

# Appian Step-by-Step #12

Exercise to Accompany  
Sites: Create a Custom and Focused User Experience

The Appian Step-by-Step series consists of 12 exercises that accompany the courses in the Appian Developer learning path. Exercises build upon each other. Complete exercises in order and keep the app and all objects until you are done with the project.

- 1** Welcome to the Appian Developer Learning Path
- 2** Create an Application
- 3** Manage Users and Groups
- 4** Expressions
- 5** Records Part 1: Accessing Your Data
- 6** Records Part 2: Record Type Relationships
- 7** Query Entities
- 8** Process Modeling 101: Part 1
- 9** Process Modeling 101: Part 2
- 10** Reports
- 11** Sites
- 12** **Task Report**

<b>Exercise 12: Task Report</b>	<b>3</b>
Create a Process Report	3
Modify the Process Report	4
Create a Constant for the Process Report	4
Create an Interface for the Task Report	5
Create Links to Process Tasks	7
Create the Status Filter	8
Create the Task Report	11
Add the Task Report Page to the AX Site	11
Export the Application	12
Review Security Summary	12
Check for Missing Precedents	12
Export the Application	13

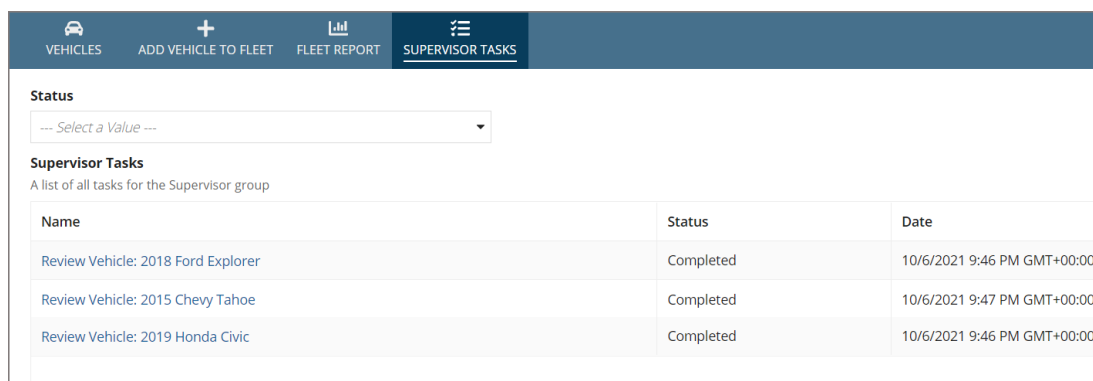
## Notice of Rights

This document was created by Appian Corporation, 7950 Jones Branch Dr, Tysons, Virginia 22102. Copyright 2021 by Appian Corporation. All rights reserved. Information in this document is subject to change. No part of this document may be reproduced or transmitted in any form by any means, without prior written permission of Appian Corporation. For more information on obtaining permission for reprints or excerpts, contact Appian Training at [academyonline@appian.com](mailto:academyonline@appian.com).

Appian Step-by-Step 21.4

## Exercise 12: Task Report

In this exercise, you'll learn how to create a Supervisor Task Report that will allow Supervisors to view and perform their vehicle approval tasks. Once your task report is assembled, you'll add it as a page to the AX site. Supervisors will use this page to view tasks, filter them by status, and navigate to the UIs holding specific tasks by clicking the hyperlinked task name. Your final site page will look as follows:



The screenshot shows a web interface for 'Supervisor Tasks'. At the top is a navigation bar with four tabs: 'VEHICLES', 'ADD VEHICLE TO FLEET', 'FLEET REPORT', and 'SUPERVISOR TASKS' (which is active). Below the navigation bar, there is a 'Status' filter section with a dropdown menu showing '--- Select a Value ---'. Underneath is the 'Supervisor Tasks' section, which includes a subtitle 'A list of all tasks for the Supervisor group' and a table. The table has three columns: 'Name', 'Status', and 'Date'. It contains three rows of task data, each with a hyperlinked name, a status of 'Completed', and a timestamp.

