Table Des Matières

Getting started

Who should read this manual ?

This manual is destined for webmasters and companies who wantto use Automne 4 for web site projects. It will guide you in an intuitive manner through the features of Automne 4, from the simplest to the most complex. Whatever your role: editor, integrator, webmaster, administrator or even decision-maker....this manual will give you all the pertinent and practical information you need.

Organization of this manual

Each part is organized according to the following model:

1. Essentials
2. Properties
3. Possible actions

This three-part division allows you to have a gradual understanding of the features of Automne and to go directly to the most important features for you. You will thus quickly acquire a general knowledge of Automne and a deeper understanding of the points which concern you the most.

Prerequisites

To use Automne you must have the following technical knowledge:

- You must know the following Web standards: XHTML, CSS, XML.
- You must have have a dedicated or shared server to host Automne.
- Your server must accept PHP and MySQL and be correctly configured.

Web standards: XHTML, CSS, XML, W3C

XHTML :

XHTML is a language used to write pages for the World Wide Web (also called the Web). Originally conceived of as a successor to HTML, from which it takes a large part of its syntax, XHTML allows one to write web pages (also called web documents).

XHTML is more recent and more demanding than HTML, but it is also more adapted than the latter for everything displayed on devices other than computers (mobile phones, PDA, televisions....)

XHTML is used by Automne to create the pages it manages. The pages are created from XML models as XHTML respects the syntax of XML, which is not the case with HTML. It is thus possible to mix XHTML and XML tags.

> Note that XHTML and HTML evolve together (XHTML will soon go to version 2 and HTML to version 5). By design Automne can generate pages equally in either language.

CSS :

CSS allows one to separate the content of HTML or XHTML documents from their presentation.

This has a number of advantages:

- Improved access for handicapped persons,
- Changes in structure and presentation can be made more easily,
- Reduced complexity of document architecture,
- etc...

XML :

XML is a markup language whose primary goal is to manage text data.

Unlike HTML, XHTML and other languages constrained by their standards, XML can be expanded as long as some simple writing rules are respected.

In addition, its relative ease of use and versatility has made XML one of the most widely used languages over the last few years. It is for these reasons that Automne uses this language at the level of its rows and pages so that they can be managed easily by anyone.

The structure of XML accepted by Automne is described in detail in the interface of the software, as well as in this documentation.

W3C :

The World Wide Web Consortium, or W3C, is a standards organization.

It was founded in October 1994 as a consortium to promote the compatibility of Web technologies such as HTML, XHTML, XML, RDF, CSS, PNG, SVG and SOAP. It does not issue norms in the European sense, but makes valuable recommendations for the industry.

PHP, MySQL and Javascript

PHP

PHP is a language used especially for the production of dynamic web pages, that is to say pages whose content is updated every time they are viewed.

Automne is based upon PHP for several reasons:

- it is very popular and supported by the majority of web servers,
- it is conceived for the web and is very adapted to its needs,
- it is accessible without sacrificing advanced capabilities.

It allows Automne to benefit from an important community and permits you to easily make evolutions. If you are not interested in PHP and prefer to concentrate on content, no problem. With Automne you can forget the language even exists.

It is necessary, however, to know how to configure PHP on your web server, at least for the installation of the application.

Automne requires **at least PHP version 5.2.0**. You must also remember to correctly configure the operating options of PHP.

For this you must modify the following guidelines in your file php.ini :

memory_limit = 64M
magic_quotes_gpc = Off
magic_quotes_runtime = Off
magic_quotes_sybase = Off
safe_mode = Off
register_globals = Off

Depending on your host, these guidelines may only be accessible by modifying a file .htaccess at the root of your site. Consult your host or the FAQ, which lists a number of hosts and the guidelines to apply. Do not hesitate to ask questions in the forum if you encounter difficulties either in installing Automne or setting the parameters correctly.

Using PHP CLI :

Automne may require using PHP CLI to execute background scripts on the server. As explained in the FAQ, CLI is not indispensable but strongly recommended for sites with many users or large volumes of content.

On a shared server, CLI will rarely be available. On a dedicated server, however, it is up to you to install it. (See the PHP site for more information). In order for Automne to use CLI on your system you must:

- On a Unix/Linux server, ensure that CLI can be found on the user PATH used by Apache.
- On a Windows server, specify its location during installation. You can next modify this location by modifying the constant PATH_PHP_CLI_WINDOWS in the file /config.php in Automne.
- On a Unix server (Linux, Solaris, Mac OS X, etc.), you can also specify this location if CLI is not detected automatically during installation (if it is not found in the PATH of your server). You can specify this location

by modifying the constant PATH_PHP_CLI_UNIX in the file /config.php of Automne.

For all modifications of the file /config.php d'Automne, refer to the file /cms_rc.php in order to know the constants employed and the possible values.

MySQL

MySQL is a very popular relational database management system (RDBMS) for the creation of internet sites. It allows one to store, manipulate and restore the data you entrust to it.

Automne has chosen MySQL for the data management because it is free and open source and like PHP assures the durability of your site thanks to its versatility and its community.

No knowledge is MySQL is required to use Automne, which will take care of storing and retrieving your data for you in a totally transparent fashion.

Automne requires **at least MySQL version 5.0**. Next, you will have to get the necessary information about the connection to the database you will be asked for during the installation of Automne. This information is usually provided by your host:

- Server of the database (host),
- Name of the database,
- User(s) accessing the database,
- Connection password for the database.

Javascript

Javascript is a programming language integrated in web pages and executed by the navigator. It is complementary to PHP, which only executes itself on the server. In addition to allowing the manipulation of pages after they load, it authorizes updates without reloading the page.

This language is today widely used to create a more agreeable experience for the user. For example, our menu on the left uses it for ergonomic reasons.

Automne uses the most precise javascript technologies to offer an efficient and agreeable interface with the help of the library ExtJs or tools such as Blackbirdand and Pretty Print to aid with the conception of complex sites. You can next add any javascript library you want to make the sites created with Automne more interactive.

Dedicated server / shared server

A server is a computer upon which software called a web server is used to deliver (serve) the content of your website and web pages to web users.

A server is identified by an IP address. This is the equivalent of the GPS coordinates of a house. For example: 192.168.125.12. Although it is possible to consult a web site by typing the IP address of its server, it is generally preferable to use a domaine name we associate with an IP address. This is the equivalent of the mailing address of a house. For example: mysite.com.

Automne requires the use of the **Apache web server** in order to function. There are no restrictions regarding the version of your Apache web server but versions 1.3x are the fastest and use less resources on your server.

In order to configure your Apache server there are only two prerequisites:

- Automne must be installed in the root directory of your web server and not in a subdirectory.
- Apache must allow the use of .htaccess files to manage its configuration.

  If you want to install several instances of Automne on the same server, use the virtualhost of Apache to create as many virtual hosts as you want on your server.

Putting a site on a server to make it available on the internet is called hosting. There are two principal types of hosting: a dedicated server or a shared server.

Dedicated server

A dedicated server is a machine whose resources serve your site alone. It is as if your site lived in a house; it would have all the rooms to itself and to use at its leisure. This solution allows sites to be quick no matter what their size or complexity.

Shared server

A shared server is a machine whose resources are divided between several users. It is as if your site lived in a hotel. Like hotels, shared servers range from little hotels with toilets and showers down the hall to five star hotels with air-conditioned rooms and satellite television.

Automne only needs a server which supports PHP5 and MySQL (more information on the page PHP / MySQL) and it will be discrete, quick and easy to use even on a "no-frills" shared server.

Many providers offer dedicated or shared hosting. In the case of shared hosting, you must make sure the service meets the prerequisites for Automne.

Some helpful development tools

Graphics

You will undoubtedly need tools allowing you to modify or create images to improve your design. If you do not have a favorite tool, this list of our favorites may be useful.

Windows

Paint.net (freeware): Contrary to the software supplied with Windows, paint.net is equipped with many features allowing for the easy editing of images and applying more complex effects.

Photofiltre (free/paid): There are two versions of photofiltre, one free, but old and not updated; the other must be paid for. The free version allows you to make the most classic modifications (cropping, color, noise....); the paid version adds advanced features such as layer management. It should be noted that the free version is not authorized for commercial use.

Fireworks (paid): A part of the Adobe Creative Suite (in the same family as Dreamweaver and Photoshop, it is a powerful tool conceived for the production of images for the web. It allows very precise work for your web images.

Photoshop (paid): Well-known among graphic artists, Photoshop is very powerful software originally created for the modification and manipulation of photographs. Little by little it has become used for the creation of images for websites even though this tool is difficult to master and expensive. You must therefore develop a strategy to make it worth the cost.

Multiplatform

Gimp (freeware): A free alternative to Photoshop, Gimp is the most popular image editor for Linux. It is therefore furnished with the majority of distributions. It is likewise available for Windows and MacOS, among others.

Pixel image editor (paid): Also known as Pixel32, Pixel image editor is software for retouching images and is inspired by Photoshop. It is available for Windows and MacOS, among others.

Development

There are a number of development tools for PHP/javascript or to edit SQL requests and no matter whether it is for Windows, Linux or MacOS, there are many to choose from without passing through the pain of repeated tests. For this reason we list our favorites below .

Windows

Jedit: A relatively little-known editor available for Windows, Mac and Linux which can manage a large number of formats. In addition it has a plug-in system to expand its capabilities.

Scite: A very light editor which does not need installation. Its is very "no frills" and recommended for users looking for a lightweight and simple editor.

Pspad : Pspad is a very complete editor that will please any developer.

Linux

Jedit : A relatively little-known editor available for Windows, Mac and Linux which can manage a large number of formats. In addition is has a plug-in system to expand its capabilities.

VIM : A modal editor derived from VI while adding a friendlier interface. Its power and relative simplicity make it an editor of choice for Linux users.

Gedit/Kedit... Most Linux distributions are furnished with text editors that are sufficient for basic development.

Once your site is developed you will no doubt want to be sure your visitors see it as it should be; for this you will have to test it with different navigators and undoubtedly pass through a tweaking phase. The tools listed here will help you during this phase.

> These lists of software are not exhaustive and are only supplied for your information. WS Interactive has no link with the publishers of the software cited.

The most common navigators are Internet Explorer, Firefox, Safari, Chrome.

For Firefox, Safari and Chrome a test of the latest version will usually be sufficient For Internet Explorer you will have to (ideally) test versions 6, 7 and 8.

For this you can use tools such as multiple IE and Ietester or install several virtual machines, each with a version of Internet Explorer, taking into account that this will require at least three Windows licenses.

There are also services to test sites online such as Adobe browser lab.

> The administration of Automne is not compatible with Internet Explorer 6 for reasons described in the FAQ. This does not stop you from using Automne to create sites compatible with this version of the navigator.

Automne Concepts

Automne is based upon several distinct concepts that allow easier creation and administration of your internet site. These different concepts cover the following themes:

- presentation (Page models, Content rows and Client area) ;
- collection and treatment (Content rows and Polymod) ;
- administration and rights (Validation and publication, user rights).

Page models

A fundamental principle of CMS is the separation between content and presentation. In other words the graphics and the information contained in the site are totally independent from one another.

The page models thus serve to define a common thread for your site's pages most commonly grouped together: the banner, the navigation menu, the footer and the background of your site.

In most cases your site will be constituted of two or three page models:

- A model for the homepage,
- A model for the internal pages,
- And sometimes a model for the form pages.

When you create each page of your site you must select the page model that it should use. You can at any time and in a few clicks select another model for the page in question.

> Generally, for sites which need to take into account many languages, we will have models for each of these languages. This allows us to personalize the skeleton of the page according to the language.
>
> For example, we have the following models:
> - Welcome FR
> - Welcome EN
> - Interior FR
> - Interior EN

This principle assures a homogeneous presentation of all the pages of the site.

As each page is not identical in its content, the page models will contain information concerning the modifiable zones of the site, also called client spaces.

It is thanks to these client spaces that you can insert, in a simple and intuitive manner, the content of each page with the help of content rows.

Model with 1 client space:

By choosing ranges the user can add any kind of content in the dedicated zone:

Model with 2 client spaces:

By choosing ranges the user can add any kind of content in the dedicated zones:

At any moment the user can switch their page from the first model to the second, which will modify the graphic presentation of the page but preserve the content.

Attention must always be paid to the compatibility of models:
To guarantee the preservation of data in switching models, the two models must have similar client spaces.
See Client spaces and Page properties**.**

Code XML :

The following code presents a very simple model.

We find the doctype, the html tags, head and body, in this case only one standard client module. This last tag allows adding content to the page.

One should also note the tag <atm-meta-tags /> which is obligatory in all models. When regenerating the page, this tage is automatically replaced by a code indispensable for Automne to function.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
<head>
    <title><atm-constant name="APPLICATION_LABEL" /> : <atm-title /></title>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <atm-meta-tags />
</head>
<body>
<div id="page">
    <atm-clientspace module="standard" id="first" />
</div>
</body>
</html>
```

- Get more information on page models.

Client spaces

Définition

Examples

Example of a page in edit mode with a model having two client spaces (outlined in green):

Example of a page in edit mode. In this client space several rows are arranged. It is possible to add others, then delete them or modify their content:

Things to remember

A page has 2 distinct parts:

The form:

A page uses a "page model" (this is a kind of "skeleton" for the page), which in turn contains one or several "client spaces".

Each client space permits one or many "content rows".

The content:

The content is prepared by the site editor using the "content rows".

You simply edit and enter the desired content (text, images, etc...).

How it works

The tag <atm-clientspace />

In a page model, client spaces are defined by the tag <atm-clientspace />.

Note that it a self-closing tag with 2 attributes:

- **module:** This is the code name of the module. Generally a "standard" module is used; it is the principal module of Automne: page management. It is possible, however, to indicate all other modules.
- **id :** This is the identifier of the client space within the page model. It must be unique. When you insert content rows in this client space, Automne associates the rows with this unique identifier.

This gives us, for example: **<atm-clientspace module="standard" id="first" />**

We advise you to name your client spaces according to their priority in the model:

- "**first**" for the most important client space
- "**second**" for the second client space
- "**third**" for the third
- ... and so on...

The principle reason for this is that in order to switch one page model for another it is necessary for the models to be compatible. That is to say they have client spaces whose identifiers (id) correspond. If a page is associated with a page model that has 2 client spaces and we switch to a model possessing only 1, a loss of data from the second client space will result. Thus, by naming your client spaces, you can be sure that the data belonging to the most important client space is conserved in case you switch the model.

Regeneration of a page

Technically, when a page is regenerated, Automne will look for the rows associated with this page and client space and then replace the tag with the corresponding content.

Here is an exampe of a "client space" tag within the model:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
<head>
   <title><atm-constant name="APPLICATION_LABEL" /> : <atm-title /></title>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
   <atm-meta-tags />
</head>
<body>
<div id="page">
   <div id="rightBox">
      <atm-clientspace module="standard" id="first" />
```

```
    </div>
    <div id="leftBox">
      <atm-clientspace module="standard" id="second" />
    </div>
</div>
</body>
</html>
```

Concrete example

Module : pnews (created with the module generator polymod)

Tag : <atm-clientspace module="pnews" id="homepagenews" type="homepage" />

From this tag Automne will call the corresponding module by the codename "pnews".

It is a polymod module having the classes and features necessary to handle this tag.

The tag is interpreted by the polymod module in the following manner: the module is configured to understand this tag and returns the content to the following file:

/automne/templates/mod_[code du module]_[attribut "type" du tag clientspace].php

In our example, Automne will automatically insert, in place of the tag, the contents of the file/automne/templates/mod_pnews_homepage.php.

> This example uses the polymod module, the module generator behind Automne. The behavior regarding the clientspace tag is already configured.
>
> It is therefore possible to experience different behavior with another module.

- Get more information about Client Spaces

Content rows

The content rows are made up of one or many content blocks that you place in your client spaces. The content rows allow for input and organization of your site.

These blocks can contain many types of information:

- Title,
- Text,
- Image,
- Documents,
- Flash Animation,
- Module.

There are several default rows in Automne 4 (text, image, title h1, title h2, Google map...) You can also create your own content rows to enrich your site with several other types of information.

All content rows are independent from one another. There are no restrictions on the number of rows or their placement on the site. **The content rows are usable everywhere and at any moment in the client spaces of your page models.**

Thanks to the content rows, you have total freedom in the organization and content of your site, as well as an ease of personalization for your pages; all this without technical constraints.

Examples

By default Automne proposes predefined rows. It is therefore practical, even indispensable, to create your own rows. Here are some examples of rows and their content blocks:

A row "Title and image" may be composed of blocks:

- Title (chain of 255 characters)
- Image (Image file whose maximum size, thumbnail and caption are configurable)

A row "Title, subtitle and text" may be composed of blocks:

- Title (chain of 255 characters)
- Subtitle (chain of 255 characters)
- Text (text HTML, with wysiwyg toolbar for formatting)

Good to know:

For each of the blocks of a row, in most cases the user will be able to enter the desired content.

If the user does not give information about a data block, this will not be displayed in production. In a row "Title, subtitle and text" it is thus possible to give information only about the title and the text without indicating the subtitle.

Instead of leaving it open to the editor to insert content, it is also possible to automatically insert dynamic content from a module, or to put PHP code directly in the row. The features offered are thus very flexible.

Remember that dynamic help is available while you edit a row model.

- Get more information on content rows

The Polymod

The Polymod (for Polymorphic module) is a module generator. It is a powerful tool used to create and manipulate data modules without having to manage (and know) SQL and PHP.

With Polymod it is possible, for example, to manage the input and storage of articles with title, content publication date and category, thus allowing you to integrate a Blog on your site.

The Polymod takes charge of the following elements that you can combine as you want in order to achieve exactly what you need to express yourself :

- Short text
- Long text (formatted or not)
- Number (integer or floating)
- Image
- File
- Category
- Link
- Date
- Language
- User or user group
- Boolean (Yes - No)
- etc.

This list is evolving and, if necessary for your project, you can even create your own kinds of fields.

It is very important to understand that Polymod cannot be used to create display (display being managed by content rows) and is focused exclusively on storing data.

The separation of content and display allows you to concentrate and deal with each one independently; this separation allows you to save time and offer an uncompromised experience to your users.

The Polymod is integrated by default at the core of Automne.

- Get more information about the Polymod

The principle of validation and publication

In the case of a site where many users need to publish content, it is interesting (and sometimes essential) to establish a verification policy before making the site available to visitors.

These validations assure that the information furnished by users is relevant and worthwhile.

The concept of validation is included in Automne; indeed, it is at its core. Thus, all content destined to be displayed to the visitor (text, image, video...) must go through a validation stage.

We therefore find the following schema
Creation / Modification of content » Validation » Display to visitors.

This schema allows you, for example, to verify texts before publication (publication is done to make content visible to the visitor); in the case of comments upon an article, you can assure that the tone of the comments is appropriate for your site.

In addition to the necessity of validating content to be displayed to visitors, it is commonly necessary to add an idea of the timeliness of the information. For example, a page promoting a concert or selling tickets to such an event is of no interest once the date of the event has passed. Automne allows you to manage publication periods, permitting you to easily manage the periods during which an object (page, image, video....) is visible to visitors.

Of course the principle of validation leads to another principle, that of user level (without which every user would be able to validate).

- Get more information about the principle of validation and publication

User rights

Users

A site is often the fruit of many peoples' labor. Some of these people are concerned with the text, others the images, still others with the development of unique features. All of these people have particular needs and expectations; to meet them it is necessary to make distinctions among them and this is where the idea of users comes in.

The users are the people who need to interact with the site. Typically there is an administrator--who manages the site, validates what is displayed, adds or deletes pages--and the editors, who create relevant content for visitors.

Each user has a unique ID and password. Once identified, Automne modifies itself in order to give the user access to all the information and only the information they need.

User groups

Let us take the example of our editors; if you have five editors it is obvious that configuring each account to allow the same options is a pure loss of time.

It is for this reason that it is possible to create user groups. A user group defines the role of users that are included and allows you to make adjustments for all of them at once. The rights of our editors can be configured in one place.

Rights

As mentioned above, Automne adapts itself to each user in order to give them access only to information they need. This selection is based upon the principle of "rights".

Imagine that the editors of your website edit the announces for concerts in your venue. On one part of the site one can reserve tickets thanks to member information such as an email address and password. There is no reason for the editors to have access to member information, is there? With Automne you can do everything necessary to assure that access rights for member information are not given to this group of editors.

Summarizing the rights and user accounts is a simple and powerful way to create different and optimized access levels to the information on your site.

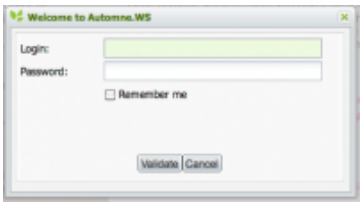
- Get more information about user rights

Using the interface

Connect to the administration platform using the URL given by your administrator, generally: www.yourdomain.com/automne/admin.

Enter your identification user name and passord, and validate

- **Identification:** root
- **Password :** automne



> Using the option "Remember me", can allow anyone who has access to your machine to use Automne under your identification. This option lasts 15 days maximum, after which you must reconnect with your id codes.

For obvious reasons, it is highly recommended to change the user name's password from the first log on.
This account is the "super administrator" who has the highest rights.

Once connected, you have access to the interface of Automne.
According to the rights you have, you will see all or part of the elements described in the following pages.

Composition

The interface is made of 3 parts (see the screen shots):

1- The central zone

Your site is visible within this zone. You can surf freely, following the intern links. It is also your working space regarding the modification and the creation of pages.

2- The bar tab

This bar is your tool bar. It contains, among other things, the availablesactions for the current page visible in the central zone.

3- The control panel

It permits access to modules or the tools specific to the administration of the site.

The tabs

Search 

Once validated, the results of the search will appear in a window. It will be possible to filter the results according to your access viewing rights.

The search can be done in several ways:

- entering an existing page number, the page concerned will be displayed in the central zone,
- entering a keyword, you can search by your elements' titles.

  Several tags allow you to generate a particular type of search :

  template:id search the pages using the model whose login is *id*

  row:id search the pages using the rows whose login is *id*

  linkFrom:id search the pages linked by the page whose login is *id*

  linkTo:*id* search the pages relating the page whose login is *id*

  user:id search the user whose loggin is *id*

  group:id search the users whose group has *id as login*

Site plan 

Your site is structured according to a tree stucture. **The page 1 is your site's root**.
Each page depends on this root and the body of pages builds your site. To explore your site through the tree structure and reach a particuliar page, use + to open the site map.

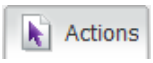
Your pages appear according to the following structure :

- An icon, representing the page's status of validation,
- The title,
- The number in brackets. This number is the single login of the page.

Add to favorites 

Clicking on this button, you put the page into your favorites.
This can be useful if you regularly visit this page. Your favorites are accessible from the control panel under the tab "Editing pages".

When a page is in your favorites, the background of the button changes, and you see orange slashes.  . To remove a page from your favorites, you just have to click back on this button.

Actions 

According the status of the page, you have access to several specific actions for the page currently visible in the central zone.

- **Move the page :** open the tree structure in move mode.You will see a little move icon at the right of each page. You can move the page by dragging and dropping.

  To have the right to move a group of pages, you need to have the right to produce pages (administration tab of the rights administration).

- **Copy the page:** open the tree structure. You will have to select the page you want to copy. The copy keeps the original structure and content.
- **Regenerate the page:** causes page regeneration.
- **Cancel the content modification:** cancel the draft of a page.

If another user has already started a draft, all modifications will be lost.

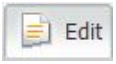
- **Submit the modifications to validation:** if there is a current draft, it permits submitting it to validation. It also cancels the draft.
- **Release the page :** allows the release of the page.

Create 

Allows creating a page under the page currently visible in the central zone.

Properties 

This window contains all the properties of the page currently visible in the central zone. Property of the page.

Edit 

Allows modifying the content of the page and the page setting of the rows.

Preview 

This tab is only active when the page has been modified from its online version and the modifications have been submitted for validation. It is the version of the page to be put online.

Right tab

This tab bears the name and status of the current page and allows you to see the page as the visitors of the site see it.

By rolling over a tab, a short description of its purpose will appear.

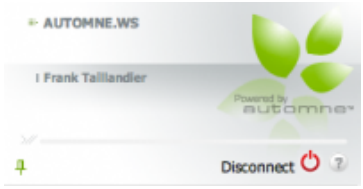
The control panel

This panel allows access to general tools for the whole site. To make it appear put the cursor into the right zone of the screen.

It closes automatically after a few seconds. To make it stay open, click on the pin , so that the panel will remain open.

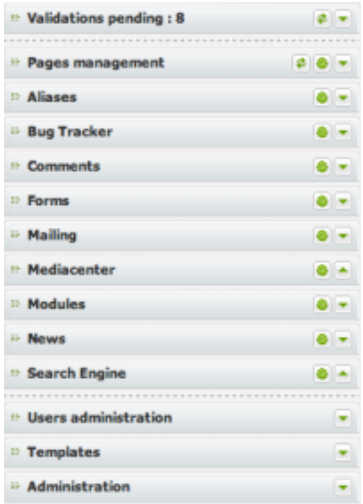
General information:

- The name of your site: you can change it in Automne's parameters, under the adminstration button of the control panel,
- Your name,
- The Automne logo: by clicking on it, you open your site in a new window,
- A link to the Automne site,
- The bar representing the scripts in progress,
- A button to logout
- A button "About Automne"



Administration's elements of Automne 4 :

- Validation in hold,
- Access to module administration,
- Access to administration tools.



Each administration element of Automne is visible according to the rights you have.

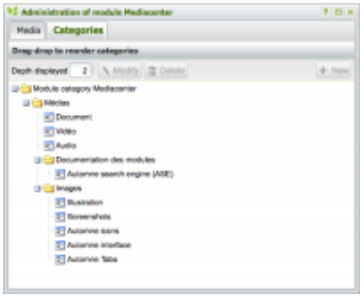
The windows

Beyond tabs and central panel, your site is visible through the interface's central part.

All the actions made in the Automne's interface are set in floating windows offering several characteristics:

- You can remove them,
- You can enlarge most of them (a double click on the top menu of the window or a click on the 'enlarge' button in the upper right opens it in full screen).
- You can close the window by clicking on the button 'close' in the upper right of the window,
- You can close a window only if all the windows depending on it are closed,
- Hovering over the « ? » up right, you will find pop-up help concerning the opened window's features
- Most of the windows offer, at top, a menu for searching and filtering, permitting quick acces to the data you are looking for.

Example of a modal window in Automne's administration interface :



Attention: when you open a window, the background turns grey and it' becomes impossible to click on the elements of the background (apart from the windows already opened).

To access to the background, you just have to close the windows currently opened.

Page creation

The pages on your site are all structured in the same fashion.

- They are first composed of a page template which defines the layout of your page.
- Next, the template can have one or many client areas in which you can insert content. The definition of client areas is made during the creation of page templates.
- Finally each client space can contain content rows which are modular structures that allow the addition of different types of content, text, images, flash, etc...

Section corresponding to page design in the Automne interface.

Page models

Presentation

A fundamental principle of CMS is the separation between content and presentation. In other words the graphics and the information contained in the site are totally independent from one another.

The page models thus serve to define a common thread for your site's pages most commonly grouped together: the banner, the navigation menu, the footer and the background of your site.

In most cases your site will be constituted of two or three page models:

- A model for the homepage,
- A model for the internal pages,
- And sometimes a model for the form pages.

When you create each page of your site you must select the page model that it should use. You can at any time and in a few clicks select another model for the page in question.

This principle assures a homogeneous presentation of all the pages of the site.

As each page is not identical in its content, the page models will contain information concerning the modifiable zones of the site, also called client spaces.

It is thanks to these client spaces that you can insert, in a simple and intuitive manner, the content of each page with the help of content rows.

Creating a model:

A page model is made up of properties which make it possible to classify it in relation to Automne's other page models and to define its use in Automne.

It also has an XML definition that contains the XML code necessary for the pages that use the model to function correctly.

Creating a page model

When you create a page model, you can specify some properties:

Label:

This is the title of the model.

The label must allow you to see if the page model is model of a welcome page or an interior page, etc. It must allow you to identify the model as quickly and as simply as possible.
It is used in particular to choose the model when a new page is created.

Example : "Interior - EN"

Description :

Explicitly describes the meaning of the label and the use of the model.

Example: "Page model for interior pages of the English site."

Group:

Allows you to classify the models according to certain characteristics.

The groups are used to search for and organize the models, but also to affect user rights. Example: one can prohibit an editor from using models from the "EN" group.

Example: a page model can be classified in the "EN" category so that it corresponds to the page model of the English site.

New groups:

Allows you to create a new classification group.

Example: It is possible to create an "ES" group that corresponds to the page model in Spanish.

Sites:

Allows you to define which site(s) can be used. Example: One can choose to restrict the use of a model to the Spanish site alone.

Thumbnail:

Graphic visualization of a page model in order to see it before applying it to pages.

It allows you to have a view of the models when you create a new page.

Definition XML (tags): Correspond to the XML structure of a model:

The procedure to follow in order to insert a model is as follows:

1. Prepare the XHTML model.
2. Transform the links, meta-data and the content zones.

1. Prepare the XHTML model:

First of all, it is useful to prepare an XHTML model from the images supplied. This XHTML model must be able to function on all platforms and navigators (according to their specifications, the versions and/or platforms may be restricted.)

Attention! This model must be in XHTML format, that is to say compatible with XML. In effect, a parser will go over the model not for validation, but to test that the file is well-written (the tags must be closed, the attribution values placed in quotation marks, the entities must be declared, etc.)
The best way to test the form of an XHTML model is to try and import it as a model into the administration of the application.

2. Transform the model:

- See more detail about the XML definition **of models.**

Default content rows:

The default content rows are the rows which will be inserted automatically into your pages when you create a new page with the current model.

Use the rows frequently employed in the pages using your model.

You can:

- Add a content row.
- Move a content row.
- Delete a content row.

Print:

Allows you to choose which content zones (client spaces) to print, once the printing is activated. (You can set these parameters in Automne Parameters).

You can next make a link to a print page with the tag <atm-print-link />.

> If the module "Search engine" (ASE) is installed, it is necessary to activate printing (You can set these parameters in Automne Parameters) and to define the printable client spaces.
> In effect, to index the "Page management" module, the search engine only indexes pages having printable content.

XML Definition

The XML definition of a page model is constituted in the following manner:

- **XHTML code** will be included in the pages of the site using this model.
- **XML code** defines the logical areas in which Autmone will work.

There are three principle types of XML tags in the page models:

- atm-linx : creates navigation links between Atomne pages.
- atm-clientspace : defines content zones.
- Other tags : recover information about the page (title, metadata, etc.)

A model can contain the following XML tags:

Page title:

<atm-title /> Self-closing non-attributed tag: It will be replaced by the page title field.

Page metadata:

<atm-meta-tags /> Self-closing non-attributed tag: It will be replaced by the metadata fields.

This tag is obligatory, it must be located in the tag <head> of your XHTML model because it manages certain functions of Automne. It is this tag, for example, that activates the page editing functions. It adds javascripts to the style sheets furnished by the different modules which will be employed in these pages.

This tag generates the following metadata:

Name	Metadata	Possible Values
Description	<meta name="description" content="" />	Open
Keywords	<meta name="keywords" content="" />	Open
Category	<meta name="category" content="" />	Open
Robots	<meta name="robots" content="" />	(all, index, follow, noindex, nofollow)
Language	<meta name="language" content="" />	Open
Author	<meta name="author" content="" />	Open
Copyright	<meta name="copyright" content="" />	Open
Navigator cache	<meta http-equiv="pragma" content="no-cache" />	Applied only if this option is checked
Generator	<meta name="generator"	Automatic

	content="Automne" />	

Call scripts tag <atm-js-tags />

<atm-js-tags files="/js/js1.js,/js/js2.js" />

This tag concatenates several javascript files in one, thus accelerating the loading of your web pages.

The files listed this way will be concatenated and compressed before being served to the internaut. An advanced management of the navigator cache is employed.

- **Files** attribute: Javascript files are included in the page (separated by commas).

The files to be used must be located in the directory or sub-directory of /js/.

Example :

<atm-js-tags files="/js/jquery.js,/js/common.js" />

> In development, as long as the "activate system debugger" parameter is active, Javascript minimization is not performed; thus it is possible to run the Javascript code more easily.
>
> On the contrary, in production, as long as the "activate system debugger" parameter is inactive, minimization results in btter performance.

Call style sheets tag <atm-css-tags />

<atm-css-tags files="/css/css1.css,/css/css2.css" />

This tag concantenates several file sheets files into one, thus accelerating the loading of your web pages.

The files listed this way will be concatenated and compressed before being served to the internaut. An advanced management of the navigator cache is employed.

- **Files** attribute: CSS style sheets are included in the page (separated by commas).
- **Media** attribute (optional): specifies the media output employed, among them:
  - all (by default if the media attribute is not employed)
  - aural
  - braille
  - embossed
  - handheld
  - print
  - projection
  - screen
  - tty
  - tv

The files to be used must be located in the directory or sub-directory of /js/.

Example :

<atm-css-tags files="/css/common.css,/css/accueil.css" />

Page client spaces:

<atm-clientspace id="identifiant" module="codename" /> Self-closing tag. It has the following attributes:

- **id**: unique identifier of the model's client space. You can put whatever alphanumeric value you want as long as it is unique among all the atm-clientspace tags of the model.
- **module**: defines which module will supply the content of the client space. By default, if you wish to be able to insert content rows in this client space, you must specify a "standard" value.

This tag defines the position of a client space in the page. It is this tag that makes it possible to insert content rows in the pages which will be created from this model.

Example : <atm-clientspace id="left-column" module="standard" />

PHP constants :

<atm-constant name="constant" /> Self-closing tag. It has the following attributes:

- **name**: name of the constant that must be returned.

This tag allows you to obtain the value of every PHP constant.

Example : <atm-constant name="APPLICATION_LABEL" /> This code will return the name of the constant APPLICATION_LABEL which in Automne is the name of your instance of Automne.

Link to print page:

<atm-print-link keeprequest="true"> ... {{href}} ... </atm-print-link> This tag has the following attributes:

- **keeprequest**: this attribute is optional and sends the GET and POST values of the current page to the print page. Accepted values: **true** or **false** (default).

This tag will replace the following special value:

- **{{href}}**: address of the print page for the current page.

This tag creates a link towards the print page (if it exists) for the current page.

Example : <atm-print-link>Imprimer</atm-print-link>

Date of the latest page update:

<atm-last-update format="m/d/Y H:i:s"> ... {{date}} ... {{firstname}} ... {{lastname}} ... </atm-last-update>This tag has the following attributes:

- **format**: Format of the date display according to date format of PHP.

This tag will replace the following special values:

- **{{date}}** : Date of the latest page update in the format specified by the format attribute.
- **{{firstname}}**: First name of the user who made the last page modification.
- **{{lastname}}**: Last name of the user who made the last page modification.

This tag displays the date or the author of the latest update of the current page.

Exemple :<atm-last-update format="m/d/Y H:i:s"><small>Dernière mise à jour : {{date}} par {{firstname}} {{lastname}}</small></atm-last-update>

Identifier of the current page: 339

`{{pageID}}`

This value will be replaced by the identifier of the page you are on.

PageID is very useful for finding the identifier of the page you are on.

A simple example of an XML model code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
   <title><atm-constant name="APPLICATION_LABEL" /> : <atm-title /></title>
   <atm-css-tags files="/css/common.css,/css/index.css" media="screen" />
   <atm-css-tags files="/css/print.css" media="print" />
   <atm-meta-tags />
   <atm-js-tags files="/js/sifr.js,/js/common.js,/js/jquery-1.3.2.js,/js/menu.js" />
</head>
<body>
   <h1><atm-title /></h1>

   <div>
      <atm-clientspace module="standard" id="center" />
   </div>
   <br /><br />

   <atm-print-link keeprequest="true"> <a href="{{href}}" target="_blank">Print</a></atm-print-link><br />

   <atm-last-update format="m/d/Y H:i:s"><small>Last update : {{date}} by {{firstname}}
{{lastname}}</small></atm-last-update>
</body>
</html>
```

This simple example gives us a model that could be employed to create pages in which it is possible to insert content rows.

On the other hand it lacks a system of links allowing navigation between the different pages thus created.

- These links are created with the tags atm-linx.

Managing navigation links: atm-linx tags

The principle of links between Automne pages

The pages of sites managed by Automne create a tree. Like all websites it is possible to create links between the different pages in this tree.

Automne allows you to define rules for automatically creating links between pages according to their position in the tree.

You may, for example, want a given area of a page to include links to all the sub pages of the current page. You may also want to have the path (the Ariadne's thread) between any two given points in the tree, such as between the welcome page and the current page in order to to create a navigation history.

Perhaps you would simply like to have a link to a page: for example the welcome page or any other page, no matter where it is in the tree.

These are merely a few examples of the possiblities offered for creating links between different pages in the tree.

Another essential point is the maintenance of these links over time. Automne dynamically manages the different links found on Automne pages. For example, if a page is deleted, all the links pointing to this page will be automatically deleted on all the pages of the tree. This eliminates the problem of "broken links".

In the same way, if an area displays links to all the sub pages of a given page and if a new page is created in its place, it will automatically be added to the existing list of links.

To do this, you must employ XML markers in your page models or in your content rows specifically describing the type of link you want to make. These markers are the atm-linx tags.

The atm-linx tags create 'internal' links on the site. These links generate navigation links from one page to another. **They will follow the tree schema you define for your site.**

Format of atm-linx tags:

Notation convention: in the following source codes, the code between brackets [] is optional.

```
<atm-linx type="linxtype">
   <selection>...</selection>
   [<selection>...</selection>]
   <display>...</display>
   [<display>...</display>]
   [<noselection>...</noselection>]
</atm-linx>
```

Linxtype where the value of the attribute type may be one of the following:

- **"direct":** direct link to one or more pages.
- **"sublinks":** link to sub-pages of a page.

- **"desclinks":** "descendant" links from the root to the current page. Parameters permit fixing the depart and arrival pages.
- **"recursivelinks" :** "recursive" links display a complete page tree.

The content of an atm-linx tag is separated into three distinct zones:

- The **<selection>** tag(s) select the target pages of the links to be created.
- The **<display>** tag(s) specify how to display the targeted pages.
- The optional **<noselection>** tag specifies a display in the case where no pages were found by the **<selection>** tag(s).

The sub-links and decendant links do not go outside of websites. On the other hand, direct links do not have this restriction.

1. Format of 'selection' tags.

The **selection** tag defines the target pages of the link. It can contain a declaration of conditions, in the case where target pages must meet certain conditions before being selected.

The format of 'selection' tags:

```
<selection [noerror="1"] [crosswebsite="1"]>
   [<condition>...</condition>]
   <start>'nodespec declaration'[, 'nodespec declaration' ...]</start>
   [<start>'nodespec declaration'[, 'nodespec declaration' ...]</start>]
   [<stop>'nodespec declaration'</stop>]
</selection>
```

Selecting target pages has three steps:

- The departure point(s), represented by a **start** tag.
- The destination target in the form of a path description (**"desclinks"** link), represented by a **stop** tag.
- Some conditions (optional) represented by the **condition** tag.

The optional attributes of the **selection** tag:

- **noerror**: Eliminates error messages when a non-existant page is created. Values accepted: *0, 1, true, false* (default).
- **crosswebsite**: By default, a link cannot target a page belonging to another site managed by Automne. This attribute can go beyond this limitation. Values accepted: *0, 1, true, false* (default).

This table describes the number of start and stop tags possible according to the type of atm-linx tag:

LINKTYPE	START	STOP
direct	1+	
sublinks	1	
desclinks	1	1
recursivelinks	1	

2. Format of 'nodespec declaration'.

The **nodespec** are the specifications of targeted pages. They can refer to a page by its reference or by its position in relation to the current page (the page where the link is found) or the root.

The format:

<nodespec type="NODESPECTYPE" value="NODESPECVALUE" [reloffset="OFFSET"] />

A table summarizing the possible values of attributes:

NODESPECTYPE	NODESPECVALUE	OFFSET
node	Identifer of the target page. Example: 23	N/A
relative	self: indicates the current page	N/A
	brother: indicates the page with same father as the current page	1: right brother -1: left brother
	father: un ancestor of the current page	1: father 2: grandfather
	root: the site root	0: root 1: the son of the root on the branch leading to the current page 2: grandson ...

ATTENTION: By "site root" we mean the root closest to the parent site!

3. Format of 'noselection declaration'

noselection declares an alernate display in the case where the **'selection**' tag does not target any page. It is optional.

The format:

<noselection>Valid XHTML code</noselection>

4. Format of 'display declaration'.

The **display** is the HTML specification which will replace the linx tag for all the targeted pages defined in the declaration selection. A display can have many conditions and there can be many displays in the atm-linx tag. In this case, for each target, one can look at the displays in their order of appearance in the tag; if the target meets all the conditions, this display will be adopted. If no display is sufficient, the target is rejected. If a display has no conditions, the default display is called up. It is useless to put many displays without conditions, as those following the first will be ignored.

The format:

```
<display>
   [<condition>...</condition>]
   <htmltemplate> Valid XHTM code </htmltemplate>
   [<subleveltemplate> Valid XHTML code </subleveltemplate>]
</display>
```

Tag <htmltemplate>: it has the XHTML code which generates the links for selected pages.

In the <htmltemplate> tag, the following special values can appear:

- **"{{href}}":** will be replaced by the absolute URL of the target.
- **"{{title}}":** will be replaced by the "link title" field of the target.
- **"{{jstitle}}":** same as the preceding tag, but the single and double quotes will be freed in order to include them in the javascript or the attributes of the XHTML tags.
- **"{{pagetitle}}":** Title of the page.
- **"{{jspagetitle}}":** Title of the page, but the single and double quotes will be freed in order to include them in the javascript or the attributes of the XHTML tags.
- **"{{desc}}":** Description of the page.
- **"{{id}}":** Displays the unique identifier of the target page.

- **{{currentClass}}:** Displays the CMS_current if the current target is the same as the current page. This information can be useful for post treatment of CSS or javascript.
- **id="{{currentID}}":** Displays the id="CMS_current" if the current target is the same as the current page. This information can be useful for post treatment of CSS or javascript.
- **"{{number}}":** Uniquely for sublink type links. Displays the number of account links. This information can be useful for post treatment of CSS or javascript.
- **"{{modulo}}":** Uniquely for sublink type links. Displays alternately 0 or 1 for each page. This information can be useful for post treatment of CSS or javascript.
- **{{isParent}}:** Displays the CMS_parent if the current target is a parent of the current page. This information can be useful for post treatment of CSS or javascript.
- **id="{{isParent}}":** Affiche id="CMS_parent" if the current target is a parent of the current page. This information can be useful for post treatment of CSS or javascript.
- **class="{{isParent}}":** Uniquely for sublink type links. Displays class="CMS_parent" if the current target is a parent of the current page. This information can be useful for post treatment of CSS or javascript.

The optional **<subleveltemplate>** tag defines an optional frame for the content of the htmltemplate tag for every type of link. It contains an XHTML code zone in which the **{{sublevel}}** value must be defined or the code produced by the **<htmltemplate>** placed.

Specificity of recursivelinks tags:

The syntax of the display tag for 'recursivelinks' type links has some nuances compared with other tags:

```
<display [mode="MODE" root="ROOT"]>
   [<condition>...</condition>]
   <htmltemplate> Valid XHTML code </htmltemplate>
   <subleveltemplate> Valid XHTML code </subleveltemplate>
   [<mode>MODE</mode>]
   [<root>ROOT</root>]
</display>
```

In addition to the special values of other tags, the <htmltemplate> tag has these supplementary values:

- **"{{lvlClass}}":** Uniquely for recursivelinks type links. Displays CMS_lvln. being the current level of recursivity (number). This information can be useful for post treatment of CSS or javascript..
- **"{{typeClass}}":** Uniquely for recursivelinks type links. Displays CMS_sub if the current target has sublevels, CMS_nosub if the target does not have sublevels, or CMS_open if the current target has sub levels and these levels are displayed (uniquely in close mode). This information can be useful for post treatment of CSS or javascript..
- **"{{sublevel}}":** Uniquely for recursivelinks type links. Position of the recursive zone. It is in this place the supplementary level of recursivity (depth) will be displayed if it exists.

The optional **<subleveltemplate>** tag defines an optional frame for the level of recursivity. It contains an XHTML code zone in which the **{{sublevel}}** value must be defined or the following level of recursivity (depth) must be placed. This HTML area can also contain the **{{lvlClass}}** value which will be replaced, in the same manner as the **<htmltemplate>** tag, by the current level of recursivity.

The optional **<mode>** tag defines the display mode of a **recursivelinks** tag. It is only useful for this kind of tag. **<mode>** accepts two values: **open** or **close**. A tag having the *open* value displays the entire tree. A *closed* tag will only display the tree area necessary to locate the current page..

> **From version 4.0.0rc2:** the **<mode>** can be replaced by a **mode** attribute on the **<display>** tag. This attribute also accepts the **open** or **close** values.
> It is also possible to add a tag or an **<root>** attribut which displays (or not) the first level of recursivity. This tag (or attribute) accepts the values 0 or 1 to activate or not the display of the first level of recursivity.

4. Format of 'condition declaration'.

The **condition** defines a rule to verify a target it it wants to meet a condition. It contains a value tag that will determine the value which will serve as a comparison with the other value of a tested target.

The format:

```
<condition property="PROPERTY" operator="OPERATOR">
   <value type="VALUETYPE" [property="VALUEPROPERTY"]>VALUEDATA</value>
</condition>
```

Where, in a simplified fashion:

- **PROPERTY:** can contain the following values: *rank, title, id, lvl, father*
- **OPERATOR:** can contain the following values: *==, !=, >, <, >=, <=, %2==*

  IMPORTANT: These characters must be coded in HTML to avoid any parsing problems (the character < becomes < the character > becomes &gt).

- **VALUETYPE:** can contain the following valuespeut contenir les valeurs suivantes : *data, nodeproperty*
- **VALUEPROPERTY:** can contain the following values: *rank, title, id*
- **VALUEDATA:** can contain the following values: valeur scalaire ou tag **nodespec**

Where, more detailed:

- **PROPERTY** is a property of the target which will serve for the test, or "rank" of the target (order number) in the target table. The properties allowed are restricted: title or id for the title or the reference of the target, "rank" for its rank (position relative to its brothers or sisters), "lvel" for its level of recursivity (depth) uniquely for recursivelinks.
- **OPERATOR** is a simple comparison operator: ("==", ">=", ...) or composed: "%2==". There is no limit to composed operators, except that they must be binary.

- **VALUETYPE** defines the type of value which will serve as the comparison. It can have these values: data or nodeproperty; the first will determine if the value between the starting tag and the end of the tag value is an unchanged scalar value. The second will determine if it is a value worth taking a nodespec.
- **VALUEPROPERTY** is optional, defining the properties that one must extract from nodespec that will follow in order to serve as a value.
- **VALUEDATA** defines the value (in the case where VALUETYPE="data") or the nodespec from which will be extracted the propoerty in order to define one of two values of the comparison.

5. Some examples of links created with atm-linx tags:

Direct link:

```
<atm-linx type="direct">
   <selection>
      <start><nodespec type="node" value="23" /></start>
   </selection>
   <display>
      <htmltemplate><a href="{{href}}">{{title}}</a></htmltemplate>
   </display>
</atm-linx>
```

This is the most simple tag; it will be replaced by a link to page 23 (if it exists); which will create the following XHTML code:

```
<a href="url_23">Page title</a>
```

Sublinks link:

```
<atm-linx type="sublinks">
   <selection>
      <start><nodespec type="relative" value="self" /></start>
   </selection>
   <display>
      <htmltemplate><li><a href="{{href}}">{{title}}</a></li></htmltemplate>
      <subleveltemplate><ul>{{sublevel}}</ul></subleveltemplate>
   </display>
</atm-linx>
```

Again, this tag is very simple, it will be replaced by a list of links to all the subpages (if there are any) of the current page This creates the following XHTML code:

```
<ul>
   <li><a href="url_fils1">Title subpage 1</a></li>
   <li><a href="url_fils2">Title subpage 2</a></li>
</ul>
```

Simple desclinks link:

```
<atm-linx type="desclinks">
   <selection>
      <start><nodespec type="relative" value="root" /></start>
      <stop><nodespec type="relative" value="self" /></stop>
   </selection>
   <display>
      <htmltemplate><a href="{{href}}">{{title}}</a></htmltemplate>
   </display>
</atm-linx>
```

Here is a simple example of a desclinks link. It creates a list of links going to the welcome page (root); for this the current page will create the following XHTML code:

```
<a href="url_home">Home page title</a><a href="url_page_bellow_home">Title of the page bellow the home page</a><a href="current_url">Current page title</a>
```

Complex desclinks link:

```
<atm-linx type="desclinks">
   <selection>
      <start><nodespec type="relative" value="root" /></start>
      <stop><nodespec type="relative" value="self" /></stop>
   </selection>
   <display>
      <condition property="id" operator="!=">
         <value type="nodeproperty" property="id"><nodespec type="relative" value="self" /></value>
      </condition>
      <htmltemplate><a href="{{href}}" title="{{jstitle}}">{{title}}</a> &gt;</htmltemplate>
   </display>
   <display>
      <condition property="id" operator="==">
         <value type="nodeproperty" property="id"><nodespec type="relative" value="self" /></value>
      </condition>
      <htmltemplate>{{title}}</htmltemplate>
   </display>
</atm-linx>
```

This example is a little more complex because it has certain conditions. It cannot link to the current page, which is useless as this is the page where we already are! It will add chevrons between each link to clearly separate the navigation path. This generates the following XHTML code:

```
<a href="url_home">Home page Title</a> &gt;<a href="url_page_bellow home">Title of the page bellow the home</a> &gt;Titre page actuelle
```

"Open" recursivelinks link:

```
<atm-linx type="recursivelinks">
   <selection>
      <start><nodespec type="relative" value="root" /></start>
      <condition property="lvl" operator="&lt;=">
         <value type="data">3</value>
      </condition>
   </selection>
   <display>
      <mode>open</mode>
```

```
      <htmltemplate><li class="{{lvlClass}} {{typeClass}} {{currentClass}}"><a class="{{lvlClass}}"
href="{{href}}">{{title}}</a>{{sublevel}}</li></htmltemplate>
      <subleveltemplate><ul class="{{lvlClass}}">{{sublevel}}</ul></subleveltemplate>
   </display>
</atm-linx>
```

This link generates an embedded, bulleted list describing the entire tree under the welcome page (root) until the third level maximum.
This generates the following XHTML code:

```
<ul class="CMS_lvl1">
   <li class="CMS_lvl1 CMS_sub ">
      <a class="CMS_lvl1" href="url_accueil">Titre page Accueil</a>
      <ul class="CMS_lvl2">
         <li class="CMS_lvl2 CMS_nosub ">
            <a class="CMS_lvl2" href="url_sous_page1">Titre sous page 1</a>
         </li>
         <li class="CMS_lvl2 CMS_sub ">
            <a class="CMS_lvl2" href="url_sous_page2">Titre sous page 2</a>
            <ul class="CMS_lvl3">
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-1">Titre sous page 2 -
1</a></li>
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-2">Titre sous page 2 -
2</a></li>
               <li class="CMS_lvl3 CMS_nosub CMS_current"><a class="CMS_lvl3" href="url_page_actuelle">Titre
page actuelle</a></li>
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-4">Titre sous page 2 -
4</a></li>
            </ul>
         </li>
         <li class="CMS_lvl2 CMS_sub ">
            <a class="CMS_lvl2" href="url_sous_page3">Titre sous page 3</a>
            <ul class="CMS_lvl3">
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page3-1">Titre sous page 3 -
1</a></li>
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page3-2">Titre sous page 3 -
2</a></li>
            </ul>
         </li>
      </ul>
   </li>
</ul>
```

You will note the presence of classes which allow you to know what the current page is (CMS_current), twhat the different levels are (CMS_lvl*n*) and if each of the pages has subpages (CMS_sub ou CMS_nosub). All these classes allow you to add style sheets as well as the necessary javascripts.

"Closed" recursivelinks link:

```
<atm-linx type="recursivelinks">
   <selection>
      <start><nodespec type="relative" value="root" /></start>
      <condition property="lvl" operator="&amp;lt;=">
         <value type="data">3</value>
      </condition>
```

```
   </selection>
   <display>
      <mode>close</mode>
      <htmltemplate><li class="{{lvlClass}} {{typeClass}} {{currentClass}}"><a class="{{lvlClass}}"
href="{{href}}">{{title}}</a>{{sublevel}}</li></htmltemplate>
      <subleveltemplate><ul class="{{lvlClass}}">{{sublevel}}</ul></subleveltemplate>
   </display>
</atm-linx>
```

The only difference from the preceding example is found in the <mode> tag which in this case specifies that he links tree must be "closed."

This generates the following XHTML code:

```
<ul class="CMS_lvl1">
   <li class="CMS_lvl1 CMS_sub ">
      <a class="CMS_lvl1" href="url_accueil">Titre page Accueil</a>
      <ul class="CMS_lvl2">
         <li class="CMS_lvl2 CMS_nosub "><a class="CMS_lvl2" href="url_sous_page1">Titre sous page 1</a></li>
         <li class="CMS_lvl2 CMS_sub ">
            <a class="CMS_lvl2" href="url_sous_page2">Titre sous page 2</a>
            <ul class="CMS_lvl3">
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-1">Titre sous page 2 -
1</a></li>
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-2">Titre sous page 2 -
2</a></li>
               <li class="CMS_lvl3 CMS_nosub CMS_current"><a class="CMS_lvl3" href="url_page_actuelle">Titre
page actuelle</a></li>
               <li class="CMS_lvl3 CMS_nosub "><a class="CMS_lvl3" href="url_sous_page2-4">Titre sous page 2 -
4</a></li>
            </ul>
         </li>
         <li class="CMS_lvl2 CMS_sub "><a class="CMS_lvl2" href="url_sous_page3">Titre sous page 3</a></li>
      </ul>
   </li>
</ul>
```

The only difference is in the display of subpages of subpage 3 which in this case are not visible because they are not found in a direct branch of the current page.

Multiple and conditional link:

```
<atm-linx type="direct">
   <selection>
      <start><nodespec type="relative" value="self" /></start>
      <start><nodespec type="node" value="12" /></start>
      <start><nodespec type="node" value="15" /></start>
   </selection>
   <display>
      <condition property="title" operator=">=">
         <value type="data">C</value>
      </condition>
      <htmltemplate><a href="{{href}}">{{title}}</a></htmltemplate>
   </display>
</atm-linx>
```

This code creates links to pages 12, 15 and the current page. Only, however, if they have a title which begins with the letter C or more on the ASCII scale.

There will be many other possible examples to add in order to describe the many possibilities offered by the atm-linx tag. Do not hesitate to consult those already existing in the demo suplied with Automne or to pose your questions in the forum.

Actions on page models

Creating a model.

Creating a page model corresponds to adding a new page template to your site. This new page model will be available when you create or modify a page.

Attention! By default, a new model is not active; in order to use it, the model must be activated with the "Activate" button.

Once created, the XML definition of the row is saved on the server as an XML file in the folder /automne/templates.

Modifying a model

Modifying a page model allows you to change its properties (name, description, thumbnail), but also to change its groups and sites in order to restrict its use by certain user profiles.

It is also possible to change the XML definition of your page model, its content rows and client spaces available when printing the page.

> Changing the XML definition of a model already used in the pages will apply the new formatting to all the pages using the model.

Activating / de-activating a model

Activating a page model determines if it can be used when creating or modifying a page. When it is created, a model is inactive by default in order to allow you to finalize it before it is used by editors.

Once the model is created and its XML definition is finished, remember to activate it so that editors can use it.

> De-activing a model has no impact on the pages already using it.

Deleting a model

You can delete a model only if it is no longer assigned to any page. You can see pages using a page model in the information returned when searching for a page model.

Deleting a page model is irreversible. Once deleted, your page model can no longer be found and this action is not submitted for validation.

Be careful not to delete a page model that may be used in the future; it is better in this case to de-activate it to prevent it from being used in pages.

Searching for a model

Searching alows you to easily find a page model using the filter display on the right. You can search by:

- Name,
- Description,

- Group,
- Site,
- Page,
- state (active/inactive).

See pages

See pages lists all the pages using this page model.

Regenerate pages

Regenerate pages recreates the cache of all pages using this model.

Content rows

Content rows are made up of one or more blocks of content that you put in your client spaces. Content rows permit input and organize the content of your site.

These blocks can contain all types of information:

- Title
- Text,
- Image,
- Documents,
- Flash animation,
- Module.

There are a number of default rows in Automne 4 (text, image, title h1, title h2, google map ...). You can also create yourn own content rows to enrich your site with many kinds of information.

All content rows are independent from one another. There are no restrictions on the number of times or the placement of rows on the site. **Content rows are usable everywhere at any time in the client spaces of your page models.**

Thanks to content rows, you have complete freedom in organizing the content of your site as well as an ease of customization without technical constraints.

- Creating a content row
- XML definition for classic content rows
- XML definition for module content rows
- Actions for content rows

Creating content rows

Label:

The title of the content row. The label must make it possible to identify the type of content managed by the row.

Example : "title", "subtitle", "image"...

Hint: A numeric prefix can be used to order the rows by their type of content.

The label can also contain the model to which the row can be attributed.
Example: "title - Welcome" "title" - interior "image-welcome" ...

Description:

Clearly describes the meaning and expected use of a content row. Example: "Row model title of an Interior model".

Group:

Allows you to classify the models according to certain characteristics.

You can next manage the profiles of users able to use this or that group of rows in the profile properties of users or user groups.

Example: A row model can belong to the "Administrator" row group. The rights for this row group will then only be assigned to users called "Administrator", thus limiting the use of the pages to those who possess the right to use them.

New groups:

Allows you to create a new classification group. Example: It is possible to create a group "Editor news" that corresponds to the row models able to be used by a person having editor rights over the row models and the news module

Page models:

Allows you to define the rows available according to the page model used. Example: An "interior title" row will be available for the model of the "Interior" page. A "welcome title" row will be available for the "Welcome" page model.

Icon:

Allows you to know easily what type of data the content row was created for. Example: A text, an image, flash, data of a module... Certain icons represent a combination of several types of data, for example two images side by side, or one on the left and one on the right. The icons are very useful for indentifying the rows quickly; choose them carefully.

New icon:

If there is no icon for the row, you can add another one with this field. Don' t hesitate to make use of it.

Types of data in row models:

Text: Text data.

Média: Image data ('jpg' / 'png' / 'gif'), documents (doc, pdf, etc.), video, audio.

Flash animation: Flash data, animation in '.swf'.

Download a file: Downloads a file with a click on the generated link.

Google maps: Inserts a Google Maps" map. To use the Googlemaps service you must have a valid Google key valid for the site. Next, write the line below in the config.php file (located at the root of your site):

define("GOOGLE_MAPS_KEY", "INSERT_GOOGLE_KEY_HERE");

Insert your key in place of INSERT_GOOGLE_KEY_HERE.

> From 4.0.0 of Automne the use of the Google Map row requires havin specified a Googlemaps key for your site.

Module rows

Form Module: Calls up the Form module and generates a form on the page.

News Module: Calls up the News module and generates the News on the page.

News - Search: Calls up the News module, generates one or more news items as well as t<o search fields: by keyword and category.

Latest news: Calls up the News module and generates the most recent news on the page.

Media library: Calls up the Media library module and generates Media on the page.

XML definition for classic content rows

Title or subtitle (255 characters max):

<block module="standard" type="varchar" id="uniqueID">{{data}}</block>

- **uniqueID:** Unique identifier of the block in the row.

The following values will be replaced in the tag:

- **{{data}}:** Text content

Formatted text (HTML):

<block module="standard" type="text" id="uniqueID">{{data}}</block>

- **uniqueID:** Unique identifier of the block in the row.

The following values will be replaced in the tag:

- **{{data}}:** Formatted content (HTML).

Image :

<block module="standard" type="image" id="uniqueID" maxWidth="value" minWidth="value">{{data}}</block>

- **uniqueID:** Unique identifier of the block in the row.
- **value:** Minimum or maximum value authorized for the image in pixels. The attributes maxWidth and minWidth are optional.

The following values will be replaced in the tag:

- **{{data}}:** Image code and link to zoom view, if it exists.
- **{{label}} or {{linkLabel}}:** Name / Image text, if it exists.
- **{{jslabel}}:** Name of the image (separated for inclusion in the javascript or attribute tag).
- **{{imageZoomHtml}}:** HTML code displaying the picture zoom if it exists.
- **{{imagePath}}:** Directory of the image on the server.
- **{{imageName}}:** Name of the image file on the server.
- **{{imageZoomHref}}:** Address (directory and name) of the image zoom file on the serverd.
- **{{imageZoomName}}:** Name of the image zoom file on the server.
- **{{imageWidth}}:** Width of the image file on ther server.
- **{{imageHeight}}:** Height of the image file on ther server.
- **{{imageZoomWidth}}:** Width of the image zoom file on ther server.
- **{{imageZoomHeight}}:** Height of the image zoom file on ther server.

File - attached:

<block module="standard" type="file" id="uniqueID">{{data}}</block>

- **uniqueID :** Unique identifier of the block in the row.

The following values will be replaced in the tag:

- **{{data}}:** Link to the file if it exists.
- **{{href}}:** Address (URL) of the file.
- **{{label}}:** Label of the file.
- **{{jslabel}}:** Label of the file (separated for inclusion in the javascript or attribute tag).
- **{{size}}:** Size of the file in mega octets.
- **{{filename}}:** Name of the file.
- **{{originalfilename}}:** Original name of the file.
- **{{type}}:** File extension.
- **{{icon}}:** Icon of the file if it exists.

Flash animation:

<block module="standard" type="flash" id="uniqueID">{{data}}</block>

- **uniqueID :** Unique identifier of the block in the row.

The following values will be replaced in the tag:

- **{{data}}:** Flash animation content.

Title of the current page:

<atm-title />

Value of a constant (see the constant files of Automne) :

<atm-constant name="constantName" />

- **constantName :** Name of the constant declared. All constants and especially those of files /cms_rc.php and /config.php are concerned

Link(s) to one or several pages:

<atm-linx type="linxType">linxDefinition</atm-linx>

- **linxType :** Type of link among:
  - **direct :** Direct link to a specific page.
  - **sublinks :** Link to the sub pages of the current page.
  - **desclinks :** Navation history from one page to another.
  - **recursivelinks :** Structure of the sub links from a given page.
- **linxDefinition :** See complete atm-linx tags syntax in Automne documentation.

Link to a printable page:

<atm-print-link keeprequest=true">{{href}}</atm-print-link>

- **{{href}} :** Replaced by the link to the printable page.
- Attribute **keeprequest :** Optional, allows to conserve the attributes GET from which the printed page is called (defaut : false)

Date of last update:

<atm-last-update format="d-m-Y">{{date}} {{firstname}} {{lastname}}</atm-last-update>

- **{{date}} :** Date of the last update of the content of the page. the format of the date shown is given by the attribute **format** of the tag.
- **{{firstname}} :** First name of the person responsible for the last update.
- **{{lastname}} :** Last name of the person responsible for the last update.

Javascript files of the page:

<atm-js-tags files="/js/js1.js,/js/js2.js" />

This tag allows you to concatenate several javascript files into one. This accelerates therefore the loading of the web page. The files listed are concatenated and compressed before being served to the user. An advanced management of the navigator cache is employed.

- Attribute **files** : Javascript files to include in the page (separated by commas). These files should imperatively be found in the directory /js/ or in a sub directory.

You can deactivate this minification of javascript files in the config.php file by defining the APPLICATION_JS_AND_CSS_COMPRESSION constant to false.

define("APPLICATION_JS_AND_CSS_COMPRESSION","false");

CSS style sheets of the page:

<atm-css-tags files="/css/css1.css,/css/css2.css" />

This tag allows you to concatenate several CSS files into one. This accelerates therefore the loading of the web page. The files listed are concatenated and compressed before being served to the user. An advanced management of the navigator cache is employed.

- Attribute **files** : CSS files to include in the page (separated by commas). These files should imperatively be found in the directory /css/ or in a sub directory.
- Attribute **media** (optional) : allows to specify the exit media employed among: **all**, **aural**, **braille**, **embossed** , **handheld**, **print**, **projection**, **screen**, **tty**, **tv**. By default, if this attribute is absent, the media **all** will be employed.

You can deactivate this minification of javascript files in the config.php file by defining the APPLICATION_JS_AND_CSS_COMPRESSION constant to false.

Identifier of the page:

```
{{pageID}}
```

- **pageID :** Replaced by the identifier of the current page.

Display in the administration:

<atm-admin> ... </atm-admin>

The content of this tag will be visible only if you are in the administration of Automne.

<atm-noadmin> ... </atm-noadmin>

The content of this tag will not be visible if you are in the administration of Automne.

Display in the page edition:

<atm-edit> ... </atm-edit>

The content of this tag will be visible only if you are currently editing the page in the administration of Automne.

<atm-noedit> ... </atm-noedit>

The content of this tag will not be visible if you are currently editing the page in the administration of Automne.

XML definition of polymod module rows

Each polymod module in Automne can use a **<block>** tag in the content rows, in order to use more or less complex dynamic content.

Here we deal with polymod modules, integrated into Automne's core and thus perfectly compatible with these tags. Many parsing features are offered by using the **<block>** tag in a polymod module.

Module blocks

> Dynamic help is available directly from the Automne administration interface.?You can access it while editing modules (pages and rows) by clicking on Help, under the "XML definition" tab.

All the polymod modules can natively use the <block> tag.

<block module="pnews" id="blockID" language="languageCode"> ... </block>

This tag displays data specific to the module. It must surround all the tags related to the treatment of module data.

- **blockID :** Unique identifier of a block of content in the row. Two content blocks of the same row must not have identical identifiers,
- **languageCode :** facultative) Language code related to this content block from among the codes available for your application (by default "fr" and "en").If not present the language of the page will be used.
- If not present, the default language of Automne will be used.
- **lifetime** (optional attribute) : This attribute lets you specify the duration of cache implemented on this block of data. It accepts the following values:
    - **0, false** : Disable the client side cache for this block.
    - **integer greater than 1** : Enables caching for the specified duration (in seconds).
    - **1, true, auto** : Enables the cache automatically (default if the attribute does not exist).
      The automatic cache to a maximum of 24 hours. It will be activated only if the data block contains no PHP code. For more information see the operation of Polymod cache datas.

All the tags cited below must be situated in a block tag when the module is a "polymod." If this is not the case the tags will not be correctly parsed and errors may occur. See the section regarding the block tag for PHP modules.

Searching module data

Searches for a given object type :

<atm-search what="objet" name="searchName">...</atm-search>

- **object**: Type of object to search for (in the form {object})
- **searchName**: Name of the search: unique identifier for the search in the row.
- A **public** attribute (facultative) may be added to specify a search in the public or edited area. It accepts true as a value for a public search (default) or false for a search in an edited area.

This tag can contain the following sub-tag :

Display results

<atm-result search="searchName"> ... </atm-result>

The content of this tag will be read for each result found for the search in progress.

- **searchName :** Name of the search to which to apply the parameter.
- A **return** attribute (facultative) can be added in order to specify the type of result returned. By default a search returns objects, but in the aim of improving results, it is possible to specify the two return values:
  - **POLYMOD_SEARCH_RETURN_IDS** will return only the identifier of the result. It can be recovered by {resultid}
  - **POLYMOD_SEARCH_RETURN_OBJECTSLIGHT** will return the result but without loading the sub-objects it can contain in its different fields. Attention, this parameter is not possible for a public search.

The following values will be replaced in the tag :

- **{firstresult} :** True if the current result is the first of the current page.
- **{lastresult} :** True if the current result is the last of the current page.
- **{resultcount} :** Number of results in the page.
- **{maxresults} :** Number of results for the search.
- **{maxpages} :** Number of pages maximum for the current search.
- **{currentpage} :** Number of the current page for the current search.
- **{resultid} :** Identifier of the result. It is useful in the case of a search returning uniquely the identifiers of results (return parameter with the value POLYMOD_SEARCH_RETURN_IDS).

No result

<atm-noresult search="searchName"> ... </atm-noresult>

The content of the tag will be displayed if no result is found for the current search.

- **searchName :** Name of the search to which to apply the parameter.

Search parameter

<atm-search-param search="searchName" type="paramType" value="paramValue" mandatory="mandatoryValue" />

Limits the results of the search to the given parameters.

- **searchName** : Name of the search to which to apply the parameter.
- **paramType** : Type of parameter, perhaps the form **{objet:champ:fieldID}** to filter the search for the value of a given field (see the dynamic help of the field).

Example, for a search "mySearch" in the "Recent Events" object, one wants a parameter for the "Category" field with the value {request:int:cat} :

<atm-search-param search="maRecherche" type="{Actualite:Categorie:fieldID}" value="{request:int:cat}" mandatory="true" />

Equally, it can be a fixed name, including :

- **object :** The value is an object of the form {objet}, for example {News}

- **item :** The value is an identifier of a polymod element. The value must be a positive integer. For example {var:int:newsid} can correspond to the '42' value

  To display the identifier of a polymod element pass your mouse over the corresponding fields in the module management.

- **items :** The value is a table of identifiers of polymod elements, in the array form (id1, id2, id3, ...). The search only uses these identifiers to return results. This is to say that the results of the search can only be from among the identifiers indicated in the table.

  For example a table PHP $items = array(1, 2, 3, 7); which is recovered with {var:array:items}

- **itemsOrdered :** The value is a table of identifiers of polymod elements, identical to the items type, but here the results will be sorted as defined by the string value of the table. Any other sorting will be ignored.

  For example a PHP array $items = array(1, 7, 3, 2); which is recovered by {var:array:items}.

- **keywords :** The value is a chain of characters.

  For example {var:string:keyword} which corresponds to the value "Search test"

  One can distinguish 2 cases for the keyword type :

  1. The object for which a search is made is not indexed by the "Search engine" module:?In this case the polymod searches for the string value (words separated by spaces, commas and semicolons) in the indicated fields as “searchable.”
  2. The object for which a search is made is indexed by the « Search engine » module:?In this case the search engine ASE effectuates its own search and returns a table of polymod element identifiers to polymod. The search is thus based on the relevance of results from the search engine.

  For your information, a search via the ASE search engine takes only the first 1000 results into account.

- **publication date after**: The value is a date in the format of the current language. This parameter verifies that the object is a primary resource and has a publication start date greater than or equal to the indicated date.

  Example in French (language code = fr) : 15/03/2010.?Example in English (language code = en) : 03/15/2010.

  The Polymod module automatically converts the format of the entered date (for French this will be d/m/y, for English in SQL format: y-m-d.??The SQL request performed in the background verifies that :

  - the object is a primary resource,
  - the publication start date is different from "0000-00-00",
  - the publication start date is greater than or equal to the value indicated.

- **publication date before** : The value is a date in the format of the current language. This parameter verifies that the object is a primary resource and has a publication start date lesser than or equal to the indicated date.

Example in French (language code = fr): 15/03/2010.?Example in English (language code = en): 03/15/2010.

The Polymod module automatically converts the format of the entered date (for French this will be d/m/y, for English in SQL format: y-m-d.??The SQL request made in the background verifies that:

- the object is a "primary resource"?
- the publication start date is different from "0000-00-00"?
- the publication start date is greater than or equal to the value indicated.

paramValue :

Search parameter value. If the value is 'block' you can specify this value when creating a row in the page.

When the valued of a search parameter is "block", this allows an editor to select a predefined value while editing the page containing the polymod search. ??It is sometimes useful to recover the selected value to use it in PHP

If the value is defined, it is stored in a table in the following form: $blockAttributes['search']['mySearchName']['fieldID']??For example, for a search named "newsHomePage", if we establish a search parameter for the field that has 5 as an identifier and with a value defined as "block", the value selected by the editor can be recovered here: $blockAttributes['search']['newsHomePage'][5]

mandatoryValue :

Boolean (true or false), specifies whether this search parameter of optional or obligatory.

operator :

A supplementary operator parameter adds a specific behavior to the types of field in the filter. The value accepted by this parameter is explained in the field help, if it accepts this parameter.

Here are the operator attribute values according to the different existing fields:

- **Categories**:
  - editableOnly: Only searches for objects associated with the editable category or categories supplies as a parameter.
  - strict: Only searches for objects associated with a category in parameter (the sub categories are not taken into account).
  - not in: Only searches for objects that are not associated with the category in parameter.
  - not in strict: Only searches for objects that are not associated with the category in parameter, in strict fashion (the sub-categories are not taken into account).
- **Character chains** :
  - like: Allows making a search for an incomplete value. Use the % character to specify the unknown part. For example, "cha%" returns, "chariot", "champion", etc.
- **Date** :
  - >= : Greater than or equal to
  - <= : Lesser than or equal to.
  - > : Greater than.
  - < : Lesser than
  - >= or null : Greater than or equal to or zero value.
  - <= or null : Lesser than or equal to or zero value.
  - > or null : Greater than or zero value.

    - < or null : Lesser than or zero value.
    - >= and not null : Greater than or equal to and non zero.
    - <= and not null : Lesser than or equal to and non zero.
    - > and not null : Greater than and non zero.
    - < and not null : Lesser than and non zero.
- **Integer** :
    - < : Lesser than
    - <= : Lesser than or equal to
    - > : Greater than
    - >= : Greater than or equal to
- **Floating number (point)**
    - < : Lesser than
    - <= : Lesser than or equal to
    - > : Greater than
    - >= : Greater than or equal to
    - like: Allows making a search for an incomplete value. Use the % character to specify the unknown part. For example, "cha%" returns, "chariot", "champion", etc.

Attention: If the operator attribute is not defined in the tag atm-search-param, the search is effected by default with the following parameter :

- Basic field "text" : like '%value%'?
- Basic field "string" : like '%value%'?
- Basic field "date" : like 'value%'?
- Basic field "integer" (tableau) : in(value)?
- Basic field "integer" (entier) : = 'value'??

Also see: Inheriting fields

Example of category field with a selectable value:

<atm-search-param search="homePageNews" type="{Actualite:Categorie:fieldID}" value="block" mandatory="true" />

Example of a "keyword" search parameter :

<atm-search-param search="homePageNews" type="keywords" value="{request:string:keywords}" mandatory="true" />

Displaying a given page of results (the number of results of a page is specified by the tag atm-search-limit):

<atm-search-page search="searchName" value="pageValue" />

- **searchName**: Name of the search to which to apply the parameter.
- **pageValue**: Numerical value of the page to display.

Limiting the number of results of a page:

<atm-search-limit search="searchName" value="limitValue" />

- **searchName** : Name of the search to which to apply the limit..
- **limitValue**: Numerical value of the limit to apply. If the value is 'block' you can specify this value when creating a row in the page.

Ordering the results:

<atm-search-order search="searchName" type="orderType" direction="directionValue" />

- **searchName** : Name of the search to which to apply the limit..
- **orderType** : Type of value to which to apply the order, which can be the form {objet:champ:fieldID} or a name of a fixed type such as: objectID, random, publication date after, publication date before.
- **directionValue**: Direction to apply : asc for ascending, desc for descending.If the value is 'block' you can specify this value when creating a row in the page.

Functions

Displaying the list of pages for the search in progress:

<atm-function function="pages" maxpages="maxpagesValues" currentpage="currentpageValue" displayedpage="displayedpagesValue">
<pages> ... {n} ... </pages>
<currentpage> ... {n} ... </currentpage>
<start> ... {n} ... </start>
<previous> ... {n} ... </previous>
<next> ... {n} ... </next>
<end> ... {n} ... </end>
</atm-function>

This function displays the list of all the pages possible to search.

- maxpagesValue: Maximum number of pages to loop (usually: {maxpages}).
- currentpageValue: Number of the current page from which you are searching (usually: {currentpage}).
- displayedpagesValue: Number of pages to display.
- The <pages> tag will be read for each page to list except the current page and the value {n} will be replaced by the page number.
- The optional tag <currentpage> will be read for the current page. If it does not exist, the <pages> will be used in its place.
- The optional tag <start> will be read for the first page.
- The optional tag <previous> will be read for the preceding page.
- The optional tag <next> will be read for the following page.
- The optional tag <end> will be read for the last page.

Works tags

Display the content of the tag if the condition is met:

<atm-if what="condition"> ... </atm-if>

- **condition** : Condition to meet to display the content of the tag. Common use is to validate the presence of a non negative value. This condition may also take all the valid forms of a PHP condition (see: The structures of a PHP control). The condition will be met if the value exists or if it is not negative or if it is not equal to false.
- An attribute **name** (optional) can be added if you want to run the opposite condition (else) using a tag atm-else (see below). This attribute can contain only alphanumeric characters (a-zA-Z0-9_. -).

<atm-else for="if-name"> ... </atm-else>

- **for**: name of the atm-if tag of reference. The content of this tag will be displayed only if the condition of the tag atm-if of reference is not met.
- An attribute **what** (optional) may be added if you want to add an additional condition to display the content

of this tag. Common usage is to validate the presence of a nonzero value. This condition may also take all the valid forms of a PHP condition (see: The structures of a PHP control). The condition will be met if the value exists or if it is not negative or if it is not equal to false.

Looping over a set of objects

<atm-loop on="objets"> ... </atm-loop>

- **objects** : Collection of objects. Employed to deal with all the objects in a collection of multiple objects (called multi-object).
- **reverse** (facultative) : attribute that can be added to inverse the order of results (value : Boolean true, false).

The following values will be replaced in the tag :

- **{firstloop} :** True if the current object is first in the list of objects.
- **{lastloop} :** True if the current object is last in the list of objects.
- **{loopcount} :** Number of the current object in the list of objects.
- **{lastloop} :** True if the current object is last in the list of objects.
- **{maxloops} :** Number of objects to loop.

Adding an attribute to an XHTML parent tag (containing this tag)

<atm-parameter attribute="attributeName" value="attributeValue" />

- **attributeName :** Name of the attribute to add.
- **attributeValue :** Value of the attribute to add.

Assigning a value to a variable

<atm-setvar vartype="type" varname="name" value="varValue" />

- **type :** Type of variable to assign : request, session or var.
- **name :** Name of the variable to assign Attention, reassigning an existing variable deletes to old value.
- **varValue :** value to assign to the variable.

Create an opening or a closing XHTML tag

In some cases, it may be interesting to create an opening or closing XHTML tag inside another tag (eg an atm-if tag). But this scenario breaks the XML syntax. In this case, you can use both tags:

<atm-start-tag tag="tagName" />

- **tag** : Name of the opening tag to display.
- You can then add as many attributes you want, they will all be added to the generated tag.

<atm-end-tag tag="tagName" />

- **tag** : Name of the closing tag to display.

Example :

Your row contains the following code:

```
<atm-if what="condition1">
   <atm-start-tag tag="div" class="class1" />
</atm-if>
<atm-if what="condition2">
   <atm-start-tag tag="div" class="class2" />
</atm-if>
... some XHTML content ...
<atm-tag-end tag="div" />
```

If condition 1 is fulfilled, you will get the following code:

```
   <div class="class1">
... some XHTML content ...
</div>
```

Reloading an area in AJAX

<atm-xml what="condition"> ... </atm-xml>

Reloads a specific area of the page via an AJAX request.

If the HTTP request calling the current page contains the parameter verifying the attribute condition, only the content of the atm-xml will be sent in response to the HTTP request. The rest of the page content will be ignored.

For more information, study the Polymod modules supplied with the Automne demo, they use this feature.

- **condition** : Condition à remplir pour obtenir le contenu du tag atm-xml.

Example with JQuery

Your row contains the following code:

```
<atm-xml what="{request:string:out} == 'xml'">
   ... news search ...
</atm-xml>
```

If we make the following Ajax request to the page with the following row

```
$.ajax({
  type: "POST",
  url: pageURL,
  data: 'out=xml&cat=' + $('#cat').val(),
  success: displaySearch
});
```

The search for recent events will be re-launched with the parameter of the selected category and only the content of this area returned:

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <error>0</error>
   <data><![CDATA[ ... the content of the atm-xml tag corresponds to the new recent events search ... ]]></data>
</response>
```

Referencing an element to the cache manager:

<atm-cache-reference element="element" />

- **element** : Referencing a module using his codename: **cms_aliases**, **cms_forms**, **cms_pdf**, **comments**, **pmedia**, **pnews**, **standard** or an Automne user or group using the value: **users**.

This tag allows you to inform the cache manager that you use in a PHP code inserted into the block a given element (module or Automne users/groups). Thus, the cache manager can monitor the changes made on this element to refresh the cache if the referenced element is modified.

General variables

Variables related to pages

The variables related to pages haves the following form: **{page:id:type}**:

- **id :** Identifier of the page to be referenced, perhaps a numerical identifier or even "self" to refer to a current page.
- **type :** Type of data desired for the page from among the following: url (adress of the page), printurl (adress of the print page), id (identifier of the page), title (name of the page).

Attention, if you use this to generate the href of a link, this will always be displayed even if the user does not haverights to consult the page. Only atm-linx links verify rights before display.

Variables related to sent data (via an address or form)

The variable correspond to a variable coming from the submission of a form or even from a parameter of the link leading to the current page. They have the following form **{request:type:name}**:

- **type :** Corresponds to the type of variable expected, from among the following:
  - **int :** integer,
  - **string :** character chain,
  - **bool :** Boolean,
  - **array :** Table of values,
  - **email :** valid email,
  - **date :** valid date without hour (returns a date in the MySQL format: YYYY-MM-DD),
  - **datetime :** valid date with hour (returns a date in the MySQL format: YYYY-MM-DD HH:MM:SS).
  - **localisedDate :** valid date without hour (returns a date in the format of your current language: jj/mm/aaaa).
  - **name :** Corresponds to the name of the desired variable (name of the parameter in the URL or name of the form field).

Session variables

These variables are available during navigation of the site by the visitor. They have the following form: **{session:type:name}**

- **type :** Corresponds to the type of variable expected, from among the following:
  - **int :** integer,
  - **string :** character chain,
  - **bool :** Boolean,
  - **array :** Table of values,
  - **email :** valid email,

- **date :** valid date without hour (returns a date in the MySQL format: YYYY-MM-DD),
- **datetime :** valid date with hour (returns a date in the MySQL format: YYYY-MM-DD HH:MM:SS).
- **localisedDate :** valid date without hour (returns a date in the format of your current language: jj/mm/aaaa).
- **name :** Corresponds to the name of the desired variable (name of the parameter in the URL or name of the form field).

Variables

These variables correspond to the classic PHP variables. They have the following form: **{var:type:name}**:

- **type :** Corresponds to the type of variable expected, from among the following:
  - **int :** integer,
  - **string :** character chain,
  - **bool :** Boolean,
  - **array :** Table of values,
  - **email :** valid email,
  - **date :** valid date without hour (returns a date in the MySQL format: YYYY-MM-DD),
  - **datetime :** valid date with hour (returns a date in the MySQL format: YYYY-MM-DD HH:MM:SS).
  - **localisedDate :** valid date without hour (returns a date in the format of your current language: jj/mm/aaaa).
  - **name :** Corresponds to the name of the desired variable (name of the parameter in the URL or name of the form field).

Variables related to users

Variables related to users have the following form: **{user:id:type}**.

- **id :** User name of the user to be referenced, perhaps a numerical identifier or even "self" to refer to a current user (only if the client-side rights verification is activated in the Automne parameters).

- **type :** Corresponds to the type of variable expected, from among the following :

  - **login** : username,
  - **fistName** : first name,
  - **lastName** : last name,
  - **fullName** : first and last name,
  - **email** : email,
  - **language** : returns a CMS_language object corresponding to the language of the user's profile,
  - **active** : Boolean, true for active, false for inactive,
  - **deleted** : Boolean, true for deleted, false for existing

Form tags (création - modification).

Creating/modifying client-side objects

<atm-form what="objet" name="formName"> ... </atm-form>

This tag creates an input form for a new object (if this tag is not in a search result) or of the modification for an existing object (if this tag is found in a search result).

- **objet :** Type of object to input (in the firm {object})
- **formName :** Name of the form: unique identifier for the form in the row.

The following values will be replaced in the tag :

- **{filled} :** True if the form is filled in correctly and its validation causes no errors
- **{required} :** True if the form is filled in correctly and the required fields are left empty
- **{malformed} :** True if the form is not correctly filled in and the values of certain fields are incorrect.

This tag can contain the following tags

Display of required fields

<atm-form-required form="formName">
... {requiredname} ...
</atm-form-required>

The content of the tag will be read for each field required while the form is being filled in

- **formName :** Nom du formulaire sur lequel appliquer le tag.

Les valeurs suivantes seront remplacées dans le tag :

- **{firstrequired} :** True if the current required field is the first of the current field.
- **{lastrequired} :** True if the current required field is the last of the current field.
- **{requiredcount} :** Number of the required field in the current form
- **{maxrequired} :** Number of required fields in the current form
- **{requiredname} :** Name of the required field in the current form
- **{requiredfield} :** Required object field in the current form.

Displaying malformed fields:

<atm-form-malformed form="formName">
... {malformedname} ...
</atm-form-malformed>

The content of the tag will be read for each malformed field while the form is being filled in

- **formName :** Name of the form to which to apply the tag.

The following values will be replaced in the tag:

- **{firstmalformed} :** True if the current malformed field is the first of the current form
- **{lastmalformed} :** True if the current malformed field is the last of the current form
- **{malformedcount} :** Number of the malformed field in the current form.
- **{maxmalformed} :** Number of malformed fields in the current form.
- **{malformedname} :** Name of the malformed field in the current form.
- **{malformedfield} :** Malformed object field in the current form.

Display of an input field

<atm-input field="{object:field}" form="formName" />

This tag will be replaced by the input area (form field) necessary for the correct input of information related to the type of field specified.

- **formName :** Name of the form to which to apply the tag.
- **{object:field} :** Object field managed by the form for which the input must be made

This tag can next immediately have the HTML attributes that will be reposted in the HTML of the generated field (width, height, id, class, etc.).

Display of a field with complex verification

```
<atm-input field="{object:field}" form="formName">
 ... <atm-input-callback return="returnValue" /> ...
</atm-input>
```

This tag validates the fields in an atm-form

- **returnValue** : The presence of an atm-input-callback tag with the attribute return="valid" will validate the content of a field; in the absence of the tag or value other than "valid" the field will return a malformed error.

This example searches to sees if a user already exists for a given email, in which case blocking re-registration.

```
<atm-input id="participant-email" field="{Participant:Email}" form="quizInscription">
 <atm-search what="{Participant}" name="emailCheck">
 <atm-search-param search="emailCheck" type="{Participant:Email:fieldID}"
value="{request:string:participantEmail}" mandatory="true" />
 <atm-result search="emailCheck">
 <atm-input-callback return="invalid" />
 </atm-result>
 <atm-noresult search="emailCheck">
 <atm-input-callback return="valid" />
 </atm-noresult>
 </atm-search>
</atm-input>
```

XML definition for PHP module rows

Each module in Automne can generally use a **<block>** tag in content rows, in order to use simple or complex dynamic content.

Here we deal with classic modules, made manually with PHP.

Data blocks

<block module="cms_forms" id="blockID" type="formular"> ... </block>

Modules created in PHP take into account the <block> tag in the content rows from Automne v. 4
Prior to this it was necessary to indicate the inclusion of <block> tag classes in the module concerned.

For a classic module the <block> tag will be replaced by default with a "module model". This is a file located in /automne/templates/.

The block tag is replaced by the file **/automne/templates/mod_[module]_[type].php**.

Where **[module]** is the codename of the module and **[type]** is the type indicated in the block type.
Be careful to use a unique type, without special characters.

Example :

The following tag:

<block module="cms_forms" id="blockID" type="test"> ... </block>

will be replaced by the content of the file:

/automne/templates/mod_cms_forms_formular.php

> It is very useful to use the module models.
> They insert a PHP code from an external file to that of the row, and authorize the modification of headers via PHP.

Recovery of the attributes values

In your module model, it is possible to recover the attributes of the block tag. The value of attributes is automatically stored in the table **$mod_[module]**.

Where [module] is the codename of the module.

Example :

For the tag:

<block module="formations" id="formationsAccueil" type="homepage" myAttribute="maformation"> ... </block>

The file /automne/templates/mod_formations_homepage.php will contain, for example:

```
<?php
// Module template
echo $mod_cms_forms['myAttribute']; // maformation
?>
```

Actions on content rows

Creating a row

A row allows you to input content with a fixed format. The creation of a new row allows you to add a new kind of content to your pages or new formatting for any given kind of data.

Once created, the XML definition of the row is saved on the server as an XML file in the /automne/templates/rows folder

Modifiying a row

Modifiying an existing row changes its properties (name, description), but also its groups in order to restrict its use to certain user profiles.

It is also possible to change the XML definition of your row to adapt the formatting of the data it contains.

> Changing the XML definition of a row already used in pages will apply the new formatting to all pages using this row.

Activating / de-activating a row

Activating a row determines if it can be used when creating or modifying a page. When it is created, a row is inactive by default in order to allow you to finalize it before it is used by editors.

Once the row is created and its XML definition is finished, remember to activate it so that editors can use it.

If a row is created for a unique use (for a particular need in a page), you can de-activate it after it it is inserted to avoid it being re-used in another page.

> De-activing arow has no impact on the pages already using it.

Deleting a row

You can delete a row only if it is no longer assigned to any page. You can see pages using a row in the information returned when searching for a row.

Deleting a row is irreversible. Once deleted, your row can no longer be found and this action is not submitted for validation.

Be careful not to delete a row that may be used in the future; it is better in this case to de-activate it to prevent it from being used in pages

Searching for a model

- Name,
- Description,
- Group,

- Page.

Cache on polymod rows

Since Automne 4.0.2, XHTML code produced by polymod modules rows can be cached.

This cache concerns the generated content by the XML code in the rows inside block tags:

<block module="polymod-codename"> ... </block>

Please check rows tags syntax for the polymod modules for more information about this tag.

The cache system was written to improve the performance of the display of modules data and also for SEO. Of course, it doesn't affect the freshness of the data displayed to the users. Automne will override rows cache parameters everytime you update the modules datas in the administration interface

Processing modes

1.

Automatic cache

By default, every row containing a block tag related to a polymod module will have a 24 hour cache preset.

The default mode is available for all rows for blocks that doesn't contain a **cache** attribute of if the attribute **cache** is set to 1:

<block module="polymod-codename" cache="1"> ... </block>

Moreover if a block contains PHP code, cache will be automatically disabled.

The default duration if 24 hours can be modified in the config.php file via the CACHE_MODULES_DEFAULT_LIFETIME constant.

For instance, it you want the cache to last one week:

define("CACHE_MODULES_DEFAULT_LIFETIME", "604800");

2.

Forced cache

You can force the duration of a cache by specifying the number of seconds in the cache attribute of the block tag.

For instance if we want the cache to last for one hour :

<block module="polymod-codename" cache="3600"> ... </block>

If you force the cache and you have PHP code in your row, you can tell the cache system what your PHP code refers to. So Automne will be able to automatically reset your cache if the reference is modified.

The atm-cache-reference allows to specify what your code refers to.

3.

Disabled cache

You can force to disable the cache if you set the cache attribute of a block tag to zero (0).

<block module="polymod-codename" cache="0"> ... </block>

Reset cache and exceptions

- Polymod modules cache can be globally disabled, you have to set CACHE_MODULES_DATA constant in the config.php file to false. define("CACHE_MODULES_DATA", "false");
- You can also delete the whole cache in Automne administration in the server settings, "File Access" tab by clicking on the "reset Polymod cache" button.
- **Cache will be disabled** if POST data are present when the page containing the row is loading
- **Cache content for the row will be automatically deleted** if the content of the module(s) it refers is modified (either addition, modification or deletion of elements or categories)

Pages administration

Your site is structured according to a tree stucture. **The page 1 is your site's root**.

The tree strycture's organisation is free. Yet some rules will permite you to get a site plan stuctured and clear but also usable for the creation of your models.

You can consider some pages as 'folders' in which you'll store subpages. This allows to increase the clarity of your tree structure and permits to create groups of pages, that you could list dynamically, exemple for the creation of a navigation menu.

If your site is made fo several subsites, as part of a several language site for exemple, create a subpage for each subsite, under the root of your site.

The différent state of the page

Page Creation


- New page created pending publication validation.


- New page created refused publication validation.


- New page pending deleted validation.


- New page refused deleted validation.

Page modification


- Page published.


- Modified page pending publication validation.


- Modified page refused publication validation


- Page un-published.


- Modified un-published page pending validation.


- Modified un-published page refused validation.


- Published page pending deleted validation.


- Modified published page pending deleted validation.


- Published page refused deleted validation.


- Un-published page pending deleted validation.

- Modified un-published page pending deleted validation.

- Un-published page refused deleted validation.

State icon

-  Locked, the page is currently locked cause somoene is editing the content.
-  Draft, the content of the page is at the moment a draft..

Page archive

- Published page pending archive validation.

- Page refused archive validation.

- Modified published page pending archive validation.

- Un-published page pending archive validation.

- Un-published page refused archive validation.

- Modified un-published page pending archive validation.

- New un-published page pending archive validation.

- New page refused archive validation.

Order of links

- Order of links pending validation.

- Order of links refused validation.

Page properties

General

All the modifications on the page properties are submitted for validation and will be on line only after this step:

- **Title:** The title is in the browser heading. It is also usfull for the search engine optimization of your page. You can insert up to 256 characters.
  The page title is the one used by the special marker <atm-title />.
  The page title and the link are two datas that can be used in an <atm-linx> tag in your templates
- **Link:** It's the name of the link towards the page that will appear for instance in a navigation menu. The links are automatically created in each page according to its position in the tree structure. You can insert up to 256 characters.

WARNING: if a page is not published, you cannot create an internal link towards it!

- **Template:** Template used by the page. A template is made of a fix page structure with dynamic content spaces containing rows of content (titles, paragraphs, images, attached files...).

WARNING: using an incompatible model may cause loss of data. Incompatible models are written in red. A model is compatible when it consists of the same client spaces (same count, same id).

- **Address :** This is the address of the online version of the page .The page title is the element used to make the page address (the URL). When you change the page title, you can change the address thereof, ckecking the box « Update the page address ». If you don't check the box, the address will stay unchanged.

WARNING: changing the title may break external links toward the page, and affect the links displayed by search engines too.

WARNING: You have to check the address option with caution. Indeed, when you're checking this option, the file name generated changes according to the link wording . If an internet use;r bookmarked this URL, he may stumble upon a freaking 404 page (page not found) next time.

- **Redirection :** Redirects automatically a page toward another internal or external page. By default no redirection is employed. Choose the appropriate type: internal link within your website, or external link. For internal pages enter the identification number, or select from your site plan. For external sites you must begin your link with "http://".

Others scopes unchangable from the page properties:

- **Identification (ID) :** This is the reference of the page in the tree sstructure. This cannot be modified.. it is used to make Automne's links.
- **Website:** Main Site : Permites to see if Automne manage several sites (plate-forme multi-sites). The site can be modified only by an administrator (Administration / Sites administration)
- **Print page active :** This informs you on the print state of the page. This function is managed by the administrators having the management rights on the templates.

Dates & alerts

- **Publication start :** Date of beginning of publication. To immediatly un-publish a page currently online, simply postdate the beginning of publication. Respect the mm/dd/yyyy format.
- **Publication end :** This date makes it possible to publish and un-publish pages automatically. (If you leave empty the date completion of publication, the page will be permanent). To un-publish immediately a page on line for an indefinite period, simply antedate the end of publication. The links pointing towards an un-published page will be also removed until the next publication. Respect the mm/dd/yyyy format.
- **Alert delay :** Period after which one wants to be warned by e-mail that the contents of the page need to be modified. Enter "0" for no alerts.
- **Alert date :** Alternative to the Alarm Delay, makes it possible to be informed by e-mail on a precise date of your choice. Respect the mm/dd/yyyy format.
- **Alert message :** You can assoiate a personalised message of the alert which will be included in the mail sent.

Search engines

These fields are used by the national and international search engines and may also be used by your website's internal search engine. The proper use of these fields makes it easier for the page to be found in th results of a search engine. Click "save" to save the modifications.

Description

Give a description to your page, that can be displayed in the search engine results

Keywords

Define the keywords, put a comma between each keyword.

Category

Define a category for your page. You can insert up to 256 characters.

Robots

Users can specify a particular request to search engines. For example, if you don't want the page to be indexed in the search engines, indicate in the Robots field 'noindex' without the quotation marks. See your administrator for more information on the use of search engine meta tags.

Meta data

Automne enables you to manage the most current meta tags. These tags are hidden in the source code to give specific information concerning the page. These tags are not obligatory and are only there as an indication. Some tags may not appear if they are deactivated by the administrator. Click "save" to save the modifications.

- **Author:** Type in the name of the author of the page. This tag is for information only. You can insert up to 256 characters.
- **Reply to e-mail :** Type in an e-mail address to allow the visitors to contact you. You can insert up yo 256 characters.

- **Copyright :** Type in a date and/ or a copyright name. You can insert up to 256 characrters.
- **Language used :** Type in the language used for the page.

  The page language can be determinant as part of the development of some rows linked to moduls.

- **Browser cache :** Check the box to force the option "no cache".
- **Free meta data :** Enter other meta data in html format. Contact your administrator to ensure proper syntax. This is the right place to add for instance links for your RSS feeds.

Subpages

Subpages list with the possibility to remove the pages. You can remove and organize the pages using drag and drop.

> To be allowed to remove a group of pages, you must have the right to regenerate pages (administration tab of rights administration).

Log

This log displays the history of all the modifications made by users on this very page.

Page creation

The first thing to do before creating a new page is to choose its position in the tree structure, that is to say post the page which will directly be over the page you want to create.

Nevertheless, each page can be removed in the tree structure after it's creation, by someone who's got the rights for.

To create a page, you can :

- create a new page
- copy an existing page.

Create a new page

Once you have choosen the parent page, check "create" in the page administration menu bar.

In the opening window, enter all properties required for the page creation.

Only the title is required**.** If you don't fill in the link title, it will take the Title value. You can change this two fields whenever you want in the page properties tab.

You can create the page with or without the default rows. Checking the box, you have access to the modification of you page with the rows defined by default for the model.

Otherwise, you start with a empty page. In either case, you can modify, re-arrange, delete rows of your page at any moment.

Then you've got to choose the model : the model defines the general settings of your page. if an image was defined for the model, this is the one that will appears in the available models zone. only the models you're allowed to use will appear. To use a model, you need :

- the model to be active
- to have the rights on the model group from which the model is part of.

You have the possibility to screen the models posted entering a keyword of the model into the field "Filter...". Since 3 characters, the search is possible, and it is refined function of the rest of what you write.

Hit the Create button and your page will be created, but won't be visible online until you publish it. You've got now to edit its content. You aren't force to write the page content just after you created it. You can come back to the page edition whenever you want

Copy an existing page

Once you have choose the page to copy, check "Actions/ Copy the page".

You can copy the page with or without the content. Even if you don't want to keep the content, you will keep the rows structure.

You can modify the page template.

If you select an un-compatible template (marked in red), you'll loose all or part of the content and of the rows structure.

Check the page you want to design as parent page of the new page. You only see in the tree structure, the page for the which you have edition right. On the top of the tree structure, the field « search» permites you to filter from the page title. you just have to enter at least 3 characters to start the search. Then, the search refine with the rest of what you write.

All the base data are copied, including the page Title. To change the data follow the same procedures as for the classic modifications of the page properties.

The page creation is submit to validation. Function of the rights you have, you have the possibility or no to validate the content for on-lining.

To validate a page, you have the possibility :

- to stop by the tab « Pending validation » on the side panel.
- to check the status icon. If a validation is required, you'll have access to a quick menu allowing the page validation, or to reach the ripe validation options.

When the page validation is done, the modifications provided are visible online just after its regeneration.

Page content edition

Once a page is created, you can edit its content at any moment. Only some zones are free for edition. These are the zones created as 'client space' in your models.

A page is structured with content rows, which you can consider as data pack piled up once on the other. Packs can be of various type and you can create very specifics content rows function of your needs.

There is two phases you have to respect in order to change the page content :

- Create its structure : that means add or delete the row(s) required.You can add or delete a row at any moment.
- Edit the rows content.

The first phase is very important. Even if you can re-order the content rows ou edit them whenever you want, you won't be able to change your rows without loosing their content.

Take your time to define your page structure and how you want to display your content, it'll directly impact the users comprehension as well as search engine optimization.

Rows administration

Definition

Content rows are data packs. Some standard rows already exist.If you've got the apropriate rights, you can change or create new rows, depending of your needs.

Rows, are made up of one or several data types. When there is no content, that is to say if the row is empty, those datas types are reprsented in a specific way.

Standard type datas :

- pre-formatted text
- free text zone.
- image: an image specifying that no image was found.
- flash : an image specifying that no flash object was found.
- file : a wording "enclose file".
- form : an image representing a form.

Characteristic datas of a module.

As long as you didn't specifically enter your content, empty rows are not visible online (more details about type datas).

To reach your page modifications, check the box « Change»

A red frame appears all around your page, that means you're on "Changing" mode. Green dotted line zones also appear, this indicate the client-spaces position, that is to say the publishable zones position. **You could only act within those zones.**

Add new content

To add content you need first to add a row. Check "New row".
Now you have to choose, where in the page you want to add the row. You can also change your row position later.

Once you have checked the wanted position; it becomes green and you can select the row to add. You can see in the list only the authorized rows for your profile and for the current template. Choose the row you need and click "add".

The "empty" row appears.

Rearrange your content

When you mouse over the content rows in the page, a frame appears around each of them.. This frame highlight the space of the content row and give you access to the possible actions on the current row:

- Check the icon **« Delete this row »** to delete definitely the row,
- Check the icon **« Bring up this row »** to move the row one place up.

- Check the icon **« Bring down this row »** to move the row one place down,
- Check the icon **« Bring down this row to the editable zone bottom »** to move the row at the bottom of the client space,
- Check the icon **« Bring up this row to the editable zone bottom »** to move the row at the top of the client space,
- At least, check **« Drag and drop this row »** to move the row anywhere in the page. It works between different client spaces too.

If there more than a single client space in the page, you will have access to two others options :

- Check **« put this row into the right editable zone »** to remove the row in the following client space,
- Check **« put this row into the left editable zone »** to remove the row in the previous client space.

Edition of the different rows

When you mouse over the rows, two others buttons appear on top of each editable block.

Pre-formatted text

Pre-formatted content is often used for datas like page title. Content limits can impose itself. You don't have the possibility to add XHTML. By default you can insert up to 256 characters.

Enter your content and check the icon « Validate ».

WYSIWYG text edition

This content permits you to insert up rich content. To change your text format, you can use the visual editor or WYSIWYG in which you'll find some of the word processing functions. So it is really easy to format the text even if you don't know HTML langage.

Insert up your content and check "validate". You're back to the page and modifications are now considered. Check « return» to cancel modifications and come back on the page before modifications.

The visual editor is available for each free page block administrated by Automne. The interface propose a word processing toolbar, it allows redactors to format the t est quicly respecting xHTLM norms required for the application. The styles applied to textare linked to style sheets defined in the visual identity.
Do not hesitate to enlarge the edition window according your needs, "checking" one of the corners..

WARNING : a WYSIWYG editor isn't exactly a word processing software. A direct copy and paste from a word processing software can break your styles and layout. To paste up Word text, prefer the button "Paste as plain text", this will clean the included word processing styles inappropriate for the web. you can structure and style your text within the WYSIWYG editor who provides the necessary tools to make sure you use the right styles specifically made for your website.

The toolbar used for rows like « free text » are called « Default » . They are editable from the menu Templates/ WYSIWYG toolbars into the sidepanel, if you've got administration rights on your site.

To change some "style" menu elements, you need administration rights and you have to edit the "editorstyles.xml" file This file is available from the sidepanel Templates/ Stylesheets. You can freely change this file, this will permits you to access to the defined styles directly in the drop-down menu.
You'll also have to include the element style into the CSS styles of your website, so that it can be applied in your published pages.

To create or change a personal template, you need administration rights and you have to change the « editortemplates.xml » file. This file is available from the sidepanel Templates/ Stylesheets.

To access a table properties, right click on the table. The following options appears :

Insertion / Internal link : to create an internal link to another page, select text and click on the "Automne link" icon. A window with the tree structure of the pages you're allowed to modify opens. You can browse the tree structure and select a destination page by clicking on it.

The checking box "Keep the link content if the page do not exists anymore" permits you to choose if the text on which the link is done must disappear at the same time as the page targeted by the link. If this box is un-checked and the target page deleted or dispublished, the link content will disappear (this can be embarrassing if it is situated in the middle of a paragraph).

Into the "target" menu, you can choose to open the link in the same window, into a new window or in a pop-up, which you must specify the dimensions.

The « Advanced» tab allows you to add more informations like a longer title or a language. These informations are important if you want to improve the accessibility of your content.

To change the link, you have to right click on the link or on the link icon to modify it. To delete it, select the link and check the icon "Delete the link".

Insertion / External link : to create a link to an another page on the web, select some text and choose "external link". Select the link type (URL, email ou anchor) and enter the complet address without quotation marks or space. The link can be opened into the same window, into a new window or in a pop-up, which you'll have to determinate the dimensions.

To change the link, you just have to choose the word or the sentence containing the link and enter a new address. To delete it, select the word containing the link and choose Delete the link

The « Advanced» tab allows you to add more informations like a longer title or a language. These informations are important if you want to improve the accessibility of your content.

WARNING : to create a "anchor" link in the same page you have to edit the source code. Anchors are automactically detected only within a single free editable region.

Images

The principle of image modification is the same as content modification. We first click on the of the selected image, that will open a window.

Browse your hard disk to select an image and fill the alternative text input field only if required.

WARNING : images can have size limitation. If necessary, you'll be warmed of the maximum and/or minimum value for the width. Nevertheless, you can enter your image and use the resize function, in order to adapt the image to the defined format.

To re-size an image, first insert an image then click modify. You see your image and a floating window with options that allows you to freely change the size of your image. The image is re-sized withtout moodifying the height/width ratio. Click on « Apply » into see the modifications.
Click on the close button of the pop-in window to come back to you page

WARNING : using the resize function several times can lead to an alteration of the image quality.

Flash animations

This row permits you to add Flash animations in pages. As for images or enclosed file, Flash animation is a block inserted into a template. If you don't have access to the Flash animation row, ask your administrator to create one.

First, browse for a .swf file on your hard disk then enter the animation width and height in pixels.
Fill in the others parameters according to your Flash file and to your needs.

The maximum weight is given underneath the "File" field. In order to delete a file, click on the "Delete" icon.

File

You can also add some files for download like a PDF in your page.

Browse your computer for a file - all formats are allowed except some like EXE files for security reasons - and give a title for the link to this file.

The maximum weight is given underneath the "File" field. In order to delete a file, click on the "Delete" icon.

Form

You can insert forms in your pages. You can create and edit them in the forms module.

Edit the block to select and preview a form. You can filter forms by categories or language. Once you're done, you can close the window.

You can directly access the forms module by clicking on "Manage module datas used by this block" provided that you have sufficient rights on the module.

Polymod/modules rows

Module rows allows you to display datas from your modules (news for instance)

On some module rows you have to specify one or many parameters (like a category or the number of items to display). Click the edit button, enter a value, apply your changes and the module datas are then displayed.

Possible actions

Each one of your modification is automatically saved in the page draft.

You have access to several actions, all along the page modifications, available on the top of the visualisation zone of your page:

Options : allows you to activate or disactivate specifics options of your page edition,

- **Animated rows moving :** Enable visual effect when you move rows. Uncheck it if you need better performance.
- **Window scroll:** when this option is activated, the browser window is automatically placed on the row top when you update its content. Uncheck it if you need better performance.

- **New row :** allows you to add a row in the current page

- **Publish this content :** available if you have page validation rights. It performs two operations at the same time : submission and page validation so that your page is directly published online.The current draft is always deleted after publishing.
- **Submit to validation :** when the page validation is over, you can submit your page to validation. Your page will be visible on line only after its validation. This action also delete the draft, as it is now submit to validation.
- **Preview:** Preview the current draft to check how it will exactly appear online

User rights / groups

With Automne it is possible to define rights for:

- A unique user,
- A user group.

