### 2.2.4 Диаграмма классов (class diagram)

В программной инженерии диаграмма классов в унифицированном языке моделирования (UML) представляет собой тип статической структурной схемы, которая описывает структуру системы, показывая классы системы, их атрибуты, операции (или методы), а также отношения между объектами.

Диаграмма классов является основным строительным блоком объектно-ориентированного моделирования. Он используется как для общего концептуального моделирования системного приложения, так и для детального моделирования перевода моделей в программный код. Диаграммы классов также могут использоваться для моделирования данных. Классы в диаграмме классов представляют, как основные элементы, взаимодействия в приложении, так и классы, которые будут запрограммированы.

При проектировании системы несколько классов идентифицируются и группируются вместе в диаграмме классов, которая помогает определить статические отношения между ними. При детальном моделировании классы концептуального проектирования часто разбиваются на несколько подклассов.

Для дальнейшего описания поведения систем эти диаграммы классов могут быть дополнены диаграммой состояний или машиной состояний UML.

(А диаграммы тут пока и нет, она будет, но чуть позже, когда я посмотрю как изнутри выглядит приложение)

### 2.2.5 Диаграмма компонентов (component diagram)

Диаграмма компонентов — элемент языка моделирования UML, статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

С помощью диаграммы компонентов представляются инкапсулированные классы вместе с их интерфейсными оболочками, портами и внутренними структурами (которые тоже могут состоять из компонентов и коннекторов).

Компоненты связываются через зависимости, когда соединяется требуемый интерфейс одного компонента с имеющимся интерфейсом другого компонента. Таким образом иллюстрируются отношения клиент-источник между двумя компонентами.

Зависимость показывает, что один компонент предоставляет сервис, необходимый другому компоненту. Зависимость изображается стрелкой от интерфейса или порта клиента к импортируемому интерфейсу.

Когда диаграмма компонентов используется, чтобы показать внутреннюю структуру компонентов, предоставляемый и требуемый интерфейсы составного компонента, могут делегироваться в соответствующие интерфейсы внутренних компонентов.

(А она тоже тут появится после того, как я посмотрю, как изнутри выглядит приложение, чтобы не переделывать её потом).