

Frontend installation and configuration

1. *Preconditions*

- Apache HTTP server with working PHP4 or PHP5 running under user „apache“ and group „apache“. If your operating system uses different username nad/or group you should alter
- Running instance of OasisMiddleware.
- A directory writable by apache user or apache group eg. /tmp/oasisfe-cache

2. *Installation*

- Extract the ZIP archive to the WWW directory of Apache eg. /var/www/OasisFrontend
- Change the value of MIDDLEWARE_URL constant in file config/conf_local.php
- Change value of DIR_ATTR_CACHE constant in file config/conf_local.php which should refer to the directory writable by apache user. Frontend will use this directory to store transalted OASIS attributes fetched from the OasisMiddleware.

3. *Main Configuration Items*

All important counstants concerning OasisFrontend configuration are located in config/conf_local.php.

Constant name	Description
DEBUG_MODE_ENABLED	The goal of this constant is to provide simple yet powerfull mechnism to control debug messages. If the value is TRUE, the function debug() writes messages to STDERR which is most likely the file /var/log/httpd/error_log Messages are prepended by FRONTEND: prefix. to distinguish among applications writing to errorlog.
MIDDLEWARE_URL	URL of running OasisMiddleware instance used by frontend. This is probably the only configuration item user need to change.
DEFAULT_LANGUAGE	default is „en“
WAIT_FOR_DATA_DELAY	Sleep between checking of loaded data to appear (in seconds). Another important part of the waiting mechanism is WAIT_FOR_DATA_TIMEOUT. It is possible to optimise this value to achieve better performance/responsivness.
WAIT_FOR_DATA_TIMEOUT	How long should the Frontend wait for Middleware (in seconds). Another important part of the waiting mechanism is WAIT_FOR_DATA_DELAY. It is possible to optimise this value to achieve better performance/responsivness.
DIR_ATTR_CACHE	Where to store serialized list of all attributes downloaded from middleware. Recomendand value is

Constant name	Description
	/dev/shm/. Set the value to /dev/null/ if you want to turn the caching off. The directory name must end with '/
DEFAULT_RESULT_COUNT	Default number of results fetched from middleware and rendered on one page.
DEFAULT_ORDERBY_ATTRIBUTE	Default attribute name used to order the results.
MAX_RESULT_COUNT	Maximum number of results fetched from middleware and rendered on one page.
MAX_LISTING_PAGES	Maximum number of pages in pager used by result grid.

4. Design customization

Our web-frontend architecture is designed according to MVC (Model View Controller) architectural pattern [17] which solves the problem of separation of data and user interface by decoupling data access and business logic from data presentation and user interaction, by introducing an intermediate component: the controller:

- **Model:** The domain-specific representation of the information on which the application operates. This comprises persistent repository and entities implementing the application logic. From the OasisFrontend's point of view, this is the OasisMiddleware layer. From OasisMiddleware's point of view this is DB-Adapter layer.
- **View:** This layer renders the user interface. Concerning the OasisFrontend, all related files are located in `tpl/` directory, that is why we are referring to the term template-layer.
- **Controller:** Processes and responds to events, typically user actions, and may invoke changes on the model. In OasisFrontend the controller is implemented as PHP class `Application`. Each action has been implemented as an extension of abstract class `BaseAction`. The action is instantiated by controller according to `ACT HTTP` request parameter. During the lifecycle of each action, the method `setActionTemplate()` must be called in order to specify the template name used in template layer. In case of an error, or undefined template, the default `tpl/error.php` template will be used.

In order to change design specific parts of the frontend one should look into `tpl/` directory. The main template file `tpl/skeleton.php` specifies common HTML structure augmented with the `tpl/style.css` file which defines common CSS rules. Moreover, each action automatically uses specific CSS file located in directory `tpl/autocss/`.

Example: `setActionTemplate('my_action.php')` implies that `tpl/my_action.php` will be used as a template together with `tpl/autocss/my_action.php` as additional CSS file.

Following table describes the structure of design-related files and directories.

directory or file	description
<code>tpl/</code>	Contains all design-related files.
<code>tpl/autocss/</code>	Files included automatically according to action's template.

directory or file	description
tpl/js/	All JavaScripts included in application
tpl/img/	All images used in application.
tpl/img/*/*.gif	Language related images such as Search button, Login button ...
tpl/skeleton.php	Main template file which includes other templates according to \$action->getTemplate() value.
tpl/style.css	Common CSS definitions.
tpl/error.php	Error page
tpl/layout_*.php	Code which implements particular layout such as Detail, List view, Grid view...
tpl/inc_*.php	included design-related scripts

5. Authentication / Authorization

If available rules and requests on authentication and authorization demands at the OASIS Middleware

There is a LOGIN page in OasisFrontend where user could provide username and password. After submitting the login form, the authentication data are sent by OasisMiddleware to each DB-Adapter registered in the system. Each DB-Adapter responds to the request and returns a description text about the authentication process. This text is intended only to inform user whether the username:password has been accepted or not (and why).

Afterwards, OasisFrontend sends the username/password pair in further queries to middleware which sends it to DB-Adapters. DB-Adapter should use this pair to alter the search results in an unspecified way. It could provide more results and/or slightly modified results comparing to results without authentication.

5.1. Reflection of User Interface / Frontend (Description of Fields requestes)

If you wish you might also add some information about former interface proposals that have been rejected by the OASIS group.

6. Localization

Default set of supported languages includes: English (en), German (de), French (fr), Polish (pl), Czech (cz), Slovak (sk). The localization process consists of two main parts:

1. **Frontend part:** In order to support a new language, the user interface must be modified to provide all textual descriptions in the new language.
2. **Middleware part:** While the Middleware contains the list of all OASIS attributes, it is necessary to translate all labels, descriptions and enumerations.

6.1. Frontend Localization

Localized descriptions which could be rendered on the web page are located in languages/lang_{LANG}.php file, where {LANG} is the ISO-639-1 language code [18].

Each language is represented by national flag symbol, located in `languages/lang_{LANG}.png` file.

The last part of frontend localization is to replace images of buttons containing text such as Search, Advanced Search, Login etc. These images are located in `tpl/img/{LANG}/` directory.

6.2. Middleware Localization

In chapter **Fehler! Verweisquelle konnte nicht gefunden werden.** we described the process of modification of `oasis-attributes.xml` document containing localized values of all OASIS attributes.

In order to include a new language we have to perform following actions:

- Element `<language>` must be inserted as a subelement of `<languages>`
- Element `<label>` must be inserted as a subelement of each `<attribute>`
- Element `<description>` must be inserted as a subelement of each `<attribute>`
- Element `<enumeration>` must be inserted as a subelement of each `<attribute>`, which has `type="enum"`
- XML file must be recompiled using mini-compiler (see chapter Customization of OASIS Attributes)
- Finally, cache of all connected frontends must be deleted in order to synchronize changes.

Viliam Simko