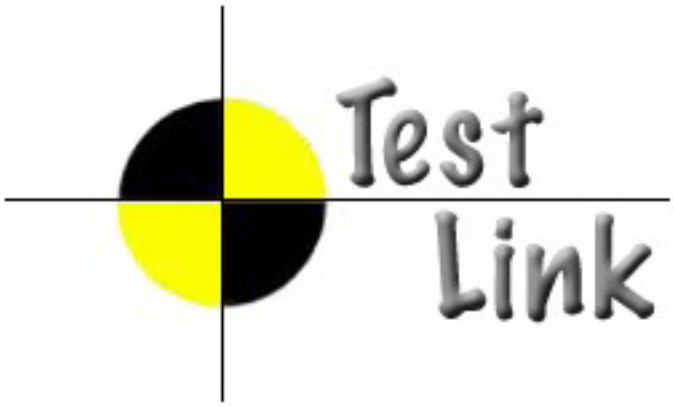


インストール & 構成マニュアル

Revesion 1.3
TestLink バージョン 1.7

翻訳: Testing Engineer’s Forum (TEF) - ソフトウェアテスト技術者交流会 - 「TEF 有志による TestLink 日本語化プロジェクト」部会

目次

1　ドキュメントのスコープ

このドキュメントは、TestLink 1.7 のインストールと構成方法に関するリファレンスと技術情報を提供します。初めの部分はインストールの手順、次の部分は構成方法の説明になっています。最新のドキュメントは TestLink のホームページで参照することができます。さらに TestLink フォーラムで問題の解決方法を質問することもできます。

インストール手順の要約:

1. 使用するサービス環境のインストール
2. Web サーバへの展開
3. データベーステーブルの作成とデータの追加 (新規作成、または、以前のデータベースからの移行)
4. 構成ファイルの編集
5. PHP の拡張設定
6. ログイン

TestLink には構成の設定やデータベース構築を容易にするインストールスクリプトが含まれています

2　システム要件

TestLink に必要な環境は以下の通りです：

データベース

- MySQL 4.1.x またはそれ以上 (4.0.x は UTF-8 をサポートしていません)
- PostgreSQL 8.x またはそれ以上
- Microsoft SQL Server 2000

スクリプト

- PHP 5.x またはそれ以上 (推奨 5.2)

Web サーバ

- Apache 1.3.x、2.x またはそれ以上
- IIS 3 またはそれ以上

詳細は PHP のドキュメントを参照してください

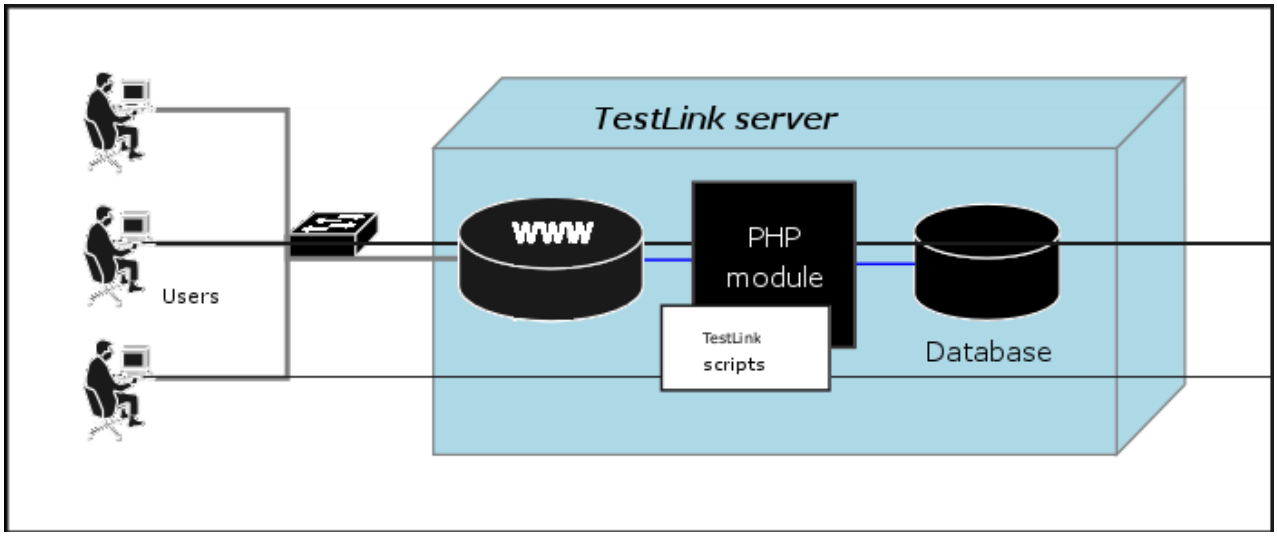
バグ管理システム（オプション）

- Bugzilla 0.19.1 またはそれ以上
- Mantis 1.0.1 またはそれ以上
- JIRA 3.1.1 またはそれ以上
- TrackPlus

