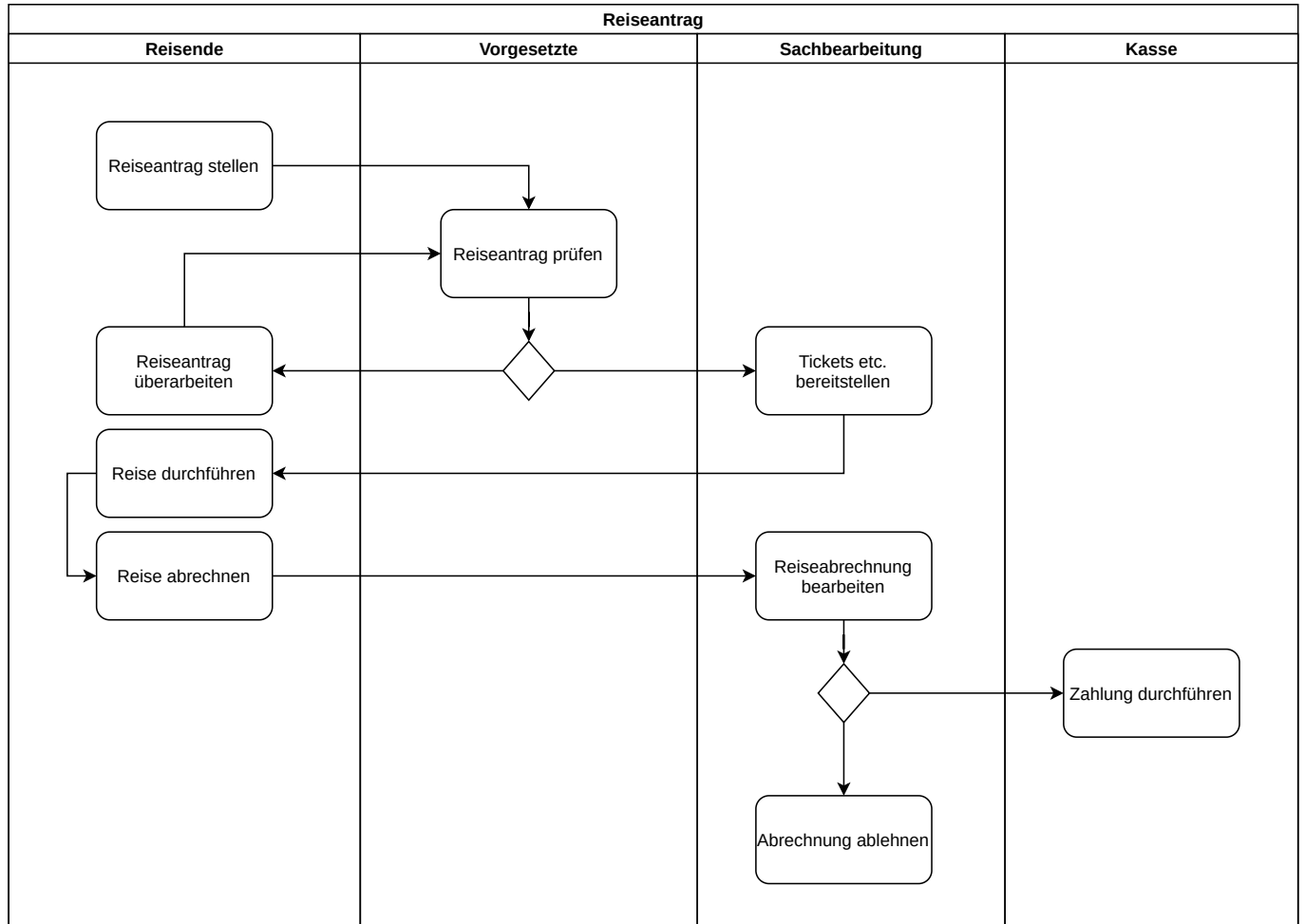


FlexDMA Schnelleinstieg

Das nachfolgende Tutorial bietet einen Schnelleinstieg in die wichtigsten Features der Workflow-Engine FlexDMA am Beispiel eines Reiseantragsworkflows.



