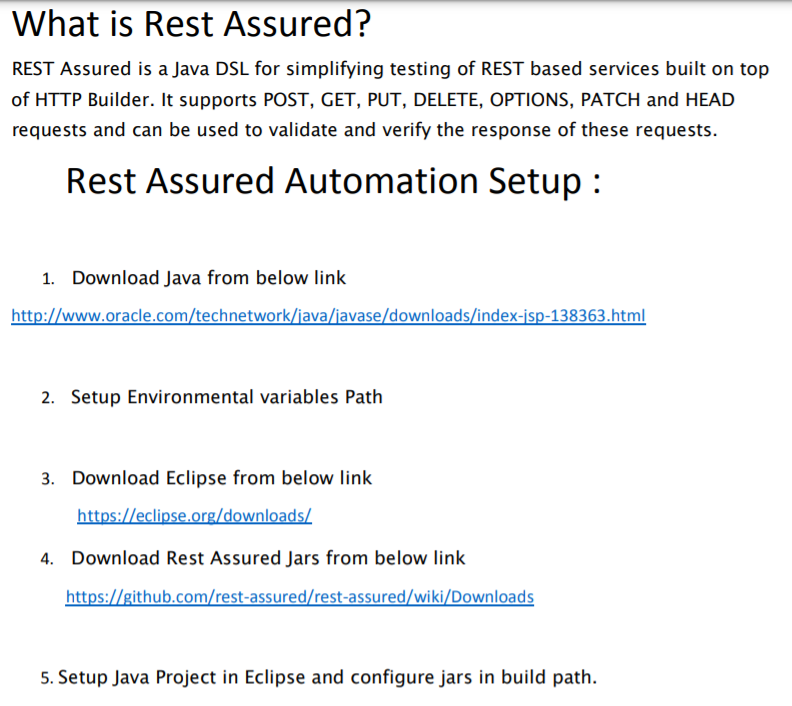
RestAssuredAPITestingNotes



Endpoint:AddresswhereAPIishostedontheServer.

HTTPmethodswhicharecommonlyusedtocommunicatewithRestAPI’sare

**GET,POST,PUT,andDELETE**

GET-TheGETmethodisusedtoextractinformationfromthegivenserverusingagivenURI.WhileusingGETrequest,itshouldonlyextractdataandshouldhavenoothereffectonthedata.NoPayload/Bodyrequired

**HowtosendinputdatainGET?**  
Ans:UsingQueryParameters

POST-APOSTrequestisusedtosenddatatotheserver,forexample,customerinformation,fileupload,etc.usingHTMLforms.

**HowtosendinputdatainPOST?**  
Ans:UsingFormParameters/BodyPayload

PUT-Replacesallcurrentrepresentationsofthetargetresourcewiththeuploadedcontent.

DELETE-RemovesallcurrentrepresentationsofthetargetresourcegivenbyaURI.

**Resources:  
ResourcesrepresentAPI/CollectionwhichcanbeaccessedfromtheServer**

Google.com/maps  
google.com/search  
google.com/images

**PathParameters:**  
***Pathparameters***arevariablepartsofaURLpath.Theyaretypicallyusedtopointtoaspecificresourcewithinacollection,suchasauseridentifiedbyID

<https://www.google.com/Images/1123343>  
<https://www.google.com/docs/1123343>  
<https://amazon.com/orders/112>

<https://www.google.com/search?q=newyork&oq=newyork&aqs=chrome..69i57j0l7.2501j0j7&sourceid=chrome&ie=UTF-8>

**QueryParameters:**  
QueryParameterisusedtosort/filtertheresources.

QueryParametersareidentifiedwith?””

https://amazon.com/orders?sort\_by=2/20/2020

**Headers/Cookies**:

Headersrepresentthemeta-dataassociatedwiththeAPIrequestandresponse.Inlaymanterms,weweresendingAdditionaldetailstoAPItoprocessourrequest.  
Example:Authorizationdetails

**EndPointRequestURLcanbeconstructedasbelow**  
BaseURL/resource/(Query/Path)Parameters

GoogleMapsAddAPI(POST):

ThisAPIWilladdnewplaceintoServer

**CompleteURL:**[https://rahulshettyacademy.com](https://rahulshettyacademy.com/)**/maps/api/place/add/json?key=qaclick123**

**BaseURL**:[https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

**Resource**:/maps/api/place/add/json

**QueryParameters**:key=qaclick123

**HttpMethod**:POST

**SampleBody**:

{

"location":{

"lat":-38.383494,

"lng":33.427362

},

"accuracy":50,

"name":"Frontlinehouse",

"phone\_number":"(+91)9838933937",

"address":"29,sidelayout,cohen09",

"types":[

"shoepark",

"shop"

