

# Dynamic Forms and Widgets with QGIS Expressions

By Example of «Mehrjahresprogramm Natur und Landschaft» Kt. Solothurn

## Outline

- Default values and updates
- Constraints dependent on other widget values
- Data defined widget visibility
- More sophisticated overlay functions
- Sending form data and map extents to external systems (reporting)

The screenshot shows a configuration panel for a widget. It is divided into three main sections:

- Widget Type:** A dropdown menu is set to 'List'. Below it, under 'Treatment of Empty Cells', the 'NULL' radio button is selected.
- Constraints:** This section contains several checkboxes: 'Not null' (checked), 'Unique' (unchecked), and 'Enforce expression constraint' (unchecked). There are also fields for 'Expression' and 'Expression description', and checkboxes for 'Enforce not null constraint' and 'Enforce unique constraint'.
- Defaults:** The 'Default value' field contains the text 'pression:=nummer, min\_inscribed\_circle\_radius:=2.0, sort\_by\_intersection\_size:='desc''. Below this, the 'Preview' field shows '88, 90294, 90208'. The 'Apply default value on update' checkbox is checked and highlighted with a red rectangle.

If «Apply default value on update» is enabled, then this widget can react on changes in other fields (widgets)

Multiple dependencies (one dependent widget depending on another dependent widget) should be avoided

# Flexible constraints with expressions – dependent on other fields / widgets

**▼ Constraints**

Not null  Enforce not null constraint

Unique  Enforce unique constraint

Expression

Expression description

Enforce expression constraint

**▼ Defaults**

Default value

Preview 0.00

Apply default value on update

Constraint on «Abgeltung» (compensation) is dependent on «Wiesenkategorie» (grassland category)

Result of expression needs to result in 0 or 1 [true|false]

Limitation: expression description (explaining a constraint violation) cannot be data-defined.

# Flexible constraints with expressions – dependent on other fields / widgets

```
wiesenkategorie_abgeltung_ha >=
```

lower bound

```
CASE
  WHEN wiesenkategorie = 'Kat_RF' THEN 0
  WHEN wiesenkategorie = 'Kat_RF_II' THEN 100
  WHEN wiesenkategorie = 'Kat_II_RF' THEN 200
  WHEN wiesenkategorie = 'Kat_II_artenreicheWiese' THEN 400
  WHEN wiesenkategorie = 'Kat_I_besondersartenreicheWiese' THEN 600
END
```

```
AND
```

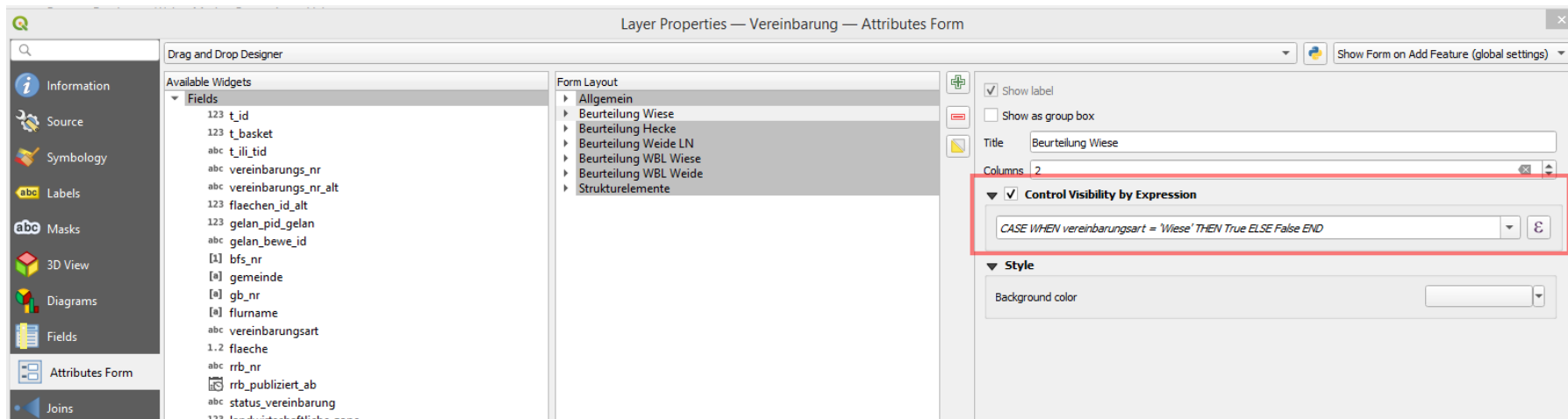
```
wiesenkategorie_abgeltung_ha <=
```

upper bound

```
CASE
  WHEN wiesenkategorie = 'Kat_RF' THEN 0
  WHEN wiesenkategorie = 'Kat_RF_II' THEN 100
  WHEN wiesenkategorie = 'Kat_II_RF' THEN 300
  WHEN wiesenkategorie = 'Kat_II_artenreicheWiese' THEN 500
  WHEN wiesenkategorie = 'Kat_I_besondersartenreicheWiese' THEN 800
END
```

Result of Expression needs to be 0|1 or True|False

# Data defined visibility of Group Containers



Works for «tabs» and «group boxes» (not for single widgets)

Can be dependent on other fields / widgets

# More sophisticated overlay functions

Get proper overlay results (spatial relations).

