

## Table of Contents

<b>PREREQUISITES</b> .....	<b>2</b>
<b>REFERENCE INFORMATION</b> .....	<b>3</b>
<b>COMPONENT REFERENCE</b> .....	<b>3</b>
<b>SERVICEABILITY</b> .....	<b>3</b>
<b>COMMAND VARIABLE LEGEND THE VARIABLES IN THE FOLLOWING TABLE ARE USED IN EXAMPLES THROUGHOUT THIS DOCUMENT:</b> .....	<b>3</b>
<b>CONFIGURATION AND CUSTOMIZATION SUMMARY</b> .....	<b>3</b>
<b>DEPLOYING THE MAF CONFIGURATION APPLICATION</b> .....	<b>5</b>
<b>RECOMMENDED ENVIRONMENT CONFIGURATION</b> .....	<b>5</b>
<b>DOWNLOADING THE DOCKER IMAGE THAT IS STORED IN THE ENTITLED REGISTRY</b> .....	<b>5</b>
<b>RUNNING THE DOCKER IMAGE</b> .....	<b>6</b>
<b>USING THE MAF CONFIGURATION APPLICATION</b> .....	<b>6</b>
<b>PREREQUISITES</b> .....	<b>6</b>
<b>IBM MAXIMO APPLICATION SERVER APPLICATION AUTHORIZATION SETTINGS</b> .....	<b>6</b>
<b>OPENING THE APPLICATION</b> .....	<b>7</b>
<b>LOG IN</b> .....	<b>7</b>
<b>MODIFYING APPLICATIONS</b> .....	<b>7</b>
<b>DUPLICATING APPLICATIONS</b> .....	<b>10</b>
<b>REDOWNLOADING APPLICATIONS</b> .....	<b>12</b>
<b>DELETING APPLICATIONS</b> .....	<b>12</b>
<b>CONFIGURING APPLICATIONS</b> .....	<b>12</b>
<b>ANATOMY OF AN APPLICATION XML FILE</b> .....	<b>13</b>
<b>REFERENCING OTHER APPLICATION XML FILES</b> .....	<b>16</b>
<b>UPGRADING CONFIGURATION CHANGES TO A NEW VERSION OF THE APPLICATION</b> .....	<b>16</b>
<b>COMMAND VARIABLE LEGEND THE VARIABLES IN THE FOLLOWING TABLE ARE USED IN EXAMPLES THROUGHOUT THIS DOCUMENT:</b> .....	<b>16</b>
<b>DOWNLOAD AND EXTRACT THE SOURCE FILES FOR THE APPLICATION</b> .....	<b>17</b>
<b>SAVE THE CONFIGURATION UPDATES YOU MADE FOR A CURRENT APPLICATION</b> .....	<b>17</b>
<b>UPGRADING THE MAXIMO APPLICATION SERVER</b> .....	<b>17</b>
<b>REAPPLYING YOUR CHANGES TO THE NEW VERSIONS OF THE APPLICATION XML FILE</b> .....	<b>18</b>
<b>PUBLISHING THE UPDATED APPLICATION TO THE MAXIMO APPLICATION SERVER</b> .....	<b>18</b>
<b>FILTERING RECORD RETURNED FROM A QUERY</b> .....	<b>18</b>
<b>DATA SOURCES</b> .....	<b>19</b>
<b>XML CONFIGURATION EXAMPLES</b> .....	<b>21</b>

CHANGING A LABEL AND ADDING A FIELD.....	21
<b>SIGNATURE OPTIONS.....</b>	<b>24</b>
<b>CONFIGURING QUERIES.....</b>	<b>24</b>
CONFIGURING QUERIES ON THE MAXIMO APPLICATION SERVER .....	24
CONFIGURING QUERIES IN THE APP.XML FILE OF AN APPLICATION .....	26
FOR EXAMPLE, THE ASSIGNEDWOLIST QUERY FROM THE MXAPIWODETAIL OBJECT STRUCTURE RETURNS WORK ORDER ASSIGNMENT DATA TO AN APPLICATION.....	26
<b>CONFIGURING IBM MAXIMO HEALTH AND PREDICT – UTILITIES.....</b>	<b>27</b>
MAXIMO HEALTH AND PREDICT – UTILITIES SIGNATURE OPTIONS.....	27
MAXIMO HEALTH, MAXIMO PREDICT, AND MAXIMO HEALTH AND PREDICT - UTILITIES XML CONFIGURATION.....	28
<b>CUSTOMIZING APPLICATIONS .....</b>	<b>36</b>
<b>LOCATING THE REPORT WORK PAGE.....</b>	<b>38</b>
<b>CREATING A DATASOURCE .....</b>	<b>38</b>
<b>CREATING A LOOKUP.....</b>	<b>39</b>
<b>SETTING THE DISPLAY OF THE FIELD .....</b>	<b>40</b>
<BORDER-LAYOUT FILL-PARENT="TRUE" PADDING="TRUE" MIDDLE-BORDER="TRUE" ID="">.....	41
<b>SAVING XML CONFIGURATION CHANGES .....</b>	<b>41</b>
<b>CREATING CUSTOM CODE .....</b>	<b>42</b>

# Configuring and customizing applications with the Maximo Application Framework (MAF) Configuration application

You can use the MAF Configuration application to update applications that are built using the Maximo Application Framework. The MAF Configuration application supports IBM Maximo Mobile, IBM Maximo Health, and IBM Maximo Manage.

The MAF Configuration application works through Docker container technology and Maximo Application Framework tools. Maximo Application Framework tools are used to create a configuration environment within a Docker image on your local system. Download, modify application XML and JavaScript, preview, and publish your configured Maximo applications with minimal setup.

## Prerequisites

- A minimum of 8 GB RAM on the development system (16 GB recommended).

- Docker Desktop installed on your development system. (<https://docs.docker.com/get-docker/>). Ensure that your system meets all of the Docker Desktop requirements. Set the Memory Resources preference in Docker Desktop to 8 GB of memory.
- A working directory on your local system.
- If you are using Windows, you must have the Windows Subsystem for Linux (WSL) version 2 installed on your system.

## Reference information

### Component reference

An IBM Maximo component reference is available from the side panel of the application. This information describes the properties and XML code of Maximo components that are included in Maximo for EAM applications. Select the Component documentation icon to access the

### Serviceability

The Maximo Application Framework Configuration Application docker image includes network tools that are commonly used to troubleshoot deployment issues. Tools such as ping, tracepath, nano, ifconfig, netstat, traceroute, and dig can be used to collect information about your deployment when you are engaged with IBM Support.

reference information.

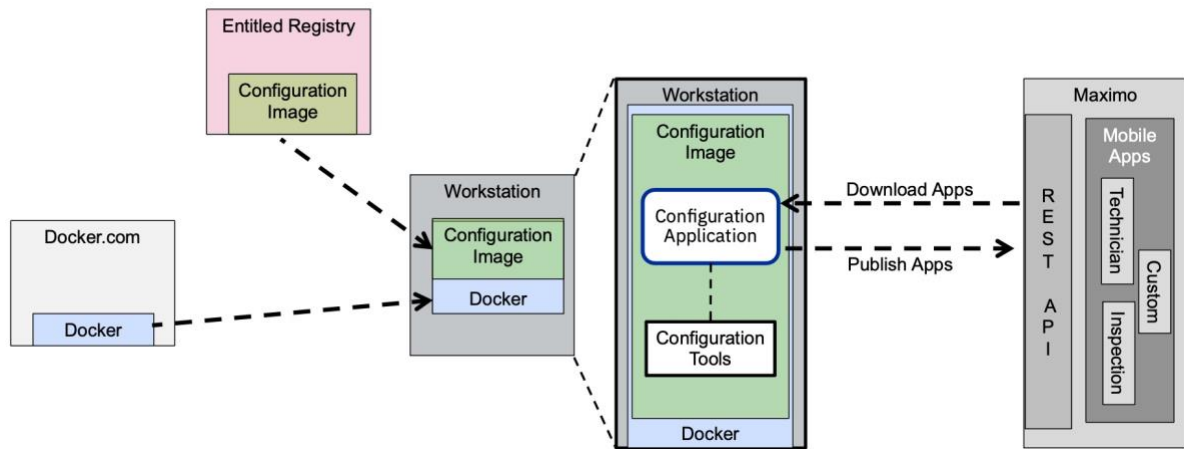
### Command variable legend

The variables in the following table are used in examples throughout this document:

Variable	Description
<code>\${maximo_workspace_directory}</code>	The working directory that you create on your local system.
<code>\${app_directory}</code>	The directory on your local system that contains application code when you download an application for configuring.
<code>\${maximo_app_name}</code>	The name of the Maximo application that you are configuring. For example, Inspections or Technician.

## Configuration and customization summary

## Deployment - (Maximo 7.6.1.2 & MAS 8.x)



Configuring and customizing a Maximo for EAM application consists of the following steps:

1. Install Docker Desktop on your local development system.
2. Download the MAF Configuration application image from the entitlement registry.

This process can take several minutes. While you wait, you can configure the [IBM Maximo application server application authorization settings](#).

3. Start the MAF Configuration application Docker container.
4. Maximo server config (api key, etc)
5. Log in to the application.
6. Select an application to modify.
7. Modify the application.

Configuration refers to updates to the display data contained in application XML code. Customization refers to the creation of JavaScript code to support application functionality. For example, you might configure an application by adding a new field to an application page. If the field required JavaScript to retrieve data, you would customize the application by creating JavaScript to support that field.

8. Save your changes.
9. View the updated application locally using the Preview function.

Previewing an application rebuilds the application and starts a React server, which can take an extended amount of time.

10. Publish the application to the Maximo application server.
11. **IMPORTANT:** Return to the Application list page. Select the Edit link for the application that you published. From the Configure application window, set the Mobile option to **Yes**.

## Deploying the MAF Configuration application

### Recommended environment configuration

For best results, there are two recommended ways to structure your configuration environment.

1. Host your entire environment in a single VM, including the Maximo application server, The MAF Configuration application, and your browser.
2. A Maximo application server deployed in a VM and hosting the MAF Configuration application and browser on your local system. Ensure that you can access the Maximo application server VM from your local system.

Hosting the MAF Configuration application and browser in one VM and deploying the Maximo application server in a separate VM is not recommended.

If the MAF Configuration application Docker image is hosted in a VM and you cannot access the Maximo application server during the configuration login, but you can access the Maximo application server from a browser running in the same VM, this identifies a Container network accessibility problem. IBM Support cannot assist with resolving internal connectivity issues within your network.

Additionally, you should maintain separate workspaces for each version of Maximo that you support. For example, you should establish separate workspaces for version 8.5 and 8.6.

### Downloading the Docker image that is stored in the entitled registry

The MAF Configuration application is provided in a Docker container image that is stored in the IBM® Entitled Registry. Before you can download the MAF Configuration application Docker container image from the IBM Entitled Registry, you must obtain an entitlement key.

1. Log in to [MyIBM Container Software Library](#) by using the IBM ID and password that is associated with the entitled software.
2. In the Entitlement keys section, click **Get entitlement key**.
3. Click **Copy key**.
4. Paste the key into an empty file so that it is available to you when you start to deploy the MAF Configuration application on Docker.
5. To log in to the IBM Entitled Registry, run the following command:

```
docker login cp.icr.io --username cp --password entitlement_key
```

Replace the *entitlement\_key* variable with the key you copied earlier.

6. To download the MAF Configuration application Docker container image, run the following command, replacing *x* with the latest image available:

```
docker pull cp.icr.io/cp/manage/maf-tools:8.7.x
```

## Running the Docker image

After the image download is complete, run the Docker image using the following command, replacing *x* with the latest image available:

```
docker run -it --privileged -p 3001:3001 -p 3006:3006 -v  
${maximo_workspace_directory}:/graphite/.workspace -it  
cp.icr.io/cp/manage/maf-tools:8.7.x
```

The `${maximo_workspace_directory}` variable refers to a working directory that you create on your local system. When you specify a port number, you must use a value between 3000 and 3050. If you are using a Windows system, you might have to enable file sharing for the directory you designate as your workspace.

If you run the Docker image from a Windows system, you must include the `-e` parameter to the command, replacing *x* with the latest image available.

```
docker run -it --privileged -p 3001:3001 -p 3006:3006 -v  
${maximo_workspace_directory}:/graphite/.workspace -it -e  
CHOKIDAR_USEPOLLING=true cp.icr.io/cp/manage/maf-tools:8.7.x
```

You can use the `GRAPHITE_RELEASE` parameter to reduce the size of the application package that you publish back to the Maximo application server. Use of this parameter may slow performance of the publish action of the MAF Configuration application. You can increase the memory assigned to the Docker container to improve performance.

```
docker run -it --privileged --env GRAPHITE_RELEASE=1 -p 3001:3001 -p  
3006:3006 -v ${maximo_workspace_directory}:/graphite/.workspace -it  
cp.icr.io/cp/manage/maf-tools:8.7.x
```

Maximo requires a valid security certificate to retrieve and publish applications from the MAF Configuration application in production environments. In development or demonstration environments, you can use the `NODE_TLS_REJECT_UNAUTHORIZED` parameter to bypass system checks for a valid security certificate.

```
docker run -it --privileged -env NODE_TLS_REJECT_UNAUTHORIZED=0 -p 3001:3001  
-p 3006:3006 -v ${maximo_workspace_directory}:/graphite/.workspace -it  
cp.icr.io/cp/manage/maf-tools:8.7.x
```

## Using the MAF Configuration application

### Prerequisites

Ensure API Key is enabled on the Maximo application server. Refer to [Interfacing with REST apis using API keys](#) and [API Keys](#) for more information.

### IBM Maximo application server application authorization settings

1. Open the object structures application, located within Maximo Integration Module, and select the **OSLCMAFAPPDATA** object structure.

2. Choose the **Configure Object Structure Security** action and then select **Use Object Structure for Authorization Name** if that option is not already selected.
3. Open the Security Group application and click the Object Structure tab.
4. Grant the group of the MAF Configuration application user access to the **OSLCMAFAPPDATA** and object structure if it is not already included.

Repeat this process for the **WPEDITSETTING** object structure.

If you add a new object structure in a Maximo Datasource to an existing or duplicated application, you need to grant access to that object structure.

### Opening the application

In a browser, open the following URL:

<http://localhost:3001>

### Log in

1. Enter the Maximo application server URL.
2. Enter your username and password.

Alternatively, you can authenticate to the application server using an API key value.

1. Select Log in using an API Key.
2. Enter the Maximo application server URL.
3. Enter the API Key value.

Specify the URL of the Maximo Manage application server when connecting to the IBM Maximo Application Suite. In addition, you must use an API Key to authorize access to the server. When connecting to a Maximo Asset Management 7.6.1.x application server, you should provide the URL of the Maximo 7.6.1.x application server. You can use either a username and password or an API Key to access the IBM Maximo Asset Management 7.6.1.x application server.

### Modifying applications

From the Application list page, select an application to modify.

App name	Description	Status	Last update	Last updated by	Version	Revision
<a href="#">DAS</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">INSPECTION</a>	Inspections	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">LOGIN</a>		ACTIVE	2021-07-06T08...	UPDATEDB	2.6.137	0
<a href="#">MANAGE-SHELL</a>	Manage Shell A...	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">MAXDEMO-APP</a>	Graphite Demo ...	ACTIVE	2021-07-06T08...	UPDATEDB	2.6.137	0
<a href="#">TECHMOBILE</a>	Technician	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">TECHNICIANNEW</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">TECHNICIANTS</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">TEST</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0

Items per page: 10 | 1-9 of 9 items | 1 | 1 of 1 pages

The application XML is available for editing in the XML editor.

The screenshot shows the IBM Maximo XML editor interface for the TECHMOBILE application. The navigation tree on the left includes components like 'maximo-application Maximo Mobile', 'properties nba7e', 'menu k9p4v', 'maximo-datasource synonymdomainData', 'maximo-datasource woDetailds', 'maximo-datasource dsFailureList', 'maximo-datasource dsworktype', and 'pages pages' (which is selected). The main editor area displays XML code for a list item, including details for a task controller, dialog, and data list. The code includes attributes for icons, labels, and data sources.

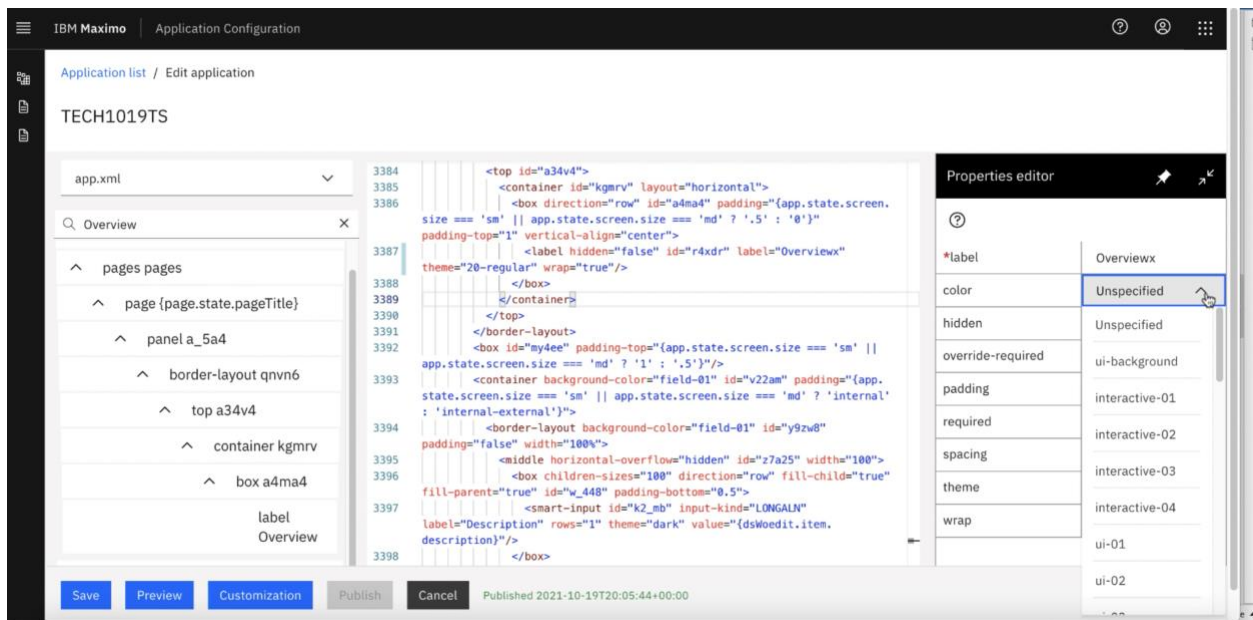
Select the file that you want to modify and make changes using the embedded XML editor.

