CHEAT SHEET

VUEJS 2 PROGRESSIVE FRAMEWORK

BASICS, DIRECTIVES & INSTANCES

Created by @Manz (https://twitter.com/Manz)

https://lenguajejs.com/

d Directives `<elem v-directive></elem>`

BASIC DIRECTIVES

v-text="s" set textContent (text)
{{ v }} use expression or data SHORTHAND
v-html="s" set innerHTML (raw HTML) XSS
v-pre skip render {{ vars }} (inside)
v-cloak hide element until vue is ready
v-once only render {{ vars }} first time

CONTROL DIRECTIVES

v-show="b" css-hide if false (always exists)
v-if="b" render content if true, else none
v-else-if="b" chained v-if condition 2.1.0+
v-else otherwise, render this content

LOOPS DIRECTIVES

v-for="i in elem" iterate elem (for...in)
v-for="(v, i) in a" v=value, i=index
v-for="(v, k, i) in o" v=value, k=key, i=index
v-for="i in n" iterate from 1 to num
v-for="i in s" iter. from first to last char
:key="n s" hint for identify elements

BINDING DIRECTIVES

MODEL FORM BINDING (2-WAY)

v-model="s" bind on input/select/textarea
v-model.lazy="s" listen "change" event
v-model.number="n" cast to number
v-model.trim="s" strip left/right spaces

HTML ATTRIBUTE BINDING

v-bind="o s" bind key:val on element attrs
v-bind:*attr*="s" bind "s" on attr (ex: alt <img>)
:*attr*="s" binding var "s" on attr SHORTHAND
:*attr*="b" only show attr if true (ex: disabled)
:*attr*.prop ="s" binding as DOM property
:*attr*.camel ="s" kebab-c → camelCase 2.1.0+
:*attr*.sync ="s" expand into v-on handler 2.3.0+

CLASS BINDING — INLINE EQUIVALENT

:class="s" add string to class ="'string'"
:class="a s" add all to class ="[s1,s2]"
:class="o b" add key if true ="{key:b}"

STYLE BINDING — INLINE EQUIVALENT

:style="s" apply string styles ="'prop:val'"
:style="o s" apply k:v styles ="{prop:val}"
:style="a o s" apply objects ="[o1,o2]"

EVENTS DIRECTIVES

v-on="o f" add n-events (key:func) 2.4.0+
v-on:*event*="f" add event listener
@*event*="f" event listener SHORTHAND
@*event*="f(param, $event)" inline
@*event*.stop="f" call .stopPropagation()
@*event*.prevent="f" call .preventDefault()
@*event*.capture="f" capture mode listen
@*event*.self="f" trigger only from this
@*event*.once="f" trigger at most once
@*event*.*keyCode*="f" trigger on key
@*event*.*sysKey*="f" trigger on key 2.1.0+
@*event*.native="f" listen native event
@*event*.*button*="f" only mouse button 2.2.0+
@*event*.passive="f" add passive event 2.3.0+

{ } Template Syntax — Templates

BASIC (MUSTACHE SYNTAX)

{{ var }} show data (props, data...)
{{ bool ? 'Yes' : 'No' }} ternary condition
{{ var + 's' }} JS short expression

ADVANCED (MUSTACHE SYNTAX)

{{ computed }} complex expression
{{ var | filter }} apply filter over var

f Custom Filters

CALLBACK VALUES: f (v) prev. value

GENERAL SYNTAX (GLOBAL)

Vue.filter(s name, f filter_fn)

i Vue Instance — vm = ViewModel

PROPERTIES

o vm.$data data vue observing
o vm.$props component props 2.2.0+

READ-ONLY PROPERTIES

e vm.$el root DOM element
o vm.$options instance custom options
v vm.$parent parent instance
v vm.$root root instance (or itself)
a vm.$children direct childs instances
o vm.$slots distribution slots
o vm.$scopedSlots scoped slots 2.1.0+
o vm.$refs elems registered w/ref attrs
b vm.$isServer server-side instance SSR
o vm.$attrs parent-scope attrs bindings
o vm.$listeners parent-scope v-on ev. l.

METHODS

f vm.$watch(f exp, f fn, o options)
* vm.$set(o a tgt, s n key, * v)
* vm.$delete(o a tgt, s n key)

METHODS (CUSTOM EVENTS)

vm.$on(s a s ev, f fn) listen c. ev.
vm.$once(s ev, f fn) only once
vm.$off(s a s ev, f fn) remove c. ev.
vm.$emit(s name, a args) trigger

METHODS (LIFECYCLE)

v vm.$mount(e s el) manual mount
vm.$forceUpdate() force to re-render
vm.$nextTick(f fn) next DOM update
vm.$destroy() destroy vue instance

e Custom Events — Communication

EMIT CUSTOM EVENT FROM CHILD TO PARENT

emits custom event w/data to parent:
`this.$emit('myevent-kebab-case')`

CAPTURE CUSTOM EVENT ON PARENT

capture custom event from child:
`@myevent="method-handler"`

KEYCODE ALIAS

enter up
tab down
delete left
esc right
space custom

SYSKEY

ctrl
alt
shift
meta

exact

BUTTON

left
middle
right

WATCH OPTIONS

b deep nested
b immediate

s Special Tags & Attrs `<elem attr></elem>`

SPECIAL ATTRIBUTES

ref="s" register elem or child component
slot="s" name of component slot
slot-scope="s" scoped slot (expr) 2.5.0+
key="n s" hint for vDOM identify node

SPECIAL TAGS

<component> meta-component (dynamic)
is="s o" used in dynamic components
inline-template include templates inline
<keep-alive> cache comp. w/o destroy
include="s" only cache match names 2.1.0+
exclude="s" not cache match names 2.1.0+
max="n" maximum comp. to cache 2.5.0+
<slot> serve as content distribution slot
name="s" name slot

t Vue Transitions — Transition & Animations

SINGLE TRANSITIONS

<transition> transition effect for elem/comp.
name="s" auto expand name (def: "v")
appear apply transition on init render
css="b" apply css transition classes
type="s" set "transition" or "animation"
mode="s" set timing "out-in" or "in-out"

CLASSES FOR APPLY TRANSITION CLASS

enter	*-class	*-to-class	*-active-class
leave	*-class	*-to-class	*-active-class
appear	*-class	*-to-class	*-active-class

EVENTS

enter	before-*	after-*	*-cancelled
leave	before-*	after-*	*-cancelled
appear	before-*	after-*	*-cancelled

MULTIPLE TRANSITIONS

<transition-group> trans. effect for multiple
tag="s" convert to html tag 'span'
move-class="s" overwrite during trans.
same trans. props (except mode) & events

d Custom Directives — Create your v-some!

GENERAL SYNTAX (GLOBAL)

Vue.directive(...)
❶ (s name, o f hooks) global register
❷ (s name, f bind_and_update_fn) SHORTHAND

HOOK METHODS

.bind: f when bound to element (only once)
.inserted: f when inserted into parent node
.update: f after content has updated
.componentUpdated: f same, and children
.unbind: f when unbound from elem (once)

HOOK METHODS ARGUMENTS

e el when bound to element (only once)

HOOK METHODS ARGUMENTS (READ-ONLY)

o binding data object with...
s .name directive name, w/o v- prefix
* .value directive value, computed
* .oldValue prev value (update hooks)
s .expression value as string
s .arg argument (ex: v-sample:foo)
o .modifiers mods list (ex:v-sample.foo)
e vnode virtual node produced by vue
e oldVnode prev virtual node (update hooks)

n number
s string
b boolean
b def: true

c component
d date
r regexp
f function

a array
s symbol
t datatype
* any

o object
undefined
e HTMLElement
v Vue instance

CHEAT SHEET

VUEJS 2 PROGRESSIVE FRAMEWORK

COMPONENTS & SINGLE FILE COMPONENTS

Created by @Manz (https://twitter.com/Manz) https://lenguajejs.com/

c Vue Components — Create your component

GENERAL SYNTAX (GLOBAL)

Vue.component(...)
❶ (s id, o def) register a global component
❷ (s id) retrieve a registered component

DEFINITION / NAME

.name: s set/change name to component

DEFINITION / DOM OPTIONS

.el: s e existing DOM element to mount on
.template: s string template as markup
.render: ce alternative with render functions
.renderError: ce alt. output for errors 2.2.0+

DEFINITION / DATA OPTIONS

.props: a s o attrs exposed from parent
.propsData: o pass props to an instance
.data: f data object (reactive). Return a o
.computed: o f cached computed data
.methods: o f methods object data
.watch: o f observe for changes (old, new)

DEFINITION / LIFECYCLE HOOKS

.beforeCreate: f before data observation
.created: f after instance is created
.beforeMount: f before mount step
.mounted: f after mount step
.beforeUpdate: f before DOM patched
.updated: f after vDOM re-render & patch
.activated: f kept-alive c. activated
.deactivated: f kept-alive c. deactivated
.beforeDestroy: f before instance destroy
.destroyed: f after instance destroyed
.errorCaptured: f (err, vm, info) 2.5.0+

DEFINITION / ASSETS

.directives: o name-directive hash
.filters: o name-filter hash
.components: o name-component hash

DEFINITION / COMPOSITION

.parent: v set parent-child relationship
.mixins: a o merge objects (priority: own)
.extends: o extends from other component
.provide: o data inject into descendants 2.2.0+
.inject: a o retrieve data from ancestor 2.2.0+