Schrittweises Vorgehen in der Entwicklungsumgebung

Neues Prozessmodell anlegen

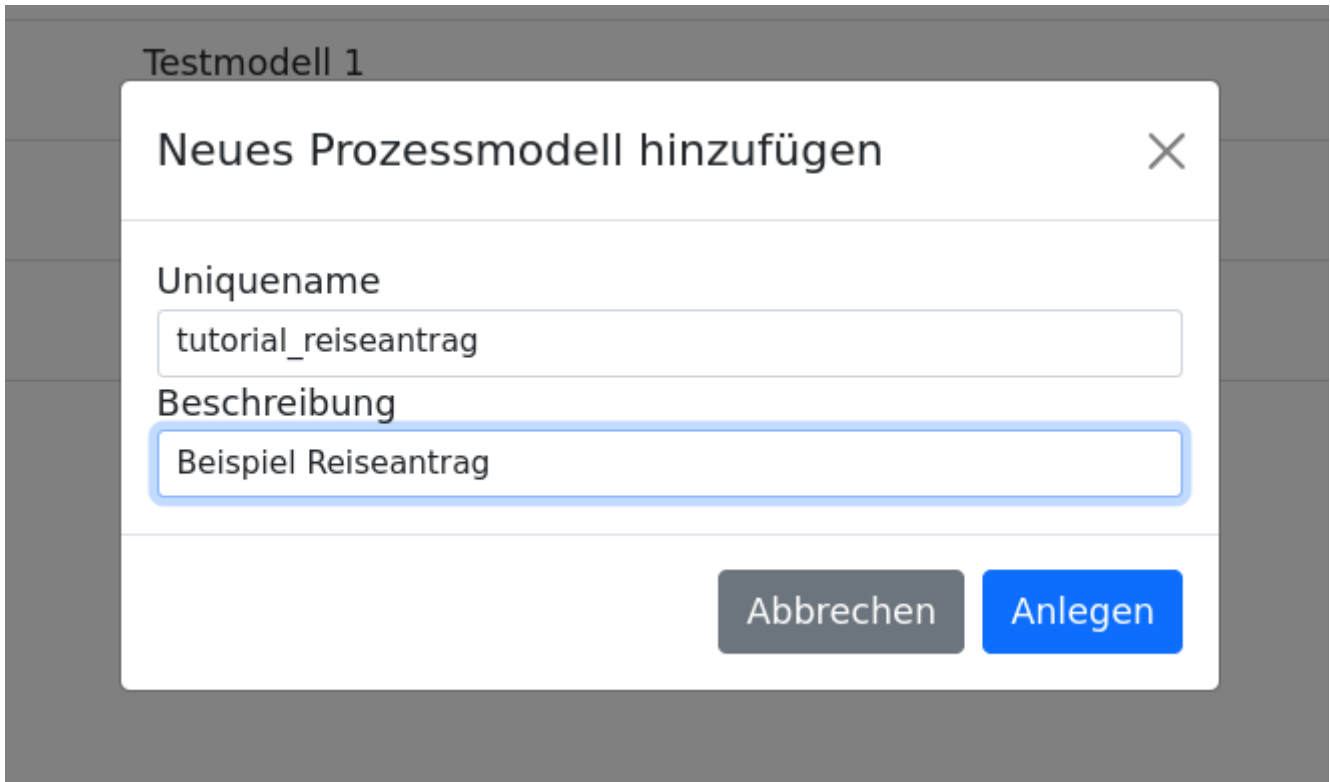
FlexDMA Entwicklungsumgebung

Bestehendes Prozessmodell auswählen

Eindeutiger Name	Beschreibung	
test_issue_61	Test zu Issue 61 in Gitlab	Öffnen
oth_abschlussarbeiten	Anmeldung von Abschlussarbeiten	Öffnen
test_1	Testmodell 1	Öffnen
oth_beschaffung	Beschaffungsantrag OTH	Öffnen
test_issue_70	Test zu Issue 70 in Gitlab - type-Duplikate	Öffnen

[Neues Prozessmodell anlegen](#)

Es beginnt mit einen Klick auf den Button **Neues Prozessmodell anlegen**.



Testmodell 1

Neues Prozessmodell hinzufügen

Uniquename

tutorial_reiseantrag

Beschreibung

Beispiel Reiseantrag

Abbrechen Anlegen

Danach erhalten Sie ein neues leeres Prozessmodell, in dessen Schema bereits ein minimales Code-Beispiel mit einem einzelnen Attribut (sample) bereitgestellt wird.

