Eigene Funktionen

- Ortsteil überOpenstreetmap holen
- Aktelleus Datum mit Uhrzeit in ein Feld setzten

Ortsteil überOpenstreetmap holen

Beschreibung:

Eine Frontendfunktion die über einen Button ausgelöst werden kann um den Ortsteil zu holen.
Übergeben werden die Felder aus dem Formular wo die Funktion drin ist.
Und als Rückgabe der Ortsteil der dann dem Feld zugewiesen wird.
Frontend Funktionen sind Java Script Funktionen

Die Funktion ist in zwei teile geteilt.

Einmal die httpget Funktion die ein Data Objekt zurück gibt
Und die parse Funktion in der httpget aufgerufen wird

```
function httpGet(url, onSuccess, onError) {
    console.log("Anfrage wird gestartet:", url);
    fetch(url, {
        method: "GET",
        headers: {
            "User-Agent": "Rei3-Frontend"
        }
    })
        .then(response => {
            if (!response.ok) {
                throw new Error(`HTTP-Fehler! Status: ${response.status}`);
            }
            return response.json();
        })
        .then(data => {
            if (onSuccess) {
                onSuccess(data);
            }
```

```
        })
        .catch(error => {
            console.error("Fehler bei der HTTP-Anfrage:", error);
            if (onError) {
                onError(error);
            }
        });
}


function httpGetold(url, onSuccess, onError) {
app.dialog_show("Info","Start");
    const xhr = new XMLHttpRequest();
    xhr.open("GET", url, true);
    app.dialog_show("Info","Open");
    xhr.onload = function () {
        if (xhr.status >= 200 && xhr.status < 300) {
            try {
                const response = JSON.parse(xhr.responseText);
                if (onSuccess) {
                    onSuccess(response);
                }
            } catch (error) {
                app.dialog_show("Info","Fehler beim Verarbeiten der HTTP-Antwort:"+ error);
                if (onError) {
                    onError(error);
                }
            }
        } else {
            app.dialog_show("Info","HTTP-Fehler:"+xhr.status+" "+xhr.statusText);
            if (onError) {
                onError(xhr.statusText);
            }
        }
    };

    xhr.onerror = function () {
        app.dialog_show("Info","Netzwerkfehler bei der HTTP-Anfrage.");
        if (onError) {
            onError("Netzwerkfehler");
```

```
        }
    };
app.dialog_show("Info","Ende httpget");
    xhr.send();

}

function fetchSuburbFromOSM() {
    // Werte aus den Formularfeldern abrufen
    const street = app.get_field_value({F3: 0 customer_adresses.street});
    const postcode = app.get_field_value({F5: 0 customer_adresses.plz});
    const city = app.get_field_value({F6: 0 customer_adresses.city});

    // Überprüfen, ob die erforderlichen Felder ausgefüllt sind
    if (!street || !postcode || !city) {
         app.dialog_show("Info", "Bitte füllen Sie alle erforderlichen Felder aus (Straße, Postleitzahl, Ort).");
        return;
    }

    // Anfrage-URL für die Nominatim API erstellen
    const query = `${street}, ${postcode}, ${city}`;
    const url =
`https://nominatim.openstreetmap.org/search?q=${encodeURIComponent(query)}&format=json&addressdetail
s=1`;

    // HTTP-GET-Aufruf
    httpGet(
        url,
        function (data) {
            // Erfolg: API-Antwort analysieren

            if (data.length > 0) {
            
                const address = data[0].address;
                const suburb = address.village || address.hamlet || null;
□□□□
                if (suburb) {
                    app.set_field_value({F9: 0 customer_adresses.district}, suburb, true);
                    app.dialog_show("Erfolg", `Ortsteil/Hamlet: ${suburb}`);
                } else {
```

```
                app.dialog_show("Info", "Dieser Ort hat keinen spezifischen Ortsteil.");
                app.set_field_value({F9: 0 customer_adresses.district}, "", true);
            }
        } else {
            app.dialog_show("Warnung", "Keine Ergebnisse für die eingegebene Adresse gefunden.");
            app.set_field_value({F9: 0 customer_adresses.district}, "", true);
        }

    },
    function (error) {
        // Fehler: Meldung ausgeben
        app.dialog_show("Fehler", "Die Anfrage an OpenStreetMap ist fehlgeschlagen.");
    }
  );

}
fetchSuburbFromOSM();
```

