

Java CobinCabin iRaCha 생성자

생성자(constructor)

정의

- 생성자(constructor)
 - 객체가 생성될 때에 필드에게 초기값을 제공하고 필요한 초기화 절차를 실행하는 메소드
 - 클래스 내에서 선언
 - 리턴 값이 없지만 void 키워드를 사용하지 않고 생략한다
 - 하나의 클래스에 여러 개의 생성자가 있을 수 있다
 - 인스턴스 초기화 메서드
- 생성자의 조건
 - 생성자의 이름은 클래스의 이름과 같아야 한다
 - 생성자는 리턴 값이 없다.

생성자(creator)

정의

- 생성자 정의

```
클래스 이름 (타입 변수명, 타입 변수명,...){
    //초기화 코드
}
```

```
class Test{
    Test(){
    }
```

```
    Test(String str, int num){
        //수행 코드
    }
}
```

```
Test tst = new Test();
```

1. 연산자 **new**에 의해서 메모리에 **Test**클래스의 인스턴스가 생성된다.
2. 생성자 **Test()**가 호출되어 수행된다
3. **Test**인스턴스의 주소가 반환되어 참조 변수 **tst**에 저장된다.

생성자(creator)

기본 생성자

- 모든 클래스는 반드시 하나 이상의 생성자가 정의되어 있어야 한다.
- 기본 생성자는 컴파일러에 의해 자동 생성
 - 소스파일의 클래스에 생성자가 하나도 정의되지 않은 경우 자동으로 추가한다
 - 클래스의 접근 제어자가 **public**인 경우에는 기본 생성자도 '**public** 클래스이름() {}'이 추가된다

생성자(creator)

매개변수가 있는 생성자

- 생성자도 매개변수를 사용하여 호출 시 값을 넘겨 받아서 인스턴스의 초기화 작업에 사용할 수 있다

```
public class Test00 {  
    public static void main(String[] args) {  
        Data1 d1 = new Data1();  
        Data2 d2 = new Data2(100);  
    }  
}
```

CobinCabin
iRaCha

생성자(creator)

this 포인터

- 메소드나 생성자에서 **this**는 현재 객체를 나타낸다.
 - 인스턴스 자신을 가리키는 참조 변수이다.
 - 인스턴스 멤버에 대해서만 사용 할 수 있다.
 - 모든 인스턴스에 지역 변수로 숨겨진 채로 존재한다
- **this()**
 - 같은 클래스의 다른 생성자를 호출 할 때 사용한다.
 - 생성자 간의 호출에서 생성자의 이름 대신 **this**를 사용한다
- 생성자 호출 조건
 - 생성자의 이름으로 클래스 대신 **this**를 사용한다
 - 한 생성자에서 다른 생성자를 호출할 때는 반드시 첫 줄에서만 호출이 가능하다.

생성자(*constructor*)

생성자 오버로딩

- 생성자 오버로딩
 - 메소드처럼 생성자도 오버로딩 될 수 있다.

CobinCabin
iRaCha

변수 초기화

정의

- 멤버 변수와 배열의 초기화는 선택적
 - 초기화를 하지 않아도 자동으로 변수의 자료형에 맞는 기본값으로 초기화가 이루어진다.
- 지역변수는 사용하기 전에 반드시 초기화를 해야 한다

```
class InitTest{  
    int x;  
    int y = x;  
  
    void method1(){  
        int i;  
        int j = i; //컴파일 에러  
    }  
}
```


변수 초기화

정의

- 멤버 변수는 초기화를 하지 않아도 자동으로 변수의 자료형에 맞는 기본값으로 초기화가 이루어진다.

자료형	기본값
boolean	false
char	'\u0000'
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d 또는 0.0
참조형 변수	null

변수 초기화

초기화 방법

- 초기화 방법
 - 명시적 초기화
 - 초기화 블록
 - 생성자
- 인스턴스 초기화 블록
 - 인스턴스 변수를 초기화 하는데 사용
- 클래스 초기화 블록
 - 클래스 변수를 초기화 하는데 사용

변수 초기화

초기화 블록

- 인스턴스 초기화 블록
 - 인스턴스 변수의 복잡한 초기화에 사용된다
 - 생성자와 같이 인스턴스를 생성할 때 마다 수행
- 클래스 초기화 블록
 - 클래스 변수의 복잡한 초기화에 사용된다
 - 인스턴스 초기화 블록 앞에 **static**를 붙인다
 - 클래스가 메모리에 처음 로딩될 때 한번만 수행
- 메서드 내에서와 같이 조건문, 반복문, 예외처리 구문 등을 사용 할 수 있다

```
class InitBlock{
    static{    //클래스 초기화 블록    }

    {    //인스턴스 초기화 블록    }
}
```

변수 초기화

멤버변수의 초기화 시기와 순서

- 클래스 변수의 초기화 시점
 - 클래스가 처음 로딩 될 때 단 한 번 초기화 된다
 - 프로그램 실행 도중 클래스에 대한 정보가 요구 되어질 때 클래스는 메모리에 로딩된다
- 클래스 변수의 초기화 순서
 - 기본값
 - 명시적 초기화
 - 클래스 초기화 블록

CobinCabin
iRaCha

변수 초기화

멤버변수의 초기화 시기와 순서

- 인스턴스 변수의 초기화 시점
 - 인스턴스가 생성될 때마다 각 인스턴스 별로 초기화가 이루어진다.
- 인스턴스 변수의 초기화 순서
 - 기본값
 - 명시적 초기화
 - 인스턴스 초기화 블록
 - 생성자

CobinCabin
iRaCha