云效

使用指南

使用指南

企业管理

企业信息和成员管理

创建企业

创建入口

如果用户还没有创建或加入任何企业，第一次登录会提示创建新的企业，点击即可开始创建企业：

如果用户已经加入某个企业，可以通过右上角”+”入口新建企业：


搜索
新建任务
新建缺陷
新建需求
新建企业
新建项目
新建项目集

填写企业名称

输入所在企业的名称，然后点击立即创建即可完成企业创建：

进入企业

企业创建完毕后当前账号会自动进入创建的企业，然后开始进行项目、工作项、应用、代码和发布等管理工作：

管理企业

管理入口

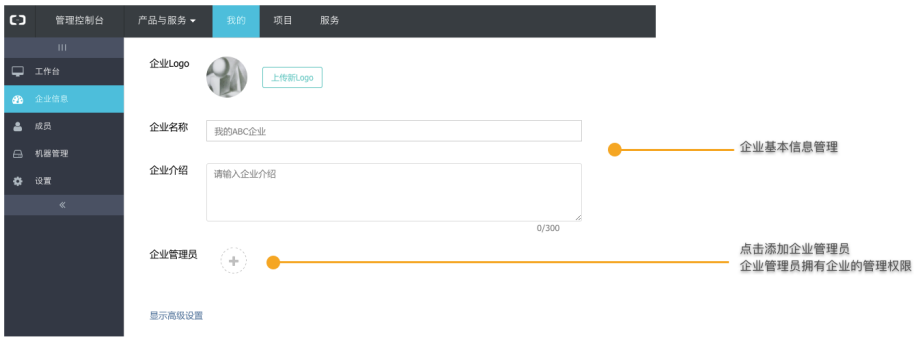
点击右上角企业名称，出现企业设置、企业切换和添加企业成员入口：

- 点击企业设置可进入企业基本信息管理界面
- 如果同时加入多个企业，点击切换企业可进行企业切换
- 点击添加企业成员即可进入企业成员添加页面

基本信息管理

在企业信息设置界面:

- 可以设置企业Logo、企业名称和企业介绍等信息
- 可以添加企业管理员，该管理员拥有企业的管理权限



企业成员管理

点击顶部导航的添加企业成员入口或企业基本信息左侧的“成员”进入成员管理界面，点“添加成员”可邀请成员加入当前企业：

点这里立即体验RDC

项目协作

项目与项目集管理

项目与项目集管理

在云效项目管理中，项目分为两类：
**

- **第一类**：标准的研发项目。项目团队在项目里一起协作，对需求进行分析、排期、迭代实现、测试和发布。
- **第二类**：业务空间。用来进行业务线划分，并管理各业务线的需求、文档和缺陷等。

研发项目和业务空间在功能上没有差别，只是概念上的分离，且二者在默认启用的服务种类上有所区别。例如，创建业务空间时，默认启用的服务包括：需求、缺陷和文档；而研发项目还默认启用迭代、测试用例等服务。但创建完毕后，业务空间和研发项目可以随意增减这些服务。

项目类别	适用场景
研发项目	专项成立、有明确结束时间的项目。例如，双十一大促某专项项目：没有明确结束时间，但研发团队比较固定，对某个产品或应用不断迭代实现功能、进行发布。
业务空间	PD 专门用来管理产品或业务的需求池，用于进行业务划分的父子结构。例如，先在某个 BU 建立一个总的空间，然后按不同业务线划分子业务空间。

项目管理

1. 入口

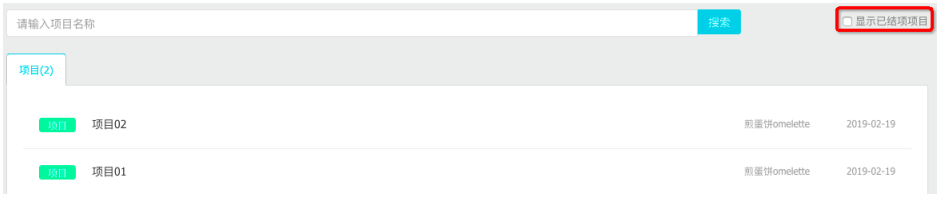
点击顶部 **“项目”** 导航，选择 **“项目列表”**。页面展示用户收藏、参与的项目以及创建的项目集。此外还支持在全站范围内搜索项目。

如果还没有项目，可以点击 **“新建项目”** 或

 选择 **“新建项目”** 进行创建

。

点击 **“查看全部”** ，可以搜索自己有权限且想要查看的项目，可以显示已归档的项目。



2. 创建项目

项目创建向导会引导你创建一个新的项目，你可以根据自己的使用场景选择研发项目或业务空间。选择“研发项目” ，将默认提供需求管理、迭代管理、测试用例、应用发布等服务，适用于从提出需求到发布上线的研发全流程的管理。选择“业务空间” ，将默认提供需求管理、缺陷跟踪等服务，可用作需求池缺陷池。无论选择

何种类型的项目模板，您都可以根据自己的需求，随时启用或停用其中的任何服务。

1. 点击 **“新建项目”** ，系统跳转至新建项目页面，如下图所示。
2. 选择项目类型、输入项目名称等信息，点击 **“确定”** “概况”**页。
3. 项目完结后，可以进行 **“归档”** ，归档后的项目可通过设置搜索条件搜索到。

3. 项目概况页

点击项目名称，进入某一个具体的项目后，可以在项目空间内管理项目成员、管理项目公告和设置项目各种属性、启用项目所需要的服务（注意：上述各种设置和服务启用需要项目管理员权限）：

项目权限模型

1. 权限类型

管理权限

- 项目中所有服务及数据的访问和操作权限（如创建编辑需求、任务、缺陷，创建编辑迭代，查看图表编辑图表等）
- 项目相关信息的设置权限（如成员管理、配置需求模板工作流、编辑项目基本信息、结项或删除项目等）

非管理权限

- 项目中所有服务及数据的访问和操作权限（如创建编辑需求、任务、缺陷，创建编辑迭代，查看图表编辑图表等）

2. 项目成员权限

- 管理员： 默认拥有管理权限
- PM： 默认拥有管理权限
- 其他角色：默认拥有非管理权限，可在项目中修改

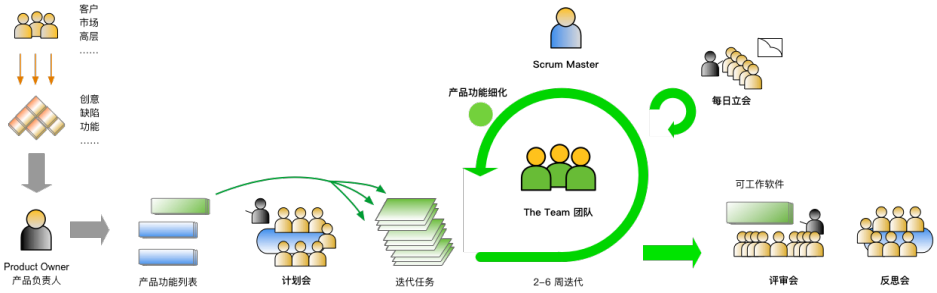
3. 非项目成员权限

- 公开项目：非项目成员默认拥有非管理权限，可访问项目，以及查看和操作所有数据（文档除外）
- 私有项目：非项目成员无法访问项目，也无法查看操作任何数据

管理迭代

迭代是敏捷开发的概念，是一种有开始时间和结束时间的轻量级计划，用来明确规划在开始和结束时间之间需要实现的需求、需要修复的缺陷和需要完成的任务。一个典型迭代的周期从 1 到 6 周不等，团队可根据自己的节奏或业务的需要来确定迭代周期。

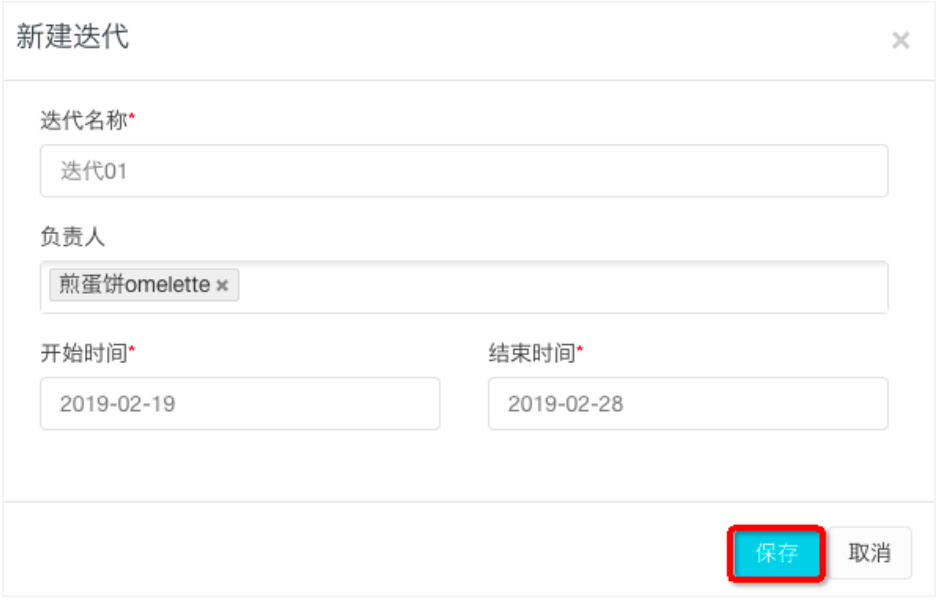
以典型的 Scrum 为例，迭代规划的具体流程为：用户和业务方提出的需求和缺陷，由产品负责人（PD）来统一管理，经分析、评估、拆分和PK后，确定优先级，在计划会（排期会）上和 ScrumMaster（PM）和研发团队进行排期，进入迭代。研发同学在迭代周期中，对自己负责的需求进行任务拆分、拉代码变更分支，并且每天更新进度和状态。在迭代的后期，需求实现评审后，进行发布，相关需求状态自动设为完成。至此，迭代完成，如果有未完成的工作，移到下一个迭代。



在云效中，迭代是一种服务，里面包含了便捷的迭代创建和规划功能，支持在迭代计划时快捷指定工作项（需求、缺陷、任务）、负责人并快速输入预计工时。

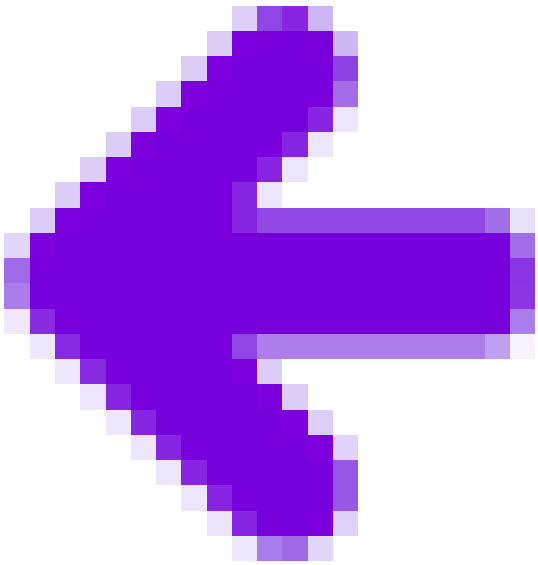
1. 新建迭代

1. 由项目左侧导航栏 **“迭代”** “设置 > 服务” **“迭代管理”** 页面。
2. 点击 **“新建迭代”** ，弹出 **“新建迭代”** 窗口，完成迭代信息填写并点击 **“保存”** 后。
3. 新建成功的迭代会展示在迭代列表中。

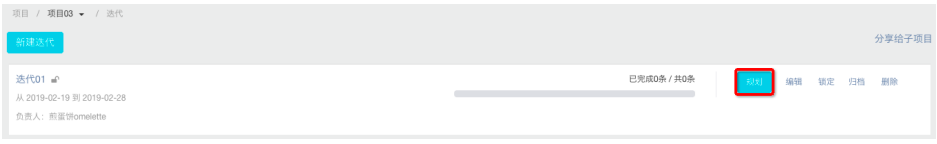


2. 规划迭代

1. 在迭代列表页，选择某个迭代并点击 **“规划”** 。
2. 进入迭代规划页面，勾选需要规划入此次迭代的工作项并点击



，随后，这些任务项便会展示在左侧列表中。



提示：

- 支持通过筛选条件对工作项进行高级过滤和搜索。
- 支持在如上页面直接修改工作项的优先级、指派给和预计工时等信息
- 点击需求、任务或缺陷的名称，系统跳转至工作项详情页，可对其信息和属性进行修改。
- 支持按照工作项的创建时间、更新时间、优先级等进行排序。

3. 迭代详情页

1. 返回迭代列表页面，点击 **“迭代名称”** 进入详情页面，即可看到此迭代下的所有工作项。
2. 工作项支持三种视图方式：列表、树状列表、看板。
3. 在如下页面，工作项可按照创建时间先后或不同的分组显示。
4. 可直接对工作项状态、优先级（分为紧急、高、中、低）、工时等信息进行修改，或点击工作项名称进入其详情页进行信息和属性的修改。

项目集管理

项目管理协会（PMI）把项目集定义为：经过协调管理以便获取单独管理这些项目时无法取得的收益和控制的一组相关联的的项目。这个定义有点绕，简单一点来说，如果有一组互相关联的项目，需要互相协作而获得一个共同的目标，那么可以放在一个项目集里面进行统一管理，例如：双十一大促多项目管理和协同。

云效的项目集管理功能可以让您轻松管理和监控多项目的进展和风险，更好地促进项目间协作。

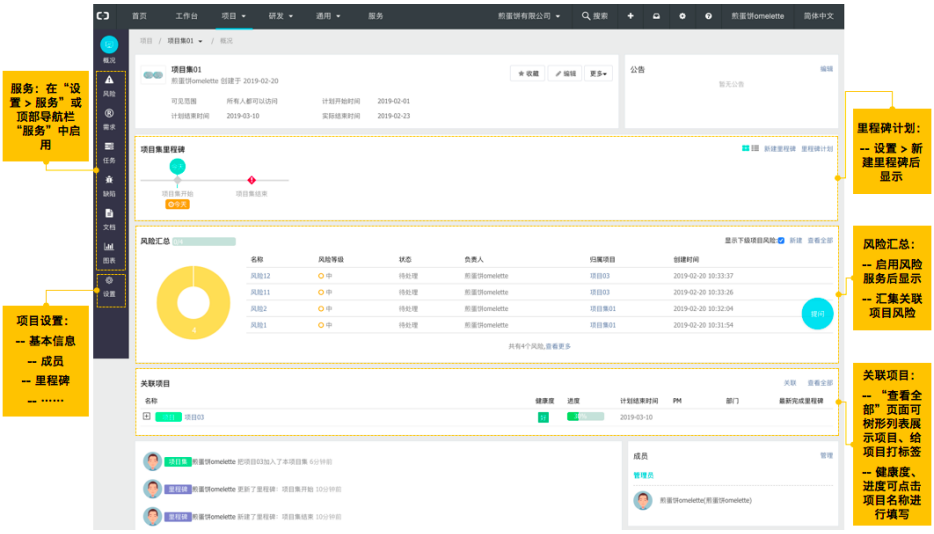
1. 创建项目集

在项目列表页，点击 **“新建项目集”** 或

 选择 **“新建项目集”** ，然后根据向导创建项目集。项目集创建后，列表和搜索和普通项目一样，统一管理。

1. 点击 **“新建项目集”** “确定” ** 即可创建成功。
2. 创建成功后，进入项目集概况页。可对项目集相关信息，如：里程碑、风险、公告、成员等内容进行编辑。

2. 项目集概况页

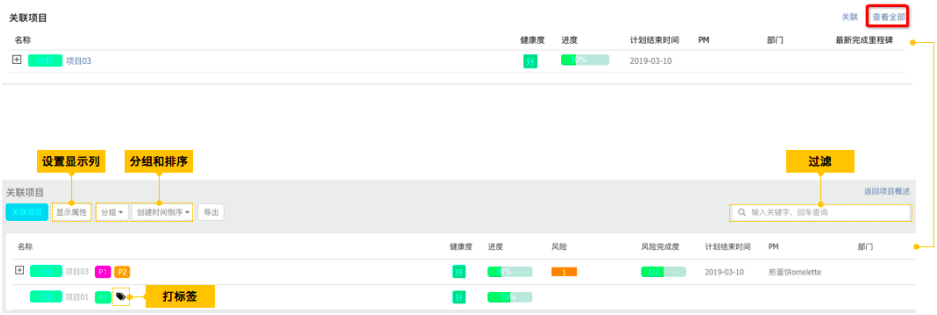


3. 管理关联项目

通过条目化项目列表管理，可以方便查看项目信息，了解项目进度、健康度和风险等信息。另外，通过标签功能，可以方便对项目进行任意维度的分组管理，以快速管理重点关注项目。

过滤/分组/排序

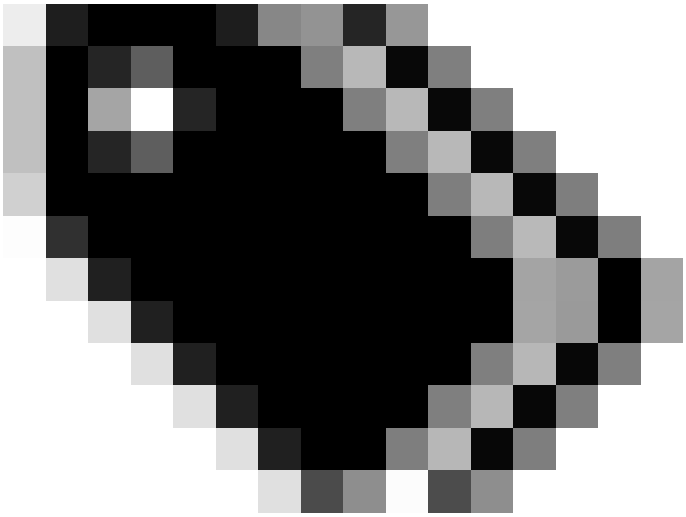
在项目集概况页可直接关联项目。点击 **“查看全部”** 后进入关联项目列表页，可设置显示列、分组和排序、过滤、还可以给项目打标签。



给项目打标签

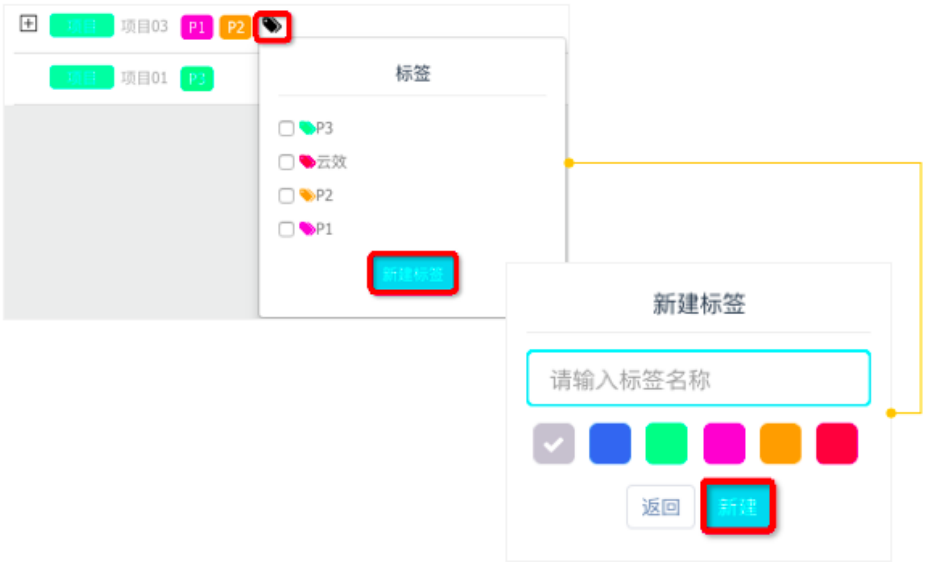
**

1. 在关联项目列表页面，将鼠标置于某个项目上时，会出现标签图标



。

2. 点击 **“新建标签”** “新建” ** 即可给项目打标签，然后可根据标签对项目进行过滤和分组。

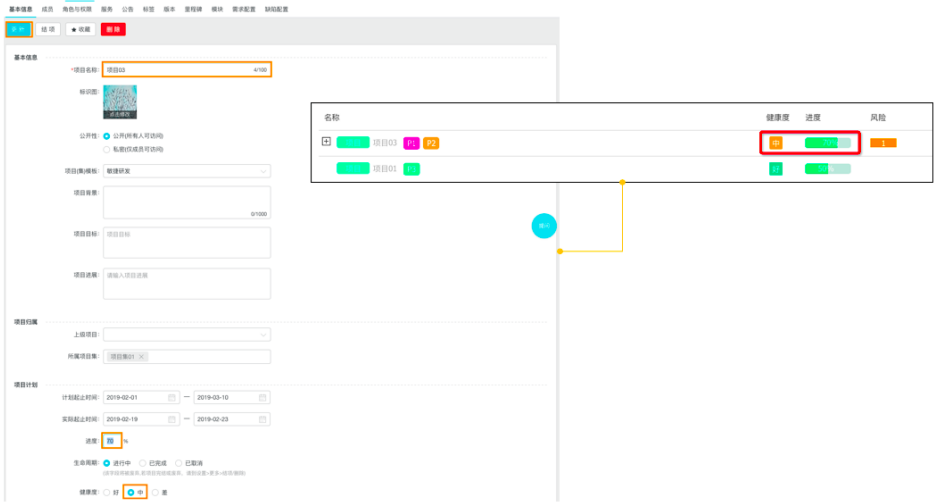


项目基本信息

**

项目的进度和健康度等信息分权给关联项目的负责人进行填写，然后实时汇总到项目集。

1. 点击项目左侧导航栏 **“设置”** 进入项目基本信息页面。可填写所属项目集、项目计划起止时间、实际起止时间、健康度、进度和项目类别等信息。
2. 也可对项目的成员、服务、标签、里程碑等进行设置。
3. 项目集关联项目的进度、健康度等信息更新后，会在关联项目列表中显示。

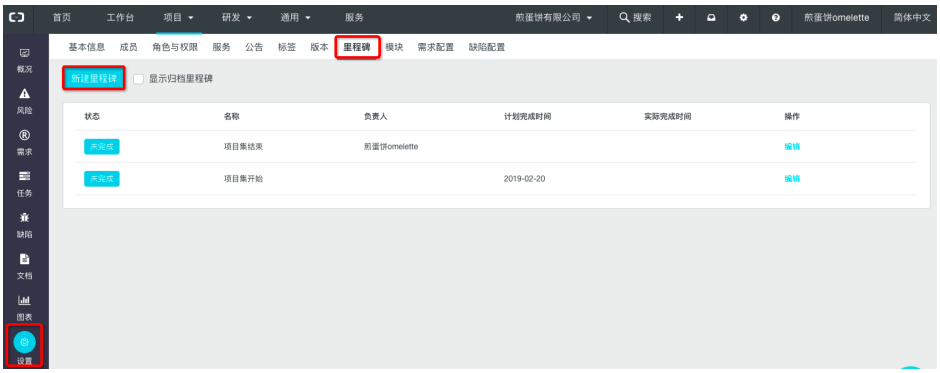


里程碑计划管理

里程碑计划功能让项目集管理者清晰定义项目集目标和任务，并对关联项目的里程碑计划进行实时监控。

1. 启用里程碑计划

在某一项目或项目集中，点击左侧导航栏中的 **“设置”** ，然后点击 **“里程碑”** tab 页。新增了第一个里程碑数据后，在项目或项目集概况页中便会显示里程碑区块。



2. 管理和计划里程碑

延期未完成的里程碑会高亮显示，里程碑完成后高亮消失，这样有助于项目和项目集负责人关注未完成的目标。

1. 点击上图右上角的 **“里程碑计划”** ，可进入详细的里程碑计划页面。
2. 关于里程碑的具体使用说明，请参考**里程碑计划**章节。

风险管理

风险管理功能使风险透明化，帮助项目和项目集管理者对风险进行实时监控，并对风险进行有效跟进和追踪。

风险分为高、中、低三个等级，是综合风险的影响和发生概率的一个指标。项目集管理者应在项目集全局范围内制定整体的风险等级标准，根据风险对项目集整体目标达成的影响来对风险进行定级。

1. 启用风险服务

在某一项目或项目集中，点击左侧导航栏中的 **“设置** > 服务” ，启用风险服务。启用后，在项目或项目集概况页会显示风险区块。项目集关联的项目创建风险对象后，会自动汇总到项目集风险区块里面。

2. 风险汇总

风险汇总区块自动汇总关联项目（在项目集里）或子项目（在父项目里）的未完成风险，并和当前项目集或项目的风险一起显示。

3. 项目集关联项目的风险信息显示

在项目集关联项目列表中，列出每个项目的风险完成占比信息。

4. 风险状态更新自动发送通知

在每一个风险对象的详情页里，风险状态发生改变时，或者对风险相关方案进行讨论时，会自动发送邮件和旺旺消息到作者（风险创建人）、指派给和抄送。

此外，可以将每个风险拆分多个子任务，分别指派给不同的人进行协同处理和跟进。

里程碑计划

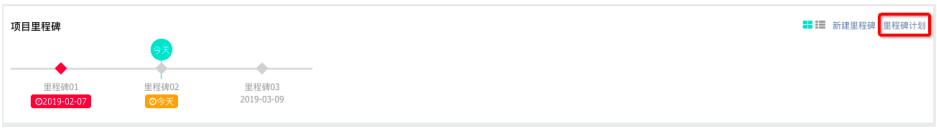
产品介绍

里程碑计划支持从项目集的维度，整体规划所有子孙项目集或项目的里程碑。

1. 支持项目树状视图，清晰展示项目集与项目间的层级结构。
2. 支持查看、编辑项目的负责人、起止时间以及时长等项目属性，轻松管理项目信息。
3. 甘特图上的里程碑支持拖拽，灵活更新里程碑完成时间。
4. 支持新建、编辑里程碑，以及快速更新里程碑状态。
5. 支持导出里程碑，便于数据统计、汇报。

入口

由项目左侧导航栏 **“概况”** 进入项目概况页。点击 **“里程碑”** 区块右上角 **“里程碑计划”** ，即可进入里程碑页面。



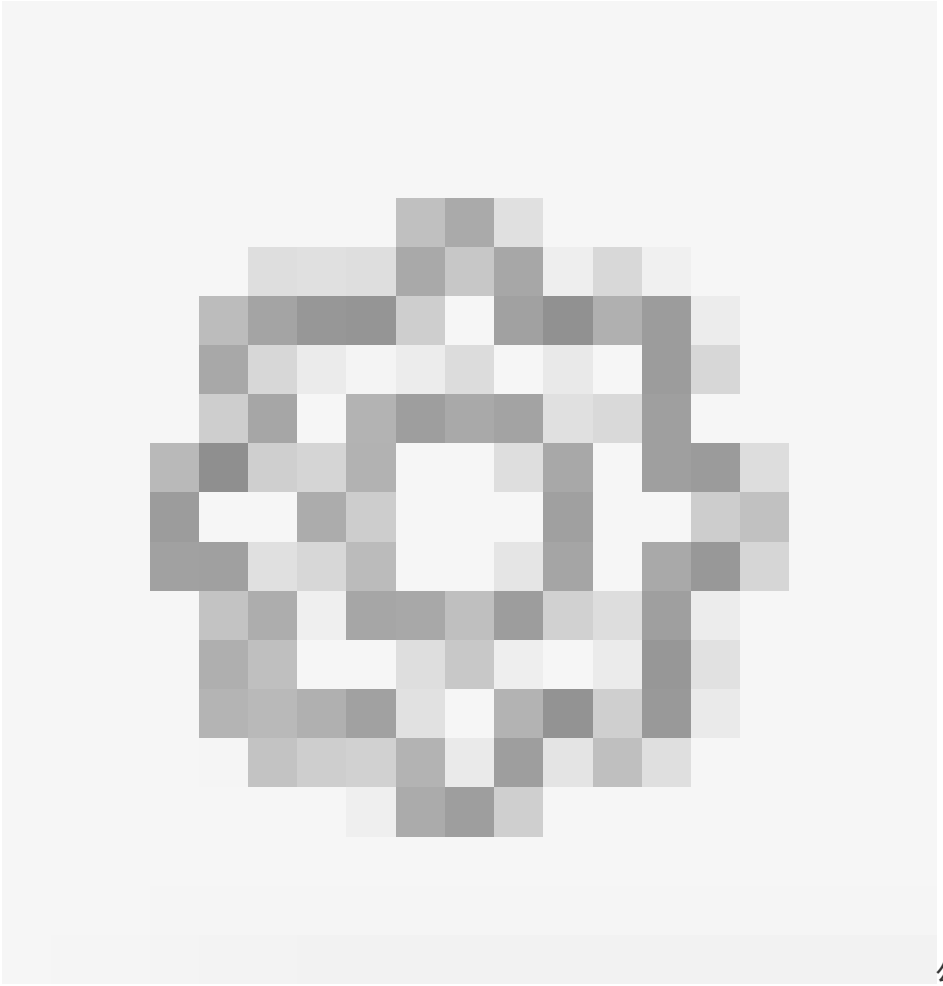
里程碑计划示例图

功能使用说明

1. 里程碑计划列表

进入里程碑计划后，页面左侧展示项目集计划相关信息，如项目集名称、开始时间、结束时间等；页面右侧以甘特图形式显示对应日期的里程碑计划。页面功能包括如下图所示几部分：

A. 配置显示列

点击  勾选信息项后即可显示在里程碑列表中。

B. 新建里程碑

1. 点击 **项目 (集) 名称** 左侧列图标

选择 **“新建里程碑”**。

2. 填写里程碑相关信息，点击 **“确认”** 后，即可创建新的里程碑。

C. 支持排序

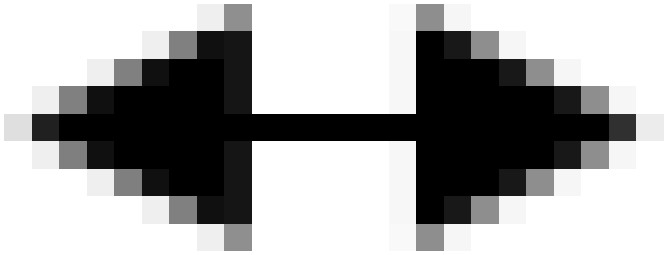
支持按照项目名称、开始时间、结束时间的升序或降序排列项目。

D. 更新项目属性

双击项目属性，如：项目成员（PM）、开始时间、结束时间等，可直接对其进行编辑。

E. 支持调整项目列宽

将鼠标置于图示位置后，会出现



符号，将其水平拖动可调整项目列表宽度。

2. 甘特图

里程碑计划页面右侧为项目里程碑的甘特图，页面功能包括如下图所示几部分：

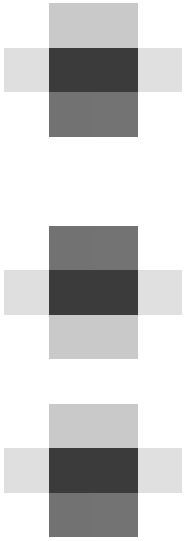
A. 图例说明

在里程碑图例中，里程碑节点根据时间及完成情况的不同而区别显示，里程碑节点状态说明如下。

B. 过滤

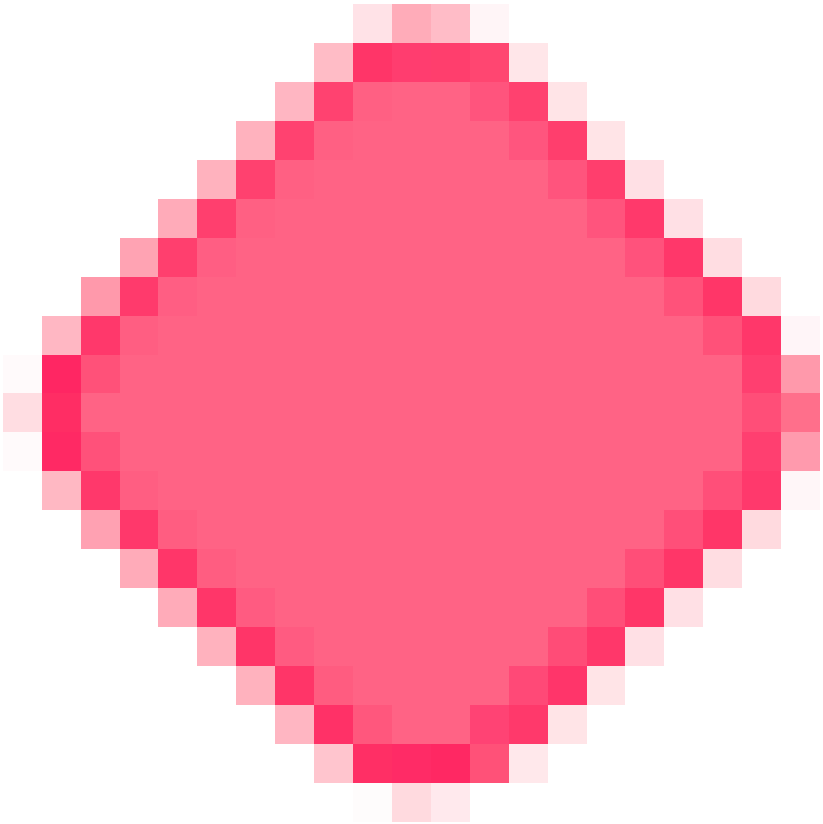
支持通过“超期未完成”和“项目标签”两个过滤条件，对里程碑进行筛选显示。

C. 支持导出



点击 ，支持“导出”、“显示项目分隔线”和“全屏显示”。

D. 支持拖拽里程碑



点击
新里程碑计划完成时间。

拖动里程碑，可快速更

E. 左右拖动甘特图

将鼠标置于甘特图上，出现手形图标

，水平移动鼠标，即可左右拖动甘特图。

F. 快速更新里程碑

点击 **“置位完成”** 或 **“置为取消”** 可快速修改里程碑状态。

G. 编辑里程碑

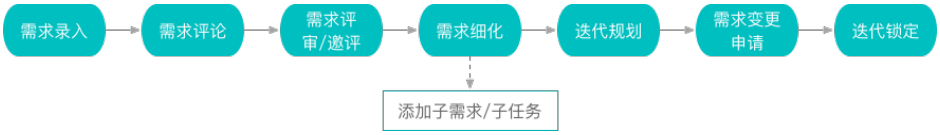
点击 **“编辑”** 可以对里程碑相关信息进行修改。

需求管理

入口

由项目左侧导航栏 **“需求”** 进入 **“需求管理”** 页面，支持三种视图方式：列表、树状列表和看板（若导航栏中无“需求”，可在 **“设置 > 服务”** 中启用）。

需求管理流程



功能使用说明

1. 需求录入

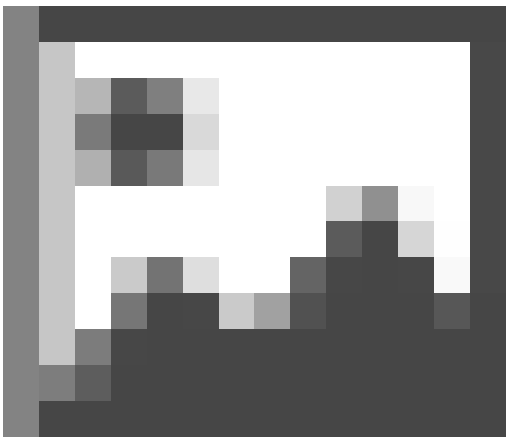
录入需求是比较简单的操作，点击图中的 **“新建需求”** 按钮，完成需求标题和正文的填写后，再点击 **“保存”** 即可。

提示：

- 编辑需求时，可点击 **“最大化”** 放大窗口。
- 编辑需求正文时，可直接按 **CTRL+V** 进行贴图。
- 编辑需求正文时，可点击 **“Markdown”** “富文本” ** 可预览效果。

录入需求步骤：

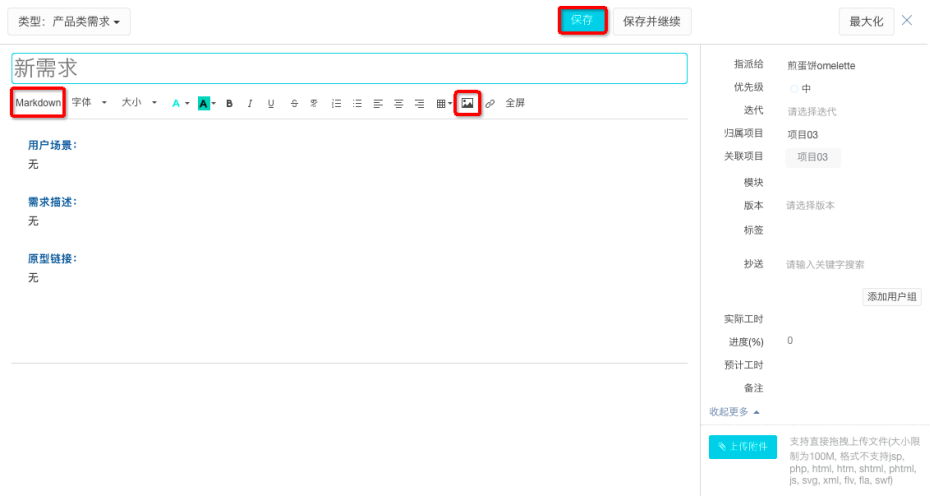
1. 点击 **“新增需求”** 填写需求标题和内容 **点击**



上传本地图片或

CTRL+V 贴图 “保存” **。

2. 保存后，点击 **“编辑描述”** 即可对需求正文进行修改。



2. 团队线上需求讨论

在形成结论之前，可利用 **“需求评论”** 功能对需求进行讨论。所有讨论会完整记录下来，且实时发送邮件和钉钉消息通知 **“指派给”** 和 **“抄送”** 用户（二者可在需求详情页指定），发出评论的人不会收到邮件和消息通知。

添加评论步骤：

1. 点击关联关系区下方的 **“评论”** “提交评论” **。
2. 评论提交后，可点击 **“回复”** 在该条评论的基础上再进行评论。

3. 发起需求评审

需求在线讨论后，形成结论，PD 把最终结论补充在需求描述正文。为了正式记录各方意见，PD 可选择对需求进行正式评审。

提示：

- 需求每次保存会产生一个修订版，系统会记录当前评审的版本。这样评审后即使有用户修改了需求正文，仍然可以找回当时评审的版本。
- 被邀请的评审人会收到钉钉消息，点击消息会进入评审界面进行评审，评审人可进一步邀请其他用户一起评审。

- 所有评审者通过后，整个需求评审才会通过。
- 需求评审可设置截止时间，超过截止时间如果仍然没有明确通过或不通过，会自动超时通过（建议线下先约定好此截止时间）。

发起需求评审步骤：

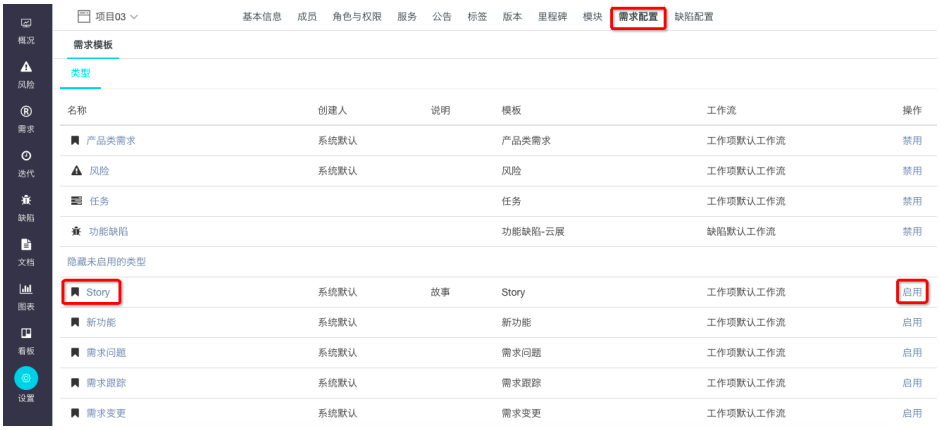
1. 点击关联关系区下方的 **“评审”** “发起评审” **填写主题、评审人、期望完成日期** “提交” **。
2. 评审人编写评审意见，选择通过与否或者待定，也可邀请其他用户一起评审。

4. 需求细化

在需求进入迭代前，需要对需求进行细化。一般来说，一个迭代一般是 1-2 周的周期，所以进入迭代的需求粒度不能太大，一般为 Story 级别。Story 是一个从用户角度出发的端到端的小粒度功能，符合 INVEST 原则：

I（Independent）：独立。
N（Negotiable）：可协商，不能定太死，开发过程可变通。
V（Valuable）：有价值。
E（Estimable）：可估算，不能估算意味着无法做相对准确的计划，一般是因为粒度还不够小。
S（Small）：足够小，一个迭代能够做完，不能跨迭代。
T（Testable）：可测试。

云效需求支持不断拆分，直到 Story 级别（**注意：** 本操作需到 **“设置 > 需求配置 > 需求模板”** 中启用 **Story** 需求类型）。



创建子需求步骤：

当需求范围过大时，可将需求拆分成子需求或子任务。点击 **“新建 > 子需求/子任务”** 可创建子需求；点击 **“关联 > 子需求/子任务”** 可将已有的需求设置为当前需求的子需求。

1. 在关联关系区，点击 **“新建** >子需求” **。
2. **“需求类型”** “确认” ** 后，子需求显示于列表中。
3. 点击子需求名称，进入需求编辑页面，可点击 **“编辑描述”** 编写子需求正文内容。

5. 与团队一起规划迭代

迭代一般由 PM 来创建和管理。每个迭代具体要排期哪些内容，PD 定优先级。研发团队根据 Story 估算、团队速率和可承受并发度等确定能做多少内容。

6. 需求变更申请

为了保证避免干扰，一个迭代周期内需要尽量保证内容的稳定性而不能随意增减需求和修改需求。云效提供迭代锁定的功能。锁定后，只有迭代负责人和项目管理员才有权限往里面增减需求和修改需求内容。如果一定要改，可以和迭代负责人协商或走正式需求变更申请流程。走需求变更流程后，如果所有需求变更评审人通过或超时通过，需求所作的变更（修改需求内容、需求移出移入迭代）会自动生效。

申请需求变更步骤：

1. 点击关联关系区 **“需求变更 申请需求变更”**。
2. 填写完变更信息填写页面后，点击 **“发送”** ，会生成变更评审。
3. 邀请到相关人员进行评审，评审人在 **“待办任务”** “钉钉” **收到消息后，点击链接后进行评审意见编写，选择通过与否或者待定，也可邀请其他用户一起评审。

风险管理

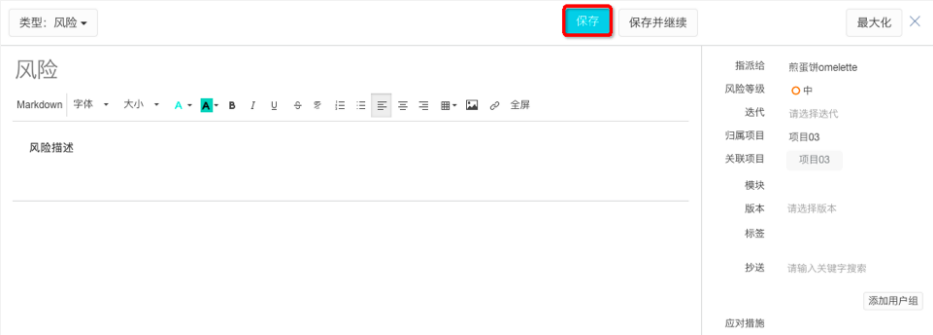
入口

由项目左侧导航栏 **“风险”** 进入 **“风险”** 管理页面（若导航栏中无“风险”，可在 **“设置 > 服务”** 或顶部导航栏 **“服务”** 中启用）。

功能使用说明

1. 新建风险

1. 点击 **“新建风险”** “保存” ** 即可。
2. 风险列表支持两种试图方式：列表和树状列表（与需求管理的列表、树状视图相似）。
3. 风险分为高、中、低三个等级。



2. 风险详情

在风险列表点击标题，右侧显示风险详情，点击 **“最大化”** 可进入全屏的详情页面。

- 左侧主体为描述区，可编辑风险的描述内容。
- 可新建子任务、设置关联关系、添加评论、查看描述的历史修订、查看操作记录。
- 右侧为属性区，展示和编辑系统默认属性和自定义属性（可在 **“设置** 需求配置 **类型** 风险 **模板”** 中配置）。

3. 风险上升

在大型项目关联中，当项目层级较多时，底层项目的风险，可选择性的向上层项目透出。如果某个风险比较重要，可在风险详情，点击上方操作栏的 **“风险上升”** ，可决定该风险需要上报到哪个层级。

迭代管理

迭代是敏捷开发的概念，它是有开始和结束时间的轻量级计划，用来明确规划在开始和结束时间之间需要实现的需求、需要修复的缺陷和需要完成的任务。一个典型迭代的周期从1到6周不等，团队可根据自己的节奏或业务的需要来确定迭代周期。

以典型的Scrum为例，迭代规划的具体流程为：

1. 用户和业务方提出的需求和缺陷，由Product Owner（产品负责人）来统一管理，经分析、评估、拆分和PK后，确定优先级，在计划会（排期会）上和ScrumMaster（迭代负责人）和研发团队进行排期，进入迭代
2. 研发同学在迭代周期里面，对自己负责的需求进行任务拆分、拉代码变更分支，并且每天更新进度和状态
3. 需求实现进行测试和验收后，进行发布，相关需求状态自动设为完成
4. 迭代完成后，如果有未完成的工作，移到下一个迭代

和团队一起进行迭代开发

创建迭代

迭代一般由ScrumMaster来创建和管理。ScrumMaster主要职责制定最佳工作模式，协调团队开发和跟进解决blocker，并保护团队避免受到外部干扰。

在项目里点左侧“迭代”TAB可创建迭代：

规划迭代内容

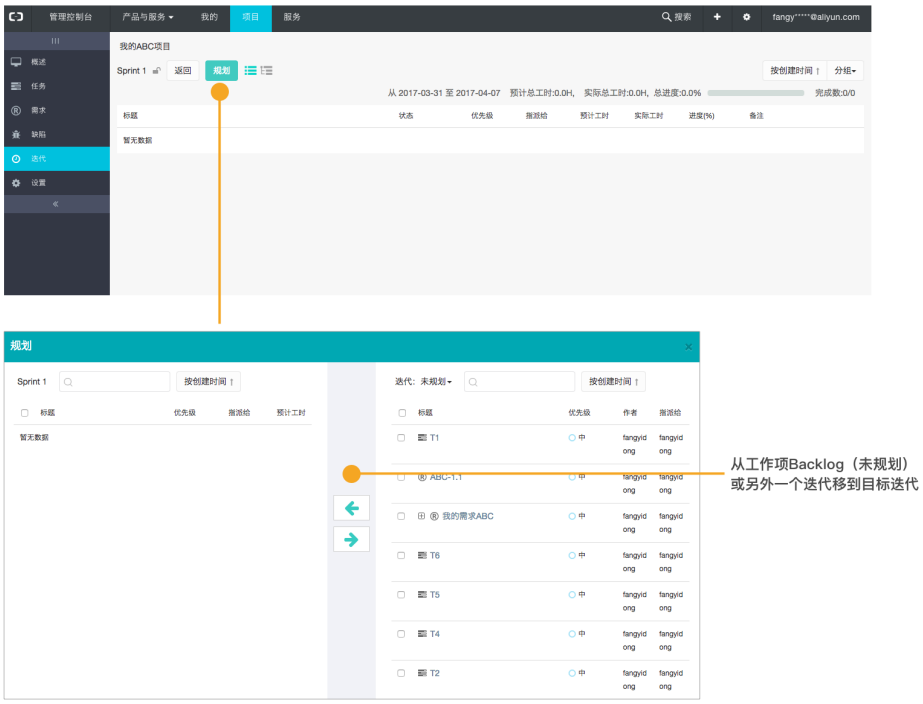
每个迭代具体要排期哪些内容，Product Owner定优先级，研发团队根据需求估算、团队速率和可承受并发度等确定能做多少内容。

在云效里面，把工作项（需求、任务、缺陷）规划进迭代有3种方式：

在工作项详情页，找到“迭代”字段，选择目标迭代

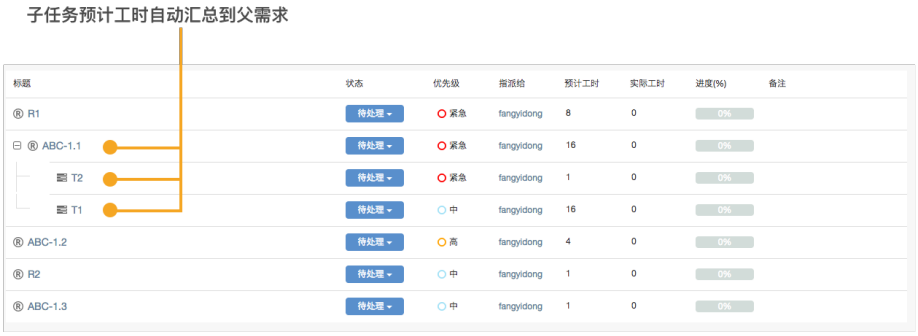
在工作项列表页，直接在迭代列点击选中目标迭代

在迭代里面，点“规划”按钮，可批量把工作项拉入迭代

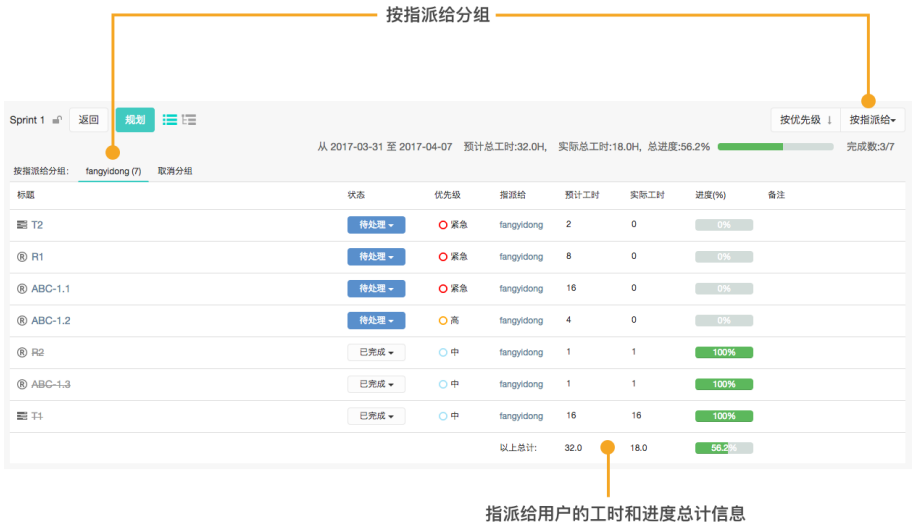


在规划迭代的时候，Product Owner按工作项优先级从高到低，进行需求讲解，然后研发一起进行预计工时评估（不需要很精确，而是快速进行评估），云效会自动对工时进行汇总，如果总工时到达团队一个迭代内可用工时，Product Owner停止讲解，并把剩余工作项移除迭代：

研发可以选择对需求进行任务分解，然后针对每个任务进行更细致的工时估算，这些工时会自动汇总到父需求:



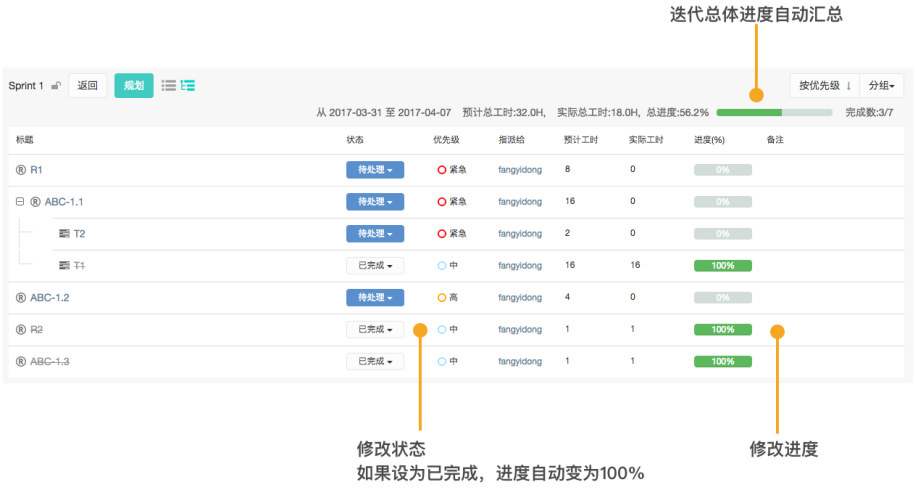
ScrumMaster可以对工作项按指派给分组，从而掌握团队成员的工作负荷分布，必要时，对任务分配进行平衡



：

迭代执行和跟进

研发负责的工作项完成后，把状态设为已完成，进度自动更新为100%，迭代总体进度会自动进行重新计算:



看板

云效看板支持看板方法标准实践，帮助团队更好的协作和管理交付过程。通过云效看板功能可以帮助团队：

- 更好地可视化端到端价值（需求）流动，确保产品、开发、测试等职能的前后拉通；
- 支持任务按所属模块或不同端展开为子列，确保不同模块或不同端任务向所属的需求对齐；
- 明确定义各列的流转规则，确保在交付过程中内建质量；
- 限制各列工作项的并行数目，促进需求的快速持续交付；
- 凸显交付过程中的问题、瓶颈和阻碍项，让团队聚焦应该关注的问题等。

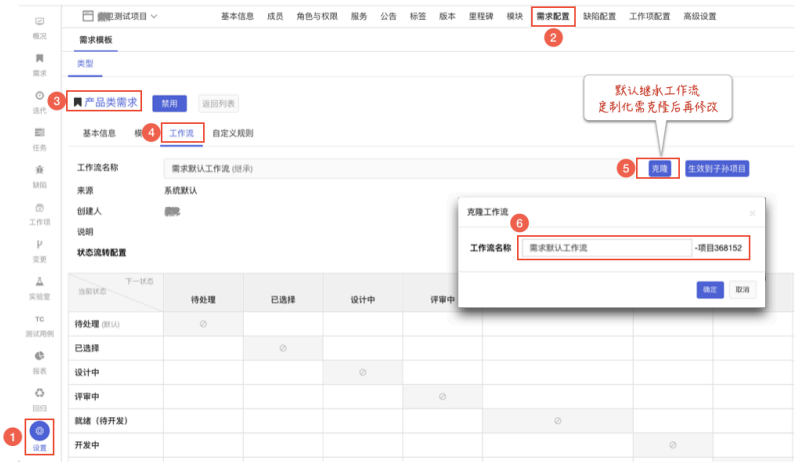
云效端到端价值（需求）流动的样例：

启用看板入口

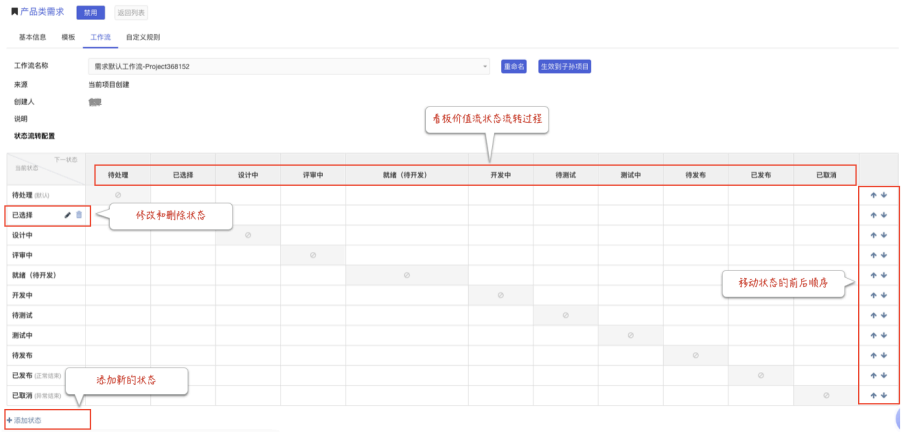
看板配置步骤和日常使用

1. 看板工作流配置

为了更好地可视化需求流动过程，确保产品、开发、测试等职能的前后拉通，需在看板上设置需求的流转状态，如需求池、需求分析、待开发、开发中、测试、上线等，这些流转状态将体现为看板的列布局。请参考下图进入需求工作流的配置。



请参考下图配置需求工作流，包括状态的增加、删除、编辑以及顺序调整等。



1. 需求和任务双层看板的支持

云效看板支持任务按所属模块或不同端展开为子列，从而支持由需求和任务构成的双层看板，确保任务向所属的需求对齐，引导团队快速完成需求的开发、联调和交付。

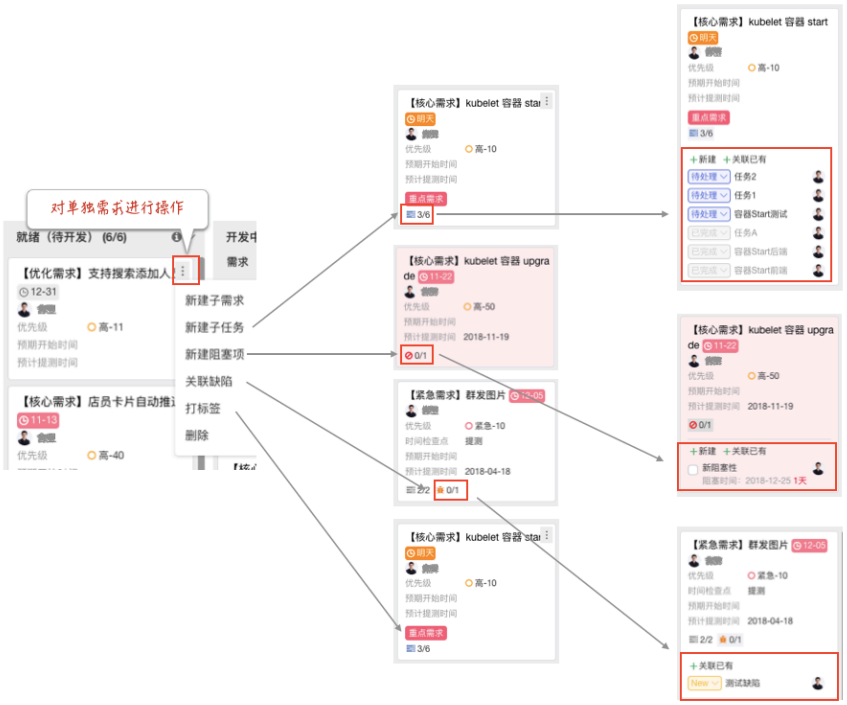
1. 定义流转规则

所谓流转规则是指需求进入特定状态（某个列）的准入条件，明确定义流转规则帮助团队将质量内建到开发过程中，而不是依赖最后的环节。请参考下图定义流转规则。

1. 限制各列工作项的并行数目

限制并行数目，是看板方法的标准实践，它帮助团队更即时的发现瓶颈并采取应对措施，从而促进需求更快速的流动，而不是在过程中积压。当某列的实际数目达到或超过上限时，则不允许新的需求进入，或提醒用户超限。