Aktelleus Datum mit Uhrzeit in ein Feld setzten

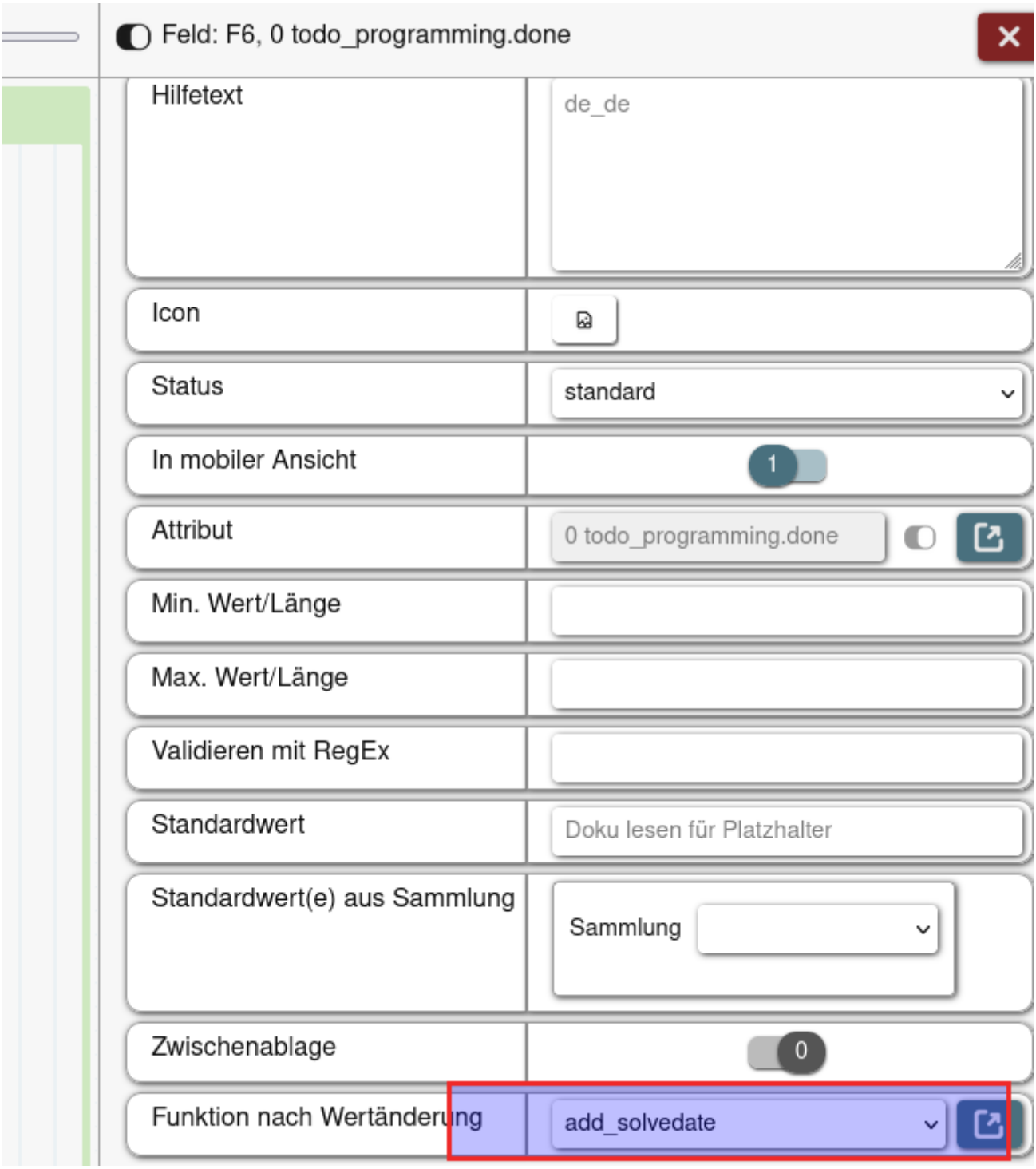
Beschreibung:

Man möchte das ein Datumsfeld beim speichern zum Beispiel gesetzt wird.
Funktioniert auch, falls das Datumsfeld versteckt ist oder disabled ist.

