University of Colorado
Boulder | Colorado Springs | Denver | Anschutz Medical Campus

OnBase by Hyland

OnBase Guide: Unity Forms - Validating Field Values with RegEx (Regular Expressions)

Goal: To ensure user-entered values are valid before form submission

Complexity Level: Departmental Unity Form Developers

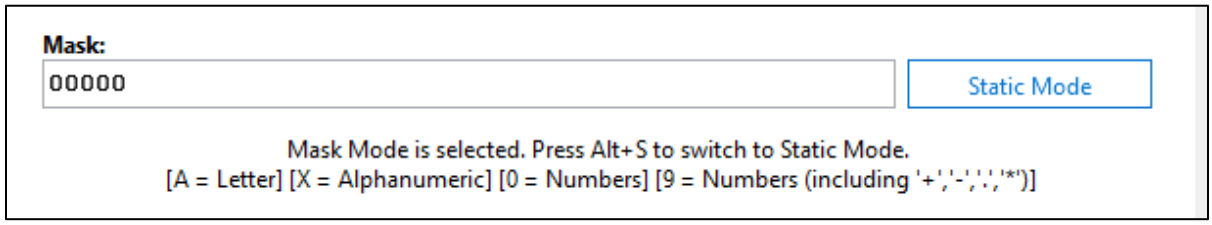
3/12/2025

Table of Contents

Background 3
Using Scripts on Unity Forms 3
Field Invalidation 4
Using the GEN - OnBase - Email Address Validation Script 5
Using the GEN - OnBase - Generalized Unity Form RegEx Search Script 8
Using the Script for Validation (Does the input match the given pattern?) 9
Using the Script to Get Output 11

Background

Unity form fields can use data sets or masking to enforce certain data validation requirements on values entered by users. If a value does not match the mask configured for a field, the form cannot be submitted until the value is corrected. For example, to ensure a 5-digit class number is entered, you could configure this mask:



For more information on masks, refer to the Unity form and System Admin MRGs.

In cases where masking or other existing custom action types are not sufficient for data validation, we have two Unity scripts available for your use:

- GEN - OnBase - Email Address Validation
- GEN - OnBase - Generalized Unity Form RegEx Search

The first can be used for email address entries, the second is a bit more complex and can be used for whatever data validation requirements you have. Both use regular expressions (regex) to provide results that are not possible with the standard set of custom actions.

In some cases, regular expressions may not be necessary and a simpler string function may fit the need better (ex. substring, contains, starts with).

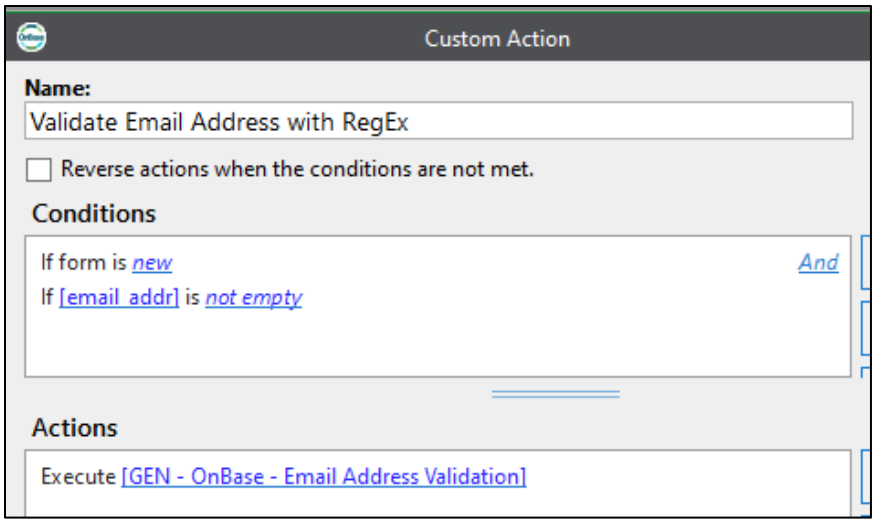
For assistance, please contact UIS_DM_Support@cu.edu.

Using Scripts on Unity Forms

In order to run a script on a form, you will need to add a custom action. For more information on custom actions, please refer to the Unity Form MRG.

More information on using scripts within Unity Form custom actions is available in the Unity Script Usage section of the admin handbook.

