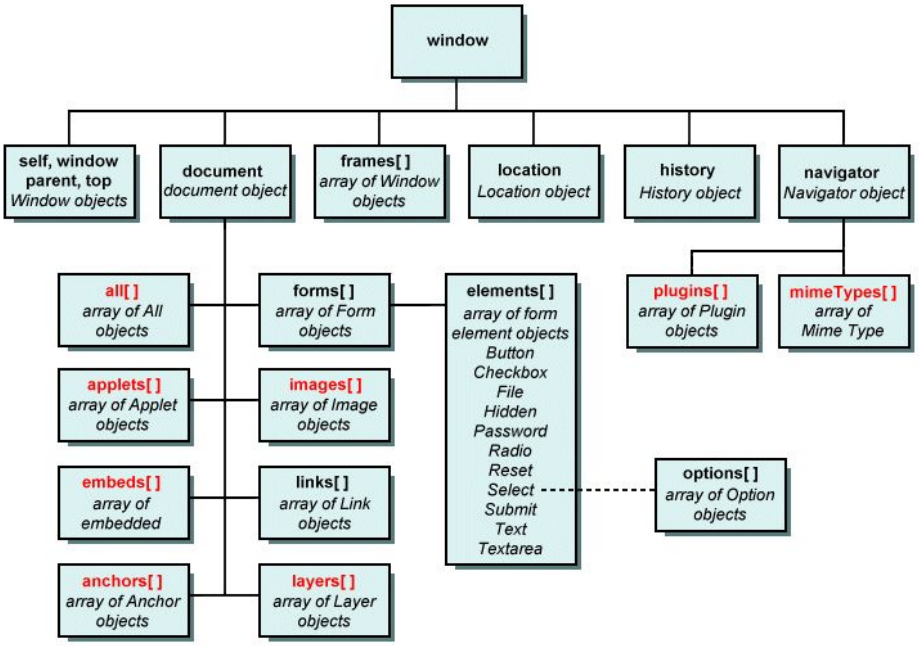
VanillaJS 1주차 내용 정리

1. 프로그래밍 언어
    - 정의 : data + logic 을 처리할때 사용하는 컴퓨터와 대화하기 위한 언어
    - 컴파일 언어 vs 인터프리터 언어
    - 언어의 유형
        - 절차적 언어(procedure language) : C
        - 객체 지향 언어(object oriented programming) : Java, C++, C#
        - 함수형 언어(functional programming) : Scala
        - 논리형 언어(일단은 크게 신경쓰지 않아도 됨)

2. 자바스크립트의 특징
    - 인터프리터 언어이다.
    - 동적 프로토타입 기반 객체 지향 언어이다.
    - 동적 타입언어이다.
    - 함수가 일급 객체다.(first class object) , 함수가 중심인 언어이다 정도로 이해하면 된다.
        - 변수 혹은 데이터를 구조에 담을 수 있다.
        - 파라미터로 전달할 수 있다.
        - 반환값으로 사용할 수 있다.
        - 이 특성을 활용하면 고차함수를 구현할 수 있어 함수형 프로그래밍이 가능
        - 고차함수(high order function) : 함수를 받아서 함수를 반환하는 것을 말한다. 다른 함수를 이용해서 완전히 새로운 함수를 조립하는 방법으로 프로그램을 만들 수 있다.
    - 함수가 클로저를 정의한다.
        - 클로저라는 게 있고, 함수라는게 있는데
        - 클로저(closure)를 표현할때 함수를 활용한다는 의미이다.
        - 이것은 나중에 뒤에서 클로저라는 설명할 때 이해하면 된다.

3. 자바스크립트 언어의 기술적 요소
    - ECMAScript(코어 언어)
    - 클라이언트 측의 고유한 기술 요소
        - ECMAScript가 규정한 코어 언어 + 웹브라우져의 API(DOM)
        - 웹 브라우져의 API
            1. Window 인터페이스 : 자바스크립트로 브라우저 또는 창을 조작하는 기능을 제공한다.
            2. DOM(Document Object Model) : 자바스크립트로 HTML 문서의 요소를 제어하는 기능을 제공한다.
            3. XMLHttpRequest : 서버와 비동기로 통신하는 기능을 제공한다.

