03/11/2020

Bosbec feature: Present voting results from Bosbec forms

English

Bosbec AB
Färögatan 33
164 51 Kista

Phone: +46 (8) 22 25 25
E-Mail: info@bosbec.se
www.bosbec.se

Table of Contents

Introduction

This document and the Live Result Snippet template allows you export voting results from Bosbec forms. One example of use scenario would be to display the voting results graphically as an extension on a web page. This tutorial will guide you through such an example where a small web application is provided which will present the results graphically of a given form.

This document consists of two main sections. Initially, the workflow which can be imported as a template from the Workflow Library and requires some minor configurations to be made operative. Second, the web application. This app is developed in Javascript with the AngularJS framework, although how the voting results is presented is arbitrary and can be done in several different ways. The only thing required is a correct API-call to Bosbec, either by HTTP-IN, or in this case REST API.

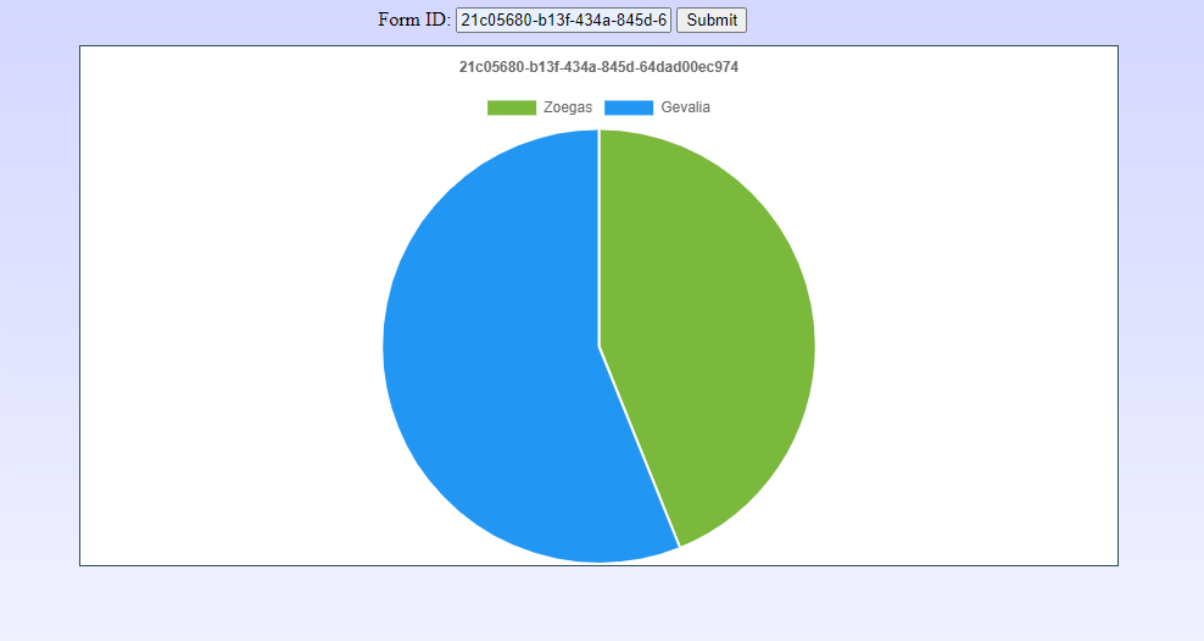

Figure 1: Web extension of a live result presentation

Requirements

- This workflow is assumed to work with a form which contains one question with two options.
- An API-token is required to request the form data from Bosbec with either HTTP-IN or REST API. To obtain an API-token, please look in the Further Reading-section for more information.

