Daimler Supplier workshop

Swagger use cases

Table of contents

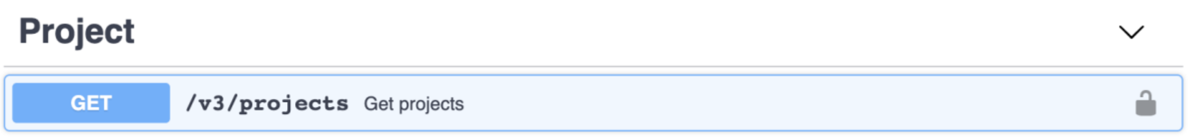
Exploring project structure

Finding the Central Issue Inbox project

There are two ways to find a project using the Swagger API. Via listing all available ones or searching using the *keyName* attribute.

Listing projects

List all available projects using **GET /v3/projects**.

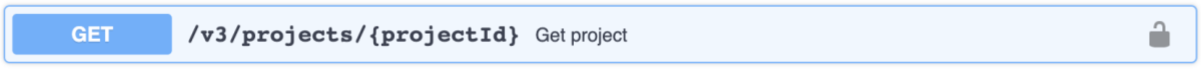


It will return the references of all visible projects for the authenticated user:

GET /v3/projects

```
[
  {
    "id": 9562,
    "name": "Master Data",
    "type": "ProjectReference"
  },
  {
    "id": 9563,
    "name": "Central Issue Inbox",
    "type": "ProjectReference"
  },
  {
    "id": 9564,
    "name": "Automated driving",
    "type": "ProjectReference"
  }
]
```

GET /v3/projects/{projectId} will provide more detailed information about a specifc project.

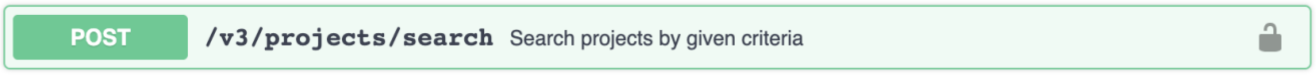

GET /v3/projects/{projectId} Get project

GET /v3/projects/9563

```
{
  "id": 9563,
  "name": "Central Issue Inbox",
  "description": "Inbox for unassigned issues",
  "descriptionFormat": "Wiki",
  "version": 2,
  "keyName": "CWS",
  "closed": false,
  "deleted": false,
  "template": false,
  "createdAt": "2020-08-05T09:03:33.982",
  "createdBy": {
    "id": 7284,
    "name": "Zoran.Tomaskovic@intland.com",
    "type": "UserReference"
  },
  "modifiedAt": "2020-08-05T09:03:35.236",
  "modifiedBy": {
    "id": 7284,
    "name": "Zoran.Tomaskovic@intland.com",
    "type": "UserReference"
  }
}
```

Searching by keyName

In case you know the *keyName* of the project you can get detailed project information using **POST /v3/projects/search**.



Request payload:

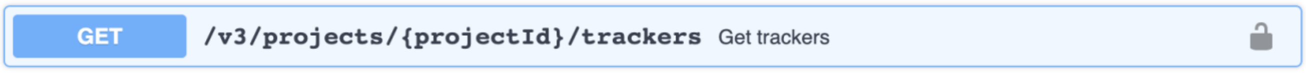
```
{
  "keyName": "CWS"
}
```

Response:

```
{
  "page": 1,
  "pageSize": 1,
  "total": 1,
  "projects": [
    {
      "id": 9563,
      "name": "Central Issue Inbox",
      "description": "Inbox for unassigned issues",
      "descriptionFormat": "Wiki",
      "version": 2,
      "keyName": "CWS",
      "closed": false,
      "deleted": false,
      "template": false,
      "createdAt": "2020-08-05T09:03:33.982",
      "createdBy": {
        "id": 7284,
        "name": "Zoran.Tomaskovic@intland.com",
        "type": "UserReference"
      },
      "modifiedAt": "2020-08-05T09:03:35.236",
      "modifiedBy": {
        "id": 7284,
        "name": "Zoran.Tomaskovic@intland.com",
        "type": "UserReference"
      }
    }
  ]
}
```

Finding the Issues tracker

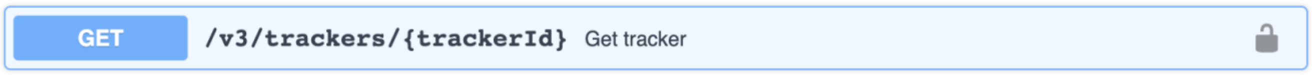
Once we got hold on the project information we can list the available trackers using **GET /v3/projects/{projectId}/trackers**



GET /v3/projects/9563/trackers

```
[
  {
    "id": 13245064,
    "name": "Issues",
    "type": "TrackerReference"
  }
]
```

For detailed information:



GET /v3/trackers/13245064

```
{
  "id": 13245064,
  "name": "Issues",
  "description": "Issue to be assigned to a domain for bugfixing",
  "descriptionFormat": "Wiki",
  "keyName": "ISSUE",
  "version": 1,
  "createdAt": "2020-08-05T09:03:34.102",
  "createdBy": {
    "id": 7284,
    "name": "Zoran.Tomaskovic@intland.com",
    "type": "UserReference"
  },
  "type": {
    "id": 2,
    "name": "Bug",
    "type": "TrackerTypeReference"
  },
  "deleted": false,
  "hidden": false,
  "color": "#ffbc6b",
  "usingWorkflow": true,
  "onlyWorkflowCanCreateNewReferringItem": true,
  "usingQuickTransitions": true,
  "defaultShowAncestorItems": false,
  "defaultShowDescendantItems": false,
  "project": {
    "id": 9563,
    "name": "Central Issue Inbox",
    "type": "ProjectReference"
  },
  "availableAsTemplate": false
}
```

Extracting the Issues tracker schema

In order to be able to create an item we need to find out what kind of fields are configured in the Issues tracker.

We can get the field references using **GET /v3/trackers/{trackerId}/fields** and the complete tracker schema using **GET /v3/trackers/{trackerId}/schema**.

The **/fields** endpoint will provide an overview listing only *FieldReference*s.

GET /v3/trackers/13245064/fields

```
[
  {
    "id": 0,
    "name": "ID",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 1,
    "name": "Tracker",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 2,
    "name": "Priority",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 3,
    "name": "Title",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  ...
  {
    "id": 7,
    "name": "Status",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  ...
  {
    "id": 1006,
    "name": "Occurrence",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  ...
  {
    "id": 1038,
    "name": "Sync Allowed",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  ...
]
```

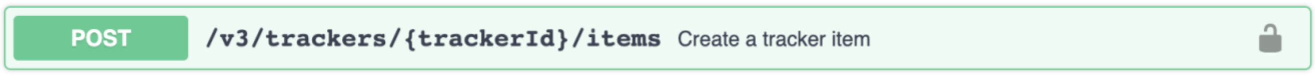
The **/schema** endpoint will provide detailed information about every field configured.

