



Quality **Forward**

# BTS 連携マニュアル

VERISERVE

最終更新日 : 2021/10/26

# 目次

第1章	BTS 連携時の注意点	2
1.1.1.	共通確認事項	2
1.1.2.	BTS 別確認事項	2
第2章	BTS 連携を行う	3
2.1.	Redmine と連携する	3
2.1.1.	Redmine のベース URL を設定する	4
2.1.2.	バグ曲線、グラフデータ取得用 URL を設定する	5
2.1.3.	最近のインシデント取得用 URL を設定する	8
2.1.4.	新規チケット作成画面の URL	8
2.2.	JIRA サーバー型と連携する	8
2.2.1.	ユーザ名とパスワードを入力する	9
2.2.2.	URL・コンテキストパスを設定する	10
2.2.3.	バグ曲線、グラフデータ取得用 JQL を設定する	11
2.2.4.	最近のインシデント取得用 JQL を設定する	12
2.2.5.	新規チケット作成画面の URL を設定する	13
2.3.	JIRA クラウド型と連携する	15
2.3.1.	ユーザ名とパスワードを入力する	15
2.3.2.	バグ曲線、グラフデータ取得用 JQL を設定する	16
2.3.3.	最近のインシデント取得用 JQL を設定する	16
2.3.4.	新規チケット作成画面の URL を設定する	17
第3章	よくある質問と回答	22
3.1.	Q: BTS 疎通ができているか確認したい	22
3.2.	Q: BTS 疎通ができない	23
3.3.	Q: BTS の件数がグラフに反映されない	23
3.4.	Q: BTS の Open/Close の件数がレポートに反映されるのはいつですか？	23
3.5.	Q: 過去の BTS の Open/Close 数を修正したい	24
3.6.	Q: JIRA のクラウド型を使用したい	24

3.7. Q: Redmine で BTS 疎通ができない .....	25
3.8. Q: BTS で集計されるチケット範囲を絞りたい .....	25
3.9. Q: チケットを CLOSE にしたのに欠陥 OPEN 数が変わらない .....	28

## 第1章 BTS 連携時の注意点

BTS 連携をご利用の際、以下の確認が必要です。

### 1.1.1. 共通確認事項

<b>REST API 設定</b>	有効化されていることを確認してください
<b>IP 制限</b>	アクセス制限が有効な場合、以下 IP アドレスからのアクセスを許可してください。 <b>QualityForward IP アドレス</b> 13.112.115.12 13.113.53.12 52.197.246.217 52.197.44.200 18.177.168.126 52.69.201.102

### 1.1.2. BTS 別確認事項

<b>JIRA Cloud 版</b>	7.0 以上	標準サポート
<b>JIRA Server 版</b>		AWS 上の QualityForward クラウド版から、
<b>Redmine</b>	2.0 以上	ホスティングしている環境への HTTP 通信を穴開けする必要があります。

## 第2章 BTS 連携を行う

ご使用の BTS を選択し、レポート機能と連携設定することができます。

### 2.1. Redmine と連携する

テストフェーズ一覧のテストフェーズ設定から、連携する BTS で「Redmine」を選択すると Redmine に連携するための設定項目が表示されます。



テストフェーズ一覧

▶ アクティブ 6    🗄️ アーカイブ 0

名前を検索

テストフェーズ名 ▲

0525_1 📅 2021/05/25 ~ 2021/06/25 <b>設定</b>
0525_2 📅 2021/05/25 ~ 2021/06/25 設定

## 連携するBTS

BTS連携に関する設定は[こちらの資料](#)をご確認下さい。

BTS

なし ▼

なし

**Redmine**

JIRA

BTS

Redmine ▼

**必須** ベースURL

例：https://xxx/?key=yyyy

①レポート画面での redmine へのリンクなどに利用します

**必須** バグ曲線、グラフデータ取得用URL

例：https://xxx/projects/test/issues.json?key=yyyy&query\_id=x

①バグ情報とチャートの取得に利用します。事前にすべてのチケットのjsonが取得できることをご確認ください

## 連携するBTSの追加設定

最近のインシデント取得用URL

例：https://xxx/projects/test/issues.json?key=yyyy&query\_id=y

①レポート画面で最近のインシデントにバグ曲線と別の情報を表示したい場合に設定してください。事前にレポート画面に表示したい内容のjsonが取得できることをご確認ください

