Orientación a Objetos

Elaborado por:
Gary Briceño

Contenido:

1. Paradigmas de programación
2. Programación Orientada a Objetos
3. Clases vs Objetos
4. Clases: Estado y Comportamiento
5. Paradigma de OO:
    - Abstraccion
    - Herencia
    - Polimorfismo
    - Encapsulamiento
6. Herencia Multiple: Duck Typing

Paradigma de Programacion

Los paradigmas de programación son **modelos** para **resolver problemas** comunes con nuestro código. Son caminos, guías, reglas, teorías y fundamentos que agilizan nuestro desarrollo y evitan que reinventemos la rueda.

Paradigmas populares:

- Estructurada
- Orientada a Objetos
- Funcional

Referencia:

- https://platzi.com/blog/paradigmas-programacion/

Programación Orientada a Objetos

- Escribir código basado en objetos.
- Es una técnica para describir las acciones de sistemas complejos.
- Describe un sistema como la interacción de múltiples objetos.

Que es un Objeto?

Conjunto de datos y un comportamiento asociado.

Conjunto de atributos y métodos.

Clases y Objetos

- La clase es el template que se utiliza para crear el objeto.
- El objeto es único, y se debe poder identificar como único.
- Un objeto es una instancia de la clase.

Diagramas - UML

- Los diagramas son ayudas para entender un problema.
- Uno debe diagramar hasta poder entender el problema.
- Sirve como un lenguaje común.

Diagrama de Clases: Lenguaje Común

Atributos

- Los objetos tienen un estado, los atributos describen el estado de los objetos.
- Los atributos describen la característica de un objeto, el estado actual.
- Los atributos deben ser tomados basados en el contexto del problema que se esta solucionando.

Diagrama de Clases: Atributos

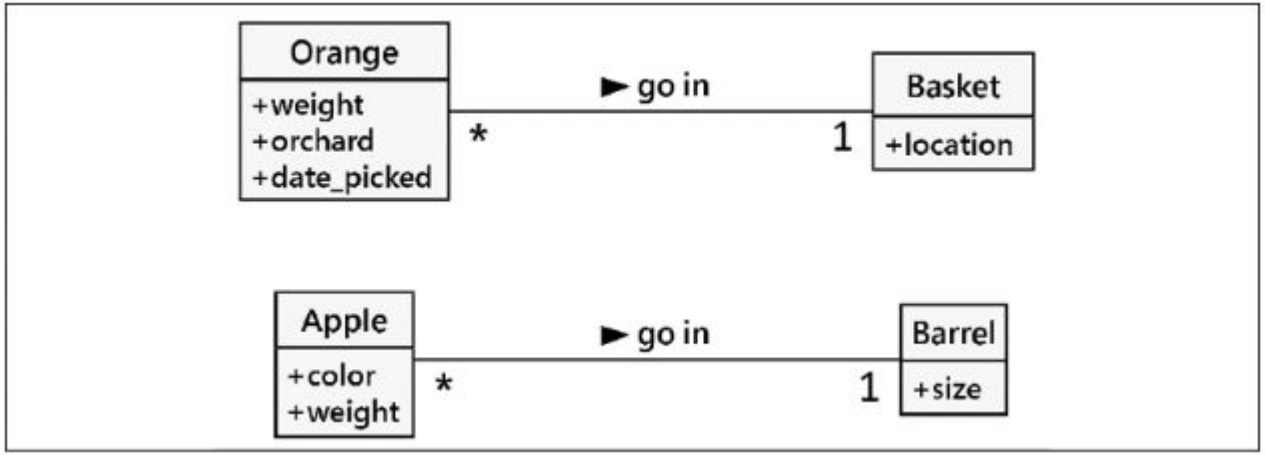
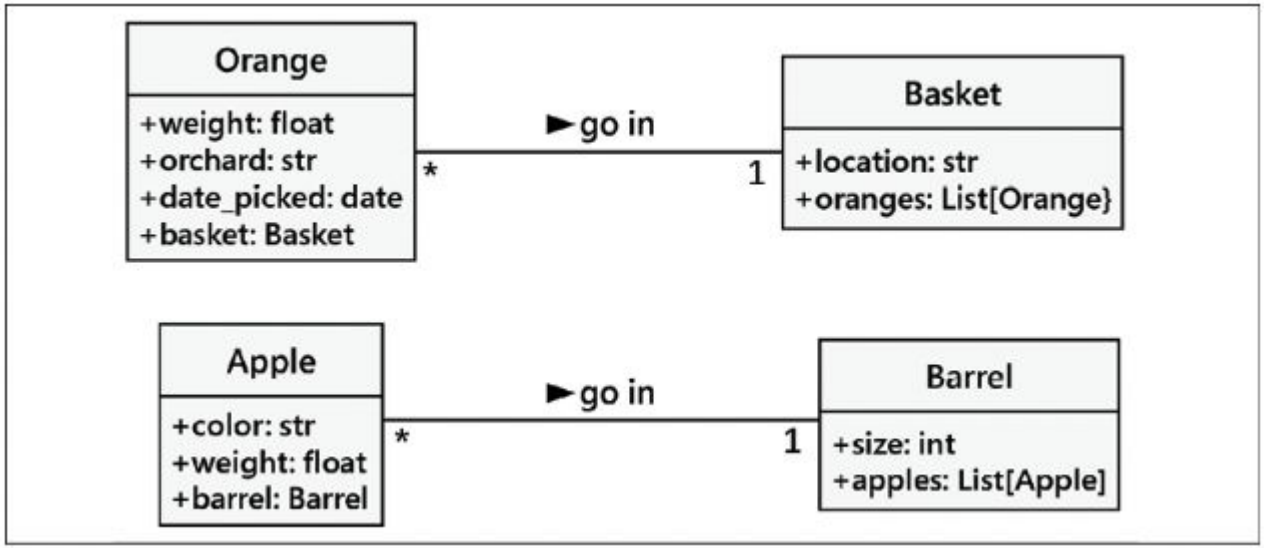