],

"website":"http://google.com",

"language":"French-IN"

}

**SampleResponse**

{

"status":"OK",

"place\_id":"928b51f64aed18713b0d164d9be8d67f",

"scope":"APP",

"reference":"736f3c9bec384af62a184a1936d42bb0736f3c9bec384af62a184a1936d42bb0",

"id":"736f3c9bec384af62a184a1936d42bb0"

}

GoogleMapsDeleteAPI(POST):

ThisAPIWilldeleteexistingplacefromServer

**CompleteURL**:https://rahulshettyacademy.com/maps/api/place/delete/json?key=qaclick123

**BaseURL**:https://rahulshettyacademy.com

**Resource**:/maps/api/place/delete/json

**QueryParameters:**key

**Httprequest**:POST

**SampleBody**:

{

"place\_id":"928b51f64aed18713b0d164d9be8d67f"//(ThisvaluecomesfromAddplaceresponse)

}

**SampleResponse**

{

"status":"OK"

}

GoogleMapsgetPlaceAPI(GET):

**ThisAPIWillgetexistingplacedetailsfromServer**

**CompleteURL:**http://rahulshettyacademy.com/maps/api/place/get/json?place\_id=xxxx&key=qaclick123

**BaseURL**:[https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

**Resource**:/maps/api/place/get/json

**Parameters**:key,place\_id//(place\_idvaluecomesfromAddplaceresponse)

**Httprequest**:GET

Note:Keyvalueishardcodedanditisalwaysqaclick123

**SampleResponsefortheProvidedPlace\_Id**

{

"location":{

"lat":-38.383494,

"lng":33.427362

},

"accuracy":50,

"name":"Frontlinehouse",

"phone\_number":"(+91)9838933937",

"address":"29,sidelayout,cohen09",

"types":["shoepark","shop"],

"website":"http://google.com",

"language":"French-IN"

}

GoogleMapsPutPlaceAPI(PUT):

ThisAPIWillupdateexistingplaceinServerwithnewvalues

**CompleteURL:**http://rahulshettyacademy.com/maps/api/place/get/json?place\_id=xxxx&key=qaclick123

**BaseURL**:[https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

**Resource**:/maps/api/place/update/json

**QueryParameters**:key,place\_id//(place\_idvaluecomesfromAddplaceresponse)

**HttpMethod**:PUT

Note:Keyvalueishardcodedanditisalwaysqaclick123

**SampleRequest:**

{

"place\_id":"8d2573bdf6ceec0e474c5f388fa917fb",

"address":"70winterwalk,USA",

"key":"qaclick123"

}

**SampleResponsefortheProvidedPlace\_Id**

{

"location":{

"lat":-38.383494,

"lng":33.427362

},

"accuracy":50,

"name":"Frontlinehouse",

"phone\_number":"(+91)9838933937",

"address":"29,sidelayout,cohen09",

"types":["shoepark","shop"],

"website":"http://google.com",

"language":"French-IN"

}



**Library API :**

BaseURI : http://216.10.245.166

1. **Resource** : Library/Addbook.php       **Method** : POST

**Input Payload : Json**:

{

"name":"Learn Appium Automation with Java",

"isbn":"bcd",

"aisle":"227",

"author":"John foe"

}

**Output Json**

{

"Msg": "successfully added",

"ID": "bcd227"

}

1. **Resource** : /Library/GetBook.php?AuthorName=somename **Method** : GET

**Output Json** :

Output the array of Json object books with all below  details

{

Name : “bookname”   ( String)

Isbn :  “A2fdsf”   (String)

Aisle : 32 (Integer)

}

1. **Resource** : Library/GetBook.php?ID=3389      - **Method** : GET

**Output Json :**

{

"book\_name": "Selenium automation using Java",

"isbn": "a23hd738",

"aisle": "1223"

}

1. **Resource** :/Library/DeleteBook.php      **Method** : POST

**Input Payload : Json:**

{

"ID" : "a23h345122332"

}

**Output Response** :

{  
  
msg : book is successfully deleted”

}



JIRA

Download Jira from below link

https://www.atlassian.com/software/jira/download

Authentication API

https://developer.atlassian.com/jiradev/jira-apis/jira-rest-apis/jira-rest-api-tutorials/jira-rest-api-example-cookie-based-authentication