It is possible to associate a user with a user group while preserving the user's management rights; these rights will remain advanced, coherent and easy to maintain. To access the user management interface you must have the right of "user management".

During the installation of Automne, you must connect to the administrator account that includes all the rights for the site. Once connected you thus have access to the two following links via the sidebar under the title "User management":

- User profiles (link to User management page)
- Group profiles (link to the User group management page)

The "your profile" link is a practical shortcut that allows you to access the interface for user management by directly displaying your profile.

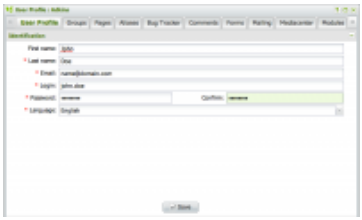
User management

Definition

The user management interface displays all the users registered for the application and allows one to manage their rights according to the different existing modules.

Some rules regarding users:

- Users can create other users if and only if they have "User management" rights.
- A user can be associated with one or more groups.



Properties

When a user is selected, several tabs are available to modify their profile and rights:

- User profile,
- Associated groups,
- Administration,
- Modules: a tab is displayed for each module available for administration.

When a new user is created, only the **"User profile"** tab is active. As soon as the user is created, you can access all the other tabs.

"User profile" tab 

Identification:

- **First name:** User's first name.
- **Last name:** User's last name.
- **Email :** User's email address (if necessary, alerts will be sent to this address).
- **Login:** User login for Automne (this login must be unique and be composed of at least 1 character).
- **Password:** User password (a password of at least 5 characters is required and it must be different from the login. For security reasons, we suggest a password of at least 8 characters containing letters and numbers).

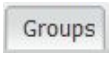
- **Confirm:** It will be necessary to re-type the previously indicated password. This confirmation verifies that no errors were made when the password was input.
- **Language:** User's primary language. Only languages defined by the application are available in the selection box.

Contact info:

- **Position:** User's position in the company.
- **Department:** Department to which the user belongs.
- **Telephone:** User's land line .
- **Cell:** User's cell phone number.
- **Fax:** User's fax number.
- **Address:** Mailing address of the user. This information is broken into 3 lines (for example, line 1 = floor or building, line 2 = street and number, line 3 = empty).
- **Zip code:** User's zip code (post code).
- **City:** User's city.
- **Sate or province:** User's state or province.
- **Country:** User's country.

Email alerts:

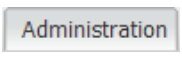
- **Modification of your profile:** Send an email to the user when their profile is modified.
- **Validations :** Send an email to a user when a page is pending validation in one of the sections to which the user has validation rights.
- **Page alerts:** Send an email to a user when an alert is signaled for a page the user is working on.

"Associated groups" tab 

The list of all groups is under this tab.

For each group we can see if the user is associated (or not) with the group (the checkbox is checked or not). A filter can be used to display only the groups to which the user belongs.

It is possible to associate or disassociate the user from groups by checking or unchecking the checkboxes in front of the desired groups. The association with a group is saved immediately.

"Administration" tab 

This tab is detailed in the Administration rights page.

"Modules" tab 

A tab is displayed for each module available in the administration.

Thus one can always find the "Pages" tab as it is the principle module in Automne, integrated in its core. We also find the "Forms" tab because it is the model furnished during the installation of Automne.

These tabs are detailed in the Assigning rights page.

Actions

- **Create a user:** Allows one to access the form to create a new user:
- **Modify a user:** Only when a user is selected: it permits one to open the user's profile for modification.

- **Deactivate a user:** Deactivate selected user: this user can no longer be authenticated. The user is not deleted and their profile can still be modified.
- **Delete a user:** The user's profile becomes inactive and no longer appears in the application's user list. Please note that the user is still present in the database because the profile may be linked to other resources. Nevertheless the user's login and password are empty and the user is marked as deleted.
- **Search for a user:** By default the first users are displayed, listed in alpabetical order. It is possible to search for users and filter the results by:
    - Keyword (last name, first name or login),
    - Letter (first letter of the last name),
    - Group (list of groups).

The results can be listed according to the properties of the users by clicking on the column title.

For example one can list the result by "First name".

User group management

Definition

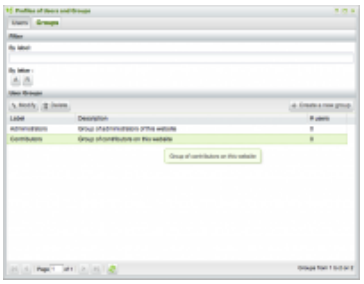
The user group management interface displays all the user groups registered by the application and allows one to manage their rights according to the different existing modules.

Some rules regarding user groups:

- Users can create user groups if and only if they have the "user management" right.
- A user group can contain none, one or many users.

Remarks:

- The user group interface looks a lot like the user management interface. Only the "group profile" tab is simplified compared to that of a user.
- When you modify the rights of a group, this immediately modifies the corresponding rights for all users affected, that is to say all users who belong to that group.



Properties

When a user group is selected, many tabs are available to modify its profile and rights:

- Group profile
- Users
- Administration
- Modules: a tab is displayed for each module available in the administration.

When a new user group is created only the "User profile" tab is active. Once the user group is created you can access all the other tabs.

"Group profile" tab 

- **Label:** Group name for identification. Choose an easily identifiable name, for example "Administrators" or

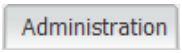
"All users."
- **Description :** Detailed description of the group: this can indicate the nature or purpose of the group.

"Users" tab 

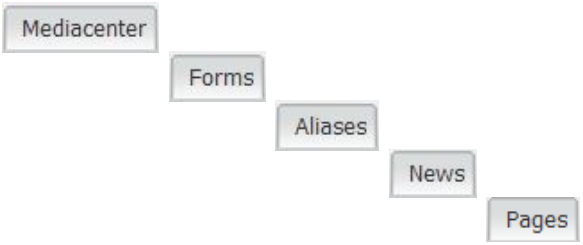
The list of existing users is presented in this tab.

For each user one can see if the user is associated (or not) with the group (the checkbox is checked or not). A filter allows one to display only the users who are associated with the currently selected group.

It is possible to associate or disassociate a user from the selected group by checking or unchecking the checkboxes in front of each user. The association with a group is saved immediately.

"Administration" tab 

This tab is detailed on the Administration rights page.

"Module" tab 

A tab is displayed for each module available in the administration.

Thus one can always find the "Pages" tab as it is the principle module in Automne, integrated in its core. We also find the "Forms" tab because it is the model furnished during the installation of Automne.

These tabs are detailed on the Assigning rights page.

Actions

- **Create a new group:** Allows one to access a form to create a new group.
- **Modifier:** Only when a group is selected: allows one to open a group profile for modification.
- **Delete a group:** Delete the group: the group will no longer appear in the list of user groups in the application. This action does not delete any of the users associated with this group.
- **Search for a user:** By default, the first groups are displayed, listed in alphabetical order. It is possible to filter groups by:
    - Key word (label)
    - Letter (first letter of the label)

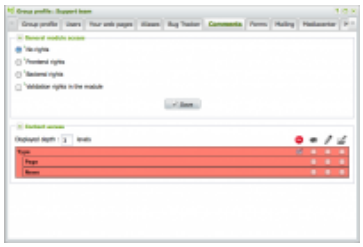
Results can be listed according to group properties by clicking the title of the column.

For example one can list results by the number of users associated with a group.

Assigning rights

The user management interface allows you to specify the rights for users and user groups.

Example of the modification of rights regarding the content of the "Comments" module:



Assigning rights

Unique user, without associated group(s):

The rights for a user without a group are unique to that user.

User belonging to only one group:

The rights for a user associated with only one group derive from the group.

> When a user belongs to one or more groups, their rights derive from their association with this (or these) group(s).

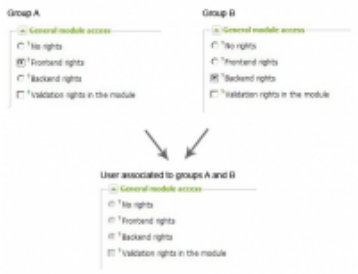
User belonging to one or more groups:

If a user belongs to more than one group, their rights are determined according to the different groups. The user rights are in this way the sum of the rights of the groups to which they belong.

Determining rights:

- **Openness takes precedence over closure:**

That is to say that if a user is associated with two groups, the rights of the group with greater access will be assigned.



- **The more specialized rights take precedence:**

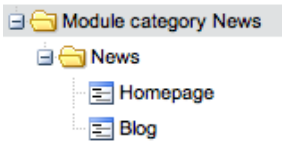
For the rights regarding categories or pages, it is always the right concerning the lowest level of the hierarchy which takes precedence.

> It is not possible to directly modify the rights of a user belonging to one or more groups: this information appears in grey. In this case you must modify the rights of each of these groups separately.
>
> The fact of not being able to edit the rights of a user belonging to several groups allows you to see clearly the rights derived from their association with these groups: by choosing the user we can thus see, in read-only mode, the rights he has. This can be useful if you want to verify the user's rights after modifying the rights of one of the groups to which they belong.

Example:

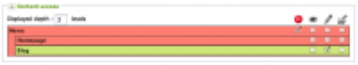
For a "News" module, or the tree of the following categories:

A root category News has 2 children : Homepage and Blog.

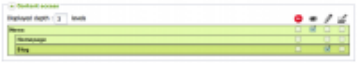


Group G1 defines the "See" right in the News category.

Group G2 defines the "Edit" right in the Blog category.



A user associated with group G1 and G2 has the following rights:



Explanations:

For each category Automne calculates the corresponding rights.

- If a right is defined, it retrieves the highest right.
- If no rights are defined, it retrieves the right of the parent category.

That gives us:

- News: "See" (the highest right defined by group G1)
- Welcome: "See" (no definition, thus passed on from the "See" right of the parent category)
- Blog : "Edit" (the highest right defined by group G1)

The different rights to assign

The different pages of this category detail the attribution of rights one can find with the tabs of the user management interface, after having selected a group.

- Module rights.
    - Page modules
    - Other modules
- Category rights.
- Administration rights.

Page rights

The "Pages" module is special in Automne because it is interdependent with its core.

Because of this it has certain properties that disinguish it from other modules.

Properties

Shared rights:

General access rights to content are the same as those described in the module rights section.

Access rights to content:

Management of page rights is identical to rights management for module categories. Consult this section for more detail about this kind of rights management.

Only a few differences are noted:

- The term "pages" is used instead of "categories".
- Page management is possible via the Automne page tree.
- There is no category management right.

> The "category management" right is not present here because the tree does not handle categories.
> We nevertheless find global management rights via the module access rights, already cited above.
> Administration rights permit access to page management.

Rights for pages models:

Rights for page models authorize their use according to the groups defined in the list.

Rights for row models:

Rights for row models authorize their use according to the groups defined in the list.

View of page-specific rights :

Module rights

Rights validation

Before looking at the definition of rights regarding modules, one must know that Autonme distinguishes between 2 cases in which rights are verified:

Administration interface

From the Automne administration interface, access rights are always verified because the user must be authenticated in order to access the administration. This includes access rights for both modules and content.

Public site

On the public site access rights are not verified unless the Automne parameter "Activate client-side rights verification" is activated. If it is not activated, no rights are verified and each visitor can see all the data in the module.

Definition

For each Automne module you can configure the following rights:

- General module access
- Access to content

  The "Page management" module integrated at the core of Automne has 2 additional notions not found in other modules:

  - Rights for page models
  - Rights for row models

Properties

For each module we find the following properties:

General module access:

- **No right:**

The user has no access to the module.
When the page is displayed the user will not see the content of the <block> tags (see content rows) concerning the polymod modules for which the user has no access.

- **Client-side consultation right:**

The user has access from the client-side of modules on the site, that is, the public side.

Access to the module from Automne administration is not authorized.

- **Administration right:**

The user has access from the client-side of modules on the site and from Automne administration.

- **Module validation right:**

The user can validate the elements of the module for which the user has editing rights.

Importance of rights:

The highest rights in descending order (validation rights being separate):

- Administration right (Automne administration)
- Consultation right (public part)
- No right (no access).

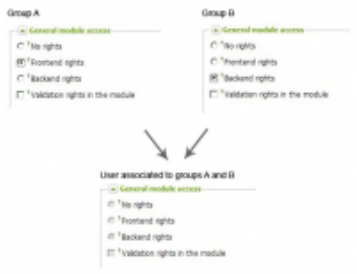
This information is taken into account when a user belongs to more than one group and these groups have access rights to different modules: the highest right is the only one taken into account.

> Note: If a user has administration rights for at least 1 module, this means the user will have access to the administration interface of Automne (yoursite/automne/admin/).

Example :

Module	Group A	Group B
General access rights for News module	No rights	Client-side consultation rights

- **If user X belongs to group A:** the user has no access rights for the News module
- **If user X belongs to group B:** the user has client-side consultation rights for the News module
- **If user X belongs to groups A and B:** the user has client-side consultation rights for the News module (we preserve the highest right from among the groups to which the user belongs.



Content access

For certain modules, supplementary rights are required in order to access elements: these are content access rights.

In Automne, content access rights are linked to the **categorization of elements**.

For the "Page management" module, the categorization is the page tree.
For all other modules, categorization is effected by categories: content access rights.thus concern the modules in which at least 1 element uses categories.

To be able to specify rights for certain elements, one must separate these elements into categories (that is to sat at least 1 "categories" field per element).
Similarly, if an element has at least 1 "categories" field, content access rights are automatically verified.

> For polymod modules it is easy to add a "categories" field. Using polymod allows for the automatic verification of rights.
>
> Traditional modules, developed manually, must be based upon manual rights verification, as much for module access as for content access (via categories).
> Automne categories allow one to realize this verification quite simply.

All content access rights are detailed on the category rights page.

Category rights

Definition

It is possible to attribute rights to the category tree of module elements.

For this the element must have at least one category field.

> The "Page management" module is not subject to this condition because for this module the page tree replaces the categories.

Properties

4 prinicipal rights are possible for each category:

- **No rights:** the user has no rights for the associated elements.
- **Viewing associated elements:** the user can see the associated elements.
- **Editing associated elements:** the user can edit the associated elements.
- **Categories management:** the user can edit the categories.

Here is the order of priority for rights, from weakest to strongest:

1. No rights
2. Visualization
3. Editing
4. Category management.

For example a user with editing rights will be able to edit *and* see the associated elements.

Actions

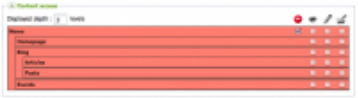
Allocations

The allocation of rights for each category follows the following rules:

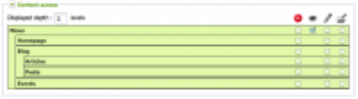
- It is only possible to allocate one right per category from among the 4 rights previously seen.
- When we specify a right for a category that has sub-categories, the latter inherit the specified right.
- For a sub-category is is not possible to define a right if it is different from the direct parent category.

Example of a category tree (News module):

Example: "No right" for the root category:



Example: "Visualization of associated elements" right for the root category:



Rights determination

This section explains how the category rights are determined:

For each category, here is the order of priority that rights are taken into account:

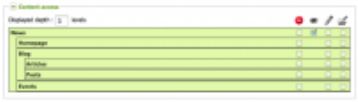
1. Position of the category in the tree
2. Priority of right (among the four presented previously)

Example :

The 'News' category is the root category of our 'News' module. The root category signifies that it does not have a parent.

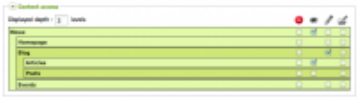
By default all categories rights are inherited from those of the parent category.

1. Here we give the "view the associated elements" right for the root category. All the child categories thus inherit the "view" right. It is impossible to redefine the "view" right for the direct children of the root category.

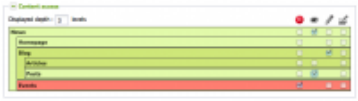


2. Nevertheless we specify that the "Blog" category has the "editing" right. From this fact the 'Articles' and 'Posts' categories inherit the editing right and we can again give them the "see" right. We can no longer give them the "editing" right as they have already inherited it.

3. One can specify the "view" right for the 'Articles' category.



4. Finally one can give "no rights" for the 'Events' category.



For a user associated with more than one group

User rights derive from those specified for the groups to which the user belongs..

The priorities indicated above are taken into account: if for the same category group A has the visualization right and group B has the editing right (thus higher), the resulting right is the editing right, conforming to the priorities.

Administration rights

Definition

Administration rights define the principle access to the administration interface of Automne. They are thus rights of the highest priority.

Properties

- **Administrator:** This is the most important right in Automne. It makes it possible to perform every action without constraints. This right is reserved for administrators of the highest level.
- **Regenerate pages:** This right permits one to regenerate pages manually and to controle background tasks.
- **Row management:** This right permits one to manage all the content ranges in Automne.
- **Model management:** This right permits one to manage all the page models in Automne.
- **Action log:** This right permits one to have access to the action log. It also allows one to purge this log.
- **User management:** This right permits one to have access to manage Automne users. It allows one to modify users and groups and thus, their access rights.
- **Branch duplication:** This right permits one to duplicate page branches.

  When a user has Administrator rights they are automatically associated with all the other administration rights and, more generally, has a status which authorizes all actions without limitations.

Activating client-side verification rights

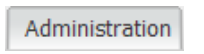
Definition

Automne distinguishes between sites with client-side verification rights and those which do not have this utility.

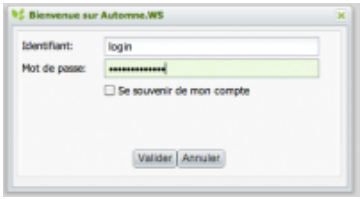
More simply you can distinguish two aspects:

- Visitors to your site can see all the published content (the case of a classic Internet site).
- Visitors to your site cannot see all the published content (in the case of an Intranet or Extranet, for example) and you would like to give people access only to certain information according to their rights.

This distinction is realized in Automne thanks to the "Activating client-side verification rights" parameter. This parameter can be configured via the Automne Parameters interface; the link can be found in the sidebar of Automne administration under the Administration tab.



We advise determining from the very beginning of your project if it will be necessary to distinguish users according to their rights, thus having an authentification form on your site. If this is the case the parameter "Activating client-side verification rights" must be activated.



Activating the parameter "Activating client-side verification rights" can significantly change the way you develop and administer your site.
It is therefore a decision which must be well-thought out and which developers must take into account.

The "Technical aspects" heading (below) details more precisely how Automne puts rights verification in place.

Sites without client-side verification rights

The parameter "Activating client-side verification rights" does not need to be activated.

On your public site every potential visitor to your site cannot be identified because the site does not have the means of authentification (an authentification form, for example).

Because of this Automne cannot distinguish users and thus rights are not verified. By default, each user thus has access to all the resources published on the site.

Sites with client-side verification rights

The parameter "Activating client-side verification rights" is activated.
On your site each visitor is now identified by default as the visitor "Public user" (Login "anonymous, ID 3).

As long as the visitor is not authenticated with their user account, they have the rights defined for "Public user".

Attention must be paid when defining "Public user" rights so that too large or too limited rights are not given to **non-authenticated users.**

> If you encounter probems accessing certain data on your site when you are not identified, you must verify the access rights for the user "Public user".

If a visitor is authenticated using an authentification form created with the help of the form module, the visitors will then have the rights assigned to the user's account.

Technical aspects

The parameter "Activating client-side verification rights" corresponds to the constant APPLICATION_ENFORCES_ACCESS_CONTROL.
When this constant returns true, the parameter is activated..

> Activating or disactivating this parameter requires one to regenerate the entire site so that cached pages are recreated with the code necessary to verify visitor rights.

While this parameter is active, every display linked to this page and every display of a module element entails verifying the rights corresponding to the current user.

This parameter thus has an impact on the general performance of your site; activating it is thus recommended only if there is a real need.

As for PHP, a user object (CMS_profile_user) is automatcally created via the variable $cms_user.

With the "system debugging" parameter activated, you can display the information for the current user and a webpage with this code:

```
<?php
pr($cms_user);
?>
```

To find out more:

For more information on this subject, don't hesitate to consult the forum.

Content validation

Content validation is the process of assuring that only pertinent and finalized information will be displayed to users.

It is possible to validate publications, pages and a set of modules. The elements to be validated are listed in the "Pending validation" section of the Automne menu.

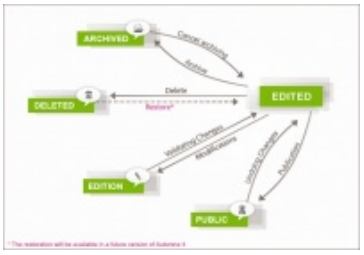
There are five states for the modules and pages in Automne:

- Archived,
- Deleted,
- Editing,
- Edited,
- Public.

It should be noted that non-standard modules have only three of these five states:

- Deleted,
- Edited,
- Public.

A module or page in the Edited state acts as the source for all actions except for the invalidation of a modification, the invalidation of an archive and the validation of a modification.



Publication

As the name says, publication is the act of putting pages or module elements in the front-end of your site so that they can be consulted by site visitors.

It is important to know that front-end designates a part accessible for reasons other than the Administration of Automne. That is to say that if your site has a member section accessible only by login ID and password, this section will be part of the front-end of the site because it is outside of Automne administration. To put it another way, the front-end is everything not found under the address YOUR_SITE/automne/admin.

An element may be published temporarily thanks to publication dates, which can be useful in the case of announcements of events, for example. Temporary publication can de done on every page and module element which is a pimary resource (that is to say directly subject to validation).

Automatic publication and unplublication :

To publish or unpublish content at a given date:

- Specify the publication dates.
- Validate the element.

For automatic publication and unpublication to work, you must add the following script to the cron (planned tasks) of your server:

0 0 * * * www-data php /racine_du_site/automne/classes/scripts/daily_routine.php

Here www-data is used to run Apache. This code causes the script /racine_du_site/automne/classes/scripts/daily_routine.php to run every night at midnight. More information on how cron works.

> This feature requires PHP CLI on your server. If CLI is not present, automatic publication and unpublication of content will be effected when a user connects to the administration interface of Automne.

Page validation

Page validation adheres to the same principles as the validation of a standard module.
While you modify a page it is in Editing mode and when finished it goes into Edited mode. It is only at this time that you can validate the publication of the page. In other words the modification cycle consists of 3 steps.

Modification > modification validation > publication validation.

Typically the validation of the modification is made by the person effectuating the modification, once they are sure the modification accords with his or her intentions. The validation of the publication is next effectuated by an administrator after making sure that the modification is appropriate for users of the site.

How it works

In the **"Validations pending"** section of the Automne menu, one can find listed all the archives / deletions / creations / modifications of the page to be validated.

This information is grouped by creation / modification on one hand and deletion / archive on the other. To see the actions in one group click on its name; you will then see a panel listing actions to validate at top and the validation options at the bottom.

For all pages you can choose to:

- **Accept the validation:** which authorizes the modification (passing from Edited to Public status).
- **Refuse the validation:** in this case the page is not modified and will be flagged as being in need of re-modification.
- **Transfer the validation:** the person to whom you choose to transfer the modification will make the decision to accept or not the modifications of the page. In this case, it is advisable to add a comment explaining the reasons for the transfer.

Whatever your decision, you can explain it with a comment which can also serve to follow the modifications.

To help you make a good decision you can:

- Preview the page,
- See the page online,
- Modify the page.

These three actions are available in the gap between the top and bottom part of the panel.

You can also choose to handle the validation of several pages of the same group at one time. For this, check the checkboxes corresponding to the pages to validate; next click **validate by lot.**
You can only validate a page lot; if you want to transfer or refuse the validation you must do it one page at a time.

On the validation panel you can at any moment change groups by selecting the type of validation in the "Validation types" rollover menu at the top of the window.

Module validation

Reminder of object types

Objects have a resource that determines how the validation of the object works.

Objects without resources have elements visible to visitors to the site from its inception; no validation is required.

Objects with a primary resource have elements which must be validated before being made available to visitors.

Objects with a secondary resource depend upon another object and their elements cannot be validated with the validation of the object (with a primary resource) to which it is linked.

For modules having categories, rights depend upon these categories; if they do not have categories rights depend directly on the module. Putting objects of a module online can only be done by a user having both validation rights and administration rights for that module.

Please note that when editing a page containing modules, their display is determined by the rights a user has concerning said modules. Thus if such a user has administration rights for the module they will see the content with Edited status; if not, they will see content with Public status.

How it works

Validation is the last step before making content available or removing an object from a site. The validation procedure is the same as for pages, with the notable exception of the following actions: preview, view online and modification, which are not available in the modules.

Please note that it is possible to validate objects from the object list of a module or to open the validation panel by clicking on the object status image and choosing "Validate" or "See the validation options."

Manage your modules and their content

It is possible to add a set of modules to the core of Automne in order to add features specific to the needs of each site.

By default Automne contains the most common modules:

- Library
- Update management
- Form creation
- Page alias creation

You are able, however, to add as many modules as you want.

Managing Polymod modules

What's a polymorphic module ?

The module generator called "Polymod" (for Polymorphic module) is an Automne module whose strength is to be highly configurable and evolutive.
It was created with the goal of responding to the most widespread needs in terms of structured data management without having to specifically use PHP code.

Context

Considering the following statement :

- The applications for managing data on the web respond to identical criteria in most cases. Developing each dynamic content management application independently amounts to taking a part of what exists and making some modifications to meet new demands.
- Considering the evolution of Automne, developing modules that take into account of all the features of the the core began to take longer and longer. At the same time, the possibility of unforeseen problems was becoming equally as great.
- The development of specific modules does not necessarily follow the developments of Automne's core features.

Here is the solution we found :

Aware of these inherent problems in the development model and guided by years of experience, we were able to define a configurable module model where the configuration capacity could respond to the greatest number of needs identified in terms of application management for dynamic data.

One module to rule them all

Applications such as:

- News and blog posts,
- Documents management (library),
- Photos management (gallery),
- Links management (blogroll),
- Directory,
- Schedule,
- etc.

have as their goal the management of data, essentially text, associated with files or images. This data can have a number of attributes: language, category, author, etc.

We can thus define a number of basic elements (imagine Lego bricks) that we can assemble in order to create a product responding to any given need.

This building blocks are as follows (non-exhaustive list):

- Short text field

- Long text field (formatted or not)
- Integer field
- Image field
- File field
- Category field
- Link field
- Date field
- Language field
- User field

Each of these fields, depending on the type, can then have several parameters (for example, for the date field one might need an hour as well as a date, or the date alone may be sufficient).In addition, the amount of all these elements will bluid an element which can itself become part of element of the largest scale.

Example

To simplify the understanding of these concepts, let us take an example: we want to model a library.

We can divide a library into aisles to classify books by certain criteria: cover, chapters, authors, subject, theme, etc.

The book is an element which also has distinct elements: cover, chapters, authors, subject, theme, etc.

We can next subdivide the chapters by elements: title and pages. Each page has a collection of paragraphs as well as a number.

The authors can also de described in a detailed fashion: first name, last name, nationality, biography, birthdate, address, etc.

In short, each element of our library can thus be subdivided until we arrive at the elementary level of information, and it is there that we find the building blocks of our content. These bricks are then put together to define more important structures, themselves assembled to define yet even more complex structures.

