VVE Referenční příručka

3.5

Generováno programem Doxygen 1.5.4

Wed Jan 21 21:13:19 2009

Obsah

Kapitola 1

VVE Hierarchie adresářů

1.1 VVE Adresáře

Následující hierarchie adresářů je zhruba, ale ne úplně, řazena podle abecedy:

Kapitola 2

VVE Rejstřík prostorů jmen

2.1 VVE Seznam prostorů jmen

Zde naleznete seznam všech dokumentovaných prostorů jmen se stručným popisem:

Kapitola 3

VVE Rejstřík hierarchie tříd

3.1 VVE Hierarchie tříd

Zde naleznete seznam, vyjadřující vztah dědičnosti tříd. Je seřazen přibližně (ale ne úplně) podle abecedy:

Kapitola 4

VVE Rejstřík tříd

4.1 VVE Seznam tříd

Následující seznam obsahuje především identifikace tříd, ale nacházejí se zde i další netriviální prvky, jako jsou struktury (struct), unie (union) a rozhraní (interface). V seznamu jsou uvedeny jejich stručné popisy:

Kapitola 5

VVE Rejstřík stránek

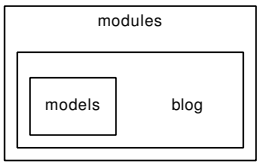
5.1 VVE Ostatní stránky

Následující seznam odkazuje na další stránky projektu:

Kapitola 6

VVE Dokumentace k adresářům

6.1 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/blog/



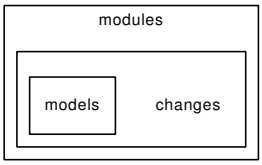
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.2 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/changes/



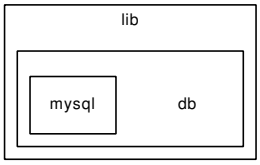
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.3 Reference k adresáři /home/cuba/work-net/vve3_2/lib/db/



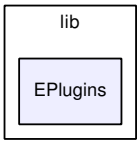
Adresáře

- adresář mysql

Soubory

- soubor **db.class.php**
- soubor **db.interface.php**
- soubor **delete.class.php**
- soubor **insert.class.php**
- soubor **select.class.php**
- soubor **update.class.php**

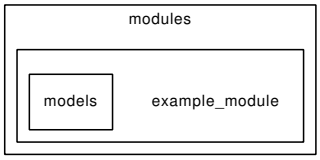
6.4 Reference k adresáři /home/cuba/work-net/vve3_2/lib/EPlugins/



Soubory

- soubor **changes.class.php**
- soubor **csvdata.class.php**
- soubor **eplugin.class.php**
- soubor **progressbar.class.php**
- soubor **scroll.class.php**
- soubor **sendmail.class.php**
- soubor **userfiles.class.php**
- soubor **userimages.class.php**
- soubor **verifyimage.class.php**

6.5 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/example_module/



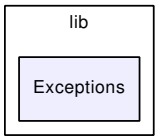
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

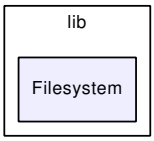
6.6 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/Exceptions/



Soubory

- soubor **coreException.class.php**
- soubor **errors.class.php**

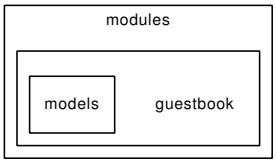
6.7 Reference k adresáři /home/cuba/work-net/vve3_2/lib/Filesystem/



Soubory

- soubor **dir.class.php**
- soubor **file.class.php**
- soubor **imagefile.class.php**
- soubor **zipfile.class.php**

6.8 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/guestbook/



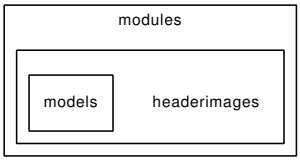
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **panel.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.9 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/headerimages/



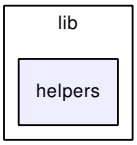
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

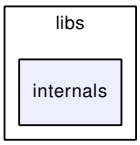
6.10 Reference k adresáři /home/cuba/work-net/vve3_2/lib/helpers/



Soubory

- soubor **ctrlhelper.class.php**
- soubor **datetimectrlhelper.class.php**
- soubor **dbctrlhelper.class.php**
- soubor **helper.class.php**
- soubor **localectrlhelper.class.php**
- soubor **mailctrlhelper.class.php**
- soubor **textctrlhelper.class.php**

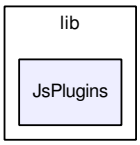
6.11 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/smarty/libs/internals/



Soubory

- soubor **core.assemble_plugin_filepath.php**
- soubor **core.assign_smarty_interface.php**
- soubor **core.create_dir_structure.php**
- soubor **core.display_debug_console.php**
- soubor **core.get_include_path.php**
- soubor **core.get_microtime.php**
- soubor **core.get_php_resource.php**
- soubor **core.is_secure.php**
- soubor **core.is_trusted.php**
- soubor **core.load_plugins.php**
- soubor **core.load_resource_plugin.php**
- soubor **core.process_cached_inserts.php**
- soubor **core.process_compiled_include.php**
- soubor **core.read_cache_file.php**
- soubor **core.rm_auto.php**
- soubor **core.rmdir.php**
- soubor **core.run_insert_handler.php**
- soubor **core.smarty_include_php.php**
- soubor **core.write_cache_file.php**
- soubor **core.write_compiled_include.php**
- soubor **core.write_compiled_resource.php**
- soubor **core.write_file.php**

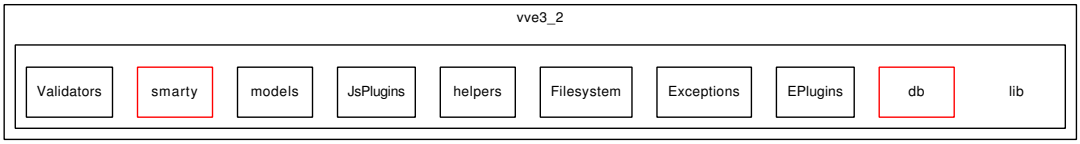
6.12 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/JsPlugins/



Soubory

- soubor **jquery.class.php**
- soubor **jsplugin.class.php**
- soubor **jspluginjsfile.class.php**
- soubor **lightbox.class.php**
- soubor **progressbarjs.class.php**
- soubor **submitform.class.php**
- soubor **switchcontenteasy.class.php**
- soubor **tabcontent.class.php**
- soubor **tinymce.class.php**

6.13 Reference k adresáři /home/cuba/work-net/vve3_2/lib/



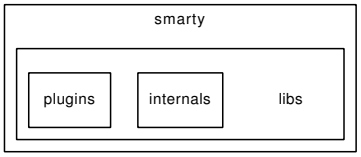
Adresáře

- adresář db
- adresář EPlugins
- adresář Exceptions
- adresář Filesystem
- adresář helpers
- adresář JsPlugins
- adresář models
- adresář smarty
- adresář Validators

Soubory

- soubor **action.class.php**
- soubor **article.class.php**
- soubor **auth.class.php**
- soubor **category.class.php**
- soubor **config.class.php**
- soubor **container.class.php**
- soubor **controller.class.php**
- soubor **files.class.php**
- soubor **form.class.php**
- soubor **images.class.php**
- soubor **links.class.php**
- soubor **locale.class.php**
- soubor **mainmenu.class.php**
- soubor **messages.class.php**
- soubor **module.class.php**
- soubor **moduledirs.class.php**
- soubor **panel.class.php**
- soubor **rights.class.php**
- soubor **routes.class.php**
- soubor **sessions.class.php**
- soubor **sitemap.class.php**
- soubor **specialfunctions.class.php**
- soubor **template.class.php**
- soubor **uploadfiles.class.php**
- soubor **urlparam.class.php**
- soubor **urlrequest.class.php**
- soubor **view.class.php**

6.14 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/smarty/libs/



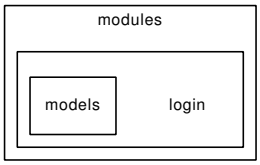
Adresáře

- adresář internals
- adresář plugins

Soubory

- soubor **Config_File.class.php**
- soubor **Smarty.class.php**
- soubor **Smarty_Compiler.class.php**

6.15 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/login/



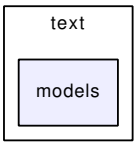
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

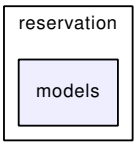
6.16 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/text/models/



Soubory

- soubor **textdetail.php**

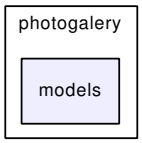
6.17 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/reservation/models/



Soubory

- soubor **CoursesList.php**

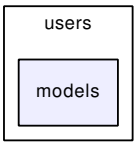
6.18 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/photogalery/models/



Soubory

- soubor **galerieslist.php**
- soubor **galerydetail.php**
- soubor **photodetail.php**
- soubor **sectiondetail.php**
- soubor **sectionlist.php**

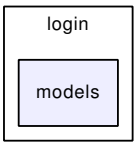
6.19 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/users/models/



Soubory

- soubor **userdetail.php**
- soubor **userslist.php**

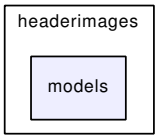
6.20 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/login/models/



Soubory

- soubor **userdetail.php**

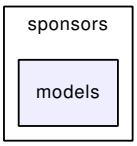
6.21 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/headerimages/models/



Soubory

- soubor **headerimagedetail.php**

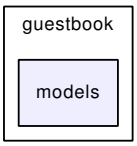
6.22 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/sponsors/models/



Soubory

- soubor **sponsorDetail.php**
- soubor **sponsorsList.php**

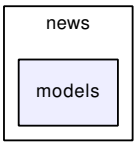
6.23 Reference k adresáři /home/cuba/work-net/vve3_2/modules/guestbook/models/



Soubory

- soubor **sponsorDetail.php**
- soubor **sponsorsList.php**

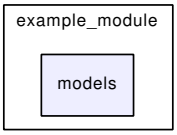
6.24 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/news/models/



Soubory

- soubor **newsdetail.php**
- soubor **newslist.php**

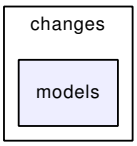
6.25 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/example_module/models/



Soubory

- soubor **newslist.php**

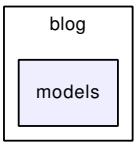
6.26 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/changes/models/



Soubory

- soubor **changeslist.php**

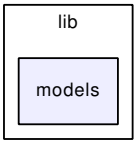
6.27 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/blog/models/



Soubory

- soubor **blogdetail.php**
- soubor **blogSectionDetail.php**
- soubor **blogslist.php**
- soubor **sectionDetail.php**

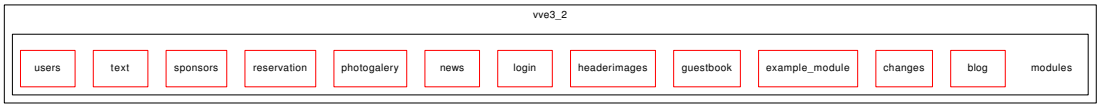
6.28 Reference k adresáři /home/cuba/work-net/vve3_2/lib/models/



Soubory

- soubor **dbmodel.class.php**
- soubor **filemodel.class.php**
- soubor **model.class.php**

6.29 Reference k adresáři /home/cuba/work-net/vve3_2/modules/



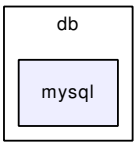
Adresáře

- adresář blog
- adresář changes
- adresář example_module
- adresář guestbook
- adresář headerimages
- adresář login
- adresář news
- adresář photogalery
- adresář reservation
- adresář sponsors
- adresář text
- adresář users

Soubory

- soubor **initialWww.php**
- soubor **menu.class.php**

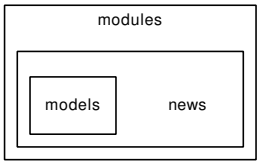
6.30 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/db/mysql/



Soubory

- soubor **db.class.php**
- soubor **delete.class.php**
- soubor **insert.class.php**
- soubor **select.class.php**
- soubor **update.class.php**

6.31 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/news/



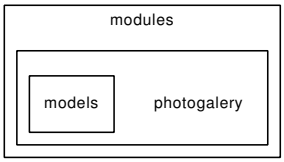
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **panel.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.32 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/photogalery/



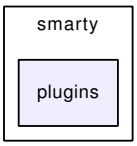
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **panel.class.php**
- soubor **routes.class.php**
- soubor **sitemap.class.php**
- soubor **view.class.php**

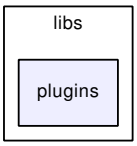
6.33 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/smarty/plugins/



Soubory

- soubor **function.html_engine_image.php**
- soubor **function.html_engine_style.php**
- soubor **postfilter.cztypo.php**
- soubor **resource.engine.php**
- soubor **resource.module.php**

6.34 Reference k adresáři /home/cuba/work-net/vve3_-2/lib/smarty/libs/plugins/

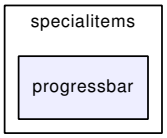


Soubory

- soubor **block.textformat.php**
- soubor **compiler.assign.php**
- soubor **function.assign_debug_info.php**
- soubor **function.config_load.php**
- soubor **function.counter.php**
- soubor **function.cycle.php**
- soubor **function.debug.php**
- soubor **function.eval.php**
- soubor **function.fetch.php**
- soubor **function.html_checkboxes.php**
- soubor **function.html_image.php**
- soubor **function.html_options.php**
- soubor **function.html_radios.php**
- soubor **function.html_select_date.php**
- soubor **function.html_select_time.php**
- soubor **function.html_table.php**
- soubor **function.mailto.php**
- soubor **function.math.php**
- soubor **function.popup.php**
- soubor **function.popup_init.php**
- soubor **modifier.capitalize.php**
- soubor **modifier.cat.php**
- soubor **modifier.count_characters.php**
- soubor **modifier.count_paragraphs.php**
- soubor **modifier.count_sentences.php**
- soubor **modifier.count_words.php**
- soubor **modifier.date_format.php**
- soubor **modifier.debug_print_var.php**
- soubor **modifier.default.php**
- soubor **modifier.escape.php**
- soubor **modifier.indent.php**
- soubor **modifier.lower.php**
- soubor **modifier.mb_upper.php**
- soubor **modifier.nl2br.php**
- soubor **modifier.regex_replace.php**
- soubor **modifier.replace.php**

- soubor **modifier.replacearray.php**
- soubor **modifier.spacify.php**
- soubor **modifier.string_format.php**
- soubor **modifier.strip.php**
- soubor **modifier.strip_tags.php**
- soubor **modifier.truncate.php**
- soubor **modifier.upper.php**
- soubor **modifier.wordwrap.php**
- soubor **outputfilter.czechtypo.php**
- soubor **outputfilter.trimwhitespace.php**
- soubor **shared.escape_special_chars.php**
- soubor **shared.make_timestamp.php**

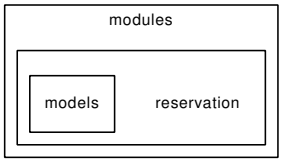
6.35 Reference k adresáři /home/cuba/work-net/vve3_-2/specialitems/progressbar/



Soubory

- soubor **progressbar.php**

6.36 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/reservation/



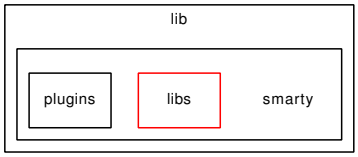
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

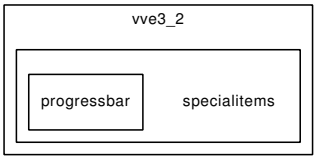
6.37 Reference k adresáři /home/cuba/work-net/vve3_2/lib/smarty/



Adresáře

- adresář libs
- adresář plugins

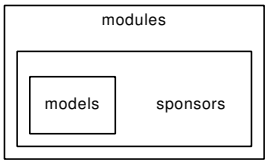
6.38 Reference k adresáři /home/cuba/work-net/vve3_-2/specialitems/



Adresáře

- adresář progressbar

6.39 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/sponsors/



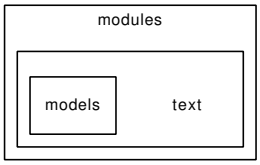
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **panel.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.40 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/text/



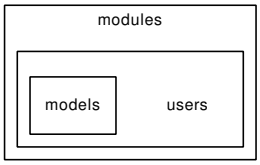
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

6.41 Reference k adresáři /home/cuba/work-net/vve3_-2/modules/users/



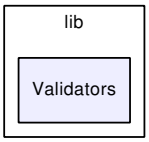
Adresáře

- adresář models

Soubory

- soubor **action.class.php**
- soubor **controler.class.php**
- soubor **routes.class.php**
- soubor **view.class.php**

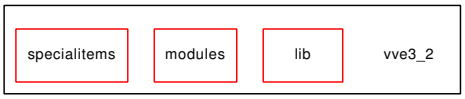
6.42 Reference k adresáři /home/cuba/work-net/vve3_2/lib/Validators/



Soubory

- soubor **timevalidator.class.php**
- soubor **urlvalidator.class.php**
- soubor **validator.class.php**

6.43 Reference k adresáři /home/cuba/work-net/vve3_2/



Adresáře

- adresář lib
- adresář modules
- adresář specialitems

Soubory

- soubor **app.php**

Kapitola 7

VVE Dokumentace prostorů jmen

7.1 Dokumentace prostoru jmen Action

class Copyright (c) 2008 Jakub Matas

7.1.1 Detailní popis

class Copyright (c) 2008 Jakub Matas

Verze:

$Id: db.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro vytvoření db konektoru

7.2 Dokumentace prostoru jmen module

Kontroler pro obsluhu fotogalerie.

7.2.1 Detailní popis

Kontroler pro obsluhu fotogalerie.

Verze:

0.0.1

Autor:

Jakub Matas <jakubmatas@gmail.com> 2008 photogalery for VVE v3.0.1

Last number CoreError: 15

7.3 Dokumentace prostoru jmen Smarty

Config_File class.

7.3.1 Detailní popis

Config_File class.

Template compiling class.

Project: Smarty: the PHP compiling template engine File: Smarty_Compiler.class.php.

Project: Smarty: the PHP compiling template engine File: Smarty.class.php.

Smarty shared plugin.

Smarty plugin.

Config file reading class.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

2.6.19 Copyright: 2001-2005 New Digital Group, Inc. Andrei Zmievski <andrei@php.net> public plugins This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA For questions, help, comments, discussion, etc., please join the Smarty mailing list. Send a blank e-mail to smarty-general-subscribe@lists.php.net http://smarty.php.net/ 2001-2005 New Digital Group, Inc. Monte Ohrt

Autor:

Andrei Zmievski <andrei@php.net>

Verze:

2.6.19

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Monte Ohrt

Autor:

Andrei Zmievski <andrei@php.net>

Verze:

2.6.19 2001-2005 New Digital Group, Inc.

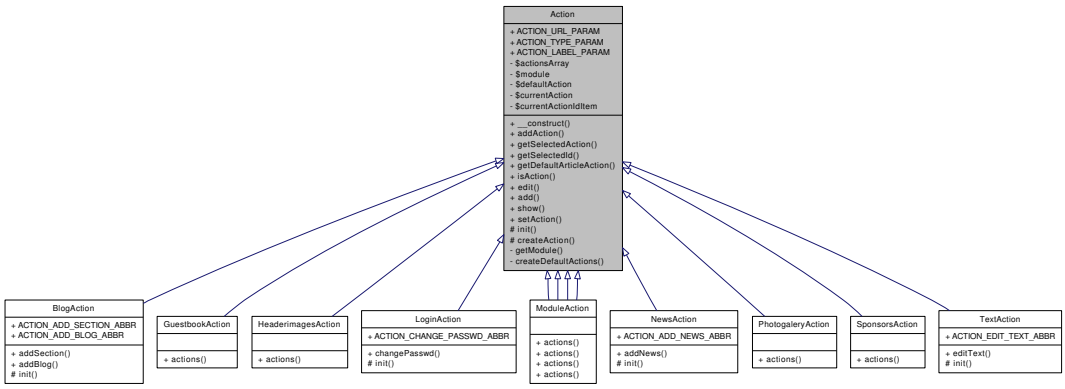
Kapitola 8

VVE Dokumentace tříd

8.1 Dokumentace třídy Action

Třída pro obsluhu akcí.

Diagram dědičnosti pro třídu Action



Veřejné metody

- __construct ()

  Konstruktor.

- addAction ($actionAbbr, $actionName, $actionLabel, $isDefault=false)

  Metoda přidá požadovanou akci do seznamu akcí.

- getSelectedAction ()

  Funkce vrazí která akce je definována viz dokumentace.

- getSelectedId ()

  Funkce vrazí která akce je definována viz dokumentace.

- getDefaultArticleAction ()

Metoda vrací výchozí akci pro kontroler při zobrazení článku.

- isAction ()

  Funkce zjišťuje, jesli byla akce nastavena.

- edit ()

  Zakladni metoda pro vygenerování akce edit.

- add ($idModule=null)

  Zakladni metoda pro vygenerování akce add.

- show ($idModule=null)

  Zakladni metoda pro vygenerování akce show.

Statické veřejné metody

- static setAction ($action, $idItem)

  Metoda vrací výchozí akci pokud není definována.

Veřejné atributy

- const **ACTION_URL_PARAM** = 'url'
- const **ACTION_TYPE_PARAM** = 'action'
- const **ACTION_LABEL_PARAM** = 'label'

Chráněné metody

- init ()

  Metoda pro inicializaci akcí.

- createAction ($actionAbbr)

  Metoda vytvoří objekt pro danou akci pro vložení do URL.

8.1.1 Detailní popis

Třída pro obsluhu akcí.

Třída obsluhuje přenášené akce v URL. Slouží také pro generování vlastních akcí v modulu, jejich úpravu. Podle zvolené akce se volí také kontroler modulu.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro obsluhu akcí

Definice je uvedena na řádku 14 v souboru action.class.php.

8.1.2 Dokumentace konstruktoru a destruktoru

8.1.2.1 Action::__construct () [final]

Konstruktor.

Parametry:

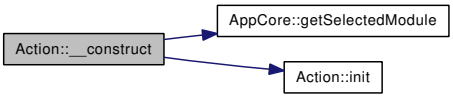
Module – objekt modulu (nutný pro zjištění id modulu)

Definice je uvedena na řádku 68 v souboru action.class.php.

Odkazuje se na AppCore::getSelectedModule() a init().

```
68                                      {
69       $this->module = AppCore::getSelectedModule();
70 //    Vytvoření uživatelských akcí
71       $this->createDefaultActions();
72         $this->init();
73    }
```

Tato funkce volá...



8.1.3 Dokumentace k metodám

8.1.3.1 Action::addAction ($ *actionAbbr*, $ *actionName*, $ *actionLabel*, $ *isDefault* = `false`) [final]

Metoda přidá požadovanou akci do seznamu akcí.

Parametry:

string – zkratka akce
string – název akce pro kontroler
string – jazykový název akce - je přenášen do url
boolean – jestli má být daná akce výchozí

Definice je uvedena na řádku 104 v souboru action.class.php.

Používá se v add(), edit(), TextAction::init(), NewsAction::init(), LoginAction::init(), BlogAction::init() a show().

```
104                                                                                                                 {
105         if(!key_exists($actionAbbr, $this->actionsArray)){
106            $this->actionsArray[$actionAbbr] = array(self::ACTION_TYPE_PARAM => $actionName,
107                                                     self::ACTION_LABEL_PARAM => $actionLabel);
108            if($isDefault){
109               $this->defaultAction = $actionAbbr;
110            }
111         }
112      }
```

8.1.3.2 Action::getSelectedAction ()

Funkce vrazí která akce je definována viz dokumentace.

Návratová hodnota:

string – název prováděné akce viz dokumentace

Definice je uvedena na řádku 119 v souboru action.class.php.

```
119                                        {
120         return $this->actionsArray[self::$currentAction][self::ACTION_TYPE_PARAM];
121      }
```

8.1.3.3 Action::getSelectedId ()

Funkce vrazí která akce je definována viz dokumentace.

Návratová hodnota:

integer – id modulu, který má provést akci

Definice je uvedena na řádku 128 v souboru action.class.php.

```
128                                       {
129         return $this->selectedId;
130      }
```

8.1.3.4 Action::getDefaultArticleAction ()

Metoda vrací výchozí akci pro kontroler při zobrazení článku.

Návratová hodnota:

string – název akce při článku

Definice je uvedena na řádku 136 v souboru action.class.php.

```
136                                                 {
137         return $this->actionsArray[$this->defaultAction][self::ACTION_TYPE_PARAM];
138     }
```

8.1.3.5 Action::isAction ()

Funkce zjišťuje, jesli byla akce nastavena.

Návratová hodnota:

boolean – true pokud byla akce nastavena

Definice je uvedena na řádku 145 v souboru action.class.php.

```
146     {
147         if(self::$currentAction != null AND self::$currentActionIdItem == $this->getModule()->getId()){
148             return true;
149         }
150         return false;
151     }
```

8.1.3.6 Action::edit ()

Zakladni metoda pro vygenerování akce edit.

Návratová hodnota:

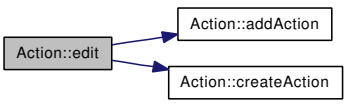
string – akce edit pro url

Definice je uvedena na řádku 161 v souboru action.class.php.

Odkazuje se na addAction() a createAction().

```
161                             {
162         $actionAbbr = 'e';
163         $this->addAction($actionAbbr, "edit", _('uprava'));
164         return $this->createAction($actionAbbr);
165     }
```

Tato funkce volá...



8.1.3.7 Action::add ($ *idModule* = null)

Zakladni metoda pro vygenerování akce add.

Návratová hodnota:

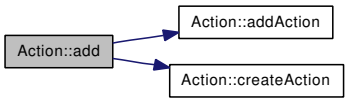
string – akce edit pro url

Definice je uvedena na řádku 171 v souboru action.class.php.

Odkazuje se na addAction() a createAction().

```
171                                         {
172         $actionAbbr = 'a';
173         $this->addAction($actionAbbr, "add", _('pridani'));
174         return $this->createAction($actionAbbr);
175     }
```

Tato funkce volá...



8.1.3.8 Action::show ($ *idModule* = null)

Zakladni metoda pro vygenerování akce show.

Návratová hodnota:

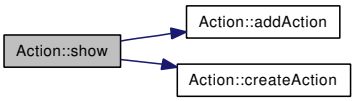
string – akce edit pro url

Definice je uvedena na řádku 181 v souboru action.class.php.

Odkazuje se na addAction() a createAction().

```
181                                         {
182         $actionAbbr = 's';
183         $this->addAction($actionAbbr, "show", _('ukaz'), true);
184         return $this->createAction($actionAbbr);
185     }
```

Tato funkce volá...



8.1.3.9 Action::createAction ($ *actionAbbr*) [protected]

Metoda vytvoří objekt pro danou akci pro vložení do URL.

Parametry:

string $actionAbbr – identifikátor akce

Návratová hodnota:

pole obsahující název, zkratku, id item

Definice je uvedena na řádku 192 v souboru action.class.php.

Používá se v add(), edit() a show().

```
192                                                 {
193       $action = $this->actionsArray[$actionAbbr];
194
195         $return = array();
196         $return[0]= $action[self::ACTION_LABEL_PARAM];
197         $return[1]= $actionAbbr;
198         $return[2]= $this->getModule()->getId();
199
200         return $return;
201 //    echo $returnString = $action[self::ACTION_LABEL_PARAM].self::ACTION_URL_LABEL_TYPE_SEP
202 //       .$actionAbbr.self::ACTION_URL_TYPE_ID_SEP.$this->getModule()->getId();
203 //    return $returnString;
204     }
```

8.1.3.10 static Action::setAction ($ *action*, $ *idItem*) [static]

Metoda vrací výchozí akci pokud není definována.

Návratová hodnota:

string //TODO není korektní asi lepší použít setAction Metoda nastavuje akci v url

Parametry:

string $action – řetězec s akcí

integer $idItem – id item pro danou akci

Definice je uvedena na řádku 221 v souboru action.class.php.

Používá se v Links::checkActionUrlRequest().

```
221                                                     {
222       self::$currentAction = $action;
223       self::$currentActionIdItem = $idItem;
224     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/action.class.php

8.2 Dokumentace třídy AppCore

Vypecky Engine.

Veřejné metody

- setAppMainDir ($appMainDir)

  Metoda nastavuje hlavní adresář aplikace.

- createMainMenu ()

  Metoda vytvoří hlavní menu aplikace.

- assignMainVarsToTemplate ()

  Metoda přiřadí do šablony hlavní proměnné systému.

- renderTemplate ()

  metoda vyrenderuje šablonu

- runModules ()

  Metoda spouští moduly //TODO.

- runPanel ($side)

  Metoda inicializuje a spustí levý panel.

- selectCategory ()

  metoda vybere, která kategorie je vybrána a uloží je di objektu kategorie

- **assignCoreErrorsToTpl** ()
- assignModuleVarsToTpl ()

  Metoda přiřadí všechny proměnné do šablonovacího systému //TODO není omplementována, vytvořit načítání do šablony.

- runSitemap ()

  Metoda vytváří sitemapu a odesílá ji na výstup.

- runApp ()

  Hlavní metoda provádění aplikace.

Statické veřejné metody

- static getInstance ()

  Singleton instance objektu.

- static getAppWebDir ()

  Metoda vrací adresář aplikace.

- static createApp ()

Metoda vygeneruje instanci aplikace pokud již instance existuje, bude vyhozena vyjímka Instance aplikace je singleton.

- static sysConfig ()

  Metoda vrací objekt na systémovou konfiguraci.

- static getTepmlateFaceDir ($fullDir=true)

  Metoda vrací adresář ke zvolenému vzhledu.

- static getTepmlateDefaultFaceDir ($fullDir=true)

  Metoda vrací adresář k výchozímu vzhledu.

- static getDbConnector ()

  Metoda vrací objekt db conektoru.

- static getSelectedModule ()

  Metoda vrací objekt aktuálního modulu.

- static getSellectedCategory ()

  Metoda vrací onformace o práve zpracovávané kategori nebo false.

- static & getModuleMessages ()

  Metoda vrací objekt pro zprávy modulů.

- static & getModuleErrors ()

  Metoda vrací objekt pro chybové zprávy modulů.

Veřejné atributy

- const **MAIN_ENGINE_PATH** = ’./’
- const **APP_MAIN_FILE** = ’index.php’
- const **MAIN_CONFIG_FILE** = "config.xml"
- const **MAIN_CONFIG_DB_SECTION** = "db"
- const **MODULES_DIR** = "modules"
- const **ENGINE_EPLUINS_DIR** = ’EPlugins’
- const **ENGINE_JSPLUINS_DIR** = ’JsPlugins’
- const **ENGINE_HELPERS_DIR** = ’helpers’
- const **ENGINE_VALIDATORS_DIR** = ’Validators’
- const **ENGINE_MODELS_DIR** = ’models’
- const **ENGINE_FILESYSTEM_DIR** = ’Filesystem’
- const **ENGINE_CACHE_DIR** = ’cache’
- const **TEMPLATES_DIR** = "templates"
- const **TEMPLATES_IMAGES_DIR** = "images"
- const **TEMPLATES_STYLESHEETS_DIR** = ’stylesheets’
- const **SPECIALITEMS_DIR** = ’specialitems’
- const **MODULE_DBTABLES_PREFIX** = "dbtable"
- const **MODULE_CONTROLLER_SUFIX** = ’Controller’
- const **MODULE_VIEWER_SUFIX** = ’View’
- const **MODULE_MAIN_CONTROLLER_PREFIX** = ’Main’

- const **MODULE_PANEL_CLASS_SUFIX** = 'Panel'
- const **MODULE_PANEL_CONTROLLER** = 'panelController'
- const **MODULE_PANEL_VIEWER** = 'panelView'
- const **MODULE_SITEMAP_SUFIX_CLASS** = 'SiteMap'
- const **MEDIA_URL_PARAM_TYPE** = 'media'
- const **GETTEXT_DEFAULT_DOMAIN** = 'messages'
- const **GETTEXT_DEFAULT_LOCALES_DIR** = './locale'
- const **GETTEXT_MDOMAIN** = 'module_messages'
- const **FACES_DIR** = 'faces'
- const **FACE_DEFAULT_NAME** = 'default'
- **$coreErrors** = null
- **$auth** = null

8.2.1 Detailní popis

Vypecky Engine.

Hlavní třída aplikace - singleton Obsluhuje celou aplikaci a její komponenty.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

jakub

Date

2008-12-30 01:35:31 +0100 (Tue, 30 Dec 2008)

LastChangedBy

jakub

LastChangedDate

2008-12-30 01:35:31 +0100 (Tue, 30 Dec 2008)

Hlavní třída aplikace(Singleton) GNU General Public License v. 2 viz. Docs/license.txt

Definice je uvedena na řádku 16 v souboru app.php.

8.2.2 Dokumentace k metodám

8.2.2.1 static AppCore::getInstance () [static]

Singleton instance objektu.

Návratová hodnota:

AppCore

Definice je uvedena na řádku 375 v souboru app.php.

```
376     {
377         if (null === self::$_coreInstance) {
378             self::$_coreInstance = new self();
379         } else {
380             throw new Exception("Aplication object have been created");
381         }
382         return self::$_coreInstance;
383     }
```

8.2.2.2 static AppCore::getAppWebDir () [static]

Metoda vrací adresář aplikace.

Návratová hodnota:

string – adresář aplikace

Definice je uvedena na řádku 389 v souboru app.php.

```
390     {
391         return self::$_appWebDir;
392     }
```

8.2.2.3 static AppCore::sysConfig () [static]

Metoda vrací objekt na systémovou konfiguraci.

Návratová hodnota:

Config – objekt konfigurace

Definice je uvedena na řádku 408 v souboru app.php.

Používá se v MainMenu::__construct(), Category::getDefaultCategory(), Eplugin::getSysConfig() a Controller::getSysConfig().

```
408                                     {
409         return self::$sysConfig;
410     }
```

8.2.2.4 static AppCore::getTepmlateFaceDir ($ *fullDir* = `true`) [static]

Metoda vrací adresář ke zvolenému vzhledu.

Parametry:

boolean – jestli se má vrátit celá cesta nebo jemo část od hlavního adresáře

Návratová hodnota:

string – adresář zvoleného vzhledu

Definice je uvedena na řádku 418 v souboru app.php.

```
418                                                                 {
419         if($fullDir){
420             return self::$_appWebDir.DIRECTORY_SEPARATOR.self::FACES_DIR.DIRECTORY_SEPARATOR.self::$te
421         } else {
422             return self::FACES_DIR.URL_SEPARATOR.self::$templateFaceDir.URL_SEPARATOR;
423         }
424     }
```

8.2.2.5 static AppCore::getTepmlateDefaultFaceDir ($ *fullDir* = `true`) [static]

Metoda vrací adresář k výchozímu vzhledu.

Parametry:

boolean – jestli se má vrátit celá cesta nebo jemo část od hlavního adresáře

Návratová hodnota:

string – adresář výchozího vzhledu

Definice je uvedena na řádku 432 v souboru app.php.

```
432                                                                 {
433         if($fullDir){
434             return self::$_appWebDir.DIRECTORY_SEPARATOR.self::FACES_DIR.DIRECTORY_SEPARATOR.self::$te
435         } else {
436             return self::FACES_DIR.URL_SEPARATOR.self::$templateDefaultFaceDir.URL_SEPARATOR;
437         }
438     }
```

8.2.2.6 static AppCore::getDbConnector () [static]

Metoda vrací objekt db conektoru.

Návratová hodnota:

DbConnector – objekt db konektoru

Definice je uvedena na řádku 445 v souboru app.php.

Používá se v SiteMap::__construct(), Panel::__construct(), DbModel::__construct(), DbCtrlHelper::__construct(), Eplugin::__construct(), Category::factory() a DbModel::getDb().

```
445                                             {
446         return self::$dbConnector;
447     }
```

8.2.2.7 static AppCore::getSelectedModule () [static]

Metoda vrací objekt aktuálního modulu.

Návratová hodnota:

Module – objekt vybraného modulu

Definice je uvedena na řádku 454 v souboru app.php.

Používá se v Validator::__construct(), UrlRequest::__construct(), Panel::__construct(), Form::__construct(), Eplugin::__construct(), Controller::__construct(), Action::__construct(), Locale::bindTextDomain(), Model::getModule() a Locale::switchToModuleTexts().

```
454                                             {
455         return self::$selectedModule;
456     }
```

8.2.2.8 static AppCore::getSellectedCategory () [static]

Metoda vrací onformace o práve zpracovávané kategori nebo false.

Návratová hodnota:

array – právě zpracovávaná kategorie (název, id)

Definice je uvedena na řádku 462 v souboru app.php.

Používá se v Eplugin::getLinks().

```
462                                             {
463         if(!empty (self::$currentCategory)){
464             return self::$currentCategory;
465         } else {
466             return false;
467         }
468     }
```

8.2.2.9 static& AppCore::getModuleMessages () [static]

Metoda vrací objekt pro zprávy modulů.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 673 v souboru app.php.

Používá se v Validator::__construct(), Form::__construct(), File::__construct(), Eplugin::__construct() a Controller::__construct().

```
673                                                 {
674         return self::$messages;
675     }
```

8.2.2.10 static& AppCore::getModuleErrors () [static]

Metoda vrací objekt pro chybové zprávy modulů.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 681 v souboru app.php.

Používá se v Validator::__construct(), Form::__construct(), File::__construct(), Eplugin::__construct() a Controller::__construct().

```
681                                                 {
682         return self::$userErrors;
683     }
```

8.2.2.11 AppCore::setAppMainDir ($ *appMainDir*)

Metoda nastavuje hlavní adresář aplikace.

Parametry:

string – hlavní adresář aplikace

Definice je uvedena na řádku 781 v souboru app.php.

```
781                                                 {
782         self::$_appWebDir = $appMainDir;
783     }
```

8.2.2.12 AppCore::renderTemplate ()

metoda vyrenderuje šablonu

Plánované úpravy

dořešit při neexistenci ostatní typů medií

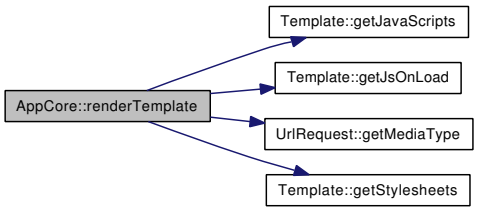
Definice je uvedena na řádku 884 v souboru app.php.

Odkazuje se na Template::getJavaScripts(), Template::getJsOnLoad(), UrlRequest::getMediaType() a Template::getStylesheets().

```
884                                     {
885         //     načtení doby zpracovávání aplikace
886         List ($usec, $sec) = Explode (' ', microtime());
887         $endTime = ((float)$sec + (float)$usec);
```

```
888         $this->coreTpl->addVar("MAIN_EXEC_TIME", round($endTime-$this->_stratTime, 4));
889         $this->coreTpl->addVar("COUNT_ALL_SQL_QUERY", Db::getCountQueries());
890
891
892         //     Přiřazení popisků do šablony
893         $this->assignEngineLabelsToTpl();
894
895         //     Přiřazení javascriptů a stylů
896         $this->assignVarToTpl("STYLESHEETS", Template::getStylesheets());
897         $this->assignVarToTpl("JAVASCRIPTS", Template::getJavaScripts());
898         $this->assignVarToTpl("ON_LOAD_JS_FUNCTIONS", Template::getJsOnLoad());
899
900         //     Přiřazení proměných z hlavní šablony
901         $this->assginTplObjToTpl($this->coreTpl);
902
903
907         //     zvolení vzhledu
908         //     vybraný vzhled šablony
909         if(file_exists(self::getTepmlateFaceDir().self::TEMPLATES_DIR.DIRECTORY_SEPARATOR.'index.tpl')
910             $faceFilePath = self::getTepmlateFaceDir().self::TEMPLATES_DIR.DIRECTORY_SEPARATOR;
911         }
912         //     Výchozí vzhled
913         else if(file_exists(self::getTepmlateDefaultFaceDir().self::TEMPLATES_DIR.DIRECTORY_SEPARATOR.
914             $faceFilePath = self::getTepmlateDefaultFaceDir().self::TEMPLATES_DIR.DIRECTORY_SEPARATOR;
915         }
916         //     Vzhled v engine
917         else {
918             $faceFilePath = '';
919         }
920
921         //     Zvolení zobrazovaného média
922         //     Medium pro tisk
923         if(UrlRequest::getMediaType() == UrlRequest::MEDIA_TYPE_PRINT){
924             $this->template->display($faceFilePath."index-print.tpl");
925         }
926         //     Specielní media
927         else if(file_exists($faceFilePath."index-".UrlRequest::getMediaType().".tpl")){
928             $this->template->display($faceFilePath."index-".UrlRequest::getMediaType().".tpl");
929         }
930         //     Výchozí médium (www)
931         else {
932             $this->template->display($faceFilePath."index.tpl");
933         }
934
935
936         //     switch (AppCore::media()) {
937         //        case AppCore::MEDIA_PRINT_TYPE:
938         //           $this->template->display($faceFilePath."index-print.tpl");
939         //        break;
940         //
941         //        default:
942         //           $this->template->display($faceFilePath."index.tpl");
943         //        break;
944         //     }
945
946         //     Při debug levelu vyšším jak 3 zobrazit výpis šablony
947         if(self::$debugLevel >= 3){
948             echo "<pre style='text-align: left'>";
949             print_r($this->template->get_template_vars());
950             echo "</pre>";
951         }
952     }
```

Tato funkce volá...



8.2.2.13 AppCore::runModules ()

Metoda spouští moduly //TODO.

Plánované úpravy

– implementovat do modelu

Definice je uvedena na řádku 978 v souboru app.php.

Odkazuje se na Locale::bindTextDomain(), Locale::getDefaultLang(), Category::getId(), Category::getLabel(), Locale::getLang(), Locale::switchToEngineTexts() a Locale::switchToModuleTexts().

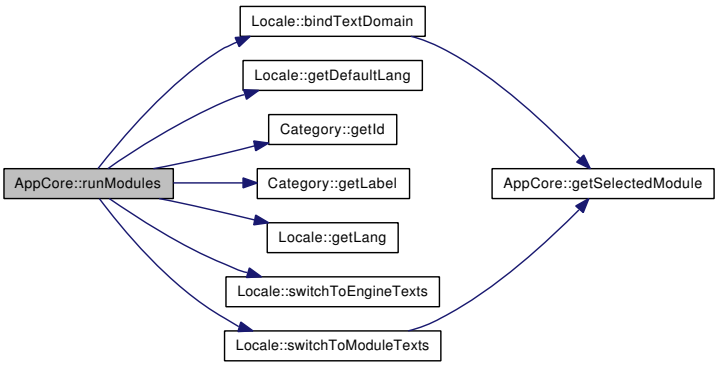
```
978                                 {
979         //načtení dat z db
980         $itemsTable = self::sysConfig()->getOptionValue("items_table", Config::SECTION_DB_TABLES);
981         $modulesTable = self::sysConfig()->getOptionValue("modules_table", Config::SECTION_DB_TABLES);
982         $userNameGroup = $this->auth->getGroupName();
983 
984         //     Vytvoření dotazu pro db
988         $catItemsSelect = self::$dbConnector->select()
989         ->from(array("item" => $itemsTable), array("label" => "IFNULL(item.label_".Locale::getLang()."
990                                             "alt" => "IFNULL(item.alt_".Locale::getLang().", item.alt_
991         ->join(array("module" => $modulesTable), "item.id_module=module.id_module", null)
992         ->where("item.".Rights::RIGHTS_GROUPS_TABLE_PREFIX.$userNameGroup." LIKE \"r__\"")
993         ->where("item.id_category = '".Category::getId()."'", "and")
994         ->order("item.priority", "desc")
995         ->order("label");
996 
997 
998         $items = self::$dbConnector->fetchObjectArray($catItemsSelect);
999 
1000         //     procházení prvků stránky
1001         if($items != null){
1002             //  Nastavení prováděné kategorie
1003             self::$currentCategory = array(Category::COLUM_CAT_LABEL => Category::getLabel()
1004                 , Category::COLUM_CAT_ID => Category::getId());
1005             foreach ($items as $itemIndex => $item) {
1006                 //         Vytvoření objektu šablony
1007                 $template = new Template();
1008 
1009                 //            Vytvoření objektu pro práci s modulem
1010                 $module = new Module($item, $this->getModuleTables($item));
1011                 //            klonování modulu do statické proměné enginu
1012                 self::$selectedModule = clone $module;
1013 
1014                 //            vytvoření pole se skupinama a právama
1015                 $userRights = array();
1016                 foreach ($item as $collum => $value){
```

```
1017                    if (substr($collum,0,strlen(Rights::RIGHTS_GROUPS_TABLE_PREFIX)) == Rights::RIGHT
1018                        $userRights[substr($collum,strlen(Rights::RIGHTS_GROUPS_TABLE_PREFIX), strlen
1019                    }
1020                }
1021                //            Vytvoření objektu pro přístup k právům modulu
1022                $moduleRights = new Rights($this->auth, $userRights);
1023
1024
1025                //            načtení souboru s akcemi modulu
1026                if(file_exists('.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR .
1027                    include_once '.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR
1028                } else {
1029                    new CoreException(_("Nepodařilo se nahrát akci modulu ") . $module->getName(), 12
1030                }
1031
1032                //            Vytvoření objektu akce
1033                $action = null;
1034                $actionClassName = ucfirst($module->getName()).'Action';
1035                if(class_exists($actionClassName)){
1036                    $action = new $actionClassName();
1037                } else {
1038                    $action = new Action();
1039                }
1040
1041                //            načtení souboru s cestami (routes) modulu
1042                if(file_exists('.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR .
1043                    include_once '.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR
1044                } else {
1045                    new CoreException(_("Nepodařilo se nahrát cestu modul ") . $module->getName(), 10
1046                }
1047
1048                //            Vytvoření objektu cesty (routes)
1049                $routes = null;
1050                $routesClassName = ucfirst($module->getName()).'Routes';
1051                if(class_exists($routesClassName)){
1052                    $routes = new $routesClassName();
1053                } else {
1054                    $routes = new Routes();
1055                }
1056
1057                //            Vytvoření ubjektu UrlReqestu
1058                $urlRequest = new UrlRequest($action, $routes);
1059
1060                //            načtení souboru s kontrolerem modulu
1061                if(file_exists('.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR .
1062                    require_once '.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATOR
1063                } else {
1064                    new CoreException(_("Nepodařilo se nahrát controler modulu ") . $module->getName(
1065                }
1066
1067                //         Vytvoření objektu kontroleru
1068                $controllerClassName = ucfirst($module->getName()).'Controller';
1069                if(class_exists($controllerClassName)){
1070                    //             Vytvoření objektu kontroleru
1071                    $controller = new $controllerClassName($action, $routes, $moduleRights);
1072
1073                    //             Volba metody kontroluru podle urlrequestu
1074                    $requestName = $urlRequest->choseController();
1075                    $requestControllerName = $requestName.AppCore::MODULE_CONTROLLER_SUFIX;
1076
1077                    //             Příprava a nastavení použití překladu
1078                    Locale::bindTextDomain($module->getName());
1079
1080                    //             Zvolení překladu na modul
1081                    Locale::switchToModuleTexts($module->getName());
1082
1083                    //             Spuštění kontroleru
```

```
1084                    if(method_exists($controller, $requestControllerName)){
1085                        $ctrlResult = $controller->$requestControllerName();
1086                    } else {
1087                        $ctrlResult = $controller->mainController();
1088                        // Vrácení překladu na engine
1089                        Locale::switchToEngineTexts();
1090                        new CoreException(_("Action Controller ").$requestControllerName._(" v modulu
1091                    }
1092
1093                    //        načtení souboru s viewrem modulu
1094                    if(file_exists('.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARATO
1095                        require_once '.' . DIRECTORY_SEPARATOR . self::MODULES_DIR . DIRECTORY_SEPARA
1096                    } else {
1097                        Locale::switchToEngineTexts();
1098                        new CoreException(_("Nepodařilo se nahrát viewer modulu ") . $module->getName
1099                    }
1100
1101                    //              Donastavení šablon
1102                    $template->setModule($module);
1103
1104                    //              Spuštění viewru
1105                    //              Pokud proběhl kontroler v pořádku
1106                    //Není-li v kontroleru přiřazen výstup
1107                    if($ctrlResult === null){
1108                        $ctrlResult = true;
1109                    }
1110
1111                    //              Spuštění phledu
1112                    if($ctrlResult){
1113                        $controller->runView($template, $requestName.AppCore::MODULE_VIEWER_SUFIX);
1114                    } else {
1115                        new CoreException(_('Controler modulu "') . $module->getName()
1116                            . _('" nebyl korektně proveden'), 21);
1117                    }
1118
1119                    //              Uložení šablony a proměných do hlavní šablony
1120                    $this->assginTplObjToTpl($template, 'MODULES_TEMPLATES');
1121
1122                    //  Vrácení překladu na engine
1123                    Locale::switchToEngineTexts();
1124                    unset($template);
1125                    unset($controller);
1126                    $isModuleControlerRun = true;
1127                } else {
1128                    new CoreException(_("Nepodařilo se vytvořit objekt controleru modulu ") . $module
1129                }
1130
1131                //            Vrácení překladu na engine pro jistotu
1132                Locale::switchToEngineTexts();
1133
1134                //            přepnutí překladu na engine
1135                textdomain(self::GETTEXT_DEFAULT_DOMAIN);
1136
1137                //            odstranění proměných
1138                unset($module);
1139                self::$selectedModule = null;
1140                unset($model);
1141                unset($action);
1142                unset($routes);
1143                unset($controller);
1144                unset($actionCtrl);
1145            }
1146        } else {
1147            if(new Links() == new Links(true)){
1148                new CoreException(_("Nepodařilo se nahrát prvky kategorie z databáze."), 9);
1149            } else {
1150                $redir = new Links(true);
```

```
1151                 $redir->category()->action()->article()->rmParam()->reload();
1152             }
1153         }
1154     }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/app.php

8.3 Dokumentace třídy Article

Třída pro práci s článekem (article).

Veřejné metody

- __construct ()

  Konstruktor nastaví klíč článku z url.

- getArticle ()

  Metoda vrací id článku.

- createUrl ($label, $id)

  Metoda vytvoří řetězec článku pro url.

- isArticle ()

  Metoda vrací true pokud je článek nastaven.

- __toString ()

  Magická metoda pro vrácení stringu.

Statické veřejné metody

- static setCurrentArticleId ($id)

  Metoda nastaví id aktuálního článku.

Veřejné atributy

- const **ARTICLE_URL_SEPARATOR** = '-'

8.3.1 Detailní popis

Třída pro práci s článekem (article).

Třída obsluhuje parametr článku, který je přenášen v URL. Umožňuje přístup přímo k názvu článku. Je propojena s Třídou routes, protože cesta se odvozuje od názvu článku.

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro obsluhu článku přenášeného v URL

Definice je uvedena na řádku 15 v souboru article.class.php.

8.3.2 Dokumentace k metodám

8.3.2.1 static Article::setCurrentArticleId ($ *id*) [static]

Metoda nastaví id aktuálního článku.

Parametry:

integer $id – id článku

Definice je uvedena na řádku 39 v souboru article.class.php.

Používá se v Links::checkArticleUrlRequest().

```
39                                                  {
40        self::$currentArticleId = $id;
41    }
```

8.3.2.2 Article::getArticle ()

Metoda vrací id článku.

Návratová hodnota:

integer

Definice je uvedena na řádku 47 v souboru article.class.php.

```
47                                  {
48        return (int)self::$currentArticleId;
49    }
```

8.3.2.3 Article::createUrl ($ *label*, $ *id*)

Metoda vytvoří řetězec článku pro url.

Parametry:

<*type*> $label

<*type*> $id

Definice je uvedena na řádku 56 v souboru article.class.php.

```
56                                          {
57       $textHelp = new TextCtrlHelper();
58       return $textHelp->utf2ascii($label).self::ARTICLE_URL_SEPARATOR.$id;
59    }
```

8.3.2.4 Article::isArticle ()

Metoda vrací true pokud je článek nastaven.

Návratová hodnota:

boolean – true při nastavení článku

Definice je uvedena na řádku 65 v souboru article.class.php.

```
65                                {
66       if(self::$currentArticleId != null){
67          return true;
68       }
69       return false;
70    }
```

8.3.2.5 Article::__toString ()

Magická metoda pro vrácení stringu.

Návratová hodnota:

string – article

Definice je uvedena na řádku 76 v souboru article.class.php.

```
76                                  {
77       return (string)self::$currentArticleId;
78    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/article.class.php

8.4 Dokumentace třídy Auth

Třída pro autorizaci.

Veřejné metody

- __construct ($dbConnector, Messages &$errors)

  Konstruktor, provádí autorizaci.

- isLogin ()

  Metoda vrací je-li uživatel přihlášen.

- getGroupName ()

  Metoda vrací název skupiny ve které je uživatel.

- getGroupId ()

  Metoda vrací id skupiny ve které je uživatel.

- getUserId ()

  Metoda vrací id uživatele.

- getUserName ()

  Metoda vrací název uživatele.

- getUserMail ()

  Metoda vrací mail uživatele.

Statické veřejné metody

- static isLoginStatic ()

  Metoda vrací jestli je uživatel přihlášen.

Veřejné atributy

- const **CONFIG_USERS_TABLE_NAME** = 'users_table'
- const **CONFIG_GROUPS_TABLE_NAME** = 'groups_table'
- const **USER_NAME** = 'username'
- const **USER_MAIL** = 'mail'
- const **USER_ID** = 'id_user'
- const **USER_ID_GROUP** = 'id_group'
- const **USER_GROUP_NAME** = 'group_name'
- const **USER_LOGIN_TIME** = 'logintime'
- const **USER_IS_LOGIN** = 'login'
- const **USER_LOGIN_ADDRESS** = 'ip_address'

8.4.1 Detailní popis

Třída pro autorizaci.

Třída obsluhuje přihlášení/odhlášení uživatele a práci s vlastnostmi (jméno, email, id, skupinu, atd.) přihlášeného uživatele.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: auth.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu autorizace uživatele

Definice je uvedena na řádku 13 v souboru auth.class.php.

8.4.2 Dokumentace k metodám

8.4.2.1 Auth::isLogin ()

Metoda vrací je-li uživatel přihlášen.

Návratová hodnota:

boolean – je li uživatel přihlášen

Definice je uvedena na řádku 273 v souboru auth.class.php.

```
273                         {
274         return $this->login;
275     }
```

8.4.2.2 static Auth::isLoginStatic () [static]

Metoda vrací jestli je uživatel přihlášen.

Návratová hodnota:

boolean – true pokud je uživatel přihlášen

Definice je uvedena na řádku 281 v souboru auth.class.php.

```
281                                         {
282         return self::$isLogIn;
283     }
```

8.4.2.3 Auth::getGroupName ()

Metoda vrací název skupiny ve které je uživatel.

Návratová hodnota:

string – název skupiny

Definice je uvedena na řádku 290 v souboru auth.class.php.

```
290                                 {
291         return $this->userGroupName;
292     }
```

8.4.2.4 Auth::getGroupId ()

Metoda vrací id skupiny ve které je uživatel.

Návratová hodnota:

integer – id skupiny

Definice je uvedena na řádku 298 v souboru auth.class.php.

```
298                                 {
299         return $this->userGroupId;
300     }
```

8.4.2.5 Auth::getUserId ()

Metoda vrací id uživatele.

Návratová hodnota:

integer – id uživatele

Definice je uvedena na řádku 306 v souboru auth.class.php.

```
306                                 {
307         return $this->userId;
308     }
```

8.4.2.6 Auth::getUserName ()

Metoda vrací název uživatele.

Návratová hodnota:

string – název uživatele

Definice je uvedena na řádku 314 v souboru auth.class.php.

```
314                                 {
315         return $this->userName;
316     }
```

8.4.2.7 Auth::getUserMail ()

Metoda vrací mail uživatele.

Návratová hodnota:

string – mail uživatele

Definice je uvedena na řádku 322 v souboru auth.class.php.

```
322                                 {
323         return $this->userMail;
324     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/auth.class.php

8.5 Dokumentace třídy BlogAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu BlogAction

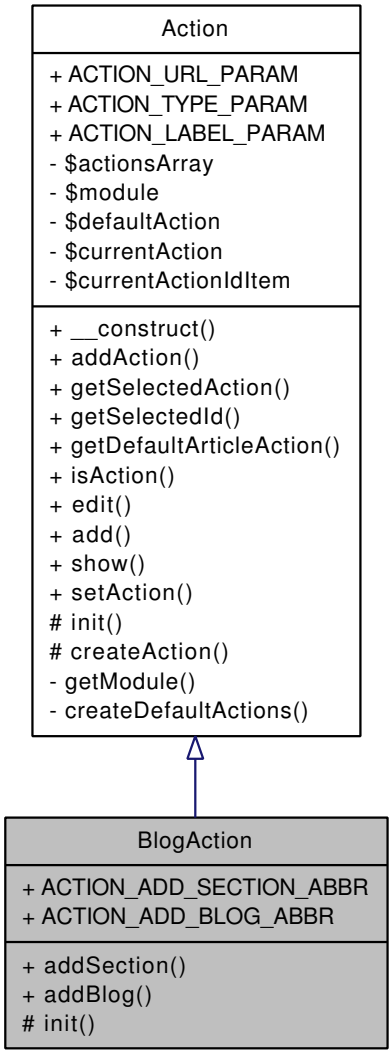
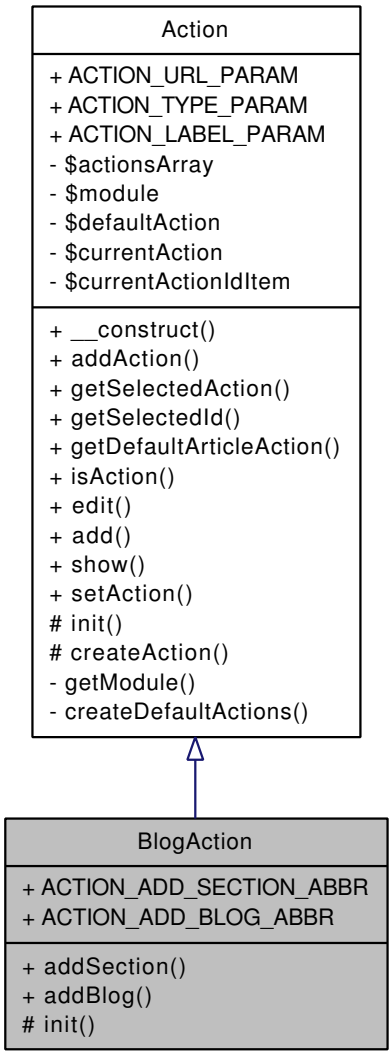


Diagram tříd pro BlogAction:



Veřejné metody

- **addSection** ()
- **addBlog** ()

Veřejné atributy

- const **ACTION_ADD_SECTION_ABBR** = 'as'
- const **ACTION_ADD_BLOG_ABBR** = 'ab'

Chráněné metody

- init ()

  Metoda pro inicializaci akcí.

8.5.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/blog/action.class.php

8.6 Dokumentace třídy BlogController

Controler modulu Blogu Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu BlogController

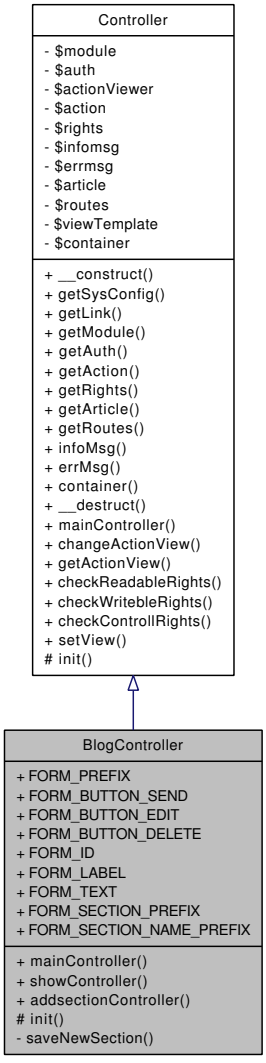
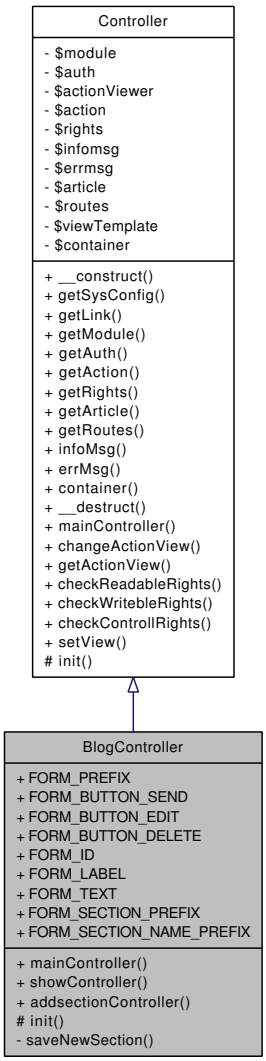


Diagram tříd pro BlogController:



Veřejné metody

- mainController ()

  Kontroler pro obrazení seznamu blogů.

- **showController** ()
- addsectionController ()

  Kontroler pro přidání sekce.

Veřejné atributy

- const **FORM_PREFIX** = ’blog_’
- const **FORM_BUTTON_SEND** = ’send’

- const **FORM_BUTTON_EDIT** = 'edit'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **FORM_ID** = 'id'
- const **FORM_LABEL** = 'label_'
- const **FORM_TEXT** = 'text_'
- const **FORM_SECTION_PREFIX** = 'section_'
- const **FORM_SECTION_NAME_PREFIX** = 'label_'

Chráněné metody

- init ()

  Inicializační metoda pro kontroler.

8.6.1 Detailní popis

Controler modulu Blogu Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 8 v souboru controler.class.php.

8.6.2 Dokumentace k metodám

8.6.2.1 BlogController::init () [protected]

Inicializační metoda pro kontroler.

je spuštěna vždy při vytvoření objektu kontroleru

Reimplementuje stejnojmenný prvek z Controller.

Definice je uvedena na řádku 83 v souboru controler.class.php.

```
83                                  {
84 //          $this->pokusparam = new UrlParam('pokus', '([0-9]+)');
85 //          if($this->pokusparam->isValue()){
86 //              echo "TADY.....................".$this->pokusparam->getValue()."<br>";
87 //          };
88      }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/blog/controler.class.php

8.7 Dokumentace třídy BlogRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu BlogRoutes

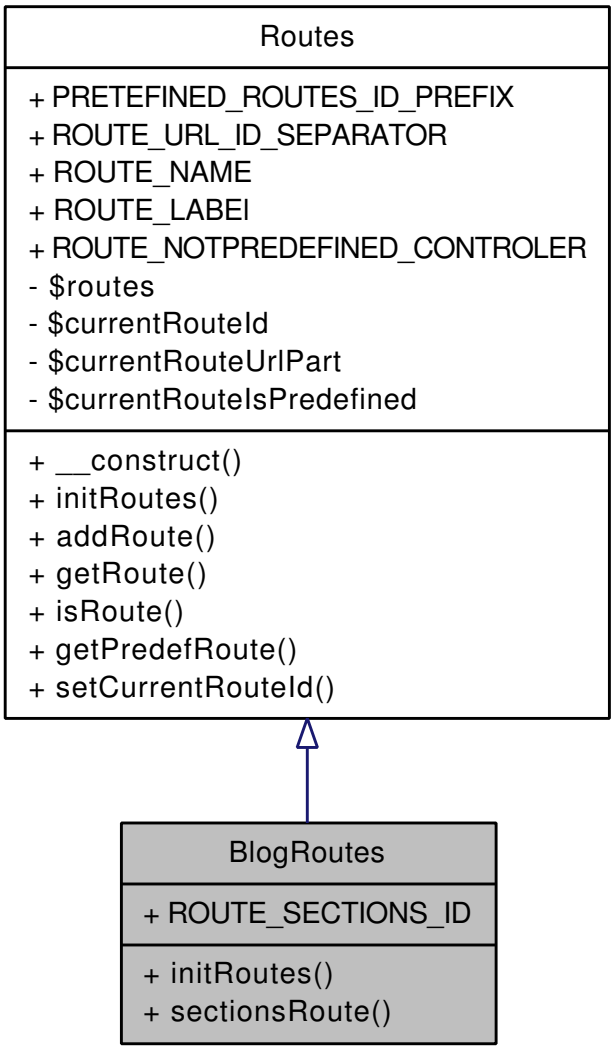
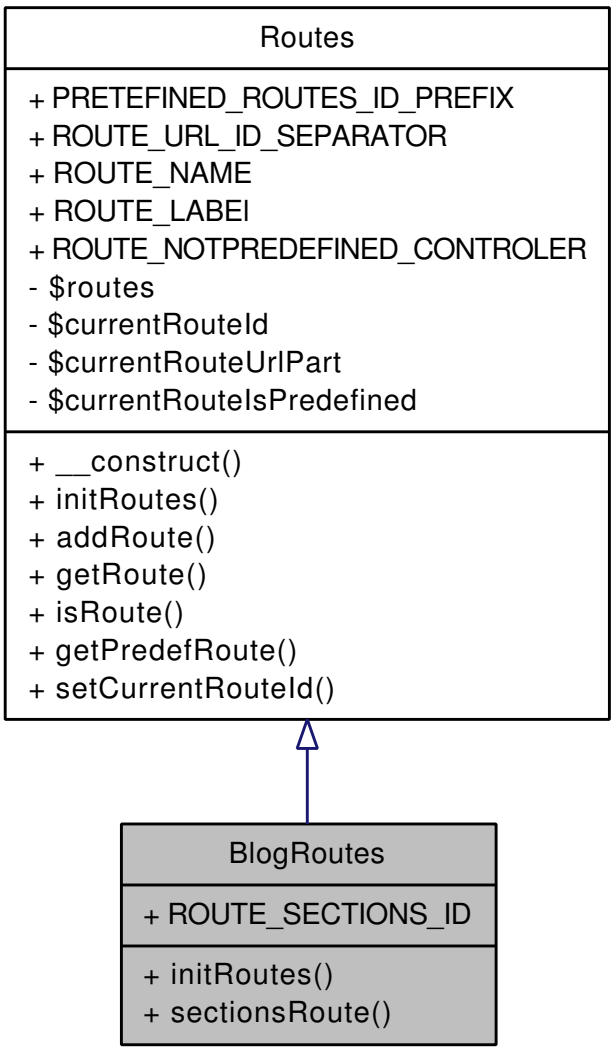


Diagram tříd pro BlogRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

- **sectionsRoute** ()

Veřejné atributy

- const **ROUTE_SECTIONS_ID** = 1

8.7.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/blog/routes.class.php

8.8 Dokumentace třídy BlogView

Viewer modulu blogu Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu BlogView

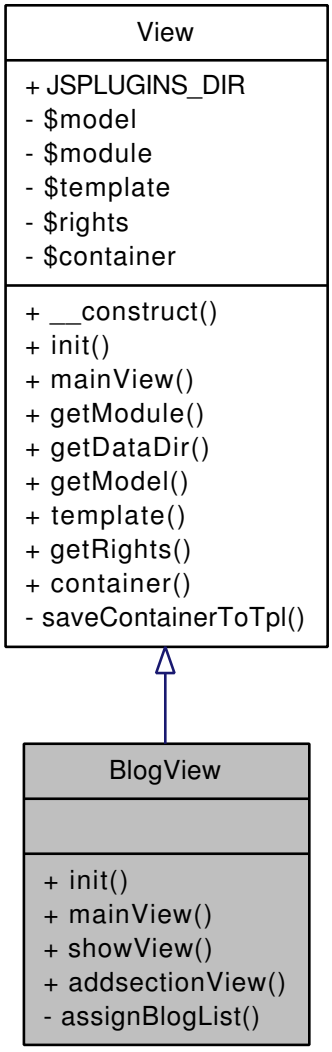
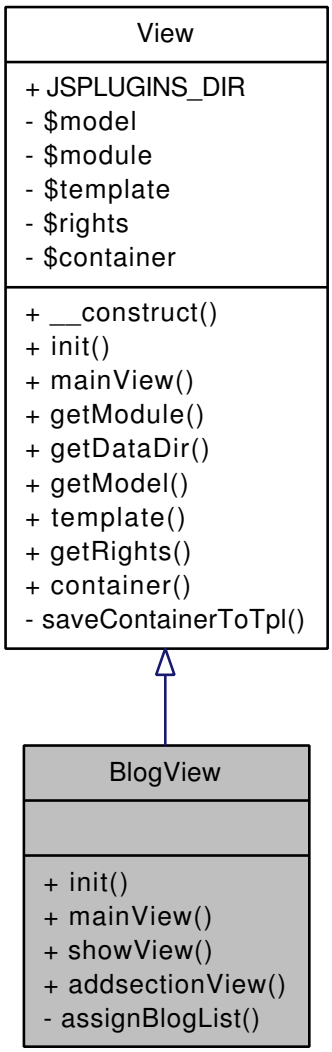


Diagram tříd pro BlogView:



Veřejné metody

- init ()

  Inicializace.

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- **showView** ()
- addsectionView ()

  Viewer pro přidání sekce.

8.8.1 Detailní popis

Viewer modulu blogu Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 8 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/blog/view.class.php

8.9 Dokumentace třídy Category

Třída obsluhuje práci se zvolenou kategorií.

Statické veřejné metody

- static factory (Auth $auth)

  konstruktor načte informace o kategorii

- static setCurrentCategoryId ($id)

  Nastavuje id aktuální kategorie.

- static getCurrentCategory ()

  Metoda vrací část url adresy s kategorií.

- static getDefaultCategory ()

  Metoda načte informace o výchozí kategorii.

- static isDefault ()

  Metoda vrací true pokud vybraná kategorie je výchozí kategorií.

- static getLabel ()

  Metoda vrací název kategorie.

- static getId ()

  Metoda vrací id kategorie.

- static getSectionId ()

  Metoda vrací id sekce kategorie.

- static getSectionLabel ()

  Metoda vrací název sekce kategorie.

- static getUrlKey ()

  Metoda vrací urlkey kategorie.

- static isLeftPanel ()

  Metoda vrací jesli je zapnut levý panel.

- static isRightPanel ()

  Metoda vrací jesli je zapnut pravý panel.

- static getParam ($param)

  Metoda vrací požedovaný parametr.

Veřejné atributy

- const **CAT_PARAMS_SEPARATOR** = ';'
- const **COLUM_CAT_LABEL** = 'clabel'
- const **COLUM_SEC_LABEL** = 'slabel'
- const **COLUM_CAT_ID** = 'id_category'
- const **COLUM_SEC_ID** = 'id_section'
- const **COLUM_CAT_URLKEY** = 'urlkey'
- const **COLUM_CAT_LPANEL** = 'left_panel'
- const **COLUM_CAT_RPANEL** = 'right_panel'
- const **COLUM_CAT_PARAMS** = 'params'
- const **COLUM_CAT_SHOW_IN_MENU** = 'show_in_menu'
- const **COLUM_CAT_PROTECTED** = 'protected'

8.9.1 Detailní popis

Třída obsluhuje práci se zvolenou kategorií.

Třída umožňuje základní přístu k vlastnostem kategorie a volbu jejího obsahu podle práv uživatele. Načítá také kategorii, která je výchozí nebo zvolená.

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro obsluhu zvolené kategorie

Definice je uvedena na řádku 14 v souboru category.class.php.

8.9.2 Dokumentace k metodám

8.9.2.1 static Category::factory (Auth $ *auth*) [static]

konstruktor načte informace o kategorii

Parametry:

Db object – konektor k databázi

string – název $_GET proměné s klíčem kategorie

Definice je uvedena na řádku 112 v souboru category.class.php.

Odkazuje se na AppCore::getDbConnector().

Používá se v AppCore::selectCategory().

```
112                                                 {
113       //    nastavení db
114       self::$_dbConnector = AppCore::getDbConnector();
115       self::$_auth = $auth;
116
117       //    vbrání kategorie
118       if(self::$currentCategoryId != null){
119          self::_loadSelectedFromDb(self::$currentCategoryId);
120       } else {
121          self::_loadDefaultFromDb();
122       }
123    }
```

Tato funkce volá...

8.9.2.2 static Category::setCurrentCategoryId ($ *id*) [static]

Nastavuje id aktuální kategorie.

Parametry:

integer $id – id aktuální kategorie

Definice je uvedena na řádku 129 v souboru category.class.php.

Používá se v Links::checkCategoryUrlRequest().

```
129                                                     {
130       self::$currentCategoryId = $id;
131    }
```

8.9.2.3 static Category::getCurrentCategory () [static]

Metoda vrací část url adresy s kategorií.

Návratová hodnota:

array – pole s částmi pro URL

Definice je uvedena na řádku 137 v souboru category.class.php.

```
137                                                 {
138         $array = array(Links::LINK_ARRAY_ITEM_ID => self::$currentCategoryId,
139                        Links::LINK_ARRAY_ITEM_NAME => self::$currentCategoryName);
140         return $array;
141     }
```

8.9.2.4 static Category::getDefaultCategory () [static]

Metoda načte informace o výchozí kategorii.

Návratová hodnota:

array – pole s prvky výchozí kategorie

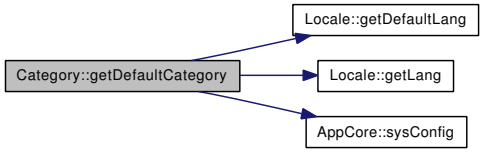
Definice je uvedena na řádku 244 v souboru category.class.php.

Odkazuje se na Locale::getDefaultLang(), Locale::getLang() a AppCore::sysConfig().

```
244                                                 {
245         $catTable = AppCore::sysConfig()->getOptionValue("category_table", "db_tables");
246         $secTable = AppCore::sysConfig()->getOptionValue("section_table", "db_tables");
247         $itemsTable = AppCore::sysConfig()->getOptionValue("items_table", "db_tables");
248         $userNameGroup = self::$_auth->getGroupName();
249 
250 
251         $catSelect = self::$_dbConnector->select()->from(array("cat" => $catTable),
252            array("clabel" => "IFNULL(cat.label_".Locale::getLang().", cat.label_"
253            .Locale::getDefaultLang().")", "id_category", self::COLUM_CAT_LPANEL,
254            self::COLUM_CAT_RPANEL, self::COLUM_SEC_ID, self::COLUM_CAT_PARAMS))
255         ->join(array("item" => $itemsTable), "cat.id_category = item.id_category", "inner", null)
256         ->join(array("sec" => $secTable), "cat.id_section = sec.id_section", "inner",
257            array("slabel" => "IFNULL(sec.label_".Locale::getLang().", sec.label_"
258            .Locale::getDefaultLang().")"))
259         ->where("item.".Rights::RIGHTS_GROUPS_TABLE_PREFIX.$userNameGroup." LIKE \"r__\"")
260         ->where("cat.active = 1", "and")
261         ->order("sec.priority", "desc")
262         ->order("cat.priority", "desc")
263         ->order("clabel")
264         ->limit(0,1);
265 
266         $catArray = self::$_dbConnector->fetchObject($catSelect,true);
267 
268         //    Pokud nebyla načtena žádná kategorie
269         if(empty($catArray)){
270            throw new CoreException(_('Nepodařilo se načíst výchozí kategorii. Chyba v konfiguraci.'),2);
271         }
272 
273         $catArr = array ();
274         $catArr[self::COLUM_CAT_LABEL] = $catArray->{self::COLUM_CAT_LABEL};
275         $catArr[self::COLUM_CAT_ID] = $catArray->{self::COLUM_CAT_ID};
276         $catArr[self::COLUM_SEC_ID] = $catArray->{self::COLUM_SEC_ID};
277 //      $catArr[self::COLUM_CAT_URLKEY] = $catArray->{self::COLUM_CAT_URLKEY};
278         $catArr[self::COLUM_CAT_LPANEL] = $catArray->{self::COLUM_CAT_LPANEL};
279         $catArr[self::COLUM_CAT_RPANEL] = $catArray->{self::COLUM_CAT_RPANEL};
280         $catArr[self::COLUM_SEC_LABEL] = $catArray->{self::COLUM_SEC_LABEL};
281         $catArr[self::COLUM_CAT_PARAMS] = self::parseParams($catArray->{self::COLUM_CAT_PARAMS});
```

```
282
283         return $catArr;
284     }
```

Tato funkce volá...



8.9.2.5 static Category::isDefault () [static]

Metoda vrací true pokud vybraná kategorie je výchozí kategorií.

Návratová hodnota:

boolean – true pokud je výchozí kategorie

Definice je uvedena na řádku 290 v souboru category.class.php.

```
290                                     {
291         return self::$_categoryIsDefault;
292     }
```

8.9.2.6 static Category::getLabel () [static]

Metoda vrací název kategorie.

Návratová hodnota:

string – název kategorie

Definice je uvedena na řádku 317 v souboru category.class.php.

Používá se v Controller::getLink(), AppCore::runModules() a AppCore::selectCategory().

```
317                                     {
318         return self::$_categoryLabel;
319     }
```

8.9.2.7 static Category::getId () [static]

Metoda vrací id kategorie.

Návratová hodnota:

integer – id kategorie

Definice je uvedena na řádku 325 v souboru category.class.php.

Používá se v Controller::getLink() a AppCore::runModules().

```
325                                     {
326         return self::$_categoryId;
327     }
```

8.9.2.8 static Category::getSectionId () [static]

Metoda vrací id sekce kategorie.

Návratová hodnota:

integer – id sekce kategorie

Definice je uvedena na řádku 333 v souboru category.class.php.

```
333                                             {
334         return self::$_sectionId;
335     }
```

8.9.2.9 static Category::getSectionLabel () [static]

Metoda vrací název sekce kategorie.

Návratová hodnota:

string – název sekce kategorie

Definice je uvedena na řádku 341 v souboru category.class.php.

```
341                                               {
342         return self::$_sectionName;
343     }
```

8.9.2.10 static Category::getUrlKey () [static]

Metoda vrací urlkey kategorie.

Návratová hodnota:

string – urlkey kategorie

Definice je uvedena na řádku 349 v souboru category.class.php.

```
349                                       {
350         return self::$_categoryUrlkey;
351     }
```

8.9.2.11 static Category::isLeftPanel () [static]

Metoda vrací jesli je zapnut levý panel.

Návratová hodnota:

boolena – true pokud je panel zapnut

Definice je uvedena na řádku 357 v souboru category.class.php.

Používá se v AppCore::runApp().

```
357                                     {
358       return self::$_categoryLeftPanel;
359     }
```

8.9.2.12 static Category::isRightPanel () [static]

Metoda vrací jesli je zapnut pravý panel.

Návratová hodnota:

boolena – true pokud je panel zapnut

Definice je uvedena na řádku 365 v souboru category.class.php.

Používá se v AppCore::runApp().

```
365                                      {
366       return self::$_categoryRightPanel;
367     }
```

8.9.2.13 static Category::getParam ($ *param*) [static]

Metoda vrací požedovaný parametr.

Parametry:

string – index parametru

Návratová hodnota:

string – parametr

Definice je uvedena na řádku 374 v souboru category.class.php.

