Service locator pattern

The **service locator pattern** is a design pattern used in software development to encapsulate the processes involved in obtaining a service with a strong abstraction layer. This pattern uses a central registry known as the "service locator", which on request returns the information necessary to perform a certain task.[1]

1 Advantages

- The "service locator" can act as a simple run-time linker. This allows code to be added at run-time without re-compiling the application, and in some cases without having to even restart it.
- Applications can optimize themselves at run-time by selectively adding and removing items from the service locator. For example, an application can detect that it has a better library for reading JPG images available than the default one, and alter the registry accordingly.
- Large sections of a library or application can be completely separated. The only link between them becomes the registry.

2 Disadvantages

- Things placed in the registry are effectively black boxes with regards to the rest of the system. This makes it harder to detect and recover from their errors, and may make the system as a whole less reliable.
- The registry must be unique, which can make it a bottleneck for concurrent applications.
- The registry can be a serious security vulnerability, because it allows outsiders to inject code into an application.
- The registry hides the class' dependencies, causing run-time errors instead of compile-time errors when dependencies are missing.
- The registry makes the code more difficult to maintain (opposed to using Dependency injection), because it becomes unclear when you would be introducing a breaking change.
- The registry makes code harder to test, since all tests need to interact with the same global service locator class to set the fake dependencies of a class under test.

3 See also

- Dependency injection
- Dependency inversion principle
- Java Naming and Directory Interface

4 References

[1] http://martinfowler.com/articles/injection.html#UsingAServiceLocator

5 External links

- Sample code
- Game Programming Patterns: Service Locator
- Dependencies In Disguise

6 Text and image sources, contributors, and licenses

6.1 Text

- **Service locator pattern** *Source:* https://en.wikipedia.org/wiki/Service_locator_pattern?oldid=682974712 *Contributors:* Kku, Furrykef, Beland, Andreas Kaufmann, Alaibot, ThomasO1989, Derekgreer, Andy Dingley, KathrynLybarger, Rosiestep, GaryChamberlain, Lsoares, Kmahesh74, Iudaeus, Paullik.paul, Tim-bezhashvyly and Anonymous: 11

6.2 Images

- **File:System-installer.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/db/System-installer.svg *License:* Public domain *Contributors:* The Tango! Desktop Project *Original artist:* The people from the Tango! project

6.3 Content license

- Creative Commons Attribution-Share Alike 3.0