# 嵌套类_局部类

@M了个J
李明杰

https://github.com/CoderMJLee
http://cnblogs.com/mjios

码拉松

# 嵌套类（Nested Class）

- 嵌套类：定义在另一个类中的类

```
public class OuterClass {
    // 静态嵌套类
    static class StaticNestedClass {

    }
    // 非静态嵌套类（内部类）
    class InnerClass {

    }
}
```

- 在嵌套类外层的类，称为：外部类（Outer Class）

- 最外层的外部类，称为：顶级类（Top-level Class）

# 内部类（Inner Class）

- 内部类：没有被 `static` 修饰的嵌套类，非静态嵌套类

- 跟实例变量、实例方法一样，内部类与外部类的实例相关联
  - 必须先创建外部类实例，然后再用外部类实例创建内部类实例
  - 内部类不能定义除编译时常量以外的任何 `static` 成员

- 内部类可以直接访问外部类中的所有成员（即使被声明为 `private`）

- 外部类可以直接访问内部类实例的成员变量、方法（即使被声明为 `private`）

小码哥教育 SEEMYGO

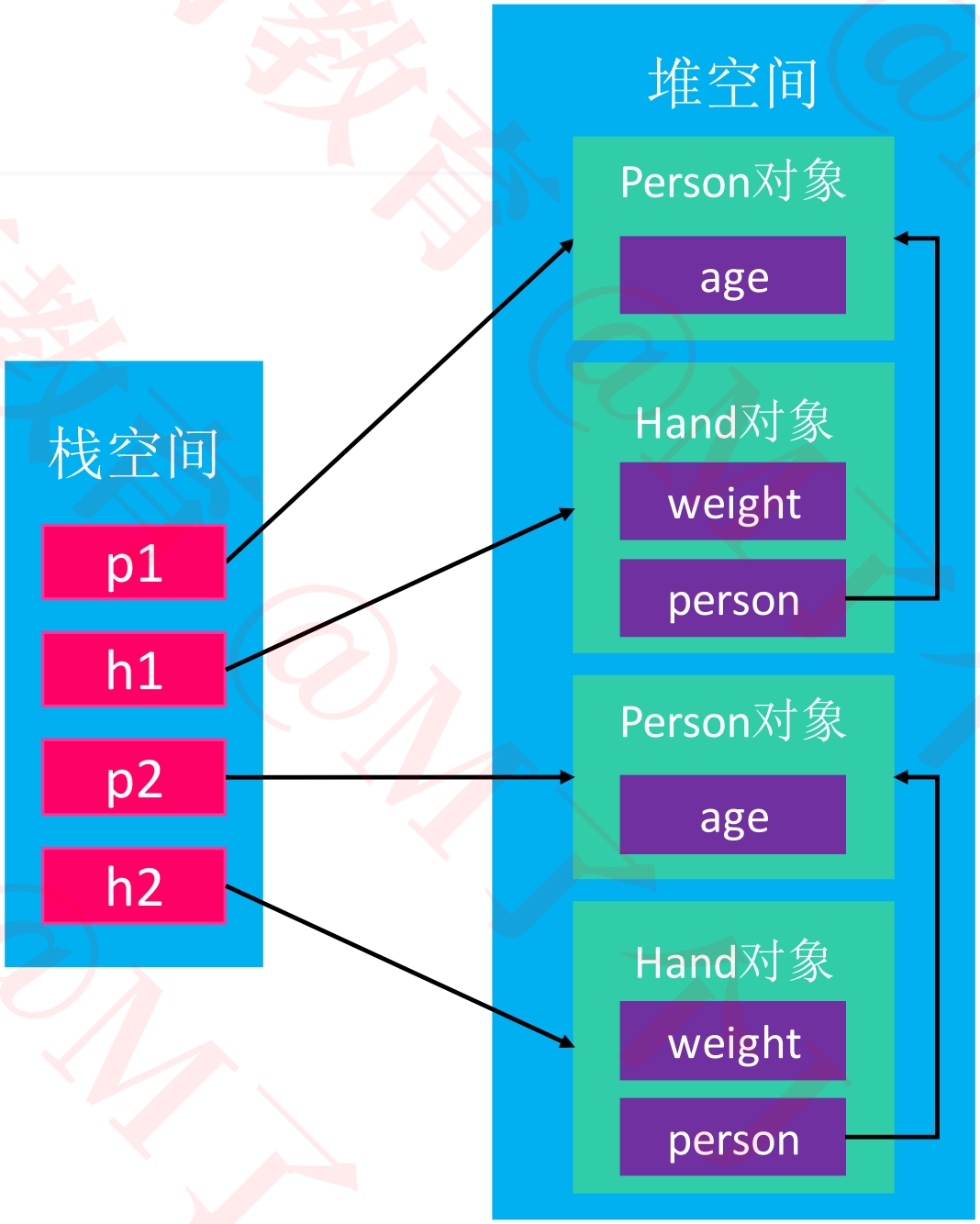
# 内部类举例

```
public class Person {
    private int age;

    public class Hand {
        private int weight;
    }
}

Person p1 = new Person();
Hand h1 = p1.new Hand();

Person p2 = new Person();
Hand h2 = p2.new Hand();
```