GET /v3/trackers/13245064/schema

```
[
  {
    "id": 0,
    "name": "ID",
    "type": "IntegerField",
    "hidden": false,
    "valueModel": "IntegerFieldValue",
    "mandatoryInStatuses": []
  },
  ...
  {
    "id": 3,
    "name": "Title",
    "description": "Short description of the defect",
    "type": "TextField",
    "hidden": false,
    "valueModel": "TextFieldValue",
    "mandatoryInStatuses": [
      {
        "id": 0,
        "name": "Unset",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 3,
        "name": "New / Unassigned",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 5,
        "name": "In Verification",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 7,
        "name": "Closed",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 8,
        "name": "Open In Domain",
        "type": "ChoiceOptionReference"
      }
    ],
    "trackerItemField": "name"
  },
  ...
  {
    "id": 7,
    "name": "Status",
    "description": "Current status of the issue",
    "type": "OptionChoiceField",
    "hidden": false,
    "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
    "mandatoryInStatuses": [],
    "multipleValues": false,
    "options": [
      {
        "id": 0,
        "name": "Unset",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 3,
        "name": "New / Unassigned",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 5,
        "name": "In Verification",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 7,
        "name": "Closed",
        "type": "ChoiceOptionReference"
      },
      {
        "id": 8,
        "name": "Open In Domain",
        "type": "ChoiceOptionReference"
      }
    ],
    "trackerItemField": "status",
    "referenceType": "ChoiceOptionReference"
  },
  ...
]
```

Creating a tracker item

To create a tracker item we can use the **POST /v3/trackers/{trackerId}/items** endpoint.


POST `/v3/trackers/{trackerId}/items` Create a tracker item

The hard way

In order to understand how the system works we need to construct the request body ourselves. This way we will learn how the fields are provided, what information helps us to create our *TrackerItem* model.

For first we need to find the mandatory fields in our target state. Looking through the tracker schema we can find which fields are mandatory in `"New / Unassigned"` status.

For example the *Title* field has the `"New / Unassigned"` set in *mandatoryInStatuses*:

```
{
  "id": 3,
  "name": "Title",
  "description": "Short description of the defect",
  "type": "TextField",
  "hidden": false,
  "valueModel": "TextFieldValue",
  "mandatoryInStatuses": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 3,
      "name": "New / Unassigned",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 5,
      "name": "In Verification",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 7,
      "name": "Closed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 8,
      "name": "Open In Domain",
      "type": "ChoiceOptionReference"
    }
  ],
  "trackerItemField": "name"
},
```

So there are 10 mandatory fields:

```
[
  {
    "id": 3,
    "name": "Title",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 14,
    "name": "Severity",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 80,
    "name": "Description",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 1006,
    "name": "Occurrence",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 1015,
    "name": "Verification By Test Group",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 1020,
    "name": "Send To Domain",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 1038,
    "name": "Sync Allowed",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 10001,
    "name": "Detected On Date",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 10020,
    "name": "Test Setup",
    "type": "FieldReference",
    "trackerId": 13245064
  },
  {
    "id": 10022,
    "name": "Test Environment",
    "type": "FieldReference",
    "trackerId": 13245064
  }
]
```

These fields needs to be set on *TrackerItem* model.

To get to know more about the *TrackerItem* model read our TrackerItem model structure article.

Here will only cover the setting of the mandatory fields.

Title

Field schema:

```
{
  "id": 3,
  "name": "Title",
  "description": "Short description of the defect",
  "type": "TextField",
  "hidden": false,
  "valueModel": "TextFieldValue",
  "mandatoryInStatuses": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    ...
  ],
  "trackerItemField": "name"
}
```

There are three important properties:

- *id*: the identifier of the field
- *valueModel*: the model which needs to be created in order to set value on a field
- *trackerItemField*: If there is an explicitly defined attribute in the *TrackerItem* model, the attribute name is listed here.

As the `"Title"` field has *trackerItemField* attribute we can set it pretty easily on the *TrackerItem* model.

```
{
  "name": "Training issue",
}
```

Severity

Field schema:

```
    {
      "id": 14,
      "name": "Severity",
      "description": "VoCA (Voice of Customer Audit) - &#34;customer-relevance&#34; of
  the Issue",
      "type": "OptionChoiceField",
      "hidden": false,
      "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
      "mandatoryInStatuses": [ ... ],
      "multipleValues": false,
      "options": [
        {
          "id": 0,
          "name": "Unset",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 1,
          "name": "VoCA Prio 1",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 2,
          "name": "VoCA Prio 2",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 3,
          "name": "VoCA Prio 3",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 4,
          "name": "VoCA Prio 4",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 7,
          "name": "Not Applicable",
          "type": "ChoiceOptionReference"
        }
      ],
      "trackerItemField": "severities",
      "referenceType": "ChoiceOptionReference"
    },
```

It has a *trackerItemField* as `"severities"` which will accept a list of *ChoiceOptionReference* based on the documentation on the swagger UI: CB:/v3/swagger/editor.spr

For more details checkout the Finding which model needs to be used in an explicitly defined field section.

As the **Severity** field is a choice option field all the available options are listed in the field definition from the tracker schema:

```
"options": [
  {
    "id": 0,
    "name": "Unset",
    "type": "ChoiceOptionReference"
  },
  {
    "id": 1,
    "name": "VoCA Prio 1",
    "type": "ChoiceOptionReference"
  },
  ...
```

Adding the *severity* to our *TrackerItem* model:

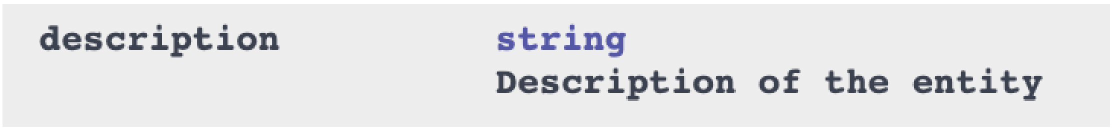
```
{
  "name": "Training issue",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ]
}
```

Description

Field schema:

```
  {
    "id": 80,
    "name": "Description",
    "description": "Detailed Issue Description",
    "type": "WikiTextField",
    "hidden": false,
    "valueModel": "WikiTextFieldValue",
    "mandatoryInStatuses": [ ... ],
    "trackerItemField": "description"
  },
```

The *description* field on *TrackerItem* model:



```
description          string
                     Description of the entity
```

Adding the *description* to our *TrackerItem* model:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ]
}
```

Occurrence

Field schema:

```
{
  "id": 1006,
  "name": "Occurrence",
  "description": "Frequency Issue occurred during testing",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [ ... ],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "Always",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 2,
      "name": "Often",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 3,
      "name": "Sometimes",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 4,
      "name": "Once",
      "type": "ChoiceOptionReference"
    }
  ],
  "referenceType": "ChoiceOptionReference"
},
```

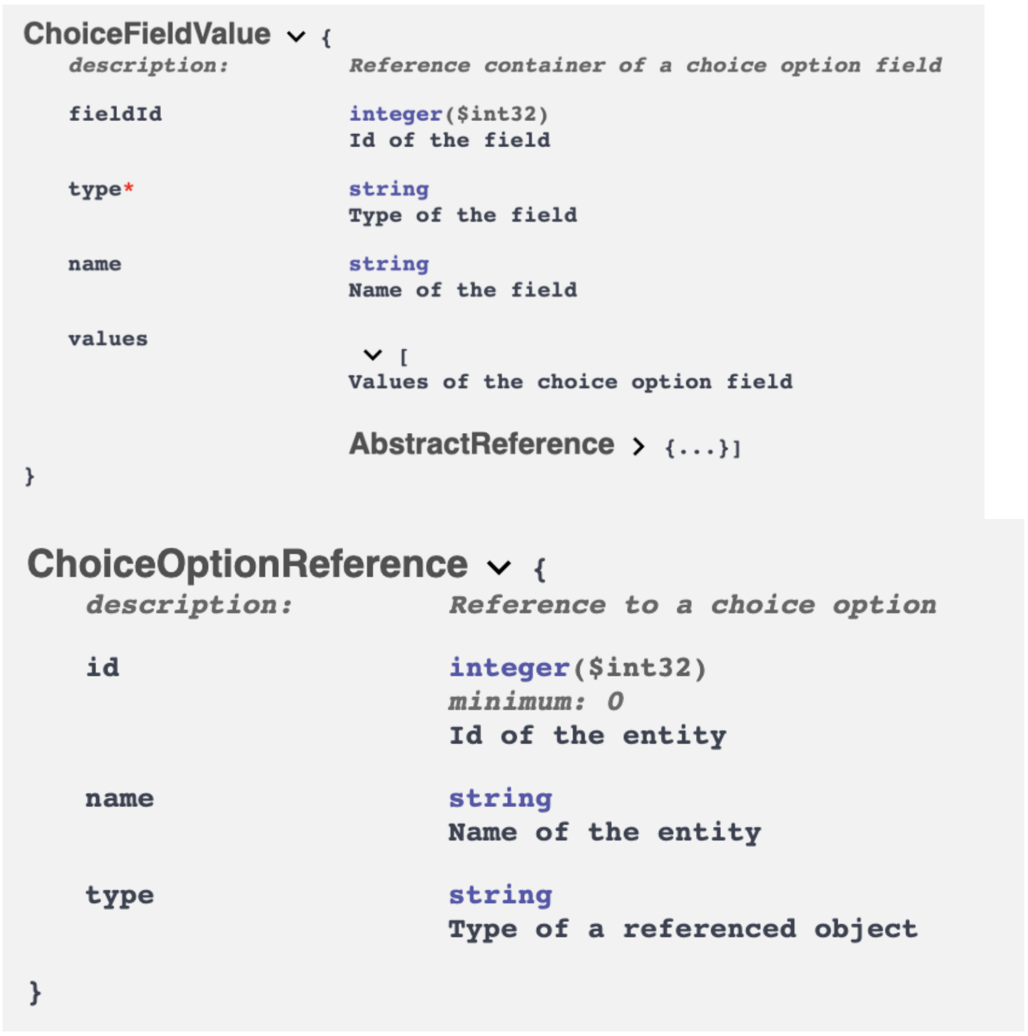
This is the first field where **we don't have** *trackerItemField* defined which means that we need to handle it a *customField.*

In this case the *valueModel* property will help us to set the value:

```
"valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
```

It tells us that we need to construct a *ChoiceFieldValue* which will contain a *ChoiceOptionReference* as an implementation of *AbstractReference*.

Let's see the definitions in the Models section:



```
ChoiceFieldValue ∨ {
    description:        Reference container of a choice option field

    fieldId             integer($int32)
                        Id of the field

    type*               string
                        Type of the field

    name                string
                        Name of the field

    values
                         ∨ [
                        Values of the choice option field

                        AbstractReference > {...}]
}

ChoiceOptionReference ∨ {
    description:        Reference to a choice option

    id                  integer($int32)
                        minimum: 0
                        Id of the entity

    name                string
                        Name of the entity

    type                string
                        Type of a referenced object

}
```

The type field will always refer to the type of the containing object.

Let's construct our *valueModel*:

```
        {
          "fieldId": 1006,
          "name": "Occurrence",
          "type": "ChoiceFieldValue",
          "values": [ ... ]
        },
```

Adding the **Occurrence** to our *TrackerItem* model with a selected option:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ],
  "customFields": [
    {
      "fieldId": 1006,
      "name": "Occurrence",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Often",
          "type": "ChoiceOptionReference"
        }
      ]
    }
  ]
}
```

Verification By Test Group

Field schema:

```
{
  "id": 1015,
  "name": "Verification By Test Group",
  "description": "Defect needs to be verified by a test group before closing",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [ ... ],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "Yes",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 2,
      "name": "No",
      "type": "ChoiceOptionReference"
    }
  ],
  "referenceType": "ChoiceOptionReference"
},
```

This field needs the same *ChoiceFieldValue<ChoiceOptionReference>* model as **Occurrence**.

Adding **Verification By Test Group** to our *TrackerItem* model with a selected option:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ],
  "customFields": [
    {
      "fieldId": 1006,
      "name": "Occurrence",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Often",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1015,
      "name": "Verification By Test Group",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 1,
          "name": "Yes",
          "type": "ChoiceOptionReference"
        }
      ]
    }
  ]
}
```

Send To Domain

Field schema:

```
{
  "id": 1020,
  "name": "Send To Domain",
  "description": "Indicates that domain Defect needs to be created",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [ ... ],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "No",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 2,
      "name": "Yes",
      "type": "ChoiceOptionReference"
    }
  ],
  "referenceType": "ChoiceOptionReference"
},
```

This field needs the same *ChoiceFieldValue<ChoiceOptionReference>* model as **Occurrence**.

Adding **Send To Domain** to our *TrackerItem* model with a selected option:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ],
  "customFields": [
    {
      "fieldId": 1006,
      "name": "Occurrence",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Often",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1015,
      "name": "Verification By Test Group",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 1,
          "name": "Yes",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1020,
      "name": "Send To Domain",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Yes",
          "type": "ChoiceOptionReference"
        }
      ]
    }
  ]
}
```

Sync Allowed

Field settings:

```
{
  "id": 1038,
  "name": "Sync Allowed",
  "description": "Allow the issue synchronisation",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "title": "Sync Allowed",
  "mandatoryInStatuses": [ ... ],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "Supplier External Tool",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 2,
      "name": "No",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 3,
      "name": "Internal Jira Project",
      "type": "ChoiceOptionReference"
    }
  ],
  "referenceType": "ChoiceOptionReference"
},
```

This field needs the same *ChoiceFieldValue<ChoiceOptionReference>* model as **Occurrence**.

