(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116166318 A
(43)申请公布日 2023.05.26

(21)申请号 202111415894.7
(22)申请日 2021.11.25
(71)申请人 中移(苏州)软件技术有限公司
地址 215163 江苏省苏州市高新区昆仑山路58号1幢中移软件园
申请人 中国移动通信集团有限公司
(72)发明人 王伟伟
(74)专利代理机构 北京派特恩知识产权代理有限公司 11270
专利代理师 李路遥 张颖玲
(51)Int.Cl.
G06F 8/72(2018.01)
G06F 8/71(2018.01)
G06F 8/33(2018.01)
G06F 8/30(2018.01)
G06F 16/903(2019.01)

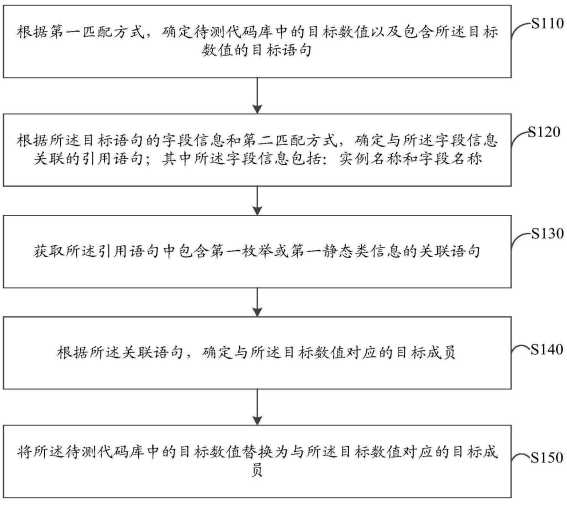
权利要求书2页 说明书12页 附图5页

(54)发明名称

代码中目标数值的处理方法、装置、设备和存储介质

(57)摘要

本发明实施例公开了一种代码中目标数值的处理方法、装置、设备和存储介质。方法包括：根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句；根据目标语句的字段信息和第二匹配方式，确定与字段信息关联的引用语句；字段信息包括：实例名称和字段名称；获取引用语句中包含第一枚举或第一静态类信息的关联语句；根据关联语句，确定与目标数值对应的目标成员；将待测代码库中的目标数值替换为与目标数值对应的目标成员。本发明在代码检查过程中通过自动识别代码中的魔法值，并在代码中已存在该魔法值对应的枚举值或静态成员时，进行自动进行替换，提高了代码编译成功率，减少冗余代码量，降低人工核查的工作量。



CN 116166318 A

1.一种代码中目标数值的处理方法,其特征在于,所述方法包括:

根据第一匹配方式,确定待测代码库中的目标数值以及包含所述目标数值的目标语句;

根据所述目标语句的字段信息和第二匹配方式,确定与所述字段信息关联的引用语句;其中所述字段信息包括:实例名称和字段名称;

获取所述引用语句中包含第一枚举或第一静态类信息的关联语句;

根据所述关联语句,确定与所述目标数值对应的目标成员;

将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

2.根据权利要求1所述的方法,其特征在于,所述根据所述目标语句的字段信息和第二匹配方式,确定与所述字段信息关联的引用语句,包括:

根据所述目标语句中的实例名称,在所述待测代码库中确定与所述实例名称对应的类相关联的代码块;

在所述代码块中筛选与所述实例名称和所述字段名称关联的引用语句。

3.根据权利要求1所述的方法,其特征在于,所述方法还包括:

若所述关联语句中没有与所述目标数值对应的目标成员,则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

4.根据权利要求1所述的方法,其特征在于,所述方法还包括:

根据所述目标语句的字段信息,在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句;其中所述标注语句中包含第二枚举或第二静态类信息;

确定所述标注语句中与所述目标数值对应的目标成员。

5.根据权利要求4所述的方法,其特征在于,所述确定所述标注语句中与所述目标数值对应的目标成员,包括:

若所述第二枚举或第二静态类信息中没有与所述目标数值对应的目标成员,则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

6.根据权利要求1所述的方法,其特征在于,所述根据所述关联语句,确定与所述目标数值对应的目标成员,包括:

若存在与所述目标数值对应的多个目标成员,则输出提示信息;其中,所述提示信息用于指示用户确定第一目标成员,并删除除所述第一目标成员以外的其他目标成员;所述第一目标成员为所述多个目标成员中的任一目标成员。

7.一种代码中目标数值的处理装置,其特征在于,所述装置包括:

查找单元,用于根据第一匹配方式,确定待测代码库中的目标数值以及包含所述目标数值的目标语句;

第一确定单元,用于根据所述目标语句的字段信息和第二匹配方式,确定与所述字段信息关联的引用语句;其中所述字段信息包括:实例名称和字段名称;

获取单元,用于获取所述引用语句中包含第一枚举或第一静态类信息的关联语句;

替换单元,用于根据所述关联语句,确定与所述目标数值对应的目标成员,以及将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

8.根据权利要求7所述的装置,其特征在于,所述装置还包括:第二确定单元;

所述第二确定单元,用于根据所述目标语句的字段信息,在所述待测代码库中的标注

信息中确定与所述字段信息关联的标注语句,和确定所述标注语句中与所述目标数值对应的目标成员;其中所述标注语句中包含第二枚举或第二静态类信息。

9.一种代码中目标数值的处理设备,包括存储器和处理器,所述存储器存储有可在处理器上运行的程序,其特征在于,所述处理器执行所述程序时实现权利要求1至6任一项所述方法中的步骤。

10.一种计算机可读存储介质,其上存储有计算机程序,其特征在于,所述计算机程序被处理器执行时实现权利要求1至6任一项所述方法中的步骤。

代码中目标数值的处理方法、装置、设备和存储介质

技术领域

[0001] 本发明涉及互联网技术领域，特别涉及一种代码中目标数值的处理方法、装置、设备和存储介质。

背景技术

[0002] 代码中有魔法值会造成代码可读性低(与代码量成正比)。还会造成维护困难，改动一个数值便要大动干戈，牵一发而动全身。因此应当尽力消灭或减少魔法值，提高维护效率和代码可读性。

[0003] 目前的代码检查工具，仅扫描代码中是否含有魔法值，如含有魔法值，则无法通过编译，以此来强制开发人员不在代码中使用魔法值。但是开发人员任然可以通过定义字符串等形式使用魔法值以规避代码检查工具的警告，代码的可读性仍无改善。

[0004] 且，通过集成开发环境(IDE，Integrated Development Environment)的重构功能，能自动为魔法值创建枚举或静态类，如此时已有枚举或静态类定义该魔法值，则仍会再次创建一个类似的枚举或静态类。随着重构的进行，在代码库中，可能会看到“ProjectTypeEnum”，“MyProjectTypeEnum”，“TypeEnum”等类似的枚举，均为项目当前阶段的枚举，通过集成开发环境对魔法值重构仍可能导致代码更加混乱。