DOM(Document Object Model) hierarchy
https://www.startertutorials.com/ajwt/document-object-model.html



- HTML5의 주요 API
  - Drag and Drop : HTML 요소 혹은 파일을 끌어서 다른 HTML 요소세 놓을 때 데이터를 전달하는 기능을 제공한다.
  - Blob : 이진 데이터를 다루는 기능을 제공한다.
  - File : 로컬 파일 시스템을 읽고 쓸 수 있는 기능을 제공한다.
  - Web Workers : 프로그램 여러개를 멀티스레드로 병렬 처리하는 기능을 제공한다.
  - Web Storage : 대용량이며 저장기간에 제한이 없는 데이터를 로컬에 저장하는 기능을 제공한다.
  - Indexed Database : 로컬에 키-값(key-value) 타입의 관계형 데이터베이스 기능을 제공한다.
  - WebSockets : 서버와의 양방향 통신 기능을 제공한다.
  - Geolocation : GPS 등의 위치 정보를 다루는 기능을 제공한다.
  - Canvas : 2차원, 3차원 그래픽스 기능을 제공한다.

- 서버 측 자바스크립트의 고유한 기술 요소
  - 웹서버 구현 언어 : Perl, PHP, Python, Ruby
  - Node.js : 구글이 개발한 자바스크립트 실행 환경
  - Rhino : 오픈 소스로 개발되어 현재는 모질라(Mozilla)가 관리하고 있는 자바스크립트 실행 환경
  - Aptana Jaxer : 압타니(Aptana) 사가 개발하고 현재는 오픈 소스로 개발되고 있는 자바스크립트 실행 환경

4. ECMAScript 6(ECMAScript 2015)

- 2015년 6월에 권고
- 2009년에 권고된 ECMAScript 5 이후로 가장 큰 변화
- ECMAScript 6 이후로 ECMAScript 2015,2016,2017,2018... 로 연호를 붙인다.
- ECMAScript 6에 추가된 주요기능
  - 템플릿 리터럴
  - Symbol
  - 블록 범위
  - Math, Number, String의 새로운 메소드
  - 화살표 함수
  - 함수 매개변수에 추가된 기능
  - 이터레이터/제너레이터
  - 객체 리터럴에 추가된 기능
  - Object에 추가된 메소드
  - 비구조화 할당
  - 전개 연산자
  - Map, Set, WeakMap, WeakSet
  - ArrayBuffer와 형식화 배열
  - Array에 추가된 메소드
  - 정규 표현식에 추가된 메소드
  - Promise
  - 클래스
  - 모듈
  - Proxy, Reflect
  - 꼬리 재귀 최적화
- es6 웹 브라우저 지원 현황 확인
  - https://kangax.github.io/compat-table/es6/
- ECMAScript 2016에 추가된 내용
  - 거듭제곱 연산자
  - Array.prototype.includes()
- ECMAScript 2017에 추가될 내용
  - SIMD(Single Instruction/Multiple Data)로 데이터 수준의 병렬 연산 지원
  - Async Function으로 비동기 처리 작성 지원
  - Decorators로 클래스에 기능 더하기
  - 객체의 잔여 프로퍼티를 새로운 객체에 할당하는 Rest Properties

5. 자바스크립트의 역사
   - 1995년 Netscape Communications의 브렌던 아이크(Brendan Eich)가 개발
   - Netscape Navigator 2.0에 구현
   - 1996년에 Microsoft 사의 Internet Explorer 3.0에 탑재
   - 초창기에는 Netscape Navigator와 Internet Explorer에서 모두 동작하는 코드를 만들기 어려웠다.
   - 1997년 ECMAScript 표준화 진행
   - 현재는 오래된 브라우저외에 대다수 브라우저의 호환성 문제는 해소

6. 실습 준비하기
   - 웹 브라우저와 Node.js 설치하기
     - 웹 브라우저 : 최신 Chrome Browser(버전 77.0.3865.120(공식 빌드) (64비트))

