iFinder Searchbar Sitesearch Integration - Template Modification

For Version: 5.2

Document Version: 1

Last Edited: March 2019

Copyright and Trademarks

IntraFind Software AG
Landsberger Straße 368
80687 Munich
Germany

Phone:	+49 89 3090446-0
Fax:	+49 89 3090446-29
Web:	https://www.intrafind.com/

Table of Contents

1 iFinder Searchbar Sitesearch integration– Introduction

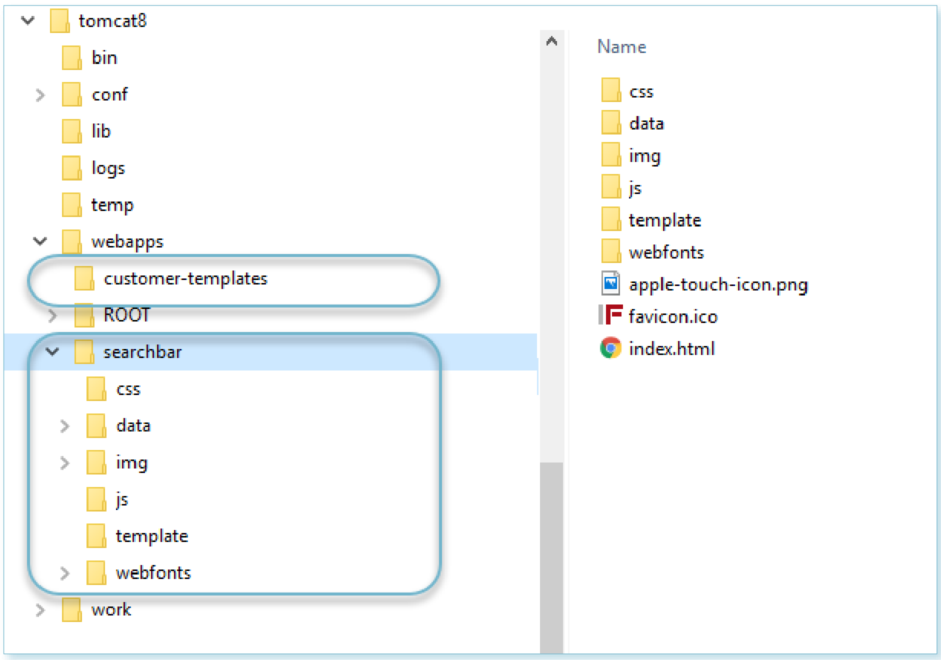
IFinder5 elastic offers a searchbar which can easily be integrated into existing environments, for instance:

- Customer portals
- Customer websites and intranets
- Web apps

This document describes the searchbar variant on which SiteSearch is based. It comes with only a part of the iFinder Searchbar features.

2 Recommended folder structure for deployment on web server:

We recommend the following structure for deploying the searchbar on your web server. If you need to modify or customize some of the templates used for the searchbar, this setup ensures that you do not overwrite customized template files with updates of the searchbar.

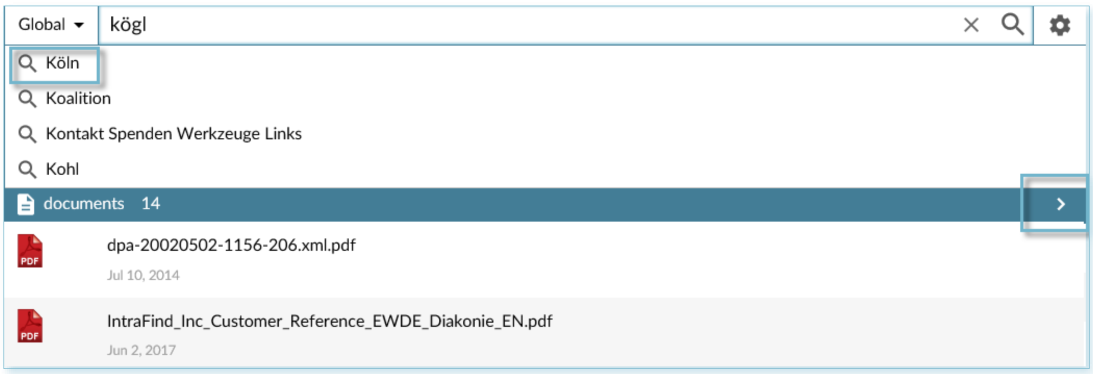


```
tomcat8
  bin
  conf
  lib
  logs
  temp
  webapps
    customer-templates
    ROOT
    searchbar
      css
      data
      img
      js
      template
      webfonts
  work
```

```
Name
css
data
img
js
template
webfonts
apple-touch-icon.png
favicon.ico
index.html
```