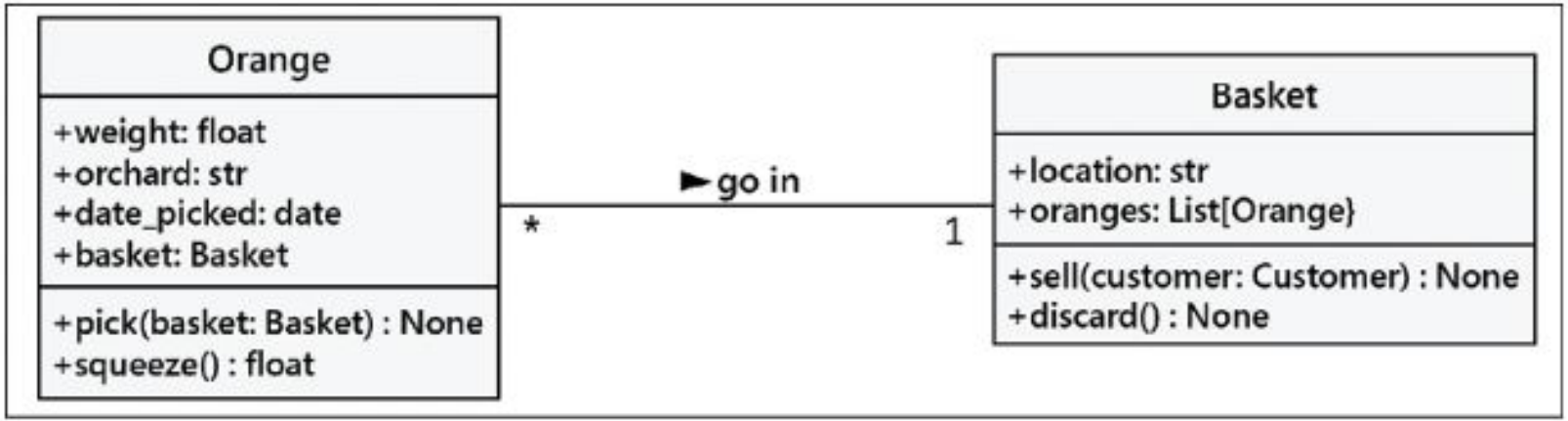
Diagrama de Clases: Atributos



Comportamiento

- El comportamiento son acciones que se pueden realizar.
- El comportamiento se describe mediante métodos o funciones.
- Los métodos pueden acceder a los atributos y también recibir parámetros y retornar valores.

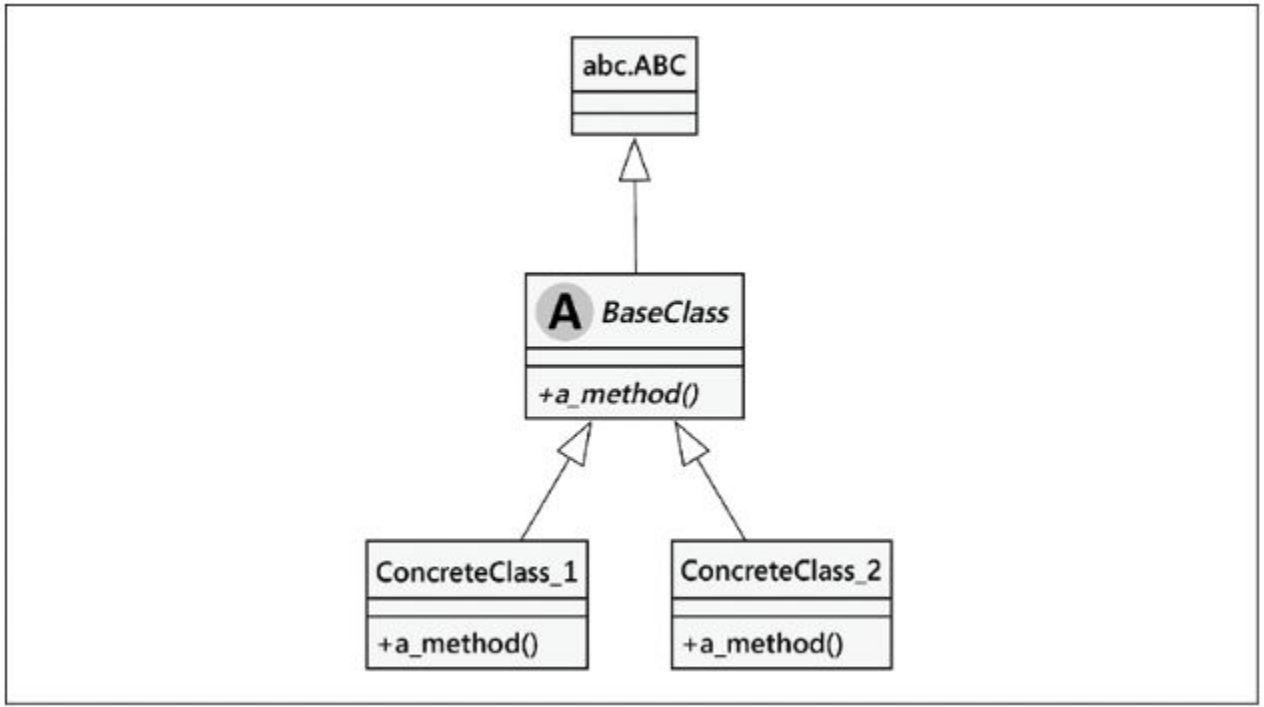
Diagrama de Clases: Comportamiento



Abstraccion

- Los objetos en los programas, representan objetos reales, pero esto no los convierte en objetos reales.
- El modelo es una abstracción de un concepto real.
- Abstracción ayuda a lidiar con el nivel de detalle que se requiere.

