



MyBatis

Delivoro Sistemas

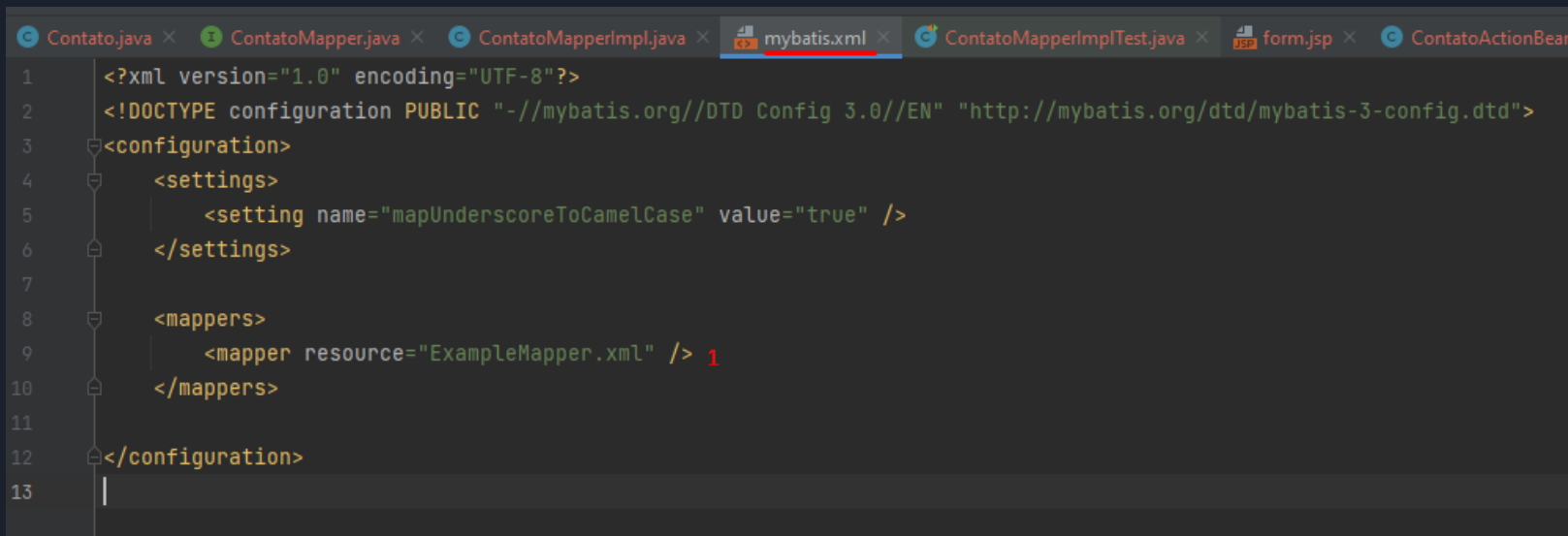


Visão Geral

- Framework de Persistência
 - <https://mybatis.org>
- Facilita uso de Bancos de Dados Relacionais
- Reduz a quase zero a escrita de código Java para
 - Conexão
 - Envio dos comandos SQL
 - Binding de parâmetros de consulta
 - Lidar com resultados e resultsets retornados
- Pode trabalhar com configurações em XML e através de annotations
- Integra bem com o Spring Framework

Configuração do MyBatis em XML

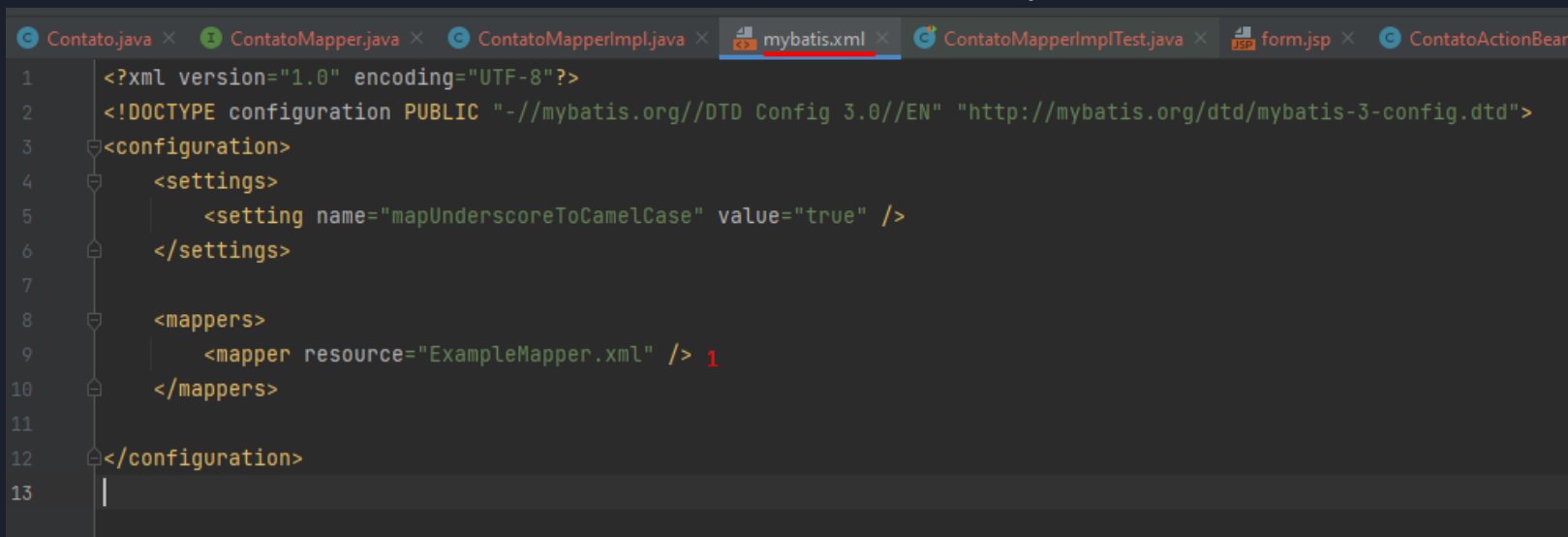
- É mais flexível e deixa os arquivos Java mais limpos
- Adequado para cenários onde temos consultas longas ou complexas



```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">
3  <configuration>
4    <settings>
5      <setting name="mapUnderscoreToCamelCase" value="true" />
6    </settings>
7
8    <mappers>
9      <mapper resource="ExampleMapper.xml" /> 1
10   </mappers>
11
12 </configuration>
13 |
```

