Тема 7.
Диаграмма классов

Диаграмма классов

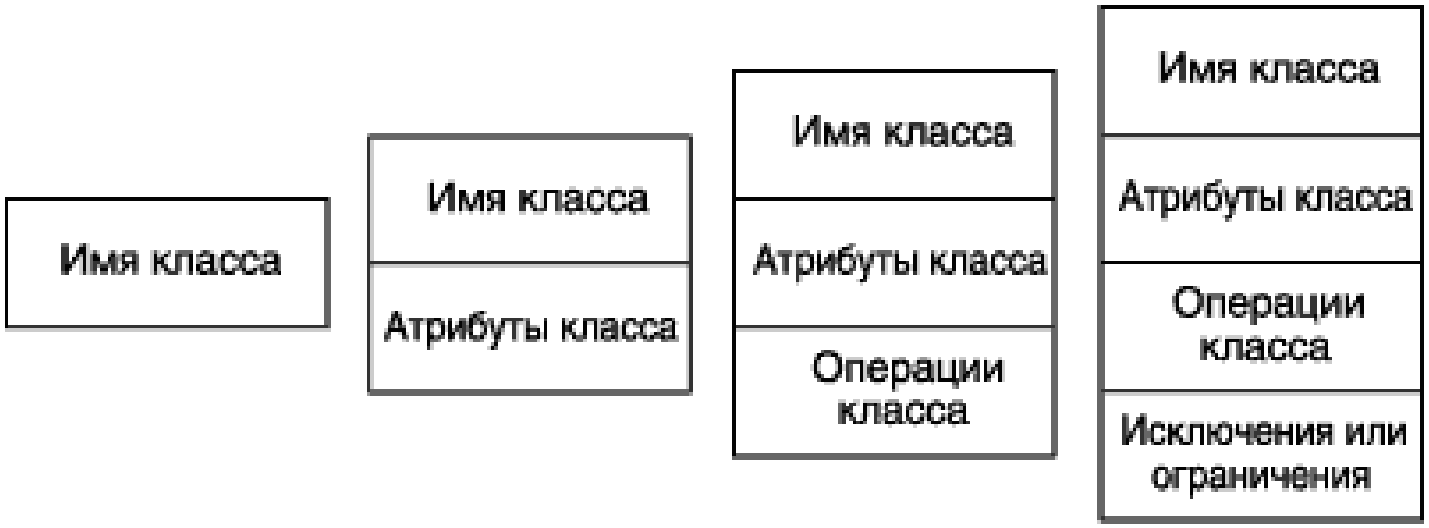
Диаграмма *классов* (class diagram) — структурная диаграмма языка UML, на которой представлена совокупность статических элементов модели - *классы* с *атрибутами* и *операциями*, а также связывающие их отношения.

Статическое структурное представление системы.

Класс

Класс представляет собой абстрактное описание множества однородных объектов, которые имеют общий набор свойств и обладают одинаковым поведением.
Объект - экземпляр соответствующего *класса*.

Графическое обозначение класса

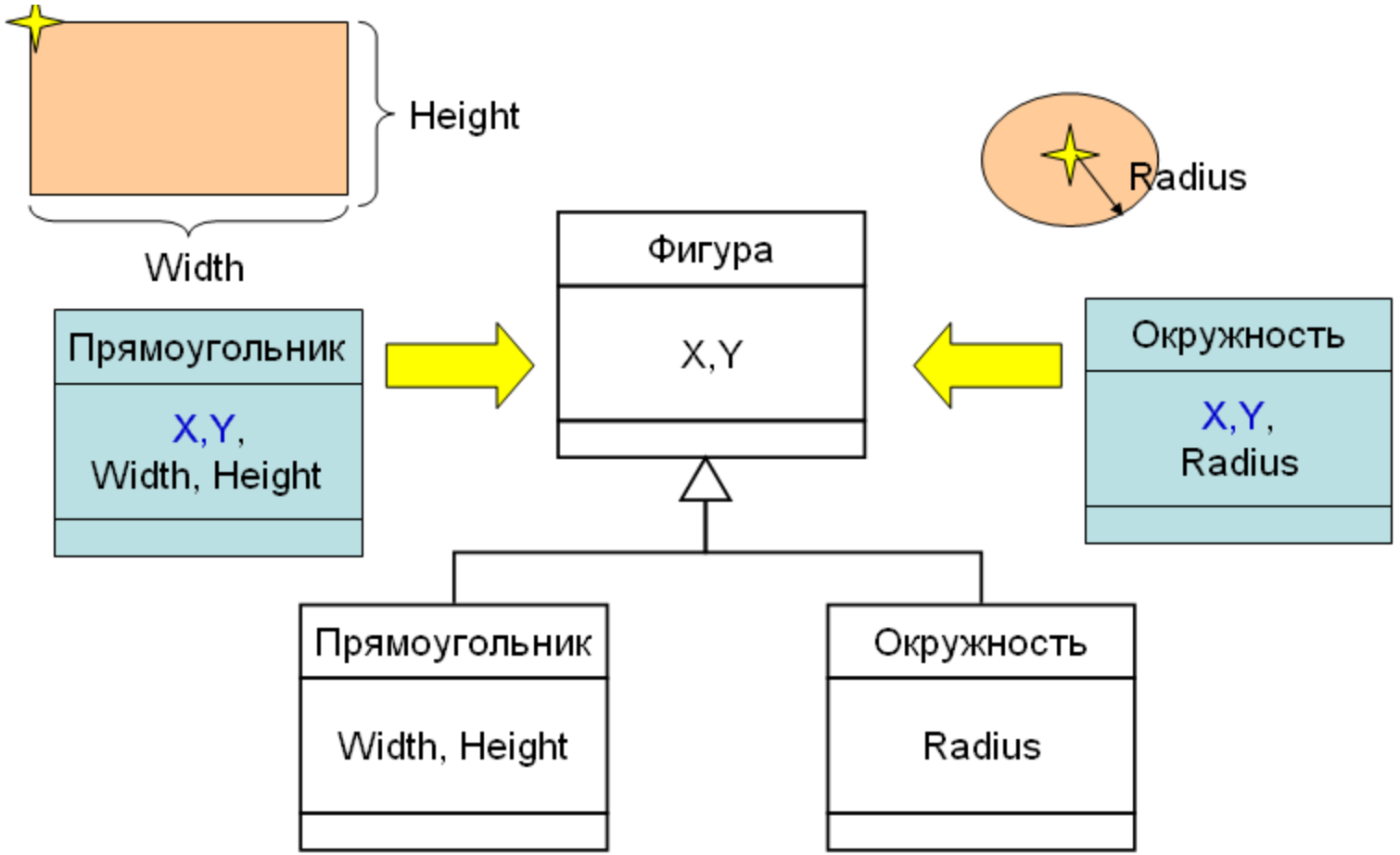


Графическое обозначение класса

Абстрактный класс

Конкретный класс (concrete class) — *класс*, на основе которого непосредственно могут быть созданы объекты.

