Dunnington Software Development Incorporated

Moving At The Pace Of Integrity!

Briefing-Dokument: Erweiterungen der Constellation UI

Executive Summary

Dieses Dokument fasst die zentralen Methoden zur Erweiterung und Anpassung der Benutzeroberfläche von Pega Constellation zusammen. Die Optionen reichen von Low-Code-Lösungen mit schneller Wertschöpfung bis hin zu anspruchsvollen, entwicklerintensiven Ansätzen für vollständig maßgeschneiderte Benutzererlebnisse.

Die Erweiterungsmöglichkeiten lassen sich wie folgt kategorisieren:

- **Constellation-Portale:** Die standardmäßige Low-Code-Option zur schnellen Erstellung von Back-Office- und Self-Service-Anwendungen mit vordefinierten Vorlagen.
- **Constellation Web Embed:** Eine Methode, um Pega-Workflows und UI-Komponenten nahtlos in bestehende externe Websites zu integrieren. Sie verwendet moderne HTML5-Webkomponenten und ermöglicht eine thematische Anpassung an das Branding der Host-Seite.
- **Constellation DX Components:** Eine Erweiterung der Constellation-UI-Bibliothek, die es React-Entwicklern ermöglicht, zweckspezifische, wiederverwendbare UI-Komponenten zu erstellen. Diese Komponenten integrieren sich nahtlos in die Low-Code-Autorenerfahrung von App Studio, bringen jedoch Kosten in Form von Entwicklungszeit und Wartungsaufwand mit sich.
- **Constellation SDKs:** Toolsets für Frontend-Entwickler, um Pega-Anwendungen mit einem alternativen, vom Kunden bevorzugten Designsystem (z. B. Angular Material, Material UI) zu erstellen. Sie nutzen die ConstellationJS Engine für verbesserte Leistung und Qualität.
- **Constellation DX API:** Eine Reihe von REST-API-Endpunkten, die die höchste Flexibilität bieten. Sie ermöglichen es erfahrenen Entwicklern, eine vollständig benutzerdefinierte Frontend-Erfahrung auf Basis eines beliebigen Technologiestacks zu entwickeln, indem sie direkt auf die Case-Daten und UI-Metadaten von Pega zugreifen.

Die Wahl der richtigen Methode hängt von den spezifischen Geschäftsanforderungen, dem gewünschten Grad der Anpassung, den verfügbaren Entwicklerkompetenzen und dem Investitionsrahmen ab.

Dunnington Software Development Incorporated

Moving At The Pace Of Integrity!

Analyse der Erweiterungspunkte für die Constellation UI

Das Constellation Design System von Pega bietet eine umfassende Bibliothek von UI-Komponenten und eine vordefinierte Architektur, die darauf ausgelegt ist, die Entwicklungszeit für Anwendungen zu verkürzen. Da jedoch jeder Pega-Kunde einzigartige Anforderungen hat, werden mehrere Erweiterungspunkte angeboten, die ein Spektrum von Low-Code-Konfiguration bis hin zu vollständiger UI-Anpassung abdecken.

Überblick über die verfügbaren Erweiterungen

Die Erweiterungsoptionen unterscheiden sich im Hinblick auf den erforderlichen Entwicklungsaufwand und die Investition. Die folgende Tabelle ordnet die Methoden von niedrigem zu hohem Aufwand.

Erweiterungsmethode	Beschreibung	Entwicklungsaufwand
Constellation-Portale	Vordefinierte Out-of-the-Box-Portale für Standard- und Self-Service-Anwendungen.	Niedrig
Web Embed	Einbettung von Constellation-UI-Elementen in externe Webseiten mittels Webkomponenten.	Relativ niedrig
DX Components	Erstellung benutzerdefinierter, wiederverwendbarer UI-Komponenten mit React.	Mittel (Frontend-Kenntnisse erforderlich)
SDKs	Erstellung von Pega-Anwendungen mit einem alternativen Designsystem (z. B. Angular, React).	Hoch (Frontend-Entwickler erforderlich)
DX API	Direkter API-Zugriff zur Erstellung einer vollständig benutzerdefinierten UI mit beliebiger Technologie.	Sehr hoch (Expertenkenntnisse erforderlich)

Constellation Web Embed

Mit Constellation Web Embed können Pega-Workflows, Landing Pages und Case-Erfahrungen direkt in eine bestehende Website integriert werden. Dies ermöglicht es, Pega-Funktionalität Nutzern anderer Websites zur Verfügung zu stellen.

