



# Vue JS

(с) Сергей Синица, [sin@kubsu.ru](mailto:sin@kubsu.ru)

# HTML/CSS



HTML - структура документа, основными тегами, форма с HTML5 атрибутами (required, placeholder...).

Позиционирование: поток, float, абсолютное, относительное позиционирование и flexbox.

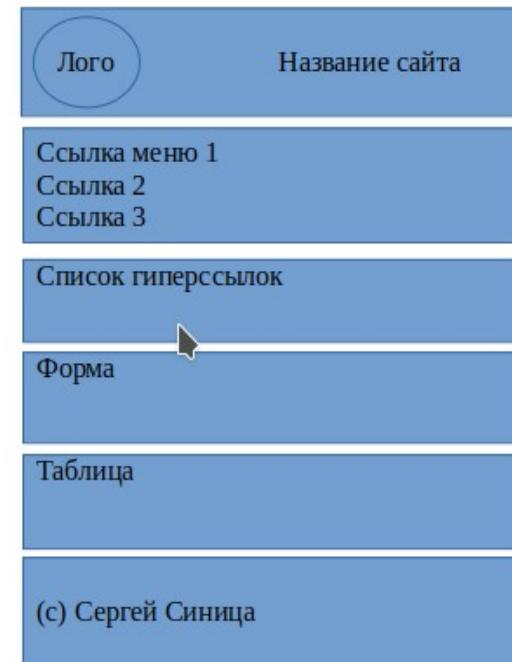
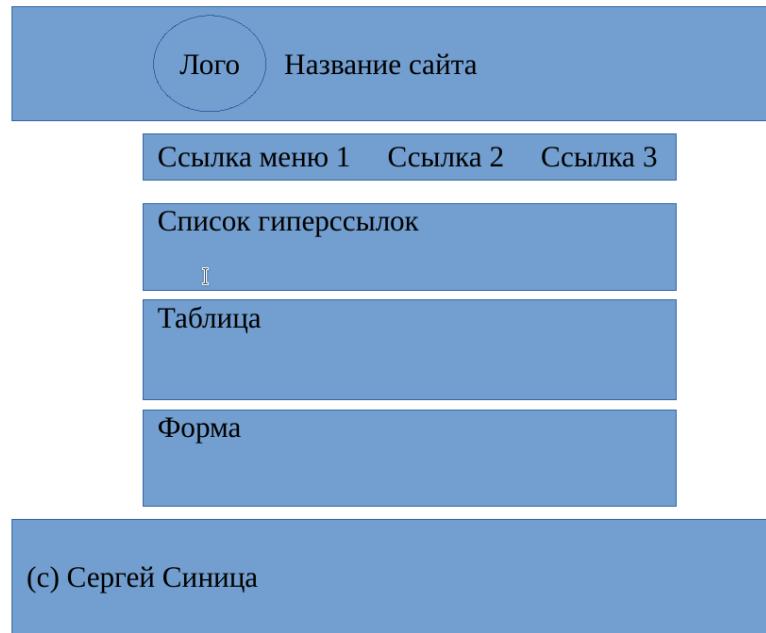
CSS во внешнем файле и inline. Классы CSS.

Основы алгоритма каскада, веса селекторов.

Отладка DOM в браузере.

# Задание 1

Сверстать на Bootstrap + БЭМ адаптивно по макету страницы с формой:



← header

← nav

← footer

Однострочное текстовое поле:  
начальное значение

Текстовое поле email:  
Введите вашу почту

Текстовое поле даты:  
08 / 13 / 2019

Многострочное текстовое поле:  
начальное значение

Список (combobox):  
Метка2

Список (listbox с множественным выбором):  
Метка1  
Метка2  
Метка3

Радиокнопки:  
Подпись радиокнопки 1   Подпись радиокнопки 2

Чекбокс:  
Подпись чекбокса ставим справа  
Отправить

# ES6



Прототипное программирование, JSON, ООП и классы в JS.

Основы ES6.

Подключение модулей.

Promise и fetch.

Async/Await.

## Задание 2

Написать класс для отправки произвольной формы на [formcarry.com](http://formcarry.com) или [slapform.com](http://slapform.com) с помощью fetch. С выводом результата без перезагрузки страницы, проверкой ошибок и отменой длительной операции. Подключить к форме предыдущего задания.

# Введение в Vue JS



Введение в Vue JS и реактивное программирование, компонентный подход.

Vue 2 vs 3.

Способы подключения VueJS.

Инициализация приложения и компонентов.

Работа через cli с VueJS.

Введение в webpack.

Переменные среды в VueJS.

# Vue JS



Релиз в 2014

2.0 в 2016 (виртуальный DOM, SSR, JSX)

3.0 в 2020

Используют Alibaba, Baidu, Xiaomi.

Использует GitLab.

Входит в ядро Laravel.

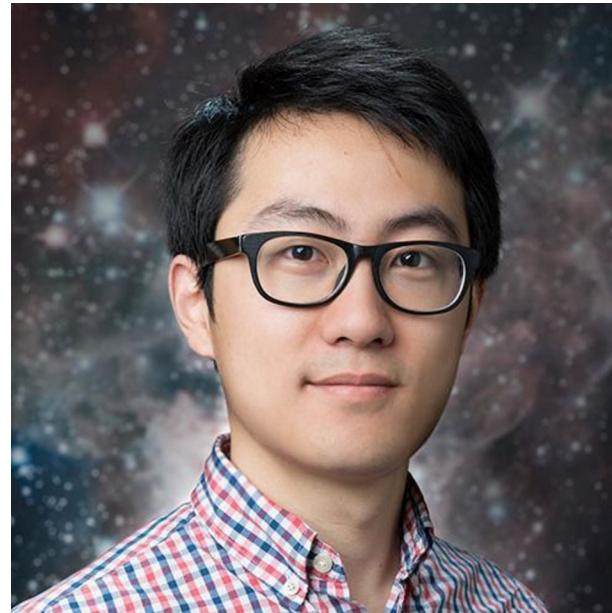
Входит в Битрикс.

<https://github.com/vuejs/vue>

Vue (pronounced /vju:/, like view) is a **progressive framework** for building user interfaces. It is designed from the ground up to be incrementally adoptable, and can easily scale between a library and a framework depending on different use cases. It consists of an approachable core library that focuses on the view layer only, and an ecosystem of supporting libraries that helps you tackle complexity in large Single-Page Applications.

## Browser Compatibility

Vue.js supports all browsers that are [ES5-compliant](#) (IE8 and below are [not supported](#)).

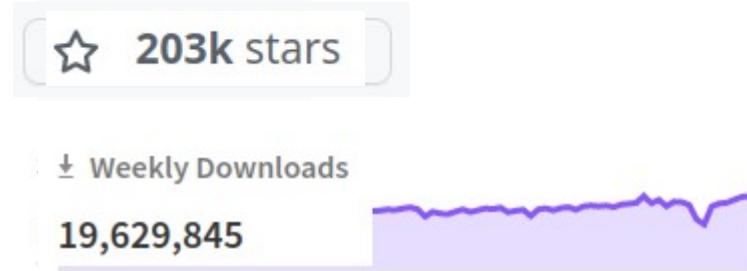


Evan You (ex Google and Meteor Dev Group dev)

Документальный фильм о нем и создании Vue.js:

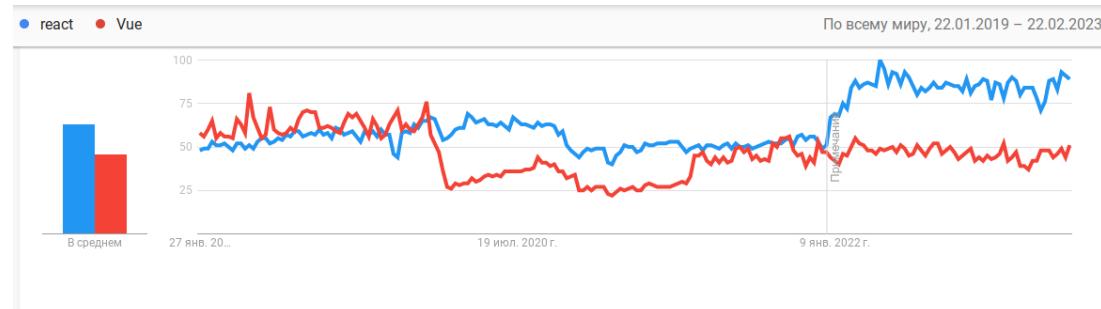
<https://www.youtube.com/watch?v=OrxmtDw4pVI>

# Vue JS vs React



📦 64,479 Dependents

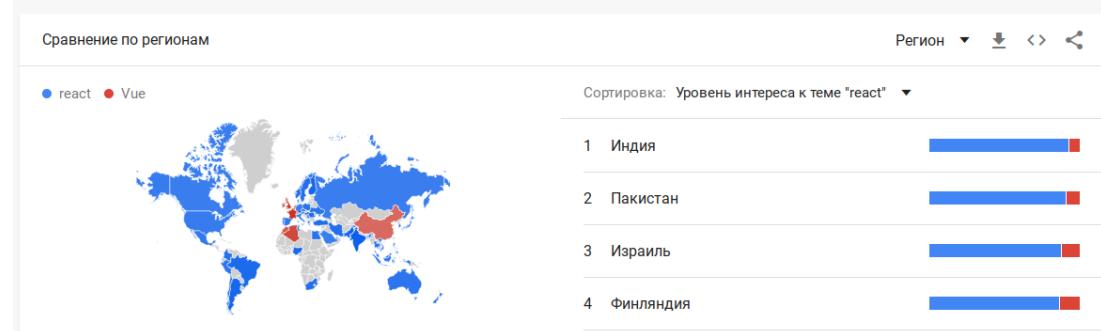
📦 101,339 Dependents



- Популярность (React)

- Простота изучения (Vue)

- Стабильность (React)



# Vue JS ecosystem

Project	Status	Description
<a href="#">vue-router</a>	<a href="#">npm v3.4.9</a>	Single-page application routing
<a href="#">vuex</a>	<a href="#">npm v3.6.0</a>	Large-scale state management
<a href="#">vue-cli</a>	<a href="#">npm v4.5.9</a>	Project scaffolding
<a href="#">vue-loader</a>	<a href="#">npm v15.9.5</a>	Single File Component ( <code>*.vue</code> file) loader for webpack
<a href="#">vue-server-renderer</a>	<a href="#">npm v2.6.12</a>	Server-side rendering support
<a href="#">vue-class-component</a>	<a href="#">npm v7.2.6</a>	TypeScript decorator for a class-based API
<a href="#">vue-rx</a>	<a href="#">npm v6.2.0</a>	RxJS integration
<a href="#">vue-devtools</a>	<a href="#">chrome web store v5.3.3</a>	Browser DevTools extension

RxJS is a library for composing asynchronous and event-based programs by using observable sequences. It provides one core type, the [Observable](#), satellite types (Observer, Schedulers, Subjects) and operators inspired by Array methods ([map](#), [filter](#), [reduce](#), [every](#), etc) to allow handling asynchronous events as collections.

Vite — скафолдинг от Эвана Ю

Pinia — аналог VueX

Vuetify — набор компонентов Material UI

Nuxt JS — фреймворк над фреймворком :)

# Экземпляр Vue

Vue

2

```
import Vue from 'vue'  
import App from './App.vue'  
  
Vue.config.productionTip = false  
  
new Vue({  
  render: h => h(App),  
}).$mount('#app')
```

---

```
// this mixin affects both below instances  
Vue.mixin({ /* ... */ })  
  
const app1 = new Vue({ el: '#app-1' })  
const app2 = new Vue({ el: '#app-2' })
```

Vue

3

```
import { createApp } from 'vue'  
import App from './App.vue'  
import './index.css'  
  
createApp(App).mount('#app')
```

---

```
const app = createApp(App)  
// This configuration effect only 1 instance  
app.mixin(/* ... */)  
app.mount('#app')
```

Глобальная конфигурация усложняет тестирование, но упрощает  
переиспользование конфигурации несколькими приложениями.  
<https://ru.vuejs.org/v2/guide-instance.html>

# Single File Components (\*.vue)



```
<template>
  <p>{{ greeting }} World!</p>
</template>

<script>
module.exports = {
  data: function () {
    return {
      greeting: 'Hello'
    }
  }
}
</script>

<style scoped>
p {
  font-size: 2em;
  text-align: center;
}
</style>
```

## Vue

```
<template>
  <!-- template code -->
</template>
<script>
export default {
  name: "App",
  props: {
    /*...*/
  }
  components: {
    /*...*/
  },
  data(){
    return {
      /*...*/
    }
  },
  computed:{
    /*...*/
  },
  watch:{
    /*...*/
  },
  methods: {
    /*...*/
  },
  created(){
    /*...*/
  },
  mounted(){
    /*...*/
  }
};
</script>
```

# Vue 3 composition API

- Оптимальное использование TypeScript;
- Повторное использование кода;
- Структурирование сложных компонентов;

Setup() позволяет определить, какие данные/функции передаются из компонента обратно в шаблон. Всё, что он возвращает - доступно в шаблоне.



## Vue 3

```
<template>
  <!-- your template code -->
</template>
<script>
export default {
  setup() {
    // more code to write
  }
};
</script>
```

Setup Vue 3 заменяет:

- Components
- Props
- Data
- Methods
- Computed Properties
- Lifecycle methods

Composition API is a set of APIs that allows us to author Vue components using imported functions instead of declaring options. It is an umbrella term that covers the following APIs:

- [Reactivity API](#), e.g. ref() and reactive(), that allows us to directly create reactive state, computed state, and watchers.
- [Lifecycle Hooks](#), e.g. onMounted() and onUnmounted(), that allow us to programmatically hook into the component lifecycle.
- [Dependency Injection](#), i.e. provide() and inject().

# Установка Vue CLI

1. Use Node NVM to install NodeJS LTS, install Vue CLI globally, create project:

```
wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.36.0/install.sh | bash  
nvm install --lts  
npm install -g @vue/cli  
vue create myproject
```

(runs 2 minutes)

# Установка Vue CLI



2. See the result:

README.md

# myproject

## Project setup

...

npm install

...

### Compiles and hot-reloads for development

...

npm run serve

...

### Compiles and minifies for production

...

npm run build

...

### Lints and fixes files

...

npm run lint

...

