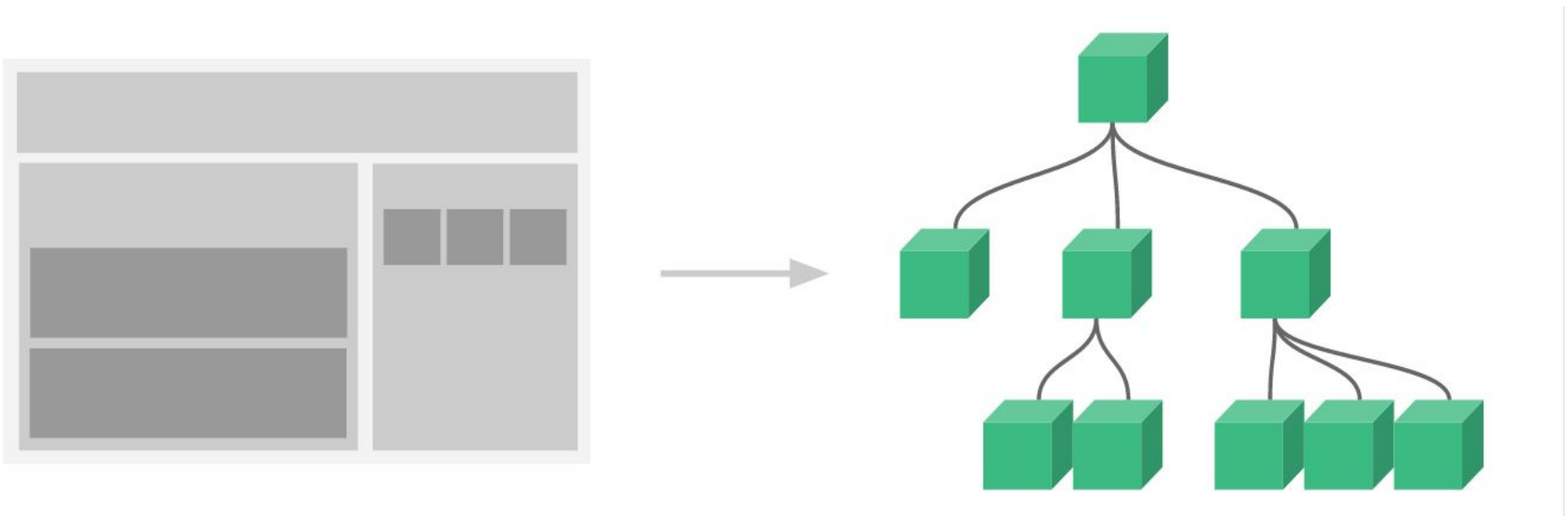
# Как спроектировать VueJS компоненты и не прострелить себе ногу?

Алексей Иванов
TrueConf
2018

TrueConf

# Зачем вообще нужны компоненты?

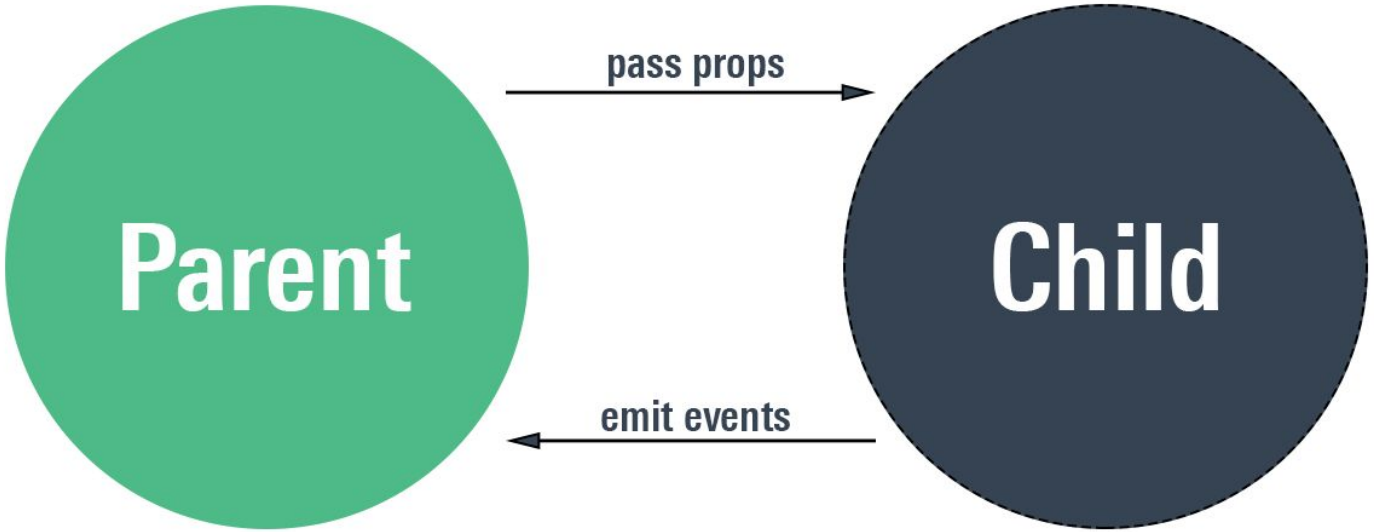


Источник: Google

TrueConf

# Способы связи компонентов

## Передача параметров и пользовательские события




Источник: Google

# Рассмотрим пример

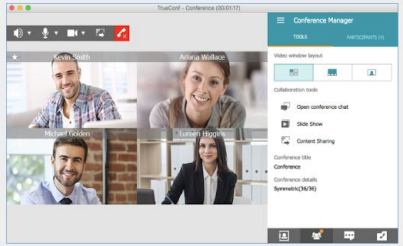
TrueConf

Download | Recommended equipment | Personal area

Windows | macOS | Linux | Android | iOS

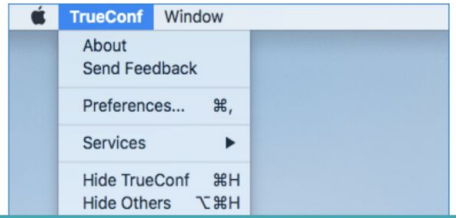
Download TrueConf for macOS application.

Download




Download on the Mac App Store

1. Install and launch TrueConf for macOS application.
2. Connect to **ruiu2.trueconf.name** server automatically or manually by following the instruction:
   - Select **Preference...** in the application menu and go to **Network** section.




Server: ruiu2.trueconf.name

Powered by TrueConf Server version 4.4.2.10054, video conferencing software. More at www.trueconf.com.

# Способы связи компонентов

## Передача входных параметров

simple.vue

```
// Parent component

<navigation-tabs
:selected="selectedTab"
:list="tabList"
@select="selectedTab = $event">
</navigation-tabs>

<content
:selected="selectedTab">
</content>
```

simple.vue

```
// Child component (navigation-tabs)

props: {
 selected: String,
 list: Array,
},

methods: {
 handler(value) {
   this.$emit('selectTab', value);
 }
}
```

## Пользовательские события

# Рассмотрим пример

TrueConf

Download | Recommended equipment | Personal area

English (English)
Russian (Русский)
Chinese (汉语)
Farsi (فارسی)
French (Français)
German (Deutsch)
Italian (Italiano)

## Welcome to TrueConf Server Guest Page

DOWNLOA

Download application t
conference right

SCHEDULE CONFERENCE

Schedule video conferences and webinars to join via a link.

CONTACT ADMINISTRATOR

If you experience any problems, please contact your administrator.

