INSA RENNES

Institut National des Sciences Appliquées
Ecole publique d'ingénieurs

Introduction à Doxygen
Analyse, Conception et Programmation Orientée Objet
Quatrième année - Informatique

2010-2011

Peggy Cellier, Département Informatique
peggy.cellier@insa-rennes.fr
Gabriel Cirio, Département Informatique
gabriel.cirio@irisa.fr
Laurent George, Département Informatique
l.george@inria.fr
Maud Marchal, Département Informatique
maud.marchal@insa-rennes.fr
Léo Terziman, Département Informatique
leo.terziman@inria.fr

1

Doxygen

1.1 Présentation

Doxygen est un outil d'auto-documentation pour le C, C++, Java, Objective-C, Python, IDL, Fortran, VHDL, PHP, C#. Au moyen de balises placées autour du code et permettant de décrire fonctions/classes/structures/..., il est possible de générer une documentation technique sous plusieurs formats (HTML, XML, ...). Le principe est assez similaire à Javadoc, outil d'auto-documentation pour Java.

À la base, Doxygen est un utilitaire qui parcourt des fichiers sources et génère des pages de documentation en fonction de la configuration voulue par l'utilisateur. Cette configuration est stockée dans un fichier Doxyfile. Afin de faciliter l'utilisation de Doxygen, on peut utiliser Doxywizard, une GUI (Graphical User Interface) qui permet de gérer plus facilement la configuration et le lancement de Doxygen.

Un manuel, aide et exemples sont disponibles à l'adresse : http ://www.stack.nl/˜dimitri/doxygen

1.2 Installation

- **Doxywizard**
  Télécharger l'outil nécessaire à l'adresse suivante et suivre les instructions :
  http ://www.stack.nl/˜dimitri/doxygen/download.html
- **Graphviz**
  On peut aussi éventuellement installer graphviz si on veut faire apparaître des graphes d'appels ou de classes dans la documentation :
  http ://www.graphviz.org/Download..php

1.3 Commenter le code

Le principe est de mettre des commentaires respectant une certaine syntaxe pour pouvoir être analysés.

Les blocs commentaires qui seront interprétés par Doxygen peuvent commencer et finir de plusieurs façons :

```
/**
* Commentaires Doxygen
*/

/*!
* Commentaires Doxygen
*/

///
/// Commentaires Doxygen
///

//!
//! Commentaires Doxygen
//!
```

La syntaxe des blocs à base de balises permet à Doxygen de connaître la nature du code commenté et les différents paramètres qui vont avec. Attention, si la syntaxe est incorrecte ou incomplète, Doxygen n'inclura pas la documentation associée dans le fichier final (penser à regarder dans le fichier log pour voir les messages issus de la génération de la documentation).

Voici les blocs les plus utiles, pour les autres, vous pouvez consulter la documentation complète sur le site web.

Nature du bloc	**Syntaxe Doxygen**

Fichier	`/**`
	` * \file nom du fichier`
	` * \brief breve description de son contenu`
	` * \author nom prenom`
	` * \version X.Y`
	` * \date date de creation`
	` *`
	` * Description plus detaillee du contenu du fichier`
	` *`
	` */`

Fonction	`/**`
	`* \fn prototype de la fonction`
	`* \brief description breve de la fonction`
	`*`
	`* Un peu plus de details sur la fonction`
	`*`
	`* \param[in] nom description du parametre lu`
	`* mais pas modifie.`
	`* \param[out] nom description du parametre`
	`* modifie mais pas lu.`
	`* \param[in,out] nom description du parametre`
	`* lu et modifie.`
	`*`
	`* \return le type de retour et sa description`
	`*/`

Nature du bloc	Syntaxe Doxygen

Macro	`/**`
	`* \def nom de la macro`
	`* Description de la macro`
	`*/`

Structure / Classe / Enumération /. . .	`/**`
	`* \struct nom de la structure`
	`*  (\class pour une classe,`
	`*  \enum pour un type enumere, ...)`
	`* \brief brieve description de l'objet`
	`*`
	`* Description detaillee de l'objet`
	`*/`

Champ / membre d'une structure / Classe / Enumération . . .	`/*!< Commentaire sur le champ/membre de la` `* structure/classe */`

1.4 La génération de documentation

Sous Doxywizard, il y a deux modes de configuration de la génération de la documentation, Wizard et Expert. Les deux mettent à jour le Doxyfile mais le premier est plus simple à utiliser. Le deuxième permet de personnaliser la documentation.