1. 日常问题进行标记和跟踪

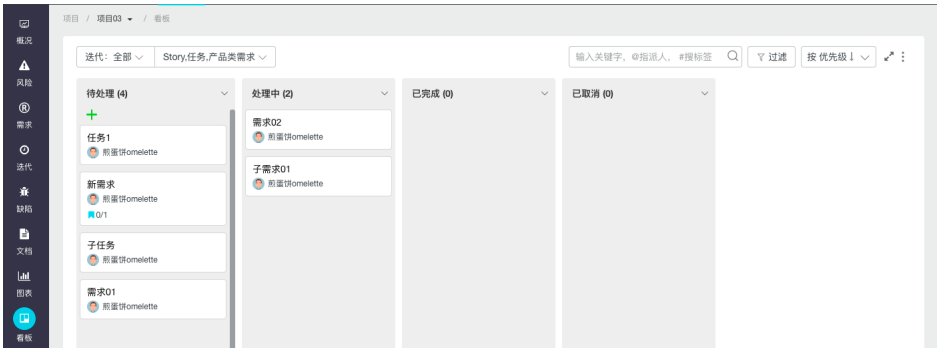


1. 应用看板进行日常跟踪

具体操作

入口

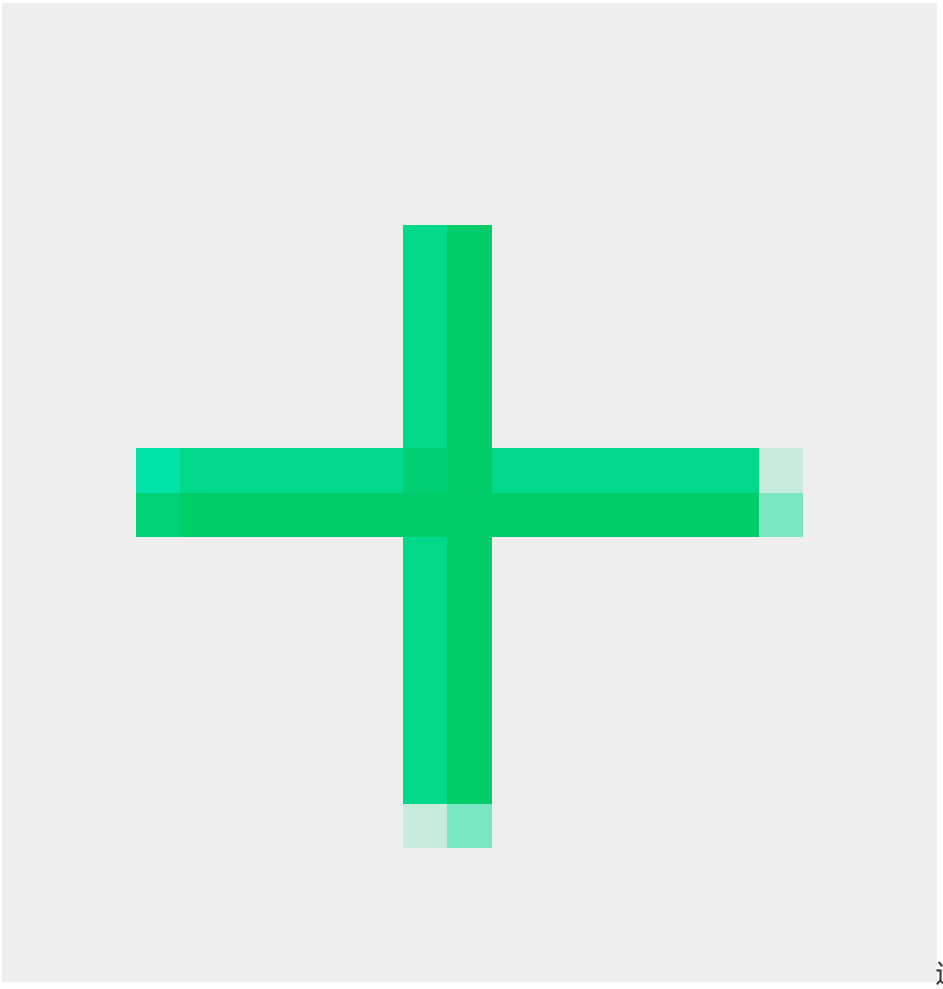
1. 由项目左侧导航栏 **“看板”** “看板管理” **“设置** 服务” **“服务”** 中启用）。
2. **“任务”** “需求” **“看板视图”** “点击进入新版看板” **即可跳转至看板管理页面。



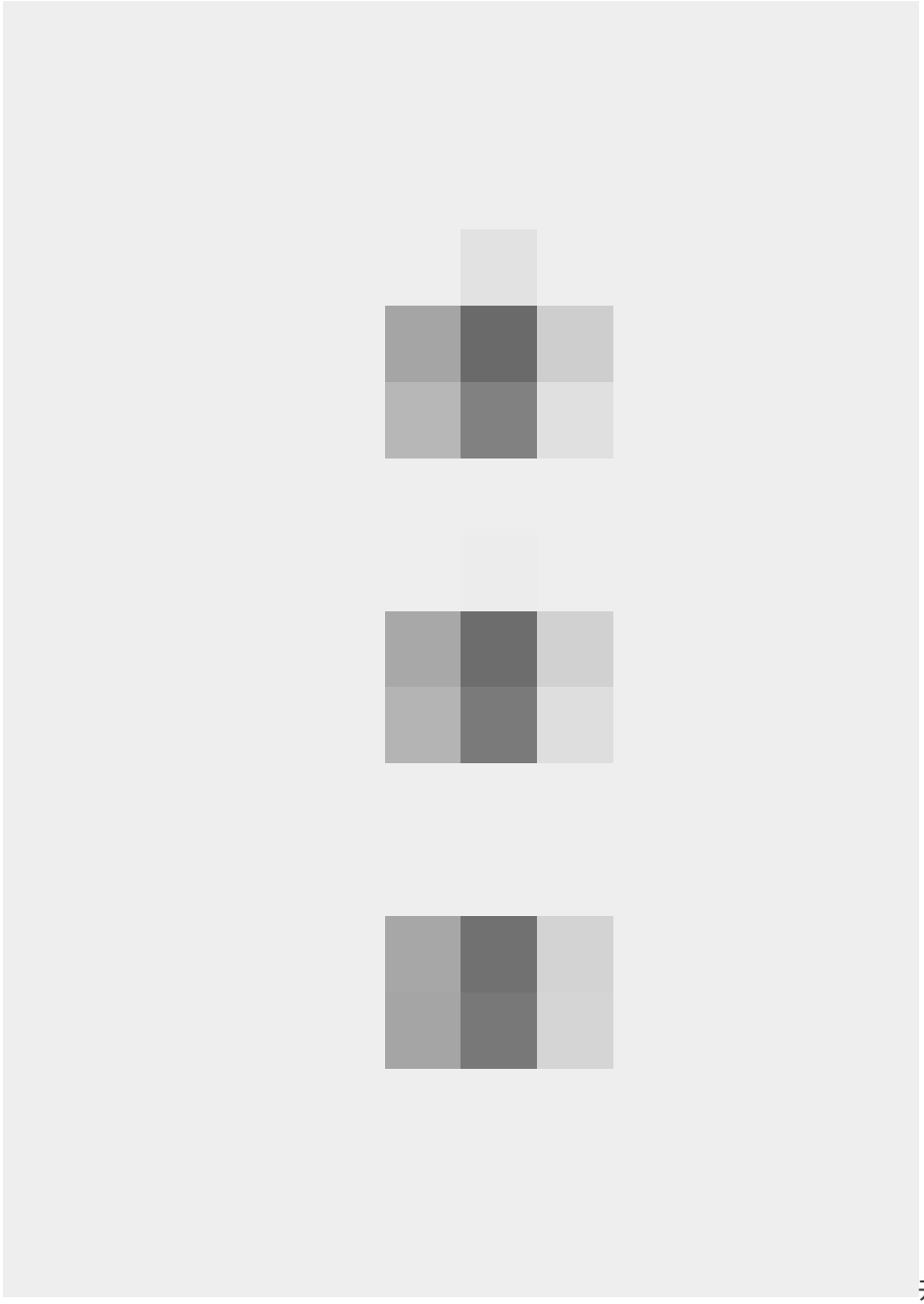
功能使用说明

下面将以需求为例，对看板的使用方法进行说明。

1. 新建需求

1. 点击  选择新建产品类需求，在输入框中输入需求名称，如“研发，从一个明确的需求开始”，然后敲击“回车”即可。

2. 点击 可选择指派人。
3. 需求新建成功后，将鼠标置于该需求上，点击



并选择 **“删除”** 即可删除该需求。

2. 拖动需求更新状态

当需求状态有变化时，长按此需求可将其拖动至相应的卡片中，如下图所示从 **“待处理”** 移动至 **“处理中”**

。

3. 打标签

支持给需求添加不同的标签，以便清晰显示需求属性。

1. 将鼠标置于需求上，点击

，选择 **“打标签”** 。

2. 输入标签名搜索并选择已有标签即可。
3. 如需新增或修改标签，可点击 **“管理标签”** 进行相关操作。

4. 添加子需求/子任务/阻塞项/缺陷

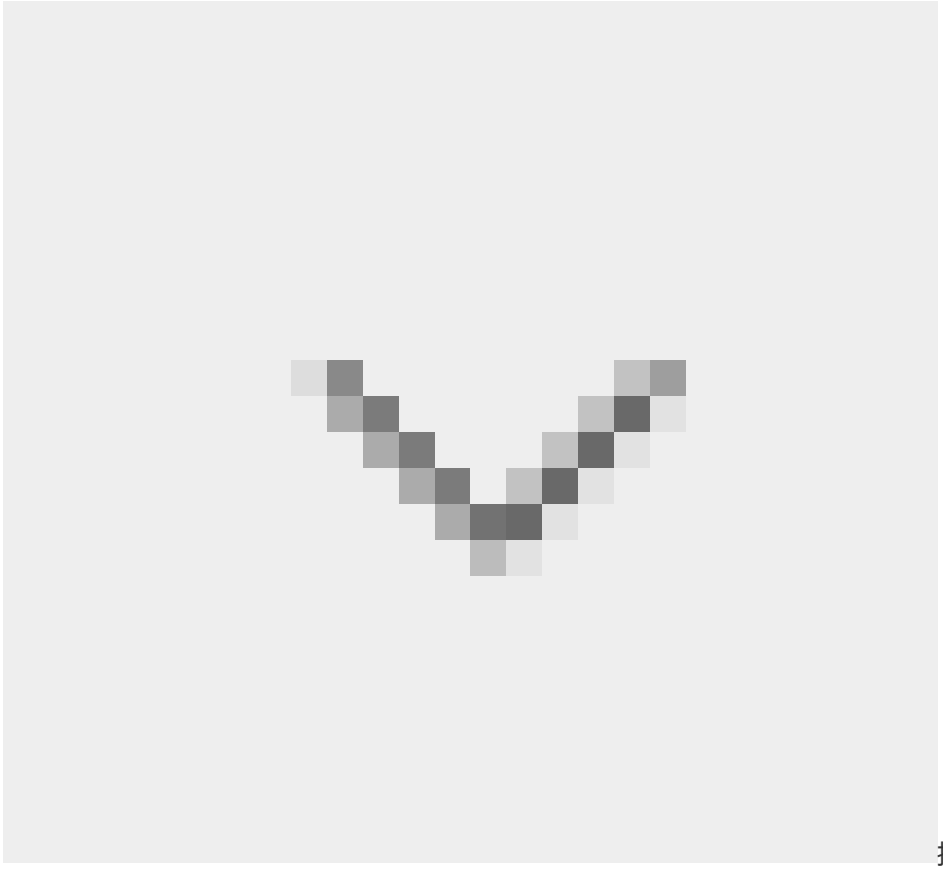
支持为需求新建或关联已有的子需求、子任务、阻塞项或缺陷。

1. 如上图所示，点击 “添加子需求” 、 “添加子任务” 、 “添加阻塞项” 或 “添加缺陷” ，如果是 **“新建”** 子需求/子任务/阻塞项，输入标题并敲击 “回车” 即可。
2. 如果是 **“关联已有”** 的需求/子任务/阻塞项/缺陷，则支持通过输入标题或 ID 对其进行搜索，然后敲击 “回车” 即可。
3. 支持便捷修改子需求、子任务等的状态。

5. 创建子任务分组

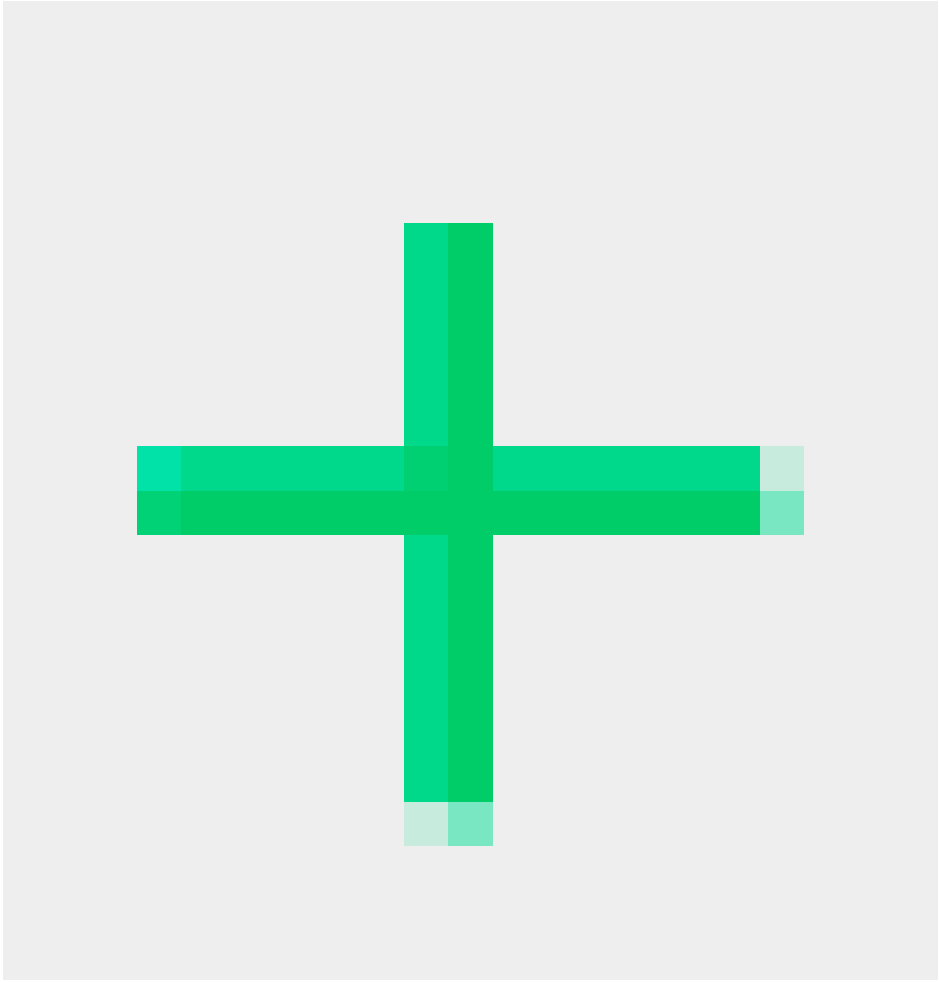
如果需求下存在多个子任务，支持对其进行分组管理（也可直接新增子任务）。

1. 点击各阶段卡片右上角

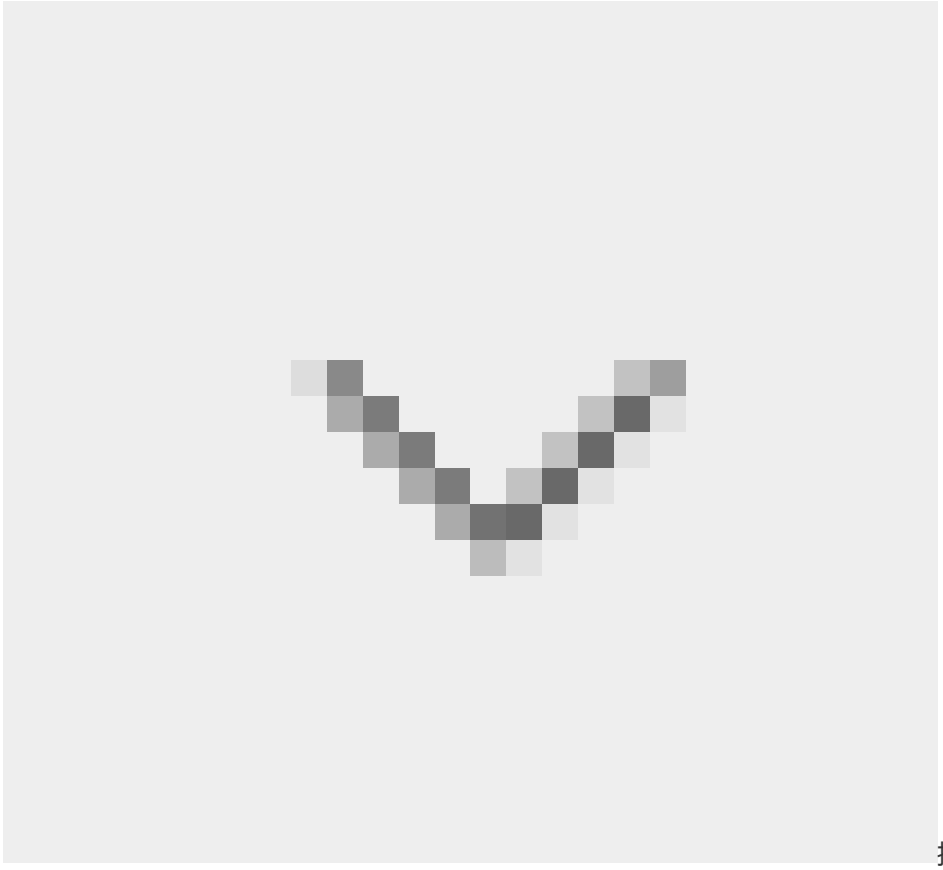


按钮，选择 **“展开子任务”**。

2. 新建子任务。点击

 ，输入子任务标题并敲击“回车”。

3. 创建分组。点击子任务卡片右上角



按钮，选择 **“在前面新建分组”** “在后面新建分组”** 即可在此子任务前后创建分组。

4. 分组创建后，可将子任务移动至任一分组中，并修改其状态，如“待处理”、“已完成”或“已取消”。
5. 支持删除、重命名分组。

6. 卡片数量限制

卡片数量限制即对并行需求数量进行限制。

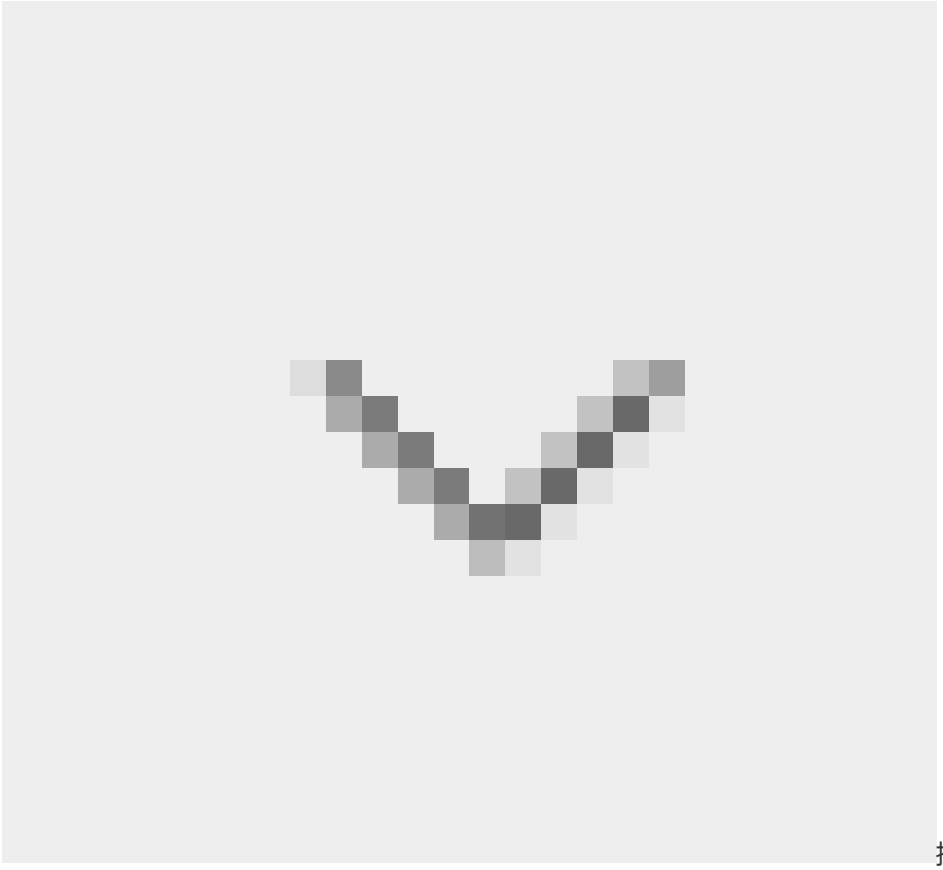
1. 点击各阶段卡片右上角

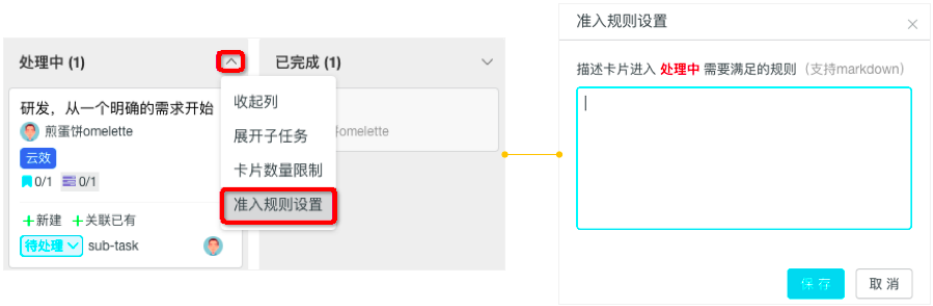
按钮，选择 **“卡片数量限制”**。

2. 各阶段中，默认允许的卡片数量最大值为 10，用户可自行修改。当超过卡片数量限制时，可选择 **“出现警告提示”** “阻止新卡片进入”**。
3. 设置完毕后，点击 **“保存”** 即可。

7. 准入规则设置

可以为各阶段的卡片设置需求进入的规则，其目的是使团队成员更加清晰状态流转，即何时可以将需求移动至何种阶段。准入规则最好是由团队成员共同协商制定。

点击各阶段卡片右上角  按钮，选择 **“准入规则设置”** ，输入规则并点击 **“保存”** 即可。



文档管理

产品介绍

项目管理中集成了在线结构化文档管理服务“文档”，方便用户记录项目中的文档、会议纪要等场景。

入口

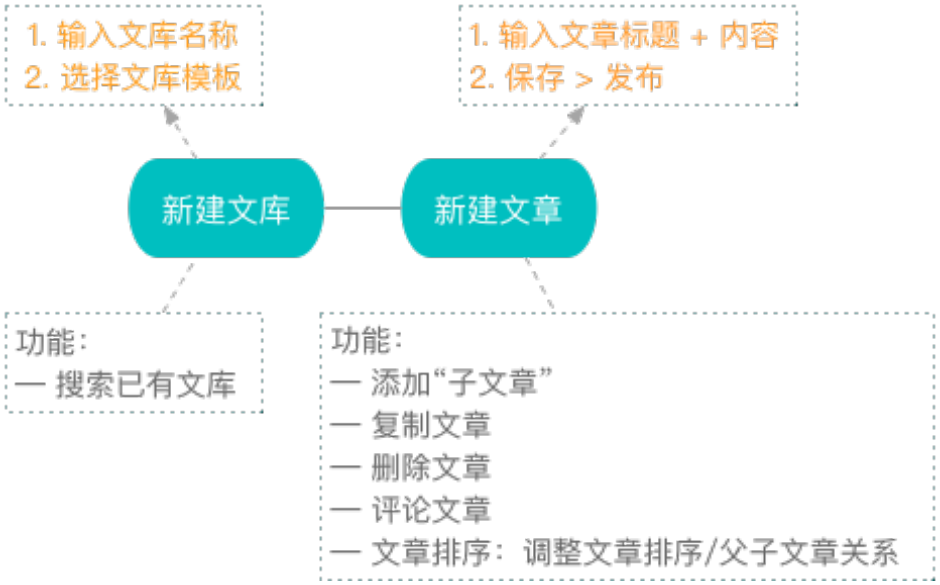
由项目左侧导航栏 **“文档”** 进入 **“文档”** 管理页面（若导航栏中无“文档”，可在 **“设置 > 服务”** 或顶部导航栏 **“服务”** 中启用）。

提示：

- 当文库数量较多时，支持通过文库名对文库进行搜索。
- 文库创建后，可点击 **“编辑”** 对文库名称进行修改。

功能介绍

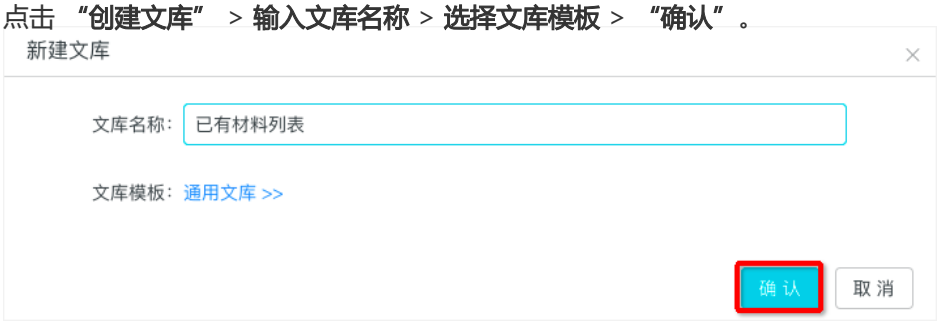
在文档管理页面，可创建文库并在文库下新建文章。



功能使用说明

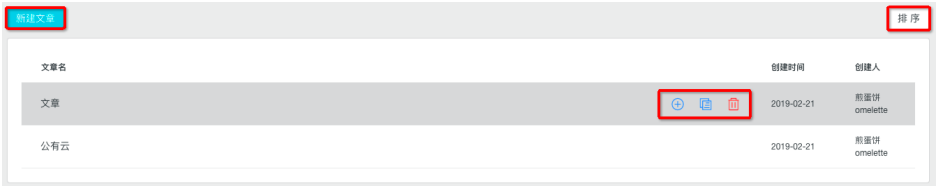
1. 新建文库

点击 **“创建文库”** > **输入文库名称** > **选择文库模板** > **“确认”** 。



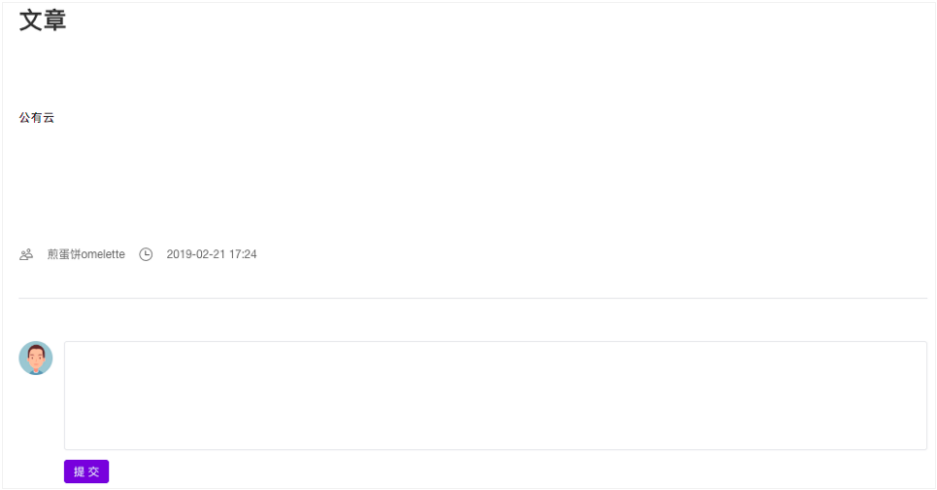
2. 新建文章

1. 选择文库，点击 **“新建文章** 输入文章标题和内容 **保存** 发布” **。
2. 文章新建后，可为此文章新增子文章、进行文章复制、删除。存在多篇文章时，点击 **“排序”** 可调整文章排列顺序以及父子文章关系。



提示：

- 文章发布后，其他用户才能看到文章内容。其他用户可对此文章进行评论，也可 **“回复”** 某条评论。
- 文章被删除后, 将无法找回。删除父文章时，其下所有子孙文章将被同时删除。若要删除文章，请先把所有子孙文章移动后，再删除。



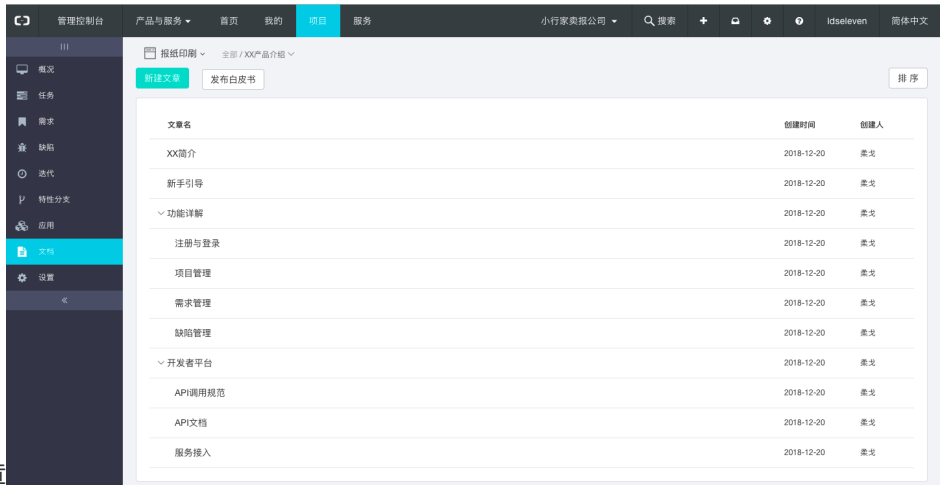
编写可对外发布的文档（如产品说明书）

公司的产品说明书、对外公示的文档资料等，都可在云效中编写并对外发布，发布后，公网环境的任何人都可以阅读此文档。创建文库时，选择“产品说明书”模板，即可编写此类文档。

操作说明

选择“产品说明书”模板，创建文库

新建、编写文章



3. 发布白皮书

1）发布：内容编写完成后，点击“发布白皮书”，将会把当前文档的内容发布到公网环境（此公网环境可访问的文档，下文称为白皮书），点击“前往阅读”可看到白皮书效果，获得此链接的任何人可访问阅读。

2）更新：发布白皮书后，可在云效中继续更新文档，点击“更新白皮书”，可将当前的文档内容更新到白皮书。

写好文档后，发布白皮书：

发布后，前往阅读，或继续编写，更新白皮书：

白皮书示例：

项目图表

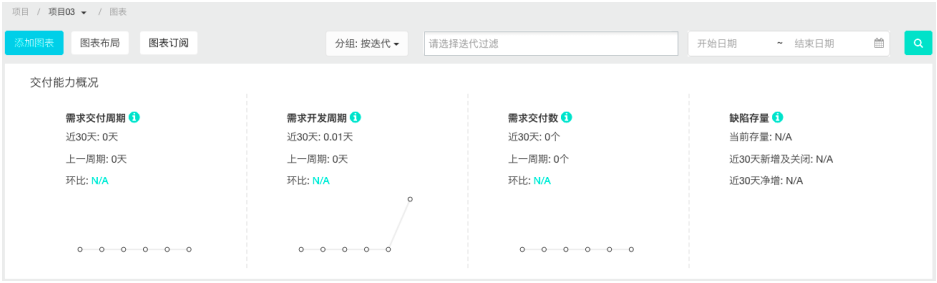
产品介绍

云效推出的项目图表功能，提供项目整个生命周期内的统计图表。采集的数据包括：项目进度风险、迭代、工作项（需求、缺陷、任务）等。

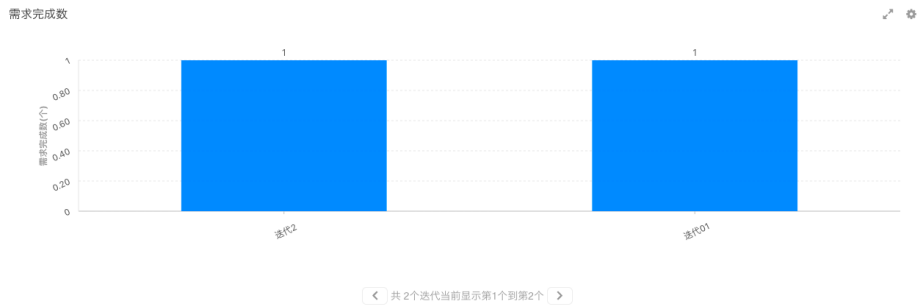
1. 提供敏捷实践图表，包括：有效缺陷平均关闭时长、需求研发阶段平均停留时长。
2. 支持灵活的图表自定义，自由选择统计对象、指标和分组方式（支持自定义属性）、过滤条件等。
3. 支持图表订阅，包括：配置定时任务、发送邮件通知。
4. 提供交付能力概况，展示敏捷实践重点指标。如：需求交付/开发周期、需求交付数、缺陷存量等。

图表示例

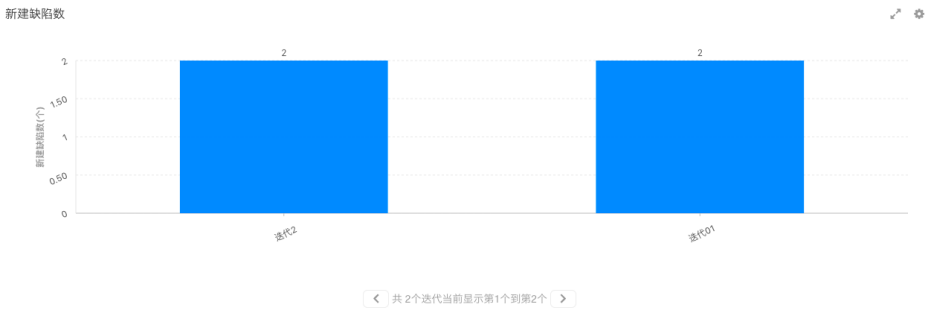
1. 交付能力概况



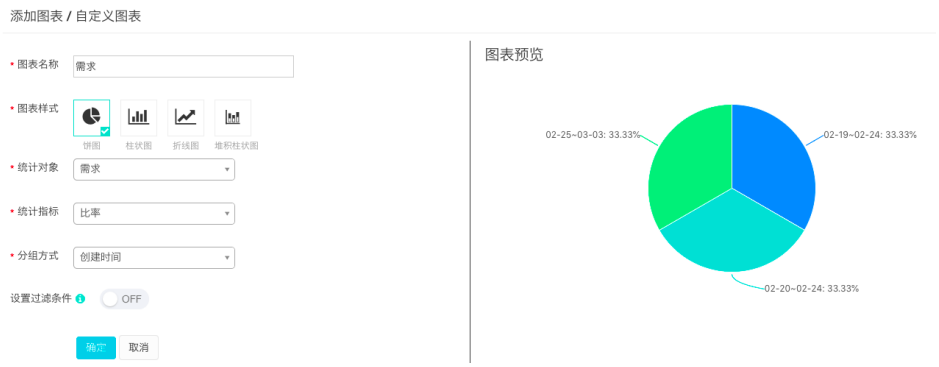
2. 需求完成数



3. 新建缺陷数



4. 自定义图表



5. 图表订阅

功能使用说明

1. 启用图表服务

由项目左侧导航栏 **“设置 > 服务”** 或顶部导航栏 **“服务”** 中启用 **“图表”** 功能。

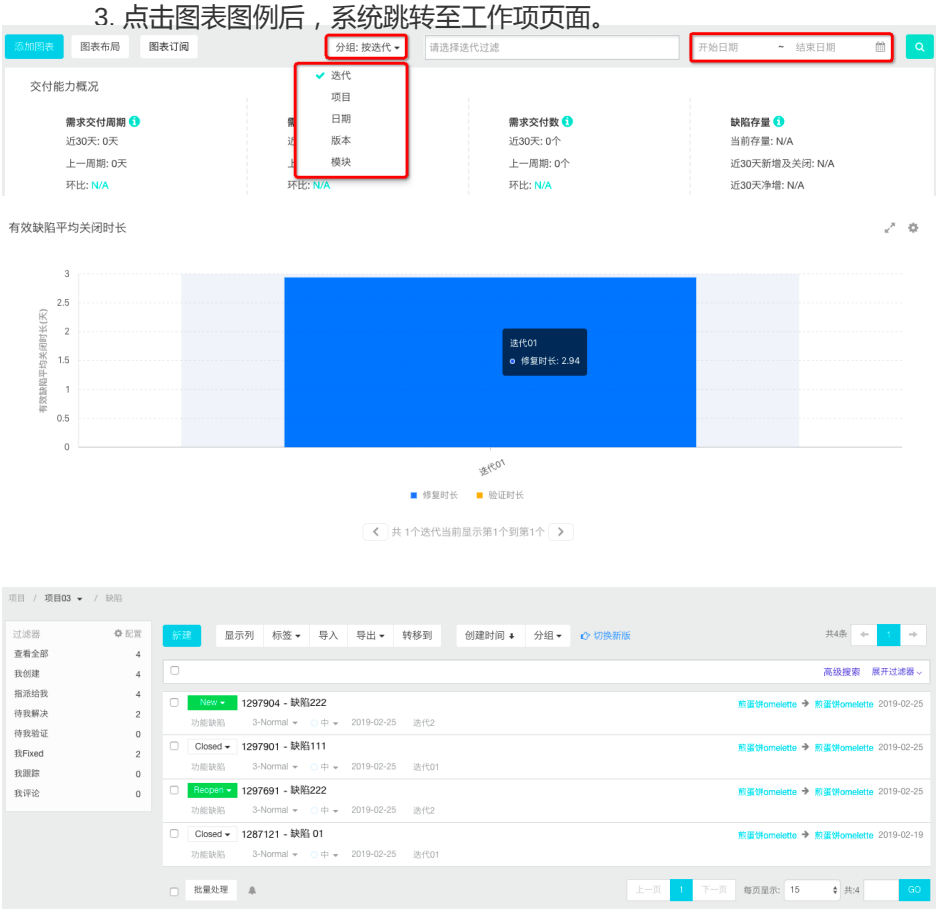
2. 添加图表

1. 进入“图表”服务，点击 **“添加图表”** 。
2. 点击 **“系统默认图表”** 勾选需要的图表类型 **“保存”** ，保存成功后页面自动跳转至图表列表页。
3. 点击 **“自定义图表”** 填写图表名称 **选择样式** 选择统计对象/指标/分组方式 > 设置过滤条件（可选）” **。

3. 图表交互

1. 可设置全局分组方式、统一时间过滤区间。
2. 支持 hover 图表图例（见图2）。

3. 点击图表图例后，系统跳转至工作项页面。



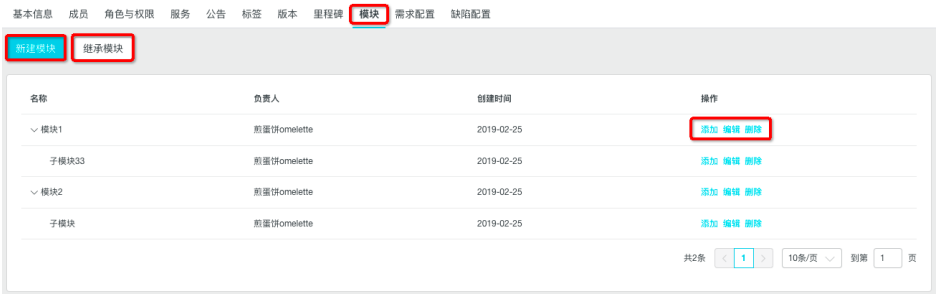
项目设置和企业设置

入口

在项目列表页点击项目名称，进入某一具体项目后，点击左侧导航栏 **“设置”** 进入项目设置页。可对项目基本信息、成员、标签、里程碑、需求配置等进行设置。

模块

1. 点击 **“模块”** tab 页可查看当前项目的模块、进行模块新建、编辑或删除操作。
2. 在新建或编辑模块时，可为该模块设置父模块，以树状结构展示。
3. 如果该项目的上级项目，即父项目中存在模块，在设置此项目模块时可点击 **“继承模块”** 从上级项目处继承模块。



- **工作项关联模块**

1. 在工作项详情的模块属性下拉列表中，可设置该工作项关联到哪个模块。
2. 在工作项列表中，可按模块过滤及分组，选择 **“包含子模块”** ，则过滤/分组的工作项数据，将按照父模块汇总，选择 **“不包含子模块”** ，则过滤/分组的工作项数据将按照其关联的模块平铺开来，对应到具体的模块上。

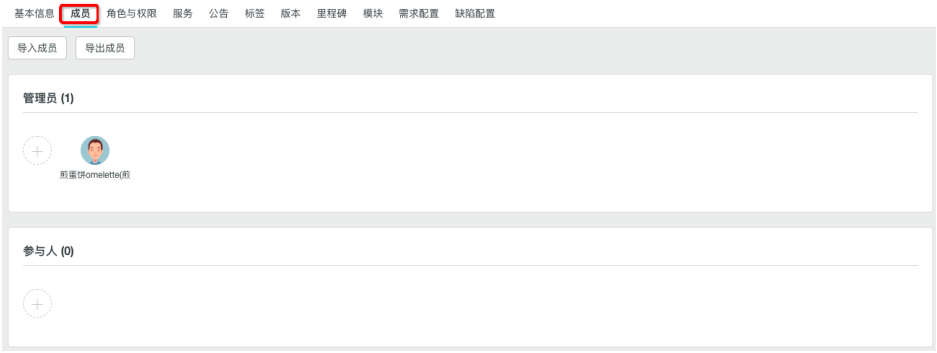
版本

1. 点击 **“版本”** tab 页中，可新建版本，并在已有版本下添加子版本，版本支持树状结构展示。
2. 版本不支持继承，可归档。
3. 工作项可关联到版本，关联后，可按照版本进行过滤或分组。

成员

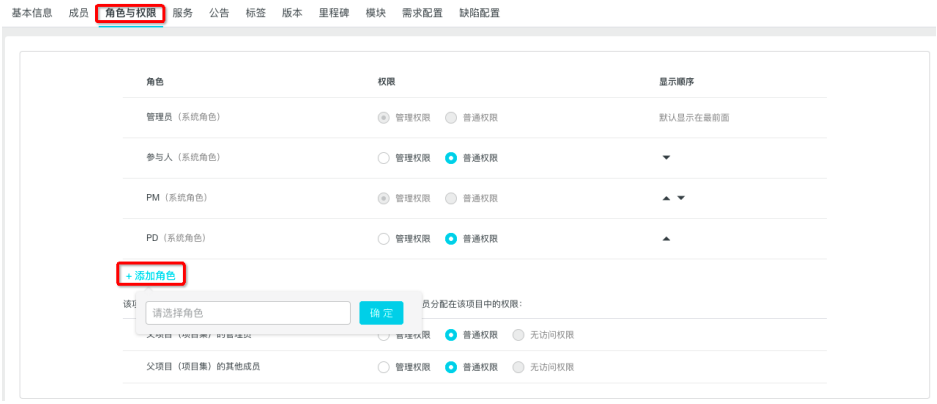
1. 点击 **“成员”** tab 页可编辑项目角色及成员。
2. 新建的项目，默认只有 **“管理员”**（默认为项目的创建者）和 **“参与者”** 两种角色。
3. 管理员可编辑成员，点击“导入成员”，可从其他项目中导入成员，导入的成员会追加到项目中，原有成员不变）；点击 **“导出成员”**，将以 Excel 形式导出成员。
4. 点击角色区块前面的加号

，可搜索企业中的用户，然后将其添加到该项目该角色中。



角色与权限

在 **“成员与权限”** tab 页可为项目添加角色，并设置角色权限。



项目管理员权限

1. 项目相关的配置操作，大部分需要管理员权限，不支持权限自定义。
2. 目前需要管理员权限的操作有：项目基本信息的修改、服务的启用禁用、成员的修改、新建模块、新建版本、新建里程碑、工作项配置的修改。

需求配置

1. 需求、任务可以有不同的类型，例如需求可分为产品类需求和技术类需求。系统默认提供一些类型，企业管理员可新增。
2. 每一种类型，可以配置自己的默认描述内容、模板、工作流（状态流转）。系统默认给每个类型配置了基本的系统属性，这些属性不能修改或删除，企业管理员可新增；系统默认给每个类型配置了基本的工作流，企业管理员可修改。
3. 企业管理员在企业设置页面的配置方案将生效到所有的项目。同时，每个项目可自己重新编辑配置，覆盖企业的设置。

缺陷配置

缺陷配置同需求配置。

企业级后台配置

入口

1. 一是：点击 **“项目** >需求配置 **>某个类型** >模板 **>添加更多属性** >点此新建（项目管理员权限）” **
2. 二是：点击云效顶部导航栏公司名称并选择 **“企业设置** >工作项流程与模板” **。


项目03
基本信息 成员 角色与权限 服务 公告 标签 版本 里程碑 模块 需求配置 缺陷配置
需求模板
类型
产品类需求
禁用
返回列表
基本信息 模板 工作流 自定义规则
模板名称: 产品类需求
创建人:系统默认
说明:
默认描述
继承上级项目/部门中设置的描述
全屏
用户场景:
无
需求描述:
无
原型链接:
无
名称 类型 描述 操作
继承自部门或上级项目的属性
状态 * 必填 单选列表 工作项的状态
指派给 * 必填 单选列表 工作项的指派人
优先级 * 必填 单选列表 工作项的优先级
归属项目 * 必填 单选列表 工作项归属于某一个项目
关联项目 多选列表 关联项目
迭代 单选列表 工作项所属的迭代
模块 多选列表 工作项的模块
版本 多选列表 工作项的版本
标签 多选列表 工作项的标签
抄送 多选列表 工作项抄送的用户
备注 文本
进度(%) 浮点数 进度百分比
预计工时 浮点数
实际工时 浮点数
+ 添加更多属性

类型/模板/自定义属性/状态/工作流

企业设置页面如下图，在此页面可切换到 **“类型”**、**“模板”**、**“工作流”**、**“自定义属性”** 和 **“状态”** 的 tab 页对相应数据进行新建和修改操作，这些操作将在整个企业内生效。

关联方案

1. 点击 **“关联方案”** tab 页，可设置需求、任务、风险的默认关联方案。
2. 关联方案决定某一类型使用何种模板和工作流。如下图所示，以类型 **“技术类需求”** “需求问题” **“工作项默认工作流”**。
3. 点击 **“编辑”**，可修改此类型使用的模板和工作流。

项目级配置

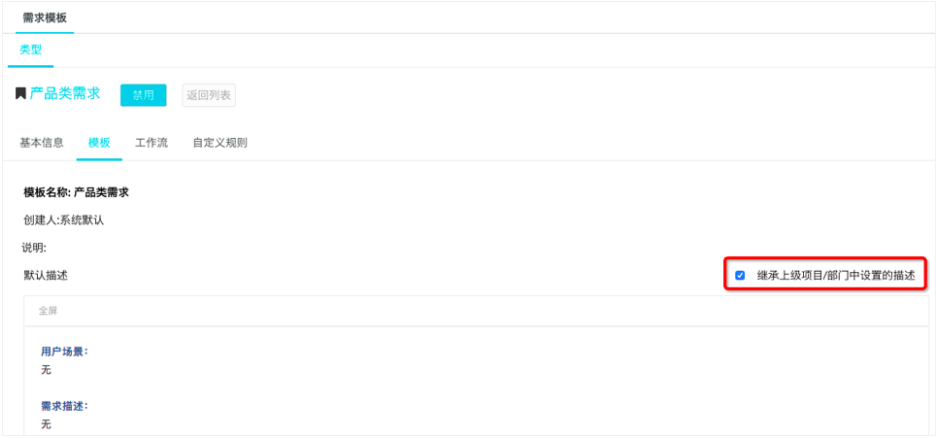
项目中类型的启用与禁用

1. 在企业中配置的类型，可在项目中看到。但只有 **“启用”** 后，才能在新建工作项时使用该类型。
2. 点击“**设置 > 需求配置**”进入该项目的类型列表，可见当前项目已启用的类型。在此页面可启用或禁用某些类型。

项目中类型的配置

- **类型的默认描述**

1. 在类型列表选择某一类型进入其页面后，点击 **“模板”** ，可看到当前配置的默认描述。
2. 类型的默认描述是选择性继承的，默认勾选 **“继承上级项目/部门中设置的描述”** 。去掉勾选后可自行修改描述内容。



- **类型的自定义属性**

类型的自定义属性是强制继承的：企业的配置 > 父项目的配置 > 子项目的配置：

1. 下层会累积继承所有上层设置的自定义属性。
2. 下层可以继续追加自定义属性，不能删除上层的属性。
3. 下层可以修改属性的可选值、默认值、提示文案。

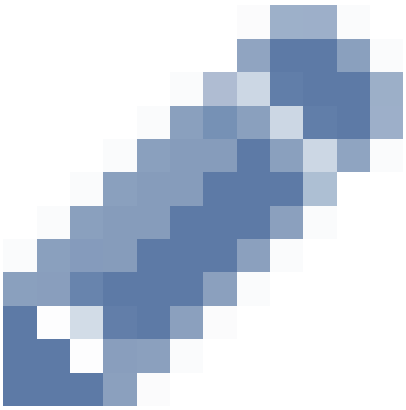
在类型列表选择某一类型进入其页面后，点击 **“模板”** ，可看到当前配置的自定义属性。可添加属性、修改属性的默认值、可选值，本项目添加的属性可设置是否必填等。

名称	类型	描述	操作
继承自部门或上级项目的属性			
状态 * 必填	单选列表	工作项的状态	
指派给 * 必填	单选列表	工作项的指派人	
优先级 * 必填	单选列表	工作项的优先级	
归属项目 * 必填	单选列表	工作项归属于某一个项目	
关联项目	多选列表	关联项目	
迭代	单选列表	工作项所属的迭代	
模块	多选列表	工作项的模块	
版本	多选列表	工作项的版本	
标签	多选列表	工作项的标签	
抄送	多选列表	工作项抄送的用户	
备注	文本		
进度(%)	浮点数	进度百分比	
预计工时	浮点数		
实际工时	浮点数		
✚ 添加更多属性			

- **类型的工作流**

1. 类型的工作流是默认继承的，无法直接修改。
2. 若想修改工作流，可点击 **“克隆”** ，克隆一份工作流后，即可进行添加状态、设置流转限制等操作，也可将当前的工作流一键生效到所有子孙项目。
3. 状态矩阵中，打勾表示从对应的 **“当前状态”** “下一状态” **，同时该行的** “当前状态” ** 将不能流转到其他未打勾的状态。
4. 如果某一行未勾选任何的流转，则表示该行对应的状态，可任意流转到其他状态。

克隆后，工作流可修改，当前状态旁出现



编辑及删除入口。右侧支持改变状态顺序，下方可添加状态。

工作项规则配置

工作项

需求、缺陷、任务统称为工作项，代表项目中需要处理的具体事务。

- 需求：通常来说，为实现一个用户诉求，我们会创建一条需求来跟进，例如“里程碑快到期时，需要通知项目管理员”。
- 任务：项目跟进或者需求处理的过程中，要完成许多的任务，我们可以创建任务来跟进，例如“加购2台服务器” “周五前完成需求评审”。
- 缺陷：如果产品功能有问题，我们可以创建缺陷来跟进，例如“里程碑到期时，消息通知重复发送了两次”

工作项工作流

需求状态：待处理 > 处理中 > 已完成 > 已取消（用户可自行配置，见项目设置和企业设置）

任务状态：待处理 > 处理中 > 已完成 > 已取消（用户可自行配置，见项目设置和企业设置）

缺陷状态：淘宝采用的缺陷生命周期模型从2008年就确定了，并且没有变化，状态如下：

New: 新增加的、需要解决的BUG。
Open: 正在定位问题，或正在解决中，或已经解决但未部署生效。
reopen：重新打开、激活，需要解决的BUG。
Fixed: BUG已经解决，并且修改后程序已部署生效。
Closed: 验证后，此BUG可以关闭。
Reopen: 此BUG需要再解决。
Later: 此BUG不在本项目的工作范围内，在后续版本中修复。
Worksforme: 不能在当前环境中重现。
Duplicate:和其它BUG描述现象重复。可以配置选择Duplicate状态时必填关联的缺陷ID。
Invalid: 属于测试人员对测试需求的理解错误。
External: 问题是由其他外部的原因引起，需要由外部处理。
ByDesign：属于按照产品设计实现，不是问题。
Wont’ fix：问题确实出现过，但是由于产品改动已经修复或者功能废弃，问题目前已不需要解决

缺陷的状态精简一下分为三类：待处理、已处理、已关闭。我们为了细化区分这三大类，人为地增加了状态。

状态其实应该是这样的一个级联分类：

待处理：New、Open、Reopen
已处理：Fixed，Wont’ fix，Later，Worksforme，Duplicate，Invalid，External，ByDesign
已关闭：Closed

已处理中的8个状态都可以视为问题处理人对此问题的处理意见。我们暂且叫它为：解决方案。

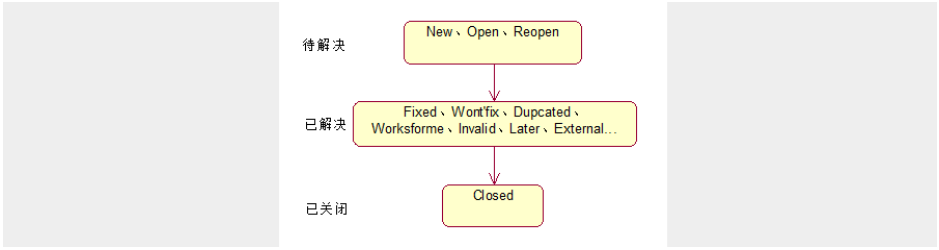
有效BUG：
1、解决方案为 Duplicate、Invalid、ByDesign的BUG是无效BUG，其余情况均为有效BUG（New、Open、Reopen、Fixed、Wont’ fix、Later，Worksforme、External）；
2、如果一个BUG被Reopen多次，则以最后一次的解决方案来判断是否是有效bug。

经过以上调整，我们就有了以下规则：

1、所有缺陷最终都应该由缺陷验证者或者与之平级权限的人变更到Closed状态。非Closed状态的都被认为是活动的缺陷。

2、解决者可以将“待处理”状态的缺陷置为任意“已处理”状态。

状态流转图如下图：



再用文字来解释一下上图的一个经典流程：

1、测试（验证者）发现了问题。提交一个问题。（状态为New)
2、开发（解决者）去解决。先Open或者直接选择一个解决方案。
3、测试通过就关闭它。如果验证后不正确，那就Reopen。再返回到步骤2。

流程自动化

工作项跟进的过程中，常常遇到一些流程跟进的问题。例如，你可能苦恼于需求状态总是更新不及时，数据统

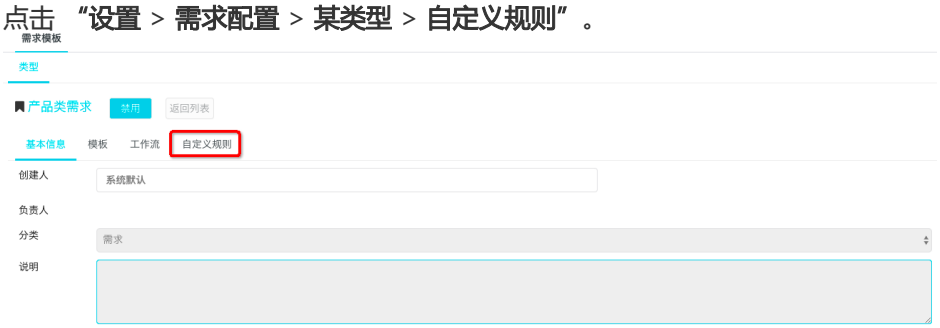
计不准确；你可能苦恼于制定的流程规约难以落地，例如需求评审通过后要记录一堆字段，有的需要指给开发，有的需要进行第二次评审等。

为了将大家从手动更新工作项的痛苦中解救出来，云效上线了流程自动化功能。让你可以专注于需求评审、专注于代码开发、专注于变更发布。需求状态将会随着你的行为自动进入待评审、开发中、已完成等状态。流程更加智能，统计数据也能更加准确啦！

1. 我们通过规则配置来实现工作项流程自动化。
2. 我们提供了需求开发过程中的几个关键事件：需求发起评审、需求评审通过、工作项关联了变更、工作项关联的变更全部发布上线。
3. 我们提供了一些可自动执行的操作：状态自动变更、弹窗填写属性、属性自动变更。
4. 通过自由组合组合事件和操作，可以配置出想要的自动化流程。
5. 可支持如下场景：
    i. 需求发起评审时，状态自动变为评审中。
    ii. 需求评审通过时，状态自动变为评审通过，并自动指派给开发TL。
    iii. 需求关联了变更时，状态自动变为开发中。
    iv. 需求关联的变更全部发布后，状态自动变为已完成。
6. 此外，我们还提供了两条系统默认的规则：父工作项完成时自动关闭子工作项；子工作项完成时自动关闭父工作项。
7. 规则默认是继承自父项目，如果想要自己配置，打开自主配置开关即可。