CONNECT TO THE CONFERENCE

If you know conference ID, you can connect to it in your browser.

Server: ruiu2.trueconf.name

Language:

Powered by TrueConf Server version 4.4.2.10054, video conferencing software. More at www.trueconf.com.

# Способы связи компонентов

simple.vue

```
// Root component

<root-header
:language="language"
@changeLanguage="language = $event">
</root-header>

...

<root-header
:language="language"
@changeLanguage="language = $event">
</root-header>
```

**Корневой компонент**

TrueConf

# Способы связи компонентов

simple.vue

```
// Root-header component

<language-selector
:language="language"
@changeLanguage="$emit('changeLanguage', $event)">
</language-selector>

props: {
 language: String,
},
```

Хедер

simple.vue

```
// Root-footer component

<language-selector
:language="language"
@changeLanguage="$emit('changeLanguage', $event)">
</language-selector>

props: {
 language: String,
},
```

Футер

# Способы связи компонентов

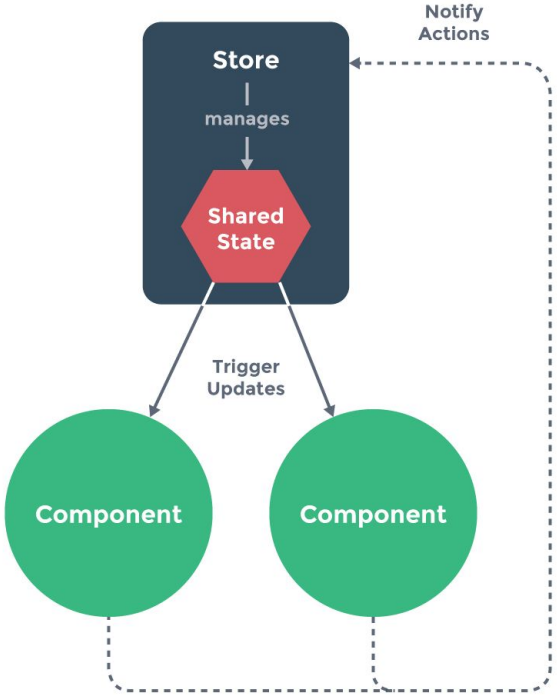
simple.vue

```
// Language-selector component

props: {
 language: String,
},

methods: {
 changeLanguage(lang) {
   this.$emit('changeLanguage', lang);
 }
},
```

**Компонент выбора языка**

TrueConf

# Способы связи компонентов

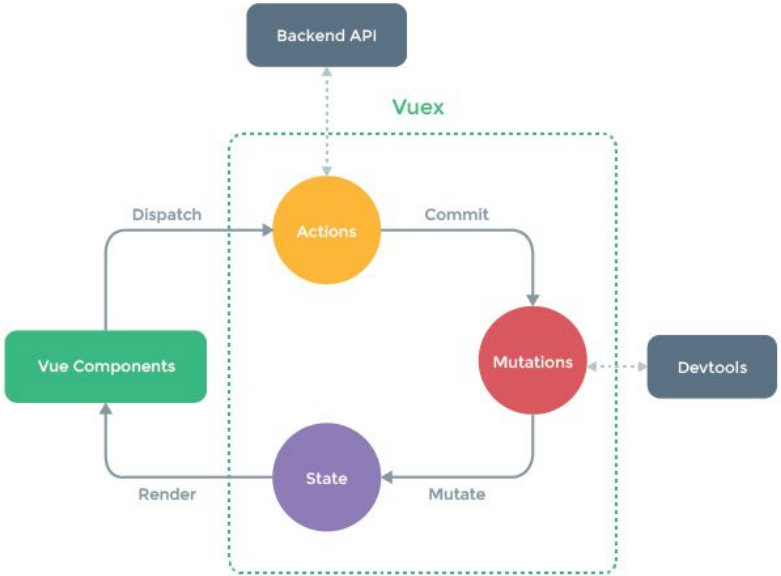



Источник: Google

## Vuex-хранилище

# Способы связи компонентов

## Vuex-хранилище




Источник: Google

# Способы связи компонентов

simple.vue

```
state: {
 language: 'en',
},
getters: {
 appLanguage(state) {
   return state.language;
 },
},
mutations: {
 _setLanguage(state, lang) {
   state.language = lang;
 },
},
actions: {
 changeLanguage({ commit }, lang) {
   localization.$setLocale(lang);
   commit('_setLanguage', lang);
 },
}
```

**Vuex модуль**

TrueConf

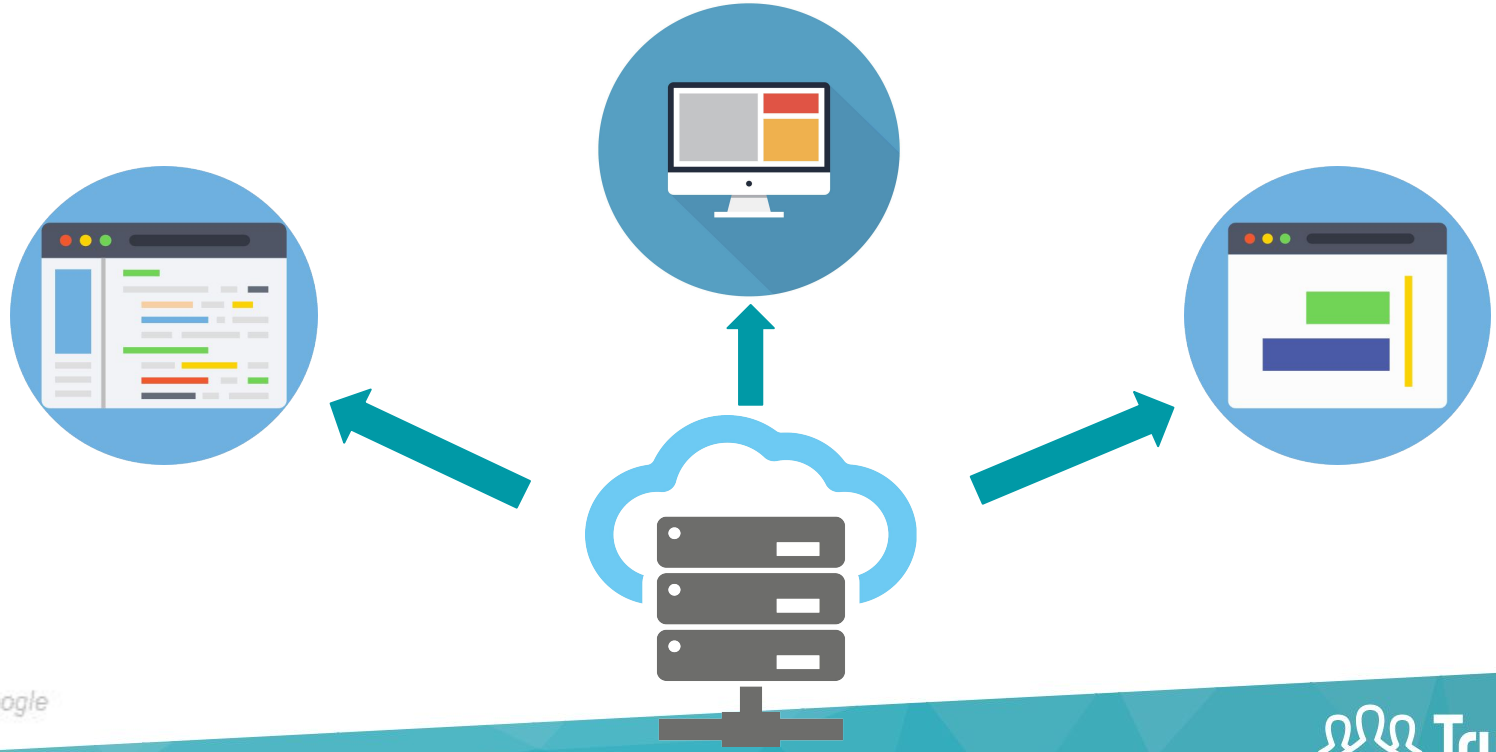
# Способы связи компонентов

simple.vue

```
// Language-selector component

methods: {
 changeLanguage(lang) {
   this.$store.dispatch('changeLanguage', lang);
 }
},
computed: {
 language(lang) {
   return this.$store.getters.appLanguage ;
 }
},
```