3 Inside and outside communication with the searchbar via EventBus

EventBus is used for inside and outside communication for and with the searchbar.
EventBus is involved whenever a search is triggered, for instance by clicking one of the auto-completes in the search dropdown or one of the links, like **Search in documents**.


Global
kögl
Köln
Koalition
Kontakt Spenden Werkzeuge Links
Kohl
documents 14
dpa-20020502-1156-206.xml.pdf
Jul 10, 2014
IntraFind_Inc_Customer_Reference_EWDE_Diakonie_EN.pdf
Jun 2, 2017

This chapter lists some standard parameters that are triggered with typical searchbar events. For an extensive list of all parameters and options available with the EventBus, please refer to the internet.

SEARCHBAR_RENDERED_INITIALLY:,

Parameter: -
Is executed when the searchbar is loaded.

QUERY_RESPONSE_RECEIVED:

Parameter:

- query – Specify query
- sendTime: Send time for query
- receiveTime: Receipt time for query

Is called for each request of the application (except retrieval of configuration and messages)

FILTER_TOGGLED:

Parameter:

- isSet: Boolean: indicates if filter is visible

Is called when filter visibility changes via click on **Filter** icon.

SEARCH_TRIGGERED:

Parameter:

- query: Content of input field

Is called when an explicit search is carried out, either by clicking the **Search** icon or when hitting **Enter** from the search input field.

SEARCH_CATEGORY_TRIGGERED:

Parameter:

- query: Content of input field
- id: ID used to select a “connector” from config.json for the query

Is called when a search is carried out via one of the jump points of the search dropdown, for example **Search in Documents** or **Search in People**, etc.

SEARCH_ITEM_TRIGGERED:

Parameter:

- category: Type of hit
- clickTarget: element clicked
- item: data of hit in JSON
- query: Content of input field
- origin: original event

Is called when one of the hits in the search dropdown is clicked.

SEARCH_SCROLL_TRIGGERED:

Parameter:

- query: Content of input field
- start: next page to load

Is called when user is scrolling to load further results.

SORT_RESULT_EVENT_TRIGGERED:

Parameter:

- sortParams: sorting parameter

Is called when sorting is clicked.

NEW_RESULT_LOADED:

Parameter:

- id: id of the new result list

Is called when a new hitlist is loaded. Can be used for communication with the host application for switching to the new hitlist (e.g. for loading a new tab)

RESULT_TARGET_CHANGED:

Parameter:

- oldTarget: ID of the previous hitlist
- newTarget: ID of the hitlist that will be loaded

When using several hitlists (for instance with tabs), this event should be used to tell the searchbar that there was a change.

4 Modifying a template

Caution: This guide is intended for technical users and assumes that you are familiar with HTML and JavaScript. The configuration file is critical to the operation of the system.
Please always make a copy of config.json before editing it.

4.1 Introduction

The layout and content of the configuration elements making up the searchbar, the search dropdown, the source-based result lists and other details use the handlebars template engine and the content resides in so-called **handlebars template files** (*.hbs).

Important Note: For better performance, the handlebars templates referenced in the standard configuration *are compiled into a JavaScript object* during the build process of the application.
This means that they are ***not available for editing*** in their original form. There is a way to customize the templates, however. This is described in the following.

It is possible to modify the following elements, if necessary:

- Search dropdown templates (SWYT)
- Hitlist templates

Other elements, for instance facets or index fields templates should not be touched.

As part of the installation package, we deliver a set of standard templates used with the default installation of the searchbar. You can use these templates as examples and copy their content to create the new reference template (see below for the required steps).

However, **editing the templates themselves will have no effect on the layout and look of your searchbar installation**.

