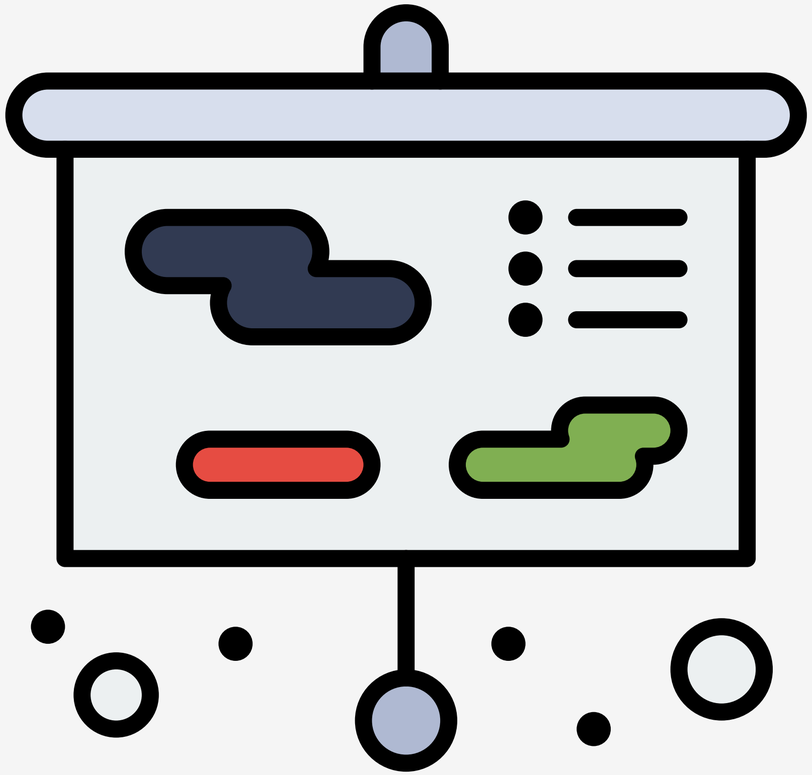
# INTRODUCTION TO JAVASCRIPT

Lecture 16

# TODAY'S TOPICS



- Frameworks and Libraries
- Introduction to jQuery
- Review: Color Picker

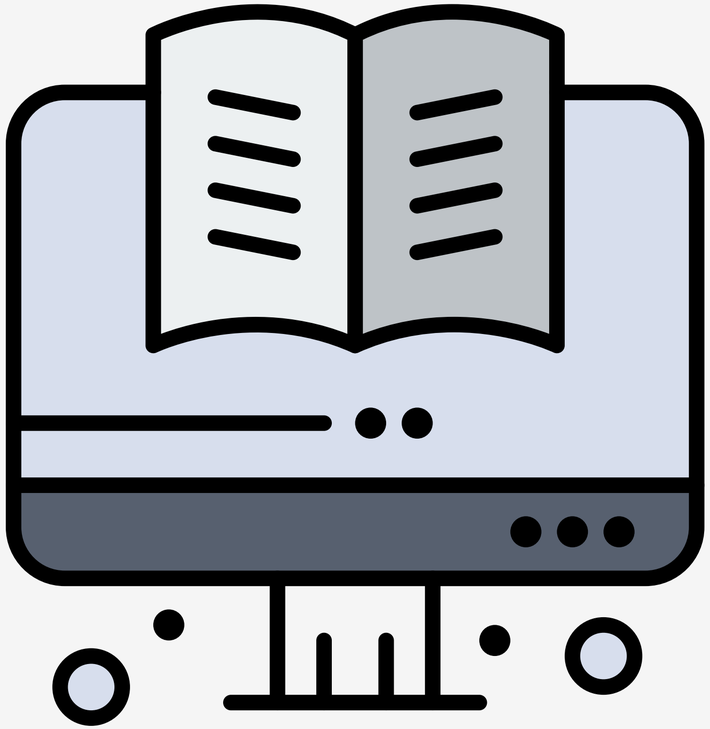
# ANNOUNCEMENTS

- Sign-in Sheet

# QUESTIONS?

# FRAMEWORKS & LIBRARIES

# FRAMEWORKS & LIBRARIES



- Don't Repeat Yourself (DRY)
- Don't reinvent the wheel.
- Frameworks and Libraries are collection of functions and methods
- Often adhere to a coding pattern or methodology
- Libraries can be added to an existing project and used as desired
- Framework are used to build new projects