## Использование Vuex

# Создание библиотеки компонентов



Источник: Google

TrueConf

# Создание библиотеки компонентов

## Использование Vuex

```
// Vuex

// Getters:
// -appLanguage

// Actions:
// -changeLanguage
```

Источник: Google

TrueConf

# Создание библиотеки компонентов

**Пользовательские события и параметры**

```
// Props:
// -appLanguage

// Events:
// -changeLanguage
```

Источник: Google

TrueConf

# Способы связи компонентов

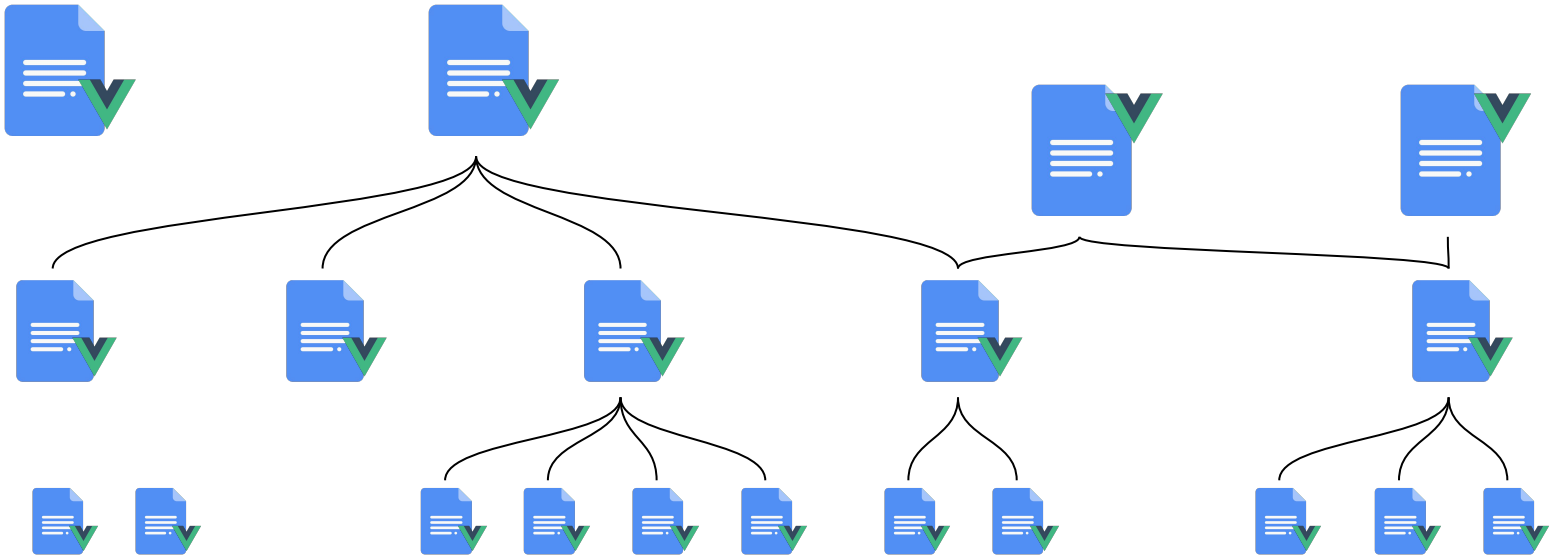
## Пользовательские события

- \+ простой и удобный механизм обмена данными между дочерним и родительским компонентом
- \+ отсутствие жесткой зависимости от родительского компонента
- − необходимость реализовывать длинные цепочки связей в случае работы с большим количеством вложенных компонентов

## Использование Vuex

- \+ удобная прямая работа с глобальным состоянием приложения независимо от количества вложенных компонентов
- − не подходит для создания независимых библиотек компонентов

# Что делать, если компонент получается с большой вложенностью?



Источник: Google

# Рассмотрим пример

TrueConf

Скачать | Рекомендуемое оборудование | Personal area

## Авторизация

TrueConf ID

Пароль

Забыли пароль?