```
374                                              {
375       if(isset(self::$_categoryParams[$param])){
376          return self::$_categoryParams[$param];
377       } else {
378          return null;
379       }
380     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/category.class.php

8.10 Dokumentace třídy ChangesController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu ChangesController

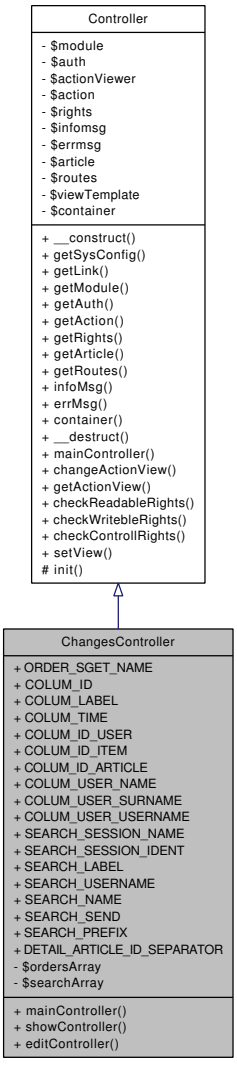
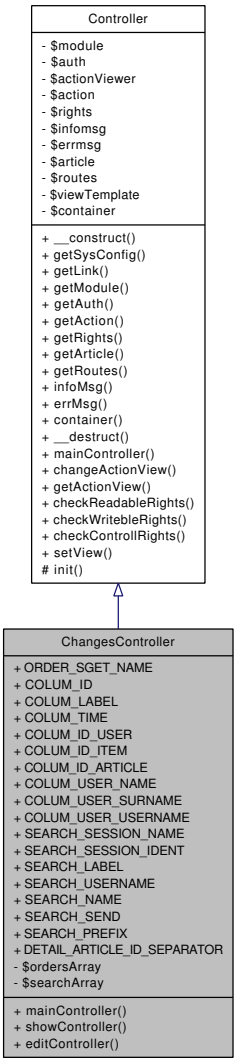


Diagram tříd pro ChangesController:



Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- showController ()

  Metoda pro zobrazení detailu zástupce.

- editController ()

  Metoda pro úpravu.

Veřejné atributy

- const **ORDER_SGET_NAME** = 'order'

- const **COLUM_ID** = 'id_parent'
- const **COLUM_LABEL** = 'label'
- const **COLUM_TIME** = 'time'
- const **COLUM_ID_USER** = 'id_user'
- const **COLUM_ID_ITEM** = 'id_item'
- const **COLUM_ID_ARTICLE** = 'id_article'
- const **COLUM_USER_NAME** = 'name'
- const **COLUM_USER_SURNAME** = 'surname'
- const **COLUM_USER_USERNAME** = 'username'
- const **SEARCH_SESSION_NAME** = 'search'
- const **SEARCH_SESSION_IDENT** = 'changes'
- const **SEARCH_LABEL** = 'label'
- const **SEARCH_USERNAME** = 'username'
- const **SEARCH_NAME** = 'name'
- const **SEARCH_SEND** = 'send'
- const **SEARCH_PREFIX** = 'changes_search_'
- const **DETAIL_ARTICLE_ID_SEPARATOR** = '-'

8.10.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/changes/controler.class.php

8.11 Dokumentace třídy ChangesEPlugin

EPlugin pro sledování změn ve stránce.

Diagram dědičnosti pro třídu ChangesEPlugin

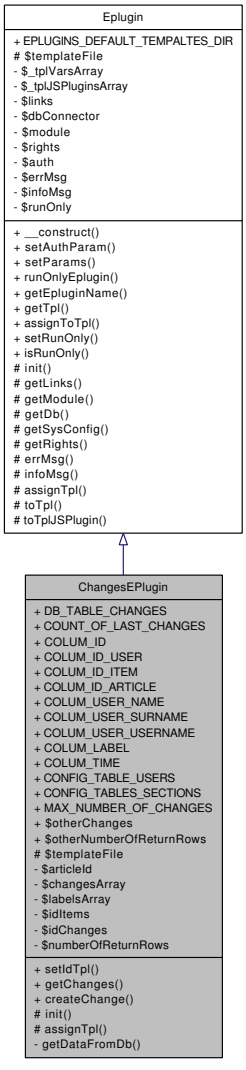
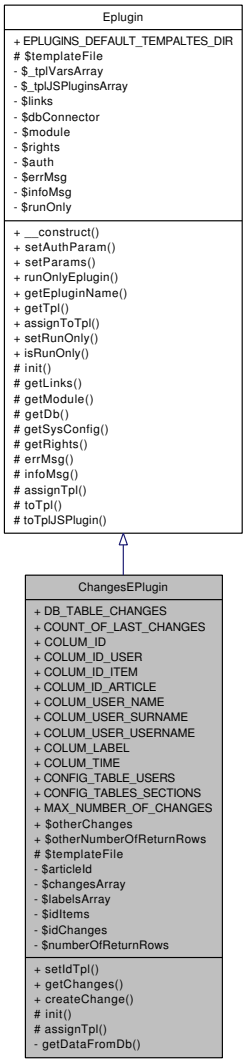


Diagram tříd pro ChangesEPlugin:



Veřejné metody

- setIdTpl ($id)

  Metoda nastaví id šablony pro výpis.

- getChanges ($idArticles=null, $idItems=null)

  Metoda vrací objekt změny s načtenými změnami u zadaných id článků.

- createChange ($label, $idArticle, $idItem=null)

  Metoda uloží změnu do db.

Veřejné atributy

- const **DB_TABLE_CHANGES** = 'changes'

- const **COUNT_OF_LAST_CHANGES** = 1
- const **COLUM_ID** = 'id_change'
- const **COLUM_ID_USER** = 'id_user'
- const **COLUM_ID_ITEM** = 'id_item'
- const **COLUM_ID_ARTICLE** = 'id_article'
- const **COLUM_USER_NAME** = 'name'
- const **COLUM_USER_SURNAME** = 'surname'
- const **COLUM_USER_USERNAME** = 'username'
- const **COLUM_LABEL** = 'label'
- const **COLUM_TIME** = 'time'
- const **CONFIG_TABLE_USERS** = 'users_table'
- const **CONFIG_TABLES_SECTIONS** = 'db_tables'
- const **MAX_NUMBER_OF_CHANGES** = 30

Statické veřejné atributy

- static **$otherChanges** = array()
- static **$otherNumberOfReturnRows** = array()

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda obstarává přiřazení proměných do šablony.

Chráněné atributy

- **$templateFile** = 'changes.tpl'

8.11.1 Detailní popis

EPlugin pro sledování změn ve stránce.

Třída slouží pro sledování provedených změn ve stránce, článku, text, atd. Změnu jsou ukládány do db, a lze s nimi dále pracovat(vyhledávat, atd.) pomocí modulu changes. Sám využívá vlastní šablonu pro zobrazení posledních změn. Používá také JsPlugin SwitchContentEasy pro zkrytí změn na stránce.

Verze:

$Id: changes.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída Epluginu pro zobrazování a logování provedených změn

Plánované úpravy

implementovat možnost zobrazení změn ve vlastním novém okně

Definice je uvedena na řádku 16 v souboru changes.class.php.

8.11.2 Dokumentace k metodám

8.11.2.1 ChangesEPlugin::setIdTpl ($ *id*)

Metoda nastaví id šablony pro výpis.

Parametry:

ineger – id šablony (jakékoliv)

Definice je uvedena na řádku 116 v souboru changes.class.php.

```
116                                 {
117         $this->idChanges = $id;
118     }
```

8.11.2.2 ChangesEPlugin::getChanges ($ *idArticles* = `null`, $ *idItems* = `null`)

Metoda vrací objekt změny s načtenými změnami u zadaných id článků.

Parametry:

mixed – array nebo integer s id článku (popřípadě podpole s id item a id článků)

Návratová hodnota:

Changes – vrací objekt Changes (tedy sebe)

Definice je uvedena na řádku 125 v souboru changes.class.php.

```
125                                                                 {
126         $this->articleId = $idArticles;
127         $this->idItems = $idItems;
128 
129         $this->getDataFromDb();
130         return $this;
131     }
```

8.11.2.3 ChangesEPlugin::createChange ($ *label*, $ *idArticle*, $ *idItem* = `null`)

Metoda uloží změnu do db.

Parametry:

string – popis změny

integer – id článku u kterého byla změna provedena

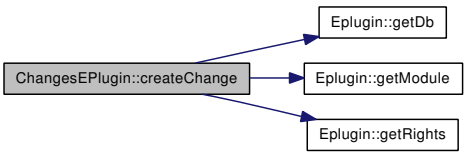
integer – (option) id item u ktré byla změna provedena

Definice je uvedena na řádku 139 v souboru changes.class.php.

Odkazuje se na Eplugin::getDb(), Eplugin::getModule() a Eplugin::getRights().

```
139                                                                    {
140       $sqlInser = $this->getDb()->insert()->into(self::DB_TABLE_CHANGES);
141
142       if($idItem == null){
143          $sqlInser = $sqlInser->colums(self::COLUM_ID_ARTICLE, self::COLUM_ID_ITEM, self::COLUM_ID_USE
144                         ->values($idArticle, $this->getModule()->getId(), $this->getRights()->getAuth(
145
146       } else {
147          $sqlInser = $sqlInser->colums(self::COLUM_ID_ARTICLE, self::COLUM_ID_ITEM, self::COLUM_ID_USE
148                         ->values($idArticle, $idItem, $this->getRights()->getAuth()->getUserId(), $lab
149       }
150
151 //    vložení záznamu
152       $this->getDb()->query($sqlInser);
153    }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/changes.class.php

8.12 Dokumentace třídy ChangesView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu ChangesView

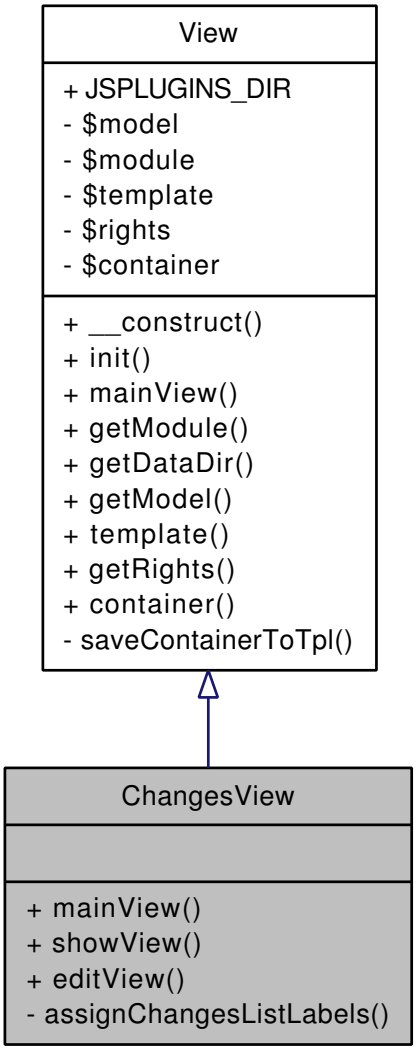
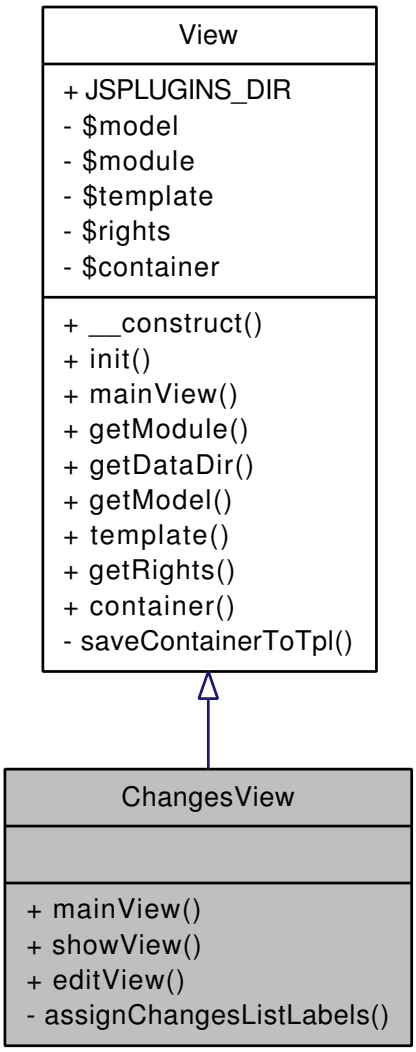


Diagram tříd pro ChangesView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- showView ()

  Viewer pro zobrazení detailu.

- **editView** ()

8.12.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/changes/view.class.php

8.13 Dokumentace třídy Config

Třída pro obsluhu konfiguračního souboru.

Veřejné metody

- __construct ($configFile, &$coreErrors)
  Konstruktor třídy.
- getOptionValue ($option, $parentKey=null)
  Vrátí požedovanou hodnotu z konfiguračního souboru.

Veřejné atributy

- const **SECTION_DB_TABLES** = 'db_tables'

8.13.1 Detailní popis

Třída pro obsluhu konfiguračního souboru.

Třída slouží k získávání parametrů z konfiguračního souboru.

Copyright (c) 2008 Jakub Matas

Verze:

$Id:config.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu konfiguračního souboru

Definice je uvedena na řádku 13 v souboru config.class.php.

8.13.2 Dokumentace konstruktoru a destruktoru

8.13.2.1 Config::__construct ($ *configFile*, &$ *coreErrors*)

Konstruktor třídy.

Parametry:

string – soubor s konfigurací

Definice je uvedena na řádku 37 v souboru config.class.php.

```
37                                                  {
38         $this->_configFile = $configFile;
39         $this->_configArray = array();
40         $this->_initConfigFile();
41     }
```

8.13.3 Dokumentace k metodám

8.13.3.1 Config::getOptionValue ($ *option*, $ *parentKey* = `null`)

Vrátí požedovanou hodnotu z konfiguračního souboru.

Parametry:

string – volba, která se má vrátit

string – skupina volby, která se má vrátit

Plánované úpravy

dodělat vracení z větší hloubky stromu

Definice je uvedena na řádku 95 v souboru config.class.php.

```
95                                                              {
96       $return = null;
97       if($parentKey == null){
98             if(!isset($this->_configArray[$option])){
99                $error = new CoreException(_("Volba ").$option._(" není definována"), 102);
100            } else {
101               $return = $this->_configArray[$option];
102            }
103      } else {
104            if(!isset($this->_configArray[$parentKey][$option])){
105               $error = new CoreException(_("Volba ").$option._(" v sekci ").$parentKey._("není defino
106            } else {
107               return $this->_configArray[$parentKey][$option];
108            }
109      }
110      return $return;
111   }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/config.class.php

8.14 Dokumentace třídy Container

Třída pro přenos daty mezi controlerrem a viewrem.

Veřejné metody

- __construct ()
  Konstruktor.
- addData ($index, $value)
  Metoda uloži zadaná data pod zadaný index.
- getData ($index)
  Metoda vrací data ze zadaného indexu.
- addEplugin ($index, Eplugin &$eplugin)
  Metoda uloži zadaný eplugin pod zadaný index.
- getEplugin ($index)
  Metoda vrací eplugin ze zadaného indexu.
- addLink ($index, Links $link)
  Metoda uloži zadaný link pod zadaný index.
- getLink ($index)
  Metoda vrací link ze zadaného indexu.
- getAllData ()
  Metoda vrací všechna uložená data v poli.
- getAllLinks ()
  Metoda vrací všechny uložené odkazy v poli.

8.14.1 Detailní popis

Třída pro přenos daty mezi controlerrem a viewrem.

Třída umožňuje přenášet data meti kontrolery a pohledy. Popřípadě jejich přímou prezentaci do šablony modulu.

Verze:

$Id: container.class.php 3.0.5 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu dat mezi viewrem a controlerem

Definice je uvedena na řádku 13 v souboru container.class.php.

8.14.2 Dokumentace k metodám

8.14.2.1 Container::addData ($ *index*, $ *value*)

Metoda uloži zadaná data pod zadaný index.

Parametry:

string/integer – index dat
mixed – data

Definice je uvedena na řádku 47 v souboru container.class.php.

```
47                                          {
48       //TODO dodělat kontrolu, jestli položka už není vložena
49       $this->data[$index] = $value;
50    }
```

8.14.2.2 Container::getData ($ *index*)

Metoda vrací data ze zadaného indexu.

Parametry:

string/integer – index pod kterým jsou data uložena

Návratová hodnota:

mixed – uložená data

Definice je uvedena na řádku 58 v souboru container.class.php.

```
58                                      {
59       if(isset($this->data[$index])){
60          return $this->data[$index];
61       } else {
62          return null;
63       }
64    }
```

8.14.2.3 Container::addEplugin ($ *index*, Eplugin &$ *eplugin*)

Metoda uloži zadaný eplugin pod zadaný index.

Parametry:

string/integer – index epluginu
Eplugin – objekt epluginu

Definice je uvedena na řádku 72 v souboru container.class.php.

```
72                                                      {
73       //TODO dodělat kontrolu, jestli položka už není vložena
74       $this->ePlugins[$index] = $eplugin;
75    }
```

8.14.2.4 Container::getEplugin ($ *index*)

Metoda vrací eplugin ze zadaného indexu.

Parametry:

string/integer – index pod kterým je Eplugin uložen

Návratová hodnota:

Eplugin – uložený Eplugin

Definice je uvedena na řádku 83 v souboru container.class.php.

```
83                                      {
84       if(isset($this->ePlugins[$index])){
85          return $this->ePlugins[$index];
86       } else {
87          throw new CoreException(_('Požedovaný eplugin nebyl přiřazen'));
88          return false;
89       }
90    }
```

8.14.2.5 Container::addLink ($ *index*, Links $ *link*)

Metoda uloži zadaný link pod zadaný index.

Parametry:

string/integer – index linku

Links – objekt linku

Definice je uvedena na řádku 98 v souboru container.class.php.

```
98                                                  {
99       //TODO dodělat kontrolu, jestli položka už není vložena
100       $this->links[$index] = $link;
101    }
```

8.14.2.6 Container::getLink ($ *index*)

Metoda vrací link ze zadaného indexu.

Parametry:

string/integer – index pod kterým je link uložen

Návratová hodnota:

Links – uložený link

Definice je uvedena na řádku 109 v souboru container.class.php.

```
109                                 {
110       if(isset($this->links[$index])){
111          return $this->links[$index];
112       } else {
113          return null;
114       }
115    }
```

8.14.2.7 Container::getAllData ()

Metoda vrací všechna uložená data v poli.

Návratová hodnota:

array – pole s daty

Definice je uvedena na řádku 122 v souboru container.class.php.

```
122                                 {
123       return $this->data;
124    }
```

8.14.2.8 Container::getAllLinks ()

Metoda vrací všechny uložené odkazy v poli.

Návratová hodnota:

array – pole s daty

Definice je uvedena na řádku 131 v souboru container.class.php.

```
131                                 {
132       return $this->links;
133    }
```

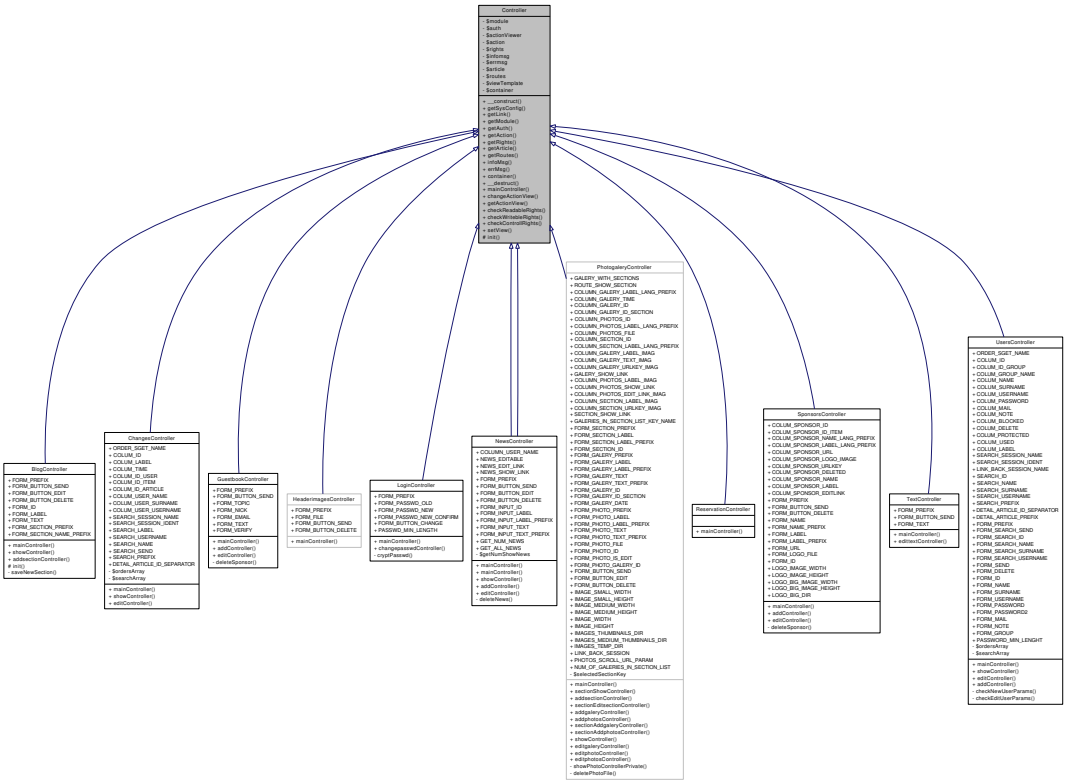
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/container.class.php

8.15 Dokumentace třídy Controller

Abstraktní třída tvorbu kontroleru modulu.

Diagram dědičnosti pro třídu Controller



Veřejné metody

- __construct (Action $action, Routes $routes, Rights $rights)

  Konstruktor třídy vytvoří a naplní základní vlastnosti pro modul.

- getSysConfig ()

  Metoda vrací objekt systémové konfigurace.

- getLink ($clear=false)

  Metoda vrací odkaz na objekt pro práci s odkazy.

- getModule ()

  Metody vrací objekt modulu.

- getAuth ()

  Metody vrací objekt autorizace.

- getAction ()

  Metoda vrací objekt na akci.

- getRights ()

  Metoda vrací objekt s právy na modul.

- getArticle ()

  Metoda vrací objekt s článkem.

- getRoutes ()

  Metoda vrací objekt s cetsami - routes.

- infoMsg ()

  Metoda vrací objekt s informačními zprávami.

- errMsg ()

  Metoda vrací objekt s chybovými zprávami.

- container ()

  Metoda vrací objekt kontaineru pro přenos dat mezi controlerem a viewrem.

- __destruct ()

  Vrací objekt modulu.

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- changeActionView ($newActionView)

  Metoda změní výchozí actionViewer na zadaný.

- getActionView ()

  Metoda vrací zvolený actionViewer nebo false pokud je nulový.

- checkReadableRights ()

  Metoda kontroluje práva pro čtení modulu.

- checkWritebleRights ()

  Metoda kontroluje práva pro zápis do modulu.

- checkControllRights ()

  Metoda kontroluje práva pro plný přístup k modulu.

- setView ($view)

  metoda nastaví název použitého viewru (bez přívlasku View)

Chráněné metody

- init ()

  Inicializační metoda pro kontroler.

8.15.1 Detailní popis

Abstraktní třída tvorbu kontroleru modulu.

Třída slouží jako základ pro tvorbu kontroleru modulu. Poskytuje přístup k vlastnostem modulu, práv hláškám, článku a kontaineru(přenos dat do pohledu)

Verze:

$Id: controller.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída kontroleru modulu

Definice je uvedena na řádku 13 v souboru controller.class.php.

8.15.2 Dokumentace konstruktoru a destruktoru

8.15.2.1 Controller::__construct (Action $ *action*, Routes $ *routes*, Rights $ *rights*) [final]

Konstruktor třídy vytvoří a naplní základní vlastnosti pro modul.

Parametry:

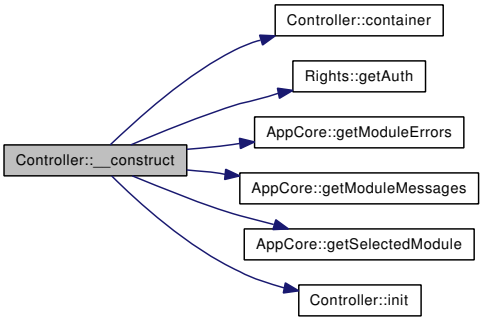
Module – objekt modulu

Definice je uvedena na řádku 86 v souboru controller.class.php.

Odkazuje se na container(), Rights::getAuth(), AppCore::getModuleErrors(), AppCore::getModuleMessages(), AppCore::getSelectedModule() a init().

```
86                                                                                  {
87
88        //TODO odstranit nepotřebné věci v paramtrech konstruktoru
89        $this->module = AppCore::getSelectedModule();
90        $this->action = $action;
91        $this->routes = $routes;
92        $this->auth = $rights->getAuth();
93        $this->rights = $rights;
94
95        if(AppCore::getModuleMessages() instanceof Messages){
96           $this->infomsg = AppCore::getModuleMessages();
97        }
98
99        if(AppCore::getModuleErrors() instanceof Messages){
100           $this->errmsg = AppCore::getModuleErrors();
101        }
102        $this->article = new Article();
103        $this->container = new Container();
104
105 //         Inicializace kontroleru modulu
106          $this->init();
107     }
```

Tato funkce volá...



8.15.2.2 Controller::__destruct ()

Vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 210 v souboru controller.class.php.

```
210                             {
211     }
```

8.15.3 Dokumentace k metodám

8.15.3.1 Controller::init () [protected]

Inicializační metoda pro kontroler.

je spuštěna vždy při vytvoření objektu kontroleru

Reimplementováno v BlogController.

Definice je uvedena na řádku 113 v souboru controller.class.php.

Používá se v __construct().

```
113 {}
```

8.15.3.2 Controller::getSysConfig () [final]

Metoda vrací objekt systémové konfigurace.

Návratová hodnota:

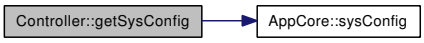
Config – objekt systémové konfigurace

Definice je uvedena na řádku 119 v souboru controller.class.php.

Odkazuje se na AppCore::sysConfig().

```
119                                     {
120         return AppCore::sysConfig();
121     }
```

Tato funkce volá...



8.15.3.3 Controller::getLink ($ *clear* = `false`) `[final]`

Metoda vrací odkaz na objekt pro práci s odkazy.

Návratová hodnota:

Links – objekt pro práci s odkazy

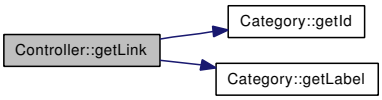
Definice je uvedena na řádku 127 v souboru controller.class.php.

Odkazuje se na Category::getId() a Category::getLabel().

Používá se v UsersController::addController(), SponsorsController::addController(), NewsController::addController(), GuestbookController::addController(), LoginController::changepasswdController(), checkControllRights(), checkReadableRights(), checkWriteableRights(), UsersController::editController(), SponsorsController::editController(), NewsController::editController(), GuestbookController::editController(), TextController::edittextController(), UsersController::mainController(), TextController::mainController(), SponsorsController::mainController(), NewsController::mainController(), LoginController::mainController(), ChangesController::mainController(), BlogController::mainController() a UsersController::showController().

```
127                                                     {
128         $link = new Links($clear);
129         $link->category(Category::getLabel(), Category::getId());
130         return $link;
131     }
```

Tato funkce volá...



8.15.3.4 Controller::getModule () `[final]`

Metody vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 137 v souboru controller.class.php.

Používá se v UsersController::addController(), SponsorsController::addController(), GuestbookController::addController(), UsersController::editController(), SponsorsController::editController(), GuestbookController::editController(), TextController::edittextController(), UsersController::mainController(), SponsorsController::mainController(), ReservationController::mainController(), NewsController::mainController(), ChangesController::mainController(), BlogController::mainController() a UsersController::showController().

```
137                                     {
138         return $this->module;
139     }
```

8.15.3.5 Controller::getAuth () [final]

Metody vrací objekt autorizace.

Návratová hodnota:

Auth – objekt autorizace

Definice je uvedena na řádku 145 v souboru controller.class.php.

Používá se v NewsController::addController(), LoginController::changepasswdController() a NewsController::mainController().

```
145                                     {
146         return $this->auth;
147     }
```

8.15.3.6 Controller::getAction () [final]

Metoda vrací objekt na akci.

Návratová hodnota:

ModuleAction – objekt akce

Definice je uvedena na řádku 153 v souboru controller.class.php.

Používá se v UsersController::mainController(), TextController::mainController(), SponsorsController::mainController(), NewsController::mainController(), LoginController::mainController(), BlogController::mainController() a UsersController::showController().

```
153                                     {
154         return $this->action;
155     }
```

8.15.3.7 Controller::getRights () [final]

Metoda vrací objekt s právy na modul.

Návratová hodnota:

Rights – objekt práv

Definice je uvedena na řádku 161 v souboru controller.class.php.

Používá se v UsersController::addController(), NewsController::addController(), LoginController::changepasswdController(), checkControllRights(), checkReadableRights(), checkWriteableRights(), TextController::edittextController(), UsersController::mainController(), TextController::mainController(), SponsorsController::mainController(), NewsController::mainController(), LoginController::mainController() a BlogController::mainController().

```
161                                     {
162         return $this->rights;
163     }
```

8.15.3.8 Controller::getArticle () [final]

Metoda vrací objekt s článkem.

Návratová hodnota:

Article – objekt článku

Definice je uvedena na řádku 169 v souboru controller.class.php.

Používá se v UsersController::editController(), SponsorsController::editController(), NewsController::editController(), GuestbookController::editController() a UsersController::showController().

```
169                                      {
170         return $this->article;
171     }
```

8.15.3.9 Controller::getRoutes () [final]

Metoda vrací objekt s cetsami - routes.

Návratová hodnota:

Routes – objekt Rout

Definice je uvedena na řádku 177 v souboru controller.class.php.

```
177                                     {
178         return $this->routes;
179     }
```

8.15.3.10 Controller::infoMsg () [final]

Metoda vrací objekt s informačními zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 185 v souboru controller.class.php.

Používá se v UsersController::addController(), SponsorsController::addController(), NewsController::addController(), GuestbookController::addController(), LoginController::changepasswdController(), UsersController::editController(), SponsorsController::editController(), NewsController::editController(), GuestbookController::editController(), TextController::edittextController() a UsersController::showController().

```
185                                     {
186         return $this->infomsg;
187     }
```

8.15.3.11 Controller::errMsg () [final]

Metoda vrací objekt s chybovými zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 193 v souboru controller.class.php.

Používá se v SponsorsController::addController(), GuestbookController::addController(), LoginController::changepasswdController(), checkControllRights(), checkReadableRights(), checkWritebleRights(), UsersController::editController(), SponsorsController::editController() a GuestbookController::editController().

```
193                                     {
194         return $this->errmsg;
195     }
```

8.15.3.12 Controller::container () [final]

Metoda vrací objekt kontaineru pro přenos dat mezi controlerem a viewrem.

Návratová hodnota:

Container – objekt kontaineru

Definice je uvedena na řádku 202 v souboru controller.class.php.

Používá se v __construct(), NewsController::addController(), NewsController::editController(), TextController::edittextController(), TextController::mainController(), NewsController::mainController(), LoginController::mainController(), GuestbookController::mainController() a BlogController::mainController().

```
202                                     {
203         return $this->container;
204     }
```

8.15.3.13 Controller::changeActionView ($ *newActionView*) [final]

Metoda změní výchozí actionViewer na zadaný.

Parametry:

string – název actionViewru

Definice je uvedena na řádku 222 v souboru controller.class.php.

```
222                                                         {
223         $this->actionViewer = $newActionView;
224     }
```

8.15.3.14 Controller::getActionView () [final]

Metoda vrací zvolený actionViewer nebo false pokud je nulový.

Návratová hodnota:

string – název nového actionViewru

Definice je uvedena na řádku 230 v souboru controller.class.php.

```
230                                     {
231         return $this->actionViewer;
232     }
```

8.15.3.15 Controller::checkReadableRights () [final]

Metoda kontroluje práva pro čtení modulu.

v opačném případě vyvolá přesměrování

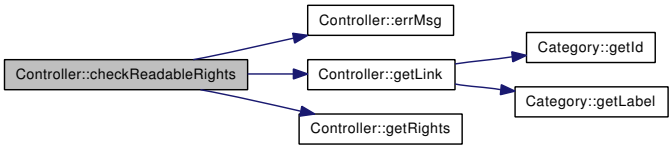
Definice je uvedena na řádku 237 v souboru controller.class.php.

Odkazuje se na errMsg(), getLink() a getRights().

Používá se v TextController::mainController(), SponsorsController::mainController(), NewsController::mainController(), GuestbookController::mainController() a BlogController::mainController().

```
237                                             {
238         if(!$this->getRights()->isReadable()){
239             $this->errMsg()->addMessage(_("Nemáte dostatčná práva pro přístup ke kategorii nebo jste byl(
240             $this->getLink(true)->reload();
241         }
242     }
```

Tato funkce volá...



8.15.3.16 Controller::checkWritebleRights () [final]

Metoda kontroluje práva pro zápis do modulu.

v opačném případě vyvolá přesměrování

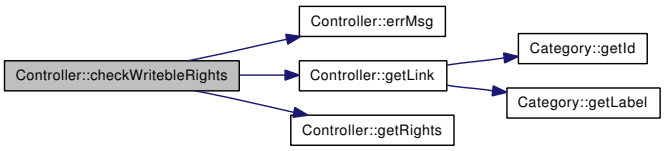
Definice je uvedena na řádku 246 v souboru controller.class.php.

Odkazuje se na errMsg(), getLink() a getRights().

Používá se v SponsorsController::addController(), NewsController::addController(), GuestbookController::addController(), BlogController::addsectionController(), LoginController::changepasswdController(), SponsorsController::editController(), NewsController::editController(), GuestbookController::editController() a TextController::edittextController().

```
246                                                 {
247         if(!$this->getRights()->isWritable()){
248             $this->errMsg()->addMessage(_("Nemáte dostatčná práva pro přístup ke kategorii nebo jste byl(
249             $this->getLink(true)->reload();
250         }
251     }
```

Tato funkce volá...



8.15.3.17 Controller::checkControllRights () [final]

Metoda kontroluje práva pro plný přístup k modulu.

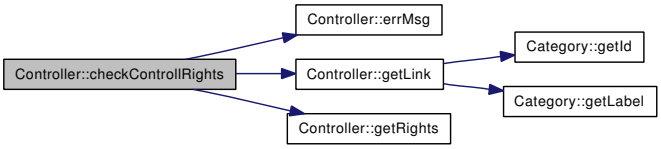
v opačném případě vyvolá přesměrování

Definice je uvedena na řádku 255 v souboru controller.class.php.

Odkazuje se na errMsg(), getLink() a getRights().

```
255                                                 {
256         if(!$this->getRights()->isControll()){
257             $this->errMsg()->addMessage(_("Nemáte dostatčná práva pro přístup ke kategorii nebo jste byl(
258             $this->getLink(true)->reload();
259         }
260     }
```

Tato funkce volá...



8.15.3.18 Controller::setView ($ *view*) [final]

metoda nastaví název použitého viewru (bez přívlasku View)

Parametry:

string – název viewru

Definice je uvedena na řádku 267 v souboru controller.class.php.

```
267                                         {
268         $this->actionViewer = $view;
269     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/controller.class.php

8.16 Dokumentace třídy CoreException

Třída pro obsluhu vyjímek v enginu (chybových hlášek).

Veřejné metody

- __construct ($message, $code=0)

  Konstruktor.

- **__toString** ()
- getException ()

  Metoda vygeneruje chbovou hlášku.

8.16.1 Detailní popis

Třída pro obsluhu vyjímek v enginu (chybových hlášek).

Třída rozšiřuje třídu Exception a přidává některé další prvky. Výstupní zpráva jsou zhromažďovány a použity pro výstup v enginu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: coreException.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro zachtáváni chyb enginu

Plánované úpravy

Není plně implementována a chce dodělat

Definice je uvedena na řádku 14 v souboru coreException.class.php.

8.16.2 Dokumentace k metodám

8.16.2.1 CoreException::getException ()

Metoda vygeneruje chbovou hlášku.

Návratová hodnota:

string – vygenerovaná hláška

Definice je uvedena na řádku 37 v souboru coreException.class.php.

```
37                                  {
38       $trace = $this->getTrace();
39       return __CLASS__ . ": [{$this->code}]: {$this->message} in file {$this->getFile()} on line {$this
40     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Exceptions/coreException.class.php

8.17 Dokumentace třídy CsvDataEplugin

Třída EPluginu práci s daty ve formátu cvs.

Diagram dědičnosti pro třídu CsvDataEplugin

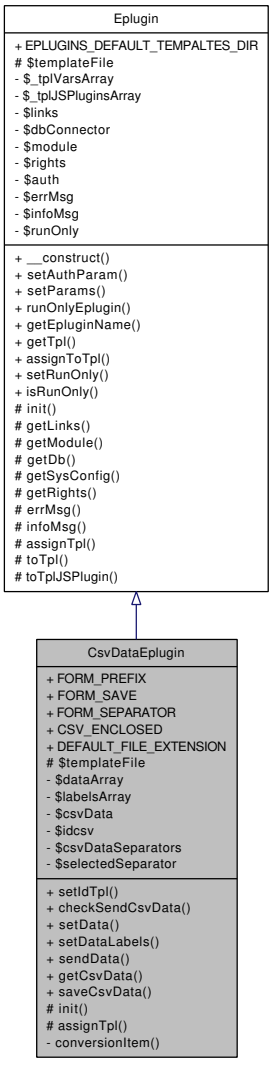
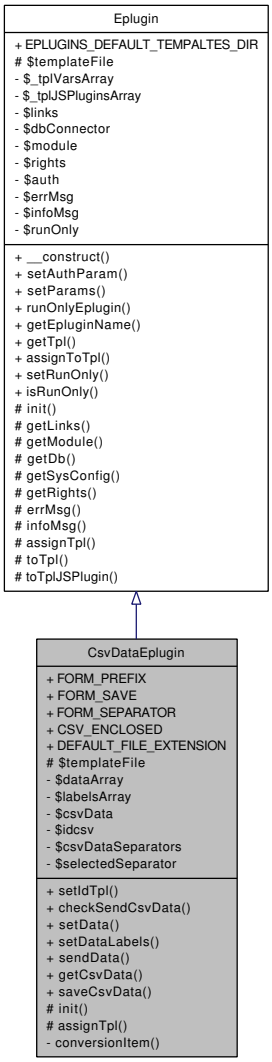


Diagram tříd pro CsvDataEplugin:



Veřejné metody

- setIdTpl ($id)

  Metoda nastaví id šablony pro výpis.

- checkSendCsvData ()

  Metoda kontroluje, jestli se odesílají data v csv.

- setData ($data)

  Metoda nastaví data, která se budou převádět na csv řetězec.

- setDataLabels ($labels)

  Metoda nastaví popisky k datům (nutný stejný počet popisku jako dat).

- sendData ($fileName= 'data')

Metoda odešle data v csv formátu jako soubor pomocí hlaviček.

- getCsvData ()

  Metoda vrací data csv jako řetězec, např, pro uložení do db.

- saveCsvData ($file)

  Metoda uloží csv data do zadaného souboru.

Veřejné atributy

- const **FORM_PREFIX** = 'csv_'
- const **FORM_SAVE** = 'save'
- const **FORM_SEPARATOR** = 'separator'
- const **CSV_ENCLOSED** = '"'
- const **DEFAULT_FILE_EXTENSION** = '.csv'

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda obstarává přiřazení proměných do šablony.

Chráněné atributy

- **$templateFile** = 'csvdata.tpl'

8.17.1 Detailní popis

Třída EPluginu práci s daty ve formátu cvs.

Třída umožňuje exportovat a pracovat s daty ve formátu cvs, který se dá bezproblémově naimportovat do jakéhokoliv programu (Excel, Kmail, Thunderbid, atd). Sám obsahuje šablonu s formulářem pro stažení dat.

Verze:

$Id: csvdata.class.php 3.0.0 beta1 2.11.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída Epluginu pro práci s daty ve formátu csv

Definice je uvedena na řádku 15 v souboru csvdata.class.php.

8.17.2 Dokumentace k metodám

8.17.2.1 CsvDataEplugin::setIdTpl ($ *id*)

Metoda nastaví id šablony pro výpis.

Parametry:

ineger – id šablony (jakékoliv)

Definice je uvedena na řádku 89 v souboru csvdata.class.php.

```
89                                  {
90        $this->idUserFiles = $id;
91    }
```

8.17.2.2 CsvDataEplugin::checkSendCsvData ()

Metoda kontroluje, jestli se odesílají data v csv.

Návratová hodnota:

boolean – true pokud jsou datat odesílána

Definice je uvedena na řádku 98 v souboru csvdata.class.php.

```
98                                      {
99        if(isset($_POST[self::FORM_PREFIX.self::FORM_SAVE])){
100           $this->selectedSeparator = $this->csvDataSeparators[$_POST[self::FORM_PREFIX.self::FORM_SEPAR
101           return true;
102       } else {
103           return false;
104       }
105   }
```

8.17.2.3 CsvDataEplugin::setData ($ *data*)

Metoda nastaví data, která se budou převádět na csv řetězec.

Parametry:

array – pole s daty (dvourozměrné pole)

Definice je uvedena na řádku 112 v souboru csvdata.class.php.

```
112                                   {
113       $this->dataArray = $data;
114   }
```

8.17.2.4 CsvDataEplugin::setDataLabels ($ *labels*)

Metoda nastaví popisky k datům (nutný stejný počet popisku jako dat).

Parametry:

array – pole s popisky

Definice je uvedena na řádku 120 v souboru csvdata.class.php.

```
120                                         {
121         $this->labelsArray = $labels;
122     }
```

8.17.2.5 CsvDataEplugin::sendData ($ *fileName* = 'data')

Metoda odešle data v csv formátu jako soubor pomocí hlaviček.

Plánované úpravy

ověřit a popípadě dodělat správné odesílání hlavičky

Definice je uvedena na řádku 128 v souboru csvdata.class.php.

Odkazuje se na getCsvData().

```
128                                             {
129         $csvData = $this->getCsvData();
130
131         header("Cache-Control: must-revalidate, post-check=0, pre-check=0");
132         header("Content-Length: " . strlen($csvData));
133         // Output to browser with appropriate mime type, you choose ;)
134         header("Content-type: text/x-csv");
135         //header("Content-type: text/csv");
136         //header("Content-type: application/csv");
137         header("Content-Disposition: attachment; filename=".$fileName.self::DEFAULT_FILE_EXTENSION);
138         echo $csvData;
139         exit;
140     }
```

Tato funkce volá...

8.17.2.6 CsvDataEplugin::getCsvData ()

Metoda vrací data csv jako řetězec, např, pro uložení do db.

Návratová hodnota:

string – řetězec s daty

Definice je uvedena na řádku 147 v souboru csvdata.class.php.

Používá se v sendData().

```
147                                 {
148         $csvString = null;
149
150         if(!empty($this->dataArray) AND is_array($this->dataArray)){
151            foreach ($this->dataArray as $row) {
152               $rowString = null;
153
154               foreach ($row as $cell) {
155                  $rowString .= $this->conversionItem($cell).$this->selectedSeparator;
156               }
157 //            Odstraní poslední oddělovač
158               $rowString = substr($rowString, 0, strlen($rowString)-1);
159               $csvString .= $rowString."\n";
160            }
161         }
162         return $csvString;
163     }
```

8.17.2.7 CsvDataEplugin::saveCsvData ($*file*)

Metoda uloží csv data do zadaného souboru.

Parametry:

string – název souboru

Plánované úpravy

implementovat ukládání do souboru na serveru (např. pro další stažení)

Definice je uvedena na řádku 190 v souboru csvdata.class.php.

```
190                                         {
191         ;
192     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/csvdata.class.php

8.18 Dokumentace třídy CtrlHelper

Abstraktní třída pro Controler Helper.

Diagram dědičnosti pro třídu CtrlHelper

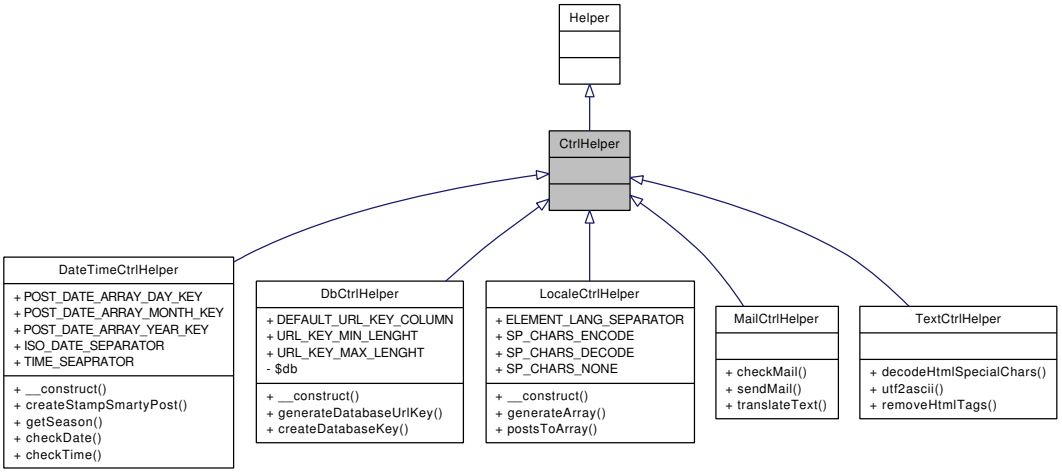
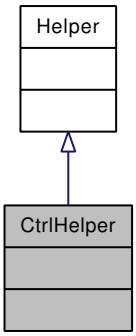


Diagram tříd pro CtrlHelper:



8.18.1 Detailní popis

Abstraktní třída pro Controler Helper.

Třída je základní prvek pro jakýkoliv kontrolhelper, který je použit v kontroleru.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: ctrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro vytvoření controler helperu

Definice je uvedena na řádku 13 v souboru ctrlhelper.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/ctrlhelper.class.php

8.19 Dokumentace třídy DateTimeCtrlHelper

Ttřída Controll Helperu pro práci s datumy a časy.

Diagram dědičnosti pro třídu DateTimeCtrlHelper

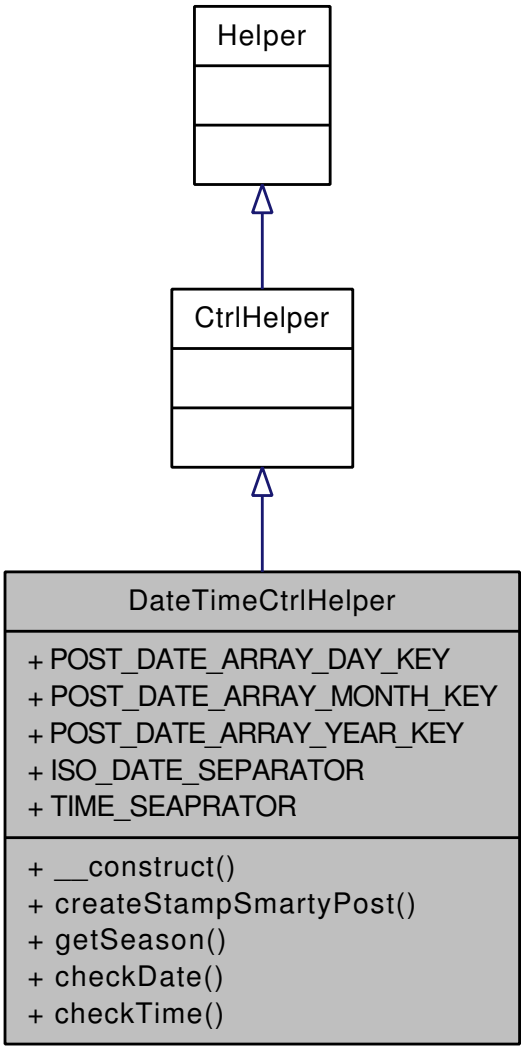
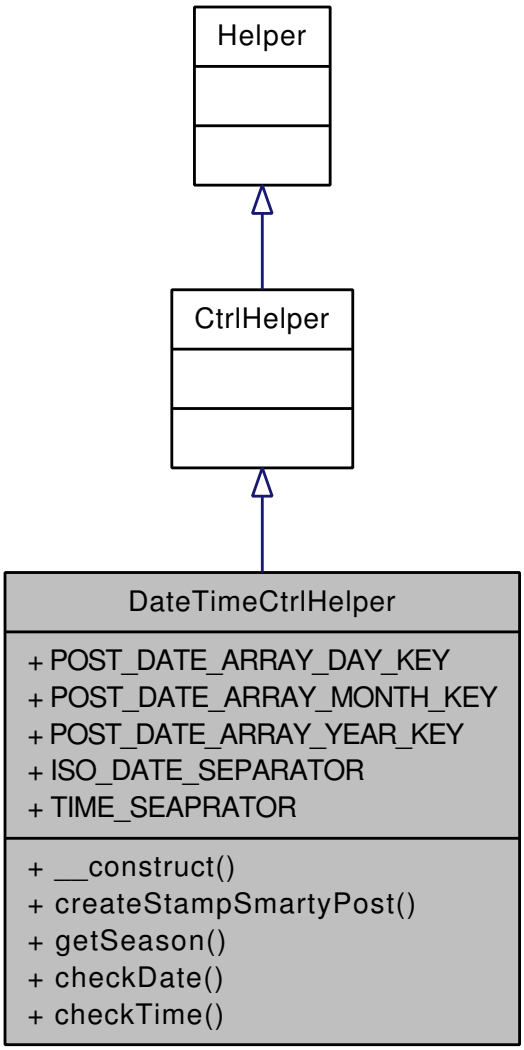


Diagram tříd pro DateTimeCtrlHelper:



Veřejné metody

- __construct ()

  Konstruktor třídy.

- createStampSmartyPost ($postName)

  Metoda vrací časové razítko ze zadaného pole, vygenerovaného pomocí SMARTY.

- getSeason ($timestamp)

  Metoda vrací dané období (jaro, léto , podzim, zima).

- checkDate ($date, $timestam=false)

  Metoda testuje zadané datum, pokud je v pořádku je vráceno v ISO podobe YYYY-MM-DD.

- checkTime ($time, $timestamp=false)

Metoda zkontroluje zadaný čas (hh:mm).

Veřejné atributy

- const **POST_DATE_ARRAY_DAY_KEY** = 'Date_Day'
- const **POST_DATE_ARRAY_MONTH_KEY** = 'Date_Month'
- const **POST_DATE_ARRAY_YEAR_KEY** = 'Date_Year'
- const **ISO_DATE_SEPARATOR** = '/'
- const **TIME_SEAPRATOR** = ':'

8.19.1 Detailní popis

Ttřída Controll Helperu pro práci s datumy a časy.

Třidá poskytuje metody pro práci s datumem a čase. Jejich kontrolu a prasování. Kontrolu ročního období atd.

Verze:

$Id: datetimectrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s datumi a časi v kontroleru - helper

Definice je uvedena na řádku 13 v souboru datetimectrlhelper.class.php.

8.19.2 Dokumentace k metodám

8.19.2.1 DateTimeCtrlHelper::createStampSmartyPost ($ *postName*)

Metoda vrací časové razítko ze zadaného pole, vygenerovaného pomocí SMARTY.

Parametry:

string – název $_POST pole s prvky datumu

Plánované úpravy

dodělat generování hodin:minut:sekund

Definice je uvedena na řádku 47 v souboru datetimectrlhelper.class.php.

```
47                                                      {
48         $day = $month = $year = $hour = $minute = $second = null;
49
50         if(isset($_POST[$postName][self::POST_DATE_ARRAY_DAY_KEY])){
51            $day = (int)$_POST[$postName][self::POST_DATE_ARRAY_DAY_KEY];
52         }
53
54         if(isset($_POST[$postName][self::POST_DATE_ARRAY_MONTH_KEY])){
```

```
55          $month = (int)$_POST[$postName][self::POST_DATE_ARRAY_MONTH_KEY];
56       }
57
58       if(isset($_POST[$postName][self::POST_DATE_ARRAY_YEAR_KEY])){
59          $year = (int)$_POST[$postName][self::POST_DATE_ARRAY_YEAR_KEY];
60       }
61       $timestamp = mktime($hour, $minute, $second, $month, $day, $year);
62       return $timestamp;
63    }
```

8.19.2.2 DateTimeCtrlHelper::getSeason ($ *timestamp*)

Metoda vrací dané období (jaro, léto , podzim, zima).

Parametry:

integer – timestamp

Návratová hodnota:

integer – číslo období (0-3)

Definice je uvedena na řádku 71 v souboru datetimectrlhelper.class.php.

```
71                                      {
72       $month = date("n",$timestamp);
73       $day = date("j",$timestamp);
74
75       switch($month){
76          case 1: case 2: return 3;
77          case 3: if ($day < 21) return 3; else return 0;
78          case 4: case 5: return 0;
79          case 6: if ($day < 21) return 0; else return 1;
80          case 7: case 8: return 1;
81          case 9: if ($day < 23) return 1; else return 2;
82          case 10: case 11: return 2;
83          case 12: if ($day < 21) return 2; else return 3;
84       }
85    }
```

8.19.2.3 DateTimeCtrlHelper::checkDate ($ *date*, $ *timestam* = `false`)

Metoda testuje zadané datum, pokud je v pořádku je vráceno v ISO podobe YYYY-MM-DD.

Parametry:

string – kontrolované datum

boolean(option) – true pokud má být vrácen timestamp

Návratová hodnota:

string – vrací datum v ISO podobě nebo false

Zastaralé

– je implementována ve validátoru TimeValidator

Definice je uvedena na řádku 94 v souboru datetimectrlhelper.class.php.

Odkazuje se na Locale::getLang().

```
94                                                      {
95       if (ereg("([0-9]{1,2}).([0-9]{1,2}).([0-9]{4})",$date, $regs) AND checkdate ($regs[2], $regs[1],
96          $date = $regs[2].self::ISO_DATE_SEPARATOR.$regs[1].self::ISO_DATE_SEPARATOR.$regs[3];
97          $time = mktime(null,null,null,$regs[2],$regs[1],$regs[3]);
98       } else if (ereg ("([0-9]{1,2})/([0-9]{1,2})/([0-9]{4})",$date, $regs) AND checkdate ($regs[1], $r
99          $date = $regs[1].self::ISO_DATE_SEPARATOR.$regs[2].self::ISO_DATE_SEPARATOR.$regs[3];
100          $time = mktime(null,null,null,$regs[1],$regs[2],$regs[3]);
101       } else {
102          $date = $time = false;
103       }
104
105       if($timestam){
106          return $time;
107       } else {
108          return $date;
109       }
110    }
```

Tato funkce volá...

8.19.2.4 DateTimeCtrlHelper::checkTime ($ *time*, $ *timestamp* = `false`)

Metoda zkontroluje zadaný čas (hh:mm).

Parametry:

string – kontrolovaný čas

boolean(option) – true pokud má být vrácen timestamp

Návratová hodnota:

string – vrací čas v podobě HH:MM nebo false

Zastaralé

– je implementována ve validátoru TimeValidator

Definice je uvedena na řádku 120 v souboru datetimectrlhelper.class.php.

```
120                                                     {
121       $nums = explode(self::TIME_SEAPRATOR, $time, 2);
122
123       $allOk = true;
124 //    Jestli se jedná o hodinu
125       if(!isset($nums[0]) OR $nums[0] < 0 OR $nums[0] > 23){
126           $allOk = false;
127       }
128
129 //    Jestli se jedná o minuty
130       if(!isset($nums[1]) OR $nums[1] < 0 OR $nums[1] > 59){
131           $allOk = false;
```

```
132       }
133
134       if($allOk){
135          if(!$timestamp){
136             return $nums[0].self::TIME_SEAPRATOR.$nums[1];
137          } else {
138             return mktime($nums[0], $nums[1]);
139          }
140       } else {
141          return false;
142       }
143    }
```

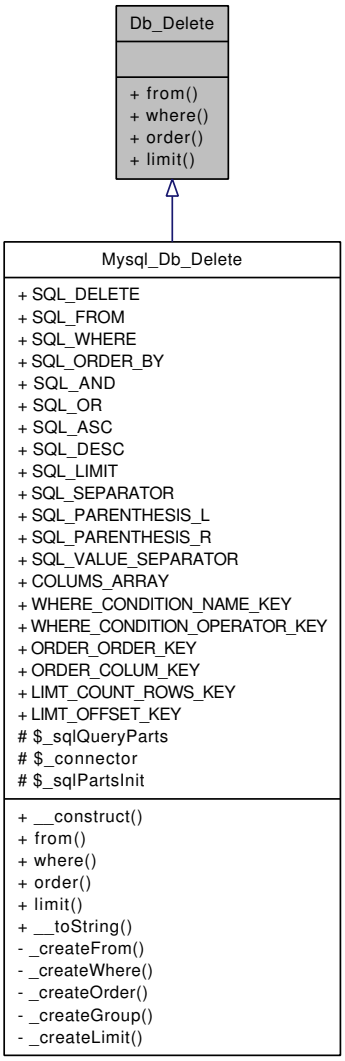
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/datetimectrlhelper.class.php

8.20 Dokumentace třídy Db_Delete

Abstraktní třída pro mazání záznamů z db.

Diagram dědičnosti pro třídu Db_Delete



Veřejné metody

- from ($table, $columsArray=null)

  Metoda nastavuje z které tabulky se bude mazat klauzule FROM.

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- order ($colum, $order=self::SQL_ASC)

  Metoda přiřadí řazení sloupcu v SQL dotazu.

- limit ($rowCount, $offset)

  Metoda přidá do SQL dotazu klauzuli LIMIT.

8.20.1 Detailní popis

Abstraktní třída pro mazání záznamů z db.

Třída zobrazuje prvky třídy, které musí být použity v jednotlivých implementacích databázových konektorů.

Verze:

$Id: delete.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro mazání záznamů z db

Definice je uvedena na řádku 13 v souboru delete.class.php.

8.20.2 Dokumentace k metodám

8.20.2.1 Db_Delete::from ($ *table*, $ *columsArray* = `null`) `[abstract]`

Metoda nastavuje z které tabulky se bude mazat klauzule FROM.

Parametry:

string/array – tabulka ze které se bude vybírat, u pole index označuje alias tabulky

string/array – které sloupce se mají mazat

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementováno v Mysql_Db_Delete.

8.20.2.2 Db_Delete::where ($ *condition*, $ *operator* = `self::SQL_AND`) `[abstract]`

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka

string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementováno v Mysql_Db_Delete.

8.20.2.3 Db_Delete::order ($ *colum*, $ *order* = `self::SQL_ASC`) `[abstract]`

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementováno v Mysql_Db_Delete.

8.20.2.4 Db_Delete::limit ($ *rowCount*, $ *offset*) `[abstract]`

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementováno v Mysql_Db_Delete.

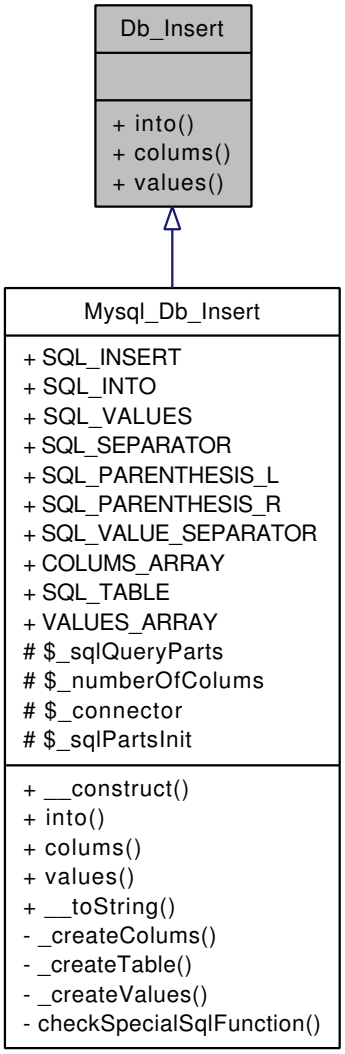
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/delete.class.php

8.21 Dokumentace třídy Db_Insert

Abstraktní třída pro vkládání záznamů do db.

Diagram dědičnosti pro třídu Db_Insert



Veřejné metody

- into ($table)

  Metoda nastavuje do které tabulky se bude zapisovat klauzule INTO.

- colums ($colums)

  Metody vatváří sloupce, které se budou zapisovat.

- values ($value)

  Metoda přiřadí hodnoty sloupscům.

8.21.1 Detailní popis

Abstraktní třída pro vkládání záznamů do db.

Třída zobrazuje prvky třídy, které musí být použity v jednotlivých implementacích databázových konektorů.

Verze:

$Id: insert.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro vkládání záznamů do db

Definice je uvedena na řádku 13 v souboru insert.class.php.

8.21.2 Dokumentace k metodám

8.21.2.1 Db_Insert::into ($ *table*) [abstract]

Metoda nastavuje do které tabulky se bude zapisovat klauzule INTO.

Parametry:

string – tabulka do které se bude zapisovat

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementováno v Mysql_Db_Insert.

8.21.2.2 Db_Insert::colums ($ *colums*) [abstract]

Metody vatváří sloupce, které se budou zapisovat.

Parametry:

mixed – sloupce (string nebo array)

string – sloupce (neomezený počet parametrů)

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementováno v Mysql_Db_Insert.

8.21.2.3 Db_Insert::values ($ *value*) [abstract]

Metoda přiřadí hodnoty sloupscům.

Parametry:

string – hodnota sloupce (proměnný počet parametrů)

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementováno v Mysql_Db_Insert.

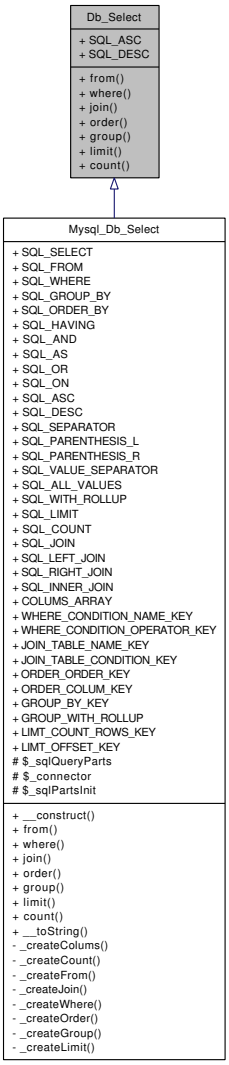
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/insert.class.php

8.22 Dokumentace třídy Db_Select

Abstraktní třída pro výběr záznamů z db.

Diagram dědičnosti pro třídu Db_Select



Veřejné metody

- from ($tableArray, $columsArray="∗")

  Metoda nastavuje z které tabulky se bude načítat klauzule FROM.

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- join ($tableArray, $condition, $joinType=null, $columsArray="∗")

  Metody vytvoří část pro klauzuli JOIN.

- order ($colum, $order=self::SQL_ASC)
  Metoda přiřadí řazení sloupcu v SQL dotazu.
- group ($colum, $withRollup=false)
  Metoda přiřadí slouření sloupců v SQL dotazu pomocí klauzule GROUP BY.
- limit ($rowCount, $offset)
  Metoda přidá do SQL dotazu klauzuli LIMIT.
- count ($alias=null, $colum=self::SQL_ALL_VALUES)
  Metoda přidává do dotazu sloupce s počem záznamů.

Veřejné atributy

- const SQL_ASC = 'ASC'
  Vybrané konstanty pro SQL dotazy.
- const **SQL_DESC** = 'DESC'

8.22.1 Detailní popis

Abstraktní třída pro výběr záznamů z db.

Třída zobrazuje prvky třídy, které musí být použity v jednotlivých implementacích databázových konektorů.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: select.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro výběr záznamů z db

Definice je uvedena na řádku 13 v souboru select.class.php.

8.22.2 Dokumentace k metodám

8.22.2.1 Db_Select::from ($ *tableArray*, $ *columsArray* = "*") [abstract]

Metoda nastavuje z které tabulky se bude načítat klauzule FROM.

Parametry:

string/array – tabulka ze které se bude vybírat, u pole index označuje alias tabulky
string/array – které sloupce mají být vabrány (option)

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.2 Db_Select::where ($ *condition*, $ *operator* = `self::SQL_AND`) `[abstract]`

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka

string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.3 Db_Select::join ($ *tableArray*, $ *condition*, $ *joinType* = `null`, $ *columsArray* = `"*"`) `[abstract]`

Metody vytvoří část pro klauzuli JOIN.

Parametry:

string/array – název tabulky (alias je generován z prvních dvou písmen) nebo pole kde index je alias tabulky

string – podmínka v klauzuli ON, je třeba zadat i s aliasy

string – typ JOIN operace hodnoty jsou: JOIN, LEFT, RIGHT, INNER

string/array – název sloupců, které se mají vypsatm, výchozí jsou všechny ("*"). U pole označuje index alias prvku. Pokud je zadáno null, nebude načten žádný sloupec

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.4 Db_Select::order ($ *colum*, $ *order* = `self::SQL_ASC`) `[abstract]`

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.5 Db_Select::group ($ *colum*, $ *withRollup* = `false`) `[abstract]`

Metoda přiřadí slouření sloupců v SQL dotazu pomocí klauzule GROUP BY.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) WITH ROLLUP false(default)/true

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.6 Db_Select::limit ($ *rowCount*, $ *offset*) `[abstract]`

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

8.22.2.7 Db_Select::count ($ *alias* = `null`, $ *colum* = `self::SQL_ALL_VALUES`) `[abstract]`

Metoda přidává do dotazu sloupce s počem záznamů.

Parametry:

string – alias pod kterým má být vrácena hodnota

string – které sloupce se mají vybrat (option) default: všechny

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementováno v Mysql_Db_Select.

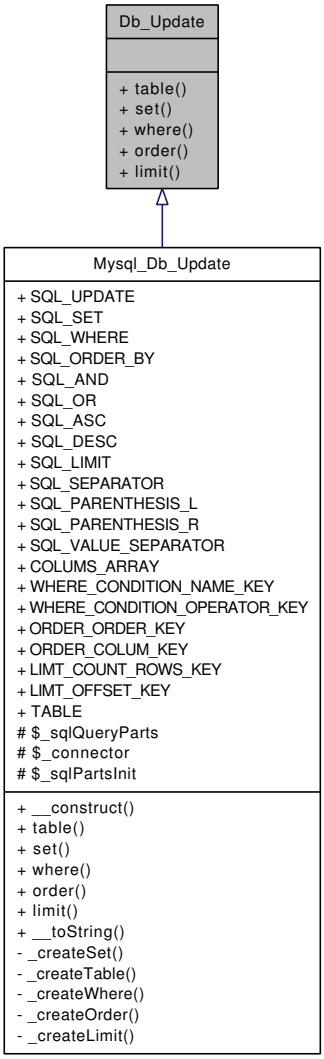
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/select.class.php

8.23 Dokumentace třídy Db_Update

Abstraktní třída pro aktualizaci záznamů v db.

Diagram dědičnosti pro třídu Db_Update



Veřejné metody

- table ($table)

  Metoda nastavuje v které tabulce se bude upravovat klauzule UPDATE table.

- set ($values)

  Metoda nastavuje, které hodnoty se upraví (název sloupce) => (hodnota).

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- order ($colum, $order=self::SQL_ASC)
  Metoda přiřadí řazení sloupcu v SQL dotazu.
- limit ($rowCount, $offset)
  Metoda přidá do SQL dotazu klauzuli LIMIT.

8.23.1 Detailní popis

Abstraktní třída pro aktualizaci záznamů v db.

Třída zobrazuje prvky třídy, které musí být použity v jednotlivých implementacích databázových konektorů.

Verze:

$Id: update.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro aktualizaci záznamů v db

Definice je uvedena na řádku 13 v souboru update.class.php.

8.23.2 Dokumentace k metodám

8.23.2.1 Db_Update::table ($ *table*) [abstract]

Metoda nastavuje v které tabulce se bude upravovat klauzule UPDATE table.

Parametry:

string – tabulka která se bude upravovat

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementováno v Mysql_Db_Update.

8.23.2.2 Db_Update::set ($ *values*) [abstract]

Metoda nastavuje, které hodnoty se upraví (název sloupce) => (hodnota).

Parametry:

array – pole s hodnotamik

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementováno v Mysql_Db_Update.

8.23.2.3 Db_Update::where ($ *condition*, $ *operator* = `self::SQL_AND`) `[abstract]`

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka

string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementováno v Mysql_Db_Update.

8.23.2.4 Db_Update::order ($ *colum*, $ *order* = `self::SQL_ASC`) `[abstract]`

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementováno v Mysql_Db_Update.

8.23.2.5 Db_Update::limit ($ *rowCount*, $ *offset*) `[abstract]`

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementováno v Mysql_Db_Update.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/update.class.php

8.24 Dokumentace třídy DbCtrlHelper

Ttřída Controll Helperu pro zjednodušení práce s DB.

Diagram dědičnosti pro třídu DbCtrlHelper

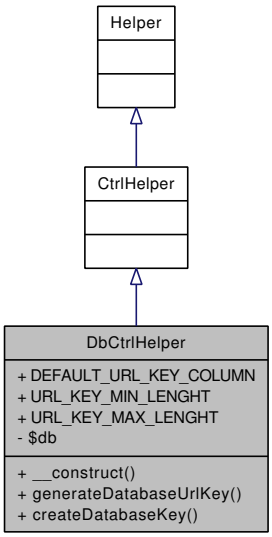
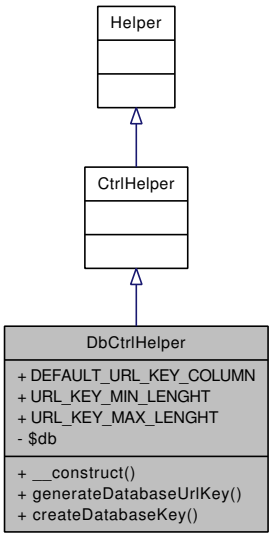


Diagram tříd pro DbCtrlHelper:



Veřejné metody

- __construct ()

  Konstruktor třídy.

- generateDatabaseUrlKey ($textForGenerate, $dbTable, $urlKeyColum=self::DEFAULT_URL_-KEY_COLUMN, $minLenght=self::URL_KEY_MIN_LENGHT, $maxLenght=self::URL_KEY_-MAX_LENGHT)

  Metoda vytvoří unikátní url klíč v zadané tabulce.

- createDatabaseKey ($text, $keysArray=null, $minLenght=self::URL_KEY_MIN_LENGHT, $maxLenght=self::URL_KEY_MAX_LENGHT)

  Funkce vytvoří klíč o velikosti 50 znaků, který je určen pro uložení do db.

Veřejné atributy

- const **DEFAULT_URL_KEY_COLUMN** = 'urlkey'
- const **URL_KEY_MIN_LENGHT** = 9
- const **URL_KEY_MAX_LENGHT** = 50

8.24.1 Detailní popis

Ttřída Controll Helperu pro zjednodušení práce s DB.

Třída poskytuje metody pro jednodušší práci s prvky pro db. Například genervání unikátních db klíčů.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: dbctrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s prvky db v kontroleru(modelech) - helper

Definice je uvedena na řádku 13 v souboru dbctrlhelper.class.php.

8.24.2 Dokumentace k metodám

8.24.2.1 DbCtrlHelper::generateDatabaseUrlKey ($ *textForGenerate*, $ *dbTable*, $ *urlKeyColum* = `self::DEFAULT_URL_KEY_COLUMN`, $ *minLenght* = `self::URL_KEY_MIN_LENGHT`, $ *maxLenght* = `self::URL_KEY_MAX_LENGHT`)

Metoda vytvoří unikátní url klíč v zadané tabulce.

Parametry:

string – text ze kterého se má klíč generovat

string – název tabulky

string – název sloupce s urlklíči

Návratová hodnota:

string – vygenerovaný url klíč

Definice je uvedena na řádku 54 v souboru dbctrlhelper.class.php.

Odkazuje se na createDatabaseKey().

```
54
55        $sqlSelecturlKey = $this->db->select()->from($dbTable, $urlKeyColum);
56
57        $urlkeysArray = $this->db->fetchAssoc($sqlSelecturlKey);
58
59        $newUrlKeys = array();
60        foreach ($urlkeysArray as $urlKey) {
61           array_push($newUrlKeys, $urlKey[$urlKeyColum]);
62        }
63
64
65        $urlKey = $this->createDatabaseKey($textForGenerate, $newUrlKeys, $minLenght, $maxLenght);
66
67        return $urlKey;
68    }
```

Tato funkce volá...

8.24.2.2 DbCtrlHelper::createDatabaseKey ($ *text*, $ *keysArray* = `null`, $ *minLenght* = `self::URL_KEY_MIN_LENGHT`, $ *maxLenght* = `self::URL_KEY_MAX_LENGHT`)

Funkce vytvoří klíč o velikosti 50 znaků, který je určen pro uložení do db.

Parametry:

string – text ze kterého se má klíč vytvořit

array – pole již existujících klíčů

integer – minimální počet znaků

integer – maximální počet znaků

Návratová hodnota:

string – vygenerovaný klíč pro db

Definice je uvedena na řádku 80 v souboru dbctrlhelper.class.php.

Používá se v generateDatabaseUrlKey().

```
81    {
82        // odstranění tagů
83        $text=ereg_replace("<[^>]+>", "", $text);
84
85        // pevod na ascii
86        $textHelper = new TextCtrlHelper();
87        $newKey = $textHelper->utf2ascii($text);
88        unset($textHelper);
89        $newKey = substr($newKey, 0, $maxLenght);
90
91        // Porpvnání klíču již uloženyých a vytvoření unikátního
92        if($keysArray != null){
```

```
93          $step = 1;
94          $newUniqueKey = $newKey;
95          $uniqueKey = false;
96
97          while ($uniqueKey != true) {
98             if(!in_array($newUniqueKey, $keysArray)){
99                $uniqueKey = true;
100            } else {
101               $newUniqueKey=$newKey."-".$step++;
102            }
103         }
104         $newKey=$newUniqueKey;
105      }
106
107      if(strlen($newKey) < $minLenght){
108         $newKey = str_pad($newKey, $minLenght, ".", STR_PAD_BOTH);
109      }
110
111      return $newKey;
112   }
```

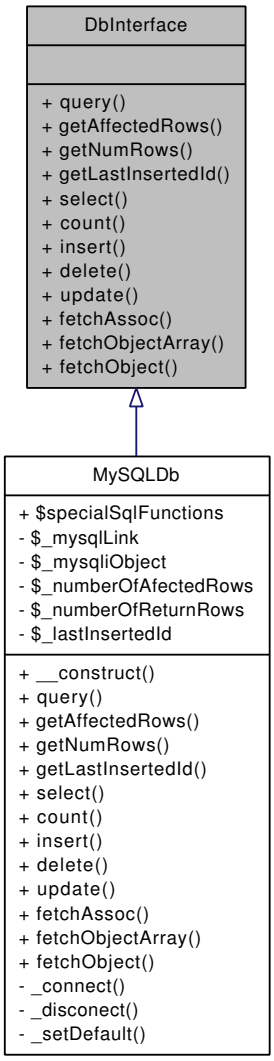
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/dbctrlhelper.class.php

8.25 Dokumentace rozhraní DbInterface

Třída obsluhuje interface pro db konektory.

Diagram dědičnosti pro třídu DbInterface



Veřejné metody

- query ($sqlQuery)

  Metoda provede daný sql dotaz na databázi.

- getAffectedRows ()

  Metoda vrací počet ovlivněných záznamů příkazy INSERT, UPDATE, DELETE, REPLACE.

- getNumRows ()

  Metoda vrací počet vrácených záznamů příkazy SELECT.

- getLastInsertedId ()

  Metoda vrací id posledního vloženého záznamu pomocí INSERT.

- select ()

  Metoda pro výběr prvků z db.

- count ($table, $condition=null)

  Metoda vrací počet záznamů v zadané tabulce.

- insert ()

  Metoda pro vložení prvků do db.

- delete ()

  Metoda pro smazání prvků z db.

- update ()

  Metoda pro úpravu prvků z db.

- fetchAssoc ($sqlQuery, $oneArray=false)

  Metoda provede sql dotaz a výstup doplní do asociativního pole.

- fetchObjectArray ($sqlQuery)

  Metoda provede sql dotaz a výstup přiřadí do pole objektů mysqli.

- fetchObject ($sqlQuery)

  Metoda vrací řádek z db jako objekt.

8.25.1 Detailní popis

Třída obsluhuje interface pro db konektory.

Třída zobrazuje prvky základní třídy dbkonektoru, které musí být použity v jednotlivých implementacích databázových konektorů.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: db.interface.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída s interfacem pro vytváření db konekorů

Definice je uvedena na řádku 13 v souboru db.interface.php.

8.25.2 Dokumentace k metodám

8.25.2.1 DbInterface::query ($ *sqlQuery*)

Metoda provede daný sql dotaz na databázi.

Parametry:

string – sql dotaz

Implementováno v MySQLDb.

8.25.2.2 DbInterface::getAffectedRows ()

Metoda vrací počet ovlivněných záznamů příkazy INSERT, UPDATE, DELETE, REPLACE.

Návratová hodnota:

integer – počet ovlivněných záznamu

Implementováno v MySQLDb.

8.25.2.3 DbInterface::getNumRows ()

Metoda vrací počet vrácených záznamů příkazy SELECT.

Návratová hodnota:

integer – počet vrácených záznamu

Implementováno v MySQLDb.

8.25.2.4 DbInterface::getLastInsertedId ()

Metoda vrací id posledníího vloženého záznamu pomocí INSERT.

Návratová hodnota:

integer – id posledního záznamu

Implementováno v MySQLDb.

8.25.2.5 DbInterface::select ()

Metoda pro výběr prvků z db.

Návratová hodnota:

Db_Select – objekt pro přístup k db;

Implementováno v MySQLDb.

8.25.2.6 DbInterface::count ($ *table*, $ *condition* = `null`)

Metoda vrací počet záznamů v zadané tabulce.

Parametry:

string – název tabulky
string – podmínka

Návratová hodnota:

integer – počet záznamů v tabulce

Implementováno v MySQLDb.

8.25.2.7 DbInterface::insert ()

Metoda pro vložení prvků do db.

Návratová hodnota:

Db_Insert – objekt pro přístup k db;

Implementováno v MySQLDb.

8.25.2.8 DbInterface::delete ()

Metoda pro smazání prvků z db.

Návratová hodnota:

Db_Delete – objekt pro přístup k db;

Implementováno v MySQLDb.

8.25.2.9 DbInterface::update ()

Metoda pro úpravu prvků z db.

Návratová hodnota:

Db_Update – objekt pro přístup k db;

Implementováno v MySQLDb.

8.25.2.10 DbInterface::fetchAssoc ($ *sqlQuery*, $ *oneArray* = `false`)

Metoda provede sql dotaz a výstup doplní do asociativního pole.

Parametry:

string – SQL dotaz
boolean – jestli se má vrátit pouze pole s prvky, nebo pole s poly prvků

Návratová hodnota:

array/boolean – asociativní pole s výsledky sql dotazu nebo false při prázdném výsledku

Implementováno v MySQLDb.

8.25.2.11 DbInterface::fetchObjectArray ($ *sqlQuery*)

Metoda provede sql dotaz a výstup přiřadí do pole objektů mysqli.

Parametry:

string – SQL dotaz

Návratová hodnota:

array – pole objektů MySQLi_Result

Implementováno v MySQLDb.

8.25.2.12 DbInterface::fetchObject ($ *sqlQuery*)

Metoda vrací řádek z db jako objekt.

Parametry:

string – sql dotaz

Návratová hodnota:

MySQLi_Result – objekt s prvky z db

Implementováno v MySQLDb.

Dokumentace pro toto rozhraní byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/db.interface.php

8.26 Dokumentace třídy DbModel

Abstraktní třída pro Db Model.

Diagram dědičnosti pro třídu DbModel

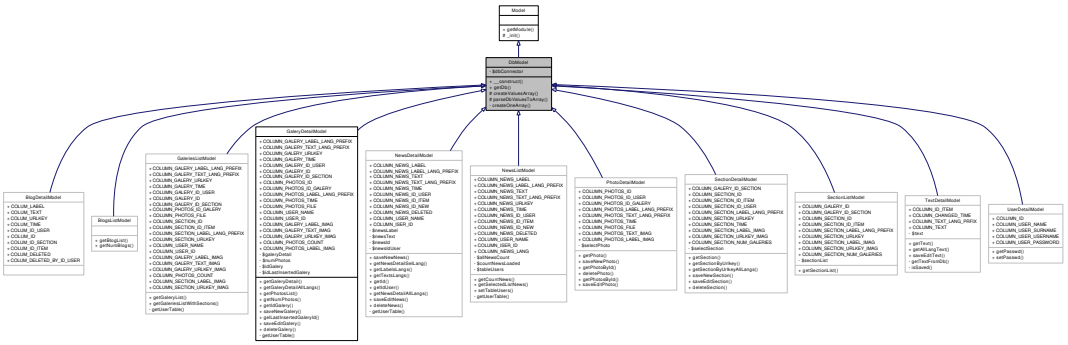
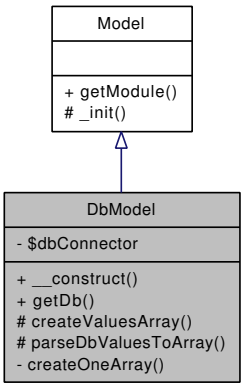


Diagram tříd pro DbModel:



Veřejné metody

- __construct ()

  Konstruktor třídy.

- getDb ()

  Metoda vrací objekt ke konektoru databáze.

Chráněné metody

- createValuesArray ($name, $val)

  Metoda vytvoří pole s hodnotami a klíči vhodnými pro uložení do db.

- parseDbValuesToArray ($values, $prefixArray, $separator= '_')

  Metoda rozparsuje pole databáze na strukturované pole.

8.26.1 Detailní popis

Abstraktní třída pro Db Model.

Tříta pro vytvoření modelu, přistupujícího k databázi. Umožňuje základní práce s databází.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: dbmodel.class.php 3.0.55 26.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro vytvoření modelu pro práci s databází

Definice je uvedena na řádku 15 v souboru dbmodel.class.php.

8.26.2 Dokumentace k metodám

8.26.2.1 DbModel::getDb () [final]

Metoda vrací objekt ke konektoru databáze.

Návratová hodnota:

Dbinterface

Definice je uvedena na řádku 40 v souboru dbmodel.class.php.

Odkazuje se na AppCore::getDbConnector().

Používá se v GaleryDetailModel::deleteGalery(), GaleryDetailModel::getGaleryDetail(), GaleryDetailModel::getGaleryDetailAllLangs(), GaleryDetailModel::getNumPhotos(), GaleryDetailModel::getPhotosList(), GaleryDetailModel::saveEditGalery() a GaleryDetailModel::saveNewGalery().

