- **Installing Dependency Software**

1. jdk
2. node.js
3. intelhaxm-android
4. git-2.13.0
5. Extract gradle-4.0 in C drive and rename folder as Gradle
6. Android studio

- **Adding System Variables**

  set JAVA_HOME=C:\Program Files\Java\jdk1.8.0_131

  set GRADLE_HOME=C:\Gradle

  set ANDROID_HOME=C:\Users\lab1-18\AppData\Local\Android\sdk

---

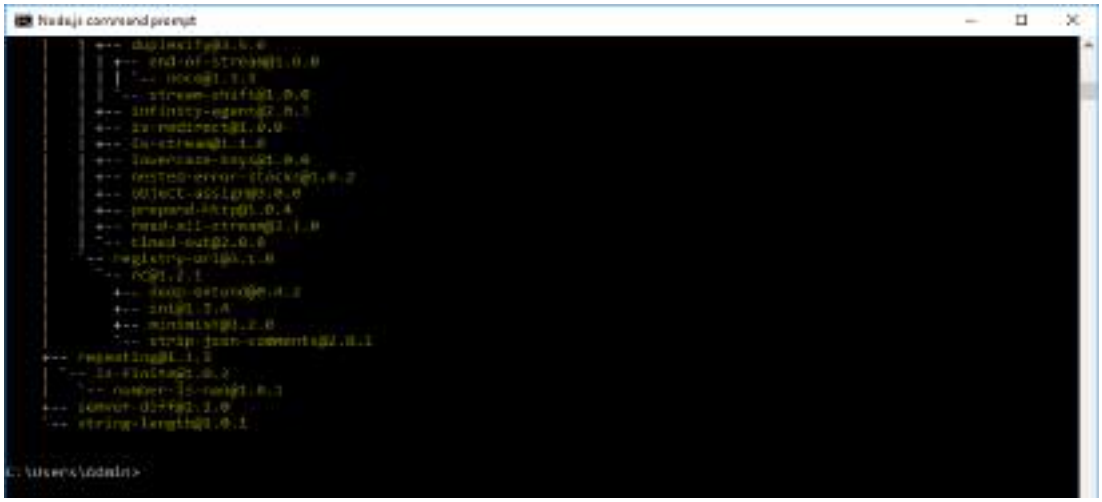
- **Path Setting**

Path= C:\Program Files\Java\jdk1.8.0_60\bin; C:\Gradle\bin; C:\Program Files\Git\cmd; C:\Program Files\nodejs; C:\Program Files\Git\bin; C:\Program Files\nodejs\node_modules\npm\bin; C:\Users\lab1-18\AppData\Roaming\npm

---

- **For Installing CORDOVA**

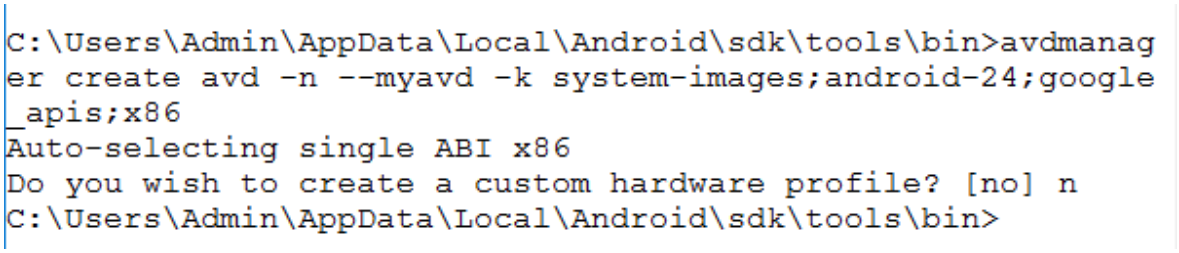
  Start command prompt and type command **npm install -g cordova**



Creating avd (Android Virtual Device)

1. Open command prompt and go to
   "**C:\Users\Admin\AppData\Local\Android\sdk\tools\bin**"

2. Create avd by using command
   avdmanager create avd -n --myavd -k system-images;android-24;google_apis;x86



PRACTICAL No. 1 (A)

Creating and Building Simple "Hello World" App using Cordova

1. Create cordova projects folder in D drive (we shall create all cordova projects in the same folder).
2. Open command prompt.
3. Change directory to D and go to cordova projects folder as shown below:

```
C:\Users\Admin>d:

D:\>cd cordova projects

D:\cordova projects>
```

4. Now create new cordova project with the help of following command:

   cordova create Hello io.cordova.hello HelloApp

```
D:\cordova projects>cordova create Hello io.cordova.hello HelloApp
Creating a new cordova project.

D:\cordova projects>
```

   - **Hello** is the directory name where the app is created.
   - **io.cordova.hello** is the domain value.
   - **HelloApp** is the title of your app.

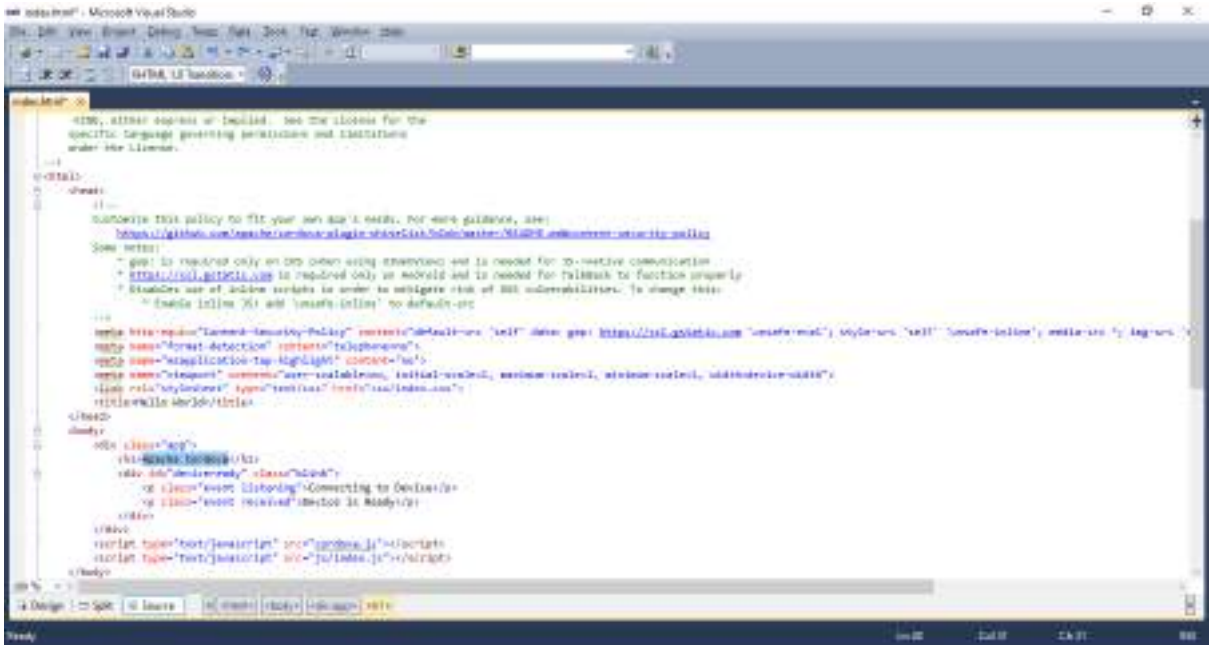
5. Go to Hello folder and add android platform with the help of following command:

   cordova platform add android

```
D:\cordova projects>cd Hello

D:\cordova projects\Hello>cordova platform add android
```

```
D:\cordova projects\Hello>cordova platform add android
Using cordova-fetch for cordova-android@~6.2.2
Adding android project...
Creating Cordova project for the Android platform:
        Path: platforms\android
        Package: io.cordova.hello
        Name: HelloApp
        Activity: MainActivity
        Android target: android-25
Subproject Path: CordovaLib
Android project created with cordova-android@6.2.3
Discovered plugin "cordova-plugin-whitelist" in config.xml. Adding
it to the project
Installing "cordova-plugin-whitelist" for android

               This plugin is only applicable for versions of cordo
va-android greater than 4.0. If you have a previous platform versio
n, you do *not* need this plugin since the whitelist will be built
in.

Adding cordova-plugin-whitelist to package.json
Saved plugin info for "cordova-plugin-whitelist" to config.xml
--save flag or autosave detected
Saving android@~6.2.3 into config.xml file ...

D:\cordova projects\Hello>
```

6. For “Hello World” app, we shall make changes in index.html file which is present at the location **“D:\cordova projects\myapp\www”.** We can open index.html file with various applications (notepad, wordpad, visual studio), preferably wordpad or visual studio.



