Питон
Курс лекций
Лекция двенадцатая

© Зайцев Вадим Валерьевич, 2002–2010,
spider.vz@gmail.com

6.3 Модульность

Во многих современных объектно-ориентированных языках программирования (и питон — не исключение) классы не являются единственным возможным механизмом декомпозации. Классы и объекты составляют логическую структуру системы, они классифицируются и размещаются по модулям, которые образуют физическую структуру системы. Каждый модуль представляет собой почти замкнутую подсистемы: классы внутри модуля могут иметь тесную взаимосвязь, тогда как внешние связи слабы или же вовсе отсутствуют.

Модульность — это свойство системы, которая была разложена на внутренне связные, но слабо связанные между собою модули.

Модульность позволяет программисту группировать абстракции и хранить их сообразно тому, как они связаны между собой. Правильное разбиение программы на модули — задача, не уступающая по сложности таким проблемам OOD, как выбор набора абстракций или отделение реализации объекта от интерфейса.

На макроуровне модулей (*макро* относительно уровня классов) также используется деление на интерфейс и реализацию. Несмотря на то, что в одних языках программирования это деление более явно и вписано в синтаксис, а в других почти нивелируется, такое разделение является одной из немногих испытанных технологий проектирования, поэтому широко используется программистами и документаторами. Интерфейс модуля иногда называют спецификацией, придавая ему таким образом чисто декларативный смысл. Это даёт моральное право называть реализацию телом модуля.

Очевидно, что задача группировки классов по модулям имеет два крайних решения:

1. Помещение всех классов в один модуль.

   Метод пригоден только для небольших или малых задач, в скольнибудь серьёзных проектах он может вызывать осложнения как в до-

кументировании, так и в поддержании системы с течением времени. Зачастую же этот метод используют при начальном проектировании части большой системы как отдельного приложения.

2. Помещение каждого класса в свой модуль.

   Вполне подходящий метод для систмы из нескольких очень больших классов, чрезвычайно мало связанных между собой и большей частью используемых по раздельности. Например, стандартный питоновский модуль `cmd` (используемый для написания интерпретаторов команд) содержит только класс `Cmd`, выполняющий все необходимые операции. Но размещение в нескольких тысячах файлов мелких одиночных объектов или вообще констант (которые в чисто объектно-ориентированных языках также являются объектами) — это дурная привычка, не приводящая ни к чему, кроме долгой перетрансляции всей системы при малейших изменениях и сизифовому труду документаторов.

Согласно Гради Бучу, «деление программы на модули бессистемным образом гораздо хуже, чем отсутствие модульности вообще». И во многих случаях это действительно так. Число всевозможных распределений N классов по модулям равно 2^N, а сколько из них разумных? На этот вопрос даже дискретная математика ответа не даёт.