

Dokumentenvorlage am Beispiel Angebot

Einleitung

Manchmal ist es sinnvoll von einer handvoll Dokumente eine Vorlage auswählen zu können, damit dem Anwendern die kontinuierliche Eingabe von immer den gleichen Datenpunkten erspart bleiben kann.

In diesem Beispiel ist eine Vorlage für den Dokumententyp Angebot (engl. Quotation) aufgeführt.

Planung

Wir legen einen neuen Dokumententyp namens "Angebotsvorlage" an. Dieser enthält genau die gleichen Felder, die das Angebot enthält minus die Felder, die im Anwendungsbeispiel nicht gebraucht werden. Minimal brauchen wir daher alle Pflichtfelder aus dem Dokumententyp "Angebot".

Nicht möglich ist es manchmal die Tabellen 1:1 mit zu nehmen. Im Angebot werden innerhalb der Tabelle 'items' die "Quotation Item" in mehrfachen Positionen gelistet. In unserem erstellen Dokumententyp "Angebotsvorlage" kann das "Quotation Item" zwar mit einer Tabelle eingebaut werden, ist aber nicht brauchbar. Grund sind ERPNext Funktionen, die im Hintergrund ausgeführt wenn wir eine neue Zeile von Quotation Item in einer Quotation anlegen.

Aus diesem Grund bauen wir einen zweiten Dokumententyp "Angebotsvorlage Item". Dieser enthält genau die gleichen Felder, die das "Quotation Item" enthält minus die Felder, die im jeweiligen Anwendungsbeispiel nicht gebraucht werden.

Sobald ein Dokument vom Typ "Angebotsvorlage" erstellt werden kann ist dieses zu speichern. Wir stellen im nächsten Schritt ein neues "Angebot" indem wir die Vorlage öffnen und folgend der Anleitung am Ende der Seite.

Skript

Das Skript wird als Client Skript eingebunden und beruht auf dem Doctype "Quotation".

The screenshot shows the Frappe Client Script editor interface. At the top, there's a navigation bar with 'Client Script' and 'Quotation-Form'. Below this, the editor is titled 'Quotation-Form' and 'Disabled'. On the left sidebar, there are sections for 'Assigned To', 'Attachments', 'Shared With', and 'Tags'. The main area is divided into 'DocType' (set to 'Quotation'), 'Apply To' (set to 'Form'), and 'Script'. The script is a JavaScript function that triggers a validation and refresh process. Below the script, there's a 'Client Script Help' section with a brief explanation of client-side scripts.

```
1 * frappe.ui.form.on('Quotation', {
2 *   validate(frm) {
3 *     console.log("Trigger validate");
4 *     //frappe.model.set_value("Quotation Item", ctn, value);
5 *     console.log(frm.doc.items);
6 *     for (let i = 0; i < frm.doc.items.length; i++) {
7 *       var item = frm.doc.items[i];
8 *       if(item.positionsart != "Standard"){
9 *         frappe.model.set_value(item.doctype, item.name, "rate", "0");
10 *        frappe.refresh_fields("rate");
11 *      }
12 *    }
13 *  },
14 *   refresh(frm) {
15 *     frm.add_custom_button('Angebotsvorlage', function () { frm.trigger('get_items') }, __('Get Items From'));
16 *   },
17 *   get_items(frm){
18 *     start_dialog(frm);
19 *   }
20 * });
21
22 * function start_dialog(frm) {
23 *   let dialog = new frappe.ui.form.MultiFieldDialog({
24 *     fields: [
25 *       {
26 *         fieldname: 'item_code',
27 *         label: 'Item Code',
28 *         reqd: 1,
29 *         type: 'Text'
30 *       },
31 *       {
32 *         fieldname: 'item_name',
33 *         label: 'Item Name',
34 *         reqd: 1,
35 *         type: 'Text'
36 *       },
37 *       {
38 *         fieldname: 'positionsart',
39 *         label: 'Positionsart',
40 *         reqd: 1,
41 *         type: 'Text'
42 *       },
43 *       {
44 *         fieldname: 'description',
45 *         label: 'Description',
46 *         reqd: 1,
47 *         type: 'Text'
48 *       },
49 *       {
50 *         fieldname: 'qty',
51 *         label: 'Qty',
52 *         reqd: 1,
53 *         type: 'Text'
54 *       },
55 *       {
56 *         fieldname: 'uom',
57 *         label: 'Uom',
58 *         reqd: 1,
59 *         type: 'Text'
60 *       },
61 *       {
62 *         fieldname: 'rate',
63 *         label: 'Rate',
64 *         reqd: 1,
65 *         type: 'Text'
66 *       }
67 *     ],
68 *     primary_action_label: 'Save',
69 *     primary_action: function() {
70 *       frappe.call({
71 *         method: 'frappe.client.insert',
72 *         args: {
73 *           doctype: 'Quotation Item',
74 *           values: {
75 *             item_code: item_code,
76 *             item_name: item_name,
77 *             positionsart: positionsart,
78 *             description: description,
79 *             qty: qty,
80 *             uom: uom,
81 *             rate: rate
82 *           }
83 *         }
84 *       });
85 *     }
86 *   });
87 * }
```