小码哥教育 SEEMYGO

# 内部类举例

```
public class Company {
    private String name;
    public Company(String name) {
        this.name = name;
    }
    public void fire(Employee e) {
        System.out.println(name + " fire " + e.no);
    }

    public class Employee {
        private int no;
        public Employee(int no) {
            this.no = no;
        }
        public void show() {
            System.out.println(name + " : " + no);
        }
    }
}
```

```
Company c = new Company("Google");
Employee e = c.new Employee(1);
e.show(); // Google : 1
c.fire(e); // Google fire 1
```

小码哥教育 SEEMYGO

# 内部类细节

```
public class OuterClass {
    private int x = 1;

    public class InnerClass {
        private int x = 2;
        public void show() {
            System.out.println(x);
            System.out.println(this.x);
            System.out.println(OuterClass.this.x);
        }
    }
}

new OuterClass().new InnerClass().show();
// 2 2 1
```

# 静态嵌套类（Static Nested Class）

- 静态嵌套类：被 `static` 修饰的嵌套类

- 静态嵌套类在行为上就是一个顶级类，只是定义的代码写在了另一个类中

- 对比一般的顶级类，静态嵌套类多了一些特殊权限
  - 可以直接访问外部类中的成员（即使被声明为 `private`）

小码哥教育 SEEMYGO

# 什么情况使用嵌套类?

- 如果类 A 只用在类 C 内部，可以考虑将类 A 嵌套到类 C 中
  - 封装性更好
  - 程序包更加简化
  - 增强可读性、维护性

- 如果类 A 需要经常访问类 C 的非公共成员，可以考虑将类 A嵌套到类 C 中
  - 另外也可以根据需要将类 A 隐藏起来，不对外暴露

- 如果需要经常访问非公共的实例成员，设计成内部类（非静态嵌套类），否则设计成静态嵌套类
  - 如果必须先有 C 实例，才能创建 A 实例，那么可以将 A 设计为 C 的内部类

# 局部类（Local Class）

- 局部类：定义在代码块中的类（可以定义在方法中、for 循环中、if 语句中等）

- 局部类不能定义除编译时常量以外的任何 static 成员

- 局部类只能访问 final 或者 有效 final 的局部变量
  - 从 Java 8 开始，如果局部变量没有被第二次赋值，就认定为是有效 final

- 局部类可以直接访问外部类中的所有成员（即使被声明为 private）
  - 局部类只有定义在实例相关的代码块中，才能直接访问外部类中的实例成员（实例变量、实例方法）

小码哥教育 SEEMYGO

# 局部类举例

```
public class TestLocalClass {
    private int a = 1;
    private static int b = 2;
    private static void test1() {}
    private void test2() {}

    public void test3() {
        int c = 2;

        class LocalClass {
            static final int d = 4;
            void test4() {
                System.out.println(a + b + c + d);
                test1();
                test2();
            }
        }

        new LocalClass().test4();
    }
}
```