Use the navigation tree to locate specific components in the XML code. Select an entry to locate the component in the application XML file. For example, select the **pages** component to show all the pages that are defined in the application.



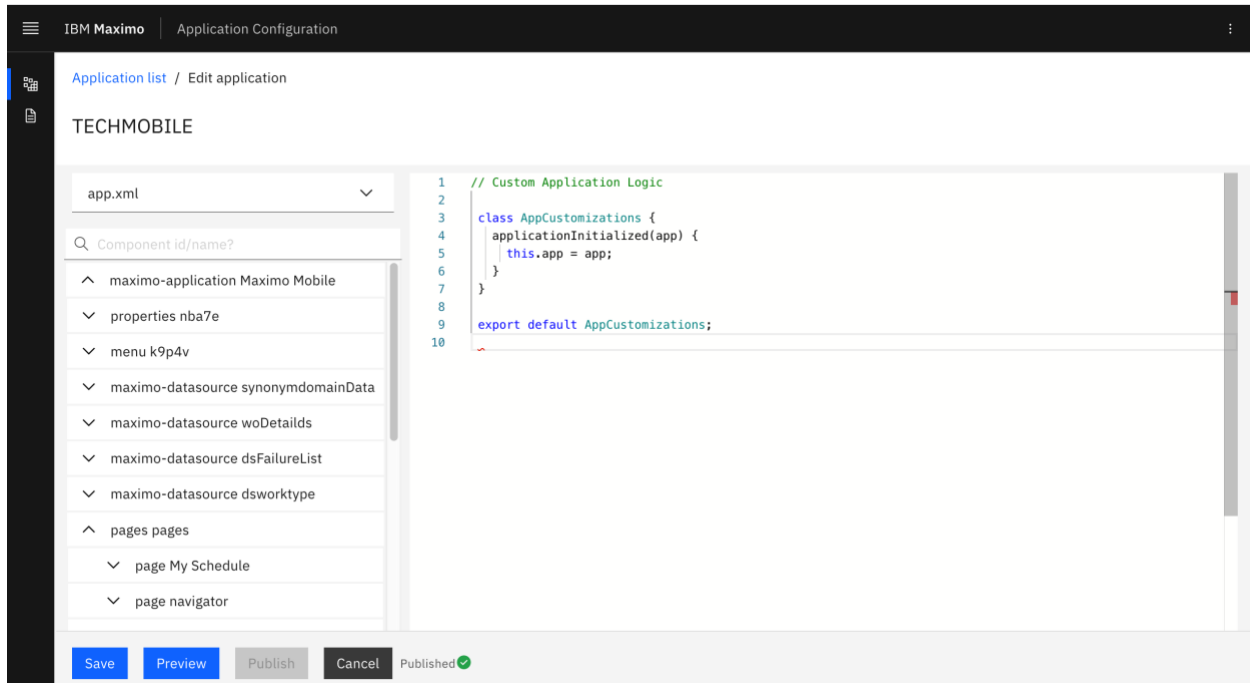
You can use the component search field to filter entries in the navigation tree. For example, if you search for **dialog**, the navigation tree displays only dialog components that are found in application pages. Alternatively, you can also search for a component ID. By default, searches return results that are like the term you entered. To perform a search for an exact term, add an equals sign before the search entry. For example, **=id771**.

As an alternative to the XML editor, you can use the Properties editor to update component values in the XML file. Select a component from the navigation tree to view the properties for that component. Required attributes are marked with an asterisk.



Attribute names should be declared using lowercase letters, for example, `item.somename`. Attributes always be declared and used in lowercase because the Maximo application server provides the data field in JSON as lowercase.

If your changes require custom JavaScript to implement, you can add code to the `AppCustomizations.js` file and then reference it in the application XML code.



After you complete your changes, you can preview the application in your browser before you publish it on the Maximo application server.

Do not publish an application immediately after you start editing the application. When you start editing an application, the local React preview server is started and may take a few minutes to complete the startup process. Errors can occur if you publish at the same time that the React server is starting. View the terminal console for the message that the preview server is available.

Once an application is published, mobile devices can then download the new version of the application. If you edit application files outside of the MAF Configuration application, then you must ensure the integrity of those files. Use a virus or malware scanner before you publish the files to the Maximo application server.

### Duplicating applications

If you don't want to modify a default application, you can duplicate it, give it a new name, and then change the new application.

Select the application that you want to duplicate and then select Duplicate from the application menu.

IBM Maximo | Application Configuration

### Application list

App name	Description	Status	Last update	Last updated by	Version	Revision	
<a href="#">DAS</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDDB	8.0.0.0	0	
<a href="#">INSPECTION</a>	Inspections	ACTIVE	2021-07-06T08...	UPDATEDDB	8.0.0.0	0	
<a href="#">LOGIN</a>		ACTIVE	2021-07-06T08...	UPDATEDDB	2.6.137	0	
<a href="#">MANAGE-SHELL</a>	Manage Shell A...	ACTIVE	2021-07-06T08...	UPDATEDDB	8.0.0.0	0	
<a href="#">MAXDEMO-APP</a>	Graphite Demo ...	ACTIVE	2021-07-06T08...	UPDATEDDB	2.6.137	0	
<a href="#">TECHMOBILE</a>	Technician	ACTIVE	2021-07-06T08...	UPDATEDDB	8.0.0.0	0	<a href="#">Edit</a> <ul style="list-style-type: none"> <li><a href="#">Duplicate</a></li> <li><a href="#">Redownload</a></li> </ul>
<a href="#">TECHNICIANNEW</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDDB	8.0.0.0	0	
<a href="#">TECHNICIANTS</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDDB	8.0.0.0	0	
<a href="#">TEST</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDDB	8.0.0.0	0	

Items per page: 10 | 1-9 of 9 items | 1 | 1 of 1 pages

Specify a name for the new application and provide a brief description. If the application is a mobile application, ensure that the Mobile option is enabled.

### Duplicate application

\*Application name  
TECHMOBILE Copy

Brief description  
Technician

Mobile?  Yes

Last update: 2021-07-06T08:47:13-04:00

Last updated by: UPDATEDDB

Cancel Duplicate

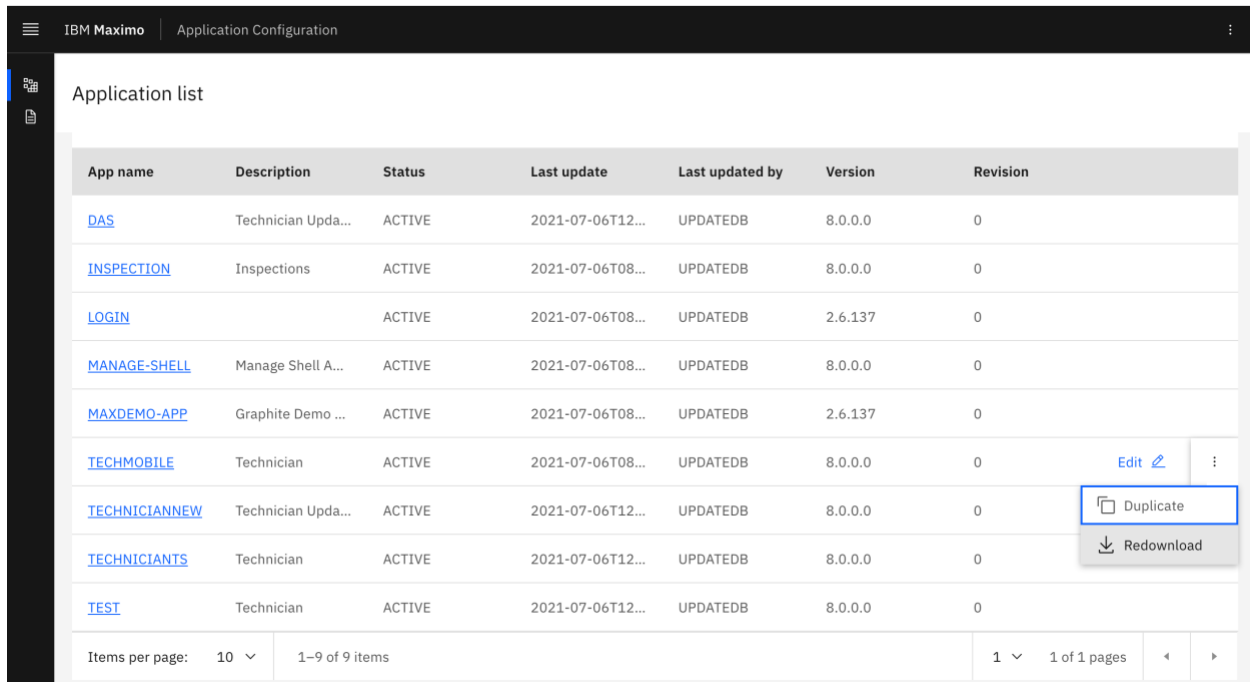
You can modify the name and description of any application from the application list by selecting the Edit action.

When you duplicate an application, it inherits the security settings from the original application. You can change security settings for an application with the Security Groups application. Refer to [Creating security groups](#) for more information.

When you duplicate a mobile application, the **Mobile?** option is disabled by default. Disabling the **Mobile?** option prevents the duplicated application from automatically being pushed to mobile devices. You can re-enable the option to designate the duplicated application as a mobile application, making it available to mobile devices.

### Redownloading applications

You can download the most recently published version of an application from the Maximo application server anytime by selecting Redownload from the application menu.



The screenshot shows the IBM Maximo Application Configuration interface. The main content area is titled "Application list" and contains a table with the following columns: App name, Description, Status, Last update, Last updated by, Version, and Revision. The table lists several applications, including DAS, INSPECTION, LOGIN, MANAGE-SHELL, MAXDEMO-APP, TECHMOBILE, TECHNICIANNEW, TECHNICIANTS, and TEST. The TECHNICIANNEW row is selected, and a context menu is open over it, showing options for "Duplicate" and "Redownload". The "Redownload" option is highlighted with a blue border. At the bottom of the table, there is a pagination control showing "Items per page: 10" and "1-9 of 9 items".

App name	Description	Status	Last update	Last updated by	Version	Revision
<a href="#">DAS</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">INSPECTION</a>	Inspections	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">LOGIN</a>		ACTIVE	2021-07-06T08...	UPDATEDB	2.6.137	0
<a href="#">MANAGE-SHELL</a>	Manage Shell A...	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">MAXDEMO-APP</a>	Graphite Demo ...	ACTIVE	2021-07-06T08...	UPDATEDB	2.6.137	0
<a href="#">TECHMOBILE</a>	Technician	ACTIVE	2021-07-06T08...	UPDATEDB	8.0.0.0	0
<a href="#">TECHNICIANNEW</a>	Technician Upda...	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">TECHNICIANTS</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0
<a href="#">TEST</a>	Technician	ACTIVE	2021-07-06T12...	UPDATEDB	8.0.0.0	0

When you redownload an application, any unpublished updates that you made to an application are discarded.

### Deleting applications

You can delete applications that you have created through the duplication process from the application menu. Default applications cannot be deleted.

### Configuring applications

Maximo for EAM applications can include several application XML files. In the case of the Technician application, there are two files, **app.xml** and **wo-card-group.xml**. The **wo-card-group.xml** file is referenced in the **app.xml** file.

Remember that configuring a Maximo for EAM application is defined as modifying existing presentation XML for an application. Customization refers to creating new JavaScript code to support a configuration change.

Common configurations include adding new fields to an application card, updating field labels, and adding a new card.

Whenever you are modifying an application component, refer to the Maximo component reference for details about properties, attributes, and elements.

The application XML files are downloaded to your local working directory when you select an application to configure. Back up these files before you edit them to ensure that you can restore these files if necessary.

### Anatomy of an application XML file

The `app.xml` file contains global elements and attributes that are defined at the beginning of the file. Individual pages and elements of those pages are defined later in the file.

<b>Maximo-application</b>
<b>Application navigation menu</b>
<b>Application navigation tiles for mobile navigation</b>
<b>Application state that defines variables for all application pages</b>
<b>Application data sources that defines data sources for all pages</b>
<b>Pages container that includes all application pages.</b>
<b>Individual page</b>
<b>Page state variables</b>
<b>Page level data sources</b>
<b>Page header</b>
<b>Page body</b>
<b>Dialogs used in page</b>

This screen is a section of the Technician application XML that focuses on global elements and properties. The data sources defined at the top of the file can be referenced and used by elements defined throughout the file.

```
1 <maximo-application controller="AppController" theme="touch" product-name="Max
2 > <properties id="nba7e">...
6 </properties>
7 > <menu slot="navigation-items" id="k9p4v">...
11 </menu>
12 > <maximo-datasource id="synonymdomainData" lookup-data="true" object-structur
23 </maximo-datasource>
24 > <maximo-datasource id="dsFailureList" lookup-data="true" default="true" page
34 </maximo-datasource>
35 > <pages id="pages">...
2001 </pages>
2002 > <messages id="n5_2r">...
2025 </messages>
2026 </maximo-application>
```

<b>XML element</b>	<b>Description</b>
<maximo-application>	The root element of the Maximo application XML file.
<properties>	The property definitions used across the application.
<menu>	The navigational side bar appearing on the left side of the application.
<maximo-datasource>	<p>The definition of a Maximo data source. It can be defined at a global level or more targeted if nested within a component structure.</p> <p>Data sources provide data to a component that is defined in application XML. The &lt;maximo-datasource&gt; element identifies a Maximo Object Structure (API) that is used to retrieve data for that data source.</p> <p>If you add a data source, include only the attributes that you need. Including all of the attributes of the data source can lead to performance issues.</p>
<pages>	This element contains the details of each page that is displayed in the application. Most configuration work that you perform involves page structures.
<messages>	This element contains information about messages that are used by the application.

This screen is a section of the Technician application XML that is focused on a page structure. The data sources defined at this location of the file is referenced and used by elements defined within the local page definition.

```

35 <pages id="pages">
36 <page padding="true" id="schedule" path="/schedule" title="My Schedule" icon="maximo:schedule" contr
37 <states id="nmvrw">...
51 </states>
52 <maximo-datasource id="dswolist" object-structure="mxapiwodetail" saved-query="uxtechnicianownerfi
53 <schema id="gjw7p">...
118 </schema>
119 <maximo-datasource-override id="todaywoassignedDS" pre-load="true" saved-query="ASSIGNEDWOLIST"/>
120 <maximo-datasource-override id="pmduewolistDS" saved-query="PMWOLIST" where="schedfinish=&quot;f
121 </maximo-datasource>
122 <maximo-datasource id="wodetails" object-structure="mxapiwodetail" item-url="{page.params.href}">...
247 </maximo-datasource>
248 <maximo-datasource id="dsnewreading" lookup-data="true" object-structure="mxapiaIndomain" page-size
257 </maximo-datasource>
258 <json-datasource id="dsstatusDomainList" src="{}" items="statusItems" selection-mode="single" id
263 </json-datasource>
264 <header-template hide-breadcrumb="true" title-column-width="{app.state.screen.size === 'md' ? 48 :
273 </header-template>
274 <stacked-panel selected-index="{page.state.selectedSwitch}" id="qzfnm">...
323 </stacked-panel>
324 <dialogs id="wnynq">...
549 </dialogs>
550 </page>

```

XML element	Description
<states>	Properties that are used across the application.
<maximo-datasource>	<p>The definition of a Maximo data source.</p> <p>In this example, it is embedded within a &lt;page&gt; element. This data source is defined for only that page.</p> <p>If you add a data source, include only the attributes that you need. Including all of the attributes of the data source can lead to performance issues.</p>
<json-datasource>	<p>A JSON data source that provides static data to a component defined in application XML.</p> <p>In this example, it is embedded within a &lt;page&gt; element. This data source is defined for only that page.</p>
<header-template>	A layout for a header section of a page.
<stacked-panel>	This element defines the panels that are used in the page.
<dialogs>	This element defines the dialog boxes that pop up in the page.
<id>	A unique identifier for an element. When you add an element, you do not need to assign a value to this attribute. The application build process assigns the element a unique ID.

	If you plan to reference this element in another part of the XML, you can create a unique ID value. Do not change an ID value after it is assigned.
--	---

## Referencing other application XML files

Splitting application XML components across several files can help with maintenance and modularization efforts. You can reference an application XML file by using the `<include>` element.

In the following example for the Technician application, an XML file that contains work order card components is included in the main `app.xml` file:

```
<include src="./wo-card-group.xml" id="jwq72"/>
```

## Upgrading configuration changes to a new version of the application

When you upgrade to a new Maximo application server release, the upgrade process overwrites application XML files with new versions. You must preserve the configuration changes you made to those application XML files before you start the upgrade process. You can then reapply your configuration changes to the application XML files of the new release.

The migration process does not preserve comments made in the XML files. You must manually save and reapply comments after the migration process has completed.

## Command variable legend

The variables in the following table are used in examples throughout this document:

Variable	Description
<code>\${original_xml_name}</code>	The name of the out-of-the-box application XML file from the current release.
<code>\${modified_xml_name}</code>	The name of the application XML file from the current release that you modified.
<code>\${delta_output_xml_name}</code>	The name of the file that contains delta information from the comparison of the original and modified application XML files.
<code>\${delta_xml_name}</code>	The name of the delta XML file.
<code>\${input_xml_name}</code>	The name of the out-of-the-box application XML file from the new release. Your configuration changes are applied to the contents of this file.
<code>\${output_xml_name}</code>	The name of the output XML file.



Download and extract the source files for the application

1. Open a command line and run the Docker image using the following command, replacing *x* with the latest image available:

```
docker run -it --privileged -p 3001:3001 -p 3006:3006 -v  
${maximo_workspace_directory}:graphite/.workspace -it  
cp.icr.io/cp/manage/maf-tools:8.7.x
```

2. In a browser, open the following URL:

<http://localhost:3001>

3. Enter the Maximo application server URL and your username and password.
4. Select an application from the application list to configure.

When you select an application to configure, the files for that application are downloaded to the workspace directory that you specified.

Save the configuration updates you made for a current application

Open a command line and run the following command to compare the current application XML file contents to the original version shipped with the product replacing *x* with the latest image available:

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v ${app_directory}:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x diff --original src/${original_xml_name}  
--modified src/${modified_xml_name} --deltaOutput src/${delta_xml_name}
```

For example,

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v  
/Users/myname/Documents/myworkspacedirectory/MAXADMIN-  
myserver.mycompany.com.9083/myappdirectory:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x diff --original src/app.orig.xml --  
modified src/app.xml --deltaOutput src/mydeltaforappxml.xml
```

This command produces a file that contains the differences that are found between the current application XML and the version that shipped with the product.

