CloudStudio API Reference

Introduction

The CloudStudio API exposes an interface to access and manipulate CloudStudio resources. All CloudStudio resources are accessed and manipulated in a similar way.

Requests to the CloudStudio API have to use either the GET or POST method. GET requests are used for functions that do not change the state of the database. POST requests are used for functions that make changes to the database.

The content type of requests to the CloudStudio API must be `application/x-www-form-urlencoded`. The response has content type `application/json`. This asynchronism allows us to provide parameters for both GET and POST requests similarly and still retrieve comprehensive JSON objects, and is used by many widely used APIs (e.g. SoundCloud).

/api/login

Method: POST

Log into CloudStudio with your username and password. Returns a session ID that will be required for further API calls, as well as the username and user privileges.

Parameters

Parameter name	Description
username	Your username
password	Your password

Example

Request

```
curl "http://cloudstudio:7330/api/login" \
  -d "username=John" \
  -d "password=burgers"
```

Response

```
{
   "sessionId": "f40309335f82e044fa04c6f267aa62fd",
   "username": "John",
   "isAdmin": false,
   "isCreator": false
}
```

/api/repositories

Method: GET

Retrieves a list of all repositories you have access to.

Parameters

Parameter name	Description
sessionId	Your session ID

Example

Request

```
curl "http://cloudstudio:7330/api/repositories?\
  sessionId=YOUR_SESSION_ID"
```

Response

```
{
   "repositories": [
      {
         "repositoryAlias": "BankAccountDemo",
         "repositoryDescription": "Dealing with banks and accounts.",
         "repositoryUrl": "https://github.com/foo/bankaccountdemo",
         "repositoryOwner": "John",
         "users": [
            "David",
            "Isabelle",
            "John"
         ],
      },
      {
         (...)
      }
   ]
}
```

/api/repositoryInformation

Method: GET

Retrieves a list of users and branches for a given repository. Also returns the timestamp of the last origin information update.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias

Example

Request

```
curl "http://localhost:7330/api/repositoryInformation?\
  sessionId=YOUR_SESSION_ID&\
  repositoryAlias=BankAccountDemo"
```

Response

```
{
   "repositoryAlias": "BankAccountDemo",
   "repositoryDescription": "Dealing with banks and accounts.",
   "repositoryUrl": "https://github.com/foo/bankaccountdemo",
   "repositoryUsers": [
      "David",
      "Isabelle",
      "John"
   ],
   "repositoryBranches": [
      "master",
      "test_branch"
   ],
   "lastOriginUpdate": "2015-03-27 23:09:32",
   "lastOriginUpdateDiff": "1m"
}
```

/api/setRepositoryInformation

Method: POST

Updates repository information. You need to be administrator or repository owner.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
repositoryDescription	Description for repository
repositoryUrl	URL to remote repository

Example

Request

```
curl "http://cloudstudio:7330/api/setRepositoryInformation" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld" \
  -d "repositoryDescription=This is a Hello World project." \
  -d "repositoryUrl=https://github.com/foo/helloworld"
```

Response

```
{}
```

/api/branchAwareness

Method: GET

Retrieves branch level awareness information for a repository. For every branch, the active users represent the users that have this particular branch checked out currently.

For every user in branch, a relation to the origin is given. This value can be EQUAL, AHEAD, BEHIND, FORK, LOCAL_BRANCH or REMOTE_BRANCH.

Relationship with origin	Description
EQUAL	The latest branch commit is the same for the user and the origin.
AHEAD	The user has made commits and is directly ahead of the origin.
BEHIND	New commits have been pushed to the origin and the user is directly behind.
FORK	The user has made commits but other new commits have been pushed to the origin in the meantime.
LOCAL_BRANCH	This branch is a local branch for the user.
REMOTE_BRANCH	This branch only exists on the remote but not on the user's local repository.

For the relationships AHEAD, BEHIND and FORK a distance specifies the shortest distance between the current commit for the client and the origin.

For every user, the "lastUpdate" field refers to the last time that the client has sent an update to CloudStudio. "lastUpdateDiff" is the elapsed time since the last update, e.g. "2h" (2 hours) or "5d" (5 days).

