CPSC1520 – JavaScript 1 Exercise: Making Decisions

Decisions (Introduction)

Often, an interaction with the user may yield more than one result. In these cases, a decision must be made to determine which action should be executed based on the current state of the page. This can be done by associating specific conditions to desired actions.

Conditions are essentially Boolean values (**true** or **false**) that form the basis for whether a code block is executed or not. They can be built as expressions (for example using relational or equality operators) or could be based on the returned value from a function.

Relational Operators

Operator	Description
>	Greater than – yields true if the left operand is greater than the right operand
>=	Greater than or equal to – yields true if the left operand is greater than or equal to the right operand
<	Less than – yields true if the left operand is less than the right operand
<=	Less than or equal to – yields true if the left operand is less than or equal to the right operand

Table 1. Relational operators

Equality Operators

Operator	Description
==	Equals – yields true if the left operand is similar to the right operand
===	Identity – yields true if the left operand is identical to the right operand
!=	Not equals – yields true if the left operand is not similar to the right operand
!==	Non-identity – yields true if the left operand is not identical to the right operand

Table 2. Equality operators

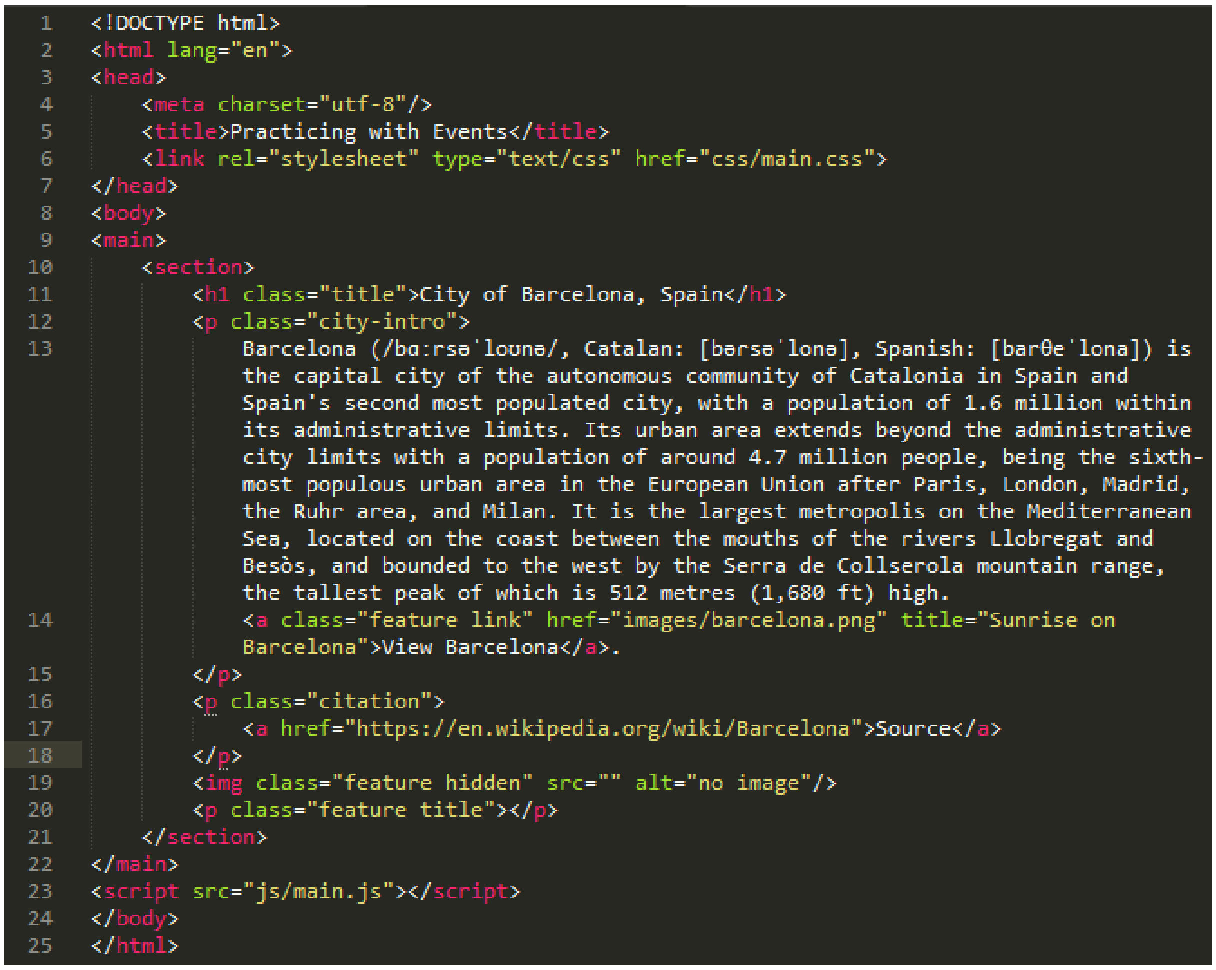
If Statement

To demonstrate how simple decisions can be made, we will revisit the previous exercise (image-description). Previously, your task was to add an event listener to a link on the page that when clicked would display a photo. In this exercise, you will add the functionality to hide the photo by clicking the same link[1].

