

Usando o Banco H2



Introdução

- Nessa aula acrescentaremos o Banco H2 no serviço implementado na aula anterior. Ou seja, ao invés de criarmos um objeto da classe A “na mão” trabalharemos com o Banco H2.
- Conforme descrito em aula anterior: o Banco de Dados H2 é um banco de dados escrito em Java que funciona em memória volátil, ou seja, não é armazenado em HD/SSD. Dessa forma, a cada instanciação da aplicação ele será reconstruído.



Configurando o Banco de Dados H2

- O primeiro passo é acrescentar as dependências do Banco H2 no arquivo **pom.xml** (arquivo gerenciado pelo Maven):
- Acrescentamos o que se encontra dentro do quadro de borda vermelha.

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

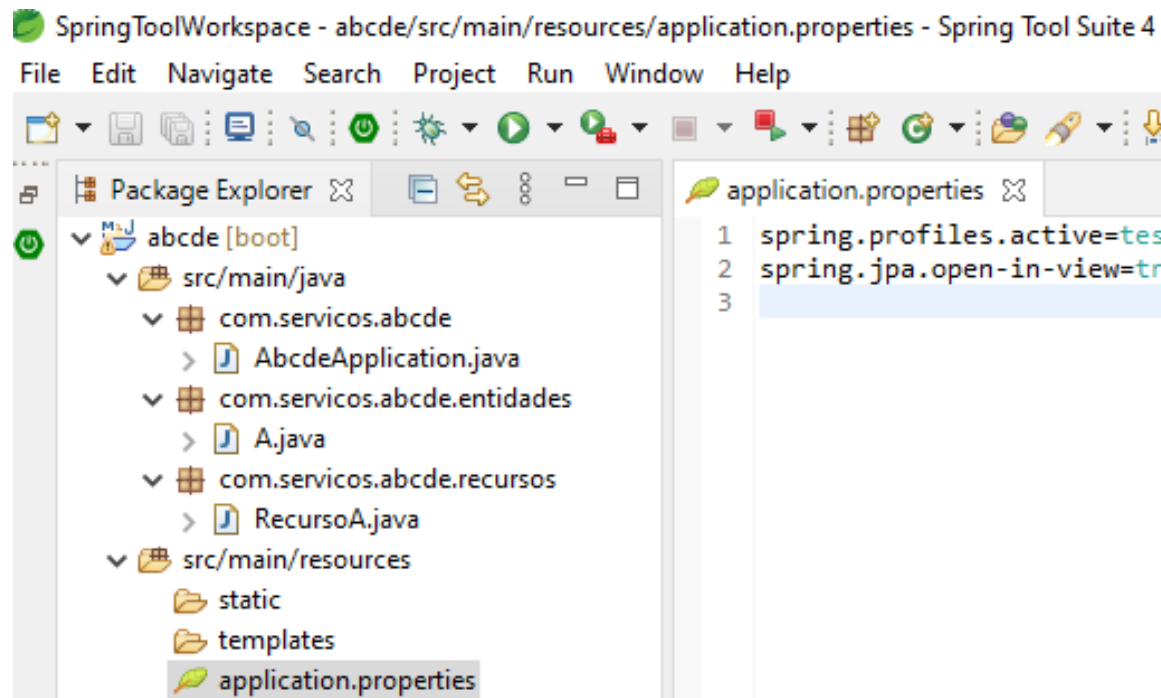
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>
  <dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
  </dependency>
</dependencies>
```



Configurando o perfil de teste

- No arquivo `application.properties` escreveremos 2 linhas:

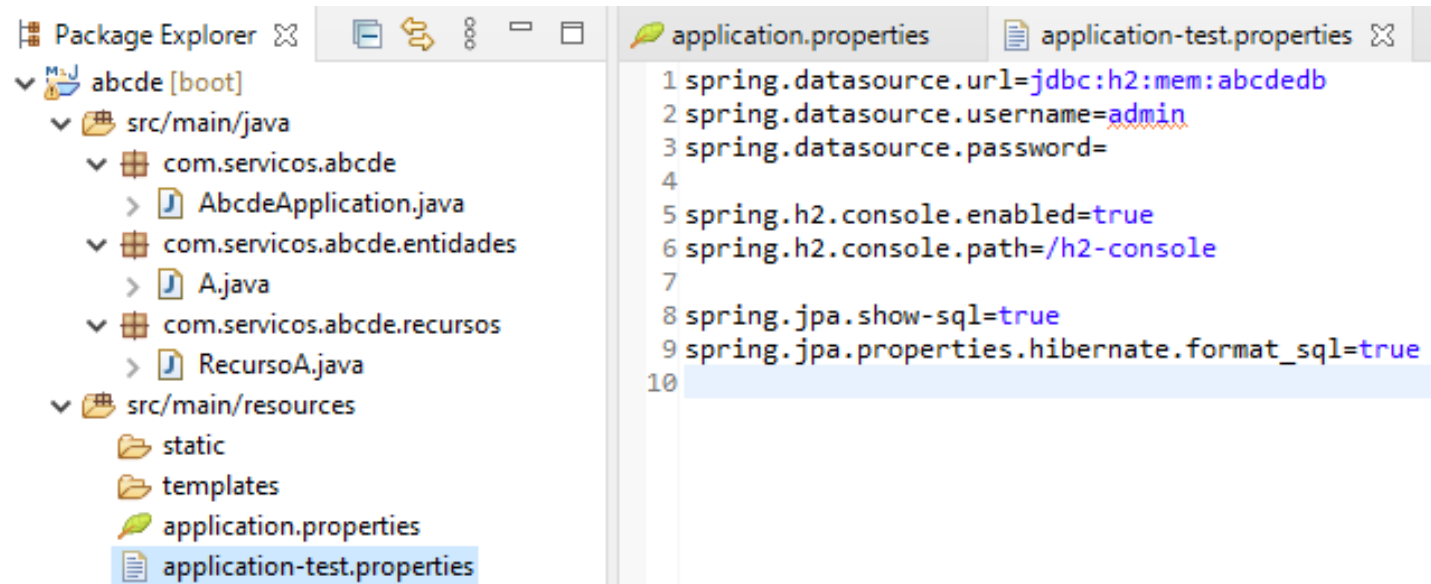


