beauty코드 이쁘게 정렬

setTimeout(() => {},2000); ->이런식으로 시간을 지정하는 건 권장하지 않음!(고객마다 인터넷,PC환경에 따라 속도가 달라질 수 있음)

동기식 처리 : 코드가 작성된 순서대로 실행

비동기식 처리 : 작성 순서와 상관없이 지정된 순서에 의해 실행

callback : 하나의 함수 안에 다른 함수를 호출

fucntion 특정함수(function callback){

특정조건:

callback()

}

sync는 현재 사용할 함수를 비동기로 처리하겠다는 선언자

await는 비동기로 순차 처리하기 위해서 수행할 API에 붙이는 선언자

acync 는 어떤 값을 return 하던간에 Promise 가 리턴된다.

async function 함수명 () { await 비동기\_처리\_메서드\_명() ; }



lintOnSave : false

component 쓰는 3가지 step

1)분리한 컴포넌트를 부모에서 import 한다.



2)components : {}에 만든 컴포넌트의 이름 등록



3)원하는 곳에 가져다 쓰기



props 3스텝 - read-only임 수정XX



1)원하는 곳에 자식 컴포넌트를 가져다 쓴다.

근데, 그냥 <자식 컴포넌트 이름/> 이라고 쓰는 게 아니라 v-bind를 같이 써준다.

<자식컴포넌트이름 v-bind: 자식 데이터 = "부모 데이터 이름"/>



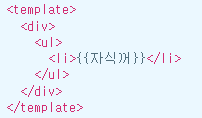
\* v-bind -> ‘:’으로도 사용가능

2)자식에서 props 등록

- 물려받은 데이터를 props라고 생각하면 됨



3) 부모한테서 가져온 데이터를 자유롭게 사용한다.



v-for와 props 바인딩

아무리 컴포넌트화 한들, 데이터 개수가 xxx 개라면 v-for를 사용할 수밖에 없다!

자식 컴포넌트 가져다 쓰는 부분에 for문을 같이 돌리는 방법을 알아보자.

1)v-for와 v-bind를 동시 사용한다.

v-for = “아이템 in 데이터”

v-bind : 전해줄 아이템=“아이템”



\*v-bind -> ‘:’으로도 사용가능

2)똑같이 props 등록 후 자유롭게 사용





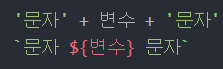


:style=”~”집어넣을 때 object 자료도 가능

백틱기호 or 쿼테이션 `



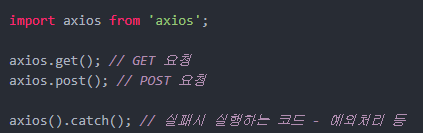
변수작성

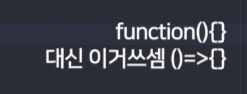


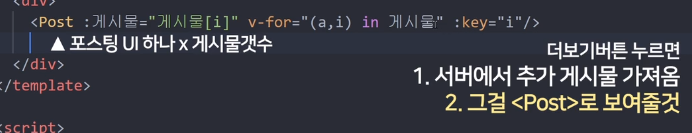
[element 이름] v-for="(value, key) in [반복가능한 변수 이름]">

Get 요청 : {url로 요청} 데이터 서버에서 가져올 때

Post 요청 : {url로 요청} 서버로 데이터를 보낼 때







참고글 : <https://stackoverflow.com/questions/27120757/failed-to-execute-createobjecturl-on-url>

https://jw910911.tistory.com/119

createObjectURL가 더 이상 Google Chrome용 미디어 스트림 개체를 허용하지 않기 때문에 발생

slot 로 부모->자식 데이터 전송

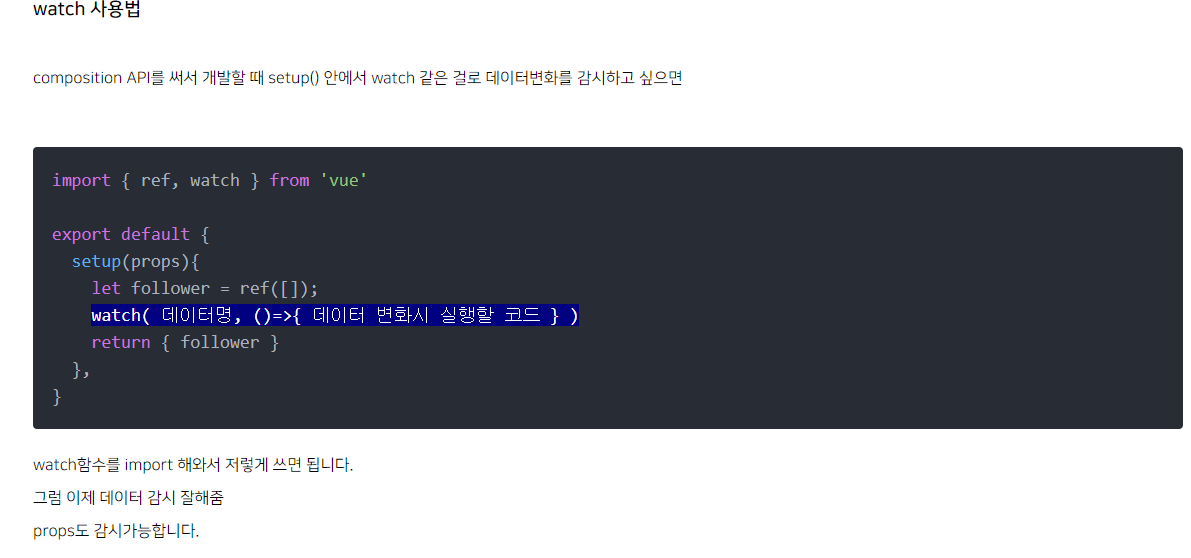
1. 자식은 구멍 뚫기
2. <컴포넌트>태그 사이에 데이터 넣기
3. 태그 안에서 데이터바인딩 할때만 사용 가능

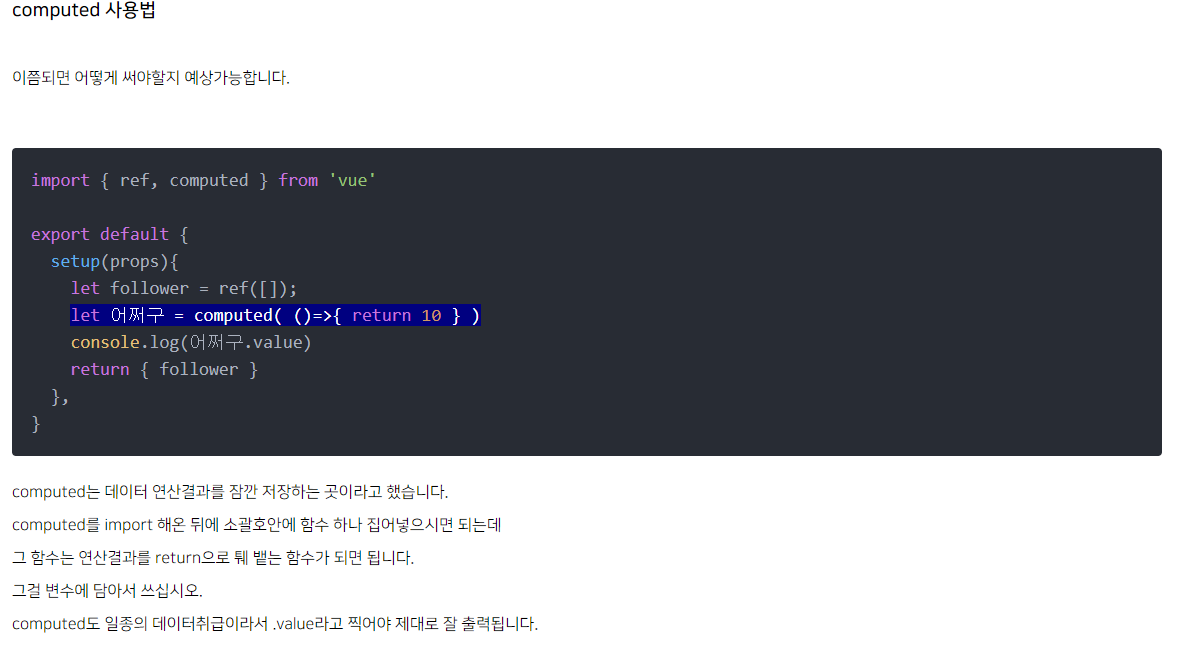
자식 데이터 부모가 써야될 경우는 slot props사용

1. <slot :자식데이터=”자식데이터”>
2. 부모는<template v-slot=”작명”>후에{{작명.자식데이터}}

Commit : mutations 부탁

Dispatch : actions 부탁









style scoped:현재 파일만 css적용

데이터만들 때 ref() reactive()

methods, watch, component, props, event, mitt, v-if, v-for

- 실시간 렌더링이 필요하면 데이터만들고 데이터바인딩

- 함수가 필요하면 methods

- HTML이 너무 길면 component

- 컴포넌트간에 데이터 주고받고 싶으면 props, custom event, mitt

- 데이터 감시하고 싶으면 watch

swiper : https://swiperjs.com/ 접속해서 다운

https://m.blog.naver.com/anedthh/222014406404

----------장고

{% 탬플릿태그 %} : 해당 변수의 값을 그 자리에 치환

block/endblock

Django가 html을 렌더링할 때 쓰이는 문법으로 html 기본 base를 상속받아 렌더링할 때

특정 부분에 해당하는 부분만 상속받은 자식 html 내에서 원하는 양식으로 변경하고 싶을 때

쓰이는 문법

Template 파일로 만들어 코드를 재사용

베이스 템플릿의 block과 endblock 사이에 중복되지 않는 코드

# Django - views.py와 urls.py, url 매핑 규칙, 템플릿, 스태틱 파일(정적 컨텐츠),

# 장고 템플릿의 블락(block)

<https://lar542.github.io/Django/2019-06-17-first_django_ptoject/>

# Django에서 제이쿼리 load와 장고의 extneds, include 를 복합적으로 이용한 페이지 만들기 - 페이지 참조 및 템플릿 확장

<https://my-repo.tistory.com/39>

Django 공식 사이트

<https://tutorial.djangogirls.org/ko/template_extending/>

$과 \_

https://eotkd4791.github.io/javascript/JavaScript15/

자바스크립트 유효성 검사

https://beop07.tistory.com/7

var x = document.getElementsByName("csrfmiddlewaretoken")

const headers = {"X-CSRFTOKEN": x[0].value}

let url = "/partner/hm\_checkplus\_start"

let formData = new FormData();

formData.append('fcn', "join")

axios.post(url, formData, {headers: headers})

