Оглавление

[ОБЩИЕ ПРИНЦИПЫ ИСПОЛЬЗОВАНИЯ ДИАГРАММ ИЯЗЫКА UML 2](#_Toc164805099)

[Виды UML диаграмм 3](#_Toc164805100)

[Диаграмма классов 5](#_Toc164805101)

[Основные способы изображения классов 5](#_Toc164805102)

[Поля классов 5](#_Toc164805103)

[Методы классов 6](#_Toc164805104)

# ОБЩИЕ ПРИНЦИПЫ ИСПОЛЬЗОВАНИЯ ДИАГРАММ ИЯЗЫКА UML

Язык UML используется в 3 условных режимах.

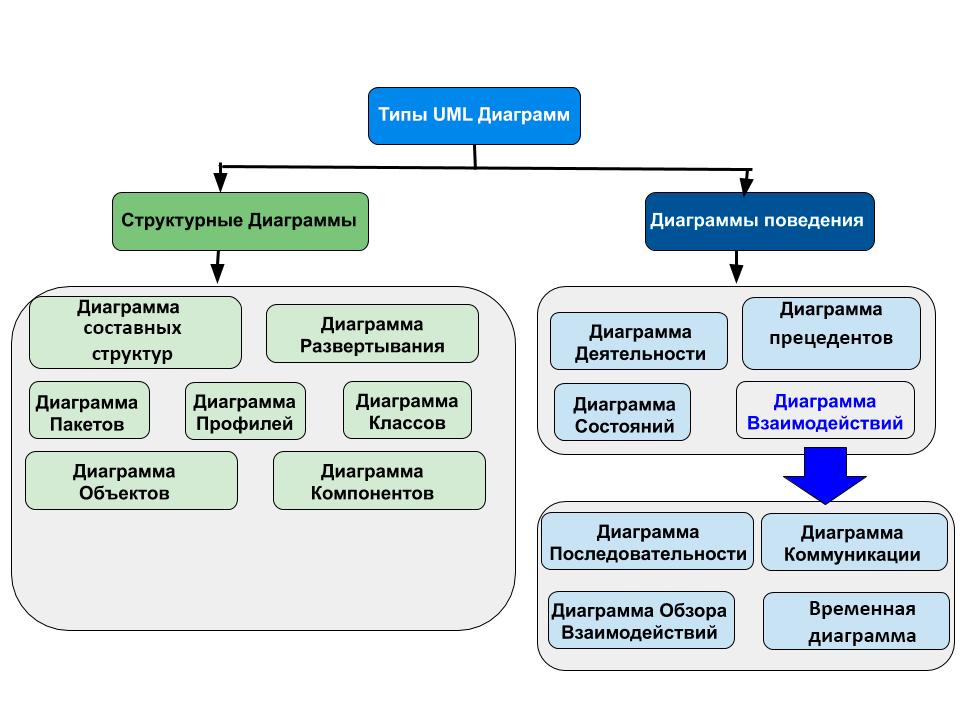
Первым режимом является эскизирование. Эскизирование необходимо для обмена между разработчиками информацией о различных аспектах системы. Данный вид наиболее часто диаграммы используются в программной документации при обсуждении, постановке задачи на проектирование.

Прямая разработка необходима для построение максимально полной, детальной модели для программиста (программист должен иметь возможность следовать им при реализации программы прямо и не особо задумываясь). Данный вид диаграмм наиболее часто используются в программной реализации.

Обратная разработка необходима, когда диаграммы строятся на основе кода, чтобы объяснить, как система работает. Данный вид диаграмм наиболее часто используются при описании структуры существующей программы.

