<https://habr.com/ru/post/439648/>

Класс называется вложенным (nested), если он определен внутри другого класса.

То есть класс просто определен внутри другого, даже не важно статически определен или не статически. Вложенный класс создается для того, чтобы обслуживать окружающий его класс. Если вложенный класс оказывается полезен в каком-либо ином контексте, он должен стать классом верхнего уровня.  
  
*Применение*  
  
Вложенные классы применяются в тех случаях, когда нужно написать небольшой вспомогательный код для другого класса. Вложенный класс создают также, чтобы скрыть его переменные и методы от внешнего мира. Таким образом, вложенный класс еще один элегантный способ ограничения области видимости.

В Java существуют 4 типа вложенных (nested) классов:

1. Статические вложенные, объявленные статически, называются вложенными статическими классами.
2. Внутренние классы — когда объект внутреннего класса связан с объектом обрамляющего класса. Не статические вложенные классы называются внутренними классами, если они связанны с внешним классом.
3. Локальные классы — объявленные внутри блока кода и не являющиеся членом обрамляющего класса. В этом случае можно рассматривать класс как локальную переменную типа класс.
4. Анонимные классы – наследуемые, от какого либо класса, классы в которых при объявлении не задано имя класса.

Существует четыре категории вложенных классов:

1. Статические вложенные классы и не статические вложенные классы. Вложенные классы, объявленные статически, называются вложенными статическими классами.

### Статический вложенный класс (StaticInnerClass)

public class OuterClass {

public static class StaticInnerClass{

}

}

Мы не имеем доступа к не статическому полю внешнего класса, через статический контекст вложенного класса. Это подобно тому, как мы не имеем доступа из статического метода к нестатическим переменным класса. Точно также из статического вложенного класса мы не имеем доступа к нестатическим переменным внешнего класса.  
  
Но мы имеем доступ к приватным статическим полям внешнего класса из вложенного статичного класса.

<https://habr.com/ru/post/439648/>

### Вложенный/внутренний класс (InnerClass) — когда объект внутреннего класса связан с объектом обрамляющего класса. Не статические вложенные классы называются внутренними классами, если они связанны с внешним классом.

public class OuterClass {

public class InnerClass{

}

}

1. Они существуют только у объектов, потому для их создания нужен объект.
2. Внутри Java класса не может быть статических переменных. Если вам нужны какие-то константы или что-либо еще статическое, выносить их нужно во внешний класс. Это связано с тесной связью нестатического вложенного класса с внешним классом.
3. У класса полный доступ ко всем приватным полям внешнего класса. Данная особенность работает в две стороны.
4. Можно получить ссылку на экземпляр внешнего класса. Пример: Airplane.this – ссылка на самолет, this – ссылка на крыло.

**Причины использования вложенных классов (Nesred Classes)  
Зачем использовать вложенные/внутренние классы?**  
  
Причины использования вложенных классов такие. Если класс полезен только для одного другого класса, то вполне логично встроить его в этот класс и хранить их вместе. Использование вложенных классов увеличивает инкапсуляцию. Рассмотрим два класса верхнего уровня, A и B, где B нужен доступ к членам, которые иначе были бы объявлены закрытыми.Скрывая класс «B» в пределах класса «А», члены класса «А» могут быть объявлены закрытыми, и «B» может получить доступ к ним. Кроме того, сам «B» может быть скрыт от внешнего мира.

Отличие в том, что внутренний класс связан с внешним классом через экземпляр, или через объект класса.Чтобы создать экземпляр внутреннего класса, нам нужно сначала создать экземпляр внешнего класса. Затем создать внутренний объект, в пределах внешнего объекта, таким образом:

Outer outerObject = new Outer();

Outer.InnerClass innerObject = outerObject.new InnerClass(); *// создание экземпляра* внутреннего класса

Внутренние классы есть смысл использовать, если они будут использовать элементы родителя,  
чтобы не передавать лишнего в конструкторах.

**Из него видны:**все (даже private) свойства и методы OuterClassа обычные и статические.

**Может содержать:**  
— только обычные свойства и методы (не статические).

Внутренний класс неявно наследуется от внешнего класса, хотя мы не используем ключевое слово **extends**в случае с классом или **implements**в случае с интерфейсом. То есть, во внутреннем классе мы можем использовать весь унаследованный функционал внешнего класса. Может показаться, что это сомнительно. Но это дает нам более гибкий подход. Таким образом мы можем использовать во внутреннем классе, функционал унаследованный от внешнего класса.

Внутренний класс ведет себя как обычный класс за тем исключением, что его объекты могут быть созданы только внутри внешнего класса.

Внутренний класс имеет доступ ко всем полям внешнего класса, в том числе закрытым с помощью модификатора private. Аналогично внешний класс имеет доступ ко всем членам внутреннего класса, в том числе к полям и методам с модификатором private.

Ссылку на объект внешнего класса из внутреннего класса можно получить с помощью выражения Внешний\_класс.this, например, Person.this.

Объекты внутренних классов могут быть созданы только в том классе, в котором внутренние классы опеределены. В других внешних классах объекты внутреннего класса создать нельзя.

«Каждый внутренний класс может независимо наследовать определенную реализацию.  
  
Внутренний класс не ограничен при наследовании в ситуациях, где внешний класс уже наследует реализацию.»

### Локальный класс (LocalClass) — объявленные внутри блока кода и не являющиеся членом обрамляющего класса. В этом случае можно рассматривать класс как локальную переменную типа класс.

public class OuterClass {

public void someMethod(){

class LocalClass{

}}}

**Локальный класс** — это вложенный класс, объявленный внутри другого класса и некоторого блока кода этого класса, то есть объявленный между фигурными скобками {}. Этот блок может быть статическим блоком, циклом, телом if и т.д.

Локальный класс объявленный внутри блока кода другого класса не является членом класса, к которому относится блок, а принадлежит самому блоку, точно так же, как обычная локальная переменная. Такие классы недоступны за пределами внешнего класса, поскольку нет никаких способов обращения к ним, но их экземпляры – это обычные объекты, которые позволяется, например, передавать в качестве аргументов или возвращать из методов.  
Время жизни локального внутреннего класса, это время пока существует хотя бы одна ссылка на него. Такой класс существует внутри блока кода и время его жизни ограниченно этим блоком.

### Анонимный класс (имени нет)– наследуемые, от какого либо класса, классы в которых при объявлении не задано имя класса. *В лабе использовался для перенаправления записи логера.*

Локальный класс без имени. Наследует какой-то класс, или имплиментирует какой-то интерфейс.

public class OuterClass {

public void someMethod(){

Callable callable = new Callable() {

@Override

public Object call() throws Exception {

return null;

}

};

}

}