Riza Satria Perdana, S.T., M.T.

Teknik Informatika - STEI ITB

Reflection

Pendahuluan

Pemrograman Berorientasi Objek

Overview

- Mengenal Implementasi bytecode Java
- Reflection API
- Kegunaan Reflection API
- Memanfaatkan Reflection API untuk sistem plugin

Implementasi bytecode Java

- Semua program Java akan dikompilasi menjadi *bytecode* sebelum dieksekusi
  - ✓ Bytecode adalah bahasa mesin untuk JVM (Java Virtual Machine)
  - ✓ Umumnya bytecode ini akan disimpan dalam filesystem dalam bentuk file .class
- Tidak ada kewajiban bahwa bytecode yang akan dieksekusi harus berada dalam file (bisa dari sumber lain)

Class Loader

- Class Loader adalah bagian dalam JVM yang bertugas meload kelas ke dalam memori
- Defaultnya (dengan Class Loader bawaan Java) kelas dapat diload dari:
  - ✓ filesystem (yang paling umum),
  - ✓ file jar (jika file kelas dikompresi),
  - ✓ jaringan (misalnya dalam kasus applet)

```
import java.net.URL;
import java.net.URLClassLoader;

public class NetworkLoaderExample {
    public static void main(String[] args) throws Exception {
        URL url = new URL("http://example.com/classes/");
        URL[] urls = new URL[]{url};

        ClassLoader loader = new URLClassLoader(urls);
        Class<?> clazz = loader.loadClass("com.example.MyRemoteClass");

        Object obj = clazz.getDeclaredConstructor().newInstance();
        System.out.println("Loaded class: " + obj.getClass().getName());
    }
}
```

Custom Class Loader

- Programmer dapat membuat Class Loader sendiri yang bisa meload kelas dari mana saja (jaringan, file, dll) bahkan menciptakan kelas 'on the fly' (saat runtime)
  - Class Loader ini disebut dengan Custom Class Loader
- Custom Class Loader dibuat dengan menurunkan kelas abstrak java.lang.ClassLoader
  - Contoh pembuatan ClassLoader ada di dokumentasi API Java (Lihat dokumentasi API kelas ClassLoader)
  - Umumnya programmer tidak perlu membuat class Loader sendiri

Reflection API

- Class Loader bisa menciptakan kelas baru pada saat runtime, namun kelas baru tersebut mungkin tidak bisa digunakan karena programmer tidak mengetahui apa saja method dan property yang ada pada kelas baru (jika kelas tersebut tidak diciptakan programmer yang sama)
- Untuk bisa melihat dan mengeksekusi isi sebuah kelas baik yang dikenal maupun tak dikenal saat kompilasi, Java memiliki Reflection API

Terima Kasih