Contributing to the AtoM Documentation

An introduction to AtoM’s docs, Sphinx, and reStructuredText

Dan Gillean
Artefactual Systems
AtoM documentation

User manual
User guide to creating, editing, accessing, importing, exporting, and translating content in AtoM. Includes administering the system via the user interface.

Administrator manual
Instructions for installing, upgrading, importing and more. Includes administering the system via the command-line interface.

OVERVIEW
This section provides an overview of the technical architecture and requirements.

- What is AtoM?
- Technical requirements
- Entity types
- Descriptive standards

GETTING STARTED
This section describes how to start using AtoM once it has been installed. It provides a quick orientation to the home page, how to log in and log out, the various user roles in the system, the main page types a user will encounter, and how to switch languages.

- Home page
- Log in
- User roles
- Page types
- Choose language
- Change password

ADD/EDIT CONTENT
This section describes how to add, edit, and delete content and objects from within the system.

CUSTOMIZATION
This section describes how to customize the system's appearance and behavior.

MAINTENANCE
- Command line tools
- Clear cache
- Web analytics
- Logging
- Monitoring
- Populate search index
- Data backup
- Tuning server parameters
- Troubleshooting
- Debug mode
- Elasticsearch

Version 2.3
[Table of contents]

CONTENTS
- AtoM documentation
  - User manual
  - Overview
  - Getting started
  - Add/edit content
  - Access content
  - Translate
  - Reports/printing
  - Import/export
  - Administer
  - Data entry / templates
  - Glossary
  - Administrator manual
    - installation
    - Maintenance
    - Customization
    - Security

SEARCH
Open the general index or type your search in the search box.

AVAILABLE DOCUMENTS
- Version 2.0
- Version 2.1
- Version 2.2
- Version 2.3
Three Manuals:

- **User manual**: Using the application via the user interface
- **Administrator’s manual**: installation, maintenance, upgrades, customization, troubleshooting
- **Developer’s manual**: API, development environments

AtoM documentation is **versioned** for each major release
On the AtoM wiki:

• Information about the documentation
• How to contribute via GitHub
• Contribution guidelines

https://wiki.accessstomemory.org/Resources/Documentation/
FIRST, A STORY...
2013:  
1 PROJECT...  
3 WIKIS  
4 THEMES  

And AtoM 2.0 on the way...
WIKI CHALLENGES

No versioning (e.g. ICA-AtoM 1.2, 1.3; AtoM 2.0)
No enforced structure
Too easy to create orphaned pages
No easy output to other formats
http://onemillionskates.com/inside-edge/a-true-hero-dad-this-ones-for-you
Documentation generator
Open source
Created by Python community
Builds on existing open source projects:
- reStructured Text (markup language)
- Docutils (text processing system)
ADVANTAGES

• Structured documentation
  • Built around a table of contents you define

• Versioning
  • Easily maintained in a repository with many branches

• Easy output to other formats

• Glossary, footnotes, asides, figures, tables, etc...

• Built-in themes or custom themes

• Automated indices

• Strong support for code documentation

http://cheezburger.com/6705529088
reST BASICS

Example of a :term:`glossary term` and a :ref:`reference <section-anchor>` used in a paragraph

.. image:: /images/my-image.png
   :align: center
   :width: 80%
   :alt: my image alt text

Text can be **strong** or have *emphasis*
.. IMPORTANT::

This is an admonition. Sphinx also supports NOTE, TIP, SEEALSO, WARNING, CAUTION, HINT, etc.

This is a `link`_ that can be reused

.. _link: http://www.example.com

This is an `inline link`_ <http://www.example2.com>__

.. code-block:: xml

    <scopecontent>My scope and content</scopecontent>

* This is a bulleted list.
* It has two items, the second item uses two lines.

1. This is a numbered list.
2. It has two items too.

#. This is a numbered list.
#. It has two items too.
ADD A NEW FUNCTION FROM THE MAIN MENU

A new function can be added at any time, from anywhere in the application, via the main menu, provided that a user has sufficient access privileges to create new records. For more information on user permissions, see: User roles and Default user permissions settings in AtoM.

