20180208 8회차

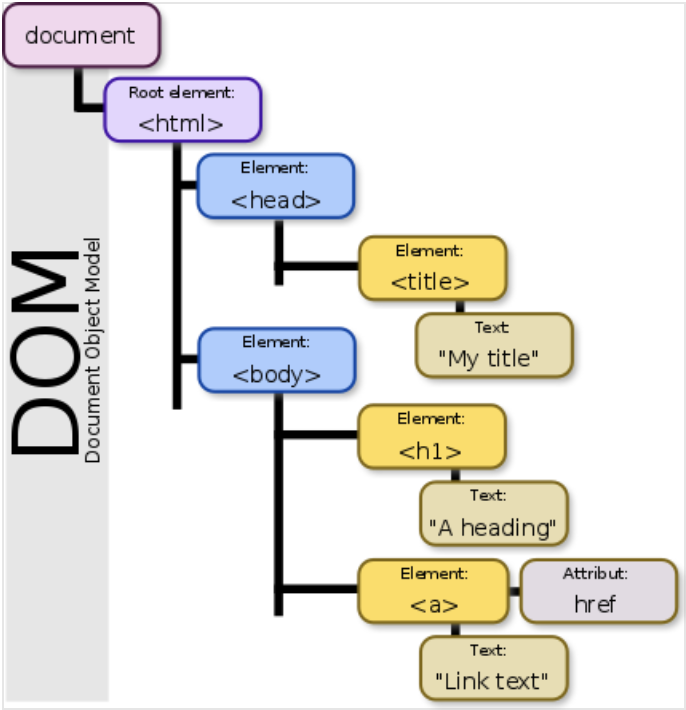
JAvascript

# Dom

Dom이란 브라우저에서는 HTML코드를 DOM이라는 객체형태의 모델로 저장한다.

이저장된정보를 DOM Tree라고 한다.

결국 HTML elemet는 Tree형태로 저장된다



## DOM의 예시

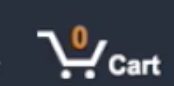
### getElementById()

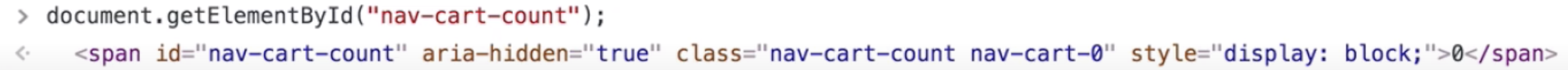
id정보를 통해서 찾는다.

ex) document.getElementById(“id값”);

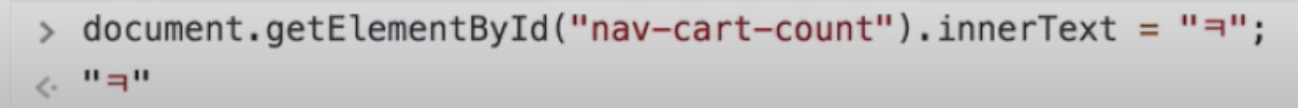
id값에 해당되는 것을 찾아준다.

ex2) document.getElementById(“id값”).innerText=”ㅋ”;





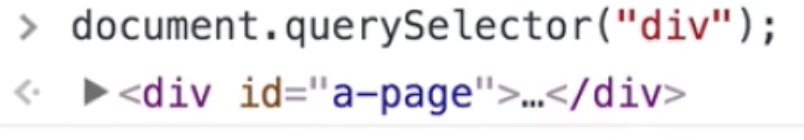




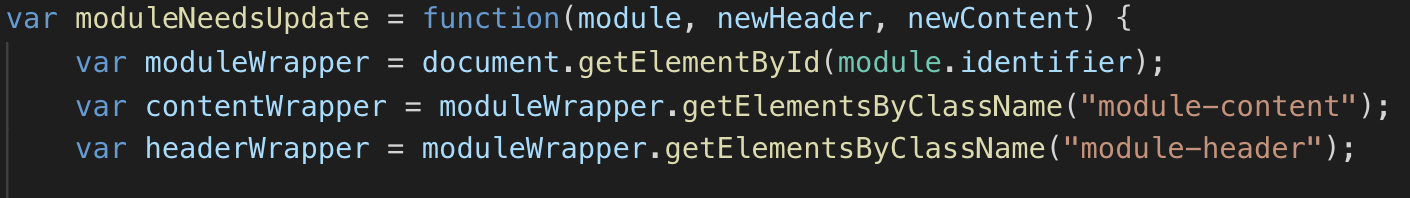
### querySelector(“div”);

div값만 찾아준다.

