Deltek

Deltek Touch 4.2 for Maconomy® 2.6.2

Layout Configuration Guide

October 27, 2023

This edition published October 2023.

Contents

Overview

The Deltek Touch for Maconomy Layout Configuration Guide describes tasks necessary for the customization of settings and layouts of the Deltek Touch for Maconomy 4.2 application using the RESTful web services. For example, the customers may want to add or remove fields to/from a screen.

Note: The official name of the application is *Deltek Touch for Maconomy*. This document only uses it at first mention. The succeeding instances of the application name display *Deltek Touch*. In addition, the application name in *Apple App Store* and *Google Play Store* displays *Deltek Touch for Maconomy*.

For technical information, such as pre-installation requirements and compatibility mode definition, see the *Deltek Touch for Maconomy Touch Admin Guide.*

How This Document is Organized

This document is organized so that you can go quickly to the section you need. Use Ctrl+F to easily find the term you are searching for or see the table below to find the section to help you.

Section	Description	Purpose of This Section
Customization	Includes an overview of the customization process and several customization tasks you can perform on Deltek Touch screens	▪ Details customization process using Extender ▪ Instructions on rewriting custom layouts and settings ▪ Provides steps and examples on how to customize fields, actions, and screens
Settings	Includes information on modifiable server-side settings	▪ Provides a list of parameters and descriptions, including valid and default values
Layouts	Includes information on layout configuration, layout files, components, and attributes	▪ Details customization capabilities and limitations ▪ Helps differentiate non-editable and editable layout components ▪ Details layout files and changes ▪ Syntax and descriptions

Audience

The guide is intended for extension consultants or power users who make settings and/or layout changes to Deltek Touch. This document explains only common configuration tasks and field attributes.

> **Note:** Once you have configured the system, you must log on again to Deltek Touch to ensure that all changes take effect. In addition, if you are not using Extender, you need to restart the Coupling Service.

Revision History

A summary of recent changes in this document is listed in the following table:

Update	Version	Description
Added Touch 4.2 changes	4.2	Updated several sections to include changes about the app version.
Added Touch 4.1 changes	4.1	Updated several sections to include changes about the app version and related enhancements
Settings Configuration	4.1	Added MileageAttachmentType and ShowDefaultMileageReceiptName, revised MaxAttachmentFileSize, and removed AnalyticksKey to support attaching documents to mileage sheets
Detach a Receipt	4.1	Included instructions for detaching receipts in mileage sheets
What's New	4.0	Provides overview of release updates
Removed MScript-related information	4.0	All MScript-related content were removed in relation to the MScript Touch API removal.
Customization Process Using Maconomy Extender	4.0	Provides updated customization process for .JSON files in Extender
Rewrite Customizations in REST	4.0	Provides instructions on how to redo the layout and setting customizations in Extender
Settings Configuration	4.0	Moved from the Deltek Touch for Maconomy Installation Guide and includes a list of customizable server-side settings
Guides & Resources	4.0	Updated the list of Maconomy documentation references
Restructured document and streamlined content	4.0	Outdated materials were removed from the document and overall content was restructured. See How This Document is Organized for details.

What's New

There are no changes to the server settings and the layouts.

CUSTOMIZATION

Customization Tasks

This chapter discusses the customization process and several customization tasks you can perform on Deltek Touch screens.

You can customize Deltek Touch by using Maconomy Extender 1.8.8 or later.

Using Extender for customizations is highly recommended since it is a user-friendly editor and allows easy deployment and source control for your customizations.

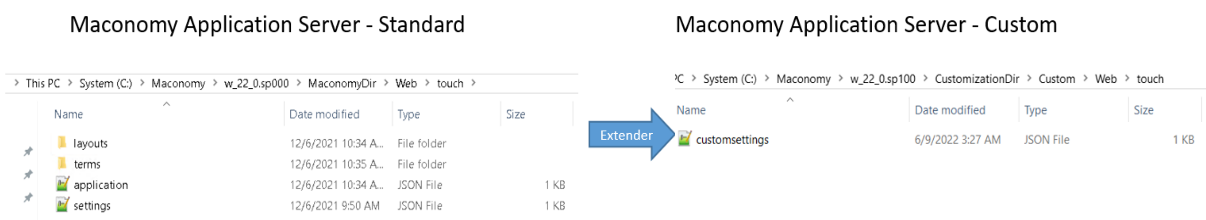
Customization Process Using Maconomy Extender

Maconomy Extender 1.8.8 (and later) supports Touch customizations, including the following items:

- **Layout Configuration**: This allows changes to screen layouts, such as adding, removing, and reordering fields on screens.
- **Server-Side Settings**: This allows changes to settings relevant for the business functionality in Deltek Touch, such as submit mode, week start, single sign-on, and custom submit timesheet action.

Starting with Touch 4.0 (Maconomy 2.6) all Touch layouts and settings are in .JSON format and can be customized similarly to the web client files. With Extender, you can customize Touch on all layers (Solution, Extension, Custom, and shortname specific Custom).

For example, if you customize on the Custom layer, your files will be deployed on the Maconomy server in the …/CustomizationDir/Custom/Web/touch folder, as shown in the following figure:

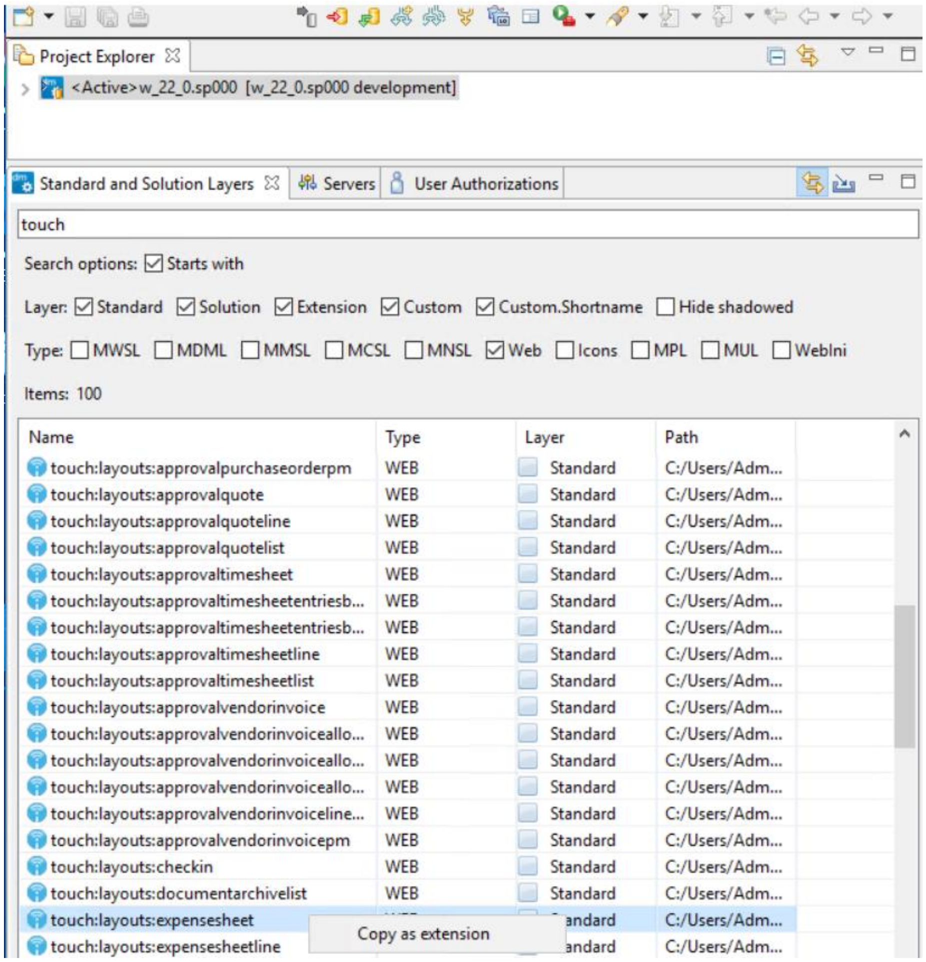


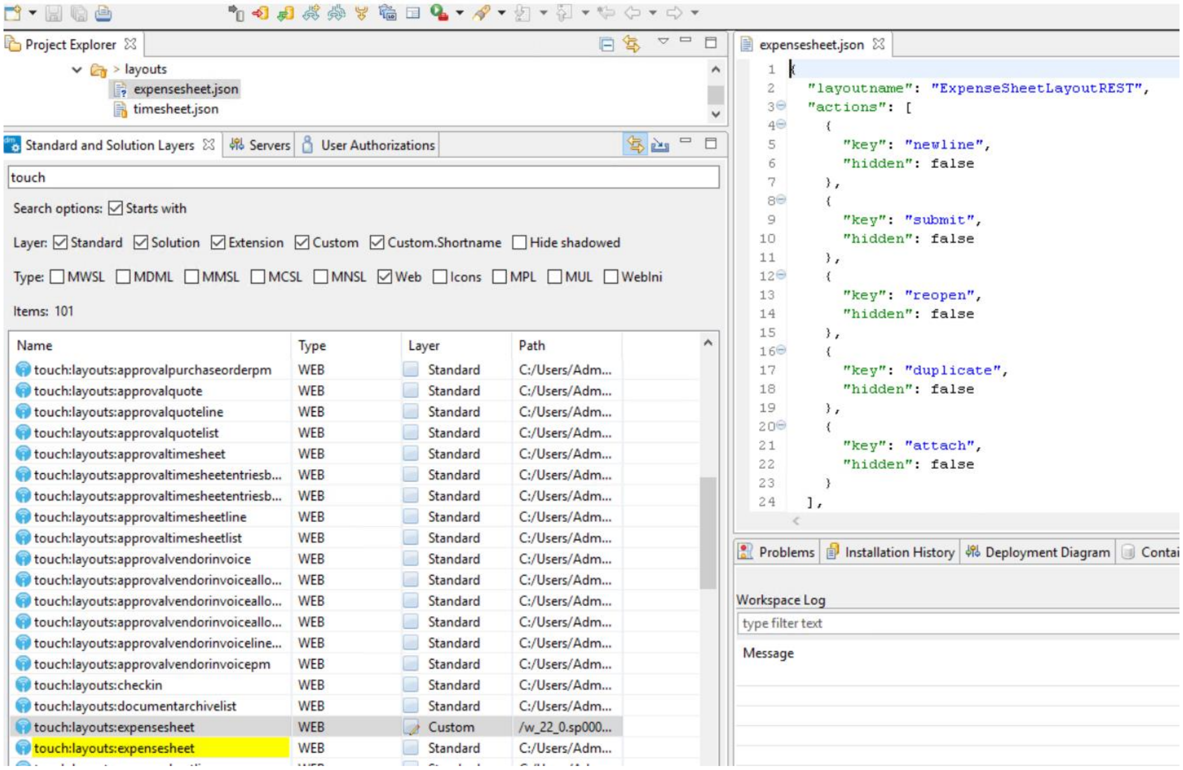
In Extender, you can see the entire REST Touch API but only some should be customized. The following list shows which items are customizable:

- **Layouts:** This folder contains the list of layouts used by Touch, including individual layout files. All of which can be customized.
- **Terms:** This folder contains the terms used by Touch. You can change the term strings but must update the Maconomy dictionaries to include the translation for new strings.
- **application.json:** This is the entry point for the REST Touch API, and *should not be customized*.
- **customsettings.json:** This is where you should customize settings.
- **schema.json:** This serves as documentation for the REST Touch API, and *should not be customized*.
- **settings.json:** This contains all settings and respective standard values, which *should not be customized*.

Customize Layouts with Extender

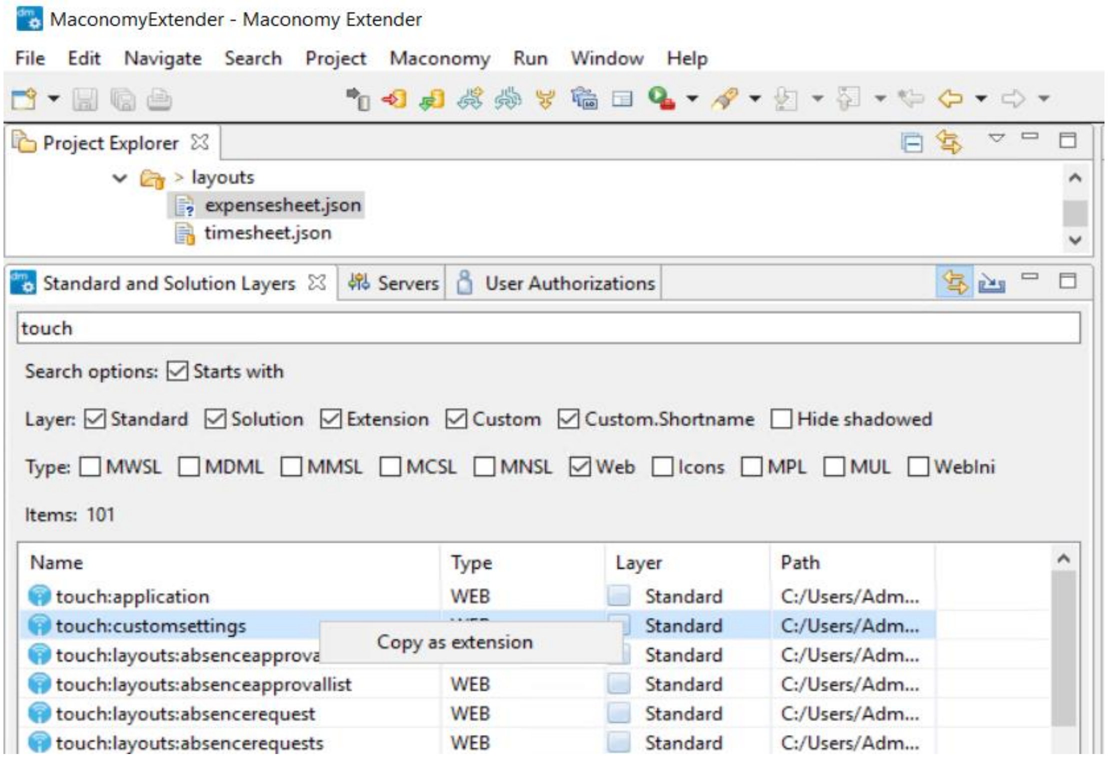
In Extender, open your repository and search for the standard version of your layout file then right-click and select **Copy as extension**. This creates a custom layout file which you can edit as shown in the following figures:

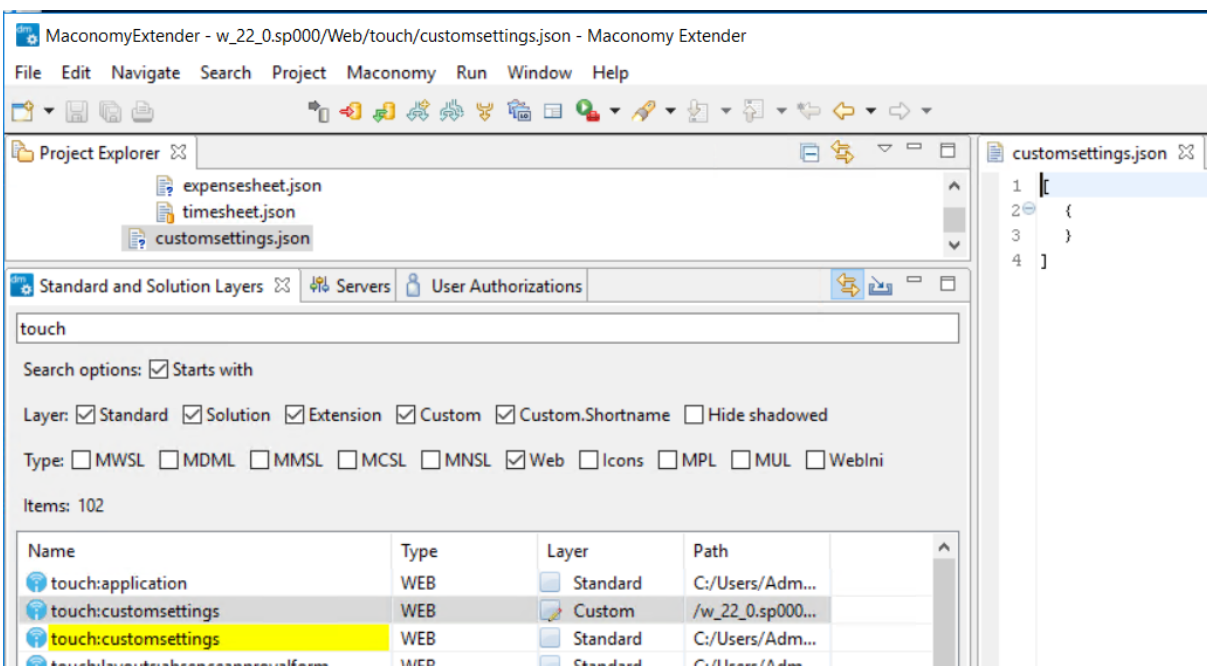




Customize Settings with Extender

In Extender, open your repository and search for **customsettings.JSON** (*not settings.JSON*), then right-click and select **Copy as extension**. This creates a custom layout file which you can edit as shown in the following figures:


MaconomyExtender - Maconomy Extender
File Edit Navigate Search Project Maconomy Run Window Help
Project Explorer
> layouts
expensesheet.json
timesheet.json
Standard and Solution Layers
Servers
User Authorizations
touch
Search options: Starts with
Layer: Standard Solution Extension Custom Custom.Shortname Hide shadowed
Type: MWSL MDML MMSL MCSL MNSL Web Icons MPL MUL WebIni
Items: 101
Name
Type
Layer
Path
touch:application
WEB
Standard
C:/Users/Adm...
touch:customsettings
Standard
C:/Users/Adm...
Copy as extension
touch:layouts:absenceapprova
Standard
C:/Users/Adm...
touch:layouts:absenceapprovallist
WEB
Standard
C:/Users/Adm...
touch:layouts:absencerequest
WEB
Standard
C:/Users/Adm...
touch:layouts:absencerequests
WEB
Standard
C:/Users/Adm...

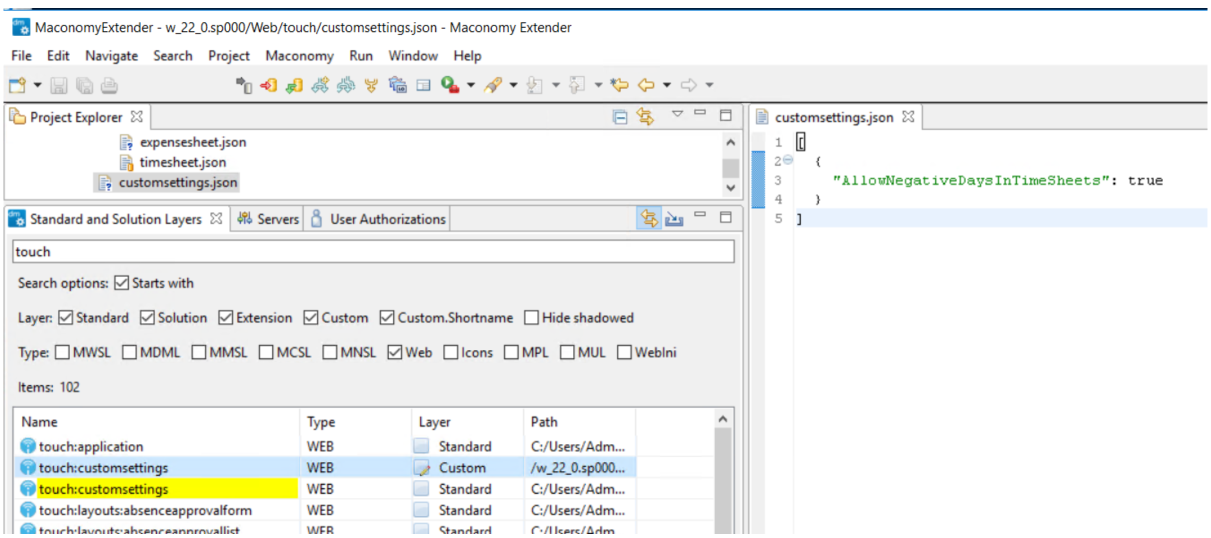

MaconomyExtender - w_22_0.sp000/Web/touch/customsettings.json - Maconomy Extender
File Edit Navigate Search Project Maconomy Run Window Help
Project Explorer
expensesheet.json
timesheet.json
customsettings.json
customsettings.json
1 [
2 {
3 }
4]
Standard and Solution Layers
Servers
User Authorizations
touch
Search options: Starts with
Layer: Standard Solution Extension Custom Custom.Shortname Hide shadowed
Type: MWSL MDML MMSL MCSL MNSL Web Icons MPL MUL WebIni
Items: 102
Name
Type
Layer
Path
touch:application
WEB
Standard
C:/Users/Adm...
touch:customsettings
WEB
Custom
/w_22_0.sp000...
touch:customsettings
WEB
Standard
C:/Users/Adm...

Note:

- settings.JSON contains all settings and the respective standard values.
- customsettings.JSON contains settings with custom values only.

  When your system is upgraded, your custom settings are secured if you change these in the customsettings.JSON.

By default, the file is empty. You should add the setting you want to customize with the new value only. You can review all the available settings in settings.JSON but must not change any values. Changes you make in settings.JSON are reflected on the next Maconomy upgrade.



Note: For more details, see the *Deltek Maconomy Extender Handbook*.

Rewrite Customizations in REST

When you upgrade to Maconomy 2.6 (or higher), all existing Touch customizations become obsolete and are not reflected in the Touch app due to the MScript web service removal. The technology used for the Touch API is changed from MScript .l files to REST .JSON files.

Redo the layout and setting customizations in Extender. Heavyweight customizations (for example, functional changes in .ms files), if any, need to be redone as Java extensions.

Rewrite Custom Layouts

Deltek recommends you start with the new .JSON layout then add your customizations instead of starting with your custom .l files and converting these into valid .JSON files.

To recreate your custom layouts:

1. Identify your custom layout files. Previously, these were included in …/MaconomyMScript/Custom/Layouts.
2. For each custom layout file, identify the customizations by comparing the custom and standard version of the .l file.
3. For each custom layout file (.l), search for the corresponding new standard layout file (.JSON).

   > **Attention:** The following changes are made to the layout file names:
   >
   > - File extensions are changed from .l to .JSON.
   > - File names are in lowercase.
   > - The LayoutREST suffix was removed.
   >
   > For example, the TimesheetLayoutREST.l layout is changed to timesheet.JSON.
   >
   > A comparison of the .l and .JSON layouts are available in the Layout File Matrix section.

4. Follow the steps in Customize Layouts with Extender.
5. Get the content from the standard layout and then add your customizations.

   > **Attention:** Since there are many changes in the standard layout files, it is recommended that you get the content from the standard layout, and then add your customizations. It is important to take note of the following changes:
   >
   > - The syntaxes are changed to comply with the JSON format, which means additional " marks are added. Boolean values (true and false) do not need to be included in " marks. But all string values should be between ".
   > - To translate terms, use T$ on the left side instead of #T on the right side.
   > - All .l specific translation tags (#T, #N, and #K) should be removed.

Example

In **…/MaconomyMScript/Custom/Layouts**, there is **MileageSheetLineLayoutREST.l**.

When you compare …/MaconomyMScript/**Custom**/Layouts/MileageSheetLineLayoutREST.I with …/MaconomyMScript/**Standard**/Layouts/MileageSheetLineLayoutREST.I, you determine the customization is to make the Unit Price field read-only.

To redo the customization:

1. Search for the new standard file (MileageSheetLineLayoutREST.I is now mileagesheetline.JSON).
2. In Extender, take the following steps:
   a. Search for the standard mileagesheetline.JSON, then right-click and select **Copy as extension**.
   b. Change the custom mileagesheetline.JSON by adding your customizations in valid .JSON syntax.

   Sample code:

```
{
    "source": "UnitPriceCurrency",
    "T$label”: "Unit Price",
    "type": "readonlyfield",
    "mask": "decimal"
},
```

Rewrite Custom Settings

To recreate custom settings:

1. Identify the custom settings you need. Previously, these were included in …/MaconomyMScript/Custom/DeltekTouch.I.

   To identify the custom settings, you can compare the …/MaconomyMScript/**Custom**/DeltekTouch.I and …/MaconomyMScript/**Standard**/DeltekTouch.I files.
2. Follow the steps in Customize Layouts with Extender.

Example

If you need to enable ICR and daily time sheets, you should have a customized customsettings.JSON containing the following code:

```
[
  {
      "CanChangeSettingICR": true,
      "DefaultValueForSettingICR": true,
      "ShowSettingICR": true,
      "Submit": "daily"
  }
]
```

> **Note:** All settings are described in Settings Configuration.

Configure a Timesheet Line or Add Job Screen

To configure the Timesheet Line or Add Job screen:

1. Open your repository in Extender.
2. Search for the standard **timesheetlogentry.JSON,** then right-click and select **Copy as extension**.
3. Open the custom **timesheetlogentry.JSON** and change the file accordingly.

Configure an Approval or a Summary Screen

To configure an Approval or Summary screen:

1. Open your repository in Extender.
2. Search for the standard layout file, then right-click and select **Copy as extension**.

   If you want to add or update the fields in the layout file, you can use any field available in the related Maconomy container.

   > **Note:** For the list of layouts and corresponding containers, see Appendix A: Screens, Layouts, & Containers.

Add a New Search or Lookup Field

To add a lookup type field to a REST layout:

> **Note:** For example, add **Specification3Name** to **timesheetlogentryweekly.JSON**.

1. Create custom layouts.JSON.
2. Modify the custom layouts.JSON and add a new lookup layout as shown in the following code:

   ```
   {
          "$ref": "find_specification3"
   },
   ```

3. When creating a new lookup layout, the layoutname should be the same with the containerName found on the specification's relatedContainer. This layout name is case-sensitive. For example, `"layoutname": "find_specification3"`.
4. Use the custom lookup layout by adding the following code in the timesheetlogentryweekly.JSON.

   In adding the field specification3 to this layout, searchlayoutname should be the same with what you set with layouts.JSON.

Sample code:

```
{       "T$label": "Location of Work",
        "source": "Specification3Name",
        "searchlayout": "find_specification3",
        "type": "lookupfield",
         "queryfieldname": "Specification3Name",
        "displayfields":  [
        {
            "source": "Specification3Name",
             "queryfieldname": "Specification3Name"
        },
         {
             "label": " - "
         },
         {
             "source": "Specification3DescriptionVar",
             "queryfieldname": "Description"
          }
        ]
},
```

Modify Search or Lookup Fields

This procedure allows you to restrict the data shown in basic and advanced search screens.

> **Note:** For the list of layouts and corresponding containers, see Appendix A: Screens, Layouts, & Containers.

> **Attention:** When using popup fields in the restriction, the popup value must be included between double backslashes. For example, `jobgroup = JobGroupType'\\utvikling/investering\\`.

Restrict Data in Basic Search Screens

Basic search screens include screens like **Find Activity** and **Find Task**. All basic search screens behave similarly.

To modify the search data for a specific field on one screen, add the restrictions to the screen layout. However, to modify the search data for a specific field on multiple screens, add the restrictions to the corresponding search layout. You can add the restriction on the search layout, and the value is automatically appended on the screen layout. For example:

- If for timesheets you want to use only tasks in Company 1, whereas for expense sheets you want to use only tasks in Company 2, take the following steps:

    - Add the Company 1 restriction to the TaskName lookup field on the Timesheet Line screen layout (timesheetlogentryweekly.JSON or timesheetlogentrydaily.JSON).
    - Add the Company 2 restriction to the TaskName lookup field on the Expense Sheet Line screen layout (expensesheetline.JSON).
- If you want to use only tasks in Company 1 everywhere in Touch (time, expense, mileage, and so on), take the following step:
    - Add the Company 1 restriction to the Find Task search layout (find_tasklistline.JSON).

To customize basic search screens:

1. Customize the standard layout, either the screen or search layout, containing the field to be modified.
2. Modify the custom layout to include the restriction. To append your condition, insert the following code:

```
"options" : {
   "restriction": "samplequery = 1"
}
```

Modify the highlighted text according to your specification. Replace `samplequery` with the `queryfieldname` you want to restrict and replace `1` with the value you want shown on the screen.

Examples

- Restriction on the lookup field in the screen layout (timesheetlogentryweekly.JSON)

```
{
     "source": "TaskName",
     "T$label": "Task",
     "searchlayoutname": "find_tasklistline",
     "queryfieldname": "taskname",
     "type": "lookupfield",
     "options": {
     "restriction": "blocked=false"
     },
     "displayfields": [
     {
     "source": "taskname",
     "queryfieldname": "taskname"
     },
     {
     "label": " - "
     },
     {
     "source": "tasktextvar",
     "queryfieldname": "description"
```

```
        }
      ]
    },
```

- Restriction in the search layout (find_tasklistline.JSON)

```
{
    "protectedregion": false,
    "region": "SearchFieldArea",
    "type": "searchfield",
    "filterkeys": ["taskname", "description"],
    "T$placeholder": "Task Number, Name"
    "options": {
         "restriction": "blocked=false"
            }
},
```

Restrict Data in Advanced Search Screens

Advanced search screens include **Find Vendor**, **Find Job**, and **Address** search screens. The data is provided by two or more REST requests (filter and/or foreignkey) to different containers. You can customize the restriction on all these REST requests.

The following list discusses the 3 advanced search screens and options available for customization:

1. **Find Vendor:** This screen is used on the **Vendor** field in purchase orders, and has the following options:
   - **recentlyused**: This option refers to the REST request which retrieves the jobs used on the most recent purchase orders. When you open the screen, this request is invoked.
   - **all:** This option refers to the REST request which retrieves all jobs. When you tap the magnifying glass icon, this request is invoked.

   A restriction added on an option is applied only on the REST request related to the option. Restrictions can be added in the screen layout (purchaseorder.JSON) and/or search layout (findvendor_purchaseorder.JSON).

   If you add a restriction on the findvendor_purchaseorder.JSON search layout, it will append its value to the "all" restriction from the screen layout. This is an option if you want to add the restriction inside the "all" option.
2. **Find Job:** This screen is used on the **Job** field on timesheet lines, expense sheets and expense sheet lines, mileage sheet and mileage sheet lines, and purchase orders and purchase order lines. This screen has the following options:
   - **recentlyused:** This option refers to the REST request which retrieves the jobs used on the most recent items (for example, timesheets). When you open the screen, this request is invoked.
   - **all:** This option refers to the REST request which retrieves all jobs. When you tap the magnifying glass icon, this request is invoked.
   - **favorite:** This option refers to the REST request which retrieves favorites. When you tap the Favorite tab, this request is invoked.

- **myjob:** This option refers to the REST request which retrieves jobs you have access to based on employee control. This request is invoked when you open the screen, if the **ShowJobsWithEmployeeControl** server setting is set to **True**.

You can add any option (Recently Used, My, All, and Favorite) that is applicable in the following contexts:

- **Timesheet line options:** recently used, my, all, favorite
- **Expense sheet options:** recently used, all
- **Expense sheet line options:** recently used, all, favorite
- **Mileage sheet options:** recently used, all
- **Mileage sheet line options:** recently used, all, favorite
- **Purchase order options:** recently used, all
- **Purchase order line options:** recently used, all

3. **Address:** The **From** and **To** address search screens are available for mileage sheet lines. You can only add the restriction on the mileagesheetline.JSON screen layout.

To customize advanced search screens:

1. Customize the standard layout, either the screen or search layout, containing the field to be modified.
2. Modify the custom layout to include the restriction. Take any of the following actions:
   - Specify a default restriction, which is applied to all REST requests issued for the advanced search screen.

     Sample code:

     ```
     "options" : {
        "restriction": "samplequery = 1"
     }
     ```

   - Specify different restrictions for the different REST requests used for the advanced search screen. The REST requests are represented by option names (for example, all). The option names (option1, option2) are specific to the advanced search screens.

     Sample code:

     ```
     "options": {
      "option1": {
         "restriction": "samplequery = 1"
      },
      "option2": {
         "restriction": "samplequery = 2"
      }
     ```

   Modify the highlighted texts according to your specification. Replace `samplequery` with the `queryfieldname` you want to restrict and replace `1` with the value you want shown on the screen.

