

Modul 8

# VueJS Basic



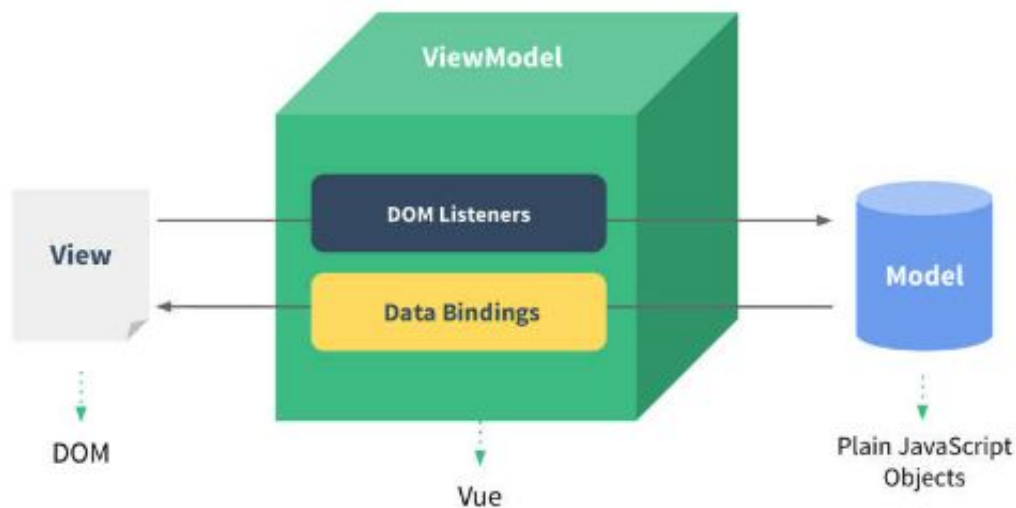
PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
UNIVERSITAS ATMA JAYA YOGYAKARTA  
SEMESTER GASAL 2020/2021

## Yang harus disiapkan

1. Text editor (disarankan VSCode)
2. Web browser

## Apa itu VueJS?

Vue JS adalah framework javascript untuk membangun antarmuka web yang interaktif. VueJS menggunakan konsep **MVVM** (Model-View-ViewModel) yang terhubung dengan **DOM** (Document Object Model) website sehingga VueJS dapat mengakses dan memanipulasi data dan tampilan/antarmuka website.



## MVVM

MVVM (Model-View-ViewModel) merupakan sebuah fasilitas yang menjembatani antara tampilan dan juga proses.

- **Model** : objek yang mewakili data
- **View** : objek yang mewakili tampilan yang dapat dilihat oleh pengguna
- **ViewModel** : objek yang menyimpan abstraksi dari tampilan/view dan menangani semua proses yang berhubungan dengan pengolahan data.

## Persiapan Guided

Pertama, siapkan satu folder dan buat sebuah file bernama GD1.html lalu masukan template html 5 biasa.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>
<body>

</body>
</html>
```

Setelah itu kita akan mengimport script VueJS versi 2. Masuk ke website dokumentasi VueJS 2 dan buka bagian instalasi atau akses melalui link berikut

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

Lalu klik tombol berikut untuk mengunduh file javascript VueJS



Simpan file vue.js **di folder yang sama** dengan GD1.html lalu sertakan file js yang sudah diunduh di dalam tag `<script>` seperti berikut :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <script src="vue.js"></script>
</head>
<body>

</body>
</html>
```

Untuk menggunakan Vue JS, kita akan menulis code kita di dalam **Vue instance**.

Buatlah Vue instance di dalam tag `<script>` baru :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <script src="vue.js"></script>
</head>
<body>

</body>
<script>
  new Vue({})
</script>
</html>
```

Code akan kita tulis di dalam `{}`.

Kita juga harus menghubungkan Vue instance dengan elemen HTML dengan menentukan area yang dapat dimanipulasi Vue.



```
<body>
  <div id="app"></div>
</body>
<script>
  new Vue({
    el: '#app'
  })
</script>
```

Pada Vue instance, ada key `el` dan value-nya adalah `'#app'`. Lalu ada `<div>` baru dengan id `'app'`. Code tersebut membuat `<div>` yang id-nya `'app'` terhubung dengan Vue instance.

Dengan ini, kita bisa memakai Vue JS di dalam `<div>` itu.

Vue JS memiliki banyak fitur. Namun, kali ini kita akan mengenal fitur dasar saja.

Beberapa fitur Vue JS yang akan kita pelajari pada modul ini :

#### **Guided 1 -----**

1. Data & methods
2. Data binding
3. Two way data binding

#### **Guided 2 -----**

4. Conditional rendering
5. List rendering
6. Event handling

## Guided 1

Pada Guided 1, kita akan membuat website sederhana yang akan terlihat seperti ini



Pertama, kita akan menampilkan kata-kata “Hello world!”. Gunakan code dari bagian Persiapan Guided lalu ubah dan tambahkan code yang digarisbawahi.

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Guided 1</title>
  <script src="vue.js"></script>
</head>

<body>
  <div id="app">
    <h1>{{ judul }}</h1> <!-- Pemanggilan judul menggunakan '{{}}' -->
  </div>
</body>
<script>
  new Vue({
    el: "#app",
    data: {
      <u>judul: "Hello world!", // Deklarasi judul</u>
    }
  });
</script>

</html>
```

**data** adalah bagian yang menampung data-data yang kita butuhkan. Saat ada data berubah, Vue akan bereaksi dan merubah view/ tampilan web. Jenis/tipe data yang bisa disimpan adalah tipe data primitif maupun non-primitif dari JS.

Setelah itu, kita akan membahas mengenai **methods**. **methods** adalah tempat dimana kita menuliskan fungsi-fungsi yang kita butuhkan. Kita dapat memanggil method di dalam `{{ }}` yang tadi kita gunakan. Kita juga dapat memberi parameter pada fungsi. Sebagai contoh, tambahkan code berikut

```
<body>
  <div id="app">
    <h1>{{ judul }}</h1>

    <p>{{ belajar('Vue JS') }}</p> <!-- memanggil method belajar() -->
  </div>
</body>
<script>
  new Vue({
    el: "#app",
    data: {
      judul: "Hello world!",
    },
    methods: {
      belajar(pelajaran) {
        return `Aku sedang belajar ${pelajaran}`;
      },
    },
  });
</script>
```

Save dan lihat hasilnya di browser. Tampilan akan berubah menjadi seperti ini

# Hello world!

Aku sedang belajar Vue JS

Selanjutnya, kita akan membuat bagian link.

# Hello world!

Aku sedang belajar Vue JS

Klik link untuk mengakses [Dokumentasi Vue JS](#)

Untuk mengarahkan link, tentu tag yang kita gunakan adalah `<a href="">`. Namun, kita ingin link yang ada pada href bersifat dinamis/sesuai dengan link yang ada di `data`. Maka dari itu, kita harus menggunakan sebuah **directive**. Tambahkan code berikut

```
<p>{{ belajar('Vue JS') }}</p>

<p>
  Klik link untuk mengakses
  <a v-bind:href="link"> <!-- bisa v-bind:href atau :href -->
    Dokumentasi Vue JS
  </a>
</p>
</div>
</body>
<script>
  new Vue({
    el: "#app",
    data: {
      judul: "Hello world!",
      link: "https://vuejs.org/v2/guide/", // link untuk href
    },
  })
```

Directive adalah perintah khusus untuk melakukan sesuatu. Directive `v-bind` berguna untuk melakukan **data binding** atribut href dengan data `link` yang kita miliki.

Kenapa harus pakai `v-bind`? Kenapa nggak pakai `{{}}` ?

Karena untuk bagian atribut, tidak bisa pakai `{{}}`



Terakhir, kita akan membuat inputan yang bisa menerima teks dan menampilkan teks yang ditulis di inputan tersebut secara sinkron.

# Hello world!

Aku sedang belajar Vue JS

Klik link untuk mengakses [Dokumentasi Vue JS](#)

  
**xixixi :^)**

Sekarang buatlah inputan dengan menambah code berikut :

```
<p>
  Klik link untuk mengakses
  <a :href="link">Dokumentasi Vue JS</a>
</p>

<div>
  <input v-model="teks" placeholder="tulis sesuatu..." />
  <h2>{{ teks }}</h2>
</div>
</div>
</body>
```

Lalu tambahkan teks ke dalam data

```
<script>
  new Vue({
    el: "#app",
    data: {
      judul: "Hello world!", // Deklarasi judul
      link: "https://vuejs.org/v2/guide/",
      teks: "",
    },
  },
```

Yang baru saja kita lakukan disebut **two-way data binding** atau input binding.

Dilakukan dengan cara menuliskan directive `v-model` ke tag `<input>`.

v-model membuat value input sinkron dengan teks yang ada di data. Sehingga jika kita merubah inputan, value dari teks juga akan ikut berubah. Sementara itu, <h2> digunakan untuk menampilkan value property teks agar perubahan dapat terlihat.

