Аннотации JPA

Hibernate поддерживает аннотации, определенные в спецификации JPA. Использование данных аннотаций позволяет сохранить совместимость программного кода при использовании другой реализации JPA, отличной от Hibernate.

@Access — для указания типа доступа связанного класса сущности, сопоставленного супер класса или встраиваемого класса или атрибута сущности.

@AssociacionOverride — для переопределения реляционных отношений таких как один к одному, многие к одному, один ко многим, многие ко многим (@OneToOne, @ManyToOne, @OneToMany, @ManyToMany) в классах унаследованных от встраиваемых (embeddable) или сопоставляемых супер классов (mapped superclass).

@AssociacionOverrides — для группировки нескольких аннотаций @AssociacionOverride.

@AttibuteOverride — для переопределения сопоставляемых атрибутов Entity классов унаследованных от встраиваемых (embeddable) или сопоставляемых супер классов (mapped superclass). Подробнее...

@AttibuteOverrides — для группировки нескольких аннотаций @AttributeOverride.

@Basic — указывает на простейший тип маппинга данных на колонку таблицы базы данных. Также в параметрах аннотации можно указать fetch стратегию доступа к полю и является ли это поле обязательным или нет.

@Cacheable — для определения хранения объекта в кэше второго уровня в зависимости от установленного свойства [shared-cache-mode] в файле persistence.xml. При значении [shared-cache-mode] - ENABLE SELECTIVE - в кэше второго уровня будут храниться только объекты помеченные аннотацией @Cacheable. При значении [shared-cache-mode] - DISABLE SELECTIVE - в кэше второго уровня будут храниться только объекты не помеченные аннотацией @Cacheable.

@CollectionTable — для указания таблицы базы данных, в которой хранятся значения базовой или встраиваемой коллекции типов.

@Column — для указания соответствия между атрибутом базовой сущности Entity класса и столбцом таблицы базы данных. Подробнее...

@ColumnResult — аннотация @ColumnResult используется в сочетании с аннотациями @SqlResultSetMapping или @ConstructorResult для отображения столбца SQL для заданного запроса SELECT.

@ConstructorResult — в сочетании с аннотациями @SqlResultSetMapping для сопоставления столбцов заданного запроса SELECT определенному конструктору объекта.

@Convert — для определения реализации AttributeConverter, используемой для преобразования текущего аннотированного базового атрибута. Если AttributeConverter использует autoApply, все атрибуты сущностей с одним и тем же целевым типом будут автоматически преобразованы.

@Converter — аннотирование используется, чтобы указать, что текущая реализация AttributeConverter аннотации может использоваться в качестве конвертора основных атрибутов JPA. Если атрибуту autoApply присвоено значение true, поставщик JPA автоматически преобразует все базовые атрибуты с тем же типом Java, как определено текущим преобразователем.

@Converts — для группирования нескольких аннотаций @Convert.

@DiscriminatorColumn — для указания имени столбца дискриминатора и типа дискриминатора для стратегий наследования (Inheritance) SINGLE\_TABLE и JOINED.

@DisccriminatorValue — для определения того, какое значение столбца дискриминатора используется для отображения текущего аннотированного объекта для стратегий наследования (Inheritance) SINGLE\_TABLE и JOINED.

@ElementCollection — используется для указания/загрузки коллекции/сторонней коллекции, но не для связи (как @OneToMany или д.р.) двух Entity. И при этом никакого дополнительного поля в БД не появляется.

@Embeddable — для указания встраиваемых типов. Как и базовые типы, встраиваемые типы не имеют никакой идентичности, управляемой их собственностью.

@Embedded — чаща всего используется для создания составного ключа.

@EmbeddedId — говорит, что это поле будет из @Embedded.

@Entity — текущий класс представляет тип сущности - Entity класса. В отличие от базовых и встраиваемых типов, типы сущностей имеют идентичность, а их состояние управляется базовым контекстом Persistence. Подробнее...

@EntityListeners — для указания массива классов слушателя обратного вызова, которые используются текущей аннотированной сущностью.

