[EN] 02. The Extensions of the Editor and Administrator Application

Introduction

The extensions of the Editor and Administrator Application are based on the Java Plugin Framework (JPF). All possible extensions (extension points) defined for the Editor and Administrator Applications are described below.

Note

The programming interfaces described below are in `dcore-app-extension-api` library.

The **Metadata** Extension

The **metadata** extension is for expanding the metadata import and export capabilities of the application. The extension makes it possible to add the function of importing/exporting metadata from/to an external format.

The extension has two parameters:

- class – the name of the class which implements the programming interface of that extensions, and
- name – the name of the extension.

The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.metadata.MetadataFinder`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

There are four metadata extensions installed by default:

- importing metadata from the MARC format (`dlibra-app-extension-mf-marc`),
- importing metadata from the XML format (`dcore-app-extension-mf-xml`),
- import metadata from the BibTeX format (`dlibra-app-extension-mf-bibtex`), and
- exporting metadata in the RDF format – that extension is integrated with the Editor and Administrator Application; it is not defined in a separate project.

The configuration of particular extensions is presented here.

The **fileParser** Extension

The **fileParser** Extension finds files related to a particular file. After the main file has been selected in the Editor and Administrator Application (for example, in the publication creator), the system automatically checks which files are related to that file – by default, with the use of fileParser extensions. By default, HTML and DjVu extensions are installed in the application. If no extension is defined for the selected file, the editor must decide what kind of files will be the content of the publication.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.fileparser.FileParser`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The following extensions of that type are installed by default:

- the DjVu format – that extension finds files related to the given DjVu file (`dcore-app-extension-fp-djvu`),
- the HTML format – that extension finds files related to the given HTML file (`dcore-app-extension-fp-html`).

The **interfaceLanguage** Extension

The interfaceLanguage Extension makes it possible to add translations of the user interface of the Editor and Administrator Application. The default interfaceLanguage extensions are for Polish, English, and Ukrainian. The extension allows adding translations of the interface. Please contact us for the list of phrases/words for translation.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.language.InterfaceLanguage`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The following extensions of that type are installed by default:

- Polish – for making all user interface words/phrases available in Polish (`dlibra-app-extension-il-pl`), and
- English – for making all user interface words/phrases available in English (`dlibra-app-extension-il-en`).

The **graphicProvider** Extension

That extension makes it possible to change the look of graphic elements in the Editor and Administrator Application. When an extension of that type is added, the look of icons and images in the application can be changed so that they match the (cultural or technical) context better. It is recommended that only one plugin of that type be installed the application because the application does not have a mechanism for selecting a graphic context (in analogy to the language selection).

By default, the application is offered with a pre-installed plugin which ensures the default look (`dlibra-app-extension-gp`).

The **sourceLocator** Extension

That extension makes it possible to locate files with metadata for later import. It allows the introduction of a new function for locating metadata files.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.sourcelocator.SourceLocator`. SourceLocator. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The following extensions of that type are installed by default:

- Z39.50 – makes it possible to locate metadata files with the use of Z39.50 servers (`dlibra-app-extension-sl-z3950`).

The configuration of particular extensions is presented here.

The **dictionaryManager** Extension

That extension makes it possible to import and export attribute value groups from and to internal files.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.dictionarymanager.DictionaryManager`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The following extensions of that type are installed by default:

- import of value groups from files in the MARC format – it makes it possible to read files in the MARC-21 format (`dlibra-app-extension-dl-marc`); saving to files is not supported; and
- import/export of value groups from/to files in the XML format – it makes it possible to save and read files in the XML format. It can be used for copying attribute values between libraries (`dcore-app-extension-dm-dictcopy`).

The configuration of particular extensions is presented here.

The **filesHandler** Extension

That extension makes it possible to define procedures to be carried out after the main file has been selected in the new publication creator. Those procedures may include file processing on a drive, displaying dialog boxes, etc. A procedure defined by such an extension can result in a change of the selected main file and of the remaining files added to an edition. If there is more than one plugin for that extension in the application, the plugins are called one by one, until one of them processes the files correctly (does not throw an exception).

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.fileshandler.FilesHandler`. The filesHandler Extension For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The following extensions of that type are installed by default:

- JPG presentations – if a file containing files in the JPG format has been selected, that plugin makes it possible to define the order of images and to assign a description to every image; the information is saved in file Presentation Data.xml which is marked as the main file (`dcore-app-extension-fh-jpg`).