Condition

The first step will be to identify what exactly the conditions are for displaying the photo or hiding it. You will need to inspect the HTML to know what states are possible.

[1] While this effect can be achieved through the use of the el.classList.toggle() function, an approach using if-else is used here for instructional purposes.



```html
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="utf-8"/>
5       <title>Practicing with Events</title>
6       <link rel="stylesheet" type="text/css" href="css/main.css">
7   </head>
8   <body>
9   <main>
10      <section>
11          <h1 class="title">City of Barcelona, Spain</h1>
12          <p class="city-intro">
13              Barcelona (/bɑːrsəˈloʊnə/, Catalan: [bərsəˈlonə], Spanish: [barθeˈlona]) is
                the capital city of the autonomous community of Catalonia in Spain and
                Spain's second most populated city, with a population of 1.6 million within
                its administrative limits. Its urban area extends beyond the administrative
                city limits with a population of around 4.7 million people, being the sixth-
                most populous urban area in the European Union after Paris, London, Madrid,
                the Ruhr area, and Milan. It is the largest metropolis on the Mediterranean
                Sea, located on the coast between the mouths of the rivers Llobregat and
                Besòs, and bounded to the west by the Serra de Collserola mountain range,
                the tallest peak of which is 512 metres (1,680 ft) high.
14              <a class="feature link" href="images/barcelona.png" title="Sunrise on
                Barcelona">View Barcelona</a>.
15          </p>
16          <p class="citation">
17              <a href="https://en.wikipedia.org/wiki/Barcelona">Source</a>
18          </p>
19          <img class="feature hidden" src="" alt="no image"/>
20          <p class="feature title"></p>
21      </section>
22  </main>
23  <script src="js/main.js"></script>
24  </body>
25  </html>
```

Figure 1. Exercise source HTML

Currently the img.feature element (line 19 above) also has class hidden, which means the image is currently not visible to the user. The condition for whether the image needs to be revealed or not can then be determined by the presence of the hidden class:

```
featureImage.classList.contains('hidden') === true
```

*Example 1. Sample call to contains, which returns either **true** or **false***

Now that we have a condition to check, we can use an if-statement to implement the check for us:

```
// Only set src and remove hidden class if the image is not visible
if (featureImage.classList.contains('hidden') === true) {
    featureImage.src = featureLink.href;
    featureImage.classList.remove('hidden');
}
```

Example 2. Implementation of if statement to reveal the hidden image

You can now update the featureLinkHandler function with this if-statement (additions are highlighted):

```
function featureLinkHandler(evt) {

    let featureImage = document.querySelector('img.feature');
    if (featureImage.classList.contains('hidden') === true) {
        featureImage.src = featureLink.href;
        featureImage.classList.remove('hidden');
    }

    evt.preventDefault();
}
```

Example 3. Update for the featureLinkHandler function to make use of if-statement

Now when the a.feature.link is clicked, the img.feature element will only be updated if it was not hidden at the time the link was clicked.

If-Else Statement

The updates we've made only handle the case to show the image if it was not currently hidden. The next task is to do the opposite: hide the image if it is currently visible. This can be done quite easily as we have already identified the condition to determine if it is visible or not.

Else

The if-statement can be paired with an accompanying else-statement. While the if-statement defines a code block to execute when the condition is true, the else-statement defines a code block that should be executed when the condition is false. Since it is associated with an existing condition, all that is needed is the presence of the **else** and its code block. In this example, if the image is *not* hidden then we should update the src of the img.feature to an empty string and add the hidden class back again.

[An important note, an else-statement must be associated with an if-statement and should appear immediately after one.]

```
function featureLinkHandler(evt) {

    let featureImage = document.querySelector('img.feature');
    if (featureImage.classList.contains('hidden') === true) {
        featureImage.src = featureLink.href;
        featureImage.classList.remove('hidden');
    } else {
        featureImage.src = '';
        featureImage.classList.add('hidden');
    }

    evt.preventDefault();
}
```

Example 5. Complete update for the featureLinkHandler function featuring if-else statement

What we have now is a working toggle, the link will either display the image or hide it depending on its current state.

Final Note on Conditions

In the example above we have compared the returned value from classList.contains() to the Boolean value true. While this is valid, it’s not necessary as the condition is based simply on whether or not it is true or false. Since the contains() function returns either of these two values (true or false), it is not necessary to make a comparison for it to work as a condition. We can update our code as follows:

```
function featureLinkHandler(evt) {

    let featureImage = document.querySelector('img.feature');
    if (featureImage.classList.contains('hidden')) {
        featureImage.src = featureLink.href;
        featureImage.classList.remove('hidden');
    } else {
        featureImage.src = '';
        featureImage.classList.add('hidden');
    }

    evt.preventDefault();
}
```

Example 6. Update the condition to use the functions returned value

Exercise

1. Update the example so that the text for the link changes to ‘Hide Barcelona’ when the image is visible and then back to ‘View Barcelona’ when the image is not visible.
2. With this change made, update the condition for whether to display the image or not to be based on the current text being displayed for the link (e.g. if the link currently shows ‘View Barcelona’, then clicking the link should display the image).