



软件开发生产线 CodeArts

代码检查用户指南

天翼云科技有限公司

目 录

1 使用流程	4
2 角色与权限	7
3 准备工作	8
4 任务创建	10
4.1 创建 Repo 源码源检查任务.....	10
4.2 创建通用 Git 源码源检查任务.....	10
5 规则集设置（可选）	12
5.1 代码检查规则集列表.....	12
5.2 新增自定义检查规则集.....	13
5.3 删除自定义检查规则集.....	14
5.4 修改自定义检查规则集.....	14
5.5 导出规则集.....	15
5.6 查看规则详情.....	15
6 任务设置（可选）	17
6.1 概述.....	17
6.2 设置基本信息.....	17
6.3 设置规则集.....	19
6.4 设置忽略文件.....	19
6.5 设置质量门禁.....	20
6.6 设置任务执行计划.....	21
6.7 开启/关闭消息通知.....	22
6.8 设置检查模式.....	23
6.9 设置分支合并自动检查任务.....	23
6.10 设置代码提交时自动检查任务.....	24
6.11 设置新问题起始时间.....	24
6.12 设置源文件编码.....	25
6.13 设置自定义镜像.....	25
6.14 设置问题责任人精准匹配.....	26

7 任务执行	27
7.1 执行代码检查任务	27
7.2 设置多分支检查	27
8 检查结果查看	29
8.1 查看代码检查详情	29
8.2 查看代码问题	30
8.3 查看代码度量问题	31
8.4 查看检查日志	33

1 使用流程

背景信息

代码检查是基于云端实现的代码检查服务。建立在多年自动化源代码静态检查技术积累与企业级应用经验的沉淀之上，为用户提供代码风格、通用质量与网络安全风险等丰富的检查能力，提供全面质量报告、便捷的问题闭环处理帮助企业有效管控代码质量，助力企业成功。

使用流程

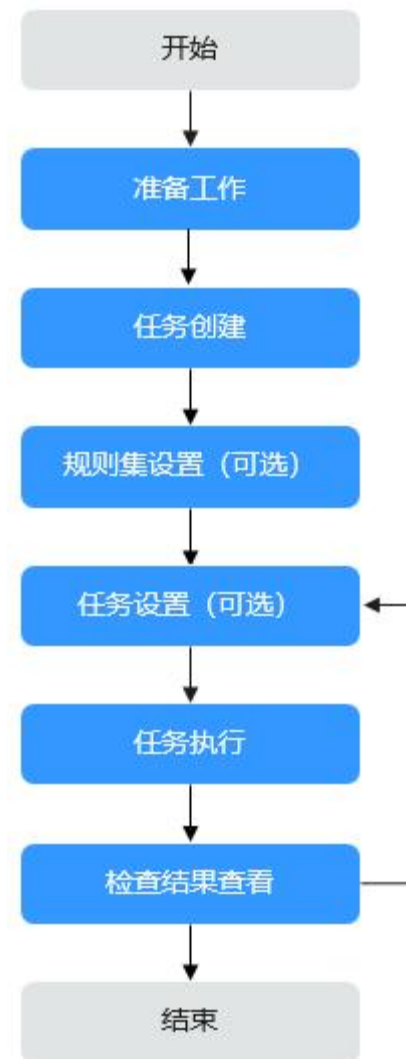


表 1-1 使用流程说明

序号	流程	说明
1	准备工作	在使用代码检查任务前，请完成项目创建等准备工作。
2	任务创建	您可创建不同类型的代码检查任务。
3	规则集设置 (可选)	您可根据实际需求新增、复制、修改、删除或导出自定义规则集。
4	任务设置 (可选)	您可对已有的检查任务进行修改或配置。
5	任务执行	您可对目标代码检查任务执行检查，及时发现源代码中的代码问题。
6	检查结果查看	代码检查任务执行完成后，您可查看检查结果并修改代码

序号	流程	说明
		问题。

2 角色与权限

针对项目下资源（任务、规则集等），介绍代码检查中用户默认的角色权限说明，如表 2-1 所示。

表 2-1 代码检查默认角色权限说明

资源权限/角色	项目创建者	项目经理	开发人员	测试经理	测试人员	项目参与者	浏览者	运维经理	
代码检查任务	创建	√	√	√	×	×	×	×	×
	执行	√	√	√	×	×	×	×	×
	查看	√	√	√	×	×	×	×	×
	修改	√	√	√	×	×	×	×	×
	删除	√	√	×	×	×	×	×	×
规则集	设置默认规则集	√	√	×	×	×	×	×	×
问题	查看	√	√	√	×	×	×	×	×
	修改	√	√	√	×	×	×	×	×
	导出	√	√	√	×	×	×	×	×