[0005] 可见，现有的代码检查工具和集成开发环境的重构功能均无法有效的解决代码中存在魔法值导致的代码可读性低的问题。

发明内容

[0006] 有鉴于此，本发明实施例期望提供一种代码中目标数值的处理方法、装置、设备和存储介质。

[0007] 本发明实施例的技术实施例是这样实现的：

[0008] 本发明实施例提供一种代码中目标数值的处理方法，所述方法包括：

[0009] 根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句；

[0010] 根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句；其中所述字段信息包括：实例名称和字段名称；

[0011] 获取所述引用语句中包含第一枚举或第一静态类信息的关联语句；

[0012] 根据所述关联语句，确定与所述目标数值对应的目标成员；

[0013] 将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

[0014] 在上述方案中，所述根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句，包括：

[0015] 根据所述目标语句中的实例名称，在所述待测代码库中确定与所述实例名称对应的类相关联的代码块；

[0016] 在所述代码块中筛选与所述实例名称和所述字段名称关联的引用语句。

[0017] 在上述方案中，所述方法包括：

[0018] 若所述关联语句中没有与所述目标数值对应的目标成员，则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0019] 在上述方案中，所述方法还包括：

[0020] 根据所述目标语句的字段信息，在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句；其中所述标注语句中包含第二枚举或第二静态类信息；

[0021] 确定所述标注语句中与所述目标数值对应的目标成员。

[0022] 在上述方案中，所述确定所述标注语句中与所述目标数值对应的目标成员，包括：

[0023] 若所述第二枚举或第二静态类信息中没有与所述目标数值对应的目标成员，则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0024] 在上述方案中，所述根据所述关联语句，确定与所述目标数值对应的目标成员，包括：

[0025] 若存在与所述目标数值对应的多个目标成员，则输出提示信息；其中，所述提示信息用于指示用户确定第一目标成员，并删除除所述第一目标成员以外的其他目标成员；所述第一目标成员为所述多个目标成员中的任一目标成员。

[0026] 本发明实施例提供一种代码中目标数值的处理装置，所述装置包括：

[0027] 查找单元，用于根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句；

[0028] 第一确定单元，用于根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句；其中所述字段信息包括：实例名称和字段名称；

[0029] 获取单元，用于获取所述引用语句中包含第一枚举或第一静态类信息的关联语句；

[0030] 替换单元，用于根据所述关联语句，确定与所述目标数值对应的目标成员，以及将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

[0031] 在上述方案中，所述装置还包括：第二确定单元；

[0032] 所述第二确定单元，用于根据所述目标语句的字段信息，在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句，和确定所述标注语句中与所述目标数值对应的目标成员；其中所述标注语句中包含第二枚举或第二静态类信息。

[0033] 本发明实施例提供一种设备，包括存储器和处理器，所述存储器存储有可在处理器上运行的程序，所述处理器执行所述程序时实现上述所述方法的任一步骤。

[0034] 本发明实施例提供一种计算机可读存储介质，其上存储有计算机程序，该计算机程序被处理器执行时实现上述所述方法的任一步骤。

[0035] 本发明实施例提供的代码中目标数值的处理方法、装置、设备和存储介质，本实施例通过自动识别代码中的魔法值，并在代码中已存在该魔法值对应的枚举值或静态成员时，进行自动进行替换，从而代替开发人员手动修改并重新提交的过程，提高了代码编译成功率，减少冗余代码量，降低人工核查的工作量。

附图说明

[0036] 图1为本发明实施例提供的一种代码中目标数值的处理方法实现流程示意图；

[0037] 图2为本发明实施例提供的一种现有的代码检查技术方案原理示意图；

[0038] 图3为本发明实施例提供的智能代码处理方案的原理示意图；

[0039] 图4为本发明实施例提供的另一种智能代码处理方案的原理示意图；

[0040] 图5为本发明实施例提供的一种代码中目标数值的处理装置的组成结构示意图；

[0041] 图6为本发明实施例中提供的设备的一种硬件实体结构示意图。

具体实施方式

[0042] 为使本发明实施例的目的、技术方案和优点更加清楚，下面将结合本发明实施例中的附图，对发明的具体技术方案做进一步详细描述。以下实施例用于说明本发明，但不用来限制本发明的范围。

[0043] 本实施例提出一种代码中目标数值的处理方法，图1为本发明实施例代码中目标数值的处理方法实现流程示意图，如图1所示，该方法包括：

[0044] 步骤S110：根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句；

[0045] 步骤S120：根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句；其中所述字段信息包括：实例名称和字段名称；

[0046] 步骤S130：获取所述引用语句中包含第一枚举或第一静态类信息的关联语句；

[0047] 步骤S140：根据所述关联语句，确定与所述目标数值对应的目标成员；

[0048] 步骤S150：将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

[0049] 在一实施例中，所述目标数值为常量，包括但不限于常数和字符串常量。

[0050] 在一实施例中，所述目标成员包括：枚举值和\或静态成员。

[0051] 在一实施例中，所述第一匹配方式包括但不限于正则匹配法，所述正则匹配法为利用正则表达式的匹配识别方法，其中，本实施例采用的正则表达式包括：左值或右值为常量的正则表达式等式和设置参数为常量的正则表达式。

[0052] 以JAVA编程语言为例，其用于确定目标数值的正则表达等式可为：

[0053] (\S+).get(\w+)\(\)[=|<=|>=](\w+)，或\"[\s|\S]+\").equals\((\S+).get(\w+)\(\)\)(左值或右值为常量的正则表达式)；或(\S+).set(\w+)\((\d+)\)(类型的设置(set)方法中参数为常数)。

[0054] 通过将待测代码库中的代码与上述正则表达式进行匹配，可自动识别出与正则表达式匹配的语句中的目标数值、实例名称和字段名称。本实施例，通过正则匹配的方式进行目标数值的查找，提升了查找待测代码库中目标数值的效率。

[0055] 在一实施例中，可根据代码的撰写规则，确定给出多种可匹配识别出目标数值的正则表达式，减少目标数值的漏查的风险。

[0056] 在一实施例中，第二匹配方式用于确定所述待测代码库中包含所述字段信息的所有代码块。

[0057] 在一实施例中，所述根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句，包括：根据所述目标语句的实例名称或字段名称，在待测代码库中的匹配包含所述实例名称或字段名称的代码块，若该代码块中包含所述实例名称或字段

名称,则匹配成功,代码块中包含该实例名称或字段名称的语句为字段信息对应的引用语句。

[0058] 在一实施例中,获取所述引用语句中包含第一枚举或第一静态类信息的关联语句,可通过正则匹配的方式实现。例如:(\S+).getStatus\(\)[=|<=|>=]([\w|.]+),若通过该正则表达式,匹配到以下结果:

[0059] dto.getStatus()<CertificationStatusEnum.Expired.getCode(),那么,根据该正则可自动识别出dto为实例名称,Status为字段名称,CertificationStatusEnum.Expired.getCode()为枚举类中的枚举值。

[0060] 在获取了包含第一枚举或第一静态类信息的关联语句后,通过比较目标数值和关联语句的枚举或静态类信息,可确定与目标数值等值的枚举值或静态成员。

[0061] 在一实施例中,本实施例提供的代码中目标数值的处理方法,可应用于在集成开发环境(IDE)智能提示中或应用于代码库或持续集成工具中。

[0062] 本实施例,通过自动识别代码中的魔法值,并在代码中已存在该魔法值对应的枚举值或静态成员时,进行自动进行替换,从而代替开发人员手动修改并重新提交的过程,提高了代码编译成功率,减少冗余代码量,降低人工核查的工作量。

[0063] 在一些实施例中,所述根据所述目标语句的字段信息和第二匹配方式,确定与所述字段信息关联的引用语句,包括:

[0064] 根据所述目标语句中的实例名称,在所述待测代码库中确定与所述实例名称对应的类相关联的代码块;

[0065] 在所述代码块中筛选与所述实例名称和所述字段名称关联的引用语句。

[0066] 在一实施例中,具体的,根据通过正则表达式识别出的目标语句中的实例名称,在所述待测代码库中查找所述实例的声明语句,并根据声明语句确定关于所述实例对应的类,以及类的完整包名,在扫描待测代码库中包含该完整包名或部分包名,或包含类的名称的代码块。确定扫描出的代码块所在的上下文中是否包含对应的实例名称和字段名称。

[0067] 在一些实施例中,所述方法包括:

[0068] 若所述关联语句中没有与所述目标数值对应的目标成员,则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0069] 在一实施例中,若在待测代码库中没有与目标数值等值的枚举或静态类信息,则在所述枚举或静态类中自动生成与目标数值等值的枚举值或静态成员,然后再将待测代码库中的目标数值替换为对应的枚举值或静态成员,或输出提示信息,提示用户重新进行代码魔法值的检查。

[0070] 在一些实施例中,在根据第一匹配方式,确定待测代码库中的目标数值以及包含所述目标数值的目标语句后,所述方法还包括:

[0071] 根据所述目标语句的字段信息,在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句;其中所述标注语句中包含第二枚举或第二静态类信息;

[0072] 确定所述标注语句中与所述目标数值对应的目标成员。

[0073] 在一实施例中,标注信息为代码中的注释语句,用于对代码内容进行解释方便对代码的理解。

[0074] 在一实施例中,所述方法还包括:根据预设规则,确定所述代码库中的标注信息。

在一实施例中,所述预设规则与所述代码的代码类型相关。例如JAVA的标注信息是以"@注释名"在代码中存在的,因此预设规则为筛选代码中包含@符号的语句。

[0075] 本实施例,通过查找将字段信息与枚举或静态类关联的注释语句,根据该注释语句,确定与目标数值等值的枚举值或静态成员。相比于对整个待测代码库的扫描,更加直接和快捷。

[0076] 在一些实施例中,所述确定所述标注语句中与所述目标数值对应的目标成员,包括:

[0077] 若所述第二枚举或第二静态类信息中没有与所述目标数值对应的目标成员,则在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0078] 在一实施例中,若标注信息的枚举或静态类中,没有与所述目标数值等值的目标成员,则在所述枚举或静态类中新增与所述目标数值等值的目标成员。

[0079] 在一些实施例中,所述根据所述关联语句,确定与所述目标数值对应的目标成员,包括:

[0080] 若存在与所述目标数值对应的多个目标成员,则输出提示信息;其中,所述提示信息用于指示用户确定第一目标成员,并删除除所述第一目标成员以外的其他目标成员;所述第一目标成员为所述多个目标成员中的任一目标成员。

[0081] 若待测代码库中一个目标数值存在多个枚举值和静态成员与之对应,则提示用户确定一个保留的枚举值或静态成员,删除多余冗杂的枚举或静态类从而优化代码。

[0082] 在一些实施例中,在根据第一匹配方式,确定待测代码库中的目标数值以及包含所述目标数值的目标语句后,先根据所述目标语句的字段信息,在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句,若标注信息中没有与所述字段信息关联的标注语句,则根据所述目标语句的字段信息和第二匹配方式,确定与所述字段信息关联的引用语句;其中所述字段信息包括:实例名称和字段名称;获取所述引用语句中包含第一枚举或第一静态类信息的关联语句;根据所述关联语句,确定与所述目标数值对应的目标成员。由于通过标注信息确定目标数值对应的目标成员的方式更为快捷,如此,通过优先选择扫描标注信息的方式确定目标成员,可进一步的提升代码中目标数值的替换效率,减少不必要的搜索和操作。

[0083] 以下为结合上述实施例的具体示例:

[0084] 魔法值为在代码中直接出现的数值,只有在这个数值记述的那部分代码中才能明确了解其含义。

[0085] 以JAVA代码为例:

[0086] if(“LXPF”.equals(projectDto.segment)){

[0087] //TODO

[0088] }

[0089] projectDto.setSegment(“LXPF”);

[0090] 其中的“LXPF”就是魔法值,if中的条件表示项目的当前阶段为“立项批复(数据库中的缩写为首字母缩写:LXPF)”。对于第一次接触该代码的编程人员,很难理解其含义。

[0091] 经研究,现有代码检查工具仅在代码中出现魔法值时给出警告,而无法实现对代码中魔法值自动识别和替换,如图2所示,图2为一种现有的代码检查技术方案的原理示意

图。且现有集成开发环境(IDE),可在重构时自动为魔法值创建枚举或者静态类,将其作为枚举中的枚举值或静态类中的静态类成员。

[0092] 且开发人员仍可能通过定义如下变量来规避代码检查:

[0093] public class MyConstant{

[0094] public static final String LXPF＝“LXPF”;

[0095] }

[0096] if(MyConstant.LXPF.equals(projectDto.segment)){

[0097] //TODO

[0098] }

[0099] projectDto.setSegment(MyConstant.LXPF);

[0100] 此时虽然该代码不再被代码检查工具警告,但代码的可读性仍无改善,且通过集成开发环境(IDE)对魔法值重构仍可能导致代码更加混乱。

[0101] 基于此,本示例提出通过一种智能化的手段,自动识别代码中的魔法值,如代码中已存在该魔法值对应的枚举值或静态成员,则会自动进行替换,从而代替开发人员手动修改并重新提交的过程的技术方案,如图3所示,图3本示例的智能代码处理方案的原理示意图。

[0102] 本示例提出的一种智能代码处理方法,包括:

[0103] 步骤1:通过正则匹配代码库中的魔法值。

[0104] 具体的,扫描源码中的魔法值,可通过正则表达式等方式进行匹配,例如:匹配等式左值或右值为常数,如:(\S+).get(\w+)\(\)[＝|<＝|>＝](\w+)、(\″[\s|\S]+\″).equals\((\S+).get(\w+)\(\)\);或,类型的set方法中参数为常数,如(\S+).set(\w+)\((\d+)\)。

[0105] 以正则(\S+).get(\w+)\(\)[＝|<＝|>＝](\w+)为例,通过该正则表达式,如匹配到以下结果:dto.getStatus()>＝2,根据该正则可自动识别出dto为实例名称,Status为字段名称,2为魔法值。

[0106] 步骤2:通过上下文查找匹配到的表达式中描述的类和字段名称,并查找代码库中关于实体字段的全部引用。

[0107] 具体的,扫描步骤1中所涉及的包含魔法值的语句中字段对应的类成员是否有其他引用。以步骤1中描述的“dto.getStatus()>＝2”为例,现已知晓dto为实例名称,Status为字段名称,2为魔法值。首先获取该dto实例所在代码块,根据正则匹配查找上下文中形如“Applicant dto＝xxx”的声明语句,获取该类的完整包名,如“com.sample.Applicant”,再扫描源码包中的其他java类,代码中是否包含“com.sample.Applicant”或引用中包含“import com.sample”且代码中包含Applicant,在分析其代码块所在上下文中是否包含其对应的实例化对象,上述过程可通过多个条件组合匹配来实现。

[0108] 步骤2的扫描结果样例如下所示:

[0109] Applicant dto＝projectService.get(id);...

[0110] if(dto.getStatus()>＝CertificationStatusEnum.Issued.getCode()and dto.getStatus()<CertificationStatusEnum.Expired.getCode())...

[0111] Project dto＝projectService.get(id);...

[0112] if(ProjectSegmentEmun.PendingApproval.getAbbr().equals(dto.getSegment()))...

[0113] Applicant dto＝new Applicant();...

[0114] dto.setStatus(CertificationStatusEnum.Issued.getCode());...

[0115] Project dto＝new Project();...

[0116] dto.setSegment(ProjectSegmentEmun.PendingApproval.getAbbr());...

[0117] 步骤3:遍历步骤2中的全部引用,找出相关的枚举或静态类。

[0118] 具体的,在步骤2的扫描结果中,再通过正则匹配相关的比较或赋值语句,如(\S+).getStatus\(\)[＝|<＝|>＝]([\w|.]+),通过该正则表达式,如匹配到以下结果:dto.getStatus()<CertificationStatusEnum.Expired.getCode(),根据该正则可自动识别出dto为实例名称,Status为字段名称,CertificationStatusEnum.Expired.getCode()为枚举类中的枚举值。

[0119] 例如:“Applicant dto＝projectService.get(id);...if(dto.getStatus()＝＝1)...”中使用了魔法值,如需进行替换,首先要找到实例dto对应的类,代码如下:

[0120] package com.sample

[0121] ....

[0122] public class Applicant{

[0123] private Long id;

[0124] private String firstName;

[0125] private String lastName;

[0126] ....

[0127] private Byte status;

[0128] ....

[0129] //Constructors,Getter,Setter

[0130] }

[0131] 再找到源码中对Applicant的getStatus()成员方法的全部引用,获取使用了枚举或静态类的相关语句,如:

[0132] dto.getStatus()>＝CertificationStatusEnum.PendingReview.getCode()

[0133] dto.getStatus()>＝CertificationStatusEnum.Issued.getCode()

[0134] 由此,可将Applicant中的getStatus()成员方法与CertificationStatusEnum进行关联。从而将源码中与Applicant的getStatus()成员方法有关的全部魔法值,替换为枚举CertificationStatusEnum中等值的成员方法。

[0135] 例如:CertificationStatusEnum.PendingReview.getCode()的结果为1,则可将if(dto.getStatus()＝＝1)自动替换为:

[0136] if(dto.getStatus()＝＝CertificationStatusEnum.PendingReview.getCode()),

[0137] 将dto.setStatus(1)替换为:

[0138] dto.setStatus(CertificationStatusEnum.PendingReview.getCode())。

[0139] 若步骤3中成员方法到对应的枚举值不止一个,如:

[0140] dto.getStatus()>=CertificationStatusEnum.PendingReview.getCode()

[0141] dto.getStatus()>=CertificationConstant.Status.Issued

[0142] 说明代码有异味,其中存在冗余的枚举或静态类,此时提示用户删除多余的冗余枚举或静态类来优化代码,也可通过该映射方式自动完成。如用户选择使用CertificationStatusEnum,则会将全部与CertificationConstant.Status相关的代码做替换,替换为CertificationStatusEnum中对应的成员方法。

[0143] 如CertificationStatusEnum.PendingReview.getCode()与CertificationConstant.Status.Issued值相等,则将代码中的CertificationConstant.Status.Issued全部替换为CertificationStatusEnum.PendingReview.getCode()

[0144] 步骤4:根据魔法值内容找到枚举或静态类中匹配的值进行替换。

[0145] 若步骤3中确定的相关的枚举或静态类中确定与魔法值匹配的枚举值或静态成员,若匹配成功则根据根据匹配的枚举值或静态值替换魔法值。若匹配失败则执行步骤5.

[0146] 步骤5:创建新的枚举值或静态类变量。

[0147] 在步骤4匹配失败后,创建与匹配失败的魔法值等值的枚举值或静态类变量。

[0148] 除此以外,本示例还提出了另一种更为简便的关联方法,在用户定义字段时,在该代码上面一行通过自定义注解进行标注,注解中包含字段对应的枚举或实体类。后续即可直接将涉及到该类成员的魔法值替换为对应的枚举值/静态成员。相关过程可参照图4,图4为本示例提出的另一种智能代码处理方案的原理示意图。

[0149] 例如:定义以下类:

[0150] public class Applicant{

[0151] private Long id;

[0152] private String firstName;

[0153] private String lasttName;

[0154] ....

[0155] @MyCustAnno(value=CertificationConstant.Status.class)

[0156] private Byte status;

[0157] ....

[0158] //Constructors,Getter,Setter

[0159] }

[0160] 注解“@MyCustAnno”标注了字段status可与枚举或静态类关联,而“value=CertificationConstant.Status.class”表示与字段status关联的枚举类或静态类为CertificationConstant.Status

[0161] 对源代码进行处理时可直接将代码中的dto.getStatus()==1和dto.setStatus(1)根据MyCustAnno找到其对应的枚举类CertificationConstant.Status,替换魔法值后转换为dto.getStatus()==CertificationConstant.Status.PendingApproval和dto.setStatus(CertificationConstant.Status.PendingApproval)。如在对应的枚举类或静态类中不含该值,则可自动新增。

[0162] 通常的,开发者在定义数据库字段时,经常将枚举字段定义为tinyint类型或int类型,这种类型的字段,在通过现有技术生成实体类时,并无法自动转为相应的枚举类型或

静态类型，如直接修改生成实体类的字段类型为枚举类型，则需要调整框架代码，成本高昂；此时通过上述方法进行关联，即可解决对应的问题。

[0163] 本示例通过将应用程序代码中涉及使用魔法值的代码的相关类成员，与枚举或静态类相关联，并实现自动替换的方法。本示例的方法可应用于在集成开发环境(IDE)智能提示中，通过该方法进行提示或进行代码替换；或应用于代码库或持续集成工具中，通过该方式进行代码优化。

[0164] 本示例一方面，通过在本地自动检索并替换魔法值，无需再通过代码扫描后，针对每个魔法值分别修改，节省大量时间。也可发现冗余的枚举类或静态类，对代码进行优化。另一方面，通过在远程持续集成工具或配置库自动检索并替换魔法值，可免去开发者修改再重新提交，重新构建的过程，增加持续集成工具编译的成功率，避免无意义的重复编译，同时降低了服务器资源开销以及人工核查处理的工作量。

[0165] 本实施例提出一种代码中目标数值的处理装置，图5为本发明实施例代码中目标数值的处理装置的组成结构示意图，如图5所示，所述装置500包括：

[0166] 查找单元501，用于根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句；

[0167] 第一确定单元502，用于将所述获取单元中的梯度幅值对所述光流场进行加权处理，获得加权光流场；

[0168] 获取单元503，用于获取所述引用语句中包含第一枚举或第一静态类信息的关联语句；

[0169] 替换单元504，用于根据所述关联语句，确定与所述目标数值对应的目标成员；将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员。

[0170] 在一些实施例中，所述第一确定单元，还用于根据所述目标语句中的实例名称，在所述待测代码库中确定与所述实例名称对应的类相关联的代码块；在所述代码块中筛选与所述实例名称和所述字段名称关联的引用语句。

[0171] 在一些实施例中，所述装置还包括：第一生成单元，用于在所述关联语句中没有与所述目标数值对应的目标成员时，在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0172] 在一些实施例中，所述装置还包括：第二确定单元；

[0173] 所述第二确定单元，用于根据所述目标语句的字段信息，在所述待测代码库中的标注信息中确定与所述字段信息关联的标注语句，和确定所述标注语句中与所述目标数值对应的目标成员；其中所述标注语句中包含第二枚举或第二静态类信息。

[0174] 在一些实施例中，所述装置还包括：第二生成单元，用于在所述第二枚举或第二静态类信息中没有与所述目标数值对应的目标成员时，在所述枚举或静态类中新增与所述目标数值对应的目标成员。

[0175] 在一些实施例中，所述装置还包括：提示单元，用于当存在与所述目标数值对应的多个目标成员时输出提示信息；其中，所述提示信息用于指示用户确定第一目标成员，并删除除所述第一目标成员以外的其他目标成员；所述第一目标成员为所述多个目标成员中的任一目标成员。

[0176] 以上装置实施例的描述，与上述方法实施例的描述是类似的，具有同方法实施例

相似的有益效果。对于本发明装置实施例中未披露的技术细节，请参照本发明方法实施例的描述而理解。

[0177] 需要说明的是，本发明实施例中，如果以软件功能模块的形式实现上述的代码中目标数值的处理方法，并作为独立的产品销售或使用时，也可以存储在一个计算机可读取存储介质中。基于这样的理解，本发明实施例的技术实施例本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台控制服务器(可以是个人计算机、服务器、或者网络服务器等)执行本发明各个实施例所述方法的全部或部分。而前述的存储介质包括:U盘、移动硬盘、只读存储器(Read Only Memory,ROM)、磁碟或者光盘等各种可以存储程序代码的介质。这样，本发明实施例不限制于任何特定的硬件和软件结合。

[0178] 对应地，本发明实施例提供一种代码中目标数值的处理设备，包括存储器和处理器，所述存储器存储有可在处理器上运行的计算机程序，所述处理器执行所述程序时实现上述实施例提供的控制方法中的步骤。

[0179] 对应地，本发明实施例提供一种计算机可读存储介质，其上存储有计算机程序，该计算机程序被处理器执行时实现上述实施例提供的控制方法中的步骤。

[0180] 这里需要指出的是:以上存储介质和设备实施例的描述，与上述方法实施例的描述是类似的，具有同方法实施例相似的有益效果。对于本发明存储介质和设备实施例中未披露的技术细节，请参照本发明方法实施例的描述而理解。

[0181] 需要说明的是，图6为本发明实施例中代码中目标数值的处理设备的一种硬件实体结构示意图，如图6所示，该代码中目标数值的处理设备600的硬件实体包括:处理器601和存储器603，可选地，所述代码中目标数值的处理设备600还可以包括通信接口602。

[0182] 可以理解，存储器603可以是易失性存储器或非易失性存储器，也可包括易失性和非易失性存储器两者。其中，非易失性存储器可以是只读存储器(ROM,Read Only Memory)、可编程只读存储器(PROM,Programmable Read-Only Memory)、可擦除可编程只读存储器(EPROM,Erasable Programmable Read-Only Memory)、电可擦除可编程只读存储器(EEPROM,Electrically Erasable Programmable Read-Only Memory)、磁性随机存取存储器(FRAM,ferromagnetic random access memory)、快闪存储器(Flash Memory)、磁表面存储器、光盘、或只读光盘(CD-ROM,Compact Disc Read-Only Memory);磁表面存储器可以是磁盘存储器或磁带存储器。易失性存储器可以是随机存取存储器(RAM,Random Access Memory)，其用作外部高速缓存。通过示例性但不是限制性说明，许多形式的RAM可用，例如静态随机存取存储器(SRAM,Static Random Access Memory)、同步静态随机存取存储器(SSRAM,Synchronous Static Random Access Memory)、动态随机存取存储器(DRAM,Dynamic Random Access Memory)、同步动态随机存取存储器(SDRAM,Synchronous Dynamic Random Access Memory)、双倍数据速率同步动态随机存取存储器(DDRSDRAM,Double Data Rate Synchronous Dynamic Random Access Memory)、增强型同步动态随机存取存储器(ESDRAM,Enhanced Synchronous Dynamic Random Access Memory)、同步连接动态随机存取存储器(SLDRAM,SyncLink Dynamic Random Access Memory)、直接内存总线随机存取存储器(DRRAM,Direct Rambus Random Access Memory)。本发明实施例描述的存储器403旨在包括但不限于这些和任意其它适合类型的存储器。

