Tasks

Release 15.4.1.1

CONTACT Software

29.10.2018

Inhaltsverzeichnis

KAPITEL 1

Einleitung

Mit *CONTACT Tasks* und seinem übersichtlichen Cockpit verfügen Anwender über eine komplette persönliche Übersicht aller anstehenden Aufgaben und Prüfpunkte mit direktem Zugang zu den wichtigsten Kontextinformationen und den Sign-off-Funktionen.

KAPITEL 2

Aufgabenklassen

Alle Objekte mit Aufgabencharakter (im Weiteren „Aufgaben“) können von *CONTACT Tasks* angezeigt werden. Dazu müssen die Objektklassen eingerichtet werden. Dies kann im Menübaum über *Administration/Konfiguration → Konfiguration → Aufgaben → Aufgabenklassen* erfolgen.

Felder des Dialogs zum Anlegen einer Aufgabenklassenzuordnung

Name Eindeutiger Name der Zuordnung. Dieser wird als Primärschlüssel verwendet und sollte nach Möglichkeit ein englischer Bezeichner ohne Leer- und Sonderzeichen sein. Behandeln Sie diesen Namen wie eine technische Systemkonstante.

Klassenname Der Name einer Objektklasse aus dem Data Dictionary. Stellen Sie sicher, dass die Klasse für die REST API freigeschaltet ist.

Objektregel Hier kann eine Objektregel referenziert werden, welche die Menge der anzuzeigenden Objekte einschränkt. Es sollte festgelegt werden, welche Objekte aus der zugeordneten Klasse als persönliche, aktive Aufgaben des Anwenders betrachtet werden sollen. Beispielsweise ist eine Projektaufgabe für einen Anwender persönlich interessant, wenn der Anwender als Verantwortlicher der entsprechenden Aufgabe eingetragen ist und der Aufgabenstatus „Bereit“ oder „Umsetzung“ ist.

Ein Objekt sollte anhand dieser Konfiguration nur in genau *eine* Aufgabenklasse fallen. Stellen Sie sicher, dass alle Kombinationen aus *Klassenname* und *Objektregel* im System sich nicht überschneidende Objektmengen bilden.

Aktivitätenkontext Name einer PowerScript Reference mit Kardinalität 1, die für alle Objekte definiert, aber nicht zwingend auflösbar sein muss. Ist hier ein gültiger Name eingetragen, wird das aufgelöste Beziehungsobjekt als Kanal für die zur Aufgabe angezeigten Aktivitäten verwendet. Ansonsten ist die Aufgabe selbst der Kanal.

Fälligkeit Name eines nativen Datumsattributs der Objektklasse, das das Fälligkeitsdatum enthält. Die Angabe des Fälligkeitsattributs ist optional. Ist kein Attribut angegeben, besitzt keine Aufgabe dieser Klasse ein Fälligkeitsdatum.

Tipp: Die Darstellung eines Aufgabenobjekts und dessen Funktionalitäten, wie z.B. Erledigung, sind abhängig von der tatsächlichen Objektklasse. Bitte wenden Sie sich für die Konfiguration zusätzlicher Objektklassen in *CONTACT Tasks* an Ihren zuständigen Projektleiter bei CONTACT Software.

2.1 Zuordnung von Operationen

CONTACT Tasks bietet im Tabellenkopf auf den aktuell selektierten Aufgaben verfügbare Operationen an. Hier stehen alle für die Web UI freigeschalteten Operationen des Operationskontextes *CsTasksModal* zur Verfügung.

Objekte, die im Detailbereich in Beziehungsblöcken dargestellt sind, werten den Operationskontext *CsTasksDetailReference* (klassenunabhängig) aus.

KAPITEL 3

Spaltendefinition

CONTACT Tasks stellt aktuelle Aufgaben in einer Tabelle dar. Die angezeigten Spalten sind konfigurierbar. Im Auslieferungszustand bringt *CONTACT Tasks* diese Spalten mit:

Aktiv?	System?	Name
Ja	Ja	(Selektionsspalte)
Ja	Ja	Ungelesen?
Ja	Nein	Typ
Ja	Nein	Name
Ja	Nein	Status
Ja	Ja	Fällig am
Ja	Nein	Tags
Nein	Nein	Verantwortlich
Nein	Nein	Priorität
Nein	Nein	Aufwand
Nein	Nein	Fortschritt [%]
Nein	Ja	Überfällig?

