Google code

Google App Engine

Defining Data Classes with JDO

You can use JDO to store plain Java data objects (sometimes referred to as "Plain Old Java Objects" or "POJOs") in the datastore. Each object that is made persistent with the PersistenceManager becomes an entity in the datastore. You use annotations to tell JDO how to store and recreate instances of your data classes.

Note: Earlier versions of JDO use `.jdo` XML files instead of Java annotations. These still work with JDO 2.3. This documentation only covers using Java annotations with data classes.

Class and Field Annotations

Each object saved by JDO becomes an entity in the App Engine datastore. The entity's kind is derived from the simple name of the class (inner classes use the `$` path without the package name). Each persistent field of the class represents a property of the entity, with the name of the property equal to the name of the field (with case preserved).

To declare a Java class as capable of being stored and retrieved from the datastore with JDO, give the class a `@PersistenceCapable` annotation. For example:

```
import javax.jdo.annotations.PersistenceCapable;

@PersistenceCapable
public class Employee {
    // ...
}
```

Fields of the data class that are to be stored in the datastore must be declared as persistent fields. To declare a field as persistent, give it a `@Persistent` annotation:

```
import java.util.Date;
import javax.jdo.annotations.Persistent;

// ...
    @Persistent
    private Date hireDate;
```

To declare a field as not persistent (it does not get stored in the datastore, and is not restored when the object is retrieved), give it a `@NotPersistent` annotation.

Tip: JDO specifies that fields of certain types are persistent by default if neither the `@Persistent` nor `@NotPersistent` annotations are specified, and fields of all other types are not persistent by default. See the DataNucleus documentation for a complete description of this behavior. Because not all of the App Engine datastore core value types are persistent by default according to the JDO specification, we recommend explicitly annotating fields as `@Persistent` or `@NotPersistent` to make it clear.

The type of a field can be any of the following. These are described in detail below.

- one of the core types supported by the datastore
- a Collection (such as a `java.util.List<...>`) or an array of values of a core datastore type
- an instance or Collection of instances of a `@PersistenceCapable` class
- an instance or Collection of instances of a Serializable class
- an embedded class, stored as properties on the entity

A data class must have one field dedicated to storing the primary key of the corresponding datastore entity. You can