**1 请简述三个HTML5中的表单元素及其作用**

**参考答案**

<input type="email" />作用：用来输入Email地址，它会验证文本框的内容是否为合法的Email地址。

<input type="url" />作用：用来输入Url地址，它会验证文本框的内容是否为合法的Url地址

<input type="datetime" />作用：输入时间和日期的文本框

**2 编写一个文本框，可以让用户输入或选择职业，效果如图-1所示**

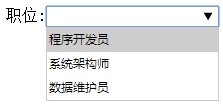


图-1

**参考答案**

1. **<**form action**=**"regist.do" method**=**"post"**>**
2. 职位**:<**input type**=**"text" id**=**"txtWork" list**=**"lstWork" **/>**
3. **<**datalist id**=**"lstWork"**>**
4. **<**option value**=**"程序开发员"**></**option**>**
5. **<**option value**=**"系统架构师"**></**option**>**
6. **<**option value**=**"数据维护员"**></**option**>**
7. **</**datalist**>**
8. **</**form**>**

**3 编写一个文本框，要求只能输入字母、数字以及下划线，长度为6-12位，效果如图-2所示**

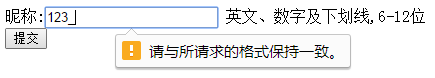


图-2

**参考答案**

1. **<**form action**=**"regist.do" method**=**"post"**>** 昵称**:<**input type**=**"text" id**=**"uname" required pattern**=**"[a-zA-Z0-9\_]{6,12}" /> <span id="unameShow">英文、数字及下划线,6-12位</span**>** **<**br **/>** **<**input type**=**"submit" value**=**"提交" /> </form**>**

**4 验证文本框的录入**

页面上有文本框用于录入用户名，效果如图－3所示。

http://pdf7.tarena.com.cn/tts8_source/ttsPage/WEB/WEB_A_V02/HTML5CORE/DAY01/EXERCISE/01/index.files/image003.png

图－3

录入完毕后，验证文本框中的录入，要求只能输入字母、数字以及下划线，且长度为6-12位。如果录入错误，在文本框后面显示自定义的提示信息，而不是系统默认的弹框提示效果，效果如图-4所示：

http://pdf7.tarena.com.cn/tts8_source/ttsPage/WEB/WEB_A_V02/HTML5CORE/DAY01/EXERCISE/01/index.files/image004.png

图－4

**参考答案**

页面代码如下所示：

1. **<**form id**=**"myForm"**>**
2. **<**input type**=**"text" id**=**"uname" required pattern**=**"[a-zA-Z0-9\_]{6,12}"**>**
3. **<**span id**=**"unameShow"**>**英文、数字及下划线，**6-12**位**</**span**>**
4. **<**br **/>**
5. **<**input type**=**"button" value**=**"提交" onclick**=**"onSubmit()"**>**
6. **</**form**>**
7. **<**script type**=**"text/javascript"**>**
8. **var** uname **=** document**.**getElementById**(**"uname"**);**
9. **function** onSubmit**(){**
10. **if(**uname**.**validity**.**valid**){**
11. document**.**getElementById**(**"myForm"**).**submit**();**
12. **}else{**
13. **var** unameShow **=** document**.**getElementById**(**"unameShow"**);**
14. **if(**uname**.**validity**.**valueMissing**){**
15. unameShow**.**innerHTML **=** "请输入用户名"**;**
16. **}else** **if(**uname**.**validity**.**patternMismatch**){**
17. unameShow**.**innerHTML **=** "用户名应该是6-12位英文、数字及下划线的组合"**;**
18. **}**
19. **}**
20. **}**
21. **</**script**>**

**5 请简述setCustomValidity方法的作用**

**参考答案**

HTML5表单验证给前端开发带来了许多便利，但是默认提示不是很友好。我们可以通过setCustomValidity方法来自定义提示信息，更准确的提示给用户。

**1 以下哪些标签能够在页面中播放视频( )**

A、<embed>

B、<mp4>

C、<video>

D、<audio>

**参考答案**

答案：AC

A答案，可以播放视频，但不符合W3C规范，无法通过验证，不推荐使用。当然，如果不在乎规范的话，那么就另当别论了。

B答案，在HTML5中根本就不存在<mp4>标签

C答案，标准的HTML5的视频播放标签，可以支持mp4以及ogg格式的视频播放

D答案，是播放音频的

**2 编程实现自选视频文件进行播放**

用户可以自己先择文件进行播放，效果如下图所示:



图-1

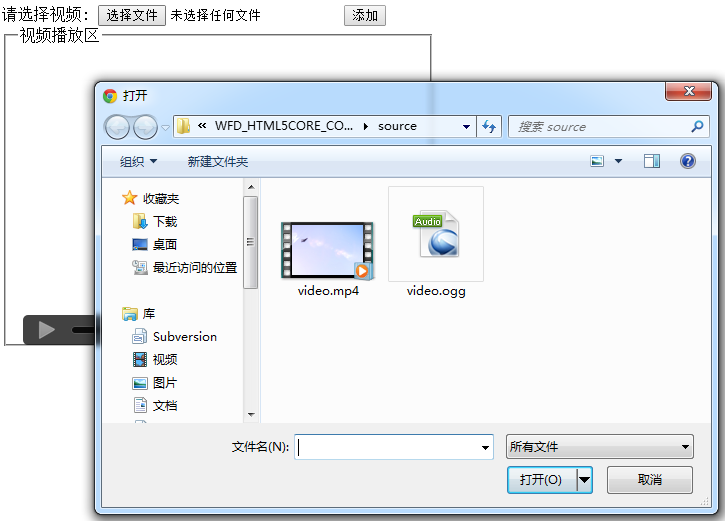


图-2



图-3

**参考答案**

此案例用户可以自己选择视频文件的播放，所以需要提供一个文件选择框。当用户选择好文件后点击添加时，可以将选择好的视频显示在视频播放区，并可以进行播放。

实现此案例，需要按如下步骤进行

**步骤一：制作页面**

在页面上添加文件框、按钮等元素，代码如下

1. **<**html lang**=**"en"**>**
2. **<**head**>**
3. **<**meta charset**=**"utf-8" **/>**
4. **<**style**>**
5. fieldset**{**width**:**400px**;**height**:**300px**;}**
6. **</**style**>**
7. **</**head**>**
8. **<**body**>**
9. 请选择视频**:**
10. **<**input type**=**'file' id**=**'fileVideo' **/>**
11. **<**input type**=**'button' id**=**'btn' value**=**'添加' **/>**
12. **<**fieldset**>**
13. **<**legend**>**视频播放区**</**legend**>**
14. **<**video id**=**'myVideo' controls width**=**'400' height**=**'300'**></**video**>**
15. **</**fieldset**>**
16. **</**body**>**
17. **</**html**>**

**步骤二：添加初始化方法以及封装根据ID获取元素方法**

通过$()函数获取指定id所对应的元素，并且获取fileVideo元素以及btn元素，为btn元素添加事件,代码如下所示：

1. **<**script**>**
2. //封装方法，方便获取指定id的元素
3. **function** $**(**id**){**
4. returndocument**.**getElementById**(**id**);**
5. **}**
6. **function** initial**(){**
7. fileVideo**=**$**(**'fileVideo'**),**btn**=**$**(**'btn'**);**
8. btn**.**addEventListener**(**'click'**,**btnClick**,false);**
9. **}**
10. window**.**addEventListener**(**'load'**,**initial**,false);**
11. **</**script**>**

**步骤三：实现btnClick方法**

通过btnClick方法完成btn元素的click事件，在方法中可以获取用户选择的文件的路径，但是由于浏览器对文件选择的操作有一个默认的地址路径，所以我们这里就将文件先拷贝到一个指定的路径，然后在程序中主要通过指定路径名称，再加上文件名字来拼凑完整的视频路径，代码如下所示：

1. functionbtnClick**(**event**){**
2. varfile**=**fileVideo**.**value**;**
3. varurl **=** file**.**substring**(**file**.**lastIndexOf**(**'**\\**'**)+1);**
4. $**(**'myVideo'**).**src**=**'source/'**+**url**;**
5. **}**

本案例完整代码如下:

1. **<!**DOCTYPE html**>**
2. **<**html lang**=**"en"**>**
3. **<**head**>**
4. **<**meta charset**=**"utf-8" **/>**
5. **<**script**>**
6. //封装方法，方便获取指定id的元素
7. **function**$**(**id**){**
8. returndocument**.**getElementById**(**id**);**
9. **}**
11. functioninitial**(){**
12. fileVideo**=**$**(**'fileVideo'**),**btn**=**$**(**'btn'**);**
13. btn**.**addEventListener**(**'click'**,**btnClick**,false);**
14. **}**
16. functionbtnClick**(**event**){**
17. varfile**=**fileVideo**.**value**;**
18. varurl **=** file**.**substring**(**file**.**lastIndexOf**(**'**\\**'**)+1);**
19. $**(**'myVideo'**).**src**=**'source/'**+**url**;**
20. **}**
22. window**.**addEventListener**(**'load'**,**initial**,false);**
23. **</**script**>**
24. **<**style**>**
25. fieldset**{**width**:**400px**;**height**:**300px**;}**
26. **</**style**>**
27. **</**head**>**
28. **<**body**>**
29. 请选择视频**:**
30. **<**input type**=**'file' id**=**'fileVideo' **/>**
31. **<**input type**=**'button' id**=**'btn' value**=**'添加' **/>**
32. **<**fieldset**>**
33. **<**legend**>**视频播放区**</**legend**>**
34. **<**video id**=**'myVideo' controls width**=**'400' height**=**'300'**></**video**>**
35. **</**fieldset**>**
36. **</**body**>**
37. **</**html**>**

**3 简述HTML5 Canvas元素的作用**

**参考答案**

Cavas元素用于在网页上绘制图形，该元素标签的强大之处在于可以直接在HTML上进行绘图。通过调用canvas对象的getContext方法来获取绘图环境。getContext只需要一个参数：绘图环境的类型。在游戏中，通常使用2d类型的绘图环境。常用代码如下所示：

1. **<**canvas id**=**"canvas1"**></**canvas**>**
2. **<**script**>**
3. **function** getCanvas**(){**
4. elem**=**document**.**getElementById**(**"canvas1"**);**
5. canvas**=**elem**.**getContext**(**'2d'**);**
6. **}**
7. **</**script**>**

**1 在网页上实现滚动的小球**

在网页上实现一个小球在不停的滚动，碰到边缘后进行反弹。大体效果如图所示。



图-1

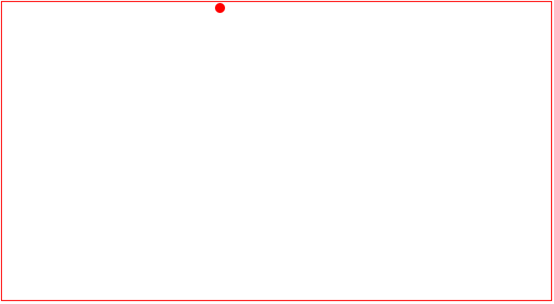


图-1

**参考答案**

**步骤一：创建页面并绘制小球**

创建页面，并添加canvas标签以及样式，代码如下

1. **<**html lang**=**"en"**>**
2. **<**head**>**
3. **<**meta charset**=**"utf-8" **/>**
4. **<**style**>**
6. canvas**{**border**:**1px solid red**;}**
7. **</**style**>**
8. **</**head**>**
9. **<**body**>**
10. **<**canvas id**=**"canvas" width**=**"554" height**=**"300"**></**canvas**>**
11. **</**body**>**
12. **</**html**>**

**步骤二：根据指定坐标，绘制小球**

当前程序要根据位置不停的创建小球，所以创建一个方法，专门绘制小球。该方法要接收两个参数x和y，用来表示小球的横纵坐标。小球的半径可以固定为5。代码如下所示。

1. **function** drawArc**(**x**,**y**){**
2. //自定义画圆方法
3. context2D**.**fillStyle**=**"red"**;**
4. context2D**.**beginPath**();**
5. context2D**.**arc**(**x**,**y**,5,0,**Math**.**PI **\*** **2** **,false);**
6. context2D**.**fill**();**
7. context2D**.**closePath**();**
8. **}**

**步骤三：计算小球的新位置，重新绘制小球**

定义全局的x,y变量，表示小球的横纵坐标，从0开始。dx,dy分别表示小球的横向步长以及纵向步长，context2D表示待操作的canvas画布。创建一个方法，该方法用于根据画布的边界来重新计算小球的坐标，并重新调用刚才创建好的绘制小球的方法。由于重复调用绘制小球的方法会在画布上不停的画小球，这样的话会在画布上画多个小球。所以每次重新绘制小球的时候，需要将画布上的东西擦除掉再重新绘制。详细代码如下所示。

1. **<**script language**=**"javascript"**>**
2. **var** context2D**;**
3. //小球从(0,0)开始移动，横向步长为2，纵向步长为3
4. **var** x**=0,**y**=0,**dx**=2,**dy**=3;**
5. **function** draw**(){**
6. context2D**.**fillStyle**=**'red'**;**
7. context2D**.**clearRect**(0,0,**canvas**.**width**,**canvas**.**height**);**
8. //判断边界
9. **if(**x **+** dx **>** canvas**.**width **||** x **+** dx **<** **0)**
10. dx **\*=** **-1;**
11. **if(**y **+** dy **>** canvas**.**height **||** y **+** dy **<** **0)**
12. dy **\*=** **-1;**
13. x **+=** dx**;**
14. y **+=** dy**;**
15. drawArc**(**x**,**y**);**
16. **}**
17. **</**script**>**

**步骤四：创建初始化函数**

通过初始化函数获取canvas的context，并且通过计时器控件实现循环调用绘图方法。详细代码如下所示。

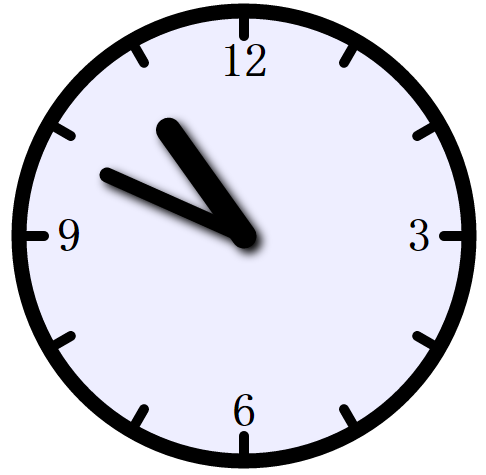
1. window**.**onload **=** **function(){**
2. canvas**=**document**.**getElementById**(**'canvas'**);**
3. context2D**=**canvas**.**getContext**(**'2d'**);**
4. setInterval**(**draw**,20);**
5. **}**

本案例完整代码如下：

1. **<!**DOCTYPE html**>**
2. **<**html lang**=**"en"**>**
3. **<**head**>**
4. **<**meta charset**=**"utf-8" **/>**
5. **<**script language**=**"javascript"**>**
6. **var** context2D**;**//小球从(0,0)开始移动，横向步长为2，纵向步长为3
7. **var** x**=0,**y**=0,**dx**=2,**dy**=3;**
8. **function** draw**(){**
9. context2D**.**fillStyle**=**'red'**;**
10. context2D**.**clearRect**(0,0,**canvas**.**width**,**canvas**.**height**);**
11. //判断边界
12. **if(**x **+** dx **>** canvas**.**width **||** x **+** dx **<** **0)**
13. dx **\*=** **-1;**
14. **if(**y **+** dy **>** canvas**.**height **||** y **+** dy **<** **0)**
15. dy **\*=** **-1;**
16. x **+=** dx**;**
17. y **+=** dy**;**
18. drawArc**(**x**,**y**);**
19. **}**
20. **function** drawArc**(**x**,**y**){**
21. //自定义画圆方法
22. context2D**.**fillStyle**=**"red"**;**
23. context2D**.**beginPath**();**
24. context2D**.**arc**(**x**,**y**,5,0,**Math**.**PI **\*** **2** **,false);**
25. context2D**.**fill**();**
26. context2D**.**closePath**();**
27. **}**
28. window**.**onload **=** **function(){**
29. canvas**=**document**.**getElementById**(**'canvas'**);**
30. context2D**=**canvas**.**getContext**(**'2d'**);**
31. setInterval**(**draw**,20);**
32. **}**
33. **</**script**>**
34. **<**style**>**
35. canvas**{**border**:**1px solid red**;}**
36. **</**style**>**
37. **</**head**>**
38. **<**body**>**
39. **<**canvas id**=**"canvas" width**=**"554" height**=**"300"**></**canvas**>**
40. **</**body**>**
41. **</**html**>**

**1 使用SVG绘制钟表**

使用 SVG绘制一个钟表，其效果如图－1所示。并通过JS控制钟表的运行。



图－1

**参考答案**

页码代码如下所示：

1. **<!**DOCTYPE HTML**>**
2. **<**html**>**
3. **<**head**>**
4. **<**title**>**Clock**</**title**>**
5. **<**script**>**
6. **function** updateTime**()** **{**
7. // 更新钟表时间
8. **var** now **=** **new** Date**();** // 当前时间
9. **var** min **=** now**.**getMinutes**();** // 分钟数
10. **var** hour **=** **(**now**.**getHours**()** **%** **12)** **+** min**/60;** // 小时数
11. **var** minangle **=** min**\*6;** // 每分钟6度
12. **var** hourangle **=** hour**\*30;** // 每小时30度
13. // 获取钟表指针的SVG元素
14. **var** minhand **=** document**.**getElementById**(**"minutehand"**);**
15. **var** hourhand **=** document**.**getElementById**(**"hourhand"**);**
16. // 设置相应SVG元素的transform属性控制表针移动
17. minhand**.**setAttribute**(**"transform"**,** "rotate(" **+** minangle **+** ",50,50)"**);**
18. hourhand**.**setAttribute**(**"transform"**,** "rotate(" **+** hourangle **+** ",50,50)"**);**
19. // 每分钟更新一次
20. setTimeout**(**updateTime**,** **60000);**
21. **}**
22. **</**script**>**
23. **<**style**>**
24. /\* 钟表样式 \*/
25. #clock **{**
26. stroke**:** black**;**
27. stroke**-**linecap**:** round**;**
28. fill**:** #eef**;**
29. **}**
30. #face **{** stroke**-**width**:** 3px**;}** /\* 表面边框 \*/
31. #ticks **{** stroke**-**width**:** **2;** **}** /\* 刻度 \*/
32. #hourhand **{**stroke**-**width**:** 5px**;}** /\* 时针 \*/
33. #minutehand **{**stroke**-**width**:** 3px**;}** /\* 分针 \*/
34. #numbers **{**
35. /\* 表盘数字 \*/
36. font**-**family**:** sans**-**serif**;** font**-**size**:** 7pt**;** font**-**weight**:** bold**;**
37. text**-**anchor**:** middle**;** stroke**:** none**;** fill**:** black**;**
38. **}**
39. **</**style**>**
40. **</**head**>**
41. **<**body onload**=**"updateTime()"**>**
42. **<**svg id**=**"clock" viewBox**=**"0 0 100 100" width**=**"500" height**=**"500"**>**
43. **<!--** 表盘 **-->**
44. **<**circle id**=**"face" cx**=**"50" cy**=**"50" r**=**"45"**/>**
45. **<!--** 刻度线 **-->**
46. **<**g id**=**"ticks"**>**
47. **<**line x1**=**'50' y1**=**'5.000' x2**=**'50.00' y2**=**'10.00'**/>**
48. **<**line x1**=**'72.50' y1**=**'11.03' x2**=**'70.00' y2**=**'15.36'**/>**
49. **<**line x1**=**'88.97' y1**=**'27.50' x2**=**'84.64' y2**=**'30.00'**/>**
50. **<**line x1**=**'95.00' y1**=**'50.00' x2**=**'90.00' y2**=**'50.00'**/>**
51. **<**line x1**=**'88.97' y1**=**'72.50' x2**=**'84.64' y2**=**'70.00'**/>**
52. **<**line x1**=**'72.50' y1**=**'88.97' x2**=**'70.00' y2**=**'84.64'**/>**
53. **<**line x1**=**'50.00' y1**=**'95.00' x2**=**'50.00' y2**=**'90.00'**/>**
54. **<**line x1**=**'27.50' y1**=**'88.97' x2**=**'30.00' y2**=**'84.64'**/>**
55. **<**line x1**=**'11.03' y1**=**'72.50' x2**=**'15.36' y2**=**'70.00'**/>**
56. **<**line x1**=**'5.000' y1**=**'50.00' x2**=**'10.00' y2**=**'50.00'**/>**
57. **<**line x1**=**'11.03' y1**=**'27.50' x2**=**'15.36' y2**=**'30.00'**/>**
58. **<**line x1**=**'27.50' y1**=**'11.03' x2**=**'30.00' y2**=**'15.36'**/>**
59. **</**g**>**
60. **<!--** 表盘数字**-->**
61. **<**g id**=**"numbers"**>**
62. **<**text x**=**"50" y**=**"18"**>12<**/text><text x="85" y="53">3</text**>**
63. **<**text x**=**"50" y**=**"88"**>6<**/text><text x="15" y="53">9</text**>**
64. **</**g**>**
65. **<!--** 表针 **-->**
66. **<**g id**=**"hands"**>**
67. **<**line id**=**"hourhand" x1**=**"50" y1**=**"50" x2**=**"50" y2**=**"24"**/>**
68. **<**line id**=**"minutehand" x1**=**"50" y1**=**"50" x2**=**"50" y2**=**"20"**/>**
69. **</**g**>**
70. **</**svg**>**
71. **</**body**>**
72. **</**html**>**

**1 简述HTML5中Geolocation API的三个方法的作用和使用方法。**

**参考答案**

getCurrentPosition(success,error,opts)：该方法用于获取用户当前所在位置。success参数是获取定位成功后的回调函数，该函数有一个position参数，可以通过其coords属性得到经纬度、海拔、速度和精度等信息；error可选参数是获取定位失败后的回调函数，该函数有一个error对象，里面有详细的错误信息；opts可选参数是可以指定是否需要高精度位置信息、该位置信息的过期时间及超时时间等。

watchPosition(success,error,opts)：该方法和getCurrentPosition方法非常相似，但其可以不断的监视当前位置，一旦用户位置发生更改，就会调用指定回调函数。

clearWatch方法配合watchPosition使用，在不需要位置信息的时候调用停止对位置的监视。

**2 完成如下图所示的页面，并实现能够将列表拖进垃圾箱的操作**



图-1

图-1中要完成的效果为用户可以拖拽列表1-列表6中的任何一个列表到垃圾箱中。拖拽进垃圾箱中的列表从页面上消失。

**参考答案**

**步骤一：创建页面**

首先创建一个纯文本文件，并修改名称为drag\_remove.html,然后在文本编辑工具中打开此文件，为其添加html、css代码,完成页面效果。代码如下所示：