CreateIssue

https://docs.atlassian.com/jira/REST/cloud/#api/2/issue-createIssue



Request and Response Spec Builders :

ADD PLACE :

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

.body(add\_place\_json)

.when().post("/maps/api/place/add/json").

then().assertThat().statusCode(200). contentType("application/json")

extract().response();

GET\_PLACE

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

when (). get("/maps/api/place/get/json").

then().assertThat().statusCode(200).contentType("application/json").extract().response();

DELETE\_PLACE

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

.body(“delete\_Place\_json”)

.when().post("/maps/api/place/delete/json").

then().assertThat().statusCode(200). contentType("application/json").extract().response();

Build -Request Spec Builder-

req= **new** RequestSpecBuilder().setContentType(ContentType.***JSON***)

.setBaseUri("XXXX")

.addQueryParam("key","qaclick123")

.build();

*given*().spec (req ).body(add\_place\_json) .post(“/maps/api/place/add/json).

Build Response Spec Builder:

**res = new** ResponseSpecBuilder().expectStatusCode(200).expectContentType(ContentType.***JSON***). build();

then().spec(re).extract().response();

Rewriting Test with Request and Response Spec Builder :

*given*().spec(req).body(add\_place\_json).post(“/maps/api/place/add/json).

then().spec(res).extract().response();



|  |  |  |
| --- | --- | --- |
| Git task | Notes | Git commands |
| [**Tell Git who you are**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-config) | Configure the author name and email address to be used with your commits.  Note that Git [strips some characters](http://stackoverflow.com/questions/26159274/is-it-possible-to-have-a-trailing-period-in-user-name-in-git/26219423#26219423) (for example trailing periods) from user.name. | git config --global user.name "Sam Smith"  git config --global user.email sam@example.com |
| [**Create a new local repository**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-init) |  | git init |
| [**Check out a repository**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-clone) | Create a working copy of a local repository: | git clone /path/to/repository |
| For a remote server, use: | git clone username@host:/path/to/repository |
| [**Add files**](https://www.atlassian.com/git/tutorials/saving-changes#git-add) | Add one or more files to staging (index): | git add <filename>  git add \* |
| [**Commit**](https://www.atlassian.com/git/tutorials/saving-changes#git-commit) | Commit changes to head (but not yet to the remote repository): | git commit -m "Commit message" |
| Commit any files you've added with git add, and also commit any files you've changed since then: | git commit -a |
| [**Push**](https://www.atlassian.com/git/tutorials/syncing#git-push) | Send changes to the master branch of your remote repository: | git push origin master |
| [**Status**](https://www.atlassian.com/git/tutorials/inspecting-a-repository#git-status) | List the files you've changed and those you still need to add or commit: | git status |
| [**Connect to a remote repository**](https://www.atlassian.com/git/tutorials/syncing#git-remote) | If you haven't connected your local repository to a remote server, add the server to be able to push to it: | git remote add origin <server> |
| List all currently configured remote repositories: | git remote -v |
| [**Branches**](https://www.atlassian.com/git/tutorials/using-branches) | Create a new branch and switch to it: | git checkout -b <branchname> |
| Switch from one branch to another: | git checkout <branchname> |
| List all the branches in your repo, and also tell you what branch you're currently in: | git branch |
| Delete the feature branch: | git branch -d <branchname> |
| Push the branch to your remote repository, so others can use it: | git push origin <branchname> |
| Push all branches to your remote repository: | git push --all origin |
| Delete a branch on your remote repository: | git push origin :<branchname> |
| [**Update from the remote repository**](https://www.atlassian.com/git/tutorials/syncing) | Fetch and merge changes on the remote server to your working directory: | git pull |
| To merge a different branch into your active branch: | git merge <branchname> |
| View all the merge conflicts:  View the conflicts against the base file:  Preview changes, before merging: | git diff  git diff --base <filename>  git diff <sourcebranch> <targetbranch> |
| After you have manually resolved any conflicts, you mark the changed file: | git add <filename> |
| **Tags** | You can use tagging to mark a significant changeset, such as a release: | git tag 1.0.0 <commitID> |
| CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using: | git log |
| Push all tags to remote repository: | git push --tags origin |
| [**Undo local changes**](https://www.atlassian.com/git/tutorials/undoing-changes) | If you mess up, you can replace the changes in your working tree with the last content in head:  Changes already added to the index, as well as new files, will be kept. | git checkout -- <filename> |
| Instead, to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it, do this: | git fetch origin  git reset --hard origin/master |
| **Search** | Search the working directory for foo(): | git grep "foo()" |