Sec01

자바스크립트 사용법 익히기

작성자: 강태경



# 조건문/반복문의 다양한 형태

자바스크립트는 조건문(if문) / 반복문(for문) 의 기본형태 외에 유사한 기능을 제공하는 다른 사용법이 있음. 조건문에는 3항연산자(리액트에서 사용)와 switch 문이 있고, 반복문에서는 while문이 있음

* 조건문   
  - 3항 연산자 (\*\*\*\*\*\*) : if문처럼 간단한 조건문에 사용, 리액트에서 자주 사용  
   사용법 : 조건식이 참이면 실행문을 변수에 저장하고, 아니면 실행문2가 변수에 저장됨  
   let 변수 = (조건식) ? “실행문” : “실행문2”;  
   예 :   
   let num = 2; // 초기값 지정  
   // 3항 연산자 사용  
   let result = ( num == 2 ) ? “num이 2임” : “num 이 2가 아님”;  
   // 결과 출력  
   alert(result);   
    
  - switch 문 :   
   사용법 : 변수가 값이면 실행문 실행 , 값2 이면 실행문2 실행, 모두 아니면 실행문n 이 실행됨  
   switch(변수) {  
   case 값:   
   실행문;  
   break;  
   case 값2 :  
   실행문2;  
   break;  
   …  
   default :  
   실행문n;  
   }
* 반복문 : while 문  
  - while 문 : 주로 무한히 반복되는 반복문에서 사용됨( 형태가 간단 )  
  사용법 : 조건식이 참일 동안만 실행문이 반복 실행됨, 거짓이면 반복을 중단하고 빠져나옴.  
   초기값;  
   while(조건식) {  
   실행문;  
   증감식;  
   }  
     
   예)   
   let i = 0; // 초기값  
   while(i<3) {  
   alert(i); // 실행문  
   i++; // 증감식  
   }
* 객체 : 세상의 모든 사물, 사람 등은 객체로 표현할 수 있음.   
   예) 고양이, 강태경, 강아지, 노트북, 책상, 의자 등  
  사용법 : css 사용법과 유사함  
  // 객체 정의(선언)  
  let 객체명 = {  
   속성: 값,  
   속성2 : 값2,  
   …  
   속성n : 값n  
  }  
    
  // 객체 사용(호출)  
  alert(객체명.속성명)  
    
  예) 고양이 객체 정의  
   속성 - 이름 : 나비, 색깔 : 흰색, 나이: 1살, 소유자: true  
    
  // 객체 정의  
   let cat = {  
   name : “나비”,  
   color : white,  
   age : 1,  
   isOwner: true  
   }  
    
  // 객체 사용(호출)  
  alert(cat.name); // 나비 출력  
  alert(cat.color); // white 출력
* 자료형 : 문자열(String), 숫자(Number), 참/거짓(true), 객체(배열, 일반객체), null/undefined 등이 있음( 참고 : 잘 쓰이지 않는 Symbol 자료형이 있음)
* 문자열 : 따옴표로 묶어서 표기  
  예) let value = “abcde”;
* 숫자 : 따옴표를 묶지 않고 표기  
  예) let value = 1;
* 참/거짓 : true, false  
  예) let value = true;
* 배열객체 : 여러가지 값을 저장하는 곳(장소), 변수명은 같고 변수명에 번호(인덱스번호)를 붙여 사용함, 번호는 처음 0 부터 순차적으로 n까지 사용됨  
  예) let arr = [“a”, “b”, “c”];  
   alert(arr[0]); // a 출력됨  
   alert(arr[1]); // b 출력됨
* null : 아무것도 포함되지 않는 값을 의미, 주로 초기화시 과거에 사용되었으나 현대적인 코딩에서는 null 로 초기화하는 것을 추천하지 않음  
  예) let value = null; // value 값 null 로 초기화(과거 코딩)  
   let value2 = “”; // 초기화시 “” 또는 0 으로 초기화하는 것을 추천함
* undefined : 정의되지 않는 자료형, 주로 어떤 객체에 그 속성이 없는데 속성을 접근하면 에러가 발생하는데 그때 자주 볼 수 있음  
   예)   
   let cat = {  
   name: “나비”,  
   age : 1  
   }  
   alert(cat.code); // undefined 에러 발생, cat 객체에 속성 code 가 없음
* 함수 : 중학교때 배운 수학시간 함수와 유사하며 미리 정해 놓은 코드를 재사용할 수 있게 만들어 주는 기능(\*\*\*\*\*)  
  장점) 코딩 생산성 향상(코딩 라인이 짧아짐)  
    
  - 매개변수 : 함수내에서만 사용되는 변수로 주로 함수 밖에서 값을 넘기면 매개변수에 임시 저장해서 함수 안에서 코딩시 사용됨  
  - return : 함수 안에서 코딩된 결과를 함수밖으로 내보내기 할 때 사용되는 예약어  
    
  사용법 :   
  1) 1st 사용 형태( 매개변수 없음 , return 예약어 안씀)  
  // 함수 정의(선언)  
  function 함수명() {  
   실행문;  
  }   
    
  // 함수 사용(실행, 호출)  
  함수명();   
    
  예)   
  // 함수 정의  
  function myAlert() {  
   alert(“안녕하세요”);  
   alert(“안녕하세요 2”);  
  }  
    
  // 함수 사용(실행, 호출)  
  myAlert(); // 안녕하세요 안녕하세요2 가 출력됨  
    
  2) 2nd 사용 형태(매개변수 있음, return 예약어 안씀)  
  : 1st 형태보다 좀 더 효율적인 함수 형태  
  // 함수 정의(선언)  
  function 함수명(매개변수) {  
   실행문(매개변수);  
  }  
  // 함수 사용(실행, 호출)  
  함수명(매개변수);   
    
  예)  
  // 함수 정의  
  function myAlert(params) {  
   alert(params);  
   alert(params + 2);  
  }  
    
  // 함수 사용(실행, 호출)  
  // 1st 형태보다 매개변수 값만 넘겨서 다양한 출력 상황에서 좀 더 유연한 코딩을 할 수 있음( 1st 함수 형태는 다시 함수를 작성해야 hello hello2를 출력할 수 있음)  
  myAlert(“안녕하세요”); // 안녕하세요 안녕하세요2 가 출력됨  
  myAlert(“hello”); // hello hello2 가 출력됨  
    
  3) 3rd 사용 형태(매개변수 있음, return 예약어 있음)  
  : return 변수 -> 변수의 값을 함수밖으로 내보내기 할 수 있음  
  function 함수명(매개변수) {  
   return 실행문(매개변수);  
  }  
  let 결과변수 = 함수명(매개변수);  
    
  예)  
  // 함수 정의  
  function myAlert(params) {  
   return params;  
  }  
    
  // 함수 사용(실행, 호출)  
  // 함수의 실행 결과값을 변수에 저장해 둘 수 있음  
  let result = myAlert(“안녕하세요”);   
  alert(result); // 안녕하세요 출력  
  alert(result + 2); // 안녕하세요 2 출력