

Section 7

Creating a Modal Window Using Web Technology

Setting up the development environment

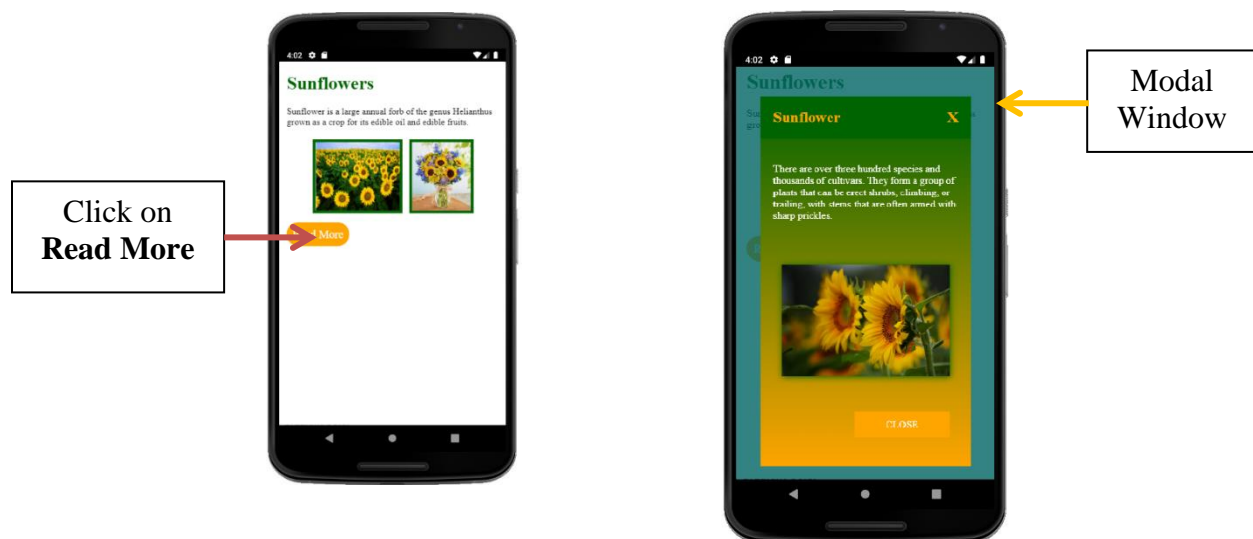
To start our class practice, first set the development environment in Android Studio. Create an **Empty Activity** with an activity **Name** as **Activity 7 – Type your last name**

Once Android Studio is set with an **assets folder**, we can open VS code text editor, create an **index.html** file and save it on **assets folder**.

Great! Now that we have our development environment set up, we can start working on our app layout using HTML.

Modal Windows

Modal windows are boxes that opens in front of the app. They are widely used for advertisements, promotion code, item description, login/register forms, etc.



For this activity, we are going to create a modal window using CSS's transition, opacity, pointer-event, and background gradient properties.

Prepare a cordova project folder (chapter 1). Open a text editor and save a HTML file as *index.html* in the *www* folder within the cordova project folder.

Now, in the *index.html* file, we are going to create the modal window structure in HTML. We create **Read More** link:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="index.css" type="text/css" rel="stylesheet"/>
    <title>Modal by Huixin Wu</title>
  </head>
  <body>
    <h3>Sunflowers</h3>
    <p>Sunflower is a large annual forb of the genus Helianthus grown as a crop for its edible oil and edible fruits. </p>
    <figure>
      
      
    </figure>
    <a href="#openModal" class="linkModel">Read More</a>
  </body>
</html>
```

HTML

CSS

```
body{ padding: 12px; }
h1{ color: darkgreen; }
.linkModel{
text-align:center;
background-color: orange;
font-size: 20px;
padding: 10px;
border-radius: 20px;
text-decoration:none;
color: white;
}
figure img:nth-child(1){
width: 50%;
border: solid darkgreen 5px;
margin: 2%;
height: 120px;
float: left;
}
figure img:nth-child(2){
width: 35%;
border: solid darkgreen 5px;
margin: 2%;
height: 120px;
}
```

Now it is time to create the modal window. From the HTML code, the code line:

```
<p class="linkModel"><a href="#openModal">Read More</a></p>
```

HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="index.css" type="text/css" rel="stylesheet"/>
    <title>Modal by Huixin Wu</title>
  </head>
  <body>
    <h1>Sunflowers</h1>
    <p>Sunflower is a large annual forb of the genus Helianthus grown as a crop for its
    edible oil and edible fruits. </p>
    <figure>
      
      
    </figure>
    <a href="#openModal" class="linkModel">Read More</a>

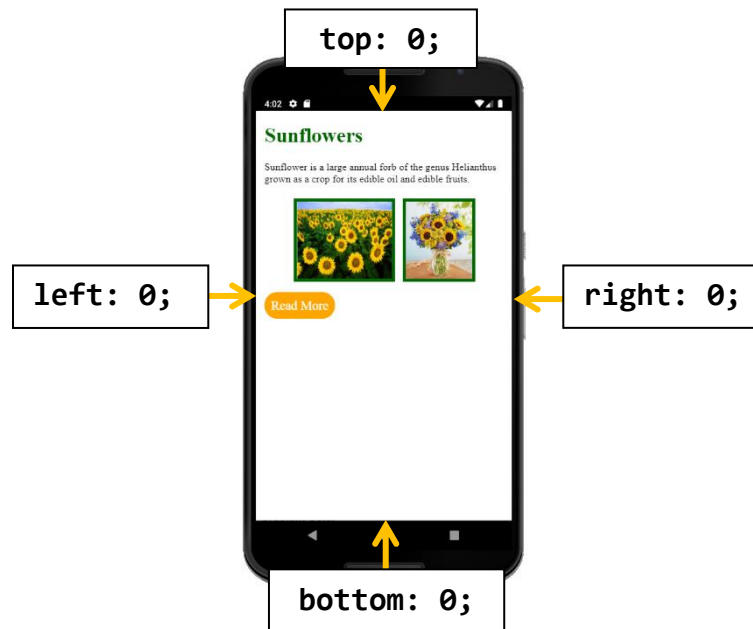
    <div id="openModal" class="modalWindow">

    </div>

  </body>
</html>
```

we can see that the modal is linked to an element **id** as **"#openModal"**. Therefore, we have to create the modal element using **<div>** tag and we will **id** it as: **id="openModal"**. For modal window's **<div>**, we will also add a class name as: **class="modalWindow"** for css attributes.

In the CSS file, since the modal will cover the entire app view, we can set the top, right, left, and bottom to 0:



The other property is to add 80% of transparency to the background color using **rgba**:

`background: rgba(0,100,100,0.8);`

```
.modalWindow {  
  position: fixed;  
  top: 0;  
  right: 0;  
  bottom: 0;  
  left: 0;  
  background-color: rgba(0,100,100,0.8);  
}
```

CSS

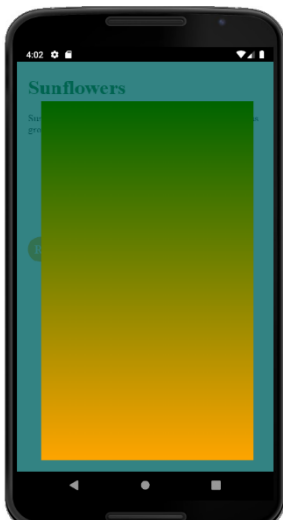


Having the background set, we can add another division with class name **msgText**.

HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modal Window</title>
    <link rel="stylesheet" href="modal.css" type="text/css">
  </head>
  <body>
    <h1>Sunflowers</h1>
    <p>Sunflower is a large annual forb of the genus Helianthus grown as a
      crop for its edible oil and edible fruits. </p>
    <figure>
      
      
    </figure>
    <a href="#openModal" class="linkModel">Read More</a>
    <!-- Modal Window container -->
    <div id="openModal" class="modalWindow">
      <div class="msgText">
      </div>
    </div>
  </body>
</html>
```

In the CSS file, since **msgText** will display in-front of the background, for this, this division will have position relative. Also, for **msgText** we can add linear-gradient background, center, width of 80%, height of 300px, border- radius, and 40% of top margin.



CSS

```
.msgText{
width: 80%;
background: linear-gradient(darkgreen,orange);
margin: auto;
margin-top: 20%;
height: 80%;
}
```

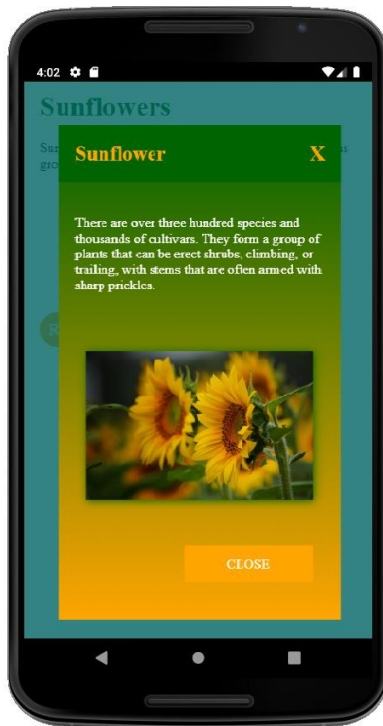
Also, **msgText** is used as a message window. Therefore, we can add a title to this message window using **<h2>**, a close symbol and link using **<a>** element, an image ****, and a text container using **<p>**. The next element to add to the message window **msgText** is a link to close it. For this, we can use an **<a>** tag and link it to the top of the web app using: **href="#"**. Also, we can class it as **class="modal-close"** for CSS attributes.

HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Modal Window</title>
    <link rel="stylesheet" href="modal.css" type="text/css">
  </head>
  <body>
    <h3 id="home">Sunflowers</h3>
    <p>Sunflower is a large annual forb of the genus Helianthus grown as a
      crop for its edible oil and edible fruits. </p>
    <figure>
      
      
    </figure>
    <a href="#openModal" class="linkModel">Read More</a>
    <!-- Modal Window container -->
    <div id="openModal" class="modalWindow">
      <div class="msgText">
        <h2>Sunflower <a href="#home" class="modal-closeX">X</a> </h2>
        <p>There are over three hundred species and thousands of cultivars.
          They form a group of plants that can be erect shrubs, climbing, or
          trailing, with stems that are often armed with sharp prickles.</p>
        
        <a href="#home" class="modal-close">CLOSE</a>
      </div>
    </div>
  </body>
</html>
```

In CSS, we add padding to **<h3>** and **<p>**, and align the text to justify in **<p>**. Remember, to call **<h3>** and **<p>** element within the **.msgText** division, we will need to call the division first follows with the element within it, for example: **.msgText h3**.

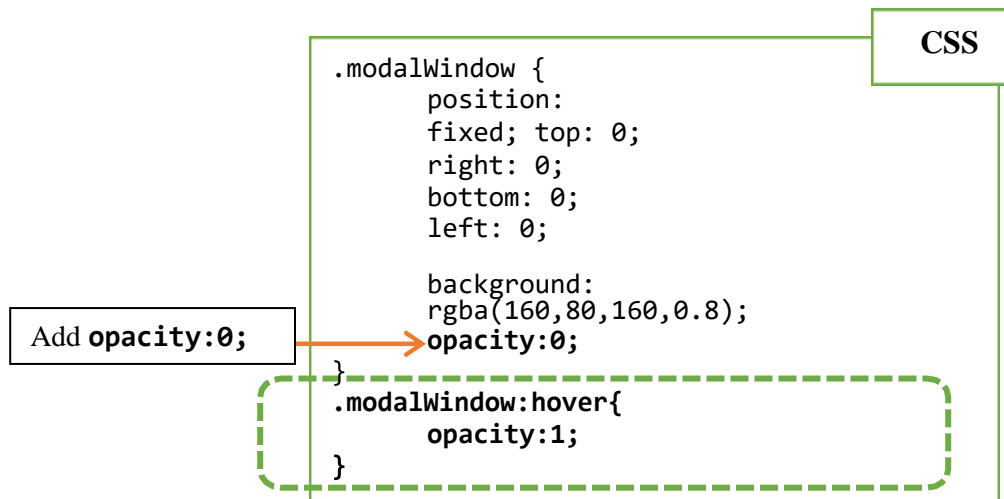
The app should look as the following:



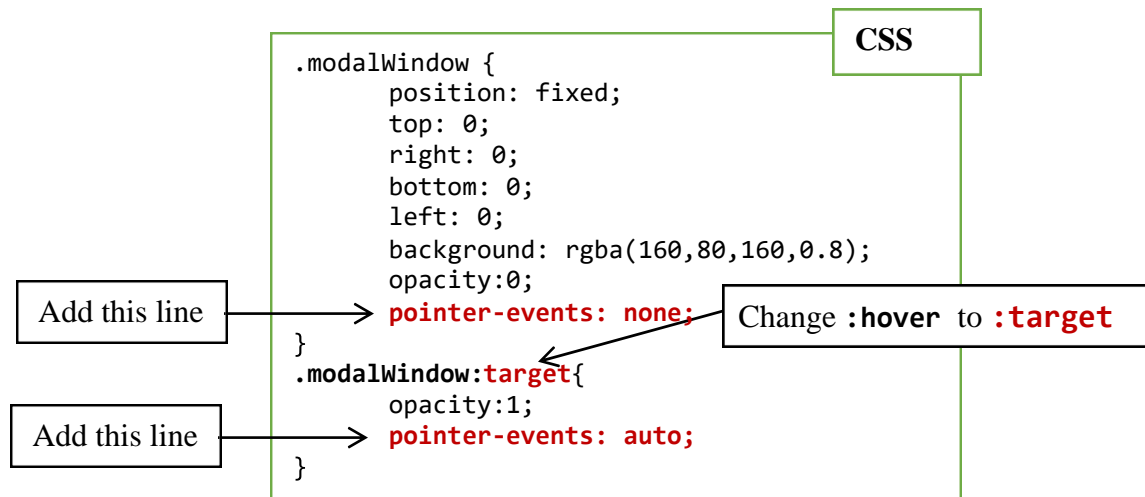
CSS

```
.msgText h2{
  padding: 20px;
  background-color: darkgreen;
  color: orange;
}
.modal-closeX{
  color:orange;
  float: right;
  text-decoration: none;
}
.msgText p{
  padding: 20px;
  color:white;
}
.msgText img{
  width: 80%;
  height: auto;
  margin:10%;
  box-shadow: 0px 0px 10px 5px darkgreen;
}
.modal-close {
  color:white;
  background-color: orange;
  padding: 5px 20px;
  float: right;
  text-decoration: none;
  margin-right: 10%;
  margin-top: 10%;
}
```

Once we have the model set, we can set the opacity attribute to it. The idea is to make the modal window to invisible, **opacity:0;** , when the web app is loaded, then when the model link is hover, the modal window will become visible, **opacity: 1;**



If we run the app, the message window will not close when we click on the CLOSE link. This happens because we did not set the **pointer-events** to **target** an element that has the CLOSE link. For this, we can include **pointer-events:none**; to the element that has the anchor tag **<a>** beneath it. In this activity, we target the modal window **.modalWindow** and when click on the CLOSE link, the model window will close, that is why we need to change **.modalWindow:hover** to **.modalWindow:target** and add a **pointer-events: auto**; which means that the any anchor **<a>** tag under **.modalWindow** will react to the pointer events. In other words, when we click on the CLOSE link, the anchor **<a>** tag will take us to the beginning to the web app as states with **href="#"**.



We can also add some animation to our modal window, for example, we can make the modal window to **ease-in** when it opens.

```

.modalWindow:target{
  opacity:1;
  pointer-events: auto;
  transition: ease-in 0.5s;
}

```