

These slides are part of the downloadable resources of *The Complete Guide to Django REST Framework and Vue JS* Udemy course.

Let's keep in touch:

- ▶ YouTube Channel: https://www.youtube.com/channel/UCxPWtz5N--X3IyYJ13Zr99A?sub_confirmation=1
- ▶ Twitter: <https://www.twitter.com/pymike00>
- ▶ GitHub: <https://github.com/pymike00>



Vue JS

Objectives



Vue JS - Objectives

Welcome to this new section about **Vue JS**!

The goal for this section is to provide you a complete introduction to the framework.

You will learn the framework's philosophy and all its most important features, getting ready to learn and implement more advanced concepts in the final project of the course, where we are going to build a full fledged Single Page Application with DRF and Vue CLI.

Vue JS

Lessons

- Introduction to Vue JS
 - First Vue Instance
 - Events and Methods
 - Conditional Rendering
 - Class & Style Binding
 - List Rendering with v-for
 - Computed Properties
 - Forms and User Input
 - Components and Props
 - How to use \$emit
 - Competency Assessment and Project Solution
-

Vue JS - Reference Links

<https://vuejs.org/>

Introduction to Vue JS

What is Vue JS ?

Vue JS - Introduction

Vue JS is an open source *JavaScript framework* for building user interfaces, reactive components and Single Page Applications (SPA).

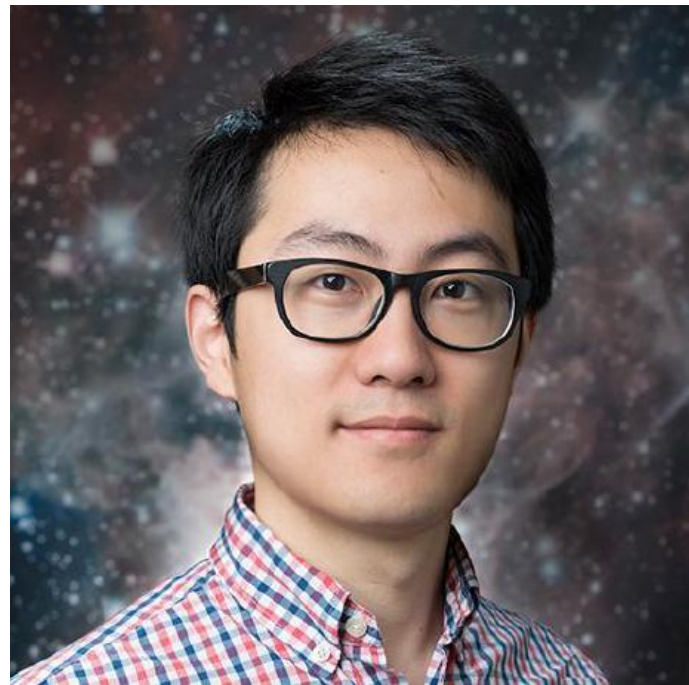
It is called a *progressive framework* because of its specific design, that makes it easy to create both simple and complicated projects.

Vue JS - Introduction

It was released in 2014 by Evan You, a former Google employee, and reached version 1.0 only by the end of 2015, with the 2.0 released at the end of 2016 that made its popularity grow considerably.

Despite its young age, the framework has in a short time been able to gain broad consensus and wide adoption by companies and developers all over the world, despite being developed by an independent team (unlike React and Angular!)

It is currently the most starred JavaScript framework on GitHub.