# FRAMEWORKS & LIBRARIES

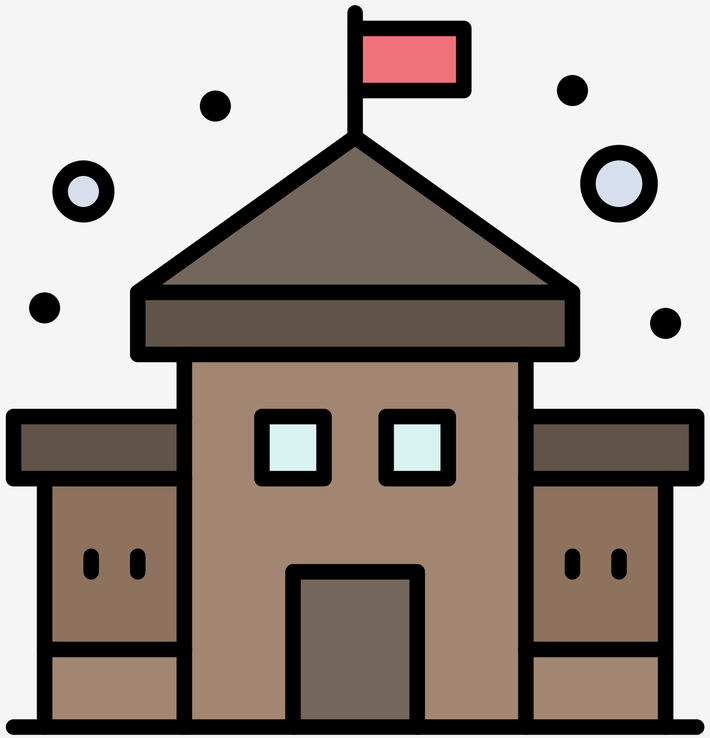
- Angular (2016)
- Angular.js (2010)
- Backbone.js (2010)
- Chaplin.js (2012)
- Dojo (2005)
- Ember.js (2011)
- Knockout (2010)
- jQuery (2006)
- Meteor (2012)
- MooTools (2007)
- React.js (2013)
- Polymer (2015)
- Prototype (2005)
- Underscore.js (2009)
- Vue.js (2014)

# FRAMEWORKS & LIBRARIES

- Angular (2016)
- Angular.js (2010)
- Backbone.js (2010)
- Chaplin.js (2012)
- Dojo (2005)
- Ember.js (2011)
- Knockout (2010)
- jQuery (2006)
- Meteor (2012)
- MooTools (2007)
- React.js (2013)
- Polymer (2015)
- Prototype (2005)
- Underscore.js (2009)
- Vue.js (2014)