- **Technologie:** Im Gegensatz zur Pega Web Mashup, die <iframe>-Komponenten verwendet, basiert Web Embed auf HTML5-Webkomponenten. Dies ermöglicht eine nahtlosere Integration.
- **Anpassung:** Web Embeds unterstützen Themes. Low-Code-Entwickler können in App Studio ein Theme auswählen, um das Erscheinungsbild an das Branding der Host-Seite anzupassen. Zusätzliche Theme-Parameter im Web-Embed-Skript ermöglichen weitere Anpassungen.
- **Erstellung:** Ein neuer Web Embed wird über den Tab **Channel** in App Studio erstellt, wo der erforderliche Markup-Code automatisch generiert wird.

Constellation DX Components

Constellation DX Components erweitern die Standard-UI-Bibliothek von Constellation. Sie werden eingesetzt, wenn spezifische Anforderungen an die Benutzeroberfläche nicht mit den standardmäßigen Case-Management-Mustern und Konfigurationsoptionen erfüllt werden können.

- **Grundprinzip:** Sie ermöglichen es Entwicklern, zweckspezifische UI-Komponenten zu erstellen, die sich nahtlos in die Autorenerfahrung von App Studio integrieren.
- **Technologie:** Die Entwicklung erfolgt mit **React**, einer Open-Source-JavaScript-Bibliothek. Entwickler müssen zudem wissen, wie die Constellation-APIs PCore und PConnect aufgerufen und die Komponenten in die Pega Platform™ integriert werden.
- **Implementierung:** Die kompilierte Version einer Komponente wird auf die Pega Platform hochgeladen und als Rule-UI-Component gespeichert. Anschließend kann sie in App Studio in einer Ansicht konfiguriert werden.
- **Kosten und Abwägung:** Constellation basiert auf dem Paradigma "Konfiguration vor Anpassung". Die Erstellung von DX Components stellt eine Anpassung dar und führt zu längerer Entwicklungszeit, potenziell mehr Fehlern und komplexeren Upgrades. Sie sollten nur dann verwendet werden, wenn die Funktionalität für den Case-Management-Prozess des Kunden unerlässlich ist und nicht durch Out-of-the-Box-Muster erreicht werden kann.
- **Entwicklungswerkzeuge:** Der **Constellation DX Component Builder**, der über npm (Node.js Package Manager) installiert wird, bietet ein CLI-Tool zur Initialisierung, Strukturierung (Scaffolding) und Veröffentlichung der Komponenten.
- **Ressourcen:** Die **Constellation DX Components UI Gallery** bietet eine Sammlung von gebrauchsfertigen und anpassbaren Komponenten, inklusive Quellcode auf GitHub, Designanforderungen und Best Practices.

Arten von DX Components

Es können drei verschiedene Arten von Constellation DX Components erstellt werden:

1. **Field Components:** Dienen zur Anzeige eines Feldes auf eine Weise, die mit Standardkomponenten nicht möglich ist.
    a. *Beispiel:* Die LocationInput-Komponente integriert Google Maps basierend auf der Adresseingabe des Benutzers. Weitere Beispiele sind TextArea, Button, Slider oder Signature Capture zur Erfassung von Unterschriften.
2. **Widget Components:** Verbessern die Benutzerfreundlichkeit durch eingebaute Logik und Funktionalität. Sie sind typischerweise eine einzigartige Kombination mehrerer Standard-UI-Komponenten.
    a. *Beispiel:* Die FileInput-Komponente ermöglicht das Hochladen von Dateien per Drag-and-Drop. Das Task List-Widget dient der Verfolgung des Aufgabenfortschritts.
3. **Layout Template Components:** Bieten neue Layouts für Constellation Views (Details Views, Forms, Pages), die mit den Standard-Layoutvorlagen von App Studio nicht realisierbar sind. Sie helfen, Struktur, Responsivität und Barrierefreiheit zu verbessern.
    a. *Beispiel:* Die CompareTableLayout-Komponente ermöglicht den direkten Vergleich verschiedener Objekte nebeneinander.

Constellation SDKs

Die Constellation SDKs (Software Development Kits) sind für Szenarien gedacht, in denen eine strikte Einhaltung der Marken- und Designvorgaben des Kunden erforderlich ist. Sie ermöglichen es Frontend-Entwicklern, Pega-Anwendungen mit einem anderen Designsystem als dem von Pega Constellation zu erstellen.

- **Kernfunktionalität:** Die SDKs verbinden die **ConstellationJS Engine APIs** von Pega mit einem alternativen Designsystem. Sie nutzen dieselbe Beschleunigungsschicht wie die native Constellation UI, was die Implementierung vereinfacht und Qualität sowie Leistung im Vergleich zu herkömmlichen DX API Starter Packs verbessert.
- **Wichtiger Hinweis:** SDKs sind an eine bestimmte Release-Version gebunden. Die verwendete SDK-Version muss mit der Pega Infinity Server-Version übereinstimmen.