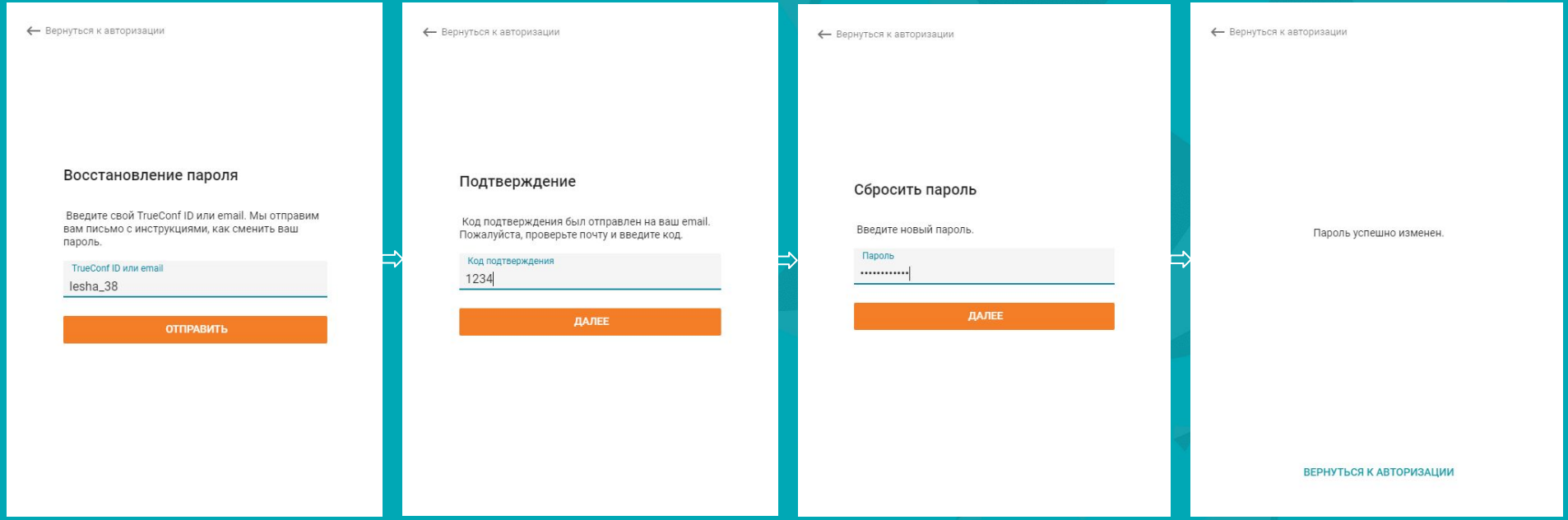
SIGN IN

Создать учетную запись

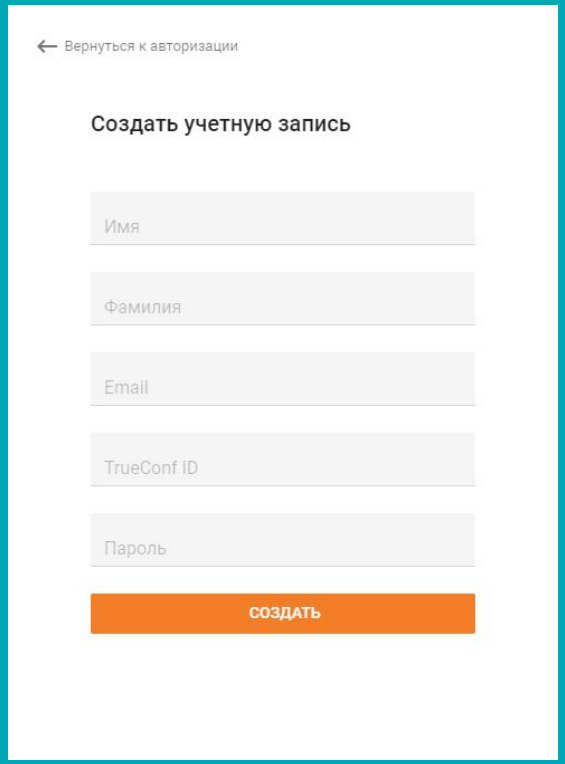
Сервер: ruj39.trueconf.name

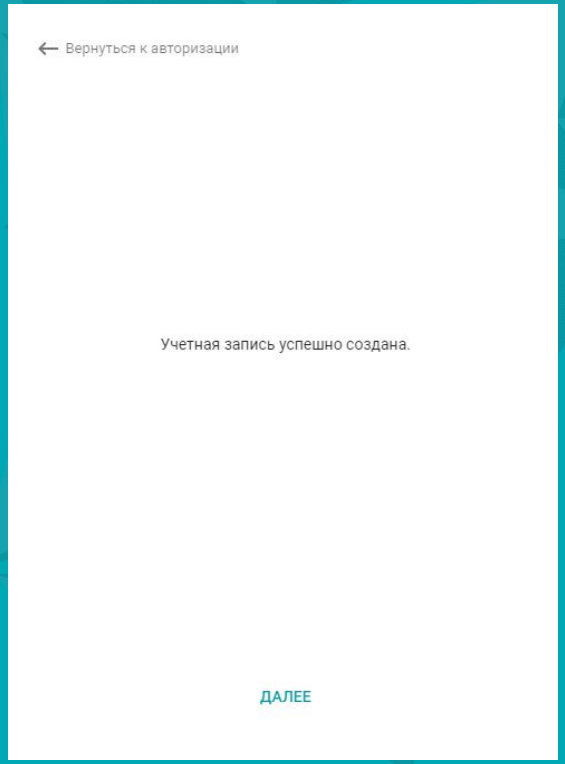
Служба работает на ПО для видеоконференцсвязи TrueConf Server версии 4.4.2.10065. Подробнее на www.trueconf.ru.

# Что делать, если компонент получается с большой вложенностью?

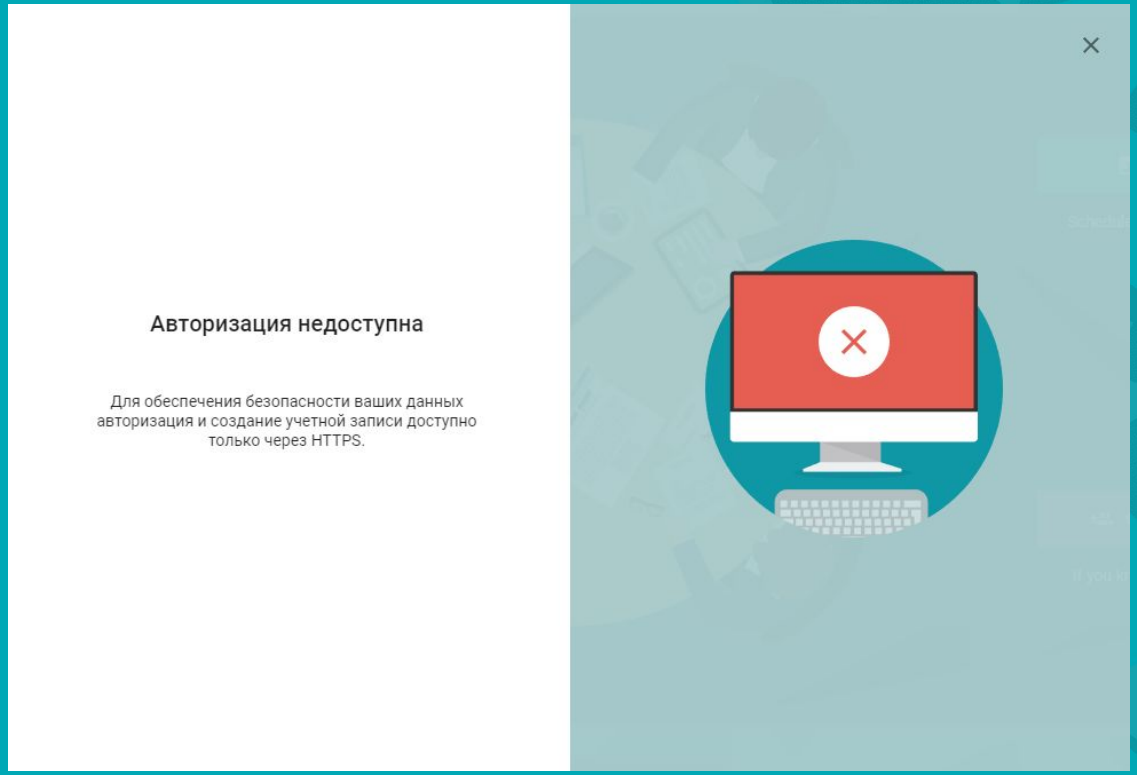


# Что делать, если компонент получается с большой вложенностью?

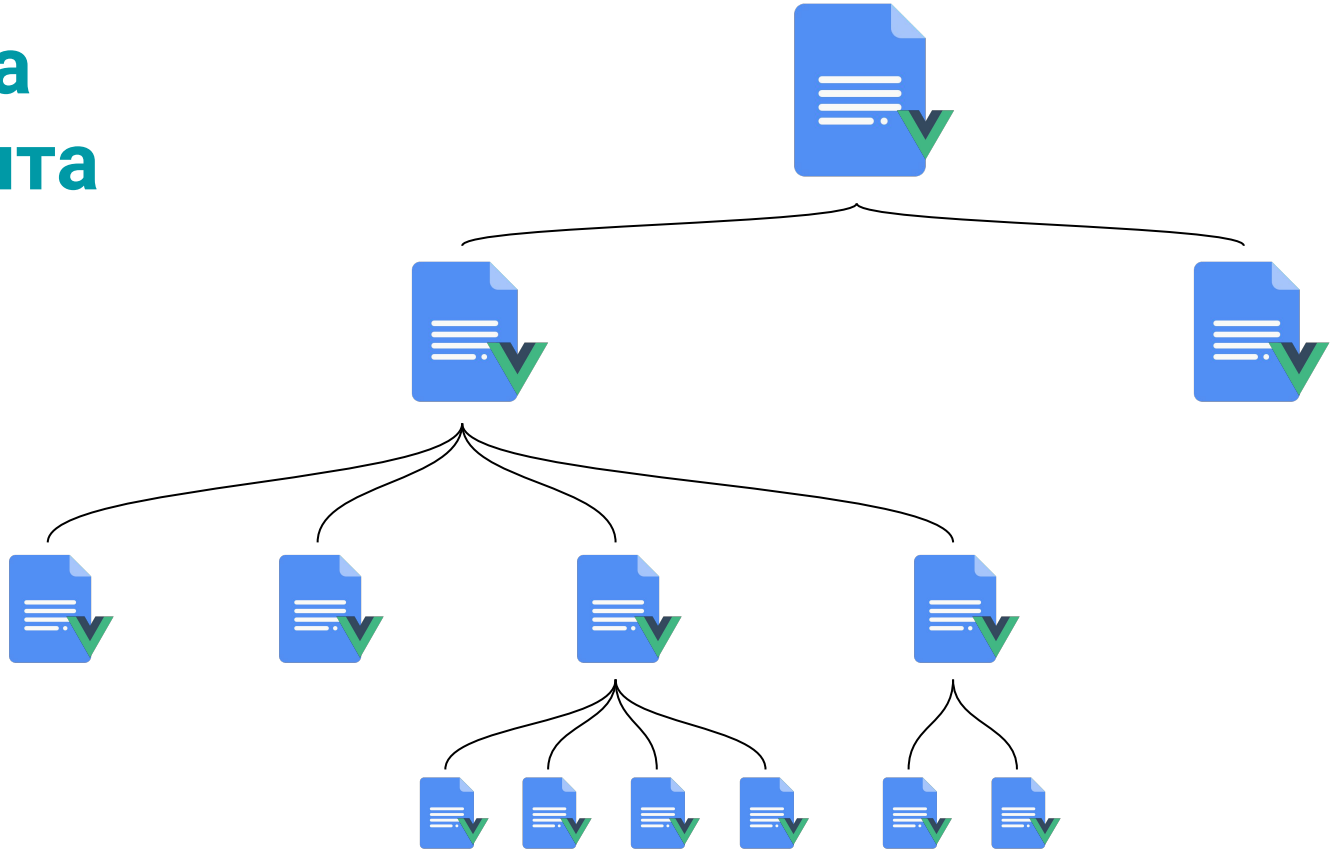


# Что делать, если компонент получается с большой вложенностью?

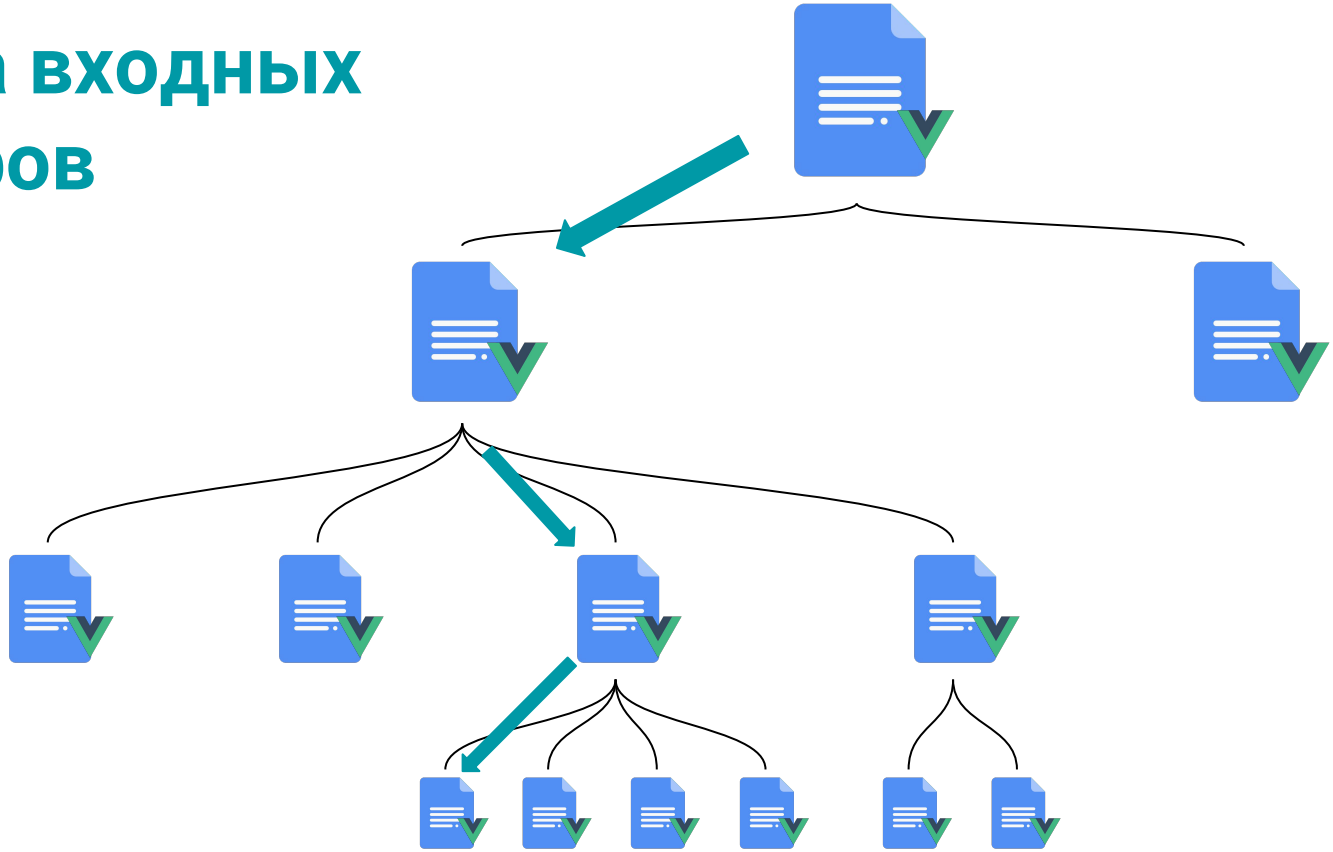


# Структура компонента



Источник: Google

# Передача входных параметров



Источник: Google

# Передача входных параметров

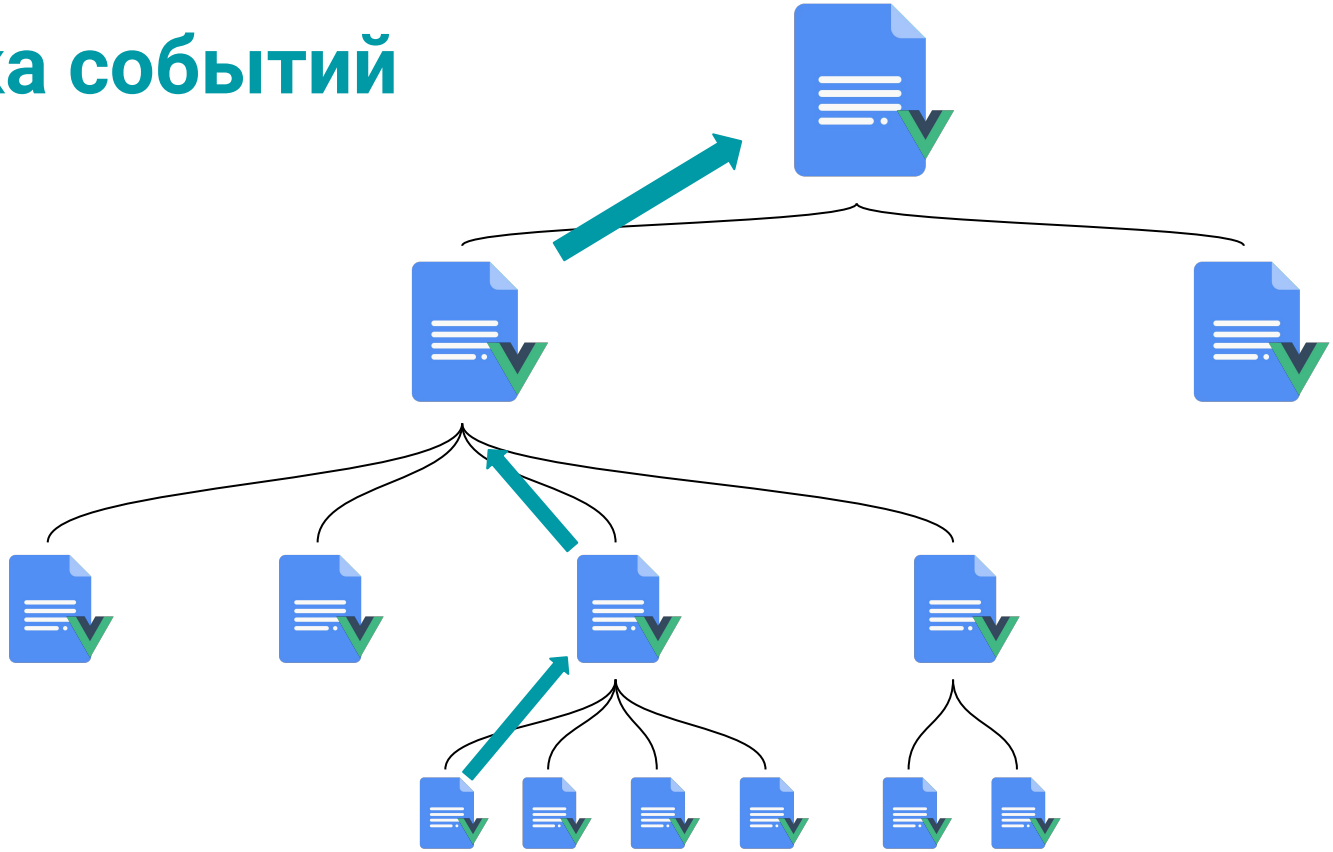
simple.vue

```
// Parent component

...
<authorization
:permissions="permissions"
startScreen="auth"
...>
</authorization>
...
```

simple.vue

```
// Child component

props: {
 startScreen: {
  type: String,
  default: "auth"
 },
 permissions: {
  type: Object,
  default: null
 },
}
```

# Обработка событий



Источник: Google

# Обработка событий

simple.vue

```
// Parent component
...

<component-n
@eventA="$emit('eventA')"
@eventB="$emit('eventB', $event)"
...>
</component-n>

