

---

---

## ★ブログ記事のオブジェクト (MT::Entry) の操作

---

---

ブログを構成するオブジェクトの中で、ブログ記事 (MT::Entry) はもっともよく操作するオブジェクトです。この節では、ブログ記事のオブジェクトを操作する方法を解説します。

### ■ ブログ記事の情報をオブジェクトに読み込む

まず、ブログ記事の情報をオブジェクトに読み込む方法を解説します。ブログのオブジェクト (MT::Blog) と同様に、読み込みはMT::Entryクラスのloadメソッドで行います。

#### ● MT::Entryクラスを使うことを宣言する

プログラムの中でブログ記事のオブジェクトを操作する場合、その前に以下の一文を入れて、MT::Entryクラスを使うことを宣言します。

```
use MT::Entry;
```

#### ● loadメソッドの書き方

MT::Entryクラスに限らず、Movable Typeの各種の情報をオブジェクトに読み込む際には、それぞれのクラスのloadメソッドを使います。loadメソッドの基本的な書き方は、以下のようになります。

##### ①書き方その1

```
クラス名->load(オブジェクトのID)
```

##### ①書き方その2

```
クラス名->load(検索条件, パラメータ)
```

「クラス名」には、読み込みたいオブジェクトのクラス名を指定します。たとえば、ブログ記事をオブジェクトに読み込むなら、クラス名は「MT::Entry」とします。

1つのオブジェクトだけを読み込む場合は、前述の1つ目の書き方を 사용합니다。この場合、loadメソッドのパラメータとして、そのオブジェクトのIDを指定します。一方、複数のオブジェクトを一度に読み込む場合は、2つ目の書き方を 사용합니다。この場合は、検索条件等を指定して、読み込むオブジェクトを検索します。

「検索条件」では、主に「〇〇プロパティの値が□□である」という形で、検索するオブジェクトの条件を指定します。プロパティ名と検索する値の組を、ハッシュのキーと値として表わし、そのハッシュのリファレンスを「検索条件」の部分に指定します。ブログ記事をオブジェクトに読み込む場合、表2.3のようなプロパティを指定することができます。

1つのハッシュに複数の条件を指定することもできます。その場合、それらすべての条件を満たすブログ記事が読み込まれます。

また、「パラメータ」の部分には、並べ替えの方法や、読み込むオブジェクトの数など、オブジェクトを読み込む際のパラメータを指定します。パラメータの名前と、そのパラメータに渡す値の組を、ハッシュのキーと値の組として表わします。そして、そのハッシュのリファレンスを渡します。

表2.3 ブログ記事のオブジェクトの主なプロパティ

プロパティ	内容
id	ブログ記事のID
blog_id	ブログ記事が属するブログのID
author_id	ブログ記事を書いたユーザーのID
status	ブログ記事の公開状態 (下書き／公開など)
authored_on	ブログ記事の日時 (ブログ記事の編集画面の「公開日」で指定した日時)
created_on	ブログ記事を始めて保存した日時
modified_on	ブログ記事を最後に更新した日時
title	ブログ記事のタイトル
text	ブログ記事の本文
text_more	ブログ記事の追記 (ブログ記事の編集画面の「続き」の欄に入力した内容)
excerpt	ブログ記事の概要
comment_count	ブログ記事についてのコメントの数
ping_count	ブログ記事についてのトラックバックの数
basename	ブログ記事のファイル名
allow_comments	ブログ記事にコメントをつけられるかどうか
allow_pings	ブログ記事にトラックバックをつけられるかどうか

## ●特定のブログのブログ記事を読み込む

「検索条件」と「パラメータ」の具体的な指定方法の例をいくつか紹介します。まず、特定のブログに属するブログ記事を読み込む方法から解説します。この場合、loadメソッドの検索条件として、以下のようなハッシュリファレンスを渡します。なお、loadメソッドの「パラメータ」の部分は指定しません。

```
{ blog_id => ブログのID }
```

たとえば、IDが1番のブログから、ブログ記事をすべて読み込みたいとします。そして、読み込まれたブログ記事のオブジェクト群を、配列@entriesに割り当てたいとします。この場合、loadメソッドを以下のように書きます。

```
@entries = MT::Entry->load({ blog_id => 1 });
```

## ●公開されているブログ記事のみ読み込む

ブログ記事を読み込んでその情報を一般向けに出力する場合、未公開や日時指定になっているブログ記事の情報は、通常は出力すべきではありません。このようなときには、公開されているブログ記事だけを読み込むと良いです。

ブログ記事の公開状態は、statusプロパティで表されます。このプロパティは表2.4の定数のいずれかの値を取ります。

loadメソッドの検索条件に、「status => 公開状態」の条件を追加すると、その条件を満たすブログ記事だけを読み込むことができます。たとえば、IDが1番のブログから、公開されているブログ記事だけを読み込んで、配列@entriesに割り当てるには、リスト2.5のように書きます。

リスト2.5 IDが1番のブログから公開されているブログ記事を読み込む

```
01 @entries = MT::Entry->load({ blog_id => 1,  
02                               status => MT::Entry::RELEASE() });
```

表2.4 ブログ記事の公開状態を表す定数

定数	公開状態
MT::Entry::HOLD()	未公開 (下書き)
MT::Entry::RELEASE()	公開
MT::Entry::REVIEW()	レビュー
MT::Entry::FUTURE()	日時指定

## ●一定期間のブログ記事のみ読み込む

ブログでは、記事を月別や年別などの一定期間に分けて扱うこともあります。プログラムでこのような処理を行うには、一定の期間に公開されたブログ記事のみ読み込むことが必要になります。

ブログ記事の日付を表すプロパティとして、`authored_on` / `created_on` / `modified_on`の3つがあります。ブログ記事の編集画面で日付として入力する値は、`authored_on`プロパティに対応します。したがって、一定期間のブログ記事のみ読み込むには、一般的には`authored_on`プロパティの値で期間を指定します。

`load`メソッドの検索条件の書き方は、表2.5のようになります。開始日時 / 終了日時とも、14桁の数値 (年は4桁、月日時分秒はそれぞれ2桁) で表します。たとえば、2009年1月23日1時23分45秒は「20090123012345」と表します。

また、期間を指定して記事を読み込む場合、`load`メソッドのパラメータに「`range => { authored_on => 1 }`」または「`range_incl => { authored_on => 1 }`」を指定します。

`range`の場合、`authored_on`プロパティの値が開始日時より大きく、終了日時より小さい記事が読み込まれます。開始日時 / 終了日時はぴったりの記事は読み込まれません。一方、`range_incl`の場合、開始日時 / 終了日時にぴったりの記事も読み込まれます。

例えば、以下の条件をすべて満たすブログ記事を読み込んで、配列`@entries`に割り当てるとします。

- ①IDが1番のブログに属する
- ②公開されている
- ③2009年1月中 (2009年1月1日0時0分0秒～2009年1月31日23時59分59秒) に書いた

この場合、リスト2.6のように書きます。

表2.5 期間を指定してブログ記事を読み込む場合の検索条件の書き方

条件	loadメソッドの検索条件の書き方
開始日時と終了日時の間のブログ記事を読み込む	authored_on => [ 開始日時, 終了日時 ]
開始日時からのブログ記事を読み込む	authored_on => [ 開始日時 ]
終了日時より前のブログ記事を読み込む	authored_on => [ undef, 終了日時 ]

リスト2.6 日付を指定してブログ記事を読み込む

```
01 @entries = MT::Entry->load({ blog_id => 1,
02                               status => MT::Entry::RELEASE(),
03                               authored_on => [ 20090101000000,
04                                                  20090131235959 ] },
05                               { range_incl => { authored_on => 1 } });
```

## ● ブログ記事を並べ替える

ブログ記事を読み込む際に、プロパティの値によって並べ替えたいこともあります。たとえば、ブログ記事を日付の新しい順に並べ替えるような場合が相当します。

この場合、loadメソッドの「パラメータ」の部分に、「sort => 'プロパティ名」と「direction => 'ascendまたはdescend」の2つを追加します。ascend/descendはそれぞれ昇順/降順<sup>(※)</sup>を表します。

たとえば、IDが1番のブログ記事で、公開されているものを、日付の新しい順（降順）に読み込みたいとします。そして、読み込んだブログ記事を配列@entriesに割り当てるとします。これは、リスト2.7のように書くことができます。

リスト2.7 ブログ記事を並べ替えて読み込む

```
01 @entries = MT::Entry->load({ blog_id => 1,
02                               status => MT::Entry::RELEASE() },
03                               { sort => 'authored_on',
04                               direction => 'descend' });
```

(※) 昇順/降順

値が小さい順に並べ替えることを「昇順」と呼び、その逆に並べ替えることを「降順」と呼びます。たとえば、「1, 7, 4, 3, 6」の5つの数字を昇順で並べ替えると、「1, 3, 4, 6, 7」になります。一方、降順だと「7, 6, 4, 3, 1」になります。

## ●一部のブログ記事のみ読み込む

多数のブログ記事がある場合、その一部だけを読み込みたいこともあります。たとえば、「最近の10件のブログ記事を読み込む」といった場合です。このときは、loadメソッドの「パラメータ」の部分に、「offset => 記事数」や「limit => 記事数」を追加します。

offsetは、条件に合うブログ記事のうち、先頭から指定した件数を飛ばして、その次から読み込むのに使います。またlimitは、読み込むブログ記事の件数を指定するのに使います。

たとえば、以下のようにブログ記事を読み込んで、配列変数@entriesに割り当てたいとします。これはリスト2.8のように書くことができます。

- ①IDが1番のブログに属する
- ②日付の新しい順に並べ替える
- ③先頭の10件を飛ばし、その次の5件を読み込む

リスト2.8 一部のブログ記事のみ読み込む

```
01 @entries = MT::Entry->load({ blog_id => 1 },
02                               { sort => 'authored_on',
03                               direction => 'descend',
04                               offset => 10,
05                               limit => 5 });
```

## ■ ブログ記事の情報を得る

個々のブログ記事の情報（タイトルや本文など）は、ブログ記事のオブジェクトのプロパティから得ることができます。主なプロパティは、71ページの表2.3にあげたとおりです。

たとえば、変数\$entryにブログ記事のオブジェクトを割り当てたとします。この場合、「\$entry->title」は、そのブログ記事のタイトルを表します。また、「\$entry->text」はブログ記事の本文を表します。

## ● ブログ記事の情報を表示する例

リスト2.9は、IDが1番のブログから、公開されている最近のブログ記事を10件読み込んで、そのタイトルと日付を表組みで出力する例です（画面2.3）。主な行の内容は

以下の通りです。

①1行目 : use MT::Entry;

MT::Entryクラスを使うことを宣言します。

②5~9行目 : my @entries = MT::Entry->load(・・・);

loadメソッドを使って、IDが1番のブログから、公開されている最近のブログ記事を10件読み込む処理を行っています。このプログラムの中心的な処理です。

「IDが1番のブログ」「公開されている」という2つの条件から、loadメソッドの検索条件の部分は、「blog\_id => 1」と、「status => MT::Entry::RELEASE()」になります。

また、「最近の」ということから、ブログ記事は日付の新しい順（降順）に並べ替えます。したがって、loadメソッドの「パラメータ」の部分に、「sort => 'authored\_on'」と「direction => 'descend'」の記述が必要です。

さらに、読み込む記事数は10件ですので、loadメソッドの「パラメータ」の部分に、「limit => 10」も追加します。

③10行目 : for my \$entry (@entries) {

読み込んだブログ記事群（配列@entries）から、ブログ記事を一つずつ順に取り出し、繰り返しを行います。

④12行目 : print '<td>' . \$entry->id . '</td>';

ブログ記事（\$entry）のidプロパティを、tdタグで囲んで出力します。idプロパティはブログ記事のIDを表します。

⑤13~14行目

10行目と同じ要領で、ブログ記事のタイトル（titleプロパティ）と、日付（authored\_onプロパティ）を出力します。

なお、authored\_onプロパティは14桁の数値になっていて、そのままではやや読みづらいです。「MT::Util」というモジュールの関数を使うと、「2009年01月23日 12時34分56分」のように書式づけることもできます。これについては、213ページで解説します。

なお、リスト2.9を含むインデックステンプレートのサンプルファイルは、「part2」フォルダの「entry1.tpl」です。

リスト2.9 ブログ記事の情報を表示する例

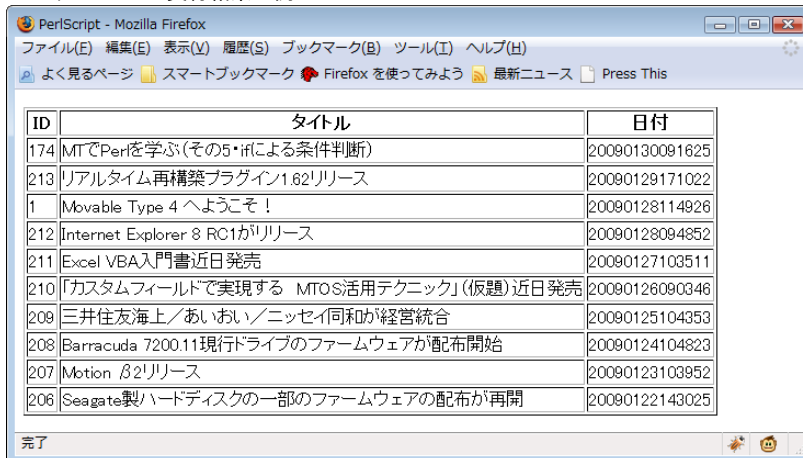
```
01 use MT::Entry;
02
03 print '<table border="1">';
04 print '<tr><th>ID</th><th>タイトル</th><th>日付</th></tr>';
```

```

05 my @entries = MT::Entry->load({ blog_id => 1,
06                               status => MT::Entry::RELEASE() },
07                               { sort => 'authored_on',
08                               direction => 'descend',
09                               limit => 10 });
10 for my $entry (@entries) {
11     print '<tr>';
12     print '<td>' . $entry->id . '</td>';
13     print '<td>' . $entry->title . '</td>';
14     print '<td>' . $entry->authored_on . '</td>';
15     print '</tr>';
16 }
17 print '</table>';

```

画面2.3 リスト2.9の実行結果の例



ID	タイトル	日付
174	MTでPerlを学ぶ(その5・ifによる条件判断)	20090130091825
213	リアルタイム再構築プラグイン1.62!リリース	20090129171022
1	Movable Type 4 へようこそ!	20090128114926
212	Internet Explorer 8 RC1がリリース	20090128094852
211	Excel VBA入門書近日発売	20090127103511
210	「カスタムフィールドで実現する MTO S活用テクニック」(仮題)近日発売	20090126090346
209	三井住友海上/あいおい/ニッセイ同和が経営統合	20090125104353
208	Barracuda 7200.11現行ドライブのファームウェアが配布開始	20090124104823
207	Motion 3.2リリース	20090123103952
206	Seagate製ハードディスクの一部のファームウェアの配布が再開	20090122143025

## ■前後のブログ記事を読み込む

あるブログ記事を基準にして、日付順で前後の記事を得たい、ということもあります。これは、ブログ記事のオブジェクトの「next」と「previous」というメソッドで得ることができます。

たとえば、あるブログ記事を変数\$entryに割り当てているとします。この場合、以下のようにすると、変数\$nextは1つ次のブログ記事を指すオブジェクトになります。

```
$next = $entry->next;
```