Upgrading the Maximo application server

After you have saved all of the updates that you have made to the application in the currently deployed version, you can upgrade to the new version.

Reapplying your changes to the new versions of the application XML file.

Run the following command to incorporate the configuration information that you saved into the new version of the application XML shipped with the product replacing *x* with the latest image available:

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v ${app_directory}:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x apply --delta src/${delta_xml_name} --  
input src/${orig_xml_name} --output src/${orig_xml_name}
```

For example,

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v  
/Users/myname/Documents/myappworkingdirectory:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x apply --delta src/mydeltaforappxml.xml -  
-input src/app.xml --output src/app.xml
```

Publishing the updated application to the Maximo application server

After reapplying changes to the new versions of the application XML file, you can publish the application from the MAF configuration tool UI or from the command line. If you publish the application through the UI, you can preview and verify the application before publishing it.

To publish the updated application from the command line, run the following command to build an application image, replacing *x* with the latest image available:

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v ${app_directory}:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x build:production
```

Package and publish the production image for the application using the following command, replacing *x* with the latest image available:

```
docker run -it --privileged --entrypoint "/graphite/scripts/graphite-  
tools.js" --workdir /graphite/app -v ${app_directory}:/graphite/app  
cp.icr.io/cp/manage/maf-tools:8.7.x upload
```

Filtering record returned from a query

The number of records returned from a query can sometimes be unmanageable on mobile devices. In some cases, you might want to limit the number of records received. For example, you might want to limit the number of records displayed in a work list.

You can modify the query located on the Maximo application server using the Object Structure application. For example, if you want to use the work order owner value instead of assignments, you can change the ASSIGNEDWOLIST query located within the MXAPIWODETAIL object structure. The new WHERE clause would be:

```
(siteid = (select defsite from maxuser where userid =:USER) and status in  
(select value from synonymdomain where domainid = 'WOSTATUS' and maxvalue not  
in ('CAN', 'CLOSE', 'COMP', 'WAPPR')) and istask=0 and owner= (select laborcode
```

```
from labor, maxuser where labor.personid = maxuser.personid and
maxuser.userid=:USER)
```

Then, in the XML file that you are configuring, change the `uxtechnicianownerfilter` value to `ASSIGNEDWOLIST`.

```
<maximo-datasource controller="ScheduleDataController" id="dswolist" object-
structure="mxapiwodetail" order-by="wopriority" pre-load="false" saved-
query="ASSIGNEDWOLIST" selection-mode="single" where=wopriority
```

## Data sources

Data sources provide data to a component that is defined in application XML. A JSON data source provides static data. A Maximo data source identifies a Maximo Object Structure (API) that is used to retrieve data from the Maximo application server for that data source.

The following screen shows a JSON data source:

```
766 <json-datasource id="jtoolsds" src="{[]}" items="wptool" schema="schema" id-attribute="wpitemid">
767 <schema id="a8vqr">
768 <attribute name="itemnum" id="dgy62"/>
769 <attribute name="description" id="y2wmq"/>
770 <attribute name="itemqty" id="e8xk5"/>
771 <attribute name="location" id="qz7dd"/>
772 <attribute name="locationDesc" id="qzdwy"/>
773 <attribute name="wonumDesc" id="z8_nm"/>
774 <attribute name="locHours" id="vpgxp"/>
775 <attribute name="itemnumDesc" id="n9r2v"/>
776 <attribute name="wpitemid" id="grz5a"/>
777 </schema>
778 </json-datasource>
```

The following two screens show snippets of Maximo data source definitions:

```
52 <maximo-datasource id="dswolist" object-structure="mxapiwodetail" saved-query="uxtechnicianownerfilter" order-by="wopriority"
53 pre-load="false" controller="ScheduleDataController" selection-mode="none">
54 <schema id="gjjw7p">
55 <attribute name="plussgeojson" id="wymyz"/>
56 <attribute name="wonum" unique-id="true" searchable="true" id="g4dk9"/>
57 <attribute name="description" searchable="true" id="ej423"/>
58 <attribute name="location.description" searchable="true" id="d_m_3"/>
59 <attribute name="location.location" searchable="true" id="gy6zb"/>
60 <attribute name="wopriority" searchable="true" id="x3nyj"/>
```

```
849 <maximo-datasource id="woDetailResource" default="true" object-structure="mxapiwodetail"
850 where="wonum=&quot;{page.params.wonum}&quot; and siteid=&quot;{page.params.siteid}&quot;" controller="WorkOrderDataController"
851 item-url="{page.params.href}">
852 <schema id="qmvzy">
853 <attribute name="plussgeojson" id="yn258"/>
854 <attribute name="workorderid" unique-id="true" id="p6d4b"/>
855 <attribute name="wonum" id="q9qmr"/>
856 <attribute name="title" id="gmp_2"/>
857 <attribute name="description" id="kq9yq"/>
```

Each data source has an `id` attribute, which is used when a UI component references the data source.

The `saved-query` attribute references an object structure query that is associated to the object structure that is defined in the `object-structure` attribute. The `where` clause identifies the filtering of the data in the data source. The schema consists of a list of attributes. These attributes identify the API fields that are available to the UI component using this data source.

A Maximo data source can include child data sources which show data that is related to the parent or outer data source. For example, the outer data source might be work order data. A child data source might be tasks that are related to an instance of the outer work order data source.

A field can be identified as `{item.wonum}`, where `item` represents the current record of the data source. The term `item` is a reserved variable. Do not confuse it with the use of `item.` as a Maximo dot notation that refers to a Maximo relationship name that called `item`.

The following Maximo data source snippet shows an example of how to override the `saved-query` and `where` properties of the `pmduewolistDS` data source:

```
121 <maximo-datasource-override id="pmduewolistDS" saved-query="PMWOLIST"
122 where="schedfinish=>&quot;&amp;SYSDAY&amp;&quot; and schedfinish<=&quot;&amp;SYSDAY&amp;+7D&quot;"/>
```

A Maximo data source can be overridden where any property of the data can be replaced by a property value in the override. Do not override the `object-structure` property.

In this next series of examples, a `wo` data source is used in several contexts.

```
<maximo-datasource id="wo" object-structure="mxapiwodetail"
where="location=&quot;SHIPPING&quot;" pre-load="false" page-size="10"
include-counts="false" order-by="wonum ascending"
controller="WODetailController">
  <maximo-datasource id="taskWO" relationship="showtasks">
  </maximo-datasource>
</maximo-datasource>
```

In the following table example, `wo` is a detail data source for work order details and `taskWO` includes the work order tasks for that data source. It is updated at the `currentItem` of the parent whenever it is updated.

```
<table datasource="wo" id="qexqa">
  <table-column name="wonum" sigoption="workview.read" id="jw2jb">
    <link label="{item.wonum}" page="woDetailPage" id="n8wnx"/>
  </table-column>
  <table-column name="estlabcost" license="tpaebase" id="sadsd"/>
</table>
```

In this case, the table is using the `wo` data source. The table column `wonum` uses a custom cell render to render a link and the value of the link is **item.wonum**. The `item` variable always points to the current item or record in a data source as it is being rendered.

Consider the following data sources:

```
<maximo-datasource id="wo" object-structure="mxapiwodetail"
where="location='SHIPPING'" pre-load="false" page-size="10"
include-counts="false" order-by="wonum ascending"
controller="WODetailController">
</maximo-datasource>
```

```
<maximo-datasource id="woview" object-structure="mxapiwodetail"
where="location='SHIPPING'" pre-load="false" page-size="10"
include-counts="false" order-by="wonum ascending"
controller="WODetailController">
</maximo-datasource>
```

These two data sources have no relationship to each other. A typical application can have several data sources. Some data sources are related while others are not.

Consider the following fields:

```
<field value="{ wo.item.description}" />
<field value="{ woview.item.description}" />
```

The `wo` and `wodescription` values are data sources. The `item` value is the current record. The `description` value is the value that is specified for the field. If the data source moves to a new item or record, then this field is updated.

## XML configuration examples

The following examples show how to configure XML for the Technician application.

---

---

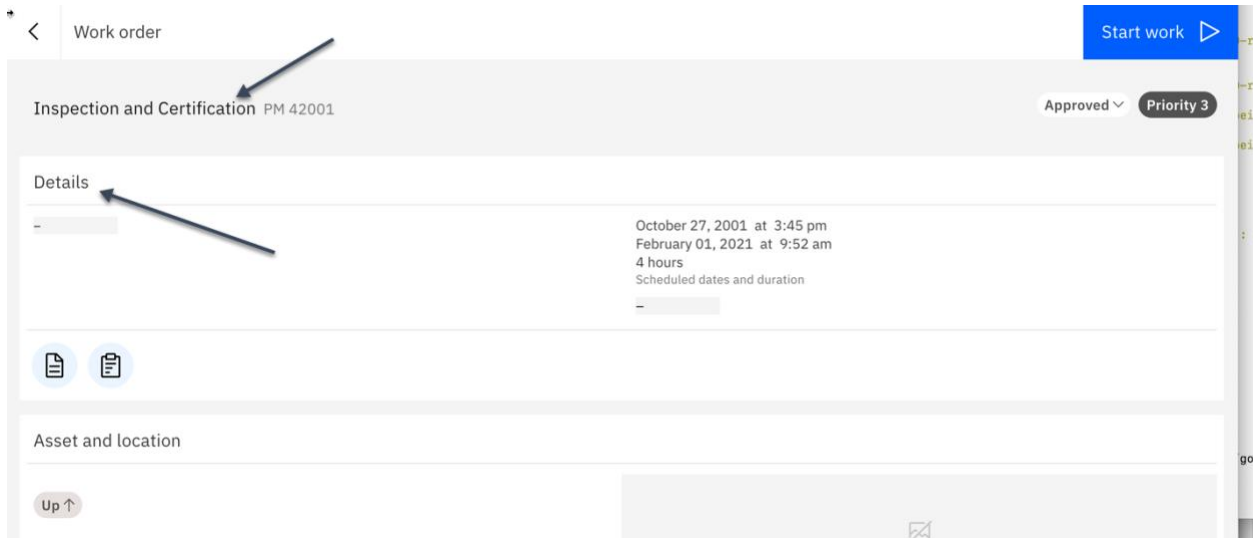
### Changing a label and adding a field

A common configuration change to make to an application is to change the name of a field label. You can also add a field that does not currently exist on an application page.

In the following example, an update is made to a label and a field for a work order. The label currently reads Details. The new field is added to the work order details.

---

---



The following example displays an XML snippet that has values that are used in the UI:

```
1151 </header-template>
1152 <adaptive-row id="qb8m7">
1153   <adaptive-column xlarge-width="65" large-width="55" medium-width="100" small-width="100" id="qxwv">
1154     <border-layout fill-parent="true" padding="true" id="zrb3y">
1155       <middle vertical-align="center" id="w7q4y">
1156         <box fill-child="true" padding-start="{app.state.screen.size === 'sm' || app.state.screen.size === 'md' ? '0' : '-.25'}" id="w7q4y">
1157           <wrapped-text size="large" label="{woDetailResource.item.description}" sub-label="{woDetailResource.item.sub-label}" id="w7q4y">
1158             {woDetailResource.item.description}
1159           </wrapped-text>
1160         </box>
1161       </middle>
1162     </border-layout>
1163   </adaptive-column>
1164   <adaptive-column xlarge-width="35" large-width="45" medium-width="100" small-width="100" id="dwb">
1165     <box fill-child="true" padding-bottom="{app.state.screen.size === 'sm' || app.state.screen.size === 'md' ? '0' : '-.25'}" id="dwb">
1166       <tag-group align="none" tags="{woDetailResource.item.computedWODtlStatusPriority}" id="j85kz">
1167         {woDetailResource.item.computedWODtlStatusPriority}
1168       </tag-group>
1169     </box>
1170   </adaptive-column>
1171 </adaptive-row>
1172 <box padding="{app.state.screen.size === 'sm' || app.state.screen.size === 'md' ? '0' : '-.25'}" id="d3rqy">
1173   <container background-color="field-01" padding="{app.state.screen.size === 'sm' || app.state.screen.size === 'md' ? '0' : '-.25'}" id="d3rqy">
1174     <border-layout padding="false" fill-parent="true" top-border="true" id="egry_">
1175       <top id="d3rqy">
1176         <label label="Details" theme="20-regular" id="bp5bq"/>
1177       </top>
1178       <middle id="kevgg">
1179         <border-layout fill-parent="true" top-border="true" id="md8yv">
1180           <top id="bkxw4">
```

Now, change the label to **WO Details** and replace the description with the supervisor who is assigned to the work order:

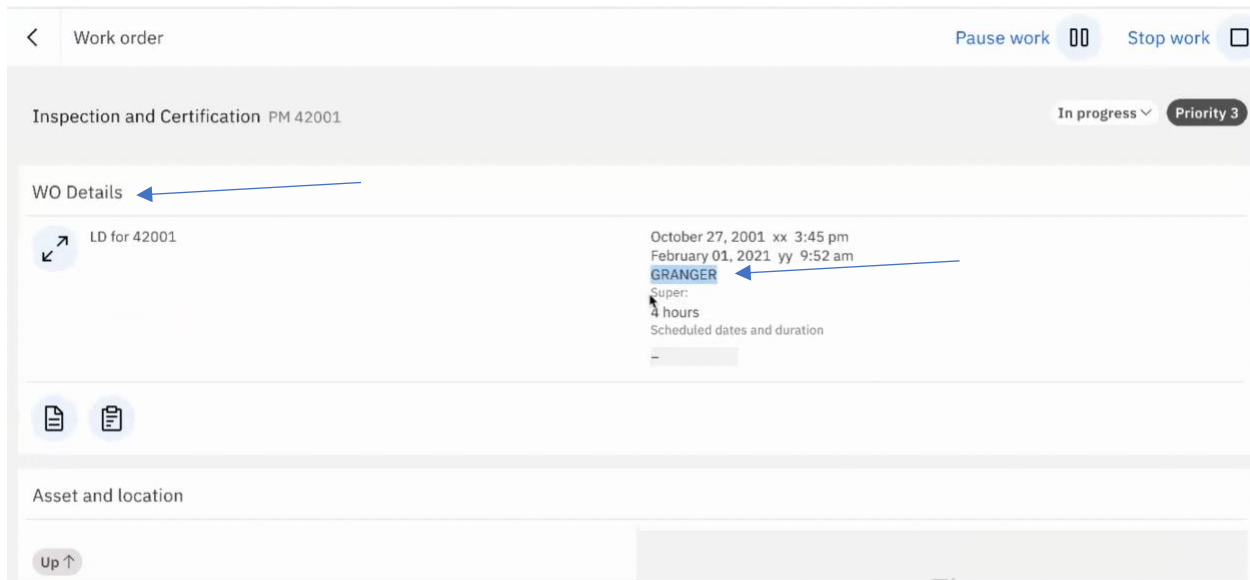
```
1150 <button slot= "buttons" label= "Start travel" icon= "carbon:play" loading= {page.state.loading} K1
1151 </header-template>
1152 <adaptive-row id="qb8m7">
1153 <adaptive-column xlarge-width="65" large-width="55" medium-width="100" small-width="100" id="qxw
1154 <border-layout fill-parent="true" padding="true" id="zrb3y">
1155 <middle vertical-align="center" id="w7q4y">
1156 <box fill-child="true" padding-start="{app.state.screen.size === 'sm' || app.state.screen.s
1157 <wrapped-text size="large" label="{woDetailResource.item.supervisor}" sub-label="{woDeta
1158 </box>
1159 </middle>
1160 </border-layout>
1161 </adaptive-column>
1162 <adaptive-column xlarge-width="35" large-width="45" medium-width="100" small-width="100" id="dwb
1163 <box fill-child="true" padding-bottom="{app.state.screen.size === 'sm' || app.state.screen.siz
1164 <tag-group align="none" tags="{woDetailResource.item.computedWODtlStatusPriority}" id="j85kz'
1165 </box>
1166 </adaptive-column>
1167 </adaptive-row>
1168 <box padding="{app.state.screen.size === 'sm' || app.state.screen.size === 'md' ? '0' : '-.25'}" id
1169 <container background-color="field-01" padding="{app.state.screen.size === 'sm' || app.state.scre
1170 <border-layout padding="false" fill-parent="true" top-border="true" id="egry_">
1171 <top id="d3rqy">
1172 <label label="WO Details" theme="20-regular" id="bp5bq"/>
1173 </top>
1174 <middle id="kevgg">
```

One more change is needed. Add the supervisor to the `woDetailResource` data source so that it is available from the API.

```
1 <page id="workOrderDetails" title="Work order" controller="WorkOrderDetailsController">
  <maximo-datasource id="woDetailResource" default="true" object-structure="mxapiwodetail" where="wor
    <schema id="qmvzy">
      <attribute name="supervisor" id="v55_6"/>
      <attribute name="plussgeojson" id="y1258"/>
      <attribute name="workorderid" unique-id="true" id="p6d4b"/>
      <attribute name="wonum" id="q9qmr"/>
      <attribute name="title" id="gmp_2"/>
      <attribute name="description" id="kq9yq"/>
      <attribute name="description_longdescription" id="m75qa"/>
      <attribute name="problemcode" id="kn86y"/>
      <attribute name="failurecode" id="ka5d_"/>
      <attribute name="status" id="g35x5"/>
      <attribute name="status_description" id="a33d_"/>
      <attribute name="statusdate" id="gjnk4"/>
      <attribute name="date" id="b4226"/>
      <attribute name="duration" id="awq9y"/>
      <attribute name="type" id="rk8_4"/>
      <attribute name="number" id="k69_8"/>
```

The `id` attribute was automatically generated after the application was built.

After the changes are implemented, the application page uses the updated values:



---

## Signature options

You can use signature options to control the visibility of a UI component in an application. For example, if you want to show or hide a control on an application page, you can use the `sigoption` attribute.