```
40                                  {
41         return AppCore::getDbConnector();
42     }
```

Tato funkce volá...

8.26.2.2 DbModel::createValuesArray ($ *name*, $ *val*) [protected]

Metoda vytvoří pole s hodnotami a klíči vhodnými pro uložení do db.

Parametry:

string $name – název pro sloupec

mixed $val – hodnota sloupce

Definice je uvedena na řádku 53 v souboru dbmodel.class.php.

```
53                                                      {
54       $separator = '_';
55       $returnArray = array();
56       if(func_num_args()%2 == 0){
57          $argv = func_get_args();
58
59 //         Skáčeme po dvou, protože první je sloupec a adruhý je hodnota
60          for ($step = 0 ; $step < func_num_args() ; $step=$step+2) {
61             if(is_array($argv[$step+1])){
62                $returnArray = array_merge($returnArray, $this->createOneArray(
63                      $argv[$step+1],$argv[$step]));
64             } else {
65                $returnArray[$argv[$step]] = $argv[$step+1];
66             }
67          }
68       } else {
69          new CoreException(_('Nebyl předán potřebný počet argumentů'));
70       }
71
72       return $returnArray;
73    }
```

8.26.2.3 DbModel::parseDbValuesToArray ($ *values*, $ *prefixArray*, $ *separator* = '_') [protected]

Metoda rozparsuje pole databáze na strukturované pole.

Parametry:

array $values – pole s hodnotami

array $values – pole s prefixy, které se mají parsovat

Definice je uvedena na řádku 110 v souboru dbmodel.class.php.

```
110                                                                                    {
111       if(!is_array($prefixArray)){
112          $prefixArray = array($prefixArray);
113       }
114
115       $returnArr = array();
116       foreach ($values as $key => $var) {
117          $matches = array();
118          if(eregi('^([a-zA-Z0-9]*)'.$separator.'([a-zA-Z0-9]*)$', $key, $matches)
119             AND in_array($matches[1], $prefixArray)){
120                $returnArr[$matches[1]][$matches[2]]=$var;
121          } else {
122             $returnArr[$key]=$var;
123          }
124       };
125       return $returnArr;
126    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/models/dbmodel.class.php

8.27 Dokumentace třídy Dir

Třída pro práci s adresáři.

Veřejné metody

- __construct ($dirName=null)

  Konstruktor vytvoří objekt adresáře.

- checkDir ($directory)

  Metoda otestuje existenci adresáře, a pokud neexistuje pokusí se jej vytvořit.

- createDirs ($path)

  Funkce vytvoři zadaný adresář i podadresáře, pokud neexistují.

- rmDir ($filepath)

  Metoda maže rekjurzivně zadaný adresář/soubor.

- checkDirPath ($path)

  Metoda kontroluje jestli je cesta zadána správně, jinak vrací opravenou cestu.

8.27.1 Detailní popis

Třída pro práci s adresáři.

umožnuje jejich vytváření, mazání, přejmenovávání a kontrolu existence

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída pro práci s adresáři

Plánované úpravy

– refaktoring, tak aby to byl objekt plnohodnotný

Definice je uvedena na řádku 13 v souboru dir.class.php.

8.27.2 Dokumentace konstruktoru a destruktoru

8.27.2.1 Dir::__construct ($ *dirName* = `null`)

Konstruktor vytvoří objekt adresáře.

Parametry:

string $dirName – (option) název adresáře

Definice je uvedena na řádku 30 v souboru dir.class.php.

```
30                                                  {
31       $this->dir = $dirName;
32
33 //      if(AppCore::getModuleErrors() instanceof Messages){
34 //         $this->errmsg = AppCore::getModuleErrors();
35 //      }
36    }
```

8.27.3 Dokumentace k metodám

8.27.3.1 Dir::checkDir ($ *directory*)

Metoda otestuje existenci adresáře, a pokud neexistuje pokusí se jej vytvořit.

Parametry:

string – název adresáře

Plánované úpravy

dodělat přidávání lomítek před adresář

Definice je uvedena na řádku 44 v souboru dir.class.php.

Odkazuje se na createDirs().

```
44                                          {
45       //doplnění posledního lomítka za dest adresář
46       if($directory[strlen($directory)-1] != "/"){
47          $directory .= "/";
48       }
49
50       if(!file_exists($directory) OR !is_dir($directory)){
51          return $this->createDirs($directory);
52       }
53       return true;
54    }
```

Tato funkce volá...

8.27.3.2 Dir::createDirs ($ *path*)

Funkce vytvoři zadaný adresář i podadresáře, pokud neexistují.

Parametry:

string – adresář

Definice je uvedena na řádku 61 v souboru dir.class.php.

Používá se v checkDir().

```
62    {
63       if (is_dir($path)){
64          return true;
65       }
66
67       if(mkdir($path, 0777, true)){
68 //          $return = mkdir($path, 0777, true);
69          if(chmod($path, 0777)){
70             return true;
71          } else {
72             new CoreException(_('Adresáři ').$path._(' se nepodařilo přidělit správná oprávnění'), 2);
73          }
74       } else {
75          new CoreException(_('Adresář ').$path._(' se nepodařilo vytvořit, zkontrolujte prosím oprávněn
76       }
77    }
```

8.27.3.3 Dir::rmDir ($ *filepath*)

Metoda maže rekjurzivně zadaný adresář/soubor.

Parametry:

string – adresář/soubor, který se má smazat

Návratová hodnota:

boolena – true pokud se podařilo adresář/soubor smazat

Definice je uvedena na řádku 85 v souboru dir.class.php.

```
86    {
87       if (is_dir($filepath) && !is_link($filepath)){
88          $dir = opendir($filepath);
89          if ($dir){
90             while (($sf = readdir($dir)) !== false){
91                if ($sf == '.' || $sf == '..'){
92                   continue;
93                }
94                if (!$this->rmDir($filepath.'/'.$sf)){
95                   new CoreException($filepath.'/'.$sf._(' soubor nemohl být smazán.'));
96                }
97             }
98             closedir($dir);
99          }
100          return rmdir($filepath);
101       }
102       if(file_exists($filepath)){
103          return unlink($filepath);
104       }
105    }
```

8.27.3.4 Dir::checkDirPath ($ *path*)

Metoda kontroluje jestli je cesta zadána správně, jinak vrací opravenou cestu.

Parametry:

string – cesta

Návratová hodnota:

string – opravená cesta

Definice je uvedena na řádku 114 v souboru dir.class.php.

```
114                                      {
115       if(($path[strlen($path)-1] != '/') AND ($path[strlen($path)-1] != '\\')){
116          $path.=DIRECTORY_SEPARATOR;
117       }
118       return $path;
119    }
```

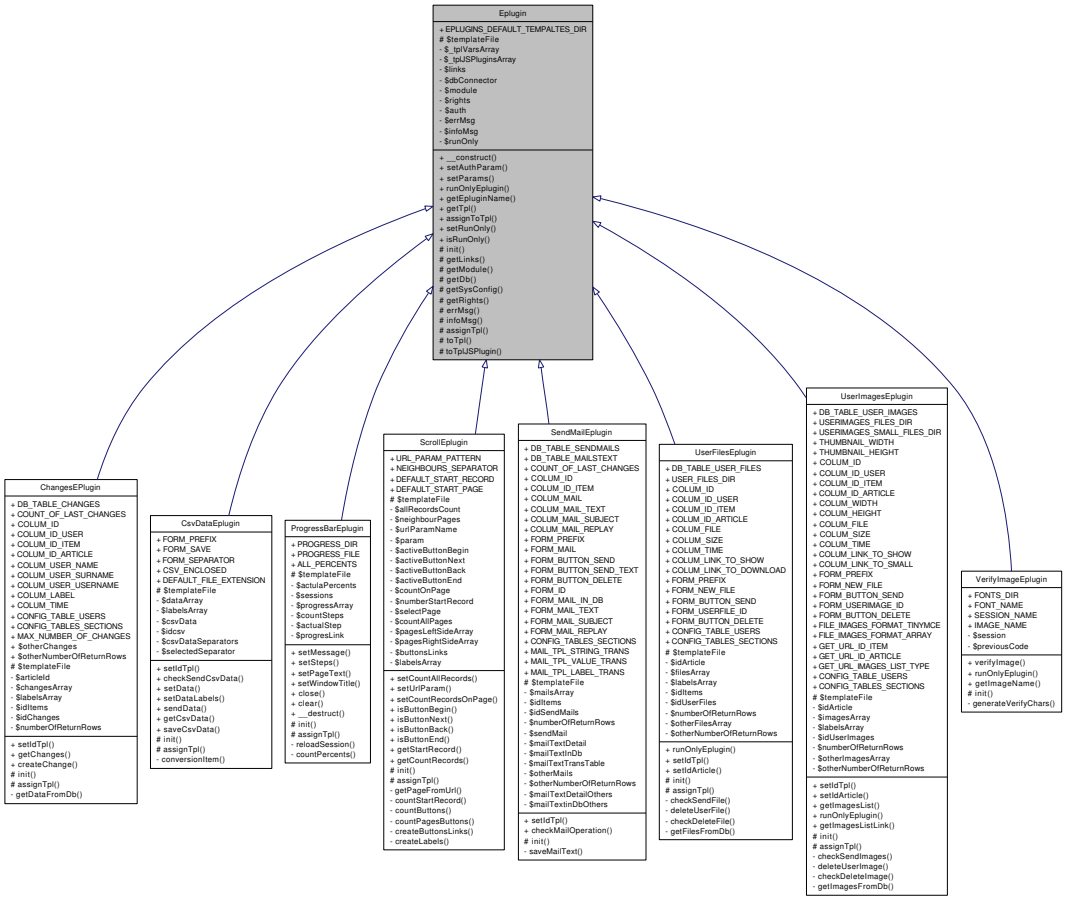
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Filesystem/dir.class.php

8.28 Dokumentace třídy Eplugin

Abstraktní třída pro Engine Pluginy - EPlugins.

Diagram dědičnosti pro třídu Eplugin



Veřejné metody

- __construct (Rights $rights=null)

  Konstruktor třídy, spouští metodu init();.

- setAuthParam (Auth $auth)

  Metoda nastavuje knihovnu epluginu.

- setParams ()

  Metoda nastavuje knihovnu epluginu je použita v epluginech.

- runOnlyEplugin ()

  Metoda se využívá pro načtení proměných do stránky, je volána při volání parametru stránky pro EPlugin.

- getEpluginName ()

  Metoda vrací název epluginu.

- getTpl ()

  Metoda vrací název šablony.

- assignToTpl (Template $template)

  Metoda zařadí proměné epluginu do šablony Je volána z viewru.

Statické veřejné metody

- static setRunOnly ($value)

  Metoda nastavuje že je eplugin spouštěn samostatně nebo ne.

- static isRunOnly ()

  Metoda vrací jestli je eplugin spouštěn samostatně nebo jako součást stránky.

Veřejné atributy

- const **EPLUGINS_DEFAULT_TEMPALTES_DIR** = AppCore::TEMPLATES_DIR

Chráněné metody

- init ()

  Inicializační metoda EPluginu.

- getLinks ($clear=false, $onlyWebRoot=false)

  Metoda vrací objekt k tvorbě odkazů.

- getModule ()

  Metoda vrací objekt modulu.

- getDb ()

  Metoda vrací objekt k připojení k db.

- getSysConfig ()

  Metoda vrací objekt ke konfiguraci enginu.

- getRights ()

  Metoda vrací objekt autorizace a právům k modulům.

- errMsg ()

  Metoda vrací objekt chbových zpráv.

- infoMsg ()

  Metoda vrací objekt informačních zpráv.

- assignTpl ()

Metoda obstarává přiřazení proměných do šablony.

- toTpl ($tplValueName, $value)

  Metoda pro asociování proměných do šablony.

- toTplJSPlugin ($jsPlugin)

  Metoda přidává zadaný objekt JSpluginu do šablony.

Chráněné atributy

- **$templateFile** = null

8.28.1 Detailní popis

Abstraktní třída pro Engine Pluginy - EPlugins.

Třída obsahuje základní metody pro vytváření EPluginu a práci s nimi (např. scroll, comments, atd.).

Copyright (c) 2008 Jakub Matas

Verze:

$Id: eplugin.class.php 3.1.8 13.11.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro vytvoření Epluginu

Plánované úpravy

implementovat generování názvu souborů pro zvolený eplugin

Definice je uvedena na řádku 14 v souboru eplugin.class.php.

8.28.2 Dokumentace konstruktoru a destruktoru

8.28.2.1 Eplugin::__construct (Rights $ *rights* = null)

Konstruktor třídy, spouští metodu init();.

Plánované úpravy

dodělat, tak ať se to nčítá třeba přímo z modulu nebo kategoorie, nebo jádra

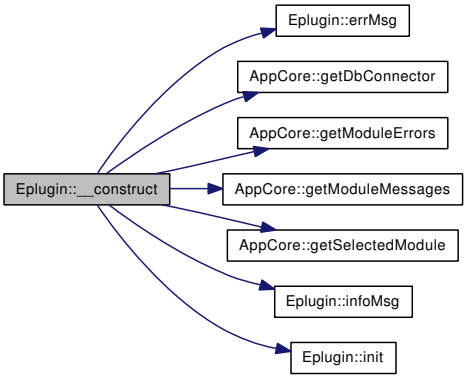
Definice je uvedena na řádku 90 v souboru eplugin.class.php.

Odkazuje se na errMsg(), AppCore::getDbConnector(), AppCore::getModuleErrors(), AppCore::getModuleMessages(), AppCore::getSelectedModule(), infoMsg() a init().

```
91    {
92       $this->module = AppCore::getSelectedModule();
93       $this->dbConnector = AppCore::getDbConnector();
94
```

```
95        if($rights instanceof Rights){
96           $this->rights = $rights;
97           $this->auth = $rights->getAuth();
98        }
102 //     else if(AppCore::getSelectedModule() instanceof Module){
103 //             $this->rights =
104 //     }
105
106       if(AppCore::getModuleErrors() instanceof Messages){
107          $this->errMsg = AppCore::getModuleErrors();
108       } else {
109          $this->errMsg = new Messages();
110       }
111       if(AppCore::getModuleMessages() instanceof Messages){
112          $this->infoMsg = AppCore::getModuleMessages();
113       } else {
114          $this->infoMsg = new Messages();
115       }
116       $this->init();
117    }
```

Tato funkce volá...



8.28.3 Dokumentace k metodám

8.28.3.1 Eplugin::setAuthParam (Auth $ *auth*)

Metoda nastavuje knihovnu epluginu.

Parametry:

Auth – objekt autorizace

Definice je uvedena na řádku 123 v souboru eplugin.class.php.

```
123                                          {
124       $this->auth = $auth;
125    }
```

8.28.3.2 static Eplugin::setRunOnly ($ *value*) [static]

Metoda nastavuje že je eplugin spouštěn samostatně nebo ne.

Parametry:

boolean $value – true pokud je eplugin spoštěn samostatně

Definice je uvedena na řádku 131 v souboru eplugin.class.php.

Používá se v AppCore::runApp().

```
131                                             {
132         self::$runOnly = $value;
133     }
```

8.28.3.3 static Eplugin::isRunOnly () [static]

Metoda vrací jestli je eplugin spouštěn samostatně nebo jako součást stránky.

Návratová hodnota:

boolean – true pokud je spuštěn samostatně

Definice je uvedena na řádku 140 v souboru eplugin.class.php.

Používá se v VerifyImageEplugin::init().

```
140                                         {
141         return self::$runOnly;
142     }
```

8.28.3.4 Eplugin::getLinks ($ *clear* = false, $ *onlyWebRoot* = false) [protected]

Metoda vrací objekt k tvorbě odkazů.

Návratová hodnota:

Links – objekt odkazů

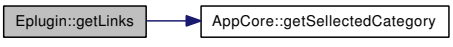
Definice je uvedena na řádku 168 v souboru eplugin.class.php.

Odkazuje se na AppCore::getSellectedCategory().

Používá se v ProgressBarEplugin::init().

```
169     {
170         $link = new Links($clear, $onlyWebRoot);
171         $cat = AppCore::getSellectedCategory();
172         if($cat != false){
173             $link->category($cat[Category::COLUM_CAT_LABEL], $cat[Category::COLUM_CAT_ID]);
174         }
175 //        $cat = Category::getCurrentCategory();
176         return $link;
177     }
```

Tato funkce volá...



8.28.3.5 Eplugin::getModule () [protected]

Metoda vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 184 v souboru eplugin.class.php.

Používá se v ChangesEPlugin::createChange().

```
185     {
186        return $this->module;
187     }
```

8.28.3.6 Eplugin::getDb () [protected]

Metoda vrací objekt k připojení k db.

Návratová hodnota:

DbInterface – objekt Db

Definice je uvedena na řádku 195 v souboru eplugin.class.php.

Používá se v ChangesEPlugin::createChange().

```
196     {
197        return $this->dbConnector;
198     }
```

8.28.3.7 Eplugin::getSysConfig () [protected]

Metoda vrací objekt ke konfiguraci enginu.

Návratová hodnota:

Config – objekt Config

Definice je uvedena na řádku 205 v souboru eplugin.class.php.

Odkazuje se na AppCore::sysConfig().

```
206     {
207        return AppCore::sysConfig();
208     }
```

Tato funkce volá...

8.28.3.8 Eplugin::getRights () [protected]

Metoda vrací objekt autorizace a právům k modulům.

Návratová hodnota:

Rights – objekt Rights

Definice je uvedena na řádku 215 v souboru eplugin.class.php.

Používá se v ChangesEPlugin::createChange().

```
216     {
217        return $this->rights;
218     }
```

8.28.3.9 Eplugin::errMsg () [protected]

Metoda vrací objekt chbových zpráv.

Návratová hodnota:

Messages – objekt chybových zpráv

Definice je uvedena na řádku 225 v souboru eplugin.class.php.

Používá se v __construct(), SendMailEplugin::checkMailOperation() a VerifyImageEplugin::verifyImage().

```
226     {
227        return $this->errMsg;
228     }
```

8.28.3.10 Eplugin::infoMsg () [protected]

Metoda vrací objekt informačních zpráv.

Návratová hodnota:

Messages – objekt informačních zpráv

Definice je uvedena na řádku 235 v souboru eplugin.class.php.

Používá se v __construct() a SendMailEplugin::checkMailOperation().

```
236     {
237        return $this->infoMsg;
238     }
```

8.28.3.11 Eplugin::getEpluginName () [final]

Metoda vrací název epluginu.

Návratová hodnota:

string – název epluginu

Definice je uvedena na řádku 244 v souboru eplugin.class.php.

```
244                                         {
245         return strtolower(get_class($this));
246     }
```

8.28.3.12 Eplugin::getTpl () [final]

Metoda vrací název šablony.

Návratová hodnota:

string – název šablony

Definice je uvedena na řádku 252 v souboru eplugin.class.php.

```
253     {
254 //      echo $this->templateFile;
255         return $this->templateFile;
256     }
```

8.28.3.13 Eplugin::assignToTpl (Template $ *template*)

Metoda zařadí proměné epluginu do šablony Je volána z viewru.

Parametry:

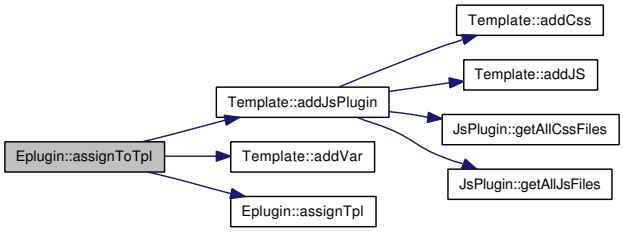
Template – objekt šablony

Definice je uvedena na řádku 264 v souboru eplugin.class.php.

Odkazuje se na Template::addJsPlugin(), Template::addVar() a assignTpl().

```
264                                                 {
265 //      Přiřazení proměnných do pole
266         $this->assignTpl();
267
268 //      Vložení proměných do šablony
269         foreach ($this->_tplVarsArray as $var => $value) {
270             $template->addVar($var, $value);
271         }
272
273 //      Vložení JSPluginů
274         foreach ($this->_tplJSPluginsArray as $jsPlugin) {
275             $template->addJsPlugin($jsPlugin);
276         }
277     }
```

Tato funkce volá...



8.28.3.14 Eplugin::toTpl ($ *tplValueName*, $ *value*) [final, protected]

Metoda pro asociování proměných do šablony.

Parametry:

string – název proměnné

mixed – hodnota proměnné

Definice je uvedena na řádku 291 v souboru eplugin.class.php.

Používá se v ScrollEplugin::assignTpl(), ProgressBarEplugin::assignTpl(), CsvDataEplugin::assignTpl() a ChangesEPlugin::assignTpl().

```
292     {
293         $this->_tplVarsArray[$tplValueName] = $value;
294     }
```

8.28.3.15 Eplugin::toTplJSPlugin ($ *jsPlugin*) [final, protected]

Metoda přidává zadaný objekt JSpluginu do šablony.

Parametry:

JSPlugin – objekt JSPluginu

Definice je uvedena na řádku 300 v souboru eplugin.class.php.

Používá se v ProgressBarEplugin::assignTpl() a ChangesEPlugin::assignTpl().

```
300                                                 {
301         array_push($this->_tplJSPluginsArray, $jsPlugin);
302     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/eplugin.class.php

8.29 Dokumentace třídy Errors

Třída slouží pro obsluhu chyb ve stránce.

Veřejné metody

- __construct ()
  Konstruktor třídy Vytvoří pole pro ukládání chyb.
- addError ($message)
  Metoda přidává chybu do pole chyb.
- getErrorsToStdIO ()
  Metody vypíše chyby na standartní výstup.
- isEmpty ()
  Metoda vrací jestli je pole se zprávami prázdné.

8.29.1 Detailní popis

Třída slouží pro obsluhu chyb ve stránce.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: errors.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu chyb

Zastaralé

Plánované úpravy

odstranit (popřípadě pokud se využije dodělat popisy a dokumentaci)

Definice je uvedena na řádku 13 v souboru errors.class.php.

8.29.2 Dokumentace k metodám

8.29.2.1 Errors::isEmpty ()

Metoda vrací jestli je pole se zprávami prázdné.

Návratová hodnota:

boolean – true pokud je pole prázdné

Definice je uvedena na řádku 51 v souboru errors.class.php.

```
52    {
53       return empty($this->errorsArray);
54    }
```

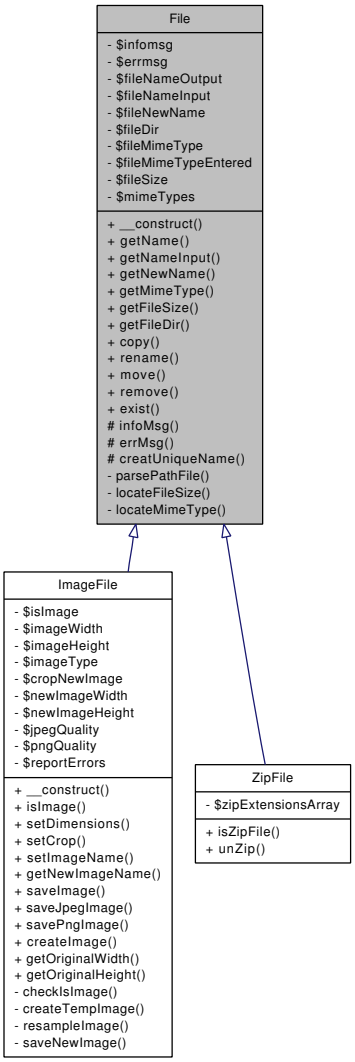
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Exceptions/errors.class.php

8.30 Dokumentace třídy File

Třída pro obsluhu souborů.

Diagram dědičnosti pro třídu File



Veřejné metody

- __construct ($file, $dir=null, $inputFile=null, $mimeType=null, $size=null)

  Konstruktor třídy.

- getName ($withDir=false)

  Metoda vrací název výstupního (OUTPUT) souboru, pokud existuje nový vrací ten.

- getNameInput ($withDir=false)

  Metoda vrací název vstupního (INPUT) souboru.

- getNewName ()
  Metoda vrací nový název souboru pokud existuje jinak starý.
- getMimeType ()
  Metoda vrací mime typ souboru.
- getFileSize ()
  Metoda vrací velikost souboru.
- getFileDir ()
  Metoda vrací adresář souboru.
- copy ($dstDir, $fileName=null)
  Metoda kopíruje soubor do zadaného adresáře, kontroluje jméno a vytváří unikátní název.
- rename ($newName)
  Metoda přejmenuje soubor na nový název.
- move ($dstDir, $newName=null)
  Metoda přesune soubor do zadaného adresáře, vytvoří unikátní název nebo přejmenuje na zadaný název.
- remove ()
  Metoda vymaže soubor z flesystému.
- exist ()
  Metoda zjišťuje jestli soubor existuje.

Chráněné metody

- infoMsg ()
  Metoda vrací objekt s informačními zprávami.
- errMsg ()
  Metoda vrací objekt s chybovými zprávami.
- creatUniqueName ($destinationDir, $newName=null, $number=0)
  Funkce vytvoří nový název souboru, který je v zadaném adresáři unikátní.

8.30.1 Detailní popis

Třída pro obsluhu souborů.

Třída poskytuje základní metody pro práci se soubory, zjišťování mime typu, ukládání do filesystému, kopírování, mazání.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0 $Revision:$

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída pro obsluhu souborů

Definice je uvedena na řádku 14 v souboru file.class.php.

8.30.2 Dokumentace konstruktoru a destruktoru

8.30.2.1 File::__construct ($ *file*, $ *dir* = `null`, $ *inputFile* = `null`, $ *mimeType* = `null`, $ *size* = `null`)

Konstruktor třídy.

Parametry:

string $file – název souboru (může být i s cestou - sám si to rozparsuje)

string $dir – (option) cesta k souboru (pokud není zapsána, pokusí se cestu parsovat z názvu souboru)

string $inputFile – název vstupního souboru např. z $_FILES pokud je odlišný od výstupního

string $mimeType – mime typ souboru

integer $size – velikost souboru

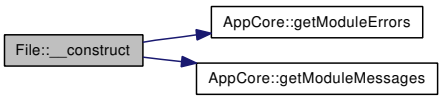
Definice je uvedena na řádku 140 v souboru file.class.php.

Odkazuje se na AppCore::getModuleErrors() a AppCore::getModuleMessages().

```
140                                                                                {
141       if(AppCore::getModuleMessages() instanceof Messages){
142          $this->infomsg = AppCore::getModuleMessages();
143       }
144
145       if(AppCore::getModuleErrors() instanceof Messages){
146          $this->errmsg = AppCore::getModuleErrors();
147       }
```

```
148
149         // Pokud je vložen objekt File
150         if($file instanceof File){
151            $this->fileDir = $file->getFileDir();
152            $this->fileNameOutput = $file->getName();
153            $this->fileNameInput = $file->getNameInput();
154            $this->fileMimeType = $file->getMimeType();
155            $this->fileSize = $file->getFileSize();
156         } else {
157
158            // Pokud je zadán tmp name
159            if($inputFile != null){
160               // rozparsování cesty a soubou u tmp
161               $arr = $this->parsePathFile($inputFile);
162               if($arr != false){
163                  $this->fileNameInput = $arr[2];
164                  $this->fileDir = $arr[1];
165               }
166               $this->fileNameOutput = $file;
167            }
168            // pokud je zadán název popřípadě cesta
169            else {
170               if($dir == null){
171                  // rozparsování cesty a soubou u tmp
172                  $arr = $this->parsePathFile($file);
173                  if($arr != false){
174                     $this->fileNameOutput = $arr[2];
175                     $this->fileNameInput = $arr[2];
176                     $this->fileDir = $arr[1];
177                  }
178               } else {
179                  $this->fileNameOutput = $file;
180                  $this->fileNameInput = $file;
181                  $this->fileDir = $dir;
182               }
183            }
184
185            $this->fileMimeTypeEntered = $mimeType;
186            $this->fileMimeType = $this->locateMimeType();
187
188            if($size != null){
189               $this->fileSize = $size;
190            }
191         }
192      }
```

Tato funkce volá...



8.30.3 Dokumentace k metodám

8.30.3.1 File::infoMsg () [final, protected]

Metoda vrací objekt s informačními zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 198 v souboru file.class.php.

```
198                                     {
199         return $this->infomsg;
200     }
```

8.30.3.2 File::errMsg () [final, protected]

Metoda vrací objekt s chybovými zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 206 v souboru file.class.php.

```
206                                     {
207         return $this->errmsg;
208     }
```

8.30.3.3 File::getName ($ *withDir* = false)

Metoda vrací název výstupního (OUTPUT) souboru, pokud existuje nový vrací ten.

Parametry:

boolean $withDir – jestli má být vrácena i část s adresářem

Návratová hodnota:

string – název souboru

Definice je uvedena na řádku 228 v souboru file.class.php.

Odkazuje se na getFileDir().

Používá se v creatUniqueName() a ImageFile::saveImage().

```
228                                             {
229         $return = null;
230         if($this->fileNewName == null){
231             $return = $this->fileNameOutput;
232         } else {
233             $return = $this->fileNewName;
234         }
235
236         if($withDir){
237             $return = $this->getFileDir().$return;
238         }
239
240         return $return;
241     }
```

Tato funkce volá...

8.30.3.4 File::getNameInput ($ *withDir* = `false`)

Metoda vrací název vstupního (INPUT) souboru.

Parametry:

boolean $withDir – jestli má být vrácena i část s adresářem

Návratová hodnota:

string – název souboru

Definice je uvedena na řádku 249 v souboru file.class.php.

Odkazuje se na getFileDir().

Používá se v copy(), exist() a remove().

```
249                                                 {
250         $return = $this->fileNameInput;
251
252         if($withDir){
253            $return = $this->getFileDir().$return;
254         }
255
256         return $return;
257     }
```

Tato funkce volá...

8.30.3.5 File::getNewName ()

Metoda vrací nový název souboru pokud existuje jinak starý.

Návratová hodnota:

string – název souboru

Definice je uvedena na řádku 264 v souboru file.class.php.

```
264                                 {
265         if($this->fileNewName == null){
266            return $this->fileNameOutput;
267         } else {
268            return $this->fileNewName;
269         }
270     }
```

8.30.3.6 File::getMimeType ()

Metoda vrací mime typ souboru.

Plánované úpravy

nutná portace na PECL rozšíření o informací o souboru

Návratová hodnota:

string – mime typ souboru

Definice je uvedena na řádku 278 v souboru file.class.php.

Používá se v ZipFile::isZipFile().

```
278                                 {
279         return $this->fileMimeType;
280     }
```

8.30.3.7 File::getFileSize ()

Metoda vrací velikost souboru.

Návratová hodnota:

integer – velikost souboru

Definice je uvedena na řádku 287 v souboru file.class.php.

```
287                                 {
288         if($this->fileSize == -1){
289             $this->fileSize = $this->locateFileSize();
290         }
291         return $this->fileSize;
292     }
```

8.30.3.8 File::getFileDir ()

Metoda vrací adresář souboru.

Návratová hodnota:

string – adresář souboru

Definice je uvedena na řádku 299 v souboru file.class.php.

Používá se v getName() a getNameInput().

```
299                                 {
300         return $this->fileDir;
301     }
```

8.30.3.9 File::copy ($ *dstDir*, $ *fileName* = `null`)

Metoda kopíruje soubor do zadaného adresáře, kontroluje jméno a vytváří unikátní název.

Parametry:

string $dstDir – cílový adresář
string $fileName – (option) název souboru

Návratová hodnota:

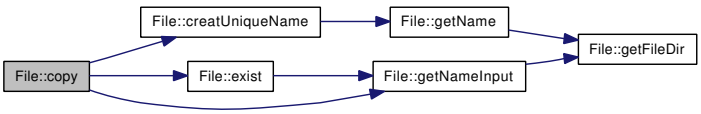
boolean – true pokud byl soubor zkopírován

Definice je uvedena na řádku 311 v souboru file.class.php.

Odkazuje se na creatUniqueName(), exist() a getNameInput().

```
311                                                 {
312       // Kontrola adresáře
313       if(!file_exists($dstDir)){
314          $dirObj = new Dir();
315          $dirObj->createDirs($dstDir);
316          unset ($dirObj);
317       }
318
319 //      Kontrola jména
320       $newFile = $this->creatUniqueName($dstDir);
321       $this->fileNewName = $newFile;
322
323       if($this->exist()){
324          $return = copy($this->getNameInput(true), $dstDir.$newFile);
325          chmod($dstDir.$fileName, 0666);
326       } else {
327          $return = false;
328       }
329
330       return $return;
331    }
```

Tato funkce volá...



8.30.3.10 File::rename ($ *newName*)

Metoda přejmenuje soubor na nový název.

Parametry:

string $newName – nový název souboru

Návratová hodnota:

boolean – true pokud byl soubor přejmenován

Definice je uvedena na řádku 339 v souboru file.class.php.

```
339                                     {
340       ;
341    }
```

8.30.3.11 File::move ($ *dstDir*, $ *newName* = null)

Metoda přesune soubor do zadaného adresáře, vytvoří unikátní název nebo přejmenuje na zadaný název.

Parametry:

string $dstDir – cílový adresář

string $newName – (option) nový název souboru

Návratová hodnota:

boolean – true pokud byl soubor přesunut

Definice je uvedena na řádku 351 v souboru file.class.php.

```
351                                                 {
352         ;
353     }
```

8.30.3.12 File::remove ()

Metoda vymaže soubor z flesystému.

Návratová hodnota:

boolean – true pokud byl soubor odstraněn

Definice je uvedena na řádku 360 v souboru file.class.php.

Odkazuje se na exist() a getNameInput().

```
360                             {
361         if($this->exist()){
362             return unlink($this->getNameInput(true));
363         }
364     }
```

Tato funkce volá...

8.30.3.13 File::exist ()

Metoda zjišťuje jestli soubor existuje.

Návratová hodnota:

boolean – true pokud soubor existuje

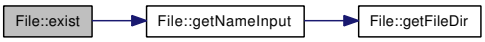
Definice je uvedena na řádku 371 v souboru file.class.php.

Odkazuje se na getNameInput().

Používá se v copy() a remove().

```
371                             {
372         return file_exists($this->getNameInput(true));
373     }
```

Tato funkce volá...



8.30.3.14 File::creatUniqueName ($ *destinationDir*, $ *newName* = `null`, $ *number* = `0`) `[protected]`

Funkce vytvoří nový název souboru, který je v zadaném adresáři unikátní.

Parametry:

string – adresář, kde se bude soubor vytvářet

string – (option) nový název souboru

integer – (option) číslo které se přikládá za soubor

Návratová hodnota:

string – nový název souboru

Plánované úpravy

– dodělat při přijmu souboru se dvěmi příponami

Definice je uvedena na řádku 416 v souboru file.class.php.

Odkazuje se na getName().

Používá se v copy() a ImageFile::saveImage().

```
416                                                                                                 {
417 //      $numberOfArguments = 2;
418 //      $arguments = func_num_args();
419
420         // Pokud je zadáno pouze číslo
421         if(is_int($newName)){
422            $newFileName = $this->getName() ;
423            $addNumber = $newName;
424         } else if($newName != null) {
425            $newFileName = $newName;
426            $addNumber = $number;
427         } else {
428            $newFileName = $this->getName();
429            $addNumber = $number;
430         }
431
432         //doplnění posledního lomítka za dest adresář
```

```
433       if($destinationDir[strlen($destinationDir)-1] != "/" AND $addNumber == 0){
434          $destinationDir .= "/";
435       }
436
437       //rozdělení názvu souboru na název a příponu
438       $file_ext = array();
439       eregi('^([^.]*).(.*)$', strtolower($newFileName), $file_ext);
440       $file_name_short = $file_ext[1];
441       $file_name_extension = $file_ext[2];
442
443       //odstraneni nepovolenych zanků a složení dohromady
444       $sFunction = new SpecialFunctions();
445
446       $file_name_short = $sFunction->utf2ascii($file_name_short);
447       unset($sFunction);
448
449
450       if($addNumber == 0){
451          $createNewFileName=$file_name_short.'.'.$file_name_extension;
452       } else {
453          $createNewFileName=$file_name_short.$addNumber.'.'.$file_name_extension;
454       }
455
456       // kontrola existence
457       if(file_exists($destinationDir.$createNewFileName)){
458          $createNewFileName = $this->creatUniqueName($destinationDir, (++$addNumber));
459       } else {
460          $this->fileNewName = $createNewFileName;
461       }
462
463       return $createNewFileName;
464    }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Filesystem/file.class.php

8.31 Dokumentace třídy FileModel

Abstraktní třída pro ouborový Model.

Diagram dědičnosti pro třídu FileModel

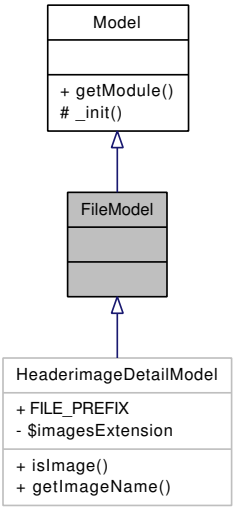
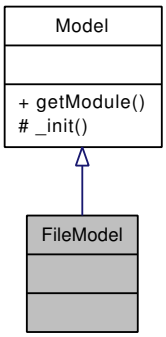


Diagram tříd pro FileModel:



8.31.1 Detailní popis

Abstraktní třída pro ouborový Model.

Třída pro vytvoření objektu modelu, pro práci se soubory. Obsahuje pouze přístup k vybranému modulu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: filemodel.class.php 3.0.55 26.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro vytvoření souborového modelu

Definice je uvedena na řádku 13 v souboru filemodel.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/models/filemodel.class.php

8.32 Dokumentace třídy Files

Třída pro obsluhu souborů.

Veřejné metody

- copyAs ($srcFile, $dstDir, $newName)

  Metoda překopíruje zadaný soubor do zadaného adresáře pod novým jménem //TODO not implemented.

- createNewFileName ($file, $destinationDir)

  Funkce vytvoří nový název souboru, který je v zadaném adresáři unikátní.

- checkDir ($directory)

  Metoda otestuje existenci adresáře, a pokud neexistuje pokusí se jej vytvořit.

- unZip ($src_file, $dest_dir=false, $create_zip_name_dir=true, $overwrite=true)

  Rozbali zip soubor do cílového adresáře.

- deleteFile ($dstDir, $file)

  Metoda smaže zadaný soubor.

- rmDir ($filepath)

  Metoda maže rekjurzivně zadaný adresář/soubor.

- exist ($file)

  Metoda zjišťuje, zdali zadaný soubor existuje.

- checkDirPath ($path)

  Metoda kontroluje jestli je cesta zadána správně, jinak vrací opravenou cestu.

8.32.1 Detailní popis

Třída pro obsluhu souborů.

Základní třída pro práci se soubory. Umožňuje základní přístup k filesystému, tj. vytváření a mazání adresářů, kopírování, mazání a přesun souborů souborů, popřípadě zjišťování jejich existence.

Verze:

$Id: files.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu souborů

Definice je uvedena na řádku 14 v souboru files.class.php.

8.32.2 Dokumentace k metodám

8.32.2.1 Files::copyAs ($ *srcFile*, $ *dstDir*, $ *newName*)

Metoda překopíruje zadaný soubor do zadaného adresáře pod novým jménem //TODO not implemented.

Parametry:

string – název souboru

string – adresář, kam se má soubor nakopírovat

string – název souboru

Definice je uvedena na řádku 26 v souboru files.class.php.

```
26                                                      {
27       if(!file_exists($dstDir)){
28          $this->createDirs($dstDir);
29       }
30       $return = copy($srcFile, $dstDir.$newName);
31
32       chmod($dstDir.$newName, 0666);
33
34       return $return;
35
36    }
```

8.32.2.2 Files::createNewFileName ($ *file*, $ *destinationDir*)

Funkce vytvoří nový název souboru, který je v zadaném adresáři unikátní.

Parametry:

string – název souboru

string – adresář, kde se bude soubor vytvářet

Návratová hodnota:

string – nový název souboru

Definice je uvedena na řádku 73 v souboru files.class.php.

```
73                                                              {
74       $numberOfArguments = 3;
75       $arguments = func_num_args();
76
77       $addNumber = 0;
78 //    Pokud je argumentů více než dva, je použit v rekurzi
79       if($arguments >= $numberOfArguments){
80          $addNumber = func_get_arg($numberOfArguments-1);
81       }
82
83       //doplnění posledního lomítka za dest adresář
84       if($destinationDir[strlen($destinationDir)-1] != "/" AND $addNumber == 0){
85          $destinationDir .= "/";
86       }
87
88       //rozdělení názvu obrázku na název a příponu
```

```
89        $file_name=strtolower($file);
90        $file_name_short=ereg_replace("\.[a-zA-Z]{3,4}$", "", $file_name);
91        $tempCount=strlen($file_name_short);
92        $tempCountAll=strlen($file_name);
93        $file_name_extension=substr($file_name,$tempCount, $tempCountAll-$tempCount);
94
95        //odstraneni nepovolenych zanků a složení dohromady
96        $sFunction = new SpecialFunctions();
97
98        $file_name_short = $sFunction->utf2ascii($file_name_short);
99        unset($sFunction);
100
101
102        if($addNumber == 0){
103           $new_file_name=$file_name_short.$file_name_extension;
104        } else {
105           $new_file_name=$file_name_short.$addNumber.$file_name_extension;
106        }
107
108        if(file_exists($destinationDir.$new_file_name)){
109           $new_file_name = $this->createNewFileName($file, $destinationDir, (++$addNumber));
110        }
111
112        return $new_file_name;
113     }
```

8.32.2.3 Files::checkDir ($ *directory*)

Metoda otestuje existenci adresáře, a pokud neexistuje pokusí se jej vytvořit.

Parametry:

string – název adresáře

Definice je uvedena na řádku 119 v souboru files.class.php.

```
119                                        {
120        //TODO dodělat přidávání lomítek před adresář
121
122        //doplnění posledního lomítka za dest adresář
123        if($directory[strlen($directory)-1] != "/"){
124           $directory .= "/";
125        }
126
127        if(!file_exists($directory) OR !is_dir($directory)){
128           $this->createDirs($directory);
129        }
130     }
```

8.32.2.4 Files::unZip ($ *src_file*, $ *dest_dir* = `false`, $ *create_zip_name_dir* = `true`, $ *overwrite* = `true`)

Rozbali zip soubor do cílového adresáře.

Parametry:

string – Cesta k zip souboru

string – Cesta, kam se zip soubor rozbalí, (false rozbalí soubor do aktuálního adresáře se zip souborem)

boolean – Jestli má být zip soubor rozbalen do adresáře se stejným jménem

boolean – Jestli mají být soubory přepsány

Návratová hodnota:

boolean Succesful or not

Definice je uvedena na řádku 142 v souboru files.class.php.

```
143    {
144       if(function_exists("zip_open"))
145       {
146          if(!is_resource(zip_open($src_file)))
147          {
148             $src_file=dirname($_SERVER['SCRIPT_FILENAME'])."/".$src_file;
149          }
150
151          if (is_resource($zip = zip_open($src_file)))
152          {
153             $splitter = ($create_zip_name_dir === true) ? "." : "/";
154             if ($dest_dir === false) $dest_dir = substr($src_file, 0, strrpos($src_file, $splitter))."
155
156             // Create the directories to the destination dir if they don't already exist
157             $this->createDirs($dest_dir);
158
159             // For every file in the zip-packet
160             while ($zip_entry = zip_read($zip))
161             {
162                // Now we're going to create the directories in the destination directories
163
164                // If the file is not in the root dir
165                $pos_last_slash = strrpos(zip_entry_name($zip_entry), "/");
166                if ($pos_last_slash !== false)
167                {
168                   // Create the directory where the zip-entry should be saved (with a "/" at the end)
169                   $this->createDirs($dest_dir.substr(zip_entry_name($zip_entry), 0, $pos_last_slash+1)
170                }
171
172                // Open the entry
173                if (zip_entry_open($zip,$zip_entry,"r"))
174                {
175
176                   // The name of the file to save on the disk
177                   $file_name = $dest_dir.zip_entry_name($zip_entry);
178
179                   // Check if the files should be overwritten or not
180                   if ($overwrite === true || $overwrite === false && !is_file($file_name))
181                   {
182                      // Get the content of the zip entry
183                      $fstream = zip_entry_read($zip_entry, zip_entry_filesize($zip_entry));
184
185                      if(!is_dir($file_name))
186                      file_put_contents($file_name, $fstream );
187                      // Set the rights
188                      if(file_exists($file_name))
189                      {
190                         chmod($file_name, 0777);
191                      }
192                   }
193
194                   // Close the entry
195                   zip_entry_close($zip_entry);
196                }
197             }
198             // Close the zip-file
```

```
199             zip_close($zip);
200          }
201          else
202          {
203             //        echo "No Zip Archive Found.";
204             return false;
205          }
206
207          return true;
208       }
209       else
210       {
211          if(version_compare(phpversion(), "5.2.0", "<"))
212          $infoVersion="(use PHP 5.2.0 or later)";
213
214          new CoreException(_('Je potřeba PHP zip rozšíření pro práci se zip archívy ').$infoVersion);
215       }
216    }
```

8.32.2.5 Files::deleteFile ($ *dstDir*, $ *file*)

Metoda smaže zadaný soubor.

Parametry:

string – cesta k souboru

string – název souboru

Návratová hodnota:

boolean – true pokud byl soubor smazán

Definice je uvedena na řádku 226 v souboru files.class.php.

```
226                                                {
227       $deleted = false;
228       if(file_exists($dstDir.$file)){
229          if(unlink($dstDir.$file)){
230             $deleted = true;
231          }
232       }
233       return $deleted;
234    }
```

8.32.2.6 Files::rmDir ($ *filepath*)

Metoda maže rekjurzivně zadaný adresář/soubor.

Parametry:

string – adresář/soubor, který se má smazat

Návratová hodnota:

boolena – true pokud se podařilo adresář/soubor smazat

Definice je uvedena na řádku 242 v souboru files.class.php.

```
243    {
244       if (is_dir($filepath) && !is_link($filepath)){
245          if ($dh = opendir($filepath)){
246             while (($sf = readdir($dh)) !== false){
247                if ($sf == '.' || $sf == '..'){
248                   continue;
249                }
250                if (!$this->rmDir($filepath.'/'.$sf)){
251                   new CoreException($filepath.'/'.$sf._(' soubor nemohl být smazán.'));
252                }
253             }
254             closedir($dh);
255          }
256          return rmdir($filepath);
257       }
258       if(file_exists($filepath)){
259          return unlink($filepath);
260       }
261    }
```

8.32.2.7 Files::exist ($*file*)

Metoda zjišťuje, zdali zadaný soubor existuje.

Parametry:

string – název a cesta k souboru

Definice je uvedena na řádku 268 v souboru files.class.php.

```
268                                {
269       return file_exists($file);
270    }
```

8.32.2.8 Files::checkDirPath ($*path*)

Metoda kontroluje jestli je cesta zadána správně, jinak vrací opravenou cestu.

Parametry:

string – cesta

Návratová hodnota:

string – opravená cesta

Definice je uvedena na řádku 279 v souboru files.class.php.

```
279                                      {
280       if(($path[strlen($path)-1] != '/') AND ($path[strlen($path)-1] != '\\')){
281          $path.=DIRECTORY_SEPARATOR;
282       }
283       return $path;
284    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/files.class.php

8.33 Dokumentace třídy Form

Třída pro obsluhu formuláře Třída implementuje řešení pro obsluhu formulářových prvku.

Veřejné metody

- __construct ($formPrefix=null)

  Konstruktor nastaví základní parametry.

- setPrefix ($prefix)

  Metoda nastavuje prefix použitý ve formuláři.

- crSubmit ($name)

  Metody vytváří prvek typu INPUT - SUBMIT, tedy prvke pro potvrzení formuláře.

- crInputText ($name, $obligation=false, $langs=false, $specialValidation=self::VALIDATION_NONE, $code=self::CODE_HTMLENCODE, $maxChars=null, $minChars=null)

  Metody vytváří prvek typu INPUT - TEXT.

- crInputHidden ($name, $obligation=false, $specialValidation=self::VALIDATION_NONE, $code=self::CODE_HTMLENCODE)

  Metody vytváří prvek typu INPUT - TEXT.

- crInputCheckboxn ($name)

  Metody vytváří prvek typu INPUT - CHECKBOX.

- crInputFile ($name, $obligation=false)

  Metody vytváří prvek typu INPUT - FILE.

- crInputPassword ($name, $obligation=false, $specialValidation=self::VALIDATION_NONE, $code=self::CODE_HTMLENCODE, $maxChars=null, $minChars=null)

  Metody vytváří prvek typu INPUT - TEXT.

- crTextArea ($name, $obligation=false, $langs=false, $code=self::CODE_HTMLENCODE, $maxChars=null, $minChars=null)

  Metody vytváří prvek typu TEXTAREA.

- checkForm ()

  Metoda zkontroluje, jestli byl formulář odeslán a překontroluje všechny prvky.

- getErrorItems ()

  Metoda vrací pole s chybně zadanými prvky.

- getValues ($oneArray=false, $withPrefix=false, $operator= '_')

  Metoda vrací hodnoty formuláře jako pole hodnot.

- getValue ($itemName, $oneArray=false, $withPrefix=false, $separator= '_')

  Metoda vrací hordnodu prvku ve formuláři.

- setValue ($itemName, $value)

Metoda nastavuje hodnotu prvkum formuláře.

- debug ()

Veřejné atributy

- const ITEM_NAME = 'name'

  Názvy parametrů formuláře.

- const **ITEM_VALUE** = 'value'
- const **ITEM_OBLIGATION** = 'obligation'
- const **ITEM_LANGS** = 'langs'
- const **ITEM_CODE** = 'code'
- const **ITEM_VALIDATION** = 'validation'
- const **ITEM_MAX_LENGHT** = 'maxlenght'
- const **ITEM_MIN_LENGHT** = 'minlenght'
- const INPUT_SUBMIT = 'inputsubmit'

  Názvy prvků ve formuláři.

- const **INPUT_TEXT** = 'inputtext'
- const **INPUT_HIDDEN** = 'inputhidden'
- const **INPUT_PASSWORD** = 'inputpasswd'
- const **INPUT_CHECKBOX** = 'inputcheckbox'
- const **INPUT_FILE** = 'inputfile'
- const **INPUT_TEXTAREA** = 'textarea'
- const **CODE_NONE** = 0
- const **CODE_HTMLENCODE** = 1
- const **CODE_HTMLDECODE** = 2
- const **VALIDATION_NONE** = 0
- const **VALIDATION_EMAIL** = 1
- const **VALIDATE_DATE** = 2
- const **VALIDATE_TIME** = 3
- const ERROR_MISSING = 'missing'

  Index pro chybějící zprávy.

- const **POST_FILES_ERROR** = 'error'
- const **POST_FILES_ORIGINAL_NAME** = 'name'
- const **POST_FILES_SIZE** = 'size'
- const **POST_FILES_TYPE** = 'type'
- const **POST_FILES_TMP_NAME** = 'tmp_name'

8.33.1 Detailní popis

Třída pro obsluhu formuláře Třída implementuje řešení pro obsluhu formulářových prvku.

Umožňuje kontrolu jejich odeslání, správného vyplnění zadaných dat, jejich načtení a upráva. Lze pomocí ní také vybrat data z formuláře a rovnou předat modelu pro zápis. Umožňuje také generování podle jazykového nastavení

Verze:

$Id:$ VVE3.5.0

Revision

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída pro obsluhu formulářových prvků

Plánované úpravy

Dodělat další validace, implementovat ostatní prvky formulářů

Definice je uvedena na řádku 16 v souboru form.class.php.

8.33.2 Dokumentace konstruktoru a destruktoru

8.33.2.1 Form::__construct ($*formPrefix* = `null`) `[final]`

Konstruktor nastaví základní parametry.

Parametry:

string $formPrefix – prefix formulářových prvků první úrovně

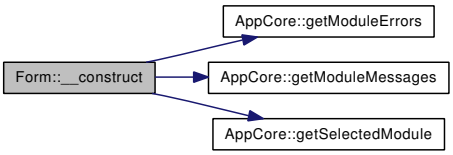
Definice je uvedena na řádku 164 v souboru form.class.php.

Odkazuje se na AppCore::getModuleErrors(), AppCore::getModuleMessages() a AppCore::getSelectedModule().

```
164                                                     {
165         if(AppCore::getSelectedModule() instanceof Module){
166             $this->module = AppCore::getSelectedModule();
167         }
168 
169         if(AppCore::getModuleMessages() instanceof Messages){
170             $this->infomsg = AppCore::getModuleMessages();
171         }
```

```
172
173         if(AppCore::getModuleErrors() instanceof Messages){
174            $this->errmsg = AppCore::getModuleErrors();
175         }
176
177         $this->formPrefix = $formPrefix;
178     }
```

Tato funkce volá...



8.33.3 Dokumentace k metodám

8.33.3.1 Form::setPrefix ($ *prefix*)

Metoda nastavuje prefix použitý ve formuláři.

Parametry:

string $prefix – prefix formulářových prvků

Návratová hodnota:

Form

Definice je uvedena na řádku 201 v souboru form.class.php.

```
201                                     {
202         $this->formPrefix = $prefix;
203         return $this;
204     }
```

8.33.3.2 Form::crSubmit ($ *name*)

Metody vytváří prvek typu INPUT - SUBMIT, tedy prvke pro potvrzení formuláře.

Parametry:

string $name – název tlačítka

Návratová hodnota:

Form

Definice je uvedena na řádku 216 v souboru form.class.php.

```
216                                     {
217         $this->formStructure[self::INPUT_SUBMIT] = $name;
218         $this->formValues[$name] = false;
219         return $this;
220     }
```

8.33.3.3 Form::crInputText ($ *name*, $ *obligation* = `false`, $ *langs* = `false`, $ *specialValidation* = `self::VALIDATION_NONE`, $ *code* = `self::CODE_HTMLENCODE`, $ *maxChars* = `null`, $ *minChars* = `null`)

Metody vytváří prvek typu INPUT - TEXT.

Parametry:

string $name – Název prvku

boolean $obligation – (option) jestli se jedná o povinný prvek

boolean $langs – (option) jestli má prvek možnost jazykové mutace

mixed $specialValidation – (option) typ vylidace prvku (výchozí je žádná, odvíjí se od konstat třídy pro validaci) nebo funkci (is_string,...)

int $code – (option) typ kódování (výchozí je dekódování všech prvků na html entity, odvíjí se od konstatn třídy po kódovaní)

int $maxChars – maximální počet znaků

int $minChars – minimální počet znaků

Návratová hodnota:

Form

Definice je uvedena na řádku 236 v souboru form.class.php.

```
238                                         {
239
240       $inputArray = array ();
241       $inputArray[self::ITEM_NAME] = $name;
242       $inputArray[self::ITEM_OBLIGATION] = $obligation;
243       $inputArray[self::ITEM_LANGS] = $langs;
244       $inputArray[self::ITEM_CODE] = $code;
245       $inputArray[self::ITEM_VALIDATION] = $specialValidation;
246       $inputArray[self::ITEM_MAX_LENGHT] = $maxChars;
247       $inputArray[self::ITEM_MIN_LENGHT] = $minChars;
248
249       $this->formStructure[self::INPUT_TEXT][$name] = $inputArray;
250
251       if($langs){
252          $this->formValues[$name] = $this->createLangArray();
253       } else {
254          $this->formValues[$name] = null;
255       }
256       return $this;
257    }
```

8.33.3.4 Form::crInputHidden ($ *name*, $ *obligation* = `false`, $ *specialValidation* = `self::VALIDATION_NONE`, $ *code* = `self::CODE_HTMLENCODE`)

Metody vytváří prvek typu INPUT - TEXT.

Parametry:

string $name – Název prvku

boolean $obligation – (option) jestli se jedná o povinný prvek

mixed $specialValidation – (option) typ vylidace prvku (výchozí je žádná, odvíjí se od konstat třídy pro validaci) nebo lze volat funkci (is_numeric,...)

int $code – (option) typ kódování (výchozí je dekódování všech prvků na html entity, odvíjí se od konstatn třídy po kódovaní)

Návratová hodnota:

Form

Definice je uvedena na řádku 270 v souboru form.class.php.

```
271                                                                                 {
272
273         $inputArray = array ();
274         $inputArray[self::ITEM_NAME] = $name;
275         $inputArray[self::ITEM_OBLIGATION] = $obligation;
276         $inputArray[self::ITEM_CODE] = $code;
277         $inputArray[self::ITEM_VALIDATION] = $specialValidation;
278
279         $this->formStructure[self::INPUT_HIDDEN][$name] = $inputArray;
280
281         $this->formValues[$name] = null;
282
283         return $this;
284     }
```

8.33.3.5 Form::crInputCheckboxn ($ *name*)

Metody vytváří prvek typu INPUT - CHECKBOX.

Parametry:

string $name – Název prvku

Návratová hodnota:

Form

Definice je uvedena na řádku 292 v souboru form.class.php.

```
292                                             {
293         $this->formStructure[self::INPUT_CHECKBOX][$name][self::ITEM_NAME] = $name;
294         $this->formValues[$name] = false;
295
296         return $this;
297     }
```

8.33.3.6 Form::crInputFile ($ *name*, $ *obligation* = `false`)

Metody vytváří prvek typu INPUT - FILE.

Parametry:

string $name – Název prvku

boolean $obligation – jestli je zadaný prvek povinný

Návratová hodnota:

Form

Definice je uvedena na řádku 306 v souboru form.class.php.

```
306                                                         {
307       $this->formStructure[self::INPUT_FILE][$name][self::ITEM_NAME] = $name;
308       $this->formStructure[self::INPUT_FILE][$name][self::ITEM_OBLIGATION] = $obligation;
309       $this->formValues[$name] = null;
310
311       return $this;
312    }
```

8.33.3.7 Form::crInputPassword ($ *name*, $ *obligation* = `false`, $ *specialValidation* = `self::VALIDATION_NONE`, $ *code* = `self::CODE_HTMLENCODE`, $ *maxChars* = `null`, $ *minChars* = `null`)

Metody vytváří prvek typu INPUT - TEXT.

Parametry:

string $name – Název prvku

boolean $obligation – (option) jestli se jedná o povinný prvek

boolean $langs – (option) jestli má prvek možnost jazykové mutace

mixed $specialValidation – (option) typ vylidace prvku (výchozí je žádná, odvíjí se od konstat třídy pro validaci) nebo funkci (is_string,...)

int $code – (option) typ kódování (výchozí je dekódování všech prvků na html entity, odvíjí se od konstatn třídy po kódovaní)

int $maxChars – maximální počet znaků

int $minChars – minimální počet znaků

Návratová hodnota:

Form

Definice je uvedena na řádku 328 v souboru form.class.php.

```
330                                             {
331
332       $inputArray = array ();
333       $inputArray[self::ITEM_NAME] = $name;
334       $inputArray[self::ITEM_OBLIGATION] = $obligation;
335       $inputArray[self::ITEM_CODE] = $code;
336       $inputArray[self::ITEM_VALIDATION] = $specialValidation;
337       $inputArray[self::ITEM_MAX_LENGHT] = $maxChars;
338       $inputArray[self::ITEM_MIN_LENGHT] = $minChars;
339       $this->formStructure[self::INPUT_PASSWORD][$name] = $inputArray;
340       return $this;
341    }
```

8.33.3.8 Form::crTextArea ($ *name*, $ *obligation* = `false`, $ *langs* = `false`, $ *code* = `self::CODE_HTMLENCODE`, $ *maxChars* = `null`, $ *minChars* = `null`)

Metody vytváří prvek typu TEXTAREA.

Parametry:

string $name – Název prvku

boolean $obligation – (option) jestli se jedná o povinný prvek

boolean $langs – (option) jestli má prvek možnost jazykové mutace

int $code – (option) typ kódování (výchozí je dekódování všech prvků na html entity, odvíjí se od konstatn třídy po kódovaní)

int $maxChars – maximální počet znaků

int $minChars – minimální počet znaků

Návratová hodnota:

Form

Definice je uvedena na řádku 356 v souboru form.class.php.

```
357                                                                         {
358
359         $inputArray = array ();
360
361         $inputArray[self::ITEM_NAME] = $name;
362         $inputArray[self::ITEM_OBLIGATION] = $obligation;
363         $inputArray[self::ITEM_LANGS] = $langs;
364         $inputArray[self::ITEM_CODE] = $code;
365         $inputArray[self::ITEM_MAX_LENGHT] = $maxChars;
366         $inputArray[self::ITEM_MIN_LENGHT] = $minChars;
367
368         $this->formStructure[self::INPUT_TEXTAREA][$name] = $inputArray;
369
370         if($langs){
371            $this->formValues[$name] = $this->createLangArray();
372         } else {
373            $this->formValues[$name] = null;
374         }
375         return $this;
376     }
```

8.33.3.9 Form::checkForm ()

Metoda zkontroluje, jestli byl formulář odeslán a překontroluje všechny prvky.

Návratová hodnota:

boolean

Definice je uvedena na řádku 387 v souboru form.class.php.

```
387                                 {
388         //    Pokud byl formulář odeslán
389 //        echo "<pre>";
390 //        print_r($_POST);
```

```
391 //      echo "</pre>";
392       if(isset ($_POST[$this->formPrefix.$this->formStructure[self::INPUT_SUBMIT]]) OR
393          isset ($_POST[$this->formPrefix.$this->formStructure[self::INPUT_SUBMIT].'_x'])){
394
395          $this->fillinForm();
396
397          return !$this->isError;
398       }
399       return false;
400    }
```

8.33.3.10 Form::getErrorItems ()

Metoda vrací pole s chybně zadanými prvky.

Návratová hodnota:

array – pole s názvy chybně zadaných prvků

Definice je uvedena na řádku 410 v souboru form.class.php.

```
410                                    {
411       return $this->errorItems;
412    }
```

8.33.3.11 Form::getValues ($ *oneArray* = `false`, $ *withPrefix* = `false`, $ *operator* = `'_'`)

Metoda vrací hodnoty formuláře jako pole hodnot.

Parametry:

boolean $oneArray(option) – true pokud má být vráceno pole s jednou hloubkou, všechny indexy podpolí budo sloučeny s hlavními indexy pomocí operátoru

boolean $withPrefix(option) – jestli do indexů bude přidán také prefix formuláře

string $operator(option) – oddělovací operátor mezi indexy při slučování

Definice je uvedena na řádku 422 v souboru form.class.php.

```
422                                                                                    {
423       $returnaArray = array();
424       //    Pokud má být prefix tak se doplní
425       if($withPrefix){
426          foreach ($this->formValues as $key => $val) {
427             $returnaArray[$this->formPrefix.$key] = $val;
428          }
429       } else {
430          $returnaArray = $this->formValues;
431       }
432
433       if($oneArray){
434          $returnaArray = $this->createOneArrayByKeys($returnaArray, null, $operator);
435       }
436       return $returnaArray;
437    }
```

8.33.3.12 Form::getValue ($ *itemName*, $ *oneArray* = `false`, $ *withPrefix* = `false`, $ *separator* = `'_'`)

Metoda vrací hordnodu prvku ve formuláři.

Parametry:

string $itemName – název formulářového prvku

boolean $withPrefix – jestli se má vracet i prefix formuláře

boolean $oneArray – jestli má být vráce pole o jednom rozměru, klíče budou sloučeny za sebe podle separátoru

string $separator – oddělovač klíčů v jednorozměrném poli

Definice je uvedena na řádku 447 v souboru form.class.php.

```
447                                                                                             {
448       $value = null;
449       if(key_exists($itemName, $this->formValues)){
450          //    if(isset ($this->formValues[$itemName])){
451          if($oneArray AND is_array($this->formValues[$itemName])){
452             //        foreach ($this->formValues[$itemName] as $key => $val) {
453             //          $value[$this->formPrefix.$key]
454             //        }
455             if(!$withPrefix){
456                $value = $this->createOneArrayByKeys($this->formValues[$itemName]);
457             } else {
458                $value = $this->createOneArrayByKeys($this->formValues[$itemName], $this->formPrefix);
459             }
460          } else {
461             //        if(!$withPrefix){
462             $value = $this->formValues[$itemName];
463             //        } else {
464             //          $value = $this->formValues[$itemName];
465             //        }
466          }
467
468
469       }
470       return $value;
471    }
```

8.33.3.13 Form::setValue ($ *itemName*, $ *value*)

Metoda nastavuje hodnotu prvkum formuláře.

Parametry:

string $itemName – název prvku

mixed $value – hodnota prvku

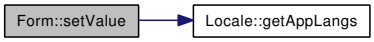
Definice je uvedena na řádku 478 v souboru form.class.php.

Odkazuje se na Locale::getAppLangs().

```
478                                               {
479       $item = $this->findItem($itemName);
480
```

```
481       if(!empty($item)){
482          //      Jedná lise o jazykovou verzy
483          if(isset ($item[self::ITEM_LANGS]) AND $item[self::ITEM_LANGS]){
484             $langs = Locale::getAppLangs();
485             foreach ($langs as $lang) {
486                if(isset ($value[$lang])){
487                   $this->formValues[$itemName][$lang] = $value[$lang];
488                }
489             }
490          } else {
491             $this->formValues[$itemName] = $value;
492          }
493          return true;
494       } else {
495          return false;
496       }
497    }
```

Tato funkce volá...



8.33.3.14 Form::debug ()

Plánované úpravy

Odstranit

Definice je uvedena na řádku 1120 v souboru form.class.php.

```
1120                            {
1121       echo "<pre>";
1122       print_r($this);
1123       echo '</pre>';
1124    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/form.class.php

8.34 Dokumentace třídy GaleryDetailModel

Model pro detail galerie.

Diagram dědičnosti pro třídu GaleryDetailModel

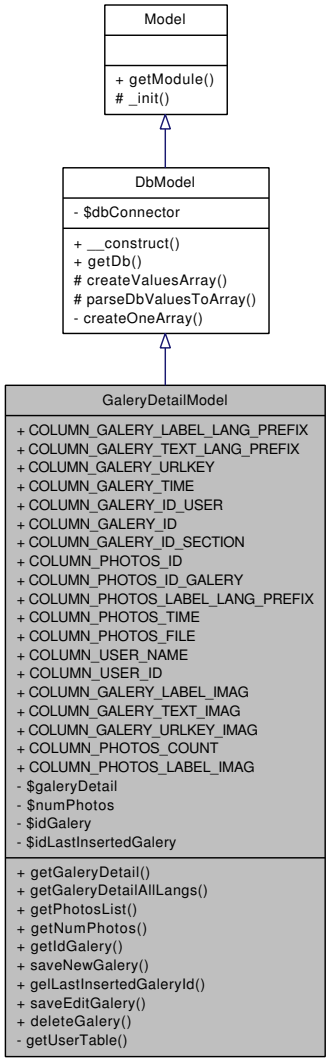
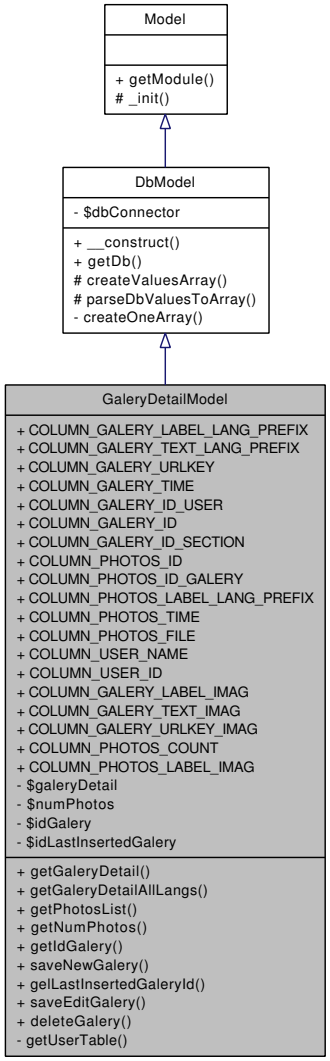


Diagram tříd pro GaleryDetailModel:



Veřejné metody

- getGaleryDetail ($urlKey)

  Metoda načte detail galerie.

- getGaleryDetailAllLangs ($urlKey)

  Metoda načte detail galerie se všemi jazykovými mutacemi.

- getPhotosList ($idGalery)

  Metoda načte seznam fotek v galerii.

- getNumPhotos ($idGalery)

  Metoda vrací počet fotek v galerii.

- getIdGalery ()

Metoda vrací id vybrané galerie.

- saveNewGalery ($galeryArray, $idSection, $mainLabel, $time=null, $idUser=0)

  Metoda uloží novou galerii.

- gelLastInsertedGaleryId ()

  Metoda vrací id poslední vložené galerie.

- saveEditGalery ($galeryArray, $urlkey, $idGalery=null)

  Metoda uloží upravenou galerii do db.

- deleteGalery ($id)

  Metoda vymaže galerie z db.

Veřejné atributy

- const **COLUMN_GALERY_LABEL_LANG_PREFIX** = 'label_'
- const **COLUMN_GALERY_TEXT_LANG_PREFIX** = 'text_'
- const **COLUMN_GALERY_URLKEY** = 'urlkey'
- const **COLUMN_GALERY_TIME** = 'time'
- const **COLUMN_GALERY_ID_USER** = 'id_user'
- const **COLUMN_GALERY_ID** = 'id_galery'
- const **COLUMN_GALERY_ID_SECTION** = 'id_section'
- const **COLUMN_PHOTOS_ID** = 'id_photo'
- const **COLUMN_PHOTOS_ID_GALERY** = 'id_galery'
- const **COLUMN_PHOTOS_LABEL_LANG_PREFIX** = 'label_'
- const **COLUMN_PHOTOS_TIME** = 'time'
- const **COLUMN_PHOTOS_FILE** = 'file'
- const **COLUMN_USER_NAME** = 'username'
- const **COLUMN_USER_ID** = 'id_user'
- const **COLUMN_GALERY_LABEL_IMAG** = 'galerylabel'
- const **COLUMN_GALERY_TEXT_IMAG** = 'galerytext'
- const **COLUMN_GALERY_URLKEY_IMAG** = 'galeryurlkey'
- const **COLUMN_PHOTOS_COUNT** = 'num_photos'
- const **COLUMN_PHOTOS_LABEL_IMAG** = 'photolabel'

8.34.1 Detailní popis

Model pro detail galerie.

Definice je uvedena na řádku 6 v souboru galerydetail.php.

8.34.2 Dokumentace k metodám

8.34.2.1 GaleryDetailModel::getGaleryDetail ($ *urlKey*)

Metoda načte detail galerie.

Parametry:

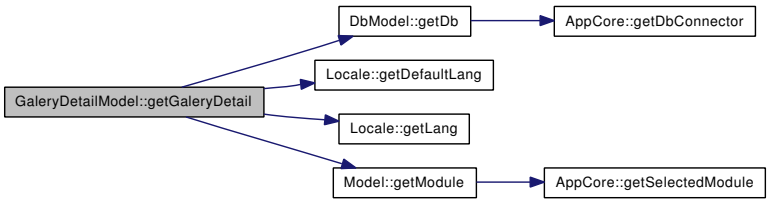
string – url klíč galerie

Definice je uvedena na řádku 105 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb(), Locale::getDefaultLang(), Locale::getLang() a Model::getModule().

```
105                                          {
106       if($this->galeryDetail == null){
107          $sqlSelectGaleries = $this->getDb()->select()->from(array('gals' => $this->getModule()->getDb
108                                   self::COLUMN_GALERY_TEXT_IMAG => "IFNULL(gals.".self::COLUMN_GALERY_T
109                                   self::COLUMN_GALERY_ID, self::COLUMN_GALERY_ID_USER, self::COLUMN_GAI
110                                   ->join(array('users' => $this->getUserTable()), 'users.'.self::COLUMN
111                                   ->join(array('photos'=>$this->getModule()->getDbTable()), 'photos.'.s
112                                   ->group('photos.'.self::COLUMN_PHOTOS_ID_GALERY)
113                                   ->where('gals.'.self::COLUMN_GALERY_URLKEY.' = \''.$urlKey.'\'')
114                                   ->order('gals.'.self::COLUMN_GALERY_TIME, 'DESC')
115                                   ->order('gals.'.self::COLUMN_GALERY_LABEL_LANG_PREFIX.Locale::getDefa
116
117          $this->galeryDetail = $this->getDb()->fetchAssoc($sqlSelectGaleries, true);
118
119          $this->idGalery = $this->galeryDetail[self::COLUMN_GALERY_ID];
120       }
121
122
123       return $this->galeryDetail;
124
125    }
```

Tato funkce volá...



8.34.2.2 GaleryDetailModel::getGaleryDetailAllLangs ($ *urlKey*)

Metoda načte detail galerie se všemi jazykovými mutacemi.

Parametry:

string – url klíč galerie

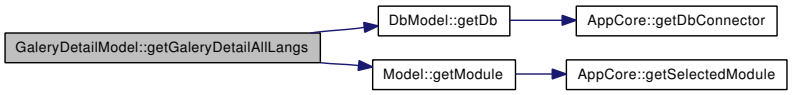
Definice je uvedena na řádku 133 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb() a Model::getModule().

```
133                                                  {
134          $sqlSelectGaleries = $this->getDb()->select()->from($this->getModule()->getDbTable(2))
135                                   ->where(self::COLUMN_GALERY_URLKEY.' = \''.$urlKey.'\'');
136
137          $galeryDetail = $this->getDb()->fetchAssoc($sqlSelectGaleries, true);
138
```

```
139
140         return $galeryDetail;
141
142     }
```

Tato funkce volá...



8.34.2.3 GaleryDetailModel::getPhotosList ($ *idGalery*)

Metoda načte seznam fotek v galerii.

Parametry:

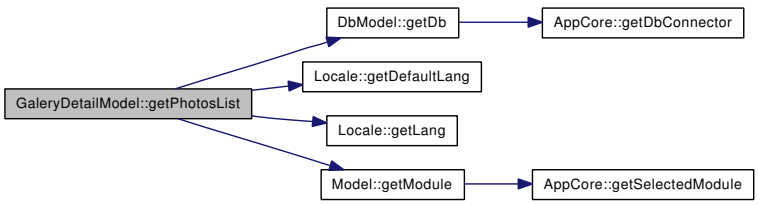
integer – id galerie

Definice je uvedena na řádku 163 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb(), Locale::getDefaultLang(), Locale::getLang() a Model::getModule().

```
163                                             {
164 //      Načtení fotek galerie
165         $sqlSelect = $this->getDb()->select()->from(array('photos'=>$this->getModule()->getDbTable(1)),
166                     photos.".self::COLUMN_PHOTOS_LABEL_LANG_PREFIX.Locale::getDefaultLang().")", self::C
167                     ->where('photos.'.self::COLUMN_PHOTOS_ID_GALERY." = '".$idGalery."'")
168                     ->order('photos.'.self::COLUMN_PHOTOS_TIME)
169                     ->order('photos.'.self::COLUMN_PHOTOS_LABEL_LANG_PREFIX.Locale::getDefaultLang());
170
171         $photos = $this->getDb()->fetchAssoc($sqlSelect);
172
173         return $photos;
174     }
```

Tato funkce volá...



8.34.2.4 GaleryDetailModel::getNumPhotos ($ *idGalery*)

Metoda vrací počet fotek v galerii.

Parametry:

integer – id galerie

Návratová hodnota:

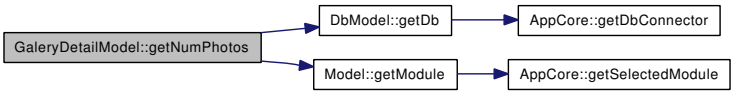
integer – počet fotek

Definice je uvedena na řádku 182 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb() a Model::getModule().

```
182                                             {
183       if($this->numPhotos == null){
184          $sqlCount = $this->getDb()->select()->from($this->getModule()->getDbTable(), array("count"=>"
185                                        ->where(self::COLUMN_GALERY_ID." = '".$idGalery."'");
186
187
188 //        Zjištění počtu záznamů
189          $count = $this->getDb()->fetchObject($sqlCount);
190          $this->numPhotos = $count->count;
191       }
192
193       return $this->numPhotos;
194    }
```

Tato funkce volá...



8.34.2.5 GaleryDetailModel::getIdGalery ()

Metoda vrací id vybrané galerie.

Návratová hodnota:

integer – id galerie

Definice je uvedena na řádku 201 v souboru galerydetail.php.

```
201                                 {
202       return $this->idGalery;
203    }
```

8.34.2.6 GaleryDetailModel::saveNewGalery ($ *galeryArray*, $ *idSection*, $ *mainLabel*, $ *time* = `null`, $ *idUser* = `0`)

Metoda uloží novou galerii.