...
```

simple.vue

```
// Child component

methods: {
 eventA() {
   this.$emit('eventA');
 },
 eventB(value) {
   this.$emit('eventB', value);
 }
},
```

# Обработка событий

simple.vue

```
// Parent component
...

<component-n
@eventHandler="$emit('eventHandler', $event)"
...>
</component-n>

...
```

simple.vue

```
// Child component

methods: {
 eventA() {
  this.$emit('eventHandler', 'eventA');
 },
 eventB(value) {
  this.$emit('eventHandler', {
   name: 'eventB',
   data: value
  });
 }
},
```

# Обработка событий

## Event Bus

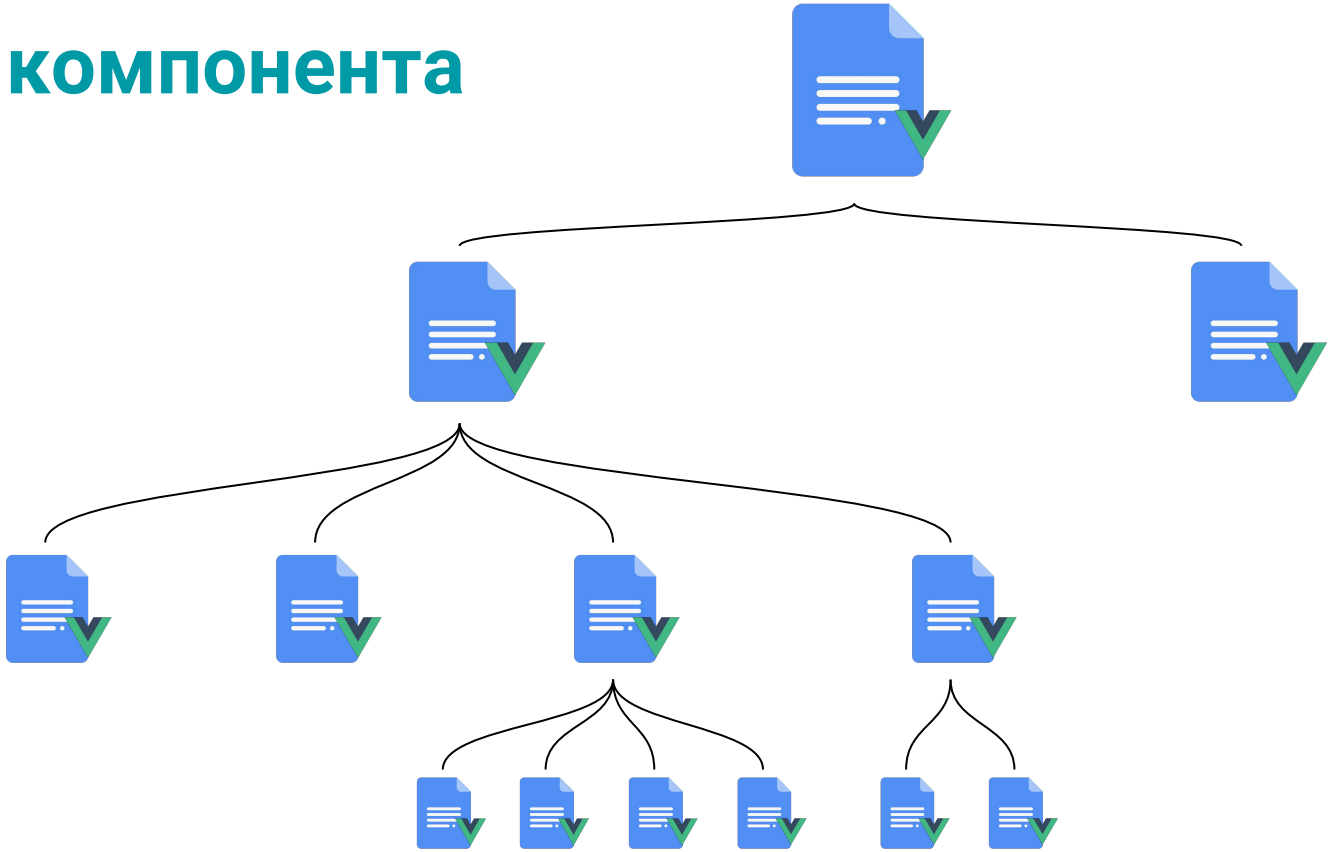
simple.vue

```
// Parent component

created() {
 this.$eventBus.$on('eventA', this.handlerA);
 this.$eventBus.$on('eventB', this.handlerB);
},
beforeDestroy() {
 this.$eventBus.$off('eventA');
 this.$eventBus.$off('eventB');
}
```

simple.vue

```
// Child component

methods: {
 eventA() {
   this.$eventBus.$emit('eventA');
 },
 eventB() {
   this.$eventBus.$emit('eventB');
 },
},
```