3 准备工作

在创建代码检查任务前，请先做好如下准备工作：

- [进入代码检查页面](#)
- [创建项目](#)
- [创建代码仓库](#)（针对 Repo 源码源）
- [创建服务扩展点](#)（针对通用 Git 源码源）

进入代码检查页面

进入代码检查服务的两种方式为：项目入口和首页入口。

- **项目入口**
 - a. 在软件开发生产线首页，单击目标项目名称，进入项目详情页面。
 - b. 单击项目内“代码 > 代码检查”，进入代码检查页面。

说明

项目内菜单位置跟“外观设置”的主题、布局有关，此处以默认外观“无限+经典”为例。如需修改外观，请单击右上角**用户名**，单击“外观设置”进行修改。

- **首页入口**

在软件开发生产线首页，单击“服务 > 代码检查”，进入代码检查页面。

创建项目

首次使用代码检查服务时，请在需求管理中完成项目的创建。

若您已创建项目，请跳过此步骤。

创建代码仓库

请在代码托管中完成代码仓库的创建。

若您已创建代码仓库，请跳过此步骤。

创建服务扩展点

创建代码检查任务时，若选择通用 Git 源码源，请先创建服务扩展点。

若您已创建服务扩展点，请跳过此步骤。

4 任务创建

4.1 创建 Repo 源码源检查任务

前提条件

您已完成准备工作，具体操作请参见[准备工作](#)。

操作步骤

此处以[项目入口](#)方式为例进入代码检查页面。

- 步骤 1 在代码检查页面，单击“新建任务”。
- 步骤 2 进入“新建任务”页面，勾选 Repo 源码源。
- 步骤 3 选择您已创建的代码仓，单击“操作”列中“新建任务”。

您也可以单击“新建代码仓”，完成代码仓库的创建。

- 步骤 4 进入“代码检查详情”页面，单击“开始检查”，即可执行代码检查任务。执行完代码检查任务后，还可以根据需要查看检查结果。

---结束

4.2 创建通用 Git 源码源检查任务

前提条件

您已完成准备工作，具体操作请参见[准备工作](#)。

操作步骤

此处以[项目入口](#)方式为例进入代码检查页面。

- 步骤 1 在代码检查页面，单击“新建任务”。
- 步骤 2 进入“新建任务”页面，源码源选择“通用 Git”。

步骤 3 根据实际情况设置 Endpoint 实例、Repository、Branch，并选择检查语言。

您可在“Endpoint 实例”区域单击“管理”，进入“服务扩展点管理”页面，新建通用 Git 服务扩展点，详见[创建服务扩展点](#)。

步骤 4 单击“确定”，进入“代码检查详情”页面，单击“开始检查”，即可执行代码检查任务。执行完代码检查任务后，还可以根据需要查看检查结果。

---结束

5 规则集设置（可选）

5.1 代码检查规则集列表

代码检查默认支持 JAVA、C++、PHP、JAVASCRIPT、TYPESCRIPT、HTML、CSS、GO 和 PYTHON 语言的规则集，且每种语言类型对应多种不同级别的规则集，系统规则集列表请参见表 5-1。

表 5-1 系统规则集类型

语言	系统规则集类型	支持扫描文件后缀
C++	• Generic Inspection Rule Set	.c/.cc/.cpp/.cxx/.cp/.c++/.inc/.inl /.mm/.h/.hh/.hpp/.hxx/.h++
Java	• Generic Inspection Rule Set	.java
Go	• Generic Inspection Rule Set	.go
Python	• Generic Inspection Rule Set	.py
JavaScript	• Generic Inspection Rule Set	.js/.jsx/.vue
TypeScript	• Generic Inspection Rule Set	.ts/.tsx/.vue
CSS	• Generic Inspection Rule Set	.css
HTML	• Generic Inspection Rule Set	.html/.htm/.html.hl/.xht
PHP	• Generic Inspection Rule Set	.php/.php3/.php4/.php5/.phps/.a w/.ctp/.phakefile/.php_cs/.php_c s.dist/.phpt/.phtml





规则集入口

步骤 1 在软件开发生产线首页，单击“服务 > 代码检查”，进入到代码检查页面。

步骤 2 在代码检查页面，单击“规则集”页签，进入检查规则集列表页面。

根据您的需求新增、修改、复制、删除、查看、设置或导出规则集，具体说明请参见表 5-2。

表 5-2 规则集操作说明

操作项	说明
新增规则集	在“规则集”页签，单击“新建”即可自定义规则集，请参见 新增自定义检查规则集 。
配置规则集	单击规则集所在行的“操作”列中的  ，进入“规则集配置”页面，可以根据实际需要勾选规则及设置问题级别。
修改规则集基本信息	进入“规则集配置”页面，单击“修改基本信息”，可以根据实际需要修改规则集名称和描述，请参见 修改规则集基本信息 。
复制规则集	单击规则集所在行“操作”列中的  ，在弹框中设置规则集名称即可复制该规则集。 说明 规则集名称不能重复。
默认规则集（项目下）	在项目详情页面选择“代码 > 代码检查 > 规则集”，进入规则集列表页面，单击规则集所在行的  ，选择“默认规则集”即可设置该规则集为该项目下代码检查任务的默认检查规则集。
删除规则集	单击规则集所在行的  ，选择“删除”即可删除该自定义规则集，请参见 删除自定义检查规则集 。 说明 系统规则集、非本人创建和已使用的规则集无法删除。
查看规则集	单击规则集名称可以查看规则集详情。
设置规则集	进入某个检查详情页面，选择“设置 > 规则集”页签，选中已新建好的规则集设置为任务的检查规则集。请参见 设置规则集 。
导出规则集	进入某个规则集页面，单击“导出”可导出该规则集中所有规则，请参见 导出规则集 。

---结束

5.2 新增自定义检查规则集

- 代码检查除了系统规则集类型外，您还可以根据实际需要自定义代码检查规则集。
- 每个规则集模板最少设置一条规则。

前提条件

目前仅支持配置单语言的规则集，即一种规则集只能配置同一种语言类型的检查规则。

操作步骤


- 步骤 1 进入[代码检查页面](#)，单击“规则集”页签，默认进入规则集列表页面。
- 步骤 2 单击“新建”，在弹出的“新建检查规则集”窗口设置规则集名称、检查语言。
(可选) 您也可以单击下拉框选择复制于已有的规则集。
- 步骤 3 完成后，单击“确定”。
- 步骤 4 根据需要勾选规则名称、设置问题级别，单击右上角“保存”。
- 步骤 5 进入“代码检查详情”页面选择“设置 > 规则集”，单击“已包含语言”所在行重新获取代码仓语言，打开目标语言的开关，并选中您已配置好的自定义规则集。具体请参见[设置规则集](#)。
- 步骤 6 单击“开始检查”，即按照自定义规则集执行检查。

---结束

5.3 删除自定义检查规则集

当自定义检查规则集已关联在代码检查任务时，您无法删除该规则集。只有将关联使用该规则集的代码检查任务删除或将代码检查任务关联其它规则集后，方可删除该自定义规则集。

操作步骤

- 步骤 1 进入[代码检查页面](#)，单击“规则集”页签，进入规则集列表页面。
- 步骤 2 如果检查规则集已被使用，请先将使用该规则集的代码检查任务删除或将代码检查任务改用其它规则集。
- 步骤 3 单击规则集所在行的，选择“删除”可以删除检查规则集。

说明

- 只有创建者可以删除规则集。
- 系统规则集和已被使用的规则集不能删除。

---结束

5.4 修改自定义检查规则集

您可以根据需要修改自定义检查规则集基本信息，包括规则集名称和描述。

前提条件

- 系统规则集不能修改。
- 只能修改自定义的规则集。

操作步骤

- 步骤 1 进入[代码检查页面](#)，单击“规则集”页签，进入规则集列表页面。
- 步骤 2 您可在搜索框中选择语言，筛选规则集。
- 步骤 3 单击规则集名称，进入规则集详情页面。
- 步骤 4 单击“修改基本信息”。
- 步骤 5 修改规则集名称或描述信息后，单击“确定”。

说明

“检查语言”不支持修改。

---结束

5.5 导出规则集

您可以将规则集下载到本地，查看该规则集内所有规则的详细内容。

操作步骤

- 步骤 1 进入[代码检查页面](#)，单击“规则集”页签，进入规则集列表页面。
- 步骤 2 您可在搜索框中选择语言，筛选规则集。
- 步骤 3 单击规则集名称，进入规则集详情页面。
- 步骤 4 单击“导出”，即可导出该规则集中已启用的规则。

---结束

5.6 查看规则详情

规则详情包含：问题级别、正确示例、错误示例和修复建议等。

操作步骤：方式一

- 步骤 1 进入[代码检查页面](#)，单击“规则”页签。
- 步骤 2 进入检查规则页面，单击不同语言类型，显示对应的规则详情。

---结束

操作步骤：方式二

- 步骤 1 进入[代码检查页面](#)，单击“规则集”页签。
- 步骤 2 进入规则集列表页面，单击规则集名称或检查规则数量。

步骤 3 进入规则列表页面，单击规则名称，显示对应的规则详情。

---结束

6 任务设置（可选）

6.1 概述

您可对已有的检查任务进行修改或任务设置，任务设置详细操作如表 6-1 所示。

表 6-1 代码检查任务设置操作说明

任务设置项	操作说明
基本信息	可以修改任务名称、修改默认分支、删除任务，具体请参见 设置基本信息 。
规则集	可根据实际需要重新启用代码检查规则集，具体请参见 设置规则集 。
忽略文件	请参见 设置忽略文件 。
质量门禁	请参见 设置质量门禁 。
执行计划	请参见 设置任务执行计划 。
通知管理	请参见 开启/关闭消息通知 。
检查模式	请参见 设置检查模式 。
集成与服务	请参见 设置分支合并自动检查任务 、 设置代码提交时自动检查任务 。
高级选项	请参见 设置新问题起始时间 、 设置源文件编码 、 设置自定义镜像 、 设置问题责任人精准匹配 。

6.2 设置基本信息

您可以根据需要设置任务的基本信息，如检查任务名称、默认分支，您也可以删除检查任务。

📖 说明

- 任务创建者、项目创建者或项目管理员可以修改、删除代码检查任务。详见[角色与权限](#)。
- 开发人员可以修改、删除自己的创建的任务，不可以删除他人的任务。

修改检查任务名称和默认分支

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 基本信息”，修改任务名称和默认分支。


The screenshot shows a configuration page for a code check task. It has three main sections: 'Task Name', 'Default Branch', and 'Delete Task'.
1. 'Task Name': A text input field containing a blurred name, with a blue '重命名' (Rename) button to its right.
2. 'Default Branch': A dropdown menu currently showing 'master', with a note '定时执行检查的分支基于该默认分支。' (The branch for scheduled checks is based on this default branch.) and a grey '保存' (Save) button to its right.
3. 'Delete Task': A section with a warning '此操作会删除当前任务下所有数据，且不可恢复!' (This operation will delete all data under the current task and is irreversible!) and a grey '删除任务' (Delete Task) button to its right.

---结束

删除检查任务

- 在代码检查页面删除

步骤 1 进入[代码检查页面](#)，单击代码检查任务所在行“操作”列中.

步骤 2 在下拉框中，选择.

步骤 3 在“删除任务”弹框中，输入该任务名称。

步骤 4 单击“确定”即可删除任务。

---结束

- 在任务设置页面删除

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 基本信息”。

步骤 3 单击“删除任务”，并在弹框中输入当前任务名称。

步骤 4 完成后，单击“确定”即可删除任务。

---结束

6.3 设置规则集

您可根据实际需要更换代码检查规则集，新启用的规则集将在下一次检查时生效。

说明


操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 规则集”。

步骤 3 在“已包含语言”区域，打开目标语言开关，并在“启用规则集”区域单击需更换的规则集。

说明

首次进入“规则集”页面时，需要先单击“已包含语言”所在行  重新获取规则集。

步骤 4 在弹框中，单击“确定”即可更换语言规则集。

步骤 5（可选）当选中的规则集需配置扩展参数时，单击“检查参数”，配置代码检查扩展参数并开启配置开关，完成后单击“确认”。

---结束

6.4 设置忽略文件

支持用户以任务为单元，设置代码检查的文件范围：

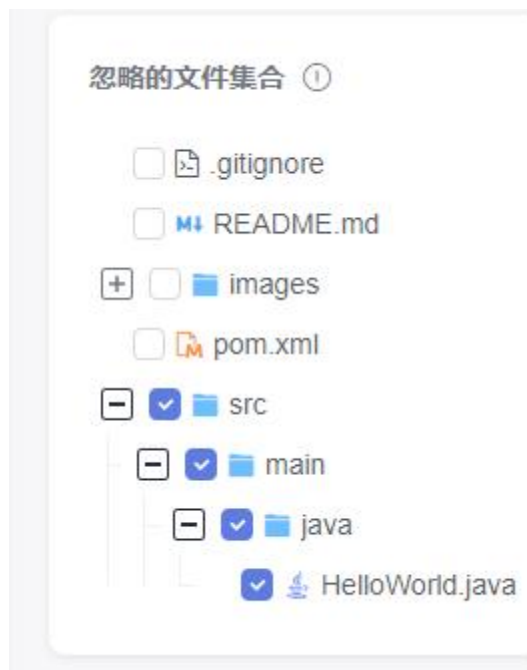
- 不勾选文件路径时，则检查所选仓库分支的所有代码。
- 勾选指定文件路径，则该指定路径下的文件将排除检查。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 忽略文件”。

步骤 3 在“忽略的文件集合”区域，勾选无需进行检查的文件路径。



步骤 4 设置完成后，执行代码检查任务，将按照设置排除选中的文件路径执行检查。

---结束

6.5 设置质量门禁

根据质量要求设置门禁阈值，即可了解您的项目是否已准备好投入生产。

门禁质量设置支持租户级、项目级以及任务级，优先级为租户级 > 项目级 > 任务级。入口分别如下：

- 租户级：进入代码检查服务首页，选择“配置中心”，默认显示质量门禁详情。



- 项目级：进入项目详情页面，选择“代码 > 代码检查 > 配置中心 > 质量门禁”。



- 任务级：进入代码检查详情页面，选择“设置 > 质量门禁”。本节以任务级门禁设置为例介绍操作步骤。

说明

项目级和任务级支持事件[通知设置](#)。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 质量门禁”。