Parametry:

array – pole s parametry galerie

integer – id sekce pro galerii

string – hlavní popis galerie

integer – časové razítko kdy byla galerie vytvořena

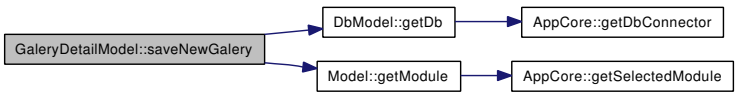
integer – id uživatele, který galerii přidal

Definice je uvedena na řádku 214 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb() a Model::getModule().

```
214                                                                                                                                                {
215 //       vygenerování url klíče
216          $dbHelper = new DbCtrlHelper();
217          $urlKey = $dbHelper->generateDatabaseUrlKey($mainLabel, $this->getModule()->getDbTable(2), se
218
219          //Vygenerování sloupců do kterých se bude zapisovat
220          $columsArray = array_keys($galeryArray);
221          $valuesArray = array_values($galeryArray);
222          array_push($columsArray, self::COLUMN_GALERY_ID_SECTION);
223          array_push($valuesArray, $idSection);
224          array_push($columsArray, self::COLUMN_GALERY_URLKEY);
225          array_push($valuesArray, $urlKey);
226
227          array_push($columsArray, self::COLUMN_GALERY_TIME);
228          if($time == null){
229             array_push($valuesArray, time());
230          } else {
231             array_push($valuesArray, $time);
232          }
233
234          array_push($columsArray, self::COLUMN_GALERY_ID_USER);
235          array_push($valuesArray, $idUser);
236
237
238          $sqlInsert = $this->getDb()->insert()->into($this->getModule()->getDbTable(2))
239                ->colums($columsArray)
240                ->values($valuesArray);
241
242       //    Vložení do db
243       if($this->getDb()->query($sqlInsert)){
244          $this->idLastInsertedGalery = $this->getDb()->getLastInsertedId();
245          return true;
246       } else {
247          return false;
248       }
249    }
```

Tato funkce volá...



8.34.2.7 GaleryDetailModel::gelLastInsertedGaleryId ()

Metoda vrací id poslední vložené galerie.

Návratová hodnota:

integer – id galerie

Definice je uvedena na řádku 255 v souboru galerydetail.php.

```
255                                             {
256         return $this->idLastInsertedGalery;
257     }
```

8.34.2.8 GaleryDetailModel::saveEditGalery ($ *galeryArray*, $ *urlkey*, $ *idGalery* = null)

Metoda uloží upravenou galerii do db.

Parametry:

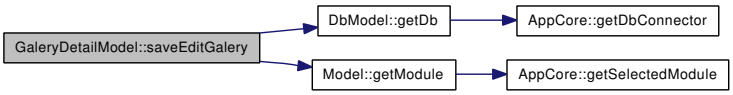
array – pole s detaily galerie

Definice je uvedena na řádku 265 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb() a Model::getModule().

```
265                                                                                 {
266         //TODO dodělat generování nového url klíče
267
268         $sqlInsert = $this->getDb()->update()->table($this->getModule()->getDbTable(2))
269                                     ->set($galeryArray)
270                                     ->where(self::COLUMN_GALERY_URLKEY." = '".$urlkey."'");
271
272         if($idGalery != null){
273            $sqlInsert = $sqlInsert->where(self::COLUMN_GALERY_ID." = ".$idGalery);
274         }
275
276         // vložení do db
277         if($this->getDb()->query($sqlInsert)){
278            return true;
279         } else {
280            return false;
281         };
282     }
```

Tato funkce volá...



8.34.2.9 GaleryDetailModel::deleteGalery ($ *id*)

Metoda vymaže galerie z db.

Parametry:

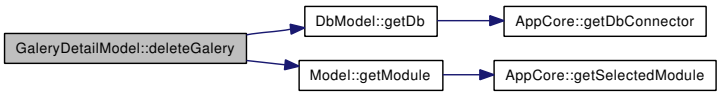
integer – id galerie

Definice je uvedena na řádku 289 v souboru galerydetail.php.

Odkazuje se na DbModel::getDb() a Model::getModule().

```
289                                     {
290       //    Končný výmaz z db
291       $sqlDelete = $this->getDb()->delete()->from($this->getModule()->getDbTable(2))
292                                     ->where(self::COLUMN_GALERY_ID.' = '.$id);
293
294       return $this->getDb()->query($sqlDelete);
295
296    }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/photogalery/models/galerydetail.php

8.35 Dokumentace třídy GuestbookAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu GuestbookAction

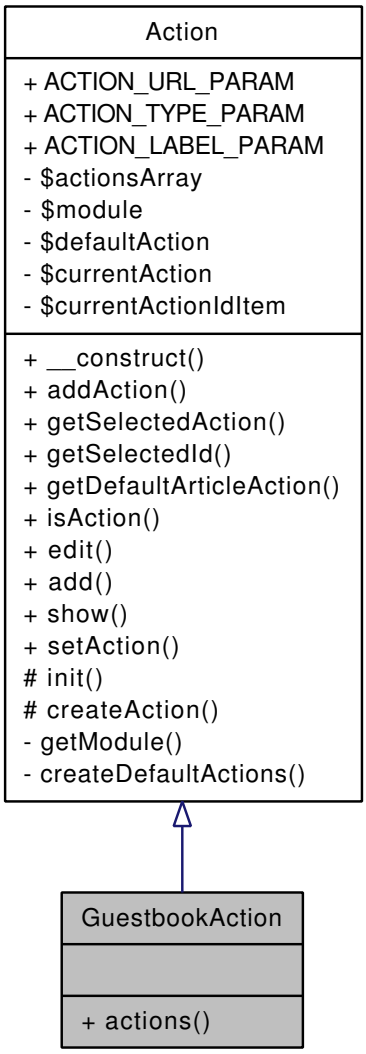
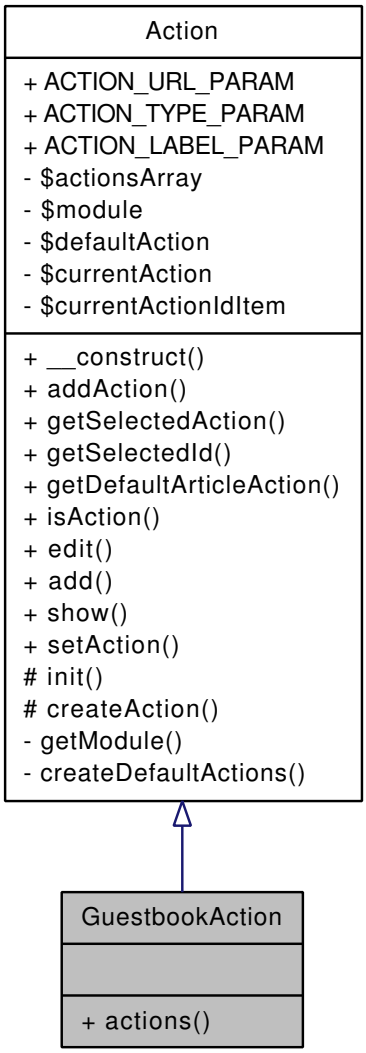


Diagram tříd pro GuestbookAction:



Veřejné metody

- **actions** ()

8.35.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/guestbook/action.class.php

8.36 Dokumentace třídy GuestbookController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu GuestbookController

Diagram tříd pro GuestbookController:

Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- addController ()

  Kontroler pro obsluhu přidání sponzora.

- editController ()

  Controler pro úpravu spozora.

Veřejné atributy

- const **FORM_PREFIX** = 'guestbook_'

- const **FORM_BUTTON_SEND** = 'send'
- const **FORM_TOPIC** = 'topic'
- const **FORM_NICK** = 'nick'
- const **FORM_EMAIL** = 'email'
- const **FORM_TEXT** = 'text'
- const **FORM_VERIFY** = 'verify'

8.36.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/guestbook/controler.class.php

8.37 Dokumentace třídy GuestbookRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu GuestbookRoutes

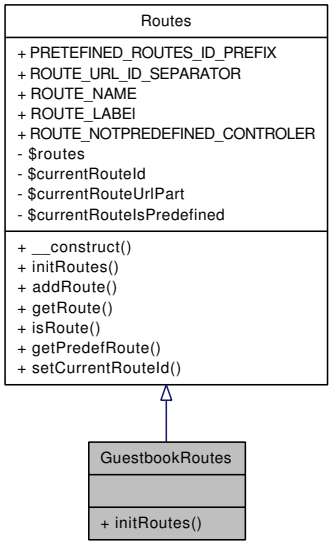
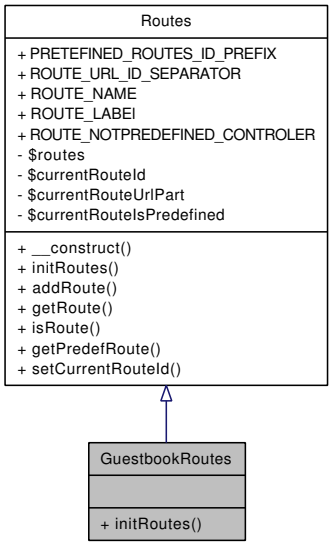


Diagram tříd pro GuestbookRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

8.37.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/guestbook/routes.class.php

8.38 Dokumentace třídy GuestbookView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu GuestbookView

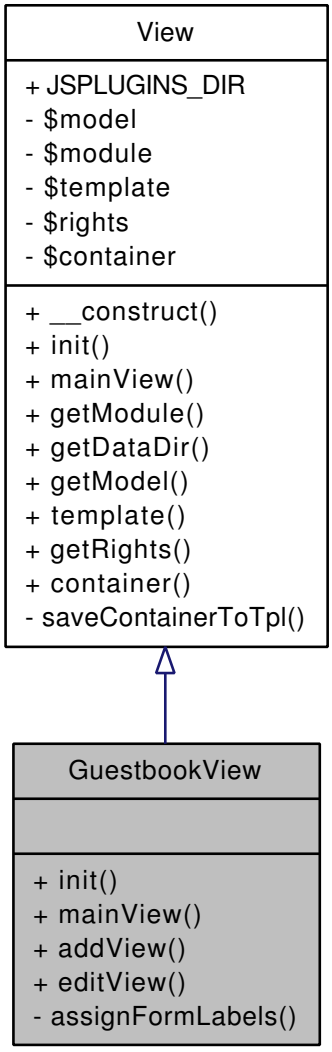
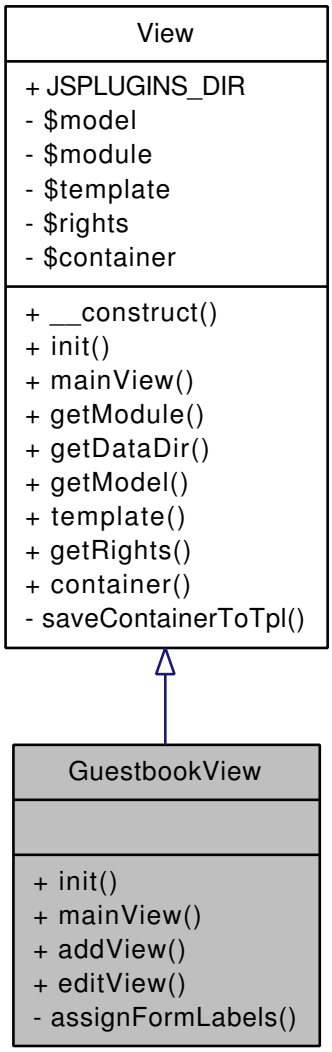


Diagram tříd pro GuestbookView:



Veřejné metody

- init ()

  Metoda, která se provede vždy.

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- addView ()

  Viewer pro přidání sponzora.

- editView ()

  Viewver pro zobrazení editace.

8.38.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/guestbook/view.class.php

8.39 Dokumentace třídy HaderimagesAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu HaderimagesAction

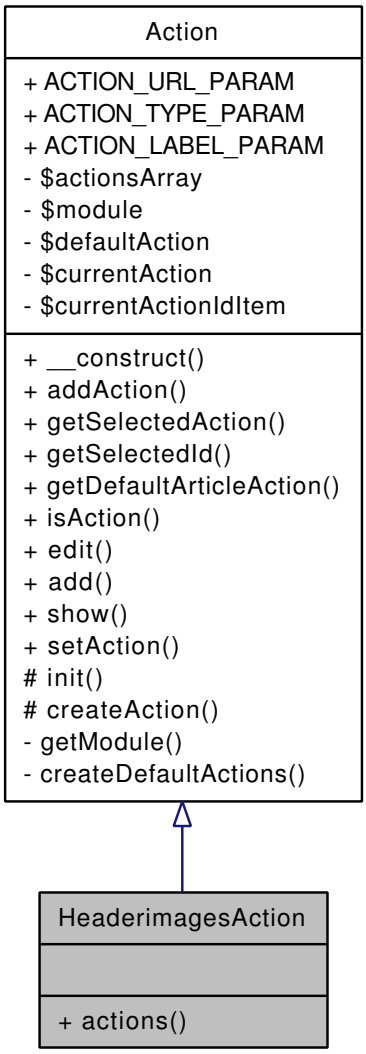
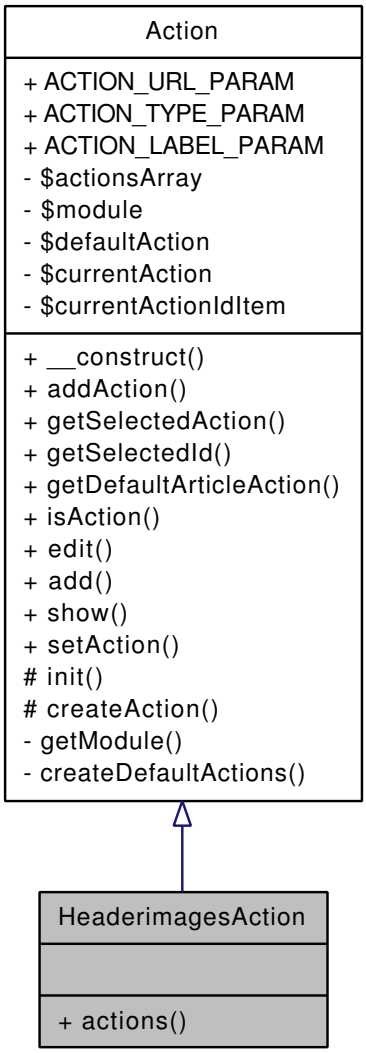


Diagram tříd pro HeaderimagesAction:



Veřejné metody

- **actions** ()

8.39.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/headerimages/action.class.php

8.40 Dokumentace třídy HeaderimagesRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu HeaderimagesRoutes

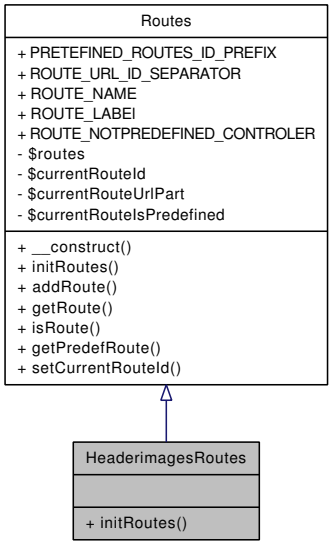
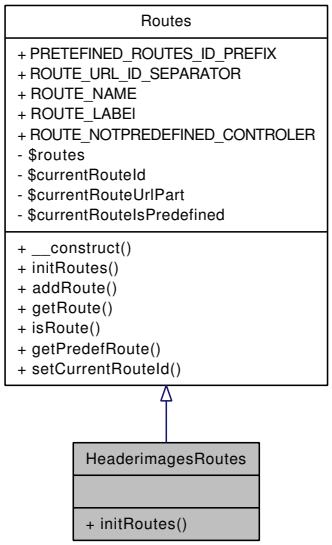


Diagram tříd pro HeaderimagesRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

8.40.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/headerimages/routes.class.php

8.41 Dokumentace třídy HeaderimagesView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu HeaderimagesView

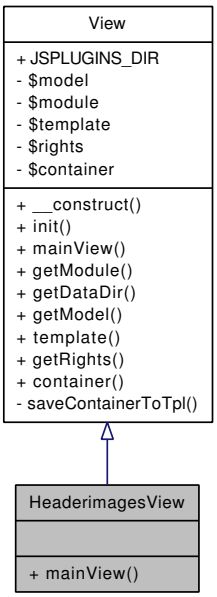
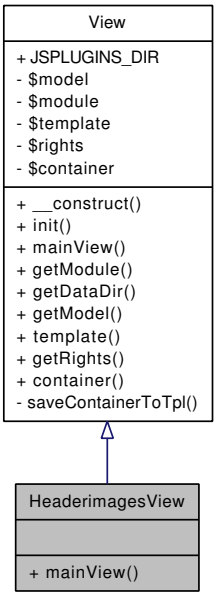


Diagram tříd pro HeaderimagesView:



Veřejné metody

- mainView ()

Hlavní abstraktní třída pro vytvoření pohledu.

8.41.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

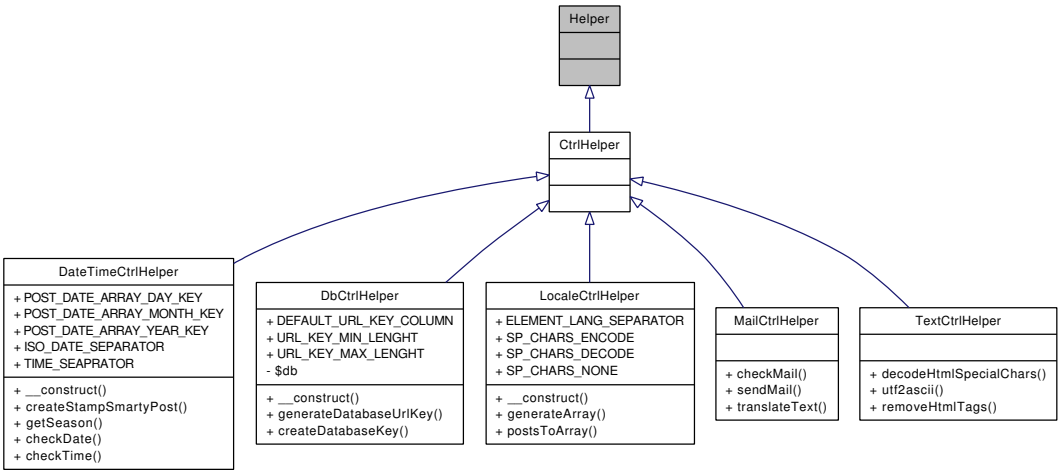
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/headerimages/view.class.php

8.42 Dokumentace třídy Helper

Abstraktní třída pro Helper.

Diagram dědičnosti pro třídu Helper



8.42.1 Detailní popis

Abstraktní třída pro Helper.

Základní třída pro jakýkoliv helper.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: helper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro vytvoření helper

Definice je uvedena na řádku 12 v souboru helper.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/helper.class.php

8.43 Dokumentace třídy ImageFile

Třída pro práci s obrázky Třída pro základní práci s obrázky.

Diagram dědičnosti pro třídu ImageFile

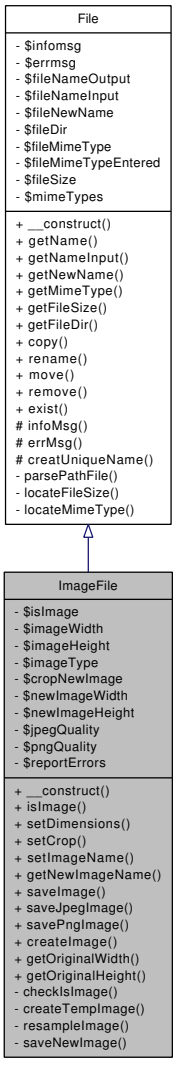
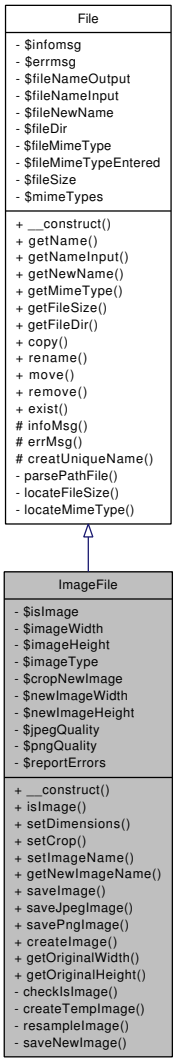


Diagram tříd pro ImageFile:



Veřejné metody

- __construct ($file, $dir=null, $reportErrors=true)

  Konstruktor třídy.

- isImage ()

  Matedoa zjišťuje, jestli je daný soubor obrázek.

- setDimensions ($width, $height)

  metoda nastaví rozměry obrázku

- setCrop ($crop)

  Metoda zapíná/vypíná ořezání obrázku.

- setImageName ($name)

Metoda nastaví jméno nového obrázku.

- getNewImageName ()

  Metoda vrací název uloženého obrázku.

- saveImage ($dstDir, $width=null, $heigh=null, $newName=null, $imageType=null)

  Metoda uloží obrázek ve stejném formátu, v jakém byl zadán.

- saveJpegImage ($dstDir, $width=null, $heigh=null, $newName=null)

  Metoda uloží JPEG obrázek do zadané cesty.

- savePngImage ($dstDir, $width=null, $heigh=null, $newName=null)

  Metoda uloží PNG obrázek do zadané cesty.

- createImage ($srcFile, $destFile, $width, $height, $cropSize=false, $jpegQuality=85, $pngQuality=9)

  Funkce vytvoří obrázek ze zadaného obrázku a uloží jej do specifikovaného souboru.

- getOriginalWidth ()

  Metoda vrací rozměr původního obrázku - Šířku.

- getOriginalHeight ()

  Metoda vrací rozměr původního obrázku - Výšku.

8.43.1 Detailní popis

Třída pro práci s obrázky Třída pro základní práci s obrázky.

Umožňuje jejich ukládání, ořezávání, změnu velikost a změnu formátu obrázku.

Verze:

Id

VVE3.5.0 $Revision:$

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída pro práci s obrázky

Definice je uvedena na řádku 14 v souboru imagefile.class.php.

8.43.2 Dokumentace konstruktoru a destruktoru

8.43.2.1 ImageFile::__construct ($ *file*, $ *dir* = `null`, $ *reportErrors* = `true`)

Konstruktor třídy.

Parametry:

Messages $errors – objekt chybových hlášek
string – název souboru
boolean – jestli mají být hlášky hlášeny nebo ne

Definice je uvedena na řádku 87 v souboru imagefile.class.php.

```
87                                                                  {
88
89        if($file instanceof File){
90           parent::__construct($file);
91        } else {
92           parent::__construct($file, $dir);
93        }
94
95        $this->reportErrors = $reportErrors;
96
97        $this->checkIsImage();
98     }
```

8.43.3 Dokumentace k metodám

8.43.3.1 ImageFile::isImage ()

Matedoa zjišťuje, jestli je daný soubor obrázek.

Návratová hodnota:

boolean – true pokud se jedná o obrázek se kterým umí pracovat

Definice je uvedena na řádku 105 v souboru imagefile.class.php.

Používá se v saveImage().

```
105                              {
106
107        return $this->isImage;
108     }
```

8.43.3.2 ImageFile::setDimensions ($ *width*, $ *height*)

metoda nastaví rozměry obrázku

Parametry:

integer – šířka v px

integer – výška v px

Definice je uvedena na řádku 144 v souboru imagefile.class.php.

```
144                                                 {
145         $this->newImageWidth = $width;
146         $this->newImageHeight = $height;
147     }
```

8.43.3.3 ImageFile::setCrop ($ *crop*)

Metoda zapíná/vypíná ořezání obrázku.

Parametry:

boolean – true pro zapnutí ořezání

Definice je uvedena na řádku 153 v souboru imagefile.class.php.

```
153                                 {
154         $this->cropNewImage = $crop;
155     }
```

8.43.3.4 ImageFile::setImageName ($ *name*)

Metoda nastaví jméno nového obrázku.

Parametry:

string – název nového obrázku

Definice je uvedena na řádku 163 v souboru imagefile.class.php.

```
163                                     {
164         $this->newImageName = $name;
165     }
```

8.43.3.5 ImageFile::getNewImageName ()

Metoda vrací název uloženého obrázku.

Návratová hodnota:

string – název obrázku

Definice je uvedena na řádku 172 v souboru imagefile.class.php.

```
172                                     {
173         return $this->newImageName;
174     }
```

8.43.3.6 ImageFile::saveImage ($ *dstDir*, $ *width* = null, $ *heigh* = null, $ *newName* = null, $ *imageType* = null)

Metoda uloží obrázek ve stejném formátu, v jakém byl zadán.

Parametry:

string – cílový adresář, kam se obrázek ukládá

int – šířka výsledného obrázku

int – výška výsledného obrázku

string – nový název obrázku

int – typ výsledného obrázku (constant IMAGETYPE_XXX)

Návratová hodnota:

boolean – true pokud se obrázek podařilo uložit

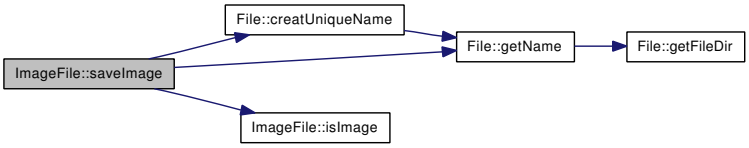
Definice je uvedena na řádku 188 v souboru imagefile.class.php.

Odkazuje se na File::creatUniqueName(), File::getName() a isImage().

Používá se v saveJpegImage() a savePngImage().

```
188
189         $saved = false;
190         $tmpImage = $this->createTempImage();
191
192         if($width == null){
193            $width = $this->imageWidth;
194         }
195         if($heigh == null){
196            $heigh = $this->imageHeight;
197         }
198         if($newName == null){
199            $newName = $this->getName();
200         }
201
202         //    test jestli je zpracováván obrázek
203         if($this->isImage()){
204            //       Test názvu souboru
205            $newName = $this->creatUniqueName($dstDir);
206            $newImage = $this->resampleImage($tmpImage,$width,$heigh);
207
208            if($imageType == null){
209               $imageType = $this->imageType;
210            }
211
212            $saved = $this->saveNewImage($newImage, $imageType, $dstDir, $newName);
213            imagedestroy($newImage);
214         }
215         return $saved;
216     }
```

Tato funkce volá...



8.43.3.7 ImageFile::saveJpegImage ($ *dstDir*, $ *width* = `null`, $ *heigh* = `null`, $ *newName* = `null`)

Metoda uloží JPEG obrázek do zadané cesty.

Parametry:

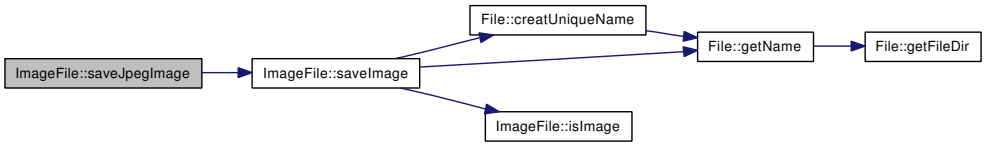
string – cílový adresář, kam se obrázek ukládá
int – šířka výsledného obrázku
int – výška výsledného obrázku
string – nový název obrázku

Definice je uvedena na řádku 226 v souboru imagefile.class.php.

Odkazuje se na saveImage().

```
226                                                                                                    {
227       $this->saveImage($dstDir, $width, $heigh, $newName, IMAGETYPE_JPEG);
228     }
```

Tato funkce volá...



8.43.3.8 ImageFile::savePngImage ($ *dstDir*, $ *width* = `null`, $ *heigh* = `null`, $ *newName* = `null`)

Metoda uloží PNG obrázek do zadané cesty.

Parametry:

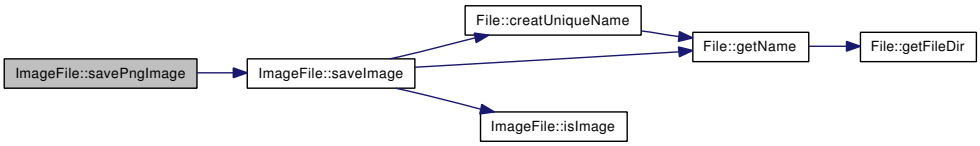
string – cílový adresář, kam se obrázek ukládá
int – šířka výsledného obrázku
int – výška výsledného obrázku
string – nový název obrázku

Definice je uvedena na řádku 238 v souboru imagefile.class.php.

Odkazuje se na saveImage().

```
238                                                                                                                 {
239         $this->saveImage($dstDir, $width, $heigh, $newName, IMAGETYPE_GIF);
240     }
```

Tato funkce volá...



8.43.3.9 ImageFile::createImage ($ *srcFile*, $ *destFile*, $ *width*, $ *height*, $ *cropSize* = `false`, $ *jpegQuality* = `85`, $ *pngQuality* = `9`)

Funkce vytvoří obrázek ze zadaného obrázku a uloží jej do specifikovaného souboru.

Parametry:

string – zdrojový soubor ($_FILE['file']['tmp_file'])

string – cesta a cílový soubor

integer – max. šířka výsledného obrázku

integer – max. výška výsledného obrázku

boolean – true pro ořezání obrázku na zadané velikosti, false pouze pro změnu velikosti

integer – výsledná kvalita jpg komprese (default 85)

integer – výsledná kvalit png komprese (default 9

Návratová hodnota:

boolean – true, jestliže byl obrázek uspěšně vytvořen a ulože

Definice je uvedena na řádku 390 v souboru imagefile.class.php.

```
390
391         //    Výpočet nové velikosti
392         $imageProperty = getimagesize($srcFile);
393         if($newImageType != false){
394            if($cropSize == false){
395               //          obrázek je na šířku
396               if($imageProperty[0] > $imageProperty[1]){
397                  $imageRate = $imageProperty[1]/$imageProperty[0];
398                  $height = $imageRate*$width;
399               }
400               //          obrázek je na výšku
401               else {
402                  $imageRate = $imageProperty[0]/$imageProperty[1];
403                  $width = $imageRate*$height;
404               }
405
406               $newImage = imagecreatetruecolor($width, $height);
407               imagecopyresampled($newImage, $tempImage, 0,0,0,0, $width, $height, $imageProperty[0],
408            } else {
409               //       Ořezání obrázku do jedné velikosti
410               $newImage = imagecreatetruecolor($width, $height);
```

```
411
412                $scale = (($width / $imageProperty[0]) > ($height / $imageProperty[1])) ? ($width / $im
413                $newW = $width/$scale;    // jak by mel byt zdroj velky (pro poradek :)
414                $newH = $height/$scale;
415
416                // ktera strana precuhuje vic (kvuli chybe v zaokrouhleni)
417                if (($imageProperty[0] - $newW) > ($imageProperty[1] - $newH)) {
418                   $imageX = floor(($imageProperty[0] - $newW)/2);
419                   $imageY = 0;
420                   $imageWidth = floor($newW);
421                   $imageHeight = $imageProperty[1];
422
423                   //             $src = array(floor(($imageProperty[0] - $newW)/2), 0, floor($newW), $
424                }
425                else {
426                   $imageX = 0;
427                   $imageY = floor(($imageProperty[1] - $newH)/2);
428                   $imageWidth = $imageProperty[0];
429                   $imageHeight = floor($newH);
430
431                   //             $src = array(0, floor(($imageProperty[1] - $newH)/2), $imageProperty[
432                }
433                ImageCopyResampled($newImage, $tempImage, 0,0, $imageX, $imageY, $width, $height, $imag
434             }
435
436             //       uložení obrázku
437             if ($newImageType == IMAGETYPE_JPEG){
438                $newImageFunction($newImage, $destFile, $jpegQuality);
439             } else if($newImageType == IMAGETYPE_JPEG){
440                $newImageFunction($newImage, $destFile, $pngQuality);
441             } else {
442                $newImageFunction($newImage, $destFile);
443             }
444             ImageDestroy($newImage);
445             ImageDestroy($tempImage);
446
447             return true;
448          }
449          return false;
450       }
```

8.43.3.10 ImageFile::getOriginalWidth ()

Metoda vrací rozměr původního obrázku - Šířku.

Návratová hodnota:

integer – šířka obrázku

Definice je uvedena na řádku 457 v souboru imagefile.class.php.

```
457                                        {
458          return $this->imageWidth;
459       }
```

8.43.3.11 ImageFile::getOriginalHeight ()

Metoda vrací rozměr původního obrázku - Výšku.

Návratová hodnota:

integer – výška obrázku

Definice je uvedena na řádku 466 v souboru imagefile.class.php.

```
466                                         {
467           return $this->imageHeight;
468       }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Filesystem/imagefile.class.php

8.44 Dokumentace třídy Images

Třída pro práci s obrázky Třída pro základní práci s obrázky.

Veřejné metody

- __construct (Messages $errors, $file, $reportErrors=true)
  Konstruktor třídy.
- isImage ()
  Matedoa zjišťuje, jestli je daný soubor obrázek.
- setDimensions ($width, $height)
  metoda nastaví rozměry obrázku
- setCrop ($crop)
  Metoda zapíná/vypíná ořezání obrázku.
- setImageName ($name)
  Metoda nastaví jméno nového obrázku.
- getNewImageName ()
  Metoda vrací název uloženého obrázku.
- saveImage ($dstDir, $width=null, $heigh=null, $newName=null, $imageType=null)
  Metoda uloží obrázek ve stejném formátu, v jakém byl zadán.
- saveJpegImage ($dstDir, $width=null, $heigh=null, $newName=null)
  Metoda uloží JPEG obrázek do zadané cesty.
- savePngImage ($dstDir, $width=null, $heigh=null, $newName=null)
  Metoda uloží PNG obrázek do zadané cesty.
- createImage ($srcFile, $destFile, $width, $height, $cropSize=false, $jpegQuality=85, $pngQuality=9)
  Funkce vytvoří obrázek ze zadaného obrázku a uloží jej do specifikovaného souboru.
- getOriginalWidth ()
  Metoda vrací rozměr původního obrázku - Šířku.
- getOriginalHeight ()
  Metoda vrací rozměr původního obrázku - Výšku.

8.44.1 Detailní popis

Třída pro práci s obrázky Třída pro základní práci s obrázky.

Umožňuje jejich ukládání, ořezávání, změnu velikost a změnu formátu obrázku.

Verze:

$Id: images.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s obrázky

Zastaralé

– odstranit, protože se používá třída ImageFile která pracuje i se souborem

Definice je uvedena na řádku 16 v souboru images.class.php.

8.44.2 Dokumentace konstruktoru a destruktoru

8.44.2.1 Images::__construct (Messages $ *errors*, $ *file*, $ *reportErrors* = `true`)

Konstruktor třídy.

Parametry:

string – název souboru

Definice je uvedena na řádku 100 v souboru images.class.php.

```
100                                                                    {
101         $this->errors = $errors;
102         $this->imageFile = $file;
103         $this->reportErrors = $reportErrors;
104
105         $this->checkIsImage();
106     }
```

8.44.3 Dokumentace k metodám

8.44.3.1 Images::isImage ()

Matedoa zjišťuje, jestli je daný soubor obrázek.

Návratová hodnota:

boolean – true pokud se jedná o obrázek se kterým umí pracovat

Definice je uvedena na řádku 113 v souboru images.class.php.

Používá se v saveImage().

```
113                                 {
114         return $this->isImage;
115     }
```

8.44.3.2 Images::setDimensions ($ *width*, $ *height*)

metoda nastaví rozměry obrázku

Parametry:

integer – šířka v px

integer – výška v px

Definice je uvedena na řádku 160 v souboru images.class.php.

```
160                                                 {
161         $this->newImageWidth = $width;
162         $this->newImageHeight = $height;
163     }
```

8.44.3.3 Images::setCrop ($ *crop*)

Metoda zapíná/vypíná ořezání obrázku.

Parametry:

boolean – true pro zapnutí ořezání

Definice je uvedena na řádku 169 v souboru images.class.php.

```
169                                 {
170         $this->cropNewImage = $crop;
171     }
```

8.44.3.4 Images::setImageName ($ *name*)

Metoda nastaví jméno nového obrázku.

Parametry:

string – název nového obrázku

Definice je uvedena na řádku 179 v souboru images.class.php.

```
179                                     {
180         $this->newImageName = $name;
181     }
```

8.44.3.5 Images::getNewImageName ()

Metoda vrací název uloženého obrázku.

Návratová hodnota:

string – název obrázku

Definice je uvedena na řádku 188 v souboru images.class.php.

```
188                                     {
189         return $this->newImageName;
190     }
```

8.44.3.6 Images::saveImage ($ *dstDir*, $ *width* = null, $ *heigh* = null, $ *newName* = null, $ *imageType* = null)

Metoda uloží obrázek ve stejném formátu, v jakém byl zadán.

Parametry:

string – cílový adresář, kam se obrázek ukládá

int – šířka výsledného obrázku

int – výška výsledného obrázku

string – nový název obrázku

int – typ výsledného obrázku (constant IMAGETYPE_XXX)

Návratová hodnota:

boolean – true pokud se obrázek podařilo uložit

Definice je uvedena na řádku 204 v souboru images.class.php.

Odkazuje se na isImage().

Používá se v saveJpegImage() a savePngImage().

```
204
205         $saved = false;
206         $tmpImage = $this->createTempImage();
207
208         if($width == null){
209            $width = $this->imageWidth;
210         }
211         if($heigh == null){
212            $heigh = $this->imageHeight;
213         }
214         if($newName == null){
215            $newName = $this->newImageName;
216         }
217
218 //      test jestli je zpracováván obrázek
219         if($this->isImage()){
220 //         Test názvu souboru
221            $newName = $this->checkNewImageName($dstDir, $newName);
222
223            $newImage = $this->resampleImage($tmpImage,$width,$heigh);
224
225            if($imageType == null){
226               $imageType = $this->imageType;
227            }
228
229 //         Kontrola cílového adresáře, jestli obsahuje lomítko
230            $files = new Files();
231            $dstDir = $files->checkDirPath($dstDir);
232            unset($files);
233
```

```
234           $saved = $this->saveNewImage($newImage, $imageType, $dstDir.$newName);
235           imagedestroy($newImage);
236       }
237       return $saved;
238   }
```

Tato funkce volá...

8.44.3.7 Images::saveJpegImage ($ *dstDir*, $ *width* = null, $ *heigh* = null, $ *newName* = null)

Metoda uloží JPEG obrázek do zadané cesty.

Parametry:

string – cílový adresář, kam se obrázek ukládá

int – šířka výsledného obrázku

int – výška výsledného obrázku

string – nový název obrázku

Definice je uvedena na řádku 248 v souboru images.class.php.

Odkazuje se na saveImage().

```
248                                                                                                    {
249       $this->saveImage($dstDir, $width, $heigh, $newName, IMAGETYPE_JPEG);
250   }
```

Tato funkce volá...

8.44.3.8 Images::savePngImage ($ *dstDir*, $ *width* = null, $ *heigh* = null, $ *newName* = null)

Metoda uloží PNG obrázek do zadané cesty.

Parametry:

string – cílový adresář, kam se obrázek ukládá

int – šířka výsledného obrázku

int – výška výsledného obrázku

string – nový název obrázku

Definice je uvedena na řádku 260 v souboru images.class.php.

Odkazuje se na saveImage().

```
260                                                                                                                 {
261         $this->saveImage($dstDir, $width, $heigh, $newName, IMAGETYPE_GIF);
262     }
```

Tato funkce volá...

8.44.3.9 Images::createImage ($ *srcFile*, $ *destFile*, $ *width*, $ *height*, $ *cropSize* = `false`, $ *jpegQuality* = `85`, $ *pngQuality* = `9`)

Funkce vytvoří obrázek ze zadaného obrázku a uloží jej do specifikovaného souboru.

Parametry:

string – zdrojový soubor ($_FILE['file']['tmp_file'])

string – cesta a cílový soubor

integer – max. šířka výsledného obrázku

integer – max. výška výsledného obrázku

boolean – true pro ořezání obrázku na zadané velikosti, false pouze pro změnu velikosti

integer – výsledná kvalita jpg komprese (default 85)

integer – výsledná kvalit png komprese (default 9

Návratová hodnota:

boolean – true, jestliže byl obrázek uspěšně vytvořen a ulože

Definice je uvedena na řádku 406 v souboru images.class.php.

```
406
407
408
409         //    Výpočet nové velikosti
410         $imageProperty = getimagesize($srcFile);
411         if($newImageType != false){
412            if($cropSize == false){
413               //          obrázek je na šířku
414               if($imageProperty[0] > $imageProperty[1]){
415                  $imageRate = $imageProperty[1]/$imageProperty[0];
416                  $height = $imageRate*$width;
417               }
418               //          obrázek je na výšku
419               else {
420                  $imageRate = $imageProperty[0]/$imageProperty[1];
421                  $width = $imageRate*$height;
422               }
423
424               $newImage = imagecreatetruecolor($width, $height);
425               imagecopyresampled($newImage, $tempImage, 0,0,0,0, $width, $height, $imageProperty[0],
426            } else {
427               //       Ořezání obrázku do jedné velikosti
428               $newImage = imagecreatetruecolor($width, $height);
429
430               $scale = (($width / $imageProperty[0]) > ($height / $imageProperty[1])) ? ($width / $im
431               $newW = $width/$scale;    // jak by mel byt zdroj velky (pro poradek :)
```

```
432                $newH = $height/$scale;
433
434                // ktera strana precuhuje vic (kvuli chybe v zaokrouhleni)
435                if (($imageProperty[0] - $newW) > ($imageProperty[1] - $newH)) {
436                   $imageX = floor(($imageProperty[0] - $newW)/2);
437                   $imageY = 0;
438                   $imageWidth = floor($newW);
439                   $imageHeight = $imageProperty[1];
440
441                   //             $src = array(floor(($imageProperty[0] - $newW)/2), 0, floor($newW), $
442                }
443                else {
444                   $imageX = 0;
445                   $imageY = floor(($imageProperty[1] - $newH)/2);
446                   $imageWidth = $imageProperty[0];
447                   $imageHeight = floor($newH);
448
449                   //             $src = array(0, floor(($imageProperty[1] - $newH)/2), $imageProperty[
450                }
451                ImageCopyResampled($newImage, $tempImage, 0,0, $imageX, $imageY, $width, $height, $imag
452             }
453
454             //       uložení obrázku
455             if ($newImageType == IMAGETYPE_JPEG){
456                $newImageFunction($newImage, $destFile, $jpegQuality);
457             } else if($newImageType == IMAGETYPE_JPEG){
458                $newImageFunction($newImage, $destFile, $pngQuality);
459             } else {
460                $newImageFunction($newImage, $destFile);
461             }
462             ImageDestroy($newImage);
463             ImageDestroy($tempImage);
464
465             return true;
466          }
467       return false;
468    }
```

8.44.3.10 Images::getOriginalWidth ()

Metoda vrací rozměr původního obrázku - Šířku.

Návratová hodnota:

integer – šířka obrázku

Definice je uvedena na řádku 475 v souboru images.class.php.

```
475                                      {
476       return $this->imageWidth;
477    }
```

8.44.3.11 Images::getOriginalHeight ()

Metoda vrací rozměr původního obrázku - Výšku.

Návratová hodnota:

integer – výška obrázku

Definice je uvedena na řádku 484 v souboru images.class.php.

```
484                                     {
485         return $this->imageHeight;
486     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/images.class.php

8.45 Dokumentace třídy JQuery

Třída JsPluginu TabContent.

Diagram dědičnosti pro třídu JQuery

Diagram tříd pro JQuery:

Veřejné metody

- addUICore ()

  Metody pro přidávání částí jQuery UI, effects.

- addUIDraggable ()

  Metoda přidá efekty UI - draggable (přesunování).

- addUIDroppable ()

  Metoda přidá efekty UI - droppable (odstraňování).

- addUIResizable ()

  Metoda přidá efekty UI - resizable (změna velikosti).

- addUISelectable ()

Metoda přidá efekty UI - selectable (označování).

- addUISortable ()

  Metoda přidá efekty UI - sortable (řazení).

- addWidgentAccordion ()

  Metoda přidá widgent UI - accordion (roztahování boxů).

- addWidgentDialog ()

  Metoda přidá widgent UI - dialog (box dialogu).

- addWidgentSlider ()

  Metoda přidá widgent UI - slider (posunovač).

- addWidgentTabs ()

  Metoda přidá widgent UI - tabs (tabulka boxů - záložky).

- addWidgentDatepicker ()

  Metoda přidá widgent UI - datepicker (box s výběrem data).

- addWidgentProgressBar ()

  Metoda přidá widgent UI - progressbar (pregress bar).

- addEffectCore ()

  Metoda přidá efekt UI - core (jádro efektů).

- addEffectAll ()

  Metoda přidá efekty UI - all (všechny efekty).

Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

8.45.1 Detailní popis

Třída JsPluginu TabContent.

Třída slouží pro práci se záložkovým menu (tj. boxy se záložkovým přepínáním obsahu). Je úzce zpata z šablonou. //TODO dodělat tvorbu scriptu pro spuštění, tak aby se dal vložit přímo do šablony a ghenerován byl zde.

Verze:

Id

form.class.php 434 2008-12-30 00:35:31Z jakub

VVE3.3.0

Revision

434

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída JsPluginu pro záložkový box

Definice je uvedena na řádku 15 v souboru jquery.class.php.

8.45.2 Dokumentace k metodám

8.45.2.1 JQuery::addUICore ()

Metody pro přidávání částí jQuery UI, effects.

Metoda přidá jádro pro efekty UI

Definice je uvedena na řádku 40 v souboru jquery.class.php.

Odkazuje se na JsPlugin::addJsFile().

Používá se v addUIDraggable(), addUIDroppable(), addUIResizable(), addUISelectable(), addUISortable(), addWidgentAccordion(), addWidgentDialog(), addWidgentProgressBar(), addWidgentSlider() a addWidgentTabs().

```
40                                  {
41      $this->addJsFile(new JsPluginJsFile("jquery-ui-core.packed.js"));
42      return $this;
43    }
```

Tato funkce volá...

8.45.2.2 JQuery::addWidgentDatepicker ()

Metoda přidá widgent UI - datepicker (box s výběrem data).

Plánované úpravy

zkontrolovat závislos s obrázky umístěnými ve složce pluginu

Definice je uvedena na řádku 155 v souboru jquery.class.php.

Odkazuje se na JsPlugin::addJsFile().

```
155                                         {
156     //deps
160     $this->addJsFile(new JsPluginJsFile("jquery-ui-datepicker.packed.js"));
161     return $this;
162   }
```

Tato funkce volá...

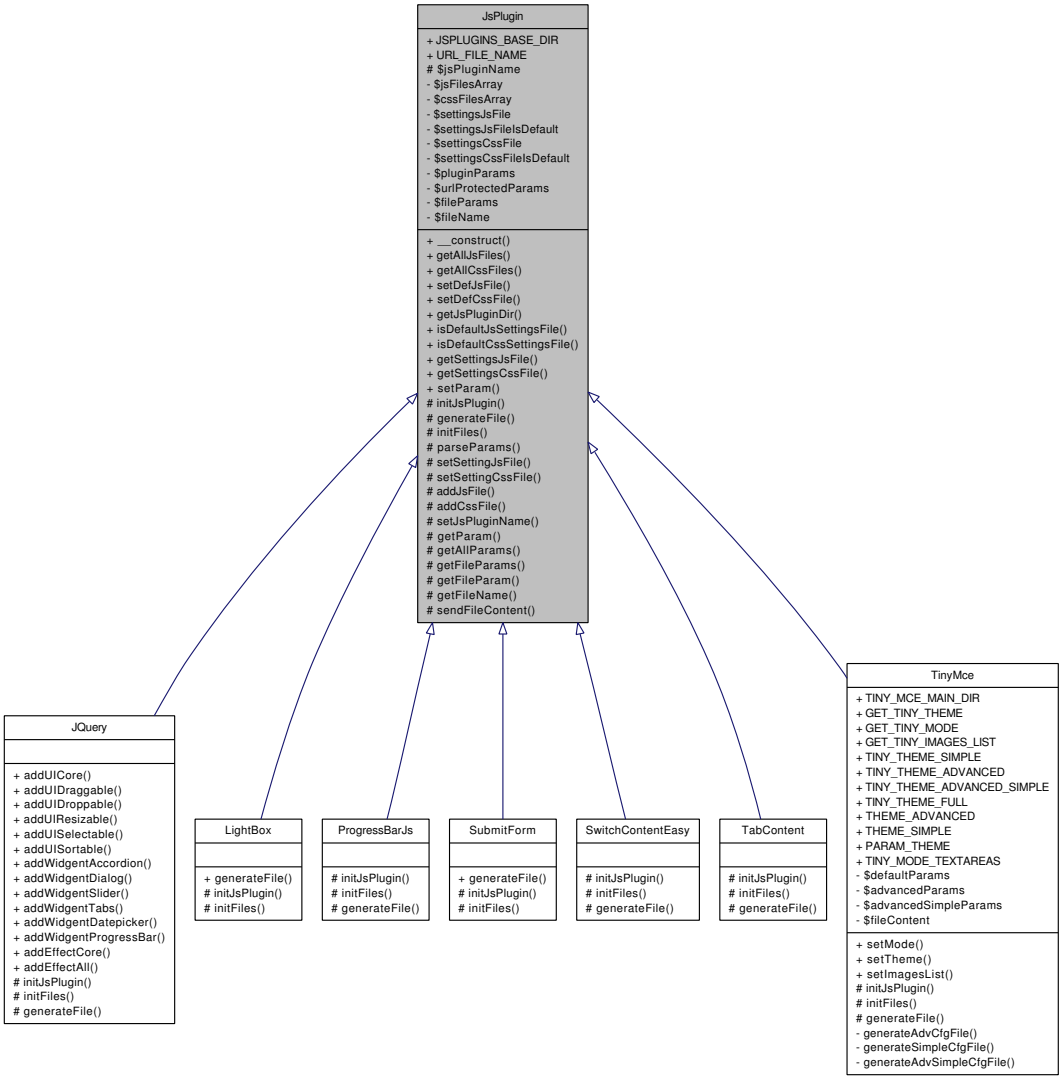
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/jquery.class.php

8.46 Dokumentace třídy JsPlugin

Abstraktní třída pro obsluh JavaScript Pluginů JsPlugins.

Diagram dědičnosti pro třídu JsPlugin



Veřejné metody

- __construct ()

  Konstruktor třídy.

- getAllJsFiles ()

  Metoda vrací pole všech Js souborů pluginu.

- getAllCssFiles ()

  Metoda vrací pole všech Css souborů pluginu.

- setDefJsFile (JsPluginJsFile $jsFile)

  Metoda nastaví nový výchozí js soubor nastavení.

- setDefCssFile ($cssFile)

  Metoda nastaví nový výchozí css soubor nastavení.

- getJsPluginDir ()

  Metoda vrací název adresáře pluginu.

- isDefaultJsSettingsFile ()

  Metoda vrací true pokud se použije výchozí js soubr s nastavením JsPluginu.

- isDefaultCssSettingsFile ()

  Metoda vrací true pokud se použije výchozí css soubr s nastavením JsPluginu.

- getSettingsJsFile ()

  Metoda vrací název výchozího js souboru s nastavením.

- getSettingsCssFile ()

  Metoda vrací název výchozího css souboru s nastavením.

- setParam ($paramName, $paramValue)

  Metoda nastaví zadaný parametr pluginu.

Veřejné atributy

- const **JSPLUGINS_BASE_DIR** = 'jscripts'
- const **URL_FILE_NAME** = 'file'

Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- generateFile ()

  Metoda se využívá pro načtení proměných do stránky, je volána při volání parametru stránky pro JsPlugin a je pouze zpracována tato metoda (generování nastavení atd).

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- parseParams ()

  Metoda parsuje parametry pro generování souboru.

- setSettingJsFile (JsPluginJsFile $jsFile)

  Metoda nastavuje js soubor, který se načte místo výchozího.

- setSettingCssFile ($cssFile)

Metoda nastavuje css soubor, který se načte místo výchozího.

- addJsFile (JsPluginJsFile $jsFile)
  Metoda přídává js soubor se scriptem.
- addCssFile ($cssFile)
  Metoda přídává css soubor js pluginu.
- setJsPluginName ($pluginName)
  Metoda nastaví název js pluginu – je totžný s názvem adresáře (case insensitive).
- getParam ($paramName)
  Metoda vrací zadaný parametr pluginu.
- getAllParams ()
  Metoda vrací všechny parametry parametr.
- getFileParams ()
  Metoda vrací předané parametry v url souboru.
- getFileParam ($paramName)
  Metoda vrací předaný parametr v url souboru.
- getFileName ()
  Metoda vrací název zpracovávaného souboru.
- sendFileContent ($content)
  metoda odešla obsah na výstup a ukončí script

Chráněné atributy

- **$jsPluginName** = null

8.46.1 Detailní popis

Abstraktní třída pro obsluh JavaScript Pluginů JsPlugins.

Třída slouží jako základ pro tvorbu JsPluginů a jejich implementaci. Poskytuje základní přístup k parametrům JsPlugin. Umožňuje také přímé generování souborů pro tvorbu dynamických nastavení a obsahů. Vše je ovládáno přes poheld (viewer).

Copyright (c) 2008 Jakub Matas

Verze:

$Id: jsplugins.class.php 3.1.8 beta1 13.11.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro práci s JsPluginy

Definice je uvedena na řádku 14 v souboru jsplugin.class.php.

8.46.2 Dokumentace k metodám

8.46.2.1 JsPlugin::setSettingJsFile (JsPluginJsFile $ *jsFile*) [final, protected]

Metoda nastavuje js soubor, který se načte místo výchozího.

Parametry:

JsPluginJsFile – název js souboru (je umístěn v adresáři modulu)

Definice je uvedena na řádku 147 v souboru jsplugin.class.php.

Používá se v TinyMce::initFiles() a setDefJsFile().

```
147                                                                    {
148         $this->settingsJsFile = $jsFile;
149     }
```

8.46.2.2 JsPlugin::setSettingCssFile ($ *cssFile*) [final, protected]

Metoda nastavuje css soubor, který se načte místo výchozího.

Parametry:

string – název css souboru (je umístěn v adresáři modulu)

Definice je uvedena na řádku 155 v souboru jsplugin.class.php.

Používá se v setDefCssFile().

```
155                                                          {
156         $this->settingsCssFile = $cssFile;
157     }
```

8.46.2.3 JsPlugin::addJsFile (JsPluginJsFile $ *jsFile*) [final, protected]

Metoda přídává js soubor se scriptem.

Parametry:

JsPluginJsFile – název js souboru

Definice je uvedena na řádku 163 v souboru jsplugin.class.php.

Používá se v JQuery::addEffectAll(), JQuery::addEffectCore(), JQuery::addUICore(), JQuery::addUIDraggable(), JQuery::addUIDroppable(), JQuery::addUIResizable(), JQuery::addUISelectable(), JQuery::addUISortable(), JQuery::addWidgentAccordion(), JQuery::addWidgentDatepicker(), JQuery::addWidgentDialog(), JQuery::addWidgentProgressBar(), JQuery::addWidgentSlider(), JQuery::addWidgentTabs(), TinyMce::initFiles(), TabContent::initFiles(), SwitchContentEasy::initFiles(), SubmitForm::initFiles(), ProgressBarJs::initFiles(), LightBox::initFiles(), JQuery::initFiles() a SwitchContentEasy::initJsPlugin().

```
164     {
165         array_push($this->jsFilesArray, $jsFile);
166     }
```

8.46.2.4 JsPlugin::addCssFile ($ *cssFile*) [final, protected]

Metoda přídává css soubor js pluginu.

Parametry:

string – název css souboru

Definice je uvedena na řádku 172 v souboru jsplugin.class.php.

Používá se v TabContent::initFiles() a LightBox::initFiles().

```
173    {
174       array_push($this->cssFilesArray, $cssFile);
175    }
```

8.46.2.5 JsPlugin::getAllJsFiles () [final]

Metoda vrací pole všech Js souborů pluginu.

Návratová hodnota:

array – pole js souborů

Definice je uvedena na řádku 181 v souboru jsplugin.class.php.

Používá se v Template::addJsPlugin().

```
182    {
183
184       return $this->jsFilesArray;
185    }
```

8.46.2.6 JsPlugin::getAllCssFiles () [final]

Metoda vrací pole všech Css souborů pluginu.

Návratová hodnota:

array – pole Css souborů

Definice je uvedena na řádku 191 v souboru jsplugin.class.php.

Používá se v Template::addJsPlugin().

```
192    {
193       return $this->cssFilesArray;
194    }
```

8.46.2.7 JsPlugin::setDefJsFile (JsPluginJsFile $ *jsFile*) [final]

Metoda nastaví nový výchozí js soubor nastavení.

Parametry:

JsPluginJsFile – název js souboru

Definice je uvedena na řádku 200 v souboru jsplugin.class.php.

Odkazuje se na setSettingJsFile().

```
201     {
202        $this->settingsJsFileIsDefault = false;
203        $this->setSettingJsFile($jsFile);
204     }
```

Tato funkce volá...

8.46.2.8 JsPlugin::setDefCssFile ($ *cssFile*) [final]

Metoda nastaví nový výchozí css soubor nastavení.

Parametry:

string – název css souboru

Definice je uvedena na řádku 210 v souboru jsplugin.class.php.

Odkazuje se na setSettingCssFile().

```
211     {
212        $this->settingsCssFileIsDefault = false;
213        $this->setSettingCssFile($cssFile);
214     }
```

Tato funkce volá...

8.46.2.9 JsPlugin::setJsPluginName ($ *pluginName*) [final, protected]

Metoda nastaví název js pluginu – je totžný s názvem adresáře (case insensitive).

Parametry:

string – název JsPluginu

Definice je uvedena na řádku 220 v souboru jsplugin.class.php.

Používá se v TinyMce::initJsPlugin(), TabContent::initJsPlugin(), SwitchContentEasy::initJsPlugin(), SubmitForm::initJsPlugin(), ProgressBarJs::initJsPlugin(), LightBox::initJsPlugin() a JQuery::initJsPlugin().

```
221    {
222       $this->jsPluginName = $pluginName;
223    }
```

8.46.2.10 JsPlugin::getJsPluginDir () [final]

Metoda vrací název adresáře pluginu.

Návratová hodnota:

string – název adresáře

Definice je uvedena na řádku 229 v souboru jsplugin.class.php.

```
230    {
231       $dir = '.'.ModuleDirs::DIR_SEPARATOR.self::JSPLUGINS_BASE_DIR.ModuleDirs::DIR_SEPARATOR
232             .strtolower($this->jsPluginName).ModuleDirs::DIR_SEPARATOR;
233
234       return $dir;
235    }
```

8.46.2.11 JsPlugin::isDefaultJsSettingsFile () [final]

Metoda vrací true pokud se použije výchozí js soubr s nastavením JsPluginu.

Návratová hodnota:

boolean – true pro výchozí nastavení

Definice je uvedena na řádku 241 v souboru jsplugin.class.php.

```
242    {
243       return $this->settingsJsFileIsDefault;
244    }
```

8.46.2.12 JsPlugin::isDefaultCssSettingsFile () [final]

Metoda vrací true pokud se použije výchozí css soubr s nastavením JsPluginu.

Návratová hodnota:

boolean – true pro výchozí nastavení

Definice je uvedena na řádku 250 v souboru jsplugin.class.php.

```
251    {
252       return $this->settingsCssFileIsDefault;
253    }
```

8.46.2.13 JsPlugin::getSettingsJsFile () [final]

Metoda vrací název výchozího js souboru s nastavením.

Návratová hodnota:

JsPluginJsFile – název js souboru

Definice je uvedena na řádku 259 v souboru jsplugin.class.php.

```
260     {
261        $file = $this->settingsJsFile;
262
263        if(!empty($this->pluginParams)){
264           $params = http_build_query($this->pluginParams);
265           $file.='?'.$params;
266        }
267
268        return $file;
269 //     return $this->settingsJsFile.;
270     }
```

8.46.2.14 JsPlugin::getSettingsCssFile () [final]

Metoda vrací název výchozího css souboru s nastavením.

Návratová hodnota:

string – název css souboru

Definice je uvedena na řádku 276 v souboru jsplugin.class.php.

```
277     {
278        return $this->settingsCssFile;
279     }
```

8.46.2.15 JsPlugin::setParam ($ *paramName*, $ *paramValue*) [final]

Metoda nastaví zadaný parametr pluginu.

Parametry:

string – název parametru

mixed – hodnota paramettru

Definice je uvedena na řádku 287 v souboru jsplugin.class.php.

Používá se v TinyMce::initJsPlugin(), TinyMce::setImagesList(), TinyMce::setMode() a TinyMce::setTheme().

```
287                                                     {
288        $this->pluginParams[$paramName] = $paramValue;
289     }
```

8.46.2.16 JsPlugin::getParam ($ *paramName*) [final, protected]

Metoda vrací zadaný parametr pluginu.

Parametry:

string – název parametru

Návratová hodnota:

mixed – hodnota parametru

Definice je uvedena na řádku 297 v souboru jsplugin.class.php.

```
297                                                 {
298         if(isset($this->pluginParams[$paramName])){
299            return $this->pluginParams[$paramName];
300         } else {
301            return null;
302         }
303     }
```

8.46.2.17 JsPlugin::getAllParams () [final, protected]

Metoda vrací všechny parametry parametr.

Návratová hodnota:

array – pole s parametry

Definice je uvedena na řádku 310 v souboru jsplugin.class.php.

```
310                                             {
311         return $this->pluginParams;
312     }
```

8.46.2.18 JsPlugin::getFileParams () [final, protected]

Metoda vrací předané parametry v url souboru.

Návratová hodnota:

array

Definice je uvedena na řádku 318 v souboru jsplugin.class.php.

```
318                                             {
319         return $this->fileParams;
320     }
```

8.46.2.19 JsPlugin::getFileParam ($ *paramName*) [final, protected]

Metoda vrací předaný parametr v url souboru.

Parametry:

string – název parametru

Návratová hodnota:

mixed – hodnota parametru

Definice je uvedena na řádku 327 v souboru jsplugin.class.php.

```
327                                                 {
328         if(isset($this->fileParams[$paramName])){
329             return $this->fileParams[$paramName];
330         } else {
331             return null;
332         }
333     }
```

8.46.2.20 JsPlugin::getFileName () [final, protected]

Metoda vrací název zpracovávaného souboru.

Návratová hodnota:

string – název souboru

Definice je uvedena na řádku 339 v souboru jsplugin.class.php.

Používá se v TinyMce::generateFile().

```
339                                         {
340         return $this->fileName;
341     }
```

8.46.2.21 JsPlugin::sendFileContent ($ *content*) [final, protected]

metoda odešla obsah na výstup a ukončí script

Parametry:

string – obsah souboru

Definice je uvedena na řádku 347 v souboru jsplugin.class.php.

```
347                                                 {
348         header("Content-Length: " . strlen($content));
349         header("Content-type: application/x-javascript");
350         echo $content;
351         exit();
352     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/jsplugin.class.php

8.47 Dokumentace třídy JsPluginJsFile

Pomocná třída JsPluginu.

Veřejné metody

- __construct ($file)

  Konstruktor.

- setParam ($paramName, $paramValue)

  Metoda nastaví parametr souboru.

- getParams ()

  Metoda vrací všechny parametry jako pole.

- __toString ()

  Metoda převede soubor na řetězec a přidá za něj parametry.

8.47.1 Detailní popis

Pomocná třída JsPluginu.

Třída slouží pro práci s javascript soubory v JsPluginu. Umožňuje jednoduché nasatvené parametrů souboru.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: jspluginjsfile.class.php 3.1.8 beta1 13.11.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třida pro práci s javascript soubry JsPluginu

Definice je uvedena na řádku 13 v souboru jspluginjsfile.class.php.

8.47.2 Dokumentace konstruktoru a destruktoru

8.47.2.1 JsPluginJsFile::__construct ($*file*)

Konstruktor.

Parametry:

string – název souboru

Definice je uvedena na řádku 33 v souboru jspluginjsfile.class.php.

```
33                                  {
34          $this->jsFile = $file;
35      }
```

8.47.3 Dokumentace k metodám

8.47.3.1 JsPluginJsFile::setParam ($ *paramName*, $ *paramValue*)

Metoda nastaví parametr souboru.

Parametry:

string – název parametru
mixed – hodnota parametru

Definice je uvedena na řádku 43 v souboru jspluginjsfile.class.php.

```
43                                                          {
44        $this->fileParams[$paramName] = $paramValue;
45    }
```

8.47.3.2 JsPluginJsFile::getParams ()

Metoda vrací všechny parametry jako pole.

Návratová hodnota:

array – pole parametrů

Definice je uvedena na řádku 51 v souboru jspluginjsfile.class.php.

```
51                                  {
52        return $this->fileParams;
53    }
```

8.47.3.3 JsPluginJsFile::__toString ()

Metoda převede soubor na řetězec a přidá za něj parametry.

Návratová hodnota:

string – soubor s parametry

Definice je uvedena na řádku 60 v souboru jspluginjsfile.class.php.

```
60                            {
61        $file = $this->jsFile;
62
63        if(!empty($this->fileParams)){
64           $params = http_build_query($this->fileParams);
65           $file.='?'.$params;
66        }
67
68        return $file;
69    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/jspluginjsfile.class.php

8.48 Dokumentace třídy LightBox

Třída JsPluginu LightBOX.

Diagram dědičnosti pro třídu LightBox

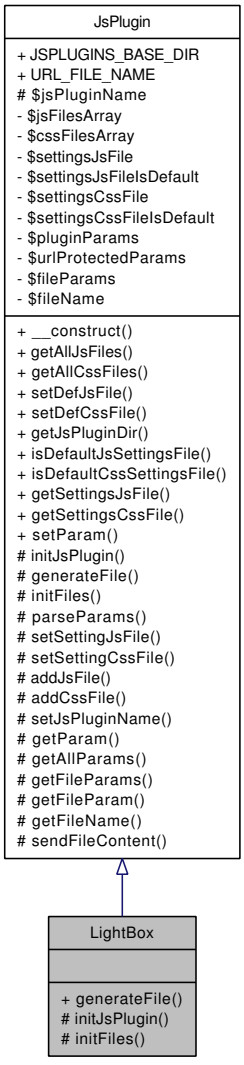
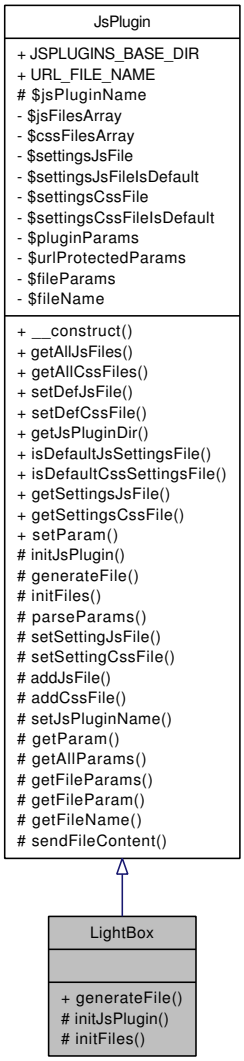


Diagram tříd pro LightBox:



Veřejné metody

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

8.48.1 Detailní popis

Třída JsPluginu LightBOX.

Třída pro vkládání pluginu pro zobrazovaní popup oken s obrázkem.

Verze:

$Id: lightbox.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu LightBox

Viz také:

http://www.dynamicdrive.com/dynamicindex4/lightbox2/index.htm

Definice je uvedena na řádku 13 v souboru lightbox.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/lightbox.class.php

8.49 Dokumentace třídy Links

Třída pro práci s odkazy.

Veřejné metody

- __construct ($clear=false, $onlyWebRoot=false)

  Konstruktor nastaví základní adresy a přenosový protokol.

- category ($catName=null, $catId=null)

  Metoda nastavuje název a id kategorie.

- route ($name=null, $id=null)

  Metoda nastavuje název a id routy.

- article ($article=null, $idArticle=null)

  Metoda nastavuje název article.

- media ($media=null)

  Metoda nastavuje typ media.

- file ($file=null)

  Metoda nastavuje parametr nazev epluginu.

- action ($action=null)

  Metoda nastavuje název action.

- lang ($lang=null)

  Metoda nastavuje název lokalizace.

- param ($name, $value=null)

  Metoda nastavuje prametry do url (je li jich více).

- rmParam ($name=null)

  Metoda odstraní daný parametr z url.

- reload ($link=null)

  Metoda nastavuje znovunahrání stránky.

- __toString ()

  Metoda převede objekt na řetězec.

- getLinkToDownloadFile ($dir, $file)

  Metoda vrací link pro stažení souboru pomocí specílního dwsouboru.

Statické veřejné metody

- static setTransferProtocol ($protocol)

  metoda nastavuje transportní protokol

- static getTransferProtocol ()

  Metoda vrací přenosový protokol.

- static getMainWebDir ()

  Metoda vrací adresu k web aplikaci.

- static checkLangUrlRequest ($lang)

  Metoda kontroluje, jestli se jedná o nastaveni jazykové mutace FORMAT: en, cs, de.

- static checkCategoryUrlRequest ($category)

  Metoda kontroluje, jestli se jedná o kategorii zadanou v url FORMAT: nazev-id.

- static checkRouteUrlRequest ($route)

  Metoda kontroluje, jestli se jedná o routu zadanou v url FORMAT: FORMAT: nazev-rid nebo nazev-r{char}id.

- static checkArticleUrlRequest ($article)

  Metoda kontroluje, jestli se jedná o článek v URL FORMAT: nazev-id.

- static checkActionUrlRequest ($action)

  Metoda kontroluje, jestli se jedná o akci v URL FORMAT: nazev-id.

- static chackOtherUrlParams ($params)

  Metoda rozparsuje ostatní parametry v URL.

Veřejné atributy

- const **URL_PARAMETRES_SEPARATOR** = '&'
- const **URL_SEPARATOR_LINK_PARAMS** = '?'
- const **URL_PARAMETRES_SEPARATOR_IN_URL** = '&'
- const **URL_SEP_PARAM_VALUE** = '='
- const **URL_SEP_ARTICLE_ID** = '-'
- const **URL_SEP_CAT_ID** = '-'
- const **URL_ACTION_LABEL_TYPE_SEP** = '_'
- const **URL_ACTION_TYPE_ID_SEP** = '-'
- const **DOWNLOAD_FILE** = 'download.php'
- const **DOWNLOAD_FILE_FILE_PARAM** = 'file'
- const **DOWNLOAD_FILE_DIR_PARAM** = 'url'
- const **LINK_ARRAY_ITEM_NAME** = 'name'
- const **LINK_ARRAY_ITEM_ID** = 'id'
- const **LINK_ARRAY_ITEM_OPTION** = 'option'
- const **PRETEFINED_ROUTES_ID_PREFIX** = 'p'

8.49.1 Detailní popis

Třída pro práci s odkazy.

Třída pro tvorbu a práci s odkazy aplikace, umožňuje jejich pohodlnou tvorbu a validaci, popřípadě změnu jednotlivých parametrů. Umožňuje také přímé přesměrování na zvolený (vytvořený) odkaz pomocí klauzule redirect.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro práci s odkazy

Definice je uvedena na řádku 15 v souboru links.class.php.

8.49.2 Dokumentace konstruktoru a destruktoru

8.49.2.1 Links::__construct ($ *clear* = `false`, $ *onlyWebRoot* = `false`)

Konstruktor nastaví základní adresy a přenosový protokol.

Parametry:

boolean – (option)true pokud má být vrácen čistý link jenom s kategorií(pokud je vybrána) a jazykem

boolean – (option)true pokud má být vráce naprosto čistý link (web root)

Definice je uvedena na řádku 240 v souboru links.class.php.

```
240                                                                 {
241         $this->clearLink = $clear;
242         $this->onlyWebRoot = $onlyWebRoot;
243         $this->_init();
244     }
```

8.49.3 Dokumentace k metodám

8.49.3.1 static Links::setTransferProtocol ($ *protocol*) [static]

metoda nastavuje transportní protokol

Parametry:

string – přenosový protokol (např. http://)

Definice je uvedena na řádku 254 v souboru links.class.php.

```
254                                                             {
255         self::$user_transfer_protocol = $protocol;
256     }
```

8.49.3.2 static Links::getTransferProtocol () [static]

Metoda vrací přenosový protokol.

Návratová hodnota:

string – přenosový protokol

Definice je uvedena na řádku 262 v souboru links.class.php.

```
262                                                 {
263         if(self::$user_transfer_protocol == null){
264             return UrlRequest::TRANSFER_PROTOCOL;
265         } else {
266             return self::$user_transfer_protocol;
267         }
268     }
```

8.49.3.3 static Links::getMainWebDir () [static]

Metoda vrací adresu k web aplikaci.

Návratová hodnota:

string – adresa ke kořenu aplikace

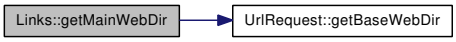
Definice je uvedena na řádku 276 v souboru links.class.php.

Odkazuje se na UrlRequest::getBaseWebDir().

Používá se v UserImagesEplugin::getImagesListLink() a TinyMce::initJsPlugin().

```
276                                                 {
277 //          return self::getTransferProtocol().self::$serverName.self::$webUrl.self::COOL_URL_SEPARATOR;
278             return UrlRequest::getBaseWebDir();
279         }
```

Tato funkce volá...



8.49.3.4 Links::category ($ *catName* = `null`, $ *catId* = `null`)

Metoda nastavuje název a id kategorie.

Parametry:

string – klíč kategorie

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 291 v souboru links.class.php.

```
291                                                                 {
292         if($catName != null AND $catId != null){
293             $help = new TextCtrlHelper();
294 
295             $this->category[self::LINK_ARRAY_ITEM_NAME] = rawurlencode($help->utf2ascii($catName));
296             $this->category[self::LINK_ARRAY_ITEM_ID] = $catId;
297         }
298         return $this;
299     }
```

8.49.3.5 Links::route ($ *name* = `null`, $ *id* = `null`)

Metoda nastavuje název a id routy.

Parametry:

string – název cesty

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 307 v souboru links.class.php.

```
307                                                 {
308         //    Pokud je předána předdefinovaná cesta
309         if(is_array($name)){
310             $this->route[self::LINK_ARRAY_ITEM_NAME] = $name[0];
311             $this->route[self::LINK_ARRAY_ITEM_ID] = $name[1];
312             $this->route[self::LINK_ARRAY_ITEM_OPTION] = true;
```

```
313         }
314         //        pokud je předána uživatelská cesta
315         if($name != null AND $id != null){
316            $this->route[self::LINK_ARRAY_ITEM_NAME] = $name;
317            $this->route[self::LINK_ARRAY_ITEM_ID] = $id;
318            $this->route[self::LINK_ARRAY_ITEM_OPTION] = false;
319         }
320         return $this;
321     }
```

8.49.3.6 Links::article ($ *article* = `null`, $ *idArticle* = `null`)

Metoda nastavuje název article.

Parametry:

string – jméno článku

integer – id článku

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 330 v souboru links.class.php.

```
330                                                                 {
331         if($article != null AND $idArticle != null AND is_numeric($idArticle)){
332            $this->article[self::LINK_ARRAY_ITEM_ID] = $idArticle;
333            $article = htmlspecialchars_decode($article);
334            $help = new TextCtrlHelper();
335            $article = $help->utf2ascii($article);
336            $article = urlencode($article);
337            $this->article[self::LINK_ARRAY_ITEM_NAME] = $article;
338         } else {
339            $this->article[self::LINK_ARRAY_ITEM_ID] = null;
340            $this->article[self::LINK_ARRAY_ITEM_NAME] = null;
341         }
342         return $this;
343     }
```

8.49.3.7 Links::media ($ *media* = `null`)

Metoda nastavuje typ media.

Parametry:

string – jméno media

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 351 v souboru links.class.php.

```
351                                         {
352         $this->mediaType = $media;
353         return $this;
354     }
```

8.49.3.8 Links::file ($ *file* = null)

Metoda nastavuje parametr nazev epluginu.

Parametry:

string – jméno epluginu

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 362 v souboru links.class.php.

```
362                                     {
363       $this->file = $file;
364       return $this;
365    }
```

8.49.3.9 Links::action ($ *action* = null)

Metoda nastavuje název action.

Parametry:

string – jméno akce (action např. edit, show atd.)

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 373 v souboru links.class.php.

```
373                                         {
374       if(!empty ($action)){
375          $this->action[self::LINK_ARRAY_ITEM_NAME] = $action[0];
376          $this->action[self::LINK_ARRAY_ITEM_OPTION] = $action[1];
377          $this->action[self::LINK_ARRAY_ITEM_ID] = $action[2];
378       } else {
379          $this->action= null;
380          //            $this->action[self::LINK_ARRAY_ITEM_OPTION] = $action[1];
381          //            $this->action[self::LINK_ARRAY_ITEM_ID] = $action[2];
382       }
383       return $this;
384    }
```

8.49.3.10 Links::lang ($ *lang* = null)

Metoda nastavuje název lokalizace.

Parametry:

string – jméno jazyka (action např. cs, en, de atd.)

Návratová hodnota:

Links – objket Links

Definice je uvedena na řádku 392 v souboru links.class.php.

Používá se v __toString().

```
392                                     {
393         $this->lang = $lang;
394         return $this;
395     }
```

8.49.3.11 Links::param ($ *name*, $ *value* = null)

Metoda nastavuje prametry do url (je li jich více).

Parametry:

array – pole parametrů (název=>hodnota)

boolean – jestli se mají použít i ostatní parametry

array – pole parametrů, které se mají odstranit //TODO zatím neimplementováno!!

Návratová hodnota:

Links – objket Links TODO není implementována, možná nebude třeba Metoda nastavuje parametr do url

Parametry:

string – název parametru

string – hodnota parametru

Návratová hodnota:

Links – objket Links Metoda odstraňuje zadaný parametr z url

Parametry:

string – název parametru, který se odebere Metoda přidá nebo změní daný parametr v URL

mixed $name – objekt UrlParam nebo string

mixed $value – (option) hodnota parametru, jen v případě přímého předávání názvu parametru

Definice je uvedena na řádku 456 v souboru links.class.php.

```
456                                             {
457         //        Je vkládán objekt parametru
458         if($name instanceof UrlParam){
459            //            $name = new UrlParam();
460            // Pokud se nejedná o normálový
461            if(!$name->isNormalParam()){
462               $add = true;
463               // Projdem pole a otestujem hodnoty naproti regulérnímu výrazu parametru
464               foreach ($this->paramsArray as $paramKey => $param) {
465                  if(ereg($name->getPattern(), $param)){
466                     $add = false;
```

```
467                   $this->paramsArray[$paramKey] = rawurlencode($name);
468                }
469             }
470
471             //               Pokud parametr v url není
472             if($add){
473                array_push($this->paramsArray, rawurlencode($name));
474             }
475          } else {
476             $this->paramsNormalArray[$name] = rawurlencode($value);
477          }
478       }
479       //        Jedná se o běžně zadaný parametr
480       else {
481          if(!is_array($name)){
482             $this->paramsNormalArray[$name] = rawurlencode($value);
483          } else {
484             foreach ($name as $key => $val) {
485                $this->paramsNormalArray[$key] = rawurlencode($val);
486             }
487          }
488       }
489       return $this;
490    }
```

8.49.3.12 Links::rmParam ($ *name* = null)

Metoda odstraní daný parametr z url.

Parametry:

mixed $name – (option) název parametru, který se má odstranit nebo objekt UrlParam. Pokud zůstane nezadán, odstraní se všechny parametry

Návratová hodnota:

Links

Definice je uvedena na řádku 500 v souboru links.class.php.

