Что такое Vue.js?

- Vue (произносится /vjuː/, примерно как view) — это прогрессивный фреймворк для создания пользовательских интерфейсов. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA, Single-Page Applications), если использовать его совместно с современными инструментами и дополнительными библиотеками.

Экземпляр vue

- Каждое приложение Vue начинается с создания нового экземпляра Vue с помощью функции Vue:

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

- При создании экземпляра vue необходимо передать объект опций. Большая часть этого руководства посвящена описанию, как вы можете использовать эти опции для достижения желаемого поведения. Для справки вы также можете посмотреть полный список опций в справочнике API.

Данные и методы

- Когда экземпляр Vue создан, он добавляет все свойства, найденные в его объекте data, в систему реактивности Vue. Когда значения этих свойств изменятся, представление будет «реагировать», обновляясь в соответствии с новыми значениями.

Шаблон

- Наиболее простой способ связывания данных — это текстовая интерполяция с использованием синтаксиса Mustache (двойных фигурных скобок):

```
<span>Сообщение: {{ msg }}</span>
```

- Выражение в фигурных скобках будет заменено значением свойства msg соответствующего объекта данных. Кроме того, оно будет обновлено при любом изменении этого свойства.
- Возможно также выполнение однократной интерполяции, которая не обновится при изменении данных — используйте для этой цели директиву v-once, но учтите, что это повлияет сразу на все связанные переменные в рамках данного элемента:

```
<span v-once>Это сообщение никогда не изменится: {{ msg }}</span>
```

шаблоны

- В ядре Vue.js находится система, которая позволяет декларативно отображать данные в DOM, используя простые шаблоны:

```
17
18      <body>
19          <div id="app">
20              {{ message }}
21          </div>
22
23          <script>
24  var app = new Vue({
25      el: '#app',
26      data: {
27          message: 'Hello Vue!'
28      }
29  })
30          </script>
31      </body>
```

Сырой HTML

- Значение выражения, обрамлённого двойными фигурными скобками, подставляется как простой текст, а не как HTML. Если вы хотите вывести HTML, вам понадобится директива v-html:

```
<div v-html="rawHtml"></div>
```

Атрибуты

- Синтаксис двойных фигурных скобок не работает с HTML-атрибутами, используйте вместо него директиву v-bind:

```
<div v-bind:id="dynamicId"></div>
```

```
<button v-bind:disabled="isButtonDisabled">Кнопка</button>
```

Использование выражений JavaScript

```
{{ number + 1 }}

{{ ok ? 'YES' : 'NO' }}

{{ message.split('').reverse().join('') }}

<div v-bind:id="'list-' + id"></div>
```

Условия и циклы

```
<div id="app-3">
  <span v-if="seen">Сейчас меня видно</span>
</div>
```

```
var app3 = new Vue({
  el: '#app-3',
  data: {
    seen: true
  }
})
```

Директивы

- Директивы — это специальные атрибуты с префиксом v-. В качестве значения такие атрибуты принимают одно выражение JavaScript (за исключением v-for, о которой ниже). Директива реактивно применяет к DOM изменения при обновлении значения этого выражения. Давайте вспомним пример, который мы видели во введении:

```
<p v-if="seen">Сейчас меня видно</p>
```

- В данном случае директива v-if удалит или вставит элемент <p> в зависимости от истинности значения выражения seen.

Условия и циклы

- директива v-for может быть использована для отображения списков, используя данные из массива

```
<div id="app-4">
  <ol>
    <li v-for="todo in todos">
      {{ todo.text }}
    </li>
  </ol>
</div>
```

```
var app4 = new Vue({
  el: '#app-4',
  data: {
    todos: [
      { text: 'Посадить дерево' },
      { text: 'Построить дом' },
      { text: 'Вырастить сына' }
    ]
  }
})
```

V-model

- Vue также содержит директиву v-model, позволяющую легко связывать элементы форм и состояние приложения:

```
<div id="app-6">
  <p>{{ message }}</p>
  <input v-model="message">
</div>
```

```
var app6 = new Vue({
  el: '#app-6',
  data: {
    message: 'Hello Vue!'
  }
})
```