新規チケット作成画面のURL

例：https://xxx/projects/test/issues/new

①テストサイクル実行画面からBTSへの起票を行う際に利用します

### 2.1.1. Redmine のベース URL を設定する

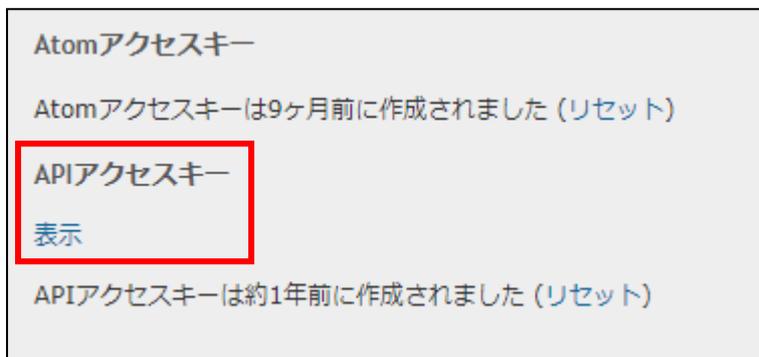
Redmine のベース URL の入力を行います。ベース URL はバグの優先度設定の取得、および「最近のインシデント一覧」のチケットのリンク生成のため利用します。

ベース URL は、Redmine のホームの URL です。

Redmine のルートにあたる URL を指し、 [https://xxxxxx.xxxx/] の場合と[https://xxxxxx.xxxx/redmine/] である場合の 2 パターンあります。URL の後には「?key=API キー」を指定します。

API キーは Redmine の個人設定から取得することができます。手順は以下の通りです。

- (1) Redmine にログインし、画面右上の個人設定を開きます。
- (2) API アクセスキーの表示をクリックすると API キーが表示されます。



## 2.1.2. バグ曲線、グラフデータ取得用 URL を設定する

バグ曲線とパイチャート表示用の URL の入力を行います。本手順で作成するクエリ設定で、取得するチケットを絞込むことができます。

**必須** バグ曲線、グラフデータ取得用URL

例 :

①バグ情報とチャートの取得に利用します。事前にすべてのチケットのjsonが取得できることをご確認ください

- (1) Redmine の個人設定から API キーを取得します。
- (2) Redmine の対象のプロジェクト開き、URL を取得します。  
例) <https://xxx.xxxx/projects/xxxx/>

(3) 手順(2)で取得した URL に「issues.json?key=API キー」を追加します。

例) <https://xxx.xxxx/projects/xxxx/issues.json?key=xxxx>

(4) 取得する情報のフィルタ条件を設定し、対象チケットの絞り込みを行います。バグ一覧を取得するために「すべて」の「バグ」を対象にし、適用ボタンを押します。

The screenshot shows the 'チケット' (Issues) page with the following filter settings:

- フィルタ: ステータス (すべて), トラッカー (等しい)
- 検索条件: Bug
- アクション: 適用 (highlighted), クリア, 保存

※ここでは「すべて」の「バグ」を対象にしていますが、フィルタ条件はプロジェクト方針に合わせて自由に設定していただけます。

(5) すべてのバグの一覧が表示されたら保存ボタンを押します。

The screenshot shows the 'チケット' (Issues) page with the same filter settings as above. The '保存' (Save) button is highlighted with a red box.

(6) 新しいクエリに名前を付けて保存します。

新しいクエリ

名称

表示  自分のみ  
 すべてのユーザー  
 次のロールのみ:  
 Manager  
 Developer  
 Reporter

全プロジェクト向け

オプション

デフォルトの項目

グループ条件

表示  説明  
合計  予定工数  作業時間

フィルタ

ステータス   
 トラッカー

ソート条件

1:    
2:    
3:

- (7) チケット一覧右側のカスタムクエリ一覧に手順(6)で作成したクエリが表示されます。作成したクエリ名をクリックします。

チケット

[すべてのチケットを表示](#)

[サマリー](#)

[カレンダー](#)

[ガントチャート](#)

[インポート](#)

カスタムクエリ

- (8) URL の最後にクエリ ID が表示されるので、「query\_id=xx」をコピーします。

`/issues?query_id=25`

- (9) 手順(3)までで作成した URL の最後に手順(8)の「&query\_id=xx」を入力します。  
例)[https://xxx.xxxx/projects/xxxx/issues.json?key=xxxx&query\\_id=xx](https://xxx.xxxx/projects/xxxx/issues.json?key=xxxx&query_id=xx)

- (10) 手順(9)でできた URL をブラウザのアドレス欄に直接入力します。json の取得が確認できたら URL を登録欄に入力し、登録ボタンを押します。

### 2.1.3. 最近のインシデント取得用 URL を設定する

連携した BTS の「最近のインシデント一覧」を表示するため、指定された範囲のチケット情報を取得します。未設定の場合は「バグ曲線、グラフデータ」に設定した URL から情報を取得します。

連携するBTSの追加設定

最近のインシデント取得用URL

例：https://xxx/projects/test/issues.json?key=yyyy&query\_id=y

●レポート画面で最近のインシデントにバグ曲線と別の情報を表示したい場合に設定してください。事前にレポート画面に表示したい内容のjsonが取得できることをご確認ください

※URL の取得は手順 [2.1.2](#) と同様に行います。

※最近のインシデント取得用 URL はバグ曲線やチャートと別の情報を表示させたい場合に指定してください。

### 2.1.4. 新規チケット作成画面の URL

新規チケット作成画面の URL を設定すると、テストサイクルでテストを実行中に右クリックから簡単にチケットへの起票を行うことができます。

チケットの起票画面の URL を記載してください。

例) https://xxx/projects/test/issue/new

## 2.2. JIRA サーバー型と連携する

テストフェーズ一覧のテストフェーズ設定から、連携する BTS で「JIRA」を選択すると JIRA

に連携するための設定項目が表示されます。

## テストフェーズ一覧

▶ アクティブ **6**    📁 アーカイブ **0**

名前を検索

テストフェーズ名 ▲

0525\_1  
📅 2021/05/25 ~ 2021/06/25  
**✎ 設定**

0525\_2  
📅 2021/05/25 ~ 2021/06/25  
✎ 設定

## 連携するBTS

BTS連携に関する設定は[こちらの資料](#)をご確認下さい。

BTS

なし ▼

なし  
Redmine  
**JIRA**

### 2.2.1. ユーザ名とパスワードを入力する

情報を取得するために、JIRA に登録済みのユーザ名とパスワードを入力します。

BTS連携に利用するのは各個人のクレデンシャルではなく、専用に作成した物をご利用ください。

**必須** ユーザ名

ユーザ名

**必須** パスワード

パスワード

## 2.2.2. URL・コンテキストパスを設定する

取得したい課題の登録されたプロジェクトを含む JIRA の URL を設定します。コンテキストパスは JIRA 側で設定を行っている場合にのみ入力してください。

※コンテキストパスは「/xxx」の形式で入力してください。

**必須** JIRAのURL

https://xxx.jp/

**必須** バグ曲線、グラフデータ取得用JQL

例：project="QF1"

●バグ情報とチャートの取得に利用します。事前にすべてのIssueが「新しい順で」取得できることをご確認ください

### 連携するBTSの追加設定

コンテキストパス

例：/jira

●JIRA側で設定している場合にのみ入力してください

- (1) JIRA の管理メニューからシステムを選択します。



(2) 一般設定を開きます。



The screenshot shows the JIRA Administration interface. The top navigation bar includes 'ダッシュボード', 'プロジェクト', '課題', 'ボード', and '作成'. The main content area is titled '管理' with a search bar. A sidebar on the left lists various administration categories, with '一般設定' (General Settings) highlighted with a red box. The main content area displays the '設定' (Settings) page for '一般設定', showing various configuration options.

設定	
一般設定	
タイトル	QFJIRA
モード	非公開
認証の最大試行回数	3
サインアップ時に CAPTCHA を使用	オフ
ベース URL	http://jira-eval.vtsuite.net:8080
メールの差出人	\${fullname} (JIRA)
概要	
多言語対応	

(3) 一般設定内にあるベース URL をコピーし、JIRA の URL に入力します。



The screenshot shows a close-up of the '設定' (Settings) page for '一般設定'. The 'ベース URL' (Base URL) field is highlighted with a red box, containing the value 'http://jira-eval.vtsuite.net:8080'.

