

目次

1. スキーマを作成する。.....	2
2. スキーマを登録する。.....	5
3. エントリを定義する。.....	6
4. エントリを投稿する。.....	9
5. エントリを表示する(デザイン定義)。.....	11
6. エントリー一覧を表示する。.....	18

チュートリアル: 簡単な日記帳を作る

最初のステップとして、簡単な日記帳を作ってみましょう。

「件名」と「本文」があるだけの簡単なものです。

1. スキーマを作成する。

では、まず「日記帳」のスキーマを定義します。

スキーマは「XML ファイル」として作成します。

コラム: XML って何？

XML について分からない方はご心配なく。HTML と書き方はほとんど同じです。

```
<name>名前</name>
```

のように、<タグ名>データ</タグ名>の形式になっているテキストデータのことです。

理解しておく必要がある HTML との違いは次の2点だけです。

① 閉じタグが無いタグの場合について。

HTML の IMG タグ (例:) のように「閉じタグが無い」タグは、 のように書かなければいけません。「>」の前に「/」をつけます。

も、
になります。もちろん、閉じタグとセットにして「
</br>」と書けば OK ですが、ちょっと冗長な書き方ですね。
だけの方がシンプルです。

② 属性は必ず""で囲う

HTML では、<form action=post > のように書けましたが、属性値は必ず""で囲って、<form action="post"> と書かなければいけません。

気をつける必要があるのはこれぐらいです。まずは、とりあえずやってみましょう！

【！】 注意

CMS Designer では、スキーマファイルやデザインファイルなどの XML ファイルは UTF-8 の文字コード (漢字コード) で保存する必要があります。UTF-8 でファイルを保存するには、Windows 標準添付の「メモ帳でファイルを保存する際に「文字コード」から「UTF-8」を選んで保存すれば OK です。個人的には「秀丸エディタ」をお勧めします。

では、さっそくスキーマファイルを作ってみましょう。

まず、スキーマ名を決めます。半角英数字で名前をつけます。

日記帳ですから、素直に「diary」としましょう。

次のようなファイルを作成します。

diary.schema.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<schema name="diary" caption="日記帳" >
  <data name="title" type="text" caption="件名" />
  <data name="body" type="textarea" caption="本文" />
</schema>
```

ファイル名は、「スキーマ名」+「.schema.xml」の形式になります。拡張子は「xml」です。

そして、保存します。UTF-8 で保存するのを忘れなく！

これを見ていて、何か思い出しませんか・・・？ そう、HTML の form の書き方にそっくりです。form タグが schema タグになり、input タグが data タグになっているような感じです。

とりあえず一つ一つ、見ていきましょう。

1 行目: <?xml version="1.0" encoding="UTF-8"?>

これは、おまじないだと思ってください。何も考えずにこのまま入力すれば OK です。

2 行目: <schema name="diary" caption="日記帳" >

2 行目は、**schema** というタグを使って「ここからスキーマ定義が始まるよ」と示しています。HTML の form タグで「ここからフォームが始まるよ」というのと似ていますね。

ついでにここで、スキーマ名とその日本語名を定義しています。**name** 属性に先ほど決めたスキーマ名「diary」、**caption** 属性に「日記帳」と設定します。

3 行目: `<data name="title" type="text" caption="件名" />`

これが、スキーマの項目の定義になります。このように **data** タグを使って項目を定義します。

ここでは「件名」を定義しています。件名ですから、項目名を分かりやすく「title」としましょう(スキーマ名と同じく半角英数で決めます)。

type 属性には「text」を設定しています。これは、HTML の input タグの `type="text"` と似ていて、この項目が1行テキストボックスによる文字列の入力項目であることを指定します。

件名	<input type="text" value="件名です。"/>	←こんなイメージです。
----	------------------------------------	-------------

caption の「件名」は、テキストボックスの横に表示されるラベルです。入力するときに分かりやすいよう、日本語で設定します。例えば「タイトル」とか「見出し」でもいいかもしれませんね。

4 行目: `<data name="body" type="textarea" caption="本文" />`

もう一つ入力項目を設定しています。「本文」ですね。日記帳ですから件名だけでなく本文もないといけません。本文には項目名として(なんでも構いませんがここでは)「body」という名前をつけましょう。

type には、件名と同じく「text」でもいいのですが、1行テキストボックスでは本文を入力するには狭すぎるので、「textarea」と設定します。**type="textarea"** は、この項目が複数行にわたる入力を受け付ける広いテキストボックス項目である事を指定します。

本文	<div>本文はある程度大きな領域でないと入力しづらいですね。 改行も入れたいですし。</div>	←こんなイメージです。
----	---	-------------

5 行目: `</schema>`

最後に、「スキーマ定義はここまでです」という閉じタグを書いて完了です。

どうですか？

慣れるまでは戸惑うこともあるかもしれませんが、formタグを書いていると思えば特に難しくないはずです。

2. スキーマを登録する。

作成したスキーマファイル(ここでは"diary.schema.xml")をサーバにアップロードします。

スキーマを登録する為のフォルダを作成して、そこにファイルを置きます。

まず、下のフォルダへ移動してください。