Inclusion criteria:

- `min_overlap` (area in map units)
- `min_inscribed_circle_radius` (the maximum possible inscribed circle in the intersection area)

Optional:

- `«sort_by_intersection_size»`  
(*descending/ascending*)
- `«return_details»`:  
*get back «map» data structure with  
«feature\_id», «expression\_result»,  
«overlay\_area», «max\_inscribed\_circle\_radius»*

The screenshot shows a portion of a QGIS expression editor. At the top, there is an 'Expression description' text box and an unchecked checkbox for 'Enforce expression constraint'. Below this is a section titled 'Defaults' with a downward-pointing arrow. This section is highlighted with a red border. It contains a 'Default value' text box with the expression `pression:=number, min_inscribed_circle_radius:=2.0, sort_by_intersection_size:='desc'`, a clear button (X), and a help button (E). Below the text box is a 'Preview' field showing the output `88, 90294, 90208`. At the bottom of the 'Defaults' section is a checked checkbox for 'Apply default value on update'.

# More sophisticated overlay functions

Example: automatically assign municipality, Flurnamen, parcel numbers, etc. through spatial relations

Vereinbarung - Feature Attributes

Algemein Beurteilung Wiese

t\_id 438260 ✓ t\_id\_tid 64e49dc-1-526a-4471-86a4-6357ccc78f2c ✓

Vereinbarungsart Wiese ✓ Status der Vereinbarung in Bearbeitung ✓

Vereinbarungs-Nr 614245\_WI\_04 ✓ Alte Vereinbarungs-Nr (Optional wenn migriert) NULL ✓

Alte Flächen-ID NULL ✓ Fläche [ha] 2.89 ✓

GELAN Person Henzi Christoph & Dalhäuser Doris (4512, Bellach) ✓ GELAN Bewirtschaftungseinheit (260269,634782,634782) ✓

RRB-Nr NULL ✓ RRB Publikationsdatum NULL ✓

Flurname

Value	Value
Grössli	2556
Allmend ✓	Bfs-Nr ✓

Gemeinde(n)

Value	Value
Setzach ✓	88
	GB_Nr ✓
	90294
	90208

Landwirtschaftliche\_Zone NULL ✓ UZI\_Subregion 1.5 (Freiburger, Berner und Solothurner Mittelland, östliche Waadt) ✓

Dateipfad\_oder\_URL undefiniert ✓ Kontrollintervall (Jahre) 4 ✓

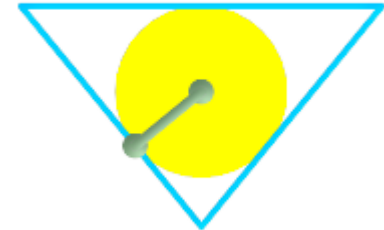
Bemerkung

MJPNL\_Version MJPNL-2032 ✓



Example: expression to automatically assign Flurnamen based on spatial relations

```
overlay_intersects(  
  layer:='AV Flurname',  
  expression:=flurname,  
  min_inscribed_circle_radius:=3,  
  sort_by_intersection_size:='desc'  
)
```