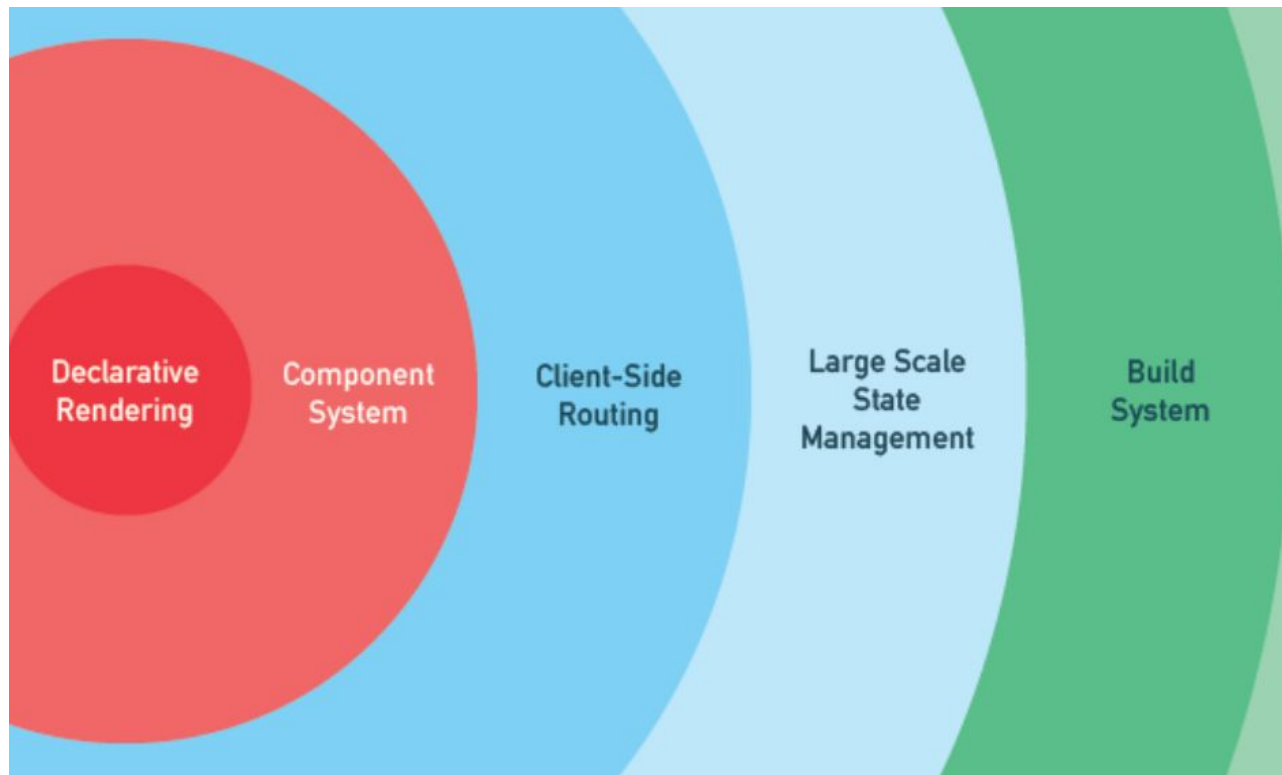


Vue JS - Introduction

We can appreciate Vue's great power combined with its ease of use, community support, excellent documentation and the progressive approach to development, made possible by a large number of official support libraries.



A Progressive Framework: Application Complexity vs Framework Complexity



Vue JS - Introduction

The Vue ecosystem is formed by a series of complementary official projects, which allow us to extend its capabilities far beyond the "core view layer" according to our needs:

- Vue-Router (*Single Page Application Routing*)
- Vuex (*Large Scale State Management*)
- Vue-CLI (*Project Scaffolding - Rapid Development Toolkit*)
- Vue-Loader (Single File Component (*.vue file) loader for webpack)
- Vue-DevTools (*Browser DevTool Extension*)
- Vue-Server-Renderer
- Vue-Class-Component
- Vue RXs

Vue JS - Introduction

Now that we have understood the Framework's philosophy and approach to development it is time to start writing practical examples.

Let's Code!

Introduction to Vue.js - Reference Links

<https://vuejs.org/>

<https://github.com/vuejs>

<https://twitter.com/youyuxi>

First Vue Instance

Vue JS - First Vue Instance

In this lesson we are going to create our first Vue Instance!

We are going to use Vue by including it in our web pages with a script tag, getting the code via Content Delivery Network.

We are also going to talk about *reactivity*, a very important feature that will allow our HTML content to dynamically change thanks to Vue.

Let's get started!

First Vue Instance - Reference Links

<https://vuejs.org/v2/guide/instance.html>

<https://vuejs.org/v2/api/#Options-Data>

<https://vuejs.org/v2/guide/syntax.html#Interpolations>

<https://vuejs.org/v2/guide/syntax.html#Directives>



Events and Methods



Vue JS - Events and Methods

In this lesson we are going to learn how to let our Vue application listen for DOM events that we can use to trigger different methods defined in the application itself.

While doing so we are going to introduce the **v-on** directive!

Let's get started!

Events and Methods - Reference Links

<https://vuejs.org/v2/api/#v-on>

Conditional Rendering

Vue JS - Conditional Rendering

In this lesson we are going to learn how to programmatically show elements in our web pages when certain conditions are met.

We are going to introduce the following directives: **v-if**, **v-else**, **v-else-if**, **v-show**.

Let's get started!

Conditional Rendering - Reference Links

<https://vuejs.org/v2/api/#v-for>



Class & Style Binding



Vue JS - Class & Style Binding

As we all know, one of the most common tasks when creating a graphical interface for a website, is adding style to the HTML elements using CSS.

