

JIRA REST API参考

JIRA 7.2.1

欢迎来到JIRA REST API参考。这页文档JIRA可用的REST资源，预计HTTP响应代码和样本请求一起。您可以使用JIRA的REST API构建使用JIRA的附加组件Atlassian的连接，开发JIRA和其他应用程序或脚本之间的互动与整合JIRA。

入门

学习JIRA REST API的最好办法是尝试一下。如果你在云中使用JIRA，Atlassian的连接是构建插件的JIRA一个简单的框架。尝试在我们JIRA插件教程之一Atlassian的Connect文档。对于JIRA服务器的JIRA REST API的高度概括，先从 JIRA REST API的家。

认证

与JIRA工作的任何身份验证将与REST API工作。在首选的身份验证方法是OAuth的（例子）和HTTP基本（使用SSL时），这是在这两个文件的JIRA REST API教程。其它支持的方法包括HTTP Cookie和信任的应用程序。

JIRA使用浏览器基于Cookie的身份验证，这样你就可以从Javascript在页面上调用REST和依赖于浏览器已经建立的验证。以再现的JIRA登录页面的行为（例如，以显示认证错误消息给用户）可以POST向资源。/auth/1/session

身份验证连接Atlassian的附加

Atlassian的带内置连接JIRA附加组件可以以轻松和一致认证与JIRA API请求，并验证查询参数的完整性使用JWT。咨询Atlassian的连接JWT认证文档，以了解更多信息。

URI结构

JIRA的REST API的通过URI路径可以访问的资源（数据实体）。要使用REST API，应用程序将HTTP请求并解析响应。该JIRA REST API使用JSON作为通信格式和标准的HTTP方法，如GET，PUT，POST和DELETE（请参阅下面的这些方法可用于每个资源API描述）。对于JIRA的REST API资源的URI的结构如下：

```
http://host:port/context/rest/api-name/api-version/resource-name
```

目前有两种可用的API名称，这将在下面进一步讨论的：

- auth - 用于认证相关的操作，并
- api - 为一切。

目前的API版本2。然而，也有一个符号的版本，称为latest，解决由给定的JIRA实例所支持的最新版本。举个例子，如果你想获取问题的JSON表示JRA-9从Atlassian的公开问题追踪您就可以访问：

```
https://jira.atlassian.com/rest/api/latest/issue/JRA-9
```

有一个WADL包含用于JIRA REST API中的每个资源的文档文件。它可[在这里](#)。

扩张

为了简化API响应，JIRA REST API使用资源扩张。这意味着API只会返回资源的部分明确要求时。

您可以使用expand查询参数指定您希望扩展实体的逗号分隔的列表，按名称识别它们。例如，附加?expand=names,renderedFields到一个问题的URI请求在响应中的翻译的字段名称和HTML呈现字段值的包容。上面继续我们的例子中，我们可以使用下面的URL来获取信息JRA-9：

```
https://jira.atlassian.com/rest/api/latest/issue/JRA-9?expand=names,renderedFields
```

要发现每个实体的标识，看expand在父对象的属性。在下面的JSON例如，资源声明部件为是可扩展的。

```
{
  "expand": "widgets",
  "self": "http://www.example.com/jira/rest/api/resource/KEY-1",
  "widgets": [
    {
      "widgets": [],
      "size": 5
    }
  ]
}
```

您可以使用点符号到另一个实体中指定的实体扩展。例如 ?expand=widgets.widgets将扩大Widgets集合，也是每个插件的fringe财产。

分页

JIRA使用分页限制响应大小返回一个潜在的大集合的项目资源。一个分页API的请求会导致值数组包裹在一些分页的元数据JSON对象，例如：

```
{
  "startAt": 0,
  "maxResults": 10,
  "total": 200,
  "values": [
    /* 结果0 */,
    /* 1的结果 */,
    /* 2的结果 */
  ]
}
```

客户端可以使用“startAt”和“maxResults”参数来检索结果所需要的数字。

该“maxResults”参数表示有多少结果每页返回。每个API可能对返回的项目数量不同的限制。

的“startAt”参数指示哪些项目应作为在结果中的网页的第一项。

重要的是：该响应包含“总”的字段，它表示包含在所有页的实体的总数。这个数字可能会改变为客户端请求的后续页。客户端应该总是假设所请求的页面可以为空。REST API的消费者也应该考虑外地是可选的。在情况下，计算这个值的时候太贵了，我们可以不响应包括此。

订购

由某一特定领域的一些资源支持排序。排序请求是在提供**ORDERBY**查询参数。请参阅该文档的具体使用方法，看看哪些字段他们支持，如果他们支持订购的。

排序都可以是升序或降序。默认情况下它的上升。要指定排序使用“-”或“+”号。例子：

- ? **ORDERBY** = 名称
由“名”升序
- ? **ORDERBY** = +名
由“名”升序
- ? **ORDERBY** = -name
由“名”降序排列

实验方法

方法标记为实验可能会更改，恕不较早的通知。我们正在寻找你对这些方法的反馈。

特殊的请求和响应头

- 的**X-AUSERNAME** - 其中包含验证用户的任一用户名或'匿名'响应头。
- 的**X-Atlassian**的令牌 - 这接受的multipart / form-data的将只与处理请求的方法'X-Atlassian'的令牌：不检查'头'。

错误响应

大多数资源将除了状态代码返回响应主体。通常情况下，返回的实体的JSON模式如下：

```
{
  "id": "https://docs.atlassian.com/jira/REST/schema/error-collection#",
  "title": "Error Collection",
  "type": "object",
  "properties": {
    "errorMessages": {
      "type": "array",
      "items": {
        "type": "string"
      }
    },
    "errors": {
      "type": "object",
      "patternProperties": {
        ".+": {
          "type": "string"
        }
      },
      "additionalProperties": false
    },
    "status": {
      "type": "integer"
    }
  },
  "additionalProperties": false
}
```

资源 [隐藏所有方法](#)

API / 2 / [隐藏所有方法](#)

提供对当前用户的权限信息。

▼ 获取权限

GET /rest/api/2//mypermissions

返回所有的权限在系统中是否当前登录的用户拥有它们。您可以选择提供特定的上下文来获取权限（`projectKey`或专案编号或`issueKey`或`issueId`）

如果没有上下文提供的项目，如果用户有任何项目许可相关的权限将返回`true`

如果提供了项目背景下，如果用户在指定的项目的权限项目相关的权限将返回`true`。为正在使用的问题的数据（例如当前代理人）确定的权限，真会如果用户满足许可条件中的任何问题在于项目返回

如果提供了问题的背景下，它会返回用户是否有该特定问题的每个权限

注：以上意味着问题级别权限（`EDIT_ISSUE`为例），`hasPermission`没有提供上下文时可能是真实的，或者提供一个项目上下文时，但可能是任何给定的（或全部）出具虚假。如果记者分别给予`EDIT_ISSUE`许可，这将发生（例如）。这是因为，任何用户可以是一个记者，除了在一个具体的问题，其中所述记者已知的情况下。

全局权限仍然会返回所有范围。

此前版本6.4服务返回的项目权限对应于`com.atlassian.jira.security.Permissions.Permission`常数项。由于这些6.4密钥被认为已过时，并对应于`com.atlassian.jira.permission.ProjectPermissions`定义的常量该服务回报系统项目的许可密钥。与传统钥匙的权限也还是返回向后兼容性，它们都标有一个属性`deprecatedKey = TRUE`。属性缺少与当前键项目的权限。

请求

查询参数

参数	类型	描述
<code>projectKey</code>	串	- 项目范围的关键返回权限。
<code>projectId</code>	串	- 项目范围ID返回权限。
<code>issueKey</code>	串	- 该问题的范围主要返回权限。
<code>issueId</code>	串	- 该问题范围ID返回权限。

回复

状态200 - 应用程序/JSON 返回JIRA以及用户是否拥有所有这些权限的列表。

例

```
{
  "permissions": [
    "EDIT_ISSUE": {
      "id": "12",
      "key": "EDIT_ISSUES",
      "name": "Edit Issues",
      "type": "PROJECT",
      "description": "Ability to edit issues.",
      "havePermission": true
    }
  ]
}
```

► 架构

状态400 返回如果项目或问题ID无效。

状态404 如果找不到该项目或问题ID或键返回。

▼ 获取所有权限

GET /rest/api/2//permissions

返回出现在JIRA实例中的所有权限 - 通过插件添加全球，项目和全局对象

回复

状态200 - 应用程序/JSON 返回JIRA所有权限的列表。

例

```
{
  "permissions": [
    {
      "key": "BULK_CHANGE",
      "name": "Bulk Change",
      "type": "GLOBAL",
      "description": "Ability to modify a collection of issues at once. For example, resolve multiple issues in one step."
    }
  ]
}
```

► 架构

状态401 返回未经授权的请求

状态403 返回用户无需管理权限

API / 2 /应用程序的属性 隐藏所有方法

▼ Get 属性

GET /rest/api/2/application-properties

返回一个应用程序属性。

请求

查询参数

参数	类型	描述
key	串	包含属性值的字符串
permissionLevel	串	当读取一个列表指定列表中的所有项目的权限级别见{@link com.atlassian.jira.bc.admin.ApplicationPropertiesService>EditPermissionLevel{}}
keyFilter	串	当获取列表允许通过按键如财产的启动被过滤列表中的“jira.if.”取对子级只可编辑，其键开始“jira.if.”这些权限。这是一个正则表达式。

回复

状态200 - 应用程序/JSON 如果属性存在返回，目前已验证的用户有权限查看。包含属性的完整的表示。

例

```
[
  {
    "id": "jira.home",
    "key": "jira.home",
    "value": "/var/jira/jira-home",
    "name": "jira.home",
    "desc": "JIRA home directory",
    "type": "string",
    "defaultValue": ""
  },
  {
    "id": "jira.clone.prefix",
    "key": "jira.clone.prefix",
    "value": "CLONE -",
    "name": "The prefix added to the Summary field of cloned issues",
    "desc": null,
    "type": "string",
    "defaultValue": "CLONE -"
  }
]
```

► 架构

状态404 如果属性不存在，或当前身份验证的用户没有权限查看它返回。

▼ 通过宁静的表设置属性

PUT /rest/api/2/application-properties/{id}

通过PUT修改应用程序属性。“价值”字段存在于PUT将覆盖现有的值。

请求

例

```
{
  "id": "jira.home",
  "value": "/var/jira/jira-home"
}
```

▶ 架构

回复

状态200 如果属性存在返回，目前已验证的用户有权对其进行编辑。

▶ 架构

状态403 返回如果当前身份验证的用户没有权限编辑属性。

状态404 如果属性不存在，或当前身份验证的用户没有权限查看它返回。

▼ 获得高级设置

GET /rest/api/2/application-properties/advanced-settings

返回时所显示的“常规配置>高级设置”页面上的属性。

回复

状态200 - 应用程序/JSON 返回所有的属性，在“常规配置>高级设置”页面中显示。

例

```
[
  {
    "id": "jira.home",
    "key": "jira.home",
    "value": "/var/jira/jira-home",
    "name": "jira.home",
    "desc": "JIRA home directory",
    "type": "string",
    "defaultValue": ""
  },
  {
    "id": "jira.clone.prefix",
    "key": "jira.clone.prefix",
    "value": "CLONE -",
    "name": "The prefix added to the Summary field of cloned issues",
    "desc": null,
    "type": "string",
    "defaultValue": "CLONE -"
  }
]
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态403 返回如果当前用户不是管理员。

API / 2 / applicationrole 隐藏所有方法

提供对JIRA的应用程序角色REST访问。

▼ 得到所有

GET /rest/api/2/applicationrole

返回系统中的所有ApplicationRoles。也将返回一个包含ApplicationRoles集合的一个版本散列ETag头。

回复

状态200 - 应用程序/JSON 返回所有ApplicationRoles系统

例

```
[ {
    "key": "jira-software",
    "groups": [
        "jira-software-users",
        "jira-testers"
    ],
    "name": "JIRA Software",
    "defaultGroups": [
        "jira-software-users"
    ],
    "selectedByDefault": false,
    "defined": false,
    "numberOfSeats": 10,
    "remainingSeats": 5,
    "userCount": 5,
    "userCountDescription": "5 developers",
    "hasUnlimitedSeats": false,
    "platform": false
},
{
    "key": "jira-core",
    "groups": [
        "jira-core-users"
    ],
    "name": "JIRA Core",
    "defaultGroups": [
        "jira-core-users"
    ],
    "selectedByDefault": false,
    "defined": false,
    "numberOfSeats": 1,
    "remainingSeats": 1,
    "userCount": 0,
    "userCountDescription": "0 users",
    "hasUnlimitedSeats": false,
    "platform": true
} ]
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态403 返回如果当前用户不是管理员。

▼ 把散

PUT /rest/api/2/applicationrole

更新与传递数据的ApplicationRoles如果版本散列是相同的服务器。只有组和默认组角色的设定可能会更新。请求更改密钥或角色的名称将被忽略。可以接受的是通过仅被更新为存在于服务器，但不是在数据与更新，也不会被删除的角色的角色。

请求

头参数

参数	类型	描述
If-Match	串	

例

```
{ "key": "jira-software",
  "groups": [
    "jira-software-users",
    "jira-testers"
  ],
  "name": "JIRA Software",
  "defaultGroups": [
    "jira-software-users"
  ],
  "selectedByDefault": true
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回更新ApplicationRole如果更新成功。

例

```
[  
  {  
    "key": "jira-software",  
    "groups": [  
      "jira-software-users",  
      "jira-testers"  
    ],  
    "name": "JIRA Software",  
    "defaultGroups": [  
      "jira-software-users"  
    ],  
    "selectedByDefault": false,  
    "defined": false,  
    "numberOfSeats": 10,  
    "remainingSeats": 5,  
    "userCount": 5,  
    "userCountDescription": "5 developers",  
    "hasUnlimitedSeats": false,  
    "platform": false  
  },  
  {  
    "key": "jira-core",  
    "groups": [  
      "jira-core-users"  
    ],  
    "name": "JIRA Core",  
    "defaultGroups": [  
      "jira-core-users"  
    ],  
    "selectedByDefault": false,  
    "defined": false,  
    "numberOfSeats": 1,  
    "remainingSeats": 1,  
    "userCount": 0,  
    "userCountDescription": "0 users",  
    "hasUnlimitedSeats": false,  
    "platform": true  
  }  
]
```

▶ 架构

状态401 返回如果当前用户没有权限进行编辑的角色。

状态412 如果**IF-Match**头不为空并且包含不同版本的服务器返回。

状态403 返回如果当前用户不是管理员。

状态404 如果角色不存在返回。

▼ 得到

GET /rest/api/2/applicationrole/{key}

返回ApplicationRole与如果存在通过关键。

回复

状态200 - 应用程序/ JSON 如果存在返回ApplicationRole。

例

```
{  
  "key": "jira-software",  
  "groups": [  
    "jira-software-users",  
    "jira-testers"  
  ],  
  "name": "JIRA Software",  
  "defaultGroups": [  
    "jira-software-users"  
  ],  
  "selectedByDefault": false,  
  "defined": false,  
  "numberOfSeats": 10,  
  "remainingSeats": 5,  
  "userCount": 5,  
  "userCountDescription": "5 developers",  
  "hasUnlimitedSeats": false,  
  "platform": false  
}
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态403 返回如果当前用户不是管理员。

状态404 如果角色不存在返回。

▼ 放

PUT /rest/api/2/applicationrole/{key}

更新ApplicationRole与传递的数据。只有组和默认组角色的设定可能会更新。请求更改密钥或角色的名称将被忽略。

可选: 如果versionHash通过**IF**-通过匹配的头请求将被拒绝, 如果不一样的服务器

请求

头参数

参数	类型	描述
If-Match	串	的版本的散列来更新。可选参数

例

```
{
  "key": "jira-software",
  "groups": [
    "jira-software-users",
    "jira-testers"
  ],
  "name": "JIRA Software",
  "defaultGroups": [
    "jira-software-users"
  ],
  "selectedByDefault": true
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回更新ApplicationRole如果更新成功。

例

```
{
  "key": "jira-software",
  "groups": [
    "jira-software-users",
    "jira-testers"
  ],
  "name": "JIRA Software",
  "defaultGroups": [
    "jira-software-users"
  ],
  "selectedByDefault": false,
  "defined": false,
  "numberOfSeats": 10,
  "remainingSeats": 5,
  "userCount": 5,
  "userCountDescription": "5 developers",
  "hasUnlimitedSeats": false,
  "platform": false
}
```

▶ 架构

状态401 返回如果当前用户没有权限进行编辑的角色。

状态412 如果IF-Match头不为空并且包含不同版本的服务器返回。

状态403 返回如果当前用户不是管理员。

状态404 如果角色不存在返回。

API / 2 / 附件 隐藏所有方法

▼ 获取附件

GET /rest/api/2/attachment/{id}

返回一个附件，包括实际附加的文件的URI的元数据。

回复

状态200 - 应用程序/JSON 附着的元数据的JSON表示。表示不包含附件本身，但包含的URI可用于下载实际的附加文件。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2.0/attachments/10000",
  "filename": "picture.jpg",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "created": "2016-09-06T09:48:12.753+0000",
  "size": 23123,
  "mimeType": "image/jpeg",
  "content": "http://www.example.com/jira/attachments/10000",
  "thumbnail": "http://www.example.com/jira/secure/thumbnaill/10000"
}
```

▶ 架构

状态403 主叫用户不允许查看所请求的附件。

状态404 任何:

有与所请求的id的附件
附件功能被禁用

▼ 删除附件

DELETE /rest/api/2/attachment/{id}

取下一个问题的附件。

回复

状态204 删除成功

状态403 主叫用户不允许移除请求的附件。

状态404 任何:

有与所请求的id的附件
附件功能被禁用

▼ 展开人体

试验

GET /rest/api/2/attachment/{id}/expand/human

努力扩大附件。输出为人类可读，并随时可能更改。

回复

状态200 - 应用程序/JSON 的附件展开的内容JSON表示。空的参赛名单意味着，附件无法进行扩展。这是不是空的，损坏或没有存档的。

例

```
{
  "id": 7237823,
  "name": "images.zip",
  "entries": [
    {
      "path": "MG00N067.JPG",
      "index": 0,
      "size": "119 kB",
      "mediaType": "image/jpeg",
      "label": "MG00N067.JPG"
    },
    {
      "path": "Allegro from Duet in C Major.mp3",
      "index": 1,
      "size": "1.36 MB",
      "mediaType": "audio/mpeg",
      "label": "Allegro from Duet in C Major.mp3"
    },
    {
      "path": "long/path/thanks/to/lots/of/subdirectories/inside/making/it/quite/hard/to/reach/the/leaf.txt",
      "index": 2,
      "size": "0.0 kB",
      "mediaType": "text/plain",
      "label": "long/path/thanks/to.../reach/the/leaf.txt"
    }
  ],
  "totalEntryCount": 39,
  "mediaType": "application/zip"
}
```

► 架构

状态403 主叫用户不允许查看所请求的附件。

状态404 任何:

有与所请求的id的附件
附件功能被禁用

状态409 不支持的压缩格式。

▼ 展开机

试验

GET /rest/api/2/attachment/{id}/expand/raw

努力扩大附件。输出为原应通过日久向后兼容。

回复

状态200 - 应用程序/JSON 的附件展开的内容JSON表示。空的参赛名单意味着，附件无法进行扩展。这是不是空的，损坏或没有存档的。

例

```
{
  "entries": [
    {
      "entryIndex": 0,
      "name": "Allegro from Duet in C Major.mp3",
      "size": 1430174,
      "mediaType": "audio/mpeg"
    },
    {
      "entryIndex": 1,
      "name": "1rm.rtf",
      "size": 331,
      "mediaType": "text/rtf"
    }
  ],
  "totalEntryCount": 24
}
```

▶ 架构

状态403 主叫用户不允许查看所请求的附件。

状态404 任何:

有与所请求的id的附件
附件功能被禁用

状态409 不支持的压缩格式。

▼ 获取附件元

GET /rest/api/2/attachment/meta

返回一个附件的元信息，特别是如果它们被启用，允许的最大上传大小。

回复

状态200 - 应用程序/JSON 的附着能力JSON表示。这种资源的消费者可能还需要检查，如果登录的用户有权上传或以其它方式使用{@link com.atlassian.jira.rest.v2.permission.PermissionsResource}操纵附件。

例

```
{
  "enabled": true,
  "uploadLimit": 1000000
}
```

▶ 架构

API / 2 / 审计 [隐藏所有方法](#)

资源占审计记录

▼ 获取记录

GET /rest/api/2/auditing/record

返回使用提供的参数过滤审计记录

请求

查询参数

参数	类型	描述
offset	INT	- 从搜索开始记录的数
limit	INT	- 返回结果的最大数量（如果限制为<= 0或> 1000，它将被设置做默认值： 1000）
filter	串	- 文本查询；将返回的每个记录都必须包含在其字段之一所提供的文本
from	串	- 时间戳过去；“从”必须小于或等于‘到’，否则结果集将为空只记载，凡在同一时刻或“从”时间戳后创建的响应提供
to	串	- 时间戳过去；“从”必须小于或等于‘到’，否则结果集将为空只记载，凡在同一时刻或创建早于“到”时间戳响应提供

回复

状态200 - 应用程序/JSON 返回与请求的查询参数过滤列表审计记录

例

```
{
  "id": 1,
  "summary": "User created",
  "remoteAddress": "192.168.1.1",
  "authorKey": "administrator",
  "created": "2014-03-19T18:45:42.967+0000",
  "category": "user management",
  "eventSource": "JIRA Connect Plugin",
  "description": "Optional description",
  "objectItem": {
    "id": "user",
    "name": "user",
    "typeName": "USER",
    "parentId": "/",
    "parentName": "JIRA Internal Directory"
  },
  "changedValues": [
    {
      "fieldName": "email",
      "changedFrom": "user@atlassian.com",
      "changedTo": "newuser@atlassian.com"
    }
  ],
  "associatedItems": [
    {
      "id": "jira-software-users",
      "name": "jira-software-users",
      "typeName": "GROUP",
      "parentId": "1",
      "parentName": "JIRA Internal Directory"
    }
  ]
}
```

▶ 架构

状态400 在未处理的错误时，同时取审计记录**状态403** 如果用户没有管理权限返回

▼ 添加记录

POST /rest/api/2/auditing/record

存放在审计日志记录

请求

例

```
{
  "summary": "User created",
  "created": null,
  "category": "USER_MANAGEMENT",
  "objectItem": {
    "id": "user",
    "name": "user",
    "typeName": "USER",
    "parentId": "/",
    "parentName": "JIRA Internal Directory"
  },
  "changedValues": [
    {
      "fieldName": "email",
      "changedFrom": "user@atlassian.com",
      "changedTo": "newuser@atlassian.com"
    }
  ],
  "associatedItems": [
    {
      "id": "jira-software-users",
      "name": "jira-software-users",
      "typeName": "GROUP",
      "parentId": "1",
      "parentName": "JIRA Internal Directory"
    }
  ]
}
```

▶ 架构

回复

状态201 返回如果记录成功存储。**状态400** 在未处理的错误时，同时取审计记录**状态403** 如果用户没有管理权限返回API / 2 / 头像 [隐藏所有方法](#)

▼ 获取所有系统的化身

GET /rest/api/2/avatar/{type}/system

返回给定类型的所有系统的化身。

回复

状态200 - 应用程序/JSON 返回包含系统替身列表的图。的地图返回到与项目/KEY/替身REST结束点的形状是一致的。

例

```
{
  "system": [
    {
      "id": "1000",
      "owner": "fred",
      "isSystemAvatar": true,
      "isSelected": false,
      "isDeletable": false,
      "urls": {
        "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
        "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
        "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
        "48x48": "http://localhost:8090/jira/secure/useravatar?size=large&avatarId=10040"
      },
      "selected": false
    }
  ]
}
```

▶ 架构

状态500 如果返回在检索化身的列表时发生错误。

▼ 存储临时头像

POST /rest/api/2/avatar/{type}/temporary

创建临时头像

请求

查询参数

参数	类型	描述
filename	串	文件的名称被上传
size	长	文件大小

回复

状态201 - 应用程序/JSON 临时头像裁剪说明

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态400 Validation失败。例如文件大小超出最大附件大小。

状态500 返回如果在临时化身转换为真正的化身出现错误

状态403 如果该请求不contain有效XSRF令牌返回

▼ 创建临时的头像

POST /rest/api/2/avatar/{type}/temporaryCrop

更新临时化身的种植指导。

请求

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "needsCropping": false
}
```

▶ 架构

回复

状态201 - 应用程序/JSON

状态400 返回如果裁剪坐标无效。**状态500** 如果返回裁剪过程中临时化身出现错误。

API / 2 / 评论/ { commentId } /属性 隐藏所有方法

▼ 获取属性键 实验

GET /rest/api/2/comment/{commentId}/properties

返回由键或用id标识注释的所有属性的钥匙。

回复

状态200 - 应用程序/ JSON 如果注释被发现返回。

例

```
{
  "keys": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
      "key": "issue.support"
    }
  ]
}
```

▶ 架构

状态400 返回如果注释键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限浏览评论。**状态404** 如果返回与给定的密钥或ID的评论不存在，或者给定键的属性是找不到的。

▼ 删除特性 实验

DELETE /rest/api/2/comment/{commentId}/properties/{propertyKey}

移除的键或用id标识注释的属性。THS用户删除属性是需要有权限管理评论。

回复

状态400 返回如果注释键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态204** 返回是否成功删除注释属性。**状态403** 如果返回调用用户没有权限编辑评论。**状态404** 如果返回与给定的密钥或ID的评论不存在，或者给定键的属性是找不到的。

▼ 设置属性 实验

PUT /rest/api/2/comment/{commentId}/properties/{propertyKey}

设置指定注释的属性的值。

你可以使用这个资源来存储反对键或用id标识的注释自定义数据。谁存储数据的用户需要具有的权限来管理评论。

回复

状态200 返回如果评论属性成功更新。**状态201** 如果成功创建注释属性返回。**状态400** 返回如果注释键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限管理评论。**状态404** 如果与给定的密钥或ID的评论不存在返回。

▼ Get属性 实验

GET /rest/api/2/comment/{commentId}/properties/{propertyKey}

返回由键或由ID标识的评论给定键的属性的值。谁检索的属性的用户需要有权限读取的注释。

回复

状态200 - 应用程序/JSON 如果注释财产被发现返回。

例

```
{
  "key": "issue.support",
  "value": [
    "hipchat.room.id": "support=123",
    "support.time": "1m"
  ]
}
```

▶ 架构

状态400 返回如果注释键或ID无效。

状态401 返回如果调用用户没有通过验证。

状态403 如果返回调用用户没有权限浏览评论。

状态404 如果返回与给定的密钥或ID的评论不存在，或者给定键的属性是找不到的。

API / 2 / 组件 隐藏所有方法

▼ 创建组件

POST /rest/api/2/component

通过创建一个POST组件。

请求

例

```
{
  "name": "Component 1",
  "description": "This is a JIRA component",
  "leadUserName": "fred",
  "assigneeType": "PROJECT_LEAD",
  "isAssigneeTypeValid": false,
  "project": "PROJECTKEY",
  "projectId": 10000
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果组件创建成功返回。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/component/10000",
  "id": "10000",
  "name": "Component 1",
  "description": "This is a JIRA component",
  "lead": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "assigneeType": "PROJECT_LEAD",
  "assignee": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "realAssigneeType": "PROJECT_LEAD",
  "realAssignee": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  }
}
```

▶ 架构

- 状态401** 如果调用者没有登录，并没有权限在项目中创建组件返回。
- 状态403** 返回如果调用者进行身份验证并没有权限在项目中创建组件。
- 状态404** 如果该项目不存在，或当前身份验证的用户没有权限查看它返回。

▼ 取得组件

GET /rest/api/2/component/{id}

返回一个项目的组成部分。

回复

- 状态200** - 应用程序/JSON 返回一个项目的组成一个完整的JSON表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/component/10000",
  "id": "10000",
  "name": "Component 1",
  "description": "This is a JIRA component",
  "lead": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "assigneeType": "PROJECT_LEAD",
  "assignee": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "realAssigneeType": "PROJECT_LEAD",
  "realAssignee": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  }
}
```

▶ 架构

- 状态404** 返回如果存在与给定键的部件，或者如果调用用户不具有权限浏览的组件。

▼ 更新组件

PUT /rest/api/2/component/{id}

通过PUT修改组件。目前在把任何领域将覆盖现有值。为方便起见，如果一个字段不存在，它被忽略。

如果leadUserName是一个空字符串（“”）的元件引线将被删除。

请求

例

```
{
  "name": "Component 1",
  "description": "This is a JIRA component",
  "leadUserName": "fred",
  "assigneeType": "PROJECT_LEAD",
  "isAssigneeTypeValid": false,
  "project": "PROJECTKEY",
  "projectId": 10000
}
```

▶ 架构

回复

- 状态200** 如果该组件中存在返回，目前已验证的用户有权对其进行编辑。

▶ 架构

状态403 返回如果当前身份验证的用户没有权限编辑组件。

状态404 如果组件不存在，或当前身份验证的用户没有权限查看它返回。

▼ 删除

DELETE /rest/api/2/component/{id}

删除项目的组成部分。

请求

查询参数

参数	类型	描述
moveIssuesTo	串	新组件适用于它的“ID”部分将被删除的问题。如果该值为null，则‘ID’组件仅仅是从相关的所有主题中删除。

回复

状态204 返回组件是否成功删除。

状态403 返回如果当前身份验证的用户没有权限删除的组件。

状态404 如果组件不存在，或当前身份验证的用户没有权限查看它返回。

▼ 取得组件相关的问题

GET /rest/api/2/component/{id}/relatedIssueCounts

返回与此相关的组件问题计数。

回复

状态200 - 应用程序/JSON 如果该组件中存在返回，目前已验证的用户有权限查看。包含了与此相关的组件问题计数。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/component/10000",
  "issueCount": 23
}
```

► 架构

状态404 如果组件不存在，或当前身份验证的用户没有权限查看它返回。

API / 2 / 配置 隐藏所有方法

▼ 获取配置

GET /rest/api/2/configuration

返回如果JIRA的可选功能启用或禁用的信息。如果启用了实时跟踪，它也返回有关时间跟踪配置的详细信息。

回复

状态200 - 应用程序/JSON 返回的JIRA可选功能的配置。

例

```
{
  "votingEnabled": true,
  "watchingEnabled": true,
  "unassignedIssuesAllowed": false,
  "subTasksEnabled": false,
  "issueLinkingEnabled": true,
  "timeTrackingEnabled": true,
  "attachmentsEnabled": true,
  "timeTrackingConfiguration": {
    "workingHoursPerDay": 8,
    "workingDaysPerWeek": 5,
    "timeFormat": "pretty",
    "defaultUnit": "day"
  }
}
```

► 架构

状态401 返回如果用户没有登录。

API / 2 / 的CustomFieldOption 隐藏所有方法

▼ 获取自定义字段选项

GET /rest/api/2/customFieldOption/{id}

返回具有给定id的自定义字段选项的完整表示。

回复

状态200 - 应用程序/JSON 如果自定义字段选项存在且由主叫用户可见返回。

例