Examples

- Restriction on the **Vendor** lookup field in the screen layout (purchaseorder.JSON)

```
{
    "T$label": "Vendor",
    "source": "suppliernumber",
    "queryfieldnames": ["vendornumber", "suppliernumber"],
    "type": "vendorlookupfield",
    "mode": "project",
    "searchlayoutname": "findvendor_purchaseorder",
    "options": {
          "recentlyused": {
                "restriction": "AllowPurchaseOrders = 0"
          },
          "all": {
                "restriction": "vendorgroup = VendorGroupType'\\europe\\"
          }
      },
    "displayfields": [
    {
          "source": "suppliernumber",
          "queryfieldnames": ["vendornumber", "suppliernumber"]
    },
    {
          "label": " - "
    },
    {
          "source": "name1",
          "queryfieldname": "name1"
    }
    ]
},
```

- Restriction in the Job search layout (find_jobheader_timesheet.JSON)

```
{
      "protectedregion": false,
      "region": "SearchFieldArea",
      "type": "searchfield",
      "filterkeys": ["jobnumber", "jobname", "name1"],
      "T$placeholder”: "Job Number, Name, Customer Name",
      "options": {
          "recentlyused": {
                "restriction": "jobnumber = 'MTJ03'"
```

```
            },
            "myjob": {
                    "restriction": "jobnumber = 'MTJ03'"
            },
            "all": {
                    "restriction":
        "companynumber={api_currentemployee}.{companynumber}"
            }
            }
},
```

- Restriction on the **From** and **To** fields in the screen layout (mileagesheetline.JSON)

```
{
   "source": "MileageFrom",
   "container": "find_recentlyusedmileagefromu",
   "T$label": "From",
   "type": "locationpicker",
   "options": {
      "restriction": "sample=1"
   }
}
```

Customize Actions

This section describes how you can modify an action or add a new one to the action menu of all Touch screens with exception to Approvals.

Modify an Action

To modify an action:

> **Note:** For example, customize the **Submit** action on the **Expense Sheet** screen.

1. Customize the standard layout corresponding to the Touch screen.
2. In the custom layout, search for your action and then add the source, which is the name of the Maconomy action you would like to use

   Sample code:

```
{
        "layoutname": "ExpenseSheetLayoutREST",
        "actions": [
                { "key": "newline", "hidden": false},
                { "key": "submit", "hidden": false, "source": "action:mysubmit"},
                { "key": "reopen", "hidden": false},
```

```
            { "key": "duplicate", "hidden": false},
            { "key": "attach", "hidden": false}
      ],
```

Note: For more information on action attributes, see Actions.

Add a New Action

For you to add a new action in Touch, the action must already be available in Maconomy. However, some Maconomy actions, like the **Print Timesheet** action, cannot be added in Touch.

Note: When you add a new action, an action menu is added to the screen if it is not initially available. The action menu will contain all new actions in the order these actions appear in the layout.

To add a new action to a Touch screen:

Note: For example, add a new **Start Timer** action on the **Timesheet** screen.

1. Customize the standard layout corresponding to the Touch screen.
2. Modify the custom layout to add the new action.

   Sample code:

```
{
      "layoutname": "TimesheetLayoutREST",
      "actions": [
            { "key": "submit", "hidden": false},
            { "key": "reopen", "hidden": false},
            { "key": "starttimer", "hidden": false,
       "source":"action:MyStartTimerAction", "T$title": "Start Timer"}
      ],
```

Note: For more information on action attributes, see Actions.

Customize Timesheet Actions

When you add new actions on the **Timesheet** screen, you can modify either the timesheet.JSON layout for weekly timesheets, or timesheetdays.JSON layout for daily timesheets.

The **Timesheet** screen has **Submit** and **Reopen** standard actions and no action menu. Standard Touch actions are shown in the action menu on Touch screens with exception to the **Timesheet** screen, which shows **Submit**, **Reopen**, and **+** (New Line) as buttons. You can neither rename nor move these buttons to the action menu but you can hide these on the screen.

Example

If you want to hide the **Submit** standard action on the **Weekly Timesheets** screen and add your **Submit** custom action, modify the following code in the timesheet.JSON custom layout:

```
"layoutname":"TimesheetLayoutREST",
"actions": [
        { "key": "submit", "hidden": true },
        { "key": "reopen", "hidden": false },
        { "key": "mysubmit", "hidden": false, "source": "action:mysubmit",
"T$title": "Submit"}
],
```

An action menu is added, which contains your **Submit** custom action, and the **Submit** standard action is hidden from the **Timesheet** screen.

Add Related Option List Fields

To add option list type fields:

> **Note:** For example, add **State (**SelectedOption1) and **County** (SelectedOption2) to **timesheetlogentryweekly.JSON**. Values display in **County** depend on the value selected in **State**.

1. Customize layouts.JSON.
2. Modify layouts.JSON and add the new lookup layout as shown in the following example:

```
{
     "$ref": "find_theoption"
 },
```

3. When creating a new lookup layout, the layoutname should be the same as the containerName found in the specification's relatedContainer. This layout name is case-sensitive. For example, `"layoutname": "find_theoption"`.
4. Use the custom lookup layout in timesheetlogentryweekly.JSON. When adding the option listsfields to this layout, searchlayoutname should be the same as what you set with layouts.JSON.

   For adding State (=SelectedOption1), you need to add to the following code to timesheetlogentryweekly.JSON:

```
{
     "T$label": "State",
     "source": "selectedoption1",
     "type": "lookupfield",
     "searchlayoutname": "find_theoption",
     "queryfieldname": "name",
     "displayfields": [
     {
        "source": "selectedoption1",
```

```
            "queryfieldname": "name"
         }
      ]
   },
```

For adding County (=SelectedOption2), you need to add the following code:

```
{
    "T$label": "County",
    "source": "selectedoption2",
    "type": "lookupfield",
    "searchlayoutname": "find_theoption",
    "queryfieldname": "name",
    "restriction": "remark1 = selectedoption1",
    "displayfields": [
    {
        "source": "selectedoption2",
        "queryfieldname": "name"
    }
  ]
},
```

Change the Style

This procedure allows you to change the existing style.

To change the style:

1. Go to the Custom folder.
2. Add the CssClass.l file.
3. Check the `className` name you want to modify.
4. Customize layouts.JSON.
5. In custom layouts.JSON, include `CssClass`.

> **Note:** Make sure that you place the ClssClass.JSON file in the first line of the layouts.JSON file.

Example

The following sample code shows how to change the color:

```
{"CssClass":[
   {
         "className": "text_medium_bold",
```

```
            "style": {
                  "fontsize": "16px!important",
                  "textalignment": "left",
                  "color": "#f2b600"
            }
      }
   ]}
```

Change Fields to Read-Only

These procedures allow you to change editable fields to read-only.

Amount Fields

You can change an editable amount field (any real number with two decimals) to read-only.

> **Note:** For example, Change the **Unit Price** field in the **mileagesheetline.JSON** custom layout from editable to read-only.

To change an amount field to read-only:

1. Go to the mileagesheetline.JSON custom file.
2. Add the following values:

```
{
   "source": "UnitPriceCurrency",
   "T$label": "Unit Price",
   "type": "readonlyfield",
   "mask": "decimal"
},
```

Lookup Fields

You can change editable lookup fields to read-only.

> **Note:** For example, change the **Requisitioner** field in the **purchaseorder.JSON** custom layout from editable to read-only.

To change a lookup field to read-only:

1. Go to the purchaseorder.JSON custom file.
2. Add the following values:

```
   "T$label": "Requisitioner",
   "source": "requisitioneremployeenumber",
   "queryfieldname": "employeenumber",
```

```
"type": "lookupfield",
"searchlayoutname": "find_requisitioner",
"mask": "disabled",
```

String Fields

You can change editable string fields to read-only.

> **Note:** For example, change the **Daily Description** field in the timesheetlogentrydaily.JSON custom layout from editable to read-only.

To change a string field to read-only:

1. Go to the timesheetlogentrydaily.JSON custom file.
2. Add the following values:

```
"type": "textareafield",
"T$label": "Daily Description",
"source": "longdescriptionday",
"cls": "longText",
"mask": "disabled"
```

Boolean Fields

Copy to come.

SETTINGS

Settings Configuration

You can modify the following server-side settings in **customsettings.JSON**.

Attention: Use Extender for customizing settings. You can view all settings and the respective default values in settings.JSON.

Copy the settings you need to customize to customsettings.JSON and change the value as needed. Do not change settings.JSON as it could cause potential issues when you upgrade Maconomy.

Although not required, if your changes do not reflect after you log on to Touch again, restart the Coupling Service.

Key	Valid Options	Default Value	Description
ShowSettingOpenIn	True or False	True	This option allows you to hide the **Open In** field on the **Settings** screen. ▪ If it is set to **true**, users can see **Open In** on **Settings**. ▪ If **false**, **Open In** does not display.
DefaultValueForSettingOpenIn	▪ Timesheet ▪ Approvals ▪ Expense Sheets ▪ Quick Capture ▪ Mileage Sheets ▪ Summary	Timesheet	This option represents the default value of the **Open In** field on the **Settings** screen.
CanChangeSettingOpenIn	True or False	True	This option allows you to control if users can change the **Open In** setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Open In** on **Settings**. ▪ If **false**, users cannot change the **Open In** value.
ShowSettingUseWeekNumbers	True or False	True	This option allows you to hide the **Use Week Number** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Use Week Number** on **Settings**.

Key	Valid Options	Default Value	Description
			▪ If **false**, **Use Week Number** does not display.
DefaultValueForSettingUseWeekNumbers	True or False	False	This option represents the default setting of the **Use Week Number** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Use Week Number** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingUseWeekNumbers	True or False	True	This option allows you to control if users can change the **Use Week Number** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Use Week Number** on **Settings**. ▪ If **false**, users cannot change the **Use Week Number** value.
ShowSettingWeekStart	True or False	True	This option allows you to control if users can change the **Week Start** setting on the **Settings** screen. ▪ If it is set to **true**, users can see **Week Start** on **Settings**. ▪ If **false**, **Week Start** does not display.
DefaultValueForSettingWeekStart	▪ 0 ▪ 1	1	This option represents the value of the **Week Start** setting on the **Settings** screen. ▪ If it is set to **0**, the assigned day is Sunday. ▪ If **1**, the assigned day is Monday.
DefaultValueForSettingWeekStart	True or False	True	This option allows you to control if users can change the value of the **Week Start** setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Week Start** on **Settings**. ▪ If **false**, users cannot change the **Week Start** value.

Key	Valid Options	Default Value	Description
ShowSettingKeepOnTimesheetAsDefault	True or False	True	This option allows you to hide the **Keep on timesheet as default** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Keep on timesheet as default** on **Settings**. ▪ If **false**, **Keep on timesheet as default** does not display.
DefaultValueForSettingKeepOnTimesheetAsDefault	True or False	True	This option represents the default setting of the **Keep on timesheet as default** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Keep on timesheet as default** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingKeepOnTimesheetAsDefault	True or False	True	This option allows you to control if users can change the **Keep on timesheet as default** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Keep on timesheet as default** on **Settings**. ▪ If **false**, users cannot change the **Keep on timesheet as default** value.
ShowSettingRefreshTimeSheetOnOpen	True or False	True	This option allows you to hide the **Refresh When Opening** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Refresh When Opening** on **Settings**. ▪ If **false**, **Refresh When Opening** does not display.
DefaultValueForSettingRefreshTimeSheetOnOpen	True or False	False	This option represents the default setting of the **Refresh When Opening** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Refresh When Opening** is **ON Settings**. ▪ If **false**, it is **OFF**.

Key	Valid Options	Default Value	Description
CanChangeSettingRefreshTimeSheetOnOpen	True or False	True	This option allows you to control if users can change the **Refresh When Opening** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Refresh When Opening** on **Settings**. ▪ If **false**, users cannot change the **Refresh When Opening** value.
ShowSettingRegistrationUnit	True or False	True	This option allows you to hide the **Registration Unit** field on the **Settings** screen. ▪ If it is set to **true**, users can see **Registration Unit** on **Settings**. ▪ If **false**, **Registration Unit** does not display.
DefaultValueForSettingRegistrationUnitMinutes	▪ 1 minute (0-59) ▪ 2 minutes ▪ 5 minutes ▪ 10 minutes ▪ 15 minutes	15 minutes	This option represents the default setting of the **Registration Unit** toggle switch on the **Settings** screen. It is used if time registration is in hours and minutes.
DefaultValueForSettingRegistrationUnitDays	▪ Half day ▪ Quarter day ▪ Fifths ▪ Tenths	Half day	This option represents the default value of the **Registration Unit** setting for time registration in days on the **Settings** screen.
CanChangeSettingRegistrationUnit	True or False	True	This option allows you to control if users can change the **Registration Unit** setting on the **Settings** screen. ▪ If it is set to **true**, users can change the **Registration Unit** on **Settings**. ▪ If **false**, users cannot change the **Registration Unit** value.
ShowSettingExpenseSheet	True or False	True	This option allows you to hide the **Expense Sheet** toggle switch on the **Settings** screen.

Key	Valid Options	Default Value	Description
			▪ If it is set to **true**, users can see **Expense Sheet** on **Settings**. ▪ If **false**, **Expense Sheet** does not display.
DefaultValueForSettingExpenseSheet	True or False	True	This option represents the default setting of the **Expense Sheet** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Expense Sheet** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingExpenseSheet	True or False	True	This option allows you to control if users can change the **Expense Sheet** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Expense Sheet** on **Settings**. ▪ If **false**, users cannot change the **Expense Sheet** value.
ShowSettingICR	True or False	True	This option allows you to hide the **Use ICR for Expenses** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Use ICR for Expenses** on **Settings**. ▪ If **false**, **Use ICR for Expenses** does not display.
DefaultValueForSettingICR	True or False	False	This option represents the default setting of the **Use ICR for Expenses** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Use ICR for Expenses** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingICR	True or False	False	This option allows you to control if users can change the **Use ICR for Expenses** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the value of **Use ICR for Expenses** on **Settings**.

Key	Valid Options	Default Value	Description
			▪ If **false**, users cannot change the **Use ICR for Expenses** value.
ICRPictureMaxFileSizeAndroid	200 to 2500 KB	1000 KB	This option allows you to control the maximum file size of receipts attached to the expense reports when the ICR functionality is enabled on an Android device. Setting the file size to 2000 KB offers better resolution but the photos you attach when you use ICR on an Android phone can exceed 2 MB.
ICRAllowStitchingAndroid	True or False	True	This option allows you to control if users can utilize the stitch functionality when ICR is enabled on an Android device. ▪ If it is set to **true**, users can combine 2 to 4 receipt photos into a PDF. ▪ If **false**, users cannot combine receipt photos into a PDF.
ICRPDFQualityAndroid	1.0 to 5.0	1.5	This option allows you to control the quality of the photos to be combined in a PDF when ICR is enabled on an Android device. The recommended photo quality is 1.5 to 2.0. Anything beyond 2.0 creates PDF files exceeding 10 MB.
ICRPictureMaxFileSizeiOS	200 to 2500 KB	1500 KB	This option allows you to control the maximum file size of receipts attached to the expense reports when the ICR functionality is enabled on an iOS device. Setting the file size to 2000 KB offers better resolution but the photos you attach when you use ICR on an iPhone can exceed 2 MB.
ICRAllowStitchingiOS	True or False	True	This option allows you to control if users can utilize the stitch functionality when ICR is enabled on an iOS device. ▪ If it is set to **true**, users can combine 2 to 4 receipt photos into a PDF. ▪ If **false**, users cannot combine receipt photos into a PDF.
ICRPDFQualityiOS	1.0 to 5.0	1.5	This option allows you to control the quality of the photos to be combined in a PDF when ICR is enabled on an iOS

Key	Valid Options	Default Value	Description
			device. The recommended photo quality is 1.5 to 2.0. Anything beyond 2.0 creates PDF files exceeding 10 MB.
ShowSettingMileageSheet	True or False	True	This option allows you to hide the **Mileage Sheet** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Mileage Sheet** on **Settings**. ▪ If **false**, **Mileage Sheet** does not display.
DefaultValueForSettingMileageSheet	True or False	True	This option represents the default setting of the **Mileage Sheet** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Mileage Sheet** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingMileageSheet	True or False	True	This option allows you to control if users can change the **Mileage Sheet** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Mileage Sheet** on **Settings**. ▪ If **false**, users cannot change the **Mileage Sheet** setting.
ShowSettingPurchaseOrder	True or False	True	This option allows you to hide the **Purchase Order** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Purchase Order** on **Settings**. ▪ If **false**, **Purchase Order** does not display.
DefaultValueForSettingPurchaseOrder	True or False	True	This option represents the default setting of the **Purchase Order** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Purchase Order** is **ON Settings**. ▪ If **false**, it is **OFF**.

Key	Valid Options	Default Value	Description
CanChangeSettingPurchaseOrder	True or False	True	This option allows you to control if users can change the **Purchase Order** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Purchase Order** on **Settings**. ▪ If **false**, users cannot change the **Purchase Order** setting.
ShowSettingAbsenceRequests	True or False	True	This option allows you to hide the **Absence Requests** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Absence Requests** on **Settings**. ▪ If **false**, **Absence Requests** does not display.
DefaultValueForSettingAbsenceRequests	True or False	True	This option represents the default setting of the **Absence Requests** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Absence Requests** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingAbsenceRequests	True or False	True	This option allows you to control if users can change the **Absences Requests** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Absence Requests** on **Settings**. ▪ If **false**, users cannot change the **Absence Requests** setting.
ShowSettingAllowanceRequests	True or False	True	This option allows you to hide the **Allowance Requests** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Allowance Requests** on **Settings**. ▪ If **false**, **Allowance Requests** does not display.

Key	Valid Options	Default Value	Description
DefaultValueForSettingAllowanceRequests	True or False	True	This option represents the default setting of the **Allowance Requests** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Allowance Request** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingAllowanceRequests	True or False	True	This option allows you to control if users can change the **Allowance Requests** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Allowance Requests** on **Settings**. ▪ If **false**, users cannot change the **Allowance Requests** setting.
ShowSettingRejections	True or False	True	This option allows you to hide the **Rejections** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Rejections** on **Settings**. ▪ If **false**, **Rejections** does not display.
DefaultValueForSettingRejections	True or False	True	This option represents the default setting of the **Rejections** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Rejections** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingRejections	True or False	True	This option allows you to control if users can change the **Rejections** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Rejections** on **Settings**. ▪ If **false**, users cannot change the **Rejections** setting.
ShowSettingApprovals	True or False	True	This option allows you to hide the **Approvals** toggle switch on the **Settings** screen.

Key	Valid Options	Default Value	Description
			▪ If it is set to **true**, users can see **Approvals** on **Settings**. ▪ If **false**, **Approvals** does not display.
DefaultValueForSettingApprovals	True or False	False	This option represents the default setting of the **Approvals** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Approvals** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingApprovals	True or False	True	This option allows you to control if users can change the **Approvals** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Approvals** on **Settings**. ▪ If **false**, users cannot change the **Approvals** setting.
IncludeTimeSheetApproval	True or False	True	This option allows you to control timesheet approval in Touch. ▪ If it is set to **true**, you can approve timesheets (headers and lines). ▪ If **false**, you cannot approve timesheets (headers and line).
IncludeExpenseSheetApproval	True or False	True	This option allows you to control expense and mileage sheet approval in Touch. ▪ If it is set to **true**, you can approve expense sheets (headers and lines). ▪ If **false**, you cannot approve expense sheets (headers and line).
IncludePurchaseOrderApproval	True or False	True	This option allows you to control purchase order approval in Touch. ▪ If it is set to **true**, you can approve purchase orders (headers and lines). ▪ If **false**, you cannot approve purchase orders (headers and line).
IncludeVendorInvoiceApproval	True or False	True	This option allows you to control vendor invoice approval in Touch.

Key	Valid Options	Default Value	Description
			▪ If it is set to **true**, you can approve vendor invoices (headers and lines). ▪ If **false**, you cannot approve vendor invoices (headers and line).
IncludeJobInvoiceApproval	True or False	True	This option allows you to control job draft invoice approval in Touch. ▪ If it is set to **true**, you can approve job invoices (headers and lines). ▪ If **false**, you cannot approve job invoices (headers and line).
IncludeJobQuoteApproval	True or False	True	This option allows you to control job quote approval in Touch. ▪ If it is set to **true**, you can approve job quotes (headers and lines). ▪ If **false**, you cannot approve job quotes (headers and line).
IncludeAbsenceApproval	True or False	True	This option allows you to control absence request approval in Touch. ▪ If it is set to **true**, you can approve absence requests (headers and lines). ▪ If **false**, you cannot approve absence requests (headers and line).
IncludeAllowanceAprova	True or False	True	This option allows you to control allowance request approval in Touch. ▪ If it is set to **true**, you can approve allowance requests (headers and lines). ▪ If **false**, you cannot approve allowance requests (headers and line).
ShowSettingInvoiceAllocationLines	True or False	True	This option allows you to hide the **Invoice Allocation Lines** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Invoice Allocation Lines** on **Settings**. ▪ If **false**, **Invoice Allocation Lines** does not display.

Key	Valid Options	Default Value	Description
DefaultValueForSettingInvoiceAllocationLines	▪ Job invoice allocation lines only ▪ All invoice allocation lines ▪ Job and All default	Job and All default	This option represents the default setting of the **Invoice Allocation Lines** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Invoice Allocation Lines** is **ON Settings**. ▪ If **false**, it is **OFF**. Note: This is used only in MScript approvals
CanChangeSettingInvoiceAllocationLines	True or False	True	This option allows you to control if users can change the **Invoice Allocation Lines** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Invoice Allocation Lines** on **Settings**. ▪ If **false**, users cannot change the **Invoice Allocation Lines** setting.
ShowSettingSubstituteApproval	True or False	True	This option allows you to hide the **Substitute Approval** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Substitute Approval** on **Settings**. ▪ If **false**, **Substitute Approval** does not display.
DefaultValueForSettingSubstituteApproval	True or False	True	This option represents the default setting of the **Substitute Approval** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Substitute Approval** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingSubstituteApproval	True or False	True	This option allows you to control if users can change the **Substitute Approval** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Substitute Approval** on **Settings**. ▪ If **false**, users cannot change the **Substitute Approval** setting.

Key	Valid Options	Default Value	Description
ShowSettingIncludeSubstituteApprovalsInCounters	True or False	True	This option allows you to hide the **Include Substitute Approvals in Counters** toggle switch on the **Settings** screen. ▪ If it is set to **true**, users can see **Include Substitute Approvals in Counters** on **Settings**. ▪ If **false**, **Include Substitute Approvals in Counters** does not display.
DefaultValueForSettingIncludeSubstituteApprovalsInCounters	True or False	True	This option represents the default setting of the **Include Substitute Approvals in Counters** toggle switch on the **Settings** screen. ▪ If it is set to **true**, **Include Substitute Approvals in Counters** is **ON Settings**. ▪ If **false**, it is **OFF**.
CanChangeSettingIncludeSubstituteApprovalsInCounters	True or False	True	This option allows you to control if users can change the **Include Substitute Approvals in Counters** toggle switch setting on the **Settings** screen. ▪ If it is set to **true**, users can change the setting of **Include Substitute Approvals in Counters** on **Settings**. ▪ If **false**, users cannot change the **Include Substitute Approvals in Counters** setting.
AllowNegativeDaysInTimeSheets	True or False	False	This option is available to all supported Maconomy versions. Determines if you are allowed to enter negative days on timesheet lines on the **Timesheet** and **Timesheet Line** screens ▪ If this is set to **true**, you are allowed to enter both positive and negative days on timesheet lines. ▪ If **false** (default), you are allowed to enter only positive days on timesheet lines.
AllowNegativeHoursInTimeSheets	True or False	False	This option is available in all supported Maconomy versions.

Key	Valid Options	Default Value	Description
			Determines if you are allowed to enter negative hours on time sheet lines in **Timesheet** and **Timesheet Line** screens ▪ If **true**, you are allowed to enter both positive and negative hours on timesheet lines ▪ If **false** (default), you are allowed to enter only positive hours on timesheet lines
CopyRemarkFromTimeSheetLineToFavorite	True or False	False	This option is available only in Maconomy 2.0 and higher. This setting controls if the **Remark** field should be copied from the time sheet line to the favorite when creating a new favorite based on a time sheet line ▪ If true, the **Remark** field value is copied from the timesheet line into the favorite (which is creaed from the time sheet line) ▪ If false, the **Remark** field value is NOT copied from the timesheet line into the favorite (which is creaed from the time sheet line)
ExpenseAttachmentType	string	“header and line”	This option is available only in Maconomy 2.0 and higher. This setting controls where can you attach receipts to expenses, and use the ICR functionality to scan the receipts. It can have one of the following values: ▪ "**none**" = This value indicates that it is not possible to attach receipts to expenses at all, and the ICR functionality is not available. ▪ "**header**" = This value indicates that it is possible to attach receipts only to the expense sheet header, and the ICR functionality is not available. ▪ "**line**" = This value indicates that it is possible to attach receipts only to the expense sheet line. The ICR functionality is available. ▪ "**header and line**" = This value indicates that it is possible to attach receipts to both the expense sheet

Key	Valid Options	Default Value	Description
			header and line. The ICR functionality is available. ▪ "**header in quick capture, line in expense**" = This value indicates that it is possible to attach receipts using Quick Capture on the **Expense Sheet** screen and, subsequently, attach the receipt from the **Expense Sheet** screen to the **Expense Sheet Line** screen. The ICR functionality is not available. **Note:** When the ICR functionality is not available, the following conditions apply: ▪ The server settings for ICR are not applicable. ▪ The **Use ICR for Expenses** toggle switch is not displayed on the **Settings** screen. ▪ There is no Veryfi plugin. You can use **Quick Capture** if enabled, but you cannot utilize it to scan receipts.
MileageAttachmentType	string	"header and line"	This setting controls where can you attach receipts to mileage sheets. It can have one of the following values: ▪ "**none**": This value indicates that it is not possible to attach receipts to mileage sheets at all. ▪ "**header**": This value indicates that it is possible to attach receipts to the mileage sheet header only. ▪ "**line**": This value indicates that it is possible to attach receipts to the mileage sheet line only. ▪ "**header and line**" = This value indicates that it is possible to attach receipts to both the mileage sheet header and line.
MaxAttachmentFileSize	Integer	1000	Determines the maximum file size (in KB) of the pictures attached to expense

Key	Valid Options	Default Value	Description
			sheets and mileage sheets in Deltek Touch. ▪ If set to **0**, it generates the smallest file size possible, but the quality might be low. This is recommended only if you experience performance penalties when attaching documents. ▪ If set to non-zero value, the functionality is unchanged; it generates the highest quality possible for the given maximum file size. For expense sheets, this setting is disregarded when the ICR feature is enabled.
NotificationRecalculationInterval	Integer	1800	This option is available only in Maconomy 2.1.1 and higher. This setting controls the frequency (in seconds) of recalculating the notification (approval and rejected time) counters in the background. ▪ For disabling the background recalculation, use -1. ▪ Setting under 5 min (300 sec) may lead to performance penalties on the Maconomy server.
searchMaxRowcount	0-100	30	This is used to restrict the number of rows that a search returns.
ShowDefaultExpenseReceiptName	Boolean	false	It allows you to see and use a default expense receipt name when taking a picture to attach to an expense sheet, both in quick capture and expense sheets flow. ▪ If **true**, a default receipt name will be shown. You can accept or replace it. ▪ If **false**, there is no default receipt name, and you must enter the receipt name manually. This setting is disregarded when the ICR feature is enabled.
ShowDefaultMileageReceiptName	Boolean	False	It allows you to see and use a default mileage receipt name.

Key	Valid Options	Default Value	Description
			▪ If **true**, a default receipt name will be shown. You can accept or replace it. ▪ If **false**, there is no default receipt name, and you must enter the receipt name manually.
ShowJobsWithEmployeeControl	True or False	False	This option is available only in Maconomy 2.0 and higher. It allows you to see the employee's jobs (based on Employee Control) on the **Find Job** screen when creating a time sheet line using a job ▪ If true, the **All** tab displays the **Recently used projects** and **My Jo**b list (jobs the user has access to via Employee Control) ▪ If false, the **All** tab displays the **Recently used projects only** list.
ShowOnlyFavoritesInJobSearch	Boolean	false	It determines the content of the job search (Find Job) screen, invoked from the **Time, Expense** and **Mileage Sheet Line** screens. ▪ If **true**, the job search (Find Job) screen shows only favorites. When creating a time, expense or mileage sheet line, you can only select a favorite, not a job. ▪ If **false**, the job search (Find Job) screen shows jobs and favorites (All and Favorites tabs). When creating a time, expense or mileage sheet line, you can select a favorite or a job.
ShowZeroHourJobsInSummary	True or False	True	This option is available in all supported Maconomy versions. It controls the **Summary** screen, Job tab content. ▪ If true (default), the timesheet summary will show all job/task/activity combinations, used in the selected weekly time sheet ▪ If false, the timesheet summary will show only job/task/activity combinations, used in the selected

Key	Valid Options	Default Value	Description
			weekly time sheet, which are having hours registered
Submit	Daily, Weekly	Weekly	This option specifies whether you register and submit time on a daily or weekly basis.

LAYOUTS

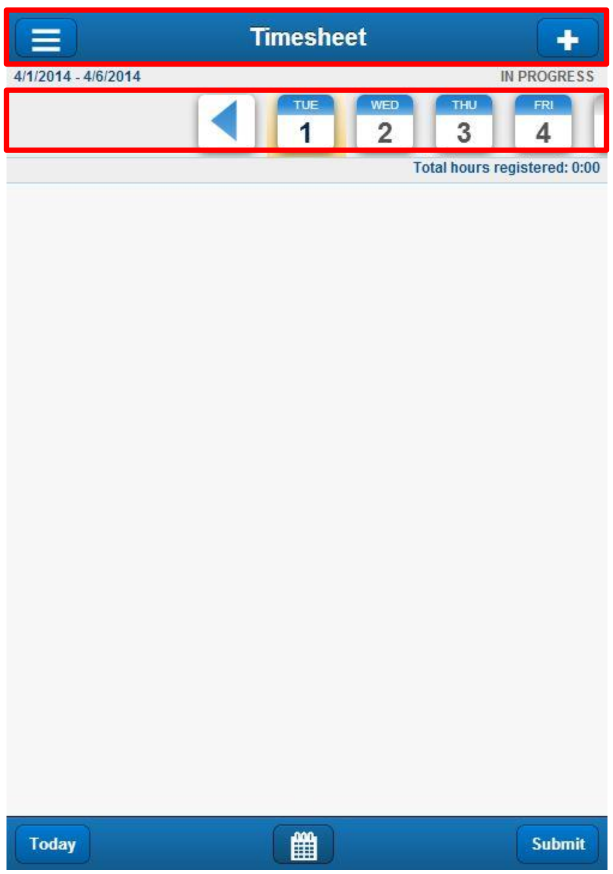
Layout Configuration

The layout customization has been introduced to help clients control fields that they want to display on and apply some styling or formatting to a Deltek Touch screen based on CSS technology.

Deltek Touch supports configuration of fields and columns on several screens (such as **Timesheet Line**, **Add Job**, **Approval**, and **Summary**), allowing you to show or hide fields that are available in the underlying Maconomy container. The layout customization capabilities, however, do not apply to all controls present on a particular screen. The common portion of the screen that you cannot customize is the title bar area, where you can usually find the screen name (for example, see red box with “Timesheet” in Figure 1).

> **Note:** The layout customization capabilities do not apply to the **Weeks**, **Calendar** (both screen and carousel), **Settings**, and document viewer.

Figure 1: Timesheet Screen



For each Deltek Touch release, there is a set of standard layouts that controls screens in the application. When you install the web service, these standard layouts are added to the **\MaconomyDir\Web\touch\layouts** folder.

The first time you customize a layout, copy it to the **CustomizationDir** folder, which allows you to preserve your customization when Deltek Touch is upgraded. Deltek recommends you do not make any changes to the standard layouts because these will be lost if you upgrade the app.

> **Note:** If you upgrade Deltek Touch, you need to update your custom files manually in the CustomizationDir folder to reflect the potential changes in the standard files.

Customization Capabilities

Deltek Touch now allows you to perform the following tasks:

- Add new fields from the underlying Maconomy container
- Format a specific element on a screen
- Change the font color, size, or style (bold, italic, or underline)
- Add a margin and padding
- Apply an ellipsis for long text (truncation)
- Add a column to some screens (such as **Summary**).
- Interchange order of lines
- Remove, and modify fields from a layout.
- Add, modify, and remove actions

> **Note:** You need to log in again to Deltek Touch for the changes to take effect.

Approval

- You cannot customize the title and icons.
- You cannot customize action buttons (such as **Approve**, **Reject**, **Undo Approve/Reject**, and **Message to Employee**).
- You cannot customize columns with and .

Summary

- You cannot customize the upper portion of a screen, which includes the title, **Submit** button, and header details.
- You cannot customize the column with . You can only customize column headers and fields.

Log Entry

You can add any field from the table part of the SpeedSheet window (DailyTimeSheetLine database table), any field from the table part of the Time Sheets window (TimeSheetLine database table), and some of the variables from the table part of SpeedSheet window.

- Each field or variable displays on a new line.
- Variables are read-only.
- Fields can either be read-only or editable, depending on the data model from the server.
- You cannot customize fonts.