To create a new function from the main menu

1. In the main menu located in the header bar, click the Add menu and select “Function” from the drop-down menu.
2. AtoM takes you to a blank edit page for data entry.
3. On loading, the edit page displays the record with all information areas closed; click on an information area to access the fields grouped under it. Enter data as required.

4. You can quit the process at any time by clicking the “Cancel” button in the button block; any data already entered will not be saved and no new record will be added. Note that simply navigating away from the page by any other means, without first clicking “Create” will also result in no new record being created.
5. To save the new function record, click the “Create” button in the button block at the bottom of the record. AtoM will redirect you to the function’s view page, so you can review your work.
To create a new function from the main menu

1. In the main menu located in the header bar, click the Add menu and select Function from the drop-down menu.
2. AtOM takes you to a blank edit page for data entry.
3. On loading, the edit page displays the record with all information areas closed; click on an information area to access the fields grouped under it. Enter data as required.

4. You can quit the process at any time by clicking the “Cancel” button in the button block; any data already entered will not be saved and no new record will be added. Note that simply navigating away from the page by any other means, without first clicking “Create” will also result in no new record being created.
5. To save the new function record, click the “Create” button in the button block at the bottom of the record. AtOM will redirect you to the function’s view page, so you can review your work.
ADD A NEW FUNCTION FROM THE MAIN MENU

A new function can be added at any time, from anywhere in the application, via the main menu, provided that a user has sufficient access privileges to create new records. For more information on user permissions, see User roles and Default user permissions settings in AtoM.

To create a new function from the main menu:

1. In the main menu located in the header bar, click the menu and select “Function” from the drop-down menu.
2. AtoM takes you to a blank edit page for data entry.
3. On loading, the edit page displays the record with all information areas closed; click on an information area to access the fields grouped under it. Enter data as required.
4. You can quit the process at any time by clicking the “Cancel” button in the button block; any data already entered will not be saved and no new record will be added. Note that simply navigating away from the page by any other means, without first clicking “Create” will also result in no new record being created.
5. To save the new function record, click the “Create” button in the button block at the bottom of the record. AtoM will redirect you to the function’s view page, so you can review your work.
ADD A NEW FUNCTION FROM THE MAIN MENU

A new function can be added at any time, from anywhere in the application, via the main menu, provided that a user has sufficient access privileges to create new records. For more information on user permissions, see: User roles and Default user permissions settings in AtoM.

To create a new function from the main menu

1. In the main menu located in the header bar, click the + “Add” menu and select “Function” from the drop-down menu.
2. AtoM takes you to a blank edit page for data entry.
3. On loading, the edit page displays the record with all information areas closed; click on an information area to access the fields grouped under it. Enter data as required.
4. You can quit the process at any time by clicking the “Cancel” button in the button block; any data already entered will not be saved and no new record will be added. Note that simply navigating away from the page by any other means, without first clicking “Create” will also result in no new record being created.
5. To save the new function record, click the “Create” button in the button block at the bottom of the record. AtoM will redirect you to the function’s view page, so you can review your work.
DETAILED CONTRIBUTION GUIDELINES

Outlined on the AtoM wiki

Notes on getting started and elements of style for consistency across the project

LEARN MORE ABOUT SPHINX AND reST

• Sphinx documentation: http://www.sphinx-doc.org/
• Full reStructuredText docs: http://docutils.sourceforge.net/rst.html
CONTRIBUTING DOCUMENTATION CHANGES WITH VAGRANT:

Using Vagrant, git, and a text editor to view, edit, and add documentation from your local computer
# CLI 101 REMINDERS

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocomplete a directory/...</td>
<td>&gt; TAB</td>
</tr>
<tr>
<td>Show last command</td>
<td>&gt; UP ARROW</td>
</tr>
<tr>
<td>Abort a process</td>
<td>&gt; CTRL+C</td>
</tr>
<tr>
<td>Change to a directory</td>
<td>&gt; cd &lt;path/to/dir&gt;</td>
</tr>
<tr>
<td>Go up one directory</td>
<td>&gt; cd ..</td>
</tr>
<tr>
<td>Go to home directory</td>
<td>&gt; cd</td>
</tr>
<tr>
<td>Clear current commands out of view</td>
<td>&gt; clear</td>
</tr>
</tbody>
</table>

