Can I Use Additional Form Validation to Stop a Submission?

Last Modified on 06/24/2026 11:32 am EDT

Yes, Digioh allows you to set up form validation, in addition to the basic email syntax validation provided by default in the standard email field. This is useful, for instance, to validate that your Zip Code fields have 5 numbers, or if you use a service like BriteVerify to validate that the submitted email exists.

In the new Custom JS (Boxes), there are a few "event triggers" where you could accomplish this.

1. Before Form Validation
2. After Form Validation
3. Before Submit

These 3 "Event Triggers" are unique from the others. These 3 allow you to return different "types" to achieve different results. Specifically, if you return one of the following things, it will have a different execution result:

1) return false;

Simply halts execution, does not submit the form. So if you return false for "Before Form Validation" then the normal validation will never run, and the form will never get submitted. But if you return false "After Form Validation" then the normal validation will run, but your custom JS function will only run if it makes it past the normal validation. But if it makes it that far, since you are returning false, it will halt execution and the form will never get submitted. And finally, if you return false "Before Submit" then the normal validation will run, but the form will never get submitted.

2) return "some error message";

In this case, if you return a string, it will be displayed as an error message. This is a way to display a custom validation failure message. It works exactly like returning "false" above. But in this case, it also displays an error message to the user.

3) return anObject;

If you return an object, it will get used as the "form" data for the submission. It overrides the existing form data.

Note: The custom JavaScript functions receive 3 input parameters as follows: (api, s, x) which you can reference within your custom JS function code. The parameters "api" and "s" will always be the same for all of the custom JS "event trigger" functions. However, in the case of the unique 3 "event trigger" functions above, the "x" parameter is an object that will contain the current form data. So let's say you want to add a "password" field in "custom_1". Here is how you would do that in the custom function

"Before Submit":

```
x.custom_1 = "some_random_pw";
x.additional_field = "anything";
return x;
```

Note how "x" is modified and the modified version is returned. Since you are returning an object here, this tells the form submission process to use that object instead of the original form data object.

4) return true;

(or **return;**) This simply continues with the normal form submission flow.