步骤 3 在“质量门禁”区域，根据质量要求开启门禁开关，并设置门禁阈值。



步骤 4 完成后，单击“保存”。

您可单击“开始检查”执行检查任务并查看检查结果。

若问题级别数超过设定阈值，任务概览页面提示门禁结果为“不通过”。

---结束

6.6 设置任务执行计划

根据实际需要，设置代码检查任务的执行计划，定时执行默认分支的检查任务。

📖 说明

- 建议在非流水线使用情况下进行配置。
- 配置当天时间间隔必须在 5 分钟以上。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 执行计划”。

步骤 3 开启“定时执行”开关，并设置自动执行时间。

支持以周为周期，配置一周内某天、某时间段开始执行代码检查任务。

步骤 4 完成后，单击“保存”。

---结束

6.7 开启/关闭消息通知

您可以根据实际需求，设置代码检查任务事件类型的通知方式。

门禁失败通知设置支持项目级和任务级，入口分别如下：

- 项目级：进入项目详情页面，选择“代码 > 代码检查 > 配置中心 > 通知管理”。



- 任务级：进入代码检查详情页面，选择“设置 > 通知管理”。本节以任务级通知设置为例介绍操作步骤。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 通知管理”。

步骤 3 根据需要开启或关闭事件类型的通知状态。

- 开启服务动态通知，即通知当前任务所属项目的特定成员，可在消息中心查看消息。
- 开启邮件通知，即通知当前任务所属项目的特定成员，可在邮件中查看消息。

---结束

6.8 设置检查模式

您可以根据实际需求设置检查模式，提高检查效率。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 检查模式”。

步骤 3 在“MR 检查模式”区域，选择“扫描变更文件”或“全量扫描”。默认选择“扫描变更文件”。



步骤 4 设置检查目录。

支持检查代码仓指定目录。

- 不配置检查目录时，检查代码仓中所有目录下的代码文件。
- 配置检查目录时，只检查配置目录中的文件。



📖 说明

支持配置多个检查目录，多个目录需要用英文逗号分隔。例如：

- 如果要检查 dir1,dir2，检查目录配置为 dir1,dir2。
- 如果要检查 dir1 目录下的 dir3，检查目录配置为 dir1/dir3。


步骤 5 完成后，单击“保存”。

---结束


6.9 设置分支合并自动检查任务

当代码托管中有分支合并时，在代码检查服务中可以设置自动检查代码检查任务，并生成一个子任务，即代码仓库中有分支合并，就会重新生成一个代码检查子任务。

操作步骤

- 步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。
- 步骤 2 进入“代码检查详情”页面，选择“设置 > 集成与服务 > CodeArts Repo”。
- 步骤 3 选择是否勾选“MR 检查状态”，勾选即表示开启代码评审（MR）的自动化检查。
- 步骤 4 当所在任务的代码仓库中有分支合并时，代码检查服务会自动生成一个代码检查子任务。
 1. 进入代码检查任务详情页面。
 2. 单击，在代码托管中新建 MR 请求。

说明

3. 单击 master ▼，通过下拉框中，可查看所有的 MR 触发代码检查的历史记录。

---结束

6.10 设置代码提交时自动检查任务

当代码托管中有代码提交成功时，在代码检查服务中自动检查源分支和目标分支。

操作步骤

- 步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。
- 步骤 2 进入“代码检查详情”页面，选择“设置 > 集成与服务 > 持续集成”。
- 步骤 3 在“持续集成”区域，勾选“代码提交时执行”。

说明

仅对已创建任务的分支有效。

- 步骤 4 单击“保存”。

---结束

6.11 设置新问题起始时间

在每个任务的基础上可以设置新问题起始时间。起始时间设置之后要重新扫描，问题的发现时间大于设置时间将被划分为新问题。如果没有设置起始时间，连续两次的检查结果对比，新增的就是新问题数。