```
/public-html/cmsdesigner/config/schema/
```

このフォルダの下に、スキーマ名(ここでは"diary")でフォルダを新規作成してください。

```
/public-html/cmsdesigner/config/schema/diary/
```

そうしたら、先ほど作ったスキーマファイルをここにアップロードします。

```
/public-html/cmsdesigner/config/schema/diary/diary.schema.xml
```

これでOKです。パーミッションは特に指定する必要はありません。

3. エントリを定義する。

スキーマを定義しただけではまだ、エントリを投稿していくことはできません。

スキーマとは別に、「エントリ定義」を行います。エントリ定義とは「このスキーマを使って、この場所に記事を保管していきますよ」という情報です。

スキーマ定義とエントリ定義が別になっていることで、例えば「日記」というスキーマを元に、「友達向けの日記」というエントリと「一般向け日記」というエントリを別々に管理することが可能になります。それぞれの日記毎に同じスキーマ定義をコピーする必要がなく、便利です。

「一般向け日記」のエントリ定義をしてみましょう。エントリ名を「public_diary」とします。エントリ保管用のフォルダを新規作成します。まず、下のフォルダへ移動してください。

```
/public-html/cmsdesigner/data/entry/
```

このフォルダの下に、エントリ名（ここでは“public_diary”）でフォルダを新規作成してください。

```
/public-html/cmsdesigner/data/entry/public_diary/ （※パーミッションを 707 に設定）
```

エントリ用フォルダ（ここでは“public_diary”）のパーミッションは“707”に設定してください。

次に、/public-html/cmsdesigner/config/site.config.xml ファイルをエディタで編集します。このファイルも UTF-8 の文字コードですので、UTF-8 対応のエディタで編集してください。

開いた状態では、次のようになっています。

site.config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<site>
  <manager>
    <user name="*****" password="*****" />
  </manager>
  <entries>
    <entry name="news1" schema="news" caption="新着情報" />
  </entries>
</site>
```

entry タグで定義されている部分が、エントリ定義です。それ以外は無視してOKです。

初期状態ではデモ用として「新着情報」というエントリが定義されています（太字の部分）。これはデモを確認した後なら消してしまって構いません。

ではここに、「public_diary」の定義を追加します。

site.config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<site>
  <manager>
    <user name="" password="" />
  </manager>
  <entries>
    <entry name="news1" schema="news" caption="新着情報" />
    <entry name="public_diary" schema="diary" caption="一般向け日記" />
  </entries>
</site>
```

やり方としては、新着情報のエントリ定義をコピーして、それぞれ

- ・name 属性に、エントリ名「public_diary」、
- ・schema 属性に、スキーマ名「diary」、
- ・caption 属性に、日本語での説明「一般向け日記」

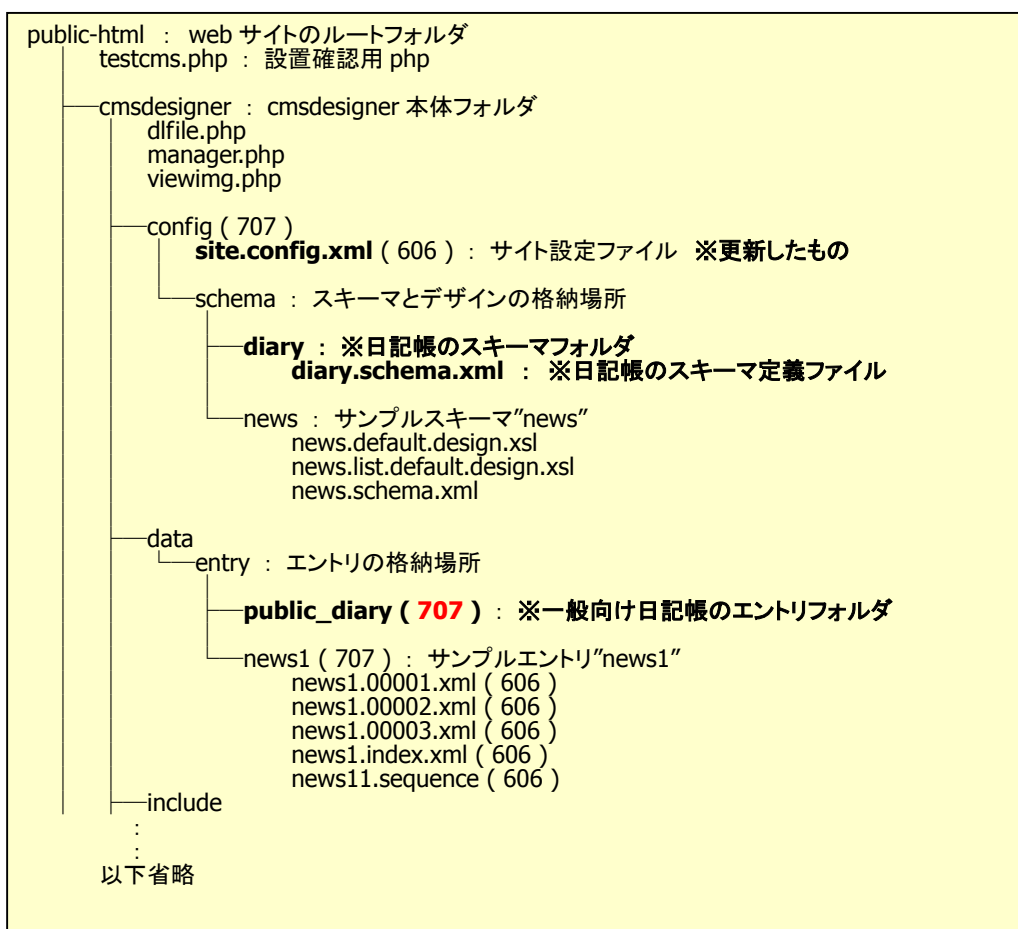
のように変更します。

保存して、サーバ上のファイルも更新してください。

これでエントリ定義は完了です。

ここでちょっと一休みして、現在の状況を確認してみましょう。

サーバ上のファイルは、次のページの図のようになっていますか？



太字の箇所が、今回変更した部分です。

それではコンテンツ管理へログインしてください。

いまくいけば、コンテンツ一覧に「一般向け日記」が表示されています。

CMS Designer コンテンツ管理

【注意事項】
 ※ ブラウザの「戻る」を使うと正常に動作しません。
 ※ 編集／削除／送信内容は、保存するまでは確定されません。

■コンテンツの編集
 編集するコンテンツを選んでください。

コンテンツ名	一覧へ
新着情報	一覧を表示
一般向け日記	一覧を表示

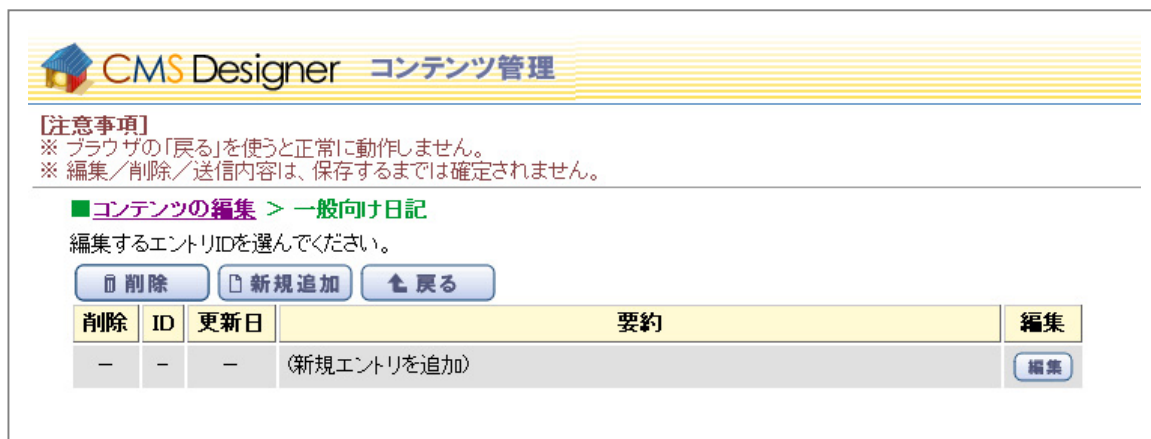
表示されていない場合は、site.config.xml をもう一度見直して、サーバへアップロードしなおしてください。

4. エントリを投稿する。

さあ、サイト管理者になったつもりで、エントリを投稿してみましょう。

コンテンツ管理画面から、「一般向け日記」の編集画面へ入ってください。

エントリ一覧には、まだ一件もデータが存在していません(新規に作ったエントリ定義ですから当たり前ですね)。




■コンテンツの編集 > 一般向け日記

編集するエントリIDを選んでください。

削除 新規追加 戻る

削除	ID	更新日	要約	編集
-	-	-	(新規エントリを追加)	編集

「新規追加」ボタンを押して、エントリの新規作成画面へ入ります。



■コンテンツの編集 > 一般向け日記 > (新規エントリ)


《現在の編集位置》
エントリルート

編集を保存 キャンセル

説明	入力欄
件名	<input type="text"/>
本文	<div><div></div><div></div></div>

スキーマ定義で定義した通りの画面が表示されています。ここに好きな文字列を入力して、「編集を保存」を押せば、入力内容が新規エントリとして保存されます。

試しに、次のように入力してみましょう。



[注意事項]
 ※ ブラウザの「戻る」を使うと正常に動作しません。
 ※ 編集／削除／送信内容は、保存するまでは確定されません。

■ **コンテンツの編集** > **一般向け日記** > **(新規エントリ)**

《現在の編集位置》
 エントリルート

説明	入力欄
件名	<input type="text" value="件名です。"/>
本文	<input type="text" value="本文です。今日は何もありませんでした。おそまつ。"/>

この通りでなくても構いませんが、入力したら「編集を保存」を押し、「一般向け日記」のリンク又は「キャンセル」ボタンを押して前の画面(エントリー一覧)に戻ってください。



[注意事項]
 ※ ブラウザの「戻る」を使うと正常に動作しません。
 ※ 編集／削除／送信内容は、保存するまでは確定されません。

■ **コンテンツの編集** > **一般向け日記**

編集するエントリIDを選んでください。

削除	ID	更新日	要約	編集
<input type="checkbox"/>	00001	2004/11/30 11:41:14	件名です。	<input type="button" value="編集"/>
-	-	-	(新規エントリを追加)	<input type="button" value="編集"/>

エントリー一覧に、先ほど保存したエントリが追加されています。

IDは「00001」となっています。このIDは後で画面表示の際に必要となるので、覚えておいてください。

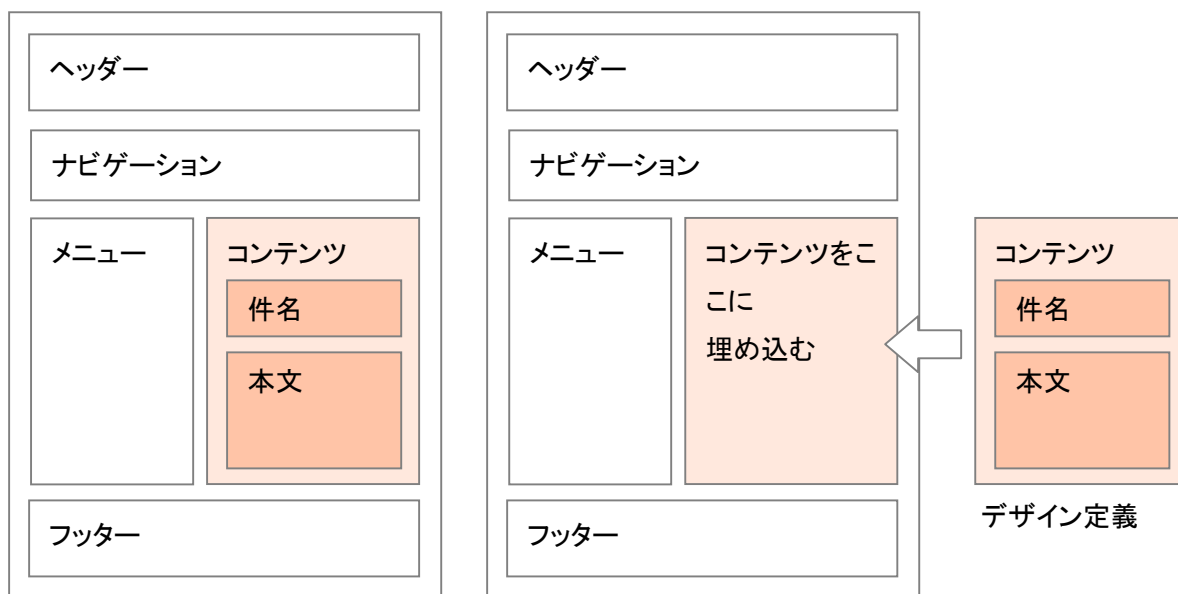
他にもいくらかでも投稿できますので、がんがん投稿してみてください。

もしこれまでの流れで、変な Error や Warning が大量に出たりした場合は、スキーマ定義とエントリ定義の手順をもういちど見直してください。特に、エントリフォルダのパーミッションを 707 にするのを忘れないようにしてください。707 でダメな場合は、777 も試してみてください。

5. エントリーを表示する（デザイン定義）。

エントリーをいくら投稿しても、どこかにそれを公開しなければ意味がありません。
それでは最後の肝である、デザイン定義をやってみましょう。

一般的な CMS では、コンテンツの出力先として HTML ページを丸ごと1つの「テンプレート」として定義しますが、CMS Designer では所謂「テンプレート」に相当するものと、それを表示する場所である HTML ページは別になります。



一般的な CMS の「テンプレート」

本体の HTML に直接コンテンツのデザインも記述している。

CMS Designer の「デザイン定義」と埋め込み先 HTML

本体の HTML には「埋め込む場所」だけ記述しておき、
コンテンツのデザインは別途記述している。

これは一見、複雑になったように見えますが、このようにコンテンツ部分を本体から切り離して扱えることで、逆に本体の HTML はシンプルになります。また、別の HTML に同じコンテンツを埋め込みたい場合など、デザイン定義をそのまま使いまわすことができます。

ではまず、デザイン定義からやってみましょう。その後、埋め込み先 HTML を簡単に作って表示させてみます。

ここからは「XSL」という、業界標準の XML 加工言語を使います。

細かい部分は気にせず、流れを追ってみてください。

/public-html/cmsdesigner/config/schema/diary/ のフォルダに、以下のファイルを作成します。
ファイル名は、「スキーマ名」+「.」+「デザイン名」+「.design.xml」の形式になります。
ここではデザイン名として"default"と付けています。拡張子は"xml"ではなく"xsl"です。

diary.default.design.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="EUC-JP" omit-xml-declaration="yes" />

  <xsl:template match="/entry">
    <xsl:value-of select="title" />
    <xsl:value-of select="body" />
  </xsl:template>

</xsl:stylesheet>
```

このファイルも、これまでの XML ファイルと同様に、UTF-8 の文字コードで保存してください。
内容の説明に移ります。重要なのは太字の部分だけです。
他の部分は基本的に定型句なので、このままコピーしてください。

```
<xsl:value-of select="title" />
```

この行で、「件名」を出力しています。<xsl:value-of select="xxx" />と書くと、「xxx というデータを見つけたらここに内容を出力せよ」という意味になります。

スキーマ定義で件名に"title"という名前を付けたことを覚えていますか？

よってこれは、「件名の内容をここに出力せよ」と指示していることになります。

```
<xsl:value-of select="body" />
```

title の出力が分かれば、これも分かりますね。件名には"body"と名づけましたから、ここで body を出力するよう、指示しています。

項目が増えたら、これも増やしていけばいい訳です。簡単ですね。

ちなみに、例えば件名の出力指示をここに書かなければ、データ上に件名が存在しても、出力されないだけでエラーにはなりません。

出来上がったこのファイルを(しつこいようですが、UTF-8 で保存してください)、サーバーの以下の場所にアップロードします。

/public-html/cmsdesigner/config/schema/diary/**diary.default.design.xml**

このように、デザイン定義ファイルはスキーマと同じ場所に保管していきます。

では次に、このデザインを出力する先の画面を作成します。

いろいろとデザインを凝りたいところですが、今は話をシンプルにする為、簡単な内容にします。

nikki.php

```
<?php require( "cmsdesigner/include/view.php.inc" ); ?>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=euc-jp">
  <title>テストページです。</title>
</head>
<body>
テストページ<br>
<table border="1">
  <tr>
    <td>
      メニュー部分
    </td>
    <td>
      <?php cmsview::entry( "public_diary", "default" ) ?>
    </td>
  </tr>
</table>
</body>
</html>
```

出力先の画面は、php として実行する必要がある為、拡張子を「php」にします。拡張子以外のファイル名は自由につけてください。

太字の部分に注目してください。

まず一行目、

```
<?php require( "cmsdesigner/include/view.php.inc" ); ?>
```

これは定型句です。

出力先の全ての画面の先頭に埋め込んでください。

```
<meta http-equiv="Content-Type" content="text/html; charset=euc-jp">
```

charset を「euc-jp」にします。出力先となる画面は、全て EUC-JP の文字コードで保存してください。スキーマ定義やデザイン定義など、各種 XML ファイルは UTF-8 ですが、php ファイルは EUC-JP で保存する必要があります。

ちょっとややこしいですが、ようはこういう事です。

ファイルを編集して保存する際の文字コード

- ・拡張子が「xml」又は「xsl」 → UTF-8 で保存。
- ・拡張子が「php」 → EUC-JP で保存。

HTML に目を向けると、2列のテーブルを作って、左側に「メニュー」、右側に今回の日記エントリを表示するようにしています。

埋め込み部分を見ると、次のようになっています。

```
<?php cmsview::entry( "public_diary", "default" ) ?>
```

埋め込み内容の指示形式

```
<?php cmsview::entry( "エントリー名", "デザイン名" ) ?>
```

エントリ1件を表示するには、cmsview::entry 命令を使います。

つまりここで、エントリ「public_diary」を、デザイン「default」を使って出力せよ、という指定をしています。

このファイルは、サイトのルートフォルダ直下にアップロードしてください。

public-html/nikki.php

さあ、では、nikki.php を呼び出してみましょう。

「5. 4 エントリを投稿する」の最後に投稿したエントリの ID を覚えていますか？

“00001”ですね。指定したエントリを表示するには、次のように書きます。

指定エントリを画面に表示する URL

http://あなたのサイト/nikki.php?eid=エントリ ID

例えばあなたのウェブサイトが http://www.hogehoge.com/ ならば、

http://www.hogehoge.com/nikki.php?eid=00001

が、先ほど投稿したエントリを表示する為の URL になります。

成功すると、次のような画面が表示されるはずです。

テストページ

メニュー部分 件名です。本文です。今日は何もありませんでした。おそまつ。

エラーが出てしまった場合は、ファイル名や URL を再度確認してみてください。

ところで、先ほどの出力画面を見ると、件名の後にすぐ本文が続いてしまっています。

それは、デザイン定義で単に title と body をそのまま並べて出力している為です。

例えば間に BR タグを挟む、などをすれば改行が入って少し見やすくなりますが、もう少しそれっぽく、DIV タグでデザインしてみましょう。

デザイン定義ファイルを再度開いてください。

diary.default.design.xsl

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="EUC-JP" omit-xml-declaration="yes" />

  <xsl:template match="/entry">
    <xsl:value-of select="title" />
    <xsl:value-of select="body" />
  </xsl:template>

</xsl:stylesheet>
```

この太字周辺を、次のようにしてみます。

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="EUC-JP" omit-xml-declaration="yes" />

  <xsl:template match="/entry">
    <div class="diary_title">
      <xsl:value-of select="title" />
    </div>
    <div class="diary_body">
      <xsl:value-of select="body" />
    </div>
  </xsl:template>

</xsl:stylesheet>
```

ファイルを保存して、サーバー上のファイルも更新してください。

デザイン定義では、<xsl:ほげほげ>で始まるタグ以外のタグは、そのままの形で出力されます。よって、この結果、

```
<div class="diary_title">
  件名です。
</div>
<div class="diary_body">
  本文です。今日は何もありませんでした。おそまつ。
</div>
```

というテキストが埋め込まれることになります。

class 指定だけでも CSS がどこかで定義されていないといけないので、作成しましょう。

diary.default.css

```
.diary_title {
  background : #2E8B57;
  padding : 5px;
  margin : 0px;
  font-weight : bold;
  color : #FFFFFF;
}
.diary_body {
  margin : 0px;
  border : 2px solid #2E8B57;
  padding : 5px;
}
```

こんな感じでしょうか。この辺はデザイナーさんの方がお詳しいでしょうから、お好きにどうぞ。

CSS ファイルの文字コードについては CMS Designer では CSS は管理外ですので、好きな文字コードで保存して頂いて構いませんが、php に合わせて EUC-JP にしておくとか何とか便利かもしれません。

サーバーへの保存先は、

public_html/diary.default.css

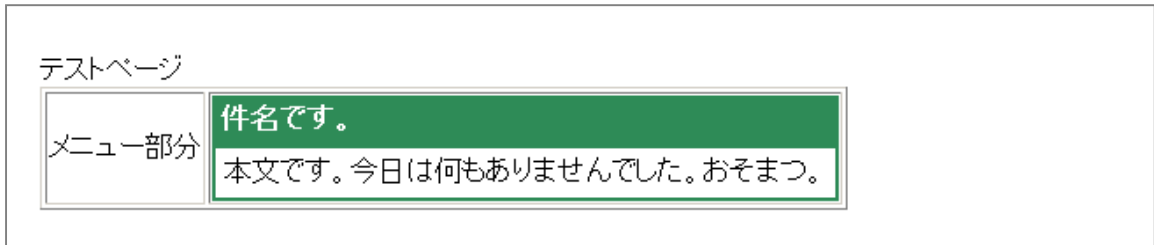
としておきます。

最後にこの CSS を読み込むように、nikki.php に CSS へのリンクを張りましょう。

nikki.php

```
<?php require( "cmsdesigner/include/view.php.inc" ); ?>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=euc-jp">
  <link rel="stylesheet" href="diary.default.css" type="text/css">
  <title>テストページです。</title>
</head>
:
省略
```


nikki.php をサーバーへアップロードしたら、再度、
http://あなたのサイト/nikki.php?eid=エントリ ID
へアクセスしてみてください。



上記のような画面が表示されていますか？
なんとなくそれっぽくなってきましたね。

このように、基本的にはデザイン定義ファイルには DIV タグや TABLE タグなどのレイアウト情報だけを含めるようにして、色やフォントの指定は CSS 側に持つようにします。

場合によっては、デザイン定義ファイルには全て DIV タグだけを含めるようにして、レイアウトも CSS でやってしまうという方法もあります。

尚、表示がおかしければ、ブラウザからソースを開いてみて、以下の事を確認してください。

(1) **<div class="diary_title">などの div タグが出力されていない。**

→デザイン定義ファイル(diary.default.design.xml)が更新されていません。

(2) **CSS へのリンクが張られていない。**

→nikki.php が更新されていません。

(3) (1)も(2)も **OK**なのに、表示がおかしい。

→CSS ファイルが正しくないか、正しいファイル名でアップロードされていません。

6. エントリー一覧を表示する。

先ほどのデザイン定義は、「エントリー1件分」を表示するデザインでした。

しかし、エントリーはどんどん増えていく為、どこかに「エントリー一覧」も必要です。

まさか、閲覧者に「nikki.php?eid=12345」などを入力してもらうわけにもいきません。

CMS Designer では、エントリーの一覧を表示する際も、先ほどのエントリー1件分のデザイン方法と同じやり方でデザインを定義できます。

diary.list.default.design.xsl

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="EUC-JP" omit-xml-declaration="yes" />

