# JavaScript 클래스 기초 Episode 1 : 클래스 정의 방법(3가지 비교)

## 특징

|  |  |
| --- | --- |
| 방식 | 특징 |
| 프로토타입 방식 | 일반적인 클래스 제작방법  인스턴스보다 공통된 메서드를 공유해서 사용  jQuery도 이 방식으로 만들어져 있음 |
| 함수 방식 | 간단한 클래스 제작 시 사용  인스턴스마다 Method가 독립적으로 만들어지는 단점이 있음(이는 중복으로 이어진다) |
| 리터럴 방식 | 클래스 만드는 용도는 아니며 여러개의 매개변수를 그룹으로 묶어 매개변수로 보낼 때 사용  정의와 함께 인스턴스가 만들어지는 장점이 있음. 그러나! 인스턴스는 오직 하나만 나들 수 있음. |

## 정의 방법 비교

|  |  |  |
| --- | --- | --- |
| 프로토타입 방식 | 함수 방식 | 리터럴 방식 |
| function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  클래스이름.prototype.메서드1=function(){  }  클래스이름.prototype.메서드2=function(){  }  } | function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  this.메서드1=function(){  }  this.메서드2=function(){  }  } | var 인스턴스 = {  속성1:초기값,  속성2:초기값,  메서드1:function(){  }  메서드2:function(){  }  } |

## 인스턴스 생성 방법

|  |  |  |
| --- | --- | --- |
| 프로토타입 방식 | 함수 방식 | 리터럴 방식 |
| var 인스턴스 = new 클래스 이름(); | | var 인스턴스 = {  } |
| 프로토타입, 함수 방식 모두 new 명령어로 인스턴스를 생성한다.  new를 빼면 일반적인 함수가 호출되어 오류가 발생한다. | | 클래스 선언과 동시에 인스턴스가 자동으로 생성된다. |

## 객체 외부에서 속성과 메서드 접근 방법

|  |  |  |
| --- | --- | --- |
| 프로토타입 방식 | 함수 방식 | 리터럴 방식 |
| function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  클래스이름.prototype.메서드1=function(){  }  클래스이름.prototype.메서드2=function(){  }  } | function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  this.메서드1=function(){  }  this.메서드2=function(){  }  } | var 인스턴스 = {  속성1:초기값,  속성2:초기값,  메서드1:function(){  }  메서드2:function(){  }  } |
| var 인스턴스 = new 클래스이름();  인스턴스.속성1;  인스턴스.메서드1(); | | 인스턴스.속성1;  인스턴스.메서드1(); |

프로토타입, 함수 방식은 위의 밑줄 친 부분으로 인스턴스를 생성하고나서 속성, 메서드를 접근합니다.

## 객체 내부에 속성, 메서드 접근

|  |  |  |
| --- | --- | --- |
| 프로토타입 방식 | 함수 방식 | 리터럴 방식 |
| function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  클래스이름.prototype.메서드1=function(){  alert(this.속성1);  this.메서드2();  }  클래스이름.prototype.메서드2=function(){  }  } | function 클래스이름(){  this.속성1 = 초기값;  this.속성2 = 초기값;  this.메서드1=function(){  **alert(this.속성1);**  **this.메서드2();**  }  this.메서드2=function(){  }  } | var 인스턴스 = {  속성1:초기값,  속성2:초기값,  메서드1:function(){  **alert(this.속성1);**  **this.메서드2();**  }  메서드2:function(){  }  } |

접근 연산자를 기준으로 좌측에는 자기 자신을 나타내는 this 속성값을, 우측에는 속성과 메서드 이름을 작성해서 접근합니다.