Your custom action should look something like this:



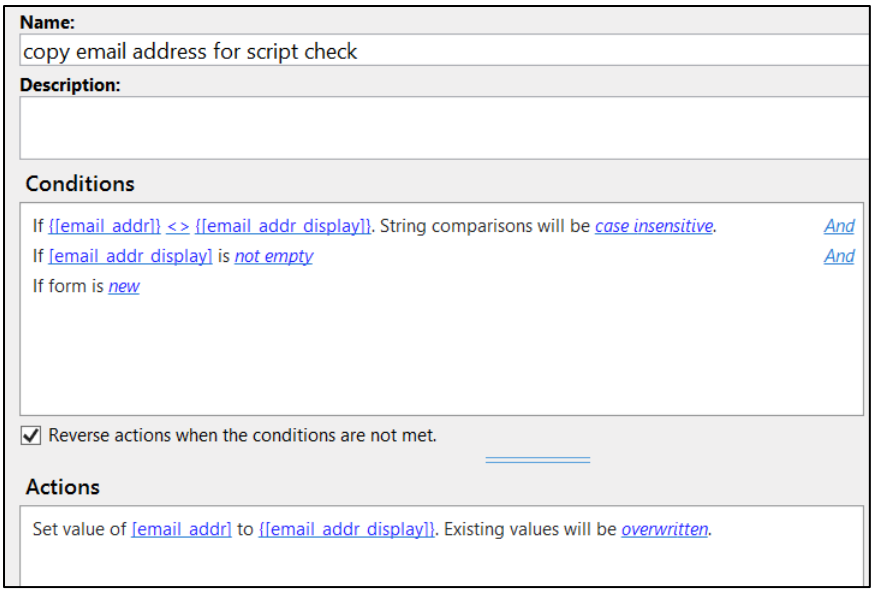
Field Invalidation

NOTE: Additional steps to copy field values to manage field invalidation are no longer required since our upgrade to 22.1. Non-repeater fields which have been marked invalid can now be read as script inputs.

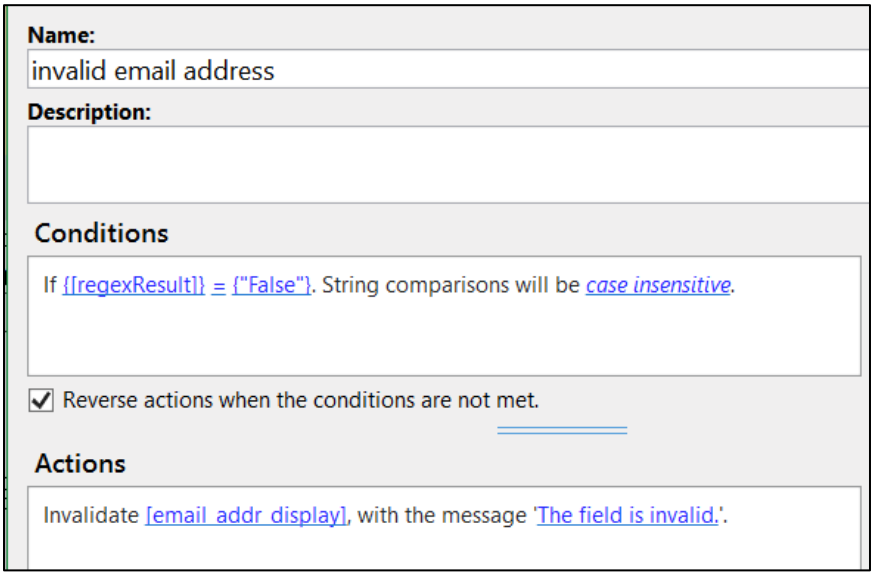
This information is preserved for reference only.

Copies of field values can still be useful for other purposes, such as determining whether the value of a field has been changed and requires re-validation.

> In prior OnBase versions, if a field value was invalidated, that value could not be used as a script input (scripts ignored values of invalidated fields). So if you marked the email_addr field invalid to prevent submission, once the user makes a correction to the value the script cannot re-process the updated value since the field is already invalid.
>
> In addition to the email_addr field which would be hidden, you'd need a field to display to users for entry. Whenever the value the user enters was changed, it would be copied to the email_addr field for the script to validate that value.



If there is a problem, the display field could be made invalid to prevent form submission.