Instead, you need to proceed as follows.

4.2 Steps required for modifying searchbar template content

To modify a template for the searchbar, you first need to register an additional handlebars template file (.hbs) in the configuration file (config.json). This new file is then used to define new templates which can be addressed in the configuration via unique IDs.

This one file will contain all the custom templates, each wrapped into its own script id.

Here is a step-by-step overview summarizing the steps required for modifying template contents for your searchbar. Each step is explained in detail in the subsequent chapters.

Steps required for modifying your searchbar templates:

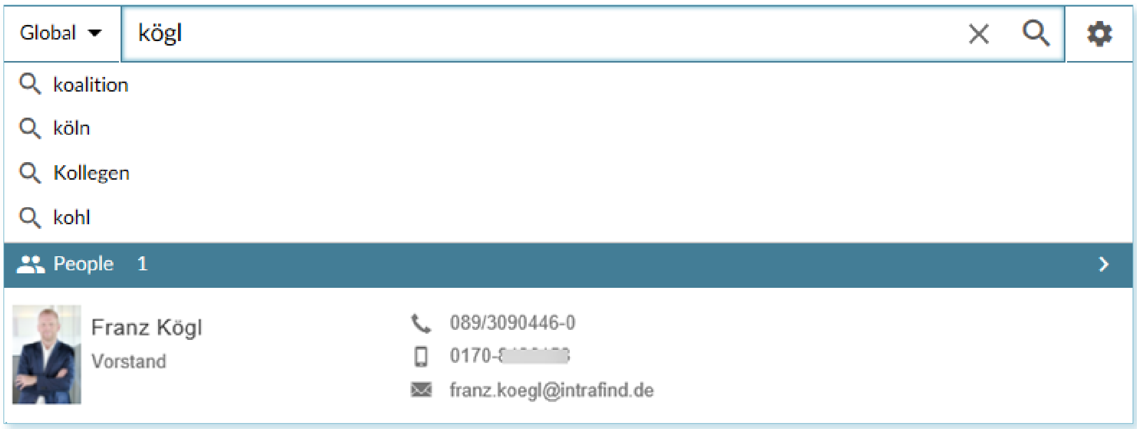
1. Find out what the configuration element you want to change is called, i.e. how it is referenced in config.json. (for example people)
2. Locate the associated template reference in config.json (example item.template: people.hbs)
3. Create a new template file on a path on the web server. Ideally, this is outside the searchbar folder, but in a parallel customer template folder.
4. Create the required template in the new customer-template file. You need a unique script tag for each template you want to modify.

   (Example `<script id="id_for_my_new_people_template"> … </script>`)
   This id should only contain ascii characters and no whitespace.
5. Open the sample people.hbs template that comes with the installation package; view the content and copy the required content in between the script tags.
6. Add a TEST heading to the copied content to later see whether you are referencing the changed template ID.
7. Under "components" in config.json, edit the value of "templatesUrl" to point to the location of your new hbs file, e.g. `/custom-templates/custom.hbs`
8. Under "resulttype", add or edit a result type using your new template. As value of "hitTemplate", use the id of your new template definition. This is the string that you supplied as script id for this template in your custom templates file.
9. Add new index content to the customer template; use `{{debug}}` option to find out which content is available.

The following example will clarify how you can modify a template and exemplify each of the above steps in detail.

Sample Scenario

In our sample scenario, we want to modify the **SWYT** template for **people** by adding the **Wiki entry URL** for the respective person. The standard people template lists the following details:



Franz Kögl
Vorstand
089/3090446-0
0170-[redacted]
franz.koegl@intrafind.de

4.2.1 Finding out which reference to use

1. First you need to determine what the configuration element you want to change is called, i.e. how it is *referenced* in config.json.

   All search-drop down configuration elements are listed in the categories array under components:

   ```
   "categories": [
           "autocomplete",
           "synonyms",
           "translations",
           "people",
           "documents"
         ],
   ```

2. Then you search for this reference/category name in config.json, search for “people”.
3. For each dropdown configuration element, there is an array showing the respective item and wrapper files.

Note: Templates are grouped into wrapper template files. Take note of the file name used for the itemTemplate file.
(The itemTemplate defines the list of item template associated with the respective element.)