```
<button ${applicationname.sigoptionname}="read" />  
<button ${objectstructurename.sigoptionname}="read" />
```

The `read` value is the name of the signature option for this application. If the signature option is granted access to the user group of the user, then the button is displayed. If not, the button is hidden.

Signature options are valid only in connected mode.

## Configuring queries

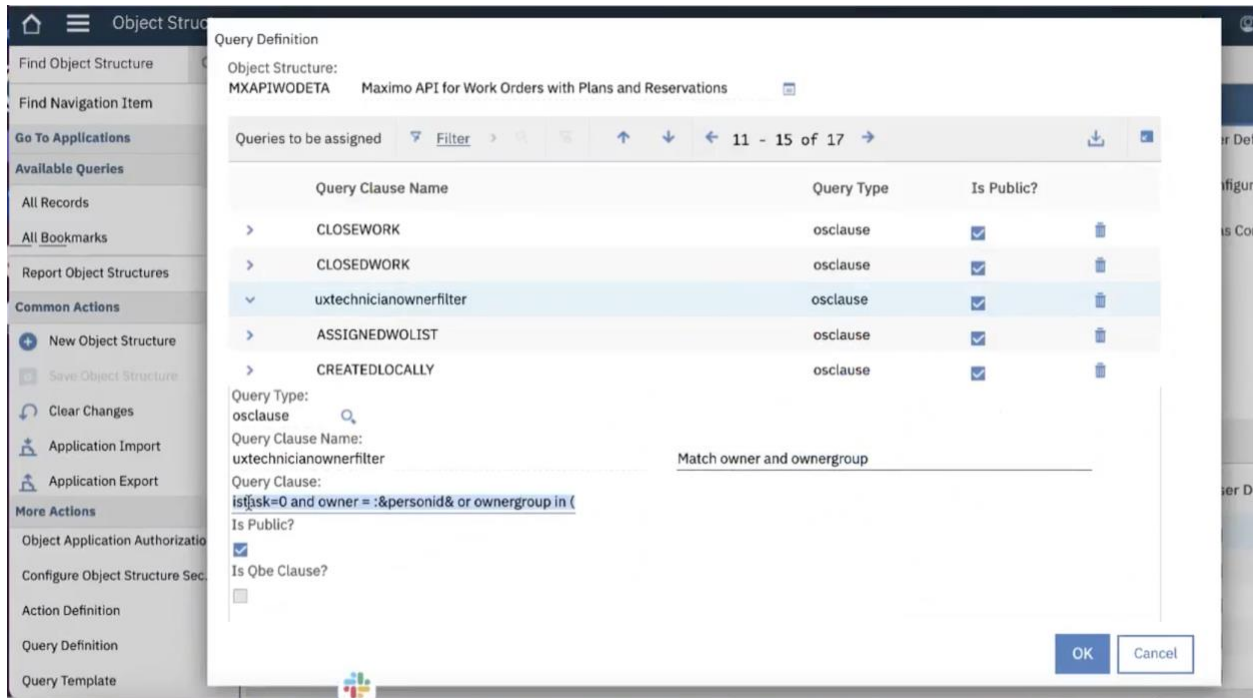
You might need to create or modify queries used by applications. You can modify existing queries on the Maximo application server, or you can extend queries in the `app.xml` for an application.

### Configuring queries on the Maximo application server

If you want to use a custom query across many different applications, you might want to update a default query defined on the Maximo application server. Customizing a query on the server alleviates the need to make changes to the `app.xml` file for each application.



For example, if your organization uses a custom term for a work order status, you could update the **uxtechnicianownerfilter** query clause of the MXAPIWODETAIL object.



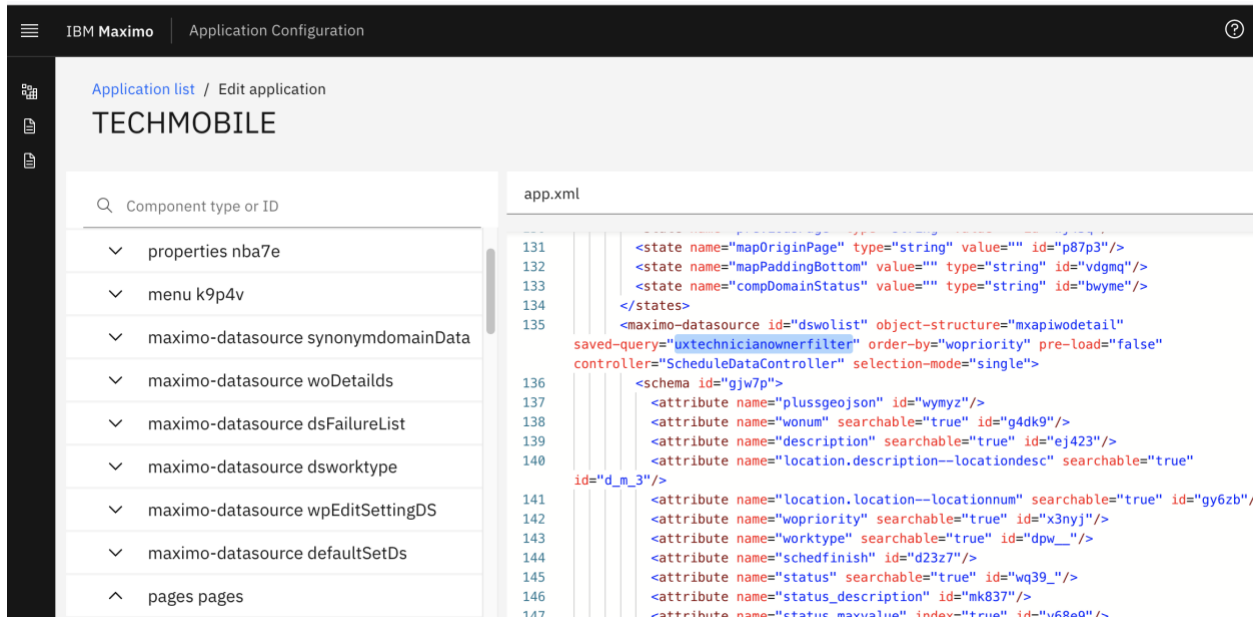
By default, the uxtechnicianownerfilter query clause returns all non-task-related work orders where the owner is the current user or the user is a responsible party of a particular group.

```
istask=0 and owner = :&personid& or ownergroup in (select persongroup from persongroupteam where resparty=:&personid&)
```

If your organization uses the term **Actively working** to describe a work order that is in progress, you can further filter results returned by the query by appending the condition **status='Actively working'**.

```
istask=0 and owner = :&personid& or ownergroup in (select persongroup from persongroupteam where resparty=:&personid& and status='Actively working')
```

Now queries associated with the mxapiwodetail API return all non-task-related work orders where the owner is the current user or the user is a responsible party of a particular group with a status of Actively working.



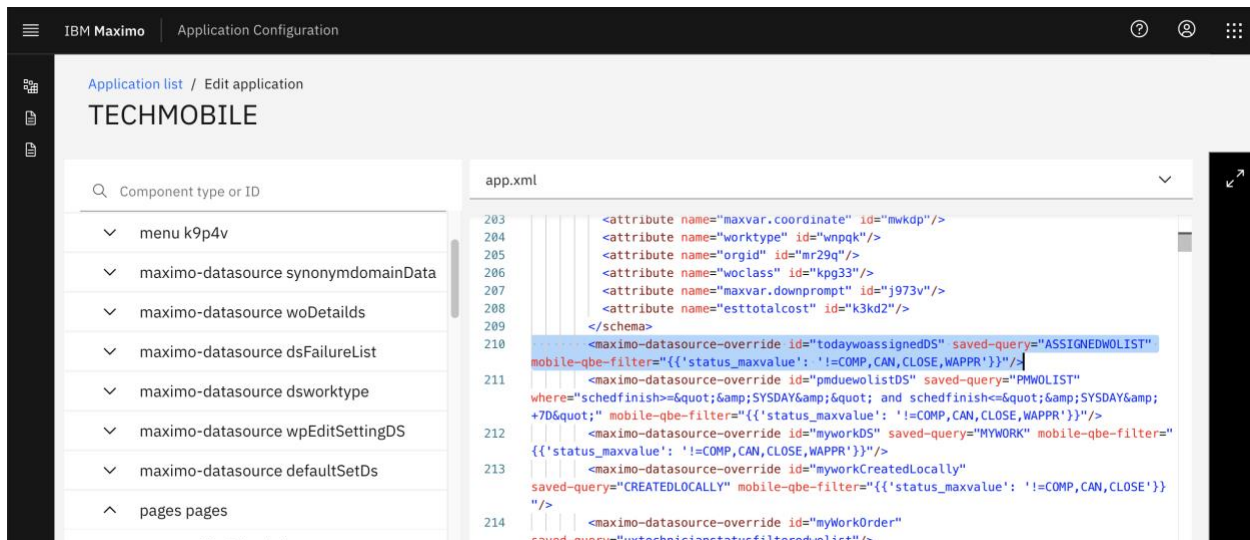
## Configuring queries in the app.xml file of an application

If you want to customize a query for an application, you can override and extend a default query defined on the Maximo application server. Customizing a query in the app.xml file for a specific application alleviates the need to make changes to the default query which impacts all applications that use that query.

For example, the ASSIGNEDWOLIST query from the MXAPIWODETAIL object structure returns work order assignment data to an application.

```
(siteid = (select defsite from maxuser where userid =:USER) and status in
(select value from synonymdomain where domainid = 'WOSTATUS' and maxvalue not
in ('CAN', 'CLOSE', 'COMP', 'WAPPR')) and istask=0 and wonum in (select wonum
from assignment where laborcode = (select laborcode from labor, maxuser where
labor.personid = maxuser.personid and maxuser.userid=:USER) and (siteid =
(select defsite from maxuser where userid =:USER))))
```

The Technician application uses a datasource override to filter results of the ASSIGNEDWOLIST query to the application. The mobile-qbe-filter attribute is used to exclude query data that is returned if the application is operating in offline mode.



So, for example, work orders with a status of COMP will not be returned to the Technician application if it is in offline mode.

## Configuring IBM Maximo Health and Predict – Utilities

Configuring the IBM Maximo Health and Predict – Utilities application by modifying existing presentation XML code.

Maximo Health and Predict – Utilities signature options

Several Maximo Health and Predict – Utilities sigoptions should not be modified during application configuration.

### Predict

PREDICTASSETCOLS  
 PREDICTASSETDETAILS  
 PREDICTASSETGROUP  
 PREDICTASSETVIEW  
 PREDICTWORKQUEUE

### Utilities

EUASSETCOLS  
 EUASSETDETAILS Change the fields displayed in the asset info section  
 EUSCORES  
 EUCONTAINER  
 EUMODELRUNASSET  
 EUMODELRUNGROUP  
 EUMODELVIEW

Maximo Health, Maximo Predict, and Maximo Health and Predict - Utilities XML configuration  
The following examples show how to configure XML for the Maximo Health, Maximo Predict, and Maximo Health and Predict - Utilities applications for supported configurations.

- Change the fields displayed in the asset info section
- Remove the MRR card
- Change the order of the top row of cards
- Change the order of the sections
- Change the data showing in the asset detail card
- Create a new card

#### *Change the fields displayed in the asset info section*

Locate the assetDetail page and then add the new attribute to the assetDetailResource datasource, For example, changeby.

Add an attribute to an asset page (individual page for an individual asset). Examples of an attribute would be asset number, site, and location.

```
<maximo-datasource id="assetDetailResource" default="true" depends-
on="{page.params.linkedDS}" dependent-id="{page.params.ids}" object-
structure="mxapiasset" controller="AssetDetailResourceController" notify-
when-parent-loads="false">
  <schema id="ywr9_">
    <attribute name="assetuid" unique-id="true" id="jw3q2"/>
    <attribute name="assetnum" searchable="true" id="j927r"/>
    <attribute name="parent" id="custAssetParentAttr"/>
    <attribute name="changeby" id="custassetchangeby"/>
    <attribute name="location.location" id="wnb57"/>
    <attribute name="location.description" id="pwwnk"/>
    <attribute name="assettype" searchable="true" id="qry56"/>
    <attribute name="siteid" searchable="true" id="gqejr"/>
    <attribute name="age" id="rlnb05"/>
    ...
  </schema>
</maximo-datasource>
```

```

</maximo-datasource>
<maximo-datasource id="assetDetailResource" default="true" depends-on="{page
ids}" object-structure="mxapiasset" controller="AssetDetailResourceController" no
<schema id="ywr9_">
  <attribute name="assetuid" unique-id="true" id="jw3q2"/>
  <attribute name="assetnum" searchable="true" id="j927r"/>
  <attribute name="parent" id="custAssetParentAttr"/>
  <attribute name="changeby" id="custassetchangeby"/>
  <attribute name="location.location" id="wnb57"/>
  <attribute name="location.description" id="pwwnk"/>
  <attribute name="assettype" searchable="true" id="qry56"/>
  <attribute name="siteid" searchable="true" id="gqejr"/>
  <attribute name="age" id="rlnb05"/>
  <attribute name="description" searchable="true" id="rp979"/>
  <attribute name="status" searchable="true" id="xz9j7"/>
  <attribute name="installdate" id="xm_49"/>
  <attribute name="totalcost" id="ppn3k"/>

```

Add a Changed by field label.

```

<container id="nbpwn">
  <condition for="localCond" id="mqp7p">
    <set property="value"
value="{assetDetailResource.item.location.description}"
when="{assetDetailResource.item.location.description}" id="ya9wq"/>
    <set property="value"
value="{assetDetailResource.item.location.location}"
when="{!assetDetailResource.item.location.description}" id="bapav"/>
  </condition>
  <field label="Location" padding="none" icon-
label="carbon:location" label-placement="start"
value="{assetDetailResource.item.location.description}" id="localCond"/>
  <field label="Changed by" padding="none" label-
placement="start" value="{assetDetailResource.item.changeby}"
id="localChangedBy"/>
</container>

```

```

1449 <container id="nbpwn">
1450 <condition for="localCond" id="mqp7p">
1451 <set property="value" value="{assetDetailResource.item.location.description}" when="{assetDetailResource.
item.location.description}" id="ya9wq"/>
1452 <set property="value" value="{assetDetailResource.item.location.location}" when="{!assetDetailResource.
item.location.description}" id="bapav"/>
1453 </condition>
1454 <field label="Location" padding="none" icon-label="carbon:location" label-placement="start" value="
{assetDetailResource.item.location.description}" id="localCond"/>
1455 <field label="Changed by" padding="none" label-placement="start" value="{assetDetailResource.item.changeby}"
id="localChangedBy"/>
1456 </container>
1457 <container id="nbpdwn" hidden="{!page.state.assetFlags}" layout="horizontal" align-self="start" padding="none">
1458 <button on-click="showFlags" icon="carbon:flag" id="btnFlag"/>
1459 <label padding="none" label="{page.state.assetFlags}" id="exxx4md"/>
1460 </container>
1461 </container>
1462 </container>

```

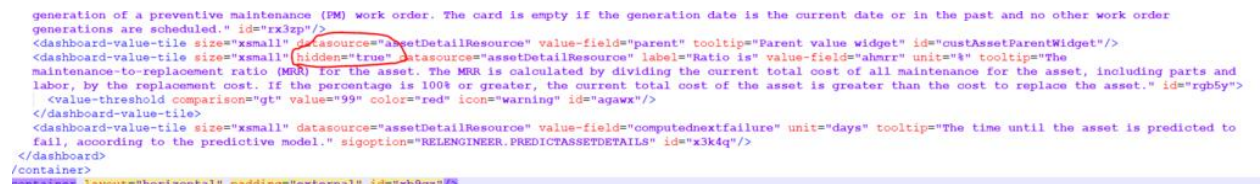
### Remove the MRR card

The MRR card is the Maintenance-to-replacement ratio card and it appears on pages for individual assets. If you do not want to capture cost information, you can hide the MRR card. This technique can also be used to hide the Predict card.

Locate each MMR card component, add the hidden attribute, and then set the hidden attribute to true.

```
<dashboard-value-tile size="xsmall" hidden="true"
datasource="assetDetailResource" label="Ratio is" value-field="ahmrr"
unit="%" tooltip="The maintenance-to-replacement ratio (MRR) for the
asset. The MRR is calculated by dividing the current total cost of all
maintenance for the asset, including parts and labor, by the replacement
cost. If the percentage is 100% or greater, the current total cost of the
asset is greater than the cost to replace the asset." id="rgb5y">

    <value-threshold comparison="gt" value="99" color="red"
icon="warning" id="agawx"/>
</dashboard-value-tile>
```



### Change the order of the top row of cards

There is a row of value cards at the top of individual asset pages. You can change the order of the cards to align with your business needs. For example, you might want to switch the order of the risk and criticality cards.