ex)document.qureySelector(“div”);

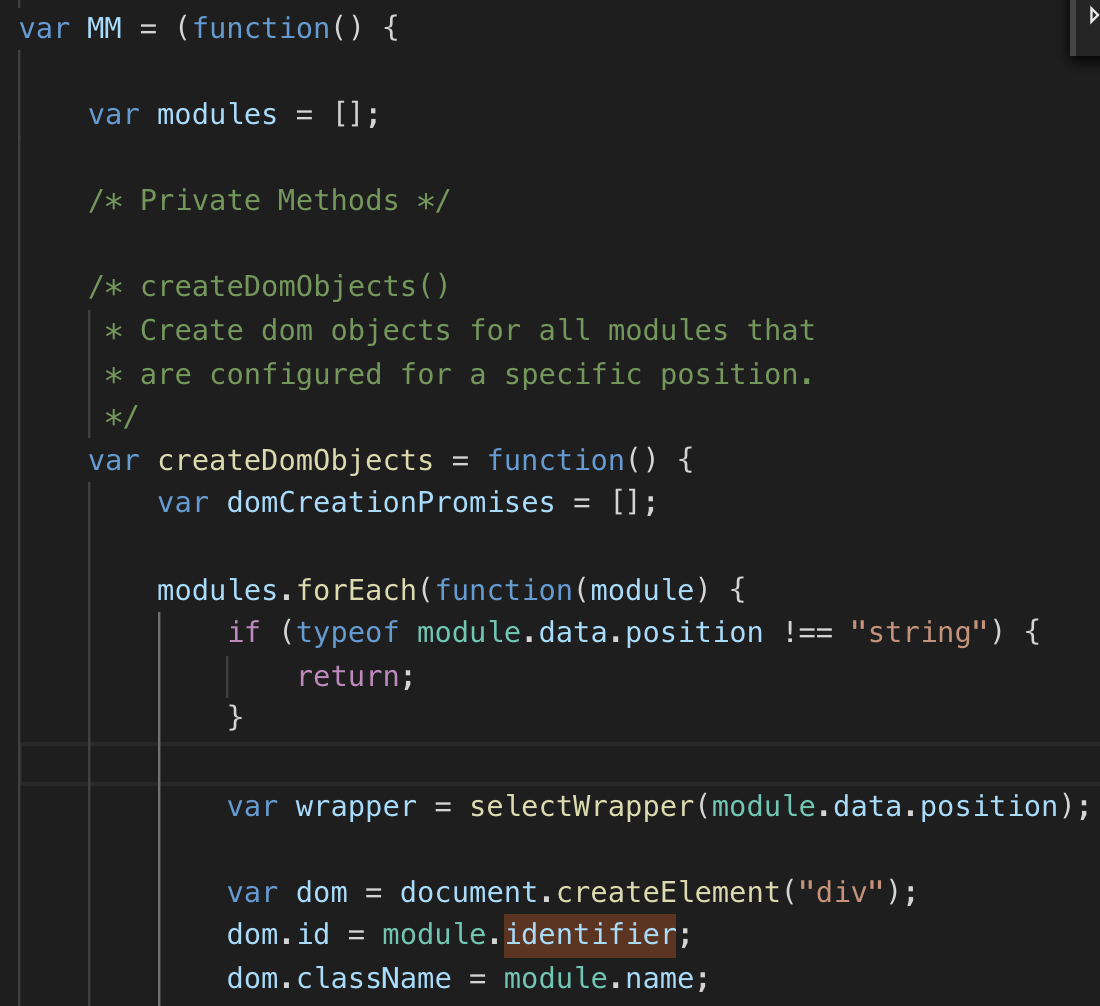


## 실전 응용



var contentWrapper = moduleWrapper.getElementsByClassName("module-content");

// module-content라는 클래스를 찾고 그클래스는 contentWrapper로 지정한다 라는 뜻



module.identifier는 해당 dom의 id로 넣어준다는 뜻

# EVENT

마우스로 조절, 스크롤 , 클릭, 이동 키보드 입력 등등이 EVENT

표준방법

var el = document.getElementById(“outside”);

el.addEventListener(“click”, function(){

//do something

}, false);

두번째인자 함숮는 이벤트가 발생할때 실행되는 함수로 이벤트 핸들러 또는 이벤트 리스너라고한다.

