Здравствуйте!

Настоящая инструкция представляет собой руководство по подготовке к запуску приложения ThreadsApp и инструкцию пользователя. Руководство состоит из двух частей: первая часть – подготовка СУБД MySQL и создание базы данных приложения, вторая часть – запуск приложения и третья часть – инструкция по работе с приложением.

# Подготовка СУБД MySQL и создание базы данных приложения

Перед запуском необходимо создать базу данных приложения. В разделе описана последовательность действий с использованием MySQL Workbench 6.3 CE.

**Шаг 1 – подключение к MySQL**

Откройте MySQL Workbench как на рисунке 1 и выберете подключение к локальной машине (localhost), согласно рисунку 1.

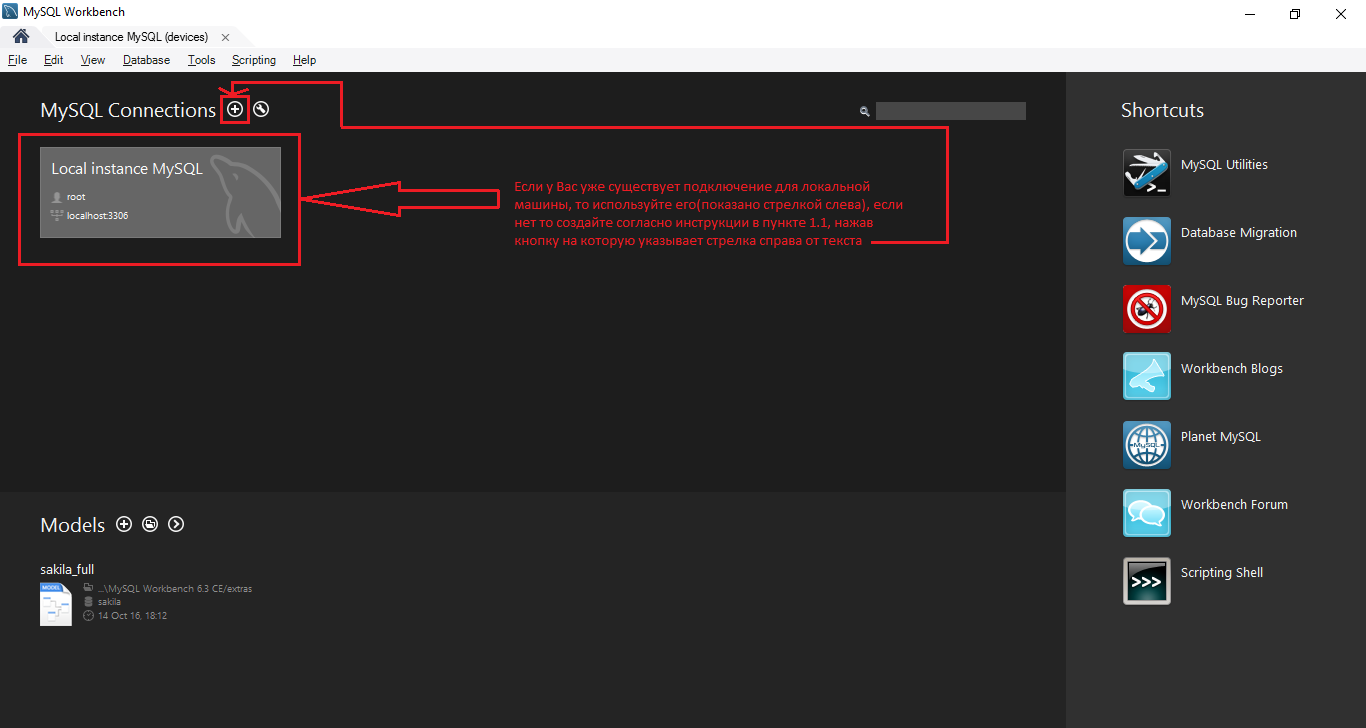


Рисунок 1. Выбор подключения к MySQL

**Шаг 2 – создание базы данных**

На рисунке 2 представлена последовательность создания базы данных devices, с которой будет работать приложение:

1. Нажать кнопку «Добавить базу данных», откроется окно как на рисунке 2;
2. Ввести название базы данных;
3. Выбрать кодировку базы данных согласно рисунку 2 (latin1 – default collation);
4. Нажать кнопку «**Apply»**;

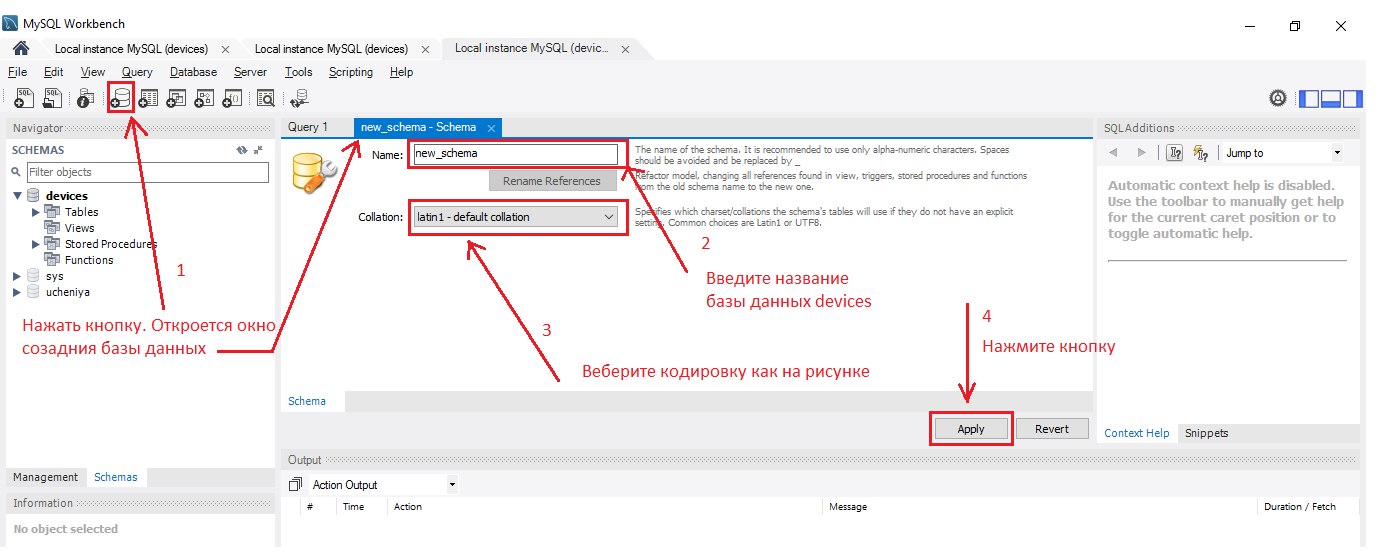


Рисунок 2. Создание базы данных приложения

Откроется окно как на рисунке 3, где необходимо нажать кнопку «**Apply».**

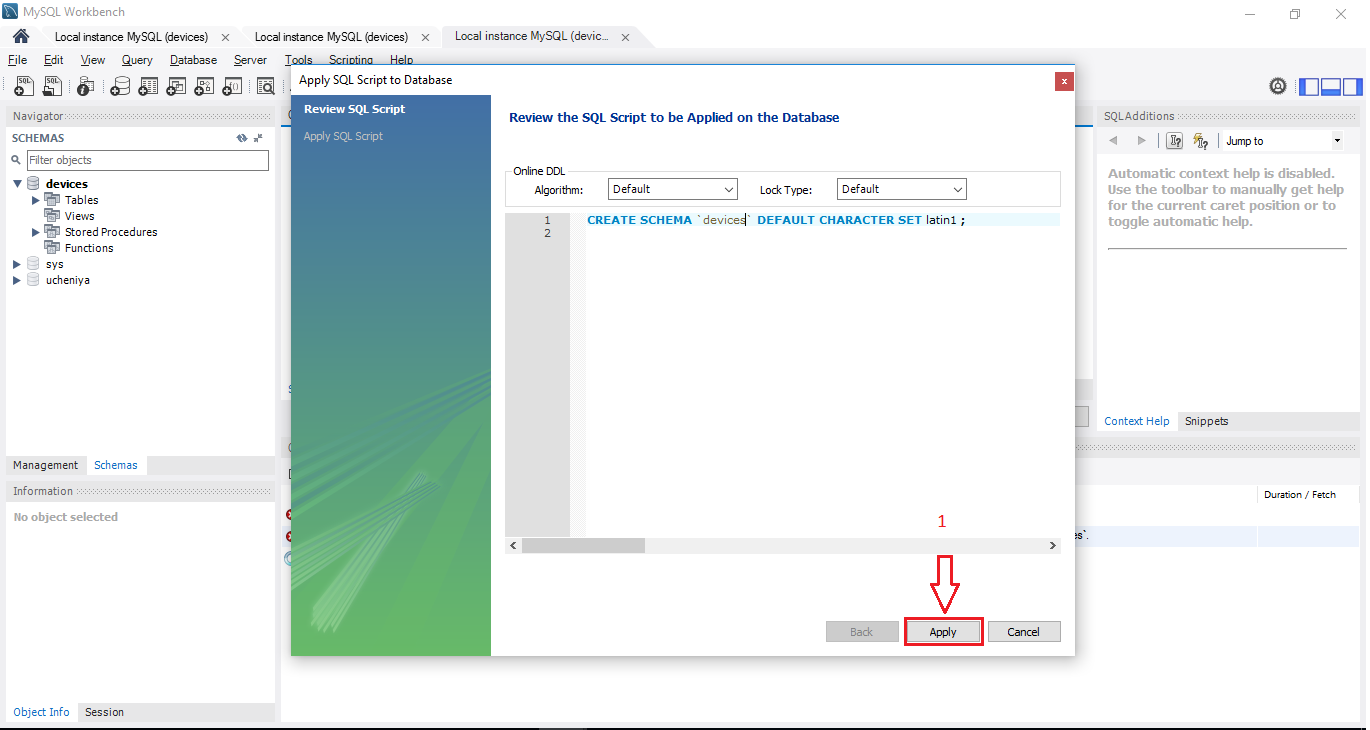


Рисунок 3. Выполнение скрипта создания базы данных devices

**Шаг 3 – создание таблиц**

После успешного создания базы данных выделить базу данных devices и:

1. Нажать кнопку «Создать новый SQL-скрипт для выполнения запросов» как на рисунке 4;
2. В открывшееся окно добавить код создания трех таблиц для приложения из листинга 1;
3. Нажать кнопку «Выполнить запрос»;
4. Нажать кнопку обновить базу;