@EntityResult — с аннотацией @SqlResultSetMapping для сопоставления выбранных столбцов сущности.

@Enumerated — атрибут entity представляет перечисляемый тип.

@ExcludeDefaultListeners — текущая аннотированная сущность пропускает вызов любого слушателя по умолчанию.

@ExcludeSuperlassListeners — текущая аннотированная сущность пропускает вызов слушателей, объявленных его суперклассом (классом предком).

@FieldResult — с аннотацией @EntityResult для сопоставления выбранных столбцов полям определенного объекта.

@ForeignKey — для указания связанного внешнего ключа сопоставления @JoinColumn. Аннотация @ForeignKey используется только в том случае, если включен инструмент автоматического создания и корректировки схемы базы данных, и в этом случае аннотация позволяет настроить определение базового внешнего ключа.

@GeneratedValue — аннотация указывает метод генерации значения идентификатора (автоматически генерируется с использованием столбца идентификации, последовательности базы данных или генератора таблиц). Hibernate поддерживает сопоставление @GeneratedValue даже для идентификаторов UUID.

@Id — аннотация указывает идентификатор объекта. Объект должен всегда иметь атрибут идентификатора, который используется при загрузке объекта в данном контексте сохранения.

@IdClass — , если текущий объект определяет составной идентификатор. Отдельный класс инкапсулирует все атрибуты идентификатора, которые зеркалируются текущим сопоставлением объектов.

@Index — для создания индекса базы данных если включен инструмент автоматического создания и корректировки схемы базы данных.

@Inheritance — аннотирование используется для указания стратегии наследования для данной иерархии классов сущностей.

@JoinColumn — для указания столбца FOREIGN KEY, используемого при присоединении к ассоциации объекта или встраиваемой коллекции.

@JoinColumns — для группирования нескольких аннотаций @JoinColumn, которые используются при сопоставлении объектов или встраиваемой коллекции с использованием составного идентификатора.

@JoinTable — для указания таблицы связей между двумя другими таблицами базы данных.

@Lob — текущий аннотированный атрибут объекта представляет большой тип объекта.

@ManyToMany — для указания отношения объектов базы данных «многие-ко-многим».

@ManyToOne — для указания отношения «многие-к-одному» для объектов базы данных.

@MapKey — мы не просто получаем из связанной сущности все ее переменные, а и сразу группируем их по одной из этих переменных, используемой в виде ключа в Map.

@MapKeyClass — для указания ключа java.util.Map.

@MapKeyColumn — для указания столбца базы данных, в котором хранится ключ java.util.Map, где ключ карты является базовым типом.

@MapKeyEnumerated — в коллекции java.util.Map тип ключа будет Java Enum.

@MapKeyJoinColumn — ключ java.util.Map является ассоциацией сущностей. Ключевой столбец карты - это FOREIGN KEY в таблице связей, который также присоединяет таблицу владельца карты к таблице, в которой находится значение Map.

@MapKeyJoinColumns — для группирования нескольких сопоставлений @MapKeyJoinColumn, когда ключ ассоциации java.util.Map использует составной идентификатор.

@MapKeyTemporal — ключом ассоциации java.util.Map является @TemporalType (например, DATE, TIME, TIMESTAMP).

@MappedSuperlass — текущие атрибуты аннотированного типа наследуются любой Entity сущностью класса-наследника. Подробнее...

@MapsId — идентификатор объекта сопоставляется текущей аннотированной ассоциацией @ManyToOne или @OneToOne.

@NamedAttributeNode — для указания каждого индивидуального узла атрибута, который необходимо извлечь с помощью диаграммы сущностей.

@NamedEntityGraph — для указания графа сущностей, который может использоваться запросом сущности, чтобы переопределить план выборки по умолчанию.

@NamedEntityGraphs — для группировки нескольких аннотаций @NamedEntityGraph.

@NamedNativeQuery — для указания нативного SQL-запроса, который впоследствии можно найти по его имени.

@NamedNativeQueries — для объединения нескольких аннотаций @NamedNativeQuery.