設定	
一般設定	
タイトル	QFJIRA
モード	非公開
認証の最大試行回数	3
サインアップ時に CAPTCHA を使用	オフ
ベース URL	http://jira-eval.vtsuite.net:8080
メールの差出人	\${fullname} (JIRA)
概要	
多言語対応	

### 2.2.3. バグ曲線、グラフデータ取得用 JQL を設定する

JQL は「JIRA Query Language」の略で、JIRA 専用のクエリ言語を指します。取得するプロ

プロジェクトや課題のタイプなどを絞り込むために JQL を設定する必要があります。未設定の場合は JIRA に登録されている全てのプロジェクト、課題が対象となります。

**必須** バグ曲線、グラフデータ取得用JQL

例：project="QF1"

①バグ情報とチャートの取得に利用します。事前にすべてのIssueが「新しい順で」取得できることをご確認ください

例) 特定のプロジェクトを対象とする場合は project="プロジェクトキー"

例) 特定の課題タイプを対象とする場合は issueType = "課題タイプ"

※OPEN/CLOSE 情報は JIRA 側の設定に依存します。

## 2.2.4. 最近のインシデント取得用 JQL を設定する

連携した BTS の「最近のインシデント一覧」を表示するため、指定された範囲のチケット情報を取得します。未設定の場合は「バグ曲線、グラフデータ」に設定した JQL から情報を取得します。

**連携するBTSの追加設定**

コンテキストパス

例：/jira

①JIRA側で設定してる場合のみ入力してください

**最新のインシデント用JQL**

例：project = QF AND issuetype = バグ

①「バグ曲線、グラフデータ取得用JQL」と異なる情報を表示したい場合に入力してください

※URL の取得は手順 [2.2.3](#) と同様に行います。

※最近のインシデント取得用 JQL はバグ曲線やチャートと別の情報を表示させたい場合に指定してください。

## 2.2.5. 新規チケット作成画面の URL を設定する

新規チケット作成画面の URL を設定すると、テストサイクルでテストを実行中に右クリックから簡単にチケットへの起票を行うことができます。

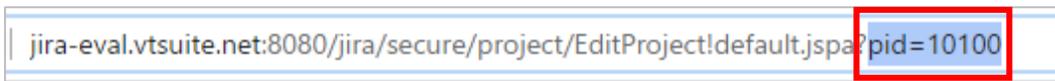
- (1) JIRA の URL の後に「/secure/CreateIssueDetails!init.jspa?pid=**yyy**&issuetype=**zzz**」を追加します。PID と issuetype は以下手順で設定します。
- (2) プロジェクト設定を開きます。



- (3) 詳細情報を選択します。

A screenshot of the 'プロジェクト設定' (Project Settings) page. The '詳細情報' (Details) tab is selected and highlighted with a red box. The page is divided into two main sections: '要約' (Summary) on the left and '詳細情報' (Details) on the right. The '要約' section includes links for 'プロジェクトを再インデックス化' and 'プロジェクトを削除', and a list of '課題タイプ' (Issue Types) such as 'エピック', 'サブタスク', 'タスク', 'バグ', '改善', and '新機能'. The '詳細情報' section contains several input fields: '名前\*' (Name) with the value 'テストプロジェクト1', 'キー\*' (Key) with the value 'QF3', 'URL', 'プロジェクトタイプ\*' (Project Type) with a dropdown menu showing 'Software', 'プロジェクトカテゴリー' (Project Category) with the value 'None', 'アバター\*' (Avatar) with a selection icon and the text '画像の選択', and '説明' (Description).

- (4) 詳細情報画面の URL の最後に記載されている PID を新規チケット作成画面の URL に設定します。



jira-eval.vtsuite.net:8080/jira/secure/project/EditProject!default.jspa?pid=10100

- (5) プロジェクト設定画面で課題タイプを選択します。



- (6) 課題タイプの画面 URL に記載されている issuetype を新規チケット作成画面の URL に設定します。



jira-eval.vtsuite.net:8080/jira/plugins/servlet/project-config/QF3/issuetypes/10104/workflow

※JIRA との接続に失敗する場合は以下の項目を確認してください

1. ログインに失敗する場合は JIRA で直接ログインをした後に連携設定を試みてください  
(JIRA のログインを複数回失敗すると CAPTCHA 認証が必要となります)
2. プロジェクト設定権限があることを確認してください
3. JIRA の認証設定が BASIC 認証となっていることを確認してください

