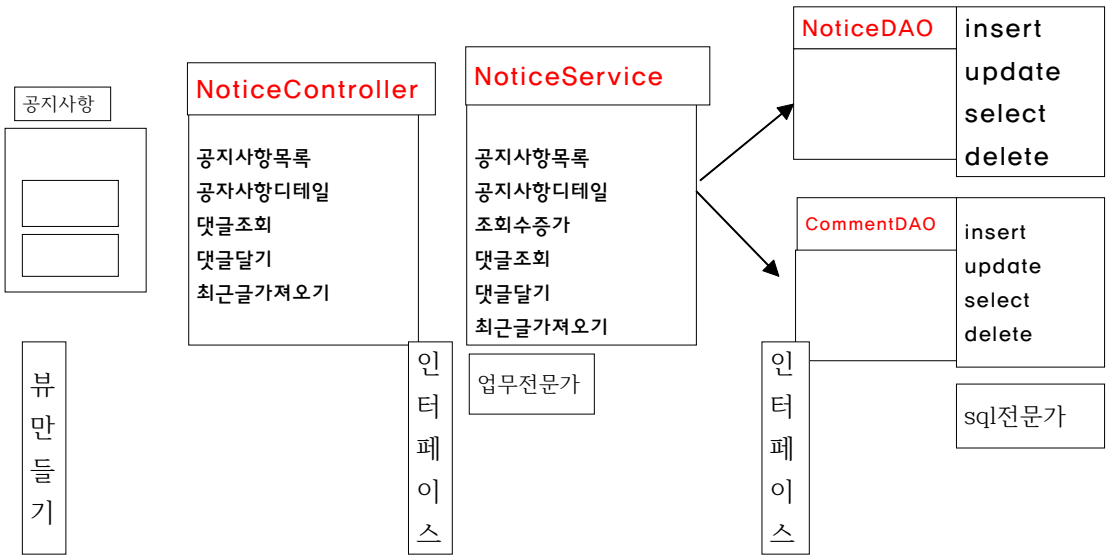
◎ 공지사항 서비스만들기



NoticeService 인터페이스 – 구현체 만들기
NoticeDAO 인터페이스 – 구현체 만들기
CommentDAO 인터페이스 – 구현체 만들기

■인터페이스를 두면 좋은 점

※느슨한결합
NoticeDAO가 여러 구현체로 구현될 수 있음 . JDBC , JDBCTemplate , Mybatis , JPA
DAO를 변경하는 것이 편리함 (버전업데이트)

※인터페이스를 두면 , 약속기반으로 각 자 영역을 비동기로 개발해 나갈 수 있다.

스프링 데이터베이스 테스트하기

dependency

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>
<dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8</artifactId>
    <version>21.7.0.0</version>
</dependency>
```

```java
public class DBTest1 {

	public static void main(String[] args) throws ClassNotFoundException, SQLException {
		String driver = "oracle.jdbc.driver.OracleDriver" ;
		String url="jdbc:oracle:thin:@localhost:1521:xe";
		String user="system";
		String password="1234";


	Class.forName(driver);
	Connection conn = DriverManager.getConnection(url, user, password); // 데이터베이스의 연결을 얻는다.
	Statement stmt  = conn.createStatement(); // Statement를 생성한다.


	System.out.println( conn);
	String query = "SELECT sysdate from dual"; // 시스템의 현재 날짜시간을 출력하는 쿼리(query)
	ResultSet rs = stmt.executeQuery(query); // query를 실행한 결과를 rs에 담는다.

	// 실행결과가 담긴 rs에서 한 줄씩 읽어서 출력
	while (rs.next()) {
	    String curDate = rs.getString(1);  // 읽어온 행의 첫번째 컬럼의 값을 String으로 읽어서 curDate에 저장
	    System.out.println(curDate);       // 2022-01-11 13:53:00.0
	}

    }

}
```

```java
public class DBTest2 {

	public static void main(String[] args) throws SQLException {

		// spring jdbc 사용해서 데이터베이스 연동
	String driver = "oracle.jdbc.driver.OracleDriver" ;
		String url="jdbc:oracle:thin:@localhost:1521:xe";
		String user="system";
		String password="1234";

	DriverManagerDataSource ds = new DriverManagerDataSource();
	ds.setDriverClassName(driver);
	ds.setUrl(url);
	ds.setUsername(user);
	ds.setPassword(password);

	Connection conn = ds.getConnection(); //
	System.out.println("conn = " + conn);
	System.out.println(conn!=null);

	Statement stmt  = conn.createStatement(); // Statement를 생성한다.
	String query = "SELECT  * from member_tbl_11"; // 시스템의 현재 날짜시간을 출력하는 쿼리(query)
	ResultSet rs = stmt.executeQuery(query); // query를 실행한 결과를 rs에 담는다.

	// 실행결과가 담긴 rs에서 한 줄씩 읽어서 출력
	while (rs.next()) {
	    String field1 = rs.getString(1);  // 읽어온 행의 첫번째 컬럼의 값을 String으로 읽어서  항목1에 저장
	    String field2 = rs.getString(2);  // 읽어온 행의 첫번째 컬럼의 값을 String으로 읽어서  항목2에 저장
	    System.out.println( field1 + " " + field2 );        // 2022-01-11 13:53:00.0
	}

	}

}
```

```
public class DBTest3 {

	public static void main(String[] args) throws SQLException {
		ApplicationContext ac = new
GenericXmlApplicationContext("file:src/main/webapp/WEB-INF/spring/**/root-context.xml");
	DataSource ds = ac.getBean(DataSource.class);
	Connection conn = ds.getConnection(); //
	System.out.println("conn = " + conn);
	System.out.println(conn!=null);

	}

}
```

root-context,xml 작성

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://www.springframework.org/schema/beans
https://www.springframework.org/schema/beans/spring-beans.xsd">

	<!-- Root Context: defines shared resources visible to all other web components -->


		<!-- Root Context: defines shared resources visible to all other web components -->

	 <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
		<property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"></property>
		<property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"></property>
		<property name="username" value="System"></property>
		<property name="password" value="1234"></property>
	</bean>
</beans>
```

```java
@Component
public class TestDao {
	@Autowired
	DataSource ds;

	public void select() {
		Connection conn = null;
	PreparedStatement pstmt = null;
	ResultSet rs= null;
	String sql = "select * from  member_tbl_11";
	try {
		conn = ds.getConnection();
		pstmt = conn.prepareStatement(sql);
		rs  = pstmt.executeQuery();

		while( rs.next()){
			System.out.println( rs.getString(1));
			System.out.println( rs.getString(2));
			System.out.println( rs.getString(3));
		}
		rs.close();
		pstmt.close();
		conn.close();
			} catch (SQLException e) {
				e.printStackTrace();
	  } finally {
			close(rs, pstmt, conn);
	  }
	}


	private  void  close(AutoCloseable ...autoCloseables) {
		for( AutoCloseable  a: autoCloseables ) {
			try { if( a!= null) a.close();} catch (Exception e) {	e.printStackTrace();}

		}
	}


}
```

```java
@Controller
public class DBController {

	@Autowired
	TestDao dao;


	@RequestMapping(value = "/db", method = RequestMethod.GET)
	public String dbtest() {
		dao.select();
		return "db";
	}

}
```