4.2.2 Locating the required template reference in the config file

Locating the required SWYT template reference for people:

There are two places in the searchbar for people details:

- The people element in the **resulttype** list.

  Hit in result list → *hitTemplate*
  Example Personen / People:

```
"people": {
  "title": "ifs.category.people",
  "hitTemplate": "app/templates/default/resultlist/people.hbs",
  "showMenu": true,
  "menuItems": [
    "forward",
    "favorite"
  ]
},
```

- The people element in the **search dropdown**

  Configuration element in search dropdown → *…SWYT…*
  Example Personen / People:

```
"people": {
  "name": "people",
  "wrapperTemplate": "app/templates/default/swyt/people-wrapper.hbs",
  "itemTemplate": "app/templates/default/swyt/people.hbs",
  "display": [
    "lastname"
  ],
  "delay": 400,
  "minLength": 3,
  "searchOnClick": false,
  "type": "swyt",
  "searchByAutocompleteSuggestion": false
},
```

In our example, we want to modify the people template in the search dropdown, i.e.

1. In **config.json** search for "people".
2. Take note of the assigned item template reference.

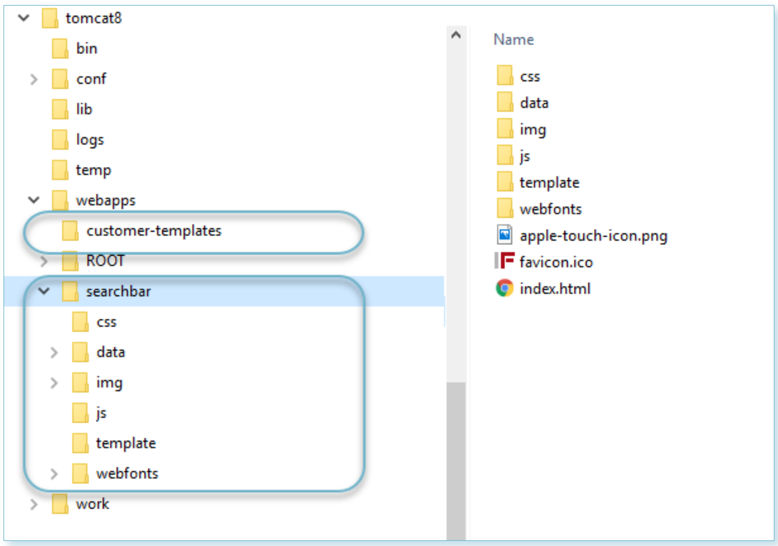
4.2.3 Registering a new template file

For registering a new template file (which will implement the changes you require) use the parameter "templatesUrl" and specify the path where the new customer-specific template resides.

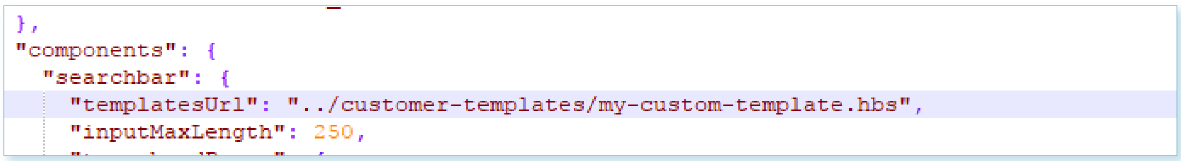
Note: Remember to reload the page each time you have modified config.json.

```
...
"components": {
    "searchbar": {
      "templatesUrl": "../path to/new/customer-template.hbs",  <-- file with new template
        ...
```
…

Note: We recommend implementing a separate template folder for customer templates on the web server *next* to the searchbar folder containing the searchbar deployment. This ensures that customized templates are not overwritten with updates of the searchbar.



If you follow this recommendation the path to the template URL is:



```
},
"components": {
  "searchbar": {
    "templatesUrl": "../customer-templates/my-custom-template.hbs",
    "inputMaxLength": 250,
```

4.2.4 Creating the required template in the template file

The next step is to create the template file. Within this file (customer-template.hbs), you require a separate **<script> tag** for each template you want to modify.

Creating a script tag with a unique script ID in customer-template.hbs:

```
<script id="id_for_my_new_people_template" type="text/x-handlebars-template">
        //CONTENT OF TEMPLATE
</script>
```

script id: Each script tag is defined with a **unique script ID** which can later be referenced in config.json.

type: The type indicates the template type and is ***mandatory***, since templates without type are ignored completely.

4.2.5 Opening sample template and copying content into the script tag

Viewing the source code of the original template file:

It is recommended to check the source code of the original template as a starting point.

The sample templates that come with the search bar installation package reside here:

app/templates

In our example, we want to change the SWYT configuration element for people and add a link to the Wiki page of the respective person.

As we determined in Step 1, the assigned value is **people.hbs** and the following path:

```
"app/templates/default/swyt/people.hbs"
```

This is the path to the original template: It has the following content:

```
<div class="ifs-person-card">
    <div class="media">
        <div class="media-left">
            <img class="media-object  ifs-avatar-img"
                 src="{{authenticateIfJWTPresent thumbnail}}"
                 data-opener="person-img"
                 onerror="this.src='{{rootDir}}/{{iconPath}}/../shared/person-
dummy.png'">
        </div>
        <div class="media-body">
            <div class="row">
                <div class="col-sm-5">
                    <h4 class="media-heading">{{ firstname }} {{ lastname }}</h4>
                    <p>{{ jobtitle }}</p>
                </div>

                <div class="col-sm-7">
                    <div class="ifs-person-info-short hidden-xs">
                        {{#if phone }}
                            <a href="tel:{{ phone }}"><i class="glyphicons
glyphicons-earphone"></i> <span data-opener="person-phone">{{ phone }}</span></a>
                        {{/if}}
                        {{#if mobile }}
                            <a href="tel:{{ mobile }}"><i class="glyphicons
glyphicons-iphone"></i> <span data-opener="person-mobile">{{ mobile }}</span></a>
                        {{/if}}
                        {{#if email }}
                            <a href="mailto:{{email}}"><i class="glyphicons
glyphicons-envelope"></i> <span>{{ email }}</span></a>
                        {{/if}}
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
```

Note: Remember that editing these templates directly will have no effect on your searchbar installation. Instead, you can use them for filling the new **customer-template.hbs** with the desired content.

Copy the code of the existing template into the script tag:

1. Open the folder where the sample templates reside and open the respective html template, in our example people.hbs
2. Select the entire code and copy it into the script tag in **customer-template.hbs** that we created in the previous chapter, i.e.

```
<script id="id_for_my_new_people_template" type="text/x-handlebars-template">
        //COPY CONTENT HERE
</script>
```

4.2.6 Adding a TEST heading to the copied content

3. Next, add an additional **TEST** heading to the copied content as shown here, in order to be able to verify quickly whether the changed template is being used.

   (See the green insertion below.)

Our **customer-template.hbs** now looks as follows.

Customer-template.hbs with TEST heading:

```
<script id="id_for_my_new_people_template" type="text/x-handlebars-template">
        <h1>TEST</h1>
        <div class="ifs-person-card">
                <div class="media">
                        <div class="media-left">
                                <img class="media-object  ifs-avatar-img"
                                        src="{{authenticateIfJWTPresent thumbnail}}"
                                        data-opener="person-img"
                                        onerror="this.src='{{rootDir}}/{{iconPath}}/../shared/person-
dummy.png'">
                        </div>
                        <div class="media-body">
                                <div class="row">
                                        <div class="col-sm-5">
                                                <h4 class="media-heading">{{ firstname }} {{ lastname
}}</h4>
                                                <p>{{ jobtitle }}</p>
                                        </div>
                                        <div class="col-sm-7">
                                            <div class="ifs-person-info-short hidden-xs">
                                                {{#if phone }}
                                                    <a href="tel:{{ phone }}"><i class="glyphicons
glyphicons-earphone"></i> <span data-opener="person-phone">{{ phone }}</span></a>
                                                {{/if}}
                                                {{#if mobile }}
                                                    <a href="tel:{{ mobile }}"><i class="glyphicons
glyphicons-iphone"></i> <span data-opener="person-mobile">{{ mobile }}</span></a>
                                                {{/if}}
                                                {{#if email }}
                                                    <a href="mailto:{{email}}"><i class="glyphicons
glyphicons-envelope"></i> <span>{{ email }}</span></a>
                                                {{/if}}
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
        </div>
</script>
```

4.2.7 Referencing the new template in config.json

In the next step, you need to modify the configuration file so that it will use/reference the newly defined template **customer-template.hbs**.

