static 이란?

참고: <https://vaert.tistory.com/101> <https://ifcontinue.tistory.com/2>

* 정적인, 움직이지 않는 (메모리에 고정)
* 모든 객체가 공유

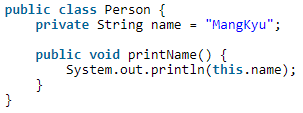
static은 변수나 메소드에 키워드로 사용된다. static 키워드를 사용한 변수는 클래스가 메모리에 올라갈 때 자동으로 생성된다. 즉, 인스턴스(객체) 생성 없이 바로 사용 가능하다. 따라서 객체를 생성하지 않아도 호출할 수 있으므로 편리하고 속도도 빠르다.

* 자주 변하지 않는 일정한 값이나 설정 정보 같은 공용 자원에 대한 접근에 있어 매번 메모리에 로딩하거나 값을 읽어들이는 것보다 일종의 ‘전역변수’와 같은 개념을 통해 접근하는 것이 비용도 줄이고 효율도 높일 수 있다.
* 인스턴스 생성 없이 바로 사용 가능하기 때문에 프로그램 내에서 공통으로 사용되는 데이터들을 관리할 때 이용한다.

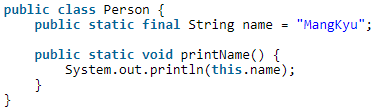
예) <https://mangkyu.tistory.com/47>

Java에서 Static 변수는 메모리에 한번 할당되어 프로그램이 종료될 때 해제되는 변수로, 메모리에 한번 할당되므로 여러 객체가 해당 메모리를 공유하게 됩니다.

예를 들어, 세상 모든 사람의 이름이 'MangKyu'인 세상에 살고있다고 가정을 하겠습니다. 이럴때면 우리는 아래와 같이 객체를 만들 수 있습니다.



하지만 위와 같은 클래스를 통해 100명의 Person 객체를 생성하면, "MangKyu"라는 같은 값을 갖는 메모리가 100개나 중복해서 생성되게 됩니다. 이러한 경우에 static을 사용하여 여러 객체가 하나의 메모리를 참조하도록 하면 메모리 효율이 더욱 높아질 것입니다. 또한 "MangKyu"라는 이름은 결코 변하지 않는 값이므로 final 키워드를 붙여주며, 일반적으로 Static은 상수의 값을 갖는 경우가 많으므로 public으로 선언을 하여 사용합니다. 이러한 이유로, 일반적으로 static 변수는 public 및 final과 함께 사용되어 public static final로 활용 됩니다.



위의 코드를 보면, printName이라는 함수 역시 static 키워드가 붙어서 static메소드가 되었는데, static 변수는 static 메소드를 통해 접근하도록 권장되기 때문입니다.

왜 자바에서 static 사용을 지양해야 하는가?

참고: <https://stackoverflow.com/questions/7026507/why-are-static-variables-considered-evil%22>

번역: <https://unabated.tistory.com/entry/왜-자바에서-static의-사용을-지양해야-하는가>

문제점

* 메모리 회수 불가
* Static이 붙은 변수는 클래스와 같은 영역에 생기고 클래스와 동일하게 메모리에서 항상 상주하게 된다. 이 때문에 static으로 어떤 데이터나 객체를 연결해서 사용하게 되면 메모리 회수가 되지 않아 시스템이 가동되면서 점점 느려진다.
* 지나치게 많은 static 변수를 사용하게 된다면 메모리 회수를 할 수 없어서 가상머신이 메모리 부족을 겪게 된다.
* 객체지향적이지 않다.
* Static 변수는 각 객체의 데이터들이 캡슐화(외부에서 접근하여 수정할 수 없도록 해야 한다는) 원칙에 위배된다. Static은 스코프(변수가 유효한 범위)를 고려할 필요가 없는 경우, 즉 전역변수를 사용할 때 유용하며 이는 절차지향적 프로그래밍 관점에서 중요한 개념이다.
* 테스트, 확장 등이 어려움
* 코드 여러부분에서 영향을 받을 수 있게 되고 테스트가 어려워진다. 반면에 이를 객체화하여 인스턴스로 생성하면 테스트 중에 그 변수가 어떤 상태를 가지고 있는지 추론하는 것이 보다 간단해진다.
* 하나의 가상머신에서 어떤 프로그램 두 카피가 돌고 있다고 가정할 때, 만약 이 두 카피가 동일한 static 변수를 공유하게 된다면 서로의 상태에 영향을 주게 된다. 더불어 오버라이딩을 할 수 없는 static 멤버들 때문에 클래스를 확장하는 것이 어려워진다.

정적 메소드는 언제 사용해야 하는가?

참고: <https://stackoverflow.com/questions/2671496/java-when-to-use-static-methods>

1. 유틸리티 클래스로 작성되고, 변화를 가정하지 않는다.
2. 메소드가 인스턴스 변수를 사용하지 않는다.
3. 인스턴스 생성에 의존하지 않는다.
4. 메소드가 공유되고 있다면 정적 메소드로 추출해낼 수 있다.
5. 메소드가 변화되지 않고 오버라이딩 되지 않는다.

오버라이딩(Overriding)

상속(extends)이란 상위 클래스에 있는 변수, 메소드 등을 전부 가져와서 쓸 수 있게 만드는 것이다. 만약 부모에게 상속받은 메소드 중 마음에 들지 않는 것이 있을 때 @Override로 재정의하여 내가 원하는 대로 코딩할 수 있다. 오버라이딩된 메소드는 부모 클래스의 원래 메소드보다 우선되기 때문에 자식 객체에서 메소드 호출 시 오버라이딩 된 메소드가 호출된다.

1. 원래 메소드와 동일한 리턴 타입, 메소드 이름, 매개변수 리스트를 가져야 한다.
2. 접근 제한을 더 강하게 오버라이딩할 수 없다.
3. 새로운 예외를 throws할 수 없다.

오버로딩(Overloading)

같은 이름의 생성자 또는 메소드를 여러 개 선언하는 것

매개값을 다양하게 입력받아 처리할 수 있도록 하기 위함이다.

* 매개변수의 타입, 개수, 순서 중 하나가 달라야 한다.
* 리턴타입만 다른 경우는 메소드 오버로딩이 성립되지 않는다.

ex) setCellValue(int, int, string)과 setCellValue(int, int, double)