FlexDMA Entwicklungsumgebung

Prozessmodell bearbeiten: tutorial_reiseantrag

Beschreibung

Beispiel Reiseantrag

Schema Code

```
1 schema_uniquename: tutorial_reiseantrag
2 schema_title: Beispiel Reiseantrag
3 initial_status: initial
4 attributes:
5 - attribute_name: sample
6   label: Beispiel
7   type: string
8   control: textbox
```

Schema Speichern

View hinzufügen

Aktuelle Modellversion für Produktivbetrieb freigeben

Prozessdiagramm anzeigen

Danach können Sie im Reiter Schema das Datenmodell ihres Prozessmodells definieren. Es soll am Ende alle Felder enthalten, die in den Prozessschritten benötigt werden.

Vollständiges Schema des Beispiel-Reiseantrags

```
schema_uniquename: tutorial_reiseantrag
schema_title: Beispiel Reiseantrag
initial_status: initial
types:
  hotel:
    attributes:
      - attribute_name: name
        label: Name
        type: string
        control: textbox
      - attribute_name: kosten
        label: "Kosten"
        type: string
        regex: "^[0-9]+(,[0-9]+)?$"
        control: textbox
  reisetyp:
    attributes:
      - attribute_name: typ
        label: Reisetyp
        type: string
        control: textbox
    loader:
      type: python
      code_location: "tutorial_reiseantrag/loaders/reisetyp.py"
attributes:
  - attribute_name: reisetyp
    label: "Typ der Reise"
    type: reisetyp
    control: select
    required: true
  - attribute_name: vorname
    label: Vorname
    type: string
    control: textbox
    required: true
  - attribute_name: nachname
    label: Nachname
    type: string
    control: textbox
    required: true
  - attribute_name: zielort
    label: Zielort
    type: string
    control: textbox
  - attribute_name: beginn
    label: Reisebeginn
    type: date
    control: datepicker
  - attribute_name: ende
    label: Reiseende
    type: date
    control: datepicker
```

- ```
- attribute_name: uebernachtungen
 label: "Übernachtungen"
 type: hotel
 cardinality: many
 control: tableedit
- attribute_name: genehmigungskommentar
 label: "Kommentar der/des Vorgesetzten"
 type: string
 control: textarea
```

Für das Attribut *reisetyp* vom Typ *reisetyp* ist im Schema ein dynamisches Laden mittels eigenem Code in Python oder SQL vorgesehen worden. Aus diesem Grund muss als nächstes die dafür vorgesehene Funktion im Reiter **Code** definiert werden. Wie das geht zeigt der nächste Screenshot.

## FlexDMA Entwicklungsumgebung

Prozessmodell bearbeiten: tutorial\_reiseantrag

Beschreibung

Beispiel Reiseantrag

Schema Code

tutorial\_reiseantrag/loaders/reisetyp.py

Programmiersprache: python

Anwendungskontext : load

```
1 result = []
2 { "typ": "Dienstreise" },
3 { "typ": "Fortbildungsreise" }
4]
```

Damit sind die grundlegenden Arbeiten an unserem Prozessmodell abgeschlossen und wir können uns nun um die Abbildung der einzelnen Prozessschritte kümmern.

## Abbildung der Prozessschritte als Views

Durch einen Klick auf den Button **View hinzufügen** in der folgenden Maske kann eine neue View angelegt werden.