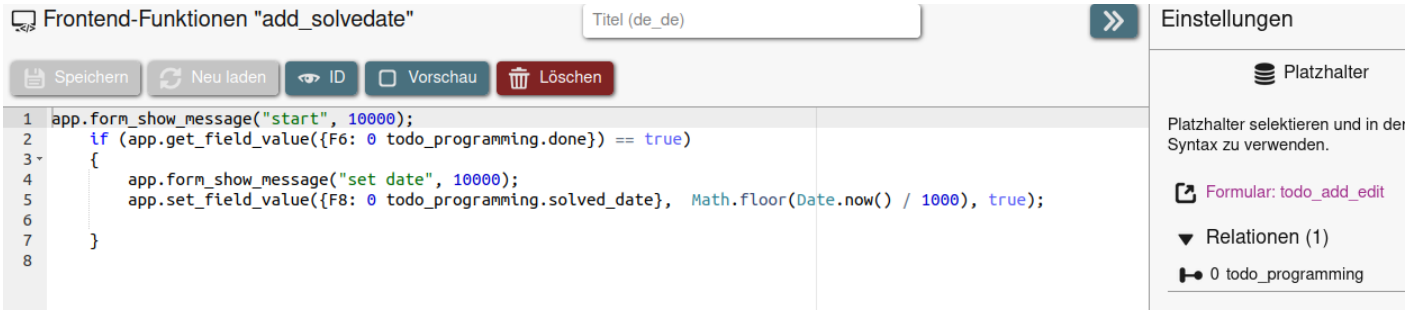
Lösung:

Das machen wir mit einer Frontend Funktion

Diese kann zum beispiel bei einer Wert Anderung aufgerufen werden:



Wir erstellen also erst die Frontendfunktion und lassen diese auf das jeweilige Formualr verknüpfen



Nun der Code

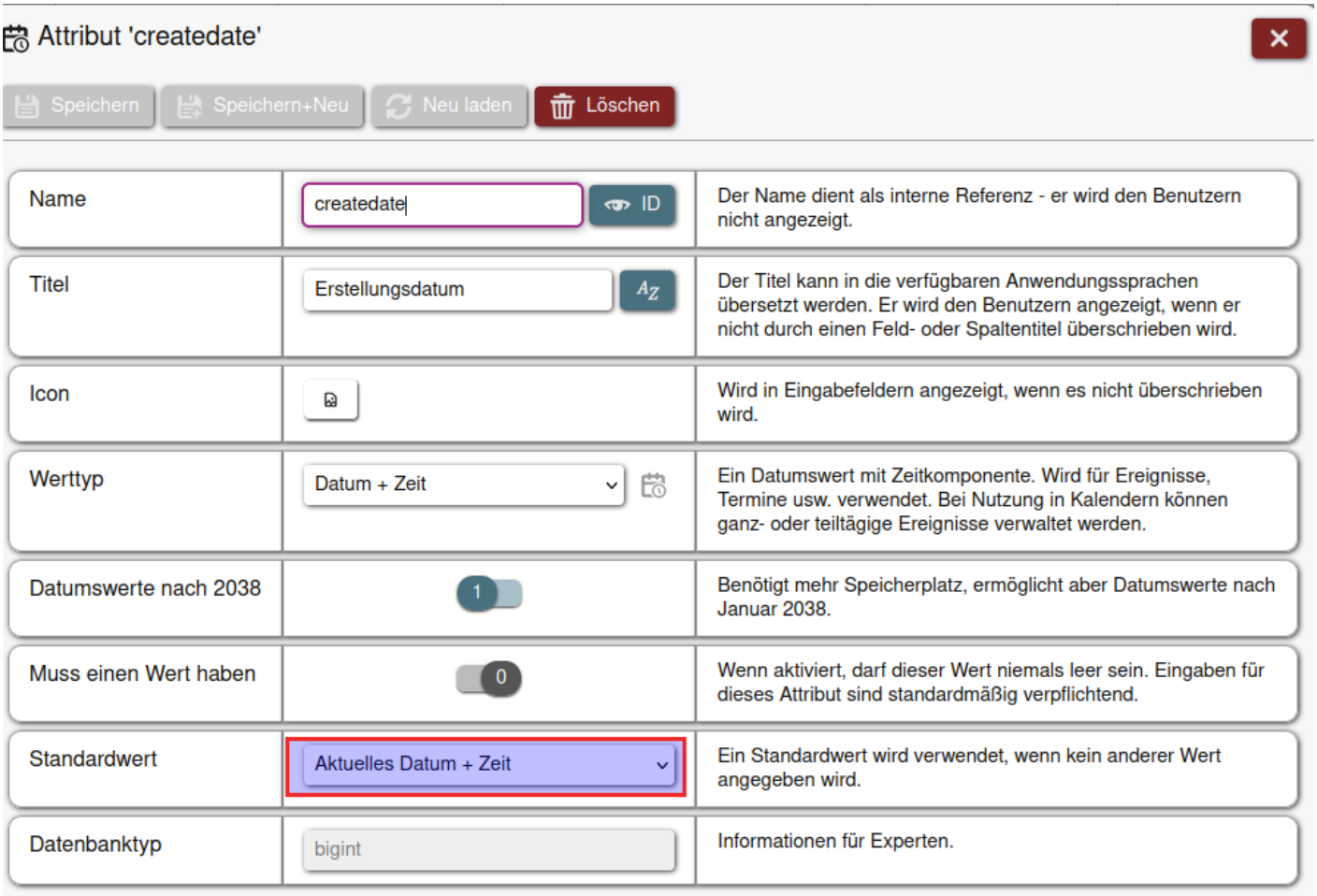
```
app.form_show_message("start", 10000);
if (app.get_field_value({F6: 0 todo_programming.done}) == true)
{
  app.form_show_message("set date", 10000);
  app.set_field_value({F8: 0 todo_programming.solved_date},  Math.floor(Date.now() / 1000), true);

}
```

