
Technical Note

Field Validation Using Regular Expressions

Author(s): R&D Department

Publication date: December 8, 2006

Revision date: May 17, 2010



Field Validation Using Regular Expressions

Overview

In REP++, field values were usually validated using an input mask. However, you can now use a regular expression instead of an input mask.

This article describes how to use regular expressions to validate a REP++ field.

Adding an EMAIL field to the contact management application

You will use the contact management application to illustrate how to validate a REP++ field using a simplified regular expression.

1. Open the DEMO system in REP++*studio*.
2. Create a new field named EMAIL (to simplify the process, do not attach the field to a database column).
3. Add the newly created EMAIL field to the CLIENT group.
4. To validate the EMAIL field, you can specify a regular expression either at the field level (i.e. the validation will be applied in all groups that use this field) or at the group level (i.e. the validation will be applied only in the group where it has been specified).
 - To add the validation at the field level, go to the EMAIL field editor and set the Regular Expression text to `^\w+@[a-zA-Z_]+\.[a-zA-Z]{2,3}$`.
 - To add the validation at the group level, go to the CLIENT group editor, select the EMAIL field and set the Regular Expression text to `^\w+@[a-zA-Z_]+\.[a-zA-Z]{2,3}$`.

The regular expression is described as follows.

Method	Description
<code>^\w+</code>	Matches one or more occurrences of word characters (letters and digits) at the beginning of a line.
<code>@</code>	Matches a single instance of itself.
<code>[a-zA-Z_]+</code>	Matches one or more occurrences of any letter or underscore.
<code>\.</code>	Matches a single instance of ".".
<code>[a-zA-Z]{2,3}\$</code>	Matches two or three letters at the end.

Technical Note

The screenshot shows the 'Field editor' window for the 'EMAIL' field. The 'System*' field is set to 'DEMO', 'Section*' to '\$ROOT', and 'Field*' to 'EMAIL'. The 'Column' is selected from a dropdown. The 'Type*' is 'AL Alpha string', and the 'Control Type*' is '10 Edit box'. The 'Mask' section contains a 'Regular Expression' field with the value '^w+@[a-zA-Z_]+\.[a-zA-Z]{2,3}\$'. Other fields like 'Minimum', 'Maximum', 'Decimals', 'Output Mask', and 'CDN Command' are present but empty.

Figure 1. The EMAIL field in the Field editor.

The screenshot shows the 'CLIENT Rowset editor' with a table of fields and an 'Options' section for the 'EMAIL' field.

Field	Type	Maximum	Column
CLIENTCODE	Upper...	16	CLIENTCODE
CLIENTFIRSTNAME	Alpha ...	40	CLIENTFIRSTNAME
CLIENTLASTNAME	Alpha ...	40	CLIENTLASTNAME
EMAIL	Alpha ...	256	
CLIENTTYPE	Num s...	16	CLIENTTYPE
CLIENTSALESTODATE	Money	18	CLIENTSALESTODATE
CIICODE	Upper...	16	CIICODE
CIENAME	Alpha ...	80	CIENAME
CREATIONDATE	Date ...	19	CREATIONDATE

The 'Options' section for the 'EMAIL' field includes:

- SQL Command: [dropdown]
- Regular Expression: ^w+@[a-zA-Z_]+\.[a-zA-Z]{2,3}\$
- SQL Command for List: [dropdown]
- Output Mask: [text box]
- CDN Command: [text box]
- Alternate Name: [text box]
- Options:
 - Invisible field
 - Field not accessible
 - Default answer fixed
 - No echo
 - Answer required
 - Primary key component
 - Update in DB ignored
 - Not a DB field
 - Choice list by SQL cmd (1 time)
 - Choice list by SQL cmd (x times)
 - Execute a join at validation time
 - Execute a join at any time
 - Validate the join
 - Automatic date stamping at update
 - Automatic date stamping at insertion
 - Default answer by SQL cmd
 - Field out of rowset
 - User-defined flag
 - DBMS auto-incrementation

Figure 2. The EMAIL field options in the CLIENT Rowset editor.

5. To quickly test the behaviour of the EMAIL field, use the REP++ wizard to generate a Windows® or Web application and try entering different values for the EMAIL field.

Technical Note

The screenshot shows the 'Client Editor' interface. On the left is a table of clients. The main form contains fields for Client Code, First Name, Last Name, EMAIL, Client Type, Sales To Date, Company Code, Creation Date, and Modification Date. The EMAIL field contains 'Me@sssss' and has a red warning message: 'The value must be a valid formatted string'. An orange arrow points to this warning. Below the main form is a table of addresses.

Client Code	First Name	Last Name
BRANMICK	Mickaël	Brand
CHIRMARI		Chirac
CLAUDAN	Clauss	Dan
FERRALDO	Aldo	Ferrarini
FLEUSYLV	Sylvain	Fleury
JEANMARI	Mario	Jean
LULUCY	Luoy	Lu
STRAAAR	Aaron	Strauss
THIEANTH	Anthony	Thierry
TREMPIER	Pierre	Tremblay

Address Code	Address #1	Address #2	City	Postal Code	Province/State
8	2525 March Fields		Winnipeg	G4H4G4	Alberta
15	120 rue Latulipe		Ottawa	H3A3C8	Ontario

Figure 3. Regular expression validation at work – invalid email. A warning appears to the right of the invalid field.

The screenshot shows the 'Client Editor' interface. The EMAIL field now contains 'Me@sssss.com' and the red warning message is gone. The rest of the form and tables are the same as in Figure 3.

Client Code	First Name	Last Name
BRANMICK	Mickaël	Brand
CHIRMARI		Chirac
CLAUDAN	Clauss	Dan
FERRALDO	Aldo	Ferrarini
FLEUSYLV	Sylvain	Fleury
JEANMARI	Mario	Jean
LULUCY	Luoy	Lu
STRAAAR	Aaron	Strauss
THIEANTH	Anthony	Thierry
TREMPIER	Pierre	Tremblay

Address Code	Address #1	Address #2	City	Postal Code	Province/State
8	2525 March Fields		Winnipeg	G4H4G4	Alberta
15	120 rue Latulipe		Ottawa	H3A3C8	Ontario

Figure 4. Regular expression validation at work – valid email.

References

The simplified regular expression for email address validation was taken from the online regular expression library <http://regexlib.com/>. There are plenty of resources and tutorials on the Web devoted to regular expressions.

[http://msdn2.microsoft.com/en-us/library/2k3te2cs\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/2k3te2cs(VS.80).aspx)

<http://regexlib.com/>

<http://www.regular-expressions.info/>

<http://www.codeproject.com/dotnet/RegexTutorial.asp>