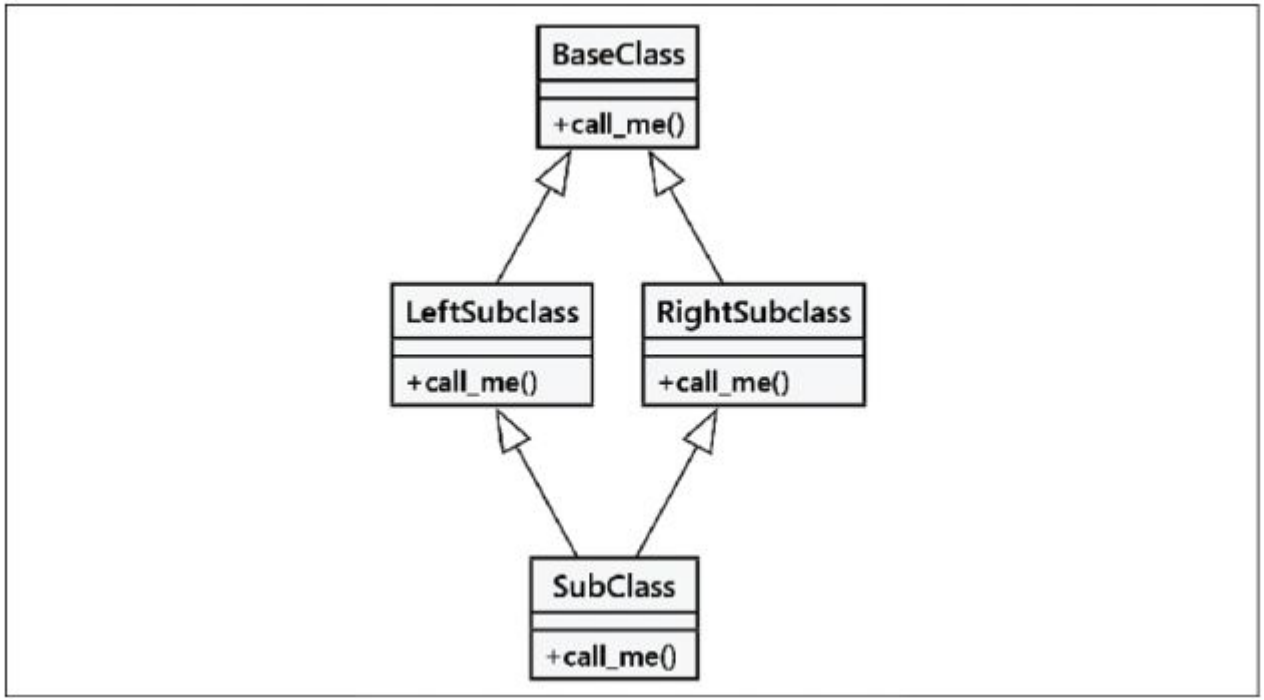
Clases Abstractas



Herencia

- Una clase obtiene todos los atributos y el comportamiento de otra clase.
- Una clase hija hereda de la clase padre.
- Sirve para crear clases nuevas a partir de clases preexistentes.
- En Python, por defecto todos heredan de la clase object.

Problemas de la Herencia Múltiple



Duck Typing

- Es usado en lenguajes de tipado dinámico
- Donde el tipo del objeto no es tan importante, como el método en sí.
- No se verifica el tipo, por el contrario se verifica la presencia del método.

