

TestRail 参照/欠陥プラグインでフィールドのラベル表記がされない不具合を回避するUIスクリプト

• Date : 2025-02-12

参照/欠陥プラグインにおいて、フィールドのラベル表記がされない不具合について、ご迷惑をおかけして申し訳ありません。2025年2月時点では8.1以降のバージョンにて該当の不具合が修正される見込みです。それまでの間、本UIスクリプトにて不具合の回避をお願いいたします。

概要

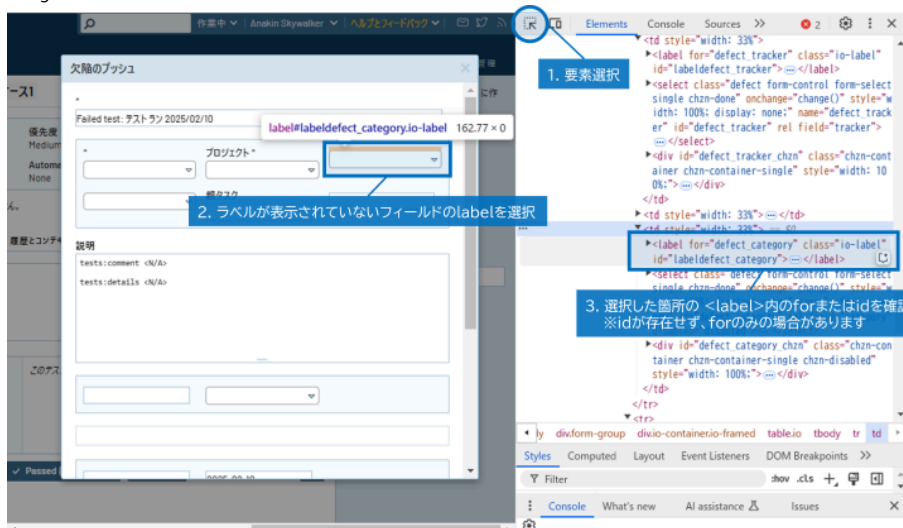
- TestRail 8.0.6.1029で確認された「欠陥プラグインでカスタムフィールド名が表示されない」不具合を回避するためのUIスクリプトです。
- このUIスクリプトは、8.0.6にて動作を確認しています。
- UIスクリプトを適用すると、以下のようにプッシュダイアログ上に正しくラベルが表示されるようになります。

The image shows two versions of the 'Push Defect' dialog in TestRail. The left version is the original, where custom fields are labeled with generic names like 'カスタムフィールド: テキスト' (Custom Field: Text). The right version is the modified version after applying the UI script, where the labels are correctly updated to reflect the field type, such as 'カスタムフィールド: テキスト' (Custom Field: Text), 'カスタムフィールド: リスト' (Custom Field: List), 'カスタムフィールド: 整数' (Custom Field: Integer), 'カスタムフィールド: 日付' (Custom Field: Date), 'カスタムフィールド: 長いテキスト' (Custom Field: Long Text), 'カスタムフィールド: 真偽値' (Custom Field: Boolean), 'カスタムフィールド: ユーザー' (Custom Field: User), and 'カスタムフィールド: キーバリュー' (Custom Field: Key-Value).

設定手順

1. ラベルが表示されていないフィールドの確認

1. TestRailにログインし、参照または欠陥のPushダイアログを確認します。
 - 連携されている外部システムごとに確認する必要があります。
2. 欠陥のプッシュダイアログ上で、ラベルが表示されていないフィールドがある場合、Webブラウザの開発者ツールでHTML要素のidまたはforの値を確認します。
 - Google Chromeの例:



※ idとforの両方が存在する場合はどちらでも構いません。

- 以下のように、ラベルが表示されていない<label>のidまたはforをテキストエディタ等にメモします。

例:

```
labeldefect_tracker
labeldefect_category
defect_subject
```

2. UIスクリプトの設定

1. TestRailに管理者権限を持つアカウントでログインします。
2. TestRailの [管理] > [カスタマイズ] > UIスクリプト タブから、"+ UIスクリプトの追加" ボタンを押下します。
3. このReadmeの末尾に記載されているUIスクリプトをテキストエリアに貼り付けます。
4. UIスクリプト内、ラベルのidまたはforの値を指定する実装箇所を修正します。
 - 指定方法は **"idまたはforの値" : "表示する文字列"** です。
 - スクリプトには、Redmine, Jira, Backlogの例を記載しています。
 - お使いの環境や設定によりidやforの値は異なりますので、ご自身で確認した結果を指定してください。
 - RedmineとJIRAなど2つ以上のツールと連携している場合でも、UIスクリプトに記載されている設定例のようにまとめてidやforを指定できます。
5. (必要に応じて) UIスクリプトの対象とする画面を設定します。
 - 本UIスクリプトでは、テストランとテストケースの一覧画面が表示されている場合に処理が実行されます。
`includes: ^runs/view|^suites/view`
 - 他の画面にて欠陥のプッシュを行う場合は、includesに画面のURLを追加します。(例:|^projects/overviewを追加する)
6. "UIスクリプトの保存" ボタンを押下します。

```
/**
 * ★各 mapping 用の設定 (id属性またはfor属性に設定された値に対応)
 * [指定の形式] "idまたはforの属性値" : "表示するラベル文字列"
 */
var labelTextMapping = {
  // Redmine (<label id="...">)
  "labeldefect_tracker": "トラッカー",
  "labeldefect_category": "カテゴリ",
  "labeldefect_assigned_to": "担当者",
  "labeldefect_estimated_hours": "予定工数",
  "labeldefect_custom_field_string_1": "カスタムフィールド: テキスト",
  "labeldefect_custom_field_list_2": "カスタムフィールド: リスト",
  "labeldefect_custom_field_string_4": "カスタムフィールド: 整数",
  "labeldefect_custom_field_date_5": "カスタムフィールド: 日付",
  // Redmine (<label for="...">)
  "defect_subject": "題名",
  "defect_custom_field_text_6": "カスタムフィールド: 長いテキスト",
  "defect_custom_field_listmulti_3": "カスタムフィールド: リスト (複数選択)",
  "defect_custom_field_bool_7": "カスタムフィールド: 真偽値",
  "defect_custom_field_user_8": "カスタムフィールド: ユーザー",
  "defect_custom_field_keyvalue_9": "カスタムフィールド: キーバリュー",
  "defect_custom_field_keyvaluemulti_11": "カスタムフィールド: キーバリュー (複数選択)",
  // JIRA (<label for="...">)
  "defect_customfield_10435": "カスタム日付",
  "defect_customfield_10436": "カスタムユーザー",
  // Backlog (<label for="...">)
  "defect_issue_type": "種別",
  "defect_category": "カテゴリー",
  "defect_milestone": "マイルストーン",
  "defect_version": "発生バージョン"
};
```

期待通りに動作しない場合

- Webブラウザの開発者ツールを起動し、参照や欠陥のPushの操作を行った際のコンソールのログをご確認ください。
- 動作しない場合によくある原因は以下の2つです。
 - UIスクリプトの実行に失敗していないか (javascriptに文法エラーがあると動作しません)
 - 指定しているidやforの値に間違いがないか
- なお、本UIスクリプトにはDEBUG用のコンソールログが実装されていますので、必要に応じてDEBUGモードをONにして動作を確認ください。 `var DEBUG_MODE = 1;` と変更することでコンソールログが出力されるようになります。

```
// DEBUG_MODE: 1 ならログ出力、0 ならログ出力しない
var DEBUG_MODE = 0;
function debugLog() {
  if (DEBUG_MODE) {
    console.log.apply(console, arguments);
  }
}
```

注意事項

動作上の懸念事項

- ポーリング（pollAndUpdateLabels）の条件・注意点
 - setInterval によるポーリングは1秒間隔、最大10回（＝約10秒）で実行されています。
 - ページやモーダルを読み込みが環境依存で10秒以上かかる場合、更新対象のラベルがその期間内に更新されず、結果として空のままになる可能性があります。
- 更新判定の条件
 - ラベルの更新状態は、ラベル内のテキストノードが空であるかどうかで判断しています。
 - 一時的なプレースホルダーや他スクリプトによって一瞬でも空文字になった場合、再更新を試みるため、意図しないタイミングで更新処理が走る可能性があります。
- 対象となるタイミングの依存
 - pollAndUpdateLabels の起動は preparePush 関数のオーバーライドを通じて行われています。
 - もし preparePush が呼ばれないシナリオや、呼び出しのタイミングが想定外の場合、ポーリングが発生せず、ラベル更新が実施されないリスクがあります。

動作しない／期待通りに動かない可能性のあるケース

- DOM要素の特定に関する問題
 - getLabelByKey はまず document.getElementById(key) で、該当しなければ querySelector('label[for="' + key + "']') で取得を試みます。
 - 対象のラベルがこれらの属性（id または for）に合致しない場合、ラベルが見つからず更新が行われません。
- 動的追加・変更への対応
 - 初期実行時に取得できなかったラベル（後から動的に追加される場合など）には、MutationObserver の設定や初回更新処理が適用されない可能性があります。
- ブラウザ互換性
 - MutationObserver を利用しているため、これをサポートしない古いブラウザではラベルのDOM変化が検知できず、期待通りの更新が行われなくなる可能性があります。
- グローバル関数のオーバーライド
 - waitAndOverridePreparePush で App.Defects.preparePush を上書きしていますが、対象の関数が存在しない場合は1秒ごとに再試行します（無限にリトライする可能性もある）。
 - 他のスクリプトや後からの再定義によって上書きが無効になった場合、ラベルのポーリングが始まらない可能性があります。
- DOM構造や他スクリプトとの競合
 - ラベル要素の内部構造に依存して更新（例：label.firstChild のテキスト取得）しているため、他のスクリプトがラベルの内容を操作すると、意図しない挙動（例：上書きの競合や無限更新ループ）が起こる可能性があります。

以上