

Documenter ses projets ActionScript avec ASDoc

par Vincent Petithory ([Mes articles](#))

Date de publication : 13/06/2008

Dernière mise à jour :

Cet article traite de la mise en place des différents fichiers à créer et des quelques règles de bases à respecter pour générer sa documentation via ASDoc.

I - Description et rôle.....	3
II - Documenter une classe.....	3
II-A - Exemple et règles.....	3
II-B - Cas des fonctions getter et setter.....	3
II-C - Tags ASDoc.....	4
II-D - Balises XHTML utilisables.....	5
III - Génération de la documentation.....	5
III-A - Ressources nécessaires.....	5
III-B - Emplacement des fichiers.....	6
III-C - Création des fichiers.....	6
III-C-1 - Fichier BAT.....	6
III-C-2 - Fichier XML de configuration.....	6
III-C-3 - Fichier XML manifest.....	8
III-D - Exemples.....	8
IV - Maintenance.....	10

I - Description et rôle

L'ASDoc est à actionScript ce que la javadoc est à java. L'ASDoc contient un compilateur qui permet de générer des documentations de classes ou de packages entiers à l'aide de XSLT. L'ASDoc est disponible dans le Flex SDK 3. Son intérêt est d'avoir une documentation publique permettant à toute personne voulant utiliser une classe d'avoir une notice précise et standardisée. La documentation de Flex est générée par ASDoc.

II - Documenter une classe

II-A - Exemple et règles

La documentation d'une classe répond à un certain nombre de règles.

- Pour chaque déclaration d'un élément actionScript (déclaration d'une variable, d'une constante, d'une classe, d'une méthode), ASDoc générera un élément associé dans la documentation de cette classe.
- La déclaration d'un commentaire associé à un élément se fait immédiatement avant celui-ci.
- L'écriture des commentaires doit respecter une syntaxe valide XHTML.
- Chaque nouvelle ligne du commentaire doit comporter un astérisque * ainsi qu'un espace.

Un exemple typique de documentation d'un élément (ici une méthode) :

<pre> 30 31 32 33 34 35 36 37 38 39 </pre>	<pre> /** * Uses the LZW dictionary algorithm to compress a file. * * <p>Note that the input file must contain byte data (coded on 8 bits).</p> * * @param sourceByteArray A byte array representing the file. * @return The LZW-compressed byte array. This byte array contains short data (coded on 16 bits). */ public static function compress(sourceByteArray:ByteArray):ByteArray { </pre>	<p>Debut de commentaire de doc</p> <p>Commentaire principal</p> <p>Commentaire secondaire</p> <p>Tags ASDoc</p> <p>Fin de commentaire</p>
--	---	---

Tous les commentaires peuvent être écrits sur plusieurs lignes. En revanche, pour obtenir des sauts de ligne et paragraphes dans la documentation finale, il est nécessaire d'utiliser des balises XHTML. Utiliser <p></p> pour générer un nouveau paragraphe. Le premier <p></p> est automatiquement associé au commentaire secondaire.

II-B - Cas des fonctions getter et setter

En ActionScript 3.0, certaines variables non publiques s'accompagnent de fonctions getter et setter (ou non). Afin de documenter correctement ces groupes d'éléments qui ne doivent en former qu'un seul dans la doc, certaines conventions sont à suivre.

```

/**
 * @private
 * Memorisation de la propriété exemple
 */
private var _exemple:String = "defaultValue";

/**
 * Ceci est un exemple de documentation d'une variable
 * privée possédant des méthodes <code>getter</code> et <code>setter</code>.
 *
 *
 * @default "defaultValue"
 */
public function get exemple():String
{
    .....
    return _exemple;
}

/**
 * @private
 */
public function set exemple(value:String):void
{
    .....
    // corps de la méthode.
}

```

Exemple

Le principe est d'insérer le tag **@private** lors de la déclaration de la variable private.

Par convention, on spécifie la fonction get en premier (en effet, la fonction set peut ne pas être spécifiée si on souhaite laisser la variable en lecture seule).

On documente alors normalement la méthode getter comme si on documentait la variable, en spécifiant sa valeur par défaut par exemple. On ne met pas le tag **@return**, puisque ce n'est pas la méthode getter en elle-même que l'on documente.