## FlexDMA Entwicklungsumgebung

Prozessmodell bearbeiten: tutorial\_reiseantrag

Beschreibung

Beispiel Reiseantrag

Schema Code

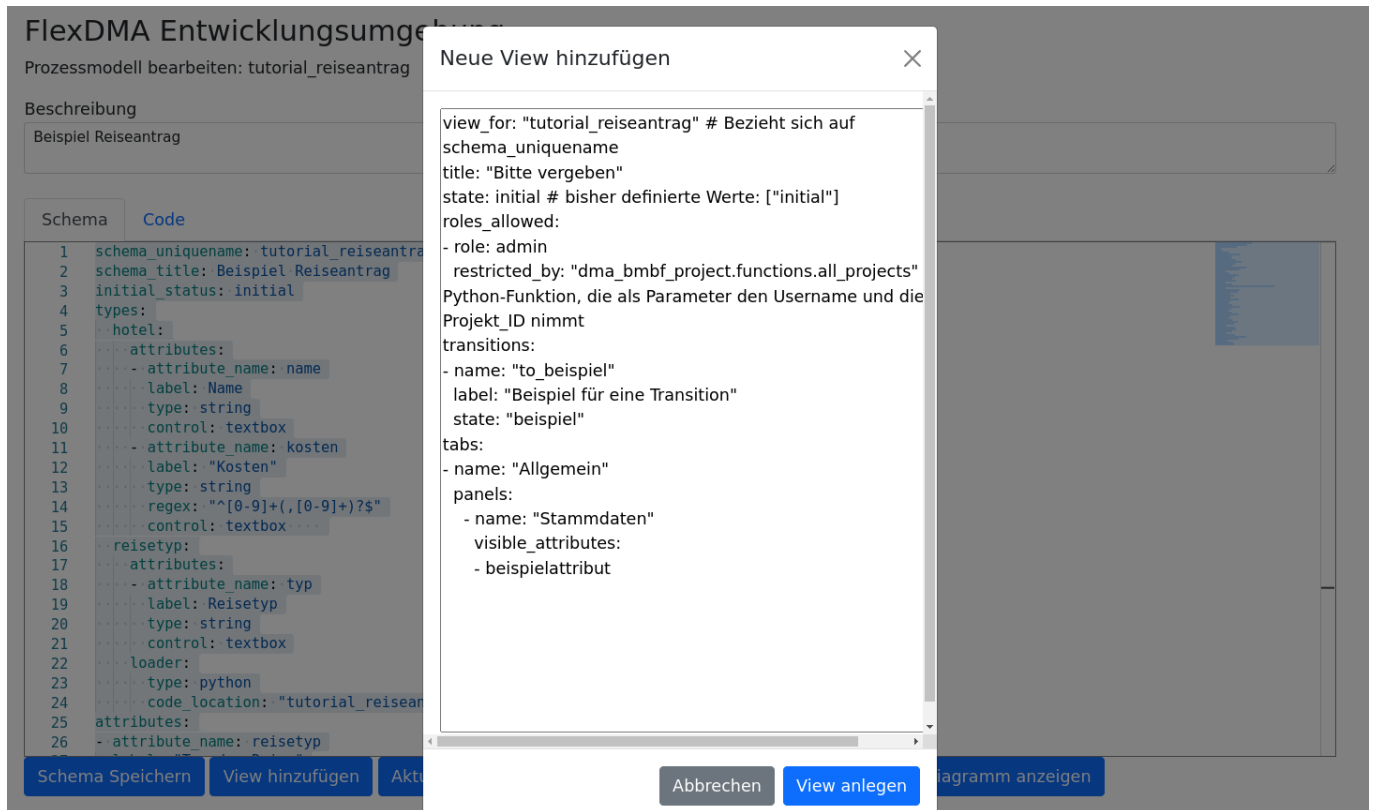
```

1 schema_uniquename: tutorial_reiseantrag
2 schema_title: Beispiel Reiseantrag
3 initial_status: initial
4 types:
5 hotel:
6 attributes:
7 attribute_name: name
8 label: Name
9 type: string
10 control: textbox
