Difference Between JPA, Hibernate, and Spring Data JPA

Java Persistence API (JPA)

- Specification (JSR 338) for ORM in Java
- Only defines interfaces, no implementation
- Examples: Hibernate, EclipseLink, OpenJPA

Hibernate

- ORM tool and implementation of JPA
- Provides native API and JPA support
- Handles session, transactions, and queries

Spring Data JPA

- Spring abstraction over JPA (like Hibernate)
- No JPA implementation itself
- Reduces boilerplate code
- Integrates tightly with Spring

Code Comparison

Hibernate:

```
public Integer addEmployee(Employee employee) {
    Session session = factory.openSession();
    Transaction tx = null;
    Integer employeeID = null;
    try {
        tx = session.beginTransaction();
        employeeID = (Integer) session.save(employee);
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) tx.rollback();
        e.printStackTrace();
    } finally {
        session.close();
    }
    return employeeID;
}
```

Spring Data JPA:

```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {}

@Autowired
private EmployeeRepository employeeRepository;

@Transactional
public void addEmployee(Employee employee) {
    employeeRepository.save(employee);
}
```

Summary Table

Feature	JPA	Hibernate	Spring Data JPA
Type	Specification	ORM Tool	Abstraction Layer
Implementation	No	Yes	No (uses JPA impl.)
Boilerplate Reduction	No	No	Yes
Spring Integration	No	Yes	Yes
Code Complexity	Medium	High	Low

Reference Links

- https://dzone.com/articles/what-is-the-difference-between-hibernate-and-sprin-1
- https://www.javaworld.com/article/3379043/what-is-jpa-introduction-to-the-java-persistence-api.html