レポート

レポートモジュールは、簡単かつ強力なレポーティング機能を提供します。 この機能によって、 Trac
データベースのチケット情報を取得することができます。

TracReports ではレポートの形式を定義するための方法として、独自フォーマットではなく、 SQL の `SELECT` 文を使用することにしました。

> Note: 現在の形式のレポートモジュールは、 Trac
> 開発チームでデータベースのスキーマにあわせる作業が必要になるという深刻な制限事項があるため、段階的に廃止します。より柔軟性に富みユーザビリティに
> クエリモジュール
> が代替手段として提供されます。どこかの時点でレポートモジュールを完全に削除することが出来るように、クエリモジュールで実現できないレポートがある間
>
> 以下の trac.ini のように無効化するだけで、レポートモジュールをクエリモジュールで完全に置き換えることができます:
>
> ```
> [components]
> trac.ticket.report.* = disabled
> ```
>
> これによって、ナビゲーションバーの "チケットを見る" (英語版では "View Tickets")
> でのデフォルトのハンドラがクエリモジュールになります。もし可能ならば、この設定を有効にして、レポート機能がなくなることによって生じる不都合を報告
>
> ほとんど確実に httpd を再起動する必要があるでしょう。

レポートは以下の基本的なパーツから構成されます:

- ID -- ユニークな (連番の) 識別子
- レポート名 (Title) -- レポートのタイトル
- 説明 (Description) -- WikiFormatting で記述された、レポートの説明。
- レポート本体 (Report Body) -- 後に述べるフォーマットで規定された、レポートクエリの結果。
- フッタ (Footer) -- レポート本体を異なる形式でダウンロードするためのリンク。

ソートの並び順変更

単純なレポート (特にグループ化されていないもの) では、カラムのヘッダをクリックすれば、そのカラムでソートすることが出来ます。

カラムのヘッダがハイパーリンク (赤)
になっていれば、クリックすることでそのカラムでのソートができます。並び順を逆にするには、もう一度クリックします。

レポートの番号を変更する

レポートの ID を変更する必要がある場合があるかもしれませんが、おそらくレポート自体を編集する方がベターです。というのも Trac
のデータベースを変更する必要があるからです。 report 表は以下のようなスキーマとなっています (0.10 相当):

- id integer PRIMARY KEY
- author text
- title text
- query text
- description text

ID を変更すると レポート一覧 (Available Reports) での表示順と番号、レポートのパーマリンクが変更されます。以下のような SQL を実行すると ID
が変更されます:

```
update report set id=5 where id=3;
```

メンテナンス結果、データベースの一貫性を保つ必要があることに留意してください (例えば ID はユニークでなければなりませんし、 SQLite
などデータベースの上限値を超えることはできません)。

チケットをナビゲート

レポートクエリ結果の 1 チケットをクリックするとそのチケットが表示されるでしょう。表示されたチケットのメインメニューバーのすぐ下にある
次のチケット (英語版では Next Ticket) または 前のチケット (英語版では Previous Ticket)
リンクをクリックすることによって他のチケットに移動するか、レポートに戻る (英語版では Back to Report)
リンクをクリックしてレポートページに戻ることができます。

あなたは安全にチケットを編集することができます。またチケットの編集結果を保存した後で、 次のチケット/前のチケット/レポートに戻る
(英語版では Next/Previous/Back to Report)
のリンクを使用して結果を行き来することが可能です。しかし、あなたがチケットへの操作を終えてレポートに戻るときに、どのチケットが変更されたかのヒントは
(カスタムクエリについては TracQuery#NavigatingTickets を参照して下さい) 。 (0.11 以降)

ダウンロードできるフォーマット

通常表示される HTML でのビューの加え、レポートはいろいろな形式で使用することができます。
レポートページの一番下に、利用可能なデータ形式の一覧があります。望む形式のリンクをクリックすれば、
その形式でのレポートをダウンロードすることができます。

カンマ区切りテキスト - CSV (Comma Separated Values)

1 レコードを 1 行として、各カラムをカンマ (',') で区切ったプレーンテキストとしてダウンロードできます。 Note: CSV
形式を保つため、各カラムのデータに改行文字やカンマがある場合、その位置で切り取られます。