# NPM и Package.json



```
{  
  "name": "myproject",  
  "version": "0.1.0",  
  "private": true,  
  "scripts": {  
    "serve": "vue-cli-service serve",  
    "build": "vue-cli-service build",  
    "lint": "vue-cli-service lint"  
  },  
  "dependencies": {  
    "core-js": "^3.6.5",  
    "vue": "^2.6.11"  
  },  
  "devDependencies": {  
    "@vue/cli-plugin-babel": "~4.5.0",  
    "@vue/cli-plugin-eslint": "~4.5.0",  
    "@vue/cli-service": "~4.5.0",  
    "babel-eslint": "^10.1.0",  
    "eslint": "6.7.2",  
    "eslint-plugin-vue": "^6.2.2",  
    "vue-template-compiler": "^2.6.11"  
  },  
  "eslintConfig": {  
    "root": true,  
    "env": {  
      "node": true  
    },  
    "extends": [  
      "plugin:vue/essential",  
      "eslint:recommended"  
    ],  
    "parserOptions": {  
      "parser": "babel-eslint"  
    },  
    "rules": {}  
  },  
  "browserslist": [  
    "> 1%",  
    "last 2 versions",  
    "not dead"  
  ]  
}
```

# Webpack



vue inspect > output.js

<https://cli.vuejs.org/ru/guide/webpack.html>

webpack.prod.js 417 Bytes

```
1 const webpack = require('webpack');
2 const merge = require('webpack-merge');
3 const UglifyJSPlugin = require('uglifyjs-webpack-plugin');
4 const common = require('./webpack.common.js');

5 module.exports = merge(common, {
6   plugins: [
7     new UglifyJSPlugin(),
8     new webpack.DefinePlugin({
9       'process.env': {
10         'NODE_ENV': JSON.stringify('production')
11       }
12     })
13   ]
14 });
15 );
```

webpack.dev.js 359 Bytes

```
1 const webpack = require('webpack');
2 const path = require('path');
3 const merge = require('webpack-merge');
4 const common = require('./webpack.common.js');

5 module.exports = merge(common, {
6   plugins: [
7     new webpack.DefinePlugin({
8       'process.env': {
9         'NODE_ENV': JSON.stringify('development')
10       }
11     })
12   ]
13 });
14 );
```

# Webpack



```
1 const path = require('path');
2 const webpack = require('webpack');
3 const CleanWebpackPlugin = require('clean-webpack-plugin');
4 const BundleAnalyzerPlugin = require('webpack-bundle-analyzer').BundleAnalyzerPlugin;
5
6 module.exports = {
7   entry: {
8     store: './src/store.js',
9     dialogPage: './src/dialogPage.jsx',
10    loginRoute: './src/loginRoute.js',
11    checkoutRoute: './src/checkoutRoute.js',
12    preEnrollRoute: './src/preEnrollRoute.js',
13    registerRoute: './src/registerRoute.js',
14    loginBlock: './src/loginBlock.jsx',
15    cartPreview: './src/cartPreview.jsx',
16    productButton: './src/productButton.jsx',
17  },
18  output: {
19    filename: '[name].bundle.js',
20    chunkFilename: '[name].bundle.js',
21    path: path.resolve(__dirname, '../paukshte/web/libraries/paukshte/'),
22  },
23  resolve: {
24    extensions: ['.js', '.jsx', '.json', '*'],
25    modules: [path.resolve(__dirname, 'src'), 'node_modules'],
26  },
27  devtool: 'source-map',
28  module: {
29    rules: [
30      {
31        test: /\.js|\.jsx$/,
32        exclude: /node_modules/,
33        use: {
34          loader: 'babel-loader',
35        },
36      },
37      {
38        test: /\.js|\.jsx$/,
39        exclude: /node_modules/,
40        use: ['babel-loader', 'eslint-loader'],
41      },
42    ],
43  },
44  plugins: [
45    new CleanWebpackPlugin(['../paukshte/web/libraries/paukshte/'], {
46      allowExternal: true,
47    }),
48    new webpack.optimize.CommonsChunkPlugin({
49      name: 'common',
50      minChunks: 2,
51    }),
52    new BundleAnalyzerPlugin({
53      analyzerMode: 'static',
54      reportFilename: './analyze/report.html',
55      openAnalyzer: false,
56    }),
57  ],
58  },
59  },
60  },
61  },
62  },
63  },
64  },
65  },
66  },
67  },
68  },
69  },
70  },
71  },
72  },
73  },
74  },
75  },
76  },
77  },
78  },
79  },
80  },
81  },
82  },
83  },
84  },
85  },
86  },
87  },
88  },
89  },
90  },
91  },
92  },
93  },
94  },
95  },
96  },
97  },
98  },
99  },
100  },
101  },
102  },
103  },
104  },
105  },
106  },
107  },
108  },
109  },
110  },
111  },
112  },
113  },
114  },
115  },
116  },
117  },
118  },
119  },
120  },
121  },
122  },
123  },
124  },
125  },
126  },
127  },
128  },
129  },
130  },
131  },
132  },
133  },
134  },
135  },
136  },
137  },
138  },
139  },
140  },
141  },
142  },
143  },
144  },
145  },
146  },
147  },
148  },
149  },
150  },
151  },
152  },
153  },
154  },
155  },
156  },
157  },
158  },
159  },
160  },
161  },
162  },
163  },
164  },
165  },
166  },
167  },
168  },
169  },
170  },
171  },
172  },
173  },
174  },
175  },
176  },
177  },
178  },
179  },
180  },
181  },
182  },
183  },
184  },
185  },
186  },
187  },
188  },
189  },
190  },
191  },
192  },
193  },
194  },
195  },
196  },
197  },
198  },
199  },
200  },
201  },
202  },
203  },
204  },
205  },
206  },
207  },
208  },
209  },
210  },
211  },
212  },
213  },
214  },
215  },
216  },
217  },
218  },
219  },
220  },
221  },
222  },
223  },
224  },
225  },
226  },
227  },
228  },
229  },
230  },
231  },
232  },
233  },
234  },
235  },
236  },
237  },
238  },
239  },
240  },
241  },
242  },
243  },
244  },
245  },
246  },
247  },
248  },
249  },
250  },
251  },
252  },
253  },
254  },
255  },
256  },
257  },
258  },
259  },
259  },
260  },
261  },
262  },
263  },
264  },
265  },
266  },
267  },
268  },
269  },
270  },
271  },
272  },
273  },
274  },
275  },
276  },
277  },
278  },
279  },
280  },
281  },
282  },
283  },
284  },
285  },
286  },
287  },
288  },
289  },
290  },
291  },
292  },
293  },
294  },
295  },
296  },
297  },
298  },
299  },
299  },
300  },
301  },
302  },
303  },
304  },
305  },
306  },
307  },
308  },
309  },
309  },
310  },
311  },
312  },
313  },
314  },
315  },
316  },
317  },
318  },
319  },
319  },
320  },
321  },
322  },
323  },
324  },
325  },
326  },
327  },
328  },
329  },
329  },
330  },
331  },
332  },
333  },
334  },
335  },
336  },
337  },
338  },
339  },
339  },
340  },
341  },
342  },
343  },
344  },
345  },
346  },
347  },
348  },
349  },
349  },
350  },
351  },
352  },
353  },
354  },
355  },
356  },
357  },
358  },
359  },
359  },
360  },
361  },
362  },
363  },
364  },
365  },
366  },
367  },
368  },
369  },
369  },
370  },
371  },
372  },
373  },
374  },
375  },
376  },
377  },
378  },
379  },
379  },
380  },
381  },
382  },
383  },
384  },
385  },
386  },
387  },
388  },
389  },
389  },
390  },
391  },
392  },
393  },
394  },
395  },
396  },
397  },
398  },
399  },
399  },
400  },
401  },
402  },
403  },
404  },
405  },
406  },
407  },
408  },
409  },
409  },
410  },
411  },
412  },
413  },
414  },
415  },
416  },
417  },
418  },
419  },
419  },
420  },
421  },
422  },
423  },
424  },
425  },
426  },
427  },
428  },
428  },
429  },
430  },
431  },
432  },
433  },
434  },
435  },
436  },
437  },
438  },
439  },
439  },
440  },
441  },
442  },
443  },
444  },
445  },
446  },
447  },
448  },
449  },
449  },
450  },
451  },
452  },
453  },
454  },
455  },
456  },
457  },
458  },
459  },
459  },
460  },
461  },
462  },
463  },
464  },
465  },
466  },
467  },
468  },
469  },
469  },
470  },
471  },
472  },
473  },
474  },
475  },
476  },
477  },
478  },
479  },
479  },
480  },
481  },
482  },
483  },
484  },
485  },
486  },
487  },
488  },
489  },
489  },
490  },
491  },
492  },
493  },
494  },
495  },
496  },
497  },
498  },
499  },
499  },
500  },
501  },
502  },
503  },
504  },
505  },
506  },
507  },
508  },
509  },
509  },
510  },
511  },
512  },
513  },
514  },
515  },
516  },
517  },
518  },
519  },
519  },
520  },
521  },
522  },
523  },
524  },
525  },
526  },
527  },
528  },
529  },
529  },
530  },
531  },
532  },
533  },
534  },
535  },
536  },
537  },
538  },
539  },
539  },
540  },
541  },
542  },
543  },
544  },
545  },
546  },
547  },
548  },
549  },
549  },
550  },
551  },
552  },
553  },
554  },
555  },
556  },
557  },
558  },
559  },
559  },
560  },
561  },
562  },
563  },
564  },
565  },
566  },
567  },
568  },
569  },
569  },
570  },
571  },
572  },
573  },
574  },
575  },
576  },
577  },
578  },
579  },
579  },
580  },
581  },
582  },
583  },
584  },
585  },
586  },
587  },
588  },
589  },
589  },
590  },
591  },
592  },
593  },
594  },
595  },
596  },
597  },
598  },
599  },
599  },
600  },
601  },
602  },
603  },
604  },
605  },
606  },
607  },
608  },
609  },
609  },
610  },
611  },
612  },
613  },
614  },
615  },
616  },
617  },
618  },
619  },
619  },
620  },
621  },
622  },
623  },
624  },
625  },
626  },
627  },
628  },
629  },
629  },
630  },
631  },
632  },
633  },
634  },
635  },
636  },
637  },
638  },
639  },
639  },
640  },
641  },
642  },
643  },
644  },
645  },
646  },
647  },
648  },
649  },
649  },
650  },
651  },
652  },
653  },
654  },
655  },
656  },
657  },
658  },
659  },
659  },
660  },
661  },
662  },
663  },
664  },
665  },
666  },
667  },
668  },
669  },
669  },
670  },
671  },
672  },
673  },
674  },
675  },
676  },
677  },
678  },
679  },
679  },
680  },
681  },
682  },
683  },
684  },
685  },
686  },
687  },
688  },
689  },
689  },
690  },
691  },
692  },
693  },
694  },
695  },
696  },
697  },
698  },
699  },
699  },
700  },
701  },
702  },
703  },
704  },
705  },
706  },
707  },
708  },
709  },
709  },
710  },
711  },
712  },
713  },
714  },
715  },
716  },
717  },
718  },
719  },
719  },
720  },
721  },
722  },
723  },
724  },
725  },
726  },
727  },
728  },
729  },
729  },
730  },
731  },
732  },
733  },
734  },
735  },
736  },
737  },
738  },
739  },
739  },
740  },
741  },
742  },
743  },
744  },
745  },
746  },
747  },
748  },
749  },
749  },
750  },
751  },
752  },
753  },
754  },
755  },
756  },
757  },
758  },
759  },
759  },
760  },
761  },
762  },
763  },
764  },
765  },
766  },
767  },
768  },
769  },
769  },
770  },
771  },
772  },
773  },
774  },
775  },
776  },
777  },
778  },
779  },
779  },
780  },
781  },
782  },
783  },
784  },
785  },
786  },
787  },
788  },
789  },
789  },
790  },
791  },
792  },
793  },
794  },
795  },
796  },
797  },
798  },
799  },
799  },
800  },
801  },
802  },
803  },
804  },
805  },
806  },
807  },
808  },
809  },
809  },
810  },
811  },
812  },
813  },
814  },
815  },
816  },
817  },
818  },
819  },
819  },
820  },
821  },
822  },
823  },
824  },
825  },
826  },
827  },
828  },
829  },
829  },
830  },
831  },
832  },
833  },
834  },
835  },
836  },
837  },
838  },
839  },
839  },
840  },
841  },
842  },
843  },
844  },
845  },
846  },
847  },
848  },
849  },
849  },
850  },
851  },
852  },
853  },
854  },
855  },
856  },
857  },
858  },
859  },
859  },
860  },
861  },
862  },
863  },
864  },
865  },
866  },
867  },
868  },
869  },
869  },
870  },
871  },
872  },
873  },
874  },
875  },
876  },
877  },
878  },
879  },
879  },
880  },
881  },
882  },
883  },
884  },
885  },
886  },
887  },
888  },
889  },
889  },
890  },
891  },
892  },
893  },
894  },
895  },
896  },
897  },
898  },
899  },
899  },
900  },
901  },
902  },
903  },
904  },
905  },
906  },
907  },
908  },
909  },
909  },
910  },
911  },
912  },
913  },
914  },
915  },
916  },
917  },
918  },
919  },
919  },
920  },
921  },
922  },
923  },
924  },
925  },
926  },
927  },
928  },
929  },
929  },
930  },
931  },
932  },
933  },
934  },
935  },
936  },
937  },
938  },
939  },
939  },
940  },
941  },
942  },
943  },
944  },
945  },
946  },
947  },
948  },
949  },
949  },
950  },
951  },
952  },
953  },
954  },
955  },
956  },
957  },
958  },
959  },
959  },
960  },
961  },
962  },
963  },
964  },
965  },
966  },
967  },
968  },
969  },
969  },
970  },
971  },
972  },
973  },
974  },
975  },
976  },
977  },
978  },
979  },
979  },
980  },
981  },
982  },
983  },
984  },
985  },
986  },
987  },
988  },
989  },
989  },
990  },
991  },
992  },
993  },
994  },
995  },
996  },
997  },
998  },
999  },
999  },
1000  },
```

vue inspect > output.js

<https://cli.vuejs.org/ru/guide/webpack.html>

# Переменные среды Vue CLI

Переменные окружения можно указать в специальных файлах в корне проекта:

```
.env          # загружается во всех случаях  
.env.local    # загружается во всех случаях, игнорируется git  
.env.[mode]   # загружается только в указанном режиме работы  
.env.[mode].local # загружается только в указанном режиме работы, игнорируется git
```

Такой `.env` файл просто содержит пары ключ=значение требуемых переменных окружения:

```
FOO=bar  
VUE_APP_NOT_SECRET_CODE=some_value
```

## ВНИМАНИЕ

Не храните никаких секретов (например, приватных ключей API) в приложении!

Так как переменные окружения внедряются в сборку, то любой желающий сможет увидеть их, изучив файлы сборки приложения.

Можно получить доступ к переменным окружения из кода приложения:

```
console.log(process.env.VUE_APP_NOT_SECRET_CODE)
```



Режимы работы — важная часть проектов Vue CLI. По умолчанию есть три режима работы:

- `development` используется `vue-cli-service serve`
- `test` используется `vue-cli-service test:unit`
- `production` используется `vue-cli-service build` И `vue-cli-service test:e2e`

Можно переопределять используемый для команды режим по умолчанию опцией `--mode`.

Например, если необходимо использовать переменные для разработки в команде сборки:

```
vue-cli-service build --mode development
```

При запуске `vue-cli-service` загрузит переменные окружения из всех соответствующих файлов. Если в них не указана переменная `NODE_ENV`, то она установится соответствующим образом. Например, `NODE_ENV` будет установлена в `"production"` в режиме `production`, `"test"` в режиме `test`, а по умолчанию `"development"` в противном случае.

Затем `NODE_ENV` определяет основной режим работы приложения — разработка, `production` или тестирование — и, следовательно, какую конфигурацию webpack требуется создавать.

Например, для `NODE_ENV=test` Vue CLI создаёт конфигурацию webpack, которая оптимизирована и предназначена для модульных тестов. В ней не обрабатываются изображения и другие ресурсы, которые не требуются для модульных тестов.

Аналогично, при `NODE_ENV=development` создаётся конфигурация webpack, которая включает горячую перезагрузку модулей (HMR), не добавляет хэши в имена файлов ресурсов и не создаёт сборку для вендоров, чтобы обеспечить быструю пересборку при запуске сервера разработки.

При запуске `vue-cli-service build`, значение всегда должно быть `NODE_ENV=production` для получения приложения готового к публикации, независимо от окружения куда будет осуществляться публиковаться.

# Browserslist, БД caniuse

"> 1%" is a stats from caniuse, browser usage DB is updated via

```
npx browserslist@latest --update-db
```

caniuse-lite has been successfully updated

Target browser changes:

```
- and_chr 85
+ and_chr 86
- and_ff 79
+ and_ff 82
- chrome 84
- firefox 80
+ firefox 82
- ios_saf 13.3
- opera 70
+ opera 72
```

3. Browserslist is used to add polyfills automatically.

By default, it passes useBuiltIns: 'usage' to @babel/preset-env which automatically detects the polyfills needed based on the language features used in your source code. This ensures only the minimum amount of polyfills are included in your final bundle. However, this also means if one of your dependencies has specific requirements on polyfills, by default Babel won't be able to detect it.

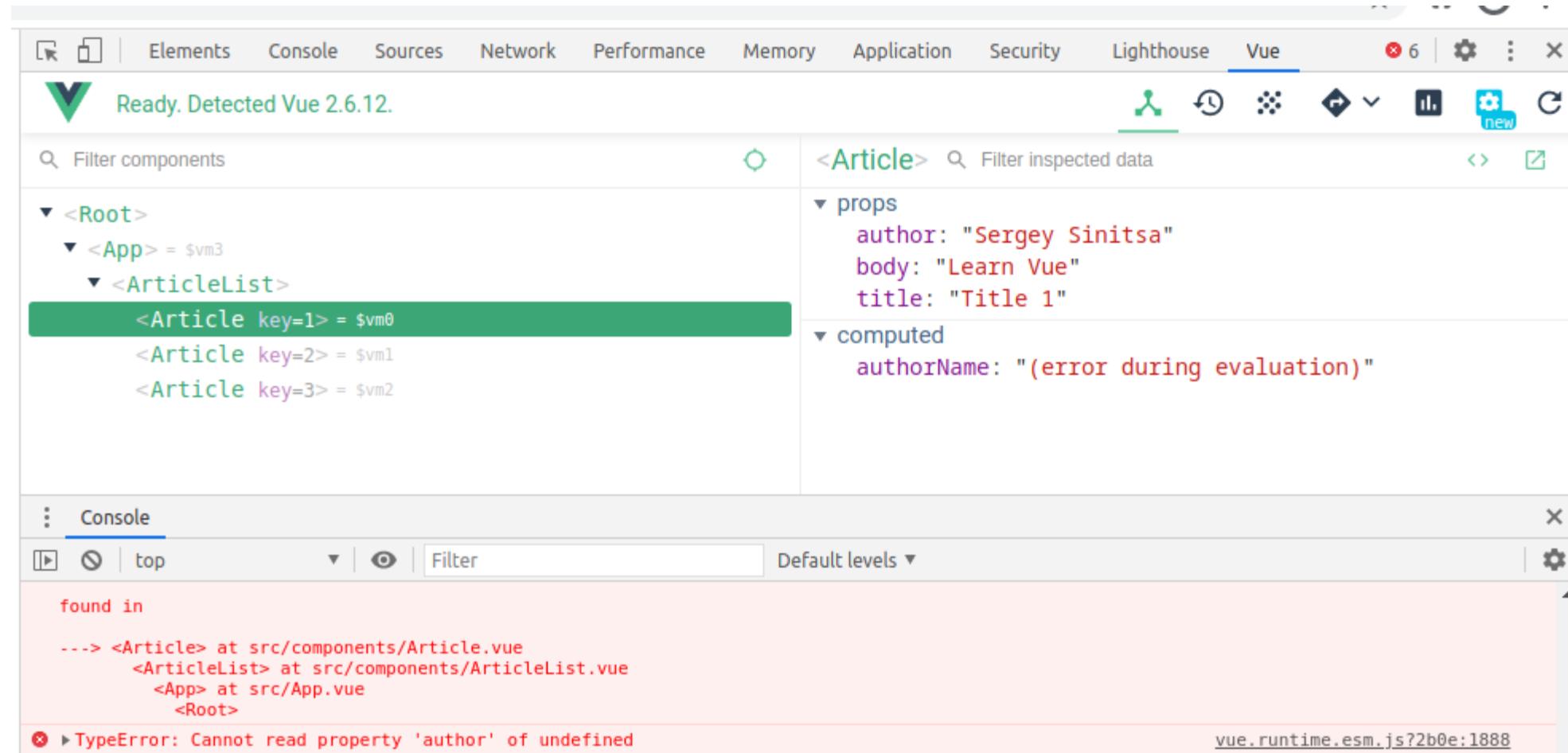
"last 2 versions": for all browsers, even Opera Mini (100 million users in Africa!)

"not dead": dead means browsers without official support or updates for 24 months. Right now it is IE 10, IE\_Mob 11, BlackBerry 10, BlackBerry 7, Samsung 4 and OperaMobile 12.1.

"defaults": Browserslist's default browsers (> 0.5%, last 2 versions, Firefox ESR, not dead).

See <https://cli.vuejs.org/guide/browser-compatibility.html#usebuiltins-usage>

# Vue DevTools



The screenshot shows the Vue DevTools extension for a browser. The top navigation bar includes tabs for Elements, Console, Sources, Network, Performance, Memory, Application, Security, Lighthouse, and **Vue**. The Vue tab is active, indicated by a green underline.

The left panel displays a component tree:

- <Root>
  - <App> = \$vm3
    - <ArticleList>
      - <Article key=1> = \$vm0
      - <Article key=2> = \$vm1
      - <Article key=3> = \$vm2

The <Article key=1> component is currently selected, highlighted with a green background. The right panel shows its state:

  - props**:
    - author: "Sergey Sinitsa"
    - body: "Learn Vue"
    - title: "Title 1"
  - computed**:
    - authorName: "(error during evaluation)"

The bottom panel is the **Console** tab, which includes a **Filter** input field and a **Default levels** dropdown. It lists errors and warnings:

  - found in
    - > <Article> at src/components/Article.vue
    - <ArticleList> at src/components/ArticleList.vue
    - <App> at src/App.vue
    - <Root>
  - ✖ TypeError: Cannot read property 'author' of undefined

A red error message is also visible in the bottom right corner: `vue.runtime.esm.js?2b0e:1888`.

<https://github.com/vuejs/vue-devtools>

# Задание 3



Установить Vue CLI и DevTools.  
Сгенерировать пример приложения на  
Vue 2.

# Изолированные стили компонента



```
src > components > ▼ Article.vue > {} "Article.vue"
1  <template>
2    <div>
3      <h3>{{ title }}</h3>
4      <div>
5        | {{ body }}
6        | </div>
7      </div>
8    </template>
9
10 <script>
11 export default {
12   name: 'Article',
13   props: {
14     title: String,
15     body: String
16   }
17 }
18 </script>
19
20 <!-- Add "scoped" attribute to limit CSS to this component only -->
21 <style scoped>
22 h3 {
23   color: #999;
24 }
25 </style>
26
```

# Welcome to Your Vue.js App

Title 1

Learn Vue

Title 2

Learn about single-file components

Title 3

Fall in love

```
<!DOCTYPE html>
<html lang="en"> [event]
  > <head>[...]</head>
  ><body>
    > <noscript>[...]</noscript>
    ><div id="app">
      ><div class="hello" data-v-44de4ab4="">
        >h1 data-v-44de4ab4="">Welcome to Your Vue.js App</h1>
      ><ul data-v-44de4ab4="">
        ><div data-v-72bad814="" data-v-44de4ab4="">[...]</div>
        ><div data-v-72bad814="" data-v-44de4ab4="">[...]</div>
        ><div data-v-72bad814="" data-v-44de4ab4="">
          ><h3 data-v-72bad814="">Title 3</h3>
          ><div data-v-72bad814="">Fall in love</div>
        </div>
      </ul>
    </div>
  </body>
</html>
```

The screenshot shows the Chrome DevTools Elements tab with the DOM tree on the left and the Style panel on the right. A blue bar highlights the `<div data-v-72bad814="">Title 3</h3>` element in the tree.

**Element Styles:**

- element { } inline
- h3[data-v-72bad814] { color: #999; } inline:2
- Inherited from ul
- ul[data-v-44de4ab4] { list-style-type: none; } inline:5
- Inherited from div#app
- #app { font-family: Avenir, Helvetica, Arial, sans-serif; color: #2c3e50; } inline:2

**Computed:**

- color: #2c3e50

**Event Listener:**

- click

**Take a New Screenshot**

# Синтаксис шаблонов



Vue

2

```
<template>
  <div>
    <h1>{{ msg }}</h1>→
    <button @click="count++">count is: {{ count }}</button>
    <p>Edit <code>components/HelloWorld.vue</code> to test hot module replacement.</p>
  </div>
</template>
```

Ошибка если более одного root template

```
[vue/no-multiple-template-root]
The template root requires exactly one element.
```

Vue

3

```
<template>
  <h1>{{ msg }}</h1>
  <button @click="count++">count is: {{ count }}</button>
  <p>Edit <code>components/HelloWorld.vue</code> to test hot module replacement.</p>
</template>
```

syntax.html

<https://ru.vuejs.org/v2/guide/>

# Условная отрисовка

```
<h1 v-if="awesome">Vue восхитителен!</h1>
<h1 v-else>0, нет 😞</h1>

<template v-if="ok">
  <h1>Заголовок</h1>
  <p>Абзац 1</p>
  <p>Абзац 2</p>
</template>
```

```
<div v-if="type === 'A'>
  A
</div>
<div v-else-if="type === 'B'">
  B
</div>
<div v-else-if="type === 'C'">
  C
</div>
<div v-else>
  Не A/B/C
</div>
```

<https://ru.vuejs.org/v2/guide/conditional.html>

При вводе пользователем и переключении loginType input не пересоздается, меняется только placeholder:

```
<template v-if="loginType === 'username'">
  <label>Имя пользователя</label>
  <input placeholder="Введите имя пользователя">
</template>
<template v-else>
  <label>Email</label>
  <input placeholder="Введите адрес email">
</template>
```

Используем key если такое поведение не желательно:

```
<template v-if="loginType === 'username'">
  <label>Имя пользователя</label>
  <input placeholder="Введите имя пользователя" key="username-input">
</template>
<template v-else>
  <label>Email</label>
  <input placeholder="Введите адрес email" key="email-input">
</template>
```

Элемент всегда в DOM и скрывается через CSS  
display:none:

```
<h1 v-show="ok">Привет!</h1>
```

# Отрисовка списков



```
<ul id="example-1">
  <li v-for="item in items" :key="item.message">
    {{ item.message }}
  </li>
</ul>
```

```
<ul id="example-2">
  <li v-for="(item, index) in items">
    {{ parentMessage }} - {{ index }} - {{ item.message }}
  </li>
</ul>
```

```
<ul id="v-for-object" class="demo">
  <li v-for="value in object">
    {{ value }}
  </li>
</ul>
```

```
new Vue({
  el: '#v-for-object',
  data: {
    object: {
      title: 'How to do lists in Vue',
      author: 'Jane Doe',
      publishedAt: '2016-04-10'
    }
  }
})
```

# Вычисляемые свойства



Vue  
2

```
<template>
<div id="app">
  <p>Upper: {{ upperName }} out of {{ name }}</p>
</div>
</template>
<script>
export default {
  data(){
    return {
      name: "author",
    }
  },
  computed: {
    upperName(){
      return this.name.toUpperCase() + "VUE_3";
    }
  };
</script>
```

Vue  
3

```
<template>
<div id="app">
  <p>Upper: {{ upperName }} out of {{ name }}</p>
</div>
</template>
<script>
import { ref, computed } from "vue";
export default {
  setup() {
    const name = ref("author");

    const upperName = computed(() => {
      return name.value.toUpperCase() + "VUE_3";
    });
    return { name, upperName };
  }
};
</script>
```

<https://ru.vuejs.org/v2/guide/computed.html>  
04-computed-property

# Связывание CSS-классов и inline стилей

```
<div v-bind:class="{ active: isActive }"></div>
```

```
<div v-bind:style="{ color: activeColor, fontSize: fontSize + 'px' }"></div>
```

```
<div  
  class="static"  
  v-bind:class="{ active: isActive, 'te  
></div>
```

```
data: {  
  activeColor: 'red',  
  fontSize: 30  
}
```