Adding **Sync Allowed** to our *TrackerItem* model with a selected option:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ],
  "customFields": [
    {
      "fieldId": 1006,
      "name": "Occurrence",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Often",
          "type": "ChoiceOptionReference"
        }
      ]
    },
        {
          "fieldId": 1015,
          "name": "Verification By Test Group",
          "type": "ChoiceFieldValue",
          "values": [
            {
              "id": 1,
              "name": "Yes",
              "type": "ChoiceOptionReference"
            }
          ]
        },
        {
          "fieldId": 1020,
          "name": "Send To Domain",
          "type": "ChoiceFieldValue",
          "values": [
            {
              "id": 2,
              "name": "Yes",
              "type": "ChoiceOptionReference"
            }
          ]
        },
        {
          "fieldId": 1038,
          "name": "Sync Allowed",
          "type": "ChoiceFieldValue",
          "values": [
            {
              "id": 1,
              "name": "Supplier External Tool",
              "type": "ChoiceOptionReference"
            }
          ]
        }
      ]
    }
```

Detected On Date

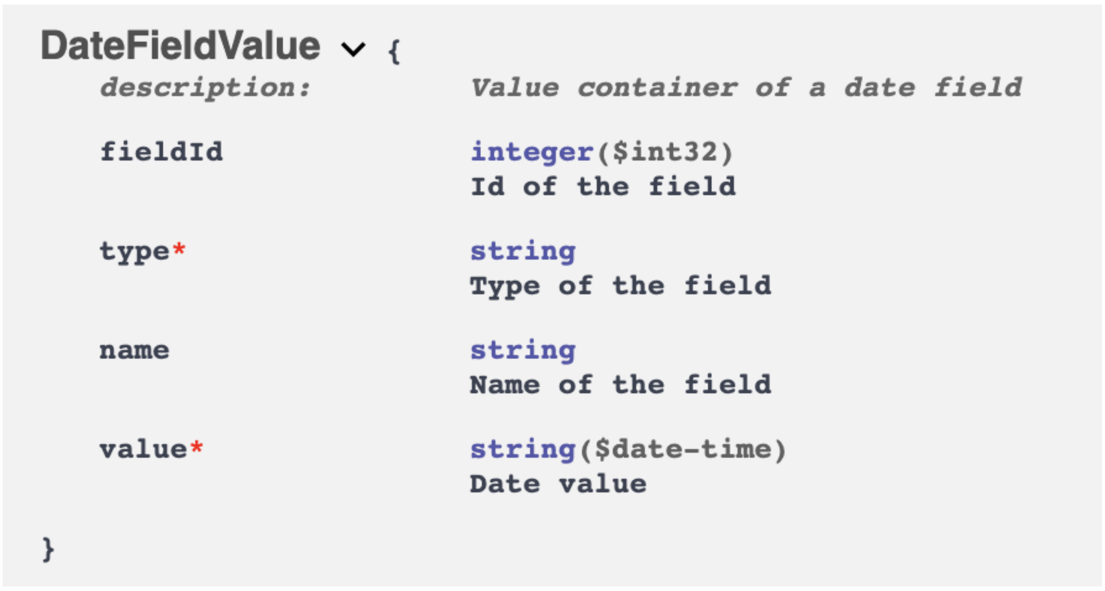
Please note that this is an automatically generated field, but we can check it for the sake of an example.

Field settings:

```
{
  "id": 10001,
  "name": "Detected On Date",
  "description": "Date Issue was detected",
  "type": "DateField",
  "hidden": false,
  "valueModel": "DateFieldValue",
  "mandatoryInStatuses": [ ... %%(background-color:initial;)]%!
},
```

This field is a *customField* as there isn't any *trackerItemField* defined and it will accept a *DateFieldValue*.

The definition from the Models section:



```
DateFieldValue ⌄ {
    description:        Value container of a date field

    fieldId             integer($int32)
                        Id of the field

    type*               string
                        Type of the field

    name                string
                        Name of the field

    value*              string($date-time)
                        Date value

}
```

Constructing the valueModel:

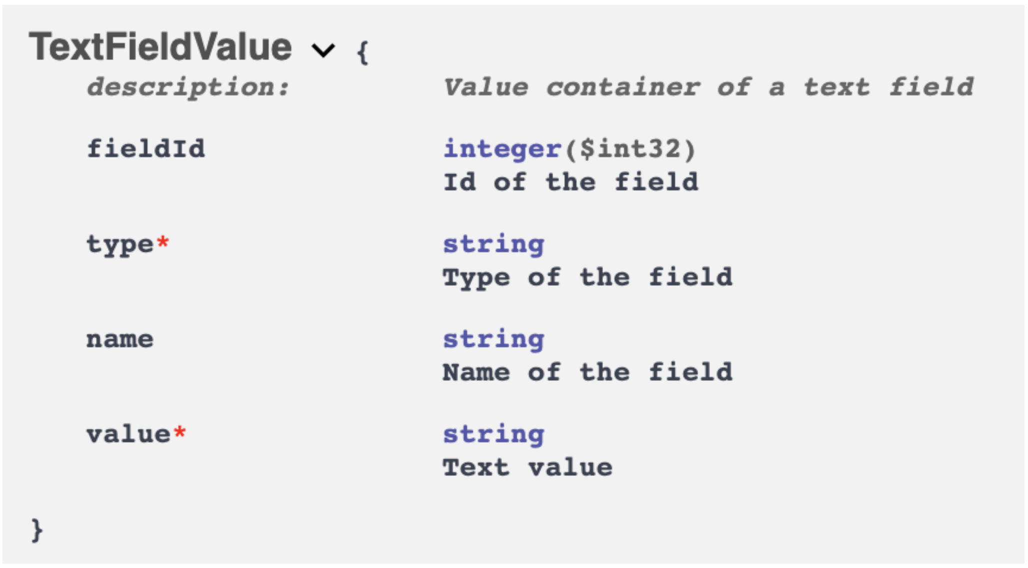
```
{
  "fieldId": 10001,
  "name": "Detected On Date",
  "value": "2020-08-10T09:00:00.000",
  "type": "DateFieldValue"
},
```

Test Setup

Field settings:

```
    {
      "id": 10020,
      "name": "Test Setup",
      "description": "List of Components with Hardware/Software-Date on the test
environment (“Quick test” data)",
      "type": "TextField",
      "hidden": false,
      "valueModel": "TextFieldValue",
      "mandatoryInStatuses": [ … ]
    },
```

It will need a *TextFieldValue* model in the *customFields* list:



```
TextFieldValue ∨ {
    description:        Value container of a text field

    fieldId             integer($int32)
                        Id of the field

    type*               string
                        Type of the field

    name                string
                        Name of the field

    value*              string
                        Text value

}
```

Constructing the valueModel:

```
      {
        "fieldId": 10020,
        "name": "Test Setup",
        "value": "{text value}",
        "type": "TextFieldValue"
      },
```

Adding **Test Setup** to our *TrackerItem* model:

```
  {
    "name": "Training issue",
    "description": "Training description",
    "severities": [
      {
        "id": 1,
        "name": "VoCA Prio 1",
        "type": "ChoiceOptionReference"
      }
    ],
    "customFields": [
      {
        "fieldId": 1006,
        "name": "Occurrence",
        "type": "ChoiceFieldValue",
        "values": [
          {
            "id": 2,
            "name": "Often",
            "type": "ChoiceOptionReference"
          }
        ]
      },
      {
        "fieldId": 1015,
        "name": "Verification By Test Group",
        "type": "ChoiceFieldValue",
        "values": [
          {
            "id": 1,
            "name": "Yes",
            "type": "ChoiceOptionReference"
          }
        ]
      },
      {
        "fieldId": 1020,
        "name": "Send To Domain",
        "type": "ChoiceFieldValue",
        "values": [
          {
            "id": 2,
            "name": "Yes",
            "type": "ChoiceOptionReference"
          }
        ]
      },
      {
        "fieldId": 1038,
        "name": "Sync Allowed",
        "type": "ChoiceFieldValue",
        "values": [
          {
            "id": 1,
            "name": "Supplier External Tool",
            "type": "ChoiceOptionReference"
          }
        ]
      },
      {
        "fieldId": 10020,
        "name": "Test Setup",
        "value": "Training setup",
        "type": "TextFieldValue"
      }
    ]
  }
```

Test Environment

Field settings:

```
    {
      "id": 10022,
      "name": "Test Environment",
      "description": "Number of Testbench or Car (Finas-Number or name) in which the
  test was carried out",
      "type": "TextField",
      "hidden": false,
      "valueModel": "TextFieldValue",
      "mandatoryInStatuses": [ ... ]
    },
```

This is a same text field as **Test Setup** so we need to provide the same structure.

Final request body

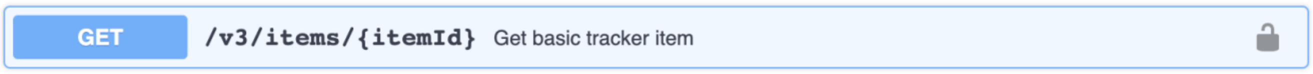
Adding **Test Environment** to our *TrackerItem* model:

```
{
  "name": "Training issue",
  "description": "Training description",
  "severities": [
    {
      "id": 1,
      "name": "VoCA Prio 1",
      "type": "ChoiceOptionReference"
    }
  ],
  "customFields": [
    {
      "fieldId": 1006,
      "name": "Occurrence",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Often",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1015,
      "name": "Verification By Test Group",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 1,
          "name": "Yes",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1020,
      "name": "Send To Domain",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 2,
          "name": "Yes",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 1038,
      "name": "Sync Allowed",
      "type": "ChoiceFieldValue",
      "values": [
        {
          "id": 1,
          "name": "Supplier External Tool",
          "type": "ChoiceOptionReference"
        }
      ]
    },
    {
      "fieldId": 10020,
      "name": "Test Setup",
      "value": "Training setup",
      "type": "TextFieldValue"
    },
    {
      "fieldId": 10022,
      "name": "Test Environment",
      "value": "Training environment",
      "type": "TextFieldValue"
    }
  ]
}
```

The easy way

The easiest way to create a tracker item is to copy an existing one.

Create an item in the codeBeamer UI and use the **GET /v3/items/{itemId}** to retrieve an existing *TrackerItem* model.


GET /v3/items/{itemId} Get basic tracker item

After deleting all the read-only fields we can just modify the values and use the model as a template for the future.

Sending issue to domain

After creating an issue we will need to set some additional fields and push it into Open In Domain state in order to create a defect for the Daimler responsible.

Using **PUT /v3/items/{itemId}/fields** endpoint we can modify only a set of field values. It uses a different approach of the **PUT /v3/items/{itemId}** endpoint where the whole tracker item state must be provided.

PUT **/v3/items/{itemId}/fields** Update field of tracker item

This endpoint accepts a list of *valueModel*s so we will need to handle the built-in fields as custom fields while constructing the request.

Setting the required fields

We need to set three fields: System, Device, Model.

System

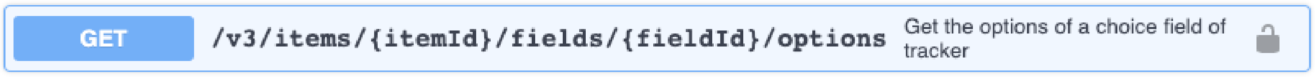
Field schema:

```
  {
    "id": 1002,
    "name": "System",
    "description": "System (defined by functions and required components) which
caused the Defect",
    "type": "TrackerItemChoiceField",
    "hidden": false,
    "valueModel": "ChoiceFieldValue<TrackerItemReference>",
    "mandatoryInStatuses": [
      {
        "id": 8,
        "name": "Open In Domain",
        "type": "ChoiceOptionReference"
      }
    ],
    "multipleValues": false,
    "referenceType": "TrackerItemReference"
  },
```

This field needs *ChoiceFieldValue<TrackerItemReference>* valueModel so we will need to put a *TrackerItemReference* into *ChoiceFieldValue*:

```
    {
      "fieldId": 1002,
      "name": "System",
      "type": "ChoiceFieldValue",
      "values": [ <TrackerItemReferences> ]
    },
```

To get the available options choice fields we can call the **GET /v3/items/{itemId}/fields/{fieldId}/options**


GET /v3/items/{itemId}/fields/{fieldId}/options Get the options of a choice field of tracker

GET /v3/items/{itemId}/fields/1002/options

```
{
  "page": 1,
  "pageSize": 25,
  "total": 2,
  "references": [
    {
      "id": 2750873,
      "name": "Rocket Science",
      "type": "TrackerItemReference"
    },
    {
      "id": 2750874,
      "name": "Warp System",
      "type": "TrackerItemReference"
    }
  ]
}
```

The final *valueModel*:

```
{
  "fieldId": 1002,
  "name": "System",
  "type": "ChoiceFieldValue",
  "values": [
    {
      "id": 2750873,
      "name": "Rocket Science",
      "type": "TrackerItemReference"
    },
  ]
},
```

Device

Field schema:

```
{
  "id": 1003,
  "name": "Device",
  "description": "Device/component which caused the Issue",
  "type": "TrackerItemChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<TrackerItemReference>",
  "mandatoryInStatuses": [
    {
      "id": 8,
      "name": "Open In Domain",
      "type": "ChoiceOptionReference"
    }
  ],
  "multipleValues": false,
  "referenceType": "TrackerItemReference"
},
```

Getting choice options:

GET /v3/items/{itemId}/fields/1003/options

```
{
  "page": 1,
  "pageSize": 25,
  "total": 2,
  "references": [
    {
      "id": 2750876,
      "name": "Rocket Engine Controller",
      "type": "TrackerItemReference"
    },
    {
      "id": 2751022,
      "name": "Rocket Tachometer",
      "type": "TrackerItemReference"
    }
  ]
}
```

The final *valueModel*:

```
{
  "fieldId": 1003,
  "name": "Device",
  "type": "ChoiceFieldValue",
  "values": [
    {
      "id": 2750876,
      "name": "Rocket Engine Controller",
      "type": "TrackerItemReference"
    },
  ]
},
```

Model

Field definition:

```
{
  "id": 1001,
  "name": "Model",
  "description": "Vehicle Model in which Issue was discovered",
  "type": "TrackerItemChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<TrackerItemReference>",
  "mandatoryInStatuses": [],
  "multipleValues": false,
  "referenceType": "TrackerItemReference"
},
```

Getting choice options:

GET /v3/items/{itemId}/fields/1001/options

```
{
  "page": 1,
  "pageSize": 25,
  "total": 3,
  "references": [
    {
      "id": 2750867,
      "name": "BR214",
      "type": "TrackerItemReference"
    },
    {
      "id": 2750869,
      "name": "BR217",
      "type": "TrackerItemReference"
    },
    {
      "id": 2750868,
      "name": "BR257",
      "type": "TrackerItemReference"
    }
  ]
}
```

The final *valueModel*:

```
{
  "fieldId": 1001,
  "name": "Model",
  "type": "ChoiceFieldValue",
  "values": [
    {
      "id": 2750869,
      "name": "BR217",
      "type": "TrackerItemReference"
    },
  ]
},
```

Final request body

So finally we can provide three values for the three mandatory fields as a *fieldValues* list.

PUT /v3/items/{itemId}/fields

```
{
    "fieldValues": [
        {
          "fieldId": 1002,
          "name": "System",
          "values": [
            {
              "id": 2750873,
              "name": "Rocket Science",
              "type": "TrackerItemReference"
            }
          ],
          "type": "ChoiceFieldValue"
        },
        {
          "fieldId": 1003,
          "name": "Device",
          "values": [
            {
              "id": 2750876,
              "name": "Rocket Engine Controller",
              "type": "TrackerItemReference"
            }
          ],
          "type": "ChoiceFieldValue"
        },
        {
          "fieldId": 1001,
          "name": "Model",
          "values": [
            {
              "id": 2750869,
              "name": "BR217",
              "type": "TrackerItemReference"
            }
          ],
          "type": "ChoiceFieldValue"
        }
    ]
}
```

Making the status transition

Status field schema:

```
{
  "id": 7,
  "name": "Status",
  "description": "Current status of the issue",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 3,
      "name": "New / Unassigned",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 5,
      "name": "In Verification",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 7,
      "name": "Closed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 8,
      "name": "Open In Domain",
      "type": "ChoiceOptionReference"
    }
  ],
  "trackerItemField": "status",
  "referenceType": "ChoiceOptionReference"
},
```

This an example where the explicitly defined fields are handled as custom field values for **PUT /v3/items/{itemId}/fields.** Because of this we need to construct a *ChoiceFieldValue*<*ChoiceOptionReference*> and provide it in the *fieldValues* list.

PUT /v3/items/{itemId}/fields

```
{
  "fieldValues": [
    {
      "fieldId": 7,
      "name": "Status",
      "values": [
        {
            "id": 8,
            "name": "Open In Domain",
            "type": "ChoiceOptionReference"
          }
      ],
      "type": "ChoiceFieldValue"
    }
  ]
}
```

Rejecting a defect

Once a defect is created in domain and the Daimler responsible did put it into In Progress status the supplier can reject it in case when it's not their responsibility to fix it.

Setting the required fields

Two fields need to be set: Supplier Status and Reject Reason

Supplier Status

Field schema

```
{
  "id": 1018,
  "name": "Supplier Status",
  "description": "Current status of the defect in the supplier workflow",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "Supplier Not Assigned",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 2,
      "name": "Supplier Rejected",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 3,
      "name": "Supplier Assigned",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 4,
      "name": "Supplier Analyzed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 5,
      "name": "Supplier Implemented",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 6,
      "name": "Supplier Verified",
      "type": "ChoiceOptionReference"
    },
    …
  ],
  "referenceType": "ChoiceOptionReference"
},
```

The *valueModel*:

```
{
  "fieldId": 1018,
  "name": "Supplier Status",
  "values": [
    {
      "id": 2,
      "name": "Supplier Rejected",
      "type": "ChoiceOptionReference"
    }
  ],
  "type": "ChoiceFieldValue"
},
```

Reject Reason

Field definition:

```
    {
      "id": 1028,
      "name": "Reject Reason",
      "description": "Reason / Justification for rejecting Defect (e.g. out of scope,
missing information)",
      "type": "OptionChoiceField",
      "hidden": false,
      "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
      "mandatoryInStatuses": [],
      "multipleValues": false,
      "options": [
        {
          "id": 0,
          "name": "Unset",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 1,
          "name": "Not In Scope",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 2,
          "name": "Missing Info",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 4,
          "name": "Not Responsible",
          "type": "ChoiceOptionReference"
        },
        {
          "id": 5,
          "name": "Not Reproducible",
          "type": "ChoiceOptionReference"
        }
      ],
      "referenceType": "ChoiceOptionReference"
    },
```

The *valueModel*:

```
          {
            "fieldId": 1028,
            "name": "Reject Reason",
            "values": [
              {
                "id": 4,
                "name": "Not Responsible",
                "type": "ChoiceOptionReference"
              }
            ],
            "type": "ChoiceFieldValue"
          }
```

Final request body

We can now set the mandatory field values.

PUT /v3/items/{itemId}/fields

```
{
    "fieldValues": [
        {
          "fieldId": 1018,
          "name": "Supplier Status",
          "values": [
            {
              "id": 2,
              "name": "Supplier Rejected",
              "type": "ChoiceOptionReference"
            }
          ],
          "type": "ChoiceFieldValue"
        },
        {
          "fieldId": 1028,
          "name": "Reject Reason",
          "values": [
            {
              "id": 4,
              "name": "Not Responsible",
              "type": "ChoiceOptionReference"
            }
          ],
          "type": "ChoiceFieldValue"
        }
    ]
}
```

Making the status transition

Status field definition:

```
        {
          "id": 7,
          "name": "Status",
          "description": "Current status of the defect",
          "type": "OptionChoiceField",
          "hidden": false,
          "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
          "mandatoryInStatuses": [],
          "multipleValues": false,
          "options": [
            {
              "id": 0,
              "name": "Unset",
              "type": "ChoiceOptionReference"
            },
            {
              "id": 1,
              "name": "Open",
              "type": "ChoiceOptionReference"
            },
            {
              "id": 4,
              "name": "Fixed",
              "type": "ChoiceOptionReference"
            },
            {
              "id": 7,
              "name": "Closed",
              "type": "ChoiceOptionReference"
            },
            {
              "id": 10,
              "name": "Verification Pending",
              "type": "ChoiceOptionReference"
            },
            {
              "id": 11,
              "name": "In Progress",
              "type": "ChoiceOptionReference"
            }
          ],
          "trackerItemField": "status",
          "referenceType": "ChoiceOptionReference"
        },
```

Request:

PUT /v3/items/{itemId}/fields

```
{
    "fieldValues": [
        {
          "fieldId": 7,
          "name": "Status",
          "values": [
            {
              "id": 1,
              "name": "Open",
              "type": "ChoiceOptionReference"
            }
          ],
          "type": "ChoiceFieldValue"
        }
    ]
}
```

Fixing a defect

Once the defect is created in domain and the Daimler responsible did put it into *In Progress* status the supplier can start to fix it. Once it fixed the defect should be pushed into *Fixed* status.

Setting the required fields

Two fields are required to full out: Supplier Status and Fixed In Release.

Supplier Status

Field schema:

```
{
  "id": 1018,
  "name": "Supplier Status",
  "description": "Current status of the defect in the supplier workflow",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [],
  "multipleValues": false,
  "options": [
    …
    {
      "id": 4,
      "name": "Supplier Analyzed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 5,
      "name": "Supplier Implemented",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 6,
      "name": "Supplier Verified",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 7,
      "name": "Supplier Closed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 8,
      "name": "Supplier Fixed",
      "type": "ChoiceOptionReference"
    }
  ],
  "referenceType": "ChoiceOptionReference"
},
```

The *valueModel*:

```
{
  "fieldId": 1018,
  "name": "Supplier Status",
  "values": [
    {
      "id": 8,
      "name": "Supplier Fixed",
      "type": "ChoiceOptionReference"
    }
  ],
  "type": "ChoiceFieldValue"
}
```

Fixed In Release

Field schema:

```
{
  "id": 10003,
  "name": "Fixed In Release",
  "description": "Supplier release in which Defect was fixed",
  "type": "TextField",
  "hidden": false,
  "valueModel": "TextFieldValue",
  "mandatoryInStatuses": [
    {
      "id": 4,
      "name": "Fixed",
      "type": "ChoiceOptionReference"
    }
  ]
},
```

The *valueModel*:

```
{
  "fieldId": 10003,
  "name": "Fixed In Release",
  "value": "Supplier release 1",
  "type": "TextFieldValue"
}
```

Final request body

We can now set the values for the two mandatory field values.

PUT /v3/items/{itemId}/fields

```
{
  "fieldValues": [
    {
      "fieldId": 1018,
      "name": "Supplier Status",
      "values": [
        {
          "id": 8,
          "name": "Supplier Fixed",
          "type": "ChoiceOptionReference"
        }
      ],
      "type": "ChoiceFieldValue"
    },
    {
      "fieldId": 10003,
      "name": "Fixed In Release",
      "value": "Supplier release 1",
      "type": "TextFieldValue"
    }
  ]
}
```

Making the status transition

Status field definition:

```
{
  "id": 7,
  "name": "Status",
  "description": "Current status of the defect",
  "type": "OptionChoiceField",
  "hidden": false,
  "valueModel": "ChoiceFieldValue<ChoiceOptionReference>",
  "mandatoryInStatuses": [],
  "multipleValues": false,
  "options": [
    {
      "id": 0,
      "name": "Unset",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 1,
      "name": "Open",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 4,
      "name": "Fixed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 7,
      "name": "Closed",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 10,
      "name": "Verification Pending",
      "type": "ChoiceOptionReference"
    },
    {
      "id": 11,
      "name": "In Progress",
      "type": "ChoiceOptionReference"
    }
  ],
  "trackerItemField": "status",
  "referenceType": "ChoiceOptionReference"
},
```

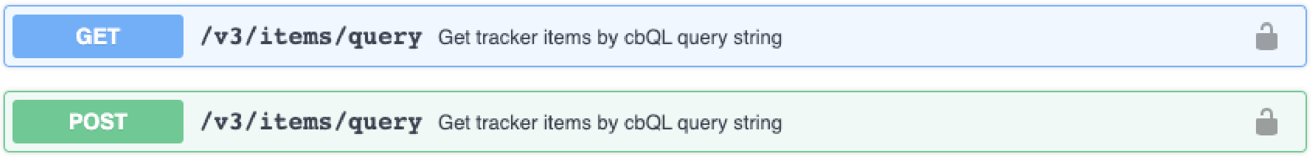
Request:

PUT /v3/items/{itemId}/fields

```
{
  "fieldValues": [
    {
      "fieldId": 7,
      "name": "Status",
      "values": [
        {
          "id": 4,
          "name": "Fixed",
          "type": "ChoiceOptionReference"
        }
      ],
      "type": "ChoiceFieldValue"
    }
  ]
}
```

Querying items

Using **GET /v3/items/query** or **POST /v3/items/query** can be used to extract detailed tracker item information using complex cbQL queries.


GET /v3/items/query Get tracker items by cbQL query string

POST /v3/items/query Get tracker items by cbQL query string

Both **GET** and **POST** methods have the same functionality. The **POST** method was introduced for use cases where the URL length limitation would compromise the **GET** query.

Getting all Issues which were modified since a given date

The following cbQL query can be used as *queryString* parameter:

```
tracker.id IN (13245407) AND modifiedAt >= '2020-08-11 00:00:00'
```

Getting all Defects which having a specific device

Finding devices in Master Data tracker:

GET /v3/projects

```
...
  {
    "id": 9562,
    "name": "Master Data",
    "type": "ProjectReference"
  },
...
```

GET /v3/projects/9562/trackers

```
...
  {
    "id": 13243948,
    "name": "Device",
    "type": "TrackerReference"
  },
...
```

GET /v3/trackers/13243948/items

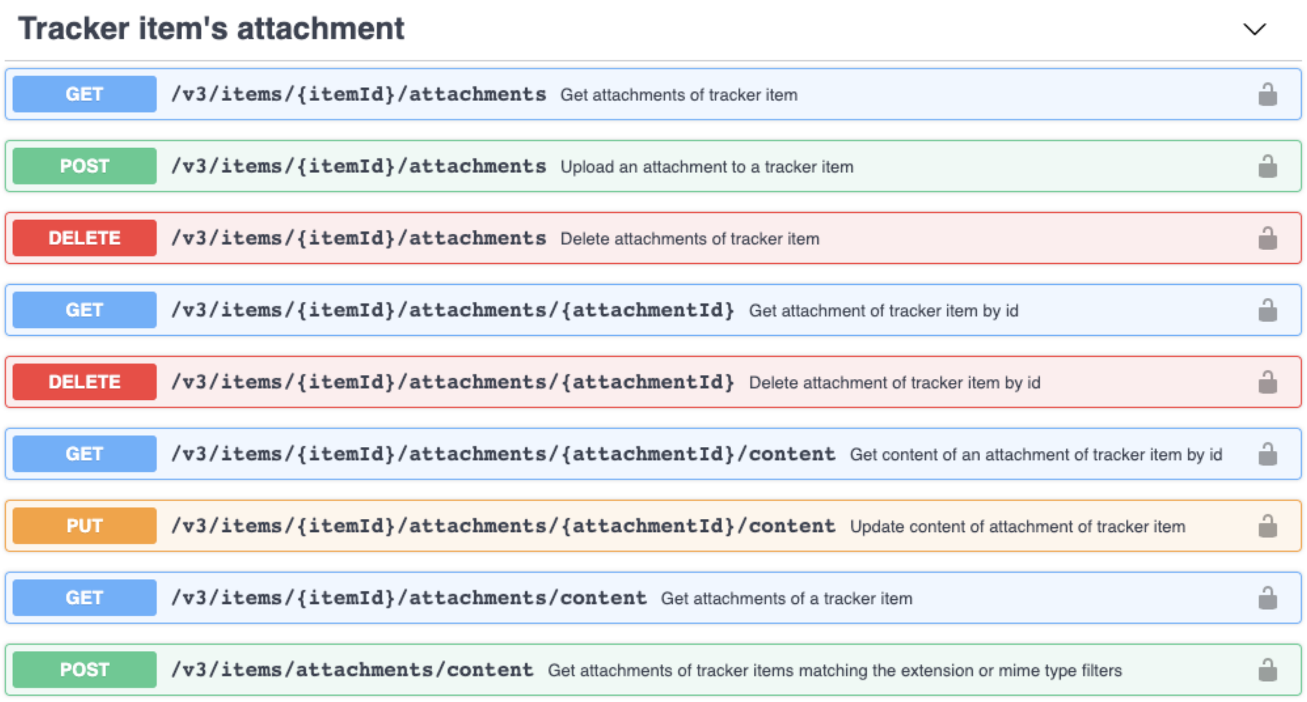
```
{
  "page": 1,
  "pageSize": 25,
  "total": 2,
  "itemRefs": [
    {
      "id": 2751022,
      "name": "Rocket Tachometer",
      "type": "TrackerItemReference"
    },
    {
      "id": 2750876,
      "name": "Rocket Engine Controller",
      "type": "TrackerItemReference"
    }
  ]
}
```

The following cbQL query can be used as *queryString* parameter:

```
tracker.id IN (13245407) AND referenceToId = 2750876
```

Attaching files

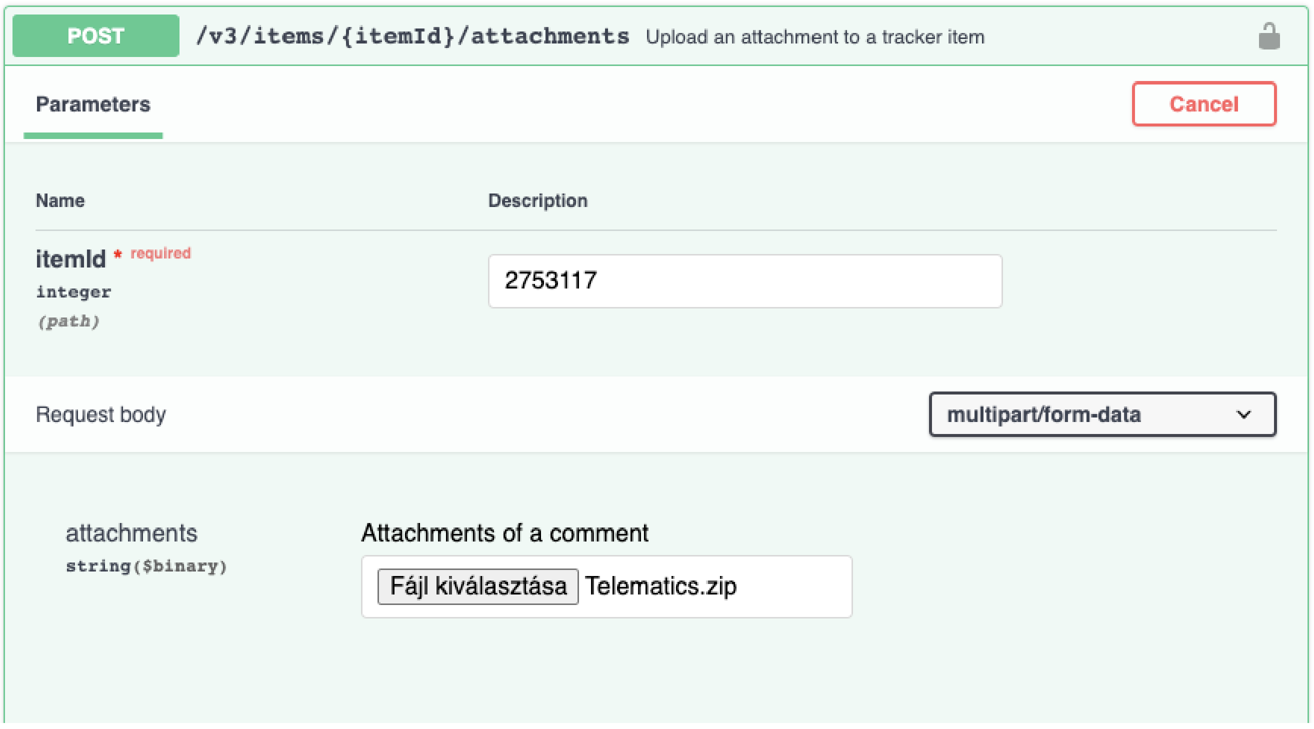
There is a rich attachment API in the Tracker item's attachment section:



Tracker item's attachment

GET	`/v3/items/{itemId}/attachments`	Get attachments of tracker item
POST	`/v3/items/{itemId}/attachments`	Upload an attachment to a tracker item
DELETE	`/v3/items/{itemId}/attachments`	Delete attachments of tracker item
GET	`/v3/items/{itemId}/attachments/{attachmentId}`	Get attachment of tracker item by id
DELETE	`/v3/items/{itemId}/attachments/{attachmentId}`	Delete attachment of tracker item by id
GET	`/v3/items/{itemId}/attachments/{attachmentId}/content`	Get content of an attachment of tracker item by id
PUT	`/v3/items/{itemId}/attachments/{attachmentId}/content`	Update content of attachment of tracker item
GET	`/v3/items/{itemId}/attachments/content`	Get attachments of a tracker item
POST	`/v3/items/attachments/content`	Get attachments of tracker items matching the extension or mime type filters

To attach a file we can use the **POST /v3/items/{itemId}/attachments** using the id of the target item and the attachment file.



POST `/v3/items/{itemId}/attachments` Upload an attachment to a tracker item

Name	Description
itemId * required integer *(path)*	2753117

Request body: multipart/form-data

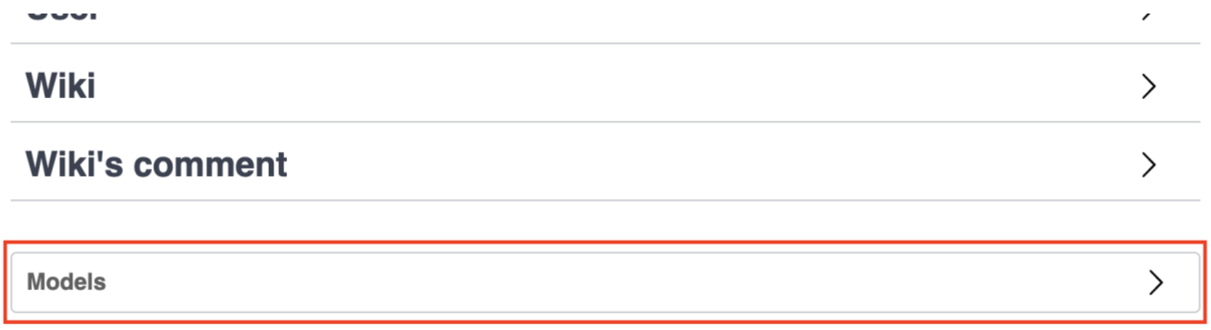
attachments
string($binary)

Attachments of a comment

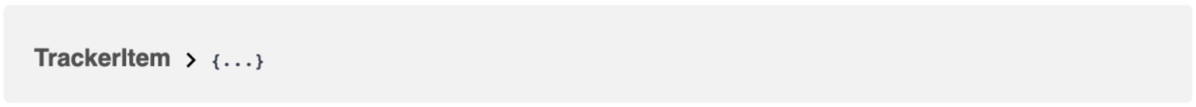
Telematics.zip

Finding which model needs to be used in an explicitly defined field

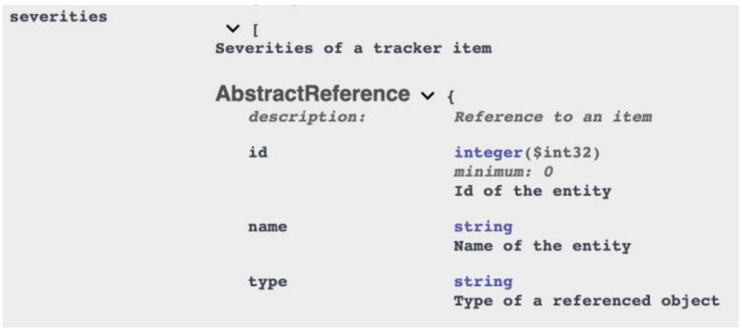
Open the Models section at the bottom of the page:



Find the TrackerItem model:



Find the severities property:

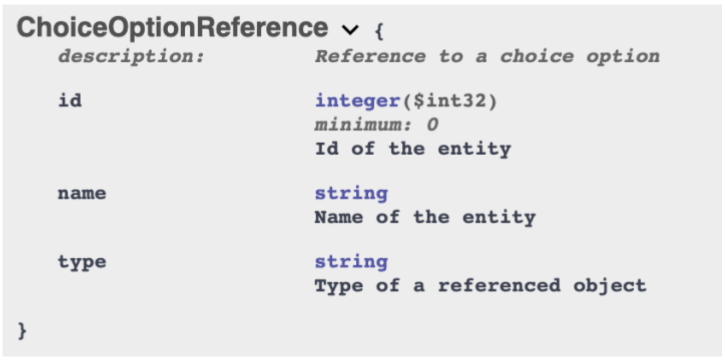


```
severities          [
                    Severities of a tracker item

                    AbstractReference {
                       description:        Reference to an item

                       id                  integer($int32)
                                           minimum: 0
                                           Id of the entity

                       name                string
                                           Name of the entity

                       type                string
                                           Type of a referenced object
```

It will accept an AbstractReference and we can find out the *referenceType* based on the field settings:

```
"referenceType": "ChoiceOptionReference"
```

Now we can look up the definition of the *ChoiceOptionReference* in the Models section:



```
ChoiceOptionReference {
   description:        Reference to a choice option

   id                  integer($int32)
                       minimum: 0
                       Id of the entity

   name                string
                       Name of the entity

   type                string
                       Type of a referenced object
}
```

This will provide the layout for our *valueModel*.