Фреймворк Vue.js

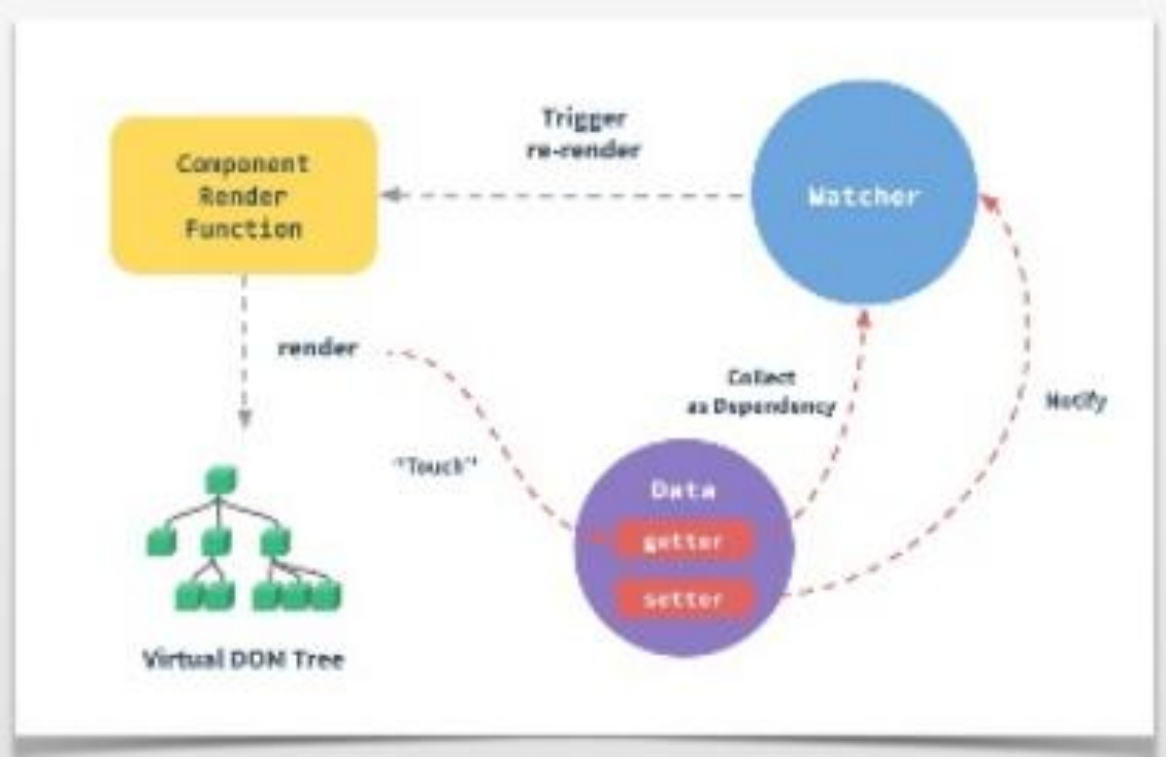
Самостійна робота з курсу “Програмування Інтернет” для спеціальності 121 “Інженерія програмного забезпечення”

Що таке Vue.js

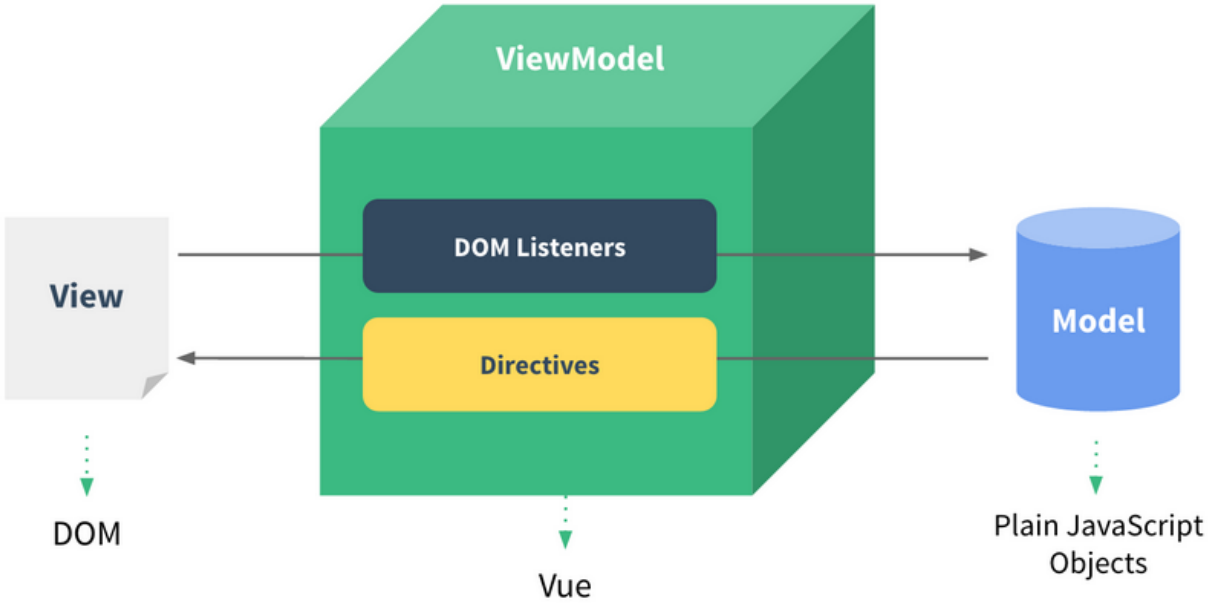
- Vue.js – це JavaScript-фреймворк що використовує шаблон MVVM для створення інтерфейсів користувача на основі моделей даних, через реактивне зв'язування даних.
- Розробник - *Evan You, 2014, Google*
- Поточна версія Vue.js 2.5.16 (травень2018)
- Сайт https://vuejs.org
- Входить в трійку самих популярних JS-фреймворків, поряд з Angular та React
- Підходе для додавання front-end функціональності в уже готові проекти на основі інших технологій, так і для розробки окремих складних SPA-додатків
- Vue.js включено в Laravel 5.3

