



## Platform apps

### Planon Software Suite

Version: L105

© 1997 - 2024 Planon. All rights reserved.

Planon and the Planon logo are registered trademarks of Planon Software Development B.V. or its affiliates. All other product and company names mentioned herein are trademarks or registered trademarks of their respective companies. Planon Software Development B.V., its affiliates and/or licensors own the copyright to all Planon software and its associated data files and user manuals.

Although every effort has been made to ensure this document and the Planon software are accurate, complete and up to date at the time of writing, Planon Software Development B.V. does not accept liability for the consequences of any misinterpretations, errors or omissions.

A customer is authorized to use the Planon software and its associated data files and user manuals within the terms and conditions of the license agreement between customer and the respective legal Planon entity as soon as the respective Planon entity has received due payment for the software license.

Planon Software Development B.V. strictly prohibits the copying of its software, data files, user manuals and training material. However, customers are authorized to make a back-up copy of the original CD-ROMs supplied, which can then be used in the event of data loss or corruption.

No part of this document may be reproduced in any form for any purpose (including photocopying, copying onto microfilm, or storing in any medium by electronic means) without the prior written permission of Planon Software Development B.V. No copies of this document may be published, distributed, or made available to third parties, whether by paper, electronic or other means without Planon Software Development B.V.'s prior written permission.

# About this Document

## Intended Audience

This document is intended for *Planon Software Suite* users.

## Contacting us

If you have any comments or questions regarding this document, please send them to: [support@planonsoftware.com](mailto:support@planonsoftware.com).

## Document Conventions

### **Bold**

Names of menus, options, tabs, fields and buttons are displayed in bold type.



### *Italic text*

Application names are displayed in italics.

### CAPITALS

Names of keys are displayed in upper case.

## Special symbols

	Text preceded by this symbol references additional information or a tip.
	Text preceded by this symbol is intended to alert users about consequences if they carry out a particular action in Planon.

# Table of Contents

Before you start.....	6
Audience.....	6
Planon architecture.....	7
Planon architecture overview.....	7
Application server.....	7
Web server.....	7
Application management.....	8
Business object.....	8
Field definer.....	8
TSI.....	8
Layout.....	9
Concepts.....	10
App.....	10
AppBuilder.....	10
App icon color.....	10
Business rule.....	11
Component.....	11
Custom views.....	12
JAX-RS.....	13
Marketplace.....	14
Metadata.....	14
Mobile card extension.....	17
Namespace.....	18
Platform API version.....	19
Query definition.....	20
Scheduled task.....	20

Settings.....	20
Verified.....	21
Apps Management.....	22
Configuration.....	22
Licensing.....	22
App license delivery.....	23
Installing apps.....	24
App installation process.....	25
Re-installing an app.....	26
Module settings.....	27
Applying the app's configuration.....	28
Uninstalling apps.....	29
Updating apps.....	30
Manually updating apps.....	30
Automatically updating apps.....	30
Checking for and installing updates.....	32
Easily testing apps.....	32
Creating notifications.....	33
Broadcast messages.....	33
Displaying embedded content.....	33
App translations.....	34
App usage report.....	35
Apps Development.....	37
Field descriptions.....	38
AppCenter fields.....	38
Index.....	42

# Before you start...

This document provides the framework for creating apps and describes the required infrastructure for doing so.

In addition, this document explains the concepts, the requirements and procedures for creating your own app.



Planon highly values and only accepts apps that meet rigorous quality requirements. To pass validation, you should carefully read this document and adhere to the instructions.

## Audience

Although the Planon Software Development Kit aims to provide high-level information to any kind of user with a basic understanding of the Planon application, it is really meant for two specific user types:

- Developers
  - Technically qualified persons whose main goal is to build apps on top of existing Planon functionality that interact with the Planon application.
  - Mostly interested in understanding the technical requirements, details and infrastructure for creating apps.
- Application Managers
  - Functionally adept people who are less interested in understanding the technical complications.
  - Would like to understand what apps do and how they need to be (de)installed, configured or updated.

# Planon architecture

The **Planon application** is based on a multi-layered / multi-tier architecture, in which user interface(s), web interfaces/web services, business logic, database logic and integration logic are physically separated. This makes it possible to manage and maintain each layer individually with relatively low risk to the business. The locations of these components are given below:

- Mobile interfaces reside on the mobile device.
- Web interfaces/web services reside on the web server (Tomcat).
- Business logic resides on the application server (WildFly).
- Operational data resides in a relational database. Example, Oracle for on premise and MS SQL Server for both cloud / on premise. Specialized database for specific purpose, such as: MongoDB for BI/reporting.

## Planon architecture overview

For a description and illustration of **Planon software suite's** three-tier architecture, please see the [Deployment overview](#).

## Application server

The application server is a Java-based framework that implements Java EE platform APIs and provides the standard Java EE services. Planon uses **JBoss WildFly** as its application server. To further improve the server security, Planon uses a hardened and customized version of WildFly, not the out of the box version.

The application server is the place where the business logic of the Planon application resides. This application data is used by various client applications in Planon through various protocols.

## Web server

The web server is basically a web container that implements the JSP/Servlet part of Java EE. Planon uses **Apache Tomcat** to host its client interface implementation. Tomcat caters to the user interface logic and communication to the application server.

The web server uses the HTTP protocol. It receives an HTTP request and responds with an HTTP response.

# Application management

The Planon application is delivered as a preconfigured system with predefined processes. To adapt the configuration to the organization's requirements, it is possible to make changes in the application

An application manager is responsible for tweaking the configuration and shaping the organization's process and for maintaining the application for all Planon users.

## Business object

A business object (BO in short) is a logical unit of functionality and data that refers to a facility management concept, For example, work order, request, budget type, etc.

In Planon, there are two type of BOs, **system BO** and **user-defined BO (UDBO)**. A system BO offers standard sets of fields and statuses to work with. An UDBO on the other hand, is highly customizable. You can create your own fields, actions, statuses and status transitions. Still, a UDBO is always based on a system BO. For example, you can create the user-defined business objects **Customer** and **Supplier**, based on the **Personnel** business object.

