# VUE.JS

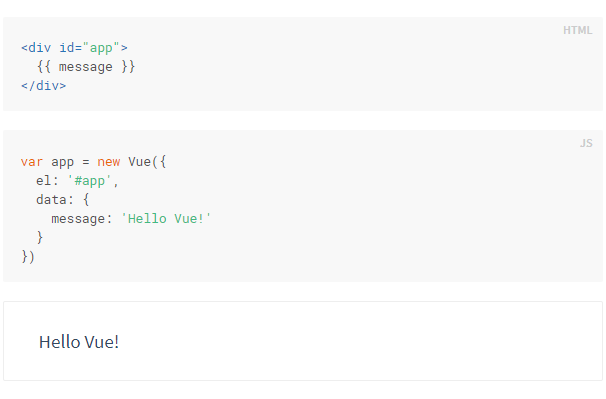
## Introduction

### 1.1 What is Vue.js ?

Vue.js is a new framework that is specialised in building user interfaces. Vue.js is built in such a way that it’s design is made from the start and thereafter progressively adopted. It’s primary and basic library is focused only on the visualisation layer, being easy to retrieve and integrate with different other existing projects and libraries. But then again, Vue.js is also in perfect condition and capability to propel single-page applications when used in combination with supportive libraries and modern tools.

### 1.2 Declarative rendering

At the core of Vue.js there is a system that enables us to declaratively render data to the HTML DOM (Document Object Model) using simple template syntax:



### 1.3 Relation to Custom Elements

You may have noticed that Vue components are very similar to Custom Elements, which are part of the Web Components Spec. That’s because Vue’s component syntax is loosely modeled after the spec. For example, Vue components implement the Slot API and the is special attribute. However, there are a few key differences:

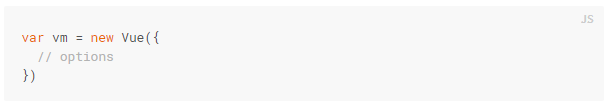
The Web Components Spec has been finalized, but is not natively implemented in every browser. Safari 10.1+, Chrome 54+ and Firefox 63+ natively support web components. In comparison, Vue components don’t require any polyfills and work consistently in all supported browsers (IE9 and above). When needed, Vue components can also be wrapped inside a native custom element.

Vue components provide important features that are not available in plain custom elements, most notably cross-component data flow, custom event communication and build tool integrations.

## The Vue instance

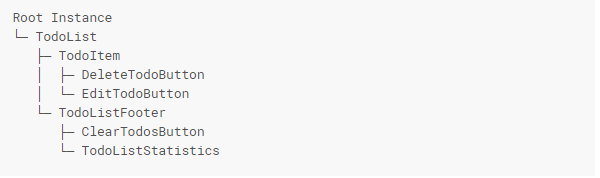
### 2.1 Creating a Vue Instance

In every Vue application, the starting point is by creating a new Vue instance with the Vue function:



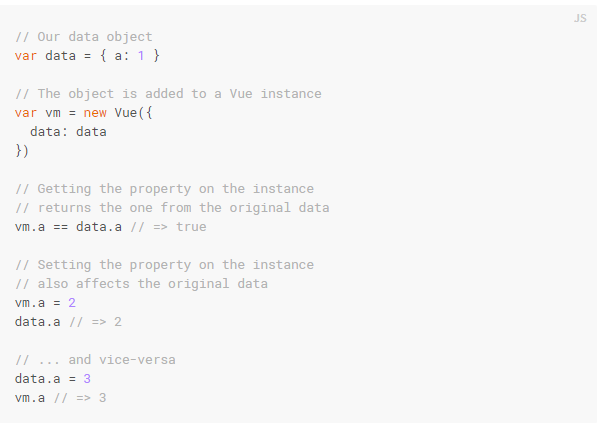
Although not strictly associated with the MVVM pattern, Vue’s design was partly inspired by it. As a convention, we often use the variable vm (short for ViewModel) to refer to our Vue instance.

When you create a Vue instance, you pass in an options object. The majority of this guide describes how you can use these options to create your desired behavior. For reference, you can also browse the full list of options in the API reference.

A Vue application consists of a root Vue instance created with new Vue, optionally organized into a tree of nested, reusable components. For example, a todo app’s component tree might look like this:

### 2.2 Data and Methods

When a Vue instance is created, it adds all the properties found in its data object to Vue’s reactivity system. When the values of those properties change, the view will “react”, updating to match the new values.



When this data changes, the view will re-render. It should be noted that properties in data are only reactive if they existed when the instance was created.

## Brief Conclusion

Vue.js is aframework that lets the user make both the visual part of a website, yet also being able to create big and well structured sinle-page applications. It’s a well rounded framework that comes in handy more than you’d think. Now that I’ve finished this short presentation I am actually interested in finding out more about this and use it in different scenarios.