application-test.properties



Configurando o perfil de teste

- O próximo passo é criar o arquivo (application-test.properties) de configuração para o perfil de teste:

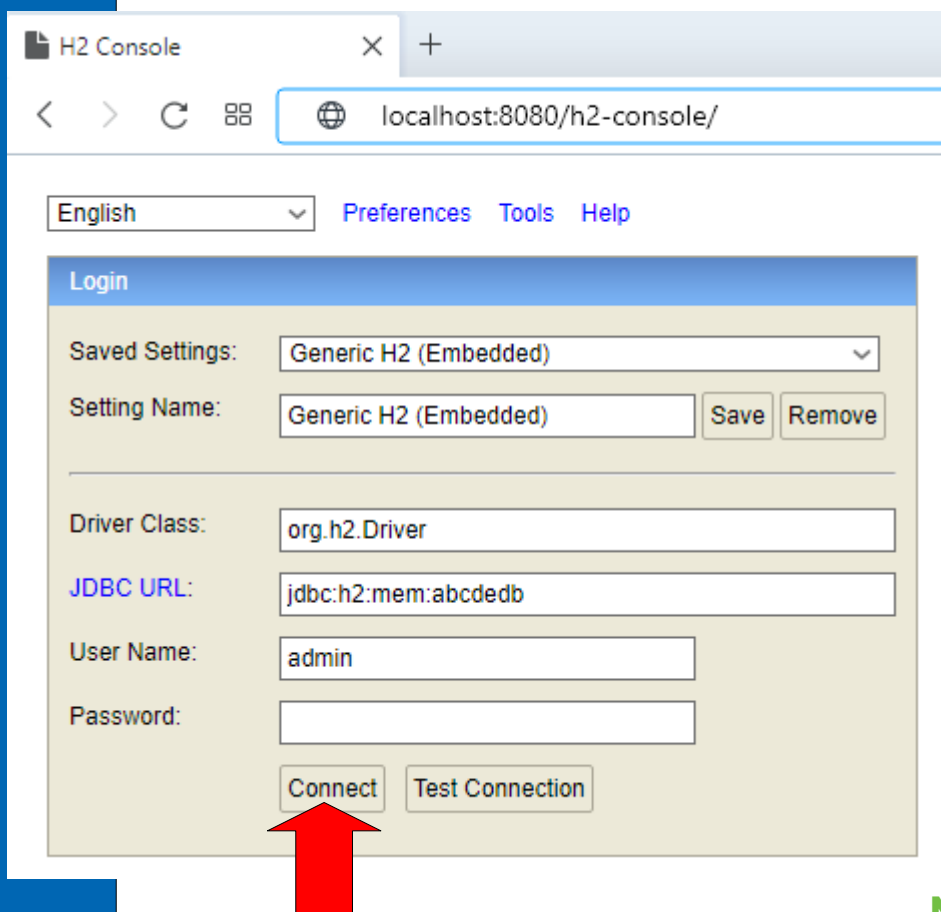


Configurando o perfil de teste

- String de conexão (abceddb é o banco de dados em memória): `spring.datasource.url=jdbc:h2:mem:abceddb`
- Usuário (admin): `spring.datasource.username=admin`
- Senha (vazio): `spring.datasource.password=`
- O console está habilitado e o caminho é /h2-console:
`spring.h2.console.enabled=true`
`spring.h2.console.path=/h2-console`
- As linhas abaixo são para mostrar o SQL gerado no log e torná-lo mais legível:
`spring.jpa.show-sql=true`
`spring.jpa.properties.hibernate.format_sql=true`



Acessando o console do H2



H2 Console

localhost:8080/h2-console/

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) Save Remove

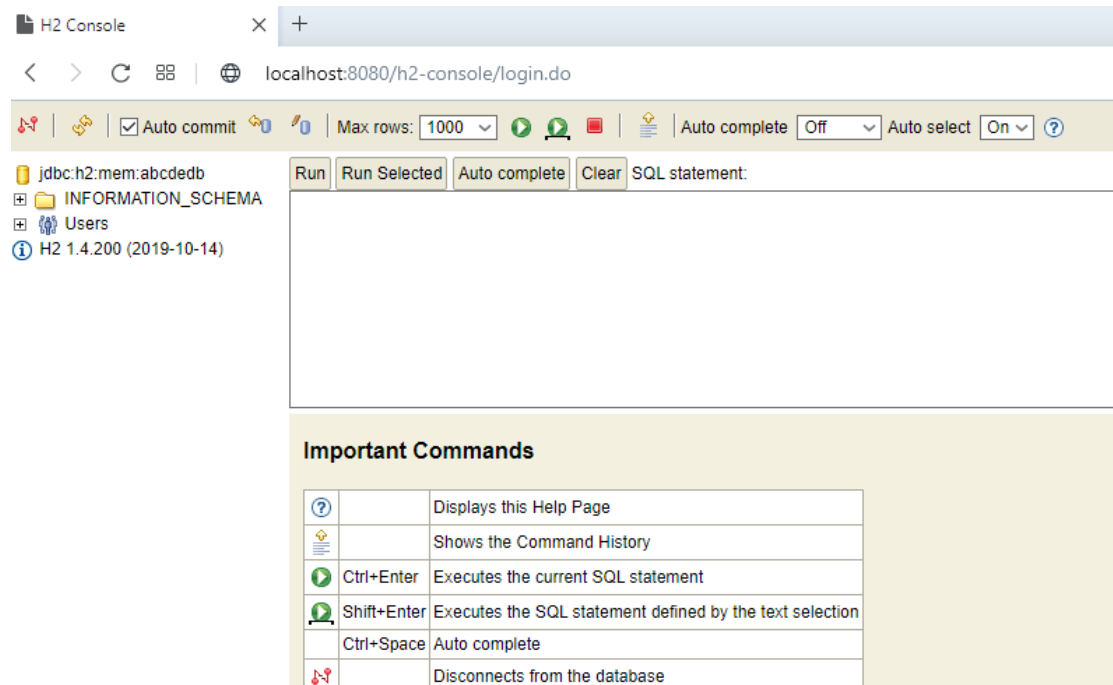
Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:abcdedb

User Name: admin

Password:

Connect Test Connection



H2 Console

localhost:8080/h2-console/login.do

Auto commit Max rows: 1000 Auto complete Off Auto select On

jdbc:h2:mem:abcdedb

INFORMATION_SCHEMA

Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement:

Important Commands

?	Displays this Help Page
	Shows the Command History
Ctrl+Enter	Executes the current SQL statement
Shift+Enter	Executes the SQL statement defined by the text selection
Ctrl+Space	Auto complete
	Disconnects from the database

Não esqueça que o serviço precisa estar rodando
Run As → Spring Boot App



Mappeamento do JPA – classe A

- Utilizaremos uma série de anotações para instruir o JPA de forma que possamos converter de objetos para o modelo relacional.

@Entity: indica que a classe A será uma entidade e deve ser gerada uma tabela.

@Table: possibilita definir o nome da tabela que queremos gerar (se não colocado a tabela criada se chamará A, que o nome da classe).

@Id: define o atributo que será chave primária da tabela.

@GeneratedValue: define a estratégia de geração da chave primária (no caso de IDENTITY será por auto incremento).