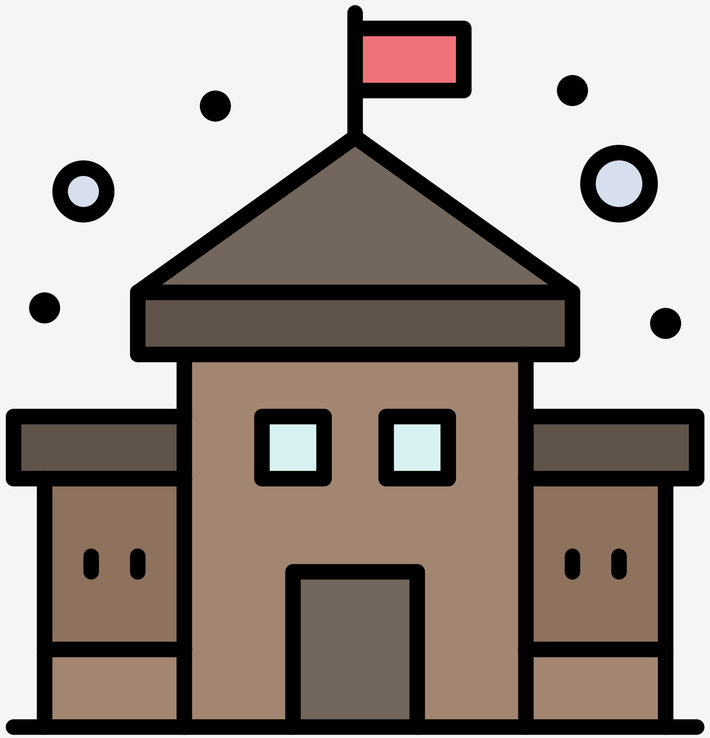
# INTRODUCTION TO JQUERY

# INTRODUCTION TO JQUERY



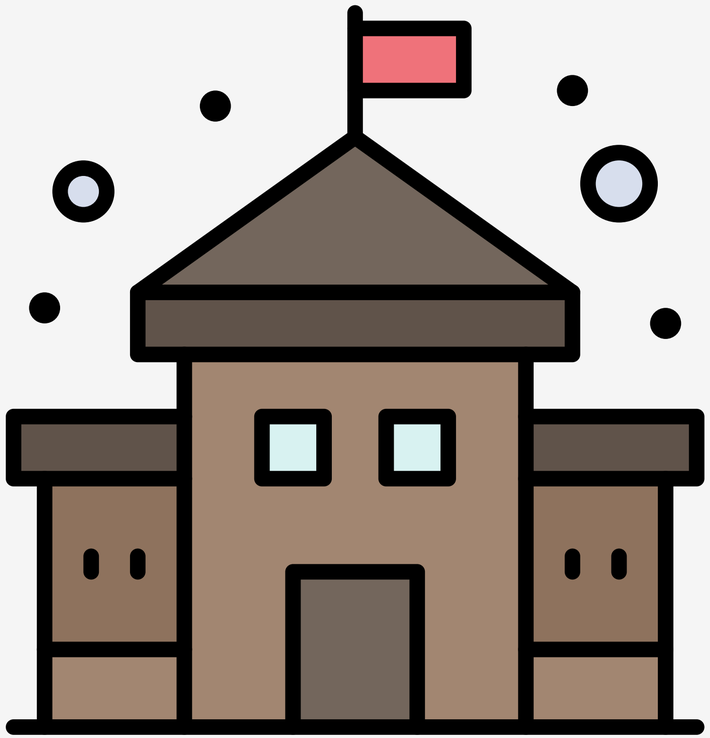
- It is *JAVASCRIPT*
- It is a library, not a framework
- A collection a functions and objects that simplifies writing JavaScript
- Included cross-browser support for DOM manipulation
- Simplified the process of using XMLHTTPRequest

# JQUERY - PROS



- Easy to learn
- Easy to integrate
- Great documentation
- Lots of resources
- Lots, LOTS of legacy code

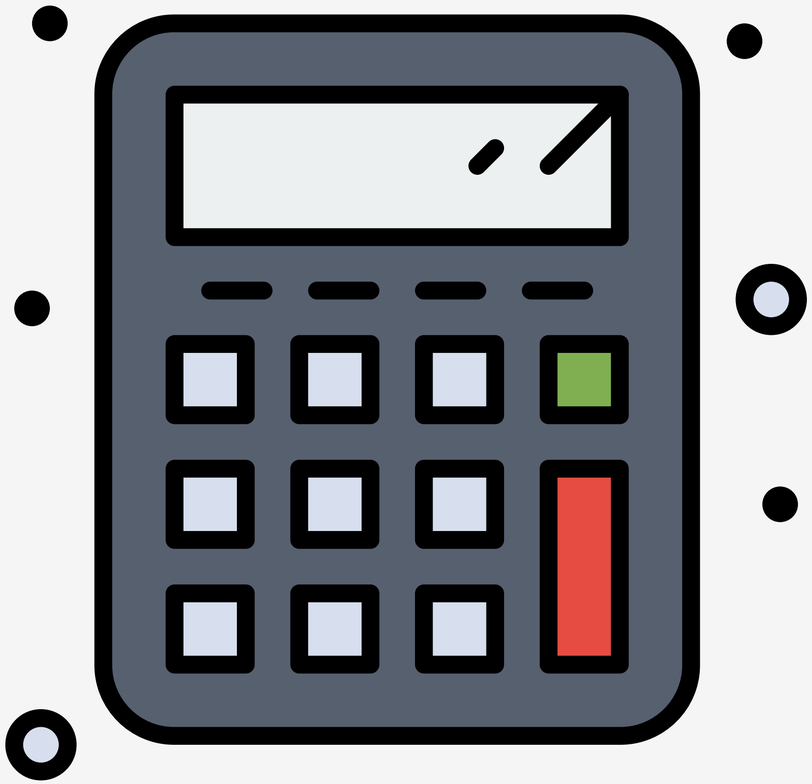
# JQUERY - CONS



- No native browser support
- JavaScript is Better
- Browser Support is Better
- Less desired than the modern Frameworks