タブ区切り

CSV と似ていますが、水平タブ文字 (/ t) で区切られる点が違います。

RSS - XML コンテンツ配信

全てのレポートは、 XML/RSS 2.0 での配信が可能です。 RSS フィードを購読するにはページ下部にある、オレンジ色の 'XML'
アイコンをクリックしてください。 Trac での RSS 対応についての一般的な情報は、 TracRss に記述しています。

カスタムレポートを作成する

カスタムレポートを作成するためには、 SQL を楽に書ける程度の知識が必要です。

レポートは基本的に、 Trac が実行できる形式の、名前がついた特定 SQL です。 レポートに指定された SQL は、直接 Web
インタフェースから閲覧したり、 作成したりできます。

通常のレポートは、 'ticket' 表に対する、カラムの選択や、ソート指定を伴った SELECT 文となります。

Ticket 表のカラム

ticket 表は、以下のカラムを持ちます:

- id -- チケットID
- type -- チケット分類
- time -- 登録日時
- changetime -- 最終更新日時
- component -- コンポーネント
- severity -- 重要度
- priority -- 優先度
- owner -- 担当者
- reporter -- 報告者
- cc -- 関係者
- version -- バージョン
- milestone -- マイルストーン
- status -- ステータス
- resolution -- 解決方法
- summary -- チケットの概要

- description -- チケットについての完全な説明
- keywords -- キーワード

各カラムに対応する属性の詳細な説明は、 TracTickets に記述しています。

優先度順、登録日時順の全未解決チケット

例: 優先度順、登録日時順の全未解決チケット

```
SELECT id AS ticket, status, severity, priority, owner,
      time as created, summary FROM ticket
 WHERE status IN ('new', 'assigned', 'reopened')
 ORDER BY priority, time
```

上級トピック: 動的変数の使用

レポートに汎用性を持たせる手段として、 動的変数 をレポート SQL で使用する方法があります。
簡単に言うと、動的変数とは、クエリを実行する前に置き換えられる 特別な 文字列のことです。

クエリで動的変数を使う方法

動的変数を使うためのシンタックスは単純です。 '$' に続いて、大文字で変数名となる語を挿入してください。

例:

```
SELECT id AS ticket,summary FROM ticket WHERE priority=$PRIORITY
```

レポート閲覧時、 $PRIORITY に値を当てはめるためには、レポートの URL に引数として変数を与えてください。この変数名に '$' を入れてはいけません。

例:

```
 http://trac.edgewall.org/reports/14?PRIORITY=high
```

複数の値を使用する場合、各値を '&' で区切ります。

例:

```
 http://trac.edgewall.org/reports/14?PRIORITY=high&SEVERITY=critical
```

特殊な定数

実用的なレポートのために、定義済みの動的変数が用意されています。これらは URL に値を設定しなくても、自動的に値が割り当てられます。

- $USER -- ログインに使用したユーザ名。

例 (私が担当になっているチケット一覧):

```
SELECT id AS ticket,summary FROM ticket WHERE owner=$USER
```

上級トピック: 表示形式のカスタマイズ

Trac には、レイアウトのカスタマイズや、グルーピング、ユーザ定義の CSS 利用などによる
もっと複雑なレポートの作成も可能です。このようなレポートを作成するには、 Trac
のレポートエンジンが出力を制御するためのステートメントを含む、特別な SQL を使用します。

特別なカラム

レポートを整形するため、 TracReports はクエリの結果から '特定の' カラム名を 探します。このような '特定の' 名前で、最終的なレポートのレイアウトやスタイルが 処理され、変更されます。

自動的に整形されるカラム名

- ticket -- チケットの ID が入っているカラムで使用します。該当する ID のカラムにハイパーリンクされます。 (訳注: summary というカラム名もチケットにハイパーリンクされます。日本語版では ■■ でもリンクします。)
- created, modified, date, time -- 日付や時刻に整形されます。 (訳注: datetime という列名にすると日時で整形されます。日本語版では ■■ で終わるカラムは time に、 ■■ で終わるカラムは date に、 ■■ で終わるカラムは datetime に、それぞれ整形されます。)
- description -- チケットの説明が入っているカラムで使用します。 Wiki エンジンで処理されます。 (訳注: 日本語版では ■■ でも整形されます。)