What is possible:

As we have just seen, the possibilities offered by such a system are quite vast, but for all of this, it is not possible to do everything. The module generator (Polymod) is capable of assembling elementary bricks of content into compex content structures but at this time, this is its "only" capability.
It is thus not currently possible to perform actions from these assemblages. We can take for example a newsletter. It is made up of a collection of content (which can be managed by Polymod) and this content is next sent by email to a group of recipients. This send action is not yet within reach of the system.

More precisely, it is possible to:

Administration side of Automne:

- Create and manage the content structures of each module.
- Insert content into these different structures.
- Search for this different content.
- Manage the access rights for modifying this content.

Public side of Automne (on websites managed by Automne):

- Create and manage a collection of rows to search for and display content.

- Create an RSS feed for different content.
- Manage the access rights for reading content.
- Create a plug-in for the visual editor (WYSIWYG) to use this content in the rows of different types of text pages.

To know more:

- Creating / modifying / deleting a module
- Creating / modifying / deleting an object
- Creating / modifying / deleting a field
- Creating / modifying / deleting a Wysiwyg plugin
- Creating / modifying / deleting an RSS flux

Tutorial :

- Example of creating a Polymod module

Creating / Modifying / Deleting a module

Definition

Creating a polymod module is quick and easy. In a few minutes you can define a new module which is completely configurable.

Define your objects and the fields which compose them. The administration interface will be automatically created for you and ready to use immediately.

In the "Applications management" interface, the polymod modules are listed in the selection menu "Application to edit". They appear in red, contrary to the specific modules which are in black.

Select a polymod module from the list, or click the "New" button to access to form to create or modify a Polymd module.

It is not currently possible to delete a polymod module.
Find out more on the subject in the Automne forum.

Properties

These are the form fields for creating and modifying a polymod module:

- **Label:** Corresponds to the name of the module. This label appears on the right administration sidebar. Example: the "News" label.
- **Identifier:** The unique lidentity of the module. You must here indicate a chain of characters different from other modules, without using special characters (alphanuleric characters only). This information will not be visible on the site; it is a code used in a database. In polymod, we advise prefixing this identifier with the letter "p". Example: the ientifier "pnews" for the News module.

  Be aware that the field is only available during the creation of a module. Once the module is created this property can no longer be modified.

- **Protecting the files to download:** If the box is checked, the files to be downloaded (PDF, images, etc.) will be filtered before each download. If rights are neccessary to see them, access will be denied to anyone without sufficient rights.
  The name of the file will be systematicallty cleaned of control data added by Automne. Activating this option involves a heavier load on the server for all downloads. Do not activate it unless there is a real need.

Actions

When a polymod module is created, form validation creates a database module.

The module will appear in the right sidebar and gives access to modify the properties and parameters of the moule

In order to have access to input elements for this module one must create an object having at least one field.

Creating / modifying / deleting an object

Définition

Un module est composé d'un ou plusieurs objets. Un « objet » est en fait la structure qui englobe les informations d'un élément. Ces informations sont structurées et définies avec des champs.

Création / modification

Lorsqu'un nouvel objet est créé et qu'il possède au moins 1 champ, il devient possible de créer de nouveaux éléments de cet objet.

Exemple : Pour le module « Actualités » on a un objet « Actualité » qui possède plusieurs champs (titre, catégorie, etc). Une fois cette structure mise en place, il est possible de créer des éléments « Actualité », chacun ayant des données spécifiques pour chaque champ : ce seront les différentes actualités affichées sur le site.

Le nombre d'objets par module n'est pas limité, cependant dans la pratique il est rare d'avoir plus de 5 objets pour un même module. Tout dépend de la structure de votre module.

> Notez qu'un objet appartenant à un module ne peut pas être utilisé par un objet appartenant à un module différent. Cette information doit être prise en compte avant le développement de vos rangées.

Toutefois, si un plugin WYSIWYG est associé à un objet d'un module, ce plugin peut être disponible pour tous les champs de type texte, et pour tous les modules polymod. Voir la documentation associée au plugin WYSIWYG.

Suppression

Pour supprimer un objet il faut que cet objet ne possède pas de champs. Si un objet possède déjà des champs, il faut donc les supprimer avant de supprimer l'objet lui-même.

En supprimant un objet :

- Vous rendez inaccessible l'ensemble des éléments qui ont été créés avec cet objet.
- Les champs d'autres objets qui possèdent des liaisons avec l'objet supprimé ne seront plus disponibles.
- Il faut supprimer toute référence à cet objet dans les rangées de vos pages, sans quoi les rangées pourraient appeler un objet inexistant et renvoyer des erreurs.

Propriétés

Formulaire de création

Lors de la création ou de de la modification d'un objet, un formulaire permet de renseigner les champs suivants :

- **Titre :** Libellé de l'objet. Cette information apparaîtra lors de la création des éléments de cet objet. Par exemple « Actualité ».
- **Description :** Permet de décrire l'objet plus en détail. Par exemple en décrivant sa fonction : pour « Actualité » nous pouvons indiquer « Représente une actualité publiée sur le site ».
- **Ressource :** Indique le type de ressource de l'objet : 3 choix sont disponibles :
    - **Ressource primaire :** L'objet sera soumis au processus de validation : il y aura donc plusieurs

statut pour chaque élément, notamment « en édition » et « publié ». Le changement de statut est soumis à validation.

A noter qu'1 seul objet par module peut posséder un type de ressource primaire.

- **Ressource secondaire :** L'objet dépend d'un objet en ressource primaire : en cas de modifications de l'objet en ressource secondaire les changements ne seront appliqués qu'une fois que son parent, en ressource primaire, aura été validé. Il peut y avoir plusieurs objets en ressource secondaire par module.
- **Aucune ressource :** L'objet n'est pas soumis au processus de validation : la notion de statut n'existe pas et toute modification d'un élément est prise en compte immédiatement.

- **Indexation :** Ce champ est uniquement disponible si le module « Moteur de recherche » (ASE, Automne Search Engine) est installé.
  - Si la case est cochée, l'objet sera indexé par le moteur de recherche.
    Attention : si l'objet en question appartient en tant que champ à un objet indexé (en tant que sous-objet d'un autre objet indexé), il est inutile de l'indexer.
  - La zone de texte « Adresse du lien vers l'objet » permet de définir une adresse dynamique qui sera utilisé par le lien dans la liste des résultats affichés par le moteur de recherche. Cette adresse est requise pour permettre de cliquer sur le résultat renvoyé par le moteur de recherche. Habituellement cette adresse pointe vers une page qui affiche le détail de l'élément.

Attention : les caractères spéciaux doivent être correctement encodés. Par exemple le « & » devient un « &».

A noter que l'indexation de l'objet est effectuée ici, mais il convient de définir également quels champs de l'objet seront indexés par le moteur de recherche, sans quoi le moteur ne saurait pas quelles données indexer. Voir la section Champs d'un objet polymod.

Formulaire de modification

Lors de la modification d'un objet les champs suivants sont disponibles, en plus de ceux précédemment cités :

- **Libellé composé :** Permet de définir un libellé construit dynamiquement à partir des données des champs de l'objet. Par exemple pour l'objet « Actualité » nous pourrions souhaiter indiquer la mention « Actualité » et la date de publication de l'élément. Dans ce cas nous indiquerions le libellé suivant : « Actualité {Actualite:formatedDateStart|d/m/Y}».
- **Adresse de prévisualisation :** permet de définir l'adresse vers laquelle pointent les liens de prévisualisation, disponibles lorsqu'on affiche la liste des éléments côté administration d'Automne. Par exemple : « {page:14:url}?id={Actualite:id} » indique que le lien pointera vers la page 14 en passant l'identifiant de l'élément en paramètre.
- **Visible sur l'accueil du module :** permet d'afficher ou non le lien vers l'administration des éléments côté administration. En laissant cette case décochée vous interdisez l'accès à la modification des éléments côté administration. Cette fonction peut être utile si vous ne souhaitez pas que certains utilisateurs accèdent à la modification de certains éléments.
- **Affichage des résultats côté admin :** permet de définir un affichage personnalisé des éléments lorsqu'on les affiche en liste, côté administration d'Automne.

Notez que l'interface d'administration possède un affichage par défaut, et se base sur les paramètres des champs de chaque objet : tous les champs dont le paramètre « Visible dans les résultats d'une recherche » est activé s'afficheront dans l'interface.

Résumé de l'objet

Lorsque l'objet est créé et sélectionné, un résumé de ses propriétés s'affiche :

- **Description :** Affiche la description de l'objet
- **Ressource :** Indique le type de ressource de l'objet (ressource primaire, secondaire, ou aucune ressource)
- **Visible sur l'accueil du module :** Indique si l'objet sera accessible depuis l'interface d'administration du module.
- **Libellé composé :** Indique si l'objet possède un libellé composé.
- **Utilisation de l'objet par d'autres objets :** Indique si cet objet est utilisé par un autre objet.

Structure de l'objet

Lorsque l'objet est créé, sélectionné, et possède au moins 1 champ, il devient possible d'afficher sa structure. Cliquez sur le bouton « Structure » :

La structure affiche tous les champs de l'objet de façon arborescente, de façon à détailler la composition de l'objet de manière exhaustive. Certains champs sont en effet constitués d'autres objets appartenant au même module. Voir la documentation sur les champs polymod.

Apparaissent en grisés les champs qui sont au 3ème niveau ou plus dans l'arborescence de l'objet. Ces champs ne seront pas accessibles lors d'une recherche simple sur l'objet. Pour des raisons de performances ces sous-objets ne sont pas chargés automatiquement.

Cependant il est possible de forcer le chargement automatique de ces champs : c'est un paramètre qu'il faut activer.

> Notez que cette fonction est rarement utilisée car elle peut causer de forts ralentissement dans le traitement des données. Voir la documentation sur les champs polymod.

Liste des champs de l'objet

Lorsqu'un objet est sélectionné tous ses champs sont affichés sous forme de tableau :

- **Titre :** Libellé du champ.
- **Type de données :** type de données, parmi ceux disponibles.
- **Description :** description textuelle du champ.
- **Actions :** Boutons pour supprimer ou modifier le champ.
- **Ordre :** Vous permet de modifier l'ordre d'affichage des champs avec un glisser-déposer. Lorsque vous déplacez un champ, le bouton « Sauvegarder l'ordre le nouvel ordre » apparaît à la fin de la liste des champs. Ce bouton enregistre l'ordre actuellement visible.

Actions

- Pour créer un objet cliquez sur « Nouveau ».
- Vous pouvez également choisir de modifier un objet existant en le sélectionnant puis en cliquant sur « Modifier ».

Fields Creation and modification

Definition

Object fields structure and define the information of the elements which will be created from these objects.

An object is thus constituted of one or more fields. The number of fields per object is not limited but it is better not to overload an object with too many fields, especially complex fields (for example, “multi-object” fields).

Example : For the “Recent Events” module there will be “News” objects that have several fields:

- Title (string field),
- Category (categories field)
- Description (text field with WYSIWYG editor).

Properties

Field editing form

The form for creating and modifying a field is composed of the following fields:

- **Datatype**: Defines the type of data in the field.
- **Title**: Specifies the title of the field. This label will be visible while editing an element.
- **Description**: Describes a field. The description generally complements the title of a field. The description is visible while editing an element; it can be found next to the title of the field.
- **Parameters**: Specifies the parameters for the field. Certain parameters are common to all fields, other are specific according to the type of field.

Groups of fields

We can organize the fields into 3 types:

- **Simple objects**: Fields which do not involve the use of other objects,
- **Compound objects**: Fields that use other objects. The value will be a unique element of these objects. The objects designated by a “Compound objects” field are usually called “sub-objects”,
- **Multiple compound objects**: Fields that use other objects. The value can be an element or several elements of these objects. Objects designated “multiple compound objects” are usually called “sub-objects”.

Parameters of fields

Common parameters:

Each type of field has parameters. Some are common to all fields:

- **Required field**: Defines whether or not the value of the field is necessary in order to validate the saving of an object,
- **Add a new search from** (or search by key-word in this field): Note that this field has 2 effects:
  - Searches a field in the admin interface of the elements,
  - If the object is defined so as not to be indexed by the “Search Engine” module (ASE, Automne Search Engine): during a client-side search for a search parameter of “keywords” the field will be searched (with the help of a “like %value%”).
- **Indexed in the search engine**: this field is only available if the object is defined as indexed by the “Search Engine” module (ASE, Automne Search Engine): If this is the case, the Search Engine will index the value of the field when it indexes objects,
- **Visible in the results of a search**: Displays the field and its value in the list of results of the elements admin interface.

Other parameters are specific to certain types of fields. We will detail these parameters by detailing the types of fields.

Parameters of “compound object” fields

All multiple compound object fields have the same parameters because it is a particular type of field.

Forcing sub-objects to load: This parameter forces the loading of sub-objects situated above the second level of the object tree.

Attention! For performance reasons this parameter is generally deactivated. Do not activate in unless data is missing during certain page loads and if the cause is known. Activating this parameter can lead to a significant loss of performance.

Parameters of *multiple compound objects* fields

All multiple compound object fields have the same parameters because it is a particular type of field.

- **These objects can be edited**: Specifies if the element can be edited directly on the form of their parent element.
  - **No**: The value will be selected from among a list of available elements, in the form of options in a selection box.
  - **Yes**: The value can be selected from among a list of available elements, in the form of options in a selection box. It will also be possible to modify a selected element or to create a new one. The forms will thus overlap.
- **Force sub-objects to load**: This parameter forces the loading of sub-objects situated above the second level of the object tree.

  Attention! For performance reasons this parameter is generally deactivated. Do not activate in unless data is missing during certain page loads and if the cause is known. Activating this parameter can lead to a significant loss of performance.

- **Deactivate the association of sub-objects**: This option will prevent the use of sub-elements created outside of the main objects. It is only useful if "these objects can be edited" is active. If this parameter is active it will only be possible to add a new element or to modify one already associated.

The different types of types of fields

Details of types of “simple object” fields

- **Boolean :** Specifies a state. (Yes – no).
  This type of data is usually used for every field where the value must be either 0 or 1. For example a “Visible public side” field defines if an element should be displayed or not. The value will be verified when searching for elements on the public side,

- **Categories**: Categorizes objects and manages their access rights. Specific parameters:
  - **Multiple categories**: Specifies if the field accepts a unique value (identifier of a category) or a multiple value (several identifiers or categories),
  - **Highest level category**: Specifies the highest-level category in order to display only its sub-categories in the selection menu,
  - **Default category**: Indicates which category will be selected by default if no category is chosen,
  - **Authorize the association of unused categories**: In the case of a row in which a search parameter has a “block” value, this indicates if the unused categories (those not associated with an element) must be displayed or not in the field value selection interface on the admin side,
  - **Width of selection box (pixels)**: Indicates the width of the field selection menu in a form. This parameter is only used in the case of a “multiple categories” field,
  - **Height of selection boxes (pixels)**: Indicates the height of the filed selection menu in a form. This parameter is only used in the case of a “multiple categories” field.
- **String**: String containing 225 characters maximum, without HTML. Specific parameters:
  - **Maximum number of characters**: Indicates the maximum number of characters for the field value. This number must be below 256 characters. Note that the number of characters is not limited by the SQL structure but controlled only in PHP,
  - **The value of the field must be a valid email address**: Indicates if the string indicated in the value must be a valid email address. If this is the case, a verification is performed when the value is saved,
  - **Format to respect**: Specifies a format to respect using a PERL regular expression. See format help.
- **Text field**: Long text field, with or without HTML. Specific parameters:
  - **HTML authorized**: Specifies if the field authorizes HTML or not. If yes, the WYSIWYG editor will be integrated into the input field,
  - **Type of text editor toolbar (WYSIWYG)**: If the file authorizes HTML, this defines which WYSIWYG toolbar to use in the input field. See the WYSIWYG toolbar section.

    A noter que ce paramètre permet l'utilisation de certains plugins spécifiques à Automne et intégré via la barre d'outils du WYSIWYG. Par exemple l'utilisation du plugin lié à la docuthèque qui permet d'intégrer des éléments « Documents » dans le champ de type texte,

  - **Width of the editor**: Specifies the width of the input field. Value indicated in pixels,
  - **Height of the editor**: Specifies the height of the input field. Value indicated in pixels.
- **Date**: Field containing the date in the format of the current language. Specific parameters:
  - **Save the current date**: If no value is indicated during input, the current date will be saved automatically,
  - **With hours-minutes-seconds management**: Defines if the format of the date will indicate hours, minutes and seconds,
  - **Date of object creation**: If activated, this parameter automatically saves the object creation date.

    Note that by “object creation date” we mean the first time the element input form is validated, when the date field is present in the form. In effect, if the date field is added to

objects after the element is created, the date saved will be the date when the form is again validated for this element and not the real object creation date.

- **Date of object update**: If activated, this parameter automatically saves the object modification date. That is to say the date which the data input form is validated,
- **Time Shift**: If “today’s date”, “creation date” or “update date” is selected, this parameter shifts the value by the indicated duration (See the format of the strtotime feature).

- **File**: Field containing a file. Specific parameters:
  - **Use a thumbnail for the file**: Saves a thumbnail which will be associated with the file. Authorized extensions: gif, jpg, png (this list of extensions is integrated into the core of the Polymod module and cannot currently be modified in administration),
  - **Maximum width of the thumbnail in pixels**: Used only if a thumbnail is specified. If the thumbnail exceeds this width it will be resized,
  - **Maximum height of the thumbnail in pixels**: Used only if a thumbnail is specified. If the thumbnail exceeds this height it will be resized,
  - A**uthorize using files from FTP directory**: Uses an Automne directory in order to transfer large files via FTP),
  - **FTP directory path to use**: Specifies the path from which the files will be recovered. Used only if the use of an FTP directory is specified. You must indicate an empty path to prevent using an FTP directory as a source for documents,
  - **Authorized extensions**: Indicates the extensions that are authorized to load on the server: these extensions, separated by a comma, are considered as a whitelist. Note that if the “Prohibited extensions” parameter is defined, the corresponding extensions will be verified as well as these,
  - **Prohibited extensions**: Indicates the extensions which are not authorized to load on the server: these extensions, separated by a comma, are considered as a blacklist. Note that if the “Authorized extensions” parameter is defined, the corresponding extensions will be verified as well as these,

    By default the following extensions are prohibited: exe, php, pif, vbs, bat, com, scr, reg

    Attention: do not modify this list unless really necessary; there can be serious consequences for site and server security.

- **Image**: Field containing an image. Specific parameters:
  - **Maximum width of the thumbnail in pixels**: Used only if a thumbnail is specified. If the thumbnail exceeds this width it will be resized,
  - **Maximum height of the thumbnail in pixels**: Used only if a thumbnail is specified. If the thumbnail exceeds this height it will be resized,
  - **Use a separate zoom image**: Loads a zoom image in addition to the original,
  - **Use the original thumbnail for the zoom image**: Used only if the “Use a separate zoom image” is not activated. When this parameter is activated the original image loaded on the server is conserved as the zoom image,
  - **Length of the thumbnail in search results**: Used only if the field is visible in the results of an admin side search. This parameter defines the length of the image in the list of results. The length must be indicated in pixels. By default the length is 16px.
- **Language**: Language of the object: Establishes a relationship with the languages available in the

application. If the object is referenced for the “Search Engine” module (ASE or Automne Search Engine), this field is needed for correct indexing. For each element with a “Language” field, the Search Engine verifies the defined language and adapts to the words stemming from this language, if it is supported,

- **Link**: Field containing a link. This field accepts different kinds of links:
  - **Internal** (towards an Automne page of the current site),
  - **External** (to another site),
  - To a file (loads a file on the server). Note that this type of field blocks by default the following extensions: exe, php, pif, vbs, bat, com, scr, reg.

    This list is defined by the PHP constant: php FILE_UPLOAD_EXTENSIONS_DENIED (defined in /cms_rc.php)

    Each link can be sent to different destination:

    - Current window,
    - New window,
    - Pop-up window.
- **Integer**: Integer of 11 figures maximum.

  Specific parameters:

  - **Can be zero**: Specifies if the value can be zero, or not.
  - **Can be negative**: Specifies if the value can be negative, or not.
  - **Unit**: Specifies the value unit. The indicated unit will be displayed next to the value when the field is displayed. This parameter can also be recovered by a personalized display. For example: euros, meters, m², minutes, etc.
- **Floating number**: String containing a floating point number with 255 characters maximum. Specific parameters:
  - **Maximum number of characters**: Specifies the maximum number of characters authorized for the field value. This number cannot exceed 255 characters.
  - **The field value can be a negative number**: Specifies if the value can be zero, or not.
  - **Format to respect**: Specifies a format to respect using a PERL regular expression. See format help .
  - **Unit**: Specifies the value unit. The indicated unit will be displayed next to the value when the field is displayed. This parameter can also be recovered by a personalized display. For example: euros, meters, m², minutes, etc.
- **Notification by email**: This field sends an email notification when validating an element. If this field is part of an object, the validation of each element of the object sends one or more emails according to the field parameters. Specific parameters:
  - **Email subject**: Defines the subject of the email sent. The value must be a string of characters.
  - **Email body**: Defines the body of the email sent. The value can either be an HTML text (where it is possible to use tags specific to the module: see help available for the field), or an Automne page.
  - **Send choice**: When editing objects, chooses if the email is sent, or not.
  - **Send**: Indicates when emails are sent.
    - **At validation**: The email will be sent automatically at object validation
    - **System event**: The email will be sent by a specified system event. In this case the developer must put in place a PHP code specifically to send mails.
  - **Include files**: Includes the files of objects as an email attachment.

- **Email sender**: Specifies an address for the email sender. If no value is indicated, the “postmaster” address will be used.
- **Inclusion**: Defines the users or groups who will be included or excluded from the email notification.
  - **yes**: the selected users/groups will receive notifications
  - **no**: the selected users/groups will not receive notifications.

- **Page**: Chooses an Automne page. The value corresponds to the identifier of an Automne page. A link to the site map is proposed to the user allowing him/her to select the desired page.
- **User/Group**: Associates one or more users or groups with the element. Specific parameters:

Note that if this parameter is used it excludes the others.

- **The value is the current user**: Specifies a format to respect using a PERL regular expression. See format help.
- **User groups**: If this parameter is selected the value will correspond to user groups. If not, it will be users.
- **Multiple users or groups**: If this parameter is activated, the choice can be multiple. The value will be a table of identifiers (users, or user groups).
- **Inclusion**: Defines the users or groups who will be included or excluded from the email notification.
  - **yes**: the selected users/groups will receive notifications
  - **no**: the selected users/groups will not receive notifications.
- **User creating the object**: Specifies a format to respect using a PERL regular expression. See format help.

Inheriting fields

Technically, each field is inherited from a common base, thereby extending its properties. All fields inherit the common field but are sometimes made up of multiple types. For example an email field is made up of three sub-fields: integer, date and integer.

Boolean:

- Integer (int)

Category :

- Integer (int)

Date :

- Date (date)

Email :

- Integer (int)
- Date (date)
- Integer (int)

File :

- String (string)
- String (string)
- String (string)
- Entier (int)
- String (string)

Flottant :

- Integer (int)
- Booléen
- String (string)
- String (string)

Link :

- String (string)

Image :

- String (string)
- String (string)
- String (string)

Integer :

- Integer (int)

Language :

- String (string)

Page :

- Integer (int)

String :

- String (string)

Text :

- Text (text)

User / group :

- Integer (int)

Knowing the sub-fields can be useful in order to know how searches are done in these fields: By default a tag will search according to the value of each sub-field. See XML definition of polymod rows.

Actions

- To add a field, click on the “New” button,
- Can also modify each field with the “Modify” button,
- To delete a field use the “Delete” button.
- Attention: this action deletes a field and should be done with caution; some rows may use this field and will not work if the field is deleted. Delete a field only if you are certain that the action will not have undesired consequences.

- The order of fields can be modified. One can simply drag and drop a file to the desired location. The order of fields is generally used for the editing form of an element; the fields are thus displayed according to the order defined therein.

WYSIWYG plugin creation/edition

Definition

WYSIWYG plugin feature

A WYSIWYG plugin adds a button in the WYSIWYG toolbar for an HTML row. It allows you to insert a link from an item stored in a module. It can be a text link or an image link.

For instance, you can insert an image from the media module via WYSIWYG:

Inserting into text can be of two types :

- Insertion of HTML around a selected text in the WYSIWYG.

  Example: adding a link to text to a document managed by a module or to a page displaying the detail of a module element.

- Insertion of HTML in any given text at the position of the cursor.

  Example: adding HTML code describing a given module element such as news, a data sheet, a document and its metadata, etc.

  It is possible to create several WYSIWYG plugins for each of the objects created with the Polymod module generator: there is no limit.

Two important points :

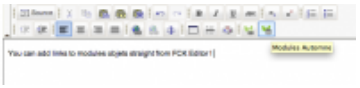
- WYSIWYG modules only permit inserting existing data from a module into a text (they do not have as an objective the creation or modification of data).
- The elements inserted into a text follows the evolution of the source of the element in the module. For example, a document inserted into a text will be dynamically modified if the source document is modified in the module. In the same manner, it will be deleted if the source document is deleted.

How to insert a WYSIWYG plugin into a WYSIWYG toolbar

WYSIWYG modules only permit inserting existing data from a module into a text (they do not have as an objective the creation or modification of data). (see WYSIWYG toolbar models).

For example, if there is a WYSIWYG plugin for a “Recent Events” object in the “Recent Events” module, it is possible to edit the “Default” WYSIWYG toolbar by adding a “Recent Events” button.

Adding this button to the toolbar allows one to use the WYSIWYG plugins of the polymod module concerned with the HTML text fields using this toolbar.

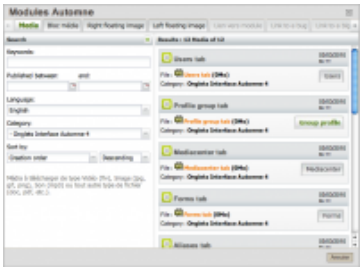


Access to the button of the polymod module in the toolbar is submitted to the content access rights for this module.

While editing of a field for HTML text (contained in a text row or even in another polymod module) with WYSIWYG toolbar or if you have the polymod button concerned, you can use this button to access the interface for selecting and inserting elements.

Display elements can be configured when creating the WYSIWYG plugin.

In this way one can limit the number of results, display the elements of a specific category, arrange the results in ascending order for particular data, etc.



Properties

When editing a WYSIWYG plugin, the following fields are available :

Title :

Gives a title to a WYSIWYG plugin. This title will be used in the toolbar to select the desired plugin, according to the polymod module.

Description :

Indicates a complementary description for a WYSIWYG plugin: generally the description indicates the nature of the plugin and its functions. For example “Displays the title of a recent event with a link to details”.

Only objects that meet these parameters :

Filters the elements which will be displayed as available when selecting elements in the WYSIWYG plugin interface.

Définition XML :

Defines the information to display in the text feild when a polymod element is inserted. It is the heart of the plugin: when the user has selected their element in the interface of element selection and validates the insertion in the field, the displayed result in the text field is defined by the XML definition (see detail below.)

For example one can choose to display only the title of the element, or the totality of fields or even an image with a link, etc....The are many possiblities. See the help available below the field.