```
<dashboard-gauge-tile size="xsmall"
datasource="assetDetailResource" value-field="apmriskskiperror" units="%"
value-trend-field="apm_scorerisk.scoretrendvalue"
trend="{assetDetailResource.item.apm_scorerisk.scoretrenddirection === 1 ?
'up' : assetDetailResource.item.apm_scorerisk.scoretrenddirection === -1 ?
'down' : null}" color-
trend="{assetDetailResource.item.apm_scorerisk.scoretrenddirection === 1 ?
'red' : assetDetailResource.item.apm_scorerisk.scoretrenddirection === -1 ?
'green' : 'blue'}" tooltip="The probability of a high-impact failure.
Configure how this score is calculated on the Scoring page." bound-
datasource="{assetDetailResource}" bound-datasource-
item="{assetDetailResource.item}" bound-datasource-field="apmriskskiperror"
error-label="Score not calculated" error-
description="{assetDetailResource.item.apm_scorerisk.lsuccessdatetime
? 'The score can\\\'t be calculated. The last score was ' +
assetDetailResource.item.apm_scorerisk.apmnumval + ', and it was calculated
at ' + assetDetailResource.item.apm_scorerisk.lastsuccessstime + ' on ' +
assetDetailResource.item.apm_scorerisk.lastsuccessdate + '.'"
: '' }" error-tooltip-text="To try calculating the score
again, click Actions > Recalculate scores.
{assetDetailResource.item.apm_scorerisk.ahmethodology.msg}" id="mky35">
    <value-threshold-datasource datasource="riskahblthresholdrange"
lower-field="lower" upper-field="upper" color-field="color" label-
field="label" id="z9vwv"/>
```

```

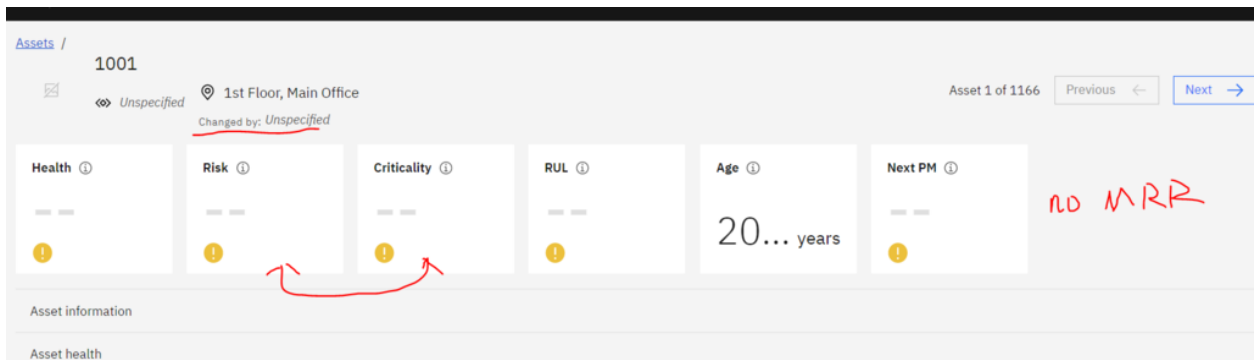
</dashboard-gauge-tile>
<dashboard-gauge-tile size="xsmall"
datasource="assetDetailResource" value-field="apmcriticalityiperror"
tooltip="The importance of the asset to business processes. Configure how
this score is calculated on the Scoring page." bound-
datasource="{assetDetailResource}" bound-datasource-
item="{assetDetailResource.item}" bound-datasource-
field="apmcriticalityiperror" error-label="Score calculation failed" error-
description="{assetDetailResource.item.apm_scorecriticality.lsuccessdatetime
? 'The score can\\\'t be calculated. The last score was ' +
assetDetailResource.item.apm_scorecriticality.apmnumval + ', and it was
calculated at ' +
assetDetailResource.item.apm_scorecriticality.lastsuccesstime + ' on ' +
assetDetailResource.item.apm_scorecriticality.lastsuccessdate + '.'
: '' }" error-tooltip-text="For more information, review the
Criticality details card. To try calculating the score again, click Actions >
Recalculate scores.
{assetDetailResource.item.apm_scorerisk.ahmethodology.msg}" id="z5xkr">
<value-threshold-datasource
datasource="criticalityahblthresholdrange" lower-field="lower" upper-
field="upper" color-field="color" label-field="label" id="p_x4v"/>
</dashboard-gauge-tile>

```

```

1493 </dashboard-gauge-tile>
1494 <dashboard-gauge-tile size="xsmall" datasource="assetDetailResource" value-field="apmriskskiperror" units=""
value-trend-field="apm_scorerisk.scoretrendvalue" trend="{assetDetailResource.item.apm_scorerisk.scoretrenddirection == 1 ?
'up' : assetDetailResource.item.apm_scorerisk.scoretrenddirection == -1 ? 'down' : null}" color-trends="{assetDetailResource.
item.apm_scorerisk.scoretrenddirection == 1 ? 'red' : assetDetailResource.item.apm_scorerisk.scoretrenddirection == -1 ?
'green' : 'blue'}" tooltip="The probability of a high-impact failure. Configure how this score is calculated on the Scoring page."
bound-datasource="{assetDetailResource}" bound-datasource-item="{assetDetailResource.item}"
bound-datasource-field="apmriskskiperror" error-label="Score not calculated" error-description="{assetDetailResource.item.
apm_scorerisk.lsuccessdatetime
? 'The score can\\\'t be calculated. The last score was ' + assetDetailResource.item.apm_scorerisk.apmnumval +
', and it was calculated at ' + assetDetailResource.item.apm_scorerisk.lastsuccesstime + ' on ' + assetDetailResource.item.
apm_scorerisk.lastsuccessdate + '.'
: '' }" error-tooltip-text="To try calculating the score again, click Actions > Recalculate scores.
{assetDetailResource.item.apm_scorerisk.ahmethodology.msg}" id="mky35">
1495 <value-threshold-datasource datasource="riskahblthresholdrange" lower-field="lower" upper-field="upper"
color-field="color" label-field="label" id="z9vww"/>
1496 </dashboard-gauge-tile>
1497 <dashboard-gauge-tile size="xsmall" datasource="assetDetailResource" value-field="apmcriticalityiperror"
tooltip="The importance of the asset to business processes. Configure how this score is calculated on the Scoring page."
bound-datasource="{assetDetailResource}" bound-datasource-item="{assetDetailResource.item}"
bound-datasource-field="apmcriticalityiperror" error-label="Score calculation failed" error-description="{assetDetailResource.
item.apm_scorecriticality.lsuccessdatetime
? 'The score can\\\'t be calculated. The last score was ' + assetDetailResource.item.apm_scorecriticality.

```







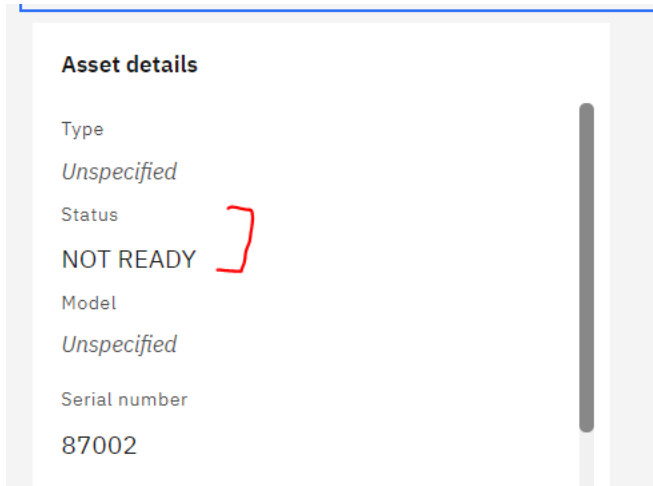
### Change the data showing in the asset detail card

Each asset detail card contains attributes for a specific asset, for example, the site or asset number.

You can change the details displayed by searching for the value `assetDetailResource` and then adding the new field. You can also search for the value `Asset details` and then adding the field to the dashboard-tile component.

```
id="er79k">
    <dashboard-tile title="Asset details" size="mediumthin"
    <container layout="vertical" id="bd5rz">
        <field label="Type"
value="{assetDetailResource.item.assettype}" header="true" padding="none"
id="rddde"/>
        <field label="Status"
value="{assetDetailResource.item.status}" header="true" padding="none"
id="custAssetDetailStatus"/>
        <field label="Model"
value="{assetDetailResource.item.pluscmodelnum}" header="true"
padding="bottom" id="ky35v"/>
        <field label="Serial number"
value="{assetDetailResource.item.serialnum}" header="true" padding="none"
id="g_eq7"/>
        <field label="Installation date"
value="{assetDetailResource.item.installdate}" header="true" padding="top"
id="eqgeq"/>
    </container>
</dashboard-tile>
```

```
<container layout="vertical" id="bd5rz">
  <field label="Type" value="{assetDetailResource.item.assettype}" header="true" padding="none" id="rddde"/>
  <field label="Status" value="{assetDetailResource.item.status}" header="true" padding="none"
id="custAssetDetailStatus"/>
  <field label="Model" value="{assetDetailResource.item.pluscmodelnum}" header="true" padding="bottom"
id="ky35v"/>
  <field label="Serial number" value="{assetDetailResource.item.serialnum}" header="true" padding="none"
id="g_eq7"/>
  <field label="Installation date" value="{assetDetailResource.item.installdate}" header="true" padding="top"
id="eqgeq"/>
</container>
```



### *Create a new card*

You can create a new card to display in asset pages.

Locate the assetDetailResource datasource and add new field.

```
<maximo-datasource id="assetDetailResource" default="true" depends-  
on="{page.params.linkedDS}" dependent-id="{page.params.ids}" object-structure="mxapiasset"  
controller="AssetDetailResourceController" notify-when-parent-loads="false">  
  <schema id="ywr9_">  
    <attribute name="assetuid" unique-id="true" id="jw3q2"/>  
    <attribute name="assetnum" searchable="true" id="j927r"/>  
    <attribute name="parent" id="custAssetParentAttr"/>  
    <attribute name="changeby" id="custassetchangeby"/>  
    <attribute name="location.location" id="wnb57"/>  
    <attribute name="location.description" id="pwwnk"/>  
    <attribute name="assettype" searchable="true" id="qry56"/>  
    <attribute name="siteid" searchable="true" id="gqejr"/>  
    <attribute name="age" id="rlnb05"/>  
    ...  
  </schema>  
</maximo-datasource>
```

```

    <attribute name="defaults" id="sydeCs0X4"/>
  </schema>
</maximo-datasource>
<maximo-datasource id="assetDetailResource" default="true" depends-on="{page.
ids}" object-structure="mxapiasset" controller="AssetDetailResourceController" noti
  <schema id="ywr9_">
    <attribute name="assetuid" unique-id="true" id="jw3q2"/>
    <attribute name="assetnum" searchable="true" id="j927r"/>
    <attribute name="parent" id="custAssetParentAttr"/>
    <attribute name="changedby" id="custassetchangedby"/>
    <attribute name="location.location" id="wnb57"/>
    <attribute name="location.description" id="pwwnk"/>
    <attribute name="assettype" searchable="true" id="qry56"/>
    <attribute name="siteid" searchable="true" id="gqejr"/>
    <attribute name="age" id="rlnb05"/>

```

Locate the value card section and add the new value card.

```

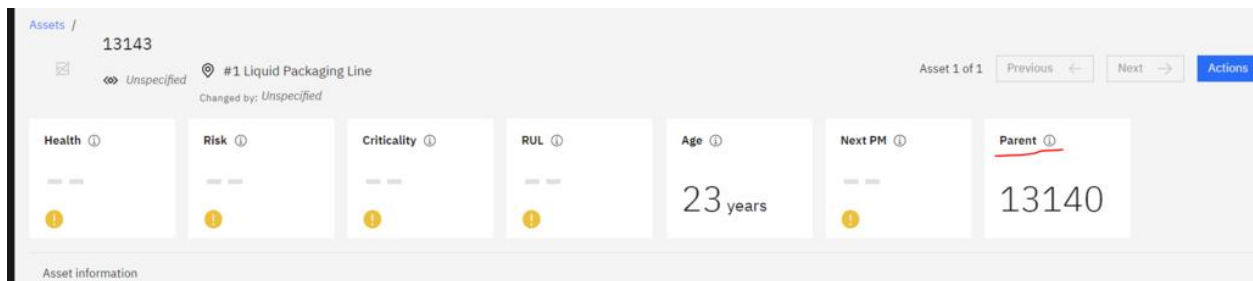
<dashboard-value-tile size="xsmall" datasource="assetDetailResource" value-
field="parent" tooltip="Parent value widget" id="custAssetParentWidget"/>

```

```

16 <!-- dashboard-value-tile size="xsmall" datasource="assetDetailResource" value-field="parent" tooltip="Parent value
17 <!-- dashboard-value-tile size="xsmall" datasource="assetDetailResource" label="Ratio is" value-field="ahmrr"
unit="%" tooltip="The maintenance-to-replacement ratio (MRR) for the asset. The MRR is calculated by dividing the current total
cost of all maintenance for the asset, including parts and labor, by the replacement cost. If the percentage is 100% or greater,

```



Create a new section and move a card to the new section

You can create a new section, for example, asset criticality, and then move the criticality detail widget to the new section.

```

    <accordion-item title="Asset criticality" open="true"
id="custAssetSection1">
    <dashboard id="custAssetSectionDashboard1">
...//the criticality detail widget

    </dashboard>
  </accordion-item>

```

```


1643 |         </dashboard-tile>
1644 |     </dashboard>
1645 | </accordion-item>
1646 | <accordion-item title="Asset criticality" open="true" id="custAssetSection1">
1647 |     <dashboard id="custAssetSectionDashboard1">
1648 |         <dashboard-tile title="Criticality details {assetDetailResource.item.apm_scorecriticality.apmmethod
' + assetDetailResource.item.apm_scorecriticality.lastcalctime + ' on ' + assetDetailResource.item.apm_scorecriticality
lastcaldate : ''}" size="medium" id="jnj6r1">...
1747 |     </dashboard-tile>
1748 | </dashboard>
1749 | </accordion-item>
1750 | <accordion-item title="Predictions" id="qave2" sigoption="RELENGINEER.PREDICTASSETDETAILS">
1751 |     <container layout="horizontal" id="xnvky">
1752 |         <dashboard id="k57">

```

Asset criticality

**Criticality details**

Scoring group: *Unspecified*



This asset doesn't have a criticality score yet, or the score was deactivated. Configure the

Asset timeline

Operational status

### Unsupported configurations

Here is a list of configuration updates that are not currently supported.

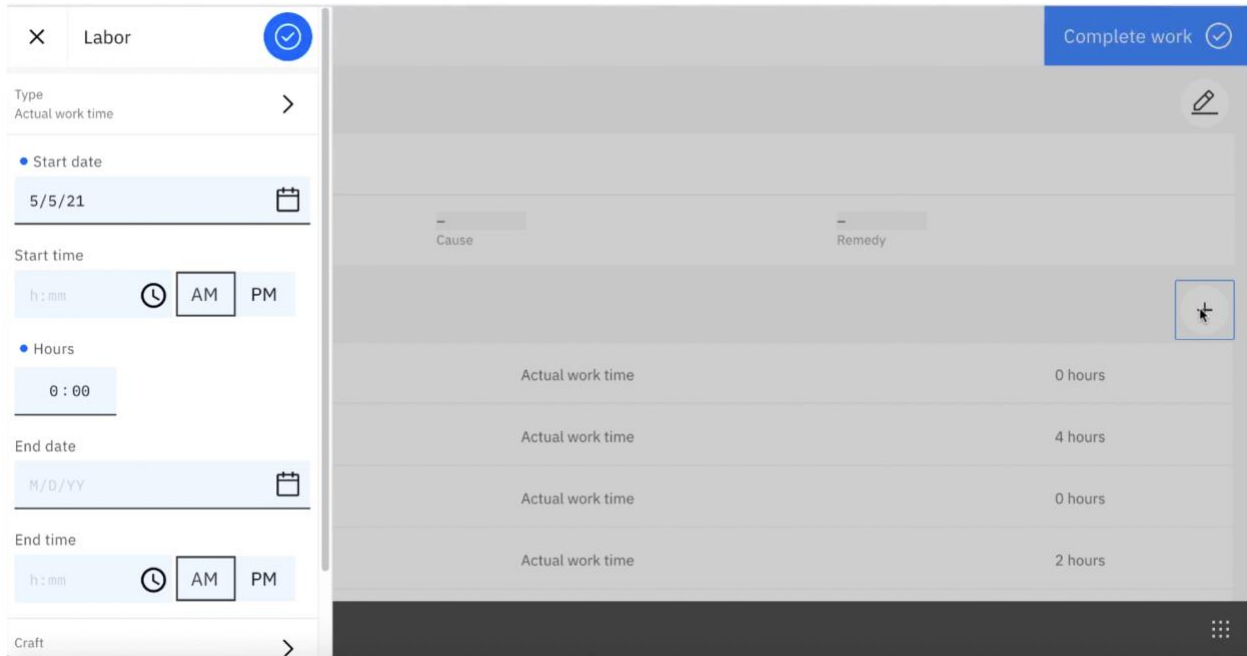
- Changing the name of a card.
- Adding an action to the Action menu.
- Updating tooltip text.
- Reusing a Predict card to display the output of a custom model.
- Changing the fields and dropdown menu values of a Predict card.
- Changing the chart name, axis values, and dropdown menu values of a Predict chart.
- Defining, adding, or removing lines from an Advanced line chart.

### Customizing applications

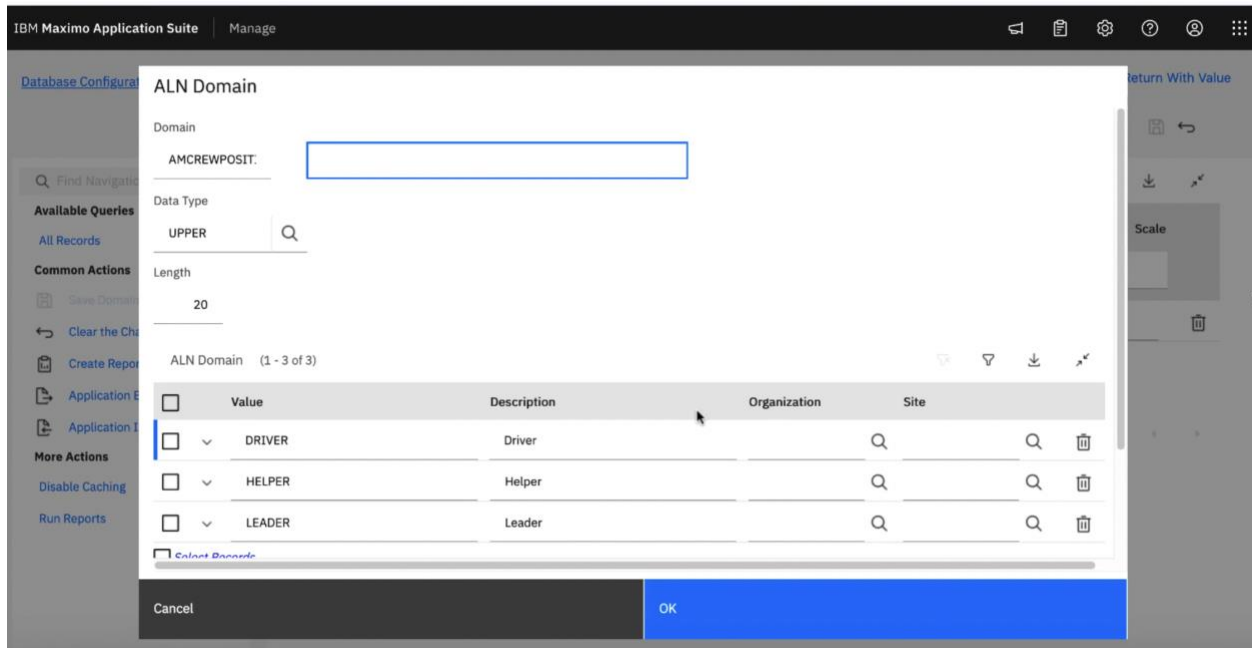
Customization refers to creating new JavaScript code to support a configuration change. A common customization is adding a new field to a card that is associated to a domain. In the

following example, a new field is added to the labor reporting page of the Technician application.

From the Report work page of a Work Order, a new crew position field is required to record additional Labor reporting data.



The crew position field is very similar to the `Type` field which reports the labor transaction type. We can use it as a model for the crew position field. The crew position field is associated with the `AMCREWPOSITION` ALN domain.



Returning to the Maximo Mobile Configuration interface, select the Technician application and open the **app.xml** file in the XML editor.

## Locating the Report work page

Search the file using the string `<page` and cycle through the results to locate the application page for reporting work.