Абстрактный класс (abstract class) — *класс*, который не имеет объектов.



Атрибуты класса

Атрибут класса служит для представления отдельного свойства или признака, который является общим для всех объектов данного *класса*.

<квантор видимости> <имя атрибута> [кратность] :<тип атрибута> = <исходное значение>

Сокрытие внутреннего устройства объекта (инкапсуляция)

- Обеспечивает создание иллюзии простоты для пользователя (Г. Буч)
- Необходима для защиты от пользователя внутреннего устройства объекта
- В языках программирования реализуется на основе ограничения доступа к атрибутам и операциям путем использования кванторов видимости

Атрибуты класса: кванторы видимости

Кванторы видимости (visibility)

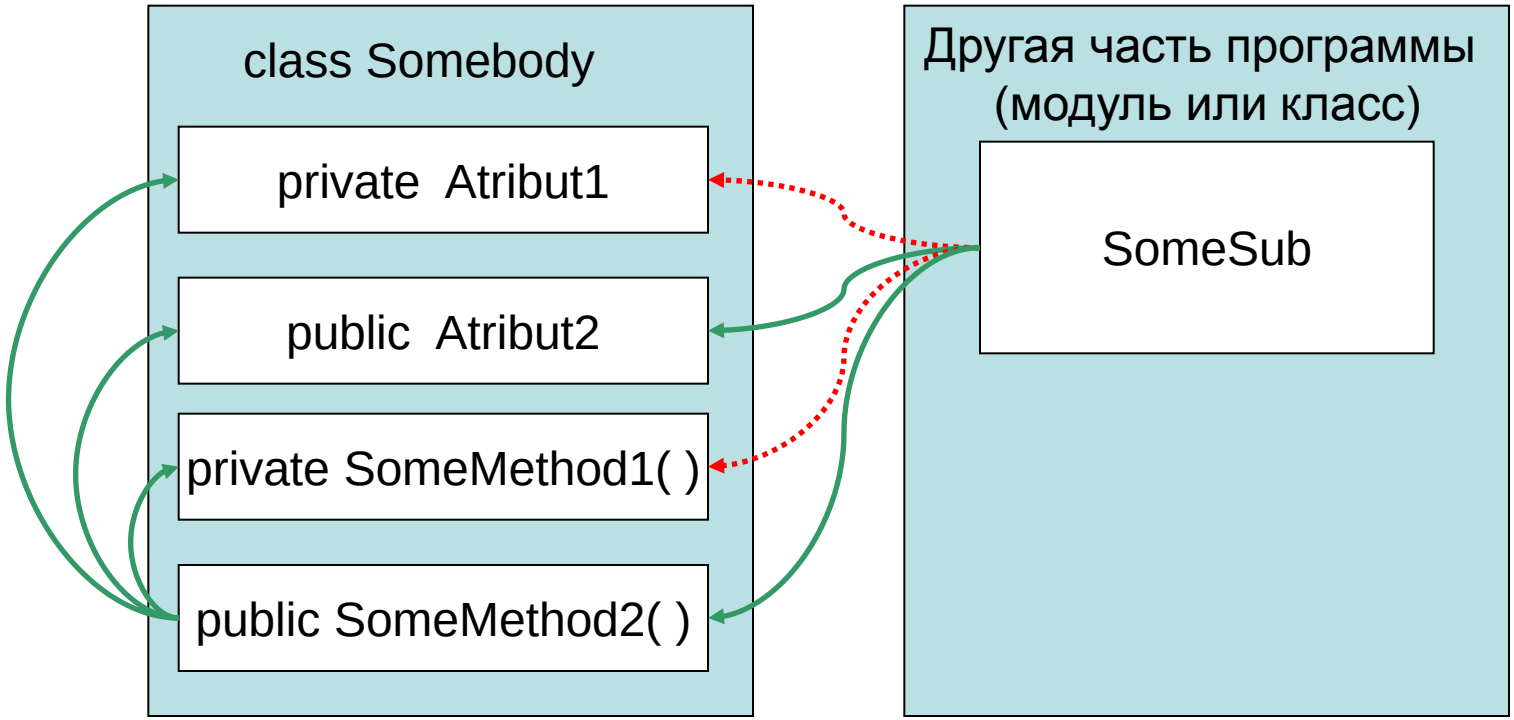
«-» (private) виден только из операций этого же класса

«#» (protected) виден только из операций этого же класса и классов создаваемых на его основе

«+» (public) общедоступный

«~» (package) доступен только для классов данного пакета

Доступ к закрытым и открытым атрибутам и операциям



- public –модификатор открытого доступа
- private – модификатор закрытого доступа

Доступ возможен

Доступ невозможен

Пример инкапсуляции

Пластиковая карта VISA
- Номер счета: длинное - Пин-код:целое - Сумма: финансовый
-Дешифрация(пин) +Авторизация(пин) +Остаток( ) +Снятие(сумма) +Зачисление( )

Телевизор
-НастройкиКаналов -ДешифраторСигнала -Преобразователь
-Самодиагностика() +Включить() +Выключить() -ДекодерСигнала() +ПереключениеКанала()

Атрибуты класса: кратность

Кратность атрибута характеризует общее количество конкретных *атрибутов* данного типа, входящих в состав отдельного *класса*.

[нижняя граница .. верхняя граница]
"*" - произвольное целое число

Операции класса

Операция (operation) - это сервис, предоставляемый каждым экземпляром *класса* по требованию своих клиентов, в качестве которых могут выступать другие объекты, в том числе и экземпляры данного *класса*.