```
{
  "self": "http://localhost:8090/jira/rest/api/2.0/customFieldOption/3",
  "value": "Blue"
}
```

▶ 架构

状态404 如果返回的自定义字段选项不存在，或者不是给主叫用户可见。

API / 2 /仪表盘 隐藏所有方法

的/dashboard资源。

▼ 名单

GET /rest/api/2/dashboard

返回所有仪表盘的列表，随意过滤它们。

请求

查询参数

参数	类型	描述
filter	串	施加到仪表板的列表中的一个可选的过滤器。有效值包括“favourite”用于返回唯一喜欢的仪表板，以及“my”用于返回由主叫用户所拥有的仪表板。
startAt	INT	第一个仪表板的指数回归（从0开始）。必须是0或多个的 maxResults
maxResults	INT	一个提示，仪表盘的最大数量在每个调用返回。需要注意的是JIRA服务器保留征收权maxResults的限制比一个客户提供的价值，会费缺乏或资源或任何其他条件低。发生这种情况时，你的结果将被截断。来电者应经常检查返回maxResults要确定有效使用的价值。

回复

状态200 返回仪表板的列表。

例

```
{
  "startAt": 10,
  "maxResults": 10,
  "total": 143,
  "prev": "http://www.example.com/jira/rest/api/2/dashboard?startAt=0",
  "next": "http://www.example.com/jira/rest/api/2/dashboard?startAt=10",
  "dashboards": [
    {
      "id": "10000",
      "name": "System Dashboard",
      "self": "http://www.example.com/jira/rest/api/2/dashboard/10000",
      "view": "http://www.example.com/jira/secure/Dashboard.jspa?selectPageId=10000"
    },
    {
      "id": "20000",
      "name": "Build Engineering",
      "self": "http://www.example.com/jira/rest/api/2/dashboard/20000",
      "view": "http://www.example.com/jira/secure/Dashboard.jspa?selectPageId=20000"
    }
  ]
}
```

▶ 架构

▼ 获取仪表板

GET /rest/api/2/dashboard/{id}

返回一个单一的仪表板。

回复

状态200 返回一个单一的仪表板。

例

```
{
  "id": "10000",
  "name": "System Dashboard",
  "self": "http://www.example.com/jira/rest/api/2/dashboard/10000",
  "view": "http://www.example.com/jira/secure/Dashboard.jspa?selectPageId=10000"
}
```

▶ 架构

状态404 返回如果没有与指定的**id**的仪表盘，或者如果用户没有权限看到它。

API / 2 / 仪表/ {} dashboardId /项目/ {}的itemId /属性 隐藏所有方法

▼ 获取属性键

GET /rest/api/2/dashboard/{dashboardId}/items/{itemId}/properties

返回由**id**标识的仪表板项目的所有属性的钥匙。

回复

状态200 - 应用程序/ JSON 如果仪表板项目被发现返回。

例

```
{
  "keys": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
      "key": "issue.support"
    }
  ]
}
```

▶ 架构

状态400 返回如果仪表板的项目**ID**无效。

状态404 如果与给定**id**仪表板项目不存在，或者用户没有权限来查看它返回。

▼ 删除属性

DELETE /rest/api/2/dashboard/{dashboardId}/items/{itemId}/properties/{propertyKey}

移除按键式或由**id**标识的仪表板项目的属性。THS用户删除属性是需要有权限管理仪表板项目。

回复

状态400 返回如果仪表板的项目**ID**无效。

状态204 如果返回成功删除仪表板项目属性。

状态403 返回如果调用用户没有权限编辑仪表板项目。

状态404 如果与给定**id**仪表板项目不存在，或者用户没有权限来查看它返回。

▼ 设置属性

PUT /rest/api/2/dashboard/{dashboardId}/items/{itemId}/properties/{propertyKey}

设置指定的仪表板项目的属性值。

你可以使用这个资源来存储对由**ID**标识的仪表板项目自定义数据。谁存储数据的用户需要具有的权限管理信息中心项。

回复

状态200 返回如果仪表板项目属性成功更新。

状态201 如果成功创建仪表板项目属性返回。

状态400 返回如果仪表板的项目**ID**无效。

状态403 如果返回调用用户没有权限管理仪表板项目。

状态404 如果与给定**id**仪表板项目不存在，或者用户没有权限来查看它返回。

▼ Get 属性

GET /rest/api/2/dashboard/{dashboardId}/items/{itemId}/properties/{propertyKey}

返回与由id标识的仪表板项目给定键的属性的值。谁检索的属性的用户需要有权限读取仪表板项目。

回复

状态200 - 应用程序/JSON 如果仪表板项目属性被发现返回。

例

```
{
  "key": "issue.support",
  "value": [
    "hipchat.room.id": "support-123",
    "support.time": "1m"
  ]
}
```

▶ 架构

状态400 返回如果仪表板的项目ID无效。

状态404 如果与给定id仪表板项目不存在，或者用户没有权限来查看它返回。

API / 2 / 现场 [隐藏的所有方法](#)**▼ 创建自定义字段**

POST /rest/api/2/field

创建使用定义自定义字段（对象封装的自定义字段数据）

请求

例

```
{
  "name": "New custom field",
  "description": "Custom field for picking groups",
  "type": "com.atlassian.jira.plugin.system.customfieldtypes:group picker",
  "searcherKey": "com.atlassian.jira.plugin.system.customfieldtypes:group picker searcher"
}
```

▶ 架构

回复

状态201 如果自定义字段创建返回。

例

```
{
  "id": "customfield_10101",
  "name": "New custom field",
  "custom": true,
  "orderable": true,
  "navigable": true,
  "searchable": true,
  "clauseNames": [
    "cf[10101]",
    "New custom field"
  ],
  "schema": {
    "type": "project",
    "custom": "com.atlassian.jira.plugin.system.customfieldtypes:project",
    "customId": 10101
  }
}
```

▶ 架构

状态400 返回如果输入是无效的（例如，无效值）。

状态500 如果返回除了自定义字段创建过程中发生。

▼ 获取领域

GET /rest/api/2/field

返回所有字段的列表，系统和自定义

回复

状态200 - 应用程序/JSON 包含JSON所有可见字段的完整表示。

例

```
[ {
    "id": "description",
    "name": "Description",
    "custom": false,
    "orderable": true,
    "navigable": true,
    "searchable": true,
    "clauseNames": [
        "description"
    ],
    "schema": {
        "type": "string",
        "system": "description"
    }
},
{
    "id": "summary",
    "name": "Summary",
    "custom": false,
    "orderable": true,
    "navigable": true,
    "searchable": true,
    "clauseNames": [
        "summary"
    ],
    "schema": {
        "type": "string",
        "system": "summary"
    }
}
]
```

▶ 架构

API / 2 / 过滤器 隐藏所有方法

资源，搜索。

▼ 创建过滤器

POST /rest/api/2/filter

创建一个新的过滤器，并返回新创建的过滤器。目前台只使用默认的用户共享权限的权限

请求

查询参数

参数	类型	描述
expand	串	参数扩大

例

```
{
    "name": "All Open Bugs",
    "description": "Lists all open bugs",
    "jql": "type = Bug and resolution is empty",
    "favorite": true
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回过滤器的JSON表示

例
<pre>{ "self": "http://www.example.com/jira/rest/api/2/filter/10000", "id": "10000", "name": "All Open Bugs", "description": "Lists all open bugs", "owner": { "self": "http://www.example.com/jira/rest/api/2/user?username=fred", "name": "fred", "avatarUrls": { "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred", "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred", "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred", "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred" }, "displayName": "Fred F. User", "active": false }, "jql": "type = Bug and resolution is empty", "viewUrl": "http://www.example.com/jira/issues/?filter=10000", "searchUrl": "http://www.example.com/jira/rest/api/2/search?jql=type%20=%20Bug%20and%20resolution%20is%20empty", "favorite": true, "sharePermissions": [], "subscriptions": { "size": 0, "items": [], "max-results": 1000, "start-index": 0, "end-index": 0 } }</pre>

▶ 架构

状态400 如果返回输入无效（未提供如过滤器名称）。

▼ 获取过滤器

GET /rest/api/2/filter/{id}

返回给定的过滤器的ID

请求

查询参数

参数	类型	描述
expand	串	参数扩大

回复

状态200 - 应用程序/JSON 返回过滤器的JSON表示

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/filter/10000",
  "id": "10000",
  "name": "All Open Bugs",
  "description": "Lists all open bugs",
  "owner": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "jql": "type = Bug and resolution is empty",
  "viewUrl": "http://www.example.com/jira/issues/?filter=10000",
  "searchUrl": "http://www.example.com/jira/rest/api/2/search?jql=type%20%3D%20Bug%20and%20resolution%20is%20empty",
  "favourite": true,
  "sharePermissions": [],
  "subscriptions": {
    "size": 0,
    "items": [],
    "max-results": 1000,
    "start-index": 0,
    "end-index": 0
  }
}
```

▶ 架构

状态400 返回如果有仰视的过滤器的问题给出的ID

▼ 编辑过滤器

PUT /rest/api/2/filter/{id}

更新现有的过滤器，并返回其新的值。

请求

查询参数

参数	类型	描述
expand	串	参数扩大

例

```
{
  "name": "All Open Bugs",
  "description": "Lists all open bugs",
  "jql": "type = Bug and resolution is empty",
  "favourite": true
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回过滤器的JSON表示

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/filter/10000",
  "id": "10000",
  "name": "All Open Bugs",
  "description": "Lists all open bugs",
  "owner": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "jql": "type = Bug and resolution is empty",
  "viewUrl": "http://www.example.com/jira/issues/?filter=10000",
  "searchUrl": "http://www.example.com/jira/rest/api/2/search?jql=type%20%3D%20Bug%20and%20resolution%20is%20empty",
  "favorite": true,
  "sharePermissions": [],
  "subscriptions": {
    "size": 0,
    "items": [],
    "max-results": 1000,
    "start-index": 0,
    "end-index": 0
  }
}
```

▶ 架构

状态400 返回如果有更新了给定id的过滤器的问题**▼ 删除过滤器**

DELETE /rest/api/2/filter/{id}

删除过滤器。

回复**状态400** 如果发生错误，返回。**状态401** 返回如果调用用户没有通过验证。**状态204** 返回如果过滤器被成功取出。**▼ 默认列**

GET /rest/api/2/filter/{id}/columns

返回给定过滤器的默认列。当前登录的用户将被用作制造这种请求的用户。

回复**状态200** - 应用程序/JSON 返回列的列表进行配置给定用户

▶ 架构

状态500 返回如果检索列配置发生了错误。**状态404** 如果过滤器没有任何列返回。**▼ 集列**

PUT /rest/api/2/filter/{id}/columns

设置为给定的过滤器默认列。

请求**回复****状态200** 当列保存成功返回**状态500** 返回如果检索列配置发生了错误。**▼ 重置列**

DELETE /rest/api/2/filter/{id}/columns

重置列给定的过滤器，使得过滤器不再有其自己的列配置。

回复

状态500 返回如果检索列配置发生了错误。

状态204 当返回的列被重置/删除成功

▼ 获取共享权限

GET /rest/api/2/filter/{id}/permission

返回给定过滤器的所有共享权限。

回复

状态200 - 应用程序/JSON 如果返回成功。

例

```
[  
  {  
    "id": 10000,  
    "type": "global"  
  },  
  {  
    "id": 10010,  
    "type": "project",  
    "project": {  
      "self": "http://www.example.com/jira/rest/api/2/project/EX",  
      "id": "10000",  
      "key": "EX",  
      "name": "Example",  
      "avatarUrls": {  
        "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10000",  
        "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10000",  
        "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10000",  
        "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10000"  
      },  
      "projectCategory": {  
        "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",  
        "id": "10000",  
        "name": "FIRST",  
        "description": "First Project Category"  
      }  
    }  
  },  
  {  
    "id": 10010,  
    "type": "project",  
    "project": {  
      "self": "http://www.example.com/jira/rest/api/2/project/MKY",  
      "id": "10002",  
      "key": "MKY",  
      "name": "Example",  
      "avatarUrls": {  
        "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10002",  
        "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10002",  
        "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10002",  
        "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10002"  
      },  
      "projectCategory": {  
        "self": "http://www.example.com/jira/rest/api/2/projectCategory/10002",  
        "id": "10002",  
        "name": "SECOND",  
        "description": "Second Project Category"  
      }  
    }  
  }  
]
```

▶ 架构

状态401 如果返回的用户没有登录。

状态404 当给定的ID过滤器不存在，或者当用户没有权限查看该过滤器返回。。

▼ 添加共享权限

POST /rest/api/2/filter/{id}/permission

增加了共享权限给定的过滤器。添加一个全局权限将删除过滤器的所有以前的权限。

请求

例

```
{  
  "type": "group",  
  "groupname": "jira-administrators"  
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果返回成功。

例

```
[  
  {  
    "id": 10000,  
    "type": "global"  
  },  
  {  
    "id": 10010,  
    "type": "project",  
    "project": {  
      "self": "http://www.example.com/jira/rest/api/2/project/EX",  
      "id": "10000",  
      "key": "EX",  
      "name": "Example",  
      "avatarUrls": {  
        "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10000",  
        "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10000",  
        "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10000",  
        "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10000"  
      },  
      "projectCategory": {  
        "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",  
        "id": "10000",  
        "name": "FIRST",  
        "description": "First Project Category"  
      }  
    }  
  },  
  {  
    "id": 10010,  
    "type": "project",  
    "project": {  
      "self": "http://www.example.com/jira/rest/api/2/project/MKY",  
      "id": "10002",  
      "key": "MKY",  
      "name": "Example",  
      "avatarUrls": {  
        "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10002",  
        "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10002",  
        "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10002",  
        "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10002"  
      },  
      "projectCategory": {  
        ...  
      }  
    }  
  }  
]
```

▶ 架构

状态400 回到获准输入bean是无效或当用户没有共享的过滤器或当用户无法编辑定的过滤器的权限。**状态401** 如果返回的用户没有登录。**状态404** 当给定的ID过滤器不存在，或者当用户没有权限查看该过滤器返回。。

▼ 获取共享权限

GET /rest/api/2/filter/{id}/permission/{permissionId}

返回给定过滤器的单一共享权限。

回复

状态200 - 应用程序/JSON 如果返回成功。

例

```
{  
  "id": 10000,  
  "type": "global"  
}
```

▶ 架构

状态401 如果返回的用户没有登录。**状态404** 当过滤器或许可给定id不存在，或者当用户没有权限查看该过滤器返回。

▼ 删除共享权限

DELETE /rest/api/2/filter/{id}/permission/{permissionId}

移除给定的过滤器的共享权限。

回复

状态200 - 应用程序/JSON**状态204** 如果返回成功。**状态404** 当过滤器或许可给定id不存在，或者当用户没有权限查看该过滤器返回。

▼ 获取默认的共享范围

GET /rest/api/2/filter/defaultShareScope

返回登录用户的默认共享范围。

回复

状态200 - 应用程序/JSON 返回登录用户的默认共享范围。

例

```
{  
    "scope": "GLOBAL"  
}
```

▶ 架构

状态400 返回, 如果有仰视用于登录的用户偏好中的问题

▼ 设置默认的共享范围

PUT /rest/api/2/filter/defaultShareScope

设置登录用户的默认共享范围。可用的值有全局和专用。

请求

例

```
{  
    "scope": "GLOBAL"  
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回登录用户的新的默认共享范围。

例

```
{  
    "scope": "GLOBAL"  
}
```

▶ 架构

状态400 返回, 如果有一个问题设定登录的用户的偏好

▼ 获取最喜爱的过滤器

GET /rest/api/2/filter/favourite

返回登录用户的喜爱的过滤器。

请求

查询参数

参数	类型	描述
expand	串	参数扩大

回复

状态200 - 应用程序/JSON 返回的过滤器列表的JSON表示

例

```
[ {
    "self": "http://www.example.com/jira/rest/api/2/filter/10000",
    "id": "10000",
    "name": "All Open Bugs",
    "description": "Lists all open bugs",
    "owner": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    },
    "jql": "type = Bug and resolution is empty",
    "viewUrl": "http://www.example.com/jira/issues/?filter=10000",
    "searchUrl": "http://www.example.com/jira/rest/api/2/search?jql=type%20%3D%20Bug%20and%20resolution%20is%20empty",
    "favourite": true,
    "sharePermissions": [],
    "subscriptions": {
        "size": 0,
        "items": [],
        "max-results": 1000,
        "start-index": 0,
        "end-index": 0
    }
},
{
    "self": "http://www.example.com/jira/rest/api/2/filter/10010",
    "id": "10010",
    "name": "My issues",
    "description": "Issues assigned to me",
    "owner": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    }
}]
```

▶ 架构

API / 2 / 组 隐藏所有方法

▼ 创建组

POST /rest/api/2/group

通过给定的组参数创建一个组

返回REST表示对请求的组。

请求

▶ 架构

回复

状态201 - 应用程序/JSON 返回JSON格式的JIRA组充分的代表性。

例

```
{ "name": "jira-administrators",
  "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-administrators",
  "users": {
    "size": 1,
    "items": [
      {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "displayName": "Fred F. User",
        "active": false
      }
    ],
    "max-results": 50,
    "start-index": 0,
    "end-index": 0
  },
  "expand": "users"
}
```

▶ 架构

状态400 如果返回用户请求一个空的组名或组已存在

状态401 返回如果当前用户没有通过验证。

状态403 如果当前用户没有管理员权限返回。

▼ 获取组 [弃用]

GET /rest/api/2/group

返回REST表示对请求的组。如果允许“用户”的扩大提供选项来获得属于指定组的活跃用户及其子组的列表。您可以通过用户列表页面使用指标的参数

扩展。例如，从指数10获得用户15指数采用“用户[10:15]”扩大值。这将返回6用户（如果有至少16个用户，这组）。索引是从0开始的，包容的。
此资源已过时，请使用组/成员API来代替。

请求

查询参数

参数	类型	描述
groupname	串	要求组的名称。
expand	串	字段列表扩大。目前仅可扩大为“用户”。

回复

状态 - 应用程序/JSON

▼ 删除组

DELETE /rest/api/2/group

删除一组由下式给出组参数。

返回任何内容

请求

查询参数

参数	类型	描述
groupname	串	(强制) 组的名称删除。
swapGroup	串	如果您删除组和内容仅限于该组，内容将从所有用户隐藏。为了防止这种情况，使用此参数指定不同组的限制（仅限意见和worklogs）转让。

回复

状态200 - 应用程序/JSON 如果该组被删除返回。

状态400 如果用户请求一个不存在的组返回。

状态401 返回如果当前用户没有通过验证。

状态403 如果当前用户没有管理员权限返回。

状态404 如果没有找到请求的组返回。

▼ 获得用户群

GET /rest/api/2/group/member

该资源返回分页谁是指定的组及其子组成员的用户列表。在页面的用户通过用户名排序。该资源的用户需要有系统管理员或管理员权限。

请求

查询参数

参数	类型	描述
groupname	串	对于哪些成员将返回该组的名称。
includeInactiveUsers	布尔	不活动的用户将被纳入响应，如果设置为true。 默认值: 假
startAt	长	第一用户的组的索引返回（0开始）。 默认值: 0
maxResults	INT	用户的最大数目，返回（最多50个）。 默认值: 50

回复

状态200 - 应用程序/JSON 该组中的用户的分页列表。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/group/member?groupname=jira-administrators&includeInactiveUsers=false&startAt=2&maxResults=2",
  "nextPage": "http://www.example.com/jira/rest/api/2/group/member?groupname=jira-administrators&includeInactiveUsers=false&startAt=4&maxResults=2",
  "maxResults": 2,
  "startAt": 3,
  "total": 5,
  "isLast": false,
  "values": [
    {
      "self": "http://example/jira/rest/api/2/user?username=fred",
      "name": "Fred",
      "key": "fred",
      "emailAddress": "fred@atlassian.com",
      "avatarUrls": {},
      "displayName": "Fred",
      "active": true,
      "timeZone": "Australia/Sydney"
    },
    {
      "self": "http://example/jira/rest/api/2/user?username=barney",
      "name": "Barney",
      "key": "barney",
      "emailAddress": "barney@atlassian.com",
      "avatarUrls": {},
      "displayName": "Barney",
      "active": false,
      "timeZone": "Australia/Sydney"
    }
  ]
}
```

▶ 架构

状态400 返回, 如果提供的团体的名称为空

状态401 返回如果用户没有登录。

状态403 返回如果调用用户不是管理员或系统管理员

状态404 如果指定的组不存在返回

▼ 将用户添加到组

POST /rest/api/2/group/user

添加给定的用户到一个组。

返回该组的当前状态。

请求

查询参数

参数	类型	描述
groupname	串	要求组的名称。

例

```
{
  "name": "charlie"
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回JSON格式的JIRA组充分的代表性。

▶ 架构

状态400 或者返回如果用户请求一个空的组名用户已经属于该组。

状态401 返回如果当前用户没有通过验证。

状态403 如果当前用户没有管理员权限返回。

状态404 如果请求的组未找到或要求用户找不到返回。

▼ 从组中删除用户

DELETE /rest/api/2/group/user

移除一组给定的用户。

返回任何内容

请求

查询参数

参数	类型	描述
groupname	串	要求组的名称。
username	串	用户从一组中删除

回复

状态200 - 应用程序/ JSON 如果用户从该组中删除。

状态400 如果返回用户请求一个空的组名

状态401 返回如果当前用户没有通过验证。

状态403 如果当前用户没有管理员权限返回。

状态404 或者返回如果请求的组未发现被请求的用户或未找到

API / 2 / 组 隐藏所有方法

REST端点在一组选取器搜索组

▼ 查找组

GET /rest/api/2/groups/picker

返回匹配给定查询子组。这主要是为与该组选择器的使用，所以返回的基团含有的html用作选择器的建议。该集团也裹着也包含在选择器中使用的开头一个响应对象，特别是显示y匹配组第X。

基团的数目返回由系统属性“jira.ajax.autocomplete.limit”不限

该团体将是唯一的和分类。

请求

查询参数

参数	类型	描述
query	串	一个字符串匹配组against
exclude	串	
maxResults	INT	
userName	串	

回复

状态200 - 应用程序/ JSON 返回即使没有组给定的子字符串匹配

例

```
{
  "headers": "Showing 20 of 25 matching groups",
  "total": 25,
  "groups": [
    {
      "name": "jdog-developers",
      "html": "<b>j</b>dog-developers"
    },
    {
      "name": "juvenal-bot",
      "html": "<b>j</b>juvenal-bot"
    }
  ]
}
```

▶ 架构

API / 2 / groupuserpicker 隐藏所有方法

▼ 查找用户和组

GET /rest/api/2/groupuserpicker

返回用户和与突出匹配查询组的列表。此资源不能被匿名访问。

请求

查询参数

参数	类型	描述
query	串	使用的字符串搜索用户名, 姓名或电子邮件地址
maxResults	INT	用户的最大数量返回（默认为50）。最大允许值为1000。如果指定的值比这个数字更高，搜索结果将被截断。
showAvatar	布尔	
fieldId	串	自定义字段标识, 如果这个请求来自一个自定义字段, 如用户选择器。可选的。
projectId	串	项目ID列表, 以进一步限制搜索此参数可以多次出现在多个项目IDS通过。不支持逗号分隔值。当fieldId存在此参数仅使用。
issueTypeId	串	问题类型的ID列表, 以进一步限制搜索。此参数可以多次出现在多个问题类型IDS通过。不支持逗号分隔值。特殊值如-1（所有标准问题类型）-2（所有子任务问题类型）的支持。当fieldId存在此参数仅使用。

回复

状态200

▶ 架构

API / 2 / 问题 隐藏所有方法

▼ 创建问题

POST /rest/api/2/issue

创建一个问题或JSON表示一个子任务。

可以在创建进行设置，在任何领域的参数或更新参数字段可以用确定/**REST / API / 2 / 问题 / createmeta**资源。如果字段没有被配置为显示在创建屏幕上，那么它将不会在createmeta，并且如果它被提交会发生场验证错误。

创建一个子任务类似于创建一个普通的问题，有两个重要的区别：

该issueType字段必须对应一个子任务问题类型（可以使用/issue/createmeta发现子任务问题类型），和你必须提供一个parent在外地的问题创建包含ID或父问题的关键要求。

请求

例

```
{
  "update": {
    "worklog": [
      {
        "add": {
          "timeSpent": "60m",
          "started": "2011-07-05T11:05:00.000+0000"
        }
      }
    ]
  },
  "fields": {
    "project": {
      "id": "10000"
    },
    "summary": "something's wrong",
    "issuetype": {
      "id": "10000"
    },
    "assignee": {
      "name": "homer"
    },
    "reporter": {
      "name": "smithers"
    },
    "priority": {
      "id": "20000"
    },
    "labels": [
      "bugfix",
      "blitz_test"
    ],
    "timetracking": {
      "originalEstimate": "10",
      "remainingEstimate": "5"
    },
    "security": {
      "id": "10000"
    },
    "versions": [
      {
        "id": "10000"
      }
    ]
  }
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回的链接创建问题。

例

```
{
  "id": "10000",
  "key": "TST-24",
  "self": "http://www.example.com/jira/rest/api/2/issue/10000"
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效字段值，等等）。

例

```
{
  "errorMessages": [
    "Field 'priority' is required"
  ],
  "errors": {}
}
```

▼ 创建问题

POST /rest/api/2/issue/bulk

从创建JSON表示问题或子任务。

创建一个批量操作的许多问题。

创建一个子任务类似于创建一个普通的问题。更多细节可以在**createIssue**部分找到: {@link IssueResource#createIssue (IssueUpdateBean)}

请求

例

```
{
  "issueUpdates": [
    {
      "update": {
        "worklog": [
          {
            "add": {
              "timeSpent": "60m",
              "started": "2011-07-05T11:05:00.000+0000"
            }
          }
        ]
      },
      "fields": {
        "project": {
          "id": "10000"
        },
        "summary": "something's wrong",
        "issuetype": {
          "id": "10000"
        },
        "assignee": {
          "name": "homer"
        },
        "reporter": {
          "name": "smithers"
        },
        "priority": {
          "id": "20000"
        },
        "labels": [
          "bugfix",
          "blitz_test"
        ],
        "timetracking": {
          "originalEstimate": "10",
          "remainingEstimate": "5"
        },
        "security": {
          "id": "10000"
        },
        "versions": [

```

▶ 架构

回复

状态201 - 应用程序/JSON 返回一个链接到创建的问题。

例

```
{
  "issues": [
    {
      "id": "10000",
      "key": "TST-24",
      "self": "http://www.example.com/jira/rest/api/2/issue/10000"
    },
    {
      "id": "10001",
      "key": "TST-25",
      "self": "http://www.example.com/jira/rest/api/2/issue/10001"
    }
  ],
  "errors": []
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效字段值，等等）。

例

```
{
  "status": 400,
  "elementErrors": [
    "errorMessages": [
      "Field 'priority' is required"
    ],
    "errors": {}
  ],
  "failedElementNumber": 3
}
```

▶ 架构

▼ 找问题

GET /rest/api/2/issue/{issueIdOrKey}

返回给定问题关键问题的完整表示。

一个问题JSON由问题的关键，字段的集合，一个连接到工作流程变换分资源，和（可选），支持它的任何字段的HTML渲染的值（例如，如果维基语法的说明或注释启用）。

该fields参数（可多次指定）给出了一个逗号分隔的字段列表中应对包括。这可以用于检索字段的子集。一个特定的领域可以通过负前缀就被排除在外。

默认情况下，所有的(*all)字段在此得到发放的资源返回。注意：做一个JQL搜索时默认的是不同的-默认有刚通航领域(*navigable)。

*all - 包括所有领域

*navigable - 包括刚刚通航领域

summary, comment - 仅包含了总结和评论

-comment-包括除注释一切（默认为*all为GET-问题）

*all, -comment - 除了包含所有评论

该{@code性能参数}类似于{@code场}并指定用逗号分隔的问题属性列表中包含。不同于{@code领域}，属性默认不包括在内。要包含所有这些发送{@code？属性=*所有}。您也可以只包括指定的属性或排除与负一些属性（-）标志。

{@code *所有} - 包括所有属性

{@code *所有, -prop1} - 包括除了{@code为prop1}所有属性

{@code为prop1, 为prop1} - 包括{@code为prop1}和{@code prop2}性能

JIRA将试图通过找出问题的issueIdOrKey路径参数。这可能是一个问题的id，或问题的关键。如果无法通过精确匹配中发现的问题，JIRA也将查找问题不区分大小写的方式，或者通过寻找，看看问题是否被感动了。在这两种情况下，请求将正常进行（302或其他重定向将不退还）。包含在响应中的关键问题，将表明问题的主要的当前值。

该expand参数用于包括，在默认情况下，响应的部分隐藏。这可以用于为包括：

renderedFields - 以HTML格式字段值

names - 每场的显示名称

schema - 模式为每个字段描述一种类型的场的

transitions - 对给定问题的所有可能的转换

operations - 这可能在问题被应用于所有的候选条件操作

editmeta - 每个字段可以如何编辑的信息。它包含字段的架构也是如此。

changelog - 给定问题的所有更改历史记录

versionedRepresentations-各领域的REST表示。有些领域可能包含较新版本。RESET表示屈指可数。最大的数字始终代表最新的版本。建议在最新版本被使用。版本这些字段其提供一个更近的REST表示。包括后versionedRepresentations“域”字段变为隐藏。

请求

查询参数

参数	类型	描述
fields	串	字段列表返回的问题。默认情况下，所有字段返回。
expand	串	

properties

串

属性的列表，以返回该问题。默认情况下没有属性被返回。

回复

状态200 - 应用程序/ JSON 返回JSON格式的JIRA问题的完整表示。

例

```
{
  "expand": "renderedFields,names,schema,operations,editmeta,changelog,versionedRepresentations",
  "id": "10002",
  "self": "http://www.example.com/jira/rest/api/2/issue/10002",
  "key": "EX-1",
  "fields": {
    "watcher": [
      {
        "self": "http://www.example.com/jira/rest/api/2/issue/EX-1/watchers",
        "isWatching": false,
        "watchCount": 1,
        "watchers": [
          {
            "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
            "name": "fred",
            "displayName": "Fred F. User",
            "active": false
          }
        ]
      }
    ],
    "attachment": [
      {
        "self": "http://www.example.com/jira/rest/api/2.0/attachments/10000",
        "filename": "picture.jpg",
        "author": {
          "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
          "name": "fred",
          "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
          },
          "displayName": "Fred F. User",
          "active": false
        },
        "created": "2016-09-06T09:48:12.753+0000",
        "size": 23123,
        "mimeType": "image/jpeg",
        "content": "http://www.example.com/jira/attachments/10000",
        "thumbnail": "http://www.example.com/jira/secure/thumbnail/10000"
      }
    ]
  }
}
```