使用環境

- OS には依存しません (Linux と Win32 環境で確認済みです)

MySQL は様々なサーバで使用できるのでお勧めです



3 インストール

インストールは自動または手動で行うことができます。以前のバージョンの TestLink から更新する場合はアップグレードの章 を参照して下さい。

3.1 インストールの準備

インストールを始める前に以下のことを行ってください。

- 次の環境を用意します：
  - PHP をインストールした Web サーバ
  - データベースサーバ

詳細は当該製品のドキュメントを参照してください。上記環境を一括してインストールすることができる統合パッケージを使用する方法もあります
例： XAMPP, EasyPHP, Uniform Server など
TestLink 1.7 では PHP4 をサポートしていません

- Web サーバに TestLink のインストールファイルを転送してください。(ftp, scp, 等) 次のステップでは telnet か ssh でサーバーマシンにログインする必要があります
- 任意のディレクトリに転送したファイルを解凍します。通常は以下のようなコマンドを使用します:

  ```
  # tar zxvf <filename.tar.gz>
  ```

  もしくは

  ```
  # gunzip <filename.tar.gz>
  # tar xvf <filename.tar>
  ```

  Winzip や Stuffit などの他の解凍プログラムでも同じような操作でアーカイブを解凍できます。解凍したらディレクトリ名を”testlink”のような簡単な名前に変更してください。名前の変更には”mv”コマンドを使用します。(Windows の場合は”ren”コマンドやエクスプローラを使用します)

  ```
  # mv <directory_name> testlink
  ```

- インストールや更新を続行します。

3.2 自動インストール

TestLink には構成の設定やデータベース構築を容易にするインストールスクリプトが含まれています。システムにインストールするための基本的な手順を以下に示します。手順は Unix を主な対象としていますが、Windows でも同じように行うことができます。複雑な構成を行わないのであればインストールから TestLink を使えるようになるまでにかかる時間は 10 分から 30 分程度です。なお、インストール手順は 1.6 から変更されています。インストール後はデータベースのテーブルと基本構成ファイルを作成します。

- Web ブラウザで にアクセスし’New installation’ を選びます。
- このページでは以下の操作を行います：

- Web サーバや PHP 構成の基本パラメータチェック、データベースのバージョンチェック
- データベースのタイプや場所、ユーザやパスワードの指定インストールには、アドミニストレータ権限を持つユーザとパスワードが必要です。これは変更、選択、挿入、更新の各操作を行う権利を要求するためです。また、インストールには、データベースや (データ) テーブルを作成するためにインデックス、作成、削除の各操作を行う権利も必要です。
- データベースとテーブルを作成

  警告： デフォルトの管理者権限のアカウントが作成されます。

  次のアカウントとパスワードが作成されます：admin / admin

  TestLink に最初にログインするときは、このアカウントを使用してください。ログイン後、まず最初に最低でも一つの管理者権限のアカウントを作成してください。そして最初の管理者権限のアカウント (admin) を無効にするか削除してください。この管理者権限のアカウントは再度作成することができますが、セキュリティ上の理由からアカウントを削除することを強く推奨します。

  注意： パッケージの設定後デフォルトの管理者権限のアカウントを削除してください。
- インストール後のシステムをチェック

- インストールが成功したら、セキュリティ上の理由から下記のディレクトリ以下を削除してください。`<testlinkdir>/install/`
- 次は特定のセットアップ事項に基づいた構成を行います。構成パラメータの詳細な説明は構成の章 を参照して下さい。

3.3 手動インストール

手動でのインストールを行う場合は以下の手順で行ってください。(手動インストールは推奨されていません) データベースをインストールするために MySQL のコマンドツールかデータベースクライアント (例 phpMyAdmin) のいずれかを使用することができます。

- コマンドツールでの MySQL の準備：
  - 新しい空の MySQL データベースを作成します。MySQL 4.1 以上 (UTF8 サポート) の場合、次のようなコマンドを入力します

    ```
    # CREATE DATABASE testlink CHARACTER SET utf8 COLLATE utf8_general_ci
    ```

    UTF8 をサポートするには上記の他に `<testlinkdir>/config.inc.php` ファイルの ”DB_SUPPORTS_UTF8”を”TRUE”に設定する必要があります。詳細は構成の章 を参照して下さい。
  - 作成したデータベースにテーブルを作成します

    ```
    # mysql -u <user> -p<password> <dbname> <
      <testlinkdir>/install/sql/<yourdb>/testlink_create_tables.sql
    ```

    例：

    ```
    # mysql -u testlink -ppass testlink <
      /var/www/html/testlink/install/sql/mysql/testlink_create_tables.sql
    ```

– 作成したデータベースに初期データ (デフォルトのアカウントや役割) を入力します

```
# mysql -u <user> -p<password> <dbname> <
  <testlinkdir>/install/sql/<yourdb>/testlink_create_default_data.sql
```

- phpMyAdmin を使用する場合：
  - メインページから新しいデータベースを作成します (UTF8 文字セットを推奨)
  - （オプション）新しいユーザを作成し、データベースに対して適切な権限のアサインを行います
  - 左シートから作成したデータベースを選択します
  - SQL ウィンドウに移動します
  - `/install/sql/<yourdb>/testlink_create_tables.sql` ファイルの SQL 要求をアップロードして、そのスクリプトを実行します
  - `/install/sql/<yourdb>/testlink_create_default_data.sql` ファイルの SQL 要求をアップロードして、そのスクリプトを実行します
- `<testlinkdir>/config_db.inc.php` ファイルを作成し、以下の例を参考にデータを設定してください：

```
<?php
// Automatically Generated by TestLink Installer
define('DB_TYPE', 'mysql');
define('DB_USER', 'testlinker');
define('DB_PASS', 'testlink_pass');
define('DB_HOST', 'localhost');
define('DB_NAME', 'tl_master');
?>
```

- (オプション) TestLink からデータベースに接続するためのユーザを作成します。ユーザにはデータベースに対して適切な権限を与えることを忘れないようして下さい。(最低限必要な権限は選択、挿入、更新、削除です) 作成したユーザは config_db.inc.php にも記述されている必要があります。ユーザを新規に作成せずに適切な権限をもった既存のユーザを使用することもできます。
- Linux / UNIX 上では、Web サーバによって書き込みが可能なように templates_c ディレクトリの権限を変えてください。TestLink のルートディレクトリから以下のコマンドを実行します。

```
# chmod 777 gui/templates_c
```

- TestLink にログインします。デフォルトのアカウントは以下を使用します：
  ユーザ名: admin
  パスワード: admin
- このパスワードは、安全のためにログインしたらすぐに変更してください。変更していない場合は TestLink から通告されます。
- インストールが成功したら、セキュリティ上の理由から下記のディレクトリ以下を削除してください。
  `<testlinkdir>/install/`

- 次は特定のセットアップ事項に基づいた構成を行います。構成パラメータの詳細な説明は構成の章 を参照して下さい。
- 問題点やフィードバックに関しては、TestLink のバグ管理システムに報告をお願いします。

3.4 アップグレード

主要バージョンへの更新は、手動更新と自動更新 (スクリプト経由) のいずれかの方法で行うことができます。過去の主要バージョンからデータベースにいくつかの変更が加えられたため、それらのデータベースを引き続き使用することができません。修正プログラムへの更新 (例:1.7.1 → 1.7.2) では上記は必須ではありません。

※訳注: TestLink 1.7.0 → 1.7.1 の間ではデータベーススキーマが変更されていますので、更新作業が必要になります。

3.4.1 修正プログラムへの更新

例えば 1.6.0 → 1.6.1 のように修正プログラムを公開することがあります。この場合はデータベーススキーマは変更されていません。

- 前のバージョンのファイルを保存します
- 保存したら全てのファイルをディレクトリから削除します
- 同じディレクトリに新しいバージョンをコピーします
- 上記ディレクトリに config_db.inc.php ファイルをコピーし、前のバージョンに適用していた設定に沿って構成パラメータを修正してください。
- 以上で更新の完了です

3.4.2 主要バージョンへの自動更新

- システム要件を確認し、インストールの準備の章 に記述されている手順に従います
- Web ブラウザから を表示してください
- 'Upgrade Installation' を選択してください。スクリプトが実行されるので終了まで待ちます
- 更新が成功したら、セキュリティ上の理由から下記のディレクトリ以下を削除してください。`<testlinkdir>/install/`
- 次は特定のセットアップ事項に基づいた構成を行います。構成パラメータの詳細な説明は構成の章 を参照して下さい
- 問題点やフィードバックに関しては、TestLink のバグ管理システムに報告をお願いします

3.4.3 主要バージョンへの手動更新

この章は前のバージョンに対して加えられた変更点を説明しています。主要バージョンへの更新は自動更新を推奨しています。この章の内容は特殊なケースや参考情報としてのみ参照してください。データベースに加えられた変更やインストールスクリプトを熟知すれば手動更新を行うことができるようになります。現在のバージョンや新しいバージョンに含まれているデータベースを作成する SQL ファイルの内容を比較することも理解することに役立ちます。

3.5 旧バージョンとの互換性

3.5.1 データベーススキーマの変更

- ユーザのパスワードを暗号化 (1.5)
- 要件仕様機能のためにテーブルを追加 (1.6) requirements req_coverage requirement_doc
- 添付ファイル機能を追加 (1.7)
- カスタムフィールドを追加 (1.7)