## addEventListener

-test.html

<html>

<header></header>

<body>

<div class="outside">outside element</div>

</body>

<script src="test.js"></script>

</html>

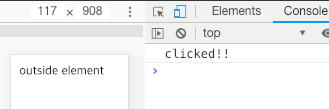
-test.js

var el = document.querySelector(".outside");

el.addEventListener("click", function(){

console.log('clicked!!');

});



마우스로 클릭을 하면 clicked가뜬다

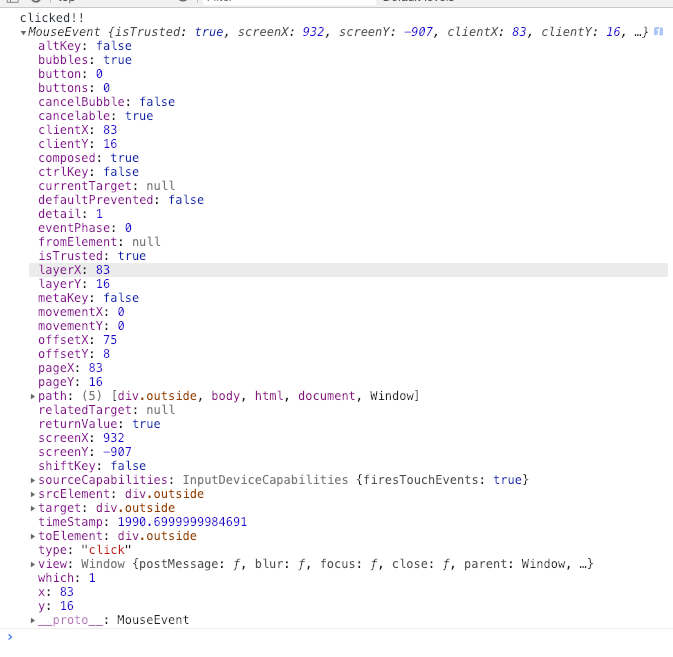
-test.js수정

var el = document.querySelector(".outside");

el.addEventListener("click", function(e){

console.log('clicked!!',e);

});



클릭하면 마우스정보(event object)도 같이 콘솔창에 같이뜬다.

응용->

var el = document.querySelector(".outside");

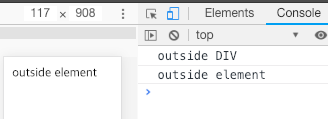
el.addEventListener("click", function(e){

var target = e.target;

console.log(target.className, target.nodeName);

console.log(target.innerText);

});

클릭한 것(outside클래스)의 정보가 뜬다.

\

var el = document.querySelector(".outside");

el.addEventListener("click", function(e){

var target = e.target;

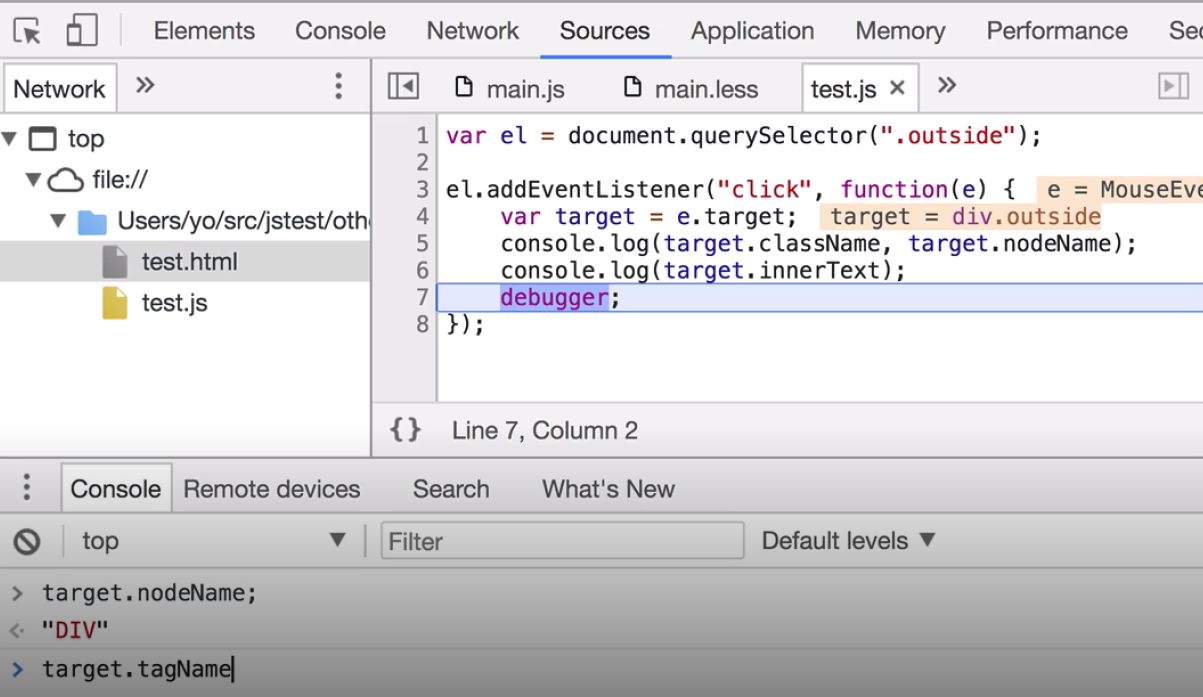
console.log(target.className, target.nodeName);

console.log(target.innerText);

debugger;

});

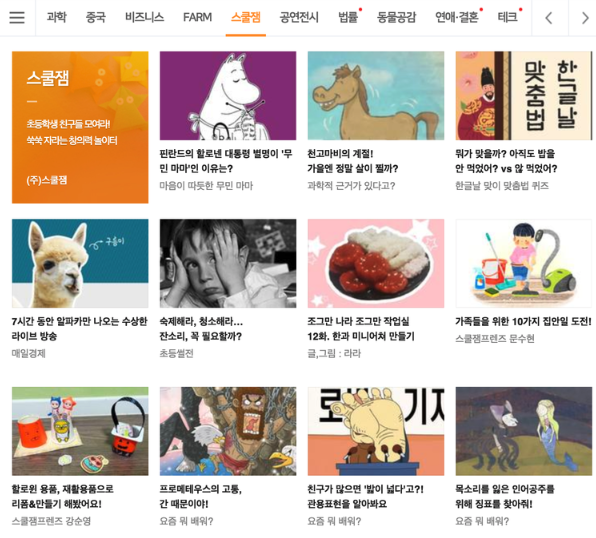
클릭을 하면 디버거가 실행된다.



이런식의 활용가능

# Ajax

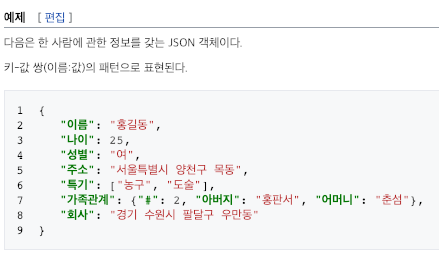
웹에 데이터를 갱신할때, 브라우저 새로고침 없이 서버로부터 데이터를 받는 것.



예를 들어서 탭을 다른곳으로 누르면 그 전체다 새로고침이 아니라 탭부분만 바뀌어서 보이는 것이다. 이것을 비동기적으로 서버로부터 데이터를 받는다고 한다.

이때 json으로 데이터를 주고받는다. json이란 파일형식이다.

<https://ko.wikipedia.org/wiki/JSON> 참고를 해보면

json의 예제 형식이다.

객체와 표현방식이 같다는 것을 알 수 있다.

이런식으로 json.txt로 저장

function ajax(data) {

var oReq = new XMLHttpRequest();

oReq.addEventListener("load", function() {

console.log(this.responseText);

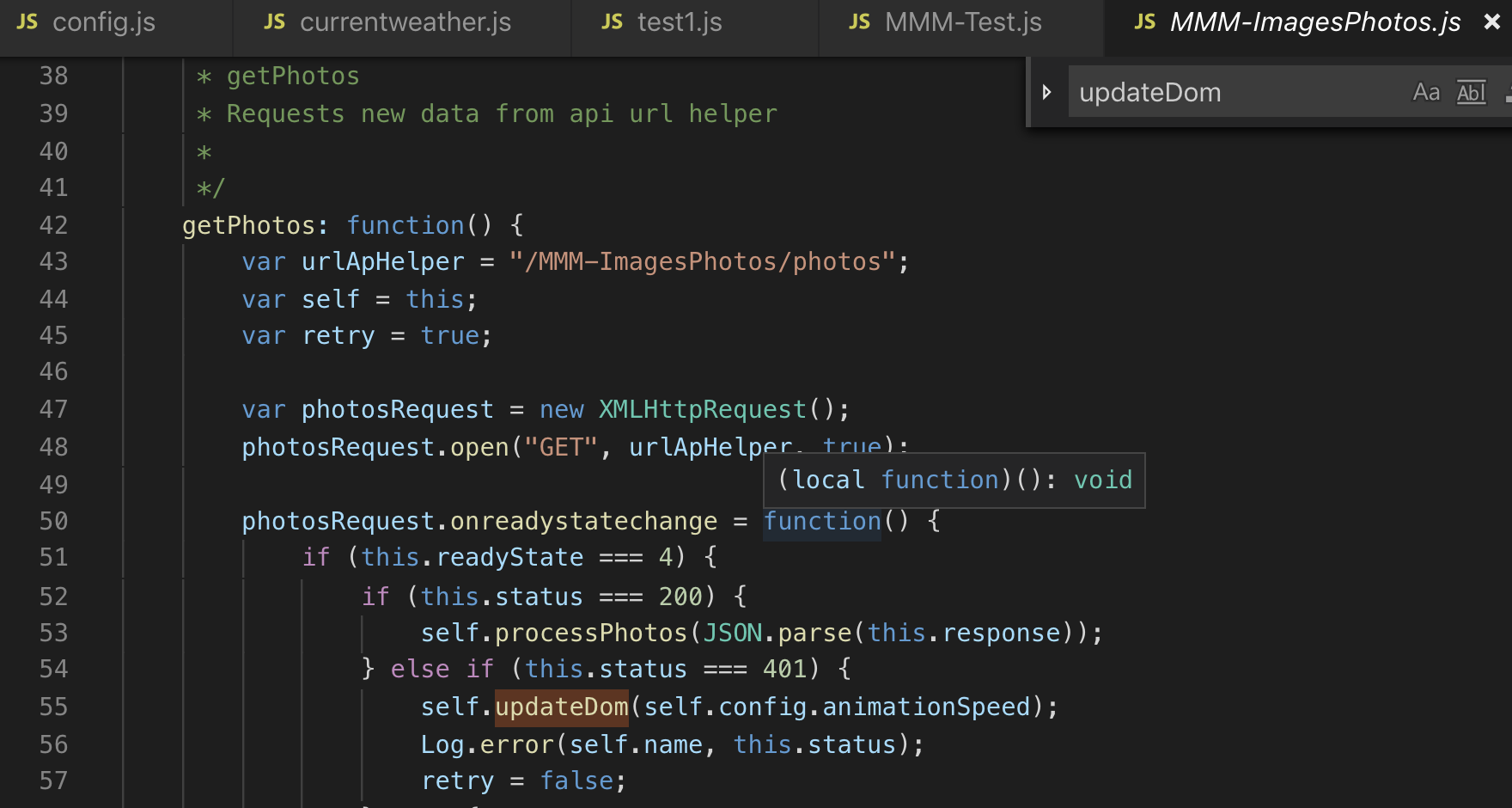
});

oReq.open("GET", "http://www.example.org/getData?data=data");//parameter를 붙여서 보낼수있음.

oReq.send();

}

실제 스마트미러 내의 imagesPhoto.js 의 module중 관련되어있다.



var urlApHelper = "/MMM-ImagesPhotos/photos";

        var self = this;

        var retry = true;

        var photosRequest = new XMLHttpRequest();

        photosRequest.open("GET", urlApHelper, true);

확인가능참고로 XMLHttpRequest는 http프로토콜만 사용하는 파일만 가능하다.

