3장. 타입과 추상화

해리 벡의 영국 지하철 노선도

- 초기의 지하철 노선도는 실제와 유사한 물리적 지형 위에 노선과 역 간의 거리를 사실적으로 묘사하고 있었다.
  - 사실적인 정보가 오히려 노선도를 이해하기 어렵게 만든다.
  - 사람들이 알고 싶은 건 역과 역 사이의 연결성을 얼마나 직관적으로 표현했는가이다.
- 해리 벡이 지형과 축척은 무시하고 역 사이의 연결성에만 집중한 혁신적인 노선도를 창조했다.
  - 지도의 '정확성'을 버리고, 노선도를 이용하는 '목적'에 집중했다.
  - 승객이 꼭 알아야 하는 사실만 정확하게 표현하고 몰라도 되는 정보는 무시하는 '추상화'를 적용했다.

추상화를 통한 복잡성 극복

- 진정한 의미의 추상화란
  - 현실에서 출발하되 불필요한 부분을 도려내가면서 사물의 본질을 드러나게 하는 과정
  - 불필요한 부분을 무시함으로써 현실에 존재하는 복잡성을 극복하여 단순화하는 것
- 훌륭한 추상화란
  - 목적에 부합하는 것 (추상화의 수준, 이익, 가치는 목적에 의존적이다.)

추상화

> 어떤 양상, 세부사항, 구조를 좀 더 명확하게 이해하기 위해 특정 절차나 물체를 의도적으로 생략하거나 감춤으로써
> 복잡도를 극복하는 방법

- 첫째, 구체적인 사물들 간의 공통점은 취하고 차이점은 버리는 일반화를 통해 단순하게 만든다.

- 둘째, 중요한 부분을 강조하기 위해 불필요한 세부 사항을 제거함으로써 단순하게 만든다.

객체지향과 추상화

> ♣ *... 정원사들은 트럼프처럼 생겼고 몸에는 스페이드 무늬가 그려져 있었다. ...*
> *... 그들은 정원사들처럼 몸이 납작했고 네모난 몸 모서리에 팔다리가 달려 있었다.*
> *...*
> *... 손님들 뒤로 왕관을 얹은 빨간 비단 방석을 든 하트 잭이 걸어왔다.*
> *그리고 행렬의 마지막에 이르러 마침내 하트 왕과 하트 여왕이 그 모습을 드러냈다.*
> *...*
>
> ***'기껏 해야 트럼프에 불과해. 무서워할 필요 없어."***

- 정원사, 하트 왕, 하트 여왕의 차이점은 무시하고 '트럼프'라는 유사성을 기반으로 **추상화**하여 바라본다.

그룹으로 나누어 단순화하기

- 정원사, 병사, 신하, 왕자와 공주, 왕과 왕비들, 하트 잭, 하트 왕과 하트 여왕
  - 명확한 경계를 가지고 서로 구별할 수 있기 때문에 이들 각각은 **객체**이다.
  - 이들 각각의 차이점을 무시하면 '트럼프'라고 줄여 말할 수 있다. (**추상화**)
- 결과적으로 앨리스는 정원에 있는 인물들을 트럼프와 토끼 두 그룹으로 나눴다.
  - 다수의 개별적 인물이 아닌 두 그룹으로 정원을 바라보기 때문에 정원에 내재된 복잡성을 효과적으로 감소시킨다.

개념

- 공통점을 기반으로 객체들을 묶기 위한 그릇을 **개념(concept)**라고 한다.
  - '트럼프'는 공통점을 가진 객체들을 포괄할 수 있는 개념이다.
- 개념을 이용하면 객체를 여러 그룹으로 **분류(classification)**할 수 있다.
  - 개념은 공통점을 기반으로 객체를 분류할 수 있는 일종의 체 역할을 한다.
- 객체가 어떤 개념 그룹의 일원이 될 때 객체를 그 개념의 **인스턴스(instance)**라고 한다.
  - 객체란 특정한 개념을 적용할 수 있는 구체적인 사물을 의미한다.

- 개념이 객체에 적용됐을 때, 객체를 개념의 인스턴스라고 한다.

개념의 세 가지 관점

- **심볼(symbol)** : 개념을 가리키는 간략한 이름이나 명칭

> 트럼프

- **내연(intension)** : 개념의 완전한 정의를 나타내며, 내연을 통해 객체가 개념에 속하는지 여부를 확인할 수 있음

> 몸이 납작하고 두 손과 두 발이 네모난 몸 모서리에 달려있다.

- **외연(extension)** : 개념에 속하는 모든 객체의 집합(set)

> 정원사, 병사, 신하, 왕자와 공주, 하객으로 참석한 왕과 왕비들, 하트 잭, 하트 왕과 하트 여왕

객체를 분류하기 위한 틀

- **분류**란 객체에 특정한 개념을 적용하는 작업이다.
  - 객체를 적절한 개념에 따라 분류하지 못하면 유지보수가 어렵고 변화에 쉽게 대처하기 힘들다.
  - 객체를 적절한 개념에 따라 분류하면 유지보수가 용이하고 변경에 유연하게 대처할 수 있다.
  - 따라서, 인지능력을 발휘하여 최대한 직관적으로 분류해야 한다.

분류는 추상화를 위한 도구다

- 추상화를 통해 복잡한 세상을 제어 가능한 수준으로 단순화할 수 있다.

타입

타입 == 개념

- 컴공 세상에서 '**타입**'이란 앞에서 설명한 '**개념**'과 완전히 동일한 의미를 가진다.

데이터 타입

- 메모리 안에 저장된 '10010001001001...'과 같은 값이 어떤 타입인 지 어떻게 알 수 있을까?
    - 데이터에 다른 데이터를 더하거나 빼거나 나누거나 곱할 수 있음 → 숫자형으로 분류
    - 데이터가 여러 문자로 구성되어 있고 다른 문자와 연결될 수 있음 → 문자열형으로 분류
    - 데이터를 이용해 어떤 사실에 대한 참/거짓을 이야기할 수 있음 → 논리형으로 분류

    → 데이터를 목적에 따라 분류하기 시작하면서 프로그램이 언어 안에 **타입 시스템**이 자라나기 시작했다.

    → 타입 시스템의 목적은 데이터가 잘못 사용되지 않도록 제약사항을 부과하는 것이다.
- **데이터 타입에 대한 중요한 두 가지 사실** 💡
    - 어떤 데이터가 어떤 타입에 속하는 지 결정하는 것은 **데이터에 적용할 수 있는 작업**이다.
        - 어떤 데이터에 어떤 연산자를 적용할 수 있느냐가 그 데이터의 타입을 결정한다.
    - 타입에 속한 데이터를 메모리에 어떻게 표현하는지는 외부로부터 철저하게 감춰진다.
        - 연산 작업을 효과적으로 수행하기 위한 표현이 선택되며, 개발자는 표현 방식을 몰라도 사용하는 데 지장이 없다.

객체와 타입

- 객체지향 프로그램을 작성할 때 우리는 객체를 일종의 데이터처럼 사용한다.
- 객체를 타입에 따라 분류하고 타입 이름을 정하는 것은 프로그램에서 사용할 새로운 데이터 타입을 선언하는 것과 같다.
- **타입에 대한 중요한 두 가지 사실** 💡
    - 어떤 객체가 어떤 타입에 속하는 지 결정하는 것은 **객체가 수행하는 행동**이다.
        - 어떤 객체들이 동일한 행동을 수행할 수 있다면 동일한 타입으로 분류될 수 있다.
    - 객체의 내부적인 표현은 외부로부터 철저하게 감춰진다.
        - 행동을 효과적으로 수행할 수 있다면, 내부 상태를 어떤 방식으로 표현하더라도 무방하다.

