onestream

# BAI Parser Solution Release Notes

PV8.0.0 SV103

Any warranty with respect to the software or its functionality will be expressly given in the Subscription License Agreement or Software License and Services Agreement between OneStream and the warrantee. This document does not itself constitute a representation or warranty with respect to the software or any related matter.

# Table of Contents

# Release PV8.0.0 SV103

The theme of this Hotfix release is bug fixes.

> **NOTE:** Only the Release Notes were updated for this release. The BAI Parser Solution Guide has not been updated except to change the release number and copyright if needed.

## Prerequisites

- This solution is compatible with OneStream Platform v8.
- This release requires a minimum 8.0 Platform release.

## Fixed Issues

- The BAI parser will successfully map files with fund type 'D'. All related formatting, including customer references and transaction detail values, will be mapped in the file accordingly. (MP3-6809)

## Contact Support

Contact OneStream Support by registering at:

Support - OneStream Software

# Release PV800 SV102

The theme of this release are enhancements.

> **NOTE:** Only the Release Notes were updated for this release. The BAI Parser Solution Guide has not been updated except to change the release number and copyright if needed.

## Prerequisites

- This solution is compatible with OneStream Platform v8.
- This release requires a minimum 8.0 Platform release.

## New Features

None

## Enhancements

- When imported, files with warnings will still display all available information so you can review your data. (MP3-6350)
- When Account Identifier and Fund Type fields are not valid on imported files, the file will still be parsed with warnings. (MP3-6328, MP3-6348)

## Fixed Issues

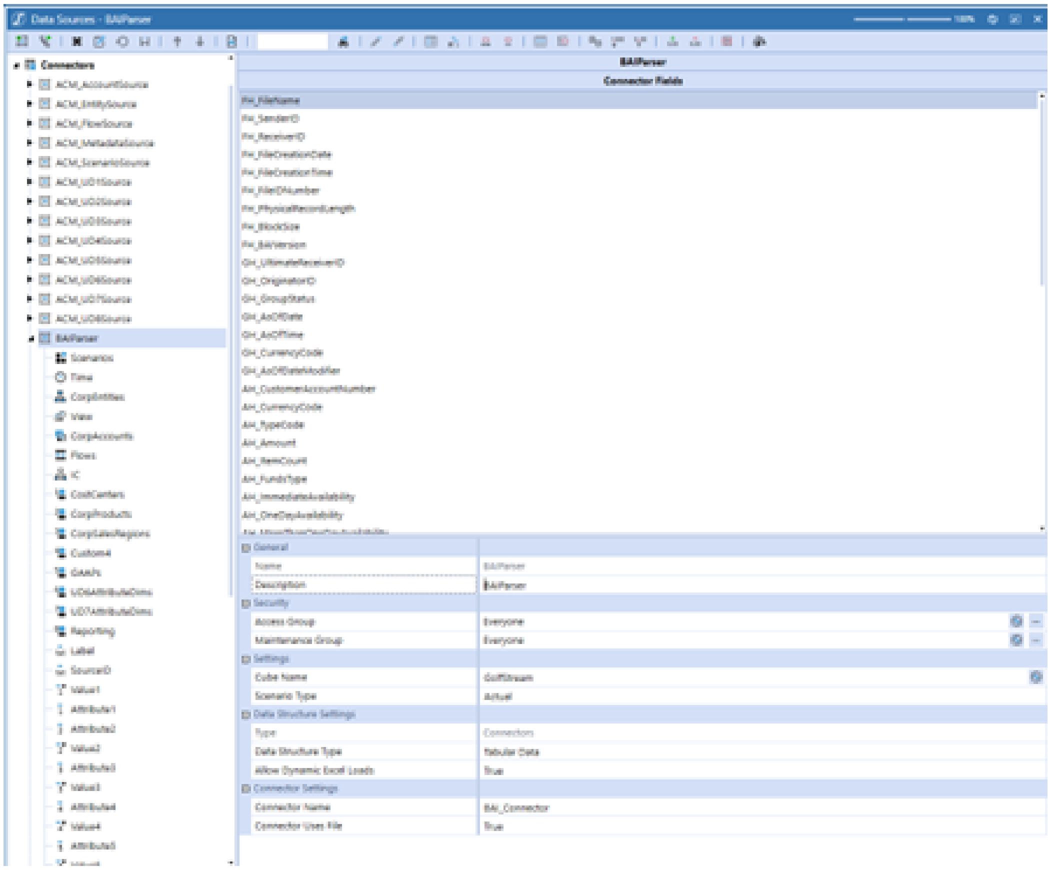
None

# Known Issues

- Automation of the data load is impacted by the Connector Uses File setting.

