DBCP

01.DBCP

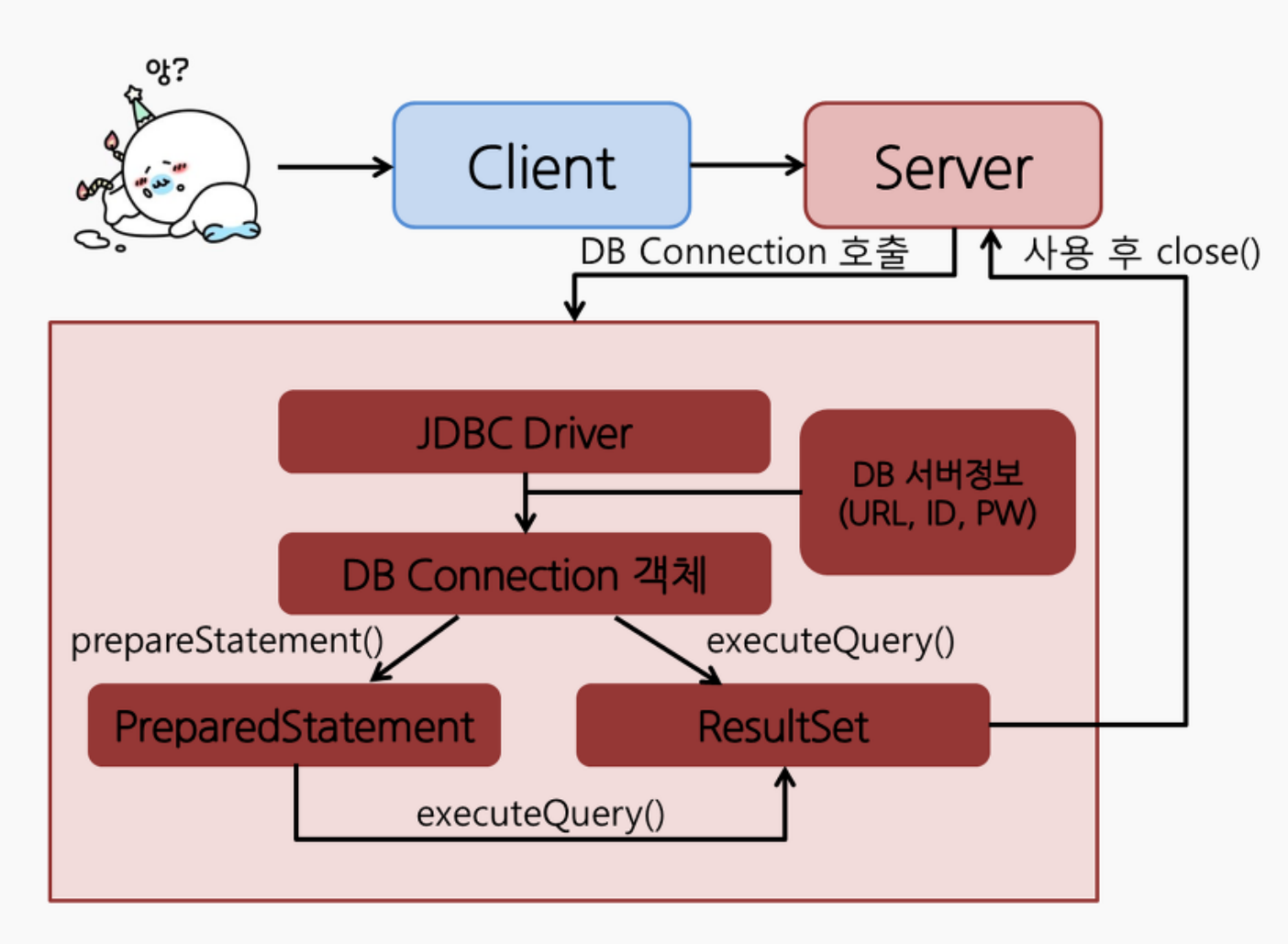
DBCP란?

- DataBase Connection Pool의 약자이며, 데이터베이스와 애플리케이션을
- 효율적으로 연결하는 **커넥션 풀 라이브러리**

DBCP 사용 이유

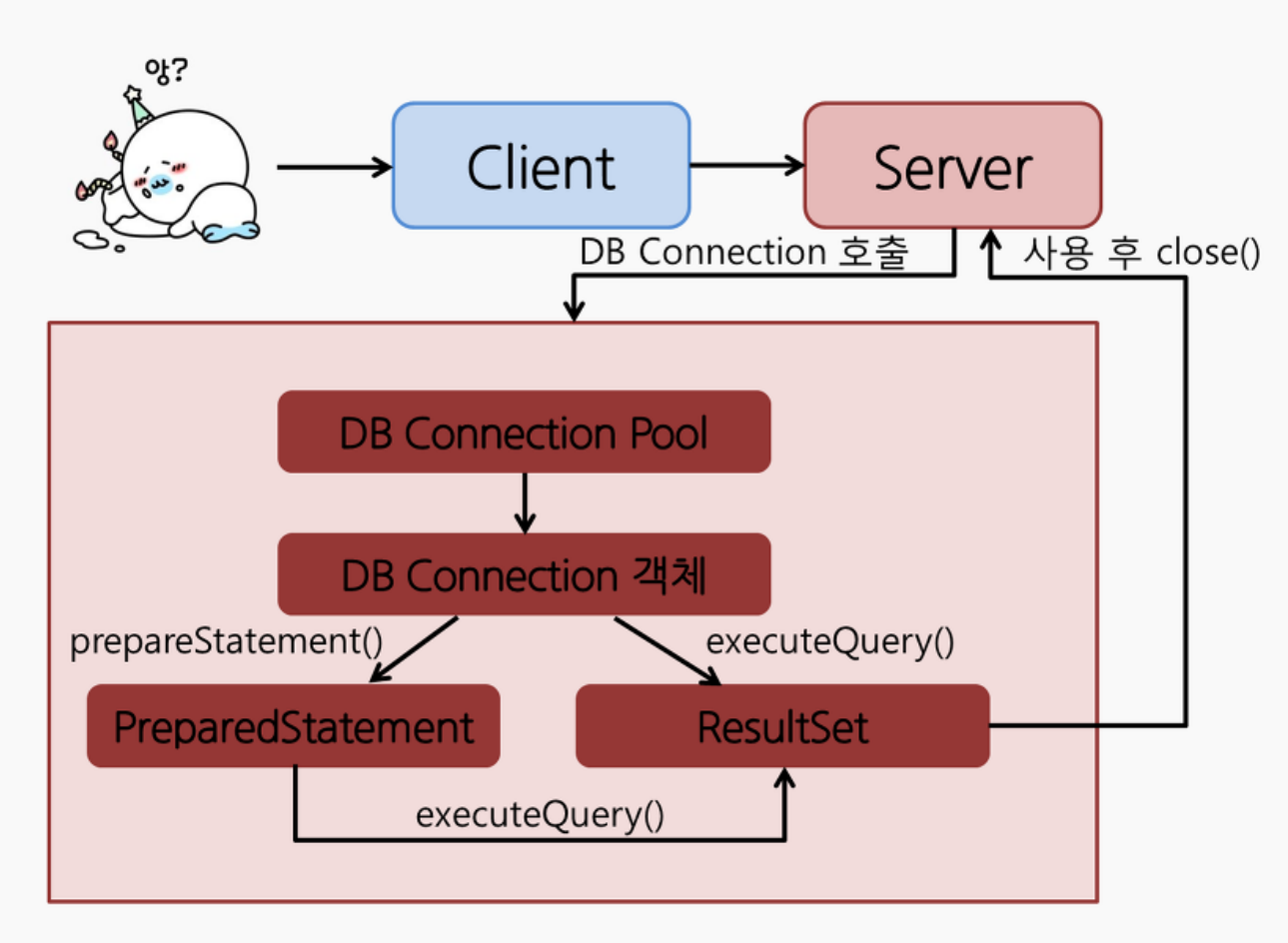
시스템에 접근하는 사용자가 많아 서버에 부하가 생기는 경우, 시스템의 상황에 따라 성능 이슈가 발생할 확률이 높고 구축한 시스템에 대해 최적화를 해야 하는 상황이 발생하게 되는데 보통 성능 최적화를 위해 테스트하고 확인하는 것 중에 하나가 DB Connection pool과 Thread 개수를 설정을 조절하는 것이다.

단일 Connection 사용 구조



① 페이지 요청(DBMS 연결 요청)이 들어왔을 때, DB Connection을 호출
② DB 접속을 위해 JDBC 드라이버를 로드
③ DB 서버 정보(서버 URL, 사용자 아이디, 비밀번호)를 입력하여 DB 연결
④ DB Connection 객체 생성
⑤ 쿼리 수행 후 결과 값 반환
⑥ DB Connection close() ☞ Connection을 닫음(메모리 낭비 방지)

Connection Pool 사용



① 페이지 요청(DBMS 연결 요청)이 들어왔을 때, **DB Connection Pool에는 이미 DB Connection 객체가 저장**되어 있다.
(WAS 실행 시 개발자가 설정한만큼의 Connection이 생성된다)
② DBCP에서 DB Connection 객체를 가져다 사용
③ 쿼리 수행 후 결과 값 반환
④ DB Connection closes() ☞ **Connection을 닫는 것이 아니라 Connection Pool로 반환**된다.

01.DBCP

필요 파일

- JDBC 드라이버 : 데이터베이스와 연결하기 위한 드라이버
- Collections : 자카르타 Pool API의 jar 파일
- DBCP : DBCP API 관련 jar 파일
- Pool : Pool API가 사용하는 자카르타 Collection API의 jar 파일

관련 jar 파일을 다운로드

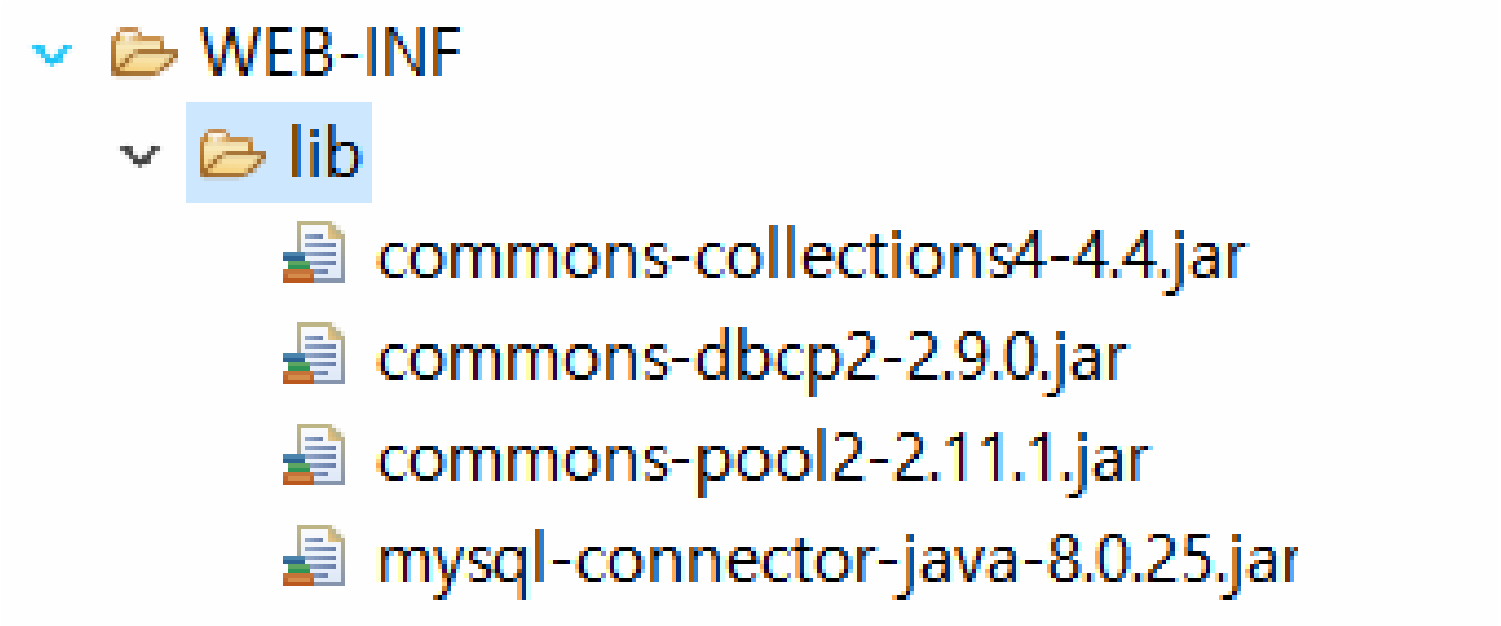
1) DBCP
commons.apache.org/proper/commons-dbcp/download_dbcp.cgi

2) Collections
commons.apache.org/proper/commons-collections/download_collections.cgi

3) pool
commons.apache.org/proper/commons-pool/download_pool.cgi

01.DBCP

- lib에 jar 붙이기



01.DBCP

- context.xml

```
 <Resource
name="jdbc/MysqlDB"
auth="Container"
type="javax.sql.DataSource"
driverClassName="com.mysql.cj.jdbc.Driver"
username="root"
password="1234"
url="jdbc:mysql://localhost:3306/testdb?serverTimezone=UTC"
maxWait="5000"
/>
```

auth : 컨테이너를 자원 관리자로 기술
name : JDBC이름, 변경 가능
driverClassName : JDBC 드라이버
type : 웹에서 이 리소스를 사용할 때 DataSource로 리턴됨
username : 접속계정
password : 접속할 계정 비밀번호
loginTimeout : 연결 끊어지는 시간
maxActive : 최대 연결 가능한 Connection수 (기본 20개)
maxIdle : Connection pool 유지를 위해 최대 대기 connection 숫자
maxWait : 사용 가능한 커넥션이 없을 때 커넥션 회수를 기다리는 시간 (1000 = 1초)
testOnBorrow : db에 test를 해볼 것인지

01.DBCP

- context.xml

```
<resource-ref>

  <description>jsptest db</description>
  <res-ref-name>jdbc/MysqlDB</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Containter</res-auth>

</resource-ref>
```

description : 설명
res-ref-name : JDBC 이름, <Resource>의 name 부분과 동일하게 입력
res-type : <Resource>의 type 부분과 동일하게 입력
res-auth : <Resource>의 auth 부분과 동일하게 입력

01.DBCP

- Dbconntest.jsp

```
%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import = "java.sql.*,javax.sql.*,javax.naming.*" %>
<%
   Connection conn = null;

   try{
         Context initCtx = new InitialContext();
         Context envCtx = (Context) initCtx.lookup("java:comp/env");
         DataSource ds= (DataSource)envCtx.lookup("jdbc/MysqlDB");
         conn = ds.getConnection();
         PreparedStatement pstmt = conn.prepareStatement("select * from stdtbl");
         ResultSet rs = pstmt.executeQuery();

   out.println("<h3>연결되었습니다.</h3>");
   }catch(Exception e){

   }finally{
   conn.close();
   }
   %>
```

01.DBCP

```
Context initCtx = new InitialContext();
//InitialContext 객체 initCtx를 생성

Context envCtx = (Context) initCtx.lookup("java:comp/env");
//initCtx의 lookup("java:comp/env")메소드를 사용해 큰따옴표안에 기술된 이름
"java:comp/env"에 해당하는 객체를 찾아서 envCtx 변수에 넣는다

DataSource ds= (DataSource)envCtx.lookup("jdbc/MysqlDB");
// lookup("java:comp/env")메소드를 사용해 "jdbc/DB이름"를 가지고 객체를 얻어내서
DataSource 객체 타입으로 형 변환 후 ds 변수에 저장

conn = ds.getConnection(); //ds 객체의 getConnection()메소드를 사용해서 커넥션
풀로부터 커넥션 객체를 얻어내어 conn 변수에 저장한다. conn객체를 사용해서 DB와 연동
```