МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Расчётно-графическое задание

по дисциплине «»

«Шаблон - пул объектов»

Факультет: АВТ

Преподаватель: Романов Е.Л.

Группа:

Выполнил:

Новосибирск, 2014

Оглавление

ВВЕДЕНИЕ 3
УЧЕБНО-МЕТОДИЧЕСКИЙ МАТЕРИАЛ ПО ШАБЛОНУ 3
ПРАКТИКА ПРИМЕНЕНИЯ ШАБЛОНА 4
ОПИСАНИЕ РАЗРАБОТКИ 6
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 8
СПИСОК ЛИТЕРАТУРЫ 9
ПРИЛОЖЕНИЕ ..... **Ошибка! Закладка не определена.**

ВВЕДЕНИЕ

Шаблон - пул объектов, может значительно повысить производительность системы; его использование наиболее эффективно в ситуациях, когда создание экземпляров некоторого класса требует больших затрат, объекты в системе создаются часто, но число создаваемых объектов в единицу времени ограничено.

Примером паттерна, в реальной жизни может служить боулинг, а точнее, когда вы должны сменить вашу обувь при посещении боулинг-клуба. Полка с обувью - прекрасный пример пула объектов. Когда вы хотите играть, вы берете с неё пару. А после игры, вы возвращаете обувь обратно на полку. Столовая, также может служить иллюстрацией работы шаблона, подходя за едой вы сначала берете поднос, а после вы возвращаете его, и он проходит процедуру "очистки", и возвращается в состояние готовности.

УЧЕБНО-МЕТОДИЧЕСКИЙ МАТЕРИАЛ ПО ШАБЛОНУ

Объектный пул (англ. object pool) - порождающий шаблон проектирования, набор инициализированных и готовых к использованию объектов. Когда системе требуется объект, он не создаётся, а берётся из пула. Когда объект больше не нужен, он не уничтожается, а возвращается в пул [1]. Если в пуле нет ни одного свободного объекта, возможна одна из трёх стратегий:

1. Расширение пула.
2. Отказ в создании объекта, аварийная остановка.
3. В случае многозадачной системы, можно подождать, пока один из объектов не освободиться.

Желательно, чтобы все многократно используемые объекты, свободные в некоторый момент времени, хранились в одном и том же пуле объектов. Тогда ими можно управлять на основе единой политики. Для этого класс Object Pool проектируется с помощью паттерна Singleton. Архитектура паттерна Singleton основана на идее использования глобальной переменной, имеющей следующие важные свойства:

- Такая переменная доступна всегда. Время жизни глобальной переменной - от запуска программы до её завершения.
- Предоставляет глобальный доступ, то есть, такая переменная может быть доступна из любой части программы.

Однако, использовать глобальную переменную некоторого типа непосредственно невозможно, так как существует проблема обеспечения единственности экземпляра, а именно, возможно создание нескольких переменных того же самого типа (например, стековых).

Для решения этой проблемы паттерн Singleton возлагает контроль над созданием единственного объекта на сам класс. Доступ к этому объекту осуществляется через статическую функцию-член класса, которая возвращает указатель или ссылку на него. Этот объект будет создан только при первом обращении к методу, а все последующие вызовы просто возвращают его адрес. Для обеспечения уникальности объекта, конструкторы и оператор присваивания объявляются закрытыми [2].

Часто встречаемые ошибки (ловушки) при написании кода шаблона - пул объектов :

1. После того, как объект возвращен, он должен вернуться в состояние, пригодное для дальнейшего использования. Если объекты после возвращения в пул оказываются в неправильном или неопределённом состоянии, такая конструкция называется объектной клоакой (англ. object cesspool).
2. Повторное использование объектов также может привести к утечке информации. Если в объекте есть секретные данные (например, номер кредитной карты), после освобождения объекта эту информацию надо затереть.

ПРАКТИКА ПРИМЕНЕНИЯ ШАБЛОНА

Wikipedia дает три примера использования данного шаблона [1]:

- Информация об открытых файлах в DOS.
- Информация о видимых объектах во многих компьютерных играх (хорошим примером является движок DOOM). Эта информация актуальна только в течение одного кадра; после того, как кадр выведен, список опустошается.
- Компьютерная игра для хранения всех объектов на карте, вместо того, чтобы использовать обычные механизмы распределения памяти, может завести массив такого размера, которого заведомо хватит на все объекты, и свободные ячейки держать в виде связного списка. Такая конструкция повышает скорость, уменьшает фрагментацию памяти и снижает нагрузку на сборщик мусора (если он есть).