Name	Status	Date
<a href="#">Review Vehicle: 2018 Ford Explorer</a>	Completed	10/6/2021 9:46 PM GMT+00:00
<a href="#">Review Vehicle: 2015 Chevy Tahoe</a>	Completed	10/6/2021 9:47 PM GMT+00:00
<a href="#">Review Vehicle: 2019 Honda Civic</a>	Completed	10/6/2021 9:46 PM GMT+00:00

To build a task report, you'll start by building a process report that will help you to pull data from the AX New Vehicle process model. At a high-level, you'll complete these steps to create the task report for Supervisors:

- Create a process report
- Create a constant to point to the process report
- Build an interface to display tasks, link tasks to individual approval forms, and add a Status filter to this interface
- Create a task report
- Add the task process report as a page to the Acme Exercise Site

Follow the steps below to build a task report.

### Create a Process Report

Let's create the task process report to show the tasks that are assigned to the Supervisor group:

1. From within the **AX Document** folder, click **New**, and select **Process Report**.
2. Retain the **Duplicate existing process report** selection, and search and select **My Tasks** in the **Process Report to Duplicate** box. If no suggestions appear for **My Tasks**, use **active\_tasks** instead.
3. Name it **AX Supervisor Process Report**, and then change the description to "A list of tasks for the AX Supervisors group."

4. Make sure the **Save In** folder is set to **AX Documents**, and click **Create**.

## Modify the Process Report

In the previous step, you created the AX Supervisor Process Report by duplicating the out-of-the-box My Task report, which by default displays all tasks for the logged in user. In this section, you'll modify this process report to display tasks assigned to the AX Supervisor group:

1. Locate and open the **AX Supervisor Process Report**.
2. On the top menu bar, select **All** next to **Status**, and then click **Edit**.
3. Navigate to the **Filters** tab, and click **New Rule**. In the first dropdown menu, select **Assigned To**, and then type **AX Supervisors** in the text entry field.

**Report Options**

General Data **Filters** Indicators Display

**Default Filters**

+ New Rule

Only include data that meets all of the selected criteria:


- ☐ Active
- ☐ Completed
- ☐ One of the last 0 completed
- ☐ Started within the last 0 days
- ☐ Completion time is outside 0 standard deviations
- ☐ Ahead of schedule
- ☐ Behind schedule
- ☐ Due today
- ☒ Assigned To = AX Supervisors

**Quick Filters** ☒ Show as Toolbar

+ Add Filter Set + Move Up + Move Down X Delete Display filters as ☒ Links ☐ Dropdowns

Filter Set	Delete
<input type="checkbox"/> Status:	X
<input type="checkbox"/> Priority:	X
<input type="checkbox"/> Due:	X

SAVE CANCEL

4. Click **Save** to close the dialog box, and then click the save icon  on the report menu bar to save the process report.

## Create a Constant for the Process Report

Before you create the task report interface, you need to create a constant for the AX Tasks Process Report. This constant will be used in the interface that you will create in the next step.

1. From within the **AX Constants** folder, click **New**, and select **Constant** from the dropdown menu.
2. Type **AX\_SUPERVISOR\_TASK\_PROCESS\_REPORT\_POINTER** for the **Name**, and add a description: "Points to the AX Supervisor Process Report object."
3. In the **Type** field, select **Document**.
4. In the **Value** field, type and select **AX Supervisor Process Report**. Click **Create**.

**Create Constant**

☒ Create from scratch ☐ Duplicate existing constant

**Name \***

AX\_SUPERVISOR\_TASK\_PROCESS\_REPORT\_POINTER

**Description**

Points to the AX Supervisor Process Report object.