# ADDING JQUERY

# JQUERY - CONS

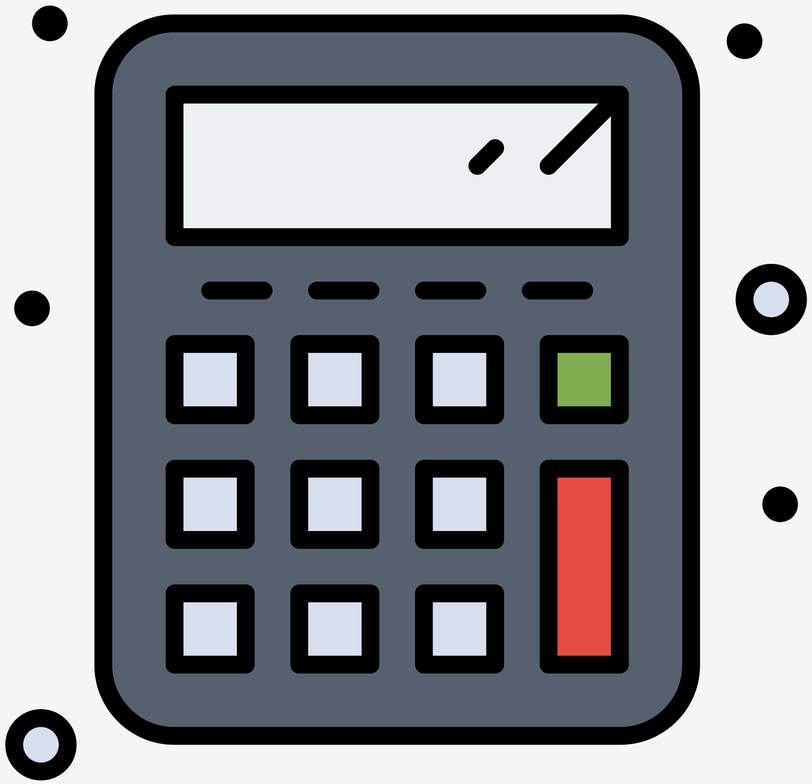


- jQuery must be added to your HTML
- Two Option:
  - Locally
  - Content Delivery Network (CDN)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>jQuery</title>
  <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
</head>
<body>
  ...
</body>
</html>
```

# DOM MANIPULATION

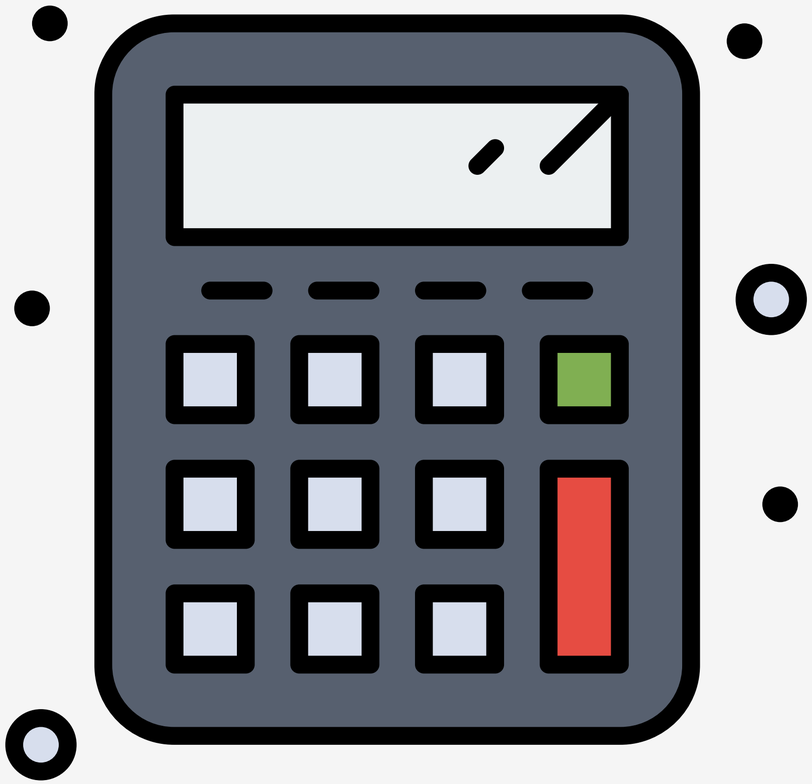
# DOM MANIPULATION



- The `jQuery()` method makes selecting and retrieving elements easier
- It uses CSS Selectors like `querySelector`
- The `$` can be used as a shorthand

```
// Using Vanilla JavaScript
const $button = document.getElementById('button')

// Using jQuery
const $button = $('#button')

// Using vanilla JavaScript
const $button = document.querySelector('.button')

// Using jQuery
const $button = $('.button')

// Using vanilla JavaScript
const $button = document.querySelectorAll('.button')