入口

点击 **“设置 > 需求配置 > 某类型 > 自定义规则”** 。



功能使用说明

1. 开启项目自主配置

默认继承父项目的配置，开启本项目自主配置后，可自行编辑。

2. 启用系统规则

父工作项完成时自动关闭子工作项，子工作项完成时自动关闭父工作项。

1. 点击上图中的 **“启用”** 即可。
2. 点击规则名称，可查看规则说明。

3. 配置自定义规则

- **示例 1：** “开发中” **，并指派给相关人员。

1. 点击 **“自定义流程规则”** “+添加” ** 按钮。
2. 输入 **“规则名称”** “触发事件” **“工作项新关联变更”** 。
3. **“执行操作”** “状态自动流转” **“自动更新属性”** 并设置指派人。
4. 点击 **“保存”** “返回规则列表” **，新增的规则即显示在规则列表中。

- **示例 2：**“已发布”**。

1. 输入 **“规则名称”** “触发事件” **“工作项关联的变更发布上线”**。
2. **“执行操作”** “状态自动流转”** 并进行设置。
3. 点击 **“保存”** “返回规则列表”**，新增的规则即显示在规则列表中。

- 规则配置完成后，当触发事件发生，该规则会被触发执行，规则执行日志可在工作项操作记录中查看。
- **注意：**“工作流 **添加状态”** “企业设置 **工作项流程与模板** 状态”）。

消息通知

消息通知方式

与“我”相关的数据有更新，默认会发送消息通知给“我”，消息以邮件形式发送。

会发送通知的事件

1. 与项目相关的消息：

- 项目归档：发送给项目所有成员
- 概况页导出子项目列表/关联项目列表：发送给操作者
- 导出里程碑：发送给操作者
- 导出项目成员：发送给操作者

2. 与工作项相关的消息：

- 工作项**状态变更**：发送给工作项相关人员——包括创建者、指派给、抄送人
- 工作项**改变指派给**：发送给工作项相关人员
- 工作项添加**抄送人**：发送给新的抄送人
- 工作项添加**评论**：发送给工作项相关人员
- 工作项列表**导出**：发送给操作人
- **注意**：除导出操作外，所有的消息均不会发送给操作人。

代码管理

代码管理概述

code.aliyun.com

阿里云云效旗下的code.aliyun.com提供源代码托管服务。与国外的代码托管服务相比，它快速稳定。与国内的其他代码托管服务相比，它在底层与云效上从需求到开发的众多功能相集成。

在云效中各入口创建的Git库、Git库组，均实际存储在code.aliyun.com。一些设置也是在code.aliyun.com完成，比如个人全局设置。

与云效一样，code.aliyun.com也是使用阿里云账号登录。这意味着，也可以使用阿里云的RAM（主子账号）。

这里是code.aliyun.com的帮助文档的首页。

下面是个人初始设置相关的：

- 要想在本地命令行使用ssh协议（也就是使用类似git@code.aliyun.com:some-grp/some-git.git这样的Git库地址时）访问服务器端，需要在这里配置SSH Key。详见如何创建和添加SSH Keys和Windows下正确配置客户端以使用SSH协议。
- 要想在本地命令行使用http协议（也就是使用类似https://code.aliyun.com/some-grp/some-git.git这样的Git库地址时）访问服务器端，需要在这里设置用户名和密码。详见相关帮助。**注意**，这里所说的用户名和密码，与访问Web页面所需的阿里云账户和登录密码是不同的概念。

其他常问问题：

- 使用web hooks(英文)

云效的代码服务

云效的代码服务对code.aliyun.com提供的底层服务进行了封装，并引入企业概念。具体来说：

- 管理云效上本企业的代码库和组，或把已有组纳入企业名下，请在云效的代码服务上完成。详见代码库管理。此外，在使用新建一站式方案向导时，也可能自动创建Git组/库，或把已有Git组纳入本企业名下。
- 管理代码库和组的权限，云效的代码服务也提供了比code.aliyun.com更友好的方式。详见权限管理。它背后也是基于code.aliyun.com的权限模型(英文)。
- 此外，从“我的”->“分支”菜单进入，可以查看自己在各代码库中的分支。而在每个RDC项目里，亦可以从“分支”菜单项进入，配置一个到多个关联到该项目的代码库，于是将显示这个（些）代码库的分支列表等信息。以上，详见分支浏览与操作。

附：Git基础

学习Git，推荐阅读开源电子书Pro Git(中文版)。此外，可参阅下述快捷帮助文档：

- 开始在命令行中使用Git

- 基础的命令行命令
- Git基本命令

code.aliyun.com

code.aliyun.com概述

关于Code界面是英文的说明

首先Code作为研发协同代码托管的基础设施，我们在开源软件基础上进行了分布式改造。着重解决了稳定性、性能及安全问题，并经过了阿里这样大体量实战检验。升级替换此前老版本的Code，界面也跟阿里内部版本一样只有英文版。对于喜欢英文原版的程序员这是一项福利，避免被不太准确的中文翻译误导的同时，也表达了对开源原版的敬意。对于喜欢中文界面的程序员，我们建议使用云效的代码服务。另外，云效也提供了一站式研发协同的全部工具链，欢迎大家使用。

阿里云Code基础知识指南

一步步学习如何在命令行及阿里云Code上开启工作之旅。点击查看指南。

个人设置

要想在本地命令行使用ssh协议（也就是使用类似git@code.aliyun.com:some-grp/some-git.git这样的Git库地址时）访问服务器端，需要在这里配置SSH Key。详见如何创建和添加SSH Keys和Windows下正确配置客户端以使用SSH协议。要想在本地命令行使用http协议（也就是使用类似https://code.aliyun.com/some-grp/some-git.git这样的Git库地址时）访问服务器端，需要在这里设置用户名和密码。详见相关帮助。**注意**，这里所说的用户名和密码，与访问Web页面所需的阿里云账户和登录密码是不同的概念。

其他常问问题

使用web hooks(英文)

code容量限制

目前我们提供的单个project的存储上限是2G，一般来讲，Code提供存储2GB代码已经非常大了，只要不存在当云盘滥用的情况，足够个人或团队使用。万一遇到超限，有2种办法，清理误提交的大文件或者通过页面右下角Vone联系我们通过提高限额（目前是免费试用期，可以先提高限额，免费期过后进行收费）。清理Git库中历史上的大文件或含密码文件，参考 https://rtyley.github.io/bfg-repo-cleaner 或者参考 https://github.com/rtyley/bfg-repo-cleaner/issues/65在project页面TAG标签的右边有当前仓库大小值。

code数量限制

每个帐号创建的仓库数不能超过50个。

父子帐号的关系： 所有子帐号创建的库都叠加到父帐号上面； 父帐号创建的库是所有子帐号的集合。

与RDC企业的关系：个人创建的code库数据，与rdc企业没有关系；同一个帐号通过不同的企业创建的库叠加合并计算。

Key has already been taken 怎么解决

“Key has already been taken”这个提示的意思是“这个key已经在Code平台中添加过了，Key是全局唯一的。”请确认你的机器是否是公用的机器，如果是那一般是其他同学已经加过了。如果你确认其它同学没添加过，那就直接重置SSHKey吧。点击查看添加SSHKey方法

基于【自建git库迁移到云code】最佳实践

基于【自建git库迁移到云code】最佳实践

适用场景

场景一：本地自建gitlab仓库，计划采用【云上代码服务】；
场景二：阿里云代码托管服务快速稳定；与云效上从需求到开发的众多功能相集成；
场景三：企业想节约资产&维护人力成本时，可使用【云上代码服务】；

此实践方案优势

1. pipeline流水线与云上代码仓库无缝集成；

2. 节省企业成本：减免搭建、备份私服的机器成本，节约网络带宽和流量。
3. 无需专业人员维护，公司人员更聚焦于业务的发展。

概念介绍

- 开始在命令行中使用Git
- 如何创建和添加SSH Keys
- 基础的命令行命令
- Git基本命令
- Windows下正确配置客户端以使用SSH协议
- 如何使用http的方式clone
- 如何修改代码平台项目权限说明
- 如何恢复分支的代码

其他代码托管迁移云code操作指南

自建gitlab迁移云code管理

第一步，访问https://code.aliyun.com，新建项目；
第二步，新建组；
第三步，导入项目选择git其他仓库的链接，输入本地仓库地址，设置项目权限；点击创建项目；
第四步，完成迁移，成员赋权；

代码托管在其他Git托管站点

假定你的源代码托管在GitHub上bar组的foo库中。现在打算改为托管到https://code.aliyun.com。
第一步，若有必要，通过代码服务页面创建属于当前公司的Git组，比如baz。
第二步，通过代码服务页面在该Git组中创建一个新的Git库foo。
第三步，将原Git库克隆到本地。

1. $ git **clone** --mirror git@github.com:bar/foo.git

第四步，将本地Git库推送到https://code.aliyun.com。

1. $ **cd** foo.git

   $ git push git@code.aliyun.com:baz/foo.git

代码存放在本地

如果已有代码在用户本地，请这样操作：
第一步，若有必要，通过代码服务页面创建属于当前公司的Git组，比如baz。
第二步，通过代码服务页面在该Git组汇总创建一个新的Git库，比如foo。
第三步，把已有代码加入版本控制，并推送到该新建Git库：

$ **cd** existing_folder

4. $ git init
5. $ git remote add origin git@code.aliyun.com:baz/foo.git
6. $ git add .
7. $ git commit -m "import"
8. $ git push -u origin master

代码服务

代码库管理

概念介绍

Git库是指托管在https://code.aliyun.com的Git库，即https://code.aliyun.com中的project。

Git组是若干个上述Git库的集合，即https://code.aliyun.com中的Group。

Git组归属到具体某个企业。于是Git组中的Git库也归属到这个企业。

Git库和组的列表

打开云效代码服务首页，是Git库和Git组两个标签页，分别是Git库和Git组列表。

在这里列出的，是当前用户有权限看到的，且属于当前企业的Git库和Git组。

搜索框用于在Git库或组的列表中搜索。

新建Git库或组

在Git组标签页中，点击“+新建Git组”，输入Git组名和描述后点击“确认”，即可创建一个新的Git组。该Git组属于当前企业。当前用户在该Git组是owner角色。

类似的，在Git库标签页中，点击“+新建Git库”，输入Git组名、Git库名和描述后点击“确认”，即可创建一个新的Git库。

其中，Git组必须是已存在的，属于当前企业的，且当前用户在该Git组中是master或owner角色。

已有Git库或组的权限管理

请点击该Git库或组条目中，“权限管理”按钮，前往权限管理页面。

已有Git库或组的其他操作

请点击该Git库或组条目中，库或组的名称，前往https://code.aliyun.com相应页面。

将已有代码纳入管理

代码已托管在https://code.aliyun.com

假定你的源代码所在Git库名为foo，托管在https://code.aliyun.com上的bar组。由于bar组不属于当前企业，因此在CRP代码服务中看不到该组和该库。在这种情况下，可以这样操作：

第一种方法：把整个代码组归到该企业名下。在代码组列表页面中，点击“关联已有组”，可以把用户自己是owner或master角色，且尚不属于其他企业的代码组，归属到当前企业。

第二种方法：把个别代码库归到该企业名下。第一步，若有必要，通过代码服务页面创建属于当前公司的Git组，比如baz。第二步，登录https://code.aliyun.com，在该Git库的Settings页面下方，进行Transfer project操作，将该库迁移到baz组。由于baz组属于当前公司，该库就属于当前公司，于是在CRP代码服务中就可以看到该库。执行第二步操作时，当前用户必须是bar组的owner角色，以及baz组的master或owner角色。

代码托管在其他Git托管站点

假定你的源代码托管在GitHub上bar组的foo库中。现在打算改为托管到https://code.aliyun.com。

第一步，若有必要，通过代码服务页面创建属于当前公司的Git组，比如baz。

第二步，通过代码服务页面在该Git组中创建一个新的Git库foo。

第三步，将原Git库克隆到本地。

```
$ git clone --mirror git@github.com:bar/foo.git
```

第四步，将本地Git库推送到https://code.aliyun.com。

```
$ cd foo.git
$ git push git@code.aliyun.com:baz/foo.git
```

代码存放在本地

如果已有代码在用户本地，请这样操作：

第一步，若有必要，通过代码服务页面创建属于当前公司的Git组，比如baz。

第二步，通过代码服务页面在该Git组汇总创建一个新的Git库，比如foo。

第三步，把已有代码加入版本控制，并推送到该新建Git库：

```
$ cd existing_folder
$ git init
$ git remote add origin git@code.aliyun.com:baz/foo.git
$ git add .
$ git commit -m "import"
$ git push -u origin master
```

代码库管理员及其权限

代码库管理员当前就是企业管理员对应的人。代码库管理员自动拥有企业所有代码组的owner权限。

权限管理

功能概述

通过权限管理功能，可以查看当前用户自己在特定Git库/组上的权限。当用户在特定Git库/组上有Master或Owner角色权限时，TA还可以查看和修改该Git库/组的其他成员的权限。

概念介绍

Git库即https://code.aliyun.com/中的Project。在Git库这级，可以把一个人设为以下四种角色之一：

- Guest：能看概况，可留言，但看不到源代码。
- Reporter：能看到源代码。
- Developer：可读写，但不能推送(push)改动到受保护分支(protected branch)。
- Master：可读写，甚至推送(push)改动到受保护分支(protected branch)。有一些管理权限，比如管理成员，但不能删除Git库、不能调整Visibility Level等。

Git组即https://code.aliyun.com/中的Group，可以包含若干个Git库。在Git组这级上设的权限，对组内的库都有效。如果一个人既在组上具有角色，又在其中某个库上具有角色，那么在该库的实际权限，取两者中权限高的。

具体说来，GitLab组这级，可以把一个人设为以下五种角色之一：

- Guest：能看各库概况，可留言，但看不到源代码。
- Reporter：能看到各库源代码。
- Developer：各库可读写，但不能推送(push)改动到受保护分支(protected branch)。
- Master：各库可读写，甚至推送(push)改动到受保护分支(protected branch)。有一些管理权限，比如在组中创建新Git库、管理Git库的成员，但不能管理GitLab组的成员、不能删除组或库，不能调整库的Visibility Level等。
- Owner：拥有GitLab组及其所属Git库的所有读写和管理权限。

以上是大致介绍，详细介绍见https://code.aliyun.com的相关帮助文档。

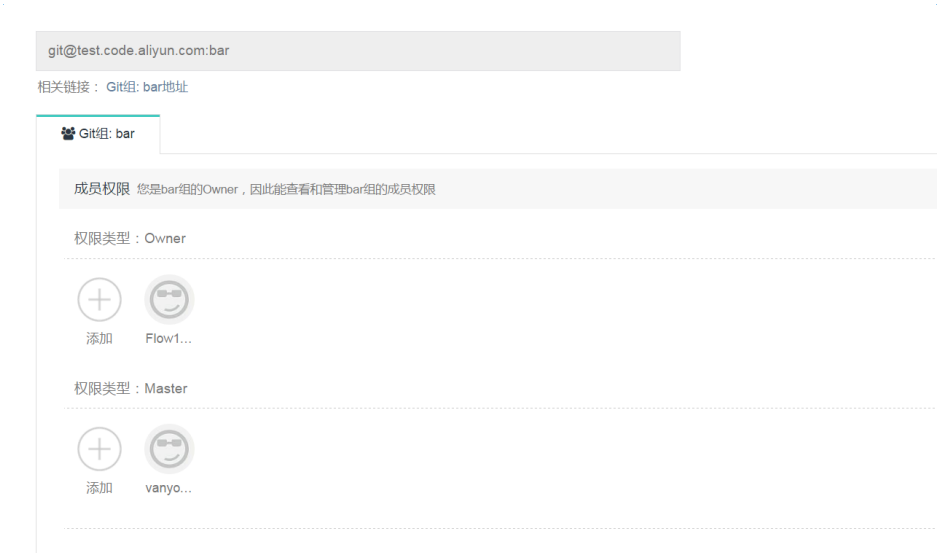
前往特定Git库/组的权限管理页面

云效代码服务的首页，显示当前用户所在的Git库/组的列表。每行显示当前用户在该Git库/组的角色，以及“权限”按钮。点击该按钮，进入该Git库/组的权限管理页面。

特定Git组的权限管理页面

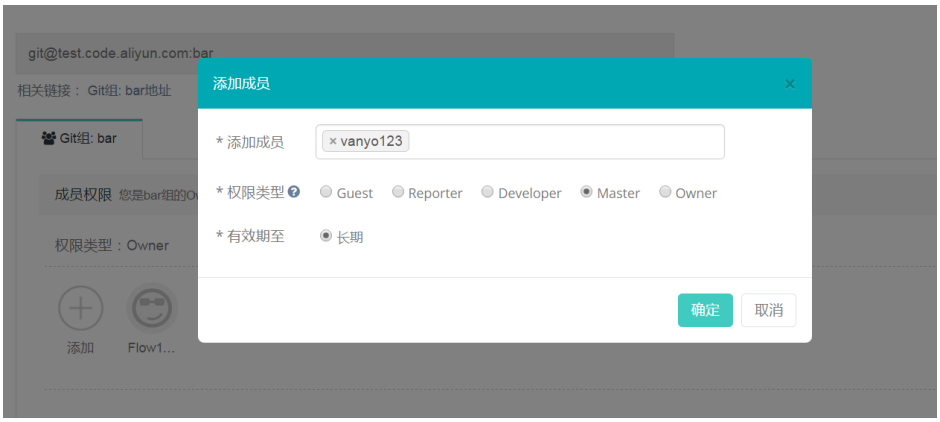
若当前用户在该Git组上的角色为Guest、Reporter或Developer，将显示用户在该Git组的角色。用户可以操作将自己的权限降低：从Reporter到Guest，或者从Developer到Reporter或Guest。

若当前用户在该Git组上的角色为Master或Owner，那么他不仅能查看和操作自己的权限，还可以查看和操作该组其他成员的权限，或添加新成员。当前所有成员按照角色分组显示：

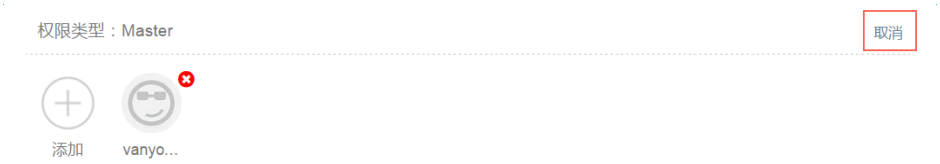


点击成员头像，可查看和修改已有成员权限：

点击左侧“+”号，可添加新成员：



点击右侧“删除成员”，进而点击特定成员头像右上方“-”号，可删除该成员：



特定Git库的权限管理页面

该页面包含两个标签页：该Git库的权限管理，和该Git库所在组的权限管理。后者页面功能类似于上一小节。下面重点介绍前者，即该Git库的权限管理部分。

页面仅显示本人的角色权限信息，还是显示该库所有成员的角色权限信息，取决于当前用户在该Git库的实际权限。即，TA在该Git库的角色（若有），与TA在该Git库所在Git组的角色（若有）中，权限高的。

如果两个角色中，最高权限是Guest或Reporter或Developer，则TA只能看到本人在该Git库的角色，同时可以将自己的权限降级。

如果两个角色中，最高权限是Master或Owner，则TA能看到该Git库上所有成员，并可添加、修改、删除成员权限。

企业的代码管理员

企业的代码管理员默认就是企业的管理员（修改功能即将上线）。本企业名下每当新增Git组时，代码管理员将自动获得它的Owner成员权限。

持续交付流水线（新版）

快速开始

本文中将介绍云效新版流水线的基本概念以及快速入门，云效新版流水线在云效原有能力的基础上对底层调度以及上层交互进行了全面的优化，以更开放的形式帮助用户可以快速创建现代化的持续交付流程。

基本概念

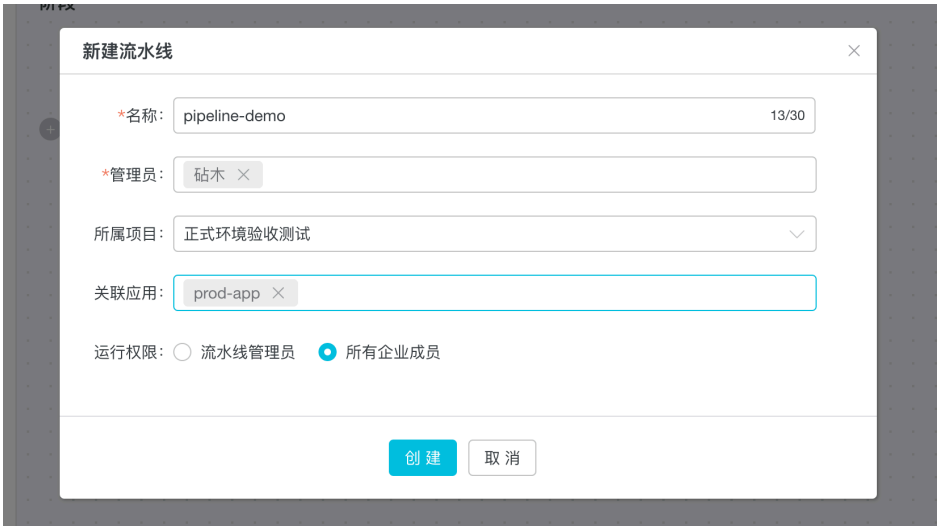
- 输入源：持续交付的原始物料，如Git仓库。目前云效支持阿里云Code，码云以及用户自建的Git作为流水线输入源；
- 阶段：在流水线中需要按顺序执行的一组任务的集合，一个阶段可以是手动运行也可以是自动运行的。**阶段之间可以是串行也可以是并行的；**
- 任务：在阶段中具体需要完成的动作，目前任务主要包含两类，一类是与工作区相关的（AgentJob）以及与工作区无关的（Agentless Job），工作区相关任务会在固定的工作目录中执行用户定义的任务。而工作区无关任务主要是调用云效提供的服务，如测试服务，发布服务，代码规约扫描以及安全扫描等；

快速入门

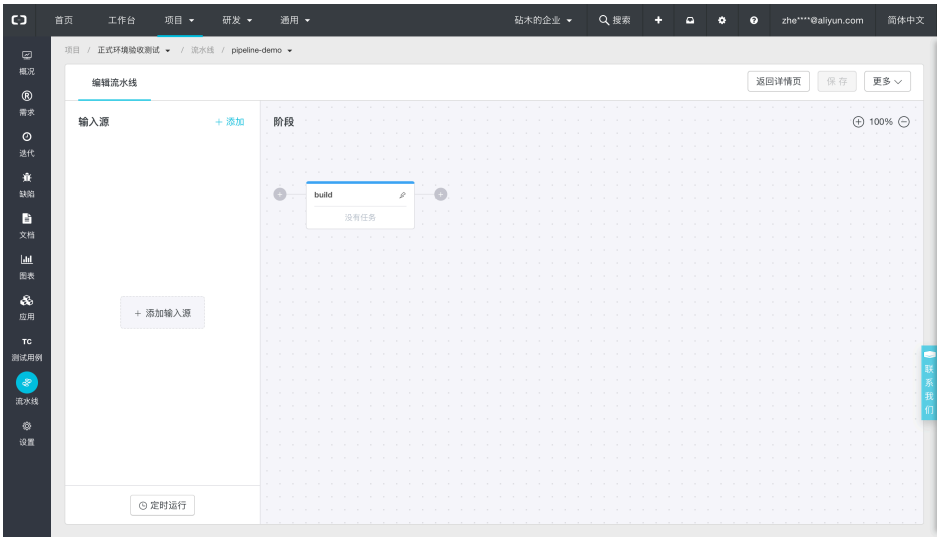
用户可以通过主菜单“研发-流水线”快速进入到流水线列表

创建流水线

点击流水线列表右上角的“新建流水线”按钮，开始创建新版流水线，点击后，设置流水线基本信息，如下所示：

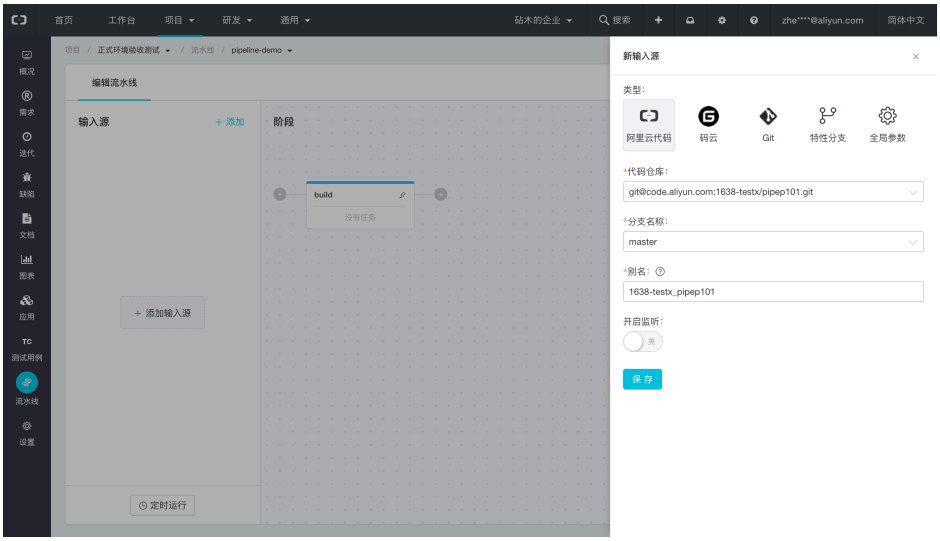


流水线可以关联到项目以及应用，以方便后续的聚合管理。默认情况下流水线是可以被企业下所有成员触发并运行也可以只允许流水线管理员触发。点击创建即可进入到流水线的编排界面。如下所示：



添加输入源

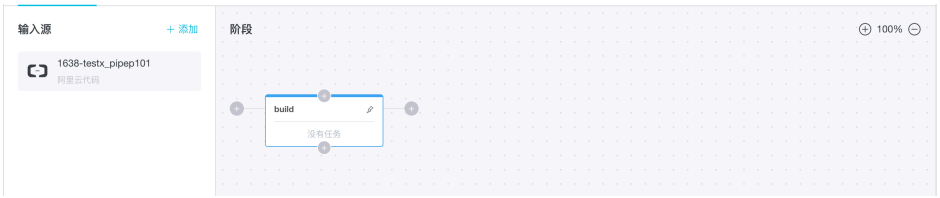
点击输入源的“+添加输入源”按钮，可以为流水线添加输入源，如下所示：



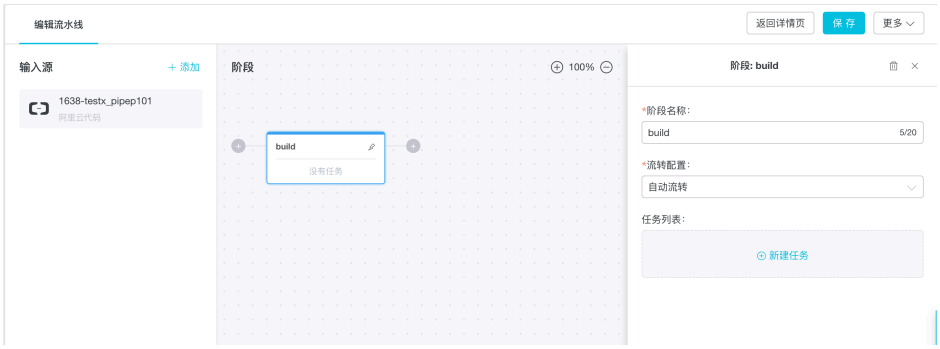
在示例中以阿里云Code为例，添加了一个应用的源码，如果需要代码提交自动触发流水线，则需要点击打开“监听”开关，当该代码库有任何代码变更时都将自动触发流水线的运行。点击保存完成输入源添加。

添加阶段

在阶段面板中，默认会包含一个空的build阶段。将鼠标移动到build阶段卡片上后如下所示：



通过上下方面的“+”按钮，用户可以添加一个与build并行执行的阶段。通过左右的“+”可以添加一个与build串行执行的阶段。点击阶段卡片，进入阶段详情配置，如下所示：

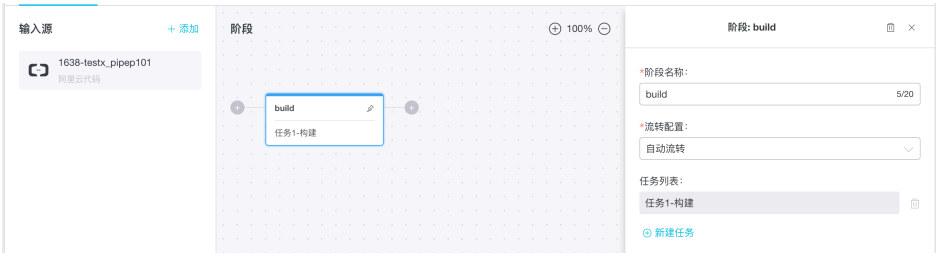


在侧拉面板中，用户可以定义当前阶段的名称以及流转方式，如果是自动流水线，流水线触发后该阶段会自动运行，如果是手动流转，该阶段则需要人工确认后运行。在任务列表中，用户可以为该阶段添加一组串行执行的任务。

添加任务

点击阶段任务列表的新建任务按钮，如下所示：

如上图所示，对于使用了release文件定义源码构建的用户可以添加“应用构建”，对于未使用release文件的用户可以添加“构建”任务来自定义构建过程，“构建任务”是一个工作区相关任务，用户可以在其中添加任意命令。

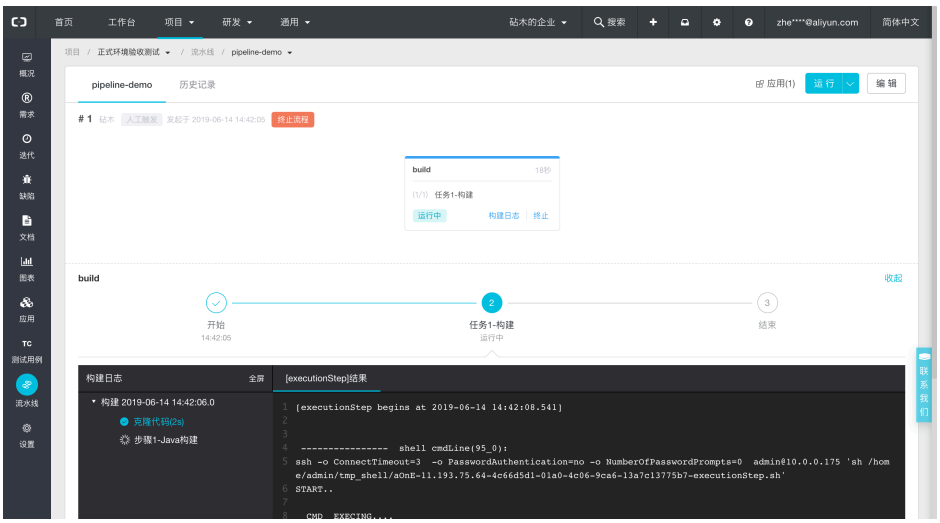


添加“构建”任务后，并点击该任务，进入到任务详情配置，如下所示：

在上图中，构建任务中，我们添加了一个Java构建的步骤。 点击保存按钮，完成流水线定义并返回详情页。

运行流水线

点击详情页右上角的运行按钮，触发流水线运行，即可开始你的持续交付之旅。



自由模式

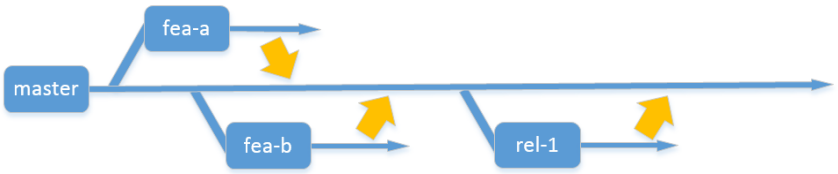
自由模式

自由模式，顾名思义，用户可以使用任何分支（包括master）进行打包、发布等操作。

在自由模式下，常见用master分支这一条分支来承载开发、集成和发布，这被称作主干开发方式。使用这种方式，只有在特定情况下，才会使用其他的分支。包括：

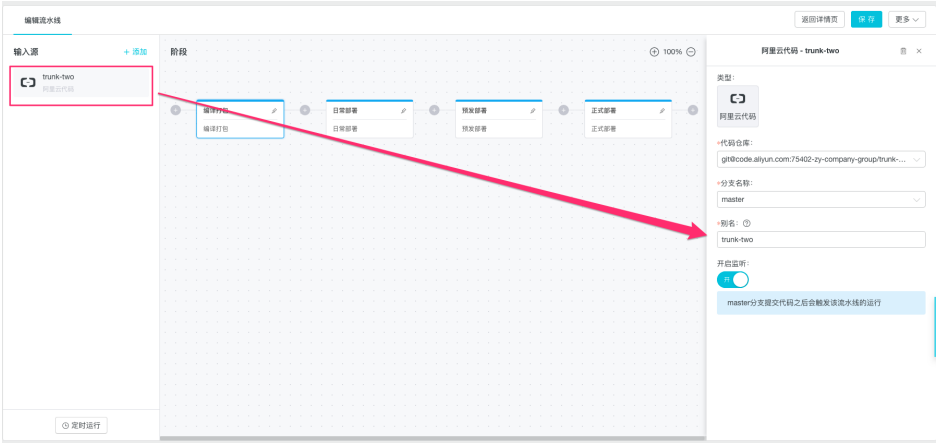
- 确有必要时，拉出feature分支开发特定feature，开发完成并验证后合并回master。
- 确有必要时，拉出release分支，发布特定版本。随后合并回master。

如下所示，是自由模式下分支管理的典型流程：

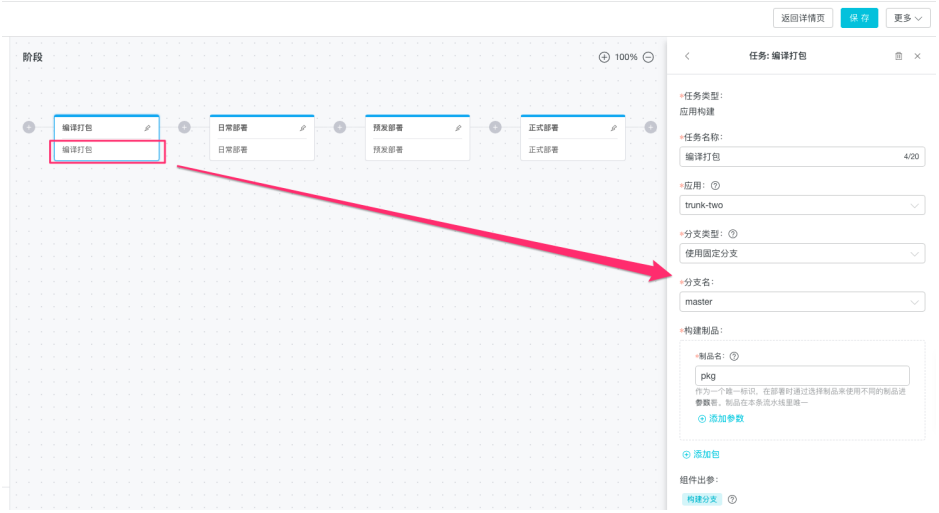


在流水线中集成

自由模式时，在流水线上，通常默认配置为，master分支变化时自动触发流水线运行，取master分支做构建，并随后部署和发布。流水线默认监听配置如下，可以根据实际发布使用修改监听分支或者移除监听。如下所示：



此外如果期望使用非master分支进行构建发布，则需要修改编译打包阶段的分支设置：

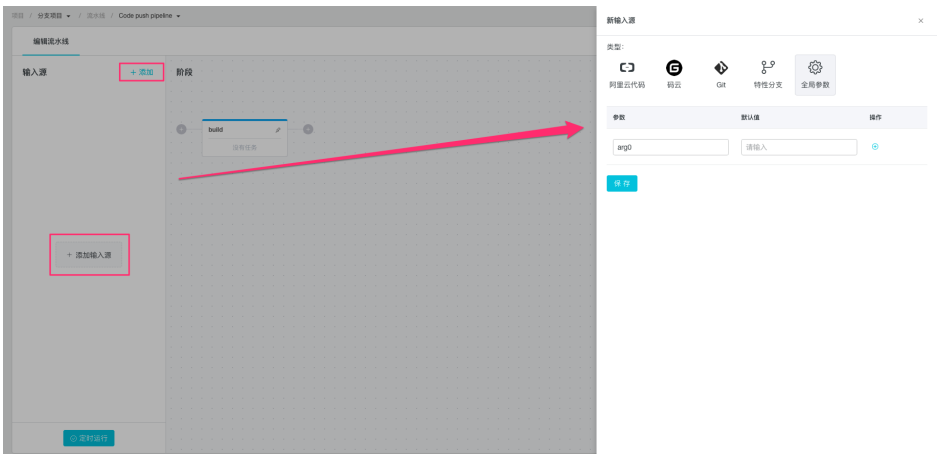


全局参数

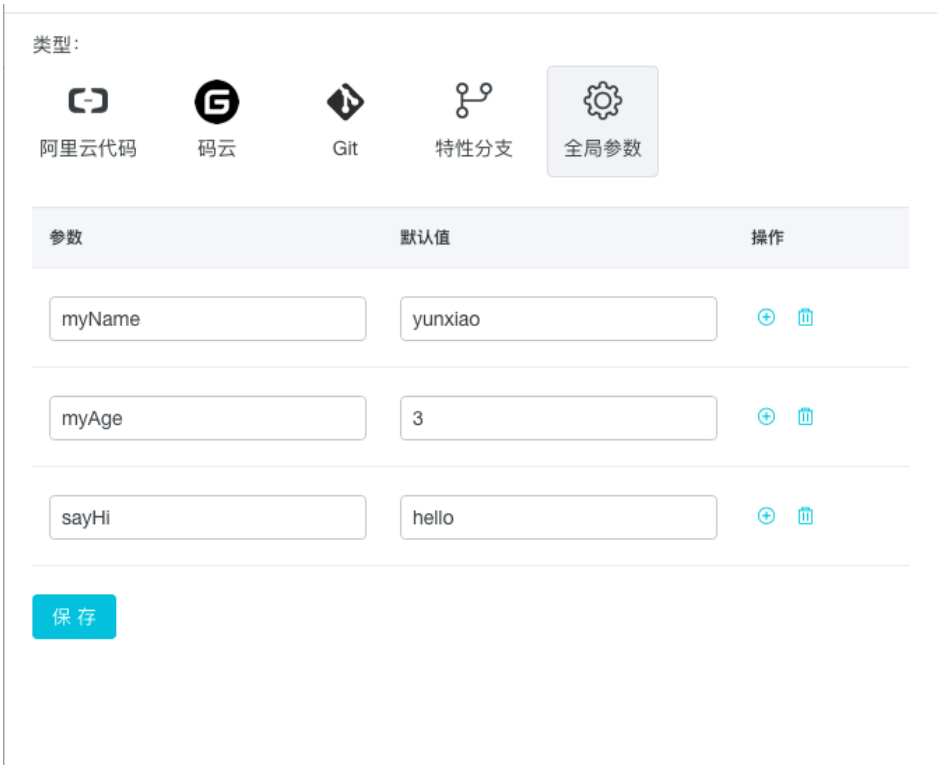
流水线提供全局参数来设定一些变量，在配置流水线时，可以通过引用全局变量的方式支持一些需要参数变化的场景。

参数设定

全局参数的配置入口在流水线编辑页面的输入源配置中，如下所示：



用户可以添加任意参数，并设置默认值：



参数使用

全局变量设定后，可以在配置具体任务时通过特殊变量符号进行引用。

- 使用举例1:

如应用任务，选择分支时，可以选取使用流水线参数，通过 ${参数名}方式进行引用，这样每次流水线执行时，会读取全局变量中设定的值来进行构建。如下所示：

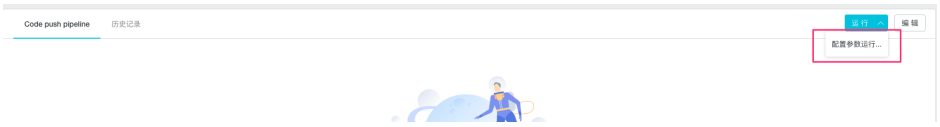
- 使用举例2：

在自定义脚本中，可以通过 ${参数名}的方式引用全局变量。 如下所示：

运行时指定参数

全局变量设定后，正常触发时都会使用默认值，如果在触发时想改变全局变量的值，可以在手工触发时（请参

考触发策略部分文档说明)配置运行参数:



设置全局变量的值，然后点击确认，本次执行将使用更改后的变量值进行执行。更改的变量仅针对本次执行有效，并不会影响原始默认值。

触发策略

流水线的触发策略主要支持以下常用策略，手工触发，监听代码提交触发，定时触发。根据不同的使用场景，你可以选择适合的方式来进行流水线执行的控制。

手动触发

直接在页面上进行流水线的运行，触发入口有以下几处：

- 流水线列表页：您可以在操作列点击运行来触发流水线运行。除了需要有对应流水线运行权限外，包含特性分支的流水线是不能在列表页面直接执行的，包含特性分支的流水线通过image.png符号表明，这种类型的流水线只能在流水线详情页面中操作执行。

- 流水线详情页：您可以点击运行按钮来触发流水线执行

- 手动参数触发：运行下拉有配置参数运行的方式，配合全局变量设置和选择运行阶段的方式来个性化执行流水线。

代码提交触发

在代码提交时触发流水线的执行，开启代码提交触发流水线，需要你在流水线中增加输入源配置，目前在阿里云代码和Git输入源下，提供了开启监听的功能，两者的配置方式略有差别。对于阿里云代码输入源，直接开启下方“开启监听”的开关即可。

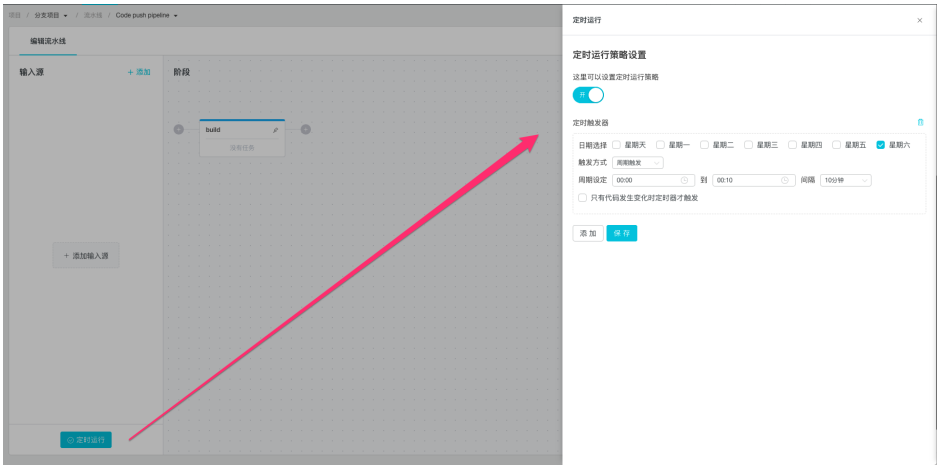
- 阿里云Code：开启自动监听

- 自定义Git: Webhook

通过上述方式配置好代码监听或者Webhook后，在相应的代码地址和分支上提交代码后就可以触发流水线的运行了。

定时触发

您可以通过设置定时配置来周期性的触发流水线的执行。在编辑流水线时，可以点击定时运行，然后配置定时配置。此外，定时触发还支持代码变化检测，仅在代码发生变化时进行触发，定时触发的最小间隔时间为10分钟。



其它

上述的触发策略中，除了手工触发可以个性化设置全局变量和选择运行阶段外，其余方式会运行所有阶段，全局变量也将全部使用默认值。

通过钉钉机器人发送群消息

用户可以在流水线中配置钉钉群通知插件，为钉钉群自动发送流水线运行信息。

在钉钉群里添加钉钉机器人

在钉钉群中创建钉钉机器人，请参阅以下文档：

钉钉机器人配置详细文档

请注意钉钉机器人的安全设置选择自定义关键词，并且关键词为流水线。如下所示。

图1.在钉钉群中添加机器人

在流水线中配置钉钉群通知插件

复制钉钉机器人的 webhook 地址，编辑流水线的任务，在任务插件中选择钉钉群通知，并填写 webhook 地址和运行时机。如下图所示。

图2， 在流水线中配置使用钉钉群通知

构建

云效流水线通过支持不同的构建组件，对各种语言提供了构建打包能力，以便随后流水线上的部署任务进行部署。

构建能力

云效流水线提供了一系列的基础构建环境环境为用户提供了开箱即用的构建能力。

构建语言支持

语言	支持版本
Java	jdk1.6,jdk1.7,jdk1.8,jdk1.9 (Maven3.5, Gradle4.1)
Nodejs	6.11,7.10,8.13,9.11,10.15,11.15,12.2
Python	2.7,3.5,3.6,3.7
Ruby	-
Php	5.6,7.0,7.1,7.2
Go	1.8,1.9,1.11,1.12

对应的构建能力都是通过基础构建机支持， 基础环境已经支持了上述构建能力，如上述能力无法支持您的构建需求，请您联系我们的客服以获取更多的支持。

构建依赖

Maven仓库

使用阿里云公共代理库和云效私有仓库

云效会默认使用构建代码库的**根目录**下Maven的**settings.xml**文件中配置的依赖库。

如果用户项目代码库的**根目录没有Mavensettings.xml**文件，那么云效构建时会为用户自动生成一个settings.xml文件，并且配置连接阿里云公共代理仓库maven.aliyun.com和云效提供的企业私有仓库。 用户通过云效构建时，会默认使这两个仓库的二方库作为构建依赖。

使用企业自建私有的Maven仓库

如果需要指定连接企业自建私有的Maven仓库，需要在构建代码库的根目录下，添加 settings.xml 文件，并在settings.xml指定依赖的Maven仓库。

（1）源码仓库中添加settings.xml比如我要构建的项目为java_demo， 则可以把settings.xml放置到java_demo的根目录下：

（2）源码仓库中添加settings.xml

```
 <mirror>
<id>alimaven</id>
<name>aliyun maven</name
<url>http://maven.aliyun.com/nexus/content/groups/public/</url
```

```
<mirrorOf>central</mirrorOf>
</mirror>
```

Npm 国内镜像源

默认使用NPM依赖下载

npm 是 Node.js 默认依赖管理工具。云效Nodejs构建，默认通过npm进行依赖下载， npm的使用请参考官方文档

使用CNPM加速依赖下载

npm安装插件是从国外服务器下载，受网络影响大，可能出现异常，淘宝 NPM 镜像提供了国内镜像源，可以加速依赖的下载。云效的构建环境默认提供 cnpm 工具，如需[淘宝 NPM 镜像]，可以将构建命令中的 npm 命令替换为 cnpm 即可从淘宝NPM镜像源拉取依赖。cnpm跟npm用法完全一致，只是在执行命令时将npm改为cnpm即可。

Go 语言提供镜像源代理

阿里云提供了官方的Go Module代理仓库服务，镜像源地址：https://mirrors.aliyun.com/goproxy/云效的构建默认会通过该该代理来避免DNS污染导致的模块拉取缓慢或失败的问题，加速你的构建。

Gradle构建

构建基础镜像中内置了Gradle4.1，如果需要其它的版本，建议使用Gradle Wrapper。使用方式：

1. 在代码库中初始化Gradle Wrapper：
   https://docs.gradle.org/current/userguide/gradle_wrapper.html#sec:adding_wrapper

   修改代码库中的gradle/wrapper/gradle-wrapper.properties文件，修改其中的distributionUrl为云效提供的镜像地址，如：

```
distributionBase=GRADLE_USER_HOME
distributionPath=wrapper/dists
distributionUrl=https\://rdc-public-software.oss-cn-hangzhou.aliyuncs.com/gradle/gradle-5.6.4-bin.zip
zipStoreBase=GRADLE_USER_HOME
zipStorePath=wrapper/dists
```

3. 使用./gradlew替换gradle命令即可。

云效镜像中支持目前支持的gradle版本：4.4.1, 4.5.1, 4.6, 4.7, 4.8.1, 4.9, 4.10.3, 5.3.1, 5,4.1, 5.5.1, 5.6.4

持续交付流水线（老版）

概述

基本概念与原理

本文档仅适用于2019年6月13日之前创建并未迁移到新版持续交付流水线的企业，使用新版持续交付的企业（2019年6月13日后创建的企业都已经使用新版本），请参看文档

欢迎您了解和使用云效的持续交付相关功能！

为了帮助您快速理解和掌握使用方法，本文概要介绍一下相关基本概念和原理。理解了它们，就摸清了云效持续交付的脉络，学习具体内容就会容易很多。

项目

项目是一个“工作场所”。一伙人（或者一个人）为了一个特定的场景（比如开发一个应用/产品），在这个“工作场所”一起办公，方便地使用这个“工作场所”所关联的各种工具和各类数据。这不仅可以包括这个项目相关的需求、任务、缺陷这些工作项并进行迭代排期，也可以包括这个项目对应的代码库、应用、流水线等（详见下文）。典型的情况，一个项目对应一个代码库及其相应的一个应用，但也可能多个代码库和应用。

流水线

流水线的本质是研发-交付的流程，它把流程中的不同阶段和任务串接在一起，并且（可以设置为）自动化地一步一步地执行。简单的例子，手工触发，构建并部署到一个特定的环境，是一条基本的流水线。复杂的例子，源代码提交自动触发，通过各个环节和阶段的构建、部署、各种检测工作，直到上线，是一条完整的端到端的流水线。详情请从流水线概述开始阅读。

代码库

云效文档中提到代码库，是指存放源代码等内容的Git库。代码库托管在阿里云的code.aliyun.com。要想在本地查看、编写和提交源代码，请首先在您本地机器上安装Git，并设置ssh key以便连接到code.aliyun.com服务器端。更多内容请参考代码服务概述。

构建

构建指根据代码库中的源代码编译构建打包，它是流水线上的一个任务。云效根据Git库中源代码根目录下的<应用名称>.release文件构建打包，以便部署并运行。<应用名称>.release文件用键-值对儿的形式定义了如何

把源代码构建打包。详情请从构建概述开始阅读。

应用

从源代码的角度看，源代码存放在一个个代码库里。而从部署运行的角度看，一个个Web应用被部署并运行起来，相互配合，实现Web网站的功能。每个Web应用，通常对应了根据一个代码库里的源代码构建生成的包，以及它运行时相关的基础软件和环境的设置等等。它是部署运维的一个(最小)单元。这是应用这个概念的由来。

环境

每个Web应用，在集成测试的环境（在云效中通常称作日常环境）、预发的环境（称作预发环境）、对外提供服务的环境（称作正式环境）等不同的环境里运行。每个环境中，该应用运行在若干台机器（虚机/容器）上。部署时，可能分期分批。每台机器的部署，有特定的方法和步骤。这些都是定义在这个应用的特定环境上。

发布与部署

在云效中，发布是指一次部署以及相关的一些操作，它是流水线上的一个任务。如前文所述，发布和部署的配置，是在应用的环境上。应用、环境、发布与部署的详情，请从应用部署概述开始阅读。

上手步骤

本文档仅适用于2019年6月13日之前创建并未迁移到新版持续交付流水线的企业，使用新版持续交付的企业（2019年6月13日前创建的企业都已经使用新版本），请参看文档

在不同的使用场景里，配置和使用云效进行软件开发和发布的方法略有不同，但它们遵循基本相同的大致步骤：

新建和配置企业

新建和配置企业的一般方法见企业信息和成员管理。

一些场景下，还需要进一步做一些的企业级设置。比如，

- 如果使用脚本（而不是通过EDAS服务或容器服务）来部署Web应用，需要在“企业设置”->“机器管理”中配置机器资源以供各应用使用。详见部属配置：通过脚本部署中的相关描述。
- 如果使用容器服务来部署Web应用，需要在“企业设置”->“容器服务账号”中设置。详见部属配置：通过容器服务部署中的相关描述。

具体有哪些场景，需要做什么企业设置，请到下文这里查看。

使用向导创建一站式解决方案

向导可以帮助你创建(或关联)包括项目、代码库、应用、流水线在内的一站式解决方案。于是，从需求到发布上线的管理和操作，都可以在此一站式完成。

在使用向导的过程中，选择不同的选项，产生适合不同场景的配置。比如，是Web应用还是无线应用，是通过脚本部署还是通过EDAS或容器服务部署，等等。

一站式方案的大部分配置，可以在使用向导创建的过程中自动完成。

以上，详见新建向导概述。

更多配置

在不同场景下，还有一些配置信息，需要在使用一站式创建向导后，继续设置或调整。随着时间的推移，在试用过程中也可能想增加或调整一些配置。

一般而言，使用一站式创建向导后，依如下步骤检查/调整配置：

1. 先配置好构建部分（可能包括Docker image的生成），详见构建概述。运行默认流水线直到此步骤能成功。
2. 然后配置日常环境的部署，详见部署配置概述。运行默认流水线直到此步骤能成功。
3. 配置预发（如果没有可删除流水线上对应步骤）、正式环境的部署，运行默认流水线直到此步骤能成功。
4. 考虑配置流水线，增加、修改、删除流水线上的节点。比如增加自动检查和测试等任务。详见流水线概述。

日常使用

日常使用云效做集成和发布，主要是在流水线上完成。详见流水线概述。

另外，当“开发模式”选择使用“分支模式”（而不是“自由模式”或“Git Flow模式”）时，分支上的操作显得比较重要。因为每一条feature分支在开发完成后，需要执行“提交待发布”操作。这个操作，可以在该分支详情页点击完成，也可以在分支列表页面直接点击完成。详见分支模式和分支模式下的流水线。

不同场景下的上手方法

以下是一些典型场景的上手方法：

- 典型场景：通过脚本部署Web应用
- 典型场景：通过EDAS部署Web应用
- 典型场景：通过容器服务部署Web应用
- 典型场景：无线应用

典型场景：通过脚本部署Web应用

当在创建向导中选择了创建Web应用，并选择了通过自定义脚本部署时，有如下配置请检查修改或添加：

构建配置

源代码根目录下的<应用名称>.release文件是构建的配置。一般来说，它已经自动生成好，特定情况需要进一步补充。请参考Web应用构建配置。构建配置的更多内容请从构建配置概述开始了解。

部署配置

需要把机器资源添加到云效上本企业名下。机器资源通常是阿里云ECS，请先行购买。此外也支持使用其他途径的机器资源。

随后，在具体应用的具体环境中，配置本企业名下的机器。

除了配置机器资源，还需要配置/调整部署的批次，以及一台机器上部署的过程，比如使用的启动脚本和停止脚本。

根据创建向导中的选项，日常环境可能已为您关联了临时免费使用的机器。请替换。并继续配置预发环境和正式环境。

以上，详见部署配置：通过脚本部署。部署配置的更多内容请从部署配置概述开始了解。

其他内容

请参考上手步骤。

典型场景：通过EDAS部署Web应用

当在创建向导中选择了创建Web应用，并选择了通过EDAS部署时，有如下配置请检查修改或添加：

构建配置

源代码根目录下的<应用名称>.release文件是构建的配置。一般来说，它已经自动生成好，特定情况需要进一步补充。具体请参考使用EDAS部署时的构建配置。构建配置的更多内容请从构建配置概述开始了解。

部署配置

需要在EDAS系统中，为该应用的每个(打算使用EDAS的)环境，分别创建和配置EDAS应用。

随后，在云效中，把该应用的每个(打算使用EDAS的)环境，配置为关联到相应EDAS应用。

以上，详见部署配置：通过EDAS部署。部署配置的更多内容请从部署配置概述开始了解。

其他内容

请参考上手步骤。

典型场景：通过容器服务部署Web应用

当在创建向导中选择了创建Web应用，并选择了通过容器服务部署时，有如下配置请检查修改或添加：

构建配置

源代码根目录下的<应用名称>.release文件是构建的配置。一般来说，它已经自动生成好，特定情况需要进一步补充。注意其中不仅要构建产生包，还要进一步生成Docker image。请参考Docker镜像构建配置。构建配置的更多内容请从[构建配置概述]https://help.aliyun.com/document_detail/59292.html)开始了解。

部署配置

你需要在阿里云容器服务中已经有集群，应用和服务。随后在云效上，导入集群证书。

在具体应用的具体环境页面中，点击部署配置，选择该环境要发到容器中对应的集群、应用、服务。

以上，详见部署配置：通过容器服务部署。部署配置的更多内容请从部署配置概述开始了解。

其他内容

请参考上手步骤。

成员权限

应用成员及权限说明如下：

#	应用owner	SCM	开发负责人	PE
基本信息-编辑	√	√		√
变更内容-编辑	√	√	√	√
流程-查看	√	√	√	
流程-编辑	√	√	√	
角色-编辑	√			√
中间版本-编辑	√	√	√	√

新建向导

新建向导概述

在云效中，有多种途径来新建从需求到发布上线的一站式的研发协同解决方案。

其中，对于刚开始接触云效用户，首页的“快速开始”是开始了解和使用云效的最佳选择。详见新建：快速开始以及云效快速入门中的介绍。

首页的“更多选择”向导则提供了更多选择。这包括，可以选择使用一个已有的代码库而非总是新建一个；可以把Wweb应用打包为Docker镜像并使用阿里云容器服务部署；可以通过EDAS部署Web应用；可以构建发布无线应用而不仅仅是Web应用，等等。详见新建：更多选择

除了首页这两个主要入口，还有其他一些入口，起不同作用。详见新建：其他入口.

