Vue.js with Salesforce.com Cheat Sheet

Commonly used markup syntax, basic app and component setup and helper object for remoteActions.

Libraries

```
// Dev Libraries

// https://unpkg.com/vue/dist/vue.js
// https://unpkg.com/vue-router/dist/vue-router.js

// Prod Libraries

// https://unpkg.com/vue/dist/vue.js
// https://unpkg.com/vue-router/dist/vue-router.js
```

Basic Syntax

```
// If/Else

// <div v-if="true">
//   Show if True
// </div>
// <div v-else>
//   Show if False
// </div>


// // Bind CSS

// v-bind:class="{ 'class-name': truthyValue, 'class-name-two': truthyValue }"

// v-bind:class="classObject"

// Bind Disabled

// v-bind:disabled="isDisabled"

// Repeat

// <div v-for="item in items">
//       {{ item.prop }}
// </div>

// Click handler

// v-on:click="method"
```

Vue.js with Salesforce.com Cheat Sheet

Commonly used markup syntax, basic app and component setup and helper object for remoteActions.

Basic App

```
// Basic App

let app = new Vue({
  el: '#cssSelector',
  //router: router,
  data: {
    values: 'to Start with'
  },
  created: function () {
    // lifecycle event, can be used to load data from Salesforce when app starts
  },
  methods: {
    method1: function () {
      // do something like call salesforce with user interaction
    }
  }
})
```

Basic Component

```
// Basic Component

let contactsVM = Vue.component('component-name', {
  template: `
    <div>{{ startData }}</div>
  `,
  components: {
    //child components
  },
  created: function () {
    // lifecycle event, can be used to load data from Salesforce when app starts
  },
  data: function () {
    // must be a function, return the object that has your data
    return { startData: 'test' }
  },
  methods: {
    method1: function () {
      // do something like call salesforce with user interaction
    }
  }
})
```

Vue.js with Salesforce.com Cheat Sheet

Commonly used markup syntax, basic app and component setup and helper object for remoteActions.

JavaScript Service with RemoteActions Helper

```
// Salesforce Basic Service Remote Actions

let sfService = (() => {

  function callRemote(methodName, params, resolve, reject) {
    console.log(...params)
    Visualforce.remoting.Manager.invokeAction(
      methodName,
      ...params,
      function (result, event) {
        console.log({ event })
        console.log({ result })

        if (event.status) {
          resolve(result)
        }
      },
      {
        //Options I am not setting
      }
    );
  }

  function javaScriptMethod() {
    return new Promise((resolve, reject) => {
      callRemote('ApexController.ApexMethod', [], resolve, reject)
    })
  }

  return {
    javaScriptMethod: javaScriptMethod
  }
})()
```