Import Execution Results - REST

- Importing Execution Results
  - Xray JSON results
  - Xray JSON results Multipart
  - Cucumber JSON results
  - Cucumber JSON results Multipart
  - Behave JSON results
  - Behave JSON results Multipart
  - JUnit XML results
  - JUnit XML results Multipart
  - TestNG XML results
  - TestNG XML results Multipart
  - NUnit XML results
  - NUnit XML results Multipart
  - xUnit XML results
  - xUnit XML results Multipart
  - Robot Framework XML results
  - Robot Framework XML results Multipart
  - Multiple Execution Results
- How results are mapped to Test entities

Importing Execution Results

Execution results can be imported to Jira through JSON/XML representation formats specified in Import Execution Results.

For each import file format, Xray provides a specific REST endpoint:

Xray JSON format	**/rest/raven/1.0/import/execution**
Xray JSON format multipart	**/rest/raven/1.0/import/execution/multipart**
Cucumber JSON output format	**/rest/raven/1.0/import/execution/cucumber**
Cucumber JSON output format multipart	**/rest/raven/1.0/import/execution/cucumber/multipart**
Behave JSON output format	**/rest/raven/1.0/import/execution/behave**
Behave JSON output format multipart	**/rest/raven/1.0/import/execution/behave/multipart**
JUnit XML output format	**/rest/raven/1.0/import/execution/junit**
JUnit XML output format multipart	**/rest/raven/1.0/import/execution/junit/multipart**
TestNG XML output format	**/rest/raven/1.0/import/execution/testng**
TestNG XML output format multipart	**/rest/raven/1.0/import/execution/testng/multipart**
NUnit XML output format	**/rest/raven/1.0/import/execution/nunit**
NUnit XML output format multipart	**/rest/raven/1.0/import/execution/nunit/multipart**
xUnit XML output format	**/rest/raven/1.0/import/execution/xunit**
xUnit XML output format multipart	**/rest/raven/1.0/import/execution/xunit/multipart**
Robot Framework XML output format	**/rest/raven/1.0/import/execution/robot**
Robot Framework XML output format multipart	**/rest/raven/1.0/import/execution/robot/multipart**
Compressed .zip file (e.g., Calabash execution results)	**/rest/raven/1.0/import/execution/bundle**

```
"summary": "Update existing test",
```

compressed zip *file to import or a* ***JSON file*** *to import.*

Xray JSON results

When importing execution results using Xray JSON result format in a Continuous Integration environment, you can specify which Test Execution issue to import the results on using the "**testExecutionKey**" property. Alternatively, you can create a new Test Execution for the execution results and specify the Test Execution issue fields in the "**info**" object.

Updating an existing Test Run using Xray format REST API will reset all dataset related fields. This means that all current iteration data and dataset present in the Test Run will be replaced with the new information given in the REST API request.

Import the execution results present in query variable "***executionResults***".

Request

Example 1: new Test Execution

Example Input

```
{
    "info" : {
        "project" : "DEMO",
        "summary" : "Execution of automated tests for release v1.3",
                "description" : "This execution is automatically created when importing execution results
from an external source",
                "version" : "v1.3",
                "user" : "admin",
                "revision" : "1.0.42134",
                "startDate" : "2014-08-30T11:47:35+01:00",
                "finishDate" : "2014-08-30T11:53:00+01:00",
                "testPlanKey" : "DEMO-100",
                "testEnvironments": ["iOS", "Android"]
    },
    "tests" : [
        {
            "testKey" : "DEMO-6",
            "start" : "2014-08-30T11:47:35+01:00",
            "finish" : "2014-08-30T11:50:56+01:00",
            "comment" : "Successful execution",
                        "testVersion" : "v2"
            "status" : "PASS"
        },
        {
            "testKey" : "DEMO-7",
            "start" : "2014-08-30T11:51:00+01:00",
            "finish" : "2014-08-30T11:52:30+01:00",
            "comment" : "Execution failed. Example #5 FAIL.",
            "status" : "FAIL",
                        "evidences" : [
                    {
                        "data":
"iVBORw0KGgoAAAANSUhEUgAABkIAAAO9CAYAAADezXv6AAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAAEn(...base64
file enconding)",
                                        "filename": "image21.jpg",
                    "contentType": "image/jpeg"
                }
                        ],
            "examples" : [
                "PASS",
                "PASS",
                "PASS",
                "PASS",
                "FAIL"
            ],
                        "steps": [
                {
                    "status": "PASS",
                    "comment": "Coment on Test Step Result 1",
                    "evidences" : [
                        {
                            "data":
"iVBORw0KGgoAAAANSUhEUgAABkIAAAO9CAYAAADezXv6AAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAAEn(...base64
file enconding)",
                                                        "filename": "image22.jpg",
                        "contentType": "image/jpeg"
                        }
                                    ],
                                    "actualResult": "Actual result on Test Step 1"
                                }
                        ],
                    "defects" : [
                                    "DEMO-10",
                                    "DEMO-11"
                ]
        }
    ]
}
```

Example 2: update existing Test Execution

Example Input

```
{
    "testExecutionKey": "DEMO-1206",
    "info" : {
        "summary" : "Execution of automated tests for release v1.3",
                "description" : "This execution is automatically created when importing execution results
from an external source",
                "version" : "v1.3",
                "user" : "admin",
                "revision" : "1.0.42134",
                "startDate" : "2014-08-30T11:47:35+01:00",
                "finishDate" : "2014-08-30T11:53:00+01:00",
                "testPlanKey" : "DEMO-100",
                "testEnvironments": ["iOS", "Android"]
    },
    "tests" : [
        {
            "testKey" : "DEMO-6",
            "start" : "2014-08-30T11:47:35+01:00",
            "finish" : "2014-08-30T11:50:56+01:00",
            "comment" : "Successful execution",
            "status" : "PASS",
                        "testVersion" : "v2"
        }
     ]
}
```

Example 3: create new Test

Example Input

```
{
    "testExecutionKey" : "DEMO-1206",
    "tests" : [
        {
            "status": "FAIL",
            "steps": [
                {
                    "status": "PASS"
                },
                {
                    "status": "FAIL"
                }
            ],
            "testInfo": {
                "summary": "Create new test",
                "testType": "Manual",
                "projectKey": "DEMO",
                "steps": [
                    {
                        "action": "Step action 1",
                        "data": "Data 1",
                        "result": "Step result 1"
                    },
                    {
                        "action": "Step action 2",
                        "data": "Data 2",
                        "result": "Step result 2"
                    }
                ]
            }
        }
    ]
}
```

Example 4: update existing Test, choosing a specific test version

Example Input

```
{
    "testExecutionKey" : "DEMO-1206",
    "tests" : [
        {
            "status": "FAIL",
                        "testVersion":"v2",
            "steps": [
                {
                    "status": "PASS"
                },
                {
                    "status": "FAIL"
                }
            ],
            "testKey": "DEMO-1207",
            "testInfo": {
                "summary": "Update existing test",
                "testType": "Manual",
                "projectKey": "DEMO",
                "steps": [
                    {
                        "action": "Step action 1",
                        "data": "Data 1",
                        "result": "Step result 1"
                    },
                    {
                        "action": "Step action 2",
                        "data": "Data 2",
                        "result": "Step result 2"
                    }
                ]
            }
        }
    ]
}
```

Example 4: Importing test results with Test Run Custom Fields

Example Input

```
{
    "info": {
        "summary": "Execution of automated tests for release v1.3",
        "description": "This execution is automatically created when importing execution results from an
external source",
        "user": "admin",
        "startDate": "2014-08-30T11:47:35+01:00",
        "finishDate": "2014-08-30T11:53:00+01:00"
    },
        "tests": [
            {
                "testKey": "DEMO-25",
                "start": "2014-08-30T11:47:35+01:00",
                "finish": "2014-08-30T11:50:56+01:00",
                "comment": "Successful execution",
                "status": "FAIL",
                "customFields": [
                    {
                        "id": 1,
                        "value": "true"
                    },
                    {
                        "id": 2,
                        "value": "13400"
                    },
                    {
                        "id": 3,
                        "value": "Test instance is refreshed"
                    }
                ]
            }
        ]
}
```

Example Request

curl -H "Content-Type: application/json" -X POST -u admin:admin --data @data.json http://yourserver/rest/raven/1.0/import/execution

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Xray JSON results Multipart

Xray provides another endpoint if you want to create or update a Test Executions and have control over all Test Execution fields. It allows you to send two JSON files, the normal Xray JSON result and a JSON similar to the one Jira uses to create/update issues. For more information about that second format, check the Jira documentation here. Note that in this endpoint the `info` property in the Xray Json result part will be ignored.

Updating an existing Test Run using Xray format REST API will reset all dataset related fields. This means that all current iteration data and dataset present in the Test Run will be replaced with the new information given in the REST API request.

Import the execution results present in query variable "***executionResults***".

Request

Example 1: new Test Execution

Result Json


```
{
    "tests" : [
        {
            "testKey" : "DEMO-6",
            "start" : "2014-08-30T11:47:35+01:00",
            "finish" : "2014-08-30T11:50:56+01:00",
            "comment" : "Successful execution",
            "status" : "PASS"
        },
        {
            "testKey" : "DEMO-7",
            "start" : "2014-08-30T11:51:00+01:00",
            "finish" : "2014-08-30T11:52:30+01:00",
            "comment" : "Execution failed. Example #5 FAIL.",
            "status" : "FAIL",
                        "evidences" : [
                    {
                        "data":
"iVBORw0KGgoAAAANSUhEUgAABkIAAAO9CAYAAADezXv6AAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAAEn(...base64
file enconding)",
                                        "filename": "image21.jpg",
                    "contentType": "image/jpeg"
                }
                        ],
            "examples" : [
                "PASS",
                "PASS",
                "PASS",
                "PASS",
                "FAIL"
            ],
                        "steps": [
                {
                    "status": "PASS",
                    "comment": "Comment on Test Step Result 1",
                    "evidences" : [
                        {
                            "data":
"iVBORw0KGgoAAAANSUhEUgAABkIAAAO9CAYAAADezXv6AAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAAEn(...base64
file enconding)",
                                                                "filename": "image22.jpg",
                            "contentType": "image/jpeg"
                        }
                                        ],
                                        "actualResult": "Actual Result on Test Step 1"
                                    }
                        ],
                    "defects" : [
                                        "DEMO-10",
                                        "DEMO-11"
                    ]
        }
    ]
}
```

Info JSON (Test Execution)

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Brand new Test execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ],
                "customfield_10032" : [
                        "TES-38"
                ]
        }
}
```

Example 2: update existing Test Execution

Result Json

```
{
    "testExecutionKey": "DEMO-1206",
    "tests" : [
        {
            "testKey" : "DEMO-6",
            "start" : "2014-08-30T11:47:35+01:00",
            "finish" : "2014-08-30T11:50:56+01:00",
            "comment" : "Successful execution",
            "status" : "PASS"
        }
     ]
}
```

Info JSON (Test Execution)

```
{
        "fields": {
                "customfield_10032" : [
                        "a_label"
                ],
                "description": "update the issue description"
        }
}
```

Example Request

curl -u admin:admin -F info=@issueFields.json -F result=@results.json http://yourserver/rest/raven/1.0/import/execution/multipart

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Cucumber JSON results

After executing Cucumber features, you must import the outputted JSON execution results to Jira using the following endpoint:

Import the execution results created with the Cucumber JSON output formatter. For more information please check the Cucumber reports documentation (example here).

Request

Example

Example Input

```
[
  {
    "keyword": "Feature",
    "name": "Arithmetic Operations",
    "line": 3,
    "description": "",
    "tags": [
      {
        "name": "@DEMO-48",
        "line": 1
      },
      {
        "name": "@REQ_DEMO-45",
        "line": 2
      }
    ],
    "id": "arithmetic-operations",
    "uri": "features/1_DEMO-45.feature",
    "elements": [
      {
        "comments": [
          {
            "value": "#In order to avoid silly mistakes",
            "line": 4
          },
          {
            "value": "#As a math idiot ",
            "line": 5
          },
          {
            "value": "#I want to be told the result of basic arithmetic operations between two numbers",
            "line": 6
          }
```

```
],
"keyword": "Scenario Outline",
"name": "Add two Numbers",
"line": 18,
"description": "",
"tags": [
  {
    "name": "@TEST_DEMO-47",
    "line": 9
  }
],
"id": "arithmetic-operations;add-two-numbers;;2",
"type": "scenario",
"steps": [
  {
                "embeddings": [
     {
                        "mime_type": "text/plain",
                        "data": "{data base64}"
     },{
                        "mime_type": "text/plain",
                        "data": "{data base64}"
     }
    ],
    "keyword": "Given ",
    "name": "I have entered 20 into the calculator",
    "line": 11,
    "match": {
      "arguments": [
        {
          "offset": 15,
          "val": "20"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:14"
    },
    "result": {
      "status": "passed",
      "duration": 487000
    }
  },
  {
    "keyword": "And ",
    "name": "I have entered 30 into the calculator",
    "line": 12,
    "match": {
      "arguments": [
        {
          "offset": 15,
          "val": "30"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:14"
    },
    "result": {
      "status": "passed",
      "duration": 340000
    }
  },
  {
    "keyword": "When ",
    "name": "I press add",
    "line": 13,
    "match": {
      "arguments": [
        {
          "offset": 8,
          "val": "add"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:18"
```