Client Script Help

Client Scripts are executed only on the client-side (i.e. in Forms). Here are some examples to get you started

```
// The fetch-from fields
var fields = [
  "item_code",
  "item_name",
  "positionsart",
  "description",
  "qty",
  "uom",
  "rate"];

frappe.ui.form.on('Quotation', {
```

```

refresh(frm) {
  frm.add_custom_button('Angebotsvorlage', function () { frm.trigger('get_items') }, __('Get Items From'));
},
get_items(frm){
  start_dialog(frm);
}
});

function start_dialog(frm) {
  let dialog = new frappe.ui.form.MultiSelectDialog({

    // Read carefully and adjust parameters
    doctype: "Angebotsvorlage", // Doctype we want to pick up
    target: frm,
    setters: {
      // MultiDialog Filterfields
      // customer: frm.doc.customer,
    },
    date_field: "creation", // "modified", "creation", ...
    get_query() {
      // MultiDialog Listfilter
      return {
        filters: { }
      };
    },
    action(selections) {
      for(var n = 0; n < selections.length; n++){
        var name = selections[n];
        frappe.db.get_doc("Angebotsvorlage", name) // Again, the Doctype we want to pick up
          .then(doc => {
            // Remove the first empty element of the table
            if(!('item_code' in frm.get_field("items").grid.grid_rows[0].doc)){
              frm.get_field("items").grid.grid_rows[0].remove();
            }

            // Run through all items of the template quotation
            for(var n = 0; n < doc.angebotsvorlage_item.length; n++){
              // Declare variables and add table row
              var item=doc.angebotsvorlage_item[n];

```


> Selling > Quotation > new-quotation-2
 Search or type a command (Ctrl + G)
Help

New Quotation Not Saved
Get Items From
...
Save

Series *
 SAL-QTN-.YYYY.-

Quotation To *
 Customer

Customer *

Company *
 Beispiel Unternehmen & Technology Consulting GmbH

Datum *
 2024.01.01

Valid Till
 2024.12.31

Order Type *
 Sales

Currency and Price List

Angebotsvorlage 1
 Opportunity

Im folgenden MultiSelect-Dialog wählen wir die gewünschte Angebotsvorlage aus und selektieren diese in der Mehrfachauswahl (1) und bestätigen mit (2).

> Selling > Quotation > new-quotation-2
 Search or type a command (Ctrl + G)
Help

New Quotation Not Saved
Get Items From
...
Save

Select Angebotsvorlages

Name

☐ Name
☒ Die Angebotsvorlage 1

Make Angebotsvorlage
 Get Items 2

Items

No.	Item
1	

 Add Multiple

Total Quantity
 0

Quantity
 0

Rate (EUR)
 € 0,00

Edit
 Download Upload

Danach erscheinen die Positionen aus der Angebotsvorlage im Angebot.

Bekannter Bug: Der Dialog funktioniert nur einmalig. Bei mehrfachem 'Get Items From'-'>'Angebotsvorlage' werden Leerzeilen in der Positionstabelle eingefügt.

Weitere Ausbaustufen

Anstelle des Client Script, der von uns entwickelt wurde, kann eventuell die ERPNext eigene Funktion in Zeile 95 in folgendem Code-Blob verwendet werden.

<https://github.com/frappe/erpnext/blob/develop/erpnext/selling/doctype/quotation/quotation.js>

Die Funktion an Position (1) ist eine oft verwendete Ressource und kann wiederverwendet werden. Die Aufgerufene Methode (2) ist spezifisch um ein Angebot zu erstellen. Das ganze ist als Client Skript anzulegen.

```
if (this.frm.doc.docstatus===0) {
  this.frm.add_custom_button(__('Opportunity'),
    function() {
      1 erpnext.utils.map_current_doc({
        method: "erpnext.crm.doctype.opportunity.opportunity.make_quotation", 2
        source_doctype: "Opportunity",
        target: me.frm,
        setters: [
          {
            label: "Party",
            fieldname: "party_name",
            fieldtype: "Link",
            options: me.frm.doc.quotation_to,
            default: me.frm.doc.party_name || undefined
          },
          {
            label: "Opportunity Type",
            fieldname: "opportunity_type",
            fieldtype: "Link",
            options: "Opportunity Type",
            default: me.frm.doc.order_type || undefined
          }
        ],
        get_query_filters: {
          status: ["not in", ["Lost", "Closed"]],
          company: me.frm.doc.company
        }
      })
    }, __("Get Items From"), "btn-default");
}
```

Die Setters können wahrscheinlich im ersten Schritt leer gelassen werden, genauso wie die Filter.

```
this.frm.add_custom_button(__('Opportunity'),
  function() {
```

```

erpnext.utils.map_current_doc({
    method: "erpnext.crm.doctype.opportunity.opportunity.make_quotation",
    source_doctype: "Opportunity",
    target: me.frm,
    setters: [
    ],
    get_query_filters: {
    }
})
}, __("Get Items From"), "btn-default");

```

Die in der Funktion aufgerufene whitelist-Methode make_quotation beinhaltet den Backend-Skript im Codeblob

<https://github.com/frappe/erpnext/blob/6954dd6329f2af0db88413fe933c5e1a111bcb9b/erpnext/crm/doctype/opportunity/opportunity.py>

Dieser Blob kann wahrscheinlich gecopied, gepasted werden. Codeblob an Position (1) kann ignoriert oder gelöscht werden. Codeblob an Position (2) auch.

Die Zeilen (3) und (4) müssen angepasst werden. Das ganze ist als gewhitelistetes Server Skript anzulegen.


```

248 @frappe.whitelist()
249 def make_quotation(source_name, target_doc=None):
250     def set_missing_values(source, target):
251         from erpnext.controllers.accounts_controller import get_default_taxes_and_charges
252         quotation = frappe.get_doc(target)
253
254         company_currency = frappe.get_cached_value('Company', quotation.company, "default_currency")
255
256         if company_currency == quotation.currency:
257             exchange_rate = 1
258         else:
259             exchange_rate = get_exchange_rate(quotation.currency, company_currency,
260                 quotation.transaction_date, args="for_selling")
261
262         quotation.conversion_rate = exchange_rate
263
264     # get default taxes
265     taxes = get_default_taxes_and_charges("Sales Taxes and Charges Template", company=quotation.company)
266     if taxes.get('taxes'):
267         quotation.update(taxes)
268
269     quotation.run_method("set_missing_values")
270     quotation.run_method("calculate_taxes_and_totals")
271     if not source.with_items:
272         quotation.opportunity = source.name
273
274     doclist = get_mapped_doc("Opportunity", source_name, {
275         "Opportunity": {
276             "doctype": "Quotation",
277             "field_map": {
278                 "opportunity_from": "quotation_to",
279                 "name": "enq_no"
280             }
281         },
282         "Opportunity Item": {
283             "doctype": "Quotation Item",
284             "field_map": {
285                 "parent": "prevdoc_docname",
286                 "parenttype": "prevdoc_doctype",
287                 "uom": "stock_uom"
288             },
289             "add_if_empty": True
290         }
291     }, target_doc, set_missing_values)
292
293     return doclist

```

1

2

3

4

Abstimmung mit Herr Steffen Paul

26.11.2021

Die Felder im Kasten Eigenschaften sollten vom Angebot bis hin zur Ausgangsrechnung durchgezogen werden.

Eigenschaften

Angebotsposition Beschaffung

Positionsnummer Menge Mengeneinheit Produktnummer Produkttext kurz

02 1 Stk. 1004-17 Moldex3D Professional Paket 2019 ☐ Nur Text

Preis Liste Verkauf 50.000,00 € Preis Verkauf Position Einzel 50.000,00 € Einzelrabatt 0,00 % ☐ Seitenun

Preis Verkauf Position 50.000,00 € Gesamtrabatt ---

Produkttext lang

- 1 x Studio
- 1 x Studio Advanced
- 1 x Flow
- 1 x Pack
- 1 x Cool (TC)
- 1 x Warp
- 1 x 3D Coolant CFD
- 8 PP

Pos. Art Bestandtei

Tabellen

Tab 1	Tab 2	Tab 3	Tab 4	Tab 5	Tab 6	Tab 7	Tab 8	Tab 9	Tab 10
02	1	Stk.	1004-17	Moldex3D Professional Paket 2019	- 1 x Studio	50.000,00 €	50.000,00 €	50.000,00 €	50.000,00 €
					- 1 x Studio Advanced				
					- 1 x Flow				
					- 1 x Pack				

Versuch die Ausbaustufe der
Angebotsvorlage auf die
Opportunity anzuwenden
Clientscript

E

> Clientskript > Opportunity-Form

Suche oder Befehl eingeben (Strg + G)

Hilfe

A

Opportunity-Form

Aktiviert

Gehe zu Opportunity

Skript für Child Table hinzufügen

<

>

🖨

⋮

Speichern

Zuweisungen

+

Anhänge

Datei anhängen

+

Geteilt mit

+

Schlagworte

Schlagwort hinzufügen...