11 attribute_name: kosten
12 label: "Kosten"
13 type: string
14 regex: "^[0-9]+(,[0-9]+)?$"
15 control: textbox
16 reisetyp:
17 attributes:
18 attribute_name: typ
19 label: Reisetyp
20 type: string
21 control: textbox
22 loader:
23 type: python
24 code_location: "tutorial_reiseantrag/loaders/reisetyp.py"
25 attributes:
26 attribute_name: reisetyp

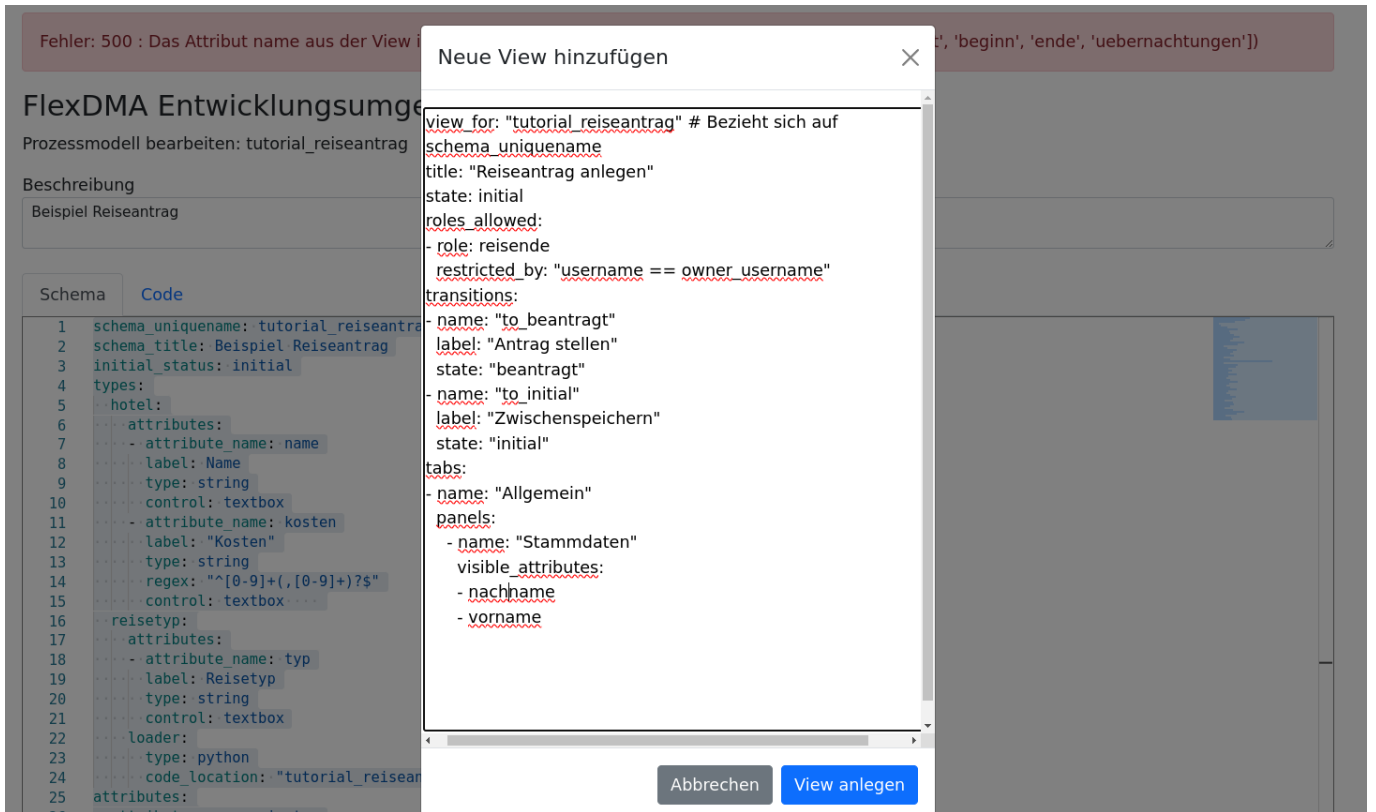
```