```
        },
        "result": {
          "status": "passed",
          "duration": 327000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 50 on the screen",
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "50"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 11723000
        }
      }
    ]
  },
  {
    "comments": [
      {
        "value": "#In order to avoid silly mistakes",
        "line": 4
      },
      {
        "value": "#As a math idiot ",
        "line": 5
      },
      {
        "value": "#I want to be told the result of basic arithmetic operations between two numbers",
        "line": 6
      }
    ],
    "keyword": "Scenario Outline",
    "name": "Add two Numbers",
    "line": 19,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-47",
        "line": 9
      }
    ],
    "id": "arithmetic-operations;add-two-numbers;;3",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 2 into the calculator",
        "line": 11,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "2"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 992000
        }
```

```
      },
      {
        "keyword": "And ",
        "name": "I have entered 5 into the calculator",
        "line": 12,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "5"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 775000
        }
      },
      {
        "keyword": "When ",
        "name": "I press add",
        "line": 13,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "add"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 322000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 7 on the screen",
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "7"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 423000
        }
      }
    ]
  },
  {
    "comments": [
      {
        "value": "#In order to avoid silly mistakes",
        "line": 4
      },
      {
        "value": "#As a math idiot ",
        "line": 5
      },
      {
        "value": "#I want to be told the result of basic arithmetic operations between two numbers",
        "line": 6
      }
```

```
],
"keyword": "Scenario Outline",
"name": "Add two Numbers",
"line": 20,
"description": "",
"tags": [
  {
    "name": "@TEST_DEMO-47",
    "line": 9
  }
],
"id": "arithmetic-operations;add-two-numbers;;4",
"type": "scenario",
"steps": [
  {
    "keyword": "Given ",
    "name": "I have entered 0 into the calculator",
    "line": 11,
    "match": {
      "arguments": [
        {
          "offset": 15,
          "val": "0"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:14"
    },
    "result": {
      "status": "passed",
      "duration": 384000
    }
  },
  {
    "keyword": "And ",
    "name": "I have entered 40 into the calculator",
    "line": 12,
    "match": {
      "arguments": [
        {
          "offset": 15,
          "val": "40"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:14"
    },
    "result": {
      "status": "passed",
      "duration": 313000
    }
  },
  {
    "keyword": "When ",
    "name": "I press add",
    "line": 13,
    "match": {
      "arguments": [
        {
          "offset": 8,
          "val": "add"
        }
      ],
      "location": "features/step_definitions/calculator_steps.rb:18"
    },
    "result": {
      "status": "passed",
      "duration": 280000
    }
  },
  {
    "keyword": "Then ",
    "name": "the result should be 40 on the screen",
```

```
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "40"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 350000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 32,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;2",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 8 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "8"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 344000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 4 into the calculator",
        "line": 26,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "4"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 292000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
```

```
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 291000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 2 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "2"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 320000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 33,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;3",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 12 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "12"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 1102000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 3 into the calculator",
        "line": 26,
        "match": {
```

```
          "arguments": [
            {
              "offset": 15,
              "val": "3"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 891000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 291000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 4 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "4"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 339000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 34,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;4",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 3 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
```

```
            {
              "offset": 15,
              "val": "3"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 304000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 1 into the calculator",
        "line": 26,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "1"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 309000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 257000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 5 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "5"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 840000
        }
      }
    ]
  }
 ]
}
]
```

Example Request

curl -H "Content-Type: application/json" -X POST -u admin:admin --data @cucumber_output.json http://yourserver/rest/raven/1.0/import/execution/cucumber

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Cucumber JSON results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution fields. It allows you to send two JSON files, the normal Cucumber result JSON and a JSON similar to the one Jira uses to create new issues. For more information about that second format, check the Jira documentation here.

Import the execution results created with the Cucumber JSON output formatter. For more information, please check the Cucumber reports documentation (example here).

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

Result Json

```
[
  {
    "keyword": "Feature",
    "name": "Arithmetic Operations",
    "line": 3,
    "description": "",
    "tags": [
      {
        "name": "@DEMO-48",
        "line": 1
      },
      {
        "name": "@REQ_DEMO-45",
        "line": 2
      }
    ],
    "id": "arithmetic-operations",
    "uri": "features/1_DEMO-45.feature",
```

```
"elements": [
  {
    "comments": [
      {
        "value": "#In order to avoid silly mistakes",
        "line": 4
      },
      {
        "value": "#As a math idiot ",
        "line": 5
      },
      {
        "value": "#I want to be told the result of basic arithmetic operations between two numbers",
        "line": 6
      }
    ],
    "keyword": "Scenario Outline",
    "name": "Add two Numbers",
    "line": 18,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-47",
        "line": 9
      }
    ],
    "id": "arithmetic-operations;add-two-numbers;;2",
    "type": "scenario",
    "steps": [
      {
                    "embeddings": [
          {
                                "mime_type": "text/plain",
                                "data": "{data base64}"
          },{
                                "mime_type": "text/plain",
                                "data": "{data base64}"
          }
        "keyword": "Given ",
        "name": "I have entered 20 into the calculator",
        "line": 11,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "20"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 487000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 30 into the calculator",
        "line": 12,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "30"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 340000
```

```
        }
      },
      {
        "keyword": "When ",
        "name": "I press add",
        "line": 13,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "add"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 327000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 50 on the screen",
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "50"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 11723000
        }
      }
    ]
  },
  {
    "comments": [
      {
        "value": "#In order to avoid silly mistakes",
        "line": 4
      },
      {
        "value": "#As a math idiot ",
        "line": 5
      },
      {
        "value": "#I want to be told the result of basic arithmetic operations between two numbers",
        "line": 6
      }
    ],
    "keyword": "Scenario Outline",
    "name": "Add two Numbers",
    "line": 19,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-47",
        "line": 9
      }
    ],
    "id": "arithmetic-operations;add-two-numbers;;3",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 2 into the calculator",
```

```
        "line": 11,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "2"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 992000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 5 into the calculator",
        "line": 12,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "5"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 775000
        }
      },
      {
        "keyword": "When ",
        "name": "I press add",
        "line": 13,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "add"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 322000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 7 on the screen",
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "7"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 423000
        }
      }
    ]
  },
```

```
{
  "comments": [
    {
      "value": "#In order to avoid silly mistakes",
      "line": 4
    },
    {
      "value": "#As a math idiot ",
      "line": 5
    },
    {
      "value": "#I want to be told the result of basic arithmetic operations between two numbers",
      "line": 6
    }
  ],
  "keyword": "Scenario Outline",
  "name": "Add two Numbers",
  "line": 20,
  "description": "",
  "tags": [
    {
      "name": "@TEST_DEMO-47",
      "line": 9
    }
  ],
  "id": "arithmetic-operations;add-two-numbers;;4",
  "type": "scenario",
  "steps": [
    {
      "keyword": "Given ",
      "name": "I have entered 0 into the calculator",
      "line": 11,
      "match": {
        "arguments": [
          {
            "offset": 15,
            "val": "0"
          }
        ],
        "location": "features/step_definitions/calculator_steps.rb:14"
      },
      "result": {
        "status": "passed",
        "duration": 384000
      }
    },
    {
      "keyword": "And ",
      "name": "I have entered 40 into the calculator",
      "line": 12,
      "match": {
        "arguments": [
          {
            "offset": 15,
            "val": "40"
          }
        ],
        "location": "features/step_definitions/calculator_steps.rb:14"
      },
      "result": {
        "status": "passed",
        "duration": 313000
      }
    },
    {
      "keyword": "When ",
      "name": "I press add",
      "line": 13,
      "match": {
        "arguments": [
          {
```

```
              "offset": 8,
              "val": "add"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 280000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 40 on the screen",
        "line": 14,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "40"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 350000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 32,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;2",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 8 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "8"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 344000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 4 into the calculator",
        "line": 26,
        "match": {
          "arguments": [
            {
              "offset": 15,
```

```
              "val": "4"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 292000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 291000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 2 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "2"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 320000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 33,
    "description": "",
    "tags": [
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;3",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 12 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "12"
```

```
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 1102000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 3 into the calculator",
        "line": 26,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "3"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 891000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 291000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 4 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "4"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:22"
        },
        "result": {
          "status": "passed",
          "duration": 339000
        }
      }
    ]
  },
  {
    "keyword": "Scenario Outline",
    "name": "Divide Two Numbers",
    "line": 34,
    "description": "",
    "tags": [
```

```
      {
        "name": "@TEST_DEMO-46",
        "line": 23
      }
    ],
    "id": "arithmetic-operations;divide-two-numbers;;4",
    "type": "scenario",
    "steps": [
      {
        "keyword": "Given ",
        "name": "I have entered 3 into the calculator",
        "line": 25,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "3"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 304000
        }
      },
      {
        "keyword": "And ",
        "name": "I have entered 1 into the calculator",
        "line": 26,
        "match": {
          "arguments": [
            {
              "offset": 15,
              "val": "1"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:14"
        },
        "result": {
          "status": "passed",
          "duration": 309000
        }
      },
      {
        "keyword": "When ",
        "name": "I press divide",
        "line": 27,
        "match": {
          "arguments": [
            {
              "offset": 8,
              "val": "divide"
            }
          ],
          "location": "features/step_definitions/calculator_steps.rb:18"
        },
        "result": {
          "status": "passed",
          "duration": 257000
        }
      },
      {
        "keyword": "Then ",
        "name": "the result should be 5 on the screen",
        "line": 28,
        "match": {
          "arguments": [
            {
              "offset": 21,
              "val": "5"
```

```
              }
            ],
            "location": "features/step_definitions/calculator_steps.rb:22"
          },
          "result": {
            "status": "passed",
            "duration": 840000
          }
        }
      ]
    }
  ]
]
```

Info JSON (Test Execution)

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for cucumber Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ],
                "customfield_10032" : [
                        "TES-38"
                ]
        }
}
```

Example Request

curl -u admin:admin -F info=@createTestExec.json -F result=@results.json http://yourserver/rest/raven/1.0/import/execution/cucumber/multipart

Assigning Test Environment(s) to Test Execution

It's possible to assign Test Environment(s) to the newly-created Test Execution. For that, you need to pass the ID of the custom field corresponding to the "Test Environments" custom field. In the JSON example below, it is 10030 for the info object.

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

```
{
    "fields": {
        "project": {
            "key": "XRAY"
        },
        "summary": "Test Execution for cucumber Execution",
        "issuetype": {
            "id": "10009"
        },
        "customfield_10030" : [
            "iOS", "Android"
        ]
    }
}
```

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Behave JSON results

After executing Behave features, you must import the outputted JSON execution results to Jira using the following endpoint:

Import the execution results created with the Behave JSON output formatter.

Request

Example

Example Input

```
[
        {
                "status": "failed",
                "elements": [
```

```
{
        "name": "Test automatic",
        "keyword": "Scenario",
        "tags": [
                "XTP-11"
        ],
        "steps": [
                {
                        "name": "I have entered 20 into the calculator",
                        "keyword": "Given",
                        "step_type": "given",
                        "result": {
                                "status": "failed",
                                "duration": 3.0994415283203125e-03
                        },
                        "match": {
                                "location": "steps/tutorial.py:13",
                                "arguments": []
                        },
                        "location": "1 (8).feature:7"
                },
                {
                        "name": "I have entered 30 into the calculator",
                        "keyword": "And",
                        "step_type": "given",
                        "result": {
                                "status": "failed",
                                "duration": 2.5033950805664062e-03
                        },
                        "match": {
                                "location": "steps/tutorial.py:17",
                                "arguments": []
                        },
                        "location": "1 (8).feature:8"
                },
                {
                        "name": "I press add",
                        "keyword": "When",
                        "step_type": "when",
                        "result": {
                                "status": "failed",
                                "duration": 2.288818359375e-03
                        },
                        "match": {
                                "location": "steps/tutorial.py:21",
                                "arguments": []
                        },
                        "location": "1 (8).feature:9"
                },
                {
                        "name": "the result should be 50 on the screen",
                        "keyword": "Then",
                        "step_type": "then",
                        "result": {
                                "status": "failed",
                                "duration": 2.2172927856445312e-03
                        },
                        "match": {
                                "location": "steps/tutorial.py:25",
                                "arguments": []
                        },
                        "location": "1 (8).feature:10"
                }
        ],
        "location": "1 (8).feature:6",
        "type": "scenario"
},
{
        "name": "Test -- @2.1 Consumer Electronics",
        "keyword": "Scenario Outline",
        "tags": [
```

```
                        "XTP-11"
                ],
                "steps": [
                        {
                                "name": "I put \"iPhone\" in a blender",
                                "keyword": "Given",
                                "step_type": "given",
                                "result": {
                                        "status": "failed",
                                        "duration": 5.1021575927734375e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:29",
                                        "arguments": [
                                                {
                                                        "name": "thing",
                                                        "value": "iPhone"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:16"
                        },
                        {
                                "name": "I switch the blender on",
                                "keyword": "When",
                                "step_type": "when",
                                "result": {
                                        "status": "failed",
                                        "duration": 3.4809112548828125e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:34",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:17"
                        },
                        {
                                "name": "it should transform into \"toxic waste\"",
                                "keyword": "Then",
                                "step_type": "then",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.6941299438476562e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:38",
                                        "arguments": [
                                                {
                                                        "name": "other_thing",
                                                        "value": "toxic waste"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:18"
                        }
                ],
                "location": "1 (8).feature:27",
                "type": "scenario"
        },
        {
                "name": "Test -- @2.2 Consumer Electronics",
                "keyword": "Scenario Outline",
                "tags": [
                        "XTP-11"
                ],
                "steps": [
                        {
                                "name": "I put \"Galaxy Nexus\" in a blender",
                                "keyword": "Given",
                                "step_type": "given",
                                "result": {
```

```
                                        "status": "failed",
                                        "duration": 3.814697265625e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:29",
                                        "arguments": [
                                                {
                                                        "name": "thing",
                                                        "value": "Galaxy Nexus"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:16"
                        },
                        {
                                "name": "I switch the blender on",
                                "keyword": "When",
                                "step_type": "when",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.5033950805664062e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:34",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:17"
                        },
                        {
                                "name": "it should transform into \"toxic waste\"",
                                "keyword": "Then",
                                "step_type": "then",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.8133392333984375e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:38",
                                        "arguments": [
                                                {
                                                        "name": "other_thing",
                                                        "value": "toxic waste"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:18"
                        }
                ],
                "location": "1 (8).feature:28",
                "type": "scenario"
        }
    ],
    "name": "",
    "keyword": "Feature",
    "tags": [
            "XTP-2"
    ],
    "location": "1 (8).feature:2"
  }
]
```


Example Request

curl -H "Content-Type: application/json" -X POST -u admin:admin --data @cucumber_output.json http://yourserver/rest/raven/1.0/import/execution/behave

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Behave JSON results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution fields. It allows you to send two JSON files, the normal Behave's result JSON and a JSON similar to the one Jira uses to create new issues. For more information about that second format, check the Jira documentation here.

Import the execution results created with the Behave JSON output formatter.

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

Result JSON

```
[
        {
                "status": "failed",
                "elements": [
                        {
                                "name": "Test automatic",
                                "keyword": "Scenario",
                                "tags": [
                                        "XTP-11"
                                ],
                                "steps": [
                                        {
                                                "name": "I have entered 20 into the calculator",
                                                "keyword": "Given",
                                                "step_type": "given",
                                                "result": {
                                                        "status": "failed",
                                                        "duration": 3.0994415283203125e-03
                                                },
                                                "match": {
                                                        "location": "steps/tutorial.py:13",
                                                        "arguments": []
                                                },
                                                "location": "1 (8).feature:7"
                                        },
                                        {
                                                "name": "I have entered 30 into the calculator",
                                                "keyword": "And",
                                                "step_type": "given",
                                                "result": {
```

```
                                        "status": "failed",
                                        "duration": 2.5033950805664062e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:17",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:8"
                        },
                        {
                                "name": "I press add",
                                "keyword": "When",
                                "step_type": "when",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.288818359375e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:21",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:9"
                        },
                        {
                                "name": "the result should be 50 on the screen",
                                "keyword": "Then",
                                "step_type": "then",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.2172927856445312e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:25",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:10"
                        }
                ],
                "location": "1 (8).feature:6",
                "type": "scenario"
        },
        {
                "name": "Test -- @2.1 Consumer Electronics",
                "keyword": "Scenario Outline",
                "tags": [
                        "XTP-11"
                ],
                "steps": [
                        {
                                "name": "I put \"iPhone\" in a blender",
                                "keyword": "Given",
                                "step_type": "given",
                                "result": {
                                        "status": "failed",
                                        "duration": 5.1021575927734375e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:29",
                                        "arguments": [
                                                {
                                                        "name": "thing",
                                                        "value": "iPhone"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:16"
                        },
                        {
                                "name": "I switch the blender on",
                                "keyword": "When",
                                "step_type": "when",
```

```
                                "result": {
                                        "status": "failed",
                                        "duration": 3.4809112548828125e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:34",
                                        "arguments": []
                                },
                                "location": "1 (8).feature:17"
                        },
                        {
                                "name": "it should transform into \"toxic waste\"",
                                "keyword": "Then",
                                "step_type": "then",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.6941299438476562e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:38",
                                        "arguments": [
                                                {
                                                        "name": "other_thing",
                                                        "value": "toxic waste"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:18"
                        }
                ],
                "location": "1 (8).feature:27",
                "type": "scenario"
        },
        {
                "name": "Test -- @2.2 Consumer Electronics",
                "keyword": "Scenario Outline",
                "tags": [
                        "XTP-11"
                ],
                "steps": [
                        {
                                "name": "I put \"Galaxy Nexus\" in a blender",
                                "keyword": "Given",
                                "step_type": "given",
                                "result": {
                                        "status": "failed",
                                        "duration": 3.814697265625e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:29",
                                        "arguments": [
                                                {
                                                        "name": "thing",
                                                        "value": "Galaxy Nexus"
                                                }
                                        ]
                                },
                                "location": "1 (8).feature:16"
                        },
                        {
                                "name": "I switch the blender on",
                                "keyword": "When",
                                "step_type": "when",
                                "result": {
                                        "status": "failed",
                                        "duration": 2.5033950805664062e-03
                                },
                                "match": {
                                        "location": "steps/tutorial.py:34",
                                        "arguments": []
                                },
```

```
                                        "location": "1 (8).feature:17"
                                },
                                {
                                        "name": "it should transform into \"toxic waste\"",
                                        "keyword": "Then",
                                        "step_type": "then",
                                        "result": {
                                                "status": "failed",
                                                "duration": 2.8133392333984375e-03
                                        },
                                        "match": {
                                                "location": "steps/tutorial.py:38",
                                                "arguments": [
                                                        {
                                                                "name": "other_thing",
                                                                "value": "toxic waste"
                                                        }
                                                ]
                                        },
                                        "location": "1 (8).feature:18"
                                }
                        ],
                        "location": "1 (8).feature:28",
                        "type": "scenario"
                }
        ],
        "name": "",
        "keyword": "Feature",
        "tags": [
                "XTP-2"
        ],
        "location": "1 (8).feature:2"
    }
]
```


Info JSON


```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for cucumber Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ],
                "customfield_10032" : [
                        "TES-38"
                ]
        }
}
```


Example Request

curl -u admin:admin -F info=@createTest.json -F result=@results.json http://yourserver/rest/raven/1.0/import/execution/behave/multipart

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

JUnit XML results

After executing JUnit tests, you must import the outputted XML execution results to Jira using the following endpoint:

Import the execution results created with the JUnit XML output formatter. For more information, please check the documentation about JUnit integration.

Request

QUERY PARAMETERS

parameter	type	description
`projectKey`	*String*	- key of the project where the test execution (if the **testExecKey** parameter wasn't provided) and the tests (if they aren't created yet) are going to be created.
testExecKey	String	- key of the Test Execution.
testPlanKey	String	- key of the Test Plan; if you specify the Test Plan, the Tests will be added automatically to the Test Plan if they're not part of it.
testEnvironments	String	- a string containing a list of test environments separated by ";"
revision	String	- source code and documentation version used in the test execution.
fixVersion	String	- the Fix Version associated with the test execution (it supports only one value).

multipart/form-data:

***"file"** : a **MultipartFormParam** containing a **XML file** to import.*

Example

JUnit Report XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<testsuite tests="15" failures="0" name="ut.com.xpandit.raven.service.impl.IssueDataSetTest" time="0.163"
errors="0" skipped="0">
  <properties>
    ...
  </properties>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidLimitOverflowOption_returnsExpectedSubset" time="0.114"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withNullOptionsAndValidIssue_throwsIllegalArgumentException" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidEmptyOptions_returnsAllIssues" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptions_returnsExpectedTests" time="0.016"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndInvalidColumnSearchOption_returnsAllTests" time="0.007"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidLimitUnderOption_returnsExpectedSubset" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMachesIssueKey_returnsExpectedTestWithMatchedKey"
time="0.006"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidSummaryColumnAscSortOption_returnsExpectedIssuesInAscOrder" time="
0.006"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidSummaryColumnDescSortOption_returnsExpectedIssuesInDescOrder" time="
0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMatchesAllElements_returnsAllTests" time="0.001"
/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidColumnSearchOptionThatMatchesOneElement_returnsOneTest" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidColumnSearchOptionThatMatchesNoIssue_returnsEmptyList" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMachesNoIssue_returnsEmptyList" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidKeyColumnDescSortOption_returnsExpectedIssuesInDescOrder" time="0.001"
/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidKeyColumnAscSortOption_returnsExpectedIssuesInAscOrder" time="0.001"/>
</testsuite>
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/junit?projectKey=XTP

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/junit?testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/junit?projectKey=XTP&testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/junit?projectKey=XTP&testPlanKey=XTP-12&revision=v2.1.0

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  }
}
```

400 BAD_REQUEST : **application/json :** Returns the error.

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

JUnit XML results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution and Test fields. It allows you to send one XML file (the JUnit report) and two JSON files similar to the one Jira uses to create new issues. For more information about that JSON format, check the Jira documentation here.

As of the creation of Test issues, some field values in the JSON file will follow some rules:

- Issue Type: Overridden by the Test issue type;
- Summary: Overridden by the internally generated summary;
- Parent: This field is ignored;
- Project: If the there is not a Test issue with the same Generic Test Definition, then it will be created in the provided project;
- Generic Test Definition: Not allowed to provide a value for this field.

Import the execution results created with the JUnit XML output formatter. For more information, please check the documentation about JUnit integration.

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

JUnit Report XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<testsuite tests="15" failures="0" name="ut.com.xpandit.raven.service.impl.IssueDataSetTest" time="0.163"
errors="0" skipped="0">
  <properties>
    ...
  </properties>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidLimitOverflowOption_returnsExpectedSubset" time="0.114"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withNullOptionsAndValidIssue_throwsIllegalArgumentException" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidEmptyOptions_returnsAllIssues" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptions_returnsExpectedTests" time="0.016"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndInvalidColumnSearchOption_returnsAllTests" time="0.007"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidLimitUnderOption_returnsExpectedSubset" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMachesIssueKey_returnsExpectedTestWithMatchedKey"
time="0.006"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidSummaryColumnAscSortOption_returnsExpectedIssuesInAscOrder" time="
0.006"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidSummaryColumnDescSortOption_returnsExpectedIssuesInDescOrder" time="
0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMatchesAllElements_returnsAllTests" time="0.001"
/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidColumnSearchOptionThatMatchesOneElement_returnsOneTest" time="0.002"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidColumnSearchOptionThatMatchesNoIssue_returnsEmptyList" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidGlobalSearchOptionThatMachesNoIssue_returnsEmptyList" time="0.001"/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidKeyColumnDescSortOption_returnsExpectedIssuesInDescOrder" time="0.001"
/>
  <testcase classname="ut.com.xpandit.raven.service.impl.IssueDataSetTest" name="
testApplyOptions_withValidIssueAndValidKeyColumnAscSortOption_returnsExpectedIssuesInAscOrder" time="0.001"/>
</testsuite>
```

Test Exec Info JSON

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for junit Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ]
        }
}
```

Test Info JSON

```
{
    "fields": {
        "description": "Automated Test",
        "priority" : {
                "id": "10"
        },
        "labels": [
            "Testing",
            "Automation"
        ]
    }
}
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" -F "info=@testExec.json" -F "testInfo=@test.json" http://yourserver/rest/raven/1.0/import/execution/junit/multipart

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira. The following Test issues were also created with success.

Example Output


```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "success": [
            {
                "self": "http://localhost:8080/rest/api/2/issue/10201",
                "id": "10201",
                "key": "XNP-25"
            },
            {
                "self": "http://localhost:8080/rest/api/2/issue/10202",
                "id": "10202",
                "key": "XNP-26"
            }
        ]
    }
}
```


200 OK : **application/json**: Some results where successfully imported to Jira. But the following Test issues failed to be created due to the following reasons.

Example Output


```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "error": [
            {
                "messages": [
                    "Field 'customfield_10005' cannot be set. It is not on the appropriate screen, or
unknown."
                ],
                "testDefinition": "ut.com.xpandit.raven.service.impl.IssueDataSetTest.
testApplyOptions_withNullOptionsAndValidIssue_throwsIllegalArgumentException"
            }
        ]
    }
}
```


400 BAD_REQUEST : **application/json :** Returns the error.

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

TestNG XML results

After executing TestNG tests, you must import the outputted XML execution results to Jira using the following endpoint:

Import the execution results created with the TestNG XML output formatter. For more information please check the documentation about TestNG integration.

Request

QUERY PARAMETERS

parameter	type	description
`projectKey`	*String*	- key of the project where the Test Execution (if the **testExecKey** parameter wasn't provided) and the tests (if they aren't created yet) are going to be created.
testExecKey	String	- key of the Test Execution.
testPlanKey	String	- key of the Test Plan; if you specify the Test Plan, the Tests will be added automatically to the Test Plan if they're not part of it.
testEnvironments	String	- a string containing a list of test environments separated by ";"
revision	String	- source code and documentation version used in the test execution.
fixVersion	String	- the Fix Version associated with the test execution (it supports only one value).

multipart/form-data:

***"file"** : a **MultipartFormParam** containing a **XML file** to import.*

Example

TestNG XML Report

```
<?xml version="1.0" encoding="UTF-8"?>
<testng-results skipped="0" failed="2" ignored="0" total="8" passed="6">
  <reporter-output>
  </reporter-output>
  <suite name="TestAll" duration-ms="33" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
    <groups>
    </groups>
    <test name="calculator" duration-ms="33" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:
00Z">
      <class name="com.xpand.java.CalcTest">
        <test-method status="PASS" signature="setUp()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]" name="
setUp" is-config="true" duration-ms="9" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
        </test-method> <!-- setUp -->
        <test-method status="PASS" signature="CanAddNumbers()[pri:0, instance:com.xpand.java.
CalcTest@36d4b5c]" name="CanAddNumbers" duration-ms="2" started-at="2018-03-06T11:53:00Z" finished-at="2018-
03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
                        <attribute name="summary">
              <![CDATA[As a user, I can add numbers]]>
            </attribute> <!-- summary -->
            <attribute name="description">
              <![CDATA[Tests that a sequence of numbers can be added]]>
            </attribute> <!-- description -->
          </attributes>
        </test-method> <!-- CanAddNumbers -->
        <test-method status="PASS" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="0" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
```

```
            <param index="0">
              <value>
                <![CDATA[1]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[2]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[3]]>
              </value>
            </param>
          </params>
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="FAIL" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="1" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
            <param index="0">
              <value>
                <![CDATA[2]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[3]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[4]]>
              </value>
            </param>
          </params>
          <exception class="java.lang.AssertionError">
            <message>
              <![CDATA[expected [4] but found [5]]]>
            </message>
            <full-stacktrace>
              <![CDATA[java.lang.AssertionError: expected [4] but found [5]
        at org.testng.Assert.fail(Assert.java:93)
        at org.testng.Assert.failNotEquals(Assert.java:512)
        at org.testng.Assert.assertEqualsImpl(Assert.java:134)
        at org.testng.Assert.assertEquals(Assert.java:115)
        at org.testng.Assert.assertEquals(Assert.java:388)
        at org.testng.Assert.assertEquals(Assert.java:398)
        at com.xpand.java.CalcTest.CanAddNumbersFromGivenData(CalcTest.java:40)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:108)
        at org.testng.internal.Invoker.invokeMethod(Invoker.java:661)
        at org.testng.internal.Invoker.invokeTestMethod(Invoker.java:869)
        at org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1193)
```

```
        at org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:126)
        at org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:109)
        at org.testng.TestRunner.privateRun(TestRunner.java:744)
        at org.testng.TestRunner.run(TestRunner.java:602)
        at org.testng.SuiteRunner.runTest(SuiteRunner.java:380)
        at org.testng.SuiteRunner.runSequentially(SuiteRunner.java:375)
        at org.testng.SuiteRunner.privateRun(SuiteRunner.java:340)
        at org.testng.SuiteRunner.run(SuiteRunner.java:289)
        at org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
        at org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
        at org.testng.TestNG.runSuitesSequentially(TestNG.java:1301)
        at org.testng.TestNG.runSuitesLocally(TestNG.java:1226)
        at org.testng.TestNG.runSuites(TestNG.java:1144)
        at org.testng.TestNG.run(TestNG.java:1115)
        at org.apache.maven.surefire.testng.TestNGExecutor.run(TestNGExecutor.java:283)
        at org.apache.maven.surefire.testng.TestNGXmlTestSuite.execute(TestNGXmlTestSuite.java:75)
        at org.apache.maven.surefire.testng.TestNGProvider.invoke(TestNGProvider.java:120)
        at org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:
373)
        at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:334)
        at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:119)
        at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:407)
]]>
            </full-stacktrace>
          </exception> <!-- java.lang.AssertionError -->
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="PASS" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="0" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
            <param index="0">
              <value>
                <![CDATA[-1]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[1]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[0]]>
              </value>
            </param>
          </params>
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
```

```
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="FAIL" signature="CanDoStuff()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanDoStuff" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <exception class="java.lang.AssertionError">
            <message>
              <![CDATA[null]]>
            </message>
            <full-stacktrace>
              <![CDATA[java.lang.AssertionError: null
        at org.testng.Assert.fail(Assert.java:93)
        at org.testng.Assert.assertNotEquals(Assert.java:897)
        at org.testng.Assert.assertNotEquals(Assert.java:902)
        at com.xpand.java.CalcTest.CanDoStuff(CalcTest.java:86)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:108)
        at org.testng.internal.Invoker.invokeMethod(Invoker.java:661)
        at org.testng.internal.Invoker.invokeTestMethod(Invoker.java:869)
        at org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1193)
        at org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:126)
        at org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:109)
        at org.testng.TestRunner.privateRun(TestRunner.java:744)
        at org.testng.TestRunner.run(TestRunner.java:602)
        at org.testng.SuiteRunner.runTest(SuiteRunner.java:380)
        at org.testng.SuiteRunner.runSequentially(SuiteRunner.java:375)
        at org.testng.SuiteRunner.privateRun(SuiteRunner.java:340)
        at org.testng.SuiteRunner.run(SuiteRunner.java:289)
        at org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
        at org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
        at org.testng.TestNG.runSuitesSequentially(TestNG.java:1301)
        at org.testng.TestNG.runSuitesLocally(TestNG.java:1226)
        at org.testng.TestNG.runSuites(TestNG.java:1144)
        at org.testng.TestNG.run(TestNG.java:1115)
        at org.apache.maven.surefire.testng.TestNGExecutor.run(TestNGExecutor.java:283)
        at org.apache.maven.surefire.testng.TestNGXmlTestSuite.execute(TestNGXmlTestSuite.java:75)
        at org.apache.maven.surefire.testng.TestNGProvider.invoke(TestNGProvider.java:120)
        at org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:
373)
        at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:334)
        at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:119)
        at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:407)
]]>
            </full-stacktrace>
          </exception> <!-- java.lang.AssertionError -->
          <reporter-output>
          </reporter-output>
        </test-method> <!-- CanDoStuff -->
        <test-method status="PASS" signature="CanDivide()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanDivide" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanDivide -->
        <test-method status="PASS" signature="CanMultiplyX()[pri:0, instance:com.xpand.java.
CalcTest@36d4b5c]" name="CanMultiplyX" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-
06T11:53:00Z">
          <reporter-output>
          </reporter-output>
```

```
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanMultiplyX -->
        <test-method status="PASS" signature="CanSubtract()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanSubtract" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanSubtract -->
        <test-method status="PASS" signature="tearDown()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="tearDown" is-config="true" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:
53:00Z">
          <reporter-output>
          </reporter-output>
        </test-method> <!-- tearDown -->
      </class> <!-- com.xpand.java.CalcTest -->
    </test> <!-- calculator -->
  </suite> <!-- TestAll -->
</testng-results>
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/testng?projectKey=XTP

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/testng?testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/testng?projectKey=XTP&testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/testng?projectKey=XTP&testPlanKey=XTP-12&fixVersion=v2.1.0

Responses

200 OK : ***application/json*** : Successful. The results where successfully imported to Jira.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  }
}
```

400 BAD_REQUEST : ***application/json :*** *Returns the error.*

401 UNAUTHORIZED : ***application/json*** *: The Xray license is not valid.*

500 INTERNAL SERVER ERROR : ***application/json*** *: An internal error occurred when importing execution results.*

TestNG XML results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution and Test fields. It allows you to send one XML file (the TestNG report) and two JSON files similar to the one Jira uses to create new issues. For more information about that JSON format, check the Jira documentation here.

As of the creation of Test issues, some field values in the JSON file will follow some rules:

- Issue Type: Overridden by the Test issue type;
- Summary: Overridden by the internally generated summary, or by the summary attribute if provided in the xml result file;
- Description: Overridden by the description attribute if provided in the xml result file;
- Parent: This field is ignored;
- Project: If the there is not a Test issue with the same Generic Test Definition, then it will be created in the provided project;
- Generic Test Definition: Not allowed to provide a value for this field.

Import the execution results created with the NUnit XML output formatter. For more information please check the documentation about TestNG integration.

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

TestNG XML Report

```
<?xml version="1.0" encoding="UTF-8"?>
<testng-results skipped="0" failed="2" ignored="0" total="8" passed="6">
  <reporter-output>
  </reporter-output>
  <suite name="TestAll" duration-ms="33" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
    <groups>
    </groups>
    <test name="calculator" duration-ms="33" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:
00Z">
      <class name="com.xpand.java.CalcTest">
        <test-method status="PASS" signature="setUp()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]" name="
setUp" is-config="true" duration-ms="9" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
        </test-method> <!-- setUp -->
        <test-method status="PASS" signature="CanAddNumbers()[pri:0, instance:com.xpand.java.
CalcTest@36d4b5c]" name="CanAddNumbers" duration-ms="2" started-at="2018-03-06T11:53:00Z" finished-at="2018-
03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
```

```
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbers -->
        <test-method status="PASS" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="0" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
            <param index="0">
              <value>
                <![CDATA[1]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[2]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[3]]>
              </value>
            </param>
          </params>
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="FAIL" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="1" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
            <param index="0">
              <value>
                <![CDATA[2]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[3]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[4]]>
              </value>
            </param>
          </params>
          <exception class="java.lang.AssertionError">
            <message>
              <![CDATA[expected [4] but found [5]]]>
            </message>
            <full-stacktrace>
              <![CDATA[java.lang.AssertionError: expected [4] but found [5]
        at org.testng.Assert.fail(Assert.java:93)
        at org.testng.Assert.failNotEquals(Assert.java:512)
```

```
        at org.testng.Assert.assertEqualsImpl(Assert.java:134)
        at org.testng.Assert.assertEquals(Assert.java:115)
        at org.testng.Assert.assertEquals(Assert.java:388)
        at org.testng.Assert.assertEquals(Assert.java:398)
        at com.xpand.java.CalcTest.CanAddNumbersFromGivenData(CalcTest.java:40)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:108)
        at org.testng.internal.Invoker.invokeMethod(Invoker.java:661)
        at org.testng.internal.Invoker.invokeTestMethod(Invoker.java:869)
        at org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1193)
        at org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:126)
        at org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:109)
        at org.testng.TestRunner.privateRun(TestRunner.java:744)
        at org.testng.TestRunner.run(TestRunner.java:602)
        at org.testng.SuiteRunner.runTest(SuiteRunner.java:380)
        at org.testng.SuiteRunner.runSequentially(SuiteRunner.java:375)
        at org.testng.SuiteRunner.privateRun(SuiteRunner.java:340)
        at org.testng.SuiteRunner.run(SuiteRunner.java:289)
        at org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
        at org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
        at org.testng.TestNG.runSuitesSequentially(TestNG.java:1301)
        at org.testng.TestNG.runSuitesLocally(TestNG.java:1226)
        at org.testng.TestNG.runSuites(TestNG.java:1144)
        at org.testng.TestNG.run(TestNG.java:1115)
        at org.apache.maven.surefire.testng.TestNGExecutor.run(TestNGExecutor.java:283)
        at org.apache.maven.surefire.testng.TestNGXmlTestSuite.execute(TestNGXmlTestSuite.java:75)
        at org.apache.maven.surefire.testng.TestNGProvider.invoke(TestNGProvider.java:120)
        at org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:
373)
        at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:334)
        at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:119)
        at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:407)
]]>
            </full-stacktrace>
          </exception> <!-- java.lang.AssertionError -->
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="PASS" signature="CanAddNumbersFromGivenData(int, int, int)[pri:0, instance:com.
xpand.java.CalcTest@36d4b5c]" name="CanAddNumbersFromGivenData" duration-ms="0" started-at="2018-03-06T11:53:
00Z" data-provider="ValidDataProvider" finished-at="2018-03-06T11:53:00Z">
          <params>
            <param index="0">
              <value>
                <![CDATA[-1]]>
              </value>
            </param>
            <param index="1">
              <value>
                <![CDATA[1]]>
              </value>
            </param>
            <param index="2">
              <value>
                <![CDATA[0]]>
              </value>
            </param>
```

```
          </params>
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanAddNumbersFromGivenData -->
        <test-method status="FAIL" signature="CanDoStuff()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanDoStuff" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <exception class="java.lang.AssertionError">
            <message>
              <![CDATA[null]]>
            </message>
            <full-stacktrace>
              <![CDATA[java.lang.AssertionError: null
        at org.testng.Assert.fail(Assert.java:93)
        at org.testng.Assert.assertNotEquals(Assert.java:897)
        at org.testng.Assert.assertNotEquals(Assert.java:902)
        at com.xpand.java.CalcTest.CanDoStuff(CalcTest.java:86)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:108)
        at org.testng.internal.Invoker.invokeMethod(Invoker.java:661)
        at org.testng.internal.Invoker.invokeTestMethod(Invoker.java:869)
        at org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1193)
        at org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:126)
        at org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:109)
        at org.testng.TestRunner.privateRun(TestRunner.java:744)
        at org.testng.TestRunner.run(TestRunner.java:602)
        at org.testng.SuiteRunner.runTest(SuiteRunner.java:380)
        at org.testng.SuiteRunner.runSequentially(SuiteRunner.java:375)
        at org.testng.SuiteRunner.privateRun(SuiteRunner.java:340)
        at org.testng.SuiteRunner.run(SuiteRunner.java:289)
        at org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
        at org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
        at org.testng.TestNG.runSuitesSequentially(TestNG.java:1301)
        at org.testng.TestNG.runSuitesLocally(TestNG.java:1226)
        at org.testng.TestNG.runSuites(TestNG.java:1144)
        at org.testng.TestNG.run(TestNG.java:1115)
        at org.apache.maven.surefire.testng.TestNGExecutor.run(TestNGExecutor.java:283)
        at org.apache.maven.surefire.testng.TestNGXmlTestSuite.execute(TestNGXmlTestSuite.java:75)
        at org.apache.maven.surefire.testng.TestNGProvider.invoke(TestNGProvider.java:120)
        at org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:
373)
        at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:334)
        at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:119)
        at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:407)
]]>
            </full-stacktrace>
          </exception> <!-- java.lang.AssertionError -->
          <reporter-output>
          </reporter-output>
        </test-method> <!-- CanDoStuff -->
        <test-method status="PASS" signature="CanDivide()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanDivide" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
```

```
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanDivide -->
        <test-method status="PASS" signature="CanMultiplyX()[pri:0, instance:com.xpand.java.
CalcTest@36d4b5c]" name="CanMultiplyX" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-
06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanMultiplyX -->
        <test-method status="PASS" signature="CanSubtract()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="CanSubtract" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:53:00Z">
          <reporter-output>
          </reporter-output>
          <attributes>
            <attribute name="test">
              <![CDATA[]]>
            </attribute> <!-- test -->
            <attribute name="requirement">
              <![CDATA[CALC-1235]]>
            </attribute> <!-- requirement -->
            <attribute name="labels">
              <![CDATA[core]]>
            </attribute> <!-- labels -->
          </attributes>
        </test-method> <!-- CanSubtract -->
        <test-method status="PASS" signature="tearDown()[pri:0, instance:com.xpand.java.CalcTest@36d4b5c]"
name="tearDown" is-config="true" duration-ms="0" started-at="2018-03-06T11:53:00Z" finished-at="2018-03-06T11:
53:00Z">
          <reporter-output>
          </reporter-output>
        </test-method> <!-- tearDown -->
      </class> <!-- com.xpand.java.CalcTest -->
    </test> <!-- calculator -->
  </suite> <!-- TestAll -->
</testng-results>
```

Test Exec Info JSON

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for TestNG Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ]
        }
}
```

Test Info JSON

```
{
    "fields": {
        "description": "Automated Test",
        "priority" : {
                "id": "10"
        },
        "labels": [
            "Testing",
            "Automation"
        ]
    }
}
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" -F "info=@testExec.json" -F "testInfo=@test.json" http://yourserver/rest/raven/1.0/import/execution/testng/multipart

Responses

200 OK **: application/json :** Successful. The results where successfully imported to Jira. The following Test issues were also created with success.

Example Output


```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "success": [
            {
                "self": "http://localhost:8080/rest/api/2/issue/10201",
                "id": "10201",
                "key": "XNP-25"
            },
            {
                "self": "http://localhost:8080/rest/api/2/issue/10202",
                "id": "10202",
                "key": "XNP-26"
            }
        ]
    }
}
```


200 OK : **application/json**: Some results where successfully imported to Jira. But the following Test issues failed to be created due to the following reasons.

Example Output


```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "error": [
            {
                "messages": [
                    "Field 'customfield_10005' cannot be set. It is not on the appropriate screen, or
unknown."
                ],
                "testDefinition": "com.xpand.java.CalcTest.CanAddNumbersFromGivenData"
            }
        ]
    }
}
```


400 BAD_REQUEST **: application/json :** Returns the error.

401 UNAUTHORIZED **: application/json :** The Xray license is not valid.

500 INTERNAL SERVER ERROR **: application/json :** An internal error occurred when importing execution results.

NUnit XML results

After executing NUnit tests, you must import the outputted XML execution results to Jira using the following endpoint:

Import the execution results created with the NUnit XML output formatter. For more information please check the documentation about NUnit integration.

Request

QUERY PARAMETERS

parameter	type	description
`projectKey`	*String*	- key of the project where the Test Execution (if the **testExecKey** parameter wasn't provided) and the tests (if they aren't created yet) are going to be created.
testExecKey	String	- key of the Test Execution.
testPlanKey	String	- key of the Test Plan; if you specify the Test Plan, the Tests will be added automatically to the Test Plan if they're not part of it.
testEnvironments	String	- a string containing a list of test environments separated by ";"
revision	String	- source code and documentation version used in the test execution.
fixVersion	String	- the Fix Version associated with the test execution (it supports only one value).

multipart/form-data:

***"file"** : a **MultipartFormParam** containing a **XML file** to import.*

Example

NUnit Report XML

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<test-run id="0" testcasecount="14" total="14" passed="13" failed="1" inconclusive="0" skipped="0" asserts="
14" result="Failed" portable-engine-version="3.3.0.0" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26
14:36:03Z" duration="0.140400">
<test-suite type="Assembly" id="1021" name="x, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null"
fullname="x, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null" runstate="Runnable" testcasecount="14"
result="Failed" site="Child" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="
0.110549" total="14" passed="13" failed="1" inconclusive="0" skipped="0" asserts="14">
  <settings>
    <setting name="WorkDirectory" value="C:\Users\Sergio\x" />
  </settings>
  <failure>
    <message><![CDATA[One or more child tests had errors]]></message>
  </failure>
  <test-suite type="TestFixture" id="1000" name="TestClass" fullname="TestClass" classname="TestClass"
runstate="Runnable" testcasecount="2" result="Failed" site="Child" start-time="2016-12-26 14:36:03Z" end-
time="2016-12-26 14:36:03Z" duration="0.084668" total="2" passed="1" failed="1" inconclusive="0" skipped="0"
asserts="2">
    <failure>
      <message><![CDATA[One or more child tests had errors]]></message>
    </failure>
    <test-suite type="ParameterizedMethod" id="1003" name="SubtractTest" fullname="TestClass.SubtractTest"
classname="TestClass" runstate="Runnable" testcasecount="2" result="Failed" site="Child" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.080887" total="2" passed="1" failed="1"
inconclusive="0" skipped="0" asserts="2">
        <properties>
            <property name="Requirement" value="DEV-771" />
          </properties><failure>
        <message><![CDATA[One or more child tests had errors]]></message>
      </failure>
      <test-case id="1001" name="SubtractTest(1)" fullname="TestClass.SubtractTest(1)" methodname="
SubtractTest" classname="TestClass" runstate="Runnable" seed="1166833138" result="Failed" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.043525" asserts="1">
        <failure>
          <message><![CDATA[  Expected: 10
But was:  1
]]></message>
          <stack-trace><![CDATA[at TestClass.SubtractTest(Int32 x) in C:\Users\Sergio\x\TestClass.cs:line 13
]]></stack-trace>
        </failure>
      </test-case>
      <test-case id="1002" name="SubtractTest(10)" fullname="TestClass.SubtractTest(10)" methodname="
SubtractTest" classname="TestClass" runstate="Runnable" seed="1003146807" result="Failed" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="12.000098" asserts="1" />
```

```
    </test-suite>
  </test-suite>
  <test-suite type="TestSuite" id="1022" name="x" fullname="x" runstate="Runnable" testcasecount="12" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.015218" total="12"
passed="12" failed="0" inconclusive="0" skipped="0" asserts="12">
    <test-suite type="TestFixture" id="1004" name="CalculatorTests" fullname="x.CalculatorTests" classname="x.
CalculatorTests" runstate="Runnable" testcasecount="12" result="Passed" start-time="2016-12-26 14:36:03Z" end-
time="2016-12-26 14:36:03Z" duration="0.014979" total="12" passed="12" failed="0" inconclusive="0" skipped="
0" asserts="12">
      <test-suite type="ParameterizedMethod" id="1008" name="CanAddNumbers" fullname="x.CalculatorTests.
CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-
time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.004228" total="3" passed="3" failed="
0" inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="Requirement" value="DEV-771" />
          </properties>
        <test-case id="1005" name="CanAddNumbers(1,1,2)" fullname="x.CalculatorTests.CanAddNumbers(1,1,2)"
methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1846389584" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.001194" asserts="1" />
        <test-case id="1006" name="CanAddNumbers(-1,-1,-2)" fullname="x.CalculatorTests.CanAddNumbers(-1,-1,
-2)" methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1113780989" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000067" asserts="1" />
        <test-case id="1007" name="CanAddNumbers(100,5,105)" fullname="x.CalculatorTests.CanAddNumbers
(100,5,105)" methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1585332966"
result="Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000103"
asserts="1" />
      </test-suite>
      <test-suite type="ParameterizedMethod" id="1020" name="CanDivide" fullname="x.CalculatorTests.
CanDivide" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.004041" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="Requirement" value="DEV-771" />
          </properties>
        <test-case id="1017" name="CanDivide(1,1,1)" fullname="x.CalculatorTests.CanDivide(1,1,1)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="1285501252" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000354" asserts="1" />
        <test-case id="1018" name="CanDivide(-1,-1,1)" fullname="x.CalculatorTests.CanDivide(-1,-1,1)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="1436436719" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000073" asserts="1" />
        <test-case id="1019" name="CanDivide(100,5,20)" fullname="x.CalculatorTests.CanDivide(100,5,20)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="213310888" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000060" asserts="1" />
      </test-suite>
      <test-suite type="ParameterizedMethod" id="1016" name="CanMultiply" fullname="x.CalculatorTests.
CanMultiply" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.002759" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
        <test-case id="1013" name="CanMultiply(1,1,1)" fullname="x.CalculatorTests.CanMultiply(1,1,1)"
methodname="CanMultiply" classname="x.CalculatorTests" runstate="Runnable" seed="1192735127" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000331" asserts="1">
                  <properties>
            <property name="label" value="multiplication" />
          </properties>
          </test-case>
        <test-case id="1014" name="CanMultiply(-1,-1,1)" fullname="x.CalculatorTests.CanMultiply(-1,-1,1)"
methodname="CanMultiply" classname="x.CalculatorTests" runstate="Runnable" seed="39988064" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000059" asserts="1">
                  <properties>
            <property name="label" value="multiplication" />
          </properties>
          </test-case>
        <test-case id="1015" name="CanMultiply(100,5,500)" fullname="x.CalculatorTests.CanMultiply
(100,5,500)" methodname="CanMultiplyAgain" classname="x.CalculatorTests" runstate="Runnable" seed="
1462346243" result="Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="
0.000052" asserts="1">
                  <properties>
            <property name="requirement" value="DEV-34" />
          </properties>
          </test-case>
      </test-suite>
```

```
      <test-suite type="ParameterizedMethod" id="1012" name="CanSubtract" fullname="x.CalculatorTests.
CanSubtract" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.002827" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="requirement" value="DEV-328" />
          </properties>
        <test-case id="1009" name="CanSubtract(1,1,0)" fullname="x.CalculatorTests.CanSubtract(1,1,0)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="1019357734" result="Failed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000303" asserts="1">
          <failure>
            <message><![CDATA[Error subtracting]]></message>
          </failure>
      </test-case>
        <test-case id="1010" name="CanSubtract(-1,-1,0)" fullname="x.CalculatorTests.CanSubtract(-1,-1,0)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="1322022615" result="Failed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000056" asserts="1" >
              <failure>
        <message><![CDATA[Error subtracting]]></message>
      </failure>
      </test-case>
        <test-case id="1011" name="CanSubtract(100,5,95)" fullname="x.CalculatorTests.CanSubtract(100,5,95)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="4493553" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000053" asserts="1" />
      </test-suite>
    </test-suite>
  </test-suite>
</test-suite>
</test-run>
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/nunit?projectKey=XTP

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/nunit?testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/nunit?projectKey=XTP&testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/nunit?projectKey=XTP&testPlanKey=XTP-12&revision=v2.1.0

