Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"

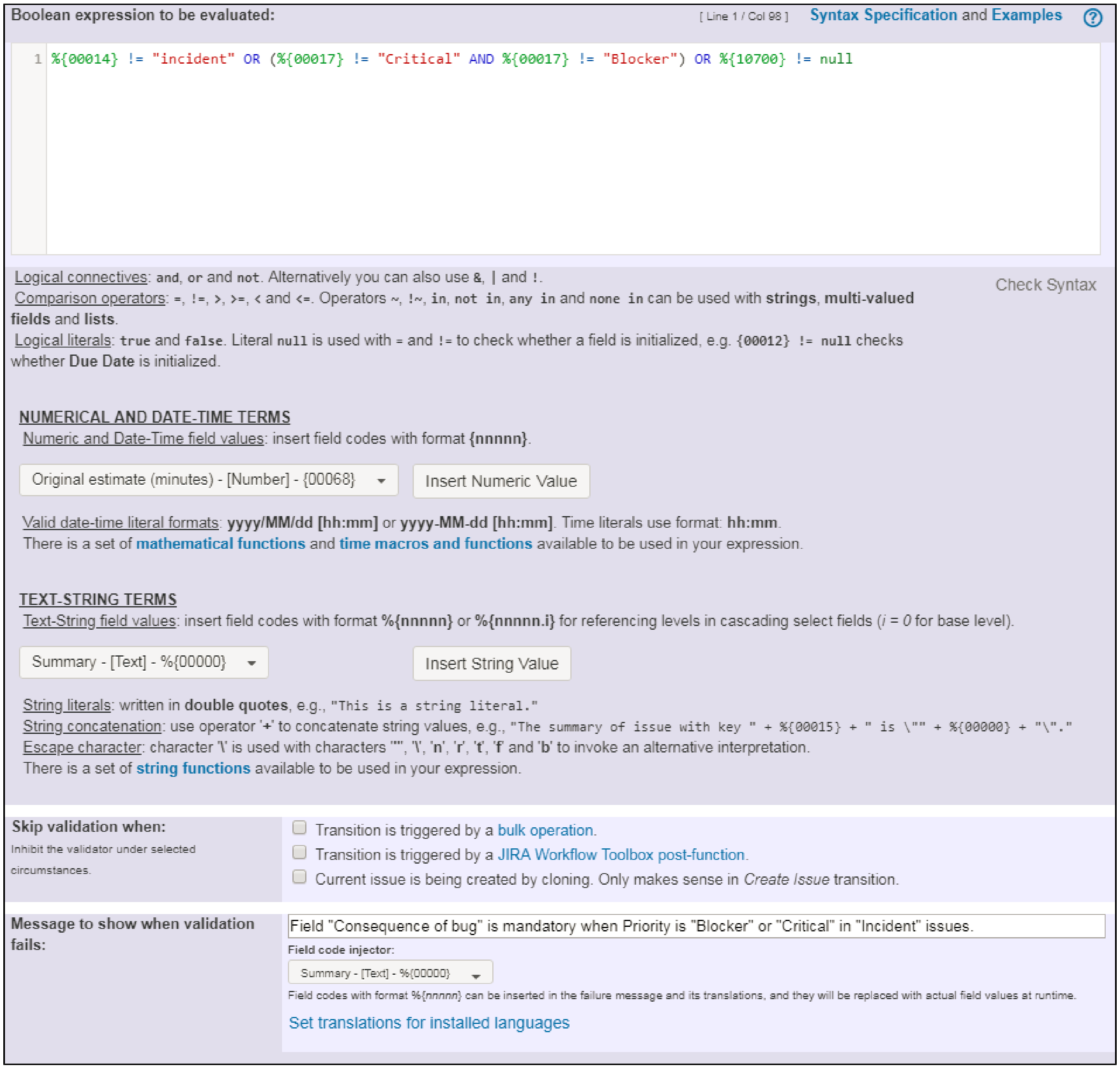
On this page

Features used to implement the example

- **Boolean validator with math, date-time or text-string terms**
- **Validation based on regular expression**

Example: Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"

We will be able to implement this requirement using **Boolean validator with math, date-time or text-string terms**. I have implemented it successfully in a beta version of the plugin using the following configuration:



Boolean expression to be evaluated: [Line 1 / Col 98] Syntax Specification and Examples

```
%{00014} != "incident" OR (%{00017} != "Critical" AND %{00017} != "Blocker") OR %{10700} != null
```

Logical connectives: `and`, `or` and `not`. Alternatively you can also use `&`, `|` and `!`.
Comparison operators: `=`, `!=`, `>`, `>=`, `<` and `<=`. Operators `~`, `!~`, `in`, `not in`, `any in` and `none in` can be used with **strings**, **multi-valued fields** and **lists**.
Logical literals: `true` and `false`. Literal `null` is used with `=` and `!=` to check whether a field is initialized, e.g. `{00012} != null` checks whether **Due Date** is initialized.

