What you will learn in Topic 3

Azzeddine RIGAT

Topic 3: Introduction to Hibernate

- Introduction to Hibernate
- Setting Up Hibernate Development Environment
- Hibernate Configuration with Annotations
- Hibernate CRUD Features Create, Read, Update and Delete
- Hibernate Advanced Mappings
- Hibernate Advanced Mappings - @OneToOne
- Hibernate Advanced Mappings - @OneToMany
- Hibernate Advanced Mappings - Eager vs Lazy Loading
- Hibernate Advanced Mappings - @OneToMany - Unidirectional
- Hibernate Advanced Mappings - @ManyToMany

Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Introduction to Hibernate

Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate Overview

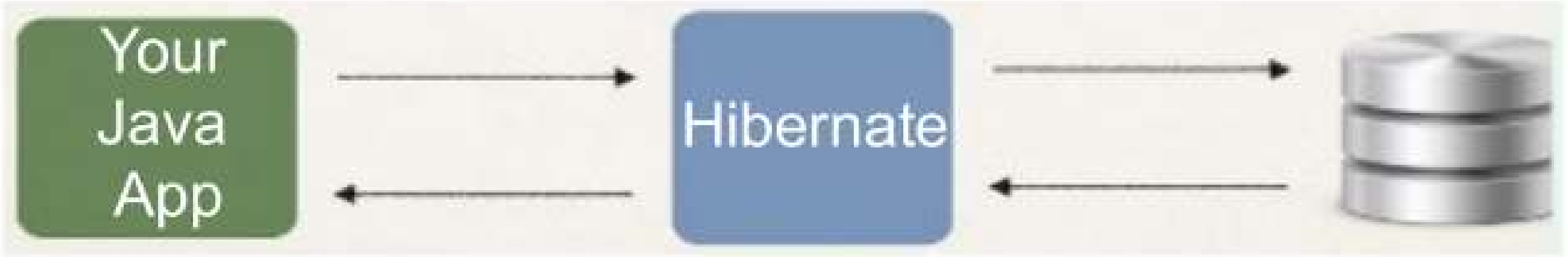
Topics

1. What is Hibernate?
2. Benefits of Hibernate
3. Code Snippets

Hibernate Overview

What is Hibernate?

1. A framework for persisting / saving Java objects in a database
2. www.hibernate.org



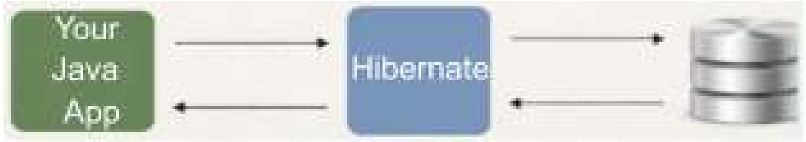
Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate Overview

Benefits of Hibernate

1. Hibernate handles all of the low-level SQL
2. Minimizes the amount of JDBC code you have to develop
3. Hibernate provides the Object-to-Relational Mapping (ORM)



Topic 3, Introduction to Hibernate

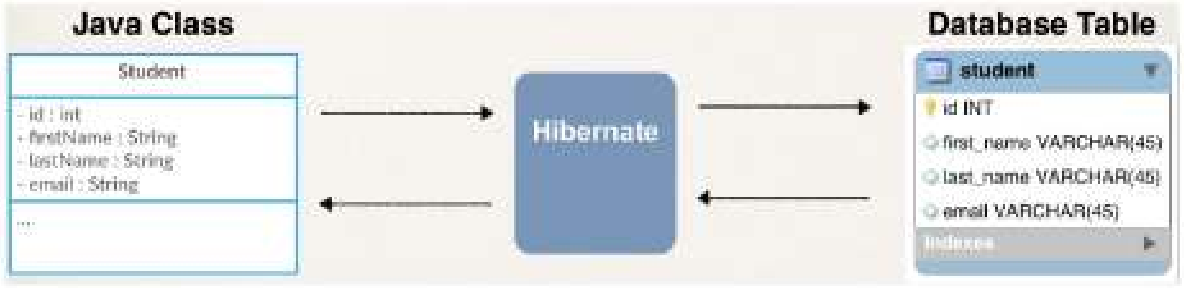
Azzeddine RIGAT

Hibernate Overview

Benefits of Hibernate

Object-To-Relational Mapping (ORM)

The developer defines mapping between Java class and database table



Topic 3, Introduction to Hibernate

Azzeddine
RIGAT

Hibernate Overview

Saving a Java Object with Hibernate

```
// create Java object
Student theStudent = new Student("Xu", "Ming", "xu@javaweb.edu");

// save it to database
int theId = (Integer) session.save(theStudent);
```

Azzeddine RIGAT

Hibernate Overview

Retrieving a Java Object with Hibernate

```
// create Java object
Student theStudent = new Student("Xu", "Ming", "rigat@javaweb.edu");

// save it to database
int theId = (Integer) session.save(theStudent);

// now retrieve from database using the primary key
Student myStudent = session.get(Student.class, theId);
```

Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate Overview

Querying for Java Objects

```
Query query = session.createQuery("from Student");
List<Student> students= query.list();
```

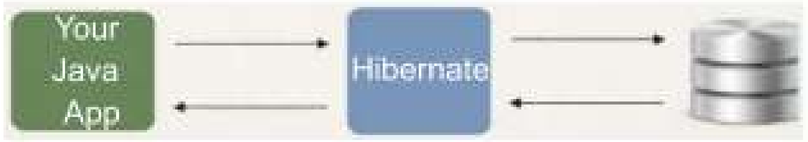
Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate Overview

Hibernate CRUD Apps

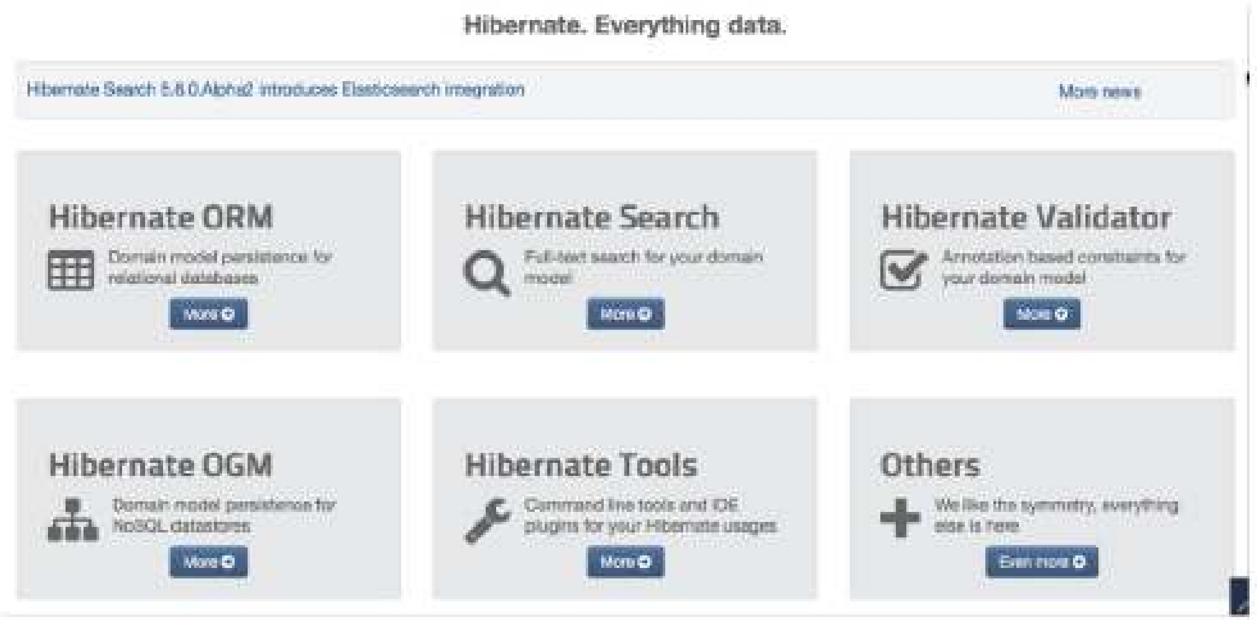
- **C**reate objects
- **R**ead objects
- **U**pdate objects
- **D**elete objects



Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate is actually more than ORM!