1. Go the respective position in config.json, i.e. the people category in the search dropdown section).
2. Change the **itemTemplate ID** assignment from **people.hbs** to the NEW script ID, i.e.:

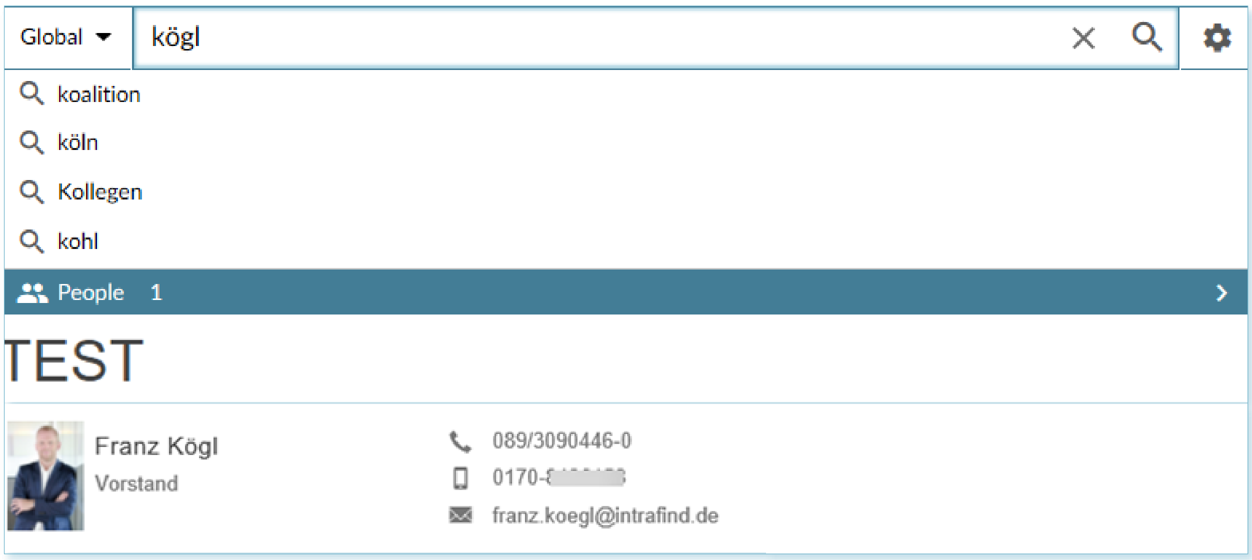
```
"people": {
  "name": "people",
  "wrapperTemplate": "app/templates/default/swyt/people-wrapper.hbs",


  "hitTemplate": id_for_my_new_people_template",
  "display": [
    "lastname"
  ],
```

Note: This is an ID assignment and not a path to a physical file. During compilation of the template files the respective template path is used to locate the templates on the file system. Once compiled this path acts as an ID to reference the compiled template.

4. Save and reload the page.

You should be able to see the updated version of the template now (with the TEST heading).


Global
kögl
koalition
köln
Kollegen
kohl
People 1

TEST

Franz Kögl
Vorstand
089/3090446-0
0170-[redacted]
franz.koegl@intrafind.de

4.2.8 Adding new index content to the template

In the above example, we simply added text to the existing template (Test heading). In our generic sample scenarios, we want to add additional content available from the iFinder index, i.e. the URL with the wiki link for the respective person.