DEFINITION / MISC

.delimiters: a s set delimiters ['{{','}}']
.functional: b set stateless & instanceless
.model: o s custom prop/event v-model 2.2.0+
.inheritAttrs: b set parent-scope inherit 2.4.0+
.comments: b preserve comments 2.4.0+

x Reuse Components — Mixins & Extends

REUSABILITY & COMPOSITION (GLOBAL)

Vue.extend(o def) create "subclass"
Vue.mixin(o mixin) apply & merge mixin

INITIAL SNIPPET FOR MIXINS

```
import { m1 } from './mixins/m1.js';
export default {
  mixin: [m1]
```

```
export const = {...}
```

DOM MOUNT

auto mount:
`el:'#app'`
manual mount:
`vm.$mount(el)`

```
function(createElement) {
  return createElement(App)
}
```

`h => h(App)`

h: hyperscript

RENDER FUNCTIONS

h(el,def,child)
el: s c element
def: o definition
child: s a s elems

PROPS COMPLEX SYNTAX

type: t a t datatype,
required: b bool,
default: * value,
validator: f boolean

o Data Options — Main data options

PROPS (EXTERNAL DATA)

❶ props a s [...] attrs from out — SHORT SYNTAX
props o { ... }
❷ { key: t a t datatype, ... } — FAST SYNTAX
❸ { key: props options, ... } — COMPLEX SYNTAX
propsData o { ... } pass props to instance

DATA

❶ data o { ... } data object — SIMPLE SYNTAX
❷ data f { return o { ... } } — IN COMPONENTS

COMPUTED DATA (CACHED)

computed o f {...} cached data objects
❶ { key: f function, ... } — FAST SYNTAX
❷ { key: { get: f , set: f } — COMPLEX SYNTAX

METHODS (FUNCTIONS)

methods o f { ... } methods object
{ key: f function, ... }

WATCH (KEY IS NAME FROM DATA TO WATCH)

watch o { ... } run when data name changes
❶ { key: f function, ... }
❷ { key: s methodName, ... }
❸ { key: { handler: f , deep: b } }
❹ { key: { handler: f , immediate: b } }
❺ { key: a f [...] }
❻ { s 'obj.name': f }

CALLBACK VALUES
f (new_v, old_v)

EXAMPLE SNIPPET FOR WATCH

```
❶ a: (v,old) => {...};
❷ b: 'fnName',
❸ c: { handler: (v,old) => {...}, deep: b }
❻ 'd.a': (v,old) => {...}
```

A Global API — Other main options

ASYNC UPDATE QUEUE

Vue.nextTick(f fn, o ctx) defer exec func. to next DOM update cycle (wait changes)
❶ .nextTick(f fn) normal use
❷ .nextTick().then(f fn) w/promise 2.1.0+

SET/UNSET REACTIVE DATA

Vue.set(o a tgt, s n key, * v) add data
Vue.delete(o a tgt, s n key) del data

OTHERS

Vue.use(o f plugin) install Vue plugin
Vue.compile(s tpl) compile to render func.
s Vue.version return Vue version string

v Vue — Create new Vue instance

INITIAL CODE FOR VUEJS

```
import Vue from 'vue'
const vm = new Vue({
  el: '#app',
  ...
})
```

VUE BUILDS
compiler render f.
runtime vue core
full core+compiler

s Single File Components — .vue files

TEMPLATES (HTML CODE)

<template> component html code
lang="s" preprocessor (pug)
src="s" externalize code on file

STYLES (CSS CODE)

<style> component css code (style)
lang="s" less, sass, scss, stylus, postcss
src="s" externalize code on file
scoped apply css only to this component

SCRIPT (JAVASCRIPT CODE)

<script> component javascript code
lang="s" preprocessor (ts)
src="s" externalize code on file

INITIAL SNIPPET FOR SFC COMPONENT

```
import File from './File';
❶export default { ...                    JAVASCRIPT
❷export default Vue.extend({ ...         TYPESCRIPT
  name: 'file',
  components: {
    'file-component': File
```

s Structure — Vue File Structure

INITIAL FILE & FOLDER STRUCTURE

```
src                 // source files
  assets            // dynamic assets
  components        // ui components
  mixins            // reused defs
  router            // routing data
  store             // vuex, flux, ...
  translations      // i18n translations
  utils             // helpers
  views             // view components
  main.js           // app entry file
  App.vue           // main app component
```

c Vue.config — Configuration API

PROPERTIES

.optionMergeStrategies o f (par, child, vm)
.ignoredElements a s customElems
.keyCodes o n define keycode aliases

ERRORS/WARNINGS HANDLER

.errorHandler f (err, vm, info 2.2.0+)
.warnHandler f (msg, vm, trace) 2.4.0+

FLAGS

.silent b supress all vue logs/warnings
.devtools b allow vue devtools inspect
.performance b monitoring vue perf.
.productionTip b show tips on startup