**Type \***

Document

☐ Array (multiple values)

**Value**

AX Supervisor Process Report

**Environment Specific**

☐ Different environments need to have different values for this constant

**Save In \***

AX Constants

**CANCEL** **CREATE**

## Create an Interface for the Task Report

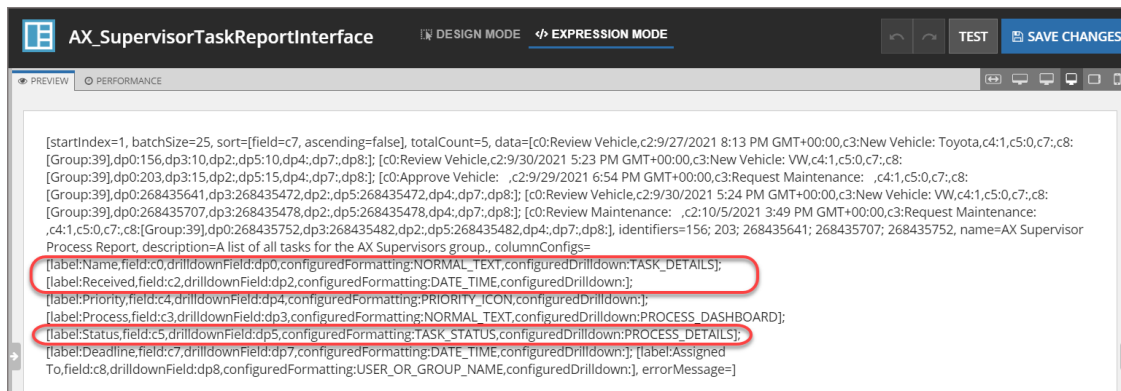
In this section, you'll create an interface to display all tasks in a grid. Later, you'll use this interface to create a task report.

1. From within the **AX Interfaces** folder, click **New**, and select **Interface** from the dropdown menu. Name it **AX\_SupervisorTaskReportInterface**, and add a brief description "Interface to display tasks to Supervisors." Make sure the **Save In** folder is set to **AX Interfaces**, and click **Create**.

2. First, you will use the `a!queryProcessAnalytics()` function to populate the empty interface with data from the active tasks process report. You will do this in order to figure out what columns you want to display to Supervisors. In the interface, click **Expression Mode**, and enter the following expression into the **Interface Definition**:

```
a!textField(readonly: true, value: a!queryProcessAnalytics(
  report: cons!AX_SUPERVISOR_TASK_PROCESS_REPORT_POINTER ))
```

3. Scan the data in the **Preview** tab. You'll notice that it is the text representation of the process report data.



The columns you want to display in the grid are:

- C0 - Task name
- C2 - Date Time
- C5 - Task Status

4. Now that you know which columns you want to display, you are ready to set up your read-only grid. Delete the code in the **Interface Definition**, and then replace it with the following expression:

```
a!sectionLayout(  
  contents: { a!gridField(  
    label: "Supervisor Tasks",  
    instructions: "A list of all tasks for the Supervisor  
group",  
    labelPosition: "ABOVE",  
    data: a!queryProcessAnalytics(  
      report: cons!AX_SUPERVISOR_TASK_PROCESS_REPORT_POINTER,  
      query: a!query(pagingInfo: fv!pagingInfo)),  
    columns: {  
      a!gridColumn(  
        label: "Name",  
        sortField: "c0",  
        value: fv!row.c0 ),  
      a!gridColumn(  
        label: "Status",  
        sortField: "c5",  
        value: fv!row.c5),  
      a!gridColumn(  
        label: "Date",  
        sortField: "c2",  
        value: fv!row.c2 ),},  
    rowHeader: 1)}})
```

