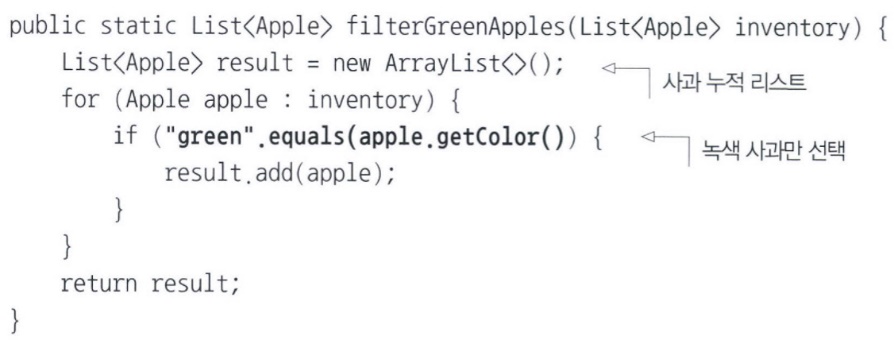
**[이 장의 내용]**

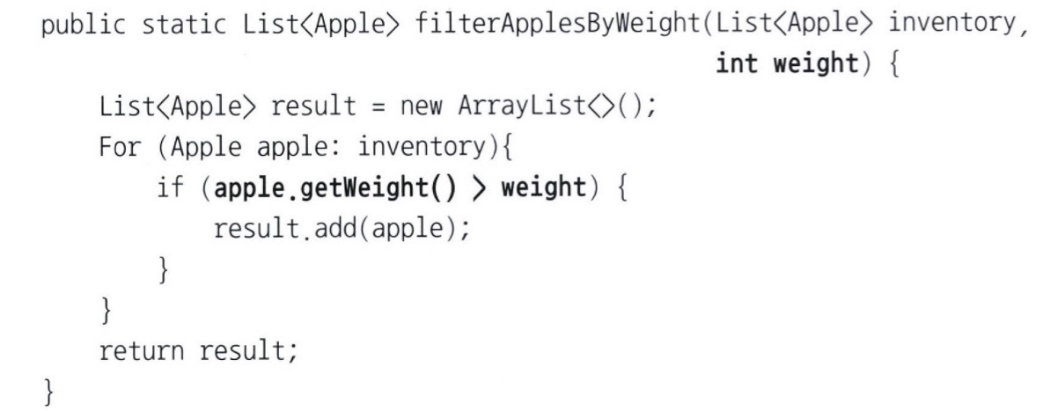
* 변화하는 요구사항에 대응
* 동작 파라미터화
* 익명 클래스
* 람다 표현식 미리보기
* 실전 예제

**2.1 변화하는 요구사항에 대응하기**

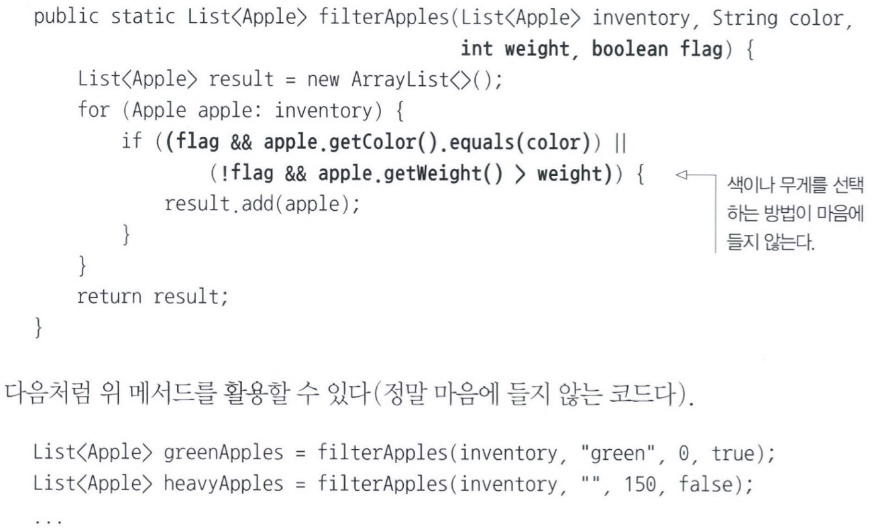
2.1.1 첫번째 시도: 녹색 사과 필터링

* 
  + 농부가 처음에는 녹색 사과만 필터링을 진행했다가, 옅은 녹색, 어두운 빨간색, 노란색 등으로 필터링하고자 할때 단순히 복사 붙여넣기로는 깰끔한 코드가 되지 않음
* 비슷한 코드를 구현한 다음 추상화해라
  + 해당 방법으로 요구사항에 대응해야 함