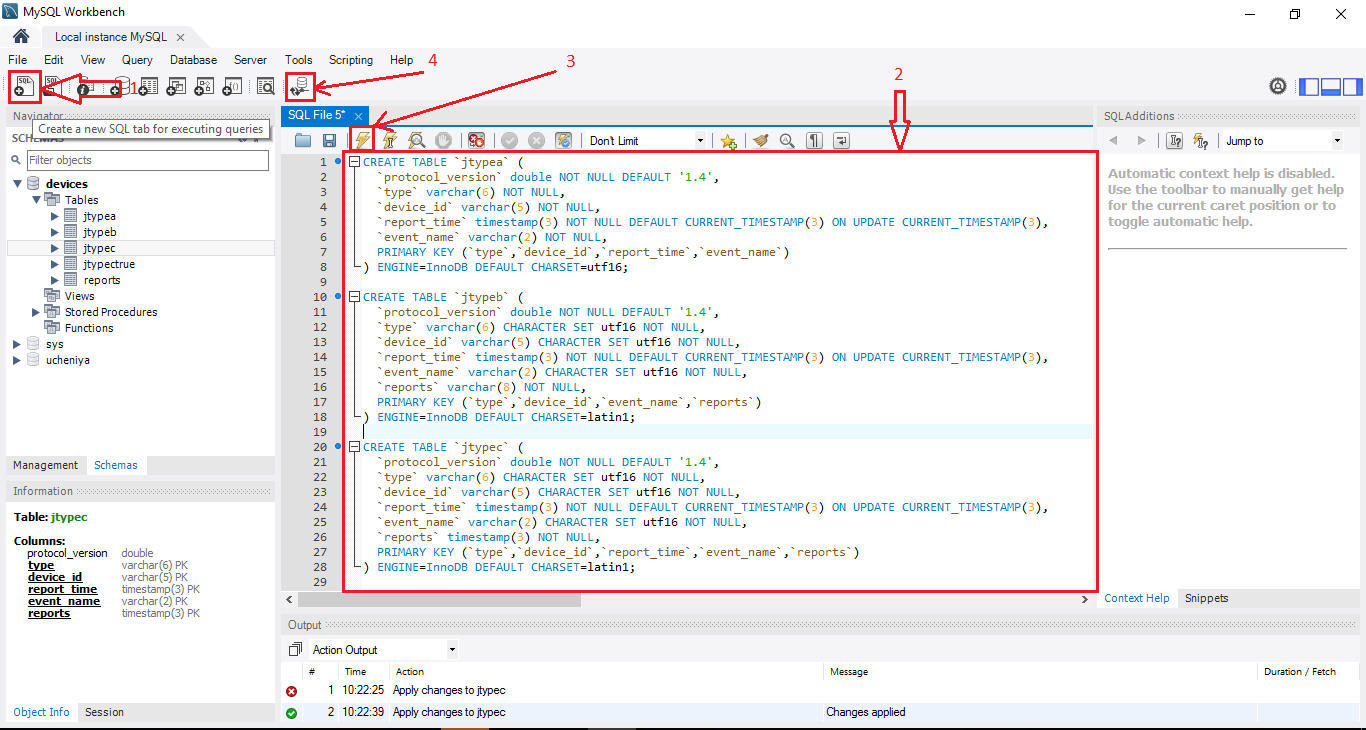


Рисунок 4. Создание таблиц

**Листинг 1**

CREATE TABLE `jtypea` (

`protocol\_version` double NOT NULL DEFAULT '1.4',

`type` varchar(6) NOT NULL,

`device\_id` varchar(5) NOT NULL,

`report\_time` timestamp(3) NOT NULL DEFAULT CURRENT\_TIMESTAMP(3) ON UPDATE CURRENT\_TIMESTAMP(3),

`event\_name` varchar(2) NOT NULL,

PRIMARY KEY (`type`,`device\_id`,`report\_time`,`event\_name`)

) ENGINE=InnoDB DEFAULT CHARSET=utf16;

CREATE TABLE `jtypeb` (

`protocol\_version` double NOT NULL DEFAULT '1.4',

`type` varchar(6) CHARACTER SET utf16 NOT NULL,

`device\_id` varchar(5) CHARACTER SET utf16 NOT NULL,

`report\_time` timestamp(3) NOT NULL DEFAULT CURRENT\_TIMESTAMP(3) ON UPDATE CURRENT\_TIMESTAMP(3),

`event\_name` varchar(2) CHARACTER SET utf16 NOT NULL,

`reports` varchar(8) NOT NULL,

PRIMARY KEY (`type`,`device\_id`,`event\_name`,`reports`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `jtypec` (