Referencia:

- https://www.geeksforgeeks.org/duck-typing-in-python/

Interfaces y Encapsulamiento

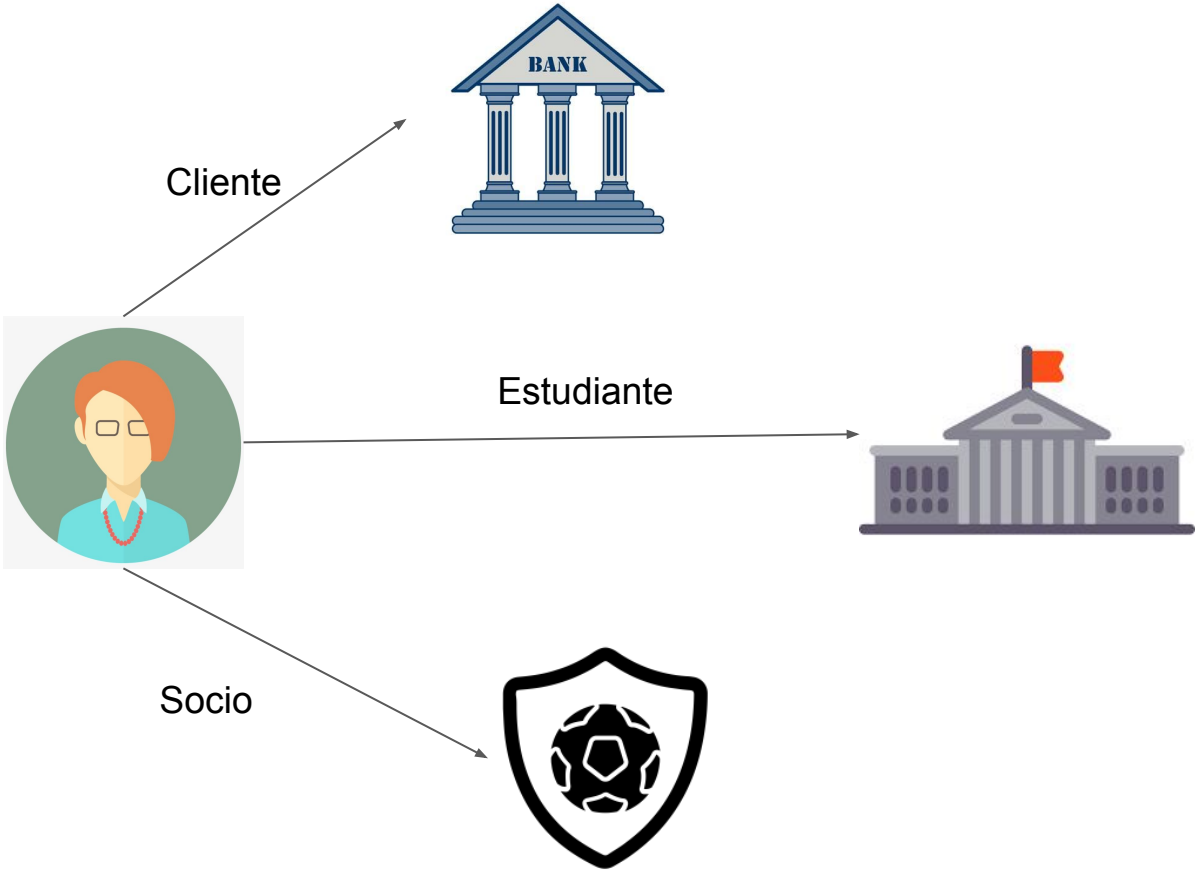
- Ejemplo del TV y el control remoto, de cómo el control remoto es una interfaz para simplificar la interacción con el TV o ENCAPSULAR la complejidad.
- El poder ocultar la complejidad del comportamiento, es denominado Encapsulamiento.

Polimorfismo:

- Poli: muchos, morfo: formas
- Los objetos pueden tener diferente comportamiento, según como sean accedidos.
- Gracias al duck typing, no se requiere una interfaz, solo que tengan el método que se quiere llamar.

Referencia:

- https://ellibrodepython.com/polimorfismo-en-programacion


BANK
Cliente
Estudiante
Socio

Polimorfismo: