２００６年度
オープンソースソフトウェア活用基盤整備事業

「1CD Linux/UNIX リマスタリング
ツールの開発」
プラグイン開発説明書

更新履歴

修正日	修正内容
2007/02/21	設計書のプラグイン説明記事を分冊として作成

目次

1. はじめに

本書は EZ Tune LiveCD（以降「リマスタリングツール」）の説明書である。
リマスタリングツールのプラグインを開発するときに必要となる情報を記述する。

2. プラグイン仕様

2.1. プラグイン共通

2.1.1. プラグインの配置

リマスタリングツールのプラグインはプラグイン毎にプラグインの格納ディレクトリを作成し、その配下に資源を格納する。格納ディレクトリの名前はプラグイン本体のファイル名から拡張子を除いたものでなければならない（プラグイン本体のファイル名が「knoppix50.rb」の場合、そのファイルは「knoppix50/knoppix50.rb」となる）。

また、プラグイン毎の格納ディレクトリは Ruby のライブラリ検索パス(Ruby のデフォルトのライブラリイブラリディレクトリ、カレントディレクトリ、および、環境変数 RUBYLIB に設定されているディレクトリ）配下の「remastering/plugins 」ディレクトリに格納する。

例 1）KNOPPIX5.0 の場合の格納ディレクトリ

```
/usr/lib/ruby/1.8/remastering/plugins/knoppix50_ja/
        knoppix50.rb      ‥      KNOPPIX5.0 プラグイン本体
        knoppix50.jpg     ‥      画像ファイル
        knoppix50.glade   ‥      glade2 プロジェクトファイル
          ：
```

例 2）プラグイン格納ディレクトリの構成例

```
$ unset RUBYLIB   # 環境変数 RUBYLIB を削除した状態

  このときのプラグインは検索パスは
    ● /usr/lib/ruby/1.0/remastering/plugins/*
    ● . /remastering/plugins/*
  となる


$ export RUBYLIB=/home/knoppix/tool  # 環境変数 RUBYLIB を設定した状態

  このときのプラグインは検索パスは
    ● /usr/lib/ruby/1.0/remastering/plugins/*
    ● . /remastering/plugins/*
  これに加えて
    ● /home/knoppix/tool/remastering/plugins/*
  となる
```

リマスタリングツールがロードするファイルはプラグイン本体のみであるため、その他のファイルの名前ディレクトリは任意で構わない（上記の例で knoppix ディレクトリ配下にサブディレ

クトリを作成しても構わない）。

2.1.2. プラグインの名前

ファイル名	*プラグイン名*.rb（半角英数字のみ）
クラス名	プラグイン名の先頭１文字を大文字にした名前

2.1.3. プラグインの基底クラス

リマスタリングツールではプラグインの作成を支援するために、プラグインの基底クラスを用意する。
基底クラスはプラグインのタイプ毎に用意され、作成するプラグインに対応した基底クラスを継承するようにする。

以下にリマスタリングツールが用意するプラグイン基底クラスの一覧を示す。

項番	プラグインの型	基底クラス名	require で取り込むライブラリ名
1	OS	PluginBaseOs	remastering/pluginBaseOs
2	PACKAGE	PluginBasePackage	remastering/pluginBasePackage
3	MEDIA	PluginBaseMedia	remastering/pluginBaseMedia
4	TEST	PluginBaseTest	remastering/pluginBaseTest
5	EMULATOR	PluginBaseEmulator	remastering/pluginBaseEmulator
6	EXPORT	PluginBaseExport	remastering/pluginBaseExport

コード例

```
require 'remastering/pluginBaseOs'

class Knoppix5 < PluginBaseOs        マーカ部分が基底クラスの読み込みと継承を示す

      :

end
```

2.1.4. プラグインで定義するメソッド

項番	メソッド名	メソッド説明
1	initialize ［必須］	プラグインクラスのコンストラクタ。 プラグイン固有の初期化を実施する。 【形式】 void initialize(pluginDir) 【引数】 1. pluginDir プラグイン格納ディレクトリ 【復帰値】 なし
2	start_plugin ［必須］	プラグインロード時に一度呼び出されるバージョンチェック処理。 引数で渡されるリマスタリングツールのバージョンで動作可能か否かをチェックする。 動作不能な場合(false を返却した場合）、直ちにプラグイン終了処理である end_plugin メソッドが呼び出される。 【形式】 BOOL start_plugin(version) 【引数】 1. version リマスタリングツールのバージョン 【復帰値】 true プラグインは動作可能 false プラグインは動作不能
3	end_plugin ［必須］	リマスタリングツール終了時に一度呼び出される。 プラグイン固有の終了処理を実施する。 【形式】 void end_plugin() 【引数】 なし 【復帰値】 なし 【備考】 プラグインの基底クラスを使用している場合は省略可能。省略した場合、終了処理は何も実行しない。
4	get_plugin_type ［必須］	プラグインの種別を識別するためのメソッドである。 一度だけ呼び出しが行われる。 【形式】 String get_plugin_type() 【引数】 なし 【復帰値】 以下のいずれかの値を返却する。 “OS€35 OS プラグイン “MEDIA€35 メディアプラグイン “EXPORT€35 エクスポートプラグイン “PACKAGE€35パッケージプラグイン “EMULATOR€35エミュレータプラグイン “TEST€35 テストプラグイン 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、継承したプラグインのタイプが返却される。

項番	メソッド名	メソッド説明
5	get_plugin_name ［必須］	プラグインの名前を返却する。返却した名前は画面への表示に使用する。 【形式】 String get_plugin_name() 【引数】 なし 【復帰値】 プラグインの表示名 例）“QEMU€35 “KNOPPIX”など 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@display_name€3設定された値が返却される。
6	get_plugin_version ［必須］	プラグインのバージョンを返却する。 【形式】 String get_plugin_version() 【引数】 なし 【復帰値】 プラグインのバージョン 例）“1.0€35”V1.0”など 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@version€3設定された値が返却される。
6	have_about? ［必須］	バージョン情報ダイアログの有無を返却する。 trueを返却するとヘルプメニューに[*プラグイン表示名*について]が表示される。 【形式】 BOOL have_about?() 【引数】 なし 【復帰値】 true　バージョン情報ダイアログあり false　バージョン情報ダイアログなし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@have_about€3設定された値が返却される。
7	show_about ［任意］ ※ have_about?メソッドでtrueを返却する場合は必須	バージョン情報ダイアログを表示する。 【形式】 void show_about? 【引数】 なし 【復帰値】 なし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、以下に示す標準ダイアログが表示される。

項番	メソッド名	メソッド説明
		 なお、上記イメージ中の”コマンドテスト”部分はインスタンス変数“@display_name”、V1.0部分はインスタンス変数”@version、コピーライト情報はインスタンス変数”@copyright、著作権情報はインスタンス変数”@licenseからも取得される。 また、著作権情報、および、ライセンス情報は省略が可能であり、省略する場合は省略する情報に対応するインスタンス変数に nil を設定する。
8	have_option? ［必須］	オプション情報ダイアログの有無を返却する。 true を返却するとオプションメニューに[*プラグイン表示名*の設定]が表示される。 【形式】 BOOL have_option?() 【引数】 なし 【復帰値】 true　オプション情報ダイアログあり false　オプション情報ダイアログなし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@have_option”に設定された値が返却される。
9	show_option ［任意］ ※ have_option?メソッドで true を返却する場合は必須	バージョン情報ダイアログを表示する。 【形式】 void have_option?(window) 【引数】 なし 【復帰値】 なし
10	load_xml ［任意］	リマスタリング手順ファイルから自プラグインに対する情報を読み込む。 【形式】 void load_xml(root) 【引数】 root リマスタリング手順ファイルのルート要素(<RemasteringInfo>要素の REXML::Element インスタンス) 【復帰値】 なし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、リマスタリング手順を読み込んでも読み込んだ手順ファイルに対する処理は何も実施されない。
11	make_xml	リマスタリング手順ファイルに保存する自プラグイン情報を XML 化して返却する。

項番	メソッド名	メソッド説明
	［任意］	【形式】 REXML::Element make_xml() 【引数】 なし 【復帰値】 リマスタリング手順ファイルに書き込みを行う自プラグイン情報(REXML::Element インスタンス）。 返却したプラグイン情報はルート要素（<RemasteringInfo>）の子要素としてXML ファイルに書き出される。 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、自プラグインに対するリマスタリング手順は保存されない。
12	isModified? ［任意］ ※ リマスタリング手順を保存するプラグインは必須	リマスタリング手順の変更の有無を返却する。 新規にリマスタリングを実施している場合は起動時からの変更の有無を返却する。 リマスタリング手順ファイルを読み込んだ場合は、手順ファイル読み込み時からの変更の有無を返却する。 【形式】 BOOL isModified?() 【引数】 なし 【復帰値】 true　リマスタリング手順に変更あり false　リマスタリング手順に変更なし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@modified€3設定された値が返却される。
13	fix_procedure ［任意］ ※ リマスタリング手順を保存するプラグインは必須	リマスタリング手順を確定する。 本メソッドはリマスタリング手順保存成功時に呼び出される。 各プラグインは現在の設定内容を「リマスタリング手順変更なし」の状態に設定する。 次回 isModified? メソッドが呼び出されたときには本メソッド呼び出し時からの変更の有無を通知する。 【形式】 void fix_procedure() 【引数】 なし 【復帰値】 なし 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@modified€5false に設定する処理が実行される。
14	init_environ ［必須］	リマスタリング環境を初期状態に戻す。 リマスタリング作業中にメニューの[ファイル]-[新規]が実行された場合に呼び出される。 【形式】 void init_environ() 【引数】 なし 【復帰値】 なし

項番	メソッド名	メソッド説明
		【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、初期化処理は何も実行されない。
15	get_procedure_list ［任意］	操作手順一覧画面に表示する操作手順の一覧を返却する。 実行／未実行を管理する手順がない場合は定義する必要はない。 【形式】 Array get_procedure_list() 【引数】 なし 【復帰値】 Array1[Array2a, Array2b, Array2c, ‥] Array2 は個々の操作手順の情報を格納した配列である。 個々の Array2 の要素は以下のようになる。 Array2[0] = true(実行済み)／false（未実行） Array2[1] = 機能名 Array2[2] = 操作内容 Array2[3] = 操作対象
16	set_procedure_list ［任意］	操作手順一覧画面で実行／未実行を編集した結果を自プラグインの手順に反映する。 実行／未実行を管理する手順がない場合は定義する必要はない。 【形式】 void set_procedure_list(array) 【引数】 1. array 操作手順一覧リスト get_procedure_list メソッドで返却した配列と同じ形式。 実行済み／未実行の要素のみが操作手順一覧画面で変更されたものが渡される。 【復帰値】 なし
17	refresh ［任意］	画面の表示状態を最新状態に更新する。 [表示]-[最新の状態に更新]メニューを選択したときに呼び出される。 画面を持たないプラグイン、または、表示を更新する情報を表示していないプラグインは実装する必要はない。なお、更新する情報とはリマスタリング手順に書き出される情報（インストールするパッケージなど）ではなくシステム情報（インストールされているパッケージやディレクトリやファイルの増減など）である。 【形式】 void refresh 【引数】 なし 【復帰値】 なし
18	export ［任意］	リマススタリング資源のエクスポートを行う。 [ファイル]-[エクスポート]メニューを選択したときに呼び出される。 エクスポートするリマスタリング資源を持たないプラグイは実装する必要はない。 なお、リマスタリング資源とは他の環境でリマスタリング環境を構築した際にエクスポートいした内容をインポートすると同じ環境を再現できるようにする資源を指す。 また、リマスタリング手順については export メソッドとは別に make_xml メソッドが呼び出され、本体部分で保存を行うためエクスポートする必要はない。

項番	メソッド名	メソッド説明
		【形式】 BOOL export(dir) 【引数】 1. dir export ディレクトリ export 資源の格納ディレクトリの絶対パス（「利用者が選択したディレクトリ/プラグイン名」が引数として渡される。 【復帰値】 BOOL true:正常終了、false:異常終了
19	import ［任意］	リマスタリング資源のインポートを行う。 [ファイル]-[インポート]メニューを選択したときに呼び出される。 インポートする資源を持たないプラグイは実装する必要はない。なお、引数の dir にはエクスポート時と同様にプラグイン名まで格納したディレクトリ名が指定される。 【形式】 BOOL import(dir) 【引数】 1. dir import ディレクトリ import 資源の格納ディレクトリの絶対パス。 【復帰値】 BOOL true:正常終了、false:異常終了

2.2. OS プラグイン

OS プラグインは個々のＯＳに依存した処理を定義するためのプラグインである。

2.2.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”OS”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	isYou? ［必須］	リマスタリング対象メディアディレクトリが自プラグインでサポートしている OS か反映する。 【形式】 BOOL isYou?(mediaDir) 【引数】 1. mediaDir リマスタリング対象メディアディレクトリ 【復帰値】 true 自プラグインでサポートしている OS false 自プラグインでサポートしていない OS 【備考】 項番 2 以降のメソッドは isYou?メソッドで true を返却した場合のみ呼び出される。
2	get_OS_name ［必須］	リマスタリング対象の OS 名を返却する。 【形式】 String get_OS_name() 【引数】 なし 【復帰値】 リマスタリング対象 OS 名

項番	メソッド名	メソッド説明
		例）”KNOPPIX”など 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@os_name”の値が返却される。
3	get_OS_version ［必須］	リマスタリング対象のOSのバージョンを返却する。 【形式】 String get_OS_version() 【引数】 なし 【復帰値】 リマスタリング対象OSのバージョン 例）”5.0.1”など 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@os_version”の値が返却される。
4	copy_media ［必須］	リマスタリング対象メディアディレクトリからリマスタリング作業に必要な資源を作業ディレクトリにコピーする。 【形式】 BOOL copy_media(mediaDir, workDir) 【引数】 1. mediaDir リマスタリング対象メディアディレクトリ 2. workDir 作業ディレクトリ 【復帰値】 コピーの成否 true 成功 false 失敗
5	get_work_dir ［必須］	リマスタリング作業で使用している作業用ディレクトリを返却する。 【形式】 String get_work_dir( ) 【引数】 なし 【復帰値】 リマスタリング作業で使用している作業用ディレクトリの絶対パス 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@work_dir”の値が返却される。
6	get_root_dir ［必須］	リマスタリング対象OSのrootディレクトリを返却する。 ※リマスタリング対象OSのパッケージの一覧を取得する際にchrootする先のディレクトリとして使用する。 【形式】 String get_root_dir( ) 【引数】 なし 【復帰値】 リマスタリング対象OSの作業用ＯＳ上でのrootディレクトリの絶対パス 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数”@root_dir”の値が返却される。

項番	メソッド名	メソッド説明
7	get_media_dir ［必須］	作業ディレクトリにコピーしたリマスタリング対象 OS の media 形式の格納ディレクトリを返却する。 ※リマスタリング対象 OS のメディア作成を行う場合に使用する。 【形式】 String get_media_dir( ) 【引数】 なし 【復帰値】 リマスタリング対象 OS のメディア形式格納ディレクトリ 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@media_dir” の値が返却される。
8	get_package_type ［必須］	リマスタリング対象 OS で使用されるネイティブなパッケージ形式を返却する。指定可能な値はパッケージプラグインの get_plugin_name メソッドで返却される値である。初版では ”deb”または ”RPM”となる。 【形式】 String get_package_type( ) 【引数】 なし 【復帰値】 リマスタリング対象 OS で使用するパッケージの形式。 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@package_type” の値が返却される。
9	prepare_profile ［任意］	リマスタリング対象 OS の起動高速化の適用を行うための準備を行う。 can_accelerate? メソッドで true を返却した場合のみ本メソッドを定義する。 【形式】 BOOL prepare_profile 【引数】 なし 【復帰値】 BOOL true=正常終了、false=異常終了 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、準備処理は実行されない。
10	do_profile ［任意］	リマスタリング対象 OS の起動高速化のためのプロファイリング、および、その結果の適用を実施する。 can_accelerate? メソッドで true を返却した場合のみ本メソッドを定義する。 【形式】 BOOL do_profile 【引数】 なし 【復帰値】 BOOL true=正常終了、false=異常終了
11	create_os ［必須］	作業用ＯＳ上に作成したリマスタリング対象 OS からＯＳイメージを作成する。 作成した OS イメージはメディアディレクトリ上に配置し、直ちにメディアイメージを作成できる状態にしなければならない。 【形式】

項番	メソッド名	メソッド説明
		BOOL create_os 【引数】 なし 【復帰値】 BOOL true=正常終了、false=異常終了
12	get_boot_image ［必須］	ブート可能なメディアを作成する際に使用するブートイメージのファイルパスを返却する。 ブートイメージファイルのパスは指定されたブートローダに対応するものをメディアディレクトリからの相対パス形式で返却する。 【形式】 String get_boot_image(boot_loader) 【引数】 1. boot_loader　ブートローダ ※ リマスタリングツール V1.0 ではブートローダは ”isolinux” syslinux のみ。 【復帰値】 ブートイメージファイルのパス 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略する場合、インスタンス変数 ”@path_boot_image” にキー：ブートローダ名、値：ブートイメージファイルのパスの形式のハッシュを設定しておく必要がある。 例） @path_boot_image = { “isolinux” => “boot/isolinux/isolinux.bin”, “syslinux => “boot/isolinux/*” }
13	get_boot_catalog ［必須］	ブート可能なメディアを作成する際に使用するブートカタログのファイルパスを返却する。 ブートカタログのファイルのパスは指定されたブートローダに対応するものをメディアディレクトリからの相対パス形式で返却する。 【形式】 String get_boot_catalog(boot_loader) 【引数】 1. boot_loader　ブートローダ ※ リマスタリングツール V1.0 ではブートローダは ”isolinux” syslinux のみ。 【復帰値】 ブートカタログファイルのパス 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略する場合、インスタンス変数 ”@path_boot_catalog” にキー：ブートローダ名、値：ブートカタログファイルのパスの形式のハッシュを設定しておく必要がある。 例） @path_boot_catalog = { “isolinux” => “boot/isolinux/boot.cat”, “syslinux” => “boot/isolinux/boot.cat” }
14	IsInitialized? ［必須］	リマスタリング対象 OS の初期設定が既に完了しているか否かを返却する。リマスタリング手順ファイルの読み込みを行ったときに呼び出される。 【形式】 BOOL isInitialized?(workDir) 【引数】 1. workDir　作業ディレクトリ 【復帰値】

項番	メソッド名	メソッド説明
		true：初期設定完了済み。 false：初期設定未完了。
15	use_directory ［必須］	パラメタで指定されたディレクトリを作業用ディレクトリとして使用する。 リマスタリング手順ファイルの読み込みを行ったときに IsInitialized? メソッドで true を返却した場合に呼び出される。 【形式】 void use_directory(workDir) 【引数】 1. workDir 作業ディレクトリ 【復帰値】 なし
16	check_for_make_distribution ［任意］	ディストリビューション作成にあたり、設定項目のチェックを行う。チェック項目が特にない場合は省略可能。 【形式】 BOOL check_for_make_distribution() 【引数】 なし 【復帰値】 true ：入力エラーなし false：入力エラーあり
17	get_resolvconf_path ［必須］	リマスタリング対象 OS の DNS サーバ設定ファイル「resolv.conf」の絶対パスを返却する。 【形式】 String get_resolvcionf_path() 【引数】 なし 【復帰値】 resolv.conf ファイルの絶対パス。 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略する場合、インスタンス変数 ”@resolv_conf” に resolv.conf ファイルの絶対パスを設定しておく必要がある。
18	can_accelerate? ［必須］	プラグインでサポートする OS で起動高速化の適用が可能か否かを返却する。 本メソッドで true を返却した場合、ディストリビューション作成画面で起動高速化の設定が可能になる。 【形式】 BOOL can_accelerate?() 【引数】 なし 【復帰値】 起動高速化適用の可／不可（true=適用可、false=適用不可)。 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@can_accel” の値が返却される。

2.3. メディアプラグイン

メディアプラグインは CD, DVD, USB メモリなど各種メディア固有の処理を定義するためのプラグインである。

2.3.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”MEDIA”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	get_makeDist_widget	ディストリビューション作成画面に表示する Widget を返却する。 【形式】 Widget get_makeDist_widget() 【引数】 なし 【復帰値】 ディストリビューション作成画面に右上ペインに表示する Widget
2	create_media_image ［必須］	指定されたメディアディレクトリ格納ディレクトリ内のファイルから自プラグインで対応しているメディア形式のイメージファイルを作成する。 【形式】 void create_media_image(media_dir) 【引数】 1. media_dir メディアイメージ格納ディレクトリ 【復帰値】 BOOL true=正常終了、false=異常終了
3	get_output_image ［必須］	作成したメデイアイメージの絶対パスを返却する。 【形式】 String get_output_image() 【引数】 なし 【復帰値】 作成したメディアイメージの絶対パス
4	check_for_make_distribution ［必須］	ディストリビューション作成にあたり、設定された項目のチェックを行う。 【形式】 BOOL check_for_make_distribution() 【引数】 なし 【復帰値】 true ：入力エラーなし false：入力エラーあり
5	can_accelerate? ［必須］	プラグインで提供するメディアで起動高速化の適用が可能か否かを返却する。本メソッドで true を返却した場合、ディストリビューション作成画面で起動高速化の設定が可能になる。 【形式】 BOOL can_accelerate?() 【引数】 なし 【復帰値】 起動高速化適用の可／不可（true=適用可、false=適用不可)。 【備考】 プラグインの基底クラスを継承している場合は省略可能。省略した場合、インスタンス変数 ”@can_accel” の値が返却される。

2.4. エクスポートプラグイン

メディアプラグインは CD, DVD, USB メモリなど各種メディア固有の処理を定義するためのプ

ラグインである。

2.4.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”EXPORT”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	write_procedure ［必須］	リマスタリング手順をファイルに保存する。 【形式】 BOOL write_procedure(file, doc) 【引数】 1. file 保存先ファイル名 2. doc リマスタリング手順を XML 化した XML ドキュメント 【復帰値】 true ：保存成功 false：保存失敗
2	read_procedure ※ XML プラグインのみ	リマスタリング手順をファイルから読み取る。 【形式】 REXML::Document read_procedure(file) 【引数】 1. file 保存先ファイル名 【復帰値】 読み込み成功時 ：読み込んだリマスタリング手順の XML ドキュメント 読み込み失敗時 ：nil
3	get_file_type ［必須］	リマスタリング手順のファイルタイプ（ファイルの説明と拡張子）を返却する。 ※返却可能なファイルタイプの説明は 1 種類のみ。 【形式】 Array get_file_type( ) 【引数】 なし 【復帰値】 ファイルタイプ（ファイルの説明と拡張子）の配列 Array = [ファイルタイプの説明, 拡張子] 例：[“リマスタリング手順ファイル”, “rpml”]

2.5. パッケージプラグイン

パッケージプラグインは debian, RPM など各種パッケージ固有の処理を定義するためのプラグインである。

2.5.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”PACKAGE”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	get_installed_packages ［必須］	リマスタリング対象 OS にインストールされているパッケージの一覧を取得する。 【形式】 Array get_installed_packages() 【引数】 なし