XML definition of a WYSIWYG plugin:

Voila une explication sur la syntaxe à employer pour les modules WYSIWYG.

> Here is an explanation of the syntax to use for WYSIWYG modules. You will find dynamic help in the WYSIWYG plugin creation interface.

```
<atm-plugin language="languageCode">
    <atm-plugin-valid>
        ...
    </atm-plugin-valid>
    <atm-plugin-invalid>
        ...
    </atm-plugin-invalid>
    <atm-plugin-view>
        ...
    </atm-plugin-view>
 </atm-plugin>
```

The **atm-plugin-valid** tag will be read if the selected object is valid (not deleted, validated and being published and if the user has consultation rights).

The **atm-plugin-invalid** tag (facultative) will be read if the object selected is invalid (deleted, not validated or if the publications dates have expired and if the user has consultation rights for this object).

The **atm-plugin-view** tag (facultative) will replace the atm-plugin-valid tag in the visual text editor (WYSIWYG). It is principally used to display a simplified version of data and thus facilitate content modification in the editor.

languageCode : Language code relative to the content from among the following codes, among the codes available for your application (by default "fr" and "en").

{plugin:selection} : Will be replaced for the textual value selected in the editor (facultative).

Examples for the media library module, included in the Automne demo

Displays a link to a selected document using the title of the document

```
 <atm-plugin language="en">
    <atm-plugin-valid>
       <a href="{Media:File:filePath}/{Media:File:filename}" target="_blank" title="Download document
'{Media:File:fileLabel}' ({Media:File:fileExtension} - {Media:File:fileSize}MB)"><atm-if
what="{Media:File:fileIcon}"><img src="{Media:File:fileIcon}" alt="File {Media:File:fileExtension}" title="File
{Media:File:fileExtension}" />
    </atm-plugin-valid>
 </atm-plugin>
```

This code will display a link to the document containing the icon of the document type and document name.

> If the document no longer exists or is unavailable to the user, there will be nothing displayed because the tag **<atm-plugin-invalid>** is not specified.

Displays a link to a selected document on a selected text in WYSIWYG :

```
<atm-plugin language="en">
```

```
    <atm-plugin-valid>
        <a href="{Media:File:filePath}/{Media:Ficle:filename}" target="_blank" title="Download file
'{Media:File:fileLabel}' ({Media:File:fileExtension} - {Media:File:fileSize}Mo)"><atm-if
what="{Media:File:fileIcon}"><img src="{Media:File:fileIcon}" alt="File {Media:File:fileExtension}" title="File
{Media:File:fileExtension}" />
    </atm-plugin-valid>
    <atm-plugin-invalid>
        {plugin:selection}
    </atm-plugin-invalid>
 </atm-plugin>
```

This code will display a link to the document containing the document type icon selected text in the WYSIWYG

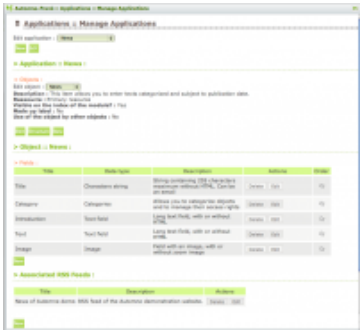
> If the document no longer exists or is unavailable to the user, only the text selected in the WYSIWYG will be displayed without a link (the tag atm-plugin-invalid only contains {plugin:selection} which represents the selected text in WYSIWYG).

Creating / Modifying an RSS feed

Definition

Creating an RSS feed for a ploymod object permits generating an RSS feed containing an ensemble of data. Generally it displays a list of the last elements published.

In order to edit an RSS feed, select an object from the module management interface, then click "new" or "modify".



It is possible to create several RSS feeds for each object created with the Polymod module generator; their number is not limited.

Properties

When editing an RSS feed, the following fields are available:

Title :

Permits giving a name to an RSS feed. This permits one to indicate the feed among all the available feeds for a polymod object.

Description :

Permits describing the RSS feed more precisely. Often, each flux has a different display; the description can be

useful for example to indicate the display which corresponds to the feed.

Address for the site feed:

An RSS feed can be used outside of your application, therefore if a user wants to access the source information, the user generally clicks on a link present on an element of the feed. This link is defined by the field "Address for the site feed."
By default the site corresponds to the address of the current site.

Author:

Permits indicating the author of the RSS feed.
You can for example indicate the name of your site or the name of the person responsible for the feed.

Email the author

Permits indicating the email of the author so that users can contact him or her if necessary.

Copyright :

Permits specifying a copyright.

Categories :

List of terms, separated by commas, permitting one to categorize the RSS feed.

Update interval and frequency:

Permits readers of the RSS feed to to have a value indicating how often the thread is updated.
By default: Once per day, minimum: twice per hour

XML Definition :

Permits defining the information to display on the RSS feed.

It is the heart of the RSS feed: when the user accesses the feed from a link or feed reeder, the information that it recovers is the result of this XML definition.

For example one can choose to display the list of latest published news.
For this it is sufficient to effect a search for the News object, then order the results by date in reverse chronological order.

It is common to put a link to a page detailing the news, in case the reader wants to know more.

Dynamic help is available on the RRS feed editing page in order to create a feed simply and more easily.

Example of XML definition:

```
<atm-rss language="en">
   <atm-search what="{Blog}" name="rss">
      <atm-search-order search="rss" type="publication date after" direction="desc" />
      <atm-result search="rss">
         <atm-rss-item>
            <atm-rss-item-url>{page:5:url}?item={Blog:id}</atm-rss-item-url>
```

```
            <atm-rss-item-title>{Blog:Title:value}</atm-rss-item-title>
            <atm-rss-item-content>
               {Blog:Introduction:htmlvalue}
               <atm-if what="{Blog:Text:value}">
                  <a href="{page:5:url}?item={Blog:id}" title="Read more about '{Blog:label}'">Read more</a>
               </atm-if>
            </atm-rss-item-content>
            <atm-rss-item-date>{Blog:formatedDateStart|rss}</atm-rss-item-date>
         </atm-rss-item>
      </atm-result>
   </atm-search>
</atm-rss>
```

Use:

Once an RSS feed has been created, it is interessing to create a link to use and/or to indicate to users that the feed exists. This allows them to subscribe.

The link has the following form:

http://yourWebSite/rss/rss.php?id=RSSFeedId
The "id" parameter indicated in the URL defines which feed the rss.php will display.

You will find a list of identifiers available for an object in the dynamic help for this object: consult the help for this object (from any row) and click on the object to display the help for the RSS fed.

For example, for a "News" object, go to any row, then to "XML Defenition", then click on "Help" and finally on the tab corresponding to the "News" module.
Then select the "News" object from the dropdown list. A dynamic help is displayed and one can see under the heading "Object features:" the list of RSS feeds available along with their respective identifiers.

Creating a link to an RSS feed

The link to a file rss.php is usually integrated into a polymod row using the following tag:

```
<atm-function function="rss" object="{News}" selected="rssId" attributeName="attributeValue">
   <a href="{url}" title="{description}">{label}</a>
</atm-function>
```

For an example, here is the link to the RSS feed for the Automne site: Read all the items

Media center module

The media center module permits you to store different media (image, video, sound...) in order to reuse them easily in your Automne pages. It is especially useful if you plan on using the same media in several pages.

Much more efficient than previous versions, it allows one to classify media by categories and offers the possibility of creating new categories according to your needs. In addition, it has a built-in, clear and and practical search engine in order to find media in one click.

Accessible directly from the row content editor, you can use your media everywhere and as much as you want, without limits.

Managing media elements

From this interface you can create, modify or delete your media.

Title

Corresponding to the title of the media, it is displayed in the media list. Choose with care for it will be indispensable for finding your media when it is inserted into a page.

Description

A detailed description of the media, it is used to research by key words.

Category

Corresponding to the type of media, you can create new categories, as explained here.

File

Corresponds to the media itself.

Label

Representing the name of the media, it is displayed in the overview of the thumbnail or thumbnail text (attribute alt).

Thumbnail

Corresponds to the miniature image of your media.

Source file

Corresponds to the media itself.

Publication dates

These dates define the display restrictions of the media on your site. If the current date is included in the range of dates indicated in these fields, your media will be visible on the site; if not, it will be automatically unpublished and disappear from the display.

> The end date of publication is not obligatory; leave it empty if you want your media to continue to be visible indefinitely.

Category management

The categories of the media library allow you to organize the different media by type, which in turn allows you to find them more easily.

To create a category, go to the administration column on the right and find "Media library" then "Category management" or click on the Categories tab from category management.

You can create as many categories as you want; you can even create subcategories by sliding one category over another.

> Media can only belong to one category. Be careful to make sure that the proposed categories are distinct from one another.

Wysiwyg plugin

The Wysiwyg plugins allow you to make links between the module and page content; in this way they allow you to choose what media to display.

Even if the procedure is similar, you have two ways of inserting media into a content zone, according to the appearance desired.

You must insert media with its thumbnail: click on the edit icon of your content zone and then click on the Automne Modules icon. You will in this way access your media list; select what you want to insert and click validate. The thumbnail will thus be a link to the media which you want to appear.

You want to insert media in the form of a text link: click on the edit icon of your content zone, select a text then click on the Automne Modules icon. You will in this way access your media list; select what you want to insert and click validate. A link to your media will appear on the text previously selected.

The Recent Activity module

Definition

The Recent Activity module is a Polymod module that permits managing the news of your site (by news we mean content displayed according to its publication date and being replaced automatically by more recent content). By default, the Recent Activity module allows one to input a title; two texts (an introduction and a more complete text); to select a category for your news and to illustrate it with an image. As a Polymod module you can extend the available fields in order to make it conform to your intentions.

The Recent Activity module uses two rows, with one placed in the homepage of the Automne demonstration site to display the latest news. The other is placed on an interior page and lists all the site news in reverse order of publication.

Managing Recent Activity elements

To access Recent Activity management, go the administration column on the right, go to "Recent Activity" then "Manage 'Recent Activity' elements".

When you create/modify/search for your news, this is likewise true of their categories.

As stated above, news is composed of a title, a category, an introduction, a text and possibly an image.

If you want to attract your visitors' attention and encourage them to click on your news, an image is almost obligatory.

In order to enhance your news items' appearance and encourage visitors to click on them, strict attention must be paid to their titles and introductions. These, in effect, determine if your visitors will look inside.
Don't hesitate to use an advanced format for your texts (the introduction as well as the text of the news itself).
Well-formatted information with a come-hither look easy on the eyes makes its point better than a frumpy spinster.

Like many elements, news can have a limited publication duration by limiting access to it for a predefined period.
For this it is enough to modify the dates from the Begin publication date to the End publication date of your news.
By default the publication date is its creation date and does not have an end date, which means it will remain available until the end of time. Of course, you can modify these values at any moment while editing the news.

Category management

The recent events categories allow you to thematically arrange your different recent events, which in turn allows you to find them more easily; this is as valuable for you as it is for your visitors.

To create a category, in the administation column on the right, go to "recent events" then "category management" or click on the Categories tab for category management.

You can create as many categories as you want; you can even create sub-categories (by sliding one category over another).

Nevetheless it must be remembered that a news item can only belong to one category. Attention must be paid so that all categories are distinct from one another and that this distinction is as clear for you as it is your users.

RSS Feed

By default the Recent Activity module of Automne has an RSS feed which lists the five last actions on your site. (To know how an RSS feed works in Automne consult the RSS feed section of the manual). The RSS feed, especially when it acts as a recent activity feed, is a very important element for client relations and a way of assuring that the people who interest you are well-informed about your updates. You will also undoubtedly want to add links to your feed from other pages or even create a new feed (one feed per category of news, for example).

To create a new feed in the Recent Activity module, click on the serrated wheel in the upper-right of the module box or go to Administration ? Module management. In "Edit application" select Recent activity and then in "Edit object" select Recent activity. In "Associated RSS feeds" next click on "New".

For more information on creating RSS feeds consult the appropriate section of the manual.

The forms module

The forms module allows one to generate and to set parameters for actions particular to forms, such as:

- Sending a message,
- Surveys,
- Identifying a user.

The forms creation assistant allows you to create complex forms with no specialized know-how.

To add a form to one of your pages, you only need to add a form row and to select the desired form by editing the row.

Managing form elements

On the "Form" menu on the sidebar you have access to:

- The management of form elements,
- To the module categories,
- To the module parameters (by clicking on the icon to the right of the tab).

Module parameters

The parameters correspond to general preferences:

- The default language when you manage form elements. You must input a valid language code (fr or en),
- The name of the wysiwyg toolbar used for editing the form. The default toolbar "cms_forms" contains tools management for the forms creation assistant, which is essential for the creation of forms.

Module categories

These permit organizing or refining your rights for different forms. Managing form elements permits accessing the ensemble of existing forms and filtering them by lanagauge and/or categories.

Creating a new form

Click on "New", taking care that you are in the language in which you want to create the form (for this click on the radio button for the desired language).

In the title field, specify a name that allows you to identify your form. This label will appear in the editing row when you want to add a form to one of your pages.

You can next choose to render a form active or not active. An inactive form is not visible immediately (thus pointless to refresh the page).

In the source field, you can create your form. For this click on the form assistant icon.

This assistant permits adding several kinds of fields. It also plays an essential role in generating unique form tags saved in a database.

The different kinds of fields you can add are:

- **Text field:** text zone on one line. You can edit the default value for the field.
- **Email:** text zone on one line. A control of the vaildity of the email will be effected at the submission of the form. You can edit the dafault value for the field.
- **Numbers:** text zone on one line. A control of the validity of characters will be effected at the submission of the form. You can edit the default value for the field.
- **URL:** text zone on one line. A control of the validity of the URL will be effected at the submission of the form. You can edit the default value for the field.
- **Password:** password text zone. Stars appear in place of the characters input by the user. You can edit the default value for the field.
- **Attachments:** field allowing the user to choose a file on his or her computer. This file will be attached in

the case of in the case of a form which involves sending an email. You can edit the default value for the field.
- **Text zone:** multi-line text zone. You can edit the default value for the field.
- **Multiple choice:** create a field of selection values. You can edit the values for the field.
- **Check box:** for creating check boxes. You can edit the default value for the field. (0 or 1)
- **Hidden field:** permits inserting a field invisible to the user (to transmit data at the submission of a form with an eye towards handling them later).
- **Validate button:** at least one field of this sort is necessary per form so that the user can submit the form.

To add a field, choose the type of field and click on "Add".

The field appears in the area above. For each of these fields you can specify:

- If it is obligatory or optional.
- Its label. If you make it obligatory, please indicate this to the user with a * before the label, for example. XHTML is supported in this field; it will be encapsulated within a "label" tag linked to the field.
- In options, you can, according to the case, modify the default value by clicking on "Default", or modify available values by clicking on "Values".
- In the actions column you can choose to delete a field or to modify its order relative to other fields.
- Click on "Validate" to confirm field creation.

Modifiying the default value:

Click on "Default", input the value and click on "Return to the form".

Attention, if you click directly on "OK" you will validate the field structure and the actions performed previously in the assistant, but not the value just immediately input.

You can enter a value for the general variables, respecting the syntax described.

Modifying the values:

- Click on **"Values"**,
- In the **"Text"** field, input the text visible to the user,
- In the **"Value"** field, input the value trasmitted at the submission of the form,
- Click on **"Add"** to add the couple "text/value" to the list of values.

In the values field, you can modify the order of values in the list and choose one to be selected by default.

Click on "Return to the form" to validate your input.

Attention, if you click directly on "OK" you will validate the field structure and the actions performed previously in the assistant, but not the value just immediately input.

Once your field has been created, you can modify the format using classic wysiwyg tools.

Finally, select the form categories, limit the number of user responses and validate. You will see your form appear in the list of existing forms.

Modifying a form

To modify a form you can:

- **Modify all the fields at once:** For this click on the form assistant icon. Be careful, however, as the

validation of field creation reinitializes the form in the wysiwyg and you lose your formatting.

- **Modify only one field.** For this, right-click on the field or its label. In the context menu, click on "Modify the field".
- **Add a field.** For this, right-click on the desired placement. In the context menu click on "Add a field".

Form actions

What is a form action?

A form action corresponds to handling data input by the user. At the moment a user validates a form, the input information is collected and handled according to actions you can define.

The form module makes features for handling data available which permit you to manage input data without any development knowledge.

Modifying and adding form actions

To modify form actions, the succession of treatments performed when a form is submitted, click on "Form actions" for the form created.

Several actions are possible and their parameters modifiable.

By default, many types of actions are available; they can be used freely, or not.

Please validate each type of action individually.

You can add actions and combine existing ones. Sending an email can be repeated as many times as necessay.

In a text zone, you can insert general variables respecting the syntax described in the actions window.

One action can authenticate a user. Within a site with restricted space, you can choose to create a form which will authenticate a user and allow him to access a certain part of your site.

This requires:

- activating the Automne parameters option which stipulates that your site uses user authentification. This will make it possible to the verify current user's rights regarding the element he or she is trying to see, on all the pages of the site.
- having defined the users and their rights. In particular, verify the rights of the "Public user", which correspond to an unidentified user. Only users having an Automne account can be authenticated through your form.

Finally, your form must contain at least one field alllowing one to login (Text field) and to input a password (Password field). You can also allow the user to remember their account, in which case you must add a boolean field to your form.

These fields must be next identified in the form actions. Automne lists the fields which correspond to the criteria of the field type for each of the fields necessary to authenticate the user.

Implementation in Automne

Adding a form to a page generates the file /automne/templates/mod_cms_forms_header.php.; it is included when the page is refreshed. This file will be handled first, before the page which includes the form is displayed.
This file contains all the treatment data desired and defined in the form actions.

The forms generated by this module use the POST method to send data and the file brought up by the action tag of the current page.

Exporting form data

As soon as forms are submitted online, you can export their content in .csv by clicking on the link "Download form dat in .CSV format".

Category management

Form categories make it possible to:

- organize your forms
- assign rights for their use and/or their administration.

You can create as many categories as you want. You can even create subcategories by sliding one category over another.

To create a category, go the the administration column on the right; go to "Forms", then "Category management" or click on the Categories tab from form management.

The Alias module

Module definition

The alias module allows quick access to a section of your site. For example, in the form of a letter or a survey.... If you want to give a particular address on your site without furnishing the entire URL, you only need to create an alias for this address.

Your alias will have the form: http://www.yourdomain.com/survey/ and point directly to a page on your site, or to an address of your choice.

To access the list of existing aliases, click on the side panel on "Alias/Manage Alias elements".

Creation/Modification of an alias

Click on "New".
In the "Alias" field type the name of your alias. In the example http://yourdomain.com/survey/, the alias is survey.

Specify the target and validate.

If more than one alias is created on the same path, only the last created will be taken into account.

Deleting an alias

Click on the "Delete" button in the list of your existing aliases.

Uses who bookmark or attempt to visit a deleted alias will be directed to a 404 page (Page not found).

Administration

The Administration of Automne includes a number of features requiring special rights in order to be accessible. The following is a list of features linked to the administration of Automne:

- Server parameters
- Automne parameters
- Multisite management
- Database
- Stylesheet management
- Javascript management
- Script management
- Wysiwyg toolbar
- Action log
- Filesystem

Server parameters

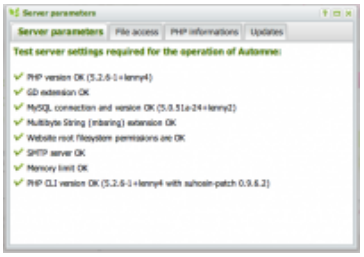
The server parameters window (which is opened via Administration > Server Parameters) is divided in 4 tabs.

Each tab allow permits managing or obtaining information about the key elements of the server; these are, respectively:

- Information about the availability of Auotmne prerequisites (Server parameters).
- Access to files for Automne (Access to files)
- The PHP installation information (PHP information)
- Automne update management (Updates)

Server parameters

The parameters required for Automne are listed here; all these parameters must be checked "OK" to assure the optimal operation of Automne.



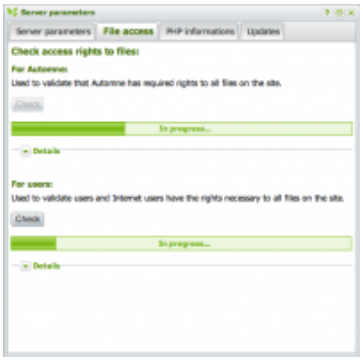
There are two exceptions: PHP CLI and SMTP.
In the case of PHP CLI, Automne has a mecanism allowing it to set its own parameters. In this case the scripts will be executed in a popup window.

If your server does not make SMTP available, you must disable the sending of email in the Automne configuration.

Access to files

Access to files allows verifying that Automne has the access rights to all the files it needs in order the function correctly.

To launch a test, click on "Verify" in the "For Automne" section. You will see the test progression and its results.



At the end of the script the following information will be given:

- Number of folders,
- Number of files,
- Disk space used.

If one or more files are not accessible for Automne they will be listed. You must then change their rights in order to make them accessible to Automne.

You can also test the access rights for users and visitors in order to assure that they have access to information on the site.

To launch a test, click on "Verify" in the "For users" section. You will see the test progression and its results.

For each file tested you will have its name and path, its function ("yes" to authorize access or "no" to forbid it) and the test status.

Example

path and name of file feature test status
File /var/smb/clients/access-v4-fr/www/automne/.htaccess (no) successfully writen

If a file does not have the good rights you must correct this to allow Automne access it.

If certain files have incorrect rights, certain features of Automne may not function correctly.

It is strongly recommended to verify the access to files before beginning to work on the site, in development or production.

PHP Information

This window lists the information of the PHP configuaration as well as that of the modules installed.

It is in fact a display of a phpinfo.

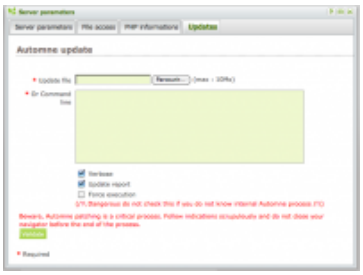
Updates

Allows updating Automne with patches.

You can know your version of Automne by clicking on the icon situated in the upper part of the right

sidebar.

To effectuate an update indicate the file to be used in File update ("shortcut" button) then validate.



Note that the Automne update is effected with update "patches".

Before updating Automne please make sure that it has file write permission.

It is generally preferable to enable the system debugger before effectuating an update in order to verify that no errors appear after the update.
Generally, make sure to test all updates with a site test in order to avoid any unwanted operations on the site in production.

Automne Parameters

Automne parameters determine the general behavior of Automne, it is therefore important to understand and optimize their configuration.

As in the parameters of the Automne modules, those for the "standard" module are available in /automne/classes/modules/moduleCodename_rc.xml :

/automne/classes/modules/standard_rc.xml

Pass your cursor over the parameters to view a description of to what it pertains.

Name of the application

The name of the Automne application. Not to be confused with the name of the site, as your application can have one or many sites.

This information is attached to the APPLICATION_LABEL constant.

Administrator email

This email address is used by Automne to signal 404 errors and all the critical errors produced on the site.

It is important that it is valid address and consulted to guarantee a minimum of health for the Automne application and the site(s) it manages.

Application email

This address will be used to send email from Automne.

In other words when Automne sends an email it will be marked as the sender. To avoid having email labelled as spam or junkmail it is advisable to ensure that this address belongs to the same domain.
For example, if Automne is on the domain automne-cms.org an address such as mail@automne-cms.org should be used.

Activating system debugging

This option activates the display of system error messages.

It deactivates the compression of Javascript and CSS files so that they can be debugged. The option must not be active for a site under construction for security reasons. While this option is deactivated, error messages are saved in the file: /automne/cms_error_log, so it is always possible to consult them via this file.

Activating debugging statistics

If system debugging is active, this option displays the statistics for the site's PHP and MySQL scripts. These statistics will be displayed at the end of the page or in the Javascript console.

Activating advanced debugging statistics

If system debugging is active, this option displays detailed statistics for the site's PHP and MySQL scripts.

A link to these statistics will appear in the Automne debugging console or at the bottom of the page.

Activating Polymod debugging

If system debugging is active, this option displays debugging info specific to Polymod modules.

Typically, when you display objects or Polymod elements from the admin interface, you will see the identifiers of each object, element, field and field values, indicated in red.

Activating client-side rights verification

If this option is active, all access rights to site elements (Pages, Modules, Module elements) will also be verified on the site's client side.

"Anonymous user" rights will be used for non-authenticated users.

Attention, this option has a significant impact on site performance. Do not activate it if the site only has elements visible to all users.

This option is generally used to define secure sections on the site (in the case of an Intranet or Extranet).

You must regenerate sites after modifying this option.

To know more about client-side rights verification

.

Activating background scripts:

Activates the treatment of scripts (sending emails, page generation, etc) as server background tasks.

This option requires PHP CLI on the server (see Server parameters).

If this option is deactivated, these tasks require that a user is connected to Automne administration to effect them correctly.

Deactivating sending emails:

Deactivates sending all emails from Automne and the site.

Use this if your server does not have SMTP or if you do not want Automne to send emails.

Activating the sending emails logs:

This option logs each email generated and sent by Automne.

Activating 404 page email alerts:

If this option is active, each 404 error generates an email alert for the administrator.

This email will contain all the information necessary for the identification of the error. This option must be activated for a site under construction.

Activating the use of printable pages:

This option generates printable pages with content areas selected at the page model level.

If this option is activated, an impression model is available in page model management. A link to a printable page can automatically be generated by the '<atm-print-link>' tag.

This option must be activated when the "search engine" module (ASE) is installed.

In effect, the search engine indexes the pages via their corresponding text, that is to say what is defined as printable.

You must regenerate sites after modifying this option, as "print" pages are regenerated at the same time as the pages.

Title of links for pages in the tree:

In the page tree, it allows toggling between page titles and labels display links to pages where the titles are too long.

Opening zoom images in a pop-up:

Activating this option opens zoom images from "image" blocks of rows in a pop-up window.

If this option is not active, zoom images will be opened in a new window.

You must regenerate sites after modifying this option.

Deactivating advanced metadata in pages:

Deactivates the "Author", "Email response" and "Copyright" metadata from pages on the site.

Authorising the insertion of images in WYSIWYG :

This option adds the "Images" plugin to the WYSIWYG toolbar.

This option should not be activated because the images inserted in this manner are not submitted for validation and

nothing controls their permanence. Inserting images via "image" block of rows is preferable.

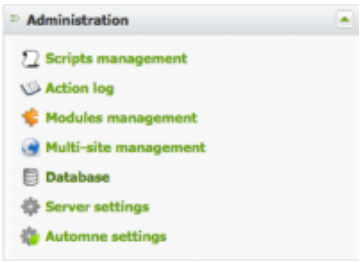
Using this option is unadvisable because images inserted in this manner are not submitted for validation and nothing ensures their permanence.

> If you want to insert images via the WYSIWYG test editor, use the appropriate plugin: with Polymod the creation of a is wysiwyg plug-in is easy.

Databases

Database access

The administration tab will show the "Database" link only if the user has administration rights.



This link leads to the phpMyAdmin application, a personalized version of which is integrated into Automne. This personalization is characterized by the fact that authentification is linked to Automne user accounts: only an Administrator can access phpMyAdmin.

Once in phpMyAdmin, the administrator can directly access the database content for consultation or modification.

ATTENTION: It is necessary to know precisely what every modification implies before directly modifying the database. It is possible that Automne will not function after a bad modification.

Websites management

Définition

Multisite management allows you to manage several websites, each with their own domain, with one Automne and thus only one hosting.

> In order to function correctly all the domains managed by Automne must be directed to the IP of the host server. More information about the IP and hosting

This option is often used when you want a site in several languages.

Your configuration may look like this :

- monsite.fr : your French site
- monsite.com : your English site
- monsite.es : your Spanish site

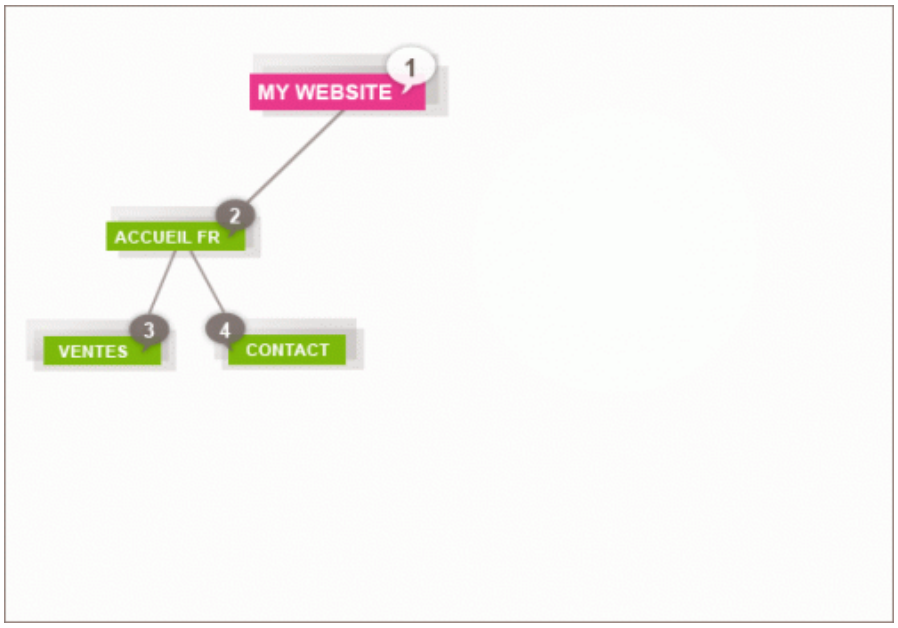
In order to understand how multisites work, one must first imagine the site in the form of a tree (sitemap).

> From the installation of Automne, your sitemap has a unique root: identifier 1. This root is naturally a redirection towards a sub-page, which is the homepage of your site.

A simple example

From your root page (identifier 1), we will create a site in French (FR). The site will be made up of 3 pages:

- a home page (ID 2)
- a "Sales" page (ID 3)
- a "Contact" page (ID 4)

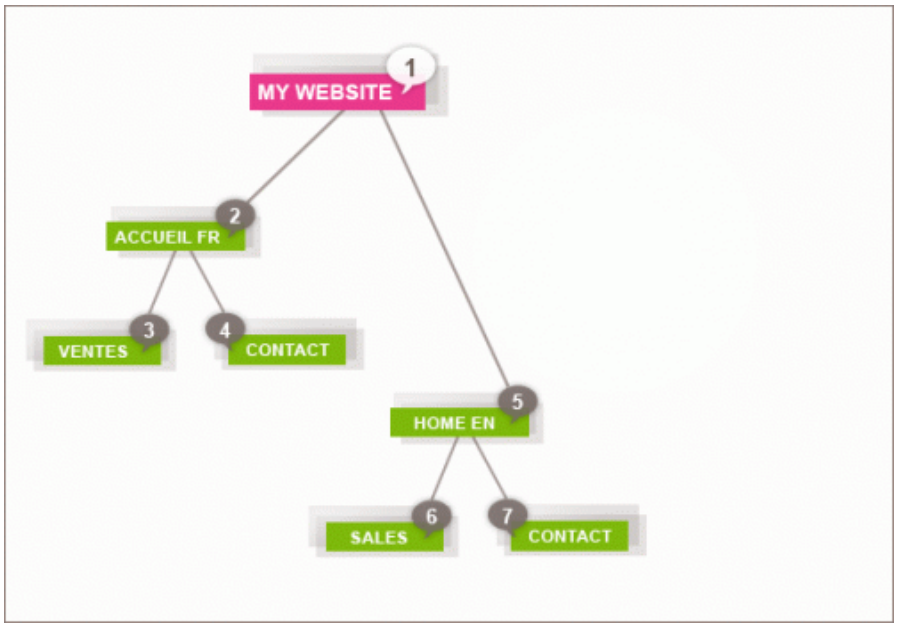


Multilingual site management : original French version

These pages are in French and correspond to the first site at which visitors arrive. This is the first in the list of site sites under Automne.

> Note that page 1 has a redirection to page 2. Nevertheless, with multisite management this redirection is not used. The redirections defined in the site management interface are managed directly in the index.php file, thus even before arriving at page 1.

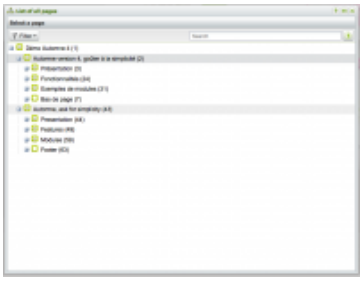
In order to add the English site, its root page must be created (just as page 2 is the root of the French site) thus putting under this page the pages of the English site.



Multilingual site management : english version added

The English site is now made up of pages 5, 6 and 7 and the address mysite.com redirects to page 5.

Our sitemap looks like the following example:



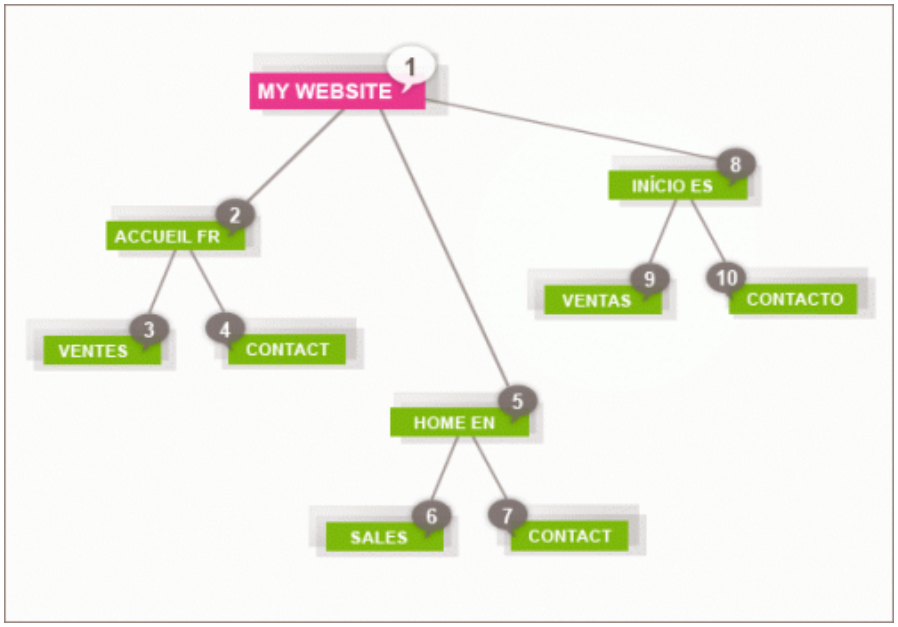
The site configuration looks like the following:

> In the case where two sites are configured with the same domain, the first in the list will be displayed for the domain. It is possible to change the order of sites by dragging and dropping.

Like the English site, the Spanish site can be placed at any level of the tree in order to make it a root. It is possible, for example (but not recommended), to begin the Spanish site on page 5, which would become a page on the

Spanish site and no longer a page on the English site.

For the rest of the example the Spanish site will have page 8 for a root, which will always be created under root 1.



Multilingual site management : spanish version added

There are thus now three root sites: 2, 5 and 8

It is important to note that by default, sites only recognize their tree. Even if the Spanish site is below page 5, for example, it will not be aware that it is below the English site. The Spanish site, thus defined in the site management interface, begins at page 8 (root), no matter where this page is situated in the overall tree. For this reason it is not possible to use a relative type atm-linx to access a page outside of the current site

To bypass this restriction it is nonetheless possible to activate the **crosswebsite** parameter..

> The limitation of a relative atm-linx is not applied to a direct link, thus it is possible to link a page outside of the current site if the identifier (ID) is known.

To facilitate site management it is possible to associate the pages templateswith one or more sites with which they can be used exclusively. This allows you to partition the styles of your sites and prevents editors from overstepping their rights.

Administration

Multisite management is done on the “Site Management” window in the administration sidepanel tab

In this window all the sites managed by Automne are listed and the most important information for each one is displayed.

Remember that the order of sites displayed is crucial because it determines which site will be displayed in the case where several sites share the same domain.

When a site is created you will first be asked to choose a root for your new site. This choice cannot be modified afterwards, so where your new site begins should be chosen carefully.

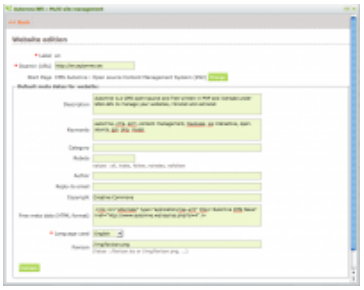
> By default at the installation of Automne, you have a principal site whose root is page 1. It is not possible to modify this root. Logically, Automne needs at least 1 principal site to function. It is, however, possible to modify the root of your other sites from the site management interface.

It is not possible to define 2 sites that point to the same root page. You can nevertheless redirect your second site to another Automne page which redirects to the desired page.

Remember that the sites are virtual; in no case will they delete the data on your site. Yet changes must be made with caution because their impact on pages is not negligible:

- The site root permits Automne to calculate dynamic links
- The indicated language defines the default language of pages.
- The metadata are inserted by default for all pages.

Once at the chosen root you have a window for both the modification and creation of a site.



Here you can define:

- **the label** of your site; the label is an important value which will be used to construct the URLs of your site. Suppose that the domain of your site is mysite.es and that its label is site_es; all the URLs of your pages will begin with www.monsites.es/site_es/. The label of a site is not modifiable so it is important to choose it well,
- **the domain** of your site (the address where it will be available),

The default metadata for your site. This information will be added in the meta of your site's pages (at least if you do not otherwise define said pages) and will be useful for navigators and search engines.

The metadata are optional with the exception of the language, which Automne uses to choose the text to display to visitors and thus easily manage a multilingual site.

Metadata in détail :

- **Description** : used to briefly describe site content, this information is often displayed by engines under the address resulting from the search
- **Keywords** : keywords indicate the themes of the site for search engines, although they are used less and

less.

- **Category :** indicates the category to which your site belongs
- **Robots :** defines the behavior you want from search engines. This option, for example, can request that content is not indexed by a search engine.
- **Author :** defines the author of the site.
- **Response Email :** defines the email address of the site manager; it is important to know that by supplying this information you make the address public and expose it to the risk of unwanted emails.
- **Copyright :** defines the copyright of your site.
- **Free metadata :** allows you to define your own meta fields with the form <meta name= "NAME" content= "VALUE" />
- **Language used :** defines the language of the site and helps Automne choose the text to display to visitors (for example the text of the Forms module).
- **Favicon :** defines the favorites icon (favicon) for your site.

You can delete all sites except the principle site. This is irreversible, so be certain before deleting a site.

If you delete a site in error, it is always possible to recreate an identical one.

Site regeneration

All modification of a site from the site management interface leads to the regeneration of impacted sites.

If necessary, Automne automatically regenerates the index file of your website: this is the file that detects the domain name and redirects to the correct site.

/index.php

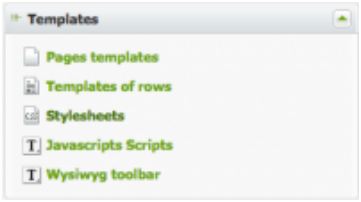
After modifying a site, it is possible that the desired redirection will not be immediately visible; Automne deals with page regeneration in a queue.

The /index.php file is modified automatically while you manually regenerate a branch or the root of a site.

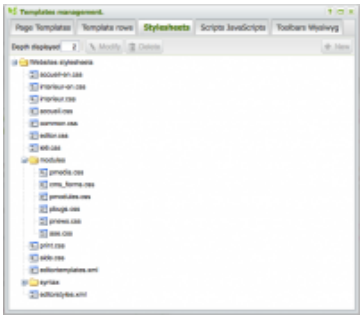
Stylesheets management

With Automne you can manage your style sheets directly from the administration (for more information about style sheets consult the pre-requirements).

Style sheet management is accessible from the right-side panel; choose the “Templates” tab, then the “Style sheets” link

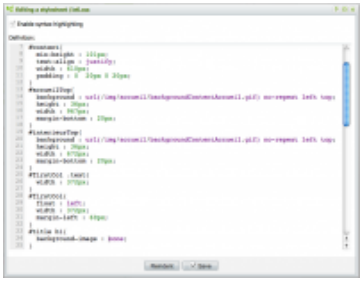


You will then have a list of your style sheets and directories stored in the CSS directory of Automne (/css).



- Editing a sheet: select and click on “Modify”.
- Deleting a sheet: click on “Delete” after selecting a sheet. Make sure you no longer need the style sheet before deleting it in order to avoid damaging the design of your site.

In editing mode you can activate the syntax coloring (check box in the upper-left) in order to have visual guides while editing.



You will note in the list of sheets the presence of two separate files. They are both for the WYSIWYG toolbar that comes with Automne (FCK Editor).

editorstyles.xml defines the styles that will be accessible in the toolbar and applicable to the elements contained

in the HTML editor. It is here that the styles bold green or bold pink, for example, must be defined.

Example :

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<Styles>
<Style name="Custom Bold" element="span">
<Attribute name="style" value="font-weight: bold;" />
</Style>
</Styles>
```

n this page we define that the style “Custom Bold” can be applied to the element span and will be applied to the CSS font-weight property: bold.

Many other possibilities are offered; to know more consult the official help for the FCK editor.

editortemplates.xml defines the XHTML blocks that will be made available in the toolbar.

If you regularly input information with the same structure but variable data in a block of text, it is a good idea to create a template.

Example

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<Templates>
 <Template title="Table whose parameters are set by Automne.">
  <Description>This is an example of a table set by Automne.</Description>
   <Html>
    <![CDATA[
     <table class="CMS_table" cellpadding="2" cellspacing="1" border="0">
       <tr>
         <th class="CMS_th">header</th>
         <th class="CMS_th">header</th>
         <th class="CMS_th">header</th>
         <th class="CMS_th">header</th>
       </tr>
       <tr>
         <td class="CMS_td">cell</td>
         <td class="CMS_td">cell</td>
         <td class="CMS_td">cell</td>
         <td class="CMS_td">cell</td>
       </tr>
     </table>
    ]]>
   </Html>
  </Template>
</Templates>
```

Here we have created a template called “Table whose parameters are set by Automne” and for which we want to display the description: “This is an example of a table set by Automne.”

We next define the basic HTML which will be available.

For more information about creating a template file, please consult the official documentation of the FCK Editor.

Other CSS files are classic and are used either for your page models or your modules. For the CSS module files, it is possible to let Automne automatically manage loading them.

For this it is sufficient to put in the /css/modules directory a file having the codename of your module. For example, for the contact module (codename pcontact) the file will be named pcontact.css. You can even specify the types of peripherals to which your CSS file is addressed.

For example, for a contact module you want to have a file destined for printing; you only have to add a sheet named pcontact-print.css to the directory.

Currently supported devices

- print : printing
- screen : display on the screen

  For the module style sheets, the sheets do not specify the devices to be considered before loading. In the example with the pcontact file, the sheet pcontact.css will be loaded as intended for all devices and the pcontact-print.css sheet will be loaded will be loaded as intended for printing.

<atm-css-tags> tag

n page models it is possible to load your style sheets via the tag of Automne. This tag automatically manages loading desired CSS files and to increase loading performance.

Exemple d'utilisation :

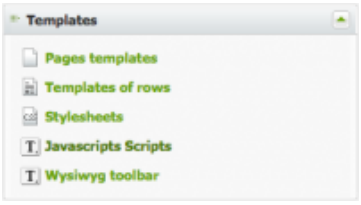
```
<html>
   <head>
      <atm-css-tags files="/css/common.css,/css/internal.css" media="all" />
      <atm-css-tags files="/css/print.css" media="print" />
   </head>
   <body></body>
</html>
```

More on the <atm-css-tag> tag.

Javascript management

With Automne you can directly manage the Javascripts on your site from the administration (for information about Javascript consult the prerequirements).

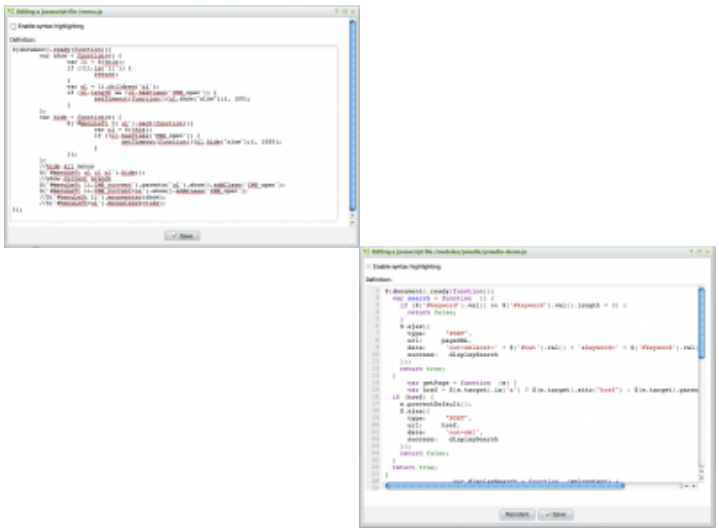
Script management is accessible from the right sidebar, "Models" tab, then the "Javascripts" link.



You will have a list of your script files and repertories stored in the repertory JS of Automne (/js).

- **Editing a script:** select it and click o "Modify"
- **Deleting a script:** click n "Delete" after selecting it. Make sure you no longer need the script before deleting it.

In modification mode you can activate the syntax color (upper-right checkbox) that allows you to have visual guides while editing.



You can let Automne manage script loading in your place.

For this you can create a repertory in the repertory module with the name of your module and put the file you want to load inside.

For example for the contact module (codename "pcontact). You can create a pcontact repertory containing the files checkcontact.js, docontact.js and showcontact.js so that they are loaded on each pages calling on the contact module.

The repertory files are loaded in alphabetical order, iso it is enough to prefix them in order to determine their loading order.

The <atm-js-tags> tag

In the page models it is possible to load your Javascripts via the <atm-js-tag> tag of Automne. This tag permits automatic loading of desired Javascripts and increases loading performance.

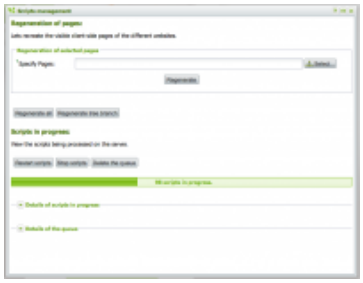
Exemple d'utilisation :

```
<html>
   <head>
      <atm-js-tags files="/js/jquery-1.3.2.js,/js/common.js" />
   </head>
   <body></body>
</html>
```

Find out more about <atm-js-tag>.

Script management

The script management window allows one to controle scripts currently running, but can also run regenerate page scripts.



Regenerating pages

Page regeneration is practical for taking into account a modification you have made on the site configuration, a template or a row without going through Automne. For example when you edit your models or rows directly on the server.

You can choose to regenerate one or many precise pages by entering their identifiers (separated by commas, or dashes to specify an interval) in the field "Specify pages". You can also select the pages if you don't know their identifier. For this click "Select" next to the field "Specify pages". You can then select your page in the tree of your site.

You can also choose to regenerate all pages of the site by choosing "Regenerate all" or regenerate all the pages descending from a precise page by choosing "Regenerate a branch".

Script management

The scripts in progress section allows you to see the scripts executed by Automne. You can decide to stop these scripts by clicking "Stop scripts", cancel their execution by clicking "Delete queue" or relaunch the execution of all scripts by clicking "Relaunch scripts".

You can follow the progress of scripts in the progress bar located under the action buttons.

You can also display details about scripts in progress or about the script queue by clicking "Detail scripts in progress" or "Detail script queue".

Details of scripts in progress:

This will display:

- the name of the script
- the date and hour when the script begins
- script status (succeeded or failed)

Details of the queue:

You will here have a list of all the scripts waiting to be executed in the order they will be treated.

Adding a PHP script to the queue and executing it

During your PHP development, you will perhaps want to effect a background script with PHP

For example a News module, codename "news".

In the class module, located in /automne/classes/modules/news.php, we indicate that the module extendss the class of the plymod module in order to inherit is essential functions:

```
<?php
class news extends CMS_polymod{
    // My class here...
}
?>
```

In the body of the class module next specify the "scriptTask" feature.

This feature allows one to indicate what the script must do:

```
<?php
function scriptTask($parameters) {
    // My parameters
    $myTask = $parameters['task']; // 'myTaskName'
    $myVar= $parameters['myVar']; // $myVar
```

```
    // My script here...
}
?>
```

In your PHP code, in a row or model, for example, indicate the the code to add a task to the queue and then execute it:

```
<?php
// Add a script to the queue
$parameters = array(
    'task'        => 'myTaskName',
    'myVar'       => $myVar,
);
CMS_scriptsManager::addScript('news', $parameters);

// Launch the queue treatment
CMS_scriptsManager::startScript();
?>
```

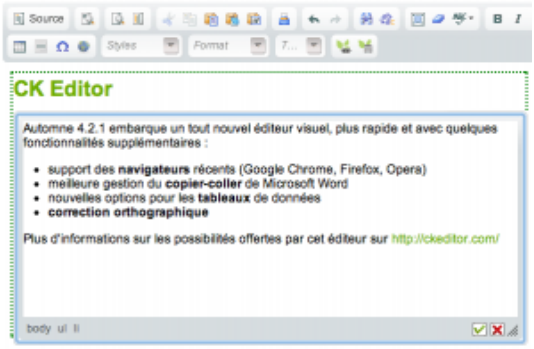
CMS_scriptsManager::addScript allows indicating what module to use, here "news", then to indicate the parameter necessary to execute the script.

CMS_scriptsManager::startScript allowing launching the execution of the queue.

Wysiwyg toolbar

Definition

Most text data in Automne can be formatted with a WYSIWYG toolbar, which allows you to see on the screen the rendition that will published on the web.

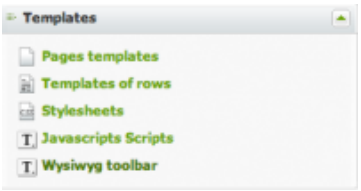


The WYSIWYG text editor used by Automne is FCKeditor, a powerful open-source editor (see the Automne reference).

A number of options are available for the toolbar and it is possible to create personalized toolbars.

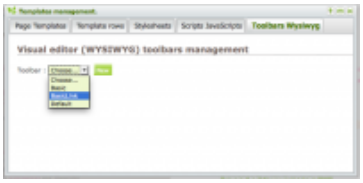
WYSIWYG toolbar management

In the right sidebar, under the Models tab, you can edit FCK Editor toolbars via "WYSIWYG toolbars"



Properties

The toolbar management interface allows you to select an existing toolbar, or create a new one.



The following properties are available while editing a toolbar:

Label:

This is the name of the toolbar.

Identifier (Codename):

This is the unique identifier of the toolbar, in the form of a short chain of characters (20 characters maximum).

Elements:

The double selection box on the left presents the features of the text editor. It has all the icons available.

On the right one can find only the features desired.

It is possible to add or remove features using the arrows between the selection boxes, or by double-clicking on the elements.

> Note that a number of features are available to edit text. Nevertheless it is often prudent to create a simple toolbar, with only the basic features to avoid giving to much liberty to the editors of your site.
>
> The following features are a good compromise for a simple yet efficient toolbar:
>
> - Edition pleine page
> ---
> - Cut
> - Copy
> - Paste as text only
> ---
> - Print
> ---
> - Cancel
> - Redo
> ---
> - Select all
> - Delete the format
> ---
> - Bold
> - Italic
> - Border
> ---
> - Ordered list
> - Non-ordered list
> ---
> - Insert / modify a link
> - Delete a link
> ---
> - Insert a special character

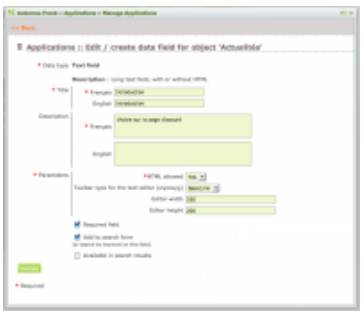
Use

The toolbars can be selected for every text field of an object created with the Ploymod module manager.

Example :

Let us take the example of a "News" module which is defined as a "News" object.

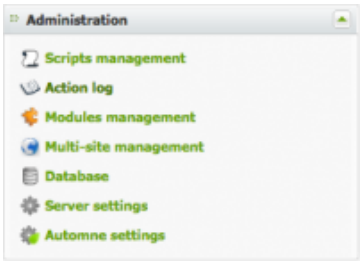
This object has a "description" text field and one can choose to use a specific toolbar (if the "HTML authorized" box is checked).

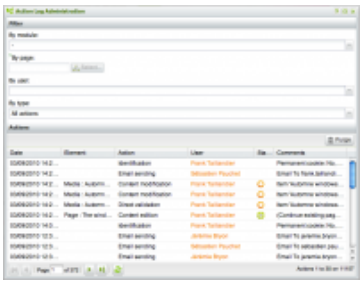


Action Log

Consulting the action log

The action log allows users having "Action log" admin rights to see all the important actions which have been made in Automne.





It is possible to consult the history of the following actions:

Publication actions

- Modification of page properties,
- Modification of page content,
- Continue the modification of page content,
- Submit page content for validation,
- Request the deletion of a page,
- Request a page to be archived,
- Validation of page modifications,
- Modification of module content,
- Request the deletion of a module element,
- Validation of module element modifications.

Administrative actuions

- Modification of page model properties,
- Modification of the XML definition of a page model,
- Modification of content row properties,
- Modification of the XML definition of content rows,
- Modification of user account properties,
- Modification of user account rights,

- Modification of user group properties,
- Modification of user group rights.

Identification actions

- Authentification of a user (with or without the "remember my account" option). IP connection and browser used.
- Automatic authentification of a user. automatique d'un utilisateur. IP connection and browser used.

Sending emails

- Sending an email to the account of an Automne user.
  Sending emails is only registered in the action log if the "Activate the sending emails logs" parameter is active.

For each of these actions, you can know the user who performed it as well as the date and hour it was performed.

Action log management

An administrator can purge the action log of actions more than two months old. To do this, search for an action according to the desired criteria and then click "Purge".

Attention, deleting the action log cannot be cancelled; this action is irreversible.

Some features of Automne use the action log to display certain information. For example, the <atm-last-update> tag depends on the presence of information in the action log in order to be used. Purging the action log must thus be done only if there are too many actions in the log.