How it works?

- Inspired by Model-view-viewmodel (MVVM) architectural pattern
- Uses Declarative Rendering
- Dependency tracking system with getter/setters

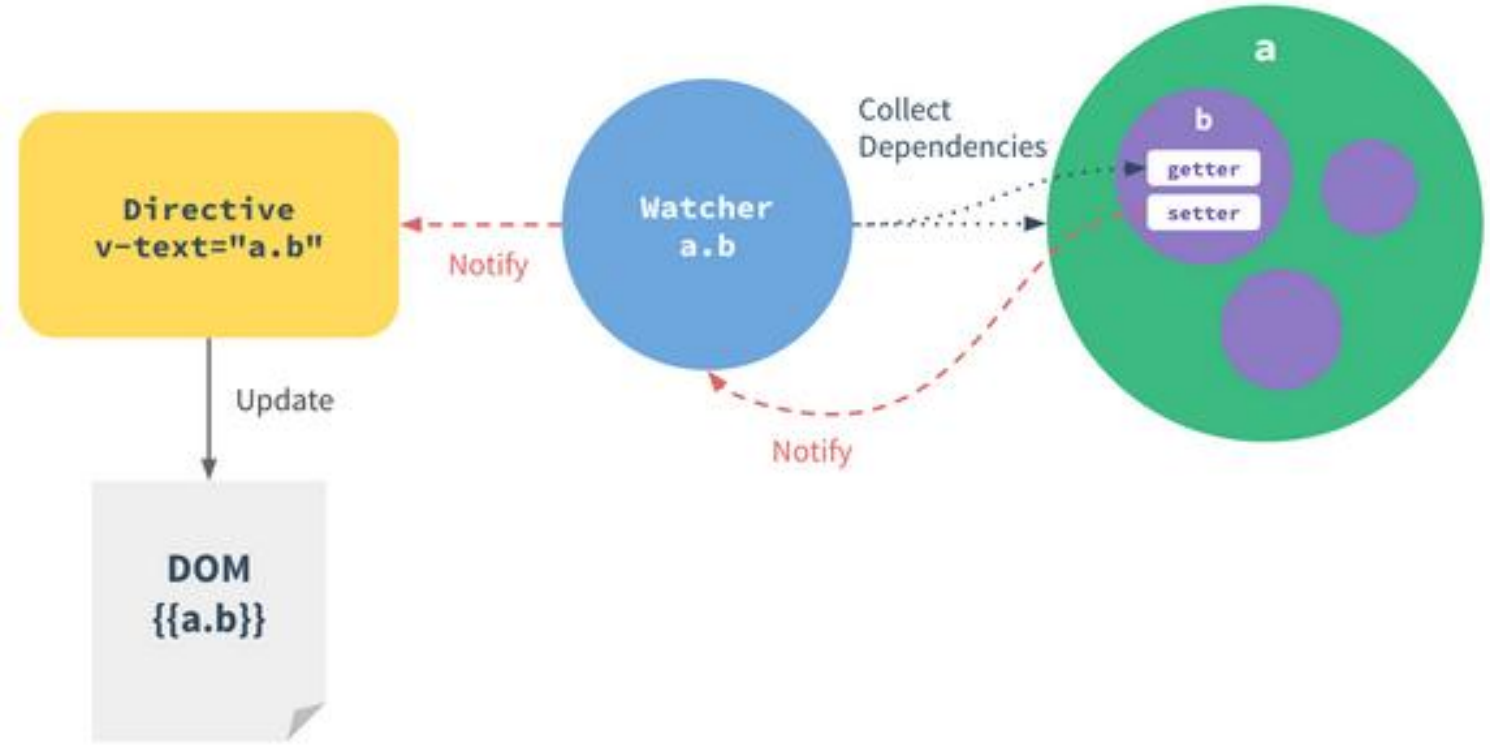


Model-view-viewmodel (MVVM)

