

## Vue Uses a “Declarative Approach”

You define the **result**, NOT the way to get there!

In “vanilla” JavaScript, you have to define all steps and control the entire way to get to a result



With frameworks like Vue, you define a template (with dynamic bindings and directives) and Vue takes care about the way to get the desired output onto the screen

# Understanding the Vue Options API

API: Application Programming Interface  
(= there is a clearly defined way of using it)

Vue Options API

You set certain options (= you provide a certain configuration) that tells Vue how to interact with the DOM it's controlling

Options for data binding, methods & more

# The Vue Instance

You “configure” Vue via the Vue app object

## Vue App Properties

**data**

**methods**

## What They Do

Set data that can be **used in the controlled HTML elements**. If **data changes**, Vue updates the DOM.

Define methods (functions) that can be **triggered from inside the controlled HTML elements** (and from inside the Vue instance).

# Methods vs Computed vs Watch

## Methods

Use with event binding OR data binding

Data binding: Method is executed for every “re-render” cycle of the component

Use for events or data that really needs to be re-evaluated all the time

## Computed

Use with data binding

Computed properties are only re-evaluated if one of their “used values” changed

Use for data that depends on other data

## Watch

Not used directly in template

Allows you to run any code in reaction to some changed data (e.g. send Http request etc.)

Use for any non-data update you want to make

## Summary

### DOM & Templates

Vue can be used to define the goal instead of the steps (→ **declarative** approach)

**Connect** Vue to HTML via “**mount**”:  
Vue **then renders the real DOM** based on the connected template

### Reactivity

Vue updates the real DOM for you when bound data changes

**Computed properties** and **watchers** allow you to react to data changes

### Data & Event Bindings

You can **bind data** via interpolation (`{{ }}`) or the **v-bind** (“`:`”) directive

You **listen for events** via **v-on** (“`@`”)

### Styling

Dynamic CSS class and inline style bindings are supported by Vue

Vue offers multiple **special syntaxes** (object-based, array-based) for efficient bindings