Draggable Components in Vue.js with Vue.Draggable

alligator.io/vuejs/drag-and-drop-vue-draggable

Sortable lists are on of the more frustrating parts of web development. The web wasn't originally created with drag-and-drop in mind, and even with the addition of native drag-and-drop support in HTML5, there's still quite a bit to hook up. With *Vue.Draggable* however, this becomes a cinch. It makes most of the decisions for you and leaves a simple, clean API for your benefit.

Installation

Install *Vue.Draggable* from NPM or Yarn:

```
# Yarn
$ yarn add vuedraggable

# NPM
$ npm install vuedraggable --save
```

Now, any component can import the *draggable* component from *vuedraggable*.

Usage

To make a list sortable, just wrap any element in a *draggable* component (usually with a v-for, but you can do individual elements as well.)

draggable can also take a *v-model* binding to update an array that contains the relevant data.

ExampleComponent.vue

```
<template>
  <div>
    <h2>Draggable Wrapper</h2>
    <draggable v-model="exampleList">
      <div v-for="text in exampleList" :key="text"></div>
    </draggable>
  </div>
</template>

<script>
import draggable from 'vuedraggable';

export default {
  components: {
    draggable
  },

  data() {
    return {
      exampleList: [
        'Item 1',
        'Item 2',
        'Item 3',
        'Item 4',
        'Item 5'
      ]
    }
  }
}
</script>
```

Voilà! The list is sortable! You can drag and drop list items around however you'd like and the changes will be reflected in the array.

Props & Events

Props:

- *value (v-model capable): Array* - The input array for the component. It's immutable, so it's recreated for every change.
- *list: Array* - An alternative to *value*, but updates the array without recreating it. (via. *Array.splice()*)
- *options: Object* - An object containing any of the valid options for

Sortable.

- *element: String* - The wrapper element used by the *draggable* component. Defaults to *div*.
- *clone: Function(VNode) -> VNode* - A method used to clone a VNode when the clone option is present. Fairly advanced. Mostly used for deep-cloning.
- *move: Function(evt) -> Boolean* - Called when an element is dragged. Returning false will cancel the operation.

Events: *start*, *add*, *remove*, *update*, *end*, *choose*, *sort*, *filter*, *clone*. Reference

Usage with Vuex

When using Vue.Draggable with Vuex, use a computed property for your array that retrieves the array from the store via a getter, and commits a mutation when set:

```
computed: {
  exampleList: {
    get() {
      return this.$store.state.exampleList
    },

    set(val) {
      this.$store.commit('setExampleList', val)
    }
  }
}
```

Additional

- Vue.Draggable works fine with transitions, just make sure that the *draggable* element is directly wrapping the *transition* or *transition-group* element. There can be nothing else in-between.
- If your list has 0 elements in it by default, make sure the element *draggable* creates is styled with enough height to be a valid drop target.

That’s all there is to it!

Now, I added home-spun drag-and-drop functionality to a couple personal projects. I'm going to go replace those shoddy implementations with *Vue.Draggable* now. Adiós!