Using the GEN - OnBase - Email Address Validation Script

This script will only validate that fields have a value which is formatted as a valid email addresses. It is recommended you use this with fields that are not populated by either of the scripts to retrieve the email address from ICS or HCM based on Student ID or Employee ID. If possible, we recommend using the email address from ICS or HCM to ensure valid inputs.

Please note this script will not confirm the address is an actual email account and can receive email, it is only checking that the entry could be a valid email address (contains valid address characters, then “@”, then a domain).

1. **The following fields will need to be configured on your form:**

- email_addr - this is the default field for the email address being validated
- regexResult – this is the default field which will be populated by the script and will be set to “True” if the value could be a valid email address or “False” if it is not.

If you want to validate other fields or additional fields, you can configure those additional email address fields as needed. Each one will need a corresponding result field.

Once your address and result fields are configured, you can specify the list of those fields with two additional input fields. Each of these list fields should have default values set. Separate field IDs with a pipe (|). The count of input fields must match the count of result/output fields.

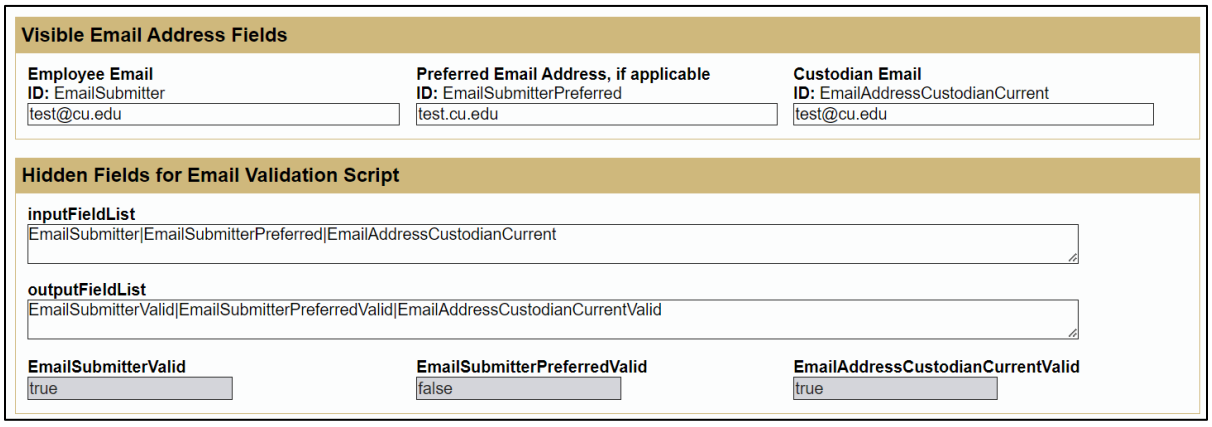
- inputFieldList – this field will contain the list of additional email address fields that need validation. Ex:

  EmailSubmitter|EmailSubmitterPreferred|EmailAddressCustodianCurrent

- outputFieldList – this field will contain the list of additional result fields where the validation result will be written. Ex:

  EmailSubmitterValid|EmailSubmitterPreferredValid|EmailAddressCustodianCurrentValid

Example of field configuration on form, using custom fields:



Additional/custom fields will be checked along with email_addr, if email_addr and regexRresult exist on the form template. Only fields with values will be checked.

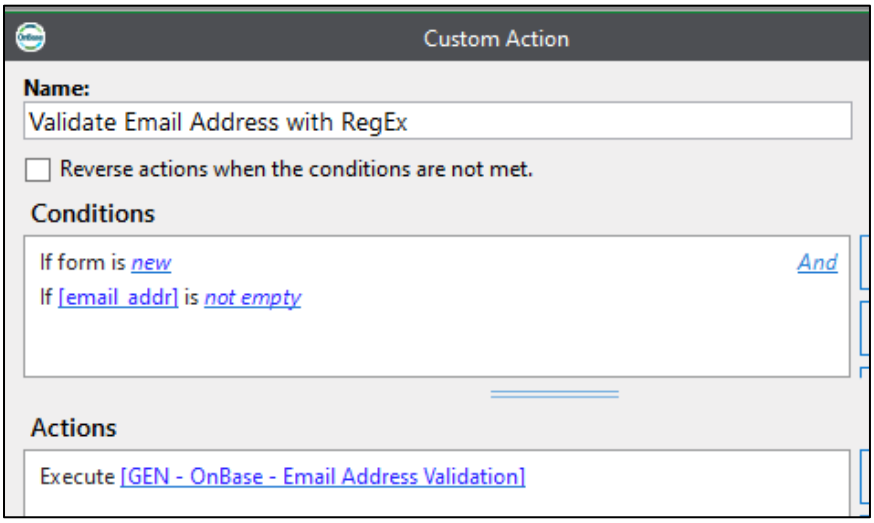
The regexResult field can be hidden if desired. If using inputFieldList and outputFieldList, it is recommended those are hidden as well.