5. Switch back to **Design Mode** and click **Save Changes**.

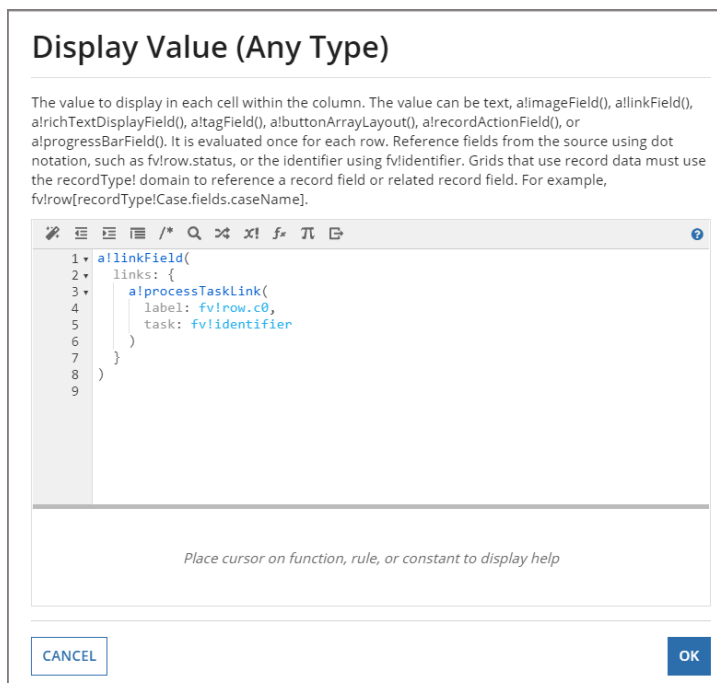
## Create Links to Process Tasks

The grid you just created displays the task name but users do not have a way to open the task. In this section, you'll link the Name column to the actual task.

1. Select the **Read-only Grid** component, then under **Columns** in the **Component Configuration** panel, click **Name (Grid Column)**.
2. Click **Display Options**, then select **Process Task Link**.

3. Click the **Expression Editor** icon next to **Display Value**, and replace the existing expression with the following expression:

```
a!linkField(  
  links: {  
    a!processTaskLink(  
      label: fv!row.c0,  
      task: fv!identifier  
    )  
  }  
)
```



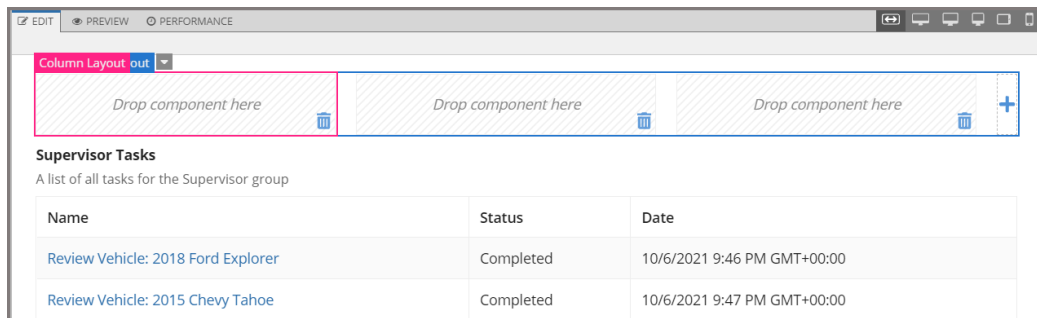
4. Click **OK**, then **Save Changes**.

## Create the Status Filter

In this exercise, you'll format the status column to show meaningful values instead of numbers, and you'll add a filter to the interface to allow Supervisors to filter tasks by the most common statuses.



