처음 접하는 VueJS + NodeJS

Vue.JS

김대희

목 차

01

VueJS란 ?

Vue.JS란?

SPA(Single Page Application) 및 뷰 모델을 가지는
MVVM(Model-View-ViewModel) 패턴을 기반으로 하는 JavaScript 라이브러리

SPA / MVVM이란 ?

SPA

최초 한번 페이지전체를 로딩한후 이후부턴 데이터만
변경해서 사용할 수 있는 웹 어플리케이션

MVVM

MVC와다르게 ViewModel이
Controller 역할을함

두 가지 특성이 있는 Front-End 라이브러리이다 !

02

VueJS Instance

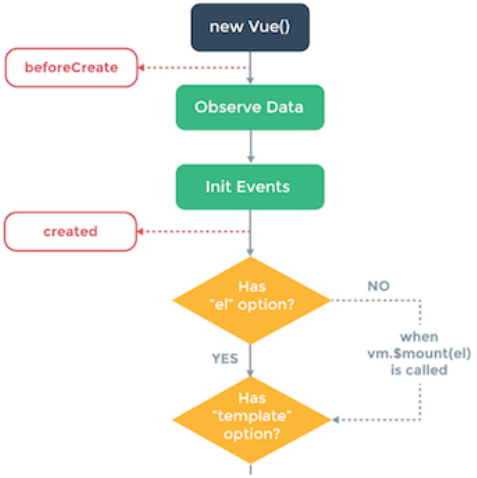
Vue라는 객체에서 Data, Event, Method , 속성을
포함하여 화면의 단위를 제어할 수 있게 해주는 것

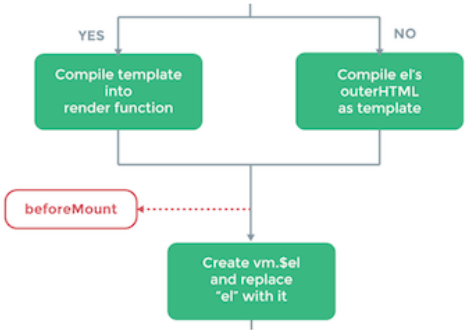
```
Var 변수 = new Vue({

    // Data , ~~ Method 등

})
```

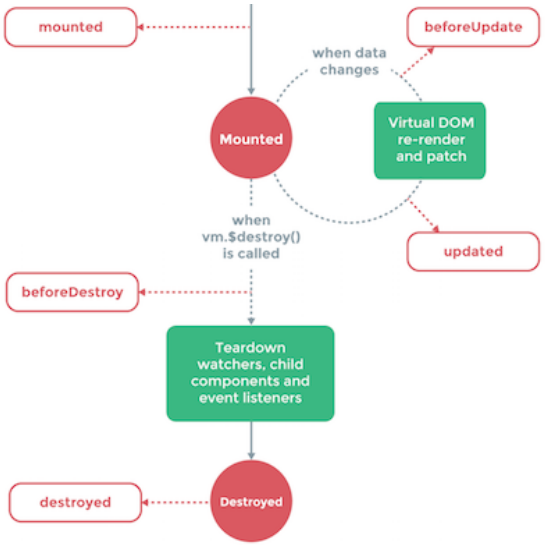
뷰 관련 옵션 : el, template
데이터 관련 옵션 : data, methods, computed
컴포넌트 관련 옵션 : components
생명 주기 훅 : created, mounted, updated, destroyed

02

VueJS Instance 생성주기







- Vue가 생성되면 event 함수 초기화
- el, template이 있는 지 확인 (template – 화면 구성요소 , el – 어떤 구성요소 인지)
- created, mounted, updated, destroyed

* beforeCreate, create는 해당 page 실행시 실행
* Mounted는 el이 있을시 적용
* Updated는 해당 el요소에 변동이 있을시 실행

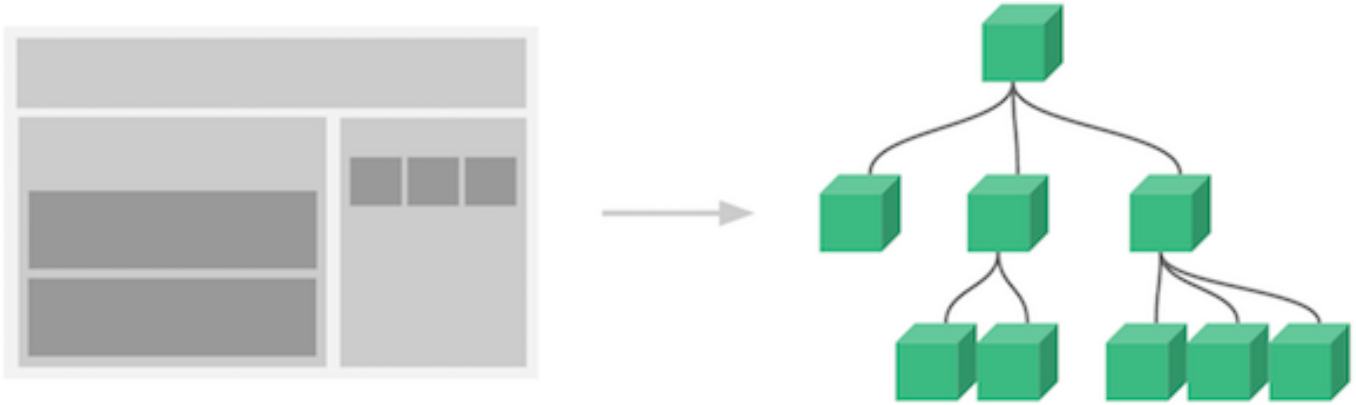
02

VueJS 요소

```
new Vue({
 template : vue에 적용시킬 template( html 요소··· model data 등)
 el: vue가 실행될 요소 ( div, img··· tag )
 methods: vue에서 적용시킬 method
 ex) v-on:click="instanceclick" 등을 사용하여 event method 처리
data: vue에서 적용시킬 data
});

// 생명주기 + Vue.JS Instance 요소만 있어도 기본적인 기능 가능.

// html 사용시
<!-- {{}}을 사용해서 vue 형식을 적용할 수 있습니다. -->
ex) <p>{{ instance }}</p>
```

03

VueJS Component



- 화면에 비춰지는 뷰의 단위를 쪼개서 재활용이 가능한 형태 관리
- 한 페이지에 있는 요소들을 각각 다르게 관리하고 상,하위 계층이 있다는 것

03

VueJS 전역 Component

```
Vue.component('my-component',{// vue.component를 사용하여 직접 vue객체에 component 등록
template : '<P> 컴포넌트 예시입니다.</P>'
});
new Vue({ // new Vue를 이용하여 el에 있는 요소에 component 등록
el: '#component'
})

//vue.component를 생성후 아래에 계속 new Vue를 만들면 해당 component들이 el요소에 적용됨
```

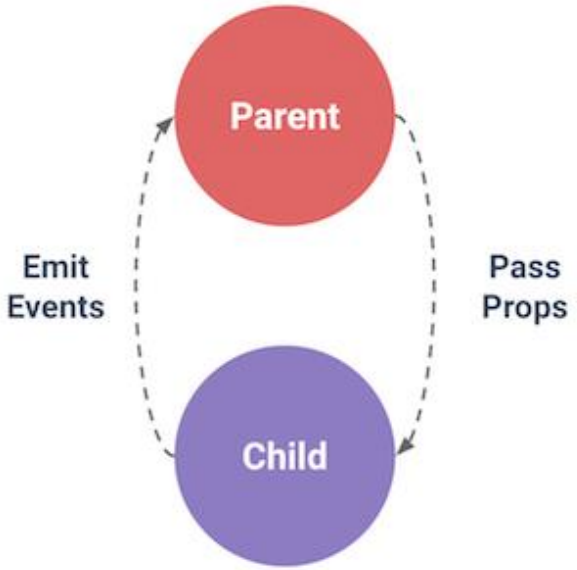
03

VueJS 지역 Component

```
var localcmp = {// component로 사용할 var를 만든다.
Template : " <P>지역 컴포넌트</P> "
}

new Vue({
El: ' #local ' ,
components:{
 ' local-component ' : localcmp
}
}) // new Vue 객체 생성시 component로 local 등록을 한다.
// 전역 컴포넌트는 Vue.component로 직접 접근, 지역  컴포넌트는 변수를 만든 후 component에 적용
```

03

VueJS 상 하위 Component



- 상위 -> 하위 컴포넌트 전달 : props 사용 (하위 -> 상위는 바로 불가능)
-

03

Props 코드 예시

```
Vue.component('child-component',{ // 해당 html <child-componemt> 요소를 작성
props: ['childdata'], //
template : '<p>{{childdata}}</p>' //v-bind:'childdata' 와 일치한 이름 작성
})
var app = new Vue({
el : '#parent', // 해당 component를 적용할 요소
data :{
outofchild : 'Hello , this Test Data ', // v-bind:childdata ="outofchild" 부분에 적용할 부분

}
})
// props를 이용하여 template등의 요소를 미리 정할 수 있다.
```

04

VueJS Router

- Vue를 이용한 SPA 제작시 유용한 라우팅 라이브러리

04

Router를 적용할 html page

```
<div class="col-md-4" id="routerdiv">
<!-- router 처리 -->
<router-link to="/test1">Go to test1</router-link>
<router-link to="/test2">Go to test2</router-link>
<router-view></router-view>
</div> // router를 적용한 div id, router에 연결할 link , router - view를 이용
<!-- vue router js --> // 아래쪽에 넣어놓은 이유는 위에 적용시 router를 적용할 div를 인식 못함.
<script src="/javascripts/vue-router/dist/vue-router.js"></script>
<script src="/javascripts/router.js"></script>
```

04

Router를 적용할 Router JS 페이지

```
var Test1 = { template: '<div>test1</div>' }
var Test2 = { template: '<div>test2</div>' }
// router 내용 설정
var routes = [
{path: '/test1', component: Test1 },
{path: '/test2', component: Test2 },] // router 연결 설정
//router 생성
var router = new VueRouter({
routes // router 변수 적용
})

var app = new Vue({
router
}).$mount('#routerdiv') // mount하여 해당 el 요소에 적용
// 이렇게 route를 설정함으로서 얻는 점은 서버를 통하지 않고도 해당 뷰만을 로딩하여 바꿀 수 있다는 점.
```

04.5

VueJS Template

v- bind : 사용시 html 요소와 대응

{{ 변수이름 }} : vue.js 표현

v- 라는 tag가 있으면 directives 처리

05

Axios를 이용한 DataBase 코드

```
var url = "http://localhost:3000/get"; // 해당 js 파일로 보낼 http url
var instance = new Vue({
// template : "<p>안녕하세요!</p>",// 화면등에 나오는 요소 ( 정적 ) + Model Data 받아옴
el : "#instance", // Vue element가 그려지는 요소
data:{
instance : ""},
methods:{
instanceclick : function(){
axios.get(url) // axios를 이용한 get 요청 ( HTTP -Module ) // axios를 통해 ejs -> js -> ejs 연결이 가능합니다.
.then(response => {
// JSON responses are automatically parsed.
this.instance = response.data.rows[0].idx; // JSON 형식으로 받아온 Data를 표시
})
.catch(e => {
this.errors.push(e)
})}})};
```

05

Axios를 이용할때 Node.js 정보 넘기기

```
router.get('/get', function (req, res, next) { // 보내온 url을 통해 js파일로 넘어옴
db.getPool().getConnection(function (err, connection) { // mysql과 연동
// Use the connection
  connection.query('SELECT * FROM board limit 1', function (err, rows) // 쿼리를 받아옴{
  if (err) console.error("err : " + err);
  console.log("rows : " + JSON.stringify(rows));
  res.json({rows: rows}); // 해당 정보를 JSON으로 보냄
  connection.release(); // connection 종료
  console.log("종료 확인");
// Don't use the connection here, it has been returned to the pool.
});
});
});
// 예전에는 이러한 역할을 Node - resour에서 처리하였으나  현재는 Axios를 통해 정보를 처리합니다.
```

Q&A

감사합니다