- Schema Speichern
- View hinzufügen
- Aktuelle Modellversion für Produktivbetrieb freigeben
- Prozessdiagramm anzeigen

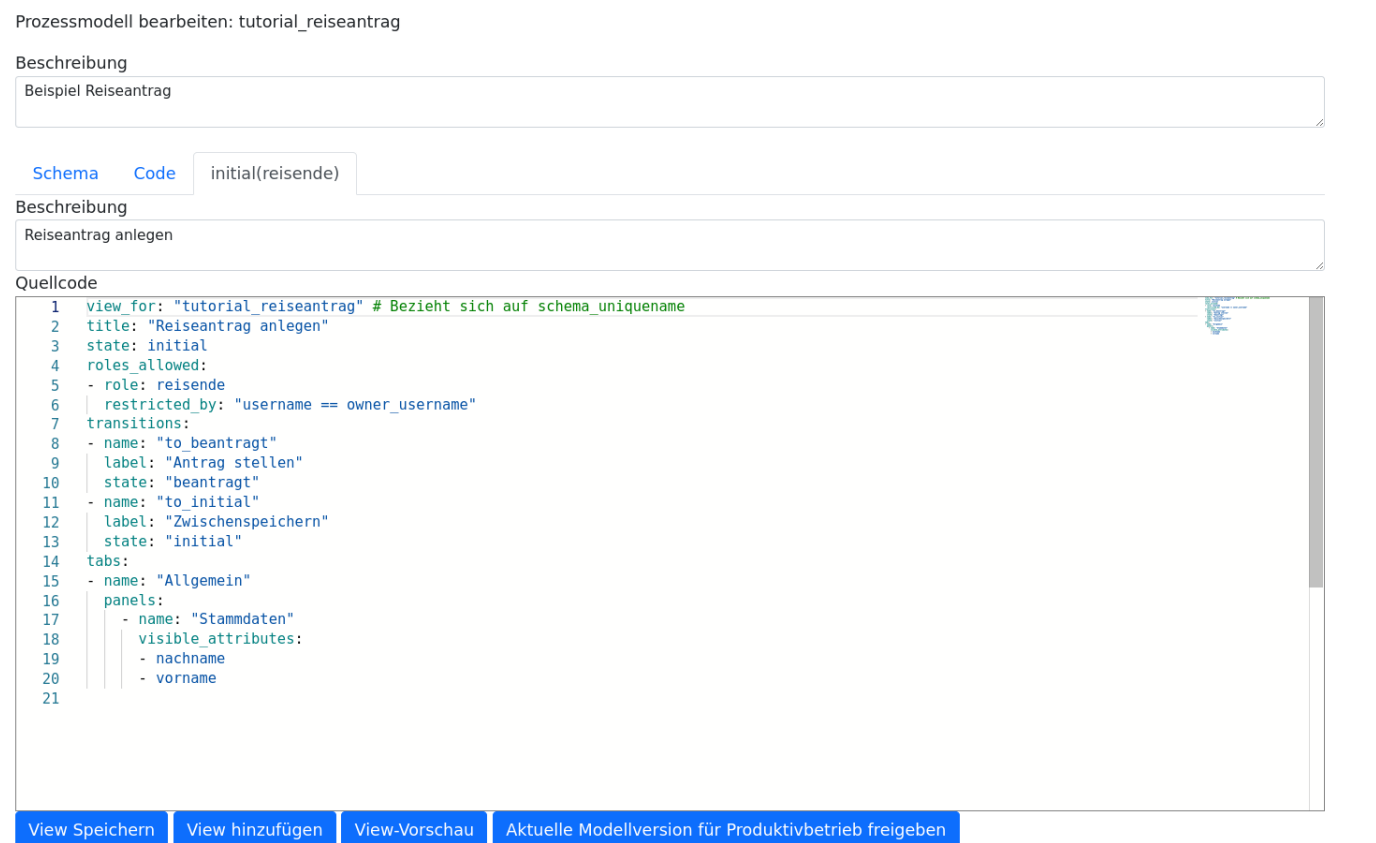
Nach dem Klick erscheint ein Dialog, der bereits einen Code-Vorlage für die Erstellung der View mit empfehlungen für die verpflichtend zu definierenden und einige weitere Attribute beinhaltet.



Dieser Code kann nun vor dem tatsächlichen Anlegen der View noch wie im folgenden Screenshot angepasst werden.



Nach dem Klick auf **View anlegen** erscheint die neue View dann in einem neuen Tab im Quellcodeeditor.



Dort lässt sie sich über den Button **View-Vorschau** auch schon so betrachten, wie sie in der finalen Anwendung dann aussehen wird.

## Formular: Reiseantrag anlegen

Allgemein

Stammdaten

Nachname

Vorname

Antrag stellen

Zwischenspeichern

Zurück

Nun kann die View des Antragstellers so gestaltet werden, wie sie final aussehen soll.