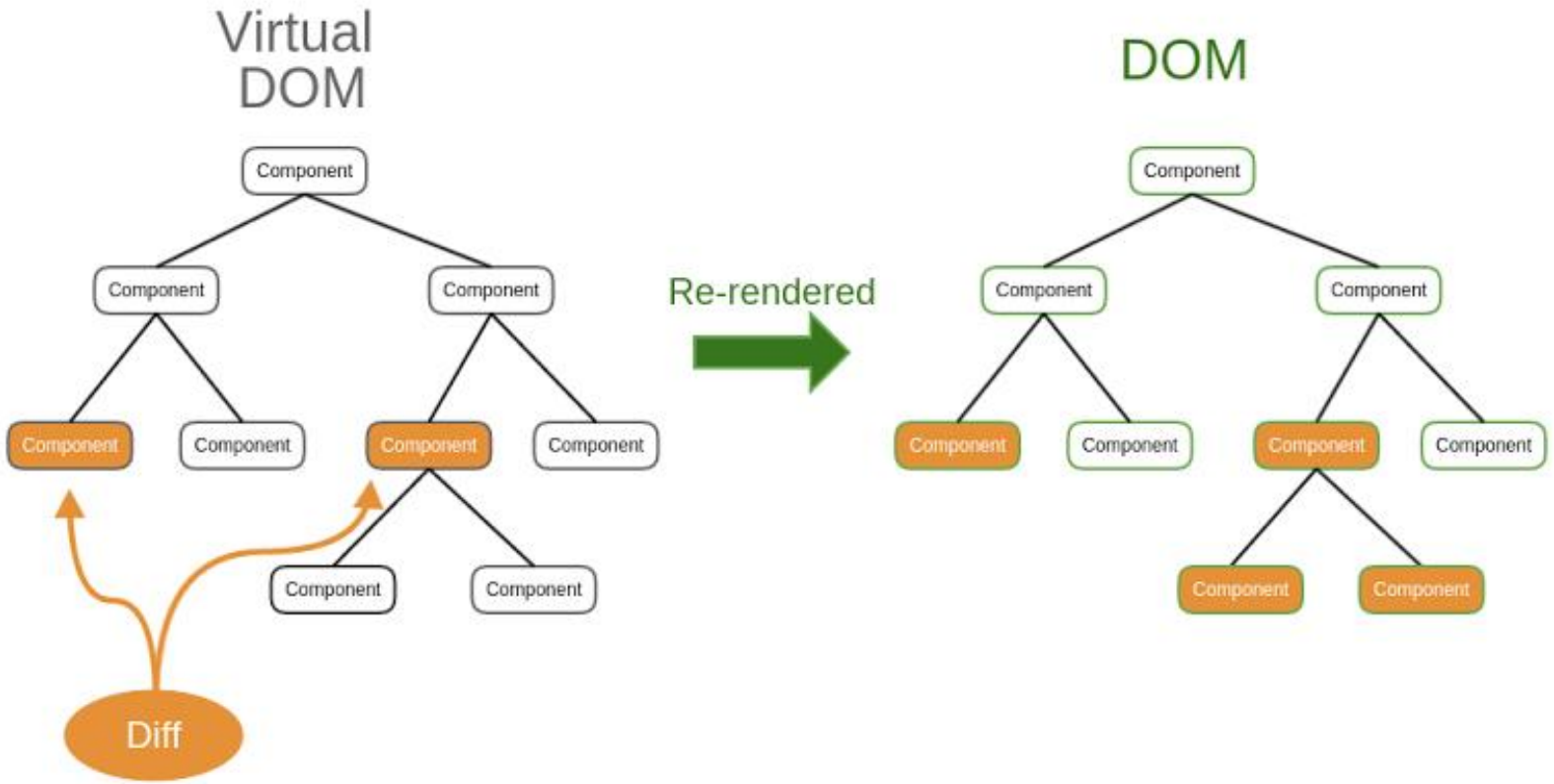


- Дозволяє відділити логіку додатка від візуальної частини (представлення)

Dependency tracking system



Virtual DOM


Virtual DOM
Component
Component
Component
Component
Component
Component
Component
Component
Component
Diff
Re-rendered
DOM
Component
Component
Component
Component
Component
Component
Component
Component

Особливості Vue.js

- Невеликий розмір (біля 20 Кв)
- Швидкий
- Віртуальний DOM
- Підтримується усіма браузерами
- Використовує компонентний підхід
- Легкий у вивченні (інтуітивне розуміння)
- Використовує всі сучасні підходи в розробці web UI
- Гнучкість і висока інтегрованість зі сторонніми технологіями
- Vue відслідковує життєвий цикл компонентів
- Архітектура Vue додатку забезпечує простий і зрозумілий поділ методів, компонентів і даних, це спрощує розширюваність

Інсталяція Vue.js

- Варіант 1. Direct <script> Include. (Просте завантаження і підключення за допомогою тега <script> - як jQuery https://vuejs.org/v2/guide/installation.html)
- Варіант 2. CDN.