При использовании таких данных:

```
data: {  
  isActive: true,  
  hasError: false  
}
```

Аналогично обычным,  
можно использовать вычисляемые  
свойства

В результате получится:

```
<div class="static active"></div>
```

<https://ru.vuejs.org/v2/guide/class-and-style.html>

05-bind-css-class-style

# Приложение

```
src > ▼ App.vue > {} "App.vue"
1  <template>
2  | <div id="app">
3  |   <ArticleList msg="Welcome to Your Vue.js App"/>
4  | </div>
5  </template>
6
7  <script>
8  import ArticleList from './components/ArticleList.vue'
9
10 export default {
11   name: 'App',
12   components: {
13     ArticleList
14   }
15 }
16 </script>
17
18 <style>
19 #app {
20   font-family: Avenir, Helvetica, Arial, sans-serif;
21   color: ■#2c3e50;
22   margin-top: 60px;
23 }
24 </style>
```

# Шаблон, условный рендеринг, цикл

src > components > ArticleList.vue > {} "ArticleList.vue" > script > default > data

```
1 <template>
2   <div class="hello">
3     <h1>{{ msg }}</h1>
4     <ul v-if="articles.length">
5       <Article
6         v-for="article in articles"
7           :key="article.id"
8           :title="article.title"
9           :body="article.body"
10        />
11      </ul>
12      <p v-else>
13        No articles in the list.
14      </p>
15    </div>
16  </template>
```

```
18 <script>
19   import Article from './Article.vue'
20
21   export default {
22     name: 'ArticleList',
23     props: {
24       msg: String
25     },
26     components: {
27       Article
28     },
29     data () {
30       return {
31         articles: [
32           {
33             id: 1,
34             title: 'Title 1',
35             body: 'Learn Vue'
36           },
37           {
38             id: 2,
39             title: 'Title 2',
40             body: 'Learn about single-file components'
41           },
42           {
43             id: 3,
44             title: 'Title 3',
45             body: 'Fall in love'
46           }
47         ]
48       }
49     },
50   }
51 </script>
```

Метод data() возвращает  
доступные в шаблоне  
данные

```
54   <style scoped>
55     h3 {
56       margin: 40px 0 0;
57     }
58     ul {
59       list-style-type: none;
60       padding: 0;
61     }
62     li {
63       display: inline-block;
64       margin: 0 10px;
65     }
66     a {
67       color: #42b983;
68     }
69   </style>
70
```

# Задание 4



Список статей в компоненте ArticleList выводит в цикле компоненты Article.

Компонент Article пишем сами, он выводит непустые поля статьи.

У статьи добавить свойство, опубликована она или нет. Прикрепить класс CSS в зависимости от свойства и по классу покрасить неопубликованные статьи в красный цвет.

У статьи сделать computed свойство, вычисляющее начертание шрифта имени автора в зависимости от того, опубликована статья или нет.

В статье использовать инлайн стили. Имя автора вывести заглавными буквами с помощью вычисляемого свойства

# Обработка событий

Vue 2

```
<template>
  <div>
    <div>Amount: {{ amount }}</div>
    <button @click="increaseAmount()">Increase Capacity</button>
  </div>
</template>

<script>
export default {
  data(){
    return {
      amount: 3,
    }
  },
  methods: {
    increaseAmount() {
      this.amount+=1
    }
  }
};
</script>
```

Vue 3

```
<template>
  <div>
    <div>Amount: {{ amount }}</div>
    <button @click="increaseAmount()">Increase Capacity</button>
  </div>
</template>

<script>
import { ref } from "vue";
export default {
  setup() {
    const amount = ref(3);
    function increaseAmount() {
      amount.value+=1
    }
    return { amount, increaseAmount };
  }
};
</script>
```



# Слежение

Vue 2

```
<template>
  <div>
    <input type="text" v-model="name" />
  </div>
</template>

<script>
export default {
  data(){
    return {
      name: null,
    }
  },
  watch: {
    name(newVal, oldVal){
      console.log(` ${newVal} ${oldVal}`);
    }
  }
};
</script>
```

<https://ru.vuejs.org/v2/guide/computed.html>  
07-watch



Vue 3

```
<template>
  <div>
    <input type="text" v-model="name" />
  </div>
</template>

<script>
import { ref, watchEffect } from "vue";
export default {
  setup() {
    const name = ref('');
    watchEffect(() => {
      console.log(name.value);
    })
    return { name };
  }
};
</script>
```

```
<template>
  <div>
    <input type="text" v-model="name" />
  </div>
</template>

<script>
import { ref, watch } from "vue";
export default {
  setup() {
    const name = ref('');
    watch(name, (newVal, oldVal) => {
      console.log(` ${newVal} ${oldVal}`);
    })
    return { name };
  }
};
</script>
```

# Работа с формами, дву направленное связывание



```
61  methods: {
62    addArticle: function(article) {
63      let newArticle = {
64        id: this.articles.length + 1,
65        ...article
66      }
67      this.articles.push(newArticle);
68      console.log(article.title);
69    }
70 }
```

```
1  <template>
2    <div class="hello">
3      <h1>{{ msg }}</h1>
4      <div v-if="articles.length">
5        <Article
6          v-for="article in articles"
7            :key="article.id"
8            :title="article.title"
9            :body="article.body"
10           :author="article.author"
11           :isPublished="article.isPublished"
12         />
13       </div>
14       <p v-else>
15         No articles in the list.
16       </p>
17       <ArticleForm title="New title" body="" author="" isPublished="" 
18           v-on:add-article="addArticle" />
19     </div>
20   </template>
21
22   <script>
23   import Article from './Article.vue'
24   import ArticleForm from './ArticleForm.vue'
```

<https://ru.vuejs.org/v2/guide/forms.html>  
08-form-submit-article

```
1 <template>
2   <div>
3     <h2>Add new article</h2>
4     <form>
5       <div><input type="text" v-model="article.title" placeholder="Title"></div>
6       <div><input type="text" v-model="article.author" placeholder="Author"></div>
7       <div><textarea v-model="article.body" placeholder="Body"></textarea></div>
8       <div><label><input type="checkbox" v-model="article.isPublished"> Published</label></div>
9       <div><input type="submit" value="Add" v-on:click.prevent="$emit('add-article', article)" />
10      </form>
11    </div>
12  </template>
13
14 <script>
15 export default {
16   name: "ArticleForm",
17   props: {
18     title: String,
19     body: String,
20     author: String,
21     isPublished: Boolean,
22   },
23   data() {
24     return {
25       article: {
26         title: this.title,
27         body: this.body,
28         author: this.author,
29         published: this.isPublished
30       }
31     }
32   }
33 };
```

# Двухстороннее связывание более одного раза

```
<input v-model="property" />
```

Или

```
<input  
  :value="property"  
  @input="property = $event.target.value"  
/>  
  
<template>  
  <div id="app">  
    <InviteForm v-model:name="inviteName" v-model:email="inviteEmail" />  
  </div>  
  <div>{{ inviteName }}</div>  
  <div>{{ inviteEmail }}</div>  
  </div>  
</template>  
  
<script>  
import InviteForm from './components/InviteForm.vue';  
import { ref } from "vue";  
  
export default {  
  name: "App",  
  components: {  
    InviteForm,  
  },  
  setup() {  
    const inviteName = ref("name");  
    const inviteEmail = ref("invite");  
    return {  
      inviteName,  
      inviteEmail,  
    };  
  },  
};  
</script>
```

InviteForm.vue:

```
<template>  
  <form>  
    <div>  
      <label for="name">Name</label>  
      <input type="text" :value="name" @input="updateName($event.target.value)" />  
    </div>  
    <div>  
      <label for="email">Email</label>  
      <input type="email" :value="email" @input="updateEmail($event.target.value)" />  
    </div>  
  </form>  
</template>  
<script>  
export default {  
  props: {  
    name: String,  
    email: String,  
  },  
  setup(props, { emit }) {  
    const updateName = (value) => {  
      emit("update:name", value);  
    };  
    const updateEmail = (value) => {  
      emit("update:email", value);  
    };  
    return { updateName, updateEmail };  
  },  
};  
</script>
```

Vue 2

```
<template>
<div id="app">
  
  <HelloWorld msg="Welcome to Your Vue.js App" />
  {{ new Date() | formatUnix}}
</div>
</template>

<script>
import HelloWorld from './components/HelloWorld.vue';
import moment from "moment";

export default {
  name: "App",
  components: {
    HelloWorld,
  },
  filters: {
    formatUnix(value) {
      if (value) {
        return moment(value).format("DD/MM/YYYY");
      }
    },
  },
}
</script>

<style>
#app {
  font-family: Avenir, Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

# Vue Filter убрали

## DateFormat.js

```
import moment from "moment";
const format = function formatUnix(value) {
  if (value) {
    return moment(value).format("DD/MM/YYYY");
  }
};
export default format;
```

```
<template>
<div id="app">
  {{formatUnix(new Date())}}
</div>
</template>

<script>
import formatUnix from "./Utils/DateFormat.js";

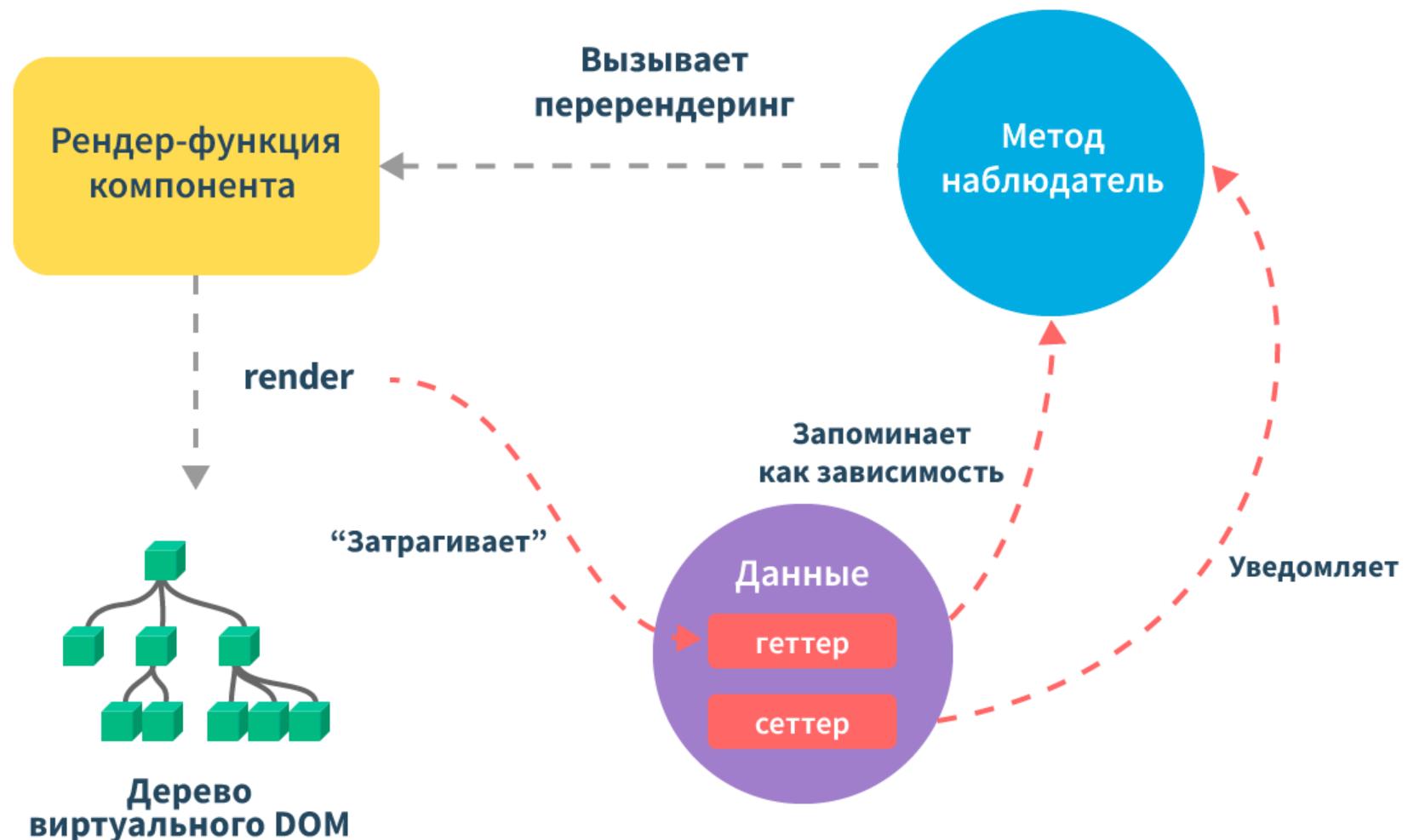
export default {
  name: "App",
  components: {},
  setup() {
    return {
      formatUnix
    };
  }
};
</script>

<style>
#app {
  font-family: Avenir, Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

Vue 3



# Реактивность



# Реактивные ссылки



Vue 2

```
<template>
  <div id="app">
    <div>Name: {{ name }}</div>
  </div>
</template>

<script>
export default {
  name: "App",
  data(){
    return {
      name: "Name"
    }
  }
};
</script>
```

Vue 3

```
<template>
  <div>Name: {{ name }}</div>
</template>

<script>
import { ref } from "vue";
export default {
  setup() {
    const name = ref('Name');
    return { name };
  }
};
</script>
```

# Жизненный цикл компонентов



Vue 2

beforeCreate()  
created()  
beforeMount()  
mounted()  
beforeUpdate()  
updated()  
beforeDestroy()

Vue 3

- beforeDestroy() стал beforeUnmount()  
- destroyed() стал unmounted()  
- onRenderTracked вызывается когда  
реактивная зависимость в первый раз  
используется из рендер функции.  
- onRenderTriggered() вызывается конда  
запускается новый рендер, что позволяет  
посмотреть что и когда перендерит  
компонент.

```
import { onBeforeMount, onMounted } from "vue";

export default {
  setup(){
    onBeforeMount(() => {
      console.log("Before mount");
    });
    onMounted(() => {
      console.log("Mounted");
    });
  }
}
```

<https://ru.vuejs.org/v2/guide/components.html>

<https://ru.vuejs.org/v2/api/#Опции——хуки-жизненного-цикла>

<https://ru.vuejs.org/v2/guide-instance.html#Диаграмма-жизненного-цикла>

09-beforemount-fetch



```
26 export default {
27   name: 'ArticleList',
28   props: {
29     msg: String
30   },
31   components: {
32     Article, ArticleForm
33   },
34   data () {
35     return {
36       articles: []
37     }
38   },
39   methods: {
40     addArticle: function(article) {
41       let newArticle = {
42         id: this.articles.length + 1,
43         ...article
44       }
45       this.articles.push(newArticle);
46       console.log(article.title);
47     }
48   },
49   beforeMount: function() {
50     fetch('/articles.json')
51       .then(response => response.json())
52       .then(articles => this.articles = articles);
53     console.log('Fetch data');
54   }
55 }
```

# Межкомпонентное взаимодействие. Общие функции (mixin и т.д.).



## Регистрация компонентов (глобальная и локальная)

Локальная регистрация:

```
import ComponentA from './ComponentA'  
import ComponentC from './ComponentC'  
export default {  
  components: {  
    ComponentA,  
    ComponentC  
  },  
  // ...  
}
```

Теперь оба компонента ComponentA и ComponentC могут быть использованы в шаблоне ComponentB.

Глобальная:

```
import CustomComponent from './components/CustomComponent.vue'  
Vue.component('CustomComponent', CustomComponent);
```

Теперь CustomComponent можно использовать в любых шаблонах.

Глобальная регистрация должна происходить до создания корневого экземпляра Vue (с помощью new Vue)

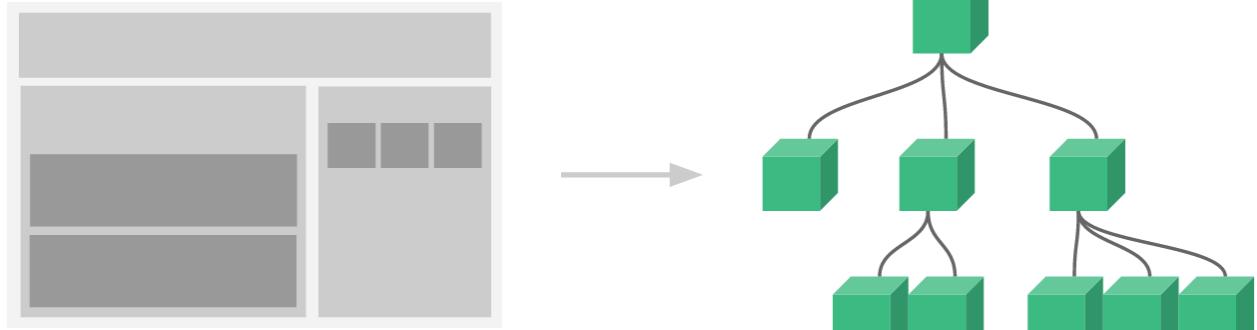
<https://ru.vuejs.org/v2/guide/components.html>

<https://ru.vuejs.org/v2/guide/components-registration.html>

10-global-component

...

<https://ru.vuejs.org/v2/guide/mixins.html>



# Задание 5



Установить Vue DevTools в Chrome или FF и VS Code с плагином Vetur.

Реализовать форму добавления статей.

При клике на кнопку в статье переключать опубликована она или нет.

С помощью слежения Watch перехватывать изменение статуса опубликованности статьи и писать в консоль смену статуса.

В хуке BeforeMount загрузить данные о статьях по сети через fetch из json-файла.

В форме использовать компонент с глобальной регистрацией (например кнопку или input).

<https://ru.vuejs.org/v2/guide/routing.html>

<https://router.vuejs.org/ru/guide/#html>

# Роутинг

npm install vue-router  
или  
vue add router

```
<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>
```

<https://ru.vuejs.org/v2/guide/routing.html>  
<https://router.vuejs.org/ru/guide/#html>

```
// 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')
```

# Динамические маршруты



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

```
const User = {
  template: '<div>Пользователь {{ $route.params.id }}</div>'
}
```

<https://router.vuejs.org/ru/guide/essentials/dynamic-matching.html>



```
src > router > JS index.js > [+] routes
 1 import Vue from 'vue'
 2 import VueRouter from 'vue-router'
 3 import Home from '../views/Home.vue'
 4 import NewArticle from '../views/NewArticle.vue'
 5 import Article from '../components/Article.vue'
 6 import store from '../store'
 7
 8 Vue.use(VueRouter)
 9
10 const routes = [
11   {
12     path: '/',
13     name: 'Home',
14     component: Home
15   },
16   {
17     path: '/about',
18     name: 'About',
19     // route level code-splitting
20     // this generates a separate chunk (about.[hash].js) for this route
21     // which is lazy-loaded when the route is visited.
22     component: () => import(/* webpackChunkName: "about" */ '../views/About.vue')
23   },
24   {
25     // dynamic route
26     path: '/article/:id',
27     component: Article,
28     // get props dynamically by route param
29     props: (route) => store.state.articles.find((x) => x.id == route.params.id)
30   },
31   {
32     path: '/new',
33     name: 'Add new article',
34     component: NewArticle
35   },
36 ]
37
38 const router = new VueRouter({
39   mode: 'history',
40   base: process.env.BASE_URL,
41   routes
42 })
43
44 export default router
```

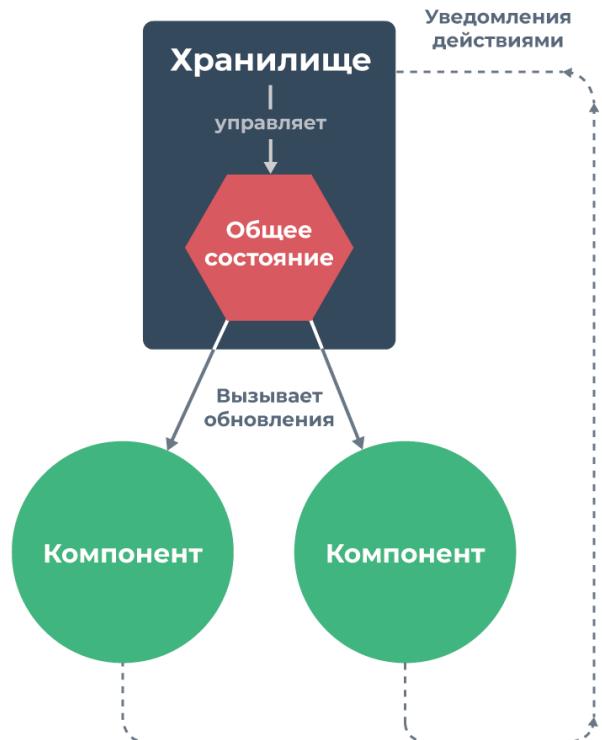
```
src > JS main.js > ...
 1 import Vue from 'vue'
 2 import App from './App.vue'
 3 import router from './router'
 4 import store from './store'
 5
 6 Vue.config.productionTip = false;
 7
 8 new Vue({
 9   router,
10   store,
11   render: h => h(App)
12 }).$mount('#app')

src > V App.vue > [+] template > [+] div#app > [+] div#nav
 1 <template>
 2   <div id="app">
 3     <div id="nav">
 4       <router-link to="/">Articles</router-link> |
 5       <router-link to="/about">About</router-link>
 6     </div>
 7     <router-view/>
 8   </div>
 9 </template>

// go to front page
this.$router.push('/');
```

```
src > components > V Article.vue > {} "Article.vue" > [+] script > [+] default > [+] methods
 1 <template>
 2   <div v-bind:class="{published: published, unpublished: !published}">
 3     <v-on:click="togglePublish">
 4       <h3 v-if="body">{{ title }}</h3>
 5       <h3 v-else v-bind:class="classObject">
 6         <router-link :to="articleUrl">{{ title }}</router-link>
 7       </h3>
```

# Управление состоянием



```
var store = {
  debug: true,
  state: {
    message: "Привет!"
  },
  setMessageAction(newValue) {
    if (this.debug) console.log("setMessageAction вызвано с ", newValue);
    this.state.message = newValue;
  },
  clearMessageAction() {
    if (this.debug) console.log("clearMessageAction вызвано");
    this.state.message = "";
  }
};

import store from './store'
```

```
new Vue({
  data: store.state, // все начальные ключи
  // store.state
  // станут реактивными
  свойствами!
  render: h => h(App)
}).$mount('#app')
```

В компонентах доступ через `this.$root.$data`  
Прямые изменения будет трудно отлаживать!

<https://ru.vuejs.org/v2/guide/state-management.html>

# Задание 6

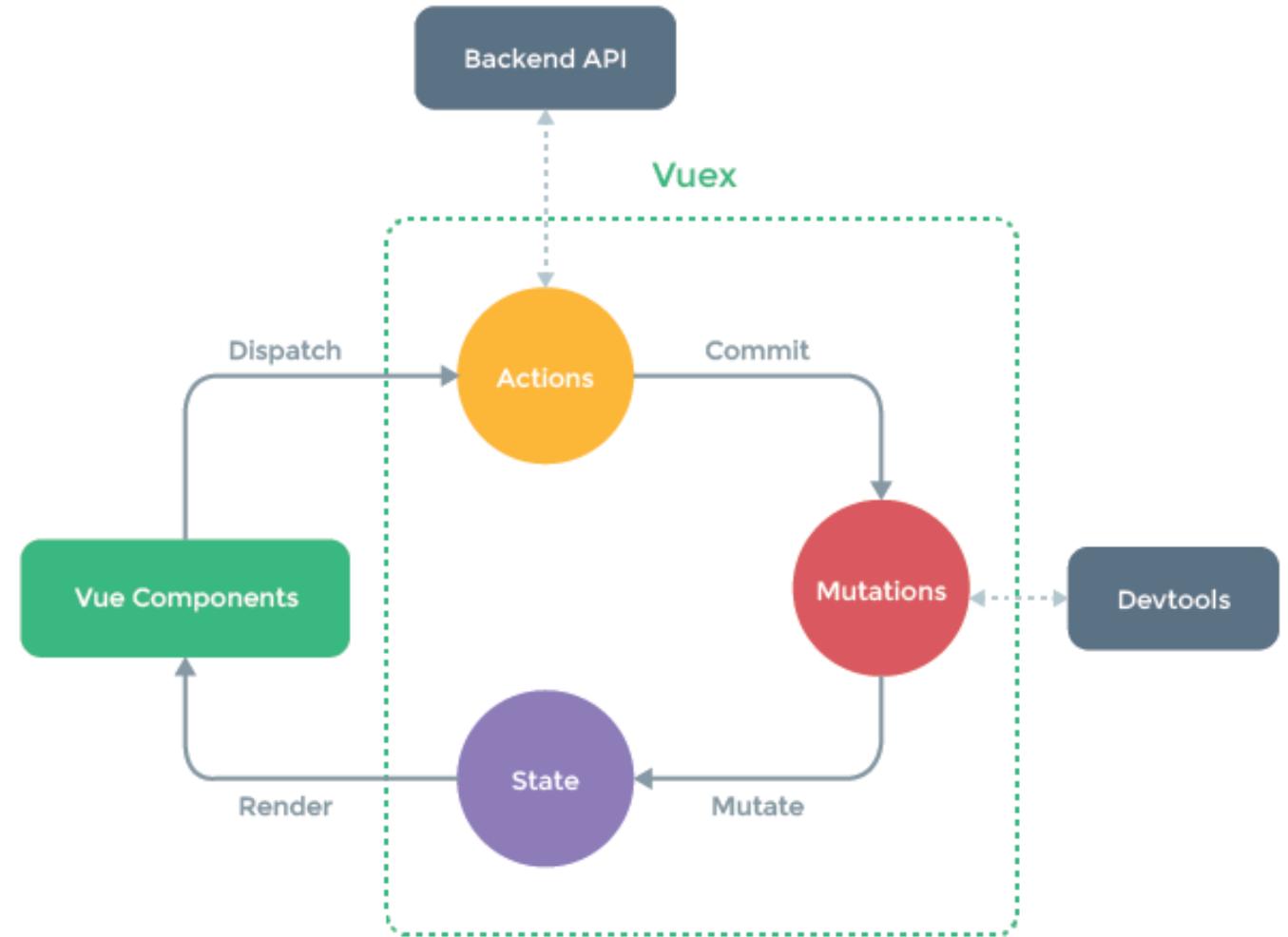
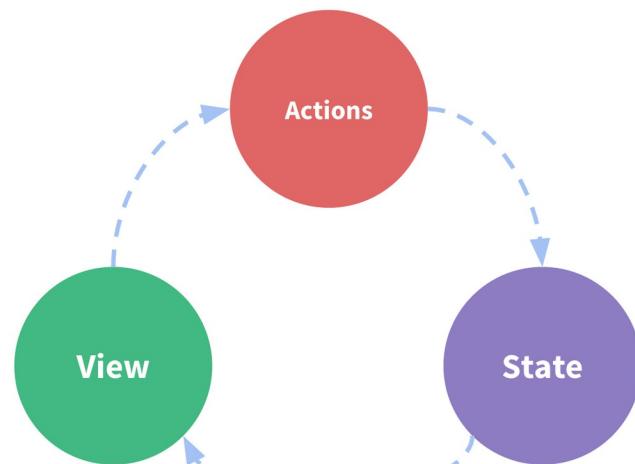


С помощью роутера Vue сделать переключение страниц:

1. Главная со списком статей со ссылками на них
2. О нас
3. Форма добавления новой статьи, после сабмита открывает главную
4. Страница статьи с url вида /article/{id}

Состояние приложения (массив статей) и методы его изменения вынести в объект state в store/index.js

# VueX



<https://vuex.vuejs.org/>

# VueX

vue add  
vuex

```
new Vue({  
  el: '#app',  
  store: store,  
})
```

```
store.commit('increment')  
  
console.log(store.state.count) // -> 1
```

<https://vuex.vuejs.org/ru/guide/state.html>

## Vuex 3.x (для Vue 2)

```
import Vue from 'vue'  
import Vuex from 'vuex'  
  
Vue.use(Vuex)  
  
const store = new Vuex.Store({  
  state: {  
    count: 0  
  },  
  mutations: {  
    increment (state) {  
      state.count++  
    }  
  }  
})
```

