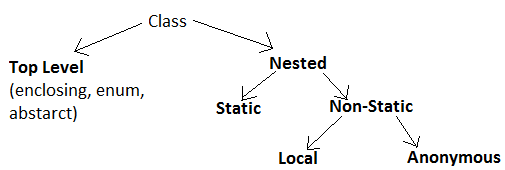
**Lesson 2**

**Виды классов**

Представим дерево классов в таком виде:



Или разобьём их на 5 групп:

1. Top Level
2. Static Nested
3. Non-static nested
4. Local
5. Anonymous

|  |  |  |
| --- | --- | --- |
|  | **Top Level (class)** | **Non-static (внутренний)** |
| Определение | Класс, который объявляется сразу в пакете. | Класс, определение которого содержится внутри другого класса. |
| Ключевые слова | 1. public, friendly 2. final, abstract, strictfp | 1. Все модификаторы доступа 2. **Не** может быть native, transient, synchronized, volatile |
| Область видимости | Public – доступен в любом пакете  Friendly – доступен внутри пакета | Public – доступен в любом пакете  Friendly – доступен внутри пакета  Protected – доступен внутри пакета и в наследниках  Private – доступен только внутри класса |
| К чему имеют доступ | 1. Получить доступ к элементам **внутреннего** класса можно только создав объект. А доступ к переменной final static можно через имя класса. 2. Вызвать статический метод **вложенного** можно через указание полного пути | Имеют прямой доступ к полям и методам внешнего класса через ссылку this |
| Можно ли создать объект напрямую | A a = new A() | Создание объекта:  Outer.Inner obj = new Outer().new Inner(); |
| От кого могут наследоваться, кого реализовывать | Без ограничений | Могут быть производными, базовыми в пределах внешнего, реализовывать интерфейсы |
| this  super | This – ссылка на самого себя  Super – ссылка на базовый класс | This – ссылка на самого себя  Super – ссылка на базовый класс  Имя\_внешнего\_класса.this – ссылка на внешний класс |
| Во что компилируются | В class . Сколько классов объявлено, столько и скомпилируется. | В байткоде внешнего класса будет отмечен как inner class. В собственном байткоде (причём файл имеет следующее название Car$MyClass.class) будет указана ссылка на sourcefile на внешний класс, и везде используется обозначение **Car$MyClass** |
| Когда используются | Всегда | Когда является неотъемлемой составляющей (подводная лодка –> двигатель для подводной лодки). Для хорошего singleTone |
| Примеры кода |  |  |
| Замечания |  | 1. Не содержат static полей, кроме final static |

|  |  |  |
| --- | --- | --- |
|  | **Static (вложенный)** | **Local (локальный)** |
| Определение | Внутренний static класс. | Внутренний класс, объявленный в методе или логическом блоке. |
| Ключевые слова | 1. Обязательно static 2. Все модификаторы доступа 3. Не может быть native, transient, synchronized, volatile | 1. Не имеет модификаторов доступа 2. Может быть strictfp, final, adbstract |
| Область видимости | Public – доступен в любом пакете  Friendly – доступен внутри пакета  Protected – доступен внутри пакета и в наследниках  Private – доступен только внутри класса | В данном логическом блоке или методе |
| К чему имеют доступ | Имеет прямой доступ к static полям и методам внешнего класса.  Для доступа к **не** static полям и методам внешнего должен создавать объект внешнего. | Имеют прямой доступ к полям и методам внешнего класса через ссылку this + к локальным переменным. |
| Можно ли создать объект напрямую | Outer.Inner obj = new Outer.Inner() | Нет |
| От кого могут наследоваться, кого реализовывать | Может быть базовым, производным и реализовывать интерфейс. | Может быть базовым, производным и реализовывать интерфейс в пределах «{ …. }» |
| this  super | Не имеет указателя this | This – ссылка на самого себя  Super – ссылка на базовый класс  Имя\_внешнего\_класса.this – ссылка на внешний класс |
| Во что компилируются | В байткоде внешнего класса будет отмечен как inner class. В собственном байткоде (причём файл имеет следующее название Car$MyClass.class) будет указана ссылка на sourcefile на внешний класс, и везде используется обозначение Car$MyClass | Пусть в классе А есть 2 локальных метода с названием B (конфликта не будет, так как они объявлены в разных местах) и локальный С. При компиляции создастся 3 файла class с именем A$1B.class , A$2B.class и A$1C.class |
| Когда используются | Предоставляет услугу внешнему классу. Для singleTone | История умалчивает |
| Примеры кода |  |  |
| Замечания | 1. Подкласс вложенного класса не наследует возможность доступа к членам внешнего класса, которым наделен суперкласс. 2. Класс вложенный в интерфейс static по умолчанию. 3. Можно объявить в классе, интерфейсе, абстрактном классе | 1. Не содержат static полей, кроме final static |

|  |  |  |
| --- | --- | --- |
|  | **Анонимный** | **Перечисления** |
| Определение | Расширяет другой класс или реализует интерфейс при объявлении одного единственного объекта. | Классы специального вида, которые: |
| Ключевые слова | Нет | Public – доступен в любом пакете  Friendly – доступен внутри пакета |
| Область видимости | Нет |  |
| К чему имеют доступ | **Не** имеет доступ к private |  |
| Можно ли создать объект напрямую | Нет. Объявление анонимного класса выполняется одновременно вместе с созданием объекта с помощью оператора new. | Нельзя создать через new. Объект перечисления создаётся один раз при загрузке перечисления в память. |
| От кого могут наследоваться, кого реализовывать | Нет | Не могут иметь наследников.  Наследуются от java.lang.Enum и реализуют Comparable и Serializable. |
| this  super | This – ссылка на себя  Super – ссылка на класс, который переопределяем |  |
| Во что компилируются | В отдельный файл с названием Main$1.class |  |
| Когда используются | Когда необходимо для одного объекта переопределить метод.  Для Runnable, Comparator, лямбда-выражения | Предоставляют типизированный, безопасный способ задания фиксированных наборов значений. |
| Примеры кода | **Runnable**    **Comparator** | **Анонимные перечисления** ( перечисления, в которых отдельные элементы могут реализовывать свое собственное поведение.) |
| Замечания | 1. Конструктор ни определить, ни переопределить нельзя. 2. Допускается вложенность друг в друга. 3. По ссылке ничего не сможем получить, чего нет в базовом 4. Если анонимный класс определяется не на классе, а на интерфейсе, то нужно переопределить все методы | 1. Элементы перечисления по умолчанию public static final 2. Если содержит абстрактный метод, то должен переопределить 3. Могут содержать вложенные классы |

|  |  |
| --- | --- |
|  | **Параметризованный** |
| Определение | С помощью шаблонов можно создать параметризованные классы и методы, что позволяет использовать более строгую типизацию |
| Ключевые слова |  |
| Область видимости | Public – доступен в любом пакете  Friendly – доступен внутри пакета |
| К чему имеют доступ |  |
| Можно ли создать объект напрямую |  |
| От кого могут наследоваться, кого реализовывать |  |
| this  super |  |
| Во что компилируются | .class |
| Когда используются |  |
| Примеры кода |  |
| Замечания | 1. Если не указать параметр, то компилятор подставит Object 2. Чтобы расширить возможности параметризованных членов используется extends |

Замечание: глубина вложенности ограничивается длиной имени файла.