工具对接接口说明

1. Gitlab

接入验证方式

|  |  |
| --- | --- |
| 参数 | 说明 |
| url | gitlab的地址 |
| private\_token | 私有token |

获得组gorup：

*GitlabGroup* getGroup(*String groupName*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| groupName | 组名称 |

创建组gorup：

*GitlabGroup* createGroup(*String groupName*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| groupName | 组名称 |

获得gitlab用户：

*GitlabUser* getUser(*Integer userId*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| userId | 用户id |

创建gitlab用户 ：

*GitlabUser* createUser(*CreateUserRequest request*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| request | 创建用户的request，包含创建用户所需的所有信息：  private *String* email; private *String* password; private *Boolean* resetPassword; private *String* username; private *String* name; private *String* skype; private *String* linkedin; private *String* twitter; private *String* websiteUrl; private *String* organization; private *Integer* projectsLimit; private *String* externUid; private *String* provider; private *String* bio; private *String* location; private *Boolean* admin; private *Boolean* canCreateGroup; private *Boolean* skipConfirmation; private *Boolean* external; private *String* avatar;  可以使用构造函数，也可以使用getter和setter。构造函数原型如下：  public CreateUserRequest(*String name*, *String username*, *String email*) |

删除gitlab用户 ：

void deleteUser(*Integer targetUserId*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| targetUserId | 目标用户id |

修改gitlab用户 ：

*GitlabUser* updateUser(*Integer targetUserId*, *CreateUserRequest request*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| targetUserId | 目标用户id |
| request | 包含用户的新信息的request，详见创建用户接口说明 |

创建gitlab项目 ：

*GitlabProject* createProject(*String projectName*, *String groupName*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projectName | 项目名称 |
| groupName | 组名称 |

删除gitlab项目 ：

void deleteProject(*Serializable projectId*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projectId | 项目id |

分配项目成员 ：

*GitlabProjectMember* assignMemberToProject(*Integer projectId*, *Integer userId*, *GitlabAccessLevel accessLevel*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projectId | 项目id |
| userId | 用户id |
| accessLevel | 访问权限：  ***Guest***(10)  ***Reporter***(20) ***Developer***(30) ***Master***(40) ***Owner***(50) |

删除项目成员：

void removeMemberFromProject(*Integer projectId*, *Integer userId*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projectId | 项目id |
| userId | 用户id |

1. Jira

根据项目关键字获得项目：

*ProjectBean* getProjectByKey(final *String projectKey*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projecKey | 项目关键字 |

获得项目角色信息（Administrators, Developers和Users）：

*RolesBean* getProjectRoles(final *String projectKey*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projecKey | 项目关键字 |

获得所有项目：

*List*<*ProjectBean*> listProject();

|  |  |
| --- | --- |
| 参数 | 说明 |

创建项目：

void createProject(*Map projectDataMap*) throws *ToolClientException*;

|  |  |
| --- | --- |
| 参数 | 说明 |
| projecDataMap | 项目信息，所需的(Key, Value)如下：  ("notificationScheme", 10000);  ("description", "");  ("lead", "admin");  ("url", "");  ("projectTemplateKey", "com.atlassian.jira-core-project-templates:jira-core-project-management");  ("avatarId", 10200);  ("issueSecurityScheme", 10000);  ("name", "Example1");  ("permissionScheme", 0);  ("assigneeType", "PROJECT\_LEAD");  ("projectTypeKey", "business");  ("key", "EX1"); // key不可为空  ("categoryId", 10000); // 可省略 |

添加项目成员：

*ProjectRoleBean* assignMembersToProject(final *String projectKey*, final *Integer projectRoleId*, final *Map usersOrGroups*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| projecKey | 项目关键字 |
| projectRoleId | 分配给的项目角色的id，如Developers角色的id是10103，需要根据具体项目具体角色而定 |
| usersOrGroups | 亟待分配的成员信息Map，若是分配用户，则key为"user"，value为用户名所组成的数组；若是分配组，则key为"group"，value为组名所组成的数组 |

根据用户名获得用户：

*UserBean* getUserByUsername(final *String username*);

|  |  |
| --- | --- |
| 参数 | 说明 |
| username | 用户名 |

1. Confluence
2. 容器云平台
3. Apm