Set up the workflow

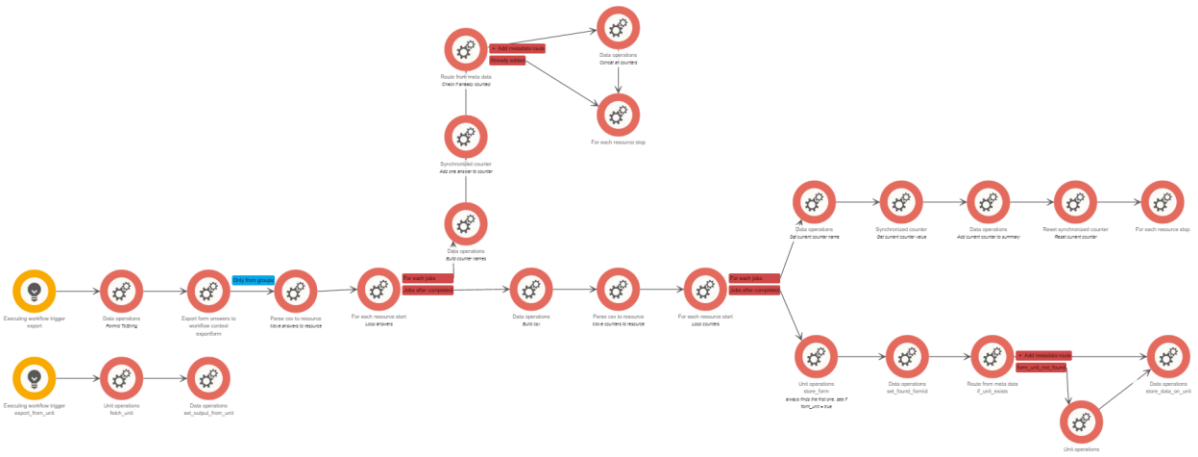

Figure 2: Live Result Snippet Workflow

One of the main reasons to use the workflow builder for your IT solutions is that it gives you the ability to easily modify the workflow as needs and requirements change. This chapter will provide some pointers to where you can make changes to the workflow.

Import the workflow

This configuration has already been built and serves as a Workflow code snippet. You only need to import the code snippet to your workflow.

1. Open “Edit” in the top left corner of the Workflow Builder.
2. Scroll down to “Workflow Library” and open it.
3. Here you will see the Workflow Library of already built code snippets and templates to streamline your work process. Expand the “Snippets”-folder.
4. Select “Live Result Snippet” to import it to the workflow canvas.

Configure the workflow snippet

Now the workflow snippet is ready be used, although to initiate the form export process some prerequisites are required.

1. If you do not have one, obtain an API-token. This can be done on the admin page of Bosbec and can be found here: https://bosbec.io/
2. The form which should be exported must have **one question, with two alternatives.** Otherwise this result presentation will not be able to parse the data correctly.
3. Activate the workflow. The workflow needs to be in an active state to be able to process incoming requests. To activate your workflow, click the toggle button in the top right corner. A pop-up should appear, and the orange triggers should turn bright orange, instead of faded.
4. Save the workflow.

Get the workflow ID

The workflow ID of your solutions is in the information side bar of the workflow. Traverse to the Workflows-page of the Bosbec Admin page. Find the workflow solution and click on it. A sidebar will appear to the right of the list, and the ID will be visible below the name of the workflow.

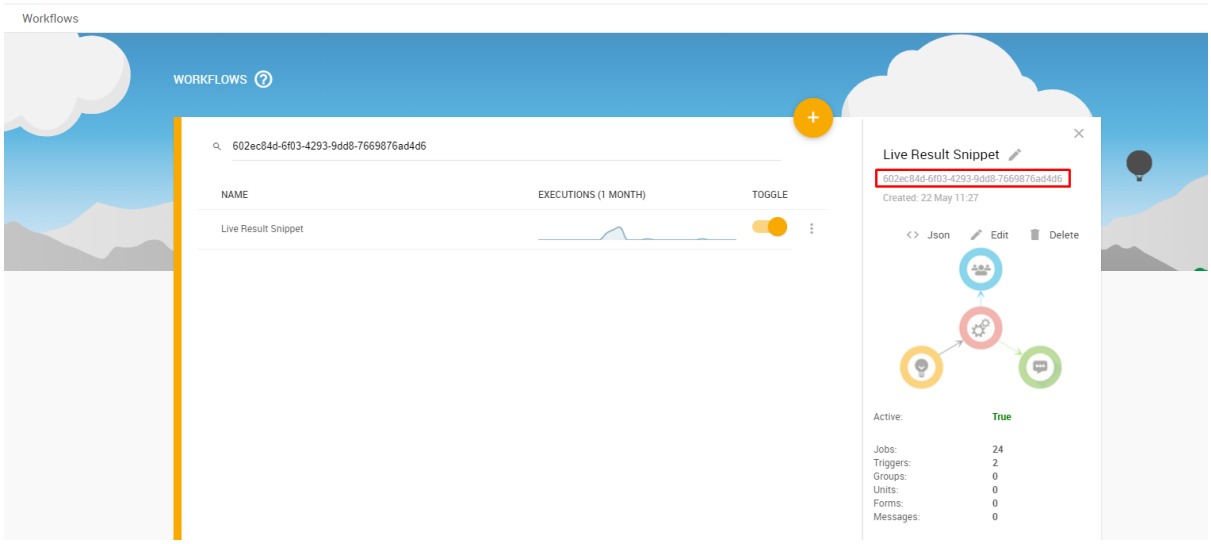

Figure 3: Workflow ID

Building the web extension

This web extension is one of minimalistic functionality, although is presented as one example of how to interact with Bosbec systems. The web application allows a user to send a form ID as an input and receive the form results as a graphical presentation. This is functionality is just one of the many uses but consists of many core concepts of interacting with Bosbec, such as API-requests and Workflow-building. The source code of the web extension in its entirety can be found in the Appendix.

The application is developed in, although not limited to, Javascript. The framework used for this application is AngularJS and the following section provides some key factors of the application, and how to request the data through API-calls to the Bosbec system.

AngularJS View

In this project, two primary element interact with the user. The form where a user inputs the form ID which will be sent in the API-request, as well as the graphical element. The graphical element is used from the Javascript library Chart.js and will plot a pie chart with the voting results.

```
<form class="form" ng-submit="submitFormId()">
    Form ID:
    <input type="text" ng-model="formId" name="formId" />
    <input type="submit" id="submit" value="Submit" />
</form>
```

Code 1: AngularJS Form to extract form ID for API-payload

```
<div class="canvas">
    <div id="form-message">{{message}}</div>
    <canvas id="pie-chart"></canvas>
</div>
```

Code 2: Chart.js canvas which will display the vote results

AngularJS Controller

The AngularJS Controller is responsible for the logic, control of data and interaction interpretation on the webpage. This controller also manages the REST-API call to the workflow and the plotting of the graph. The controller consists primarily of three components, which are listed below.

1. `$scope.submitFormId = function() {…}` : The function that executes when the form is submitted on the web page. The functionality of the webpage is embedded in this function.
2. `$http(request).then(…)` : Two REST API-requests to Bosbec workflow.
3. `New Chart(document.getElementById("pie-chart"), {…}` : The Chart.js function that will generate the graph from the API-response.

REST API-call to Bosbec Workflow

The API call to Bosbec workflow is divided into three parts. 1. The data which is sent as a payload to Bosbec, 2. The request to Bosbec REST API and 3. The request and the response handling.

1. The data which is sent as a payload to Bosbec. The code fragment below serves as a Javascript template of how to build the payload. This payload consists of the data you want to receive in the response, as well as any data you wish to send to the workflow for processing, in this case, a form ID. The data you might want to fetch from the workflow is decided in **ResponseData**. A summary of the voting result is stored in the workflow context metadata with the variable name "answer_summary". This is also where you specify which workflow that should be addressed. You need the workflow ID and the name of the trigger you wish to execute. For the Live Result Snippet, the trigger names are "export" and "export_from_unit".

```
var data = {
  workflowId: "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
  metadata: {
    formid: $scope.formId,
  },
  TriggerNames: "TRIGGER_NAME",
  RequestSettings: {
    "content-type": "application/json",
    ResponseData: {
      // Specify which data you want to get from the Bosbec platform
      answer_summary: "metadata.answer_summary"
    },
  },
};
```

Code 3: API payload to Bosbec Workflow

2. The request to Bosbec REST API needs to contain the correct address, and API-key for permission and identification, as well as the data which was set up in the previous step.

```
var request = {
  url: "https://url.com",
  method: "POST",
  headers: {
    "Content-Type": "application/json",
    "api-key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
  },
  timeout: 15 * 1000,
  data: data,
};
```

Code 3: API payload to Bosbec Workflow

3. The request and the response handling. The call can be made in different ways, and this AngularJS-application makes the request with the use of `$http`. The call will be executed, and the response-variable will contain general information about the call, but also specific data which was stated in the data-payload. Access the desired variable with the following syntax: `response.data.data.answer_summary`.

Decide which data that should be sent from Bosbec

In the Bosbec workflow builder, you can specify which data that should be transferred in an API-call. Use the job "Data Operation", and the operation "setData" to define and store values in your workflow context.

Further reading

Additional documentation and information about Bosbec integration and funcitonality can be found at https://help.bosbec.io/.

Appendix

HTML (index.html)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf 8" />
    <script src="https://code.angularjs.org/1.6.9/angular.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js@2.8.0"></script>
    <script src="app.js"></script>
    <link rel="stylesheet" href="styles.css">
    <title>Votes Presentation</title>
  </head>
  <body ng-app="app" ng-controller="chartController">
    <form class="form" ng-submit="submitFormId()">
        Form ID:
        <input type="text" ng-model="formId" name="formId" />
        <input type="submit" id="submit" value="Submit" />
    </form>
    <div class="canvas">
        <div id="form-message">{{message}}</div>
        <canvas id="pie-chart"></canvas>
    </div>
  </body>
</html>
```

CSS (style.css)

```
body {
    background: linear-gradient(rgb(212, 215, 255), white);
    background-repeat: no-repeat;
    background-attachment: fixed;
}

.canvas {
    position: relative;
    left: 25%;
    height: 75% !important;
    width: 50%;
    margin: 10px;
    background-color: white;
    border-style: solid;
    border-color: darkslategray;
    border-width: 1px;
}

.form {
    position: relative;
    left: 40%;
}

#form-message {
    position: relative;
    left: 30%;
}
```

JavaScript (app.js)

```
The angular.module("app", []).controller("chartController", [
  "$scope", "$http",
  function ($scope, $http) {
    $scope.message = "Enter your form ID to view the live results";

    $scope.submitFormId = function () {
      $scope.message = "";

      var data = {
        workflowId: "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        metadata: {
          formid: $scope.formId,
        },
        TriggerNames: "TRIGGER_NAME",
      };

      var request = {
        url: "https://url.com",
        method: "POST",
        headers: {
          "Content-Type": "application/json",
          "api-key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        },
        timeout: 15 * 1000,
        data: data,
      };

      $http(request).then(
        function (response) {
          setTimeout(function () {
            var data = {
              workflowId: "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
              metadata: {
                formid: $scope.formId,
              },
              TriggerNames: "TRIGGER_NAME",
              RequestSettings: {
                "content-type": "application/json",
                ResponseData: {
                  answer_summary: "metadata.answer_summary",
                },
              },
            };

            var request = {
              url: "https://url.com",
              method: "POST",
              headers: {
                "Content-Type": "application/json",
                "api-key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
              },
              timeout: 15 * 1000,
              data: data,
            };

...
```

```
      ...


        $http(request).then(
            function (response) {
              var answer_summary = response.data.data.answer_summary;
              var question1_summary = answer_summary.split("\n")[0];
              var question2_summary = answer_summary.split("\n")[1];

              var question1 = question1_summary.split(",")[0];
              var question1_answers = parseInt(
                question1_summary.split(",")[1]
              );

              var question2 = question2_summary.split(",")[0];
              var question2_answers = parseInt(
                question2_summary.split(",")[1]
              );

              new Chart(document.getElementById("pie-chart"), {
                type: "pie",
                data: {
                  labels: [question1, question2],
                  datasets: [
                    {
                      label: "Votes",
                      backgroundColor: ["#7AB93B", "#2196f3"],
                      data: [question1_answers, question2_answers],
                    },
                  ],
                },
                options: {
                  title: {
                    display: true,
                    text: $scope.formId,
                  },
                },
              });
            },
            function (e) {
              error(e);
              console.log("Something went wrong");
            }
          );
        }, 1000);
      },
      function (e) {
        error(e);
        console.log("Something went wrong");
      }
    );
  };
},
]);
```