Verfügbare SDKs

Derzeit sind drei Constellation SDKs verfügbar:

- **Pega + Angular:** Bietet Unterstützung für Designsysteme, die auf der Angular-Plattform basieren. Die mitgelieferten Komponenten nutzen **Angular Material**.
- **Pega + React:** Bietet Unterstützung für React-basierte Designsysteme. Die Komponenten in diesem SDK basieren auf dem **Material UI Design System**. Für bestimmte Versionen (v8.23.10 und v23.1.10) enthält es auch einen integrierten DX Component Builder.
- **Pega + Web Components:** Bietet Unterstützung für Systeme, die auf Web Components basieren. Die Container- und Präsentationskomponenten basieren auf **Lion Web Components**, die wiederum mit **LitElement** und **lit-html**erstellt wurden.

The Constellation DX API

Die Constellation Digital Experience (DX) API ist die flexibelste, aber auch aufwendigste Methode zur UI-Erweiterung. Sie besteht aus einer Reihe von modellgetriebenen REST-API-Endpunkten, die sowohl Case-Daten als auch UI-Metadaten aus der View-Regel zurückgeben.

- **Anwendungsbereich:** Die API ermöglicht es, Cases und Zuweisungen programmatisch anzuzeigen, zu erstellen und zu aktualisieren. Sie bildet die Grundlage für die Out-of-the-Box-Constellation-UI, kann aber mit jeder beliebigen Frontend-Technologie (z. B. React, Angular, Vue, Vanilla JS) verwendet werden, um ein vollständig benutzerdefiniertes Erlebnis zu schaffen.
- **Unterschied zur Traditional DX API:** Die Constellation DX API ist für View-basierte Anwendungen in App Studio optimiert. Wichtige Unterschiede sind:
    - Fokus auf View-basiertes Authoring statt auf Harnesses und Sections.
    - Unterstützung des Pega Center-out-Ansatzes.
    - Eine JSON-Struktur, die Daten klar von der Layout-Definition trennt.
    - Reduzierte Anzahl von API-Aufrufen.
    - Neue Endpunkte für zusätzliche Funktionalitäten (z. B. einem Case folgen, Case-Stages ändern).
    - Verwendung von HATEOAS, um verfügbare Aktionen in den Antworten zu referenzieren.
    - Am besten geeignet für neue Pega-Anwendungen.
- **Authentifizierung:** Die Constellation DX API verwendet zur Absicherung der Endpunkte ausschließlich **OAuth 2.0**.

Aufbau einer DX API-Anfrage

Eine API-Anfrage besteht aus mehreren Komponenten:

- **Path Parameters:** Bestimmen den aufgerufenen Endpunkt und damit die zurückgegebene Antwort (z.B. GET /assignments/{id}/actions/{actionID}).
- **Headers:**
    - x-origin-channel: Gibt den Ursprung der Anfrage an (z. B. web oder mobile), um regelbasierte Anpassungen der UI zu ermöglichen.
    - if-match und eTag: Werden für Optimistic Locking verwendet, um das versehentliche Überschreiben von Daten durch mehrere Benutzer zu verhindern. Der eTag-Header in der Antwort (basierend auf pxSaveDateTime) muss im if-match-Header der nachfolgenden Update-Anfrage übergeben werden. Bei einer Nichtübereinstimmung wird ein HTTP 409 Conflict-Fehler zurückgegeben.
- **Query Parameters:** Zusätzliche Informationen in der URL.
    - viewType: Steuert, wie viele UI-Metadaten zurückgegeben werden (Page, Form, Fields oder None).
    - pageName: Gibt den Namen einer View-Seite an, auf die zugegriffen werden soll (Standard ist pyDetails).
- **Request Body:** Erforderlich für POST-, PUT- oder PATCH-Methoden. Enthält Daten wie caseTypeID, content(Eigenschaftswerte), pageInstructions und attachments.

Verstehen der DX API-Antwort

Die JSON-Antwort der API enthält typischerweise vier wichtige Top-Level-Eigenschaften:

- **data:** Enthält Case-Details wie Status, verfügbare Aktionen, SLA-Informationen und Referenzen auf Benutzer.
- **uiResources:** Enthält die UI-Metadaten, die zum Rendern der Seite oder des Formulars benötigt werden.
- **nextAssignmentInfo:** Erscheint nur, wenn eine nachfolgende Zuweisung konfiguriert ist, und enthält Informationen darüber.
- **confirmationNote:** Eine für den Endbenutzer sichtbare Notiz, die nur in der letzten Stufe eines Cases erscheint.