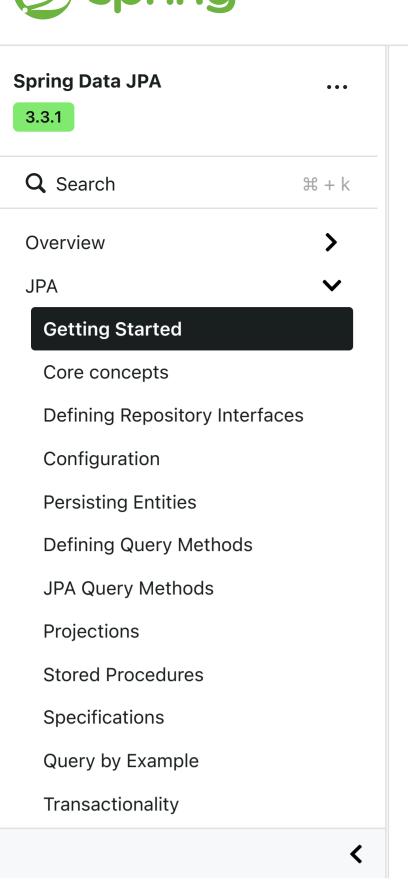
Learn 🗸



Spring Data JPA / JPA / Getting Started

Getting Started

An easy way to bootstrap setting up a working environment is to create a Spring-based project via start.spring.io or create a Spring project in Spring Tools.

Hello World Edit this Page GitHub Project Stack Overflow

Getting Started

Examples Repository

Examples Repository

The GitHub spring-data-examples repository hosts several examples that you can download and play around with to get a feel for how the library works.

Hello World

Let's start with a simple entity and its corresponding repository:

```
JAVA
@Entity
class Person {
  @Id @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;
  private String name;
  // getters and setters omitted for brevity
interface PersonRepository extends Repository<Person, Long> {
  Person save(Person person);
  Optional<Person> findById(long id);
```

Create the main application to run, as the following example shows:

```
JAVA
@SpringBootApplication
public class DemoApplication {
  public static void main(String[] args) {
    SpringApplication.run(DemoApplication.class, args);
  @Bean
  CommandLineRunner runner(PersonRepository repository) {
    return args -> {
      Person person = new Person();
      person.setName("John");
      repository.save(person);
      Person saved = repository.findById(person.getId()).orElseThrow(NoSuchElementExce
    };
```

Even in this simple example, there are a few notable things to point out:

- Repository instances are automatically implemented. When used as parameters of @Bean methods, these will be autowired without further need for annotations.
- The basic repository extends Repository. We suggest to consider how much API surface you want to expose towards your application. More complex repository interfaces are ListCrudRepository or JpaRepository.