1. From the **Components** palette, grab a **Columns** layout, and drop it above the read-only grid.

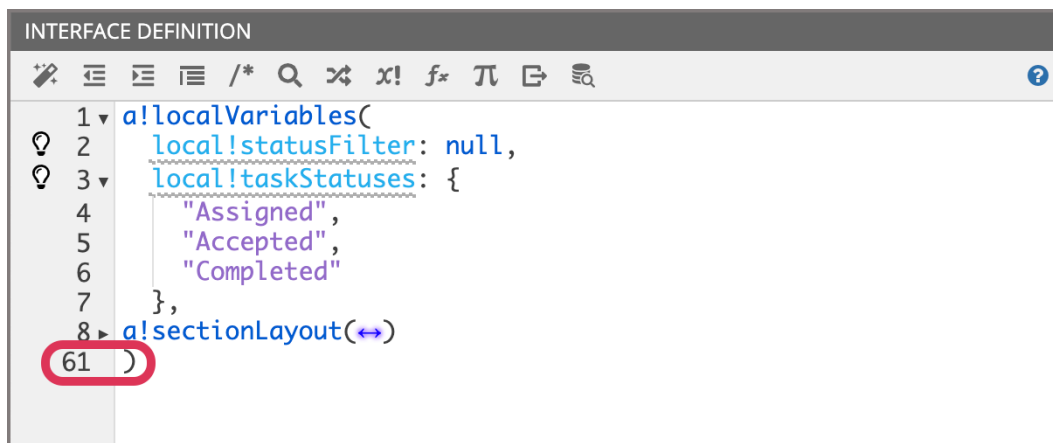


2. Next, let's add a local variable to this interface to hold the task status values that you will use in the dropdown component. This component will allow Supervisors to filter tasks by status. Click **Expression Mode**.
3. Click the arrow next to line 1 to collapse the existing expression, and press **Enter**. In line 1, type the following expression:

```
a!localVariables(
  local!statusFilter: null,
  local!taskStatuses: {
    "Assigned",
    "Accepted",
    "Completed"
  },

```

4. Scroll to the last line of code in the interface definition pane and add a right parenthesis `)`. Now, our entire expression is encapsulated within the `a!localVariables` function. This is required so that local variables declared in the beginning can be referenced by other encapsulated code further down:



5. Return to the **Design Mode**, and click **Save Changes**.
6. Next, let's drag and drop a **Dropdown** field from the **Components** palette into the first column layout on the left.
7. In the **Component Configuration** pane, configure the dropdown field as follows:
  - Type **Status** in the Label field.
  - Click the **Expression Editor** icon next to **Choice Labels**, and type the following expression: **local!taskStatuses**. This populates the Choice Labels with the list of statuses defined in Step 4.
  - Click the **Expression Editor** icon next to **Choice Values**, and type the following expression: **enumerate(3)**.
  - In the **Selected Value** and **Save Selection To** fields, use the dropdown menu to select **local!statusFilter**.
8. Next, let's add a filter to the read-only grid, so that once the Supervisor selects a value in the Status dropdown, the grid refreshes to display only the selected statuses. Select the **Read-only Grid** component, and in the **Component Configuration** pane click the **queryProcessAnalytics** link (located in the **Data Source** section).
9. Click the **query** link in **ContextGroups**, and then click the **Edit** link below **filter**. Enter the following expression into the filter (QueryFilter) dialog box:

```
a!queryFilter(
  field: "c5",
  operator: "=",
  value: local!statusFilter,
  applyWhen: not(isnull(local!statusFilter))
)
```

Click **OK**. Your filter should now work. Test the filter by selecting different values in the dropdown status menu.

10. Select the **Read-only Grid** component to replace the numbers in the Status column with labels like Assigned and Completed. In the **Component Configuration** pane, click **Status (Grid Column)** under **Columns**.
11. Click the **Expression Editor** icon next to **Display Value**, and replace the existing expression with the following expression:

```
a!forEach(
  items: fv!row.c5,
  expression: if(
    not(isnull(fv!row.c5)),
```

```
fn!index(
    local!taskStatuses,tointeger(fv!item)+1,"Other"),
    toString(fv!item))
```

12. Click **OK**, then **Save Changes** to save the interface.

## Create the Task Report

Now that you have the `AX_SupervisorTaskReportInterface`, let's create a task report object. Once you have a task report, you'll be able to add it as a page to the Acme Exercise Site.

