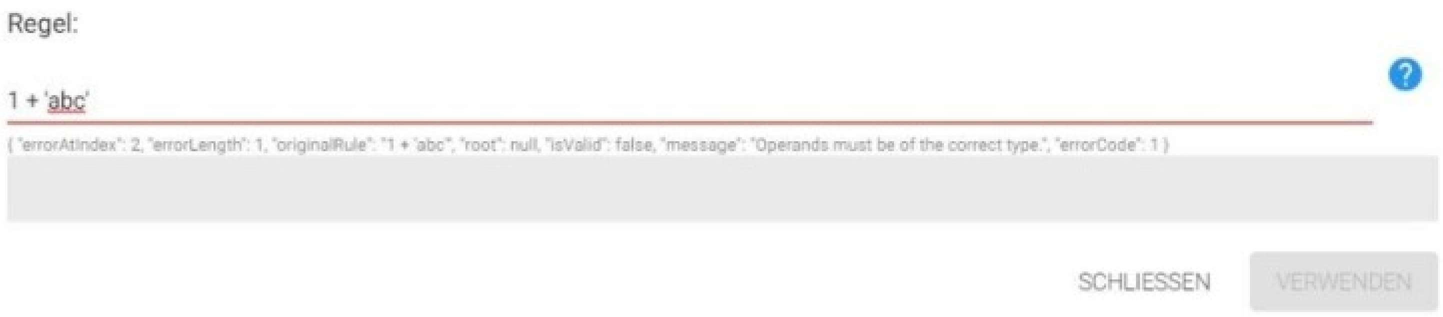
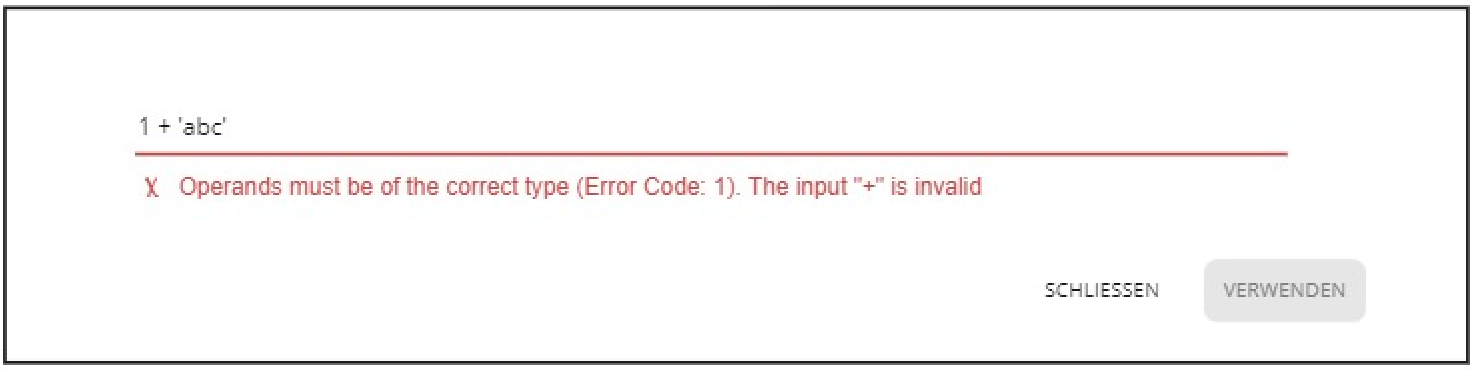


Regel:

1 + 'abc'

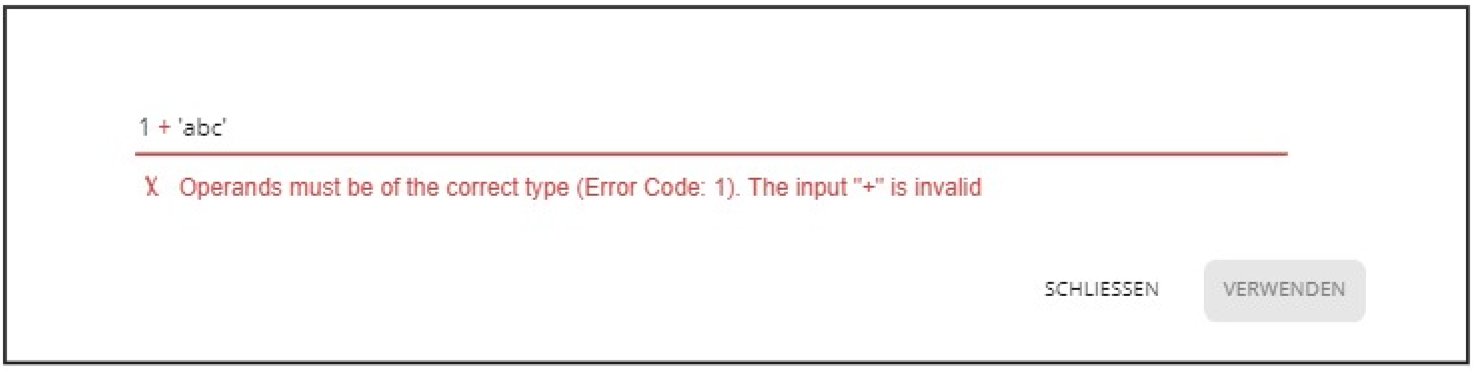
{ "errorAtIndex": 2, "errorLength": 1, "originalRule": "1 + 'abc'", "root": null, "isValid": false, "message": "Operands must be of the correct type.", "errorCode": 1 }