```
1405 </border-layout>
1406 </page>
1407 <!-- Report work -->
1408 <page id="report_work" path="/report_work" title="Report work" icon="carbon:no-image" controller="ReportWorkPageController">
1409   <maximo-datasource id="woDetailsReportWork" default="true" object-structure="mxapiwodetail" item-url="{page.params.itemhref}" controller="ReportWorkDataController">
1410     <schema id="zp59q">
1411       <attribute name="workorderid" unique-id="true" id="rnxdv"/>
1412       <attribute name="wonom" id="qbbd8"/>
1413       <attribute name="problemcode" id="gpk5a"/>
1414       <attribute name="failurecode" id="er5rq"/>
1415       <attribute name="faildate" id="a536d"/>
1416       <attribute name="failure.description" id="qav2r"/>
1417       <attribute name="problem.description" id="ax797"/>
1418       <attribute name="failure.failurelist.failurelist" id="mjjjw"/>
1419       <attribute name="problem.failurelist.failurelist" id="wdm7g"/>
1420       <attribute name="problem.failurelist.failurecode" id="d42gz"/>
1421       <attribute name="failurereport{type, linenum, failurecode, failurecode, failurecode.description}" id="pb64d"/>
1422       <attribute name="siteid" id="ab9v6"/>
1423       <attribute name="orgid" id="dggp2"/>
1424       <attribute name="maxvar.labtranstolerance" id="mjyt"/>
1425       <attribute name="status" id="djzv"/>
1426       <attribute name="allowedstates" id="xyd34"/>
1427       <attribute name="remarkdesc" id="m79p2"/>
1428       <attribute name="failureclassdelete" type="B00L" default-value="false" id="j9mpp"/>
1429       <attribute name="problemdelete" type="B00L" default-value="false" id="e8467"/>
1430       <attribute name="causedelete" type="B00L" default-value="false" id="dqy4n"/>
1431       <attribute name="remedydelete" type="B00L" default-value="false" id="m6r_j"/>
1432     </schema>
1433     <maximo-datasource id="reportWorkActualMaterialDs" relationship="uxshowactualmaterial" depends-on="woDetailsReportWork" selection-mode="none"
controller="ReportWorkDataController">
1434       <schema id="rzp2y">
1435         <attribute name="matuasetransid" unique-id="true" id="a386_"/>
1436         <attribute name="siteid" id="n94v8"/>
1437         <attribute name="itemnum" id="egk5z"/>
1438       </schema>
1439     </maximo-datasource>
1440   </page>
```

The page with the `report_work` ID is where we create XML to support the crew position field.

## Creating a datasource

Create a datasource for the crew position field by first locating an ALN domain datasource in the XML.

```

292 <!-- depends-on="wodetails" -->
293 <schema id="deknp">
294 <attribute name="startdate" id="dym6j"/>
295 <attribute name="starttime" id="z4nae"/>
296 <attribute name="startdatetime" id="z3346"/>
297 <attribute name="finishdate" id="pym8j"/>
298 <attribute name="finishtime" id="wee94"/>
299 <attribute name="finishdatetime" id="b5483"/>
300 <attribute name="regularhrs" id="d34eb"/>
301 <attribute name="transtype" id="wme9p"/>
302 <attribute name="labtransid" unique-id="true" id="e3_92"/>
303 <attribute name="timerstatus" searchable="true" id="k3ba4"/>
304 <attribute name="laborcode" searchable="true" id="xj8ve"/>
305 <attribute name="anywherefid" id="a7je9"/>
306 </schema>
307 </maximo-datasource>
308 <maximo-datasource id="dsnewreading" lookup-data="true" object-structure="mxapiaIndomain" page-size="100" selection-mode="single">
309 <schema id="qkzae">
310 <attribute name="value" unique-id="true" id="gv9ge"/>
311 <attribute name="valueid" id="gx955"/>
312 <attribute name="description" id="bkqzb"/>
313 <attribute name="domainid" searchable="true" id="v64yv"/>
314 <attribute name="siteid" id="jgbv9"/>
315 <attribute name="orgid" id="p6_8v"/>
316 </schema>
317 </maximo-datasource>
318 <json-datasource id="dsstatusDomainList" src="{[]}" items="statusItems" selection-mode="single" id-attribute="value">
319 <schema id="dpyn7">
320 <attribute name="value" id="pqyzz"/>
321 <attribute name="description" id="k8k8k"/>
322 </schema>
323 </json-datasource>
324 <header-template hide-breadcrumb="true" title-column-width="{page.state.breadcrumbWidth ? page.state.breadcrumbWidth : app.state.screen.size === 'md' ? 50 : app.

```

Copy the dsnewreading datasource and paste it within the Report Work page section of the file. Update the datasource attributes for the crew position field.

```

<maximo-datasource id="dsposition" lookup-data="true" object-
structure="mxapiaIndomain" where="domainid='&quot;AMCREWPOSITION&quot;'" page-
size="100" selection-mode="single">
  <schema id="">
    <attribute name="value" unique-id="true" id=""/>
    <attribute name="valueid" id=""/>
    <attribute name="description" id=""/>
    <attribute name="domainid" searchable="true" id=""/>
    <attribute name="siteid" id=""/>
    <attribute name="orgid" id=""/>
  </schema>
</maximo-datasource>

```

The *where* clause condition specifies that we want a field that is associated with the AMCREWPOSITION domain. With the exception of the maximo-datasource ID, all other ID values should be left empty. Unique ID values are generated when the application is rebuilt.

## Creating a lookup

Lookups reference attribute information from the database associated with domain fields. Create a lookup for the crew position field by first locating the lookup associated with the Type field. Search the XML for the string `transtype` and cycle through the results to locate the Type lookup.

```

1548 {page.state.causeValue}" id="mjjs9"/>
1549 </box>
1550 </adaptive-column>
1551 <adaptive-column small-width="100" medium-width="100" large-width="33" id="p3bw6">
1552 <box padding="0,5" children-hide-overflow="true" padding-start="{app.state.screen.size == 'sm' || app.state.screen.size == 'md' ? '1' : '5'}"
vertical-align="center" fill-parent="true" id="mvg47">
1553 <field label="Remedy" padding="{app.state.screen.size == 'sm' || app.state.screen.size == 'md' ? 'none' : 'default'}" label-placement="top" value="
{page.state.remedyValue}" id="p558g"/>
1554 </box>
1555 </adaptive-column>
1556 </adaptive-row>
1557 </middle>
1558 </border-layout>
1559 </middle>
1560 </border-layout>
1561 </panel>
1562 <dialogs id="wvzvx">
1563 <lookup id="transTypeLookup" show-search="false" lookup-heading="Type" datasource="reportworksSynonymData" on-item-click="chooseTransType" border="true"
lookup-attributes="{['value', 'description']}" height="400" width="60"/>
1564 <lookup id="craftLookup" show-search="false" lookup-heading="Craft" datasource="crafrate" border="true" on-item-click="selectCraftSkill" lookup-attributes="{
['craftdescription', 'skilllevelscdata']}" height="400" width="60"/>
1565 <sliding-drawer id="reportTimeDrawer" align="start" header-text="Labor" content-padding="false" controller="ReportWorkPageController">
1566 <panel id="d8jkr">
1567 <border-layout fill-parent="true" padding="false" id="b49b4">
1568 <start width="100" background-color="ui-01" id="qb2yp">
1569 <box padding-bottom="0,25" children-sizes="100" direction="column" fill-child="true" fill-parent="true" id="ye3yb">
1570 <border-layout fill-parent="true" padding="true" middle-border="true" id="j2rxw">
1571 <start width="90" background-color="ui-01" id="qa2er">
1572 <box children-sizes="100" padding-top="0,25" padding-bottom="0,25" fill-child="true" fill-parent="true" id="x6gy5">
1573 <field label="Type" field-class-name="14-regular" swap-position="false" value="{reportworkLaborDetailds.item.transType_description}" id="wdd_v"/>
1574 </box>
1575 </start>
1576 <end width="10" background-color="ui-01" id="kxa47">
<button icon="carbon:chevron--right" on-click="openTransTypeLookup" on-click-arg="{{'page':page,'app':app}}" kind="ghost" padding="false" id="nj99a"/>

```

Copy the `transTypeLookup` lookup and paste it into the file. Update the lookup attributes for the crew position field.

```

<lookup id="positionLookup" show-search="false" lookup-heading="Type"
datasource="dsposition" on-item-click="choosePosition" border="true" lookup-
attributes="{['value', 'description']}" height="400" width="60"/>

```

The `datasource` value has been updated to use the `datasource` that was created. The `on-item-click` value was updated to use a new JavaScript function called `choosePosition`.

## Setting the display of the field

Fields are positioned on application pages using the `box` XML element. Within the `transTypeLookup` element, copy the `box` XML used to position the `Type` field. Paste the XML code below the one you report just copied and then update the attribute values for the new field.



```

1554     </adaptive-column>
1555   </adaptive-row>
1556 </middle>
1557 </border-layout>
1558 </middle>
1559 </border-layout>
1560 </panel>
1561 <dialogs id="wvzx">
1562   <lookup id="transTypeLookup" show-search="false" lookup-heading="Type" datasource="reportworksSynonymData" on-item-click="chooseTransType" border="true"
lookup-attributes="{['value', 'description']}" height="400" width="60"/>
1563   <lookup id="craftLookup" show-search="false" lookup-heading="Craft" datasource="crafrate" border="true" on-item-click="selectCraftSkill" lookup-attributes="{
['craftdescription', 'skilllevelscdata']}" height="400" width="60"/>
1564   <sliding-drawer id="reportTimeDrawer" align="start" header-text="Labor" content-padding="false" controller="ReportWorkPageController">
1565     <panel id="d8jkr">
1566       <border-layout fill-parent="true" padding="false" id="b49b4">
1567         <start width="100" background-color="ui-01" id="gb2yp">
1568           <box padding-bottom="0.25" children-sizes="100" direction="column" fill-child="true" fill-parent="true" id="ye3yb">
1569             <border-layout fill-parent="true" padding="true" middle-border="true" id="j2rxw">
1570               <start width="90" background-color="ui-01" id="qa2er">
1571                 <box children-sizes="100" padding-top=".25" padding-bottom=".25" fill-child="true" fill-parent="true" id="x6gy5">
1572                   <field label="Type" field-class-name="14-regular" swap-position="false" value="{reportworkLaborDetailds.item.transype_description}" id="wdd_v"/>
1573                 </box>
1574               </start>
1575             <end width="10" background-color="ui-01" id="kxa47">
1576               <button icon="carbon:chevron--right" on-click="openTransTypeLookup" on-click-arg="{{'page':page,'app':app}}" kind="ghost" padding="false" id="nj99a"/>
1577             </end>
1578           </border-layout>
1579         </box>
1580       <box children-sizes="100" padding-top=".5" direction="column" fill-child="true" fill-parent="true" id="z9er7">
1581         <smart-input value="{reportworkLaborDetailds.item.startdate}" id="startDate" required="true" label="Start date"/>
1582       </box>
1583       <box children-sizes="100" padding-top=".5" fill-child="true" fill-parent="true" id="prq6g">
1584         <smart-input value="{reportworkLaborDetailds.item.starttime}" placeholder="0:00" id="startTime" label="Start time"/>

```

