Node -> node.js교과서 p85~p130

참고 : <https://developer.mozilla.org/ko/docs/Web/JavaScript>

프런트엔드 자바스크립트

-AJAX(Asynchronous JavaScript And XML): 비동기적 웹서비스를 개발할때 사용하는 기법(페이지이동없이 서버에 요청을 보내고 응답을 받는 기술

요청은 jquery나 axios같은 라이브러리를 이용하여 보낸다

코드파일경로 : node/ch2 /front.html

<script src="https://unpkg.com/axios/dist/axios.min.js"></script>

<script>

axios.get('https://www.zerocho.com/api/get')//axios.get내부에 new promise가 들어있어 then과 catch를 사용할수있다

.then((result)=>{

console.log(result);

console.log(result.dat a);

})

.catch((error)=>{

console.log(error);

});

(async()=>{//promise이므로 async/await 방식으로 변경할수있다

try{

const result = await axios.get("https://www.zerocho.com/api/get");

console.log(result);

console.log(result.data);

}catch(error){

console.log(error);

}

})();

(async()=>{

try{

const result = await axios.post("https://www.zerocho.com/api/post/json",{

name:'zerocho',

birth: 1994,

});

console.log(result);

console.log(result.data);

}catch(error){

console.log(error);

}

})();

</script>

-FormData

HTML form 태그의 데이터를 동적으로 제어할수있는기능

주로 AJAX와 같이사용된다

코드파일경로: node/ch2/formdata.html

FormData조작방법

    <script>

        const formData =new FormData();

        formData.append('name','zerocho');//append메서드로 키-값형식의 데이터를 저장할수있음

        formData.append('item','orange');

        formData.append('item','melon');

        console.log(formData.has('item'));//true,false반환

        console.log(formData.has('money'));

        console.log(formData.get('item'));

        console.log(formData.getAll('item'));

        formData.append('test',['hi','zero']);

        console.log(formData.get('test'));

        formData.delete('test');

        formData.set('item','apple');//set은 현재키를 수정하는 메서드

        console.log(formData.getAll('item'));

    </script>

실행결과



-axios로 폼데이터를 서버에 보내기

코드파일경로: node/ch2/formdata.html

        (async()=>{

            try{

                const formdata = new FormData();

                formdata.append('name','zerocho');

                formdata.append('birth',1994);

                const result = await axios.post('https://www.zerocho.com/api/post/formdata',formdata);

                console.log(result);

                console.log(result.data);

            }catch(e){

                console.log(e);

            }

        })();

실행결과



-encodeURIComponent, decodeURIComponent(window 객체의 메서드)

AJAX요청을 보낼때 주소에 한글이들어가는경우 서버가 한글을 이해하지못하는경우발생

이때 encodeURIComponent, decodeURIComponent를사용하여 처리

node에서도 사용가능

파일경로: node/ch2/ encodedecodecomponent.html

(async()=>{

            try{

                const result = await axios.get(`https://www.zerocho.com/api/search/${encodeURIComponent('노드')}`);

                //백틱으로감싸고 해당한글부분만을$()로받아와 encodeURIComonent로 변환시켜준다

                //받는쪽에서느 decodeURIComponent메서드를 이용하여 변환된 문자열을넣어주면된다

                console.log(result);

                console.log(result.data);

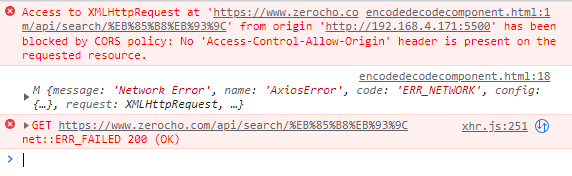
            }catch(e){

                console.log(e);

            }

        })();

실행결과



-데이터속성과 dataset

노드를 웹서버로 사용하는경우 데이터를 빈번하게 주고받게된다

이때 프런트엔드에 데이터를 내보낼때 보안을 중요시해야한다(보안과 관련이 있다면 내보내면안된다)

파일경로: node/ch2/dataform.html

<body>

    <ul>

        <li data-id="1" data-user-job="programer">Zero</li>

        <li data-id="2" data-user-job="designer">Nero</li>

        <li data-id="3" data-user-job="programer">Hero</li>

        <li data-id="4" data-user-job="ceo">Kero</li>

    </ul>

    <script>

        console.log(document.querySelector('li').dataset);

        console.log(document.querySelector('li').dataset.userJob);

        //dataset으로 접근시 앞의접두어 data-는 사라지고 -뒤에위치한 첫번째글자는 대문가된다

        //반대로 HTML택으에 dataset을 넣어 속성을 반영시킬수있다

    </script>

</body>

실행결과



chapter3 노드기능알아보기

자바스크립트는 스크립트언어이므로 미리 컴파일하지않아도 즉서에서 코드를 실행가능

-REPL(Read Eval Print Loop 읽기 해석 결과물반환 종료할때까지반복)



vs code에서는 Ctrl+`를 누르면 터미널

Ctrl+c를 두번누르거나 .exit를 입력하면 REPL을 종료

-CommonJS모듈

표준 JavaScript모듈은 아니지만 표준이 나오기 이전부터 쓰였고 노드생태계에서 가장 널리 쓰이는 모듈

module.exports로 모듈만들기

module.exports = 내보내고자하는 변수이름;

사용하는 쪽에서는 const 변수명 = require(‘해당경로 파일명’);구조분해할당을 이용하여 저장한다

이때 한가지만 주고받을때는 변수명을 받는 변수명을 바꿀수있지만 두가지이상되면 같은변수명으로 받아야한다

var.js

const odd = "cjs 홀수입니다";

const even = "cjs 짝수입니다";

module.exports ={

    odd,

    even,

};

func.js

const{odd,even} = require('./var');

//console.log(require('./var'));

function checkOddOrEven(num){

    if(num%2){//0이면 false여서 실행이안되고 0이아니면 true이기때문에 실행

        return odd;

    }

    return even;

}

//console.log(checkOddOrEven(1));

module.exports = checkOddOrEven;

index.js

const checknumber=require('./func');

console.log(checknumber(10));