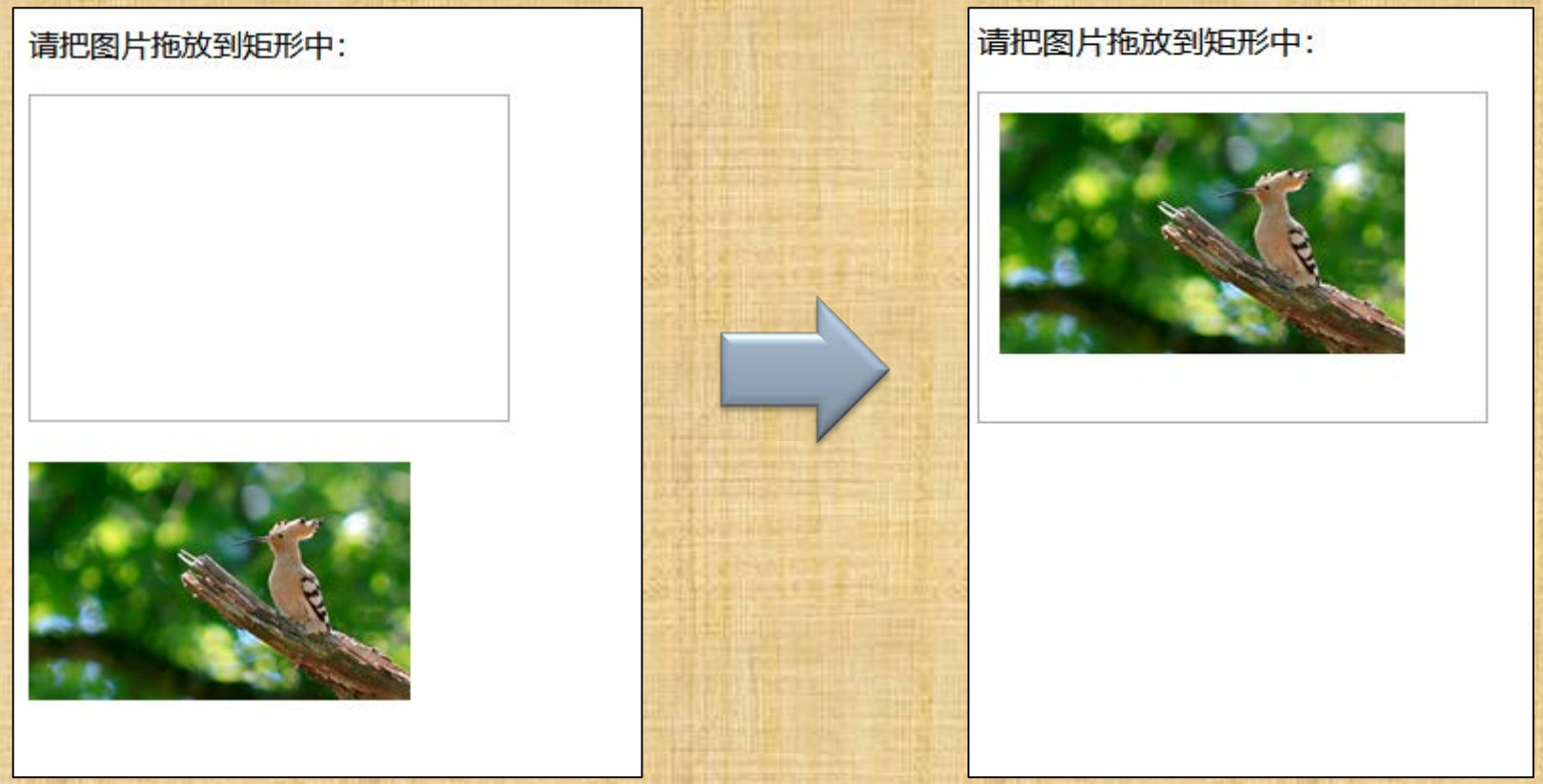
拖放HTML元素

JavaScript 代码：

```
<script type="text/javascript">
    function allowDrop(ev)
    {
        ev.preventDefault();
    }
    // 设置被拖数据的数据类型和值
    function drag(ev)
    {
        ev.dataTransfer.setData("Text",ev.target.id);
    }
    // 当放置被托数据时，会发生drop事件
    function drop(ev)
    {
        // 避免浏览器对数据默认处理
        ev.preventDefault();
        // 获得被托的数据
        var data=ev.dataTransfer.getData("Text");
        // 把被托目标放在目标元素中
        ev.target.appendChild(document.getElementById(data));
    }
</script>
```

拖放HTML元素

实现效果：



拖放图片文件

本实例主要实现将外部图片文件拖放到HTML页面内：

HTML代码如下：

```
<!-- 用来存放被拖放的图片容器。 -->
<div id="imgs" ondragover="allowDrop(event)"
ondrop="dropImg(event)">
```