```
500                                      {
501       //        Je vkládán objekt parametru
502       if($name instanceof UrlParam){
503          //            $name = new UrlParam();
504          // Pokud se nejedná o normálový
505          if(!$name->isNormalParam()){
506             // Projdem pole a otestujem hodnoty naproti regulérnímu výrazu parametru
507             foreach ($this->paramsArray as $paramKey => $param) {
508                if(ereg($name->getPattern(), $param)){
509                   unset ($this->paramsArray[$paramKey]);
510                }
511             }
512          } else {
513             unset ($this->paramsNormalArray[$name]);
514          }
515       }
516       //        Jedná se o běžně zadaný parametr
517       else if($name != null) {
518          unset($this->paramsNormalArray[$name]);
519       }
520       //        Odstranění vších parametrů (normálových i obyčejných)
521       else {
```

```
522           $this->paramsArray = array();
523           $this->paramsNormalArray = array();
524        }
525        return $this;
526     }
```

8.49.3.13 Links::reload ($ *link* = null)

Metoda nastavuje znovunahrání stránky.

Parametry:

string – externí odkaz na který se má přesměrovat (option)

Definice je uvedena na řádku 532 v souboru links.class.php.

```
532                                        {
533        if($link == null){
534           header("location: ".$this);
535        } else {
536           header("location: ".(string)$link);
537        }
538        exit();
539     }
```

8.49.3.14 Links::__toString ()

Metoda převede objekt na řetězec.

Návratová hodnota:

string – objekt jako řetězec

Plánované úpravy

What is this? Know anybody what is this doing?

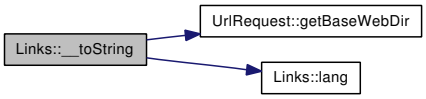
Definice je uvedena na řádku 594 v souboru links.class.php.

Odkazuje se na UrlRequest::getBaseWebDir() a lang().

```
595     {
596        //       if(!$this->relativePath){
597        $returnString = UrlRequest::getBaseWebDir();
598
599 //       if($this->file != null){
600 //          $returnString.=$this->file;
601 //       }
602        //       } else {
603        //          $returnString = './';
604        //       }
605        if($this->lang != null){
606           $returnString.=$this->getLang();
607        }
608        if($this->getCategory() != null){
609           $returnString.=$this->getCategory();
610        }
```

```
611       if($this->getRoute() != null){
612          $returnString.=$this->getRoute();
613       }
614       if($this->getArticle() != null){
615          $returnString.=$this->getArticle();
616       }
617       if($this->getAction() != null){
618          $returnString.=$this->getAction();
619       }
620
621       //        $this->getMedia();
622       //        Parsovatelné parametry
623       if($this->getParams() != null){
624          $returnString.=$this->getParams();
625       }
626
630       if($this->getFile() != null){
631          $returnString.=$this->getFile();
632       }
633
634       //        normálové parametry
635       if($this->getNormalParams() != null){
636          $returnString.=$this->getNormalParams();
637       }
638
639       return $returnString;
640    }
```

Tato funkce volá...



8.49.3.15 Links::getLinkToDownloadFile ($ *dir*, $ *file*)

Metoda vrací link pro stažení souboru pomocí specílního dwsouboru.

Parametry:

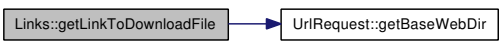
string – cesta ks ouboru
string – název souboru

Definice je uvedena na řádku 647 v souboru links.class.php.

Odkazuje se na UrlRequest::getBaseWebDir().

```
647                                                    {
648       $dwLink = UrlRequest::getBaseWebDir().self::DOWNLOAD_FILE.self::URL_SEPARATOR_LINK_PARAMS.
649       self::DOWNLOAD_FILE_DIR_PARAM.self::URL_SEP_PARAM_VALUE.urlencode($dir).
650       self::URL_PARAMETRES_SEPARATOR.self::DOWNLOAD_FILE_FILE_PARAM.self::URL_SEP_PARAM_VALUE.
651       $file;
652
653       return $dwLink;
654    }
```

Tato funkce volá...



8.49.3.16 static Links::checkLangUrlRequest ($ *lang*) [static]

Metoda kontroluje, jestli se jedná o nastaveni jazykové mutace FORMAT: en, cs, de.

Parametry:

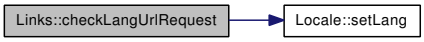
string $lang – řetězec s jazykem

Definice je uvedena na řádku 666 v souboru links.class.php.

Odkazuje se na Locale::setLang().

```
666                                                 {
667       $matches = array();
668       if(eregi('^([a-zA-Z]{2})$', $lang, $matches)){
669          Locale::setLang($matches[1]);
670          self::$currentlang = $matches[1];
671          return true;
672       }
673       return false;
674    }
```

Tato funkce volá...



8.49.3.17 static Links::checkCategoryUrlRequest ($ *category*) [static]

Metoda kontroluje, jestli se jedná o kategorii zadanou v url FORMAT: nazev-id.

Parametry:

string $category – řetězec s kategorií

Definice je uvedena na řádku 682 v souboru links.class.php.

Odkazuje se na Category::setCurrentCategoryId().

```
682                                                         {
683       $matches = array();
684       if(eregi('^([a-zA-Z0-9%\-]+)-([0-9]+)$', $category, $matches)){
685          Category::setCurrentCategoryId($matches[2]);
686          self::$currentCategory[self::LINK_ARRAY_ITEM_ID] = $matches[2];
687          self::$currentCategory[self::LINK_ARRAY_ITEM_NAME] = $matches[1];
688          return true;
689       }
690       return false;
691    }
```

Tato funkce volá...

8.49.3.18 static Links::checkRouteUrlRequest ($ *route*) [static]

Metoda kontroluje, jestli se jedná o routu zadanou v url FORMAT: FORMAT: nazev-rid nebo nazev-r{char}id.

Parametry:

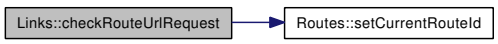
string $route – řetězec s cestou

Definice je uvedena na řádku 716 v souboru links.class.php.

Odkazuje se na Routes::setCurrentRouteId().

```
716                                                 {
717       $matches = array();
718       if(eregi('^([a-zA-Z0-9%\-]+)'.Routes::ROUTE_URL_ID_SEPARATOR.'r([a-z]?[0-9]+)$',$route, $matches
719
720          self::$currentRoute[self::LINK_ARRAY_ITEM_NAME] = $matches[1];
721          $matches2 = array();
722          $pattern = '^r'.self::PRETEFINED_ROUTES_ID_PREFIX.'([0-9]+)';
723          //    Pokud je předdefinovaná cesta
724          if(eregi($pattern, $id, $matches2)){
725             self::$currentRoute[self::LINK_ARRAY_ITEM_ID] = $matches2[1];
726             self::$currentRoute[self::LINK_ARRAY_ITEM_OPTION] = true;
727             Routes::setCurrentRouteId($matches2[2], true);
728          }
729          //    je použita výchozí cesta
730          else if(eregi('r([0-9]+)', $id, $matches2)) {
731             self::$currentRoute[self::LINK_ARRAY_ITEM_ID] = $matches2[1];
732             self::$currentRoute[self::LINK_ARRAY_ITEM_OPTION] = false;
733             Routes::setCurrentRouteId($matches2[2], false);
734          }
735          return true;
736       }
737       return false;
738    }
```

Tato funkce volá...



8.49.3.19 static Links::checkArticleUrlRequest ($ *article*) [static]

Metoda kontroluje, jestli se jedná o článek v URL FORMAT: nazev-id.

Parametry:

string $article – řetězec s článkem

Definice je uvedena na řádku 766 v souboru links.class.php.

Odkazuje se na Article::setCurrentArticleId().

```
766                                                     {
767       $matches = array();
768       if(eregi('^([a-zA-Z0-9%\-]+)-([0-9]+)$', $article, $matches)){
769          Article::setCurrentArticleId($matches[2]);
```

```
770          self::$currentArticle[self::LINK_ARRAY_ITEM_ID] = $matches[2];
771          self::$currentArticle[self::LINK_ARRAY_ITEM_NAME] = $matches[1];
772          return true;
773       }
774       return false;
775    }
```

Tato funkce volá...

8.49.3.20 static Links::checkActionUrlRequest ($ *action*) [static]

Metoda kontroluje, jestli se jedná o akci v URL FORMAT: nazev-id.

Parametry:

string $action – řetězec s akcí

Definice je uvedena na řádku 800 v souboru links.class.php.

Odkazuje se na Action::setAction().

```
800                                                    {
801       $matches = array();
802       $regex = '^([a-zA-Z0-9%\-]+)'.self::URL_ACTION_LABEL_TYPE_SEP.'([a-zA-Z]+)'
803       .self::URL_ACTION_TYPE_ID_SEP.'([0-9]+)$';
804       if(eregi($regex, $action, $matches)){
805          Action::setAction($matches[2], $matches[3]);
806          self::$currentAction[self::LINK_ARRAY_ITEM_ID] = $matches[3];
807          self::$currentAction[self::LINK_ARRAY_ITEM_OPTION] = $matches[2];
808          self::$currentAction[self::LINK_ARRAY_ITEM_NAME] = $matches[1];
809          return true;
810       }
811       return false;
812    }
```

Tato funkce volá...

8.49.3.21 static Links::chackOtherUrlParams ($ *params*) [static]

Metoda rozparsuje ostatní parametry v URL.

Parametry:

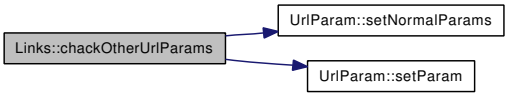
array – pole s ostatními parametry v URL

Definice je uvedena na řádku 837 v souboru links.class.php.

Odkazuje se na UrlParam::setNormalParams() a UrlParam::setParam().

```
837                                                     {
838       if(!empty ($params)){
839          //  Vytažení ostatních parametrů v url do objektu UrrlParam
840          foreach ($params as $item) {
841             // Pokud se jedná o parametr přenášený pomocí objektu
842             if($item[0] != '?'){
843                array_push(self::$currentParamsArray, rawurldecode($item));
844                UrlParam::setParam($item);
845             }
846             // Pokud  se jedná o normálový parametr
847             else {
848                // Budou parsovány
849                self::$currentParamsNormalArray = self::parseNormalParams($item);
850                UrlParam::setNormalParams(self::$currentParamsNormalArray);
851             }
852          }
853       }
854    }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/links.class.php

8.50 Dokumentace třídy Locale

Třída pro práci s locale (místním nastavením).

Statické veřejné metody

- static factory ()
  Metoda pro vytvoření prostředí třídy locales.
- static getAppLangsNames ()
  Metoda vrací pole s názvy jazyků.
- static getLocale ($lang=null)
  Metoda vrací zvolené locales pro zadaný jazyk.
- static getDefaultLang ()
  Metoda vrací výchozí jazyk aplikace (první, uvedený v configu).
- static getAppLangs ()
  Metoda vrací pole jazyků aplikace.
- static langExist ($lang)
  Metoda vrací true pokud jazyk existuje //TODO private funkce.
- static setLang ($lang)
  Metoda nastaví vybraný jazyk.
- static getLang ()
  Metoda vrací vybraný jazyk aplikace.
- static getLangUrlPart ()
  Metoda vrací vybraný jazyk aplikace, pokud není shodný s výchozím.
- static switchToEngineTexts ()
  Metoda přenastaví lokalizační texty na engine.
- static switchToModuleTexts ($moduleName=null)
  Metoda přenastaví lokalizační texty na engine.
- static bindTextDomain ($moduleName=null)
  Metoda přidá textovou doménu pro překlad.

Veřejné atributy

- const **LANG_SEPARATOR** = ';'
- const **URL_PARAM_WITH_LANG** = 'lang'
- const **LOCALES_DIR** = 'locale'

8.50.1 Detailní popis

Třída pro práci s locale (místním nastavením).

Třída slouží pro práci s jazykovým nastavením aplikace. Je určena k volbě výchozího a zvoleného jazyka aplikace. Lze s ní získat i kompletní výpis všech jazyků, a všech použitých jazyků v aplikaci.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: locale.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu jazykového nastavení

Definice je uvedena na řádku 14 v souboru locale.class.php.

8.50.2 Dokumentace k metodám

8.50.2.1 static Locale::factory () [static]

Metoda pro vytvoření prostředí třídy locales.

Parametry:

Config – objekt systémové konfigurace

Definice je uvedena na řádku 77 v souboru locale.class.php.

```
77                                      {
78       self::parseLangs();
79
80       if (!isset($_GET[self::URL_PARAM_WITH_LANG])){
81          self::$selectLang = self::$defaultLang;
82       } else {
83
84          if(self::langExist($_GET[self::URL_PARAM_WITH_LANG])){
85             self::$selectLang = $_GET[self::URL_PARAM_WITH_LANG];
86          } else {
87             self::$selectLang = self::$defaultLang;
88          }
89       }
90       self::_setLangTranslations();
91    }
```

8.50.2.2 static Locale::getAppLangsNames () [static]

Metoda vrací pole s názvy jazyků.

Návratová hodnota:

array – pole s názvy jazyků

Definice je uvedena na řádku 98 v souboru locale.class.php.

Používá se v AppCore::assignMainVarsToTemplate().

```
98                                          {
99       $returnArray = array();
100       foreach (self::getAppLangs() as $langKey => $lang) {
101          $returnArray[$lang] = self::$localesNames[$lang];
102       }
103       return $returnArray;
104    }
```

8.50.2.3 static Locale::getLocale ($ *lang* = null) [static]

Metoda vrací zvolené locales pro zadaný jazyk.

Parametry:

string – jazyk (cs, en, de, ...)

Definice je uvedena na řádku 121 v souboru locale.class.php.

```
121                                                 {
122       if($lang == null){
123          return self::$locales[self::$selectLang];
124       } else {
125          if(key_exists($lang, self::$locales)){
126             return self::$locales[$lang];
127          } else {
128             reset(self::$locales);
129             return current(self::$locales);
130          }
131       }
132    }
```

8.50.2.4 static Locale::getDefaultLang () [static]

Metoda vrací výchozí jazyk aplikace (první, uvedený v configu).

Návratová hodnota:

string – výchozí jazyk (cs, en, ..)

Definice je uvedena na řádku 138 v souboru locale.class.php.

Používá se v SponsorsController::addController(), GuestbookController::addController(), AppCore::assignMainVarsToTemplate(), SponsorsController::editController(), GuestbookController::editController(), Category::getDefaultCategory(), GaleryDetailModel::getGaleryDetail(), GaleryDetailModel::getPhotosList(), SponsorsController::mainController(), AppCore::runModules() a AppCore::runPanel().

```
138                                         {
139       return self::$defaultLang;
140    }
```

8.50.2.5 static Locale::getAppLangs () [static]

Metoda vrací pole jazyků aplikace.

Návratová hodnota:

array – pole jazyků aplikace

Definice je uvedena na řádku 157 v souboru locale.class.php.

Používá se v SponsorsController::addController(), GuestbookController::addController(), AppCore::assignMainVarsToTemplate(), SponsorsController::editController(), GuestbookController::editController(), LocaleCtrlHelper::generateArray(), LocaleCtrlHelper::postsToArray() a Form::setValue().

```
157                                         {
158         return self::$appLangs;
159     }
```

8.50.2.6 static Locale::langExist ($ *lang*) [static]

Metoda vrací true pokud jazyk existuje //TODO private funkce.

Parametry:

string – název jazyku (cs, en, ...)

Návratová hodnota:

boolean – true pro existenci jazyku

Definice je uvedena na řádku 167 v souboru locale.class.php.

```
167                                             {
168         if(in_array($lang, self::$appLangs)){
169             return true;
170         } else {
171             return false;
172         }
173     }
```

8.50.2.7 static Locale::setLang ($ *lang*) [static]

Metoda nastaví vybraný jazyk.

Parametry:

string – název jazyku

Definice je uvedena na řádku 179 v souboru locale.class.php.

Používá se v Links::checkLangUrlRequest().

```
179                                         {
180 //    echo "nastaven".$lang;
181       if(self::langExist($lang)){
182          self::$selectLang = $lang;
183       } else {
184          self::$selectLang = self::$defaultLang;
185       }
186    }
```

8.50.2.8 static Locale::getLang () [static]

Metoda vrací vybraný jazyk aplikace.

Návratová hodnota:

string – vybraný jazyk aplikace

Definice je uvedena na řádku 192 v souboru locale.class.php.

Používá se v TimeValidator::checkDate(), DateTimeCtrlHelper::checkDate(), NewsController::editController(), Category::getDefaultCategory(), GaleryDetailModel::getGaleryDetail(), GaleryDetailModel::getPhotosList(), TinyMce::initJsPlugin(), SponsorsController::mainController(), AppCore::runModules() a AppCore::runPanel().

```
192                                     {
193       return self::$selectLang;
194    }
```

8.50.2.9 static Locale::getLangUrlPart () [static]

Metoda vrací vybraný jazyk aplikace, pokud není shodný s výchozím.

Návratová hodnota:

string – vybraný jazyk aplikace

Definice je uvedena na řádku 200 v souboru locale.class.php.

```
200                                             {
201         if(self::getLang() != self::getDefaultLang()){
202             return self::$selectLang;
203         }
204         return null;
205    }
```

8.50.2.10 static Locale::switchToModuleTexts ($ *moduleName* = null) [static]

Metoda přenastaví lokalizační texty na engine.

Parametry:

string – název modulu, na který se mají texty přnastavit (option)\ pokud je prázdná použije se zvolený modul enginu

Definice je uvedena na řádku 220 v souboru locale.class.php.

Odkazuje se na AppCore::getSelectedModule().

Používá se v AppCore::runModules() a AppCore::runPanel().

```
220                                                                 {
221       if($moduleName == null AND AppCore::getSelectedModule() != null){
222          textdomain(AppCore::getSelectedModule()->getName());
223       } else if($moduleName != null){
224          textdomain($moduleName);
225       }
226    }
```

Tato funkce volá...

8.50.2.11 static Locale::bindTextDomain ($ *moduleName* = null) [static]

Metoda přidá textovou doménu pro překlad.

Parametry:

string – název modulu pro kterou se má překlad přidat (Option)

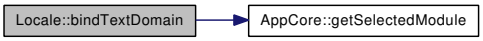
Definice je uvedena na řádku 233 v souboru locale.class.php.

Odkazuje se na AppCore::getSelectedModule().

Používá se v AppCore::runModules().

```
233                                                             {
234       if($moduleName == null AND AppCore::getSelectedModule() != null){
235          bindtextdomain(AppCore::getSelectedModule()->getName(), '.' . DIRECTORY_SEPARATOR . AppCore::
236          . DIRECTORY_SEPARATOR . AppCore::getSelectedModule()->getName() . DIRECTORY_SEPARATOR. self::
237       } else if($moduleName != null){
238          bindtextdomain($moduleName, '.' . DIRECTORY_SEPARATOR . AppCore::MODULES_DIR
239          . DIRECTORY_SEPARATOR . $moduleName . DIRECTORY_SEPARATOR. self::LOCALES_DIR);
240       }
241    }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/locale.class.php

8.51 Dokumentace třídy LocaleCtrlHelper

Ttřída lokalizačního Controll Helperu.

Diagram dědičnosti pro třídu LocaleCtrlHelper

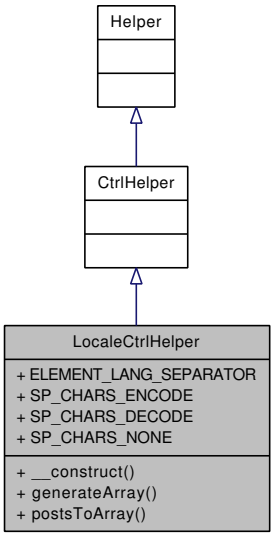
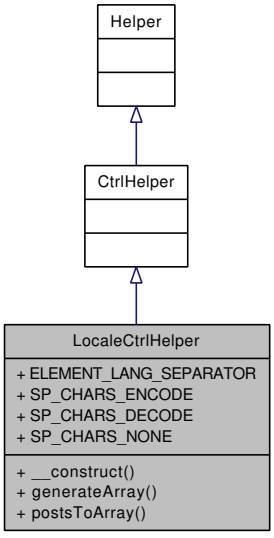


Diagram tříd pro LocaleCtrlHelper:



Veřejné metody

- __construct ()

  Konstruktor třídy Konstruktor třídy.

- generateArray ($arrayElements, $values=null, $alternativeValues=null)

Metoda vygeneruje pole pro zadávání ve více jazycích (počet jazyků je brán z celého systému).

- postsToArray ($sendPostsArray, $postPrefix=null, $specialChars=self::SP_CHARS_ENCODE, $withPrefix=false)

  Metoda převede posty na jazykové pole.

Veřejné atributy

- const **ELEMENT_LANG_SEPARATOR** = '_'
- const **SP_CHARS_ENCODE** = 'encode'
- const **SP_CHARS_DECODE** = 'decode'
- const **SP_CHARS_NONE** = 'none'

8.51.1 Detailní popis

Ttřída lokalizačního Controll Helperu.

Třída poskytuje metody pro zjednodušení práce s lokalizačními řetězci. Slouží také pro generování a parsování lokalizačních polí.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: localectrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s lokalizačními prvky v kontroleru - helper

Definice je uvedena na řádku 13 v souboru localectrlhelper.class.php.

8.51.2 Dokumentace k metodám

8.51.2.1 LocaleCtrlHelper::generateArray ($ *arrayElements*, $ *values* = `null`, $ *alternativeValues* = `null`)

Metoda vygeneruje pole pro zadávání ve více jazycích (počet jazyků je brán z celého systému).

Parametry:

array – Pole s elementy (např. label, text)

array – Pole s hodnotymi elemntů

array – Pole s alternativními hodnotami, pokud hodnota není ve values

Zastaralé

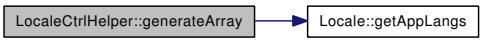
– lepší použít přímo validítor formuláře

Definice je uvedena na řádku 54 v souboru localectrlhelper.class.php.

Odkazuje se na Locale::getAppLangs().

```
54                                                                                                                  {
55          $returnArray = array();
56
57  //      Test jestli je pole
58          if(!is_array($arrayElements)){
59             $arrayElements = array($arrayElements);
60          }
61
62  //      Oddělání posledního znaku určeného pro oddělení elementu a jazyku
63          $elements = array();
64          foreach ($arrayElements as $el) {
65             if($el[strlen($el)-1] == self::ELEMENT_LANG_SEPARATOR){
66                array_push($elements, substr($el, 0, strlen($el)-1));
67             } else {
68                array_push($elements, $el);
69             }
70          }
71
72          foreach (Locale::getAppLangs() as $lang) {
73             $returnArray[$lang] = array();
74             foreach ($elements as $element) {
75                $returnArray[$lang][$element] = null;
76                if(isset($values[$element.self::ELEMENT_LANG_SEPARATOR.$lang])
77  //                AND $values[$element.self::ELEMENT_LANG_SEPARATOR.$lang] != null
78                   ){
79                   $returnArray[$lang][$element] = $values[$element.self::ELEMENT_LANG_SEPARATOR.$lang];
80                } else if (isset($alternativeValues[$element.self::ELEMENT_LANG_SEPARATOR.$lang])
81  //                AND $alternativeValues[$element.self::ELEMENT_LANG_SEPARATOR.$lang] != null
82                   ) {
83                   $returnArray[$lang][$element] = $alternativeValues[$element.self::ELEMENT_LANG_SEPARATOR
84                } else {
85                   $returnArray[$lang][$element] = null;
86                }
87             }
88          }
89          return $returnArray;
90       }
```

Tato funkce volá...



8.51.2.2 LocaleCtrlHelper::postsToArray ($ *sendPostsArray*, $ *postPrefix* = `null`, $ *specialChars* = `self::SP_CHARS_ENCODE`, $ *withPrefix* = `false`)

Metoda převede posty na jazykové pole.

Parametry:

array – pole s názvy postů

string – prefix v názvu postu

boolean – true pokud se mají znaky zakódovat

boolean – true pokud se mají zny dekódovat

boolean – jestli se mají klíče výstupního polu ukládat s prefixem

Návratová hodnota:

array – pole upravených postů

Zastaralé

– lepší použít přímo validítor formuláře

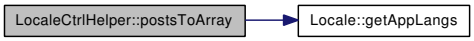
Definice je uvedena na řádku 103 v souboru localectrlhelper.class.php.

Odkazuje se na Locale::getAppLangs().

```
103
104       $returnArray = array();
105
106       if(is_array($specialChars)){
107          $specialCharsArray = true;
108       } else {
109          $specialCharsArray = false;
110       }
111
112 //    Uprava postu o doplnění znaku pro oddělení postu a jazyku
113       $postsArray = array();
114
115       if(!is_array($sendPostsArray)){
116          $sendPArray[0]=$sendPostsArray;
117       } else {
118          $sendPArray = $sendPostsArray;
119       }
120
121       if(!empty ($sendPArray)){
122          foreach ($sendPArray as $post) {
123             if($post[strlen($post)-1] == self::ELEMENT_LANG_SEPARATOR){
124                array_push($postsArray, $post);
125             } else {
126                array_push($postsArray, $post.self::ELEMENT_LANG_SEPARATOR);
127             }
128          }
129       }
130
131
132       foreach (Locale::getAppLangs() as $lang) {
133          foreach ($postsArray as $postKey => $post) {
134             if ($_POST[$postPrefix.$post.$lang] != null){
135                $sendPost = $_POST[$postPrefix.$post.$lang];
136
137                if($specialCharsArray){
138                   if($specialChars[$postKey] == self::SP_CHARS_ENCODE){
139                      $sendPost = htmlspecialchars($sendPost);
140                   } else if($specialChars[$postKey] == self::SP_CHARS_DECODE){
141                      $sendPost = htmlspecialchars_decode($sendPost);
142                   }
143                } else {
144                   if($specialChars == self::SP_CHARS_ENCODE){
145                      $sendPost = htmlspecialchars($sendPost);
146                   } else if($specialChars == self::SP_CHARS_DECODE){
147                      $sendPost = htmlspecialchars_decode($sendPost);
148                   }
149                }
150             }else {
151                $sendPost = null;
152             }
153
154             if(!$withPrefix){
155                $returnArray[$post.$lang] = $sendPost;
156                $value = &$returnArray[$post.$lang];
157             } else {
158                $returnArray[$postPrefix.$post.$lang] = $sendPost;
159                $value = &$returnArray[$postPrefix.$post.$lang];
160             }
161          }
```

```
162         }
163         return $returnArray;
164     }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/localectrlhelper.class.php

8.52 Dokumentace třídy LoginAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu LoginAction

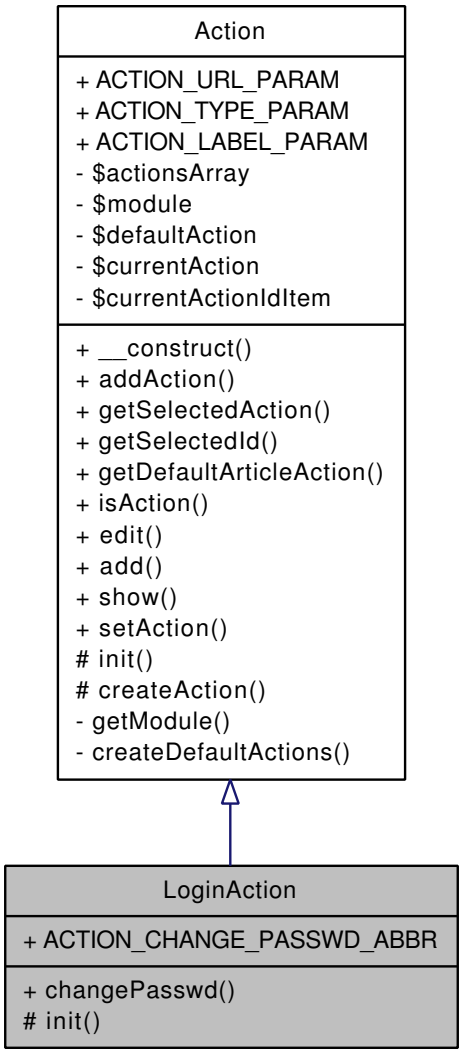
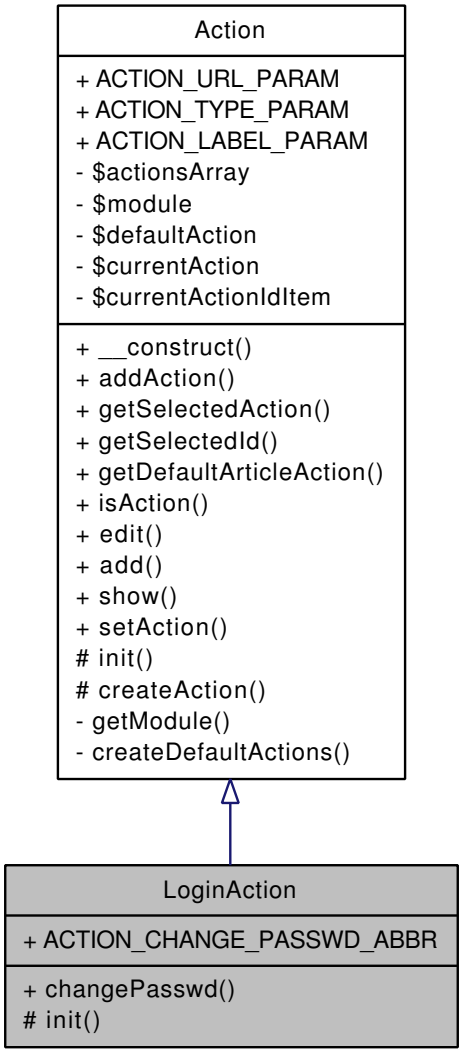


Diagram tříd pro LoginAction:



Veřejné metody

- **changePasswd** ()

Veřejné atributy

- const **ACTION_CHANGE_PASSWD_ABBR** = 'cp'

Chráněné metody

- init ()

  Metoda pro inicializaci akcí.

8.52.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/login/action.class.php

8.53 Dokumentace třídy LoginController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu LoginController

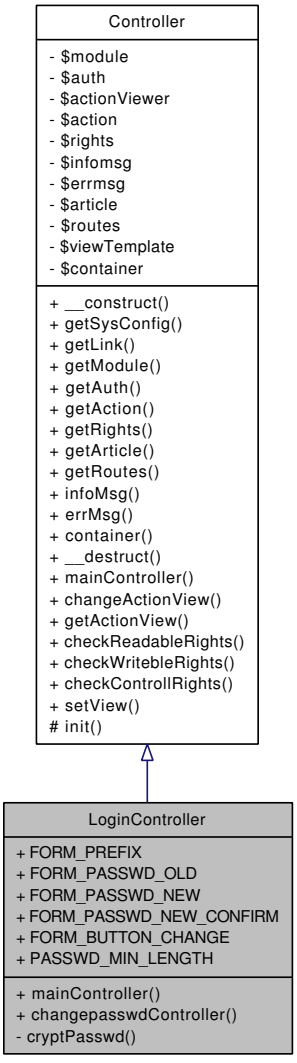


Diagram tříd pro LoginController:

Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- changepasswdController ()

  Metoda pro úpravu hesla.

Veřejné atributy

- const **FORM_PREFIX** = 'passwd_'
- const **FORM_PASSWD_OLD** = 'old'
- const **FORM_PASSWD_NEW** = 'new'

- const **FORM_PASSWD_NEW_CONFIRM** = 'new_confirm'
- const **FORM_BUTTON_CHANGE** = 'change'
- const **PASSWD_MIN_LENGTH** = 5

8.53.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/login/controler.class.php

8.54 Dokumentace třídy LoginRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu LoginRoutes

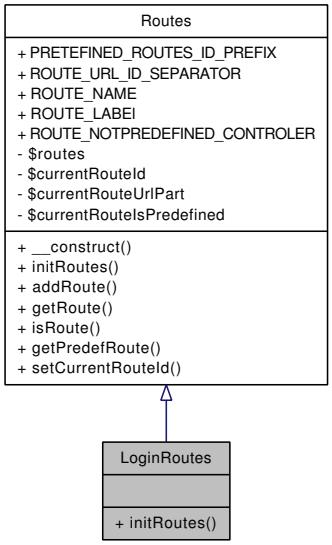
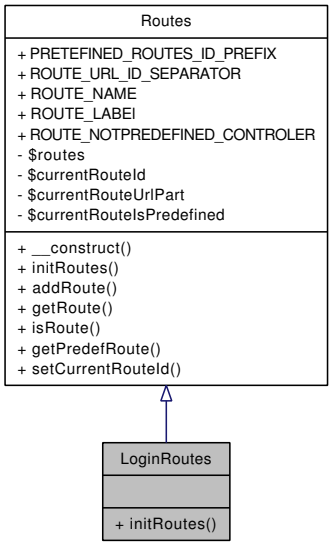


Diagram tříd pro LoginRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

8.54.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/login/routes.class.php

8.55 Dokumentace třídy LoginView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu LoginView

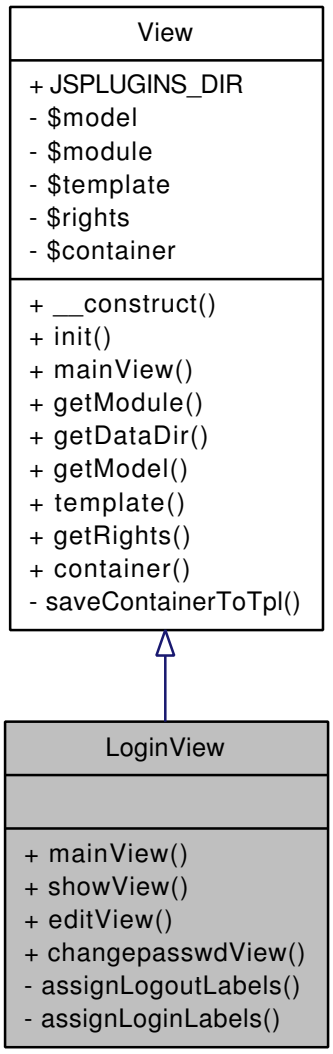
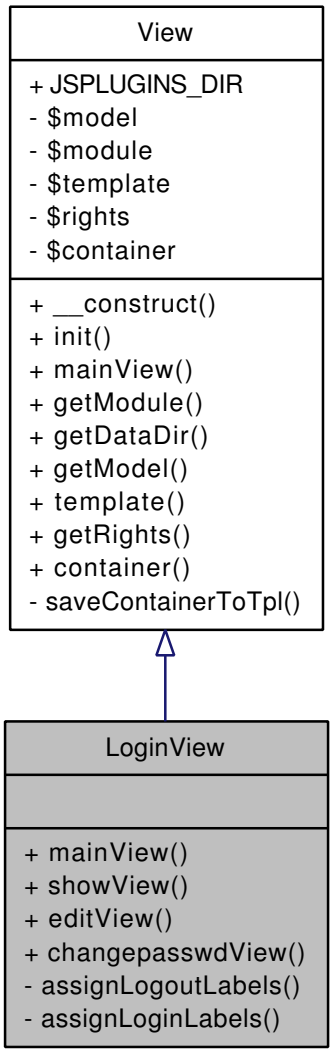


Diagram tříd pro LoginView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- showView ()

  Viewer pro zobrazení detailu.

- **editView** ()
- changepasswdView ()

  Viewwer pro zobrazení změny hesla.

8.55.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/login/view.class.php

8.56 Dokumentace třídy MailCtrlHelper

Ttřída Mail Controll Helperu pro zjednodušení práce s emaily.

Diagram dědičnosti pro třídu MailCtrlHelper

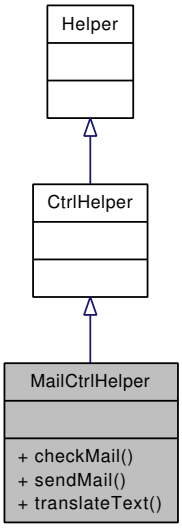
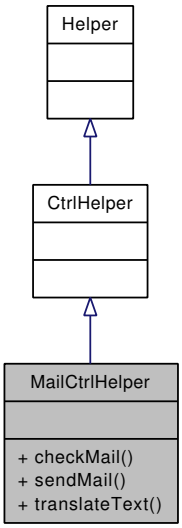


Diagram tříd pro MailCtrlHelper:



Veřejné metody

- checkMail ($email)

  Metoda kontroluje emailovou adresu.

- sendMail ($toMail, $subject, $message, $fromMail, $headers=null)

  Metoda odešle zadaný email.

- translateText ($text, $translateArray)

  Metoda přeloží podle zadaného pole vnitřek emailu.

8.56.1 Detailní popis

Ttřída Mail Controll Helperu pro zjednodušení práce s emaily.

Třída slouží pro práci s emailovými adresami a emaily(odesílání, atd).

Copyright (c) 2008 Jakub Matas

Verze:

$Id: mailctrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s mail prvky v kontroleru - helper

Definice je uvedena na řádku 12 v souboru mailctrlhelper.class.php.

8.56.2 Dokumentace k metodám

8.56.2.1 MailCtrlHelper::checkMail ($ *email*)

Metoda kontroluje emailovou adresu.

Parametry:

string – adresa, která se má kontrolovat

Návratová hodnota:

boolean – true pokud se jedná o email

Zastaralé

– je lepší použít UrlValidator

Plánované úpravy

– odstranit (je obsažena v UrlValidator)

Definice je uvedena na řádku 21 v souboru mailctrlhelper.class.php.

```
21                                      {
22         $name = '[-a-z0-9!#$%&\'*+/=?^_`{|}~]'; // znaky tvořící uživatelské jméno
23         $domain = '[a-z0-9]([-a-z0-9]{0,61}[a-z0-9])'; // jedna komponenta domény
24         return eregi("^$name+(\\.$name+)*@($domain?\\.)+$domain\$", $email);
25 //      if (eregi("^[a-z0-9_\.]+@[a-z0-9_\.]+[a-z]{2,3}$", $email)) {
26 //         return true;
27 //      }
28 //      else {
29 //         return false;
30 //      }
31     }
```

8.56.2.2 MailCtrlHelper::sendMail ($ *toMail*, $ *subject*, $ *message*, $ *fromMail*, $ *headers* = `null`)

Metoda odešle zadaný email.

Parametry:

string – na jaký mail se má zpráva odeslat
string – předmět zprávy
string – samotná zpráva
string – z jakého emailu se bude odesílat
string – (option) hlavičky emailu

Návratová hodnota:

boolean – true pokud se podařilo email odeslat

Definice je uvedena na řádku 44 v souboru mailctrlhelper.class.php.

```
44                                                                                                                  {
45 //    Vytvoření hlavičky
46 //     -=-=-=- MAIL HEADERS
47       $headers .= "From: $fromMail\n";
48       $headers .= "Reply-To: $fromMail\n";
49       $headers .= "MIME-Version: 1.0\n";
50       $headers .= "X-Priority: 3 \n";
51       $headers .= "Content-Type: text/plain; charset=UTF-8\n";
52       $headers .= "Content-Transfer-Encoding: 8bit\n";
53       $headers .= "Return-Path: ".$fromMail."\n\n";
54
55 //     -=-=-=- MAIL SUBJECT
56 //    $subject = "Information Request from MSA Shipping - Contact Form";
57
58 //     -=-=-=- MAIL TEXT
59 //    echo $toMail.'<br>'.$subject.'<br>'.$message.'<br>'.$headers;
60 //    Kompletování emailu a odeslní
61       return mail($toMail, $subject, $message, $headers);
62
63    }
```

8.56.2.3 MailCtrlHelper::translateText ($ *text*, $ *translateArray*)

Metoda přeloží podle zadaného pole vnitřek emailu.

Zastaralé

Definice je uvedena na řádku 69 v souboru mailctrlhelper.class.php.

```
69                                                                  {
70       foreach ($translateArray as $translation) {
71          $text = str_replace($translation[self::MAIL_TPL_STRING_TRANS], $translation[self::MAIL_TPL_VAI
72       }
73       return $text;
74    }
```

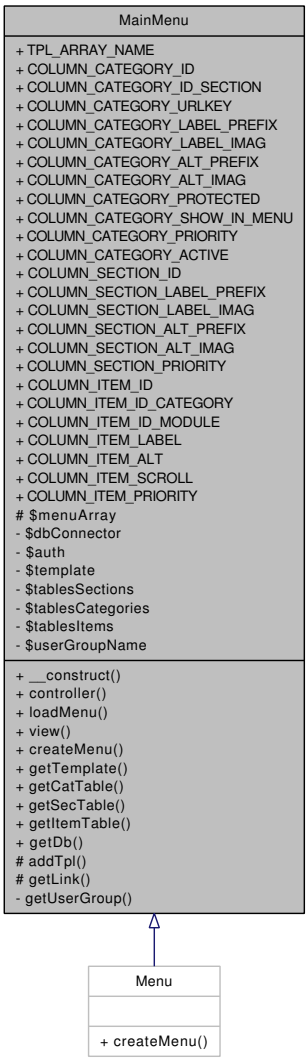
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/mailctrlhelper.class.php

8.57 Dokumentace třídy MainMenu

Abstraktní třída hlavního menu.

Diagram dědičnosti pro třídu MainMenu



Veřejné metody

- __construct (Db $dbConnector, Auth $auth)

  Konstruktor.

- **controller** ()
- loadMenu ($sqlSelect)

  Metoda provede načtení menu a přiřadí menu do pole menu pro zpracování.

- **view** ()
- createMenu ()

Abstraktní třída je volána při zobrazování menu je rozšířena při dědění.

- getTemplate ()

  Metoda vrací šablony a proměné menu.

- getCatTable ()

  Metoda vrací název tabulky s kategoriema.

- getSecTable ()

  Metoda vrací název tabulky se sekcema.

- getItemTable ()

  Metoda vrací název tabulky s itemy.

- getDb ()

  Metoda vrací objekt na db konektor.

Veřejné atributy

- const **TPL_ARRAY_NAME** = 'MAIN_MENU'
- const **COLUMN_CATEGORY_ID** = 'id_category'
- const **COLUMN_CATEGORY_ID_SECTION** = 'id_section'
- const **COLUMN_CATEGORY_URLKEY** = 'urlkey'
- const **COLUMN_CATEGORY_LABEL_PREFIX** = 'label_'
- const **COLUMN_CATEGORY_LABEL_IMAG** = 'clabel'
- const **COLUMN_CATEGORY_ALT_PREFIX** = 'alt_'
- const **COLUMN_CATEGORY_ALT_IMAG** = 'calt'
- const **COLUMN_CATEGORY_PROTECTED** = 'protected'
- const **COLUMN_CATEGORY_SHOW_IN_MENU** = 'show_in_menu'
- const **COLUMN_CATEGORY_PRIORITY** = 'priority'
- const **COLUMN_CATEGORY_ACTIVE** = 'active'
- const **COLUMN_SECTION_ID** = 'id_section'
- const **COLUMN_SECTION_LABEL_PREFIX** = 'label_'
- const **COLUMN_SECTION_LABEL_IMAG** = 'slabel'
- const **COLUMN_SECTION_ALT_PREFIX** = 'alt_'
- const **COLUMN_SECTION_ALT_IMAG** = 'calt'
- const **COLUMN_SECTION_PRIORITY** = 'priority'
- const COLUMN_ITEM_ID = 'id_item'

  Názvy sloupců v tabulce s itemy.

- const **COLUMN_ITEM_ID_CATEGORY** = 'id_category'
- const **COLUMN_ITEM_ID_MODULE** = 'id_module'
- const **COLUMN_ITEM_LABEL** = 'label'
- const **COLUMN_ITEM_ALT** = 'alt'
- const **COLUMN_ITEM_SCROLL** = 'calt'
- const **COLUMN_ITEM_PRIORITY** = 'priority'

Chráněné metody

- addTpl ($varName, $varValue)
  Metoda přiřadí proměnou do šablony.
- getLink ()
  Metoda vrací objekt pro práci s odkazy.

Chráněné atributy

- **$menuArray** = array()

8.57.1 Detailní popis

Abstraktní třída hlavního menu.

Třída slouží pro vytvoření hlavního menu aplikace z uživatelem definované třídy pro menu, a poskytuje základní přístup k prvkům menu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: mainmenu.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro vytvoření hlavního menu

Definice je uvedena na řádku 13 v souboru mainmenu.class.php.

8.57.2 Dokumentace konstruktoru a destruktoru

8.57.2.1 MainMenu::__construct (Db $ *dbConnector*, Auth $ *auth*)

Konstruktor.

Parametry:

Db – objekt databázového konektoru

Definice je uvedena na řádku 112 v souboru mainmenu.class.php.

Odkazuje se na AppCore::sysConfig().

```
112                                                 {
113         $this->dbConnector = $dbConnector;
114         $this->auth = $auth;
115         $this->template = new Template();
116 
117         $this->tablesSections = AppCore::sysConfig()->getOptionValue("section_table", "db_tables");
118         $this->tablesCategories = AppCore::sysConfig()->getOptionValue("category_table", "db_tables");
119         $this->tablesItems = AppCore::sysConfig()->getOptionValue("items_table", "db_tables");
120 
121         $this->userGroupName = $this->auth->getGroupName();
122     }
```

Tato funkce volá...

8.57.3 Dokumentace k metodám

8.57.3.1 MainMenu::loadMenu ($ *sqlSelect*)

Metoda provede načtení menu a přiřadí menu do pole menu pro zpracování.

Parametry:

DbInterface – objekt s popisem jak se má menu načíst

Definice je uvedena na řádku 155 v souboru mainmenu.class.php.

```
155                                     {
156 //    Přidání výběru jen na zvolenou skupinu
157       $sqlSelect = $sqlSelect->where(Rights::RIGHTS_GROUPS_TABLE_PREFIX.$this->getUserGroup()." LIKE \
158       $this->menuArray = $this->dbConnector->fetchAssoc($sqlSelect);
159
160       if(empty($this->menuArray)){
161          new CoreException(_("Nepodařilo se nahrát hlavní menu z databáze"), 2);
162       }
163    }
```

8.57.3.2 MainMenu::addTpl ($ *varName*, $ *varValue*) [protected]

Metoda přiřadí proměnou do šablony.

Parametry:

string – název proměnné
mixed – hodnota proměnné

Definice je uvedena na řádku 180 v souboru mainmenu.class.php.

```
180                                                 {
181       $this->template->addVar($varName, $varValue);
182    }
```

8.57.3.3 MainMenu::getTemplate ()

Metoda vrací šablony a proměné menu.

Návratová hodnota:

Template

Definice je uvedena na řádku 188 v souboru mainmenu.class.php.

```
188                                 {
189       return $this->template;
190    }
```

8.57.3.4 MainMenu::getLink () [protected]

Metoda vrací objekt pro práci s odkazy.

Návratová hodnota:

Links – objekt pro prái s odkazy

Definice je uvedena na řádku 198 v souboru mainmenu.class.php.

```
198                                 {
199         return new Links(true);
200     }
```

8.57.3.5 MainMenu::getCatTable ()

Metoda vrací název tabulky s kategoriema.

Návratová hodnota:

string – tabulka s kategoriemi

Definice je uvedena na řádku 207 v souboru mainmenu.class.php.

```
207                                 {
208         return $this->tablesCategories;
209     }
```

8.57.3.6 MainMenu::getSecTable ()

Metoda vrací název tabulky se sekcema.

Návratová hodnota:

string – tabulka se sekcema

Definice je uvedena na řádku 216 v souboru mainmenu.class.php.

```
216                                 {
217         return $this->tablesSections;
218     }
```

8.57.3.7 MainMenu::getItemTable ()

Metoda vrací název tabulky s itemy.

Návratová hodnota:

string – tabulka s itemy

Definice je uvedena na řádku 225 v souboru mainmenu.class.php.

```
225                                  {
226         return $this->tablesItems;
227     }
```

8.57.3.8 MainMenu::getDb ()

Metoda vrací objekt na db konektor.

Návratová hodnota:

DbInterface – objekt db konektoru

Definice je uvedena na řádku 233 v souboru mainmenu.class.php.

```
233                             {
234         return $this->dbConnector;
235     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/mainmenu.class.php

8.58 Dokumentace třídy Messages

Trida pro práci se zprávami.

Veřejné metody

- __construct ($saveTarget= 'session', $saveTargetName= 'message', $saveDefault=false)
  Konstruktor tridy.
- __destruct ()
  Destruktor - uloží zprávy pro uložení.
- addMessage ($messageText, $save=false)
  Funce pridava zpravu do pole se zpravami.
- isEmpty ()
  Funkce zjistuje jestli je pole se zpravami prazdne.
- getMessages ()
  Funce pro vraceni pole se zpravami.
- getMessagesPrint ()
  Funkce vypise pole se spravami na standartni vystup.

8.58.1 Detailní popis

Trida pro práci se zprávami.

Metoda slouží pro shromažďování zpráv, jak chbových tak informačních. Tyto zpráva jsou většinou vypisovány na výstup nebo ukládány.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: messages.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu hlášek //TODO refaktoring //TODO implementovat ukládání do souboru

Definice je uvedena na řádku 15 v souboru messages.class.php.

8.58.2 Dokumentace konstruktoru a destruktoru

8.58.2.1 Messages::__construct ($ *saveTarget* = `'session'`, $ *saveTargetName* = `'message'`, $ *saveDefault* = `false`)

Konstruktor tridy.

Parametry:

string – prefix pro výpis zpráv (option)
boolean – jestli se zpráva má ukládat a zobrazovat po obnovení stránky

Definice je uvedena na řádku 65 v souboru messages.class.php.

```
65                                                                                  {
66       if($saveTarget == 'session'){
67          $this->session = new Sessions();
68       }
69 //    Nastavení ukládání zpráv
70       $this->setSaveTarget($saveTarget, $saveTargetName);
71       $this->defaultSaveMessages = $saveDefault;
72       $this->getSavedMessages();
73    }
```

8.58.3 Dokumentace k metodám

8.58.3.1 Messages::addMessage ($ *messageText*, $ *save* = `false`)

Funce pridava zpravu do pole se zpravami.

Parametry:

String – text zpravy
boolean – jesli se má zpráva uložit

Definice je uvedena na řádku 106 v souboru messages.class.php.

```
106                                                  {
107       if($this->defaultSaveMessages OR $save){
108          if(!in_array($messageText, $this->messagesForSave)){
109             array_push($this->messagesForSave, $messageText);
110          }
111       } else {
112          if(!in_array($messageText, $this->messages)){
113             array_push($this->messages, $messageText);
114          }
115       }
116    }
```

8.58.3.2 Messages::isEmpty ()

Funkce zjistuje jestli je pole se zpravami prazdne.

Návratová hodnota:

boolean – true jestlize neni vlozena zadna zprava

Definice je uvedena na řádku 123 v souboru messages.class.php.

```
123                     {
124       if (sizeof($this->message) == 0){
125          return true;
126       } else {
127          return false;
128       }
129    }
```

8.58.3.3 Messages::getMessages ()

Funce pro vraceni pole se zpravami.

Návratová hodnota:

array – pole se zpravami

Sloučení uložených a zobrazovaných zpráv

Definice je uvedena na řádku 136 v souboru messages.class.php.

```
136                          {
137
138       $messages = $this->messages;
142       return $messages;
143    }
```

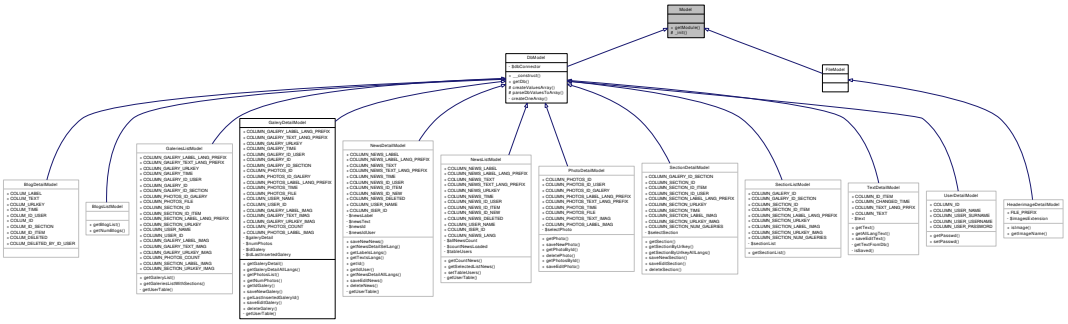
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/messages.class.php

8.59 Dokumentace třídy Model

Abstraktní třída pro Model.

Diagram dědičnosti pro třídu Model



Veřejné metody

- getModule ()
  Metoda vrací objekt modulu.

Chráněné metody

- _init ()
  Abstraktní metoda pro inicializaci modelu.

8.59.1 Detailní popis

Abstraktní třída pro Model.

Třída pro základní vytvoření objektu modelu, jak souborového tak databázového. Obsahuje pouze přístup k vybranému modulu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: model.class.php 3.0.55 26.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Abstraktní třída pro vytvoření modelu

Definice je uvedena na řádku 13 v souboru model.class.php.

8.59.2 Dokumentace k metodám

8.59.2.1 Model::getModule () [final]

Metoda vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 19 v souboru model.class.php.

Odkazuje se na AppCore::getSelectedModule().

Používá se v GaleryDetailModel::deleteGalery(), GaleryDetailModel::getGaleryDetail(), GaleryDetailModel::getGaleryDetailAllLangs(), GaleryDetailModel::getNumPhotos(), GaleryDetailModel::getPhotosList(), GaleryDetailModel::saveEditGalery() a GaleryDetailModel::saveNewGalery().

```
19                                      {
20          return AppCore::getSelectedModule();
21      }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/models/model.class.php

8.60 Dokumentace třídy Module

Třída pro práci s moduly v kategorii.

Veřejné metody

- __construct (stdClass $moduleObject, $dbTables)

  Konstruktor třídy pro práci s modulem.

- getDir ()

  Metoda vrací objekt s adresáři modulu.

- setId ($id)

  Funkce nastavi id modulu (item).

- getId ()

  Funkce vraci id modulu (item).

- setIdModule ($idModule)

  Funkce nastavi id modulu.

- getIdModule ()

  Funkce vraci id modulu.

- setName ($moduleName)

  Funkce nastavi jmeno module.

- getName ()

  Funkce vraci jmeno modulu.

- setLabel ($moduleLable)

  Funkce nastavi nazev modulu.

- getLabel ()

  Funkce vraci nazev modulu.

- setAlt ($moduleAlt)

  Funkce nastavi popis (alt) modulu.

- getAlt ()

  Funkce vraci popis (alt) modulu.

- setRecordsOnPage ($countRecords)

  Funkce nastavi počet záznamů na stránce.

- getRecordsOnPage ()

  Funkce vraci počet záznamů na stránce.

- setDataDir ($dir)

Funkce nastavi cestu k adresari s daty modulu.

- getDbTable ($tableNum=1)

  Funkce vraci jmeno databázové tabulky s daty modulu.

- setParams ($catParams)

  Funkce nastavi parametry modulu.

- getParams ()

  Metoda vraci parametry modulu.

- getSelectParam ($param)

  Metoda vraci hodnotu zvoleneho parametru modulu.

- setDbTables ($dbTables)

  Metoda nastaví tabulky modulu.

Veřejné atributy

- const **MODULE_PARAMS_SEPARATOR** = ';'

8.60.1 Detailní popis

Třída pro práci s moduly v kategorii.

Třída poskytuj ezákladní přístup k parametrům modulu. Pomocí ní lze zjišťovat např. použité databázové tabulky modulu, adresáře, a některé ostatní parametry,

Copyright (c) 2008 Jakub Matas

Verze:

$Id: module.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu modulů v kategorii

Definice je uvedena na řádku 13 v souboru module.class.php.

8.60.2 Dokumentace konstruktoru a destruktoru

8.60.2.1 Module::__construct (stdClass $ *moduleObject*, $ *dbTables*)

Konstruktor třídy pro práci s modulem.

Parametry:

integer – id modulu

string – nazev modulu

string – popis (label) modulu

string – popisný alt modulu

array – pole názvů db tabulek modulu

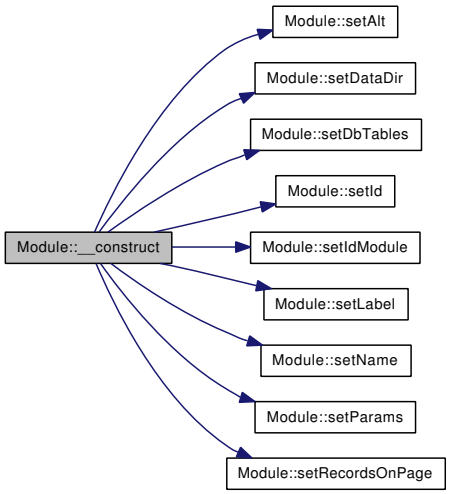
string – parametry modulu

Definice je uvedena na řádku 85 v souboru module.class.php.

Odkazuje se na setAlt(), setDataDir(), setDbTables(), setId(), setIdModule(), setLabel(), setName(), setParams() a setRecordsOnPage().

```
85                                                          {
86         $this->setId($moduleObject->id_item);
87         $this->setIdModule($moduleObject->id_module);
88         $this->setName($moduleObject->name);
89         $this->setDbTables($dbTables);
90         $this->setDataDir($moduleObject->datadir);
91         $this->setParams($moduleObject->params);
92         $this->setLabel($moduleObject->label);
93         $this->setAlt($moduleObject->alt);
94         $this->setRecordsOnPage($moduleObject->scroll);
95     }
```

Tato funkce volá...



8.60.3 Dokumentace k metodám

8.60.3.1 Module::getDir ()

Metoda vrací objekt s adresáři modulu.

Návratová hodnota:

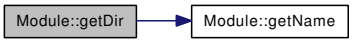
ModuleDirs – objek s adresáři modulu

Definice je uvedena na řádku 101 v souboru module.class.php.

Odkazuje se na getName().

```
101                             {
102         return new ModuleDirs($this->getName(), $this->dataDir);
103     }
```

Tato funkce volá...



8.60.3.2 Module::setId ($ *id*)

Funkce nastavi id modulu (item).

Parametry:

integer – id modulu (item)

Definice je uvedena na řádku 110 v souboru module.class.php.

Používá se v __construct().

```
111     {
112         $this->id = $id;
113     }
```

8.60.3.3 Module::getId ()

Funkce vraci id modulu (item).

Návratová hodnota:

integer – id modulu (item)

Definice je uvedena na řádku 120 v souboru module.class.php.

```
121     {
122         return $this->id;
123     }
```

8.60.3.4 Module::setIdModule ($ *idModule*)

Funkce nastavi id modulu.

Parametry:

integer – id modulu

Definice je uvedena na řádku 130 v souboru module.class.php.

Používá se v __construct().

```
131     {
132         $this->idModule = $idModule;
133     }
```

8.60.3.5 Module::getIdModule ()

Funkce vraci id modulu.

Návratová hodnota:

integer – id modulu

Definice je uvedena na řádku 140 v souboru module.class.php.

```
141     {
142        return $this->idModule;
143     }
```

8.60.3.6 Module::setName ($ *moduleName*)

Funkce nastavi jmeno module.

Parametry:

String – jmeno modulu

Definice je uvedena na řádku 150 v souboru module.class.php.

Používá se v __construct().

```
151     {
152        $this->moduleName = $moduleName;
153     }
```

8.60.3.7 Module::getName ()

Funkce vraci jmeno modulu.

Návratová hodnota:

String – jmeno modulu

Definice je uvedena na řádku 160 v souboru module.class.php.

Používá se v getDir().

```
161     {
162        return $this->moduleName;
163     }
```

8.60.3.8 Module::setLabel ($ *moduleLable*)

Funkce nastavi nazev modulu.

Parametry:

String – nazev modulu

Definice je uvedena na řádku 170 v souboru module.class.php.

Používá se v __construct().

```
171    {
172       $this->label = $moduleLable;
173    }
```

8.60.3.9 Module::getLabel ()

Funkce vraci nazev modulu.

Návratová hodnota:

String – nazev modulu

Definice je uvedena na řádku 180 v souboru module.class.php.

```
181    {
182       return $this->label;
183    }
```

8.60.3.10 Module::setAlt ($ *moduleAlt*)

Funkce nastavi popis (alt) modulu.

Parametry:

String – popis modulu

Definice je uvedena na řádku 190 v souboru module.class.php.

Používá se v __construct().

```
191    {
192       $this->alt = $moduleAlt;
193    }
```

8.60.3.11 Module::getAlt ()

Funkce vraci popis (alt) modulu.

Návratová hodnota:

String – popis modulu

Definice je uvedena na řádku 200 v souboru module.class.php.

```
201    {
202       return $this->alt;
203    }
```

8.60.3.12 Module::setRecordsOnPage ($ *countRecords*)

Funkce nastavi počet záznamů na stránce.

Parametry:

integer – počet záznamů

Definice je uvedena na řádku 210 v souboru module.class.php.

Používá se v __construct().

```
211    {
212       $this->recordsOnPage = $countRecords;
213    }
```

8.60.3.13 Module::getRecordsOnPage ()

Funkce vraci počet záznamů na stránce.

Návratová hodnota:

integer – počet záznamů

Definice je uvedena na řádku 220 v souboru module.class.php.

```
221    {
222       return $this->recordsOnPage;
223    }
```

8.60.3.14 Module::setDataDir ($ *dir*)

Funkce nastavi cestu k adresari s daty modulu.

Parametry:

String – jmeno adresare

Definice je uvedena na řádku 230 v souboru module.class.php.

Používá se v __construct().

```
231    {
232       $this->dataDir = $dir;
233    }
```

8.60.3.15 Module::getDbTable ($ *tableNum* = 1)

Funkce vraci jmeno databázové tabulky s daty modulu.

Parametry:

integer – cislo tabulky

Návratová hodnota:

String – jmeno adresare

Definice je uvedena na řádku 241 v souboru module.class.php.

```
242    {
243       if(isset($this->dbTables[$tableNum])){
244          return $this->dbTables[$tableNum];
245       } else {
246          return false;
247       }
248    }
```

8.60.3.16 Module::setParams ($ *catParams*)

Funkce nastavi parametry modulu.

Parametry:

String – parametry

Definice je uvedena na řádku 255 v souboru module.class.php.

Používá se v __construct().

```
256    {
257       if ($catParams != null){
258          $arrayValues = array();
259          $arrayValues = explode(self::MODULE_PARAMS_SEPARATOR, $catParams);
260
261          foreach ($arrayValues as $value) {
262             $tmpArrayValue = explode("=", $value);
263             $this->params[$tmpArrayValue[0]]=$tmpArrayValue[1];
264          }
265       }
266    }
```

8.60.3.17 Module::getParams ()

Metoda vraci parametry modulu.

Návratová hodnota:

Array – parametry

Definice je uvedena na řádku 273 v souboru module.class.php.

```
274    {
275       return $this->params;
276    }
```

8.60.3.18 Module::getSelectParam ($ *param*)

Metoda vraci hodnotu zvoleneho parametru modulu.

Návratová hodnota:

string – parametr

Definice je uvedena na řádku 283 v souboru module.class.php.

```
284    {
285       if(isset($this->params[$param])){
286          return $this->params[$param];
287       } else {
288          return null;
289       }
290    }
```

8.60.3.19 Module::setDbTables ($ *dbTables*)

Metoda nastaví tabulky modulu.

Parametry:

array – pole s tabulkama

Definice je uvedena na řádku 297 v souboru module.class.php.

Používá se v __construct().

```
297                                        {
298       //TODO není iplementována optimálně
299       $this->dbTables = $dbTables;
300    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/module.class.php

8.61 Dokumentace třídy ModuleAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu ModuleAction

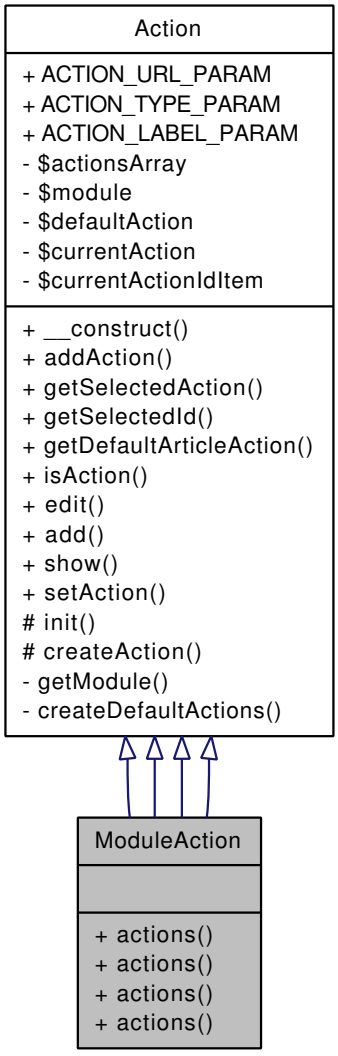
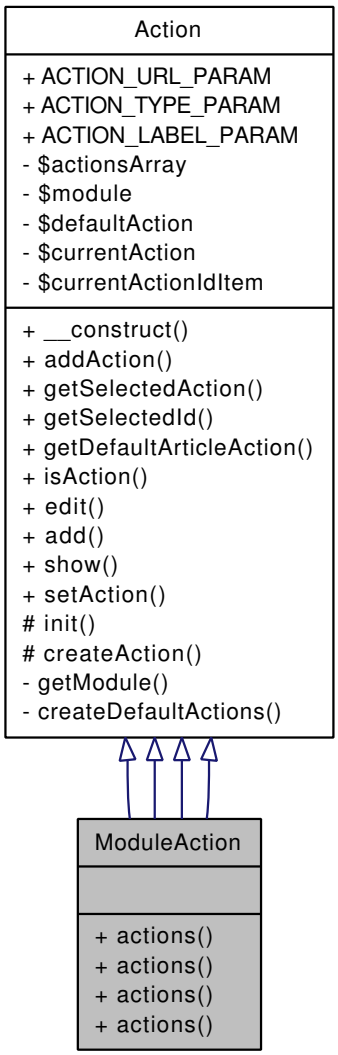


Diagram tříd pro ModuleAction:



Veřejné metody

- **actions** ()
- **actions** ()
- **actions** ()
- **actions** ()

8.61.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujících souborů:

- /home/cuba/work-net/vve3_2/modules/changes/action.class.php

- /home/cuba/work-net/vve3_2/modules/example_module/action.class.php
- /home/cuba/work-net/vve3_2/modules/reservation/action.class.php
- /home/cuba/work-net/vve3_2/modules/users/action.class.php

8.62 Dokumentace třídy ModuleDirs

Třída pro obsluhu adresářů modulu.

Veřejné metody

- __construct ($moduleDir, $moduleDataDir)

  Konstruktor vytvoří základní adresářovou strukturu.

- getMainDir ($withRelativePrefix=true)

  Metoda vrací adresář k modulu.

- getModelsDir ()

  Metoda vrací adresář s modely.

- getDataDir ($withRelPrefix=true)

  Metoda vrací cestu k datovému adresáři modulu.

- getStylesheetsDir ($withPrefix=false)

  Metody vrací cestu k adresáři css souborů modulu.

- getJavaScriptsDir ($withPrefix=true)

  Metoda vrací adresář k javascript souborům modulu.

- getTemplatesDir ($withPrefix=true)

  Metoda vrací adresář k šablonám modulu.

Statické veřejné metody

- static setWebDir ($webDir)

  Statická metoda nastaví hlavní adresář webu.

- static setWebDataDir ($dataDir)

  Statická metoda nastaví hlavní datový adresář webu.

Veřejné atributy

- const **MODULES_MAIN_DIR** = 'modules'
- const **MODELS_DIR** = 'models'
- const **TEMPLATES_DIR** = 'templates'
- const **TEMPLATES_INDEX_DIR** = '.'
- const **STYLESHEETS_DIR** = 'stylesheets'
- const **JAVASCRIPTS_DIR** = 'javascripts'
- const **DIR_SEPARATOR** = '/'

8.62.1 Detailní popis

Třída pro obsluhu adresářů modulu.

Třída slouží pro přístup k jednotlivým adresářům modulu. Pracuje jak s datovým, tak s hlavním adresářem modulu, ale i s adresáři stylesheetu a šablon modulu.

Verze:

$Id: moduledirs.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu adresářů modulu

Definice je uvedena na řádku 13 v souboru moduledirs.class.php.

8.62.2 Dokumentace konstruktoru a destruktoru

8.62.2.1 ModuleDirs::__construct ($ *moduleDir*, $ *moduleDataDir*)

Konstruktor vytvoří základní adresářovou strukturu.

Parametry:

string – adresář modulu (název)

string – datový adresář modulu

Definice je uvedena na řádku 90 v souboru moduledirs.class.php.

```
90                                                  {
91         $this->moduleDir = $moduleDir;
92         $this->moduleDataDir = $moduleDataDir;
93     }
```

8.62.3 Dokumentace k metodám

8.62.3.1 static ModuleDirs::setWebDir ($ *webDir*) [static]

Statická metoda nastaví hlavní adresář webu.

Parametry:

string – adresář webu

Definice je uvedena na řádku 100 v souboru moduledirs.class.php.

```
100                                             {
101         ModuleDirs::$mainWebDir = $webDir;
102     }
```

8.62.3.2 static ModuleDirs::setWebDataDir ($ *dataDir*) [static]

Statická metoda nastaví hlavní datový adresář webu.

Parametry:

string – datový adresář webu

Definice je uvedena na řádku 118 v souboru moduledirs.class.php.

```
118                                                 {
119       ModuleDirs::$mainDataDir = $dataDir;
120     }
```

8.62.3.3 ModuleDirs::getMainDir ($ *withRelativePrefix* = true)

Metoda vrací adresář k modulu.

Parametry:

boolean – jestli se má vráti i cesta s prefixe (nutné pro relativní přístup)

Návratová hodnota:

string – cesta modulu

Definice je uvedena na řádku 136 v souboru moduledirs.class.php.

Používá se v getJavaScriptsDir(), getModelsDir(), getStylesheetsDir() a getTemplatesDir().

```
136                                                         {
137       if($withRelativePrefix){
138          return ModuleDirs::getWebDir().self::MODULES_MAIN_DIR.self::DIR_SEPARATOR.$this->moduleDir.se
139       } else {
140          return self::MODULES_MAIN_DIR.self::DIR_SEPARATOR.$this->moduleDir.self::DIR_SEPARATOR;
141       }
142     }
```

8.62.3.4 ModuleDirs::getDataDir ($ *withRelPrefix* = true)

Metoda vrací cestu k datovému adresáři modulu.

Parametry:

boolean[option] – jestli má být vrácen s označením relativní cesty

Návratová hodnota:

string – cesta k adresáři

Definice je uvedena na řádku 156 v souboru moduledirs.class.php.

```
156                                                 {
157       if($this->moduleDataDir == null){
158          return false;
159       } else {
160             if($withRelPrefix){
161                 return self::getWebDir().self::getWebDataDir().self::DIR_SEPARATOR.$this->moduleDataDi
162             } else {
163                 return self::getWebDataDir().self::DIR_SEPARATOR.$this->moduleDataDir.self::DIR_SEPARA
164             }
165       }
166    }
```

8.62.3.5 ModuleDirs::getStylesheetsDir ($ *withPrefix* = `false`)

Metody vrací cestu k adresáři css souborů modulu.

Návratová hodnota:

string – cesta k css souborům

Definice je uvedena na řádku 172 v souboru moduledirs.class.php.

Odkazuje se na getMainDir().

```
172                                                     {
173       if($withPrefix){
174          return $this->getMainDir().self::STYLESHEETS_DIR.self::DIR_SEPARATOR;
175       } else {
176          return $this->getMainDir(false).self::STYLESHEETS_DIR.self::DIR_SEPARATOR;
177       }
178
179    }
```

Tato funkce volá...

8.62.3.6 ModuleDirs::getJavaScriptsDir ($ *withPrefix* = `true`)

Metoda vrací adresář k javascript souborům modulu.

Návratová hodnota:

string – cesta k adresáři s javascripty

Definice je uvedena na řádku 185 v souboru moduledirs.class.php.

Odkazuje se na getMainDir().

```
185                                                   {
186       if($withPrefix){
187             return $this->getMainDir().self::JAVASCRIPTS_DIR.self::DIR_SEPARATOR;
188         } else {
189             return $this->getMainDir(false).self::JAVASCRIPTS_DIR.self::DIR_SEPARATOR;
190         }
191    }
```

Tato funkce volá...

8.62.3.7 ModuleDirs::getTemplatesDir ($ *withPrefix* = `true`)

Metoda vrací adresář k šablonám modulu.

Parametry:

boolean – jestli se má vráti i cesta s prefixe (nutné pro relativní přístup)

Návratová hodnota:

string – adresář k šablonám modulu

Definice je uvedena na řádku 199 v souboru moduledirs.class.php.

Odkazuje se na getMainDir().

```
199                                                 {
200       if($withPrefix){
201          return self::TEMPLATES_INDEX_DIR.$this->getMainDir().self::TEMPLATES_DIR.self::DIR_SEPARATOR;
202       } else {
203          return $this->getMainDir(false).self::TEMPLATES_DIR.self::DIR_SEPARATOR;
204       }
205    }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/moduledirs.class.php

8.63 Dokumentace třídy ModuleRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu ModuleRoutes

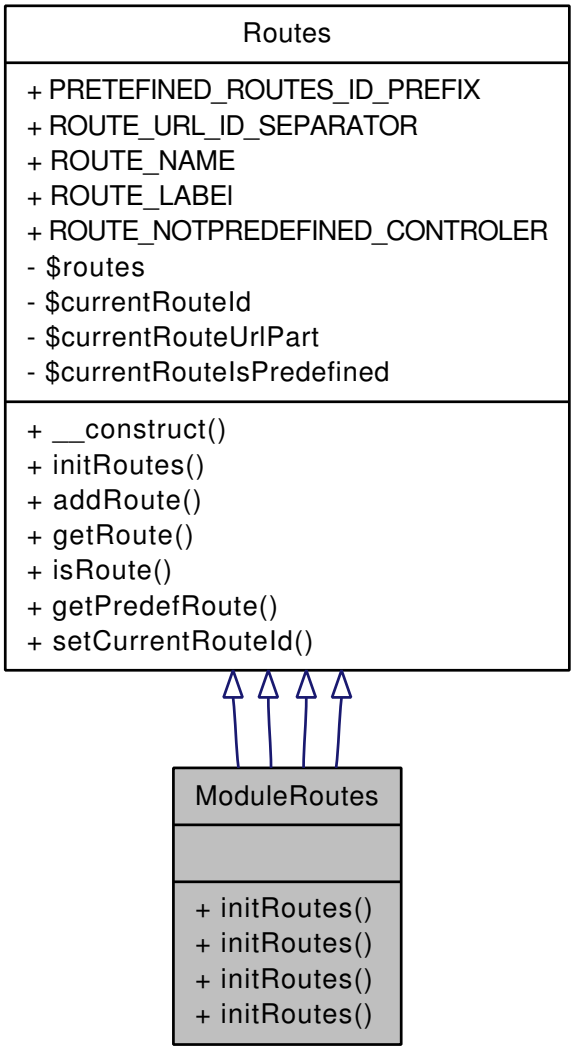
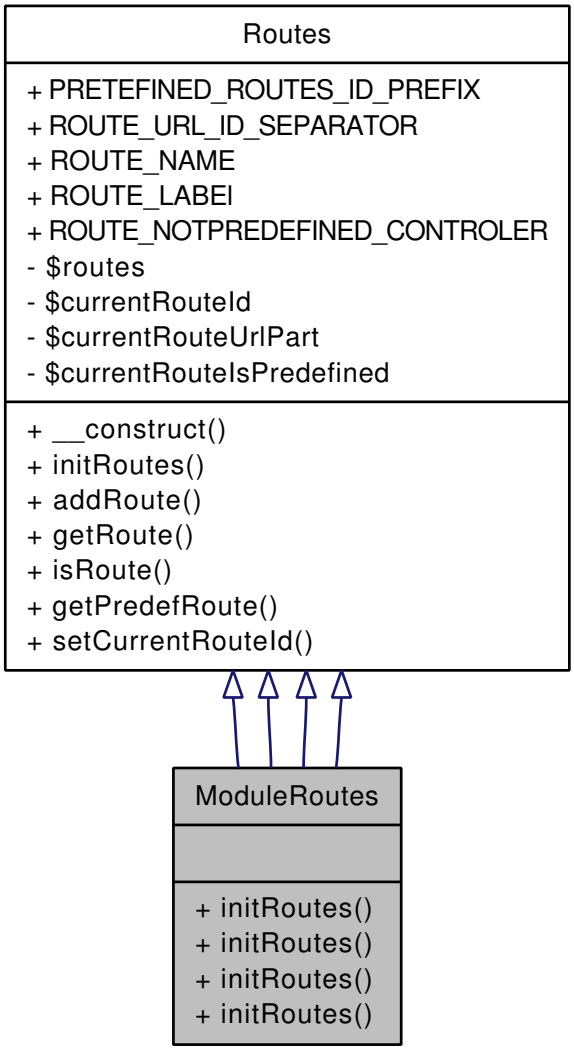


Diagram tříd pro ModuleRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

- initRoutes ()

  Metoda, která nastavuje cesty.

- initRoutes ()

  Metoda, která nastavuje cesty.

- initRoutes ()

  Metoda, která nastavuje cesty.

8.63.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujících souborů:

- /home/cuba/work-net/vve3_2/modules/changes/routes.class.php
- /home/cuba/work-net/vve3_2/modules/example_module/routes.class.php
- /home/cuba/work-net/vve3_2/modules/reservation/routes.class.php
- /home/cuba/work-net/vve3_2/modules/users/routes.class.php

8.64 Dokumentace třídy Mysql_Db_Delete

Třída pro odstraňování záznamů v MySQL DB.

Diagram dědičnosti pro třídu Mysql_Db_Delete

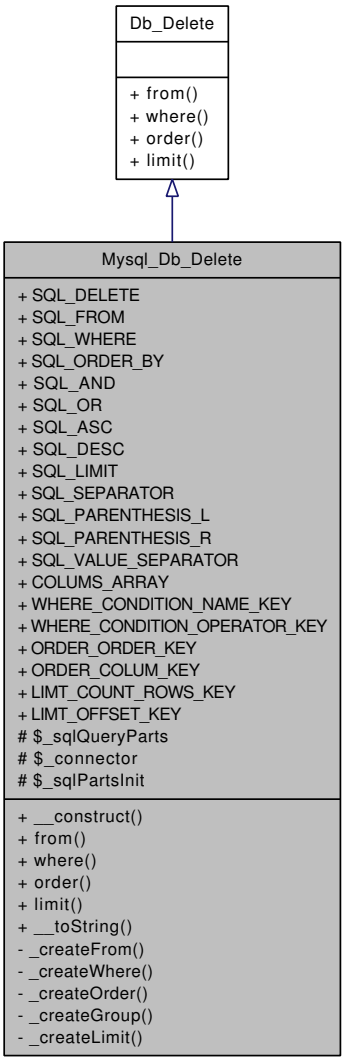
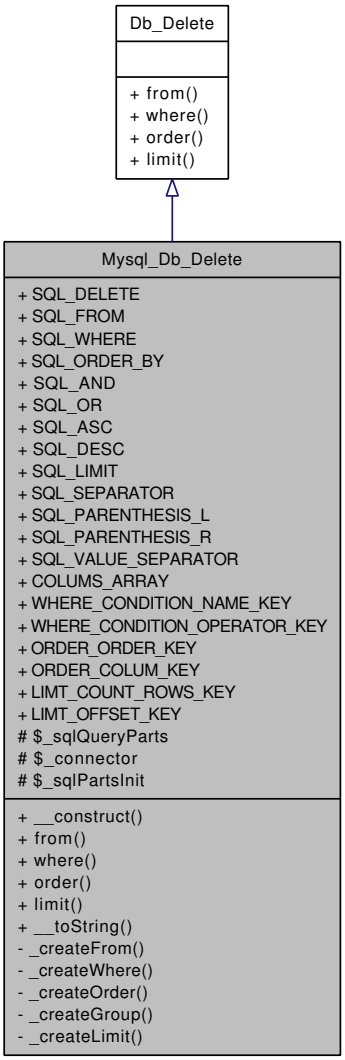


Diagram tříd pro Mysql_Db_Delete:



Veřejné metody

- __construct (Db $conector)

  Konstruktor nastaví základní parametry a odkaz na dbkonektor.

