COMP 4021
Internet Computing

JavaScript
Part 3

David Rossiter

Revisiting HTML Tables

- Before we look at more JavaScript we need to consider some of the HTML we will need in the next lab

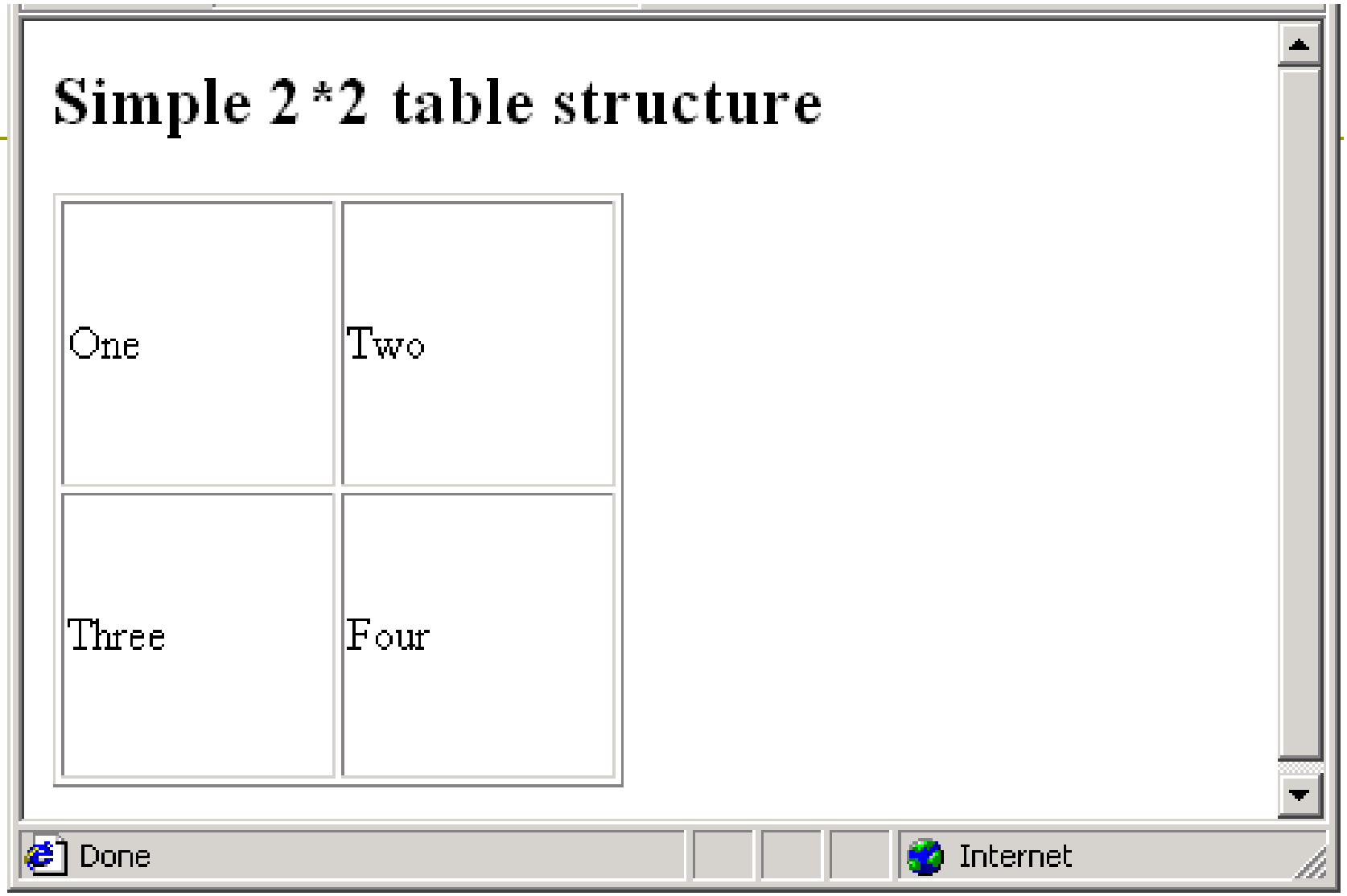
HTML Tables

```
<table border="1" width="200">
  <tr>
    <td height="100" width="100"> One </td>
    <td height="100" width="100"> Two </td>
  </tr>
  <tr>
    <td height="100" width="100"> Three </td>
    <td height="100" width="100"> Four </td>
  </tr>
</table>
```

This line of HTML defines one of the boxes in the row

There will be two rows

This HTML will set up a table with 2 rows, each row having 2 boxes


Simple 2*2 table structure
One
Two
Three
Four
Done
Internet

HTML Tables

- A table cell can contain anything – not just text
  - A cell can contain another table composed of multiple rows and columns
- For example, on the following slides images are put into the cells

HTML Tables

```
<h2>My favourite characters!</h2>
<table border="10" width="100">
  <tr>
    <td height="100" width="100"> 1.  <img src=
   "hello_kitty.gif" height="100" width="100"> </td>
</tr>
<tr>
   <td height="100" width="100"> 2. <img src="totoro.jpg"
   height="100" width="100"> </td>
</tr>
<tr>
   <td height="100" width="100"> 3.  <img src="mcmug.gif"
   height="100" width="100"> </td>
</tr>
</table>
```

HTML Tables

- The images have the same width because the width attribute of every row has been specified as 100pt

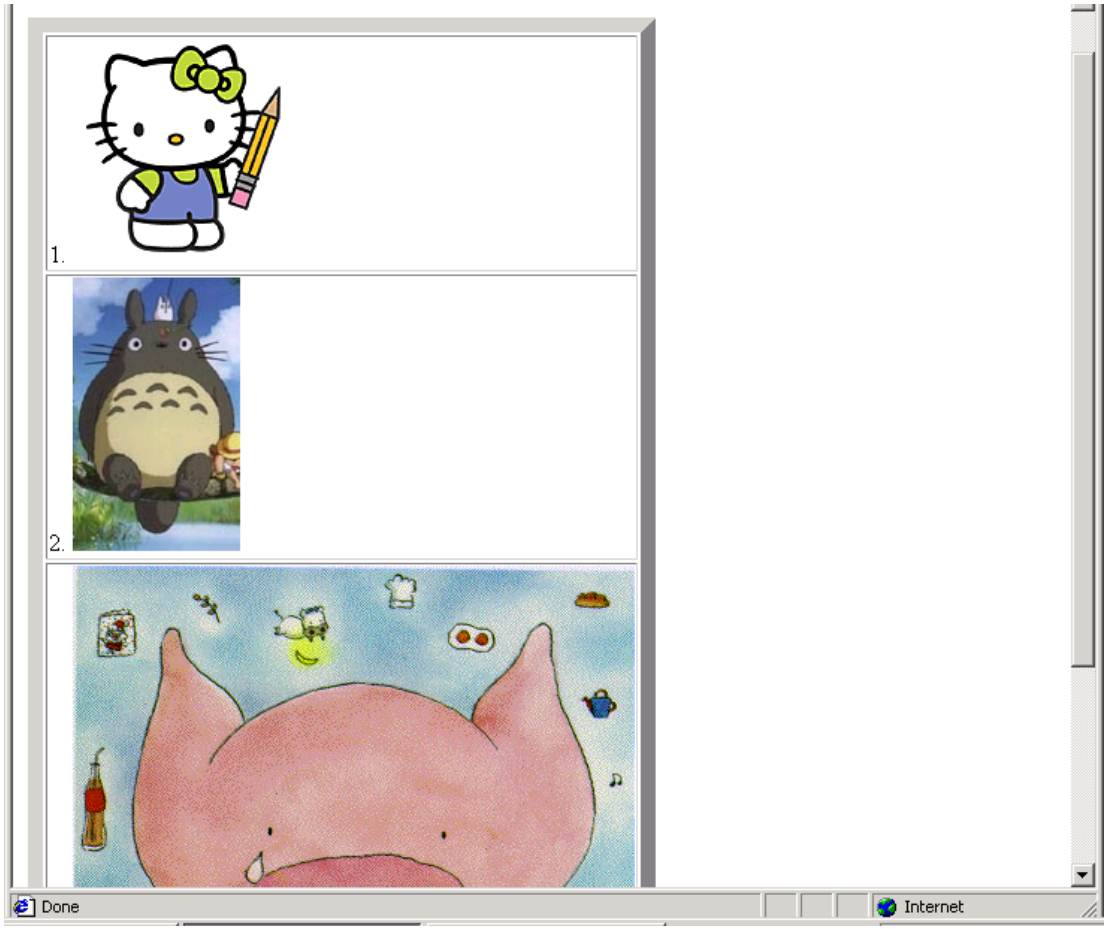
HTML Tables

What happens if we don't tell the browser to use a specific width/ height?

```
<h2>My favourite characters!</h2>
<table border="10">
   <tr>
    <td> 1.  <img src= "hello_kitty.gif"> </td>
 </tr>
 <tr>
   <td> 2. <img src= "totoro.jpg"> </td>