Prozessmodell bearbeiten: tutorial\_reiseantrag

Beschreibung

Beispiel Reiseantrag

Schema

Code

initial(reisende)

Beschreibung

Reiseantrag anlegen

Quellcode

```

1 view_for: "tutorial_reiseantrag" # Bezieht sich auf schema_uniquename
2 title: "Reiseantrag anlegen"
3 state: initial
4 roles_allowed:
5 - role: reisende
6 | restricted_by: "username == owner_username"
7 transitions:
8 - name: "to beantragt"
9 label: "Antrag stellen"
10 state: "beantragt"
11 - name: "to initial"
12 label: "Zwischenspeichern"
13 state: "initial"
14 tabs:
15 - name: "Allgemein"
16 panels:
17 - name: "Stammdaten"
18 visible_attributes:
19 - nachname
20 - vorname
21 - name: "Reisedaten"
22 visible_attributes:
23 - reisetyp
24 - zielort
25 - beginn
26 - ende

```

View Speichern

View hinzufügen

View-Vorschau

Aktuelle Modellversion für Produktivbetrieb freigeben

Das hat dann in der Vorschau das folgende Resultat:

## Formular: Reiseantrag anlegen

Allgemein

**Stammdaten**

Nachname \*

Vorname \*

**Reisedaten**

Typ der Reise \*  
Bitte auswählen...

Zielort

Reisebeginn  
01.01.1950

Reiseende  
01.01.1950

Antrag stellen    Zwischenspeichern    Zurück

Hier noch der vollständige Code unserer ersten View:

```
view_for: "tutorial_reiseantrag" # Bezieht sich auf schema_uniquename
title: "Reiseantrag anlegen"
state: initial
roles_allowed:
- role: reisende
 restricted_by: "username == owner_username"
transitions:
- name: "to_beantragt"
 label: "Antrag stellen"
 state: "beantragt"
- name: "to_initial"
 label: "Zwischenspeichern"
 state: "initial"
tabs:
- name: "Allgemein"
 panels:
 - name: "Stammdaten"
 visible_attributes:
 - nachname
 - vorname
 - name: "Reisedaten"
 visible_attributes:
 - reisetyp
 - zielort
 - beginn
 - ende
```



Damit ist die erste View unseres Prozessmodells erfolgreich erzeugt. Im Code-Abschnitt **transitions** enthält unsere erste View übrigens auch schon den Übergang zur nächsten View **Reiseantrag prüfen**, die für Nutzende mit der Rolle *vorgesetzte* bereitsteht und alle Anträge im Status *beantragt* zur Bearbeitung bereitstellt.

Über die Einschränkung (**restricted\_by**) wird festgelegt, dass Reisende in dieser View nur ihre eigenen Reiseanträge sehen dürfen.

## Eine View für Reiseantrag prüfen anlegen

Das Vorgehen zur Anlage der View *Reiseantrag prüfen* ist exakt das gleiche wie bei der View *Reiseantrag stellen*.

Der resultierende Code sieht wie folgt aus:

```
view_for: "tutorial_reiseantrag" # Bezieht sich auf schema_uniquename
title: "Reiseantrag prüfen"
state: beantragt
roles_allowed:
- role: vorgesezte
transitions:
- name: "to_initial"
 label: "Antrag ablehnen"
 state: "initial"
 post_transition:
 code_location: "tutorial_reiseantrag/transitions/send_email.py"
- name: "to_genehmigt"
 label: "Antrag genehmigen"
 state: "genehmigt"
 post_transition:
 code_location: "tutorial_reiseantrag/transitions/send_email.py"
tabs:
- name: "Allgemein"
 panels:
 - name: "Stammdaten"
 visible_attributes:
 - nachname
 - vorname
 - name: "Reisedaten"
 visible_attributes:
 - reisetyp
 - zielort
 - beginn
 - ende
 - name: "Kommentare"
 visible_attributes:
 - genehmigungskommentar
readonly_fields:
- vorname
- nachname
- zielort
```

- **beginn**
- **ende**

Die Maske enthält nun einige Felder, die vom Vorgesetzten nicht bearbeitet werden dürfen, eine E-Mail-Benachrichtigung an den Antragsteller bei Genehmigung und Ablehnung sowie vollständige Sichtrechte aller Nutzer mit der Rolle **vorgesetzte** auf alle Anträge im Status **beantragt**.

Die Vorschau der Maske:

#### Formular: Reiseantrag prüfen

Allgemein

**Stammdaten**

Nachname \*

Vorname \*

**Reisedaten**

Typ der Reise \*  
Bitte auswählen...