```
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonFormat;

@Entity
@Table(name = "tb_a")
public class A implements Serializable{
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
```



Testando ...

- Devemos rodar a aplicação Run As → Spring Boot App

[illegible]

```

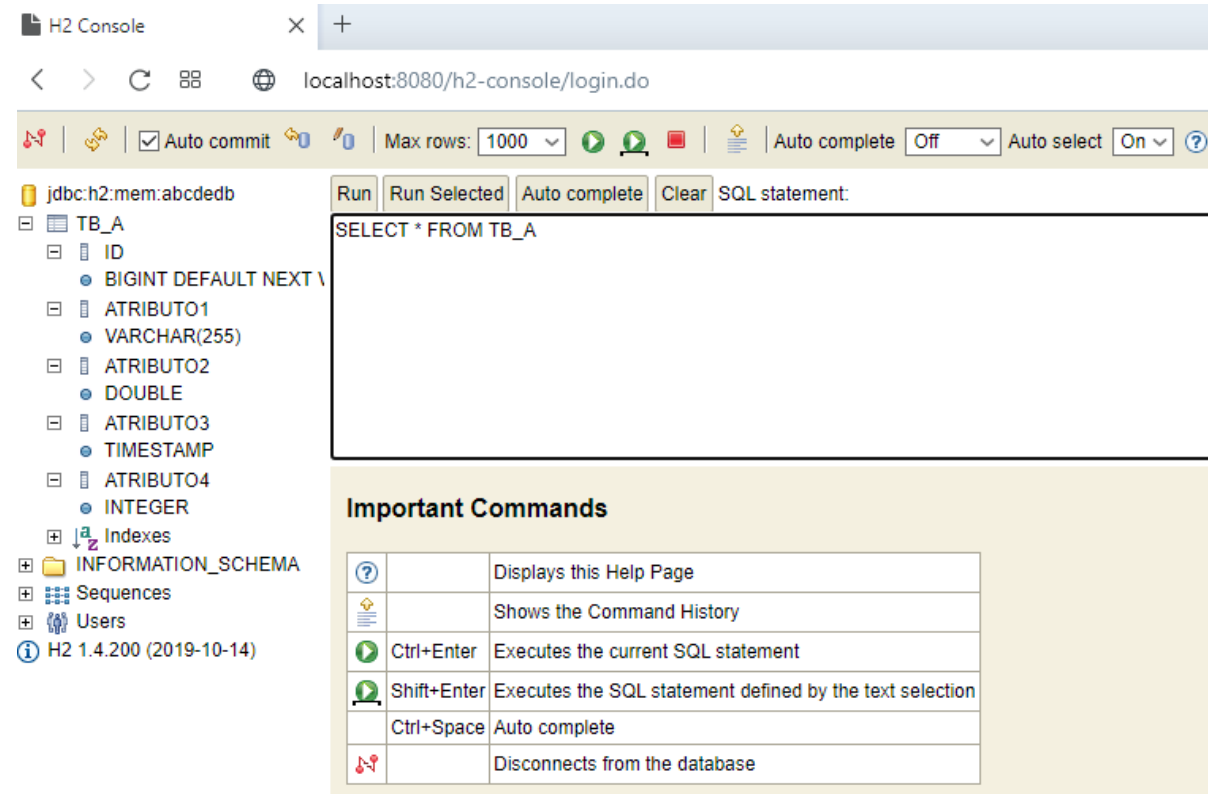
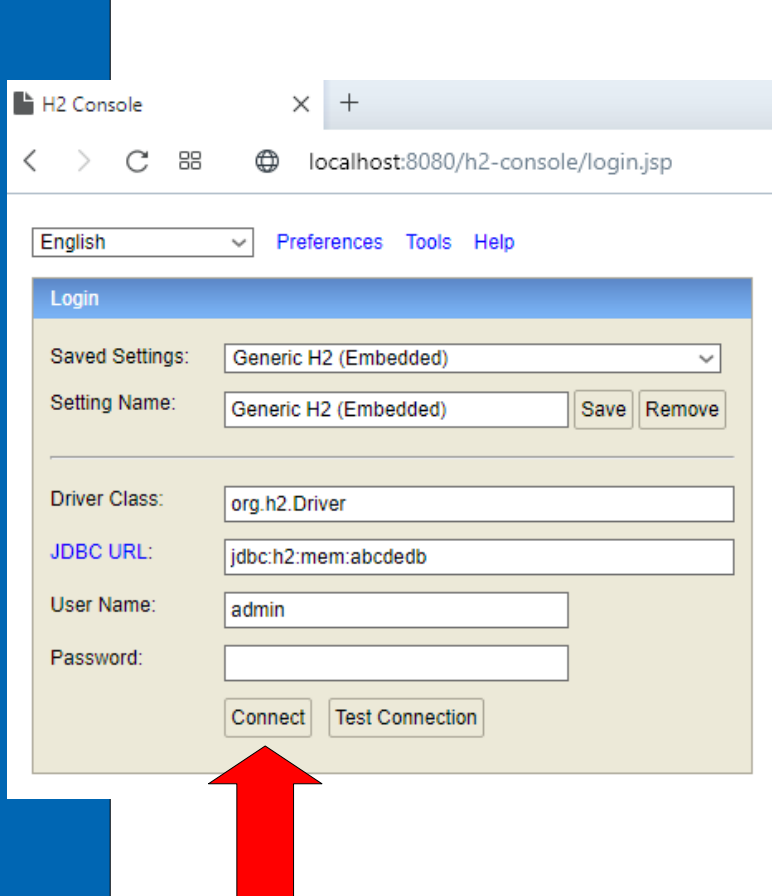
2021-06-10 14:02:26.588 INFO 13400 --- [main] com.servicos.abcde.AbcdeApplication : Starting AbcdeApplication using Java 15.0.2 on DESKTOP-JK861SJ with PID 13400 (C:\Users\Giovany\Documents\SpringToolWor
2021-06-10 14:02:26.594 INFO 13400 --- [main] com.servicos.abcde.AbcdeApplication : The following profiles are active: test
2021-06-10 14:02:28.838 INFO 13400 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2021-06-10 14:02:28.877 INFO 13400 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 16 ms. Found 0 JPA repository interfaces.
2021-06-10 14:02:30.393 INFO 13400 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2021-06-10 14:02:30.422 INFO 13400 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-06-10 14:02:30.423 INFO 13400 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.46]
2021-06-10 14:02:30.702 INFO 13400 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2021-06-10 14:02:30.702 INFO 13400 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 3934 ms
2021-06-10 14:02:30.783 INFO 13400 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2021-06-10 14:02:31.209 INFO 13400 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2021-06-10 14:02:31.222 INFO 13400 --- [main] o.s.b.a.h2.H2ConsoleAutoConfiguration : H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:abcdedb'
2021-06-10 14:02:31.647 INFO 13400 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2021-06-10 14:02:31.802 INFO 13400 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.4.31.Final
2021-06-10 14:02:32.120 INFO 13400 --- [main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2021-06-10 14:02:32.428 INFO 13400 --- [main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect

Hibernate: drop table if exists tb a CASCADE
Hibernate: create table tb a (id bigint generated by default as identity, atributo1 varchar(255), atributo2 double, atributo3 timestamp, atributo4 integer, primary key (id))
2021-06-10 14:02:33.833 INFO 13400 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2021-06-10 14:02:33.856 INFO 13400 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2021-06-10 14:02:34.725 INFO 13400 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2021-06-10 14:02:34.746 INFO 13400 --- [main] com.servicos.abcde.AbcdeApplication : Started AbcdeApplication in 9.238 seconds (JVM running for 10.329)
2021-06-10 14:02:34.748 INFO 13400 --- [main] o.s.b.a.ApplicationAvailabilityBean : Application availability state LivenessState changed to CORRECT
2021-06-10 14:02:34.752 INFO 13400 --- [main] o.s.b.a.ApplicationAvailabilityBean : Application availability state ReadinessState changed to ACCEPTING TRAFFIC

```

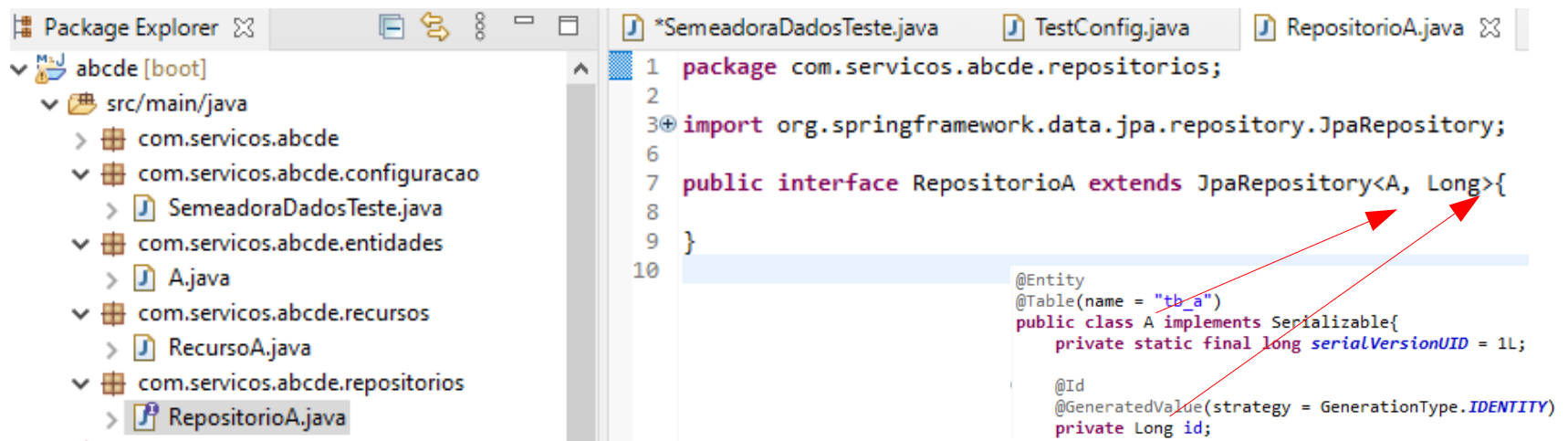


Visualizando no H2-Console



Criando RepositorioA

- Basta criar uma interface que herde de JpaRepository, conforme o padrão da imagem abaixo:

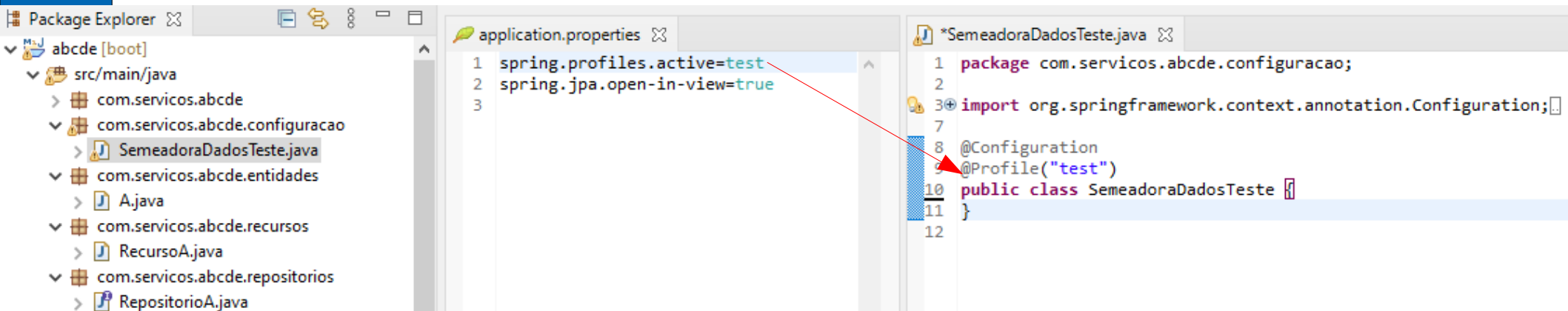


- O Spring Data JPA já possui uma implementação padrão para a interface JpaRepository e por consequência da interface que estamos definindo.
- A primeira classe do Generics de JpaRepository deve ser o tipo dos objetos que serão persistidos e a segunda classe é o tipo da chave primária utilizada.



Classe de configuração para criação dos dados do Banco H2

- Para trabalhar com o Banco H2 em “modo de teste” precisaremos criar uma classe de configuração.
- Através dessa classe faremos a inclusão de dados (database seeding – “semeando dados”) para teste do sistema que está sendo desenvolvido.



Fazendo a injeção de dependência

- A anotação `@Autowired` irá instanciar um objeto que responda a interface `RepositoryA`:

```
@Configuration
@Profile("test")
public class SemeadoraDadosTeste {

    /* Aqui é feita injeção de dependência:
     * @Autowired
     * O framework cria o objeto padrão
     * que implementa essa interface.
     */
    @Autowired
    private RepositoryA repositoryA;

}
```



Fazendo o Spring executar a SemeadoraDadosTeste

- Basta fazer a **SemeadoraDadosTeste** implementar **CommandLineRunner** e definir o método **run**:

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;

import com.servicos.abcde.repositorios.RepositorioA;

@Configuration
@Profile("test")
public class SemeadoraDadosTeste implements CommandLineRunner{

    /* Aqui é feita injeção de dependência:
     * @Autowired
     * O framework cria o objeto padrão
     * que implementa essa interface.
     */
    @Autowired
    private RepositorioA repositorioA;

    @Override
    public void run(String... args) throws Exception {
        // TODO Auto-generated method stub
    }
}
```



Semeando dados

```
@Configuration
@Profile("test")
public class SemeadoraDadosTeste implements CommandLineRunner{

    @Autowired
    private RepositorioA repositorioA;

    @Override
    public void run(String... args) throws Exception {
        A a1 = new A(null, "Atributo 11", 7.77, Instant.now(), 7);
        A a2 = new A(null, "Atributo 12", 2.77, Instant.parse("2020-07-20T19:51:09Z"), 2);

        repositorioA.saveAll(Arrays.asList(a1, a2));
    }
}
```

Console

abcde - AbcdeApplication [Spring Boot App] C:\sts-4.10.0.RELEASE\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (15 de jun. de 2021 13:31:11)

