(/)

Accessing Data with JPA

This guide walks you through the process of building an application that uses Spring Data JPA to store and retrieve data in a relational database.

What you'll build

You'll build an application that stores `Customer` POJOs (/understanding/POJO) in a memory-based database.

What you'll need

- About 15 minutes
- A favorite text editor or IDE
- JDK 1.8 (http://www.oracle.com/technetwork/java/javase/downloads/index.html) or later
- Gradle 2.3+ (http://www.gradle.org/downloads) or Maven 3.0+ (https://maven.apache.org/download.cgi)
- You can also import the code straight into your IDE:
  - Spring Tool Suite (STS) (/guides/gs/sts)
  - IntelliJ IDEA (/guides/gs/intellij-idea/)

How to complete this guide

Like most Spring Getting Started guides (/guides), you can start from scratch and complete each step, or you can bypass basic setup steps that are already familiar to you. Either way, you end up with working code.

To **start from scratch**, move on to Build with Gradle.

To **skip the basics**, do the following:

- Download (https://github.com/spring-guides/gs-accessing-data-jpa/archive/master.zip) and unzip the source repository for this guide, or clone it using Git (/understanding/Git):
  `git clone https://github.com/spring-guides/gs-accessing-data-jpa.git (https://github.com/spring-guides/gs-accessing-data-jpa.git`
- cd into `gs-accessing-data-jpa/initial`
- Jump ahead to Define a simple entity.

When you're finished, you can check your results against the code in `gs-accessing-data-jpa/complete`.

▶ Build with Gradle

▶ Build with Maven

▶ Build with your IDE

Define a simple entity

In this example, you store Customer objects, annotated as a JPA entity.

`src/main/java/hello/Customer.java`