▶ 架构

状态404 返回如果未找到所请求的问题，或者用户没有权限查看。

▼ 删除问题

DELETE /rest/api/2/issue/{issueIdOrKey}

删除的问题。

如果问题有子任务，则必须设置参数`deleteSubtasks = true`来删除该问题。没有它的子任务也被删除您不能删除的问题。

请求

查询参数

参数	类型	描述
deleteSubtasks	串	true或false表明任何子任务也应删除的字符串。如果问题没有子任务忽略此参数。如果问题有子任务，这参数丢失或假的，那么问题将不会被删除，并且将返回一个错误。

回复

状态400 如果发生错误，返回。**状态401** 返回如果调用用户没有通过验证。**状态204** 如果返回成功删除的问题。**状态403** 返回如果主叫用户不具有权限删除的问题。**状态404** 如果这个问题不存在，则返回。

▼ 编辑问题

PUT /rest/api/2/issue/{issueIdOrKey}

编辑从JSON表示的问题。

这个问题可以通过设置明确的字段值 (S) 或通过使用一个操作来改变字段的值进行更新。

可以更新，在任一字段参数或更新参数的字段，可以使用被确定/休息/ API / 2 /问题/ `{issueIdOrKey}` / editmeta的资源。如果一个字段没有被配置为显示在编辑画面上的话，就不会在editmeta，并且如果它被提交会发生场验证错误。

指定“FIELD_ID”：在“域”FIELD_VALUE是“更新”部分中的一个“套”操作的简写。

现场应以“田”或“更新”要么出现，而不是两者。

请求

查询参数

参数	类型	描述
notifyUsers	布尔	并发出通知，这个问题已更新到看着它的用户发送电子邮件。管理员或项目管理员权限才能禁用通知。 默认值：真

例

```
{"update":{"summary":[{"set":"Bug in business logic"}],"components":[{"set":""}],"timetracking":[{"edit":{"originalEstimate":"1w 1d","remainingEstimate":}}
```

▶ 架构

回复

状态400 返回如果请求的问题，更新失败。

状态204 如果它成功更新的问题返回。

状态403 返回如果用户没有权限禁用用户通知。

▼ 分配

PUT /rest/api/2/issue/{issueIdOrKey}/assignee

分配问题给用户。你可以使用这个资源分配的问题，当用户提交请求具有分配权限，但不能编辑问题的权限。如果名字是“-1”自动受让人被使用。一个空的名字将删除受让人。

请求

例

```
{ "name": "harry" }
```

▶ 架构

回复

状态400 如果没有与接收到的用户表示一个问题返回。

状态401 返回如果主叫用户不具有权限分配的问题。

状态204 如果发行成功返回分配。

状态404 如果任一问题，或者用户不存在退货。

▼ 获得评论

GET /rest/api/2/issue/{issueIdOrKey}/comment

返回一个问题的所有意见。

结果可以通过“创造”领域，这意味着增加了一个注释的日期进行排序。

请求

查询参数

参数	类型	描述
startAt	长	页面偏移，如果不指定，则默认为0
maxResults	INT	在页面上的许多结果应被包括在内。默认为50。
orderBy	串	结果的排序。
expand	串	可选标志： renderedBody (提供HTML渲染体)

回复

状态200 - 应用程序/JSON 返回与此问题相关，具有计数和分页信息意见的集合。

例

```
{
  "startAt": 0,
  "maxResults": 1,
  "total": 1,
  "comments": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/10010/comment/10000",
      "id": "10000",
      "author": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "displayName": "Fred F. User",
        "active": false
      },
      "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed",
      "updateAuthor": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "displayName": "Fred F. User",
        "active": false
      },
      "created": "2016-09-06T09:48:12.346+0000",
      "updated": "2016-09-06T09:48:12.346+0000",
      "visibility": {
        "type": "role",
        "value": "Administrators"
      }
    }
  ]
}
```

▶ 架构

状态404 如果给定id /关键问题不存在，或者如果当前身份验证的用户没有权限查看它返回。

▼ 添加评论

POST /rest/api/2/issue/{issueIdOrKey}/comment

增加了一个问题，一个新的注释。

请求

查询参数

参数	类型	描述
expand	串	可选标志: renderedBody (提供HTML渲染体)

例

```
{
  "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed",
  "visibility": {
    "type": "role",
    "value": "Administrators"
  }
}
```

▶ 架构

回复

状态201 如果返回添加成功

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/comment/10000",
  "id": "10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed",
  "updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "created": "2016-09-06T09:48:12.346+0000",
  "updated": "2016-09-06T09:48:12.346+0000",
  "visibility": {
    "type": "role",
    "value": "Administrators"
  }
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

▼ 获取评论

GET /rest/api/2/issue/{issueIdOrKey}/comment/{id}

返回一个注释。

请求

查询参数

参数	类型	描述
expand	串	可选标志： <code>renderedBody</code> (提供HTML渲染体)

回复

状态200 - 应用程序/JSON 返回JSON格式的JIRA注释的完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/comment/10000",
  "id": "10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed",
  "updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "created": "2016-09-06T09:48:12.346+0000",
  "updated": "2016-09-06T09:48:12.346+0000",
  "visibility": {
    "type": "role",
    "value": "Administrators"
  }
}
```

▶ 架构

状态404 返回如果未找到所请求的评论，或者用户没有权限查看。

▼ 更新评论

PUT /rest/api/2/issue/{issueIdOrKey}/comment/{id}

更新使用它的JSON表示现有注释。

请求

查询参数

参数	类型	描述
expand	串	可选标志： <code>renderedBody</code> (提供HTML渲染体)

例

```
{
  "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed",
  "visibility": {
    "type": "role",
    "value": "Administrators"
  }
}
```

▶ 架构

回复

状态200 如果返回更新成功

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/comment/10000",
  "id": "10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "body": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum. Sed updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "created": "2016-09-06T09:48:12.346+0000",
  "updated": "2016-09-06T09:48:12.346+0000",
  "visibility": [
    {
      "type": "role",
      "value": "Administrators"
    }
  ]
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

▼ 删除评论

DELETE /rest/api/2/issue/{issueIdOrKey}/comment/{id}

删除现有评论。

回复

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。**状态204** 如果返回删除成功

▼ 获取编辑问题元

GET /rest/api/2/issue/{issueIdOrKey}/editmeta

返回编辑的问题元数据。

在editmeta字段对应于编辑屏幕的问题领域。不能在屏幕上的字段将不会在editmeta。

回复

状态200 - 应用程序/JSON 返回包含地图FieldBeans为当前用户可编辑的字段的响应。

例

```
{
  "fields": {
    "summary": {
      "required": false,
      "schema": {
        "type": "array",
        "items": "option",
        "custom": "com.atlassian.jira.plugin.system.customfieldtypes:multiselect",
        "customId": 10001
      },
      "name": "My Multi Select",
      "hasDefaultValue": false,
      "operations": [
        "set",
        "add"
      ],
      "allowedValues": [
        "red",
        "blue"
      ]
    }
  }
}
```

▶ 架构

状态404 或者返回如果未找到所请求的问题的用户没有权限查看。

▼ 通知

POST /rest/api/2/issue/{issueIdOrKey}/notify

发送一个通知（电子邮件），以在所述请求中定义的列表或收件人。

请求

例

```
{
  "subject": "Duis eu justo eget augue iaculis fermentum.",
  "textBody": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum.",
  "htmlBody": "Lorem ipsum <strong>dolor</strong> sit amet, consectetur adipiscing elit. Pellentesque eget venenatis elit. Duis eu justo eget augue iaculis fermentum.",
  "to": [
    {
      "reporter": false,
      "assignee": false,
      "watchers": true,
      "voters": true,
      "users": [
        {
          "name": "fred",
          "active": false
        }
      ],
      "groups": [
        {
          "name": "notification-group",
          "self": "http://www.example.com/jira/rest/api/2/group?groupname=notification-group"
        }
      ]
    }
  ],
  "restrict": {
    "groups": [
      {
        "name": "notification-group",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=notification-group"
      },
      "permissions": [
        {
          "id": "10",
          "key": "BROWSE"
        }
      ]
    }
  }
}
```

▶ 架构

回复

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

状态204 回到如果增加了邮件队列成功

状态403 返回的外发邮件被禁用或没有SMTP服务器的定义。

▼ 获取远程问题链接

GET /rest/api/2/issue/{issueIdOrKey}/remotelink

代表在这个问题上的远程问题链接休息子资源。

请求

查询参数

参数	类型	描述
globalId	串	要返回的远程问题链接的ID。如果返回null（未提供）的问题的所有远程链接。 对于问题链接字段的fullexplanation请参考 https://developer.atlassian.com/display/JIRADEV/Fields+in+Remote+Issue+Links

回复

状态200 - 应用程序/JSON 对本期远程问题链接的信息。

例

```
[ {
    "id": 10000,
    "self": "http://www.example.com/jira/rest/api/issue/MKY-1/remotelink/10000",
    "globalId": "system=http://www.mycompany.com/support&id=1",
    "application": {
        "type": "com.acme.tracker",
        "name": "My Acme Tracker"
    },
    "relationship": "causes",
    "object": {
        "url": "http://www.mycompany.com/support?id=1",
        "title": "TSTSUP-111",
        "summary": "Crazy customer support issue",
        "icon": {
            "url16x16": "http://www.mycompany.com/support/ticket.png",
            "title": "Support Ticket"
        }
    },
    "status": {
        "resolved": true,
        "icon": {
            "url16x16": "http://www.mycompany.com/support/resolved.png",
            "title": "Case Closed",
            "link": "http://www.mycompany.com/support?id=1&details=closed"
        }
    }
},
{
    "id": 10001,
    "self": "http://www.example.com/jira/rest/api/issue/MKY-1/remotelink/10001",
    "globalId": "system=http://www.anothercompany.com/tester&id=1234",
    "application": {
        "type": "com.acme.tester",
        "name": "My Acme Tester"
    },
    "relationship": "is tested by",
    "object": {
        "url": "http://www.anothercompany.com/tester/testcase/1234",
        "title": "Test Case #1234",
        "summary": "Test that the submit button saves the thing",
        "icon": {
            ...
        }
    }
}]
```

▶ 架构

状态401 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限查看远程问题链接，或者如果问题链接被禁用。**状态404** 如果不存在问题或远程问题链接返回。

▼ 创建或更新远程连接问题

POST /rest/api/2/issue/{issueIdOrKey}/remotelink

创建或更新从JSON表示远程问题链接。如果提供了globalId和远程问题链路与globalId存在，远程问题链路被更新。否则，将创建远程问题链接。

请求

例

```
{
    "globalId": "system=http://www.mycompany.com/support&id=1",
    "application": {
        "type": "com.acme.tracker",
        "name": "My Acme Tracker"
    },
    "relationship": "causes",
    "object": {
        "url": "http://www.mycompany.com/support?id=1",
        "title": "TSTSUP-111",
        "summary": "Crazy customer support issue",
        "icon": {
            "url16x16": "http://www.mycompany.com/support/ticket.png",
            "title": "Support Ticket"
        }
    },
    "status": {
        "resolved": true,
        "icon": {
            "url16x16": "http://www.mycompany.com/support/resolved.png",
            "title": "Case Closed",
            "link": "http://www.mycompany.com/support?id=1&details=closed"
        }
    }
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回一个链接到创建/更新远程问题链接。

例

```
{
    "id": 10000,
    "self": "http://www.example.com/jira/rest/api/issue/MKY-1/remotelink/10000"
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

例

```
{
  "errorMessages": [],
  "errors": {
    "title": "'title' is required."
  }
}
```

状态401 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限创建/更新远程问题链接，或者如果问题挂钩是禁用的。**▼ 通过删除全局ID远程连接问题**

DELETE /rest/api/2/issue/{issueIdOrKey}/remotelink

删除与在这个问题上给定的全局ID的远程问题链接。

请求

查询参数

参数	类型	描述
globalId	串	远程问题链接的全局ID

回复**状态401** 返回如果调用用户没有通过验证。**状态204** 返回如果远程问题链接被成功摘除。**状态403** 返回如果调用用户没有权限删除远程连接问题，或者如果问题挂钩是禁用的。**状态404** 如果不存在问题或远程问题链接返回。**▼ 找ID远程连接问题**

GET /rest/api/2/issue/{issueIdOrKey}/remotelink/{linkId}

获取关于这一问题给定id远程问题链接。

回复**状态200** - 应用程序/JSON 给定ID的远程问题链接信息。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/issue/MKY-1/remotelink/10000",
  "globalId": "system=http://www.mycompany.com/support&id=1",
  "application": {
    "type": "com.acme.tracker",
    "name": "My Acme Tracker"
  },
  "relationship": "causes",
  "object": {
    "url": "http://www.mycompany.com/support?id=1",
    "title": "TSTSUP-111",
    "summary": "Crazy customer support issue",
    "icon": {
      "url16x16": "http://www.mycompany.com/support/ticket.png",
      "title": "Support Ticket"
    },
    "status": {
      "resolved": true,
      "icon": {
        "url16x16": "http://www.mycompany.com/support/resolved.png",
        "title": "Case Closed",
        "link": "http://www.mycompany.com/support?id=1&details=closed"
      }
    }
  }
}
```

▶ 架构

状态400 返回如果LINKID是不是有效的数字，或者如果给定id远程问题链接不属于特定问题。**状态401** 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限查看远程连接问题，或者如果问题挂钩是禁用的。**状态404** 如果不存在问题或远程问题链接返回。**▼ 远程更新链接的问题**

PUT /rest/api/2/issue/{issueIdOrKey}/remotelink/{linkId}

从更新JSON表示远程问题链接。没有提供任何字段都设置为null。

请求

例

```
{
  "globalId": "system=http://www.mycompany.com/support&id=1",
  "application": {
    "type": "com.acme.tracker",
    "name": "My Acme Tracker"
  },
  "relationship": "causes",
  "object": {
    "url": "http://www.mycompany.com/support?id=1",
    "title": "TSTSUP-111",
    "summary": "Crazy customer support issue",
    "icon": {
      "url16x16": "http://www.mycompany.com/support/ticket.png",
      "title": "Support Ticket"
    },
    "status": {
      "resolved": true,
      "icon": {
        "url16x16": "http://www.mycompany.com/support/resolved.png",
        "title": "Case Closed",
        "link": "http://www.mycompany.com/support?id=1&details=closed"
      }
    }
  }
}
```

▶ 架构

回复

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

例

```
{
  "errorMessages": [],
  "errors": {
    "title": "'title' is required."
  }
}
```

状态401 返回如果调用用户没有通过验证。

状态204 返回如果远程问题链接已成功更新。

状态403 如果返回调用用户没有权限更新远程问题链接，或者如果问题挂钩是禁用的。

▼ 通过删除ID远程连接问题

DELETE /rest/api/2/issue/{issueIdOrKey}/remotelink/{LinkId}

删除与在这个问题上给定id远程问题链接。

回复

状态401 返回如果调用用户没有通过验证。

状态204 返回如果远程问题链接被成功摘除。

状态403 返回如果调用用户没有权限删除远程连接问题，或者如果问题挂钩是禁用的。

状态404 如果不存在问题或远程问题链接返回。

▼ 获取转换

GET /rest/api/2/issue/{issueIdOrKey}/transitions

由当前用户获取针对此问题可能的转换的列表，以及被要求的字段和它们的类型沿着。

字段将仅在返回expand=transitions.fields。

在元数据字段对应于过渡屏幕过渡字段。不能在屏幕上的字段不会在元数据。

请求

查询参数

参数	类型	描述
transitionId	串	

回复

状态200 - 应用程序/JSON 返回指定问题的可能的过渡和执行过渡所需的字段的完整表示。

例

```
{
  "expand": "transitions",
  "transitions": [
    {
      "id": "2",
      "name": "Close Issue",
      "to": {
        "self": "http://localhost:8090/jira/rest/api/2.0/status/10000",
        "description": "The issue is currently being worked on.",
        "iconUrl": "http://localhost:8090/jira/images/icons/progress.gif",
        "name": "In Progress",
        "id": "10000",
        "statusCategory": {
          "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/1",
          "id": 1,
          "key": "in-flight",
          "colorName": "yellow",
          "name": "In Progress"
        }
      },
      "fields": {
        "summary": {
          "required": false,
          "schema": {
            "type": "array",
            "items": "option",
            "custom": "com.atlassian.jira.plugin.system.customfieldtypes:multiselect",
            "customId": 10001
          },
          "name": "My Multi Select",
          "hasDefaultValue": false,
          "operations": [
            "set",
            "add"
          ],
          "allowedValues": [
            "red",
            "blue"
          ]
        }
      }
    }
  ],
}
```

▶ 架构

状态404 或者返回如果未找到所请求的问题的用户没有权限查看。

▼ 做过渡

POST /rest/api/2/issue/{issueIdOrKey}/transitions

执行对一个问题的过渡。当执行过渡您可以更新或设置其他的问题领域。

可以在跃迁进行设置，在任一字段参数或更新参数的字段可以用来确定/**rest/api/2/issue/{issueIdOrKey}/transitions?expand=transitions.fields** 资源。如果字段没有被配置为显示在过渡屏幕上，那么它将不会在过渡元数据，并且如果它被提交会发生场验证错误。

请求

例

```
{
  "update": {
    "comment": [
      {
        "add": {
          "body": "Bug has been fixed."
        }
      }
    ]
  },
  "fields": {
    "assignee": {
      "name": "bob"
    },
    "resolution": {
      "name": "Fixed"
    }
  },
  "transition": {
    "id": "5"
  },
  "historyMetadata": {
    "type": "myplugin:type",
    "description": "text description",
    "descriptionKey": "plugin.changereason.i18.key",
    "activityDescription": "text description",
    "activityDescriptionKey": "plugin.activity.i18.key",
    "actor": {
      "id": "tony",
      "displayName": "Tony",
      "type": "mysystem-user",
      "avatarUrl": "http://mysystem/avatar/tony.jpg",
      "url": "http://mysystem/users/tony"
    },
    "generator": {
      "id": "mysystem-1",
      "type": "mysystem-application"
    },
    "cause": {
      "id": "myevent",
      "type": "mysystem-event"
    }
  }
},
```

▶ 架构

回复

状态400 如果没有指定的过渡。

状态204 如果返回的转变是成功的。

状态404 这个问题不存在，或者用户没有权限查看它

▼ 移除投票

DELETE /rest/api/2/issue/{issueIdOrKey}/votes

从问题中删除您的投票。（即“unvote”）

回复

状态204 没有返回成功。

状态404 如果用户不能删除任何理由投票返回。（用户没有对这个问题进行投票，用户是记者，投票被禁止，这个问题不存在，等等）

▼ 添加投票

POST /rest/api/2/issue/{issueIdOrKey}/votes

投你的票赞成的一个问题。

回复

状态204 没有返回成功。

状态404 返回如果用户不能投票给任何理由。（用户是记者，用户没有权限进行投票，表决是在实例禁用，该问题不存在，等等）

▼ 获得选票

GET /rest/api/2/issue/{issueIdOrKey}/votes

代表在这个问题上，选民休息子资源。

回复

状态200 - 应用程序/JSON 关于本期投票信息。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/MKY-1/votes",
  "votes": 24,
  "hasVoted": true,
  "voters": [
    {
      "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
      "name": "fred",
      "avatarUrls": {
        "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
        "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
        "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
        "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
      },
      "displayName": "Fred F. User",
      "active": false
    }
  ]
}
```

► 架构

状态404 如果用户不能在问题来看待这个问题或投票被禁止返回。

▼ 获取问题观察家

GET /rest/api/2/issue/{issueIdOrKey}/watchers

返回观察家与给定键的问题的列表。

回复

状态200 - 应用程序/JSON 返回观察者对问题的列表。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/EX-1/watchers",
  "isWatching": false,
  "watchCount": 1,
  "watchers": [
    {
      "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
      "name": "fred",
      "displayName": "Fred F. User",
      "active": false
    }
  ]
}
```

▶ 架构

状态404 返回如果未找到所请求的问题，或者用户没有权限查看。

▼ 添加守望

POST /rest/api/2/issue/{issueIdOrKey}/watchers

将用户添加到一个问题的守望者名单。

请求

例

`"fred"`

▶ 架构

回复

状态400 如果没有与接收到的用户表示一个问题返回。**状态401** 如果返回调用用户没有权限到观察者加入到观察的问题的列表。**状态204** 如果返回成功添加的守望者。**状态404** 如果任一问题，或者用户不存在退货。

▼ 删除守望者

DELETE /rest/api/2/issue/{issueIdOrKey}/watchers

移除一个问题的守望者列表中的用户。

请求

查询参数

参数	类型	描述
username	串	包含用户的名称的字符串从监视器列表中删除。不能为null。

回复

状态400 返回如果不提供用户名查询参数。**状态401** 如果返回调用用户没有权限从观察者的问题的列表中删除观察者。**状态204** 如果返回观察者被成功取出。**状态404** 如果任一问题不存在，则返回。

▼ 获取问题工作日志

GET /rest/api/2/issue/{issueIdOrKey}/worklog

返回所有工作日志的问题。注意：如果日志的工作领域是隐藏的项目工作日志将不予退还。

回复

状态200 - 应用程序/JSON 返回与此问题相关，具有计数和分页信息**worklogs**的集合。

例

```
{
  "startAt": 0,
  "maxResults": 1,
  "total": 1,
  "worklogs": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/10010/worklog/10000",
      "author": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "displayName": "Fred F. User",
        "active": false
      },
      "updateAuthor": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "displayName": "Fred F. User",
        "active": false
      },
      "comment": "I did some work here.",
      "updated": "2016-09-06T09:48:12.832+0000",
      "visibility": {
        "type": "group",
        "value": "jira-developers"
      },
      "started": "2016-09-06T09:48:12.831+0000",
      "timeSpent": "3h 20m",
      "timeSpentSeconds": 12000,
      "id": "100028",
      "issueId": "10002"
    }
  ]
}
```

▶ 架构

状态404 如果给定 `id` / 关键问题不存在，或者如果当前身份验证的用户没有权限查看它返回。

▼ 添加工作日志

POST /rest/api/2/issue/{issueIdOrKey}/worklog

添加一个新的工作日志条目的问题。

请求

查询参数

参数	类型	描述
adjustEstimate	串	(可选)，您可以提供具体的说明更新问题的剩余时间估计。有效值为 “新” - 将估计为特定值 “离开” - 取消估计因为是 “手动” - 指定一个特定的量，以增加估计剩余 “自动” - 默认选项。会自动调整的基础上的工作日志中指定的新 <code>timeSpent</code> 值
newEstimate	串	(当“新”被选择为 <code>adjustEstimate</code> 必需) 对于剩余估计字段中的新值。例如“二维”
reduceBy	串	(当需要时“手动”选择为 <code>adjustEstimate</code>) 由例如“二维”，以减少剩余的估计量

例

```
{
  "comment": "I did some work here.",
  "visibility": {
    "type": "group",
    "value": "jira-developers"
  },
  "started": "2016-09-06T09:48:12.830+0000",
  "timeSpentSeconds": 12000
}
```

▶ 架构

回复

状态201 如果返回添加成功

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

状态403 如果返回调用用户没有权限添加工作日志

▼ 获取工作日志

GET /rest/api/2/issue/{issueIdOrKey}/worklog/{id}

返回一个特定的工作日志。注意：如果日志的工作领域是隐藏该项目的工作日志将不予退还。

回复

状态200 - 应用程序/JSON 如果工作日志指定id存在返回，目前已验证的用户有权限查看。返回的响应包含工作日志JSON格式的完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/worklog/10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "comment": "I did some work here.",
  "updated": "2016-09-06T09:48:12.832+0000",
  "visibility": {
    "type": "group",
    "value": "jira-developers"
  },
  "started": "2016-09-06T09:48:12.831+0000",
  "timeSpent": "3h 20m",
  "timeSpentSeconds": 12000,
  "id": "100028",
  "issueId": "10002"
}
```

▶ 架构

状态404 如果工作日志与给定的ID不存在，或者如果当前身份验证的用户没有权限查看它返回。

▼ 更新工作日志

PUT /rest/api/2/issue/{issueIdOrKey}/worklog/{id}

更新现有工作日志条目。

注意：

可能的编辑字段有：评论，知名度，开始，timeSpent和timeSpentSeconds。

无论是timeSpent或timeSpentSeconds可以设置。

这是没有设置的字段将不会被更新。

对于一个请求是有效的，它必须具有至少一个字段的改变。

请求

查询参数

参数	类型	描述
adjustEstimate	串	(可选)， 您可以提供具体的说明更新问题的剩余时间估计。有效值为 “新” - 将估计为特定值 “离开” - 叶子估计因为是 “自动” - 默认选项。会自动调整的基础上的工作日志中指定的新timeSpent值
newEstimate	串	(当“新”被选择为adjustEstimate必需) 对于剩余估计字段中的新值。

例

```
{
  "comment": "I did some work here.",
  "visibility": {
    "type": "group",
    "value": "jira-developers"
  },
  "started": "2016-09-06T09:48:12.830+0000",
  "timeSpentSeconds": 12000
}
```

▶ 架构

回复

状态200 如果返回更新成功

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/worklog/10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "comment": "I did some work here.",
  "updated": "2016-09-06T09:48:12.832+0000",
  "visibility": {
    "type": "group",
    "value": "jira-developers"
  },
  "started": "2016-09-06T09:48:12.831+0000",
  "timeSpent": "3h 20m",
  "timeSpentSeconds": 12000,
  "id": "100028",
  "issueId": "10002"
}
```

▶ 架构

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

状态403 返回如果主叫用户没有权限来更新工作日志

▼ 删除工作日志

DELETE /rest/api/2/issue/{issueIdOrKey}/worklog/{id}

删除现有的工作日志条目。

请求

查询参数

参数	类型	描述
adjustEstimate	串	(可选)，您可以提供具体的说明更新问题的剩余时间估计。有效值为 “新” - 将估计为特定值 “离开” - 叶子估计因为是 “手动” - 指定一个特定的量，以增加估计剩余 “自动” - 默认选项。会自动调整的基础上的工作日志中指定的新timeSpent值
newEstimate	串	(当“新”被选择为adjustEstimate必需) 对于剩余估计字段中的newValue。例如“二维”
increaseBy	串	(必要时“手动”选择为adjustEstimate)，例如由“2D”，以增加估计剩余量

回复

状态400 如果返回输入无效（例如缺少必填字段，无效值，等等）。

状态204 如果返回删除成功

状态403 如果返回调用用户没有权限删除工作日志

▼ 获取创建问题元

GET /rest/api/2/issue/createmeta

返回创建问题的元数据。这包括现有的项目，问题类型和领域，包括字段类型以及是否这些字段是必需的。如果用户没有权限创建在该项目问题的项目将不予退还。

在createmeta字段对应于创建屏幕的项目/问题类型的字段。不能在屏幕上的字段将不会在createmeta。

字段将仅在返回expand=projects.issuetypes.fields。

结果可以按项目和/或问题类型进行过滤，由查询参数给出。

请求

查询参数

参数	类型	描述
projectIds	串	与projectKeys参数相结合，列出了其筛选结果的项目。如果不存在，所有的项目都返回。此参数可指定多次，和/或以逗号分隔。

号分隔的列表。Specifying不存在（或者说你不能在创建问题）的项目是不是一个错误，但它不会在结果中。

projectKeys	串	与projectIds参数相结合，列出了其筛选结果的项目。如果为null，所有的项目都返回。此参数可指定多次，和/或以逗号分隔的列表。Specifying不存在（或者说你不能在创建问题）的项目是不是一个错误，但它不会在结果中。
issuetypeIds	串	与issuetypeNames combined，列出问题的类型，用以过滤结果。如果为null，所有的问题类型返回。此参数可指定多次，和/或以逗号分隔的列表。Specifying不存在的一个问题类型不是一个错误。
issuetypeNames	串	与issuetypeIds combined，列出问题的类型，用以过滤结果。如果为null，所有的问题类型返回。此参数可指定多次，但并不解释为一个逗号分隔的列表。Specifying不存在的一个问题类型不是一个错误。

回复

状态200 - 应用程序/JSON 返回创建问题的元数据。

例

```
{
  "expand": "projects",
  "projects": [
    {
      "self": "http://www.example.com/jira/rest/api/2/project/EX",
      "id": "10000",
      "key": "EX",
      "name": "Example Project",
      "avatarUrls": {
        "48x48": "http://www.example.com/jira/secure/projectavatar?pid=10000&avatarId=10011",
        "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10000&avatarId=10011",
        "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10000&avatarId=10011",
        "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10000&avatarId=10011"
      },
      "issuetypes": [
        {
          "self": "http://www.example.com/jira/rest/api/2/issueType/1",
          "id": "1",
          "description": "An error in the code",
          "iconUrl": "http://www.example.com/jira/images/icons/issuetypes/bug.png",
          "name": "Bug",
          "subtask": false,
          "fields": {
            "issuetype": {
              "required": true,
              "name": "Issue Type",
              "hasDefaultValue": false,
              "operations": [
                "set"
              ]
            }
          }
        }
      ]
    }
  ]
}
```

▶ 架构

状态403 返回如果用户没有权限查看任何所要求的项目。

▼ 找问题选择器资源

GET /rest/api/2/issue/picker

返回建议匹配，其执行该请求的用户自动完成查询哪些问题。这种方法REST将检查用户的历史记录和用户的浏览器上下文并选择此问题，匹配查询哪些。

请求

查询参数

参数	类型	描述
query	串	查询。
currentJQL	串	其中执行请求在上下文中JQL。只有符合这JQL查询哪些问题将被纳入结果。
currentIssueKey	串	其中执行请求的问题的上下文中的关键。这是在上下文问题将不被包括在自动完成的结果，即使它的查询相匹配。
currentProjectId	串	其中执行请求的项目的上下文中的ID。建议的问题将只能从该项目。
showSubTasks	布尔	如果设置为false，子任务将不包括在列表中。
showSubTaskParent	布尔	如果设置为false，请求以子任务的上下文中执行，父问题不会被列入自动完成的结果，即使它匹配查询。

回复

状态200 - 应用程序/JSON 返回的匹配选择器参数，问题列表。

▶ 架构

API / 2 / 问题/ { issueIdOrKey } /附件

[隐藏所有方法](#)

问题附件

▼ 添加附件

POST /rest/api/2/issue/{issueIdOrKey}/attachments

添加一个或多个附件的问题。

该资源预期一个多职。媒体类型的multipart / form-data的是在RFC 1867中定义的大多数客户端库都使处理简单的多部分职位类别。例如，在Java的Apache HTTP组件库提供了[MultiPartEntity](#)，使得它简单提交多部分POST。

为了防范XSRF攻击，因为这种方法接受的multipart / form-data的，它有它XSRF保护。这意味着你必须提交的X Atlassian的令牌的头部：该请求没有检查，否则将被阻止。

包含附件必须是“文件”中的multipart / form-data的参数的名称

一个简单的例子来上传了一个名为“myfile.txt”的发行REST-123文件：

```
卷曲-D- -u管理: 管理员-X POST -H "X-Atlassian的令牌: 没有勾选" -F "file=@myfile.txt的" http://为myhost / REST / API / 2 / 问题/ TE
```

回复

[状态200](#) - 应用程序/JSON

例

```
[ {
    "self": "http://www.example.com/jira/rest/api/2.0/attachments/10000",
    "filename": "picture.jpg",
    "author": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    },
    "created": "2016-09-06T09:48:12.753+0000",
    "size": 23123,
    "mimeType": "image/jpeg",
    "content": "http://www.example.com/jira/attachments/10000",
    "thumbnail": "http://www.example.com/jira/secure/thumbnail/10000"
},
{
    "self": "http://www.example.com/jira/rest/api/2.0/attachments/10001",
    "filename": "dbleuglog.txt",
    "author": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    },
    "created": "2016-09-06T09:48:12.753+0000",
    "size": 2460,
    "mimeType": "text/plain",
    "content": "http://www.example.com/jira/attachments/10001",
    "thumbnail": "http://www.example.com/jira/secure/thumbnail/10002"
}]
```

► 架构

[状态403](#) 如果附件被禁用或退回，如果你没有权限将附件添加到这个问题。

[状态404](#) 返回如果未找到所请求的问题，用户没有权限查看，或者如果附件超出配置的最大附件大小。

API / 2 / 问题/ { issueIdOrKey } /属性

[隐藏所有方法](#)

▼ 获取属性键

[实验](#)

GET /rest/api/2/issue/{issueIdOrKey}/properties

返回由键或用id标识问题的所有属性的钥匙。

回复

[状态200](#) - 应用程序/JSON 如果发现问题返回。

例

```
{
  "keys": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
      "key": "issue.support"
    }
  ]
}
```

▶ 架构

状态400 返回如果问题键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 返回如果调用用户没有权限查看该问题。**状态404** 如果返回与给定的密钥或ID的问题不存在，或者如果与给定键的属性没有找到。▼ 删除特性 实验

DELETE /rest/api/2/issue/{issueIdOrKey}/properties/{propertyKey}

移除按键式或由id标识的问题的性质。THS用户删除属性必须有权限修改的问题。

回复

状态400 返回如果问题键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态204** 如果返回成功删除问题属性。**状态403** 如果返回调用用户没有权限修改的问题。**状态404** 如果返回与给定的密钥或ID的问题不存在，或者如果与给定键的属性没有找到。▼ 设置属性 实验

PUT /rest/api/2/issue/{issueIdOrKey}/properties/{propertyKey}

设置指定的问题的属性的值。

你可以使用这个资源来存储反对键或由id标识的问题，自定义数据。谁存储数据的用户需要具有权限编辑的问题。

回复

状态200 返回如果问题属性成功更新。**状态201** 如果创建成功，问题属性返回。**状态400** 返回如果问题键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 如果返回调用用户没有权限修改的问题。**状态404** 如果与给定的密钥或ID的问题不存在，则返回。▼ Get属性 实验

GET /rest/api/2/issue/{issueIdOrKey}/properties/{propertyKey}

返回与从由键或由id标识的问题给定键的属性的值。谁检索属性的用户需要具有的权限来读取的问题。

回复

状态200 - 应用程序/ JSON 如果问题财产被发现返回。

例

```
{
  "key": "issue.support",
  "value": {
    "hipchat.room.id": "support-123",
    "support.time": "1m"
  }
}
```

▶ 架构

状态400 返回如果问题键或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 返回如果调用用户没有权限查看该问题。

状态404 如果返回与给定的密钥或ID的问题不存在，或者如果与给定键的属性没有找到。

API / 2 / 问题/ {} issueIdOrKey /子任务 隐藏所有方法

▼ 获取子任务

GET /rest/api/2/issue/{issueIdOrKey}/subtask

返回问题的子任务列表

回复

状态200

▶ 架构

状态403 返回如果用户不能编辑问题

状态404 如果这个问题不存在，则返回

▼ 可以将子任务

GET /rest/api/2/issue/{issueIdOrKey}/subtask/move

回复

状态 - 应用程序/ JSON

▼ 将子任务

POST /rest/api/2/issue/{issueIdOrKey}/subtask/move

通过指数“由”动子任务索引“到”重新排序问题的子任务。

请求

▶ 架构

回复

状态400 返回如果从或参数出界

状态204 返回如果请求成功

▶ 架构

状态403 返回如果用户不能编辑问题

状态404 如果父问题不存在，则返回

API / 2 / issueLink 隐藏所有方法

领汇发行资源提供的功能来管理问题的链接。

▼ 链接问题

POST /rest/api/2/issueLink

创建两个问题之间的问题的链接。用户需要将被链接到另一个问题，这个问题的链接问题的权限。在请求中指定的链接类型用于创建链接并且将创建从第一个问题的第二个问题使用向外描述的链接。它还创建了第二个问题，第一个问题使用问题链接类型的向内说明的链接。它将提供的注释添加到第一个问题。注释可以限制谁可以查看它。如果指定了组，只有这个组的用户可以查看这条评论，如果roleLevel只指定谁拥有特定角色可以查看此评论的用户。谁创建问题链路用户需要属于指定组或具有指定的作用。

请求

例

```
{
  "type": {
    "name": "Duplicate"
  },
  "inwardIssue": {
    "key": "HSP-1"
  },
  "outwardIssue": {
    "key": "MKY-1"
  },
  "comment": {
    "body": "Linked related issue!",
    "visibility": {
      "type": "group",
      "value": "jira-software-users"
    }
  }
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果问题链接创建成功。**状态400** 如果它不能创建提供的注释。响应将包含指示错误消息为什么失败创建注释。如果未能创建注释没有问题的链接将被创建。**状态401** 如果用户不具有的问题，这将被链接到另一个问题链路问题的权限。**状态500** 如果创建问题链接或评论时发生错误。**状态404** 如果问题链接被禁用或无法找到的问题之一（问题可能存在，但它是不可见此用户），或无法找到指定的问题链接类型。

▼ 获取链接的问题

GET /rest/api/2/issueLink/{linkId}

返回指定ID的问题链接。

回复

状态200 - 应用程序/JSON

例

```
{
  "id": "10001",
  "type": {
    "id": "1000",
    "name": "Duplicate",
    "inward": "Duplicated by",
    "outward": "Duplicates",
    "self": "http://www.example.com/jira/rest/api/2/issueLinkType/1000"
  },
  "inwardIssue": {
    "id": "10004",
    "key": "PRJ-3",
    "self": "http://www.example.com/jira/rest/api/2/issue/PRJ-3",
    "fields": {
      "status": {
        "iconUrl": "http://www.example.com/jira/images/icons/statuses/open.png",
        "name": "Open"
      }
    }
  },
  "outwardIssue": {
    "id": "10004L",
    "key": "PRJ-2",
    "self": "http://www.example.com/jira/rest/api/2/issue/PRJ-2",
    "fields": {
      "status": {
        "iconUrl": "http://www.example.com/jira/images/icons/statuses/open.png",
        "name": "Open"
      }
    }
  }
}
```

▶ 架构

状态400 如果指定的问题链路ID无效。**状态401** 如果用户不具有的问题，这将被链接到另一个问题链路问题的权限。**状态500** 如果创建问题链接或评论时发生错误。**状态404** 如果问题链接被禁用或无法找到指定的ID的问题链接。或者是因为没有与此ID存在，或者用户没有看到链接的问题之一的权限。

▼ 删除链接的问题

DELETE /rest/api/2/issueLink/{linkId}

删除具有指定ID的问题链接。为了能够删除您必须能够查看这两个问题，一个问题链接，必须有链接问题权限的问题的至少一个。

回复

状态200 - 应用程序/JSON**状态400** 如果指定的问题链路ID无效。**状态401** 如果用户不具有对于该问题链接的源或目的地问题链路问题的权限。**状态500** 如果删除链接的问题或评论时发生错误。**状态204** 如果成功删除问题链接。**状态404** 如果问题链接被禁用或无法找到指定的ID的问题链接。或者是因为没有与此ID存在，或者用户没有看到链接的问题之一的权限。

API / 2 / issueLinkType 隐藏所有方法

其余资源检索问题链路类型的列表。

▼ 获取问题的链接类型

GET /rest/api/2/issueLinkType

返回可用问题的链接类型的列表，如果启用了问题挂钩。每个问题链接类型具有一个ID，一个名称和向外和向内链接关系的标签。

回复

状态200 - 应用程序/JSON 返回所有可用的问题的链接类型的列表。

例

```
{
  "issueLinkTypes": [
    {
      "id": "1000",
      "name": "Duplicate",
      "inward": "Duplicated by",
      "outward": "Duplicates",
      "self": "http://www.example.com/jira/rest/api/2//issueLinkType/1000"
    },
    {
      "id": "1010",
      "name": "Blocks",
      "inward": "Blocked by",
      "outward": "Blocks",
      "self": "http://www.example.com/jira/rest/api/2//issueLinkType/1010"
    }
  ]
}
```

▶ 架构

状态404 如果问题链接被禁用返回。

▼ 创建问题链接类型

POST /rest/api/2/issueLinkType

创建一个新的问题链接类型。

请求

例

```
{
  "name": "Duplicate",
  "inward": "Duplicated by",
  "outward": "Duplicates"
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 新的一期链接类型已创建。

例

```
{
  "id": "1000",
  "name": "Duplicate",
  "inward": "Duplicated by",
  "outward": "Duplicates",
  "self": "http://www.example.com/jira/rest/api/2//issueLinkType/1000"
}
```

▶ 架构

状态404 问题挂钩被禁用，或者您没有权限创建问题的链接类型。

▼ 找问题链接类型

GET /rest/api/2/issueLinkType/{issueLinkTypeId}

对于给定的问题链接类型的id返回有关此问题的链接类型的所有信息。

回复

状态200 - 应用程序/JSON 返回给定id的问题链接类型。

例

```
{
  "id": "1000",
  "name": "Duplicate",
  "inward": "Duplicated by",
  "outward": "Duplicates",
  "self": "http://www.example.com/jira/rest/api/2//issueLinkType/1000"
}
```

▶ 架构

状态404 如果问题链接被禁用或返回给定id没有问题的链接类型存在。

▼ 删除问题链接类型

DELETE /rest/api/2/issueLinkType/{issueLinkTypeId}

删除指定的问题链接类型。

回复

状态204

状态404 如果问题链接被禁用或返回给定id没有问题的链接类型存在。

▼ Update问题链接类型

PUT /rest/api/2/issueLinkType/{issueLinkTypeId}

更新指定的问题链接类型。

请求

例

```
{
  "name": "Duplicate",
  "inward": "Duplicated by",
  "outward": "Duplicates"
}
```

▶ 架构

回复

状态200

例

```
{
  "id": "1000",
  "name": "Duplicate",
  "inward": "Duplicated by",
  "outward": "Duplicates",
  "self": "http://www.example.com/jira/rest/api/2//issueLinkType/1000"
}
```

▶ 架构

状态400 返回如果提供的id是不是一个数字。

状态404 如果问题链接被禁用或返回给定id没有问题的链接类型存在。

API / 2 / issuesecurityschemes 隐藏所有方法

REST资源，允许查看该产品定义的安全方案。

▼ 获取问题的安全方案

GET /rest/api/2/issuesecurityschemes

返回定义的所有问题的保障计划。

回复

状态200 - 应用程序/JSON 返回用户是否具有管理员权限。

例

```
{
  "issueSecuritySchemes": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issuesecurityschemes/1000",
      "id": 1000,
      "name": "Default Issue Security Scheme",
      "description": "Description for the default issue security scheme",
      "defaultSecurityLevelId": 10021
    }
  ]
}
```

▶ 架构

状态401 返回如果用户没有登录。**状态403** 如果用户不具有管理员权限返回。

▼ 获取问题的安全方案

GET /rest/api/2/issuesecurityschemes/{id}

返回与沿问题的安全方案的定义。

回复

状态200 - 应用程序/JSON 如果用户具有管理员权限, 或如果方案中, 其中用户具有管理权限的一个项目使用的返回。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issuesecurityschemes/1000",
  "id": 1000,
  "name": "Default Issue Security Scheme",
  "description": "Description for the default issue security scheme",
  "defaultSecurityLevelId": 10021,
  "levels": [
    {
      "self": "http://www.example.com/jira/rest/api/2/securitylevel/10021",
      "id": "10021",
      "description": "Only the reporter and internal staff can see this issue.",
      "name": "Reporter Only"
    }
  ]
}
```

▶ 架构

状态401 返回如果用户没有登录。**状态403** 返回如果用户不具有管理员权限和方案不是在用户具有管理权限的任何项目使用。

API / 2 / 问题类型 隐藏所有方法

▼ 获得发布的所有类型

GET /rest/api/2/issuetype

返回给用户可见的所有问题类型的列表

回复

状态200 - 应用程序/JSON 如果问题的类型存在且由主叫用户可见返回。

例

```
[
  {
    "self": "http://localhost:8090/jira/rest/api/2.0/issueType/3",
    "id": "3",
    "description": "A task that needs to be done.",
    "iconUrl": "http://localhost:8090/jira/images/icons/issuetypes/task.png",
    "name": "Task",
    "subtask": false,
    "avatarId": 1
  },
  {
    "self": "http://localhost:8090/jira/rest/api/2.0/issueType/1",
    "id": "1",
    "description": "A problem with the software.",
    "iconUrl": "http://localhost:8090/jira/images/icons/issuetypes/bug.png",
    "name": "Bug",
    "subtask": false,
    "avatarId": 10002
  }
]
```

▶ 架构

▼ 创建问题类型

POST /rest/api/2/issuetype

从创建JSON表示问题类型，并增加了问题的新创建的问题类型为默认的问题类型方案。

请求

例

```
{
  "name": "name",
  "description": "description",
  "type": "standard"
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果问题类型创建成功返回。

▶ 架构

状态400 返回如果请求是无效的。这种情况发生时的名称是无效或问题类型的实例子任务已禁用的子任务。

状态401 返回如果调用用户没有通过验证。

状态403 如果返回调用用户没有权限管理JIRA。

状态409 如果已经存在具有指定名称的问题类型返回。

▼ 获取问题类型

GET /rest/api/2/issuetype/{id}

返回具有给定id的问题类型的完整表示。

回复

状态200 - 应用程序/JSON 如果问题的类型存在且由主叫用户可见返回。

例

```
{
  "self": "http://localhost:8090/jira/rest/api/2.0/issueType/3",
  "id": "3",
  "description": "A task that needs to be done.",
  "iconUrl": "http://localhost:8090/jira/images/icons/issuetypes/task.png",
  "name": "Task",
  "subtask": false,
  "avatarId": 1
}
```

▶ 架构

状态404 如果返回的问题类型不存在，或者不是给主叫用户可见。

▼ 删除问题类型

DELETE /rest/api/2/issuetype/{id}

删除指定的问题类型。如果问题类型有任何相关问题，这些问题将被迁移到在参数中指定的替代问题类型。您可以通过调用确定替代问题类型/ REST API / 2 /问题类型/ {ID} /替代资源。

请求

查询参数

参数	类型	描述
alternativeIssueTypeId	串	问题类型的ID到与被删除的问题类型相关的问题将被迁移。

回复

状态400 返回如果请求是无效的。它发生在有与正被去除的问题类型相关的问题，但它是不可能的这些问题迁移到替代问题类型。

状态401 返回如果调用用户没有通过验证。

状态204 - 应用程序/JSON 如果问题类型成功删除返回。

状态403 如果返回调用用户没有权限管理JIRA。

状态404 返回如果这是为了要除去的问题类型不存在或替代问题类型不存在。

▼ 更新问题类型

PUT /rest/api/2/issuetype/{id}

从更新JSON表示指定的问题类型。

请求

例

```
{
  "name": "name",
  "description": "description",
  "avatarId": 1
}
```

▶ 架构

回复

状态200 - 应用程序/ JSON 返回如果问题类型已成功更新。

▶ 架构

状态400 返回如果请求是无效的。这种情况发生时的名称是无效的，或者如果给定ID的头像不存在。

状态401 返回如果调用用户没有通过验证。

状态403 如果返回调用用户没有权限管理JIRA。

状态404 如果问题类型更新不存在，则返回。

状态409 如果已经存在具有指定名称的问题类型返回。

▼ 获取替代问题类型

GET /rest/api/2/issuetype/{id}/alternatives

返回所有替代问题类型对于给定的问题类型的ID列表。该列表将包含这些问题的类型，到分配给特定问题类型的问题，可以迁移。合适的替代品是被分配给相同的工作流，相同的字段结构，并且在同一屏幕方案的问题的类型。

回复

状态200 - 应用程序/ JSON 如果问题的类型存在且由主叫用户可见返回。

例

```
[
  {
    "self": "http://localhost:8090/jira/rest/api/2.0/issueType/3",
    "id": "3",
    "description": "A task that needs to be done.",
    "iconUrl": "http://localhost:8090/jira/images/icons/issuetypes/task.png",
    "name": "Task",
    "subtask": false,
    "avatarId": 1
  },
  {
    "self": "http://localhost:8090/jira/rest/api/2.0/issueType/1",
    "id": "1",
    "description": "A problem with the software.",
    "iconUrl": "http://localhost:8090/jira/images/icons/issuetypes/bug.png",
    "name": "Bug",
    "subtask": false,
    "avatarId": 10002
  }
]
```

▶ 架构

状态404 如果返回的问题类型不存在，或者不是给主叫用户可见。

▼ 创建临时的头像

POST /rest/api/2/issuetype/{id}/avatar

临时化身转换成一个真正的头像

请求

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "needsCropping": false
}
```

▶ 架构

回复

状态201 - 应用程序/ JSON 返回创建头像

例

```
{
  "id": "1000",
  "owner": "fred",
  "isSystemAvatar": true,
  "isSelected": false,
  "isDeletable": false,
  "urls": {
    "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
    "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
    "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
    "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10040"
  },
  "selected": false
}
```

▶ 架构

状态400 返回如果裁剪坐标无效

状态500 返回如果在临时化身转换为真正的化身出现错误

状态403 返回如果当前身份验证的用户没有权限来接头像

状态404 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 存储临时头像

POST /rest/api/2/issuetype/{id}/avatar/temporary

创建临时的头像。创建一个临时的头像是在上传新的头像一个问题类型的3步流程的一部分：上传，裁剪，确认。

下面的实施例示出了使用卷曲这三个步骤。饼干（会话）需要请求之间要保留，因此，使用**-b**和**-c**的。在步骤2中创建的ID需要传递到步骤3（可以简单地传递步骤2的作为步骤3的请求的整个响应）。

```
卷曲-c cookiejar.txt -X POST -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 图像/ PNG" --data二进制@ mynewavatar.png \
的 "http://本地主机: 8090 / JIRA / REST / API / 2 / 问题类型/ 1 /头像/临时文件名= mynewavatar.png"
```

```
卷曲-b cookiejar.txt -X POST -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 应用程序/ JSON" --data "{" cropperWidth": "65", "cropperOffsetX": "10", "cropperOffsetY": "16"}" \
-o tmpid.json \
HTTP: //本地主机: 8090 / JIRA / REST / API / 2 /问题类型/ 1 /头像
```

```
卷曲-b cookiejar.txt -X PUT -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 应用程序/ JSON" --data二进制@ tmpid.json \
HTTP: //本地主机: 8090 / JIRA / REST / API / 2 /问题类型/ 1 /头像
```

请求

查询参数

参数	类型	描述
filename	串	文件的名称被上传
size	长	文件大小

回复

状态201 - 应用程序/ JSON 临时头像裁剪说明

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态401 返回如果调用用户没有通过验证。

状态403 如果返回调用用户没有权限管理JIRA。

状态404 如果问题类型的更新不存在，或者如果请求不包含有效的XSRF令牌返回。

▼ 使用多部分存储临时头像

POST /rest/api/2/issuetype/{id}/avatar/temporary

创建使用多临时头像。响应被发送回存储在一个textarea JSON。这是因为客户端使用远程framing使用多提交化身。因此，我们必须从回该客户端从解析JSON给他们一个有效的HTML页面。

创建一个临时的头像是在上传新的头像一个问题类型的3步流程的一部分：上传，裁剪，确认。此端点允许您使用多部分上传的，而不是直接发送图像作为请求体。

你*必须*使用“阿凡达”作为上传参数的名称：

```
卷曲-c cookiejar.txt -X POST -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-F "avatar=@mynewavatar.png;类型=图像/ PNG" \
的 "http://本地主机: 8090 / JIRA / REST / API / 2 /问题类型/ 1 /头像/临时"
```

回复

状态201 - text / html 的临时头像裁剪指令内嵌在HTML页面。错误信息也将在页面中内嵌。

例

```
{
    "cropperWidth": 120,
    "cropperOffsetX": 50,
    "cropperOffsetY": 50,
    "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
    "needsCropping": true
}
```

▶ 架构

状态401 返回如果调用用户没有通过验证。

状态403 如果返回调用用户没有权限管理JIRA。

状态404 如果问题类型的更新不存在，或者如果请求不包含有效的XSRF令牌返回。

API / 2 / 问题类型/ {} issueTypeId / 属性 隐藏所有方法

这种资源可以存储自定义属性问题类型。

▼ 获取属性键

实验

GET /rest/api/2/issuetype/{issueTypeId}/properties

返回由id标识的问题类型的所有属性的钥匙。

回复

状态200 - 应用程序/ JSON 如果问题类型中发现返回。

例

```
{
    "keys": [
        {
            "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
            "key": "issue.support"
        }
    ]
}
```

▶ 架构

状态400 返回如果问题类型ID无效。

状态404 如果问题类型与给定的ID不存在，或者如果用户没有权限查看这个问题类型返回。

▼ 删除特性

实验

DELETE /rest/api/2/issuetype/{issueTypeId}/properties/{propertyKey}

移除由id标识的问题类型的属性。THS用户删除属性必须有权限修改问题类型。

回复

状态400 返回如果问题类型ID无效。

状态204 如果返回成功删除的问题类型属性。

状态404 返回如果问题类型与给定的ID不存在，或者给定键的属性是找不到的。

▼ 设置属性 实验

PUT /rest/api/2/issuetype/{issueTypeId}/properties/{propertyKey}

设置指定的问题类型的属性值。

你可以使用这个资源来存储对由ID标识的问题类型的自定义数据。谁存储数据的用户需要有权限修改的问题类型。

回复

状态200 返回如果问题类型属性被成功更新。

状态201 如果创建成功，问题类型属性返回。

状态400 返回如果问题类型ID无效。

状态404 如果问题类型与给定的ID不存在，或者如果用户没有权限编辑此问题类型返回。

▼ Get 属性 实验

GET /rest/api/2/issuetype/{issueTypeId}/properties/{propertyKey}

返回与由id标识的问题类型的给定键的属性的值。谁检索的属性的用户需要有权限查看的问题类型。

回复

状态200 - 应用程序/JSON 如果问题类型属性被发现返回。

例

```
{
  "key": "issue.support",
  "value": {
    "hipchat.room.id": "support-123",
    "support.time": "1m"
  }
}
```

▶ 架构

状态400 返回如果问题类型ID无效。

状态404 如果问题类型与给定的ID不存在，或者如果用户没有权限查看这个问题类型返回。

API / 2 / JQL / autocompletedata 隐藏所有方法

资源为搜索自动完成的数据。

▼ 获取自动完成

GET /rest/api/2/jql/autocompletedata

返回JQL搜索要求自动完成的数据。

回复

状态200 - 应用程序/JSON 需要JQL搜索自动完成的数据。

例

```
"(\\"visibleFieldNames\\": [(\\"value\\": \\"affectedVersion\\", \\"displayName\\": \\"affectedVersion\\", \\"auto\\": \\"true\\", \\"orderable\\": \\"true\\", \\"searchable\\": \\"true\\")]
```

▶ 架构

状态401 返回如果调用用户没有通过验证。

状态500 返回如果在生成响应时发生错误。

▼ 获取现场自动完成的查询字符串

GET /rest/api/2/jql/autocompletedata/suggestions

返回JQL搜索自动完成建议。

请求

查询参数

参数	类型	描述
fieldName	串	在生成了建议的字段名称。

fieldValue	串	这已经由用户提供的字段值的部分。
predicateName	串	在生成了建议谓词。“来自”，“通过”和“到”：建议生成只对。
predicateValue	串	这已经由用户提供的谓词值的部分。

回复

状态200 - 应用程序/JSON 为JQL搜索自动完成建议。

例

```
{
  "results": [
    {
      "value": "ActiveObjects",
      "displayName": "<b>Ac</b>tiveObjects (A0)"
    },
    {
      "value": "Atlassian Connect",
      "displayName": "Atlassian Connect (<b>AC</b>)"
    },
    {
      "value": "Atlassian Connect in JIRA",
      "displayName": "Atlassian Connect in JIRA (<b>AC</b>JIRA)"
    }
  ]
}
```

▶ 架构

API / 2 / licenseValidator 隐藏所有方法

一个REST端点的JIRA许可证提供简单的验证服务。通常使用的JIRA应用程序的安装阶段。这将返回一个对象有错误的键，值对的列表。

... 显示更多

▼ 验证

POST /rest/api/2/licenseValidator

请求

▶ 架构

回复

状态 - 应用程序/JSON

API / 2 / mypreferences 隐藏所有方法

提供了在当前登录用户的喜好。

▼ 获取偏好

GET /rest/api/2/mypreferences

当前登录的用户的回报偏好。选项键必须作为输入参数（密钥）来提供。该值返回正是因为它是。如果没有提供或错误关键参数 - 状态码404。如果价值发现 - 状态代码200。

请求

查询参数

参数	类型	描述
key	串	- 偏好的关键返回。

回复

状态 - 应用程序/JSON

▼ 套装优惠

PUT /rest/api/2/mypreferences

设置在当前登录用户的偏好。选项键必须作为输入参数（密钥）来提供。值必须作为后身体提供。如果没有提供密钥或值参数 - 状态代码404。如果首选项设置 - 状态代码204。

请求

查询参数

参数	类型	描述
key	串	- 优先的键来设定。

▶ 架构

回复

状态 - 应用程序/JSON

▼ 删除偏好

DELETE /rest/api/2/mypreferences

删除在当前登录用户的偏好。选项键必须作为输入参数（密钥）来提供。如果偏好是未设置状态码404 -- 状态码204如果关键参数不提供或错误。

请求

查询参数

参数	类型	描述
key	串	- 优先的键被除去。

回复

状态 - 应用程序/JSON

API / 2 / 我 隐藏所有方法

当前登录用户资源

▼ 获取用户

GET /rest/api/2/myself

返回当前登录的用户。此资源不能被匿名访问。

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
  "name": "fred",
  "emailAddress": "fred@example.com",
  "avatarUrls": {
    "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
    "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
    "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
    "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
  },
  "displayName": "Fred F. User",
  "active": true,
  "timeZone": "Australia/Sydney",
  "groups": {
    "size": 3,
    "items": [
      {
        "name": "jira-user",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-user"
      },
      {
        "name": "jira-admin",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-admin"
      },
      {
        "name": "important",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=important"
      }
    ]
  },
  "applicationRoles": {
    "size": 1,
    "items": []
  },
  "expand": "groups,applicationRoles"
}
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

▼ 更新用户

PUT /rest/api/2/myself

修改当前登录的用户。目前“价值”字段将覆盖现有的值。字段跳过请求将不会被改变。只有电子邮件地址和显示名称可以改变这种方式。需要用户密码。

请求

例

```
{
  "password": "abracadabra",
  "emailAddress": "eddie@atlassian.com",
  "displayName": "Eddie of Atlassian"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 如果用户存在，并且调用者有权编辑它返回。

例

```
{
  "self": "http://www.example.com/jirahttp://www.example.com/jira/rest/api/2/user/charlie",
  "key": "charlie",
  "name": "eddie",
  "emailAddress": "eddie@atlassian.com",
  "displayName": "Eddie of Atlassian"
}
```

▶ 架构

状态400 如果返回请求无效，包括不正确的密码。

状态401 返回，如果用户不被认证。

状态403 如果目录是只读的返回。

状态404 返回如果用户无法找到。

▼ 更改我的密码

PUT /rest/api/2/myself/password

来电显示修改密码。

请求

例

```
{
  "password": "new password",
  "currentPassword": "current password"
}
```

▶ 架构

回复

状态400 如果新的密码不正确/丢失或当前密码不正确返回。

状态401 返回，如果用户不被认证。

状态204 如果用户存在，并且调用者有权编辑它返回。

状态403 如果目录是只读的返回。

状态404 返回如果调用者有权更改用户密码，但用户不存在。

API / 2 / notificationscheme

隐藏所有方法

▼ 获取通知的计划

GET /rest/api/2/notificationscheme

返回分页通知计划表。为了访问通知方案，需要主叫用户有权限来管理与所请求的通知方案相关联的至少一个项目。每个方案包含配置为接收通知，这些事件的事件和收件人列表。消费者应允许在没有收件人的事件出现在响应。该名单是由该计划的名称排序。遵循/ notificationscheme / {ID} 资源的文档，了解有关返回值的所有细节。

请求

查询参数

参数	类型	描述
startAt	长	第一个通知计划的指标返回（基于0）。
maxResults	INT	通知方案的最大数目，返回（最多50个）。
expand	串	

回复

状态200 - 应用程序/JSON 该用户有权限的通知计划的分页列表。

例

```
{
  "maxResults": 6,
  "startAt": 1,
  "total": 5,
  "isLast": false,
  "values": [
    {
      "expand": "notificationSchemeEvents,user,group,projectRole,field,all",
      "id": 10100,
      "self": "http://example.com/jira/rest/api/2/notificationscheme/10100",
      "name": "notification scheme name",
      "description": "description",
      "notificationSchemeEvents": [
        {
          "event": {
            "id": 1,
            "name": "Issue created",
            "description": "Event published when issue is created"
          },
          "notifications": [
            {
              "id": 1,
              "notificationType": "Group",
              "parameter": "jira-administrators",
              "group": {
                "name": "jira-administrators",
                "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-administrators"
              },
              "expand": "group"
            },
            {
              "id": 2,
              "notificationType": "CurrentAssignee"
            },
            {
              "id": 3,
              "notificationType": "ProjectRole",
              "parameter": "10360",
              "projectRole": {
                "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
                "name": "Developers",
                "id": 10360
              }
            }
          ]
        }
      ]
    }
  ]
}
```

▶ 架构

获取计划的通知

GET /rest/api/2/notificationscheme/{id}

返回给定ID的通知计划作充分的代表性。这一资源将返回包含配置为接收通知，这些事件的事件和收件人列表的通知计划。消费者应允许在没有收件人的事件出现在响应。用户访问数据需要具有的权限来管理与所请求的通知方案相关联的至少一个项目。

通知收件人可以是：

当前的受让人 - 由NotificationType值为CurrentAssignee

记者的问题 - 由NotificationType的值是记者

当前用户 - 由NotificationType的值的currentUser

项目负责人 - 由NotificationType值为ProjectLead

元件引线 - 由NotificationType值为ComponentLead

所有观察家 - 通知类型的值是AllWatchers

配置的用户-通知类型的值是用户。参数将包含用户的密钥。如果将提供有关用户的信息的用户使用扩展参数。

配置的组-通知类型的值是组。参数将包含组的名称。如果将提供关于集团的资料组用于扩大参数。

配置的电子邮件地址 - 通知类型的值是EmailAddress的，有关电子邮件的附加信息将被提供。

用户或用户在配置的自定义字段组-通知类型的值是UserCustomField或GroupCustomField。参数将包含自定义字段的ID。如果将提供有关字段的信息字段被用来扩展参数。

配置项目角色-通知类型的值是ProjectRole。参数将包含项目中的角色ID。如果将提供有关项目的角色信息projectRole参数用于扩展。

请参阅参考的例子。

事件可以是JIRA系统事件或管理员配置的事件。在该系统中的事件的情况下，提供了有关他们的ID，名称和说明数据。在自定义事件的情况下，包括在模板事件也是如此。

请求

查询参数

参数	类型	描述

expand

串

回复

状态200 - 应用程序/JSON 如果通知方式存在，并且是可见的主叫用户返回

例

```
{
  "expand": "notificationSchemeEvents, user, group, projectRole, field, all",
  "id": 10100,
  "self": "http://example.com/jira/rest/api/2/notificationscheme/10010",
  "name": "notification scheme name",
  "description": "description",
  "notificationSchemeEvents": [
    {
      "event": [
        {
          "id": 1,
          "name": "Issue created",
          "description": "Event published when issue is created"
        }
      ],
      "notifications": [
        {
          "id": 1,
          "notificationType": "Group",
          "parameter": "jira-administrators",
          "group": [
            {
              "name": "jira-administrators",
              "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-administrators"
            },
            "expand": "group"
          ],
          {
            "id": 2,
            "notificationType": "CurrentAssignee"
          },
          {
            "id": 3,
            "notificationType": "ProjectRole",
            "parameter": "10360",
            "projectRole": [
              {
                "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
                "name": "Developers",
                "id": 10360,
                "description": "A project role that represents developers in a project",
                "actors": [
                  {
                    "id": 10240,
                    "displayName": "jira-developers",
                    "type": "atlassian-group-role-actor"
                  }
                ]
              }
            ]
          }
        ]
      ]
    }
  ]
}
```

▶ 架构

状态404 返回如果通知方案不存在，或者是不向主叫用户可见

API / 2 / 密码 隐藏所有方法

相关密码和密码策略操作REST资源。

▼ 获取密码策略

GET /rest/api/2/password/policy

返回当前密码策略的要求清单。例如，“密码必须包含至少10个字符”，“密码不能类似于用户的姓名或电子邮件地址。”等等。

请求

查询参数

参数	类型	描述
hasOldPassword	布尔	用户是否将被要求输入其当前密码。使用@code {假} (默认值)，如果这是一个新的用户或管理员强制更改其他用户的密码。 默认值： 假

回复

状态200 - 应用程序/JSON 返回的消息字符串数组。

▶ 架构

▼ 策略检查创建用户

POST /rest/api/2/password/policy/createUser

返回声明解释为什么密码政策会为新用户不允许提议口令列表。

您可以使用此方法来测试密码策略验证。这可能会在那里一个新的用户和相关密码创建一个动作之前完成，使用像那些方法 [UserService](#)。例如，你可以使用它来验证在用户界面中创建用户形式的密码，用户输入它。

用户名和新密码必须不为空进行验证。

请注意，这种方法会帮助你验证对只有政策。它不会检查可能创建新用户时，例如，检查具有相同的名称的用户是否已经存在执行任何其它的验证。

请求

例

```
{
  "username": "fred",
  "displayName": "Fred Normal",
  "emailAddress": "fred@example.com",
  "password": "secret"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回的消息字符串数组。

▶ 架构

状态400 如果返回请求是无效的，比如如果用户名或密码不指定。

▼ 策略检查更新用户

POST /rest/api/2/password/policy/updateUser

返回声明解释为什么密码政策将与现有密码的用户不允许拟议的新密码的列表。

您可以使用此方法来测试密码策略验证。这可能一个动作，其中的密码实际上是更新之前完成，使用类似方法的[ChangePassword](#)或[ResetPassword](#)。例如，你可以使用它来验证在用户界面中更改密码形式的密码，如用户输入它。

用户必须存在，并且用户名和新密码必须不为空，进行验证。

请注意，这方法将帮助您验证只针对政策。它不会检查可能提交密码更改/复位请求时，例如验证旧密码是否有效执行任何其它的验证。

请求

例

```
{
  "username": "fred",
  "oldPassword": "secret",
  "newPassword": "correcthorsebatterystaple"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回的消息字符串数组。

▶ 架构

状态400 如果返回请求是无效的，例如，如果用户名或新密码未指定。

状态404 如果用户名不对应于任何现有的用户返回。

API / 2 / permissionscheme

[隐藏所有方法](#)

资源管理权限的计划。

[... 显示更多](#)

▼ 得到许可制度

GET /rest/api/2/permissionscheme

返回所有许可方案的列表。

默认情况下只返回缩短豆。如果要包括所有计划的权限，然后指定权限扩大参数。权限将如果指定任何其他扩展参数也包括在内。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 如果返回成功。

例

```
{
  "permissionSchemes": [
    {
      "id": 10000,
      "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
      "name": "Example permission scheme",
      "description": "description"
    }
  ]
}
```

▶ 架构

状态401 如果返回的用户没有登录。**状态403** 如果返回的用户无权查看的权限方案。

▼ 创建权限方案

POST /rest/api/2/permissionscheme

创建一个新的许可方案。这种方法可以创建方案具有定义权限集，或没有。

请求

查询参数

参数	类型	描述
expand	串	

例

```
{
  "name": "Example permission scheme",
  "description": "description",
  "permissions": [
    {
      "holder": [
        {
          "type": "group",
          "parameter": "jira-developers"
        }
      ],
      "permission": "ADMINISTER_PROJECTS"
    }
  ]
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果成功创建计划返回。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
  "name": "Example permission scheme",
  "description": "description",
  "permissions": [
    {
      "id": 10000,
      "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
      "holder": [
        {
          "type": "group",
          "parameter": "jira-developers",
          "expand": "group"
        }
      ],
      "permission": "ADMINISTER_PROJECTS"
    }
  ]
}
```

▶ 架构

状态401 如果用户没有登录返回。**状态403** 返回如果用户不允许创建许可计划。

▼ 得到许可计划

GET /rest/api/2/permissionscheme/{schemeId}

返回由指定ID标识的许可方案。

请求

查询参数

参数	类型	描述

expand

串

回复

状态200 - 应用程序/ JSON 如果返回成功。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
  "name": "Example permission scheme",
  "description": "description",
  "permissions": [
    {
      "id": 10000,
      "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
      "holder": {
        "type": "group",
        "parameter": "jira-developers",
        "expand": "group"
      },
      "permission": "ADMINISTER_PROJECTS"
    }
  ]
}
```

▶ 架构

状态401 如果返回的用户没有登录。

状态403 如果返回的用户无权查看的权限方案。

状态404 如果该方案不存在，则返回。

▼ 删除权限方案

DELETE /rest/api/2/permissionscheme/{schemeId}

删除由指定ID标识的许可方案。

回复

状态401 如果返回的用户没有登录。

状态204 - 应用程序/ JSON 返回如果权限方案成功删除。

状态403 如果返回的用户是不允许删除权限方案。

▼ 更新许可计划

PUT /rest/api/2/permissionscheme/{schemeId}

更新许可方案。

如果权限列表存在，那么它会在权限方案，这基本上意味着它会覆盖在权限模式存在的任何权限授予设置。发送一个空列表将删除权限方案中的所有权限授予。

要更新只是名称和描述，根本不发送权限列表。

要添加或删除单个权限授予而不是更新整个列表一次使用的 **schemaId** /许可/资源。

请求

查询参数

参数	类型	描述
expand	串	

例

```
{
  "name": "Example permission scheme",
  "description": "description",
  "permissions": [
    {
      "holder": {
        "type": "group",
        "parameter": "jira-developers"
      },
      "permission": "ADMINISTER_PROJECTS"
    }
  ]
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回如果方案成功更新。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
  "name": "Example permission scheme",
  "description": "description",
  "permissions": [
    {
      "id": 10000,
      "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
      "holder": [
        {
          "type": "group",
          "parameter": "jira-developers",
          "expand": "group"
        }
      ],
      "permission": "ADMINISTER_PROJECTS"
    }
  ]
}
```

▶ 架构

状态401 如果用户没有登录返回。

状态403 返回如果用户不允许编辑许可计划。

状态404 如果找不到返回的权限。

▼ 得到许可计划拨款

GET /rest/api/2/permissionscheme/{schemeId}/permission

返回给定的许可方案的所有权限授予。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 如果返回成功。

例

```
{
  "permissions": [
    {
      "id": 10000,
      "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
      "holder": [
        {
          "type": "group",
          "parameter": "jira-developers",
          "expand": "group"
        }
      ],
      "permission": "ADMINISTER_PROJECTS"
    }
  ],
  "expand": "user, group, projectRole, field, all"
}
```

▶ 架构

状态401 如果返回的用户没有登录。

状态403 如果返回的用户无权查看的权限方案。

▼ 创建权限授予

POST /rest/api/2/permissionscheme/{schemeId}/permission

在创建许可方案的权限授予。

请求

查询参数

参数	类型	描述
expand	串	

例

```
{
  "holder": {
    "type": "group",
    "parameter": "jira-developers"
  },
  "permission": "ADMINISTER_PROJECTS"
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果成功创建该计划允许返回。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
  "holder": {
    "type": "group",
    "parameter": "jira-developers",
    "expand": "group"
  },
  "permission": "ADMINISTER_PROJECTS"
}
```

▶ 架构

状态401 如果用户没有登录返回。

状态403 返回如果用户不允许编辑许可计划。

▼ 删除权限方案实体

DELETE /rest/api/2/permissionscheme/{schemeId}/permission/{permissionId}

删除从许可方案的权限授予。

回复

状态401 如果用户没有登录返回。

状态204 - 应用程序/JSON 返回如果权限授予被成功删除。

状态403 返回如果用户不允许编辑许可计划。

▼ 得到许可计划津贴

GET /rest/api/2/permissionscheme/{schemeId}/permission/{permissionId}

返回由指定ID标识的权限授予。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 如果返回成功。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/permission/10000",
  "holder": {
    "type": "group",
    "parameter": "jira-developers",
    "expand": "group"
  },
  "permission": "ADMINISTER_PROJECTS"
}
```

▶ 架构

状态401 如果返回的用户没有登录。

状态403 如果返回的用户无权查看的权限方案。

▼ 获得优先

GET /rest/api/2/priority

返回所有问题的优先级列表。

回复

状态200 - 应用程序/JSON 如果优先级存在，并且用户有权限查看返回。包含JSON格式的优先充分的代表性。

例

```
[  
  {  
    "self": "http://www.example.com/jira/rest/api/2/priority/3",  
    "statusColor": "#009900",  
    "description": "Major loss of function.",  
    "iconUrl": "http://www.example.com/jira/images/icons/priorities/major.png",  
    "name": "Major"  
  },  
  {  
    "self": "http://www.example.com/jira/rest/api/2/priority/5",  
    "statusColor": "#cfcfcf",  
    "description": "Very little impact.",  
    "iconUrl": "http://www.example.com/jira/images/icons/priorities/trivial.png",  
    "name": "Trivial"  
  }  
]
```

▶ 架构

▼ 获得优先权

GET /rest/api/2/priority/{id}

返回问题的优先级。

回复

状态200 - 应用程序/JSON 如果优先级存在且由主叫用户可见返回。包含在JSON问题优先充分的代表性。

例

```
{  
  "self": "http://www.example.com/jira/rest/api/2/priority/3",  
  "statusColor": "#009900",  
  "description": "Major loss of function.",  
  "iconUrl": "http://www.example.com/jira/images/icons/priorities/major.png",  
  "name": "Major"  
}
```

▶ 架构

状态404 如果优先级不存在或不是主叫用户可见返回。

API / 2 / 项目 隐藏所有方法

▼ 获取所有项目

GET /rest/api/2/project

返回该可见在当前登录用户的项目。如果没有用户登录，它返回的使用匿名访问时可见的项目列表。

请求

查询参数

参数	类型	描述
expand	串	参数扩大
recent	INT	如果该参数设置则仅最近访问的项目由当前用户（如果没有则登录基于HTTP会话）将返回（限指定数量最大计数，但不超过20个）。

回复

状态200 - 应用程序/JSON 返回为其用户有浏览、管理或PROJECT_ADMIN项目许可的项目清单。

例

```
[  
  {  
    "self": "http://www.example.com/jira/rest/api/2/project/EX",  
    "id": "10000",  
    "key": "EX",  
    "name": "Example",  
    "avatarUrls": {  
      "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10000",  
      "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10000",  
      "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10000",  
      "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10000"  
    },  
    "projectCategory": {  
      "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",  
      "id": "10000",  
      "name": "FIRST",  
      "description": "First Project Category"  
    }  
  },  
  {  
    "self": "http://www.example.com/jira/rest/api/2/project/ABC",  
    "id": "10001",  
    "key": "ABC",  
    "name": "Alphabetical",  
    "avatarUrls": {  
      "48x48": "http://www.example.com/jira/secure/projectavatar?size=large&pid=10001",  
      "24x24": "http://www.example.com/jira/secure/projectavatar?size=small&pid=10001",  
      "16x16": "http://www.example.com/jira/secure/projectavatar?size=xsmall&pid=10001",  
      "32x32": "http://www.example.com/jira/secure/projectavatar?size=medium&pid=10001"  
    },  
    "projectCategory": {  
      "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",  
      "id": "10000",  
      "name": "FIRST",  
      "description": "First Project Category"  
    }  
  }  
]
```

▶ 架构

▼ 创建项目

POST /rest/api/2/project

创建一个新的项目。

请求

例

```
{  
  "key": "EX",  
  "name": "Example",  
  "projectTypeKey": "business",  
  "projectTemplateKey": "com.atlassian.jira-core-project-templates:jira-core-project-management",  
  "description": "Example Project description",  
  "lead": "Charlie",  
  "url": "http://atlassian.com",  
  "assigneeType": "PROJECT_LEAD",  
  "avatarId": 10200,  
  "issueSecurityScheme": 10001,  
  "permissionScheme": 10011,  
  "notificationScheme": 10021,  
  "categoryId": 10120  
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回如果项目创建成功。

例

```
{  
  "self": "http://example/jira/rest/api/2/project/10042",  
  "id": 10042,  
  "key": "EX"  
}
```

▶ 架构

状态400 如果该请求是无效的，不能创建的项目返回。

状态401 返回如果用户没有登录。

状态403 返回如果用户没有权限创建项目。

▼ 获取项目

GET /rest/api/2/project/{projectIdOrKey}

包含JSON格式的项目的完整表示。

与该项目相关的所有项目将钥匙只有在返回expand=projectKeys。

请求

查询参数

参数	类型	描述
expand	串	参数扩大

回复

状态200 - 应用程序/JSON 如果项目存在，并且用户有权限查看返回。包含JSON格式的项目的完整表示。

例

```
{
  "expand": "description,lead,url,projectKeys",
  "self": "http://www.example.com/jira/rest/api/2/project/EX",
  "id": "10000",
  "key": "EX",
  "description": "This project was created as an example for REST.",
  "lead": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "components": [
    {
      "self": "http://www.example.com/jira/rest/api/2/component/10000",
      "id": "10000",
      "name": "Component_1",
      "description": "This is a JIRA component",
      "lead": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
          "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
          "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
          "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
      },
      "assigneeType": "PROJECT_LEAD",
      "assignee": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred"
        }
      }
    }
  ]
}
```

▶ 架构

状态404 返回如果项目没有找到，或者调用用户没有权限查看。

▼ 更新项目

PUT /rest/api/2/project/{projectIdOrKey}

更新项目。

只有在JSON发送的非空值将在项目进行更新。

可用于**assigneeType**值是：“PROJECT_LEAD”和“未分配”。

请求

查询参数

参数	类型	描述
expand	串	这些参数在返回的项目扩大

例

```
{
  "key": "EX",
  "name": "Example",
  "projectTypeKey": "business",
  "projectTemplateKey": "com.atlassian.jira-core-project-templates:jira-core-project-management",
  "description": "Example Project description",
  "lead": "Charlie",
  "url": "http://atlassian.com",
  "assigneeType": "PROJECT_LEAD",
  "avatarId": 10200,
  "issueSecurityScheme": 10001,
  "permissionScheme": 10011,
  "notificationScheme": 10021,
  "categoryId": 10120
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回如果项目更新成功。

例

```
{
  "expand": "description,lead,url,projectKeys",
  "self": "http://www.example.com/jira/rest/api/2/project/EX",
  "id": "10000",
  "key": "EX",
  "description": "This project was created as an example for REST.",
  "lead": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": [
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    ],
    "displayName": "Fred F. User",
    "active": false
  },
  "components": [
    {
      "self": "http://www.example.com/jira/rest/api/2/component/10000",
      "id": "10000",
      "name": "Component 1",
      "description": "This is a JIRA component",
      "lead": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": [
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
          "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
          "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
          "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        ],
        "displayName": "Fred F. User",
        "active": false
      },
      "assigneeType": "PROJECT_LEAD",
      "assignee": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": [
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred"
        ]
      }
    }
  ]
}
```

▶ 架构

状态400 返回如果请求无效，该项目无法更新。

状态401 返回如果用户没有登录。

状态403 返回，如果用户不具有权限更新项目。

状态404 如果该项目不存在，则返回。

▼ 删除工程

DELETE /rest/api/2/project/{projectIdOrKey}

删除的项目。

回复

状态401 返回如果用户没有登录。

状态204 - 应用程序/JSON 返回如果项目成功删除。

状态403 返回如果当前身份验证的用户没有权限删除该项目。

状态404 如果该项目不存在，则返回。

▼ 创建临时的头像

POST /rest/api/2/project/{projectIdOrKey}/avatar

临时化身转换成一个真正的头像

请求

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "needsCropping": false
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回创建头像

例

```
{
  "id": "1000",
  "owner": "fred",
  "isSystemAvatar": true,
  "isSelected": false,
  "isDeletable": false,
  "urls": {
    "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
    "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
    "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
    "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10040"
  },
  "selected": false
}
```

▶ 架构

状态400 返回如果裁剪坐标无效

状态403 返回如果当前身份验证的用户没有权限来接头像

状态404 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 更新项目的头像

PUT /rest/api/2/project/{projectIdOrKey}/avatar

请求

▶ 架构

回复

状态 - 应用程序/JSON

▼ 删除头像

DELETE /rest/api/2/project/{projectIdOrKey}/avatar/{id}

删除头像

回复

状态204 - 应用程序/JSON 如果返回的头像被成功删除。

状态403 返回如果当前身份验证的用户没有权限删除组件。

状态404 如果分身不存在，或当前身份验证的用户没有权限删除它返回。

▼ 存储临时头像

POST /rest/api/2/project/{projectIdOrKey}/avatar/temporary

创建临时头像

请求

查询参数

参数	类型	描述
filename	串	文件的名称被上传
size	长	文件大小

回复

状态201 - 应用程序/JSON 临时头像裁剪说明

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态400 验证失败。例如文件大小超出最大附件大小。

状态403 如果请求不包含有效XSRF令牌返回

状态404 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 使用多部分存储临时头像

POST /rest/api/2/project/{projectIdOrKey}/avatar/temporary

创建使用多临时头像。响应被发送回存储在一个textarea JSON。这是因为客户端使用远程framing使用多提交化身。因此，我们必须给他们一个有效的HTML页面返回从客户端解析JSON。

回复

状态201 - text / html 的临时头像裁剪指令内嵌在HTML页面。错误信息也将再页面中内嵌。

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态404 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 获取所有的化身

GET /rest/api/2/project/{projectIdOrKey}/avatars

返回该可见在当前登录用户的所有化身。化身分为系统和自定义。

回复

状态200 - 应用程序/JSON 返回包含替身的列表为定制的系统替身，其中用户具有浏览项目的权限的地图。

例

```
{
  "system": [
    {
      "id": "1000",
      "owner": "fred",
      "isSystemAvatar": true,
      "isSelected": false,
      "isDeletable": false,
      "urls": {
        "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
        "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
        "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
        "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10040"
      },
      "selected": false
    }
  ],
  "custom": [
    {
      "id": "1010",
      "owner": "andrew",
      "isSystemAvatar": false,
      "isSelected": false,
      "isDeletable": true,
      "urls": {
        "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10080",
        "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10080",
        "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10080",
        "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10080"
      },
      "selected": false
    }
  ]
}
```

▶ 架构

状态404 如果返回目前已验证的用户没有查看项目的权限。

▼ 获取项目的组成部分

GET /rest/api/2/project/{projectIdOrKey}/components

包含一个指定的项目的组件的完整表示。

回复

状态200 - 应用程序/JSON 如果项目存在，并且用户有权查看其组件返回。包含JSON格式的项目的组件的完整表示。

例

```
[ {
    "self": "http://www.example.com/jira/rest/api/2/component/10000",
    "id": "10000",
    "name": "Component 1",
    "description": "This is a JIRA component",
    "lead": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    },
    "assigneeType": "PROJECT_LEAD",
    "assignee": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
    },
    "realAssigneeType": "PROJECT_LEAD",
    "realAssignee": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
            "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
            "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
            "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
            "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": true
    }
}]
```

▶ 架构

状态404 返回如果项目没有找到，或者调用用户没有权限查看。

▼ 获取所有状态

GET /rest/api/2/project/{projectIdOrKey}/statuses

获取所有的问题类型的有效状态值项目

回复

状态200 - 应用程序/JSON 如果项目存在，并且用户有权查看其组件返回。包含的问题类型与有效期为每个问题类型的状态值完整表示。

例

```
[ {
    "self": "http://localhost:8090/jira/rest/api/2.0/issueType/3",
    "id": "3",
    "name": "Task",
    "subtask": false,
    "statuses": [
        {
            "self": "http://localhost:8090/jira/rest/api/2.0/status/10000",
            "description": "The issue is currently being worked on.",
            "iconUrl": "http://localhost:8090/jira/images/icons/progress.gif",
            "name": "In Progress",
            "id": "10000"
        },
        {
            "self": "http://localhost:8090/jira/rest/api/2.0/status/5",
            "description": "The issue is closed.",
            "iconUrl": "http://localhost:8090/jira/images/icons/closed.gif",
            "name": "Closed",
            "id": "5"
        }
    ]
}]
```

▶ 架构

状态400 返回如果项目没有找到，或者调用用户没有权限查看。

▼ 更新项目类型

PUT /rest/api/2/project/{projectIdOrKey}/type/{newProjectTypeKey}

更新的一个项目的类型。

回复

状态200 - 应用程序/JSON 返回如果更新的项目类型是成功的。

例

```
{
  "expand": "description, lead, url, projectKeys",
  "self": "http://www.example.com/jira/rest/api/2/project/EX",
  "id": "10000",
  "key": "EX",
  "description": "This project was created as an example for REST.",
  "lead": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  "components": [
    {
      "self": "http://www.example.com/jira/rest/api/2/component/10000",
      "id": "10000",
      "name": "Component 1",
      "description": "This is a JIRA component",
      "lead": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
          "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
          "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
          "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
        },
        "displayName": "Fred F. User",
        "active": false
      },
      "assigneeType": "PROJECT_LEAD",
      "assignee": {
        "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
        "name": "fred",
        "avatarUrls": {
          "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred"
        }
      }
    }
  ]
}
```

▶ 架构

- 状态400** 如果请求无效，项目类型无法更新返回。
- 状态401** 返回如果用户没有登录。
- 状态403** 返回，如果用户不具有权限更新项目。
- 状态404** 如果该项目不存在，则返回。

▼ 获取项目版本分页

GET /rest/api/2/project/{projectIdOrKey}/version

返回所有版本的指定项目。结果分页。

结果可以通过以下字段可以订购：

序列
名称
开始日期
发布日期

请求

查询参数

参数	类型	描述
startAt	长	页面偏移，如果不指定，则默认为0
maxResults	INT	在页面上的许多结果应被包括在内。默认为50。
orderBy	串	结果的排序。
expand	串	参数扩大

回复

- 状态200** - 应用程序/JSON 第一个版本的项目。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/project/PR/version?startAt=0&maxResults=2",
  "nextPage": "http://www.example.com/jira/rest/api/2/project/PR/version?startAt=2&maxResults=2",
  "maxResults": 2,
  "startAt": 0,
  "total": 7,
  "isLast": false,
  "values": [
    {
      "self": "http://www.example.com/jira/rest/api/2/version/10000",
      "id": "10000",
      "description": "An excellent version",
      "name": "New Version 1",
      "archived": false,
      "released": true,
      "releaseDate": "2010-07-06",
      "overdue": true,
      "userReleaseDate": "6/Jul/2010",
      "projectId": 10000
    },
    {
      "self": "http://www.example.com/jira/rest/api/2/version/10010",
      "id": "10010",
      "description": "Minor Bugfix version",
      "name": "Next Version",
      "archived": false,
      "released": false,
      "overdue": false,
      "projectId": 10000
    }
  ]
}
```

▶ 架构

状态404 返回如果项目没有找到，或者调用用户没有权限查看。

▼ 获取项目版本

GET /rest/api/2/project/{projectIdOrKey}/versions

包含一个指定的项目版本的完整表示。

请求

查询参数

参数	类型	描述
expand	串	参数扩大

回复

状态200 - 应用程序/JSON 如果项目存在，并且用户有权查看其版本返回。包含JSON格式的项目版本的完整表示。

例

```
[
  {
    "self": "http://www.example.com/jira/rest/api/2/version/10000",
    "id": "10000",
    "description": "An excellent version",
    "name": "New Version 1",
    "archived": false,
    "released": true,
    "releaseDate": "2010-07-06",
    "overdue": true,
    "userReleaseDate": "6/Jul/2010",
    "projectId": 10000
  },
  {
    "self": "http://www.example.com/jira/rest/api/2/version/10010",
    "id": "10010",
    "description": "Minor Bugfix version",
    "name": "Next Version",
    "archived": false,
    "released": false,
    "overdue": false,
    "projectId": 10000
  }
]
```

▶ 架构

状态404 返回如果项目没有找到，或者调用用户没有权限查看。

API / 2 / 项目 / {projectIdOrKey} / 属性 [隐藏所有方法](#)▼ 获取属性键 [实验](#)

GET /rest/api/2/project/{projectIdOrKey}/properties

返回由键或用id标识项目的所有属性的钥匙。

回复

状态200 - 应用程序/JSON 如果该项目被发现返回。

例

```
{
  "keys": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
      "key": "issue.support"
    }
  ]
}
```

▶ 架构

状态400 返回如果项目键或ID无效。

状态401 返回如果调用用户没有通过验证。

状态403 返回如果主叫用户不具有权限浏览的项目。

状态404 如果返回与给定的密钥或ID的项目不存在，或者给定键的属性是找不到的。

▼ 删除特性 实验

DELETE /rest/api/2/project/{projectIdOrKey}/properties/{propertyKey}

移除按键式或由id标识的项目属性。THS用户删除属性是需要有权限来管理该项目。

回复

状态400 返回如果项目键或ID无效。

状态401 返回如果调用用户没有通过验证。

状态204 如果返回成功删除项目属性。

状态403 如果返回调用用户没有权限编辑该项目。

状态404 如果返回与给定的密钥或ID的项目不存在，或者给定键的属性是找不到的。

▼ 设置属性 实验

PUT /rest/api/2/project/{projectIdOrKey}/properties/{propertyKey}

设置指定项目的属性的值。

你可以使用这个资源来存储反对键或用id标识项目中的自定义数据。谁存储数据的用户需要具有的权限来管理项目。

回复

状态200 返回如果项目属性成功更新。

状态201 如果成功创建项目属性返回。

状态400 返回如果项目键或ID无效。

状态401 返回如果调用用户没有通过验证。

状态403 返回如果主叫用户不具有权限管理项目。

状态404 如果与给定的密钥或ID的项目不存在，则返回。

▼ Get属性 实验

GET /rest/api/2/project/{projectIdOrKey}/properties/{propertyKey}

返回与从由键或由id标识的项目的给定键的属性的值。谁检索属性的用户需要具有的权限来读取该项目。

回复

状态200 - 应用程序/JSON 如果项目财产被发现返回。

例

```
{
  "key": "issue.support",
  "value": {
    "hipchat.room.id": "support-123",
    "support.time": "1m"
  }
}
```

▶ 架构

- 状态400 返回如果项目键或ID无效。
- 状态401 返回如果调用用户没有通过验证。
- 状态403 返回如果主叫用户不具有权限浏览的项目。
- 状态404 如果返回与给定的密钥或ID的项目不存在，或者给定键的属性是找不到的。

API / 2 / 项目 / {} projectIdOrKey / 角色 隐藏所有方法

▼ 获取项目角色

GET /rest/api/2/project/{projectIdOrKey}/role

返回给定项目的ID或键的所有角色，链接到每个角色的全部细节。

回复

- 状态200 - 应用程序/JSON 返回地图的角色包含每个角色的全部细节的URI一个。该项目必须存在，并且用户必须有权限查看。

例

```
{
  "Administrators": "http://www.example.com/jira/rest/api/2/project/MKY/role/10002",
  "Users": "http://www.example.com/jira/rest/api/2/project/MKY/role/10001",
  "Developers": "http://www.example.com/jira/rest/api/2/project/MKY/role/10000"
}
```

▶ 架构

- 状态404 返回如果项目没有找到，或者调用用户没有权限查看。

▼ 获取项目角色

GET /rest/api/2/project/{projectIdOrKey}/role/{id}

返回在一个项目中某一项目的角色的细节。

回复

- 状态200 - 应用程序/JSON 返回角色名称，描述和项目参与者所请求的角色（用户和组）。该项目必须存在，并且用户必须有权限查看。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project",
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

▶ 架构

- 状态404 返回如果项目或角色没有找到，或者调用用户没有权限查看。

▼ 集演员

PUT /rest/api/2/project/{projectIdOrKey}/role/{id}

更新项目角色，包括指定的参与者（用户或组）。

请求

例

```
{
  "id": 10360,
  "categorisedActors": {
    "atlassian-user-role-actor": [
      "admin"
    ],
    "atlassian-group-role-actor": [
      "jira-developers"
    ]
  }
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回角色名称，描述和项目参与者所请求的角色（用户和组）。该项目必须存在，并且用户必须有权限查看。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project",
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

▶ 架构

状态404 如果返回演员不能添加到项目中的作用

▼ 添加用户的演员

POST /rest/api/2/project/{projectIdOrKey}/role/{id}

添加一个演员（用户或组）到项目的作用。

请求

例

```
{ "user": [ "admin" ] } or
{ "group": [ "jira-developers" ] }
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回角色名称，描述和项目参与者所请求的角色（用户和组）。该项目必须存在，并且用户必须有权限查看。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project",
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

▶ 架构

状态404 如果返回演员不能添加到项目中的作用

▼ 演员删除

DELETE /rest/api/2/project/{projectIdOrKey}/role/{id}

从项目的角色删除角色（用户或组）。

删除角色的用户： /rest/api/2/project/{projectIdOrKey}/role/{roleId}?user={username}

删除该角色的组： /rest/api/2/project/{projectIdOrKey}/role/{roleId}?group={groupname}

请求

查询参数

参数	类型	描述
user	串	用户名从该项目中删除角色
group	串	该组名从该项目中删除角色

回复

状态204 如果演员被成功地从项目角色中删除返回。

状态404 返回如果项目或角色没有找到，主叫用户无权查看，或没有权限修改演员在项目中的角色。

API / 2 / 项目 / { projectKeyOrId } / issuesecuritylevelscheme 隐藏所有方法

资源许可计划和项目联系起来。

▼ 获取问题的安全方案

GET /rest/api/2/project/{projectKeyOrId}/issuesecuritylevelscheme

返回项目问题的安全方案。

回复

状态200 - 应用程序/JSON 如果用户具有管理员权限, 或如果方案中, 其中用户具有管理权限的一个项目使用的返回。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issuesecurityschemes/1000",
  "id": 1000,
  "name": "Default Issue Security Scheme",
  "description": "Description for the default issue security scheme",
  "defaultSecurityLevelId": 10021,
  "levels": [
    {
      "self": "http://www.example.com/jira/rest/api/2/securitylevel/10021",
      "id": 10021,
      "description": "Only the reporter and internal staff can see this issue.",
      "name": "Reporter Only"
    }
  ]
}
```

► 架构

状态401 返回如果用户没有登录。

状态403 返回如果项目是主叫用户可见, 但用户不具有管理权限

状态404 返回如果该项目不存在, 或者不是给主叫用户可见

API / 2 / 项目 / { projectKeyOrId } / notificationscheme 隐藏所有方法

资源的通知计划和项目联系起来。

▼ 获取计划的通知

GET /rest/api/2/project/{projectKeyOrId}/notificationscheme

获取与项目相关的通知计划。遵循/ notificationscheme / {ID} 资源的文档, 了解有关返回值的所有细节。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 如果通知方式存在, 并且是可见的主叫用户返回

例

```
{
  "expand": "notificationSchemeEvents, user, group, projectRole, field, all",
  "id": 10100,
  "self": "http://example.com/jira/rest/api/2/notificationscheme/10100",
  "name": "notification scheme name",
  "description": "description",
  "notificationSchemeEvents": [
    {
      "event": {
        "id": 1,
        "name": "Issue created",
        "description": "Event published when issue is created"
      },
      "notifications": [
        {
          "id": 1,
          "notificationType": "Group",
          "parameter": "jira-administrators",
          "group": {
            "name": "jira-administrators",
            "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-administrators"
          },
          "expand": "group"
        },
        {
          "id": 2,
          "notificationType": "CurrentAssignee"
        },
        {
          "id": 3,
          "notificationType": "ProjectRole",
          "parameter": "10360",
          "projectRole": {
            "self": "http://www.example.com/jira/rest/api/2/project/MKY/role/10360",
            "name": "Developers",
            "id": 10360,
            "description": "A project role that represents developers in a project",
            "actors": [
              {
                "id": 10240,
                "displayName": "jira-developers",
                "type": "atlassian-group-role-actor"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

▶ 架构

状态401 返回如果用户没有登录。**状态403** 如果用户被允许访问该项目，但不是项目或JIRA的管理员，因此不能看到该项目的通知方案返回。**状态404** 返回如果通知方案不存在，或者是不向主叫用户可见

API / 2 / 项目 / {} projectKeyOrId / permissionscheme 隐藏所有方法

资源许可计划和项目联系起来。

▼ 分配权限方案

PUT /rest/api/2/project/{projectKeyOrId}/permissionscheme

分配与项目许可方案。

请求

查询参数

参数	类型	描述
expand	串	

例

```
{
  "id": 10000
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 新关联的许可计划的细节缩短。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
  "name": "Example permission scheme",
  "description": "description"
}
```

▶ 架构

状态401 返回如果用户没有登录。**状态403** 返回如果用户没有权限编辑项目的许可计划。在实践中，用户需要一个JIRA管理员。**状态404** 如果项目或许可方案未找到返回。

▼ 获得分配的权限方案

GET /rest/api/2/project/{projectKeyOrId}/permissionscheme

获取与项目中分配权限方案。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 相关的许可方案。

例

```
{
  "id": 10000,
  "self": "http://www.example.com/jira/rest/api/2/permissionscheme/10000",
  "name": "Example permission scheme",
  "description": "description"
}
```

▶ 架构

状态401 返回如果用户没有登录。

状态403 返回如果用户没有权限查看该项目的配置。在实践中，用户需要一个JRA管理员或项目管理员联系，获得分配的权限方案。

状态404 返回如果项目未找到（或用户没有权限查看项目）。

API / 2 / 项目 / {} projectKeyOrId / securitylevel 中 隐藏的所有方法

提供对当前用户的给定项目的安全级别的信息。

▼ 获取项目安全级别

GET /rest/api/2/project/{projectKeyOrId}/securitylevel

返回当前登录的用户可以访问项目的所有安全级别。如果用户没有设置问题安全许可，列表是空的。

回复

状态200 - 应用程序/JSON 返回的量，当前用户具有访问一个项目中的所有安全级别的列表。

例

```
{
  "levels": [
    {
      "self": "http://www.example.com/jira/rest/api/2/securitylevel/100000",
      "id": "100000",
      "description": "security description",
      "name": "securityLevelName"
    },
    {
      "self": "http://www.example.com/jira/rest/api/2/securitylevel/100001",
      "id": "100001",
      "description": "another security description",
      "name": "secret"
    }
  ]
}
```

▶ 架构

状态404 或者返回如果项目未找到该用户没有权限浏览它。

API / 2 / 项目 / 类型 隐藏所有方法

▼ 获取所有项目类型

GET /rest/api/2/project/type

返回所有在JIRA实例定义，没有考虑到许可证是否使用这些项目类型是否有效的项目类型。

回复

状态200 - 应用程序/JSON 返回的JIRA实例定义的所有项目类型的列表。

例

```
[
  {
    "key": "business",
    "formattedKey": "Business",
    "descriptionI18nKey": "Project type for business projects",
    "icon": "PD94bWwgd...",
    "color": "#FFFFFF"
  },
  {
    "key": "software",
    "formattedKey": "Software",
    "descriptionI18nKey": "Project type for software projects",
    "icon": "PD94bWwgd...",
    "color": "#AAAAAA"
  }
]
```

▶ 架构

▼ 找重点项目类型

GET /rest/api/2/project/type/{projectTypeKey}

返回与给定键的项目类型。

回复

状态200 - 应用程序/JSON 返回项目类型的表示指定id

例

```
{
  "key": "business",
  "formattedKey": "Business",
  "descriptionI18nKey": "Project type for business projects",
  "icon": "PD94bWwgd...",
  "color": "#FFFFFF"
}
```

▶ 架构

▼ 找键访问项目类型

GET /rest/api/2/project/type/{projectTypeKey}/accessible

返回与给定键的项目类型，如果是要登录的用户访问。这考虑到了用户是否在定义的项目类型的应用程序的许可。

回复

状态200 - 应用程序/JSON 返回项目类型的表示指定id

例

```
{
  "key": "business",
  "formattedKey": "Business",
  "descriptionI18nKey": "Project type for business projects",
  "icon": "PD94bWwgd...",
  "color": "#FFFFFF"
}
```

▶ 架构

状态401 的401的响应状态表示有在用户不是一个记录，因此，不能进行该操作

状态404 404响应状态表明，项目类型是无法访问的登录用户

API / 2 / projectCategory 隐藏所有方法

▼ 创建项目类别

POST /rest/api/2/projectCategory

通过创建一个POST项目类别。

请求

例

```
{
  "name": "CREATED",
  "description": "Created Project Category"
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果成功创建项目类别返回。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/projectCategory/10100",
  "id": "10100",
  "name": "CREATED",
  "description": "Created Project Category"
}
```

▶ 架构

状态401 如果调用者不那么登录没有权限创建项目类别返回。**状态403** 返回如果调用者进行身份验证并没有权限创建项目类别（不是全局管理员）。**状态409** 如果一个项目类别与给定的名称已经存在返回。

▼ 获取所有的项目类别

GET /rest/api/2/projectCategory

返回所有的项目类别

回复

状态200 - 应用程序/JSON 返回项目的类别列表。

例

```
[
  {
    "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",
    "id": "10000",
    "name": "FIRST",
    "description": "First Project Category"
  },
  {
    "self": "http://www.example.com/jira/rest/api/2/projectCategory/10001",
    "id": "10001",
    "name": "SECOND",
    "description": "Second Project Category"
  }
]
```

▶ 架构

状态500 如果返回在检索的项目列表中会出现错误。

▼ 删除项目类别

DELETE /rest/api/2/projectCategory/{id}

删除项目类别。

回复

状态401 如果调用者不那么登录没有权限删除项目类别返回。**状态204** 返回如果项目类别被成功删除。**状态403** 返回如果调用者进行身份验证并没有权限删除的项目类别（不是全局管理员）。**状态404** 如果项目的类别不存在或当前身份验证的用户没有权限查看它返回。

▼ 更新项目类别

PUT /rest/api/2/projectCategory/{id}

通过PUT修改项目类别。目前在把任何领域将覆盖现有值。为方便起见，如果一个字段不存在，它被忽略。

请求

例

```
{
  "name": "UPDATED",
  "description": "Updated Project Category"
}
```

▶ 架构

回复

状态200 如果项目类别存在返回，目前已验证的用户有权对其进行编辑。

▶ 架构

状态201

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/projectCategory/10100",
  "id": "10100",
  "name": "UPDATED",
  "description": "Updated Project Category"
}
```

▶ 架构

状态401 如果调用者不那么登录没有权限更改项目类别返回。

状态403 返回如果调用者进行身份验证并没有权限更改项目类别（不是全局管理员）。

状态404 如果项目的类别不存在或当前身份验证的用户没有权限查看它返回。

▼ 找ID项目类别

GET /rest/api/2/projectCategory/{id}

包含JSON格式的项目类别的表示。

回复

状态200 - 应用程序/ JSON 如果项目类别存在返回。包含JSON格式的项目类别的代表性。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/projectCategory/10000",
  "id": "10000",
  "name": "FIRST",
  "description": "First Project Category"
}
```

▶ 架构

状态404 如果项目的类别不存在或当前身份验证的用户没有权限查看它返回。

API / 2 / projectvalidate 隐藏所有方法

▼ 获取项目

GET /rest/api/2/projectvalidate/key

验证项目的关键。

请求

查询参数

参数	类型	描述
key	串	项目关键

回复

状态200 - 应用程序/ JSON 返回包含项目关键任何验证错误的ErrorCollection。

例

```
{
  "errorMessages": [],
  "errors": [
    {
      "projectKey": "A project with that project key already exists."
    }
  ]
}
```

▶ 架构

API / 2 /重新索引 隐藏所有方法

启动/停止/查询索引REST资源。

▼ 重新编制

POST /rest/api/2/reindex

揭开序幕**REINDEX**。需要管理员权限执行此**REINDEX**。

请求

查询参数

参数	类型	描述
type	串	不区分大小写字符串，表示重新索引类型。如果省略，则默认为 BACKGROUND_PREFERRED 。
indexComments	布尔	表示意见也应该重建索引。对于前景重新索引，其中的意见总是重新索引不相关。 默认值：假
indexChangeHistory	布尔	指示变更记录也应该重建索引。对于前景重新索引，在变更记录总是重新索引不相关。 默认值：假
indexWorklogs	布尔	指示变更记录也应该重建索引。对于前景重新索引，在变更记录总是重新索引不相关。 默认值：假

回复

状态202 - 应用程序/JSON 返回重新索引操作的进度表示。

例

```
{
  "progressUrl": "http://localhost:8080/jira",
  "currentProgress": 0,
  "currentSubTask": "Currently reindexing Change History",
  "type": "FOREGROUND",
  "submittedTime": "2016-09-06T09:48:12.322+0000",
  "startTime": "2016-09-06T09:48:12.322+0000",
  "finishTime": "2016-09-06T09:48:12.322+0000",
  "success": true
}
```

▶ 架构

▼ 获得重新索引信息

GET /rest/api/2/reindex

返回在系统重新索引信息。如果重新索引目前正在运行，则返回这个**REINDEX**信息。如果没有积极的指标任务，然后返回有关最新的重新索引任务运行信息，否则返回**404**表明没有重新索引已经发生。

请求

查询参数

参数	类型	描述
taskId	长	一个索引任务的 ID 您希望在获得详细信息。如果省略，则默认为标准行为，并返回在活动重新索引任务，或者最后一个任务信息，如果没有重新索引正在发生运行。。如果没有与该 ID 不重建索引任务则 404 返回。

回复

状态200 - 应用程序/JSON 返回重新索引操作的进度表示。

例

```
{
  "progressUrl": "http://localhost:8080/jira",
  "currentProgress": 0,
  "currentSubTask": "Currently reindexing Change History",
  "type": "FOREGROUND",
  "submittedTime": "2016-09-06T09:48:12.322+0000",
  "startTime": "2016-09-06T09:48:12.322+0000",
  "finishTime": "2016-09-06T09:48:12.322+0000",
  "success": true
}
```

▶ 架构

状态404 返回如果发现没有重新索引任务

▼ 重新编制问题

POST /rest/api/2/reindex/issue

重新索引一个或多个单独的问题。索引是同步进行 - 当的问题索引已完成或发生故障的调用返回。

请使用明确指定问题ID或一个JQL查询来选择问题重新索引。

请求

查询参数

参数	类型	描述
issueId	串	的ID或中的一个或多个问题的键重新索引。
indexComments	布尔	表示意见也应该重建索引。 默认值: 假
indexChangeHistory	布尔	指示变更记录也应该重建索引。 默认值: 假
indexWorklogs	布尔	指示变更记录也应该重建索引。 默认值: 假

回复

状态202 - 应用程序/JSON 返回重新索引操作的进度表示。

例

```
{
  "progressUrl": "http://localhost:8080/jira",
  "currentProgress": 0,
  "currentSubTask": "Currently reindexing Change History",
  "type": "FOREGROUND",
  "submittedTime": "2016-09-06T09:48:12.322+0000",
  "startTime": "2016-09-06T09:48:12.322+0000",
  "finishTime": "2016-09-06T09:48:12.322+0000",
  "success": true
}
```

▶ 架构

▼ 获得重新索引进度

GET /rest/api/2/reindex/progress

返回在系统重新索引信息。如果重新索引目前正在运行，则返回这个REINDEX信息。如果没有积极的指标任务，然后返回有关最新的重新索引任务运行信息，否则返回404表明没有重新索引已经发生。

请求

查询参数

参数	类 型	描述
taskId	长	一个索引任务的ID您希望在获得详细信息。如果省略，则默认为标准行为，并返回在活动重新索引任务，或者最后一个任务信息，如果没有重新索引正在发生运行。。如果没有与该ID不重建索引任务则404返回。

回复

状态200 - 应用程序/JSON 返回重新索引操作的进度表示。

例

```
{
  "progressUrl": "http://localhost:8080/jira",
  "currentProgress": 0,
  "currentSubTask": "Currently reindexing Change History",
  "type": "FOREGROUND",
  "submittedTime": "2016-09-06T09:48:12.322+0000",
  "startTime": "2016-09-06T09:48:12.322+0000",
  "finishTime": "2016-09-06T09:48:12.322+0000",
  "success": true
}
```

▶ 架构

状态404 返回如果发现没有重新索引任务

API / 2 / 重新索引/请求 隐藏所有方法

用于查询和执行REINDEX请求REST资源。

▼ 流程请求

POST /rest/api/2/reindex/request

执行任何挂起REINDEX请求。返回包含正在处理重新索引的请求ID的JSON阵列。执行是异步 - 返回的任务进度可以通过其他REST调用进行监控。

回复

状态200 - 应用程序/ JSON 返回要处理包含重新索引请求ID阵列。

▶ 架构

▼ 取得进展

GET /rest/api/2/reindex/request/{requestId}

检索单个重新索引请求的进度。

回复

状态200 - 应用程序/ JSON 详情及重新索引要求的状态。

例

```
{
  "id": 10500,
  "status": "PENDING",
  "type": "IMMEDIATE",
  "requestTime": "2016-09-06T09:48:12.303+0000"
}
```

▶ 架构

状态404 如果没有这样的要求重新索引存在返回。

▼ 取得进步散装

GET /rest/api/2/reindex/request/bulk

检索多个重新索引请求的进度。只有实际存在REINDEX请求将在结果中返回。

请求

查询参数

参数	类型	描述
requestId	串	重新索引请求ID。

回复

状态200 - 应用程序/ JSON 详情及重新索引要求的状态。

例

```
{
  "id": 10500,
  "status": "PENDING",
  "type": "IMMEDIATE",
  "requestTime": "2016-09-06T09:48:12.303+0000"
}
```

▶ 架构

API / 2 / 分辨率 隐藏所有方法

▼ 获取分辨率

GET /rest/api/2/resolution

返回所有分辨率的列表。

回复

状态200 - 应用程序/ JSON 如果决议存在，并且用户有权限查看它们返回。包含JSON格式的分辨率的全表示。

例

```
[  
  {  
    "self": "http://www.example.com/jira/rest/api/2/resolution/1",  
    "description": "A fix for this issue is checked into the tree and tested.",  
    "iconUrl": "http://www.example.com/jira/images/icons/statuses/resolved.png",  
    "name": "Fixed"  
  },  
  {  
    "self": "http://www.example.com/jira/rest/api/2/resolution/3",  
    "description": "This is what it is supposed to do.",  
    "name": "Works as designed"  
  }  
]
```

▶ 架构

▼ 获取分辨率

GET /rest/api/2/resolution/{id}

返回的决议。

回复

状态200 - 应用程序/JSON 如果分辨率存在，并且用户有权限查看返回。包含JSON格式的分辨率的全表示。

例

```
{  
  "self": "http://www.example.com/jira/rest/api/2/resolution/1",  
  "description": "A fix for this issue is checked into the tree and tested.",  
  "iconUrl": "http://www.example.com/jira/images/icons/statuses/resolved.png",  
  "name": "Fixed"  
}
```

▶ 架构

状态404 如果分辨率不存在，或者用户没有权限查看它返回。

API / 2 / 角色 隐藏所有方法

▼ 获取项目角色

GET /rest/api/2/role

得到JIRA所有可用ProjectRoles。目前，这份名单是全球性的。

回复

状态200 返回如果用户通过验证

例

```
{  
  "name": "Developers",  
  "id": 10360,  
  "description": "A project role that represents developers in a project"  
}
```

▶ 架构

状态401 返回如果您没有权限，或者您还没有登录。

状态403 返回如果请求的用户是不是管理员或系统管理员。

▼ 创建项目角色

POST /rest/api/2/role

创建一个新的ProjectRole在JIRA可用。创建的角色没有分配任何默认角色。

请求

例

```
{  
  "name": "MyRole",  
  "description": "role description"  
}
```

▶ 架构

回复

状态200 返回如果用户通过验证

例

```
{
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project"
}
```

▶ 架构

状态400 返回如果请求JSON没有一个名称字段或名称字段是无效的（空或开始或结尾空格）**状态401** 返回如果您还没有登录。**状态403** 返回如果您没有权限来创建一个角色。**状态409** 如果具有给定名称的角色已经存在返回。▼ 获取由**ID**项目角色

GET /rest/api/2/role/{id}

得到JIRA提供特定ProjectRole。

回复

状态200 返回如果用户通过验证

例

```
{
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project"
}
```

▶ 架构

状态401 返回如果您没有权限，或者您还没有登录。**状态403** 返回如果请求的用户是不是管理员或系统管理员。**状态404** 如果具有给定ID的角色不存在返回。

▼ 局部更新项目中的作用

POST /rest/api/2/role/{id}

部分更新了角色名称或说明。

请求

例

```
{
  "name": "MyRole",
  "description": "role description"
}
```

▶ 架构

回复

状态200 返回如果更新成功

例

```
{
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project"
}
```

▶ 架构

状态400 当没有给出这两个名称和描述或名称字段是无效的（空或开始或结尾空格）返回。**状态401** 返回，如果请求用户没有登录。**状态403** 返回如果请求的用户是不是管理员或系统管理员。**状态404** 如果具有给定ID的角色不存在返回。

▼ 完全更新项目中的作用

PUT /rest/api/2/role/{id}

全面更新了角色。名称和说明应予重视。

请求

例

```
{
  "name": "MyRole",
  "description": "role description"
}
```

▶ 架构

回复

状态200 返回如果更新成功

例

```
{
  "name": "Developers",
  "id": 10360,
  "description": "A project role that represents developers in a project"
}
```

▶ 架构

状态400 当没有给出返回名称或说明或名称字段是无效的（空或开始或结尾空格）**状态401** 返回，如果请求用户没有登录。**状态403** 返回如果请求的用户是不是管理员或系统管理员。**状态404** 如果具有给定ID的角色不存在返回。

▼ 删除工程的作用

DELETE /rest/api/2/role/{id}

删除一个角色。可能在未来返回**403**

请求

查询参数

参数	类型	描述
swap	长	如果给定的，删除由与给定的一个替换的作用，即使是在方案中使用的角色

回复

状态400 如果与给定的交换ID给角色不存在返回。**状态401** 返回，如果请求用户没有登录。**状态204** 返回如果删除成功。**状态403** 返回如果请求的用户是不是管理员或系统管理员。**状态404** 如果具有给定ID的角色不存在返回。**状态409** 如果项目中的作用在方案中使用和roleToSwap查询参数没有给出返回。

▼ 获取项目角色的演员为角色

GET /rest/api/2/role/{id}/actors

获取给定角色的默认行为。

回复

状态200 返回如果用户通过验证

例

```
{
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

▶ 架构

状态401 返回，如果请求用户没有登录。

状态403 返回如果请求的用户是不是管理员或系统管理员。

状态404 如果具有给定ID的角色不存在返回。

▼ 添加项目角色演员角色

POST /rest/api/2/role/{id}/actors

添加默认的演员给定角色。请求数据应该包含用户名列表或组添加的列表。

请求

例

```
{
  "user": [
    "admin"
  ]
}
```

▶ 架构

回复

状态200 返回如果更新成功

例

```
{
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

▶ 架构

状态400 如果返回的用户和组均给予或不给予或在各自的列表组或用户不存在。

状态401 返回，如果请求用户没有登录。

状态403 返回如果请求的用户是不是管理员或系统管理员。

状态404 如果具有给定ID的角色不存在返回。

▼ 删除角色项目中的角色的演员

DELETE /rest/api/2/role/{id}/actors

移除给定角色默认演员。

请求

查询参数

参数	类型	描述
user	串	如果有，将删除给定角色的演员
group	串	如果有，将删除给定角色的演员

回复

状态200 返回如果更新成功。

例

```
{
  "actors": [
    {
      "id": 10240,
      "displayName": "jira-developers",
      "type": "atlassian-group-role-actor",
      "name": "jira-developers"
    }
  ]
}
```

状态400 如果返回的用户和组没有给出，用户和组给予或提供的组或用户不存在。

状态401 返回，如果请求用户没有登录。

状态403 返回如果请求的用户是不是管理员或系统管理员。

状态404 如果具有给定ID的角色不存在返回。

API / 2 / 屏幕 隐藏所有方法

▼ 获取字段添加

GET /rest/api/2/screens/{screenId}/availableFields

获取屏幕可用字段。那些尚未被添加，即那些。

回复

状态200 对于屏幕可用字段列表

▶ 架构

状态400 如果屏幕不存在返回

状态401 如果您没有权限返回

▼ 添加标签

POST /rest/api/2/screens/{screenId}/tabs

创建选项卡定的屏幕

请求

▶ 架构

回复

状态200 - 应用程序/JSON 在JSON新创建的标签。

例

```
{
  "id": 10000,
  "name": "Fields Tab"
}
```

▶ 架构

状态400 如果返回屏幕不存在或标签名称无效

状态401 如果您没有权限返回

▼ 获取所有标签

GET /rest/api/2/screens/{screenId}/tabs

返回所有选项卡的列表，对于给定的屏幕

请求

查询参数

参数	类型	描述
projectKey	串	该项目的密钥; 此参数是可选

回复

状态200 - 应用程序/JSON 包含JSON所有可见标签的完整表示。

▶ 架构

状态400 如果屏幕不存在返回

状态401 如果您没有权限返回

▼ 重命名标签

PUT /rest/api/2/screens/{screenId}/tabs/{tabId}

鉴于重命名屏幕上的选项卡

请求

▶ 架构

回复

状态200 - 应用程序/JSON 改性JSON标签。

例

```
{
  "id": 10000,
  "name": "Fields Tab"
}
```

▶ 架构

状态400 如果返回屏幕不存在或标签名称无效**状态401** 如果您没有权限返回

▼ 删除选项卡

DELETE /rest/api/2/screens/{screenId}/tabs/{tabId}

删除选项卡给屏幕

回复

状态201 已成功删除标签**状态400** 如果屏幕或选项卡不存在，则返回**状态401** 如果您没有权限返回

▼ 添加字段

POST /rest/api/2/screens/{screenId}/tabs/{tabId}/fields

添加字段给定的标签。

请求

例

```
{
  "fieldId": "summary"
}
```

▶ 架构

回复

状态200 新增字段作为JSON

例

```
{
  "id": "summary",
  "name": "Summary"
}
```

▶ 架构

状态400 如果屏幕，选项卡或字段不存在返回。**状态401** 如果您没有权限返回

▼ 获取所有领域

GET /rest/api/2/screens/{screenId}/tabs/{tabId}/fields

获取所有字段为给定的标签

请求

查询参数

参数	类型	描述
projectKey	串	该项目的密钥; 此参数是可选

回复

状态200 对于给定的标签字段列表

▶ 架构

状态400 如果屏幕或选项卡不存在，则返回**状态401** 如果您没有权限返回**▼ 删除场**

DELETE /rest/api/2/screens/{screenId}/tabs/{tabId}/fields/{id}

删除字段从给定的标签

回复**状态201** 从标签成功移除场**状态400** 如果屏幕或选项卡不存在，则返回**状态401** 如果您没有权限返回**▼ 移动领域**

POST /rest/api/2/screens/{screenId}/tabs/{tabId}/fields/{id}/move

移动定标签上现场

请求

▶ 架构

回复**状态201** 成功移动标签**状态400** 如果屏幕或标签不存在返回。或移动coordinates无效。**状态401** 如果您没有权限返回**▼ 移动标签**

POST /rest/api/2/screens/{screenId}/tabs/{tabId}/move/{pos}

移动标签的位置

回复**状态201** 成功移动标签**状态400** 如果屏幕或选项卡不存在，则返回**状态401** 如果您没有权限返回**▼ 添加字段默认屏幕**

POST /rest/api/2/screens/addToDefault/{fieldId}

添加字段或自定义字段的默认选项卡

回复**状态201** 成功添加字段的默认屏幕/默认选项卡**状态400** 如果屏幕，选项卡或字段不存在或现场已经存在一个选择的选项卡上返回**状态401** 如果您没有管理员权限返回**API / 2 / 搜索** 隐藏的所有方法

资源，搜索。

▼ 搜索使用搜索请求

POST /rest/api/2/search

进行利用JQL搜索。

请求

例

```
{
  "jql": "project = HSP",
  "startAt": 0,
  "maxResults": 15,
  "fields": [
    "summary",
    "status",
    "assignee"
  ]
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回搜索结果的JSON表示。

例

```
{
  "expand": "names, schema",
  "startAt": 0,
  "maxResults": 50,
  "total": 1,
  "issues": [
    {
      "expand": "",
      "id": "10001",
      "self": "http://www.example.com/jira/rest/api/2/issue/10001",
      "key": "HSP-1"
    }
  ]
}
```

▶ 架构

状态400 如果有一个与JQL查询问题返回。

▼ 搜索

GET /rest/api/2/search

搜索使用JQL问题。

排序 的jql参数是一个完整的JQL 表达，并包括一个ORDER BY条款。

该fields参数（可多次指定）给出了一个逗号分隔的字段列表中应包括。这可以用于检索字段的子集。一个特定的领域可以通过负前缀就被排除在外。

默认情况下，唯一的通航（*navigable）字段在此搜索资源返回。注意：默认为在Get-问题的资源不同-默认情况下有所有字段（*all）。

*all - 包括所有领域

*navigable - 包括刚刚通航领域

summary, comment - 仅包含了总结和评论

-description-包括除了描述通航字段（默认为*navigable搜索）

*all, -comment - 除了包含所有评论

获得与POST: 如果JQL查询过大，编码为查询参数则应该张贴到此资源。

在搜索结果中扩大的问题：它可以扩大通过直接指定在这个资源传递的参数扩展返回的问题。

例如，为了扩大“更新日志”中关于搜索结果的所有问题，这是necessary指定“更新日志”，以扩大的一个值。

请求

查询参数

参数	类型	描述
jql	串	一个JQL查询字符串
startAt	INT	第一个问题的索引返回（从0开始）
maxResults	INT	问题的最大数目返回（默认为50）。的最大允许值由JIRA属性决定'jira.search.views.default.max'。如果指定的值比这个数字高，你的搜索结果将被截断。
validateQuery	布尔	是否验证JQL查询 默认 值：真
fields	串	字段列表返回每个问题。默认情况下，所有通航领域被返回。

expand 串 用逗号分隔的参数列表扩大。

回复

状态200 - 应用程序/ JSON 返回搜索结果的JSON表示。

例

```
{
  "expand": "names, schema",
  "startAt": 0,
  "maxResults": 50,
  "total": 1,
  "issues": [
    {
      "expand": "",
      "id": "10001",
      "self": "http://www.example.com/jira/rest/api/2/issue/10001",
      "key": "HSP-1"
    }
  ]
}
```

▶ 架构

状态400 如果有一个与JQL查询问题返回。

API / 2 / securitylevel 中 隐藏的所有方法

▼ 获取issuesecuritylevel

GET /rest/api/2/securitylevel/{id}

返回具有给定id的安全级别的完整表示。

回复

状态200 - 应用程序/ JSON 如果问题的类型存在且由主叫用户可见返回。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/securitylevel/10021",
  "id": "10021",
  "description": "Only the reporter and internal staff can see this issue.",
  "name": "Reporter Only"
}
```

▶ 架构

状态404 如果返回的问题类型不存在，或者不是给主叫用户可见。

API / 2 / serverInfo 隐藏所有方法

▼ 获取服务器信息

GET /rest/api/2/serverInfo

返回当前JIRA服务器的一般信息。

请求

查询参数

参数	类型	描述
doHealthCheck	布尔	

回复

状态200 - 应用程序/ JSON 返回JSON格式的服务器信息的完整表示

例