`protocol\_version` double NOT NULL DEFAULT '1.4',

`type` varchar(6) CHARACTER SET utf16 NOT NULL,

`device\_id` varchar(5) CHARACTER SET utf16 NOT NULL,

`report\_time` timestamp(3) NOT NULL DEFAULT CURRENT\_TIMESTAMP(3) ON UPDATE CURRENT\_TIMESTAMP(3),

`event\_name` varchar(2) CHARACTER SET utf16 NOT NULL,

`reports` timestamp(3) NOT NULL,

PRIMARY KEY (`type`,`device\_id`,`report\_time`,`event\_name`,`reports`)

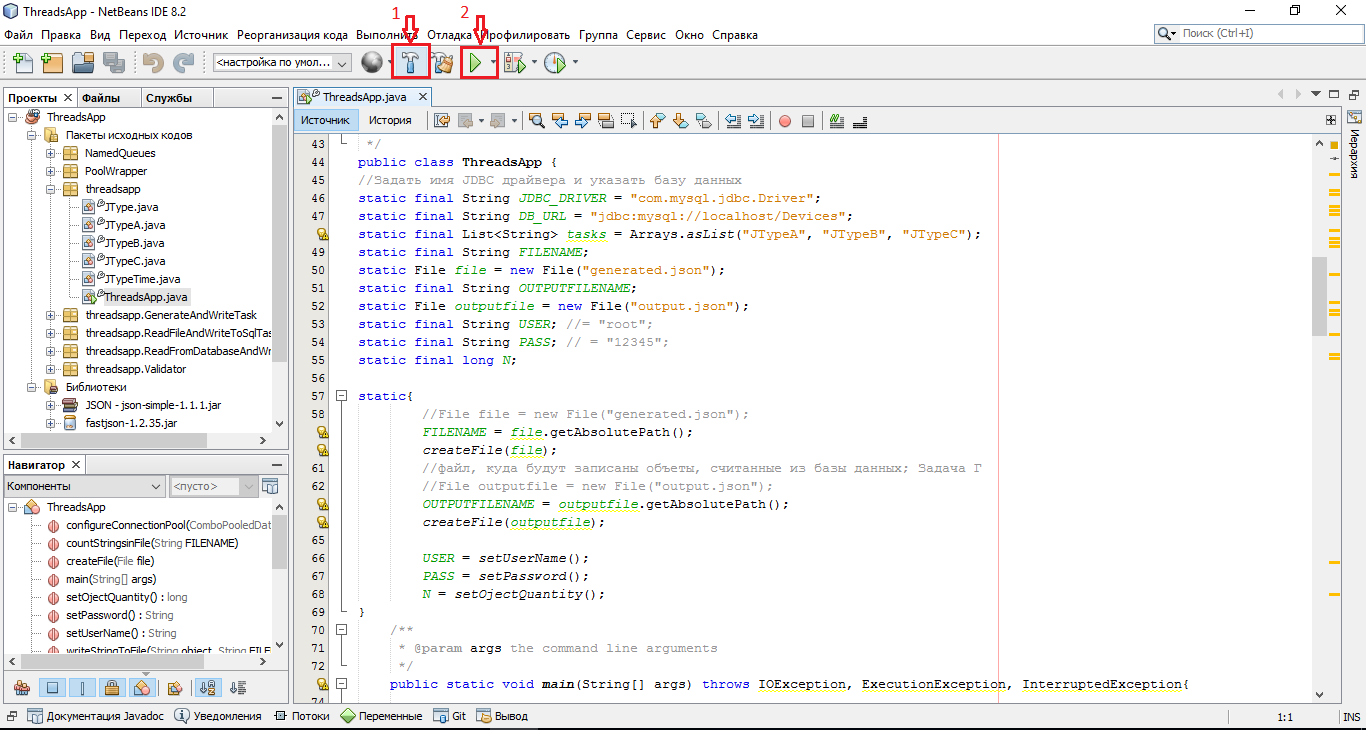
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**Конец листинга 1**

Приложение использует 3 таблицы базы данных для хранения объектов каждого типа в отдельной. Следующий шаг – запуск приложения.