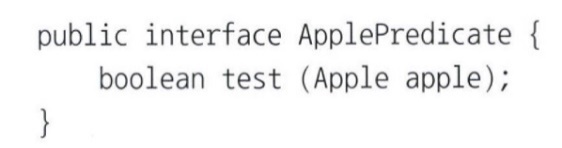
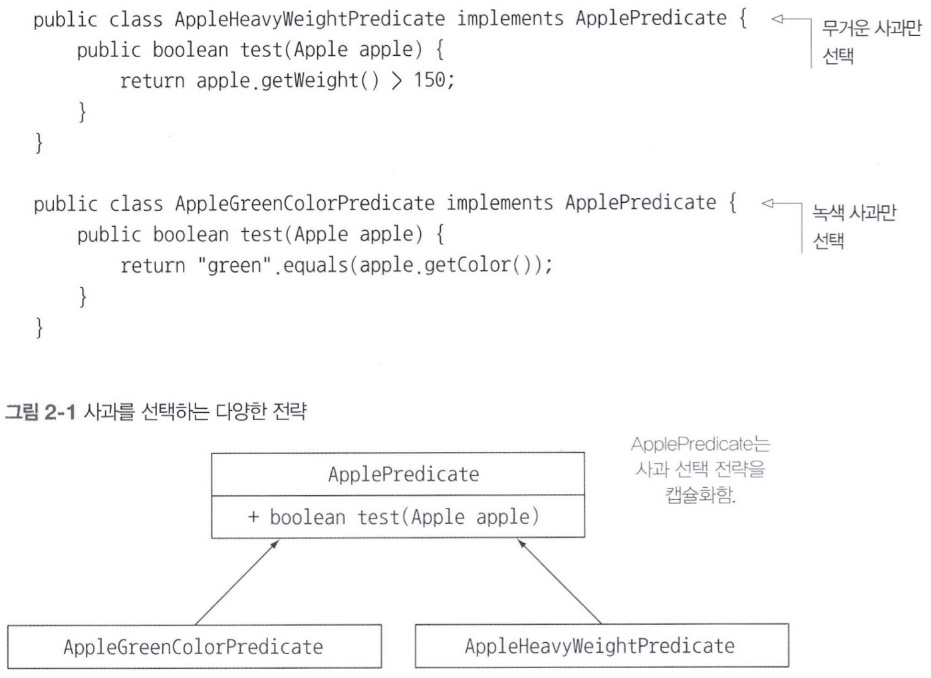
2.1.2 두번째 시도 : 색을 파라미터화

* 
  + 색을 파라미터화
* 
  + 위와 같이 파라미터를 좀 더 큰 개념으로 두어 받을 수 있음.
* 농부가 색 뿐만 아니라 무게도 추가해달라는 요구사항을 제시
  + 
    - 좋은 해결책이지만, 색 파라미터화의 코드와 중복되는 부분이 많음