拖放图片文件

JavaScript代码：

```
//禁用掉浏览器掉默认行为
function allowDrop(e) {
    e.preventDefault();
}
//当拖放结束时调用:
function dropImg(e) {
    //禁用掉浏览器掉默认行为
    e.preventDefault();
    //data为获取到的文件，只能在ondrop中得到
    var data = e.dataTransfer.files;
```

```
//获取多个文件时，遍历文件，判断文件是否为我们所要求的类型，并做出处理
for (var i = 0; i < data.length; i++) {
    //判断文件类型,indexOf()的结果假为-1,真为0
    var myType = data[i].type;
    console.log(myType.indexOf('image'));
    if (myType.indexOf('image') === 0) {
        //FileReader为html5中封装的方法，用于将文件读入内存，并读取文件中的数据
        var reader = new FileReader();
        //读取拖入文件的DataURL,无返回值。
        reader.readAsDataURL(data[i]);
        //读取文件成功时触发
        reader.onload = function () {
            //this.result为当前文件的base64解码
            //console.log(this.result);
            //创建一个img节点并添加到当前框内
            var img = document.createElement("img");
            img.src = this.result;
            img.className = "myImg";
            document.getElementById("imgs").appendChild(img);
        };
```

```
        //无论是否成功读取时都会触发，用来弹出错误或上传数据
        reader.onloadend = function () {
            //如果上传出错
            if (reader.error) {
                alert(reader.error);
            } else {
                //可以进行与服务器的上传交互
            }
        }
    } else {
        //如果传入的非图片格式
        alert("请上传图片！");
    }
}
```

拖放图片文件

CSS代码：

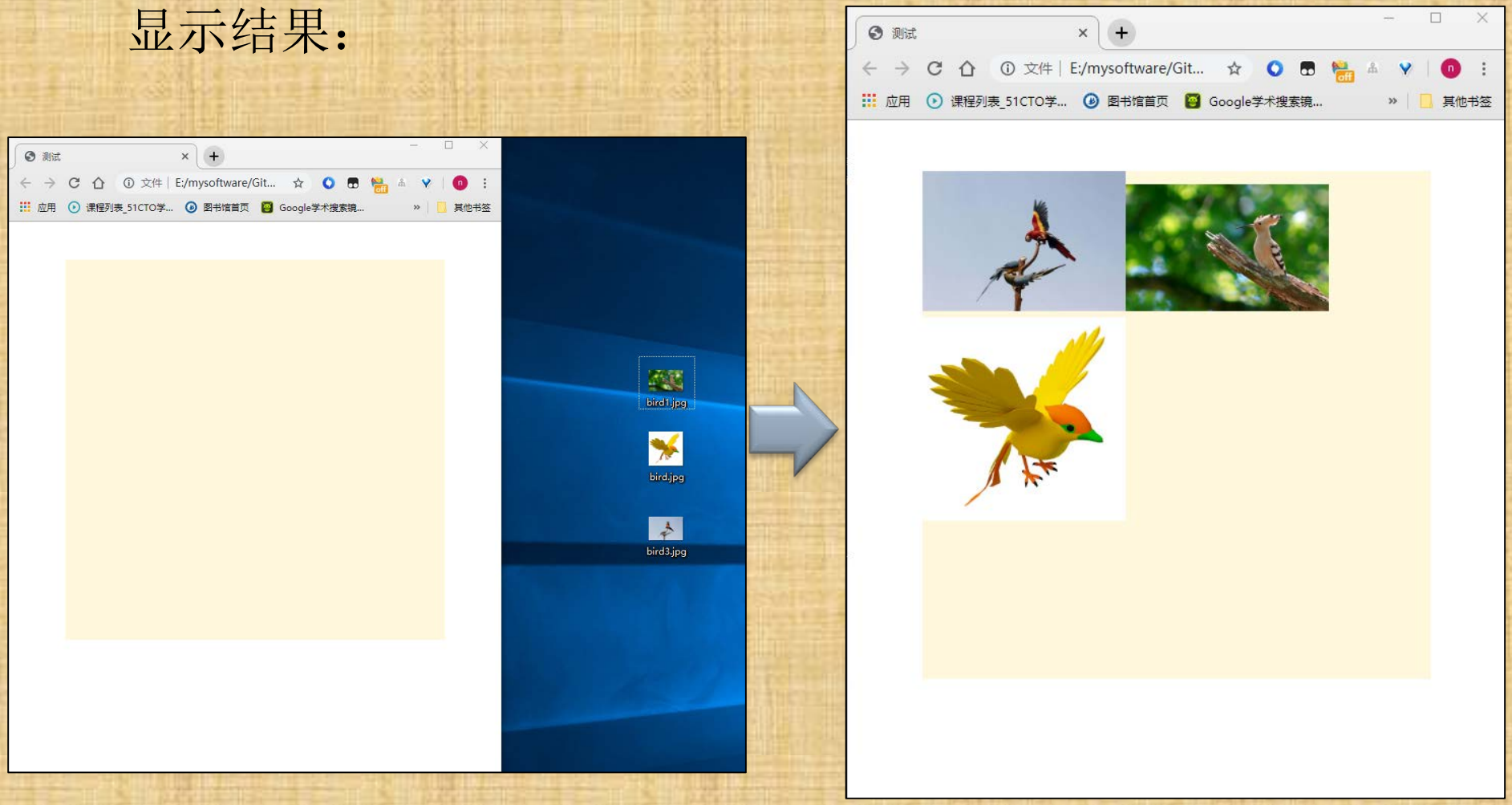
```css
* {
    margin: 0;
    padding: 0;
}

.myImg {
    width: 200px;
}

#imgs {
    width: 500px;
    height: 500px;
    background-color: cornsilk;
    margin: 50px auto;
}
```

拖放文件

显示结果：


测试
文件 | E:/mysoftware/Git...
应用
课程列表_51CTO学...
图书馆首页
Google学术搜索镜...
其他书签
bird1.jpg
bird.jpg
bird3.jpg