Responses

200 OK : ***application/json*** : Successful. The results where successfully imported to Jira.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  }
}
```

400 BAD_REQUEST : ***application/json :*** *Returns the error.*

401 UNAUTHORIZED : ***application/json*** *: The Xray license is not valid.*

500 INTERNAL SERVER ERROR : ***application/json*** *: An internal error occurred when importing execution results.*

NUnit XML results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution and Test fields. It allows you to send one XML file (the NUnit report) and two JSON files similar to the one Jira uses to create new issues. For more information about the JSON format, check the Jira documentation here.

As of the creation of Test issues, some field values in the JSON file will follow some rules:

- Issue Type: Overridden by the Test issue type;
- Summary: Overridden by the internally generated summary;
- Parent: This field is ignored;
- Project: If the there is not a Test issue with the same Generic Test Definition, then it will be created in the provided project;
- Generic Test Definition: Not allowed to provide a value for this field.

Import the execution results created with the NUnit XML output formatter. For more information please check the documentation about NUnit integration.

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

NUnit Report XML

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<test-run id="0" testcasecount="14" total="14" passed="13" failed="1" inconclusive="0" skipped="0" asserts="
14" result="Failed" portable-engine-version="3.3.0.0" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26
14:36:03Z" duration="0.140400">
<test-suite type="Assembly" id="1021" name="x, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null"
fullname="x, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null" runstate="Runnable" testcasecount="14"
result="Failed" site="Child" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="
0.110549" total="14" passed="13" failed="1" inconclusive="0" skipped="0" asserts="14">
  <settings>
    <setting name="WorkDirectory" value="C:\Users\Sergio\x" />
  </settings>
  <failure>
    <message><![CDATA[One or more child tests had errors]]></message>
  </failure>
  <test-suite type="TestFixture" id="1000" name="TestClass" fullname="TestClass" classname="TestClass"
runstate="Runnable" testcasecount="2" result="Failed" site="Child" start-time="2016-12-26 14:36:03Z" end-
time="2016-12-26 14:36:03Z" duration="0.084668" total="2" passed="1" failed="1" inconclusive="0" skipped="0"
asserts="2">
    <failure>
      <message><![CDATA[One or more child tests had errors]]></message>
    </failure>
    <test-suite type="ParameterizedMethod" id="1003" name="SubtractTest" fullname="TestClass.SubtractTest"
classname="TestClass" runstate="Runnable" testcasecount="2" result="Failed" site="Child" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.080887" total="2" passed="1" failed="1"
inconclusive="0" skipped="0" asserts="2">
        <properties>
            <property name="Requirement" value="DEV-771" />
          </properties><failure>
        <message><![CDATA[One or more child tests had errors]]></message>
      </failure>
      <test-case id="1001" name="SubtractTest(1)" fullname="TestClass.SubtractTest(1)" methodname="
SubtractTest" classname="TestClass" runstate="Runnable" seed="1166833138" result="Failed" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.043525" asserts="1">
        <failure>
          <message><![CDATA[  Expected: 10
But was:  1
]]></message>
          <stack-trace><![CDATA[at TestClass.SubtractTest(Int32 x) in C:\Users\Sergio\x\TestClass.cs:line 13
]]></stack-trace>
        </failure>
      </test-case>
      <test-case id="1002" name="SubtractTest(10)" fullname="TestClass.SubtractTest(10)" methodname="
SubtractTest" classname="TestClass" runstate="Runnable" seed="1003146807" result="Failed" start-time="2016-12-
26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="12.000098" asserts="1" />
    </test-suite>
  </test-suite>
```

```
  <test-suite type="TestSuite" id="1022" name="x" fullname="x" runstate="Runnable" testcasecount="12" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.015218" total="12"
passed="12" failed="0" inconclusive="0" skipped="0" asserts="12">
    <test-suite type="TestFixture" id="1004" name="CalculatorTests" fullname="x.CalculatorTests" classname="x.
CalculatorTests" runstate="Runnable" testcasecount="12" result="Passed" start-time="2016-12-26 14:36:03Z" end-
time="2016-12-26 14:36:03Z" duration="0.014979" total="12" passed="12" failed="0" inconclusive="0" skipped="
0" asserts="12">
      <test-suite type="ParameterizedMethod" id="1008" name="CanAddNumbers" fullname="x.CalculatorTests.
CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-
time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.004228" total="3" passed="3" failed="
0" inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="Requirement" value="DEV-771" />
          </properties>
        <test-case id="1005" name="CanAddNumbers(1,1,2)" fullname="x.CalculatorTests.CanAddNumbers(1,1,2)"
methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1846389584" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.001194" asserts="1" />
        <test-case id="1006" name="CanAddNumbers(-1,-1,-2)" fullname="x.CalculatorTests.CanAddNumbers(-1,-1,
-2)" methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1113780989" result="
Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000067" asserts="1" />
        <test-case id="1007" name="CanAddNumbers(100,5,105)" fullname="x.CalculatorTests.CanAddNumbers
(100,5,105)" methodname="CanAddNumbers" classname="x.CalculatorTests" runstate="Runnable" seed="1585332966"
result="Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000103"
asserts="1" />
      </test-suite>
      <test-suite type="ParameterizedMethod" id="1020" name="CanDivide" fullname="x.CalculatorTests.
CanDivide" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.004041" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="Requirement" value="DEV-771" />
          </properties>
        <test-case id="1017" name="CanDivide(1,1,1)" fullname="x.CalculatorTests.CanDivide(1,1,1)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="1285501252" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000354" asserts="1" />
        <test-case id="1018" name="CanDivide(-1,-1,1)" fullname="x.CalculatorTests.CanDivide(-1,-1,1)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="1436436719" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000073" asserts="1" />
        <test-case id="1019" name="CanDivide(100,5,20)" fullname="x.CalculatorTests.CanDivide(100,5,20)"
methodname="CanDivide" classname="x.CalculatorTests" runstate="Runnable" seed="213310888" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000060" asserts="1" />
      </test-suite>
      <test-suite type="ParameterizedMethod" id="1016" name="CanMultiply" fullname="x.CalculatorTests.
CanMultiply" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.002759" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
        <test-case id="1013" name="CanMultiply(1,1,1)" fullname="x.CalculatorTests.CanMultiply(1,1,1)"
methodname="CanMultiply" classname="x.CalculatorTests" runstate="Runnable" seed="1192735127" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000331" asserts="1">
                <properties>
            <property name="label" value="multiplication" />
          </properties>
          </test-case>
        <test-case id="1014" name="CanMultiply(-1,-1,1)" fullname="x.CalculatorTests.CanMultiply(-1,-1,1)"
methodname="CanMultiply" classname="x.CalculatorTests" runstate="Runnable" seed="39988064" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000059" asserts="1">
                <properties>
            <property name="label" value="multiplication" />
          </properties>
          </test-case>
        <test-case id="1015" name="CanMultiply(100,5,500)" fullname="x.CalculatorTests.CanMultiply
(100,5,500)" methodname="CanMultiplyAgain" classname="x.CalculatorTests" runstate="Runnable" seed="
1462346243" result="Passed" start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="
0.000052" asserts="1">
                <properties>
            <property name="requirement" value="DEV-34" />
          </properties>
          </test-case>
      </test-suite>
      <test-suite type="ParameterizedMethod" id="1012" name="CanSubtract" fullname="x.CalculatorTests.
CanSubtract" classname="x.CalculatorTests" runstate="Runnable" testcasecount="3" result="Passed" start-time="
```

```
2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.002827" total="3" passed="3" failed="0"
inconclusive="0" skipped="0" asserts="3">
          <properties>
            <property name="requirement" value="DEV-328" />
          </properties>
        <test-case id="1009" name="CanSubtract(1,1,0)" fullname="x.CalculatorTests.CanSubtract(1,1,0)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="1019357734" result="Failed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000303" asserts="1">
          <failure>
            <message><![CDATA[Error subtracting]]></message>
          </failure>
      </test-case>
        <test-case id="1010" name="CanSubtract(-1,-1,0)" fullname="x.CalculatorTests.CanSubtract(-1,-1,0)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="1322022615" result="Failed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000056" asserts="1" >
              <failure>
        <message><![CDATA[Error subtracting]]></message>
      </failure>
      </test-case>
        <test-case id="1011" name="CanSubtract(100,5,95)" fullname="x.CalculatorTests.CanSubtract(100,5,95)"
methodname="CanSubtract" classname="x.CalculatorTests" runstate="Runnable" seed="4493553" result="Passed"
start-time="2016-12-26 14:36:03Z" end-time="2016-12-26 14:36:03Z" duration="0.000053" asserts="1" />
      </test-suite>
    </test-suite>
  </test-suite>
</test-suite>
</test-run>
```

Test Exec Info JSON

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for nunit Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ]
        }
}
```

Test Info JSON

```
{
    "fields": {
        "description": "Automated Test",
        "priority" : {
                "id": "10"
        },
        "labels": [
            "Testing",
            "Automation"
        ]
    }
}
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" -F "info=@testExec.json" -F "testInfo=@test.json" http://yourserver/rest/raven/1.0/import/execution/nunit/multipart

Responses

200 OK **: application/json :** Successful. The results where successfully imported to Jira. The following Test issues were also created with success.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "success": [
            {
                "self": "http://localhost:8080/rest/api/2/issue/10201",
                "id": "10201",
                "key": "XNP-25"
            },
            {
                "self": "http://localhost:8080/rest/api/2/issue/10202",
                "id": "10202",
                "key": "XNP-26"
            }
        ]
    }
}
```

200 OK : **application/json**: Some results where successfully imported to Jira. But the following Test issues failed to be created due to the following reasons.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "error": [
            {
                "messages": [
                    "Field 'customfield_10005' cannot be set. It is not on the appropriate screen, or
unknown."
                ],
                "testDefinition": "x.CalculatorTests.CanMultiply"
            }
        ]
    }
}
```

400 BAD_REQUEST **: application/json :** Returns the error.

401 UNAUTHORIZED **: application/json :** The Xray license is not valid.

500 INTERNAL SERVER ERROR **: application/json :** An internal error occurred when importing execution results.

xUnit XML results

After executing xUnit tests, you must import the outputted XML execution results to Jira using the following endpoint:

Import the execution results created with the xUnit XML output formatter. For more information please check the documentation about xUnit integration.

Request

QUERY PARAMETERS

parameter	type	description
projectKey	*String*	- key of the project where the Test Execution (if the **testExecKey** parameter wasn't provided) and the tests (if they aren't created yet) are going to be created.
testExecKey	String	- key of the Test Execution.
testPlanKey	String	- key of the Test Plan; if you specify the Test Plan, the Tests will be added automatically to the Test Plan if they're not part of it.
testEnvironments	String	- a string containing a list of test environments separated by ";"
revision	String	- source code and documentation version used in the test execution.
fixVersion	String	- the Fix Version associated with the test execution (it supports only one value).

multipart/form-data:

***"file"** : a **MultipartFormParam** containing a **XML file** to import.*

Example

xUnit Report XML

```
<?xml version="1.0" encoding="utf-8"?>
<assemblies timestamp="08/10/2020 14:59:44">
  <assembly name="/Users/Projects/OurXunitTestsProject/OurXunitTestsProject/bin/Debug/netcoreapp3.1
/OurXunitTestsProject.dll" run-date="2020-08-10" run-time="14:59:44" total="9" passed="5" failed="3" skipped="
1" time="23.519" errors="0">
    <errors />
    <collection total="1" passed="0" failed="0" skipped="1" name="Test collection for OurXunitTestsProject.
OurSkippableClass" time="0.001">
      <test name="OurXunitTestsProject.OurSkippableClass.AddTwoPlusTwo_EqualsFour" type="OurXunitTestsProject.
OurSkippableClass" method="AddTwoPlusTwo_EqualsFour" time="0.0010000" result="Skip">
        <output>The calculation is wrong. It'll be fixed later.
</output>
        <traits />
      </test>
    </collection>
    <collection total="2" passed="1" failed="1" skipped="0" name="Test collection for OurXunitTestsProject.
OurTestClass" time="0.009">
      <test name="OurXunitTestsProject.OurTestClass.AddTwoPlusThree_EqualsFive" type="OurXunitTestsProject.
OurTestClass" method="AddTwoPlusThree_EqualsFive" time="0.0077767" result="Fail">
        <failure>
          <message>Assert.Equal() Failure
Expected: 5
Actual:   6</message>
          <stack-trace>   at OurXunitTestsProject.OurTestClass.AddTwoPlusThree_EqualsFive() in /Users/Projects
/OurXunitTestsProject/OurXunitTestsProject/OurTestClass.cs:line 19</stack-trace>
        </failure>
        <traits />
      </test>
      <test name="OurXunitTestsProject.OurTestClass.AddTwoPlusTwo_EqualsFour" type="OurXunitTestsProject.
OurTestClass" method="AddTwoPlusTwo_EqualsFour" time="0.0010802" result="Pass">
        <traits />
      </test>
    </collection>
    <collection total="4" passed="3" failed="1" skipped="0" name="Test collection for OurXunitTestsProject.
```

```
OurTheoryClass" time="6.005">
      <test name="OurXunitTestsProject.OurTheoryClass.SumTwoNumbers(value1: -2, value2: 3, expected: 0)"
type="OurXunitTestsProject.OurTheoryClass" method="SumTwoNumbers" time="1.5030442" result="Fail">
        <failure>
          <message>Assert.Equal() Failure
Expected: 0
Actual:   1</message>
          <stack-trace>   at OurXunitTestsProject.OurTheoryClass.SumTwoNumbers(Int32 value1, Int32 value2,
Int32 expected) in /Users/Projects/OurXunitTestsProject/OurXunitTestsProject/OurTheoryClass.cs:line 21</stack-
trace>
        </failure>
        <traits>
          <trait name="requirement" value="CALC-2088" />
          <trait name="labels" value="calculator theory" />
        </traits>
      </test>
      <test name="OurXunitTestsProject.OurTheoryClass.SumTwoNumbers(value1: -2147483648, value2: -1,
expected: 2147483647)" type="OurXunitTestsProject.OurTheoryClass" method="SumTwoNumbers" time="1.5005778"
result="Pass">
        <traits>
          <trait name="requirement" value="CALC-2088" />
          <trait name="labels" value="calculator theory" />
        </traits>
      </test>
      <test name="OurXunitTestsProject.OurTheoryClass.SumTwoNumbers(value1: 1, value2: 2, expected: 3)" type="
OurXunitTestsProject.OurTheoryClass" method="SumTwoNumbers" time="1.5002999" result="Pass">
        <traits>
          <trait name="requirement" value="CALC-2088" />
          <trait name="labels" value="calculator theory" />
        </traits>
      </test>
      <test name="OurXunitTestsProject.OurTheoryClass.SumTwoNumbers(value1: -4, value2: -6, expected: -10)"
type="OurXunitTestsProject.OurTheoryClass" method="SumTwoNumbers" time="1.5013213" result="Pass">
        <traits>
          <trait name="requirement" value="CALC-2088" />
          <trait name="labels" value="calculator theory" />
        </traits>
      </test>
    </collection>
    <collection total="2" passed="1" failed="1" skipped="0" name="Test collection for OurXunitTestsProject.
OurTraitsClass" time="17.504">
      <test name="OurXunitTestsProject.OurTraitsClass.AddTwoPlusFour_EqualsSix" type="OurXunitTestsProject.
OurTraitsClass" method="AddTwoPlusFour_EqualsSix" time="10.0027784" result="Pass">
        <traits>
          <trait name="test" value="CALC-2114" />
          <trait name="requirement" value="CALC-2088" />
          <trait name="labels" value="calculator sum core" />
        </traits>
      </test>
      <test name="OurXunitTestsProject.OurTraitsClass.AddTwoPlusFive_EqualsSeven" type="OurXunitTestsProject.
OurTraitsClass" method="AddTwoPlusFive_EqualsSeven" time="7.5012751" result="Fail">
        <failure>
          <message>Assert.Equal() Failure
Expected: 7
Actual:   8</message>
          <stack-trace>   at OurXunitTestsProject.OurTraitsClass.AddTwoPlusFive_EqualsSeven() in /Users
/Projects/OurXunitTestsProject/OurXunitTestsProject/OurTraitsClass.cs:line 29</stack-trace>
        </failure>
        <traits>
          <trait name="labels" value="calculator sum core" />
        </traits>
      </test>
    </collection>
  </assembly>
</assemblies>
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/xunit?projectKey=XTP

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/xunit?testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/xunit?projectKey=XTP&testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" http://yourserver/rest/raven/1.0/import/execution/xunit?projectKey=XTP&testPlanKey=XTP-12&revision=v2.1.0

Responses

200 OK : ***application/json*** : Successful. The results were successfully imported to Jira.

Example Output