Now we shall replace word “Apache Cordova” with “Hello world” in index.html.

Before:

```
    <meta name="viewport" content="user-scalable=no, initial-scale=1, maximum-scale=1, minimum-scale=1, width=device-width">
    <link rel="stylesheet" type="text/css" href="css/index.css">
    <title>Hello World</title>
</head>
<body>
    <div class="app">
        <h1>Apache Cordova</h1>
        <div id="deviceready" class="blink">
            <p class="event listening">Connecting to Device</p>
            <p class="event received">Device is Ready</p>
        </div>
```

After:

```
    <title>Hello World</title>
</head>
<body>
    <div class="app">
        <h1>Hello World</h1>
        <div id="deviceready" class="blink">
            <p class="event listening">Connecting to Device</p>
            <p class="event received">Device is Ready</p>
        </div>
```

Next save index.html file.

7. Build app with help of command:

cordova build android

```
D:\cordova projects\Hello>cordova build android
ANDROID_HOME=C:\Users\Admin\AppData\Local\Android\sdk
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_131

BUILD SUCCESSFUL in 3s
1 actionable task: 1 executed
Subproject Path: CordovaLib
The Task.leftShift(Closure) method has been deprecated and is sched
uled to be removed in Gradle 5.0. Please use Task.doLast(Action) in
stead.
        at build_1r9ljtzwkz8h76naed3omtg4j.run(D:\cordova projects\
Hello\platforms\android\build.gradle:137)
```

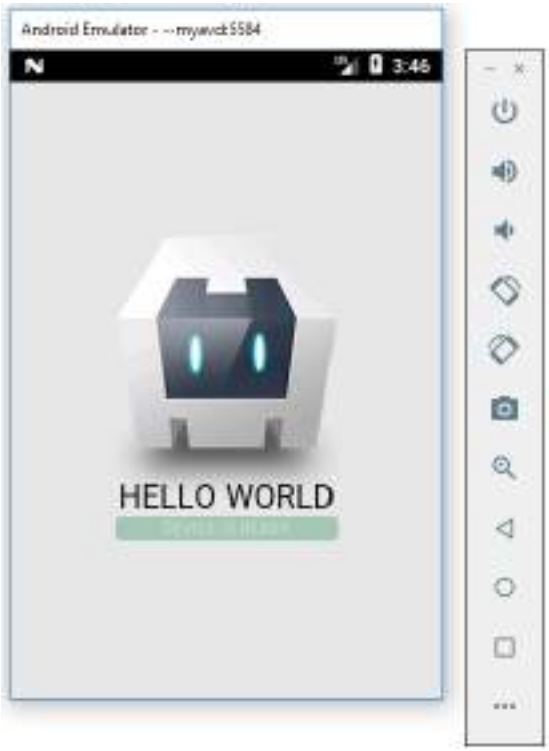
```
:transformResourcesWithMergeJavaResForDebug
:validateSigningDebug
:packageDebug
:assembleDebug
:cdvBuildDebug

BUILD SUCCESSFUL

Total time: 30.866 secs
Built the following apk(s):
        D:/cordova projects/Hello/platforms/android/build/outputs/a
pk/android-debug.apk

D:\cordova projects\Hello>
```

8. Emulate app on avd with the help of command:

cordova emulate android

```
D:\cordova projects\Hello>cordova emulate android
ANDROID_HOME=C:\Users\Admin\AppData\Local\Android\sdk
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_131
Subproject Path: CordovaLib
The Task.leftShift(Closure) method has been deprecated and is sched
uled to be removed in Gradle 5.0. Please use Task.doLast(Action) in
stead.

Waiting for emulator to start...
Waiting for emulator to boot (this may take a while)....BOOT COMPLE
TE
Skipping build...
Built the following apk(s):
        D:/cordova projects/Hello/platforms/android/build/outputs/a
pk/android-debug.apk
Using apk: D:/cordova projects/Hello/platforms/android/build/output
s/apk/android-debug.apk
Package name: io.cordova.hello
INSTALL SUCCESS
LAUNCH SUCCESS

D:\cordova projects\Hello>
```

Output:



PRACTICAL No. 1 (B)

Adding and Using Event Listeners with Functions

1. Create new cordova project with the help of following command:

   cordova create Events io.cordova.events EventListenerApp

```
D:\cordova projects>cordova create Events io.cordova.events EventListenerApp
Creating a new cordova project.

D:\cordova projects>
```

2. Go to Events folder and add android platform with the help of following command:

   cordova platform add android

```
D:\cordova projects>cd Events

D:\cordova projects\Events>cordova platform add android
Using cordova-fetch for cordova-android@~6.2.2
Adding android project...
Creating Cordova project for the Android platform:
        Path: platforms\android
        Package: io.cordova.events
        Name: EventListenerApp
        Activity: MainActivity
        Android target: android-25
Subproject Path: CordovaLib
Android project created with cordova-android@6.2.3
Discovered plugin "cordova-plugin-whitelist" in config.xml. Adding it to the
project
Installing "cordova-plugin-whitelist" for android

               This plugin is only applicable for versions of cordova-android
 greater than 4.0. If you have a previous platform version, you do *not* need
 this plugin since the whitelist will be built in.

Adding cordova-plugin-whitelist to package.json
Saved plugin info for "cordova-plugin-whitelist" to config.xml
--save flag or autosave detected
Saving android@~6.2.3 into config.xml file ...

D:\cordova projects\Events>
```

3. For adding and using Event Listeners with Functions we shall make changes in index.js file which is present at the location **"D:\cordova projects\myapp\www\js".** Open index.js file.

```
*/
var app = {
    // Application Constructor
    initialize: function() {
        document.addEventListener('deviceready', this.onDeviceReady.bind(this), false);
    },

    // deviceready Event Handler
    //
    // Bind any cordova events here. Common events are:
    // 'pause', 'resume', etc.
    onDeviceReady: function() {
        this.receivedEvent('deviceready');
    },

    // Update DOM on a Received Event
    receivedEvent: function(id) {
        var parentElement = document.getElementById(id);
        var listeningElement = parentElement.querySelector('.listening');
        var receivedElement = parentElement.querySelector('.received');

        listeningElement.setAttribute('style', 'display:none;');
        receivedElement.setAttribute('style', 'display:block;');

        console.log('Received Event: ' + id);
    }
};

app.initialize();
```

4. Inside the receivedEvent function add Event Listener as shown below:

```
document.addEventListener("volumeupbutton", volUpFunction, false);
document.addEventListener("volumedownbutton", volDownFunction, false);
```

Syntax:

```
document.addEventListener(event, function, useCapture);
```

useCapture	Optional. A Boolean value that specifies whether the event should be executed in the capturing or in the bubbling phase. Possible values: • true - The event handler is executed in the capturing phase • false- Default. The event handler is executed in the bubbling phase

There are various events that can be used in Cordova projects. The following table shows the available events.

S.No	Events & Details
1	**deviceReady** This event is triggered once Cordova is fully loaded. This helps to ensure that no Cordova functions are called before everything is loaded.
2	**pause** This event is triggered when the app is put into background.
3	**resume** This event is triggered when the app is returned from background.
4	**backbutton** This event is triggered when the back button is pressed.
5	**menubutton** This event is triggered when the menu button is pressed.
6	**searchbutton** This event is triggered when the Android search button is pressed.
7	**startcallbutton** This event is triggered when the start call button is pressed.
8	**endcallbutton** This event is triggered when the end call button is pressed.
9	**volumedownbutton** This event is triggered when the volume down button is pressed.
10	**volumeupbutton** This event is triggered when the volume up button is pressed.

5. Before the app.initialize(); add function as shown below:

```
function volUpFunction() {
   alert('I am Volume Up Button!!!');
};
function volDownFunction() {
   alert('I am Volume Down Button!!!');
};
```

```
    // Update DOM on a Received Event
    receivedEvent: function(id) {
        document.addEventListener("volumeupbutton", volUpFunction, false);
        document.addEventListener("volumedownbutton", volDownFunction, false);
        var parentElement = document.getElementById(id);
        var listeningElement = parentElement.querySelector('.listening');
        var receivedElement = parentElement.querySelector('.received');

        listeningElement.setAttribute('style', 'display:none;');
        receivedElement.setAttribute('style', 'display:block;');

        console.log('Received Event: ' + id);
    }
};
    function volUpFunction() {
        alert('I am Volume Up Button!!!');
    };
    function volDownFunction() {
        alert('I am Volume Down Button!!!');
    };

app.initialize();
```

Next save index.js file.

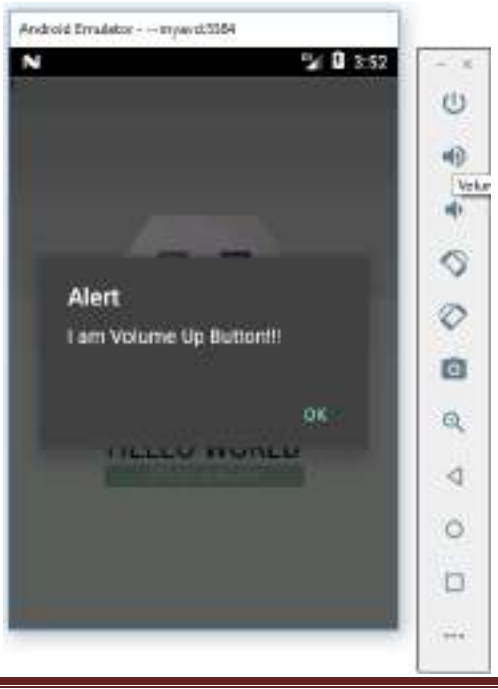
6. Build and emulate on avd.

```
D:\cordova projects\Events>cordova build android
ANDROID_HOME=C:\Users\Admin\AppData\Local\Android\sdk
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_131
<-------------> 0% INITIALIZING [0s]
> settings > Compiling D:\cordova projects\Events\platforms\android\settings
```

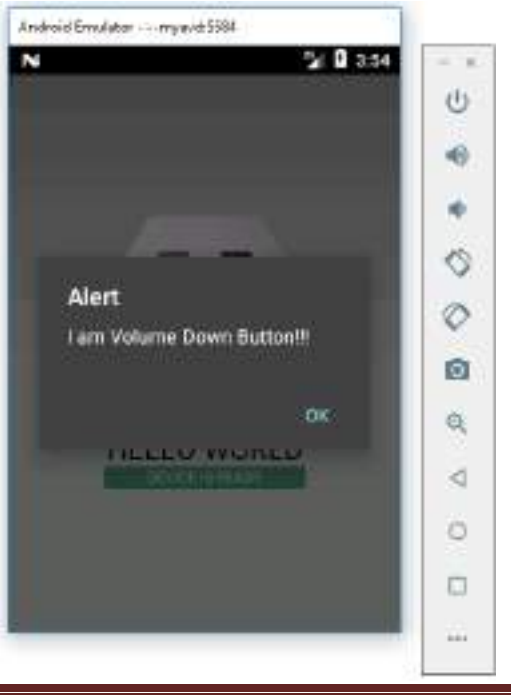
```
D:\cordova projects\Events>cordova emulate android
ANDROID_HOME=C:\Users\Admin\AppData\Local\Android\sdk
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_131
Subproject Path: CordovaLib
```

Output:

1. **Output after click volume up button**



2. **Output after click volume up button**



PRACTICAL No. 2 (A)

Adding and Using Buttons

1. Create new cordova project with the help of following command:

   cordova create Button io.cordova.Button usingButton

```
D:\cordova projects>cordova create Button io.cordova.Button usingButton
Creating a new cordova project.

D:\cordova projects>
```

2. Go to Button folder and add android platform with the help of following command:

   cordova platform add android

3. We shall add the following code in index.html file to add button in our app, under .....

   <button id="welcome">Click me</button>

```
</head>
<body>
    <div class="app">
        <button id="welcome">Click me</button>
        <div id="deviceready" class="blink">
            <p class="event listening">Connecting to Device</p>
            <p class="event received">Device is Ready</p>
        </div>
```

   Next save index.html file.

4. For using button, open index.js and add event listener inside the onDeviceReady function as shown below:

   document.getElementById("welcome").addEventListener("click",welFun);

5. And before the app.initialize(); add function as shown below:

```
function welFun() {
   alert('Welcome to cordova');
}
```

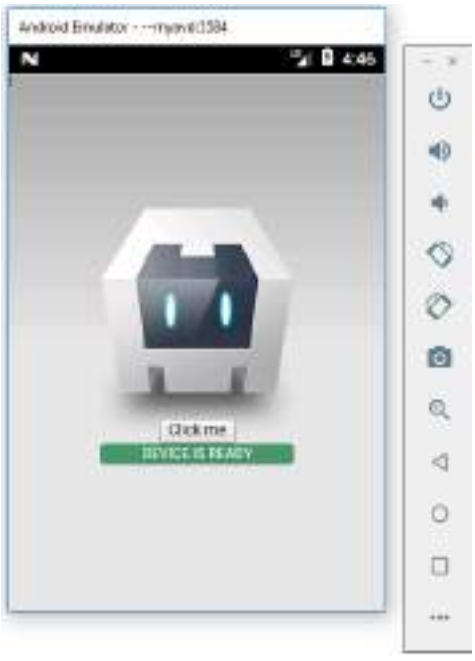
   Next save index.js file.

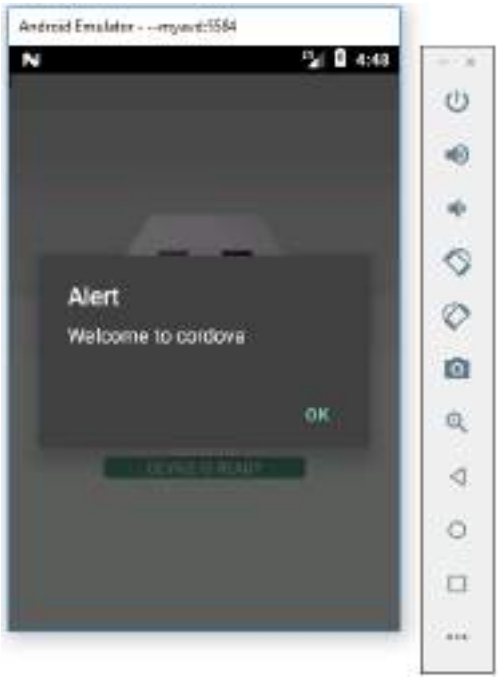
```
    onDeviceReady: function () {
        document.getElementById("welcome").addEventListener("click", welFun);
        this.receivedEvent('deviceready');
    },

    // Update DOM on a Received Event
    receivedEvent: function(id) {
        var parentElement = document.getElementById(id);
        var listeningElement = parentElement.querySelector('.listening');
        var receivedElement = parentElement.querySelector('.received');

        listeningElement.setAttribute('style', 'display:none;');
        receivedElement.setAttribute('style', 'display:block;');

        console.log('Received Event: ' + id);
    }
};

function welFun() {
    alert('Welcome to cordova');
}
app.initialize();
```

6. Build and emulate on avd.

OUTPUT:

After Click on Click me button





PRACTICAL No. 2 (B)

Handling and Using Back Button

1. Create new cordova project with the help of following command:

   cordova create BackButton io.cordova.BB BackButton

```
D:\cordova projects>cordova create BackButton io.cordova.BB BackButton
Creating a new cordova project.

D:\cordova projects>
```

2. Go to BackButton folder and add android platform with the help of following command:

   cordova platform add android

3. Open index.js file and add Event Listener inside the receivedEvent function as shown below:

```
document.addEventListener('backbutton', onBackButton,false);
```

4. And before the app.initialize(); add function as shown below:

```
function onBackButton(e) {
   e.preventDefault();
   alert('I am Back Button.');
}
```