1 + 'abc'

χ Operands must be of the correct type (Error Code: 1). The input "+" is invalid

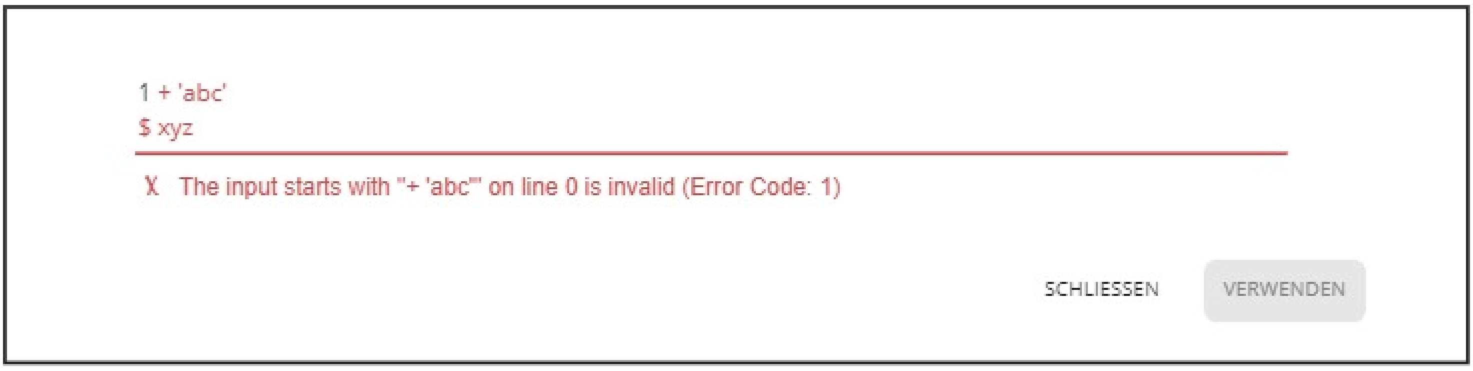
OR



1 + 'abc'

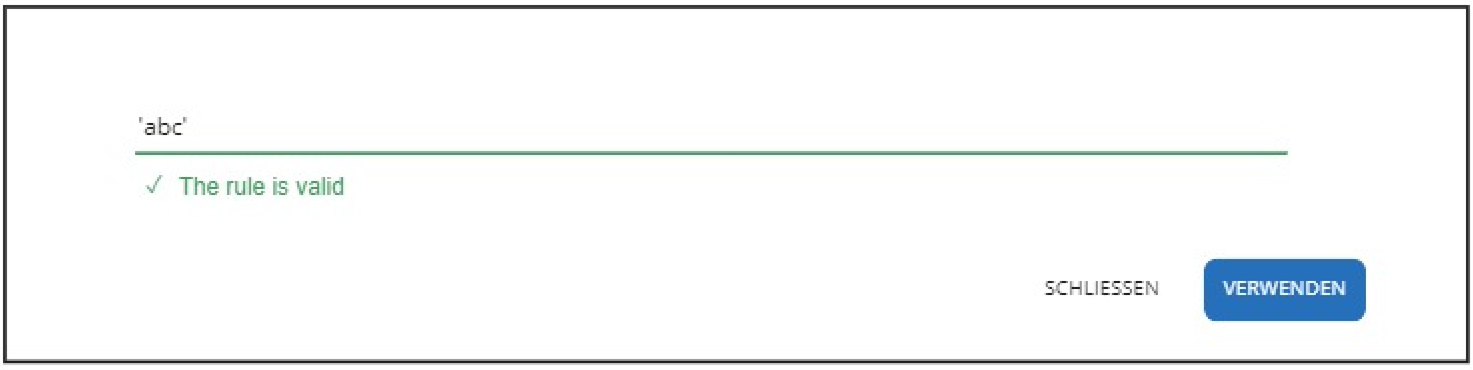
χ Operands must be of the correct type (Error Code: 1). The input "+" is invalid

Multi Line



1 + 'abc'
$ xyz

χ The input starts with "+ 'abc'" on line 0 is invalid (Error Code: 1)



'abc'

✓ The rule is valid

Notes on the UX:

- The data from backend returns the error message and also the place where the error occurs (just like in an IDE, the compiler returns where the error happens)
- Because we have the start-end points where the error occurs, we should tell the user the exact place of the error.
- If we have "\n" - newline characters, we can find the location of the errors in lines, as well. We could show this data to user for an easier location of the error.
- The highlight (red colored text) of the erroneous text would also help the user to locate the error easily.
- I think showing a green text with an approval is a good practice for consistency in the UI.
- While the user writes something to the input, a cooldown(like 0.5 seconds) should be implemented not to spam the backend server (which is also called "debouncing").