```
{
   "testExecIssue":{
      "id":"12354",
      "key":"CALC-2174",
      "self":"http://www.example.com/rest/api/2/issue/12354"
   },
   "testIssues":{
      "success":[
         {
            "id":"12302",
            "key":"CALC-2122",
            "self":"http://www.example.com/rest/api/2/issue/12302"
         },
         {
            "id":"12312",
            "key":"CALC-2132",
            "self":"http://www.example.com/rest/api/2/issue/12312"
         },
         {
            "id":"12301",
            "key":"CALC-2121",
            "self":"http://www.example.com/rest/api/2/issue/12301"
         },
         {
            "id":"12314",
            "key":"CALC-2134",
            "self":"http://www.example.com/rest/api/2/issue/12314"
         },
         {
            "id":"12242",
            "key":"CALC-2114",
            "self":"http://www.example.com/rest/api/2/issue/12242"
         },
         {
            "id":"12305",
            "key":"CALC-2125",
            "self":"http://www.example.com/rest/api/2/issue/12305"
         }
      ]
   }
}
```

400 BAD_REQUEST : ***application/json :*** *Returns the error.*

401 UNAUTHORIZED : ***application/json*** *: The Xray license is not valid.*

500 INTERNAL SERVER ERROR : ***application/json*** *: An internal error occurred when importing execution results.*

xUnit XML results Multipart

Xray provides another endpoint if you want to have control over the fields of newly created Test Executions and Tests. The endpoint accepts one XML file (the xUnit report) and two JSON files similar to the one Jira uses to create new issues, one with the fields of the Test Execution and the other with the fields of the Test. For more information about that JSON format, check the Jira documentation here.

Import the execution results created with the xUnit XML output formatter. For more information please check the documentation about xUnit integration.

Request

Example

xUnit Report XML

```
<?xml version="1.0" encoding="UTF-8"?>
<assemblies timestamp="07/31/2018 14:58:48">
  <assembly name="/Users/Projects/OurXunitTestsProject/OurXunitTestsProject/bin/Debug/netcoreapp3.1
/OurXunitTestsProject.dll" run-date="2020-08-10" run-time="15:04:09" total="15" passed="14" failed="1"
skipped="0" time="0.007" errors="0">
    <errors />
    <collection total="2" passed="1" failed="1" skipped="0" name="Test collection for xUnitDemo.SimpleTests"
time="0.070">
      <test name="xUnitDemo.SimpleTests.PassingTest" type="xUnitDemo.SimpleTests" method="PassingTest" time="
112121210.6636741" result="Pass" />
      <test name="xUnitDemo.SimpleTests.FailingTest" type="xUnitDemo.SimpleTests" method="FailingTest" time="
0.0059474" result="Fail">
        <failure exception-type="Xunit.Sdk.EqualException">
          <message><![CDATA[Assert.Equal() Failure\r\nExpected: 5\r\nActual: 4]]></message>
          <stack-trace><![CDATA[at xUnitDemo.SimpleTests.FailingTest() in C:\Users\smsf\documents\visual
studio 2015\Projects\xUnitDemo\xUnitDemo\SimpleTests.cs:line 30]]></stack-trace>
        </failure>
      </test>
    </collection>
    <collection total="13" passed="13" failed="0" skipped="0" name="Test collection for xUnitDemo.
CalculatorTests" time="0.001">
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryClassData(value1: 1, value2: 2, expected: 3)" type="
xUnitDemo.CalculatorTests" method="CanAddTheoryClassData" time="0.0002722" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryClassData(value1: -4, value2: -6, expected: -10)"
type="xUnitDemo.CalculatorTests" method="CanAddTheoryClassData" time="0.0000248" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryClassData(value1: -2, value2: 2, expected: 0)" type="
xUnitDemo.CalculatorTests" method="CanAddTheoryClassData" time="0.0000028" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryClassData(value1: -2147483648, value2: -1, expected:
2147483647)" type="xUnitDemo.CalculatorTests" method="CanAddTheoryClassData" time="0.0000017" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryMemberDataMethod(value1: 1, value2: 2, expected: 3)"
type="xUnitDemo.CalculatorTests" method="CanAddTheoryMemberDataMethod" time="0.0001179" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryMemberDataMethod(value1: -4, value2: -6, expected:
-10)" type="xUnitDemo.CalculatorTests" method="CanAddTheoryMemberDataMethod" time="0.0000054" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheoryMemberDataMethod(value1: -2, value2: 2, expected: 0)"
type="xUnitDemo.CalculatorTests" method="CanAddTheoryMemberDataMethod" time="0.0000024" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.PassingTest" type="xUnitDemo.CalculatorTests" method="
PassingTest" time="0.0000952" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheory(value1: 1, value2: 2, expected: 3)" type="xUnitDemo.
CalculatorTests" method="CanAddTheory" time="0.0001095" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheory(value1: -4, value2: -6, expected: -10)" type="
xUnitDemo.CalculatorTests" method="CanAddTheory" time="0.0000083" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheory(value1: -2, value2: 2, expected: 0)" type="xUnitDemo.
CalculatorTests" method="CanAddTheory" time="0.000002" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.CanAddTheory(value1: -2147483648, value2: -1, expected:
2147483647)" type="xUnitDemo.CalculatorTests" method="CanAddTheory" time="0.0000017" result="Pass" />
      <test name="xUnitDemo.CalculatorTests.FailingTest" type="xUnitDemo.CalculatorTests" method="
FailingTest" time="0.0000915" result="Pass" />
    </collection>
  </assembly>
</assemblies>
```

Test Exec Info JSON

```
{
    "fields": {
        "project": {
            "id": "10402"
        },
        "summary": "Test Execution for xunit Execution",
        "issuetype": {
            "id": "10007"
        },
        "components" : [
            {
            "name":"Interface"
            },
            {
            "name":"Core"
            }
        ]
    }
}
```

Test Info JSON

```
{
    "fields": {
        "description": "Automated Test",
        "priority" : {
                "id": "10"
        },
        "labels": [
            "Testing",
            "Automation"
        ]
    }
}
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@report.xml" -F "info=@testExec.json" -F "testInfo=@test.json" http://yourserver/rest/raven/1.0/import/execution/xunit/multipart

Responses

200 OK **: application/json :** Successful. The results were successfully imported to Jira. The following Test issues were also created with success.

Example Output


```
{
   "testExecIssue":{
      "id":"12357",
      "key":"CALC-2177",
      "self":"http://www.example.com/rest/api/2/issue/12357"
   },
   "testIssues":{
      "success":[
         {
            "id":"12217",
            "key":"CALC-2097",
            "self":"http://www.example.com/rest/api/2/issue/12217"
         },
         {
            "id":"12214",
            "key":"CALC-2094",
            "self":"http://www.example.com/rest/api/2/issue/12214"
         },
         {
            "id":"12213",
            "key":"CALC-2093",
            "self":"http://www.example.com/rest/api/2/issue/12213"
         },
         {
            "id":"12216",
            "key":"CALC-2096",
            "self":"http://www.example.com/rest/api/2/issue/12216"
         },
         {
            "id":"12215",
            "key":"CALC-2095",
            "self":"http://www.example.com/rest/api/2/issue/12215"
         },
         {
            "id":"12210",
            "key":"CALC-2090",
            "self":"http://www.example.com/rest/api/2/issue/12210"
         },
         {
            "id":"12211",
            "key":"CALC-2091",
            "self":"http://www.example.com/rest/api/2/issue/12211"
         }
      ]
   }
}
```


200 OK : **application/json**: Some results where successfully imported to Jira. But the following Test issues failed to be created due to the following reasons.

Example Output


```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "error": [
            {
                "messages": [
                    "Field 'customfield_10005' cannot be set. It is not on the appropriate screen, or
unknown."
                ],
                "testDefinition": "x.CalculatorTests.CanMultiply"
            }
        ]
    }
}
```


400 BAD_REQUEST **: application/json :** Returns the error.

401 UNAUTHORIZED **: application/json :** The Xray license is not valid.

500 INTERNAL SERVER ERROR **: application/json :** An internal error occurred when importing execution results.

Robot Framework XML results

After executing Robot Framework tests, you must import the output XML execution results to Jira using the following endpoint:

Import the execution results from Robot Framework XML output format.

Request

QUERY PARAMETERS

parameter	type	description
`projectKey`	*String*	- key of the project where the Test Execution (if the **testExecKey** parameter wasn't provided) and the tests (if they aren't created yet) are going to be created.
testExecKey	String	- key of the Test Execution.
testPlanKey	String	- key of the Test Plan; if you specify the Test Plan, the Tests will be added automatically to the Test Plan if they're not part of it.
testEnvironments	String	- a string containing a list of test environments separated by ";"
revision	String	- source code and documentation version used in the test execution.
fixVersion	String	- the Fix Version associated with the test execution (it supports only one value).

multipart/form-data:

***"file"** : a **MultipartFormParam** containing a **XML file** to import.*

Example

Robot Report XML


```
<?xml version="1.0" encoding="UTF-8"?>
<robot generated="20170220 14:18:54.562" generator="Robot 3.0.2 (Python 2.7.13 on win32)">
  <suite source="C:\Users\lmfv\Documents\Saco de Features\xray-1238\robot-example\robotframework-
```

```
webdemo\login_tests" id="s1" name="Login Tests">
    <suite source="C:\Users\lmfv\Documents\Saco de Features\xray-1238\robot-example\robotframework-
webdemo\login_tests\gherkin_login.robot" id="s1-s1" name="Gherkin Login">
      <test id="s1-s1-t1" name="Gherkin Valid Login">
        <kw name="Given browser is opened to login page">
          <kw name="Login Page Should Be Open" library="resource">
            <kw name="Title Should Be" library="Selenium2Library">
              <doc>Verifies that current page title equals `title`.</doc>
              <arguments>
                <arg>Log in - Your Company JIRA</arg>
              </arguments>
              <msg timestamp="20170220 14:19:07.693" level="INFO">Page title is 'Log in - Your Company JIRA'.<
/msg>
              <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:19:07.158">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:19:07.158">
            </status>
          </kw>
          <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:18:55.937">
          </status>
        </kw>
        <kw name="When user &amp;quot;admin&amp;quot; logs in with password &amp;quot;password123&amp;quot;">
          <kw name="Input Username" library="resource">
            <arguments>
              <arg>${username}</arg>
            </arguments>
            <kw name="Input Text" library="Selenium2Library">
              <doc>Types the given `text` into text field identified by `locator`.</doc>
              <arguments>
                <arg>login-form-username</arg>
                <arg>${username}</arg>
              </arguments>
              <msg timestamp="20170220 14:19:07.696" level="INFO">Typing text 'admin' into text field 'login-
form-username'</msg>
              <status status="PASS" endtime="20170220 14:19:09.314" starttime="20170220 14:19:07.696">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:09.314" starttime="20170220 14:19:07.695">
            </status>
          </kw>
          <kw name="Input Password" library="resource">
            <arguments>
              <arg>${password}</arg>
            </arguments>
            <kw name="Input Text" library="Selenium2Library">
              <doc>Types the given `text` into text field identified by `locator`.</doc>
              <arguments>
                <arg>login-form-password</arg>
                <arg>${password}</arg>
              </arguments>
              <msg timestamp="20170220 14:19:09.316" level="INFO">Typing text 'password123' into text field
'login-form-password'</msg>
              <status status="PASS" endtime="20170220 14:19:10.956" starttime="20170220 14:19:09.316">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:10.956" starttime="20170220 14:19:09.315">
            </status>
          </kw>
          <kw name="Submit Credentials" library="resource">
            <kw name="Click Button" library="Selenium2Library">
              <doc>Clicks a button identified by `locator`.</doc>
              <arguments>
                <arg>login-form-submit</arg>
              </arguments>
              <msg timestamp="20170220 14:19:10.958" level="INFO">Clicking button 'login-form-submit'.</msg>
              <status status="PASS" endtime="20170220 14:19:17.476" starttime="20170220 14:19:10.958">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:17.477" starttime="20170220 14:19:10.957">
            </status>
```

```
          </kw>
          <status status="PASS" endtime="20170220 14:19:17.478" starttime="20170220 14:19:07.695">
          </status>
        </kw>
        <kw name="Then welcome page should be open" library="resource">
          <kw name="Location Should Be" library="Selenium2Library">
            <doc>Verifies that current URL is exactly `url`.</doc>
            <arguments>
              <arg>${WELCOME URL}</arg>
            </arguments>
            <kw name="Capture Page Screenshot" library="Selenium2Library">
              <doc>Takes a screenshot of the current page and embeds it into the log.</doc>
              <msg timestamp="20170220 14:19:18.702" html="yes" level="INFO">&lt;/td&gt;&lt;/tr&gt;&lt;tr&gt;
&lt;td colspan="3"&gt;&lt;a href="selenium-screenshot-1.png"&gt;&lt;img src="selenium-screenshot-1.png"
width="800px"&gt;&lt;/a&gt;</msg>
              <status status="PASS" endtime="20170220 14:19:18.702" starttime="20170220 14:19:18.004">
              </status>
            </kw>
            <msg timestamp="20170220 14:19:18.705" level="FAIL">Location should have been 'http://localhost:
8080/secure/Dashboard.jspa' but was 'http://localhost:8080/login.jsp'</msg>
            <status status="FAIL" endtime="20170220 14:19:18.705" starttime="20170220 14:19:17.483">
            </status>
          </kw>
          <status status="FAIL" endtime="20170220 14:19:18.706" starttime="20170220 14:19:17.481">
          </status>
        </kw>
        <kw type="teardown" name="Close Browser" library="Selenium2Library">
          <doc>Closes the current browser.</doc>
          <status status="PASS" endtime="20170220 14:19:22.382" starttime="20170220 14:19:18.707">
          </status>
        </kw>
        <tags>
          <tag>WEB-1</tag>
          <tag>WEB-3</tag>
        </tags>
        <status status="FAIL" endtime="20170220 14:19:22.383" critical="yes" starttime="20170220 14:18:55.936"
>Location should have been 'http://localhost:8080/secure/Dashboard.jspa' but was 'http://localhost:8080/login.
jsp'</status>
      </test>
      <doc>A test suite with a single Gherkin style test.This test is functionally identical to the example
invalid_login.robot file.</doc>
      <status status="FAIL" endtime="20170220 14:19:22.397" starttime="20170220 14:18:54.670">
      </status>
    </suite>
    <status status="FAIL" endtime="20170220 14:22:12.549" starttime="20170220 14:18:54.567">
    </status>
  </suite>
</robot>
```


Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@output.xml" http://yourserver/rest/raven/1.0/import/execution/robot?projectKey=XTP

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@output.xml" http://yourserver/rest/raven/1.0/import/execution/robot?testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@output.xml" http://yourserver/rest/raven/1.0/import/execution/robot?projectKey=XTP&testExecKey=XNP-23

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@output.xml" http://yourserver/rest/raven/1.0/import/execution/robot?projectKey=XTP&testPlanKey=XTP-12&revision=v2.1.0

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  }
}
```

400 BAD_REQUEST : **application/json :** Returns the error.

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

Robot Framework XML results Multipart

Xray provides another endpoint if you want to create new Test Executions and have control over newly-created Test Execution and Test fields. It allows you to send one XML file (the Robot report) and a JSON similar to the one Jira uses to create new issues. For more information about that second format, check the Jira documentation here.

As of the creation of Test issues, some field values in the JSON file will follow some rules:

- Issue Type: Overridden by the Test issue type;
- Summary: Overridden by the internally generated summary;
- Parent: This field is ignored;
- Project: If the there is not a Test issue with the same Generic Test Definition, then it will be created in the provided project;
- Generic Test Definition: Not allowed to provide a value for this field.

Imports the execution results from Robot Framework XML output format. For more information please check the documentation about

Note: Currently, if you specify the Test Plan custom field, the Tests of the Test Execution will not be added automatically to the Test Plan.

Request

Example

Robot Report XML

```
<?xml version="1.0" encoding="UTF-8"?>
<robot generated="20170220 14:18:54.562" generator="Robot 3.0.2 (Python 2.7.13 on win32)">
  <suite source="C:\Users\lmfv\Documents\Saco de Features\xray-1238\robot-example\robotframework-
webdemo\login_tests" id="s1" name="Login Tests">
    <suite source="C:\Users\lmfv\Documents\Saco de Features\xray-1238\robot-example\robotframework-
webdemo\login_tests\gherkin_login.robot" id="s1-s1" name="Gherkin Login">
      <test id="s1-s1-t1" name="Gherkin Valid Login">
        <kw name="Given browser is opened to login page">
          <kw name="Login Page Should Be Open" library="resource">
            <kw name="Title Should Be" library="Selenium2Library">
              <doc>Verifies that current page title equals `title`.</doc>
              <arguments>
                <arg>Log in - Your Company JIRA</arg>
              </arguments>
              <msg timestamp="20170220 14:19:07.693" level="INFO">Page title is 'Log in - Your Company JIRA'.<
/msg>
              <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:19:07.158">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:19:07.158">
            </status>
          </kw>
          <status status="PASS" endtime="20170220 14:19:07.693" starttime="20170220 14:18:55.937">
          </status>
        </kw>
```

```
        <kw name="When user &quot;admin&quot; logs in with password &quot;password123&quot;">
          <kw name="Input Username" library="resource">
            <arguments>
              <arg>${username}</arg>
            </arguments>
            <kw name="Input Text" library="Selenium2Library">
              <doc>Types the given `text` into text field identified by `locator`.</doc>
              <arguments>
                <arg>login-form-username</arg>
                <arg>${username}</arg>
              </arguments>
              <msg timestamp="20170220 14:19:07.696" level="INFO">Typing text 'admin' into text field 'login-
form-username'</msg>
              <status status="PASS" endtime="20170220 14:19:09.314" starttime="20170220 14:19:07.696">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:09.314" starttime="20170220 14:19:07.695">
            </status>
          </kw>
          <kw name="Input Password" library="resource">
            <arguments>
              <arg>${password}</arg>
            </arguments>
            <kw name="Input Text" library="Selenium2Library">
              <doc>Types the given `text` into text field identified by `locator`.</doc>
              <arguments>
                <arg>login-form-password</arg>
                <arg>${password}</arg>
              </arguments>
              <msg timestamp="20170220 14:19:09.316" level="INFO">Typing text 'password123' into text field
'login-form-password'</msg>
              <status status="PASS" endtime="20170220 14:19:10.956" starttime="20170220 14:19:09.316">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:10.956" starttime="20170220 14:19:09.315">
            </status>
          </kw>
          <kw name="Submit Credentials" library="resource">
            <kw name="Click Button" library="Selenium2Library">
              <doc>Clicks a button identified by `locator`.</doc>
              <arguments>
                <arg>login-form-submit</arg>
              </arguments>
              <msg timestamp="20170220 14:19:10.958" level="INFO">Clicking button 'login-form-submit'.</msg>
              <status status="PASS" endtime="20170220 14:19:17.476" starttime="20170220 14:19:10.958">
              </status>
            </kw>
            <status status="PASS" endtime="20170220 14:19:17.477" starttime="20170220 14:19:10.957">
            </status>
          </kw>
          <status status="PASS" endtime="20170220 14:19:17.478" starttime="20170220 14:19:07.695">
          </status>
        </kw>
        <kw name="Then welcome page should be open" library="resource">
          <kw name="Location Should Be" library="Selenium2Library">
            <doc>Verifies that current URL is exactly `url`.</doc>
            <arguments>
              <arg>${WELCOME URL}</arg>
            </arguments>
            <kw name="Capture Page Screenshot" library="Selenium2Library">
              <doc>Takes a screenshot of the current page and embeds it into the log.</doc>
              <msg timestamp="20170220 14:19:18.702" html="yes" level="INFO">&lt;/td&gt;&lt;/tr&gt;&lt;tr&gt;
&lt;td colspan="3"&gt;&lt;a href="selenium-screenshot-1.png"&gt;&lt;img src="selenium-screenshot-1.png"
width="800px"&gt;&lt;/a&gt;</msg>
              <status status="PASS" endtime="20170220 14:19:18.702" starttime="20170220 14:19:18.004">
              </status>
            </kw>
            <msg timestamp="20170220 14:19:18.705" level="FAIL">Location should have been 'http://localhost:
8080/secure/Dashboard.jspa' but was 'http://localhost:8080/login.jsp'</msg>
            <status status="FAIL" endtime="20170220 14:19:18.705" starttime="20170220 14:19:17.483">
            </status>
```

```
          </kw>
          <status status="FAIL" endtime="20170220 14:19:18.706" starttime="20170220 14:19:17.481">
          </status>
        </kw>
        <kw type="teardown" name="Close Browser" library="Selenium2Library">
          <doc>Closes the current browser.</doc>
          <status status="PASS" endtime="20170220 14:19:22.382" starttime="20170220 14:19:18.707">
          </status>
        </kw>
        <tags>
          <tag>WEB-1</tag>
          <tag>WEB-3</tag>
        </tags>
        <status status="FAIL" endtime="20170220 14:19:22.383" critical="yes" starttime="20170220 14:18:55.936"
>Location should have been 'http://localhost:8080/secure/Dashboard.jspa' but was 'http://localhost:8080/login.
jsp'</status>
      </test>
      <doc>A test suite with a single Gherkin style test.This test is functionally identical to the example
invalid_login.robot file.</doc>
      <status status="FAIL" endtime="20170220 14:19:22.397" starttime="20170220 14:18:54.670">
      </status>
    </suite>
    <status status="FAIL" endtime="20170220 14:22:12.549" starttime="20170220 14:18:54.567">
    </status>
  </suite>
</robot>
```

Test Exec Info JSON

```
{
        "fields": {
                "project": {
                        "id": "10402"
                },
                "summary": "Test Execution for robot Execution",
                "issuetype": {
                        "id": "10007"
                },
                "components" : [
                        {
                        "name":"Interface"
                        },
                        {
                        "name":"Core"
                        }
                ]
        }
}
```

Test Info JSON

```
{
    "fields": {
        "description": "Automated Test",
        "priority" : {
                "id": "10"
        },
        "labels": [
            "Testing",
            "Automation"
        ]
    }
}
```

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@output.xml" -F "info="@testExec.json" -F "testInfo=@test.json" http://yourserver/rest/raven/1.0/import/execution/robot/multipart

Responses

200 OK : ***application/json*** *: Successful. The results where successfully imported to Jira. The following Test issues were also created with success.*

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "success": [
            {
                "self": "http://localhost:8080/rest/api/2/issue/10201",
                "id": "10201",
                "key": "XNP-25"
            },
            {
                "self": "http://localhost:8080/rest/api/2/issue/10202",
                "id": "10202",
                "key": "XNP-26"
            }
        ]
    }
}
```

200 OK : ***application/json****: Some results where successfully imported to Jira. But the* following *Test issues failed to be created due to the following reasons.*

Example Output

```
{
  "testExecIssue": {
    "id": "10200",
    "key": "XNP-24",
    "self": "http://www.example.com/jira/rest/api/2/issue/10200"
  },
  "testIssues": {
        "error": [
            {
                "messages": [
                    "Field 'customfield_10005' cannot be set. It is not on the appropriate screen, or
unknown."
                ],
                "testDefinition": "Login Tests.Gherkin Login.Gherkin Valid Login"
            }
        ]
    }
}
```

400 BAD_REQUEST : ***application/json :*** *Returns the error.*

401 UNAUTHORIZED : ***application/json*** *: The Xray license is not valid.*

500 INTERNAL SERVER ERROR : ***application/json*** *: An internal error occurred when importing execution results.*

Multiple Execution Results

In order to import multiple execution results (e.g., outputted from Calabash or Xamarin Test Cloud), you must import the bundled compressed file with multiple execution results to Jira using the following endpoint:

Import the execution results created with the Cucumber JSON output formatter. For more information, please check the Cucumber reports documentation.

Request

multipart/form-data:

"filePart" *: a* ***MultipartFormParam*** *containing a* ***compressed zip*** *file to import.*

Example Request

curl -H "Content-Type: multipart/form-data" -u admin:admin -F "file=@cucumber_results.zip" http://yourserver/rest/raven/1.0/import/execution/bundle

Responses

200 OK : **application/json** : Successful. The results where successfully imported to Jira.

Example Output

```
{
    "testExecIssue": {
           "id": "10000",
           "key": "DEMO-123",
           "self": "http://www.example.com/jira/rest/api/2/issue/10000"
     }
}
```

400 BAD_REQUEST : **application/json :** No execution results where provided**.**

401 UNAUTHORIZED : **application/json** : The Xray license is not valid.

500 INTERNAL SERVER ERROR : **application/json** : An internal error occurred when importing execution results.

How results are mapped to Test entities

Whenever importing results from some frameworks (i.e. JUnit, TestNG, NUnit, Robot framework), Xray can identify the automated test from the report /results file, based on an hardcoded criteria (such as the class name plus the class method corresponding to the automated test).

For example, in **JUnit** the *classname* and *name* properties of a *testcase* are concatenated. If no *classname* is provided then it is replaced by the the *testsuit e name* and the result will be the concatenation of the *testsuite name and testcase name.* This value becomes the *Definition* of the Generic Test to where the results are mapped.

Depending on the test automation framework, it's possible to specify the Test issue key to which report the results in the test's code.

Independently of whether the test is identified implicitly (based on some attributes present in the test result file) or explicitly (based on the provided Test key), related Test Runs are always reported against the correct Test issue. As a consequence, if you report results multiple times there won't be duplicated Test entities.

When the identification is implicit, Xray is able to create (Generic) Test entities, if needed, per each automated test; these will be reused afterwards in similar cases.

Whenever processing results from a automation framework, for each automated test result,

1. If the Test key is provided and...
    a. it exists, then create a Test Run for that Test
    b. it doesn't exist, then don't create any Test Run (since for some reason the explicitly identified Test does not exist)
2. if no Test key is provided...
    a. try to find a Test in the identified project with the same Generic Test Definition (e.g. with the same class name+class method for example)

i. if it exists, then create a Test Run for that Test
ii. if it doesn't exist, then search for a Test with the same Generic Test Definition in all JIRA projects
    1. if it exists, then create a Test Run for that Test
    2. it doesn't exist, then create a Test in the identified project (based on the project's key or the provided test execution key)

For some frameworks, including Cucumber and Behave, Tests must exist previously to the submission of results related to them.

The reason for it,resides mainly in the fact that is not possible to create the complete Test specification from the results file.