Timer-Based Animation

Jerry Cain
CS 106AJ
October 16, 2017
slides courtesy of Eric Roberts

Timer Events
- The programs from the previous slide deck respond to mouse events by adding an event listener to the GWindow object.
- JavaScript also allows you to listen for timer events, which occur after a specified time interval.
- As with mouse events, you specify the listener for a timer event in the form of a callback function that is automatically invoked at the end of the time interval.
- You can add animation to a JavaScript program by setting a timer for a short interval and having the callback function make small updates to the graphical objects in the window.
- If the time interval is short enough (typically between 20 and 30 milliseconds), the animation will appear smooth to the human eye.

Timeouts
- JavaScript supports two kinds of timers. A **one-shot timer** invokes its callback function once after a specified delay. You create a one-shot timer by calling
  ```javascript
  setTimeout(function, delay);
  ```
  where `function` is the callback function and `delay` is the time interval in milliseconds.
- An **interval timer** invokes its callback function repeatedly at regular intervals. You create an interval timer by calling
  ```javascript
  setInterval(function, delay);
  ```
The `setInterval` function returns a numeric value that you can later use to stop the timer by calling `clearInterval` with that numeric value as an argument.

A Simple Example of Animation

```javascript
function AnimatedSquare() {
  function step() {
    Square.move(dx, dy);
    stepCount++;
    if (stepCount === N STEPS) clearInterval(timer);
  }

  g = new GWindow(4, 2, 1729);
  g.add(new AnimatedSquare(), 0);
}
```

```javascript
function AnimatedSquare() {
  function step() {
    Square.move(dx, dy);
    stepCount++;
    if (stepCount === N STEPS) clearInterval(timer);
  }

  g = new GWindow(4, 2, 1729);
  g.add(new AnimatedSquare(), 0);
}
```