// Using jQuery
const $button = $('.button')
```

# DOM MANIPULATION

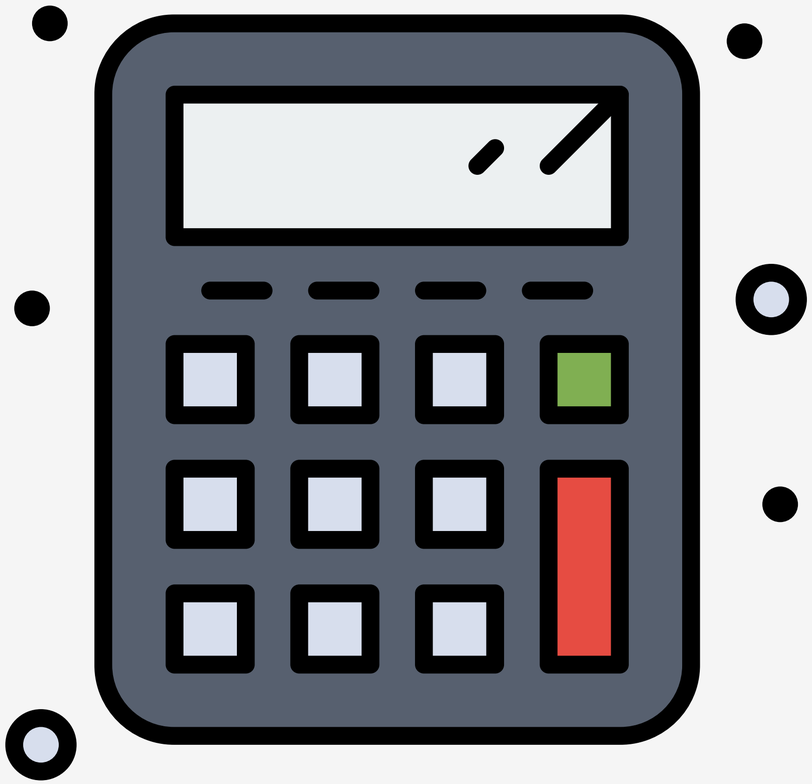


- After a jQuery element has been retrieve attributes and styles can be manipulated using methods
- The `attr()` method is used to get and set an attribute
- The `css()` method is used to update the styles of an element

```
// Using Vanilla JavaScript
const $link = document.getElementById('link')

$link.href = 'https://google.ca'
$link.style.color = 'red'

// Using jQuery
const $link = $('#link')

$link.attr('href', 'http://google.ca')
$link.css('color', 'red')
```

```
// Using Vanilla JavaScript
const $links = document.querySelectorAll('.link')

for (const $link of $links) {
  $link.style.color = 'red'
}



// Using jQuery
const $links = $('.link')

$links.css('color', 'red')
```

# DOM MANIPULATION



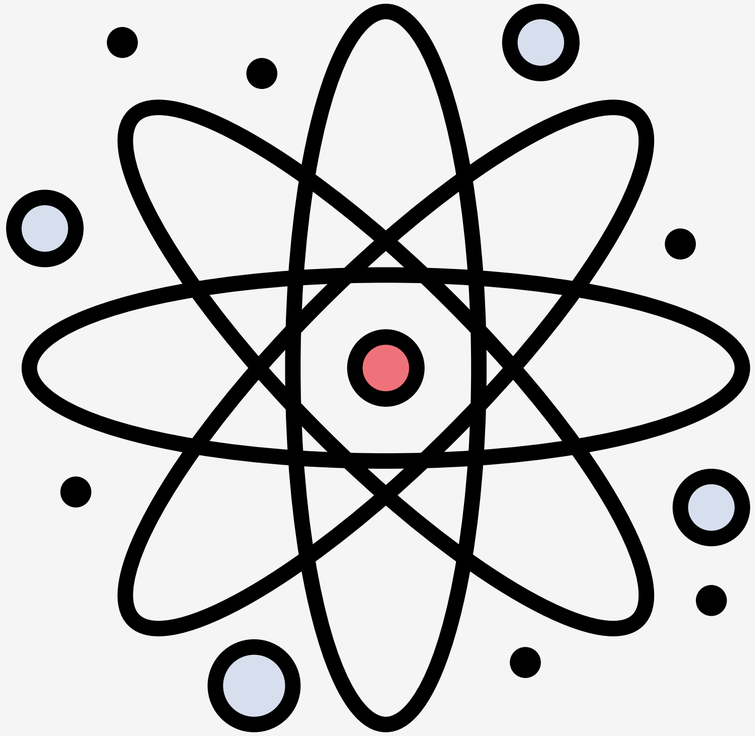
- jQuery has specific methods for working with classes
- The `addClass()` method is used to add a class
- The `removeClass()` method is used to remove a class
- The `toggleClass()` method is to toggle a class
- The `hasClass()` method determine if the element has a class

```
// Using Vanilla JavaScript
const $link = document.getElementById('link')

$link.classList.add('active')
$link.classList.remove('active')

// Using jQuery
const $link = $('#link')

