Meta-JB

Table of contents

A meta-programming approach to general data modeling.

1. Introduction

Meta-JB is a MetaClass/MetaObject layer providing generic access to model implementations, decoupling application logic from underlying implementation details, and allowing user interfaces (Swing, HTML, etc.) to be dynamically generated at runtime. By wrapping model implementations in MetaObject adapters, applications can interact with the model layer in a homogenous way.

2. Description

Meta-JB extends the Java Beans-based meta-programming concept to provide more generic access to object attributes and descriptions for any model object with an appropriate adapter. The descriptions of a class's properties (the MetaClass) and access to an object's attributes are decoupled from actual implementations by adapters implementing a Map-like name/value interface (the MetaObject). Because the thin framework is built on generic interfaces, it is not tied directly to real Java bean implementations and can also be used for anything that can access values by name. (Some examples are SQL result sets, HTTP request data, or simple hash maps.) Once a "class" has been described, the information can even be applied to different underlying implementations.

The MetaClass/MetaObject layer is a foundation for dynamically generating user-level access to application object models. Toolkits are provided for generating Swing GUIs at runtime or dynamically rendering objects as XML using the class descriptions. On the drawing board is support for generating HTML forms and views as well. Future development may also extend to a collaborative data access layer.

For more information see the <u>overview</u>, AKA "What is this thing?"

3. Who is this for?

Technically, this is for any developer doing work with data models. More practically, this toolkit will resonate with those tired of the tedium of the standard property boilerplate... from the database on up through to the user interface. If you feel like you've been writing the same four or five lines of code for every bean property then Meta-JB may provide a way out.

4. History

This project is based on personal research over the past three years, and is the third iteration of the framework. It originally started as a way to get rid of the tedium of GUI development. Many projects involve the same repetitive sorts of screens and at times several views of the same object. Deciding to retitle a field or similar was painful at best.

Inspired by the introspection/meta-programming capabilities of the Java beans architecture, it seemed reasonable to generate portions of the UI from property descriptors. Over time, this evolved into a framework that could be used with or without standard Java beans and that could serve to create entire object "forms" and, in some cases, entire screens. The Java Beans pedigree is what gives Meta-JB its "JB" but it is now much less relevant.