Nicht als „aktiv“ gekennzeichnete Spalten können optional eingeblendet werden, sind initial aber nicht eingeblendet.

„Systemspalten“ sind konfiguriert, enthalten aber von *CONTACT Tasks* errechnete Werte. Sie können für diese Spalten zwar *Attributabbildungen* (Seite 6) anlegen, diese werden jedoch immer überschrieben und haben somit keinen Effekt.

Anwendungsmodule können weitere Spalten, wie z.B. „Projekt“ mitbringen. Über *Administration/Konfiguration → Konfiguration → Aufgaben → Spalten* können Sie auch eigene Spalten einrichten.

Felder des Dialogs zum Anlegen einer Spalte

Beschriftung Eindeutiger Name der Spalte, gleichzeitig ID eines existierenden Labels. Nach Möglichkeit sollte der Name englisch sein und keine Leer- oder Sonderzeichen enthalten.

Überschrift Die Spaltenüberschrift ergibt sich automatisch anhand des Spaltennamens und der aktuellen Sprache aus den System-Beschriftungen.

Tooltip Optionales Label, das einen erklärenden Text für die Spalte bereitstellt, wenn der Mauszeiger länger darüber stehen bleibt.

Komponente Name einer Frontend-Komponente wie in der Registry von cs-web-components-base registriert. Diese muss zur Laufzeit von *CONTACT Tasks* registriert sein. In der Regel können hier also nur Komponenten aus `cs-web-components-base` und `cs-taskmanager-web` verwendet werden.

Tipp: Details zu Bedeutung und Inhalt einiger mitgelieferter Spalten finden Sie im Anwenderhandbuch.

KAPITEL 4

Attributabbildung

CONTACT Tasks stellt Aufgaben verschiedener Typen in einer Tabelle dar. Dazu ist es erforderlich, die Abbildung der Attribute eines Aufgabentyps den Spalten der Tabelle zuzuordnen. Diese Abbildungen können über das Register *Attribute* einer Aufgabenklasse festgelegt werden.

Felder des Dialogs zum Anlegen einer Attributabbildung

Aufgabe ID einer Aufgabenklasse.

Spalte Die *Spalte* (Seite 4), der das Attribut zugeordnet werden soll.

Attribut Name eines an Objekten dieses Aufgabentyps vorhandenen Attributs, einer PowerScript Reference oder einer Methode. Im Zusammenhang mit PowerScript References der Kardinalität 1 kann, ähnlich wie bei Objektregeln, durch Punkte getrennt auf Attribute des Beziehungsobjekts zugegriffen werden, z.B.: `Task.Project.cdb_project_id`.

Einige der mitgelieferten Spalten benötigen komplexe Werte. Für diese existieren in der Klasse `cs.taskmanager.mixin.WithTasksIntegration` einige Methoden, die alle „getCsTasks…“ heißen. Im Kapitel *Daten für mitgelieferte Komponenten* (Seite 12) finden Sie für mitgelieferte Komponenten Informationen zum erwarteten Datenformat.

Sie können Methoden für komplexe Werte überschreiben oder eigene verwenden. Die Signatur der Methode muss auf jeden Fall den Parameter `request` (ggf. mit `None` initialisiert) vorsehen.

Bemerkung: Das Attributfeld von Statusspalten sollte in jedem Fall die Methode `getCsTasksProceedData` enthalten. Diese wird von *CONTACT Tasks* ggf. mit dem zusätzlichen Parameter `targets` aufgerufen, um ein asynchrones Nachladen möglicher Zielstatuswerte zu ermöglichen.

KAPITEL 5

Kontext und Beziehungen

5.1 Kontext

Zu jeder Aufgabenklasse wird immer der Detailblock „Kontext“ angezeigt. Ist kein besonderer Kontext definiert, besteht dieser nur aus dem jeweiligen Aufgabenobjekt selbst. Falls doch, wird der erste aus der Konfiguration errechnete Objektpfad angezeigt.