```
{
  "baseUrl": "http://localhost:8080/jira",
  "version": "5.0-SNAPSHOT",
  "versionNumbers": [
    5,
    0,
    0
  ],
  "buildNumber": 582,
  "buildDate": "2016-09-06T09:48:11.680+0000",
  "serverTime": "2016-09-06T09:48:11.680+0000",
  "scmInfo": "1f51473f5c7b75c1a69a0090f4832cdc5053702a",
  "buildPartnerName": "Example Partner Co.",
  "serverTitle": "My Shiny New JIRA Server"
}
```

▶ 架构

API / 2 / 设置 隐藏所有方法

为改变JIRA系统设置REST资源

▼ 设置基本URL

PUT /rest/api/2/settings/baseUrl

设置配置为这个JIRA实例的基本URL。

请求

例

`http://jira.atlassian.com/`

▶ 架构

▼ 找问题导航默认列

GET /rest/api/2/settings/columns

返回问题导航系统默认列。管理员权限是必需的。

回复

状态200 - 应用程序/JSON 返回列的列表进行配置给定用户

▶ 架构

状态500 返回如果检索列配置发生了错误。

状态403 如果用户没有管理权限返回

▼ 设置问题导航仪的默认列

PUT /rest/api/2/settings/columns

设置问题导航系统默认列。管理员权限是必需的。

请求

回复

状态200 当列保存成功返回

状态500 返回如果检索列配置发生了错误。

API / 2 / 状态 隐藏所有方法

▼ 找状态

GET /rest/api/2/status

返回所有状态的列表

回复

状态200 - 应用程序/JSON 以JSON格式返回所有的JIRA问题的状态，是对用户可见的列表。

例

```
[  
  {  
    "self": "http://localhost:8090/jira/rest/api/2.0/status/10000",  
    "description": "The issue is currently being worked on.",  
    "iconUrl": "http://localhost:8090/jira/images/icons/progress.gif",  
    "name": "In Progress",  
    "id": "10000",  
    "statusCategory": {  
      "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/1",  
      "id": 1,  
      "key": "in-flight",  
      "colorName": "yellow",  
      "name": "In Progress"  
    }  
  },  
  {  
    "self": "http://localhost:8090/jira/rest/api/2.0/status/5",  
    "description": "The issue is closed.",  
    "iconUrl": "http://localhost:8090/jira/images/icons/closed.gif",  
    "name": "Closed",  
    "id": "5",  
    "statusCategory": {  
      "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/9",  
      "id": 9,  
      "key": "completed",  
      "colorName": "green"  
    }  
  }  
]
```

▶ 架构

状态404 返回如果请求的问题状态没有找到，或者用户没有权限查看。

▼ 找状态

GET /rest/api/2/status/{idOrName}

返回具有给定id或名称状态的完整表示。

回复

状态200 - 应用程序/JSON 返回的JSON格式的JIRA问题状态的完整表示。

例

```
{  
  "self": "http://localhost:8090/jira/rest/api/2.0/status/10000",  
  "description": "The issue is currently being worked on.",  
  "iconUrl": "http://localhost:8090/jira/images/icons/progress.gif",  
  "name": "In Progress",  
  "id": "10000",  
  "statusCategory": {  
    "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/1",  
    "id": 1,  
    "key": "in-flight",  
    "colorName": "yellow",  
    "name": "In Progress"  
  }  
}
```

▶ 架构

状态404 返回如果请求的问题状态没有找到，或者用户没有权限查看。

API / 2 / statuscategory

隐藏所有方法

▼ 获取状态类别

GET /rest/api/2/statuscategory

返回所有状态类别列表

回复

状态200 - 应用程序/JSON 以JSON格式返回所有的JIRA问题状态类别，是对用户可见的列表。

例

```
[  
  {  
    "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/1",  
    "id": 1,  
    "key": "in-flight",  
    "colorName": "yellow",  
    "name": "In Progress"  
  },  
  {  
    "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/9",  
    "id": 9,  
    "key": "completed",  
    "colorName": "green"  
  }  
]
```

▶ 架构

状态404 - 返回如果没有状态类别被发现，或者用户没有权限查看。

▼ 获取状态类别

GET /rest/api/2/statuscategory/{idOrKey}

返回StatusCategory具有给定id或密钥完整表示

回复

状态200 - 应用程序/JSON 返回JSON格式的JIRA问题状态类别的完整表示。

例

```
{
  "self": "http://localhost:8090/jira/rest/api/2.0/statuscategory/1",
  "id": 1,
  "key": "in-flight",
  "colorName": "yellow",
  "name": "In Progress"
}
```

► 架构

状态404 - 返回如果未找到所请求的问题状态类，或者用户没有权限查看。

API / 2 / universal_avatar 隐藏所有方法

▼ 获取化身

GET /rest/api/2/universal_avatar/type/{type}/owner/{owningObjectId}

回复

状态 - 应用程序/JSON

▼ 创建临时的头像

POST /rest/api/2/universal_avatar/type/{type}/owner/{owningObjectId}/avatar

请求

► 架构

回复

状态 - 应用程序/JSON

▼ 删除头像

DELETE /rest/api/2/universal_avatar/type/{type}/owner/{owningObjectId}/avatar/{id}

删除头像

回复

状态 - 应用程序/JSON

▼ 存储临时头像

POST /rest/api/2/universal_avatar/type/{type}/owner/{owningObjectId}/temp

创建临时头像

请求

查询参数

参数	类型	描述
filename	串	文件的名称被上传
size	长	文件大小

回复

状态 - 应用程序/JSON

▼ 使用多部分存储临时头像

POST /rest/api/2/universal_avatar/type/{type}/owner/{owningObjectId}/temp

回复

状态 - 为text / html

API / 2 / 升级 隐藏所有方法

用于执行和查询延迟升级REST资源。

▼ 现在运行升级

POST /rest/api/2/upgrade

运行任何挂起延迟升级任务。需要管理员权限才能做到这一点。

回复

状态 - 应用程序/JSON

▼ 获取升级结果

GET /rest/api/2/upgrade

返回最后一个升级任务的结果。返回{@link javax.ws.rs.core.Response#seeOther (java.net.URI中) }如果仍在运行。

回复

状态200 - 应用程序/JSON 如果返回升级完成

例

```
{
    "startTime": "2016-09-06T09:48:13.461+0000",
    "duration": 2001,
    "outcome": "SUCCESS",
    "message": ""
}
```

▶ 架构

状态303 返回如果升级任务仍在运行。

状态404 如果没有事先升级任务存在返回。

API / 2 / 用户 隐藏所有方法

▼ 获得用户

GET /rest/api/2/user

返回用户。此资源不能被匿名访问。

请求

查询参数

参数	类型	描述
username	串	用户名
key	串	用户密钥

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
  "name": "fred",
  "emailAddress": "fred@example.com",
  "avatarUrls": {
    "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
    "24x24": "http://www.example.com/jira/secure/useravatar?size=small1&ownerId=fred",
    "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
    "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
  },
  "displayName": "Fred F. User",
  "active": true,
  "timeZone": "Australia/Sydney",
  "groups": {
    "size": 3,
    "items": [
      {
        "name": "jira-user",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-user"
      },
      {
        "name": "jira-admin",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-admin"
      },
      {
        "name": "important",
        "self": "http://www.example.com/jira/rest/api/2/group?groupname=important"
      }
    ]
  },
  "applicationRoles": {
    "size": 1,
    "items": []
  },
  "expand": "groups,applicationRoles"
}
```

▶ 架构

状态401 返回如果当前用户没有通过验证。**状态404** 如果所请求的用户未找到返回。▼ 用户创建 实验

POST /rest/api/2/user

创建用户。默认情况下创建的用户不会收到电子邮件通知。如果没有设置密码字段则密码将随机生成。

请求

例

```
{
  "name": "charlie",
  "password": "abracadabra",
  "emailAddress": "charlie@atlassian.com",
  "displayName": "Charlie of Atlassian",
  "applicationKeys": [
    "jira-core"
  ]
}
```

▶ 架构

回复

状态201 如果用户创建返回。

例

```
{
  "self": "http://www.example.com/jirahttp://www.example.com/jira/rest/api/2/user/charlie",
  "key": "charlie",
  "name": "charlie",
  "emailAddress": "charlie@atlassian.com",
  "displayName": "Charlie of Atlassian"
}
```

▶ 架构

状态400 返回如果请求是无效的。**状态401** 返回，如果用户不被认证。**状态500** 如果用户不是因为其他错误创建返回。**状态403** 返回如果调用者的用户没有权限来创建用户。▼ 更新用户 实验

PUT /rest/api/2/user

修改用户。目前“值”字段将覆盖现有的值。字段跳过请求将不会被改变。

请求

查询参数

参数	类型	描述
username	串	用户名
key	串	用户密钥

例

```
{
  "name": "eddie",
  "emailAddress": "eddie@atlassian.com",
  "displayName": "Eddie of Atlassian"
}
```

▶ 架构

回复

状态200 - 应用程序 / JSON 如果用户存在，并且调用者有权编辑它返回。

例

```
{
  "self": "http://www.example.com/jirahttp://www.example.com/jira/rest/api/2/user/charlie",
  "key": "charlie",
  "name": "charlie",
  "emailAddress": "charlie@atlassian.com",
  "displayName": "Charlie of Atlassian"
}
```

▶ 架构

状态400 返回如果请求是无效的。

状态401 返回，如果用户不被认证。

状态403 返回如果调用方用户没有权限编辑的用户。

状态404 返回如果调用者确实有权限编辑用户，但该用户不存在。

▼ 删除用户 实验

DELETE /rest/api/2/user

删除用户。

请求

查询参数

参数	类型	描述
username	串	用户名
key	串	用户密钥

回复

状态400 返回如果请求是无效或发生了一些其他的服务器错误。

状态401 返回，如果用户不被认证。

状态204 返回如果用户被成功删除。

状态403 返回如果调用者没有权限删除用户。

状态404 返回如果调用者有权限删除用户，但该用户不存在。

▼ 将用户添加到应用 实验

POST /rest/api/2/user/application

将用户添加到给定的应用。管理员权限将被要求执行该操作。

请求

查询参数

参数	类型	描述
username	串	用户名

applicationKey

串

应用程序键

回复

- 状态200** 如果用户存在返回，调用者已成功添加到应用程序的权限添加用户到应用程序和用户。
- 状态400** 返回如果请求是无效的。
- 状态401** 返回，如果用户不被认证。
- 状态403** 返回如果调用者的用户没有权限添加用户到应用程序。

▼ 从应用程序中删除用户

实验

DELETE /rest/api/2/user/application

取下给定应用程序的用户。管理员权限将被要求执行该操作。

请求

查询参数

参数	类型	描述
username	串	用户名
applicationKey	串	应用程序键

回复

- 状态400** 返回如果请求是无效的。
- 状态401** 返回，如果用户不被认证。
- 状态204** 如果用户存在返回，调用者有权从应用程序中删除用户和用户已成功从应用程序中删除。
- 状态403** 返回如果调用者的用户没有权限从应用程序中删除用户。

▼ 查找批量分配用户

GET /rest/api/2/user/assignable/multiProjectSearch

返回匹配搜索字符串和可分配为所有给定项目的问题是用户的列表。此资源不能被匿名访问。

请求

查询参数

参数	类型	描述
username	串	用户名
projectKeys	串	项目的关键，我们正在寻找分配为用户，用逗号分隔
startAt	INT	第一用户的索引返回（从0开始）
maxResults	INT	用户的最大数量返回（默认为50）。最大允许值为1000。如果指定的值比这个数字更高，搜索结果将被截断。

回复

- 状态200** - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
[  
  {  
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",  
    "name": "fred",  
    "avatarUrls": [  
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",  
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",  
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",  
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"  
    ],  
    "displayName": "Fred F. User",  
    "active": false  
  },  
  {  
    "self": "http://www.example.com/jira/rest/api/2/user?username=andrew",  
    "name": "andrew",  
    "avatarUrls": [  
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=andrew",  
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=andrew",  
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=andrew",  
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=andrew"  
    ],  
    "displayName": "Andrew Anderson",  
    "active": false  
  }  
]
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态404 如果所请求的用户未找到返回。

▼ 查找用户分配

GET /rest/api/2/user/assignable/search

返回与搜索字符串匹配用户的列表。此资源不能被匿名访问。请注意，该资源应该问题键时分配的用户的列表是用于编辑检索调用。对于只创建应提供一个项目的关键。如果它被称为与编辑项目的关键分配的用户列表可能不正确。

请求

查询参数

参数	类型	描述
username	串	用户名
project	串	该项目的关键，我们发现可分配的用户为
issueKey	串	对这一问题的关键问题正在编辑，我们需要找到分配的用户提供。
startAt	INT	第一用户的索引返回（从0开始）
maxResults	INT	用户的最大数量返回（默认为50）。最大允许值为1000。如果指定的值比这个数字更高，搜索结果将被截断。
actionDescriptorId	INT	

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
  "name": "fred",
  "emailAddress": "fred@example.com",
  "avatarUrls": {
    "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
    "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
    "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
    "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
  },
  "displayName": "Fred F. User",
  "active": true,
  "timeZone": "Australia/Sydney",
  "groups": [
    {
      "size": 3,
      "items": [
        {
          "name": "jira-user",
          "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-user"
        },
        {
          "name": "jira-admin",
          "self": "http://www.example.com/jira/rest/api/2/group?groupname=jira-admin"
        },
        {
          "name": "important",
          "self": "http://www.example.com/jira/rest/api/2/group?groupname=important"
        }
      ]
    }
  ],
  "applicationRoles": {
    "size": 1,
    "items": []
  },
  "expand": "groups,applicationRoles"
}
```

▶ 架构

状态400 如果没有提供项目或问题键返回**状态401** 返回如果当前用户没有通过验证。**状态404** 如果所请求的用户未找到返回。

▼ 创建临时的头像

POST /rest/api/2/user/avatar

临时化身转换成一个真正的头像

请求

查询参数

参数	类型	描述
username	串	用户名

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "needsCropping": false
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 返回创建头像

例

```
{
  "id": "1000",
  "owner": "fred",
  "isSystemAvatar": true,
  "isSelected": false,
  "isDeletable": false,
  "urls": {
    "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
    "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
    "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
    "48x48": "http://localhost:8090/jira/secure/useravatar?size=large&avatarId=10040"
  },
  "selected": false
}
```

▶ 架构

状态400 返回如果裁剪坐标无效**状态500** 返回如果在临时化身转换为真正的化身出现错误**状态403** 返回如果当前身份验证的用户没有权限来接头像**状态404** 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 更新项目的头像

PUT /rest/api/2/user/avatar

请求

查询参数

参数	类型	描述
username	串	

▶ 架构

回复

状态 - 应用程序/JSON

▼ 删除头像

DELETE /rest/api/2/user/avatar/{id}

删除头像

请求

查询参数

参数	类型	描述
username	串	用户名

回复

状态204 - 应用程序/JSON 如果返回的头像被成功删除。

状态403 返回如果当前身份验证的用户没有权限删除头像。

状态404 如果分身不存在，或当前身份验证的用户没有权限删除它返回。

▼ 存储临时头像

POST /rest/api/2/user/avatar/temporary

创建临时的头像。创建一个临时的头像是在上传新的头像为用户提供3个步骤的一部分：上传，裁剪，确认。

下面的实施例示出了使用卷曲这三个步骤。饼干（会话）需要请求之间要保留，因此，使用**-b**和**-c**的。在步骤2中创建的ID需要传递到步骤3（可以简单地传递步骤2的作为步骤3的请求的整个响应）。

```
卷曲-c cookiejar.txt -X POST -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 图像/ PNG" --data二进制@ mynewavatar.png \
的 "http://本地主机: 8090 / JIRA / REST / API / 2 / 用户/化身/临时用户名=管理员及文件名= mynewavatar.png"
```

```
卷曲-b cookiejar.txt -X POST -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 应用程序/ JSON" --data {"cropperWidth": "65", "cropperOffsetX": "10", "cropperOffsetY": "16"} \
-o tmpid.json \
HTTP: //本地主机: 8090 / JIRA / REST / API / 2 / 用户/用户名头像管理员=
```

```
卷曲-b cookiejar.txt -X PUT -u管理: 管理员-H "X-Atlassian的令牌: 无检查" \
-H "内容类型: 应用程序/ JSON" --data二进制@ tmpid.json \
HTTP: //本地主机: 8090 / JIRA / REST / API / 2 / 用户/用户名头像管理员=
```

请求

查询参数

参数	类型	描述
username	串	用户名
filename	串	文件的名称被上传

size	长	文件大小
------	---	------

回复

状态201 - 应用程序/JSON 临时头像裁剪说明

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态500 返回如果在临时化身转换为真正的化身出现错误

状态403 如果该请求不包含有效XSRF令牌返回

状态404 返回如果当前身份验证的用户不具有编辑项目权限。

▼ 使用多部分存储临时头像

POST /rest/api/2/user/avatar/temporary

创建使用多临时头像。响应被发送回存储在一个textarea JSON。这是因为客户端使用远程framing使用多提交化身。因此，我们必须从回该客户端从解析JSON给他们一个有效的HTML页面。

创建一个临时的头像是在上传新的头像为用户提供3个步骤的一部分：上传，裁剪，确认。此端点允许您使用多部分上传的，而不是直接发送图像作为请求体。

你*必须*使用“阿凡达”作为上传参数的名称：

```
curl -c cookie.jar.txt -X POST -u 管理员:管理员 -H "X-Atlassian的令牌: 无检查" \
-F "avatar=@mynewavatar.png;类型=图像/ PNG" \
的 "http://本地主机: 8090 / JIRA / REST / API / 2 / 用户/化身/临时用户名=管理员"
```

请求

查询参数

参数	类型	描述
username	串	用户名

回复

状态201 - text/html 的 临时头像裁剪指令内嵌在HTML页面。错误信息也将在页面中内嵌。

例

```
{
  "cropperWidth": 120,
  "cropperOffsetX": 50,
  "cropperOffsetY": 50,
  "url": "http://example.com/jira/secure/temporaryavatar?cropped=true",
  "needsCropping": true
}
```

▶ 架构

状态500 返回如果在临时化身转换为真正的化身出现错误

状态404 如果用户不存在返回

▼ 获取所有的化身

GET /rest/api/2/user/avatars

返回该可见在当前登录用户的所有化身。

请求

查询参数

参数	类型	描述
username	串	用户名

回复

状态200 - 应用程序/JSON 返回包含头像列表为定制的系统头像地图

例

```
{
  "system": [
    {
      "id": "1000",
      "owner": "fred",
      "isSystemAvatar": true,
      "isSelected": false,
      "isDeletable": false,
      "urls": {
        "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10040",
        "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10040",
        "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10040",
        "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10040"
      },
      "selected": false
    }
  ],
  "custom": [
    {
      "id": "1010",
      "owner": "andrew",
      "isSystemAvatar": false,
      "isSelected": false,
      "isDeletable": true,
      "urls": {
        "16x16": "http://localhost:8090/jira/secure/useravatar?size=xsmall&avatarId=10080",
        "24x24": "http://localhost:8090/jira/secure/useravatar?size=small&avatarId=10080",
        "32x32": "http://localhost:8090/jira/secure/useravatar?size=medium&avatarId=10080",
        "48x48": "http://localhost:8090/jira/secure/useravatar?avatarId=10080"
      },
      "selected": false
    }
  ]
}
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态500 如果返回在检索化身的列表时发生错误。

状态404 如果所请求的用户未找到返回。

▼ 默认列

GET /rest/api/2/user/columns

返回给定用户的默认列。管理员权限将被要求获得列比用户当前登录的其他用户。

请求

查询参数

参数	类型	描述
username	串	用户名

回复

状态200 - 应用程序/JSON 返回列的列表进行配置给定用户

▶ 架构

状态401 返回如果当前用户是不允许请求的列给定用户。

状态500 返回如果检索列配置发生了错误。

状态404 如果所请求的用户未找到返回。

▼ 集列

PUT /rest/api/2/user/columns

设置为给定用户的默认列。管理员权限将被要求获得列比用户当前登录的其他用户。

请求

回复

状态200 当列保存成功返回

状态500 返回如果检索列配置发生了错误。

▼ 重置列

DELETE /rest/api/2/user/columns

重置默认列给定用户到系统默认值。管理员权限将被要求获得列比用户当前登录的其他用户。

请求

查询参数

参数	类型	描述
username	串	用户名

回复

状态401 返回如果当前用户是不允许请求的列给定用户。**状态500** 返回如果在重置列配置发生了错误。**状态204** 当列复位成功返回

▼ 更改用户密码

实验

PUT /rest/api/2/user/password

修改用户密码。

请求

查询参数

参数	类型	描述
username	串	用户名
key	串	用户密钥

例

```
{
  "password": "new password"
}
```

架构

回复

状态400 返回如果请求是无效的。**状态401** 返回，如果用户不被认证。**状态204** 如果用户存在，并且调用者有权编辑它返回。**状态403** 返回如果调用者没有权限更改用户密码。**状态404** 返回如果调用者有权更改用户密码，但用户不存在。

▼ 找到所有权限的用户

GET /rest/api/2/user/permission/search

返回与搜索字符串匹配和对项目或问题的所有规定权限活动用户的列表。

这个资源可以通过用户与项目或全球ADMIN或系统管理员权限ADMINISTER_PROJECT权限访问。

请求

查询参数

参数	类型	描述
username	串	用户名过滤器列表包括如果未指定的所有用户
permissions	串	逗号分隔的权限列表项目或问题返回的用户必须拥有，请参阅 权限 的JavaDoc所有可能的权限列表。
issueKey	串	对这一问题的关键问题为其返回的用户都有指定的权限。
projectKey	串	可选的项目项，以搜索与如果没有issueKey提供用户。

startAt	<i>INT</i>	第一用户的索引返回（从0开始）
maxResults	<i>INT</i>	用户的最大数量返回（默认为50）。最大允许值为1000。如果指定的值比这个数字更高，搜索结果将被截断。

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
[ {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": [
        "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
        "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
        "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
        "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    ],
    "displayName": "Fred F. User",
    "active": false
},
{
    "self": "http://www.example.com/jira/rest/api/2/user?username=andrew",
    "name": "andrew",
    "avatarUrls": [
        "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=andrew",
        "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=andrew",
        "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=andrew",
        "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=andrew"
    ],
    "displayName": "Andrew Anderson",
    "active": false
} ]
```

▶ 架构

状态400 如果没有提供项目或问题键或返回时，权限列表为空或包含无效的项

状态401 返回如果当前用户没有通过验证。

状态403 如果当前用户不具有项目管理权限返回。

状态404 如果未找到所请求的问题或项目返回。

▼ 查找选择器用户

GET /rest/api/2/user/picker

返回用户突出匹配查询的列表。此资源不能被匿名访问。

请求

查询参数

参数	类型	描述
query	串	使用的字符串搜索用户名、姓名或电子邮件地址
maxResults	<i>INT</i>	用户的最大数量返回（默认为50）。最大允许值为1000。如果指定的值比这个数字更高，搜索结果将被截断。
showAvatar	布尔	
exclude	串	

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
{ "users": [
    {
        "name": "fred",
        "key": "fred",
        "html": "fred@example.com",
        "displayName": "Fred Grumble",
        "avatarUrl": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred"
    }
],
"total": 25,
"header": "Showing 20 of 25 matching groups"
}
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态404 如果所请求的用户未找到返回。

▼ 查找用户

GET /rest/api/2/user/search

返回与搜索字符串匹配用户的列表。此资源不能被匿名访问。

请求

查询参数

参数	类型	描述
username	串	使用的查询字符串来搜索用户名, 姓名或电子邮件地址
startAt	INT	第一用户的索引返回 (从0开始)
maxResults	INT	用户的最大数量返回 (默认为50)。最大允许值为1000。如果指定的值比这个数字更高, 搜索结果将被截断。
includeActive	布尔	如果属实, 那么活跃用户都包括在结果中 (默认为true)
includeInactive	布尔	如果属实, 那么不活动的用户都包括在结果中 (默认为false)

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
[ {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
        "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
        "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
        "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
        "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
},
{
    "self": "http://www.example.com/jira/rest/api/2/user?username=andrew",
    "name": "andrew",
    "avatarUrls": {
        "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=andrew",
        "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=andrew",
        "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=andrew",
        "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=andrew"
    },
    "displayName": "Andrew Anderson",
    "active": false
} ]
```

▶ 架构

状态401 返回如果当前用户没有通过验证。

状态404 如果所请求的用户未找到返回。

▼ 找到浏览权限的用户

GET /rest/api/2/user/viewissue/search

返回与搜索字符串匹配的活跃用户列表。此资源不能被匿名访问并要求用户浏览全局权限。鉴于问题的关键这一资源将提供匹配搜索字符串并有提供这个问题的浏览权限问题用户的列表。

请求

查询参数

参数	类型	描述
username	串	用户名过滤器, 如果留空, 没有用户回来
issueKey	串	对这一问题的关键问题正在编辑, 我们需要找到可视用户提供。
projectKey	串	可选的项目项, 以搜索与如果没有issueKey提供用户。
startAt	INT	第一用户的索引返回 (从0开始)
maxResults	INT	用户的最大数量返回 (默认为50)。最大允许值为1000。如果指定的值比这个数字更高, 搜索结果将被截断。

回复

状态200 - 应用程序/JSON 返回JIRA用户JSON格式的完整表示。

例

```
[
  {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": false
  },
  {
    "self": "http://www.example.com/jira/rest/api/2/user?username=andrew",
    "name": "andrew",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=andrew",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=andrew",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=andrew",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=andrew"
    },
    "displayName": "Andrew Anderson",
    "active": false
  }
]
```

▶ 架构

状态400 如果没有提供项目或问题键返回**状态401** 返回如果当前用户没有通过验证。**状态404** 如果未找到所请求的问题或项目返回。

API / 2 / 用户/属性 隐藏所有方法

▼ 获取属性键

GET /rest/api/2/user/properties

返回由键或由ID标识的用户的所有属性的键。

请求

查询参数

参数	类型	描述
userKey	串	其属性是要返回的用户的键
username	串	其属性是要返回的用户的用户名

回复

状态200 - 应用程序/JSON 如果用户被发现返回。

例

```
{
  "keys": [
    {
      "self": "http://www.example.com/jira/rest/api/2/issue/EX-2/properties/issue.support",
      "key": "issue.support"
    }
  ]
}
```

▶ 架构

状态400 返回如果用户密钥或ID无效。**状态401** 返回如果调用用户没有通过验证。**状态403** 返回如果主叫用户不具有权限浏览的用户。**状态404** 如果返回给定键或ID的用户不存在，或者给定键的属性是找不到的。

▼ 删除属性

DELETE /rest/api/2/user/properties/{propertyKey}

移除按键式或由ID标识的用户属性。THS用户删除属性是需要有权限来管理用户。

请求

查询参数

参数	类型	描述
userKey	串	其属性的用户的关键是要除去
username	串	其属性的用户的用户名是要除去

回复

- 状态400** 返回如果用户密钥或ID无效。
- 状态401** 返回如果调用用户没有通过验证。
- 状态204** 如果返回成功删除的用户属性。
- 状态403** 如果返回调用用户没有权限编辑用户。
- 状态404** 如果返回给定键或ID的用户不存在，或者给定键的属性是找不到的。

▼ 设置属性

PUT /rest/api/2/user/properties/{propertyKey}

设置指定用户的属性值。

你可以使用这个资源来存储反对键或由ID标识的用户自定义数据。谁存储数据的用户需要具有的权限来管理用户。

请求

查询参数

参数	类型	描述
userKey	串	其属性的用户的关键是要设定
username	串	其属性的用户的用户名是要设定

回复

- 状态200** 返回如果用户属性被成功更新。
- 状态201** 如果成功创建的用户属性返回。
- 状态400** 返回如果用户密钥或ID无效。
- 状态401** 返回如果调用用户没有通过验证。
- 状态403** 返回如果主叫用户没有权限管理该用户。
- 状态404** 如果与给定的密钥或ID的用户不存在返回。

▼ Get属性

GET /rest/api/2/user/properties/{propertyKey}

返回与从键或由id标识的用户给定键的属性的值。谁检索的属性的用户需要有阅读权限的用户。

请求

查询参数

参数	类型	描述
userKey	串	其属性的用户的关键是要返回
username	串	其属性的用户的用户名是要返回

回复

- 状态200** - 应用程序/JSON 如果用户属性被发现返回。

例

```
{
  "key": "issue.support",
  "value": {
    "hipchat.room.id": "support-123",
    "support.time": "1m"
  }
}
```

▶ 架构

- 状态400 返回如果用户密钥或ID无效。
- 状态401 返回如果调用用户没有通过验证。
- 状态403 返回如果主叫用户不具有权限浏览的用户。
- 状态404 如果返回给定键或ID的用户不存在，或者给定键的属性是找不到的。

API / 2 / 版本 隐藏所有方法

▼ 创建版本

POST /rest/api/2/version

通过创建一个POST版本。

请求

例

```
{
  "description": "An excellent version",
  "name": "New Version 1",
  "archived": false,
  "released": true,
  "releaseDate": "2010-07-06",
  "userReleaseDate": "6/Jul/2010",
  "project": "PXA",
  "projectId": 10000
}
```

▶ 架构

回复

状态201 - 应用程序/JSON 如果成功创建的版本返回。

例

```
{
  "description": "An excellent version",
  "name": "New Version 1",
  "archived": false,
  "released": true,
  "releaseDate": "2010-07-06",
  "userReleaseDate": "6/Jul/2010",
  "project": "PXA",
  "projectId": 10000
}
```

▶ 架构

状态403 返回如果当前身份验证的用户没有权限编辑的版本。

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 移动版

POST /rest/api/2/version/{id}/move

在一个项目中修改版本的序列。

此举版bean有2场的替代值对：

位置

绝对位置，其可以具有值“第一”，“上次”，“早期”或“后来”

后

一个版本后把这个版本。值应为另一个版本的自链接

请求

例

```
{
  "after": "http://www.example.com/jira/rest/api/2/version/10000"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 如果版本存在返回，目前已验证的用户有权限查看。包含的版本完整表示感动。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/version/10000",
  "id": "10000",
  "description": "An excellent version",
  "name": "New Version 1",
  "archived": false,
  "released": true,
  "releaseDate": "2010-07-06",
  "overdue": true,
  "userReleaseDate": "6/Jul/2010",
  "projectId": 10000
}
```