新建：快速开始

从吊顶中点击“首页”，进入首页后，点击“快速开始”，依步骤配置。

其中，第二步“配置代码库”中，开发模式请选择“自由模式”，初次接触云效，这是最容易上手的选择，因为流程最简单。“构建并部署到临时演示环境”也请确保勾选。

点击完成，将自动进行如下工作：

新建一个项目。

新建一个代码库并关联到这个项目。向该代码库中填充了样例代码，以及构建配置。

新建一个应用，与这个代码库对应。应用也关联到这个项目。完成这个应用及其环境的基本配置。其中，免费临时借给用户一台机器，并配置到了日常环境，用于演示。

新建一条端到端的流水线。运行这条流水线，构建并部署到日常环境。

此时，可以到界面给出的演示地址，查看应用运行的页面效果。也可以进入项目或流水线，查看配置和运行情况。

当前仅为日常环境配置了供演示用的临时机器。要想配置平时用的机器，以及为预发、正式环境配置机器，请参考完善配置：通过脚本部署Web应用。

更多内容，请从完善配置：概述读起。

新建：自定义配置

从吊顶中点击“首页”，进入首页后，“快速开始”按钮的右侧，有“自定义配置”按钮。它也是一个新建一站式解决方案的向导。与“快速开始”向导的区别是，有更多选项，对应更多场景，可以按需进行选择。这包括：

- 不仅可以新建项目，也可以在已有的项目中追加代码库和应用。
- 支持自由模式及分支模式开发。详细说明
- 在自由模式下支持Aliyun及码云两种代码库。在使用码云做为代码库时，需进行授权操作。授权后即可同步码云的数据。

- 不仅可以新建Web应用，还可以新建无线客户端应用（即将上线）。
- 可以选择更多编程语言、JDK版本、Maven版本、源代码初始化模板。
- 不仅支持自定义脚本部署到机器，也支持通过EDAS部署，以及构建打包为Docker image后通过容器服务部署。（即将上线）

其中的一些选项组合，继续提供了“构建并部署到临时演示环境”这个能力，可以勾选。

对于一些选项组合，在完成这个向导后，可能还需要进一步的配置。具体内容详见完善配置：概述。

新建：其他入口

点击吊顶上的“首页”，进入首页后，“快速开始”和“更多选择”这两个按钮，是新建一站式研发协同方案的主入口，根据向导，配置好项目-代码-应用-流水线。还有一些入口，本文一并介绍：

在应用列表页点击“注册应用”

点击吊顶的“我的”，随后点击左侧菜单项“应用”，进入应用列表页。其中有“注册应用”按钮。

当前，点击“注册应用”按钮，将进入新建应用向导，它将新建或关联代码库，新建流水线，但不会关联到项目。

我们即将改进它的功能，可以新建项目或关联已有项目。

在一个已有项目中新建或关联应用/代码库/流水线

在具体项目的概述页，可以看到名为“配置代码库”的一个小卡片。点击它，将进入类似于“更多选择”的向导。唯一的区别是，不必选择项目，必然是在本项目中新建或关联代码库、应用、流水线。

类似的，在具体项目的应用列表中点击“注册应用”，也是在本项目中新建或关联代码库、应用、流水线。

而点击具体项目的分支菜单项进入后，点击“配置代码库”（若当前该项目还没有关联代码库）或点击tab页标签旁的“+”号，，也是在本项目中新建或关联代码库、应用、流水线。

仅新建代码库

点击吊顶的“我的”，随后点击左侧菜单项“代码”，进入代码库列表页。在这里点击“+新建代码库”，当前仅会新建一个空的代码库。

仅新建项目/项目集

以下入口都只会新建项目/项目集，不会新建或关联代码库/应用/流水线：

点击吊顶上的“项目”，进入项目列表页，点击“新建项目”或“新建项目集”。

点击吊顶上的“+”号，进而选择“新建项目”或“新建项目及”。

点击吊顶上的“首页”，在下部“解决方案”区域，选择“启动Scrum”项目或“大型项目集管理

”。

流水线

流水线概述

流水线的本质是研发-交付的流程，它把流程中的不同阶段和任务串接在一起，并且（可以设置为）自动化地一步一步地执行。

简单的例子，手工触发，构建并部署到一个特定的环境，是一条基本的流水线。

复杂的例子，源代码提交自动触发，通过各个环节和阶段的构建、部署、各种检测工作，直到上线，是一条完整的端到端的流水线。

学习使用流水线，请首先阅读流水线的运行，这里讲解了流水线的概念原理，以及如何使用。

通过一站式方案的新建向导创建的项目，通常已经自动生成了一条流水线。可以查看它的配置，修改补充，或者新建一条流水线。流水线的配置方法详见这里。

流水线上不同类型的流程任务，是通过不同的插件实现的。当前已有：

- 构建：构建打包，供部署使用。
- 部署：把构建成果部署到服务器运行。
- 人工卡点：需要人工判断是否OK的任务。可以用来作为流水线上人工测试、安全审核等流程卡点。
- 单元测试：自动运行一行命令，看是否能成功。

我们将研发更多插件，以支持更多类型的任务。同时，也将考虑开放插件接口，让用户可以自定义插件，放入流水线中。

以上，是对自定义流水线的介绍。目前当开发模式选用分支模式时，流水线还不是可以完全自定义的。相应情况请参考这里。

流水线的运行

本文介绍流水线的使用方法。关于云效流水线服务的概要介绍，请参考这里。关于流水线的配置，请参考这里

。

入口

当进入一个项目后，左侧菜单栏的“流水线”菜单项，是流水线的入口，点击打开流水线列表页，可选择进入具体流水线页面。

而如果左侧菜单栏没有出现“流水线”菜单项，请前往“设置”->“服务”为该项目配置流水线服务。

阶段与任务

图中，红色框“持续集成”是一个阶段，跟“日常环境测试”、“预发环境测试”、“正式发布”几个阶段串接在一起，形成整个流水线。

而每个阶段，可以包含若干任务。比如，“持续集成”阶段，包括黄色框“构建”这个任务，以及“单元测试”这个任务。两个任务串接在一起，是“持续集成”阶段的全部内容。

流水线的启动运行

可以把一条流水线设置为代码提交后自动触发、定时触发或手动触发。即使设置为前两种方式之一，也仍然可以手动触发。手动触发的方法是点击右上角的“运行流水线”按钮。

阶段的启动运行：设置为自动触发时

各阶段之间，可以设置为自动触发或手动触发。下面分别介绍。

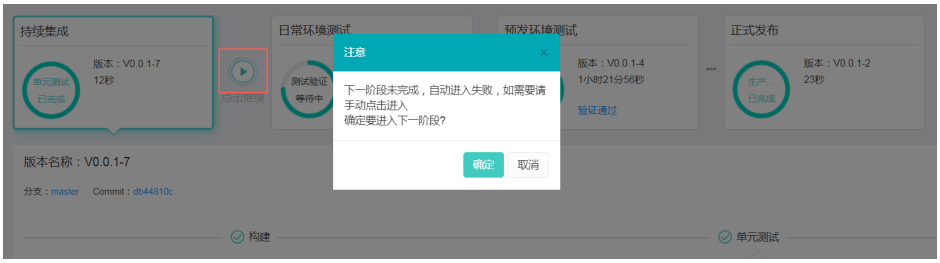
设置为自动触发时，当前一阶段完成且后一阶段空闲时，就会自动开始运行后一阶段。

当前一阶段完成时，若后一阶段正在运行，则后一阶段不受干扰，继续运行完成。待运行完成，出现空闲时，后一阶段自动启动，接续前一阶段版本运行。

若在后一阶段运行时，前一阶段完成了多次运行，则在后一阶段出现空闲时，接续前一阶段的最近一次运行版本运行。

举个例子：假定运行日常环境测试阶段需要一个小时，而运行持续集成阶段需要五分钟。在某个小时内，有三次代码提交，触发了持续集成阶段的三次运行。持续集成阶段第一次运行结束，触发了日常环境测试阶段的自动运行，将持续一小时。持续集成阶段第二次运行结束，由于日常环境测试阶段仍在运行，因此不会立即启动该阶段再次运行。持续集成阶段第三次运行结束，由于日常环境测试阶段仍在运行，因此也不会立即启动该阶段运行。待到日常环境测试阶段运行结束后，空闲下来，开始再次运行，此时是接续持续集成阶段第三次运行的版本继续运行，而不是接续持续集成阶段第二次运行的版本继续运行。

亦支持在前一阶段运行完成时，若后一阶段正在运行，强制中止后一阶段当前运行，以便后一阶段接续运行前一阶段最新版本。具体方法是，此时点击两个阶段之间的开始按钮（下图红框），然后在弹出框中点击确定：



阶段的启动运行：设置为手动触发时

设置为手动触发时，当前一阶段完成后，在两个阶段之间出现人工触发按钮，点击开始运行后一阶段。

此时，如果此时下一阶段正在运行，将弹出对话框确认中止正在运行的内容。

如果在下一阶段前，上一阶段已运行多次，则人工启动的下一阶段运行时，使用上一阶段最近一次运行的版本。

任务的启动运行

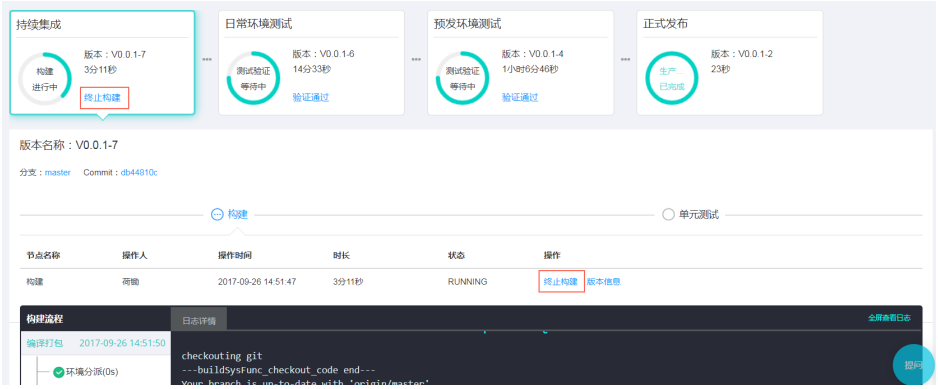
一个阶段内的串联的各任务，依次运行。前一个任务的完成，自动触发下一个任务的运行。

一个阶段内串联的各任务依次运行完毕后，该阶段才会再次运行。因此不会出现串联的各任务同时运行的情况。

另外，任务间并联，以便同时运行的功能已在开发中，敬请期待。

中止任务运行

有些类型的任务，提供了中止任务运行的方法。相应操作出现在流水线上的阶段中，和/或页面下方的任务中。以构建组件为例，见下图两个红框：



需要人参与交互的任务

任务既可以是完全自动运行的，也可以是需要人工参与的。比如，人工测试、人工发布审核等等。

此时，可以在流水线的相应阶段上，或下方相应任务中，进行人工操作。如图两个红框：

任务运行失败时的处理

当有一个任务运行失败时，该阶段的该次运行即停止。不会再触发下一阶段的运行。也就是说，本次流水线运行至此停止。

此时，如果分析是配置或环境的问题，可以考虑在修复问题后，再次运行。具体操作为，点击流水线上的“重试”按钮。如下图红框：

一个小技巧：如果希望任务运行失败时流水线能够继续运行，可以考虑改变任务的配置，让它总是向流水线返回成功。某些类型的任务能否实现这个方法，具体配置随任务类型的不同而不同。举例来说，对于单元测试组件，可以让自定义的命令行总是返回0。

附原理说明：任务间如何传递“版本”

前面提到，后一阶段接续前一阶段的版本继续运行。那么，“版本”具体是什么，如何承载呢？

流水线的第一个阶段的每次启动，都将会产生一个版本记录。典型的，包括源代码版本信息、相应包版本信息等等。该记录会逐任务逐阶段传递下去。

此外，每种任务类型，可以定义完成时向流水线上下文中，输出哪些参数。同时可以自定义开始时从流水线上下文中，获取哪些参数的值。据此，可以实现灵活的参数传递。

流水线的配置

本文介绍流水线的配置。若您尚不熟悉云效流水线服务，推荐从流水线概述开始阅读。

流水线的添加、修改与删除

当进入一个项目后，左侧菜单栏的“流水线”菜单项，是流水线的入口。（如果左侧菜单栏没有出现“流水线”菜单项，请前往“设置”->“服务”为该项目配置流水线服务。）

点击“流水线”菜单项，进入流水线列表页。此时左上角，有“新建流水线”按钮，可添加新的流水线。

流水线列表的每一行，末尾有“修改”按钮，点击进入该流水线的编辑页面，可修改该流水线的配置。而如果点击流水线列表每一行左侧的流水线标题，进入该流水线的主页面后，也可以点击主页面左上角的“编辑流水线”按钮，进入该流水线的编辑页面。如下图：

在该流水线的编辑页面，可配置如下内容：

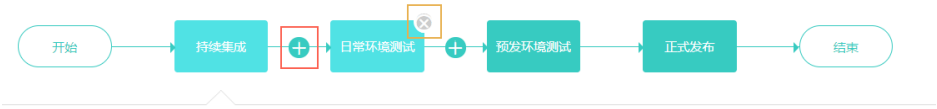
- 流水线名称。
- 流水线的管理员。仅管理员可编辑流水线。新建流水线时，当前用户被设为管理员。
- 监听设置。其中，自动触发是指，在源代码的修改被推送到服务器端代码库指定分支时，触发流水线运行。另外，配置为自动触发或定时触发后，在流水线主页也仍然可以手工触发流水线执行。

- 流水线各阶段的添加、修改与删除。详见下节。

流水线删除功能待上线。

阶段的添加、修改与删除

在流水线编辑页面的中间部分，显示流水线各阶段。当把鼠标移动到两阶段之间的连线时，页面显示两阶段间的加号图标（下图中红色框），和一个阶段的删除图标（下图中黄色框）。点击可分别新建阶段或删除一个已有阶段。



要想修改一个阶段，请用鼠标选中该阶段。于是下方将显示该阶段详情，可进行该阶段任务的添加、修改和删除，详见下一节。另外，阶段的名称也是可以修改的，只需再显示阶段名称的地方编辑即可。

任务的添加、修改与删除

选中一个阶段后，在下方按顺序显示该阶段的各已有任务，同时可以添加任务：

点击“+添加任务”，可以添加一个新任务。其中第一步是选择新任务的类型：

选中一个已有任务，在任务条目右侧，将展开该任务的配置，进而可以填写和修改。不同任务类型，其配置内容是不一样的。但通常有如下两项：

- 任务类型。这一项是在新建任务时确定的，不可修改。
- 任务名称。在流水线编辑时和运行时，都将显示该字段，标识这个任务节点。

点击每个已有任务上的删除图标，将删除该任务。

当前可选任务类型

构建

构建打包，供部署使用。这部分相关知识较多，请从构建概述读起。其中，流水线上构建任务的配置和运行，详见这里。

自定义脚本

如果您有比较定制化的需求（比如向使用自定义脚本发布静态资源 ，或者执行一些定时任务），那么您可以使

用自定义脚本。详见这里。

部署

把构建成果部署到服务器运行。这部分相关知识较多，请从应用部署概述读起。其中，流水线上部署任务的配置和运行，详见这里。

人工卡点

需要人工判断是否OK的任务。在流水线运行时，需要处理人到页面点击是否OK。这类任务可以用来作为流水

线上人工测试、安全审核等流程卡点。

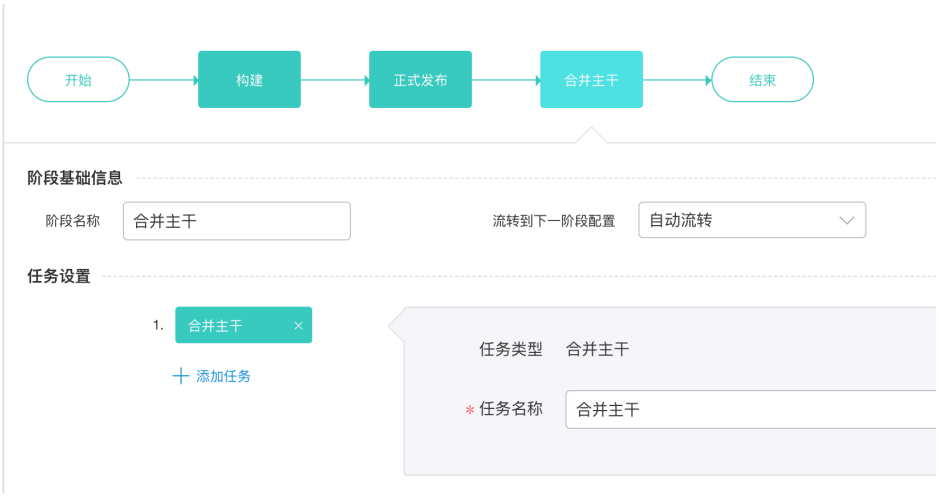
其中，处理人可以配置为具体人员（可多个），也可以配置为流水线所关联应用上的角色（仅可选一个）。

单元测试

自动运行一行命令，看是否能成功。

合并主干

您可以使用该组件将某个分支合并到主干（master），如下图所示：



系统账户

默认情况下，流水线上的与代码有关的操作（如clone，pull，push等）使用的都是当前操作人的权限。但对于合并主干的场景，不适合使用当前操作人权限。比如很多开发团队会把主干设置为保护分支，大部分开发人员没有权限进行代码push。对于这种情况，云效提供了系统账户作为解决方案。您可以在企业设置->代码托管中配置该账户。

设置系统账号之后，当企业内新建代码组时，本账号将自动成为该代码组的owner。对于已存在的代码组，请自行保证其有拥有相关权限（比如，更改系统账户时，需要手动对已存在代码组进行owner组权限赋权）。

合并主干异常排查方法

如果合并主干出现了异常，请按照如下方式进行排查。

1. 确认当前企业是否配置了系统账户。
2. 配置了系统账户时，请查看当前系统账户，是否拥有当前操作代码组的owner权限。如果没有请赋权。赋权之后，进行重试操作。
3. 未配置系统账户时，如果clone失败，请确认当前操作人拥有当前代码库权限；如果push失败，请确认当前操作人拥有当前代码库master权限或当前代码组owner权限。如果没有，请赋权之后进行重试，或者请拥有相应权限的用户进行重试操作。

分支模式下的流水线

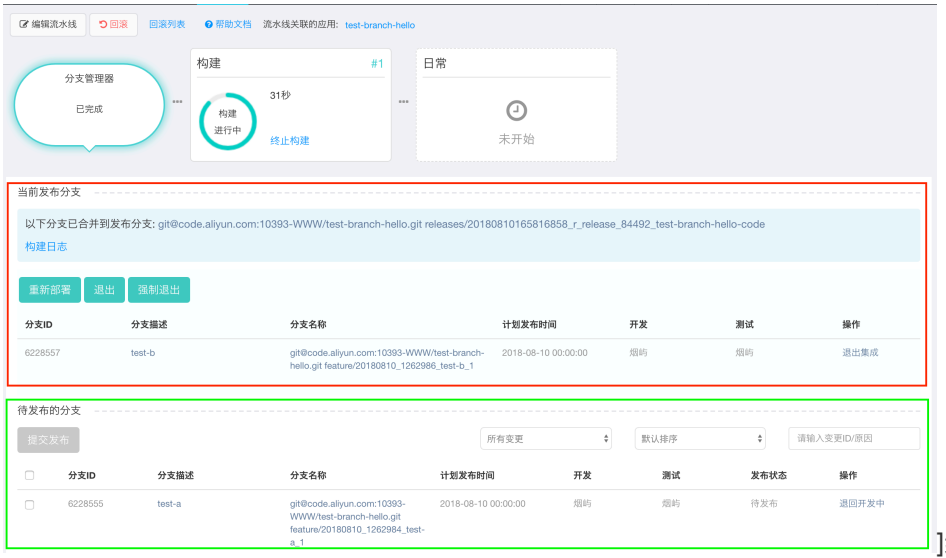
当使用向导新建一站式研发解决方案时，若研发模式选择了分支模式，则向导将自动创建分支模式所特有的流水线。

阅读本文之前，需要首先学习理解开发模式中的分支模式。详细介绍见这里。

分支模式流水线的特殊之处

分支管理

在分支模式下，feature分支（或特性分支）承载了单个缺陷或功能的开发，而release分支是用于集成和发布的。分支模式的流水线使用分支管理器来管理这两种类型的分支。在构建前，分支管理器会创建一条release分支，并将发布列表中的分支都合并到release分支上，然后用release分支的内容进行构建部署。【分支管理器】页面，有两个区域：当前发布分支和待发布的分支。



]绿色方框内展示的是待发布的分支列表，是所有已经开发完毕并做了适当检测，可以进行集成和发布的feature分支列表。做集成和发布时，就从这个列表中挑选，哪些合并到当前流程对应的release分支，以便部署和进一步集成测试。红色的方框展示的是当前发布的分支列表，就是是从待集成区中挑出来的，(打算)合并到release分支，并随后部署到当前流程相应的运行环境的feature分支列表。这个列表，反映的是当前环境中，包含了哪些代码改动。这些改动将被放在一起测试，进而发布上线用户仅需在页面上维护这两个feature分支列表，系统将自动完成release分支的创建和管理，feature分支到release分支的合并等一系列工作。

合并到主干

默认创建的分支模式流水线里面都有构建和部署（日常/预发/正式）两个阶段。正式的阶段包含两个任务【部署】和【合并主干】。【合并主干】任务执行的动作是将release分支的代码合并入master分支。

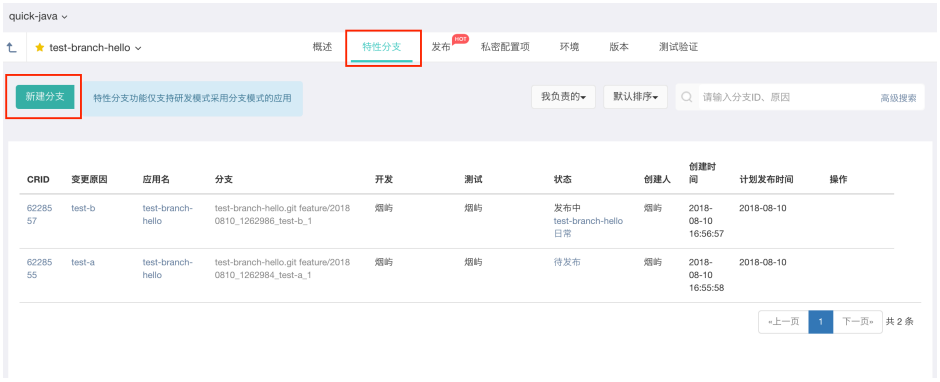
配置流水线的入口和方法

新建入口与其他流水线相同，当新建流水线时选择了分支模式的应用对应的代码库，系统会自动创建包含【分支管理器】的分支模式流水线。流水线的其他配置详见文档

日常操作

创建特性分支

创建特性分支是开始开发工作的第一步，打开应用的详情页面后，进入【特性分支】tab，点击【新建分支】按



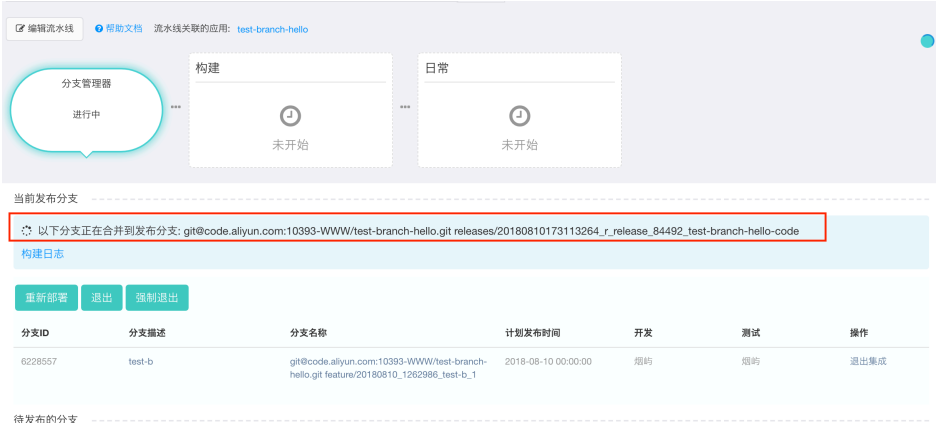
钮即可创建特性分支。

挑选分支合入release分支

特性分支创建后，默认状态是”开发中”。每个分支左侧，有“提交待发布”按钮。点击可将该分支状态置为“待发布”，也就是说，标记该特性分支已开发完毕并做了适当检测，可以进行集成和发布了。于是，该特性分支就进入了待集成区在特性分支页面，点击【提交集成】按钮，分支将进入待发布区，页面跳转至发布页面

在发布页面，勾选分支，点击【提交发布】，分支将被合并到release分支

提交后，页面将展示新创建的release分支信息



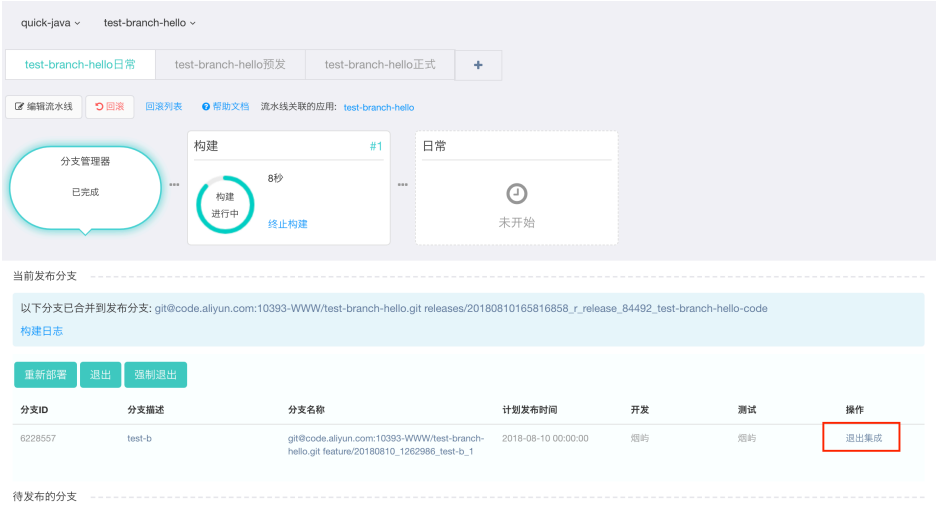
feature分支更新后，更新release分支并部署

feature分支有新提交后，【分支管理器】页面有红色提示信息，点击【重新部署】按钮将更新release分支内容并触发部署动作：

将某个特性分支的内容从集成中删除

当某个分支确定不需要部署到环境，在【分支管理器】页面，在当前发布列表右侧操作一栏，点击【退出集成】，分支将回到待发布列表，系统将：

1. 基于master分支，创建新的release分支
2. 除了该特性分支外的其他在发布区的分支合并到release分支
3. 把release分支的最新内容部署到环境



将部署成功的release分支部署到其他环境

当release分支的内容在某个环境验证通过后，需要进入下一个环境进行部署验证。在这样的场景下，可以点击流水线的部署组件，上方将展示【进入其他工作流】的按钮，如有多个环境，则有下拉框选择。点击选择后，系统会在选择的流水线将当前release分支的内容进行构建和部署，不会再创建新的release分支。

构建

构建概述

简单说来，云效流水线上的构建任务以及特性分支和分支集上的构建任务，根据指定Git库源代码根目录下的<应用名称>.release文件，进行构建打包工作，以便随后流水线上的部署任务进行部署。

<应用名称>.release文件，是用键-值对儿的形式定义了如何把源代码构建打包，在什么样的构建环境中打包，等等。。它的完整语法见可配置键的完整列表

有时我们需要构建产生不同内容的包，用于不同的运行环境（比如集成测试环境和生产环境）。甚至，为某个环境构建产生压缩包而为另一个环境产生Docker镜像。还有的时候，我们希望在构建时使用一些当时构建上下文的参数，比如构建时间、源代码分支名称等。云效支持这样的场景：

- 在流水线上的构建组件，支持一些相关的高级配置。详见流水线上的构建任务。
- 特性分支和分支集上的构建任务可以选择包标签。
- 构建过程可以受输入参数的影响。详见使用参数影响构建行为。

还有的时候，我们有一些私密配置信息，不适合与源代码存放在一起。云效提供了存放私密配置项的功能，详见这里。

以下文档给出了一些典型场景下配置构建的方法：

- Web应用构建配置
- 使用EDAS部署时的构建配置
- Docker镜像构建配置
- 无线应用构建配置

关于Maven仓库，目前云效使用全局的Nexus仓库maven.aliyun.com，供下载。若需要上传，企业可考虑搭建并使用企业私有的Maven仓库，详见在云效中使用私有Maven仓库。

流水线上的构建任务

构建任务，是云效流水线上的一类任务，它负责构建打包，供后续的部署任务使用。

构建任务的运行

构建任务一般不需要在运行时输入信息，就会自动运行。运行期间和运行结束后，可以在页面下方查看构建日志：

构建任务的配置

基本配置

在流水线编辑页面，添加任务时，请选择“构建”，并填写其基本配置：

高级配置：使用包标签

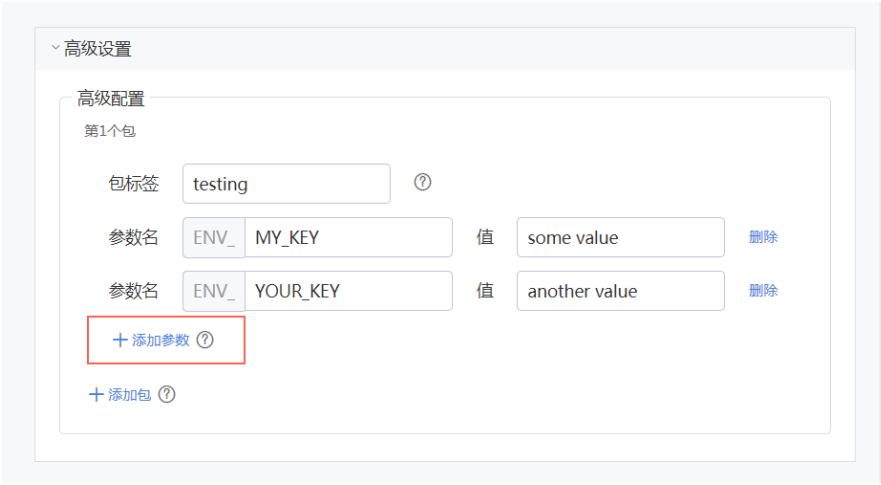
在流水线上配置构建任务时，点击打开高级配置，会看到“包标签”这个配置：

包标签的默认值是default，但可以调整这个配置。构建打包得到的包，用于不同用途时，可以在上面打上相应的标签。比如部署到日常测试环境的包和部署到预发环境的包、生产环境的包，需要有不同的内容，那么可以分别用testing、staging、production来区分这三类包。流水线上的部署组件，就可以根据需要，配置取得特定标签对应的包。比如，取得testing对应的包，用来部署日常测试环境。

那么，构建组件是如何根据包标签名的不同，打出不同的包呢？在构建时，系统会把包标签的值通过环境变量的方式，传到构建的上下文中。具体来说，该环境变量的key是PACKAGE_LABEL，值就是包标签的名字。于是，构建过程就可以据此进行调整，以产生适合这个构建目的的包。详见使用传入参数改变构建行为。

高级配置：增加输入参数

在包标签设置的下方，有“+添加参数”按钮，点击可添加key-value对：



这些key-value，将作为环境变量，传入构建过程。比如上图中，构建过程将获得ENV_MY_KEY这个环境变量，值为some value，以及ENV_YOUR_KEY这个环境变量，值为another value。构建过程就可以据此进行调整，以产生适合这个构建目的的包。详见使用传入参数改变构建行为。

高级配置：同时构建不同用途的多个包

可以在一个构建任务中，构建多个包。不同的包，至少包标签不同，还可以有不同的其他输入参数：

这些包，将在流水线上执行到该构建任务时，在多台机器上同时构建，以尽可能提高效率。其中任何一个包构建失败，将标记为该构建任务失败，流水线本次运行中止。

流水线上的自定义脚本任务

自定义脚本任务，是云效流水线上的一个任务，您可以使用该组件执行一些比较定制化的任务，比如使用自定义脚本发布静态资源 。

构建任务的配置

基本配置

在流水线编辑页面，添加任务时，请选择“自定义脚本”，并填写其基本配置：

目前支持的脚本类型：shell。运行环境为云效提供的基础环境：registry.cn-beijing.aliyuncs.com/rdc-builds/base:1.0。

环境变量

云效内置了以下环境变量供您使用：

变量名	描述	示例	备注
PIPELINE_ID	流水线ID	18	
BUILD_NUMBER	流水线运行编号	22	
EMPLOYEE_ID	操作人	aliyun_1234	
CODE_INFO	流水线里包含的代码信息	[{“appId”:45869,”appType”:”APP”,”branch”:”git@code.aliyun.com:sample-group/sample-app.git master”,”revision”:”2f238c881fbda09c403a9183513b45bd8b481b71”}]	流水线里包含代码类型的参数才会有，否则此参数为 null
VERSION	流水线里代码构建的版本	[{“appId”:45869,”appType”:”APP”,”branch”:”master”,”buildNumber”:7,”major”:0,”minor”:2,”revision”:35,”versionNumber”:”0.2.35.7”}]	在此任务前有构建任务，才会输出此参数，否则为 null
PACKAGES	流水线里代码构建打包结果	[{“appId”:45869,”fileMd5”:”ca0239f746476698d98fc1f5a4e55c3e”,”fileName”:”sample-app.tgz”,”fileUrl”:”https://rdc-build.aliyuncs.com/aone2/anonymous/build/package/download?path=0/15_0647187d-01e3-4a41-	在此任务前有构建任务，才会输出此参数，否则为null。并且：fileUrl的下载有效时间为5分钟

		8252-72775dc96890-sample-app-2018-04-12-14-40-05-747-sample-app.tgz&md5Sign=91c20112201751f83470de3edf52249d","packageLabel":"default","packageName":"sample-app","packageType":"APP"}]	

高级配置：运行时变量

您可以在触发任务时，指定不同的环境变量。如下图所示。

运行时您会看到：

点击设置参数之后，进入填写参数页面：

填写完毕之后，点击确认，流水线会继续运行。在本例子中，会继续打印出hello world。

特性分支和分支集上的构建任务

概述

如果您是在使用云效专有云版，且您的企业配置了在特性分支/集上构建的功能，那么您可以在特性分支/集上直接进行构建打包。

在特性分支上构建

在特性分支的详情页，有构建卡片，可点击构建。

您也可以在“我的”->“特性分支”列表中，看到构建卡片，并点击构建。

在一个分支集的详情页面中，有特性分支列表。列表中可以看到各特性分支的构建卡片，并点击构建。

选择包标签

在构建启动前，系统将让您选择包标签。

包标签的默认值是default，但可以调整这个配置。构建打包得到的包，用于不同用途时，可以在上面打上相应的标签。比如部署到日常测试环境的包和部署到预发环境的包、生产环境的包，需要有不同的内容，那么可以分别用testing、staging、production来区分这三类包。流水线上的部署组件，就可以根据需要，配置取得特定标签对应的包。比如，取得testing对应的包，用来部署日常测试环境。

那么，构建组件是如何根据包标签名的不同，打出不同的包呢？在构建时，系统会把包标签的值通过环境变量的方式，传到构建的上下文中。具体来说，该环境变量的key是PACKAGE_LABEL，值就是包标签的名字。于是，构建过程就可以据此进行调整，以产生适合这个构建目的的包。详见使用传入参数改变构建行为。

在特性分支集的多个特性分支上构建

分支集的详情页面中，可勾选打算构建的特性分支，然后点击“构建”按钮，一并构建。

系统除了让您选择包标签外，您还将有机会调整构建的顺序。

可配置键的完整列表

<应用名>.release文件存放在源代码所在Git库的根目录下。流水线的构建任务，根据这个文件构建打包，供后续的部署任务使用。

<应用名>.release是键-值形式的。例如：

```
code.language=oracle-jdk1.9
build.output=target/abc.war
```

这些键，可能带有前缀。比如docker.file带testing前缀，写为testing.docker.file。这些键的值，可能不是常数，而是带变量，比如docker.tag=${PACKAGE_LABEL}_${TIMESTAMP}。相关内容，详见使用传入参数改变构建行为。

下面给出这些可配置的键的完整列表：

键	默认值	可填写值	说明	是否必填
code.language	无	php5.6 php7.0 node6.x node7.x node8.x node9.x node10.x node11.x node12.x oracle-jdk1.7 oracle-jdk1.8 oracle-jdk1.9 scripts	用来确定构建使用的环境(详情)和默认构建命令(见说明1)	必填
build.command	见说明1	任意命令行	构建时执行的命令	选填
build.output	如果编程语言是node，php，scripts，则默认值为./。其它情况下，需要显式填写。	相对路径形式，从代码库根目录算起。可以是文件（比如target/xxx.war）、目录下全部文件（比如target/*，此时解压后无该目录名）或目录（比如target）。	需要最终打成tgz压缩包的内容。	选填
build.output.nottgz	False	True False	不要对build.output指定的输出物打压缩包	选填
deploy.appctl.path	无	该文件的相对路径形式，从根目	需要添加到压缩包的部署脚本文	选填

		录算起，比如appctl.sh	件 详情	
docker.repo	无	比如registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app	推送到Docker Registry上的镜像名称	制作Docker镜像则必填
docker.repo.pull	无	内容格式与docker.repo相同	当设置该值时，云效依然会使用docker.repo中的地址进行构建和push，但在传递给部署系统（比如阿里云容器服务）时，会使用docker.repo.pull指定的url为基准的镜像地址。一个典型的使用场景是阿里云容器服务集群在VPC中，希望使用registry的vpc地址进行镜像下载，则可以指定docker.repo.pull为registry-vpc.cn-hangzhou.aliyuncs.com/mynamespace/container-app	选填
docker.file	Dockerfile	该文件的相对路径形式，从根目录算起，比如Dockerfile	制作Docker镜像所用Dockerfile的路径	选填
docker.tag	${PACKAGE_LABEL}_${TIMESTAMP}	比如${TIMESTAMP}	推送到Docker Registry上的镜像标签名称	选填

说明1：build.command的默认值：

- 编程语言是Java的Web应用：mvn -U clean package -Dappname=$APP_NAME -P$PACKAGE_LABEL（关于$APP_NAME和$PACKAGE_LABEL，请参看使用参数影响构建行为）。
- 编程语言是Java的安卓无线应用：./gradlew clean assembleDebug(assembleRelease) --info -s。
- 编程语言是Node时的Web应用：npm --python=/usr/alibaba/install/python-3.5.0/bin/python3 --registry=https://registry.npm.taobao.org install --production。其中的--python部分是为了进行包含本地扩展的Node模块的编译，详见：https://github.com/nodejs/node-gyp

/https://github.com/nodejs/node-gyp。
- 其他情况，默认值为空，于是不进行构建。（可能进行生成Docker镜像、打压缩包等工作）

构建环境

本文描述云效使用的构建环境。下面会介绍可用的环境，如果您在构建中遇到了问题，可以查看构建环境调试。

基础环境

所有构建环境基于Ubuntu系统。

执行构建命令的用户是admin，拥有sudo权限。所以您可以使用sudo apt-get update && sudo apt-get install -y xxx 来安装需要的软件。

已经预装的软件：

1. g++ 4.9.2
2. gcc 4.9.2
3. make 4.0
4. curl
5. wget
6. unzip
7. python 3.5（不在PATH中，需要使用/usr/alibaba/install/python-3.5.0/bin/python3来引用）
8. git 1.9

各个语言的环境

通过设置release文件中的code.language的值，您可以使用相应语言、版本的构建环境。

Java环境

基于基础环境，并安装了：

1. maven 3.5
2. gradle 4.1

提供三个JDK版本：

1. oracle-jdk1.7（code.language=oracle-jdk1.7）
2. oracle-jdk1.8（code.language=oracle-jdk1.8）
3. oracle-jdk1.9（code.language=oracle-jdk1.9）

NodeJS环境

基于基础环境，并安装了：

1. python 2.7，默认置于PATH中，支持node-gyp编译。

提供nodejs版本：

1. node6.11.3 npm3.10.10 yarn0.27.5 cnpm6.0.0（code.language=node6.x）
2. node7.10.0 npm4.20 yarn0.27.5 cnpm6.0.0（code.language=node7.x）
3. node8.4.0 npm5.3.0 yarn0.27.5 cnpm6.0.0（code.language=node8.x）
4. node9.11.2 npm5.6.0 yarn1.15.2 cnpm6.0.0（code.language=node9.x）
5. node10.15.3 npm6.4.1 yarn1.15.2 cnpm6.0.0（code.language=node10.x）
6. node11.15.0 npm6.7.0 yarn1.15.2 cnpm6.0.0（code.language=node11.x）
7. node12.2.0 npm6.9.0 yarn1.15.2 cnpm6.0.0（code.language=node12.x）

注意：上述node及npm版本会随着相应的node大版本的更新而更新，但code.language的取值不变。比如您配置了code.language=node6.x，目前实际使用的是node6.11.3，如果node6的版本升级到了6.12.0，则您实际用到的可能就是6.12.0。

PHP环境

基于基础环境，并安装了：

1. composer 1.0

提供两个php版本：

1. php5.6 phpunit 5.7（code.language=php5.6）
2. php7.0 phpunit 6.3（code.language=php7.0）

Golang环境

基于基础环境，并安装了：

1. go-wrapper

提供两个golang版本：

1. go1.8.5（code.language=golang1.8）
2. go1.9.2（code.language=golang1.9）
3. go1.11.10（code.language=golang1.11）
4. go1.12.5（code.language=golang1.12）

Python环境

基于基础环境，提供两个python版本：

1. python2.7.13（code.language=python2.7）

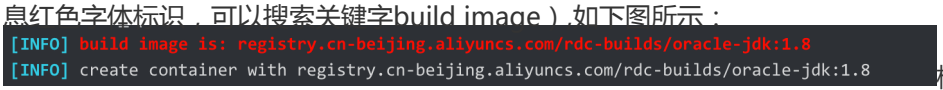
2. python3.5.0（code.language=python3.5）

其它

如果您的构建对环境没有特殊需求，可以使用code.language=scripts。此时会使用基础环境。

构建环境调试

云效使用docker镜像的方式提供构建环境，镜像可以公开下载到，具体的镜像地址可以在构建日志中看到（信息红色字体标识，可以搜索关键字build image），如下图所示：



构建命令在日志中也可以看到（信息红色标识，可搜关键字build command），如下图：

所以如果您在构建中遇到了问题，可以将相应的镜像拉取到本地，假设镜像地址为registry.cn-beijing.aliyuncs.com/rdc-builds/oracle-jdk:1.8，则使用下面的命令进行本地调试：

```
//在宿主机上执行，并进入容器
$ docker exec -ti `docker run -d registry.cn-beijing.aliyuncs.com/rdc-builds/oracle-jdk:1.8` bash
//在容器中切换到admin，因为云效会使用admin账户进行构建
root@eed5d6e8a6bc:/#su admin
// 开始您的调试，比如下载代码后，运行构建命令
git clone xxxxx
mvn -U clean package -Dmaven.test.skip=true -DskipTests=true -Dappname=test-pred -DAPP_NAME=test-pred
……
```

如果我们的镜像不能满足您的需求，建议使用自定义构建镜像。

使用传入参数改变构建行为

尽管一个应用仅对应一个<应用名称>.release文件，但根据构建时上下文中传入的参数的不同，可以改变构建的行为，输出不同的构建结果。典型的，为不同的运行环境（比如测试环境和线上环境），打出不同内容，甚至不同类型的包。下面详细讲解基于同一份构建配置文件，根据上下文参数改变构建行为的方法。

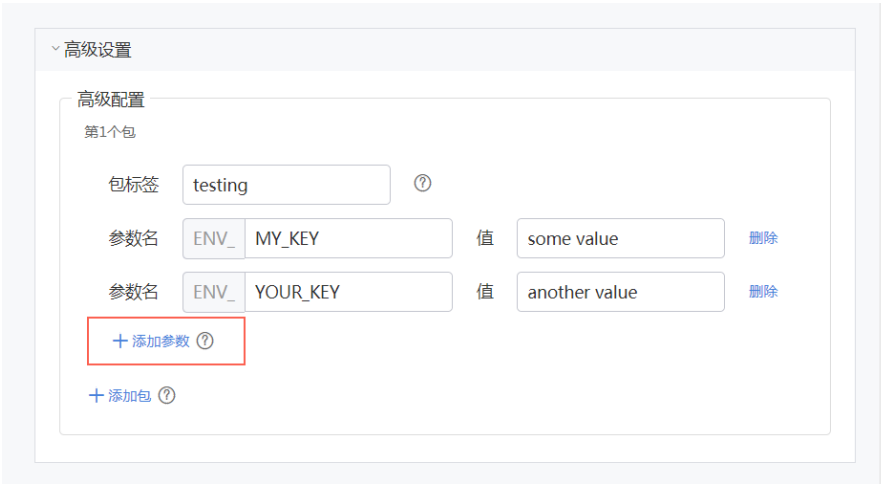
原理：包标签及其他环境变量

在流水线上配置构建任务时，点击打开高级配置，会看到“包标签”这个配置：

包标签的默认值是default，但可以调整这个配置。构建打包得到的包，用于不同用途时，可以在上面打上相应的标签。比如部署到日常测试环境的包和部署到预发环境的包、生产环境的包，分别用testing、staging、production来区分。流水线上的部署组件，就可以根据需要，配置取得特定标签对应的包。比如，取得testing对应的包，用来部署日常测试环境。

那么，构建组件是如何根据包标签名的不同，打出不同的包呢？在构建时，系统会把包标签的值通过环境变量的方式，传到构建的上下文中。具体来说，该环境变量的key是PACKAGE_LABEL，值就是包标签的名字。于是，构建过程就可以据此进行调整，以产生适合这个构建目的的包。调整方法详见下文描述。

除了包标签PACKAGE_LABEL这个环境变量，构建时系统还将把其他一些环境变量传入构建上下文。其中的一些，是系统自带的，详见**构建传入环境变量完整列表**。此外，用户还可以自定义若干key-value对：流水线上配置构建组件时的高级配置部分，在包标签设置的下方，有“+添加参数”按钮，点击可添加key-value对儿。



构建过程可以根据以上这些环境变量进行调整。详见下文。

方法1：以PACKAGE_LABEL的值作为配置键的前缀

<应用名称>.release中的键（比如docker.file），可以用PACKAGE_LABEL的值作为前缀（比如testing.docker.file）。构建时，如果找到以当时PACKAGE_LABEL的值（比如testing）作为前缀的键（比如

testing.docker.file），就将以它的值（比如Dockerfile），作为键（比如docker.file）的值。

当PACKAGE_LABEL为testing时候，云效会寻找所有不带前缀的键值，并与带testing前缀的键值，进行合并。带前缀的键值拥有更高的优先级。也就是说下面的例子中，docker.file最终的值是Dockerfile_test

```
docker.file=Dockerfile
testing.docker.file=Dockerfile_test
```

这个方法，不仅可以让特定键在为不同环境构建时，获得不同的值，甚至可以仅在特定环境获得值。比如，如果仅日常环境使用阿里云容器服务，需要进行镜像构建，但在生产环境希望使用ECS部署，不需要进行镜像构建。那么对于这个场景，可以使用下面的写法。

```
code.language=java
baseline.jdk=jdk-1.7.0_51
build.tools.maven=maven2.2.1

# 使用`PACKAGE_LABEL`的前缀（testing, staging, production）的键值，只会在相应环境的构建中生效
testing.docker.file=Dockerfile
testing.docker.repo=registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app
testing.docker.tag=${TIMESTAMP}
```

方法2：键值因为环境变量的值而改变

可以在键值中使用环境变量，例如：

```
docker.tag=${PACKAGE_LABEL}
```

还可以是组合，例如：

```
docker.tag=${PACKAGE_LABEL}_${TIMESTAMP}
```

再举个例子，使用PACKAGE_LABEL作为maven构建的profile。在release文件中配置build.command=mvn clean install -P$PACKAGE_LABEL。并且在您的pom文件中设置对应的profile，在其中定义不同环境的不同行为。

方法3：构建过程中引用环境变量

比如，设置build.command=sh build.sh，然后在build.sh中使用环境变量的值来进行判断，并执行不同的操作。

一个具体的例子：我想在不同包的构建中，使用不同的数据库配置。假设程序运行时从application.properties中读取数据库配置，则可以在代码库中放置三个配置文件：application.properties.testing，application.properties.staging，application.properties.production，在自定义流水线中也定义了的三个的PACKAGE_LABEL的值是testing，staging，production。然后在build.sh中包含这么一行，将属于相应环境的配置文件覆盖到程序读取的文件，也就是

application.properteis。

```
cp application.properties.$PACKAGE_LABEL application.properties
```

注意：上面只是一个例子，具体的配置文件的路径以你的项目的实际情况为准。

方法4：dockerfile中如何使用变量？

云效的应用中都有个<应用名称>.release文件,需要到release 文件中声明要传入的变量名称和值。

```
build.tools.docker.args=--build-arg APP_NAME=${APP_NAME} --build-arg ENV_TYPE=${ENV_TYPE} --build-arg
ENV_MY_KEY=${ENV_MY_KEY}
```

这里面的变量有两种，一种是系统内置的变量，如APP_NAME和ENV_TYPE，另一种就是用户在构建组件界面中配置的自定义变量，如MY_KEY,注意使用的时候需要加上ENV_前缀。Dockerfile 加入如下信息:

```
ARG APP_NAME
ARG ENV_TYPE
ARG ENV_MY_KEY
#将传入的应用名设置成环境变量
ENV APP_NAME ${APP_NAME}
ENV ENV_MY_KEY ${ENV_MY_KEY}
#类似用法可以发挥想象,注意shell 命令必须转换成行形式。可以续行
RUN if [[ "${ENV_TYPE}" = "testing" ]]; then echo ${APP_NAME}; fi
```

非自定义流水线时的情况

前面讲的在流水线上配置构建组件时配置PACKAGE_LABEL的值，指的是用户可自定义的流水线。目前有些情况下，系统仍然在使用不能这样自定义的流水线。典型的，当使用分支模式时，每个环境（典型的：日常测试环境、预发环境、正式环境），都对应一条不能用户灵活配置的流水线。此时，PACKAGE_LABEL的值被分别设置为testing，staging，production。（也就是旧版本云效中的ENV_TYPE的值）

此外，当不是自定义流水线时，用户也无法传入更多的自定义环境变量。

构建传入系统环境变量的完整列表

流水线上的构建任务，接受流水线框架传入的环境变量（包括系统自带的和用户在构建任务里自定义的），并可据此改变构建行为。详细介绍见使用传入参数改变构建行为。本文档列出其中所有的系统自带的环境变量。

环境变量名	说明
PIPELINE_ID	流水线 ID

PIPELINE_NAME	流水线名称，比如“前端项目发布”
PROJECT_DIR	运行命令的工作目录，比如“/root/workspace/1084-abc_docker-08191_b0wE”
PACKAGE_LABEL	包标签，比如testing、staging、production或默认值default。 详情
APP_NAME	应用名。
CODE_BRANCH	代码库分支名。
DATETIME	当前时间戳，比如2017-06-22-23-26-33。

如果流水线配置代码源，则会有以下内置环境变量：

环境变量名	说明
CI_COMMIT_REF_NAME	代码库的分支名，比如 master
CI_COMMIT_TITLE	最后一次提交的提交信息
CI_COMMIT_SHA	最后一次提交的代码版本的 commit ID：如 2bfb63d779e3648c91950f82d374a25784cdabaf

2. 流水线运行参数

在流水线编辑页面，可以定义流水线的环境变量，在流水线运行时，可以将环境变量执行过程的任何阶段使用这些变量。在添加全局变量的时候，选择 “运行时设置” 选项， 可以在流水线运行时进行参数动态配置

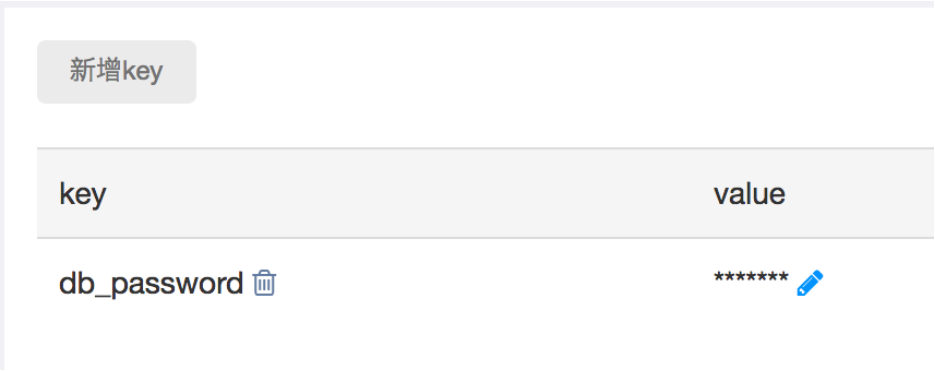
私密配置项

在应用构建中，通常会需要一些配置项，如：

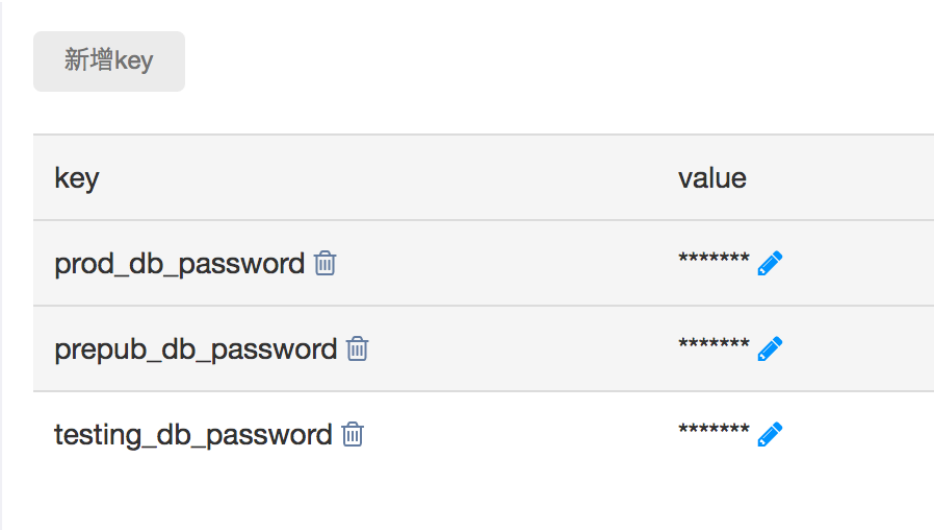
1. 功能开关
2. 依赖系统的URL
3. 数据库链接用户名密码

对于前两项，云效没有提供额外的支持，推荐您直接在代码中保存不同的配置文件，然后在构建时根据PACKAGE_LABEL的环境变量，选取正确的配置文件。详见使用传入参数改变构建行为。

第三项中的配置项会涉及一些私密信息，不适合放在代码库中。云效提供了私密配置项的保存功能。您可以在具体应用的私密配置项页面（从具体应用的顶部菜单栏中“私密配置项”菜单项进入）添加和配置**应用**级别的私密配置项，比如：



如果您需要在多个环境都使用私密配置项，则可以考虑把PACKAGE_LABEL的环境变量的值作为配置项的一部分：



配置好这些私密配置项之后，在进行构建时，云效会把这些配置项转换成为一个明文的文件，并将其放置在根目录下的rdc_security_config.properties中，比如：

rdc_security_config.properties:

```
prod_db_password=someprodpasswd
prepub_db_password=someprepasswd
tesing_db_password=sometestingpasswd
```

您可以在自己的构建脚本中读取该文件，并按照您自己的方式进行使用。其中，由于在构建上下文中可以获得环境变量PACKAGE_LABEL的值，因此可以据此知道相应私密配置项的名称，进而取得值。详见使用传入参数改变构建行为。

Web应用构建配置

概述

本文讲解Web应用构建相应的配置。关于构建的更多内容，比如<应用名>.release是什么，请从构建配置概述读起。

完成Web应用的构建配置后，请继续部署配置，参见部署配置：通过脚本部署。

Java构建

此时<应用名>.release文件基本内容：

```
code.language=oracle-jdk1.9
build.output=target/<应用名>.war
```

这意味着，将在Java构建环境（详见构建环境）中，使用Java默认构建命令mvn -U clean package -Dappname=$APP_NAME -P$PACKAGE_LABEL进行构建，随后把构建输出target/<应用名>.war打为tgz包并保存，供后续部署使用。

如果想改变构建命令，需要设置build.command。详见可配置键的完整列表中的build.command。

默认的maven settings会把所有的repository都镜像到maven.aliyun.com下载依赖，如果您需要不同的配置，只需要在代码根目录放置您的settings.xml，云效会使用该文作为构建的settings.xml。

如果需要使用私有maven仓库下载依赖或上传二方库，具体做法详见在云效中使用私有maven仓库。

Node构建

此时<应用名>.release文件基本内容：

```
code.language=node8.x
```

这意味着，将在Node构建环境（详见构建环境）中，使用Node默认构建命令npm --python=/usr/alibaba/install/python-3.5.0/bin/python3 --registry=https://registry.npm.taobao.org install --productionL进行构建，随后把构建输出./(源代码根目录)打为tgz包并保存，供后续部署使用。

如果想改变构建命令，需要设置build.command。详见可配置键的完整列表中的build.command。类似的，如果想改变打包范围，需要设置build.output。

Node构建通过engines的方式来获得特定的版本，具体方式是在package.json中添加如下片段：

```
...
```

```
"engines": {
"node": ">=5.1.0"
},
...
```

则云效会根据使用您指定的版本。该机制背后使用的是nvm，所以只要是nvm支持的版本，都可以填写。

PHP构建

此时<应用名>.release文件基本内容：

```
code.language=php7.0
```

系统将简单的把./(源代码根目录)打为tgz包并保存，供后续部署使用。如果希望构建，请设置build.command，于是将在PHP构建环境（详见构建环境）中，据此构建后再打包。

其他情况

此时<应用名>.release文件基本内容：

```
code.language=scripts
```

系统将简单的把./(源代码根目录)打为tgz包并保存，供后续部署使用。如果希望构建，请设置build.command，于是将在基础环境（详见构建环境）中，据此构建后再打包。

补充说明

灵活配置构建环境

在build.command中，可以指定任意构建命令，比如build.command=sh build.sh，所以如果需要安装软件，或者执行复杂的命令，都可以通过这种方式实现。