 </tr>
 <tr>
   <td> 3.  <img src= "mcmug.gif"> </td>
 </tr>
</table>
```

HTMLTables

- The images are shown at their ‘natural’ size
- Table is automatically expanded to fit them in



Revisiting Intervals

- Previously we looked at how to use timers/ intervals to tell the browser to do something at a later time
- Multiple timers/ intervals can be used concurrently

Two setTimeout Timers

```
<script language="JavaScript">

var wait_duration, timer1, timer2;

function set_things_up() {
        wait_duration=prompt("How long would you like to sleep?", "");
        timer1=setTimeout("show_wake_up_message()", wait_duration );

        wait_duration=prompt("How long until your next lecture?", "");
        timer2=setTimeout("show_lecture_message()", wait_duration );
}

function show_wake_up_message() {
        alert("WAKE UP! WAKE UP! WAKE UP!!");
}

function show_lecture_message() {
        alert("GO TO LECTURE! GO TO LECTURE!");
}
```

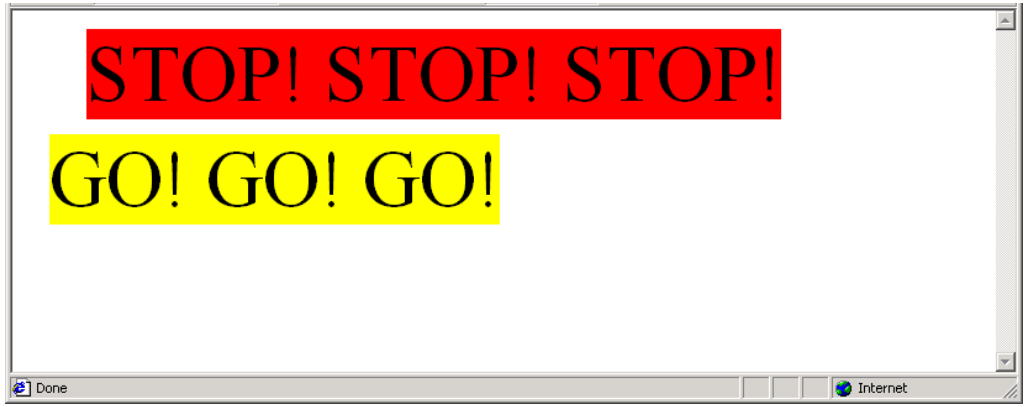
```
<body onload="set_things_up()“ >

<h1>Double alarm clock example</h1>

</body>
```

Another Two-Timer Example

- See the web site for an example of using two timers, each timer moving a different layer at a different speed