$link.addClass('active')
$link.removeClass('active')
```

```
// Using Vanilla JavaScript
const $links = document.querySelectorAll('.link')

for (const $link of $links) {
  $link.classList.add('active')
}



// Using jQuery
const $links = $('.link')

$links.addClass('active')
```

# CREATING
# DOM ELEMENTS

# DOM MANIPULATION



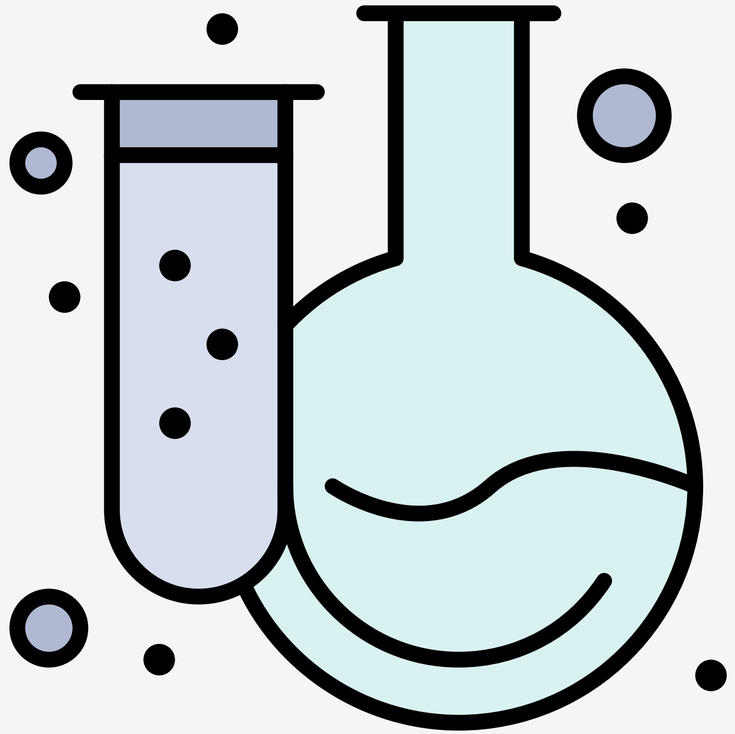
- jQuery allows for the creation of DOM elements using methods on the target element
- The `before()` method creates an element before the target
- The `after()` method creates an element after the target
- The `append()` method creates an element as the last child of the target
- The `prepend()` method creates an element as the first child of the target

```
const title = 'The Title'
const author = 'Me'

const $story = $('.story')

// before the story
$story.before(`<h2>${title}</h2>`)

// first child of story
$story.prepend('<p>Once upon a time...</p>')

// last child of story
$story.append('<p>The End.</p>')

// after the story
$story.after(`<p>by ${author}</p>`)
```

```
const colors = ['red', 'green', 'blue']

const list = []

for (const color of colors) {
  list.push(`<li>${color}</li>`)
}

$list = $('#list')
$list.append(list.join(''))
```

# DOM EVENTS

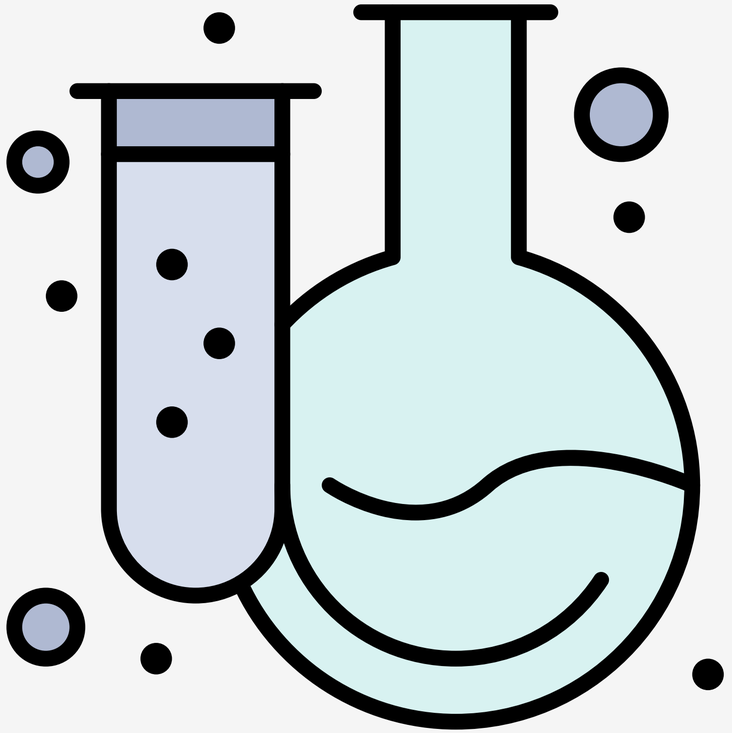
# DOM EVENTS



- jQuery has a custom method for responding to DOM Events
- The `on()` method is used to attach an event handler to the selected element
- jQuery includes many shortcuts of the `on()` method including:
  - `click()`
  - `change()`
  - `keydown()`