項番	メソッド名	メソッド説明
		【復帰値】 Array1[Array2a, Array2b, Array2c, ･･] Array2 は個々のパッケージの情報を格納した配列である。 個々の Array2 の要素は以下のようになる。 Array2[0] = パッケージ名 Array2[1] = バージョン Array2[2] = パッケージサイズ Array2[3] = パッケージ概要（説明） Array1 はインストール済みのパッケージ数の Array2 を要素に持つ。
2	get_installable_packages ［必須］	パッケージインストール／アンインストール画面の右上ペイン（タブの中）に表示するインストール可能パッケージの一覧表を表示する。 【形式】 Array　get_installable_packages() 【引数】 なし 【復帰値】 Array1[Array2a, Array2b, Array2c, ･･] Array2 は個々のパッケージの情報を格納した配列である。 個々の Array2 の要素は以下のようになる。 Array2[0] = パッケージ名 Array2[1] = バージョン Array2[2] = パッケージ概要（説明） Array1 はインストール可能パッケージ数の Array2 を要素に持つ。
3	get_depend_package ［必須］	依存パッケージの一覧を取得する。 【形式】 Array　get_depend_packages(name, version, operation, download) 【引数】 1. name　　パッケージ名 2. version　バージョン 3. operation 操作(true：インストール、false：アンインストール) 4. download 依存パッケージダウンロードの要／不要(true：要、false：不要 【復帰値】 Array1[Array2a, Array2b, Array2c, ･･] Array2 は個々のパッケージの情報を格納した配列である。 個々の Array2 の要素は以下のようになる。 Array2[0] = パッケージ名 Array2[1] = バージョン Array2[2] = パッケージ概要（説明） Array2[3] = パッケージファイル名 Array1 は選択されたパッケージ数の Array2 を要素に持つ。
4	get_makePackage_widget ［必須］	パッケージ作成画面に右上ペインに表示する Widget を返却する。 【形式】 Widget　get_makePackage_widget() 【引数】 なし 【復帰値】 パッケージ作成画面に右上ペインに表示する Widget
5	get_readable_packages	パッケージ作成機能で読み込み可能なパッケージの一覧表を返却する。

項番	メソッド名	メソッド説明
	［必須］	【形式】 Array　get_readable_packages() 【引数】 なし 【復帰値】 Array1[Array2a, Array2b, Array2c, ‥] Array2 は個々のパッケージの情報を格納した配列である。 個々の Array2 の要素は以下のようになる。 Array2[0] = パッケージ名 Array2[1] = バージョン Array2[2] = パッケージ概要（説明） Array1 はインストール可能パッケージ数の Array2 を要素に持つ。
6	update_package_index ［必須］	パッケージインデックスを更新する。 【形式】 void　update_package_index() 【引数】 なし 【復帰値】 なし
7	check_for_make_distribution ［必須］	ディストリビューション作成にあたり、設定された項目のチェックを行う。 【形式】 BOOL　check_for_make_distribution() 【引数】 なし 【復帰値】 true ：入力エラーなし false：入力エラーあり

2.6. エミュレータプラグイン

エミュレータプラグインは QEMU, VMPlayer など OS イメージをテスト走行する仮想マシンソフト固有の処理を定義するためのプラグインである。

2.6.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”EMULATOR”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	run ［必須］	仮想マシン上で指定された OS イメージを実行する。 【形式】 void　run(media_image, use_net = false, boot_cmd = “”) 【引数】 1. media_imahe　OS イメージファイル 2. use_net　ネットワーク使用の有無(true:使用、false:未使用) 3. boot_cmd　カーネルパラメタ 【復帰値】 なし
2	check_for_make_distribution	ディストリビューション作成にあたり、設定された項目のチェックを行う。 【形式】

項番	メソッド名	メソッド説明
	［必須］	BOOL check_for_make_distribution() 【引数】 なし 【復帰値】 true ：入力エラーなし false：入力エラーあり

2.7. テストプラグイン

テストプラグインは自動テスト機能で実行するテスト項目、内容を定義するためのプラグインである。

2.7.1. プラグインで記述するメソッド

以下に一覧を示すメソッドは、get_plugin_type メソッドで ”TEST”を返却した場合のみ定義する。

項番	メソッド名	メソッド説明
1	get_test_set_widget ［必須］	自動テスト設定画面/自動テスト実行画面に表示する Widget を返却する。 【形式】 Widget get_test_set_widget() 【引数】 なし 【復帰値】 自動テスト設定画面/自動テスト実行画面に表示する Widget
2	show_setting_view ［必須］	テスト項目設定部の表示のオン／オフの切り替えを行う。 ※ 自動テスト設定画面ではオン、自動テスト実行画面ではオフとなる。 【形式】 void show_setting_view(BOOL show) 【引数】 1. show テスト項目設定部のオン／オフ指定。true=表示、false=非表示 【復帰値】 なし
3	exec_test ［必須］	設定されたテスト項目を実行する。 【形式】 void exec_test() 【引数】 なし 【復帰値】 なし

－以上－