Often times the style of an element can vary dynamically, for example when the mouse hovers a certain area of the page.

In this lesson we are going to learn how to use the v-bind directive to dynamically style the HTML elements of our web pages.

Let's get started!

Class & Style Binding - Reference Links

<https://vuejs.org/v2/guide/class-and-style.html>

List Rendering with v-for

Vue JS - List Rendering with v-for

In this lesson we are going to learn how to use the **v-for** directive to render lists of elements in our web pages.

Let's get started!

List Rendering with v-for - Reference Links

<https://vuejs.org/v2/api/#v-for>

Computed Properties

Vue JS - Computed Properties

In the First Vue Instance lesson we have seen that it is possible to define and use expressions directly within our templates:

```
<div id="app">
```

```
  <h1>{{ (33 / 11) * 5 }}</h1>
```

```
</div>
```


Vue JS - Computed Properties

However it is easy to understand how using complex expressions can quickly make our code difficult to read and maintain, increasing the risk of bugs!

```
<div id="app">
```

```
  <p>{{ message.split("").reverse("").join("") }}</p>
```

```
</div>
```

Vue JS - Computed Properties

That's why for any complex logic Vue offers us the **Computed Properties**.

Similar to methods, that we have already learned to use, they differ from them because their results are cached until the values they use change.

Vue JS - Computed Properties

Sometimes called also computed values, we can think of these as an extension of our data model, useful to show values resulting from processing.

It is good practice to avoid changing the values of the data model through the computed properties, keeping them "pure".

Let's make some examples!

Computed Properties - Reference Links

<https://vuejs.org/v2/guide/computed.html#Computed-Properties>

<https://css-tricks.com/methods-computed-and-watchers-in-vue-js/>

Forms and User Input

Vue JS - Forms and User Input

In this lesson we are going to learn how to handle user input with our application, and how to properly use forms in Vue.

In doing so we are also going to introduce a new directive: ***v-model***

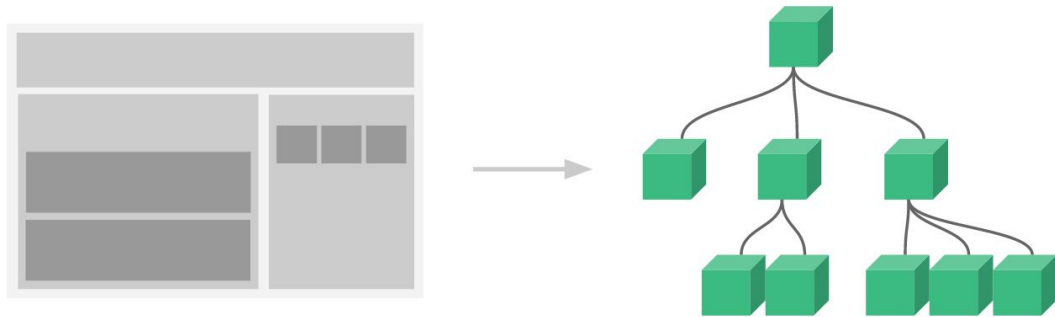
Let's get started!

Components and Props

Vue JS - Components and Props

In this lesson we are going to learn how to use **components** in Vue JS.

Components allow us to divide our apps into **reusable blocks of code**, and within the components we can write both the JavaScript logic and the HTML code.



Vue JS - Components and Props

Components allow us to keep our code **ordered** and **maintainable**, thus giving us the chance to create large scale projects easily.

We will also talk about **Props**, a special type of attribute that we use to pass data between parent and child components. When a value is passed from one component to another via Prop, it becomes a property of the component instance to which it has been passed.

Let's code some practical examples!

Components and Props - Reference Links


<https://medium.com/js-dojo/7-ways-to-define-a-component-template-in-vuejs-c04e0c72900d>

How to use \$emit

Vue JS - \$emit


In this lesson we are going to learn how to make a child component communicate with its parent using **\$emit**, giving them the ability to call methods and pass values.

Let's get started!



Vue JS

Competency Assessment



Vue JS - Competency Assessment

This section's project consists in creating a To-Do Application with Vue!

Our application's users will be able to add new elements to a list of tasks that will be shown on the app's page.

Users must also be allowed to remove tasks from the list once completed.

Let's have a look at the project!



Vue JS

Competency Assessment - Project Solution



Vue JS - Project Solution

You can download the solution for this challenge from this section's downloadable resources.