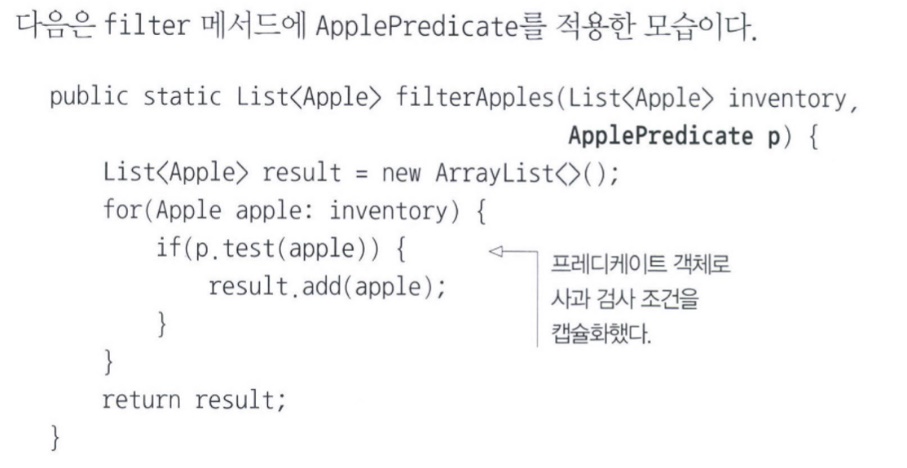
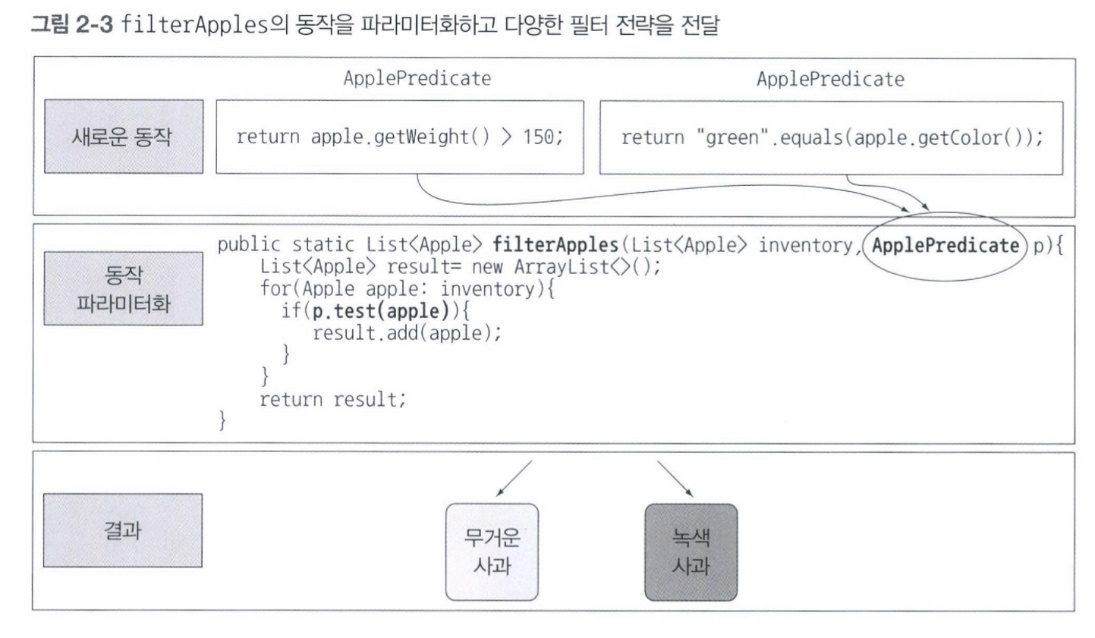
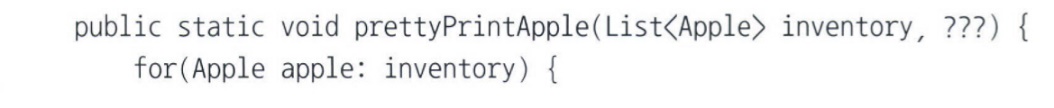
2.1.3 세번째 시도 : 가능한 모든 속성으로 필터링

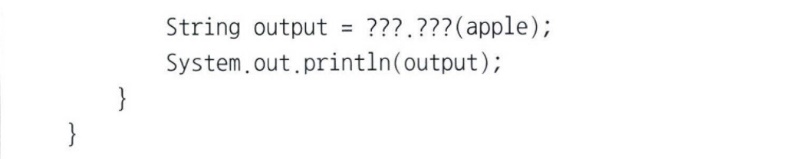
* 
  + true와 false는 무엇을 의미?
  + 또 다른 요구사항이 나타날 시 대응 가능?

**2.2 동작 파라미터화**

* 프레디케이트
  + True? False?
    - 사과의 어떤 속성에 기초해서 불린값을 반환
    - Ex ) 사과가 녹색? 150그램? Return boolean
  + 선택 조건을 결정하는 인터페이스 정의(**전략 디자인 패턴**)
    - 
    - 
    - 위 조건에 따라 filter 메서드는 다르게 동작할 것을 예상
  + 전략 디자인 패턴
    - 알고리즘 패밀리(각 알고리즘을 캡슐화)를 정희해둔 다음 런타임에 알고리즘을 선택
* 메서드의 다양한 동작
  + 동작 파라미터화는 메서드가 다양한 동작을 받아서 다양한 수행을 진행

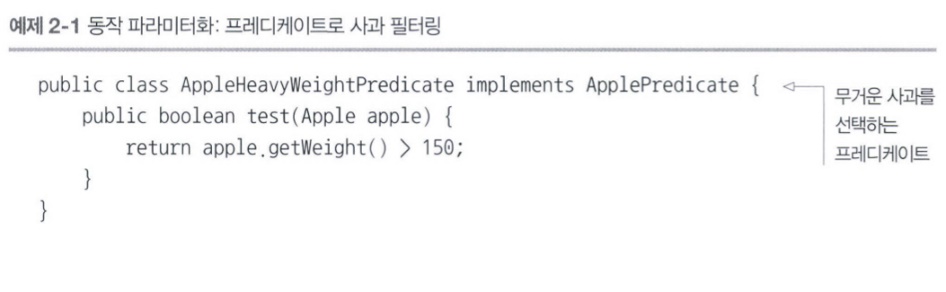
2.2.1 네번째 시도 : 추상적 조건으로 필터링

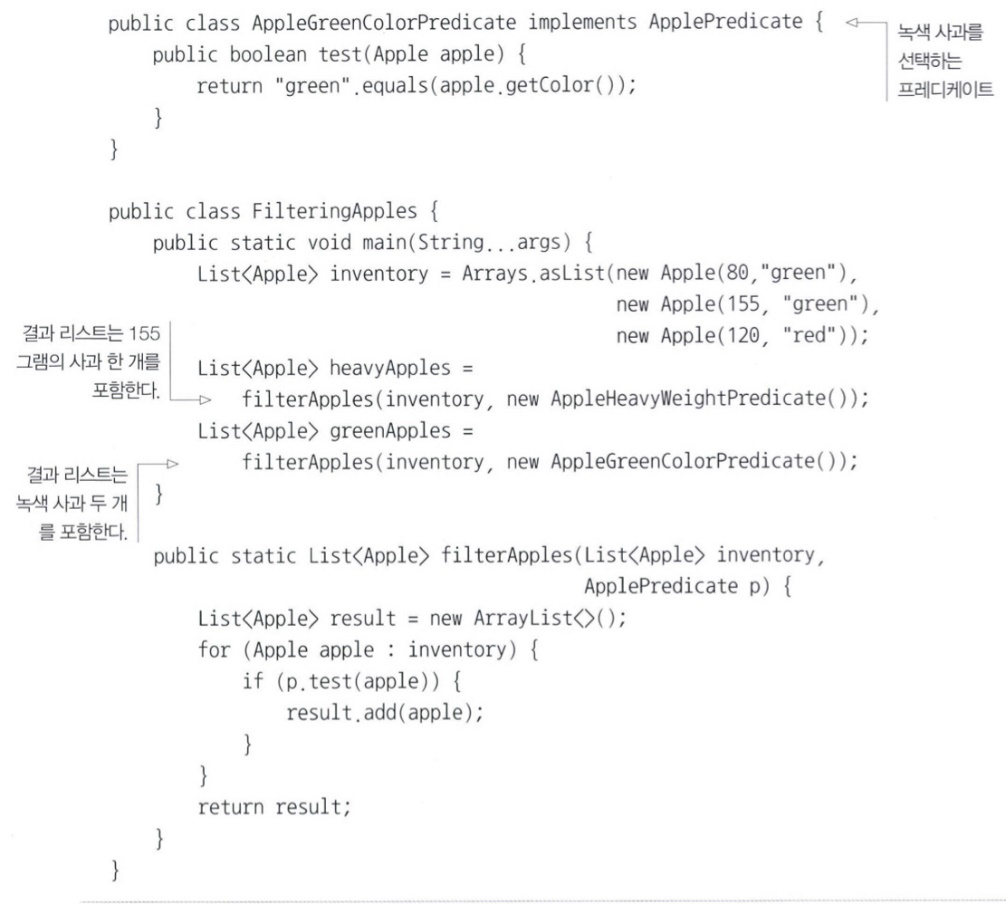
* 
  + 전달한 ApplePredicate 객체에 의해 filterApples 메서드의 동작이 결정
* 
  + 해당 메소드에서 true일 경우에만 apple이 add됨
  + 즉, 농부가 원하는 것이 무엇인지 우리는 Predicate에 정의만 해놓으면 되는 것
* 퀴즈(각각의 사과 무게 출력, 무겁냐? 가볍냐? 출력)
  + 