  <script src="https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js"></script>
  або <script src=https://unpkg.com/vue></script>
- Варіант 3. NPM (рекоментовано при розробці великих додатків)

  $ npm install vue
- Варіант 4. CLI. Vue надає офіційний CLI для швидкої генерації (scaffolding) файлової структури проекту.

  npm install -g @vue/cli
  vue create my-project

Корисні посилання

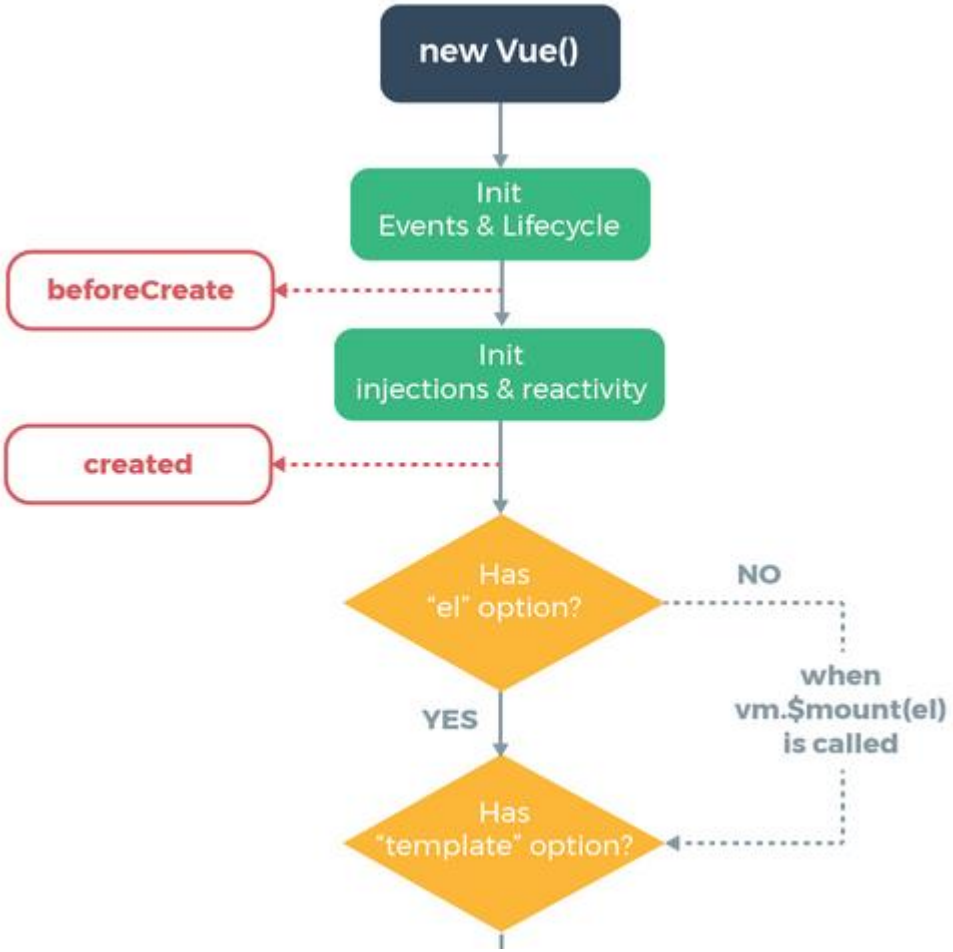
- Comparison with Other Frameworks https://vuejs.org/v2/guide/comparison.html
- Основы Vue.js (Последнее обновление: 29.08.2017) https://metanit.com/web/vuejs/1.1.php
- Promise https://learn.javascript.ru/promise
- 11 Vue.js UI Component Libraries You Should Know In 2018 https://blog.bitsrc.io/11-vue-js-component-libraries-you-should-know-in-2018-3d35ad0ae37f
- 7 Ways To Define A Component Template in VueJS https://medium.com/js-dojo/7-ways-to-define-a-component-template-in-vuejs-c04e0c72900d

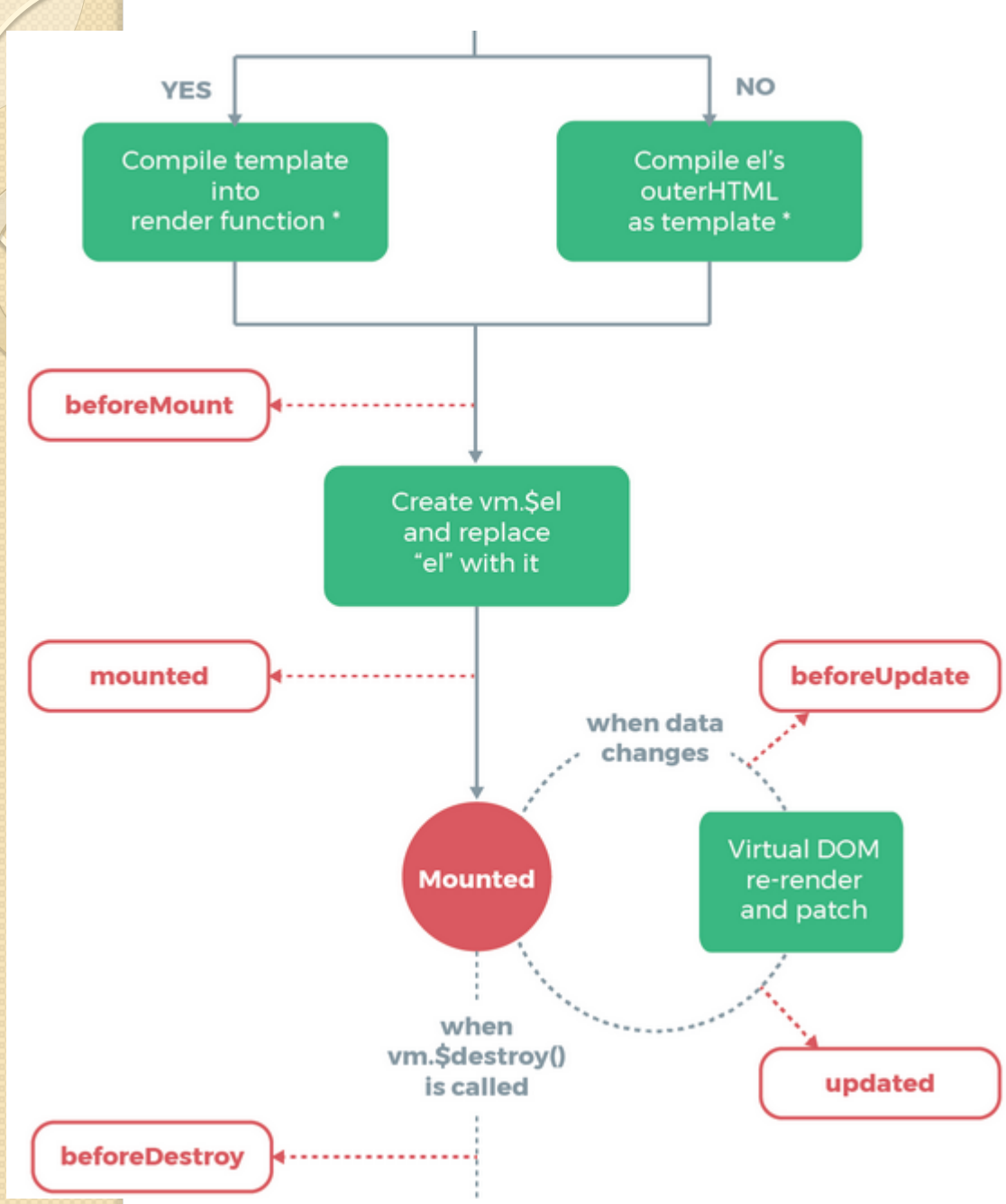
The Vue Instance

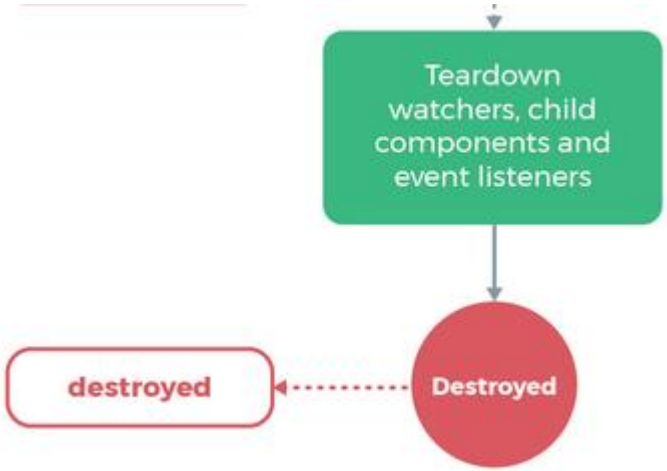
```
var vm = new Vue({
    // options
})
```

- Vue Constructor function
- Properties and Methods (data / events)
- Instance Lifecycle Hooks

The Vue instance lifecycle




YES
NO
Compile template into render function *
Compile el's outerHTML as template *
beforeMount
Create vm.$el and replace "el" with it
mounted
beforeUpdate
when data changes
Mounted
Virtual DOM re-render and patch
when vm.$destroy() is called
updated
beforeDestroy




```
var vm = new Vue({
        data: {
            a: 1
        }
        created: function () {
            console.log('a is: ' + this.a )
        }
})
vm.$watch('a', function (newVal, oldVal) {
        //this callback will be called when 'vm.a' changes
})
```

Компоненти Vue

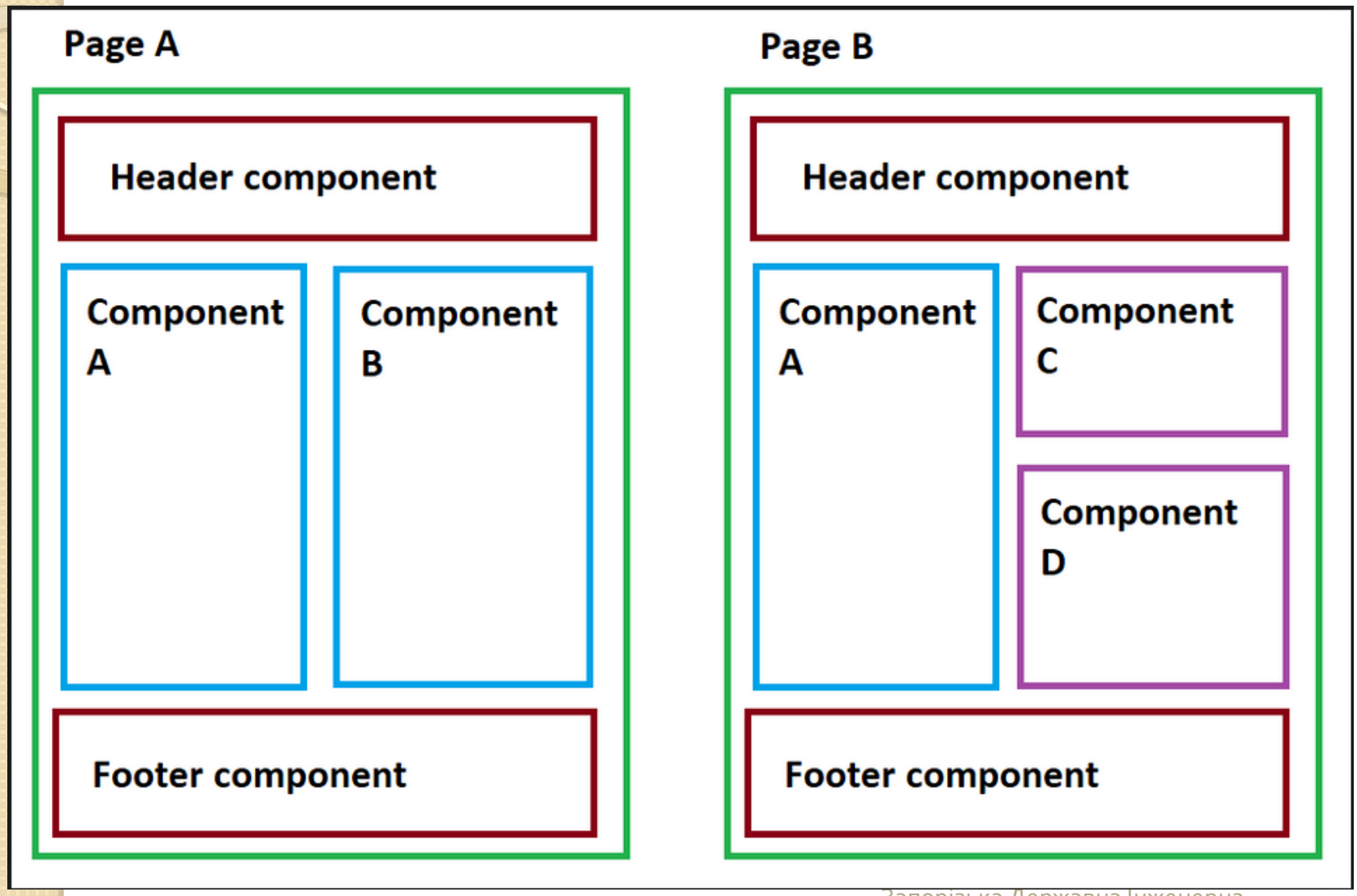
- Компоненти - це повторно використовувані екземпляри Vue зі своїм ім'ям.

```
Vue.component('button-counter', {
  data: function () {
  return {
  count: 0
  }
  },
  template: '<button v-on:click="count++">You clicked me {{ count }}
  times.</button>'
  })
```

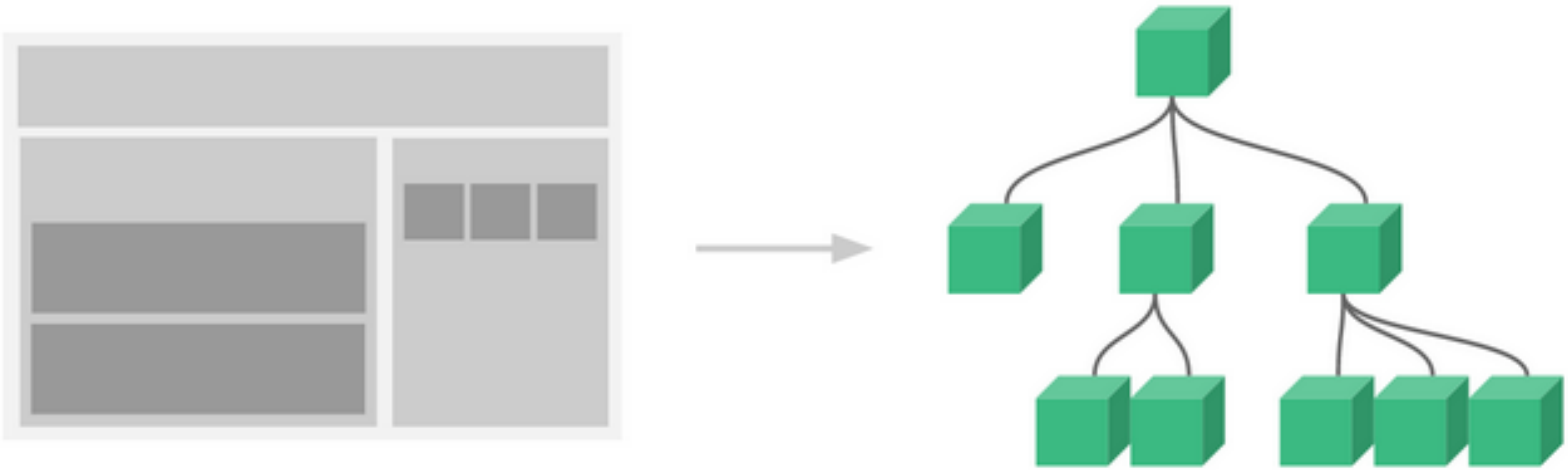
Ми можемо використовувати цей компонент як користувальницький тег всередині кореневого примірника Vue, створеного за допомогою new Vue:

```
<div id="components-demo">
  <button-counter></button-counter>
  </div>
new Vue({ el: '#components-demo' })
```

Компоненти Vue



Компоненти Vue



Анатомія компонента Vue

- Структура –> <Template>
- Представлення –> <Style>
- Данні –> <Script>

Директиви і атрибути

Директиви – це команди, що починаються з префікса –v, які приєднуються до елементів DOM

- v-text (use {{}})
- v-on (shorthand @)
- v-bind (shorthand :)
- v-show
- v-if / v-else / v-else-if
- v-for / key
- slot
- v-model
- v-pre / v-cloak / v-onse

Special attributes

- key
- ref
- slot
- is

Component Internals

- data: Internal state
- props: args
- computed: Cacheable read functions
- methods: Event handling
- Provided by plugins: Extensions

Приклад 1. Hello, World.

```
<html>
<head>
  <title>Vue App</title>
  <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
  <div id="app">
    {{text}}
  </div>
  <script>
    var vm = new Vue({
      el:'#app',
      data: {
        text: 'Hello, World!'
      }
    })
  </script>
</body>
</html>
```

Приклад 2. Динамічний Hello

```
<!DOCTYPE html>
<html>
<head>
  <title>Vue App</title>
  <script
    src="https://unpkg.com/vue/dist/
    vue.js"></script>
</head>
<body>
  <div id="app">
    <input type="text" v-
    on:input="changeName">
    <h1>Hello, {{name}}!</h1>
  </div>
  <script>
    var vm = new Vue({
      el:'#app',
      data: {
        name: 'Vue'
      },
      methods:{
          changeName: function(event){
              this.name =
      event.target.value
              }
        }
      })
    </script>
</body>
</html>
```

Приклад 3. Директива v-for

```
<!DOCTYPE html>
<html>
<head>
  <title>Vue App</title>
  <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
  <div id="app">
    <ul>
      <li v-for="dog in dogs">{{dog}}</li>
    </ul>
  </div>
  <script>
    new Vue({
      el:'#app',
      data: {
        dogs: ['Rex','Rover','Henrietta','Alan']
      }
    })
  </script>
</body>
</html>
```

Приклад 4. Директива v-for

```
<!DOCTYPE html>
<html>
<head>
  <title>Vue App</title>
  <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
  <div id="app">
    <ul>
      <li v-for="(rent, city) in averageRent">
      The average rent in {{ city }} is ${{ rent }}</li>
    </ul>
  </div>
  <script>
    new Vue({
      el:'#app',
      data: {
        averageRent: {
              London: 1650, Paris: 1730, NYC: 3680
        }
      }
    })
  </script>
</body>
</html>
```

Приклад 5. Two-Way Data Binding

```
<!DOCTYPE html>
<html>
<head>
  <title>Vue App</title>
  <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
  <div id="app">
     <input type="text" v-model="inputText">
     <p>inputText: {{ inputText }}</p>
  </div>
  <script>
    new Vue({
      el:'#app',
      data: {
        inputText: 'initial value'
      }
    })
  </script>
</body>
</html>
```

Приклад 6. Methods

```
div id="app">
    <ul>
      <li v-for="number in filterPositive(numbers)">{{ number }}</li>
    </ul>
  </div>
. . . . . .
data: {
      numbers: [-5, 0, 2, -1, 1, 0.5]
},
methods: {
      filterPositive(numbers) {
          return numbers.filter((number) => number >= 0);
      }
}
```

Приклад 7. this

```
<div id="app">
    <p>The sum of the positive numbers is {{ getPositiveNumbersSum()
    }}</p>
  </div>

data: {
        numbers: [-5, 0, 2, -1, 1, 0.5]
},
methods: {
        getPositiveNumbers() {
             return this.numbers.filter((number) => number >= 0);
            },
        getPositiveNumbersSum() {
             return this.getPositiveNumbers().reduce((sum, val) => sum + val);
            }
}
```

Computed Properties

- **Обчислювальні властивості** розташовуються на півдорозі між властивостями об'єкта даних та методами: ви можете отримати до них доступ, як якщо б вони були властивостями об'єкта даних, але вони задані як функції.
- Ознайомтесь з наступним прикладом, який приймає дані, що зберігаються у вигляді цифр, додає їх разом і виводить загальну суму на сторінку: (Приклад 8.)

```
<div id="app"><p>Sum of numbers: {{ numberTotal }}</p></div>
data: {
        numbers: [5, 8, 3]
      },
computed: {
    numberTotal() {
        return numbers.reduce((sum, val) => sum + val);
    }
}
```

Приклад 9. Директива v-if

```
<h1>Tasks</h1>
<ul>
   <li v-for="task in tasks" v-if="task.done">{{ task.text }}, Done: {{ task.done
   }}</li>
</ul>

   data: {
      tasks: [
         {text: 'Learning Vue', done: false}
         {text: 'Get Breakfast', done: true}
         {text: 'Buy Milk', dene: false}
      ]
   }
```

Vue.js devtools

Browser devtools extension for debugging Vue.js applications. (https://github.com/vuejs/vue-devtools)

- Get the Chrome Extension https://chrome.google.com/webstore/detail/vuejs-devtools/nhdogjmejiglipccpnnnanhbledajbpd
- Get the Firefox Addon https://addons.mozilla.org/en-US/firefox/addon/vue-js-devtools/

Потрібен command-line HTTP Server (https://www.npmjs.com/ , пошук: 'http-server') Installation via npm:

npm install http-server -g

Usage: http-server [path] [options]

[path] defaults to ./public if the folder exists, and ./ otherwise.

Now you can visit http://localhost:8080 to view your server

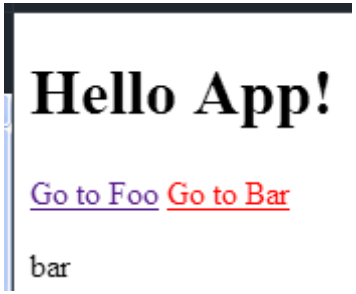
Для відлагодження треба змінити посилання:

<script src="https://vuejs.org/js/vue.js"></script>

Vue Router

- **Vue Router - офіційна бібліотека маршрутизації для Vue.js.** Вона глибоко інтегрується з ядром Vue.js
- https://router.vuejs.org/
- Створювати односторінкові додатки (SPA) використовуючи Vue + Vue Router дуже просто. За допомогою Vue.js, ми вже компонуємо свій додаток з компонентів.
- Додаючи Vue Router, ми просто співставляємо наші компоненти з маршрутами і пояснюємо Vue Router де їх відображати.
- Розглянемо приклад https://router.vuejs.org/ru/guide/

  https://jsfiddle.net/yyx990803/xgrjzsup/



Приклад

HTML

```html
<script src="https://unpkg.com/vue/dist/vue.js"></script>
<script src="https://unpkg.com/vue-router/dist/vue-router.js"></script>

<div id="app">
  <h1>Первое приложение!</h1>
  <p>
    <!-- используем компонент router-link для навигации -->
    <!-- входной параметр `to` определяет URL для перехода -->
    <!-- `<router-link>` по умолчанию отображается тегом `<a>` -->
    <router-link to="/foo">Перейти к Foo</router-link>
    <router-link to="/bar">Перейти к Bar</router-link>
  </p>
  <!-- отображаем тут компонент, для которого совпадает маршрут -->
  <router-view></router-view>
</div>
```

JavaScript

```js
// 0. Если используем модульную систему (например через vue-cli),
// импортируем Vue и VueRouter и затем вызываем `Vue.use(VueRouter)`.

// 1. Определяем компоненты для маршрутов.
// Они могут быть импортированы из других файлов
const Foo = { template: '<div>foo</div>' }
const Bar = { template: '<div>bar</div>' }

// 2. Определяем несколько маршрутов
// Каждый маршрут должен указывать на компонент.
// "Компонентом" может быть как конструктор компонента, созданный
// через `Vue.extend()`, так и просто объект с опциями компонента.
// Мы поговорим о вложенных маршрутах позднее.
const routes = [
  { path: '/foo', component: Foo },
  { path: '/bar', component: Bar }
]

// 3. Создаём экземпляр маршрутизатора и передаём маршруты в опции `routes`
// Вы можете передавать и дополнительные опции, но пока не будем усложнять.
const router = new VueRouter({
  routes // сокращённая запись для `routes: routes`
})

// 4. Создаём и монтируем корневой экземпляр приложения.
// Убедитесь, что передали экземпляр маршрутизатора в опции
// `router`, чтобы позволить приложению знать о его наличии.
const app = new Vue({
  router
}).$mount('#app')

// Всё, приложение работает! ;)
```

- Впроваджуючи маршрутизатор, ми зможемо отримати до нього доступ через this.$router, а також до поточного маршруту через this.$route всередині будь-якого компонента:

```
// Home.vue
export default {
  computed: {
    username () {
      // Мы скоро разберём что такое `params`
      return this.$route.params.username
    }
  },
  methods: {
    goBack () {
      window.history.length > 1
        ? this.$router.go(-1)
        : this.$router.push('/')
    }
  }
}
```

Динамічні шляхи

- Дуже часто нам потрібно зіставити маршрути з заданим шаблоном з одним і тим же компонентом. Наприклад, у нас може бути компонент **User**, який повинен відображатися для всіх користувачів, але з різними **ID** користувачів. У vue-router ми можемо використовувати динамічний сегмент в маршруті, щоб досягти цього:

```
const User = {
  template: '<div>Пользователь</div>'
}

const router = new VueRouter({
  routes: [
    // динамические сегменты начинаются с двоеточия
    { path: '/user/:id', component: User }
  ]
})
```

- Тепер всі URL виду /user/foo і /user/bar будуть відповідати одному маршруту.

Vuex

- state: application state as a tree
- getters: state read methods
- mutations: state write methods
- actions: user commands