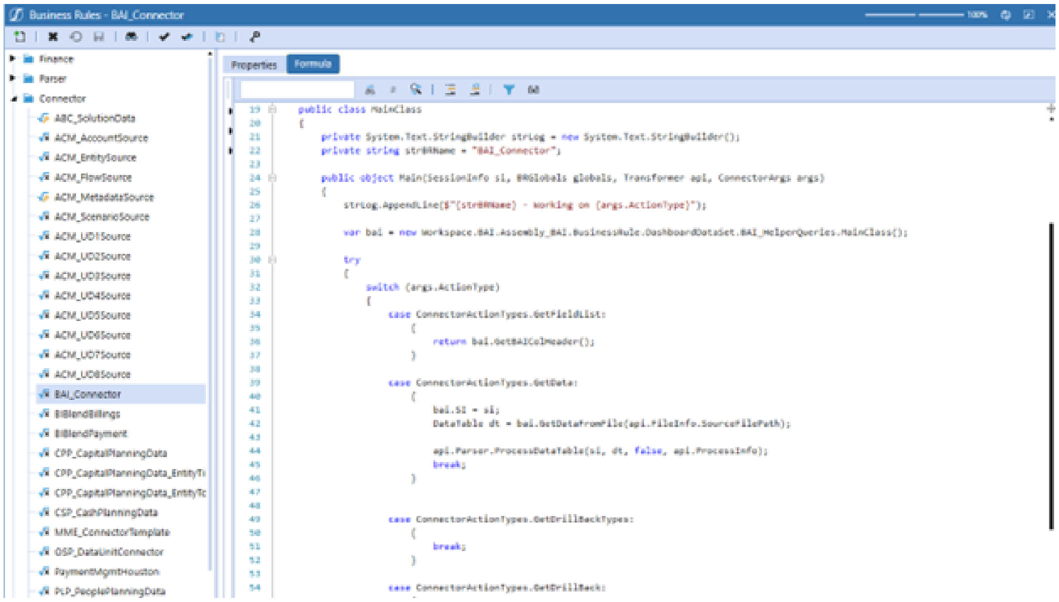
  To automate a file, follow the steps below:

  1. Navigate to the Data Source using the BAI Parser Solution.
  2. Set the **Connector Uses File** setting to **False**.



| | |
|---|---|
| **General** | |
| Name | BAIParser |
| Description | BAIParser |
| **Security** | |
| Access Group | Everyone |
| Maintenance Group | Everyone |
| **Settings** | |
| Cube Name | GolfStream |
| Scenario Type | Actual |
| **Data Structure Settings** | |
| Type | Connector |
| Data Structure Type | Tabular Data |
| Allow Dynamic Excel Loads | True |
| **Connector Settings** | |
| Connector Name | BAI_Connector |
| Connector Uses File | True |

  3. Create a copy of the BAI_Connector Connector Business Rule.
  4. Comment out line 42.



```
public class MainClass
{
    private System.Text.StringBuilder strLog = new System.Text.StringBuilder();
    private string strBRName = "BAI_Connector";

    public object Main(SessionInfo si, BRGlobals globals, Transformer api, ConnectorArgs args)
    {
        strLog.AppendLine($"{strBRName} - Working on {args.ActionType}");

        var bai = new Workspace.BAI.Assembly_BAI.BusinessRule.DashboardDataSet.BAI_HelperQueries.MainClass();

        try
        {
            switch (args.ActionType)
            {
                case ConnectorActionTypes.GetFieldList:
                    {
                        return bai.GetBAIColHeader();
                    }

                case ConnectorActionTypes.GetData:
                    {
                        bai.SI = si;
                        DataTable dt = bai.GetDataFromFile(api.FileInfo.SourceFilePath);

                        api.Parser.ProcessDataTable(si, dt, false, api.ProcessInfo);
                        break;
                    }


                case ConnectorActionTypes.GetDrillBackTypes:
                    {
                        break;
                    }

                case ConnectorActionTypes.GetDrillBack:
```

> **NOTE:** If you have an automated process that moves files to the Harvest Folder, you must add a step that also copies the file to a new folder in the OneStream File Explorer. This step must occur before the connector rule runs, so it is recommended to place this code in an extensibility rule.

