



3. 타입과 추상화

🕒 추상화를 통한 복잡성 극복

🕒 객체지향과 추상화

모두 트럼프일 뿐

그룹으로 나누어 단순화하기

개념

개념의 세 가지 관점

객체를 분류하기 위한 틀

분류는 추상화를 위한 도구다

🕒 타입

타입은 개념이다

데이터 타입

객체와 타입

행동이 우선이다



🕒 타입의 계층

트럼프 계층

일반화/특수화 관계

슈퍼타입과 서브타입

일반화는 추상화를 위한 도구다

🕒 정적 모델

타입의 목적

그래서 결국 타입은 추상화다

동적 모델과 정적 모델

클래스

“일단 컴퓨터를 조작하는 것이 추상화를 구축하고, 조작하고, 추론하는 것에 관한 모든 것이라는 것을 깨닫고 나면, (훌륭한) 컴퓨터 프로그램을 작성하기 위한 중요한 전제 조건은 추상화를 정확하게 다루는 능력이라는 것이 명확해진다.”

🕒 추상화를 통한 복잡성 극복

- 추상화란?

- 어떤 양상, 세부 사항, 구조를 좀 더 명확하게 이해하기 위해 불필요한 부분을 생략하거나 감춤으로써 **현실에 존재하는 복잡성을 극복**하는 방법
- 복잡성을 다루기 위해 추상화는 다음의 두 차원에서 이루어진다.
 1. 구체적인 사물들 간의 공통점은 취하고, 차이점은 버리는 **일반화**를 통해 단순하게 만드는 것
 2. 중요한 부분을 강조하기 위해 **불필요한 세부 사항을 제거**함으로써 단순하게 만드는 것
- 결국 추상화의 목적은 **복잡성을 이해하기 쉬운 수준으로 단순화**하는 것이다.

“현상은 복잡하다. 법칙은 단순하다. 버릴 게 무엇인지 알아내라.”



객체지향과 추상화

모두 트럼프일 뿐

정원사, 왕자와 공주, 신하, 하객, 하트 왕과 여왕 모두 트럼프의 모양을 띄고 있었는데, 앨리스는 이들을 보며 ‘기껏해야 트럼프에 불과해’라고 읊조렸다. 앨리스는 ‘트럼프’라는 **유사성**을 기반으로 **추상화**해서 바라보고 있는 것이다.

그룹으로 나누어 단순화하기

정원사, 왕자와 공주, 신하, 하객, 하트 왕과 여왕 모두 각자만의 독특한 특징이 있기에 서로 구분할 수 있다.

하지만 앨리스는 이들의 차이점을 무시한 채 하나의 **그룹**으로 묶어서 바라보았다.

즉 앨리스는 인물들을 하나씩 살펴보면서 자신이 알고 있는 ‘트럼프’의 의미에 적합한 인물은 ‘트럼프’ 그룹에 포함하고, 토끼같이 적합하지 않은 인물은 ‘트럼프’ 그룹에서 제외했다.

트럼프 그룹과 토끼 그룹이라는 두 개의 렌즈로 정원을 바라보는 것은 **복잡성을 감소**시킨다.

개념

- 앨리스가 인물들의 **차이점을 무시하고 공통점만을 취해 단순화**한 것은 추상화의 일종이다.
- 이처럼 공통점을 기반으로 객체들을 묶기 위한 그릇을 **개념(concept)**이라고 한다.
- 개념을 이용하면 객체를 여러 그룹으로 **분류(classification)**할 수 있다.
- **객체**란 특정한 개념을 적용할 수 있는 구체적인 사물을 의미하며, 개념이 객체에 적용되었을 때 객체를 개념의 **인스턴스(instance)**라고 한다.

