**Vue.js – OSNOVE JAVASCRIPT FRAMEWORK-a**

**INSTALACIJA:** <https://vuejs.org/v2/guide/installation.html>

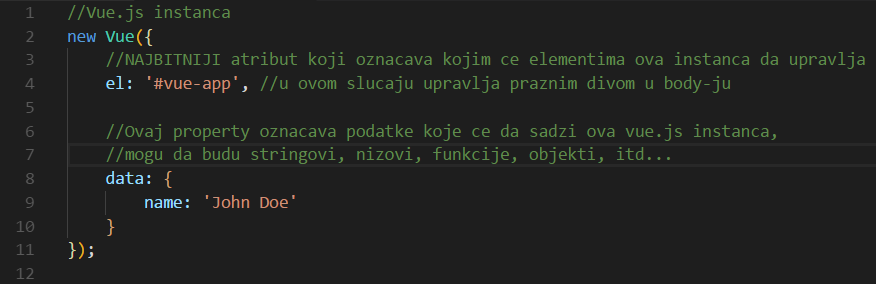
* Ako je upitanju samo jedan projekat onda se koristi script element koji je dat na sajtu. On se pise u <head> html elementa kako bi se includovao, ali ce se js pisati u externom js fajlu, koji se inkluduje na kraju <body> elementa, gde ce da bude ceo Vue.js kod.

<script src="https://cdn.jsdelivr.net/npm/vue@2.5.13/dist/vue.js"></script>

* Ako se kreiraju SPA (single page applications) ili kompleksniji projekti onda se vue.js instalira u globalu na kompu.

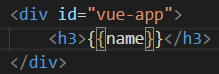
**1. KORAK – Vue.js Instanca**

* Instanca se kreira u posebnom .js fajlu. Ova vue.js instanca se moze koristiti za kontrolu dela nase aplikacije ili cele aplikacije. Vise instanci se koristi ako npr. na sajtu imamo widgete pa svaka instanca kontrolise jedan.
* Kako ova instanca kontrolise aplikaciju? U vitičastim zagradama se ovom objektu prosledjuju neke opcije (metode) šta ce aplikacija da radi.

**Dodavanje property-ja Vue.js instanci**

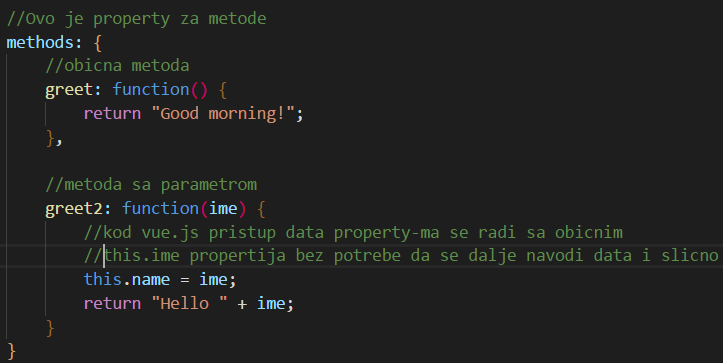
* **el: property** oznacava kojim ce se HTML elementima upravljati ova instanca. U ovom slucaju u html dokumentu postoji prazan div u body-u sa id-jem vue-app, i zato po id-ju nalazimo taj prazan div kojim ce se upravljati.
* **data: property** cuva sve podatke koji ova vue.js instanca ce da prikazuje na stranici. To mogu da budu stringovi, nizovi, funkcije, objekti, itd... U ovom slucaju postavimo property name da ima vrednost John Doe.

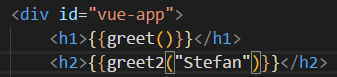
**Prikaz data vrednosti vue instance u html elementu**

* ****Vue.js prikazuje stvari na strani slicno kao kod php-a MySmarty, znaci kao template. Posto smo u js fajlu naveli kojim objektom ce da upravlja Vue instanca sada ako zelimo da ispisemo na strani neku stvar iz data property-ja onda u tom elementu koji se kontroliše samo stavimo vitičaste zagrade i u njima naziv property-ja koji se nalazi u vue instanci data property-ju, i on ce da prikaze tu vrednost.

I tako mozemo da prikazujemo koliko hocemo vrednosti.

**Poziv metoda iz vue instance**

* Isto kao i za podatke koji se smestaju u data property, metode se kod vue instance smestaju u methods: property. Pozivaju se isto u okviru HTML-a isto kao i data property-ji samo uz ().



**Pristup property-ijma vue instance**

* Kod vue.js pristup data property-ma se radi sa obicnim this.ime-property-ja bez potrebe da se dalje navodi data i slicno. Znaci oni su direktno vezani za vue instancu.

**DataBinding**

* DataBinding se koristi za spajanje atributa html elemenata sa vrednostima vue instance. Npr. ako zelimo da postavimo href u <a> html elementu da pokazuje na vrednost porperty-ja „website“ kod vue instance to ne moze da se uradi sa obicnim vitičastim zagradama, vec mora da se koristi kljucna reč **v-bind: .** Znaci ona kaze vue da spoji nesto sa nekim podatkom iz istance. Može i skraćeno da se piše samo sa **:** bez v-bind. U nasem primeru smo postavili v-bind za website property (ime property-ja se u ovom slučaju piše bez vitičastih zagrada i pod navodnicima), a : za name property kod inputa.

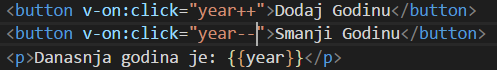
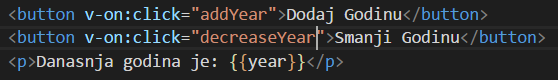


**Obrada html tag-a u Vue.js**

* I vue.js podrzava DOM, tako da i ovde moze dinamicki da se ubacuje html na stanu. Ako zelimo da direktno u vue instanci kreiramo html kod, onda da bi smo prikazali to u html fajlu moramo da unutar odredjenog html elementa dodamo atribut **v-html=“naziv propery-ja“.** I onda ce vue unutar tog elementa da postavi taj html element koji se nalazi kao string u nekom od property-ja.

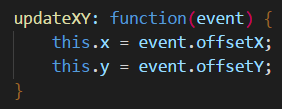
**Obrada evenat-a u Vue.js -** [**https://vuejs.org/v2/guide/events.html**](https://vuejs.org/v2/guide/events.html)

* U javaScriptu postoje razni eventi koji se u Vue.js jednostavno obradjuju sa atributom u HTML-u sa **v-on:“naziv eventa“ = „sta ce se izvrsiti“.** Sta ce se izvrsiti mozemo direktno da napisemo u HTML-u, ali to ne bi trebalo da se praktikuje jer je bolje da se odatle pozove metoda iz externog .js fajla i da se ona izvrsi. Isto kao i kod v-bind: i ovde moze krace da se zapise umesto v-on, kao **@”naziv-eventa”** Neki eventi koji postoje: click, doubleclick, hover, itd...



Direktno u HTML-u Poziv metoda

* Kada se triggeruje bilo koji event tada se kreira objekat **event** koji mozemo da koristimo u funkcijama koje se tada pokrecu. Najcesce se on koristi za koordinate misa tj. offsetX i offsetY.



Kraci zapis

event objekat

**Trigger event samo jednom (Event Modifiers)**

* Moguće je namestiti da se event koji se postavi na neki button ili nesto drugo triggeruje samo prvi put i onda svaki sledeci put nece se nista desiti. To se radi pomocu event modifier koji se zove **once i** ide odmah posle naziva eventa.

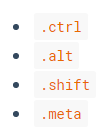
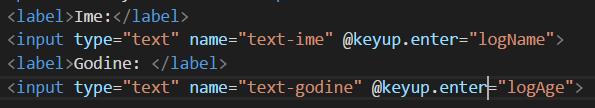


Samo jednom moze da se klikne na button i posle nista

Dodatni Modifieri

**Obrada keyboard evenat-a -** [**https://www.w3schools.com/jsref/dom\_obj\_event.asp**](https://www.w3schools.com/jsref/dom_obj_event.asp)

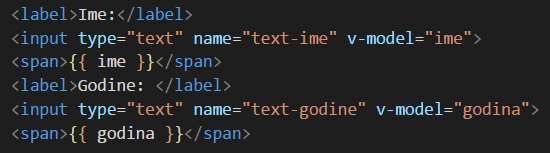
* Kao i obicne evente moguce je i obraditi evente kada se nesto unosi sa tastature, odnosno kad se pritisne i pusti neki taster. Postoje keyup, keydown i keypress. Keyup ce se aktivirati svaki put kada se pusti neki taster. Keydown i keypress se aktivira svaki put kada se pritisne neki taster. Pomoću event modifiers-a mozemo da prekinemo triggerovanje ovih evenata svaki put koriscenjem npr. enter ili alt modifiera.



Kada se unese neki tekst i pritisne enter onda ce se aktivirati funkcija

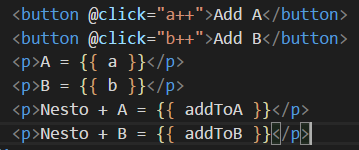
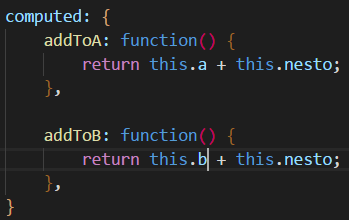
Svi keyboard modifieri

**Two-way data binding – Uporedo updejtovanje vrednosti podataka**

* Npr. ako zelimo da uporedo prikazujemo nesto dok korisnik to unosi onda se koristi two-way data binding. Kada korisnik nesto unosi ono automatski menja vrednost zadatog propertija na unetu vrednost. Koristi se ključna reč **v-model = „naziv-propertyja koji se menja“,** i nije potrebdno da se kombinuje sa keyboard eventima.

Kako se unosi ime on se odmah stampa u span element tj. Vrednost property-ja se automatski menja i prikazuje se u span elementu

**Computed properties – izvrsavanje tacno jedne funkcije**

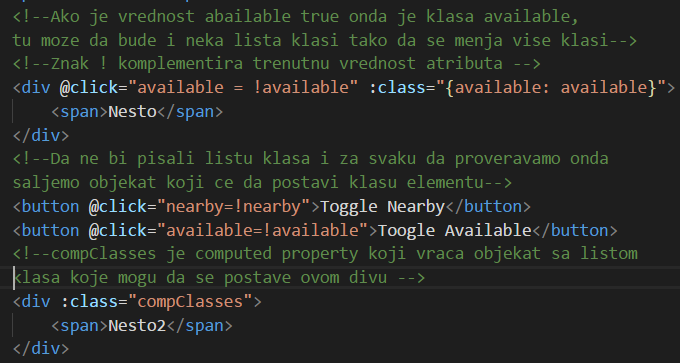
* Kod vue.js se desava da pri bilo kom eventu kada se menja neka vrednost desava se da se pozivaju sve funkcije na strani, iako ce samo jedna funkcija biti izvrsena i vrednost samo jedne promenljive ce se promeniti idalje ce se pokrenuti sve druge funkcije bez da ista urade. Da bi se to sprecilo i da bi se tacno jedna funkcija pozivala kada treba koriste se **Computed Properties** koje su iste kao i metode samo sto se ne pozivaju kao one vec kao obicna promenljiva. Oni se pozivaju samo kad se menja tacno ta promenljiva za koju su namenjeni. U vue.js instanci oni se nalaze u posebnom objektu **computed:**.

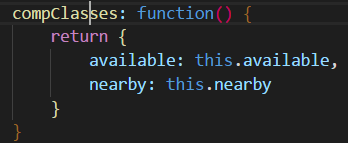
addToA i addToB su Computed Properties

Pisu se isto kao obicne funkcije

**Dinamički CSS – postavljanje atributa dinamički**

* Unutar HTML-a pomoću vue.js mozemo dinamicki da postavljamo i menjamo atribute u zavisnosti od vue.js vrednosti u instanci. Mora da se piše unutar vitičastih zagrada. I onda za sve vrednosti koje su true postavice se ime klase na tu vrednost (moze imati i vise klasa).





Primer direktnog proveravanja i sa objektom