2.17

- Dynamic Web Project 생성

- <script> </script>안에 써준다. (Header Body 둘 다)

- document.getElementById("divBox").innerHTML="김우경";

- function myFunc(){alert("함수호출입니다"); myFunc2();} / ----함수 내에서 함수 호출

- body태그에 now변수의 값을 출력
- 변수에 담아서 출력 : document.write(now);
- 바로 출력 : document.write("
Hello world
");

- window.onload
    - body태그를 모두 로딩한 후에 실행하는 함수

  ex)
  window.onload=function(){
  document.getElementById("box").innerHTML="Hello world";
  (box라는 id에 “ “안의 내용을 넣음)
  }

- var
    - 데이터타입인데 모든 형을 사용
      (문자열, 정수, 실수 등 심지어 배열도)
    - 초기화 되지 않았을 경우 undefined라는 값이 들어있다.

- 전역변수 - var가 있어도 없어도 상관x

  지역변수 - 함수 내의 변수
  - 함수내에서 var를 사용하면 새 지역변수 생성
    함수내에서 var없이 생성한다면 그건 지역이 아니라 전역변수임
  **f12를 통해 확인했을때 전역변수는 검색되지만 지역변수는 없다고 나옴

- 쌍따옴표 안에 들어갈땐 홑따옴표를 써준다 “ ‘ ’ ”

- window.onload = function(){
  alert(document.getElementById("divBox").innerHTML); //바로 실행
  var str = document.getElementById("divBox").innerHTML; //변수에 담아 실행
  str = str + "
추가된 내용입니다";
  document.getElementById("divBox").innerHTML = str;
  //수정된 str을 다시 넣어줌
  }

- alert : 경고창
  prompt : 입력받는 창 ex) var x = prompt("첫번째 숫자를 입력해주세요");

- 숫자 + 문자열 = 문자열
  - document.write(y+z);

- 디버깅
  F12 -> source -> 행번호 클릭 -> F5 -> F8(play)
  콘솔에서 직접 입력해서 확인가능

- <button id = "op" onclick="op('+')">더하기</button>
  - onclick : 클릭시 발생하는 이벤트
  - op는 함수이름, op()괄호 안에 들어가는게 함수의 파라미터로 넘길값

- value : 텍스트박스 안의 값 가져옴
  innerHTML : span, div, p 등의 내용

- 정수형변환
  parseInt() 또는 *1

- 비교연산자
  == : 안에 내용만 비교
  === : 내용 및 타입까지 비교
  != : 안에 내용만 비교 (같지 않을 경우)
  !== : 내용 및 타입까지 다르면 true

2.18

- 반복문

  continue; : 조건식이 되면 건너뜀
  break; : while문 자체를 빠져나감

- function

  function abc(){ }; : 함수 생성
  abc(); : 함수 실행

- 배열변수

  - var fruits = ["apple", "banana", "peach"]; : 배열선언(리터럴로 생성)
  - 자바스크립트는 한 배열에 문자열이나 숫자나 다 들어갈 수 있음
  - 출력 : document.write(fruits); : 배열명만 찍으면 원소들 다 찍힘
  - for(var i in food){ //for -in 문 (인덱스 in 배열)
    document.write(food[i]);

```
}
```

2.19

- 객체 (object)

  사용자 정의 객체

  - 방법1) 객체 상수로 부터 객체 생성 하는 방법

```
var myCar = {
        //속성(property) - 명사
        model : "bmw 520d",
        speed : 0,
        //메서드(method) - 동사
        accel : function(){this.speed += 10;}
};

console.log(myCar.speed);   (콘솔창에 띄움)
myCar.accel();              (메서드 호출)
console.log(myCar.speed);   (메서드 호출 후 값이 변화)
```

  - 방법2) 생성자를 이용해 객체 생성 하는 방법

```
function Car(model, speed, color){
        //따로 기본 선언 없이 바로 this.변수명으로 선언가능
        this.model = model;
        this.speed = speed;
        this.brake = function(){
                this.speed -= 10;
        }
        this.accel = function(){
                this.speed += 10;
        }
}

var myCar = new Car("bmw 520d", 0, "white");        //파라미터로 값 넘김
console.log(myCar.color);
```

- Array객체

- 배열 생성

```
var str = [ ];
var str2 = new Array();
```

2. 20

- Date 객체 생성
- new Date(); 객체생성 / 오늘날짜 가져옴
- new Date(1000); 1970년 1월 1일 기준으로 밀리세컨드 만큼 지난 시간이 세팅
- new Date(2020,10,10,시,분,초); //11월 10일 (month는 0부터 세팅)

- 시간표기법
- today.toLocaleDateString(); // 1980년 1월 3일 목요일
- today.toLocaleTimeString(); // 오전 1:28:35
- today.toLocaleString(); // 1980년 1월 3일 목요일 오전 1:28:35

- 요일구하기
- getDay() 요일 리턴
- 0(일요일)~6(토요일)까지 숫자값을 리턴

- 경과시간 구하기
- getTime() : 밀리초값을 구한다
- 1초 = 밀리초 / 1000

- 경과 일수 구하기
- 구한 경과시간을 일수로 변환
- parseInt(경과시간 /1000/60/60/24);

- x일 후의 날짜 출력
- 밀리세컨드를 구한다 (24 * 60 * 60 * 1000 * x일)
  현재시간 + 구한 밀리초
  위의 값을 new Date(a)로 날짜객체 생성
  toLocaleString 이용해 날짜형태로 출력

- Date객체의 메소드
- getDate() : 1~31일을 가져온다
- getMonth() : 0~11달을 가져온다
- getFullYear() : 년도를 가져온다 2020
- getHours() : 0~23시간을 가져온다
- getMinutes() : 0~59분을 가져온다
- getSeconds() : 0~59초를 가져온다
- getMilliseconds() : 0~999밀리초를 가져온다
- getTime() : 경과시간
- getDay() : 요일 숫자값

- 달력만들기 (Calendar.html 참조)

2. 21

- 스트링객체

- charAt(index) : index위치에있는 문자를 리턴한다
- indexOf() : 부분문자열의 위치를 검색한다 (문자가 없을시 -1리턴)
  indexOf(문자열 , 검색시작인덱스)
- concat(s1, s2) : s1문자열과 s2문자열을 연결한다 / +연산과 동일
- trim() : 앞뒤 공백을 제거한다
- ToLowerCase() : 소문자로 변경
- toUpperCase() : 대문자로 변경
- substring(from, to) : 두 위치 사이의 부분 문자열을 추출한다
  to를 생략하면 뒤쪽 모든 문자열을 출력한다.
  to 미만까지 출력한다.
- slice(start,end) : start위치부터 end위치까지 부분 문자열을 추출한다.
  substring()과의 차이점은
  slice()는 음수값을 파라미터로 적용하여
  문자열을 뒤에서 부터 출력할 수 있다.
  substring()은 음수 사용 불가능
- split() : 구분자를 기준으로 문자열을 분리한다.

2. 24

- math객체

  Math.PI : 파이상수 (3.14....)
  Math.abs(x) : 절대값
  Math.ceil(x) : 올림
  Math.floor(x) : 내림
  Math.round(x) : 반올림
  Math.max(1,2,3,4,5,6...) : 최대값
  Math.min(1,2,3,4,5,6...) : 최소값
  Math.pow(x,y) : x의 y승 (지수함수)
  Math.random() : 0부터 1까지의 소수 난수를 발생함

  원하는 범위의 랜덤값 만들기
  Math.floor(Math.random() * (최대값 - 최소값 + 1) + 최소값);
  Math.floor(Math.random() * (10 - 1 + 1) + 1); (1부터 10까지의 랜덤수)

- 주의!!
  함수내에서 함수를 다시 호출했을때 이전값이 계속 담겨있는 경우
  전역변수에 선언하고 초기화만 지역변수로 한다
  -> 왜냐면 지역내에 먼저 생성된 변수는 남아있고 변수를 하나 더 만드는 격이기때문!!

- try-catch
  ex)
  try{
  실행문
  }catch(err){
  msg = "오류발생 " + err.message;
  console.log(msg);
  }finally{
  console.log("여기는 무조건");
  }

2. 25

- 자바 스크립트에선 자료형을 안쓰면 자동으로 전역변수 지정

- BOM DOM

- setTimeout(function, millisec)
  - 정해진 시간 후에 function은 1번만 실행한다.

  setInterval(function, millisec)
  - 정해진 시간 간격으로 매번 function을 수행

  clearInterval(변수)
  - setInterval에 의해서 실행된 기능을 중지

- window 객체

  - 브라우저 모델 객체(BOM)에서 최상위 객체로서 웹 브라우저 윈도우를 나타낸다
  - open() / close()
  - setInterval() / clearInterval() / setTimeout()
  - alert() / confirm() / prompt()
  - focus()
  - 속성 : opener

  ex) var win = window.open("winsub.html","_blank","width=500,height=500");
  - self (기존창) 또는 _blank(새창)

- moveTo(x,y);
  moveTo(윈도우 창 기준으로 x,y좌표 만큼 절대적 위치이동)

- focus()
  - 마우스 클릭하여 입력상태로 만들었을때

  blur()
  - 입력상태 벗어났을때

2.26

- HTML요소의 선택
  document.getElementById(아이디)
  - 해당 아이디의 요소를 선택함

  document.getElementsByTagName(태그이름)
  - 해당 태그 이름의 요소를 모두 선택함

  document.getElementsByClassName(클래스이름)
  - 해당 클래스에 속한 요소를 모두 선택함

  document.getElementByName(name속성값)
  - 해당 name 속성을 가지는 요소를 모두 선택함

- HTML 핸들러 추가
  document.getElementById(아이디).onclick = function(){실행할 코드}
  - 마우스 클릭 이벤트와 연결된 이벤트 핸들러 코드를 추가함

- location 객체

- URL 정보를 얻어오는 객체
  href : 전체 url
  port : port정보
  host : hostname과 port
  hostname : hostname
  pathname : 경로

- location.protocol
  location.host
  location.hostname
  location.port
  location.href

- 인코딩
  decodeURI() : 자바스크립트에서 한글로 인코딩해줌

- <input type = "submit" value = "전송">
  - submit타입일 경우, onclick에 해당하는 함수 실행 후, action url로 이동

- 페이지 이동하기

  페이지 이동 <button onclick="location.href='result.html';"></button>
  페이지이동2 <button onclick="goPage('result.html');"></button>
  페이지 재이동 <button onclick="goReplace('result.html');"></button>
  (페이지 이동이 아니라 페이지 자체를 바꾼것으로 뒤로가기 불가)
  페이지 리로드 <button onclick="goReload();"></button>
  (F5 새로고침 기능과 동일)

- navigator 객체
  : 브라우저 공급자 및 버전 정보등을 포함한 브라우저에 대한 정보를 저장하는 객체
- navigator.appName : 현재 사용중인 브라우저 이름
  navigator.appVersion : 현재 사용중인 브라우저 버전 정보
  navigator.userAgent : userAgent프로퍼티로 알 수 있는 추가정보
  navigator.platform : 현재 브라우저가 실행되고 있는 운영체제
  navigator.language : 현재 브라우저의 기본 언어

- <script>는
  1) <body>태그에 바로 쓰거나
  2) <head>태그에 쓰고 window onload하거나

- firstChild
  appendChild() : 태그안에 추가로 붙임 (텍스트가 이미 존재하면 그 후로 붙임)
  lastChild()
  removeChild()
  parentNode
  createElement() : 요소 생성
  $("").createTextNode() : 요소안에 텍스트추가

- HTML요소의 생성
  document.createElement(html요소) : 지정된 HTML요소를 생성함 (p, img 등)
  document.write(텍스트); : HTML 출력 스트림을 통해 텍스트를 출력함

- 새로운 요소 생성 (이미지)

  1) 이미지 담을 요소 선택
     ex) strimg = domlist[i].firstChild.data;

  2) 이미지 객체 생성
     ex) imgEle = document.createElement("img");
         imgEle.src = "../img/" + strimg + ".jpg";

  3) 이미지 속성 정의
     ex) imgEle.width = 100;
         imgEle.height = 100;

  4) 요소의 자식으로 이미지 추가
     ex) domlist[i].appendChild(imgEle);

- <form name = "myform" action = "result.html" method = "post"></form>
  - get방식 : 파라미터가 붙음 / 아무것도 안쓰면 기본get방식
  - post방식 : 파라미터 안붙음

2.27

- form요소
  <form name = "ff"> </form>
  document.ff - form으로 접근 (name은 바로 접근 가능)

  데이터 교환 프로그램(0227 의 select_form파일)

- html요소
- return false;
  - 이전까지 실행하고 멈춤
- textarea
  rows="15" (줄 수)

- 배열에 아무것도 없을때 를 조건으로 설정한 경우
  - 아무것도 안담기면 object 형으로 바뀌어서
    == " "로 스트링 지정 또는 == 0으로 확인가능함

3.9

- radio는 name을 똑같이 해줘야 중복체크x

- 1) <body onload = "onLoadDoc();">
  2) window.onload = function(){document.body.style.backgroundColor = "yellow";};

  -> 1)과 2)는 동일하다. 그러나 2가 더 적절하다(수정시 body를 건들면 귀찮아짐)

- var doc = document.getElementsByName("eng")[0];
  -> elements라서 여러개니까 [0]으로 정해줌.

- toUpperCase() 대문자로 변경
  toLowerCase() 소문자로 변경

- <input type = "text" name ="eng" onchange = "sub();">
  -> 입력 필드를 벗어나면 onchange 이벤트가 실행된다.

- onmouseover : 마우스 오버 ex) onmouseover = "over(this);"
  onmouseout : 마우스 아웃
  onmousedown : 버튼 클릭시
  onmouseup : 버튼 클릭 뗐을때

- 함수(this)는 이 함수의 상태를 파라미터로 넘김

- 정규식

ex)
```
//이름 체크
var name = document.getElementById("name").value.trim();
if(name == ""){
	alert("이름을 입력해주세요");
	return false;
}else{
	var regName = /^[가-힣]{2,5}$/;
	if(!regName.test(name)){
		alert("한글, 2-5자로 입력해주세요");
		return false;
	}
}
```

- 드래그 & 드롭 (drag & drop)

draggable : 드래그 되는 요소를 true로 설정
dragstart : datatransfer 객체에 setData()호출하여 데이터를 설정한다.
	- 드래그 시작을 할때 해당하는 객체를 key, value 값으로 설정한다.

dragover : 드래그 도중 마우스가 다른 요소 위에 있을 때 발생한다.
	만약, 타겟 요소에서 dragover 이벤트가 발생하면 드롭을 허용한다
drop : 마우스 버튼을 놓았을때, 반드시 처리 할 내용
	dataTransfer객체에서 getData() 메서드를 이용하여 필요한 데이터를 꺼낸다.

- drop동시에 dragover

ex)

ondrop="dropEvent(event);" ondragover = "dragover(event);-----body, div에!
<img src = "가구1.jpg" draggable = "true" ondragstart = "dragstart(event);">
...

```
function dragstart(event){
	event.dataTransfer.setData("target_img", event.target.id); //(키값, 밸류값)
	//이미지를 드래그 했을 경우, 키값에 그 이미지값을 저장한다.
}
function dropEvent(event){
	imgid = event.dataTransfer.getData("target_img");
	//dragstart에서 set된 img값을 가져옴

	if(event.target.tagName == "IMG"){
	    event.target.parentNode.appendChild(document.getElementById(imgid));
	        //이미지면 위로 옮기고
```

```
        }else{
            event.target.appendChild(document.getElementById(imgid));
            //이미지 아니면 원래자리로
        }
    }
    function dragover(event){
        event.preventDefault(); //드래그를 그만한다고 닫아주는것
    }
```

3.10

* 파일 Api

- input type="file" 은 파일선택버튼

- File객체
- 파일의 기본적인 정보를 얻는다
- 파일명(name), 파일크기(size), 파일종류(type), 변경날짜(lastModifyDate)

- FileReader객체
- 파일의 내용을 읽을 수 있는 기능을 제공한다
- 자바스크립트에서 가져옴

- FileReader객체 메소드
- readAsText(file객체, 인코딩) : 파일을 읽어서 텍스트로 변환한다
  인코딩을 안쓰면 기본적으로 UTF-8
- readAsDataURL (file객체) : 파일을 읽어서 dataUrl형식의 문자열로 반환한다

- FileReader객체 속성
  result : 읽어온 파일의 내용이 저장되어 있다
  error : 오류가 발생했을 경우 오류정보가 저장되어 있다