1. **<!**DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**>**
2. **<**html xmlns**=**"http://www.w3.org/1999/xhtml"**>**
3. **<**head**>**
4. **<**meta http**-**equiv**=**"Content-Type" content**=**"text/html; charset=utf-8" **/>**
5. **<**title**>**html5 drag **&**amp**;** drop 拖拽与拖放测试**</**title**>**
6. **<**style**>**
7. body **{**
8. font**-**size**:** **84%;**
9. **}**
10. #dustbin **{**
11. width**:** 100px**;**
12. height**:** 260px**;**
13. line**-**height**:** **1.4;**
14. background**-**color**:** gray**;**
15. font**-**size**:** 36px**;**
16. font**-**family**:** "微软雅黑"**,** "Yahei Mono"**;**
17. text**-**align**:** center**;**
18. text**-**shadow**:** **-**1px **-**1px #bbb**;**
19. float**:** left**;**
20. **}**
21. **.**dragbox **{**
22. width**:** 500px**;**
23. padding**-**left**:** 20px**;**
24. float**:** left**;**
25. **}**
26. **.**draglist **{**
27. padding**:** 10px**;**
28. margin**-**bottom**:** 5px**;**
29. border**:** 2px dashed #ccc**;**
30. background**-**color**:** #eee**;**
31. cursor**:** move**;**
32. **}**
33. **.**draglist**:**hover **{**
34. border**-**color**:** #cad5eb**;**
35. background**-**color**:** #f0f3f9**;**
36. **}**
37. #dragremind **{**
38. padding**-**top**:** 2em**;**
39. clear**:** both**;**
40. **}**
41. **</**style**>**
42. **</**head**>**
43. **<**body**>**
44. **<**div id**=**"dustbin"**>**
45. **<**br **/>**
46. 垃
47. **<**br **/>**
48. 圾
49. **<**br **/>**
50. 箱
51. **</**div**>**
52. **<**div **class=**"dragbox"**>**
53. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
54. 列表**1**
55. **</**div**>**
56. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
57. 列表**2**
58. **</**div**>**
59. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
60. 列表**3**
61. **</**div**>**
62. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
63. 列表**4**
64. **</**div**>**
65. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
66. 列表**5**
67. **</**div**>**
68. **<**div **class=**"draglist" title**=**"拖拽我" draggable**=**"true"**>**
69. 列表**6**
70. **</**div**>**
71. **</**div**>**
72. **<**div id**=**"dragremind"**></**div**>**
73. **</**body**>**
74. **</**html**>**

**步骤二：在</body>之上添加<script></script>代码段，处理拖拽过程**

1. **<**script**>**
2. //获取所有能被拖拽的div元素
3. **function** getDrags**(){**
4. **var** drags **=** document**.**getElementsByTagName**(**"div"**);**
5. **var** dragList **=** **new** Array**();**
6. **for(var** i**=0;**i**<**drags**.**length**;**i**++){**
7. **var** drag**=**drags**[**i**];**
8. **if(**drag**.**className**==**'draglist'**){**
9. dragList**.**push**(**drag**);**
10. **}**
11. **}**
12. **return** dragList**;**
13. **}**
14. //获取页面的其他元素
15. **var** eleDustbin **=** document**.**getElementById**(**'dustbin'**);**
16. **var** eleRemind**=**document**.**getElementById**(**'dragremind'**);**
17. **var** eleDrags**=**getDrags**();**
18. **var** lDrags**=**eleDrags**.**length**;**
19. **var** eleDrag **=** **null;**
21. **for** **(var** i **=** **0;** i **<** lDrags**;** i **+=** **1)** **{**
22. eleDrags**[**i**].**onselectstart **=** **function()** **{**
23. **return** **false;**
24. **};**
25. eleDrags**[**i**].**ondragstart **=** **function(**ev**)** **{**
26. ev**.**dataTransfer**.**effectAllowed **=** "move"**;**
27. ev**.**dataTransfer**.**setData**(**"text"**,** ev**.**target**.**innerHTML**);**
28. ev**.**dataTransfer**.**setDragImage**(**ev**.**target**,** **0,** **0);**
29. eleDrag **=** ev**.**target**;**
30. **return** **true;**
31. **};**
32. eleDrags**[**i**].**ondragend **=** **function(**ev**)** **{**
33. ev**.**dataTransfer**.**clearData**(**"text"**);**
34. eleDrag **=** **null;**
35. **return** **false**
36. **};**
37. **}**
38. eleDustbin**.**ondragover **=** **function(**ev**)** **{**
39. ev**.**preventDefault**();**
40. **return** **true;**
41. **};**
42. eleDustbin**.**ondragenter **=** **function(**ev**)** **{**
43. **this.**style**.**color **=** "#ffffff"**;**
44. **return** **true;**
45. **};**
46. eleDustbin**.**ondrop **=** **function(**ev**)** **{**
47. **if** **(**eleDrag**)** **{**
48. eleRemind**.**innerHTML **=** '<strong>"' **+** eleDrag**.**innerHTML **+** '"</strong>被扔进了垃圾箱'**;**
49. eleDrag**.**parentNode**.**removeChild**(**eleDrag**);**
50. **}**
51. **this.**style**.**color **=** "#000000"**;**
52. **return** **false;**
53. **};**
54. **</**script**>**