Example of minimum inscribed circle in a polygon

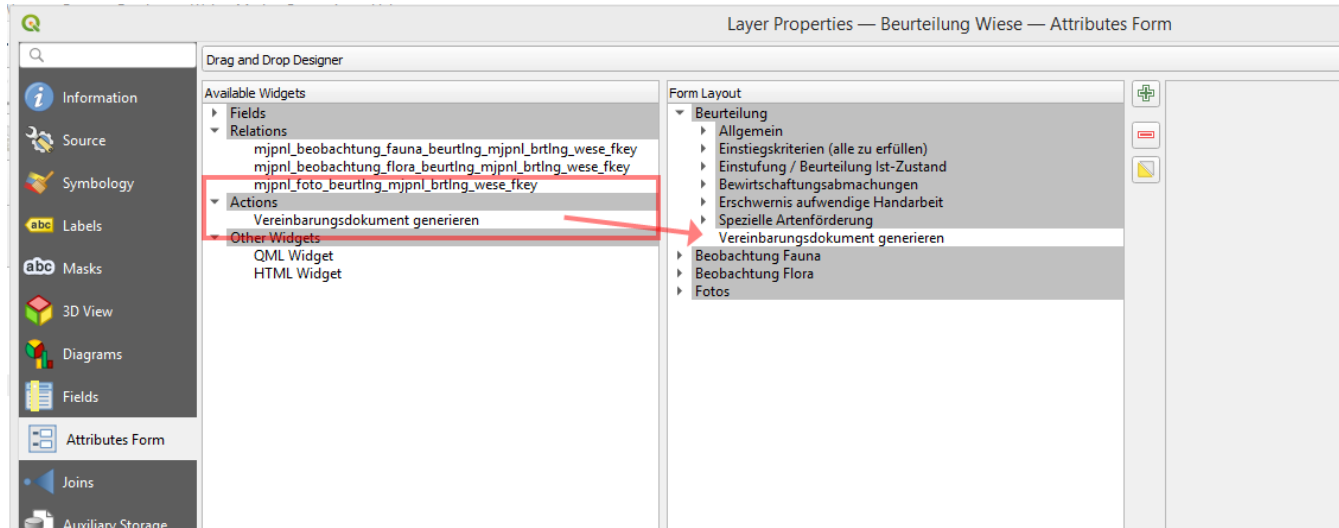
# More sophisticated overlay functions

```
attribute(  
  get_feature(  
    'GELAN Person',  
    't_id',  
    attribute(  
      get_feature(  
        'Betrieb',  
        't_id',  
        attribute(  
          get_feature(  
            'Bewirtschaftungseinheit',  
            'bewe_id',  
            array_first(  
              overlay_intersects(  
                layer:='Bewirtschaftungseinheit',  
                expression:="bewe_id",  
                sort_by_intersection_size:='des',  
                limit:=1  
              )  
            )  
          ),  
          'betrieb'  
        )  
      ),  
      'person'  
    )  
  ),  
  'pid_gelan'  
)
```

Example: get «Person» (farmer) through overlay function and several relations:

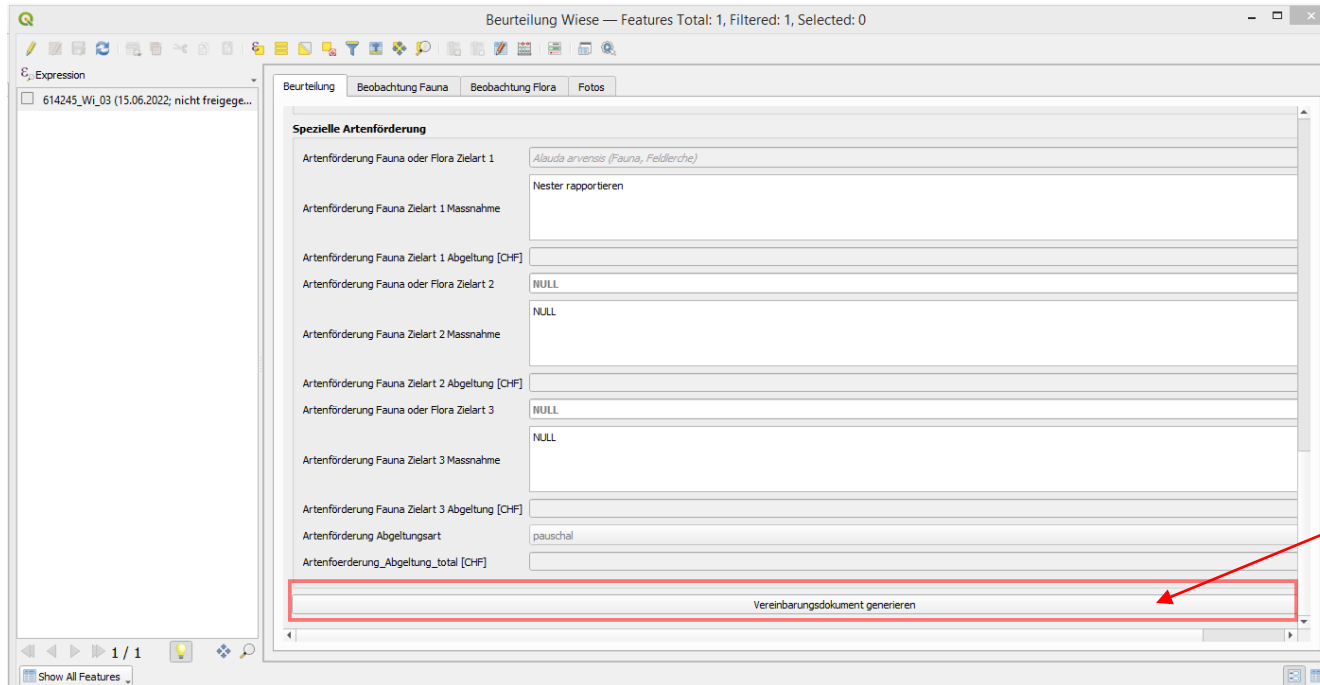
Bewirtschaftungseinheit → Betrieb → Person