# Виды UML диаграмм

В UML версии 2.5.1 определено 13 видов диаграмм:



Из структурных диаграмм выделяют следующие под-диаграммы:

1. Диаграммы классов - описывают классы программы/системы, их свойства и статические отношения, которые существуют между ними
2. Диаграммы объектов представляют снимок объектов программы/системы в определенный момент времени
3. Диаграммы компонентов отображают компоненты системы, различные виды связей между ними и интерфейсы взаимодействия
4. Диаграммы составных структур отображают внутреннюю структуру компонентов или подсистем (части, взаимосвязи (коннекторы), интерфейсы и порты)
5. Диаграммы пакетов позволяют структурировать модель или отобразить структуру системы, указывая разбиение на логические части, содержимое этих частей и взаимосвязи между ними
6. Диаграммы развертывания/размещения представляют структуру аппаратных, коммуникационных средств, а также физическое расположение элементов программного обеспечения (элементов конфигурации) на оборудовании, информационные пути и протоколы взаимодействия между элементами
7. Диаграммы профилей описывают динамическую метамодель системы с помощью стереотипов, которые специфицируют изменение свойств объектов (классов, компонентов) в случае применения к системе различных профилей, модифицирующих ее поведение

Из поведенческих диаграмм выделяют следующие под-диаграммы:

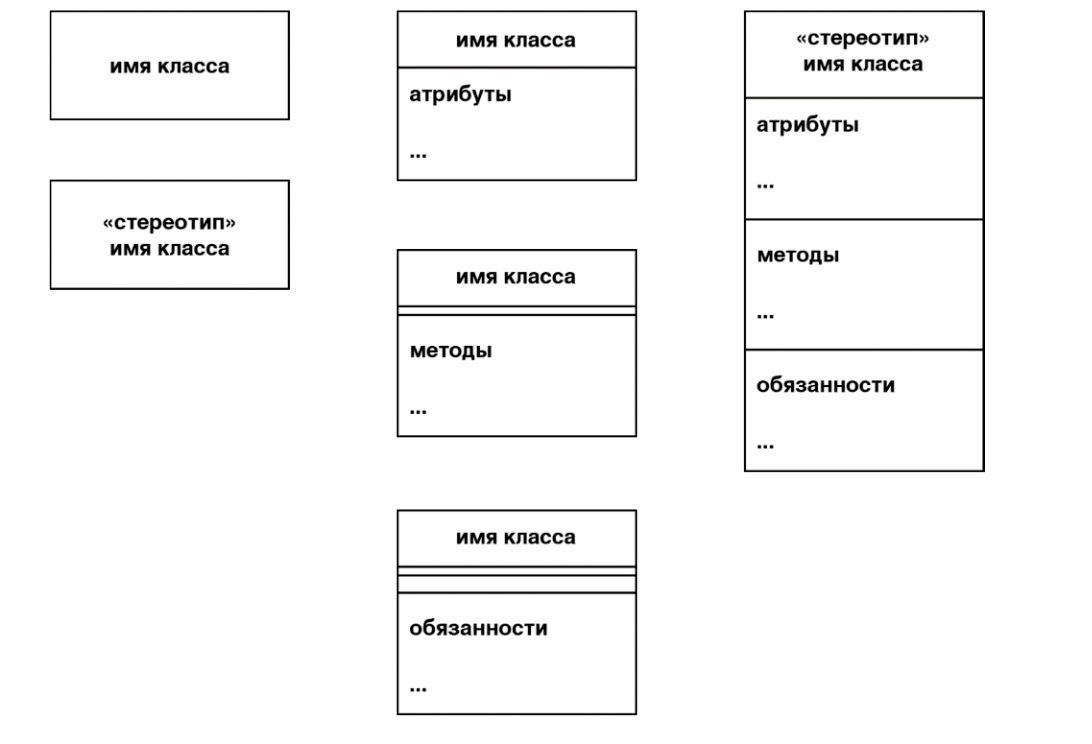
1. Диаграммы прецедентов отображают функциональное назначение и границы системы или ее части в виде набора сценариев использования системы(прецедентов) акторами (заинтересованными лицами или внешними системами)
2. Диаграммы деятельности описывают логику и процесс выполнения некоторой деятельности, включая основные шаги, переходы между ними и потоки управления
3. Диаграммы состояний (конечных автоматов) отображают всевозможные состояния системы или отдельных объектов на протяжении их жизненного цикла, события и переходы между состояниями, а также структуру сложных и параллельных состояний
4. Диаграммы взаимодействия:
   1. Диаграммы коммуникации отображают связи и обмен данными между различными объектами в процессе их взаимодействия с акцентом на структурную организацию объектов и передаваемые между ними сообщения
   2. Диаграммы последовательности описывают взаимодействие объектов с акцентом на последовательность взаимодействий во времени, логику и параллельность исполнения операций
   3. Временные диаграммы отражают взаимодействия между объектами или внутри одного объекта с акцентом на состояния объекта, события, временные параметры и ограничения
   4. Диаграммы обзора взаимодействия комбинируют инструменты диаграмм деятельности и диаграмм последовательности для отображения как общей логики выполнения процесса, так и деталей взаимодействия между участвующими в нем объектами.