1.4.A Configuration Wizard

Il y a 4 parties à renseigner :

- **Project**
  Les informations permettent d'identifier le projet, où sont les sources, où mettre la documentation résultante.

- **Mode**
  En ce qui concerne le mode d'extraction, "Documented entities only" et "All Entities" permettent de définir quels éléments du code doivent être commentés et référencés dans la documentation finale. Pour un réglage plus fin, il faut utiliser le mode Expert. La case "Include cross-referenced source code in the output" est bien pratique, elle permet de faire des allers-retours entre la documentation et le code facilement. En ce qui concerne le type de programme analysé, choisir le langage utilisé dans les fichiers sources.

- **Output**
  Choisir le format de sortie (HTML en général, il supporte plus d'options que les autres modes).

– **Diagrams**
  Indiquer si l'on veut des diagrammes ou pas. Pour la 3ème option, il faut avoir installé graphviz. Si cette option ne marche pas (le code des graphes s'affiche en dur alors qu'on voudrait un joli dessin), dans ce cas, se reporter au paragraphe sur le sujet "Dot" dans la configuration Expert. Dans les graphes générés, on peut voir l'interaction entre classes/structures, les dépendances de fichiers, etc. Les call graphs ou called by graphs permettent de voir qui appelle quoi, ce qui peut être très lourd suivant le code.

1.4.B Configuration Expert

Toutes les variables du Doxyfile peuvent être modifiées ici. Il suffit de passer la souris sur le nom de l'une d'entre elles pour voir la description. Nous allons en citer certaines très utiles au fil de la description des différentes catégories :

– **Project**
  Toutes les variables permettant de décrire l'environnement du projet. Par exemple, si vous avez sélectionné dans la configuration Wizard une optimisation pour le langage C, la variable OPTIMIZE_OUTPUT_C devrait être activée.

  Autres variables intéressantes :
  – BRIEF_MEMBER_DESC et REPEAT_BRIEF qui permettent de configurer l'emplacement des descriptions brèves dans la documentation.

  – OUTPUT_LANGAGE : on peut choisir "French" pour avoir des onglets et des liens en français.

– **Build**
  Les variables configurent ici l'extraction de la documentation : quels éléments doivent être documentés et référencés, le tri des informations, etc.

– **Messages**
  Il s'agit de la configuration de la génération du fichier de log, pour avoir un Doxygen plus ou moins bavard.

– **Input**
  Il est possible d'affiner ici la spécification des fichiers scannés par Doxygen. Typiquement, s'il y a autre chose que des fichiers sources dans l'arborescence, autant le signaler. INPUT_ENCODING est une variable utile pour un bon affichage. En effet, si on ne met pas le bon encodage des fichiers sources et qu'il y a des caractères un peu particuliers (accents par exemple) dans la documentation, on se retrouvera avec une documentation pas très agréable à regarder. Par défaut c'est UTF-8 mais essayer ISO-8859-1 éventuellement si vous avez des soucis d'affichage d'accents.

– **Source Browser**
  Ces variables permettent de configurer les fichiers sources générés en HTML et les références correspondantes dans/vers la documentation.

– **Index**
  Activer ALPHABETICAL_INDEX permet d'avoir un index des classes/structures par ordre

alphabétique.

– **HTML, LaTeX, RTF, Man, XML**
  On a la possibilité de personnaliser le style des documents en sortie (cadres HTML par exemple).

– **Dot**
  La génération des graphes se paramètre ici. On retrouve les différents graphes pouvant être générés. Si on veut utiliser graphviz, s'assurer que dans DOT_PATH se trouve le bon chemin du genre ".../graphviz/bin".

1.4.C Run

Une fois que vous avez tout renseigné suivant l'une des deux configurations, vous pouvez sauvegarder vos choix. Cela crée un Doxyfile et cela permettra de le recharger plus tard.

Si vous avez effectué un changement de configuration, il est parfois nécessaire d'effacer les fichiers générés précédemment pour que la nouvelle configuration fonctionne comme elle le devrait.

Ensuite, lancer la génération avec le bouton "Run Doxygen". Le fichier de log s'affiche, vérifier qu'il n'y a pas d'erreurs et vous pouvez ensuite visualiser la sortie en HTML directement.