You can also optionally include fields to display error messages returned by the script using the defined error fields:

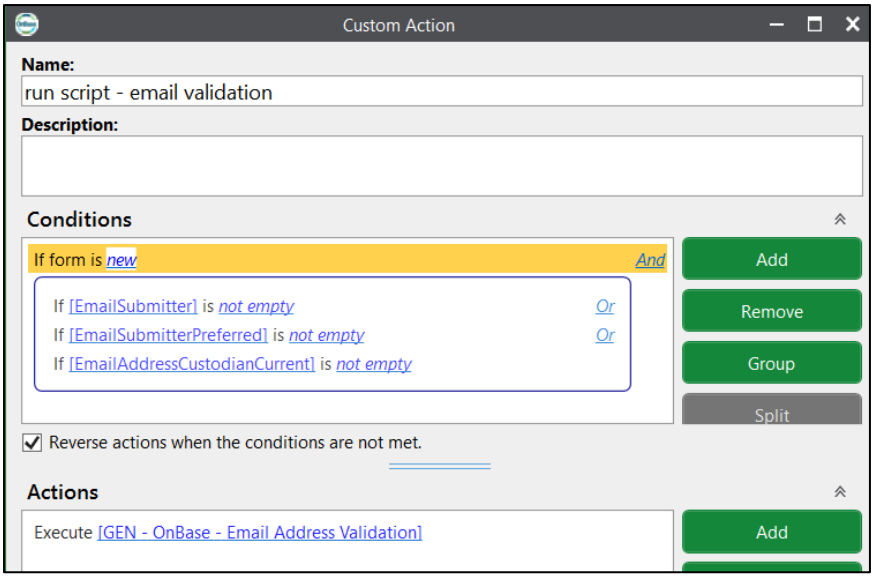
- error_message
- error_message_GENOnBaseEmailAddressValidation

2. **Configure your custom action to execute the script (as described above).**

   Example:



   Example with additional/custom validation fields:

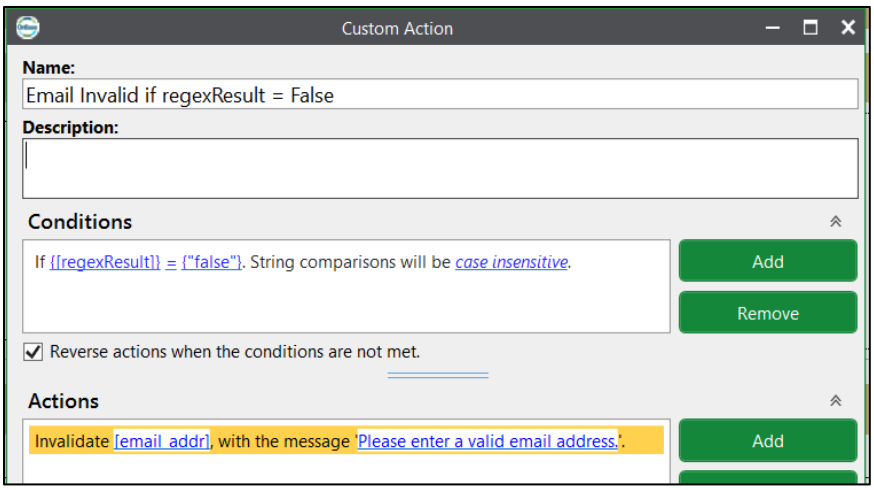


3. **Configure a custom action to handle the script result.**

The script will only return a true or false value to the regexResult field; it will not prevent submission of a form with an invalid address or take any other action. You will need to create one or more custom actions to achieve the desired result.