```
$button = $('.button')

$button.on('mouseover', function () {
  $button.text("Don't you do it!")
})

$button.on('click', function () {
  $button.text('You clicked the button!')
})

$button.on('mouseout', function () {
  $button.text("Don't do it again!")
})
```

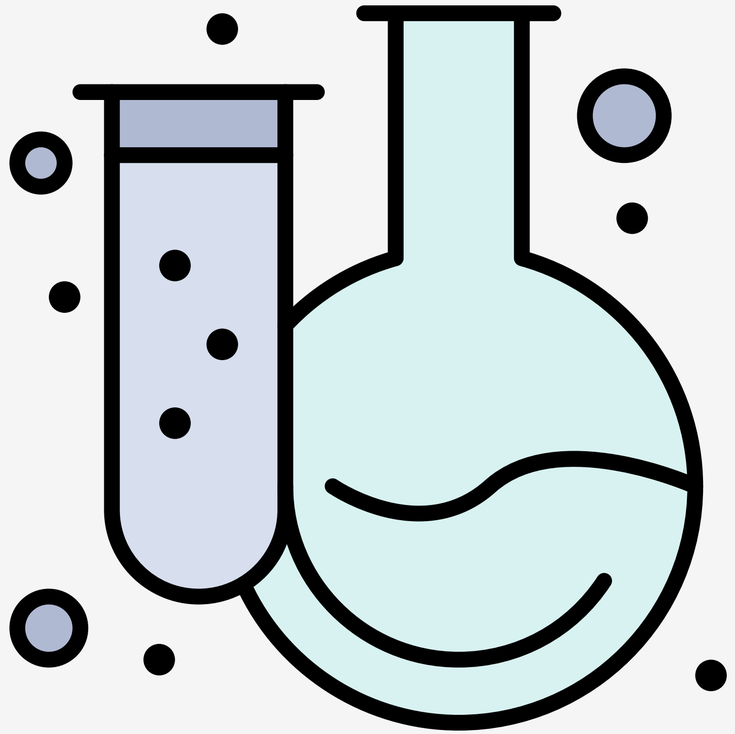
# DOM EVENTS



- The `on()` method can be applied to multiple elements
- When working with multiple element the `this` reference with jQuery wrapper must be used.

```
$buttons = $('.button')

$buttons.on('click', function () {
  $(this).text('You clicked the button!')
})
```

# DOM EVENTS



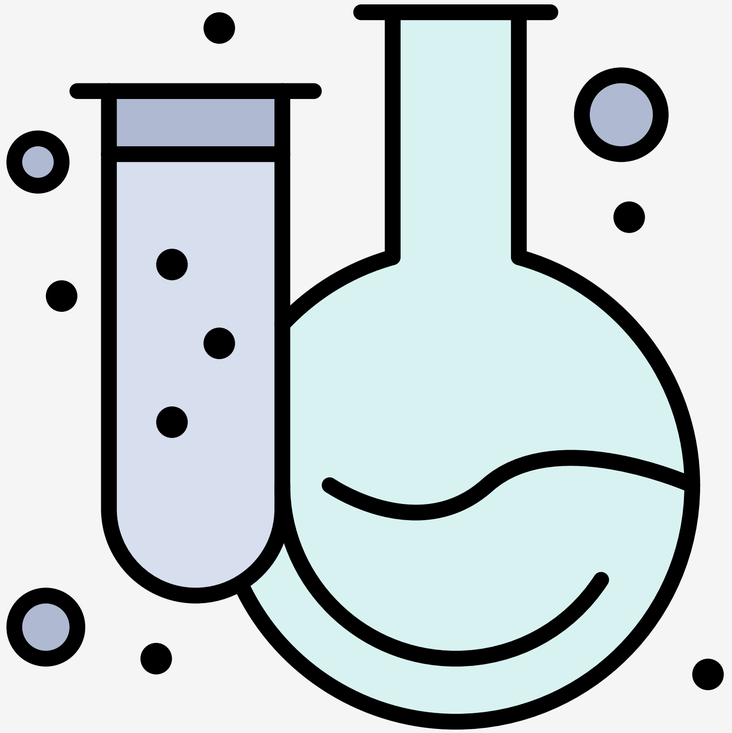
- jQuery also builds event delegation into the `on()` method
- A second selector can be provided to filter the descendants
- Allows an event to attached even if the element don't yet exist

