Принципы объектно-ориентированного проектирования в Python

Михаил Гусаров
dottedmag@dottedmag.net

RuPyRu 2007.
Омск, 10 февраля 2007

Статическая и динамическая типизация

Статически типизированные языки

- Явно типизированные

```
Foo* f = new Foo();
```

- С выведением типов

```
-- foo :: Integer -> Tree
foo i = Leaf i
```

Динамически типизированные языки

```
var bar = foo();
id foobar = [[class alloc] init];
```

Open-Closed Principle: определение

Software entities (classes, modules, functions etc)
should be open for extension, but closed for modification.

Robert Martin, *"The Open-Closed Principle"*, Engineering Notebook, The C++ Report.

Применение OCP

В статически типизированных языках

- Расширение через подклассы и реализацию интерфейсов
- “Бедный” синтаксис и семантика.

В динамически типизированных языках

- Расширение через новые реализации контрактов
- Нет проверки контрактов во время компиляции

Liskov Substitution Principle

> *If for each object O_1 of type S there is object O_2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when O_1 is substituted for O_2, then S is a subtype of T.*

Barbara Liskow, *"Family Values: A Behavioral Notion of Subtyping"*.

> *Function that uses pointers or references to base classes must be able to use objects of derived classes without knowing it.*

Robert Martin, *"Liskov Substitution Principle"*, Engineering Notebook, The C++ Report.

Liskov Substitution Principle - обобщение

Code must be able to use objects of subtypes instead of objects of supertypes without knowing it.

Проверка контрактов в динамически типизированных языках

Юнит-тестирование

- Проверка контрактов во время тестирования
- Документация: не только синтаксис, но и поведение