How to read these codes snippets

Vue components contain in the same place `<sc rip t>`, `<te mpl ate>` and `<st yle>` data contents. In order to distinguish them: *Italic code is for script*. Regular code is for template.

Directives

v-if / v-else-if / v-else	Condition	<section v-show='var == true'>show me</section>
v-show	Condition (no alternative)	<section v-if='var == true'>show me</section>
v-for	Loop (iterate dataStore elt)	<li v-for='elt in dataStoreElt'>...
v-on:event (@event)	Event listener	<button v-on:click='function'>Click!</button>
v-bind:elt (:elt)	Bind	<li v-bind:href='dataStoreElt.prop'>...
v-model	Bind forms	<input type='text' v-model='dataStoreElt' />

Share data between parent/child components

Direct data sharing between 2 linked components is possible in 3 ways: **props** & **slot** for *parent* communicating to *child*, and **$emit** for *child* communicating to *parent*.

props (Parent > Child)

Parent /	Child /
import Child from './url'	*props : ['url'] or props : {url : 'url'}*
components : {Child}	
<Child url= '/contact '>	<a :href= 'url'>

slot (Parent > Child)

Parent /	Child /
import Child from './url'	
components : {Child}	
<Child v-slot:slotName >anything to replace default content </Child>	<button> fixed data <slot name='slotName'>default free content </slot>{{nl]] </button>

Slots are **free areas** that can be named to use more than one. Default content of slots set in *child component* can be replaced as needed.

$emit (Child > Parent)

Parent /	Child /
import Child from './url'	*methods : { eventPerso(){ this.$emit('event-perso',{msg : 'hello'}) } }*
components : {Child}	
data() { return{ msg : '' } }	
methods : { setMsg(event) { this.msg = event.msg } }	
<Child @event-perso = 'setMsg' />	<button @click = 'eventPerso'>

Router

Install: `vue add router` // *App.vue wil be erased!*

route = { **path**:'/', **name**: 'Name', **component**: ImportedName *OR* () => import('../path/Component') }

<router-view /> - Display point

<router-link to='/path'></router-link>- Link using routes (no reload page)

Lifecycle

beforeCreate()/created() beforeMount()/mounted() beforeDestroy()/destroyed()

Function used in **export default {...}** to define the moment some operations must be ignitiated.

Vuex - General dataStore

State (dataStore)	Paths: **$store.state** pluggin: **{ mapState }** - use spread operator ...
Pluggin must be used in **computed** * **objects** can be used in place of **strings** to rename variables	**Vuex store/** *state: {key: 'value'}* **Component/** *Import { mapState } from 'vuex'* *computed: { ...mapState(['var1','var2']) } **
Getters (computed)	Path: **$store.getters** pluggin: **{ mapGetters }** - use spread operator ...
Pluggin must be used in **computed**	**Vuex store/** *getters: { setDate(state) { return* `${stat e.d ay} :${ sta te.m on t h} :${ sta te.y ear}` *}* **Component/** **direct/** *computed : { date() {this.$store.getters('setDate')} }* **pluggin/** *import { mapGetters } from 'vuex'* *computed : {...mapGetters(['setDate']) }*
Mutations (change State)	Path: - no access from components - Pluggin: - none -
Isolated from components CAPITALIZED_NAME args: *state, payload*	**Vuex store/** *state: { count: 0 }* *mutations: { INCREASE_COUNT(state, amount) {state.count += Number(amount)} }*
Actions (commit changes to State)	Path: **$store.dispatch** Pluggin: **{ mapActions }** - use spread operator ...

Vuex - General dataStore (cont)

Pluggin must be used in **methods**	**Vuex store/** *actions: { updateCount(context,amount) {context.commit('INCREASE_COUNT', amount)} }*
Apply mutators to State	**Component/**
args: *context, payload*	***direct/*** *methods: { update() {this.$store.dispatch('updateCount')} }*
	pluggin/ *import { mapActions } from 'vuex'*
	methods: { ...mapActions(['updateCount']) }

All data contained in this dataStore is accessible and changeable to any component.