Determining which values are available from the index:

Use the directive {{debug}} to see which content is offered by the index for the respective template. The "debug" directive always returns the actual element in the Javascript-Console:

In the new customer template file **customer-template.hbs,** position the cursor within the script tag of the new people template ID and enter debug. Go to the running instance of your searchbar installation and start a search (thereby triggering a rendering of your template):

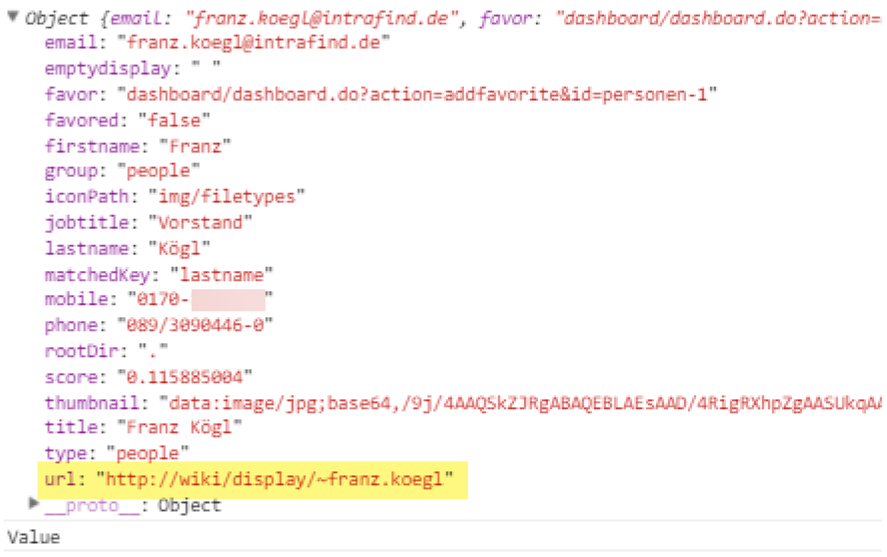
```
<script id="id_for_my_new_people_template" type="text/x-handlebars-template">
{{debug}}
<div class="ifs-person-card">
      <div class="media">
```

```
            <div class="media-left">
                  <img class="media-object  ifs-avatar-img"
                         src="{{authenticateIfJWTPresent thumbnail}}"
                         data-opener="person-img"
                         onerror="this.src='{{rootDir}}/{{iconPath}}/../shared/person-
dummy.png'">
            </div>
            <div class="media-body">
                  <div class="row">
                        <div class="col-sm-5">
                              <h4 class="media-heading">{{ firstname }} {{ lastname }}</h4>
                              <p>{{ jobtitle }}</p>
                        </div>

                        <div class="col-sm-7">
                              <div class="ifs-person-info-short hidden-xs">
                                    {{#if phone }}
                                          <a href="tel:{{ phone }}"><i class="glyphicons
glyphicons-earphone"></i> <span data-opener="person-phone">{{ phone }}</span></a>
                                    {{/if}}
                                    {{#if mobile }}
                                          <a href="tel:{{ mobile }}"><i class="glyphicons
glyphicons-iphone"></i> <span data-opener="person-mobile">{{ mobile }}</span></a>
                                    {{/if}}
                                    {{#if email }}
                                          <a href="mailto:{{email}}"><i class="glyphicons
glyphicons-envelope"></i> <span>{{ email }}</span></a>
                                    {{/if}}
                              </div>
                        </div>
                  </div>
            </div>
      </div>
</div>
</script>
```

Note: The Javascript Console must be open before rendering the template. It is also possible to output an object explicitly as follows:
{{debug someObject}}.

You can see the returned output delivered with the debug directive:



```
▼ Object {email: "franz.koegl@intrafind.de", favor: "dashboard/dashboard.do?action=
    email: "franz.koegl@intrafind.de"
    emptydisplay: " "
    favor: "dashboard/dashboard.do?action=addfavorite&id=personen-1"
    favored: "false"
    firstname: "Franz"
    group: "people"
    iconPath: "img/filetypes"
    jobtitle: "Vorstand"
    lastname: "Kögl"
    matchedKey: "lastname"
    mobile: "0170-[redacted]"
    phone: "089/3090446-0"
    rootDir: "."
    score: "0.115885004"
    thumbnail: "data:image/jpg;base64,/9j/4AAQSkZJRgABAQEBLAEsAAD/4RigRXhpZgAASUkqA
    title: "Franz Kögl"
    type: "people"
    url: "http://wiki/display/~franz.koegl"
  ▶ __proto__: Object
Value
```

The **Debug** reveals all values available with this object call. On the bottom of the list, you can see the url info to the wiki link. This is how we know that this information is available from the index and that this detail can be included into the new people template.