There are some great cheat sheets and tutorials out there – a simple internet search will uncover many options.
SETTING UP THE DOCS IN VAGRANT

Change to the `atom-docs` directory

```bash
> cd atom-docs
```

Check what branch you are on, and what others are available

```bash
> git branch --avv
```

```
vagrant$ git branch --avv
  2.3 6cdb33b [origin/2.3: behind 7] update publication permissions notes, refs #9304
* 2.4 973f7ea [origin/2.4] Xenial FOP install procedure change
```

<table>
<thead>
<tr>
<th>Branch</th>
<th>last commit</th>
<th>last commit message</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>current branch</td>
<td></td>
</tr>
</tbody>
</table>


SETTING UP THE DOCS IN VAGRANT

Check out the branch you want

> git checkout <branch name>

```
vagrant$ git checkout 2.3
Switched to branch '2.3'
Your branch is behind 'origin/2.3' by 7 commits, and can be fast-forwarded.
   (use "git pull" to update your local branch)
[~/atom-docs] (2.3)
vagrant$
```

Pull in the latest changes and rebase your work on top

> git pull --rebase
WAIT... WHAT THE HECK IS GIT??

“Worried look” by Mark Biarnès https://www.flickr.com/photos/104319263@N04/11453255504
Distributed version control and git

**Version control:**
A system for managing changes to source data (documents, code, etc) over time.

**Version control systems** help you track who made what change when, and why. They also allow you to roll back changes to your data to a previous state.

https://git-scm.com/
Local version control

- Changes are tracked on a local computer
- Simple – great for working alone
- Not effective for collaboration

Centralized version control

- Single server with all versioned files; users “check out” files
- Allows for collaboration
- Single point of failure

[Diagram showing a central VCS server with versioned files]

Distributed version control

- Full mirroring of source with each user
- Broadest potential for collaboration
- Complex merging, branching, etc. possible

Git

- Open source
- Distributed model
- Strong User Community, lots of free documentation
- Widely used and supported across platforms

https://xkcd.com/1597/
Git

• Hashes all files in the repository
• Produces a diff of changes
• Commit messages explain who did what when and why
• Supports parallel workflows with a method for resolving conflicts

https://github.com/artefactual/atom-docs/commit/6d8fe4ab9803ec3421b04bcabe215e3dbb4a77dc
GitHub is a web-based Git or version control repository and Internet hosting service. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers both plans for private and free repositories on the same account which are commonly used to host open-source software projects (Wikipedia)

https://github.com/artefactual/atom-docs/
Note: You will need a GitHub account to be able to submit changes from the Vagrant box. It’s free and really easy to set up. Head over to: https://github.com/ to create one.
Learn all things git (for free)


The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the Creative Commons Attribution Non Commercial Share Alike 3.0 license. Print versions of the book are available on Amazon.com.

There are a lot of other free books and tutorials out there as well!
OKAY COOL THX KEEP GOING
**SETTING UP THE DOCS IN VAGRANT**

Clean out the HTML working directory

```> make clean```

Rebuild the local HTML files based on your local copy

```> make html```

HTML versions of your files will be generated and placed in:

```
atom-docs/_build/html/
```
SETTING UP THE DOCS IN VAGRANT

Turn on your network discovery and find Vagrant. Vagrant must be running for this to work. Add a shortcut to the atom-docs folder for easy reference in the future.

An example in Windows Explorer - double-click to drill down until you find the `atom-docs` folder
SETTING UP THE DOCS IN VAGRANT

Open a text editor (like Notepad++, Sublime, Atom.io, etc) and drag the atom-docs folder over to it. If your text editor has the option, you can save this as a project to make things easier in the future.
Configure your global username and email

> `git config --global user.name "Jane Doe"`

> `git config --global user.email janedoe@example.com`

Use the same username and email you signed up with on GitHub. You can check these later with the following (along with a bunch of other settings):

> `git config --list`

MAKING A NEW DOC PAGE

In your text editor, add a new file. Save it with an `.rst` extension.