**Code lengkap GD 1 :**

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Guided 1</title>
    <script src="vue.js"></script>
  </head>

  <body>
    <div id="app">
      <h1>{{ judul }}</h1>

      <p>{{ belajar('Vue JS') }}</p>

      <p>
        Klik link untuk mengakses
        <a :href="link">Dokumentasi Vue JS</a>
      </p>

      <div>
        <input v-model="teks" placeholder="tulis sesuatu..." />
        <h2>{{ teks }}</h2>
      </div>
    </div>
  </body>
</html>

<script>
  new Vue({
    el: "#app",
    data: {
      judul: "Hello world!",
      link: "https://vuejs.org/v2/guide/",
      teks: "",
    },
    methods: {
      belajar(pelajaran) {
        return `Aku sedang belajar ${pelajaran}`;
      },
    },
  });
</script>
</html>
```

## Guided 2

Pada Guided 2, kita akan membuat website perkenalan diri

### Halo, aku Karen

umurku 20 tahun dan aku bekerja sebagai seorang programmer.

Aku masih lajang hehe

Kurangi umurku

Tambah umurku

Kemampuanku adalah :

- mencuci piring
- coding

Pertama-tama, siapkan template dari Persiapan Guided.

Ubah title menjadi **Guided 2** dan tambah beberapa data pada Vue instance.

```
<script>
  new Vue({
    el: "#app",
    data: {
      wanita: {
        nama: 'Karen',
        umur: 20,
        pekerjaan: 'programmer',
        skills: [
          'mencuci piring',
          'coding'
        ],
        status: 'belum menikah'
      }
    },
    methods: {},
  });
</script>
```

Perhatikan objek wanita yang baru saja ditambahkan. Objek tersebut berisi data-data yang nantinya akan kita tampilkan di web. Kita akan mulai dari bagian berikut ini

**Halo, aku Karen**

umurku 20 tahun dan aku bekerja sebagai seorang programmer.

Tambahkan beberapa code di bawah ini

```
<body>
  <div id="app">
    <h2>Halo, aku {{ wanita.nama }}</h2>
    <p>
      umurku {{ wanita.umur }} tahun dan aku bekerja
      sebagai seorang {{ wanita.pekerjaan }}.
    </p>
  </div>
</body>
```

Sama seperti Guided 1, untuk menampilkan data, kita dapat menggunakan `{{}}`.

Sekarang kita akan menambahkan bagian selanjutnya yaitu status. Jika diperhatikan, status yang ditampilkan di web bukanlah value dari properti `status` pada objek wanita. Karena kita ingin menampilkan template pesan sesuai dengan `status`.

**Halo, aku Karen**

umurku 20 tahun dan aku bekerja sebagai seorang programmer.

Aku masih lajang hehe

Maka, disini kita akan menggunakan **conditional rendering**. Conditional rendering adalah fitur Vue JS yang dapat membantu kita mengubah tampilan sesuai dengan kondisi yang kita tetapkan.

Ada 2 macam directive conditional rendering :

→ `v-show`

akan menampilkan area tertentu jika value bernilai **truthy** dan sebaliknya.

→ `v-if`, `v-else-if`, `v-else`

akan menampilkan area tertentu jika **kondisi** yang ditetapkan **terpenuhi** dan sebaliknya.

Kita akan mencoba menggunakan `v-if` terlebih dahulu. Tambahkan code berikut

```
<p>
  umurku {{ wanita.umur }} tahun dan aku bekerja
  sebagai seorang {{ wanita.pekerjaan }}.
  <br>
  Aku <span v-if="wanita.status == 'menikah'">
    Sudah menikah.
  </span>
  <span v-else-if="wanita.status == 'belum menikah'">
    masih lajang hehe
  </span>
</p>
```

`<span>` berguna sebagai pembungkus tampilan yang akan di-render jika kondisi `v-if` atau `v-else-if` terpenuhi.

Jika `status` bernilai 'menikah', maka teks yang akan di-render adalah "Sudah menikah".

Jika `status` bernilai 'belum menikah', teks yang muncul adalah "masih lajang hehe".

Sebagai percobaan, buatlah `v-else` yang akan menampilkan "dalam hubungan yang rumit" dan ubah value `status` menjadi string yang bukan "menikah" atau "belum menikah". Jika `v-else` terpilih, maka code kamu sudah benar!

Selanjutnya, kita akan membuat daftar kemampuan. Kemampuan diambil dari array `skills` objek wanita. Kita akan menggunakan `v-show` untuk menampilkan list dan kita juga akan menggunakan `list-rendering` untuk menampilkan semua skill.

## Halo, aku Karen

umurku 20 tahun dan aku bekerja sebagai seorang programmer.

Aku masih lajang hehe

Kemampuanku adalah :

- mencuci piring
- coding

Tambahkan code berikut

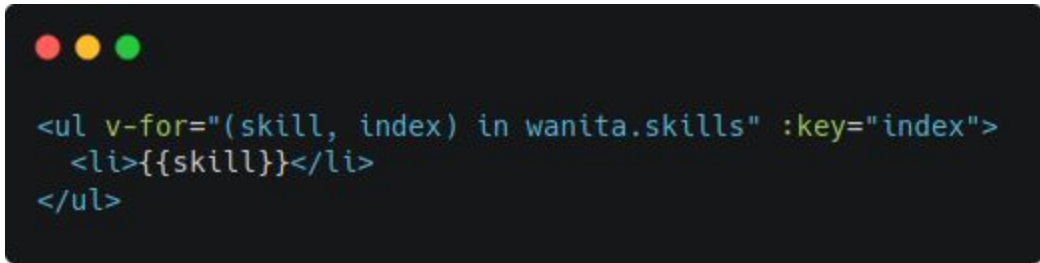
```
        masih lajang hehe
      </span>
    </p>

