08240 2D Computer Graphics: Lab 5

Aim

The aim of this lab is to add some keyboard and mouse control to the scenes that you have been developing in the labs.

Coding the JavaScript

In our previous labs we added an event listener to the page 'load' event that would call our own onLoad function when the page had loaded. All of our code was encapsulated in this onLoad function.

In order to get keyboard control in our canvas we need to make use of the event listener again, but this time we need to hook in to the 'keydown' event. This event fires whenever a key on the keyboard is pressed down. There is also a 'keyup' event that fires when a key is released.

1) In the canvas.js file locate the code that checks whether the window object has an addEventListener function. It should look something like below if you followed the previous lab guides.

```
// check to see if the browser supports
// the addEventListener function
if (window.addEventListener) {
    window.addEventListener('load', onLoad, false);
}
```

2) You can copy and paste this inside you initialise function, or better yet, make a new function inside your onLoad function called 'setupEventHandlers' or something like that and paste it in there.
3) Replace the 'load' event listener in your new function with an event listener for the 'keydown' event which calls a new event handler that we are going to create. Call it something sensible like 'onKeyDown' or 'handleKeyPresses'.

```
// check to see if the browser supports
// the addEventListener function
if (window.addEventListener) {
    window.addEventListener('keydown', onKeyDown, false);
}
```

4) Now when the event is triggered it will call our event handler; we just need to create that function now. Make sure that you call it the same as you specified in the event listener.

```
function onKeyDown(event) {

};
```

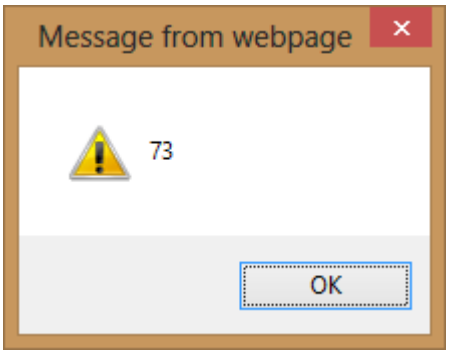
5) Note that the function takes a parameter called event. This parameter will contain properties of the event such as which key was pressed.
6) The property that we are interested in is the keyCode.

```
function onKeyDown(event) {
    var keyCode = event.keyCode;
};
```

7) The keyCode gives us a number that is equal to the ASCII key code for the key that was pressed. If you want to know what the number for a particular key is then you can find a list using Google (other search engines are available) or perhaps more usefully you can temporarily add an alert call to your function.

```
function onKeyDown(event) {
    var keyCode = event.keyCode;
    alert(keyCode);
};
```

8) The alert function will pop up a message box when it is called. In this case the message will be the number given by the keyCode. For example, if you press 'i', the specified key for zooming in, you would get the following pop up:



9) So now you know the ASCII code for 'i' is 73.
10) The best way of dealing with the keyCodes is to use a switch statement, with a different case for each keyCode that you want to handle.

```
function onKeyDown(event) {
    var keyCode = event.keyCode;
    switch(keyCode) {
        case 73: //i
            // do something here
            break; // this stops the next case from executing
    }
};
```

11) It is good practice to indent the cases so you can easily see what actions belong to each case. I find it useful to add a comment to the case to remind me what key the keyCode relates to.
12) Now you just need to replace the 'do something here' comment with code to do something. In this case you may want to alter a scale variable that you can use in your draw function to scale the canvas.

13) For adding mouse events, inside your setupEventHandlers function add event listeners to your canvas for mousedown, mouseup (for button clicks), mousemove (fires when you move the mouse on the canvas), and mousewheel (fires when the mousewheel position changes).

```
// check to see if the browser supports
// the addEventListener function
if (canvas.addEventListener) {
    canvas.addEventListener('mousedown', onMouseDown, false);
    canvas.addEventListener('mouseup', onMouseUp, false);
    canvas.addEventListener('mousemove', onMouseMove, false);
    canvas.addEventListener('mousewheel', onMouseWheel, false);
}
```

14) You will need to add functions to handle each of these events, as with the keypresses.

```
function onMouseDown(event) {

};
```

15) The event parameter has a number of useful attributes. The first is client and client which give the coordinates in canvas space of the mouse pointer. Remember that if you want these to be in world space (i.e. the space the objects in your image exist in) then you will need to transform the coordinates so that they are translated, scaled and rotated like your image.
16) For the mousewheel event handler, the parameter has an attribute wheelDelta which is the amount the wheel has moved. In principle, you probably only care if it is negative or positive, rather than how big it is.

Summary

You should now be able to add key and mouse event listeners to your canvas and handle keydown and keyup, mousedown, mouseup, mousemove, and mousewheel events.