Dafür können Sie eine Reihe von Beziehungen (PowerScript References der Kardinalität 1) für eine Aufgabenklasse angeben, die in der Reihenfolge ihrer aufsteigenden Positionen ausgewertet werden (angezeigt werden sie allerdings in umgekehrter Reihenfolge, was der Objekthierarchie entspricht):

Am Beispiel einer Checkliste wird dies verdeutlicht:

- Position 10: „ParentChecklistItem“
- Position 20: „Checklist“
- Position 30: „Task“
- Position 40: „Project“

Für eine Checkliste wird zunächst die erste Beziehung, „ParentChecklistItem“, ausgewertet. Wird hier ein Objekt identifiziert, werden die folgenden Beziehungen auf diesem Objekt ausgewertet. Sind für alle Beziehungen Objekte identifizierbar, lautet der Kontext der Checkliste „C“ also:

1. C.ParentChecklistItem.Checklist.Task.Project
2. C.ParentChecklistItem.Checklist.Task
3. C.ParentChecklistItem.Checklist
4. C.ParentChecklistItem
5. C

Ist die Checkliste weder einer Aufgabe noch einer anderen Checkliste zugeordnet, sähe der Kontext nur so aus:

1. C.Project
2. C

Felder des Dialogs zum Anlegen eines Kontexteintrags

Aufgabenklasse ID einer Aufgabenklasse.

Position Ganze Zahl, die für die ausgewählte Aufgabenklasse eindeutig sein sollte. Sie gibt die Auswertungsreihenfolge der Kontextelemente für Aufgaben dieser Klasse vor.

Powerscript Referenz 1 Name einer an Aufgaben dieser Klasse existierenden PowerScript Reference der Kardinalität 1.

5.2 Beziehungen

Zu einer Aufgabe können beliebig viele Beziehungen als Detailblöcke angezeigt werden. Für jede konfigurierte Beziehung (in Reihenfolge ihrer aufsteigenden Position), die für das jeweilige Aufgabenobjekt nicht leer ist, wird ein Block angezeigt.

Felder des Dialogs zum Anlegen einer Beziehungszuordnung

Aufgabenklasse ID einer Aufgabenklasse.

Position Ganze Zahl, die für die ausgewählte Aufgabenklasse eindeutig sein sollte. Sie gibt die Anzeigereihenfolge der Beziehungsblöcke für Aufgaben dieser Klasse vor.

PowerScript Reference Name einer an Aufgaben dieser Klasse existierenden PowerScript Reference.

Beschriftung ID eines Labels, das für die Überschrift des Blocks verwendet wird.

Objekte in Beziehungsblöcken bieten außerdem alle für das Web UI aktivierten Operationen im Operationskontext *CsTasksDetailReference* an.

KAPITEL 6

Plugins

Die Darstellung bestimmter Elemente kann mit Hilfe von Plugins selbst gestaltet werden. Plugins sind dabei im System bekannte Frontend-Komponenten.

Einer Aufgabenklasse können mehrere Plugins zugeordnet werden. Die Zuordnung geschieht über den Diskriminator, der dem Namen der Klasse entsprichen muss. Tatsächlich ausgewertet werden aber nur bestimmte Plugin-IDs.

6.1 cs-tasks-desc

Komponente, die innerhalb des Blocks „Beschreibung“ gerendert wird. In der Regel wird hier der textuelle Inhalt der Aufgabe als Langtext dargestellt, oft aber auch während der Bearbeitung nutzbare Interaktionen, wie z.B. die Änderung des Fertigstellungsgrads.

6.2 cs-tasks-custom-status-cell

Komponente, die die reguläre für Statuswerte verwendete Komponente für diese Klasse ersetzt. Sie wird für Klassen benötigt, die entweder keinen Object Lifecycle besitzen oder bei denen dieser nicht ausschlaggebend für den Anzeigestatus in *CONTACT Tasks* ist.