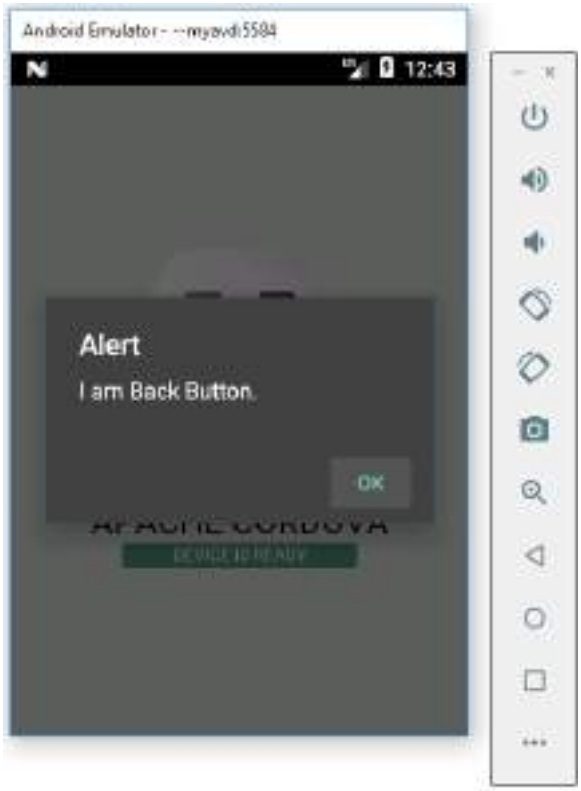
   Next save index.js file.

We usually wants to use Android back button for some app functionality like returning to previous screen. To be able to implement own functionality, we first need to disable exiting the app when the back button is pressed. This is done by using **e.preventDefault()**. Now when we press the native Android back button, the alert will appear on the screen instead of exiting the app.

```
var app = {
    // Application Constructor
    initialize: function () {
        document.addEventListener('deviceready', this.onDeviceReady.bind(this), false);
    },

    // deviceready Event Handler
    //
    // Bind any cordova events here. Common events are:
    // 'pause', 'resume', etc.
    onDeviceReady: function () {
        this.receivedEvent('deviceready');
    },

    // Update DOM on a Received Event
    receivedEvent: function (id) {
        document.addEventListener('backbutton', onBackButton,false);
        var parentElement = document.getElementById(id);
        var listeningElement = parentElement.querySelector('.listening');
        var receivedElement = parentElement.querySelector('.received');

        listeningElement.setAttribute('style', 'display:none;');
        receivedElement.setAttribute('style', 'display:block;');

        console.log('Received Event: ' + id);
    }
};
function onBackButton(e) {
    e.preventDefault();
    alert('I am Back Button.');
}
app.initialize();
```

5. Build and emulate on avd.

OUTPUT:



PRACTICAL No. 3 (A)

Installing and Using Battery Plugins

Cordova Plugman is a useful command line tool for installing and managing plugins.

1. Open the command prompt window and run the following code to install plugman.

   npm install –g plugman

```
D:\cordova projects>npm install -g plugman
```

2. Create new cordova project with the help of following command:

   cordova create Battery_plugin io.cordova.battery BatteryStatus

```
D:\cordova projects>cordova create Battery_plugin io.cordova.battery BatteryStatus
Creating a new cordova project.
```

3. Go to Batter_plugin folder and add android platform with the help of following command:

   cordova platform add android

4. Add Battery Status Plugin with the help of following command:

   cordova plugin add cordova-plugin-battery-status

```
D:\cordova projects\Battery_plugin>cordova plugin add cordova-plugin-battery-status
Installing "cordova-plugin-battery-status" for android
Adding cordova-plugin-battery-status to package.json
Saved plugin info for "cordova-plugin-battery-status" to config.xml

D:\cordova projects\Battery_plugin>
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

   window.addEventListener('batterystatus', OnBatteryStatus, false);

6. And before the app.initialize(); add function as shown below:

```
function OnBatteryStatus(info) {
   alert('BATTERY STATUS:\n Level: ' + info.level + ' % \n Is plugged? ' +
info.isPlugged);
}
```

```
    // 'pause', 'resume', etc.
    onDeviceReady: function () {
        window.addEventListener('batterystatus', OnBatteryStatus, false);
        this.receivedEvent('deviceready');
    },

    // Update DOM on a Received Event
    receivedEvent: function (id) {
        var parentElement = document.getElementById(id);
        var listeningElement = parentElement.querySelector('.listening');
        var receivedElement = parentElement.querySelector('.received');

        listeningElement.setAttribute('style', 'display:none;');
        receivedElement.setAttribute('style', 'display:block;');

        console.log('Received Event: ' + id);
    }
};
function OnBatteryStatus(info) {
    alert('BATTERY STATUS:\n Level: ' + info.level + '\n Is plugged? ' + info.isPlugged);
}
app.initialize();
```

7. Build and emulate on avd.

OUTPUT:

Additional Events

This plugin offers two additional events besides the **batterystatus** event. These events can be used in the same way as the **batterystatus** event.

S.No	Event & Details
1	**batterylow** The event is triggered when the battery charge percentage reaches low value. This value varies with different devices.
2	**batterycritical** The event is triggered when the battery charge percentage reaches critical value. This value varies with different devices.

So, we can also add the following event listeners and functions in index.js file.

```
window.addEventListener('batterylow', onBatteryLow, false);
window.addEventListener('batterycritical', onBatteryCritical, false);

function onBatteryLow(status) {
   alert('Battery is in low level' + status.level + '%');
}
function onBatteryCritical(status) {
   alert('Battery is critical now' + status.level + '%\n Start charging soon!!!');
}
```

PRACTICAL No. 3 (B)
Installing and Using Camera Plugin

1. Create new cordova project with the help of following command:

   cordova create Camera io.cordova.camera CameraPluginApp

```
D:\cordova projects>cordova create Camera io.cordova.camera CameraPluginApp
Creating a new cordova project.

D:\cordova projects>
```

2. Go to Camera folder and add android platform with the help of following command:

   cordova platform add android

3. Add Camera Plugin with the help of following command:

   cordova plugin add cordova-plugin-camera

```
D:\cordova projects\Camera>cordova plugin add cordova-plugin-camera
Installing "cordova-plugin-camera" for android
Installing "cordova-plugin-compat" for android
Subproject Path: CordovaLib
Adding cordova-plugin-camera to package.json
Saved plugin info for "cordova-plugin-camera" to config.xml

D:\cordova projects\Camera>
```

4. Now, we shall create **button** for calling the camera and **img** where the image will be displayed once taken. We shall add the following code in **index.html** file to add button and image in our app, under .....

```
<button id = "cameraButton">Camera</button><br>
<img id = "myImage" height="100" width="100"></img>
```

```
<div class="app">
    <button id = "cameraButton">Camera</button><br>
    <img id = "myImage" height="100" width="100"></img>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("cameraButton").addEventListener("click",
takePicture);
```

```
onDeviceReady: function() {
    document.getElementById("cameraButton").addEventListener("click", takePicture);
    this.receivedEvent('deviceready');
},
```

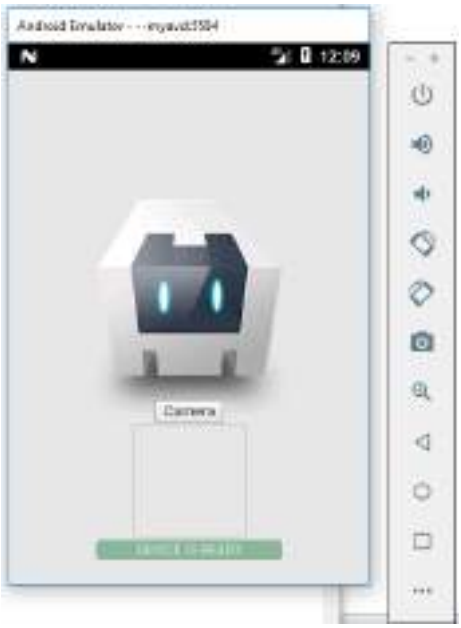
6. And before the app.initialize(); add function as shown below:

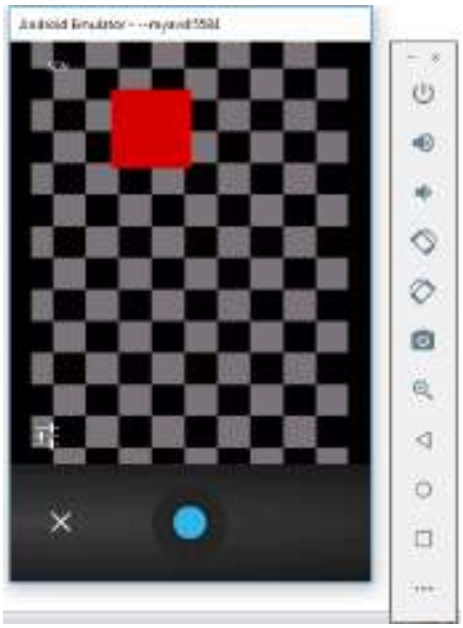
```
function takePicture() {
    navigator.camera.getPicture(onSuccess, onFailure, {
      quality: 100,
      destinationType: Camera.DestinationType.DATA_URL
    });


