

00

Chapter1

ディレクトリ構成

Chapter2

設置手順と
動作検証

Chapter3

多言語対応
の方法

Chapter4

動作モードと
ログ記録

Chapter5

その他

設置マニュアル

Ver. 1.0

■ディレクトリ構成

ダウンロードしたファイルを解凍すると下記のようなディレクトリ構成になっています。
ツール上の設定により、ディレクトリ構成が多少異なります。

●多言語対応なし、通常型の場合

- output
 - application –アプリケーションディレクトリ
 - controllers –コントローラファイル保管用
 - lib –ライブラリファイル保管用
 - log –エラーログ保管用
 - models –モデルファイル保管用
 - views –ビューファイル保管用
 - html –公開ディレクトリ
 - css –スタイルシートファイル保管用
 - js –JavaScriptファイル保管用
 - sql –テーブル定義SQLファイル保管用

●多言語対応なし、モジュール型の場合

- output
 - application –アプリケーションディレクトリ
 - lib –ライブラリファイル保管用
 - log –エラーログ保管用
 - modules –モジュールディレクトリ
 - default –デフォルトモジュール
 - controllers –コントローラ保管用
 - models –モデルファイル保管用
 - xxx –XXXモジュール※
 - controllers –コントローラ保管用
 - models –モデルファイル保管用
 - views –ビューファイル保管用
 - html –公開ディレクトリ
 - css –スタイルシートファイル保管用
 - js –JavaScriptファイル保管用
 - sql –テーブル定義SQLファイル保管用

※「xxx」は指定した「プログラムID」になります。

●多言語対応あり、通常型の場合

- output
 - application –アプリケーションディレクトリ
 - controllers –コントローラファイル保管用
 - language –翻訳ファイル保管用
 - lib –ライブラリファイル保管用
 - log –エラーログ保管用
 - models –モデルファイル保管用
 - views
 - ja –日本語のビューファイル保管用※1
 - html –公開ディレクトリ
 - css –スタイルシートファイル保管用
 - js –JavaScriptファイル保管用
 - sql –テーブル定義SQLファイル保管用

●多言語対応あり、モジュール型の場合

- output
 - application –アプリケーションディレクトリ
 - language –翻訳ファイル保管用
 - lib –ライブラリファイル保管用
 - log –エラーログ保管用
 - modules –モジュールディレクトリ
 - default –デフォルトモジュール
 - controllers –コントローラ保管用
 - models –モデルファイル保管用
 - xxx –XXXモジュール※2
 - controllers –コントローラ保管用
 - models –モデルファイル保管用
 - views
 - ja –日本語のビューファイル保管用※1
 - html –公開ディレクトリ
 - css –スタイルシートファイル保管用
 - js –JavaScriptファイル保管用
 - sql –テーブル定義SQLファイル保管用

※1 対応する言語ごとにビューファイルの保管場所が必要です。詳しくは後述します。

※2「xxx」は指定した「プログラムID」になります。

■ 設置手順

CodeSaberで生成したプログラムは下記の手順で設置します。

- ①「output.zip」を解凍する。
- ②データベースにテーブルを作成する。
- ③設定ファイルを編集する。
- ④ファイルをサーバにアップロードする。
- ⑤Zend Frameworkを入手する。
- ⑥Zend Frameworkをアップロードする。

①「output.zip」を解凍する

CodeSaberからダウンロードした「output.zip」ファイルを解凍してください。
前述のようなディレクトリ構成で解凍されます。

②データベースにテーブルを作成する

データベースに必要なテーブルを作成します。
CodeSaberが出力したファイルの中の「sql」というディレクトリに「xxx.sql」というテーブル定義が記述されたテキストファイルがありますので、これをphpMyAdminなどのデータベース管理ソフト上で実行してください。

※多言語対応のプログラムとして生成した場合はSQLファイルが複数になる場合がありますが、その場合はすべてのファイルを順番に実行してください。

③設定ファイルを編集する

利用するサーバ環境に合わせて、いくつかのファイルを編集する必要があります。
下記を参考にして編集してください。

●システム設定ファイル(application/lib/config.ini)

・3行目

fqdn ……サイトのFQDNです。「http://」を除いたドメイン名を入力します。

例) fqdn = “www.example.com”

・4行目

basePath ……ドキュメントルートのパス設定です。例えばトップページのURLが「http://www.example.com/index.html」の場合は「/」ですが、トップページが「http://www.example.com/~hoge/index.html」などのようにドメインとファイル名の間に何か入る場合はその文字「/~hoge/」を記述します。

例1) basePath = “/”

例2) basePath = “/~hoge/”

<注> 前後の「/」は忘れないようにしてください。動作不良の原因となります。

・14行目

database.host …データベースサーバのホスト情報です。データベースが別のサーバで稼働している場合はホスト名かIPアドレスを記述してください。

- 例1) database.host = "localhost"
- 例2) database.host = "db.example.com"
- 例3) database.host = "111.222.333.444"

・15行目

database.user …データベースの接続ユーザ名を記述します。

- 例) database.user = "dbuser"

・16行目

database.password …データベースの接続パスワードを記述します。

- 例) database.password = "password"

・17行目

database.schema …データベースのスキーマを記述します。

- 例) database.schema = "hoge"

・18行目

database.charset …データベース側の文字コードを記述します。CodeSaberで生成されたプログラムはデータベース側の文字コードに合わせて文字コード変換を行うように設計されていますが、Zend Frameworkの内部処理はUTF-8のため、なるべくデータベース側の文字コードをUTF-8に合わせることをお勧めします。

●htaccessファイル(html/.htaccess)

・1行目～5行目

PHPの設定を行う部分ですが、PHPの設定に問題がある場合はプログラム実行時に自動チェックする機能が働きますのでとりあえずそのままにしておいてください。

・8行目

RewriteBase …Zend Frameworkが利用するRewriteエンジンの設定です。先ほどのconfig.iniファイルの「basePath」と同じ内容を記述してください。

※稀にRewriteBaseの設定が不要のサーバがあります。どうしても正常に動作しない場合はこの行の行頭に「#」をつけてコメントアウトしてみてください。

●フロントコントローラ(html/index.php)

・12行目

システムディレクトリの設定です。このファイルから見た「application」ディレクトリの相対パスを記述します。通常は変更する必要はありませんが、サーバ環境の関係でディレクトリ構成を変更する場合は、こちらの記述も変更するようにしてください。

④ファイルをサーバにアップロードする

ファイルをサーバにアップロードします。「output」ディレクトリ以下のファイルをそのままアップロードしてください。「html」ディレクトリはサーバの設定により名前が異なる場合があります(「www」「public_html」など)ので、あらかじめサーバ管理者に確認しておいてください。

また、下記のディレクトリはアップロードする必要はありません。

・「sql」ディレクトリ

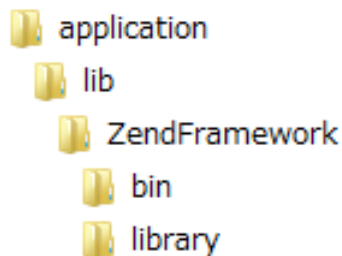
⑤Zend Frameworkを入手する

Zend Framework本体を入手します。CodeSaberのプログラム生成完了画面か、Zend Frameworkのオフィシャルサイトからダウンロードしてください。特に理由がなければ最新版のZend Frameworkをダウンロードするようにしましょう。

Zend Frameworkには「Full」版と「minimal」版がありますが、「minimal」版の方で問題ありません。

⑥Zend Frameworkをアップロードする

入手したZend Frameworkを解凍し、ディレクトリ名を「ZendFramework」にリネームしてサーバの「application/lib/」内にアップロードします。アップロード後のディレクトリ構成が下記のようになることを確認してください。



| | |
|----------|------------|
| Chapter1 | ディレクトリ構成 |
| Chapter2 | 設置手順と動作検証 |
| Chapter3 | 多言語対応の方法 |
| Chapter4 | 動作モードとログ記録 |
| Chapter5 | その他 |

■動作検証

CodeSaberで生成したプログラムは全く手を加えなくても一通りの動作をするようになっています。ブラウザでアクセスする際の各機能のURLは下記の通りです。

プログラムIDを「User」として生成した場合の例を紹介します。

●通常型の場合

- ・登録機能
http://www.example.com/User/regist
- ・一覧画面
http://www.example.com/User/list
- ・詳細画面※
http://www.example.com/User/detail
- ・編集画面※
http://www.example.com/User/edit
- ・削除画面※
http://www.exmaple.com/User/delete-conf

●モジュール型の場合

- ・登録機能
http://www.example.com/user/Regist
- ・一覧画面
http://www.example.com/user/Info/list
- ・詳細画面※
http://www.example.com/user/Info/detail
- ・編集画面※
http://www.example.com/user/Edit
- ・削除画面※
http://www.exmaple.com/user/Delete/confirm

※データベースのシーケンスをPOSTで受け取る必要があるため、直アクセスするとエラーになります。
デフォルトの状態では一覧画面経由でアクセスしてください。

以上でCodeSaberが生成したプログラムのインストールと動作検証は完了です。

CodeSaberが生成するプログラムは必要最低限のシンプルなコードです。
コントローラやモデルなどの各プログラム、ビューファイルをカスタマイズし、オリジナルのシステムを作り上げていってください。

■ 多言語対応の方法

多言語対応を「使用する」でプログラムを生成すると、多言語対応ができるように設計されたプログラムやデータベース構造で出力されます。CodeSaberが生成する多言語対応プログラムの構造について簡単に紹介します。

● 多言語対応の構造

多言語対応が必要なもの

<ビューファイル> <データベース内のデータ> <システムが出力するメッセージ>

それぞれの多言語化の方法

・ビューファイル・・・言語ごとにディレクトリを分けてビューファイルを用意する。

```

application/
├── views/
│   ├── ja/                ←日本語のビューディレクトリ
│   │   └── xxxx.html
│   ├── en/                ←英語のビューディレクトリ
│   │   └── xxxx.html
│   └── xx/                 ←xx言語のビューディレクトリ
│       └── xxxx.html

```

・データベース内のデータ・・・多言語化が必要なものと不要なものでテーブルを分けて管理する。

```

m_XXXX                ←多言語が不要な項目を格納
m_XXXX_disp           ←多言語が必要な項目を格納

```

・システムが出力するメッセージ・・・各言語ごとに翻訳ファイルを用意し、切り替えて表示させる。

```

application/
├── language/
│   ├── message-en.mo    ←英語の翻訳ファイル
│   └── message-xx.mo    ←xx言語の翻訳ファイル

```

●多言語対応の手順

CodeSaberが生成する多言語対応プログラムは、デフォルトでは日本語のみに対応しています。将来的に多言語対応できるようにしておきたい場合など、当初は日本語しか利用しないという場合は特に何もする必要はありません。

多言語に対応させるための手順を、英語に対応させる例で紹介します。

①言語定義をデータベースに登録する

言語の定義は「m_lang」というテーブルで管理されています。CodeSaberが生成したテーブル定義では日本語の定義のみが登録されている状態です。その情報を参考に英語の言語定義を登録します。

●m_langテーブル

| lang_id | locale | mail_charset | money_symbol |
|---------|--------|--------------|--------------|
| 1 | ja | ISO-2022-JP | 円 |
| 2 | en | ISO-8859-1 | \$ |

「locale」はISO-3166-1で定義されている2桁の国別コードを設定します。詳細は「http://ja.wikipedia.org/wiki/ISO_3166-1」に記載されていますのでそちらを参照してください。

また、「mail_charset」はメールで使用する文字コードを設定します。主要な言語のメール文字コードは下記の通りです。

●主要な言語とメール文字コード

| | |
|-------|-------------|
| 日本語 | ISO-2022-JP |
| 中国語 | ISO-2022-CN |
| 韓国語 | ISO-2022-KR |
| 英語 | ISO-8859-1 |
| フランス語 | ISO-8859-1 |
| ドイツ語 | ISO-8859-1 |
| イタリア語 | ISO-8859-1 |
| スペイン語 | ISO-8859-1 |

②英語版のビューファイルを用意する。

英語表示用に英語版のビューファイルを用意します。日本語版のビューファイルをすべてコピーし、日本語部分を英語に編集してください。作成したビューファイルは多言語対応のルール通り、「application/views/」ディレクトリに「en」というディレクトリを作成して設置します。

※プログラムが記述されている部分(<?php ~ ?>)は削除したり変更したりしないように注意してください。プログラム部分が変更されると、正常に動作しなくなる可能性があります。

③メッセージの翻訳ファイルを作成する

エラーメッセージ等、システムが出力する文字列も多言語化する必要があります。CodeSaberが生成したファイルの中には、システムが出力するメッセージがまとめられた「application/language-xx.po」というファイルが含まれています。このファイルを編集して英語版のメッセージを作成します。

「language-xx.po」ファイルは「Gettext」フォーマットで作成されています。Gettextフォーマットの編集には「PoEdit」というソフトが便利です。「<http://www.poedit.net/download.php>」から無料でダウンロードできます。「PoEdit」の使い方についてはネット上に色々情報がありますので、そちらを参考に作成してください。

作成したファイルの拡張子は「.mo」になります。元ファイルの「xx」を対応する言語のロケールに書き換えて保存してください。今回は英語への対応なので「message-en.mo」というファイルになります。作成したファイルを下記の通りアップロードしましょう。

```
application/
├ language/
│   └ message-en.mo
```

以上で英語への対応準備は完了です。

●アクセス方法

CodeSaberが生成した多言語対応プログラムは、下記の3つの要素により表示言語の判定を行います。

- パラメータで指定されたロケール情報
- Cookieに保存されているロケール情報
- ブラウザの優先言語に設定されているロケール情報

※上のものほど優先度が高くなります

A. パラメータで指定する場合

Zend FrameworkのURLパラメータで言語を指定する方法です。URLの末尾に「/lang/xx」(英語なら「/lang/en」)を付加してアクセスしてください。

→通常型の場合のアクセス例

- ・登録機能
<http://www.example.com/User/regist/lang/en>
- ・一覧画面
<http://www.example.com/User/list/lang/en>

→モジュール型の場合

- ・登録機能
<http://www.example.com/user/Regist/index/lang/en>
- ・一覧画面
<http://www.example.com/user/Info/list/lang/en>

B. Cookieに保存されている設定を使用する場合

CodeSaberで生成されたプログラムには言語情報をCookieに保存する処理は入っていません。
一度パラメータで指定した言語を記憶させておきたいなどの場合は、Cookie に言語情報を書き込む処理を追加してください。

| キー | 値 |
|------|----|
| lang | xx |

「xx」の部分には設定する言語のロケールを指定してください。

C. ブラウザの優先言語の設定を使用する場合

前述の「A」「B」のいずれも設定されていなければ自動的にブラウザの優先言語の設定が利用されます。
優先言語の設定方法はブラウザにより異なりますので、ご利用中のブラウザの設定画面等で探してみてください。

以上の方法を使い分けることで、様々なケースに対応できるサイトが構築できます。
動作チェックにも利用できると思いますので、色々と試してみてください。

もしうまく表示言語が切り替わらない場合は多言語対応にする手順でうまく設定できていない部分がある可能性があります。
その場合は再度手順を確認してください。

■ 動作モードとログ記録

CodeSaberが生成するプログラムには、開発や運用をより便利にするため、「動作モード切替」と「ログ記録」の機能が搭載されています。

● 動作モードの種類

| 動作モード | 概要 |
|---------|--|
| デバッグモード | システムエラー発生時にエラー内容だけでなく、発生箇所をファイル名と行番号情報で表示する。 |
| リリースモード | システムエラー発生時にエラーが発生したことのみを伝えるメッセージを表示する。 |

動作モードは「システム設定ファイル」(application/lib/config.ini)の6行目で設定します。

- ・デバッグモードの場合
execMode = "debug"
- ・リリースモードの場合
execMode = "release"

● ログ記録

デバッグモードで表示されるエラー内容を「application/log/error.log」に発生日時情報付きで記録します。動作モードは「システム設定ファイル」(application/lib/config.ini)の7行目で設定します。

- ・ログ記録ありの場合
outErrorLog = "1"
- ・ログ記録なしの場合
outErrorLog = "0"

■システム設定ファイルの内容

これまでもシステム設定ファイルの編集について何度か出てきましたが、すべての設定内容を一覧で紹介します。

●システム設定ファイル(application/lib/config.ini)

| セクション | 項目名 | 設定内容 | デフォルト値 |
|------------|-------------------|----------------------|-------------|
| global | fqdn | FQDN | (なし) |
| | basePath | ベースパス | / |
| | cryptKey | 暗号化キー(画面遷移時用) | (ランダム値)※1 |
| | execMode | 動作モード(debug/release) | debug |
| | outErrorLog | ログ出力(0:なし/1:あり) | 0 |
| | multilingual | 多言語対応(0:なし/1:あり) | 1 |
| | defaultLocale | 言語判定不能時の言語設定 | ja |
| | listRows | 一覧表示の1ページあたりの行数 | 10 |
| | dispPageNum | ページ数表示の最大数 | 5 |
| | | | |
| datasource | database.type | データベースエンジンの種類(固定) | localhost |
| | database.host | データベースのホスト名 | (なし) |
| | database.user | データベースの接続ユーザ名 | (なし) |
| | database.password | データベースの接続パスワード | (なし) |
| | database.schema | データベースのスキーマ | (なし) |
| | database.charset | データベースの文字コード | utf-8 |
| | database.cryptKey | データベースの暗号化キー | (ランダム値)※1※2 |


※1 管理者ごとに異なります(作成したユーザの暗号化キーは共通です)

※2 フォームタイプが「パスワード」の項目はこのキーで暗号化されてデータベースに登録されます。
後からこのキーを変更すると、登録されているデータを復号化できなくなってしまいますので注意してください。

■ 特殊なサーバ環境での利用について

CodeSaberが生成したプログラムはアプリケーションディレクトリが公開ディレクトリと同階層にありますが、サーバ環境によっては公開ディレクトリと同階層にディレクトリを作成できない場合があります。

そのような場合は、下記の例のように公開ディレクトリ内にアプリケーションディレクトリをアップロードしてください。

| | |
|---|---------------------|
|  html | — 公開ディレクトリ |
|  css | — スタイルシートファイル保管用 |
|  js | — JavaScriptファイル保管用 |
|  private | — 非公開領域※ |
|  application | — アプリケーションディレクトリ |
|  controllers | — コントローラファイル保管用 |
|  lib | — ライブラリファイル保管用 |
|  ZendFramework | — Zend Framework本体 |
|  log | — ログファイル保管用 |
|  models | — モデルファイル保管用 |
|  views | — ビューファイル保管用 |

※「private」ディレクトリは公開ディレクトリ上にあるため、そのままでは非公開領域になりません。
「html/private/」直下に「.htaccess」ファイルを設置して非公開設定をする必要があります。

● 「html/private/.htaccess」ファイル

```
Order deny,allow
deny from all
```

上記ファイルをアップロード後、非公開領域のファイルにブラウザから直接アクセスできないことを十分確認しておいてください。非公開領域のファイルにアクセスできるとセキュリティホールの原因となります。