

VUE TYPESCRIPT

“JQuery simplified js with dom

Then Vue simplified webapp development

So its up to You now as developer you should
simplify your app”



Corporate sw developer
.net based web solutions
by Radim

Czech republic , Portugal – Lisbon,
Austria – Vienna

Typescript

Javascript plus types, enums,
classes, interfaces, array functions...

Babel + flow Typescript

Less configurations than flow(type)

One config – tsconfig.json

Webpack - Ts-loader

So you want to start?

Great, read the styleguide first!

<https://vuejs.org/v2/style-guide/>

Vue Extend - ts

```
import Vue from "vue";

export default Vue.extend({
  props: ['name', 'initialEnthusiasm'],
  data() {
    return {
      enthusiasm: this.initialEnthusiasm,
    }
  },
  methods: {
    increment() { this.enthusiasm++; },
    decrement() {
      if (this.enthusiasm > 1) {
        this.enthusiasm--;
      }
    },
  },
},
```

```
    computed: {
      exclamationMarks(): string {
        return Array(this.enthusiasm + 1).join('!!')
      }
    }
  });
```

<https://github.com/Microsoft/TypeScript-Vue-Starter>

Vue ComponentOptions - ts better...

```
import Vue, { ComponentOptions } from 'vue'
// Declare the component's type
interface MyComponent extends Vue {
  message: string
  onClick (): void
}
export default {
  data: function () {
    return {
      message: 'Hello!'
    }
  },
  methods: {
    onClick: function () {
      // TypeScript knows that `this` is of type MyComponent
      // and that `this.message` will be a string
      window.alert(this.message);
    }
  }
}
// We need to explicitly annotate the exported options object
// with the MyComponent type
} as ComponentOptions<MyComponent>
```

Vue **class** component - ts

<https://github.com/vuejs/vue-class-component>

```
@Component({
  props: {
    propMessage: String
  },
  components: {
    Hello,
    World
  }
})
export default class App extends Vue {
  // props have to be declared for typescript
  propMessage: string
  // initial data
  msg: number = 123
  // use prop values for initial data
  helloMsg: string = 'Hello, ' + this.propMessage
  // lifecycle hook
  mounted () {
    this.greet()
  }
  // computed
  get computedMsg () {
    return 'computed ' + this.msg
  }
}
```

```
// method
greet () {
  alert('greeting: ' + this.msg)
  this.$refs.helloComponent.sayHello()
}
// dynamic component
$refs: {
  helloComponent: Hello
}
}
```

Official support

github.com/vuejs/vue-cli

Vue cli

npm install -g @vue/cli

vue create my-project

Or you download ready made templates

f.e. From github

<https://github.com/vuejs-templates>

Starting website using (project.json)

Npm run serve

Or in templates npm run dev

Specify Features

```
Vue CLI v3.0.0-alpha.7
```

```
? Please pick a preset: Manually select features
```

```
? Check the features needed for your project:
```

- ☐ TypeScript
- ☐ Progressive Web App (PWA) Support
- ☐ Router
- ☐ Vuex
- ☐ CSS Pre-processors
- ☐ Linter / Formatter
- ☐ Unit Testing
- ☒ E2E Testing

EXPLORER



Welcome



About.vue

TS main.ts



OPEN EDITORS

TESTCLI



public

src

assets

logo.png

components

HelloWorld.vue

views

About.vue

Home.vue

App.vue

TS main.ts

TS registerServiceWorker.ts

TS router.ts

TS shims.d.ts

TS store.ts

test

unit

TS HelloWorld.spec.ts

.babelrc

.gitignore

.postcssrc

jest.config.js

package-lock.json

package.json

tsconfig.json

tslint.json

vue.config.js

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

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

Type checking and linting in progress...

App running at:

- Local: <http://localhost:8080/>- Network: <http://192.168.0.62:8080/>

No type errors found

No lint errors found

Version: typescript 2.7.1, tslint 5.9.1

Time: 3948ms

Component **and** decorators

- Not repeating definitions

```
</div>
<h3>Ecosystem</h3>
<ul>
  <li><a href="https://router.vuejs.org/en/essentials/getting-started.html" target="_blank">router</a></li>
  <li><a href="https://vuex.vuejs.org/en/intro.html" target="_blank">vuex</a></li>
  <li><a href="https://github.com/vuejs/vue-devtools#vue-devtools" target="_blank">vue-devtools</a></li>
  <li><a href="https://vue-loader.vuejs.org/en" target="_blank">vue-loader</a></li>
  <li><a href="https://github.com/vuejs/awesome-vue" target="_blank">awesome-vue</a></li>
</ul>
</div>
</template>
```

```
<script lang="ts">
import { Component, Prop, Vue } from 'vue-property-decorator';
```

```
@Component
export default class HelloWorld extends Vue {
  @Prop() private msg!: string;
}
```

```
import { Component, Vue } from 'vue-property-decorator';
import HelloWorld from '@components/HelloWorld.vue'; // @ is an alias
```

```
@Component({
  components: {
    HelloWorld,
  },
})
export default class Home extends Vue {}
</script>
```

```
1  import Vue from 'vue';
2  import Router from 'vue-router';
3  import Home from './views/Home.vue';
4  import About from './views/About.vue';
5
6  Vue.use(Router);
7
8  export default new Router({
9    routes: [
10     {
11       path: '/',
12       name: 'home',
13       component: Home,
14     },
15     {
16       path: '/about',
17       name: 'about',
18       component: About,
19     },
20   ],
21 });
```

```
import Vue from 'vue';
import Vuex from 'vuex';

Vue.use(Vuex);

export default new Vuex.Store({
  state: {

  },
  mutations: {

  },
  actions: {

  },
});
```

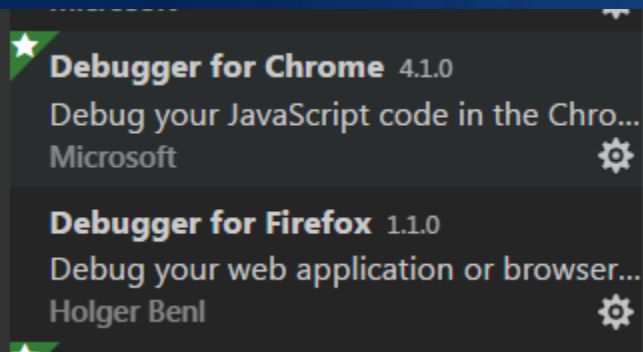
OK, what are good tools you should know

VS community - free

VS Code - free

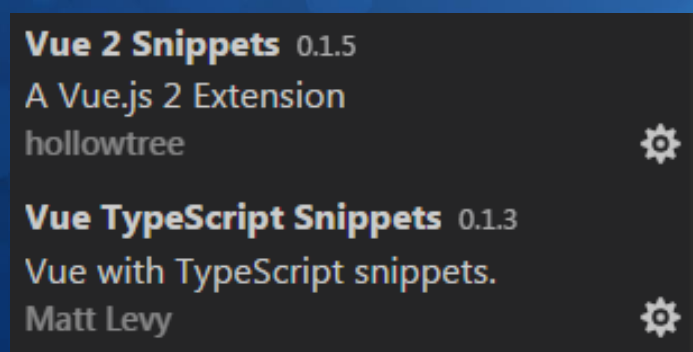
Vetur plugin

Support TS intellisense in vue singlefile components



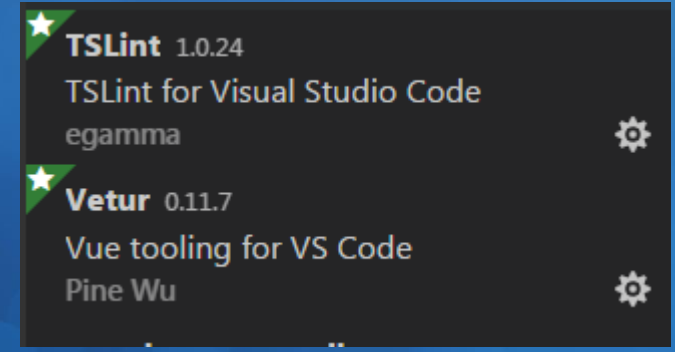
Debugger for Chrome 4.1.0
Debug your JavaScript code in the Chrome DevTools.
Microsoft

Debugger for Firefox 1.1.0
Debug your web application or browser...
Holger Benl



Vue 2 Snippets 0.1.5
A Vue.js 2 Extension
hollowtree

Vue TypeScript Snippets 0.1.3
Vue with TypeScript snippets.
Matt Levy



TSLint 1.0.24
TSLint for Visual Studio Code
egamma

Vetur 0.11.7
Vue tooling for VS Code
Pine Wu

Real World

You have some object in ts coming from your backend (node,.net, java python, php, graphql..)

Motivation: using object oriented programming with vue

You can have a representation in typescript




```
2 references
export class BEntity {
  0 references
  ID : number;
  0 references
  Count : number;
}
8 references
export class Post extends BEntity {
  0 references
  Title : string;
  0 references
  Content : string;
  0 references
  Categories : Category[];
}
6 references
export class Category extends BEntity {
  0 references
  Name : string;
  0 references
  PostCount: number;
}
```



```
import * as d from "../components/domain";
```

```
@Component({})
export default class post extends Vue {
  id = "post";

  @Prop() post: d.Post;
```

Real World – Intellisense on complex objects

```
created() {  
  this.post.  
   Title (property) Post.Title: string  
   Content  
   ID  
}
```

```
created() {  
  this.post.Categories[0].  
   Name (property) Category.Name: string  
   ID
```


Real World – props and reusable component

Mutating a prop locally is now considered an anti-pattern

Most use cases of mutating a prop can be replaced by one of these options:

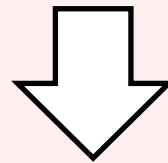
- **data property, with the prop used to set its default value**
- **computed property**

```
@Prop() post: d.Post;
```

[Vue warn]: Avoid mutating a prop directly since the value will be overwritten whenever the parent component re-renders. Instead, use a data or computed property based on the prop's value. Prop being mutated: "post"

Found in

```
--> <Compname> at src\components\post.vue  
    <Home> at src\Views\home.vue  
      <Admin> at src\Views\admin.vue  
        <App> at src\app.vue  
          <Root>
```



```
@Prop() post: d.Post;  
dpost = this.post;
```

Real World - Prop and Emit

Post.vue – emitting the event

```
<a @click.prevent="delpost(dpost.ID)" href="#">Delete</a>
```

```
@Component({})
export default class post extends Vue {
  id = "post";

  @Prop({ default: null })
  post: d.Post;
  dpost = this.post;

  ...

  delpost(ID: number) {
    this.db.bHub.deletePost(ID).then(ID => {
      //success
      this.$emit("ondeletepost", ID);
    });
  }
}
```

Post-list.vue

Passing prop and catching event

```
<div v-for="post in posts" :key="post.ID">
  <post @ondeletepost="postdeleted" :post="post"></post>
</div>
```

```
postdeleted(id: number) {
  this.getData();
}
```

Real World -
Html types
HtmlElement etc.

```
(this.$refs["scroll"] as HTMLDivElement).scrollIntoView();
```

Real World - Vuex - ts

```
let store = new Vuex.Store<State>({
  state: statee,
  mutations: {
    setvars(state, s: storeData) {
      state.vars = s;
      storage.setItem(storage.C_ENV_KEY, s); //update local storage
    },
    setdb(state, s: cl.SgnRCloud) {
      state.db = s;
    }
  }
});
store.subscribe((mutate, statee) => {
  if (mutate.type == "setvars") {
    console.log("subscribed muttate");
  }
});
```

```
const dstate: storeData = {
  count: 0,
  isAuth: false,
  token: "",
  lang: "de",
  mandantid: 0,
  location: "AT",
  servurl: host,
  dateformat: "DD.MM.YYYY",
  oauth: null
};
```

```
export interface State {
  db: cl.SgnRCloud;
  vars: storeData;
}
```

Real World - event bus, tips and tricks

Create an event bus

```
export interface State {  
  bus: Vue
```

```
const state: State = {  
  bus: new Vue(),
```

Some edit component somewhere in webapp hierarchy, that doesn't need shared state

```
//in this case I don't share any global state, didn't modify any state,  
//so I raise event on eventbus - otherwise I would commit mutation  
this.$store.state.bus.$emit("onsavepost", this.post);
```

Some container reacting on data changes, read past data

```
//in this case I want to react on some event, not sharing state, so using eventbus  
this.$store.state.bus.$on("onsavepost", (post: d.Post) => {  
  this.getData();  
});
```

Real world – typesafe eventbus

```
export class VueBus extends Vue
{
  public onSavePost(func:(p:cl.Post)=>void)
  { this.$on("onsavepost", func); }
  public emitSavePost(p:cl.Post)
  {this.$emit("onsavepost",p ); }
}
//defining state
export interface State {
  bus:VueBus
```

```
this.$store.state.bus.emitSavePost( this.post);
```

```
//in this case I want (method) VueBus.onSavePost(func: (p: d.Post) => void): void
this.$store.state.bus.onSavePost((post: d.Post) => {
  this.getData();
});
```


Questions?

Web: <http://atrock.tk>
twitter: @jackbejk
Github“ @jack85