环境变量对于构建过程的影响

关于环境变量对于构建过程的影响，请参看使用参数影响构建行为

不同环境使用不同的构建配置

云效支持为不同的运行环境打不同的包。为此，在不同的环境中使用不同的构建配置。详见使用参数影响构建行为。

修改一个环境的构建配置后，考虑相应的修改该环境的部署配置。详见应用部署概述。

关于包的管理

目前不提供压缩包的下载，该压缩包会在进行部署时候，直接传到指定机器上。详见部署配置：通过脚本部署。

使用EDAS部署时的构建配置

使用EDAS部署Web应用时，Web应用构建的配置有少许特殊性，详见下文。

关于Web应用构建的一般方法，请参考Web应用构建配置。关于构建的更多内容，请从构建配置概述读起。

完成构建配置后，请继续部署配置。关于Web应用通过EDAS部署，请参见部署配置：通过EDAS部署。关于部署的更多内容，请从部署配置概述读起。

配置云效不对构建物进行压缩

云效默认会将build.output所指示的war包或者jar包再打成tgz包，而EDAS接受的是war包或者jar包。所以需要在<应用名称>.release文件中进行如下配置，使得云效不再打包。配置示例如下：

```
...
# 打包的产物为target/xxx.war
build.output=target/xxx.war
# 不要再对 build.output 指定的输出物再进行打包
build.output.nottgz=True
...
```

一个完整的release文件的例子

假设应用名为edas-app。

edas-app.release:

```
code.language=oracle-jdk1.9
build.output=target/edas-app.war
build.output.nottgz=True
```

Docker镜像构建配置

在完成Web应用构建的基础上，继续构建生成Docker镜像，以便通过阿里云容器服务部署。本文讲解如何构建生成Docker镜像。

关于Web应用构建的一般方法，请参考Web应用构建配置。关于构建的更多内容，请从构建配置概述读起。

完成构建配置后，请继续部署配置。关于通过容器服务部署，请参见部署配置：通过容器服务部署。关于部署的更多内容，请从部署配置概述读起。

企业全局配置

镜像构建的用户名密码：一个企业内部可以共享一个docker login的用户名密码。可以在企业设置-> 容器服务账号中添加。

一个构建配置示例

下面给出一个容器构建配置的完整示例（假设应用名为container-app）。

代码库目录结构：

```
$ tree .
.
├── Dockerfile
├── pom.xml
├── src
└── container-app.release
```

构建配置文件container-app.release：

```
code.language=oracle-jdk1.9
build.output=target/container-app.war

# docker构建所用的Dockerfile的路径
docker.file=Dockerfile

# docker构建完成之后，要push到的docker repo
docker.repo=registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app

# 使用时间戳做docker tag，这样打出来的docker镜像就形如：registry.cn-
hangzhou.aliyuncs.com/mynamespace/container-app:20170622232633
docker.tag=${TIMESTAMP}
```

Dockerfile：

```
# 为自己的应用程序打一个基础镜像，把基础的软件安装好，并且包括启动的entrypoint
FROM registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app:base

# 上面提到了，云效会把container-app.tgz放到Dockerfile的同级目录，所以可以这么写，把云效打出来的软件包拷贝到镜像中
COPY container-app.tgz /home/admin/container-app.tgz
```

pom.xml，src 略。

按照上述的配置，云效会：

1. 先按照默认的Java语言的构建方式打出war包在target/container-app.war。
2. 把target/container-app.war打成container-app.tgz，并放置在代码库根目录。
3. 运行docker login命令：docker login -u "xxx" -p "xxx" registry.cn-hangzhou.aliyuncs.com。
4. 运行docker构建命令：docker build --pull -f Dockerfile -t registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app:20170622232633 /home/admin/xxxxx/container-app/
5. 再次运行第3步中的命令。
6. 运行docker push命令：docker push registry.cn-hangzhou.aliyuncs.com/mynamespace/container-app:20170622232633

构建配置详解

容器构建配置项

1. docker.repo：必填，镜像仓库的地址。
2. docker.file：选填，dockerfile相对代码根目录的相对路径。
3. docker.tag：选填，镜像tag的规则。

更多见可配置键的完整列表。

可用的环境变量

系统提供的所有环境变量见这里。下面列举几个常用的环境变量。

1. PACKAGE_LABEL：标识当前构建的参数。对于自定义流水线来说，如果您没有进行特殊配置，则PACKAGE_LABEL的值为default。您可以在自定义流水线的配置页面，对不同的包配置不同的PACKAGE_LABEL的值。如果您使用的是非自定义流水线，则PACKAGE_LABEL的值与ENV_TYPE的值相同。其值可以为testing, staging, production。
2. CODE_BRANCH：标识当前构建的分支名称。如果使用分支模式，则每次构建的分支是一个集成分支，形如：releases/20170623154859032_r_release_35191_app-code。这时最好不要使用CODE_BRANCH。

3. TIMESTAMP：当前时间戳，形如：20170622232633。

配置项默认值规则

1. docker.repo：无默认值。
2. docker.file：默认值为Dockerfile。
3. docker.tag：默认值为${PACKAGE_LABEL}_${TIMESTAMP}。

镜像构建的Context

关于镜像构建的Context的基础知识见链接。

云效会使用Dockerfile所在的路径进行镜像构建，也就是说镜像构建的Context就是Dockerfile所在的目录。

举个例子，如果Dockerfile的路径是docker/files/Dockerfile_testing，那么云效会把docker/files作为镜像构建的Context。

云效还会把打包产物自动拷贝到镜像构建的Context中。在上面的例子中，会把打出来的tgz包拷贝到docker/files目录下。tgz包的默认名称为<应用名>.tgz。这就意味着你可以在Dockerfile中直接这么写：

```
COPY `<应用名>.tgz` /home/admin/app-path/
```

典型场景

可以通过上述环境变量的不同组合来满足不同的使用场景：

不同环境使用不同的Dockerfile

release文件：

```
...
docker.file=Dockerfile_${PACKAGE_LABEL}
...
```

这样，在日常环境就会使用Dockerfile_testing，预发环境使用Dockerfile_staging，正式环境使用Dockerfile_production

使用时间戳作为镜像tag

release文件：

```
...
docker.tag=${TIMESTAMP}
...
```

这样，生成的tag就是形如20170622232633这样的字符串。

在tag中区分环境

使用同一个Dockerfile的用户，如果希望从镜像的tag中，看出来镜像是哪个环境的，则可以这样配置release文件：

```
...
# 这也是tag的默认值
docker.tag=${PACKAGE_LABEL}_${TIMESTAMP}
# 或者
docker.tag=${TIMESTAMP}_${PACKAGE_LABEL}
...
```

在tag中使用分支

有些团队会使用分支区分开发状态，比如develop分支上打出来的包都是测试包，master上打出来的是正式包。这时候可以在tag中包含CODE_BRANCH这个环境变量。比如docker.tag=${TIMESTAMP}_${PACKAGE_LABEL}_${CODE_BRANCH}。

自定义构建镜像

当构建环境中预置的的编译环境不能满足您的要求时。您可以使用自定义构建镜像的功能来定制所需的构建环境。

制作构建镜像

您需要按照如下的约束来编写构建环境使用的Dockerfile

- 使用指定的基础镜像：registry.cn-beijing.aliyuncs.com/rdc-builds/base:1.0
- 根据需要，安装软件和设置环境变量（admin为构建使用账号，不要删除或修改UID）。
- 整个镜像大小需控制在1G之内。(如需调整请提交工单)
- 请不要自定义CMD，系统脚本已指定镜像的启动命令。否则，会影响镜像启动使得镜像上传失败

一个完整的Dockerfile示例如下：

```
FROM registry.cn-beijing.aliyuncs.com/rdc-builds/base:1.0

RUN cd /tmp && \
wget http://rdc-public-software.oss-cn-hangzhou.aliyuncs.com/jdk-7u80-linux-x64.tar.gz && \
tar xf jdk-7u80-linux-x64.tar.gz -C /srv/java && \
```

```
ln -s /srv/java/jdk* /srv/java/jdk
ENV JAVA_HOME=/srv/java/jdk \
PATH=${PATH}:/srv/java/jdk/bin:/srv/java
```

在本地调试通过后，将镜像上传到阿里云或其他公网可访问的registry，且为公开权限。

录入镜像信息

镜像上传成功后，在“企业管理”->“构建镜像管理”页面，点击【新增镜像】按钮进行录入。

注意，为了保证镜像地址的准确性，云效要求精确到digest级别，所以要求镜像地址格式为：REPOSITORY@DIGEST，例如：registry.cn-hangzhou.aliyuncs.com/rdc-template/test-build@sha256:468da687c09b865e87641d2fff9de8feea048fb64ed98884214642248b364129。

如果您更新了镜像，需要到云效更新镜像digest，才能保证构建使用的是更新后的镜像。

使用构建镜像

在代码库的根目录下的<appName>.release文件，添加镜像配置：build.image=<your image repo url>。<your image repo url>需要与在镜像录入页面填写的镜像地址保持一致。修改后触发构建，新的构建会使用配置的镜像作为构建环境。

私密配置项

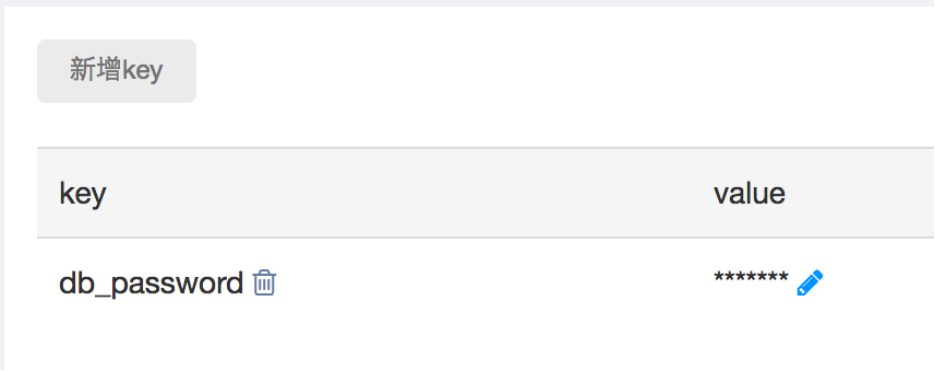
在应用构建中，通常会需要一些配置项，如：

1. 功能开关
2. 依赖系统的URL
3. 数据库链接用户名密码

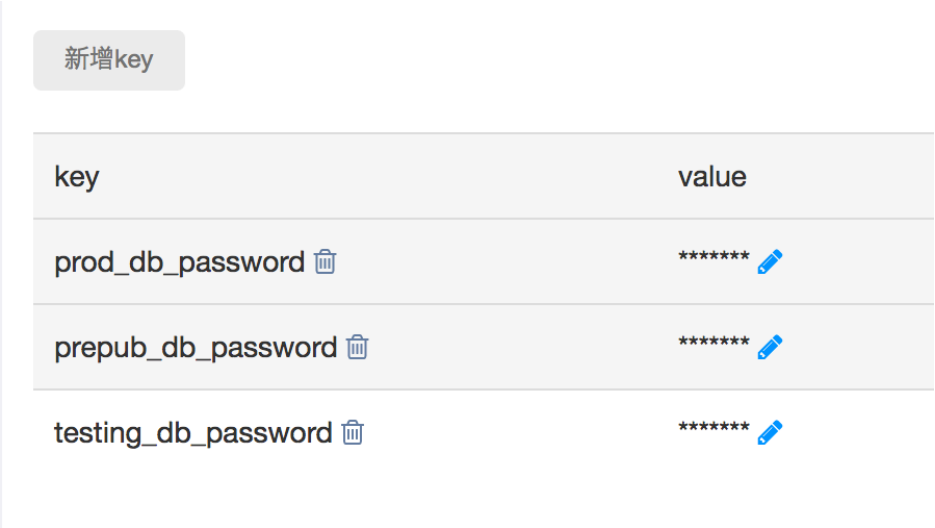
对于前两项，云效没有提供额外的支持，推荐您直接在代码中保存不同的配置文件，然后在构建时根据PACKAGE_LABEL的环境变量，选取正确的配置文件。请阅读使用传入参数改变构建行为，了解更多关于

PACKAGE_LABEL的用法。

第三项中的配置项会涉及一些私密信息，不适合放在代码库中。云效提供了私密配置项的保存功能。您可以在应用的私密配置项页面（https://rdc.aliyun.com/ec/app/xxx/securityConfig ）添加和配置**应用**级别的私密配置项。比如：



如果您需要在多个环境都使用私密配置项，则可以使用如下的方式：



配置好这些私密配置项之后，在进行构建时，云效会把这些配置项转换成为一个明文的文件，并将其放置在根目录下的rdc_security_config.properties中，比如：

rdc_security_config.properties:

```
db_password=somepasswd
```

您可以在自己的构建脚本中读取该文件，并按照您自己的方式进行使用。

实例

接下来给出一个实际使用的例子：

代码库结构:

```
├── README.md
```

```
├── build.sh
├── config
│   ├── application.prepub.properties
│   └── application.production.properties
├── app.release
├── pom.xml
└── src
```

application.prepub.properties的内容：

```
key1=somevalue
key2=somevalue
key3=somevalue
db.password=${db_password}
```

私密配置项内容：

构建配置（这里给出两个包标签，在构建时会打出两个包）：

build.sh内容：

```
// 将rdc_security_config.properties中与当前PACKAGE_LABEL有关的配置提取出来，去除PACKAGE_LABEL前缀，并转化成为可以source的格式。在这个例子中也就是：export db_password=xxxxxx，最后存入临时文件
cat rdc_security_config.properties | grep ${PACKAGE_LABEL} | sed s/^${PACKAGE_LABEL}./export\ /g > tmpconfig
cat tmpconfig
// source临时文件，将配置导入到环境变量中
source tmpconfig
// 对指定配置文件（config/application.${PACKAGE_LABEL}.properties）中的环境变量占位符使用环境变量中的值替换，从而变成私密配置项中配置的值
perl -pe 's/\$\{(.*)}/ $ENV{$1} /e' < config/application.${PACKAGE_LABEL}.properties > config/application.properties
// 打印出生成的配置文件的内容，用于调试。实际使用过程中去除该行：
cat config/application.properties
// 删除临时文件及其它环境的配置项，因为产物包不需要这些
rm -rf config/application.*.properties tmpconfig
// 做其它的打包工作，比如mvn package等
```

根据构建配置，云效会打出两个包，下面是prepub这个包的构建日志：

```
[INFO] build command is: sh build.sh
[INFO] run: sh build.sh
export db_password=someprepubdbpassword
key1=somevalue
key2=somevalue
key3=somevalue
db.password=someprepubdbpassword
```

对照上面的build.sh，可以看到config/application.properties的值已经变成了预期的内容。

开发模式

开发模式概述

开发模式意味着：

- 关于各类分支的命名、用法的约定
- 云效工具相应的行为

下面详细介绍各开发模式：

- 自由模式
- 分支模式
- Git Flow模式(待上线)

注意事项：当前开发模式暂不支持修改。

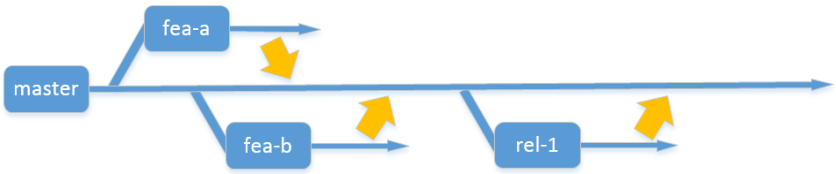
自由模式

自由模式，顾名思义，用户可以使用任何分支（包括master）进行打包、发布等操作。

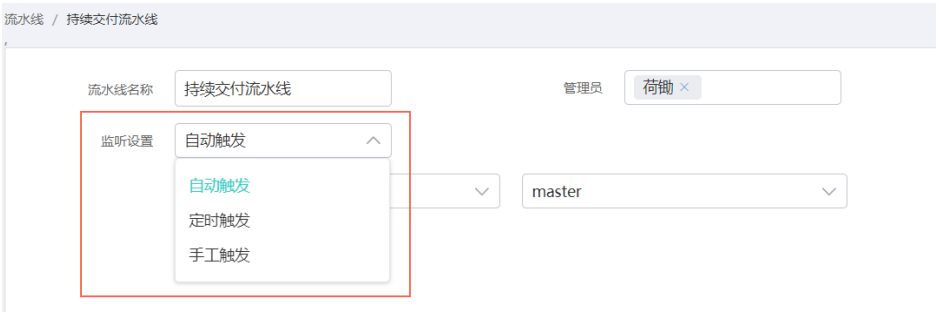
在自由模式下，常见用master分支这一条分支来承载开发、集成和发布，这被称作**主干开发方式**。使用这种方式，只有在特定情况下，才会使用其他的分支。包括：

- 确有必要时，拉出feature分支开发特定feature，开发完成并验证后合并回master。

- 确有必要时，拉出release分支，发布特定版本。随后合并回master。



自由模式时，在流水线上，通常默认配置为，master分支变化时自动触发流水线运行，取master分支做构建，并随后部署和发布。如果用其他分支构建，请修改触发条件：



以及构建任务的配置：

详见流水线的配置以及流水线上的构建任务。

分支模式

分支模式是云效支持的三种研发模式的一种。每种研发模式，不仅意味着其中各(类)分支的使用方式，也意味着云效能够向用户提供的相应支持，分支模式也不例外。事实上，云效对分支模式提供了强有力的支持：用户可以只需要关心集成和发布哪些feature分支，而对release分支创建和管理、分支间合并等一系列工作，可以托付给云效系统完成。

本文详细介绍分支模式下，各(类)分支的使用方式。关于云效提供的相应支持，详情请阅读分支模式下的流水线。

master代表最新发布版本

master分支代表最新发布版本。当需要最新发布版本的内容时，直接取分支末端即可。

不论其他哪类分支，都建议一般从master分支创建，并且经常从master分支合并，以便跟上“潮流”，减少将来集成时的各种问题，比如代码合并冲突。

每当软件正式发布前，系统会确保它基于master最新。

每当软件正式发布后，系统会把相应内容合并回master，以便让master分支始终代表最新发布版本。

一般来说，**使用者不要直接“写”东西到master分支**。把“写”的工作交给系统适时自动完成。

在各feature分支上开发

一条feature分支（又称变更分支、开发分支），通常用来承载一个缺陷的修复，或者一个需求（如果不是很大的话）的开发，或者任务分解后一个任务的开发。

一般来讲，基于master分支最新版本创建feature分支。然后在feature分支上开发、测试，直到这个feature功能完成，质量OK，准备好去集成和发布。

release分支上的集成

release分支用于集成和发布。基于master分支最新版本创建一条release分支，然后把想要集成的各条feature分支合并到这条release分支，进行部署和测试工作。

如果有新的feature分支要加入本次集成，那就把它也合并进这条release分支，然后再次部署并测试。

如果测试发现问题，就到feature分支上修复，然后把它再次合并到release分支，把修复带到release分支。

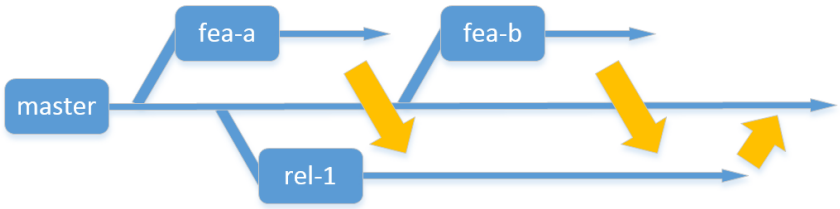
当然如果一个feature的问题太多太大，那干脆就放弃它。也就是说，新建一条release分支，把其他feature分支都合并过去，唯独不再合并这条feature分支。

就像master分支一样，release分支也是由系统自动管理的。**使用者不要直接在上面改代码**，代码修改请总是在feature分支完成。

release分支上的发布上线

当release分支上的质量足够好，想本次想上线的功能也都具备之后，就要考虑发布上线的问题啦。如前面讲的，发布上线前，会确保它基于master最新。而发布后会把release分支合并回master，让master代表最新发布版本。

以上几节介绍的内容，见下图：



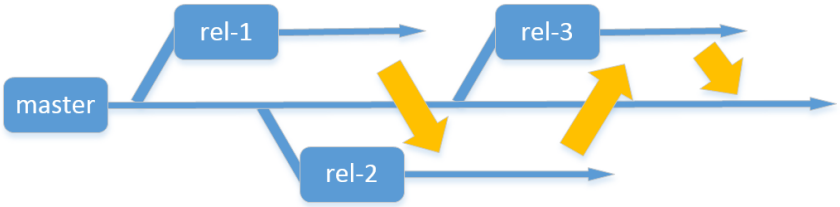
多个环境/流程时

假定要想集成发布上线，要经过日常测试环境上的测试这个流程，还要经过预发环境上的测试这个流程，那么两个流程用一条release分支就有些不合适。因为两个流程可能同时在测不同的feature分支集合。

分支模式用这个办法避免这个问题：每一个测试环境，也就是每个流程，关联它自己的release分支。日常测试、预发测试这两个环境（也就是两个流程），分别关联两条release分支。这样就不会相互影响。推而广之，为正式运行环境，也对应一条release分支。也就是说，每个环境都有对应的release分支。

当把集成成果从一个环境传递到下一个环境时，就是把一个环境下已合并到一起的feature分支，再往另一个环境对应的release分支上合并一遍……这么做有点儿笨。系统实际的做法是，基于master分支创建另一个环境对应的release分支，然后把前一个环境对应的release分支合并到新的release分支上。

本节介绍的内容，对应下图：



小结

以上就是关于分支模式这种研发模式的原理性介绍。更多细节在分支模式下的流水线中讲解。

特性分支管理

特性分支

概述

特性分支是指为一个特定的需求/任务/缺陷创建的分支，在其上完成相应开发后，一般会把它合并到集成/发布分支，与其他改动（若有）一起集成并最终发布。

当**研发模式**是**分支模式**时，云效平台为特性分支提供了特别的支持：可以查看特性分支列表，管理每次集成进入哪些特性分支，查看每次发布包括哪些特性分支，自动完成从特性分支到集成-发布分支的合并，等等。

当**研发模式**不是**分支模式**时，云效平台暂不提供特别的支持，仅提供基本的代码托管服务。您可以自行完成特性分支（若有）的创建、合并、删除等操作。

特性分支列表

点击吊顶“我的”菜单项，然后从左侧菜单中选择“特性分支”，即进入我的特性分支列表。这里列出的是我是开发者的所有的特性分支。

在进入具体项目后，从左侧菜单中选择“特性分支”，即进入该项目的各应用的特性分支的总列表。

在进入具体项目的具体应用后，从上方菜单中选择“特性分支”，即进入该应用的特性分支列表。

上述各列表中仅显示**分支模式**的应用的特性分支。另一方面，当具体应用不是**分支模式**时，它的菜单中不会出现“特性分支”菜单项。

上述各列表中的每条特性分支，有若干属性显示，有可能有相关操作可点击。

新建特性分支

在具体应用的特性分支列表页，点击左上方“新建”按钮，进入新建特性分支页面，填写各项内容，即可新建特性分支。

在新建特性分支页面，也可以选择关联代码库中已有的分支，将其注册为特性分支。于是，该分支就会出现在特性分支列表中，可对其进行特性分支相关的各种操作和查看。

特性分支详情页

在特性分支列表中，点击某条特性分支，即进入该特性分支的详情页。

在特性分支详情页，除展示信息外，通常还有一些操作按钮。比如提交待合并等操作。具体操作方法，详见分支模式中的介绍。

分支集

适用范围

本文档描述的功能仅供部分专有云用户使用。云效公有云尚未开通此功能。

概述

为了完成一个特性（比如一个需求或者研发任务），有时仅修改一个应用的源代码是不够的。这时，就要在不止一个代码库中，拉出相应的特性分支，修改源代码，并随后集成发布。这些为完成该特性的特性分支的集合，我们简称分支集。

如果本企业在云效上配置了分支集这个功能，那么当研发模式是分支模式时，云效平台为分支集提供了特别的支持：可以查看项目的和我的分支集列表，查看分支集包含的特性分支，方便地查看分支集及其各特性分支的状态并进行提交待发布等操作。

如果本企业在云效上没有配置分支集这个功能，或者当研发模式不是分支模式时，云效平台暂不提供特别的支持。您可以自行完成特性分支（若有）的创建、合并、删除等操作。

分支集列表

> 点击吊顶“我的”菜单项，然后从左侧菜单中选择“特性分支”，进而选择“特性分支”旁的“分支集”标签页，即进入我的分支集列表。这里列出的是我所在的所有的分支集。
>
> 在进入具体项目后，从左侧菜单中选择“特性分支”，，进而选择“特性分支”旁的“分支集”标签页，即进入该项目的各应用的分支集的总列表。

上述各列表中的每个分支集，有若干属性显示，有可能有相关操作可点击。

新建分支集

在分支集列表页，点击左上方“新建”按钮，进入新建分支集页面，填写各项内容，即可新建分支集。

分支集详情页

在分支集列表中，点击某个分支集，即进入该分支集的详情页。

分支集详情页的内容主要包括：

- 一些基本信息，如分支集名称、说明、各角色人员、分支集状态等。
- 该分支集包含的特性分支列表。可以在列表中查看这些特性分支的关键信息，进行高频操作，或前往特性分支详情页。详见特性分支中的介绍。

特性分支应用级集成视图

如果您的企业在使用分支模式下自定义流水线功能（目前仅对部分企业开通该功能），当使用向导新建一站式研发解决方案时，若研发模式选择了分支模式(而不是自由模式或Git Flow模式)，则向导将自动创建带有特性分支集成视图的流水线。

阅读本文之前，需要首先学习理解开发模式中的分支模式。详细介绍见这里。

如果您的企业在使用云效专有云版，且开启了全局集成功能，请前往这里阅读相应使用说明。

概述

分支模式下的应用的流水线，通常会带有一个“分支管理器”，也就是这里说的应用级特性分支集成视图。分支管理器负责把该应用上用户指定的各特性分支合并到一条发布分支，然后把发布分支上最新的源代码版本，交给流水线作为输入。流水线据此运行。

典型的，一个应用有“日常”、“预发”、“正式”三条流水线，对应日常环境的部署和测试、预发环境的部署和测试、正式发布。每条流水线有其自己的分支管理器。“日常”流水线成功运行完毕，用户可以在分支管理器中把相应的特性分支一起带到“预发”流水线，进而“正式”流水线，已发布到线上。

待集成区与集成区

每个分支管理器，有待集成区（下图红框）和集成区（下图黄框）两个区域：

两个区域，都是特性分支的列表。

待集成区里，是所有已经开发完毕并做了适当检测，可以进行集成和发布的特性分支列表。做集成和发布时，就从这个列表中挑选，哪些合并到当前流程对应的发布分支，以便部署和进一步集成测试。

集成区里，就是是从待集成区中挑出来的，(打算)合并到发布分支，并随后部署到当前流程相应的运行环境的特性分支列表。这个列表，反映的是当前环境中，包含了哪些代码改动。这些改动将被放在一起测试，进而发布上线。

用户仅需在页面上维护这两个特性分支列表，系统将自动完成发布分支的创建和管理，特性分支到release分支的合并等一系列工作。

当前流程对应的发布分支，显示在集成区中。

日常操作

把特性分支标记为可供集成

在本项目的特性分支列表页（从本项目的左侧菜单中，“特性分支”菜单项进入），或者我的特性分支列表页（点击吊顶“我的”，再点击左侧菜单中“特性分支”菜单项进入），每个特性分支左侧，有“提交待发布”按钮。点击可将该分支状态置为“待发布”，也就是说，标记该特性分支已开发完毕并做了适当检测，可以进行集成和发布了。于是，该特性分支就进入了该应用的各流水线上的分支管理器中的待集成区。

挑选特性分支合入发布分支

在待集成区（图中文案“待发布的变更”）勾选打算合并到发布分支的特性分支。然后点击“提交发布”按钮，即把这些特性分支加入到集成区，并开始自动合并工作。合并完成后，自动继续进行构建和部署等。

如果当时集成区里还没有发布分支，那么此时还没有一条发布分支和当前流水线相关联，于是系统会基于master分支创建一条新的发布分支，然后开始合并工作。

如果当时集成区里已有特性分支，那么此时已有一条发布分支和当前流水线相关联，于是系统就会继续使用这条发布分支，开始合并工作。

在合并过程中，除了合入本次新勾选的特性分支，还将逐个检查已在集成区中，也就是曾经合并到发布分支的特性分支，看是否又有更新还不在发布分支上。如果有，将合入发布分支。类似的，master分支也将被检查。以确保，合并完成后，该发布分支上，包含了新合入的各特性分支，曾合入的各特性分支，以及master分支上的最新内容。

如果合并过程中出现了需要人工解决的合并冲突，页面将提示如何人工解决冲突并继续流程。

特性分支和/或master分支有更新后，更新发布分支并部署

若某(几)条特性分支在合入发布分支后，其内容又有了更新（即，又有人向该分支做了git push操作），或者master分支上因为发布上线而有了更新，那么可以点击“重新部署”按钮一键完成：

- 相应更新发布分支，让它包含master分支和各特性分支最新的内容。
- 把发布分支的最新内容，部署到运行环境。

把某个特性分支的内容，从集成中摘除

在集成区该特性分支条目中，点击右侧的“退出”钮，即可实现该目的。

系统将：

- 基于master分支，自动创建一条新的发布分支，并关联到当前流水线。
- 把去除了该特性分支后，集成区中的所有特性分支再次合并到这条新的发布分支。
- 把发布分支的最新内容，部署到运行环境。

因此在效果上，就把该发布分支从集成中摘除了。

把当前集成的全部内容都摘除

点击集成区上方的“退出”钮。于是，集成区被清空。同时，当前流程，不再对应任何一条发布分支。

把集成内容带入另一条流水线

比如，当在日常测试环境的测试结束后，把集成内容带入预发环境进行测试。

每个流水线运行完毕，不会自动触发另一条流水线的运行。如果想把当前流水线的内容带到另一个流流水线，也就是把当前集成区中的所有特性分支带入另一个流水线的集成区，请点击页面上的“进入……”按钮（如果有）。比如“进入预发部署”。

点击“进入……”（如果有）后，系统完成如下操作：

- 将基于master创建一条新的发布分支，并关联到当前进入到的流水线。
- 把前一个流水线对应的发布分支合并到当前流水线对应的这条发布分支。
- 把集成区中所有特性分支合并到这条发布分支。这是为了保证特性分支上的任何更新也都反映到新的发布分支上。

以上，都是针对进入到的流水线（比如“预发”）。原流水线（比如“日常”），没有任何变化：集成区的特

性分支列表、对应的发布分支及其上的内容，都不会发生变化。

正式发布后合并回master

这个工作不是在分支管理器中完成的，而是在流水线上完成的。正式发布流水线中，有一个任务是“合并主干”。运行至此时，将把发布分支合并到master分支，以使得master分支总是代表最新发布版本。

特性分支全局集成视图

前言

本文档适用于云效专有云，分支模式，且您的企业启用了特性分支全局集成视图。目前在云效的阿里云公有云版尚不提供此功能。

阅读本文之前，需要首先学习理解开发模式中的分支模式。详细介绍见这里。

概述

使用特性分支全局集成视图，可以在一个网页中，看到本企业所有已开发测试完毕的特性分支/集（待提交区）、排队等待集成发布的特性分支/集（队列区），以及已合并到发布分支，正在走集成发布流水线的特性分支/集（集成区）。

一个特性分支/集，在开发测试完毕后，提交待集成，于是出现在待提交区。进而提交集成，于是出现在队列区。当条件合适时，将自动进入集成区，在此（与该应用上其他特性分支一起）合并到该应用的一条发布分支，然后发布分支的最新版本开始走该应用的流水线，经过集成测试等一系列步骤后，最终发布上线。

待提交区、队列区与集成区

鼠标点击“我的”->“全局集成”，进入全局集成视图：

视图包括三个区域，都是特性分支的列表。具体来说：

待提交区里，是所有已经开发完毕并做了适当检测，可以进行集成和发布的特性分支列表。做集成和发布时，就从这个列表中挑选，哪些适合去走集成发布流程。挑选好后，点击“提交发布”钮，于是进入队列区。当然，也可以在特性分支/集上，直接“提交发布”，跳过待提交区，直接进入队列区。

在队列区，所有分支按照先后顺序排列。从上到下依次判断，如果某个分支集，它所包含的各应用，目前在集成区都没有正在集成的特性分支，那么它就会自动进入集成区。

在集成区，每个应用，各个特性分支被合并到一条发布分支。随后，发布分支的内容去跑流水线，经过构建、部署、测试等环节，若一切顺利，最终发布到正式生产环境，发布分支被合并回master分支。

用户仅需在页面上向待提交区和队列区增减特性分支/集，系统将自动完成排队集成，特性分支到发布分支的合并等一系列工作。

日常操作

把特性分支标记为可供集成

在本项目的特性分支列表页（从本项目的左侧菜单中，“特性分支”菜单项进入），或者我的特性分支列表页（点击吊顶“我的”，再点击左侧菜单中“特性分支”菜单项进入），每个特性分支左侧，有“提交待发布”按钮。点击可将该分支状态置为“待发布”，也就是说，标记该特性分支已开发完毕并做了适当检测，可以进行集成和发布了。于是，该特性分支就进入了全局集成视图中的待提交区。

也可以在分支集页面中，选中一个或多个特性分支，点击“提交待发布”，共同进入待提交区。

挑选特性分支去排队集成

在全局集成页面的待提交区，勾选打算去排队集成的特性分支/集，点击“提交发布”，于是进入队列区。

此外，还可以一步完成上述“把特性分支标记为可供集成”“挑选特性分支去排队集成”两个步骤：在特性分支或分支集上，点击“提交发布”，于是直接进入队列区。

自动从队列区进入集成发布流水线

这一步无需人工干预，系统会按照算法（见上文介绍）自动将满足条件的特性分支/集带入集成区。于是，每个应用，各特性分支被自动合并到一条从master分支拉出来的新的发布分支，并随后走集成发布流水线，直到发布上线。

如果合并过程中出现了需要人工解决的合并冲突，页面将提示如何人工解决冲突并继续流程。

正式发布后合并回master

这个工作不是在全局集成视图上完成的，而是在流水线上完成的。流水线中，有一个任务是“合并主干”。运行至此时，将把发布分支合并到master分支，以使得master分支总是代表最新发布版本。

机器资源管理

机器资源管理概述

适用场景

云效支持不同类型的运行环境及相应的部署方法。比如，通过阿里云EDAS管理环境资源，并通过EDAS部署，详见部署配置：通过EDAS部署。比如通过阿里云容器服务管理环境资源，并通过容器服务部署，详见部署配置：通过容器服务部署。本文档与这样的场景无关。

云效也支持用户通过可以自定义的部署脚本，将应用程序直接部署到机器上运行。在这种情况下，云效就需要直接管理企业的机器资源，以便可以按部署配置：通过脚本部署中的描述，配置各应用各环境分别使用企业的哪些机器资源，进而把应用部署到那里。本文档描述云效如何管理企业的机器资源。

把已有机器关联到云效本企业

你可以在阿里云上自行购买ECS机器，随后把机器关联到云效本企业。也可以把企业自有机房等其他途径的机器，关联到云效本企业，只要这些机器可以从公网访问。详见把已有机器关联到云效本企业。

通过云效直接购买机器

你也可以通过云效直接购买阿里云ECS机器。这需要：

- 第一步，在企业管理页面**配置授权云效**，让云效以指定阿里云账户的身份完成机器购买等操作。
- 第二步，在企业管理页面**配置ECS模板**，于是云效就知道要购买什么样的机器，做怎样的初始化工作。
- 第三步，在企业管理页面**购买ECS机器**，告诉云效用哪个ECS模板，为本企业购买多少台。

把已有机器关联到云效本企业

企业机器资源的管理

企业管理员，点击右上角设置，进入“企业设置”页面，点击左侧“主机管理”，然后点击添加主机，通过安装agent的方式，将机器加入云效的管理。

按照上述方式，您可以生成一条命令，该命令可以在指定区域的多台机器上反复运行。为了安全起见，您可以设定该命令的有效时间。

用这种方法，您不仅可以把您的阿里云ECS机器纳入管理，还可以把您其他途径的机器纳入管理，只要他的机器可以访问公网。

agent安装依赖说明：

```
#该agent依赖Python2.7，当您的机器上的Python版本非2.7，或是您的机器上缺失了zlib-dev openssl-devel  bzip2-devel包，则请按照如下步骤，首先安装Python2.7：
wget "http://agent-install.oss-cn-hangzhou.aliyuncs.com/Python-2.7.13.tgz"
#下载及解压压缩包：
tar -zxvf Python-2.7.13.tgz
#安装必要的工具包：
#centos/redhat系统使用命令
yum install -y zlib-dev openssl-devel bzip2-devel
#debian/ubuntu系统使用命令
apt-get install -y zlib1g libssl-dev libbz2-dev
#在解压后的路径下执行:
./configure --with-zlib
make
make install
```

安装agent完成后，如果在“机器管理”里没有发现新加入的机器，请参照添加Agent失败FAQ进行排查

添加Agent失败FAQ

我在机器上安装了Agent，但在企业的机器列表中看不到

请按照下列步骤依次排查。执行完每一步之后，请确认问题是否解决，若未解决，请继续尝试后续步骤。

目前agent只支持64位的Linux操作系统。

确认您是否曾经使用另一个企业的agent安装命令在该机器上执行过，如果是，请删除/usr/sbin/staragent_sn（正常情况下该文件内容为机器SN，SN为机器唯一标识，并与特定企业绑定），并重装agent。

执行命令cat /usr/sbin/staragent_sn查看内容。若文件内容为空，删除此文件，并重装agent（PS：请勿手动修改该文件）。

执行命令/home/staragent/bin/staragentctl status查看agent状态。若输出异常（比如

ServerAddr为空，或者报错），请执行命令cat /home/staragent/conf/staragent.conf查看文件内容，如果文件存在，且其中的URL为rdc-xxx.aliyuncs.com或者staragent-configservice.aliyuncs.com，则为正常。如果不是，有可能是您的机器之前安装过其它产品的agent，请重装云效agent。

查看cat /home/staragent/conf/channels.conf是否存在，如果不存在，请执行命令：curl 'http://<从staragent.conf中获取的服务URL>/api/configservice?action=findChannelListForAgent&agentIpList=101.37.119.155%2C10.80.237.52&needAllChannels=true&serviceTag=ea263ff8-2d60-48f0-86c4-33a04214cad9&version=2'，如果结果类似{"appCode":"_successful_","msg":"","restCode":200,"result":{"allChannels":[],"channelIPPort":[{"ip":"100.100.18.88","port":8000},{"ip":"100.100.18.89","port":8000},{"ip":"100.100.45.99","port":8000},{"ip":"182.92.29.36","port":8000},{"ip":"182.92.29.39","port":8000},{"ip":"100.100.45.100","port":8000}]，请尝试重启agent（参看下面的agent基础操作）。如重启后/home/staragent/conf/channels.conf仍不存在，请点击云效页面右下角“提问”联系我们。如果不能返回类似结果，则表示您的机器到<从staragent.conf中获取的服务URL>的连接有问题。有可能是在安装agent时候，选错了区域。请在添加机器页面选择正确的区域，生成agent安装命令，重装agent。

执行命令cat /home/staragent/conf/channels.conf查看该文件，内容会是一个ip+port的列表，尝试运行telnet <ip> <port>，只要任意一个连通，则服务正常；如果全部不通，请检查您的网络。

EDAS的agent与云效的agent不能共存。如果您的机器上安装了EDAS的agent，请彻底卸载，或重置操作系统，再尝试安装云效agent。

附agent基础操作：

```
启动：/home/staragent/bin/staragentctl restart;
重启：/home/staragent/bin/staragentctl restart;
查看状态：/home/staragent/bin/staragentctl status;
卸载：
1. /home/staragent/bin/staragentctl stop;
2. rm -rf /home/staragent;
3. rm /usr/sbin/staragent_sn
```

导入机器失败

```
error info:Forbidden.RAM : User not authorized to operate on the specified resource, or this API doesn't support RAM. RequestId : xxx
```

解决方案： 当前用户可能为子账号 ，无权操作对应的ECS，需要主账号授权后继续导入，或者使用主账号导入

若根据以上排查手段依然未能找到问题，请点击右下角“提问”联系我们。

配置授权云效

概述

作为一站式研发协同平台，云效在很多场景下需要和阿里云的其它云服务进行交互，比如ECS购买、运维相关的云服务集成等。云效以指定的阿里云账户身份，通过Open API调用，完成上述操作。为此，在云效本企业中，企业管理员需要先做适当的授权配置。本文详细讲解如何进行这样的授权配置。

添加授权并绑定

确定付费账户

请确定一个阿里云主账户。云效将以这个账户的身份购买机器。

该阿里云主账户满足下述两个选项之一：

选项一，该阿里云主账户本身是云效用户，且是云效中本企业的企业管理员。

选项二，该阿里云主账户的某个RAM子账户是云效用户，且是云效中本企业的企业管理员。该RAM子账户还需要满足一个条件：它具有为该阿里云主账户创建RAM角色的权限。一般来说，这通过在RAM控制台中，向该RAM子账户（或其所属组）添加系统授权策略“AliyunRAMFullAccess”来完成。相关知识请参阅RAM授权策略管理帮助文档。

授权

这一步的目的是，授权云效，可以用该阿里云主账户的身份，完成购买机器等操作。这一步完成后，云效就有此权限了。至于在云效的本企业中，是否用此权力，将在下一步配置绑定。

请以上述阿里云主账户或其RAM子账户登录云效本企业，从页面右上角齿轮图标处进入“企业设置”，选择“主机管理”，进而选择“授权”，即进入授权配置页面。

在授权配置页面的“我授权并绑定”区域，点击“授权”，并在随后页面中点击确认，即可完成。

原理：授权意味着，在阿里云RAM服务中，为该主账户添加了“AliyunRDCDefaultRole”这个角色并赋予其授权策略“AliyunRDCRolePolicy”，允许云效通过该角色完成操作。可前往RAM控制台中，该账户的RAM角

色管理页面查看。相关知识请参阅RAM角色帮助文档中，“服务角色”相关内容。

错误提示与解决办法：如遇弹窗提示“权限不足”，说明该RAM子账户没有为其阿里云主账户创建RAM角色的权限。请参考上文“确定付费账户”->“选项二”解决。

绑定

这一步的目的是，告诉云效，当云效中本企业要进行购买机器等操作时，以该阿里云主账户的身份完成。

在授权配置页面（进入方法见上文）的“我授权并绑定”区域，点击“绑定”，即可完成。

修改绑定

当本企业已绑定某个阿里云主账户后，可以修改为绑定另一个阿里云主账户，同时解除原绑定关系。过程与初次配置授权时相同：

- 确定要更换到的阿里云主账户。
- 若尚未授权，以该主账户登录云效本企业，在授权配置页面的“我授权并绑定”区域点击“授权”。
- 在授权配置页面继续点击“绑定”。

解除绑定

可以解除当前本企业与某个阿里云主账户的绑定关系。方法为，企业管理员在授权配置页面的“解除绑定”区域，点击“解除”。

账户与企业解除绑定关系后，若希望亦去掉授权（即云效以该账户名义操作的权力），可前往RAM控制台中，该账户的RAM角色管理页面中，删除“AliyunRDCDefaultRole”角色。相关知识请参阅RAM角色帮助文档。

配置ECS模板

概述

云效为了方便企业管理员添加相同云效环境的机器，提供了ECS镜像模板功能，企业管理员可以按照语言类别维护ECS镜像模板，通过ECS镜像模板购买相同运行环境的ECS。

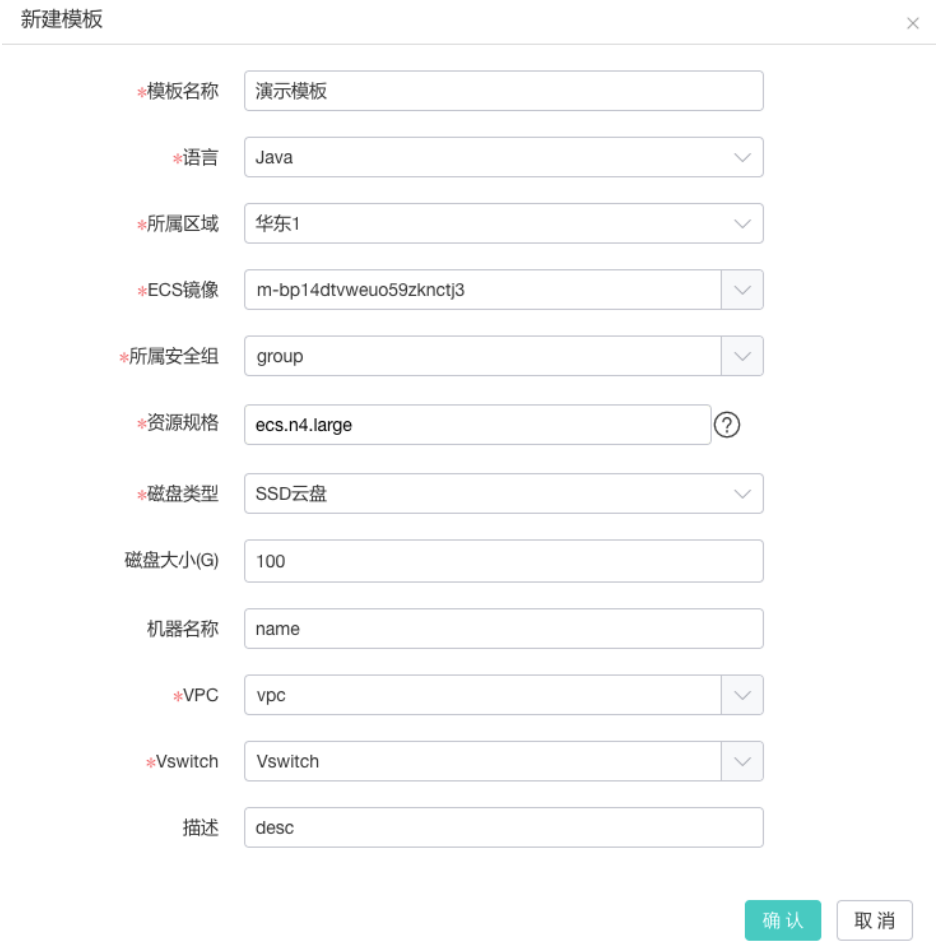
使用限制

- 只支持阿里云云服务器ECS
- 只支持VPC网络
- 已经创建ECS自定义镜像
- 已经创建VPC和交换机（Vswitch）
- 已经创建安全组

镜像维护

企业管理员，点击右上角设置，进入“企业设置”页面，点击左侧“主机管理”，然后点击“ECS镜像模板”。

确保已经完成**配置授权云效**，点击“新建模板”来添加ECS镜像模板



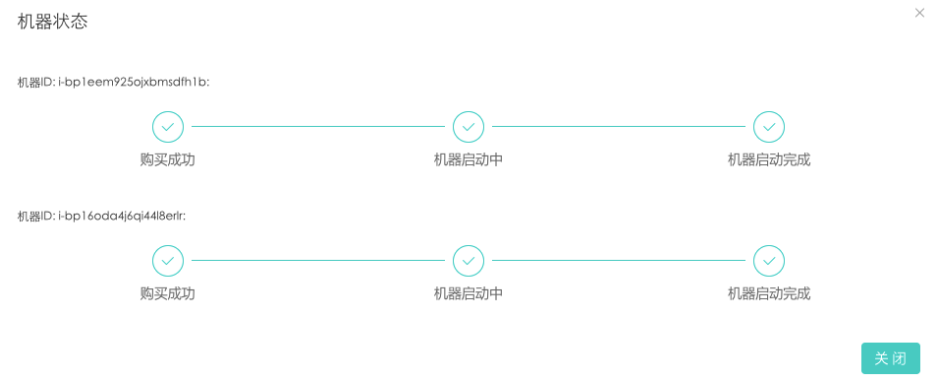
购买ECS机器

概述

云效为用户提供购买阿里云云服务器ECS功能，在使用之前，请确保先完成配置授权云效和配置ECS模板。在购买ECS完成后，云效自动启动ECS，并安装agent，直接加入到企业的机器管理列表中。

企业管理员，点击右上角设置，进入“企业设置”页面，点击左侧“主机管理”，然后点击“购买机器”购买ECS。

购买成功后，会弹出云服务器ECS启动流程。



更多关于ECS的操作，请参考ECS的帮助文档：
https://help.aliyun.com/product/25365.html?spm=a2c4g.750001.list.2.1dd87b13D2T3Xs

私有云部署

概述

云效新增私有云部署功能，支持企业私有云或者自建机房等不能访问公网机器部署。该方案只需要一台能访问公网机器作为部署代理机，其他部署应用的机器不能访问公网（如果能访问公网，务必请使用传统方式添加机器）。

快速开始

第一步：维护代理机

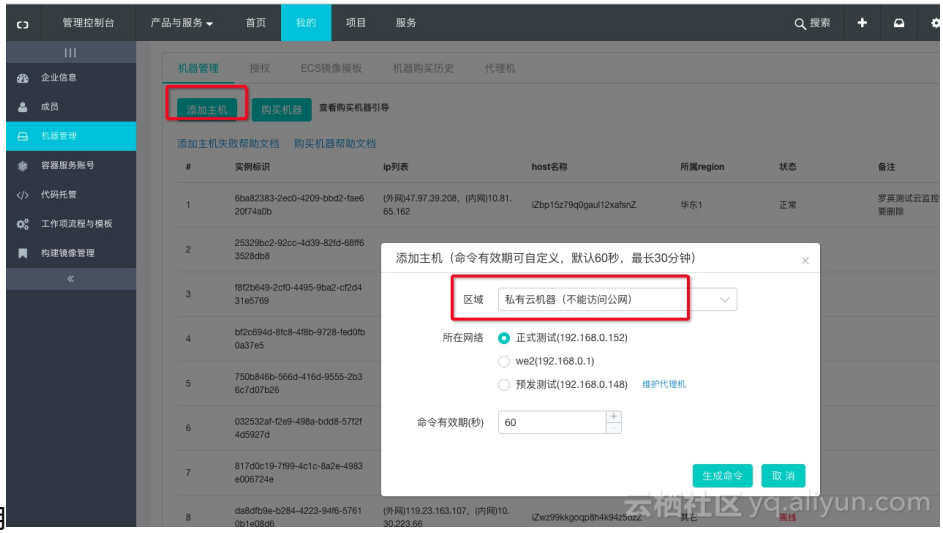
企业管理员，点击右上角设置，进入“企业设置”页面，点击左侧“机器管理”，然后点击“代理机”,维护代理机信息。

第二步：设置代理机

代理机安装docker，启动docker，拉取代理镜像（docker pull registry.cn-hangzhou.aliyuncs.com/rdc-product/private-idc-proxy:5.0），确保80，8000，443端口没有被占用，启动镜像（docker run --restart=always -p 8000:8000 -p 80:80 -p 443:443 -d registry.cn-hangzhou.aliyuncs.com/rdc-product/private-idc-proxy:5.0）。

第三步：添加主机，

选择私有云部署，生成安装命令，在不能访问公网的部署机器上安装agent，安装成功后，就可以进行部署了



。立即使用

FQA

添加机器安装agent后，在机器列表里不能展示

检查文件/home/staragent/conf/channels.conf是否存在，内容是否类似格式如下（注意，proxy机器的ip和8000之间需要是一个tab，而不能是空格）

```
<代理机器的ip>    8000
```

部署失败

检测/etc/hosts,确保已经添加了如下内容

```
<代理机器的ip> aone-build-bejing.oss-cn-beijing.aliyuncs.com
<代理机器的ip> execution-component-rdc.oss-cn-beijing.aliyuncs.com
```

测试环境管理

云服务集成

概述

为什么要在云效中集成云产品

阿里云提供了大量的优秀的云产品，比如ECS，SLB，云监控，日志服务，帮助用户进行线上服务的部署，运维，监控，告警。

但实际用起来之后，会发现一个问题。那就是有些概念，比如机器分组，会在多个产品中重复实现。假设有一个线上的Web应用，包含了5台机器。那么需要在日志服务中将这5台机器配置到一个分组，然后再在云监控中把同样的5台机器分到云监控的分组，再把这5台机器挂在某个SLB下。当应用扩容一台机器时，各个云服务的机器组也需要手工同步。造成这种不便的原因就是缺乏了一个基础的公共概念：应用。

云效作为研发协同平台，以应用为核心的。应用下面有不同的环境，每个环境对应一个机器组，使用环境的概念，就可以将各个云产品的机器组的概念统一起来。通过Open API的方式，云效可以在环境机器有变化时，把上述的这些相关服务配置好。同时应用也会成为一个访问其他各个云产品的聚合入口，从而更好的将研发过程和运维过程有机的结合起来，助力企业的DevOps转型。

使用的前提

授权云效

云效通过阿里云Open API和其它产品进行集成。通常企业会使用统一的账号来进行进行各种云产品的购买和管理，因此在使用云产品集成功能之前，需要先使用该云账号对云效进行授权。后续对这些云产品的操作都会使用以该账号的身份进行。

已集成的云产品

目前开放的云产品集成：云监控。

后续会陆续集成更多的云产品，也欢迎您在页面底部的“以上内容是否对您有帮助”环节进行评分，进而将您的需求反馈给我们。

云监控

概述

阿里云的云监控可以对一组ECS机器进行基础监控。对于Web服务来讲，每个ECS分组本质上对应的是一个应用的一个环境下的机器。在关联云监控后，云效会自动同步环境下ECS到相应的云监控分组，并自动安装云监控agent。环境关联或删除机器，都会同步维护对应的云监控分组。

在使用之前，请确保先完成配置授权云效。

使用限制

- 只支持阿里云云服务器ECS
- 只支持VPC网络

开启方式

开发人员，点击相关应用，进入“环境”页面，点击“云服务”。

点击开启radio button。如果授权账户下尚不存在云监控分组，则云效会直接弹出新建云监控分组的对话框。

填入期望的云监控分组名称和告警联系人分组。点击确定之后，云效就会使用授权的云账号创建相应的云监控分组，并把当前环境下的机器添加到该分组中。如果您的机器尚未安装云监控的agent，云效也会自动进行安装。

如果授权账户下已经存在云监控分组，则点击开启radio button之后，您可以在下拉框中选择已有的分组，或者创建新分组：

如果您选择关联已有的分组，云效只会根据该环境下机器的增删操作，对该云监控分组进行同步，不会影响该分组已经存在的机器。

去除关联

如果您不再希望云效对云监控分组进行同步，请点击关闭radio button，然后点击右上角的全部保存。则云效会停止环境内机器到云监控分组的同步操作。

功能限制

- 云效新建云监控分组后，不会关联告警模板到该分组，需要您到云监控控制台手动进行关联。

测试管理

测试用例与测试计划

RDC提供测试用例和测试计划的功能，用于帮助开发者管理和执行手工用例，针对现在测试更加轻量快捷的特点，提供了以下功能：

测试用例用于管理和组织手工用例，支持方便快捷编辑和查看用例。

测试计划于规划和执行手工用例。测试计划支持任务流概念，可以方便进行测试的评审。

- 发现方便创建和关联缺陷，并提供全面的测试报告分析

一. 测试用例管理

用例集

用例集是组织用例的方式，用于对用例进行分组。用例集支持嵌套用例集。

用例

名称：用例的一个简短描述。限定在100字以内。如果名称描述不清楚用例，请在描述中继续写。

创建人：创建测试用例的人。克隆别人的用例不会更改作者。当然，作者是可以更改的，使用批量修用例信息

，功能可以实现修改作者。

步骤：用例的具体操作步骤。

注释：对用例的补充说明。

优先级：用于标识用例执行的优先程度，可选值：P0，P1，P2，P3，默认P3。

用例的导入和导出

Excel导出

1. 选择某个测试集。选中后，将会导出该测试集下（包含子测试集）所有用例。
2. 点击Excel导出。

脑图（mm、xmind)导出

1. 选择某个测试集。选中后，将会导出该测试集下（包含子测试集）所有用例。
2. 点击MM导出。

Excel导入

导入模板如下：

编号	名称 必填	产品名	测试集	作者	优先级	标识符	步骤	注释	时间
516	桌面反复拖动分组内图标	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
518	连续点击应用图标启动应用后置于后台	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
520	连续左右滑动屏幕	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
522	连续安装并卸载应用	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
524	连续点击应用图标启动应用并退出	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
526	反复调用Widget列表	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26
528	桌面反复拖动图标操作	OS事业群/BugFree	桌面 for CB 2.9.1	果薇	P2				2015-11-26

脑图（mm，xmind导入

导入全面兼容mm，xmind格式。格式说明见：

默认将叶子节点作为用例节点，其他节点作为测试集

> 如果要将指定的节点作为用例节点则在用例名称之前添加关键字：TC或tc。此时该节点会被识别成用例节点，该节点的直接子节点被识别成步骤。再往后的子孙节点会被忽略。

> 如果需要添加优先级，则用例名称为tc:pn_casename 其中“n”为优先级（0，1，2，3）

复制、移动，引用已有用例

复制用例集

用例支持在用例集列表上进行用例集的复制，右键点击测试集，支持测试集复制。

从其他项目引用用例

右键点击测试集，选择引用。

移动测试集

支持通过拖拽测试集实现测试集的移动。

回收站

用例支持回收站，删除的用例会放在回收站中，可以在回收站对删除的用例进行拖拽恢复。

用例标签

【待补充】

二. 测试计划

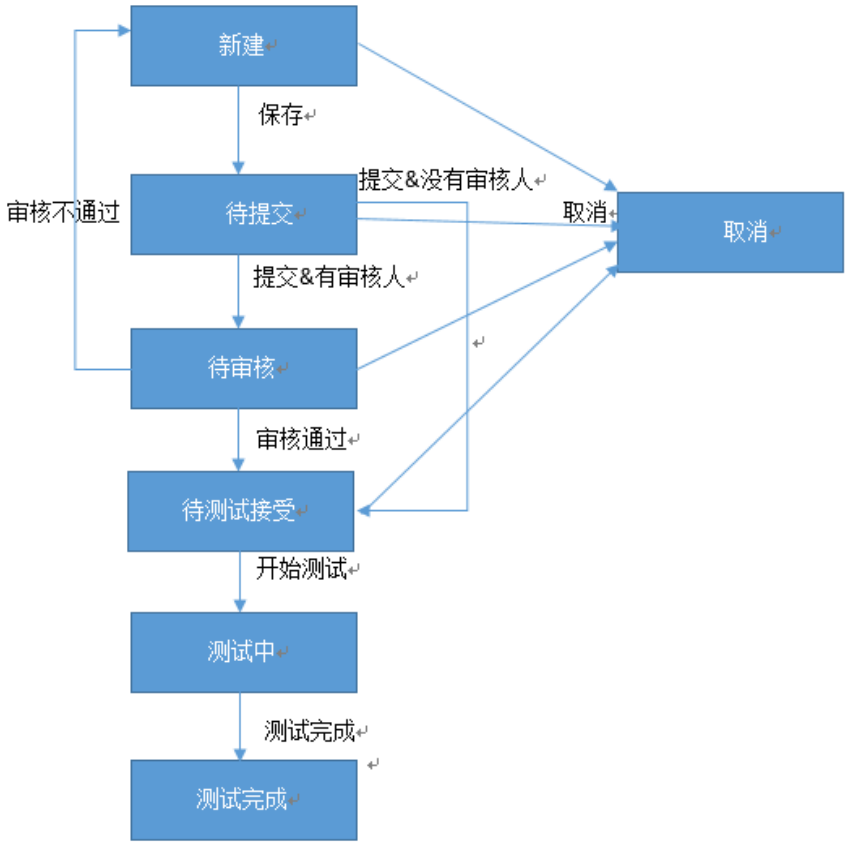
用例加入测试计划才能执行，测试计划是用来规划一次测试过程的载体。

新建测试计划

新建测试计划，可以将这个测试计划需要执行的用例加入计划，用于标识这个测试计划需要执行的内容。

测试计划流程

测试计划流程如下图：



一些规则

1. 计划所属的项目成员和指派人可执行和修改计划
2. 计划在审核中，不能进行修改

执行用例关联缺陷

用例执行时，如果发现缺陷，可以直接关联或者新建缺陷。

测试结果查看

如图，测试结果会展现测试进度，测试通过率和测试状态。

测试报告

三. 使用技巧Q&A

误删除用例如何恢复

所有删除的用例都是逻辑删除，放在了回收站内，可在回收站中恢复。

用例如何过滤？

如图 

测试计划中编辑用例会在全局生效吗？是否还要同步？

是的，全局生效，不需要同步。

阿里巴巴代码规约检测

《阿里巴巴 Java 开发手册》是阿里巴巴集团技术团队的集体智慧结晶和经验总结，经历了多次大规模一线实战的检验及不断的完善，系统化地整理成册，反馈给广大开发者. 阿里巴巴 Java 开发手册检测的能力也被集成在RDC的自动化测试服务中，可以直接对代码进行扫描以检测室是否符合阿里巴巴代码规约。

代码扫描支持 - 全量扫描和增量扫描

1. 阿里巴巴代码规约检测全量扫描通过RDC对Java代码工程进行编码规约全量检测优点： 支持跨文件引用， 代码扫描全面缺点： 但扫描速度较慢，问题量会比较多
2. 阿里巴巴代码规约检测增量扫描是基于代码的一次push，自动获取diff内容，对diff文件用编码规约规则进行扫描，并过滤出此次提交产生diff规约问题功能。（目前增量扫描功能即将上线，尽情期待）优点： 只扫描diff文件，扫描速度很快，增量问题直接关联到人，能有效防止代码提交引进新问题数。缺点： 因只扫描diff文件，不能发现跨文件引用出现的规约问题。

怎么使用RDC进行阿里巴巴代码规约检测

1. 通过”测试”服务中“阿里巴巴代码规约检测”创建扫描任务

新建代码规约扫描，填入要扫描的地址，开启代码扫描

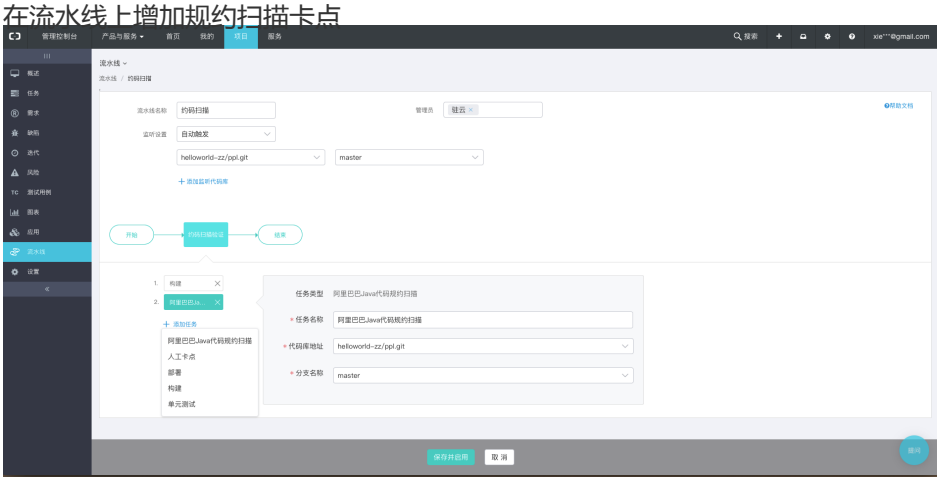
2. 配置代码提交或者定时触发“阿里巴巴代码规约检测”

功能待上线

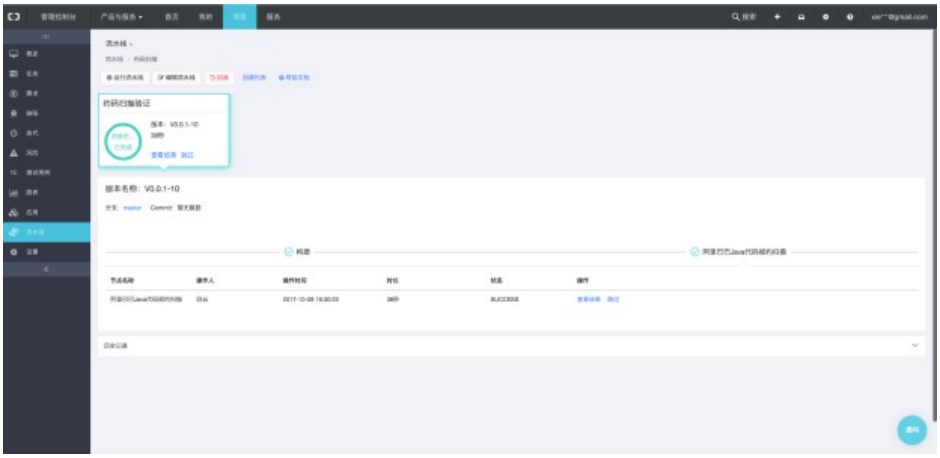
3. 发布时进行代码规约扫描并进行发布卡点

RDC支持在发布流水线上配置阿里巴巴代码规约扫描（支持全量和增量扫描），可以将阿里巴巴代码规约扫描组件加入流水线， 并设定相应的通过条件，就可以支持代码规约扫描卡点。

在流水线上增加规约扫描卡点



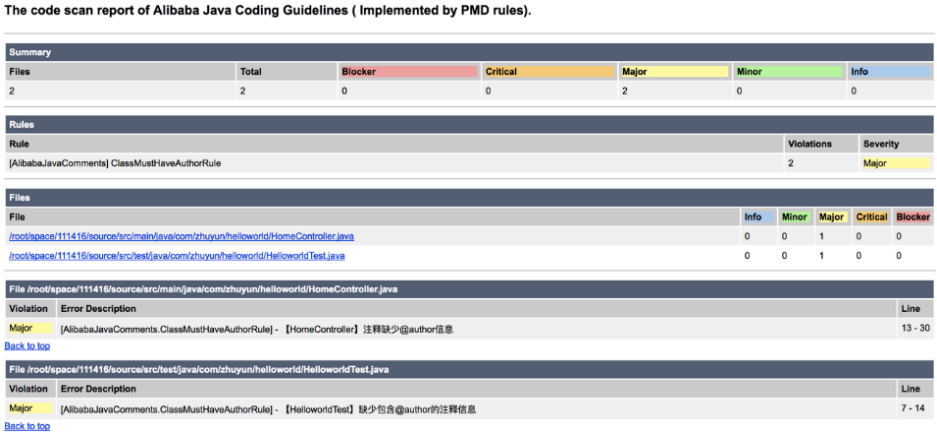
规约扫描结果在流水线上展现



查看扫描结果

执行完代码扫描任务后，可以看到相应的执行结果。

点击问题数可以查看具体问题以及相应解决方法。



The code scan report of Alibaba Java Coding Guidelines (Implemented by PMD rules).