## Vuex 4.x (для Vue 3)

```
import { createStore } from 'vuex'  
import { createApp } from 'vue'  
  
const store = createStore({  
  state () {  
    return {  
      count: 1  
    }  
  }  
})  
const app = createApp({ /* ваш корневой компонент */ })  
app.use(store)
```



src > store > **JS** index.js > [e] default

```
1 import Vue from 'vue'  
2 import Vuex from 'vuex'  
3  
4 Vue.use(Vuex)  
5  
6 export default new Vuex.Store({  
7   state: {  
8     articles: []  
9   },  
10  mutations: {  
11    fetchArticles(state, articles) {  
12      state.articles = articles;  
13    },  
14    addArticle(state, article) {  
15      state.articles.push(article);  
16    }  
17  },  
18  actions: {  
19    fetchArticles(context) {  
20      fetch('/articles.json')  
21        .then(response => response.json())  
22        .then(articles => context.commit('fetchArticles', articles));  
23      console.log('Fetch data');  
24    },  
25  },  
26  modules: {}  
27})
```

src > **JS** main.js > ...

```
1 import Vue from 'vue'  
2 import App from './App.vue'  
3 import CustomInput from './components/CustomInput.vue'  
4 import router from './router'  
5 import store from './store'  
6  
7 Vue.config.productionTip = false;  
8  
9 Vue.component('CustomInput', CustomInput);  
10  
11 store.dispatch('fetchArticles');  
12  
13 new Vue({  
14   router,  
15   store,  
16   render: h => h(App)  
17 }).$mount('#app')
```

src > views > **▼** NewArticle.vue > {} "NewArticle.vue" > **script** > [e] default

```
1 <template>  
2   <ArticleForm title="New title" body="" author="" isPublished=""  
3   ||| v-on:add-article="addArticle" />  
4 </template>  
5  
6 <script>  
7 import ArticleForm from '@/components/ArticleForm.vue'  
8 import store from '../store'  
9  
10 export default {  
11   components: {  
12     ArticleForm  
13   },  
14   methods: {  
15     addArticle: function(article) {  
16       let newArticle = {  
17         id: store.state.articles.length + 1,  
18         ...article  
19       }  
20       store.commit('addArticle', newArticle);  
21       console.log(article.title);  
22       // go to front page  
23       this.$router.push('/');  
24     }  
25   }  
26 }  
27 </script>
```

The screenshot shows the Vue DevTools interface with the "Time Travel" tab selected. On the left, a list of mutations is shown with their timestamps: "Base State" at 21:24:26, "fetchArticles" at 21:55:38, and "addArticle" at 11:27:00. The "addArticle" mutation is highlighted with a green background. On the right, the inspected state is displayed under the "(Root)" node. It shows a "mutation" object with a "payload" containing an article object (author: "yyy", body: "yy", id: 4, isPublished: true, published: true, title: "New title", type: "addArticle"). Below it is a "state" object with an "articles" array containing three objects. At the bottom, there are status messages: "Recording state on-demand..." and "displaying last received state", along with a "Load state" button.



Time Travel Debugging работает только во Vue Dev Tools 5.x, в 6+ не работает!  
<https://github.com/vuejs/devtools/discussions/1692#discussioncomment-2140897>

# Задание 7



Переделать хранение состояния (массива статей) на Vuex.

Изменения состояния реализовать в мутациях Vuex.

Загрузку статей по сети сделать в виде действия,  
вызываемого до инициализации приложения.

Вывод статей в шаблоне сделать через вычислимые свойства  
(количество статей и массив статей),

которые строятся по состоянию Vuex с помощью mapState  
либо с помощью реактивных переменных состояния Vuex напрямую.

Поотлаживать изменения состояния и попробовать Time Travel  
Debugging с помощью VueDevTools.

\* Добавление статьи сделать в виде действия, вызывающего мутацию.

Реализовать обработку ошибки и повтор загрузки статей с помощью  
кнопки «Повторить загрузку» в случае ошибки.

Разнести состояние загрузки/ошибки и хранение статей по разным  
модулям [<https://vuex.vuejs.org/guide/modules.html>]. Почитать и  
попробовать неймспейсы для состояния.

Вынести метод загрузки статей по сети в модуль в каталоге services.

# Работа по сети



Работа по сети, обработка ошибок, ожидание сетевой операции, отмена.

Библиотека Axios.

Axios VS fetch: чуть короче код,  
МОЖНО отслеживать прогресс отправки  
(fetch отслеживает только прогресс загрузки,  
А XMLHttpRequest и отправки тоже).

Мок json-сервера на 10000 порту с задержкой 1 сек.:  
npm install -g json-server  
json-server --port 10000 --delay 1000 --watch articles.json

<https://www.npmjs.com/package/vue-axios>  
<https://github.com/axios/axios>  
<https://github.com/typicode/json-server>

ES6 Module:

```
npm install --save axios vue-axios
```

Import libraries in entry file:

```
import Vue from 'vue'  
import axios from 'axios'  
import VueAxios from 'vue-axios'
```

Usage in Vue 2:

```
Vue.use(VueAxios, axios)
```

Usage in Vue 3:

```
const app = Vue.createApp(...)  
app.use(VueAxios, axios)
```

```
Vue.axios.get(api).then((response) => {  
  console.log(response.data)  
})  
  
this.axios.get(api).then((response) => {  
  console.log(response.data)  
})  
  
this.$http.get(api).then((response) => {  
  console.log(response.data)  
})
```

Suspense (Vue 3) — отложенный компонент, показывающий заглушку, пока не выполнится условие (пришли данные с бекэнда например)



## locale.js

```
: import Axios from "axios";\n\nexport default function locale() {\n  async function getLocales() {\n    return await Axios.get(\n      "...url..." \n    ).then((response) => {\n      return response.data;\n    });\n  }\n\n  return { getLocales };\n}
```

## Простой пример:

```
<template>\n  <div>{{JSON.stringify(dataLocale)}}</div>\n</template>\n\n<script>\nimport locale from "@/composables/locale";\nexport default {\n  async setup() {\n    const { getLocales } = await locale();\n    const dataLocale = await getLocales();\n    return { dataLocale };\n  },\n};\n</script>
```

## Пример suspense-компонента:

```
<template>\n  <Suspense>\n    <template #default>\n      <Locale />\n    </template>\n    <template #fallback>Loading...</template>\n  </Suspense>\n</template>\n\n<script>\nimport Locale from "@/components/Locale.vue";\nexport default {\n  components: {\n    Locale,\n  },\n};\n</script>
```

Внутри `<template #default>` помещаем компоненты, у которых метод `setup()` возвращает Promise или асинхронные компоненты (загружаемые динамически по требованию). Внутри `<template #fallback>` мы помещаем то, что отображается при загрузке (например спиннер).

# Задание 8

Настроить бекэнд, выдающий JSON статей, через json-server с задержкой несколько секунд.

Сделать сетевые операции через Axios.

Во время загрузки списка статей сделать анимацию и функцию отмены.

Реализовать обработку ошибки пользователю если бекэнд не отвечает.

В случае ошибки бекэнда или отмены пользователем предусмотреть повторное выполнение загрузки статей кнопкой «Обновить».

Вынести метод загрузки статей по сети в модуль в каталоге services.

Завести соответствующие мутации и действия Vuex, описать и вызывать их через константы в store/types.js:



```
export const Types = {
  mutations: {
    ARTICLES_REQUESTED: 'ARTICLES_REQUESTED',
    ARTICLES_SUCCEEDED: 'ARTICLES_SUCCEEDED',
    ARTICLES_FAILED: 'ARTICLES_FAILED',
    ARTICLES_CANCELED: 'ARTICLES_CANCELED',
    ARTICLE_ADD_REQUESTED: 'ARTICLE_ADD_REQUESTED',
    ARTICLE_ADD_SUCCEEDED: 'ARTICLE_ADD_SUCCEEDED',
    ARTICLE_ADD_FAILED: 'ARTICLE_ADD_FAILED',
  },
  actions: {
    ARTICLES_LOAD: 'ARTICLES_LOAD',
    ARTICLES_LOAD_CANCEL: 'ARTICLES_LOAD_CANCEL'
  },
  request_status: {
    REQUESTED: 'REQUESTED',
    SUCCEEDED: 'SUCCEEDED',
    FAILED: 'FAILED'
  }
};
```

\* – реализовать на Vue 3 с использованием suspense-компонентта.

Модульность – загрузка компонентов из нескольких файлов

Телепорт — изменение DOM, который не принадлежит приложению

Доступ к props из setup без this:

```
setup(props) {  
  watch(() => {  
    console.log(props.name);  
  });  
}
```

# Material Design, Vuetify



Введение в Material Design.

Библиотека компонентов Vuetify.

Конфигурация, лейауты, гриды и адаптивность, темизация, i11y и i18n.

Диалоги, таблицы, списки, иконки и картинки.

Формы и кнопки.

# Material Design



Системы дизайна:

- Bootstrap;
- Bulma;
- **Material Design...**



**Material Design** (Материальный дизайн) - это популяризированная Google дизайн-система, которая воплощает в себе их принципы и рекомендации по дизайну.



<https://material.io/>

# Библиотека компонентов Vuetify



*Vuetify* - библиотека компонентов Vue, созданная в соответствии со спецификациями Material Design, для быстрого создания хорошо спроектированных приложений.



<https://vuetifyjs.com/en/>

# Добавление Vuetify в проект



С помощью Vue CLI

```
vue add vuetify
```



Эта команда внесет изменения в файлы шаблона вашего проекта, папку компонентов, vue.config.js и т.д. Если вы устанавливаете Vuetify через Vue-CLI, убедитесь, что вы зафиксировали свой код, чтобы избежать любой потенциальной потери данных. Изменения шаблона можно пропустить, выбрав опцию расширенной установки.

<https://vuetifyjs.com/en/getting-started/installation/>

# Добавление Vuetify в проект



## С помощью пакетного менеджера

Добавляем зависимость

```
yarn add vuetify  
# OR  
npm install vuetify
```

Создаем плагин для Vuetify, src/plugins/vuetify.js со следующим содержимым:

```
// src/plugins/vuetify.js  
  
import Vue from 'vue'  
import Vuetify from 'vuetify'  
import 'vuetify/dist/vuetify.min.css'  
  
Vue.use(Vuetify)  
  
const opts = {}  
  
export default new Vuetify(opts)
```

Подключаем шрифт и иконки. Vuetify использует шрифт Google Roboto и иконки Material Design.

```
<link href="https://fonts.googleapis.com/css?family=Roboto:100,300,400,500,700,900" rel="stylesheet">  
<link href="https://cdn.jsdelivr.net/npm/@mdi/font@4.x/css/materialdesignicons.min.css" rel="stylesheet">
```

<https://vuetifyjs.com/en/getting-started/installation/>

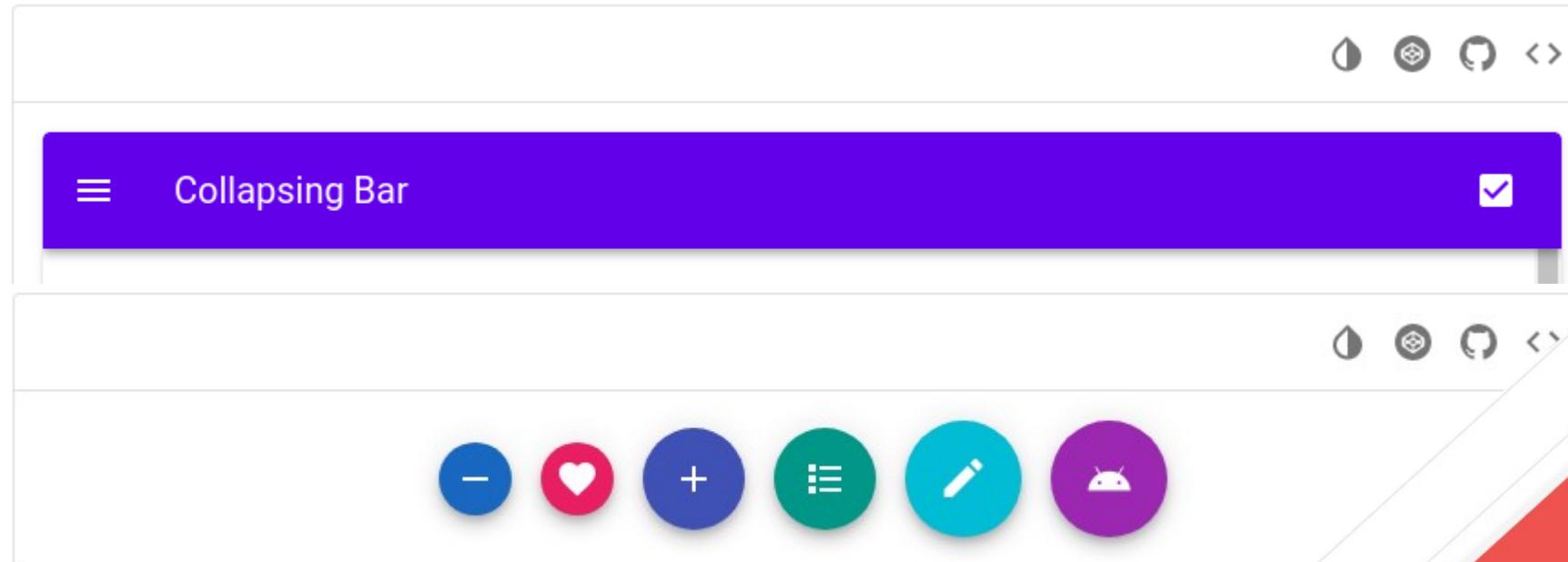
Переходим к своей основной точке входа и передаем объект Vuetify в качестве опции.

```
// src/main.js  
  
import Vue from 'vue'  
import vuetify from '@/plugins/vuetify'  
  
new Vue({  
  vuetify,  
}).$mount('#app')
```

# Компоненты



Все реализованные компоненты можно просмотреть на сайте документации в разделе “UI Components”. Их достаточно много: Alerts, Application, Aspect ratios, Avatars, Badges, Banners, Bars, Bottom navigation, Bottom sheets, Breadcrumbs, Buttons, Buttons: Floating, Action Button, Calendars, Cards, Carousels, Chips, Dialogs...



<https://vuetifyjs.com/en/getting-started/installation/>

# Демонстрация компонентов в документации

A screenshot of a Vue.js component editor interface. At the top, there's a purple header bar with the text "Collapsing Bar" and a checked checkbox icon. Below the header is a preview area showing a collapsed navigation bar. The main content area contains tabs for "TEMPLATE" and "SCRIPT", with "SCRIPT" currently selected. The script code is as follows:

```
<script>
  export default {
    data: () => ({
      collapseOnScroll: true,
    }),
  }
</script>
```

The bottom right corner of the editor has the word "VUE".

<https://vuetifyjs.com/en/components/app-bars/#props>

# Демонстрация компонентов в документации

A screenshot of a code editor interface. The top bar has icons for file operations. Below it, there are two tabs: "TEMPLATE" (selected) and "SCRIPT".

```
<template>
<v-card class="overflow-hidden">
  <v-app-bar
    :collapse="!collapseOnScroll"
    :collapse-on-scroll="collapseOnScroll"
    absolute
    color="deep-purple accent-4"
    dark
    scroll-target="#scrolling-techniques-6"
  >
    <v-app-bar-nav-icon></v-app-bar-nav-icon>

    <v-toolbar-title>Collapsing Bar</v-toolbar-title>

    <v-spacer></v-spacer>

    <v-checkbox
      v-model="collapseOnScroll"
      color="white"
      hide-details
    ></v-checkbox>
  </v-card>
</template>

<script>
  ...
</script>
```

The code demonstrates the use of the `v-app-bar` component with the `collapse` prop set to a computed value (`!collapseOnScroll`). It also shows the `collapse-on-scroll` prop being used to control the collapse behavior. The `v-spacer` component is used to create space between the toolbar title and the checkbox. The `v-checkbox` component is used to control the scroll target for the collapsing bar.

<https://vuetifyjs.com/en/components/app-bars/#props>

# Layouts Vuetify

Layout Vuetify позволяет легко и без особых усилий быстро формировать области пользовательского интерфейса приложения.

**app** - свойство сообщает Vuetify, что соответствующий компонент является частью макета вашего приложения

<https://vuetifyjs.com/en/features/layouts/>



```
<!-- App.vue -->

<v-app>
  <v-navigation-drawer app>
    <!-- -->
  </v-navigation-drawer>

  <v-app-bar app>
    <!-- -->
  </v-app-bar>

  <!-- Sizes your content based upon application components -->
  <v-main>

    <!-- Provides the application the proper gutter -->
    <v-container fluid>

      <!-- If using vue-router -->
      <router-view></router-view>
    </v-container>
  </v-main>

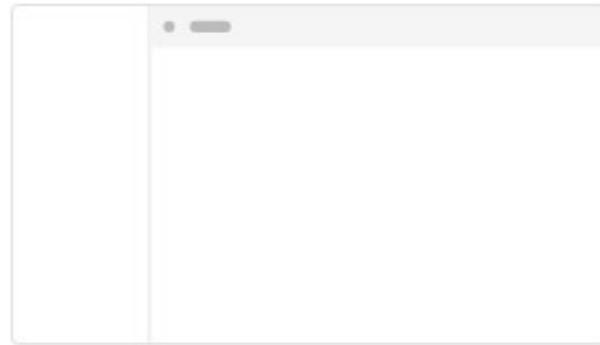
  <v-footer app>
    <!-- -->
  </v-footer>
</v-app>
```

# Layouts Vuetify



<https://vuetifyjs.com/en/features/layouts/>

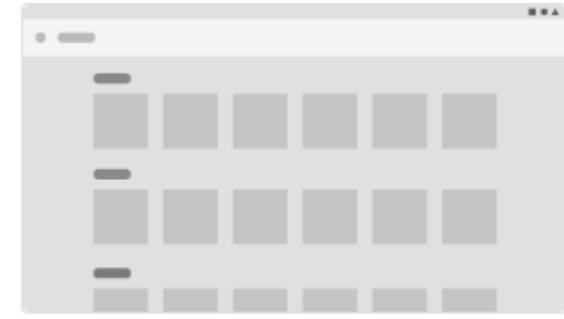
# Wireframes - готовые структуры



Base



Extended Toolbar



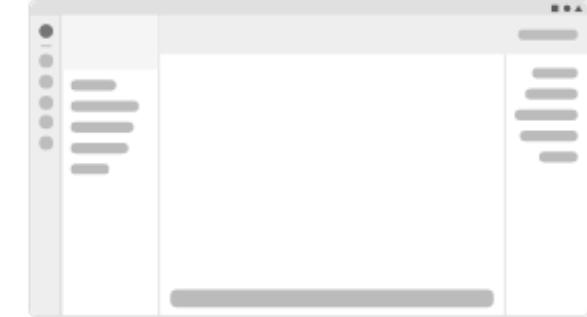
System Bar



Inbox



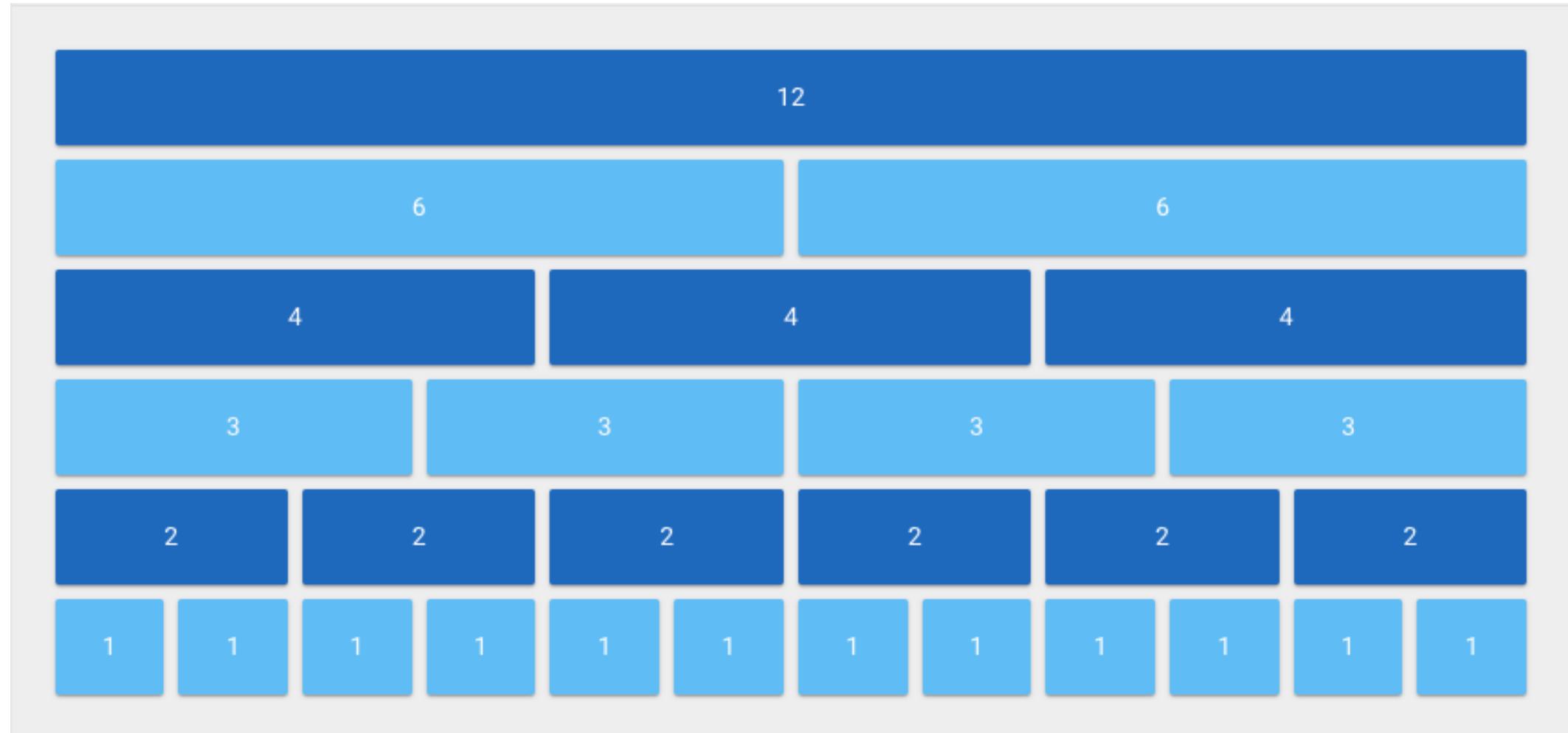
Constrained



Discord

<https://vuetifyjs.com/en/getting-started/wireframes/>

# Сетка



<https://vuetifyjs.com/en/components/grids/>

# Компоненты сетки и адаптивность



Material Design Breakpoints			
Device	Code	Type	Range
📱 Extra small	xs	Small to large phone	< 600px
💻 Small	sm	Small to medium tablet	600px > < 960px
💻 Medium	md	Large tablet to laptop	960px > < 1264px*
💻 Large	lg	Desktop	1264px > < 1904px*
💻 Extra large	xl	4k and ultra-wide	> 1904px*

\* -16px on desktop for browser scrollbar

<https://vuetifyjs.com/en/components/grids/>



```
<template>
  <v-container class="grey lighten-5">
    <v-row no-gutters>
      <v-col
        cols="12"
        sm="6"
        md="8"
      >
        <v-card
          class="pa-2"
          outlined
          tile
        >
          .col-12 .col-sm-6 .col-md-8
        </v-card>
      </v-col>
      <v-col
        cols="6"
        md="4"
      >
        <v-card
          class="pa-2"
          outlined
          tile
        >
          .col-6 .col-md-4
        </v-card>
      </v-col>
    </v-row>
  </v-container>
</template>
```

# Конфигурация темы



```
// src/plugins/vuetify.js

import Vue from 'vue'
import Vuetify from 'vuetify/lib'

import colors from 'vuetify/lib/util/colors'

const vuetify = new Vuetify({
  theme: {
    themes: {
      light: {
        primary: colors.purple,
        secondary: colors.grey.darken1,
        accent: colors.shades.black,
        error: colors.red.accent3,
      },
      dark: {
        primary: colors.blue.lighten3,
      },
    },
  },
})
```

Vuetify поддерживает как светлые, так и темные варианты спецификации Material Design.

<https://vuetifyjs.com/en/features/theme/>

# Vuetify, i11y



Доступность (**Accessibility, A11y** в англоязычной среде) в веб-разработке — это обеспечение возможности использования сайтов как можно большим числом людей, включая тех, чьи способности как-либо ограничены.

Google предоставляет подробный обзор того, как в Material Design их компоненты создаются с учетом данной специфики.

<https://material.io/design/usability/accessibility.html>  
<https://vuetifyjs.com/en/features/accessibility/>

# Vuetify, i18n



Vuetify поддерживает языковую локализацию (i18n) своих компонентах. При загрузке приложения вы можете указать доступные локали и текущую активную локаль с помощью current опции. Служба lang также поддерживает простую интеграцию с vue-i18n

<https://vuetifyjs.com/en/features/internationalization/>

```
// src/plugins/vuetify.js

import Vue from 'vue'
import Vuetify from 'vuetify/lib'

Vue.use(Vuetify)

// Translation provided by Vuetify (javascript)
import zhHans from 'vuetify/es5/locale/zh-Hans'

// Translation provided by Vuetify (typescript)
import pl from 'vuetify/src/locale/pl'

// Your own translation file
import sv from './i18n/vuetify/sv'

Vue.component('my-component', {
  methods: {
    changeLocale () {
      this.$vuetify.lang.current = 'sv'
    },
  },
})

export default new Vuetify({
  lang: {
    locales: { zhHans, pl, sv },
    current: 'zhHans',
  },
})
```

# Vuetify, i18n

Если вы создаете пользовательские компоненты Vuetify, которые необходимо подключать к механизму интернационализации, вы можете использовать `t` функцию

```
<!-- Vue Component -->

<template>
  <div class="my-component">
    {{ $vuetify.lang.t('$vuetify.my-component.text') }}
  </div>
</template>
```



Чтобы создать свой собственный перевод, используйте приведенный ниже код.

```
import { en } from 'vuetify/src/locale'

export default {
  ...en,
  key1: "key 1 internationalization",
  key2: "key 2 internationalization",

  namespace: {
    key3: "key 3 internationalization"
  }
}
```

<https://vuetifyjs.com/en/features/internationalization/>

# Vue-i18n пакет



```
// src/plugins/vuetify.js

import Vue from 'vue'
import Vuetify from 'vuetify/lib'
import VueI18n from 'vue-i18n'

Vue.use(Vuetify)
Vue.use(VueI18n)

const messages = {
  en: {
    $vuetify: {
      dataIterator: {
        rowsPerPageText: 'Items per page:',
        pageText: '{0}-{1} of {2}',
      },
    },
    sv: {
      $vuetify: {
        dataIterator: {
          rowsPerPageText: 'Element per sida:',
          pageText: '{0}-{1} av {2}',
        },
      },
    },
  }
}
```

```
// Create VueI18n instance with options
const i18n = new VueI18n({
  locale: 'sv', // set locale
  messages, // set locale messages
})

export default new Vuetify({
  lang: {
    t: (key, ...params) => i18n.t(key, params),
  },
})
```

<https://vuetifyjs.com/en/features/internationalization/>

# Vuetify Диалоги



TEMPLATE

SCRIPT

```
<template>
  <div class="text-center">
    <v-dialog
      v-model="dialog"
      width="500"
    >
      <template v-slot:activator="{ on, attrs }">
        <v-btn
          color="red lighten-2"
          dark
          v-bind="attrs"
          v-on="on"
        >
          Click Me
        </v-btn>
      </template>
    </div>
```

```
<v-card>
  <v-card-title class="headline grey lighten-2">
    Privacy Policy
  </v-card-title>

  <v-card-text>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
  </v-card-text>

  <v-divider></v-divider>

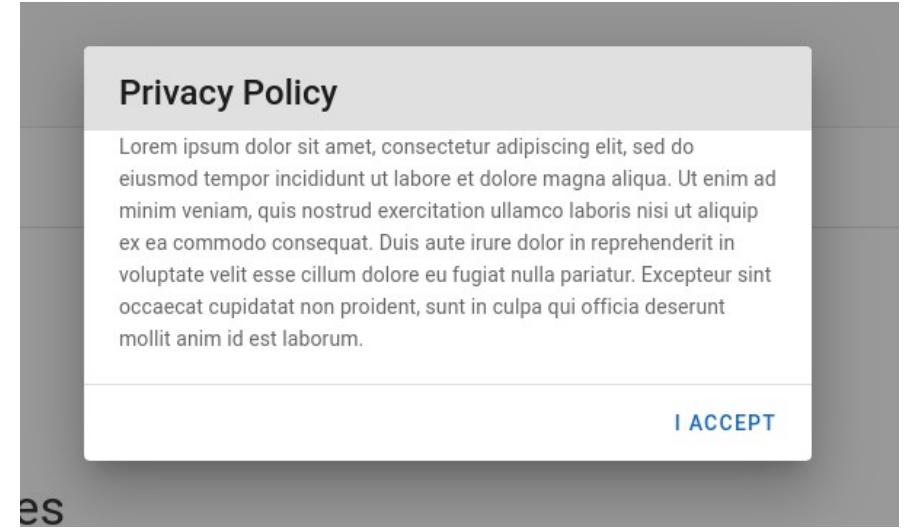
  <v-card-actions>
    <v-spacer></v-spacer>
    <v-btn
      color="primary"
      text
      @click="dialog = false"
    >
      I accept
    </v-btn>
  </v-card-actions>
</v-card>
</v-dialog>
</div>
</template>
```