## 2.3. JIRA クラウド型と連携する

テストフェーズ一覧のテストフェーズ設定から、連携する BTS で「JIRA」を選択すると JIRA に連携するための設定項目が表示されます。

### テストフェーズ一覧

▶ アクティブ **6**    🗄️ アーカイブ **0**

名前を検索

テストフェーズ名 ▲

0525\_1  
📅 2021/05/25 ~ 2021/06/25  
**設定**

0525\_2  
📅 2021/05/25 ~ 2021/06/25  
設定

### 連携するBTS

BTS連携に関する設定は[こちらの資料](#)をご確認下さい。

BTS

なし ▼

なし

Redmine

**JIRA**

### 2.3.1. ユーザ名とパスワードを入力する

クラウド型をご利用の方は API トークンの作成を行い、メールアドレスとパスワードを入力し

てください。

- (1) <https://ja.confluence.atlassian.com/cloud/api-tokens-938839638.html> を参考に API トークンの作成を行ってください。
- (2) ユーザ名/パスワードを以下に設定してください。  
ユーザ名 : メールアドレス  
パスワード : API トークン

## 2.3.2. バグ曲線、グラフデータ取得用 JQL を設定する

JQL は「JIRA Query Language」の略で、JIRA 専用のクエリ言語を指します。取得するプロジェクトや課題のタイプなどを絞り込むために JQL を設定する必要があります。未設定の場合は JIRA に登録されている全てのプロジェクト、課題が対象となります。

JQL

例 : project="QF1"

●バグ情報の取得に利用します。事前にすべてのIssueが「新しい順で」取得できることをご確認ください

例) 特定のプロジェクトを対象とする場合は project="プロジェクトキー"

例) 特定の課題タイプを対象とする場合は issueType = "課題タイプ"

※OPEN/CLOSE 情報は JIRA 側の設定に依存します。

## 2.3.3. 最近のインシデント取得用 JQL を設定する

連携した BTS の「最近のインシデント一覧」を表示するため、指定された範囲のチケット情報を取得します。未設定の場合は「バグ曲線、グラフデータ」に設定した JQL から情報を取得します。

## 連携するBTSの追加設定

コンテキストパス

例：/jira

① JIRA側で設定している場合のみ入力してください

最新のインシデント用JQL

例：project = QF AND issuetype = バグ

① 「バグ曲線、グラフデータ取得用JQL」と異なる情報を表示したい場合に入力してください

※URLの取得は手順 [2.3.2](#) と同様に行います。

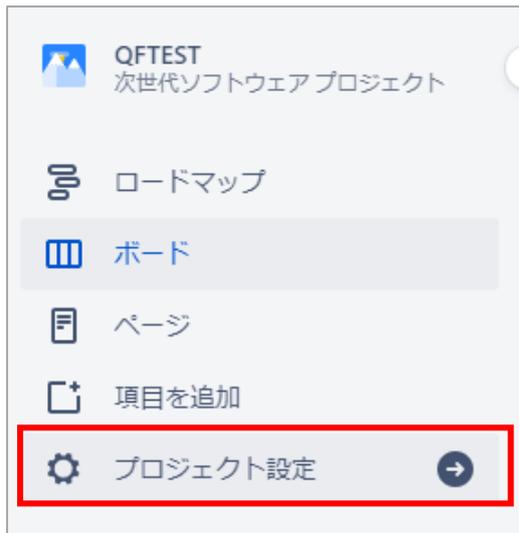
※最近のインシデント取得用JQLはバグ曲線やチャートと別の情報を表示させたい場合に指定してください。