Summary

Files	Total	Blocker	Critical	Major	Minor	Info
2	2	0	0	2	0	0

Rules

Rule	Violations	Severity
[AlibabaJavaComments] ClassMustHaveAuthorRule	2	Major

Files

File	Info	Minor	Major	Critical	Blocker
/root/space/111416/source/src/main/java/com/zhuyun/helloworld/HomeController.java	0	0	1	0	0
/root/space/111416/source/src/test/java/com/zhuyun/helloworld/HelloworldTest.java	0	0	1	0	0

File /root/space/111416/source/src/main/java/com/zhuyun/helloworld/HomeController.java

Violation	Error Description	Line
Major	[AlibabaJavaComments.ClassMustHaveAuthorRule] - 【HomeController】注释缺少@author信息	13 - 30

Back to top

File /root/space/111416/source/src/test/java/com/zhuyun/helloworld/HelloworldTest.java

Violation	Error Description	Line
Major	[AlibabaJavaComments.ClassMustHaveAuthorRule] - 【HelloworldTest】缺少包含@author的注释信息	7 - 14

Back to top

无线测试接入RDC发布流程

MQC接入RDC发布流程步骤概述

通过实验室可以在RDC发布流程中，把无线构建打包后自动的触发MQC，完成移动测试的自动验证。配置如下：

1. 在实验室创建无线测试任务
2. 在发布流程增加实验室流程验证点

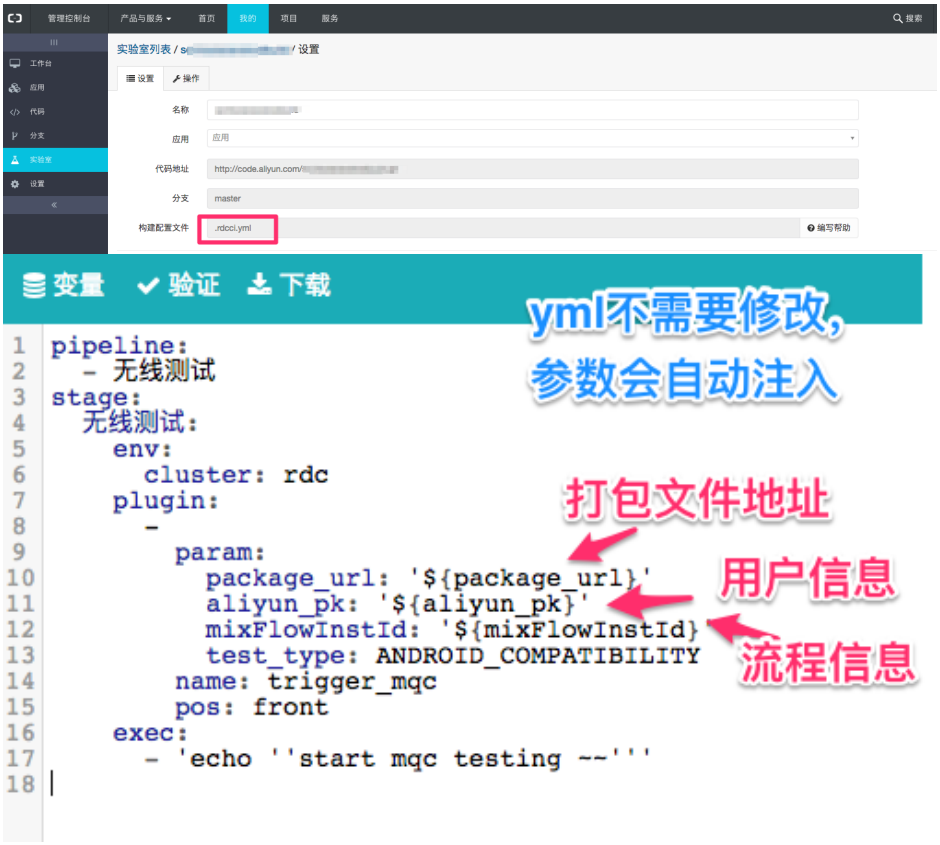
在实验室创建无线测试任务

1. 将一下内容保存到代码地址根目录下**.rdcci.yml**

```
pipeline:
- 无线测试
stage:
无线测试:
env:
cluster: rdc
plugin:
-
param:
package_url: '${package_url}'
aliyun_pk: '${aliyun_pk}'
mixFlowInstId: '${mixFlowInstId}'
test_type: ANDROID_COMPATIBILITY
name: trigger_mqc
pos: front
```

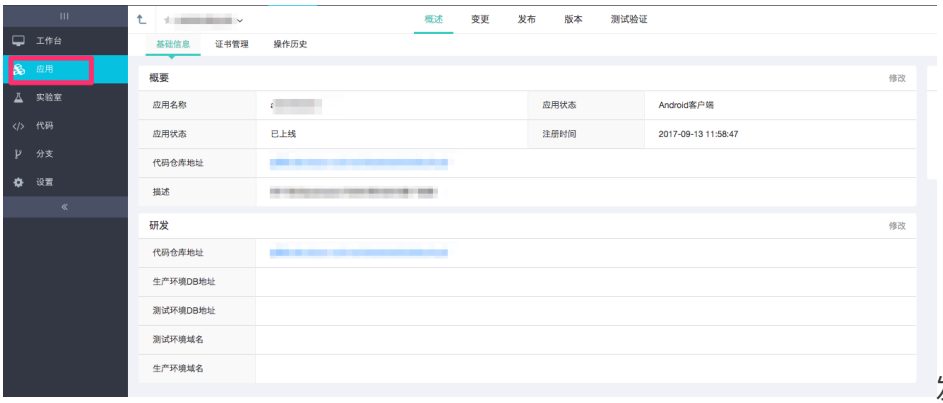
```
exec:
- 'echo ''start mqc testing ~~'''
```

2. 在实验室创建对应代码地址任务



在发布流程增加实验室流程验证点

1. 准备好你的应用和发布流程


发布流程

1. 增加发布流程的测试验证

1. 配置对应实验室任务

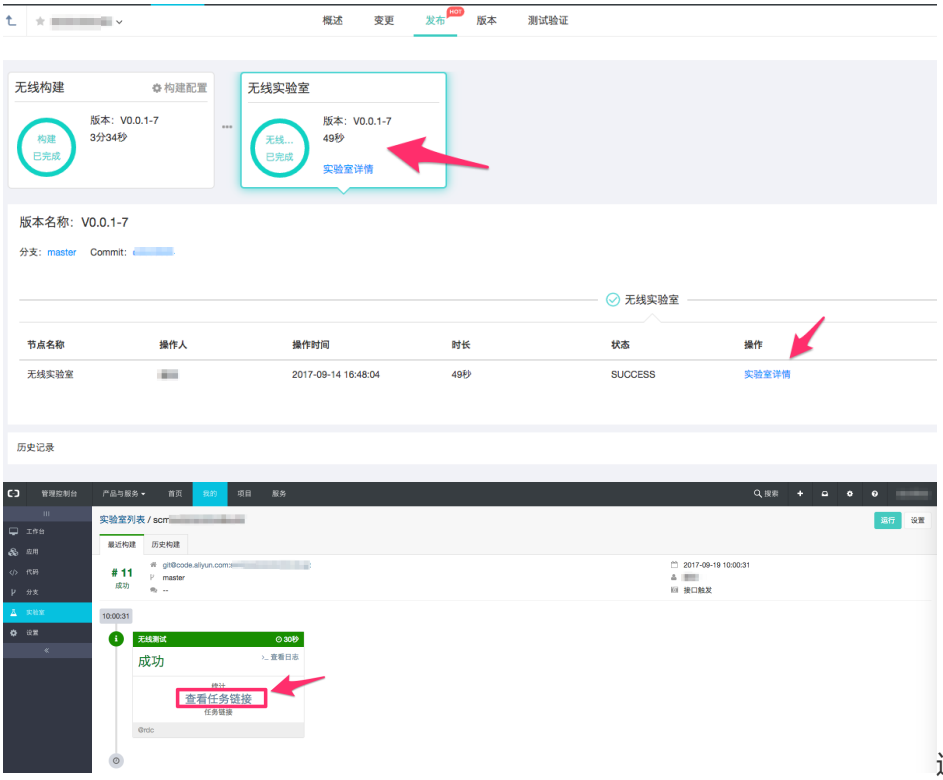
到这来我们的配置就完成了

，下面可以看一下实际运行的效果

测试验证效果一览

1. 发布流程的无线构建完成，点击进入无线测试

1. 这里可以看到测试任务运行结果，通过连接可以看到实验室的任务详情



这里可以连接到mqc执行详情页查看移动测试结果详情

附：MQC 无线测试

移动测试（MQC）是为广大企业客户和移动开发者提供真机测试服务的云平台

http://mqc.console.aliyun.com/

持续交付和质量红线

持续交付和质量红线

自动化测试保障持续交付质量

RDC提供了完备的Pipeline, 在整个研发过程开发代码提交后自动触发单元测试, 静态代码扫描。应用发布打包,部署, 自动触发集成测试, 构成了开发和测试共同参与的一套流水线. 在持续交付的实践中, 这样的做法可以有效的加快开发测试效率, 以最小的成本, 找到代码中的错误, 保持代码的质量平稳, 发布周期可预。

RDC持续交付提供质量验证卡点

验证卡点是用于保障交互质量的重要手段，为了达到持续交付的目标，我们建议通过分层测试和测试卡点通过才继续流转的方式来保障整个持续交付的顺利进行

1）代码提交后自动运行相应的实验室：系统自动监控代码提交事件，分析代码的变更情况（变化的Java文件，类，方法），自动执行测试任务。为保障开发分支的质量，这个阶段我们推荐用户配置单元测试和静态扫描。

2）在应用部署后，会根据不同的发布阶段分别流转日常阶段、预发阶段、线上阶段的测试任务执行，用于验证相应阶段的质量。

- 日常测试阶段：将分支合并主干后部署到日常环境，关联的实验室任务可以按照自己的需求进行配置；
- 预发测试阶段：将日常部署分支部署到预发环境，关联的实验室任务可以按照自己的需求进行配置；
- SIT集成测试验证：将预发部署分支部署到线上环境，关联的实验室任务可以按照自己的需求进行配置；
- 线上集成验证：将预发部署分支部署到线上环境，关联的实验室任务可以按照自己的需求进行配置；

通过实验室创建自动化测试任务

RDC是通过实验室来实现自动化测试任务，如何使用请查看“自动测试和集成”

在Pipeline上配置验证卡点

应用的测试验证配置页面配置Pipeline的测试任务和通过条件

1. 点击“添加验证点”，添加每一验证阶段关联的实验室任务。
2. 通过条件就代表对应阶段运行的标准, 默认通过条件是任务成功，但也可以添加单测成功率, 覆盖率, 静态扫描缺陷等条件来控制是否通过。

双引擎自动回归服务

失败case排查

回放失败排查思路

- 在失败case中查看是否提示“子调用 xxxxx mock 失败”,如果是，则进入后文的mock失败排查。
- 如果未提示子调用mock失败，但是出现了对比失败，则观察对比不一致字段进行对比排除及问题分析。
- 如果是**非http**接口，如果对比不一致字段中回放的response没有正常返回或者返回了null
    - 那么请查看 “录制&回放数据内容查看”中的异常信息，根据异常堆栈信息在业务代码中定位发生异常的原因，如下图位置：

- 如果是http接口，返回的 response_status 不是 200
    - response_status 502，说明服务端处理异常，需要看下业务日志或者debug代码排查原因。可能是录制参数不对或者其他。
    - response_status 302，说明是被登录跳转了，请参考，设置session mock。即将session相关读取设置为子调用，同时mock掉一些校验机制保证回放正常执行。
- 如果发现录制有子调用列表，回放没有任何子调用信息，且未提示“子调用 xxxxx mock 失败”。遇到此情况说明回放时应用流程执行失败了。那么怎么办呢？
    - 首先定位回放业务失败原因
        - 先观察回放的response，看看有没有异常码。若有，则从代码定位下出现异常原因。
        - 如果response无法分析原因，那么观察业务日志，看是否在回放时有业务异常，分析业务执行失败原因。
        - 如果没有发现系统异常，只能通过触发回放，让后debug回放流程，分析下业务执行失败原因。
    - 根据原因找到解决方案

- 如果是录制回放配置不一致倒是回放流程失败，考虑启用用例级diamond mock.
- 如果是sesson过期原因，请mock session。
- 如果是业务的特殊逻辑导致回放失败，请mock调特殊逻辑。
- 如果是本地缓存导致回放失败，请mock 本地缓存读接口。

子调用排查MOCK失败排查

日志中提示：’ 子调用 xxxxx mock 失败，从doom失败case页面可以看到如下提示：

上面截图报错是tair的读请求没有正常mock。有同学可能不理解：为什么已经配置了隔离tair为什么还要报这个错？有此疑问的同学对doom中的‘隔离’的含义没搞明白，隔离的含义是：大部分中间件隔离相当于对通过该中间件的请求进行子调用处理，即录制时记录请求入参和返回值，回放时用录制结果mock。如果录制时没调用到此中间件，回放时就没办法mock，所以会出现子调用找不到异常。

解决这个问题要分2中场景：

- 应用没有重构场景
- 应用因为重构，应用新增了一个录制时没有的子调用

p.s. 部分场景下录制次数少于回放次数也会报同样问题，这种场景同样适用如下分析排查方法。

场景一：应用没有重构

原因

原因是录制时线上没有走到该子调用，导致该子调用在录制时没有被录制到，而在回放时走到了这个子调用，导致回放失败。这句话很拗口，举个具体例子：假设有一个读取用户信息的服务逻辑如下：

```
...

User user = cache.get(id);
if(user==null){
user = userService.get(id);
```

```
}
...
```

假设过程如下：

1）录制机器命中缓存，那么userService.get(id)将不被调用。
2）当回放该条流量时，如果 cache没有缓存该用户信息，因此，user==null成立，回放机器会尝试调用 服务userService.get(id)获取子调用。
3）如果做了hsf隔离，那么doom会认为userService.get(id)是一个子调用，尝试从录制的上下文中查找这个子调用来mock，but由于录制没走到这个流程，doom会报错，此时回放失败。

解决方案

将 cache的get方法设置为自定义子调用拦截器可解决此问题。

其他类似问题还包括：

录制回放机器开关配置不一致导致的子调用找不到回放失败问题。

方案：

- 通过修改配置确保配置项一致。
- 使用同环境回放。
- 使用自定义mock屏蔽环境配置不一致。

排查过程：

- 检查是否由于缓存的原因导致录制走了缓存而线下没缓存而发生了调用则有两种处理方法：
- 将缓存查询作为一个子调用可解决。
- 将子调用的上层调用配置为子调用，上层调用涵盖了缓存查询的流程。
- 如果是因为beta机器本身的重构导致新增了一些调用，那么可以通过扩展插件解决。自定义接口里面可以采取自己构造返回值或者是读接口直接放开。
  读接口放开也可以通过 [客户端配置 -> 动态配置 -> 忽略子调用MOCK]进行配置化解决。
- 请保证线上配置和线下配置一致后再回放。
- 某些上下文信息是在埋点前的框架或者应用流程中设置的，导致回放是没有这些上下文参数。
- 将上下文读取方法作为一个子调用进行配置来解决。
- 如果是新增的一个子调用，那么流量要重新录制，确保该子调用被录制到。

场景二：应用有重构

参考 重构场景下如何解决子调用mock问题

DOOM使用常见问题

不能正常录制回放怎么办

录制不到流量怎么办？

1.排查doom客户端是否部署成功。
2.检查录制任务是否勾选了待录制的流量接口。
3.日志级别调到debug【客户端配置-动态配置-日志级别】，观察doom.log,若doom已升级到6.2.0，开日志查看器进行日志查看入下图：
图1：查看日志入口

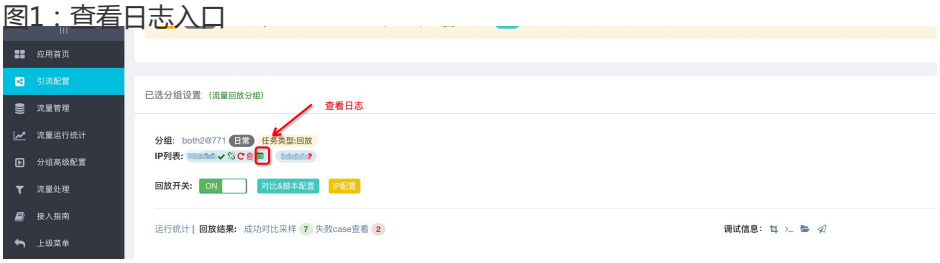


图2：排查流量为啥没录制到

- 一般来说，当前采样率为0，要么是任务没启动，要么是任务启动后没勾选该接口，请检查任务配置。
- 若isRun始终没有出现你关心的接口，请检查是否真正有流量落到这台服务器。如果是java流量入口，检查任务中是否添加该入口。

接入doom正常但不能回放怎么办？

- 排查doom客户端是否部署成功。
- 请检是否录制到流量。
- 是否开启了回放任务。

回放机器如何识别是否需要mock，mock的子调用是哪些？

是否mock和客户端配置有关。在【客户端配置-隔离的中间件】配置的中间件默认会进行mock或者隔离。例如dubbo,db等，dbmock需要依赖orm层做mock，自定义子调用也会进行mock。

子调用找不到、回放失败怎么办？

参考文档 失败case排查

应用重构如何使用doom回归

应用重构支持

子调用不可序列化

自定义子调用需要保证可序列化，若不可序列化请确认是否子调用设置不合理，例如aop切面入参带有Method类的以及httpFilter都不应该设置为自定义子调用。若一定要设置为子调用参考如下方法解决：

1.子调用中入参无法序列化的解决办法

例如：doom提示
：SubInvoke[com.epp.fincloud.service.ItemSelectionService@modifyPriceBySubChannel(ItemPriceUpdateDTO,String,List)]@[第1个入参com.epp.fincloud.dto.ItemPriceUpdateDTO无法序列化]}

如果排除
com.epp.fincloud.service.ItemSelectionService@modifyPriceBySubChannel(ItemPriceUpdateDTO,String,List) 这个子调用的第一个入参进行序列化，那么便能解决序列化问题.

配置方法：在【客户端配置->子调用配置->子调用入参忽略序列化配置】选项中添加【com.epp.fincloud.service.ItemSelectionService@modifyPriceBySubChannel(ItemPriceUpdateDTO,String,List)#0】来解决

2.在更上层可序列化的位置进行mock

3.通过插件自定义序列化

4.忽略类序列化

可将不可序列化类设置为【客户端配置-其他配置-**忽略序列化类列表**】中忽略序列化，忽略序列化后该类在回放的时候直接会被设置为null,若回放强依赖该值，那么不能采用此方法。

入参或返回值不可序列化

1.分析是否主调用配置是否合理，例如http流量直接通过http接口接入解决而不应该拦截java的controller类。
2.通过自定义序列化、忽略类序列化来解决。

doom.log为什么会出现InvalidInvokeException disconnect.invoke—> xxx 异常？（doom警告页面会提示：” 回放机器在非回放状态下执行子调用操作…” ）

【原因】
doom客户端会对‘回放分组’机器的隔离中间件做隔离【即纯回放机器不不能正常提供服务，只能用于回放】，隔离的目的是保证调用最终不会发生。例如一个写db请求，一个hsf调用等等。是为了确保回放不影响线上数据。出现这个异常一般是由于非回放流量尝试执行中间件的调用。例如beta机器接收到了http请求（我们要求测试机器要摘除vip），另外一个可能是应用本身的定时任务或者其他的调度任务。

问题一：导致应用无法正常启动

【如何解决】

- 方法1.设置客户端延迟启动，避免在启动过程中被disconnect。设置方法：【客户端配置-静态配置-基础配置-**客户端延迟启动时间**】设置的长一些。

  方法2.对于线下回放，可以使用录制&回放分组进行回放，避免隔离。

问题二：持续不定刷 disconnect异常日志

【如何解决】常用自定义断开设置

如果的确不希望回放机器正常提供服务，请做如下检查
如果抛出这个异常的入口是http请求，请摘除vip，不要让http请求进入到beta机器。
注：预发域名也可能是挂载在vipserver上，具体从psp上查看。
http://vipserver.alibaba-inc.com/app/pages/domain-list/index.html?env=pre
如果是调度任务，例如dts。请在doom客户端配置dts隔离。
如果是其他程序内部调度。请通在doom客户端配置力配置(客户端配置->子调用配置->自定义调用

断开)去避免调度执行。
例如:类 com.xxx.A 的 init() 方法 执行了 new Timer().scheduleAtFixedRate(xx); ,隔离 com.xxx.A@init() 方法就可以了。
如果是应用初始化避免慢doom先完成的初始化，可以在客户端配置->静态配置->客户端延迟启动时间 设置一个更长的延迟启动时间，默认是60s。

- 如果希望回放机器也提供服务
请在选择回放分组类型时选择‘录制&回放’分组。警告：线上回放请不要使用‘录制&回放’分组。

【是否一定需要解决】
通过doom的隔离已经避免的数据方案，因此不存在数据污染问题，不强制要求修复此问题。

【解决案例1】
异常堆栈：

```
at com.taobao.tddl.group.dbselector.AbstractDBSelector.tryExecute(AbstractDBSelector.java:442) ~[tddl-group-5.2.6.jar:na]
at com.taobao.tddl.group.dbselector.AbstractDBSelector.tryExecute(AbstractDBSelector.java:449) ~[tddl-group-5.2.6.jar:na]
at com.taobao.tddl.group.jdbc.TGroupStatement.executeQuery(TGroupStatement.java:505) ~[tddl-group-5.2.6.jar:na]
at com.taobao.tddl.transaction.log.GlobalTxLogManager.hasInited(GlobalTxLogManager.java:355) ~[tddl-transaction-5.2.6.jar:na]
... 9 common frames omitted
Caused by: com.tmall.doom.client.filter.InvalidInvokeException: disconnect.invoke-->com.mysql.jdbc.MySQLConnection@execSQL(StatementImpl,String,int,Buffer,int,int,boolean,String,Field[],boolean)
at com.tmall.doom.client.aop.handlers.RecordOrMockHandler.invoke(RecordOrMockHandler.java:151) ~[na:na]
```

经分析发现tddl中的：ScanTimeoutTask 这个类会定时扫数据库检查事务信息，对于回放机器而言可以通过在：[客户端配置->子调用配置->自定义调用断开] 中设置
：com.taobao.tddl.transaction.async.ScanTimeoutTask@scanGroup(*)
避免定时扫表逻辑解决。

回放时出现 DataSourceMockException

【原因】
如果隔离中间件选择了tddl隔离或者数据库隔离，那么需要对数据库访问进行mock。如果没有对数据库层访问进行mock则抛出该异常。
【如何解决】

- 方案一：【客户端配置】中选择db自动mock为‘是’，仅支持特定版本的ibatis、mybatis 框架。
- 方案二：【客户端配置】选择db自动mock为‘否’，配置好【DAO实现类】以便于做数据库访问层mock。

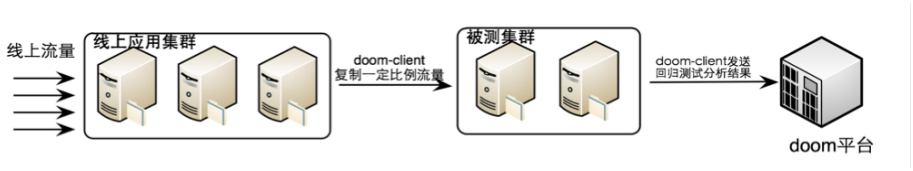
双引擎回归测试平台介绍

什么是双引擎平台

概述

双引擎自动回归平台（简称双引擎或者doom）是一个将线上真实流量复制并用于自动回归测试的平台。 通过创新的自动mock机制不仅支持读接口的回归验证，同时支持了写接口（例如用户下单接口、付款接口）的回归验证。 基于容器隔离机制以及完善的异常处理和监控机制，确保对应用本身的侵入非常小。 通过它，不仅能够实现低成本的日常自动化回归，同时通过它提供扩展能力可以支持系统重构升级的自动回归。 天猫、淘宝核心交易链路系统通过它实现低成本高覆盖率的日常自动化测试回归，同时内部的一些重大重构项目通过它保证系统的无故障升级。 它是复杂的难以通过传统方式做测试回归的业务系统以及金融类对故障十分敏感的系统的稳定性利器。

大致原理图如下



它与tcpcopy或者diffy的区别：tcpcopy、diffy是在应用外的网络层实现流量录制和回放的，它们只能实现一些只读页面的验证，而且无法实现跨环境的流量回放。双引擎是在应用内部通过aop切面编程方式实现的流量录制和回放功能，因此可以做到应用内部接口级别的回归验证，当然也支持服务级别或者http级别的回归验证。而通过独创的中间件级mock以及内部自定义的mock，可实现写流量的回归验证以及跨环境的回归验证（线上引流到测试环境）。

双引擎是一个封闭的系统吗？答案是否定的，我们希望把它打造成一个开放的平台，系统聚焦录制、回放、比对的核心能力。而数据的存储、数据的传输、中间件的扩展以及回放流程的都可以自定义扩展，使用者可以基于这些基础能力搭建适合自己业务的自动化回归平台。

目前双引擎在阿里巴巴集团内部被广泛使用，成为交易核心重构升级必不可少的利器。同时基于它扩展的‘天启’平台成为交易日常自动化回归的主要工具。

接入使用文档

应用场景

- 系统重构时，复制真实线上环境流量到被测试环境进行回归，相当于在不影响业务的情况下提前上线检测系统潜在的问题。
- 可以将录制的流量作为用例管理起来进行自动化归回。

优势

- 低成本：无需编写测试用例，通过流量录制形成丰富的测试用例。

- 高覆盖：一方面线上大量真实流量确保覆盖率，另一方面支持中间过程的验证，例如发送消息的内容、中间计算过程等等的全对象的对比验证，传统手工编写验证点很难实现。
- 支持写流量验证：（注：写流量是指可能导致有数据变更的流量）不用担心写流量回放污染应用数据，支持线上引流到测试环境以及写流量的自动化mock。
- 低应用侵入：通过隔离容器技术、字节码级别的AOP技术、中间件级MOCK避免接入类冲突以及降低接入成本。

录制回放的原理是什么

概念

主调用：待验证的流量入口，可以是http请求、也可以是一个java调用（公测版目前暂时只支持java调用）。
子调用：主调用执行流程中的一些方法调用，这些方法调用入参返回值会被记录下来用于回放时再次调用到该方法是时进行mock。

读接入模式：主调用是读接口，所有对外请求可以不进行mock，也就是不存在子调用的概念，这样的好处是可以验证到整个请求链路。

写接入模式：如果对外有写请求，那么需要通过配置中间件隔离将对外请求作为子调用进行mock。

原理解析

要实现引入线上流量来做回归验证牵涉到几个问题：流量如何复制？如何保证录制不影响业务？如何回放？如何解决写流量回放对业务的影响？如判断别被测系统bug？

流量如何复制?

java方法调用可以通过方法aop实现主调用以及子调用的入参和返回值的录制；http流量可以通过设置httpfilter或者对特定的servlet容器进行aop实现对http请求参数录制。当请求结束会启用异步线程池将录制数据序列化后发送给回放机器。

如何保证不影响业务？

针对用于生产环境的录制机器，任何流量录制异常不影响业务流程正常执行。同时在系统负载高时具备主动停止流量录制的能力。 针对用于生产环境的回放机器（beta机器），客户端可以支持指定rpc中间件不会真正对外提供服务，支持指定的消息中间件不去发送和订阅消息，同时支持指定的数据库框架不去真正执行写数据到数据库。如果是web服务器需要接入方摘除回放机器的vip。

如何解决写流量回放对业务的影响？

当回放后，可能会执行一些写逻辑，比如写分布式缓存、写db等等。而通过子调用mock的机制可以确保写不会真正发生，因此不会影响业务数据。

如判断别被测系统bug？

对于读接口，我们主要关注在相同请求下正常系统和待测系统的返回结果的差异，读接口也提倡对所有对外请

求进行mock，这样回放时能保持当时的一个现场环境，保证验证的准确性。

对于写接口, 只验证接口返回结果是不够的，需要验证它具体写入的数据是否正确。例如创建订单会调用tp的写订单接口，那么我们需要验证回放时调用tp的参数和录制时调用tp的参数是否一致。

对比默认是全对象字段逐一对比，如果存在必然不一致字段，例如时间，系统ip等等可以配置忽略该字段的对比。默认开启子调用对比后所有子调都会对比，也可以配置排除一些不关心的自调用的对比。

线上流量可以到线下回放吗？

答案是可以的，可能大家会疑问，线下库和线上都不一样，怎么跑的通？实际上如果所有对外请求都做了mock，那么线下回访并不会真正发起对外调用，都用线上数据mock，所以可以跑通。利用这点我们可以在项目测试环境回放线上执行的过程，帮助问题复现排查。

哪些调用会成为子调用？

平台有中间件隔离配置，例如配置了hsf隔离，那么所有hsf对外调用都会作为子调用。类似的还包括tair、notify等等。另外也支持自定义一个类方法作为子调用。当然我们也提供白名单配置指定某些调用不进行mock，这样可以验证到下游的读逻辑或者解决无法mock的问题。

如何解决回归对比噪音

问题

将部署了稳定代码的服务器作为流量采集源，对于待正式发布的代码，可以用录制的流量来进行回放和差异比对，以便于发现待测系统的问题。那么录制环境和回放环境所处环境不同，有一些必然不一致的信息，例如服务器ip、时间、以及一些随机数等等。怎么去处理这些差异呢？

方案

- 排除法：平台支持指定字段排除对比，将不需要的字段排除即可。
- 指定对比法：将关心的业务数据进对比。

使用双引擎对应用有哪些影响

- 数据录制有一定的系统资源消耗，具体视应用而定，一般都在可接受范围以内。
- 运行客户端大概有100~300M的额外内存消耗，请规划好应用的内存。

双引擎接入使用文档

原理介绍

平台介绍

适用范围

支持java应用在线引流自动化测试。另外请检查您的应用服务器可以正常访问域名：doom.rdc.aliyun.com

申请使用要求

流量录制回放是一个极其复杂的功能，使用有一定门槛，只有有能力使用的用户才能发挥出它的价值，因此我们对试用用户有如下要求：

- 企业有一套统一的中间件体系，以便于一次成功接入能快速推广。
- 企业至少投入一个全职研发同学或者研发能力较强的研发测试同学来参与接入以及推广工作。
- 申请人必须是技术主管及以上职位的同学，以确保资源投入。

联系方式

mufeng.qcg@alibaba-inc.com

接入文档

平台入口

入口链接

1. 应用接入申请

doom流量录制回放是应用维度的，因此首先得申请一个接入应用。

1.1 申请方法

1.1.1 非云效应用接入申请方式

1.首先进入控制台首页，**入口链接**，确认阿里云能正常登录，切换工作空间到个人空间（默认情况下进入个人空间，不需要做切换）。如下图：

1.1.2 云效应用接入申请方式

1.2 审批结果

目前公测阶段由产品后台方审批，审批完成后会通过用户在rdc绑定的邮箱发送审核结果。如果在3个工作日后还未收到通知结果，请发邮箱咨询我们。 联系人： 穆风 mufeng@aibaba-inc.com

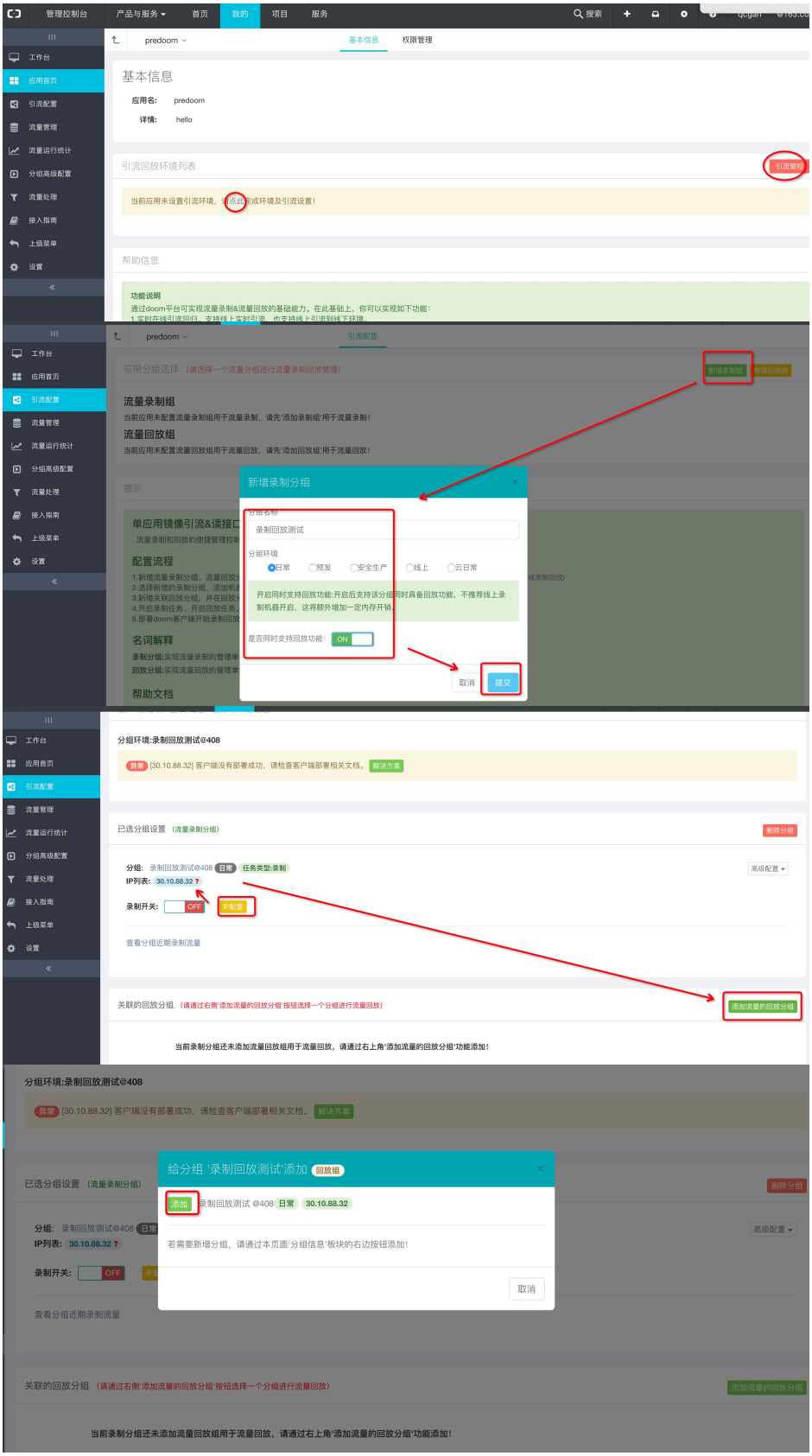
公测结束后，会把权限下放到企业管理员。

2. 快速入门

2.1 引流配置&oss空间设置

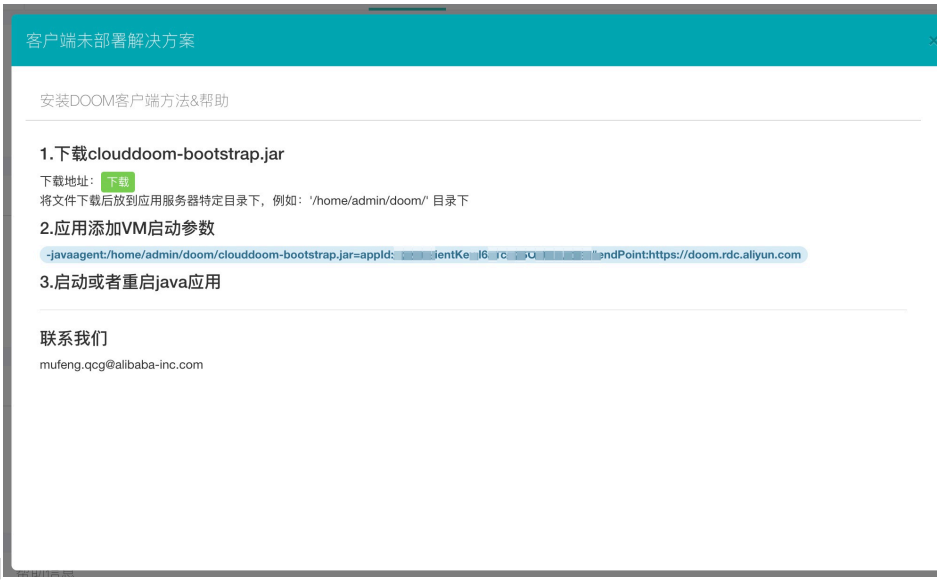
当应用申请完毕后，请联系相关同学审核。审核完成后在申请页或者应用管理页进入刚申请好的应用进行配置。

2.1.1 引流配置


管理控制台
产品与服务
首页
我的
项目
服务
基本信息
权限管理
应用名: predoom
详情: hello
引流回放环境列表
帮助信息
功能说明
引流配置
流量录制组
流量回放组
新增录制分组
录制回放测试
日常
预发
安全生产
线上
云日常
是否同时支持回放功能:
ON
取消
提交
分组环境:录制回放测试@408
已选分组设置 (流量录制分组)
录制开关:
OFF
查看分组近期录制流量
关联的回放分组
添加流量的回放分组
当前录制分组还未添加流量回放组用于流量回放，请通过右上角'添加流量的回放分组'功能添加!
给分组 '录制回放测试'添加
回放组
添加
若需要新增分组，请通过本页面'分组信息'板块的右边按钮添加!

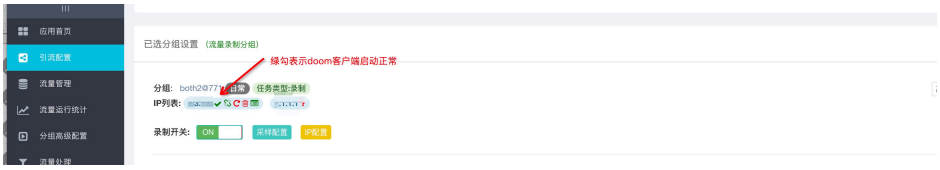
2.1.2 客户端安装

doom客户端通过设置VM javaagent参数来完成安装，当系统提示客户端没有部署成功时，点击’ 解决方案



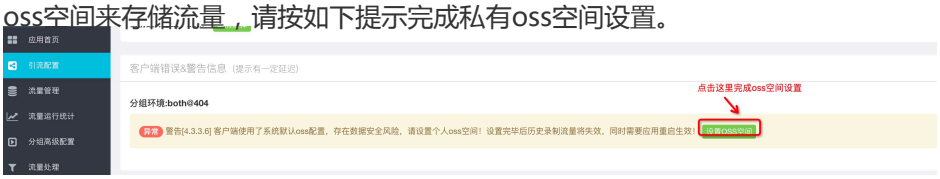
’ 会弹出如下窗口

请按提示完成clouddoom-bootstrap.jar的下载以及jvm参数的设置，然后重启应用服务。如果一切配置正常，将出现如下’绿色勾’标识，此时说明客户端已经部署正常



2.1.3 oss空间设置

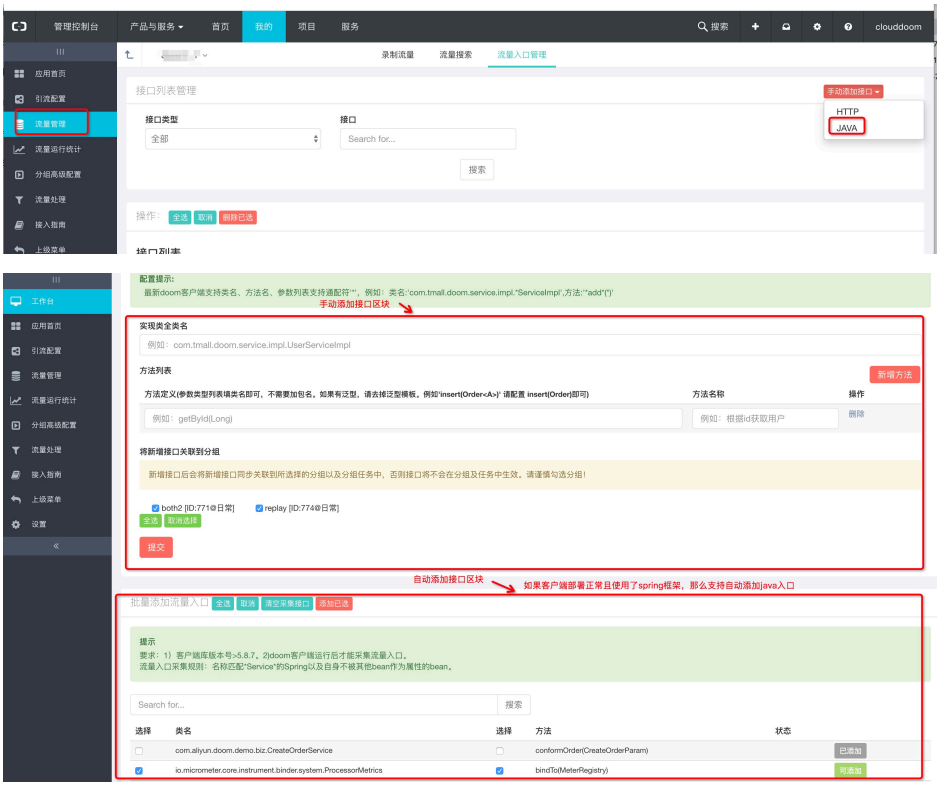
默认情况下doom提供一个默认的oss存储空间用于试用和测试，如果要录制生产环境流量，建议购买私有的oss空间来存储流量，请按如下提示完成私有oss空间设置。



完成如上配置后，HTTP类型的流量已经支持自动录制和回放。若希望录制java方法级别的录制回放，还需要进一步做JAVA接口配置。

2.2 设置JAVA流量入口的设置方法

目前支持任意spring bean或者应用中的单例进行录制回放。一般后台应用都是通过对外提供服务供给上游调用的，而服务的具体provider也是通过service bean来实现，因此针对后台应用，可以配置对应服务的bean实现流量录制回放。



2.3 镜像模式和非镜像读接口模式

2.3.1 镜像化模式

默认情况下doom会对所有支持的中间件外部请求进行镜像化mock，例如A应用通过dubbo调用B应用，那么在引流回放的时候当回放应用调用B应用时，客户端会对调用进行拦截，使用录制时的结果进行mock，避免真实调用发生。这样会带来如下好处：

- 避免回放产生脏数据。
- 保存了录制现场，避免回放因数据变化而失败。
- 支持线上流量到线下环境回放。
- 结合时间的镜像化，支持流量的持久有效。

2.3.2 非镜像化读接口模式

支持非镜像mock回放模式，但这种模式只能应用于读接口，例如查询接口，查询页面等等，但不能用户写接口，因为回放所有调用都会真实发生，会出现重复写数据问题。此模式的优势是接入配置成本会更低，且能对下游系统进行验证。配置方法：

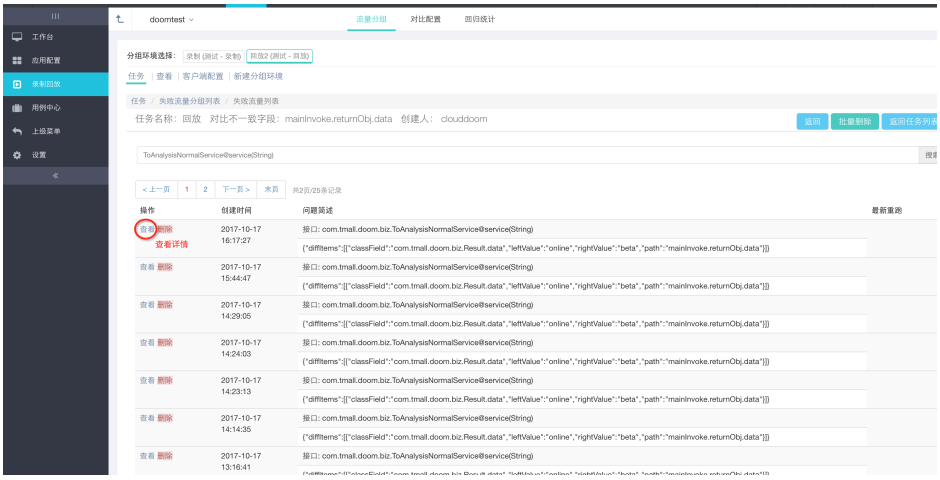
3 检查引流回放情况

3.1 引流数据及结果查看

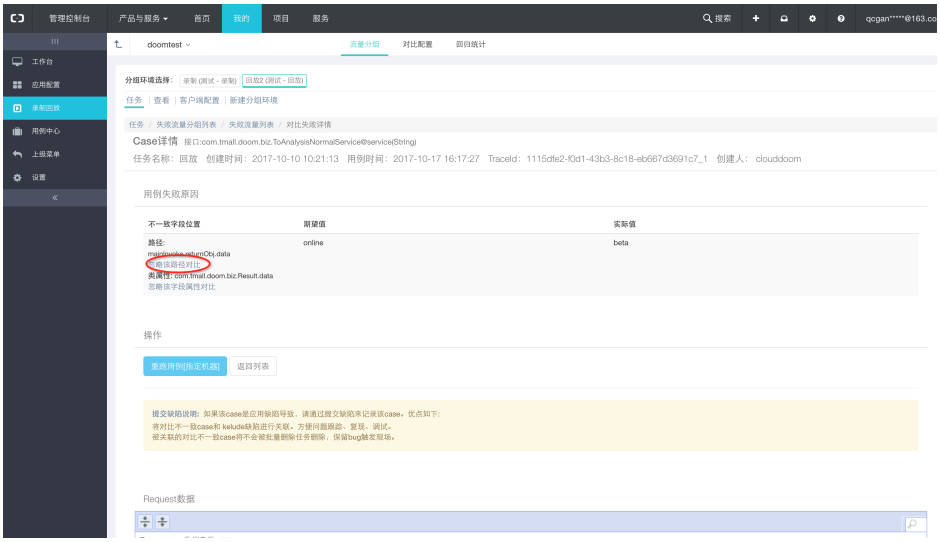
3.2 回放失败分析

进入查看失败原因

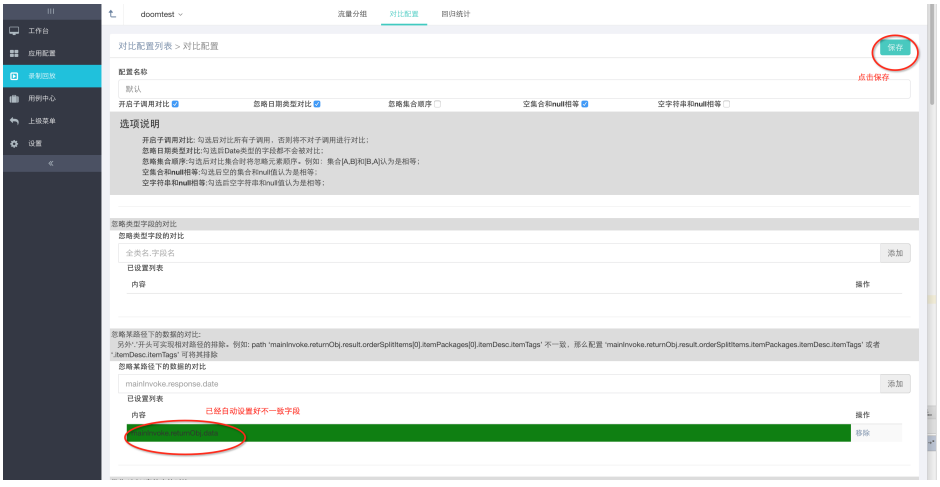
进入查看失败具体列表



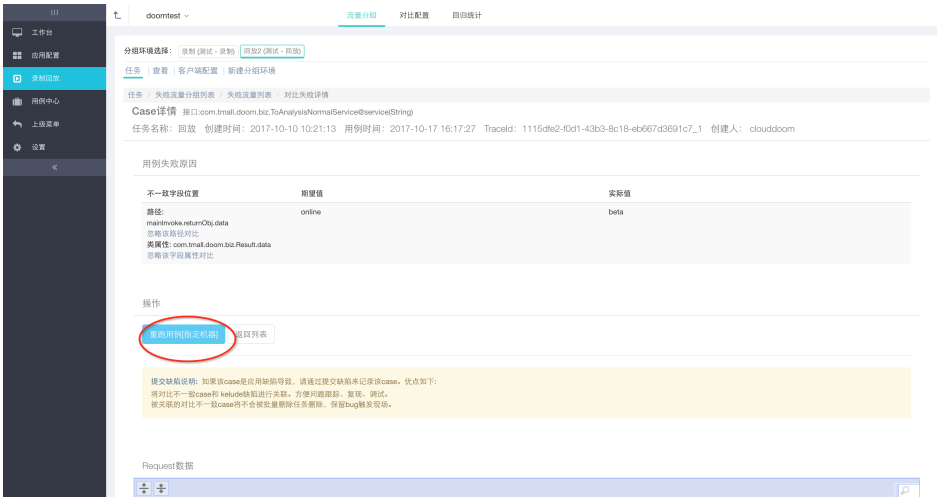
查看具体对比失败原因，查看是否回放失败，如果没有回放失败则查看不一致字段，判断是否为噪音，如果是的话可以点击排除



排除噪音字段后保存



重跑验证（重跑的前提是回放机器客户端要正常运行）



3.3 开通opensearch&设置数据表

- 进入https://opensearch-cn-beijing.console.aliyun.com/#/apps并点击创建应用
- 选择‘标准版’并立即创建
- 输入应用名例如‘doom_data’进入下一步
- 选择‘手动创建应用结构’进入下一步
- 新建数据表，字段如下

字段名称	类型	主键
id	LITERAL	是
invoke_identity	LITERAL	否
timestamp	INT	否
source_id	LITERAL	否
container_id	INT	否
trace_id	LITERAL	否
item_key	LITERAL	否
data_key	LITERAL	否
request_json	TEXT	否
response_json	TEXT	否
data_record_id	INT	否

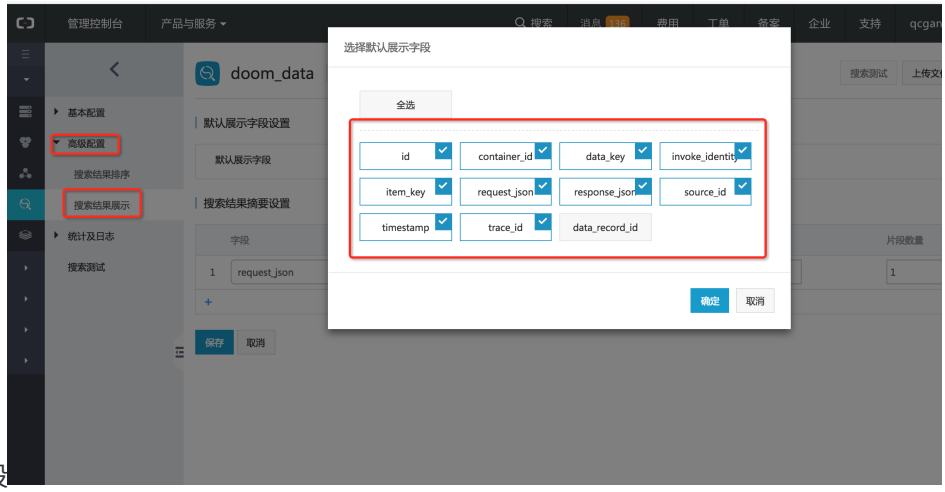
- 字段索引设置

索引名称	包含字段	分词方式
id	id	不分词
default	request_json、response_json	中文 - 基础分词
invoke_identity	invoke_identity	不分词

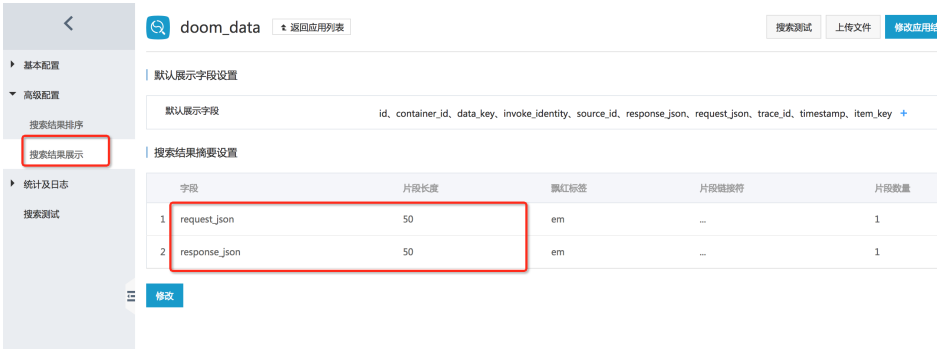
trace_id	trace_id	不分词
item_key	item_key	不分词
data_key	data_key	不分词
time_stamp	timestamp	不分词
source_id	source_id	不分词
data_record_id	data_record_id	不分词

- 在数据源设置也直接点击完成

设置默认展示字段



结果摘要设置



- 回到应用列表，激活应用（根据需求合理选择容量大小，也可以申请免费容量试用）

重构场景的支持

系统重构如何使用doom回归

重构情况下的引流回放建议

- 若预发和线上环境配置一样，可以从线上复制流量到预发环境回放。
- 若预发和吸纳上环境不一样，可从线上环境复制流量到beta机器上进行回放。

重构场景下doom的配置方法

通常系统重构应用执行流程会发生变化，这时还能使用doom回归吗？一般来说，只要流量入口没有特别大的变化，接口功能也没变化时是可以的。如果是读接口接入，即不对中间件及外部服务进行mock，那引流回放很容易就能配置。如果需要mock外部服务，那么配置相对复杂些。

读接口不做任何mock的配置方法

- [分组环境-客户端配置-隔离的中间件]取消所有隔离中间件。
- [分组环境-客户端配置-开启DB自动MOCK]选择否。

完成如上配置后doom录制回放不做子调用mock。

应用名改变了如何跨应用引流？

例如应用名从A变为B如何引流: 以一个应用为主进行doom接入配置，例如统一接入B应用，那么在B应用的

DOOM配置中录制机器配置A应用的机器进行引流，回放机器则配置B的机器。

主调用变化怎么办？

JAVA流量入口的重构

主调用以及do对象包名更改，类名没改的场景

可以在 客户端配置->静态配置->自定义扩展配置->收集机和测试机类包名修改映射 去进行包名映射配置。**要求重构后的应用要依赖原来未重构时的do类。**

主调用包名和类名均修改的场景

Java主调用处理方式：**

例如：采集机器：x.A类；回放机器变成：y.B类

在回放机器新增一个x.A类去调用y.B类（代理）。同时在回访机器新增x.A类的spring-bean(便于能回放)
在doom平台配置x.A类为调用入口。

HTTP流量入口的重构

方案一：url 变化、参数名变化可以通过插件扩展在回放前修改快照中的url以及参数确保能正常回放。

方案二：保留老的http入口存在的情况下对参数做适配，去调用新的重构逻辑，一遍验证新逻辑的正确性。

关于比对

一般来说重构后response一致的直接比对结果即可，如果返回结果格式也重构了，建议通过比对脚本提取需要比对的字段或对象进行对比。

新增子调用怎么办？

新增读接口

将此接口放开，让其真正的调用下去，通过[客户端配置->忽略子调用MOCK]进行配置。

新增了写接口：自定义插件解决

通过插件扩展实现自定义结果返回，一般写接口返回都是比较简单的，可直构造结果返回。也可以在扩展中读取线上录制的快照构造结果返回，这需要根据实际使用场景而定。

插件的使用文档待完成。

重构场景对回放机器有什么要求？

如果是预发或者线上录制，如果开启类子调用放开mock，那么要求回放机器也是预发或者线上beta环境。

测试服务

测试服务

云效测试服务是包含多种自动化测试能力，将测试进行场景分类，除了通用的单元测试，代码扫描，接口测试，也包含了阿里巴巴特有的测试能力如持续集成实验室，流量回归测试。对于测试服务使用者，通过测试服务页面能够快捷的找到所需要的测试服务入口。测试服务传送门

测试服务类型

当前运行提供的测试服务包含：

启用测试服务

测试服务在新建企业中默认启用，启用后，在“我的”->我的导航中显示“测试”菜单，点击后，默认进入测试服务新建页。如果相应的企业中，未显示测试服务，请点击设置

新建测试服务

测试服务和流水线

测试服务通过将测试能力进行抽象，提供方便创建表单，提供了通用的持续集成，持续验证，结果通知，触发调度的能力。用户在测试服务中积累的任务,可以和流水线关联，作为持续发布验证的质量关卡。

在流水线中添加测试节点

在云效的自定义流水线中可以新增测试节点。

在发布中查看测试结果

测试集合 - 查看测试日志，重跑，创建缺陷

点击详情链接，可以看到更清楚的测试集合页面。在页面上可以支持查看测试的日志，支持将失败用例重跑，并支持创建缺陷。

应用部署

应用部署概述

每个Web应用，在集成测试的环境（在云效中通常称作日常环境）、预发的环境（称作预发环境）、对外提供服务的环境（称作正式环境）等不同的环境里运行。每个环境中，该应用运行在若干台机器（虚机/容器）上。部署时，可能分期分批。每台机器的部署，有特定的方法和步骤。这些都是定义在这个应用的特定环境上。

云效支持您创建自定义环境，详见环境与环境级别。

这意味着，可以为同一应用的不同环境，配置不同的部署参数，甚至不同的部署方法。比如日常环境通过容器服务部署，而正式环境通过脚本部署。

下面详述当前云效支持的以下几种部署方式的配置：

- 部署配置：通过脚本部署
- 部署配置：通过EDAS部署
- 部署配置：部署容器服务的Swarm集群
- 部署配置：部署容器服务的Kubernetes集群
- 不需要传包，直接通过git pull的方式进行部署

环境与环境级别

本文介绍云效中的环境、环境级别的概念，及如何同时部署一个环境级别中的多个环境。目前这一部分内容，是针对分支模式这种开发模式的。

若要了解更加通用的，如何在环境上配置资源进行发布，请查看应用部署概述。

基本概念

首先要理解云效中环境级别的概念。

云效中预置了日常环境，预发环境和正式环境三个**环境级别**。每个环境级别可以包含一个或者多个具体的环境。云效在新建的应用中为您预置了日常环境、预发环境、正式环境三个**环境**，分别对应上述的三个环境级别。

使用环境级别的场景

一个环境级别代表了一组环境，比如您的应用使用阿里云容器服务进行部署，希望把同一个镜像部署到杭州和北京两个容器集群。则可以创建两个，环境级别为**正式**的环境：杭州正式，北京正式，然后在两个环境中分别

关联相应的容器集群中的相应服务。关于容器部署的更多内容，请参看部署配置：通过容器服务部署。

在分支模式下，预置的三个流水线，日常部署，预发部署，正式部署，会部署相应的环境级别中的所有环境。比如对于上述的杭州正式，北京正式两个环境，在正式部署的流程中进行部署时，会并行部署这两个环境。

新建环境

您可以在应用的概述->环境配置中添加新的环境，如图：

添加完成之后，可以看到：

此时，您的正式环境这个环境级别中包含两个**环境**：正式环境，正式环境1。

添加完成之后，您可以继续在最上层的环境tab中，编辑相应环境的部署配置。

同时部署多个环境

新环境创建完成之后，您就可以在原有的正式发布流程中，同时发布正式环境，正式环境1这两个环境了，而流程本身没有任何变化。如下图所示：

点击查看正式部署日志，可以看到多个环境的各自的部署信息。

部署配置：通过脚本部署

本文讲解如何配置通过自定义脚本把Web应用部署到指定服务器。

关于如何配置构建产生部署用的包，请参见Web应用构建配置。

关于如果进行企业机器资源的管理，如果已有主机，请参见把已有机器关联到云效本企业，如果暂时没有主机，请参见配置授权云效、配置ECS模板、购买ECS机器。

应用环境关联到机器

在“应用”-“环境”页面，点击“资源管理”，可以增加关联的机器。

提示

- 应用可以关联的机器，来自于企业管理员在企业设置中导入的机器；
- 如选择不到机器，请联系管理员导入机器；

应用部署信息配置

新建好应用之后，在环境页面，您可以看到，云效会为您预置日常、预发、正式三个环境。并且可以对每个环境做部署的配置：

注意：

当前环境正在部署时，部署配置无法修改。（若在部署过程中修改部署配置，会生成中间版本，当部署完成写入正式基线后，该中间版本会被清除掉）>注意：当前环境正在部署时，部署配置无法修改。（若在部署过程

中修改部署配置，会生成中间版本，当部署完成写入正式基线后，该中间版本会被清除掉）

不同的应用可以有自己的定制化的部署脚本，这里给出了一个基于SLB的滚动发布脚本示例，供您参考。

此外，可以把部署脚本放在代码库中，当它的内容更新时，将在部署时自动同步到各机器。详述见在代码库中存储部署脚本。

部署配置：通过EDAS部署

一、概述

EDAS是阿里云上的一个服务，提供了中间件，部署，及运维等能力，详情见EDAS文档。云效对EDAS进行了集成，可以把在云效上打出来的war包或者jar包部署到EDAS中。

为了在云效上集成EDAS，需要保证您的应用可以在云效上打出war包或者jar包。详见使用EDAS部署时的构建配置。

云效支持多种研发模式，及部署回滚等功能。EDAS提供了中间件、部署、运维、日志、监控等服务。云效与EDAS结合，可以很好的提供一站式持续交付体验。

EDAS提供了多种部署能力，云效目前只支持基于war包和jar包的部署，不支持EDAS容器部署。

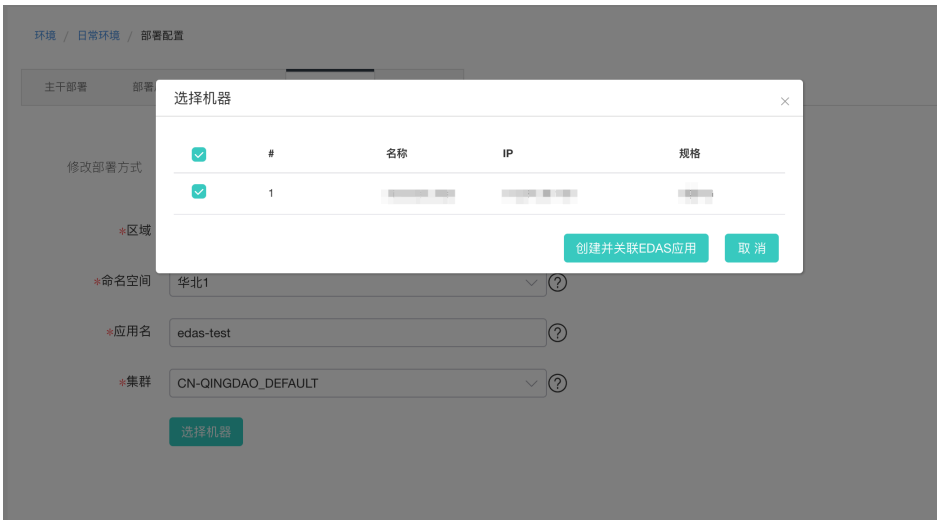
你可以在EDAS上创建应用，也可以在云效上创建EDAS应用，然后使用云效进行集成发布。

二、云效上创建EDAS应用

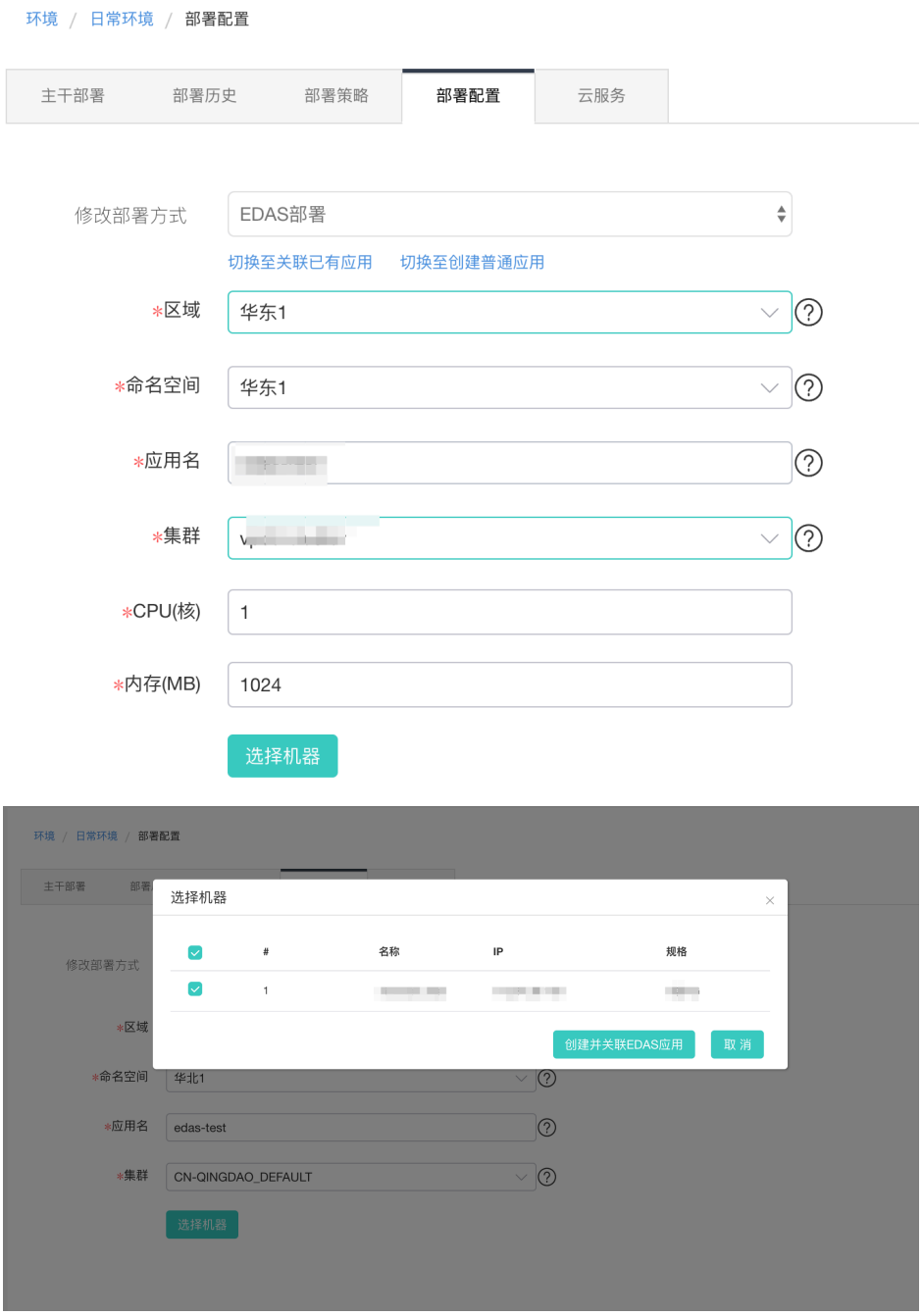
目前在云效上只支持创建普通EDAS应用和DOCKER EDAS应用，暂不支持创建Kubernetes应用。

在应用 -> 环境 -> 部署配置 的页面 可以创建EDAS应用。 创建应用前需要在EDAS控制台相应的区域下，添加相应集群，并关联好机器。

2.1 创建普通 EDAS应用



2.2 创建Docker EDAS应用



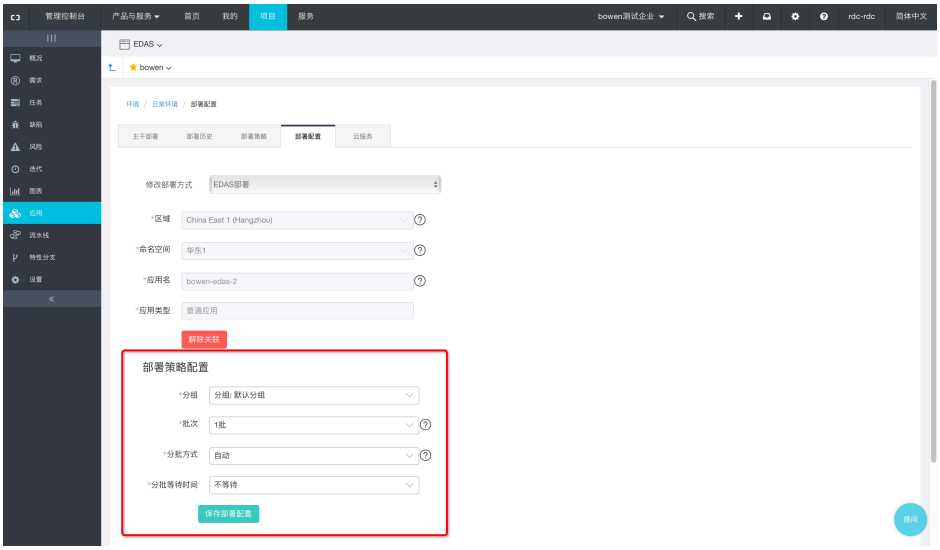
三、云效上关联EDAS应用

在应用 -> 环境 -> 部署配置 的页面配置对应的EDAS的应用ID，如图所示：

EDAS应用ID可以从EDAS应用详情页面获取：
https://edas.console.aliyun.com/#/detail/xxxxxxx/basicInfo.info。如图：

四、EDAS应用部署策略配置

当新建或关联了EDAS应用以后，用户可以配置发布策略。



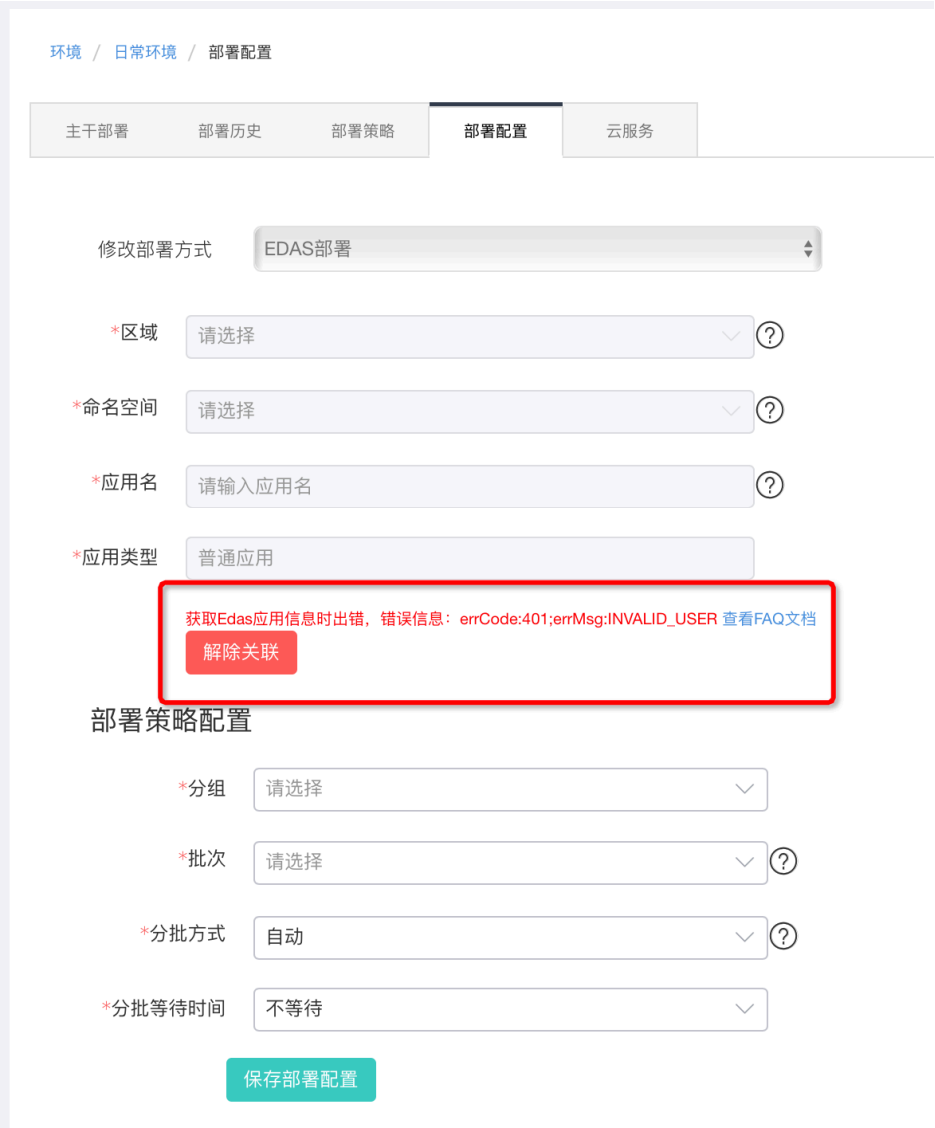
用户给可以选择具体的分组、发布批次以及分批等待时间来定制化发布策略，从而保证线上服务的稳定性。注意选择批次时的批次数应小于等于当前EDAS应用的实例数，否则在部署时会报告The batch times is greater than the number of instances.错误。

五、EDAS操作常见问题

5.1 查看EDAS应用信息出错

5.1.1 无权限查看应用信息或查看的应用不存在

在部署配置页面，查看EDAS应用信息时，出现以下错误，是由于当前用户无相应EDAS应用权限或该EDAS应用不存在。如果是权限缺失，请联系相应的EDAS应用管理员，为您的阿里云账号添加权限。关于如何添加权限，请参考EDAS账号体系。如果权限没有问题，请到EDAS控制台确认当前应用是否存在，如果应用已删除，可解除该EDAS应用与云效的关联关系。



5.2 部署应用时出错

5.2.1 未配置云效不对构建物进行压缩

使用云效进行EDAS部署时，出现以下错误，是由于未配置云效不对构建物进行压缩，可以按照下面方式解决。详见使用EDAS部署时的构建配置

5.2.2 无权限部署EDAS应用

当前操作人（比如点击“重新部署”的操作人），需要具有部署到指定EDAS应用的权限。如果没有权限，则会报如下的错误：

如遇这种情况，请联系相应的EDAS应用管理员，为您的阿里云账号添加权限。关于如何添加权限，请参考EDAS账号体系。

5.2.3 EDAS应用不存在

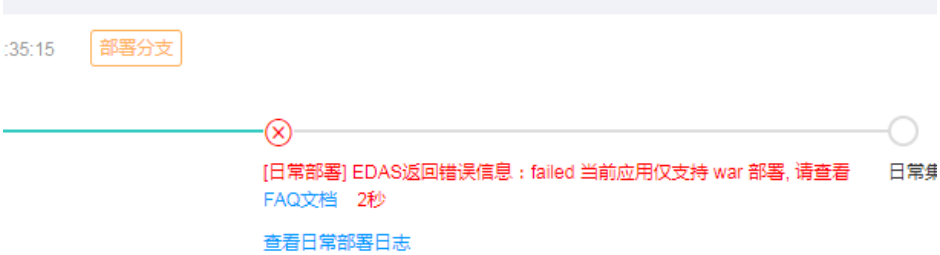
使用云效进行EDAS部署时，出现以下错误，是由于当前环境关联的EDAS应用已删除，请到EDAS控制台确认当前应用是否存在，如不存在，可以解除当前环境与该EDAS应用的关联，并重新关联可用EDAS应用。

5.2.4 EDAS应用无可部署机器

使用云效进行EDAS部署时，出现以下错误，是由于当前环境关联的EDAS应用无可部署机器，请到EDAS控制台关联机器后，重新进行部署。

5.2.5 部署包格式不正确

EDAS的ECS部署支持jar包和war包两种格式。如果你看到了下面的错误：

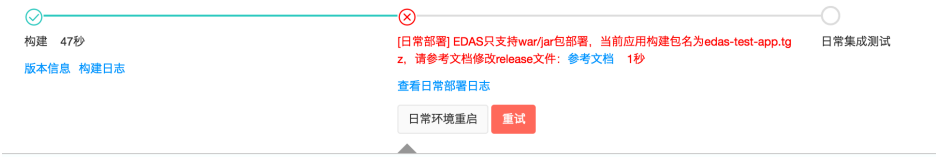


那么有两种可能：

1. 该EDAS应用所选择的容器不支持jar包的部署方式。需要您在创建EDAS应用时选择支持fatjar部署的容器版本。

2. 该EDAS应用之前使用过war部署，而本次尝试部署尝试使用jar包进行部署。

如果遇到下面的错误：



说明您的构建配置不正确，请参看构建配置进行修复。

5.2.6 EDAS应用已有部署单执行

使用云效进行EDAS部署时，出现以下错误是由于当前环境关联的EDAS应用已经存在部署单，可到EDAS控制台查看该应用部署信息，待上一个部署单部署结束后，在云效相应页面点击重试即可。

5.2.7 需要在EDAS控制台执行一次部署

使用云效进行EDAS部署时，出现以下错误时，可到EDAS控制台手动部署一次。完毕后在云效相应页面即可进行部署。

5.2.8 发布批次配置大于当前EDAS应用的实例数

使用云效进行EDAS部署时，出现以下错误是由于当前环境部署策略配置的发布批次大于EDAS应用的实例数。可以到 应用 -> 环境 -> 部署配置 的页面修改发布批次，将发布批次数改为一个较小的数字。

部署配置：部署到容器服务的Kubernetes集群

概述

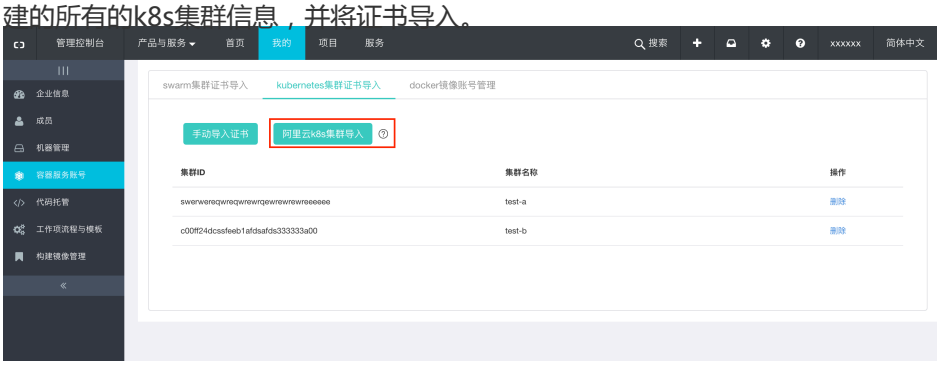
Kubernetes 是当前主流的开源容器编排技术，为容器应用的管理带来很大的便利。但是，将镜像部署到集群的过程需要开发者者有k8s知识基础，而且如果通过手工操作容易出错，影响发布效率。
云效提供构建到部署全链路自动化的流程，支持部署到阿里云容器服务的创建的k8s集群。让用户专注于业务开发，降低发布成本。

集群导入

集群属于企业信息，需要企业管理员操作，通过【企业设置】进入【容器服务账号】页面。选择【kubernetes】tab页面后,可以看到导入的操作入口。

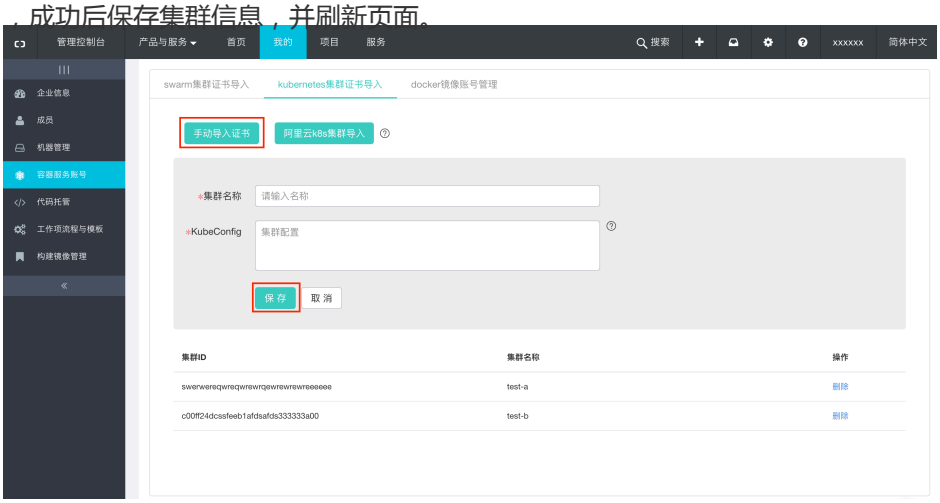
容器服务集群导入

点击【阿里云K8S集群导入】按钮， 系统会根据当前操作人（***需要是阿里云主账号***）的账号，查询容器服务创建的所有的k8s集群信息，并将证书导入。



其他托管或自建集群的导入

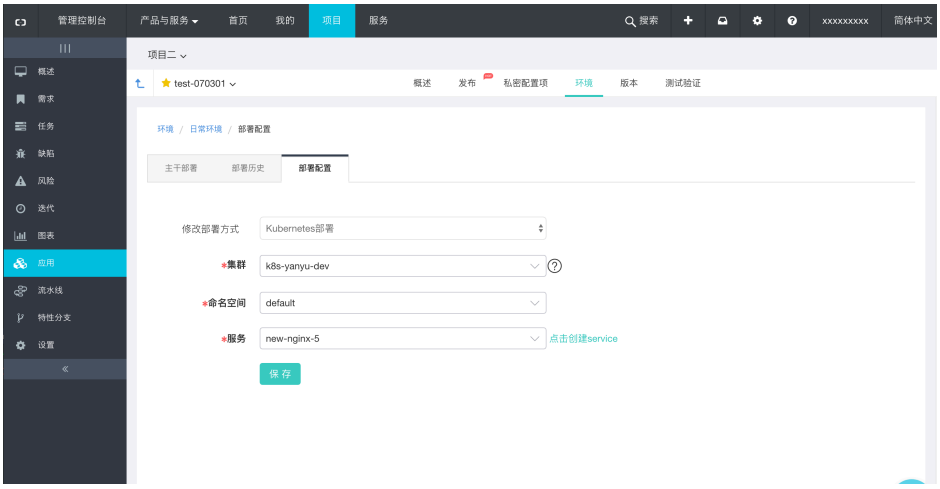
点击手动导入证书，将显示集群导入的输入操作框，填写信息后，点击【保存】按钮。系统会校验集群的证书，成功后保存集群信息，并刷新页面。



对于不需要展示的集群，可以进行删除操作。

环境配置

到【环境】页面，进入某个环境的【部署配置】。选择发布的集群、Namespace和service，点击【保存】。



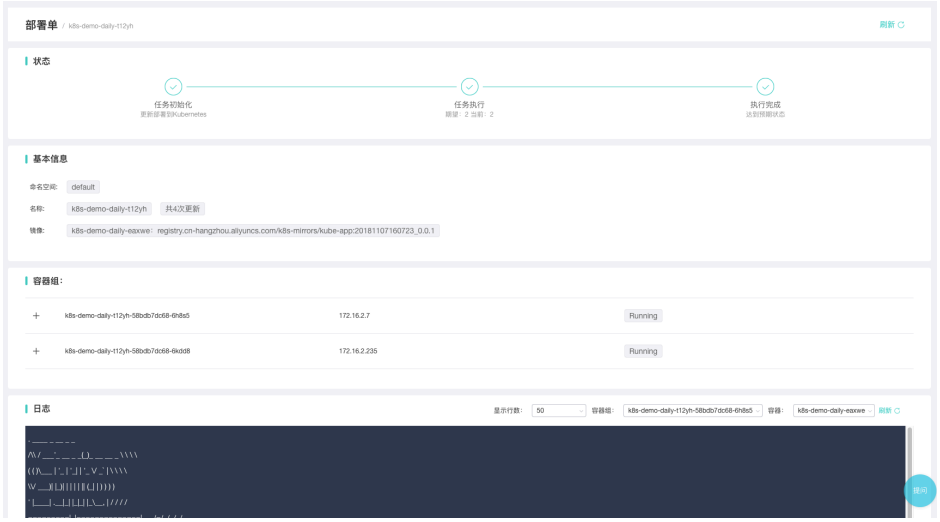
如果当前service对应的deployment不止一个，页面会展现deployment和container的选项，选择后点击【保存】即可

如果你需要新建service，点击service选项旁边的【新建service】链接，填写信息后，点击【保存】，将创建service并保存部署配置

部署操作

通过流水线进行应用的部署，在部署过程中，可以点击【查看部署日志】查看部署进度

点击后，跳转到发布单页面，查看对应的deployment（如果报没有权限，请联系创建集群的主账号给您添加访问权限）



Kubernetes蓝绿发布

Kubernetes蓝绿发布

本文将向读者介绍如何在云效中发布基于Istio的应用程序，同时利用云效提供的蓝绿发布能力更安全的发布和验证应用。

本文采用Istio官方的Bookinfo实例程序，源码地址：https://gitee.com/moo/bookinfo.git

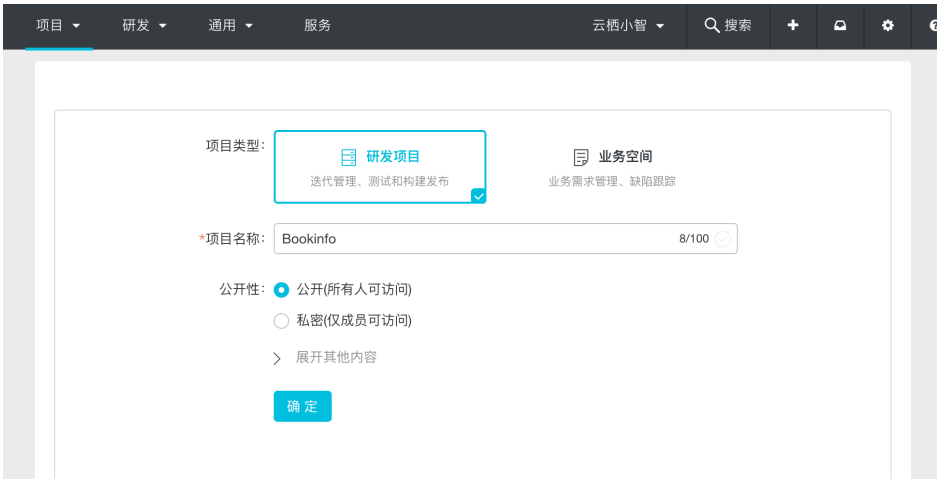
前提条件

- 在阿里云容器服务Kubernetes中创建集群，并导入到当前企业中

- Kuberntes集群安装Istio组件
- 开通阿里云镜像仓库服务用于托管容器镜像

创建项目

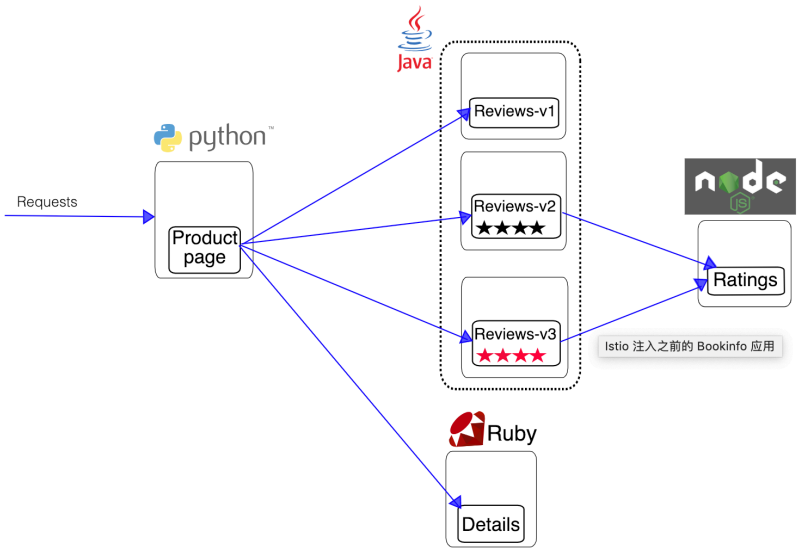
在云效中我们采用**项目**的概念来管理一组相关的应用，并且对项目提供了如敏捷看板，测试，文档等服务。通过项目可以端到端的管理应用的整个交付周期：



这里我们为Bookinfo创建一个独立的项目，创建完成后就可以进入到该项目的主页：

创建应用

Bookinfo应用是一个典型的微服务应用程序，其主要结构如下所示：



Bookinfo示例程序组要由Productpage,Reviews,Details以及Ratings4个部分组成，同时使用了Python,Java，Ruby以及Node四种不同的技术栈。

创建Productpage应用

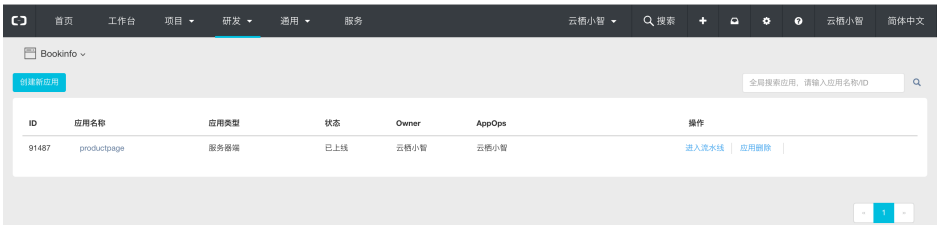
首先，我们在Bookinfo项目下创建应用productpage，如下所示：

由于示例应用源码是托管到码云(Gitee)中，我们需要完成码云授权，并关联已有源码：

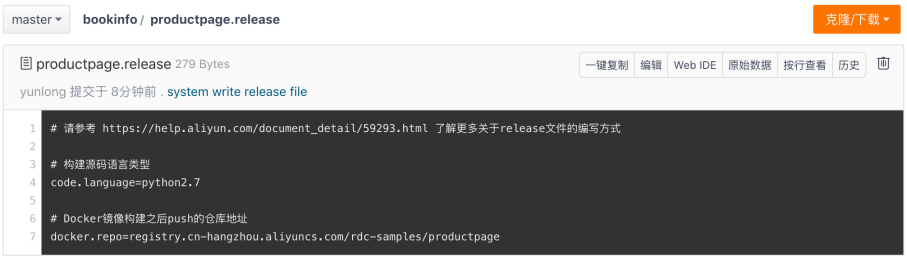
关联已有代码后，我们需要为当前应用选择相应的应用模板，当前Productpage应用采用Python进行开发，并且使用Kubernetes进行部署：

在完成编程语言以及部署选项的设置后，用户需要定义应用时如何构建的，由于需要将应用部署到Kubernetes中，这里我们勾选Docker构建选项，并且定义了当前应用镜像发布的镜像仓库为rdc-samples/productpage。云效会自定在项目中生成productpage.release文件，该文件定义了Productpage是如何构建的：

下一步，预览并创建应用即可。



查看源码，可以看到云效自动创建的release文件：https://gitee.com/moo/bookinfo/blob/master/productpage.release：

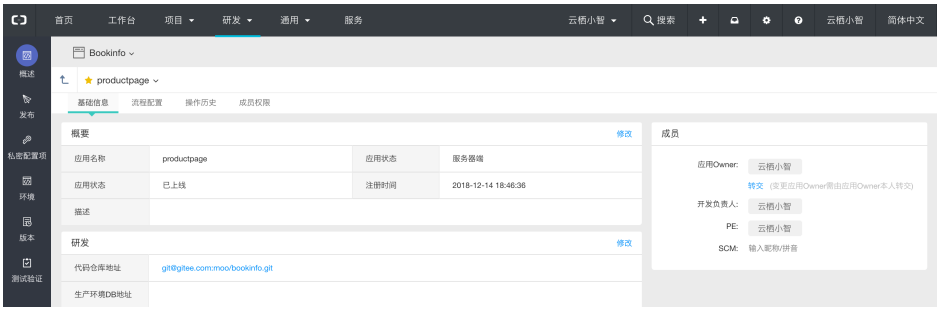


由于Productpage应用是在项目的src/productpage路径下，这里需要手动修改productpage.release文件的内容，并制定Productpage应用的Dockerfile文件路径，完整配置如下所示：

```
# 构建源码语言类型
code.language=python2.7

# Docker镜像构建之后push的仓库地址
docker.file=src/productpage/Dockerfile
docker.repo=registry.cn-hangzhou.aliyuncs.com/rdc-samples/productpage
```

进入到productpage应用，可以看到在云效中应用会包含应用的发布流水线，配置管理，环境管理以及版本等能力：



进入到应用的发布页面可以看到云效为应用自动创建的持续交付流水线。触发流水线构建，在构建阶段云效会根据productpage.release文件定义的内容对项目进行打包以及镜像构建，并且将镜像推送到阿里云镜像仓库服务：

不过由于，我们还未进行日常，预发以及正式环境的部署配置，因此当流水线运行到日常部署阶段，会出现如下错误信息：

根据提示，我们进入到日常环境的部署配置页面，这里我们为日常环境创建了名为dev的命名空间，并且为该命名空间启用了Istio的自动注入能力：

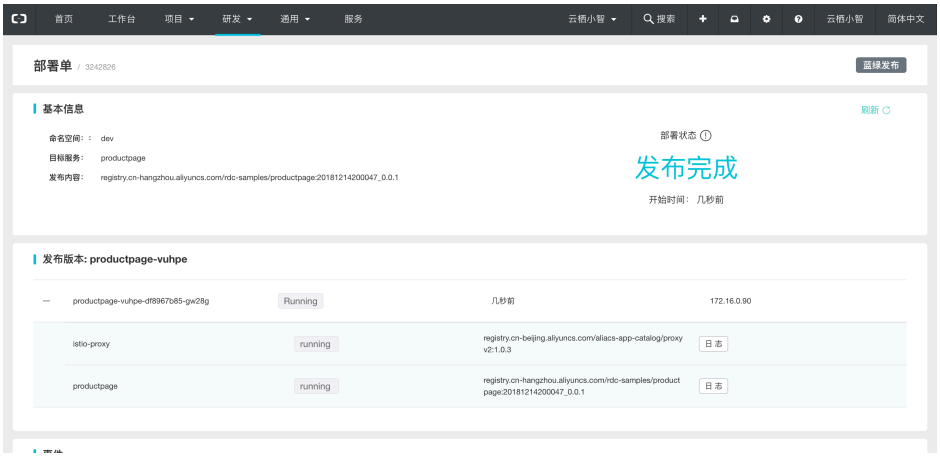
```
kubectl create namespace dev
kubectl label namespace dev istio-injection=enabled
```

选择集群，命名空间并且新建productpage服务：

这里需要指定Productpage服务的端口以及相应的容器端口，在bookinfo中productpage应用监听的是9080端口，完成配置后如下所示，对于启用了Istio支持的集群，云效会自动打开蓝绿部署选项：

完成配置后，回到流水线页面重新触发日常环境部署动作，由于是第一次部署Productpage服务，云效会直接完成服务的初始化动作：

查看发布单，可以查看具体的资源信息，这里云效自动创建了productpage的Pod实例，并且可以看到自动注入的istio-proxy容器：



部署单发布完成后，查看当前集群dev命名空间下的所有资源如下所示：

```
$ kubectl -n dev get all --selector='app=productpage'
NAME                                  READY   STATUS    RESTARTS   AGE
pod/productpage-vuhpe-df8967b85-gw28g   2/2     Running   0          3m

NAME                  TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
service/productpage   ClusterIP   172.19.0.131   <none>        9080/TCP   54m

NAME                                DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/productpage-vuhpe   1         1         1            1           3m

NAME                                           DESIRED   CURRENT   READY   AGE
replicaset.apps/productpage-vuhpe-df8967b85    1         1         1       3m
```

云效会为应用版本创建相应的DestinationRules以及VirtualService资源，并且在第一次部署完成后自动将路由规则定向到当前部署版本。

查看DestinationRules如下所示：

```
#$ kubectl -n dev get destinationrules productpage -o yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  creationTimestamp: 2018-12-14T11:59:07Z
  generation: 1
  name: productpage
  namespace: dev
```

```
  resourceVersion: "341256082"
  selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/destinationrules/productpage
  uid: a8fe5fc8-ff97-11e8-bbda-de3c7d18d080
spec:
  host: productpage
  subsets:
  - labels:
      version: vuhpe
    name: vuhpe
```

查看VirtualService如下所示：

```
#$ kubectl -n dev get virtualservice productpage -o yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  creationTimestamp: 2018-12-14T11:59:07Z
  generation: 1
  name: productpage
  namespace: dev
  resourceVersion: "341256093"
  selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/virtualservices/productpage
  uid: a90d8198-ff97-11e8-bbda-de3c7d18d080
spec:
  hosts:
  - productpage
  http:
  - route:
    - destination:
        host: productpage
        subset: vuhpe
```

使用Istio Gateway访问应用

为了能够访问Productpage应用，用户需要在dev命名空间下部署Gateway资源如下所示：

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: bookinfo-gateway
  namespace: dev
spec:
  selector:
    istio: ingressgateway
  servers:
  - hosts:
    - '*'
    port:
      name: http
      number: 80
      protocol: HTTP
```

将以上内容保存到bookinfo-gateway.yaml中，并通过Kubectl在集群中创建：

```
$ kubectl -n dev create -f bookinfo-gateway.yaml
gateway.networking.istio.io/bookinfo-gateway created
```

创建完Gateway之后，需要手动绑定Productpage服务和Gateway绑定：

```
$ kubectl -n dev edit virtualservice productpage
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  creationTimestamp: 2018-12-14T11:59:07Z
  generation: 1
  name: productpage
  namespace: dev
  resourceVersion: "341256093"
  selfLink: /apis/networking.istio.io/v1alpha3/namespaces/dev/virtualservices/productpage
  uid: a90d8198-ff97-11e8-bbda-de3c7d18d080
spec:
  # 添加gateways节点绑定bookinfo-gateway
  gateways:
  - bookinfo-gateway
  hosts:
  # 添加需要监听的域名
  - productpage.yunxiao.com
  - productpage
  http:
  - route:
    - destination:
        host: productpage
        subset: vuhpe
```

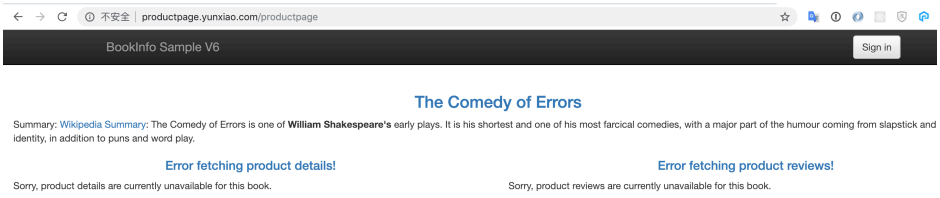
在完成VirtualService完成Gateway绑定后，用户就可以通过集群的istio Gateway访问应用，可以通过以下命令获取Istio Gateway的外网地址：

```
$ export INGRESS_HOST=$(kubectl -n istio-system get service istio-ingressgateway -o
jsonpath='{.status.loadBalancer.ingress[0].ip}')
$ export INGRESS_PORT=$(kubectl -n istio-system get service istio-ingressgateway -o
jsonpath='{.spec.ports[?(@.name=="http")].port}')
```

此时只需要在本地添加DNS设置，即可通过Gateway访问ProductPage应用：