例:

```
SELECT id as ticket, created, status, summary FROM ticket
```

整形されるカラムのカスタマイズ

カラム名の前後に 2 つのアンダースコアがついている場合 (例: __color__) は、 整形用のヒント として扱われ、レコードの整形が行われます。

- __group__ -- 指定されたカラムで、表示がグループ化されます。各グループは、それぞれセクションヘッダとクエリ結果の表を持ちます。
  __color__ -- 1 から 5
  の数値である必要があります。値によって、あらかじめ定義された色付けが行われます。一般的な使用法は、優先度別の色付けです。
  デフォルトの色付け: Color 1 Color 2 Color 3 Color 4 Color 5
- __style__ -- CSS 形式でレコードを整形できます。

例: マイルストーン別未解決チケット (優先度別色付け)

```
SELECT p.value AS __color__,
    t.milestone AS __group__,
    (CASE owner WHEN 'daniel' THEN 'font-weight: bold; background: red;' ELSE '' END) AS __style__,
      t.id AS ticket, summary
 FROM ticket t,enum p
 WHERE t.status IN ('new', 'assigned', 'reopened')
   AND p.name=t.priority AND p.type='priority'
 ORDER BY t.milestone, p.value, t.severity, t.time
```

Note: ticket 表の優先度に対応する数値は、 enum 表を結合することで 取り出しています。

行単位のレイアウト変更

デフォルトでは、全てのカラムで1行を使い、上記の指定がされていれば、 フォーマットされた形式で HTML に表示されます。それだけでなく、これから挙げる指定によって、複数行にわたってのレイアウトを行うことができます。

- column_ -- 改行。 カラム名の語尾にアンダースコア ('_') を付与した場合、以降のカラムは次の行で表示されます。
- _column_ -- 全行表示。 カラム名の前後にアンダースコア ('_') を付与した場合、そのカラムは続く行で全てのカラム幅を使って表示されます。
- _column -- データを非表示にする。 カラム名の語頭にアンダースコア ('_') を付与した場合、 HTML 出力では非表示になります。これは (CSV や RSS のような) 別フォーマットでのダウンロード時にだけ見たい情報であるときに使います。

例: アクティブなチケットを、マイルストーンでグループ化し、優先度で色付け、チケットの説明を multi-line レイアウトでリスト表示する

```
SELECT p.value AS __color__,
      t.milestone AS __group__,
      (CASE owner
         WHEN 'daniel' THEN 'font-weight: bold; background: red;'
         ELSE '' END) AS __style__,
      t.id AS ticket, summary AS summary_,              -- ## ■■■■■■■■
      component,version, severity, milestone, status, owner,
      time AS created, changetime AS modified,          -- ## ■■■■■■■■
      description AS _description_,                     -- ## ■■■■■■■■■■
      changetime AS _changetime, reporter AS _reporter -- ## HTML ■■■■■■■■■■
```

```
FROM ticket t,enum p
WHERE t.status IN ('new', 'assigned', 'reopened')
  AND p.name=t.priority AND p.type='priority'
ORDER BY t.milestone, p.value, t.severity, t.time
```

カスタムフィールドをレポートで使用する

チケットにカスタムフィールドを追加した場合(バージョン 0.8 以降の機能。 TracTicketsCustomFields 参照)、カスタムフィールドを含む SQL クエリを書くことができます。 ticket_custom テーブルを join をする必要がありますが、これは取り立てて簡単というわけではありません。

追加のフィールドを trac.ini に宣言する 前に 、チケットがデータベースに存在する場合、 ticket_custom テーブルには関連するデータを持たないことになります。これに起因する問題を回避するためには SQL の "LEFT OUTER JOIN" 節を使用してください。

Note レポートの追加、編集をするボタンを表示するためには権限を設定する必要があります。

See also: TracTickets, TracQuery, TracGuide, Query Language Understood by SQLite