Using JavaScript on a Form and Within Calculations

Download the PDF of this article.

In this Article

Related Articles

Introduction

If you are comfortable with web programming, you can use JavaScript on your forms to extend FormAssembly's functionality.

> **Note:**
> - The examples listed below are common examples that have been created by FormAssembly users. You may need to edit these examples to fit your form's needs. **Our support team does not provide custom code and cannot help edit, write, or design custom code for your forms.**
> - While FormAssembly does not have limits on the use of this feature, extensive use in combination with other, similar features may impact performance on respondent browsers and devices. Before you begin building advanced forms, we recommend reading through this Best Practices document to learn more about our recommended planning and testing practices.

You can use JavaScript in your form to perform complex arithmetic, create pop-ups and interactive elements, add additional levels of verification, and much more!

When testing JavaScript functionality within your form, you should always do so in the live version of the form, and not in the preview mode.

Using JavaScript on your Form

Inline JavaScript can be used in either the Custom Code section of your form (under Properties → Custom Code), or in a Text element (by selecting the element and clicking on the HTML button).

If you need your code available on all of your forms, then you can add it to your Custom Branding. Your JavaScript must be included inside *<script></script>* tags.

JavaScript in Calculated Fields

You can use a single line of JavaScript in calculated fields. For example, you can use the following JavaScript in a

calculated field to shorten the length of the text being displayed by 1 character (where VARIABLE is the name of a variable defined earlier in your form):

```
VARIABLE.substring(0,VARIBLE.length-1);
```

Functions defined on your form can also be referenced inside a calculated field. This is useful when you need to reference variables defined on your form. Take this function:

```
//Do something with a variable
<script>
function getPrice(count) {
  if(count < 10) {
    } else
}
</script>
```

If you have a variable defined on your form named **numberPurchased**, you can use the formula in the calculated field like this:

```
getPrice(numberPurchased);
```

Examples

- Double-Entry Email Verification within a Form
- If/Else Statements
- Form Being Submitted without Required Fields Completed
- Removing the Pop-Up Warning when Closing a Multi-Page Form
- Renaming the Previous and Next Form Buttons

Double-Entry Email Verification within a Form

If you would like your users to confirm their email address within a form by entering it twice, this is possible. This can be useful when you want to make sure that the email address a user has entered is written correctly.

This Example Form has detailed instructions on how to set this up in your own form. Please reference this form when completing the steps below.

To begin, add a text field called "Email" and mark the field as a variable. We will use the variable name **email** for this example.

Next, add in a text field called "Confirm Email" and mark the field as a variable as well. We will use the variable name **confirmEmail** for this example.

Once you have these two variables created, add in another text field, and mark it as a hidden field under the Access Control Settings. Then, add in a default value for that field with the text you would like to have displayed if the emails that are being entered don't match. Finally, mark this field as a variable as well. We will use the variable name **msg** for this example.

Finally, add in one last text field. Mark this field as a calculated field and give it the following formula:

```
email==confirmEmail?"":msg;
```

Please note, this formula will change if you use different variables than the ones listed throughout this example.

Once you have entered the formula, you can remove the field label under the presentation options for the field.

You will also need to add a Custom Validation (under Validation Options) and add the following regular expression: /^$/

Once you have completed these steps, you're ready to test out your form. Make sure to test it out in the live version, and not the preview mode.

If the calculated field appears as a gray box, you can use CSS to change the color. Here is an example, where tfa_X is the field in question.

```
<style>
#tfa_X {
color: red !important;
background-color: white !important;
border: none !important;
}
</style>
```

If/Else Statements

You can use If/Else statements to conditionally display different text or variables within a field. For example:

```
if(variableA=="Vegetarian Option"){variableA}else{variableB};
```

In the above example, if a user selects the response "Vegetarian Option" from a single or multi-select field, then this calculated field will show what you have defined as variableA. Otherwise, it will show what you have defined as variableB.

If you would like to add an additional variable, you can do so using the following code:

```
if(variableA=="Vegetarian Option"){variableA}else{if(variableA=="Other Option"){variableB}else{variableC}}
```

Form Being Submitted without Required Fields Completed

All FormAssembly validations run on JavaScript, so if you have a form being submitted where a respondent has not completed all required fields, likely, they did not have JavaScript enabled on their browser. While there is no way to force a user to enable JavaScript, you can create an alert message that will ask users to enable JavaScript if they do not have it enabled.

To do so, enter the following code in your Custom Code section (under Properties in the Form Builder):

```
<div style="color:red">
<noscript>For full functionality of this page it is necessary to <a href="http://www.enable-javascript.com/">enable Javascript.</a></noscript>
</div>
```

This will cause a pop-up to appear at the top of the form, in red, if a user doesn't have JavaScript enabled. It will also link them to directions on how to enable JavaScript.

Removing the Pop-Up Warning when Closing a Multi-Page Form

If you would like to disable the pop-up warning that appears when a respondent is closing a multi-page form, you can add the following JavaScript to the custom code section of the form (under properties):

```
<script>
wFORMS.behaviors.paging.warnOnUnload = false;
</script>
```

Renaming the Previous and Next Form Buttons

You can use the following code within the form builder under **Properties → Custom code** in order to rename the Previous and Next form buttons. You can replace the "Next Section" and "Previous Section" text with what you would like the buttons to read. You should maintain the single quote around the text as shown below.

```
<script type="text/javascript">
wFORMS.behaviors.paging.MESSAGES =
{ CAPTION_NEXT : 'Next Section', CAPTION_PREVIOUS : 'Previous Section' }
</script>
```