Overview of the Criteria and Metamodel APIs

Similar to JPQL, the Criteria API is based on the abstract schema of persistent entities, their relationships, and embedded objects. The Criteria API operates on this abstract schema to allow developers to find, modify, and delete persistent entities by invoking Java Persistence API entity operations. The Metamodel API works in concert with the Criteria API to model persistent entity classes for Criteria queries.

The Criteria API and JPQL are closely related and are designed to allow similar operations in their queries. Developers familiar with JPQL syntax will find equivalent object-level operations in the Criteria API.

The following simple Criteria query returns all instances of the `Pet` entity in the data source:

```
EntityManager em = ...;
CriteriaBuilder cb = em.getCriteriaBuilder();
CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);
Root<Pet> pet = cq.from(Pet.class);
cq.select(pet);
TypedQuery<Pet> q = em.createQuery(cq);
List<Pet> allPets = q.getResultList();
```

The equivalent JPQL query is:

```
SELECT p
FROM Pet p
```

This query demonstrates the basic steps to create a Criteria query:

1. Use an `EntityManager` instance to create a `CriteriaBuilder` object.
2. Create a query object by creating an instance of the `CriteriaQuery` interface. This query object's attributes will be modified with the details of the query.
3. Set the query root by calling the `from` method on the `CriteriaQuery` object.
4. Specify what the type of the query result will be by calling the `select` method of the `CriteriaQuery` object.
5. Prepare the query for execution by creating a `TypedQuery<T>` instance, specifying the type of the query result.
6. Execute the query by calling the `getResultList` method on the `TypedQuery<T>` object. Because this query returns a collection of entities, the result is stored in a `List`.

The tasks associated with each step are discussed in detail in this chapter.

To create a `CriteriaBuilder` instance, call the `getCriteriaBuilder` method on the `EntityManager` instance:

```
CriteriaBuilder cb = em.getCriteriaBuilder();
```

The query object is created by using the `CriteriaBuilder` instance:

```
CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);
```

The query will return instances of the `Pet` entity, so the type of the query is specified when the `CriteriaQuery` object is created to create a typesafe query.

The `FROM` clause of the query is set, and the root of the query specified, by calling the `from` method of the query object:

```
Root<Pet> pet = cq.from(Pet.class);
```

The `SELECT` clause of the query is set by calling the `select` method of the query object and passing in the query root:

```
cq.select(pet);
```

The query object is now used to create a `TypedQuery<T>` object that can be executed against the data source. The modifications to the query object are captured to create a ready-to-execute query:

```
TypedQuery<Pet> q = em.createQuery(cq);
```

This typed query object is executed by calling its `getResultList` method, because this query will return multiple entity instances. The results are stored in a `List<Pet>` collection-valued object.

```
List<Pet> allPets = q.getResultList();
```