* + 풀이
    - Interface PrettyApplePredicate 구현
    - PrettyAppleWeight & EachPrettyAppleWeight 구현
    - 첫 ??? : PrettyApplePredicate p
    - 두 ???? : p.test(apple);

**2.3 복잡한 과정 간소화**

* 익명 클래스의 필요성
  + 

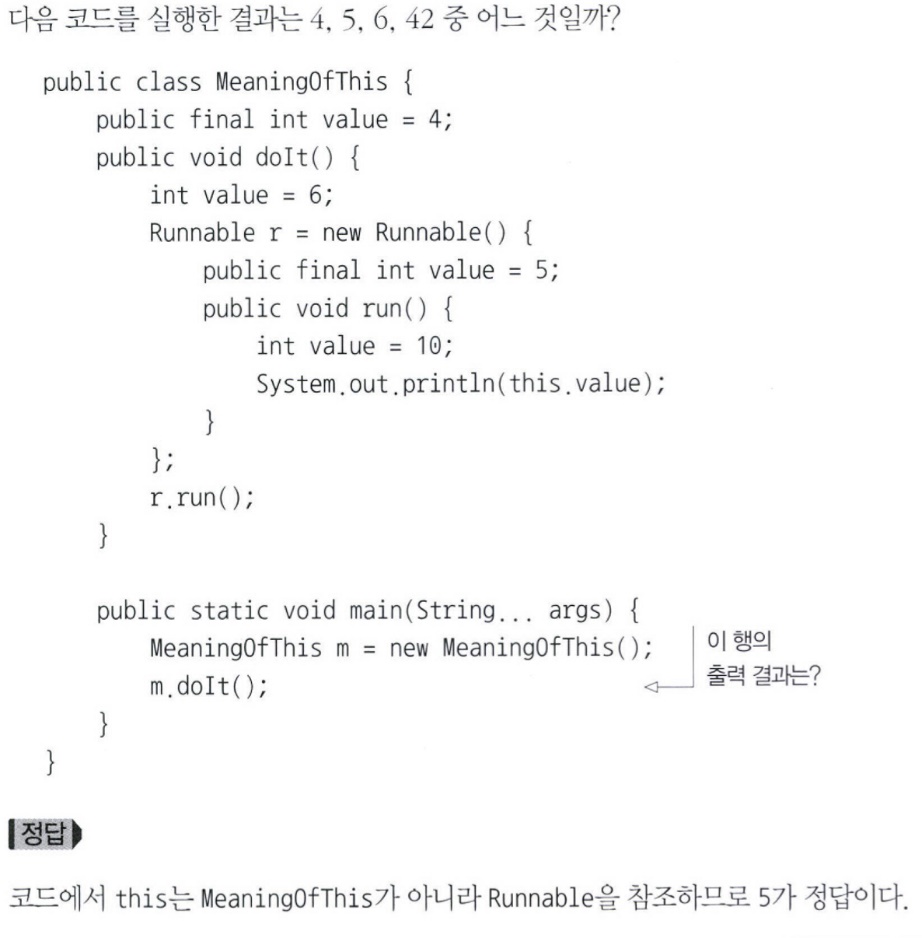


* + 로직과 관련 없는 코드가 추가 됨..

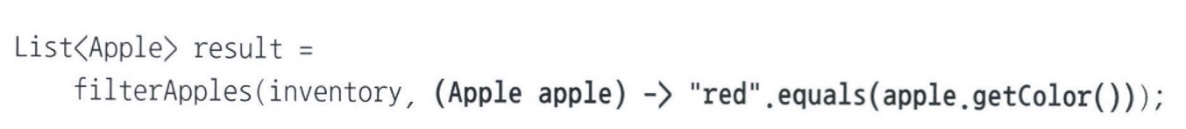
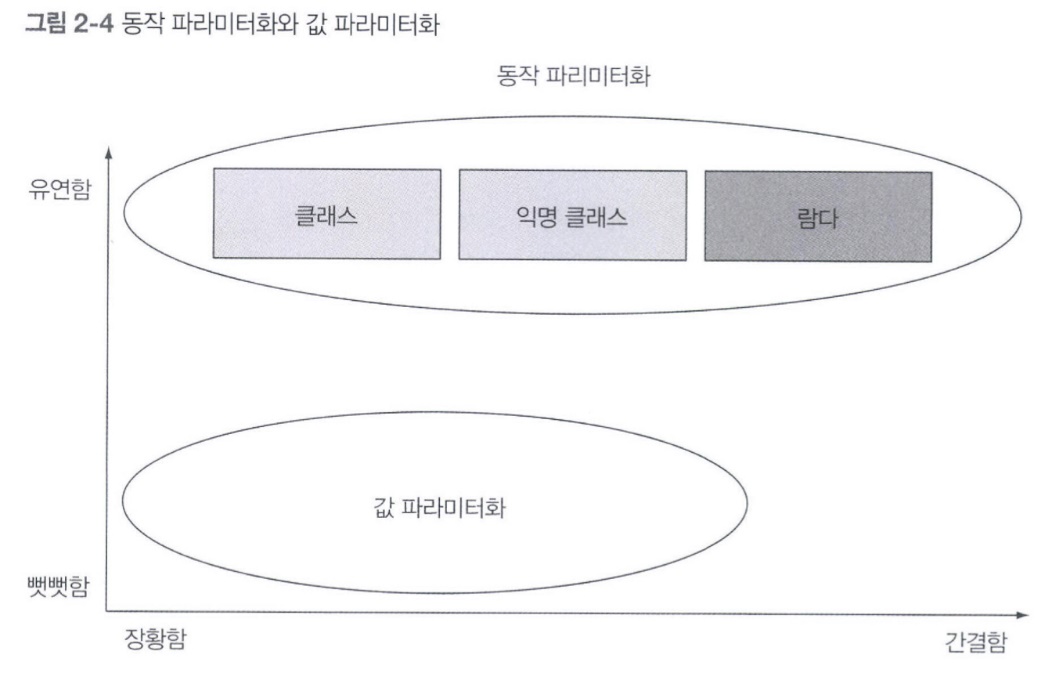
2.3.1 익명 클래스

* 클래스 선언과 인스턴스화를 동시에 가능

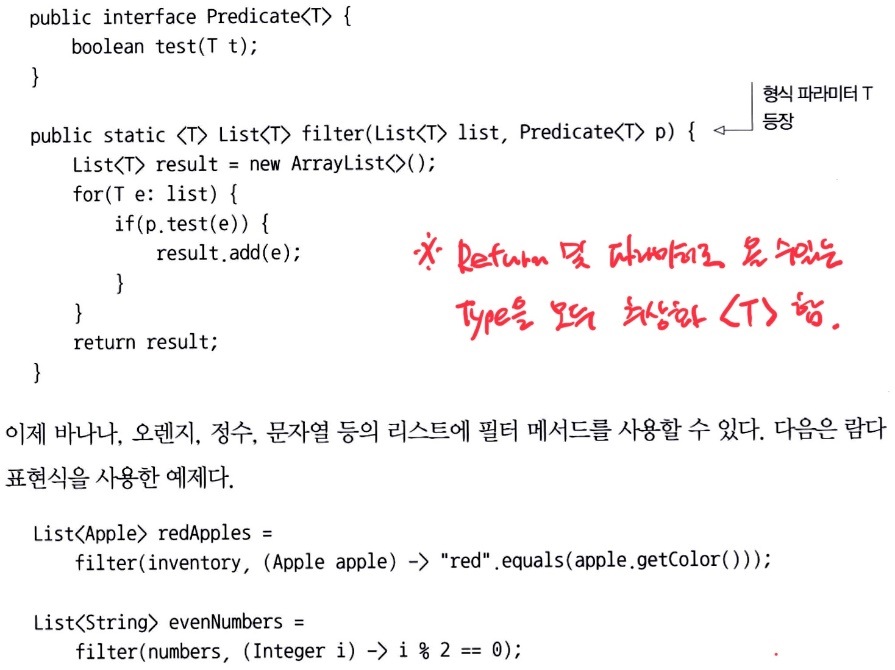
2.3.2 다섯번째 시도 : 익명 클래스 사용

* 
* 부족한 점 발생
  + 첫째, 익명 클래스는 많은 공간을 차지
  + 둘째, 익명 클래스의 사용에 익숙치 않음
* 익명 클래스 예제
  + 