5. Add code to the new connector rule that reads the file from your chosen OneStreamFile Explorer folder.

> **NOTE:** Below is an example of how you can access the folder structure under a specific workflow name:
>
> string IncomingFolder = BRApi.Utilities.GetFileShareFolder (si,FileShareFolderTypes.ApplicationIncoming, api.WorkflowProfile);

6. After defining the file path, use this code to process the file:

```
DataTable dt = bai.GetDataFromFile(SourceFilePath);
```

# Release PV800 SV101

The theme of this release is bug fixes.

> **NOTE:** Only the Release Notes were updated for this release. The BAI Parser Solution Guide has not been updated except to change the release number and copyright if needed.

## Prerequisites

- This solution is compatible with OneStream Platform v8.
- This release requires a minimum 8.0 Platform release.

## New Features

None

## Enhancements

None

## Fixed Issues

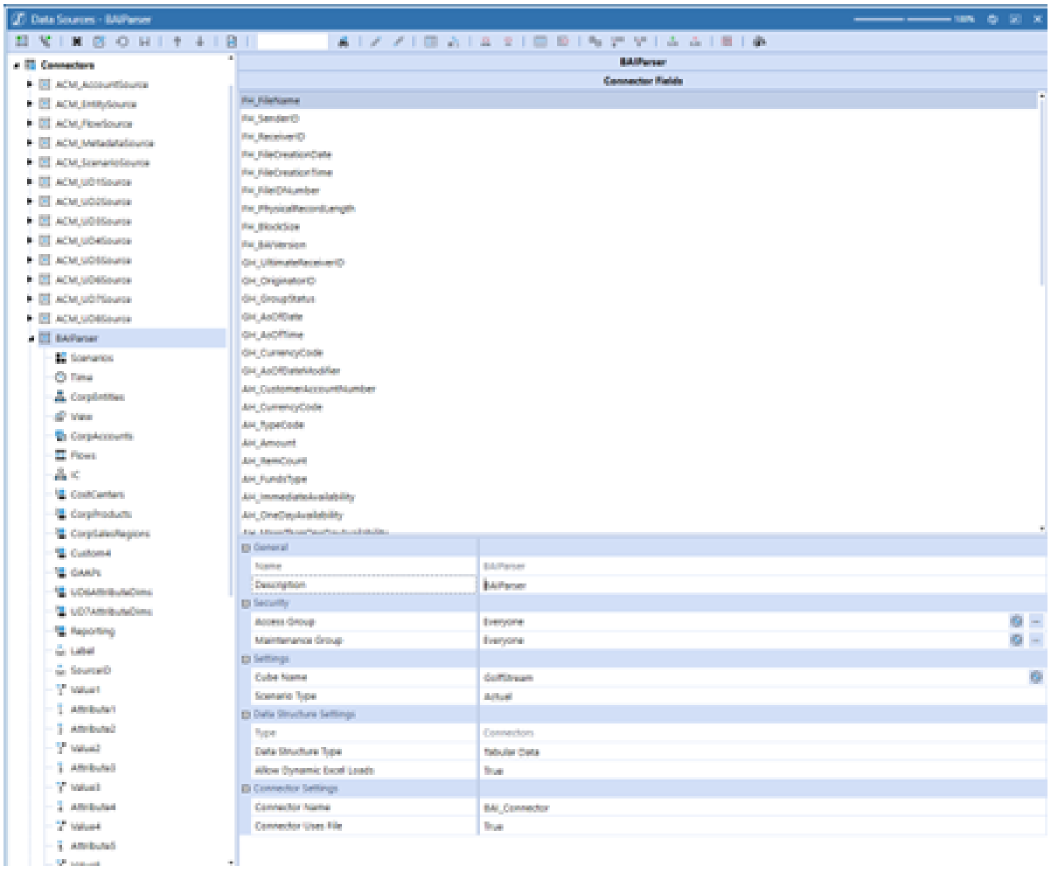
- BAI files uploaded without type codes parse based on all other codes available in the document. (MP3-4736)
- BAI files uploaded with empty, optional fields will parse without error. (MP3-4735)

# Known Issues

- Automation of the data load is impacted by the Connector Uses File setting.