    function onSuccess(imageData) {
      var image = document.getElementById('myImage');
      image.src = "" + imageData;
    }

    function onFailure(msg) {
      alert('Failed because: ' + msg);
    }
}
```

```
function takePicture() {
    navigator.camera.getPicture(onSuccess, onFailure, {
        quality: 100,
        destinationType: Camera.DestinationType.DATA_URL
    });

    function onSuccess(imageData) {
        var image = document.getElementById('myImage');
        image.src = "" + imageData;
    }

    function onFailure(msg) {
        alert('Failed because: ' + msg);
    }
}

app.initialize();
```

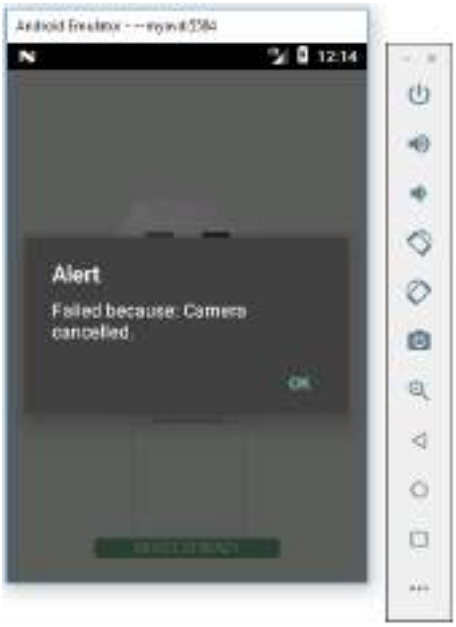
7. Build and emulate on avd.

OUTPUT:


12:09
Camera




12:13
Camera
DEVICE IS READY


12:14
Alert
Failed because: Camera cancelled.
OK

PRACTICAL No. 4 (A)
Installing and Using Contacts Plugin

1. Create new cordova project with the help of following command:

   cordova create Contacts io.cordova.contact ContactsPluginApp

```
D:\cordova projects>cordova create Contacts io.cordova.contact ContactsPluginApp
Creating a new cordova project.

D:\cordova projects>
```

2. Go to Contacts folder and add android platform with the help of following command:

   cordova platform add android

3. Add Contacts Plugin with the help of following command:

   cordova plugin add cordova-plugin-contacts

```
D:\cordova projects\Contacts>cordova plugin add cordova-plugin-contacts
Installing "cordova-plugin-contacts" for android
Installing "cordova-plugin-compat" for android
Adding cordova-plugin-contacts to package.json
Saved plugin info for "cordova-plugin-contacts" to config.xml

D:\cordova projects\Contacts>
```

4. Now, we shall create 3 **buttons** to add, search and delete contacts. We shall add the following code in **index.html** file to add buttons in our app, under .....

```
<button id = "saveButton">SAVE CONTACT</button> <br />
<button id = "searchButton">SEARCH CONTACT</button> <br />
<button id = "deleteButton">DELETE CONTACT</button>
```

```
<div class="app">
    <button id = "saveButton">SAVE CONTACT</button><br />
    <button id = "searchButton">SEARCH CONTACT</button><br />
    <button id = "deleteButton">DELETE CONTACT</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("saveButton").addEventListener("click", saveContact);
document.getElementById("searchButton").addEventListener("click", searchContact);
document.getElementById("deleteButton").addEventListener("click", deleteContact);
```

```
onDeviceReady: function() {
    document.getElementById("saveButton").addEventListener("click", saveContact);
    document.getElementById("searchButton").addEventListener("click", searchContact);
    document.getElementById("deleteButton").addEventListener("click", deleteContact);
    this.receivedEvent('deviceready');
},
```

6. And before the app.initialize(); add function as shown below:

```
function saveContact() {
    var myContact = navigator.contacts.create({ "displayName": "Ajay"});
    myContact.save(onSaveSuccess, onSaveFailure);
    function onSaveSuccess() {
       alert("Contact is saved!");
    }

    function onSaveFailure(message) {
       alert('Failed because: ' + message);
    }
}

function searchContact() {
    var options = new ContactFindOptions();
    options.filter = "Ajay";
    options.multiple = true;
    fields = ["displayName"];
    navigator.contacts.find(fields, onFindSuccess, onFindFailure, options);

    function onFindSuccess(contacts) {
       for (var i = 0; i < contacts.length; i++) {
          alert("Display Name = " + contacts[i].displayName);
       }
    }

    function onFindFailure(msg) {
       alert('Failed because: '+msg);
    }
  }


function deleteContact() {
    var options = new ContactFindOptions();
    options.filter = "Ajay";
    options.multiple = true;
    fields = ["displayName"];
    navigator.contacts.find(fields, onFindSuccess, onFindFailure, options);
```

```
    function onFindSuccess(contacts) {
        var contact = contacts[0];
        contact.remove(onRemoveSuccess, onRemoveFailure);

        function onRemoveSuccess(contact) {
            alert("Contact Deleted");
        }

        function onRemoveFailure(message) {
            alert('Failed because: ' + message);
        }
    }

    function onFindFailure(message) {
        alert('Failed because: ' + message);
    }

}
```

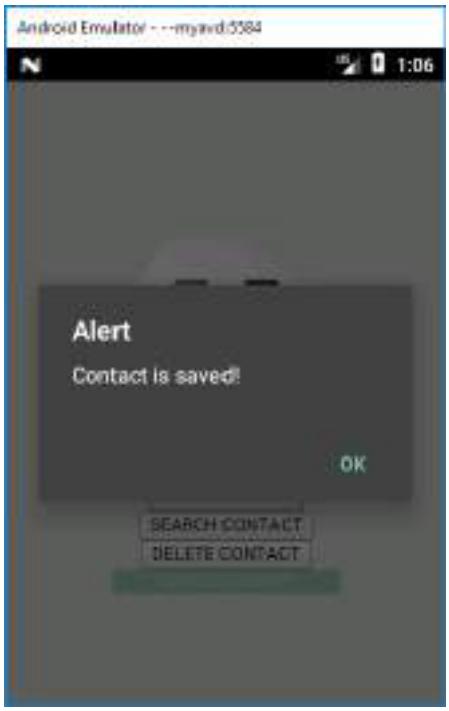
```
function saveContact() {
    var myContact = navigator.contacts.create({ "displayName": "Ajay" });
    myContact.save(onSaveSuccess, onSaveFailure);
    function onSaveSuccess() {
        alert("Contact is saved!");
    }

    function onSaveFailure(message) {
        alert('Failed because: ' + message);
    }
}

function searchContact() {
    var options = new ContactFindOptions();
    options.filter = "Ajay";
    options.multiple = true;
    fields = ["displayName"];
    navigator.contacts.find(fields, onFindSuccess, onFindFailure, options);

    function onFindSuccess(contacts) {
        for (var i = 0; i < contacts.length; i++) {
            alert("Display Name = " + contacts[i].displayName);
        }
    }

    function onFindFailure(msg) {
        alert('Failed because: ' + msg);
    }
}
```

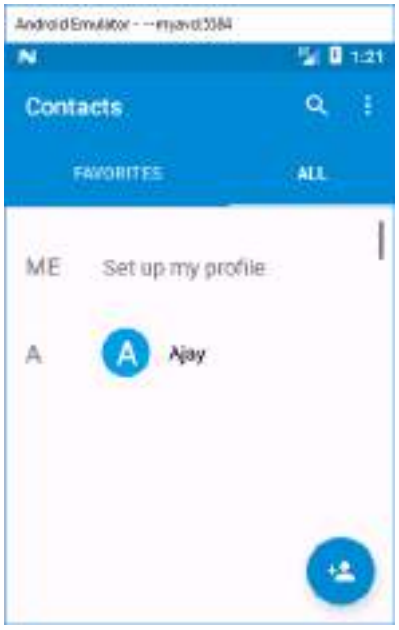
```
function deleteContact() {
    var options = new ContactFindOptions();
    options.filter = "Ajay";
    options.multiple = true;
    fields = ["displayName"];
    navigator.contacts.find(fields, onFindSuccess, onFindFailure, options);

    function onFindSuccess(contacts) {
        var contact = contacts[0];
        contact.remove(onRemoveSuccess, onRemoveFailure);

        function onRemoveSuccess(contact) {
            alert("Contact Deleted");
        }

        function onRemoveFailure(message) {
            alert('Failed because: ' + message);
        }
    }

    function onFindFailure(message) {
        alert('Failed because: ' + message);
    }

