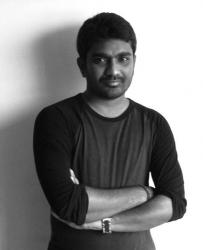
**@Temporal Annotation in JPA[[1]](#footnote-1)**

[](https://javabydeveloper.com/author/satishvarma/)

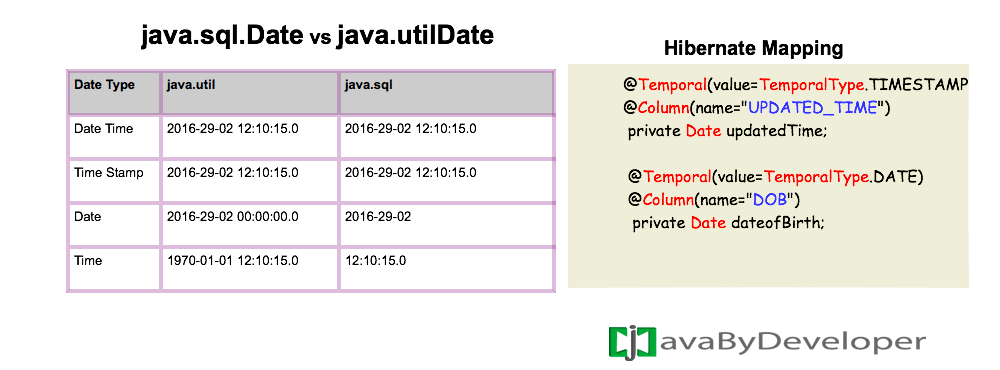
By [Satish Varma](https://javabydeveloper.com/author/satishvarma/)

February 27, 2020

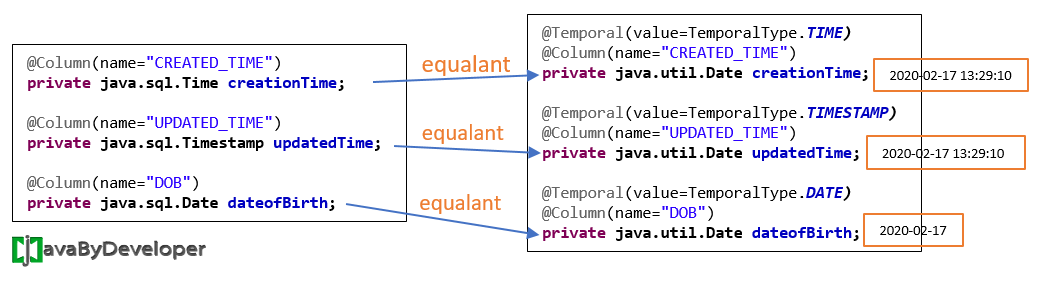
**1. Temporal annotation**

The java.util or java.time properties need to explicitly mark the SQL type correlation with the @Temporal annotation prior to Java 8. @Temporal should only be set on a java.util.Date or java.util.Calendar property.

* **@Temporal**  annotation must be specified for persistent fields or properties of type **java.util.Date** and **java.util.Calendar**.
* **@Temporal** annotation solves the one of the major issue of converting the date and time values from Java object to compatible database type and retrieving back to the application.
* The mapping between the Java 8 Date/Time classes and the SQL types is implicit, there is not need to specify the @Temporal annotation.

[](https://javabydeveloper.com/wp-content/uploads/2016/02/temporal.png)

* JPA supports  TemporalType.TIMESTAMP, TemporalType.DATE, TemporalType.TIME.
* These types are equivalent of  *java.sql.Timestamp*, *java.sql.Date*, *java.sql.Time*.



**Technologies Used in following example :**

* JPA 2.1
* Hibernate 5.2.6
* MySql 8.0
* Maven 3
* Spring Tool Suite (STS) 3.9.8
* Java 1.8

**2. @Temporal annotation mapping example**

|  |
| --- |
| @Entity(name="USER")  public class User {    @Id  @GeneratedValue(strategy = GenerationType.AUTO, generator = "native")  @GenericGenerator(name = "native", strategy = "native")  @Column(name = "ID")  private Long id;    @Column(name="USER\_NAME")  private String userName;    @Column(name="PASSWORD")  private String password;    @Temporal(value=TemporalType.TIMESTAMP)  @Column(name="CREATED\_TIME")  private Date creationTime;    @Temporal(value=TemporalType.TIMESTAMP)  @Column(name="UPDATED\_TIME")  private Date updatedTime;    @Temporal(value=TemporalType.DATE)  @Column(name="DOB")  private Date dateofBirth;    @Enumerated(value=EnumType.STRING)  @Column(name="USER\_TYPE")  private UserType userType; |

* From the above snippet line numbers 16,20,24 describes the mapping of **Temporal** type to the Entity.
* @Temporal annotation more useful to convert specific type either ***TimeStamp*** or ***Time*** or ***Date***.

**3.1 Test the mapping**

|  |
| --- |
| public static void main( String[] args )  {  EntityManagerFactory emf = null;  EntityManager entityManager = null;  EntityTransaction transaction = null;    try{  emf = Persistence.createEntityManagerFactory("jbd-pu");  entityManager = emf.createEntityManager();  transaction = entityManager.getTransaction();  transaction.begin();    User user = new User();  user.setUserName("mike");  user.setPassword("password");  // Setting Current Date  user.setCreationTime(new Date());  user.setDateofBirth(new Date());  user.setUserType(UserType.EMPLOYEE);    entityManager.persist(user);  transaction.commit();  }catch(Exception e){  transaction.rollback();  }finally{  entityManager.close();  emf.close();  }  } |

**Download Application –**[**JPA-Temporal.zip**](https://javabydeveloper.com/wp-content/uploads/2016/02/JPA-Temporal.zip) (11 KB)

**3.2. You might be interested in following articles**

1. [Transient annotation in JPA and hibernate](https://javabydeveloper.com/transient-annotation-non-persistent-field-using-jpa-hibernate/)
2. [Hibernate Types](https://javabydeveloper.com/hibernate-entity-types-vs-value-types/)

**References**

1. [Hibernate document](https://docs.jboss.org/hibernate/orm/5.4/userguide/html_single/Hibernate_User_Guide.html#basic-datetime)

1. https://javabydeveloper.com/temporal/ [↑](#footnote-ref-1)