**CORDOVA NOTES**

* **Cordova** is platform that is used for building mobile apps using HTML, CSS and JS. Cordova is like a container for connecting our web app with native mobile functionalities.
* Cordova offers a bridge for connection between web app and mobile device. **Cordova** we can make **hybrid mobile apps** that can use camera, geolocation, file system and other native mobile functions.
* Applications execute within wrappers targeted to each platform, and rely on standards-compliant API bindings to access each device's sensors, data, and network status.
* Features may include:
* **Command Line Interface (Cordova CLI)**
* **Cordova Core Components**
* **Cordova Plugins**
* **License**
* Adv: Cordova framework offers one **platform independent** for building hybrid mobile apps so we can develop one app that will be used on different mobile platform IOS, Android, Windows Phone, Amazon-fire os, blackberry, Firefox OS, Ubuntu and tizien.
  + ii. It is **faster to develop hybrid app** then native app so Cordova can **save lot of the development time.**
  + **iii.** **don't need to learn platform specific programming languages** as we use JS.
  + **iv.** huge amounts of community add ons that can be used with Cordova. Lots of libraries and frameworks are optimized for working with it.
* **6.**Limitations or disadv: **Hybrid apps are slower than native ones** so it is **not optimal to use Cordova for large apps** that require lots of data and functionality.
* ii. **Cross browser compatibility** can create lots of **issues**. Most of the time we are building apps for different platforms so the testing and optimizing can take lot of the time since we need to cover large number of devices and operating systems.
* iii. Some plugins have compatibility issues with different devices and platforms. There **are also some native APIs that aren't yet supported by Cordova**.
* Installation of cordova: Install GIT and set path, open cmd and type the following commands:
  + npm install -g cordova
* cordova -v
* **io.cordova.hellocordova** is the default reverse domain value. You should use your own domain value if possible. $ cordova platform add IOS
* cordova build android
* cordova emulate android
* cordova run android
* 7.storage API available for storing data on the client apps. This will help usage of the app when user is offline and it can also improve performance.
* 8.Creation of buttons :
* create four buttons in **index.html** file. The buttons will be located inside**div class = "app"** element.
* <button id = "setLocalStorage">SET LOCAL STORAGE</button>
* <button id = "showLocalStorage">SHOW LOCAL STORAGE</button>
* <button id = "removeProjectFromLocalStorage">REMOVE PROJECT</button>
* <button id = "getLocalStorageByKey">GET BY KEY</button>
* 9.Adding listeners: add event listeners inside **index.js** files. We will also assign **window.localStorage** to a**localStorage** variable that we will use later.
* document.getElementById("setLocalStorage").addEventListener("click", setLocalStorage);
* document.getElementById("showLocalStorage").addEventListener("click", showLocalStorage);
* document.getElementById("removeProjectFromLocalStorage").addEventListener
* ("click", removeProjectFromLocalStorage);
* document.getElementById("getLocalStorageByKey").addEventListener
* ("click", getLocalStorageByKey);
* var localStorage = window.localStorage;
* we need **to create functions that will be called when the buttons are tapped.** First function is used for adding data to local storage.
* function setLocalStorage() {
* localStorage.setItem("Name", "John");
* localStorage.setItem("Job", "Developer");
* localStorage.setItem("Project", "Cordova Project");
* }
* The next one will log data we added to console.
* function showLocalStorage() {
* console.log(localStorage.getItem("Name"));
* console.log(localStorage.getItem("Job"));
* console.log(localStorage.getItem("Project"));
* }
* create function that will delete the project from the local storage.
* function removeProjectFromLocalStorage() {
* localStorage.removeItem("Project");
* }
* we click **SHOW LOCAL STORAGE** button after we deleted the project, the output will show **null** value for the project field.
* get the local storage elements by using **key()** method which will take the index as an argument and return the element with corresponding index value.
* function getLocalStorageByKey() {
* console.log(localStorage.key(0));
* }
* The following table shows all local storage methods available.

|  |  |
| --- | --- |
| * **SN** | * **Methods & Details** |
| * 1 | * **setItem(key, value)** * Used for setting the item to local storage |
| * 2 | * **getItem(key)** * Used for getting the item from local storage. |
| * 3 | * **removeItem(key)** * Used for removing the item from local storage. |
| * 4 | * **key(index)** * Used for getting the item by using the **index** of the item in local storage. Items are sorted alphabetically |
| * 5 | * **length()** * Used for retreiving the number of items that exists in local storage. |
| * 6 | * **clear()** * Used for removing all the key/value pairs from local storage. |

* 9. Various events used in cordova:
* Cordova Events

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

* 10. right way of working with events is by using **addEventListener**.
* document.addEventListener("volumeupbutton", callbackFunction, false);
* function callbackFunction() {
* alert('Volume Up Button is pressed!')
* }
* 11.plugman is useful command line tool for installing and managing plugins. You should use **plugman** if you want your app to run on one specific platform. If you want to create **cross-platform** app you should use **cordova-cli** which will modify plugins for different platforms.
* 12. Cordova plugin is used for monitoring device's battery status. The plugin will monitor every change that happens to device's battery. camera plugin.
* 13.battery events

|  |  |
| --- | --- |
| * **Event** | * **Details** |
| * batterylow | * The event is triggered when battery charge percentage reached low value. This value varies on different devices. |
| * batterycritical | * The event is triggered when battery charge percentage reached critical value. This value varies on different devices. |

## 14. Contacts Plugin

## 15. Device Plugin

## Accelerometer Plugin/ device-motion.

## Device Orientation plugin

## This plugin will call platform native dialog UI element.-dialog

## This plugin is used for manipulating native file system on the user's device.-file plugin

* This plugin is used for uploading and downloading files.-**file transfer plugin**.
* **Geolocation** is used for getting info about device's latitude and longitude.
* This plugin is used for getting information about users locale language, date and time zone, currency etc.-**Globalization plugin**
* Cordova **media plugin** is used for recording and playing audio sounds in Cordova apps.
* **Media capture plugin** is used for accessing device's capture options.
* Network info plugin- plugin provides information about device's network.
* **Splash screen plugin**- plugin is used to display splash screen on application launch.
* **Vibration plugin**- This plugin is used for connecting to device's vibration functionality.
* **whitelist** plugin -This plugin allows us to implement whitelist policy for app's navigation. When we create new Cordova project, the **whitelist** plugin is installed and implemented by default. You can open the **config.xml** file to see **allow-intent** default settings provided by Cordova.
* Cordova plugin needs to wait until device is ready before it can be used. SPA design will improve loading speed and overall performance.
* Cordova is used for mobile world it is natural to use **touch start** and **touch end** events instead of **click** events. The click events have 300ms delay, so the clicks doesn't feel native. On the other hand, touch events aren't supported on every platform.
* Try to use hardware accelerated **CSS Transitions** instead of JavaScript animations since they will perform better on mobile devices.
* There are couple of ways to improve scrolling performance of the app. Our recommendation is to use native scrolling. When there are lots of items in the list, you should load them partially. Use loaders when necessary.
* use CSS image sprites whenever possible. Try to fit the images perfectly instead of scaling it.
* avoid shadows, gradients since they slow the rendering time of the page.
* Be sure to always test your app on as many devices and operating system versions as possible.