Map

You can allow users to use the Map feature, which opens the Google Map and displays the **From** and **To** route, on the **Expense Sheet Line** and **Mileage Sheet Line** screens.

By default, the **Map** action is shown in the Mileage Sheet Line screen and hidden in the Expense Sheet Line screen.

Deltek recommends to use the Map functionality on the Mileage Sheet Line since this is a screen dedicated for registering mileage. In case you would like to register expense and mileage in the same screen, you would need to customize the Expense Sheet Line screen by adding the proper mileage fields (From, To, VehicleType, and so on) and making the Map action visible, as illustrated in the following code:

```
"actions": [
          { "key": "newline", "hidden": false },
          { "key": "attach", "hidden": false },
          { "key": "map", "hidden": true}
]
```

You can also apply this to the previous supported API versions (≥ 2.1), assuming that end users are using the Touch 3.2 (or newer) app.

Detach a Receipt

The **Detach Receipt** action on the Expense Sheet Line and Mileage Sheet Line screens allows users to remove receipts from their expense sheet lines and mileage sheet lines without deleting the document in the Archives folder.

To include this action for users, customize the following layouts:

- For users on Maconomy 2.5.3 and Touch 3.7 (or earlier), customize ExpenseSheetLineLayoutREST.I.
- For users on Maconomy 2.6 and Touch 4.0 (or later), customize expensesheetline.JSON.
- For users on Maconomy 2.6.1 and Touch 4.1 (or later), customize expensesheetline.JSON and/or mileagesheetline.JSON to add the action to either or both Expense Sheet Line and Mileage Sheet Line screens.

```
"actions": [
    { "key": "map", "hidden": true },
    { "key": "attach", "hidden": false },
    { "key": "newline", "hidden": false },
```

```
    { "key": "detach", "hidden": false}
],
```

Layout Files

In Touch 4.0, we have one set of layouts that corresponds to the RESTful web service.

The following changes are made to the layout file names:

- The “LayoutREST” suffix was removed.
- The file names are in lowercase.
- The file extensions are changed from .I to .JSON, and the content are updated accordingly.

For example:

Touch 3.8 (or earlier)	Touch 4.0 (or later)
ExpenseSheetLayoutREST.I	expensesheet.JSON
TimesheetLayoutREST.I	timesheet.JSON

Overall, the layout files support all configuration capabilities and allows you to perform the following tasks:

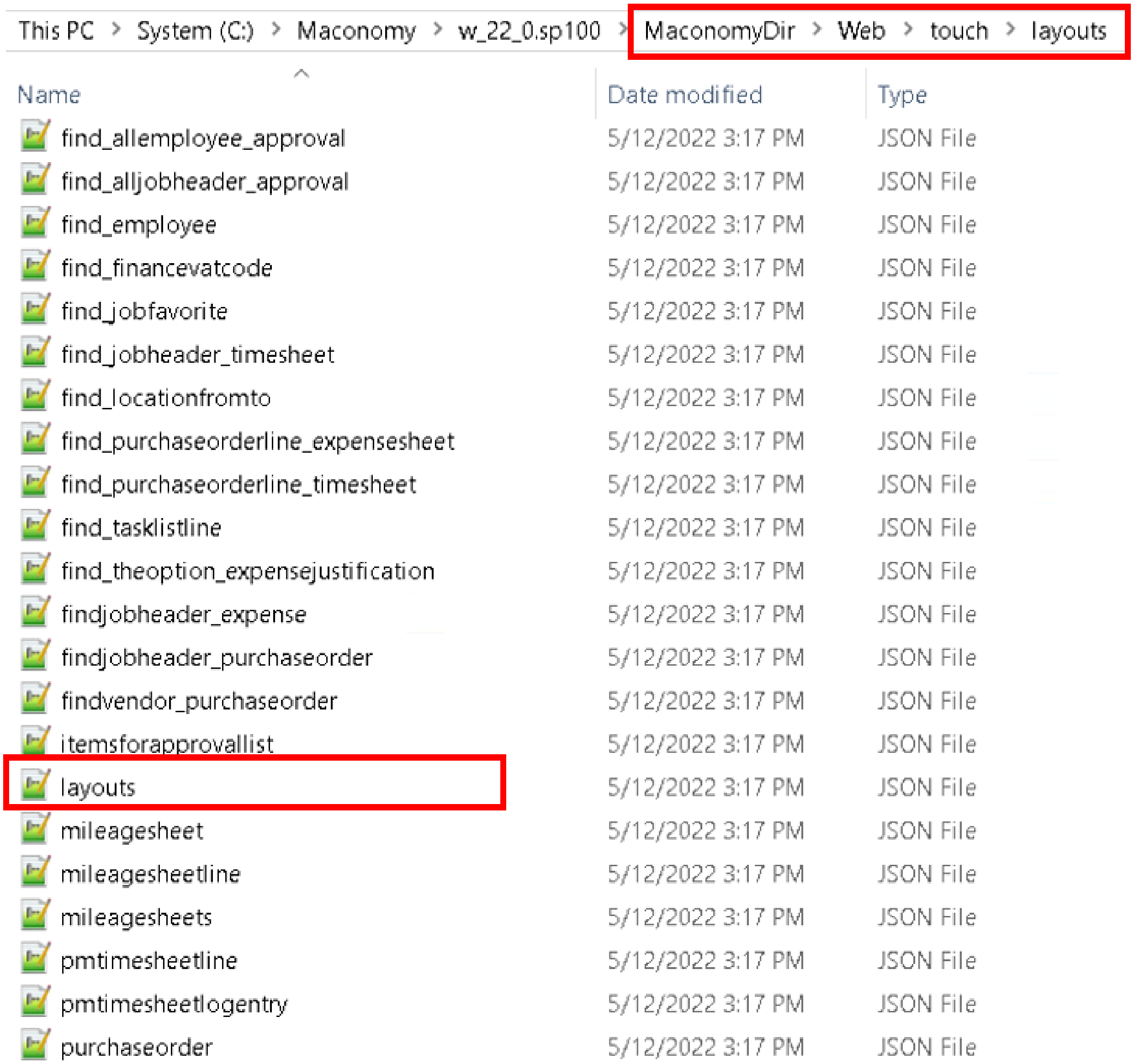
- Customize search layouts (for example: Find Job, Task List, and Mileage From)
- Have different layouts for weekly and daily timesheets
- Customize more in some screens (for example: Weeks, Calendar, and Summary)

Layout List File

The **layouts.JSON** file includes the list of layouts used by the Touch app in alphabetical order.

> **Note:** The layout list file and layouts are saved in the same location.

The released Deltek Touch product includes standard layouts and corresponding standard layout list file:


This PC > System (C:) > Maconomy > w_22_0.sp100 > MaconomyDir > Web > touch > layouts
Name
Date modified
Type
find_allemployee_approval 5/12/2022 3:17 PM JSON File
find_alljobheader_approval 5/12/2022 3:17 PM JSON File
find_employee 5/12/2022 3:17 PM JSON File
find_financevatcode 5/12/2022 3:17 PM JSON File
find_jobfavorite 5/12/2022 3:17 PM JSON File
find_jobheader_timesheet 5/12/2022 3:17 PM JSON File
find_locationfromto 5/12/2022 3:17 PM JSON File
find_purchaseorderline_expensesheet 5/12/2022 3:17 PM JSON File
find_purchaseorderline_timesheet 5/12/2022 3:17 PM JSON File
find_tasklistline 5/12/2022 3:17 PM JSON File
find_theoption_expensejustification 5/12/2022 3:17 PM JSON File
findjobheader_expense 5/12/2022 3:17 PM JSON File
findjobheader_purchaseorder 5/12/2022 3:17 PM JSON File
findvendor_purchaseorder 5/12/2022 3:17 PM JSON File
itemsforapprovallist 5/12/2022 3:17 PM JSON File
layouts 5/12/2022 3:17 PM JSON File
mileagesheet 5/12/2022 3:17 PM JSON File
mileagesheetline 5/12/2022 3:17 PM JSON File
mileagesheets 5/12/2022 3:17 PM JSON File
pmtimesheetline 5/12/2022 3:17 PM JSON File
pmtimesheetlogentry 5/12/2022 3:17 PM JSON File
purchaseorder 5/12/2022 3:17 PM JSON File

The layout list file contains an array of objects. Each object is a reference to a layout file as shown in the following figure.

layouts.json

```
[
    {
        "$ref": "absenceapprovalform"
    },
    {
        "$ref": "absenceapprovallist"
    },
    {
        "$ref": "absencerequest"
    },
    {
        "$ref": "absencerequests"
    },
    {
        "$ref": "allowanceapprovalform"
    },
    {
        "$ref": "allowanceapprovallist"
    },
    {
        "$ref": "allowanceapproveinputscreen"
    },
    {
        "$ref": "allowancerequest"
    },
    {
        "$ref": "allowancerequests"
    },
    {
        "$ref": "approvalcheckin"
    },
    {
        "$ref": "approvalexpensesheet"
    },
    {
        "$ref": "approvalexpensesheetline"
    },
    {
        "$ref": "approvalexpensesheetlinepm"
    },
```

If you need to add a new layout (for example, search layout), add the corresponding reference to the list.

Attention: You must not change the standard files located in MaconomyDir. Instead, you should copy the files to the CustomazitionDir folder (for example, **/CustomizationDir/Custom/Web/touch /layouts/layouts.JSON**).

Layout File Matrix

This section maps the old .I REST layout files to the new .JSON layout files.

Time Registration

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
TimesheetLayoutREST.I TimesheetDaysLayoutREST.I	timesheet.JSON timesheetdays.JSON
TimesheetLogEntryWeeklyLayoutREST.I TimesheetLogEntryDailyLayoutREST.I	timesheetlogentryweekly.JSON timesheetlogentrydaily.JSON
SummaryLayoutREST.I SummaryDailyLayoutREST.I	summary.JSON summarydaily.JSON
RejectedTimesheetsLayoutREST.I	rejectedtimesheets.JSON
TimeSheetWeekLayoutREST.I	timesheetweek.JSON
TimeSheetCalendarLayoutREST.I	timesheetcalendar.JSON
Find_JobHeader.I	find_jobheader.JSON
Find_TaskListLine.I	find_tasklistline.JSON
Find_JobFavorite_Timesheet	find_jobfavorite_timesheet.JSON
Find_JobHeader_Timesheet.I	find_jobheader_timesheet.JSON
Find_Activity.I	find_activity.JSON

Expense & Mileage Registration

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
ExpenseSheetsLayoutREST.I	expensesheets.JSON
ExpenseSheetLayoutREST.I	expensesheet.JSON
ExpenseSheetLineLayoutREST.I	expensesheetline.JSON
QuickCaptureExpenseSheetLineLayoutREST.I	quickcaptureexpensesheetline.JSON
QuickCaptureAttachReceiptLayoutREST.I	quickcaptureattachreceipt.JSON
MileageSheetsLayoutREST.I	mileagesheets.JSON

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
MileageSheetLayoutREST.I	mileagesheet.JSON
MileageSheetLineLayoutREST.I	mileagesheetline.JSON
Find_LocationFromTo.I	find_locationfromto.JSON
FindJobHeader_Expense.I	findjobheader_expense.JSON
DocumentArchiveListREST.I	documentarchivelist.JSON
Find_FinanceVATCode.I	find_financevatcode.JSON
Find_TheOption_ExpenseJustification.I	find_theoption_expensejustification.JSON

Approvals

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
PmTimesheetLogEntryLayoutREST.I	pmtimesheetlogentry.JSON
SupervisorTimesheetLogEntryLayoutREST.I	supervisortimesheetlogentry.JSON
ApprovalPurchaseOrderLayoutREST.I	approvalpurchaseorder.JSON
ApprovalPurchaseOrderLineLayoutREST.I	approvalpurchaseorderline.JSON
ApprovalPurchaseOrderPMLayoutREST.I	approvalpurchaseorderpm.JSON
ApprovalPurchaseOrderLinePMLayoutREST.I	approvalpurchaseorderlinepm.JSON
ApprovalVendorInvoiceListLayoutREST.I	approvalvendorinvoicelist.JSON
ApprovalVendorInvoiceLayoutREST.I	approvalvendorinvoice.JSON
ApprovalVendorInvoiceAllocationLineLayoutREST.I	approvalvendorinvoiceallocationline.JSON
ApprovalVendorInvoiceAllocationLinePMLayoutREST.I	approvalvendorinvoiceallocationlinepm.JSON
ApprovalVendorInvoicePMLayoutREST.I	approvalvendorinvoicepm.JSON
ApprovalInvoiceDraftLineLayoutREST.I	approvalinvoicedraftline.JSON
ApprovalInvoiceDraftLinesLayoutREST.I	approvalinvoicedraftlines.JSON

Attention: The REST approval layouts are used only if the Maconomy system has long text enabled. For more information, see Appendix A: Screens, Layouts, & Containers.

Search Layouts

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
Find_AbsenceType.I	find_absencetype.JSON
Find_Account.I	find_account.JSON
Find_Activity.I	find_activity.JSON
Find_Employee.I	find_employee.JSON
Find_JobFavorite_Timesheet.I **Note:** Starting Touch 3.6 API, The Find_JobFavorite_Timesheet.I is replaced by Find_JobFavorite.I. If you want to make changes specifically for timesheets, create a new layout file and add it to the LayoutList_REST.I. For example: ▪ **Timesheet** Filename: Find_JobFavorite_Timesheet.I Layoutname: #N"find_jobfavorite_timesheet" ▪ **Expense** Filename: Find_JobFavorite_Expensesheet.I Layoutname: #N"find_jobfavorite_expensesheet" ▪ **Mileage** Filename: Find_JobFavorite_Mileagesheet.I Layoutname: #N"find_jobfavorite_mileagesheet"	find_jobfavorite_timesheet.JSON **Note:** If you want to make changes specifically for timesheets, create a new layout file and add it to the layouts.JSON. For example: ▪ **Timesheet** Filename: find_jobfavorite_timesheet.JSON Layoutname: #N"find_jobfavorite_timesheet" ▪ **Expense** Filename: find_jobfavorite_expensesheet.JSON Layoutname: #N"find_jobfavorite_expensesheet" ▪ **Mileage** Filename: find_jobfavorite_mileagesheet.JSON Layoutname: #N"find_jobfavorite_mileagesheet"
Find_JobHeader_Timesheet.I	find_jobheader_timesheet.JSON
Find_LocationFromTo.I	find_locationfromto.JSON

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
Find_PurchaseOrderLine_ExpenseSheet.I	find_purchaseorderline_expensesheet.JSON
Find_PurchaseOrderLine_TimeSheet.I	find_purchaseorderline_timesheet.JSON
Find_TaskListLine.I	find_tasklistline.JSON
FindJobHeader_Expense.I	findjobheader_expense.JSON
FindJobHeader_PurchaseOrder.I	findjobheader_purchaseorder.JSON
FindVendor_PurchaseOrder.I	findvendor_purchaseorder.JSON
Find_FinanceVATCode.I	find_financevatcode.JSON

Purchase Order Layouts

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
PurchaseOrderLayoutREST.I	purchaseorder.JSON
PurchaseOrderLineLayoutREST.I	purchaseorderline.JSON
PurchaseOrdersLayoutREST.I	purchaseorders.JSON
RejectionNotificationListLayoutREST.I	rejectionnotificationlist.JSON

Absence and Allowance Requests Layouts

REST Touch 3.8 (and earlier)	Touch 4.0 (and later)
AbsenceApprovalFormLayoutREST.I	absenceapprovalform.JSON
AbsenceApprovalListLayoutREST.I	absenceapprovallist.JSON
AbsenceRequestLayoutREST.I	absencerequest.JSON
AbsenceRequestsLayoutREST.I	absencerequests.JSON
AllowanceRequestLayoutREST.I	allowancerequest.JSON
AllowanceRequestsLayoutREST.I	allowancerequests.JSON

Note: For more information, see Appendix A: Screens, Layouts, & Containers.

Layout Components & Attributes

This section discusses the non-editable and editable layout components and corresponding attributes used in customization of Deltek Touch.

Non-Editable Layout Components

Use the non-editable layout components to display a non-editable text on a screen or non-Sencha or non-HTML5 component related. It generates HTML tags-based layouts.

Component	Description
Label	Basic component in the layout. It is responsible for displaying a caption or title on the screen.
Field	This component is used to display a field from the database (using the container) on the screen.
LinearText	This component is used for grouping components, such as “Label” and “Field”. If you want to display a group of components in one line, you need to use this component.
Grid	This component is used for tabulating the data (in a columns and rows).
Column	This component used to define column in the grid component.

Editable Layouts Components

Use the editable layout components to display a server-variable controlled editable field. These components are only applicable to all Line screens, except for the **Add Job** and **Timesheet Line** screens.

Component	Description
TextField	This component is used to display a text field on a Line screen (see Figure 1).
DocumentField	This component is used to display a field on the Line screen intended for attachments. You need to define the “DocumentName” as the source field not the “DocumentArchiveNumber” field.
Form	This component is used mainly for “Line Screens” (for example, the **Log Entry** screen).
Group	This component creates a field set to group the “textfield” and “documentfield” components.

Component Layout Attributes

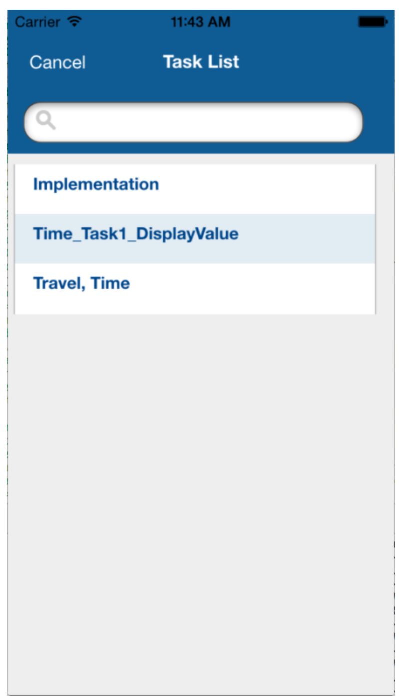
The following table describes component attributes, but not all of them can be used in a specific component:

Attribute	Description
type	This is used to specify which component to use in a layout. All the layout files already contain the component that you can use in customizing a screen. This is one of the most used attributes in the layouts.
region	This is used to state the name of the portion of a screen.
protectedregion	This is used to identify whether the portion of the screen in the Touch client is customizable or not.
style	This is used to describe the formatting or styling that needs to be applied in a specific component.

cls	The cls (short term for class) attribute is used to format the layout of pages. It defines text styles, columns, and other aspects of pages, providing an alternative means of defining the style of each block of text within a page. Once the style is defined in cls, it can be used by any page that references the CustomLayout.scss file.
	
	Sample code:
	`{`
	`"type": "lineartext",`
	`"cls":"text_medium_bold"`
	`}`

purpose	You can use this to state the whole description of a certain component or region in the layout file.
detail	This is only used for “grid” component. This attribute represents the content of grid. It refers to “row” in the grid.
header	This is only used for “grid” component. This attribute is used for describing the header title in the grid; it refers to the “column title” in the grid. Not all “grid” have a header component. The layout file already contains a header if the grid can have a “detail” or a “header.” This is dependent on the Sencha component used in the Touch client.
items	This is only used for the “lineartext” and “column” components. It is a container for the other component.
mask	This is only used in the “field” and “textfield” component. The purpose of this attribute is the following: ▪ If the field value needs to use the device short-date formatting.

Attribute	Description
	▪ If the field value is not informative but predictable (for example, Status) and you want to set a descriptive value when it displays on a screen. ▪ If the field value needs to use the Touch Standard hours formatting (for example, 2:30). ▪ If the requirements need to display the amount and currency value (for example, DKK 200.00). ▪ If the field needs to show the “Untitled” string, instead of empty string.

lookuplayoutname	This is only used for customizing the layout and search field of the lookup. You can customize the following components in the lookup:
	▪ **Search Field Area:** This area is composed of the text field component with a magnifying glass icon (see screenshot below). By entering characters in this component, it filters the records in the list.
	▪ **Lookup List Area:** This area is composed of the list component used for displaying the records that are retrieved from the Web service (see the list in the screenshot below).
	Using this attribute in the lookup field component is pointing to another layout file. You need to add this to the layouts.JSON file.
	
	
	Sample code:
	Adding the “lookuplayoutname” to the lookup field component:
	`{`
	`"source": "Deliverable",`
	`"T$label": "Remark",`

Attribute	Description

	`"searchlayout": "ActivitySearch",`
	`"type": "lookupfield",`
	`"queryfieldname": "KeyValue",`
	`"displayfields": [`
	`{`
	`"source": "ActivityTextVar",`
	`"queryfieldname": "DisplayValue"`
	`},`
	`{`
	`"label": " - ",`
	`},`
	`{`
	`"source": "ActivityNumber",`
	`"queryfieldname": "DisplayValue"`
	`}`
	`]`
	`"Lookuplayoutname": "DeliverableLookupLayout"`
	`}`
	
	Adding the layout to the layouts.JSON file:
	`{`
	`"$ref": "DeliverableLookupLayout"`
	`},`
	
	Content of a Lookup Layout File:
	`{`
	`"Layoutname": "DeliverableLookupLayout",`
	`"Layout": [`
	`{ "region": "ToolbarArea",`
	`"protectedregion": true`
	`},`
	`{`
	`"region": "SearchFieldArea",`
	`"protectedregion": false,`
	`"type": "searchfield"`
	`"filterkeys": "KeyValue, DisplayValue",`
	`"T$placeholder": "Task Name, Task Description"`

<table>
<tr><th>Attribute</th><th>Description</th></tr>
<tr><td></td><td>
<pre>
      },
      {
           "region": "LookupListArea",
           "protectedregion": false,
           "type": "grid",
           "detail":   [
                  {
                       "type": "column",
                       "style": { "width": "75%" },
                       "items": [
                              {
                                  "type": "lineartext",
                                  "style": {
                                     "fontsize": "14px",
                                     "color": "00488a",
                                     "fontweight": "bold"
                                     },
                                   "items":   [
                                         {
                                      "type": "field",
                                      "source": "KeyValue"
</pre>
The content of the lookup layout file contains three regions:

ToolbarArea: This is a protected region.
SearchFieldArea: This represents the text field component in the lookup.
LookupListArea: This represents the list component of the lookup.

Adding this attribute to the “lookupfield” component only works if the Web method used by the lookup to retrieve the data is the “search” operation or the lookup uses the search layout from DeltekTouch.I.

New layout attribute syntax:
<pre>
       "options" : {
          "restriction": "samplequery = 1"
       }
</pre>
</td></tr>
<tr><td>restriction</td><td>This is used by the lookup component, specifically for REST-enabled system. The purpose of this attribute is to dynamically filter the data coming from the response.

Sample code:
<pre>
            {
</pre>
</td></tr>
</table>

<table>
<tr><th>Attribute</th><th>Description</th></tr>
<tr><td></td><td><pre>
"T$label": "SelectedOption2 - County",
"source": "selectedoption2",
"type": "lookupfield",
"queryfieldname": "name",
"options": {
 "restriction": "remark1 = selectedoption1"
},
"displayfields": [
  {
    "source": "selectedoption2",
    "queryfieldname": "name"
  }
]
}
</pre></td></tr>
<tr><td>Mandatory</td><td>The purpose of this attribute is to require the user to put a value in a certain field. REST-enabled system can use this attribute.
Sample code:<pre>
{
    "source": "NumberOf",
    "T$label": "Quantity",
    "mandatory": "true"
}
</pre>This does not apply to read-only and disabled fields.</td></tr>
</table>

Style Layout Attributes

The following table contains the style and formatting attributes in a layout. All of the styling attributes that you can define in the layout are derived from “CSS” technology, including how the styles are applied to a certain component.

<table>
<tr><th>Attribute</th><th>Description</th></tr>
<tr><td>fontsize</td><td>This is used to define the size of a font. It accepts either in “px” (by pixel) or “em” measurement unit.
Sample code:<pre>
"style": {
"fontsize": "12px"
}
</pre></td></tr>
</table>

Attribute	Description
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_font_font-size.asp.

textdecoration	This is usually used to add an underline/overline in the text.
	Sample code:
	`"style": {`
	`"textdecoration": "underline"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_text_text-decoration.asp.

textalignment	This is used to define the alignment of a text. You can set one of the following common values in this attribute:
	▪ "right"
	▪ “left”
	▪ “center”
	Sample code:
	`"style": {`
	`"textalignment": "left"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_text_text-align.asp.

fontstyle	This is used to define the style you want to apply to a text. The common value you can set in this attribute is “italic.”
	Sample code:
	`"style": {`
	`"fontstyle": "italic"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_font_font-style.asp.

fontweight	This is used to define how thick or thin the text will be displayed on a screen. The common value you can set is **bold**.
	Sample code:
	`"style": {`
	`"fontweight": "bold"`

Attribute	Description
	`}` **Attention:** For more details about the value you can define, see http://www.w3schools.com/cssref/pr_font_weight.asp.

margintop	This is used to define the top margin of a component on a screen. This attribute usually accepts value in “px” (by pixel) measurement unit.
	Sample code:
	`"style": {`
	`"margintop": "12px"`
	`}`
	Attention: For more details about the value you can define, see
	http://www.w3schools.com/cssref/pr_margin-top.asp.

marginbottom	This is used to define the bottom margin of a component on a screen. This attribute usually accepts value in “px” (by pixel) measurement unit.
	Sample code:
	`"style": {`
	`"marginbottom": "12px"`
	`}`
	Attention: For more details about the value you can define, see
	http://www.w3schools.com/cssref/pr_margin-bottom.asp.

marginleft	This is used to define the left margin of a component on a screen. This attribute usually accepts value in “px” (by pixel) measurement unit.
	Sample code:
	`"style": {`
	`"marginleft": "12px"`
	`}`
	Attention: For more details about the value you can define, see
	http://www.w3schools.com/cssref/pr_margin-left.asp.

marginright	This is used to define the right margin of a component on a screen. This attribute usually accepts value in “px” (by pixel) measurement unit.
	Sample code:
	`"style": {`
	`"marginright": "12px"`
	`}`
	Attention: For more details about the value you can define, see

Attribute	Description
	http://www.w3schools.com/cssref/pr_margin-right.asp.

textoverflow	This is used to define the result if the text overflows in a container. The common scenario is that, if the field value you are expecting is too long and does not fit into the screen or column width, you can truncate the text by specifying an "ellipsis" (for example, **This text is too long...**).
	Sample code:
	`"style": {`
	`"textoverflow": "ellipsis"`
	`}`
	Sometimes, however, if you define an ellipsis styling, it does not apply automatically to the screen. The usual reason is that there is some external styling attribute that hinders it. To apply the styling successfully, you can add another styling attribute, which is the "overflow." The most common value you can set on this is "hidden."
	Sample code:
	`"style": {`
	`"textoverflow": "ellipsis",`
	`"overflow": "hidden"`
	`}`
	Attention: For more details about the value you can define, see
	http://www.w3schools.com/cssref/css3_pr_text-overflow.asp.

paddingtop	This is used to create padding at the top side of a component. This attribute usually accepts value in "px" (by pixel) measurement unit and does not accept any negative value.
	Sample code:
	`"style": {`
	`"paddingtop": "16px",`
	`}`
	Attention: For more details about the value you can define, see
	http://www.w3schools.com/cssref/pr_padding-top.asp.

paddingleft	This is used to create padding on the left side of a component. This attribute usually accepts value in "px" (by pixel) measurement unit and does not accept any negative value.
	Sample code:
	`"style": {`
	`"paddingleft": "16px",`

Attribute	Description
	} **Attention:** For more details about the value you can define, see http://www.w3schools.com/cssref/pr_padding-left.asp.

paddingright	This is used to create padding on the right side of a component. This attribute usually accepts value in "px" (by pixel) measurement unit and does not accept any negative value.
	 Sample code:
	 `"style": {`
	 `"paddingright": "16px",`
	 `}`
	 Attention: For more details about the value you can define, see
	 http://www.w3schools.com/cssref/pr_padding-right.asp.

paddingbotom	This is used to create padding at the bottom side of a component. This attribute usually accepts value in "px" (by pixel) measurement unit and does not accept any negative value.
	 Sample code:
	 `"style": {`
	 `"paddingbottom": "16px",`
	 `}`
	 Attention: For more details about the value you can define, see
	 http://www.w3schools.com/cssref/pr_padding-bottom.asp.

whitespace	This is used to control how the white space inside the element will be handled.
	 Sample code:
	 `"style": {`
	 `"whitespace": "nowrap"`
	 `}`
	 Attention: For more details about the value you can define, see
	 http://www.w3schools.com/cssref/pr_text_white-space.asp.

backgroundcolor	This is used for applying a background color of an element.
	 Sample code:
	 `"style": {`
	 `"backgroundcolor": "#00FF00"`
	 `}`

Attribute	Description
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_background-color.asp.

backgroundimage	This is use for applying background images of an element.
	Sample code:
	`"style": {`
	`"backgroundimage": "-webkit-linear-gradient(top, #ff1a1a,#e60000 3%,#b30000)"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_background-image.asp.

borderradius	This is used for applying border radius of an element.
	Sample code:
	`"style": {`
	`"borderradius": "2em"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/css3_pr_border-radius.asp.

bordercolor	This is used for applying border color of an element.
	Sample code:
	`"style": {`
	`"bordercolor": "#ff0000 #0000ff"`
	`}`
	The border color will apply in the top and right border of an element.
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_border-color.asp.

webkitbackgroundclip	This is used to specify the painting area of the background.
	Sample code:
	`"style": {`
	`"webkitbackgroundclip": "padding-box"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/css3_pr_background-clip.asp.

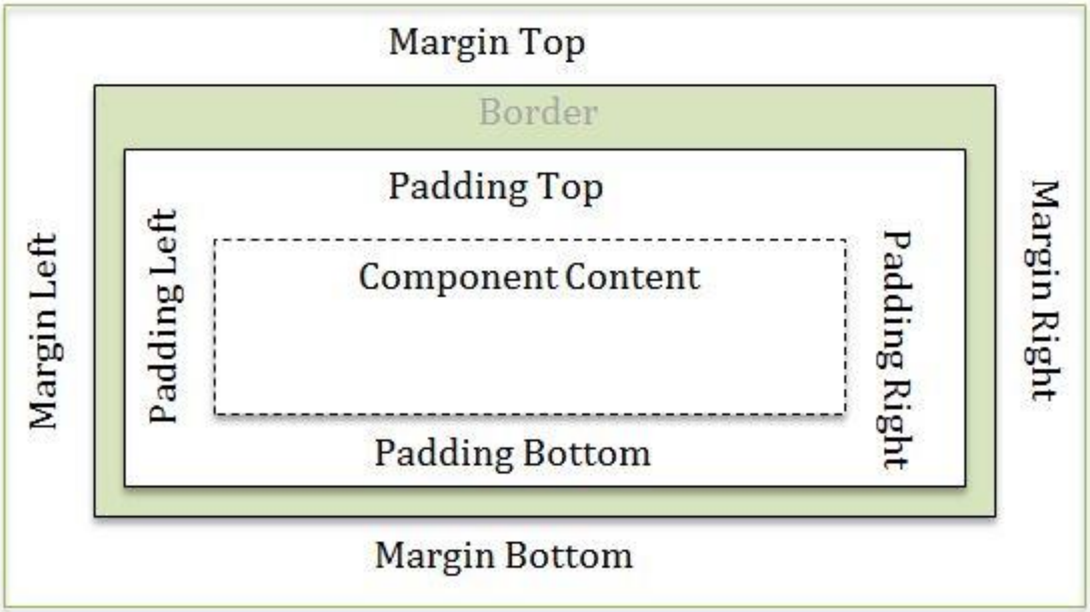
Attribute	Description

minwidth	This is used to specify the minimum width of an element.
	Sample code:
	`"style": {`
	`"minwidth": "10px"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_dim_min-width.asp.

maxwidth	This is used to specify the maximum width of an element.
	Sample code:
	`"style": {`
	`"maxwidth": "10px"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_dim_max-width.asp.

action	This is used to set the action for the link label.
	Sample code:
	`"type": "lineartext",`
	`"cls": "link_label_margintop30",`
	`"action": "open_expensesheet_header",`
	`"items": {`
	`"T$title": "Expense Sheet",`
	`"type": "linklabel"`
	`}`
	Attention: For more details about the value you can define, see http://www.w3schools.com/cssref/pr_dim_max-width.asp.

Margin and Padding Box Model



Class Layout Attributes

The following table contains the style and formatting attributes used in a Custom layout.