BOs are managed in **Field definer**. The system BOs and UDBOs are distinguished here. Each BO or UDBO has its own set of fields and statuses. Fields contain data. For a BO, you can create any number of UDBOs.

You can authorize a business object to restrict users from using the BO. Authorizing a BO provides a means to determine if or in what way users can access data (no access, read access, read-write access) and what actions can be performed .



By default, business objects are not authorized. This means that all users have full access to these business objects: all fields are accessible and all actions can be performed.

## Field definer

**Field definer** is used to:

- Modify: fields, field properties, actions and links.
- Create UDBOs, User-defined statuses and status transitions.
- Manage settings of BO, fields, display types, actions, links, statuses and status transitions.

## TSI



A TSI (Task Specific Interface) contains selection levels and selection steps with several (predefined) business objects. A TSI offers various user groups with the required functionality to perform specific tasks and activities.

A selection level displays a BO's elements list. A selection level includes multiple layouts that determine how a BO or UDBO is presented to the end user. Each layout determines how fields, tabs, actions and status transitions are displayed. For the end users, TSIs are available in the **Navigation panel**.

## Layout

A layout helps to define how screen items are displayed to an end user. A layout allows you to choose how specific fields, tabs and actions statuses must be displayed. A BO or a UDBO can have one or more layouts. The Planon provided layout for a BO is called a base layout. If a default layout does not exist for a BO or UDBO, it will use the base layout.

Layouts are managed in **Layouts**, which enables you to configure the display and description of tabs. There are two types of layouts, base and a user-defined layout. A base layout contains more specific properties and makes it possible to filter on a field when saved.

# Concepts

The following topics describe the concepts that are key to understanding the functionality.

## App

A specific solution including configuration, consisting of a set of business rules that extend the Planon application. In addition, it can also contain configuration elements.

## AppBuilder

AppBuilder is Planon's Integrated Development Environment (IDE).

AppBuilder is a software application that provides facilities for software development. Typically, an IDE combines various functions for developing software, such as:

- An editor
- A compiler/debugger
- API documentation







AppBuilder is available in Cloud environments and in non-Production modes only.


## App icon color

The app's icon color in your AppCenter may vary depending on its status.

The app icon colors convey the following meaning:

Icon color	Description	Indicates
	Orange	Not verified, no update
	Orange with exclamation mark	Not verified and an update is available in the <a href="#">Marketplace</a> .
	Green	Verified, no update

Icon color	Description	Indicates
	Green with exclamation mark	Verified and an update is available in the <a href="#">Marketplace</a> .

 Not verified does not necessarily mean that the app no longer works. It means that it has not (yet) been confirmed by the partner for the current Planon version.

[Checking for and installing updates](#)

[Verified](#)


## Business rule

Business rules are developed to extend the Planon ProCenter functionality.

Planon ProCenter calls these rules at defined moments during the execution of the application. Once installed and activated, a business rule is an integral part of the Planon ProCenter software and works in a similar way as standard predefined business rules or other functionality as provided by Planon.

We distinguish two types of business rules:

- Predefined business rules: system business rules that cannot be configured.
- Reusable business rules: business rules that can be applied and selected in a user extension usage on any business object.

 If an app is set to *Active* / *Inactive*, you must manually enable/disable the corresponding business rules. After doing this, a cache refresh of the application is required.

## Component

A piece of functionality that is delivered by the app.

Examples of components:

- Business rules
- TSI actions
- Scheduled tasks (classes)
- Endpoints
- Decision rule classes

We distinguish two types of components:

- **Predefined** components

For these components the app developer needs to indicate (in the config.json) when, where and within which context these are deployed in Planon on activating the app (e.g. on BeforeUpdate and BeforeInsert on BO Person).

- **Reusable** components

These components can be selected by the system administrator in defining for example a scheduled task, SX, TSI action on any BO etc.

Component	Reusable	Predefined
Business rule	x	x
TSI action	x	
Scheduled task	x	
Decision rule	x	

## Custom views

App developers can implement a visual presentation (custom views) that is specific to a certain system TSI level or step.

This is possible for steps that already have the possibility to switch between the standard data grid to another view. You can register custom views for the following steps:

System name	Translated name
BaseAsset	Assets
ContractFinancialObligation	Financial obligations
Floor	Floors
MoveMoveLineStep	Move lines
Person	Persons
PlannedMaintenanceActivityStep	Maintenance activities
ProjectWBSItemStep	WBS items
ReservationUnitFlexWorkplace	Flexible workplaces
ReservationUnitInventoryItem	Asset units
ReservationUnitSpace	Spaces units
ReservationVisitors	Visitors

System name	Translated name
Space	Spaces
Visitor	Visitors
WorkOrderReservationVisitors	Visitors
Workspace	Workspaces

## Activating the custom view

When the app is set to **Active**, the custom view is available in the m-n that allows the system administrator to show/hide custom views in TSIs.

## Deactivating the custom view

When the app is set to **Inactive**, the custom view will be removed from the **Linked step actions** m-n dialog in TSIs (without affecting whether custom view is selected). The view is no longer available on the step. By reactivating the app the custom view will be available as configured before deactivating the app.

## Uninstalling the custom view

When uninstalling the app, the custom view and its configuration is removed from the application.



If uninstalling is not possible, an error message will be displayed informing that the app cannot be uninstalled.

## JAX-RS

The AppBuilder supports the JAX-RS standard for developing REST endpoints.

The generated URL will resemble the following format:

### Root

```
../services/sdk/platform/jaxrs/[partnerID]/[appname]/[modulename]/[service]
```

### SDK

```
../sdk/platform/jaxrs/{partnerID}/{appname}/{modulename}/{service}
```



After installing an app, its JAX-RS resources are shown in:

- **Apps TSI > Technical info** tab: all resources of all modules.
- Apps > Details > Artifacts > Technical info: endpoints only.

## Root vs. SDK

ROOT is *stateful* and SDK is *stateless*. As a result, if the app is meant for end users, you want it to be *stateful* and the developer should use Root.

If the app is meant for system integration, it can be *stateless* and your developer would need to use SDK.

For more (technical) information about this topic, see the [developer portal](#) (login required).

## Marketplace

Planon's Marketplace is the *store* where customers can find and get apps that are developed on the Planon Platform.


See also: [Planon Marketplace](#).

## Metadata

Metadata is used to uniquely identify an app. Metadata is available at various levels in the domain model.

### App metadata

Metadata	Description	Comment
App UUID	UUID that uniquely identifies an app across all Planon instances.	Only for Planon internal references. <ul style="list-style-type: none"><li>• Stored in database.</li><li>• Can be used</li></ul>

Metadata	Description	Comment
Partner Identifier Sub-concepts: <ul style="list-style-type: none"> <li>• customer</li> <li>• department</li> </ul>	Uniquely identifies a Partner.  Is a combination of the customer (name), and partner (for partners) or apps (for customers that build their own apps), concatenated with a dot (.).	<p>in Planon caches.</p> <p>Used in the <b>App license</b>.</p> <p>Used to identify apps during installation. Since a UUID uniquely represents an app, we can handle name changes, in theory.</p> <p><b>Restrictions</b></p> <ul style="list-style-type: none"> <li>• Limit to [a-z0-9] (minimum 3, maximum 50)</li> <li>• Exactly one dot (.)</li> </ul> <p><b>Public references</b></p> <ul style="list-style-type: none"> <li>• Visible in the <b>AppCenter</b> TSI.</li> <li>• Visible in Planon configuration (TSI).</li> <li>• Visible in URLs.</li> </ul>
App Name	Describes an app. Managed by Partner. <div data-bbox="578 1801 948 1892" style="border: 1px solid blue; padding: 5px; margin-top: 10px;">  Unique in the scope of a Partner Identifier.           </div>	The app name can change; the consequences are for the app developer.

Metadata	Description	Comment
		<p>This affects the app upgrade.</p> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <p><b>1</b> Soft references to the namespace will not be updated. However, they can be detected by looking up the App UUID.</p> </div> <p><b>Restrictions</b></p> <ul style="list-style-type: none"> <li>• Limit to [a-z0-9] (minimum 3, maximum 50)</li> <li>• No dots (.)</li> </ul> <p><b>Public references</b></p> <ul style="list-style-type: none"> <li>• Visible in the <b>AppCenter</b> TSI.</li> <li>• Visible in Planon configuration (TSI).</li> <li>• Visible in URLs.</li> </ul>

Module name		
Metadata	Description	Comment
Module name	Describes an app module. Managed by Partner.	The Module name can change, but is unlikely to;



Metadata	Description	Comment
		<p data-bbox="1094 254 1354 310">consequences are for the app developer.</p> <p data-bbox="1094 331 1321 388">This will affects the app upgrade.</p> <div data-bbox="967 409 1338 531" style="border: 1px solid black; padding: 5px;"> <p data-bbox="984 422 1321 516">Changing the module name will result in app settings being lost.</p> </div> <p data-bbox="1094 548 1252 575"><b>Restrictions</b></p> <ul data-bbox="1127 596 1300 911" style="list-style-type: none"> <li>• Limit to [a-z0-9] (minimum 3)</li> <li>• No dots (.)</li> </ul> <p data-bbox="1094 930 1338 957"><b>Publish references</b></p> <ul data-bbox="1127 978 1338 1436" style="list-style-type: none"> <li>• Visible in the <b>AppCenter</b> TSI.</li> <li>• Visible in Planon configuration (TSI).</li> <li>• Visible in URLs.</li> </ul>

## Mobile card extension


An information block configurable for mobile solutions in Planon Live.

Functionally, this information block can be put to any use, you could use it to display the local weather conditions, Google Maps, traffic information, ...

These extensions can be used for and are visible in the Mobile Configuration of web definitions.

# Namespace

A namespace is a combination of metadata elements that together uniquely identify and mount the app.

Metadata	Description	Comment
Namespace	<p>The set of:</p> <ul style="list-style-type: none"><li>• Partner identifier</li><li>• App Name</li><li>• Module Name</li></ul> <p>The goal of the namespace is to identify app solutions. And to limit search scope when an app solution searches for solutions.</p> <div data-bbox="578 1045 948 1136" style="border: 1px solid black; padding: 5px;"> The partner identifier is managed by Planon.</div>	<ul style="list-style-type: none"><li>• Stored in the Planon database per solution API implementation registration (but only if the solution API implementation is configured).</li><li>• Stored in Planon solution caches.</li><li>• When an app solution is executed (because we know to which app this solution belongs) we</li></ul>

Metadata	Description	Comment
		<p>also know which other solutions it is allowed to execute. So parts of the namespace are used as a lookup value.</p>
		<p><b>Requirements</b></p> <ul style="list-style-type: none"> <li>• lowercase alphanumerical characters [a-z].</li> <li>• alphanumerical numbers [0-9].</li> <li>• Minimum 3, maximum 50 characters.</li> </ul>


## Platform API version

The Platform API version is a semantic version (**major**, **minor** and **patch**) that indicates the version of the Platform API on which an app is built.

The Platform API version maps information about the Planon version and its components. If any of the components is changed, causing a break in functionality, a new major version of the Platform API is released. This mechanism visualizes (*maps*) the compatibility between the app and the Platform API.

## Examples

- The major version of the app and the Platform API must always be the same.
- Patches remain working on the same major version.  
An app version 2.0.1 remains compatible with Platform API 2.0.0.
- Apps are only *upwards* compatible:
  - It is possible to install an app on a higher minor version (e.g. 2.0.0 on 2.1.0).
  - It is not possible to install an app on a lower minor version (e.g. 2.1.0 vs. 2.0.0). The app may be using an API function that is not available.

 Planon currently supports the following Platform types:

- general
- mobile
- web

## Query definition

Using the query builder API server-side queries can be defined and data from different business objects can be combined in one result set.

The queries are executed server-side, the framework takes care of translating all joins and filters to database queries. The results of the query are passed to the invoker.

Using this API, the database structure is hidden from the developer and (almost) no client-side data processing (filtering, combining) is needed.

 For convenience, the query definitions used by an app will be displayed on the **Technical information** tab (on the **Apps** step and the Details > App artifacts step)

## Scheduled task

*Scheduled tasks* are more or less the same as *Business rules*. However, instead of being triggered by a **Save** action, they are triggered by a timer. Where a Business rule has a reference to a business object (the one that is being saved), the Scheduled task is completely stand-alone. A scheduled task is responsible for finding and loading the business objects to operate on.

## Settings

Module settings is used to register settings that are required in the app.

There are 2 types of settings (defined by the **Scope** field):

- Settings that apply only for the app - or rather: the module as a whole

- Settings that apply to *reusable components* (SX, TSI actions, ...) within the app. These settings will be applied wherever the reusable component is applied.



For TSI actions, the Planon application manager should specify the settings for reusable components in Field Definer > BO > Details > Extended actions.

In the IDE, the app developer defines the settings that are required within the module or reusable component.

## Settings applied

The available app settings are *converted* into individual fields that will be displayed under **Settings**.

You can toggle between the classic view (json settings, schema and example) and the field settings.

- **Settings** (Module settings > General tab): displays the settings that are applied in the app.
- **Settings example** tab: If a new version of the app containing new settings is introduced, this field is updated according to the new settings, which enables you to check which settings need to be filled out still.
- **Settings schema** tab: The type/format and constraints that apply for each setting that is present in the app (also which setting is mandatory). See also [JSON-schema.org](https://json-schema.org).

## Verified

A status that denotes that the app developer (partner) has explicitly checked the app for compatibility with a Planon Live release.

However, if an app is not (yet) verified, it does not mean that the app is not compatible or will not work. It indicates that it has not been explicitly confirmed (verified).

[App icon color](#)

# Apps Management

This section describes the guidelines and procedures for the Planon application manager.

By following the steps described here, the Planon application manager can install apps and configure the Planon application as required.

## Configuration

Before being able to install apps, some configuration of the Planon application is required.

In order to install apps, the application manager must make sure that the **AppCenter TSI** is added to the navigation panel.

### Procedure

1. Add the AppCenter TSI to the application manager's navigation panel.
  - a. Go to Web client > Navigation panels, select the application manager's user group.
  - b. Go to the Navigation panels level and select Business processes (or, if you prefer, another location).
  - c. Click Add TSI and in the TSI field, browse to and select the AppCenter TSI (APPS\_Apps).
2. Log out and log in again to apply your changes.

You have completed configuring the Planon application for importing apps.

## Licensing

App license is maintained in the **Licenses TSI**, but you can also maintain the app license directly in the **AppCenter TSI** itself.

### AppCenter TSI

Here, you can add a new license or update an existing license. The information will be shown in the app's **License info** tab.

### License sub- types

- On/Off  
A license key is required to activate an app (except for apps created by yourself - when you have an IDE license).
- Named user  
This license type restricts the number of users that are linked to the app (through their user groups).

## License main types

Each of these sub-types could be present as either the following (main) type:

- a regular license.
- a test license (can be used to test apps in a non-production environment).
- a trial license (can be used to try out apps in non-production environment - see [Free trial](#) (Marketplace)).

## Validation

- For apps implementing a JAX-RS endpoint, a check will be performed to verify that only users linked to the app that implements this endpoint can access it.
- A standard named user license check will verify whether the number of linked users does not exceed the number in the license and its expiry date.

### Example

When you link a user to an app (or change the users in a user group) and there is a named users license for the app, then the number of linked users is checked.

- Each license can have an expiry date. After this date, the app can no longer be used.

For more information, see [licensing](#).

### [App license delivery](#)

## App license delivery

Regular app licenses are now included delivered via the Planon license.

On loading the Planon license, the app licenses are created/updated and (when registered to the Marketplace) the apps will be downloaded and installed automatically.

## How it works

1. When loading the Planon license, a check is performed to see if the Planon license contains (an) app license(s).  
If this is true, a log entry is created in Log viewer for tracking purposes (Source: *App license installation*).
2. When this is done, another check is performed to verify whether a [Platform account](#) has been specified (System settings).
  - If this is not the case, the application will revert to using the CloudAdmin account.
  - If the CloudAdmin account is not available (for example for on-premise customers), the app licenses will NOT be created and a log entry is created.



- Failure to set the account will be logged in the log entry.
- The CloudAdmin account is often removed by on-premise customers. Therefore, on-premise customers should ensure that a [Platform account](#) is specified!
- For the app license delivery process to work, the Platform account must be linked to the 'EnterpriseServiceAPI' [product definition](#).

3. App licenses available in the Planon licenses are subsequently processed one by one. If an app license:
  - Exists: it will be updated.
  - Does not exist: it will be created.
4. When this is completed, the log entry will be updated to reflect what has been done.
  - If **all** app licenses are processed successfully, details are specified under a header called **Processed app licenses**.
  - If app licenses cannot be processed because of an error, this too will be reflected in the log entry.

## Licensing


# Installing apps

This topic describes how to install apps.

## Marketplace



When you are registered to the [Marketplace](#), and app licenses are present in your [Planon License](#), apps are **automatically** downloaded and installed.

 On-premise customers must ensure that - in order for this to work - the technical infrastructure needs to be in place (such as, access of the Planon server to the Internet).

### Manual installation

If you do not have access to the Internet or the Marketplace - take the following steps to install your apps:

 In this case, request the app's ppk file via Planon Support.

### Procedure

1. Go to the AppCenter TSI and click Install on the action panel.

**A dialog box appears.**

2. Browse for and select the app you want to install or drag-and-drop the app's ppk file here.

**The app will appear in the elements list.**

## App installation process

If you need to manually install an app, the app (.ppk file) is configured at run time.


Some apps may require additional configuration, make sure you have completed the required [configuration](#) steps first.

**For all apps installed, the following information is displayed:**

- Its name
- Its partner identifier
- Its version
- The [Platform API](#) versions on which it is built (multiple API versions are possible)
- The Planon version on which it is built.
- A description of its functionality

Here, you can also set the app's status to **Active/Inactive**.

Some business objects may be set **Under construction** and a cache refresh may be required when activating/deactivating apps (see [Field definer](#)).

 For some apps, it is possible that the **App account** field needs to be filled in to activate the app. Only when you configure the **Module settings** in **Details** selection level, you will be able to select the **App account** field and activate the app. For more information, see [Configuring module settings](#).

- Note that it is not possible to downgrade an app to an older version. If an app is already installed, then only the same or newer versions of that app can be installed.

**Modules** level displays the relevant information of a module.

**Details** level displays the technical details of the app:

- **Module configuration:** displays the app's JSON configuration.
- **Module settings:** displays the app's settings than can be further configured.

For more information, see [Configuring module settings](#).

- **App component settings:** displays the app's component settings.
- **Artifacts:** displays the app's artifacts (jars) that are deployed.

The configuration is applied at run time, which implies that it also available for on-premise installations.



Note, however, that clustering is not supported.

### Important

When an app has various artifacts and one of these artifacts cannot be properly deployed, the app status will be *Failed*.

## Re-installing an app

When installing an app, the process is handled by a background action.

It can happen that background actions remain queued, in which case an app is *stuck* and cannot even be removed.

To overcome this situation, please proceed as follows.

### Procedure

1. In the AppCenter, select the app whose **App installation status** is **Queued**.

**The App installation status can either be:**

- Queued
- Installing
- Success
- Failed


2. On the action panel, click **Re-install**.

**The app's ppk file is already downloaded, so only the corresponding background action - whose status is **Completed with errors**- is removed and the installation is retrIGGERED.**

# Module settings

App-specific settings can be configured on **Module settings**.

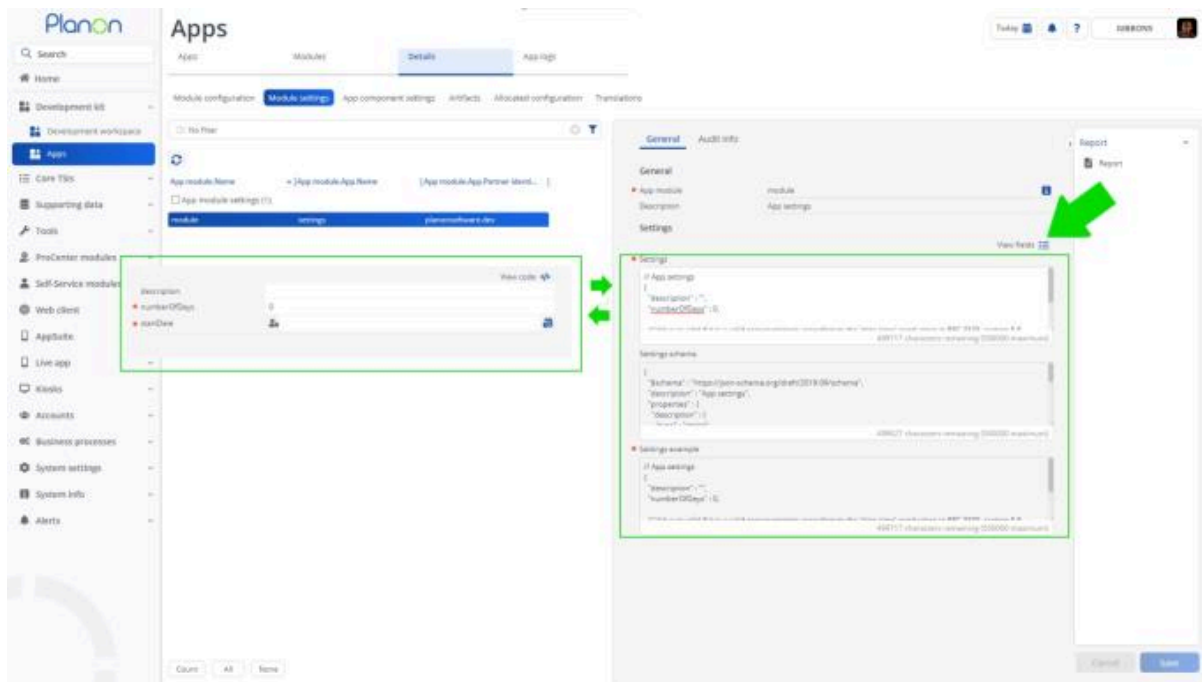
Here, the application manager can fill out the required information.

 The information that is required depends on the app.

These settings will then be applied throughout the app (where required).

## Procedure

1. When selecting the app in the elements list, the data panel will display the App module and Description.
2. Under Settings, you can fill out the required information.



## Views

By clicking the **View code/View fields** toggle, you can switch between these two views at will.

- **View fields:** provides a clean and succinct display of the required information.
- **View code:** provides an full display of the information in json format for your convenience to easily copy and reuse the settings.

## Date-time settings

Date-time settings in JSON contains the time zone.

The format is: 'YYYY-MM-DDTHH:MM:SS(+/-)HH:MM'  
where "(+/-)HH:MM" determines the UTC offset.

### List settings

list settings are shown as a read-only string field. As a result, list settings can only be entered/updated in **View code**.

In **View fields**, the values of the list are separated by a forward slash (/) and string/date-time settings are shown in between double quotes:

- **string**: "string1" / "string2" / "string3"
- **int/integer**: 1 / 2 / 3
- **bigDecimal**: 2,147.48 / 2,147 / 147.48
- **boolean**: True / False
- **Date time**: "1999-12-31T23:59:59+00:00" / "1999-12-31T23:59:59-10:00" / "1999-12-31T23:59:59+02:00"
- **Reference field**: allows you to link to a business object.

See also [AppCenter fields](#).

3. Click Save.

**After configuring the module settings, you will be able to Activate the app.**

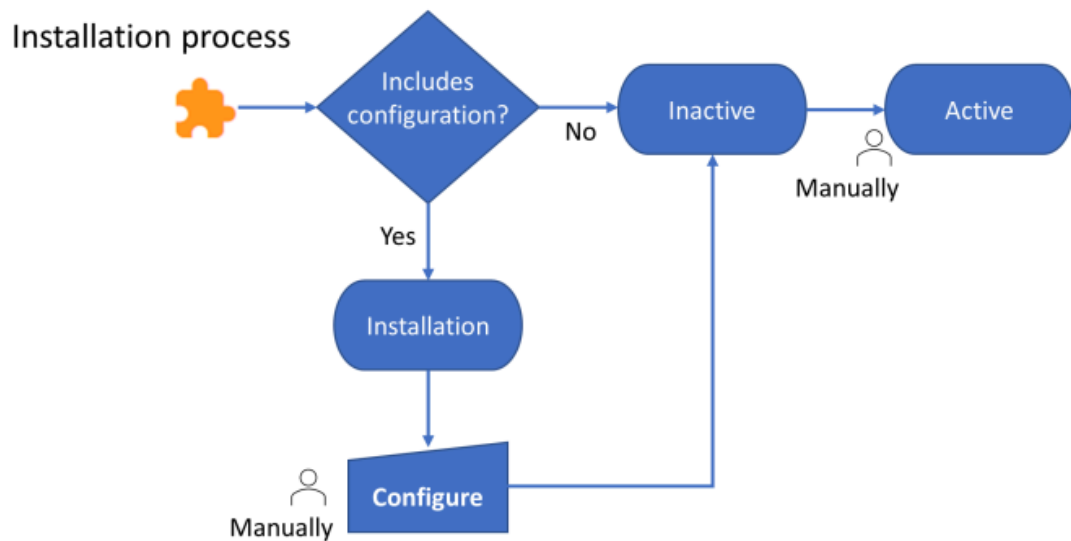
## Applying the app's configuration

Some apps come with specific configuration that is preconfigured in the app itself.

In the AppCenter, you can apply this specific configuration by using the **Configure** action.

Applying an app's configuration is part of the app's installation process and cannot be applied independently of it!

- Apps that do not have configuration, get assigned the *Inactive* status.
- Apps that have configuration, get assigned the *Installation* status.



The status model of apps distinguishes between the app's **Status** and its **Installation status**. The **Configure** action is only available if:

- The Status is *Installation* and the Installation status is *Successful*.
- The Status is *Failed* and the Installation status is *Configure*.

To apply an app's configuration, proceed as follows.

### Procedure

1. On the Apps step, select the app whose configuration you want to apply.
2. On the action panel, click Configure.

**The app's configuration will now be applied. This will take place as a background action - dependent on the app, it might take a while. Once done, the app's status changes to *Inactive*. You can then manually change the status to *Active*.**

## Uninstalling apps

This topic describes how to uninstall apps.

### Procedure

1. Go to the AppCenter TSI and click Uninstall on the action panel.

**You are prompted to confirm the action.**

2. Click OK to proceed.

The app is uninstalled from the application.

# Updating apps

You can manually install/update apps, but - for your convenience - installed apps will be automatically updated when upgrading your Planon environment.

## Manually updating apps

Updating an app is as simple as re-installing it.

### Procedure

1. Go to the AppCenter TSI and click Install on the action panel.
2. Browse for and select the app (ppk file) that you want to install/update or drag-and-drop the file here.


**When another version of the app already exists in the application, you are prompted to confirm overwriting the installed app.**

3. Click Proceed.

You have updated the app.

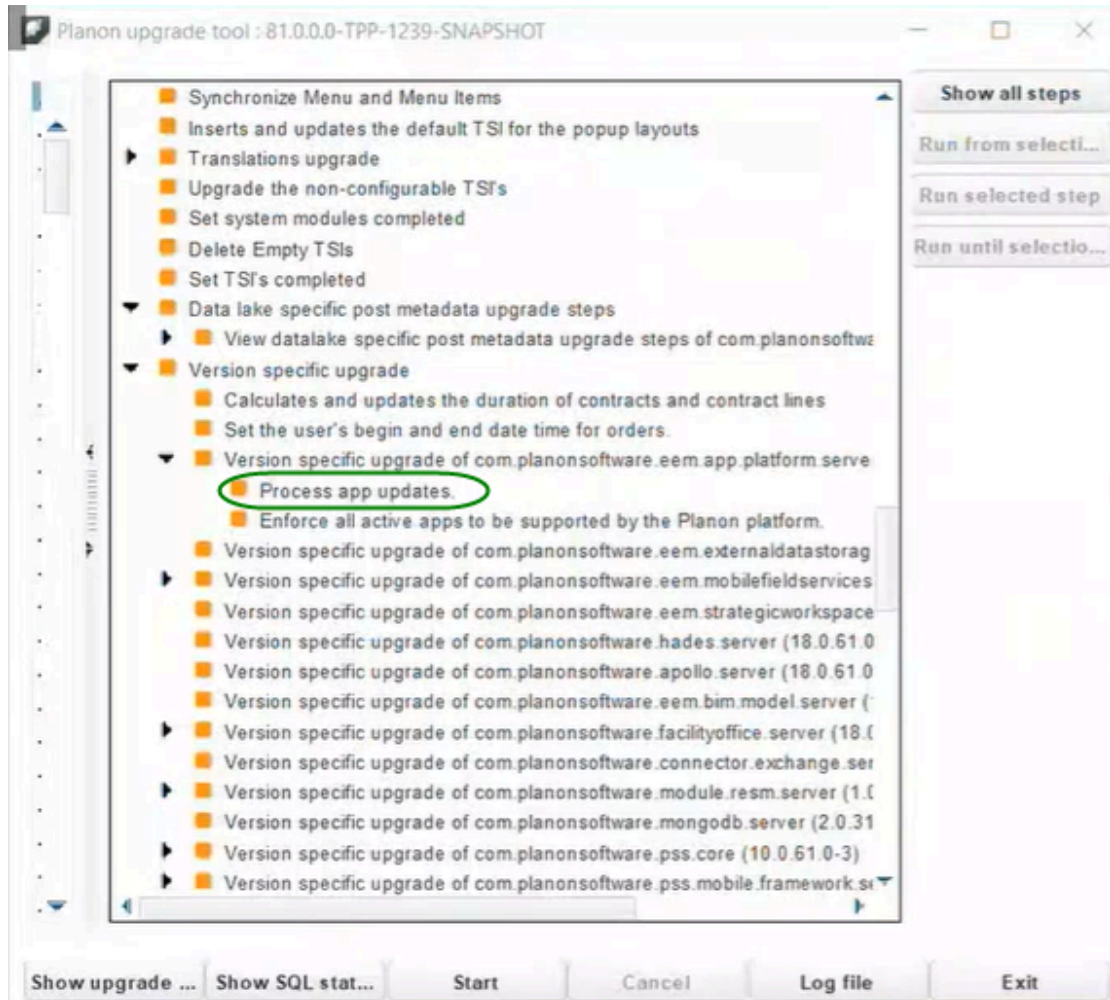
## Automatically updating apps

When upgrading your Planon environment, the apps installed in your environment will (attempted to) be updated as well if a (verified) update is available.

-  This happens automatically during upgrade and cannot be influenced.
- This only works for apps that are available in the Marketplace (it does not work for local apps).
- A connection with the Marketplace is required (see [Marketplace settings](#)).
- This only works for apps in status *Inactive* or *Active*.
- The app's status is restored after update (Active apps remain Active).

### Process

During the Planon Live upgrade, a new step verifies whether there are app updates available.



If an app update is available:

- The update will be installed.
- Any configuration that is available for apps will be applied.
- The app's status will remain as is after update (Inactive will remain Inactive / Active will remain Active).

The **Enforce all active apps to be supported by the Planon Platform** step checks if the active apps are still compatible with the Platform version (the actual compatibility). If not, the apps will be set to *Failed*.

#### Should an app update fail

- The version and status of the app will be rolled back to the previous version.
- For apps containing configuration: the app will be updated, but the status will be *Failed* (so that you can manually fix it).


## Log viewer

Per app, an entry is created in [Log viewer](#) describing the upgrade events.

## Checking for and installing updates

Whenever there is a new version available of an app, this information is registered in the Planon Marketplace.

If this is the case, you will receive a notification in your AppCenter.


 This notification will be displayed when clicking on an app for which an upgrade is available.

1. When you have installed an app in your AppCenter, you can configure and activate a scheduled task to check every 6 hours (minimum) whether an update is available.

**The name of the system task is: SYS\_APPS\_CHECK\_FOR\_UPDATE. This task will run in the background.**

2. If an update is available, the field Update available is set to Yes and an exclamation mark appears next to the app's icon in the elements list.

**If the app version is verified by the partner for this specific version of Planon ProCenter, the Verified field is set to Yes and the color of the corresponding icon will be green. If not, it is orange. For more information, see: [App icon color](#).**

 The value visible here is determined by the scheduled task and does not necessarily reflect the current status.

3. To install the update, switch the app to Inactive and click Update app in the action panel.

**The update will be installed.**

4. When the update is installed, switch the app back to Active.

[App icon color](#)

## Easily testing apps

If you are creating an app for a specific customer and you want them to download and test your new app version via the Marketplace easily (not having to send ppk files), you can now do so by using a *Test* license type.

This allows you to quickly verify a new app feature in a non-production environment.



## How it works

If ...

- you have a *Test* license type
- and a new version of an app is available, but it has not been officially approved yet
- and this new app version is compatible with your Planon version

...then you can install this version and test its functionality by clicking **Update** in the action panel (AppCenter) to install it.

## Creating notifications

You can configure a notification for alerting system administrators.

It is possible to create an alert on BO **Apps** in order to be able to alert system administrators, for example, when an app is in the *Failed* status.



For information on creating notifications, see [Alerts](#).

## Broadcast messages

A [broadcast message](#) will be displayed in the event an app cannot be activated after an upgrade.

A broadcast message will be shown to the Planon administrator group saying that "One or more apps could not be re-activated after upgrade. Please check the log viewer for details."

## Displaying embedded content

Using the iWebPage API, app developers can create an app that implements a web page (external content).

As an application manager, you can display this web page in Planon.



The app needs to be available for you to be able to select it.

### Procedure

1. In your navigation panel, go to Web client > Navigation panel.

2. Select the navigation group to which you want to add the app's web page.
3. Click Add embedded content on the action panel and in Class name, use the pop-up to select the app that implements the web page.

**The required data is automatically filled in (Partner ID, App name, App module name)**

4. Click Save and log out/log in.

The web page that is implemented by an app is now visible in the navigation panel.

## App translations

Planon Universe is deployed worldwide, hence it needs to comply with internationalization standards (i18n).

This, too, is true for apps. In the Marketplace, an app's value and deployability becomes greater if it supports translatability.

Clearly, apps that support translations potentially have a larger deployment base.

How does the Planon application support translation of these apps?

### Procedure

1. Install your app in the AppCenter.

**For apps supporting internationalization, the translations are available on a separate tab.**

2. Go to Details > Translations

**The app's translations appear in the data panel, here you can adjust them.**

- To manually provide a translation, proceed with step 3
  - To export the app's translation as part of the language file, proceed with step 4.
3. Select an item in the elements list and, in the data panel provide the proper translation for the required languages.
  4. Go to System settings > Languages and export a language.

**The exported language will appear in your browser's download location.**

5. Open the file, the app's translations are identified by its namespace (or filter on status `##0##`):

```
##A## <App namespace>.<ResourceKey> ## PlatformApp ##<Translation status> ##<Translation text> ##Z##
```

You can translate these keys in the required language(s) and import them.



- Note that the app's translation follow the Planon application's logic regarding export and import. For more information, see [Languages](#).
- For a video about this subject, see the [Planon video channel](#).
- It is possible for partners/developers to specify a locale and by doing so get translations in another language than the logged in user's. By using this mechanism, you can only retrieve app translations. For more information, see the [developer portal](#) (credentials required).

## App usage report

A report that provides an overview of [components](#) implemented by apps.

The report information can be useful to customers - it provides an overview of all app components and where they are located so that they can be tested after upgrading or after moving an instance in a DTAP street (from DEV to ACC).

How to access the report:

1. In **AppCenter**, on the action panel, click **Report**.  
The **Reporting** window opens
2. Click the **System reports** tab and select the **App usage report**.
3. Click **Preview & print** to view the output.

The report consists of multiple sections describing each component in detail:

- List of apps
- Extensions (server)  
Business rules and the BOs for which these are configured.  
Type P stands for Predefined, Type C stands for Reusable.
- Extended actions (TSI actions)
- JAX-RS endpoints

#### Extensions (server)

BO system name	Event	Type	Active	App name	Class name
Person	BU	P	yes	teamapiexamples	BRAccessSystemSettings (com.planonsoftware.team.api.examples.businessrule.BRAccessSystemSettings)
Person	BU	P	yes	teamapiexamples	BRVerifyTranslations (com.planonsoftware.team.api.examples.businessrule.translations.BRVerifyTranslations)
Property	AU	P	yes	teamapiexamples	BRCreateOutboundMessageWithPushStr (com.planonsoftware.team.api.examples.eventconnector.outbound.BRCreateOutboundMessageWithPushStr)
Property	AU	P	yes	teamapiexamples	CreateOutboundData (com.planonsoftware.team.api.examples.eventconnector.outbound.CreateOutboundData)
PSSActivity	BD	R	yes	test	PredefinedBR (businessrules.PredefinedBR)
PSSActivity	BI	C	yes	test	ReusableBR (businessrules.ReusableBR)

#### Extended Actions

BO system name	Layout Name	App name	Class name
PSSActivity		teamapiexamples	DummyTSIAction (com.planonsoftware.team.api.examples.tsiaction.DummyTSIAction)
PSSActivity	PSSActivity	teamapiexamples	EmptyAction (com.planonsoftware.team.api.examples.tsiaction.EmptyAction)

# Apps Development

This section describes the technical requirements and details for creating apps.

- Planon will deliver a fully prepared environment for apps development.
- The environment only works in Development mode - not in Test/Accept/Production!

## Prerequisites

To be able to access this functionality, the following requirements apply:

- The required license (PP0020s Integrated Development Environment) is loaded
- The **AppBuilder** TSI (Development workspace) is added to the navigation panel

- These configuration steps are only required once.
- For in-depth information on creating and testing your apps, please see the dedicated developer's site about Platform Apps! Here, you can access additional resources, such as technical documentation and a community forum: .

# Field descriptions

The following section(s) describe(s) the fields, their purpose and meaning.

## AppCenter fields

The AppCenter fields list the app details.

### Apps level

---

Field	Description
<b>General</b>	
Name	Displays the app name.
Partner identifier	Displays the app developer's partner or group identifier.
App identifier	Displays the app ID, typically the partner identifier, suffixed with the app name.
<b>Version</b>	
Version	Version of the app. The version must comply with semantic versioning ( <a href="https://semver.org/">https://semver.org/</a> ), or an error will be displayed. The first digit displays the major version number.
Built on	The Planon application version on which the app is developed.
Platform API version	The Platform API version on which the app is tested and operates.
<b>Status</b>	
System status	The app status. When the installation of an app fails, both the status and this field is set to <b>Failed</b> . When

Field	Description
App installation status	<p>installed successfully, the status is set to <b>Inactive</b>.</p> <p>Indicates the app's installation status.</p>
<b>Advanced settings</b>	
App account	<p>Select the account associated with the app. This account might be required when the app executes functionality whereby no account can be retrieved (such as: the logged on user, the account that created a scheduled task, etc.).</p>
App log level	<p>Indicate the log level for the app:</p> <ul style="list-style-type: none"> <li>• Debug (show debug, info, warning and error logs)</li> <li>• Info (show info, warning and error logs)</li> <li>• Warning (show warning and error logs)</li> <li>• Error (show error logs only)</li> </ul>
Auto-reactivate	<p>Indicates if an app is automatically (re)activated when a rebuild / or a backup restore is done on a non-production environment.</p> <p>When restoring a backup from Production to Develop/Test/Acceptance, active apps are deactivated <i>unless</i> explicitly expressed otherwise in the app.json file (by the app developer).</p>

## Modules level

This level displays the app's modules.

---

Field	Description
Name	Displays the app's module name.
App	Displays the app name.

---

## Details level

---

Field	Description
<b>Module configuration</b>	
App module	Displays the app's module name.
Configuration	Displays the module's configuration.
<b>Module settings</b>	
You can either toggle between <b>View fields</b> or <b>View code</b> .	
See also <a href="#">Module settings</a> .	
Settings	Displays the settings that are applied in the app.
Settings schema	The type/format and constraints that apply for each setting that is present in the app (also which setting is mandatory).  See also <a href="https://json-schema.org">JSON-schema.org</a> .
Settings example	If a new version of the app containing new settings is introduced, this field is updated according to the new settings, which enables you to check which settings need to be filled out still.
Multiline settings	Here, app developers can define up to 2 string settings to appear as a multiline text box in the 'Fields' view of the Platform app settings.

---

## App logs level



Displays the logging details of the selected app.

# Index

## A

- App 10
- App fields 38
- App icon color 10
- App license
  - Delivery 23
- App metadata 14
- App translations 34
- App updates 32
- App usage report 35
- App UUID 14
- Application management 8
- AppName 14
- Apps development 37
- Apps installation 24
- Apps management 22
- Apps TSI 22
- Audience 6, 6

## B

- Before you start 6
- Broadcast message 33
- Business object 8
- Business rule 11

## C

- Checking for updates 32
- Component 11
  - concepts 10
- Configuration 22
- Configure action 28
- Custom view 12

## D

- Deployment overview 7

## E

- Exporting language 34
- External content 33

## F

- Field definer 8
- Field descriptions 38

## I

- Icon color 32
- IDE 10, 37
- Installing an app 25
- iWebPage API 33

## J

- JAX-RS 13

## L

- Layout 9
- License type
  - Test 32
- License validation 22
- Licensing 22

## M

- Major 19
- Marketplace 14
- Minor 19
- Mobile card 17
- Mobile card extension 17
- Module settings 27

## N

- Namespace 18
- Notifications 33

## P

- PartnerID 14
- Patch 19
- Planon architecture 7
- Planon architecture: Application server 7
- Planon architecture:overview 7
- Planon architecture:Web server 7
- Planon Live upgrade
  - App update 30
- Platform API version 19
- Predefined 11

## Q

- Query definition 20

## R

- Re-install 26
- Requirements 6
- REST api 13
- Reusable 11

**S**

- Scheduled task 20
- settings 20
- System task 32

**T**

- Test license 32
- Testing apps 32
- TSI 8

**U**

- Uninstalling an app 29
- Updating apps 30
- Upgrading apps
  - Manually 30

**V**

- v8 13
- Verified 21
- version mapping 19
- View option 12

**W**

- What happens... 25