Zielort

Reisebeginn  
01.01.1950

Reiseende  
01.01.1950

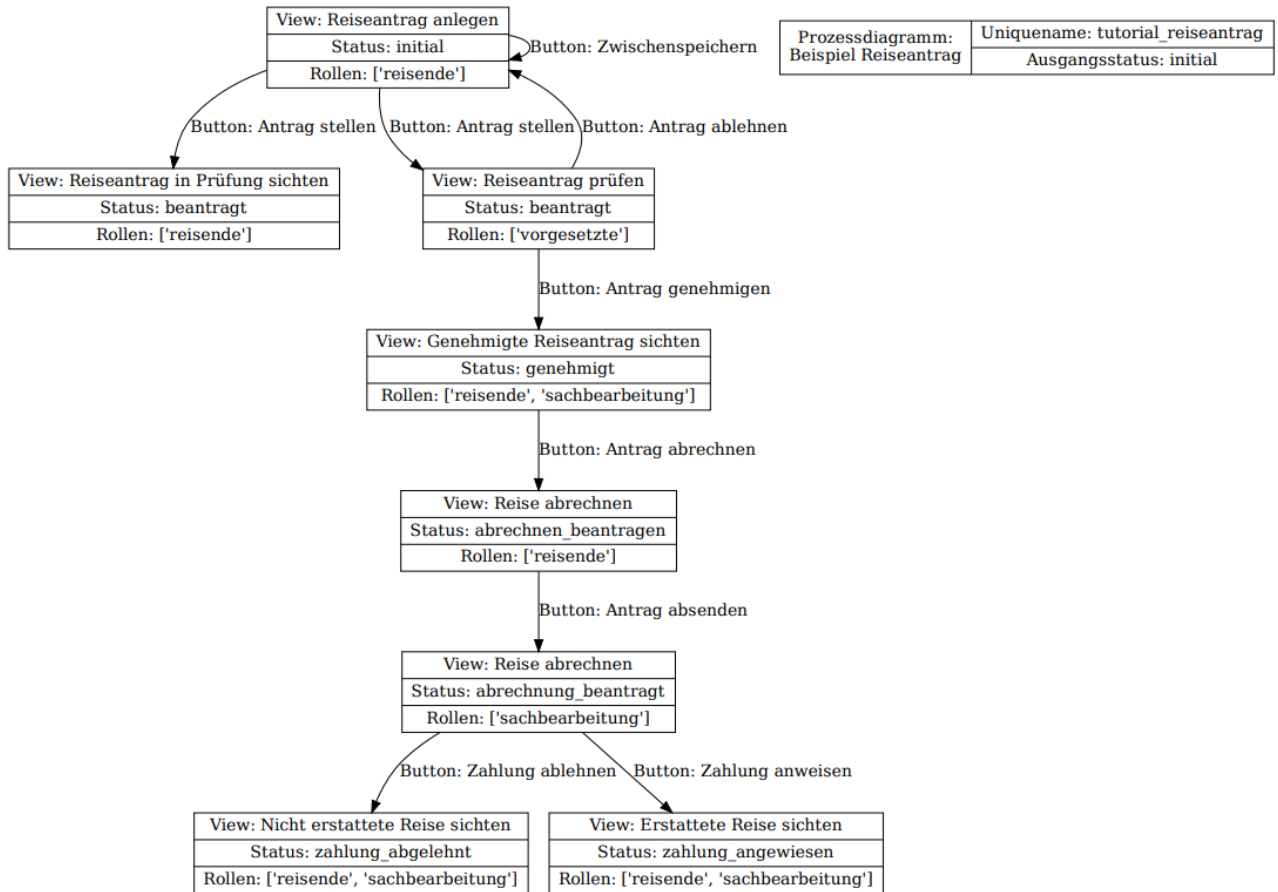
**Kommentare**

Kommentar der/des Vorgesetzten

Antrag ablehnen   Antrag genehmigen   Zurück

Auf die gleiche Weise werden nun alle weiteren laut Prozessdiagramm erforderlichen Views erzeugt.

Das von FlexDMA automatisch generierte Diagramm zum abgebildeten Prozess sieht wie folgt aus:



Ein Durchlauf durch den daraus resultierenden Prozess wird im folgenden Video gezeigt.

Den vollständigen Code der Views finden Sie unter [src/views](#) in diesem Repository.

Dieses Tutorial zeigt nur einen Ausschnitt aus den verfügbaren Funktionen der aktuellen Version von FlexDMA.