Eine ID generieren

Über die Verwendung von IDs

Ein häufiger Anwendungsfall für den Einsatz eines Workers ist das Generieren einer ID, mit der eine Instanz eindeutig identifiziert werden kann, zum Beispiel eine Bestell-, Ticket- oder Rechnungsnummer. Oft wird ein Zähler gewünscht, der eine fortlaufende Nummer generiert. Besonderes Merkmal einer ID ist aber ihre **Eindeutigkeit**: In der Datenbank soll eine ID niemals mehrfach vorhanden sein.

Die Verwendung von Countern, die fortlaufende Zahlen generieren, ist daher in verteilten System nicht optimal. Denn der Einsatz eines einfachen Zählers kann zu Datenkollisionen führen, beispielsweise wenn mehrere Anwender zeitgleich ein Formular öffnen, um eine neue Instanz anzulegen und eine Zahl dadurch doppelt vergeben wird. Auch wenn Instanzen aus der Datenbank gelöscht werden, ist die Anzeige der Zahlen nicht mehr fortlaufend, da die IDs der gelöschten Instanzen fehlen.

Bevor Sie sich für die Verwendung einer fortlaufenden ID entscheiden, wägen Sie die Vor- und Nachteile ab.

Benötigen Sie für Ihre Anwendung eine ID, bieten sich zwei Alternativen an:

- Nutzung der Metainformation **meta.createdAt**
- Generierung einer UUID

Einsatz von meta.createdAt als ID

Jede Instanz führt die Metainformation **meta.createdAt** automatisch mit sich, denn **meta.createdAt** enthält den Zeitstempel des Zeitpunkts, an dem die zugehörige Instanz angelegt wurde. Daher können Sie sowohl in der Suchübersicht als auch in Formularfeldern direkt auf **meta.createdAt** zugreifen (Eingabe als **Feldname im Container**).

Das Datum wird als Anzahl Millisekunden (long) seit dem 01.01.1970 gespeichert. Ein weiter zurückliegendes Datum hat daher intern ein **negatives Vorzeichen**.

Das Verwenden von **meta.createdAt** als eindeutige ID hat mehrere Vorteile:

- Jede Instanz führt diese Information automatisch mit sich, Sie müssen keine eigene ID generieren.
- Die Information ist bis auf die Millisekunde genau - es ist daher höchst unwahrscheinlich, dass zwei Instanzen mit genau dem gleichen Millisekundenwert gespeichert werden.
- Einfache Anzeige der Reihenfolge von Instanzen: Sie können die Suchübersicht aufsteigend oder absteigend nach **meta.createdAt** sortieren.

Eine UUID generieren

Eine weitere Möglichkeit ist das Generieren einer sogenannten UUID. IDs sollen ein Objekt eindeutig identifizieren, das heißt: Eine ID soll nur einmalig vergeben werden. UUID ist die Abkürzung für engl. *Univ ersally Unique Identifier*. Die UUID ist eine Identifikationsnummer, die aus einer 16-Byte-Zahl besteht, die hexadezimal notiert wird und in fünf Gruppen unterteilt ist.

Beispiel für eine UUID: 5e849a55-b74a-4842-b7a2-b6c4c44b5709

Die Eindeutigkeit einer zufällig generierten UUID ist zwar nicht garantiert, doch ist die Gesamtzahl zufällig generierter UUIDs so groß, dass die Wahrscheinlichkeit zweier gleicher UUIDs sehr klein ist.

Skript zur Erzeugung einer zufälligen UUID

```
var myId = uuid.v4();
container.put('id', myId);
```

Beispiel: UUID über **Worker: ID generieren** erzeugen und in ein Formularfeld ausgeben

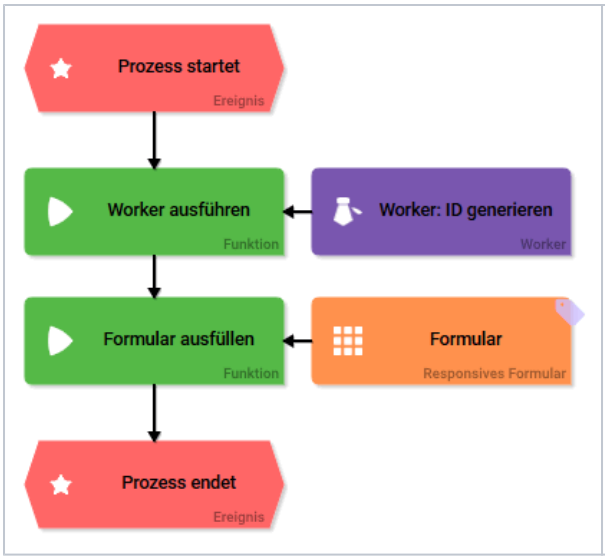
On this Page:

Related Pages:

- Code-Bibliothek

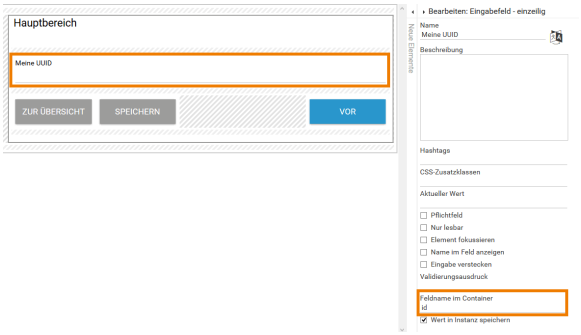
Related Documentation:

- BPaaS

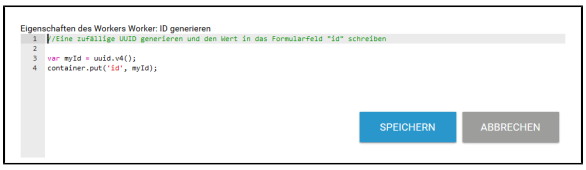


Benötigt wird eine kleine EPK mit zwei Schritten: In Schritt 1 Worker ausführen wird die UUID generiert. In Schritt 2 Formular ausfüllen wird die ID im entsprechenden Formularfeld angezeigt.

Dem Formularfeld Meine UUID wurde der **Feldname im Container** id zugewiesen:



Das Skript wird im Editor des Workers gespeichert:



Bei der Ausführung der App wird die generierte ID im Feld Meine UUID angezeigt: