The event object

Reacting to events often requires knowledge about the event itself, so this is a quick breakdown of the event object which gets passed to an event listener's callback.

Remember that the target element calls the callback function when the event occurs. When this function is called, jQuery passes an event object to it containing information about the event. This object holds a ton of useful information that can be used in the body of the function. This object, which is usually referenced in JavaScript as `e`, `evt`, or `event`, has several properties that you can use to determine the flow of your code. Try logging the object to see what's available:

```
$( 'article' ).on( 'click', function( evt ) {
    console.log( evt );
});
```

You should notice a `target` property. The `target` property holds the page element that is the target of the event. This can be extremely useful if an event listener has been set up for a number of elements:

```
$( 'article' ).on( 'click', function( evt ) {
    $( evt.target ).css( 'background', 'red' );
});
```

In the example above, an event listener is set up for every `article` element on the page. When an article is clicked an object containing information about the event is passed to the callback. The `evt.target` property can be used to access just the clicked on element! jQuery is used to select just that one element from the DOM and update its background to red.

The event object also comes in handy when you want to prevent the default action that the browser would perform. For example, setting up a `click` event listener on an anchor link:

```
$( '#myAnchor' ).on( 'click', function( evt ) {
    console.log( 'You clicked a link!' );
});
```

Clicking on the `#myAnchor` link will log the message to the console, but it will also navigate to that element's `href` attribute - potentially redirecting to a new page. The event object can be used to prevent the default action:

```
$( '#myAnchor' ).on( 'click', function( evt ) {
    evt.preventDefault();
    console.log( 'You clicked a link!' );
});
```

In the code above, the `evt.preventDefault();` line instructs the browser not to perform the default action.

Other uses include:

- `event.keyCode` to learn what key was pressed - invaluable if you need to listen for a specific key
- `event.pageX` and `event.pageY` to know where on the page the click occurred - helpful for analytics tracking
- `event.type` to find what event happened - useful if listening to a target for multiple events

The event object can be an incredibly useful tool! Learn more by checking out:

- **jQuery's Event Object**
- **event.target property**
- **DOM Level 3 Events**