UI 화면구현

<JS에서의 함수 선언과 호출의 의미>
선언(Declare)=정의(Define)
호출(Call)

만들고 사용하고 구현을 한다.

함수를 사용할 때 function()로 표현 하지만 ()만 사용하는 경우도 있다.

※ 참고
람다식(λ식) 으로 축약형으로 쓰기도 한다. - 자바 프로그래밍에서 배우는 것
람다식(lambda expression)은 익명 함수(anonymous function)를 만들기 위한 식으로, 함수를 간결하고 쉽게 표현할 수 있도록 해줍니다. 함수형 프로그래밍에서 사용한다.

람다식은 일반적으로 () => {} 형태로 표현됩니다. 괄호 안에는 매개변수가 들어갈 수 있고, 화살표 뒤에는 함수의 내용이 들어갑니다.

람다식의 예
```
const add = (a, b) => {
  return a + b;
};

// 람다식으로 표현
const add = (a, b) => a + b;
```

람다식은 간결한 코드를 작성할 수 있으며, 특히 자바스크립트에서는 콜백 함수(호출 함수)를 대체하는데 많이 사용됩니다.

매개변수에 함수를 입력

기본 자료형(String, Number)
객체 자료형(Object)
- Java와 JavaScript에서의 기본 자료형과 객체 자료형은 다음과 같습니다.(참고 사항)

Java:
기본 자료형: boolean, byte, short, int, long, float, double, char
객체 자료형: 모든 클래스와 인터페이스

JavaScript:
기본 자료형: number, string, boolean, null, undefined
객체 자료형: Object, Array, Function, Date, RegExp 등

Java에서는 모든 것이 객체이며, 기본 자료형은 객체 형태로 제공됩니다. 객체 자료형은 클래스와 인터페이

스를 포함하는 모든 것을 의미합니다. 기본 자료형과 객체 자료형 모두 변수에 할당될 수 있습니다.

JavaScript에서는 기본 자료형과 객체 자료형을 구분합니다. 기본 자료형은 원시 자료형이라고도 불리며, 값이 복사됩니다. 객체 자료형은 참조 자료형이라고도 불리며, 값이 참조됩니다. 예를 들어, number 타입의 변수에 값을 할당하면 값이 복사되지만, object 타입의 변수에 값을 할당하면 값이 참조됩니다.

배열 값 다루는 것을 하는 것 - 빅데이터, JSP등에서 다룸

<Join, Concat, Reverse, Sort, Slice, Splice, Pop&Push, Shift&Unshift, ForEach, Map, Filter, IndexOf&LastindexOf를 써서 Method 활용하기>

<Method 활용>
Object.method = array.method (JS에서는 모두 같음)

typeof를 사용해서 자료형의 형태를 확인하거나 코드를 통해서 사전에 자료형에 대해 알고 있어야 된다.

<ppt 리버스 부분>
reverse에 또 rdata.reverse를 해도 data 변수 값이 담기는 것이 같아서 결국에는 data 출력값이 같아짐

참고 - <딥카피와 샬로 카피>
객체의 복사를 이야기할 때, deep copy와 shallow copy 두 가지 개념이 있습니다.

Shallow copy는 객체를 복사할 때, 원본 객체와 같은 주소를 참조하는 복사 방식입니다. 즉, 복사한 객체가 원본 객체의 값을 수정하면 원본 객체도 같이 수정됩니다.

Deep copy는 객체를 복사할 때, 원본 객체와 다른 주소를 참조하는 복사 방식입니다. 즉, 복사한 객체가 원본 객체의 값을 수정해도 원본 객체는 영향을 받지 않습니다.

sort(함수) 함수
매개 변수이기도 하지만 sort입장에서는 callback 함수라고 말한다.

참고 - <Callback Function(콜백 함수)>
콜백 함수(callback function)란, 다른 함수에 인수로 전달되어, 그 함수 내부에서 실행되는 함수를 말합니다. 즉, 콜백 함수는 다른 함수의 인자로 전달되어 그 함수가 실행되는 도중에 호출되어 실행되는 함수입니다.

콜백 함수는 비동기적으로 실행될 때 많이 사용됩니다. 예를 들어, 웹 페이지에서 파일을 업로드하는데, 파일이 업로드되는 동안 다른 작업을 수행할 수 있도록 파일 업로드 함수에 콜백 함수를 전달하여, 파일 업로드가 완료되면 콜백 함수가 호출되어 다음 작업을 수행할 수 있도록 합니다.

콜백 함수는 함수 자체를 전달하는 것이 아니라, 함수를 가리키는 포인터(참조값)를 전달합니다. 따라서, 함수를 호출하는 측에서 콜백 함수를 호출할 때 인수를 넘겨주어야 합니다.

<참고 - 데이터 스트럭처(자료 구조)>

// 컴공 전공에서 배우는 것이라 어려운 편 //
데이터 구조(Data Structure)는 컴퓨터에서 데이터를 효율적으로 이용할 수 있도록 구성하는 방법을 말합니다. 데이터를 효율적으로 관리하기 위해서는 데이터의 종류에 따라 적절한 자료구조를 선택하여 사용해야 합니다.

자료구조에는 다양한 종류가 있습니다. 일부 자료구조는 아래와 같습니다.

배열(Array) : 일련의 동일한 자료형의 원소들로 이루어진 순서 집합
연결 리스트(Linked List) : 데이터를 연속적인 메모리 공간에 저장하지 않고, 데이터와 다음 데이터의 주소를 연결하여 구성한 자료구조
스택(Stack) : 후입선출(Last-In-First-Out)의 구조를 가진 자료구조
큐(Queue) : 선입선출(First-In-First-Out)의 구조를 가진 자료구조
트리(Tree) : 하나의 루트 노드(상위 자료)로부터 시작하여 가지(branch)로 구성된 노드들이 서로 연결된 구조
그래프(Graph) : 노드(자료)와 노드(자료)를 연결하는 간선(edge)으로 구성된 자료구조

데이터 구조(Data Structure)는 컴퓨터 프로그래밍에서 매우 중요한 역할을 합니다. 적절한 데이터 구조를 선택하면, 데이터 처리에 소요되는 시간과 메모리 사용량을 최적화할 수 있기 때문입니다. 또한, 많은 알고리즘에서도 데이터 구조를 활용하여 효율적인 처리를 구현할 수 있습니다.

// <Data Structure에 관한 자세한거는 검색 참조> //

<Shift&Unshift>
배열의 자릿수(숫자 같은 경우)나 위치를 변경할 때 사용하는 Method
Java에서 float(부동(浮動) 소수점)을 많이 사용함. float은 정수와 마찬가지로 프로그래밍 언어에서 기본적으로 제공하는 데이터 타입 중 하나이다.
자리를 밀어낼 때, shift는 왼쪽으로 밀어내기 unshift는 오른쪽으로 밀어낸다.

<forEach, Map>
배열 반복을 할 때 사용한다.
둘의 차이점은 배열 그 자체로 연산하는 것과 return value를 줘서 값을 담는 형태의 차이이다.
<Key Value - 키 값>
반복되지 않고 어렵지 않게 사용하여야 한다. - map의 구조

<index 사용>
Array에서 Value의 위치를 알아 낼 때 사용한다.
<JSON(JavaScript Object Notation), Object : {} 표시로 된 array를 칭함. - 연관배열>

<다음 장(11장) 내용>

<JS 객체(JS Object)>
앞에서는 Array에 대해서 많이 다루었음.
1) 만들기 2) 사용하기

Object = Array도 Object이지만 진짜가 Object
변수(Variable) - 배열(Array) - 객체(Object) - 객체 배열(Object Array)
Variable의 Group을 나타낼 수 있는 것이 Object
객체는 변수만 가지는 것이 아니라 함수도 가질 수 있다.

함수는 기능을 하여야 한다.
객체는 단순하게 멤버변수라는 속성값과 메소드를 가진다.

<절차 지향과 객체 지향>
절차 지향 프로그래밍(Procedural Programming)
프로그램을 순차적인 단계로 나누고 각 단계를 프로시저(서브루틴, 함수 등)로 구성하여 문제를 해결하는 방식입니다. 데이터와 프로시저를 분리하여 프로그래밍을 하는 방식으로, C, Pascal, Fortran 등에서 주로 사용됩니다
vs
객체 지향 프로그래밍(Object Oriented Programming, OOP)
객체를 중심으로 데이터와 기능(메서드)을 묶어서 처리하는 방식입니다.
객체 지향 프로그래밍에서는 현실 세계의 객체를 모델링하여 프로그램을 작성합니다. 객체는 속성(멤버 변수)과 동작(메서드)으로 구성되어 있으며, 객체 간 상호작용을 통해 문제를 해결합니다.
 Java, C++, Python 등에서 주로 사용되고 있습니다.

JS는 Java기반의 스크립트 언어이므로 객체지향 언어이다.

<객체 모델링(Object Modeling)>
속성값=상태값
클래스 - 값은 들어가있지 않고 틀만 있음.

객체는 왜 만드는가? 변수는 1개 배열은 여러개이다.
일반적으로 자동차를 예로 들면 하나의 제품이지만 속성이 여러개(색상 기능 등 여러 가지로 볼수 있다.)이다.
본래는 객체를 Class로 만들어야 한다.

JS에서는 객체를 중심으로 만든다.
<객체 만들기>
1) 변수(Variable)
2) Class
3) 생성자 함수(Constructor Function)

배열 리터럴(Array Literal)이 변수를 활용한 객체를 만드는 대표적인 방법이다.
{ , , }형태로 객체를 표시한다.
만들 때 무명 익명으로 이름이 없는 것으로 만들 수 있지만 이름을 붙여주어야 한다.
이름을 붙일 때 이름을 붙인다. var car1={ , , }형태로 하면 car1이 객체가 되면서 멤버의 변수, 속성을 들고 있게 된다.
클래스 안에 { 속성값 }으로 정의를 하면 Class로 만들 수 있다.
생성자 함수

var Car3=new Object()형태로 지정을 하고 객체.메소드=속성으로 생성자를 만들어서 나타낸다.

car3.name=c1;

car3.color='blue';

car3.wheel='4';

등을 활용하여 생성자를 만든다.