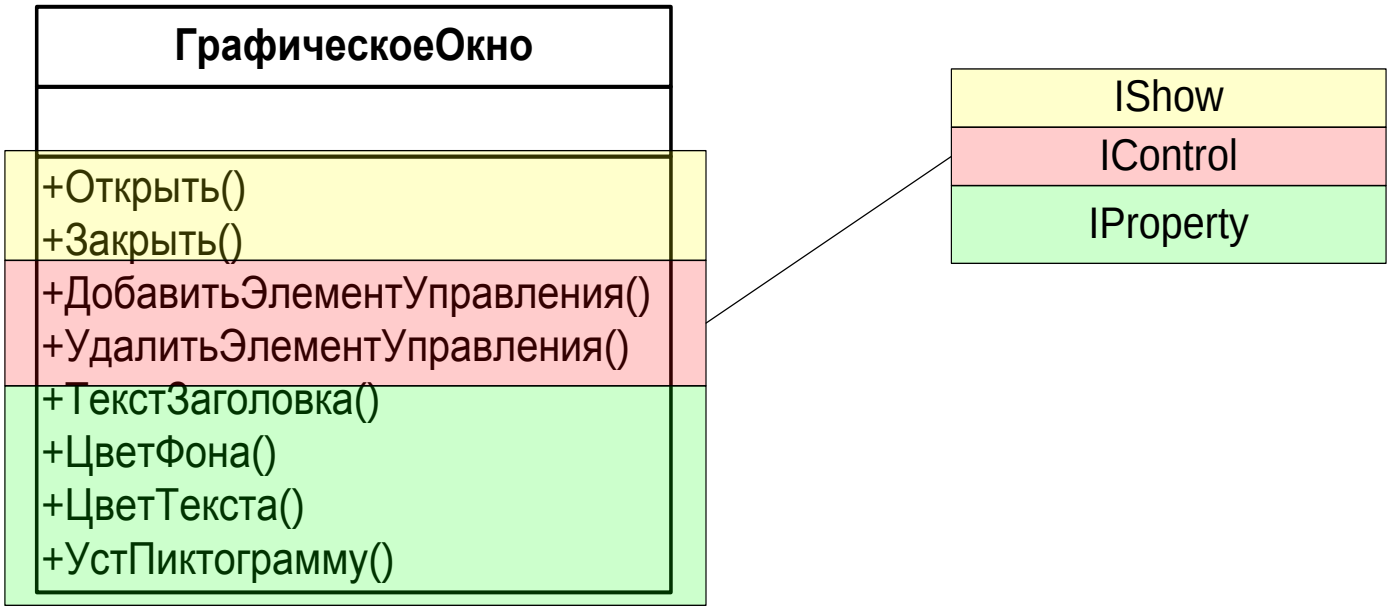
<квантор видимости> <имя операции> (
список параметров):
<выражение типа возвращаемого значения>

<направление параметра> <имя параметра>:
<выражение типа> =
<значение параметра по умолчанию>

Как нам работать с объектом?

- Использовать стандартный механизм доступа к объекту
- Стандартный механизм доступа к объекту должен:
  - Должен быть универсальным и не зависеть от языка программирования
  - Должен быть достаточно простым
  - Должен полностью обеспечивать выполнение спецификации внешнего проявления объекта

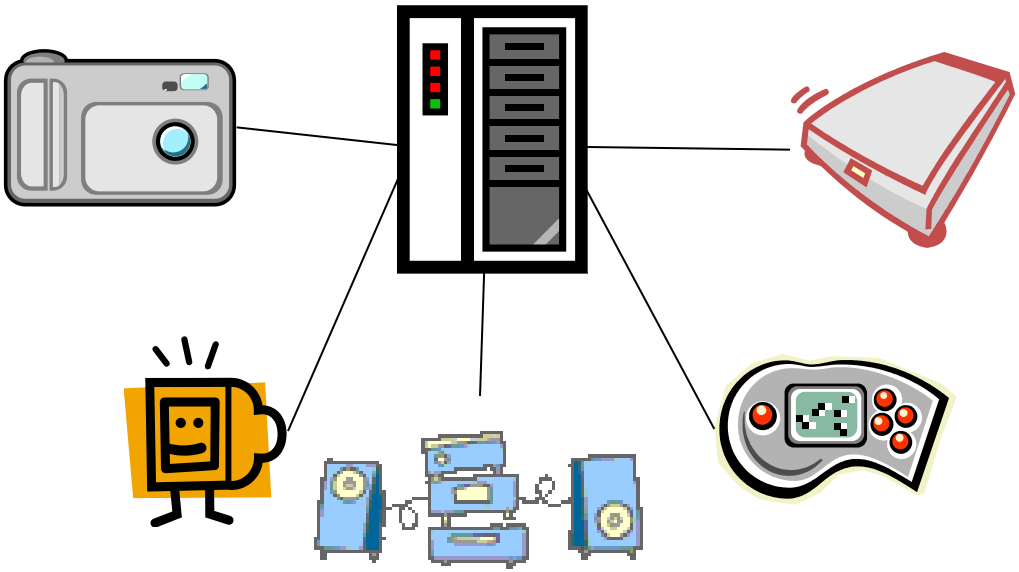
Интерфейс как логическая группа операций объекта