# Состояние компонента

Источник: Google

TrueConf

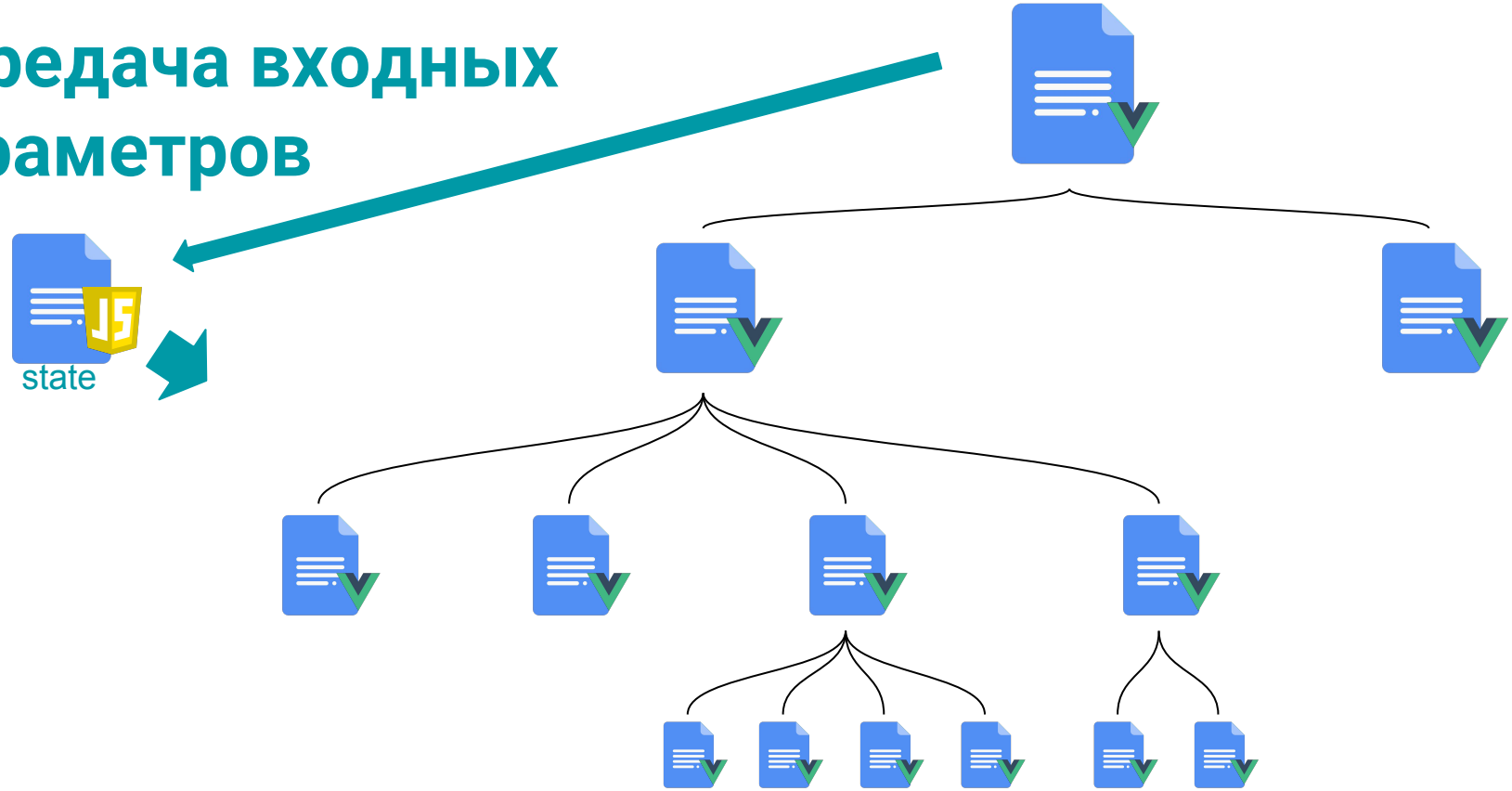
# Состояние компонента

simple.vue

```
class AuthorizationState {
 constructor() {
   // default
 }

 initConfiguration(startScreen, permissions) {
   this.setPermissions(permissions);
   this.setMode(startScreen);
 }

 changeState(value) {
   // some action
 }

 ...
}

const state = new AuthorizationState();
export default state;
```

TrueConf

# Передача входных параметров




Источник: Google

# Передача входных параметров

## Состояние компонента

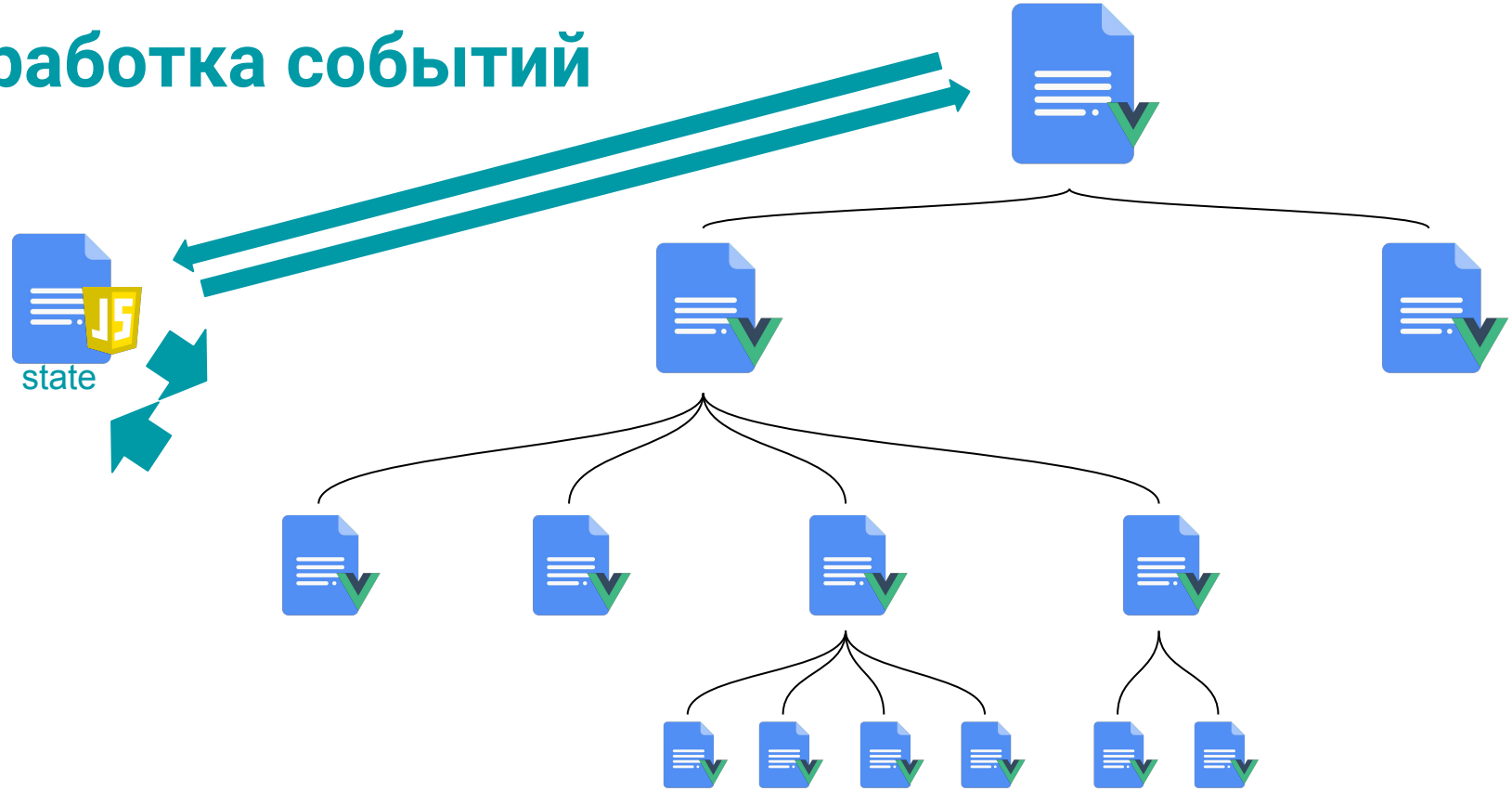
simple.vue

```
// Parent component
import state from './state';

props: {
 permissions: Object,
 startScreen: String,
},

created() {
 state.initConfiguration(this.startScreen,
this.permissions);
}
```

