CS1116/CS5018

Web Development 2

Dr Derek Bridge

School of Computer Science & Information Technology
University College Cork

Calling a function

- Defining a function, e.g.:

```
function irritate_me() {
    window.alert('Irritating, huh?');
}
```

- Executing a function by calling it, e.g.:

```
irritate_me();
```

- Arranging for a function to be executed whenever an event occurs, e.g.:

```
window.addEventListener('click', irritate_me, false);
```

Event-driven programming

- Programs that use this idea are called **event-driven programs**
- In client-side JavaScript, **events** include:
  - when the Web page has finished loading
  - when the user clicks on a hyperlink
  - when the user clicks on a button
  - when the user moves the mouse into or out of a certain region of the screen
  - and so on
- The function that runs is called the **event listener** (or **event handler**)

Event objects

- When an event occurs, relevant information about the event is stored in an object
- E.g. in the case of a mouse click, the event has properties that include
  - `clientX` and `clientY` — the viewport coordinates of the mouse click
- The event handler can access these properties, e.g.:

```
function irritate_me(event) {
    window.alert('Your click was at ' + event.clientX + ' ' + event.clientY );
}

window.addEventListener('click', irritate_me, false);
```

Question

- In fact, our JavaScript programs already contained two examples of event-driven programming
  Q: What are they?

An incomplete program

```
let canvas;
let context;
let width;
let height;

let interval_id;

let ps = [];

let player = {
    x : 0,
    y : 150,
    size : 10
};

let moveRight = false;
let moveUp = false;
let moveDown = false;

document.addEventListener('DOMContentLoaded', init, false);

function init() {
    canvas = document.querySelector('canvas');
    context = canvas.getContext('2d');
    width = canvas.width;
    height = canvas.height;


    interval_id = window.setInterval(draw, 33);
}

function draw() {
    if (ps.length < 10) {
        let p = {
            x : width,
            y : getRandomNumber(0, height),
            size : 10,
            xChange : getRandomNumber(-10, -1),
            yChange : 0
        };
        ps.push(p);
    }
    context.clearRect(0, 0, width, height);
    context.fillStyle = 'yellow';
    for (let p of ps) {
        context.fillRect(p.x, p.y, p.size, p.size);
    }
    context.fillStyle ='cyan';
    context.fillRect(player.x, player.y, player.size, player.size);
    if (player.x + player.size >= width) {
        stop();
        window.alert('YOU WIN!');
        return;
    }
    for (let p of ps) {
        if (collides(p)) {
            stop();
            window.alert('YOU LOSE!');
            return;
        }
    }
```

```
    for (let p of ps) {
        p.x = p.x + p.xChange;
        p.y = p.y + p.yChange;
        if (p.x <= -p.size) {
            p.x = width;
        }
    }
    if (moveRight) {
        player.x += 3;
    }
    if (moveUp) {
        player.y -= 3;
    }
    if (moveDown) {
        player.y += 3;
    }
}

function getRandomNumber(min, max) {
    return Math.floor(Math.random() * (max - min + 1)) + min;
}

function collides(p) {
    if (player.x + player.size < p.x ||
        p.x + p.size < player.x ||
        player.y > p.y + p.size ||
        p.y > player.y + player.size) {
        return false;
    } else {
        return true;
    }
}

function stop() {
    clearInterval(interval_id);


}
```