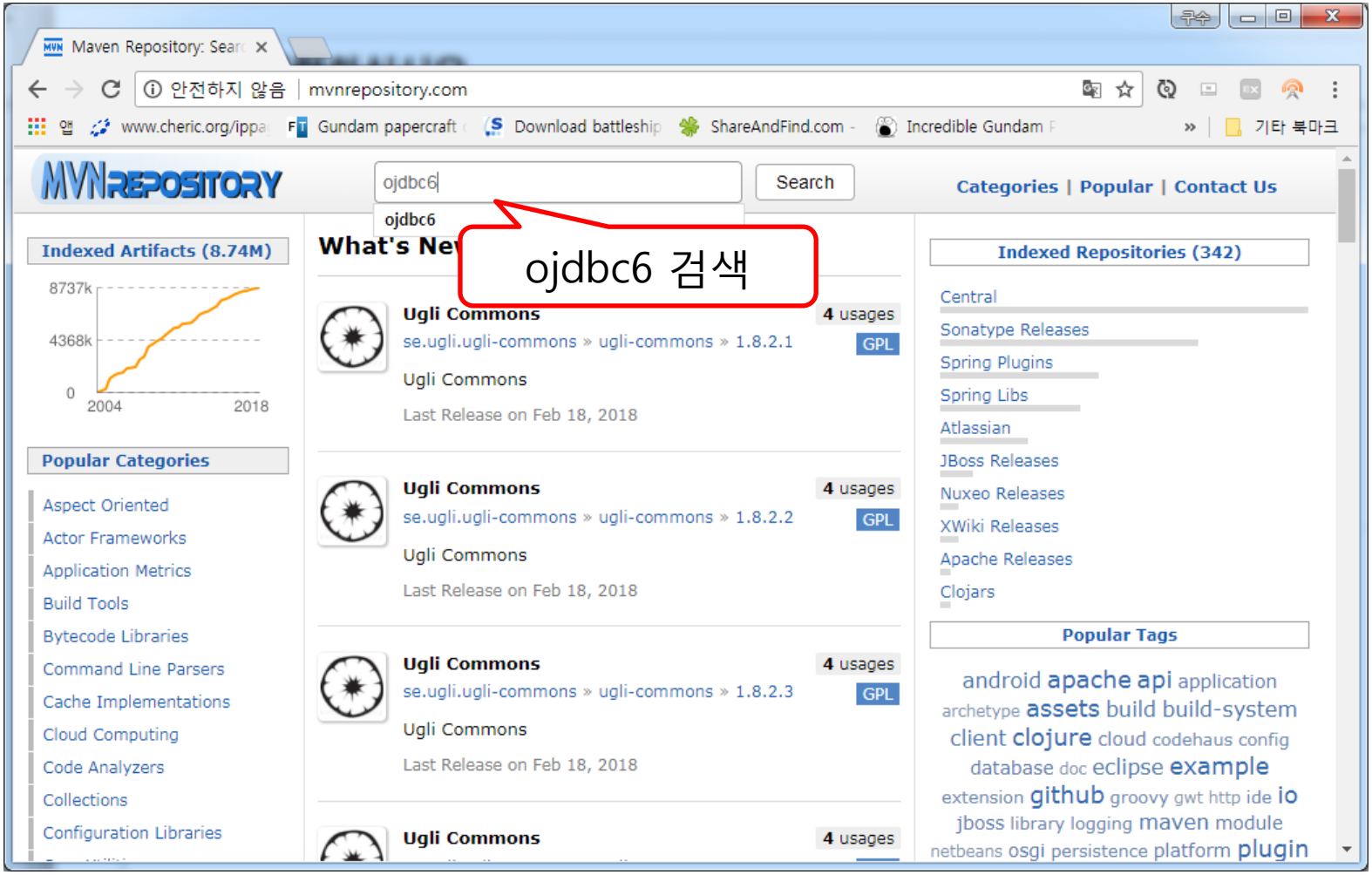
JDBC 프로그래밍
– 커넥션 : 데이터베이스 연결 –

JDBC 프로젝트 설정

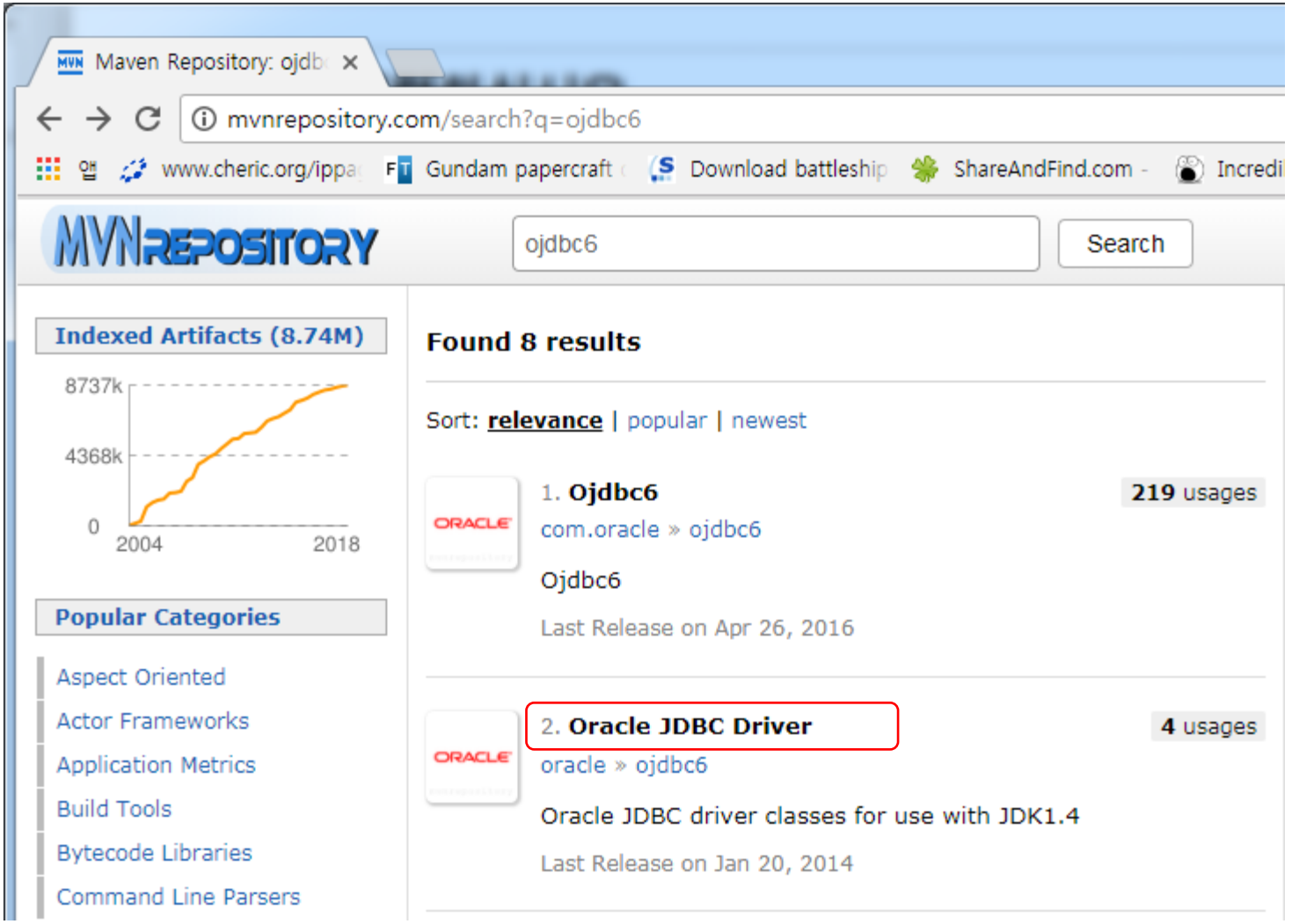
❖ 외부 라이브러리 검색 (ojdbc6)

- http://mvnrepository.com/



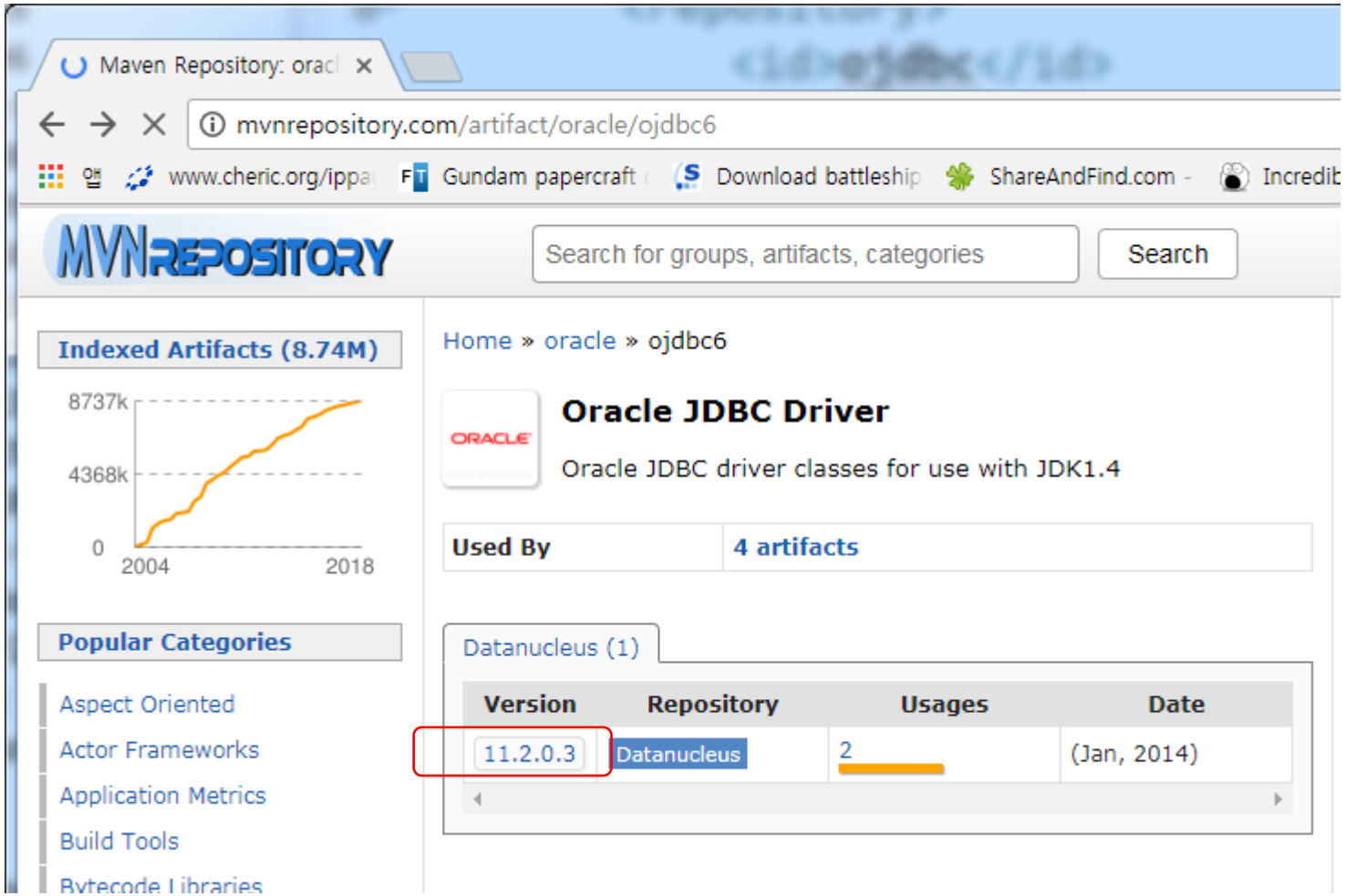
JDBC 프로젝트 설정

❖ 외부 라이브러리 검색 (ojdbc6)



JDBC 프로젝트 설정

❖ 외부 라이브러리 검색 (ojdbc6)



JDBC 프로젝트 설정

❖ 외부 라이브러리 검색 (ojdbc6)

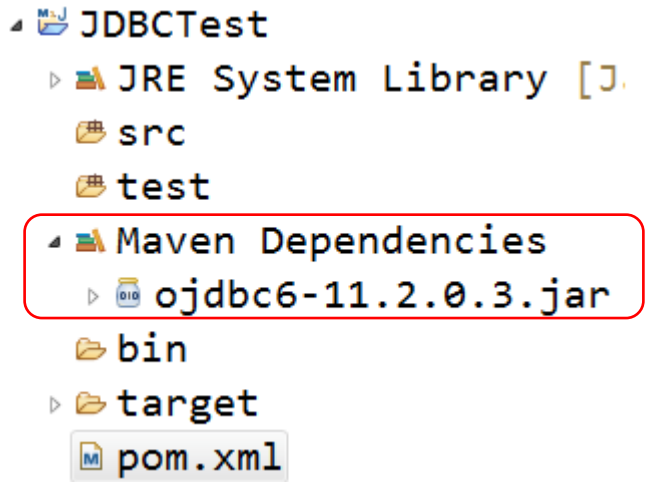
JDBC 프로젝트 설정

❖ pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" …>
    <modelVersion>4.0.0</modelVersion>
    <groupId>edu.iot</groupId>
    <artifactId>JDBCTest</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <repositories>
        <repository>
            <id>ojdbc</id>
            <url>http://www.datanucleus.org/downloads/maven2/</url>
        </repository>
    </repositories>
    <dependencies>
        <dependency>
            <groupId>oracle</groupId>
            <artifactId>ojdbc6</artifactId>
            <version>11.2.0.3</version>
        </dependency>
    </dependencies>
    :
</project>
```

JDBC 프로젝트 설정

❖ pom.xml



다운로드 받은
oracle jdbc 라이브러리

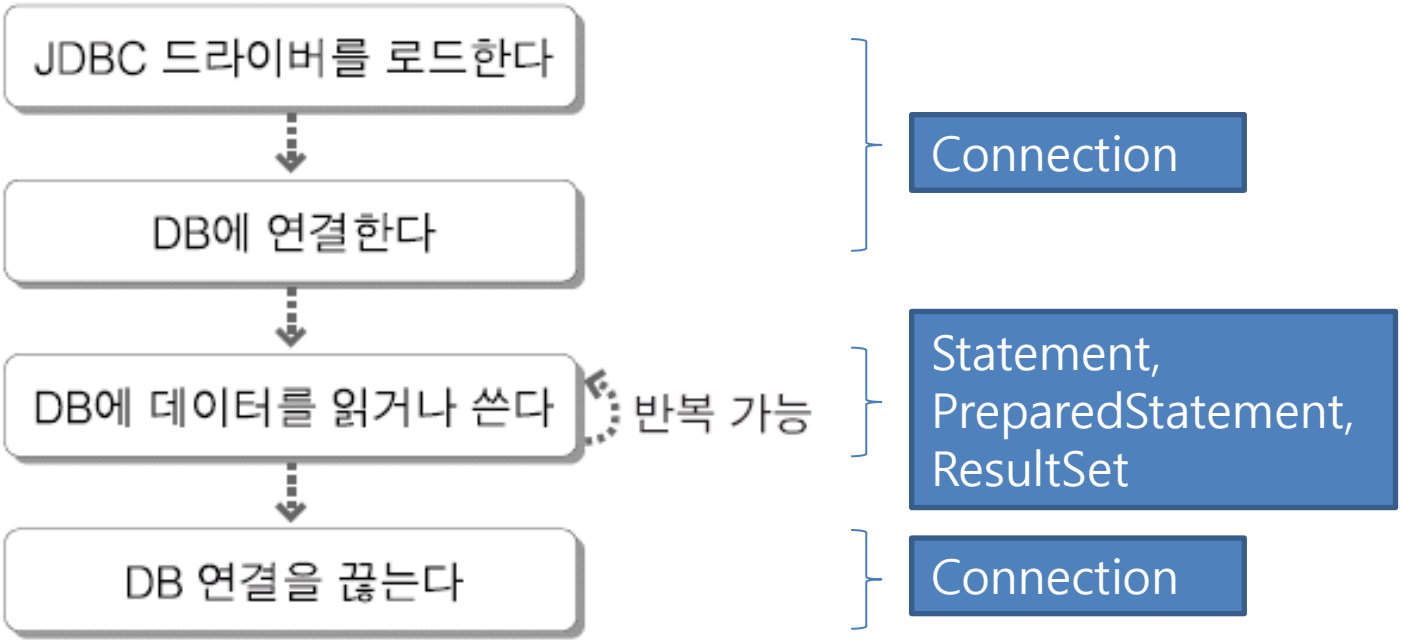
- Maven 로컬 레파지토리
  - C:\Users\<사용자ID>\.m2\repository
    - 다시 다운로드 받고 싶은 경우 해당 라이브러리를 제거 후 pom.xml 을 다시 저장

❖ JDBC 주요 클래스

- Connection
  - JDBC 드라이버 로드 및 DBMS 접속 및 해제
- Statement
  - SQL문 실행
- PreparedStatement
  - 매개변수가 있는 SQL문 실행
- ResultSet
  - SELECT 쿼리의 결과 행 집합 관리

❖ JDBC를 이용한 프로그램의 작성 방법

- 자바 프로그램에서 데이터베이스를 사용하는 4단계의 과정



Connection 관리

❖ JDBC 드라이버의 로드

- JDBC 드라이버를 로드하는 방법
  - 오라클 JDBC 드라이버의 클래스 이름을 지정

```
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    System.out.println("오라클 JDBC 드라이버 로드 성공");
} catch (ClassNotFoundException e) {
    System.err.println("드라이버 로드 실패" + e.getMessage());
}
```

Connection 관리

❖ JDBC 드라이버를 로드 프로그램

```
public class JDBCEx1 {
    public static void main(String[] args) {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            System.out.println("오라클 JDBC 드라이버 로드 성공");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Connection 관리

❖ 데이터베이스로의 연결(사용자 로그인)

- ㅇ 데이터베이스 연결 객체 생성
  - ▪ Connection 객체 – DriverManager.getConnection() 메서드의 리턴값

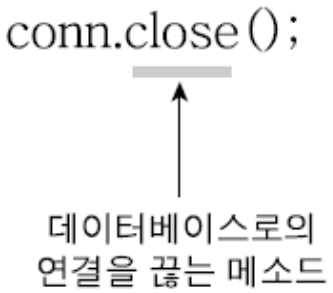
```
String url = "데이터베이스 접속 url";
Connection conn = (Connection) DriverManager.getConnection(
                                    url, "db 사용자명", " 패스워드");
```

```
// 접속 url : "jdbc드라이이버@db_서버_ip:포트번호:SID"
String url = "jdbc:oracle:thin:@localhost:1521:XE";
Connection conn = (Connection) DriverManager.getConnection(
                                        url, "hr", "hr");
```

Connection 관리

❖ 데이터베이스 연결 끊기

- 데이터베이스와 연결을 끊는 방법



  - 예외처리 필수

Connection 관리

❖ JDBC 드라이버를 로드하고 데이터베이스에 연결하는 프로그램

```
public class JDBCEx2 {
   public static void main(String[] args) {

      String url = "jdbc:oracle:thin:@localhost:1521:XE";
      try(Connection conn =
            (Connection) DriverManager.getConnection(
                                          url, "hr", "hr")) {
         System.out.println("데이터베이스에 접속했습니다.");
      } catch (SQLException e) {
         e.printStackTrace();
      }
   }
}
```

Connection 관리

❖ 접속 정보의 분리

- Source 폴더 추가
  - src/main/java
    - database.properties 파일 생성(문자셋 : UTF-8)

```
driver=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@localhost:1521:xe
username=hr
password=hr
```

❖ Properties 파일 읽기

```
public class JDBCEx3 {
   public static void main(String[] args) {

      try {
         Properties properties  = new Properties();
         String path = JDBCEx3.class.getResource(
                                 "/database.properties").getPath();
         properties.load(new FileReader(path));
         String driver = properties.getProperty("driver");
         String url = properties.getProperty("url");
         String username = properties.getProperty("username");
         String password = properties.getProperty("password");
         System.out.println(driver);

      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Connection 관리

❖ database.properties 정보로 데이터베이스 연결 클래스

```
public class Database {
   private static Connection connection;
   static {
      try {
         Properties properties  = new Properties();
         String path = Database.class.getResource(
                              "/database.properties").getPath();
         properties.load(new FileReader(path));
         String driver = properties.getProperty("driver");
         String url = properties.getProperty("url");
         String username = properties.getProperty("username");
         String password = properties.getProperty("password");

         Class.forName(driver);
         connection = (Connection) DriverManager.getConnection(
                                 url, username, password);
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
```

Connection 관리

❖ database.properties 정보로 데이터베이스 연결 클래스

```
   public static Connection getConnection() {
      return connection;
   }

   public static void close() {
      if(connection!= null) {
         try {
            connection.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
      }
   }
}
```

Connection 관리

❖ 데이터베이스 연결

```
public class JDBCEx4 {
   public static void main(String[] args) {

      Connection conn = Database.getConnection();

      System.out.println("데이터베이스 준비 완료");

      Database.close();
   }
}
```