Item Resource

Item resource is used to create new items and update existing items.

Creating a New Item

The following request is used to create a new item (issue, folder or other type) and insert it into a forest.

```
POST $baseUrl/rest/structure/2.0/item/create
```

This request should upload a specification of the creation action and coordinates of where to put the result.

Example

```
{
  "item": {
    "type": "com.almworks.jira.structure:type-folder",
    "values": { "summary": "New folder name" }
  },
  "forest": {
    "spec": { "structureId": 128 },
    "version": {
      "signature": 0,
      "version": 0
    }
  },
  "items": {
    "version": {
      "signature": 0,
      "version": 0
    }
  },
  "rowId": -100,
  "under": 0,
  "after": 0,
  "before": 0,
  "parameters": {}
}
```

Parameters

Parameter (see example above)	Meaning
`item`	Defines the item being created.
`item.type`	Item type (complete key of the module that provides this item's main functionality.) Use `com.almworks.jira.structure:type-folder` for folders and `com.almworks.jira.structure:type-issue` for issues. See also: CoreItemTypes
`item.values`	A set of values for the new item. The specific fields depend on the item. For a folder, it is "summary". For other items, see examples below.
`forest.spec`	Forest specification of the forest that will receive the new item. See ForestSpec and RestForestSpec.
`forest.version`	Last known version of the forest. The reply to this call will contain the update to that version. Use zero version (as in example) to receive full forest.
`items.version`	Last known version of instance items set. The reply to this call will contain an update to the known items. Use zero version (as in example) to receive full update.
`rowId`	Temporary ID assigned to the created issue. Must be negative. You can use `-100` in most cases.
`under` / `after` / `before`	Forest coordinates to insert the new item into. See Forest Resource.

Specific parameters for main item types

Folder

This is the example of `item` parameter for a new folder:

```
"item": {
  "type": "com.almworks.jira.structure:type-folder",
  "values": { "summary": "New folder name" }
}
```

The only parameter sent is the folder name.

Issue

This is the example of `item` parameter for a new issue:

```
"item": {
  "type": "com.almworks.jira.structure:type-issue",
  "values": {
    "issue": {
      "summary": "issue summary"
    },
    "pid": 10000,
    "issuetype": "3",
    "mode": "new",
  }
}
```

The above are the minimal fields needed to create a new issue. Note that `pid` is a number, but `issuetype` is a string.

Reply Example

The following is an example of a reply.

```
{
  "successfulActions": 1,
  "itemId": "com.almworks.jira.structure:type-issue/10100",
  "oldRowIds": [-100],
  "newRowIds": [61],

  "forestUpdates": [...],
  "itemsUpdate": {...}
}
```

Most important fields are `itemId` and `newRowIds`. More on the return fields:

Field	Explanation
`successfulActions`	A number of actions successfully performed by the server. In this case, it's either 0 or 1.
`itemId`	The ID of the newly created item. See ItemIdentity.
`oldRowIds` / `newRowIds`	Provides mapping from the temporary row IDs used for uploading the action and the real row IDs obtained after the item was inserted.
`forestUpdates`	Changes to the forest since the version passed in the request.
`itemsUpdate`	Changes to the items set since the version passed in the request.

Updating an Existing Item

The following request is used to update an existing item (issue, folder or other type).

```
POST $baseUrl/rest/structure/2.0/item/update
```

Example of the request:

```
{
  "item": {
    "itemId": "10000",
    "values": {
      "summary": "New Summary"
    }
  },
  "items": {
    "version": { "signature": 0, "version": 0 }
  },
  "forest": {
    "spec": {
      "type": "clipboard"
    },
    "version": { "signature": 0, "version": 0 }
  }
}
```


Note that although the update does not depend on the forest, the low-level API in the current version requires the request to specify a forest spec and known version of items stream. If you don't need to maintain up-to-date items cache and not interested in updates to a forest where the item is located, just use empty version in **items** field and "clipboard" forest spec – like in this example.

Parameters

Parameter (see example above)	Meaning
`item.itemId`	The ID of the item. If it is just a number, like in the example, it is an issue ID. Note that it is still a **String** value that contains issue ID. Instead of a number, it can be a canonical notation of an ItemIdentity. For example, to update a folder, use `"com.almworks.jira.structure:type-folder/123"` where 123 is the folder ID.
`item.values`	A map of values to be updated. The keys are the same as when the item is created. For updating a folder, use `"summary"`.
`items.version`	Known version of the items stream. The response will contain an update based on that number. Use zeroes, as in example, when updated is not needed.
`forest.spec` and `forest.version`	Monitored forest spec and known version of that forest. The response will contain a forest update based on those values. When not needed, use a simple forest (like clipboard in this example) and zeroed version.

Reply

The reply is similar to the reply from calling `/create` method, defined above. A positive HTTP status tells that the item has been updated. There is no `"itemId"` in the response.