Типа титульник

Оглавление

Элементы оглавления не найдены.

# Введение

Данное пособие (методические работы) описывают работу по созданию клиент-серверного приложения на базе операционной системы Android. Данное пособие состоит из трех частей, в которых Вы:

* напишете приложение для сервера Heroku (ввод данных, PosgregeSQL, Java, Groovy, HTML, CSS);
* напишете приложение для Android (Java, MySQL, Android SDK, Gradle, JSON, OkHTTP, XML)
* соедините все в одну часть, получив при этом приложение по (бля придумать че за приложение)

Порядок работ:

Глава 1. Архитектура.

1. Git. Учимся записывать и выгружать данные с репозитория.
2. Описание архитектуры проекта.
3. Локальное позиционирование Базы Данных MySQL.
4. JDBC. Будущая синхронизация баз данных

Глава 2. Серверная часть.

1. Серверы, что це таке.
2. Установка и знакомство c Heroku.
3. Рассматриваем начальную хуету, которую сгенерил Heroku.
4. POST/GET методы. Дохуевертим ваш код.

Глава 3. Клиентская часть.

1. Установка и знакомство c Android Studio.
2. Знакомство с паттернами программирования Android – MVC, MVVC и FLUX.
3. Первый активити. (логин накидаем).
4. MenuDrawer. (задание по вариантам).
5. Добаляем базу данных для хранения информации. Adapters. RecycleView.

# Задание.

Создать базу данных. Создать проект на JAVA с соединением к базе данных. Реализовать методы:

* Создания таблицы – CREATE
* Удаление таблицы – DROP
* Вставки данных – INSERT
* Удаление данных - DELETE
* Выборки всех данных – SELECT \* FROM table;
* Выборки конкретных данных – SELECT \* FROM table WHERE …

Создать тестовый класс на основе Junit 4, который протестирует функционал данного проекта в следующем порядке: создание таблицы, вставка данных 1, вставка данных 2, выборка всех данных, выборка определенных данных, удаление данных, удаление таблицы.

Залить данный файл на Git репозиторий.

В таблице должно быть не менее 100 записей, уникальность по одному или несколькими полям. Таблица имеет следующий формат:

* id – целочисленного типа, авто-инкрементируется;
* email – данные текстовые;
* password - данные тектовые;
* date – данные текстовые, формата “чч:мм дд.мм.гггг”.

Заполнить таблицу данными можно непосредственно из кода.

# Глава 1. Архитектура.

# Image result for gitGit. Учимся записывать и выгружать данные с репозитория

GIT – система версионного контроля, которая находится в свободном доступе. В данный момент, существует много разновидностей гита, которые позволяют разработчикам хранить данные в удаленном месте, т.н. облаке, изменять данные, не боясь в случае неправильных правок откатить изменения.

На данный момент самыми популярными версиями гита являются^

Githhub –

Bitbucket –

Для данной работы нам будет необходимо несколько команд: git init, git push, git pull, git commit, git branch.

$ cd /path/to/my/codebase

$ git init (1)

$ git add . (2)

$ git commit -m “comment” (3)

$ git push origin master

1. Создает директорию /path/to/my/codebase/.git
2. Добавляет все файлы в индексацию (в дальнейшем будет добавлять только те, которые были изменены).
3. Добавляет комментарий к новой точке индексации файлов
4. Кладет изменения в origin.

На самом деле веток может быть несколько. Это дает Вам возможность работать с кодом, не изменяя главную ветку реализации продукта. То есть это можно рассматривать как возможность работы с модифицированной копией проекта.

Чтобы создать новую ветку используются следующие команды:

$ git branch new\_branch\_name

После того, как Вы ввели данную команду, git автоматически создаст новую ветку под названием new\_branch\_name и сделает ее вашей основной веткой. То есть, если вы захотите залить изменения (git push), то они будут сделаны именно на ветку new\_branch\_name.

Чтобы перейти на другую ветку, необходимо ввести команду:

$ git checkout branch\_name

После этого вы будете перенесены на ветку с именем branch\_name.

Для того, чтобы выгрузить данные с гита (допустим, вы перешли на другой компьютер, у вас есть установленный для работы софт, но нет проекта, который лежит в определенной директории, скажем, github-a), необходимо ввести команду:

$ git pull origin master

Эта команда выгрузит данные с гита с ветки мастер (последние изменения кода).

# Описание архитектуры проекта

Данный проект будет содержать несколько частей. Упрощенно в нем будет 3 части, которые будут связаны между собой некоторыми интерфейсами:

1. Локальная база данных и код, который бы генерировал или записывал данные в базу данных (для инкапсуляции и защиты доступа к базе данных)
2. Серверная часть, которая будет позволять пользователю с телефона обращаться на удаленную базу данных, выгружать и загружать данные для их дальнейшей обработки и использования.
3. Клиентское приложение, которое Вам непосредственно будет предложено реализовать соответственно с Вашим вариантом задания.

Таким образом, всю систему можно представить следующим образом (рис.1):



Рисунок . Общая структура создаваемого приложения (система как черный ящик).

Из рис.1 видно, что приложение будет напрямую общаться с базой данных, которая будет реализована как облачное хранилище. Данное облачное хранилище будет заполняться с помощью локальной программы, которая позволит нам эмитировать синхронизацию баз данных.

Локальная база данных может быть реализована на чем угодно лично Вам. Однако в практических целях (ввиду массовости) и в целях облегчения установки и реализации проекта будет использоваться база данных MySQLб предоставляемая корпорацией Microsoft для локального развертывания.

Для установки соединения с базой данных, заполнения ее данными, выгрузки данных, их обработки и отправление на веб-облако будет использоваться язык Java (а также реализация JDBC, также предоставляемая корпорацией Microsoft для взаимодействия с базами данных) и его подветка – J2EE (Java to Enterprise Edition).

Для реализации приложения будет также использоваться язык программирования Java c использованием актуальных библиотек по подключению к сетевым интерфейсам (OkHTTP), парсинга файлов формата JSON (GSON), а также язык SQLite, использующийся ОС Android для взаимодействия с внутренними базами данных.

Таким образом, в конце данных лабораторных работ Вы получите полновесное, хоть и легкое приложение, которое будет иметь возможность работать с данными, положенными в удаленную базу данных, мобильное приложение, а также приложение (которое можно будет в дальнейшем оснастить графической реализацией на ПК) для обработки данных локально.

# Локальное позиционирование Базы Данных MySQL

# JDBC. Будущая синхронизация баз данных