For example, if you want to require the user to enter a corrected value you will need to add some custom actions.

A simple option is to invalidate the field that corresponds with the result.



Using the GEN - OnBase - Generalized Unity Form RegEx Search Script

This script can be used to match any kind of user-input value against any regular expression pattern. This could include:

- Validating an email address if more than one email address exists on the form or if you want to ensure a certain type of email address is entered (ex. ___@cu.edu). The screenshots in the validation example below are based on this scenario.
- Checking a field entry using a wildcard-style comparison.
- This script can also be used to get output from a regex pattern instead of simply validating the value matches the given pattern, so it greatly enhances the existing custom action functionality.

Some of these needs may also be addressed by the String Functions Script.

This script requires additional fields on the form to tell the script what field to use and the regex pattern to use. You will need to supply the regex pattern. There are a variety of resources available for learning regex or common patterns that you can use. This list is not comprehensive but should help you get started:

- https://regex101.com/

- http://www.rexegg.com/regex-quickstart.html
- https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference
- https://emailregex.com/

Using the Script for Validation (Does the input match the given pattern?)

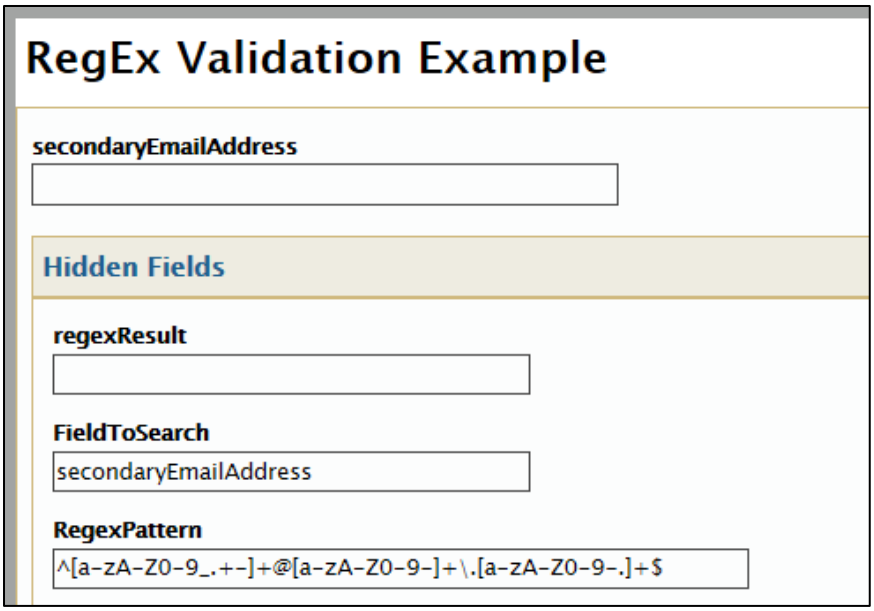
These steps are for using this script only for validation of input, not using output from the regex pattern.

1. **The following fields will need to be configured on your form:**

- *FieldToSearch* - this is what tells the script which value to validate. Set a default value on this field with the ID of the field containing the value to validate.
- *regexPattern* - this is the regex pattern used to validate the value in the field specified by *FieldToSearch*. Set a default value on this field with the pattern you want to use. Do not include quotes around the pattern.
- *regexResultGen* **or** *regexResult* - this will be populated by the script and will be set to “True” if the value matches the provided pattern or “False” if it does not.
    - If a field with the ID *regexResultGen* exists on the form template, that field will be used for the script result. If not, *regexResult* will be used.
    - This allows for both the email address validation script and this general validation script to be used on the same form without conflicts arising from using the same output field for both scripts.

These fields can all be hidden (using custom actions or secured sections) if desired. You can also optionally include fields to display error messages returned by the script.

