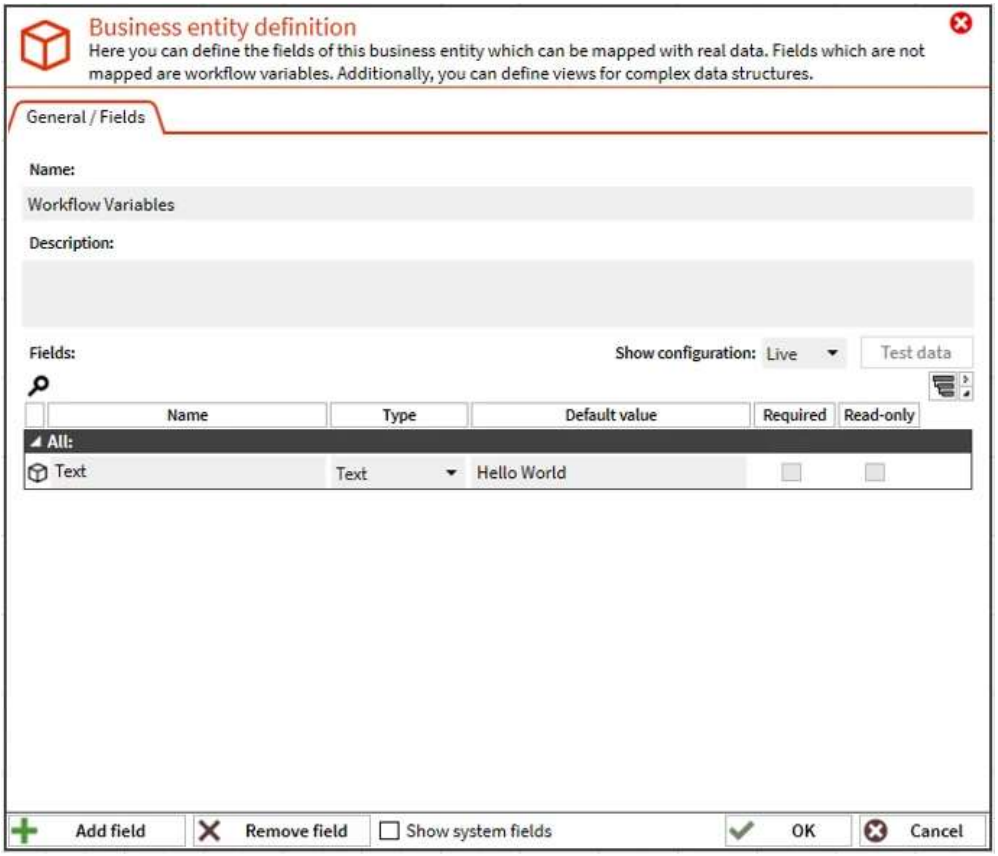
HTML Control

Dynamic Task Form

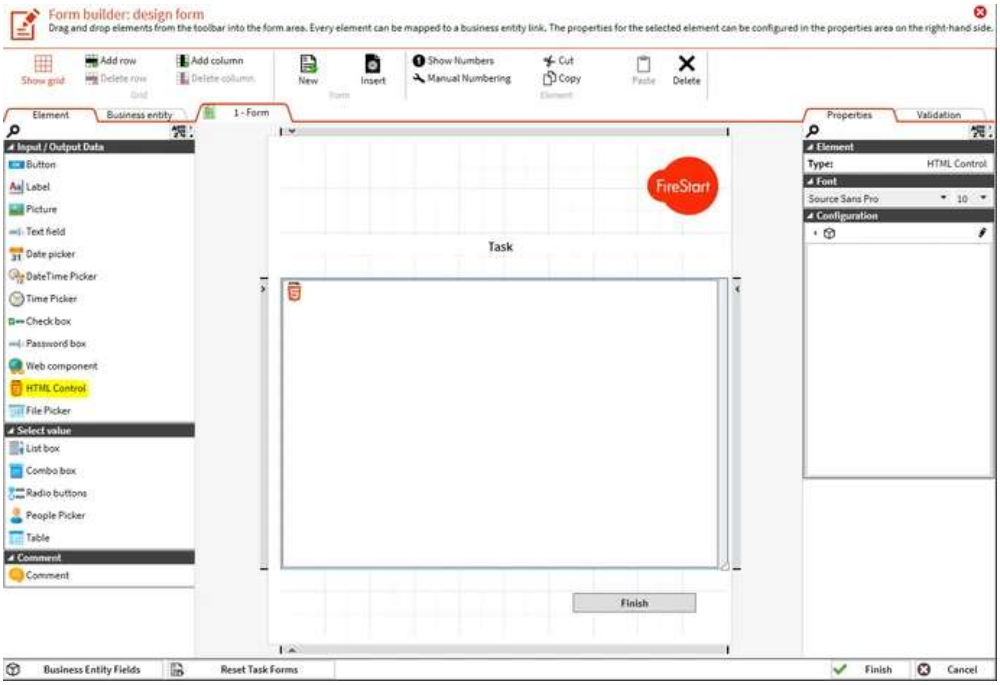
We have a new feature in FireStart which is the **HTML Control**. It provides the possibility to configure your own form content or to include external services in your form. For example, you have something like a checklist in the previous task and now you want to display all checkboxes which haven't been checked before. The **HTML Control** can be placed on the form as every other element with drag & drop. The HTML content will then be displayed where the control has been placed.

Simple HTML that uses FireStart variables

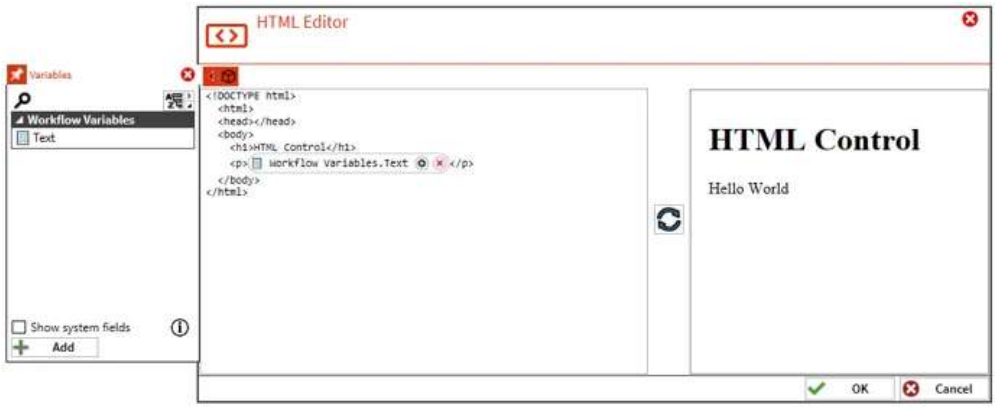
The first step is to define a Workflow Variable with a default value.



Add an **HTML Control** to your form.



After you placed it on the form you have to click it so that it is selected and then the small pen in the **Properties** tab is displayed. With clicking that pen, the **HTML Editor** opens. With the refresh button in the middle of the HTML Editor, you can preview the written content. Used variables will be replaced with their default value when previewing it. After the workflow is started, those variables will have their actual values.



Copy Code

HTML

```
<!DOCTYPE html>
<html>
  <head>
    </head>
    <body>
      <h1>
        HTML Control
      </h1>
      <p>
        {##|BELINK:00000000-0000-0000-0000-000000000000|##}
      </p>
    </body>
</html>
```

Now save and start the workflow in order to see what it looks like when executing.

HTML that uses CSS and JavaScript

To create more dynamic content you can style your HTML using CSS or program something using JavaScript. They can even use external libraries like Bootstrap or jQuery.

The example above uses external libraries via CDN:

```
HTML
```

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css"/>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"/>
```

Custom CSS rules:

CSS

```
<style>
  body{
    background-color: whitesmoke;
  }
  textarea {
    min-height:200px;
  }
</style>
```

And JavaScript code to calculate and display the length of the text in the text area:

JavaScript

```
$(document).ready(function(){
  var str = $('#text_field').val();
  $('#length_field').val(str.length);
});
```

Complete example code:

Copy Code

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css"/>
    <style>
      body{
        background-color: whitesmoke;
      }
      textarea {
        min-height:200px;
      }
    </style>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
    </script>
    <script>
      $(document).ready(function(){
        var str = $('#text_field').val();
        $('#length_field').val(str.length);
      }
                        );
    </script>
```

```
  </head>
  <body>
    <div class="container">
      <form>
        <div class="form-group">
          <label for="usr">Lorem Ipsum:</label>
          <textarea id="text_field" class="form-control" disabled>
            {##|BELINK:00000000-0000-0000-0000-000000000000|##}
          </textarea>
        </div>
        <div class="form-group">
          <label for="usr">Length of the text:</label>
          <input class="form-control" id="length_field"/>
        </div>
      </form>
    </div>
  </body>
</html>
```

Communication between the user-generated HTML Content and the FireStart form via JavaScript functions

Due to security reasons, the user-generated HTML content runs in an iFrame embedded into the FireStart form. FireStart includes a service into the user-generated content with lots of useful functions and some variables. Communication between the form and the HTML can only happen using these functions.

Variables:

- **FireStart.accessToken**
  - Gets the access token of the currently logged in user
- **FireStart.serverBaseUrl**
  - Gets the base URL of the FireStart server to call its REST API.

Functions (Parameters written in Italic are optional):

- **FireStart.getTaskData(identifier)**
  - Gets the value of a BusinessEntityLink
  - @param {string} identifier - Id of the BusinessEntityLink. The format of the variable has to be **Identifier**.
  - @returns {string} value
- **FireStart.setTaskData(identifier, value)**
  - Sets the value of a BusinessEntityLink
  - @param {string} identifier - Id of the BusinessEntityLink. The format of the variable has to be **Identifier** and the variable itself needs to be defined as **Output**.
  - @param {string} value - The value to be set, has to be in the internal format depending on the type of BusinessEntityLink, e.g. Date/Time

> **Note**: The functions get/setTaskData() operate silently on the business entity field and do NOT refresh controls, that may have the same business entity field as a data context.

- **FireStart.getControlValue(identifier,** *onlySelected, delimiter***)**
  - Gets the current value of a form control
  - @param {string} identifier - Id of the control
  - @param {boolean} onlySelected - If the control is a selection control, this flag determines if all values or only the selected values will be returned. Default is false.
  - @param {string} delimiter - Delimiter is used if the value of the control is returned as a CSV, defaults to ';'
  - @returns {string} value
- **FireStart.setControlValue(identifier, value)**
  - Sets the current value of a form control
  - @param {string} identifier - Id of the control
  - @param {string} value - Value the control should have
- **FireStart.selectControlValue(identifier, values)**
  - Selects records of a selection control by an array of IDs
  - @param {string} identifier - Id of the Control
  - @param {[]} values - Array of values that should be selected

> **Note**: The functions get/set/selectControlValue() operate on the control and return/change the currently displayed control values and also trigger a change to the business entity field if the control is mapped to one.

- **FireStart.finishTask(***successHeaderText, successDescriptionText***)**
  - Calls function to finish the task form
  - @param {string} successHeaderText - Text that appears in the message banner header in case of a successful completion of the task - optional
  - @param {string} successDescriptionText - Text that appears in the message banner description in case of a successful completion of the task - optional
- **FireStart.gotoForm(identifier)**
  - Calls function to goto the specified sub form
  - @param {string} identifier - Id or Name of the Form
- **FireStart.isTaskActive()**
  - Returns if the current task is in an active state
  - @returns {boolean} isActive