Pour finir, on exclut la méthode setter en lui affectant le tag **@private**.

II-C - Tags ASDoc

Il est nécessaire d'utiliser un certain nombre de tags pour créer une documentation efficace. Les tags de base les plus utilisés sont :

@param

La syntaxe est : **@param** paramètre description du paramètre. Ce tag permet de décrire un des paramètres d'une méthode. Créer un nouveau tag pour chaque paramètre. Inclure la valeur par défaut, si celle-ci existe, dans la description.

@return

Spécifie une valeur de retour pour la méthode documentée. La syntaxe est : **@return** description

@default

Spécifie la valeur par défaut d'une propriété La syntaxe est : **@default** value


@private

Si le tag **@private** est présent dans un commentaire, l'élément actionScript associé ne sera pas inclus dans la documentation. Cela est utile pour retirer des méthodes ou variables privées que l'utilisateur de la classe n'a pas besoin de connaître.

@see

Ce tag permet de créer un lien. Suivant le texte qui suit le tag, le lien pourra être externe (lien http) ou interne à la documentation (lien pointant vers une autre classe de la documentation). Le nombre de possibilités est assez élevé. Cf le tableau complet ci-après.

@inheritDoc

Si ce tag est présent dans un commentaire, le compilateur recherche dans les interfaces puis classes mères les occurrences de la méthode ou variable associée au commentaire, puis recopie la description principale et les tags **@param** et **@return** du commentaire trouvé dans le commentaire recevant **@inheritDoc**. Liste de tous les tags utilisables (en anglais) :  [Liste complète des tags](#)

II-D - Balises XHTML utilisables

Les plus utilisés sont les balises `<p>`, `<code>`, et ont le même rôle qu'en XHTML. La balise `<code>` est très utilisée pour formater les noms de variables, leurs valeurs, etc. A noter qu'il ne faut pas utiliser la balise `<p>` pour le premier paragraphe du commentaire d'un élément.

Liste complète des balises HTML utilisables

III - Génération de la documentation

La génération de la documentation fait appel au compilateur de l'ASDoc : `asdoc.exe`. Celui-ci est présent dans le Flex SDK 3 à l'adresse : `${chemin_du_SDK}\bin\asdoc.exe` Les templates et bibliothèques ASDoc se situent dans `${chemin_du_SDK}\asdoc\`

III-A - Ressources nécessaires

Plusieurs manières permettent de générer notre documentation plus ou moins efficacement. La plus efficace est l'utilisation de tâches ANT via eclipse, car elle nécessite peu de maintenance, mais est en revanche plus dure à mettre en place au départ, car elle nécessite des connaissances sur ANT.

La façon de procéder plus simple et malgré tout très efficace est l'utilisation de 2 fichiers XML et d'un fichier `.bat` (sous windows) ou `.sh` (sous linux)

- Le fichier `.bat` a pour rôle d'appeler la commande système permettant de lancer le compilateur..
- Un fichier xml de configuration des options à envoyer au compilateur.
- Un « manifest file » de type XML décrivant les classes à inclure dans la documentation.

III-B - Emplacement des fichiers

Dans le cadre d'une programmation de type objet, organisée sous forme de packages, il sera judicieux de placer les fichiers précédents dans le dossier contenant tous les packages (à priori la racine du projet).

III-C - Création des fichiers

III-C-1 - Fichier BAT

Voici un template de fichier bat :

```

1  @echo off
2
3  REM the path of the asdoc compiler
4  SET asdocPath="C:\Program Files\Adobe\Flex SDK 3\bin\asdoc.exe"
5
6  Rem the path of your xml configuration
7  SET xmlConfigPath=config.asdoc.template.xml
8
9  %asdocPath% -load-config+%xmlConfigPath%
```

Ce fichier lance le compilateur ASDoc avec les options spécifiées dans le fichier xml. Il faut paramétrer la variable asdocPath (chemin vers le compilateur ASDoc, dépendant de la machine utilisée). Il faut aussi paramétrer le chemin vers le xml de configuration des options du compilateur. Les chemins peuvent être relatifs ou absolus. Il est conseillé de spécifier un chemin absolu pour le compilateur.