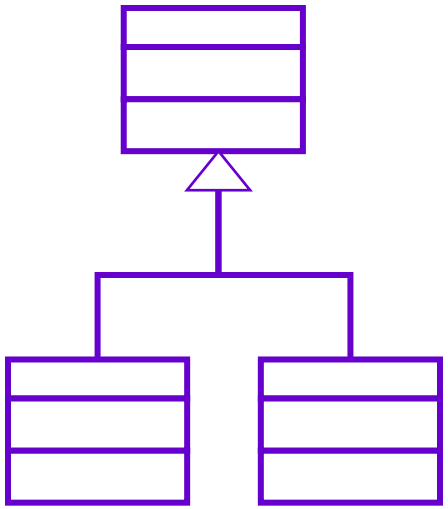


Интерфейс – логическая группа открытых (public) операций объекта

Примеры интерфейсов

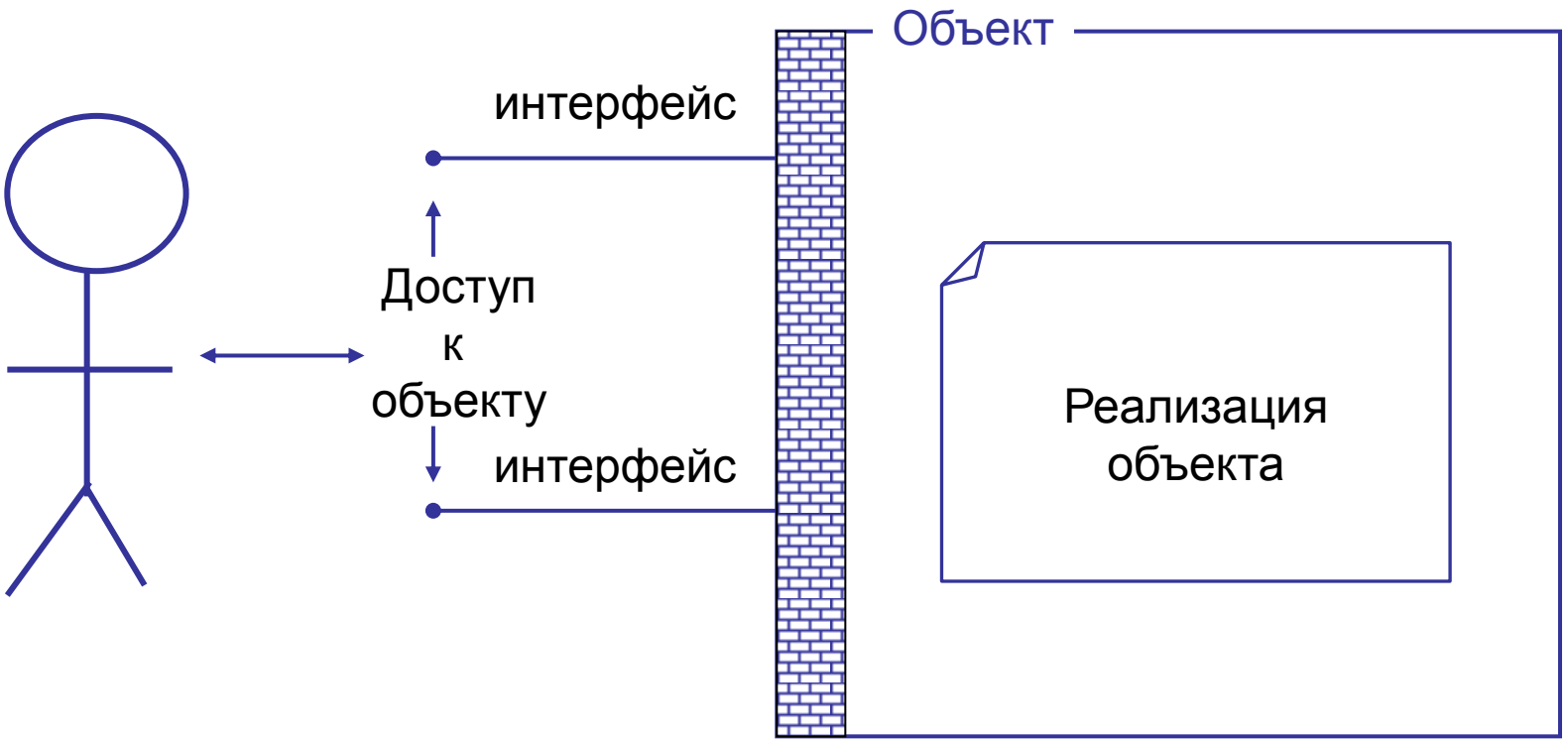
Компания разрабатывает программное обеспечение для домашнего медиацентра. Определите перечень объектов, которые в него входят, а также их интерфейсы.





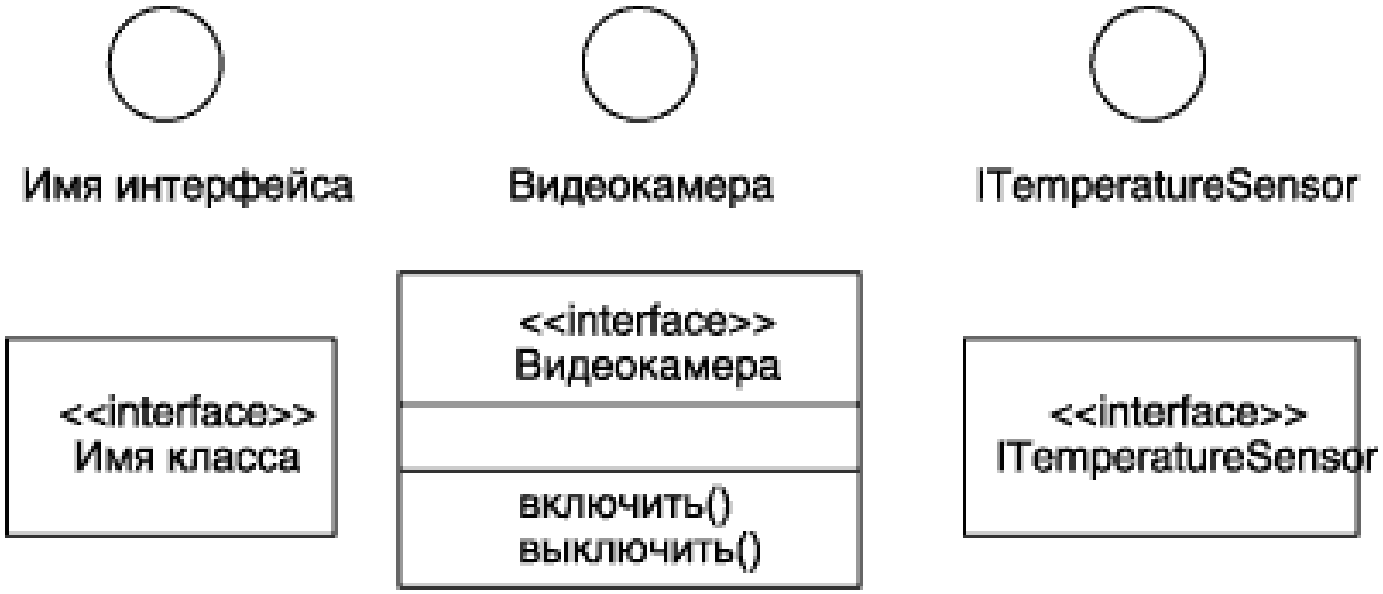
- Интерфейс управляющей системы
  - ИдентификацияУстройства
  - ПодачаКоманды
  - ВключениеОтключение
- Интерфейс телевизора
  - Команда
  - ВключениеОтключение
  - НастройкиИзображения