  <xsl:template match="/entrylist/entry">
    <div class="diary_title">
      <xsl:value-of select="title" />
    </div>
    <div class="diary_body">
      <xsl:value-of select="body" />
    </div>
  </xsl:template>

</xsl:stylesheet>
```

これが、エントリー一覧のデザイン定義ファイルです。

エントリー1件分とほとんど変わっていませんね。

違うところはファイル名ぐらいですが、一つだけ違うところがあります。

```
<xsl:template match="/entrylist/entry">
```

この部分です。前回では"/entry"だけだったのに、"/entrylist/entry"になっています。

ここは深く考えず、「エントリー一覧の場合はこのようにする」と覚えてください。

エントリー一覧のデザイン定義ファイルのファイル名は、

「スキーマ名」+「.list.」+「デザイン名」+「.design.xsl」

の形式になります。

ここではデザイン名として"default"と付けています。拡張子は同じく"xml"ではなく"xsl"です。

このファイルを

public_html/cmsdesigner/config/schema/diary/

にアップロードしてください。

次に、一覧画面を作成します。

ちょっと手抜きをして、エントリ1件分の画面とほとんど同じ画面にします。

nikkilist.php

```
<?php require( "cmsdesigner/include/view.php.inc" ); ?>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=euc-jp">
  <link rel="stylesheet" href="diary.default.css" type="text/css">
  <title>テストページです。</title>
</head>
<body>
テストページ(一覧)<br>
<table border="1">
  <tr>
    <td>
      メニュー部分
    </td>
    <td>
      <?php cmsview::listpage( "public_diary", "default" ) ?>
    </td>
  </tr>
</table>
</body>
</html>
```

エントリ1件分の時と違うのは、太字の部分だけです。

埋め込み内容の指示形式

```
<?php cmsview::listpage( "エントリ名", "デザイン名" ) ?>
```

エントリー一覧の表示には、cmsview::listpage 命令を使います。

使い方は cmsview::entry 命令と同じです。

これを、nikki.php と同じ場所にアップロードします。

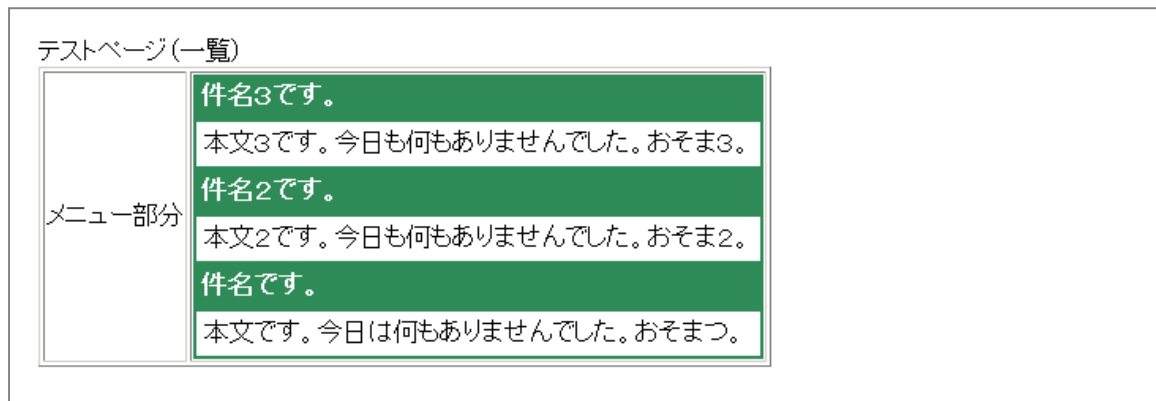
public-html/nikkilist.php

例えばあなたのウェブサイトが <http://www.hogehoge.com/> ならば、

<http://www.hogehoge.com/nikkilist.php>

が、エントリー一覧を表示する為の URL になります。

さあ、表示してみてください。



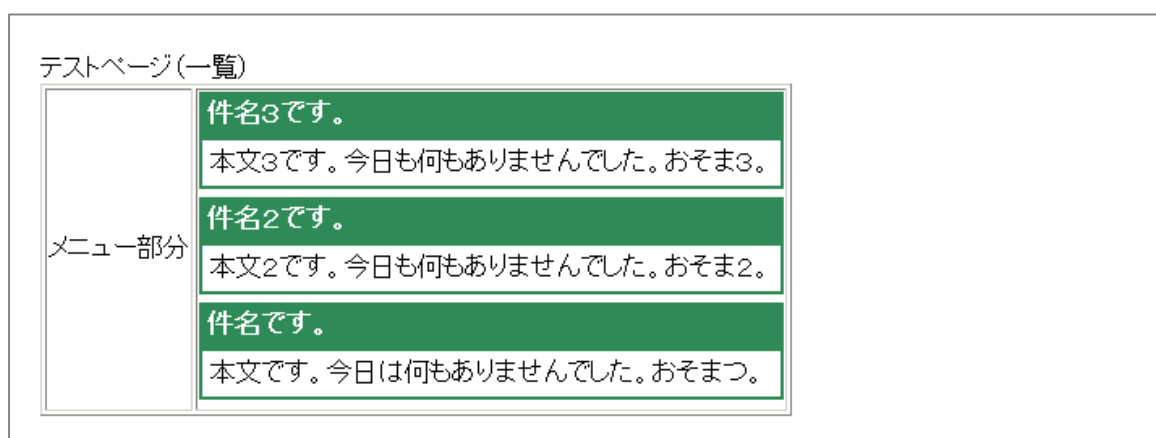
コンテンツ管理画面にてエントリーを何件登録したかで、表示内容が異なると思います。

エントリー1件用に作った CSS を使いまわしている為、私の CSS ではエントリー同士がくっついてしまっています。少し調整してみましょう。

diary.default.css