III-C-2 - Fichier XML de configuration

Ce fichier xml contient les options à passer au compilateur ASDoc.

```

<?xml version="1.0" encoding="utf-8" ?>
<flex-config>
  <compiler>
    <source-path>
      <path-element>./</path-element>          chemins sources
    </source-path>
    <namespaces>
      <namespace>
        <uri>http://myURL.com</uri>           Namespace.
        <manifest>asdoc-manifest.xml</manifest> manifest file
      </namespace>
    </namespaces>
  </compiler>
  <main-title>myMainDocTitle</main-title> Titre de la doc
  <window-title>myBrowserWindowTitle</window-title> Titre du header
  <footer>myFooterText</footer> titre de pied de page navigateur
  <output>myAsdocOutput</output> dossier de sortie de la doc
  <doc-namespaces>
    <uri>http://myURL.com</uri>           namespace à inclure
  </doc-namespaces>

  <packages>
    <package>
      <string>package.name</string>
      <string>myPackageDescription</string> Inclusion de
                                           description des
                                           packages intégrés
    </package>                               Nom + description
  </packages>

  <metadata>
    <title>myFrameWorkDocumentationTitle</title>
    <description>myFrameWorkDocumentationDescription</description>
    <publisher>publisher</publisher>
    <creator>creator</creator>           Les metadata à inclure
    <language>EN</language>
  </metadata>
</flex-config>
    
```

Les chemins sources à spécifier devront être en concordance avec le chemin des packages spécifiés dans le « manifest file ». Plusieurs chemins sources et namespaces peuvent être déclarés.

L'une des options importantes est namespaces.

Cependant, c'est la plus flexible. Les classes spécifiées dans les manifest files seront affectées à l'espace de nom (namespace) qui leur est associé.

L'option *doc-namespaces* recense les namespaces à inclure dans la documentation. Ces namespaces doivent avoir été déclarés dans l'option *namespaces*.

L'utilisation des namespaces est une des manières d'inclure les classes à documenter, mais il y en a d'autres (option *include-classes* par exemple).

Tous les chemins spécifiés peuvent être relatifs ou absolus.

III-C-3 - Fichier XML manifest

Ce fichier recense toutes les classes appartenant à un namespace (qui sera inclus ou non dans la documentation)
Ce fichier doit suivre la syntaxe suivante :

```

1  <?xml version="1.0" encoding="utf-8" ?>
2  <componentPackage>
3      <component id="ClassName" class="package.ClassName"/>
4  </componentPackage>

```

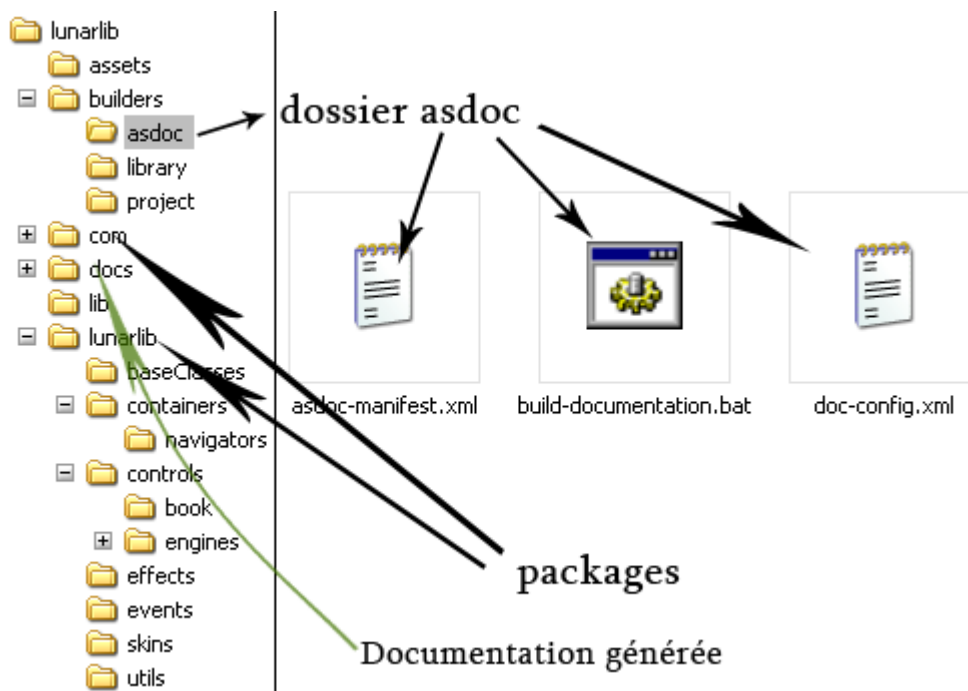
Chaque noeud *component* inclut une nouvelle classe. L'attribut *id* de ce noeud est le nom de la classe, et l'attribut *class* a pour valeur une chaîne de caractère qui serait celle qu'on ferait pour importer cette classe (par rapport au source-path spécifié en paramètre du compilateur).

