Hello World

NEXT: VUE CLI

Just for the fun's sake, let's see how easy it is to get started with Vue (with no drama).

No drama involves:

- No configuration
- No bundling technique
- No CSS
- No third party tool
- Just Vue and HTML

To get started we'll do the following:

1. Create an `index.html` on your machine in any directory of choice. The desktop is fine.
2. Add Vue from CDN to the index file:

```HTML
<html>
    <head></head>
    <body>

        <!-- Add Vue -->
        <script src="https://unpkg.com/vue/dist/vue.js"></script>
    </body>
</html>
```

1. Create a JavaScript Vue instance

```HTML
<html>
    <body>

        <script src="https://unpkg.com/vue/dist/vue.js"></script>
        <script>
           /* Create a Vue instance*/
            new Vue({
                template: '#app',
                data: function() {
                    return {
                        name: ''
                    }
                }
            }).$mount('#app');
        </script>
    </body>
</html>
```

The Vue instance must take an object that describes the Vue app. The `template` is where in our DOM Vue is allowed to perform magic.

In this case, we chose the section with a parent `div` of `app` ID.

Vue works with the Model–view–viewmodel (MV-VM) pattern so you can carry out two-way binding. The `data` property is a function (not necessarily) that returns an object. The returned object properties are used as the model and can be bound to the template (view).

1. Bind the model to view

```HTML
<html>
    <body>

        <!-- Template -->
        <div id="app">
            <div>
                <input type="text" v-model="name" placeholder="Your name?" />
                <h5>Hello, {{name}}</h5>
            </div>
        </div>

        <script src="https://unpkg.com/vue/dist/vue.js"></script>
        <script>
           /* Create a Vue instance*/
            new Vue({
                template: '#app',
                data: function() {
                    return {
                        name: ''
                    }
                }
            }).$mount('#app');
        </script>
    </body>
</html>
```

Within the Vue template (with `app` ID), we create an input and bind the name property to it using `v-model`:

```HTML
<input type="text" v-model="name" placeholder="Your name?" />
```

Then we greet the name using another type of binding called interpolation:

```HTML
<h5>Hello, {{name}}</h5>
```

The `{{}}` will not be handled as HTML but will resolve the value of data bound to it.

Note: Notice that we also have a child `div` within the main `<div id="app">`. It's a necessity for Vue to have only one root element.

When you start typing your name in the input box, expect to see the name update after the `Hello,` text.

See Codepen demo below:

See the Pen Vue.js Template by Chris Nwamba (@codebeast) on CodePen.

Getting Started
1 Introduction Free
2 Tools Free
3 Our Task Free
4 Hello World Free
Vue Basics
5 Vue CLI Free
6 Components and the Vue Instance Free
7 Template Syntax Free
8 Conditional & List Rendering Free
9 Styles & Classes Free
Routing
10 Single Page Apps Free
11 Creating Routes Free
12 Route Outlets & Links Free
13 Nested & Dynamic Routes Free
Forms
14 Two Way Binding (Reactivity) Free
15 Form Validation Free
16 Handling Form Submissions Free
API Backend
17 Prelude Free
18 Setup Free
19 Provisioning a MongoDB Database Free
20 Enabling CORS Free
21 Schemas & Models Free
22 Routes and Controllers Free
23 Testing with POSTman Free
Vuex
24 The Current Problem Free
25 State Management Free
26 Getters Free
27 Mutations Free
28 Actions Free
Using Store In Components
29 State and Actions on Components Free
30 LAB: Product List Free
31 LAB: Product Details Free
32 LAB: Admin Features Free
33 LAB: Spinner, Cart & Strore Subscriptio