操作步骤

- 步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 高级选项”。

步骤 3 在“新问题起始时间”区域，单击  设置新问题起始时间。

说明

默认起始时间为当前任务第一次扫描成功的时间，起始时间之后扫描出的所有问题都将被划分为新问题。

---结束

6.12 设置源文件编码

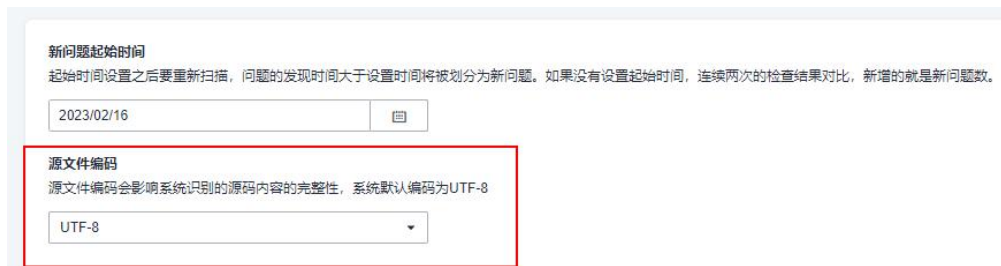
设置文件的编码方式，源文件编码会影响系统识别的源码内容的完整性，系统默认编码为 UTF-8。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 高级选项”。

步骤 3 在“源文件编码”区域，单击下拉框选择源文件编码。



步骤 4 完成后，提示修改成功。

---结束

6.13 设置自定义镜像

当使用代码检查服务时，若常用的编译构建环境缺少您需要的依赖包、工具，无法满足您的使用需求，您可通过自定义构建环境提供的基础镜像执行代码检查任务。

前提条件

- 您已构建自定义环境。
- 构建完成后，将自定义环境推送到镜像仓中。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 高级选项”。

步骤 3 在“自定义镜像”中，输入以“域名/组织名称/仓库名称:tag 名称”格式的内容。



步骤 4 完成后，单击“保存”。

---结束

6.14 设置问题责任人精准匹配

您可根据实际需求，选择是否开启问题责任人精准匹配，将问题分配给不同的成员处理。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“设置 > 高级选项”。

步骤 3 在“问题责任人精准匹配”区域，开启开关更新问题责任人匹配策略。



步骤 4 任务检查完成后，您可在“代码问题”页签下通过选择“负责人”过滤问题。

---结束

7 任务执行

7.1 执行代码检查任务

用户可以对目标代码检查任务执行检查，及时发现源代码中的代码问题。

执行代码检查有两种方式：

- 手动执行：具体请参见[操作步骤](#)。
- 定时执行：设置代码检查任务的[执行计划](#)。


前提条件

您已[创建代码检查任务](#)。

操作步骤

步骤 1 进入[代码检查页面](#)，单击代码检查任务所在行“最近一次检查”中的“立即执行”。

说明

如果代码检查任务已被执行，则单击任务所在行  重新执行任务。

您也可以单击任务名称，进入“代码检查详情”页面，单击“开始检查”。

步骤 2 根据页面提示等待片刻，执行完成后，默认进入“概览”页面。详情请参见[查看代码检查详情](#)。

---结束

7.2 设置多分支检查


在代码检查详情页面可以手动切换“已检查分支列表”和“未检查分支列表”进行检查。

前提条件

仅支持 Repo 检查任务，具体请参见[创建 Repo 源码源检查任务](#)。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 在“代码检查详情”页面，默认显示“概览”页签，单击  master ▼ 可以切换不同分支。

步骤 3 切换目标分支后，单击“开始检查”进行检查。

---结束

8

检查结果查看

8.1 查看代码检查详情

代码检查详情包括：概览、代码问题、代码度量、检查日志和设置。

前提条件

您已[创建并执行代码检查任务](#)。

查看代码检查详情

步骤 1 进入[代码检查页面](#)，搜索目标任务。

说明

在代码检查任务列表中可以查看检查任务的状态。

- 新问题：表示代码检查任务执行后需要处理的新问题。
- 未解决：表示代码检查任务执行后未解决的问题。
- 已解决：表示代码检查任务执行后已经处理完成的问题。

步骤 2 单击检查任务名称链接，查看代码检查详情。

- 概览：请参见[查看任务概览](#)。
- 代码问题：可对每个问题进行查看并修改代码问题，请参见[查看代码问题](#)。
- 代码度量：支持查看代码圈复杂度和代码重复率文件的度量值，请参见[修改代码平均圈复杂度问题](#)和[修改代码重复率问题](#)。
- 检查日志：支持查看检查日志，请参见[查看检查日志](#)。
- 设置：请参见任务设置[概述](#)。

---结束

查看任务概览

在执行完代码检查任务后，展示检查结果的数据图表。

在代码检查详情页面，默认显示“概览”页签，可以查看任务概览信息。

8.2 查看代码问题

执行代码检查任务后，您可在问题列表中对代码问题进行修改问题状态、转移责任人等操作。

操作步骤


- 步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。
- 步骤 2 进入“代码检查详情”页面，选择“代码问题”页签。详情请参见表 8-1。
- 步骤 3 您可以在左侧“过滤器”区域，根据过滤条件筛选问题，并在右侧区域勾选问题进行全量或单独操作。
 - 单击“负责人”，可将此问题提交至其他负责人。
 - 若问题无需修改，单击“问题状态”忽略问题。
 - 单击“查看上下文”，在问题详情页面中，可“修改问题状态”、“转移责任人”，也可根据“问题帮助”单击对代码进行修改。
 - 单击“创建问题单”，提交问题单。
 - 单击“查看问题单”，查看问题单详情。