Each manual (User, Admin, Dev) is organized into sections - these are represented as folders in the project. Each section also has its own subdirectory for the images used in that section of the manual.

Make and save your file in the section to which you want to add it. Use lower case, and dashes instead of spaces.

Add content to your new file! Make sure there is a section anchor at the top of the page so we can link to it later.
Now we need to add it to the `toctree` (table of contents tree) of the AtoM docs project - this tells Sphinx how to order the documents.

Each manual (User, Admin, Dev) has its own `index.rst` file. Open the one belonging to the manual to which you want to add the new page.

Depending on what subdirectory you placed your new file in, add an entry, based on how the files should be ordered if you were reading the docs like a book. Format:

```
subdirectory/filename
```

Remember to save this file after!

More about the `toctree`:

Finally, we need to add a link to the documentation homepage, so people can find your new page when they arrive at the AtoM documentation.

There is an `index.rst` file in the root `atom-docs` directory - you can add the link in the appropriate section there, and save.
REVIEW YOUR WORK LOCALLY

Hey remember this?

Rebuild the local HTML files based on your local copy

> make html

HTML versions of your files will be generated and placed in:

atom-docs/_build/html/

You can use these local HTML files to preview and proof your work in your web browser before submitting it
REVIEW YOUR WORK LOCALLY

When you run the make command, Sphinx will spit out warning or errors if you have messed up the syntax or structure - review these carefully and resolve any issues that relate to your work before proceeding!

An example of an error - a ref link has been made with no related title and section anchor. The docs throw a warning in this instance (they will still compile). Fix any warnings or errors you cause. The path tells you which file, followed by the line number.

Review ➔ Fix ➔ Save ➔ make HTML ➔ Review
Review what files you have changed

> git status -s

Shows new, modified, and deleted files in your local working area. M = modified. ?? = new file.

This example shows a new page, `premis-template.rst`, and 2 new images, being added to the documentation, along with updates to the `index` files.
COMMIT AND PUSH YOUR WORK

Add the changes in your working tree to the index

> git add <file>

Repeat as needed for each file changed. Running `git status -s` again will show you if you have added everything: green means it is now in the index and ready to be committed.

TIP: remember, using TAB in the command-line will help you autocomplete the file paths and names, so you don’t have to type them out fully.
Ready to commit! Add a short message explaining your work

> git commit -m "Message about what I’ve done"

In this example, I have committed my new PREMIS rights data entry page. The command line will show a summary of git’s actions, including creating new files.
To include a longer explanation, don’t use the message option:

```
> git commit
```

Tips on writing the commit message subject:

1. Separate subject from body with a blank line
2. Limit the subject line to 50 characters
3. Capitalize the subject line
4. Do not end the subject line with a period
5. Use the imperative mood in the subject line
6. Wrap the body at 72 characters
7. Use the body to explain what and why vs. how

The editor in your command line will open a page where you add the subject (e.g. short message) at the top, and optionally, a longer description below. Read more:

https://chris.beams.io/posts/git-commit/
Run the pull rebase command one more time

```bash
> git pull --rebase
```

This is done in case someone else is working on the docs at the same time, and has committed changes - this way, you will pull the newest work in first and rebase your work on top of it. If nothing has changed since you last ran the pull --rebase command, then you will be told your branch is up to date.

```
vagrant$ git pull --rebase
Current branch 2.4 is up to date.
[~/atom-docs] (2.4)
vagrant$ ...
```
COMMIT AND PUSH YOUR WORK

Ready to push your work!

> git push

As long as no one has been working on the exact same files as you, then the push should go through without any problems.
WIKIS STILL HAVE THEIR USES!

- Web-based editing
- Popular uses (e.g. Wikipedia) have made it familiar and easy to understand for users
- Good for rapidly changing content

AtoM Sphinx docs:
- User manual
- Admin manual

AtoM Wiki:
- Release notes
- User list
- Community resources
- Development documentation
THANKS!

AtoM docs: https://www.accesstomemory.org/docs
Documentation repo: https://github.com/artefactual/atom-docs
Sphinx documentation: http://sphinx-doc.org/