3.5.2 用語

- プロダクト → テスト計画
- コンポーネント、カテゴリー → テストスイート

3.5.3 削除された機能

- メインページの個人メトリクス構成パラメータ名： MAIN_PAGE_METRICS_ENABLED

3.5.4 テスト計画とテストプロジェクトの関連づけ

TestLink 1.0.4 はテスト計画とテストプロジェクト（旧名：プロダクト）が関連づけられていません。この問題に対処するために TestLink 1.6 からはテスト計画のテーブルに TestProjectID が追加されました。テスト計画は全てのテストプロジェクト間で使用可能でしたが、このようなテスト計画には TestProjectID に 0 を設定しています。

- 関連する構成パラメータ： config.inc.php

```
$g_show_tp_without_prodid = 1;
$g_ui_show_check_filter_tp_by_testproject = 1;
```

3.5.5 Latin から UTF8 への変換 (1.5 以下からの更新)

TestLink 1.6 からは UTF8 に対応しています。そのため、1.5 では表示されなかったデータベース内に登録された拡張文字も 1.6 では表示することができます

`<testlinkdir>/config.inc.php` ファイルの値を変更することで UTF8 をサポートをしないようにもできますが、ASCII 以外の文字を使用することができなくなります。

上記と同じ問題を抱えていたが、1.6 への更新と UTF-8 のサポートを有効にした後にそれらの文字が表示されたのならば以降の記述や演習が参考になります。また、この演習は実運用環境で実行する前にテスト環境で試すようにしてください。

次の手順を実行することにより、データベースから ASCII 以外の文字を排除して UTF8 をサポートすることができるようになります。

- mysqldump を使って現在のデータベースをバックアップします

```
# /usr/bin/mysqldump -u root testlink15 -p > testlink15.backup
```

- 各テーブルに UTF8 エンコードを定義するために testlin15.backup を編集します。各テーブルの CHARSET を latin1 から utf8 へ変更します。例えば次のようなテーブル定義を

```
ENGINE=MyISAM DEFAULT CHARSET=latin1
COMMENT='This table holds the bugs filed for each result';
```

次のように変更します

```
ENGINE=MyISAM DEFAULT CHARSET=utf8
COMMENT='This table holds the bugs filed for each result';
```

- testlink15.backup を以下のような Perl スクリプトにかけます

```
./replaceScript.pl < testlink15.backup > testlink15.cleaned
```

Perl スクリプトの内容は次のようになります

```
#!/usr/bin/perl
while (<>) {
    chomp;
    tr/\000-\177/\040/cs;
    print $_, "\n";
}
```

- 空の testlink16 データベースを UTF8 文字列を定義して作成します

```
CREATE DATABASE testlink16 CHARACTER SET utf8;
```

- 作成したデータベースに変更したテーブルをインストールします

```
# mysql testlink16 -u root -p < testlink15.cleaned
```

- 次のコマンドからデータベースの文字セットに UTF8 が設定されたか確認することができます

```
login to mysql use testlink16 mysql> \s

--------------
mysql Ver 14.7 Distrib 4.1.11, for redhat-linux-gnu (i386)
Connection id:        26
Current database:     testlink15
Current user:         bugz@localhost
SSL:                  Not in use
Current pager:        stdout
Using outfile:        ''
Using delimiter:      ;
Server version:       4.1.11
Protocol version:     10
Connection:           Localhost via UNIX socket
Server characterset:  latin1
Db characterset:      utf8
Client characterset:  latin1
Conn. characterset:   latin1
UNIX socket:          /var/lib/mysql/mysql.sock
Uptime:               36 min 55 sec
--------------
```

- Testlink 1.6 の更新を実行してください

参考リンク：

- UTF8 とは何か？
  http://www.joelonsoftware.com/articles/Unicode.html
  8 進数テーブル (8 進数の 000 - 177 の値は一般の ASCII 文字列)
- 8 進数を基に検索する Perl スクリプト
  http://web.cs.mun.ca/~michael/c/ascii-table.html
- Perl の tr 操作についての説明
  http://www.unix.org.ua/orelly/perl/learn/ch15_05.htm

3.5.6 キーワード管理

同じテストプロジェクトで同じキーワードを何度も作成することを望まない場合の設定

```
$g_allow_duplicate_keywords = FALSE;
```

4 構成

4.1 構成ファイル

全ての構成パラメータは config.inc.php やその他のファイルに記述されています。下記が構成ファイルになります。

- `<testlinkdir>/config.inc.php`
- `<testlinkdir>/config_db.inc.php`
- `<testlinkdir>/cfg/<bug_tracking_system>.cfg.php`
  - `config.inc.php` メインの構成ファイルです。詳細は下記を参照して下さい。
  - `config_db.inc.php` データベースにアクセスするための構成ファイルです。このファイルはインストーラによるインストールあるいは更新作業中に作成されます。通常はこのファイルをユーザが変更する必要はありません。
  - `/cfg/bugzilla.cfg.php`
  - `/cfg/mantis.cfg.php`
  - `/cfg/jira.cfg.php`
  - `/cfg/trackplus.cfg.php` それぞれのバグ管理システム（bugzilla, mantis, Jira, Trackplus）にアクセスするための構成ファイルです。TestLink からのこれらのシステムにアクセスしたい場合にはこれらのファイルを編集する必要があります（バグ管理システム結合機能）この機能を使用したい場合は、config.inc.php のパラメータをあわせて変更する必要があります。

4.1.1 custom_config.inc.php

config.inc.php ファイルを直接変更することでシステムの構成を変更することもできますが、構成を変更する場合は custom_config.inc.php ファイルに変更を記述する方法を推奨しています。上記の方法だと TestLink の更新の際に構成情報を保存することができます。

例：メールサーバーの設定を構成するためには config.inc.php ファイルから次の行を custom_config.inc.php にコピーし、任意の構成パラメータに変更してください。

```
$g_tl_admin_email = 'tl\_admin@127.0.0.1'; # for problem/error notification
$g_from_email = 'testlink\_system@127.0.0.1'; # email sender
$g_return_path_email = 'no\_replay@127.0.0.1';

# Urgent = 1, Not Urgent = 5, Disable = 0
$g_mail_priority = 5;

// SMTP Configuration
$g_smtp_host = 'localhost'; # SMTP server MUST BE configured

// Configure only if SMTP server requires authentication
$g_smtp_username = ''; # user
$g_smtp_password = ''; # password
```

4.2 機能の構成

config.inc.php ファイルの次のパラメータを構成することができます

- DB_SUPPORTS_UTF8
  MYSQL のバージョンが 4.1（UTF8 がサポートされていない）未満の場合は FALSE を設定してください。これによりすべてのページで文字セットが ISO-8859-1 に設定されます。またデータベース内の文字セットは latin1 に設定されます。MySQL のバージョンが 4.1 以上ならば TRUE に設定してください。これにより全てのページで UTF8 がサポートされ、データベース内のデータの文字セットは UTF8 に設定されます。

- TL_LOG_LEVEL_DEFAULT
  デフォルトのログ出力レベル (NONE, ERROR, INFO, DEBUG) を設定します。TestLink は出力されたログファイルのサイズまでは確認しません。ディスクの使用量を節約するために DEBUG レベルは開発もしくはバグの調査のみに使用してください。通常は、ERROR レベルを推奨しています。

- TL_LOG_PATH
  TestLink から出力されるログのパスとファイル名を指定します。例： /tmp/testlink.log

- TL_INTERFACE_BUGS
  このパラメータはバグ管理システムへのインタフェースを設定します。値としては以下が設定できます。

  ```
  'NO' 'BUGZILLA' 'MANTIS' 'JIRA' 'TRACKPLUS' 'TRAC'
  ```

  - bugzilla の構成については cfg/bugzilla.cfg.php を参照して下さい。サポートされるバージョン: 0.19.1
  - mantis の構成については cfg/mantis.cfg.php を参照してください。サポートされるバージョン: 1.0.1
  - JIRA の構成については cfg/jira.cfg.php を参照してください。サポートされるバージョン: JIRA 3.1.1
  - Trackplus の構成については/cfg/trackplus.cfg.php を参照してください。
  - Trac の構成については/cfg/trac.cfg.php を参照してください。

- TL_IMPORT_LIMIT
  インポートできる最大のファイルサイズを設定します。デフォルト値は 204800（byte）です。この値を増やすことでより大きいファイルをアップロードできます。関連するパラメータとして TL_IMPORT_ROW_MAX があります。このパラメータはエクスポートされるファイルの 1 行の最大サイズを設定します。この値のデフォルトは 10000 文字が設定されています。

- TL_DEFAULT_LOCALE
  このパラメータではデフォルトの地域を設定します。このパラメータに設定する値は const.inc.php ファイル内で設定されている $g_locales のいずれかの値です。デフォルト値は en_GB が設定されています。

- TL_COMPANY, TL_DOC_COPYRIGHT, TL_DOC_CONFIDENT
  このパラメータに設定された文字列は印刷されたドキュメントの表紙に出力されます（要件仕様は 1.6 以降で有効です）。使用しない場合は空白にしてください。

4.2.1 要件からのテストケース作成

TestLink のユニークな機能として要件管理があります。ソフトウェア要件仕様 (SRS) を作成し、TestLink に要件仕様と要件を登録すると各要件に対応したテストケースを作成することができるようになります (テストスイートも作成されます)。構成オブジェクトである $g_req_cfg を使用して以下を構成することができます：

- 作成されるコンポーネントの名前

```
$g_req_cfg->default_component_name
 = "Component Created by Requirement - Auto";
```

- コンポーネントのスコープ

```
$g_req_cfg->scope_for_component
 = "Component/Category/Test Cases generated from Requirements";
```

- 作成されるカテゴリの名前

```
$g_req_cfg->default_category_name = "TODO";
```

- カテゴリオブジェクトの説明

```
$g_req_cfg->objective_for_category
 = "Category/Test Cases generated from Requirements";
```

カテゴリの名前については以下を構成することができます

- カテゴリの名前に要件仕様のタイトルが使用されます

```
$g_req_cfg->use_req_spec_as_category_name = TRUE;
```

- カテゴリの名前に `$g_req_cfg->default_category_name` の値が使用されます

```
 $g_req_cfg->use_req_spec_as_category_name = FALSE;
```

4.2.2 テストプロジェクト、テストスイート、テストケースの重複

テストプロジェクト、テストスイート、テストケースについては既にあるものからコピーすることができます。ユーザはどのようにコピーするかを構成ファイルにより設定することができます。$g_check_names_for_duplicates = TRUE と設定することで、以下のチェックを行うことができます。

1. プロジェクト名が重複しない。
2. プロジェクトのテストスイート名が重複しない。
3. テストスイートのテストケース名が重複しない

$g_check_names_for_duplicates に TRUE を設定すると重複する名前が見つかった場合にどのように処理するか、$g_action_on_duplicate_name で設定することができます。以下の設定値があります。

- 重複した名前を許可します (過去のバージョン 1.0.4 および 1.5.x との互換性）'allow_repeat'
- $g_prefix_name_for_copy に設定された値とコピー元の名前から新しい名前を作成します'generate_new'

- エラーを返します'block'

  フォーマットの例:

```
$g_action_on_duplicate_name = 'allow\_repeat';
$g_prefix_name_for_copy = strftime("%Y%m%d-%H:%M:%S", time());
```

4.2.3 テストプロジェクトによるテスト計画のフィルタリング

前述したようにバージョン 1.6 ではデフォルトでテスト計画はテストプロジェクトによってフィルタリングされます。次の構成パラメータを使用することにより、ユーザインターフェースからテストプロジェクトによるテスト計画のフィルタリングを有効にするか無効にするか選択することが可能になります。

```
$g_ui_show_check_filter_tp_by_testproject
```

- テスト計画のボックス上にチェックボックスが表示されます。

```
$g_ui_show_check_filter_tp_by_testproject = TRUE;
```

- ユーザインターフェースを通さずに強制的にテスト計画のフィルタリングを有効にします。

```
$g_ui_show_check_filter_tp_by_testproject = FALSE;
```

4.2.4 テスト計画とテストプロジェクトの関連づけ

バージョン 1.6 からはテスト計画を作成するときはデフォルトで現在選択されたテストプロジェクトと関連づけられます。これはテストプロジェクト毎にテスト計画をフィルタリングできることを意味します。Teslink 1.6 より前のバージョンでは、テスト計画は特定のテストプロジェクトと関連づけられていません。1.5.x から 1.6 へ更新するときは、インストーラはテスト計画がどのテストプロジェクトと関連づけられているかを知ることができませんので Test Project ID は 0 に設定されます。その結果として古いテスト計画を見つけることができなくなっていました。

この問題を解決するために以下の構成パラメータが追加されました。

```
$g_show_tp_without_prodid = TRUE;
```

また、データベースの管理者権限から上記の関連付けを手動で行い、旧バージョンのデータのままこれらの関連機能を使用することもできます。

4.3 GUI のカスタマイズ

config.inc.php ファイルの次のパラメータを構成することができます。

- TL_TREE_KIND

  このパラメータは TestLink で使用されているツリー形式のメニューを構成します。値としては以下が設定されます。'LAYERSMENU' 'DTREE' 'JTREE' LAYERSMENU がデフォルト値です。JTREE が最もよいパフォーマンスが期待できます。その他の値では、最後にカーソルが当たっていた場所を保存できます。

- $g_fckeditor_toolbar
  fckeditor ツールバーの定義をします。このパラメータにより fckeditor ツールバーを構成できます。より詳細な情報については fckeditor のホームページを参照してください。

- TL_TPL_CHARSET
  中国語のユーザのみに適用されます。この値を以下のように設定します。

  ```
  define('TL_TPL_CHARSET', 'gb2312');
  ```

  これにより正しい文字セットが設定されます。その他の言語ではこのパラメータを設定する必要はありません。

- HTML に対する入力の長さやサイズ等の設定をハードコーディングすることを避けるため、次のファイルに設定を保存してあります`<testlinkdir>/gui/templates/input\_dimensions.conf`

4.3.1 地域ごとの日付と時間設定

定義された全ての地域の日付と時間の表示フォーマットを設定できます。const.inc.php の次の配列変数を使用して構成します。

$g_locales_date_format $g_locales_timestamp_format デフォルトの構成は次のようになっています。

```
$g_locales_date_format = array(
    'en_GB' => "%d/%m/%Y",
    'it_IT' => "%d/%m/%Y",
    'es_AR' => "%d/%m/%Y",
    'es_ES' => "%d/%m/%Y",
    'de_DE' => "%d.%m.%Y",
    'fr_FR' => "%d/%m/%Y",
    'pt_BR' => "%d/%m/%Y",
);

$g_locales_timestamp_format = array(
     'en_GB' => "%d/%m/%Y %H:%M:%S",
     'it_IT' => "%d/%m/%Y %H:%M:%S",
     'es_AR' => "%d/%m/%Y %H:%M:%S",
     'es_ES' => "%d/%m/%Y %H:%M:%S",
     'de_DE' => "%d.%m.%Y %H:%M:%S",
     'fr_FR' => "%d/%m/%Y %H:%M:%S",
     'pt_BR' => "%d/%m/%Y %H:%M:%S",
);
```

これらの配列にない地域は次の値が使用されます$g_date_format, $g_timestamp_format

フォーマット例:

```
$g_date_format ="%d/%m/%Y";
$g_timestamp_format = "%d/%m/%Y%H:%M:%S";
```

4.3.2 カスケーディング・スタイル・シート

TestLink の外観はカスケーディング・スタイル・シート (CSS) を独自に作成することで変更できます。このためには以下を変更する必要があります。

- ログイン/ログアウトページの CSS
  define('TL_LOGIN_CSS','gui/css/tl_login.css');
- 主に使われる CSS
  define('TL_TESTLINK_CSS','gui/css/testlink.css');
- レポートで使われる CSS
  define('TL_DOC_BASIC_CSS','gui/css/tl_doc_basic.css');

注意: CSS へのパスは TestLink のインストールディレクトリからの相対パスになります。
独自の CSS を使用する場合には以下の手順で作業を進めます。

1. gui ディレクトリ内に新しいディレクトリを作成します。例： gui/css/my_css/
2. TestLink で使われているオリジナルのファイルを 1) で作成したディレクトリにコピーします。必要に応じて名前を変更します。
3. 2) でコピーしたファイルを編集します。
4. config.inc.php を編集します。

```
// Original configuration
//define('TL_LOGIN_CSS','gui/css/tl_login.css');
//define('TL_TESTLINK_CSS','gui/css/testlink.css');
//define('TL_DOC_BASIC_CSS','gui/css/tl_doc_basic.css');
define('TL_LOGIN_CSS','gui/css/my_css/tl_login_acqua.css');
define('TL_TESTLINK_CSS','gui/css/my_css/testlink_acqua.css');
define('TL_DOC_BASIC_CSS','gui/css/my_css/tl_doc_basic.css');
```

4.3.3 Smarty テンプレートを使用する

別の方法でユーザインターフェースを変更したい場合は Smarty テンプレートを開発する方法があります。
const.inc.php の次の構成配列を定義します
$g_tpl 定義は次のエントリーと共に行います

```
$g_tpl['tcView']
$g_tpl['tcSearchView']
$g_tpl['tcEdit']
$g_tpl['tcNew']
$g_tpl['execSetResults']
```

これらのテンプレートは更新による上書きの危険性を避けるため、TestLink オリジナルのテンプレートとは違う名前で作成してください。
重要：これらの構成設定は TestLink 全ページに準備されているわけではありません。
標準の構成：

```
$g_tpl['tcView'] = "tcView.tpl";
$g_tpl['tcSearchView'] = "tcSearchView.tpl";
$g_tpl['tcEdit'] = "tcEdit.tpl";
$g_tpl['tcNew'] = "tcNew.tpl";
$g_tpl['execSetResults'] = "execSetResults.tpl";
```

5　ストリングファイルの場所

各言語ごとに strings.txt ファイルが保存されているフォルダが存在します。

- `<testlinkdir>/locale/de_DE/strings.txt`
- `<testlinkdir>/locale/de_DE/custom_strings.txt`
- `<testlinkdir>/locale/en_GB/strings.txt` ... 他

オリジナルの翻訳された文字列を変更せずに別の文字列に変更する場合は custom_strings.txt を使用します。このファイルは対応する言語のフォルダに保存する必要があり、オリジナルの strings.txt と同じ書式と規則で記述されている必要もあります。（strings.txt 内でオリジナルの記述をコメントアウトする必要なく値を再定義できる場合に限ります。）

6 FAQ

以下に頻繁に発生する問題を掲載します。TestLink フォーラムも参照して下さい。

- 旧バージョンから更新したら、ログインできなくなったデータベースの文字セットが異なっています。バージョン 1.6 よりデフォルトの文字セットは UTF8 になっています。config.inc.php にて DB_SUPPORTS_UTF8 を FALSE に切り替えてみて下さい。
- ログインページの代わりに Smarty エラーが表示される Linux/Unix ユーザは temp ディレクトリに正しい権限が与えられているか確認して下さい。デフォルトの temp ディレクトリ:`<testlinkdir>/gui/template_c/`

A　Mantis バグ管理システムと接続する

A.1　概要

TestLink とバグ管理システムの接続は以下のような特徴をもちます

- TestLink とバグ管理システム間の通信がデータベースのテーブルを通して行われます
- TestLink からバグ管理システムにデータを送信することはできませんが、関数の呼び出しによってデータを受信することができるようになります

全ての構成が完了した後は、ユーザは次の操作を行います。

1. 実行中のテストが失敗となります
2. ユーザは結果を保存します
3. ユーザはバグを報告するためのバグ管理システムへのリンクをクリックします
4. バグ報告後、ユーザはバグ管理システムから発行された ID を記録しておきます
5. TestLink のテスト実行ページに戻り、上記で発行された ID を入力します
6. ユーザがテスト結果を保存すると TestLink はバグ管理システムから取得したデータを表示するようになります

A.2　構成例

環境： TestLink と Mantis は同じ Web サーバにインストールされているとします

Mantis URL	http://calypso/mantis
TestLink URL	http://calypso/testlink
Mantis データベース名	mantis_bt
Mantis にアクセスする MySQL ユーザ名/パスワード	mantis_bt_user / mantis_bt_password

A.2.1　ステップ 1 Mantis の構成

- Mantis への anonymous ログインを有効にします
- 全ての公開プロジェクトへの参照権限がアサインされている Mantis のユーザ（anonymous アカウント）を作成します

Mantis の構成ファイル（config_inc.php）に次の行を追加し、適切な値に変更します。（'dummy' を適切な anonymous アカウントに書き換えます）

```
# --- anonymous login -----------
# Allow anonymous login
$g_allow_anonymous_login = ON;
$g_anonymous_account = 'dummy';
```

A.2.2　ステップ 2 TestLink と Mantis をインターフェースで接続する

`<testlinkdir>/cfg/mantis.cfg.php` ファイルを編集します

A.2.3 ステップ 3 バグ管理システムとの接続を有効にする

config.inc.php ファイルから custom_config.inc.php ファイルに次の行をコピーします。

```
//----------------------------------------------------------------------------
/** [Bug Tracking systems] */
/**
 * TestLink uses bugtracking systems to check if displayed bugs resolved,
 * verified and closed bugs. If they are it will strike through them
 *
 * NO : no bug tracking system integration
 * BUGZILLA : edit configuration in TL\_ABS\_PATH/cfg/bugzilla.cfg.php
 * MANTIS : edit configuration in TL\_ABS\_PATH/cfg/mantis.cfg.php
 * JIRA : edit configuration in TL\_ABS\_PATH/cfg/jira.cfg.php
 * TRACKPLUS : edit configuration in TL\_ABS\_PATH/cfg/trackplus.cfg.php
 */
 $g_interface_bugs = 'NO';
```

custom_config.inc.php ファイルの次の行を$g_interface_bugs = 'NO'; から、次のように変更します

```
$g_interface_bugs = 'MANTIS';
```

A.3 チェックインターフェース

構成を完了すると、テスト実行画面にバグを追加するためのアイコンが表示されます。バグを追加するときには次のチェックが実行されます。

- バグ管理システムに対象のバグ ID は存在しているか
- バグ ID のフォーマットは適切か

校正履歴

#	説明	日付	著者
1.0	Initial creation of the document in DocXML	2005/03/12	A. Morsing
1.1	Corrected title, updated structure and added new sections.	2005/04/12	M. Havlat
1.2	Added some words for MySQL 4.1, UTF8 support	2005/06/27	A. Morsing
1.3	Updated automatic installation part	2005/09/12	F. Mancardi
1.4	Updated for TL 1.6.; added configuration parameters; restructured (created pre-installation steps section); corrected layout; added phpMyAdmin steps description	2005/09/13	M. Havlat
2.0	Converted to OO2 format; added DB Charset update explanation from Kevin	2005/12/04	M. Havlat
2.1	Corrected layout for export to HTML and PDF	2005/12/11	M. Havlat
2.2	Some small changes	2005/12/17	A. Morsing
2.3	Minor layout and grammar update	2006/02/14	M. Havlat
2.4	Updated for TL 1.7	2006/11/17	M. Havlat
2.5	Updated for TL 1.7; restructured; merged BTS case; layout update (prepare for 1,7,0 release)	2007/09/13	M. Havlat