表 8-1 代码问题操作说明

操作项	说明
问题级别	致命、严重、一般、提示。
问题状态	待处理、已解决、已忽略、已修复。
检测时间	新问题和存量问题。 说明 在“设置 > 高级选项”中可以设置新问题起始时间，问题发现时间在起始时间之后的则为新问题。
语言	按照编程语言进行分类统计。
规则	按照问题命中的规则进行分类统计，便于用户一次性解决同一类问题。
标签	按照设置的工具名称分类进行统计。
负责人	按照分配问题负责人进行统计。 说明 代码检查过程中，自动将新问题分配给问题代码行的最后一次提交者。
CWE	按照安全类规则名进行分类统计。
文件名	按照源文件进行分类统计。
生成报告	将检查报告以附件形式下载到本地。
全量导出	单击“全量导出”，可以全量导出 Excel 形式的代码问题。

操作项	说明
全量修改问题状态	可变更全量代码问题状态为已解决或忽略问题。 说明 如果代码问题数是 0，则不显示该内容。
全量转移负责人	可申请将全量代码问题变更给其他责任人。
批量处理指定问题	勾选目标问题，单击“批量转移负责人”、“批量修改问题状态”或“导出”进行处理。

---结束

8.3 查看代码度量问题

代码度量可准确的反映提交代码的质量，助力用户及时发现并修改问题，提升研发效能。

- **代码圈复杂度**：衡量代码复杂度的标准。可通过圈复杂度大小判断逻辑复杂程度，并对代码进行修改，便于后期维护。
- **代码重复率**：评估代码质量风险。修改冗余代码，保证代码质量。

修改代码圈复杂度问题

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“代码度量”页签，可查看已检查任务的圈复杂度和重复率。

步骤 3 单击文件名称进入文件详情页面。

步骤 4 单击  可以查看文件源码并进行修改。

```
worker/src/main/java/worker/Worker.java
1 package worker;
2
3 import redis.clients.jedis.Jedis;
4 import redis.clients.jedis.exceptions.JedisConnectionException;
5 import java.sql.*;
6 import org.json.JSONObject;
7
8 class Worker {
9     public static void main(String[] args) {
10        try {
11            Jedis redis = connectToRedis("redis");
12            Connection dbConn = connectToDB("db");
13
14            System.err.println("Watching vote queue");
15
16            while (true) {
17                String voteJSON = redis.blpop(0, "votes").get(1);
18                JSONObject voteData = new JSONObject(voteJSON);
19                String voterID = voteData.getString("voter_id");
20                String vote = voteData.getString("vote");
21
22                System.err.printf("Processing vote for '%s' by '%s'\n", vote, voterID);
23                updateVote(dbConn, voterID, vote);
24            }
25        }
26    }
27 }
```

---结束

修改代码重复率问题

- 步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。
- 步骤 2 进入“代码检查详情”页面，选择“代码度量”页签，可查看已检查任务的“重复率”。
- 步骤 3 单击文件名称进入文件详情页面。

文件详情页面左侧的竖线标识了重复的代码块，单击标识竖线，将会显示与该段代码重复的文件列表。


```

319 // hyear is just approximate, it may need adjustment up or down by 1.
320 if (daysSinceJan0101 < absDays) {
321     hyear--;
322     absDays -= daysInYear;
323 }
324 else if (daysSinceJan0101 === absDays) {
325     hyear--;
326     absDays = this.daysToYear(hyear);
327 }
328 else {
329     if (daysSinceJan0101 > (absDays + daysInYear)) {
330
331         Duplicated By
332         result/views/app-assets/js/core/libraries/jquery_ui/globalize/cultures/globalize.culture.a...
333         Lines: 309 - 382
334         result/views/app-assets/js/core/libraries/jquery_ui/globalize/cultures/globalize.culture.a...
335         Lines: 309 - 382
336         result/views/app-assets/js/core/libraries/jquery_ui/globalize/cultures/globalize.culture.a...
337         Lines: 92 - 165
338         result/views/app-assets/js/core/libraries/jquery_ui/globalize/cultures/globalize.culture.a...
339         hmonth++;
340     }
341     hmonth--;
342     hday = daysIntoYear - this.monthDays[hmonth];
343     return [hyear, hmonth, hday];
344 }.
345 daysToYear: function(year) {
346     // calculates how many days since Jan 1, 0001
347     var yearsToYear30 = Math.floor((year - 1) / 30) * 30,
348         yearsInto30 = year - yearsToYear30 - 1,
349         days = Math.floor((yearsToYear30 * 10631) / 30) + 227013;