Key Events

- Previously we learnt about mouse events
- Now we consider key events
- For key events we are usually interested in knowing exactly which key has been pressed
- The way to handle this is a bit different to handling mouse events – for example, a keyboard event can't be applied to one particular object in the web page

Handling Key Events

- Whenever a key is pressed down when the web page is loaded the JavaScript function handle_key_press() will be executed

```
<body onkeydown="handle_key_press(event)">
. . .
</body>
```

Handling Key Presses

- The following function recognises what key has been pressed and react appropriately

```
function handle_key_press(key_event){
   var key_pressed_number, key_pressed_letter;

   key_pressed_number=key_event.keyCode;
   alert("The key you just pressed is key number " +
         key_pressed_number);

   key_pressed_letter=String.fromCharCode(key_pressed_number);
   alert("So that means that you pressed the "
                  + key_pressed_letter + " key");

   … do something depending on which key was pressed …
}
```

The Event Object

- Properties of the event object which are useful for handling key events:
- event.keyCode - returns value of key pressed
- event.shift - indicates whether “shift” is pressed
- event.ctrl - indicates whether “ctrl” is pressed
- event.alt - indicates whether “alt” is pressed

ASCII Table

- Key press values are ASCII

ASCII	Hex	Symbol
0	0	NUL
1	1	SOH
2	2	STX
3	3	ETX
4	4	EOT
5	5	ENQ
6	6	ACK
7	7	BEL
8	8	BS
9	9	TAB
10	A	LF
11	B	VT
12	C	FF
13	D	CR
14	E	SO
15	F	SI

ASCII	Hex	Symbol
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US

ASCII	Hex	Symbol
32	20	(space)
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/

ASCII	Hex	Symbol
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

ASCII	Hex	Symbol
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O

ASCII	Hex	Symbol
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

ASCII	Hex	Symbol
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o

ASCII	Hex	Symbol
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	▯

Complete Example

```
function handle_key_press(key_event){
var letter, para;

letter= String.fromCharCode(key_event.keyCode); // extract the
    letter
para=document.getElementById("output_paragraph"); // find
    the paragraph
para.innerHTML=letter;       // set the content of the paragraph
    to be the letter
}
```

```
<body
   onkeydown="handle_key_press(event)“ >
<p id="output_paragraph" style="font-size:96pt">
Please type a letter
</p>
</body>
```

Changing - .innerHTML

- From the previous example:

```
para=document.getElementById("output_paragraph"
   ); // find the element
para.innerHTML=letter;  // change the element
```