```

<box padding-bottom="0.25" children-sizes="100" direction="column" fill-
child="true" fill-parent="true" id="">
  <border-layout fill-parent="true" padding="true" middle-
border="true" id="">
    <start width="90" background-color="ui-01" id="">
      <box children-sizes="100" padding-top=".25" padding-
bottom=".25" fill-child="true" fill-parent="true" id="">
        <field label="Position" field-class-name="14-regular"
swap-position="false" value="{reportworkLaborDetailds.item.position}" id=""/>
      </box>
    </start>
    <end width="10" background-color="ui-01" id="">
      <button icon="carbon:chevron--right" on-
click="openPositionLookup" on-click-arg="{{'page':page,'app':app}}"
kind="ghost" padding="false" id=""/>
    </end>
  </border-layout>
</box>

```

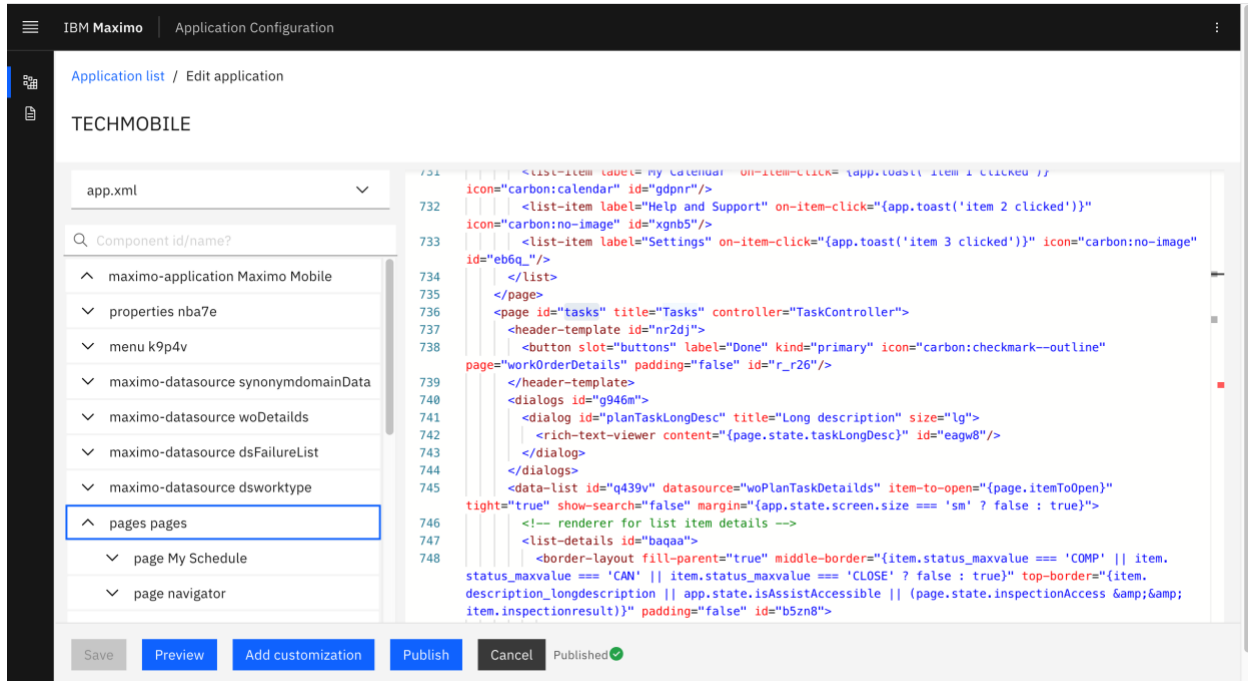
All ID values have been removed. The field label was changed to **Position** and the `reportworkLaborDetailds.item.position` attribute value was changed to retrieve the position field from the domain. The on-click event was changed to initiate the **openPositionLookup** function.

### Saving XML configuration changes

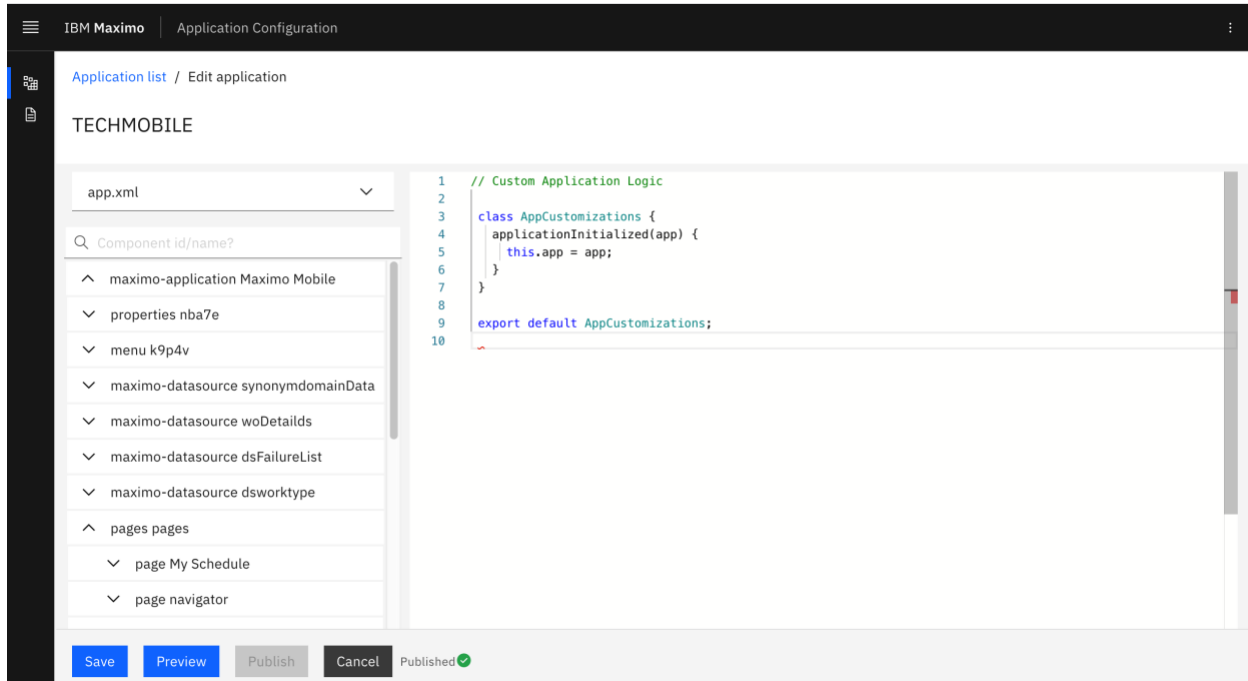
Save the changes that you have made in the **app.xml** file. The next step is to create custom JavaScript to initiate the retrieval of information from the database.

## Creating custom code

Select the Add customization button to enable the use of custom JavaScript code.



Clicking Add customization opens the **AppCustomizations.js** file for editing. This file is where all customized JavaScript is kept.



To add the crew position field to the **app.xml** file, we located an existing field to model it after. We are going to use the same approach for the JavaScript functions we need.

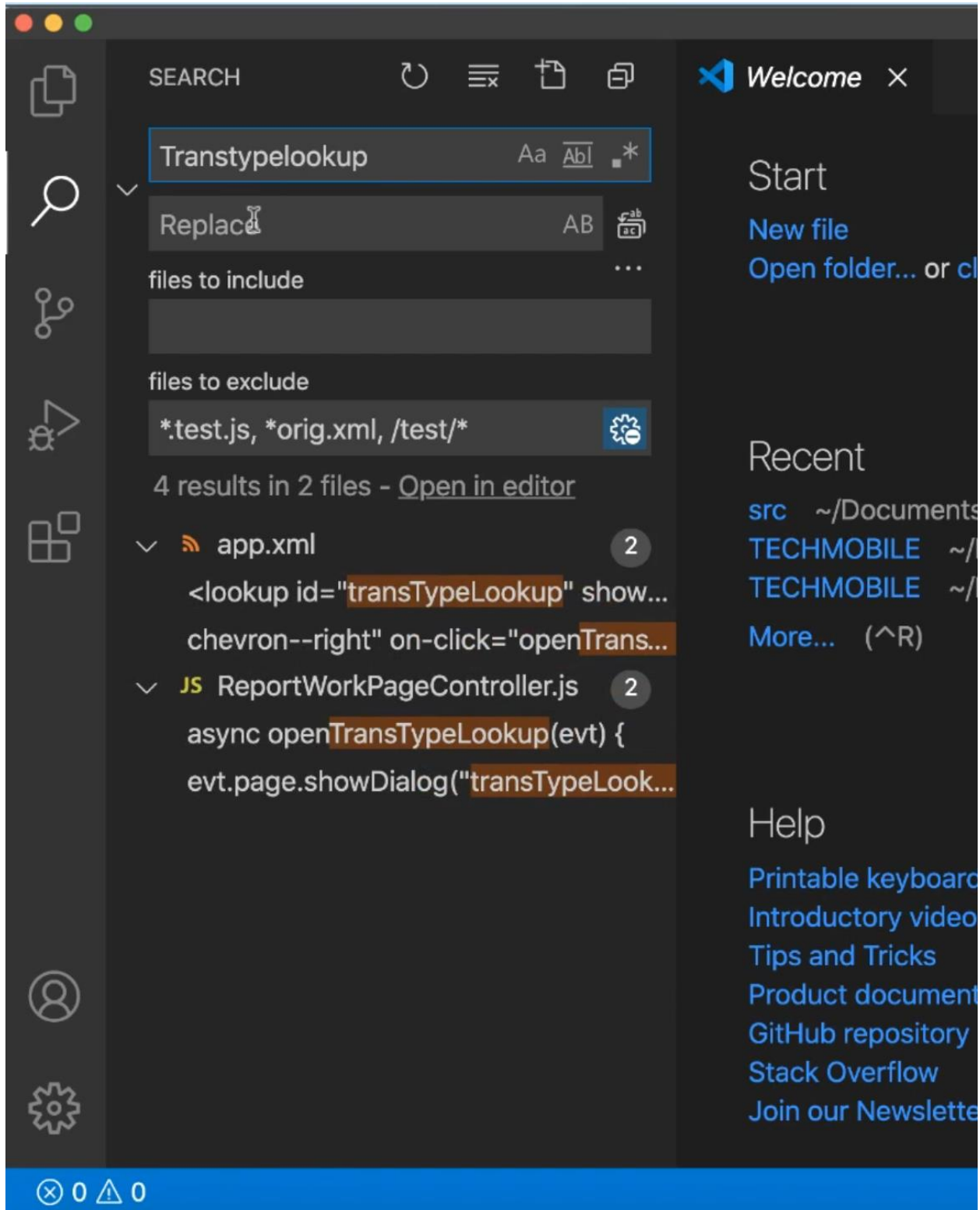
When you edit an application within Maximo Mobile Configuration, a copy of the application files is saved to your local system. The files are in the working directory that you specified in the Docker run command.

Name	Date Modified	Size	Kind
TECHMOBILE	Today at 10:37 AM	--	Folder
build	Today at 10:37 AM	--	Folder
CHANGELOG.md	Today at 10:37 AM	1 KB	text document
package.json	Today at 10:37 AM	2 KB	text document
preview.log	Today at 10:37 AM	22 KB	Log File
README.md	Today at 10:37 AM	3 KB	text document
src	Today at 10:37 AM	--	Folder
app.orig.orig.xml	Today at 10:37 AM	156 KB	XML Document
app.orig.xml	Today at 10:37 AM	156 KB	XML Document
app.xml	Today at 10:37 AM	156 KB	XML Document
AppController.js	Today at 10:37 AM	11 KB	JSON File
AppController.test.js	Today at 10:37 AM	12 KB	JSON File
assets	Today at 10:37 AM	--	Folder
AssetWoController.js	Today at 10:37 AM	2 KB	JSON File
AssetWoController.test.js	Today at 10:37 AM	3 KB	JSON File
AttachmentController.js	Today at 10:37 AM	909 bytes	JSON File
AttachmentController.test.js	Today at 10:37 AM	2 KB	JSON File
ChangeStatusController.js	Today at 10:37 AM	5 KB	JSON File
ChangeStatusController.test.js	Today at 10:37 AM	2 KB	JSON File
FailureDetailsPageController.js	Today at 10:37 AM	28 KB	JSON File
FailureDetailsPageController.test.js	Today at 10:37 AM	21 KB	JSON File
MapPageController.js	Today at 10:37 AM	482 bytes	JSON File
MapPageController.test.js	Today at 10:37 AM	651 bytes	JSON File
MaterialsPageController.js	Today at 10:37 AM	4 KB	JSON File
MaterialsPageController.test.js	Today at 10:37 AM	5 KB	JSON File
mocked	Today at 10:37 AM	--	Folder
RelatedWoController.js	Today at 10:37 AM	2 KB	JSON File
RelatedWoController.test.js	Today at 10:37 AM	3 KB	JSON File
ReportWorkDataController.js	Today at 10:37 AM	970 bytes	JSON File
ReportWorkDataController.test.js	Today at 10:37 AM	2 KB	JSON File
ReportWorkPageController.js	Today at 10:37 AM	37 KB	JSON File
ReportWorkPageController.test.js	Today at 10:37 AM	35 KB	JSON File

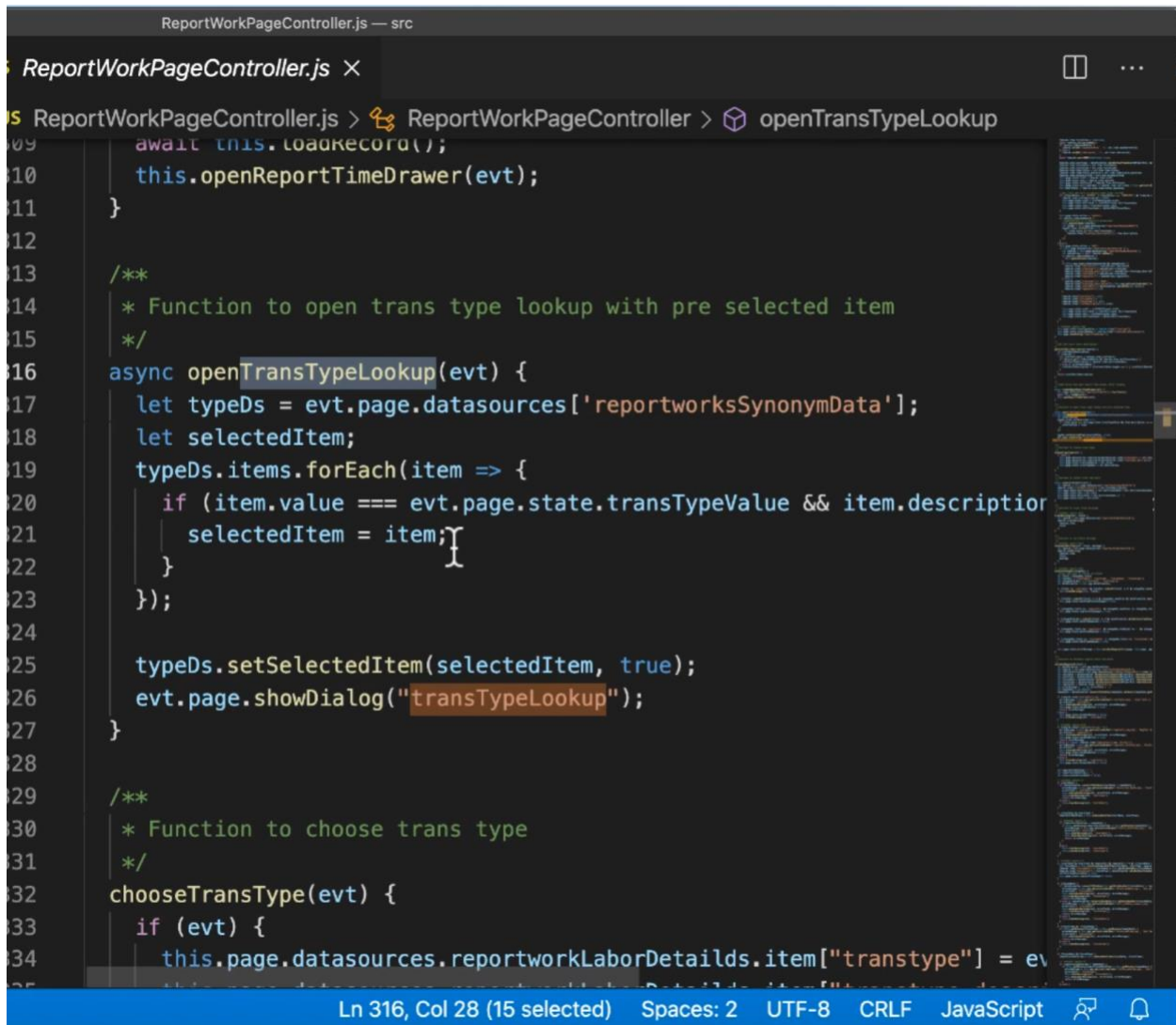
Because the crew position field is modeled after the Type field, copy the JavaScript code that supports the Type field and paste it into the **AppCustomizations.js** file. Now you can customize the JavaScript code to support the new crew position field.

The first bit of JavaScript you need to create is related to the openPositionLookup onclick event. This onclick event enables the crew position field chevron to open a list of domain values for the position.

Open an editor and search the application JavaScript files for the lookup function related to the Type field.



Opening the **ReportWorkPageController.js** file brings us to the code for the **openTransTypeLookup** function.



```
ReportWorkPageController.js — src
ReportWorkPageController.js X
ReportWorkPageController.js > ReportWorkPageController > openTransTypeLookup
109   await this.loadRecord();
110   this.openReportTimeDrawer(evt);
111 }
112
113 /**
114  * Function to open trans type lookup with pre selected item
115  */
116 async openTransTypeLookup(evt) {
117   let typeDs = evt.page.datasources['reportworksSynonymData'];
118   let selectedItem;
119   typeDs.items.forEach(item => {
120     if (item.value === evt.page.state.transTypeValue && item.description
121         selectedItem = item;
122   });
123 });
124
125 typeDs.setSelectedItem(selectedItem, true);
126 evt.page.showDialog("transTypeLookup");
127 }
128
129 /**
130  * Function to choose trans type
131  */
132 chooseTransType(evt) {
133   if (evt) {
134     this.page.datasources.reportworkLaborDetails.item["transtype"] = ev
135   }
136 }
```

Copy and paste the **openTransTypeLookup** function code into the **AppCustomizations.js** file and modify it for the **openPositionLookup** function.

```
// Add a LookUp to display domain values (cloned
// openTranstypeLookup)
async openPositionLookup(evt) {
  let typeDs = evt.page.datasources['dsposition'];
  let selectedItem;
  evt.page.showDialog("positionLookup");
}
```

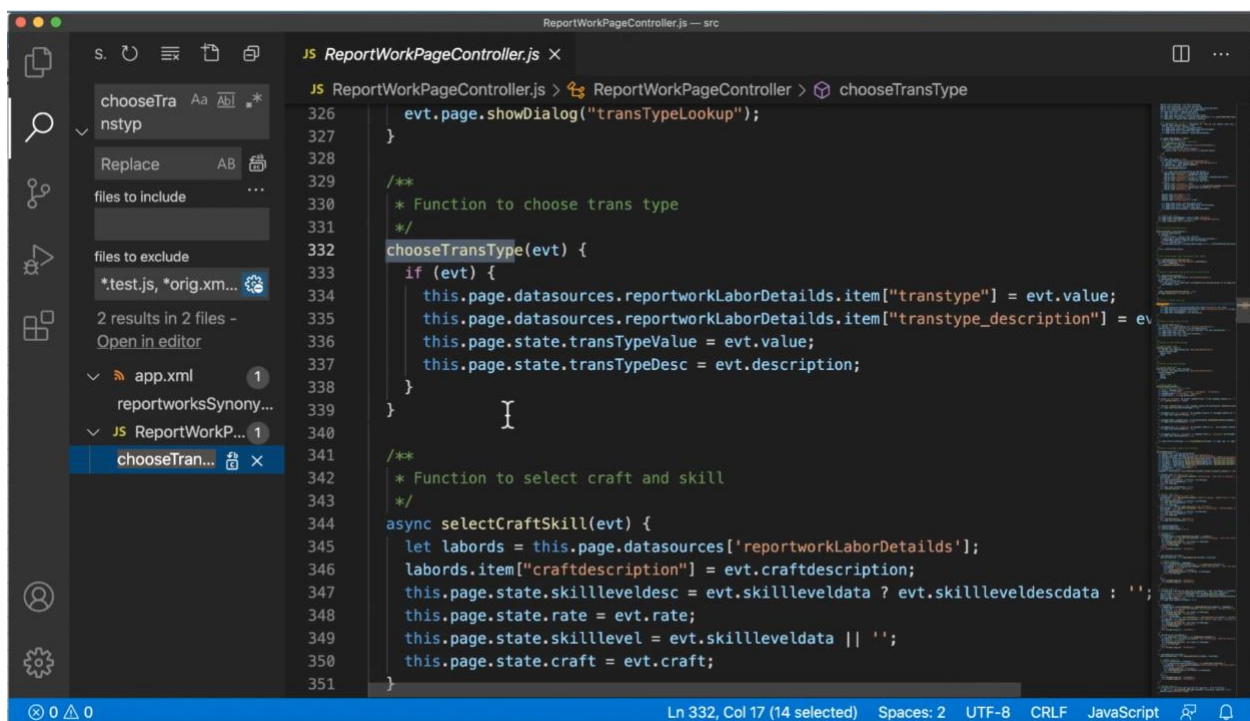
Several lines of code required when working with synonym domains were removed because the crew position field works with an ALN domain.

Next, we add JavaScript code that helps provide a reference to the page and application that we are customizing to the other functions.

```
// Capture a reference to the Page
// new method pageResumed called every time Page accessed
pageResumed(page, app) {
  this.app = app;
  this.page = page;
}
```

Next we add code that enables us to select the position type.

Search for the chooseTransType function.

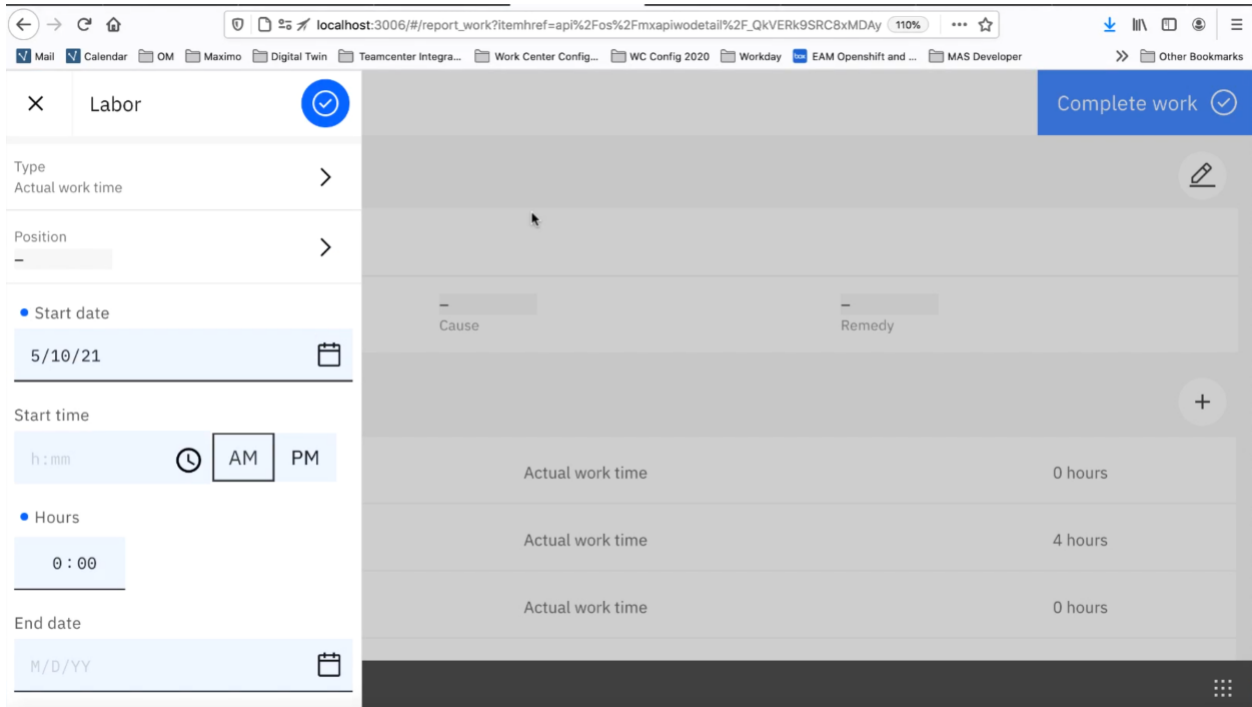
A screenshot of a code editor window titled 'ReportWorkPageController.js'. The editor shows a search for 'chooseTransType' in the file. The search results show the function definition starting at line 332. The function is named 'chooseTransType' and takes an event object 'evt' as a parameter. It contains an 'if' statement that checks if 'evt' is truthy. If true, it sets 'this.page.datasources.reportworkLaborDetailds.item["transtype"]' to 'evt.value', 'this.page.datasources.reportworkLaborDetailds.item["transtype\_description"]' to 'evt.description', 'this.page.state.transTypeValue' to 'evt.value', and 'this.page.state.transTypeDesc' to 'evt.description'. Below this function, there is another function 'async selectCraftSkill' which sets various state properties based on 'evt' parameters. The status bar at the bottom indicates 'Ln 332, Col 17 (14 selected) Spaces: 2 UTF-8 CRLF JavaScript'.

Copy and paste the chooseTransType function code into the **AppCustomizations.js** file and modify it for the choosePosition function.

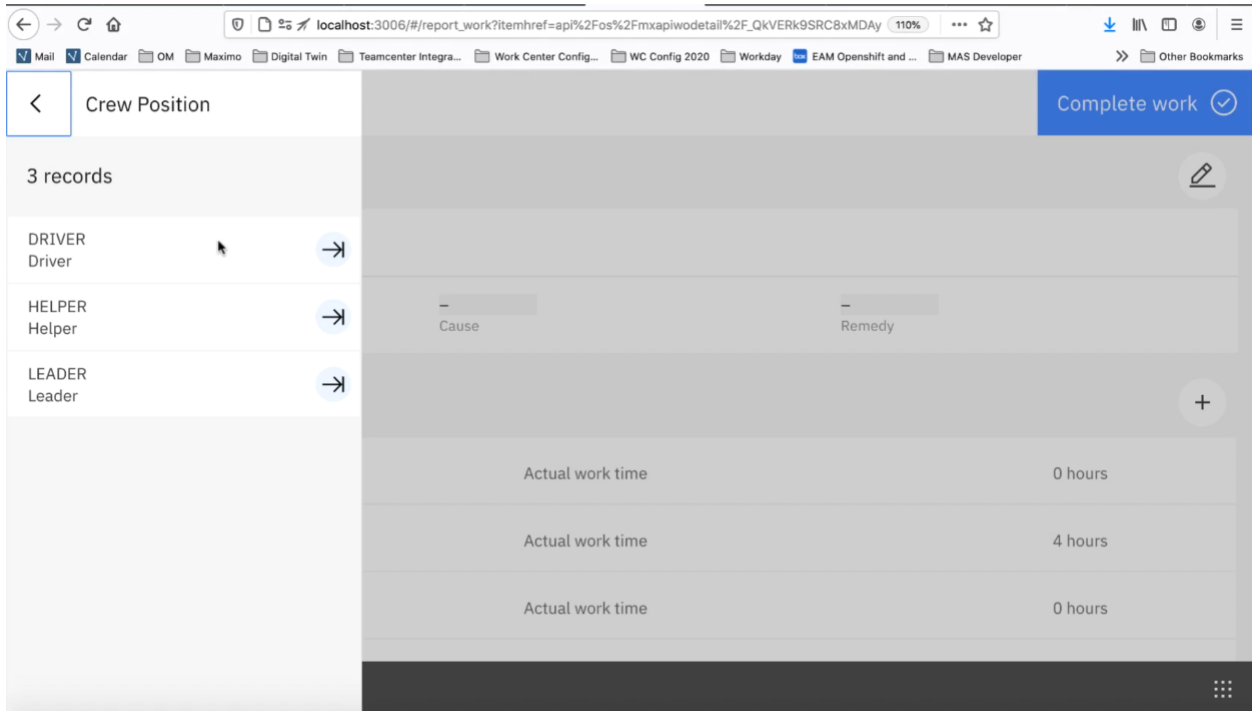
```
// Display & setting of Position field after Transaction Type on Labor
// Reporting slide out
choosePosition(evt) {
  if (evt) {
    this.page.datasources.reportworkLaborDetailds.item["position"]
    = evt.value;
    this.page.datasources.reportworkLaborDetailds.item["position_d
escription"] = evt.description;
    this.page.state.positionValue = evt.value;
  }
}
```