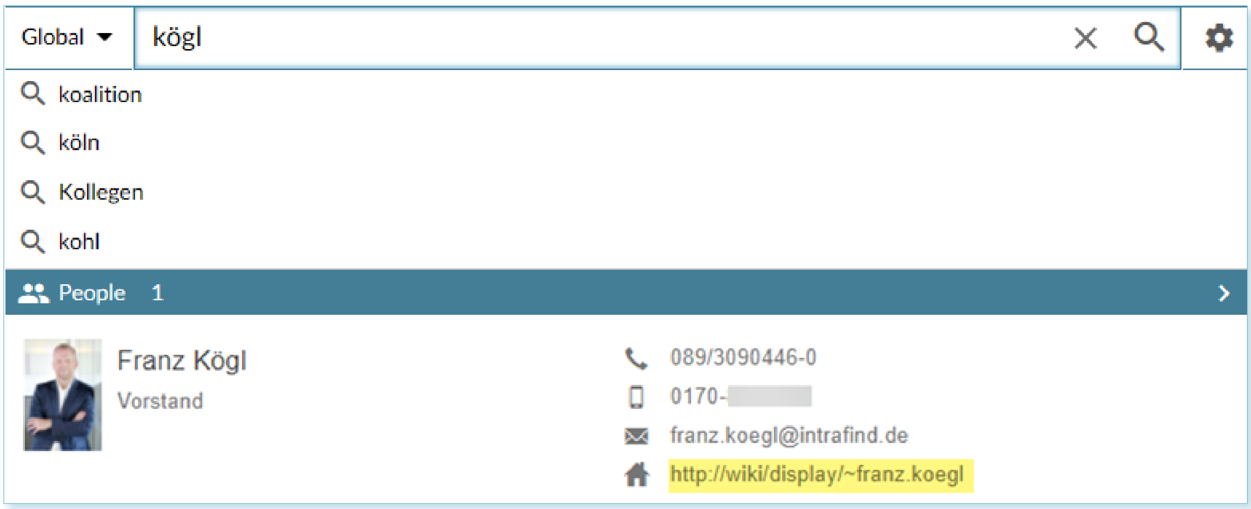
Linking the URL info into the new SWYT people template:

1. Add the URL info to the end of the template (see below).
2. Save and reload.

```
<script id="id_for_my_new_people_template" type="text/x-handlebars-template">
<div class="ifs-person-card">
      <div class="media">
            <div class="media-left">
                  <img class="media-object  ifs-avatar-img"
                         src="{{authenticateIfJWTPresent thumbnail}}"
                         data-opener="person-img"
                         onerror="this.src='{{rootDir}}/{{iconPath}}/../shared/person-
dummy.png'">
            </div>
            <div class="media-body">
                  <div class="row">
                        <div class="col-sm-5">
                              <h4 class="media-heading">{{ firstname }} {{ lastname }}</h4>
                              <p>{{ jobtitle }}</p>
                        </div>
                        <div class="col-sm-7">
                              <div class="ifs-person-info-short hidden-xs">
                                    {{#if phone }}
                                          <a href="tel:{{ phone }}"><i class="glyphicons
glyphicons-earphone"></i> <span data-opener="person-phone">{{ phone }}</span></a>
                                    {{/if}}
                                    {{#if mobile }}
                                          <a href="tel:{{ mobile }}"><i class="glyphicons
glyphicons-iphone"></i> <span data-opener="person-mobile">{{ mobile }}</span></a>
                                    {{/if}}
                                    {{#if email }}
```

```
                                          <a href="mailto:{{email}}"><i class="glyphicons
glyphicons-envelope"></i> <span>{{ email }}</span></a>
                                    {{/if}}
                                        {{#if url }}
                                        <a href=" {{url}}"><i
class="glyphicons glyphicon-home"></i> <span>{{ url
}}</span></a>
                                        {{/if}}
                              </div>
                        </div>
                  </div>
            </div>
      </div>
</div>
</script>
```

The new template now shows the URL in addition to the previous contact information.



Franz Kögl
Vorstand

089/3090446-0
0170-[redacted]
franz.koegl@intrafind.de
http://wiki/display/~franz.koegl