Configuração do MyBatis em XML

- Cada entidade deve ter o seu próprio Mapper do MyBatis
- Para cada Mapper do MyBatis é necessário definir um arquivo XML e uma interface Java
- Esse arquivo XML deve ser incluído no mybatis.xml (veja 1 abaixo)



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">
3 <configuration>
4   <settings>
5     <setting name="mapUnderscoreToCamelCase" value="true" />
6   </settings>
7
8   <mappers>
9     <mapper resource="ExampleMapper.xml" /> 1
10  </mappers>
11
12 </configuration>
13 |
```

ExampleMapper.java

- A interface deverá ter os métodos Java que serão chamados para realizar operações no banco de dados
- Em 1 será retornada uma lista de objetos do tipo Example
- Em 2 está sendo passado um argumento como referência que será usado na WHERE clause do comando find

```
public interface ExampleMapper {  
  
    /** Lista todas as entidades */  
    List<Example> list(); 1  
  
    /** Encontra uma entidade com ID especificado. */  
    Example find(@Param("id") int id); 2  
  
    /** Insere uma entidade nova */  
    void insert(Example example); 3  
  
    /** Atualiza uma entidade existente. */  
    void update(Example example); 4  
  
    /** Apaga uma entidade. */  
    void delete(@Param("id") int id); 5  
}
```

ExampleMapper.java

- Em 3 e 4 temos a passagem do objeto Example para ser inserido e atualizado no banco de dados
- Em 5 temos a mesma situação que em 2 porém nada é retornado pois o comando DELETE não retorna nada do banco de dados
- Note que essa é uma Interface
- A classe concreta será implementada pelo MyBatis

```
public interface ExampleMapper {  
  
    /** Lista todas as entidades */  
    List<Example> list(); 1  
  
    /** Encontra uma entidade com ID especificado. */  
    Example find(@Param("id") int id); 2  
  
    /** Insere uma entidade nova */  
    void insert(Example example); 3  
  
    /** Atualiza uma entidade existente. */  
    void update(Example example); 4  
  
    /** Apaga uma entidade. */  
    void delete(@Param("id") int id); 5  
}
```

ExampleMapper.xml

- Esse é o arquivo XML usado pela Interface ExampleMapper.java
- Deve ter exatamente o mesmo nome (apenas a extensão difere)
- Os nomes dos métodos no arquivo Java devem bater com os ids dos comandos no XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="training.mapper.ExampleMapper">

    <resultMap id="exampleResult" type="training.model.Example"> 1
        <id property="id" column="id" />
        <result property="name" column="name" />
    </resultMap>

    <select id="list" resultMap="exampleResult"> 2
        SELECT * FROM example
    </select>

    <select id="find" resultMap="exampleResult"> 3
        SELECT * FROM example WHERE id = #{id}
    </select>
```

ExampleMapper.xml

- Em 1 temos a definição do mapeamento entre nomes de colunas e nomes de atributos no objeto Example - utilizado em comandos SELECT
- Em 2 temos um comando SELECT que retorna múltiplas linhas e que usa o result map definido em 1
- Em 3 temos um comando SELECT que retorna uma linha - note o uso de #{}

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="training.mapper.ExampleMapper">

    <resultMap id="exampleResult" type="training.model.Example"> 1
        <id property="id" column="id" />
        <result property="name" column="name" />
    </resultMap>

    <select id="list" resultMap="exampleResult"> 2
        SELECT * FROM example
    </select>

    <select id="find" resultMap="exampleResult"> 3
        SELECT * FROM example WHERE id = #{id}
    </select>
```


Configuração do MyBatis em XML

- Os próximos comandos são auto-explicativos
- Note que o nome dado em @Param do método delete do EmpresaMapper.java deve ser igual ao usado em #{id} do comando de remoção dos dados

```
<insert id="insert"> 4
    INSERT INTO example(id, name)
    VALUES (#{id}, #{name})
</insert>

<update id="update"> 5
    UPDATE example
    SET
        name = #{name}
    WHERE id = #{id}
</update>

<delete id="delete"> 6
    DELETE FROM example WHERE id = #{id}
</delete>

</mapper>
```