jQuery Events

Making things Interactive

Objectives

- click()
- keypress()
- on()

Click()

jQuery's *click()* method is a quick and easy way to add a click listener to element(s)

```
//prints when item with id 'submit' is clicked
$('#submit').click(function(){
  console.log("Another click");
});

//alerts when ANY button is clicked
$('button').click(function(){
  alert("Someone clicked a button");
});
```

keypress()

jQuery's *keypress()* method is a quick and easy way to add a keypress listener to element(s)

```
//listen for any keypress in any text input
$('input[type="text"').keypress(function(){
  alert("text input keypress!");
});
```

on()

jQuery's *on()* works similarly to *addEventListener*.
It lets you specify the type of event to listen for.

```
//prints when item with id 'submit' is clicked
$('#submit').on('click', function(){
  console.log("Another click");
});

//alerts when ANY button is clicked
$('button').on('click', function(){
  console.log("button clicked!");
});
```

on()

It's not just for click events. *on()* supports all types of events

```
//double click event
$('button').on('dblclick', function(){
  alert("DOUBLE CLICKED!");
});

//drag start event
$('a').on('dragstart', function(){
  console.log("DRAG STARTED!");
});

//keypress event
$('input[type="text"').on('keypress', function(){
  alert("key press in an input!")
});
```

Why Use On()?

In most cases, *click()* and *on('click')* will both get the job done. HOWEVER, there is one key difference:

- *click()* only adds listeners for existing elements
- *on()* will add listeners for all potential future elements
- This will all make sense in the next video!