JavaScript의 연산자 정리

기본식

* this : 실행 문맥의 특별한 속성을 가리킴
* function : 함수를 정의
* class : 클래스를 정의
* function\* : 생성기 함수 식을 정의
* yield : 생성기 함수를 일시정지 및 재개
* yield\* : 다른 생성기 함수 또는 순회가능 객체로 위임
* async function : 비동기 함수 표현식을 정의
* await : 비동기 함수를 일시 중지했다가 다시 시작하고 promise의 resolution/rejection을 기다림
* [ ] : 배열 초기자 및 리터럴 구문
* .{ }. : 객체 초기자 및 리터럴 구문
* /ab+c/i : 정규식 리터럴 구문
* .( \_ ). : 그룹 연산자

좌변식 : 할당 대상

* property accessors : 속성 접근자는 객체의 속성 또는 메서드에 대한 접근 방법을 제공 (object.property, object[“property”])
* new : 생성자의 인스턴스를 만듦
* new.target : 생성자 문맥에서 new.target은 new에 의해 호출된 생성자를 가리킴
* super : 부모 생성자를 호출
* …obj : 전개 연산자는 함수 호출 시 매개변수 여럿이나, 배열 리터럴에서 다수의 요소를 필요로 하는 곳에서 표현식을 확장

증가 및 감소

* A++ : 접미 증가 연산자
* A-- : 접미 감소 연산자
* ++A : 접두 증가 연산자
* --A : 접두 감소 연산자

단항 연산자 : 피연산자가 하나뿐인 연산자

* delete : 객체에서 속성을 지움
* void : 식의 반환값을 버림
* typeof : 객체의 형을 판별
* + : 단항 더하기 연산자 – 피연산자를 숫자로 변환
* - : 단항 부정 연산자 – 피연산자를 숫자로 변환한 뒤 부호를 바꿈
* ~ : 비트 NOT 연산자
* ! : 논리 NOT 연산자

산술 연산자 : 피연산자로 숫자값(리터럴이나 변수)을 취하고 숫자값 하나를 반환

* + : 더하기 연산자
* - : 빼기 연산자
* / : 나누기 연산자
* \* : 곱하기 연산자
* % : 나머지 연산자
* \*\* : 거듭제곱 연산자

관계 연산자 : 비교 연산자는 피연산자를 비교하고 비교가 참인지 여뷰를 나타내는 Boolean 값을 반환

* in : 객체에 주어진 속성이 있는지를 결정
* instanceof : 객체가 다른 객체의 인스턴스인지 판별
* < : 작음 연산자
* > : 큼 연산자
* <= : 작거나 같음 연산자
* >= : 크거나 같음 연산자

같음 연산자 : 같음 연산자의 평가 결과는 항상 Boolean 형으로 비교가 참인지 나타냄

* == : 동등 연산자
* != : 부동 연산자
* === : 일치 연산자
* !== : 불일치 연산자

비트 시프트 연산자 : 피연산자의 모든 비트를 시프트하는 연산

* << : 비트 좌로 시프트 연산자
* >> : 비트 우로 시프트 연산자
* >>> : 비트 부호 없는 우로 시프트 연산자

이진 비트 연산자 : 비트 연산자는 피연산자를 32비트 집합(0과 1)으로 다루고 표준 JavaScript 숫자값을 반환한다

* & : 비트 AND
* | : 비트 OR
* ^ : 비트 XOR

이진 논리 연산자 : 보통 사용될 때 불리언 값으로 사용되고 불리언 값을 반환

* && : 논리 AND
* || : 논리 OR
* ?? : Nullish(null 또는 undefined) 통합 연산자

조건부(삼항) 연산자

* (condition ? ifTrue : ifFalse) : 조건부 연산자는 조건의 논리값에 따라 두 값 중 하나를 반환

선택적 연결 연산자

* ?. : 선택적 연결 연산자는 참조가 nullish(null 또는 undefined)인 경우 오류를 발생시키는 대신 undefined를 반환

할당 연산자 : 값을 그 우변 피연산자의 값에 따라 좌변 피연산자에 할당

* = : 할당 연산자
* \*= : 곱셈 할당
* \*\*= : 거듭제곱 할당
* /= : 나눗셈 할당
* %= : 나머지 할당
* += : 덧셈 할당
* -= : 뺄셈 할당
* <<= : 좌로 이동 할당
* >>= : 우로 이동 할당
* >>>= : 부호 없는 우로 이동 할당
* &= : 비트 AND 할당
* ^= : 비트 XOR 할당
* |= : 비트 OR 할당
* &&= : 논리적 AND 할당
* ||= : 논리적 OR 할당
* ??= : 논리적 nullish 할당
* [a, b] = [1, 2] {a, b} = {a:1, b:2} : 구조 분해 할당은 배열 또는 객체의 속성을 배열 또는 객체 리터럴과 비슷해 보이는 구문을 사용하여 변수에 할당할 수 있게 함

쉼표 연산자

* , : 쉼표 연산자는 여러 식을 단문으로 평가되게 하고 마지막 식의 결과를 반환함