- from ($table, $columsArray=null)

  Metoda nastavuje z které tabulky se bude mazat klauzule FROM.

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- order ($colum, $order=self::SQL_ASC)

  Metoda přiřadí řazení sloupcu v SQL dotazu.

- limit ($rowCount, $offset)

Metoda přidá do SQL dotazu klauzuli LIMIT.

- __toString ()
  Metoda převede objekt na řetězec.

Veřejné atributy

- const **SQL_DELETE** = 'DELETE'
- const **SQL_FROM** = 'FROM'
- const **SQL_WHERE** = 'WHERE'
- const **SQL_ORDER_BY** = 'ORDER BY'
- const **SQL_AND** = 'AND'
- const **SQL_OR** = 'OR'
- const **SQL_ASC** = 'ASC'
- const **SQL_DESC** = 'DESC'
- const **SQL_LIMIT** = 'LIMIT'
- const **SQL_SEPARATOR** = ' '
- const **SQL_PARENTHESIS_L** = '('
- const **SQL_PARENTHESIS_R** = ')'
- const **SQL_VALUE_SEPARATOR** = ','
- const **COLUMS_ARRAY** = 'COLUMS'
- const **WHERE_CONDITION_NAME_KEY** = 'condition'
- const **WHERE_CONDITION_OPERATOR_KEY** = 'operator'
- const **ORDER_ORDER_KEY** = 'ORDER'
- const **ORDER_COLUM_KEY** = 'colum'
- const **LIMT_COUNT_ROWS_KEY** = 'limit_count'
- const **LIMT_OFFSET_KEY** = 'limit_offset'

Chráněné atributy

- **$_sqlQueryParts** = array()
- **$_connector** = null

Statické chráněné atributy

- static $_sqlPartsInit
  Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

8.64.1 Detailní popis

Třída pro odstraňování záznamů v MySQL DB.

Třída obsahuje implementaci metody delete z db.interfacu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: delete.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro mazání záznamů

Definice je uvedena na řádku 13 v souboru delete.class.php.

8.64.2 Dokumentace konstruktoru a destruktoru

8.64.2.1 Mysql_Db_Delete::__construct (Db $ *conector*)

Konstruktor nastaví základní parametry a odkaz na dbkonektor.

Parametry:

Db – konektor databáze

Definice je uvedena na řádku 74 v souboru delete.class.php.

```
74                                              {
75 //    inicializace do zakladni podoby;
76       $this->_connector = $conector;
77       $this->_sqlQueryParts = self::$_sqlPartsInit;
78    }
```

8.64.3 Dokumentace k metodám

8.64.3.1 Mysql_Db_Delete::from ($ *table*, $ *columsArray* = `null`)

Metoda nastavuje z které tabulky se bude mazat klauzule FROM.

Parametry:

string/array – tabulka ze které se bude vybírat, u pole index označuje alias tabulky

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementuje stejnojmenný prvek z Db_Delete.

Definice je uvedena na řádku 88 v souboru delete.class.php.

```
88                                                  {
89       $this->_sqlQueryParts[self::SQL_FROM] = $table;
90       return $this;
91    }
```

8.64.3.2 Mysql_Db_Delete::where ($ *condition*, $ *operator* = `self::SQL_AND`)

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka

string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Delete – objekt Db_Delete

Plánované úpravy

dodělat aby se doplňovali magické uvozovky do lauzule

Reimplementuje stejnojmenný prvek z Db_Delete.

Definice je uvedena na řádku 102 v souboru delete.class.php.

```
102                                                                 {
103       $tmpArray = array();
104
105       if($operator != self::SQL_AND AND $operator != self::SQL_OR){
106          $operator = self::SQL_AND;
107          echo $operator;
108       }
109
110       $tmpArray[self::WHERE_CONDITION_NAME_KEY] = $condition;
111       $tmpArray[self::WHERE_CONDITION_OPERATOR_KEY] = strtoupper($operator);
112
113 //    pokud není vytvořeno pole podmínek -> vytvoř ho
114       if(!is_array($this->_sqlQueryParts[self::SQL_WHERE])){
115          $this->_sqlQueryParts[self::SQL_WHERE] = array();
116       }
117       array_push($this->_sqlQueryParts[self::SQL_WHERE], $tmpArray);
118       return $this;
119    }
```

8.64.3.3 Mysql_Db_Delete::order ($ *colum*, $ *order* = `self::SQL_ASC`)

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementuje stejnojmenný prvek z Db_Delete.

Definice je uvedena na řádku 129 v souboru delete.class.php.

```
129                                                         {
130       $order = strtoupper($order);
131       if(!is_array($this->_sqlQueryParts[self::ORDER_ORDER_KEY])){
132             $this->_sqlQueryParts[self::ORDER_ORDER_KEY] = array();
133          }
134
135       $columArray = array();
136       if($order == self::SQL_DESC){
137          $columArray[self::ORDER_COLUM_KEY] = $colum;
```

```
138           $columArray[self::ORDER_ORDER_KEY] = self::SQL_DESC;
139       } else {
140           $columArray[self::ORDER_COLUM_KEY] = $colum;
141           $columArray[self::ORDER_ORDER_KEY] = self::SQL_ASC;
142       }
143       array_push($this->_sqlQueryParts[self::ORDER_ORDER_KEY], $columArray);
144       return $this;
145   }
```

8.64.3.4 Mysql_Db_Delete::limit ($ *rowCount*, $ *offset*)

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Delete – objekt Db_Delete

Reimplementuje stejnojmenný prvek z Db_Delete.

Definice je uvedena na řádku 154 v souboru delete.class.php.

```
154                                             {
155       $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_COUNT_ROWS_KEY] = $rowCount;
156       $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_OFFSET_KEY] = $offset;
157
158       return $this;
159   }
```

8.64.3.5 Mysql_Db_Delete::__toString ()

Metoda převede objekt na řetězec.

Návratová hodnota:

string – objekt jako řetězec

Definice je uvedena na řádku 254 v souboru delete.class.php.

```
255     {
256         $sql = self::SQL_DELETE;
257         foreach ($this->_sqlQueryParts as $partKey => $partValue) {
258          $createMethod = '_create' . ucfirst($partKey);
259          if(method_exists($this, $createMethod)){
260             $sql .= $this->$createMethod();
261          }
262         }
263         return $sql;
264     }
```

8.64.4 Dokumentace k datovým členům

8.64.4.1 Mysql_Db_Delete::$_sqlPartsInit [static, protected]

Initializer:

```
 array(
//                                  self::COLUMS_ARRAY         => array(),
                                    self::SQL_FROM          => array(),
                                    self::SQL_WHERE            => array(),
                                    self::ORDER_ORDER_KEY      => array(),
                                    self::SQL_LIMIT            => array())
```

Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

Definice je uvedena na řádku 49 v souboru delete.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/mysql/delete.class.php

8.65 Dokumentace třídy Mysql_Db_Insert

Třída pro vkládání záznamů do MySQL DB.

Diagram dědičnosti pro třídu Mysql_Db_Insert

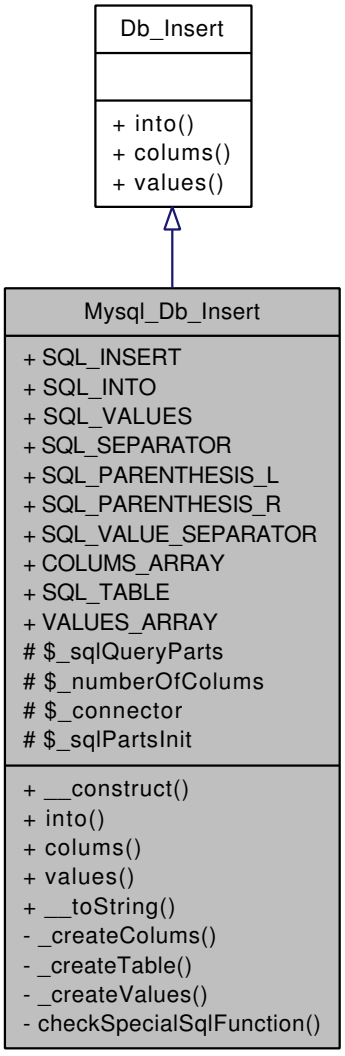
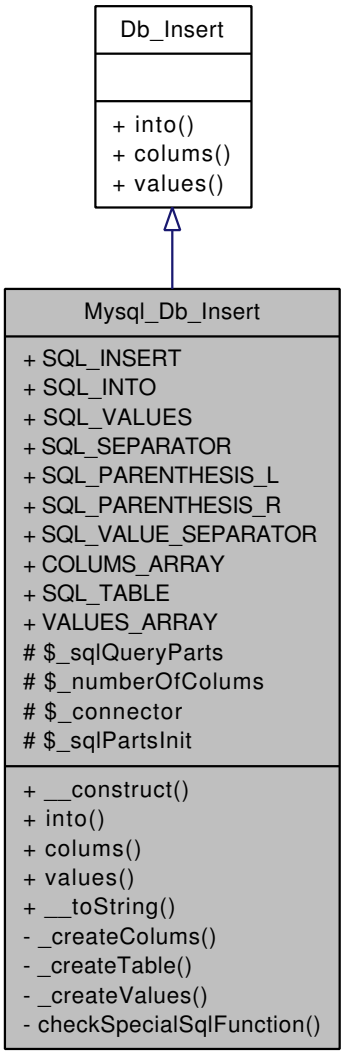


Diagram tříd pro Mysql_Db_Insert:



Veřejné metody

- __construct (Db $conector)

  Konstruktor třídy vytváří objet INSERT pro update databáze.

- into ($table)

  Metoda nastavuje do které tabulky se bude zapisovat klauzule INTO.

- colums ($colums)

  Metody vatváří sloupce, které se budou zapisovat.

- values ($value)

  Metoda přiřadí hodnoty sloupscům.

- __toString ()

Metoda převede objekt na řetězec.

Veřejné atributy

- const **SQL_INSERT** = 'INSERT'
- const **SQL_INTO** = 'INTO'
- const **SQL_VALUES** = 'VALUES'
- const **SQL_SEPARATOR** = ' '
- const **SQL_PARENTHESIS_L** = '('
- const **SQL_PARENTHESIS_R** = ')'
- const **SQL_VALUE_SEPARATOR** = ','
- const **COLUMS_ARRAY** = 'COLUMS'
- const **SQL_TABLE** = 'TABLE'
- const **VALUES_ARRAY** = 'VALUES'

Chráněné atributy

- **$_sqlQueryParts** = array()
- **$_numberOfColums** = null
- **$_connector** = null

Statické chráněné atributy

- static $_sqlPartsInit

  Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

8.65.1 Detailní popis

Třída pro vkládání záznamů do MySQL DB.

Třída obsahuje implementaci metody insert z db.interfacu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: insert.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro vkládaání záznamů do MySQL DB

Definice je uvedena na řádku 14 v souboru insert.class.php.

8.65.2 Dokumentace konstruktoru a destruktoru

8.65.2.1 Mysql_Db_Insert::__construct (Db $ *conector*)

Konstruktor třídy vytváří objet INSERT pro update databáze.

Parametry:

Db – objekt db konektoru

Definice je uvedena na řádku 68 v souboru insert.class.php.

```
68                                          {
69 //    inicializace do zakladni podoby;
70       $this->_connector = $conector;
71       $this->_sqlQueryParts = self::$_sqlPartsInit;
72    }
```

8.65.3 Dokumentace k metodám

8.65.3.1 Mysql_Db_Insert::into ($ *table*)

Metoda nastavuje do které tabulky se bude zapisovat klauzule INTO.

Parametry:

string – tabulka do které se bude zapisovat

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementuje stejnojmenný prvek z Db_Insert.

Definice je uvedena na řádku 82 v souboru insert.class.php.

```
82                                  {
83       $this->_sqlQueryParts[self::SQL_TABLE] = $table;
84       return $this;
85    }
```

8.65.3.2 Mysql_Db_Insert::colums ($ *colums*)

Metody vatváří sloupce, které se budou zapisovat.

Parametry:

mixed – sloupce (string nebo array)

string – sloupce (neomezený počet parametrů)

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementuje stejnojmenný prvek z Db_Insert.

Definice je uvedena na řádku 95 v souboru insert.class.php.

```
95                                  {
96       $countColums = func_num_args();
97       if(!is_array($colums)){
98
99          $this->_numberOfColums = $countColums;
100
101          $arrayColums = func_get_args();
102          //Doplnění pole se sloupcy do pole sloupců
103          $this->_sqlQueryParts[self::COLUMS_ARRAY] = $arrayColums;
104       } else if(is_array($colums) AND $countColums == 1) {
105          $this->_numberOfColums = count($colums);
106          $this->_sqlQueryParts[self::COLUMS_ARRAY] = $colums;
107       }
108
109       return $this;
110    }
```

8.65.3.3 Mysql_Db_Insert::values ($ *value*)

Metoda přiřadí hodnoty sloupscům.

Parametry:

string – hodnota sloupce (proměnný počet parametrů)

Návratová hodnota:

Db_Insert – objekt Db_Insert

Reimplementuje stejnojmenný prvek z Db_Insert.

Definice je uvedena na řádku 119 v souboru insert.class.php.

```
119                                 {
120       //TODO možná vyřešit naplňováním null na nezadané sloupce
121       if(is_array($value)){
122
123          reset($value);
124          if(!is_array(current($value))){
125             if(count($value) != $this->_numberOfColums){
126                new CoreException(_("Nesouhlasí počet zadaných sloupců a hodnot"));
127             } else {
128                array_push($this->_sqlQueryParts[self::VALUES_ARRAY], $value);
129             }
130          } else {
131             foreach ($value as $val) {
132                if(count($val) != $this->_numberOfColums){
133                   new CoreException(_("Nesouhlasí počet zadaných sloupců a hodnot"));
134                } else {
135                   array_push($this->_sqlQueryParts[self::VALUES_ARRAY], $val);
136                }
137             }
138          }
139       } else if(func_num_args() != $this->_numberOfColums){
140          new CoreException(_("Nesouhlasí počet zadaných sloupců a hodnot"));
141       }
142       else {
143          $valuesArray = func_get_args();
```

```
144           array_push($this->_sqlQueryParts[self::VALUES_ARRAY], $valuesArray);
145       }
146       return $this;
147     }
```

8.65.3.4 Mysql_Db_Insert::__toString ()

Metoda převede objekt na řetězec.

Návratová hodnota:

string – objekt jako řetězec

Definice je uvedena na řádku 224 v souboru insert.class.php.

```
225     {
226         $sql = self::SQL_INSERT;
227         foreach ($this->_sqlQueryParts as $partKey => $partValue) {
228          $createMethod = '_create' . ucfirst($partKey);
229          if(method_exists($this, $createMethod)){
230             $sql .= $this->$createMethod();
231          }
232          ;
233         }
234         return $sql;
235     }
```

8.65.4 Dokumentace k datovým členům

8.65.4.1 Mysql_Db_Insert::$_sqlPartsInit [static, protected]

Initializer:

```
array(self::SQL_TABLE           => null,
                                self::COLUMS_ARRAY         => array(),
                                self::VALUES_ARRAY         => array())
```

Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

Definice je uvedena na řádku 40 v souboru insert.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/mysql/insert.class.php

8.66 Dokumentace třídy Mysql_Db_Select

Třída pro výběr záznamů z MySQL DB.

Diagram dědičnosti pro třídu Mysql_Db_Select

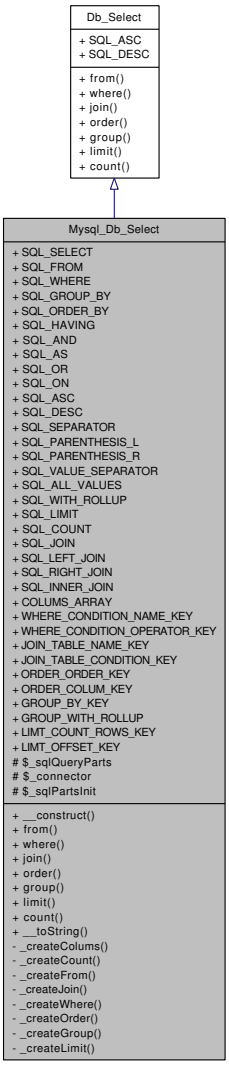
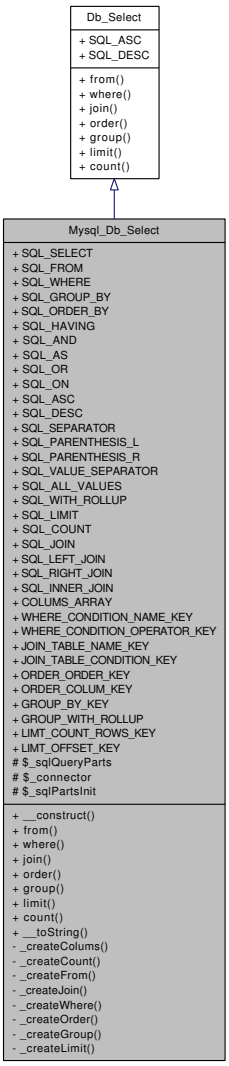


Diagram tříd pro Mysql_Db_Select:



Veřejné metody

- **__construct** (Db $conector)
- from ($tableArray, $columsArray="*")

  Metoda nastavuje z které tabulky se bude načítat klauzule FROM.

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- join ($tableArray, $condition, $joinType=null, $columsArray="*")

  Metody vytvoří část pro klauzuli JOIN U pole označuje index alias prvku.

- order ($colum, $order=self::SQL_ASC)

  Metoda přiřadí řazení sloupcu v SQL dotazu.

- group ($colum, $withRollup=false)

  Metoda přiřadí slouření sloupců v SQL dotazu pomocí klauzule GROUP BY.

- limit ($rowCount, $offset)

  Metoda přidá do SQL dotazu klauzuli LIMIT.

- count ($alias=null, $colum=self::SQL_ALL_VALUES)

  Metoda přidává do dotazu sloupce s počem záznamů.

- __toString ()

  Metoda převede objekt na řetězec.

Veřejné atributy

- const **SQL_SELECT** = 'SELECT'
- const **SQL_FROM** = 'FROM'
- const **SQL_WHERE** = 'WHERE'
- const **SQL_GROUP_BY** = 'GROUP BY'
- const **SQL_ORDER_BY** = 'ORDER BY'
- const **SQL_HAVING** = 'HAVING'
- const **SQL_AND** = 'AND'
- const **SQL_AS** = 'AS'
- const **SQL_OR** = 'OR'
- const **SQL_ON** = 'ON'
- const SQL_ASC = 'ASC'

  Vybrané konstanty pro SQL dotazy.

- const **SQL_DESC** = 'DESC'
- const **SQL_SEPARATOR** = ' '
- const **SQL_PARENTHESIS_L** = '('
- const **SQL_PARENTHESIS_R** = ')'
- const **SQL_VALUE_SEPARATOR** = ','
- const **SQL_ALL_VALUES** = '*'
- const **SQL_WITH_ROLLUP** = 'WITH ROLLUP'
- const **SQL_LIMIT** = 'LIMIT'
- const **SQL_COUNT** = 'COUNT'
- const **SQL_JOIN** = 'JOIN'
- const **SQL_LEFT_JOIN** = 'LEFT JOIN'
- const **SQL_RIGHT_JOIN** = 'RIGHT JOIN'
- const **SQL_INNER_JOIN** = 'INNER JOIN'
- const **COLUMS_ARRAY** = 'COLUMS'
- const **WHERE_CONDITION_NAME_KEY** = 'condition'
- const **WHERE_CONDITION_OPERATOR_KEY** = 'operator'
- const **JOIN_TABLE_NAME_KEY** = 'name'
- const **JOIN_TABLE_CONDITION_KEY** = 'condition'
- const **ORDER_ORDER_KEY** = 'ORDER'
- const **ORDER_COLUM_KEY** = 'colum'
- const **GROUP_BY_KEY** = 'GROUP'
- const **GROUP_WITH_ROLLUP** = 'w_rolupp'
- const **LIMT_COUNT_ROWS_KEY** = 'limit_count'
- const **LIMT_OFFSET_KEY** = 'limit_offset'

Chráněné atributy

- **$_sqlQueryParts** = array()
- **$_connector** = null

Statické chráněné atributy

- static $_sqlPartsInit

  Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

8.66.1 Detailní popis

Třída pro výběr záznamů z MySQL DB.

Třída obsahuje implementaci metody select z db-interfacu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: select.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro výběr záznamů

Definice je uvedena na řádku 14 v souboru select.class.php.

8.66.2 Dokumentace k metodám

8.66.2.1 Mysql_Db_Select::from ($ *tableArray*, $ *columsArray* = "*")

Metoda nastavuje z které tabulky se bude načítat klauzule FROM.

Parametry:

string/array – tabulka ze které se bude vybírat, u pole index označuje alias tabulky

string/array – sloupce, které se mají vybrat

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 104 v souboru select.class.php.

```
104                                                     {
105         if(is_array($tableArray)){
106 //          foreach ($tablesArray as $tableAlias => $table) {
107                 $this->_sqlQueryParts[ self::SQL_FROM][key($tableArray)] = $tableArray[key($tableArray)];
108 //          }
109             $tableAlias = key($tableArray);
110         } else {
```

```
111           $tableAlias = $tableArray[0].$tableArray[1].$tableArray[2];
112           $this->_sqlQueryParts[self::SQL_FROM][$tableAlias] = $tableArray;
113        }
114
115        if(!is_array($columsArray)){
116           $columsArray = array($columsArray);
117        }
118        $this->_sqlQueryParts[self::COLUMS_ARRAY][$tableAlias] = $columsArray;
119
120        return $this;
121     }
```

8.66.2.2 Mysql_Db_Select::where ($ *condition*, $ *operator* = `self::SQL_AND`)

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka
string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Select – objekt Db_Select //TODO dodělat aby se doplňovali magické uvozovky do lauzule

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 132 v souboru select.class.php.

```
132                                                                  {
133        $tmpArray = array();
134
135        if($operator != self::SQL_AND AND $operator != self::SQL_OR){
136           $operator = self::SQL_AND;
137        }
138
139        $tmpArray[self::WHERE_CONDITION_NAME_KEY] = $condition;
140        $tmpArray[self::WHERE_CONDITION_OPERATOR_KEY] = strtoupper($operator);
141
142 //     pokud není vytvořeno pole podmínek -> vytvoř ho
143        if(!is_array($this->_sqlQueryParts[self::SQL_WHERE])){
144           $this->_sqlQueryParts[self::SQL_WHERE] = array();
145        }
146
147        array_push($this->_sqlQueryParts[self::SQL_WHERE], $tmpArray);
148
149        return $this;
150     }
```

8.66.2.3 Mysql_Db_Select::join ($ *tableArray*, $ *condition*, $ *joinType* = `null`, $ *columsArray* = "*")

Metody vytvoří část pro klauzuli JOIN U pole označuje index alias prvku.

Pokud je zadáno null, nebude načten žádný sloupec

Parametry:

string/array – název tabulky (alias je generován z prvních dvou písmen) nebo pole kde index je alias tabulky

string – podmínka v klauzuli ON, je třeba zadat i s aliasy

string – typ JOIN operace hodnoty jsou: JOIN, LEFT, RIGHT, INNER

string/array – název sloupců, které se mají vypsatm, výchozí jsou všechny ("∗").

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 163 v souboru select.class.php.

```
163                                                                                                                 {
164       $tmpArray = array();
165       if(is_array($tableArray)){
166          $columKey = key($tableArray);
167          $tmpArray[self::JOIN_TABLE_NAME_KEY] = $tableArray[$columKey];
168          $tmpArray[self::JOIN_TABLE_CONDITION_KEY] = $condition;
169
170       } else {
171
172          $columKey = $tableArray[0].$tableArray[1].$tableArray[2];
173          $tmpArray[self::JOIN_TABLE_NAME_KEY] = $tableArray;
174          $tmpArray[self::JOIN_TABLE_CONDITION_KEY] = $condition;
175       }
176
177
178       $joinType = strtolower($joinType);
179       switch ($joinType) {
180          case "left":
181              $this->_sqlQueryParts[self::SQL_JOIN][self::SQL_LEFT_JOIN][$columKey] = $tmpArray;
182             break;
183          case "right":
184              $this->_sqlQueryParts[self::SQL_JOIN][self::SQL_RIGHT_JOIN][$columKey] = $tmpArray;
185             break;
186          case "inner":
187              $this->_sqlQueryParts[self::SQL_JOIN][self::SQL_INNER_JOIN][$columKey] = $tmpArray;
188             break;
189          default:
190             $this->_sqlQueryParts[self::SQL_JOIN][self::SQL_JOIN][$columKey] = $tmpArray;
191             break;
192       }
193
194       // přidání sloupců
195       if($columsArray != null){
196          if(is_array($columsArray)){
197             $this->_sqlQueryParts[self::COLUMS_ARRAY][$columKey] = array();
198             foreach ($columsArray as $columAlias => $columName) {
199                if(!is_int($columAlias)){
200                   $this->_sqlQueryParts[self::COLUMS_ARRAY][$columKey][$columAlias] = $columName;
201                } else {
202                   array_push($this->_sqlQueryParts[self::COLUMS_ARRAY][$columKey], $columName);
203                }
204
205             }
206
207          } else {
208             $this->_sqlQueryParts[self::COLUMS_ARRAY][$columKey] = array();
209             array_push($this->_sqlQueryParts[self::COLUMS_ARRAY][$columKey], $columsArray);
210             //       $this->_sqlParts[self::COLUMS_ARRAY][$columKey] = $coolsArray;
211          }
212       }
213
214       return $this;
215    }
```

8.66.2.4 Mysql_Db_Select::order ($ *colum*, $ *order* = `self::SQL_ASC`)

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 225 v souboru select.class.php.

```
225                                                     {
226       $order = strtoupper($order);
227       if(!is_array($this->_sqlQueryParts[self::ORDER_ORDER_KEY])){
228             $this->_sqlQueryParts[self::ORDER_ORDER_KEY] = array();
229          }
230
231       $columArray = array();
232
233       if($order == self::SQL_DESC){
234          $columArray[self::ORDER_COLUM_KEY] = $colum;
235          $columArray[self::ORDER_ORDER_KEY] = self::SQL_DESC;
236       } else {
237          $columArray[self::ORDER_COLUM_KEY] = $colum;
238          $columArray[self::ORDER_ORDER_KEY] = self::SQL_ASC;
239       }
240       array_push($this->_sqlQueryParts[self::ORDER_ORDER_KEY], $columArray);
241       return $this;
242    }
```

8.66.2.5 Mysql_Db_Select::group ($ *colum*, $ *withRollup* = `false`)

Metoda přiřadí slouření sloupců v SQL dotazu pomocí klauzule GROUP BY.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) WITH ROLLUP false(default)/true

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 252 v souboru select.class.php.

```
252                                                     {
253       if(!is_array($this->_sqlQueryParts[self::GROUP_BY_KEY])){
254          $this->_sqlQueryParts[self::GROUP_BY_KEY] = array();
255       }
256
```

```
257         $groupTmpArray = array();
258         $groupTmpArray[self::ORDER_COLUM_KEY] = $colum;
259 
260         if(!$withRollup){
261            $groupTmpArray[self::GROUP_WITH_ROLLUP] = false;
262         } else {
263            $groupTmpArray[self::GROUP_WITH_ROLLUP] = true;
264         }
265 
266         array_push($this->_sqlQueryParts[self::GROUP_BY_KEY], $groupTmpArray);
267 
268         return $this;
269 
270     }
```

8.66.2.6 Mysql_Db_Select::limit ($ *rowCount*, $ *offset*)

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 279 v souboru select.class.php.

```
279                                             {
280         $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_COUNT_ROWS_KEY] = $rowCount;
281         $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_OFFSET_KEY] = $offset;
282 
283         return $this;
284     }
```

8.66.2.7 Mysql_Db_Select::count ($ *alias* = `null`, $ *colum* = `self::SQL_ALL_VALUES`)

Metoda přidává do dotazu sloupce s počem záznamů.

Parametry:

string – alias pod kterým má být vrácena hodnota

Návratová hodnota:

Db_Select – objekt Db_Select

Reimplementuje stejnojmenný prvek z Db_Select.

Definice je uvedena na řádku 292 v souboru select.class.php.

```
292                                                                         {
293       if($alias == null){
294          array_push($this->_sqlQueryParts[self::SQL_COUNT], self::SQL_ALL_VALUES);
295       } else {
296          $this->_sqlQueryParts[self::SQL_COUNT][$alias] = self::SQL_ALL_VALUES;
297       }
298
299       return $this;
300    }
```

8.66.2.8 Mysql_Db_Select::__toString ()

Metoda převede objekt na řetězec.

Návratová hodnota:

string – objekt jako řetězec

Definice je uvedena na řádku 525 v souboru select.class.php.

```
526     {
527         $sql = self::SQL_SELECT;
528
529         foreach ($this->_sqlQueryParts as $partKey => $partValue) {
530          $createMethod = '_create' . ucfirst($partKey);
531          if(method_exists($this, $createMethod)){
532             $sql .= $this->$createMethod();
533          }
534          ;
535         }
536
537 //    echo "<pre>";
538 //    print_r($this);
539 //    echo "</pre>";
540         return $sql;
541     }
```

8.66.3 Dokumentace k datovým členům

8.66.3.1 Mysql_Db_Select::$_sqlPartsInit [static, protected]

Initializer:

```
array(self::COLUMS_ARRAY        => array(),
                                self::SQL_COUNT            => array(),
                                self::SQL_FROM          => array(),
                                self::SQL_JOIN          => array(),
                                self::SQL_WHERE            => array(),
                                self::GROUP_BY_KEY         => array(),
                                self::ORDER_ORDER_KEY      => array(),
                                self::SQL_LIMIT            => array())
```

Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

Definice je uvedena na řádku 70 v souboru select.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/mysql/select.class.php

8.67 Dokumentace třídy Mysql_Db_Update

Třída pro aktualizaci záznamů v MySQL DB.

Diagram dědičnosti pro třídu Mysql_Db_Update

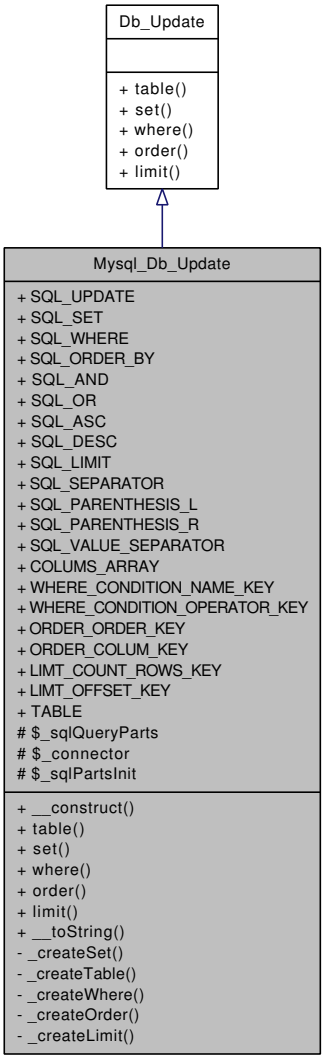
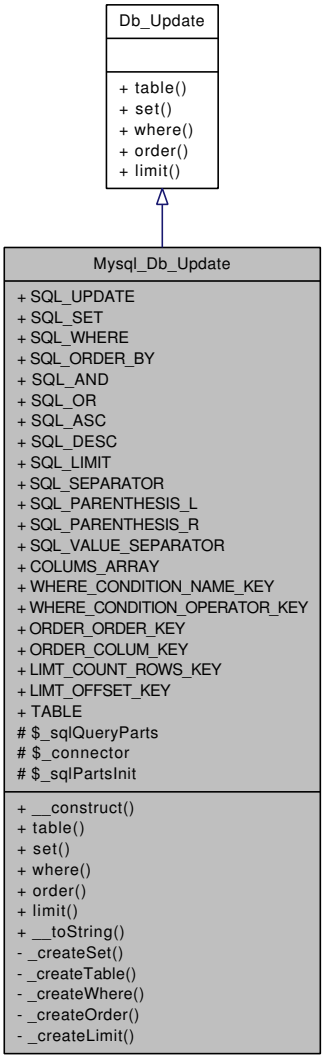


Diagram tříd pro Mysql_Db_Update:



Veřejné metody

- __construct (Db $conector)

  Konstruktor třídy vytváří objet UPDATE pro update databáze.

- table ($table)

  Metoda nastavuje v které tabulce se bude upravovat klauzule UPDATE table.

- set ($values)

  Metoda nastavuje, které hodnoty se upraví (název sloupce) => (hodnota).

- where ($condition, $operator=self::SQL_AND)

  Metody vatváří podmínku WHERE.

- order ($colum, $order=self::SQL_ASC)

Metoda přiřadí řazení sloupcu v SQL dotazu.

- limit ($rowCount, $offset)

  Metoda přidá do SQL dotazu klauzuli LIMIT.

- __toString ()

  Metoda převede objekt na řetězec.

Veřejné atributy

- const **SQL_UPDATE** = 'UPDATE'
- const **SQL_SET** = 'SET'
- const **SQL_WHERE** = 'WHERE'
- const **SQL_ORDER_BY** = 'ORDER BY'
- const **SQL_AND** = 'AND'
- const **SQL_OR** = 'OR'
- const **SQL_ASC** = 'ASC'
- const **SQL_DESC** = 'DESC'
- const **SQL_LIMIT** = 'LIMIT'
- const **SQL_SEPARATOR** = ' '
- const **SQL_PARENTHESIS_L** = '('
- const **SQL_PARENTHESIS_R** = ')'
- const **SQL_VALUE_SEPARATOR** = ','
- const **COLUMS_ARRAY** = 'SET'
- const **WHERE_CONDITION_NAME_KEY** = 'condition'
- const **WHERE_CONDITION_OPERATOR_KEY** = 'operator'
- const **ORDER_ORDER_KEY** = 'ORDER'
- const **ORDER_COLUM_KEY** = 'colum'
- const **LIMT_COUNT_ROWS_KEY** = 'limit_count'
- const **LIMT_OFFSET_KEY** = 'limit_offset'
- const **TABLE** = 'TABLE'

Chráněné atributy

- **$_sqlQueryParts** = array()
- **$_connector** = null

Statické chráněné atributy

- static $_sqlPartsInit

  Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

8.67.1 Detailní popis

Třída pro aktualizaci záznamů v MySQL DB.

Třída obsahuje implementaci metody update z db.interfacu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: update.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro aktualizaci záznamů

Definice je uvedena na řádku 14 v souboru update.class.php.

8.67.2 Dokumentace konstruktoru a destruktoru

8.67.2.1 Mysql_Db_Update::__construct (Db $ *conector*)

Konstruktor třídy vytváří objet UPDATE pro update databáze.

Parametry:

Db – objekt db konektoru

Definice je uvedena na řádku 77 v souboru update.class.php.

```
77                                          {
78 //    inicializace do zakladni podoby;
79       $this->_connector = $conector;
80       $this->_sqlQueryParts = self::$_sqlPartsInit;
81    }
```

8.67.3 Dokumentace k metodám

8.67.3.1 Mysql_Db_Update::table ($ *table*)

Metoda nastavuje v které tabulce se bude upravovat klauzule UPDATE table.

Parametry:

string – tabulka která se bude upravovat

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementuje stejnojmenný prvek z Db_Update.

Definice je uvedena na řádku 91 v souboru update.class.php.

```
91                                  {
92       $this->_sqlQueryParts[ self::TABLE] = $table;
93       return $this;
94    }
```

8.67.3.2 Mysql_Db_Update::set ($ *values*)

Metoda nastavuje, které hodnoty se upraví (název sloupce) => (hodnota).

Parametry:

array – pole s hodnotamik

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementuje stejnojmenný prvek z Db_Update.

Definice je uvedena na řádku 104 v souboru update.class.php.

```
105    {
106       foreach ($values as $key => $value){
107          $this->_sqlQueryParts[self::COLUMS_ARRAY][$key] = $value;
108       }
109       return $this;
110    }
```

8.67.3.3 Mysql_Db_Update::where ($ *condition*, $ *operator* = `self::SQL_AND`)

Metody vatváří podmínku WHERE.

Parametry:

string – podmínka

string – typ spojení podmínky (AND, OR) (výchozí je AND)

Návratová hodnota:

Db_Update – objekt Db_Update //TODO dodělat aby se doplňovali magické uvozovky do lauzule

Reimplementuje stejnojmenný prvek z Db_Update.

Definice je uvedena na řádku 122 v souboru update.class.php.

```
122                                                                {
123       $tmpArray = array();
124
125       if($operator != self::SQL_AND AND $operator != self::SQL_OR){
126          $operator = self::SQL_AND;
127       }
128
129       $tmpArray[self::WHERE_CONDITION_NAME_KEY] = $condition;
130       $tmpArray[self::WHERE_CONDITION_OPERATOR_KEY] = strtoupper($operator);
131
132 //    pokud není vytvořeno pole podmínek -> vytvoř ho
133       if(!is_array($this->_sqlQueryParts[self::SQL_WHERE])){
134          $this->_sqlQueryParts[self::SQL_WHERE] = array();
135       }
136       array_push($this->_sqlQueryParts[self::SQL_WHERE], $tmpArray);
137       return $this;
138    }
```

8.67.3.4 Mysql_Db_Update::order ($ *colum*, $ *order* = self::SQL_ASC)

Metoda přiřadí řazení sloupcu v SQL dotazu.

Parametry:

string – sloupec, podle kterého se má řadit

string – (option) jak se má sloupec řadit (ASC, DESC) (default: ASC)

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementuje stejnojmenný prvek z Db_Update.

Definice je uvedena na řádku 148 v souboru update.class.php.

```
148                                                     {
149         $order = strtoupper($order);
150         if(!is_array($this->_sqlQueryParts[self::ORDER_ORDER_KEY])){
151               $this->_sqlQueryParts[self::ORDER_ORDER_KEY] = array();
152         }
153         $columArray = array();
154 
155         if($order == self::SQL_DESC){
156            $columArray[self::ORDER_COLUM_KEY] = $colum;
157            $columArray[self::ORDER_ORDER_KEY] = self::SQL_DESC;
158         } else {
159            $columArray[self::ORDER_COLUM_KEY] = $colum;
160            $columArray[self::ORDER_ORDER_KEY] = self::SQL_ASC;
161         }
162         array_push($this->_sqlQueryParts[self::ORDER_ORDER_KEY], $columArray);
163         return $this;
164     }
```

8.67.3.5 Mysql_Db_Update::limit ($ *rowCount*, $ *offset*)

Metoda přidá do SQL dotazu klauzuli LIMIT.

Parametry:

integer – počet záznamů

integer – záčátek

Návratová hodnota:

Db_Update – objekt Db_Update

Reimplementuje stejnojmenný prvek z Db_Update.

Definice je uvedena na řádku 173 v souboru update.class.php.

```
173                                             {
174         $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_COUNT_ROWS_KEY] = $rowCount;
175         $this->_sqlQueryParts[self::SQL_LIMIT][self::LIMT_OFFSET_KEY] = $offset;
176         return $this;
177     }
```

8.67.3.6 Mysql_Db_Update::__toString ()

Metoda převede objekt na řetězec.

Návratová hodnota:

string – objekt jako řetězec

Definice je uvedena na řádku 275 v souboru update.class.php.

```
276     {
277         $sql = self::SQL_UPDATE;
278         foreach ($this->_sqlQueryParts as $partKey => $partValue) {
279          $createMethod = '_create' . ucfirst($partKey);
280          if(method_exists($this, $createMethod)){
281             $sql .= $this->$createMethod();
282          }
283         }
284         return $sql;
285     }
```

8.67.4 Dokumentace k datovým členům

8.67.4.1 Mysql_Db_Update::$_sqlPartsInit [static, protected]

Initializer:

```
array(self::TABLE               => array(),
                                self::COLUMS_ARRAY         => array(),
                                self::SQL_WHERE            => array(),
                                self::ORDER_ORDER_KEY      => array(),
                                self::SQL_LIMIT            => array())
```

Konstanta určující obsah SQL dotazu Musí mít správné pořadí, jak se má SQL dotaz řadit!!!

Definice je uvedena na řádku 52 v souboru update.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/mysql/update.class.php

8.68 Dokumentace třídy MySQLDb

Třída implementující databázový objekt typu MySQL.

Diagram dědičnosti pro třídu MySQLDb

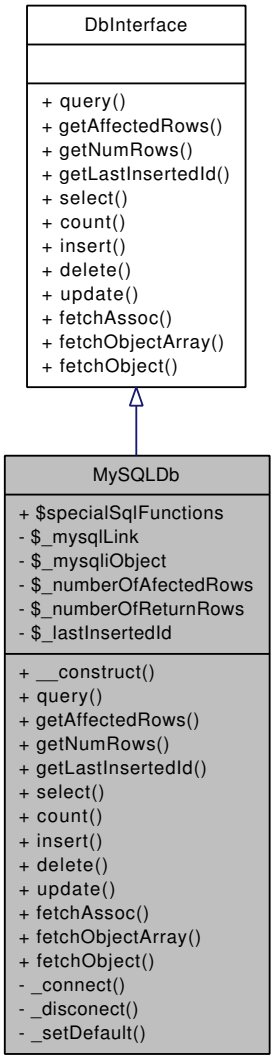
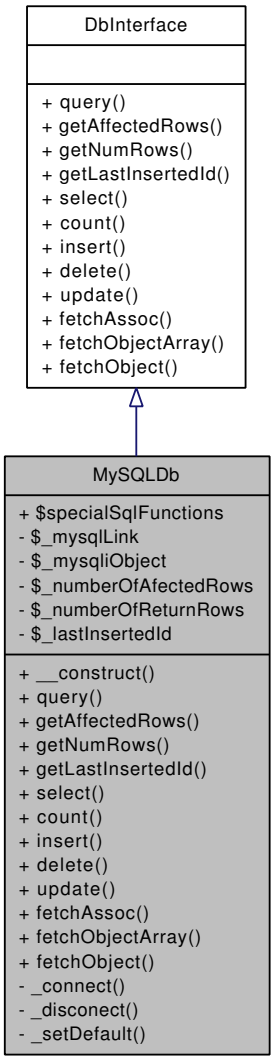


Diagram tříd pro MySQLDb:



Veřejné metody

- __construct ()

  Konstruktor třídy nastaví základní parametry.

- query ($sqlQuery)

  Metoda provede daný sql dotaz na databázi.

- getAffectedRows ()

  Metoda vrací počet ovlivněných záznamů příkazy INSERT, UPDATE, DELETE, REPLACE.

- getNumRows ()

  Metoda vrací počet vrácených záznamů příkazy SELECT.

- getLastInsertedId ()

Metoda vrací id posledního vloženého záznamu pomocí INSERT.

- select ()

  Metoda pro generování SQL dotazů typu SELECT.

- count ($table, $condition=null)

  Metoda vrací počet záznamů v zadané tabulce.

- insert ()

  Metoda pro generování SQL dotazů typu INSERT.

- delete ()

  Metoda pro generování SQL dotazů typu DELETE.

- update ()

  Metoda pro generování SQL dotazů typu UPDATE.

- fetchAssoc ($sqlQuery, $oneArray=false)

  Metoda provede sql dotaz a výstup doplní do asociativního pole.

- fetchObjectArray ($sqlQuery)

  Metoda provede sql dotaz a výstup přiřadí do pole objektů mysqli.

- fetchObject ($sqlQuery)

  Metoda vrací řádek z db jako objekt.

Statické veřejné atributy

- static **$specialSqlFunctions** = array("NOW", "TIMESTAMPDIFF", "COUNT", "IFNULL", "IF")

8.68.1 Detailní popis

Třída implementující databázový objekt typu MySQL.

Třída implementuje objet db konektoru k MySQL. Obsahuje implementace metod pro práci s SQL dotazy a samotné připojování k databázi.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: db.class.php 3.0.0 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro MySQL DB

Definice je uvedena na řádku 19 v souboru db.class.php.

8.68.2 Dokumentace konstruktoru a destruktoru

8.68.2.1 MySQLDb::__construct ()

Konstruktor třídy nastaví základní parametry.

Parametry:

string – adresa serveru
string – uživatelské jméno
string – uživatelské heslo
string – název databáze
string – prefix tabulek (option)

Definice je uvedena na řádku 66 v souboru db.class.php.

```
66                          {
67        $this->_mysqliObject = new mysqli(parent::$_serverName, parent::$_userName, parent::$_userPasswor
68     }
```

8.68.3 Dokumentace k metodám

8.68.3.1 MySQLDb::query ($ *sqlQuery*)

Metoda provede daný sql dotaz na databázi.

Parametry:

string – sql dotaz

Implementuje DbInterface.

Definice je uvedena na řádku 108 v souboru db.class.php.

Používá se v count(), fetchAssoc(), fetchObject() a fetchObjectArray().

```
108                                     {
109        $this->_connect();
110        $this->_setDefault();
111
112        Db::addQueryCount();
113
114        $result = $this->_mysqliObject->query($sqlQuery);
115        if($result != null){
116           $queryType = strtolower(substr($sqlQuery, 0, 6));
117
118           if ($queryType == "select"){
119              $this->_numberOfReturnRows=$result->num_rows;
120           } else if($queryType == "insert"){
121              $this->_lastInsertedId=$this->_mysqliObject->insert_id;
122              $this->_numberOfAfectedRows = $this->_mysqliObject->affected_rows;
123           } else if($queryType == "delete"){
124              $this->_numberOfAfectedRows = $this->_mysqliObject->affected_rows;
125           } else if($queryType == "update"){
126              $this->_numberOfAfectedRows = $this->_mysqliObject->affected_rows;
127           }
128        } else {
```

```
129           new CoreException($this->_mysqliObject->error, 204);
130           $result = false;
131       }
132       return $result;
133   }
```

8.68.3.2 MySQLDb::getAffectedRows ()

Metoda vrací počet ovlivněných záznamů příkazy INSERT, UPDATE, DELETE, REPLACE.

Návratová hodnota:

integer – počet ovlivněných záznamu

Implementuje DbInterface.

Definice je uvedena na řádku 139 v souboru db.class.php.

```
139                                     {
140       return $this->_numberOfAfectedRows;
141   }
```

8.68.3.3 MySQLDb::getNumRows ()

Metoda vrací počet vrácených záznamů příkazy SELECT.

Návratová hodnota:

integer – počet vrácených záznamu

Implementuje DbInterface.

Definice je uvedena na řádku 147 v souboru db.class.php.

```
147                                 {
148       return $this->_numberOfReturnRows;
149   }
```

8.68.3.4 MySQLDb::getLastInsertedId ()

Metoda vrací id posledníího vloženého záznamu pomocí INSERT.

Návratová hodnota:

integer – id posledního záznamu

Implementuje DbInterface.

Definice je uvedena na řádku 155 v souboru db.class.php.

```
155                                       {
156       return $this->_lastInsertedId;
157   }
```

8.68.3.5 MySQLDb::select ()

Metoda pro generování SQL dotazů typu SELECT.

Návratová hodnota:

Mysql_Db_Select/string – vrací objekt nebo řetězec dotazu SELECT

Implementuje DbInterface.

Definice je uvedena na řádku 167 v souboru db.class.php.

```
167                             {
168         return new Mysql_Db_Select($this);
169     }
```

8.68.3.6 MySQLDb::count ($ *table*, $ *condition* = `null`)

Metoda vrací počet záznamů v zadané tabulce.

Parametry:

string – název tabulky

string – podmínka

Návratová hodnota:

integer – počet záznamů v tabulce

Implementuje DbInterface.

Definice je uvedena na řádku 178 v souboru db.class.php.

Odkazuje se na query().

```
179     {
180         $select = new Mysql_Db_Select($this);
181         $sqlCount = $select->from(array("tbl"=>$table))->count("count");
182 
183         if($condition != null){
184             $sqlCount = $sqlCount->where($condition);
185         }
186         $result = $this->query($sqlCount);
187 
188         if($result != null){
189             return $result->fetch_object()->count;
190         } else {
191             return 0;
192         }
193     }
```

Tato funkce volá...

8.68.3.7 MySQLDb::insert ()

Metoda pro generování SQL dotazů typu INSERT.

Návratová hodnota:

Mysql_Db_Insert/string – vrací objekt nebo řetězec dotazu INSERT

Implementuje DbInterface.

Definice je uvedena na řádku 201 v souboru db.class.php.

```
201                             {
202         return new Mysql_Db_Insert($this);
203     }
```

8.68.3.8 MySQLDb::delete ()

Metoda pro generování SQL dotazů typu DELETE.

Návratová hodnota:

Mysql_Db_Delete/string – vrací objekt nebo řetězec dotazu DELETE

Implementuje DbInterface.

Definice je uvedena na řádku 211 v souboru db.class.php.

```
211                             {
212         return new Mysql_Db_Delete($this);
213     }
```

8.68.3.9 MySQLDb::update ()

Metoda pro generování SQL dotazů typu UPDATE.

Návratová hodnota:

Mysql_Db_Update/string – vrací objekt nebo řetězec dotazu UPDATE

Implementuje DbInterface.

Definice je uvedena na řádku 221 v souboru db.class.php.

```
221                             {
222         return new Mysql_Db_Update($this);
223     }
```

8.68.3.10 MySQLDb::fetchAssoc ($ *sqlQuery*, $ *oneArray* = `false`)

Metoda provede sql dotaz a výstup doplní do asociativního pole.

Parametry:

string – SQL dotaz

boolean – jestli se má vrátit pouze pole s prvky, nebo pole s poly prvků

Návratová hodnota:

array/boolean – asociativní pole s výsledky sql dotazu nebo false při prázdném výsledku

Implementuje DbInterface.

Definice je uvedena na řádku 232 v souboru db.class.php.

Odkazuje se na query().

```
232                                                         {
233       $queryResult = $this->query($sqlQuery);
234
235       if($queryResult){
236          $resultArray = array();
237          if(!$oneArray){
238             while ($sqlData=$queryResult->fetch_assoc()) {
239                array_push($resultArray, $sqlData);
240             }
241          } else {
242             $resultArray = $queryResult->fetch_assoc();
243          }
244          return $resultArray;
245       } else {
246          return false;
247       }
248    }
```

Tato funkce volá...

8.68.3.11 MySQLDb::fetchObjectArray ($ *sqlQuery*)

Metoda provede sql dotaz a výstup přiřadí do pole objektů mysqli.

Parametry:

string – SQL dotaz

Návratová hodnota:

array – pole objektů MySQLi_Result

Implementuje DbInterface.

Definice je uvedena na řádku 256 v souboru db.class.php.

Odkazuje se na query().

```
256                                                 {
257       $queryResult = $this->query($sqlQuery);
```

```
258
259       if($queryResult){
260          $resultArray = array();
261          while ($sqlObject=$queryResult->fetch_object()) {
262             array_push($resultArray, $sqlObject);
263          }
264          return $resultArray;
265       } else {
266          return false;
267       }
268    }
```

Tato funkce volá...

8.68.3.12 MySQLDb::fetchObject ($ *sqlQuery*)

Metoda vrací řádek z db jako objekt.

Parametry:

string – sql dotaz

Návratová hodnota:

MySQLi_Result – objekt s prvky z db

Implementuje DbInterface.

Definice je uvedena na řádku 275 v souboru db.class.php.

Odkazuje se na query().

```
276    {
277       $queryResult = $this->query($sqlQuery);
278       if($queryResult){
279          return $queryResult->fetch_object();
280       } else {
281          return false;
282       }
283    }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/db/mysql/db.class.php

8.69 Dokumentace třídy NewsAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu NewsAction

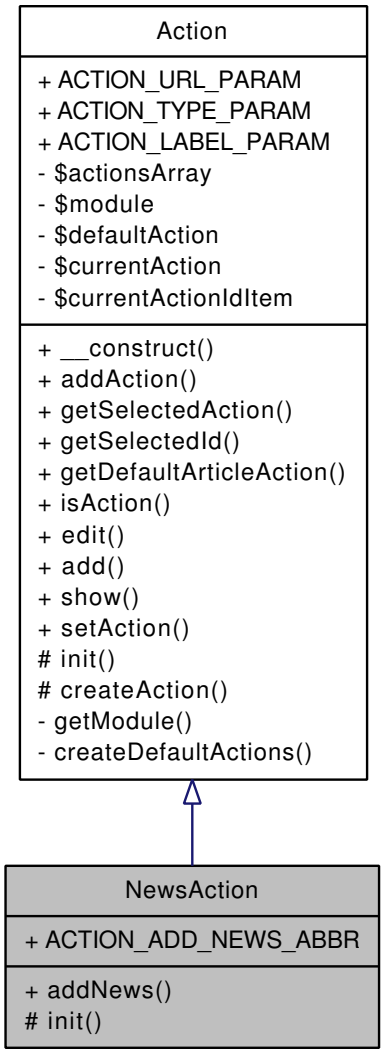
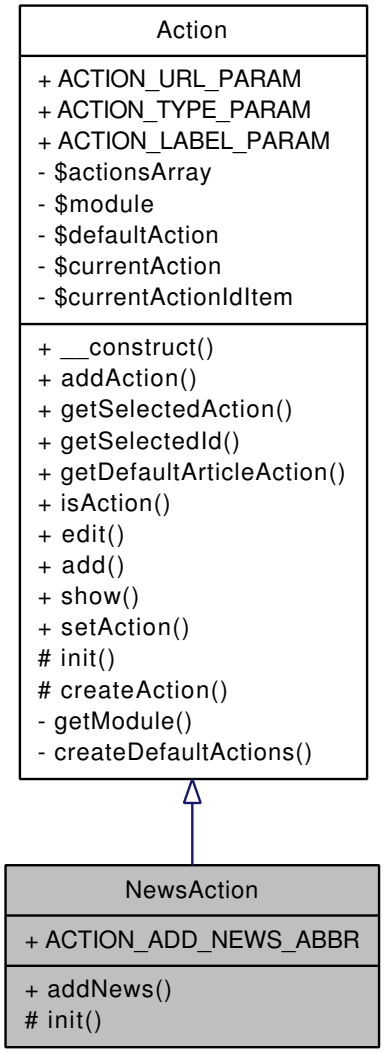


Diagram tříd pro NewsAction:



Veřejné metody

- **addNews** ()

Veřejné atributy

- const **ACTION_ADD_NEWS_ABBR** = 'an'

Chráněné metody

- init ()

  Metoda pro inicializaci akcí.

8.69.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/news/action.class.php

8.70 Dokumentace třídy NewsController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu NewsController

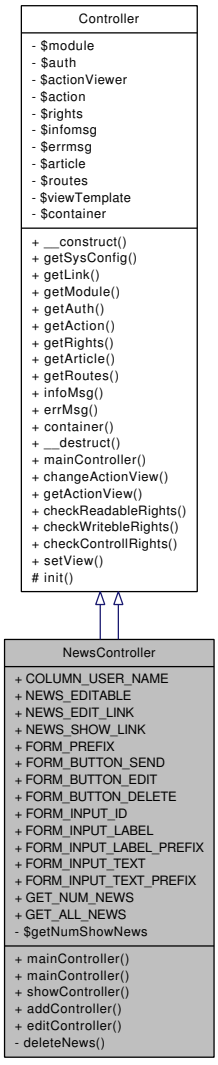
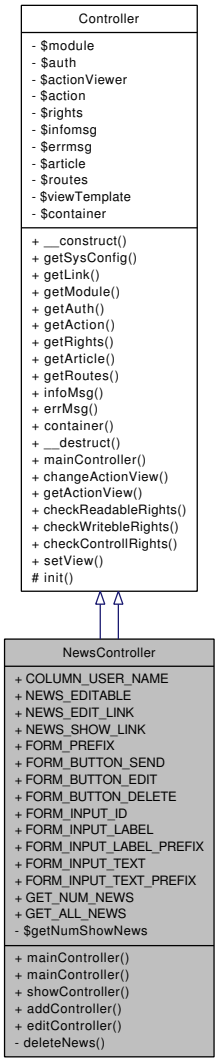


Diagram tříd pro NewsController:



Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- mainController ()

  Kontroler pro zobrazení novinek.

- **showController** ()
- addController ()

  Kontroler pro přidání novinky.

- editController ()

  controller pro úpravu novinky

Veřejné atributy

- const **COLUMN_USER_NAME** = 'username'
- const **NEWS_EDITABLE** = 'editable'
- const **NEWS_EDIT_LINK** = 'editlink'
- const **NEWS_SHOW_LINK** = 'showlink'
- const **FORM_PREFIX** = 'news_'
- const **FORM_BUTTON_SEND** = 'send'
- const **FORM_BUTTON_EDIT** = 'edit'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **FORM_INPUT_ID** = 'id'
- const **FORM_INPUT_LABEL** = 'label'
- const **FORM_INPUT_LABEL_PREFIX** = 'label_'
- const **FORM_INPUT_TEXT** = 'text'
- const **FORM_INPUT_TEXT_PREFIX** = 'text_'
- const **GET_NUM_NEWS** = 'numnews'
- const **GET_ALL_NEWS** = 'all'

8.70.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujících souborů:

- /home/cuba/work-net/vve3_2/modules/example_module/controler.class.php
- /home/cuba/work-net/vve3_2/modules/news/controler.class.php

8.71 Dokumentace třídy NewsView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu NewsView

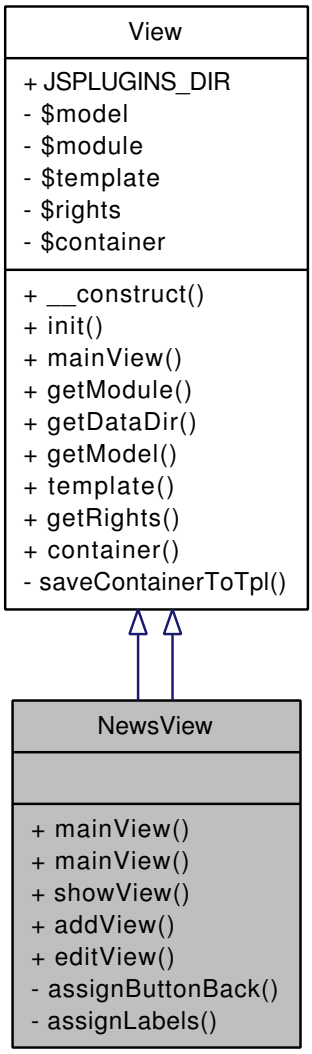
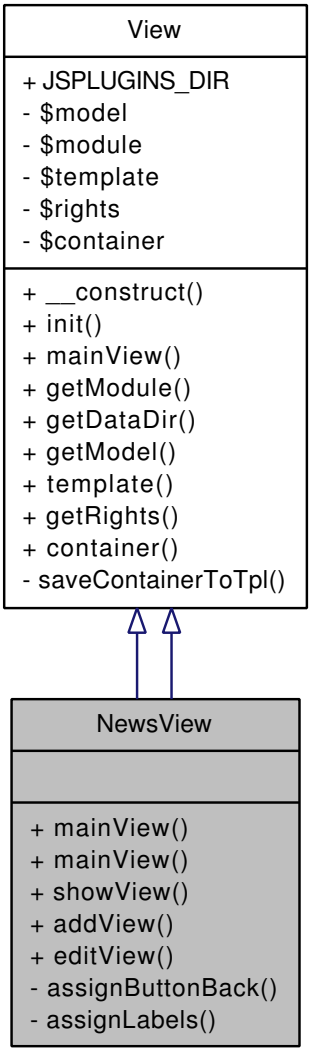


Diagram tříd pro NewsView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- **showView** ()
- addView ()

  Viewer pro přidání novinky.

- editView ()

  Viewer pro editaci novinky.

8.71.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

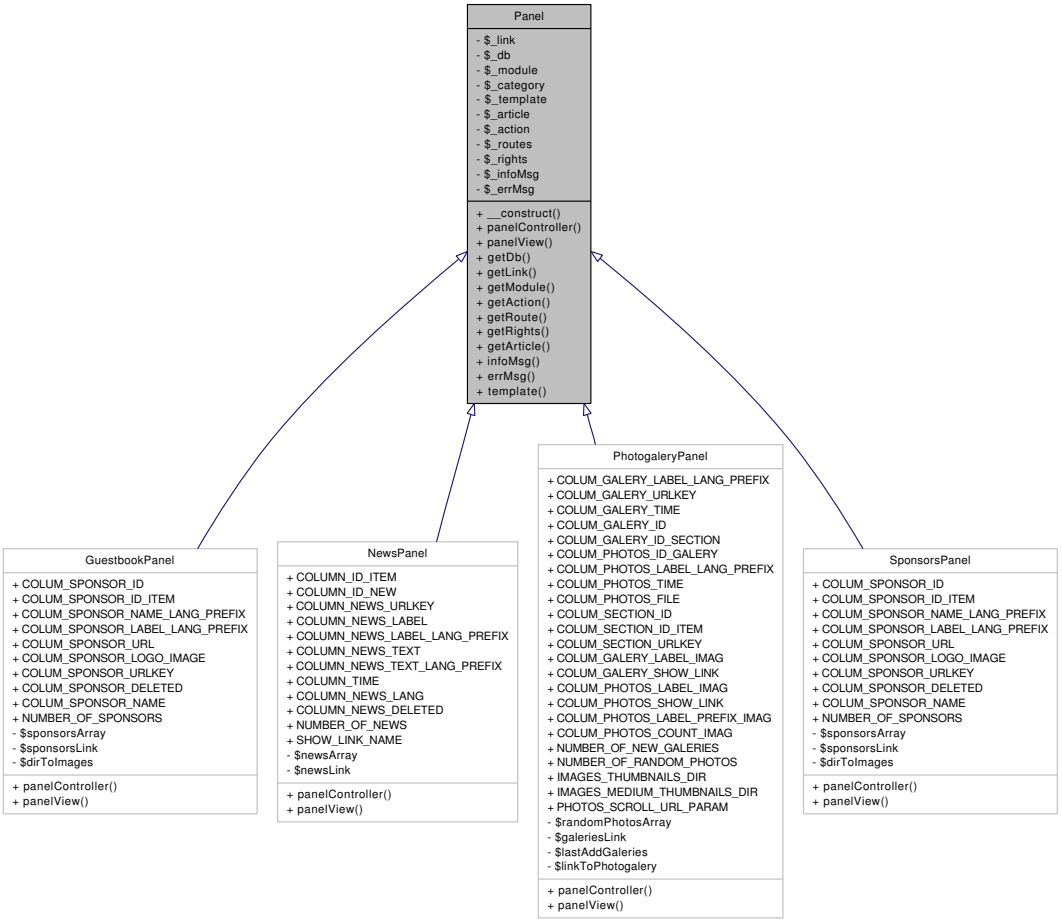
Dokumentace pro tuto třídu byla generována z následujících souborů:

- /home/cuba/work-net/vve3_2/modules/example_module/view.class.php
- /home/cuba/work-net/vve3_2/modules/news/view.class.php

8.72 Dokumentace třídy Panel

Abstraktní třída pro práci s panely.

Diagram dědičnosti pro třídu Panel



Veřejné metody

- __construct ($category, Template &$template, Rights $rights)

  Konstruktor.

- panelController ()

  Metoda controleru panelu.

- panelView ()

  Metoda viewru panelu.

- getDb ()

  Metoda vrací objekt pro přístup k db.

- getLink ($clear=true)

Metoda vrací odkaz na objekt pro práci s odkazy.

- getModule ()

  Metody vrací objekt modulu.

- getAction ()

  Metoda vrací objekt na akci.

- getRoute ()

  Metoda vrací objekt na akci.

- getRights ()

  Metoda vrací objekt s právy na modul.

- getArticle ()

  Metoda vrací objekt s článkem.

- infoMsg ()

  Metoda vrací objekt s informačními zprávami.

- errMsg ()

  Metoda vrací objekt s chybovými zprávami.

- template ()

  Metoda vrací objekt šablony, přes který se přiřazují proměnné do šablony.

8.72.1 Detailní popis

Abstraktní třída pro práci s panely.

Základní třída pro tvorbu tříd panelů jednotlivých modulu. Poskytuje prvky základního přístu jak k vlastnostem modelu tak pohledu. Pomocí této třídy se také generují šablony panelů.

Verze:

Id

panel.class.php 419 2008-11-28 23:21:19Z jakub

VVE3.3.0

Revision

419

Autor:

$Author:$

Date

$LastChangedBy:$ $LastChangedDate:$ Abstraktní třída pro práci s panely

Plánované úpravy

Není implementována práce s chybami

Definice je uvedena na řádku 16 v souboru panel.class.php.

8.72.2 Dokumentace k metodám

8.72.2.1 Panel::getDb () [final]

Metoda vrací objekt pro přístup k db.

Návratová hodnota:

DbInterface – objekt databáze

Definice je uvedena na řádku 152 v souboru panel.class.php.

```
152                                 {
153         return $this->_db;
154     }
```

8.72.2.2 Panel::getLink ($ *clear* = true) [final]

Metoda vrací odkaz na objekt pro práci s odkazy.

Návratová hodnota:

Links – objekt pro práci s odkazy

Definice je uvedena na řádku 160 v souboru panel.class.php.

```
160                                                 {
161         $link = new Links($clear);
162         return $link->category($this->_category[Category::COLUM_CAT_LABEL],
163             $this->_category[Category::COLUM_CAT_ID]);
164     }
```

8.72.2.3 Panel::getModule () [final]

Metody vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 170 v souboru panel.class.php.

Používá se v __construct().

```
170                                     {
171         return $this->_module;
172     }
```

8.72.2.4 Panel::getAction () [final]

Metoda vrací objekt na akci.

Návratová hodnota:

ModuleAction – objekt akce

Definice je uvedena na řádku 178 v souboru panel.class.php.

```
178                                     {
179         return $this->_action;
180     }
```

8.72.2.5 Panel::getRoute () [final]

Metoda vrací objekt na akci.

Návratová hodnota:

ModuleAction – objekt akce

Definice je uvedena na řádku 186 v souboru panel.class.php.

```
186                                     {
187         return $this->_routes;
188     }
```

8.72.2.6 Panel::getRights () [final]

Metoda vrací objekt s právy na modul.

Návratová hodnota:

Rights – objekt práv

Definice je uvedena na řádku 194 v souboru panel.class.php.

```
194                                     {
195         return $this->_rights;
196     }
```

8.72.2.7 Panel::getArticle () [final]

Metoda vrací objekt s článkem.

Návratová hodnota:

Article – objekt článku

Definice je uvedena na řádku 202 v souboru panel.class.php.

```
202                                     {
203         return $this->_article;
204     }
```

8.72.2.8 Panel::infoMsg () [final]

Metoda vrací objekt s informačními zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 210 v souboru panel.class.php.

```
210                                     {
211         return $this->_infoMsg;
212     }
```

8.72.2.9 Panel::errMsg () [final]

Metoda vrací objekt s chybovými zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 218 v souboru panel.class.php.

```
218                                     {
219         return $this->_errMsg;
220     }
```

8.72.2.10 Panel::template () [final]

Metoda vrací objekt šablony, přes který se přiřazují proměnné do šablony.

Návratová hodnota:

Template – objekt šablony

Definice je uvedena na řádku 226 v souboru panel.class.php.

```
226                                     {
227         return $this->_template;
228     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/panel.class.php

8.73 Dokumentace třídy PhotogaleryAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu PhotogaleryAction

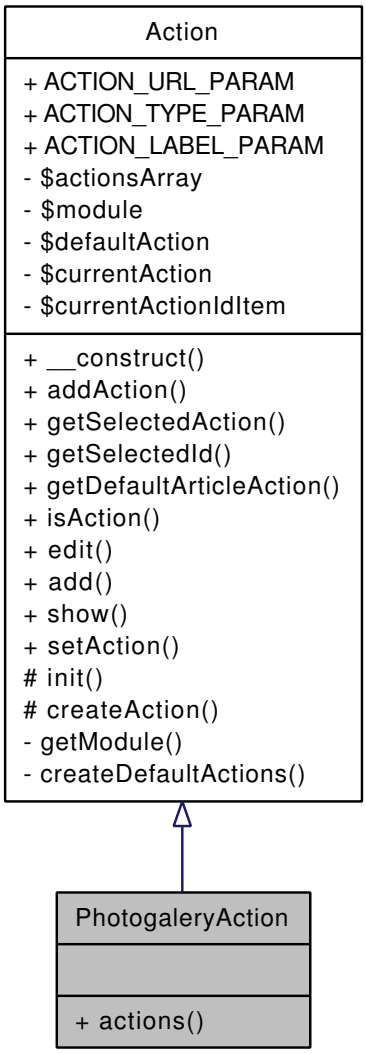
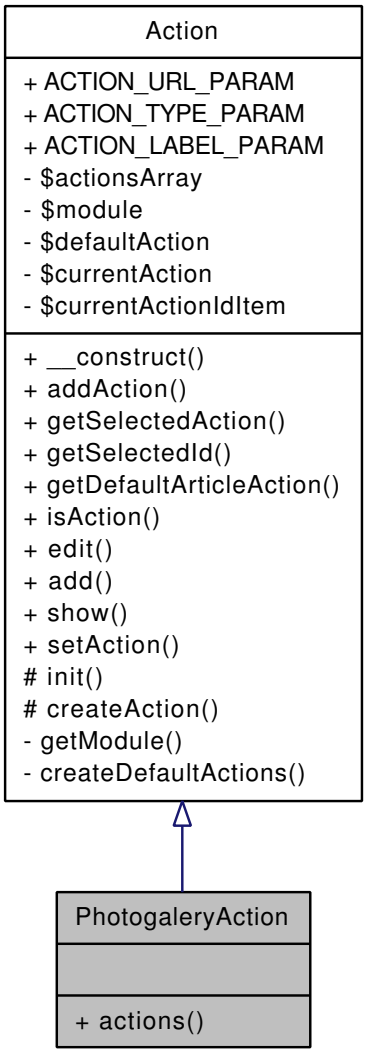


Diagram tříd pro PhotogaleryAction:



Veřejné metody

- **actions** ()

8.73.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/photogalery/action.class.php

8.74 Dokumentace třídy ProgressBarEplugin

Třída Epluginu pro práci s progrssbarem.

Diagram dědičnosti pro třídu ProgressBarEplugin

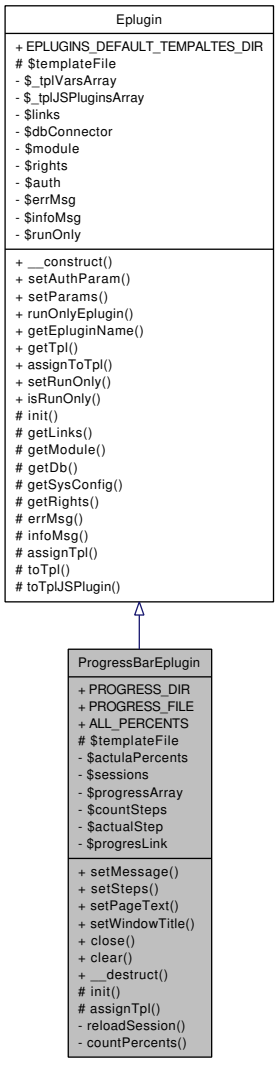
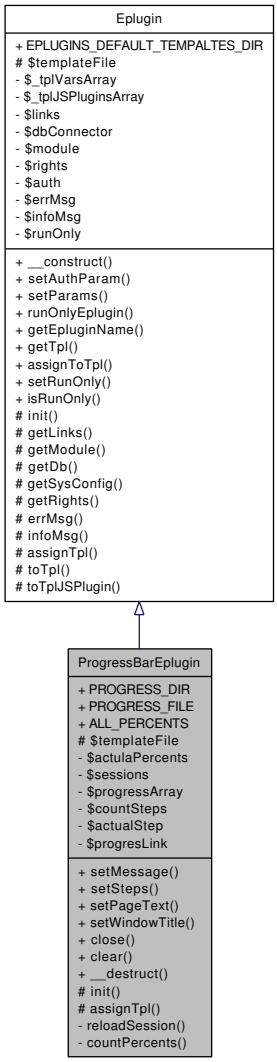


Diagram tříd pro ProgressBarEplugin:



Veřejné metody

- setMessage ($message)

  Metoda nastaví zobrazovanou správu v progressbaru.

- setSteps ($steps)

  Metoda nastavuje počet kroků v progressbaru.

- setPageText ($text=null)

  Metoda nastavuje text zprávy.

- setWindowTitle ($title)

  Metoda nastavuje titulek okna.

- close ()

Uzavření okna s progressbarem.

- clear ()

  Metoda vynuluje progressbar.

- __destruct ()

  Metoda vynuluje progresbar.

Veřejné atributy

- const **PROGRESS_DIR** = 'progressbar'
- const **PROGRESS_FILE** = 'progressbar.php'
- const **ALL_PERCENTS** = 100

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda obstarává přiřazení proměných do šablony.

Chráněné atributy

- **$templateFile** = null

8.74.1 Detailní popis

Třída Epluginu pro práci s progrssbarem.

Třída je určena pro práci s progressbarem, který zobrazuje stav provádění ve vlastním otevřeném okně. Je využit například při nahrávání více fotek.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: scroll.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída EPluginu pro práci se scrolovátky

Plánované úpravy

dodělat možnost vytvářet vlastní styl okna

Definice je uvedena na řádku 14 v souboru progressbar.class.php.

8.74.2 Dokumentace k metodám

8.74.2.1 ProgressBarEplugin::setMessage ($ *message*)

Metoda nastaví zobrazovanou správu v progressbaru.

Parametry:

string – zpráva

Definice je uvedena na řádku 96 v souboru progressbar.class.php.

```
97    {
98       $this->progressArray['message']   = $message;
99       $this->countPercents();
100       $this->actualStep++;
101       $this->reloadSession();
102       sleep(1);
103    }
```

8.74.2.2 ProgressBarEplugin::setSteps ($ *steps*)

Metoda nastavuje počet kroků v progressbaru.

Parametry:

integer – počet kroků

Definice je uvedena na řádku 109 v souboru progressbar.class.php.

```
109                                    {
110       $this->countSteps = $steps;
111       $this->actualStep = 1;
112    }
```

8.74.2.3 ProgressBarEplugin::setWindowTitle ($ *title*)

Metoda nastavuje titulek okna.

Parametry:

string – titulek okna

Plánované úpravy

implementovat nastavení titulku okna

Definice je uvedena na řádku 130 v souboru progressbar.class.php.

```
131    {
132 //    Metoda nastavuje titulek okna
133    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/progressbar.class.php

8.75 Dokumentace třídy ProgressBarJs

Třída JsPluginu ProgressBarJs.

Diagram dědičnosti pro třídu ProgressBarJs

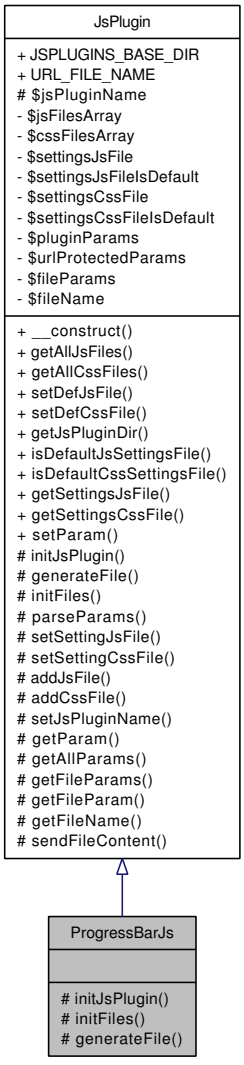
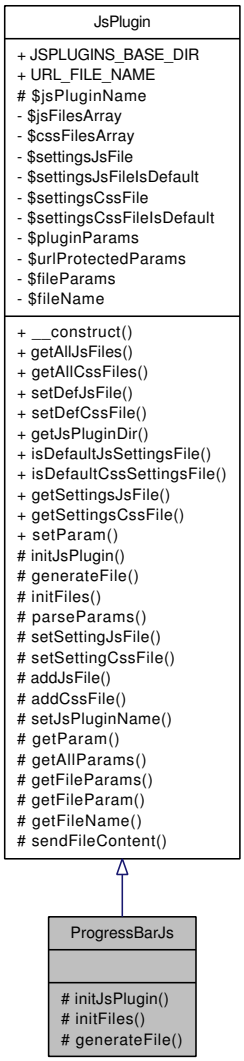


Diagram tříd pro ProgressBarJs:



Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- generateFile ()

  Metoda se využívá pro načtení proměných do stránky, je volána při volání parametru stránky pro JsPlugin a je pouze zpracována tato metoda (generování nastavení atd).

8.75.1 Detailní popis

Třída JsPluginu ProgressBarJs.

Třída slouží pro práci s progressbarem. Ten je otevřen v novém okně a zobrazuje průběh zpracování dat.

Verze:

$Id: progressbar.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu ProgressBarJs pro otevření okna s progresbarem

Definice je uvedena na řádku 13 v souboru progressbarjs.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/progressbarjs.class.php

8.76 Dokumentace třídy ReservationController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu ReservationController

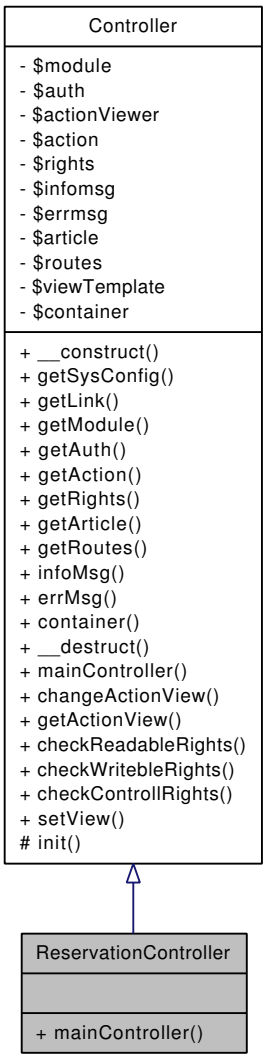
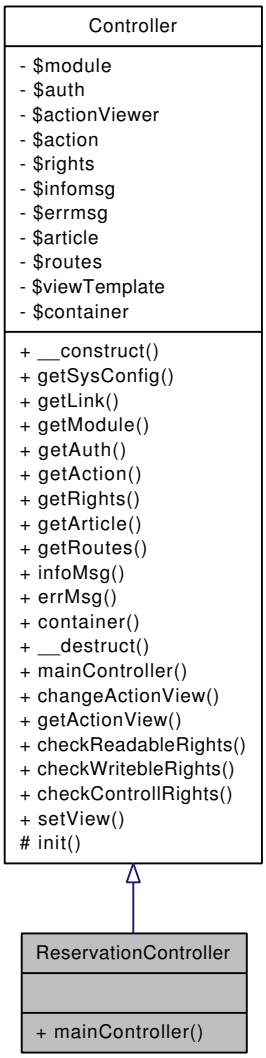


Diagram tříd pro ReservationController:



Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

8.76.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/reservation/controler.class.php

8.77 Dokumentace třídy ReservationView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu ReservationView

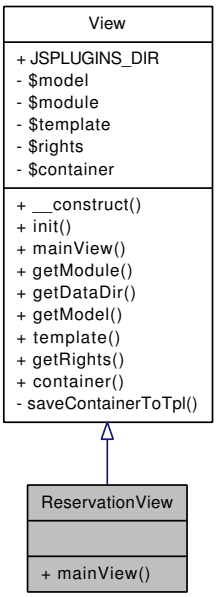
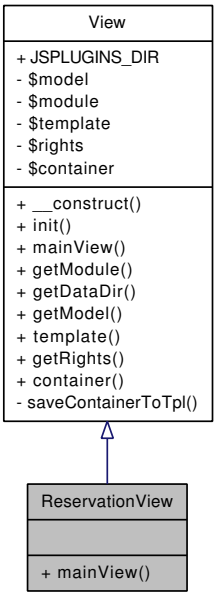


Diagram tříd pro ReservationView:



Veřejné metody

- mainView ()

Hlavní abstraktní třída pro vytvoření pohledu.

8.77.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/reservation/view.class.php

8.78 Dokumentace třídy Rights

Třída práv uživatele a jednotlivých kategorií.

Veřejné metody

- __construct (Auth $auth, $rights)

  Konstruktor.

- getAuth ()

  Metoda vrací přístup k objektu autorizace.

- isReadable ()

  Metoda vrací true pokud má uživatel právo číst.

- isWritable ()

  Metoda vrací true pokud má uživatel právo zapisovat.

- isControll ()

  Metoda vrací true pokud má uživatel plné právo.

Veřejné atributy

- const **RIGHTS_GROUPS_TABLE_PREFIX** = 'group_'

8.78.1 Detailní popis

Třída práv uživatele a jednotlivých kategorií.

Třída slouží pro práci s právy uživatele na zvolené kategorii. Umožňuje zjištění, jestli je kategorie přístupná pro čtení, zápis nebo kontrolu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: rights.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro zjišťování práv uživatele

Definice je uvedena na řádku 13 v souboru rights.class.php.

8.78.2 Dokumentace konstruktoru a destruktoru

8.78.2.1 Rights::__construct (Auth $ *auth*, $ *rights*)

Konstruktor.

Parametry:

Auth – objekt s autorizací

string – pole s práva všech skupin

Definice je uvedena na řádku 57 v souboru rights.class.php.