개념의 세 가지 관점

- 객체의 분류 장치로서 개념을 이야기할 때는 아래의 세 가지 관점을 함께 언급한다.
 - **심볼(symbol)**: 개념을 가리키는 간략한 이름이나 명칭
예) 트럼프
 - **내연(intension)**: 개념의 완전한 정의를 나타내며, 객체가 개념에 속하는지 여부를 판단하기 위한 조건
예) 몸이 납작하고 두 손과 두 발은 네모 귀통이에 달려 있는 등장인물
 - **외연(extension)**: 개념에 속하는 모든 객체의 집합(set)
예) 정원사, 병사, 신하, 왕자와 공주, 하객, 하트 왕과 하트 여왕
- 개념을 구성하는 심볼, 내연, 외연은 객체의 분류 방식에 대한 지침을 제공한다.
- 객체지향의 세계에서 가장 널리 알려진 유명인사가 클래스(class)라는 사실을 감안한다면 분류(classification)라는 개념이 얼마나 중요한지 실감할 수 있다.

객체를 분류하기 위한 틀

- **분류란 객체에 특정한 개념을 적용하는 작업이다. 객체에 특정한 개념을 적용하기로 결심했을 때 우리는 그 객체를 특정한 집합의 멤버로 분류하고 있는 것이다.**
- 객체를 적절한 개념에 따라 분류하면
 - 유지보수가 용이하고 변경이 유연하게 대처할 수 있다.
 - 개발자의 머릿속에 객체를 쉽게 찾고 조작할 수 있는 정신적인 지도를 제공한다.

분류는 추상화를 위한 도구다

- 개념을 통해 분류하는 과정은 추상화의 두 가지 차원을 모두 사용한다.
 1. 구체적인 사물 간의 공통점은 취하고 차이점은 버리는 **일반화**
 2. 중요한 부분을 강조하기 위해 불필요한 세부 사항을 제거해 **단순화**
- 추상화를 사용함으로써 우리는 극도로 복잡한 이 세상을 조금이나마 단순화할 수 있다!

타입

타입은 개념이다

- **타입 == 개념**

- 타입은 우리가 인식하고 있는 다양한 사물이나 객체에 적용할 수 있는 아이디어나 관념을 의미한다.
- 어떤 객체에 타입을 적용할 수 있을 때 그 객체를 타입의 인스턴스라고 하며, 타입을 구성하는 외연이 된다.

데이터 타입

- 데이터 타입에 관련된 두 가지 사실
 1. **타입은 데이터가 어떻게 사용되느냐에 관한 것이다.**
 - a. 데이터가 어떤 타입에 속하는지를 결정하는 것은 데이터에 적용할 수 있는 작업이다.
 - b. 어떤 데이터에 어떤 연산자를 적용할 수 있느냐가 그 데이터의 타입을 결정한다.
 2. **타입에 속한 데이터를 메모리에 어떻게 표현하는지는 외부로부터 철저하게 감춰진다.**
 - a. 데이터 타입의 표현은 연산 작업을 수행하기에 가장 효과적인 형태가 선택되며,
 - b. 개발자는 데이터 타입의 표현 방식을 몰라도 데이터를 사용하는 데 지장이 없다.
- 프로그래밍 언어 관점에서 데이터 타입이란?

데이터 타입은 메모리 안에 저장된 데이터의 종류를 분류하는 데 사용하는 메모리 집합에 관한 메타데이터다.

데이터에 대한 분류는 암시적으로 어떤 종류의 연산이 해당 데이터에 대해 수행될 수 있는지를 결정한다.

객체와 타입

- 객체를 타입에 따라 분류하고 그 타입에 이름을 붙이는 것은 결국 프로그램에서 사용할 새로운 데이터 타입을 선언하는 것과 같다.
 - 객체가 협력을 위해 어떤 책임을 지녀야 하는지를 결정하는 것이 객체지향 설계의 핵심이다.
- 따라서 데이터 타입에 관련된 두 가지 사실은 객체의 타입을 이야기할 때도 동일하게 적용된다.