0 · 0 · BEOBACHTEN

Benutzer edited this vor 3 Minute(n)

Benutzer created this vor 14 Minute(n)

DocType *

Chance

Apply To

Form

☒ Aktiviert

Skript

```
1 frappe.ui.form.on('Opportunity', {
2     refresh(frm) {
3         var me = this;
4         this.frm.add_custom_button(__('Maintenance Schedule'),
5             function () {
6                 erpnext.utils.map_current_doc({
7                     method: "erpnext.maintenance.doctype.maintenance_schedule.maintenance_schedule.make_maintenance_visit",
8                     source_doctype: "Angebotsvorlage",
9                     target: me.frm,
10                    setters: {
11                        },
12                    get_query_filters: {
13                        },
14                })
15            }, __("Get Items From"));
16    },
17 })
```

Serverscript

E

> Serverskript > Make Opportunity

Suche oder Befehl eingeben (Strg + G)

Hilfe

A

Make Opportunity

Aktiviert

<

>

🖨

⋮

Speichern

Zuweisungen

+

Anhänge

Datei anhängen

+

Geteilt mit

+

Schlagworte

Schlagwort hinzufügen...

0 · 0 · BEOBACHTEN

Benutzer edited this vor 4 Minute(n)

Benutzer created this vor 12 Minute(n)

Skripttyp *

API

☐ Deaktiviert

API-Methode

make_opportunity

☐ Gast zulassen

Skript *

```
1 def make_opportunity(source_name, target_doc=None, item_name=None, s_id=None):
2     from frappe.model.mapper import get_mapped_doc
3
4     doclist = get_mapped_doc("Angebotsvorlage", source_name, {
5         "Angebotsvorlage": {
6             "doctype": "Opportunity",
7             "field_map": {
8                 # "target_field": "source_field"
9             },
10            "validation": {
11                # "docstatus": ["=", 1]
12            },
13            # "postprocess": update_status_and_detail
14        },
15        "Angebotsvorlage Item": {
16            "doctype": "Opportunity Item",
17            "condition": lambda doc: doc.item_name == item_name,
18            # "postprocess": update_sales_and_serial
19        }
20    }, target_doc)
21
22     return doclist
```

Ergebnisse

Auf diesem Weg kann eine Childtable nicht übertragen werden. Jedoch war es möglich einen aus zwei Komponenten bestehenden Front-End / Back-End Skript zu erstellen.

Client Skript auf Opportunity

```
frappe.ui.form.on('Opportunity', {
  refresh(frm) {
    frm.add_custom_button(
      'Angebotsvorlage',
      frappe.call({
        method: "make_opportunity",
        args: {
          'doctype': 'Item'
        },
        callback: function(r){
          console.log(r);
        },
      }),
      __("Get Items From"));
  }
})
```

Back End Skript vom Typ API

The screenshot shows the 'Make Opportunity' configuration page in Frappe. The page has a top navigation bar with a search bar and a 'Hilfe' button. The main content area is divided into two sections: 'API' and 'Skript'. The 'API' section is active and shows the 'API-Methode' set to 'make_opportunity' and the 'Gast zulassen' checkbox checked. The 'Skript' section shows a single line of code: `frappe.response["message"] = "hello"`. The left sidebar contains links for 'Zuweisungen', 'Anhänge', 'Geteilt mit', and 'Schlagworte'. The bottom of the page shows a 'BEOBACHTEN' button.

Warum funktioniert dies nicht mit der Angebotsvorlage?

Der Client Script ruft hierbei nicht die Funktion `frappe.call(...)` (wie in dem letzten Beispiel), sondern die Funktion `frappe.model.mapper.get_mapped_doc(...)` (vorletztes Beispiel) auf. Dieser

Funktionsaufruf ist so wie es aussieht nicht von einem Client Script aufrufbar. Sehr schade, aber leider nicht abänderbar.

Weitere Erkenntnisse

Server Script haben ein Sicherheitslayer, welches nur eine Teilmenge der in Python üblichen Methoden & Funktionen aufzurufen:

<https://docs.erpnext.com/docs/v13/user/manual/en/customize-erpnext/server-script#23-api-scripts>

Weitere Dokumentation

<https://frappeframework.com/docs/v13/user/en/desk/scripting/server-script>

https://frappeframework.com/docs/v13/user/en/guides/basics/frappe_ajax_call#calling-standard-api

Robuste finale Version

Die Einbau- und Anbringungsanleitung findet sich im ERPNext Benutzerhandbuch und kann auf andere Doctypes angewandt werden. Siehe <https://doku.phamos.eu/books/erpnext-benutzerhandbuch/page/copy-childtable-get-items-from>

Version #22

Erstellt: 19 November 2021 12:46:38

Zuletzt aktualisiert: 23 November 2023 14:39:25