```
.diary_title {  
  background : #2E8B57;  
  padding : 5px;  
  margin : 0px;  
  font-weight : bold;  
  color : #FFFFFF;  
}  
.diary_body {  
  margin : 0px 0px 5px 0px;  
  border : 2px solid #2E8B57;  
  padding : 5px;  
}
```

下の margin を 5px 取りました(太字の部分)。



少し見やすくなりましたね。

実際のケースでは、一覧に本文まで全て表示することより、「件名のみ」を一覧で出すことの方が多いと思います。件名にはリンクを張って、個別の日記のエントリ画面へジャンプさせる、という方法を実現して、このチュートリアルを終わることにしましょう。

```
<a href="nikki.php?eid=00001">件名 1 です。</a>
```

のような出力を得られれば、一覧から個別のエントリへジャンプさせることができます。
まずは、件名と本文を表示するようにしていた一覧用のデザイン定義を以下のように修正します。

diary.list.default.design.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="EUC-JP" omit-xml-declaration="yes" />

  <xsl:template match="/entrylist/entry">
    <a href="nikki.php?eid={@id}">
      <xsl:value-of select="title" />
    </a>
    <br />
  </xsl:template>

</xsl:stylesheet>
```

body の出力部分を削除して、title を A タグで囲うようにしました。

```
<a href="nikki.php?eid={@id}">
```

この行は、A タグをそのまま出力しています。

ここで「{@id}」というのは、このエントリの ID を表します。「00001」とか「00002」などの値が入ります。
{@id} は CMS Designer が自動的に用意するデータなので、デザイナーは何も用意しなくても使えます。

例えば、<xsl:value-of select="@id" /> と書けば、ID そのものを出力させることができます。
しかし今回は A タグの href 属性の中にそれを出力したいので、その方法は使えません。タグの途中で別のタグが存在することは、XML では許されない為です。

タグの属性の内部にエントリのデータを出力したい時は、

{ データ名 }

のように、{} でデータ名を囲います。つまり、

```
<a href="nikki.php?eid={@id}" />
```

は、

```
<a href="nikki.php?eid=00001" />
```

として出力されます（エントリ ID が「00001」の場合）。

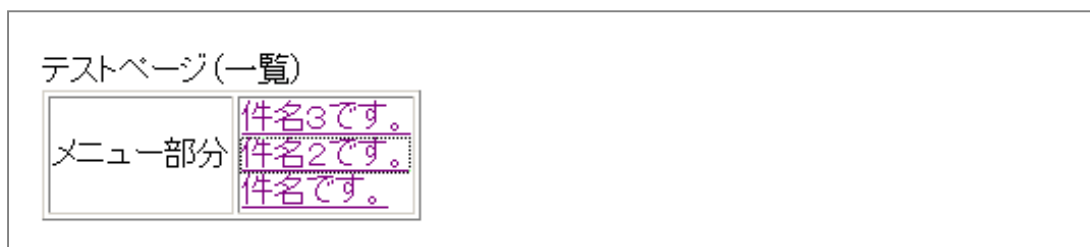
```
<br />
```

次の行の
は、BR タグを出力しています。なぜ
ではなく
かというと、XML では開始タグと終了タグは必ず1セットになっていなければならない為です。

BR のように終了タグが存在しない場合は、開始タグと終了タグを一つにまとめた
という書き方をする必要があります。

他の場合でも全て同じですので、XML のルールとしてそのまま覚えておけば OK です。BR 以外だと、IMG タグや INPUT タグなどがそれにあたります。

このデザイン定義を保存してサーバ上の定義も更新し、一覧を再表示してみましょう。



望みどおりの出力が得られました。各件名をクリックすると、それぞれの日記内容を表示します。

エラー等が表示されてしまった場合は、デザイン定義の修正内容をもういちどよく確かめてみてください(半角であるべき箇所が全角になっていたりしませんか?)。

見栄えが寂しいですが、その辺はデザイナーさんの担当です。デザインをいじる方法は既にご存知のはずですから、いくらでも見栄え良くカスタマイズしてください。

以上でこのチュートリアルは終わりです。

より詳細についてはリファレンスマニュアルを見て頂くか、別のチュートリアルをお試ください。