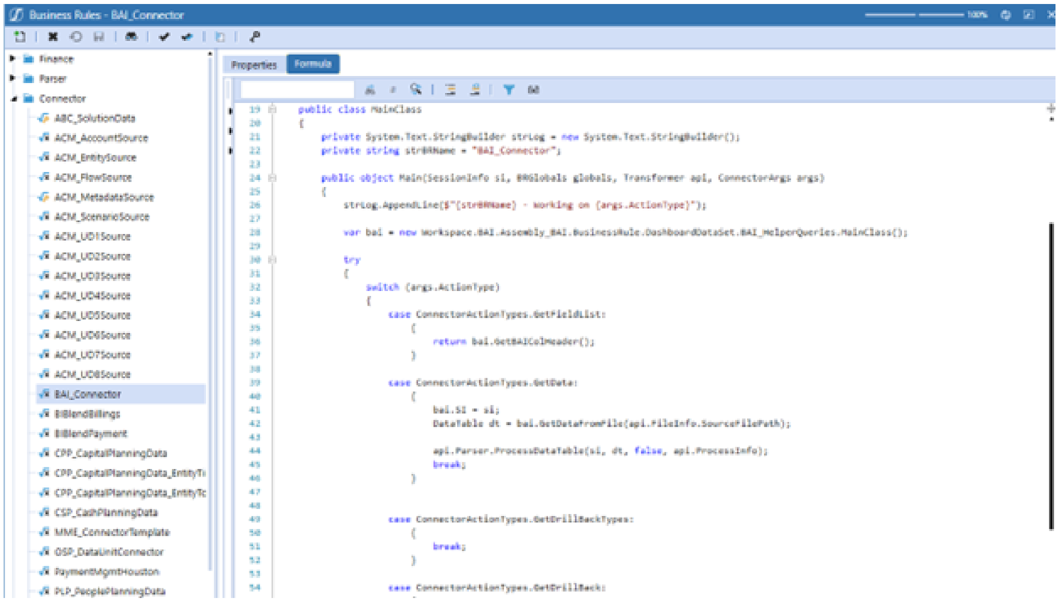
  To automate a file, follow the steps below:

  1. Navigate to the Data Source using the BAI Parser Solution.
  2. Set the **Connector Uses File** setting to **False**.



| | |
|---|---|
| General | |
| Name | BAIParser |
| Description | BAIParser |
| Security | |
| Access Group | Everyone |
| Maintenance Group | Everyone |
| Settings | |
| Cube Name | GolfStream |
| Scenario Type | Actual |
| Data Structure Settings | |
| Type | Connector |
| Data Structure Type | Tabular Data |
| Allow Dynamic Excel Loads | True |
| Connector Settings | |
| Connector Name | BAI_Connector |
| Connector Uses File | True |

  3. Create a copy of the BAI_Connector Connector Business Rule.
  4. Comment out line 42.



```
public class MainClass
{
    private System.Text.StringBuilder strLog = new System.Text.StringBuilder();
    private string strBRName = "BAI_Connector";

    public object Main(SessionInfo si, BRGlobals globals, Transformer api, ConnectorArgs args)
    {
        strLog.AppendLine($"{strBRName} - Working on {args.ActionType}");

        var bai = new Workspace.BAI.Assembly_BAI.BusinessRule.DashboardDataSet.BAI_HelperQueries.MainClass();

        try
        {
            switch (args.ActionType)
            {
                case ConnectorActionTypes.GetFieldList:
                    {
                        return bai.GetBAIColHeader();
                    }

                case ConnectorActionTypes.GetData:
                    {
                        bai.SI = si;
                        DataTable dt = bai.GetDataFromFile(api.FileInfo.SourceFilePath);

                        api.Parser.ProcessDataTable(si, dt, false, api.ProcessInfo);
                        break;
                    }


                case ConnectorActionTypes.GetDrillBackTypes:
                    {
                        break;
                    }

                case ConnectorActionTypes.GetDrillBack:
```

> **NOTE:** If you have an automated process that moves files to the Harvest Folder, you must add a step that also copies the file to a new folder in the OneStream File Explorer. This step must occur before the connector rule runs, so it is recommended to place this code in an extensibility rule.