All actions can be assigned to a button and used in drag and drop form



For the definition of the action see the following slides

All actions can be assigned to a button and used in drag and drop form



This button starts the action

# Web-Service calls using actions

Two new action types for POST requests:

- Submit URL (multipart)
- Submit URL (url encoded or JSON)

Expression function

**`represent_attributes(feature)`**

returns all attributes in a «map» data structure (key/values)

**`url_encode()`**

Encodes non-ascii characters

**Edit Action**

Type: **Submit URL (multipart)**  Capture output

Description: Vereinbarungsdokument generieren

Short Name: Leave empty to use only icon

Icon:

Action Scopes

- Layer
- Canvas
- Field
- Feature

Action Text

The action text defines what happens if the action is triggered.  
The content depends on the type.  
For the type *Python* the content should be python code  
For other types it should be a file or application with optional parameters

```
1 = with_variable(  
2   'vereinbarung',  
3   get_feature(  
4     'vereinbarung',  
5     't_id',  
6     vereinbarung  
7   ),  
8   'https://dox42.verw.rootso.org/dox42RestService.ashx?' | |  
9   url_encode(  
10  map concat(
```

123

Execute if notification matches

Enable only when editable

# Atlas images sent to web service

Requires Plugin «AtlasExportFunction» and a layout with an atlas

New expression function:

```
atlas_image(  
    dpi:=150,  
    layout_name:='Karte_Detail',  
    image_format:= png  
)
```

Can be sent as url\_encoded binary data through

# Code to send attribute data with relations to external Reporting service through Post-URLs

```
with_variable(  
  -- because we need a reference to "vereinbarung"  
  -- multiple times we store it in a temporary variable  
  'vereinbarung',  
  get_feature(  
    'Vereinbarung',  
    't_id',  
    vereinbarung  
  ),  
  -- base URL of external dox42 reporting service  
  'https://dox42.verw.rootso.org/dox42RestService.ashx?' ||  
  url_encode(  
    map_concat(  
      -- Basis URL und Parameter mit Bildern  
      map(  
        'Operation','GenerateDocument',  
        -- template Word file (Path and file name)  
        'DocTemplate','c%3A\\dox42Server\\templates\\Wiese.docx',  
        'ReturnAction.Format','pdf',  
        -- atlas print Detailkarte (default = format png)  
        'InputParam.v_karte_detail',to_base64(  
          atlas_image(  
            dpi:=150,  
            layout_name:='Karte_Detail'  
          )  
        ),  
        -- atlas print Übersichtskarte  
        'InputParam.v_karte_uebersicht',to_base64(  
          atlas_image(  
            dpi:=150,  
            layout_name:='Karte_Uebersicht'  
          )  
        )  
      ), -- end of first "map" (Basis-URL und Bilder)  
      -- Attribute Beurteilung (Wiese)  
      map_prefix_keys(represent_attributes($currentfeature),'InputParam.b_'),  
      -- Attribute der Vereinbarung  
      map_prefix_keys(  
        -- gelan_bewe_id auf Originalwert (nicht Repräsentation) zurücksetzen  
        map_insert(  
          map_delete(  
            -- gelan_pid_gelan auf Originalwert (nicht Repräsentation) zurücksetzen  
            map_insert(  
              map_delete(  
                represent_attributes(  
                  'Vereinbarung',  
                  @vereinbarung  
                ),  
                'gelan_pid_gelan'  
              ),  
              'gelan_pid_gelan',  
              attribute(  
                @vereinbarung,  
                'gelan_pid_gelan'  
              )  
            ),  
            'gelan_bewe_id'  
          ),  
          'gelan_bewe_id',  
          attribute(  
            @vereinbarung,  
            'gelan_bewe_id'  
          )  
        ),  
        'InputParam.v_'  
      ),  
      -- Attribute der GELAN Person  
      map_prefix_keys(  
        represent_attributes(  
          'GELAN Person',  
          get_feature(  
            'GELAN Person',  
            'pid_gelan',  
            attribute(  
              @vereinbarung,  
              'gelan_pid_gelan'  
            )  
          ),  
          'InputParam.p_'  
        ) -- end of map_concat  
      ) - end of url_encode  
    ) - end of with_variable
```