## 2.3.4. 新規チケット作成画面のURLを設定する

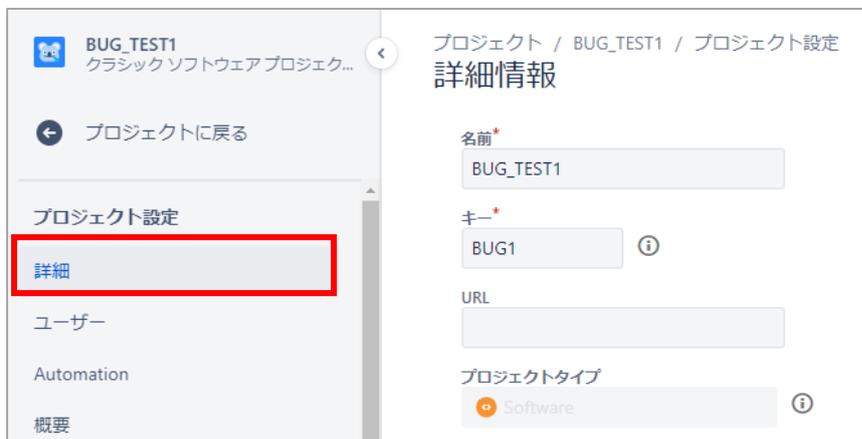
新規チケット作成画面のURLを設定すると、テストサイクルでテストを実行中に右クリックから簡単にチケットへの起票を行うことができます。

### クラシックプロジェクトをご利用の場合

- (1) 「クラシックプロジェクト」をご利用の場合、JIRAのURLの後に「/secure/CreateIssueDetails!init.jsps?pid=yyy&issuetype=zzz」を追加します。  
PIDとissuetypeは以下手順で設定します。
- (2) プロジェクト設定を開きます。



(3) 詳細情報を選択します。



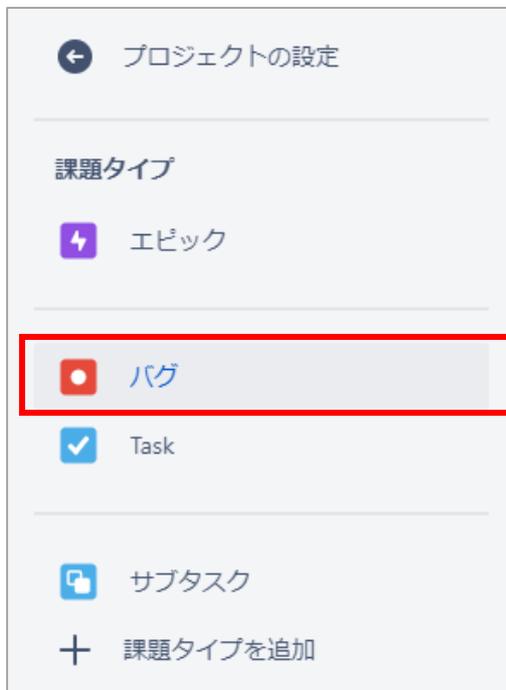
(4) 詳細情報画面の URL の最後に記載されている PID を新規チケット作成画面の URL に設定します。

`/secure/project/EditProject!default.jspa?pid=10000`

(5) プロジェクト設定画面で課題タイプを選択します。



(6) チケット新規作成時にデフォルトにしたい課題タイプを選択します。



(7) 課題タイプの画面 URL に記載されている issuetype を新規チケット作成画面の URL に設定します。



## 次世代プロジェクトをご利用の場合

次世代ソフトウェアプロジェクトでは、RestAPI から PID をご参照ください。手順は以下の通りです。

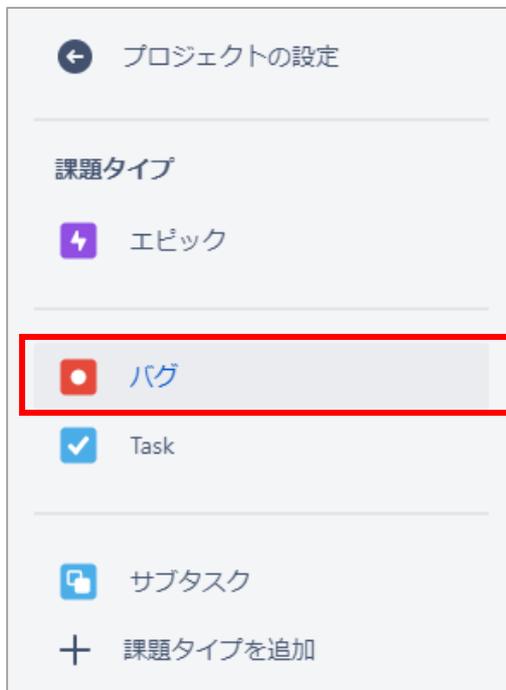
- (1) 「次世代プロジェクト」をご利用の場合、JIRA の URL の後に「/secure/CreateIssueDetails!init.jsps?pid=yyy&issuetype=zzz」を追加します。PID と issuetype は以下手順で設定します。
- (2) ブラウザに以下の URL を入力してください。JSON の一番上の「id」の項目が PID です。  
`https://[JIRA の URL]/rest/api/2/project/[プロジェクトキー]`
- (3) プロジェクト設定画面を開きます。