Spring Boot (v2.5.0)

```
2021-06-15 13:31:15.925 INFO 10936 --- [main] com.servicos.abcde.AbcdeApplication : Starting AbcdeApplication using Java 15.0.2 on DESKTOP-JKB61SJ with PID 109
2021-06-15 13:31:15.929 INFO 10936 --- [main] com.servicos.abcde.AbcdeApplication : The following profiles are active: test
2021-06-15 13:31:17.116 INFO 10936 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2021-06-15 13:31:17.230 INFO 10936 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 96 ms. Found 1 JPA repository i
2021-06-15 13:31:18.189 INFO 10936 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2021-06-15 13:31:18.210 INFO 10936 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-06-15 13:31:18.210 INFO 10936 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.46]
2021-06-15 13:31:18.552 INFO 10936 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2021-06-15 13:31:18.553 INFO 10936 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2532 ms
2021-06-15 13:31:18.622 INFO 10936 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2021-06-15 13:31:19.011 INFO 10936 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2021-06-15 13:31:19.024 INFO 10936 --- [main] o.s.b.a.h2.H2ConsoleAutoConfiguration : H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:
2021-06-15 13:31:19.340 INFO 10936 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2021-06-15 13:31:19.456 INFO 10936 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.4.31.Final
2021-06-15 13:31:19.759 INFO 10936 --- [main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2021-06-15 13:31:20.066 INFO 10936 --- [main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect

Hibernate: drop table if exists tb_a CASCADE
Hibernate: create table tb_a (id bigint generated by default as identity, atributo1 varchar(255), atributo2 double, atributo3 timestamp, atributo4 integer, primary key (id))
2021-06-15 13:31:21.265 INFO 10936 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transact
2021-06-15 13:31:21.284 INFO 10936 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2021-06-15 13:31:22.475 INFO 10936 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2021-06-15 13:31:22.490 INFO 10936 --- [main] com.servicos.abcde.AbcdeApplication : Started AbcdeApplication in 7.331 seconds (JVM running for 8.819)
2021-06-15 13:31:22.491 INFO 10936 --- [main] o.s.b.a.ApplicationAvailabilityBean : Application availability state LivenessState changed to CORRECT
Hibernate: insert into tb_a (id, atributo1, atributo2, atributo3, atributo4) values (null, ?, ?, ?, ?)
Hibernate: insert into tb_a (id, atributo1, atributo2, atributo3, atributo4) values (null, ?, ?, ?, ?)
2021-06-15 13:31:22.629 INFO 10936 --- [main] o.s.b.a.ApplicationAvailabilityBean : Application availability state ReadinessState changed to ACCEPTING_TRAFFIC
```



Semeando dados

H2 Console

localhost:8080/h2-console/login.jsp

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) Save Remove


Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:abcdedb

User Name: admin

Password:

Connect Test Connection



H2 Console

localhost:8080/h2-console/login.do

Auto commit Max rows: 1000 Auto complete Off Auto select On

jdbc:h2:mem:abcdedb

TB_A

- ID
- ATRIBUTO1
- ATRIBUTO2
- ATRIBUTO3
- ATRIBUTO4
- Indexes
- INFORMATION_SCHEMA
- Sequences
- Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement:


SELECT * FROM TB_A

SELECT * FROM TB_A;

ID	ATRIBUTO1	ATRIBUTO2	ATRIBUTO3	ATRIBUTO4
1	Atributo 11	7.77	2021-06-15 13:31:22.494083	7
2	Atributo 12	2.77	2020-07-20 16:51:09	2

(2 rows, 8 ms)

Edit



```
@Configuration
@Profile("test")
public class SemeadoraDadosTeste implements CommandLineRunner{

    @Autowired
    private RepositorioA repositorioA;

    @Override
    public void run(String... args) throws Exception {
        A a1 = new A(null, "Atributo 11", 7.77, Instant.now(), 7);
        A a2 = new A(null, "Atributo 12", 2.77, Instant.parse("2020-07-20T19:51:09Z"), 2);

        repositorioA.saveAll(Arrays.asList(a1, a2));
    }
}
```



Dúvidas?