[0183] 上述本发明实施例揭示的方法可以应用于处理器601中，或者由处理器601实现。处理器601可能是一种集成电路芯片，具有信号的处理能力。在实现过程中，上述方法的各步骤可以通过处理器601中的硬件的集成逻辑电路或者软件形式的指令完成。上述的处理器601可以是通用处理器、数字信号处理器(DSP，Digital Signal Processor)，或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件等。处理器601可以实现或者执行本发明实施例中的公开的各方法、步骤及逻辑框图。通用处理器可以是微处理器或者任何常规的处理器等。结合本发明实施例所公开的方法的步骤，可以直接体现为硬件译码处理器执行完成，或者用译码处理器中的硬件及软件模块组合执行完成。软件模块可以位于存储介质中，该存储介质位于存储器603，处理器601读取存储器603中的信息，结合其硬件完成前述方法的步骤。

[0184] 在示例性实施例中，设备可以被一个或多个应用专用集成电路(ASIC，Application Specific Integrated Circuit)、DSP、可编程逻辑器件(PLD，Programmable Logic Device)、复杂可编程逻辑器件(CPLD，Complex Programmable Logic Device)、现场可编程门阵列(FPGA，Field-Programmable Gate Array)、通用处理器、控制器、微控制器(MCU，Micro Controller Unit)、微处理器(Microprocessor)、或其他电子元件实现，用于执行前述方法。

[0185] 在本发明所提供的几个实施例中，应该理解到，所揭露的方法和装置，可以通过其他的方式实现。以上所描述的装置实施例仅仅是示意性的，例如，所述单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，如：多个单元或组件可以结合，或可以集成到另一个观测量，或一些特征可以忽略，或不执行。另外，所显示或讨论的各组成部分相互之间的通信连接可以是通过一些接口，设备或单元的间接耦合或通信连接，可以是电性的、机械的或其他形式的。

[0186] 上述作为分离部件说明的单元可以是、或也可以不是物理上分开的，作为单元显示的部件可以是、或也可以不是物理单元，即可以位于一个地方，也可以分布到多个网络单元上；可以根据实际的需要选择其中的部分或全部单元来实现本实施例的目的。

[0187] 本领域普通技术人员可以理解：实现上述方法实施例的全部或部分步骤可以通过程序指令相关的硬件来完成，前述的程序可以存储于计算机可读取存储介质中，该程序在执行时，执行包括上述方法实施例的步骤；而前述的存储介质包括：移动存储设备、只读存储器(ROM，Read-Only Memory)、磁碟或者光盘等各种可以存储程序代码的介质。

[0188] 或者，本发明实施例上述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时，也可以存储在一个计算机可读取存储介质中。基于这样的理解，本发明实施例的技术实施例本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，包括若干指令用以使得一台设备(可以是个人计算机、服务器、或者网络设备等)执行本发明各个实施例所述方法的全部或部分。而前述的存储介质包括：移动存储设备、ROM、磁碟或者光盘等各种可以存储程序代码的介质。

[0189] 本发明是实例中记载的代码中目标数值的处理方法、装置、设备和存储介质只以本发明所述实施例为例，但不仅限于此，只要涉及到该代码中目标数值的处理方法、装置、设备和存储介质均在本发明的保护范围。

[0190] 应理解，说明书通篇中提到的“一个实施例”或“一实施例”意味着与实施例有关的特定特征、结构或特性包括在本发明的至少一个实施例中。因此，在整个说明书各处出现的“在一个实施例中”或“在一实施例中”未必一定指相同的实施例。此外，这些特定的特征、结构或特性可以任意适合的方式结合在一个或多个实施例中。应理解，在本发明的各种实施例中，上述各过程的序号的大小并不意味着执行顺序的先后，各过程的执行顺序应以其功能和内在逻辑确定，而不应对本发明实施例的实施过程构成任何限定。上述本发明实施例序号仅仅为了描述，不代表实施例的优劣。

[0191] 需要说明的是，在本文中，术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含，从而使得包括一系列要素的过程、方法、物品或者装置不仅包括那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、物品或者装置所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括该要素的过程、方法、物品或者装置中还存在另外的相同要素。

[0192] 本发明所提供的几个方法实施例中所揭露的方法，在不冲突的情况下可以任意组合，得到新的方法实施例。

[0193] 本发明所提供的几个产品实施例中所揭露的特征，在不冲突的情况下可以任意组合，得到新的产品实施例。

[0194] 本发明所提供的几个方法或设备实施例中所揭露的特征，在不冲突的情况下可以任意组合，得到新的方法实施例或设备实施例。

[0195] 以上所述，仅为本发明的实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，可轻易想到变化或替换，都应涵盖在本发明的保护范围之内。因此，本发明的保护范围应以所述权利要求的保护范围为准。

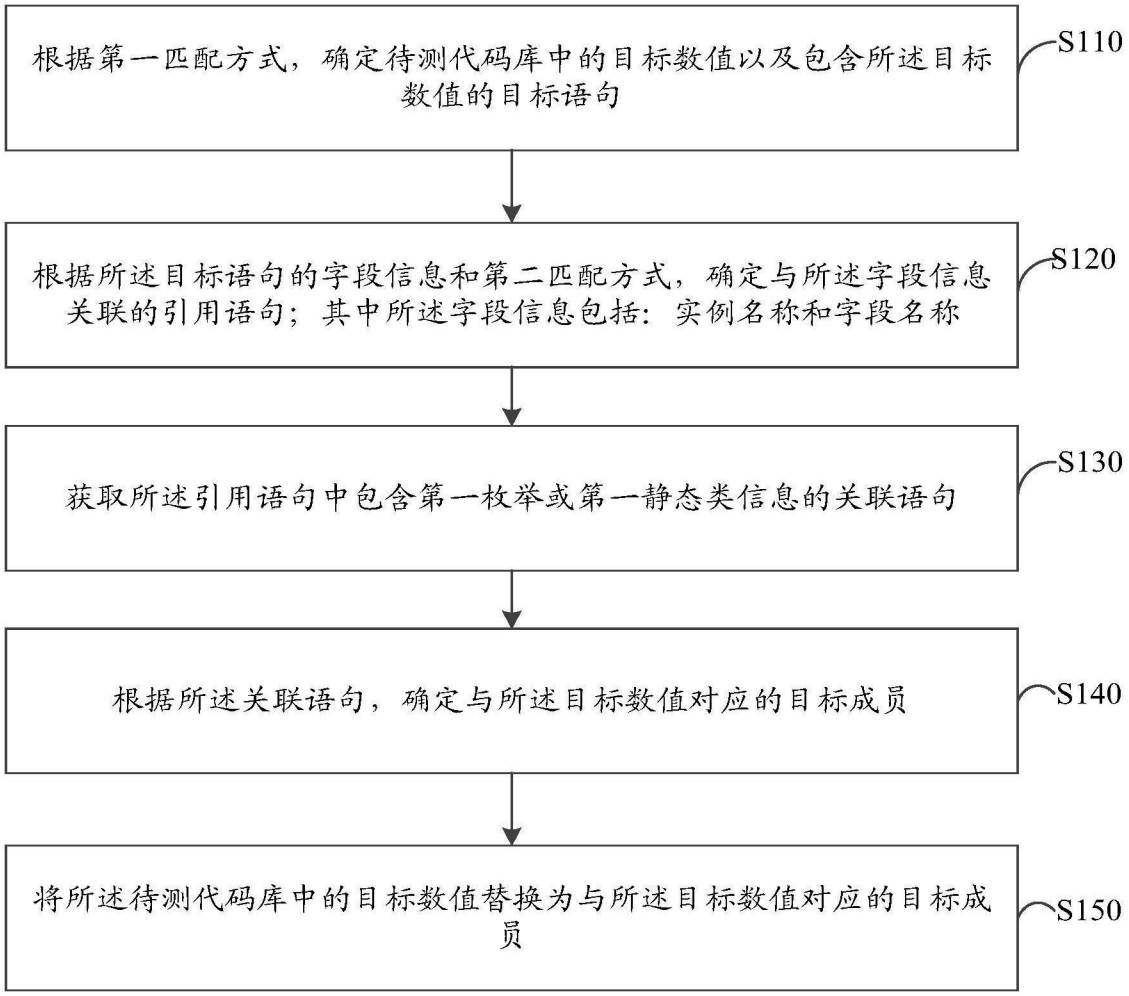

根据第一匹配方式，确定待测代码库中的目标数值以及包含所述目标数值的目标语句
S110
根据所述目标语句的字段信息和第二匹配方式，确定与所述字段信息关联的引用语句；其中所述字段信息包括：实例名称和字段名称
S120
获取所述引用语句中包含第一枚举或第一静态类信息的关联语句
S130
根据所述关联语句，确定与所述目标数值对应的目标成员
S140
将所述待测代码库中的目标数值替换为与所述目标数值对应的目标成员
S150

图1

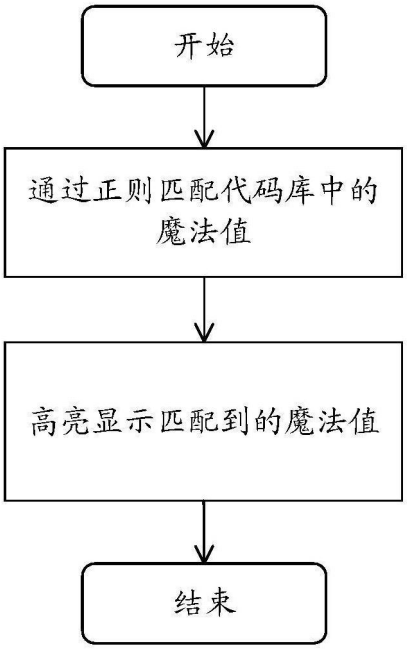


图2

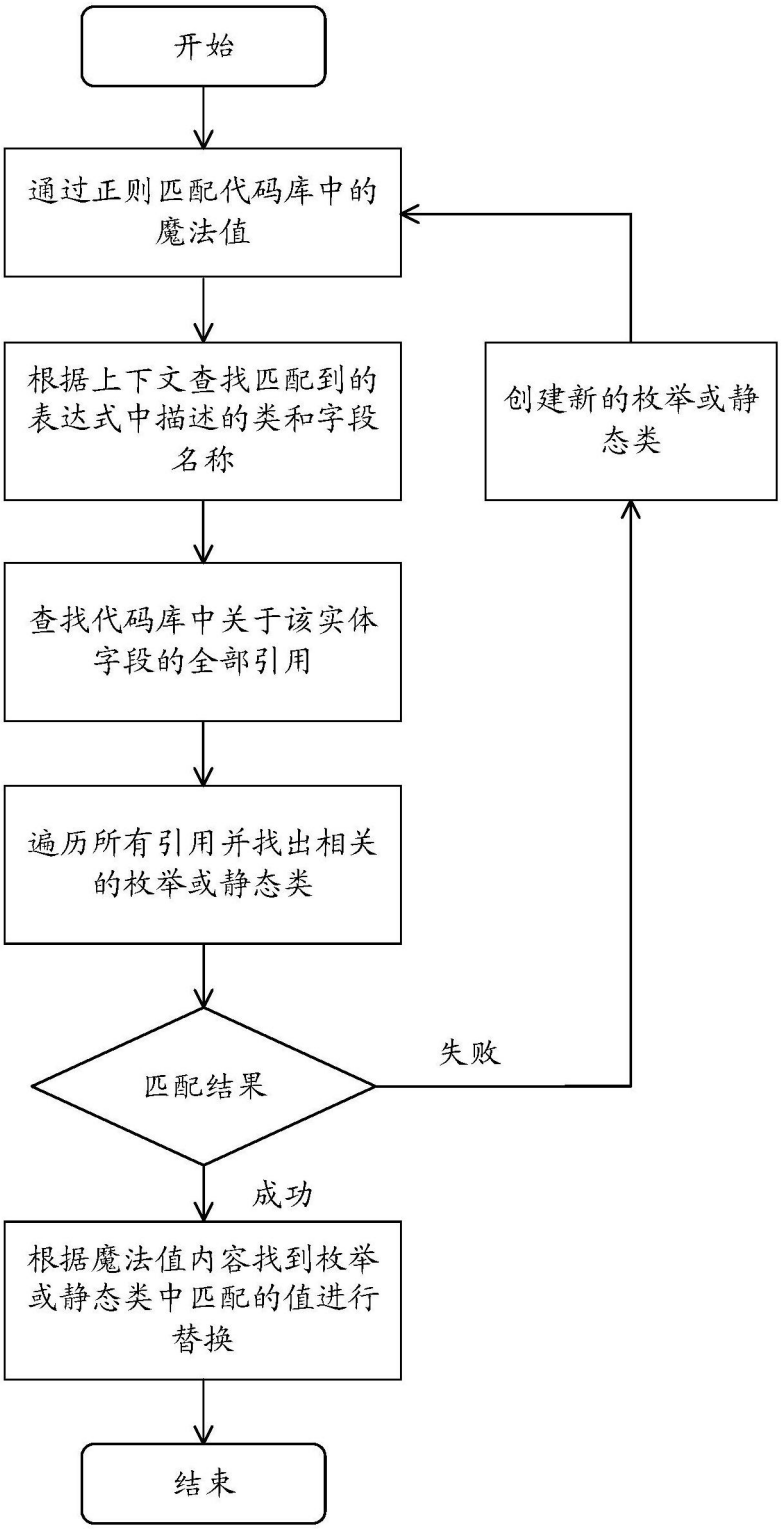

开始
通过正则匹配代码库中的魔法值
根据上下文查找匹配到的表达式中描述的类和字段名称
查找代码库中关于该实体字段的全部引用
遍历所有引用并找出相关的枚举或静态类
匹配结果
失败
创建新的枚举或静态类
成功
根据魔法值内容找到枚举或静态类中匹配的值进行替换
结束