    <p v-show="wanita.skills">Kemampuanku adalah :</p>
    <ul v-for="(skill, index) in wanita.skills" :key="index">
      <li>{{skill}}</li>
    </ul>
  </div>
</body>
<script>
  new Vue({
    el: "#app",
    data: {
      wanita: {
        nama: "Karen",
        umur: 20,
        pekerjaan: "programmer",
        skills: ["mencuci piring", "coding"],
        status: "belum menikah",
      },
    },
  },
```

Kita menaruh `v-show` pada tag `<p>` yang artinya jika kondisi `v-show` bernilai `truthy`, isi dari tag `<p>` akan muncul. Namun jika `falsy`, tidak akan muncul. Save code dan cek browser. Jika muncul, code sudah benar!

Kita akan mencoba membuat `skills` bernilai `falsy`. Ganti value `skills` menjadi `null` dan save lagi. `null` bernilai `falsy`. Maka, seluruh isi dari tag `<p>` seharusnya tidak muncul. Jika sudah berhasil tidak muncul, kembalikan array seperti semula.

Sekarang kita akan membahas *list rendering*. *List rendering* membantu kita untuk menampilkan data di dalam array yang kita miliki. Cara membuat list rendering adalah menggunakan directive `v-for`. Perhatikan code `v-for` tadi

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. The code is written in a light blue/cyan monospace font. It shows a Vue.js `v-for` directive used to iterate over an array named `wanita.skills`. The code is: 

```
<ul v-for="(skill, index) in wanita.skills" :key="index">
  <li>{{skill}}</li>
</ul>
```

`wanita.skills` merupakan array `skills` yang ada di objek `wanita`.

`skill` adalah alias untuk setiap 1 elemen yang ada di array.

`index` adalah argumen opsional yang bisa digunakan untuk mengakses indeks array.

`:key` adalah atribut yang digunakan oleh DOM Vue JS untuk membedakan tiap data yang dirender pada iterasi `v-for`. Supaya jika ada elemen array yang berubah, Vue JS masih bisa membedakan tiap elemen tersebut.

Jika kita tidak ingin memakai `index`, penulisan code dapat diubah menjadi

```
<ul v-for="skill in wanita.skills" :key="skill">
  <li>{{skill}}</li>
</ul>
```

Terakhir, kita akan membuat button dengan event *handling*. Event *handling* akan menangani *click* event saat button diklik sehingga kita dapat menambahkan umur.

## Halo, aku Karen

umurku 20 tahun dan aku bekerja sebagai seorang programmer.

Aku masih lajang hehe

Kurangi umurku

Tambah umurku

Kemampuanku adalah :

- mencuci piring
- coding

Tambahkan code berikut

```
<div>
  <button @click="kurangiUmur">Kurangi umurku</button>
  <button @click="tambahUmur">Tambah umurku</button>
</div>
```



```
methods: {
  tambahUmur() {
    this.wanita.umur++;
  },
  kurangiUmur() {
    this.wanita.umur--;
  },
},
```

@click sebenarnya adalah `v-on:click`. Penulisan @click merupakan penulisan ringkasnya. Kita bisa memanggil fungsi tanpa tanda kurung () jika tidak ada parameter. Pada saat button diklik, fungsi yang ada pada event handling akan dijalankan.

## Pengumpulan Guided

Masukan semua file (GD1.html, GD2.html, dan vue.js) dalam satu folder

Buat repo (repository) github baru bernama VueBasic\_XXXX

XXXX = 4 digit terakhir NPM

Sambungkan repo github kalian dengan folder lokal kalian dan masukan pada branch main (Lihat tutorial Git dan Github)

Kumpulkan link repo github ke situs kuliah