2.3.3 여섯번째 시도 : 람다 표현식 사용

* 
* 동작 파라미터화와 값 파라미터화의 차이
  + 

2.3.4 일곱번째 시도 : 리스트 형식으로 추상화

* 현재 filterApples는 Apple과 관련된 동작만 수행함
  + 하지만 Apple 이외의 다양한 물건에서 필터링이 작동하도록 추상화 할 수 있음
  + 

**2.4 실전 예제**

* 코드 전달 개념, 좀 더 이해하자!
  + Comparator로 정렬하기
  + Runnable로 코드 블록 실행하기
  + GUI 이벤트 처리하기

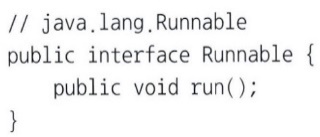
2.4.1 Comparator로 정렬하기

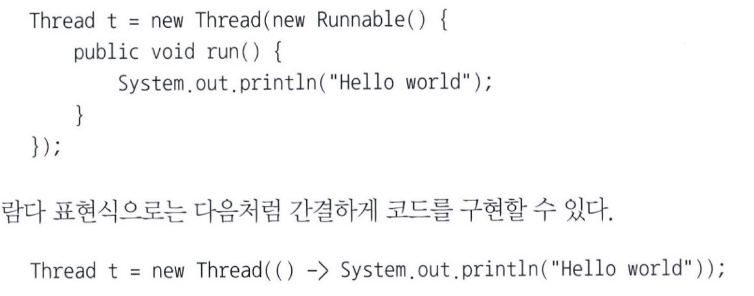
* 자바 8 Comparator의 sort 메서드의 동작을 다양화 시킴
  + 



* + - 람다식 표현
  + 농부의 요구사항이 바뀌면 새로운 요구사항에 맞는 Comparator를 만들어 sort 메서드에 전달함으로 해결 가능!

2.4.2 Runnable로 코드 블록 실행하기

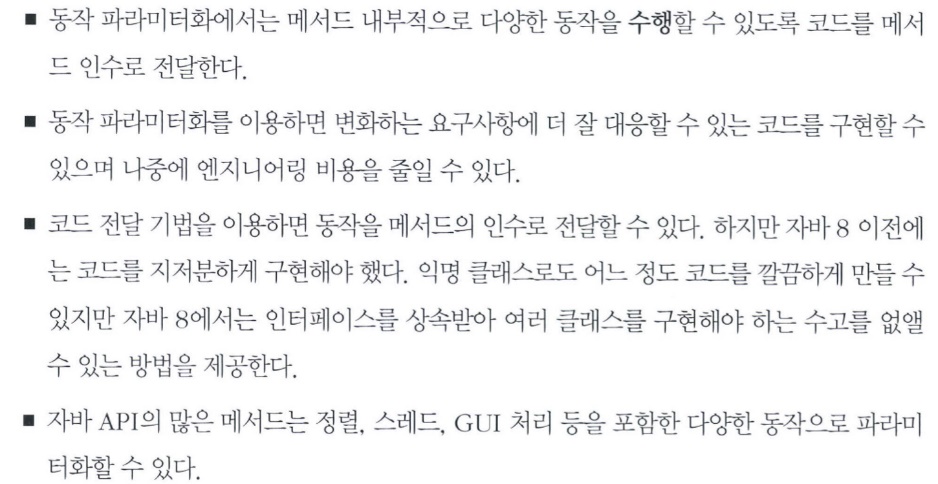
* Runnable을 이용해서 다양한 동작을 스레드로 실행 가능
  + 



2.4.3 GUI 이벤트 처리하기

* GUI 프로그래밍에서도 변화에 대응하자

**2.5 요약**

* 

질문) 전략 디자인 패턴에 대해 자세한 설명 부탁드려욥