updateDom에대하여도 설명

# 모듈 만들기

modules: [

        {

            module: "MMM-Test",

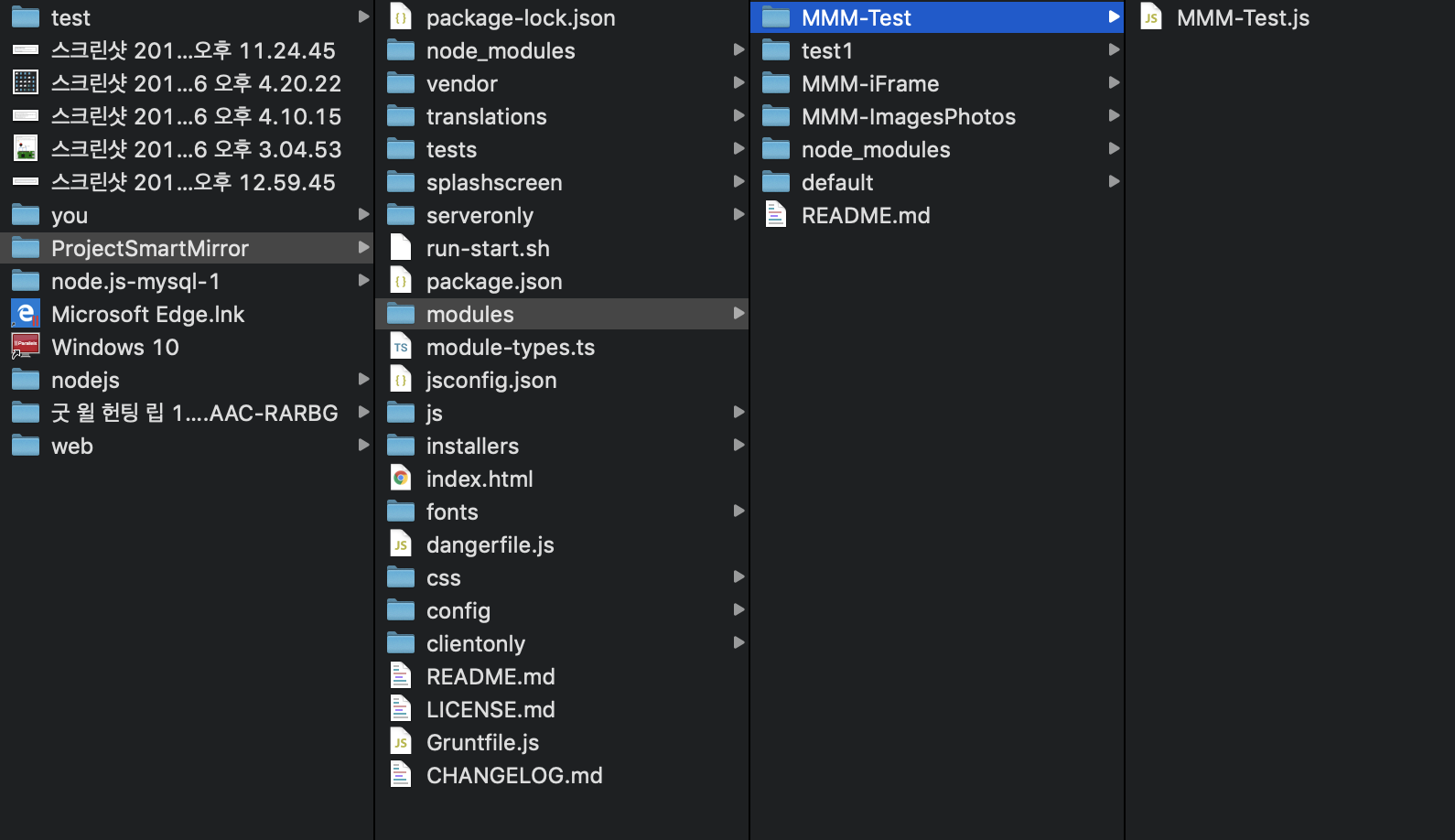
            position: "top\_left",

            config: {}

         },

    ]

config.js에 연습용 모듈 하나만 해놓음



하나의 js파일을 만들고

Module.register("MMM-Test", {

defaults: {},

start: function () {},

getDom: function() {},

notificationReceived: function() {},

socketNotificationReceived: function() {},

})

다음과 같이 기본골격만 만들어놓았다.

## getDOM

getDom을 알기 위해 getdom에 다음과 같이 추가

Module.register("MMM-Test", {

defaults: {},

start: function () {},

getDom: function() {

var element = document.createElement("div")

element.className = "myContent"

element.innerHTML = "Hello, World!"

