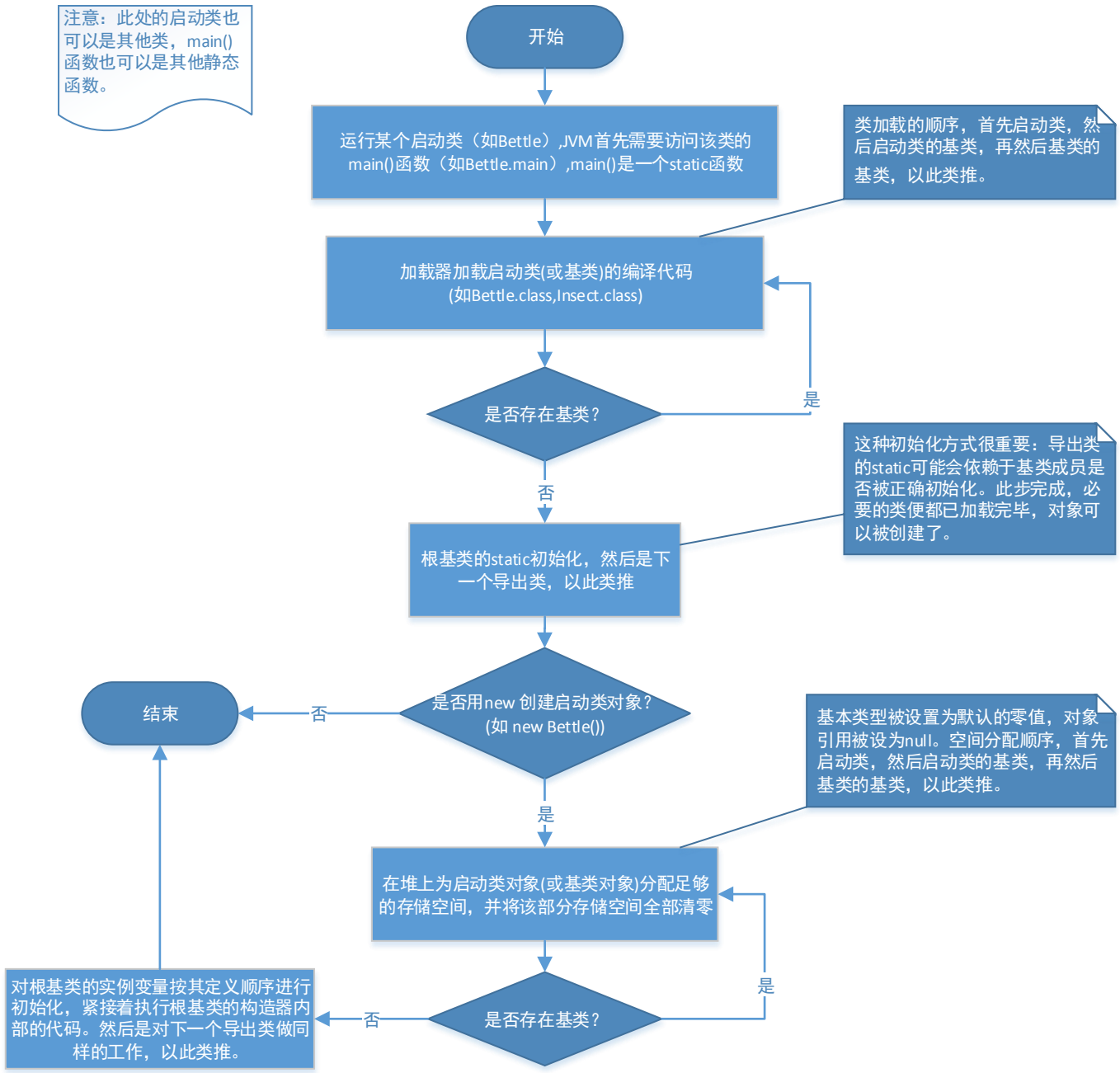
注意：此处的启动类也可以是其他类，main()函数也可以是其他静态函数。
开始
运行某个启动类（如Bettle）,JVM首先需要访问该类的main()函数（如Bettle.main）,main()是一个static函数
类加载的顺序，首先启动类，然后启动类的基类，再然后基类的基类，以此类推。
加载器加载启动类(或基类)的编译代码(如Bettle.class,Insect.class)
是否存在基类？
是
否
这种初始化方式很重要：导出类的static可能会依赖于基类成员是否被正确初始化。此步完成，必要的类便都已加载完毕，对象可以被创建了。
根基类的static初始化，然后是下一个导出类，以此类推
是否用new 创建启动类对象？(如 new Bettle())
否
结束
是
基本类型被设置为默认的零值，对象引用被设为null。空间分配顺序，首先启动类，然后启动类的基类，再然后基类的基类，以此类推。
在堆上为启动类对象(或基类对象)分配足够的存储空间，并将该部分存储空间全部清零
是
是否存在基类？
否
对根基类的实例变量按其定义顺序进行初始化，紧接着执行根基类的构造器内部的代码。然后是对下一个导出类做同样的工作，以此类推。