3. Event Driven Programming

Sam Scott, Sheridan College, 2013

Respond to an Event

1. Copy the canvas app template files to a new location and rename the folder.
2. Open `myCanvasApp.js`, find the `button1Click` function and uncomment the `debugOut` statement, then save the file.
3. Load the `index.html` file into a browser (or hit refresh) and press button 1.
4. Try adding your own code to the click function.
5. Repeat steps 2 through 4 for some of the mouse and keyboard event functions.

REMINDER: You can customize the text on your button in the `init` method.

How Does This Work?

- **DOM Events:** The functions you are adding code to have been set up to be called when the corresponding DOM events happen (DOM stands for Document Object Model, and is the internal representation of the web page).
- **Event Hookups:** In the HTML file, you will notice that certain tags have event attributes, all of which start with the word "on". For example, the `<body>` tag has the attribute `onkeydown="keyDownHelper(event)"`. This specifies that when the `onkeydown` event fires, the `keyDownHelper` function will be called.[1]
- **Event Functions:** In the `canvasAppHelper.js` file you will find the "helper" functions that are linked to all the various events. In each case, the helper function pulls some information out of the event object, and then calls a corresponding function in the `myCanvasApp.js` file. For example, `keyDownHelper` gets the key code and corresponding character and passes these to the `keyDown` function. Once again, the students are spared the messy details.

Ideas for Quick First Exercises

- Draw a circle on the screen wherever the user clicks the mouse
- Have a square follow the user around the canvas.
- Annoy the user! Show "click me" on the canvas, but remove it when the user's mouse enters the canvas area. Then put back the "click me" message when they mouse out.
- Draw a simple white on black drawing, then change it to black on white on a button click.
- Make a pac man chomp its way around the screen in response to the arrow keys.

[1] It should be noted that it is considered best practice to specify event handlers in JavaScript, and not in HTML as we are doing here. But I have violated best practices when I thought it made the code clearer for students who might want to understand how the template works.

Set Up a Timer for Animation

1. In the init method, change the value in `setTimer` to 20.
2. Create a global variable called `x` and set it to `0`.
3. In the clockTickEvent method, add the following code:

```
canvas.clearRect(x,50,50,50);
x = x + 1;
canvas.fillRect(x,50,50,50);
```

4. Save the file, and load the index to watch the square move across the screen.

Ideas for Timer Exercises

1. Start with the instructions in the previous section.
2. Make the block bounce back when it gets to the edge of the screen.
3. Move a second object at the same time.
4. Make "faster" and "slower" buttons to change the block's speed.
5. Have the block move towards the mouse, wherever it is on the canvas.

Tips on Game Creation

- Have a set of global game state variables that indicate, for example, the positions and speed of objects, whether the game is on or not, the score, etc.
- Use `clockTickEvent` as the main game engine. If the game is on, update object positions, check collisions, and clear and redraw the screen here. If not, show a splash screen.
- Use other events (keyboard, mouse, buttons, etc) to make changes to the game state variables.
- Look at the Pong example to see this approach in action.

Project Ideas

1. Implement a two player game of Pig or Nim, using the button, mouse, or keyboard events to interact with the user. Display all the game information on the canvas.
2. Create a paint app that draws when the user holds down the button and moves the mouse across the screen. Use the buttons to let them choose colors and pen sizes.
3. Write a simple action game, like pong or wack-a-mole, or dodgeball (user moves an object with the mouse and tries to dodge a set of other objects bouncing around the screen).

Extra: Fully Autonomous Client-Side Apps!

- Use the HTML5 `localStorage` object to keep info about the user for next time they return.
  http://www.w3schools.com/html/html5_webstorage.asp
- Use the HTML5 App Cache to allow a user to revisit your page even if they lose their connection.
  http://www.w3schools.com/html/html5_app_cache.asp