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias

Example

Request

```
curl "http://localhost:7330/api/branchAwareness?\
  sessionId=YOUR_SESSION_ID&\
  repositoryAlias=BankAccountDemo"
```

Response

```
{
   "branches": [
      {
         "branch": "master",
         "activeUsers": [
            {
               "username": "David",
               "lastUpdate": "2015-03-27 20:10:03",
               "lastUpdateDiff": "3h"
            },
            {
               "username": "John",
               "lastUpdate": "2015-03-27 21:33:41",
               "lastUpdateDiff": "2h"
            }
         ],
         "users":[
            {
               "username": "David",
               "relationWithOrigin": "FORK",
               "distanceFromOrigin": 2
            },
            {
               "username": "Isabelle",
               "relationWithOrigin": "EQUAL"
            },
            {
               "username": "John",
               "relationWithOrigin": "AHEAD",
               "distanceFromOrigin": 1
            }
         ]
      },
      {
         (...)
      }
   ]
}
```

/api/fileAwareness

Method: GET

Retrieves file level awareness information for a repository and branch. All your files in a branch are compared to every other users' files in the same or specified branch.

For every user a conflict type is set to either NO_CONFLICT, FILE_CONFLICT or CONTENT_CONFLICT.

Conflict Type	Description
NO_CONFLICT	The two files being compared are identical.
FILE_CONFLICT	The two files being compared are different.
CONTENT_CONFLICT	After further analysing conflicting files, by doing a three-way diff with a suitable common ancestor of both files, a merge conflict occurs.

Non-existing files are treated as empty files for this purpose.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
branch	Branch from which your files are compared
compareToBranch	Branch to which files of other users your files are compared to
showUncommitted	If true, also takes into account changes that have not yet been locally committed.
showConflicts	If true, for all files with a FILE_CONFLICT, additionally run a content conflict analysis. If false, just compare the files by their hash.
viewAsOrigin	Instead of showing from your perspective, show from the perspective of the origin ("true" or "false")

Example

Request

```
curl "http://cloudstudio:7330/api/fileAwareness?\
  sessionId=YOUR_SESSION_ID&\
  repositoryAlias=BankAccountDemo&\
  branch=master&\
  compareToBranch=master&\
  showUncommitted=false&\
  showConflicts=true&\
  viewAsOrigin=false"
```

Response

```
{
   "files": [
      {
         "filename": "README",
         "users": [
            {
               "username": "David",
               "type": "FILE_CONFLICT"
```

```
            },
            {
               "username": "Isabelle",
               "type": "NO_CONFLICT"
            },
            {
               "username": "John",
               "type": "NO_CONFLICT"
            }
         ]
      },
      {
         "filename": "src/java/Main.java",
         "users": [
            {
               "username": "David",
               "type": "NO_CONFLICT"
            },
            {
               "username": "Isabelle",
               "type": "CONTENT_CONFLICT"
            },
            {
               "username": "John",
               "type": "NO_CONFLICT"
            }
         ]
      }
   ]
}
```


/api/contentAwareness

Method: GET

Compares two files directly to each other.

The response contains a line-by-line comparison designed to be easily displayable in a side-by-side view.

For each line, a type is set as follows:

Type	Description
UNCHANGED	No changes have been made to this line.
INSERT	This line has been inserted.
MODIFIED	This line has been modified.
PAD	Padding for unmodified blocks to line up nicely.
MODIFIED_PAD	Padding for modified blocks to line up nicely.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
filename	Filename
branch	Your branch
theirUsername	User you want to compare to
compareToBranch	Branch you want to compare to
showUncommitted	If true, also take into account changes that have not yet been locally committed.
viewAsOrigin	Instead of showing from your perspective, show from the perspective of the origin ("true" or "false")

Example

Request

```
curl "http://cloudstudio:7330/api/contentAwareness?\
  sessionId=YOUR_SESSION_ID&\
  repositoryAlias=BankAccountDemo&\
  filename=README&\
  branch=master&\
  compareToBranch=master&\
  theirUsername=David\
  showUncommitted=false&\
  viewAsOrigin=false"
```

Response

```
{
   "content": [
      {
         "myContent": "Welcome to BankAccountDemo!",
         "myType": "UNCHANGED",
         "theirContent": "Welcome to BankAccountDemo!",
         "theirType": "UNCHANGED"
      },
      {
         "myContent": "",
         "myType": "PAD",
         "theirContent": "This is a project dealing with banks and accounts."
         "theirType": "INSERT"
      }
   ]
}
```

/api/contentConflict

Method: GET

Compares two files and the nearest common ancestor to each other.

