1. Анонимные классы:

<https://docs.oracle.com/javase/tutorial/java/javaOO/anonymousclasses.html>

Анонимные классы позволяют сделать ваш код более кратким. Они позволяют вам объявить и создать экземпляр класса одновременно. Они похожи на локальные классы, за исключением того, что у них нет имени. Используйте их, если вам нужно использовать локальный класс только один раз.

В то время как локальные классы являются объявлениями классов, анонимные классы являются выражениями, что означает, что вы определяете класс в другом выражении.

Как упоминалось, анонимный класс — это выражение. Синтаксис выражения анонимного класса аналогичен вызову конструктора, за исключением того, что определение класса содержится в блоке кода.

Как и локальные классы, анонимные классы могут захватывать переменные; они имеют одинаковый доступ к локальным переменным объемлющей области видимости:

Анонимный класс имеет доступ к членам окружающего его класса.

Анонимный класс не может получить доступ к локальным переменным в своей охватывающей области, которые не объявлены как окончательные или фактически окончательные.

Подобно вложенному классу, объявление типа (например, переменной) в анонимном классе скрывает любые другие объявления во внешней области видимости, имеющие то же имя, т.е. «Затенение».

Анонимные классы также имеют те же ограничения, что и локальные классы, в отношении своих членов:

- Вы не можете объявлять статические инициализаторы или интерфейсы-члены в анонимном классе.

- Анонимный класс может иметь статические члены при условии, что они являются постоянными переменными.

Обратите внимание, что вы можете объявить следующее в анонимных классах:

- Поля

- Дополнительные методы (даже если они не реализуют никаких методов супертипа)

- Инициализаторы экземпляра

- Местные классы

- Также вы не можете объявлять конструкторы в анонимном классе.

1. Перечислительные типы (enums):

<https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html>

Тип перечисления — это специальный тип данных, который позволяет переменной быть набором предопределенных констант. Переменная должна быть равна одному из предопределенных для нее значений. Общие примеры включают направления по компасу (значения СЕВЕР, ЮГ, ВОСТОК и ЗАПАД) и дни недели.

Поскольку они являются константами, имена полей перечисляемого типа пишутся прописными буквами.

В языке программирования Java тип перечисления определяется с помощью ключевого слова enum.

Вы должны использовать типы enum каждый раз, когда вам нужно представить фиксированный набор констант. Это включает в себя естественные типы перечислений, такие как планеты в нашей солнечной системе и наборы данных, где вы знаете все возможные значения во время компиляции — например, выбор в меню, флаги командной строки и так далее.

Типы enum языка программирования Java намного мощнее, чем их аналоги в других языках. Объявление перечисления определяет класс (называемый типом перечисления). Тело класса enum может включать методы и другие поля. Компилятор автоматически добавляет некоторые специальные методы при создании перечисления. Например, у них есть метод статических значений, который возвращает массив, содержащий все значения перечисления в порядке их объявления. Этот метод обычно используется в сочетании с конструкцией for-each для перебора значений перечислимого типа.

Java требует, чтобы константы были определены первыми, до любых полей или методов. Также при наличии полей и методов список констант перечисления должен заканчиваться точкой с запятой.

Примечание. Все перечисления неявно расширяют java.lang.Enum. Поскольку класс может расширять только одного родителя (см. Объявление классов), язык Java не поддерживает множественное наследование состояния (см. Множественное наследование состояния, реализации и типа), поэтому перечисление не может расширять ничего другого.

Примечание. Конструктор для перечислимого типа должен быть закрытым для пакета или с закрытым доступом. Он автоматически создает константы, определенные в начале тела перечисления. Вы не можете вызвать конструктор перечисления самостоятельно.