```
57                                                  {
58        $this->auth = $auth;
59        $this->groupRights = $rights;
60
61        $this->setRights();
62    }
```

8.78.3 Dokumentace k metodám

8.78.3.1 Rights::getAuth ()

Metoda vrací přístup k objektu autorizace.

Návratová hodnota:

Auth – objekt autorizace

Definice je uvedena na řádku 68 v souboru rights.class.php.

Používá se v Controller::__construct().

```
68                                {
69        return $this->auth;
70    }
```

8.78.3.2 Rights::isReadable () [final]

Metoda vrací true pokud má uživatel právo číst.

Návratová hodnota:

boolean – právo ke čtení

Definice je uvedena na řádku 93 v souboru rights.class.php.

```
93                                        {
94        return $this->read;
95    }
```

8.78.3.3 Rights::isWritable () [final]

Metoda vrací true pokud má uživatel právo zapisovat.

Návratová hodnota:

boolean – právo k zápisu

Definice je uvedena na řádku 101 v souboru rights.class.php.

```
101                                     {
102         return $this->write;
103     }
```

8.78.3.4 Rights::isControll () [final]

Metoda vrací true pokud má uživatel plné právo.

Návratová hodnota:

boolean – plné právo

Definice je uvedena na řádku 109 v souboru rights.class.php.

```
109                                     {
110         return $this->controll;
111     }
```

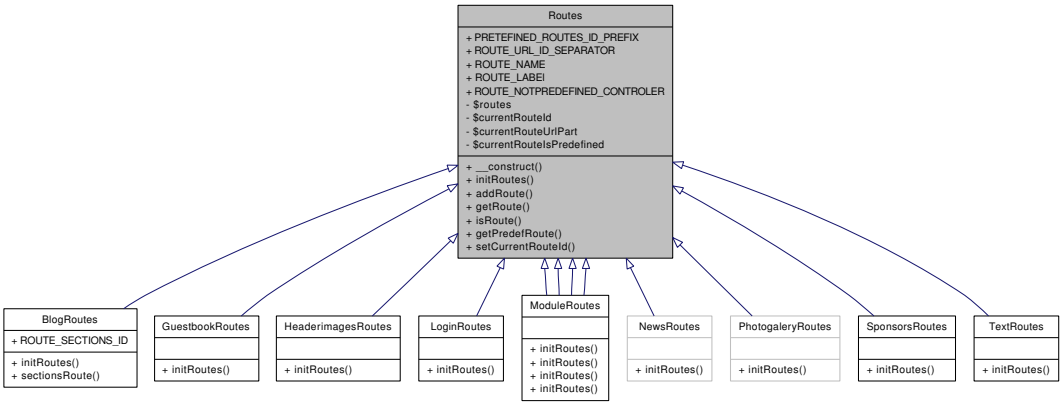
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/rights.class.php

8.79 Dokumentace třídy Routes

Třída pro obsluhu cest(routes).

Diagram dědičnosti pro třídu Routes



Veřejné metody

- __construct ()

  Konstruktor třídy.

- initRoutes ()

  Metoda, která nastavuje cesty.

- addRoute ($id, $routeName, $label)

  Metoda přidává cestu do seznamu cest.

- getRoute ()

  Metoda vrací použitou cestu.

- isRoute ()

  Metoda vrací true pokud je nastaveno cesta.

- getPredefRoute ($id)

  Metoda vrací informace o předdefinované cestě.

Statické veřejné metody

- static setCurrentRouteId ($id, $predefined=false)

  Nastavuje aktuální cestu.

Veřejné atributy

- const **PRETEFINED_ROUTES_ID_PREFIX** = 'p'
- const **ROUTE_URL_ID_SEPARATOR** = '-'
- const **ROUTE_NAME** = 'name'
- const **ROUTE_LABEl** = 'label'
- const **ROUTE_NOTPREDEFINED_CONTROLER** = 'rdefault'

8.79.1 Detailní popis

Třída pro obsluhu cest(routes).

Třida je určena k zjišťování a volby cesty pro kontroler a viewer. Také slouží pro generování vlastních cest jednotlivých modulů.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: routes.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu cest modulu

Definice je uvedena na řádku 13 v souboru routes.class.php.

8.79.2 Dokumentace konstruktoru a destruktoru

8.79.2.1 Routes::__construct ()

Konstruktor třídy.

Parametry:

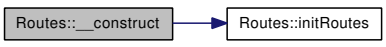
Article – objekt článku (article)

Definice je uvedena na řádku 73 v souboru routes.class.php.

Odkazuje se na initRoutes().

```
73                          {
74 //    $this->article = $article;
75       $this->initRoutes();
76    }
```

Tato funkce volá...



8.79.3 Dokumentace k metodám

8.79.3.1 static Routes::setCurrentRouteId ($ *id*, $ *predefined* = false) [static]

Nastavuje aktuální cestu.

Parametry:

integer $id – id cesty

bool $predefined – o jaký druh cesty se jedná (předdefinovaná x uživatelská)

Definice je uvedena na řádku 83 v souboru routes.class.php.

Používá se v Links::checkRouteUrlRequest().

```
83                                                                  {
84       //    Pokud je předdefinovaná cesta
85       if($predefined){
86          self::$currentRouteId = $id;
87          self::$currentRouteIsPredefined = true;
88       }
89       //    je použita výchozí cesta
90       else {
91          self::$currentRouteId = $id;
92          self::$currentRouteIsPredefined = false;
93       }
94    }
```

8.79.3.2 Routes::addRoute ($ *id*, $ *routeName*, $ *label*) [final]

Metoda přidává cestu do seznamu cest.

Parametry:

string – název cesty

Definice je uvedena na řádku 106 v souboru routes.class.php.

Používá se v BlogRoutes::initRoutes().

```
106                                                          {
107       $this->routes[$id] = array(self::ROUTE_NAME => $routeName, self::ROUTE_LABEl => $label);
108    }
```

8.79.3.3 Routes::getRoute () [final]

Metoda vrací použitou cestu.

Návratová hodnota:

string – název cesty

Definice je uvedena na řádku 114 v souboru routes.class.php.

```
114                                     {
115       if(self::$currentRouteIsPredefined){
116          return $this->routes[self::$currentRouteId][self::ROUTE_NAME];
117       } else {
118          return self::ROUTE_NOTPREDEFINED_CONTROLER;
119       }
120    }
```

8.79.3.4 Routes::isRoute ()

Metoda vrací true pokud je nastaveno cesta.

Návratová hodnota:

boolean – true pokud je cesta nastavena

Definice je uvedena na řádku 126 v souboru routes.class.php.

```
126                               {
127       if(self::$currentRouteId != null){
128          return true;
129       }
130       return false;
131    }
```

8.79.3.5 Routes::getPredefRoute ($ *id*)

Metoda vrací informace o předdefinované cestě.

Parametry:

integer $id – id routy

Návratová hodnota:

array – pole s informacemi o routě

Definice je uvedena na řádku 138 v souboru routes.class.php.

```
138                                       {
139         if(isset($this->routes[$id])){
140             $arr = array();
141
142             $arr[0] = $this->routes[$id][self::ROUTE_LABEl];
143             $arr[1] = $id;
144
145             return $arr;
146         }
147     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/routes.class.php

8.80 Dokumentace třídy ScrollEplugin

Třída Epluginu pro práci se scrolováním stránek.

Diagram dědičnosti pro třídu ScrollEplugin

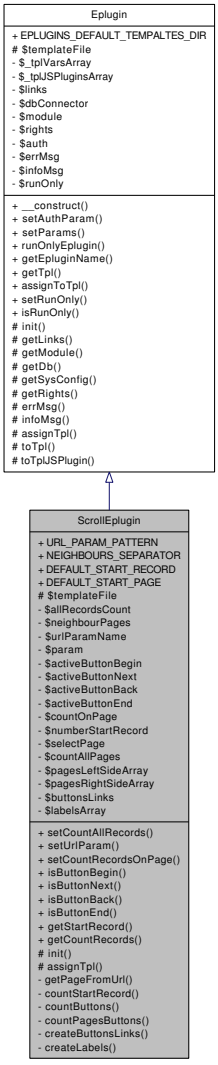
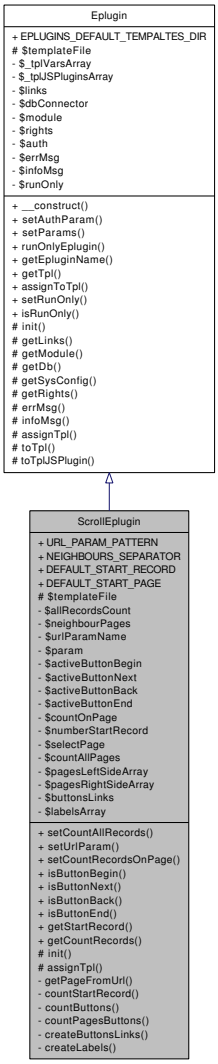


Diagram tříd pro ScrollEplugin:



Veřejné metody

- setCountAllRecords ($records)

  Metoda nastavuje celkový počet záznamů v db.

- setUrlParam ($urlParam)

  Metoda nastavuje parametr, kterým se přenáší stránka v url.

- setCountRecordsOnPage ($value)

  Funkce nastaví počet záznamů na stránce.

- isButtonBegin ()

  Metoda vrací jestli je tlačítko "Na začátek" aktivní.

- isButtonNext ()

Metoda vrací jestli je tlačítko "Další" aktivní.

- isButtonBack ()

  Metoda vrací jestli je tlačítko "Předchozí" aktivní.

- isButtonEnd ()

  Metoda vrací jestli je tlačítko "Na konec" aktivní.

- getStartRecord ()

  Metoda vrací od kterého záznamu se vypisuje.

- getCountRecords ()

  Metoda vrací počet záznamů na stránce.

Veřejné atributy

- const URL_PARAM_PATTERN = '([0-9]+)'

  Regulérní výraz pro parsování parametru s číslem stránky.

- const **NEIGHBOURS_SEPARATOR** = ' '
- const **DEFAULT_START_RECORD** = 0
- const **DEFAULT_START_PAGE** = 1

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda obstarává přiřazení proměných do šablony.

Chráněné atributy

- **$templateFile** = 'scroll.tpl'

8.80.1 Detailní popis

Třída Epluginu pro práci se scrolováním stránek.

Třída obsahuje metody pro práci s posunováním stránek. Je určena hlavně ke posunování mezi více stránkami (např. novinky). Obsahuje také vlastní šablonu, kterou lze jednoduše vložit do modulu, popřípadě si vytořit vlastní.

Verze:

$Id: scroll.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída EPluginu pro práci se scrolovátky

Definice je uvedena na řádku 15 v souboru scroll.class.php.

8.80.2 Dokumentace k metodám

8.80.2.1 ScrollEplugin::setCountAllRecords ($ *records*)

Metoda nastavuje celkový počet záznamů v db.

Parametry:

integer – celkový počet záznamů

Definice je uvedena na řádku 164 v souboru scroll.class.php.

```
165     {
166        $this->allRecordsCount = $records;
167 //     Výpočet jednotlivých prvku
168        $this->countStartRecord();
169        $this->countButtons();
170        $this->countPagesButtons();
171        $this->createButtonsLinks();
172     }
```

8.80.2.2 ScrollEplugin::setUrlParam ($ *urlParam*)

Metoda nastavuje parametr, kterým se přenáší stránka v url.

Parametry:

string

Definice je uvedena na řádku 178 v souboru scroll.class.php.

```
178                                         {
179        $this->urlParamName = $urlParam;
180
181          $this->param = new UrlParam($this->urlParamName, self::URL_PARAM_PATTERN);
182
183 //     Znovunačtení stránky z url
184        $this->getPageFromUrl();
185     }
```

8.80.2.3 ScrollEplugin::setCountRecordsOnPage ($ *value*)

Funkce nastaví počet záznamů na stránce.

Parametry:

int – počet záznamů na stránce

Definice je uvedena na řádku 205 v souboru scroll.class.php.

```
206    {
207       $this->countOnPage = $value;
208    }
```

8.80.2.4 ScrollEplugin::isButtonBegin ()

Metoda vrací jestli je tlačítko "Na začátek" aktivní.

Návratová hodnota:

boolean – true pokud je tlačítko aktivní

Definice je uvedena na řádku 326 v souboru scroll.class.php.

Používá se v assignTpl().

```
327    {
328       return $this->activeButtonBegin;
329    }
```

8.80.2.5 ScrollEplugin::isButtonNext ()

Metoda vrací jestli je tlačítko "Další" aktivní.

Návratová hodnota:

boolean – true pokud je tlačítko aktivní

Definice je uvedena na řádku 336 v souboru scroll.class.php.

Používá se v assignTpl().

```
337    {
338       return $this->activeButtonNext;
339    }
```

8.80.2.6 ScrollEplugin::isButtonBack ()

Metoda vrací jestli je tlačítko "Předchozí" aktivní.

Návratová hodnota:

boolean – true pokud je tlačítko aktivní

Definice je uvedena na řádku 346 v souboru scroll.class.php.

Používá se v assignTpl().

```
347    {
348       return $this->activeButtonBack;
349    }
```

8.80.2.7 ScrollEplugin::isButtonEnd ()

Metoda vrací jestli je tlačítko "Na konec" aktivní.

Návratová hodnota:

boolean – true pokud je tlačítko aktivní

Definice je uvedena na řádku 356 v souboru scroll.class.php.

Používá se v assignTpl().

```
357     {
358        return $this->activeButtonEnd;
359     }
```

8.80.2.8 ScrollEplugin::getStartRecord ()

Metoda vrací od kterého záznamu se vypisuje.

Návratová hodnota:

integer – strt record

Definice je uvedena na řádku 366 v souboru scroll.class.php.

```
367     {
368        return $this->numberStartRecord;
369     }
```

8.80.2.9 ScrollEplugin::getCountRecords ()

Metoda vrací počet záznamů na stránce.

Návratová hodnota:

ineteger – počet záznamů

Definice je uvedena na řádku 375 v souboru scroll.class.php.

```
376     {
377        return $this->countOnPage;
378     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/scroll.class.php

8.81 Dokumentace třídy SendMailEplugin

Třída EPluginu pro kládání mailů na které se bude odesílat.

Diagram dědičnosti pro třídu SendMailEplugin

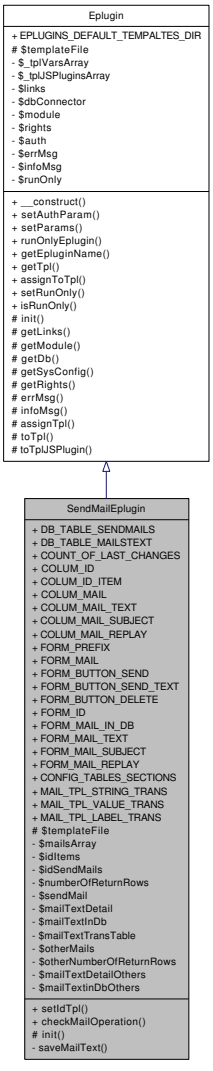
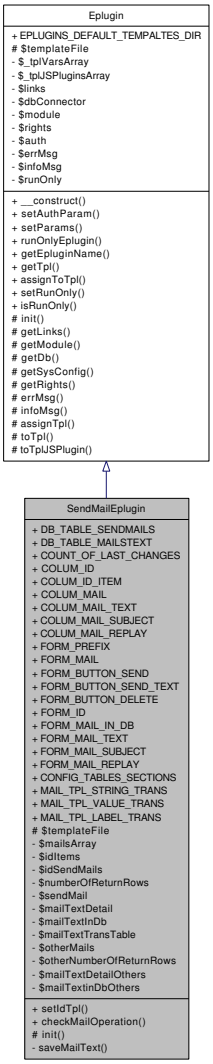


Diagram tříd pro SendMailEplugin:



Veřejné metody

- setIdTpl ($id)

  Metoda nastaví id šablony pro výpis.

- checkMailOperation ()

  Metoda kontroluje, pokud byl mail přidáván nebo mazán.

Veřejné atributy

- const **DB_TABLE_SENDMAILS** = 'sendmails'
- const **DB_TABLE_MAILSTEXT** = 'sendmailstexts'
- const **COUNT_OF_LAST_CHANGES** = 1

- const **COLUM_ID** = 'id_mail'
- const **COLUM_ID_ITEM** = 'id_item'
- const **COLUM_MAIL** = 'mail'
- const **COLUM_MAIL_TEXT** = 'text'
- const **COLUM_MAIL_SUBJECT** = 'subject'
- const **COLUM_MAIL_REPLAY** = 'replay_mail'
- const **FORM_PREFIX** = 'sendmail_'
- const **FORM_MAIL** = 'mail'
- const **FORM_BUTTON_SEND** = 'send'
- const **FORM_BUTTON_SEND_TEXT** = 'send_text'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **FORM_ID** = 'id'
- const **FORM_MAIL_IN_DB** = 'in_db'
- const **FORM_MAIL_TEXT** = 'text'
- const **FORM_MAIL_SUBJECT** = 'subject'
- const **FORM_MAIL_REPLAY** = 'replay_mail'
- const **CONFIG_TABLES_SECTIONS** = 'db_tables'
- const **MAIL_TPL_STRING_TRANS** = 'string'
- const **MAIL_TPL_VALUE_TRANS** = 'value'
- const **MAIL_TPL_LABEL_TRANS** = 'label'

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

Chráněné atributy

- **$templateFile** = array('editmail.tpl','sendmail.tpl')

8.81.1 Detailní popis

Třída EPluginu pro kládání mailů na které se bude odesílat.

Třída slouží pro zprávu emailových adresa, na které se má například odesílat nabídka. Obsahuje vlastní šablonu s editovatelným seznamem emailů.

Verze:

$Id: sendmail.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída Epluginu pro práci s maily k odesílání informací

Plánované úpravy

Dodělat podporu pro více druhů výrazů (ne jenom vyraz%, vyraz[true/false]%) u zpracování

Definice je uvedena na řádku 14 v souboru sendmail.class.php.

8.81.2 Dokumentace k metodám

8.81.2.1 SendMailEplugin::setIdTpl ($ *id*)

Metoda nastaví id šablony pro výpis.

Parametry:

ineger – id šablony (jakékoliv)

Definice je uvedena na řádku 137 v souboru sendmail.class.php.

```
137                                 {
138       $this->idSendMails = $id;
139     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/sendmail.class.php

8.82 Dokumentace třídy Sessions

Třída pro práci se $_SESSIONS.

Veřejné metody

- __construct ($sessionName= 'default_session')

  Konstruktor vytvoří objekt pro práci se session.

- add ($name, $value)

  Metoda uloží proměnou do session.

- get ($name)

  Metoda vrací obsah zadané session.

- isEmpty ($name)

  Metoda zjišťuje jestli je daná session prázdná.

- remove ($name)

  Metoda odstraní zadanou session.

- commit ()

  Metoda uloží session a znovu ji načte.

Statické veřejné metody

- static factory ($sessionName)

  Statická metoda pro nastavení session.

8.82.1 Detailní popis

Třída pro práci se $_SESSIONS.

Třída umožňuje základní přístupy k Sessions, jejich vytváření, mazání, aktualizaci atd.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: sessions.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci se SESSIONS

//TODO dodělat! není skoro implementována

Definice je uvedena na řádku 14 v souboru sessions.class.php.

8.82.2 Dokumentace konstruktoru a destruktoru

8.82.2.1 Sessions::__construct ($ *sessionName* = 'default_session')

Konstruktor vytvoří objekt pro práci se session.

Parametry:

string – název session (option)

Definice je uvedena na řádku 51 v souboru sessions.class.php.

```
51                                                          {
52       if(!self::$sessionInitalized){
53          self::factory($sessionName);
54       }
55    }
```

8.82.3 Dokumentace k metodám

8.82.3.1 static Sessions::factory ($ *sessionName*) [static]

Statická metoda pro nastavení session.

Parametry:

string – název session do ketré se bude ukládat

Definice je uvedena na řádku 25 v souboru sessions.class.php.

```
25                                              {
26       //Nastaveni session
27       session_regenerate_id(); // ochrana před Session Fixation
28       //    Nastaveni limutu pro automaticke odhlaseni
29       /* set the cache limiter to 'private' */
30
31       session_cache_limiter('private');
32       $cache_limiter = session_cache_limiter();
33
34       /* set the cache expire to 30 minutes */
35       session_cache_expire(30);
36       $cache_expire = session_cache_expire();
37
38       //session_set_cookie_params(1800);
39       session_name($sessionName);
40       session_start();
41
42       self::$sessionInitalized = true;
43    }
```

8.82.3.2 Sessions::add ($ *name*, $ *value*)

Metoda uloží proměnou do session.

Parametry:

string – název proměné

mixed – hodnota proměné

Definice je uvedena na řádku 63 v souboru sessions.class.php.

```
63                                      {
64        $_SESSION[$name] = $value;
65    }
```

8.82.3.3 Sessions::get ($ *name*)

Metoda vrací obsah zadané session.

Parametry:

string – název session

Návratová hodnota:

mixed – obsah session

Definice je uvedena na řádku 73 v souboru sessions.class.php.

```
73                              {
74        //TODO dodělat ověřování atd
75        if(isset($_SESSION[$name])){
76           return $_SESSION[$name];
77        } else {
78           return null;
79        }
80    }
```

8.82.3.4 Sessions::isEmpty ($ *name*)

Metoda zjišťuje jestli je daná session prázdná.

Parametry:

string – název session

Definice je uvedena na řádku 88 v souboru sessions.class.php.

```
88                                    {
89        if(isset($_SESSION[$name])){
90           return false;
91        } else {
92           return true;
93        }
94    }
```

8.82.3.5 Sessions::remove ($ *name*)

Metoda odstraní zadanou session.

Parametry:

string – název session

Definice je uvedena na řádku 101 v souboru sessions.class.php.

```
101                                 {
102         if(isset($_SESSION[$name])){
103            $_SESSION[$name] = null;
104            unset($_SESSION[$name]);
105         }
106     }
```

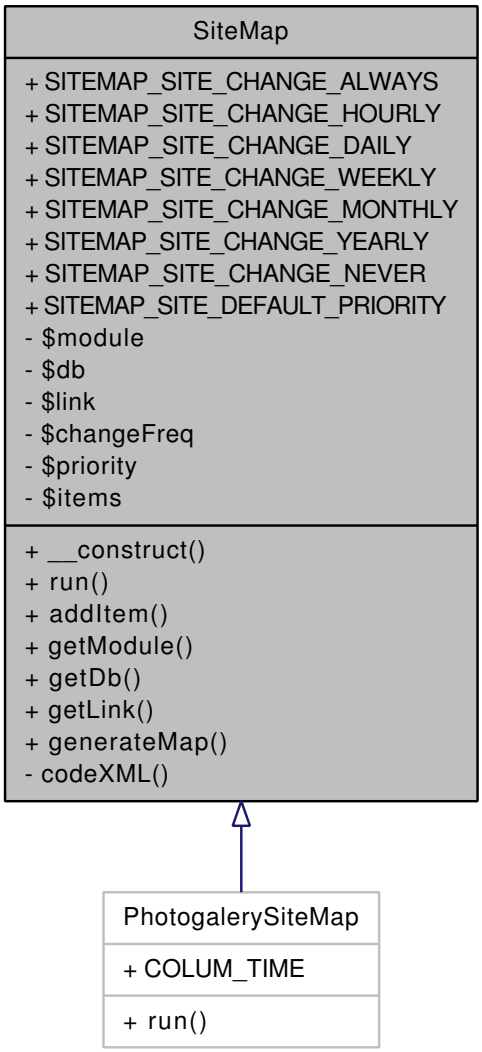
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/sessions.class.php

8.83 Dokumentace třídy SiteMap

Třída pro generování sitemapy.

Diagram dědičnosti pro třídu SiteMap



Veřejné metody

- __construct (Module $module, Links $link, $changefreq, $priority)

  Konstruktor – vytvoří prostředí pro práci se sitemap.

- run ()

  Metoda spouští proceduru pro přidávání položek do sitemap.

- addItem ($url, $lastChange, $frequency=null, $priority=null)

  Metoda přidává položku do siteamp.

- getModule ()

  Meotda vrací objekt modulu.

- getDb ()

  Metoda vrací objekt pro přístup k db.

- getLink ()

  Metoda vrací objekt odkazu.

Statické veřejné metody

- static generateMap ($mapType= 'google')

  Metoda vygeneruje mapu webu.

Veřejné atributy

- const **SITEMAP_SITE_CHANGE_ALWAYS** = 'always'
- const **SITEMAP_SITE_CHANGE_HOURLY** = 'hourly'
- const **SITEMAP_SITE_CHANGE_DAILY** = 'daily'
- const **SITEMAP_SITE_CHANGE_WEEKLY** = 'weekly'
- const **SITEMAP_SITE_CHANGE_MONTHLY** = 'monthly'
- const **SITEMAP_SITE_CHANGE_YEARLY** = 'yearly'
- const **SITEMAP_SITE_CHANGE_NEVER** = 'never'
- const **SITEMAP_SITE_DEFAULT_PRIORITY** = 0.1

8.83.1 Detailní popis

Třída pro generování sitemapy.

Třída generuje mapu webu v požadovaném formátu. Podporovány jsou formát pro google (seznam) a yahoo. Je většinou volána zvlášť a využívá soubor sitemap.class.php v modulech pro generování pro generování data poslední změny atd.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: sitemap.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro tvorbu sitemap

Definice je uvedena na řádku 14 v souboru sitemap.class.php.

8.83.2 Dokumentace konstruktoru a destruktoru

8.83.2.1 SiteMap::__construct (Module $ *module*, Links $ *link*, $ *changefreq*, $ *priority*)

Konstruktor – vytvoří prostředí pro práci se sitemap.

Parametry:

Module – objekt modulu

Definice je uvedena na řádku 74 v souboru sitemap.class.php.

Odkazuje se na AppCore::getDbConnector().

```
74                                                                                  {
75        $this->module = $module;
76        $this->db = AppCore::getDbConnector();
77        $this->link = $link;
78
79        $this->changeFreq = $changefreq;
80        $this->priority = $priority;
81    }
```

Tato funkce volá...

8.83.3 Dokumentace k metodám

8.83.3.1 SiteMap::addItem ($ *url*, $ *lastChange*, $ *frequency* = null, $ *priority* = null)

Metoda přidává položku do siteamp.

Parametry:

string – odkaz

integer – čas poslední změny (timestamp)

string – četnost změny (kostanta SITEMAP_SITE_CHANGE_...)

float – priorita (0 - 1)

Definice je uvedena na řádku 99 v souboru sitemap.class.php.

Používá se v run().

```
99                                                                                  {
100       $date = new DateTime(date(DATE_ISO8601,$lastChange));
101
102       if($frequency == null){
103          $frequency = $this->changeFreq;
104       }
105       if($priority == null){
106          $priority = $this->priority;
107       }
108
109       array_push(self::$items, array('loc' => $url,
```

```
110                                 'lastmod' => $date->format(DATE_ISO8601),
111                                 'changefreq' => $frequency,
112                                 'priority'=>$priority));
113     }
```

8.83.3.2 SiteMap::getModule ()

Meotda vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 120 v souboru sitemap.class.php.

```
120                                 {
121         return $this->module;
122     }
```

8.83.3.3 SiteMap::getDb ()

Metoda vrací objekt pro přístup k db.

Návratová hodnota:

DbInterface – objekt databáze

Definice je uvedena na řádku 129 v souboru sitemap.class.php.

```
129                             {
130         return $this->db;
131     }
```

8.83.3.4 SiteMap::getLink ()

Metoda vrací objekt odkazu.

Návratová hodnota:

Links – objekt odkazu

Definice je uvedena na řádku 137 v souboru sitemap.class.php.

Používá se v run().

```
137                                 {
138         return $this->link;
139     }
```

8.83.3.5 static SiteMap::generateMap ($ *mapType* = 'google') [static]

Metoda vygeneruje mapu webu.

Parametry:

string – pro jaký vahledávač má být mapa generována

Návratová hodnota:

string – bufer s vygenerovanou mapou

Definice je uvedena na řádku 148 v souboru sitemap.class.php.

```
148                                                         {
149       if($mapType == 'google'){
150          ob_start();
151          header('Content-type: text/xml');
152          echo '<?xml version="1.0" encoding="UTF-8"?>'."\n";
153 //       echo '<urlset xmlns="http://www.google.com/schemas/sitemap/0.84">'."\n";
154          echo '<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">'."\n";
155          foreach (self::$items as $i)
156          {
157             echo '<url>';
158             foreach ($i as $index => $_i)
159             {
160                if (!$_i) continue;
161                echo "<$index>" . self::codeXML($_i) . "</$index>\n";
162             }
163             echo "</url>\n";
164          }
165          echo '</urlset>';
166          return ob_get_clean();
167       }
168 //    Pro yahoo
169       else if($mapType == 'yahoo'){
170          ob_start();
171          header('Content-type: text/plain');
172          foreach (self::$items as $i)
173          {
174             echo $i['loc'] . "\n";
175          }
176          return ob_get_clean();
177       }
178    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/sitemap.class.php

8.84 Dokumentace třídy SpecialFunctions

Třidá se speciálními funkcemi.

Veřejné metody

- __construct ()
  Konstruktor objektu se specielními funkcemi.
- utf2ascii ($text)
  Funkce odstrani nekorektni znaky z řetězce.
- createDatabaseKey ($text, $keysArray=null, $minLenght=9, $maxLenght=50)
  Funkce vytvoří klíč o velikosti 50 znaků, který je určen pro uložení do db.
- removeAllTags ($text)
  funkce odtraní z textu všechny tagy
- decodeSpecialChars ($text)
  Funkce převede některé speciílní znaky na alternativy html použito v komentářích.
- czechTypo ($text)
  Funkce převede předložky a některé znaky s normální mezerou na nezalomitelné mezery.
- getMicroTime ()
  Funkce pro generování mikročasu.
- load_passwd ($passwd_file)
  Funkce nacte heslo ze souboru Heslo je ulozeno v md5 souctu.
- isInt ($to_validate)
  Funkce zjišťuje jestli je číslo typu integer //TODO není třeba.
- deltree ($f)
  Funkce maže zadaný adresář z filesystému i s obsahem.
- createUrl ($url)
  Metoda vytvoří za zadaného řetězce url adresu.

8.84.1 Detailní popis

Třidá se speciálními funkcemi.

Obsahuje některé funkce, které zjednodušují práci s řetězci.

Copyright (c) 2008 Jakub Matas

Verze:

$Id:$ VVE3.3.0 $Revision:$

Autor:

$Author:$ $Date:$ $LastChangedBy:$ $LastChangedDate:$ Třída se speciálními funkcemi

Zastaralé

nehrtazuje se ostatními helpery a validátory zvláště texthelperem

Definice je uvedena na řádku 14 v souboru specialfunctions.class.php.

8.84.2 Dokumentace konstruktoru a destruktoru

8.84.2.1 SpecialFunctions::__construct ()

Konstruktor objektu se specielními funkcemi.

Parametry:

Db – objekt pro přístup k databázi

Module – objekt pro přístup k vlastnostem modulu

Definice je uvedena na řádku 27 v souboru specialfunctions.class.php.

```
27                              {
28     }
```

8.84.3 Dokumentace k metodám

8.84.3.1 SpecialFunctions::utf2ascii ($ *text*)

Funkce odstrani nekorektni znaky z řetězce.

Parametry:

string – řetězec z kterého se odstraní znaky

Návratová hodnota:

string – výsledný řetězec

Definice je uvedena na řádku 36 v souboru specialfunctions.class.php.

Používá se v createDatabaseKey().

```
37     {
38        $return = Str_Replace(
39        Array("á","č","d’","é","ě","í","l’","ň","ó","ř","š","t’","ú","ů","ý","ž","Á","Č","Ď","É","Ě","Í",
40        Array("a","c","d","e","e","i","l","n","o","r","s","t","u","u","y","z","A","C","D","E","E","I","L"
41        $text);
42
43        $return = Str_Replace(Array(" ", "_"), "-", $return); //nahradí mezery a podtržítka pomlčkami
44        $return = Str_Replace(array("----","---","--"), "-", $return); //odstraní nekolik pomlcek za sebo
45        $return = Str_Replace(Array("(",")",".","!",",","\"","’"), "", $return); //odstraní ().!,"’
46        $return = StrToLower($return); //velká písmena nahradí malými.
47        return $return;
48     }
```

8.84.3.2 SpecialFunctions::createDatabaseKey ($ *text*, $ *keysArray* = null, $ *minLenght* = 9, $ *maxLenght* = 50)

Funkce vytvoří klíč o velikosti 50 znaků, který je určen pro uložení do db.

Parametry:

string – text ze kterého se má klíč vytvořit

array – pole již existujících klíčů

integer – minimální počet znaků

integer – maximální počet znaků

Návratová hodnota:

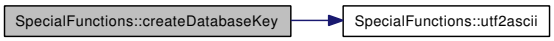
string – vygenerovaný klíč pro db

Definice je uvedena na řádku 59 v souboru specialfunctions.class.php.

Odkazuje se na utf2ascii().

```
60    {
61       // odstranění tagů
62       $text=ereg_replace("<[^>]+>", "", $text);
63
64       // pevod na ascii
65       $newKey = $this->utf2ascii($text);
66       $newKey = substr($newKey, 0, $maxLenght);
67
68       // Porpvnání klíču již uloženyých a vytvoření unikátního
69       if($keysArray != null){
70          $step = 1;
71          $newUniqueKey = $newKey;
72          $uniqueKey = false;
73
74          while ($uniqueKey != true) {
75             if(!in_array($newUniqueKey, $keysArray)){
76                $uniqueKey = true;
77             } else {
78                $newUniqueKey=$newKey."-".$step++;
79             }
80             //       echo $newUniqueKey."-".$step."<br>";
81          }
82          $newKey=$newUniqueKey;
83       }
84
85       if(strlen($newKey) < 6){
86          $newKey = str_pad($newKey, $minLenght, ".", STR_PAD_BOTH);
87       }
88       // echo $newKey;
89       return $newKey;
90    }
```

Tato funkce volá...



8.84.3.3 SpecialFunctions::removeAllTags ($ *text*)

funkce odtraní z textu všechny tagy

Parametry:

string – textový řetězec

Návratová hodnota:

string – text bez tagů

Definice je uvedena na řádku 98 v souboru specialfunctions.class.php.

```
99    {
100       // odstranění tagů
101       $text=ereg_replace("<[^>]+>", "", $text);
102       return $text;
103    }
```

8.84.3.4 SpecialFunctions::decodeSpecialChars ($ *text*)

Funkce převede některé speciílní znaky na alternativy html použito v komentářích.

Parametry:

string – zadaný text

Návratová hodnota:

string – text. u kterého jsou znaky převedeny

Definice je uvedena na řádku 113 v souboru specialfunctions.class.php.

```
114    {
115       $trans = array("#"=>"&#035;", "$"=>"&#036;","&"=>"&amp;","/"=>"&#047;","<"=>"&lt;",">"=>"&gt;","
116             "{"=>"&#123;","}"=>"&#125;","["=>"&#091;","]"=>"&#093;", "\\" =>"&#092;");
117
118       $preklad = strtr($text, $trans);
119       return $preklad;
120    }
```

8.84.3.5 SpecialFunctions::czechTypo ($ *text*)

Funkce převede předložky a některé znaky s normální mezerou na nezalomitelné mezery.

Parametry:

string – zadaný text

Návratová hodnota:

string – text. u kterého jsou převedeny předložky

Definice je uvedena na řádku 128 v souboru specialfunctions.class.php.

```
129    {
130       $czechPripositions = "(k|s|v|z|a|i|o|u|ve|ke|ku|za|ze|na|do|od|se|po|pod|před|nad|bez|pro|při|In
131
132       // překlad předložek na konci řádku
133       $pattern = "[[:blank:]]{1}".$czechPripositions."[[:blank:]]{1}";
134       $replacement = " \\1 ";
135       $text = eregi_replace($pattern, $replacement, $text);
136
137       $pattern = "&[a-z]+;".$czechPripositions."[[:blank:]]{1}";
138       $replacement = " \\1 ";
139       $text = eregi_replace($pattern, $replacement, $text);
140
141       // zkratky množin
142       $pattern = "([0-9])[[:blank:]]{1}(kč|C|V|A|W)";
143       $replacement = "\\1 \\2";
144       $text = eregi_replace($pattern, $replacement, $text);
145
146       //mezera mezi číslovkami
147       $pattern = "([0-9])([[:blank:]]{1})([0-9]{3})";
148       $replacement = "\\1 \\3";
149       $text = ereg_replace($pattern, $replacement, $text);
150       return $text;
151    }
```

8.84.3.6 SpecialFunctions::getMicroTime ()

Funkce pro generování mikročasu.

Návratová hodnota:

integer – mikročas v milisekundách

Definice je uvedena na řádku 158 v souboru specialfunctions.class.php.

```
159    {
160       List ($usec, $sec) = Explode (' ', microtime());
161       return ((float)$sec + (float)$usec);
162    }
```

8.84.3.7 SpecialFunctions::load_passwd ($*passwd_file*)

Funkce nacte heslo ze souboru Heslo je ulozeno v md5 souctu.

Parametry:

string – soubor s heslem

Návratová hodnota:

string – při úspěchu vrací string, jinak false

Definice je uvedena na řádku 171 v souboru specialfunctions.class.php.

```
172    {
173
174       if (file_exists($passwd_file))
175       {
176          if ($soubor = fopen($passwd_file, "r"))
177          {
178             $passwd_md5=fread($soubor,32);
179             fclose($soubor);
180             return $passwd_md5;
181          }
182          else
183          return 1;
184       }
185       else
186       return 1;
187    }
```

8.84.3.8 SpecialFunctions::isInt ($ *to_validate*)

Funkce zjišťuje jestli je číslo typu integer //TODO není třeba.

Parametry:

mix – proměná, která se má ověřit

Návratová hodnota:

boolean – vrací trua, jestliže je integer, jinak false

Definice je uvedena na řádku 195 v souboru specialfunctions.class.php.

```
196    {
197       $RegExp = "/^[-+]?\d+$/";
198       return (boolean)preg_match($RegExp,$to_validate);
199    }
```

8.84.3.9 SpecialFunctions::deltree ($ *f*)

Funkce maže zadaný adresář z filesystému i s obsahem.

Parametry:

dir – adresář, který se má smazat

Definice je uvedena na řádku 206 v souboru specialfunctions.class.php.

```
206                                {
207       foreach(glob($f.'/*') as $sf){
208          if (is_dir($sf) && !is_link($sf)){
209             deltree($sf);
210             rmdir($sf);
211          }else{
212             unlink($sf);
213          }
214       }
215    }
```

8.84.3.10 SpecialFunctions::createUrl ($ *url*)

Metoda vytvoří za zadaného řetězce url adresu.

Parametry:

string – řeťezec s url

Návratová hodnota:

string – opravená url

Definice je uvedena na řádku 223 v souboru specialfunctions.class.php.

```
223                                     {
224         $isProtoHeader = false;
225
226         foreach ($this->protocolsArray as $protocol) {
227            $strLen = strlen($protocol);
228            if (strncasecmp($url, $protocol, $strLen) == 0){
229               $isProtoHeader = true;
230            }
231         }
232
233         if(!$isProtoHeader){
234            $url = $this->protocolsArray[0].$url;
235         }
236
237         return $url;
238      }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/specialfunctions.class.php

8.85 Dokumentace třídy SponsorsAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu SponsorsAction

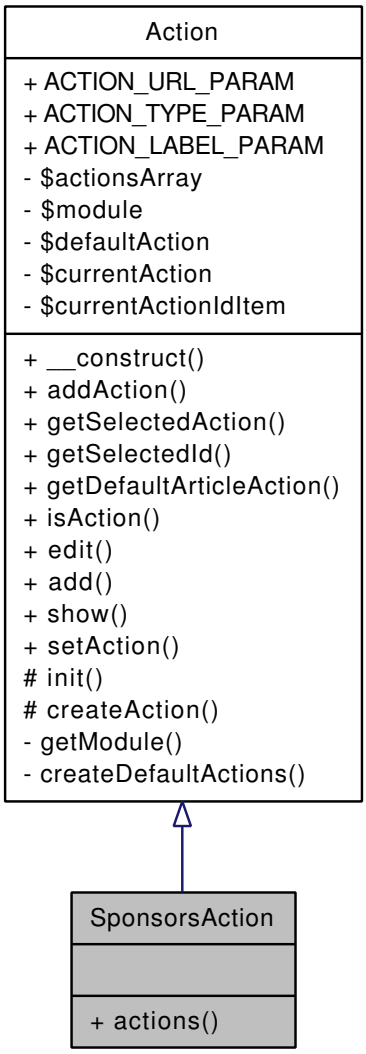
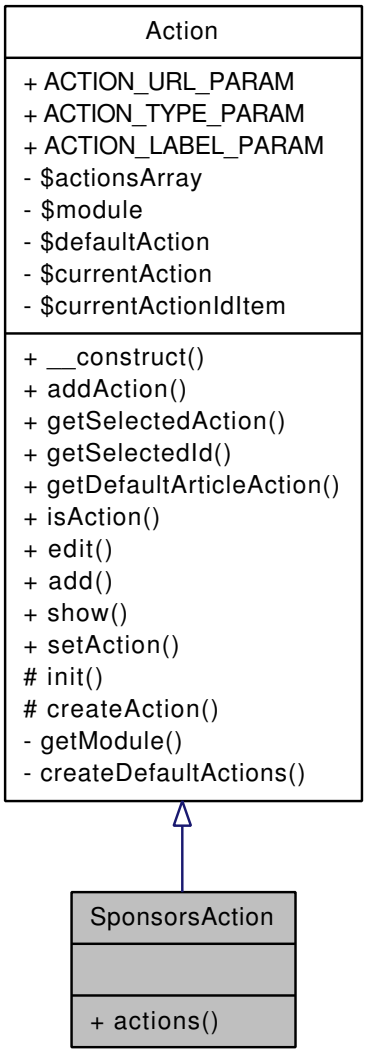


Diagram tříd pro SponsorsAction:



Veřejné metody

- **actions** ()

8.85.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/sponsors/action.class.php

8.86 Dokumentace třídy SponsorsController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu SponsorsController

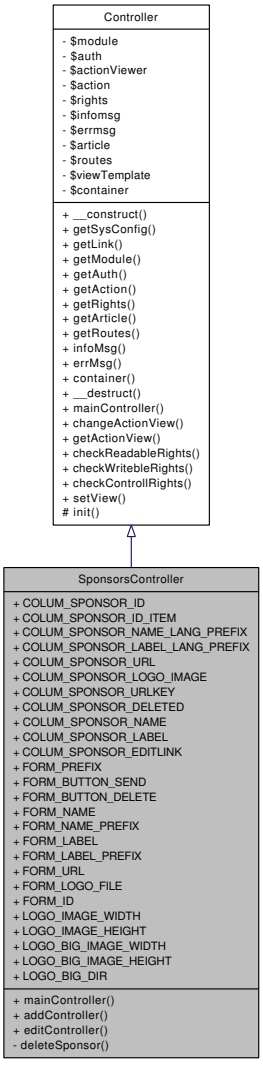
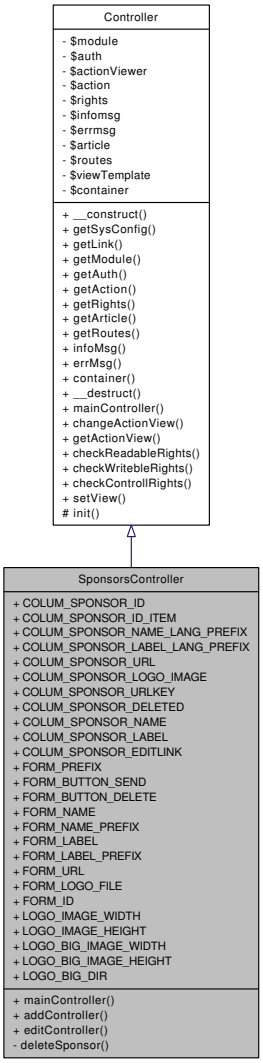


Diagram tříd pro SponsorsController:



Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- addController ()

  Kontroler pro obsluhu přidání sponzora.

- editController ()

  Controler pro úpravu spozora.

Veřejné atributy

- const **COLUM_SPONSOR_ID** = 'id_sponsor'

- const **COLUM_SPONSOR_ID_ITEM** = 'id_item'
- const **COLUM_SPONSOR_NAME_LANG_PREFIX** = 'name_'
- const **COLUM_SPONSOR_LABEL_LANG_PREFIX** = 'label_'
- const **COLUM_SPONSOR_URL** = 'url'
- const **COLUM_SPONSOR_LOGO_IMAGE** = 'logo_image'
- const **COLUM_SPONSOR_URLKEY** = 'urlkey'
- const **COLUM_SPONSOR_DELETED** = 'deleted'
- const **COLUM_SPONSOR_NAME** = 'name'
- const **COLUM_SPONSOR_LABEL** = 'label'
- const **COLUM_SPONSOR_EDITLINK** = 'editlink'
- const **FORM_PREFIX** = 'sponsor_'
- const **FORM_BUTTON_SEND** = 'send'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **FORM_NAME** = 'name'
- const **FORM_NAME_PREFIX** = 'name_'
- const **FORM_LABEL** = 'label'
- const **FORM_LABEL_PREFIX** = 'label_'
- const **FORM_URL** = 'url'
- const **FORM_LOGO_FILE** = 'logo_file'
- const **FORM_ID** = 'id'
- const **LOGO_IMAGE_WIDTH** = 60
- const **LOGO_IMAGE_HEIGHT** = 40
- const **LOGO_BIG_IMAGE_WIDTH** = 120
- const **LOGO_BIG_IMAGE_HEIGHT** = 80
- const **LOGO_BIG_DIR** = 'big'

8.86.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/sponsors/controler.class.php

8.87 Dokumentace třídy SponsorsRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu SponsorsRoutes

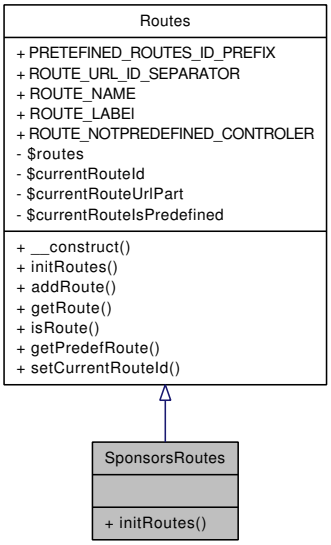
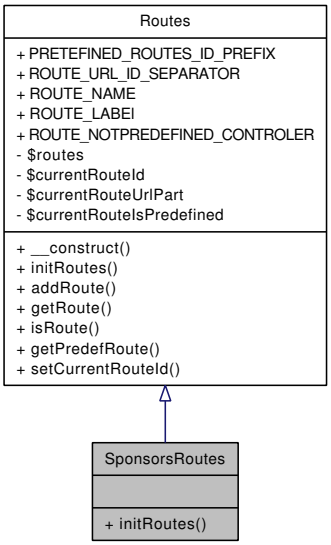


Diagram tříd pro SponsorsRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

8.87.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/sponsors/routes.class.php

8.88 Dokumentace třídy SponsorsView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu SponsorsView

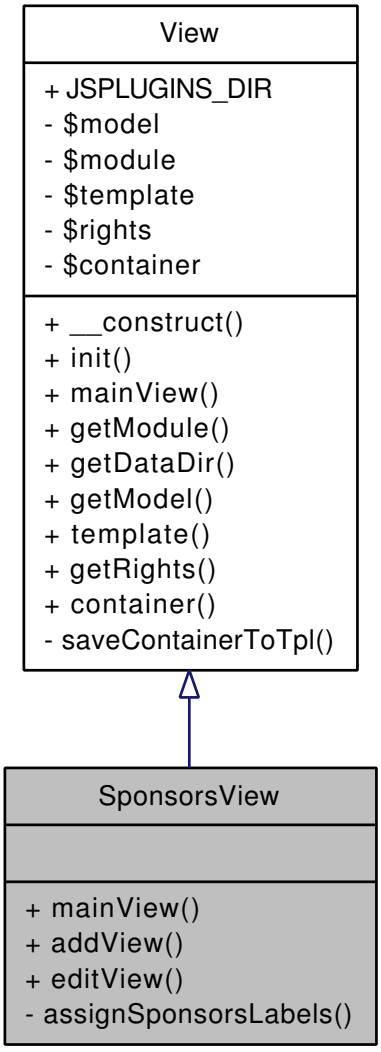
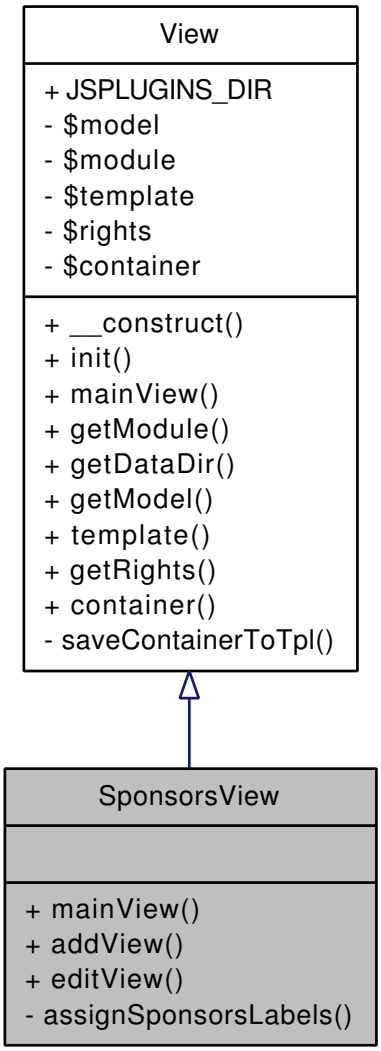


Diagram tříd pro SponsorsView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- addView ()

  Viewer pro přidání sponzora.

- editView ()

  Viewver pro zobrazení editace.

8.88.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/sponsors/view.class.php

8.89 Dokumentace třídy SubmitForm

Třída JsPluginu SubmitForm.

Diagram dědičnosti pro třídu SubmitForm

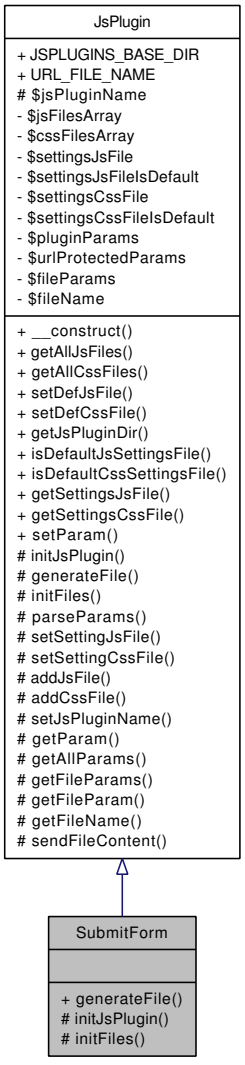
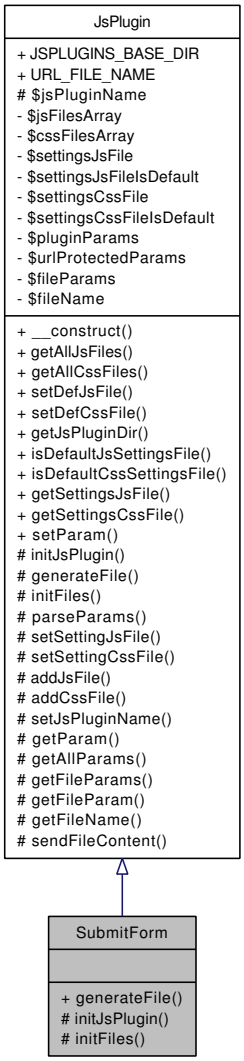


Diagram tříd pro SubmitForm:



Veřejné metody

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

Chráněné metody

- initJsPlugin ()

  Metoda inicializuje JsPlugin.

- initFiles ()

  Metoda inicializuje soubory JsPluginu.

8.89.1 Detailní popis

Třída JsPluginu SubmitForm.

Třídá vkádá funkci pro tvorbu potvrzovacího dialogu.

Verze:

$Id: submitform.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu SubmitForm na potvrzení formuláře

Definice je uvedena na řádku 12 v souboru submitform.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/submitform.class.php

8.90 Dokumentace třídy SwitchContentEasy

Třída JsPluginu SwitchContent.

Diagram dědičnosti pro třídu SwitchContentEasy

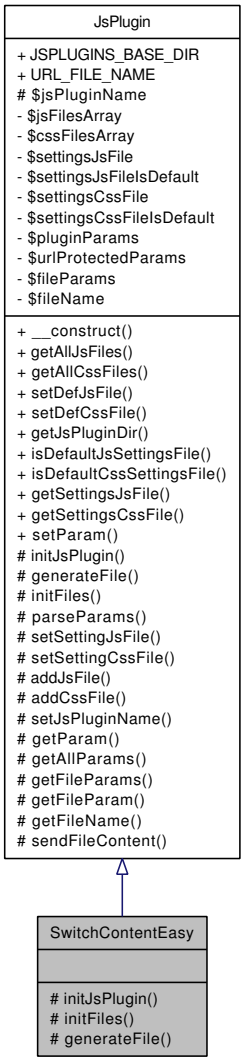
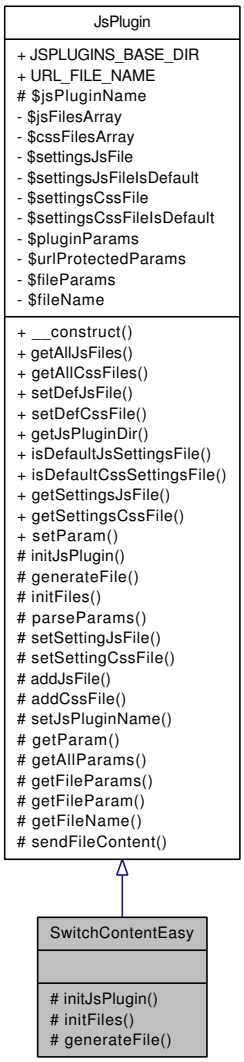


Diagram tříd pro SwitchContentEasy:



Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

8.90.1 Detailní popis

Třída JsPluginu SwitchContent.

Třída vytváří plugin pro tvorbu rozbalovacího prvku (např. div).

Verze:

$Id: switchcontenteasy.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu pro rozbalovací box

Definice je uvedena na řádku 12 v souboru switchcontenteasy.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/switchcontenteasy.class.php

8.91 Dokumentace třídy TabContent

Třída JsPluginu TabContent.

Diagram dědičnosti pro třídu TabContent

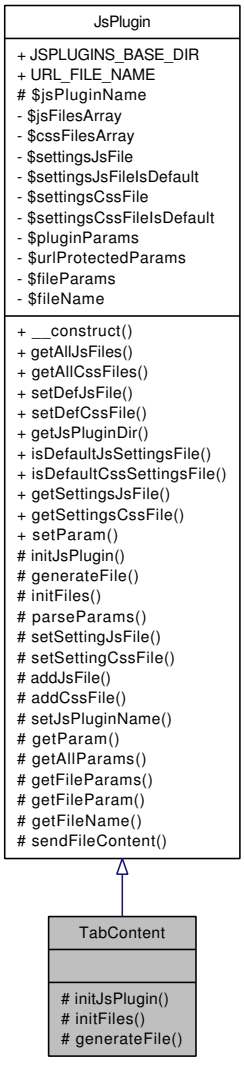
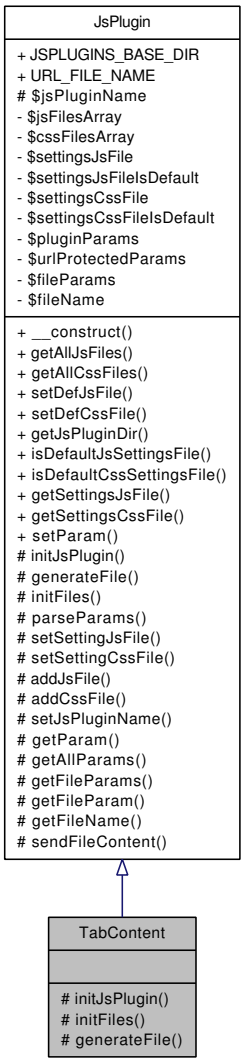


Diagram tříd pro TabContent:



Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

8.91.1 Detailní popis

Třída JsPluginu TabContent.

Třída slouží pro práci se záložkovým menu (tj. boxy se záložkovým přepínáním obsahu). Je úzce zpata z šablonou. //TODO dodělat tvorbu scriptu pro spuštění, tak aby se dal vložit přímo do šablony a ghenerován byl zde.

Verze:

$Id: tabcontenteasy.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu pro záložkový box

Definice je uvedena na řádku 14 v souboru tabcontent.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/tabcontent.class.php

8.92 Dokumentace třídy Template

Třída pro práci s šablonami modulu.

Veřejné metody

- __construct ()

  Konstruktor třídy.

- addTpl ($tplName, $engineTpl=false, $tplId=1)

  metoda přidává zadanou šablonu do výstupu

- addCss ($cssName, $withPath=false)

  metoda přidává zadany css styl do výstupu

- addJS ($javaScriptName, $withPath=false)

  metoda přidává zadaný javascript do výstupu

- setTplLabel ($name, $merge=false)

  Metoda nastaví název modulu, který bude vypsán na začátku.

- setTplSubLabel ($name, $merge=false, $separator= '-')

  Metoda nastaví podnázev modulu, který bude vypsán na začátku.

- setSubTitle ($name, $merge=false, $separator= '-')

  Metoda nastaví podtitulek modulu, který bude vypsán ve jménu okna.

- getSubTitle ()

  Metoda vrací přiřazený název titulku okna.

- setTplAlt ($name, $merge=false)

  Metoda nastaví popis (alt) modulu, který bude vypsán na začátku.

- setTplCatLink ($link=null)

  Metoda nastaví popis link na kategorii s modulem, který bude vypsán na začátku (použití asi jenom v panelech).

- addVar ($varName, $varValue, $isModuleVar=true)

  Metoda přiřazuje proměnné do šablony.

- **getTemplatesArray** ()
- getEngineVarsArray ()

  Metoda vrací pole s proměnými předanými do enginu.

- setModule (Module $module=null)

  Metoda nastavuje modul.

- addJsPlugin (JsPlugin $jsPlugin)

  Metoda přidává zadaný JsPlugin do šablony.

- addJsOnLoad ($jsFunction)

  Metoda přidá funkci do parametru OnLoad při načtení stránky.

Statické veřejné metody

- static getStylesheets ()

  statická metoda vrací pole se styly

- static getJavaScripts ()

  statická metoda vrací pole s javascripty

- static getJsOnLoad ()

  Metoda vrací pole s js funkcemi určenými k načtení po nahrátí stránky.

Veřejné atributy

- const **ITEMS_ARRAY_NAME** = 'ITEMS'
- const **TEMPLATES_ARRAY_NAME** = 'TEMPLATES'
- const **STYLESHHETS_ARRAY_NAME** = 'STYLESHEETS'
- const **JAVASCRIPTS_ARRAY_NAME** = 'JAVASCRIPTS'
- const **VARIABLES_ARRAY_NAME** = 'VARS'
- const **TEMPLATE_ID_NAME** = 'ID'
- const **TEMPLATE_FILE_NAME** = 'FILE'
- const **TEMPLATE_MODULE_LABEL** = 'LABEL'
- const **TEMPLATE_MODULE_SUBLABEL** = 'SUBLABEL'
- const **TEMPLATE_MODULE_ALT** = 'ALT'
- const **TEMPLATE_CATEGORY_LINK** = 'LINK'
- const **TEMPLATE_MODULE_STYLE_IDENT** = 'IDENT'

8.92.1 Detailní popis

Třída pro práci s šablonami modulu.

Třida obsahuje vrstvu mezi šablonovacím systémem a samotným pohledem (viewrem). Umožňuje všechny základní operace při volbě a plnění šablony a jejímu zobrazení.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: template.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu šablony

Definice je uvedena na řádku 13 v souboru template.class.php.

8.92.2 Dokumentace konstruktoru a destruktoru

8.92.2.1 Template::__construct ()

Konstruktor třídy.

Parametry:

Module – objekt modulu

Definice je uvedena na řádku 138 v souboru template.class.php.

```
138                          {
139 //    $this->module = $module;
140     }
```

8.92.3 Dokumentace k metodám

8.92.3.1 Template::addTpl ($ *tplName*, $ *engineTpl* = `false`, $ *tplId* = `1`)

metoda přidává zadanou šablonu do výstupu

Parametry:

string/array – název šablony

boolean – true pokud má být použita systémová šablona

Definice je uvedena na řádku 156 v souboru template.class.php.

```
156                                                                  {
157       $this->checkTemplatesArray();
158
159       //TODO kontrola souborů
160       //přidání šablony do pole s šablonami modulu
161       if($this->getModule() != null){
162          if($engineTpl == false){
163             if(!is_array($tplName)){
164                array_push($this->templates[$this->getModule()->getId()][self::TEMPLATES_ARRAY_NAME],
165                   array(self::TEMPLATE_FILE_NAME => $this->selectModuleTemplateFaceFile($tplName),
166                   self::TEMPLATE_ID_NAME =>  $tplId));
167             } else {
168                foreach ($tplName as $tpl) {
169                   array_push($this->templates[$this->getModule()->getId()][self::TEMPLATES_ARRAY_NAME]
170                      array(self::TEMPLATE_FILE_NAME => $this->selectModuleTemplateFaceFile($tpl),
171                      self::TEMPLATE_ID_NAME =>  $tplId));
172                }
173             }
174          } else {
175             if(!is_array($tplName)){
176                array_push($this->templates[$this->getModule()->getId()][self::TEMPLATES_ARRAY_NAME],
177 //                array(self::TEMPLATE_FILE_NAME => AppCore::getAppWebDir().DIRECTORY_SEPARATOR.AppCor
178                   array(self::TEMPLATE_FILE_NAME => $this->selectGlobalTemplateFaceFile($tplName),
179                   self::TEMPLATE_ID_NAME =>  $tplId));
180             } else {
181                foreach ($tplName as $tpl) {
182                   array_push($this->templates[$this->getModule()->getId()][self::TEMPLATES_ARRAY_NAME]
183 //                array(self::TEMPLATE_FILE_NAME => AppCore::getAppWebDir().DIRECTORY_SEPARATOR.AppCor
184                      array(self::TEMPLATE_FILE_NAME => $this->selectGlobalTemplateFaceFile($tpl),
185                      self::TEMPLATE_ID_NAME =>  $tplId));
```

```
186                 }
187             }
188         }
189     }
190 }
```

8.92.3.2 Template::addCss ($ *cssName*, $ *withPath* = `false`)

metoda přidává zadany css styl do výstupu

Parametry:

string – název scc stylu

boolean – true pokud je zadána i cesta se souborem

Definice je uvedena na řádku 254 v souboru template.class.php.

Používá se v addJsPlugin().

```
254                                                     {
255     //TODO kontrola souborů a duplicit
256     if(!$withPath){
257        if($this->getModule() != null){
258           $cssName = $this->selectModuleStylesheetFaceFile($cssName);
259 //        $cssName = $this->module->getDir()->getStylesheetsDir().$cssName;
260        }
261     }
262     array_push(self::$stylesheets, $cssName);
263 }
```

8.92.3.3 Template::addJS ($ *javaScriptName*, $ *withPath* = `false`)

metoda přidává zadaný javascript do výstupu

Parametry:

string – název javascriptu

boolean – true pokud je zadána i cesta se souborem

Definice je uvedena na řádku 292 v souboru template.class.php.

Používá se v addJsPlugin().

```
292                                                         {
293     //TODO kontrola souborů a duplicit
294     if(!$withPath){
295        if($this->getModule() != null){
296           $javaScriptName = $this->module->getDir()->getJavaScriptsDir().$javaScriptName;
297        }
298     }
299     array_push(self::$javascripts, $javaScriptName);
300 }
```

8.92.3.4 static Template::getStylesheets () [static]

statická metoda vrací pole se styly

Návratová hodnota:

array – pole se styly (obsahuje i cestu)

Definice je uvedena na řádku 306 v souboru template.class.php.

Používá se v AppCore::renderTemplate().

```
306                                         {
307         return self::$stylesheets;
308     }
```

8.92.3.5 static Template::getJavaScripts () [static]

statická metoda vrací pole s javascripty

Návratová hodnota:

array – pole s javascripty (obsahuje i cestu)

Definice je uvedena na řádku 314 v souboru template.class.php.

Používá se v AppCore::renderTemplate().

```
314                                         {
315         return self::$javascripts;
316     }
```

8.92.3.6 Template::setTplLabel ($ *name*, $ *merge* = false)

Metoda nastaví název modulu, který bude vypsán na začátku.

Parametry:

string – název

boolena – (option) jesli se má název přidat za stávající nebo přepsat

Definice je uvedena na řádku 323 v souboru template.class.php.

Používá se v setModule().

```
323                                                 {
324         if($this->getModule() != null){
325             if($merge){
326                 $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_LABEL].=$name;
327             } else {
328                 $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_LABEL]=$name;
329             }
330         }
331     }
```

8.92.3.7 Template::setTplSubLabel ($ *name*, $ *merge* = false, $ *separator* = '-')

Metoda nastaví podnázev modulu, který bude vypsán na začátku.

Parametry:

string – podnázev

boolena – (option) jesli se má název přidat za stávající nebo přepsat

string – (option) oddělovač mezi více nadpisy (default '-')

Definice je uvedena na řádku 339 v souboru template.class.php.

```
339                                                                                 {
340       if($this->getModule() != null){
341          if($merge){
342             if(isset($this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_SUBLABEL]) AN
343                $separator = ' '.$separator.' ';
344             } else {
345                $separator = null;
346                $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_SUBLABEL] = null;
347             }
348             $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_SUBLABEL].=$separator.
349          } else {
350             $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_SUBLABEL]=$name;
351          }
352
353       }
354    }
```

8.92.3.8 Template::setSubTitle ($ *name*, $ *merge* = false, $ *separator* = '-')

Metoda nastaví podtitulek modulu, který bude vypsán ve jménu okna.

Parametry:

string – podnázev

boolena – (option) jesli se má název přidat za stávající nebo přepsat

string – (option) oddělovač mezi více nadpisy (default '-')

Definice je uvedena na řádku 362 v souboru template.class.php.

```
362                                                                             {
363       if($merge){
364             if($this->pageTitle != null){
365                $separator = ' '.$separator.' ';
366             } else {
367                $separator = null;
368             }
369
370             $this->pageTitle.=$separator.$name;
371          } else {
372             $this->pageTitle=$name;
373          }
374    }
```

8.92.3.9 Template::getSubTitle ()

Metoda vrací přiřazený název titulku okna.

Návratová hodnota:

string – titulek okna

Definice je uvedena na řádku 380 v souboru template.class.php.

```
380                                 {
381         return $this->pageTitle;
382     }
```

8.92.3.10 Template::setTplAlt ($ *name*, $ *merge* = `false`)

Metoda nastaví popis (alt) modulu, který bude vypsán na začátku.

Parametry:

string – popis

boolena – (option) jesli se má popis přidat za stávající nebo přepsat

Definice je uvedena na řádku 389 v souboru template.class.php.

Používá se v setModule().

```
389                                                     {
390         if($this->getModule() != null){
391             if($merge){
392                 $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_ALT].=$name;
393             } else {
394                 $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_ALT]=$name;
395             }
396
397         }
398     }
```

8.92.3.11 Template::setTplCatLink ($ *link* = `null`)

Metoda nastaví popis link na kategorii s modulem, který bude vypsán na začátku (použití asi jenom v panelech).

Parametry:

string – link

Definice je uvedena na řádku 405 v souboru template.class.php.

```
405                                                 {
406         if($link == null){
407             $link = new Links();
408         }
409         if($this->getModule() != null){
410             $this->templates[$this->getModule()->getId()][self::TEMPLATE_CATEGORY_LINK]=$link;
411         }
412     }
```

8.92.3.12 Template::addVar ($ *varName*, $ *varValue*, $ *isModuleVar* = `true`)

Metoda přiřazuje proměnné do šablony.

Parametry:

string – název proměnné

string/array – hodnota proměnné

boolean – true pokud má být proměná zařazena do modulu (default: true)

Definice je uvedena na řádku 420 v souboru template.class.php.

Používá se v Eplugin::assignToTpl().

```
420                                                                     {
421         $this->checkVarsArray();
422
423         if($isModuleVar AND $this->getModule() != null){
424            $this->templates[$this->getModule()->getId()][self::VARIABLES_ARRAY_NAME][$varName] = $varVal
425         } else if($isModuleVar){
426            $this->templates[$varName] = $varValue;
427         } else {
428            $this->engineVars[$varName] = $varValue;
429         }
430
431     }
```

8.92.3.13 Template::getEngineVarsArray ()

Metoda vrací pole s proměnými předanými do enginu.

Návratová hodnota:

array – pole proměných

Definice je uvedena na řádku 459 v souboru template.class.php.

```
459                                     {
460         return $this->engineVars;
461     }
```

8.92.3.14 Template::setModule (Module $ *module* = `null`)

Metoda nastavuje modul.

Parametry:

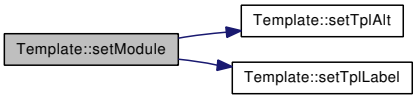
Module – objekt modulu

Definice je uvedena na řádku 467 v souboru template.class.php.

Odkazuje se na setTplAlt() a setTplLabel().

```
467                                                 {
468         $this->module = $module;
469         $this->checkTemplatesArray();
470         $this->setTplLabel($this->getModule()->getLabel());
471         $this->setTplAlt($this->getModule()->getAlt());
472 
473 //      Přiřazení identifikátoruModulu
474         $this->templates[$this->getModule()->getId()][self::TEMPLATE_MODULE_STYLE_IDENT] = $this->getMod
475 
476     }
```

Tato funkce volá...



8.92.3.15 Template::addJsPlugin (JsPlugin $ *jsPlugin*) [final]

Metoda přidává zadaný JsPlugin do šablony.

Parametry:

JsPlugin – objekt js pluginu

Definice je uvedena na řádku 483 v souboru template.class.php.

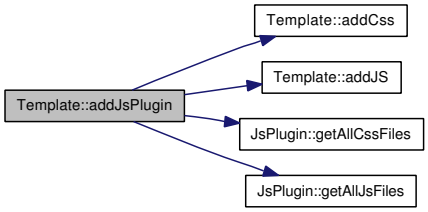
Odkazuje se na addCss(), addJS(), JsPlugin::getAllCssFiles() a JsPlugin::getAllJsFiles().

Používá se v Eplugin::assignToTpl().

```
484     {
485         if(get_parent_class($jsPlugin) == 'JsPlugin'){
486 //          Vložení ostatních js souborů pluginu
487             $jsOtherFiles = $jsPlugin->getAllJsFiles();
488             if(!empty($jsOtherFiles)){
489                 foreach ($jsOtherFiles as $jsFile) {
490                     $this->addJS($jsPlugin->getJsPluginDir().$jsFile, true);
491                 }
492             }
493 //          Vložení ostatních css souborů pluginu
494             $cssOtherFiles = $jsPlugin->getAllCssFiles();
495             if(!empty($cssOtherFiles)){
496                 foreach ($cssOtherFiles as $cssFile) {
497                     $this->addCss($jsPlugin->getJsPluginDir().$cssFile, true);
498                 }
499             }
500 
501 //          Přidání JS souboru s nastavením
502             if($jsPlugin->isDefaultJsSettingsFile() AND $jsPlugin->getSettingsJsFile() != null){
503                 $this->addJS($jsPlugin->getJsPluginDir().$jsPlugin->getSettingsJsFile(), true);
504             } else if($jsPlugin->getSettingsJsFile() != null) {
505                 $this->addJS($jsPlugin->getSettingsJsFile());
506             }
507 //          Přidání CSS souboru s nastavením
508             if($jsPlugin->isDefaultCssSettingsFile() AND $jsPlugin->getSettingsCssFile() != null){
509                 $this->addCss($jsPlugin->getJsPluginDir().$jsPlugin->getSettingsCssFile(), true);
510             } else if($jsPlugin->getSettingsCssFile() != null){
511                 $this->addCss($jsPlugin->getSettingsCssFile());
```

```
512          }
513       } else {
514          new CoreException(_("Nebyl vložen objekt JsPluginu"), 1);
515       }
516    }
```

Tato funkce volá...



8.92.3.16 Template::addJsOnLoad ($*jsFunction*) [final]

Metoda přidá funkci do parametru OnLoad při načtení stránky.

Parametry:

string – název funkce pro nahrání

Definice je uvedena na řádku 522 v souboru template.class.php.

```
522                                                 {
523       array_push(self::$onLoadJsFunctions, $jsFunction);
524    }
```

8.92.3.17 static Template::getJsOnLoad () [static]

Metoda vrací pole s js funkcemi určenými k načtení po nahrátí stránky.

Návratová hodnota:

array – pole s funkcemi

Definice je uvedena na řádku 530 v souboru template.class.php.

Používá se v AppCore::renderTemplate().

```
530                                        {
531       return self::$onLoadJsFunctions;
532    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/template.class.php

8.93 Dokumentace třídy TextAction

Třída pro obsluhu akcí v modulu.

Diagram dědičnosti pro třídu TextAction

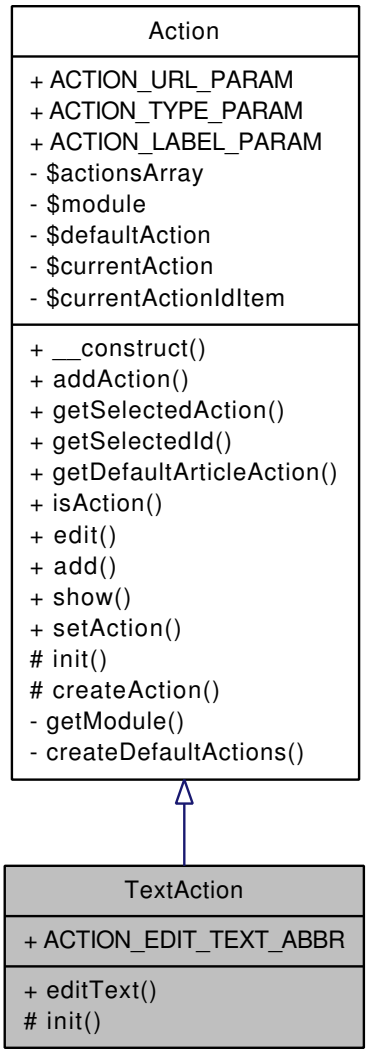
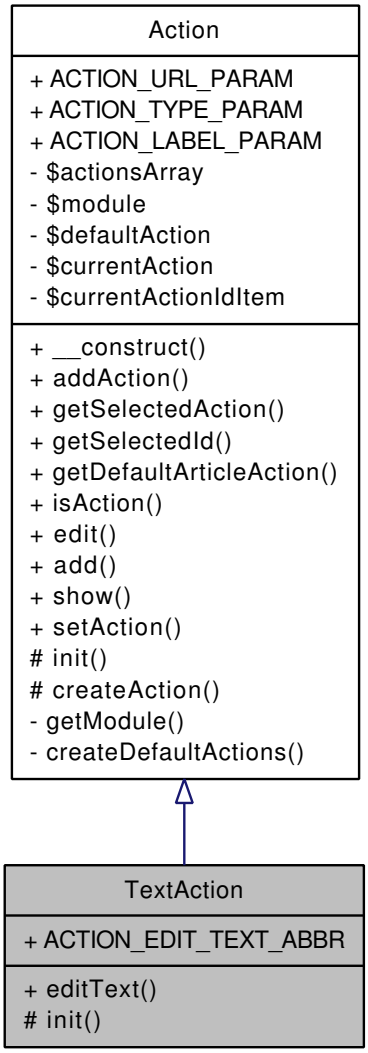


Diagram tříd pro TextAction:



Veřejné metody

- **editText** ()

Veřejné atributy

- const **ACTION_EDIT_TEXT_ABBR** = 'at'

Chráněné metody

- init ()

  Metoda pro inicializaci akcí.

8.93.1 Detailní popis

Třída pro obsluhu akcí v modulu.

Definice je uvedena na řádku 6 v souboru action.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/text/action.class.php

8.94 Dokumentace třídy TextController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu TextController

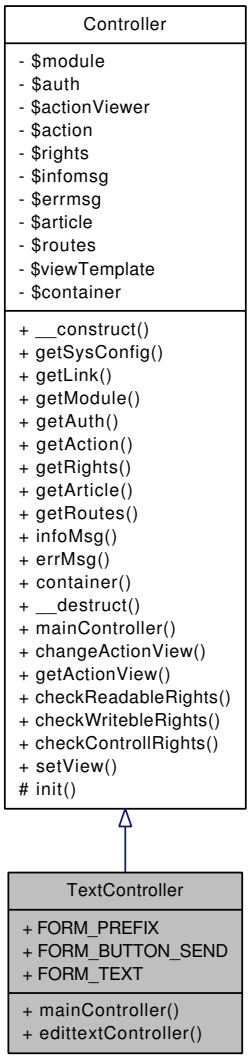
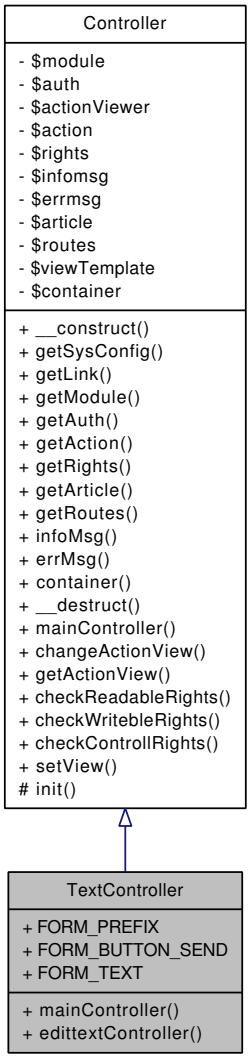


Diagram tříd pro TextController:



Veřejné metody

- mainController ()

  Kontroler pro zobrazení textu.

- edittextController ()

  Kontroler pro editaci textu.

Veřejné atributy

- const **FORM_PREFIX** = 'text_'
- const **FORM_BUTTON_SEND** = 'send'
- const **FORM_TEXT** = 'text'

8.94.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/text/controler.class.php

8.95 Dokumentace třídy TextCtrlHelper

Ttřída textového Controll Helperu pro zjednodušení práce s texty.

Diagram dědičnosti pro třídu TextCtrlHelper

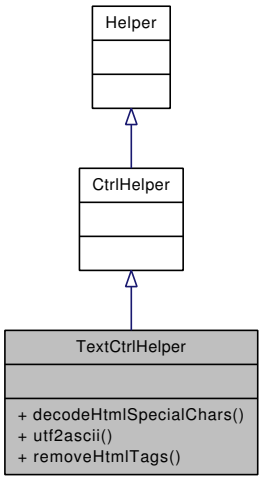
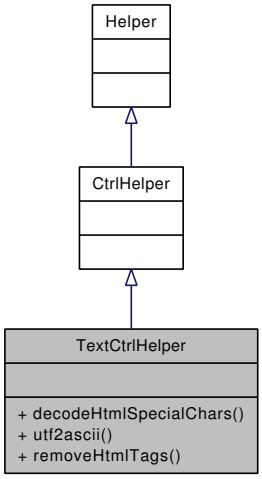


Diagram tříd pro TextCtrlHelper:



Veřejné metody

- decodeHtmlSpecialChars ($text)

  Metoda dekóduje znaky na entity html.

- utf2ascii ($text)

  Metoda odstrani nekorektni znaky z řetězce.

- removeHtmlTags ($string)

  Metoda odstraní všechny html tagy ze zadaného stringu.

8.95.1 Detailní popis

Ttřída textového Controll Helperu pro zjednodušení práce s texty.

Třída poskytuje některé funkce pro práci s textem.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: localectrlhelper.class.php 3.0.55 27.9.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro práci s textovými prvky v kontroleru - helper

Definice je uvedena na řádku 12 v souboru textctrlhelper.class.php.

8.95.2 Dokumentace k metodám

8.95.2.1 TextCtrlHelper::decodeHtmlSpecialChars ($ *text*)

Metoda dekóduje znaky na entity html.

Parametry:

string – text k dekódování

Návratová hodnota:

string – dekódovaný text

Zastaralé

Definice je uvedena na řádku 20 v souboru textctrlhelper.class.php.

