prolobjectlink-db-jlog

v. 1.0.0

User Guide

WorkLogic Project **2019-01-02**

Table of Contents

1 History

Documentation formats Here's an example of a directory structure for a site: ---------- +- src/ +- site/ +- apt/ | +- index.apt | +- fml/ | +- general.fml | +- faq.fml | +- markdown/ | +- markup.md | +- xdoc/ | +- other.xml | +- xhtml/ | +- xhtml-too.xhtml | +- site.xml ---------- The `${basedir}/src/site` directory which contains a site descriptor along with various directories corresponding to the supported document formats. Let's take a look at examples of the various document formats. * APT: The APT format ("Almost Plain Text") is a wiki-like format that allows you to write simple, structured documents (like this one) very quickly. A full reference of the APT Format is available. * FML: The FML format is the FAQ format. This is the same that were used in Maven 1.x. For more info about the FML Format check the Doxia site. * XDoc: The XDoc format is the same as used in Maven 1.x. A reference for the XDoc Format is available. * Markdown (since maven-site-plugin 3.3): Markdown is a widespread Markup language. [] Other formats are available, but the above four are recognized by default by the site plugin. Any other document format for which a Doxia parser exists can be used as well (see the list of Doxia Markup Languages), but in this case you need to add the corresponding Doxia dependency to your site plugin configuration, as explained in the last paragraph. Note that Doxia also supports several output formats, the site plugin only creates XHTML. Note that all of the above is optional - just one index file is required in one of the input trees. The paths under each format will be merged together to form the root directory of the site. After running '`mvn site:site`' on the example above you will end up with this in your `target` directory: ---------- +- target/ +- site/ +- css/ | +- images/ | +- index.html +- general.html +- faq.html +- other.html ---------- This means that `/src/site/apt/index.apt` will be available in the site as `/index.html`. So, in your site descriptor you might link to `/index.html` and `/other.html`, where the source of these two files would be `/src/site/apt/index.apt` and `/src/site/xdoc/other.xdoc`. The `css` and `images` directories contain stylesheets and images from the skin being used. You can read more about skins in the site descriptor documentation. * Adding Extra Resources You can add any arbitrary resource to your site by including them in a `resources` directory as shown below. Additional CSS files can be picked up when they are placed in the `css` directory within the `resources` directory. ---------- +- src/ +- site/ +- resources/ +- css/ | +- site.css | +- images/ +- pic1.jpg ---------- The file `site.css` will be added to the default XHTML output, so it can be used to adjust the default Maven stylesheets if desired. The file `pic1.jpg` will be available via a relative reference to the `images` directory from any page in your site. * Filtering **Note:** This feature is available in version 2.0-beta-6 or later of the Site Plugin. To filter properties into any supported documentation format, add a `.vm` extension to the filename. For example, the module for the Maven website contains a `content/xdoc/download.xml.vm` file, which uses the expression `${currentStableVersion}` to filter in a property set in the POM. **Note:** Apache Velocity is used to apply the filtering. Because Velocity uses a dot-notation to traverse beans you can **not** use dots in your properties defined in `<properties>`. If you declare these properties in your POM +-----+ *properties !-- This will not work because the name of the property has a dot in it -- my.property*My value */my.property !-- This will work because the name of the property has no dot in it -- myProperty*My other value */myProperty /properties* +-----+ and then use the expression `${my.property}` in your document, it will **not** work. If you instead use the expression `${myProperty}` it will work just fine. Alternatively, you can say `$ project.properties.get("my.property")` which is cumbersome to write. Some properties are defined by default: see Doxia Site Renderer's Velocity processing reference for more information. For a complete usage overview, see Velocity's user guide. * Internationalization Internationalization in Maven is very simple, as long as the reports you are using have that particular locale defined. For an overview of supported languages and instructions on how to add further languages, please see the related article Internationalization. To enable multiple locales, or languages, add a configuration similar to the following to your POM: ---------- *project ... build plugins plugin groupId*org.apache.maven.plugins */groupId artifactId*maven-site-plugin */artifactId version*1.0.0 */version configuration locales*en,fr */locales /configuration /plugin /plugins /build ... /project* ---------- This will generate both an English and a French version of the site. If `en`

is your current locale, then it will be generated at the root of the site, with a copy of the French translation of the site in the `fr/` subdirectory. To add your own content for that translation instead of using the default, create a subdirectory with that locale's name in your site directory and create a new site descriptor with the name of the locale in the file name. For example: ---------- +- src/ +- site/ +- apt/ | +- index.apt (Default version) | +- fr/ | +- apt/ | +- index.apt (French version) | +- site.xml (Default site descriptor) +- site_fr.xml (French site descriptor) ---------- With one site descriptor per language, the translated site(s) can evolve independently.

2 Introduction

Title

2.1 Author ------ Date

Paragraph 1, line 1. Paragraph 1, line 2.

Paragraph 2, line 1. Paragraph 2, line 2.

2.2 Section title

2.2.1 Sub-section title

2.2.1.1 Sub-sub-section title

2.Sub-sub-sub-section title

2.Sub-sub-sub-sub-section title

- List item 1.
- List item 2.
  Paragraph contained in list item 2.
  - Sub-list item 1.
  - Sub-list item 2.
- List item 3. Force end of list:

```
Verbatim text not contained in list item 3
```

1. Numbered item 1.
   A. Numbered item A.
   B. Numbered item B.
2. Numbered item 2.

List numbering schemes: [[1]], [[a]], [[A]], [[i]], [[I]].

