Полиморфизм

- Полиморфизм - улучшает реализацию кода, способствует созданию расширяемых программ.
- Инкапсуляция - новые типы за счет объединения характеристик и поведения, новые типы с сокрытой реализацией.
- Полиморфизм - логическое разделение в контексте типов.
- Наследование - работа с объектом как с с собственным так и с базовым типом. Вызов полиморфного метода - выражение разности одного типа от другого -> множество типов, один метод, разнообразный итог

- Полиморфизм - динамическое связывание, поздние связывание, связывание во время выполнения

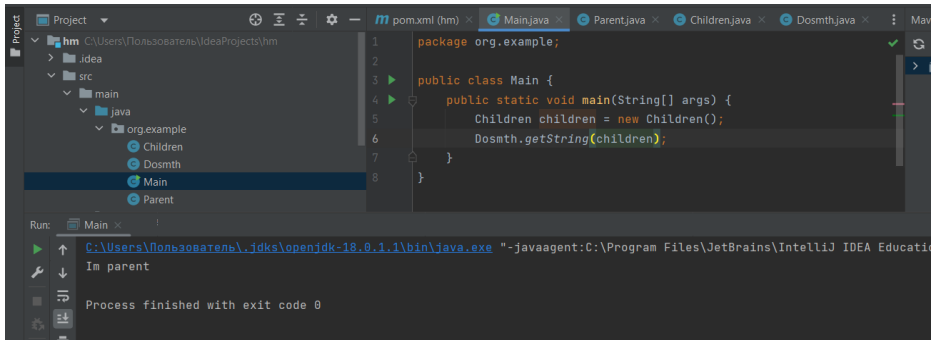
- При восходящем преобразовании происходит умышленное игнорирование типа. Зато можно осуществить реализацию функции в зависимости от представления наследника
- Пример полиморфизма - функция, в аргументах - базовый класс, в теле аргумент.делай что-то. Много наследников, каждые делают что-то по разному. При использовании функции в аргументы можно поместить разных наследников и итог будет разным. Это был полиморфизм

- Связывание - присоединение вызова метода к телу метода.
- Ранее связывание - связывание перед запуском программы
- Позднее связывание - во время выполнение программы, в зависимости от типа объекта (полиморфизм) (ака динамическое связывание, связывание на стадии выполнения). Должен существовать механизм определения фактического типа объекта.
- Для джава все, кроме финал (и соответственно приват) используется позднее связывание. Не придется принимать решений относительно использования позднего связывание - осуществляется автоматически(чуть быстрее чуть производительнее но не сильно).
- Классический пример полиморфизма
- Геометрические фигуры с функцией нарисуй, которая выполняется в зависимости от типа фигуры (класс наследник).
- Расширяемость
- Можно добавлять несколько новых типов, метод не изменяется но изменяется его работа. В расширяемую программу можно добавить новые типы данных и расширить ее потому что типо новые функции будут, вот. Полиморфизм в таком случае осуществляется с помощью наследования

- Переопределение закрытых методов
- Нельзя. И перегрузить тоже.

Поля и статические методы

Полиморфизм поддерживается только для обычных вызовов методов. Обращение к полю обработано на стадии компиляции -> надо делать поля приватными, для базы и ребенка разные имена , доступ через геторы сеторы



Пример как не надо рассмотрен в namespace

- При переопределение завершающего метода в базовом классе важно выполнить завершение производного
- Порядок завершения обратен порядку инициализации
- Что будет если вызвать в конструкторе динамически связанный метод объекта - объект не будет до конца построенным, может привести к использованию неиниц переменных
- Процесс инициализации : память под объект заполняется двоичными нулями, конструкторы базы вызываются в описанном ранее порядке, вызываются иниц членов класса, исполняется тело конструктора производного класса
- Методы, которые отноительно безопасно вызвать в конструкторе - неизменные, методы базового класса

- Ковариантность возвраемых типов - переопределенный метод производного класса может вернуть тип, производный от того, что возвращается базой
- Лучше сначала использовать композицию, чем полиморфизм

- Нисходящее преобразование и динамическое определение типов
- Гарантие правильности нисодящего преобразования необходима, тк в этом случае можно получить больше информации (методы расширяются)
- В джава все преобразования контролируются, процесс проверки - динамическое определение типов
- Есть два наследника но метод foo только у одного. Работаем через родителя. При вызове foo ошибка компиляции