图3

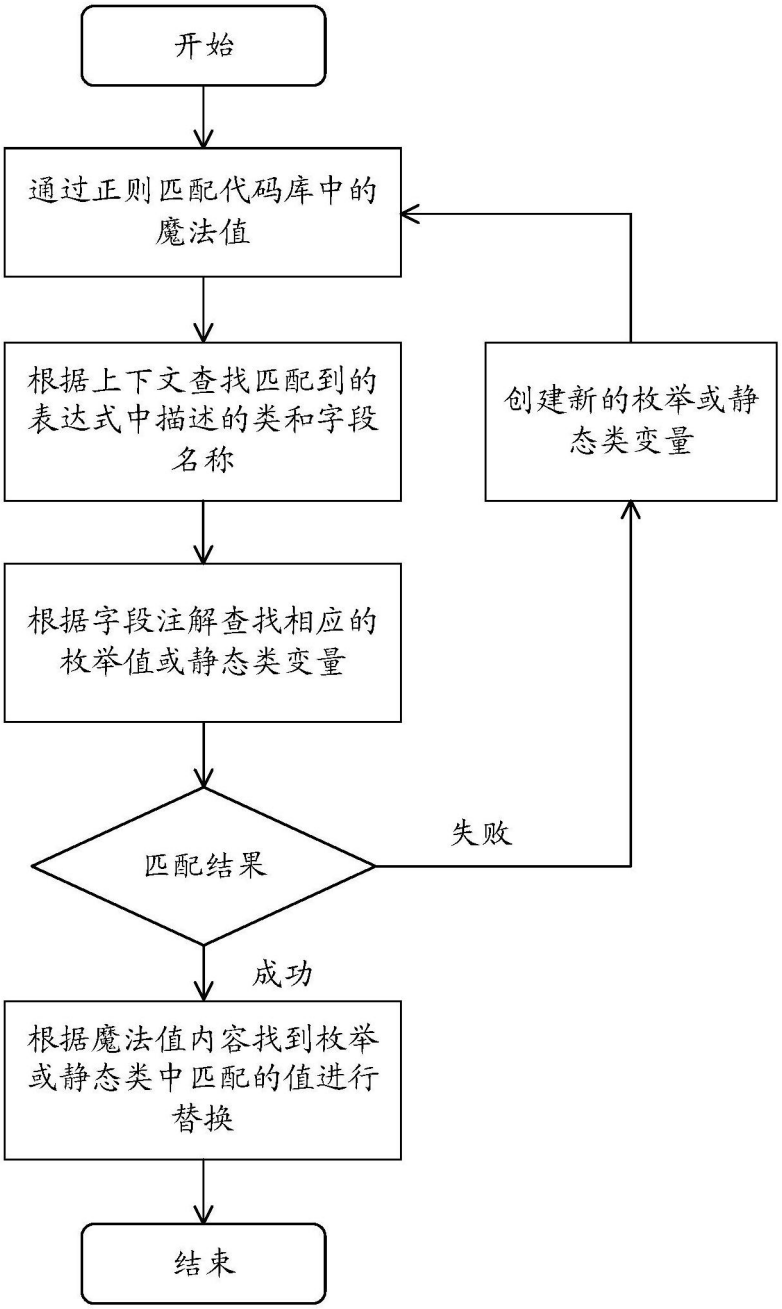

开始
通过正则匹配代码库中的魔法值
根据上下文查找匹配到的表达式中描述的类和字段名称
根据字段注解查找相应的枚举值或静态类变量
匹配结果
失败
创建新的枚举或静态类变量
成功
根据魔法值内容找到枚举或静态类中匹配的值进行替换
结束

图4

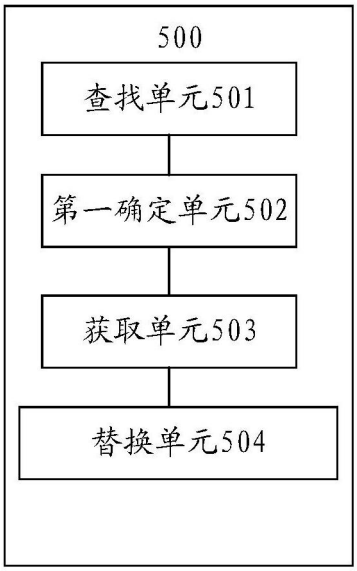

500
查找单元501
第一确定单元502
获取单元503
替换单元504

图5

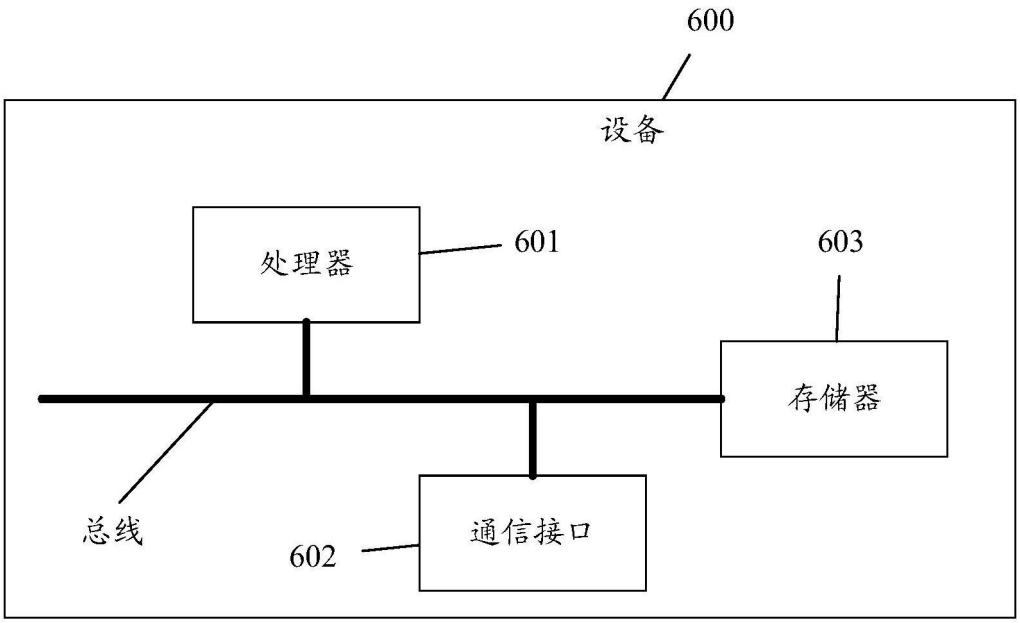

600
设备
处理器
601
603
存储器
总线
602
通信接口

图6