PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada
1^{er} semestre 2016

Actividad 07

Metaclases

Instrucciones

Las metaclases son útiles para interceptar la creación de clases, y manejarla según sea necesario.

Patricio, que trabaja para el benevolente Bucchi, es programador en lenguaje Java. En Java no existe la herencia múltiple y eso complica a Patricio, que no está acostumbrado a Python. El benevolente Bucchi ha ideado una forma de hacerle la adopción de Python más sencilla: **Seguirá habiendo herencia múltiple**, pero con una limitación: **a lo más una superclase podrá ser una clase normal, las demás deberán ser abstractas**.

Para hacerlo, decides crear una nueva **metaclase** para las clases abstractas `class Abstract(type)`. La idea es que se cumpla lo siguiente:

- Ninguna clase debe heredar de más de una no abstracta en forma directa.
- Una clase no abstracta debe sobreescribir todos los métodos que se definieron en cada clase abstracta de la que hereda.
- Una clase abstracta puede heredar de otras abstractas.
- Una clase que no hereda de una concreta (sin contar **object**) sigue siendo abstracta.

Requerimientos

- Crear la metaclase **Abstract**
- Dentro de la metaclase crear un método `def is_abstract(cls)` que detecte si una clase es **Abstract** o no.
- Dentro de la metaclase crear un método `def check_base(cls, base)` que dada una clase base, detecte si la clase implementa todos los métodos de esa clase base.

- No permitir la creación de objetos de la clase si es **Abstract**, lanzando un error.
- Al crear una clase no **Abstract** que hereda de una clase abstracta sin sobreescribir algún método de ésta, lanzar un error.
- Al final debes probar el funcionamiento de tu metaclase con un ejemplo que debería funcionar y con un ejemplo que no debería funcionar.

Notas

- **Importante:** para detener la ejecución de un programa debido a que no se cumple algún requerimiento debe usar la línea `raise Exception("message")`
- Función sin cuerpo hace referencia a una función que solo tiene un pass: `def ejemplo(): pass`.

To - DO

- Crear metaclase (0.1 pts)
- `def is_abstract(cls)` (1.5 pts)
- `def check_base(cls, base)` (1.5 pts)
- No permitir la creación de objetos de la clase si es abstracta (1.0 pts)
- Lanzar error en caso de que una clase no sobreescriba métodos (1.5 pts)
- Ejemplos (0.4 pts)

Tips

- El método `callable(func)` le puede servir para identificar si una variable corresponde a un método de instancia.
- El método `Clase.mro()` retorna una tupla con las clases de las que hereda Clase.
- Puede acceder a los atributos y métodos de un objeto usando la siguiente línea `obj.__dict__`. Esto se aplica también a las clases.

Entrega

- **Lugar:** GIT - Carpeta: Actividades/AC07
- **Hora:** 16:55