1. **어떤 객체가 어떤 타입에 속하는지를 결정하는 것은 객체가 수행하는 행동이다.**
어떤 객체들이 동일한 행동을 수행할 수 있다면 그 객체들은 동일한 타입으로 분류

될 수 있다.

2. 객체의 내부적인 표현은 외부로부터 철저하게 감춰진다.

객체의 행동을 가장 효과적으로 수행할 수만 있다면 객체 내부의 상태를 어떤 방식으로 표현하더라도 무방하다.

행동이 우선이다

- 결론적으로 **객체의 타입을 결정하는 것은 객체의 행동뿐**이다. 객체가 어떤 데이터를 보유하고 있는지는 아무런 영향도 미치지 않는다.
- 동일한 타입에 속한 객체는 내부의 데이터 표현 방식이 다르더라도 동일한 메시지를 수신하고 이를 처리할 수 있다. 다만, 내부의 데이터 표현 방식이 다르기 때문에 동일한 메시지를 처리하는 방법은 서로 다를 수밖에 없다. - **다형성**(동일한 요청에 대해 서로 다른 방식으로 응답할 수 있는 능력)
- 데이터의 내부 표현 방식과 무관하게 행동만이 고려 대상이라는 사실은 외부에 데이터를 감춰야 한다는 것을 의미한다. - **캡슐화**(외부에 행동만을 제공하고 데이터는 행동 뒤로 감춤)
- **책임-주도 설계(Responsibility-Driven Design)**
 1. 객체가 외부에 제공해야 하는 책임을 먼저 결정한다.
 2. 그 책임을 수행하는 데 적합한 데이터를 나중에 결정한 후, 데이터를 책임을 수행하는 데 필요한 외부 인터페이스 뒤로 캡슐화해야 한다.



0과 1로 이루어진 비트에 대해서 데이터 타입을 설명할 때는 이해가 어려웠다.

하지만 객체의 타입을 행동과 연관지어 설명하는 부분을 읽고나서 앞에서 쌓인 답답함이 내려갔다.

“객체의 타입을 결정하는 것은 객체의 행동뿐”이라는 말은 다형성과 캡슐화를 이해함에 있어서 큰 도움이 되었다.



타입의 계층

트럼프 계층

- 사실 정원사, 왕자와 공주, 하객, 하트 왕과 왕비는 ‘트럼프’가 아니라, 트럼프와 닮은 ‘트럼프 인간’이다.
트럼프는 납작하고 뒤집어질 수 있지만, 걸어다닐 수는 없기 때문이다.

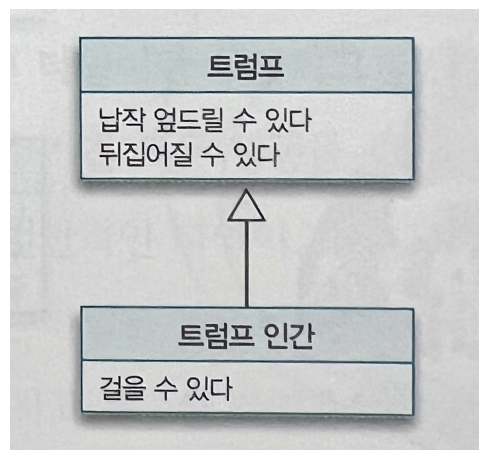
- 트럼프 인간은 트럼프의 모든 행동을 하고 있을 뿐만 아니라 더 특화된 행동을 하는 특수한 개념이며, 이러한 두 개념 사이의 관계를 **일반화/특수화 관계**라고 한다.

일반화/특수화 관계

- 두 타입 간에 일반화/특수화 관계가 성립하려면 한 타입이 다른 타입보다 더 특수하게 행동해야 하고, 반대로 한 타입은 다른 타입보다 더 일반적으로 행동해야 한다.
- 일반적인 타입은 특수한 타입보다 더 적은 수의 행동을 가지고, 특수한 타입은 일반적인 행동보다 더 많은 수의 행동을 가진다.
- 단, 특수한 타입은 일반적인 타입이 할 수 있는 모든 행동을 동일하게 수행할 수 있어야 한다.
- 주의! 내연을 의미하는 행동의 가짓수와 외연을 의미하는 집합의 크기는 서로 반대다.

슈퍼타입과 서브타입

- 더 일반적인 타입을 **슈퍼타입(Supertype)**, 더 특수적인 타입은 **서브타입(Subtype)**이라고 한다.
- 어떤 타입을 다른 타입의 서브타입이라고 말할 수 있으려면, 다른 타입을 대체할 수 있어야 한다.
- **일반화/특수화 관계 표기법**



- 슈퍼타입을 상단에, 서브타입을 하단에 위치시키고 속이 빈 삼각형으로 연결한다.
- 서브타입에서는 슈퍼타입과 중복된 행위를 생략할 수 있다.

일반화는 추상화를 위한 도구다

- 앨리스는 두 가지 추상화 기법을 사용하였다.

1. 정원에 있던 등장인물들의 차이점은 배제하고 공통점만을 강조함으로써 이들을 공통의 타입인 트럼프 인간으로 **분류**
 2. 트럼프 인간을 단순한 관점에서 바라보기 위해 불필요한 특성을 배제하고 좀 더 포괄적인 의미를 지닌 트럼프로 **일반화**
- 객체지향 패러다임을 통해 세상을 바라보는 거의 대부분의 경우에 **분류와 일반화/특수화 기법**을 동시에 적용하게 된다.

정적 모델

타입의 목적

- 타입을 사용하는 이유는 인간의 인지 능력으로는 시간에 따라 동적으로 변하는 객체의 복잡성을 극복하기가 어렵기 때문이다.
- 타입은 앨리스의 상태에 복잡성을 부과하는 시간이라는 요소를 제거함으로써 시간에 독립적인 정적인 모습으로 앨리스를 생각할 수 있게 해준다.

그래서 결국 타입은 추상화다

- 앨리스가 어떻게 변할 수 있는지 그 다양한 가능성을 고려할때는 구체적으로 키가 얼마인가보다는 단순히 키가 변할 수 있다는 가능성에 집중하는 것이 더 간단한다.
- 타입을 이용하면 객체의 동적인 특성을 추상화할 수 있다.
- 결국 타입은 시간에 따른 객체의 상태 변경이라는 복잡성을 단순화할 수 있는 효과적인 방법이다.

동적 모델과 정적 모델

- **스냅샷**: 객체가 특정 시점에 구체적으로 가지는 어떤 상태
- **객체 다이어그램**: UML에서 의미하는 스냅샷
- **동적 모델**: 스냅샷처럼 객체가 살아 움직이는 동안 상태가 어떻게 변하고 어떻게 행동하는지 포착하는 것
- **정적 모델(타입 모델)**: 객체가 가질 수 있는 모든 상태와 행동을 시간에 독립적으로 표현하는 것
- 객체지향 애플리케이션을 설계하고 구현하기 위해서는 객체 관점의 동적 모델과 객체를 추상화한 타입 관점의 정적 모델을 적절히 혼용해야 한다.
- 객체지향 프로그래밍 언어를 이용해 클래스를 작성하는 시점 → **정적인 관점에서 접근**

- 애플리케이션을 실행해 객체의 상태 변경을 추적하고 디버깅하는 시점 → **동적인 관점에서 접근**

클래스

클래스 ≠ 타입

- 객체지향 프로그래밍 언어에서 정적인 모델, 즉 '**타입을 구현**'하는 가장 보편적인 방법은 클래스를 이용하는 것이다.
- 타입은 객체를 분류하기 위해 사용하는 개념인 반면, 클래스는 단지 타입을 구현할 수 있는 여러 매커니즘 중 하나일 뿐이다.