# Диаграмма классов

**Класс** — абстрактный тип данных, задающий общие атрибуты и поведение для группы объектов.

Диаграммы классов описывают типы объектов системы (классы), их свойства и статические отношения, которые существуют между ними. Для классов могут отображаться обязанности классов, их атрибуты и методы. В общем трактовка классов в UML ничем не отличается от трактовки в ООП.

## Основные способы изображения классов



Обязательным к описанию у класса является только его название.

### Поля объектов/классов

Поля у классов описываются следующем образом:

“Метка видимости” “Имя атрибута”: “Тип данных атрибута” = “Значение по умолчанию” {“Строка свойств”}

Метка видимости – обозначает модификатор доступа к атрибутам (общее для всех):

1. + - public. Необходим для объявления публичных атрибутов
2. – - private. Необходим для объявления срытых/инкапсулированных атрибутов
3. # - protected. Необходим для объявления защищённых атрибутов
4. ~ package. Необходим для объявления атрибутов с областью видимости типа пакетный (то есть атрибутов внутри пакета)

После метки видимости можно указывать знак $ - что указываем о том, что поле или метод является статическим (то есть принадлежит классу, а не объекту)

Указывание “Значение по умолчанию” и {“Строка свойств”} для полей опциональное (их можно не указывать)

### Методы классов

Методы у классов описываются следующим образом:

“Метка видимости” “Имя метода” (“Ключевое слово” “Имя аргумента”: “Тип данных аргумента” = “Значение по умолчанию”): “Тип данных, который вернет метод” {“Строка свойств”}

Ключевое слово – обозначает какого типа атрибут. Бывает следующих видов:

* + 1. in – входной атрибут (необходим для передачи данных в метод)
    2. out – выходной параметр. Не передает каких-либо данных в метод. (аргумент, через который метод возвращает данные)
    3. inout – аргумент получаемый вводом/выводом (передает в метод какие-то данные, после данные изменяются, изменения возвращаются через аргумент)

Указание “аргументов и все что с ними связано” и {“Строка свойств”} для методов опциональное.

Для аргументов указание “Значение по умолчанию” опциональное.

При необходимости допускается указывать, что список атрибутов, операций или обязанностей неполный. При этом на новой строке перечня необходимо указать многоточие.

Для удобства на диаграмме можно отображать классы разных типов. При этом либо указывается стереотип (тип класса), либо специальное графическое обозначение. Стандарт содержит обозначения для следующих типов классов

1. Интерфейсы – описание контракта, набора видимых и доступных извне операций, определяющих поведение класса, компонента, либо часть такого поведения, если класс (компонент) предоставляет несколько интерфейсов
2. Абстрактные классы
3. Активные классы являются владельцами одного или нескольких потоков исполнения (процессов), благодаря чему могут инициировать асинхронные взаимодействия и управлять другими классами
4. Типы данных
5. Примитивные типы данных
6. Перечислимые типы данных.

При проектировании также встречаются специальные обозначения для следующих типов классов:

1. Граничные классы - обеспечивают взаимодействия с акторами (клиентами, пользователями, внешними системами)
2. Управляющие классы определяют логику работы системы
3. Классы–сущности соответствуют объектам реального мира, над которыми система выполняет некоторые операции.

Графическое отображение вышеописанных типов

Рис. 3. Обозначения для специальных типов классов