Я же использовал данный шаблон в Web-приложении. Как известно, технология распределенного программирования на основе сервлетов и JSP-страниц позволяет динамически формировать содержание интернет-сайтов при помощи информации, хранящейся в базах данных. Очень часто программисты не уделяют должного внимания скорости соединения и взаимодействию с базой данных в проектируемых ими программах. Так, на каждый запрос от пользователя создается соединение с базой данных, на что тратятся дополнительные ресурсы и время. Если же выполняются короткие запросы, то порой процесс поиска в базе данных может оказаться значительно короче, чем открытие самого соединения. Использование ограниченного количества обращений к базе данных для всех запросов от пользователей позволяет в значительной степени улучшить производительность web-сервера и самой базы данных. В связи с тем, что технология сервлетов позволяет программистам хранить информацию между запросами от пользователей, использование пула соединений может решить проблему быстродействия современных, динамических интернет-сайтов [3].

Для доступа к реляционным базам данных в java используется технология JDBC. Для того чтобы выполнить SQL-запрос, получить данные из базы или изменить их, необходимо открыть соединение к базе данных и получить объект класса Connection, на основе которого и выполняются любые действия с базой. Типичная последовательность действий при работе с JDBC выглядит примерно следующим образом:

• загрузить драйвер базы данных в память виртуальной машины;
• получить соединение (Connection) с базой данных;
• подготовить выражение-запрос (Statement);
• выполнить запрос;
• разобрать полученные данные (ResultSet).

Простейший пул соединений представляет собой класс, объединяющий набор объектов JDBC Connection и методы доступа к ним. По своей сути это - контейнер с простейшим интерфейсом, который реализует механизм управления соединениями с базой данных. Размер такого хранилища определяется программистом и может варьироваться в зависимости от предоставленных ресурсов.

ОПИСАНИЕ РАЗРАБОТКИ

Был разработан класс DBConnectionPool (на рисунке 1 представлена диаграмма класса, код класса располагается в приложение).

