



## GEMObjectSpan

Manage your objects – across systems and across versions.

The GEMObjectSpan service instantly converts any Java objects to and from XML files. Because it allows the **exchange of objects** between heterogeneous systems in an XML format that can be easily customized, it represents a key building block for business-to-business e-commerce.

If you are developing enterprise Java applications that will require maintenance releases and upgrades, then you also need to deal with the issue of **migration across versions** of your classes while preserving critical business data. The GEMObjectSpan service provides a simple, powerful, and general approach to migration.

### The Technology

GEMObjectSpan is a set of Java classes with a straightforward API for transforming the structure and public data of an object or collection of objects to XML – in as little as a single line of code and without any change to the object! The XML file can then be viewed in a browser that supports XML, updated using any editor, stored for later use, or transferred to a trading partner.

If a trading partner uses different names to represent business object classes or data types, GEMObjectSpan allows easy specification of aliases dynamically at the time the object is externalized to XML or recreated from XML.

The XML file can be read through the GEMObjectSpan services and used to recreate objects and their state. If an object definition has changed – no problem. GEMObjectSpan accounts for all typical types of changes, such as added fields, removed fields, changed field types, and obsolete classes. A well-defined migration approach and API converts all available information into a convenient object format, which can then be submitted to a migration method to complete the migration process.

### Application: Business-to-Business e-Commerce

Because GEMObjectSpan immediately creates an XML representation of an object's business data, it is an ideal solution for business-to-business transactions. It allows the developer to create the best objects to implement business rules and then convert them effortlessly to XML for business-to-business data transactions.

```
<?xml version="1.0" ?>
- <Object_1 NAME="value" CLASS="Object" TYPE="array" DIM="1" LENGTH="2" ID="1">
- <Company INDEX="0" CLASS="Company" TYPE="object" ID="2">
  <String NAME="name" CLASS="String" TYPE="object" LENGTH="32">Global Enterprise Managers, Inc.</String>
- <BusinessUnit_1 NAME="units" CLASS="BusinessUnit" TYPE="array" DIM="1" LENGTH="2" ID="3">
  - <BusinessUnit INDEX="0" CLASS="BusinessUnit" TYPE="object" ID="4">
    <int NAME="number" CLASS="int" TYPE="primitive">1</int>
    <String NAME="name" CLASS="String" TYPE="object" LENGTH="26">Power Distribution Systems</String>
    <String NAME="street" CLASS="String" TYPE="object" LENGTH="18">1434 Malcolm Drive</String>
    <String NAME="city" CLASS="String" TYPE="object" LENGTH="7">Dresher</String>
    :
    :
    :
```

Flexible, user-defined aliases for data types and element tag attributes can be specified at externalization or internalization time. In addition to making the XML representation highly readable, this is very important

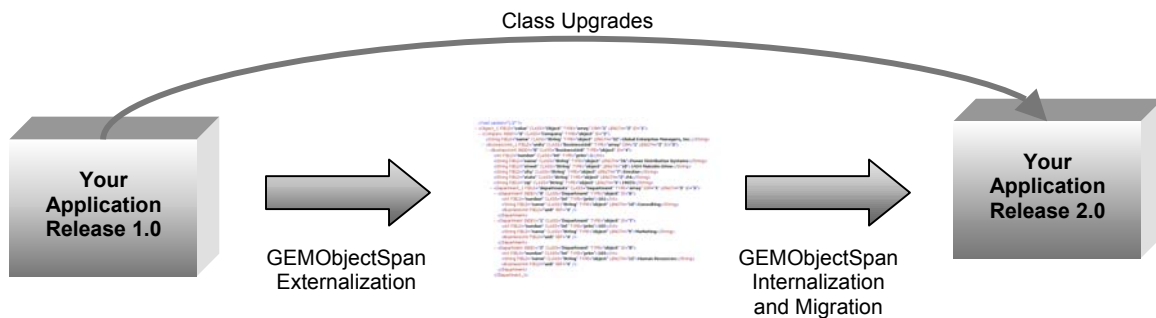


for e-commerce between partners that use differing technologies (e.g., Java, C++, or .NET) to host their representations of the business objects.

### **Application: Object Migration**

Although existing approaches, such as Java serialization based on the Serializable interface, can be used for externalizing, transporting, and internalizing objects within a homogeneous Java environment, product developers face significant challenges over time. This is because it may not be possible to reconstitute previously externalized instances after changes are made to the class. Some approaches to migration can require awkward class renaming and ad hoc procedures for dealing with new, removed, or changed fields.

Ship GEMObjectSpan as a service within your application and be ready for a smooth migration between system releases. GEMObjectSpan uses flexible XML parsing technology and a structured migration paradigm to facilitate migration between class versions. For every internalized object, a special object is created containing the class name, version, and any fields that could not be stored in the internalized object. After all fields are processed, a migration method (if one has been provided) is invoked on the internalized object. This method can perform any transformations and mappings needed to complete the migration.



### **Application: HTTP Object Transport Between Heterogeneous Object Systems**

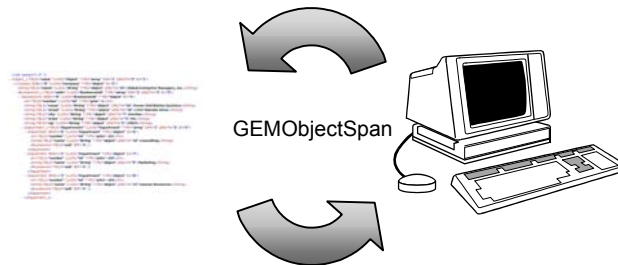
In addition to using files, GEMObjectSpan can externalize and internalize objects directly to and from a stream. This creates the opportunity for object transport between web servers and clients that is very simple and that can easily be extended for use in a mixed language environment – for example between Java server side objects and VisualBasic components on the client. It also allows extension of the powerful servlet programming approach to transfer full, serialized objects rather than only simple parameter pairs or other typical posted data.





## **Application: Instant Object Persistency**

Making objects persistent in a relational or object database is not the most practical or desirable approach for some applications, such as those placed directly on a standalone client PC. GEMObjectSpan can be used to create an instant persistent representation of objects on a local or shared file system. The value of this lightweight persistency mechanism is further enhanced by GEMObjectSpan's easy migration capability.



## **Platforms**

Because GEMObjectSpan is completely Java-based, it is portable across virtually every platform that supports Java. It has been deployed and tested on the following platforms:

- JDK 1.1.1, 1.1.7B, 1.1.8, 1.2.2, 1.4.1
- Windows NT/2000/XP, Solaris, Unix, Linux operating systems

## **The GEMObjectSpan Package**

The package includes a full API documentation set and description of the XML file format and options. An extensive set of example and test classes, including their source code, is provided.

A WrapperInterface is included, which can be used to externalize classes that do not have public fields.

The xerces XML parsing package is also included, although other Simple API for XML (SAX) compliant parsers can be used.



**Global Enterprise Managers, Inc.**  
[www.global-enterprise-mgrs.com](http://www.global-enterprise-mgrs.com)