Hibernate. Everything data.

Hibernate Search 5.6.0.Alpha2 introduces Elasticsearch integration

More news

Hibernate ORM

Domain model persistence for relational databases

More

Hibernate Search

Full-text search for your domain model

More

Hibernate Validator

Annotation based constraints for your domain model

More

Hibernate OGM

Domain model persistence for NoSQL datastores

More

Hibernate Tools

Command line tools and IDE plugins for your Hibernate usages

More

Others

We like the symmetry, everything else is here

Even more

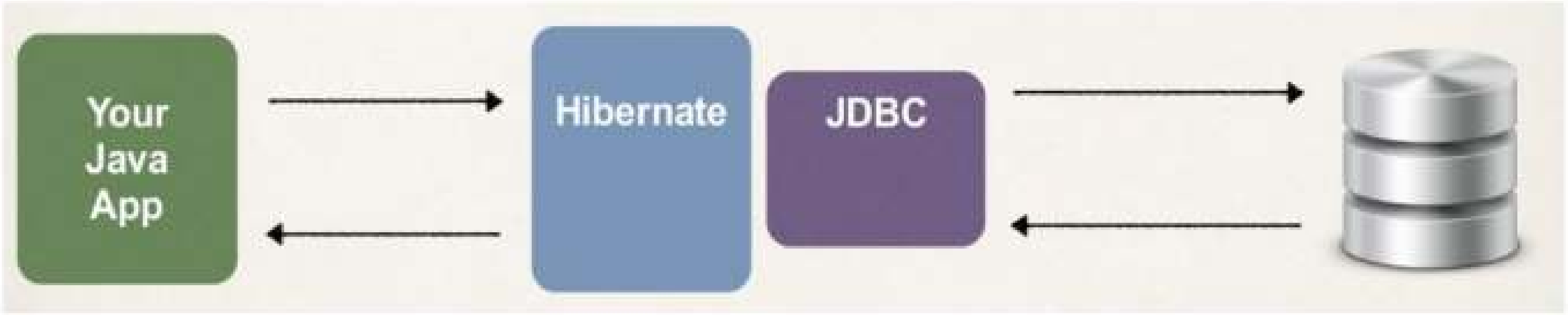
Topic 3, Introduction to Hibernate

Azzeddine RIGAT

Hibernate and JDBC

How does Hibernate relate to JDBC?

Hibernate uses JDBC for all database communications



Azzeddine RIGAT

Topic 3, Setting Up Hibernate Development Environment

Setting Up Hibernate Development Environment

Azzeddine RIGAT

Required Software

To Build Hibernate Applications, you need the following:

1. Java Integrated Development Environment (IDE)

2. Database Server

3. Hibernate JAR files and JDBC Driver

Topic 3, Setting Up Hibernate Development Environment

Azzeddine RIGAT

Install MySQL on MS Windows

1. Download MySQL
2. Install MySQL
3. Verify Installation

Topic 3, Setting Up Hibernate Development Environment

Azzeddine RIGAT

Install MySQL on MS Windows

1. Download MySQL

   https://dev..mysql.com/downloads/installer/

Check **demo-1-Hibernate-sql-scripts-and-starter**

Topic 3, Setting Up Hibernate Development Environment

Azzeddine RIGAT

Setup Database scripts

Folder: sql-scripts

1. create-user.sql

2. student-tracker.sql

Topic 3, Setting Up Hibernate Development Environment

Azzeddine RIGAT

Setup Database scripts

About:01-create-user.sql

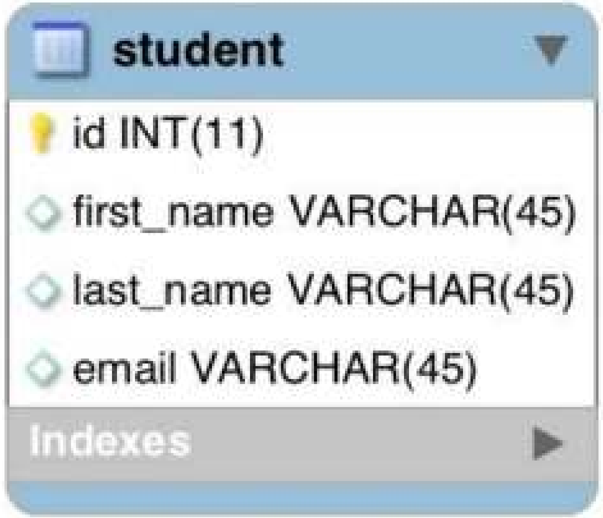
1.Create a new MySQL user for our application
   1. user id: **hbstudent**
   2. password: **hbstudent**

Azzeddine RIGAT

Setup Database scripts

About: 02-student-tracker.sql

1.Create a new database table: **student**



```
select * from hb_student_tracker.student;
```

Azzeddine RIGAT

Setup Hibernate in Eclipse

To Do List

1. Create Eclipse Project
2. Download Hibernate Files
3. Download MySQL JDBC Driver
4. Add JAR files to Eclipse Project … Build Path

Topic 3, Hibernate Configuration with Annotations

Azzeddine
RIGAT

Hibernate Configuration with Annotations

Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Test JDBC Connection - Hibernate Dev Process

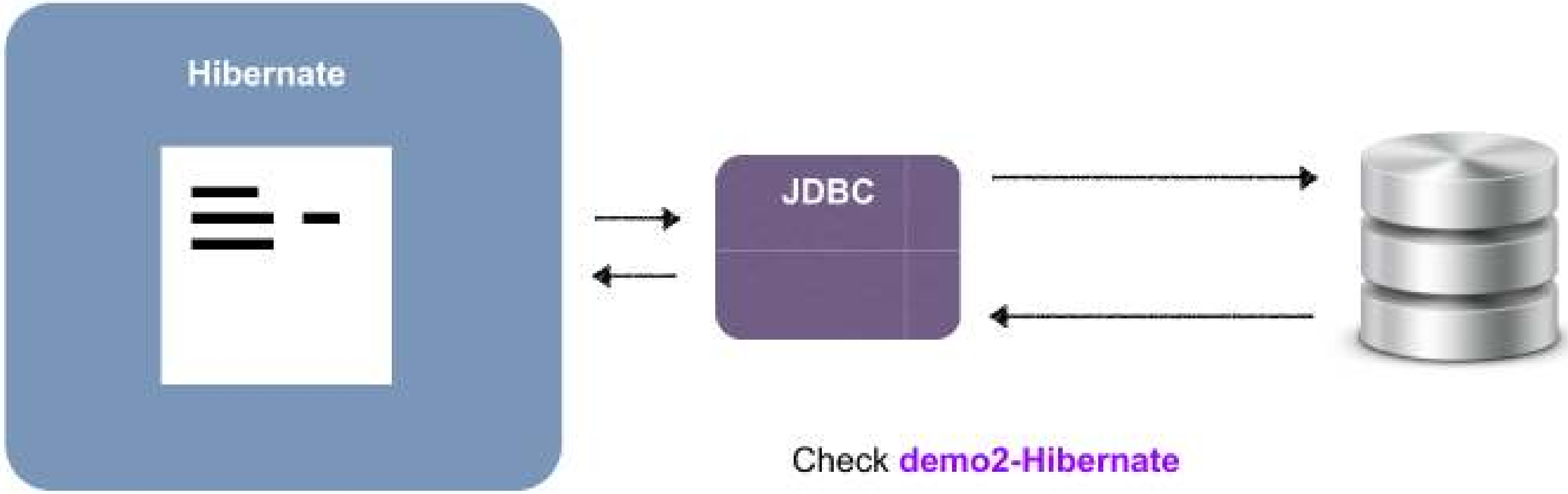
To Do List

1. Add Hibernate Configuration file
2. Annotate Java Class
3. Develop Java Code to perform database operations

Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Configuration File



Check **demo2-Hibernate**

Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Annotate Java Class

To Do List

1. Add Hibernate Configuration file

2. Annotate Java Class

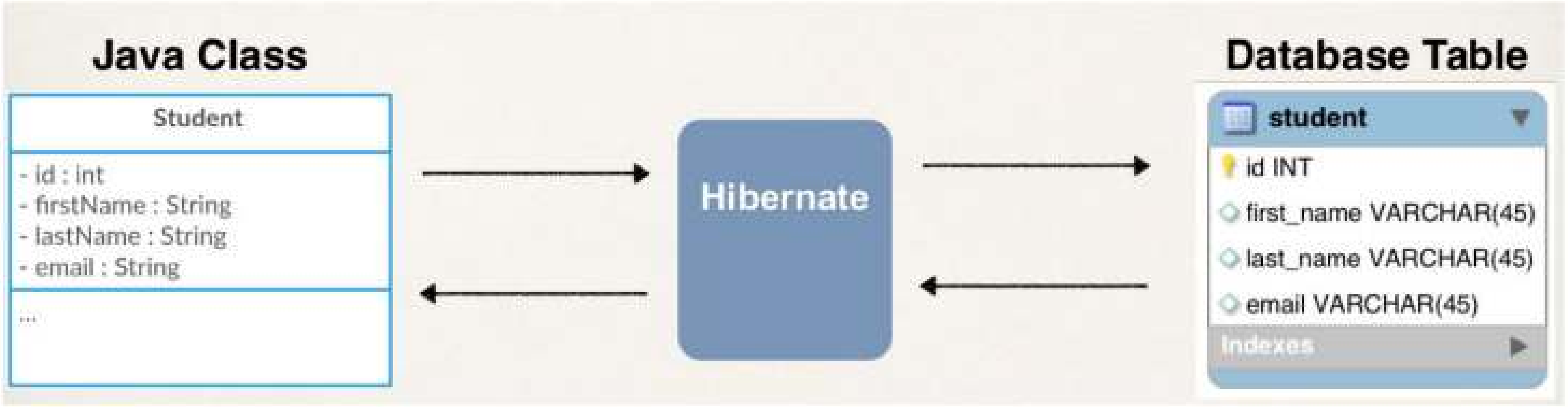
3. Develop Java Code to perform database operations

Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Entity Class Java class that is mapped to a database table

Object-to-Relational Mapping (ORM)



Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Annotate Java Class

Two Options for Mapping

1. Option 1: XML config file (legacy)

2. Option 2: Java Annotations (modern, preferred)

Topic 3, Hibernate Configuration with Annotations

Azzeddine RIGAT

Annotate Java Class

Java Annotations

1. Step 1: Map class to database table

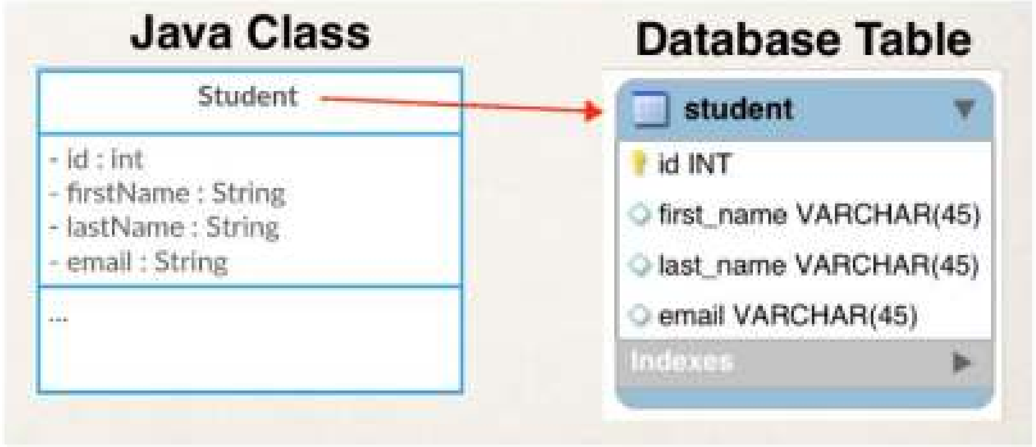
2. Step 2: Map fields to database columns

Azzeddine RIGAT

Topic 3, Hibernate Configuration with Annotations

Step 1: Map class to database table

```
@Entity
@Table(name="student")
public class Student {
...
}
```



Topic 3, Hibernate Configuration with Annotations

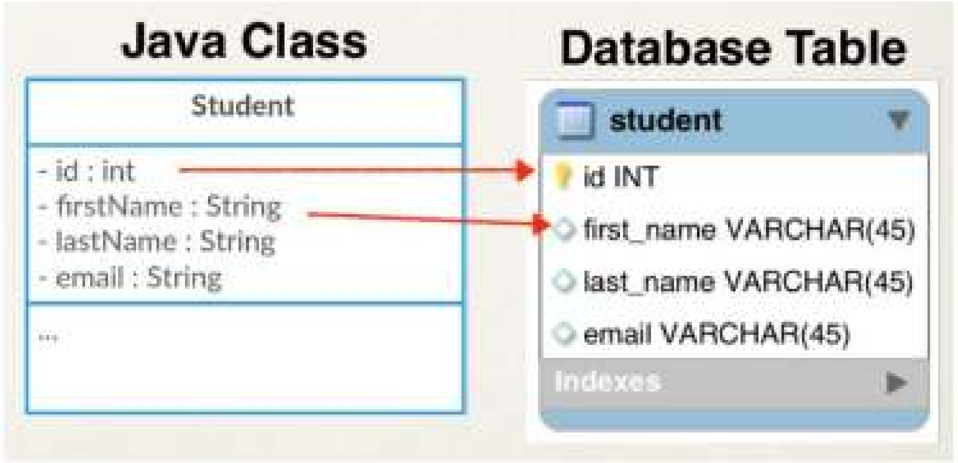
Azzeddine RIGAT

Step 2: Map fields to database columns

```
@Entity
@Table(name="student")
public class Student {

    @Id
    @Column(name="id")
    private int id;

    @Column(name="first_name")
    private String firstName;
    …
}
```



Check **demo3-Hibernate-Config**

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Hibernate CRUD Features Create, Read, Update and Delete

Hibernate Dev Process - To Do List

1. Add Hibernate Configuration file

2. Annotate Java Class

3. Develop Java Code to perform database operations

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

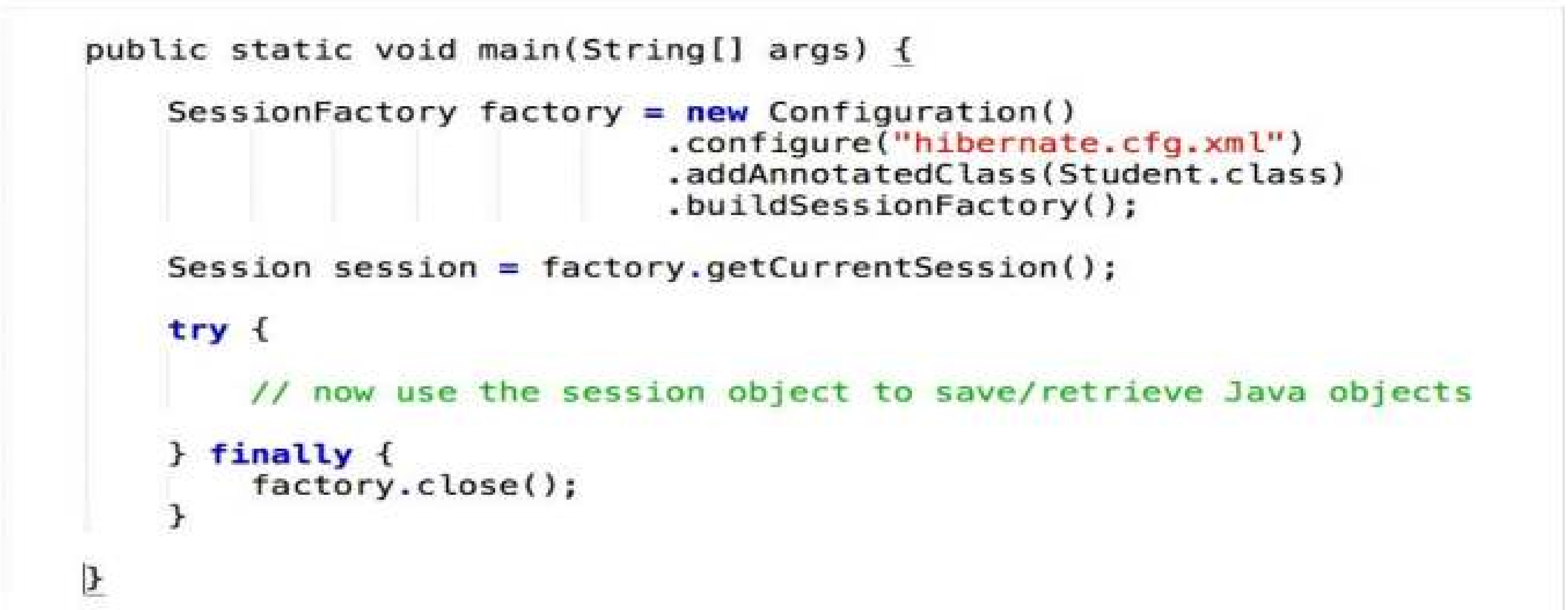
Two Key Players

Class	Description
SessionFactory	Reads the hibernate config file Creates Session objects Heavy-weight object Only create once in your app
Session	Wraps a JDBC connection Main object used to save/retrieve objects Short-lived object Retrieved from SessionFactory

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Java Code Setup



```java
public static void main(String[] args) {

    SessionFactory factory = new Configuration()
                                .configure("hibernate.cfg.xml")
                                .addAnnotatedClass(Student.class)
                                .buildSessionFactory();

    Session session = factory.getCurrentSession();

    try {

        // now use the session object to save/retrieve Java objects

    } finally {
        factory.close();
    }

}
```

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine
RIGAT

Save a Java Object

```java
try {

    // create a student object
    Student tempStudent = new Student("Paul", "Wall", "paul@luv2code.com");

    // start transaction
    session.beginTransaction();

    // save the student
    session.save(tempStudent);

    // commit the transaction
    session.getTransaction().commit();

} finally {
    factory.close();
}
```

Check **demo4-Hibernate-CreateStudent**

Hibernate and Primary Key

Primary Key

1. Uniquely identifies each row in a table
2. Must be a unique value
3. Cannot contain NULL values

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

MySQL - Auto Increment

```
CREATE TABLE student (

id int(11) NOT NULL AUTO_INCREMENT,
first_name varchar(45) DEFAULT NULL,  last_name varchar(45) DEFAULT NULL,  email varchar(45) DEFAULT
NULL,  PRIMARY KEY (id)

)
```

Hibernate Identity - Primary Key

```
@Entity  @Table(name="student")
public class Student {

   @Id  @Column(name="id")
   private int id;

   ...
}
```

Azzeddine RIGAT

Hibernate Identity - Primary Key

```
@Entity  @Table(name="student")
public class Student {

   @Id
   @GeneratedValue(strategy=GenerationType.IDENTITY)
   @Column(name="id")
   private int id;

   …
}
```

Check **demo5-Hibernate-AddMoreStudents**

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

ID Generation Strategies

Name	Description
GenerationType.AUTO	Pick an appropriate strategy for the particular database
GenerationType.IDENTITY	Assign primary keys using database identity column
GenerationType.SEQUENCE	Assign primary keys using a database sequence
GenerationType.TABLE	Assign primary keys using an underlying database table to ensure uniqueness

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Change the start point of your table index

```
alter table hb_student_tracker.student AUTO_INCREMENT=3000;
```

Delete all rows and set the index to 1

```
truncate hb_student_tracker.student;
```

Azzeddine RIGAT

Tricks

- You can define your own CUSTOM generation strategy :-)
- Create implementation of **org.hibernate.id.IdentifierGenerator**
- Override the method: **public Serializable generate(…)**

- **Always generate unique value**
- **Work in high-volume, multi-volume, multi-threaded environment**
- **If using server clusters, always generate unique value.**

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Retrieving a Java Object with Hibernate

```
// create Java object
Student theStudent = new Student("Mary", "Public", "mary@javaweb.com");

// save it to database
session.save(theStudent);
.....

//now retrieve/read from database usign the primary key
Student = myStudent = session.get(Student.class, theStudent.getId());
```

Check **demo6-Hibernate-read**

Querying Objects

- Query language for retrieving objects
- Similar in nature to SQL
- **where, like, order by, join, in, etc...**

```
List<Student> theStudents = session.createQuery("from Student").getResultList();
```

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Querying Objects

```
List<Student> theStudents =  session
.createQuery("from Student s where s.lastName='Doe'")
.getResultList();

List<Student> theStudents =  session
.createQuery("from Student s where s.lastName='Doe'"
+ " OR s.firstName='Daffy'")
.getResultList();

List<Student> theStudents =  session
.createQuery("from Student s where"
+ " s.email LIKE '%javaweb.edu'")
.getResultList();
```

Check **demo7-Hibernate-query**

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Updating Object(s)

```
int studentId = 1;

Student myStudent = session.get(Student.class, studentId);

// update first name to "Scooby"
myStudent.setFirstName("Scooby");

// commit the transaction
session.getTransaction().commit();

session.createQuery("update Student set email='foo@gmail.com'")
.executeUpdate();
```

Check **demo8-Hibernate-update**

Topic 3, Hibernate CRUD Features Create, Read, Update and Delete

Azzeddine RIGAT

Delete Object(s)

```
int studentId =1 ;
student myStudent = session.get(Studetn.class, studentId);

// delete the student
session.delete(myStudent)

// commit the transaction
session.getTransaction().commit();



session.createQuery("delete from Student where id=2")
        .executeUpdate();
```

Check **demo9-Hibernate-delete**

Topic 3, Hibernate Advanced Mappings

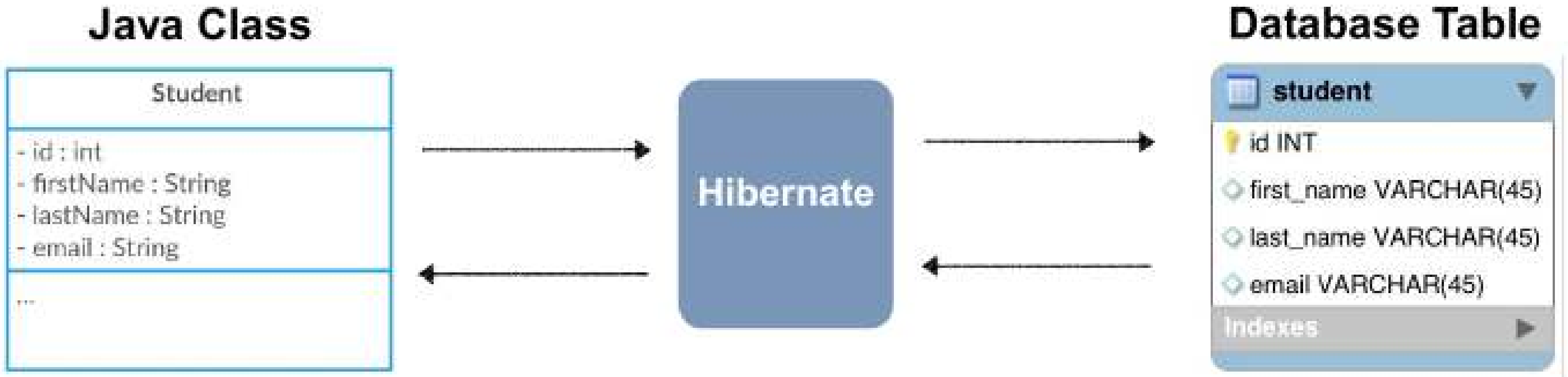
Azzeddine
RIGAT

Hibernate Advanced Mappings

Topic 3, Hibernate Advanced Mappings

Azzeddine RIGAT

Basic Mapping


Java Class
Student
- id : int
- firstName : String
- lastName : String
- email : String
...
Hibernate
Database Table
student
id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
email VARCHAR(45)
Indexes