simple.vue

```
// Child component
import state from './state';

computed: {
 permissions() {
   return state.permissions;
 },
 mode() {
   return state.mode;
 },
}
```

# Обработка событий




Источник: Google

# Обработка событий

## Состояние компонента

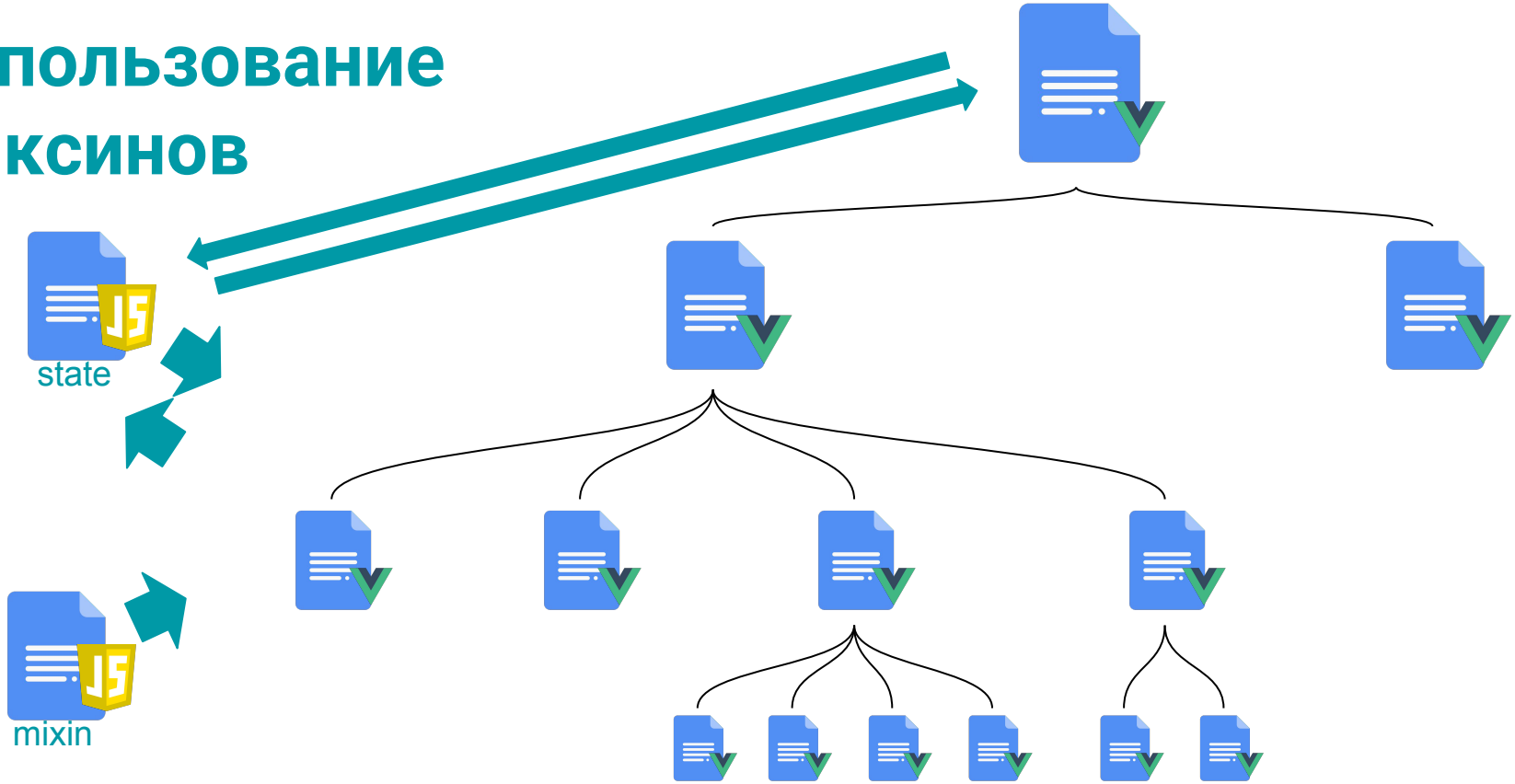
simple.vue

```
// Parent component

watch: {
 'state.event.name': {
  handler() {
   this.$emit(state.event.name,
state.event.data);
  }
},
```

simple.vue

```
// Child component

methods: {
 eventA() {
   state.sendEvent('eventA');
 },
 eventB(value) {
   state.sendEvent('eventB', value);
 },
}
```

# Использование миксинов




Источник: Google

# Использование миксинов

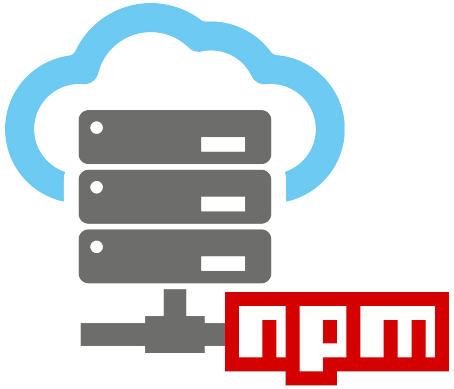
simple.vue

```
// Mixin
export default {
 methods: {
   someHandler(value) {
     // some action
   },
 },
 computed: {
   permissions() {
     return state.permissions;
   },
 },
};
```

simple.vue

```
// Child component
import mixin from './mixin';

...

mixins: [mixin],

...
```

# Подключение библиотеки




Источник: Google

# Подключение библиотеки

## Исходники

библиотека и приложение собираются единовременно

необходимо учитывать особенности сборки библиотеки в каждом приложении, где она используется

## Собранные файлы

можно использовать в любом приложении с любым способом сборки без необходимости вносить доработки в конфигурацию

требует настройки конфигурации

TrueConf

# Сборка библиотеки

simple.vue

```
// Webpack Config


...


externals: {
 'axios': 'axios',
 'dayjs': 'dayjs',
 'lodash': '_'
},


...
```

**Externals**

TrueConf

# СПАСИБО ЗА ВНИМАНИЕ!

Алексей Иванов
Ведущий веб-разработчик

+7 (495) 698-60-66
dev@vcs.su
www.trueconf.ru

TrueConf