Интерфейс и его реализация

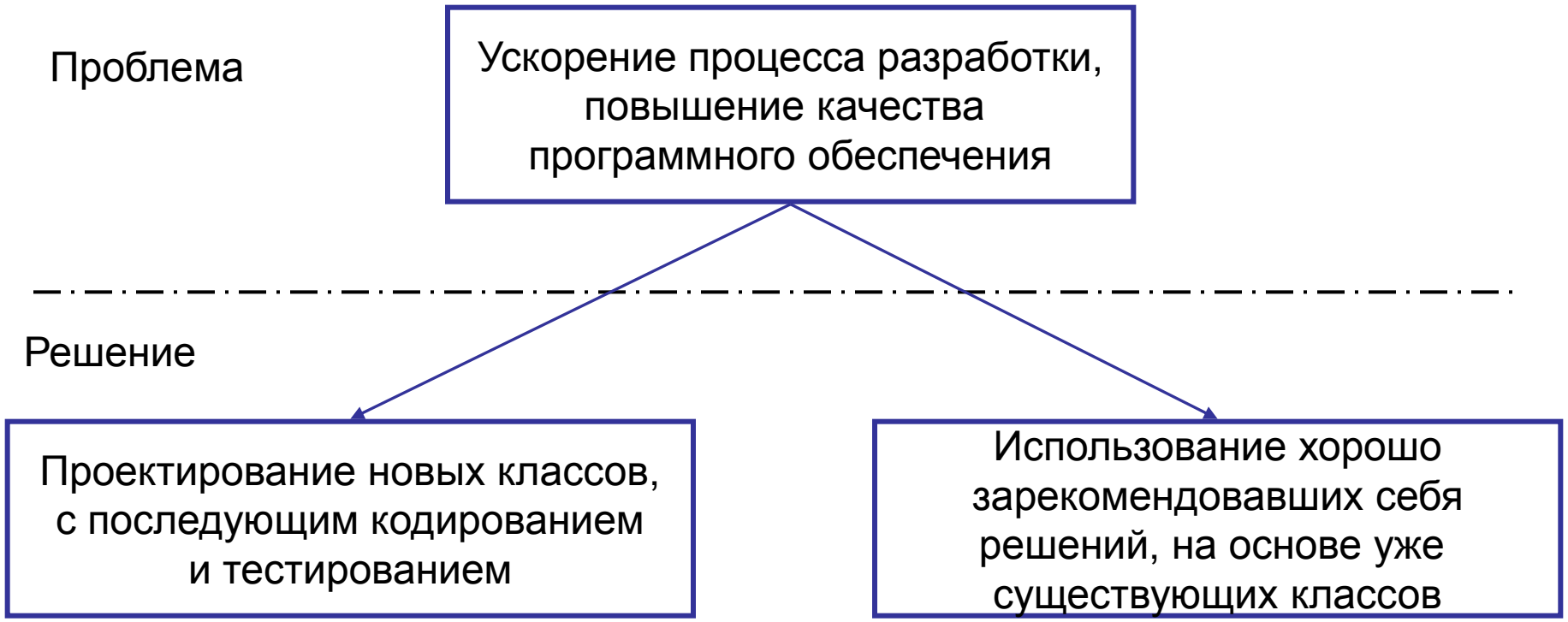


класс реализует интерфейс

Изображение интерфейса в UML



Всегда ли нужно создавать новый класс?



Существующие классы надо использовать потому что…

- Повторное использование ранее принятых решений
- Делает решение мобильным
- Существующие классы, как правило, хорошо отлажены и показали себя в работе

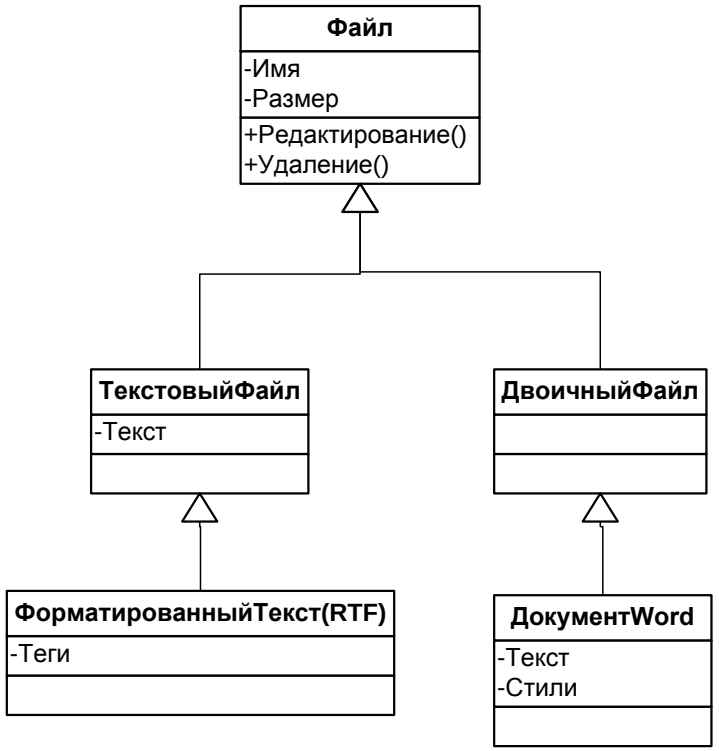
Важные понятия обобщения

- Объектно-ориентированный подход позволяет устанавливать отношение между общей сущностью и ее конкретным воплощением, которое называется **обобщением** (наследование)
- **Класс-родитель** (суперкласс) – класс, который служит основой для создания новых классов. Может быть абстрактным классом
- **Класс-потомок** (подкласс) – класс, который создается на основе класса-родителя
- Все атрибуты и операции класса-родителя независимо от кванторов видимости входят в состав класса-потомка

Графическое изображение обобщения в UML

Пример

Постройте диаграмму классов, на которой показаны
отношения обобщения между файлом, текстовым файлом,
файлом, содержащим форматированный текст
и документом Microsoft Word

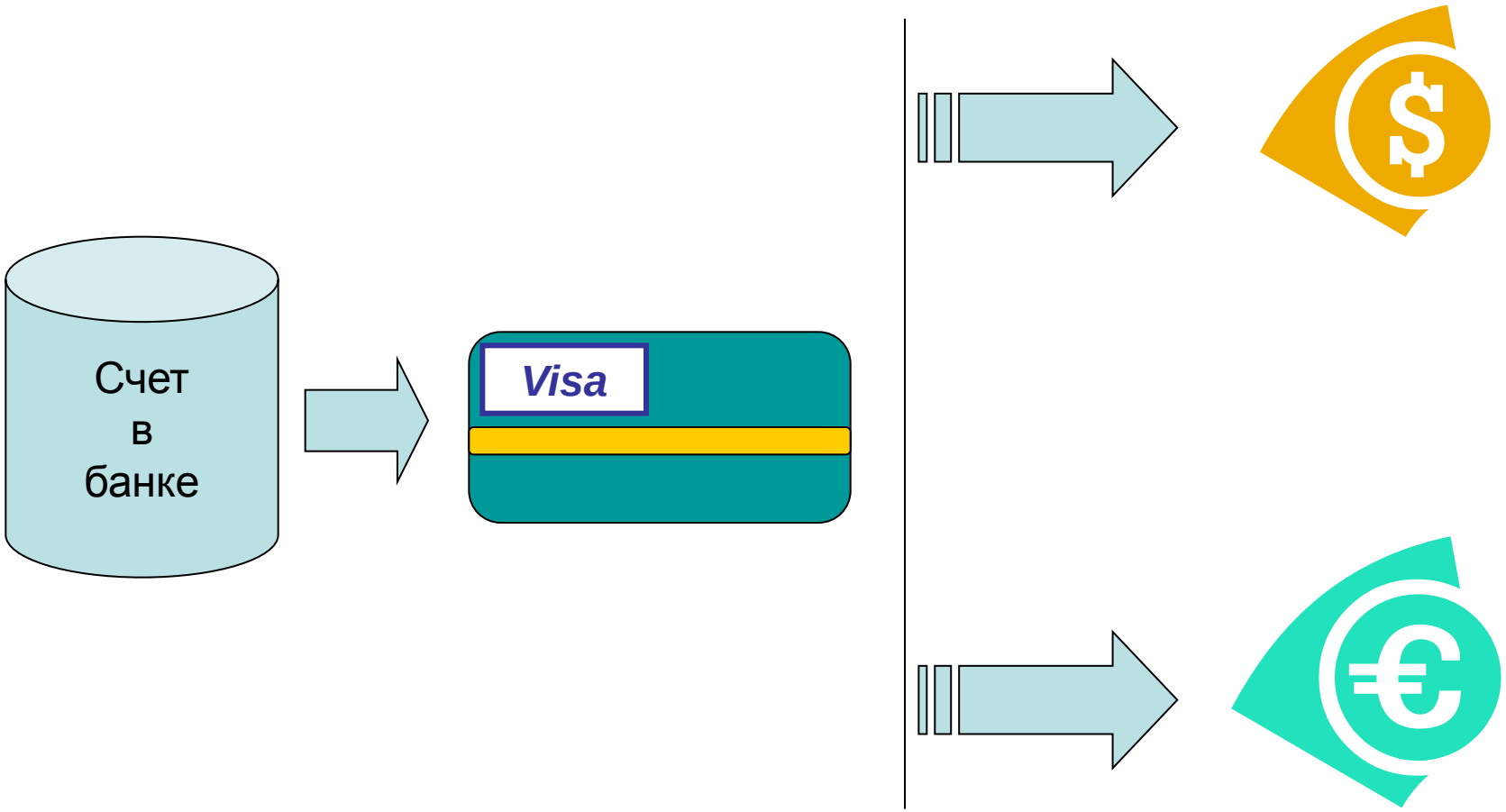


Для работы с объектом для нас главным является его интерфейс.
А все-таки какого класса объект?

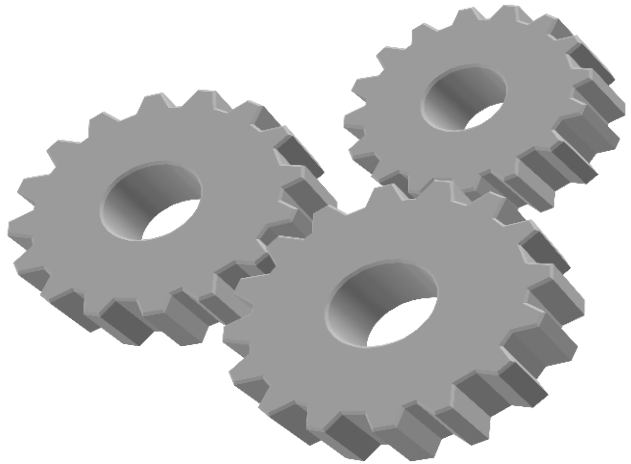
Полиморфизм

- Это способность объектов разной природы (классов) поддерживать один и тот же интерфейс, так как этого ожидает пользователь
- Является основой для реализации механизма интерфейса в языках программирования
- Тип объекта определяется в момент обращения к его операциям через интерфейс, поддерживаемый его классом

Подстановка объектов (полиморфизм)

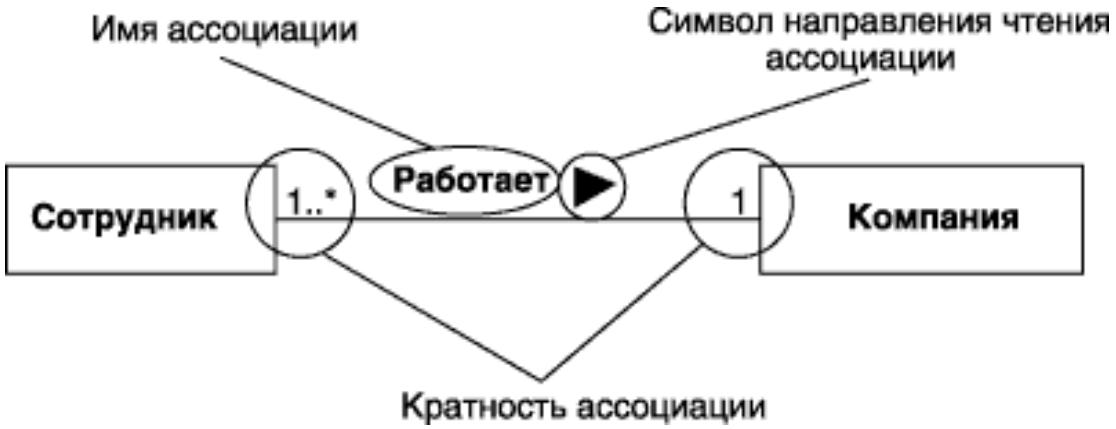


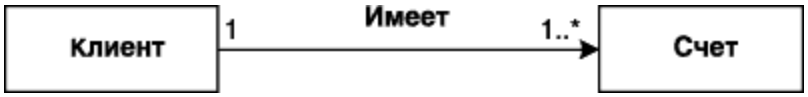
Структурные отношения между классами



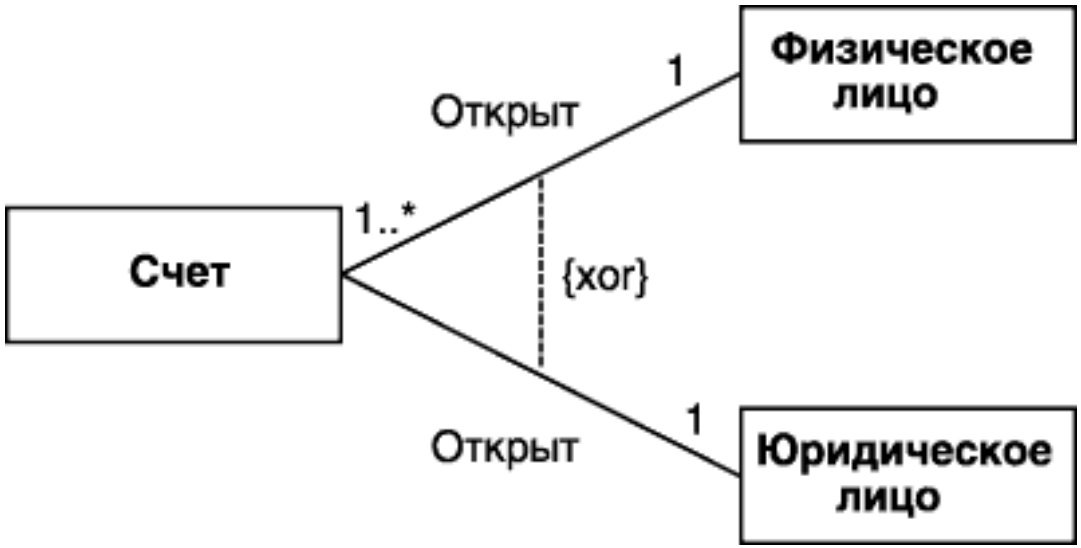
Ассоциация

- Отношение между классами и их объектами, которые имеют равноправное значение в предметной области
- Позволяет перемещаться от объектов одного класса к объектам другого

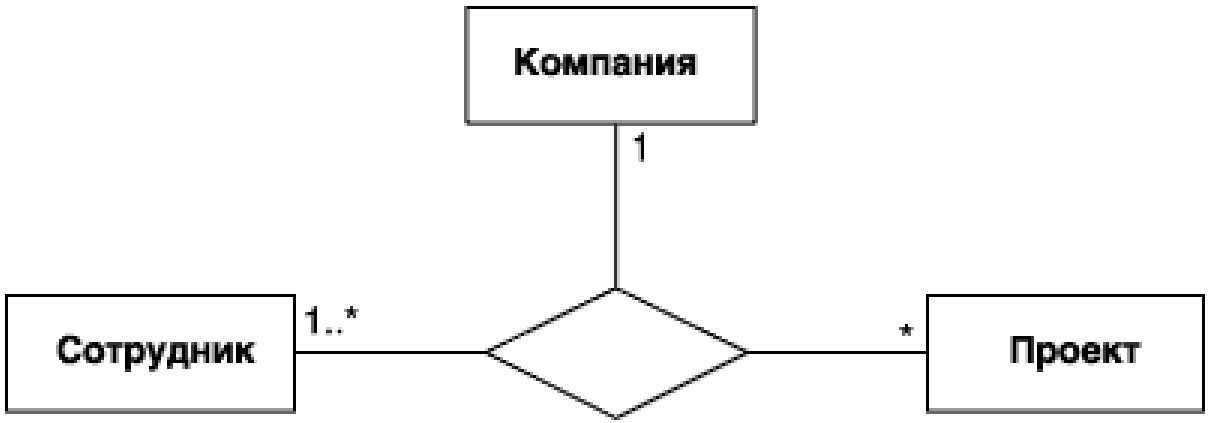




Исключающая ассоциация

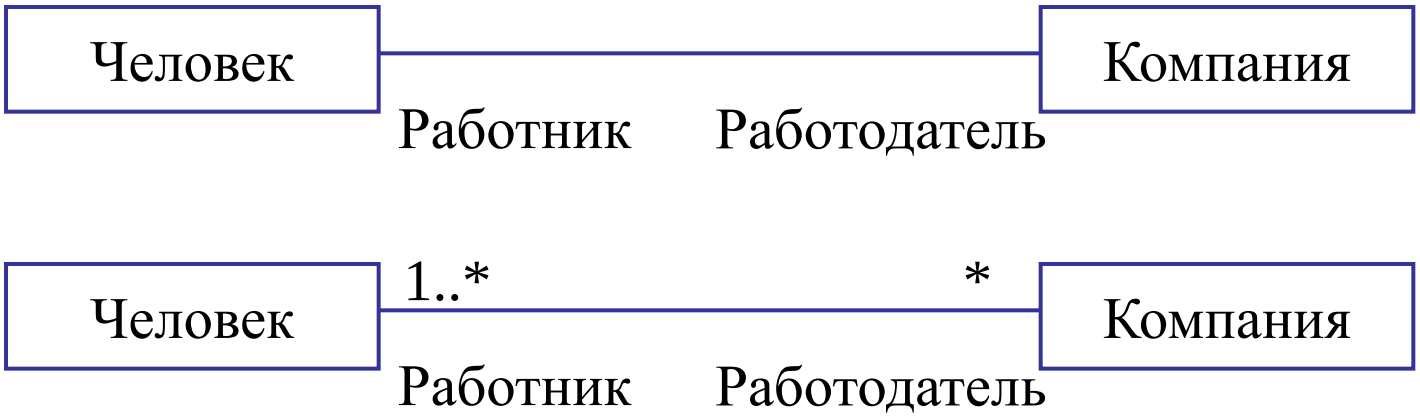


Тернарная ассоциация



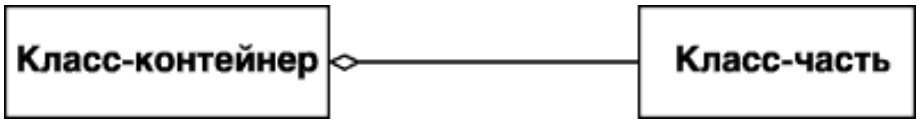
Ассоциация: роль

Роль (role) - имеющее имя специфическое поведение некоторой сущности, рассматриваемой в определенном контексте.



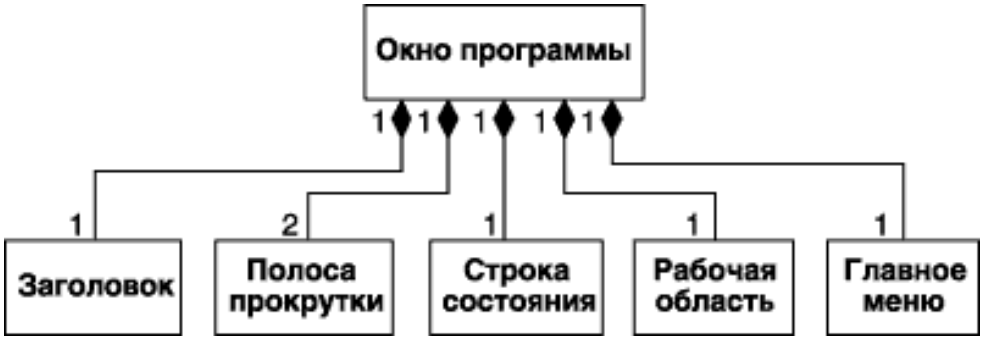
Агрегация

Агрегация (aggregation) - специальная форма ассоциации, которая служит для представления отношения типа "часть-целое" между агрегатом (целое) и его составной частью.



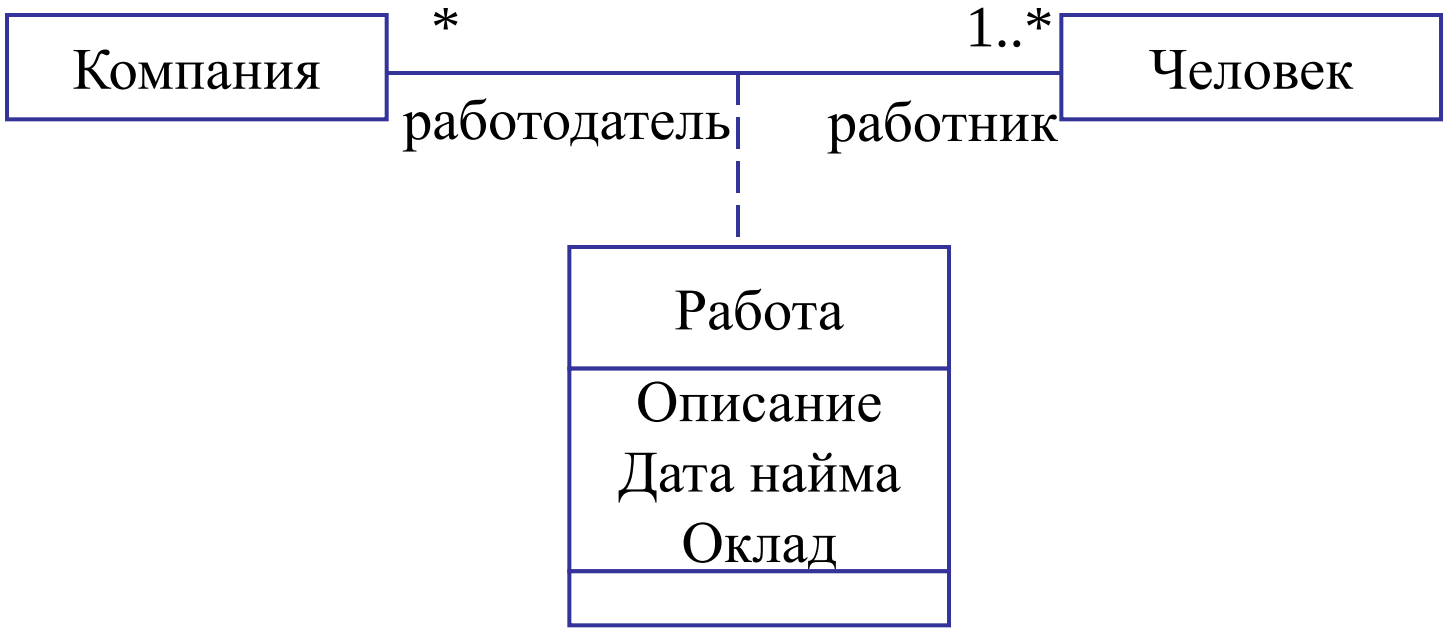
Композиция

Композиция (composition) - разновидность отношения *агрегации*, при которой составные части целого имеют такое же время жизни, что и само целое. Эти части уничтожаются вместе с уничтожением целого.



Класс-ассоциация

Сама ассоциация может быть объектом некоторого класса



Рекомендации по построению диаграммы классов

1. Заостряйте внимание только на одном аспекте статического вида системы с точки зрения проектирования.
2. Включайте в диаграмму только элементы, существенные для понимания данного аспекта.
3. Показываете детали, соответствующие требуемому уровню абстракции, опуская те, без которых можно обойтись.
4. Имя диаграммы связано с ее назначением.
5. Располагайте элементы так, чтобы свести к минимуму число пересекающиеся линий.
6. Пространственно организуйте элементы так, чтобы семантически близкие сущности располагались рядом.
7. Чтобы привлечь внимание к важным особенностям диаграммы, используйте примечания и цвет.

Алгоритм построения диаграммы классов

1. Выделить сущности предметной области – будущие классы.
2. Разместить классы системы (или подсистемы) на диаграмме.
3. Установить между классами отношения обобщения, ассоциации. Уточнить ассоциации, если они являются агрегированием или композицией.
4. Детализировать классы – создать атрибуты и операции, необходимые в этом контексте.
5. Написать примечания, если они необходимы.

Пример диаграммы классов