### 1 简述WEB Worker的作用和使用方法。

**参考答案**

Web Workers是运行在后台的JavaScript，通过WEB Worker可以实现HTML5页面中的多线程，充分利用多核CPU的优势，能够在独立运行大量计算任务的同时不影响页面的交互性，提高用户体验。

使用WEB Worker的方法是将独立任务代码置于单独JS文件中，然后通过new Worker(url)的方式创建独立线程，使用过程中通过postMessage方法和message事件进行主线程和子线程之间的通信。

### 2 改进模拟售票系统，将抢票成功率和点击窗口的速度挂钩，点击速度越快抢票成功率越高。

**参考答案**

worker内部代码：

1. **var** refreshTime**;**
2. //等待刷票或购票命令
3. addEventListener**(**"message"**,function(**e**){**
4. **switch** **(**e**.**data**.**cmd**){**
5. **case** "refresh"**:**
6. //刷票指令：10%几率有票
7. **if(**Math**.**random**()<=0.1){**
8. //有票则发送1
9. postMessage**({**result**:1,**cmd**:**e**.**data**.**cmd**});**
10. //记录刷票时间，在用户点击购票的时候使用
11. refreshTime **=** **new** Date**();**
12. **}else{**
13. //没票发送0
14. postMessage**({**result**:0,**cmd**:**e**.**data**.**cmd**});**
15. **}**
16. **break;**
17. **case** "buy"**:**
18. //买票时间
19. **var** buyTime **=** **new** Date**();**
20. **var** useTime **=** buyTime**.**getTime**()** **-** refreshTime**.**getTime**();**
21. //购票指令：如果在700毫秒以内点击，有30%概率，如果在700-900，有15%概率，大于900毫秒抢不到
22. //抢到则发送1, 没抢到发送0
23. **if(**useTime **<=** **700){**
24. **if(**Math**.**random**()** **<=** **0.3)**
25. postMessage**({**result**:1,**cmd**:**e**.**data**.**cmd**});**
26. **else**
27. postMessage**({**result**:0,**cmd**:**e**.**data**.**cmd**});**
28. **}else** **if(**useTime **<=** **900** **&&** useTime **>** **700){**
29. **if(**Math**.**random**()** **<=** **0.15)**
30. postMessage**({**result**:1,**cmd**:**e**.**data**.**cmd**});**
31. **else**
32. postMessage**({**result**:0,**cmd**:**e**.**data**.**cmd**});**
33. **}else{**
34. postMessage**({**result**:0,**cmd**:**e**.**data**.**cmd**});**
35. **}**
36. **break;**
37. **}**
38. **},false);**

### 3 简述HTML5的存储类型

**参考答案**

HTML5能够本地存储数据，在这之前都是使用cookie的。HTML5提供了下面两种本地存储方案。

localStorage:用于持久化的本地存储，数据永远不会过期，关闭浏览器也不会丢失。

sessionStorage:同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是持久化的本地存储，仅仅是会话级别的存储。

### 4 编程统计一个用户对一个页面的访问次数

编写代码统计一个用户对一个页面的访问次数，即便关闭浏览器后，下次用户登录时数据依然会保留。

**参考答案**

1. **<**script type**=**"text/javascript"**>**
2. **if** **(**localStorage**.**pagecount**)** **{**
3. localStorage**.**pagecount **=** Number**(**localStorage**.**pagecount**)** **+** **1;**
4. **}** **else** **{**
5. localStorage**.**pagecount **=** **1;**
6. **}**
7. document**.**write**(**"Visits " **+** localStorage**.**pagecount **+** " time(s)."**);**
8. **</**script**>**

### 5 什么是WEB Socket？简要描述其特点。

**参考答案**

Web Socket是HTML5中一种新的通信协议，它实现了浏览器与服务器全双工通信。其特点是客户端和服务器之间的连接是实时的、永久的，双方都可以主动发送消息。客户端不再需要通过轮询实现实时性，性能优越。服务器端与客户端之间通信无需重新建立连接。