```
20                                                  {
21         //TODO pořešid pokud je zaplé tiny mce
22         $text = htmlspecialchars_decode($text);
23
24         return $text;
25     }
```

8.95.2.2 TextCtrlHelper::utf2ascii ($ *text*)

Metoda odstrani nekorektni znaky z řetězce.

Parametry:

string – řetězec z kterého se odstraní znaky

Návratová hodnota:

string – výsledný řetězec

Definice je uvedena na řádku 33 v souboru textctrlhelper.class.php.

```
34    {
35       $return = Str_Replace(
36       Array("á","č","d’","é","ě","í","l’","ň","ó","ř","š","t’","ú","ů","ý","ž","Á","Č","Ď","É","Ě","Í",
37       Array("a","c","d","e","e","i","l","n","o","r","s","t","u","u","y","z","A","C","D","E","E","I","L"
38       $text);
39
40       $return = Str_Replace(Array(" ", "_"), "-", $return); //nahradí mezery a podtržítka pomlčkami
41       $return = Str_Replace(array("----","---","--"), "-", $return); //odstraní nekolik pomlcek za sebo
42       $return = Str_Replace(Array("(",")",".","!",",","\"","’"), "", $return); //odstraní ().!,"’
43       $return = StrToLower($return); //velká písmena nahradí malými.
44         return $return;
45    }
```

8.95.2.3 TextCtrlHelper::removeHtmlTags ($ *string*)

Metoda odstraní všechny html tagy ze zadaného stringu.

Parametry:

string $string – řětězec

Návratová hodnota:

string – řetězec bez tagů

Definice je uvedena na řádku 53 v souboru textctrlhelper.class.php.

```
53                                          {
54       $string=ereg_replace("<[^>]+>", "", $string);
55       return $string;
56    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/helpers/textctrlhelper.class.php

8.96 Dokumentace třídy TextRoutes

Třída obsluhující cesty modulu.

Diagram dědičnosti pro třídu TextRoutes

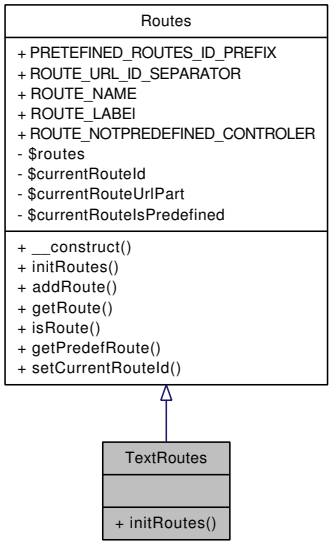
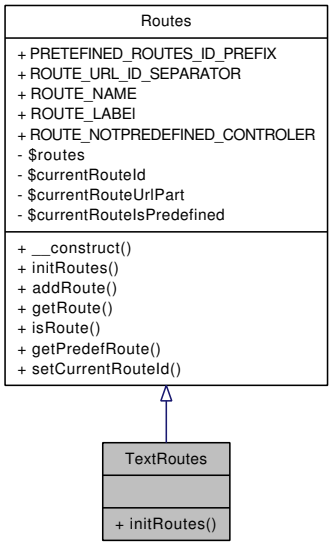


Diagram tříd pro TextRoutes:



Veřejné metody

- initRoutes ()

  Metoda, která nastavuje cesty.

8.96.1 Detailní popis

Třída obsluhující cesty modulu.

Definice je uvedena na řádku 6 v souboru routes.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/text/routes.class.php

8.97 Dokumentace třídy TextView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu TextView

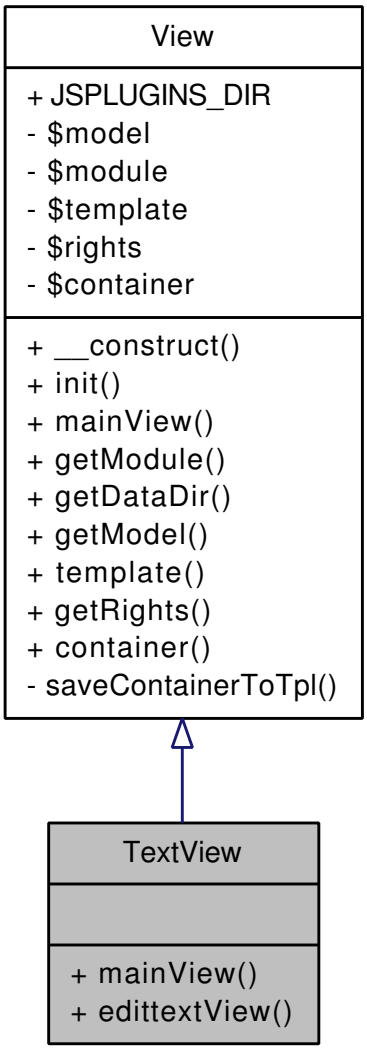
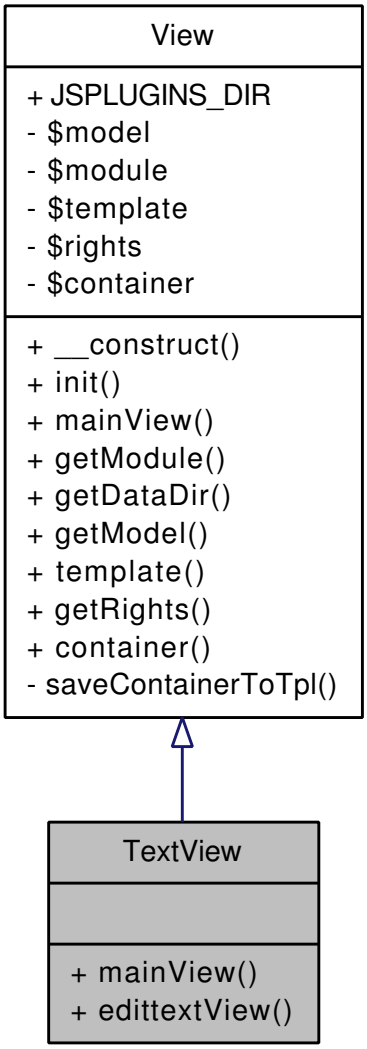


Diagram tříd pro TextView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- **edittextView** ()

8.97.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/text/view.class.php

8.98 Dokumentace třídy TimeValidator

Description of TimeValidator Třída slouží pro validaci časů a datumů.

Diagram dědičnosti pro třídu TimeValidator

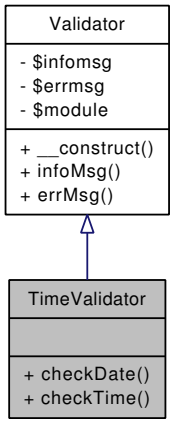
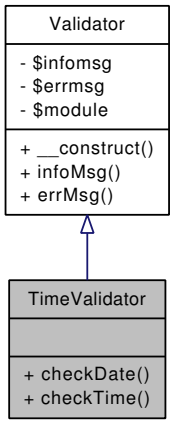


Diagram tříd pro TimeValidator:



Veřejné metody

- checkDate ($date, $timestam=false)

  Metoda testuje zadané datum, pokud je v pořádku je vráceno v ISO podobe YYYY-MM-DD.

- checkTime ($time, $timestamp=false)

  Metoda zkontroluje zadaný čas (hh:mm).

8.98.1 Detailní popis

Description of TimeValidator Třída slouží pro validaci časů a datumů.

Verze:

Id

timevalidator.class.php 419 2008-11-28 23:21:19Z jakub

VVE 3.3.0 $Revision:$

Autor:

$Author$ $Date$ $LastChangedBy$ $LastChangedDate$ Třída pro validaci časů a datumů prvků

Definice je uvedena na řádku 11 v souboru timevalidator.class.php.

8.98.2 Dokumentace k metodám

8.98.2.1 TimeValidator::checkDate ($ *date*, $ *timestam* = `false`)

Metoda testuje zadané datum, pokud je v pořádku je vráceno v ISO podobe YYYY-MM-DD.

Parametry:

string – kontrolované datum

boolean(option) – true pokud má být vrácen timestamp

Návratová hodnota:

string – vrací datum v ISO podobě nebo false

Definice je uvedena na řádku 18 v souboru timevalidator.class.php.

Odkazuje se na Locale::getLang().

```
18                                                      {
19       if (ereg("([0-9]{1,2}).([0-9]{1,2}).([0-9]{4})",$date, $regs) AND checkdate ($regs[2], $regs[1],
20          $date = $regs[2].self::ISO_DATE_SEPARATOR.$regs[1].self::ISO_DATE_SEPARATOR.$regs[3];
21          $time = mktime(null,null,null,$regs[2],$regs[1],$regs[3]);
22       } else if (ereg ("([0-9]{1,2})/([0-9]{1,2})/([0-9]{4})",$date, $regs) AND checkdate ($regs[1], $r
23          $date = $regs[1].self::ISO_DATE_SEPARATOR.$regs[2].self::ISO_DATE_SEPARATOR.$regs[3];
24          $time = mktime(null,null,null,$regs[1],$regs[2],$regs[3]);
25       } else {
26          $date = $time = false;
27       }
28
29       if($timestam){
30          return $time;
31       } else {
32          return $date;
33       }
34    }
```

Tato funkce volá...

8.98.2.2 TimeValidator::checkTime ($ *time*, $ *timestamp* = `false`)

Metoda zkontroluje zadaný čas (hh:mm).

Parametry:

string – kontrolovaný čas

boolean(option) – true pokud má být vrácen timestamp

Návratová hodnota:

string – vrací čas v podobě HH:MM nebo false

Definice je uvedena na řádku 43 v souboru timevalidator.class.php.

```
43                                                      {
44        $nums = explode(self::TIME_SEAPRATOR, $time, 2);
45
46        $allOk = true;
47 //     Jestli se jedná o hodinu
48        if(!isset($nums[0]) OR $nums[0] < 0 OR $nums[0] > 23){
49            $allOk = false;
50        }
51
52 //     Jestli se jedná o minuty
53        if(!isset($nums[1]) OR $nums[1] < 0 OR $nums[1] > 59){
54            $allOk = false;
55        }
56
57        if($allOk){
58           if(!$timestamp){
59              return $nums[0].self::TIME_SEAPRATOR.$nums[1];
60           } else {
61              return mktime($nums[0], $nums[1]);
62           }
63        } else {
64           return false;
65        }
66     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Validators/timevalidator.class.php

8.99 Dokumentace třídy TinyMce

Třída JsPluginu TinyMce.

Diagram dědičnosti pro třídu TinyMce

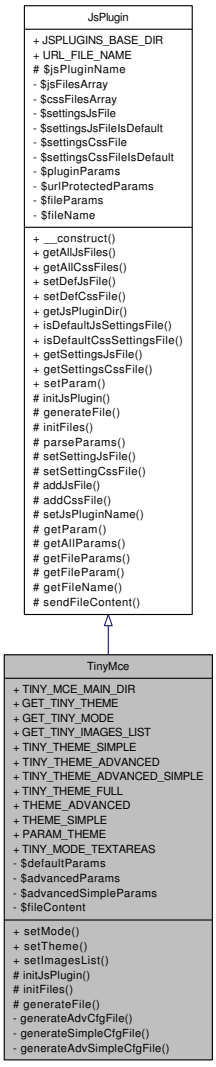
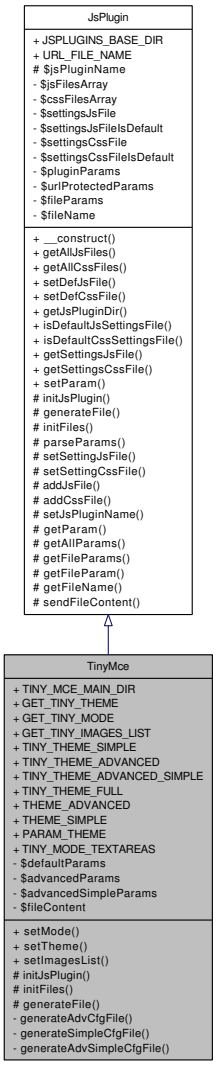


Diagram tříd pro TinyMce:



Veřejné metody

- setMode ($mode)

  Metoda nastaví typ zobrazení pluginu.

- setTheme ($theme)

  Metoda nastaví typ zobrazení pluginu.

- setImagesList ($fileLink)

  Metoda nastavuje jestli se načíst obrázky a odkud.

Veřejné atributy

- const **TINY_MCE_MAIN_DIR** = 'tinymce'

- const **GET_TINY_THEME** = 'theme'
- const **GET_TINY_MODE** = 'mode'
- const **GET_TINY_IMAGES_LIST** = 'img'
- const **TINY_THEME_SIMPLE** = 'simple'
- const **TINY_THEME_ADVANCED** = 'advanced'
- const **TINY_THEME_ADVANCED_SIMPLE** = 'advsimple'
- const **TINY_THEME_FULL** = 'full'
- const THEME_ADVANCED = 'advanced'

  Parametry pro typ theme.

- const **THEME_SIMPLE** = 'simple'
- const PARAM_THEME = 'theme'

  některé parametry v konfigu

- const **TINY_MODE_TEXTAREAS** = 'textareas'

Chráněné metody

- initJsPlugin ()

  Třída, která se provede při inicializaci pluginu.

- initFiles ()

  Metoda inisializuje všechny soubory, se kterými JsPlugin pracuje.

- generateFile ()

  Metda vytvoří výchozí konfigurační soubor.

8.99.1 Detailní popis

Třída JsPluginu TinyMce.

Třída slouží pro vytvoření JsPluginu TinyMce, což je wysiwing textový edtor, který se navazuje na textarea v šabloně. Třída umožňuje jednoduché nastavení parametrů editor (volba vzhledu, jazyka, atd).

Copyright (c) 2008 Jakub Matas

Verze:

$Id: tinymce.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída JsPluginu TinyMce

Definice je uvedena na řádku 14 v souboru tinymce.class.php.

8.99.2 Dokumentace k metodám

8.99.2.1 TinyMce::setImagesList ($*fileLink*)

Metoda nastavuje jestli se načíst obrázky a odkud.

Parametry:

integer – id itemu

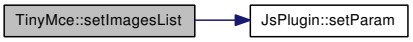
integer – id článku

Definice je uvedena na řádku 174 v souboru tinymce.class.php.

Odkazuje se na JsPlugin::setParam().

```
174                                             {
175       $this->setParam(self::GET_TINY_IMAGES_LIST, $fileLink);
176     }
```

Tato funkce volá...



Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/JsPlugins/tinymce.class.php

8.100 Dokumentace třídy UploadFiles

Třída pro obsluhu přenesených souborů.

Veřejné metody

- __construct (Messages $errors=null, $canEmpty=false)
  Konstruktor třídy.
- upload ($postName, $dest=null)
  Metoda přenese zadaný soubor.
- isUploaded ()
  Metoda vrací jestli byl soubor nahrán.
- isUploadError ()
  Metoda vrací jestli nebyl nahrán žádný soubor.
- copy ($dstDir)
  Metoda překopíruje zadaný soubor do zadaného adresáře //TODO not implemented.
- getTmpName ()
  Metoda vrací název nahraného souboru.
- getOriginalName ()
  Metoda vrací originální název souboru.
- isZipFile ()
  Metoda zjišťuje, zdali je uploadovaný soubor zip soubor.
- getMimeType ()
  Metoda vrací mime typ souboru //TODO nutná portace na PECL rozšíření o informací o souboru.

Veřejné atributy

- const **POST_FILES_ERROR** = 'error'
- const **POST_FILES_ORIGINAL_NAME** = 'name'
- const **POST_FILES_SIZE** = 'size'
- const **POST_FILES_TYPE** = 'type'
- const **POST_FILES_TMP_NAME** = 'tmp_name'

8.100.1 Detailní popis

Třída pro obsluhu přenesených souborů.

Třída poskytuje základní metody pro práci s uploadovanými soubory, jejich kontrolu, zjišťování mime typu a ukládání do filesystému.

Verze:

$Id: uploadfiles.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro obsluhu uploadovaných souborů

Definice je uvedena na řádku 13 v souboru uploadfiles.class.php.

8.100.2 Dokumentace konstruktoru a destruktoru

8.100.2.1 UploadFiles::__construct (Messages $ *errors* = null, $ *canEmpty* = false)

Konstruktor třídy.

Parametry:

Messages – objekt pro přístup ke správám (option. pokud je nul je použito CoreException)

boolean – jestli může být přenesen prázdný soubor

Definice je uvedena na řádku 92 v souboru uploadfiles.class.php.

```
92                                                                  {
93        $this->canEmpty = $canEmpty;
94        $this->errors = $errors;
95    }
```

8.100.3 Dokumentace k metodám

8.100.3.1 UploadFiles::upload ($ *postName*, $ *dest* = null)

Metoda přenese zadaný soubor.

Parametry:

string – název $_POST se souborem

string – adresář do kterého se má soubor zapsat

Návratová hodnota:

boolean – true pokud byl soubor v pořádku nahrán

Definice je uvedena na řádku 105 v souboru uploadfiles.class.php.

```
105                                                 {
106       $error = false;
107       if (is_uploaded_file($_FILES[$postName][self::POST_FILES_TMP_NAME])) {
108          $this->fileUploaded = true;
109          $this->noFileUpload = false;
110
111          $this->file = $_FILES[$postName];
112
113          $this->fileOriginalName = $_FILES[$postName][self::POST_FILES_ORIGINAL_NAME];
114
```

```
115       } else {
116          switch($_FILES[$postName][self::POST_FILES_ERROR]){
117             case 0: //no error; possible file attack!
118                $this->uploadError = $error = true;
119                $this->addError('Problém s nahráním souboru');
120                break;
121             case 1: //uploaded file exceeds the upload_max_filesize directive in php.ini
122                $this->uploadError = $error = true;
123                $this->addError('Soubor je příliš velký');
124                break;
125             case 2: //uploaded file exceeds the MAX_FILE_SIZE directive that was specified in the html
126                $this->uploadError = $error = true;
127                $this->addError('Soubor je příliš velký');
128                break;
129             case 3: //uploaded file was only partially uploaded
130                $this->addError('Soubor byl nahrán jen částečně');
131                $this->uploadError = $error = true;
132                break;
133             case 4: //no file was uploaded
134                if(!$this->canEmpty){
135                   $this->addError('Soubor nebyl vybrán');
136                   $this->uploadError = $error = true;
137                }
138                break;
139             default: //a default error, just in case!  :)
140                $this->uploadError = $error = true;
141                $this->addError('Problém s nahráním souboru');
142                break;
143          }
144       }
145       return !$error;
146    }
```

8.100.3.2 UploadFiles::isUploaded ()

Metoda vrací jestli byl soubor nahrán.

Návratová hodnota:

boolean – true pokud byl soubor nahrán

Definice je uvedena na řádku 152 v souboru uploadfiles.class.php.

```
152                                {
153       return $this->fileUploaded;
154    }
```

8.100.3.3 UploadFiles::isUploadError ()

Metoda vrací jestli nebyl nahrán žádný soubor.

Návratová hodnota:

boolean – true pokud byl soubor nahrán

Definice je uvedena na řádku 160 v souboru uploadfiles.class.php.

```
160                                  {
161       return $this->uploadError;
162    }
```

8.100.3.4 UploadFiles::getTmpName ()

Metoda vrací název nahraného souboru.

Návratová hodnota:

string – nahraný soubor

Definice je uvedena na řádku 192 v souboru uploadfiles.class.php.

```
192                                 {
193         return $this->file[self::POST_FILES_TMP_NAME];
194     }
```

8.100.3.5 UploadFiles::getOriginalName ()

Metoda vrací originální název souboru.

Návratová hodnota:

string – originální název souboru

Definice je uvedena na řádku 201 v souboru uploadfiles.class.php.

```
201                                     {
202         return $this->file[self::POST_FILES_ORIGINAL_NAME];
203     }
```

8.100.3.6 UploadFiles::isZipFile ()

Metoda zjišťuje, zdali je uploadovaný soubor zip soubor.

Parametry:

string – soubor, který se zjišťuje

Definice je uvedena na řádku 210 v souboru uploadfiles.class.php.

Odkazuje se na getMimeType().

```
210                                 {
211         if(in_array($this->getMimeType(), $this->zipExtensionsArray)){
212            return true;
213         } else {
214            return false;
215         }
216     }
```

Tato funkce volá...

8.100.3.7 UploadFiles::getMimeType ()

Metoda vrací mime typ souboru //TODO nutná portace na PECL rozšíření o informací o souboru.

Návratová hodnota:

string – mime typ souboru

Definice je uvedena na řádku 224 v souboru uploadfiles.class.php.

Používá se v isZipFile().

```
224                                 {
225         return $this->file[self::POST_FILES_TYPE];
226     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/uploadfiles.class.php

8.101 Dokumentace třídy UrlParam

Description of UrlParams Třída slouží pro práci s parametry v url, e nastavována přímo z requestu a jsou v ní uloženy všechny předané parametry i s hodnotami.

Veřejné metody

- __construct ($name, $pattern=false)
  Konstruktor třídy vytvoří objekt parametru pro URL.
- setValue ($value)
  Matoda nastavuje hodnotu parametru.
- getValue ()
  Metoda vrací hodnotu parametru v url.
- isValue ()
  Metoda vrací true pokud je nastavena hodnota parametru.
- getPattern ()
  Metoda vrací regulérní výraz pro parsování parametru.
- isNormalParam ()
  Metoda vrací jestli se jedná o parametr v url nebo normalový.
- __toString ()
  magická metoda volaná při převodu na řetězec

Statické veřejné metody

- static setParam ($param)
  Metoda nastaví daný parametr do pole parametrů.
- static setNormalParams ($params)
  Metoda parsuje a nastavuje normálové parametry (tj.
- static getNormalParams ()
  Metoda vrací pole s normálovými paarametry (jsou umístěny za otazníkem).
- static getParams ()
  Metoda vrací pole s parametry, předávanými pomocí url (před otazníkem) a oddělenými lomítky.

8.101.1 Detailní popis

Description of UrlParams Třída slouží pro práci s parametry v url, e nastavována přímo z requestu a jsou v ní uloženy všechny předané parametry i s hodnotami.

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro obsluhu parametrů v URL

Definice je uvedena na řádku 13 v souboru urlparam.class.php.

8.101.2 Dokumentace konstruktoru a destruktoru

8.101.2.1 UrlParam::__construct ($ *name*, $ *pattern* = `false`)

Konstruktor třídy vytvoří objekt parametru pro URL.

Parametry:

string – název parametru

mixed – (option) regulérní výraz pro parsování parametru. pokud není zadán, jedná se o normálový parametr. je zadávan jen výraz co je hodnota, k výrazu se přřidává název proměnné např: '([0-9]+)'

Definice je uvedena na řádku 102 v souboru urlparam.class.php.

```
102                                                         {
103         if($pattern){
104             $this->pattern = '^'.$name.$pattern.'$';
105         }
106 
107         $this->name = $name;
108 //        Kontrola existence parametru
109         $this->checkParam();
110     }
```

8.101.3 Dokumentace k metodám

8.101.3.1 static UrlParam::setParam ($ *param*) [static]

Metoda nastaví daný parametr do pole parametrů.

Parametry:

string $param – řetězec s parametrem

Definice je uvedena na řádku 59 v souboru urlparam.class.php.

Používá se v Links::chackOtherUrlParams().

```
59                                          {
60         array_push(self::$params, $param);
61     }
```

8.101.3.2 static UrlParam::setNormalParams ($ *params*) [static]

Metoda parsuje a nastavuje normálové parametry (tj.

ty co jsou obsaženy za otazníkem v url)

Parametry:

string $params – řetězec s parametry (se všemi)

Definice je uvedena na řádku 68 v souboru urlparam.class.php.

Používá se v Links::chackOtherUrlParams().

```
68                                                  {
69         self::$normalParams = $params;
70     }
```

8.101.3.3 static UrlParam::getNormalParams () [static]

Metoda vrací pole s normálovými paarametry (jsou umístěny za otazníkem).

Návratová hodnota:

array – pole s parametry

Definice je uvedena na řádku 76 v souboru urlparam.class.php.

```
76                                          {
77         return self::$normalParams;
78     }
```

8.101.3.4 static UrlParam::getParams () [static]

Metoda vrací pole s parametry, předávanými pomocí url (před otazníkem) a oddělenými lomítky.

Návratová hodnota:

array – pole s parametry

Definice je uvedena na řádku 86 v souboru urlparam.class.php.

```
86                                      {
87          return self::$params;
88      }
```

8.101.3.5 UrlParam::setValue ($ *value*)

Matoda nastavuje hodnotu parametru.

Parametry:

mixed $value – hodnota

Definice je uvedena na řádku 145 v souboru urlparam.class.php.

```
145                                     {
146         $this->value = $value;
147         return $this;
148     }
```

8.101.3.6 UrlParam::getValue ()

Metoda vrací hodnotu parametru v url.

Návratová hodnota:

mixed – hodnota parametru

Definice je uvedena na řádku 154 v souboru urlparam.class.php.

```
154                                 {
155         return $this->value;
156     }
```

8.101.3.7 UrlParam::isValue ()

Metoda vrací true pokud je nastavena hodnota parametru.

Návratová hodnota:

boolean

Definice je uvedena na řádku 162 v souboru urlparam.class.php.

```
162                                 {
163         return $this->isSet;
164     }
```

8.101.3.8 UrlParam::getPattern ()

Metoda vrací regulérní výraz pro parsování parametru.

Návratová hodnota:

string – regulérní výraz

Definice je uvedena na řádku 170 v souboru urlparam.class.php.

Odkazuje se na isNormalParam().

```
170                                 {
171         if(!$this->isNormalParam()){
172             return $this->pattern;
173         } else {
174             return false;
175         }
176     }
```

Tato funkce volá...

8.101.3.9 UrlParam::isNormalParam ()

Metoda vrací jestli se jedná o parametr v url nebo normalový.

Návratová hodnota:

boolean – true pokud se jedná o normálový parametr

Definice je uvedena na řádku 182 v souboru urlparam.class.php.

Používá se v __toString() a getPattern().

```
182                                     {
183         if($this->pattern == null){
184             return true;
185         } else {
186             return false;
187         }
188     }
```

8.101.3.10 UrlParam::__toString ()

magická metoda volaná při převodu na řetězec

Návratová hodnota:

string – url parametr

Definice je uvedena na řádku 194 v souboru urlparam.class.php.

Odkazuje se na isNormalParam().

```
194                                 {
195         if(!$this->isNormalParam()){
196             return $this->name.$this->value;
197         } else {
198             return $this->name.Links::URL_SEP_PARAM_VALUE.$this->value;
199         }
200     }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/urlparam.class.php

8.102 Dokumentace třídy UrlRequest

Description of UrlRequest Třída slouží pro parsování a obsluhu požadavků v URL adrese.

Veřejné metody

- __construct (Action $action, Routes $routes)
  Konstruktor.
- choseController ()
  Metoda zevolí typ kontroleru modulu.

Statické veřejné metody

- static factory ()
  Metoda inicializuje požadavky v URL.
- static getBaseWebDir ()
  Metoda vrací základní URL cestu k aplikaci.
- static getMediaType ()
  Metoda vrací typ média pro načtenou stránku.
- static getCurrentMediaUrlPart ()
  Metoda vrací URl část řetězce s nastaveným typem média.
- static isEplugin ()
  Metoda zjišťuje jestli byl nastaven index na eplugin.
- static getSelEpluginName ()
  Metoda vrací název zvoleného epluginu.
- static isJsplugin ()
  Metoda zjišťuje jestli byl nastaven index na jsplugin.
- static getSelJspluginName ()
  Metoda vrací název zvoleného jspluginu.

Veřejné atributy

- const **PARAM_MEDIA_TYPE_PREFIX** = 'media'
- const **MEDIA_TYPE_WWW** = 'www'
- const **MEDIA_TYPE_SITEMAP** = 'sitemap'
- const **MEDIA_TYPE_PRINT** = 'print'
- const **TRANSFER_PROTOCOL** = 'http://'
- const **GET_EPLUGIN_NAME** = 'eplugin'
- const **GET_JSPLUGIN_NAME** = 'jsplugin'
- const **URL_SEPARATOR** = '/'

8.102.1 Detailní popis

Description of UrlRequest Třída slouží pro parsování a obsluhu požadavků v URL adrese.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

$Date:$

LastChangedBy

LastChangedDate

Třída pro obsluhu a UrlReqestu

Definice je uvedena na řádku 12 v souboru urlrequest.class.php.

8.102.2 Dokumentace k metodám

8.102.2.1 static UrlRequest::getBaseWebDir () [static]

Metoda vrací základní URL cestu k aplikaci.

Návratová hodnota:

string

Definice je uvedena na řádku 234 v souboru urlrequest.class.php.

Používá se v Links::__toString(), AppCore::assignMainVarsToTemplate(), Links::getLinkToDownloadFile() a Links::getMainWebDir().

```
234                                         {
235         return self::$baseWebUrl;
236     }
```

8.102.2.2 static UrlRequest::getMediaType () [static]

Metoda vrací typ média pro načtenou stránku.

Návratová hodnota:

string

Definice je uvedena na řádku 242 v souboru urlrequest.class.php.

Používá se v AppCore::renderTemplate() a AppCore::runApp().

```
242                                         {
243         return self::$media;
244     }
```

8.102.2.3 static UrlRequest::getCurrentMediaUrlPart () [static]

Metoda vrací URl část řetězce s nastaveným typem média.

Návratová hodnota:

string – url část

Definice je uvedena na řádku 250 v souboru urlrequest.class.php.

```
250                                                 {
251         return self::$currentMediaUrlPart;
252     }
```

8.102.2.4 static UrlRequest::isEplugin () [static]

Metoda zjišťuje jestli byl nastaven index na eplugin.

Návratová hodnota:

boolean – true pokud se má zpracovávat eplugin

Definice je uvedena na řádku 291 v souboru urlrequest.class.php.

Používá se v AppCore::runApp().

```
291                                     {
292         if(isset($_GET[self::GET_EPLUGIN_NAME])){
293             return true;
294         } else {
295             return false;
296         }
297     }
```

8.102.2.5 static UrlRequest::getSelEpluginName () [static]

Metoda vrací název zvoleného epluginu.

Návratová hodnota:

string – název epluginu

Definice je uvedena na řádku 304 v souboru urlrequest.class.php.

Používá se v AppCore::runApp().

```
304                                                 {
305         return rawurldecode($_GET[self::GET_EPLUGIN_NAME]);
306     }
```

8.102.2.6 static UrlRequest::isJsplugin () [static]

Metoda zjišťuje jestli byl nastaven index na jsplugin.

Návratová hodnota:

boolean – true pokud se má zpracovávat jsplugin

Definice je uvedena na řádku 313 v souboru urlrequest.class.php.

Používá se v JsPlugin::__construct() a AppCore::runApp().

```
313                                         {
314         if(isset($_GET[self::GET_JSPLUGIN_NAME])){
315             return true;
316         } else {
317             return false;
318         }
319     }
```

8.102.2.7 static UrlRequest::getSelJspluginName () [static]

Metoda vrací název zvoleného jspluginu.

Návratová hodnota:

string – název jspluginu

Definice je uvedena na řádku 326 v souboru urlrequest.class.php.

Používá se v AppCore::runApp().

```
326                                                 {
327         return rawurldecode($_GET[self::GET_JSPLUGIN_NAME]);
328     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/urlrequest.class.php

8.103 Dokumentace třídy UrlValidator

Description of UrlValidator Třída slouží pro validaci url adres a emailů.

Diagram dědičnosti pro třídu UrlValidator

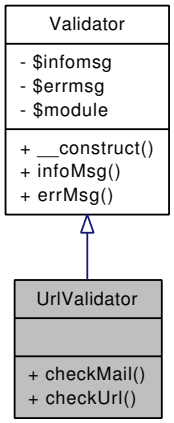
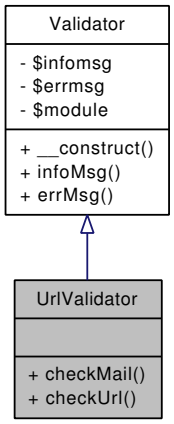


Diagram tříd pro UrlValidator:



Veřejné metody

- checkMail ($email)

  Metoda kontroluje emailovou adresu.

- checkUrl ($url)

  Metoda kontroluje správnost url adresy.

8.103.1 Detailní popis

Description of UrlValidator Třída slouží pro validaci url adres a emailů.

Verze:

Id

urlvalidator.class.php 419 2008-11-28 23:21:19Z jakub

VVE 3.3.0 $Revision:$

Autor:

$Author$ $Date$ $LastChangedBy$ $LastChangedDate$ Třída pro validaci formulářových prvků

Definice je uvedena na řádku 11 v souboru urlvalidator.class.php.

8.103.2 Dokumentace k metodám

8.103.2.1 UrlValidator::checkMail ($ *email*)

Metoda kontroluje emailovou adresu.

Parametry:

string $mail – adresa, která se má kontrolovat

Návratová hodnota:

boolean – true pokud se jedná o email

Definice je uvedena na řádku 18 v souboru urlvalidator.class.php.

```
18                                      {
19
20        if (eregi("^[a-z0-9_\.]+@[a-z0-9_\.]+[a-z]{2,3}$", $email)) {
21           return true;
22        }
23        else {
24           return false;
25        }
26     }
```

8.103.2.2 UrlValidator::checkUrl ($ *url*)

Metoda kontroluje správnost url adresy.

Parametry:

string $url – url adresa

Návratová hodnota:

boolean – vrací true pokud se jedná o url adresu

Plánované úpravy

– dodělat, není implementována

Definice je uvedena na řádku 34 v souboru urlvalidator.class.php.

```
34                                      {
35          return true;
36      }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Validators/urlvalidator.class.php

8.104 Dokumentace třídy UserFilesEplugin

Třída EPluginu pro přidávání souborů ke článku(stránce).

Diagram dědičnosti pro třídu UserFilesEplugin

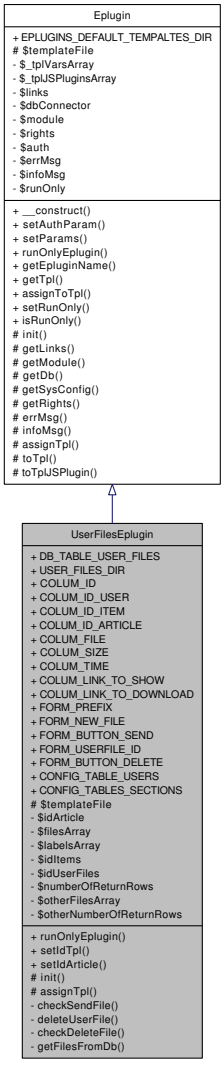
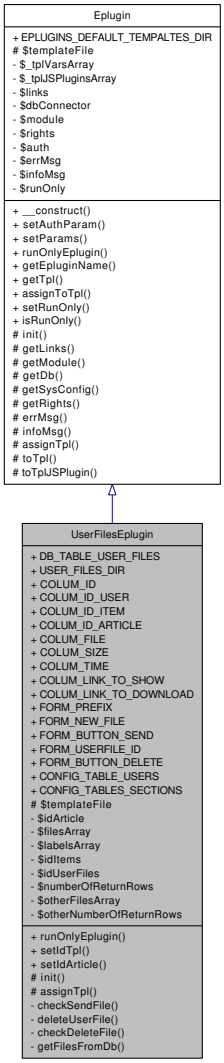


Diagram tříd pro UserFilesEplugin:



Veřejné metody

- runOnlyEplugin ()

  Metoda je spuštěna při načítání souborů.

- setIdTpl ($id)

  Metoda nastaví id šablony pro výpis.

- setIdArticle ($idArticle)

  Metoda nastavuje id článku pro který se budou ukládát soubory.

Veřejné atributy

- const **DB_TABLE_USER_FILES** = 'userfiles'

- const **USER_FILES_DIR** = 'userfiles'
- const **COLUM_ID** = 'id_file'
- const **COLUM_ID_USER** = 'id_user'
- const **COLUM_ID_ITEM** = 'id_item'
- const **COLUM_ID_ARTICLE** = 'id_article'
- const **COLUM_FILE** = 'file'
- const **COLUM_SIZE** = 'size'
- const **COLUM_TIME** = 'time'
- const **COLUM_LINK_TO_SHOW** = 'link_show'
- const **COLUM_LINK_TO_DOWNLOAD** = 'link_download'
- const **FORM_PREFIX** = 'userfiles_'
- const **FORM_NEW_FILE** = 'new_file'
- const **FORM_BUTTON_SEND** = 'send_file'
- const **FORM_USERFILE_ID** = 'id'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **CONFIG_TABLE_USERS** = 'users_table'
- const **CONFIG_TABLES_SECTIONS** = 'db_tables'

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda nastavuje id článku.

Chráněné atributy

- **$templateFile** = 'userfiles.tpl'

8.104.1 Detailní popis

Třída EPluginu pro přidávání souborů ke článku(stránce).

Třída umožňuje přidávat, mazat a sparvovat soubory přidané ke článku. Třída teké obstarává jejich výpis pomocí vlastní šablony to článku.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: userfiles.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída Epluginu pro práci se soubory, přikládanými do stránky

Plánované úpravy

dodělat mazání souborů z celého článku

Definice je uvedena na řádku 15 v souboru userfiles.class.php.

8.104.2 Dokumentace k metodám

8.104.2.1 UserFilesEplugin::setIdTpl ($ *id*)

Metoda nastaví id šablony pro výpis.

Parametry:

ineger – id šablony (jakékoliv)

Definice je uvedena na řádku 141 v souboru userfiles.class.php.

```
141                                 {
142         $this->idUserFiles = $id;
143     }
```

8.104.2.2 UserFilesEplugin::setIdArticle ($ *idArticle*)

Metoda nastavuje id článku pro který se budou ukládát soubory.

Parametry:

integer – id článku

Definice je uvedena na řádku 149 v souboru userfiles.class.php.

```
149                                             {
150         $this->idArticle = $idArticle;
151
152         $this->checkSendFile();
153
154         $this->checkDeleteFile();
155
156         $this->getFilesFromDb();
157     }
```

8.104.2.3 UserFilesEplugin::assignTpl () [protected]

Metoda nastavuje id článku.

Parametry:

integer – id článku Metoda obstarává přiřazení proměných do šablony

Reimplementuje stejnojmenný prvek z Eplugin.

Definice je uvedena na řádku 353 v souboru userfiles.class.php.

```
353                                 {
354         $this->toTpl("USERFILES_LABEL_NAME", _("Nahrané soubory"));
355         $this->toTpl("BUTTON_USERFILE_DELETE", _("Smazat"));
356         $this->toTpl("BUTTON_USERFILE_SEND", _("Přidat"));
357         $this->toTpl("FILE_NAME", _("Název souboru"));
358         $this->toTpl("FILE_SIZE_NAME", _("Velikost souboru"));
```

```
359       $this->toTpl("FILE_LINK_TO_SHOW_NAME", _("Odkaz pro zobrazení"));
360       $this->toTpl("FILE_LINK_TO_DOWNLOAD_NAME", _("Odkaz pro stažení"));
361       $this->toTpl("CONFIRM_MESAGE_DELETE", _("Opravdu smazat soubor"));
362
363       self::$otherNumberOfReturnRows[$this->idUserFiles] = $this->numberOfReturnRows;
364       $this->toTpl("USERFILES_NUM_ROWS", self::$otherNumberOfReturnRows);
365       $this->toTpl("USERFILES_ID", $this->idUserFiles);
366
367 //    if(!empty(self::$otherChanges)){
368 //       $array = self::$otherChanges;
369 //    }
370
371       $this->toTplJSPlugin(new SubmitForm());
372
373       self::$otherFilesArray[$this->idUserFiles] = $this->filesArray;
374
375       $this->toTpl("USERFILES_ARRAY",self::$otherFilesArray);
376
377
378
379    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/userfiles.class.php

8.105 Dokumentace třídy UserImagesEplugin

Třída EPluginu pro práci s obrázky ve stránce.

Diagram dědičnosti pro třídu UserImagesEplugin

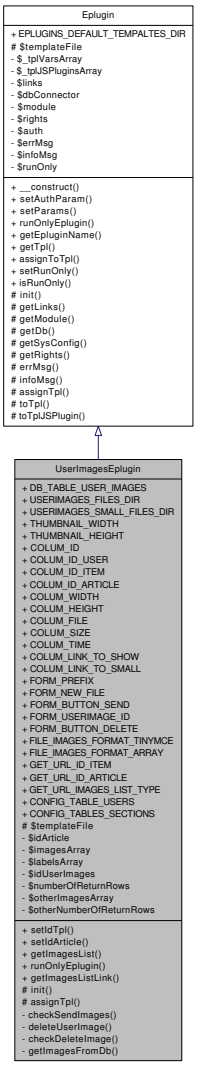
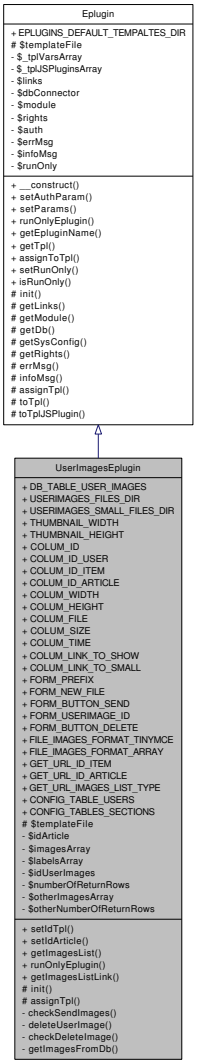


Diagram tříd pro UserImagesEplugin:



Veřejné metody

- setIdTpl ($id)

  Metoda nastaví id šablony pro výpis.

- setIdArticle ($idArticle)

  Metoda nastavuje id článku pro který se budou ukládát soubory.

- getImagesList ($idItem, $idArticle)

  Metoda načte seznam obrázků.

- runOnlyEplugin ()

  Metoda je spuštěna pokud se generuje soubor pro výstup (list obrázků).

- getImagesListLink ($type)

Metoda vrací odkaz na soubor se seznamem obrázků.

Veřejné atributy

- const **DB_TABLE_USER_IMAGES** = 'userimages'
- const **USERIMAGES_FILES_DIR** = 'userimages'
- const USERIMAGES_SMALL_FILES_DIR = 'small'

  Název adresáře s miniaturami obrázků.

- const **THUMBNAIL_WIDTH** = 110
- const **THUMBNAIL_HEIGHT** = 80
- const **COLUM_ID** = 'id_file'
- const **COLUM_ID_USER** = 'id_user'
- const **COLUM_ID_ITEM** = 'id_item'
- const **COLUM_ID_ARTICLE** = 'id_article'
- const **COLUM_WIDTH** = 'width'
- const **COLUM_HEIGHT** = 'height'
- const **COLUM_FILE** = 'file'
- const **COLUM_SIZE** = 'size'
- const **COLUM_TIME** = 'time'
- const **COLUM_LINK_TO_SHOW** = 'link_show'
- const **COLUM_LINK_TO_SMALL** = 'link_small'
- const **FORM_PREFIX** = 'userimages_'
- const **FORM_NEW_FILE** = 'new_file'
- const **FORM_BUTTON_SEND** = 'send_file'
- const **FORM_USERIMAGE_ID** = 'id'
- const **FORM_BUTTON_DELETE** = 'delete'
- const **FILE_IMAGES_FORMAT_TINYMCE** = 'tinymce'
- const **FILE_IMAGES_FORMAT_ARRAY** = 'array'
- const **GET_URL_ID_ITEM** = 'idI'
- const **GET_URL_ID_ARTICLE** = 'idA'
- const **GET_URL_IMAGES_LIST_TYPE** = 'type'
- const **CONFIG_TABLE_USERS** = 'users_table'
- const **CONFIG_TABLES_SECTIONS** = 'db_tables'

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

- assignTpl ()

  Metoda obstarává přiřazení proměných do šablony.

Chráněné atributy

- **$templateFile** = 'userimages.tpl'

8.105.1 Detailní popis

Třída EPluginu pro práci s obrázky ve stránce.

Třída slouží pro práci s obrázky v textech, jejich přidávání, mazání, zobrazování a správu. Obsahuje také svou vlastní šablonu pro jednodušší integraci k modulům. Je částečne napojena na TinyMce JsPlugin pro generování listu obrázků.

Verze:

$Id: userimages.class.php 3.1.8 beta1 13.11.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída Epluginu pro práci s obrázky, přikládanými do stránky používá také třídu TinyMce tinymce.class.php

Plánované úpravy

doělat mazání souborů z celého článku

Definice je uvedena na řádku 16 v souboru userimages.class.php.

8.105.2 Dokumentace k metodám

8.105.2.1 UserImagesEplugin::setIdTpl ($ *id*)

Metoda nastaví id šablony pro výpis.

Parametry:

ineger – id šablony (jakékoliv)

Definice je uvedena na řádku 149 v souboru userimages.class.php.

```
149                                 {
150         $this->idDwFiles = $id;
151     }
```

8.105.2.2 UserImagesEplugin::setIdArticle ($ *idArticle*)

Metoda nastavuje id článku pro který se budou ukládát soubory.

Parametry:

integer – id článku

Definice je uvedena na řádku 157 v souboru userimages.class.php.

```
157                                         {
158         $this->idArticle = $idArticle;
159         $this->checkSendImages();
160         $this->checkDeleteImage();
161         $this->getImagesFromDb();
162     }
```

8.105.2.3 UserImagesEplugin::getImagesList ($ *idItem*, $ *idArticle*)

Metoda načte seznam obrázků.

Parametry:

integer – id item

integer – id článku u kterého byla změna provedena

Definice je uvedena na řádku 275 v souboru userimages.class.php.

```
275                                                     {
276       $sqlSelect = $this->getDb()->select()->from(array('img' => self::DB_TABLE_USER_IMAGES), array(se
277                                   ->where(self::COLUM_ID_ITEM.' = '.$idItem)
278                                   ->where(self::COLUM_ID_ARTICLE.' = '.$idArticle);
279
280 //    vložení záznamu
281       $images = $this->getDb()->fetchAssoc($sqlSelect);
282
283       $returnArray = array();
284 //    Převedení na normální pole kde klíč je název souboru a hodota je cesta
285       if(!empty($images)){
286          foreach ($images as $image) {
287             $returnArray[$image[self::COLUM_FILE]] = $this->getLinks()->getMainWebDir().MAIN_DATA_DIR.
288          }
289       }
290       return $returnArray;
291    }
```

8.105.2.4 UserImagesEplugin::runOnlyEplugin ()

Metoda je spuštěna pokud se generuje soubor pro výstup (list obrázků).

Plánované úpravy

dodělat generování také do jiných typů souborů

Reimplementuje stejnojmenný prvek z Eplugin.

Definice je uvedena na řádku 366 v souboru userimages.class.php.

```
366                                     {
367       $array = $this->getImagesList(rawurldecode($_GET[self::GET_URL_ID_ITEM]),rawurldecode($_GET[self
368
369       isset($_GET[self::GET_URL_IMAGES_LIST_TYPE]) == false ? $type = null : $type = rawurldecode($_GE
370       switch ($type) {
371          case self::FILE_IMAGES_FORMAT_TINYMCE:
372             $data = TinyMce::generateListImages($array);
373             header("Content-Length: " . strlen($data));
374             header("Content-type: application/x-javascript");
375             echo $data;
376             exit();
377          break;
378          default:
379          break;
380       }
381       return false;
382    }
```

8.105.2.5 UserImagesEplugin::getImagesListLink ($ *type*)

Metoda vrací odkaz na soubor se seznamem obrázků.

Parametry:

string – typ v jakém se mají obrázky formátu vrátit

Návratová hodnota:

mixed – seznam obrázků

Plánované úpravy

dodělat tak by se daly předávat i celá pole v url parametrech, a jiné druhy souborů

Definice je uvedena na řádku 390 v souboru userimages.class.php.

Odkazuje se na Links::getMainWebDir().

```
390                                             {
391       switch ($type) {
392          case self::FILE_IMAGES_FORMAT_TINYMCE:
393             $link = new Links(true, true);
394             return rawurldecode($link->file('eplugin'.strtolower($this->getEpluginName()).'.js')->
395                   param(array(self::GET_URL_ID_ARTICLE => $this->idArticle, self::GET_URL_ID_ITEM => $
396                      self::GET_URL_IMAGES_LIST_TYPE => self::FILE_IMAGES_FORMAT_TINYMCE)));
397          break;
398
399          default:
400             $link = Links::getMainWebDir().'eplugin'.strtolower($this->getEpluginName()).'.js';
401          break;
402       }
403       return false;
404    }
```

Tato funkce volá...

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/userimages.class.php

8.106 Dokumentace třídy UsersController

Třída pro obsluhu akcí a kontrolerů modulu.

Diagram dědičnosti pro třídu UsersController

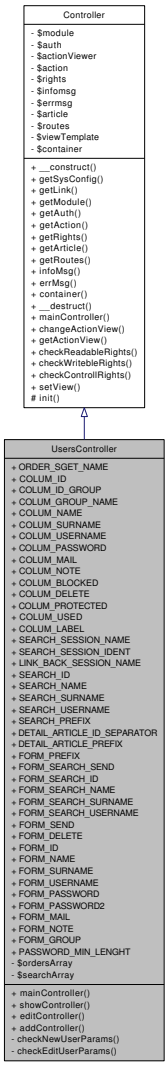
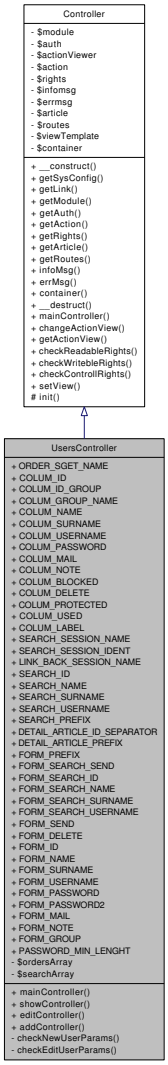


Diagram tříd pro UsersController:



Veřejné metody

- mainController ()

  Hlavní metoda třídy kontroleru, provádí se pokud není žádná akce.

- showController ()

  Metoda pro zobrazení detailu zástupce.

- editController ()

  Metoda pro úpravu.

- addController ()

  Metoda pro přidávání uživatelů.

Veřejné atributy

- const **ORDER_SGET_NAME** = 'order'
- const **COLUM_ID** = 'id_user'
- const **COLUM_ID_GROUP** = 'id_group'
- const **COLUM_GROUP_NAME** = 'group_name'
- const **COLUM_NAME** = 'name'
- const **COLUM_SURNAME** = 'surname'
- const **COLUM_USERNAME** = 'username'
- const **COLUM_PASSWORD** = 'password'
- const **COLUM_MAIL** = 'mail'
- const **COLUM_NOTE** = 'note'
- const **COLUM_BLOCKED** = 'blocked'
- const **COLUM_DELETE** = 'deleted'
- const **COLUM_PROTECTED** = 'protected'
- const **COLUM_USED** = 'used'
- const **COLUM_LABEL** = 'label'
- const **SEARCH_SESSION_NAME** = 'search'
- const **SEARCH_SESSION_IDENT** = 'user'
- const **LINK_BACK_SESSION_NAME** = 'link_back'
- const **SEARCH_ID** = 'id'
- const **SEARCH_NAME** = 'name'
- const **SEARCH_SURNAME** = 'surname'
- const **SEARCH_USERNAME** = 'username'
- const **SEARCH_PREFIX** = 'userss_search_'
- const **DETAIL_ARTICLE_ID_SEPARATOR** = '-'
- const **DETAIL_ARTICLE_PREFIX** = 'iduserdet'
- const **FORM_PREFIX** = 'user_'
- const **FORM_SEARCH_SEND** = 'search_send'
- const **FORM_SEARCH_ID** = 'search_id'
- const **FORM_SEARCH_NAME** = 'search_name'
- const **FORM_SEARCH_SURNAME** = 'search_surname'
- const **FORM_SEARCH_USERNAME** = 'search_username'
- const **FORM_SEND** = 'send'
- const **FORM_DELETE** = 'delete'
- const **FORM_ID** = 'id'
- const **FORM_NAME** = 'name'
- const **FORM_SURNAME** = 'surname'
- const **FORM_USERNAME** = 'username'
- const **FORM_PASSWORD** = 'password'
- const **FORM_PASSWORD2** = 'password2'
- const **FORM_MAIL** = 'mail'
- const **FORM_NOTE** = 'note'
- const **FORM_GROUP** = 'group'
- const **PASSWORD_MIN_LENGHT** = 5

8.106.1 Detailní popis

Třída pro obsluhu akcí a kontrolerů modulu.

Definice je uvedena na řádku 7 v souboru controler.class.php.

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/users/controler.class.php

8.107 Dokumentace třídy UsersView

Třída pro vytvoření a obsluhu pohledů.

Diagram dědičnosti pro třídu UsersView

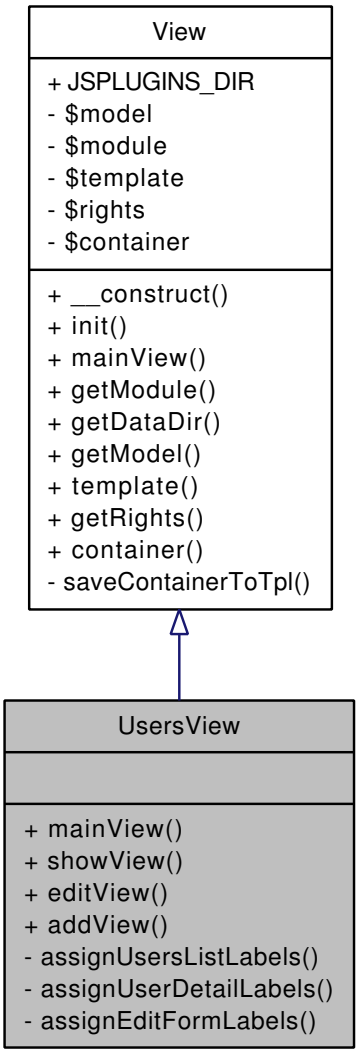
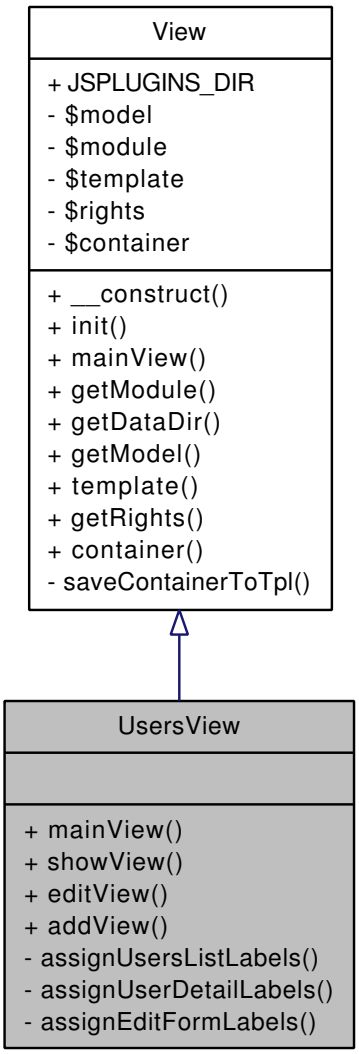


Diagram tříd pro UsersView:



Veřejné metody

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- showView ()

  Viewer pro zobrazení detailu.

- **editView** ()
- **addView** ()

8.107.1 Detailní popis

Třída pro vytvoření a obsluhu pohledů.

Definice je uvedena na řádku 7 v souboru view.class.php.

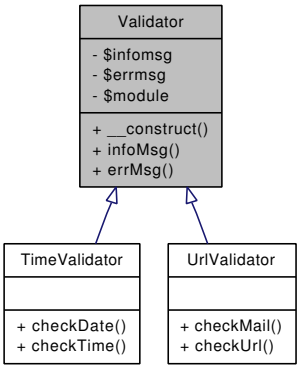
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/modules/users/view.class.php

8.108 Dokumentace třídy Validator

Třída pro validaci prvků Třída obsluhuje zakladní třídu pro tvorbu validátorů.

Diagram dědičnosti pro třídu Validator



Veřejné metody

- __construct ()

  Konstruktor nastaví základní parametry.

- infoMsg ()

  Metoda vrací objekt s informačními zprávami.

- errMsg ()

  Metoda vrací objekt s chybovými zprávami.

8.108.1 Detailní popis

Třída pro validaci prvků Třída obsluhuje zakladní třídu pro tvorbu validátorů.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: validator.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro validaci

Definice je uvedena na řádku 12 v souboru validator.class.php.

8.108.2 Dokumentace k metodám

8.108.2.1 Validator::infoMsg () [final]

Metoda vrací objekt s informačními zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 52 v souboru validator.class.php.

```
52                                  {
53        return $this->infomsg;
54    }
```

8.108.2.2 Validator::errMsg () [final]

Metoda vrací objekt s chybovými zprávami.

Návratová hodnota:

Messages – objekt zpráv

Definice je uvedena na řádku 60 v souboru validator.class.php.

```
60                                  {
61        return $this->errmsg;
62    }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Validators/validator.class.php

8.109 Dokumentace třídy VerifyImageEplugin

Description of verifyimageclass.

Diagram dědičnosti pro třídu VerifyImageEplugin

Diagram tříd pro VerifyImageEplugin:

Veřejné metody

- verifyImage ($post_name)

  Funkce ověřuje správné zadání kontrloního obrázk //TODO není třeba.

- runOnlyEplugin ()

  Metoda je spuštěna při načítání souborů epluginu Zde je odeslán soubor s obrázkem.

- getImageName ()

  Metoda vrací cestku ke kontrolnímu obrázku.

Veřejné atributy

- const **FONTS_DIR** = 'fonts'

- const **FONT_NAME** = 'Timeless-Bold.ttf'
- const **SESSION_NAME** = 'control_chars'
- const **IMAGE_NAME** = 'epluginVerifyImage.jpg'

Chráněné metody

- init ()

  Metoda inicializace, je spuštěna pří vytvoření objektu.

8.109.1 Detailní popis

Description of verifyimageclass.

Copyright (c) 2008 Jakub Matas

Verze:

Id

verifyimageclass.class.php 419 2008-11-28 23:21:19Z jakub

VVE3.3.0

Revision

419

Autor:

Author

jakub

Date

2008-11-28 23:21:19 +0000 (Pá, 28 lis 2008)

LastChangedBy

jakub

LastChangedDate

2008-11-28 23:21:19 +0000 (Pá, 28 lis 2008)

text

Definice je uvedena na řádku 11 v souboru verifyimage.class.php.

8.109.2 Dokumentace k metodám

8.109.2.1 VerifyImageEplugin::verifyImage ($ *post_name*)

Funkce ověřuje správné zadání kontrloního obrázk //TODO není třeba.

Parametry:

string – název postu s odesýlaným textem

Návratová hodnota:

boolean – true pokud se data shodují

Definice je uvedena na řádku 61 v souboru verifyimage.class.php.

Odkazuje se na Eplugin::errMsg().

```
62    {
63       $post = htmlspecialchars($_POST[$post_name]);
64       if($post == $this->previousCode){
65          return true;
66       } else {
67          $this->errMsg()->addMessage(_('Kontrolní obrázek nebyl správně opsán'));
68          return false;
69       }
70    }
```

Tato funkce volá...

8.109.2.2 VerifyImageEplugin::runOnlyEplugin ()

Metoda je spuštěna při načítání souborů epluginu Zde je odeslán soubor s obrázkem.

Plánované úpravy

– dodělat zjitování pozadí obrázku podle zvolené faces

Reimplementuje stejnojmenný prvek z Eplugin.

Definice je uvedena na řádku 92 v souboru verifyimage.class.php.

```
92                                    {
93       $fontDir = './'.self::FONTS_DIR.'/'.self::FONT_NAME;
94
95       $chars = $this->session->get(self::SESSION_NAME);
96       $backlayer = ImageCreateFromPNG('./'.AppCore::TEMPLATES_IMAGES_DIR."/control_chars_back.png");
97       $frontlayer = imagecreatefromPNG('./'.AppCore::TEMPLATES_IMAGES_DIR."/control_chars_front.png");
98       $color = ImageColorAllocate($backlayer, 0, 0, rand(0,200));
99       ImageTTFText($backlayer, rand(25,30), rand(-40,40), 20, 40, $color, $fontDir, $chars[0]);
100       $color = ImageColorAllocate($backlayer, 0, 0, rand(0,200));
101       ImageTTFText($backlayer, rand(25,30), rand(-40,40), 60, 40, $color, $fontDir, $chars[1]);
102       $color = ImageColorAllocate($backlayer, 0, 0, rand(0,200));
103       ImageTTFText($backlayer, rand(25,30), rand(-40,40), 100, 40, $color, $fontDir, $chars[2]);
```

```
104       $color = ImageColorAllocate($backlayer, 0, 0, rand(0,200));
105       ImageTTFText($backlayer, rand(25,30), rand(-40,40), 140, 40, $color, $fontDir, $chars[3]);
106       $color = ImageColorAllocate($backlayer, 0, 0, rand(0,200));
107       ImageTTFText($backlayer, rand(25,30), rand(-40,40), 180, 40, $color, $fontDir, $chars[4]);
108       imagecopy($backlayer, $frontlayer, 0, 0, 0, 0, 220, 51 );
109       header("Content-type: image/png");
110       ImagePNG($backlayer);
111       ImageDestroy($backlayer);
112       ImageDestroy($fronlayer);
113    }
```

8.109.2.3 VerifyImageEplugin::getImageName ()

Metoda vrací cestku ke kontrolnímu obrázku.

Návratová hodnota:

string – cestak k obrázku

Definice je uvedena na řádku 119 v souboru verifyimage.class.php.

```
119                                  {
120       $link = new Links(true, true, true);
121       return $link.self::IMAGE_NAME;
122    }
```

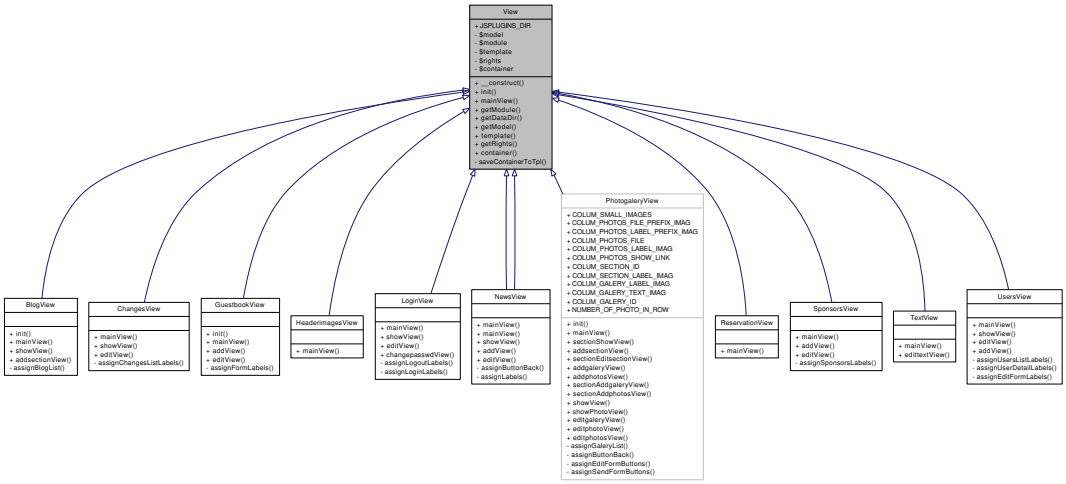
Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/EPlugins/verifyimage.class.php

8.110 Dokumentace třídy View

Abstraktní třída pro objektu viewru.

Diagram dědičnosti pro třídu View



Veřejné metody

- __construct (Module $module, Rights $rights, Template &$template, Container $container)

  Konstruktor Viewu.

- init ()

  Metoda, která se provede vždy.

- mainView ()

  Hlavní abstraktní třída pro vytvoření pohledu.

- getModule ()

  Metoda vrací objekt modulu.

- getDataDir ()

  Funkce vrací datový adresář modulu.

- getModel ()

  Funkce vrací objekt modelu.

- template ()

  Metoda vrací objekt šablony, přes který se přiřazují proměnné do šablony.

- getRights ()

  Metoda vrací objekt k právům uživatele.

- container ()

  Metoda vrací objekt kontaineru.

Veřejné atributy

- const **JSPLUGINS_DIR** = 'JsPlugins'

8.110.1 Detailní popis

Abstraktní třída pro objektu viewru.

Třídá slouží jako základ pro tvorbu Viewrů jednotlivých modulů. Poskytuje základní paramtery a metody k vytvoření pohledu modulu.

Copyright (c) 2008 Jakub Matas

Verze:

$Id: view.class.php 3.0.0 beta1 29.8.2008

Autor:

Jakub Matas <jakubmatas@gmail.com> Třída pro tvorbu a obsluhu pohledu

Definice je uvedena na řádku 12 v souboru view.class.php.

8.110.2 Dokumentace konstruktoru a destruktoru

8.110.2.1 View::__construct (Module $ *module*, Rights $ *rights*, Template &$ *template*, Container $ *container*)

Konstruktor Viewu.

Parametry:

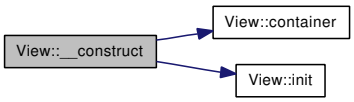
Model – použitý model

Definice je uvedena na řádku 53 v souboru view.class.php.

Odkazuje se na container() a init().

```
53                                                                                                  {
54         $this->rights = $rights;
55         $this->module = $module;
56         $this->template = $template;
57         $this->container = $container;
58
59 //      inicializace viewru
60         $this->init();
61
62         $this->saveContainerToTpl();
63     }
```

Tato funkce volá...



8.110.3 Dokumentace k metodám

8.110.3.1 View::getModule () [final]

Metoda vrací objekt modulu.

Návratová hodnota:

Module – objekt modulu

Definice je uvedena na řádku 97 v souboru view.class.php.

Používá se v getDataDir().

```
97                                      {
98         return   $this->module;
99     }
```

8.110.3.2 View::getDataDir () [final]

Funkce vrací datový adresář modulu.

Návratová hodnota:

string – datový adresář modulu

Definice je uvedena na řádku 105 v souboru view.class.php.

Odkazuje se na getModule().

```
105                                        {
106        return $this->getModule()->getDir()->getDataDir();
107     }
```

Tato funkce volá...

8.110.3.3 View::getModel () [final]

Funkce vrací objekt modelu.

Návratová hodnota:

Models – objekt modelu

Definice je uvedena na řádku 113 v souboru view.class.php.

Používá se v SponsorsView::addView(), GuestbookView::addView(), SponsorsView::editView(), GuestbookView::editView(), UsersView::mainView(), SponsorsView::mainView(), Reservation-View::mainView(), ChangesView::mainView() a UsersView::showView().

```
113                                     {
114        return $this->model;
115     }
```

8.110.3.4 View::template () [final]

Metoda vrací objekt šablony, přes který se přiřazují proměnné do šablony.

Návratová hodnota:

Template – objekt šablony

Definice je uvedena na řádku 121 v souboru view.class.php.

```
121                                     {
122         //TODO zbytečné
123         if($this->template == null){
124             return $this->template = new Template();
125         } else {
126             return $this->template;
127         }
128     }
```

8.110.3.5 View::getRights () [final]

Metoda vrací objekt k právům uživatele.

Návratová hodnota:

Rights – objekt práv

Definice je uvedena na řádku 134 v souboru view.class.php.

Používá se v UsersView::mainView(), TextView::mainView(), SponsorsView::mainView(), NewsView::mainView(), LoginView::mainView(), HeaderimagesView::mainView(), BlogView::mainView() a UsersView::showView().

```
134                                     {
135         return $this->rights;
136     }
```

8.110.3.6 View::container () [final]

Metoda vrací objekt kontaineru.

který slouží pro přenos dat z kontroleru do viewru

Návratová hodnota:

Container – objekt s daty

Definice je uvedena na řádku 143 v souboru view.class.php.

Používá se v __construct(), TextView::mainView(), NewsView::mainView(), LoginView::mainView(), HeaderimagesView::mainView() a BlogView::mainView().

```
143                                     {
144         return $this->container;
145     }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

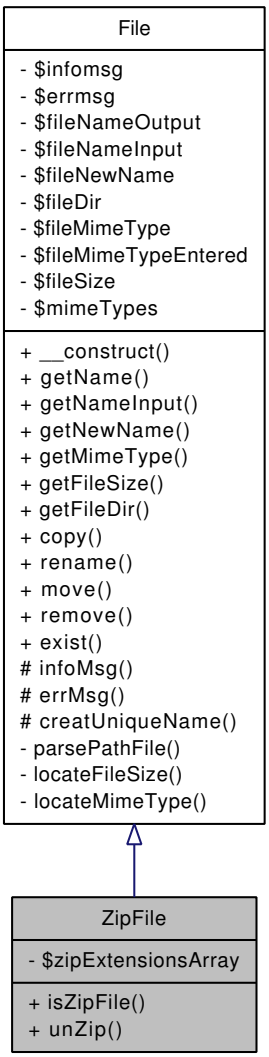
- /home/cuba/work-net/vve3_2/lib/view.class.php

8.111 Dokumentace třídy ZipFile

Metoda pro práci se soubory typu zip, umožňuje rozbalování souborů, pakování.

Diagram dědičnosti pro třídu ZipFile

Diagram tříd pro ZipFile:



Veřejné metody

- isZipFile ()

  Metoda zjišťuje, zdali je uploadovaný soubor zip soubor.

- unZip ($src_file, $dest_dir=false, $create_zip_name_dir=true, $overwrite=true)

  Rozbali zip soubor do cílového adresáře.

8.111.1 Detailní popis

Metoda pro práci se soubory typu zip, umožňuje rozbalování souborů, pakování.

Copyright (c) 2008 Jakub Matas

Verze:

Id

VVE3.5.0

Revision

Autor:

Author

Date

LastChangedBy

LastChangedDate

Třída pro práci se ZIP soubory

Definice je uvedena na řádku 11 v souboru zipfile.class.php.

8.111.2 Dokumentace k metodám

8.111.2.1 ZipFile::isZipFile ()

Metoda zjišťuje, zdali je uploadovaný soubor zip soubor.

Parametry:

string – soubor, který se zjišťuje

Definice je uvedena na řádku 25 v souboru zipfile.class.php.

Odkazuje se na File::getMimeType().

```
25                                  {
26       if(in_array($this->getMimeType(), $this->zipExtensionsArray)){
27          return true;
28       } else {
29          return false;
30       }
31    }
```

Tato funkce volá...

8.111.2.2 ZipFile::unZip ($ *src_file*, $ *dest_dir* = `false`, $ *create_zip_name_dir* = `true`, $ *overwrite* = `true`)

Rozbali zip soubor do cílového adresáře.

Parametry:

string – Cesta k zip souboru

string – Cesta, kam se zip soubor rozbalí, (false rozbalí soubor do aktuálního adresáře se zip souborem)

boolean – Jestli má být zip soubor rozbalen do adresáře se stejným jménem

boolean – Jestli mají být soubory přepsány

Návratová hodnota:

boolean Succesful or not

Definice je uvedena na řádku 43 v souboru zipfile.class.php.

```
44    {
45       if(function_exists("zip_open"))
46       {
47          if(!is_resource(zip_open($src_file)))
48          {
49             $src_file=dirname($_SERVER['SCRIPT_FILENAME'])."/".$src_file;
50          }
51
52          if (is_resource($zip = zip_open($src_file)))
53          {
54             $splitter = ($create_zip_name_dir === true) ? "." : "/";
55             if ($dest_dir === false) $dest_dir = substr($src_file, 0, strrpos($src_file, $splitter))."/
56
57             // Create the directories to the destination dir if they don't already exist
58             $this->createDirs($dest_dir);
59
60             // For every file in the zip-packet
61             while ($zip_entry = zip_read($zip))
62             {
63                // Now we're going to create the directories in the destination directories
64
65                // If the file is not in the root dir
66                $pos_last_slash = strrpos(zip_entry_name($zip_entry), "/");
67                if ($pos_last_slash !== false)
68                {
69                   // Create the directory where the zip-entry should be saved (with a "/" at the end)
70                   $this->createDirs($dest_dir.substr(zip_entry_name($zip_entry), 0, $pos_last_slash+1))
71                }
72
73                // Open the entry
74                if (zip_entry_open($zip,$zip_entry,"r"))
75                {
76
77                   // The name of the file to save on the disk
```

```
78                   $file_name = $dest_dir.zip_entry_name($zip_entry);
79
80                   // Check if the files should be overwritten or not
81                   if ($overwrite === true || $overwrite === false && !is_file($file_name))
82                   {
83                      // Get the content of the zip entry
84                      $fstream = zip_entry_read($zip_entry, zip_entry_filesize($zip_entry));
85
86                      if(!is_dir($file_name))
87                      file_put_contents($file_name, $fstream );
88                      // Set the rights
89                      if(file_exists($file_name))
90                      {
91                         chmod($file_name, 0777);
92                      }
93                   }
94
95                   // Close the entry
96                   zip_entry_close($zip_entry);
97                }
98             }
99             // Close the zip-file
100            zip_close($zip);
101         }
102         else
103         {
104            //        echo "No Zip Archive Found.";
105            return false;
106         }
107
108         return true;
109      }
110      else
111      {
112         if(version_compare(phpversion(), "5.2.0", "<"))
113         $infoVersion="(use PHP 5.2.0 or later)";
114
115         new CoreException(_('Je potřeba PHP zip rozšíření pro práci se zip archívy ').$infoVersion);
116      }
117   }
```

Dokumentace pro tuto třídu byla generována z následujícího souboru:

- /home/cuba/work-net/vve3_2/lib/Filesystem/zipfile.class.php

Kapitola 9

VVE Dokumentace souvisejících stránek

9.1 Seznam plánovaných úprav

Člen AppCore::renderTemplate() dořešit při neexistenci ostatní typů medií

Člen AppCore::runModules() – implementovat do modelu

Třída ChangesEPlugin implementovat možnost zobrazení změn ve vlastním novém okně

Člen Config::getOptionValue($option, $parentKey=null) dodělat vracení z větší hloubky stromu

Třída CoreException Není plně implementována a chce dodělat

Člen CsvDataEplugin::sendData($fileName= 'data') ověřit a popípadě dodělat správné odesílání hlavičky

Člen CsvDataEplugin::saveCsvData($file) implementovat ukládání do souboru na serveru (např. pro další stažení)

Člen DateTimeCtrlHelper::createStampSmartyPost($postName) dodělat generování hodin:minut:sekund

Třída Dir – refaktoring, tak aby to byl objekt plnohodnotný

Člen Dir::checkDir($directory) dodělat přidávání lomítek před adresář

Třída Eplugin implementovat generování názvu souborů pro zvolený eplugin

Člen Eplugin::__construct(Rights $rights=null) dodělat, tak ať se to nčítá třeba přímo z modulu nebo kategoorie, nebo jádra

Třída Errors odstranit (popřípadě pokud se využije dodělat popisy a dokumentaci)

Člen File::getMimeType() nutná portace na PECL rozšíření o informací o souboru

Člen File::creatUniqueName($destinationDir, $newName=null, $number=0) – dodělat při přijmu souboru se dvěmi příponami

Třída Form Dodělat další validace, implementovat ostatní prvky formulářů

Člen Form::debug() Odstranit

Člen JQuery::addWidgentDatepicker() zkontrolovat závislos s obrázky umístěnými ve složce pluginu

Člen Links::__toString() What is this? Know anybody what is this doing?

Člen MailCtrlHelper::checkMail($email) – odstranit (je obsažena v UrlValidator)

Člen Mysql_Db_Delete::where($condition, $operator=selfSQL_AND) dodělat aby se doplňovali magické uvozovky do lauzule

Třída Panel Není implementována práce s chybami

Třída ProgressBarEplugin dodělat možnost vytvářet vlastní styl okna

Člen ProgressBarEplugin::setWindowTitle($title) implementovat nastavení titulku okna

Třída SendMailEplugin Dodělat podporu pro více druhů výrazů (ne jenom vyraz%, vyraz[true/false]%) u zpracování

Člen UrlValidator::checkUrl($url) – dodělat, není implementována

Třída UserFilesEplugin dodělat mazání souborů z celého článku

Třída UserImagesEplugin doělat mazání souborů z celého článku

Člen UserImagesEplugin::runOnlyEplugin() dodělat generování také do jiných typů souborů

Člen UserImagesEplugin::getImagesListLink($type) dodělat tak by se daly předávat i celá pole v url parametrech, a jiné druhy souborů

Člen VerifyImageEplugin::runOnlyEplugin() – dodělat zjitování pozadí obrázku podle zvolené faces

9.2 Seznam zastaralých prvků

Člen DateTimeCtrlHelper::checkDate($date, $timestam=false) – je implementována ve validátoru TimeValidator

Člen DateTimeCtrlHelper::checkTime($time, $timestamp=false) – je implementována ve validátoru TimeValidator

Třída Errors

Třída Images – odstranit, protože se používá třída ImageFile která pracuje i se souborem

Člen LocaleCtrlHelper::generateArray($arrayElements, $values=null, $alternativeValues=null) – lepší použít přímo validítor formuláře

Člen LocaleCtrlHelper::postsToArray($sendPostsArray, $postPrefix=null, $specialChars=selfSP_CHARS_ENCODE, $ – lepší použít přímo validítor formuláře

Člen MailCtrlHelper::checkMail($email) – je lepší použít UrlValidator

Člen MailCtrlHelper::translateText($text, $translateArray)

Třída SpecialFunctions nehrtazuje se ostatními helpery a validátory zvláště texthelperem

Člen TextCtrlHelper::decodeHtmlSpecialChars($text)

Index