Le *package* à spécifier doit être en concordance avec les chemins sources spécifiés dans le fichier xml de configuration du compilateur..

III-D - Exemples

Sont joints avec ce tutorial les templates des fichiers précédents. Voici un exemple d'utilisation de ces templates, afin de comprendre comment les paramétrer. Il permet de générer la documentation de toutes les classes du package *lunarlib.utils*

L'arborescence de notre exemple est la suivante :



Et voici les fichiers créés en conséquence :

build-documentation.bat :

```

@echo off

REM the path of the asdoc compiler
SET asdocPath="C:\Program Files\Adobe\Flex SDK 3\bin\asdoc.exe"

Rem the path of your xml configuration
SET xmlConfigPath=doc-config.xml

%asdocPath% -load-config+=%xmlConfigPath%

```

doc-config.xml :

```

<?xml version="1.0" encoding="utf-8" ?>
<flex-config>
  <compiler>
    <source-path>
      <path-element>../../</path-element>
    </source-path>
    <namespaces>
      <namespace>
        <uri>http://code.lunar.com/dev/flex/lunarlib</uri>
        <manifest>asdoc-manifest.xml</manifest>
      </namespace>
    </namespaces>
  </compiler>
  <main-title>Petithory Vincent - lunarlib Framework Documentation</main-title>
  <window-title>Petithory Vincent - lunarlib Framework Documentation</window-title>
  <footer></footer>
  <output>../../docs</output>
  <doc-namespaces>
    <uri>http://code.lunar.com/dev/flex/lunarlib</uri>
  </doc-namespaces>

  <packages>
    <package>
      <string>lunarlib.utils</string>
      <string>The lunarlib.utils package contains utility classes for data comp
    </package>
  </packages>

  <metadata>
    <title>lunarlib framework documentation</title>
    <description>The documentation of the lunarlib framework.</description>
    <publisher>Vincent Petithory</publisher>
    <creator>Vincent Petithory</creator>
    <language>EN</language>
  </metadata>
</flex-config>

```

Le source-path est spécifié comme étant à 2 niveaux supérieurs, ainsi, les classes spécifiées dans le manifest file ci dessous concordent.

Le paramètre output est également 2 niveaux supérieurs, afin de créer un dossier docs dans lequel sera générée la documentation.

asdoc-manifest.xml :

```
<?xml version="1.0" encoding="utf-8" ?>
<componentPackage>
  <!-- utils -->
  <component id="ColorMatrixUtils" class="lunarlib.utils.ColorMatrixUtils"/>
  <component id="IWaveletCompressor" class="lunarlib.utils.IWaveletCompressor"/>
  <component id="LZWCompressor" class="lunarlib.utils.LZWCompressor"/>
  <component id="LZWFileCompressor" class="lunarlib.utils.LZWFileCompressor"/>
  <component id="VersatileUtils" class="lunarlib.utils.VersatileUtils"/>
  <component id="WaveletImageCompressor" class="lunarlib.utils.WaveletImageCompressor"/>
  <component id="WaveletImageCompressorExtend" class="lunarlib.utils.WaveletImageCompressorExtend"/>
</componentPackage>
```

Le manifest file contient la liste des classes à inclure dans la documentation. Ces fichiers sont joints au tutorial, ainsi que la documentation générée.

IV - Maintenance

Lorsque l'ajout de nouvelles classes est à faire, il suffit de les ajouter dans le fichier manifest. Eventuellement, lorsque les packages deviennent nombreux, d'autres espaces de nom peuvent être créés, et dans ce cas, un nouveau fichier manifest doit leur être associés.