@NamedQuery — для указания JPQL-запроса, который впоследствии можно найти по его имени.

@NamedQueries — для группирования нескольких аннотаций @NamedQuery.

@NamedStoredProcedureQuery — для указания хранимой процедуры базы данных, которую впоследствии можно найти по его имени.

@NamedStoredProcedureQueries — для группирования нескольких аннотаций @NamedStoredProcedureQuery.

@NamedSubgraph — аннотация используемая для указания субграфа в графе сущностей.

@OneToMany — для указания отношения объектов базы данных «один ко многим».

@OneToOne — для указания отношения «один к одному» объектов базы данных. Подробнее...

@OrderBy — используется для сортировки при получении (*в запросе автоматом добавляется “order by”*) текущей аннотированной коллекции. Не используется совместно с @OrderColumn.

@OrderColumn — используется для поддержания порядка в списке, путем добавления дополнительного столбца в таблицу соединения. Позиции из java.util.**List** помещаются в этот искусственный столбец с происхождением 0. Следовательно, порядок списка сохраняется ... в этом весь смысл прозрачного сохранения. Не используется совместно с @OrderBy.

@PersistenceContext — для указания EntityManager, который необходимо ввести как зависимость.

@PersistenceContexts — для группирования нескольких @PersistenceContext аннотаций.

@PersonistenceProperty — аннотацией @PersistenceContext для объявления свойств провайдера JPA, которые передаются в базовый контейнер при создании экземпляра EntityManager.

@PersonistenceUnit — для указания EntityManagerFactory, которая должна быть введена как зависимость.

@PersonistenceUnits — для группирования нескольких аннотаций @PersistenceUnit.

@PostLoad — для указания метода обратного вызова, который срабатывает после загрузки объекта.

@PostPersist — для указания метода обратного вызова, который срабатывает после сохранения объекта.

@PostRemove — для указания метода обратного вызова, который срабатывает после удаления объекта.

@PostUpdate — для указания метода обратного вызова, который срабатывает после обновления объекта.

@PrePersist — для указания метода обратного вызова, который срабатывает до того, как объект будет сохранен.

@PreRemove — для указания метода обратного вызова, который срабатывает до удаления объекта.

@PreUpdate — для указания метода обратного вызова, который срабатывает до обновления объекта.

@PrimaryKeyJoinColumn — столбец первичного ключа текущего аннотированного объекта также является внешним ключом к некоторому другому объекту (например, таблице базового класса в стратегии наследования JOINED, первичной таблице во вторичном сопоставлении таблиц или родительская таблица в отношении @OneToOne).

@PrimaryKeyJoinColumns — для группирования нескольких аннотаций @PrimaryKeyJoinColumn.

@QueryHint — для указания подсказки поставщика JPA, используемой аннотацией @NamedQuery или аннотацией @NamedNativeQuery.

@SecondaryTable — для указания вторичной таблицы для текущего аннотированного объекта Entity класса.

@SecondaryTables — для группирования нескольких аннотаций @SecondaryTable.

@SequenceGenerator — для указания последовательности базы данных, используемой генератором идентификатора текущего аннотированного объекта.

@SqlResultSetMapping — для указания отображения ResultSet собственного SQL-запроса или хранимой процедуры.

@SqlResultSetMappings — аннотация является аннотацией нескольких групп @SqlResultSetMapping.

@StoredProcedureParameter — для указания параметра @NamedStoredProcedureQuery.

@Table — для указания первичной таблицы текущего аннотированного объекта Entity класса. Подробнее...

@TableGenerator — для указания таблицы базы данных, используемой генератором идентификаторов текущего аннотированного объекта Entity класса.

@Temporal — для указания, что это поле будет java.util.Date или java.util.Calendar.

@Transient — для указания того, что данный атрибут сущности не должен сохраняться.

@UniqueConstraint — для указания уникального ограничения, которое должно быть включено генератором автоматической схемы для первичной или вторичной таблицы, связанной с текущим аннотированным объектом.

@Version — для указания атрибута версии сущности Entity класса, используемого для оптимистической блокировки.