        - https://nodejs.org/ko/download/
        - 운영체제에 맞는 Node.js를 다운받아 설치한다.
    - 텍스트 편집기 준비하기
        - Visual Studio Code
        - 메모장
        - Sublime Text3
        - WebStorm
        - Adobe brackets(내장된 Node.js 실시간 미리보기 기능을 갖추고 있음)

7. 간단한 소스 작성해보기 – 팩토리얼 계산하고 표시하기
    - file명 : exam_01_factorial.js
        - function fact(n){
        - if(n<=1) return n;
        - return n*fact(n-1);
        - }
        - for(var i=1; i<10; i++){
        - console.log(i+"!="+fact(i));
        - }
    - [결과화면]
        - 1!=1
        - 2!=2
        - 3!=6
        - 4!=24
        - 5!=120
        - 6!=720
        - 7!=5040
        - 8!=40320
        - 9!=362880

8. 프로그램 실행
    - 웹 브라우져에서 콘솔 실행하는 단축키
        - 맥 : Command + Option + i
        - 윈도우 : Ctrl + Shift + i
    - 자바스크립트 코드를 HTML 문서에 삽입하여 웹 브라우져로 실행하기
        - <script src="exam_01_factorial.js"></script>
    - Node.js의 대화형 모드로 실행하기
        - 맥 : 터미널창 실행 or 윈도우 : cmd창 실행
        - Math.sqrt(2)
        - console.log("Hello, World");
    - Node.js로 파일을 읽어 들여 실행하기
        - Node.js를 설치한 후에 exam_01_factorial.js 파일을 작성한 폴더 위치에서 아래와 같이 실행하면 결과가 출력됨.
        - #node exam_01_factorial.js

9. 문자코드(시험제외)
    - 자바스크립트 프로그램은 유니코드 문자로 작성.
    - 유니코드 : 전 세계의 문자를 포함한 문제 체계

- ECMAScript 5 : 유니코드 버전 3 이상 지원
  - http://ecma-international.org/ecma-262/5.1/#sec-2
- ECMAScript 6 : 유니코드 버젼 5.1.0 이상
  - http://ecma-international.org/ecma-262/6.0/#sec-2

10. 대소문자 구별
    - javascript의 API(함수)를 호출할 때 대소문자는 반드시 구분되어야 한다.
    - 대소문자 구분을 하지 않으면 오류가 발생한다.
    - Console.log("Hello, World"); (오류발생)
    - console.log("Hello, World"); (정상실행)

11. 토큰
    - 프로그램을 구성하는 최소단위
    - return n*fact(n-1);
    - return | n | * | fact | (| n | - | 1 |) | ;

12. 공백문자(시험제외)
    - 토큰을 공백 등을 넣지 않고 나열만 해서는 판별할 수 없다.
      - return n, returnn
    - 언어마다 공백문자를 정의하는 기준이 다름(언어마다 확인해 봐야함)
    - ECMAScript 5
      - 일본어 반각 스페이스(₩u0020), 탭(₩u0009), 수직 탭(₩u000B), 폼 피드(₩u000C),줄 바꿈 없는 공백(₩u00A0), 바이트 순서 표시 제어문자(₩uFEFF), 기타 유니코드 카테고리 Zs에 포함된 모든 문자
    - 주의
      - 일본어 전각 스페이스(₩u3000)도 Zs에 포함되므로 공백문자로 간주하나 HTML/CSS 에서는 공백문자로 취급하지 않는다.

13. 공백 생략 가능한 경우
    - a = 1 + 2 + 3 => a=1+2+3
    - function fact(n) => function fact(n)
    - { x : 1 , y : 1 } => {x:1, y:1}
    - obj . x => obj.x
    - [1 , 2 , 3] => [1,2,3]
    - a [0] = 10 => a[0]=10;

14. 프로그래밍 가독성 높이기
    - function fact(n) {if(n<=1)return n;return n*fact(n-1)}
    - function fact(n){
    - if(n<=1) return n;
    - return n*fact(n-1);
    - }

15. 문장사용법
    - console.log(i+"! = "+fact(i));
    - var x; // 변수 선언
    - message = "Hi, "+name; // 표현식의 대입

- counter++; // 변수 값 증가
- prompt("이름을 입력하세요.") // 함수호출

16. 복합문(블록문)
    - 문장 여러개를 {}(브레이스)로 감싼 코드
    - 여기에서 문장이라함은 sum = sum + x; 이것을 한 문장이라 생각하면 된다.
    - ';'이 있는 것 까지를 한문장이라고 보면된다.
    - 물론 엔터를 쳐서 문장이 구분될때도 엔터치기전까지를 한문장으로 보면 된다.
    - {
    - sum = sum + x;
    - console.log("sum = "+x);
    - }

17. 빈문장
    - 문장이 하나도 없는 문장
    - ;(세미콜론)으로 작성한다.
        - for(;;){
        - ...
        - }

18. 세미콜론 자동추가
    - 한 문장 작성후 다음줄에 문장을 작성하면 자동으로 세미콜론이 있는것 처럼 인식한다.
        - console.log("Hello")
        - console.log("Hi")
        - console.log("Hello");
        - console.log("Hi");
    - 작성한 코드와 줄바꿈한 문장이 이어진다고 판단하면 세미콜론 추가를 하지 않는다.
    - 때문에 javascript에서는 한문장이 끝나면 반드시 ';'를 붙여준다.
        - c = a + b
        - (x+y).toString()
        -
        - 이렇게 작성을 하게 되면 아래와 같이 해석을 하게 된다.
        - 작성한 문장은 두 문장이지만, 아래와 같이 한문장을 해석하게 됨을 유의해야 한다.
        - c = a + b(x+y).toString();
    - 아래와 같은 경우에는 다르게 해석하기 때문에 주의해야 한다.
        - return
        - 1;
        -
        - return 1;

19. 주석문
    - 실행은 되지 않고, 소스코드에 삽입되는 설명문
    - /**/ : 여러줄 주석
    - // : 한줄 주석

20. 변수
    - 값을 담는 상자(primitive type or reference type)

- 변수 선언
  - var x = 5; // 하나만 선언
  - var x=1, y=2; // 여러개 선언

- 아래와 같은 코드를 실행해 보자.
  - 1)
  - console.log(x);
  - var x;
  -
  - [결과]
  - undefined
  -
  - 2)
  - console.log(x) // undefined
  - var x = 5;
  - console.log(x) // 5
  -
  - [결과]
  - undefined
  - 5

- 변수명 규칙
  - 기본적으로 영어 단어를 사용한다. ex) var 나이=10; console.log(나이); // 10 출력된다.
  - 카멜 표기법이나 밑줄 표기법을 사용하여 변수의 의미를 파악할 수 있도록 이름을 붙인다.
  - 변수,함수 : 카멜(camel) ex) var coffeeName; function makeCoffee(){}
  - 변수 : 밑줄표기법(스네이크 표기법) ex) var new_name
  - 클래스 : 파스칼(pascal) ex) class Coffee(){}
  - 생성자 이름을 붙일때는 파스칼 표기법
  - 논리값을 표현하는 변수에는 이름 앞에 is를 붙인다.
  - 루프(반복문) 카운터 변수 이름으로는 i,j,k를 붙인다.

- 데이터 타입을 따로 정하지는 않는다.
- 여러가지 데이터 형태가 존재하는데, 타입을 따로 정하지는 않는다.
- “undefined”가 나오는 경우
  - 값을 아직 할당하지 않은 변수의 값
  - 없는 객체의 프로퍼티를 읽으려고 시도했을 때의 값
  - 없는 배열의 요소를 읽으려고 시도했을 때의 값
  - 아무것도 반환하지 않는 함수가 반환하는 값
  - 함수를 호출했을 때 전달받지 못한 인수의 값
- undefined는 존재하는데 값이 없을경우
- null 아무것도 없음을 할당하는 표현방식