- (4) プロジェクト設定画面で課題タイプを選択します。



(5) チケット新規作成時にデフォルトにしたい課題タイプを選択します。



(6) 課題タイプの画面 URL に記載されている issuetype を新規チケット作成画面の URL に設定します。

`/jira/software/projects/QT/settings/issuetypes/10010`

## 第3章 よくある質問と回答

### 3.1. Q: BTS 疎通ができていないか確認したい

BTS の疎通確認は疎通確認画面より行っていただけます。確認手順は以下の通りです。

- (1) テストフェーズ一覧のテストフェーズ名下部にある「Redmine（または JIRA）との疎通確認」をクリックします。

テストフェーズ名 ▼	フェーズ開始日
テストサンプル_画面遷移テスト 📅 2018/08/30 ~ 2018/09/30 🔗 設定	2018/08/30
サンプルフェーズ東京 📅 2017/02/21 ~ 2017/03/22 🔗 設定 <b>Redmineとの疎通確認</b>	2017/02/21
βリリース向けフル試験 📅 2017/02/21 ~ 2017/03/22 🔗 設定	2017/02/21

- (2) 疎通確認画面で「ベース URL の疎通確認」の項目が「○」となっていれば疎通に成功しています。「×」となっている場合は設定を見直していただく必要がございます。

チェック内容	結果
ベースURLの疎通確認	○
バグ曲線、グラフデータ取得用URLの疎通確認	○
最近のインシデント取得用URL	○

## 3.2. Q: BTS 疎通ができない

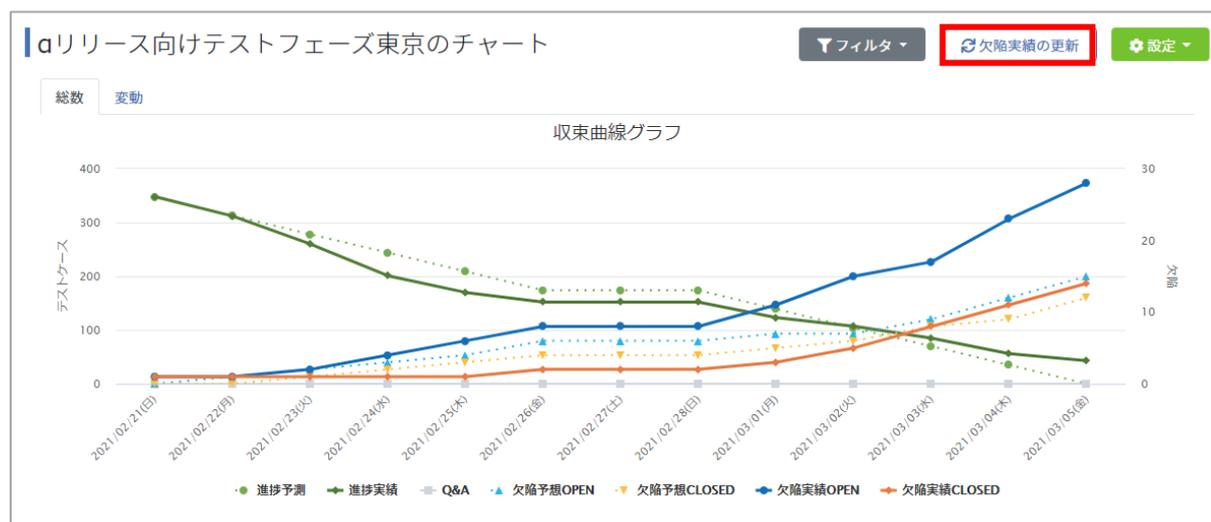
BTS に接続する際に外部のシステムである QualityForward からのアクセスが弾かれてしまう場合がございます。その場合は QualityForward の IP アドレス（[第 1 章参照](#)）からのアクセスを許可していただく必要がございます。

