Корпоративные информационные системы

# Лабораторная работа 4 – Разработка структуры классов

Написал: Репин Максим  
Студент группы: ИСТ-19-2б

Принял: Банников Руслан Юрьевич

Дата написания: 9-мар-2023

2023

# Постановка задачи

ПО TestTracker хранит порядок шагов тестирования и ожидаемый результат. Пользователи (тестировщики) проходят шаги тестирования, TestTracker фиксирует результат прохождения – достигнут ли ожидаемый результат, или найден баг в тестируемой системе.

В данной лабораторной работе необходимо разработать диаграмму классов интерфейсной и функциональной части КИС в среде построения UML-диаграмм.

# Диаграмма классов

Прежде чем приступить к разработке, необходимо разработать диаграмму классов. На рисунке 1 изображена диаграмма классов, спроектированная средствами сайта diagrams.net.

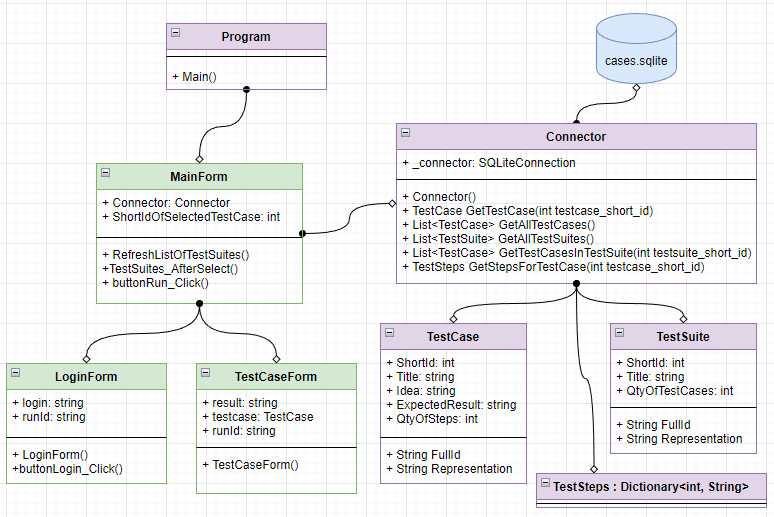
Класс Program – основной класс программы. При запуске у него вызывается метод Main(), который отрисовывает класс MainForm.

Класс LoginForm описывает отображение формы для авторизации для пользователя. Форма отрисовывается при запуске программы. По нажатию кнопки Log in вызывается метод buttonLogin\_Clicked(), который пропускает пользователя на главную форму.

Класс Connector описан для подключения к SQLite базе данных. Методы класса возвращают информацию из базы данных или загружают в неё информацию. Поле \_connector является объектом встроенного класса для подключения к базе данных. В таблице 1 описаны методы, их аргументы и возвращаемый результат, если он есть.

Таблица 1. Методы класса Connector

|  |  |  |
| --- | --- | --- |
| Название метода | Аргументы | Результат |
| Connector() | Нет | Конструктор класса |
| GetTestCase() | int testcase\_short\_id | Возвращает объект класса TestCase с указанным id, если он существует |
| GetAllTestCases() | Нет | Возвращает массив объектов всех описанных тест-кейсов |
| GetAllTestSuites() | Нет | Возвращает массив объектов всех описанных комплектов |
| GetTestCasesInTestSuite() | int testsuite\_short\_id | Возвращает массив объектов тест-кейсов из определённого комплекта |
| GetStepsForTestCase() | int testcase\_short\_id | возвращает экземпляр объекта |

 Рис. 1 – Диаграмма классов

В классе TestCase есть числовые поля ShortId, QtyOfSteps и текстовые поля Title, Idea, ExpetedResult. Также, у класса TestCase есть свойства FullId, который возвращает текст “TC” и добавляет ShortId на конце, и свойство Representation, которые возвращает FullId тест-кейса, его название и количество шагов в тест-кейсе в формате “TC999999: Название (16)”.

В классе TestSuite есть числовые поля ShortId и QtyOfTestCases и текстовое поле Title. У класса TestSuite есть такие же свойства, как и у класса TestCase, выполняющие ту же функцию, однако количество шагов заменено на количество тест-кейсов в комплекте.

Класс TestSteps наследуется от словаря с числовым ключом и текстовым значением.

В классе MainForm описан интерфейс главной формы. У этого класса есть два поля – Connector (объект класса Connector) и числовое поле ShortIdOfSelectedTestCase (id выбранного в интерфейсе TestCase’а). Метод RefreshListOfTestSuites() обновляет в интерфейсе список комплектов и тест-кейсов. Метод TestSuites\_AfterSelect() обновляет значение поля, в котором хранится выбранный TestCase. Метод buttonRun\_Click() запускает для пользователя окно формы TestCaseForm для прохождения тестирования.

Класс TestCaseForm описывает отображение окна для прохождения тестирования. Для запуска окно принимает в качестве аргументов объекты классов TestCase и TestSteps. Поле result хранит необходимый для достижения результат, поле testcase хранит объект класса TestCase, поле RunId хранит текущий установленный run id.

# Вывод

В рамках данной лабораторной работы была разработана диаграмма классов программы. Приведены поля, свойства и описаны методы и их назначение для каждого класса.

В данной лабораторной были достигнуты все поставленные цели.

Цель следующей лабораторной работы – реализовать описанные классы на языке C# для реализации программы.