# Collection Mapping in Hibernate

We can map collection elements of Persistent class in Hibernate. You need to declare the type of collection in Persistent class from one of the following types:

- java.util.List
- java.util.Set
- java.util.SortedSet
- java.util.Map
- java.util.SortedMap
- java.util.Collection
- or write the implementation of org.hibernate.usertype.UserCollectionType

The persistent class should be defined like this for collection element.

```
package com.javatpoint;

import java.util.List;

public class Question {
private int id;
private String qname;
private List<String> answers;//List can be of any type

//getters and setters

}
```

## Mapping collection in mapping file

There are many subelements of **<class>** elements to map the collection. They are **<list>**, **<bag>**, **<set>** and **<map>**. Let's see how we implement the list for the above class:

```
<class name="com.javatpoint.Question" table="q100">
        <id name="id">
```

```
          <generator class="increment"></generator>
          </id>
          <property name="qname"></property>

          <list name="answers" table="ans100">
          <key column="qid"></key>
          <index column="type"></index>
          <element column="answer" type="string"></element>
          </list>

          </class>
```

There are three subelements used in the list:

- **<key>** element is used to define the foreign key in this table based on the Question class identifier.
- **<index>** element is used to identify the type. List and Map are indexed collection.
- **<element>** is used to define the element of the collection.

This is the mapping of collection if collection stores string objects. But if collection stores entity reference (another class objects), we need to define **<one-to-many>** or **<many-to-many>** element. Now the Persistent class will look like:

```
package com.javatpoint;

import java.util.List;

public class Question {
private int id;
private String qname;
private List<Answer> answers;//Here, List stores the objects of Answer class

//getters and setters

}
```

```
package com.javatpoint;
```

```
import java.util.List;
public class Answer {
private int id;
private String answer;
private String posterName;
//getters and setters
}
```

Now the mapping file will be:

```
<class name="com.javatpoint.Question" table="q100">
          <id name="id">
          <generator class="increment"></generator>
          </id>
          <property name="qname"></property>

          <list name="answers" >
          <key column="qid"></key>
          <index column="type"></index>
          <one-to-many class="com.javatpoint.Answer" />
          </list>

          </class>
```

Here, List is mapped by one-to-many relation. In this scenario, there can be many answers for one question.

## Understanding key element

The key element is used to define the foreign key in the joined table based on the original identity. The foreign key element is nullable by default. So for non-nullable foreign key, we need to specify not-null attribute such as:

```
<key column="qid" not-null="true" ></key>
```

The attributes of the key element are column, on-delete, property-ref, not-null, update and unique.

```
<key
column="columnname"
on-delete="noaction|cascade"
not-null="true|false"
property-ref="propertyName"
update="true|false"
unique="true|false"
/>
```

## Indexed collections

The collection elements can be categorized in two forms:

- **indexed** ,and
- **non-indexed**

The List and Map collection are indexed whereas set and bag collections are non-indexed. Here, indexed collection means List and Map requires an additional element **<index>**.

## Collection Elements

The collection elements can have value or entity reference (another class object). We can use one of the 4 elements

- **element**
- **component-element**
- **one-to-many**, or
- **many-to-many**

The element and component-element are used for normal value such as string, int etc. whereas one-to-many and many-to-many are used to map entity reference.

## Upcoming topics in Collection Mapping

Mapping List

In this example, we are going to map the List element.

Mapping Bag

In this example, we are going to use the bag collection of Hibernate framework.

Mapping Set

Here, we will map the set element of collection.

← prev

next →