return element

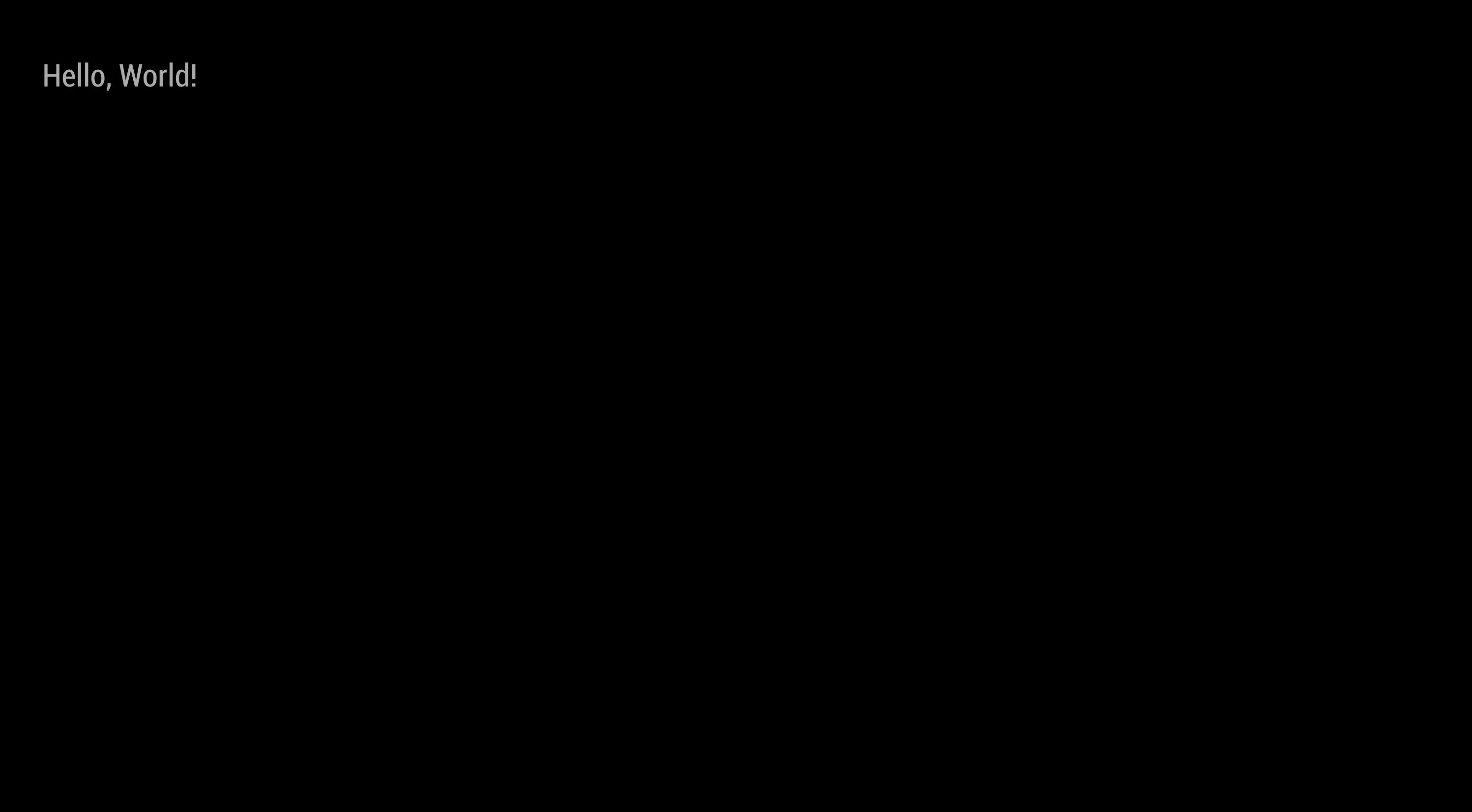
},

notificationReceived: function() {},

socketNotificationReceived: function() {},

})

실행화면



확인 완료

dev에 대하여

## Defaults:{}

Module.register("MMM-Test", {

defaults: {

foo: "I'm alive!" //추가

},

start: function () {},

getDom: function() {

var element = document.createElement("div")

element.className = "myContent"

element.innerHTML = "Hello, World!" + this.config.foo //추가

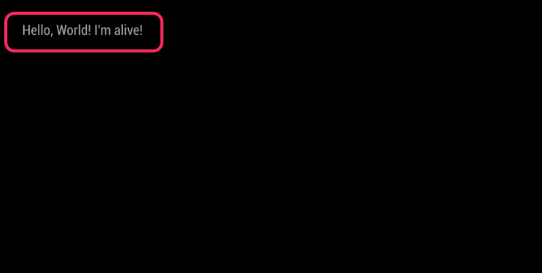
return element

},

notificationReceived: function() {},

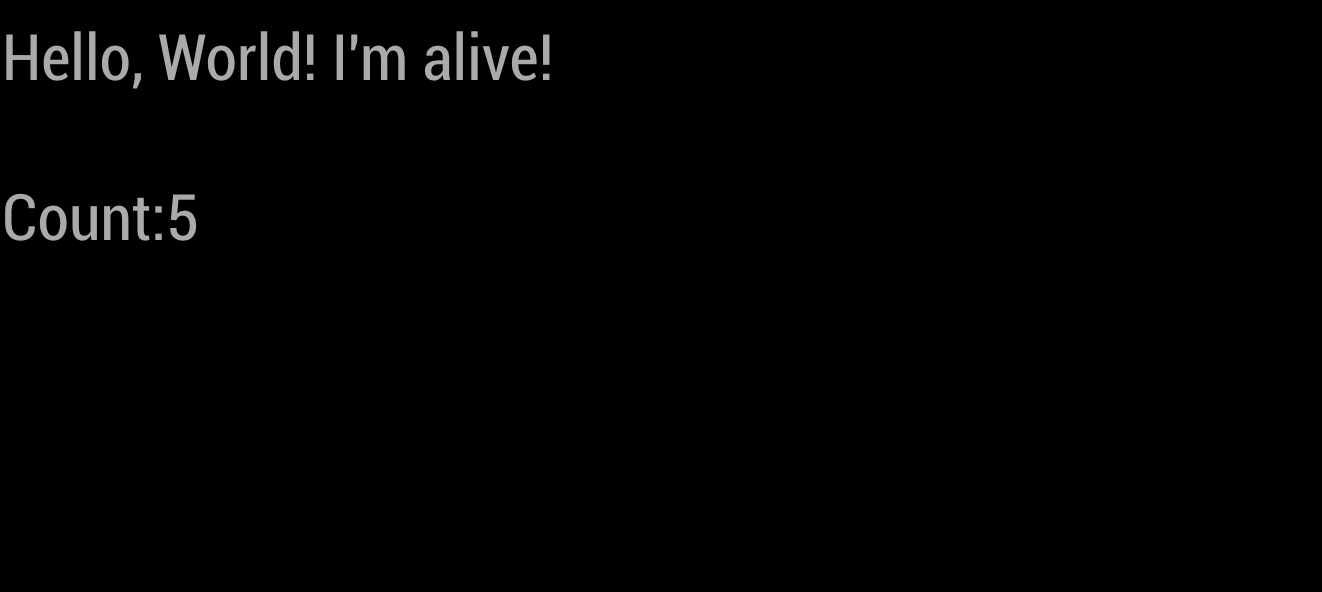
socketNotificationReceived: function() {},

})



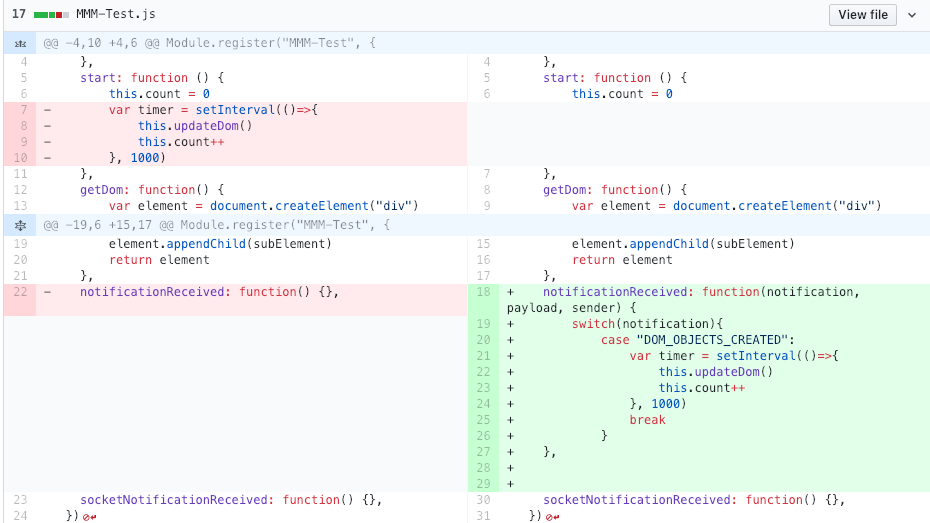
## START()





updatedom이란 getDom()이 output을 갱신 하여다시 그려주는 것. 즉 getDom을 갱신하는 것이다.

## NOTIFICATION



Module.register("MMM-Test", {

defaults: {

foo: "I'm alive!"

},

start: function () {

this.count = 0

},

getDom: function() {

var element = document.createElement("div")

element.className = "myContent"

element.innerHTML = "Hello, World! " + this.config.foo

var subElement = document.createElement("p")

subElement.innerHTML = "Count:" + this.count

subElement.id = "COUNT"

element.appendChild(subElement)

return element

},

notificationReceived: function(notification, payload, sender) {

switch(notification){

case "DOM\_OBJECTS\_CREATED":

var timer = setInterval(()=>{

this.updateDom()

this.count++

}, 1000)

break

}

},

socketNotificationReceived: function() {},

})

아까와의 차이

The result looks so same with previous codes. But your timer will be started after when “DOM\_OBJECTS\_CREATED” notification is received but not when the module is loaded.  
DOM\_OBJECTS\_CREATED notification is emitted when all of modules are loaded and rendered(with getDom()) first time. At the moment, you can manipulate your output in your module.

파일에서 그림을 삽입하거나 도형, 텍스트 상자 또는 표를 추가하고 싶으세요? 모두 추가할 수 있습니다. 리본 메뉴의 [삽입] 탭에서 필요한 옵션을 탭하면 됩니다.

[삽입] 탭에서 하이퍼링크를 추가하거나 메모를 삽입할 수 있는 사용하기 쉬운 도구를 찾아보세요.