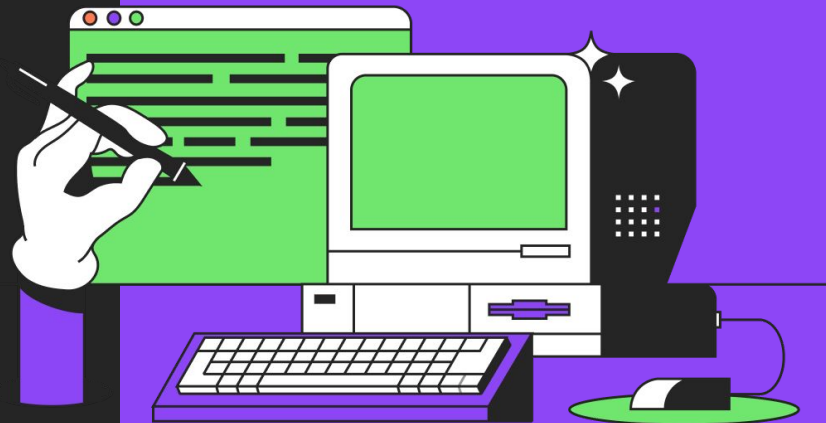


Компоненты

Урок 3





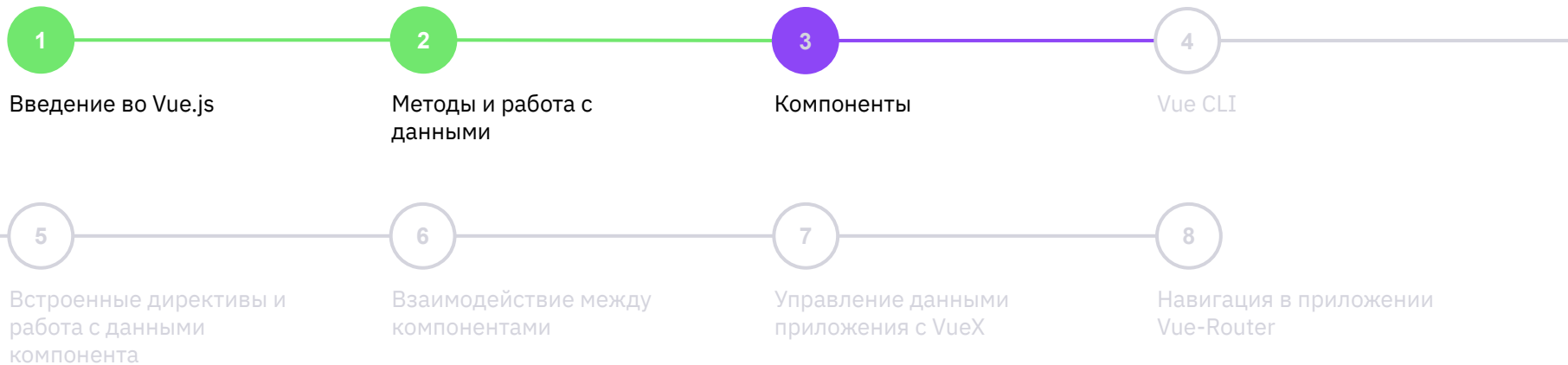
Кадочников Алексей

Frontend-разработчик

- ☀ Веб-разработчик со стажем более 9 лет
- ☀ Преподаватель GeekBrains с 2015 года
- ☀ Автор курсов по Frontend на портале Geekbrains
- ☀ Работал в таких компаниях как VK и Wizard-C



План курса





Что будет на уроке сегодня



Введение



Создание компонент



Итоги



Введение

Если приложение плохо спроектировано, работать с ним становится тяжело. Когда всё приложение описано в одном файле, в нём становится очень сложно разобраться, а кроме того, разные части приложения могут взаимодействовать непредсказуемо и неконтролируемо. Например, одноименные переменные могут оказаться в одной области видимости и перекрывать друг друга.





Создание компонент





Новый компонент

Для этого у Vue есть метод `component()`.
Этот метод принимает два аргумента: название компонента и объект с настройками:

```
1 Vue.component('some', {});
```

Название может быть любым, но если оно состоит из нескольких слов, их принято разделять дефисом:

```
1 Vue.component('some-component', {});
```



Новый компонент

Главный параметр компонента – template, html-разметка. Напишем что-нибудь простое:

```
1 Vue.component('some-component', {
2   template: '<h1>Hi Component!</h1>'
3 });
4
5
6 const app = new Vue({
7   el: '#app',
8   data: {
9
10
11   }
12 })
13
```




Ограничения шаблона

Важно помнить, что шаблон может содержать только один внешний элемент.

Будем идти от обратного, рассмотрим простую ситуацию, когда необходимо добавить 2 параграфа

```
1      Vue.component('some', {  
2          template: `  
3              <p></p>  
4              <p></p>  
5          `,  
6      });  
7
```



Проброс содержимого

Содержимое, которое находится между открывающим и закрывающим тегами компонента, можно пробрасывать внутрь и подставлять в шаблон. Для подстановки содержимого используется тег `<slot>`:

```
1 <div id="app">
2   <some-component>Hello</some-component>
3   <some-component>New</some-component>
4   <some-component>Heading</some-component>
5 </div>
6
7
8 <script>
9   Vue.component('some-component', {
10     template: '<h2><slot></slot></h2>'
11   });
12 </script>
13
```



Работа с данными

В компонентах данные тоже хранятся в поле `data`, но здесь это не объект, а функция, возвращающая объект:

```
1 Vue.component('some-component', {  
2   template: '<h2></h2>',  
3   data() {  
4     return {  
5       name: 'Frodo',  
6     };  
7   },  
8 });  
9
```



Работа с данными

Имя можно подставить в шаблон компонента, используя mustache-синтаксис (усатый синтаксис):

```
1
2   Vue.component('some-component', {
3     template: '<h2>{{ name }}</h2>',
4     data() {
5       return {
6         name: 'Frodo',
7       };
8     },
9   });
10
```



Передача свойств

В компоненты можно передавать данные извне с помощью свойств. Это значения, которые устанавливаются во время вызова компонента и которые можно использовать для преобразования и подстановки в шаблон.





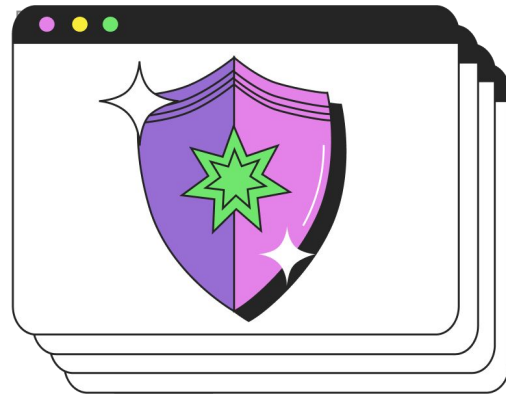
Итоги урока








Итоги урока


Мы научились создавать простые компоненты, рассмотрели вызов данных компонентов, научились создавать компоненты внутри компонентов и конечно же научились передавать значения внутри компонента.





Итоги урока

-  Узнали для чего используются компоненты
-  Рассмотрели создание компонент
-  Подвели итоги

Спасибо 
за внимание