6.3 cs-tasks-custom

Komponente, die den Inhalt eines zusätzlichen, optionalen Blocks bereitstellt. Ein Anwendungsbeispiel wäre etwa ein Block für Aufwandsbuchung zur Aufgabe.

Die Komponente sollte eine Funktion `getLabel` besitzen, die die Beschriftung des Blocks vorgibt:

```
function MyComponent(props) {
    return (
        <div>
            This is an example
        </div>
    );
}
```

```
MyComponent.getLabel = function(relship) {
    return (
        relship && (
            <span>
                {relship.get('mappedName')}
            </span>
        )
    );
}
```

6.4 Eigene Beziehungsblöcke

Beziehungen, für deren Name ein Plugin mit ID `cs-tasks-custom-relship` und Diskriminator gleich des Beziehungsnamens existiert, verwenden das Plugin an Stelle der normalen Beziehungskomponente. Diese Plugins sind nicht an der Aufgabenklasse selbst zu konfigurieren, sondern an Beziehungen zu den Klassen.

Dies kann z.B. verwendet werden, um eine verschachtelte Beziehung „flach“ anzuzeigen, wie im Fall von Mappen an Workflow-Aufgaben.

Wie auch bei eigenen Blöcken sollten Komponenten für eigene Beziehungsblöcke eine Funktion `getLabel` mitbringen.

6.5 Beispiel: Eigenes Plugin für eine Aufgabenbeschreibung hinzufügen

Im folgenden Beispiel ersetzen wir die Aufgabenbeschreibung für Projektaufgaben durch eine eigene Komponente.

Bemerkung: Im Beispiel wird das Kundenmodul „customer.plm“ verwendet. Ersetzen Sie dieses durch ihr jeweiliges Kundenmodul.

6.5.1 Schritt 1: Anlegen einer neuen Web-Anwendung

Legen Sie eine neue Web-Anwendung in Ihrem Kundenmodul an, die ihre angepassten Plugins für *CONTACT Tasks* enthalten wird: `webmake.exe create customer.plm --templates cs.taskmanager:cs-tasks-desc`

Die neue Anwendung ist anschließend in Ihrem Kundenmodul im Ordner `customer/plm/js/src`. Die Komponente `customer-plm-TaskDescription` ist ab sofort in der cs.web Registry verfügbar.

6.5.2 Schritt 2: Ändern der Plugin-Konfiguration

1. Navigieren Sie im Windows-Client zu *Administration/Konfiguration* → *Konfiguration* → *GUI* → *Web UI* → *Plugins* und suchen Sie nach dem Plugin mit ID `cs-tasks-desc`.
2. Klicken Sie auf den Reiter „Konfigurationen“. Sie sehen dort die für die einzelnen Aufgabenklassen existierenden Plugins.
3. Ändern Sie das Plugin für den Diskriminator `cs_tasks_cls_pcs_task`:
   (a) Ändern Sie „React-Komponente“ von `cs-tasks-pcs-task-plugin-PCSTaskDetails` zu `customer-plm-TaskDescription`.

(b) Löschen Sie den Eintrag im Register „Libraries“ und erstellen Sie einen neuen Eintrag mit Library `customer-plm`, Version `15.3.0` und der Modulzuordnung `customer.plm`.

6.5.3 Schritt 3: Testen und iterative Anpassung

Ihre neue Aufgabenbeschreibung sollte nun im Detailbereich angezeigt werden, wenn Sie eine Projektaufgabe in *CONTACT Tasks* selektieren.

Wollen Sie weitere Anpassungen vornehmen, editieren Sie `TaskDescription.jsx` und rufen Sie anschließend `webmake webpack customer.plm` auf, um die Änderungen für die Laufzeit im Web-Browser vorzubereiten.

KAPITEL 7

Daten für mitgelieferte Komponenten

Die folgende Tabelle zeigt die durch die Attributkonfiguration im Backend zu liefernden Wertformate. Ggf. erfolgt durch *CONTACT Tasks* noch eine automatische Konvertierung, bevor die Frontend-Komponente die Werte tatsächlich verarbeitet. Die Spalte `None?` gibt an, ob ein leerer Wert zulässig ist.