The response contains a line-by-line comparison designed to be easily displayable in a side-by-side view.

For each line, a type is set as follows:

Type	Description
UNCHANGED	No changes have been made to this line.
MODIFIED	This line has been modified.
MODIFIED_PAD	Padding for modified blocks to line up nicely.
CONFLICT	This line is conflicting.
CONFLICT_PAD	Padding for conflict blocks to line up nicely.
PAD	Padding for the blocks to line up nicely.

By definition, a conflict occurs when all three lines have been modified or only the common ancestor has been modified.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
filename	Filename
branch	Your branch
theirUsername	User you want to compare to
compareToBranch	Branch you want to compare to
showUncommitted	If true, also take into account changes that have not been locally committed yet.
viewAsOrigin	Instead of showing from your perspective, show from the perspective of the origin ("true" or "false")

Example

Request

```
curl "http://cloudstudio:7330/api/contentAwareness?\
  sessionId=YOUR_SESSION_ID&\
  repositoryAlias=BankAccountDemo&\
  filename=src/java/Main.java&\
  branch=master&\
  compareToBranch=master&\
  theirUsername=Isabelle&\
  showUncommitted=false&\
  viewAsOrigin=false"
```

Response

```
{
   "content":[
      {
         "myType": "UNCHANGED",
         "myContent": "First line.",
         "theirType": "UNCHANGED",
         "theirContent": "First line.",
         "baseType": "UNCHANGED",
         "baseContent": "First line."
      },
      {
         "myType": "MODIFIED",
         "myContent": "Only I changed this, no worries.",
         "theirType": "MODIFIED",
         "theirContent": "Second line.",
         "baseType": "MODIFIED",
         "baseContent": "Second line."
      },
      {
         "myType": "CONFLICT",
         "myContent": "I made a change.",
         "theirType": "CONFLICT",
         "theirContent": "Third line.",
         "baseType": "CONFLICT",
         "baseContent": "Me too! Whoops."
      }
   ]
}
```

/api/users

Method: GET

Retrieves a list of all users, their privileges and the date they created the account. Must be administrator to perform this operation.

Parameters

Parameter name	Description
sessionId	Your session ID

Example

Request

```
curl "http://cloudstudio:7330/api/users?\
  sessionId=YOUR_SESSION_ID"
```

Response

```
{
   "users": [
      {
         "username": "Admin",
         "joinDate": "2015-02-01 16:23:12",
```

```
            "isAdmin": true,
            "isCreator": true,
        },
        {
            "username": "David",
            "joinDate": "2015-03-02 23:01:57",
            "isAdmin": false,
            "isCreator": true,
        },
        {
            "username": "Isabelle",
            "joinDate": "2015-03-07 11:23:01",
            "isAdmin": false,
            "isCreator": false,
        },
        {
            "username": "John",
            "joinDate": "2015-03-16 10:13:41",
            "isAdmin": false,
            "isCreator": true,
        },
    ]
}
```


/api/createRepository

Method: POST

Creates a new repository and sets its owner to yourself. Repository creation rights are required for this operation.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
repositoryUrl	URL to the remote, e.g. GitHub
repositoryDescription	A short description

Example

Request

```
curl "http://cloudstudio:7330/api/createRepository" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld" \
  -d "repositoryDescription=This is a Hello World project." \
  -d "repositoryUrl=https://github.com/foo/helloworld"
```

Response

```
{}
```

/api/deleteRepository

Method: POST

Deletes a repository. You need to be administrator or the repository owner for this operation.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias

Example

Request

```
curl "http://cloudstudio:7330/api/deleteRepository" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld"
```

Response

```
{}
```

/api/addUserToRepository

Method: POST

Adds a user to a repository. Must be repository owner or administrator.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/addUserToRepository" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld" \
```

```
  -d "username=David"
```

Response

```
{}
```

/api/removeUserFromRepository

Method: POST

Removes a user from a repository. Must be repository owner or administrator.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/removeUserFromRepository" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld" \
  -d "username=David"
```

Response

```
{}
```

/api/modifyRepositoryOwner

Method: POST

Sets a new repository owner. Must be repository owner or administrator.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	New repository owner

Example

Request

```
curl "http://cloudstudio:7330/api/modifyRepositoryOwner" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "repositoryAlias=HelloWorld" \
  -d "username=David"
```

Response

```
{}
```

/api/createUser

Method: POST

Creates a new user.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username
password	New password

Example

Request

```
curl "http://cloudstudio:7330/api/createUser" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "username=David" \
  -d "password=penguins"
```

Response

```
{}
```

/api/deleteUser

Method: POST

Removes a user. Requires administrator privileges.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/deleteUser" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "username=David"
```

Response

```
{}
```

/api/deleteUser

Method: POST

Changes a user's password.

Parameters

Parameter name	Description
sessionId	Your session ID
newPassword	New password

Example

Request

```
curl "http://cloudstudio:7330/api/changePassword" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "newPassword=polarbears"
```

Response

```
{}
```

/api/giveAdminPrivileges

Method: POST

Give administrator privileges to a user. Requires administrator privileges.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/giveAdminPrivileges" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "username=David"
```

Response

```
{}
```

/api/revokeAdminPrivileges

Method: POST

Revoke a user's administrator privileges. Requires administrator privileges.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/revokeAdminPrivileges" \
  -d "sessionId=YOUR_SESSION_ID" \
```

```
  -d "username=David"
```

Response

```
{}
```

/api/giveCreatorPrivileges

Method: POST

Give repository creation privileges to a user. Requires administrator privileges.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/giveCreatorPrivileges" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "username=David"
```

Response

```
{}
```

/api/revokeCreatorPrivileges

Method: POST

Revoke a user's repository creation privileges. Requires administrator privileges.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias
username	Username

Example

Request

```
curl "http://cloudstudio:7330/api/revokeCreatorPrivileges" \
  -d "sessionId=YOUR_SESSION_ID" \
  -d "username=David"
```

Response

```
{}
```

/api/localState

Method: POST

Update the server with the user's git repository information for a single repository. This operation is used by the client periodically.

For this operation only, repository data needs to be sent as `application/json`.

Parameters

Parameter name	Description
sessionId	Your session ID
repositoryAlias	Repository alias

Example

Request

```
curl "http://cloudstudio:7330/api/localState?sessionId=YOUR_SESSION_ID&reposi
  -H "Content-Type: application/json" \
  -d "$JSON_STRING"
```

With the `$JSON_STRING` being:

```
{
   "files": [
      {
         "filename": "README",
         "branch": "master",
         "content": "This is the read-me file.",
         "committed": "committed",
         "commit": "65fcfcd7860bf95fd1dce7c01bcd886bcdf4e675"
      },
      {
         "filename": "README",
         "branch": "master",
         "content": "I made some uncommitted changed to the read-me file.",
         "committed": "uncommitted",
         "commit": "65fcfcd7860bf95fd1dce7c01bcd886bcdf4e675"
```

```
      }
   ],
   "branches": [
      {
         "commit": "73e68dd8ae12bdf7dfce4a29cd0a6cb6ce99aca8",
         "active": true,
         "branch": "master"
      }
   ],
   "commitHistory": [
      {
         "commit": "73e68dd8ae12bdf7dfce4a29cd0a6cb6ce99aca8",
         "downstreamCommits":[
            {
               "distance": 0,
               "commit": "73e68dd8ae12bdf7dfce4a29cd0a6cb6ce99aca8"
            },
            {
               "distance": 1,
               "commit": "fdaa47da4ab7f22fc06373c407c48326e70db199"
            }
         ]
      }
   ]
}
```

Response

```
{}
```

Error handling

An erroneous request results in a status code 400 (Bad Request) response. The response data is a JSON object as always and contains and error message.

Example

Response

```
{
   "error": "Insufficient privileges"
}
```