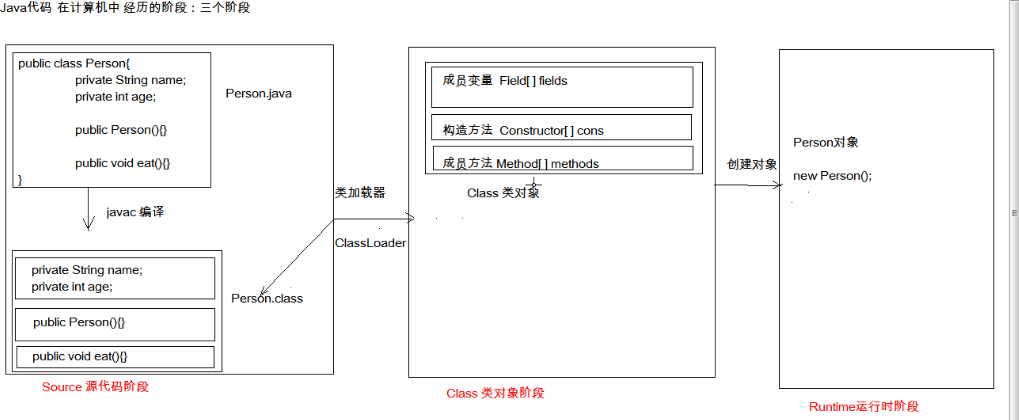
# 反射（框架设计的灵魂）

框架：半成品软件，可以在框架的基础上进行软件的开发，简化编码

反射：将类的各个组成部分封装为其他对象，这就是反射机制

1. 反射的好处：
   1. 可以在程序运行过程中，操作这些对象
   2. 可以解耦，提高程序的可扩展性
   3. 请看图
   4. 
   5. 代码的运行的四个阶段
      1. Source源代码阶段，会把代码编写成.class文件
      2. Class类对象阶段，会把成员变量 构造方法 普通方法都分开 为三个区域 并把他们放入三个数组中
      3. 最后就是 我们可以创建的问价 如：new 类名();
      4. 使用反射就像 我们在对象点类名是 出现的方法名一样，这底层就是使用反射

获取Class对象的三种方式：（返回的都是class文件，class是一个类）

1. Class.forName（“全类名”）：将字节文件加载进内存，返回class文件
   * 1. 多用于配置文件，将类名定义在配置问价中。读取问价，加载类
2. 类名.class:通过类名的属性class获取
   * 1. 多用于参数的传递
3. 对象.getclass（）：getclass（）方法在object类中定义了（意味着所有类都可以调用）
   * 1. 多用于对象的获取字节码的方式
   1. 他们的结果是一样的，但是两个类点出class 是不一样的
4. 结论：
   1. 同一个字节码文件（\*.class）在一次程序运行过程中，只会加载一次，不论通过哪种方式获取的Class对象都是同一个。（想要比较他们的对象是否相同用==来比较）

### Class对象功能：(class 类中的方法可以获取 对象的注解)

获取功能：

1. 获取成员变量们
   1. Field[] getFields();获取所有public 修饰的成员变量
   2. Field getFiel(String name);获取指定名称的public修饰的成员变量
   3. Field[] getDeclaredFields();获取所有成员的变量，不考虑修饰符（要是设置暴力反射为true）
   4. Field getDeclaredField(String name)获取指定名称的的成员变量，不考虑修饰符
      1. 上面是获取成员变量（对象的反射class文件） 下面是操作成员变量
      2. Field类中的成员方法：操作类（对应类 那个类调用他 他就是那个类）中成员变量
         1. 操作：（Obj 代表的是对应类的对象（需要创建对象），value是传值）
            1. 设置值;void set(Object obj Object value)也可以设置指定的值 内有其他方法例如 setChar() setLong()等等
            2. 获取值：get(Object obj)（这里的obj需要对应类的对象，就是上面创建的对象，不能新建）
            3. 忽略访问权限修饰的安全检查（也叫暴力反射）

setAccessuble

1. 获取构造器们
   1. Constructor<?>[] getConstructors()获取构造器 并用Constructor返回对应对象有多个构造器 就返回多个 ，只能访问公共的构造器
   2. Constructor<T> getConstructor(类<?>....ParameterTypes)和上一样 ，不过这里返回的是指定参数的构造器，这里的点点点代表的是类型的class文件如int.class,String.class 只能访问公共的构造器
   3. Constructor<T>[] getDeclaredConstructor(类<?>....parameterTypes)和上一样 ，不过这里返回的是指定参数的构造器，这里的点点点代表的是类型的class文件如int.class,String.class 不指定修饰符（需要暴力反射Constructor.setAccessuble）
   4. Constructor<?> getDeclaredConstructors()获取构造器 并用Constructor返回对应对象有多个构造器 就返回多个 ，不指定修饰符（需要暴力反射Constructor.setAccessuble）
      1. 上面是获取成员变量（对象的反射class文件） 下面是操作成员变量
      2. Constructor操作构造方法：给构造方法设置值 获取构造方法，这个构造方法可以操作对象中的值，也就是创建对象
         1. 创建对象：
            1. T newInstance(Object .....intargs),设置值，这里的点点点代表的是类型的class文件，如int.class,String.class等等 这里返回的是对应的对象 他会返回一个新的对象
            2. 如果使用空参构造方法创建对象，操作可以简化：Class对象的newInstance方法 他会返回一个新的对象
            3. getParameterCount() 获取构造器参数的个数
            4. getParameterTypes() 返回一个数组 数组中存放的是构造器参数类型，如class java.lang.String ....
2. 获取成员方法们：
   1. Method[] getMethods()获取所有的方法，只获取公共的，还有父类的重写方法
   2. Method getMethods(String name ,类<?>.......parameterTypes)获取指定名称的方法和指定方法参数（参数用class.int class.String），只获取公共的
   3. Method[] getDeclaredMethods()获取所有的方法，获取所有的方法，还有父类的重写方法
   4. Method getDeclaredMethd(String name,类<?>.....parameterTypes)获取指定名称的方法和指定方法参数（参数用class.int class.String），获取所有的方法
      1. Method运行方法
         1. 执行方法：
            1. Object inovke( Object obj,Object.....args)obj代表的是方法名称，args代表的是方法的参数（和上面的class.int String.class 对应）
3. 获取类名
   1. String getName()获取类的名称和他的包名

### 案例

需求：写一个“框架”，不能修改类的任何代码的前提下，可以帮我们创建任意类的对象，并且执行其中任意方法

实现：

配置文件

反射

步骤：

将需要创建的对象的全类名和需要执行的方法定义在配置文件中

在程序中加载读取配置文件

使用反射技术来加载文件进内存

创建对象

执行方法

需要用到和IO流相关的集合properties集合 读取文件内容

获取class目录下的配置文件 （配置文件后缀.properties）

本类 变量名=本类.class.getClassLoder();

获取class目录下的配置文件

变量名.getResourceAsStream(配置文件的名称)