Check Syntax

NUMERICAL AND DATE-TIME TERMS
Numeric and Date-Time field values: insert field codes with format **{nnnnn}**.

Original estimate (minutes) - [Number] - {00068} | Insert Numeric Value

Valid date-time literal formats: **yyyy/MM/dd [hh:mm]** or **yyyy-MM-dd [hh:mm]**. Time literals use format: **hh:mm**.
There is a set of **mathematical functions** and **time macros and functions** available to be used in your expression.

TEXT-STRING TERMS
Text-String field values: insert field codes with format **%{nnnnn}** or **%{nnnnn.i}** for referencing levels in cascading select fields (*i = 0* for base level).

Summary - [Text] - %{00000} | Insert String Value

String literals: written in **double quotes**, e.g., `"This is a string literal."`
String concatenation: use operator '**+**' to concatenate string values, e.g., `"The summary of issue with key " + %{00015} + " is \"" + %{00000} + "\"."`
Escape character: character '****' is used with characters '**"**', '****', '**n**', '**r**', '**t**', '**f**' and '**b**' to invoke an alternative interpretation.
There is a set of **string functions** available to be used in your expression.

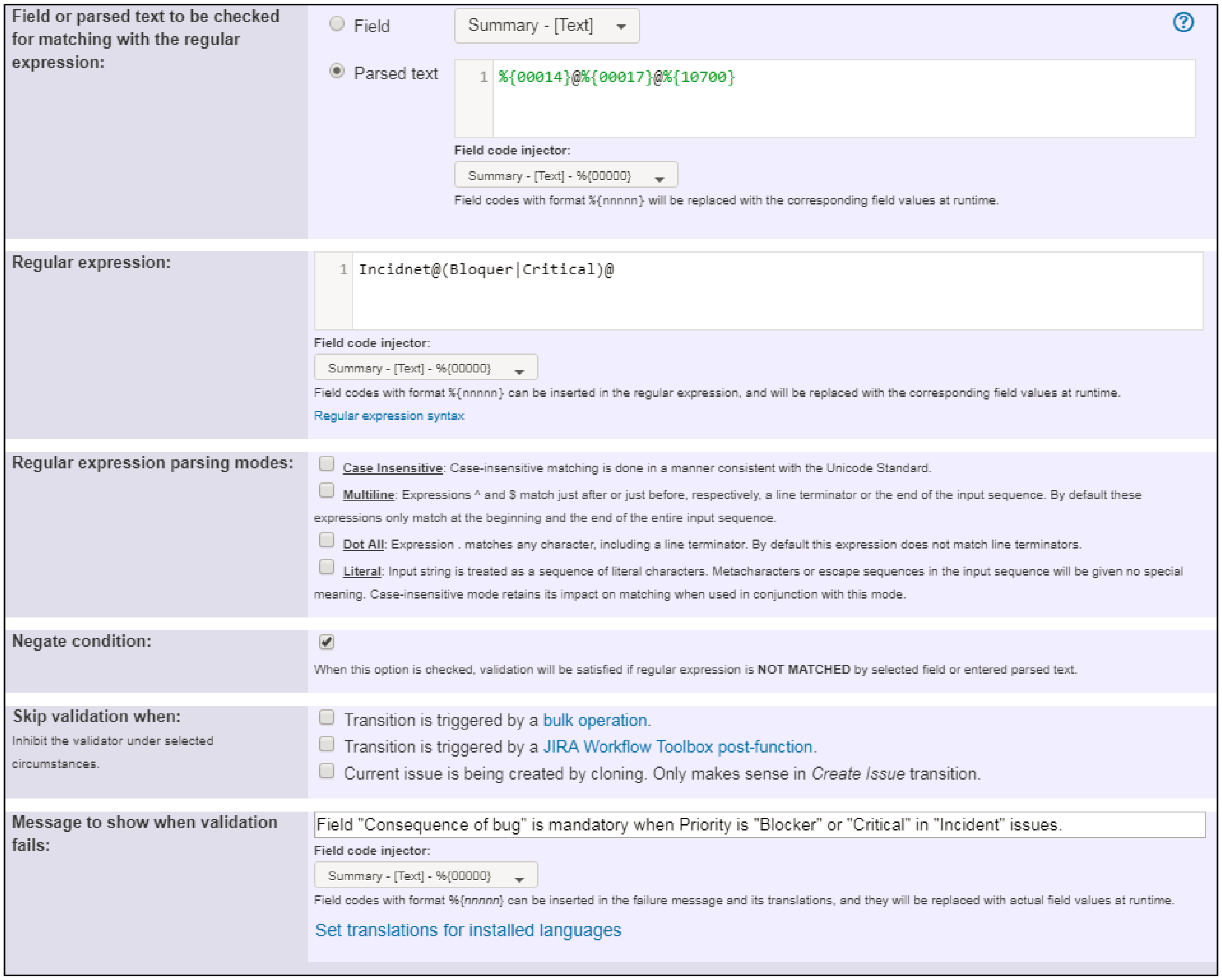
Skip validation when:
Inhibit the validator under selected circumstances.

- ☐ Transition is triggered by a bulk operation.
- ☐ Transition is triggered by a JIRA Workflow Toolbox post-function.
- ☐ Current issue is being created by cloning. Only makes sense in *Create Issue* transition.

Message to show when validation fails:
Field "Consequence of bug" is mandatory when Priority is "Blocker" or "Critical" in "Incident" issues.
Field code injector:
Summary - [Text] - %{00000}
Field codes with format %{nnnnn} can be inserted in the failure message and its translations, and they will be replaced with actual field values at runtime.
Set translations for installed languages

Text to be parsd is: **`%{00014} != "incident" OR (%{00017} != "Critical" AND %{00017} != "Blocker") OR %{10700} != null`**

Or to implement it we can use validator **Validation based on regular expression** with the configuration shown in the following screenshot:



Field or parsed text to be checked for matching with the regular expression:	( ) Field Summary - [Text] [✓ Parsed text] `%{00014}@%{00017}@%{10700}` Field code injector: Summary - [Text] - %{00000} Field codes with format %{nnnnn} will be replaced with the corresponding field values at runtime.
Regular expression:	`Incidnet@(Bloquer|Critical)@` Field code injector: Summary - [Text] - %{00000} Field codes with format %{nnnnn} can be inserted in the regular expression, and will be replaced with the corresponding field values at runtime. Regular expression syntax
Regular expression parsing modes:	☐ Case Insensitive: Case-insensitive matching is done in a manner consistent with the Unicode Standard. ☐ Multiline: Expressions ^ and $ match just after or just before, respectively, a line terminator or the end of the input sequence. By default these expressions only match at the beginning and the end of the entire input sequence. ☐ Dot All: Expression . matches any character, including a line terminator. By default this expression does not match line terminators. ☐ Literal: Input string is treated as a sequence of literal characters. Metacharacters or escape sequences in the input sequence will be given no special meaning. Case-insensitive mode retains its impact on matching when used in conjunction with this mode.
Negate condition:	✓ When this option is checked, validation will be satisfied if regular expression is **NOT MATCHED** by selected field or entered parsed text.
Skip validation when: Inhibit the validator under selected circumstances.	☐ Transition is triggered by a bulk operation. ☐ Transition is triggered by a JIRA Workflow Toolbox post-function. ☐ Current issue is being created by cloning. Only makes sense in *Create Issue* transition.
Message to show when validation fails:	Field "Consequence of bug" is mandatory when Priority is "Blocker" or "Critical" in "Incident" issues. Field code injector: Summary - [Text] - %{00000} Field codes with format %{nnnnn} can be inserted in the failure message and its translations, and they will be replaced with actual field values at runtime. Set translations for installed languages

Text to be parsed is:

```
%{00014}@%{00017}@%{10700}

Incidnet@(Bloquer|Critical)@
```

Note that:

- To evaluate the value of fields "**Priority"** (field code **%{00017}**) and "**Consequence of bug"** (field code **%{10700}**) at the same time we compose a text with both fields using character '@' as separator
- We check **"Negate validation**" since the regular expression introduced describes the kind of input we do not admit

---

Other examples of that functions

Boolean validator with math, date-time or text-string terms

Page: Block a transition until all sub-tasks have certains fields populated
Page: Block an epic's transition depending on linked issues status and due date
Page: Block or hide a transition for an issue depending on its issue links
Page: Block or unblock a transition after an issue rested a specific time in a status
Page: Block transition until all sub-tasks are in a specific status category

Related Usage Examples

- Validation on the value of a Cascading Select field
  - example
  - validator
  - custom-field
- Make different fields mandatory depending on the value of a Select List custom field
  - example
  - validator
  - custom-field

Page: Close parent issue when all sub-tasks are closed
Page: Enforce a field (Select List) to be set when another field (Radio Button) has a certain value (works with any kind of field type)
Page: Ensure that all issues linked with a certain issue link type have "Due Date" field set
Page: If field A is populated then, field B must also be populated
Page: Limit issue creation per role and issue type
Page: Limit the number of hours a user can log per day
Page: Limit valid dates for work logs
Page: Make "Time Spent" field required when there is no time logged in the issue
Page: Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"
Page: Make attachment mandatory depending on the value of certain custom field
Page: Make different fields mandatory depending on the value of a Select List custom field
Page: Make linked issues, sub-tasks and JQL selected issues progress through its workflows
Page: Make parent issue progress through its workflow
Page: Prevent issue creation if another issue with same field value already exists
Page: Reject duplicated file names in attachments
Page: Require at least one sub-task in status "Resolved" or "Closed" when "Testing required" is selected in Check-Box custom field
Page: Require issue link when resolving as duplicate
Page: Restrict parent issue from closing if it has sub-tasks that were created during a given parent issue status
Page: Restrict sub-task type creation depending on parent issue status
Page: Restrict sub-task type creation depending on parent issue type
Page: Set a condition in a global transition which only applies in a certain status
Page: Validate a custom field "Story Points" has been given a value in Fibonacci sequence
Page: Validate compatible values selection among dependent custom fields
Page: Validate only issue links created in transition screen
Page: Validate that multi-user picker custom field A does not contain any user in multi-user picker custom field B
Page: Validation and condition based on time expressions
Page: Validation based on the value of a date type project property
Page: Validation on issue attachments
Page: Validation on MIME types of issue attachments
Page: Validation on sibling sub-tasks depending on issue type and status
Page: Validation on the value of a Cascading Select field

Validation based on regular expression

Page: Make "Affects Version/s" mandatory when issue resolution is "Fixed"
Page: Make a custom field mandatory when Priority is "Critical" or "Blocker"
Page: Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"
Page: Make certain custom field required in resolve screen only if the resolution was set to "Fixed"
Page: Prevent addition of new sub-tasks if the parent issue is in "Resolved" or "Closed" status
Page: Validation on issue attachments
Page: Validation on the value of a Cascading Select field

- Validate compatible values selection among dependent custom fields
  - example
  - validator
  - custom-field
- Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"
  - example
  - validator
  - custom-field
- Validate a custom field "Story Points" has been given a value in Fibonacci sequence
  - example
  - validator
  - custom-field
- Validate that multi-user picker custom field A does not contain any user in multi-user picker custom field B
  - example
  - validator
  - custom-field
- Make attachment mandatory depending on the value of certain custom field
  - example
  - validator
  - custom-field
- Enforce a field (Select List) to be set when another field (Radio Button) has a certain value (works with any kind of field type)
  - example
  - validator
  - custom-field
- Create a dynamic set of sub-tasks based on checkbox selection with unique summaries
  - example
  - post-function
  - custom-field
  - sub-task
- Total of all story points in an epic
  - example
  - custom-field
  - calculated-field
- Show timeliness of an issue based on two date pickers
  - example
  - custom-field
  - calculated-field
- Add and remove a single or a set of items from multi valued fields
  - example
  - post-function
  - custom-field
  - issue-links
  - sub-task
- Highest value of a custom field among linked issues
  - example
  - custom-field
  - calculated-field
- Google Maps location from address
  - example
  - calculated-field
  - custom-field
- Make certain custom field required in resolve screen only if the resolution was set to "Fixed"
  - example
  - validator
  - custom-field