5. Add code to the new connector rule that reads the file from your chosen OneStreamFile Explorer folder.

> **NOTE:** Below is an example of how you can access the folder structure under a specific workflow name:
>
> string IncomingFolder = BRApi.Utilities.GetFileShareFolder (si,FileShareFolderTypes.ApplicationIncoming, api.WorkflowProfile);

6. After defining the file path, use this code to process the file:

```
DataTable dt = bai.GetDataFromFile(SourceFilePath);
```

# Release PV800 SV100

This is the initial release of the BAI Parser Solution and documentation for installation guidance.

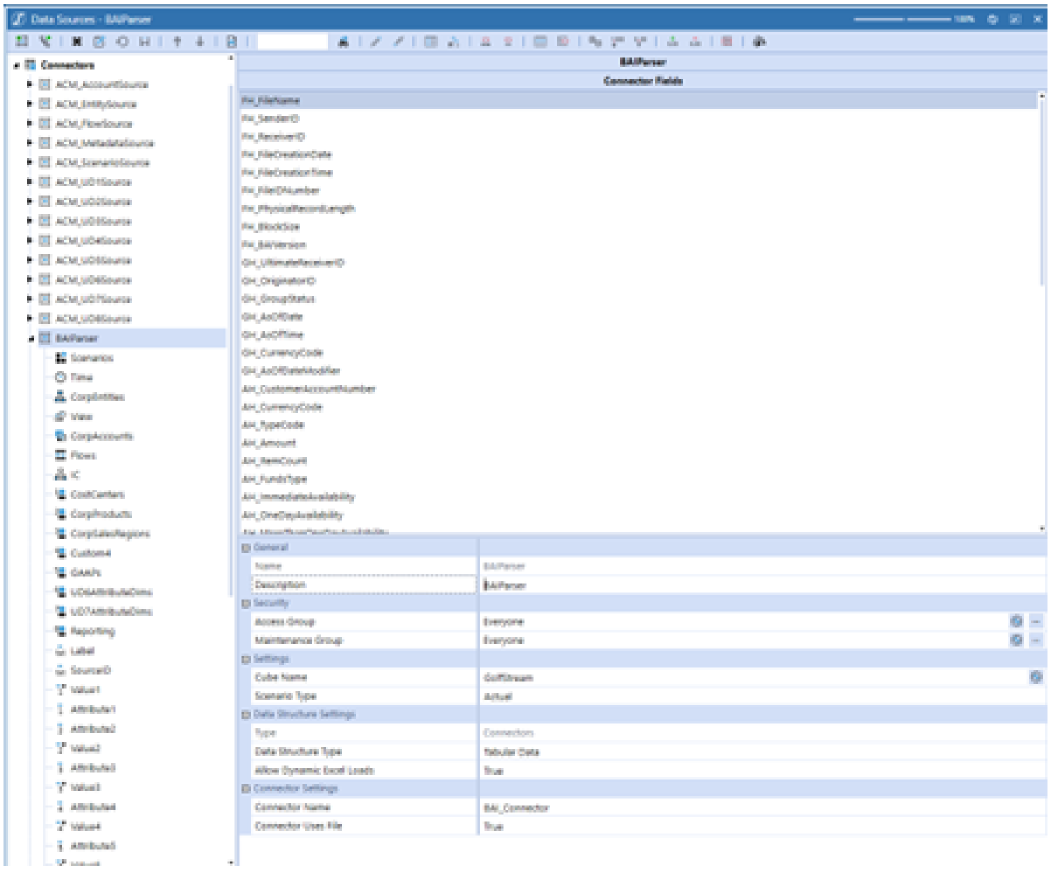
## Prerequisites

- This solution is compatible with OneStream Platform v8.
- This release requires a minimum 8.0 Platform release.