Attribute	Description

CssClass	Object that is used to classify a custom style. It is placed in the CssClass.i file and defined by Maconomy to minify style from the layout. In addition, it is helpful in formatting other aspects of the page layout.
	This new attribute of Maconomy gives you an option to overrule how a field looks.
	Sample code:
	`{"CssClass":[`
	`{`
	`"className": "text_medium_bold",`
	`"style": {`
	`"fontsize": "16px!important",`
	`"textalignment": "left",`
	`"color": "#f2b600"`
	`}`
	`}`
	`]}`

Attribute	Description

className	Under CssClass object is the ClassName, which is used to define the cls that the user wants to update.
	Sample code:
	`{`
	`"className": "text_medium_bold",`
	`"style": {`
	`"fontsize": "16px!important",`
	`"textalignment": "left",`
	`"color": "#f2b600"`
	`}`
	`}`

CustomLayout.scss

To improve the way Deltek Touch uses styles in layouts, you can use the CustomLayout.scss CSS style. Using the customLayout file minifies the layout file size from the standard folder.

Field Types

The following table displays the CSS standard layout that you can customize:

LinearText	Column	Fields

`text_lookup{`
`color: #00488a;`
`}`
`.text_center{`
`text-align: center;`
`}`
`.text_right{`
`text-align:right;`
`}`
`.text_left{`
`text-align:left;`
`}`
`.text_upper{`
`text-transform:uppercase;`
`}`
`.text_bold{`
`font-weight: bold;`
`}`
	`.columntext_15width_right{`
	`margin-left: 15px!important;`
	`text-align:right;`
	`}`
	`.columntext_right_30right{`
	`text-align:right;`
	`padding-right: 30px;`
	`}`
	`.columntext_left_20left{`
	`text-align:left;`
	`padding-left: 20px;`
	`}`
	`.columntext_20left{`
	`padding-left: 20px;`
	`}`
	`.columntext_right{`
	`text-align:right;`
	`}`
		`.field_border_radius{`
		`font-size:16px;`
		`height: 30px;`
		`width:30px;`
		`background-color:#cc3300;`
		`border-radius: 50%;`
		`display: inline-block;`
		`color: #ffffff;`
		`line-height: 28px;`
		`text-align: center;`
		`}`
		`.field_medium{`
		`font-size:14px;`
		`}`
		`.field_large_bold_30height {`
		`font-size: 16px;`
		`font-weight: bold;`
		`color: #434E5E;`

<table>
<tr><th>LinearText</th><th>Column</th><th>Fields</th></tr>
<tr>
<td>.text_marker{
color: #f2b600;
}
.text_marker_bold{
color: #f2b600;
font-weight: bold;
}
.text_required{
color: #C71B05;
}
.text_ellipsis {
white-space:nowrap;
text-overflow:ellipsis;
overflow: hidden;
}
.text_xxsmall_comment{
font-size: 9pt;
color: #f2b600;
padding: 10px 10px 0px;
}
.text_xxsmall_comment_5left{
font-size: 9pt;
color: #f2b600;
padding-left: 5px;
}
.text_xxsmall_nowrap{
font-size: 9pt;
white-space: nowrap;
}
.text_xxsmall_right{
font-size: 9pt;
text-align: right;
}
.text_xxsmall_nowrap_8left{
font-size: 9pt;
color: #475262;
white-space: nowrap;
padding-left: 8px;
}
.text_xxsmall_bold_right{
font-size: 9pt;
color: #475262;</td>
<td>.columntext_center{
text-align:center;
}
.columntext_center_20perwidth{
text-align:center;
width: 20%;
}
.columntext_small{
font-size: 12px;
}
.columntext_small_color{
font-size: 12px;
color: #475262;
}
.columntext_small_10bottom{
font-size: 12px;
margin-bottom: 10px;
}
.columntext_small_right{
font-size: 12px;
text-align:right;
}
.columntext_small_padding{
font-size: 12px;
padding-left: 35px;
}
.columntext_small_35perwidth{
font-size: 12px;
width: 35%;
}
.columntext_small_35perwidth_right{
font-size: 12px;
width: 35%;
text-align:right;
}
.columntext_small_20perwidth{
font-size: 12px;
width: 20%;
}
.columntext_small_20perwidth_right{
font-size: 12px;</td>
<td>line-height: 30px;
}
.margin_bottom10{
margin-bottom: 10px;
}</td>
</tr>
</table>

LinearText	Column	Fields

font-weight: bold;
text-align: right;
}
.text_xsmall{
font-size: 10px;
}
.text_xsmall_status{
font-size: 10px;
color: #a00;
}
.text_xsmall_marker{
font-size: 10px;
color: #f2b600;
}
.text_small{
font-size: 12px;
}
.text_small_lookup{
font-size: 12px;
color: #00488a;
}
.text_small_note{
font-size: 12px;
color: #6b6b6b;
}
.text_small_note_50left{
font-size: 12px;
color: #6b6b6b;
padding-left: 50px;
}
.text_small_wrap{
font-size: 12px;
color: #475262;
}
.text_small_nowrap{
font-size: 12px;
color: #475262;
white-space: nowrap;
}
.text_small_8right_right{
font-size: 12px;
color: #475262;
padding-right: 8px;
text-align:right;
}
	width: 20%;
	text-align:right;
	}
	.columntext_small_center{
	font-size: 12px;
	text-align:center;
	color: #434E5E;
	}
	.columntext_small_dark{
	font-size: 12px;
	color: #434E5E;
	}
	.columntext_small_center_1
	5perwidth{
	font-size: 12px;
	text-align:center;
	color: #434E5E;
	width: 15%;
	}
	.columntext_small_center_2
	5perwidth{
	font-size: 12px;
	text-align:center;
	color: #434E5E;
	width: 25%;
	}
	.columntext_small_center_2
	5perwidth{
	font-size: 12px;
	text-align:center;
	color: #434E5E;
	width: 25%;
	}	

LinearText	Column	Fields

.text_medium{
font-size: 14px;
color: #475262;
}
.text_medium_right{
font-size: 14px;
color: #475262;
text-align:right;
}
.text_medium_bold{
font-size: 14px;
font-weight: bold;
color: #475262;
}
.text_medium_bold_right{
font-size: 14px;
font-weight: bold;
color: #475262;
text-align:right;
}
.text_medium_bold_lookup{
font-size: 14px;
font-weight: bold;
color: #00488a;
}
.text_medium_bold_35perwidth{
font-size: 14px;
font-weight: bold;
color: #475262;
width: 35%;
}
.text_medium_bold_required{
color: #C71B05;
font-size: 14px;
font-weight: bold;
}
.text_medium_bold_dark{
font-size: 14px;
font-weight: bold;
color: #434E5E;
}
.text_medium_dark_right{
font-size: 14px;
color: #434E5E;		

LinearText	Column	Fields

`text-align:right;`
`}`
`.text_medium_bold_dark_right{`
`font-size: 14px;`
`font-weight: bold;`
`color: #434E5E;`
`text-align:right;`
`}`
`.text_medium_bold_dark_15right{`
`font-size: 14px;`
`font-weight: bold;`
`color: #434E5E;`
`margin-right: -15px!important;`
`}`
`.text_large_bold{`
`font-size: 16px;`
`font-weight: bold;`
`color: #475262;`
`}`		

Rules

- You can only override style based from CustomLayout.scss.
- When defining a style, always declare the "className" attribute first before the style attribute.
- After adding the CssClass.l file, make sure the file is in the first line in the LayoutList_MScript.l file.

> **Attention:** For more information, see Change the Style.

Example

The following sample code shows how to change the color:

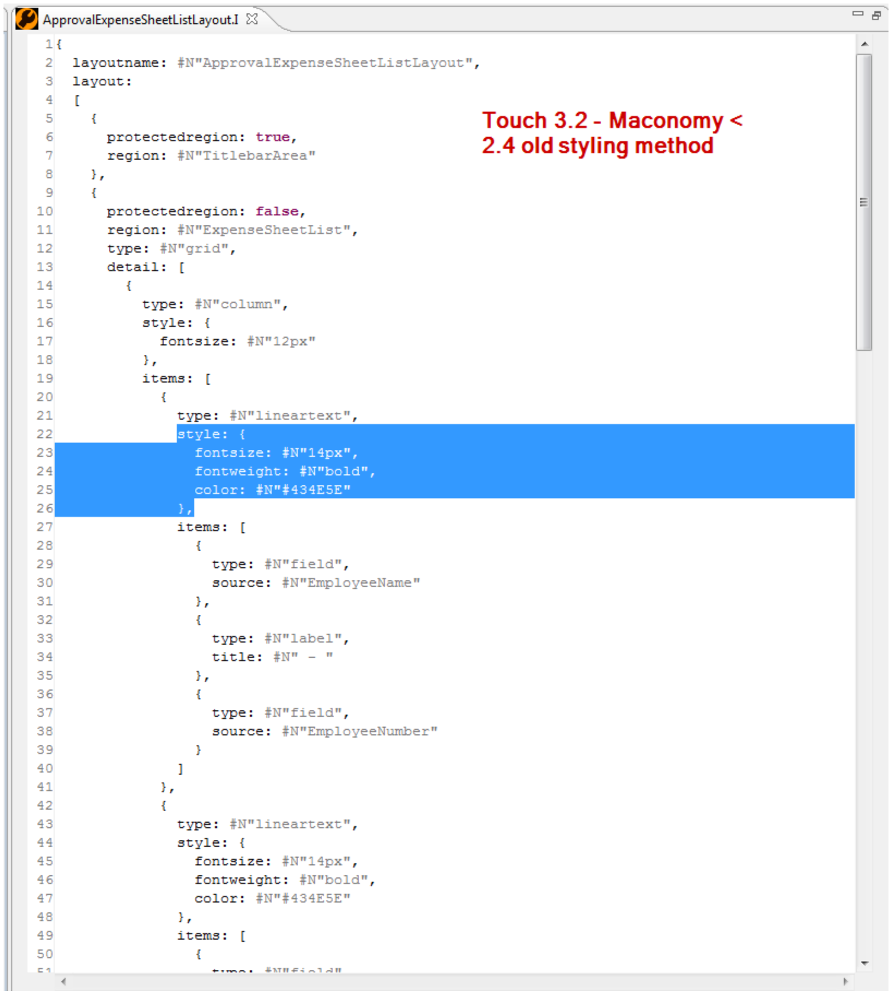
```
{"CssClass":[
        {
                "className": "text_medium_bold",
                "style": {
                        "fontsize": "16px!important",
                        "textalignment": "left",
                        "color": "#f2b600"
                }
```

```
        }
] }
```

Old and New Styling Methods

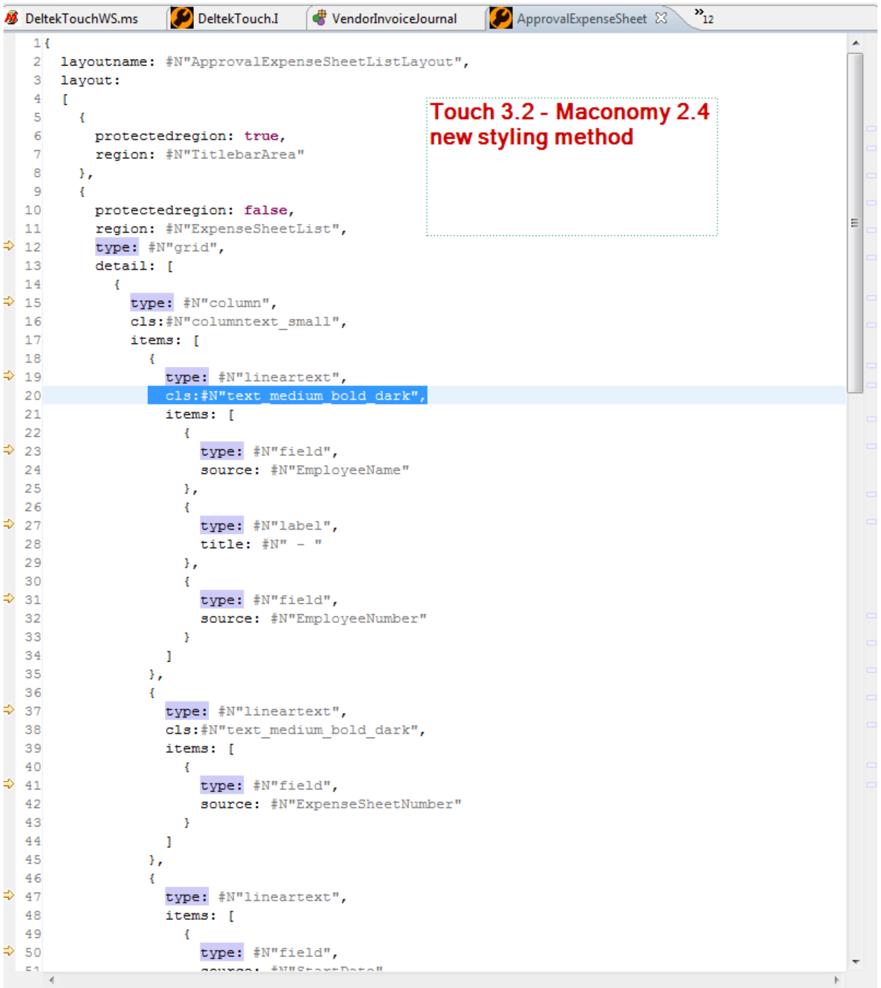
The purpose of this section is to provide a comparison between the two styling methods using sample codes.

Old Styling Method ("Style" Layout Attribute):



> **Note:** You can only use the old styling method in all Touch layouts if you are on a Maconomy version lower than 2.4.4.

New Styling Method (“cls” Layout Attribute)



Note: You can only use the new styling method in all Touch layouts if you are on Maconomy version 2.4.4 or higher.

Actions

This section discusses the **actions** section in your layouts. The **actions** section includes a list of actions which have the following attributes:

- **Key:** This attribute is the button ID, and is not included in the action REST request. You should not change this attribute for standard actions. However, for new custom actions, you must specify a unique string value within the keys used for the actions in the layout.
- **Hidden:** This attribute dictates whether the action is displayed on the screen or not. Valid options are **true** or **false**.
- **Title:** This attribute is the label of the action that is displayed on the screen, and is not included in the action REST request.
- **Parameters:** These are the action parameters (for example, `card.calculateperiodinfovar=true&card.datevar=2020-11-30`). Parameters specified in

the layout should be appended to the parameters hardcoded in the client code, if any. This is added to the action REST request after “?”.

- **Source:** This attribute is the internal name (for example, action:SubmitTimeSheet), and is added to the action REST request.

The following table lists the Touch layouts, and the corresponding Touch and Maconomy actions:

Layout	Touch Action	Maconomy Action
absenceRequest	▪ submit ▪ reopen	▪ SubmitAbsenceEntry ▪ ApproveAbsenceEntry ▪ RejectAbsenceEntry ▪ ReopenAbsenceEntry
allowanceRequest	▪ submit ▪ reopen	▪ SubmitAbsenceCalendarLine ▪ ApproveAbsenceCalendarLine ▪ RejectAbsenceCalendarLine ▪ ReopenAbsenceCalendarLine
expensesheet	▪ submit ▪ reopen ▪ newline ▪ duplicate ▪ attach	▪ SubmitExpenseSheet ▪ ReleaseExpenseSheet ▪ ApproveExpenseSheet ▪ ReopenExpenseSheet ▪ CopyExpenseSheet ▪ DuplicateExpenseSheet ▪ AttachDocument ▪ AttachDocumentToHeaderAndLine ▪ RemoveDocument
expensesheetline	▪ newline ▪ map ▪ attach	▪ CreateJobFavorite ▪ AttachDocumentToLine ▪ ShowDocument
mileagesheet	▪ submit ▪ reopen ▪ newline ▪ duplicate	▪ SubmitExpenseSheet ▪ ReleaseExpenseSheet ▪ ApproveExpenseSheet ▪ ReopenExpenseSheet ▪ CopyExpenseSheet ▪ DuplicateExpenseSheet ▪ AttachDocument ▪ AttachDocumentToHeaderAndLine

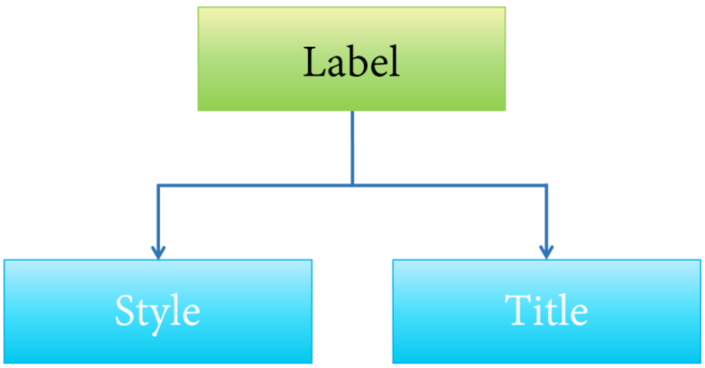
Layout	Touch Action	Maconomy Action
		▪ RemoveDocument
mileagesheetline	▪ newline ▪ map	▪ CreateJobFavorite ▪ AttachDocumentToLine ▪ ShowDocument
purchaseorder	▪ newline ▪ submit ▪ reopen ▪ close ▪ attach	▪ SubmitPurchaseOrder ▪ ApprovePurchaseOrder ▪ ClosePurchaseOrder ▪ ReopenPurchaseOrder ▪ CopyPurchaseOrder ▪ CreateSelfInvoice ▪ AttachDocument
purchaseorderline	▪ Newline	–
summarydaily	▪ submit ▪ reopen	▪ CreateTimeSheet ▪ SubmitTimeSheet ▪ SubmitTimeSheetTemporarily ▪ ReopenTimeSheet ▪ CopyTimeSheet ▪ PrintTimeSheet
summary	▪ submit ▪ reopen	▪ CreateTimeSheet ▪ SubmitTimeSheet ▪ SubmitTimeSheetTemporarily ▪ ReopenTimeSheet ▪ CopyTimeSheet ▪ PrintTimeSheet
timesheetdays	▪ submit ▪ reopen	▪ CreateTimeSheet ▪ SubmitTimeSheet ▪ SubmitTimeSheetTemporarily ▪ ReopenTimeSheet ▪ CopyTimeSheet ▪ PrintTimeSheet
timesheet	▪ submit ▪ reopen	▪ CreateTimeSheet ▪ SubmitTimeSheet

Layout	Touch Action	Maconomy Action
		▪ SubmitTimeSheetTemporarily ▪ ReopenTimeSheet ▪ CopyTimeSheet ▪ PrintTimeSheet
timesheetlogentrydaily	–	▪ CreateJobFavorite

Note: Some Maconomy default actions may not be available in Touch and vice versa. In addition, the Approval screen does not have actions in the standard layouts, and you cannot add custom actions to the screen.

Syntax Descriptions

Label



Description

Label is one of the basic components in the layout customization. It is responsible for displaying literal string or caption in the screen. It is commonly used for defining a header title in the "`grid`" component or often time used for description of the "`field`" component.

Rules

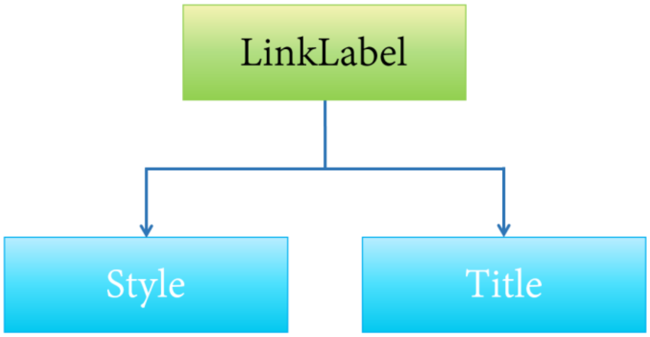
- When defining a label always declare the "`title`" attribute.
- The "`style`" attribute is optional.
- It is commonly used for defining a header title in the "`grid`" component or often time used for description of the "`field`" component.

Example

```
{
    "type": "label",
    "T$title": "Employee Number: "
},
{
    "type": "field",
    "source": "EmployeeNumber"
}
```

Output: Employee Number:001

LinkLabel



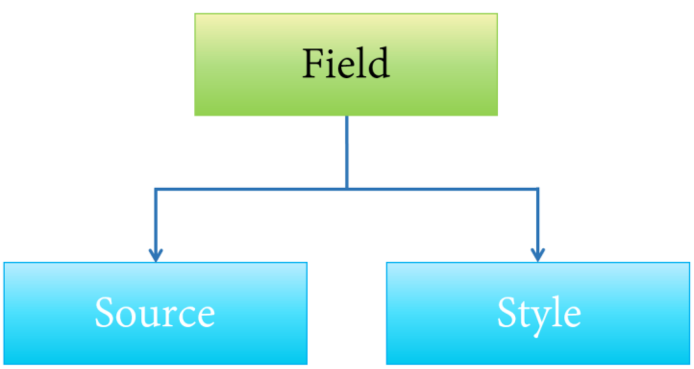
Description

LinkLabel is similar to the Label component, however, it is clickable and an action can be assigned.

Example

```
{
    "type": "linklabel",
    "T$title": "Expense sheet",
    "action": #N "open_expensesheet_header"
},
```

Field



Description

Use to display a database field on a screen.

Rules

- The "source" attribute is very important when declaring a "field" component.
- The "style" attribute is optional.
- The "mask" attribute can be used only for the "field" component (see Component Layout Attributes).
- If the value of the field does not give descriptive information on a screen
- The value of the field is very predictable (for example, the Timesheet Status can have the following values: A = Approved, R = Rejected, or S = Submitted).
- If the value of the field can be empty string and you would like to show "Untitled" instead of empty string

Examples

This is a simple way to declare a "field" component in the layout file. It displays the "EmployeeNumber" field with "Employee Number: " label.

```
{
  "type": "label",
  "T$title": "Employee Number: "
},
{
  "type": "field",
  "source": "EmployeeNumber"
}
```

```
Output: Employee Number:001
```

If you want to set a specific styling in the "field" component, you can use the "style" attribute. For example, you want to display "Employee Number" in blue.

```
{
  "type": "label",
  "T$title": "Employee Number: "
},
{
  "type": "field",
  "source": "EmployeeNumber",
  "style": {
         "color": "#00488a"
  }
}
```

```
Output: Employee Number:001
```

> **Note:** The value of color in this example is the corresponding hexadecimal value of color blue in CSS. If you want to have another styling, refer to the Style Attributes section.

Another case of the `"field"` component is that, when the field value is not bringing descriptive information on the screen and when you want it that field value be informative, you can use the `"mask"` attribute. For example, the `"field"` can only contain `"S"`, `"A"`, or `"R"` value, and you want to display it as `"Submitted"`, `"Approved"`, and `"Rejected"`, respectively.

```
{
    "type": "label",
    "T$title": "Timesheet Status"
},
{
    "type": "field",
    "source": "Status",
    "mask": {
            "Description": {
                    "T$S": "Submitted",
                    "T$A": #T"Approved",
                    "T$R": #T"Rejected"
            }
    }
}
```

Output

- If the `"Status"` field value is `"S"`, the output is:

  Timesheet Status: Submitted
- If the `"Status"` field value is `"A"`, the output is:

  Timesheet Status: Approved
- If the `"Status"` field value is `"R"`, the output is:

  Timesheet Status: Rejected

> **Note:** In the example above, the `"Status"` field is very predictable because it can only contained `"S"`, `"A"`, or `"R"` value. Appending `"#T"` to the corresponding string indicates that the language translation is automatic.
>
> You cannot specify any styling attribute on the mask attribute. In case, there is a corresponding color for each of this value, you can use the `"Color"` attribute of the mask (see the next example).

To continue the example of the "field" component with the "mask" attribute, we are apply the corresponding color for each value of the "Status" field, such as green, blue and red for "S", "A", and "R", respectively.

```
{
    "type": "label",
    "T$title": "Timesheet Status"
},
{
    "type": "field",
    "source": "Status",
    "mask": {
        "Description": {
            "T$S": "Submitted",
            "T$A": "Approved",
            "T$R": "Rejected"
        },
        "Color": {
            "S": "#53a800",
            "A": "#0080ff",
            "R": "#a00"
        }
    }
}
```

Output

- If the "Status" field value is "S", the output is:

  Timesheet Status: Submitted

- If the "Status" field value is "A", the output is:

  Timesheet Status: Approved

- If the "Status" field value is "R", the output is:

  Timesheet Status: Rejected

Another scenario is when the value of the field can be a "true" or "false", but instead of defining the value of that field as an attribute within the "Description" and/or "Color" attribute, you need to set the "truevalue" or "falsevalue". If the value is set directly, an error in the layout file occurs. For example, we have the "Invoiceable" field, which contains "true" or "false" value, but we want to display a "Billable" or "Non-Billable" string on a screen.

```
{
    "type": "field",
```

```
        "Source": "Invoiceable",
        "mask": {
                "Description": {
                        "T$truevalue": "Billable",
                        "T$falsevalue": "Non-Billable"
                }
        }
}
```

Output

- If the "`Invoiceable`" field value is "`true`", the output is:

  Billable

- If the "`Invoiceable`" field value is "`false`", the output is:

  Non-Billable

You can also add the corresponding color for each value of that field (see example below).

```
{
   "type": "field",
   "source": "Invoiceable",
   "mask": {
           "Description": {
                   "T$truevalue": "Billable",
                   "T$falsevalue": "Non-Billable"
           },
           "Color": {
                   "truevalue": "blue",
                   "falsevalue": "red"
           }
   }
}
```

Output

- If the "`Invoiceable`" field value is "`true`", the output is:

  Billable

- If the "`Invoiceable`" field value is "`false`", the output is:

  Non-Billable

Some values of the field require specific formatting requirement or need to follow the device formatting, for example, the hours and date fields.

- Most of the time, the `"hours"` field value is in the "numeric/decimal" format (for example, 1.5 means 1 hour and 30 minutes). In this way, you need to declare the `"hours"` value in the `"mask"` attribute (see example below).

```
{
    "type": "field",
    "source": "NumberOf",
    "mask": "hours"
}
```

- If you have a `"hours"` field and the date format is in the "YYYY.mm.dd" format (for example, 2014.04.02) and you want to convert the format to the device short date formatting, you can set the `"shortdate"` value in the `"mask"` attribute (see example below).

```
{
    "type": "field",
    "source": "EntryDate",
    "mask": "shortdate"
}
```

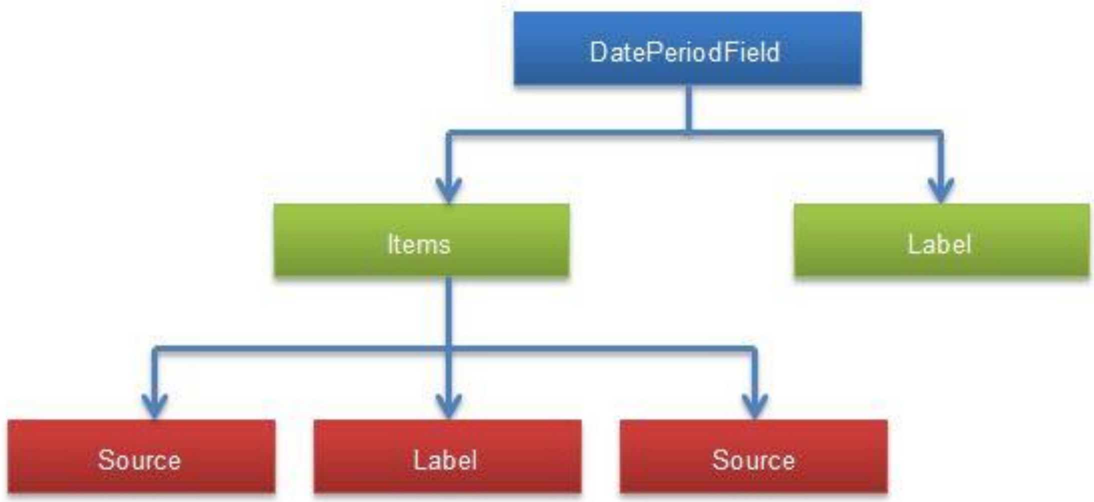
> **Note: YYYY** represents the year (for example, 2014). **dd** represents the day of the month (from 00–31). **mm** represents the month of the year (from 01–12).

If you would like to show the `"Untitled"` string instead of an empty string when the value of a field is empty string, then you can set the `"untitledtext"` value in the `"mask"` attribute (see example below):

```
{
    "type": #"field",
    "source": "description",
    "mask": "untitledtext"
}
```

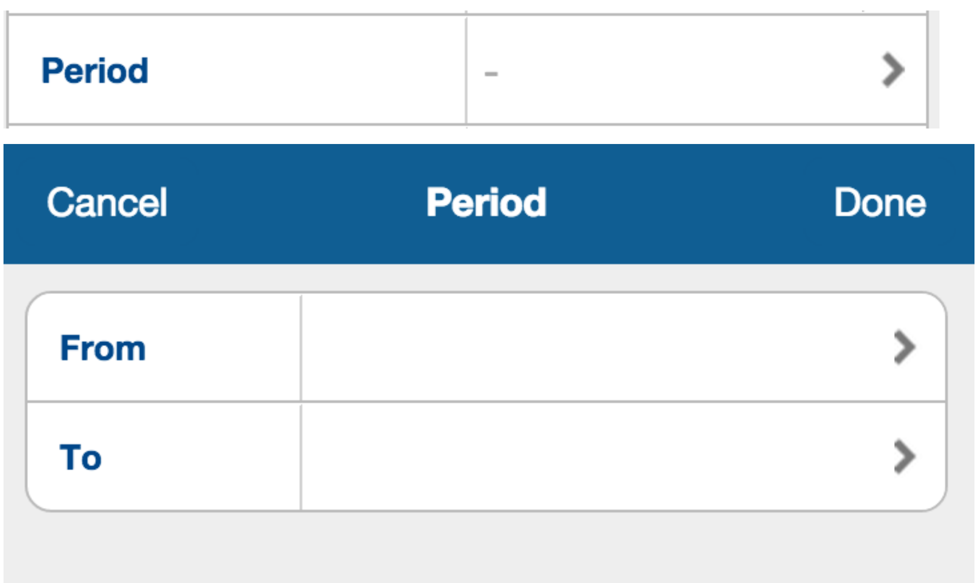
This is used, for example in the Expense Sheets screen, if the expense sheet does not have a description.

Date Period Field



Description

This component is used to enter and display a date period.



Rules

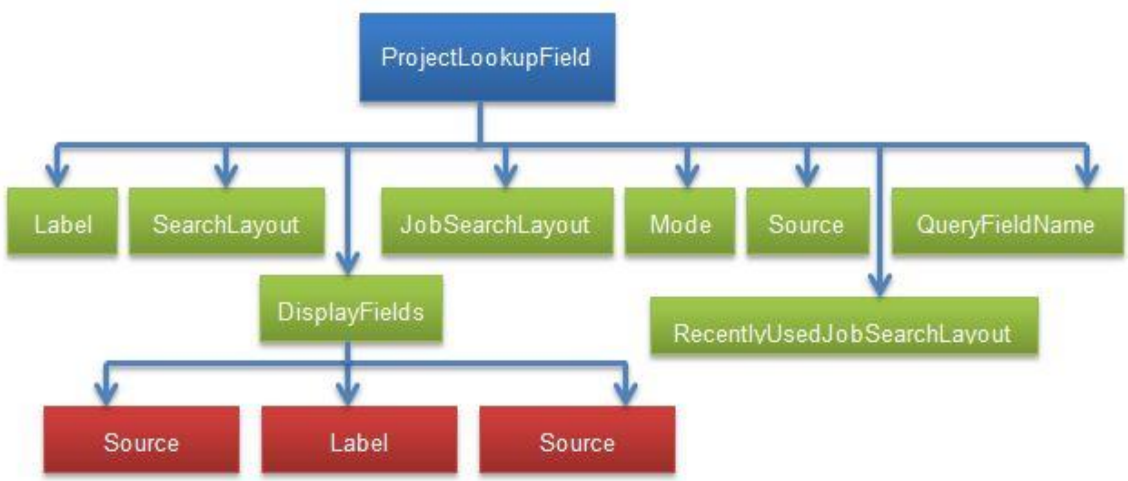
- This component does not support any styling.
- Items always contain a source, label, and another source in exact order.
- The source must be of type **Date**.
- The first label corresponds to label of the field. The second label act as a separator for the two dates.

Examples

This is a general syntax declaration of the "dateperiodfield" component:

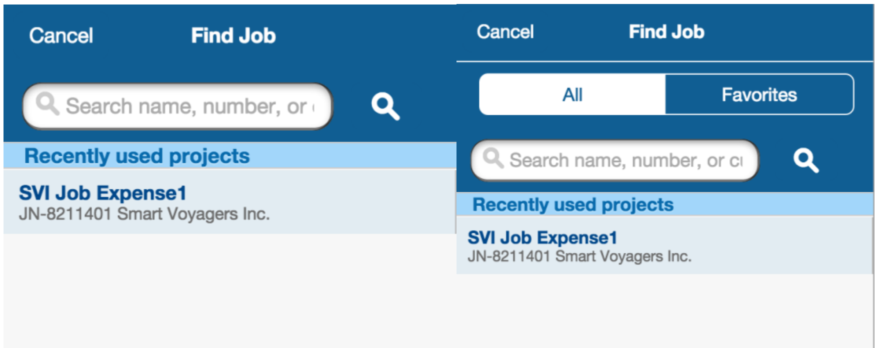
```
{
        "type": "dateperiodfield",
        "label": "Period",
        "items": [
                { "source": "FromDate" },
                { "label": "-" },
                { "source": "ToDate" }
]
}
```

Project Lookup Field



Description

This component is used for entering a job using a lookup, it could also be used to enter a job based on favorite.



Rules

- This component does not support any styling.
- This component has three modes: `"project"`, `"favorite"`, and `"projectfavorite"`. This would determine how the job selection would be.
- This component should have a `"queryfieldname = KeyValue"` attribute in the parent node.
- The `"source"` attribute inside `"displayfields"` should contain a corresponding `"queryfieldname"`.
- The `"queryfieldnames"` can be a substitute attribute name for `"queryfieldname"`. This is needed if the lookup feature is composed of “All” and “Recently Used” because there are instances that these two features are requested with two different Web service methods, which returns different field names.
- Example: `queryfieldnames: ["KeyValue1", "KeyValue2"]`

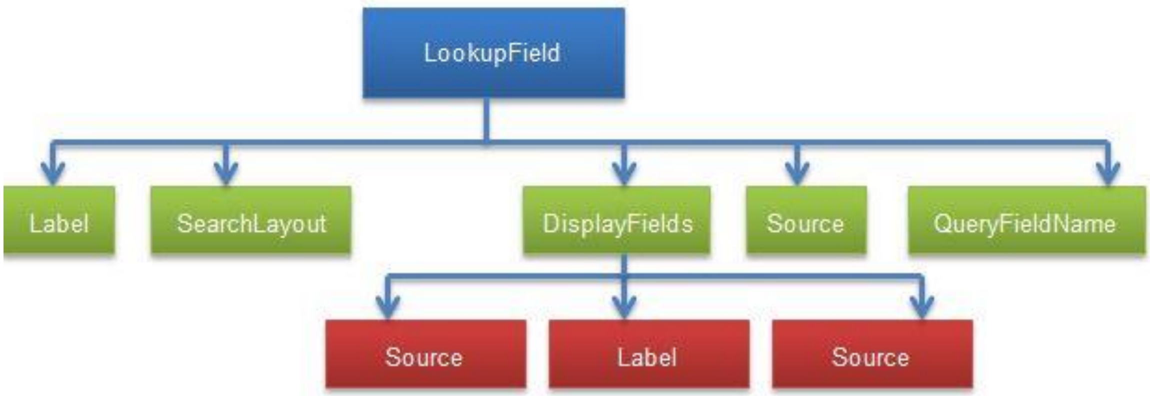
Examples

This is a general syntax declaration of the `"projectlookupfield"` component:

```
{
    "source": "JobNumber",
    "label": "Job",
    "searchlayout": "Expense_JobSearch",
    "type": "projectlookupfield",
    "mode": "projectfavorite",
    "jobsearchlayout": "Expense_JobSearch",
    "recentlyusedjobsearchlayout": "Expense_RecentlyUsedJobSearch"
    "queryfieldname": "KeyValue",
    "displayfields": [
        {
            "source": "JobNumber",
            "queryfieldname": "KeyValue"
        },
        {
```

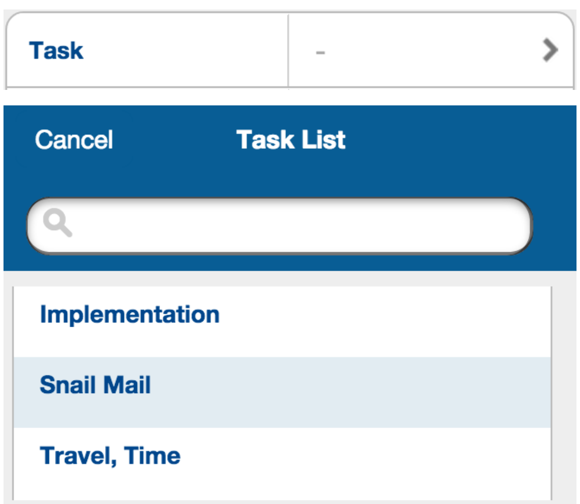
```
                    "label": " - ",
            },
            {
                    "source": "JobNameVar",
                    "queryfieldname": "DisplayValue"
            }
]
}
```

Lookup Field



Description

This component is used for entering a value using a lookup, which is commonly used in entering tasks and activities.



Rules

- This component does not support any styling.

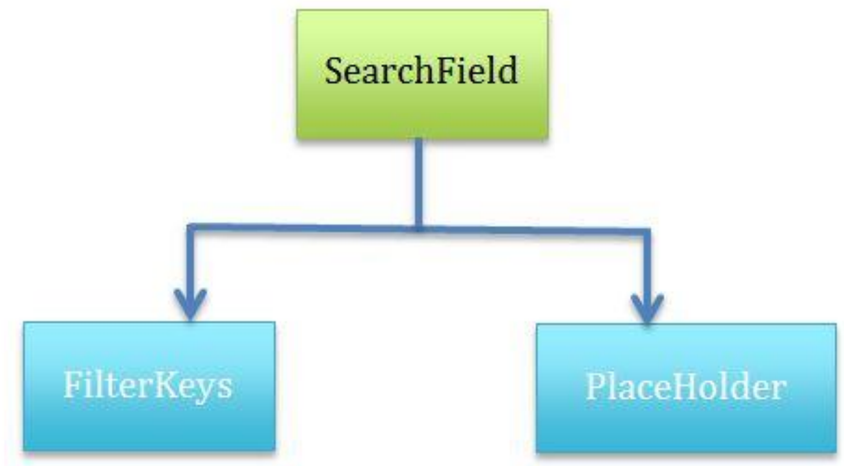
- This component should have an attribute called `"searchlayout"`.
- The `"source"` attribute inside `"displayfields"` should have a corresponding `"queryfieldname"`.
- The `"queryfieldnames"` can be a substitute attribute name for `"queryfieldname"`. This is needed if the lookup feature is composed of “All” and “Recently Used” because there are instances that these two features are requested with two different web service methods, which returns different field names.
- Example: `queryfieldnames: ["KeyValue1", "KeyValue2"]`

Examples

This is a general syntax declaration of the `"lookupfield"` component:

```
{
    "source": "TaskName",
    "label": "Task",
    "searchlayout": "Expense_TaskSearch",
    "type": "lookupfield",
    "queryfieldname": "KeyValue",
    "displayfields": [
        {
            "source": "TaskName",
            "queryfieldname": "KeyValue"
        },
        {
            "label": " - ",
        },
        {
            "source": "TaskDescriptionVar",
            "queryfieldname": "DisplayValue"
        }
]
}
```

SearchField



Description

This component is only used for the lookup layout file, which is being referenced from the "lookupfield" component. To control what field will be used to filter the record displays in the list of the lookup, you need to specify the field name in the "filterkeys" attribute. In case there are multiple fields set in the "filterkeys" attribute, it is recommended that you have a placeholder text in the search field so that you can enter the text in the "placeholder" attribute.

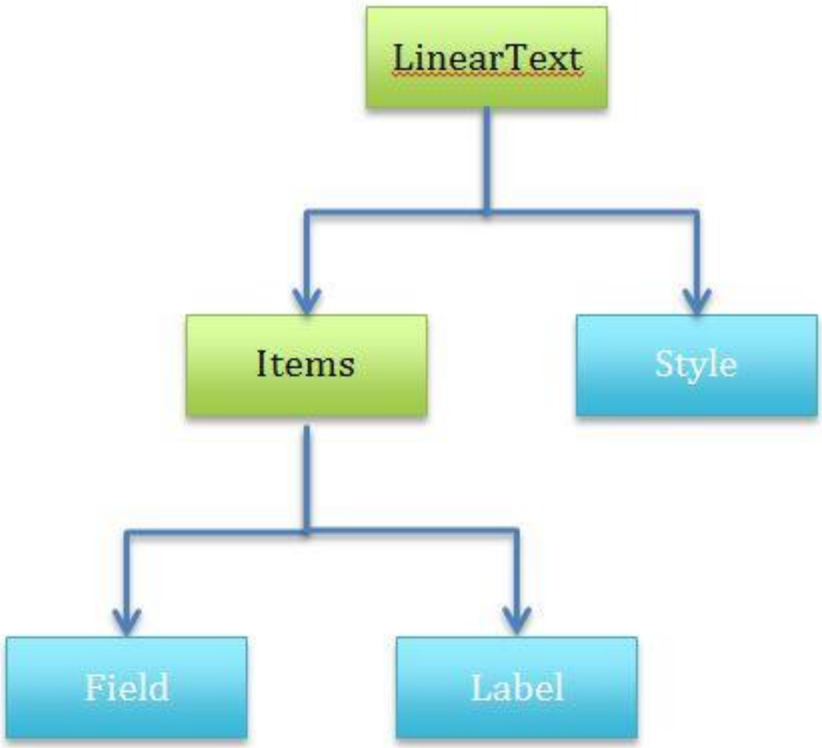
Rules

- The "filterkeys" attribute is mandatory for the "searchfield" component.
- Setting multiple fields in the "filterkeys" should be separated by comma character (for example, "KeyValue, DisplayValue").
- The value that can be set in "filterkeys" is dependent on the response of the "search" operation or the fields being returned by the search layout. (Refer to the DeltekTouch.I).
- The "placeholder" attribute is optional.
- This component must only be used in the lookup layout file.

Examples

```
{
    "region": "SearchFieldArea",
    "protectedregion": false,
    "type": "searchfield",
    "filterkeys": "KeyValue, DisplayValue",
    "T$placeholder": "Task Name, Task Description"
}
```

LinearText



Description

This component is used to group components, such as **Label** and **Field**, and display them in a single line. It can only group **Label** and **Field**.

Rules

- If there is a `"style"` defined in the `"lineartext"` component, this means that it applies to all items. For example, the style contains "fontsize:14px" and the items of the `"lineartext"` contained two (2) labels and one (1) field; those will have a 14px of font size.
- If the styling in the item is present, it overrides the styling of the `"lineartext"` component.
- You need to declare the `"items"` attribute for the component.
- The `"style"` attribute is optional.
- Always append the `"card"` prefix to the source to let the code know that this source needs to be extracted on the card part (header screen). This code, however, only to the Approvals screen.

Examples

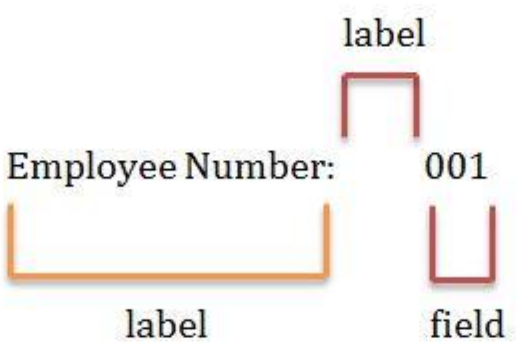
The following example is a simple way to declare a "lineartext" component in the layout file:

```
{
    "type": "lineartext",
    "items": [
            {
                    "type": "label",
                    "T$title": "Employee Number: "
            },
            {
                    "type": "label",
                    "title": "          "
            },
            {
                    "type": "field",
                    "source": "EmployeeNumber"
            }
  ]
}
```

Note: There are 10 spaces added in the following line, between "and":

`Title: #T"          "`

Output

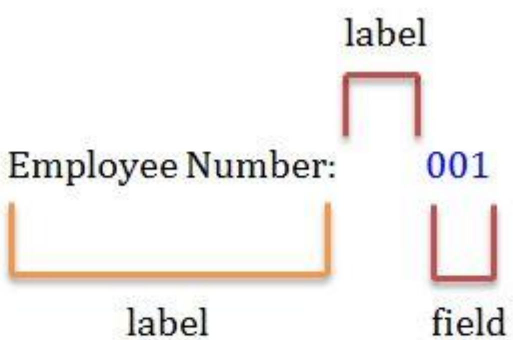


To apply "style" to the "lineartext" component (using the example above) by making the "field" item in blue, see the following example:

```
{
    "type": "lineartext",
    "style": {
```

```
            "fontsize": "14px",
            "fontweight": "bold"
    },
    "items": [
            {
                    "type": "label",
                    "T$title": "Employee Number:"
            },
            {
                    "type": "label",
                    "title": "          "
            },
            {
                    "type": "field",
                    "source": "EmployeeNumber",
                    "style": {
                            "color": "blue"
                    }
            }
    ]
}
```

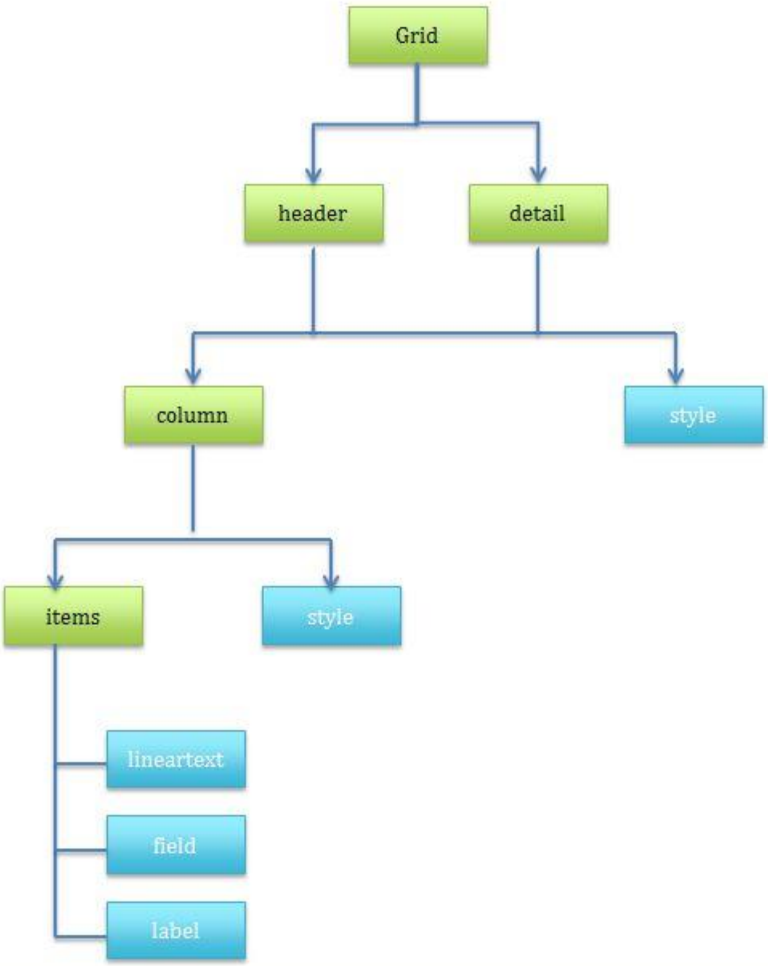
Output



Use the following code to display data on the card part (header screen):

```
{
    "type": "field",
    "source":N"card.name1"
}
```

Grid

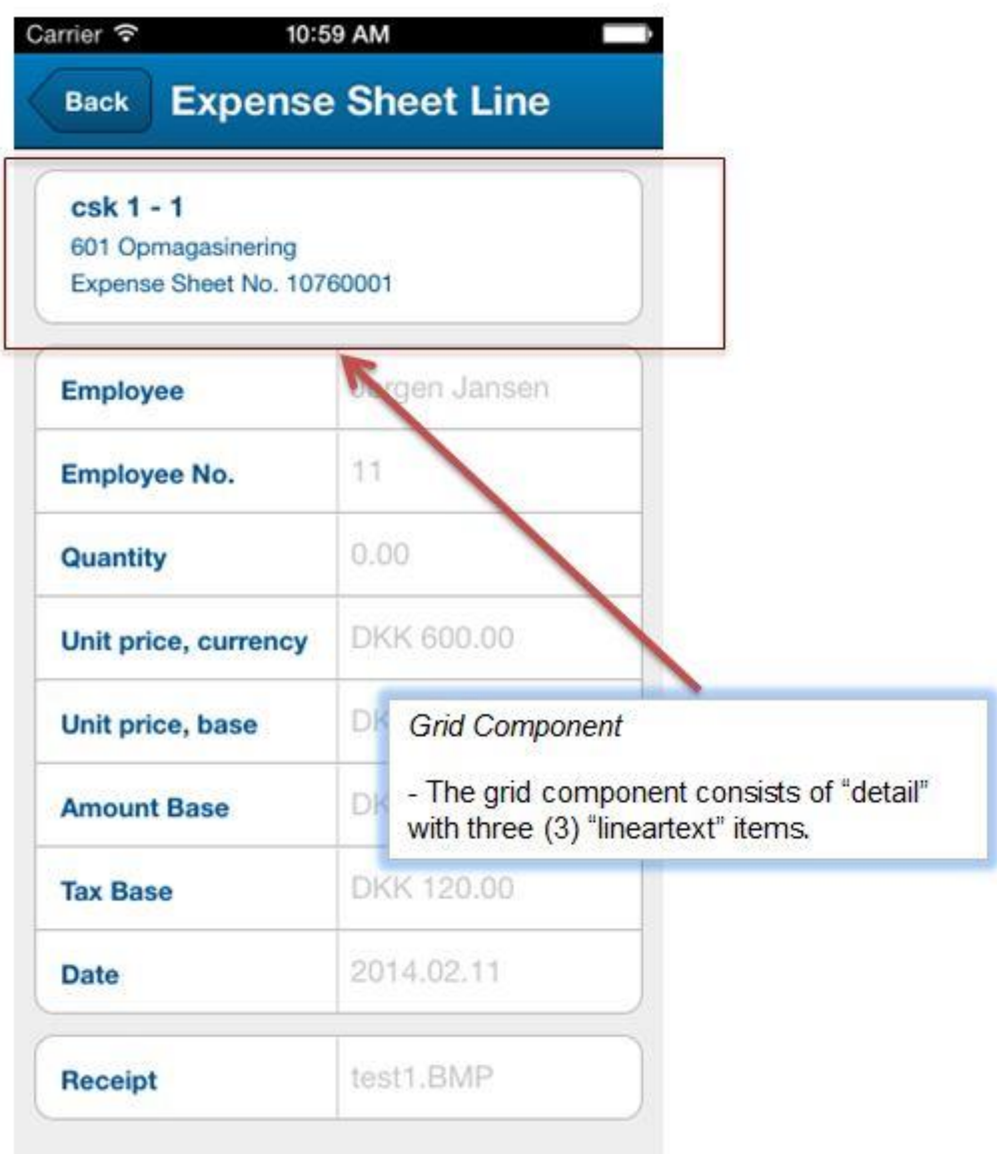


Description

This component is mainly used to display the data in a tabular representation. The `"header"` and `"detail"` are the main attributes of this component. Not all of `"grid"` in the layout file contains the `"header"` attribute. You only need the said attribute if the control you are targeting to customize on a screen contains the column title (for example, the Summary screen).

Grid is one of the very useful components in the layout customization. It is also being used on the “Line Screens” if one of the requirements is to group a set of fields in one section of the screen.

Figure 1 Sample Line Screen: Expense Sheet Line



The "`grid`" component can support multiple columns, but there are some screens you should not apply any additional columns because some of the portions in the "`grid`" component are not customizable (for example, the Timesheet screen). You can try to customize this but doing so displays a bad representation of data on the screen.

Rules

- Do not specify the "`header`" or "`detail`" attributes if the particular layout file do not contain any of these attributes. Doing so might cause some problematic rendering of the layout on the screen or Deltek Touch would not be able to read it.
- The "`header`" or "`detail`" attributes must contain at least one "`column`" component.

- By default, if there are two (2) `"column"` specified in the `"header"` or `"detail"` attributes, the first column width occupies the 70% width of the screen and the second column 30% width of the screen.
- If there are more than two (2) columns, the width is divided evenly.
- Most of the styling aspects (for example, Font Size and Font Color) in the `header"` attribute are built-in in the Touch app.

Examples

The following example is a general syntax declaration of the `"grid"` component that consists of the `"header"` and `"detail"` attributes:

```
{
    "type": "grid",
    "header": [
            {
                    "type": "column"
            },
            {
                    "type": "column"
            }
    ],
    "detail": [
            {
                    "type": "column"
            },
            {
                    "type": "column"
            }
    ]
}
```

The following example is a general syntax declaration of the `"grid"` component that consists of the `"detail"` attribute:

```
{
    "type": "grid",
    "detail": [
            {
                    "type": "column"
            },
            {
                    "type": "column"
```

```
            }
        ]
    }
```

Complete syntax declaration of the `"grid"` component that consists of the `"header"` and `"detail"` attributes.

```
        }
        ],
        "detail": [{
                        "type": "column",
                        "items": [{
                                    "type": "lineartext",
                                    "style": {
                                            "fontsize": "14px",
                                            "color": "#00488a"
                                    },
                                    "items": [
                                            {
                                                    "type": "field",
                                                    "source": "ShortDate"
                                            },
                                            {
                                                    "type": "label",
                                                    "title": " - "
                                            },
                                            {
                                                    "type": "field",
                                                    "source": "WeekDay"
                                            }
                                    ]
                            }
                        ]
                },
                {
                        "type": "column",
                        "style": {
                                "textalignment": "right",
                                "fontsize": "14px",
                                "paddingright": "10px",
                                "color": "#00488a",
```

```
                              "fontweight": "bold"
                       },
                       "items": [ {
                                     "type": "field",
                                     "source": "TotalHoursRegisteredHours",
                                     "mask": "hoursnozeroes"
                              }
                       ]
               }
       ]
}
```

Output

Here is another example of the `"grid"` component syntax declaration that consists of the `"detail"` attribute, wherein the second column is not customizable.

```
{
       "type": "grid",
```

```
      "detail": [{
                  "type": "column",
                  "items": [{
                              "type": "lineartext",
                              "style": {
                                    "fontsize": "14px",
                                    "color": "004488a",
                                    "fontweight": "bold"
                              },
                              "items": [{
                                          "type": "field",
                                          "source": "JobNameVar"
                                    },
                                    {
                                          "type": "label",
                                          "title": " - ",
                                    },
                                    {
                                          "type": "field",
                                          "source": "JobNumber"
                                    }
                              ]
                        },
                        {
                              "type": "lineartext",
                              "style": {
                                    "fontsize": "12px"
                                    "color": "#6b6b6b"
                              },
                              "items": [{
                                          "type": "field",
                                          "source": "TaskName"
                                    },
                                    {
                                          "type": "label",
                                          "title": " - "
                                    },
                                    {
                                          "type": "field",
```

```
                            "source": "TaskDescriptionVar"
                        }
                    ]
                },
                {
                    "type": "lineartext",
                    "style": {
                        "fontsize": "12px",
                        "color": "#6b6b6b"
                    },
                    "items": [
                        {
                            "type": "field",
                            "source": "CustomerNameVar"
                        }
                    ]
                },
                {
                    "type": "lineartext",
                    "style": {
                        "fontsize": "12px",
                        "color": "#6b6b6b"
                    },
                    "items": [
                        {
                            "type": "field",
                            "source": "Invoiceable",
                            "mask": {
                                "Description": {
                                    "truevalue": "Billable",
    "falsevalue": "Non-Billable"
                                }
                            }
                        }
                    ]
                }
            ]
        },
        {
```

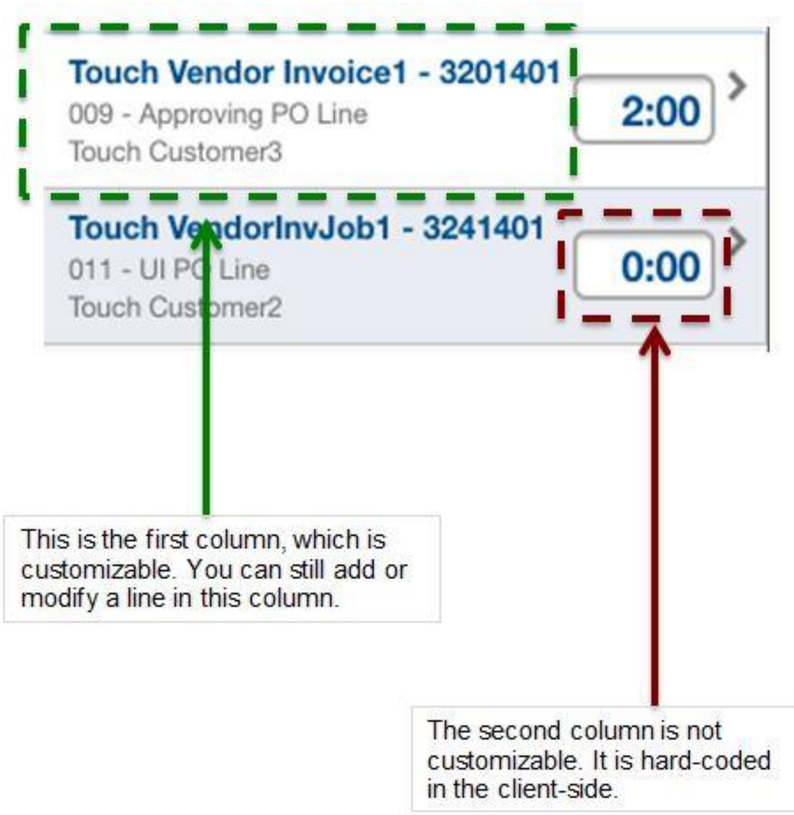
```
                        "type": "column",
                        "region": "HourField",
                        "protectedregion": true
                }
        ]
}
```

Note: The following last three lines refer to the second column, but it is not customizable. The Touch client handles the code and display of the hour field on the Timesheet screen. In short, it is hard-coded in the client-side:

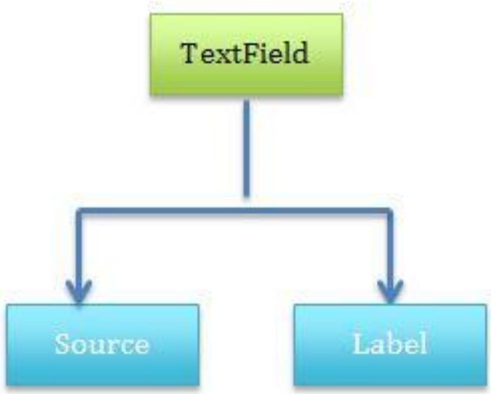
```
{
    "type": "column",
    "region": "HourField",
    "protectedregion": true
 }
```

Output

The following is a portion of layout on the **Timesheet** screen:



TextField



Description

This component is used to display database field and label on Line screens. For example, **Expense Sheet Line**:

Rules

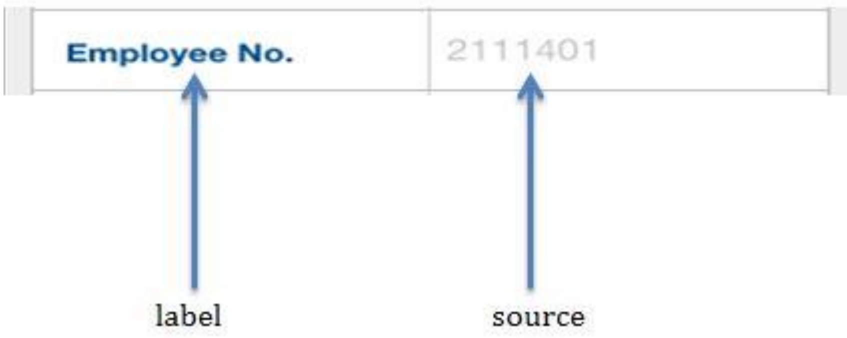
- Always specify the `"source"` and `"label"` attributes and their corresponding values.
- This component does not support styling. The formatting of the font used in a label is provided automatically by the client-side.
- Always append the "card" prefix to the source to let the code know that this source needs to be extracted on the card part (header screen). This code, however, only applies to the Approvals screen.

Examples

This is a general syntax declaration of the `"textfield"` component:

```
{
    "type": "textfield",
    "source": "EmployeeNumber",
    "T$label": "Employee No."
}
```

Output



If there is a currency and an amount field that need to display in a single `"textfield"` component, use the `"mask"` attribute.

```
{
    "type": "textfield",
    "source”: "UnitPriceCurrency",
    "label": "Unit price, currency",
    "mask": {
          "Description": {
                  "currencysource": "Currency",
                  "amountsource": "UnitPriceCurrency"
          }
  }
}
```

Output

Use the following code to display data on the card part (header screen):

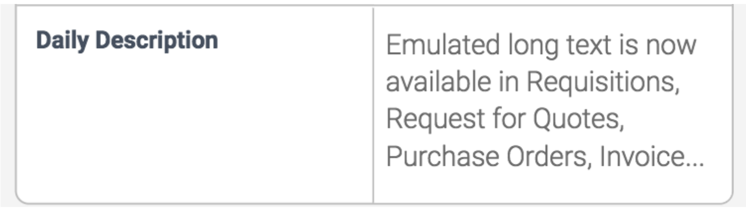
```
{
    "type": "textfield",
    "source": "card.requisitioneremployeenamevar",
    "T$label": "Requisitioner"
```

```
}
```

TextareaField

Description

This component is used to display a text area field on a Line screen. It defines a multi-line text input control and can hold an unlimited number of characters depending on the system setup. For example, Timesheet Line:



Rules

- Always specify the "`source`" and "`label`" attributes and their corresponding values.
- This component does not support styling. The formatting of the font used in a label is provided automatically by the client-side.
- The default maximum length of this component is set to **255** characters. You need to change DeltekTouch.I to enable support long text fields.

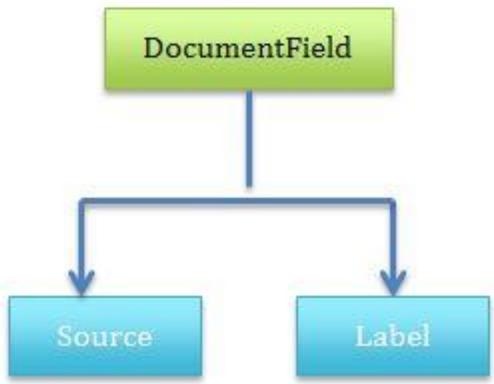
Examples

This is a general syntax declaration of the "`textareafield`" component:

```
{
    "type": "textareafield",
    "source": "Daily Description",
    "T$label": "Daily Description"
}
```

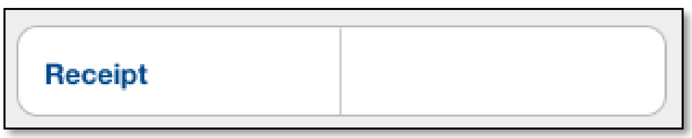
Output

DocumentField



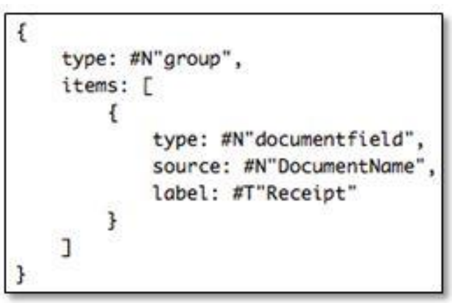
Description

This component is used for displaying attached document on a separate screen once a user taps a field (for example, PDF, PNG, or IMG files).



Rules

- Always specify the "`source`" and "`label`" attributes as well as the corresponding values.
- The "`documentfield`" component only works properly if the "`source`" attribute contains "`DocumentName`" value (see image below).

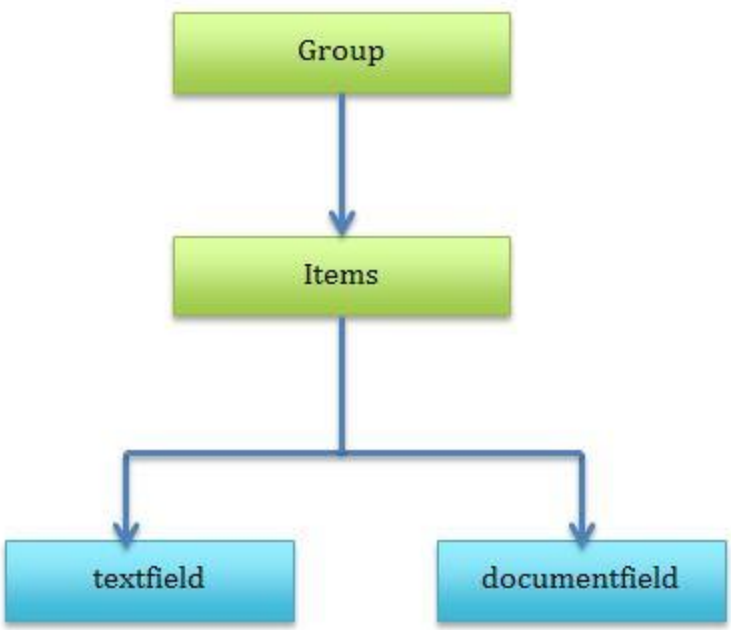


```
{
    type: #N"group",
    items: [
        {
            type: #N"documentfield",
            source: #N"DocumentName",
            label: #T"Receipt"
        }
    ]
}
```

- The "`mask`" attribute is not applicable in this component.
- This component does not support styling. The formatting of the font of a label is provided automatically by the client-side.
- The separate screen that displays the attached document is not customizable (see Figure 4 below).

Figure 2: Preview Screen

Group



Description

This component is used to group a set of fields into one component.

Rules

- This component does not support any styling.
- Only "textfield" and "documentfield" can be set in this component.
- This is a required component if you need to add a single or multiple "textfield" or "documentfield" components.

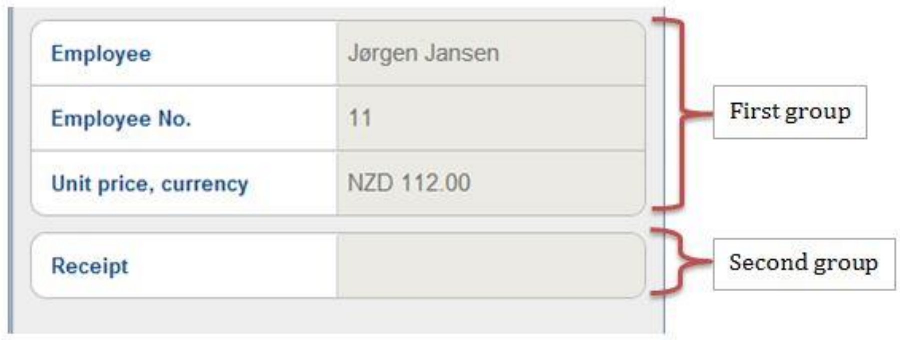
Example

This is a general syntax declaration when using the "group" component.

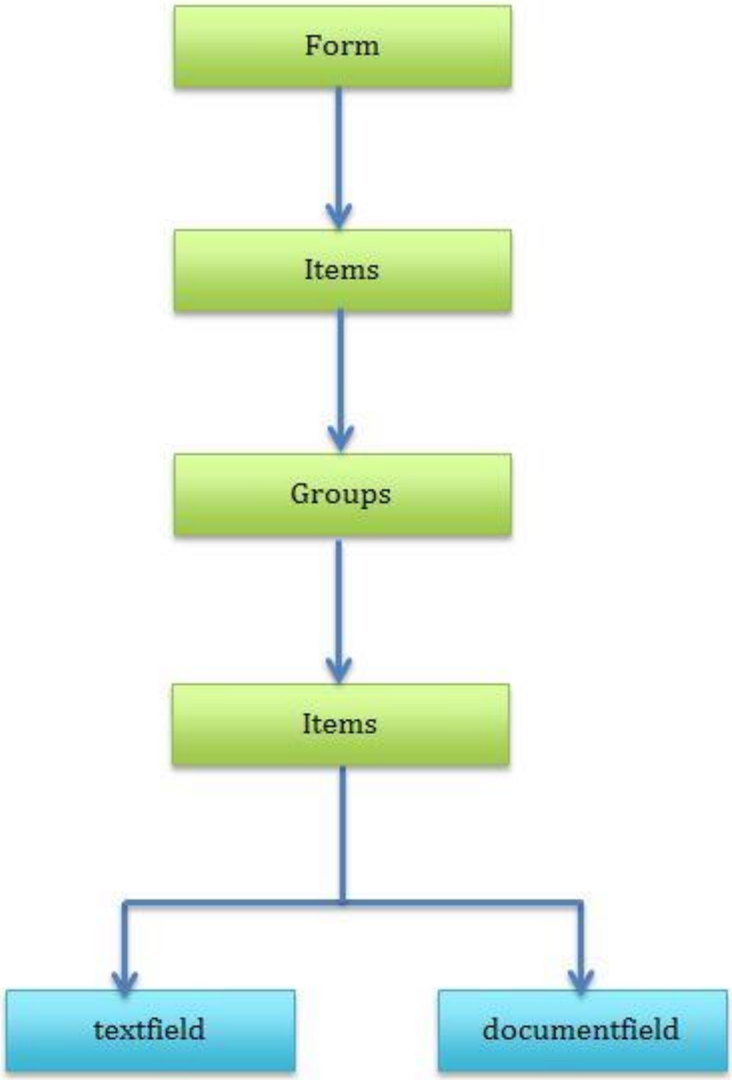
```
{
    "type": "group",
    "items": [{
                  "type": "textfield",
                  "source": "EmployeeNameVar",
                  "label":  "Employee"
          },
          {
                  "type": "textfield",
                  "source": "EmployeeNumber",
                  "T$label": "Employee No."
          },
          {
                  "type": "textfield",
                  "source": "UnitPriceCurrency",
                  "T$label": "Unit price, currency",
                  "mask": {
                          "Description": {
                                  "currencysource": "Currency",
                                  "amountsource": "UnitPriceCurrency"
                          }
                  }
          }]
},
{
    "type": "group",
    "items": [{
                  "type": "documentfield",
                  "source": "DocumentName",
                  "T$label": "Receipt"
```

```
            } ]
}
```

Output


Employee
Jørgen Jansen
Employee No.
11
Unit price, currency
NZD 112.00
Receipt
First group
Second group

Form



Description

This component is used to tell the Touch Client that the component it is encapsulating in the `"items"` attribute is designed for Line screens. Typically, it is using the HTML5 or Sencha Touch native controls (for example, textfield).

Rules

- This component does not support any styling.
- Always defined a `"group"` component in the `"items"` attribute.

- You cannot set a "textfield" or a "documentfield" directly in the "items" attribute.
- You cannot set a "form" component inside the "items" attribute.

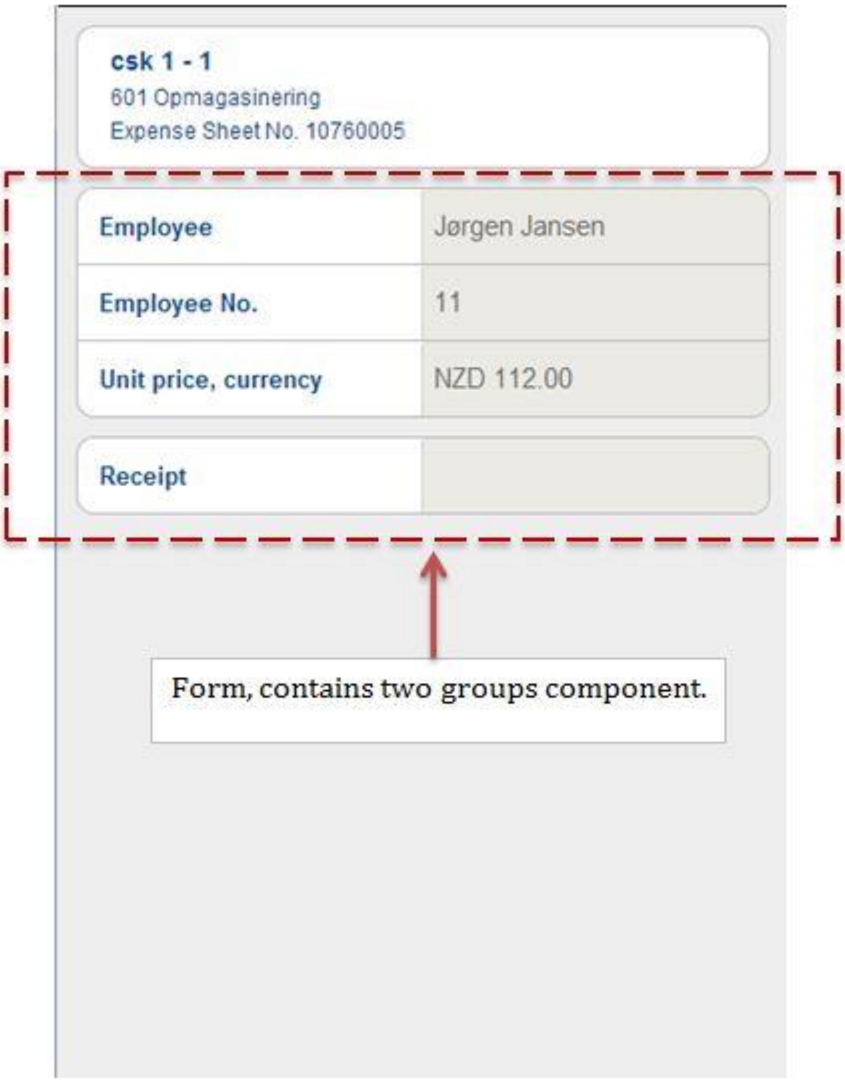
Example

This is a general syntax declaration when using the "form" component.

```
{
   "type": "form",
   "items": [{
        "type": "group",
        "items": [{
                     "type": "textfield",
                     "source": "EmployeeNameVar",
                     "T$label":  "Employee"
              },
              {
                     "type": "textfield",
                     "source": "EmployeeNumber",
                     "T$label": "Employee No. "
              },
              {
                     "type": "textfield",
                     "source": "UnitPriceCurrency",
                     "T$label": "Unit price, currency",
                     "mask": {
                            "Description": {
                                   "currencysource": "Currency",
                                   "amountsource": "UnitPriceCurrency"
                            }
                     }
              }]
        },
        {
              "type": "group",
              "items": [{
                     "type": "documentfield",
                     "source": "DocumentName",
                     "T$label": "Receipt"
        }]
```

```
        }
]}
```

Output



Attachment Icon

Description

This component is used for displaying an icon that represents "`attachment`".

Rules

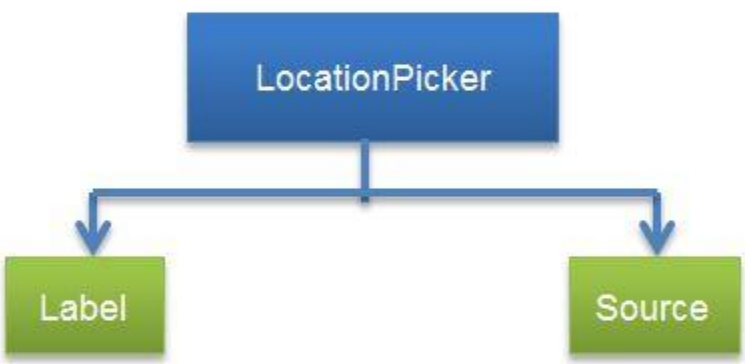
- This component should be used inside a column component.
- This component does not contain any attributes.

Example

This is a general syntax declaration of the `"attachmenticon"` component:

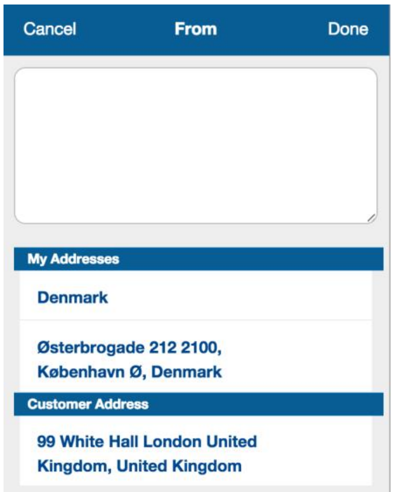
```
{
    "type": "column",
    "items": [
          {
                  "type": "attachmenticon"
          }
]
}
```

Location Picker



Description

This component is used for entering and displaying addresses.



Rules

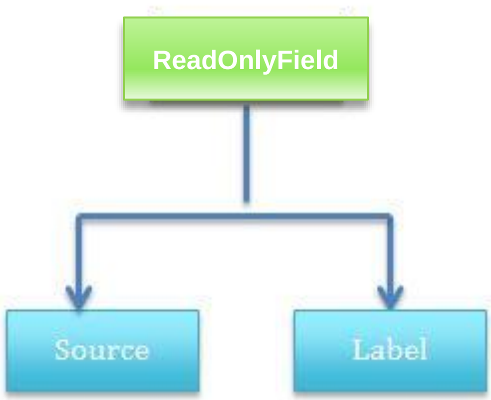
This component does not support any styling.

Example

This is a general syntax declaration of the `"locationpicker"` component:

```
{
    "source": "MileageFrom",
    "T$label": "From",
    "type": "locationpicker"
}
```

ReadOnlyField



Description

This component is used to display database field and label on the **Form** and **Line** screens that cannot be modified. For example, on the **Purchase Order** form:

Submitted	No

Rules

- Always specify the `"source"` and `"label"` attributes as well as the corresponding values.
- This component does not support styling. The formatting of the font used in a label is provided automatically by the client-side.

Examples

This is a general syntax declaration of the `"readonlyfield"` component:

```
{
    "type": "readonlyfield",
    "source": "submitted",
    "T$label": "Submitted",
    "mask": "yesno"
}
```

Output

- If the `"Submitted"` field value is `"true"`, the output is YES.
- If the `"Submitted"` field value is `"false"`, the output is NO.

Conditional Function Helpers

Description

The following functions are used to determine whether a field is available in the REST data response:

- this.defined
- this.noDefined

The layouts in Deltek Touch can be applied on several Maconomy versions (for example, same layout is applied on Maconomy 2.4 LA 1, 2.4 LA 2, 2.4 GA, and 2.4 .x). Some Maconomy versions, however, may contain fields that are not available in other Maconomy versions (for example, the **timeregistrationunit** field is available in Maconomy 2.4 GA but not in Maconomy 2.4 LA 1/ LA 2).

If you want to use such fields in the layouts, you need to check if the field is supported in the Maconomy version to which Deltek Touch is connecting using these functions.

Rules

You cannot use these functions in your custom layouts yet. You can only apply the said functions in the following standard layouts:

- absenceapprovalform.JSON
- absencerequest.JSON
- allowanceapprovalform.JSON
- allowanceapproveinputscreen.JSON
- allowancesequest.JSON

Example

```
{
    "if": ["this.defined('timeregistrationunit') &&
    (this.notDefined('timeregistrationunitprevcurrnextvar') ||
    timeregistrationunitprevcurrnextvar == 'nil')", {
    "T$label": "Time Unit",
    "source": "timeregistrationunit",
    "type": "selectfield"
 }]
 },
```

APPENDICES

Appendix A: Screens, Layouts, & Containers

The following sections display the customizable Deltek Touch screens, and the corresponding layout and containers to modify.

> **Note:** In case your layout customization is only about formatting, the container is irrelevant.

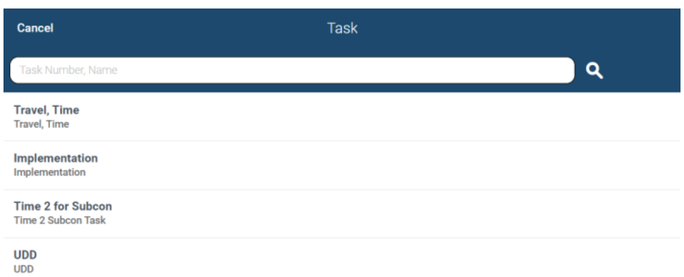
Search Screens

You can customize basic and advanced search screens by appending your own condition in REST Touch layouts.

Basic Search Screens

<table>
<tr><th>Navigation Path and Screens</th><th>REST</th></tr>
<tr><td>

Timesheet » Timesheet Line » Task (Daily)



</td><td>

Screen Layout:

timesheetlogentrydaily.JSON

Search Layout:

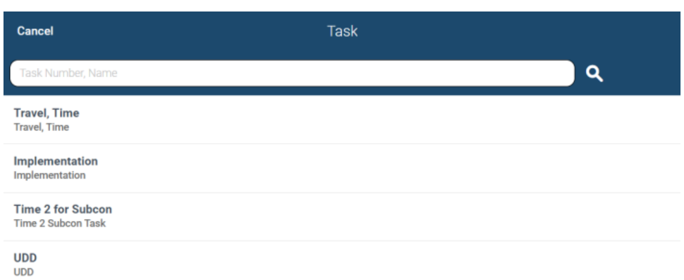
find_tasklistline.JSON

Container:

Find_TaskListLine

</td></tr>
<tr><td>

Timesheet » Timesheet Line » Task (Weekly)



</td><td>

Screen Layout:

timesheetlogentryweekly.JSON

Search Layout:

find_tasklistline.JSON

Container:

Find_TaskListLine

</td></tr>
<tr><td>

Timesheet » Timesheet Line » Activity (Daily)

</td><td>

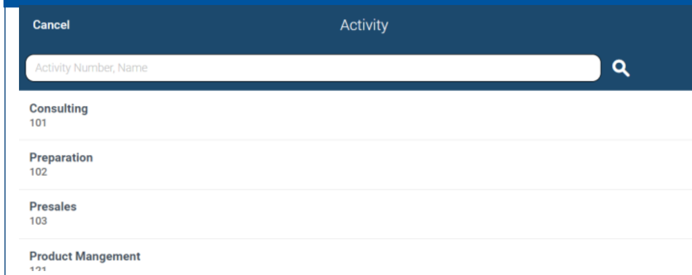
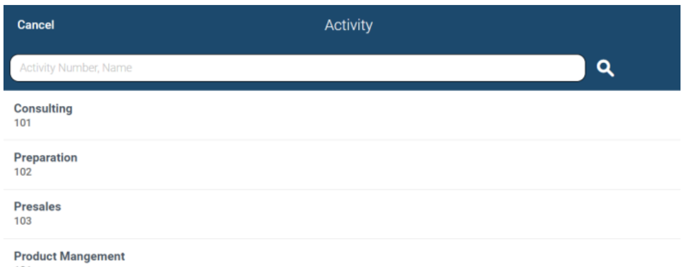
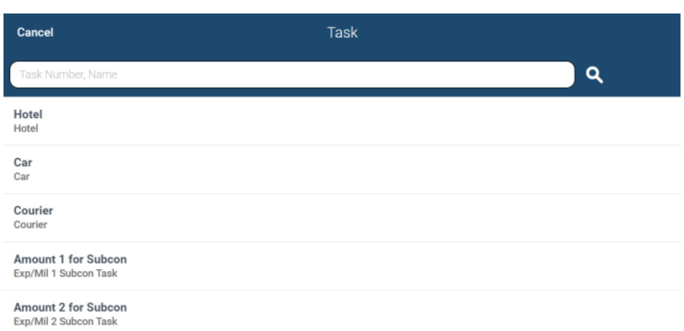
Screen Layout:

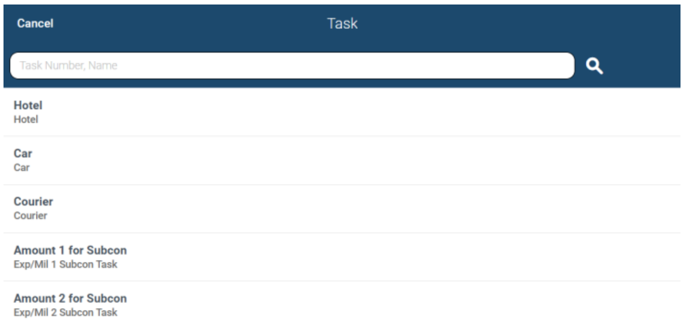
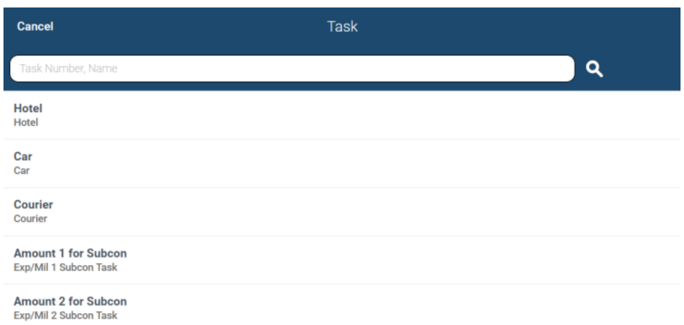
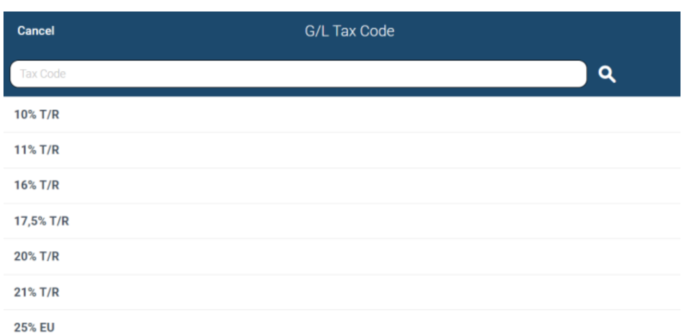
timesheetlogentrydaily.JSON

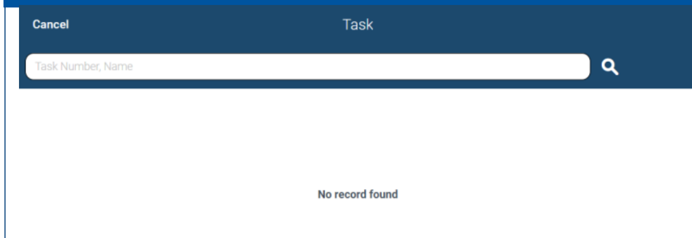
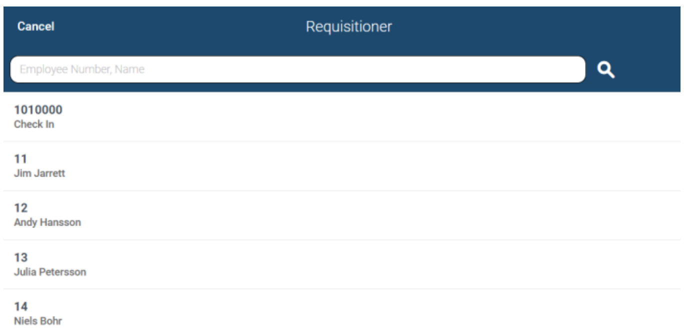
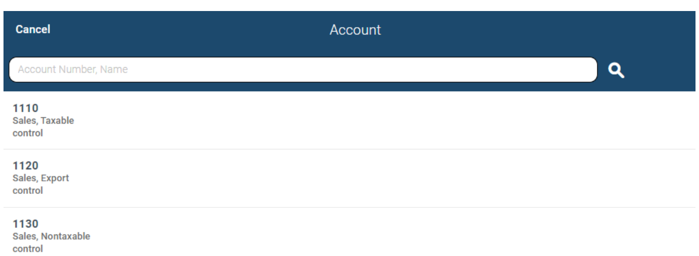
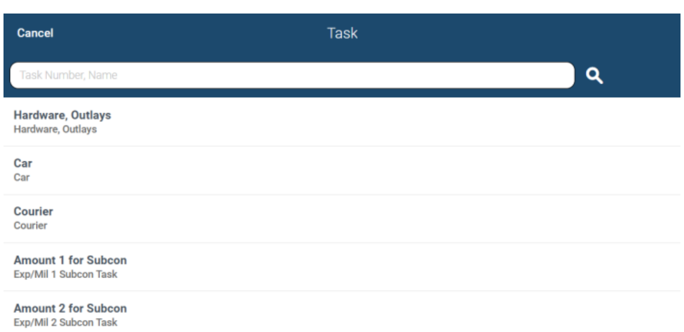
Search Layout:

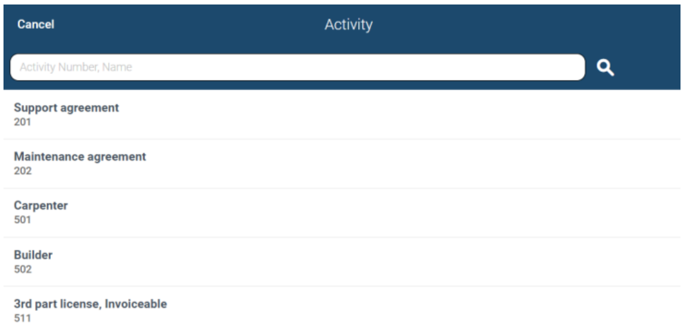
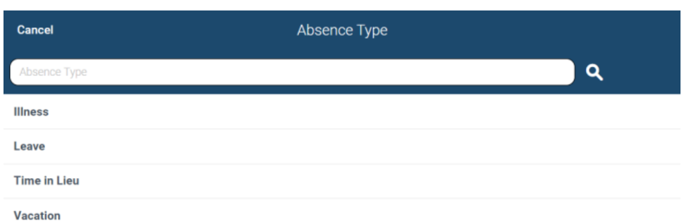
find_activity.JSON

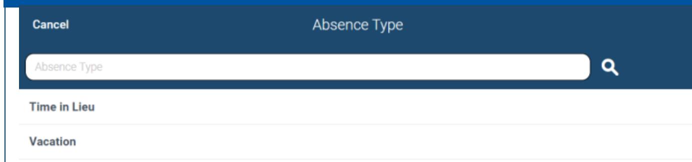
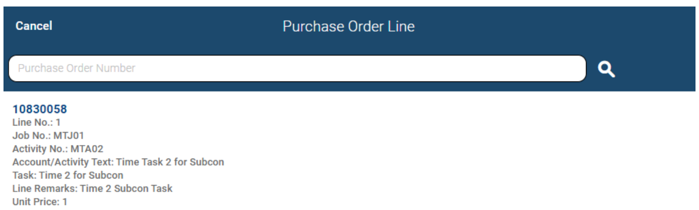
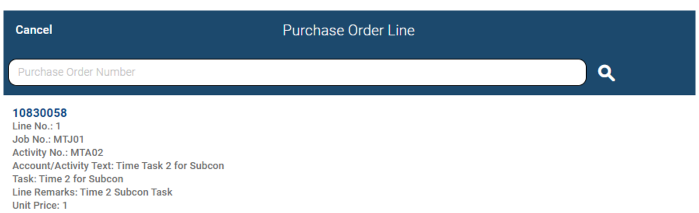
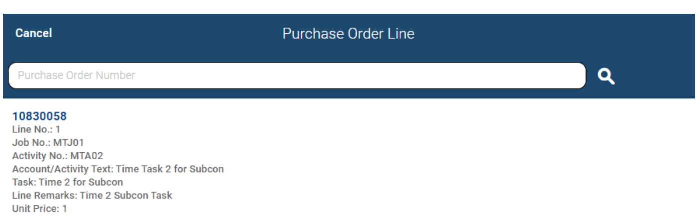
</td></tr>
</table>

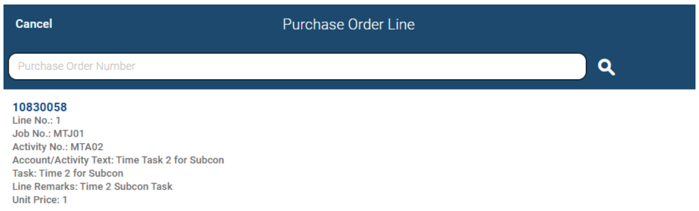
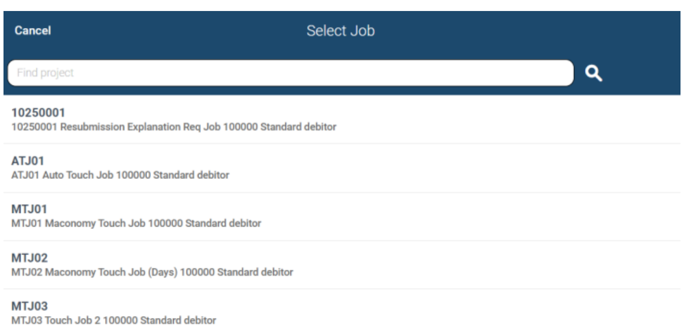
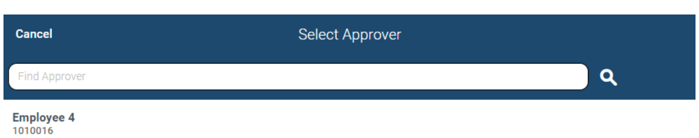
Navigation Path and Screens	REST
	**Container:** Find_Activity
Timesheet » Timesheet Line » Activity (Weekly) 	**Screen Layout:** timesheetlogentryweekly.JSON **Search Layout:** find_activity.JSON **Container:** Find_Activity
Expense Sheet » Expense Sheets » Expense Sheet Lines » Activity 	**Screen Layout:** expensesheetline.JSON **Search Layout:** find_activity.JSON **Container:** Find_Activity
Expense Sheet » Expense Sheets » Expense Sheet Lines » Task 	**Screen Layout:** expensesheetline.JSON **Search Layout:** find_tasklistline.JSON **Container:** Find_TaskListLine

Navigation Path and Screens	REST
Quick Capture » Attach Receipt » Create Expense Sheet Line 	**Screen Layout:** quickcaptureexpensesheetline.JSON **Search Layout:** find_tasklistline.JSON **Container:** Find_TaskListLine
Quick Capture » Select Expense Sheet » Expense Sheet Line 	**Screen Layout:** quickcaptureicrexpensesheetline.JSON **Search Layout:** find_tasklistline.JSON **Container:** Find_TaskListLine
Expense Sheet » Expense Sheets » Expense Sheet Lines » Tax Code 	**Screen Layout:** expensesheetline.JSON **Search Layout:** find_financevatcode.JSON **Container:** Find_FinanceVATCode
Mileage Sheets » Mileage Sheet » Mileage Sheet Line » Task	**Screen Layout:** mileagesheetline.JSON **Search Layout:** find_tasklistline.JSON

Navigation Path and Screens	REST
	**Container:** find_tasklistline
Purchase Order » Purchase Orders » Requisitioner 	**Screen Layout:** purchaseorder.JSON **Search Layout:** find_employee.JSON **Container:** Find_Employee
Purchase Order » Purchase Orders » Purchase Order » Purchase Order Lines » Account No. 	**Screen Layout:** purchaseorderline.JSON **Search Layout:** find_account.JSON **Container:** Find_Account
Purchase Order » Purchase Orders » Purchase Order Lines » Task 	**Screen Layout:** purchaseorderline.JSON **Search Layout:** find_tasklistline.JSON **Container:** Find_TaskListLine

Navigation Path and Screens	REST
Purchase Order » Purchase Orders » Purchase Order Lines » Activity  	**Screen Layout:** purchaseorderline.JSON **Search Layout:** find_activity.JSON **Container:** Find_Activity
Purchase Order » Purchase Orders » Purchase Order Lines » Employee  	**Screen Layout:** purchaseorderline.JSON **Search Layout:** find_employee.JSON **Container:** Find_Employee
Absence Requests » Absence Request » Absence Type  	**Screen Layout:** absencerequest.JSON **Search Layout:** find_absencetype.JSON **Container:** find_absencetype
Allowance Requests » Allowance Request » Absence Type	**Screen Layout:** allowancerequest.JSON **Search Layout:** find_absencetype.JSON

Navigation Path and Screens	REST
	**Container:** Find_AbsenceType
Timesheet » Timesheet Line » Purchase Order (Daily) **Timesheet » Timesheet Line » Purchase Order Line** (Daily) 	**Screen Layout:** timesheetlogentrydaily.JSON **Search Layout:** find_purchaseorderline_timesheet.JSON **Container:** Find_PurchaseOrderLine
Timesheet » Timesheet Line » Purchase Order (Weekly) **Timesheet » Timesheet Line » Purchase Order Line** (Weekly) 	**Screen Layout:** timesheetlogentryweekly.JSON **Search Layout:** find_purchaseorderline_timesheet.JSON **Container:** Find_PurchaseOrderLine
Mileage Sheet » Mileage Sheets » Mileage Sheet Line » Purchase Order **Mileage Sheet » Mileage Sheets » Mileage Sheet Line » Purchase Order Line** 	**Screen Layout:** mileagesheetline.JSON **Search Layout:** find_purchaseorderline_expensesheet.JSON **Container:** Find_PurchaseOrderLine

Navigation Path and Screens	REST
Expense Sheet » Expense Sheets » Expense Sheet Lines » Purchase Order **Expense Sheet » Expense Sheets » Expense Sheet Lines » Purchase Order Line**  	**Screen Layout:** expensesheetline.JSON **Search Layout:** find_purchaseorderline_expensesheet.JSON **Container:** Find_PurchaseOrderLine
Expense Sheet » Expense Sheets » Expense Sheet Lines » Justification	**Screen Layout:** None, hard coded in the app **Search Layout:** find_theoption_expensejustification.JSON **Container:** Find_TheOption
Approvals » Items for Approval » Select Job  	**Screen Layout:** None, hard coded in the app **Search Layout:** find_alljobheader_approval.JSON **Container:** Find_JobHeader
Approvals » Items for Approval » Select Approver  	**Screen Layout:** None, hard coded in the app **Search Layout:** find_allemployee_approval.JSON

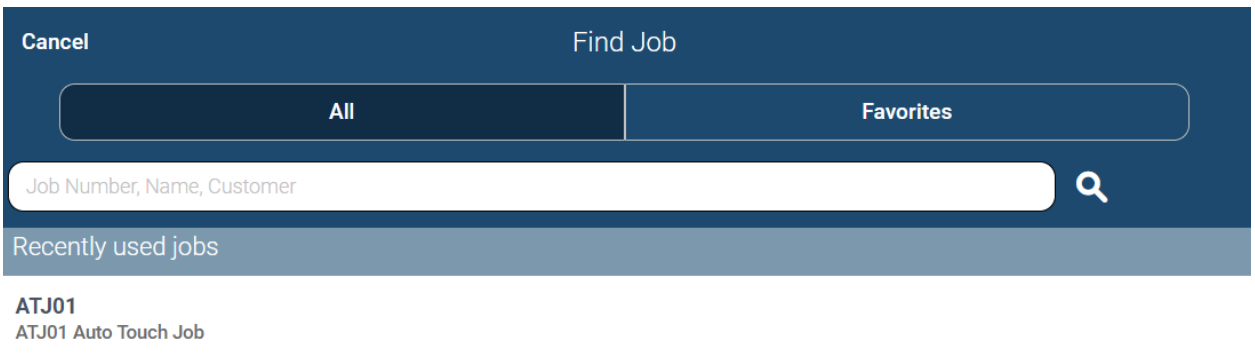
Navigation Path and Screens	REST
	Container: Find_JobHeader

Advanced Search Screens

The **Find Job**, **Find Vendor**, and **Address** screens are advanced lookup screens you can customize.

Find Job Screens

The **Find Job** screen allows you to look up your jobs, all existing jobs, any favorites, and even recently used jobs. You can modify the **Find Job** screen from the **Timesheet**, **Expense Sheet**, **Mileage Sheet**, and **Purchase Order** screens.



Navigation Path	REST
Timesheet » Timesheet Line » Find Job » All	**Screen Layout:** None, hard coded in the app **Search Layout:** find_jobheader_timesheet.JSON **Container:** Find_JobHeader
Timesheet » Timesheet Line » Find Job » Favorites	**Screen Layout:** None, hard coded in the app **Search Layout:** find_jobfavorite.JSON

Navigation Path	REST
	Container: Find_JobFavorite
Timesheet » Timesheet Line » Find Job » Recently Used Jobs	**Screen Layout:** None, hard coded in the app **Search Layout:** find_jobheader_timesheet.JSON **Container:** Find_TimesheetLineU
Timesheet » Timesheet Line » Find Job » My Jobs	**Screen Layout:** None, hard coded in the app **Search Layout:** find_jobheader_timesheet.JSON **Container:** Find_JobEmployeeControlU
Expense Sheet » Expense Sheets » Find Job » All	**Screen Layout:** expensesheet.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_JobHeader
Expense Sheet » Expense Sheets » Find Job » Recently Used Jobs	**Screen Layout:** expensesheet.JSON **Search Layout:** findjobheader_expense.JSON **Container:**

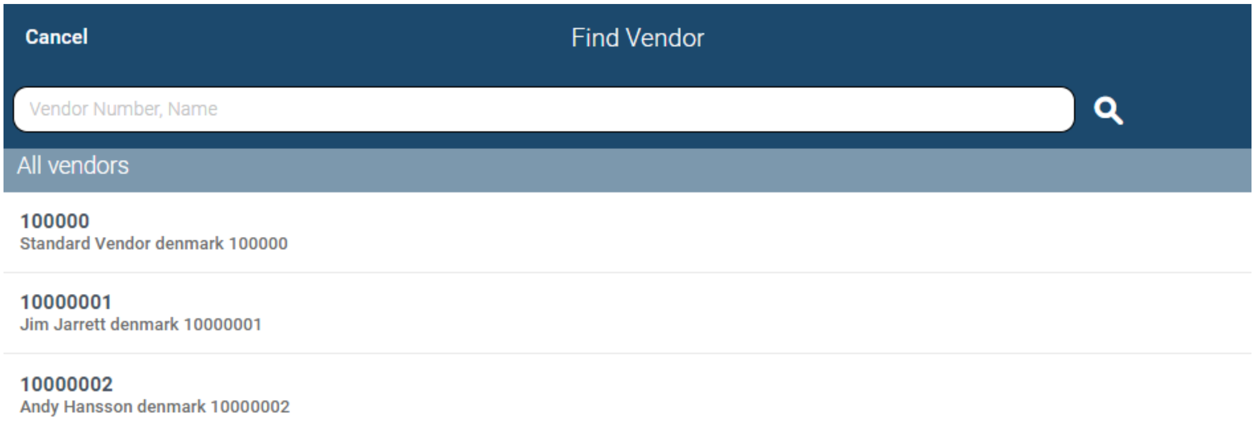
Navigation Path	REST
	Find_ExpenseSheetLineU
Expense Sheet » Expense Sheets » Expense Sheet Lines » Find Job » All	**Screen Layout:** expensesheetline.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_JobHeader
Expense Sheet » Expense Sheets » Expense Sheet Lines » Find Job » Favorites	**Screen Layout:** expensesheetline.JSON **Search Layout:** find_jobfavorite.JSON **Container:** Find_JobFavorite
Expense Sheet » Expense Sheets » Expense Sheet Lines » Find Job » Recently Used Jobs	**Screen Layout:** expensesheetline.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_ExpenseSheetLineU
Purchase Order » Purchase Orders » Purchase Order » Job » Find Job » All	**Screen Layout:** purchaseorder.JSON **Search Layout:** findjobheader_purchaseorder.JSON **Container:** Find_JobHeader

Navigation Path	REST
Purchase Order » Purchase Orders » Purchase Order » Job » Find Job » Recently Used Jobs	**Screen Layout:** purchaseorder.JSON **Search Layout:** findjobheader_purchaseorder.JSON **Container:** Find_PurchaseOrderLineU
Purchase Order » Purchase Orders » Purchase Order » Purchase Order Lines » Job » Find Job » All	**Screen Layout:** purchaseorderline.JSON **Search Layout:** findjobheader_purchaseorder.JSON **Container:** Find_JobHeader
Purchase Order » Purchase Orders » Purchase Order » Purchase Order Lines » Job » Find Job » Recently Used Jobs	**Screen Layout:** purchaseorderline.JSON **Search Layout:** FindJobHeader_PurchaseOrder.JSON **Container:** Find_PurchaseOrderLineU
Mileage Sheet » Mileage Sheet » Find Job » All	**Screen Layout:** mileagesheet.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_JobHeader

Navigation Path	REST
Mileage Sheet » Mileage Sheet » Find Job » Recently Used Jobs	**Screen Layout:** mileagesheet.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_ExpenseSheetLineU
Mileage Sheet » Mileage Sheet » Mileage Sheet Line » Find Job » All	**Screen Layout:** mileagesheetline.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_JobHeader
Mileage Sheet » Mileage Sheet » Mileage Sheet Line » Find Job » Recently Used Jobs	**Screen Layout:** mileagesheetline.JSON **Search Layout:** findjobheader_expense.JSON **Container:** Find_ExpenseSheetLineU
Mileage Sheet » Mileage Sheet » Mileage Sheet Line » Find Job » Favorites	**Screen Layout:** mileagesheetline.JSON **Search Layout:** find_jobfavorite.JSON **Container:** Find_JobFavorite

Find Vendor Screens

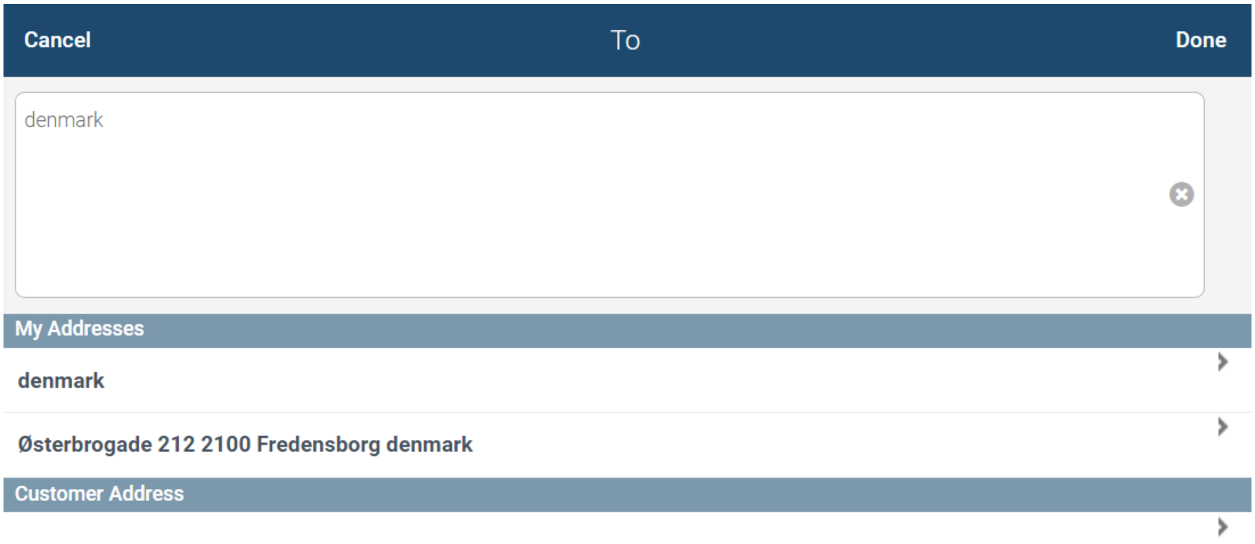
The **Find Vendor** screen allows you to look up vendors on the **Purchase Order** screen.

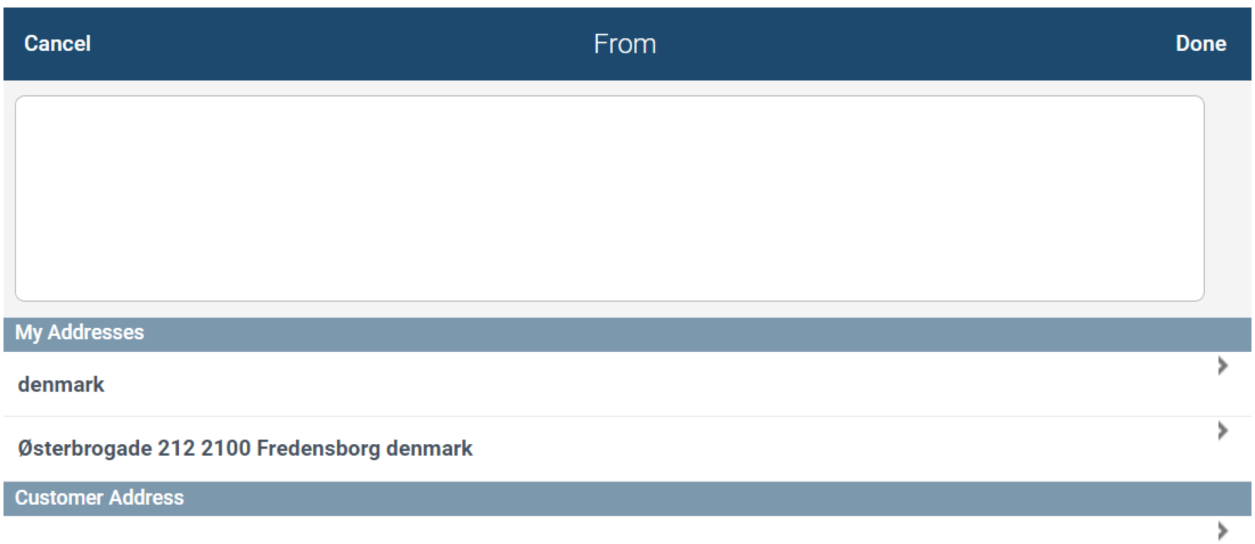


Navigation Path	REST
Purchase Order » Purchase Orders » Purchase Order » Find Vendor » Recently Used Vendor	**Screen Layout:** purchaseorder.JSON **Search Layout:** findvendor_purchaseorder.JSON **Container:** PurchaseOrders
Purchase Order » Purchase Orders » Purchase Order » Find Vendor » All	**Screen Layout:** purchaseorder.JSON **Search Layout:** findvendor_purchaseorder.JSON **Container:** find_vendor

Address Screens

The **To** and **From** search screens allow you to look up addresses you commonly use as well as any customer addresses from the **Mileage Sheet** screen.

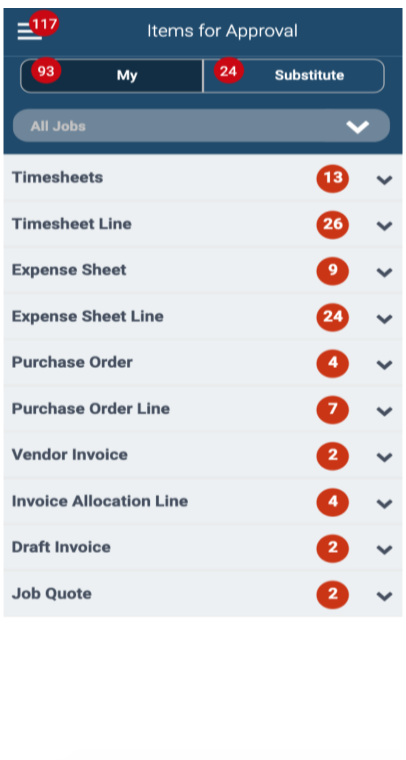




Navigation Path	REST
Mileage Sheet » Mileage Sheet » Mileage Sheet Line » From	**Screen Layout:** mileagesheetline.JSON **Search Layout:** findjobheader_expense.JSON **Container:**

Navigation Path	REST
	Find_RecentlyUsedMileageFromU
Mileage Sheet » Mileage Sheet » Mileage Sheet Line » To	**Screen Layout:** mileagesheetline.JSON **Search Layout:** find_locationfromto.JSON **Container:** Find_RecentlyUsedMileageToU

Approval Screens

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
itemsforapproval.JSON

Containers:
Timesheets –
approvetimesheetsbyemployee

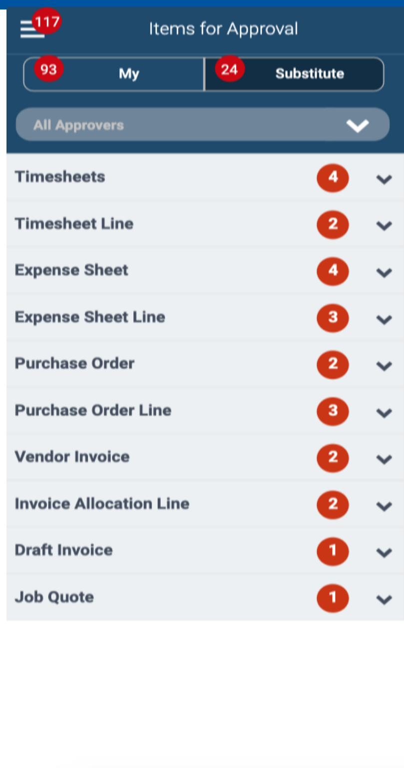
Timesheet Line –
approvetimesheetlinesbyemployee

Expense Sheets –
approveexpensesheetsbyemployee

Expense Sheet Line –
approveexpensesheetlinesbyemployee

Purchase Order –
approvepurchaseordersbyemployee

Purchase Order Line –
approvepurchaseorderlinesbyemployee</td></tr>
</table>

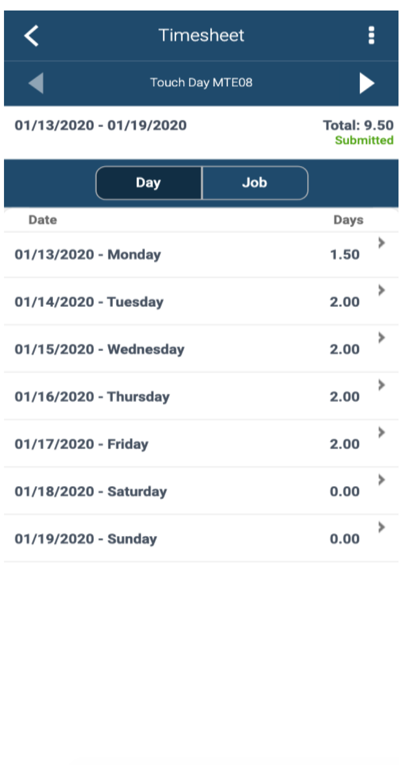
<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td>
<td>Vendor Invoice –
approvevendorinvoicesbyemployee

Invoice Allocation Line –
approvevendorinvoiceallocIinesbyemployee

Draft Invoice – approvedraftinvoicesbyemployee

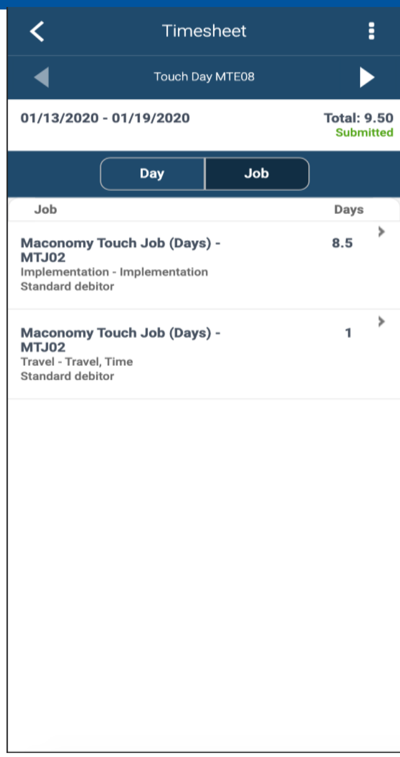
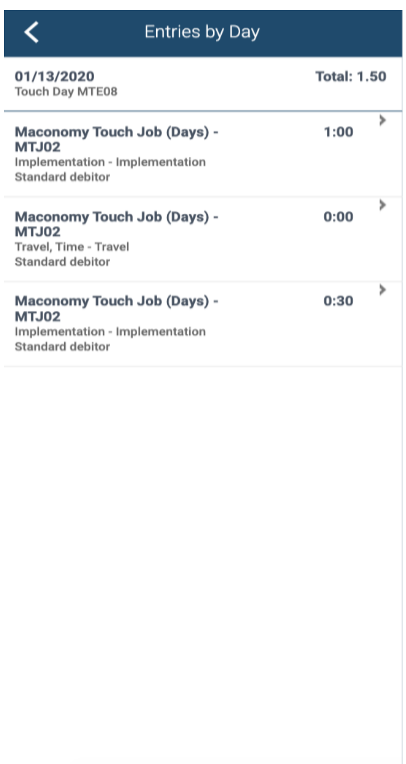
Job Quote – approvejobquotesbyemployee

Absence – absencecalendarlines

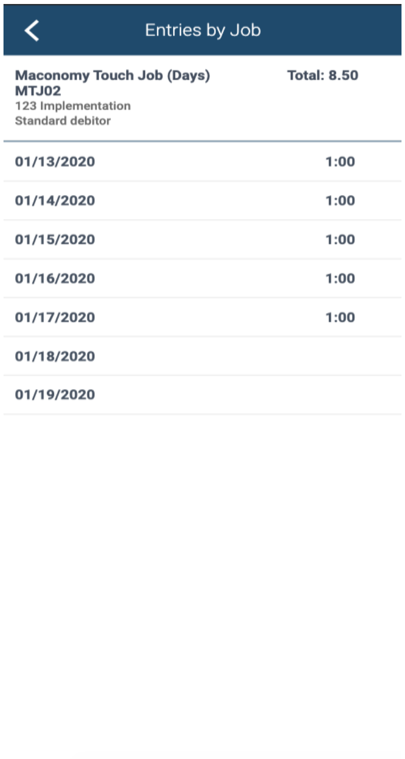
Allowance – absenceentryoverview</td></tr>
<tr><td>
</td>
<td>Layout:
approvaltimesheet.JSON

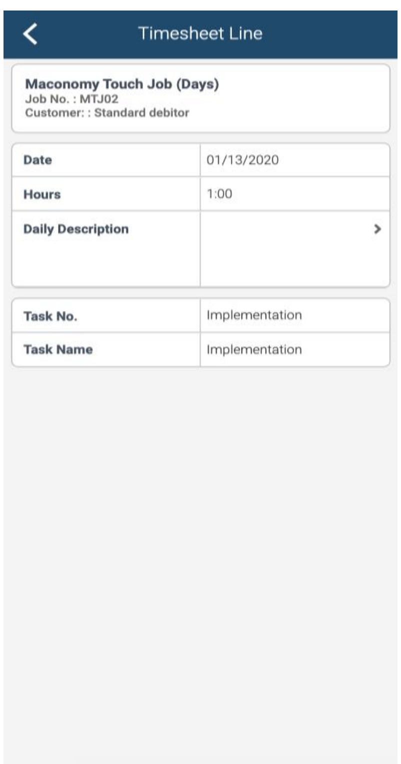
Containers:
Day Tab –
timesheets
maconomylongtext:timesheets

Job Tab –
find_timesheetlinesummaryu</td></tr>
</table>

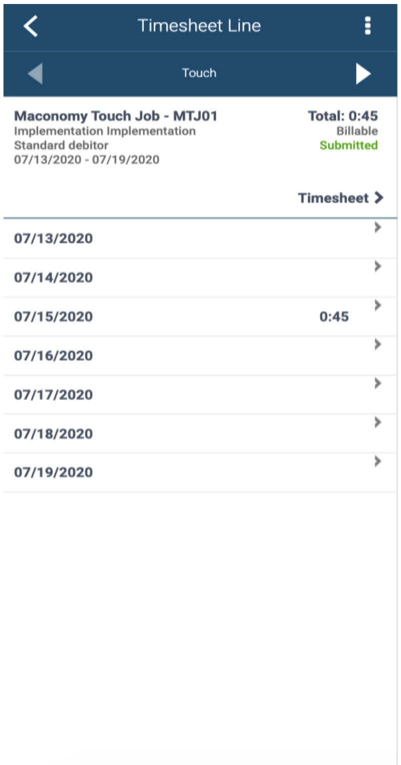
<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td></td></tr>
<tr><td>
</td><td>Layout:
approvaltimesheetentriesbyday.JSON

Container:
showpurchaseorderlines</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
Approvaltimesheetentriesbyjob.JSON

Container:
find_timesheetlinesummaryu</td></tr>
<tr><td>
</td><td>Layout:
approvatimesheetline.JSON

Container:
timesheets
maconomylongtext:timesheets</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>

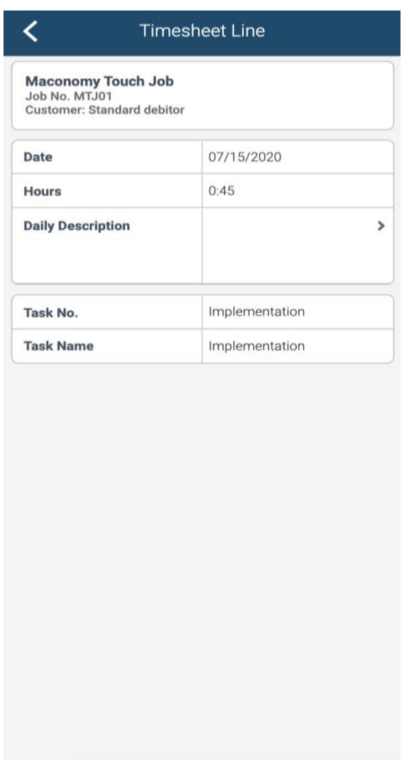
Layout:

pmtimesheetline.JSON

Container:

timeregistrationdetails

maconomylongtext:timeregistrationdetails

</td></tr>
<tr><td>
</td><td>

Layout:

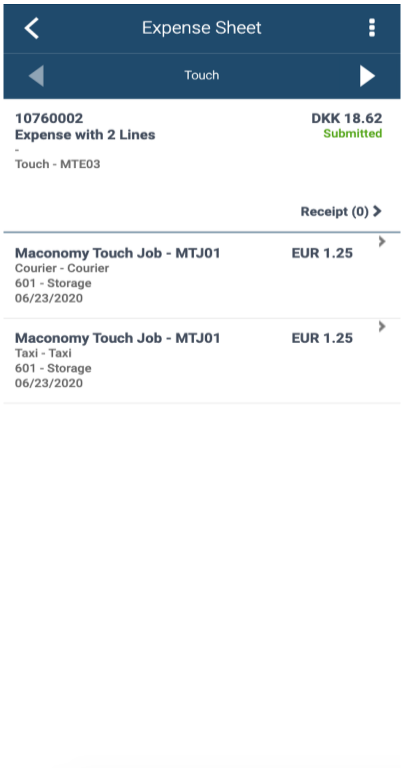
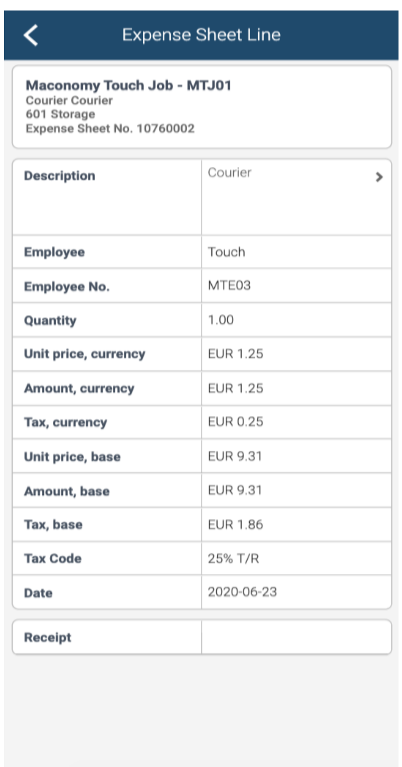
pmtimesheetlogentry.JSON

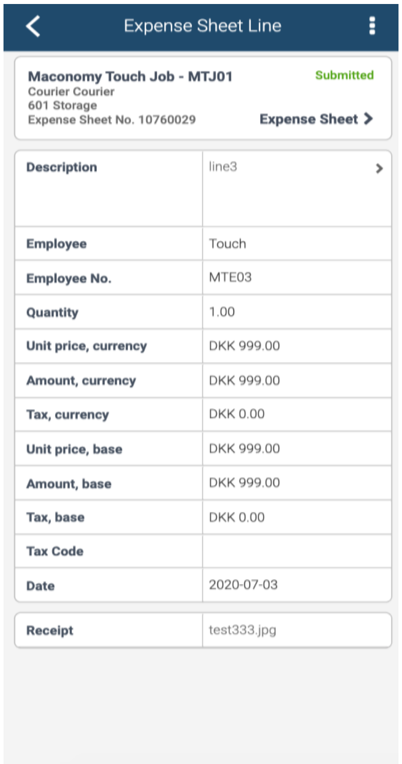
Container:

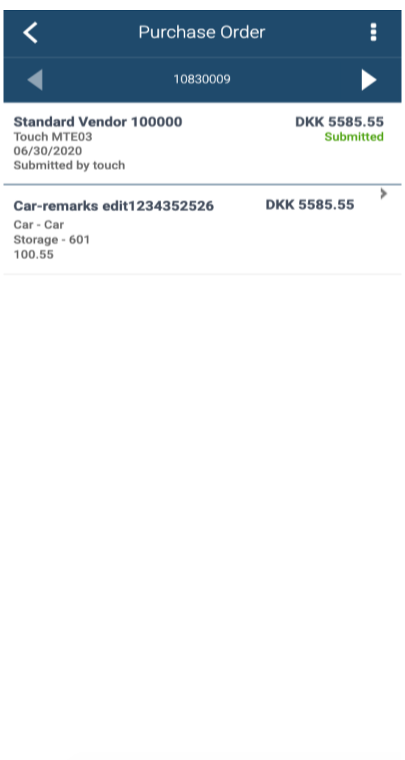
timeregistrationdetails

maconomylongtext:timeregistrationdetails

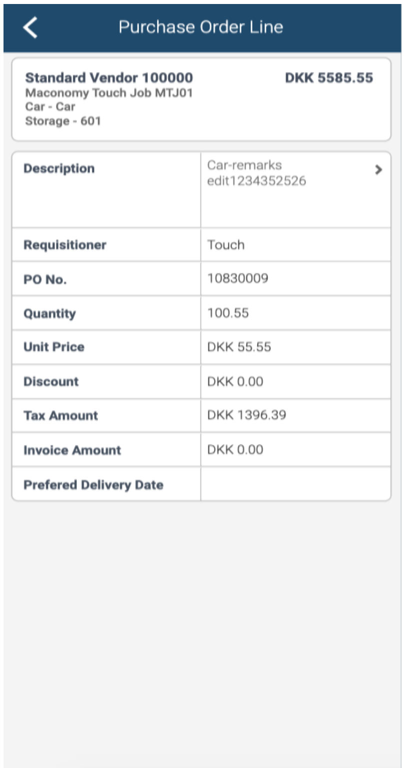
</td></tr>
</table>

Screen	REST
	**Layout:** approvalexpensesheet.JSON **Containers:** expensesheets
	**Layout:** approvalexpensesheetline.JSON **Containers:** expensesheets

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
approvalexpensesheetlinepm.JSON

Containers:
expensesheetlines</td></tr>
<tr><td>
</td><td>Layout:
approvalpurchaseorder.JSON

Containers:
purchaseorders
maconomylongtext:purchaseorders</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>


</td><td>

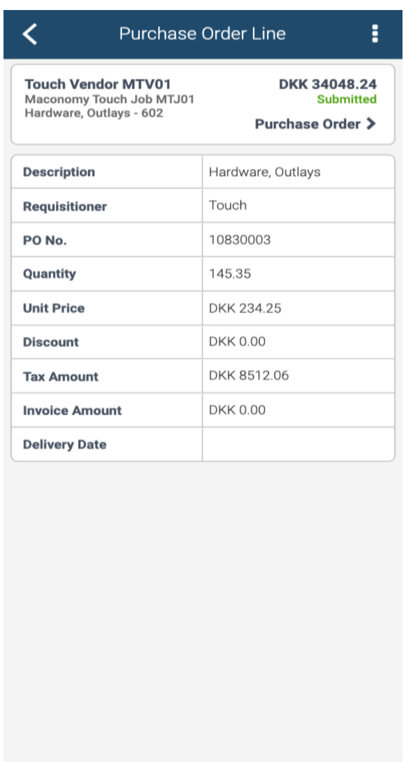
Layout:

approvalpurchaseorderlinepm.JSON

Containers:

purchaseorders

maconomylongtext:purchaseorders

</td></tr>
<tr><td>


</td><td>

Layout:

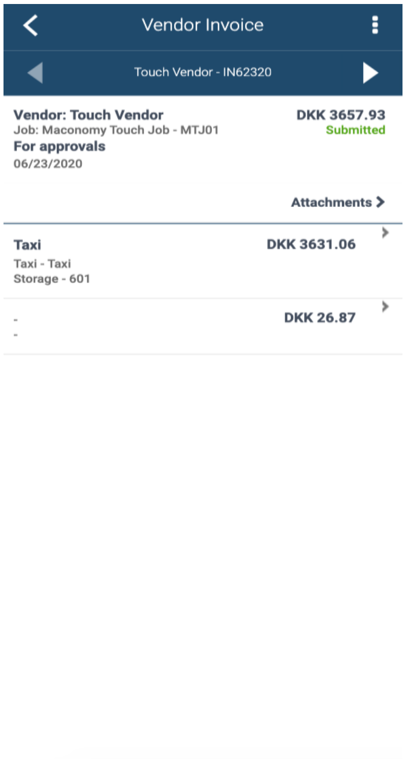
approvalpurchaseorderlineline.JSON

Container:

showpurchaseorderlines

</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>



</td><td>

Layout:

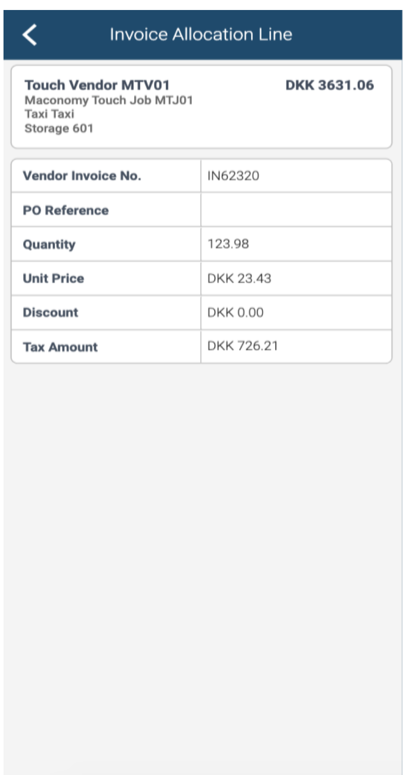
approvalvendorinvoice.JSON

Container:

invoiceallocation

maconomylongtext:invoiceallocation

</td></tr>
<tr><td>



</td><td>

Layout:

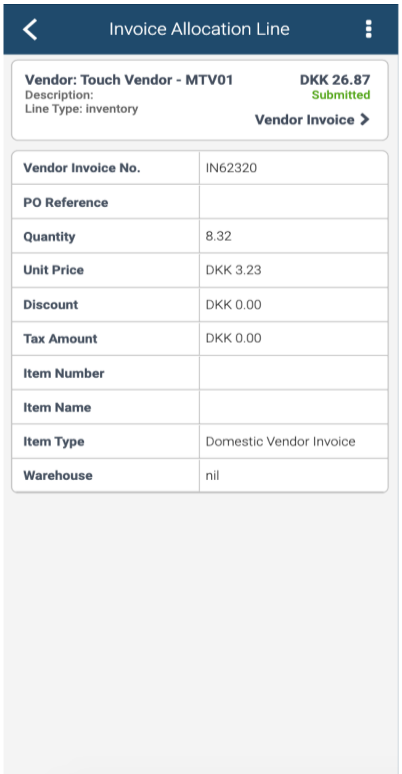
approvalvendorinvoiceallocationline.JSON

Container:

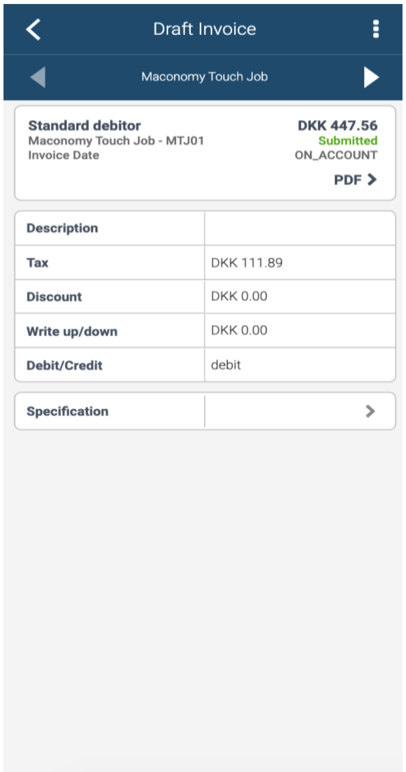
invoiceallocation

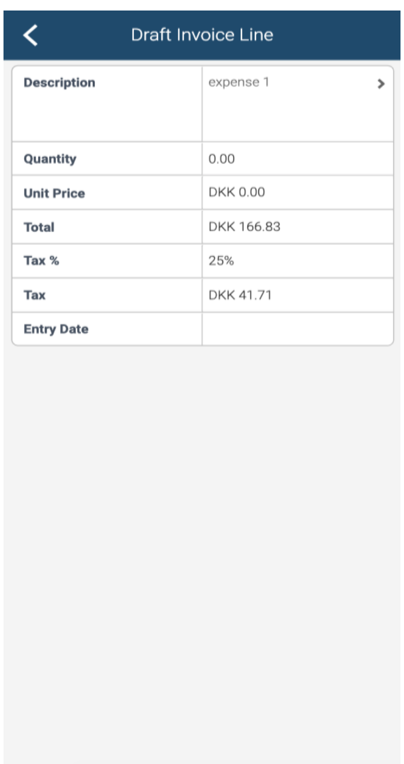
maconomylongtext:invoiceallocation

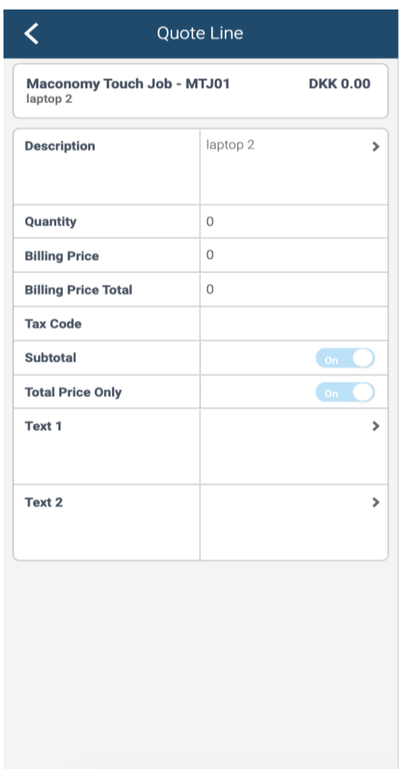
</td></tr>
</table>

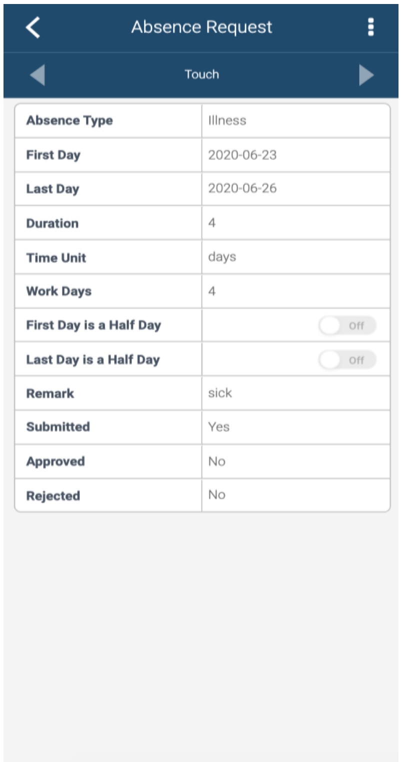
<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td>
<td>Layout:
approvalvendorinvoiceallocationlinelinepm.JSON

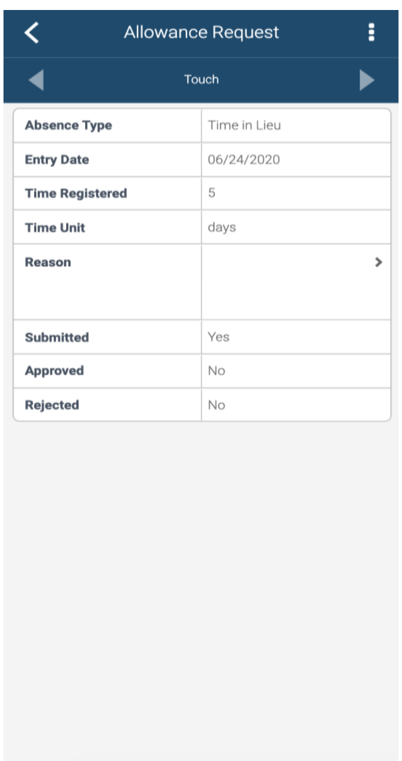
Container:
showinvoiceallocationlines</td></tr>
</table>

Screen	REST
	**Layout:** approvalinvoicedraft.JSON **Container:** invoiceediting maconomylongtext:invoiceediting

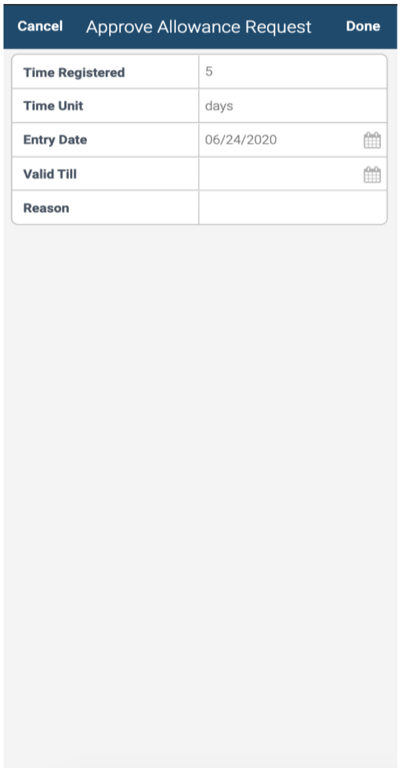
Screen	REST
	**Layout:** approvalinvoicedraftlines.JSON **Container:** invoiceediting maconomylongtext:invoiceediting
	**Layout:** approvalinvoicedraftline.JSON **Container:** invoiceediting maconomylongtext:invoiceediting

Screen	REST
	**Layout:** approvalquoteline.JSON **Container:** quoteediting maconomylongtext:quoteediting
	**Layout:** approvalquoteline.JSON **Container:** quoteediting maconomylongtext:quoteediting

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
absenceapprovalform.JSON

Container:
absencecalendarlines</td></tr>
<tr><td>
</td><td>Layout:
allowanceapprovalform.JSON

Container:
absenceentryoverview</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>


</td><td>
Layout:
allowanceapproveinputscreen.JSON

Container:
absenceentryoverview
</td></tr>
</table>

Time Registration Screens

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>

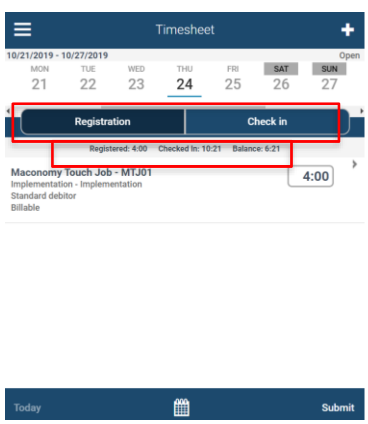

</td><td>

Layout:

timesheet.JSON

Container:

TimeRegistration

*Weekly Mode
</td></tr>
<tr><td>


</td><td>

Layout:

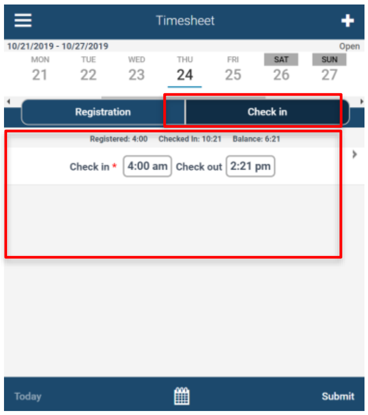
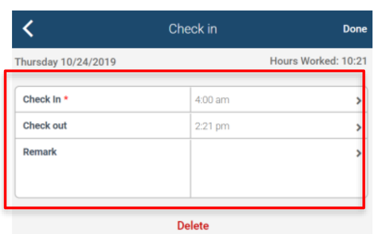
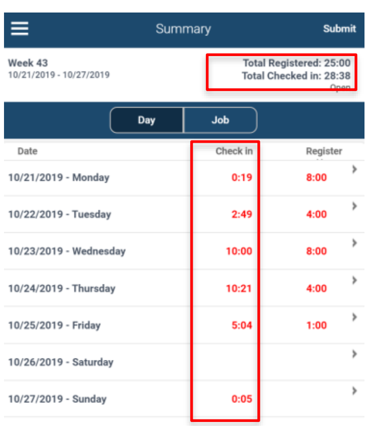
timesheet.JSON

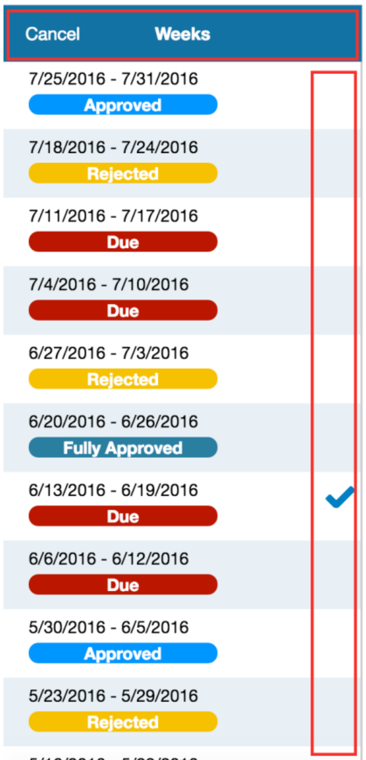
Container:

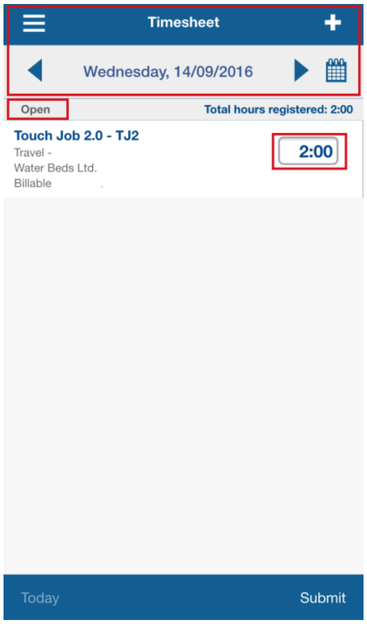
TimeRegistration

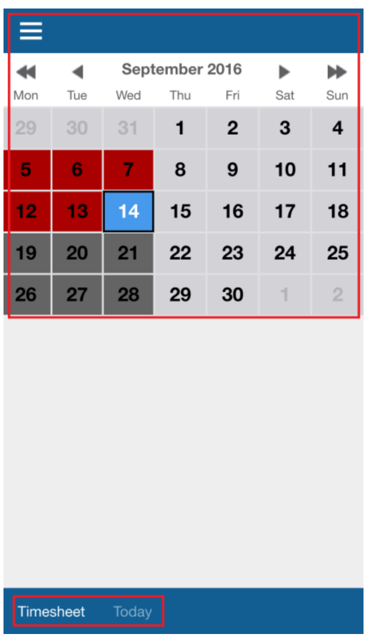
If **CheckInEnabledDay<x>Var** is **true** (where <x> refers to day 1 to day 7), the Registration and Check in tabs display, and check in is required for a given day. If it is set to **false**, there are no changes to the UI.

The Value: CheckInTimeForPeriodVar only applies if check in is required. If **CheckInBalanceForPeriodVar** is < or > 0, the total fields should be in red color.
</td></tr>
</table>

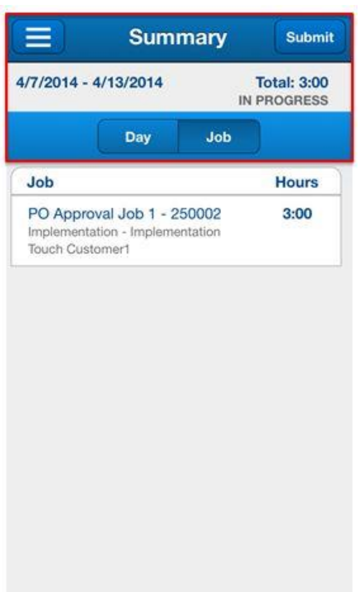
Screen	REST
	**Layout:** timesheet.JSON(for Weekly Mode) timesheetdaily.JSON (for Daily Mode) **Container:** DailyCheckIn To get from the TimeRegistration container to the DailyCheckIn, container you need to create the URL. The following are not customizable: ▪ Registered: x ▪ Checked in: y ▪ Balance: z
	**Layout:** checkin.JSON **Container:** DailyCheckIn
	**Layout:** summary.JSON (for Weekly) summarydaily.JSON (for Daily Mode) **Container:** TimeRegistration This only applies if **CheckInEnabledForPeriodVar** is set to **true.** **Total Checked In**: Value: CheckInTimeForPeriodVar If **CheckInBalanceForPeriodVar** is < or > 0, the total fields should be in red color. **Checked In** column:

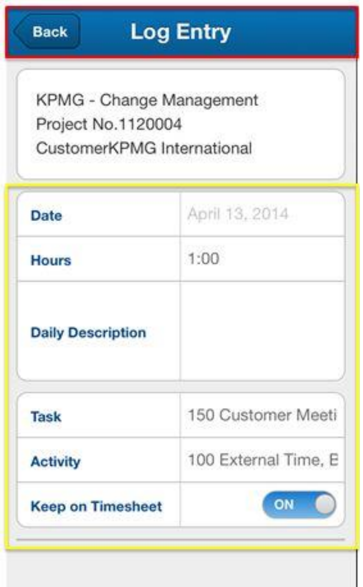
Screen	REST
	Value: CheckInTimeRoundedDay<X>var The value only displays if **CheckInEnabledDay<x>Var** is set to **true**. **If CheckInBalanceDay<x>Var** is < or > 0, the values should be in red color.
	**Layout:** timesheetweek.JSON **Container:** TimeRegistration

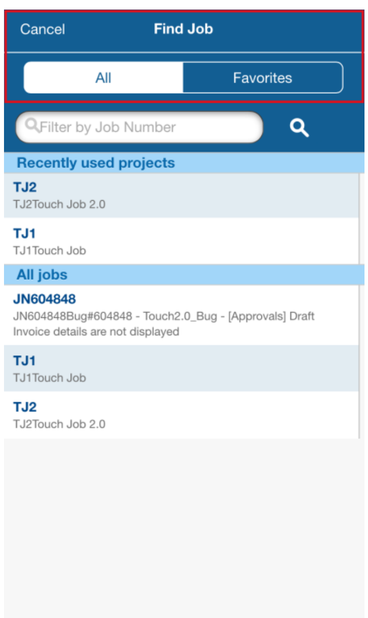
<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
timesheetdays.JSON

Container:
DailyTimeRegistration (for Daily Mode)</td></tr>
<tr><td>
</td><td>Layout:
timesheetcalendar.JSON
Only the colors for each status are customizable.

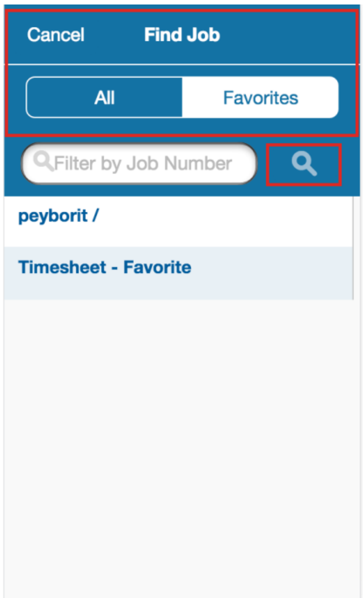
Container:
DailyTimeRegistration</td></tr>
</table>

Screen	REST
 	**Layout:** summary.JSON (for Weekly mode) summarydaily.JSON (for Daily mode) **Container:** TimeRegistration (for Weekly mode) Find_TimeSheetLineSummaryU (for Weekly mode) DailyTimeRegistration (for Daily mode) Find_TimeSheetLineSummaryU (for Daily mode)

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
timesheetlogentryweekly.JSON (for Weekly mode)
timesheetlogentrydaily.JSON (for Daily mode)

Container:
TimeRegistration (for Weekly mode)
DailyTimeRegistration (for Daily mode)</td></tr>
<tr><td>
</td><td>Layout:
find_jobheader_timesheet.JSON

Container:
Find_TimeSheetLineU
Find_JobEmployeeControlU</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>


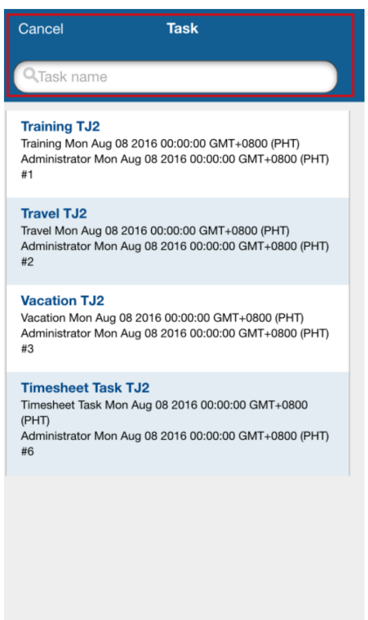
</td><td>

Layout:

find_jobfavorite_timesheet.JSON

Container:

Find_JobFavorite

</td></tr>
<tr><td>


</td><td>

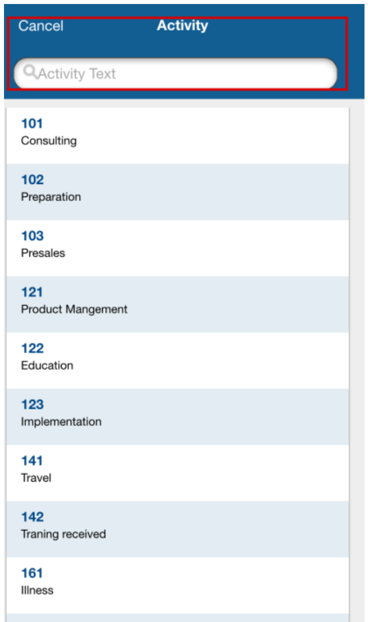
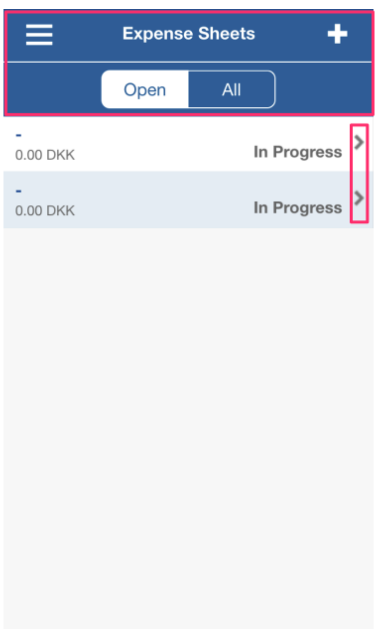
Layout:

find_tasklistline.JSON

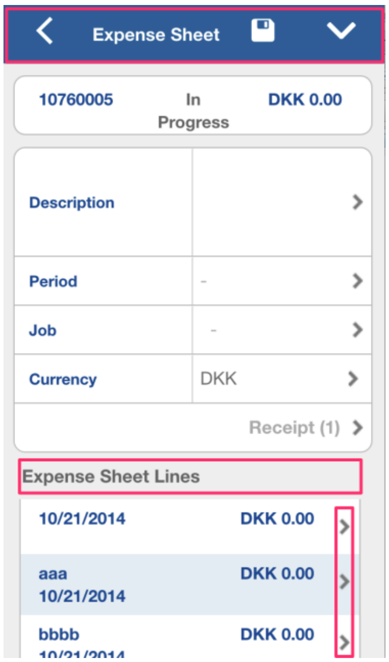
Container:

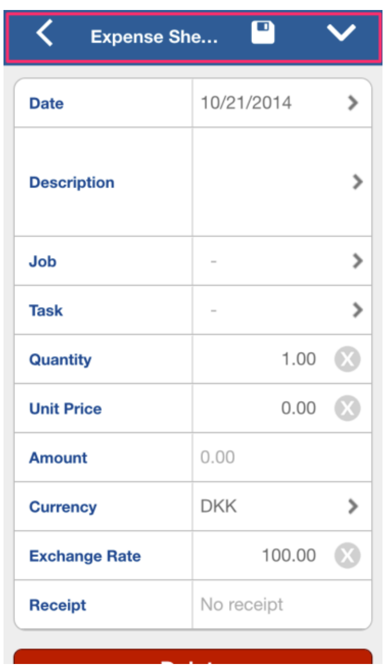
Find_TaskListLine

</td></tr>
</table>

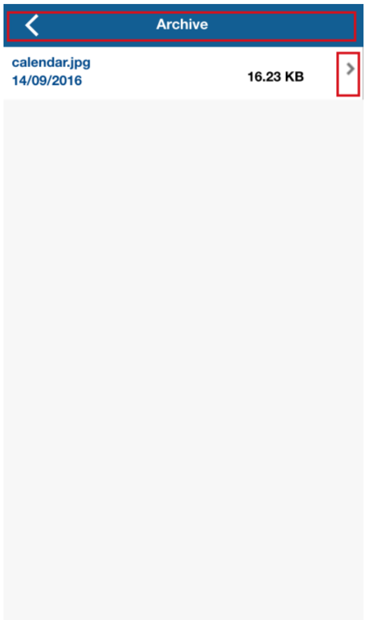
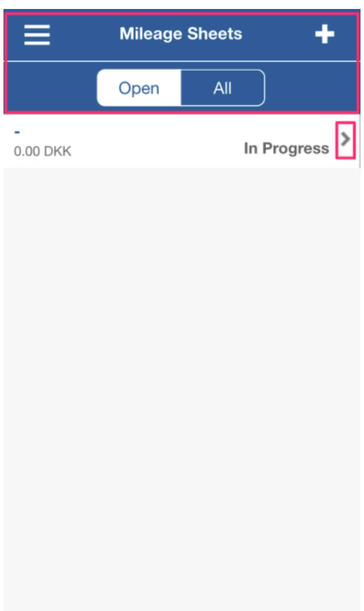
Screen	REST
	**Layout:** find_activity.JSON **Container:** Find_Activity
	**Layout:** expensesheets.JSON **Container:** ExpenseSheets

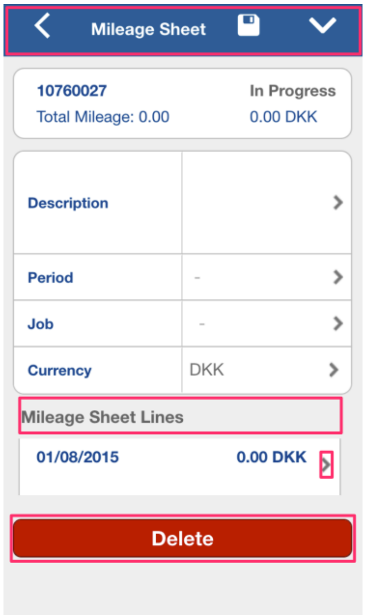
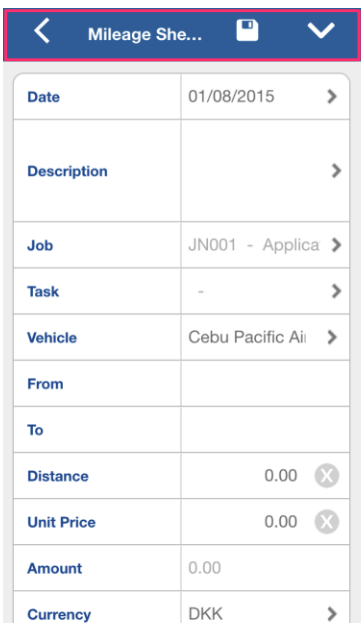
Expense & Mileage Registration Screens

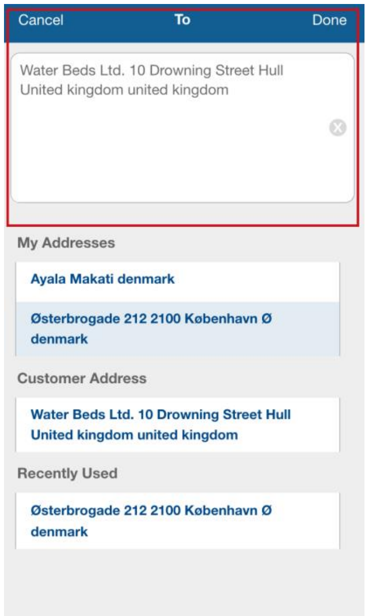
<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
expensesheet.JSON

Container:
ExpenseSheets</td></tr>
<tr><td>
</td><td>Layout:
expensesheetline.JSON

Container:
ExpenseSheets</td></tr>
</table>

Screen	REST
	**Layout:** documentarchivelist.JSON **Container:** DocumentArchives
	**Layout:** mileagesheets.JSON **Container:** MileageSheets

Screen	REST
	**Layout:** mileagesheet.JSON **Container:** MileageSheets
	**Layout:** mileagesheetline.JSON **Container:** MileageSheets

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>


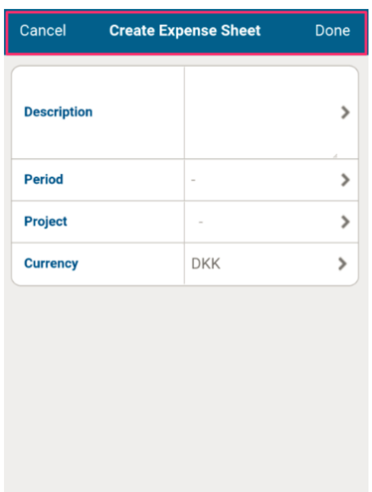
</td><td>

Layout:

find_locationfromto.JSON

Container:

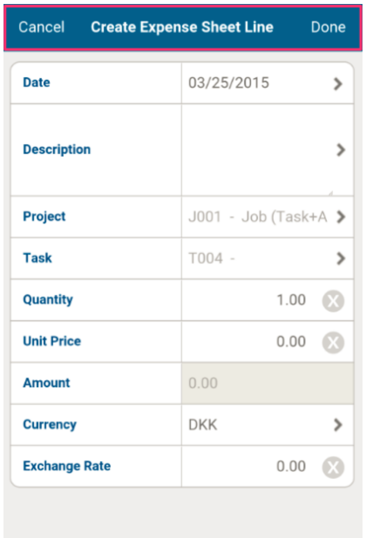
Find_RecentlyUsedMileageFromTou

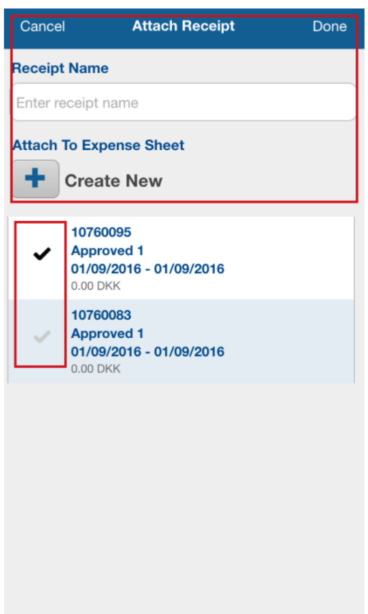
</td></tr>
<tr><td>


</td><td>

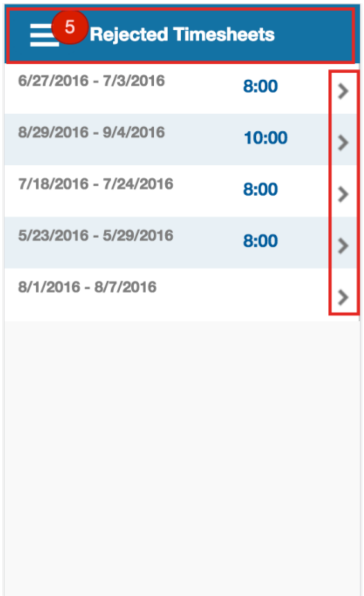
Not applicable

</td></tr>
</table>

<table>
<tr><th>Screen</th><th>REST</th></tr>
<tr><td>
</td><td>Layout:
quickcaptureexpensesheetline.JSON

Container:
ExpenseSheets</td></tr>
<tr><td>
</td><td>Layout:
quickcaptureattachreceipt.JSON

Container:
ExpenseSheets</td></tr>
</table>

Screen	REST
	**Layout:** rejectedtimesheets.JSON **Container:** Find_RejectedTimeSheetU

Appendix B: Additional Information

If you need assistance installing, implementing, or using Deltek Touch for Maconomy, Deltek makes a wealth of information and expertise readily available to you.

Customer Services

For over 30 years, Deltek has maintained close relationships with client firms, helping with their problems, listening to their needs, and getting to know their individual business environments. A full range of customer services has grown out of this close contact, including the following:

- Extensive self-support options through the Deltek Support Center
- Phone and email support from Customer Care analysts
- Technical services
- Consulting services
- Custom programming
- Classroom, on-site, and Web-based training

Attention: Find out more about these and other services from the Deltek Support Center.

Deltek Support Center

The Deltek Support Center is a support Web site for Deltek customers who purchase an Ongoing Support Plan (OSP).

The following are some of the many options that the Deltek Support Center provides:

- Search for product documentation, such as release notes, install guides, technical information, online help topics, and white papers
- Ask questions, exchange ideas, and share knowledge with other Deltek customers through the Deltek Support Center Community
- Access Cloud-specific documents and forums
- Download the latest versions of your Deltek products
- Search Deltek’s knowledge base
- Submit a support case and check on its progress
- Transfer requested files to a Customer Care analyst
- Subscribe to Deltek communications about your products and services
- Receive alerts of new Deltek releases and hot fixes
- Initiate a Chat to submit a question to a Customer Care analyst online

Attention: For more information regarding Deltek Support Center, refer to the online help available from the website.

Access Deltek Support Center

To access the Deltek Support Center:

1. Go to https://deltek.custhelp.com.
2. Enter your Deltek Support Center **Username** and **Password**.
3. Click **Login**.

Note: If you forget your username or password, you can click the **Need Help?** button on the login screen for help.

Guides & Resources

The following table lists the Deltek documentation available for this release. Except where noted, all the references listed in this table are available for download from the Deltek Support Center.

Document Name	Description
Deltek Touch for Maconomy Release Notes	This document contains a summary of the technical considerations, major features, and known issues of the application.
Touch for Maconomy Online Help for Android Touch for Maconomy Online Help for iOS	The Touch Online Help is available for both Android and iOS devices. It contains detailed information and instructions on how to use various features of the application.
Deltek Touch for Maconomy Touch Admin Guide	*This document is intended for technical consultants only.* It contains information on how to install and upgrade Deltek Touch and set up multitenancy in the application.

For Maconomy documentation and other related materials, visit the Maconomy Information Center.

About Deltek

Better software means better projects. Deltek is the leading global provider of enterprise software and information solutions for project-based businesses. More than 23,000 organizations and millions of users in over 80 countries around the world rely on Deltek for superior levels of project intelligence, management and collaboration. Our industry-focused expertise powers project success by helping firms achieve performance that maximizes productivity and revenue. www.deltek.com