▶ 架构

状态404 如果以后移动版本的版本，或者目标不存在或当前身份验证的用户没有权限查看它返回。

▼ 获取版本

GET /rest/api/2/version/{id}

返回一个项目版本。

请求

查询参数

参数	类型	描述
expand	串	

回复

状态200 - 应用程序/JSON 如果版本存在返回，目前已验证的用户有权限查看。包含的版本完整表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/version/10000",
  "id": "10000",
  "description": "An excellent version",
  "name": "New Version 1",
  "archived": false,
  "released": true,
  "releaseDate": "2010-07-06",
  "overdue": true,
  "userReleaseDate": "6/Jul/2010",
  "projectId": 10000
}
```

▶ 架构

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 更新版本

PUT /rest/api/2/version/{id}

通过PUT修改版本。目前在把任何领域将覆盖现有值。为方便起见，如果一个字段不存在，它被忽略。

请求

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/version/10000",
  "id": "10000",
  "description": "An excellent version",
  "name": "New Version 1",
  "archived": false,
  "released": true,
  "releaseDate": "2010-07-06",
  "overdue": true,
  "userReleaseDate": "6/Jul/2010",
  "projectId": 10000
}
```

▶ 架构

回复

状态200 如果版本存在返回，目前已验证的用户有权对其进行编辑。

▶ 架构

状态403 返回如果当前身份验证的用户没有权限编辑的版本。

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 删除

DELETE /rest/api/2/version/{id}

删除项目版本。

请求

查询参数

参数	类型	描述
moveFixIssuesTo	串	该版本fixVersion置问题上如果删除的版本是修复版本，如果为null，则fixVersion被删除。
moveAffectedIssuesTo	串	该版本affectedVersion置问题上如果删除的版本是受影响的版本，如果为null，则affectedVersion被删除。

回复

状态204 如果返回的版本被成功删除。**状态403** 返回如果当前身份验证的用户没有权限删除的版本。**状态404** 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 合并

PUT /rest/api/2/version/{id}/mergeto/{moveIssuesTo}

合并版本

回复

状态204 如果返回的版本被成功删除。**状态403** 返回如果当前身份验证的用户没有权限删除的版本。**状态404** 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 获取版本相关的问题

GET /rest/api/2/version/{id}/relatedIssueCounts

返回包含了给定的版本在固定和受影响的问题数量的bean。

回复

状态200 - 应用程序/JSON 如果版本存在返回，目前已验证的用户有权限查看。包含固定的，影响这个版本中，自定义字段连接到这个版本问题计数。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/version/10000",
  "issuesFixedCount": 23,
  "issuesAffectedCount": 101,
  "issueCountWithCustomFieldsShowingVersion": 54,
  "customFieldUsage": [
    {
      "fieldName": "Field1",
      "customFieldId": 10000,
      "issueCountWithVersionInCustomField": 2
    },
    {
      "fieldName": "Field2",
      "customFieldId": 10010,
      "issueCountWithVersionInCustomField": 3
    }
  ]
}
```

▶ 架构

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 删除

POST /rest/api/2/version/{id}/removeAndSwap

删除项目版本。

请求

▶ 架构

回复

状态204 如果返回的版本被成功删除。

状态403 返回如果当前身份验证的用户没有权限删除的版本。

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 获取版本尚未解决的问题

GET /rest/api/2/version/{id}/unresolvedIssueCount

返回的未解决的问题数量为给定的版本

回复

状态200 - 应用程序/JSON 如果版本存在返回，目前已验证的用户有权限查看。包含在这个版本中尚未解决的问题数量。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/version/10000",
  "issuesUnresolvedCount": 23
}
```

► 架构

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 通过获取版本ID远程版本链接

GET /rest/api/2/version/{versionId}/remotelink

返回与给定的版本ID相关联的远程版本的链接。

回复

状态200 - 应用程序/JSON 如果版本存在返回，目前已验证的用户有权限查看，这是仅限于那些用户提供项目浏览权限。包含远程版本的链接的完整表示。

例

```
{
  "links": [
    {
      "self": "http://www.example.com/version/10000/AnotherGlobalId",
      "name": "Version 1",
      "link": {
        "globalId": "AnotherGlobalId",
        "myCustomLinkProperty": false,
        "colors": [
          "cyan",
          "magenta",
          "yellow"
        ]
      }
    },
    {
      "self": "http://www.example.com/version/10000/SomeGlobalId",
      "name": "Version 1",
      "link": {
        "globalId": "SomeGlobalId",
        "myCustomLinkProperty": true,
        "colors": [
          "red",
          "green",
          "blue"
        ]
      }
    }
  ]
}
```

► 架构

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 创建或更新远程版本链接

POST /rest/api/2/version/{versionId}/remotelink

通过创建一个POST远程版本的链接。链接的全局ID将从JSON有效载荷的是，如果提供的；否则，它会产生。

请求

例

```
{
  "globalId": "SomeGlobalId",
  "myCustomLinkProperty": true,
  "colors": [
    "red",
    "green",
    "blue"
  ],
  "notes": [
    "Remote version links may take any well-formed JSON shape that is desired,",
    "provided that they fit within the maximum buffer size allowed,",
    "which is currently 32,768 characters."
  ]
}
```

▶ 架构

回复

状态201 如果成功创建的版本返回。该文件将没有内容，以及{@code 地点：}头包含自身URI为新创建的链接。

例

```
Returned if the remote version link is created or updated successfully.  
The document has no content, and a
```

状态400 返回如果JSON有效载荷为空或畸形

状态403 返回如果当前身份验证的用户没有权限编辑的版本。

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 通过删除版本ID远程版本链接

DELETE /rest/api/2/version/{versionId}/remotelink

删除一个给定的版本ID的所有远程版本的链接。

回复

状态204 返回如果远程版本的链接被成功删除。

状态403 如果当前身份验证的用户不具有项目管理权限，因此不能删除远程链接的版本返回。

状态404 返回如果版本不存在，目前已验证的用户没有权限查看它，或者版本没有任何远程链接删除

▼ 获取远程版本链接

GET /rest/api/2/version/{versionId}/remotelink/{globalId}

表示远程版本链接休息子资源

回复

状态200 - 应用程序/JSON 遥控器版本链接上的信息（或链接）

例

```
{
  "self": "http://www.example.com/version/10000/SomeGlobalId",
  "name": "Version 1",
  "link": [
    {
      "globalId": "SomeGlobalId",
      "myCustomLinkProperty": true,
      "colors": [
        "red",
        "green",
        "blue"
      ]
    }
  ]
}
```

▶ 架构

状态404 如果版本或远程版本链接不存在，或者如果用户不具有对于拥有该指定版本的项目的浏览权限返回

▼ 创建或更新远程版本链接

POST /rest/api/2/version/{versionId}/remotelink/{globalId}

通过创建一个POST远程版本的链接。链接的全球ID将从JSON有效载荷的是，如果提供的；否则，它会产生。

请求

例

```
{
  "globalId": "SomeGlobalId",
  "myCustomLinkProperty": true,
  "colors": [
    "red",
    "green",
    "blue"
  ],
  "notes": [
    "Remote version links may take any well-formed JSON shape that is desired,",
    "provided that they fit within the maximum buffer size allowed,",
    "which is currently 32,768 characters."
  ]
}
```

▶ 架构

回复

状态201 如果成功创建的版本返回。该文件将没有内容，以及{@code 地点：}头包含自身URI为新创建的链接。

例

```
Returned if the remote version link is created or updated successfully.  
The document has no content, and a
```

状态400 返回如果JSON有效载荷为空或畸形

状态403 返回如果当前身份验证的用户没有权限编辑的版本。

状态404 如果版本不存在或当前身份验证的用户没有权限查看它返回。

▼ 删除远程版本链接

DELETE /rest/api/2/version/{versionId}/remotelink/{globalId}

删除与给定的版本ID和全局ID特定的远程版本链接。

回复

状态204 返回如果远程版本的链接被成功删除。

状态403 如果当前身份验证的用户不具有项目管理权限，因此不能删除远程链接的版本返回。

状态404 返回如果版本不存在，目前已验证的用户没有权限查看，或版本没有一个链接给定的全局ID

▼ 获取远程版本链接

GET /rest/api/2/version/remotelink

返回给定的全局ID远程版本的链接。

请求

查询参数

参数	类型	描述
globalId	串	远程资源的全局ID被链接到的版本

回复

状态200 - 应用程序/JSON 包含远程版本链接列表。该用户没有权限查看的任何现有链接将被过滤掉。用户必须拥有浏览权限项目以查看链接到它的版本。

例

```
{
  "links": [
    {
      "self": "http://www.example.com/version/10000/SomeGlobalId",
      "name": "Version 1",
      "link": {
        "globalId": "SomeGlobalId",
        "myCustomLinkProperty": true,
        "colors": [
          "red",
          "green",
          "blue"
        ]
      }
    },
    {
      "self": "http://www.example.com/version/10101/SomeGlobalId",
      "name": "Version 2",
      "link": {
        "globalId": "SomeGlobalId"
      }
    }
  ]
}
```

▶ 架构

API / 2 / 工作流程 隐藏所有方法

检索工作流程REST资源。

▼ 获取所有的工作流程

GET /rest/api/2/workflow

返回所有的工作流程。

请求

查询参数

参数	类型	描述
workflowName	串	

回复

状态200 - 应用程序/JSON 如果当前验证的用户具有行政许可返回。包含所有工作流程的完整表示。

▶ 架构

▶ 架构

状态401 如果当前身份验证的用户不具有行政许可返回。

▼ 创建属性

POST /rest/api/2/workflow/{id}/properties

一个新的属性添加到一个过渡。尝试添加已经存在失败的属性。

请求

查询参数

参数	类型	描述
key	串	该属性的名称的补充。
workflowName	串	使用工作流的名称。
workflowMode	串	工作流的类型来使用。可以是“活”或“草稿”。

例

```
{
  "value": "createissue"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON

例

```
{
  "key": "jira.i18n.title",
  "value": "some.title",
  "id": "jira.i18n.title"
}
```

▶ 架构

状态400 返回如果请求是无效的。

状态403 如果用户没有管理权限返回

▼ 更新酒店

PUT /rest/api/2/workflow/{id}/properties

更新/新属性添加到一个过渡。试图更新不存在会导致在添加了新的属性的属性。

请求

查询参数

参数	类型	描述
key	串	该属性的名称的补充。
workflowName	串	使用工作流的名称。
workflowMode	串	工作流的类型来使用。可以是“活”或“草稿”。

例

```
{
  "value": "createissue"
}
```

▶ 架构

回复

状态200 - 应用程序/JSON

例

```
{
  "key": "jira.i18n.title",
  "value": "some.title",
  "id": "jira.i18n.title"
}
```

▶ 架构

状态400 返回如果请求是无效的。

状态403 如果用户没有管理权限返回

状态304 返回如果没有变化的要求实际上是由（如更新属性珍惜它已经持有）。

▼ 删除属性

DELETE /rest/api/2/workflow/{id}/properties

从传递的工作流传递的过渡删除属性。它不删除不存在的属性是一个错误。

请求

查询参数

参数	类型	描述
key	串	该属性的名称的补充。
workflowName	串	使用工作流的名称。
workflowMode	串	工作流的类型来使用。可以是“活”或“草稿”。

回复

状态200 - 应用程序/JSON

状态400 返回如果请求是无效的。

状态403 如果用户没有管理权限返回

状态304 返回如果没有变化的要求实际上是由（例如试图删除一个不存在的属性）。

▼ 获取属性

GET /rest/api/2/workflow/{id}/properties

返回与转型相关的各种属性或属性。

请求

查询参数

参数	类型	描述
includeReservedKeys	布尔	根据某些键“JIRA”。前缀是可编辑的，有些则不是。为了将其设置为true，以包括在响应中不可编辑键。

key	串	查询属性键的名称。可不放过查询返回的所有属性。
workflowName	串	使用工作流的名称。
workflowMode	串	工作流的类型来使用。可以是“活”或“草稿”。

回复

状态200 - 应用程序/JSON

例

```
{
  "key": "jira.i18n.title",
  "value": "some.title",
  "id": "jira.i18n.title"
}
```

▶ 架构

▶ 架构

状态403 如果用户没有管理权限返回

API / 2 / workflowscheme 隐藏所有方法

▼ 创建方案

POST /rest/api/2/workflowscheme

创建一个新的工作流程方案。

正文包含新方案的表示。假定不传递值被设置为默认值。

请求

例

```
{
  "name": "New Workflow Scheme Name",
  "description": "New Workflow Scheme Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId": "WorkflowName"
  }
}
```

▶ 架构

回复

状态201 新创建的方案。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": [
    {"IssueTypeId2": "WorkflowName",
     "IssueTypeId": "WorkflowName"
    },
    {"draft": false,
     "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
    }
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

▼ 按id获取

GET /rest/api/2/workflowscheme/{id}

返回请求的工作流程方案给调用者。

请求

查询参数

参数	类型	描述
returnDraftIfExists	布尔	真正当指示计划的草案, 如果存在, 应该进行查询, 而不是计划本身。 默认值: 假

回复

状态200 - 应用程序/JSON 如果该计划存在并且调用者有权查看它返回。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果请求的方案不存在, 则返回。

▼ 删除方案

DELETE /rest/api/2/workflowscheme/{id}

删除通过工作流程方案。

回复

状态400 返回如果请求的方案是积极的(即正在使用的JIRA)。

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态204 如果该计划已被删除。

状态404 如果请求的方案不存在, 则返回。

▼ 更新

PUT /rest/api/2/workflowscheme/{id}

更新通过工作流程方案。

的请求的主体是工作流方案的表示。假定不通过值来指示该领域没有任何变化。

传递表示可以有其updateDraftIfNeeded标志设置为true, 表示该草案应创建和/或更新时, 实际的方案不能进行编辑(例如, 当正在使用的项目中的方案)。未出现身体的值将不会被改变。

请求

例

```
{
  "id": 57585,
  "name": "Updated Workflow Scheme Name",
  "description": "Updated Workflow Scheme Name",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId": "WorkflowName"
  },
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 更新后的方案。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果请求的方案不存在, 则返回。

▼ 创建父草案

POST /rest/api/2/workflowscheme/{id}/createdraft

创建传递方案的草案。该草案将成为母公司的状态的副本。

回复

状态201 新创建的方案。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    }
  },
  "applicationRoles": {
    "size": 1,
    "items": []
  }
},
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

▼ 默认情况下删除

DELETE /rest/api/2/workflowscheme/{id}/default

取下通过工作流方案的默认工作流程。

请求

查询参数

参数	类型	描述
updateDraftIfNeeded	布尔	当真正将创建并返回一个草案时工作流程方案不能进行编辑（例如当其正在使用的一个项目）。

回复

状态200 返回成功。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果该方案不存在，则返回。

▼ 默认更新

PUT /rest/api/2/workflowscheme/{id}/default

设置传递的工作流程方案的默认工作流程。

传递表示可以有其updateDraftIfNeeded标志设置为true，表示该草案应创建/更新时的实际方案不能进行编辑。

请求

例

```
{
  "workflow": "WorkflowName",
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 返回成功。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果该方案不存在，则返回。

▼ 获取默认

GET /rest/api/2/workflowscheme/{id}/default

从工作流程传递计划返回默认的工作流程。

请求

查询参数

参数	类型	描述
returnDraftIfExists	布尔	真正当指示计划的草案，如果存在，应该进行查询，而不是计划本身。 默认值：假

回复

状态200

例

```
{
  "workflow": "WorkflowName"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 当工作流方案不存在，则返回。

▼ 获取由ID草案

GET /rest/api/2/workflowscheme/{id}/draft

返回请求的工作流程计划草案给调用者。

回复

状态200 - 应用程序/JSON 如果该计划存在并且调用者有权查看它返回。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    }
  },
  "applicationRoles": {
    "size": 1,
    "items": []
  }
},
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果请求的计划草案不存在返回。

▼ 通过id删除草案

DELETE /rest/api/2/workflowscheme/{id}/draft

删除通过工作流计划草案。

回复

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态204 如果该计划已被删除。

状态404 如果请求的计划草案或父母方案不存在，则返回。

▼ 更新草案

PUT /rest/api/2/workflowscheme/{id}/draft

更新工作流计划草案。如有必要，该草案将创建。

身是工作流方案的表示。假定不通过值来指示该领域没有任何变化。

请求

例

```
{
  "id": 57585,
  "name": "Updated Workflow Scheme Name",
  "description": "Updated Workflow Scheme Name",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId": "WorkflowName"
  },
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 更新/创建方案。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    }
  },
  "applicationRoles": {
    "size": 1,
    "items": []
  }
},
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果请求的方案不存在，则返回。

▼ 获取默认的草案

GET /rest/api/2/workflowscheme/{id}/draft/default

从工作流程传递计划草案给调用者返回默认的工作流程。

回复

状态200

例

```
{
  "workflow": "WorkflowName"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 当工作流方案不存在，则返回。

▼ 删除默认的草案

DELETE /rest/api/2/workflowscheme/{id}/draft/default

取下通过工作流程草案方案的默认工作流程。

回复

状态200 返回成功。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    },
    "applicationRoles": {
      "size": 1,
      "items": []
    }
  },
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果该方案不存在，则返回。

▼ 更新默认草案

PUT /rest/api/2/workflowscheme/{id}/draft/default

设置传递的工作流程草案方案的默认工作流程。

请求

例

```
{
  "workflow": "WorkflowName",
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 返回成功。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    },
    "applicationRoles": {
      "size": 1,
      "items": []
    }
  },
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果该方案不存在, 则返回。

▼ 获取问题类型草案

GET /rest/api/2/workflowscheme/{id}/draft/issuetype/{issueType}

返回传入的工作流程计划草案的问题类型的映射。

回复

状态200 - 应用程序/JSON 返回成功。

例

```
{
  "issueType": "IssueTypeId",
  "workflow": "WorkflowName"
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果任一要求计划或问题类型不存在返回。

▼ 删除问题草案型

DELETE /rest/api/2/workflowscheme/{id}/draft/issuetype/{issueType}

取下计划草案中规定的问题类型的映射。

回复

状态200 返回成功。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": {
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    },
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    },
    "applicationRoles": {
      "size": 1,
      "items": []
    }
  },
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果任一要求计划或问题类型不存在返回。

▼ 草案设置问题类型

PUT /rest/api/2/workflowscheme/{id}/draft/issuetype/{issueType}

设置传递计划草案的问题类型的映射。

传递表示可以有其updateDraftIfNeeded标志设置为true, 表示该草案应创建/更新时的实际方案不能进行编辑。

请求

例

```
{
  "issueType": "IssueTypeId",
  "workflow": "WorkflowName",
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 返回成功。

例

```
{
  "id": 17218781,
  "name": "Workflow Scheme Two",
  "description": "Workflow Scheme Two Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "originalDefaultWorkflow": "ParentsDefaultWorkflowName",
  "originalIssueTypeMappings": {
    "IssueTypeId": "WorkflowName2"
  },
  "draft": true,
  "lastModifiedUser": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "emailAddress": "fred@example.com",
    "avatarUrls": [
      "48x48": "http://www.example.com/jira/secure/useravatar?size=large&ownerId=fred",
      "24x24": "http://www.example.com/jira/secure/useravatar?size=small&ownerId=fred",
      "16x16": "http://www.example.com/jira/secure/useravatar?size=xsmall&ownerId=fred",
      "32x32": "http://www.example.com/jira/secure/useravatar?size=medium&ownerId=fred"
    ],
    "displayName": "Fred F. User",
    "active": true,
    "timeZone": "Australia/Sydney",
    "groups": {
      "size": 3,
      "items": []
    },
    "applicationRoles": {
      "size": 1,
      "items": []
    }
  },
  "lastModified": "Today 12:45",
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/17218781/draft"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果任一要求计划或问题类型不存在返回。

获取工作流程草案

GET /rest/api/2/workflowscheme/{id}/draft/workflow

返回工作流程映射或请求的映射给调用者草案。

请求

查询参数

参数	类型	描述
workflowName	串	工作流程映射返回。可以传递null返回所有的映射。必须是一个有效的工作流程的名称。

回复

状态200 - 应用程序/JSON 返回成功。

例

```
{
  "workflow": "WorkflowName",
  "issueTypes": [
    "IssueTypeId",
    "IssueTypeId2"
  ],
  "defaultMapping": false
}
```

▶ 架构

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

状态404 如果任一要求计划或工作流程不存在返回。

▼ 删除工作流程映射草案

DELETE /rest/api/2/workflowscheme/{id}/draft/workflow

删除从工作流程计划草案的通过工作流程。

请求

查询参数

参数	类型	描述
workflowName	串	删除的工作流的名称。

回复

状态200 删除了该流程的方案。

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

状态404 如果所请求的方案或工作流不存在退货。

▼ 更新工作流程映射草案

PUT /rest/api/2/workflowscheme/{id}/draft/workflow

更新计划草案, 包括通过映射。

身是工作流映射的表示。假定不通过值来指示该领域没有任何变化。

请求

查询参数

参数	类型	描述
workflowName	串	工作流映射的名称来更新。

例

```
{
  "workflow": "WorkflowName3",
  "issueTypes": [
    "IssueTypeId"
  ],
  "updateDraftIfNeeded": true
}
```

▶ 架构

回复

状态200 更新后的方案。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户, 或者如果用户没有输入一个websudo会话。

▼ 获取问题类型

GET /rest/api/2/workflowscheme/{id}/issuetype/{issueType}

返回传入的工作流程方案的问题类型的映射。

请求

查询参数

参数	类型	描述
returnDraftIfExists	布尔	真正当指示计划的草案，如果存在，应该进行查询，而不是计划本身。 默认值：假

回复

状态200 - 应用程序/JSON 返回成功。

例

```
{
  "issueType": "IssueTypeId",
  "workflow": "WorkflowName"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果任一要求计划或问题类型不存在返回。

▼ 删除问题类型

DELETE /rest/api/2/workflowscheme/{id}/issuetype/{issueType}

取下方案指定的问题类型的映射。

请求

查询参数

参数	类型	描述
updateDraftIfNeeded	布尔	当真正将创建并返回一个草案时工作流程方案不能进行编辑（例如当其正在使用的一个项目）。

回复

状态200 返回成功。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": [
    {
      "issueTypeId2": "WorkflowName",
      "issueTypeId": "WorkflowName"
    }
  ],
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果任一要求计划或问题类型不存在返回。

▼ 设置问题类型

PUT /rest/api/2/workflowscheme/{id}/issuetype/{issueType}

设置传递方案中的问题类型的映射。

传递表示可以有其**updateDraftIfNeeded**标志设置为**true**，表示该草案应创建/更新时的实际方案不能进行编辑。

请求

例

```
{
  "issueType": "IssueTypeId",
  "workflow": "WorkflowName",
  "updateDraftIfNeeded": false
}
```

▶ 架构

回复

状态200 返回成功。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": {
    "IssueTypeId2": "WorkflowName",
    "IssueTypeId": "WorkflowName"
  },
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果任一要求计划或问题类型不存在返回。

▼ 获取流程

GET /rest/api/2/workflowscheme/{id}/workflow

返回给调用者的传递计划工作流程映射或请求的映射。

请求

查询参数

参数	类型	描述
workflowName	串	工作流程映射返回。可以传递null返回所有的映射。必须是一个有效的工作流程的名称。
returnDraftIfExists	布尔	真正当指示计划的草案，如果存在，应该进行查询，而不是计划本身。 默认值：假

回复

状态200 - 应用程序/JSON 返回成功。

例

```
{
  "workflow": "WorkflowName",
  "issueTypes": [
    "IssueTypeId",
    "IssueTypeId2"
  ],
  "defaultMapping": false
}
```

▶ 架构

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果任一要求计划或工作流程不存在返回。

▼ 删除工作流程映射

DELETE /rest/api/2/workflowscheme/{id}/workflow

删除从工作流程方案通过工作流程。

请求

查询参数

参数	类型	描述
workflowName	串	删除的工作流的名称。
updateDraftIfNeeded	布尔	标志指示是否草案应创建如果需要删除从该计划的工作流程。

回复

状态200 删除了该流程的方案。**状态401** 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。**状态404** 如果所请求的方案或工作流不存在退货。

▼ 更新工作流程映射

PUT /rest/api/2/workflowscheme/{id}/workflow

更新计划，包括通过映射。

身是工作流映射的表示。假定不通过值来指示该领域没有任何变化。

传递表示可以有其**updateDraftIfNeeded**标志设置为true，表示该草案应创建/更新时的实际方案不能进行编辑。

请求

查询参数

参数	类型	描述
workflowName	串	工作流映射的名称来更新。

例

```
{
  "workflow": "WorkflowName3",
  "issueTypes": [
    "IssueTypeId"
  ],
  "updateDraftIfNeeded": true
}
```

▶ 架构

回复

状态200 更新后的方案。

例

```
{
  "id": 101010,
  "name": "Workflow Scheme One",
  "description": "Workflow Scheme One Description",
  "defaultWorkflow": "DefaultWorkflowName",
  "issueTypeMappings": [
    {
      "IssueTypeId2": "WorkflowName",
      "IssueTypeId": "WorkflowName"
    }
  ],
  "draft": false,
  "self": "http://www.example.com/jira/rest/api/2/workflowscheme/101010"
}
```

▶ 架构

状态401 返回;如果不存在用户，或者如果用户没有输入一个websudo会话。

API / 2 / 工作日志 隐藏所有方法

▼ 获取**worklogs**的**IDS**, 因为删除

GET /rest/api/2/worklog/deleted

返回**worklogs** ID，然后删除自定时间被删除**worklogs**的时间。设置**worklogs**的回报率被限制在1000个元素。这个API将不会返回最后一分钟时被删除**worklogs**。

请求

查询参数

参数	类型	描述
since	长	日期时间在Unix时间戳格式从何时起将返回删除 worklogs 。 默认值: 0

回复

状态200 - 应用程序/JSON 返回的工作日志变化的JSON表示。

例

```
{
  "values": [
    {
      "worklogId": 103,
      "updatedTime": 1438013671562
    },
    {
      "worklogId": 104,
      "updatedTime": 1438013672165
    },
    {
      "worklogId": 105,
      "updatedTime": 1438013693136
    }
  ],
  "since": 1438013671562,
  "until": 1438013693136,
  "self": "http://www.example.com/jira/worklog/updated/updated/deleted?since=1438013671136&since=1438013693136&since=1438013671136",
  "nextPage": "http://www.example.com/jira/worklog/updated/updated/deleted?since=1438013671136&since=1438013693136&since=1438013671136",
  "lastPage": true
}
```

▶ 架构

▼ 获取IDS worklogs

POST /rest/api/2/worklog/list

返回worklogs对于给定的工作日志的ID。仅worklogs到主叫用户有权限，将包含在结果中。设置worklogs的回报率被限制在1000个元素。

请求

例

```
{
  "ids": [
    1,
    2,
    5,
    10
  ]
}
```

▶ 架构

回复

状态200 - 应用程序/JSON 返回搜索结果的JSON表示。

例

```
{
  "self": "http://www.example.com/jira/rest/api/2/issue/10010/worklog/10000",
  "author": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "updateAuthor": {
    "self": "http://www.example.com/jira/rest/api/2/user?username=fred",
    "name": "fred",
    "displayName": "Fred F. User",
    "active": false
  },
  "comment": "I did some work here.",
  "updated": "2016-09-06T09:48:12.832+0000",
  "visibility": {
    "type": "group",
    "value": "jira-developers"
  },
  "started": "2016-09-06T09:48:12.831+0000",
  "timeSpent": "3h 20m",
  "timeSpentSeconds": 12000,
  "id": "100028",
  "issueId": "10002"
}
```

▶ 架构

状态400 返回如果请求包含超过1000个ID或为null

▼ 获取worklogs的IDS，因为修改

GET /rest/api/2/worklog/updated

返回worklogs worklogs的ID和更新的时间，因为给定的时间被更新。设置worklogs的回报率被限制在1000个元素。这个API将不会返回最后一分钟时更新worklogs。

请求

查询参数

参数	类型	描述
----	----	----

since 长 日期时间在Unix时间戳格式从何时起将返回更新worklogs。
默认值：0

回复

状态200 - 应用程序/JSON 返回的工作日志变化的JSON表示。

例

```
{
  "values": [
    {
      "worklogId": 103,
      "updatedTime": 1438013671562
    },
    {
      "worklogId": 104,
      "updatedTime": 1438013672165
    },
    {
      "worklogId": 105,
      "updatedTime": 1438013693136
    }
  ],
  "since": 1438013671562,
  "until": 1438013693136,
  "self": "http://www.example.com/jira/worklog/updated?since=1438013671136",
  "nextPage": "http://www.example.com/jira/worklog/updated/updated?since=1438013671136&since=1438013693136",
  "lastPage": true
}
```

▶ 架构

认证/1/会话 [隐藏所有方法](#)

实现一个REST资源获取会话cookie。

▼ 登录

POST /rest/auth/1/session

创建在JIRA用户一个新的会话。一旦会话已成功创建可用于通过传递相应的HTTP Cookie头访问任何JIRA的遥控器的API，也是web用户界面。

要注意的是通常最好使使用HTTP与REST API BASIC验证。然而，该资源可以用于模仿JIRA的登录页面的行为（例如，以显示登录在错误的用户）。

请求

例

```
{
  "username": "fred",
  "password": "freds_password"
}
```

▶ 架构

回复

状态200 如果调用者进行身份验证返回有关调用者的会话信息。

例

```
{
  "session": {
    "name": "JSESSIONID",
    "value": "12345678901234567890"
  },
  "loginInfo": {
    "failedLoginCount": 10,
    "loginCount": 127,
    "lastFailedLoginTime": "2016-09-06T09:48:10.855+0000",
    "previousLoginTime": "2016-09-06T09:48:10.855+0000"
  }
}
```

请注意，响应包含Set-Cookie的HTTP头必须由调用者兑现。如果您使用的cookie识别HTTP客户端然后它会处理所有Set-Cookie自动头。因为设置这是重要的JSESSIONID单独的cookie可能不足以用于认证工作。

▶ 架构

状态401 如果登录由于无效的凭据失败时返回。

状态403 如果登录被拒绝因CAPTCHA要求，throttling，或任何其他理由退换。在403状态码的情况下可能的是所提供的凭证是有效的，但不允许用户在这个时间点登陆。

▼ 登出

DELETE /rest/auth/1/session

注销当前用户从JIRA的，破坏现有的会话，如果有的话。

回复

状态401 返回如果调用者未经过身份验证。

状态204 返回如果用户已成功注销。

▼ 当前用户

GET /rest/auth/1/session

返回有关当前已验证用户的会话信息。如果来电者未经过身份验证，他们会得到一个**401 Unauthorized**状态码。

回复

状态200

例

```
{  
    "self": "http://www.example.com/jira/rest/api/2.0/user/fred",  
    "name": "fred",  
    "loginInfo": {  
        "failedLoginCount": 10,  
        "loginCount": 127,  
        "lastFailedLoginTime": "2016-09-06T09:48:10.855+0000",  
        "previousLoginTime": "2016-09-06T09:48:10.855+0000"  
    }  
}
```

▶ 架构

状态401 返回如果调用者未经过身份验证。

认证/1/websudo

[隐藏所有方法](#)

▼ 发布

DELETE /rest/auth/1/websudo

这种方法的无效任何当前WebSudo会话。

回复

状态204 如果没有发生错误返回