```
this.page.state.positionDesc = evt.description;
}
}
```

Click the Preview button to see how the field appears in the application.

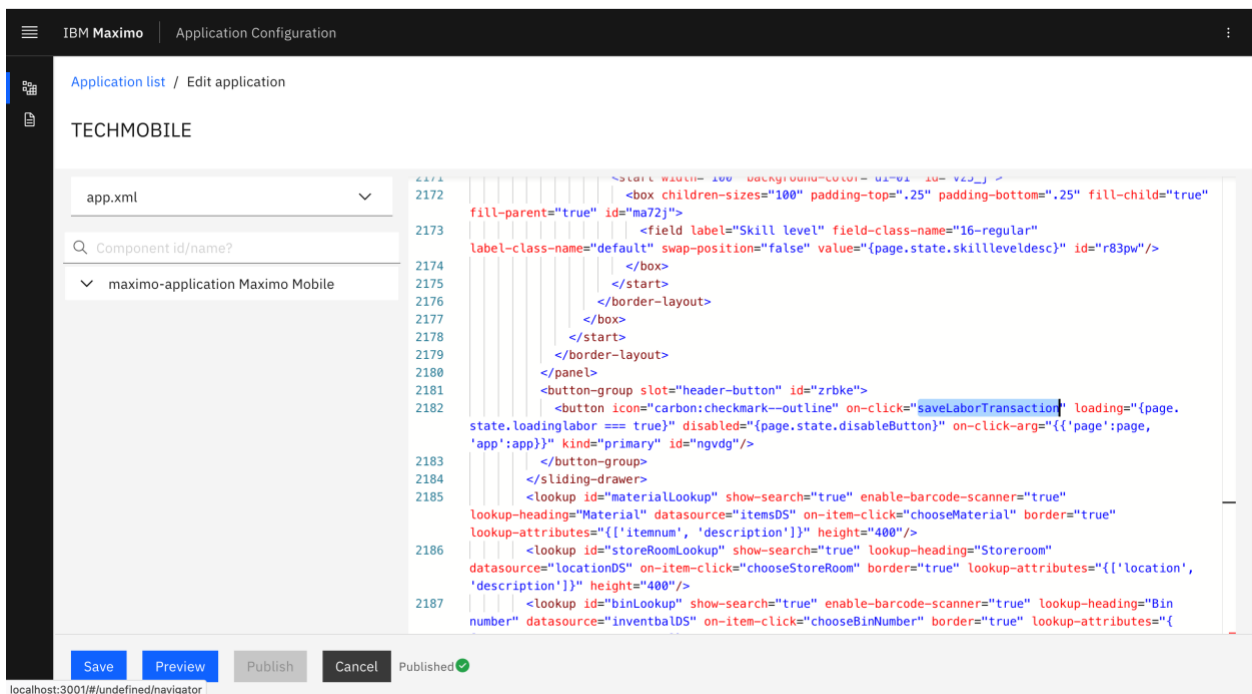


Click the Position field chevron to see the domain lookup values for this field.



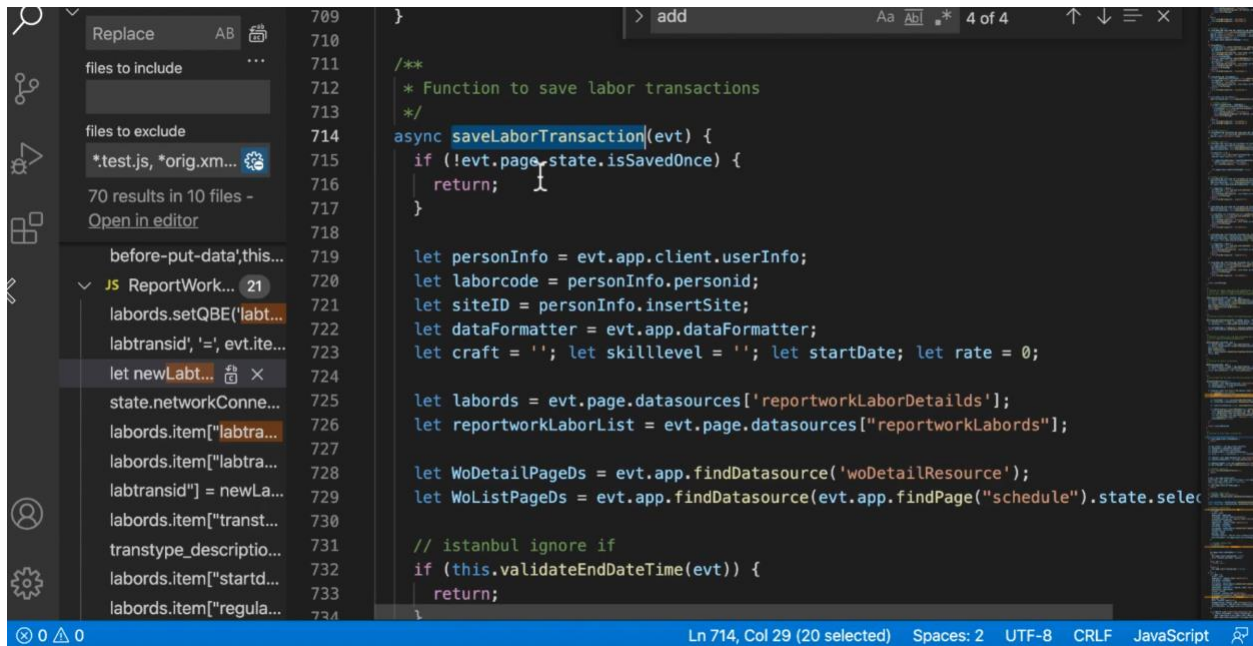
Include code to save Position field data to the labor transaction record. Labor data is saved when you select the blue icon with the checkmark symbol.

Save the updates made to the **AppCustomization.js** file and edit the **app.xml** file. We want to find the XML entry for the save icon.



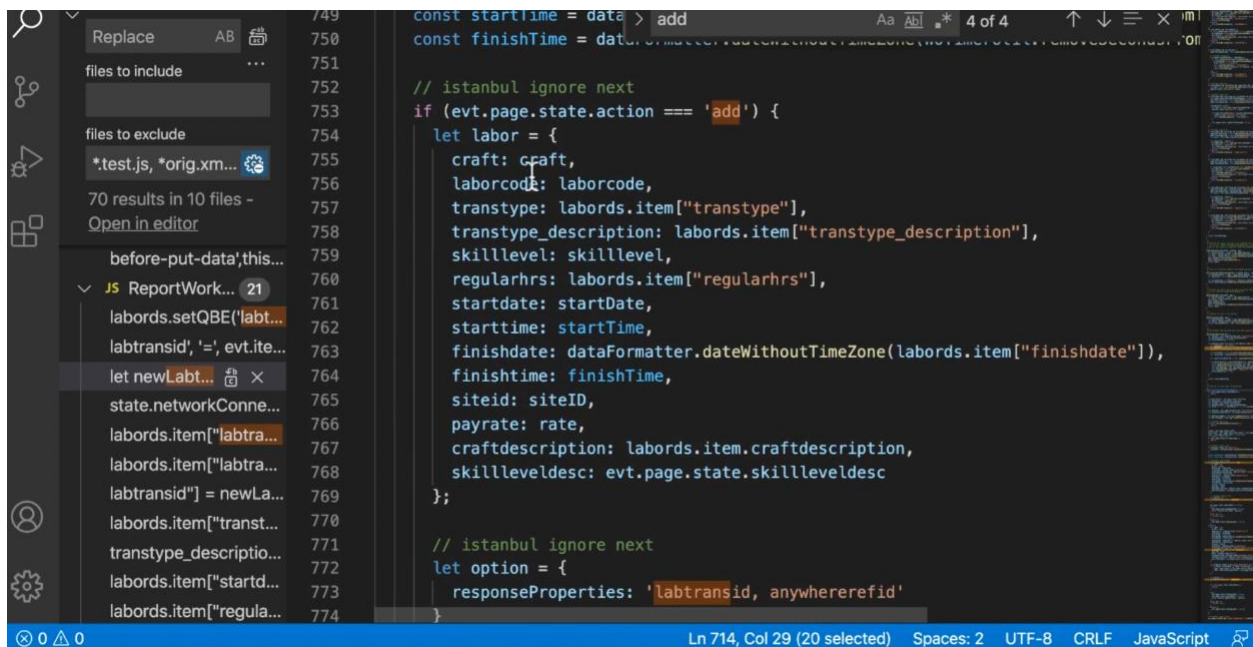


The onclick event is tied to the saveLaborTransaction function. Search the application JavaScript files for that function.



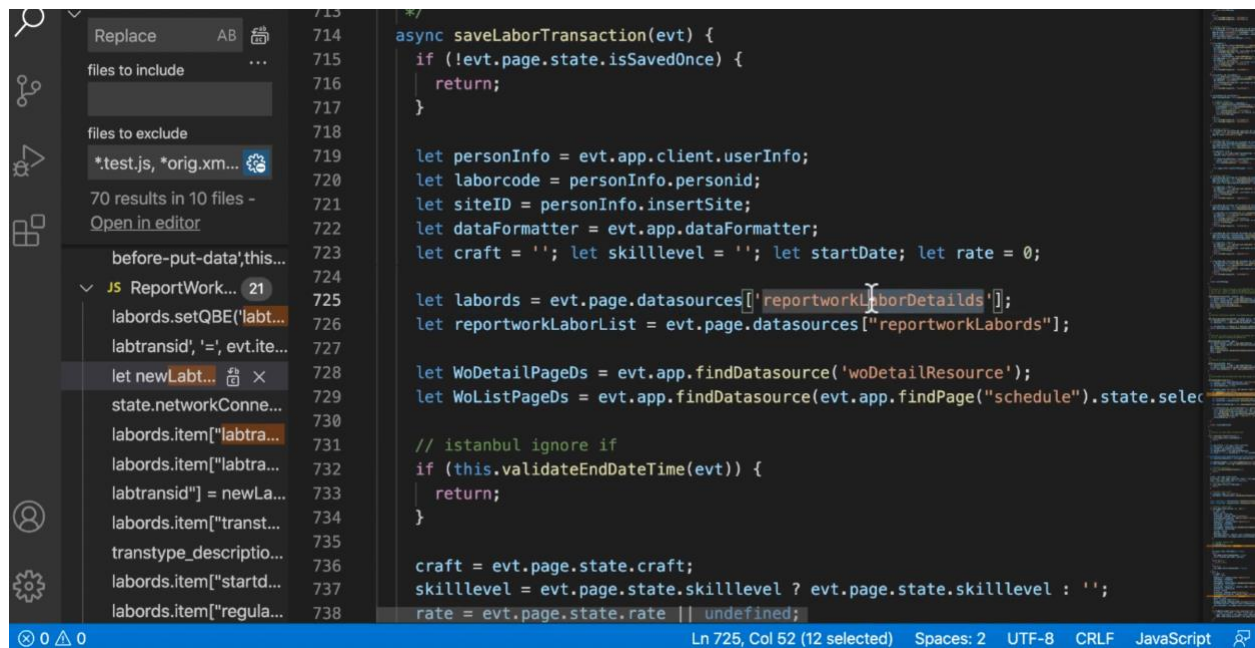
```
709 }
710
711
712 /**
713  * Function to save labor transactions
714  */
715 async saveLaborTransaction(evt) {
716   if (!evt.page.state.isSavedOnce) {
717     return;
718   }
719
720   let personInfo = evt.app.client.userInfo;
721   let laborcode = personInfo.personid;
722   let siteID = personInfo.insertSite;
723   let dataFormatter = evt.app.dataFormatter;
724   let craft = ''; let skilllevel = ''; let startDate; let rate = 0;
725
726   let labords = evt.page.datasources['reportworkLaborDetails'];
727   let reportworkLaborList = evt.page.datasources["reportworkLabords"];
728
729   let WoDetailPageDs = evt.app.findDatasource('woDetailResource');
730   let WoListPageDs = evt.app.findDatasource(evt.app.findPage("schedule").state.select);
731
732   // istanbul ignore if
733   if (this.validateEndDateTime(evt)) {
734     return;
735   }
736 }
```

Scroll down the function code to view the list of fields that are saved. We need to include the Position field in that list without directly modifying the **ReportWorkPageController.js** file.



```
749 const startTime = data > add
750 const finishTime = data.format(new Date(new Date(startTime.getTime() + 60 * 60 * 1000).toLocaleDateString('en-US', {
751   year: 'numeric', month: 'numeric', day: 'numeric', hour: 'numeric', minute: 'numeric', second: 'numeric',
752   timeZone: 'UTC'
753 }))).getTime();
754
755 // istanbul ignore next
756 if (evt.page.state.action === 'add') {
757   let labor = {
758     craft: craft,
759     laborcode: laborcode,
760     transtype: labords.item["transtype"],
761     transtype_description: labords.item["transtype_description"],
762     skilllevel: skilllevel,
763     regularhrs: labords.item["regularhrs"],
764     startdate: startDate,
765     starttime: startTime,
766     finishdate: dataFormatter.dateWithoutTimeZone(labords.item["finishdate"]),
767     finishtime: finishTime,
768     siteid: siteID,
769     payrate: rate,
770     craftdescription: labords.item.craftdescription,
771     skillleveldesc: evt.page.state.skillleveldesc
772   };
773
774   // istanbul ignore next
775   let option = {
776     responseProperties: 'labtransid, anywhereferid'
777   }
778 }
```

The list of items to add includes a reference to a `labords` variable. Scrolling a bit further back in the function code, you see that `labords` is associated with the `reportworkLaborDetails` datasource.



```
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Replace AB
files to include
files to exclude
*.test.js, *orig.xml...
70 results in 10 files -
Open in editor
before-put-data,this...
JS ReportWork... 21
labords.setQBE('labt...
labtransid', '=', evt.ite...
let newLabt...
state.networkConne...
labords.item["labtra...
labords.item["labtra...
labtransid"] = newLa...
labords.item["transt...
transtype_descriptio...
labords.item["startd...
labords.item["regula...
```

Capture events occurring for this datasource to apply changes to this datasource before saving the data on the Maximo application server.

Returning to the Maximo Mobile Configuration editor, we create two new functions.

```
// new function to inject Position field on event of
// Labor transaction Save
onSaveLabtrans (evt) {
    evt.record.position=evt.datasource.currentItem.position;
}

// new function to capture events for the
// reportworkLaborDetails datasource and call the onSaveLabtrans function
onDatasourceInitialized(ds) {
    if (ds.name == "reportworkLaborDetails") {
        ds.on('before-put-data', this.onSaveLabtrans.bind(this));
    }
}
```

The `onDatasourceInitialized` function acts when the `reportworkLaborDetails` datasource initializes. Any time the datasource has an event before a put action, the `onSaveLabtrans` function is initialized.

The `onSaveLabtrans` function adds the Position field value that was selected to the datasource record when the `saveLaborTransaction` function in the **ReportWorkPageController.js** file initiates a put command.

```
763     finishdate: dataFormatter.dateWithoutTimeZone(labords.item["finishdate"]),
764     finishtime: finishTime,
765     siteid: siteID,
766     payrate: rate,
767     craftdescription: labords.item.craftdescription,
768     skillleveldesc: evt.page.state.skillleveldesc
769   };
770
771   // istanbul ignore next
772   let option = {
773     responseProperties: 'labtransid, anywherefid'
774   }
775
776   evt.page.state.isSavedOnce = false;
777   try {
778     evt.page.state.loadinglabor = true;
779     await labords.put(labor, option);
780   }
781   catch (err) {
782     //handle error
783   }
784   finally {
785     evt.page.state.loadinglabor = false;
786   }
787 } else {
```

The configuration and customization of the Position field is now complete. Use the Maximo Mobile Configuration Preview feature to verify that position data can be included in the Labor details for a record.