```

单击文件名称，可对比查看两文件的重复代码块。

源文件	对比文件
<pre> 284 1."06MM/yyyy hh:mm tt" 285 F."06MM/yyyy hh:mm:ss tt" 286 M."06 MM/yyyy" 287 }, 288 convert: { 289 // Adapted to Script from System.Globalization.HijriCalendar 290 ticks1970: 62135596800000, 291 // number of days leading up to each month 292 monthDays: [0, 30, 59, 89, 118, 148, 177, 207, 236, 266, 295, 325, 355], 293 minDate: -42921673600000, 294 maxDate: 21432120019999, 295 // The number of days to add or subtract from the calendar to accommodate the variances 296 // in the start and the end of Ramadan and to accommodate the date difference between 297 // countries/regions. May be dynamically adjusted based on user preference, but should 298 // remain in the range of -2 to 2, inclusive. 299 hijriAdjustment: 0, 300 toGregorian: function(year, hmonth, hday) { 301 var daysSinceJan0101 = this.daysToYear(year) + this.monthDays[hmonth] + hday - 1 - this.hijriAdjustment; 302 // 98400000 = ticks per day 303 var ticks = new Date(daysSinceJan0101 * 98400000 - this.ticks1970); 304 // adjust for timezone, because we are interested in the gregorian date for the same timezone 305 // but ticks in javascript is always from GMT, unlike the server were ticks counts from the base 306 // date in the current timezone 307 gdate.setMinutes(gdate.getMinutes() + gdate.getTimezoneOffset()); 308 return gdate; 309 }, 310 fromGregorian: function(gdate) { 311 if (gdate < this.minDate gdate > this.maxDate) return null; 312 var ticks = this.ticks1970 + (gdate - gdate.getTimezoneOffset()) * 98400000; 313 daysSinceJan0101 = Math.floor(ticks / 98400000) + 1 - this.hijriAdjustment; 314 // very particular formula determined by someone smart, adopted from the server-side implementation. 315 // it approximates the hijri year 316 var hday, hmonth, hyear = Math.floor((daysSinceJan0101 - 227013) / 30) + 10631 + 1, 317 absDays = this.daysToYear(hyear); </pre>	<pre> 307 1."06MM/yyyy hh:mm tt" 308 F."06MM/yyyy hh:mm:ss tt" 309 M."06 MM/yyyy" 310 }, 311 convert: { 312 // Adapted to Script from System.Globalization.HijriCalendar 313 ticks1970: 62135596800000, 314 // number of days leading up to each month 315 monthDays: [0, 30, 59, 89, 118, 148, 177, 207, 236, 266, 295, 325, 355], 316 minDate: -42921673600000, 317 maxDate: 21432120019999, 318 // The number of days to add or subtract from the calendar to accommodate the variances 319 // in the start and the end of Ramadan and to accommodate the date difference between 320 // countries/regions. May be dynamically adjusted based on user preference, but should 321 // remain in the range of -2 to 2, inclusive. 322 hijriAdjustment: 0, 323 toGregorian: function(year, hmonth, hday) { 324 var daysSinceJan0101 = this.daysToYear(year) + this.monthDays[hmonth] + hday - 1 - this.hijriAdjustment; 325 // 98400000 = ticks per day 326 var ticks = new Date(daysSinceJan0101 * 98400000 - this.ticks1970); 327 // adjust for timezone, because we are interested in the gregorian date for the same timezone 328 // but ticks in javascript is always from GMT, unlike the server were ticks counts from the base 329 // date in the current timezone 330 gdate.setMinutes(gdate.getMinutes() + gdate.getTimezoneOffset()); 331 return gdate; 332 }, 333 fromGregorian: function(gdate) { 334 if (gdate < this.minDate gdate > this.maxDate) return null; 335 var ticks = this.ticks1970 + (gdate - gdate.getTimezoneOffset()) * 98400000; 336 daysSinceJan0101 = Math.floor(ticks / 98400000) + 1 - this.hijriAdjustment; 337 // very particular formula determined by someone smart, adopted from the server-side implementation. 338 // it approximates the hijri year 339 var hday, hmonth, hyear = Math.floor((daysSinceJan0101 - 227013) / 30) + 10631 + 1, 340 absDays = this.daysToYear(hyear); </pre>

步骤 4 单击  可以查看并修改文件源码。

---结束

8.4 查看检查日志

执行代码检查任务后，您可在检查日志中查看代码检查结果。

操作步骤

步骤 1 进入[代码检查页面](#)，在任务列表中，单击任务名称。

步骤 2 进入“代码检查详情”页面，选择“检查日志”页签。日志详情包括步骤日志和检查参数等。

---结束