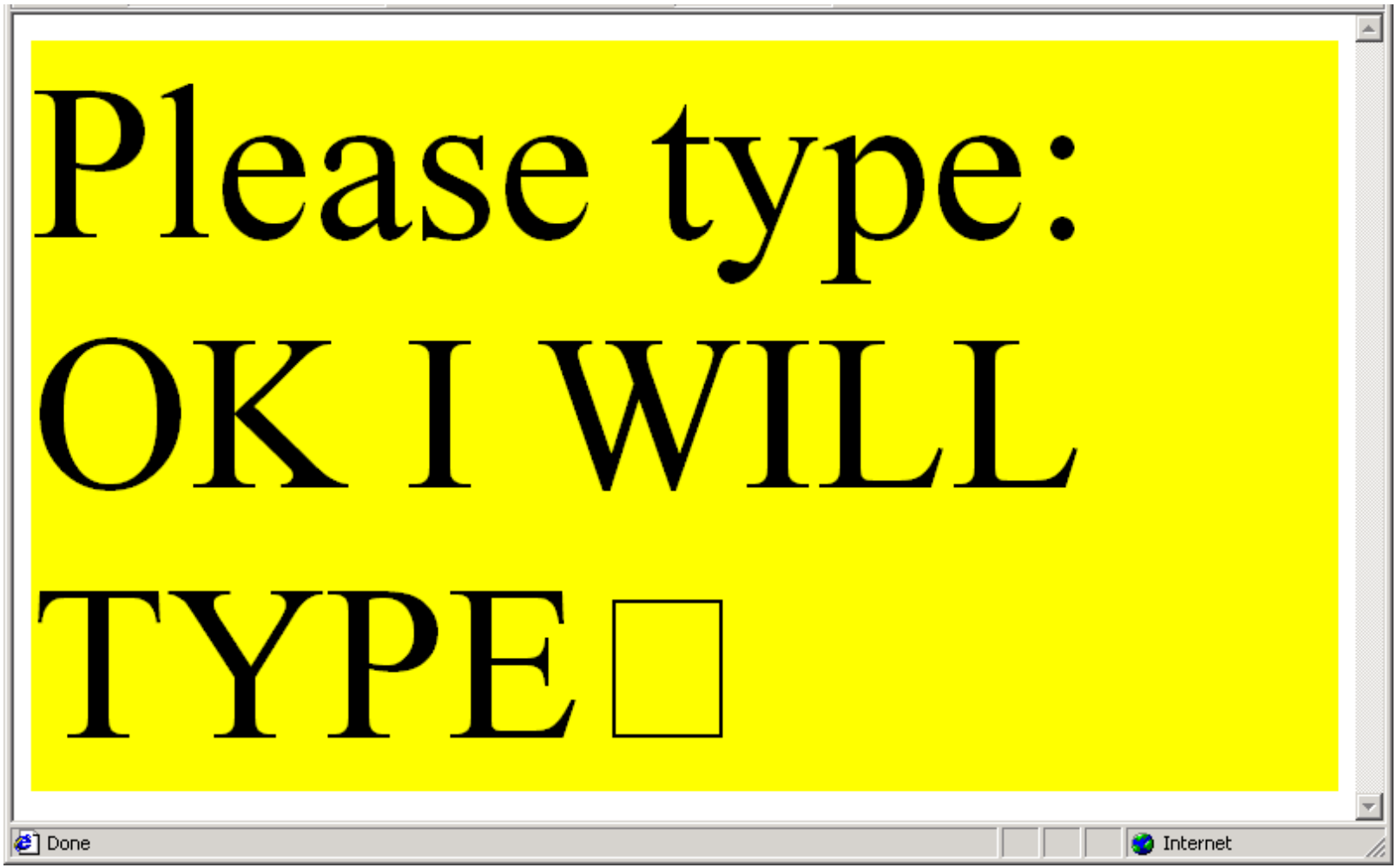
- .innerHTML changes the text of something
- You can change the text inside anything that contains text, e.g., paragraph, div, list, header, etc.

Extending - .innerHTML

- We can also use .innerHTML to find out the text which is already inside an object

```
para=document.getElementById("output_paragraph"); // find
    the element
para.innerHTML=para.innerHTML + " more text!";
                                        // add more text to it
```

- The last line of code takes the text that is already there, appends more text to it, and puts the result back


Please type:
OK I WILL
TYPE
Done
Internet

The Image Array

- document.images[] is an array containing all images in a web page
  - The first image in the web page is document.images[0]
  - The second image is document.images[1], etc.

- Change the .src property of the image to change an image. For example:
  - document.images[0].src=“mypicture.jpg"

Example Code

```
<body>
<h1>Click on an image to change the .src of that
   image ...</h1>

<img src="arnold.jpg" onclick="document.images[0].src =
   'totoro.jpg' ">

<img src="jackie_chan.jpg" onclick="document.images[1].src
   = 'hello_kitty.gif' ">

<img src="jet_li.jpg" onclick="document.images[2].src =
   'mcmug.gif' ">
</body>
```

http://course.cs.ust.hk/comp303/hkust/lectures/f2005_javascript_examples_for_lab3/09_image_arra - Microsoft Internet Explorer