<https://vuetifyjs.com/en/components/dialogs/>

# Vuetify Диалоги



```
<script>
  export default {
    data () {
      return {
        dialog: false,
      }
    },
  }
</script>
```



es

<https://vuetifyjs.com/en/components/dialogs/>

# Vuetify, Таблицы. Простые таблицы



```
<template>
  <v-simple-table>
    <template v-slot:default>
      <thead>
        <tr>
          <th class="text-left">
            Name
          </th>
          <th class="text-left">
            Calories
          </th>
        </tr>
      </thead>
      <tbody>
        <tr
          v-for="item in desserts"
          :key="item.name"
        >
          <td>{{ item.name }}</td>
          <td>{{ item.calories }}</td>
        </tr>
      </tbody>
    </template>
  </v-simple-table>
</template>
```

```
<script>
  export default {
    data () {
      return {
        desserts: [
          {
            name: 'Frozen Yogurt',
            calories: 159,
          },
          {
            name: 'Ice cream sandwich',
            calories: 237,
          },
          {
            name: 'Eclair',
            calories: 262,
          },
          {
            name: 'Cupcake',
            calories: 305,
          },
        ],
      }
    }
  }
</script>
```

Name	Calories
Frozen Yogurt	159
Ice cream sandwich	237
Eclair	262
Cupcake	305

## Полезные свойства:

**fixed-header** фиксирует шапку

**dense** компактный режим

<https://vuetifyjs.com/en/components/simple-tables/>

# Vuetify, Таблицы. v-data-table



Позволяет сортировать, искать, разбивать на страницы, редактировать содержимое и выбирать строки

Десерт (порция 100 г)	Калорий	Жиры (г)	Углеводы (г)	Белки (г)	Утюг (%)
Замороженный йогурт	159	6	24	4	1%
Сэндвич с мороженым	237	9	37	4.3	1%
Молния	262	16	23	6	7%
Кекс	305	3.7	67	4.3	8%
Имбирный пряник	356	16	49	3.9	16%

Строк на странице: 5 ▾ 1-5 из 10 < >

Реализацию фильтрации, поиска, выбора и пр. можно увидеть на странице документации компонента

<https://vuetifyjs.com/en/components/data-tables/>

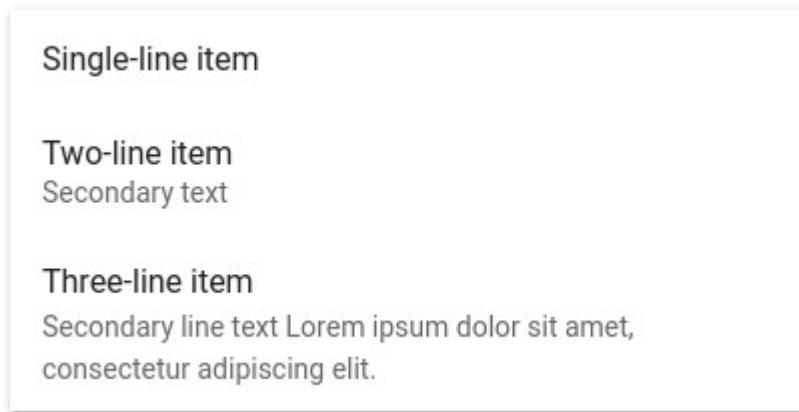
```
<template>
<v-data-table
  :headers="headers"
  :items="desserts"
  :items-per-page="5"
  class="elevation-1"
></v-data-table>
</template>
```

```
<script>
export default {
  data () {
    return {
      headers: [
        {
          text: 'Dessert (100g serving)',
          align: 'start',
          sortable: false,
          value: 'name',
        },
        { text: 'Calories', value: 'calories' },
        { text: 'Fat (g)', value: 'fat' },
        { text: 'Carbs (g)', value: 'carbs' },
        { text: 'Protein (g)', value: 'protein' },
        { text: 'Iron (%)', value: 'iron' },
      ],
      desserts: [
        {
          name: 'Frozen Yogurt',
          calories: 159,
          fat: 6.0,
          carbs: 24,
          protein: 4.0,
          iron: '1%',
        },
        {
          name: 'Ice cream sandwich',
          calories: 237,
          fat: 9.0,
          carbs: 37,
          protein: 4.3,
          iron: '1%',
        },
        {
          name: 'Vegan burger',
          calories: 356,
          fat: 16.0,
          carbs: 49,
          protein: 3.9,
          iron: '16%',
        },
        {
          name: 'Milkshake',
          calories: 262,
          fat: 16.0,
          carbs: 23,
          protein: 6.0,
          iron: '7%',
        },
        {
          name: 'Cookie',
          calories: 305,
          fat: 3.7,
          carbs: 67,
          protein: 4.3,
          iron: '8%',
        },
      ],
    }
  }
}
```

# Vuetify, Таблицы. Списки v-list



Компонент используется для отображения информации. Он может содержать аватар, контент, действия, подзаголовки и многое другое. Списки представляют контент таким образом, чтобы упростить идентификацию конкретного элемента в коллекции. Они обеспечивают единый стиль для организации групп текста и изображений.



## Основные компоненты:

```
<v-list>           <v-list-item-content>
<v-list-item-group> <v-list-item-subtitle>
<v-list-item>       <v-list-item-action>
<v-list-item-icon>
```

<https://vuetifyjs.com/en/components/lists>

```
<template>
<v-card
  class="mx-auto"
  max-width="400"
  tile
>
  <v-list-item>
    <v-list-item-content>
      <v-list-item-title>Single-line item</v-list-item-title>
    </v-list-item-content>
  </v-list-item>

  <v-list-item two-line>
    <v-list-item-content>
      <v-list-item-title>Two-line item</v-list-item-title>
      <v-list-item-subtitle>Secondary text</v-list-item-subtitle>
    </v-list-item-content>
  </v-list-item>

  <v-list-item three-line>
    <v-list-item-content>
      <v-list-item-title>Three-line item</v-list-item-title>
      <v-list-item-subtitle>
        Secondary line text Lorem ipsum dolor sit amet,
      </v-list-item-subtitle>
      <v-list-item-subtitle>
        consectetur adipiscing elit.
      </v-list-item-subtitle>
    </v-list-item-content>
  </v-list-item>
</v-card>
</template>
```

# Vuetify, Иконки v-icon



Компонент предоставляет большой набор символов, список иконок можно найти здесь: <https://materialdesignicons.com/>. Чтобы использовать необходимо добавить префикс *mdi* к имени значка внутри компонента.

```
<v-icon  
  large  
  color="green darken-2"  
>  
  mdi-domain  
</v-icon>  
  
<v-icon  
  large  
  color="blue darken-2"  
>  
  mdi-message-text  
</v-icon>  
  
<v-icon  
  large  
  color="purple darken-2"  
>  
  mdi-dialpad  
</v-icon>
```



Полезные свойства:

**x-small**

**small**

**medium (default)**

**large**

**x-large**

<https://vuetifyjs.com/en/components/icons/>

# Vuetify, Картинки v-img



Компонент используется для отображения адаптивного изображения с отложенной загрузкой



Полезные свойства:

```
gradient="to top right, rgba(100,115,201,.33), rgba(25,32,72,.7)"  
contain  
max-height  
height  
aspect-ratio  
...
```

```
<v-img  
  lazy-src="https://picsum.photos/id/11/10/6"  
  max-height="150"  
  max-width="250"  
  src="https://picsum.photos/id/11/500/300"  
></v-img>
```

<https://vuetifyjs.com/en/components/images/>

# Vuetify, Кнопки v-btn



Компонент заменяет стандартную кнопку HTML. Для изменения фона или цвета текста можно использовать любой вспомогательный класс цвета. Также доступны свойства размеров как и у иконок.

NORMAL      PRIMARY      ERROR      DISABLED



```
<v-btn  
  class="mx-2"  
  fab  
  dark  
  small  
  color="primary"  
>  
  <v-icon dark>  
    mdi-minus  
  </v-icon>  
</v-btn>
```

Полезные свойства:

rounded  
outlined  
plain  
text  
tile

...

```
<template>  
  <v-row  
    align="center"  
    justify="space-around"  
>  
    <v-btn depressed>  
      Normal  
    </v-btn>  
    <v-btn depressed  
      color="primary"  
>  
      Primary  
    </v-btn>  
    <v-btn depressed  
      color="error"  
>  
      Error  
    </v-btn>  
    <v-btn depressed  
      disabled  
>  
      Disabled  
    </v-btn>  
  </v-row>  
</template>
```

<https://vuetifyjs.com/en/components/buttons/>

<https://vuetifyjs.com/en/components/floating-action-buttons/>

# Vuetify, Формы v-form



Основные компоненты:

```
<v-autocomplete></v-autocomplete>  
<v-checkbox></v-checkbox>  
<v-combobox></v-combobox>
```

```
<v-file-input  
  accept="image/*"  
  label="File input"></v-file-input>
```

```
<v-input></v-input>  
<v-overflow-btn></v-overflow-btn>  
<v-radio-group></v-radio-group>  
<v-radio></v-radio>
```

```
<v-range-slider  
  hint="Im a hint"  
  max="50"  
  min="-50"></v-range-slider>
```

```
<v-select  
  :items="items"  
  label="Standard"></v-select>
```

```
<v-slider  
  hint="Im a hint"  
  max="50"  
  min="-50"></v-slider>
```

```
<v-switch></v-switch>
```

```
<v-textarea  
  autocomplete="email"  
  label="Email"></v-textarea>
```

```
<v-text-field></v-text-field>
```

<https://vuetifyjs.com/en/components/forms/> (В меню пункт “Form inputs & controls”)

# Vuetify, Формы v-form



Когда дело доходит до валидации формы, Vuetify имеет множество интеграций и встроенных функций. Из коробки можно использовать [Vee-validate](#) и [vuelidate](#).

Имя	Фамилия	Электронная почта
<input type="text"/> 0 / 10	<input type="text"/> 0 / 10	<input type="text"/>

v-form компонент упрощает добавление проверки ввода данных. Все компоненты имеют свойство `rules`, которое принимает смешанный массив типов `function`, `boolean` и `string`. Они позволяют указать условия, при которых ввод действителен или недействителен. Каждый раз, когда значение ввода изменяется, каждая функция в массиве получает новое значение, и каждый элемент массива будет оцениваться. Если функция или элемент массива возвращает `false` или `string`, проверка не удалась, и `string` значение будет представлено как сообщение об ошибке.

<https://vuetifyjs.com/en/components/forms/>

# Vuetify, Формы v-form

```
<template>
<v-form v-model="valid">
  <v-container>
    <v-row>
      <v-col
        cols="12"
        md="4"
      >
        <v-text-field
          v-model="firstname"
          :rules="nameRules"
          :counter="10"
          label="First name"
          required
        ></v-text-field>
      </v-col>

      <v-col
        cols="12"
        md="4"
      >
        <v-text-field
          v-model="lastname"
          :rules="nameRules"
          :counter="10"
          label="Last name"
          required
        ></v-text-field>
      </v-col>
    </v-row>
  </v-container>
</v-form>
</template>
```

```
<v-col
  cols="12"
  md="4"
>
  <v-text-field
    v-model="email"
    :rules="emailRules"
    label="E-mail"
    required
  ></v-text-field>
</v-col>
</v-row>
</v-container>
</v-form>
</template>
```

```
<script>
export default {
  data: () => ({
    valid: false,
    firstname: '',
    lastname: '',
    nameRules: [
      v => !!v || 'Name is required',
      v => v.length <= 10 || 'Name must be less than 10 characters',
    ],
    email: '',
    emailRules: [
      v => !!v || 'E-mail is required',
      v => /.+@.+\..test(v) || 'E-mail must be valid',
    ],
  })
}
</script>
```

<https://vuetifyjs.com/en/components/forms/>



# Тестирование компонентов. Jest



Модульное тестирование — фундаментальная часть разработки программного обеспечения. В модульных тестах выполняются небольшие фрагменты (единицы) кода в изоляции для упрощения добавления новых функциональных возможностей и отслеживания ошибок. Однофайловые компоненты Vue позволяют просто писать модульные тесты для компонентов в изоляции. Это поможет вам разрабатывать новую функциональность с уверенностью, что вы не ломаете работу существующей, и помогает другим разработчикам понять, как работает компонент.

Jest — фреймворк для тестирования. Это один из самых быстрых фреймворков для тестирования однофайловых Vue-компонентов (SFC).

<https://vuetifyjs.com/en/getting-started/unit-testing/>

# Задание 9



Оформить UI приложения красиво с использованием компонентов Vuetify и перекрытием свойств:  
v-navigation-drawer, v-main, v-footer,  
v-container, v-row, v-col,  
v-list, v-list-item, v-list-icon, v-list-item-content  
v-progress-circular  
v-dialog, v-simple-table  
v-carousel, v-card  
v-form, v-text-field  
v-textarea, v-btn, v-img

Add new article

New title

Author

Body

Published



**ADD**

Menu

-  Articles
-  About
-  Grid
-  Some components

Vue Tutorial

# Welcome to Your Vue.js App



Title 1  
Author: SERGEY SINITSA

**EXPLORE** **UNPUBLISH**



Title 2  
Author: SERGEY SINITSA

**EXPLORE** **PUBLISH**



Title 3  
Author: SERGEY SINITSA

**EXPLORE** **UNPUBLISH**

**ADD NEW ARTICLE**

2022 – Vuetify

# Проект