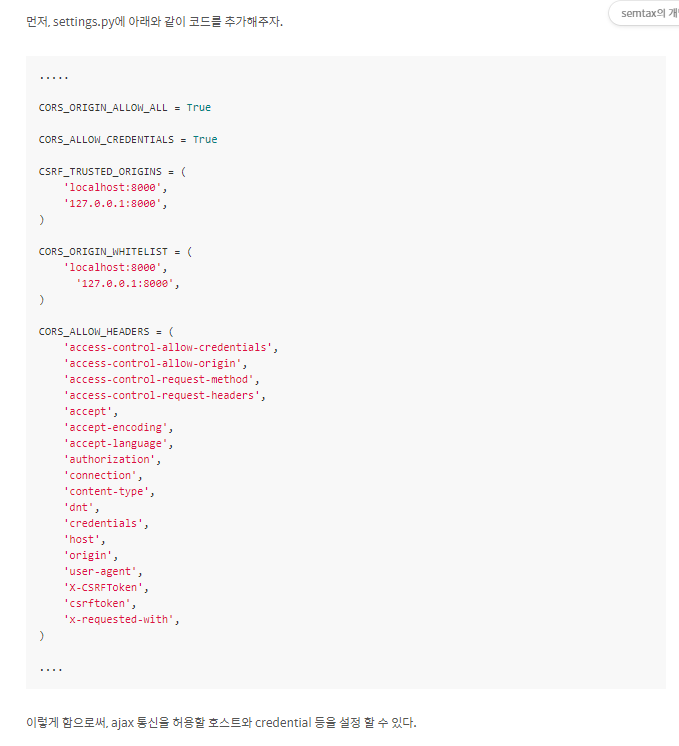
<https://integer-ji.tistory.com/362>

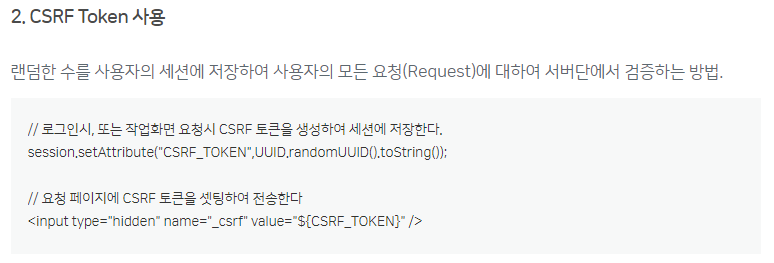
csrf : Cross-site request forgery 공격의 약자로, 사용자가 자기 의지와는 상관없이

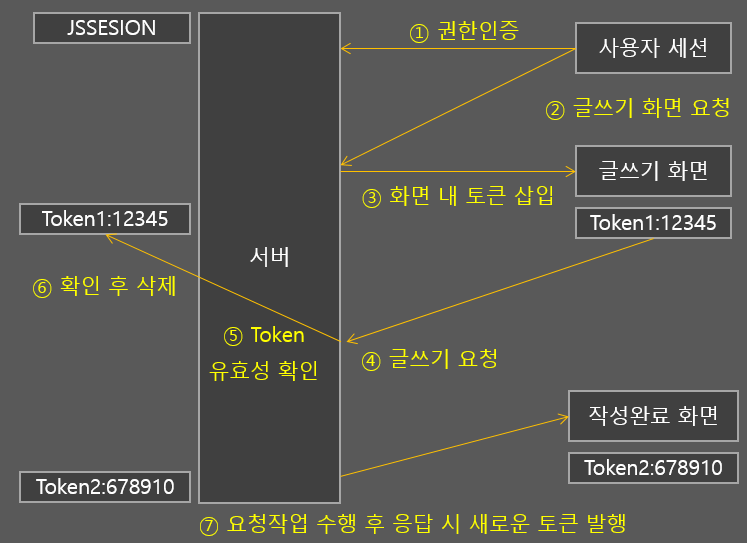
공격자가 의도한 행위를 특정 웹사이트에 요청하게 하는 방식으로 수행하는 공격

Django 에서는 CSRF 토큰 이라는 일종의 난수 비슷한것을 도입

<https://docs.djangoproject.com/en/3.0/ref/csrf/>







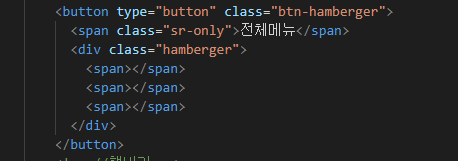
정규표현식

<https://blogpack.tistory.com/560>

* formData.append(name, value)
* formData.append(name, blob, fileName)
* formData.set(name, value)
* formData.set(name, blob, fileName)

widthratio : 주어진 값과 최대 값의 비율을 계산한 다음 주어진 값에 비율을 적용함.

{# 템플릿 태그 : {%명령%} 형식으로, python 명령을 실행한다 ( if, for 등등 ) #}



for [변수] in [문자열, 리스트, 튜플]:

[수행부분]

<https://blockdmask.tistory.com/379>