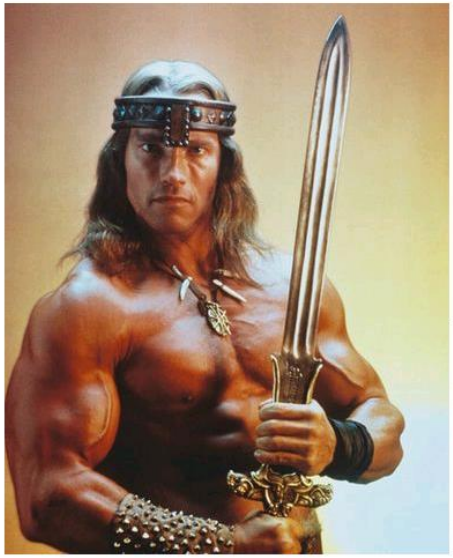
File Edit View Favorites Tools Help

Back Search Favorites Media

Address http://course.cs.ust.hk/comp303/hkust/lectures/f2005_javascript_examples_for_lab3/09_image_array.html

Google Search Popups okay Check AutoLink AutoFill Options

Click on an image to change the .src of that image ...



Done Internet

Click

f2005_javascript_examples_for_lab3/09_image_arra - Microsoft Internet Explorer

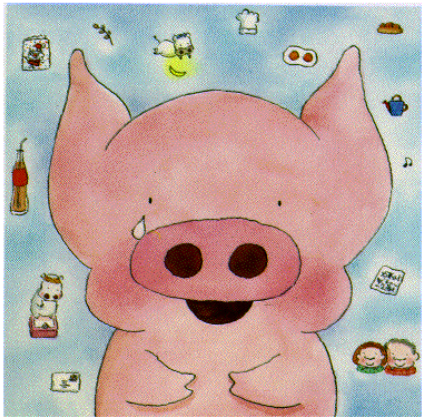
Media

/f2005_javascript_examples_for_lab3/09_image_array.html

Go Links

Popups okay Check AutoLink AutoFill Options

Click on an image to change the .src of that image ...



Done Internet

Multimedia in HTML5: <video>

- HTML5 supports <video> and <audio>
- Browsers supporting HTML5 (i.e., most browsers) have native support of video and audio objects without using external software

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

Try it out

- controls vs autoplay
- First <source> tag with recognized format is played

Multimedia in HTML5: <audio>

- "controls" and multiple <source> tags are similar to <video> :

```
<audio controls>
    <source src="horse.mp3" type="audio/mpeg">
    <source src="horse.ogg" type="audio/ogg">
Your browser does not support the audio element.
</audio> Try it out
```

Older Technique: <embed> in HTML4

- **embed** a video/audio object played by plugins:

```
<embed id="bgmusic" type="application/x-mplayer2"
src="12_fun_music.mid" hidden="true" autostart="true"
loop="true"></embed>
```

- Browser loads the MIDI file and use MediaPlayer to play it immediately (autostart="true")
- Media is played by a plugin, not natively by browser
- Better use direct JavaScript control (next slide)

JavaScript Control over Audio

```
<script language="JavaScript">

    function play_sound() {
          //stop playing the 'hit' sound file if it is playing
          document.getElementById( "hit" ).stop();

          //play the 'hit' sound file
          document.getElementById( "hit" ).play();
    }
</script>

 <body onmousedown="play_sound()">
 <h1>Play sound file using JavaScript</h1>
 If you click on this page, a sound file will be played.

 <embed id="hit" type="application/x-mplayer2"
     src="hit.wav" hidden="true" autostart="false" >
 </embed>
 </body>
 </html>
```

- **embed** tag exposes attributes "stop", "pause" and "play", etc., to JavaScript
- Audio will not play until mouse down

Take Home Message

- We look at more examples of HTML and JavaScript
  - <table> as a convenient way to align objects
  - Events and user interactions
  - Handling of Non-text objects: <embed>, <video>
- Difference between playing a video natively by the browser and by a plugin