```
182.92.244.178 productpage.yunxiao.com
```

打开浏览器访问http://productpage.yunxiao.com/productpage，如下所示：

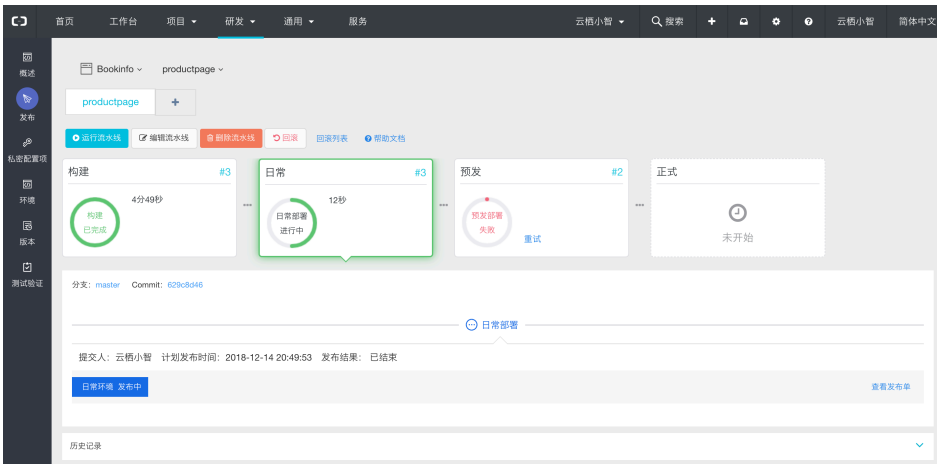


不过由于目前我们只部署了Productpage应用，因此当前页面会提示“Error fetching product details!”和“Error fetching product reviews!”的错误信息。

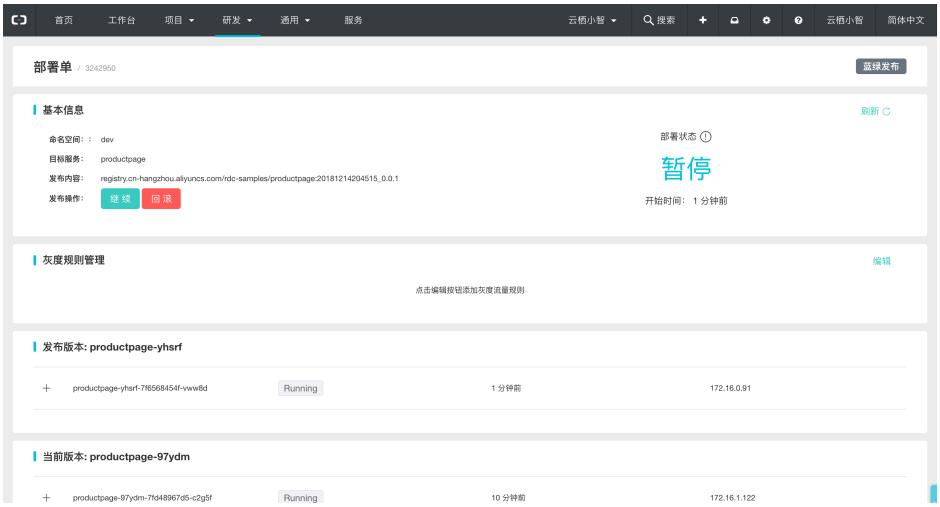
使用蓝绿发布

在Productpage应用第一次发布时由于集群中并不存在任何资源，因此云效会直接创建资源并且将部署设置为成功。在第二次部署应用时，由于底层资源已存在，再次运行流水线，会触发正式的蓝绿部署流程。修改productpage.html文件如下所示：

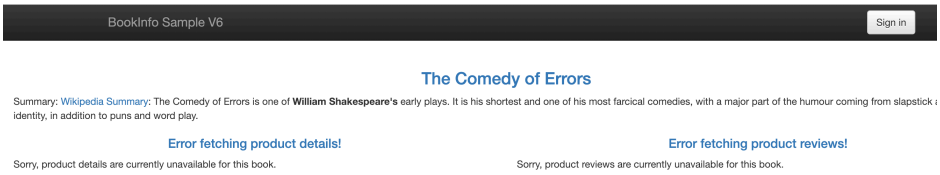
并重新触发流水线，此时当进入到部署阶段后部署任务会进入等待状态：



点击查看发布单按钮，进入发布单页面，可以看到云效为当前服务创建了两个Deploymet版本：



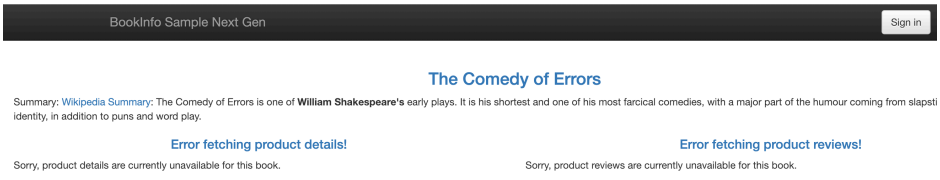
在默认情况下，当前应用的所有流量依然全部指向旧版的应用实例，访http://productpage.yunxiao.com/productpage可以验证当前应用的流量情况:



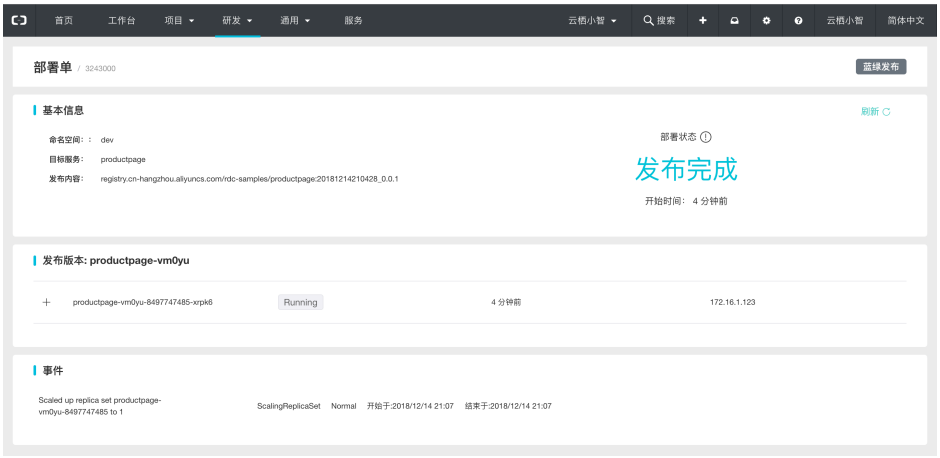
在部署暂停时如果希望某些流量能够进入到新版应用，例如，某些特定的URL或者是HTTP Header中包含特定值的请求。那可以直接在发布单中编辑灰度规则即可。例如，我们希望所有/productpage的请求都直接进入新版，那如下所示，添加一条灰度规则即可：

预览生成的YAML并下发规则：

在灰度规则更新成功后，此时如果再次访问应用，那流量则会进入到新版的ProductPage中：



在确认新版应用能够按照预期工作后，在发布单点击继续按钮，即可完成本次发布过程。发布完成后云效会自动移除旧版应用实例，并且将所有流量规则指向新版本。



小结

在这部分中，我们完成了对bookinfo中productpage应用的发布，通过使用云效可以让用户几乎0成本的将应用接入到Istio模式，并且使用蓝绿部署模式安全的对软件进行升级和发布。

Kubernetes分批发布

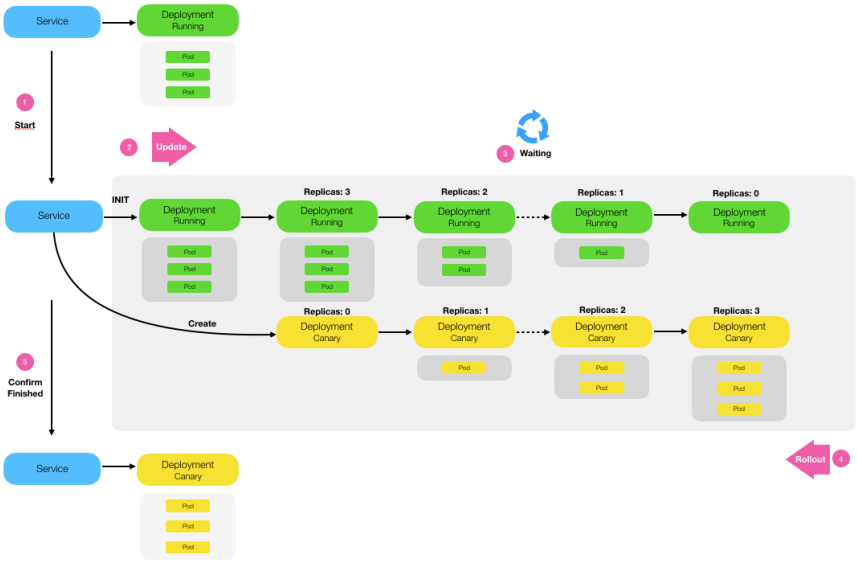
本文将介绍如何在云效中使用分批发布功能部署Kubernetes下的应用程序

前置条件

- 已在当前企业中导入可用的Kubernetes集群，未导入的用户可以参考文档导入Kubernetes集群

分批发布原理

云效中分批发布原理如下所示：



在执行分批发布过程中，云效会自动为当前Service关联的Deployment实例创建副本，并通过控制2个版本Deployment的Replicas数量实现，应用的分批发布。 在整个发布过程中Service的流量会同时转发到2个版本的Deployment实例中。在应用升级完成后，就版本的Deployment会自动删除。

部署配置

进入到环境的部署配置页面，切换当前环境的部署方式到**Kubernetes部署**，如下所示：

配置项	说明
发布策略	选择当前环境的发布策略，目前支持Kubernetes的原生滚动升级以及云效的分批发布策略
发布批次	发布分批总数，当发布批次数>当前应用实际的副本数时，已当前副本数为准
集群	当前应用发布的目标Kubernetes集群
命名空间	当前服务所在的命名空间

服务	当前部署所对应的Service对象
容器	如果当前Service已关联Deployment，那需要指定升级的Container对应的名称，用以兼容单Pod多容器的模式

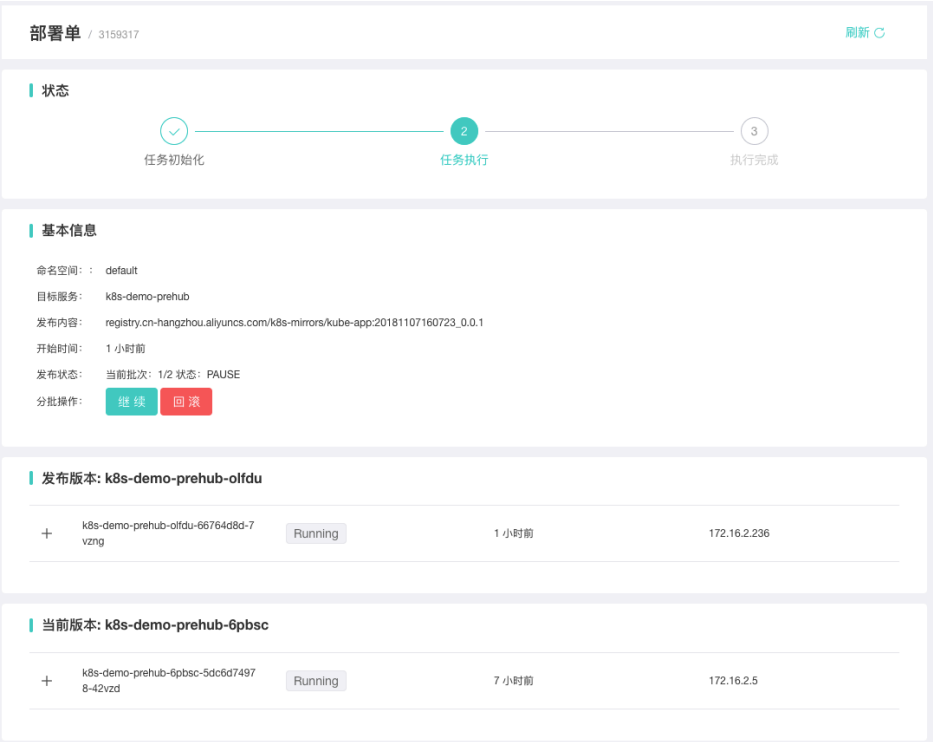
分批发布策略：

- 分批发布第一批暂停：按批次对应用进行升级，并且在第一批次执行完成后，暂停。等待用户确认是否继续后续批次的发布，后续批次发布过程不再暂停。
- 分批发布每批暂停：按批次对应用进行升级，在每批发布执行完成后都需要用户确认是否继续下一批次的任务

执行分批发布

在触发部署阶段后，用户可以通过流水线的部署单按钮进行**部署单**：

进入到发布单后，用户可以查看当前应用的发布状态：



在发布单页面中，用户可以执行以下操作：

操作	说明
继续	继续执行下一批次发布任务

回滚	回滚当前应用到部署前状态

常见问题

Q: 第一次部署部署应用程序时，为什么没有分批暂停？

A：当用户首次部署应用时，由于当前服务（Service）未关联任何的部署组（Deployment）实例，因此发布过程不会进行分批。云效会直接按照当前部署配置中的**发布批次数**作为应用的副本数，并直接初始化Deployment资源

Q：当Deployment的Replicas副本数为1时，为什么流水线会部署失败？

A：当Service关联Deployment的副本数为1时，不满足分批发布的要求，请手动调整Deployment的副本数后重试

Q：为什么实际的分批数和我设置的分批数不同？

A：为了避免分批发布对用户部署的影响，当分批发布数大于当前用户应用的副本数时，会以当前已有的副本数，作为分批发布的实际批次数

Q： 为什么分批发布时，应用访问请求可能会出现异常

A：默认Service在进行流量转发时只要Pod状态为Running就会转发流量，而应用可能还在启动中。 请为Deployment配置Readiness探针

流水线上的部署任务

部署任务，是云效流水线上的一类任务，它负责把构建得到的包，部署到运行环境并启动。

部署任务的运行

部署任务一般不需要在运行时输入信息，就会自动运行。运行期间和运行结束后，可以在页面下方点击“查看发布单”，查看更多细节：

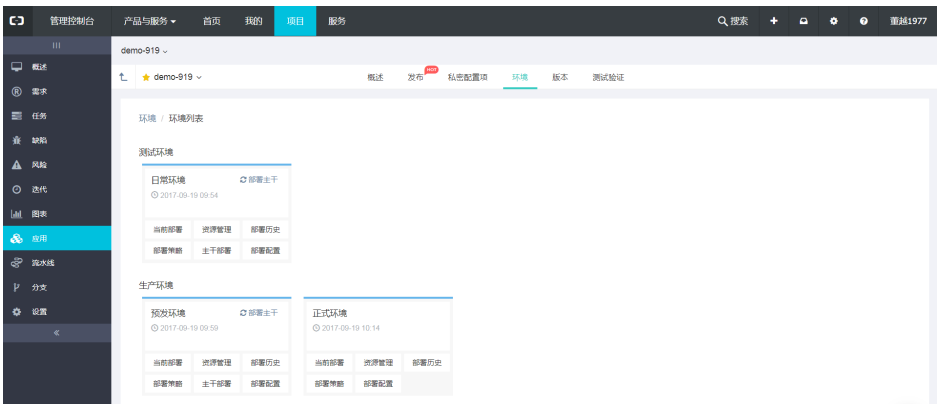
部署任务的配置

在流水线编辑页面，添加任务时，请选择“部署”，并填写其基本配置：

配置的核心思路是，选择合适的包，部署到合适的地方。其中，选择合适的包，是“应用”和“包标签”这两项决定的。部署到合适的地方，是“应用”和“环境”这两项决定的。

“包标签”是构建时用来区分同一个应用的不同用途（比如为不同运行环境）的包，而打上的标签。详见**流水线上的构建任务**中，对包标签的介绍。这里是选择本次部署所需要的包对应的标签。

“应用”和“环境”的概念见**这里**。环境的配置，包括环境关联到哪些机器、部署时使用的脚本等等。详见部属配置系列文档，比如**部署配置：通过脚本部署**。环境的配置是从应用入口进入的，进入后选择“环境”菜单项：



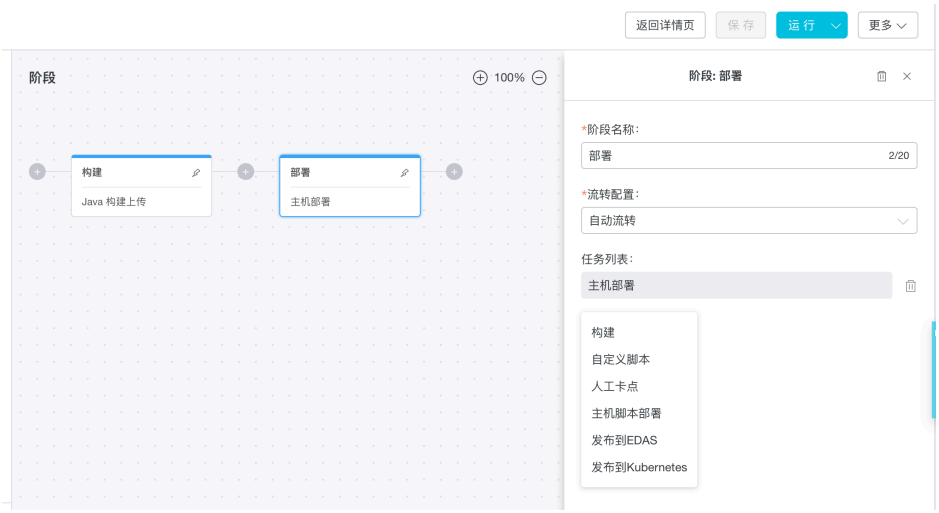
流水线上的部署任务（新版）

本文档适用于2019-07-01之后创建的企业。

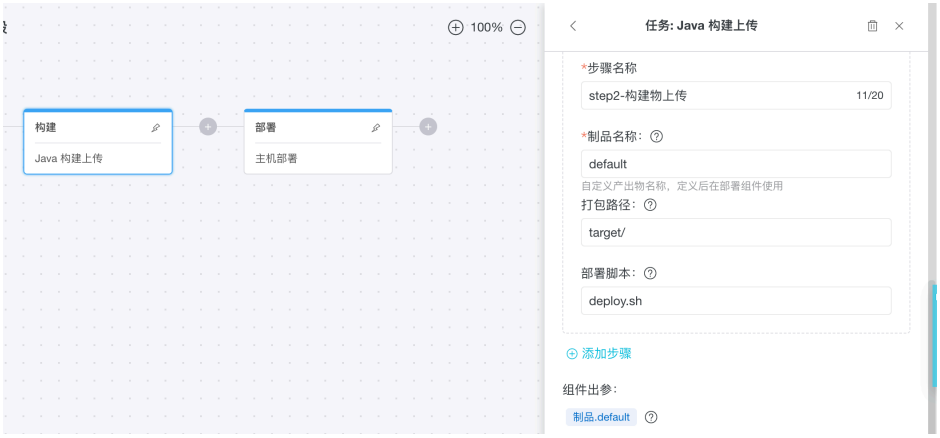
部署任务，是云效流水线上的一类任务，它负责把构建得到的包，部署到运行环境并启动。

部署任务的配置

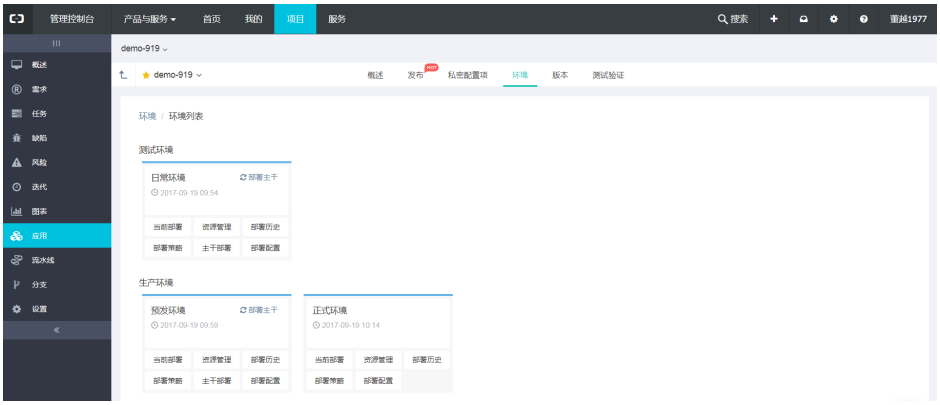
在流水线编辑页面，添加任务时，请选择“主机脚本部署”，“发布到EDAS”，“发布到Kubernetes”中的其中一个，并填写其基本配置：



在任务中，选择上游构建任务输出的制品。创建应用及环境。



“应用”和“环境”的概念见这里。环境的配置，包括环境关联到哪些机器、部署时使用的脚本等等。详见部属配置系列文档，比如部署配置：通过脚本部署。环境的配置是从应用入口进入的，进入后选择“环境”菜单项：



部署任务的运行

部署任务一般不需要在运行时输入信息，就会自动运行。运行期间和运行结束后，可以在页面下方点击“打开发布单”，查看更多细节：

使用“git pull”的方式更新应用

如果您的应用不需要打包，在生产服务器上直接通过git pull的方式进行更新，那么可以按照如下的方式进行操作。

release文件

在您的代码库根目录中添加<应用名>.release文件（如果不存在的话）。内容如下：

```
code.language=scripts

# 将当前的git版本号写入元信息文件
build.command=git rev-parse HEAD > rdc_build_meta

# 告诉云效把元信息文件打包成package.tgz
build.output=rdc_build_meta
```

部署配置

按照如下方式进行部署配置（您可以在应用->环境->部署配置中找到如下的配置表单）。

下载路径：/home/admin/package.tgz (需要您保证/home/admin目录存在，或者替换成实际存在的某个目录)

解压目录：/home/admin/package-explode （可以按照您的需求，替换成别的目录）

Stop：echo noops (如果不需要stop，随便填即可；如果需要，按实际情况填写。)

Start：cd /home/admin/app && git fetch && git checkout `cat /home/admin/package-explode/rdc_build_meta`（这条命令把构建时打包的rdc_build_meta文件解压出来，然后checkout到文件中指定的版本）

执行用户：admin （这个例子中使用的是admin用户进行部署，您可以替换成实际的用户）

添加Agent失败FAQ

我在机器上安装了Agent，但在企业的机器列表中看不到

请按照下列步骤依次排查。执行完每一步之后，请确认问题是否解决，若未解决，请继续尝试后续步骤。

目前agent只支持64位的Linux操作系统。

确认您是否曾经使用另一个企业的agent安装命令在该机器上执行过，如果是，请删除/usr/sbin/staragent_sn（正常情况下该文件内容为机器SN，SN为机器唯一标识，并与特定企业绑定），并重装agent。

执行命令cat /usr/sbin/staragent_sn查看内容。若文件内容为空，删除此文件，并重装agent（PS：请勿手动修改该文件）。

执行命令/home/staragent/bin/staragentctl status查看agent状态。若输出异常（比如ServerAddr为空，或者报错），请执行命令cat /home/staragent/conf/staragent.conf查看文件内容，如果文件存在，且其中的URL为rdc-xxx.aliyuncs.com或者staragent-configservice.aliyuncs.com，则为正常。如果不是，有可能是您的机器之前安装过其它产品的agent，请重装云效agent。

查看cat /home/staragent/conf/channels.conf是否存在，如果不存在，请执行命令：curl 'http://<从staragent.conf中获取的服务URL>/api/configservice?action=findChannelListForAgent&agentIpList=101.37.119.155%2C10.80.237.52&needAllChannels=true&serviceTag=ea263ff8-2d60-48f0-86c4-33a04214cad9&version=2'，如果结果类似{"appCode":"_successful_","msg":"","restCode":200,"result":{"allChannels":[],"channelIPPort":[{"ip":"100.100.18.88","port":8000},{"ip":"100.100.18.89","port":8000},{"ip":"100.100.45.99","port":8000},{"ip":"182.92.29.36","port":8000},{"ip":"182.92.29.39","port":8000},{"ip":"100.100.45.100","port":8000}]，请尝试重启agent（参看下面的agent基础操作）。如重启后/home/staragent/conf/channels.conf仍不存在，请提交工单联系我们。如果不能返回类似结果，则表示您的机器到<从staragent.conf中获取的服务URL>的连接有问题。有可能是在安装agent时候，选错了区域。请在添加机器页面选择正确的区域，生成agent安装命令，重装agent。

执行命令cat /home/staragent/conf/channels.conf查看该文件，内容会是一个ip+port的列表，尝试运行telnet <ip> <port>，只要任意一个连通，则服务正常；如果全部不通，请检查您的网络。

EDAS的agent与云效的agent不能共存。如果您的机器上安装了EDAS的agent，请彻底卸载，或重置操作系统，再尝试安装云效agent。

附agent基础操作：

```
启动：/home/staragent/bin/staragentctl restart;
重启：/home/staragent/bin/staragentctl restart;
查看状态：/home/staragent/bin/staragentctl status;
卸载：
   1. /home/staragent/bin/staragentctl stop;
   2. rm -rf /home/staragent;
   3. rm /usr/sbin/staragent_sn
```

若根据以上排查手段依然未能找到问题，请提交工单联系我们。

在ECS上使用容器镜像

本文描述如何在ECS上使用云效上构建出的镜像。

镜像的来源可以是**应用构建**（2019-07-01之前创建的企业可见），或者**构建**任务（2019-07-01之后创建的企业可见）中的“构建镜像上传”步骤。

配置的方式与**通过脚本部署**和**流水线上的部署任务（老版）**或者在**流水线上的部署任务（新版）**类似，本文仅介绍不同之处。

前提条件

使用**应用构建**或者**构建**任务构建出镜像。

部署配置

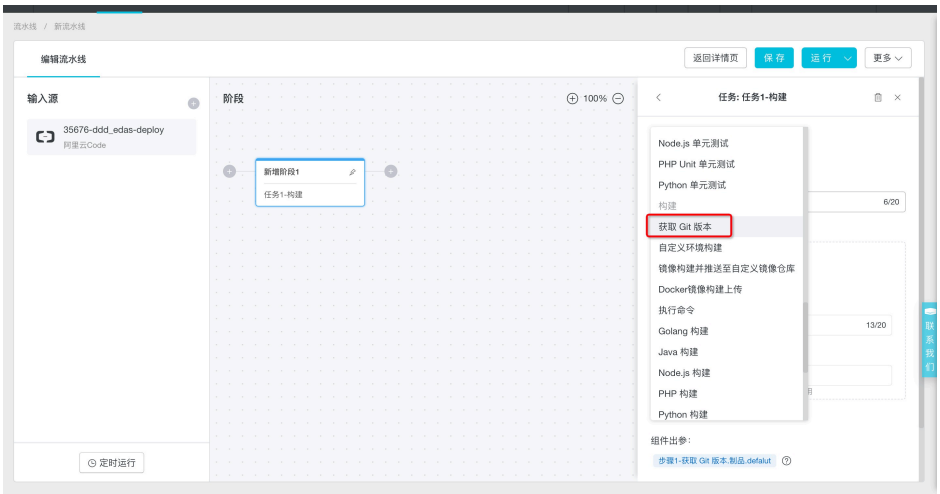
如上图所示

1. 留空“下载路径”
2. 在“部署脚本”中，使用$imageId来表示**应用构建**中构建出来的完整镜像地址。

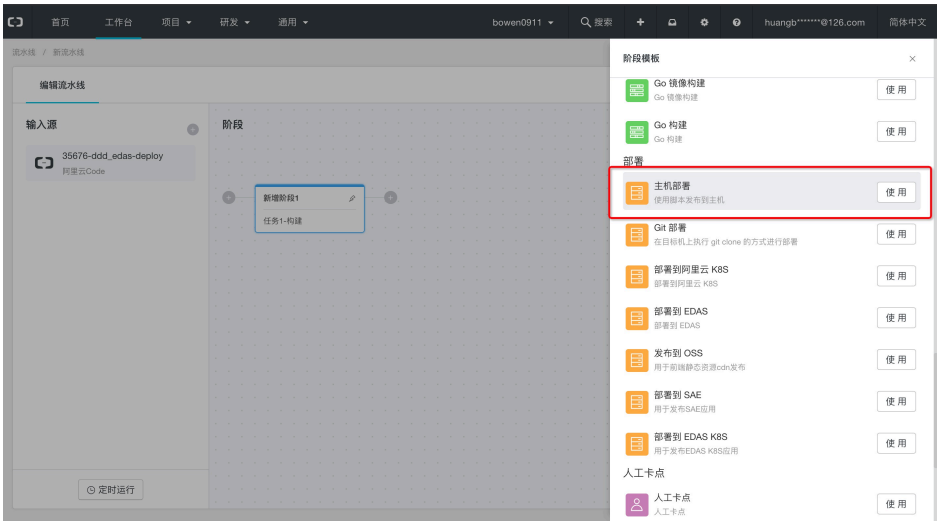
无构建部署

如果部署时不需要拉取包，而只是通过git更新代码，可是使用我们无构建部署的能力。

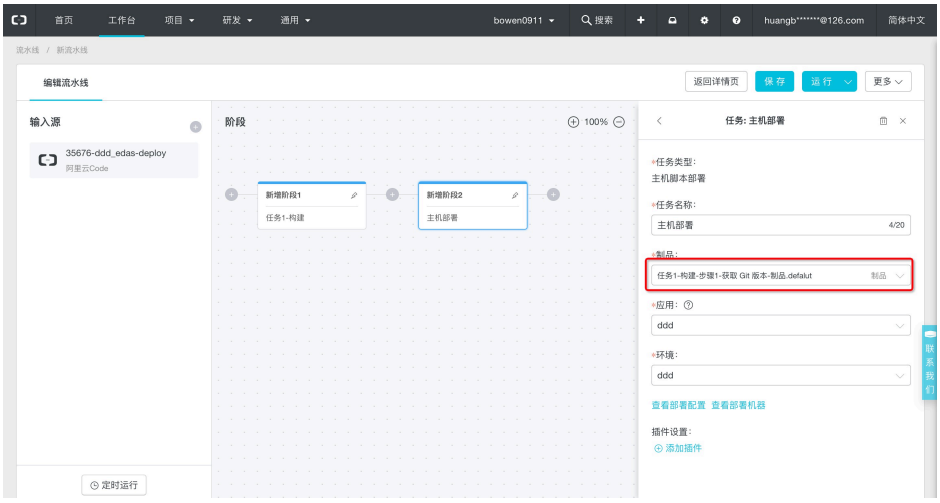
首先给流水线添加一个代码源，新增一个空的阶段，在阶段中选择构建任务，并添加一个 获取git版本的步骤。



然后再新建一个阶段，选择主机部署模板。



在主机部署任务中制品选择为 获取git版本输出的制品，并选定应用和环境。



打开该环境的部署配置页面。当运行流水线时，上游会传递三个变量供部署配置使用。GIT_REPO: git地址 GIT_BRANCH: git分支COMMIT_ID: 提交版本号

部署脚本可以直接使用这几个变量完成拉取代码操作。服务器上需要自行解决拉取代码的认证问题，比如可以

将服务器上ssh公钥配置到代码库中。

Helm私有仓库

使用Helm私有仓库

开通Helm私有仓库

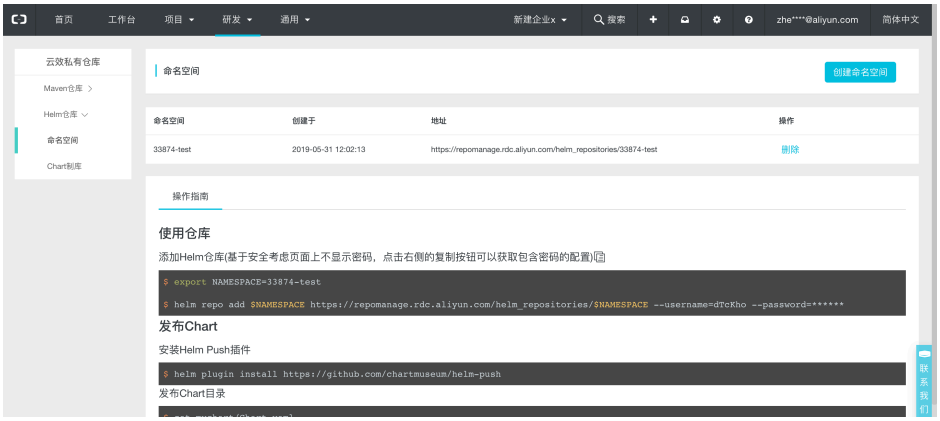
点击开通私有仓库用户在云效开通私有仓库服务后，会默认打开Helm私有仓库支持

创建命名空间

> 当前云效每个企业最大命名空间数为5个

进入私有仓库，并进入Helm私有仓库命名空间选项，并点击“创建命名空间”按钮，按照对话框提示输入需要创建的仓库名称。

创建完成后，云效会自动为当前命名空间生成仓库访问地址，以及独立的用户名/密码



添加Helm仓库

查看操作指南，并复制添加命令以及用户名密码（仓库用户名密码会自动复制到粘贴板）


```
export NAMESPACE=<Your NAMESPACE>
helm repo add $NAMESPACE https://repomanage.rdc.aliyun.com/helm_repositories/$NAMESPACE --
username=<Your Username> --password=<Your Password>
helm repo list # 查看已添加的仓库
```

上传Chart包到Helm仓库

云效Helm私有仓库兼容官方Helm协议，可以使用Helm-Push插件上传Chart

安装Helm-Push插件

```
helm plugin install https://github.com/chartmuseum/helm-push
```

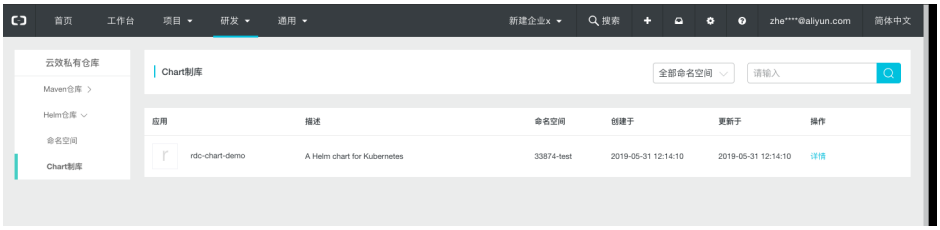
准备Chart

```
$ helm create rdc-chart-demo
Creating rdc-chart-demo
```

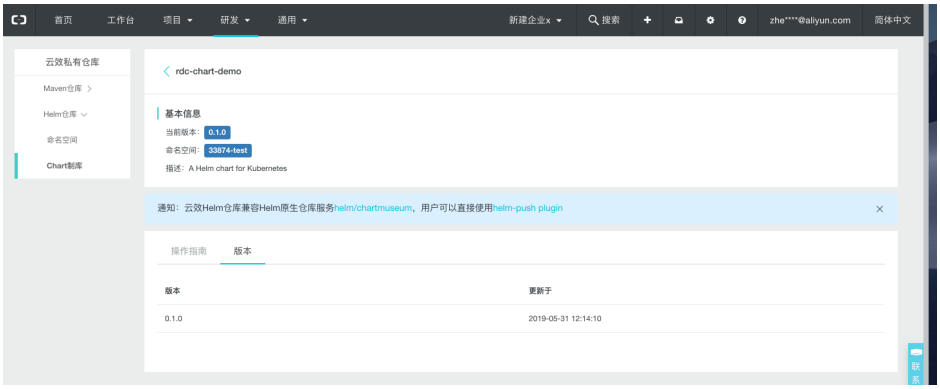
使用Helm Push上传Chart

```
$ helm push rdc-chart-demo $NAMESPACE
Pushing rdc-chart-demo-0.1.0.tgz to 33874-test...
Done.
```

查看上传结果，进入云效Helm私有仓库,并进入Chart制库，如下所示：



查看已上传的Chart实例，在详情中，我们可以查看当前Chart的详细信息以及相应的版本记录：



部署Chart到Kubernetes集群

Chart上传到云效Helm私有仓库之后，用户可以使用Helm命令行工具将Chart部署到Kubernetes集群：

更新版本索引：

```
$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "33874-test" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete.  Happy Helming!
```

查询Chart

```
$ helm search $NAMESPACE
NAME                        CHART VERSION    APP VERSION    DESCRIPTION
33874-test/rdc-chart-demo    0.1.0            1.0            A Helm chart for Kubernetes
```

部署Release实例：

```
$ helm install 33874-test/rdc-chart-demo
NAME:   vocal-billygoat
LAST DEPLOYED: Fri May 31 12:19:13 2019
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Service
NAME                         TYPE       CLUSTER-IP     EXTERNAL-IP  PORT(S)  AGE
vocal-billygoat-rdc-chart-demo  ClusterIP  172.19.233.76  <none>       80/TCP   1s

==> v1beta2/Deployment
NAME                         DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
vocal-billygoat-rdc-chart-demo  1        1        1           0          1s

==> v1/Pod(related)
NAME                                      READY  STATUS    RESTARTS  AGE
vocal-billygoat-rdc-chart-demo-6dc7944d84-nfvrg  0/2    Init:0/1  0         0s


NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=rdc-chart-demo,release=vocal-billygoat" -o
jsonpath="{.items[0].metadata.name}")
  echo "Visit http://127.0.0.1:8080 to use your application"
  kubectl port-forward $POD_NAME 8080:80
```

Maven私有仓库

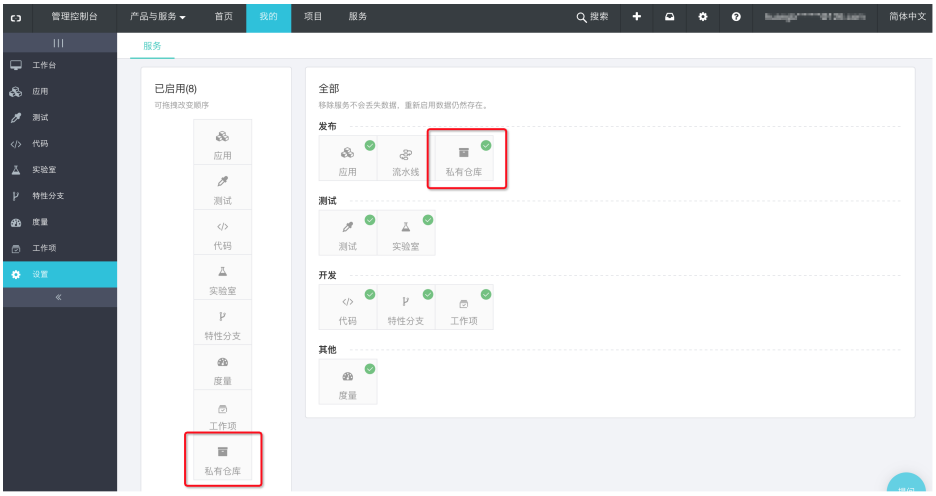
Maven仓库

背景

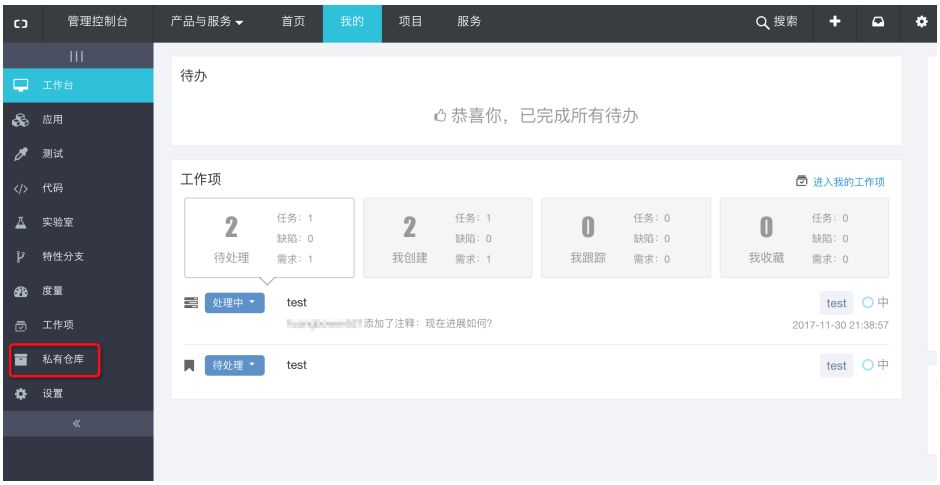
在云效中如果需要上传、下载私有的二方库，可以使用云效的企业级Maven私有仓库服务。

将私有仓库服务加入侧边栏

点击‘我的’链接，选择左侧菜单栏中的‘设置’按钮，将‘私有仓库’服务加入到左侧菜单栏中。



这样在左侧菜单栏会看到‘私有仓库’链接。



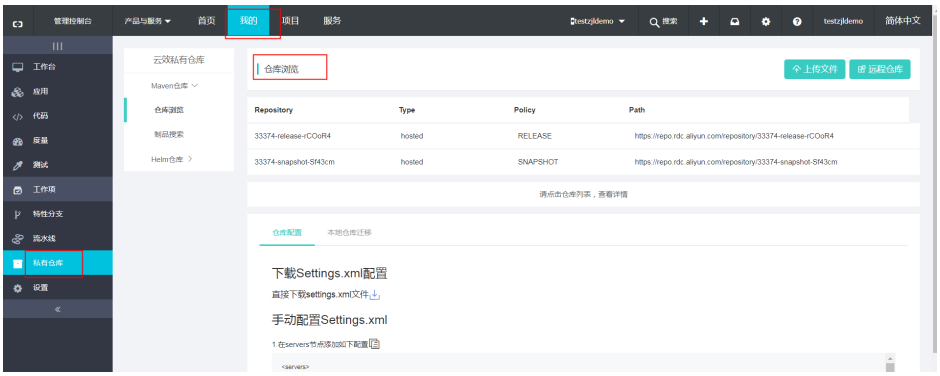
开通仓库

虽然启用了私有仓库服务，但云效并没有真正的为您创建企业级Maven私有仓库。点击左侧菜单栏’私有仓库’链接后，如果您是企业管理员，会出现以下界面：

‘点击开通’即可开通仓库服务。

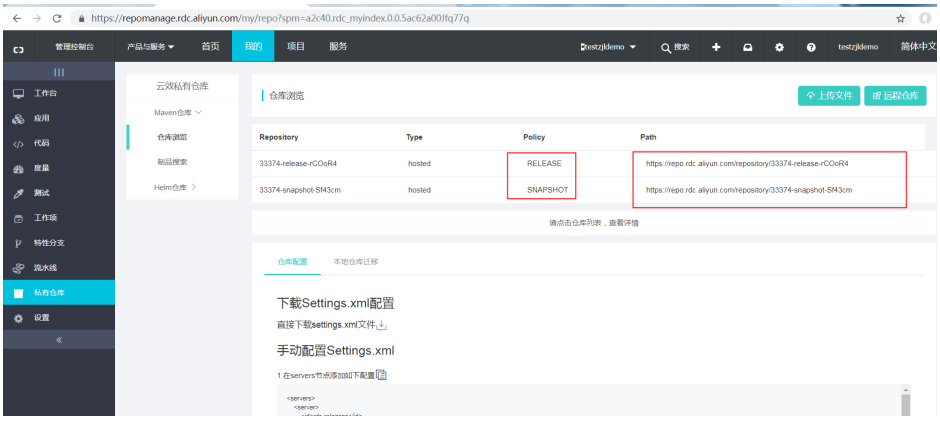
企业的普通用户并没有开通仓库的权限，则需要联系您的企业管理员进行开通操作。

开通成功以后显示界面如下：



仓库地址

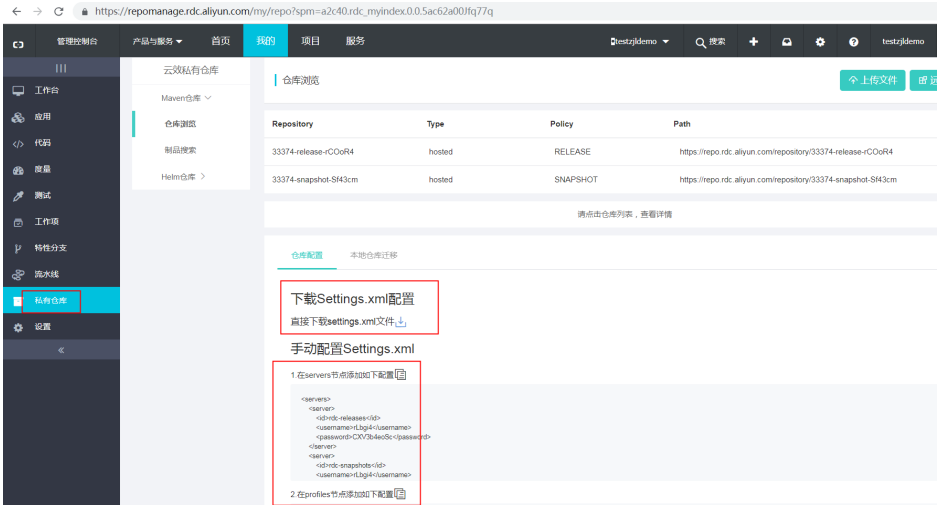
云效会自动为该企业生成两个Maven私库，一个是Release仓库，用于存储正式版本的二方库；另一个是Snapshot仓库，用于存放Snapshot版本的二方库。



settings.xml配置

私有仓库不允许匿名上传和下载二方库，云效为每个私有仓库生成了相应的用户名和密码。请注意不要泄露该

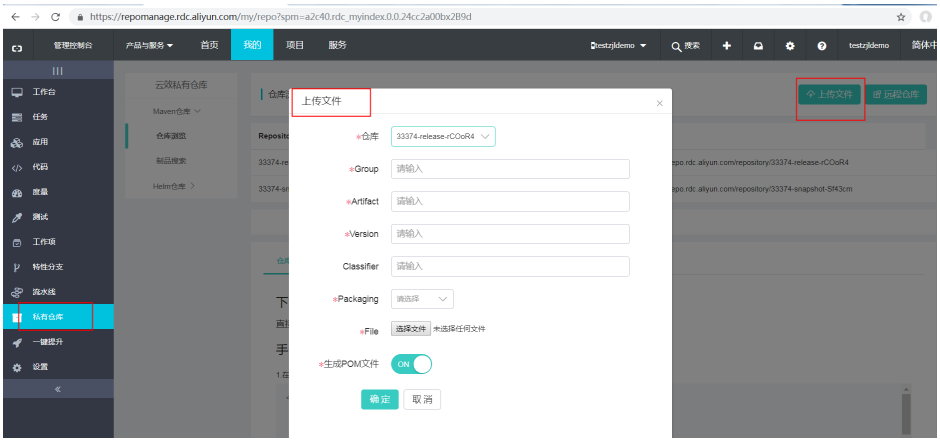
用户名和密码。



用户可以通过该页面下载完整的settings.xml文件,也可以根据自己的需求在settings.xml文件中添加公共仓库的镜像地址。

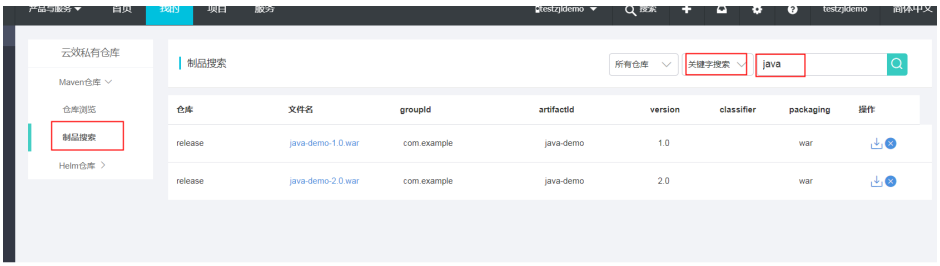
上传二方库

用户可以通过UI上传二方库。目前支持通过GAV的模式进行上传，单个二方库的大小限制为300M。

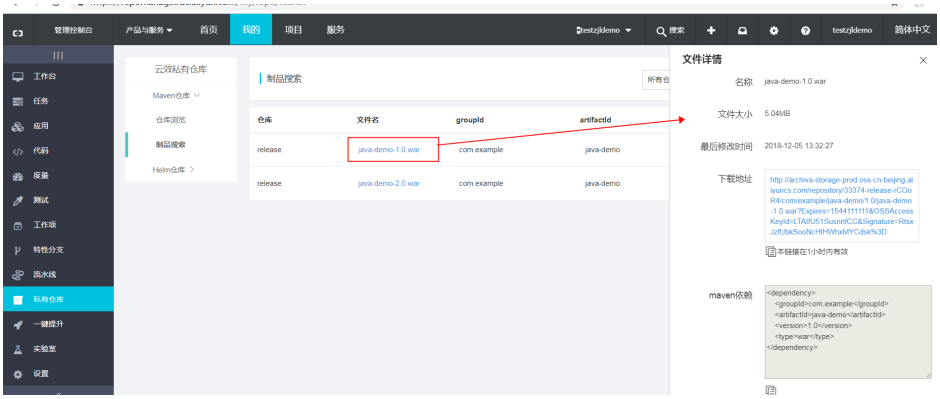


检索

对二方库的检索支持关键字搜索和GAV搜索两种模式。



用户可以查看检索出来的二方库的基本信息，也可以下载二方库。



在云效构建中使用私有仓库服务

默认情况下如果用户开通了Maven私有仓库服务，那么通过云效构建时，云效会自动尝试从该企业的私有仓库中拉取需要的二方库，用户无需进行额外的配置。而如果用户自己在代码库根目录中定制了settings.xml文件，那么就需要用户自行把私有仓库的配置信息添加到该文件中。具体可以通过前面的介绍来了解私有仓库的settings.xml配置。

用户可以查看在云效构建中使用Maven私有仓库服务了解更多内容。

公共代理库

maven.aliyun.com代理了很多公共的maven仓库。使用maven.aliyun.com中的仓库地址作为下载源，速度更快更稳定。

代理的仓库列表

仓库名称	代理源地址	使用地址
central	https://repo1.maven.org/maven2/	https://maven.aliyun.com/repository/central 或

		https://maven.aliyun.com/nexus/content/repositories/central
jcenter	http://jcenter.bintray.com/	https://maven.aliyun.com/repository/jcenter 或 https://maven.aliyun.com/nexus/content/repositories/jcenter
public	central仓和jcenter仓的聚合仓	https://maven.aliyun.com/repository/public 或 https://maven.aliyun.com/nexus/content/groups/public
google	https://maven.google.com/	https://maven.aliyun.com/repository/google 或 https://maven.aliyun.com/nexus/content/repositories/google
gradle-plugin	https://plugins.gradle.org/m2/	https://maven.aliyun.com/repository/gradle-plugin 或 https://maven.aliyun.com/nexus/content/repositories/gradle-plugin
spring	http://repo.spring.io/libs-milestone/	https://maven.aliyun.com/repository/spring 或 https://maven.aliyun.com/nexus/content/repositories/spring
spring-plugin	http://repo.spring.io/plugins-release/	https://maven.aliyun.com/repository/spring-plugin 或 https://maven.aliyun.com/nexus/content/repositories/spring-plugin
grails-core	https://repo.grails.org/grails/core	https://maven.aliyun.com/repository/grails-core 或 https://maven.aliyun.com/nexus/content/repositories/grails-core
apache snapshots	https://repository.apache.org/snapshots/	https://maven.aliyun.com/repository/apache-snapshots 或 https://maven.aliyun.com/nexus/content/repositories/apache-snapshots

配置指南

maven配置指南

打开maven的配置文件(windows机器一般在maven安装目录的conf/settings.xml)，在

<mirrors></mirrors>标签中添加mirror子节点:

```
<mirror>
    <id>aliyunmaven</id>
    <mirrorOf>*</mirrorOf>
    <name>阿里云公共仓库</name>
    <url>https://maven.aliyun.com/repository/public</url>
</mirror>
```

如果想使用其它代理仓库,可在<repositories></repositories>节点中加入对应的仓库使用地址。以使用spring代理仓为例：

```
<repository>
    <id>spring</id>
    <url>https://maven.aliyun.com/repository/spring</url>
    <releases>
        <enabled>true</enabled>
    </releases>
    <snapshots>
        <enabled>true</enabled>
    </snapshots>
</repository>
```

gradle配置指南

在build.gradle文件中加入以下代码:

```
allprojects {
    repositories {
        maven { url 'https://maven.aliyun.com/repository/public/' }
        mavenLocal()
        mavenCentral()
    }
}
```

如果想使用maven.aliyun.com提供的其它代理仓，以使用spring仓为例，代码如下:

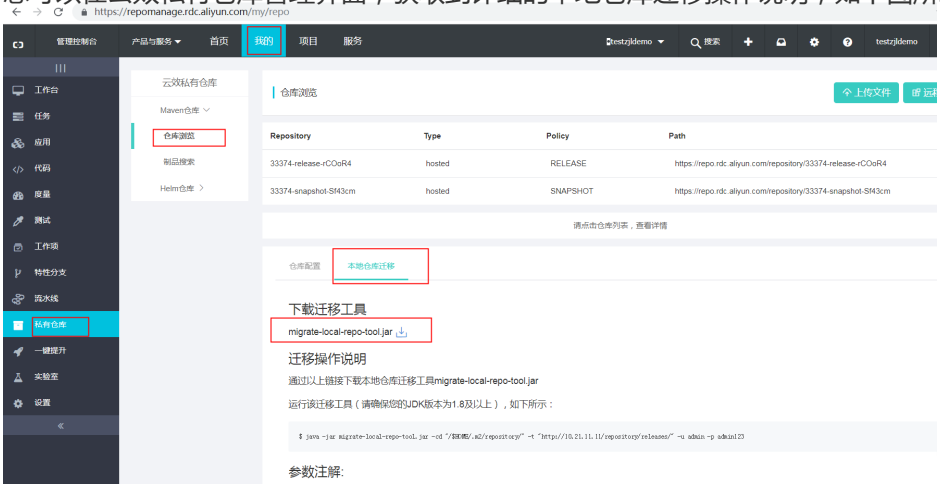
```
allProjects {
    repositories {
        maven { url 'https://maven.aliyun.com/repository/public/' }
        maven { url 'https://maven.aliyun.com/repository/spring/'}
        mavenLocal()
        mavenCentral()
    }
}
```

将已有私库迁移至云效私库

本文档帮助您将已有Maven私库中的制品包批量迁移到云效的Maven私库中。

本地制品迁移

您可以在云效私有仓库管理界面，获取到详细的本地仓库迁移操作说明，如下图所示：



操作步骤：

1.下载迁移工具migrate-local-repo-tool.jar

2.在您本地运行该迁移工具，（请首先确保您的JDK版本为1.8及以上）。运行命令如下：

```
java
 -jar migrate-local-repo-tool.jar
 -cd "/$HOME/.m2/repository/"
 -t "http://10.21.11.11/repository/releases/"
 -u admin
 -p admin123
```

参数注解：

-cd 您要迁移的本地目录，例如：/$HOME/.m2/repository/

-t 目标仓库地址（您可以在【私有仓库】界面点击仓库地址，获取您的目标仓库地址）

-u 用户名

-p 密码

注：用户名和密码为您要上传的目标仓库用户名及密码，您可在setting.xml中获取对应仓库的username和password

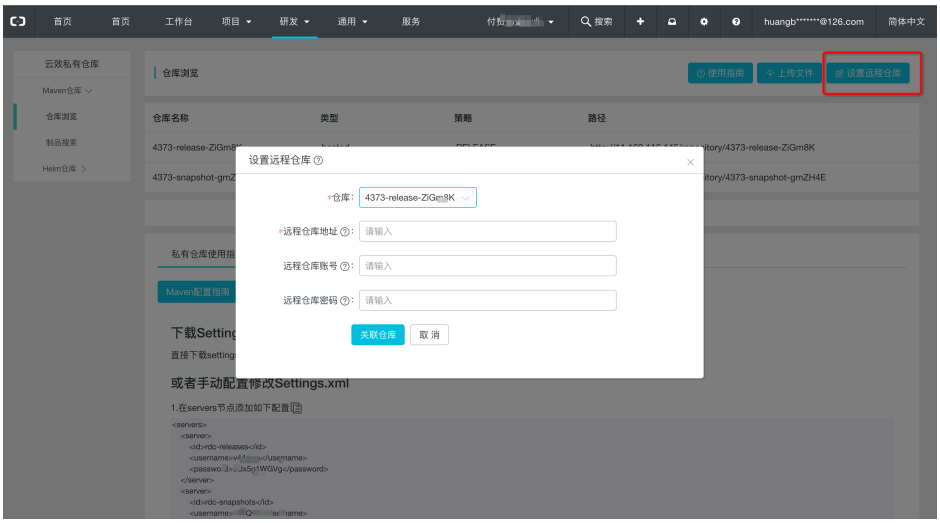
根据您的实际需求指定合适的参数，然后执行该命令，稍等片刻，您的本地仓库中的a制品将会被批量迁移到云

效中您所指定的Maven私库中。

如果迁移的本地目录中文件目录过多或者目录层级过深，可能会导致迁移命令卡死或者返回异常。推荐做法是只迁移你自己的私有制品到私有仓库中，构建时拉取公共制品包可以使用我们提供的**公共代理库**。比如假设你的私有制品都放置在/$HOME/.m2/repository/com/alibaba/**目录中，你可以将com/alibaba/**/目录复制一份到一个空的目录中，比如复制到/tmp/repo/中中，然后运行迁移命令时将-cd命令参数指定为/tmp/repo/，这样迁移工具只会迁移你的私有制品。

添加现有的私库作为云效私库的远程仓库

云效提供了关联其他仓库为远程仓库的功能。这样云效私库既具备上传下载包的能力，又具备代理其他仓库的能力。当使用云效私库下载包时，它也会尝试从远程仓库拉取包，并且缓存在云效私库。



远程仓库地址为您想要代理的私库地址，这个地址必须是公网可以访问的。如果该私库可以匿名访问，那么无需配置访问账号和密码。点击关联仓库可以保存配置。您随后可以修改配置或者解除关联。注意只有已经缓存在云效私库的制品包才能被搜索到。**注意**:如果远程仓库的网络环境不佳会导致拖慢云效私库的下载速度。

在云效构建中使用Maven私有仓库服务

当用户开通了Maven私有仓库服务后，云效会为用户生成两个私有仓库，一个用于存放release版本的二方库，一个用于存储SNAPSHOT版本的二方库。

release仓库地址示例:

```
https://repo.rdc.aliyun.com/repository/24409-release-87w1FL/
```

SNAPSHOT仓库地址示例：

```
https://repo.rdc.aliyun.com/repository/24409-snapshot-AA0Hx0/
```

云效构建时从私有仓库下载二方库

如果用户项目代码库的根目录没有Maven的settings.xml文件，那么云效构建时会为用户自动生成一个settings.xml文件。该文件不仅包括了maven.aliyun.com等公共仓库地址，也自动引入了该企业的两个私有仓库地址。所以用户通过云效构建时，无需任何额外配置就可以实现下载私有仓库中的二方库。

如果用户项目代码库的根目录定制了Maven的settings.xml文件，那么用户需要自行将私有仓库的配置信息添加到该文件中。具体可以参考Maven私有仓库服务。

通过云效上传二方库到私有仓库

如果想通过流水线发布二方库到私有仓库，可以先在项目代码库根目录的pom.xml中指定分发的仓库地址，示例如下：

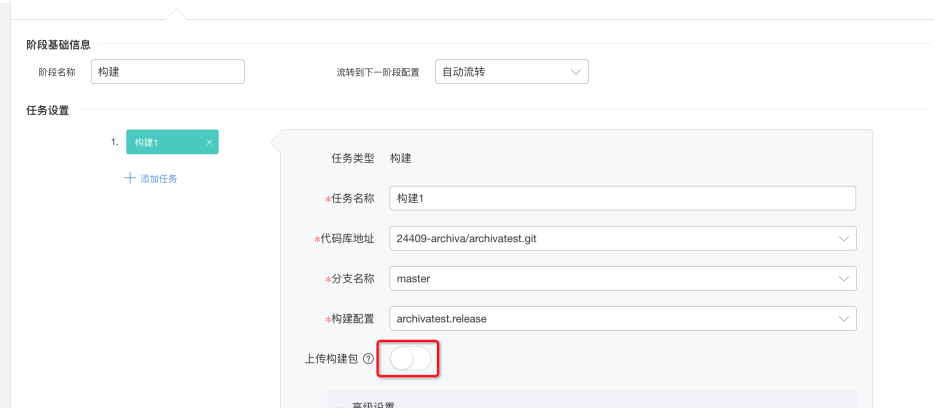
```
<distributionManagement>
  <repository>
    <id>rdc-releases</id>
    <url>http://repo.rdc.aliyun.com/repository/24409-release-87w1FL/</url>
  </repository>
  <snapshotRepository>
    <id>rdc-snapshots</id>
    <url>https://repo.rdc.aliyun.com/repository/24409-snapshot-AA0Hx0/</url>
  </snapshotRepository>
</distributionManagement>
```

项目代码库根目录的<应用名>.release中指定构建命令为上传二方库，例如：

```
build.command=mvn clean deploy -Dmaven.test.skip
```

如果你的pom.xml配置的软件包版本是以-SNAPSHOT结尾，比如版本为1.0-SNAPSHOT，会自动发布到snapshot仓；如果想发到release仓，可以将命令改为build.command=mvn clean deploy -Dmaven.test.skip -P release。

然后在云效中创建一条流水线，创建一个构建任务。示例配置如下：



注意：需要将上传构建包参数关闭。

也可以在构建命令中指定分发的仓库地址。方式是在Maven命令中指定-DaltDeploymentRepository参数。

```
build.command=mvn -DaltDeploymentRepository=rdc-releases::default::https://repo.rdc.aliyun.com/repository/24409-release-87w1FL/ deploy -Dmaven.test.skip
```

altDeploymentRepository指定了id::layout::url。在云效的Maven私有仓库服务中release仓库的id为rdc-releases。SNAPSHOT仓库的id为rdc-snapshots。layout一般使用默认值default，而url则为release仓库或SNAPSHOT仓库的url。上传到SNAPSHOT仓库的示例命令如下：

```
build.command=mvn -DaltDeploymentRepository=rdc-snapshots::default::https://repo.rdc.aliyun.com/repository/24409-snapshot-AA0Hx0/ deploy -Dmaven.test.skip
```

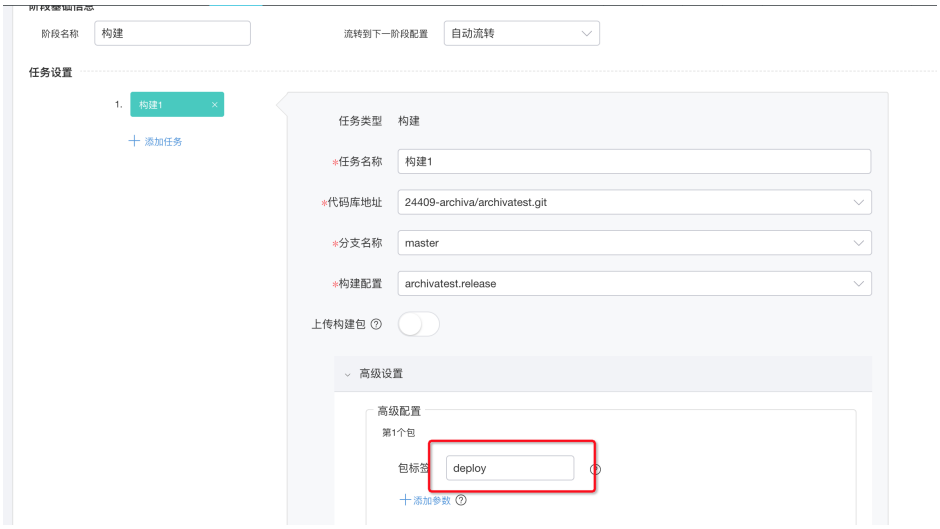
单应用同时支持应用构建和二方库发布

如果一个项目既要实现打包和部署，又要为其他项目提供SDK二方库，那么单个build.command配置就无法满足这种场景。您可以使用传入参数改变构建行为中的方式，使用PACKAGE_LABEL区分不同的构建命令。一个完整的例子如下。

在代码库根目录的<应用名>.release文件中指定如下配置项:

```
deploy.build.command=mvn -DaltDeploymentRepository=rdc-releases::default::https://repo.rdc.aliyun.com/repository/24409-release-87w1FL/ deploy -Dmaven.test.skip
```

这个配置项使用前缀deploy作为包标签。然后创建一条流水线，在构建任务中打开高级配置项，进行如下配置：



与上一个构建任务唯一不同的是这里将高级配置中的包标签的值改为deploy，这样触发构建时执行的就是deploy.build.command中指定的命令。

研发度量

研发总览

研发总览

功能概述

研发度量是RDC为企业用提供的研发效能度量的功能，通过对在RDC中企业的项目管理，代码活动， 应用发布， 测试等研发协同功能中进行数据的同步抽取清洗分析建模，为企业提供研发效能分析和解读。

度量指标定义

- 需求完成数： 统计时间段内，该团队完成的需求数； 这个指标通常用于度量整个团队的交付能力，越多代表团队的交付越多。
- 需求平均完成时长： 统计时间段内，该团队完成的需求从创建到终态（正常结束）的平均时长（自然日按天计算）；这个指标是用于度量团队的交付效率，越短代表团队的交付越快越敏捷。
- 新增需求数：统计时间段内，该团队被指派的需求数量；这个指标用于反映整个团队的负载。
- 新增缺陷数：统计时间段内，该团队被指派的缺陷数量；这个指标用于度量团队产生的缺陷数，间接反映质量。
- 缺陷Reopen率： 统计时间段内，该团队缺陷Reopen次数除以解决缺陷数；这个指标用于度量团队缺陷的修复质量，Reopen率越高代表返工越多。
- 缺陷平均修复时长：统计时间段内，该团队关闭的缺陷从创建到Fixed状态的平均时长（自然日按天计算）；这个指标用于度量开发团队缺陷的修复效率，时长越短，代表缺陷修复越快。
- 缺陷平均关闭时长：统计时间段内，该团队关闭的缺陷从Fixed到Closed状态的平均时长（自然日按天计算）；这个指标用于度量测试团队缺陷的验证效率，时长越短，代表缺陷被验证的越快。
- 人均提交代码量——统计时间段内，该团队提交的代码数量/人数；
- 代码规约扫描 ——统计时间段内，该团队相关应用通过规约扫描的Critical Issue， Major Issue数量，越低代表代码质量越高。

功能介绍

研发总览

研发总览提供企业研发关键指标和其趋势的分析展现能力

应用质量

应用质量对企业的应用构建发布的质量进行分析

项目进度

项目进度对企业的活跃项目进行进度追踪和效率分析