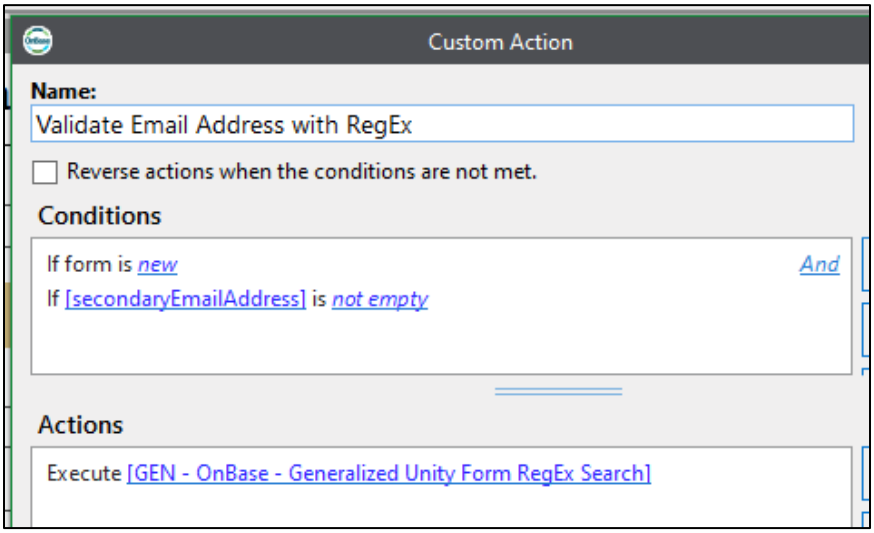
Here is an example for validating a secondary email address not filled by an ICS or HCM script (here the labels on the fields are the same as the ID):



Note: The pattern shown here is simpler than the one used in the dedicated email address validation script and therefore less accurate.

2. **Configure your custom action to execute the script (as described above).**

   Example:

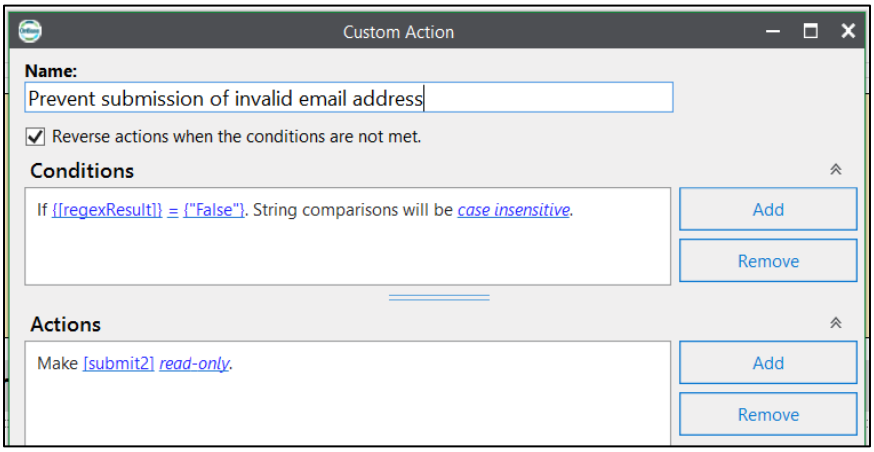


3. **Configure a custom action to handle the script result.**

   The script will only return a true or false value to the *regexResult* field, it will not prevent submission of a form with an invalid address or take any other action. You will need to create one or more custom actions to achieve the desired result.

For example, if you want to require the user to enter a corrected value you will need to add some custom actions.

A simple option is to prevent submission of the form by making the submit button hidden or read-only. This could also include showing a paragraph with an explanation. This solution has drawbacks for accessibility/usability.



If you have multiple values to use with this script, you can create additional custom actions to update the values of the *FieldToSearch* and *regexPattern* fields and re-execute the script for subsequent values.

Using the Script to Get Output

These steps are for using this script in order to return output for the portion of the value that matches the regex pattern.

In this basic example, an academic program is entered in a form field. Based on the last letter of the program, we will display a message indicating whether it is a graduate program. This should give you an idea of the possibilities for how the script could be used and how to configure it.

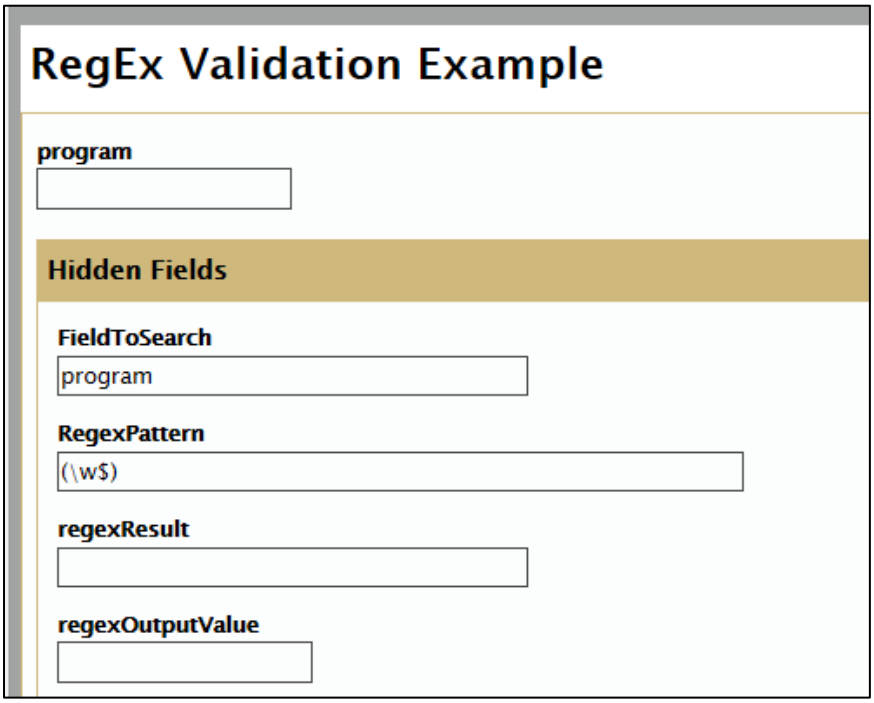
1. **The following fields will need to be configured on your form:**

- *FieldToSearch* - this is what tells the script which value to validate. Set a default value on this field with the ID of the field containing the value to validate.
- *regexPattern* - this is the regex pattern used to validate the value in the field specified by *FieldToSearch*. Set a default value on this field with the pattern you want to use. Do not include quotes around the pattern.
- *regexResultGen* **or** *regexResult* - this will be populated by the script and will be set to “True” if the value matches the provided pattern or “False” if it does not.
    - If a field with the ID *regexResultGen* exists on the form template, that field will be used for the script result. If not, *regexResult* will be used.

    - This allows for both the email address validation script and this general validation script to be used on the same form without conflicts arising from using the same output field for both scripts.
- *regexOutputValue* - this will be populated with the output from the regex pattern if the script evaluates to True.

These fields can all be hidden (using custom actions) if desired. You can also optionally include fields to display error messages returned by the script.

Here is an example to return only the last letter of the value entered in the program field (here the labels on the fields are the same as the ID):

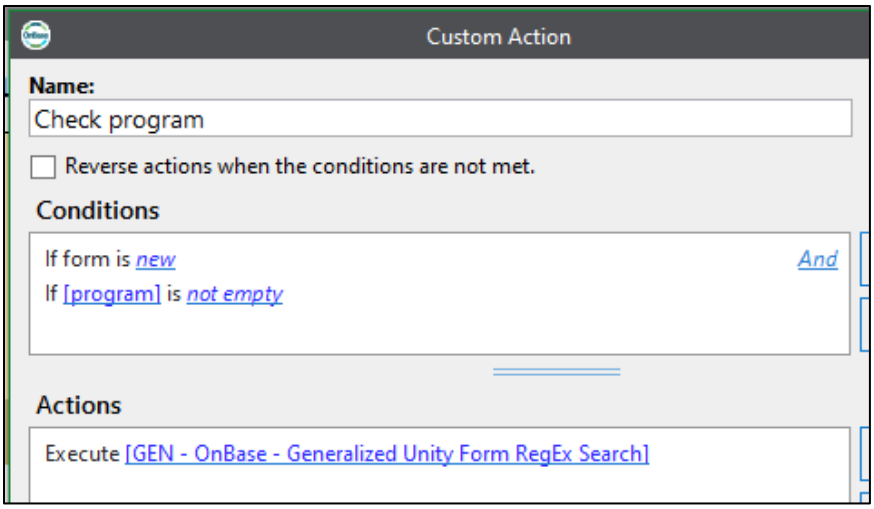


You can also optionally include fields to display error messages returned by the script using the defined error fields:

- error_message
- error_message_GENOnBaseGeneralizedUnityFormRegexSearch

2. Configure your custom action to execute the script (as described above).

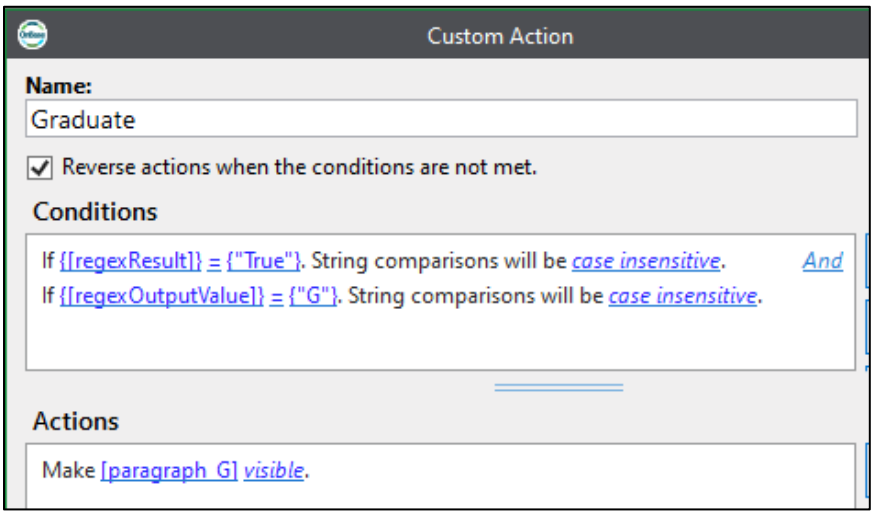
Example:

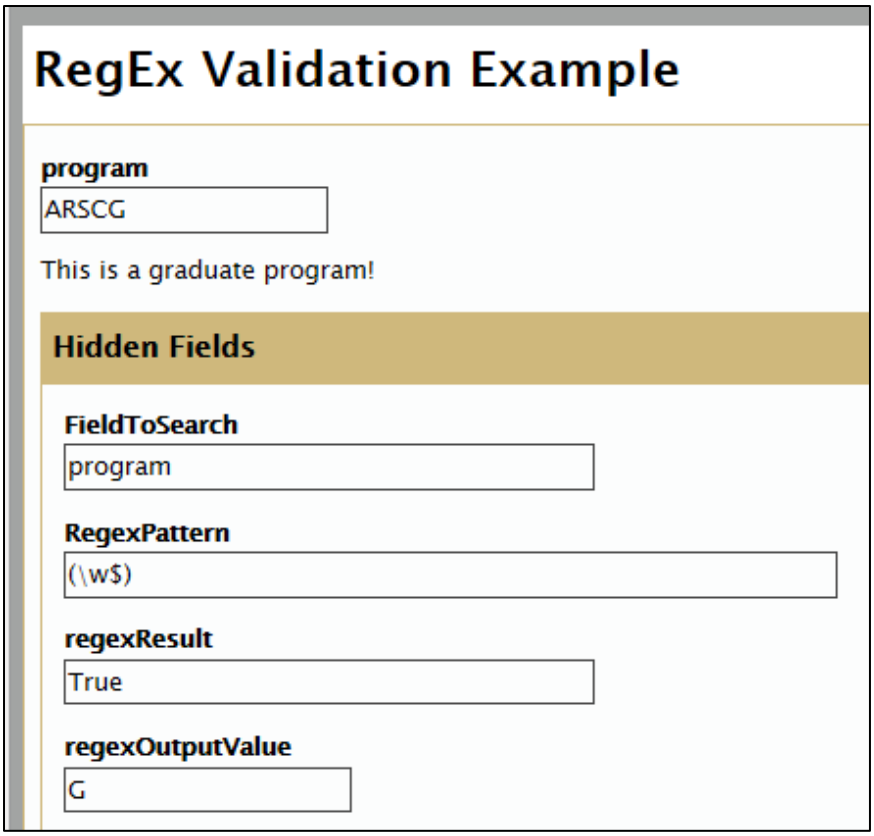


3. **Configure a custom action to handle the script result.**

   The script will only return a true or false value to the *regexResultGen/regexResult* field. Any matching string will be written to the *regexOutputResult* field. You will likely need to create one or more custom actions to achieve the desired result.

   In this example, when the last letter of the program code is “G” we display a message that the program is a graduate program.





If you have multiple values to use with this script, you could create additional custom actions to update the values of the *FieldToSearch* and *regexPattern* fields and re-execute the script for subsequent values.