Objenesis
- About
- License
- Download

Documentation
- Twenty Second Tutorial
- The Details
- Embedding

Project Details
- Getting Support
- Acknowledgements
- Source Repository
- Release notes
- Who is using Objenesis

About

Objenesis is a small Java library that serves one purpose:

- **To instantiate a new object of a particular class.**

When would you want this?

Java already supports this dynamic instantiation of classes using `Class.newInstance()`. However, this only works if the class has an appropriate constructor. There are many times when a class cannot be instantiated this way, such as when the class contains:

- Constructors that require arguments.
- Constructors that have side effects.
- Constructors that throw exceptions.

As a result, it is common to see restrictions in libraries stating that classes must require a default constructor. Objenesis aims to overcome these restrictions by bypassing the constructor on object instantiation.

Typical uses

Needing to instantiate an object without calling the constructor is a fairly specialized task, however there are certain cases when this is useful:

- **Serialization, Remoting and Persistence** - Objects need to be instantiated and restored to a specific state, without invoking code.
- **Proxies, AOP Libraries and Mock Objects** - Classes can be subclassed without needing to worry about the super() constructor.
- **Container Frameworks** - Objects can be dynamically instantiated in non-standard ways.

Getting Started

1. Download Objenesis.
2. Read the twenty second tutorial.

How it Works

Objenesis uses a variety of approaches to attempt to instantiate the object, depending on the type of object, JVM version, JVM vendor and SecurityManager present. These are described in full in the detailed documentation.

Avoiding the Dependency

For something as straight forward as instantiating an object, it can be a pain introducing yet another library dependency into your project. Objenesis is easy to embed in your existing library so your end users don't even know it's there.

Supported JVMs

The list of tested JVMs is available here. Other JVMs might be supported be haven't been tested yet. You can test your environment using the TCK and we would be glad to have feedback on any supported / unsupported JVM.

Benchmarking

We do more and more benchmarking to make sure Objenesis has great performance but also to settled the best way to use it for multiple use-cases. Results are added here. Please, share your own results. Especially, if they differ from ours.