# Запуск приложения

1. Скачать проект из репозитория на GitHub и открыть в NetBeans;
2. Собрать проект, нажав кнопку **«Собрать проект»** как на рисунке 5 (стрелка 1);
3. Если Вы желаете запустить проект из NetBeans, то нажмите кнопку **«Запустить проект»** как на рисунке 5 (стрелка 2);
4. Если Вы желаете запустить проект из командной строки, то перейдите в папку с проектом … \ThreadsApp-toTomskLab\dist и перейдите к пункту 5);
5. Выполните команду java -jar "ThreadsApp.jar" ;
6. Для работы с приложением перейдите к разделу 3.

Рисунок 5. Сборка и запуск проекта в NetBeans

# Работа с приложением

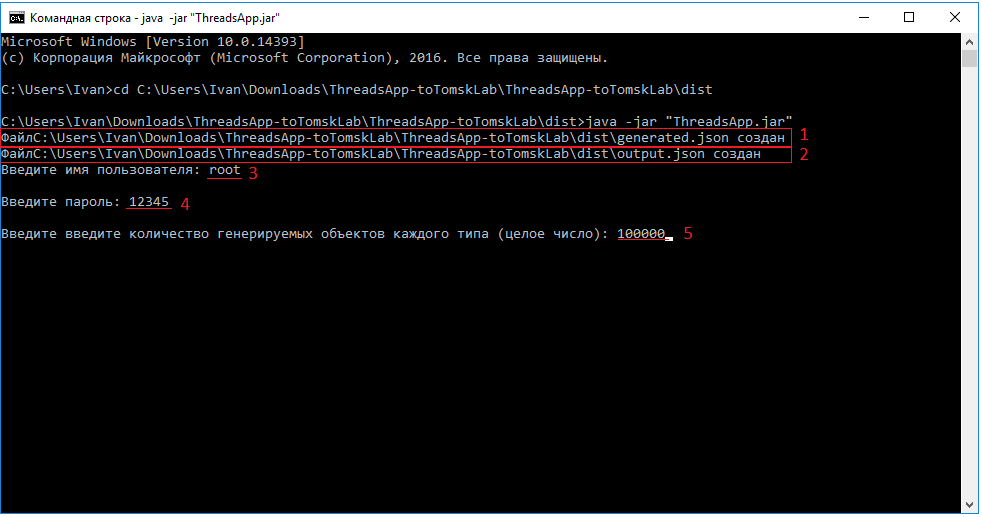


Рисунок 6. Работа с приложением

При первом запуске приложения будут созданы 2 файла: generated.json и output.json.

1. generated.json – содержит преобразованные в строки JSON-объекты, которые сгенерированы в рамках задачи а) – генерация JSON-объектов трёх типов.
2. output.json – содержит преобразованные в строки JSON-объекты, которые считаны из базы данных в рамках задачи г) – чтение сгенерированных JSON-объектов из базы данных.

Если пользователь захочет заново запустить программу, то ему будет предложен выбор: записывать вновь сгенерированные/прочитанные JSON-объекты в существующие файлы из пунктов 1) и 2) или создать новые файлы.

1. Ввести имя пользователя базы данных;
2. Ввести пароль базы данных;
3. Для удобства тестирования пользователю предлагается ввести количество генерируемых объектов каждого типа, т.е. если пользователь вводит 1000, то будет сгенерировано 1000 JSON-объектов типа A, 1000 JSON-объектов типа B и 1000 JSON-объектов типа C.

# Используемые инструменты

Для создания программы использовались библиотеки:

* fastJson 1.2.35 – для парсинга и создания JSON-объектов;
* c3p0 - последняя версия, для создания пула соединений с базой данных;
* Драйвер JDBC MySQL - mysql-connector-java-5.1.23

Используемая версия СУБД MySQL Community Server (GPL) version 5.7.16;

SQL код создания базы данных.