Was macht der Code, wir fragebn ab ob Feld 6 Ein boolean Feld wahr also true ist.
Ist dies der Fall soll das Feld F8 ein Datumsfeld auf das Aktuelle Datum gesetzt werden.
Der Wert ist der Wert in Sekunden, das Datumsfeld wiederum interpretiert daraus das Datum und Uhrzeit.
So kann in dem Datumsfled in den benutzereinstellungen jede anziege von Datumsformaten genutzt werden. Entwerder AMI oder Europisch oder oder.

Der eigentliche Code für die aktuelle Zeit ist:

```
Math.floor(Date.now() / 1000)
```

Für die Erstellung eines Datensatzes mit Aktuellem Datum, brauchen wir keine Funktion, das kann man in der Tabelle/Relation hinterlegen. Wir brauchen die Funktion oben zum nachträglichen ändern.


Attribut 'createdate'
Speichern
Speichern+Neu
Neu laden
Löschen
Name
createdate
ID
Der Name dient als interne Referenz - er wird den Benutzern nicht angezeigt.
Titel
Erstellungsdatum
Der Titel kann in die verfügbaren Anwendungssprachen übersetzt werden. Er wird den Benutzern angezeigt, wenn er nicht durch einen Feld- oder Spaltentitel überschrieben wird.
Icon
Wird in Eingabefeldern angezeigt, wenn es nicht überschrieben wird.
Werttyp
Datum + Zeit
Ein Datumswert mit Zeitkomponente. Wird für Ereignisse, Termine usw. verwendet. Bei Nutzung in Kalendern können ganz- oder teiltägige Ereignisse verwaltet werden.
Datumswerte nach 2038
1
Benötigt mehr Speicherplatz, ermöglicht aber Datumswerte nach Januar 2038.
Muss einen Wert haben
0
Wenn aktiviert, darf dieser Wert niemals leer sein. Eingaben für dieses Attribut sind standardmäßig verpflichtend.
Standardwert
Aktuelles Datum + Zeit
Ein Standardwert wird verwendet, wenn kein anderer Wert angegeben wird.
Datenbanktyp
bigint
Informationen für Experten.