```coffeescript
$list = $('#list')

$list.on('click', 'li', function () {
  $(this).css('text-decoration', 'line-through')
})
```

# HANDS-ON

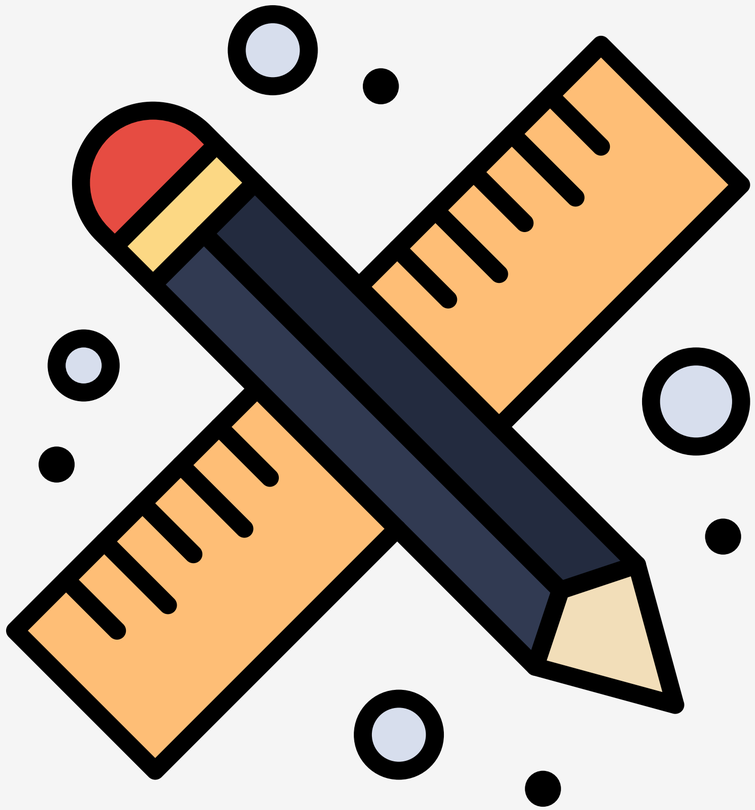
# PRACTICE

# COLOR PICKER - NOT GRADED



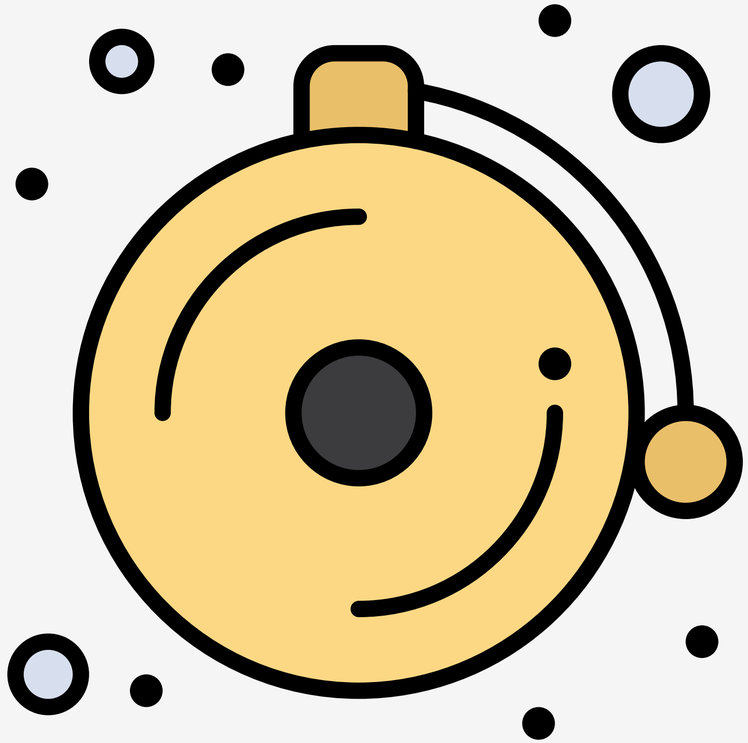
- Complete the Color Picker assignment using jQuery
- Don't forget to add jQuery

# DOUBLE JEOPARDY - BONUS



- *GITHUB CLASSROOM ASSIGNMENT*
- Create an interactive game of Jeopardy using jQuery
- Dynamically populate the game board with titles, values, clues, and answers
- Add Event Listeners to populate and show the overlay and to show the answer
- *TEST YOUR CODE*
- Submit the URL to your repository
- *DUE:* **Wed. Dec. 11 @ 11:59 PM**

# NEXT TIME...



- Ajax with jQuery
- jQuery Plugins