}
```

7. Build and emulate on avd.

OUTPUT:

After click on ADD CONTACT button





After click on FIND CONTACT button

After click on DELETE CONTACT button

PRACTICAL No. 4 (B)
Installing and Using Device Plugin

1. Create new cordova project with the help of following command:

   cordova create DevicePlugin io.cordova.dev DevicePluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create DevicePlugin io.cordova.dev DevicePluginApp
Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>
```

2. Go to DevicePlugin folder and add android platform with the help of following command:

   cordova platform add android

3. Add Device Plugin with the help of following command:

   cordova plugin add cordova-plugin-device

```
E:\Cordova Projects AjaySIESCE\DevicePlugin>cordova plugin add cordova-plugin-device
Installing "cordova-plugin-device" for android
Adding cordova-plugin-device to package.json
Saved plugin info for "cordova-plugin-device" to config.xml

E:\Cordova Projects AjaySIESCE\DevicePlugin>
```

4. Now add a **button** in app with the help of following code:

```
<button id = "deviceInfo">Device Information</button>
```

```
<div class="app">
    <button id = "deviceInfo">Device Information</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("deviceInfo").addEventListener("click", Info);
```

```
onDeviceReady: function() {
    document.getElementById("deviceInfo").addEventListener("click", Info);
    this.receivedEvent('deviceready');
},
```

6. And before the app.initialize(); add function as shown below:

```
function Info() {
   alert("Cordova Version: " + device.cordova +
    "\nDevice Model: " + device.model +
    "\nDevice Platform: " + device.platform +
    "\nDevice UUID: " + device.uuid +
    "\nDevice Version: " + device.version);
}
```

```
function Info() {
    alert("Cordova Version: " + device.cordova +
      "\nDevice Model: " + device.model +
      "\nDevice Platform: " + device.platform +
      "\nDevice UUID: " + device.uuid +
      "\nDevice Version: " + device.version);
}
app.initialize();
```

7. Build and emulate on avd.

OUTPUT:

After click on Device Information button

1. The device.cordova returns the version of Cordova running on the device.
2. The device.model returns the name of the device's model or product.
3. The device.platform returns the device's operating system name.
4. The device.uuid returns the device's Universally Unique Identifier.
5. The device.version returns the operating system version.

PRACTICAL No. 4 (C)
Installing and Using Device Motion Plugin (Accelerometer)

This plugin provides access to the device's accelerometer. The accelerometer is a motion sensor that detects the change (*delta*) in movement relative to the current device orientation, in three dimensions along the *x*, *y*, and *z* axis.

1. Create new cordova project with the help of following command:

   cordova create AcceleroPlugin io.cordova.acl AccelerometerPluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create AcceleroPlugin io.cordova.acl AccelerometerPluginApp

Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>
```

2. Go to AcceleroPlugin folder and add android platform with the help of following command:

   cordova platform add android

3. Add Device Motion Plugin with the help of following command:

   cordova plugin add cordova-plugin-device-motion

```
E:\Cordova Projects AjaySIESCE\AcceleroPlugin>cordova plugin add cordova-plugin-device-motion
Installing "cordova-plugin-device-motion" for android
Adding cordova-plugin-device-motion to package.json
Saved plugin info for "cordova-plugin-device-motion" to config.xml

E:\Cordova Projects AjaySIESCE\AcceleroPlugin>
```

4. Now add 2 buttons in app with the help of following code:

```
<button id = "getAcl">GET ACCELERATION</button>
<button id = "watchAcl">WATCH ACCELERATION</button>
```

```
<div class="app">
    <button id = "getAcl">GET ACCELERATION</button>
    <button id = "watchAcl">WATCH ACCELERATION</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("getAcl").addEventListener("click", getAcceleration);
document.getElementById("watchAcl").addEventListener("click", watchAcceleration);
```

```
onDeviceReady: function() {
    document.getElementById("getAcl").addEventListener("click", getAcceleration);
    document.getElementById("watchAcl").addEventListener("click", watchAcceleration);
    this.receivedEvent('deviceready');
},
```

6. And before the app.initialize(); add function as shown below:

```
function getAcceleration() {
    navigator.accelerometer.getCurrentAcceleration(onGetSuccess,
onGetFailure)
    function onGetSuccess(acl) {
        alert('Acceleration X: ' + acl.x +
         '\n Acceleration Y: ' + acl.y +
         '\n Acceleration Z: ' + acl.z +
         '\n Timestamp: ' + acl.timestamp);
    };
    function onGetFailure() {
        alert('Error!!!');
    };
}

function watchAcceleration() {
    var aclOptions = {
        frequency: 3000
    }
    var watchID = navigator.accelerometer.watchAcceleration(onWatchSuccess,
onWatchFailure, aclOptions);

    function onWatchSuccess(acl) {
        alert('Acceleration X: ' + acl.x +
         '\n Acceleration Y: ' + acl.y +
         '\n Acceleration Z: ' + acl.z+
         '\n Timestamp: ' + acl.timestamp);

        setTimeout(function () {
            navigator.accelerometer.clearWatch(watchID);
        }, 10000);
    };

    function onWatchFailure() {
        alert('Error!!!');
    };
}
```

```
function getAcceleration() {
    function getAcceleration() {
     navigator.accelerometer.getCurrentAcceleration(onGetSuccess, onGetFailure)
     function onGetSuccess(acl) {
         alert('Acceleration X: ' + acl.x +
          '\n Acceleration Y: ' + acl.y +
          '\n Acceleration Z: ' + acl.z +
          '\n Timestamp: ' + acl.timestamp);
     };
     function onGetFailure() {
         alert('error!!!');
     };
}
function watchAcceleration() {
    var aclOptions = {
        frequency: 3000
    }
    var watchID = navigator.accelerometer.watchAcceleration(onWatchSuccess, onWatchFailure, aclOptions);

    function onWatchSuccess(acl) {
        alert('Acceleration X: ' + acl.x +
         '\n Acceleration Y: ' + acl.y +
         '\n Acceleration Z: ' + acl.z+
         '\n Timestamp: ' + acl.timestamp);

        setTimeout(function () {
            navigator.accelerometer.clearWatch(watchID);
        }, 10000);
    };

    function onWatchFailure() {
        alert('Error!!');
    };
}
```

7. Build and emulate on avd.

OUTPUT:

Device in vertical position:

Device in horizontalal position:

PRACTICAL No. 5 (A)
Installing and Using Device Orientation Plugin

This plugin provides access to the device's compass. The compass is a sensor that detects the direction or heading that the device is pointed, typically from the top of the device. It measures the heading in degrees from 0 to 359.99, where 0 is north.

1. Create new cordova project with the help of following command:
   cordova create Device_Orientation io.cordova.orientation DeviceOrientationPluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create Device_Orientation io.cordova.orientation DeviceOrientationPluginApp
Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>
```

2. Go to Device_Orientation folder and add android platform with the help of following command:
   cordova platform add android

3. Add Contacts Plugin with the help of following command:
   cordova plugin add cordova-plugin-device-orientation

```
E:\Cordova Projects AjaySIESCE\Device_Orientation>cordova plugin add cordova-plugin-device-orientation
Installing "cordova-plugin-device-orientation" for android
Adding cordova-plugin-device-orientation to package.json
Saved plugin info for "cordova-plugin-device-orientation" to config.xml

E:\Cordova Projects AjaySIESCE\Device_Orientation>
```

4. Now add 2 buttons in app with the help of following code:

```
<button id = "getOri">GET ORIENTATION</button>
<button id = "watchOri">WATCH ORIENTATION</button>
```

```
<div class="app">
    <button id = "getOri">GET ORIENTATION</button>
    <button id = "watchOri">WATCH ORIENTATION</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("getOri").addEventListener("click", getOrientation);
document.getElementById("watchOri").addEventListener("click", watchOrientation);
```

```
onDeviceReady: function() {
    document.getElementById("getOri").addEventListener("click", getOrientation);
    document.getElementById("watchOri").addEventListener("click", watchOrientation);
    this.receivedEvent('deviceready');
},
```

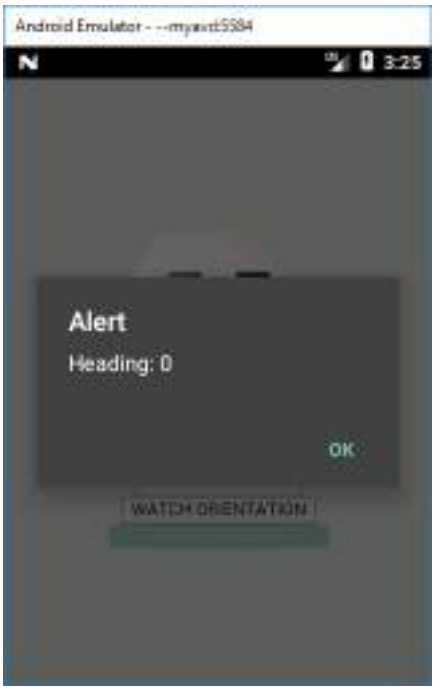
6. And before the app.initialize(); add function as shown below:

```
function getOrientation() {
    navigator.compass.getCurrentHeading(onGetSuccess, onGetFailure);

    function onGetSuccess(heading) {
        alert('Heading: ' + heading.magneticHeading);
    };

    function onGetFailure(msg) {
        alert('CompassError: ' + msg.code);
    };
}

function watchOrientation() {
    var compassOptions = {
        frequency: 2500
    }
    var watchID = navigator.compass.watchHeading(onWatchSuccess,
      onWatchFailure, compassOptions);

    function onWatchSuccess(heading) {
        alert('Heading: ' + heading.magneticHeading);

        setTimeout(function () {
            navigator.compass.clearWatch(watchID);
        }, 10000);
    };

    function onWatchFailure(msg) {
        alert('CompassError: ' + msg.code);
    };
}
```

```
function getOrientation() {
    navigator.compass.getCurrentHeading(onGetSuccess, onGetFailure);

    function onGetSuccess(heading) {
        alert('Heading: ' + heading.magneticHeading);
    };

    function onGetFailure(msg) {
        alert('CompassError: ' + msg.code);
    };
}

function watchOrientation() {
    var compassOptions = {
        frequency: 2500
    }
    var watchID = navigator.compass.watchHeading(onWatchSuccess,
      onWatchFailure, compassOptions);

    function onWatchSuccess(heading) {
        alert('Heading: ' + heading.magneticHeading);

        setTimeout(function () {
            navigator.compass.clearWatch(watchID);
        }, 10000);
    };

    function onWatchFailure(msg) {
        alert('CompassError: ' + msg.code);
    };
}
```

7. Build and emulate on avd.

OUTPUT:



PRACTICAL No. 5 (B)
Installing and Using Dialogs Plugin

This plugin provides access to some native dialog UI elements such as alert, confirm, prompt and beep.

1. Create new cordova project with the help of following command:

   cordova create Dialogs io.cordova.dialogs DialogsPluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create Dialogs io.cordova.dialogs DialogsPluginApp
Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>
```

2. Go to Dialogs folder and add android platform with the help of following command:

   cordova platform add android

3. Add Contacts Plugin with the help of following command:

   cordova plugin add cordova-plugin-dialogs

```
E:\Cordova Projects AjaySIESCE\Dialogs>cordova plugin add cordova-plugin-dialogs
Installing "cordova-plugin-dialogs" for android
Adding cordova-plugin-dialogs to package.json
Saved plugin info for "cordova-plugin-dialogs" to config.xml

E:\Cordova Projects AjaySIESCE\Dialogs>
```

4. Now add 4 buttons in app with the help of following code:

```
<button id = "alertButton">ALERT</button><br />
<button id = "confirmButton">CONFIRM</button><br />
<button id = "promptButton">PROMPT</button><br />
<button id = "beepButton">BEEP</button>
```

```
<div class="app">
    <button id = "alertButton">ALERT</button><br />
    <button id = "confirmButton">CONFIRM</button><br />
    <button id = "promptButton">PROMPT</button><br />
    <button id = "beepButton">BEEP</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("alertButton").addEventListener("click", getAlert);
document.getElementById("confirmButton").addEventListener("click",
getConfirmation);
document.getElementById("promptButton").addEventListener("click", getPrompt);
document.getElementById("beepButton").addEventListener("click", getBeep);
```

```
onDeviceReady: function() {
    document.getElementById("alertButton").addEventListener("click", getAlert);
    document.getElementById("confirmButton").addEventListener("click", getConfirmation);
    document.getElementById("promptButton").addEventListener("click", getPrompt);
    document.getElementById("beepButton").addEventListener("click", getBeep);
    this.receivedEvent('deviceready');
```

6. And before the app.initialize(); add function as shown below:

```
function getAlert() {
    var message = "Hi, I am Alert Dialog!!!";
    var title = "Alert Dialog Box";
    var buttonName = "OK";
    navigator.notification.alert(message, alertCallback, title, buttonName);
    function alertCallback() {
        alert("Alert is Dismissed!");
    }
}

function getConfirmation() {
    var message = "Are you a registered user?";
    var title = "Confirmation Dialog Box";
    var buttonLabels = "YES,NO";
    navigator.notification.confirm(message, confirmCallback, title, buttonLabels);
    function confirmCallback(buttonIndex) {
        alert("You clicked on button number: " + buttonIndex);
    }
}

function getPrompt() {
    var message = "Please provide your contact number?";
    var title = "Registration";
    var buttonLabels = "Cancel,Ok";
    var defaultText = ""
    navigator.notification.prompt(message, promptCallback,
      title, buttonLabels, defaultText);
    function promptCallback(result) {
        alert("You clicked on button number: " + result.buttonIndex +
         "\n You contact number is: " + result.input1);
    }
}
function getBeep() {
    navigator.notification.beep(2);
}
```

```
function getAlert() {
    var message = "Hi, I am Alert Dialog!!!";
    var title = "Alert Dialog Box";
    var buttonName = "OK";
    navigator.notification.alert(message, alertCallback, title, buttonName);

    function alertCallback() {
        alert("Alert is Dismissed!");
    }
}
function getConfirmation() {
    var message = "Are you a registered user?";
    var title = "Confirmation Dialog Box";
    var buttonLabels = "YES,NO";
    navigator.notification.confirm(message, confirmCallback, title, buttonLabels);

    function confirmCallback(buttonIndex) {
        alert("You clicked on button number: " + buttonIndex);
    }
}
function getPrompt() {
    var message = "Please provide your contact number?";
    var title = "Registration";
    var buttonLabels = "Cancel,Ok";
    var defaultText = ""
    navigator.notification.prompt(message, promptCallback,
      title, buttonLabels, defaultText);

    function promptCallback(result) {
        alert("You clicked on button number: " + result.buttonIndex +
         "\n You contact number is: " + result.input1);
    }
}

function getBeep() {
    navigator.notification.beep(2);
}
```

7. Build and emulate on avd.

OUTPUT:


Alert
Alert is Dismissed!
OK


Confirmation Dialog Box
Are you a registered user?
NO
YES


Alert
You clicked on button number: 1
OK


Registration
Please provide your contact number?
+919988776655
OK
CANCEL


Alert
You clicked on button number: 2
You contact number is:
+919988776655
OK

PRACTICAL No. 6 (A)

Installing and Using Globalization Plugin

This plugin is used for getting information about users' local language, date and time zone, currency, etc.

1. Create new cordova project with the help of following command:

   cordova create Global io.cordova.glob GlobalPluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create Global io.cordova.glob GlobalPluginApp
Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>
```

2. Go to Global folder and add android platform with the help of following command:

   cordova platform add android

3. Add Globalization Plugin with the help of following command:

   cordova plugin add cordova-plugin-globalization

```
E:\Cordova Projects AjaySIESCE\Global>cordova plugin add cordova-plugin-globalization
Installing "cordova-plugin-globalization" for android
Adding cordova-plugin-globalization to package.json
Saved plugin info for "cordova-plugin-globalization" to config.xml

E:\Cordova Projects AjaySIESCE\Global>
```

4. Now add 4 buttons in app with the help of following code:

```
<button id = "languageButton">LOCALE LANGUAGE</button><br />
<button id = "localeNameButton">LOCALE NAME</button><br />
<button id = "dateButton">DATE</button><br />
<button id = "currencyButton">CURRENCY</button>
```

```
<div class="app">
    <button id = "languageButton">LOCALE LANGUAGE</button><br />
    <button id = "localeNameButton">LOCALE NAME</button><br />
    <button id = "dateButton">DATE</button><br />
    <button id = "currencyButton">CURRENCY</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

```
document.getElementById("languageButton").addEventListener("click", getLanguage);
document.getElementById("localeNameButton").addEventListener("click", getLocaleName);
document.getElementById("dateButton").addEventListener("click", getDate);
document.getElementById("currencyButton").addEventListener("click", getCurrency);
```

```
onDeviceReady: function() {
    document.getElementById("languageButton").addEventListener("click", getLanguage);
    document.getElementById("localeNameButton").addEventListener("click", getLocaleName);
    document.getElementById("dateButton").addEventListener("click", getDate);
    document.getElementById("currencyButton").addEventListener("click", getCurrency);
    this.receivedEvent('deviceready');
},
```

6. And before the app.initialize(); add function as shown below:

```
function getLanguage() {
    navigator.globalization.getPreferredLanguage(onSuccess, onFailure);

    function onSuccess(language) {
        alert('language: ' + language.value);
    }

    function onFailure() {
        alert('Error getting language!!!');
    }
}

function getLocaleName() {
    navigator.globalization.getLocaleName(onSuccess, onFailure);

    function onSuccess(locale) {
        alert('locale: ' + locale.value);
    }

    function onFailure() {
        alert('Error getting locale!!!');
    }
}
```

```
function getDate() {
    var date = new Date();

    var options = {
        formatLength: 'short',
        selector: 'date and time'
    }
    navigator.globalization.dateToString(date, onSuccess, onFailure, options);

    function onSuccess(date) {
        alert('date: ' + date.value);
    }

    function onFailure() {
        alert('Error getting dateString!!!');
    }
}

function getCurrency() {
    var currencyCode = 'INR';
    navigator.globalization.getCurrencyPattern(currencyCode, onSuccess,
onFailure);

    function onSuccess(pattern) {
        alert('pattern: ' + pattern.pattern + '\n' +
         'code: ' + pattern.code + '\n' +
         'fraction: ' + pattern.fraction + '\n' +
         'rounding: ' + pattern.rounding + '\n' +
         'decimal: ' + pattern.decimal + '\n' +
         'grouping: ' + pattern.grouping);
    }

    function onFailure() {
        alert('Error getting pattern!!!');
    }
}
```

```
function getLanguage() {
    navigator.globalization.getPreferredLanguage(onSuccess, onFailure);

    function onSuccess(language) {
        alert('language: ' + language.value);
    }

    function onFailure() {
        alert('Error getting language!!!');
    }
}

function getLocaleName() {
    navigator.globalization.getLocaleName(onSuccess, onFailure);

    function onSuccess(locale) {
        alert('locale: ' + locale.value);
    }

    function onFailure() {
        alert('Error getting locale!!!');
    }
}

function getDate() {
    var date = new Date();

    var options = {
        formatLength: 'short',
        selector: 'date and time'
    }
    navigator.globalization.dateToString(date, onSuccess, onFailure, options);

    function onSuccess(date) {
        alert('date: ' + date.value);
    }

    function onFailure() {
        alert('Error getting dateString!!!');
    }
}

function getCurrency() {
    var currencyCode = 'INR';
    navigator.globalization.getCurrencyPattern(currencyCode, onSuccess, onFailure);

    function onSuccess(pattern) {
        alert('pattern: ' + pattern.pattern + '\n' +
         'code: ' + pattern.code + '\n' +
         'fraction: ' + pattern.fraction + '\n' +
         'rounding: ' + pattern.rounding + '\n' +
         'decimal: ' + pattern.decimal + '\n' +
         'grouping: ' + pattern.grouping);
    }

    function onFailure() {
        alert('Error getting pattern!!!');
    }
}
```

7. Build and emulate on avd.

OUTPUT:


Alert
language: en-US
OK


Alert
locale: en-US
OK


Alert
OK


Alert
code: INR
fraction: 2
rounding: 0
decimal: .
grouping: ,
OK

PRACTICAL No. 6 (B)

Installing and Using Network Information Plugin

1. Create new cordova project with the help of following command:

   cordova create Network io.cordova.netInfo NetInfoPluginApp

```
E:\Cordova Projects AjaySIESCE>cordova create Network io.cordova.netInfo NetInfoPluginApp
Creating a new cordova project.

E:\Cordova Projects AjaySIESCE>_
```

2. Go to Network folder and add android platform with the help of following command:

   cordova platform add android

3. Add Network Information Plugin with the help of following command:

   cordova plugin add cordova-plugin-network-information

```
E:\Cordova Projects AjaySIESCE\Network>cordova plugin add cordova-plugin-network-information
Installing "cordova-plugin-network-information" for android
Adding cordova-plugin-network-information to package.json
Saved plugin info for "cordova-plugin-network-information" to config.xml

E:\Cordova Projects AjaySIESCE\Network>_
```

4. Now add a button in app with the help of following code:

   <button id = "netInfoButton">GET NETWORK INFORMATION</button>

```
<div class="app">
    <button id = "netInfoButton">GET NETWORK INFORMATION</button>
    <div id="deviceready" class="blink">
```

5. Open index.js file and add Event Listener inside the onDeviceReady function as shown below:

document.getElementById("netInfoButton").addEventListener("click", netInfo);
document.addEventListener("offline", onOffline, false);
document.addEventListener("online", onOnline, false);

```
onDeviceReady: function() {
    document.getElementById("netInfoButton").addEventListener("click", netInfo);
    document.addEventListener("offline", onOffline, false);
    document.addEventListener("online", onOnline, false);
    this.receivedEvent('deviceready');
```

6. And before the app.initialize(); add function as shown below:

```
function netInfo() {
    var networkState = navigator.connection.type;
    var states = {};
    states[Connection.UNKNOWN] = 'Unknown connection';
    states[Connection.ETHERNET] = 'Ethernet connection';
    states[Connection.WIFI] = 'WiFi connection';
    states[Connection.CELL_2G] = 'Cell 2G connection';
    states[Connection.CELL_3G] = 'Cell 3G connection';
    states[Connection.CELL_4G] = 'Cell 4G connection';
    states[Connection.CELL] = 'Cell generic connection';
    states[Connection.NONE] = 'No network connection';
    alert('Connection type: ' + states[networkState]);
}
function onOffline() {
    alert('You are now offline!');
}

function onOnline() {
    alert('You are now online!');
}
```

```
function netInfo() {
    var networkState = navigator.connection.type;
    var states = {};
    states[Connection.UNKNOWN] = 'Unknown connection';
    states[Connection.ETHERNET] = 'Ethernet connection';
    states[Connection.WIFI] = 'WiFi connection';
    states[Connection.CELL_2G] = 'Cell 2G connection';
    states[Connection.CELL_3G] = 'Cell 3G connection';
    states[Connection.CELL_4G] = 'Cell 4G connection';
    states[Connection.CELL] = 'Cell generic connection';
    states[Connection.NONE] = 'No network connection';
    alert('Connection type: ' + states[networkState]);
}
function onOffline() {
    alert('You are now offline!');
}

function onOnline() {
    alert('You are now online!');
}
```

7. Build and emulate on avd.

OUTPUT:


Alert
You are now online!
OK


GET NETWORK INFORMATION


Alert
Connection type: Cell 4G connection
OK


Alert
You are now offline!
OK