1. In your application, click **New**, and select **Report**. Name this report **AX Supervisor Task Report**, and add a description: "Task report object to show Supervisor tasks."
2. Type and select **AX\_SupervisorTaskReportInterface** in the **Interface** field, and select the **Save as Task Report** checkbox. Click **Create**.

3. To set security for this report, click **Add Users and Groups**, and add **AX All Users** as **Viewers**, and **AX Administrators** as **Administrators**. Click **Save**.

## Add the Task Report Page to the AX Site

Next, add the AX Supervisor Task Report to the Acme Exercise Site.

1. Open the **Acme Exercise Site**, and click **Add Page**. Configure this page as follows:
  - Title the page **Supervisor Tasks**.
  - Click the dropdown menu for the icons, and then browse or search for an appropriate icon.
  - Select **Report** in the **Type** field.

- Select **AX Supervisor Task Report** in the **Content** field.
- Select **Only show when** under **Visibility**, and enter the expression:

```
a!isUserMemberOfGroup(loggedInUser() ,
cons!AX_SUPERVISORS_POINTER)
```

- Click **OK**.
2. Click **Save Changes**. To preview the new Supervisor Tasks page, click the **URL** for the site, and select the new Supervisors Tasks tab. Now, the Supervisors are able to see their tasks in a single place, filter them by status, and approve or deny the addition of vehicles to the fleet.


## Export the Application

The import and export of applications is disabled for Appian Community Edition. However, for applications outside of Appian Community Edition, you will want to export your finished application as a zip file. Once you export the app, you can store the file, or import it into a different environment. In order to export the app, you will prepare for the export by:

- Reviewing your app's security settings
- Checking for missing precedents, or objects


These two quick checks will ensure that your app is export-ready and will work as expected in its new environment. When you export future applications, follow the steps below.

### Review Security Summary

1. Check the security summary of the app to ensure that every object has been secured correctly. Click the **Settings** menu  at the top right of the app, and select **Security Summary** from the dropdown menu.
2. All objects should have a security setting, and only groups and not individual users should be used for security. Review your **Security Summary**, making sure that there are no unsecured objects, and all objects are secured at least through the **AX Administrators** group. Review that the **AX All Users** group is assigned the **Viewer** rights.


### Check for Missing Precedents

A missing precedent refers to an object referenced in your application that belongs to a different app. Usually, this happens when you reuse a shared object. To prevent problems, always check for missing precedents, and add the shared objects to your app before exporting it. Follow the steps below to complete this important check.

1. Click the **Settings** menu  at the top right of the app, and select **Missing Precedents** from the dropdown menu.
2. If there are any missing precedents in your app, review them to verify whether you need to add the objects to your app, or the dependency should be removed. For example, sometimes developers may include an erroneous object into their app by clicking the incorrect auto-suggestion. There are other reasons why you may get a missing precedent message:
  - Many teams create a dedicated application for deploying shared components. This is done to avoid clutter. If this is your scenario, you may need to ignore the missing precedent message. The app with shared components is usually deployed separately, or it might be already present in your target environment.
  - Missing precedents can also occur if you erroneously create an object from within a different app. In this instance, add the object to your app.
3. If you find a precedent that should be added to your app, select the precedent and click **Add to App**. Alternatively, you can add multiple precedents by clicking **Add all to application**. Click **Check Again** to run another missing precedents check that no further dependencies were discovered.

## Export the Application

Once you complete the above checks, export your app.

1. Click the **Settings** menu  at the top right of the app, and select **Export Application** from the dropdown menu. Verify the name of your app, and update it to include the date of the export.
2. Click **Export**, and then click **Download Package**.
3. If you receive an error message, open the export log and correct any problems. Download the newly created **zip file**, so you can take your work with you to a different environment.