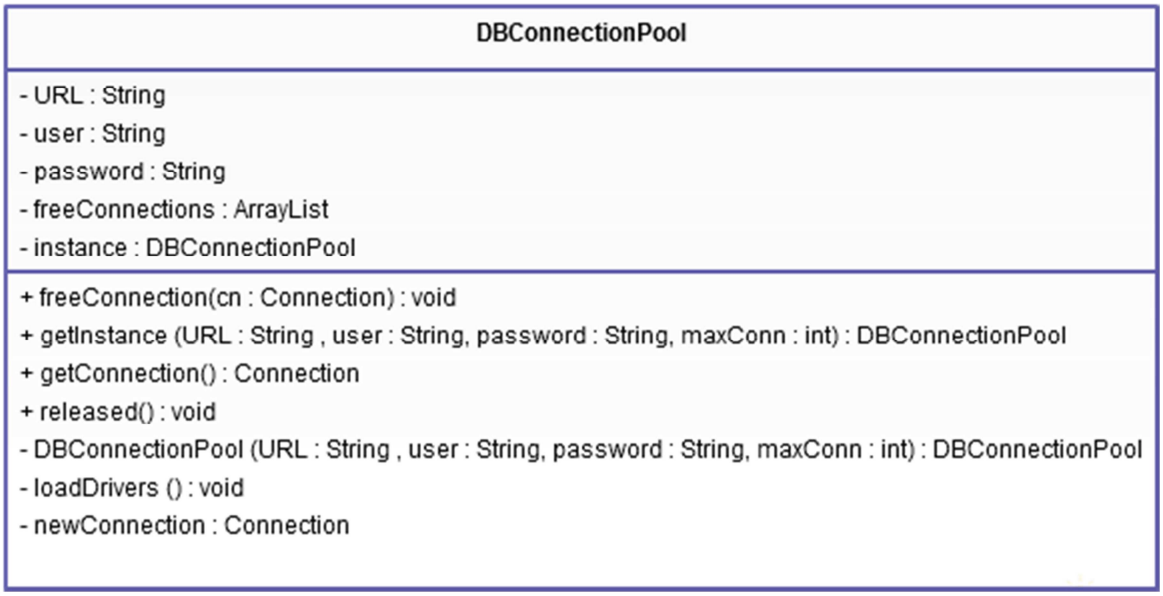

Рисунок 1 - Диаграмма класса DBConnectionPool

Конструктор DBConnectionPool объявлен как private, что защищает его от вызовов вне собственного класса. Данное ограничение, позволяет использовать в полной мере возможности паттерна "Одиночка", возможности которого, будут полностью рассмотрены в контексте использования метода getInstance(). В листинге 1 представлен конструктор DBConnectionPool.

```
private DBConnectionPool(String URL, String user, String
password, int maxConn) {
        this.URL = URL;
        this.user = user;
        this.password = password;
        this.maxConn = maxConn;
        freeConnections=new ArrayList();
        loadDrivers(); //регистрация драйвера
    }
```

Листинг 1 – Конструктор класса DBConnectionPool

В результате вызова конструктора, происходит инициализация основных переменных экземпляра, создание потока вывода сообщений, а также, регистрация JDBC-драйвера для конкретной базы данных, за это отвечает вызываемый метод loadDrivers().

Статический метод класса getInstance() полностью контролирует создание экземпляра класса DBConnectionPool, предоставляя различным запросам

ссылку только на один единственный объект типа DBConnectionPool. Это классическая модель одиночки, именно такой подход, следует использовать в тех случаях, когда создание и инициализация экземпляра требует значительных ресурсов, а результат не всегда востребован. В листинге 2 представлен метод getInstance().

```
public static synchronized DBConnectionPool getInstance(String
URL, String user, String password, int maxConn) {
        if (instance == null) {
            instance = new DBConnectionPool(URL, user, password,
                                                    maxConn);
        }
        return instance;
    }
```

Листинг 2 – Метод getInstance() класса DBConnectionPool

Метод getConnection() предоставляет возможность получить соединение с базой данных в виде объекта Connection. Внутри метода, проверяется содержимое контейнера, представляющего собой объект класса ArrayList. В том случае, когда контейнер содержит элементы, из него извлекается одно из доступных соединений и возвращается объекту-клиенту. Если же, свободных соединений в контейнере нет, тогда создается новое соединение, которое также возвращается объекту-клиенту.

В методе getConnection() существует проверка на доступность соединения. Если возникает исключение, то процесс получения соединения будет рекурсивно повторяться до тех пор, пока ресурсы контейнера не истощатся, после чего, будет создан новый экземпляр объекта Connection.

В том случае, если контейнер не содержит элементов, программа автоматически создает новое соединение при помощи метода newConnection(). Для этого, используются URL-идентификатор, имя пользователя и пароль. Метод возвращает ссылку на объект Connection.

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Результаты небольшого тестирования :

Таблица 1 - Результаты тестирования

Запросов нового объекта	Один поток, без пула (сек.)	Один поток, с пулом (сек.)	25 задач, без пула	25 задач, с пулом
1 000	0,002	0,003	0,027	0,012
10 000	0,010	0,001	0,272	0,039
25 000	0,030	0,003	0,609	0,189
50 000	0,048	0,006	1,285	0,287
1 000 000	0,959	0,125	27,956	8,345

Как видно из тестирования, использовать пул при одном потоке и 1 000 запросов не выгодно. Во всех остальных случаях, время затраченное на запросы с шаблоном - пул объектов значительно меньше, чем без него. На числе запросов 1 млн., разница во времени для одного потока достигла 0,834 сек., для задач разница составила 19,61 сек.

СПИСОК ЛИТЕРАТУРЫ

1 Объектный пул. Материал из Википедии — свободной энциклопедии - [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Объектный_пул.

2 Паттерн object Pool (пул объектов) - [Электронный ресурс]. Режим доступа: http://cpp-reference.ru/patterns/creational-patterns/singleton/

3 Курняван Б. Создание web-приложений на языке Java с помощью сервлетов, JSP и EJB. - М.: «Лори», 2005. - 880 с.