## Known Issues

- Automation of the data load is impacted by the Connector Uses File setting. To automate a file, follow the steps below:

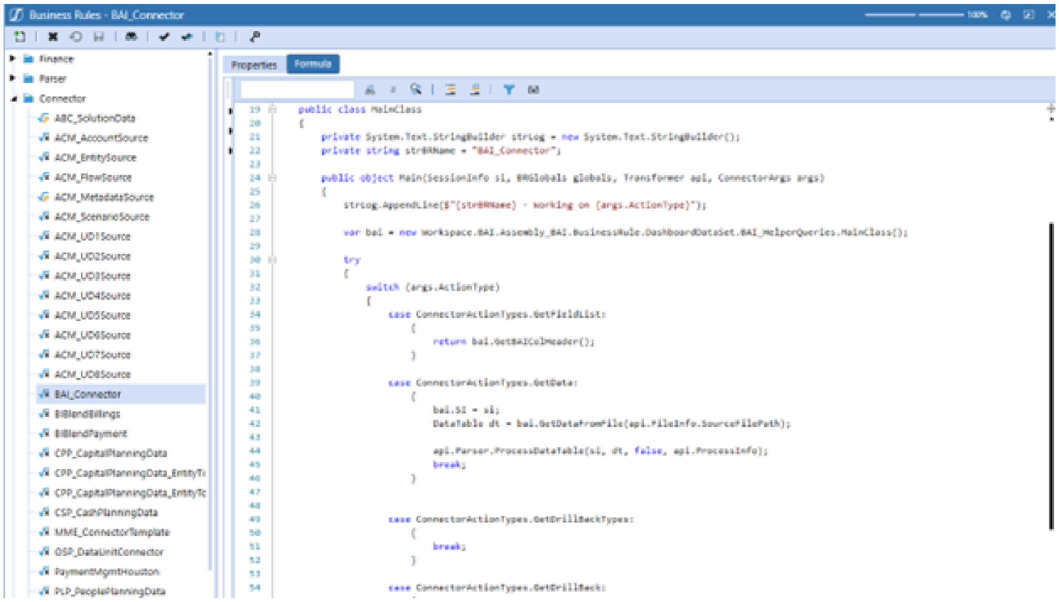
  1. Navigate to the Data Source using the BAI Parser Solution.
  2. Set the **Connector Uses File** setting to **False**.



| | |
|---|---|
| **General** | |
| Name | BAIParser |
| Description | BAIParser |
| **Security** | |
| Access Group | Everyone |
| Maintenance Group | Everyone |
| **Settings** | |
| Cube Name | GolfStream |
| Scenario Type | Actual |
| **Data Structure Settings** | |
| Type | Connector |
| Data Structure Type | Tabular Data |
| Allow Dynamic Excel Loads | True |
| **Connector Settings** | |
| Connector Name | BAI_Connector |
| Connector Uses File | True |

3. Create a copy of the BAI_Connector Connector Business Rule.

4. Comment out line 42.



```
public class MainClass
{
    private System.Text.StringBuilder strLog = new System.Text.StringBuilder();
    private string strBRName = "BAI_Connector";

    public object Main(SessionInfo si, BRGlobals globals, Transformer api, ConnectorArgs args)
    {
        strLog.AppendLine($"{strBRName} - Working on {args.ActionType}");

        var bai = new Workspace.BAI.Assembly_BAI.BusinessRule.DashboardDataSet.BAI_HelperQueries.MainClass();

        try
        {
            switch (args.ActionType)
            {
                case ConnectorActionTypes.GetFieldList:
                    {
                        return bai.GetBAIColHeader();
                    }

                case ConnectorActionTypes.GetData:
                    {
                        bai.SI = si;
                        DataTable dt = bai.GetDataFromFile(api.FileInfo.SourceFilePath);

                        api.Parser.ProcessDataTable(si, dt, false, api.ProcessInfo);
                        break;
                    }


                case ConnectorActionTypes.GetDrillBackTypes:
                    {
                        break;
                    }

                case ConnectorActionTypes.GetDrillBack:
```

> **NOTE:** If you have an automated process that moves files to the Harvest Folder, you must add a step that also copies the file to a new folder in the OneStream File Explorer. This step must occur before the connector rule runs, so the code needs to be placed in an extensibility rule.

5. Add code to the new connector rule that reads the file from your chosen OneStreamFile Explorer folder.

6. After defining the file path, use this code to process the file:

```
DataTable dt = bai.GetDataFromFile(SourceFilePath);
```

> **NOTE:** This is an example of how to access the folder structure under a specific workflow name:
>
> string IncomingFolder = BRApi.Utilities.GetFileShareFolder

```
(si,FileShareFolderTypes.ApplicationIncoming,
api.WorkflowProfile);
```