本設定は Redmine のシステム管理者権限を持っているユーザー様にご対応いただく必要がございます。

## 3.3. Q: BTS の件数がグラフに反映されない

フェーズ毎チャート画面の右上にある「欠陥実績の更新」ボタンを押してください。

※BTS の情報が自動で反映されるタイミングは 1 日 3 回（8 時・12 時・18 時）です。



## 3.4. Q: BTS の Open /Close の件数がレポートに反映されるのはいつですか？

1 日 3 回（8 時・12 時・18 時）です。任意のタイミングで更新したい場合は、「欠陥実績の更新」ボタンを押すことでレポートに反映されます。

### 3.5. Q: 過去の BTS の Open/Close 数を修正したい

フェーズ毎チャート画面右上にある設定の「バグ情報のアップロード」から修正が可能です。



### 3.6. Q: JIRA のクラウド型を使用したい

以下の手順に従い設定を行ってください。

- (1) [こちら](#)を参考に API トークンの作成を行ってください。
- (2) ユーザ名/パスワードを以下のように設定してください。  
ユーザ名 : メールアドレス  
パスワード : API トークン

## 3.7. Q: Redmine で BTS 疎通ができない

Redmine の設定が有効になっていない可能性があります。

- (1) 「管理」 → 「設定」 → 「API」 タブを開きます。
- (2) 「REST による Web サービスを有効にする」 にチェックを入れます。



設定

全般 表示 認証 API プロジェクト チケットトラッキング ファイル

RESTによるWebサービスを有効にする

JSONPを有効にする

保存

- (3) 「保存」 ボタンを押します。

## 3.8. Q: BTS で集計されるチケット範囲を絞りたい

集計されるチケットの範囲を絞るには、Redmine 側で絞り込みを行ったものを指定していただく必要があります。Redmine での絞り込み手順は以下の通りです。

- (1) Redmine の個人設定から API キーを取得します。
- (2) Redmine の対象のプロジェクト開き、URL を取得します。  
例) <https://xxx.xxxx/projects/xxxx/>
- (3) 手順(2)で取得した URL に「issues.json?key=API キー」を追加します。例)  
`https://xxx.xxxx/projects/xxxx/issues.json?key=xxxx`

- (4) Redmine で対象チケットの絞込みを行います。バグ一覧を取得するために「すべて」の「バグ」を対象にし、適用ボタンを押します。

チケット

▼ フィルタ

ステータス すべて ▼

トラッカー 等しい ▼ Bug ▼ □

▶ オプション

適用  クリア

※ここでは「すべて」の「バグ」を対象にしていますが、フィルタ条件はプロジェクト方針に合わせて自由に設定していただけます。

- (5) すべてのバグの一覧が表示されたら保存ボタンを押します。

チケット

▼ フィルタ

ステータス すべて ▼

トラッカー 等しい ▼ Bug ▼ □

▶ オプション

適用  クリア

- (6) 新しいクエリに名前を付けて保存します。

新しいクエリ

名称

表示  自分のみ  
 すべてのユーザー  
 次のロールのみ:  
 Manager  
 Developer  
 Reporter

全プロジェクト向け

オプション

デフォルトの項目

グループ条件

表示  説明  
合計  予定工数  作業時間

フィルタ

ステータス   
 トラッカー

ソート条件

1:    
2:    
3:

- (7) チケット一覧右側のカスタムクエリ一覧に手順(6)で作成したクエリが表示されます。作成したクエリ名をクリックします。

チケット

[すべてのチケットを表示](#)

[サマリー](#)

[カレンダー](#)

[ガントチャート](#)

[インポート](#)

カスタムクエリ

- (8) URL の最後にクエリ ID が表示されるので、「query\_id=xx」をコピーします。

- (9) 手順(3)までで作成した URL の最後に手順(8)の「&query\_id=xx」を入力します。  
例)https://xxx.xxxx/projects/xxxx/issues.json?key=xxxx&query\_id=xx

(10) 手順(9)でできた URL をブラウザのアドレス欄に直接入力します。json の取得が確認できたら URL を登録欄に入力し、登録ボタンを押します。

### 3.9. Q: チケットを CLOSE にしたのに欠陥 OPEN 数が変わらない

欠陥 OPEN 数はチケットの総数を表しているため、グラフが減ることはありません。