Bemerkung: Ein + vor einem Schlüsselwert bedeutet, dass dieser zwingend erforderlich ist.

Komponente	None?	Format
CalculatedPrioCell	Nein	{prio: int, tooltip: str}
DateCell	Ja	datetime.datetime
IconCell	Ja	{+icon: str, tooltip: str}
ObjectCell	Ja	{@id: str, +system:icon_link: str, system:ui_link: str, +system:description: str}
OverdueCell	Ja	bool
PrioCell	Ja	str
ProceedCell	Ja	{current: StatusCell value, targets: [StatusCell values]}
ReadStatusCell	Ja	int
StatusCell	Nein	{custom_renderer: str, data: {priority: int, +status: int, +label: str, +color: str, mandatory: [{key: value}]}}
TagsCell	Ja	[str]

KAPITEL 8

Wertebereiche

Die folgende Tabelle zeigt alle nicht frei wählbaren zulässigen Wertebereiche der Komponenten an.

Komponente	Schlüssel	Wertebereich
CalculatedPrioCell	prio	0 - 100
IconCell	icon	Statische URL einer Bilddatei
ObjectCell	@id	REST URL des Objekts
ObjectCell	@system:icon_link	Statische URL einer Bilddatei
ObjectCell	@system:ui_link	Frontend-URL des Objekts
PrioCell		cs_tasks_prio_high, cs_tasks_prio_medium, cs_tasks_prio_low
ReadStatusCell		0 - 1
StatusCell	custom_renderer	Name einer Frontendkomponente
StatusCell	color	Hexadezimal RGB, z.B. Rot: #FF0000

KAPITEL 9

Filterbare Kontextklassen

Um den Filter nach Kontextobjekten um eigene Klassen zu erweitern, müssen Klassen einzeln freigeschaltet werden.

Dazu kann ein Eintrag in der Klasse `cs_tasks_context` angelegt werden. Sie finden die Klasse im Navigationsbaum unter *Administration/Konfiguration → Konfiguration → Aufgaben → Aufgabenkontexte*.

KAPITEL 10

Einstellungen

Einstellungen für *CONTACT Tasks* und die Aufgabentabelle werden als *Benutzereinstellungen* (siehe Administrationshandbuch *CONTACT Elements: Administration und Konfiguration*) gespeichert. Als Standareinstellungen gelten dabei die Einstellungen für den Benutzer `cs.taskmanager.dflt`.

10.1 Einstellungen für *CONTACT Tasks*

Unter `setting_id='cs.taskmanager'` und `setting_id2='settings'` sind für *CONTACT Tasks* selbst relevante Einstellungen abgelegt.

Der Wert muss gültiges JSON sein und die folgenden Schlüssel enthalten:

notificationInterval Intervall in Millisekunden, in dem innerhalb laufender Sitzungen nach neuen oder bereits abgeschlossenen Aufgaben gesucht wird. Werte, die kleiner als `5000` sind, werden als `5000` interpretiert.

Optional werden noch diese Schlüssel ausgewertet:

size Breite des Detailbereichs als gültiger Wert der CSS-Eigenschaft „width“.

10.2 Tabelleneinstellungen

Einstellungen für die Tabelle finden Sie unter `setting_id='cs.webcomponents.table'` und `setting_id2='cs-taskmanager-web-tasks-table'`.

Der Wert muss gültiges JSON sein. Zur genaueren Beschreibung lesen Sie bitte die Dokumentation der Tabellenkomponente in `cs.web`.

10.3 Aktualisierung mitgelieferter Tabelleneinstellungen

Anwendungsmodule, die Plugins für *CONTACT Tasks* mitbringen, können auch Tabelleneinstellungen mitbringen. Diese werden beim Update in die Standardeinstellungen aufgenommen und - sofern sie nicht bereits existieren - in Anwendereinstellungen.

Sie können diesen Mechanismus auch manuell aufrufen:

```
from cs.taskmanager import settings
settings.update_all_settings()
```

Abbildungsverzeichnis

Tabellenverzeichnis