행동이 우선이다

- 객체의 타입을 결정하는 것은 동일한 **행동**뿐이다.
  - **동일한 행동 == 동일한 책임 == 동일한 메시지 수신**
  - 이 때, 동일한 메시지를 처리하는 방식은 서로 다를 수 있다. (**다형성**)
- 외부에 행동만을 제공하고 데이터는 행동 뒤로 감춰야 한다. (**캡슐화**)
- 객체의 내부 데이터가 아닌, 객체가 외부에 제공해야 하는 행동을 먼저 생각해야한다.
  - 객체가 외부에 제공해야 하는 책임을 먼저 결정하고
  - 그 책임을 수행하는 데 적합한 데이터를 나중에 결정한 후
  - 데이터를 책임을 수행하는 데 필요한 외부 인터페이스 뒤로 캡슐화해야 한다.

  → **책임-주도 설계(Responsibility-Driven Design)**은 DDD의 단점을 개선하기 위해 고안되었다.

타입의 계층

트럼프 계층

- 트럼프 카드는 걸어다닐 수 없는데 '트럼프'가 아닌 '트럼프 인간'으로 분류해야하지 않을까?
  - '트럼프'는 '납작 엎드리고 뒤집어질 수 있다'.
  - '트럼프 인간'은 '납작 엎드리고 뒤집어질 수 있고 걸을 수 있다'.
  - 트럼프는 트럼프 인간이 될 수 없지만, 트럼프 인간은 트럼프가 될 수 있다.

  → 트럼프는 트럼프 인간을 포괄하는 좀 더 일반적인 개념이다. (**일반화/특수화 관계**)

일반화/특수화 관계

- 일반화/특수화 관계를 결정하는 것은 **객체의 행동**이다.
  - 일반적인 타입은 특수한 타입에 비해 더 적은 수의 행동을 가진다.
  - 이 때 특수한 타입은 일반적인 타입이 할 수 있는 모든 행동을 동일하게 수행할 수 있어야 한다.

슈퍼타입과 서브타입

- 일반적인 타입을 **슈퍼타입(Supertype)**, 특수한 타입을 **서브타입(Subtype)**이라고 한다.

- 서브타입은 슈퍼타입을 대체할 수 있어야 한다.
- 슈퍼타입의 행동은 서브타입에게 자동으로 상속된다.

일반화는 추상화를 위한 도구다

- 객체지향에서 세상을 바라볼 때 분류와 일반화/특수화 기법을 동시에 적용하게 된다.

정적 모델

타입의 목적

- 시간에 따라 동적으로 변하는 객체의 복잡성을 극복하기 어렵다.
- 타입은 상태에서 시간이라는 요소를 제거함으로써 시간에 독립적인 정적인 모습으로 객체를 다룰 수 있게 해준다.

그래서 결국 타입은 추상화다

- 타입을 이용하면 객체의 동적인 특성을 추상화할 수 있다.

동적 모델과 정적 모델

- **동적 모델**
  - 스냅샷처럼 객체가 살아 움직이는 동안 상태가 어떻게 변하고 어떻게 행동하는지 포착하는 것
- **타입 모델 == 정적 모델**
  - 객체가 가질 수 있는 모든 상태와 모든 행동을 시간에 독립적으로 표현하는 것
- 객체 관점의 동적 모델과 객체를 추상화한 타입 관점의 정적 모델을 적절히 혼용해서 사용해야 한다.

클래스

- 클래스 ≠ 타입
  - 객체를 분류하는 기준은 타입이며, 타입을 나누는 기준은 객체가 수행하는 행동이다.
  - 객체를 분류하기 위해 타입을 결정한 후, 이를 구현할 수 있는 방법 중 하나가 클래스이다.
- 클래스는 타입을 구현하기 위해 프로그래밍 언어에서 제공하는 구현 매커니즘일 뿐이다.