Defined term 1

of definition list.

Defined term 2

of definition list.

```
Verbatim text
                        in a box
```

--- instead of +-- suppresses the box around verbatim text.

Figure caption

Centered cell 1,1	Left-aligned cell 1,2	Right-aligned cell 1,3

cell 2,1	cell 2,2	cell 2,3

Table caption

No grid, no caption:

cell	cell
cell	cell

Horizontal line:

2.3 ^L New page.

Italic font. **Bold** font. `Monospaced` font.

Anchor. Link to anchor. Link to http://www.pixware.fr. Link to showing alternate text. Link to Pixware home page.

Force line
break.

Non breaking space.

Escaped special characters: ~, =, -, +, *, [,], <, >, {, }, \.

Copyright symbol: ©, ©, ©.

3 FAQ

3.1 Frequently Asked Questions

General

1. What is the difference between `mvn site` and `mvn site:site`?
2. How do I Integrate static (X)HTML pages into my Maven site?
3. How to include a custom Doxia module, like Twiki?
4. How can I validate my xdoc/fml source files?
5. How does the Site Plugin use the <url> element in the POM?

Specific issues

1. Why do my absolute links get translated into relative links?
2. Why don't the links between parent and child modules work when I run "`mvn site`"?
3. Can I use entities in xdoc/fml source files?

3.2 General

What is the difference between `mvn site` and `mvn site:site`?

`mvn site`

Calls the *site* **phase** of the site **lifecycle**. Full site lifecycle consists in the following life cycle phases: `pre-site`, `site`, `post-site` and `site-deploy`. See Lifecycle Reference. Then it calls plugin goals associated to `pre-site` and `site` phases.

`mvn site:site`

Calls the *site* **goal** of the site **plugin**. See site:site.

[top]

How do I Integrate static (X)HTML pages into my Maven site?

You can integrate your static pages by following these steps:

- Put your static pages in the resources directory, `${basedir}/src/site/resources`
- Create your `site.xml` and put it in `${basedir}/src/site`
- Link to the static pages by modifying the menu section, create items and map them to the filenames of the static pages

[top]

How to include a custom Doxia module, like Twiki?

The site plugin handles out-of-box apt, xdoc and fml formats. If you want to use a custom format like Twiki, Simple DocBook, or XHTML (or any other document format for which a doxia parser exists, see the list of Doxia Markup Languages), you need to specify the corresponding Doxia module dependency, e.g. for Twiki:

```
<project>
  ...
  <build>
    <plugins>
      ...
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-site-plugin</artifactId>
        <dependencies>
          <dependency>
            <groupId>org.apache.maven.doxia</groupId>
            <artifactId>doxia-module-twiki</artifactId>
            <version><!-- doxia version appropriate to the site plugin version
          </dependency>
        </dependencies>
      </plugin>
    </plugins>
  </build>
  ...
</project>
```

Note that the doxia version has to be adjusted to the site-plugin version you are using, see the Migration Guide. In particular, for site plugin versions `>=2.1` you need to use doxia `>=1.1`.

[top]

How can I validate my xdoc/fml source files?

Since version 2.1.1 of the Site Plugin, there is a `validate` configuration parameter that switches on xml validation (default is off). Note that in the current implementation of the parser used by Doxia, validation requires an independent parsing run, so that every source file is actually parsed twice when validation is switched on.

If validation is switched on, **all** xml source files need a correct schema and/or DTD definition. See the Doxia documentation on validating xdocs, and the schema definitions for xdoc and fml.

[top]

How does the Site Plugin use the <url> element in the POM?

The Site Plugin does not use the <url> element in the POM. The project URL is just a piece of information to let your users know where the project lives. Some other plugins (e.g. the project-info-report-plugin) may be used to present this information. If your project has a URL where the generated site is deployed, then put that URL into the <url> element. If the project's site is not deployed anywhere, then remove the <url> element from the POM.

On the other hand, the <distributionManagement.url> is used in a multi-module build to construct relative links between the generated sub-module sites. In a multi module build it is important for the parent and child modules to have **different** URLs. If they have the same URL, then links within the combined site will not work. Note that a proper URL **should** also be terminated by a slash ("/").

[top]

3.3 Specific issues

Why do my absolute links get translated into relative links?

This happens because the Site Plugin tries to make all URLs relative, when possible. If you have something like this defined in your `pom.xml`:

```
<url>http://www.your.site.com/</url>
```

and create links in your `site.xml` (just an example) like this:

```
<links>
  <item name="Your Site" href="http://www.your.site.com/"/>
  <item name="Maven 2" href="http://maven.apache.org/maven2/"/>
</links>
```

You will see that the link to "Your site" will be a relative one, but that the link to "Maven 2" will be an absolute link.

There is an issue for this in JIRA, where you can read more about this.

[top]

Why don't the links between parent and child modules work when I run "`mvn site`"?

What "`mvn site`" will do for you, in a multi-project build, is to run "`mvn site`" for the parent and all its modules **individually**. The links between parent and child will **not** work here. They **will** however work when you deploy the site.

If you want to test this, prior to deployment, you can run the `site:stage` goal as described in the usage documentation instead.

[top]

Can I use entities in xdoc/fml source files?

Yes. Entity resolution has been added in Doxia version 1.1, available in Site Plugin 2.1 and later.

There is a catch however. In the current implementation (as of maven-site-plugin-2.1.1), entities are only resolved by an independent validation run. Therefore, if you want to use entities, you **have** to switch on validation for your xml source files. See MSITE-483.

[top]