The **Tool** Extension

That extension makes it possible to add new elements to the "Tools" menu in the Administrator or Editor Application (see the figure below). The extension has two parameters:

- class, which specifies the name of the class which implements the programming interface of the extension; the programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.tool.Tool`; for more information about that interface, see the comments on the methods and the interface (JavaDocs), and
- type, which determines the use of the tool, that is, if the tool belongs to the Administrator Application or to the Editor Application; so the parameter can only have one of the two values: ADMIN (for the Administrator Application) or EDITOR (for the Editor Application); when the parameter is set, the tool will be displayed in the "Tools" menu in the Administrator Application or in the Editor Application.

The following extensions of that type are installed by default:

- a publication list in a group publication – for saving publications which belong to a selected group publication, in a file in the HTML format (`dcore-app-extension-tl-leaflist`); and
- an tool for cleaning an attribute dictionary – for removing those values which are not assigned to any publication; that extension is directly integrated with the Administrator Application; it is not defined in a separate project.

Apart from the extensions mentioned above, the team which develops the dLibra system has prepared sample tools – which illustrate the possibilities of tool-type extensions – for programmers interested in developing their own extensions.

The **miniatureProvider** Extension

That extension makes it possible to automatically generate thumbnails for added publications, on the basis of files selected for the editions. Several thumbnails can be generated for one publication, and the editor can choose one of them.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.miniatureprovider.MiniatureProvider`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

The plugin installed by default to implement that extension is a plugin which creates presentations on the basis of files in the JPG format (see the description of the filesHandler extension). Besides, in the Editor Application, there are integrated plugins for generating thumbnails from the following files:

- the most popular graphics file formats (JPG, PNG, GIF),
- files in the DjVu format, and
- files in the PDF format.

The **eventListener** Extension

An extension of the eventListener type makes it possible to react to events in the Editor and Administrator system.

That extension is represented in the dLibra software as a Java-language programming interface `pl.psnc.dlibra.app.extension.eventListener.EventListener`..

The interface has one method – called eventPushed – with the following parameters:

- event – representing an event which took place in the Editor and Administrator Application; that parameter is an implementation of interface pl.psnc.dlibra.app.extension.eventlistener.AppEvent, and it makes it possible to determine the identifier of the object to which the event partains and to identify the event type, which can be one of the values determined in calculation pl.psnc.dlibra.app.extension.eventlistener.AppEventType; and
- serverInterface – representing the dLibra server access interface, which can be used for referring to services which make it possible to retrieve data pertaining to objects in the dLibra system.

In the default configuration of the Editor and Administrator Application, there are no extensions of that type. The extension has been designed for tools which carry out specific activities in external systems and which, for obvious reasons, are not supported by the Editor and Administrator Application. A sample extension was created for programmers who want to create such a tool.

The **objectPanel** Extension

Extensions of that type make it possible to add an additional tab in the Editor Application, which will be visible in the lower part of the screen after an object of a particular type has been selected. A plugin of that kind can freely prepare the content of such a tab, and it can make use of access to server functions for that purpose. The extension can also be used for displaying additional information related to a selected object – for example, information retrieved from external systems.

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.objectpanel.ObjectPanel`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

By default, no extension of that type is installed in the Editor Application, but a sample implementation – which demonstrates the capabilities of the extension – can be downloaded.

The **dataSource** Extension

That extension type makes it possible to add new ways of defining publication content. In the creator for adding a new publication (as well as adding files to a publication and for replacing files in a publication), on the file selection page, the user can choose one of the installed **dataSource** extensions. Then, instead of the standard file selection panel, the panel prepared by that extension will be displayed. When the user has selected files for publication in that panel, the extension passes that information on to the Editor Application, and the user can go to next steps of the creator. The extension makes it possible to indicate not only files on the local drive but also resources on the Internet, which will be downloaded by the dLibra server and saved in its repository (the files can also be downloaded with the use of the editor's computer).

The extension has one parameter, class, which specifies the name of the class which implements the programming interface of the extension. The programming interface (Java language) for that extension is `pl.psnc.dlibra.app.extension.datasource.DataSource`. For more information about that interface, see the comments on the methods and the interface (JavaDocs).

By default, no extension of that type is installed in the Editor Application, but a sample implementation – which demonstrates the capabilities of the extension – can be downloaded.