아이템 2. 생성자에 매개변수가 많다면 빌더를 고려하라

생성일	@2021년 8월 19일 오후 6:32
태그	

아이템 2. 생성자에 매개변수가 많다면 빌더를 고려하라

```
public NutritionFacts(int servingSize, int servings) { ...}
public NutritionFacts(int servingSize, int servings, int calories) {...}
```

- 점층적 생성자 패턴
  - 기존에는 위 코드처럼 점층적 생성자 패턴을 즐겨 사용했었다. 하지만 매개변수가 많아질 수록 클라이언트 코드를 작성하거나 읽기 어렵다!

```
public class NutritionFacts {
  private int servingSize = -1;
  private int servings = -1;

  public NutritionFacts(){}
  //세터 메서드
  public void setServingSize(int val) {this.servingSize = val;}
  public void setServings(int val) {this.servings = val;}
}
```

- 자바빈즈 패턴
  - 이러한 자바빈즈 패턴의 단점은 객체 하나를 만들려면 여러 개의 메서드를 호출해야 하고, 객체가 완전히 생성되기 전까지는 일관성이 무너진 상태에 놓이게 된다.
  - 일관성이 무너지는 문제 때문에 클래스를 불변으로도 만들 수 없다.

```
public class NutritionFacts {
  private final int servingSize;
  private final int servings;
  private final int calories;

  public static class Builder {
    private final int servingSize;
```

```
    private final int servings;
    private int calories = 0;

    public Builder(int servingSize, int servings) {
      this.servingSize = servingSize;
      this.servings = servings;
    }

    public Builder calories(int val) {this.calories = val; return this;}
    public NutritionFacts build() { return new NutritionFacts(this);}
  }

  private NutritionFacts(Builder builder) {
    this.servingSize = builder.servingSize;
    this.servings = builder.servings;
    this.calories = builder.calories;
  }
}
```

- 빌더패턴
  - 체이닝 기법을 사용하여 사용자가 보기에 매우 직관적이다. 명명된 선택적 매개변수를 흉내낸 것이기도 하다.
  - 사용 예) NutritionFacts cocaCola = new NutritionFacts.Builder(240,8).calories(100).build();
  - t생성자나 정적 팩토리가 처리해야 할 매개변수가 많다면 빌더 패턴을 선택하는게 더 낫다.