Ontwikkelingshulp voor PL/SQL (3)

Technische documentatie PL/SQL applicaties

In een serie van vier artikelen geeft Lucas Jellema, technisch consultant bij AMIS, een overzicht van ideeën, patronen, richtlijnen en tools die PL/SQL ontwikkelaars kunnen overnemen uit de wereld van Java en J2EE. Hij gaat onder meer in op concepten uit de Java programmeertaal die een toepassing hebben in PL/SQL maar ook op open source-tools voor onder meer logging, unit testen, automatiseren van batch-operaties en generatie van documentatie. Dit artikel is bestemd voor PL/SQL ontwikkelaars; kennis van Java is niet vereist. De source code voor dit artikel kan worden gedownload van http://www.amis.nl/tech_artikelen.php?id=158.

We zullen in dit artikel ingaan op open source tools voor generatie van technische documentatie bij respectievelijk PL/SQL modules (PLDoc en Natural Language) en Forms (FoReDoclet). In het volgende artikel zullen we kennismaken met het door Oracle als 'vlaggenschip voor database design en development' naar voren geschoven tool JDeveloper. We kijken dan naar de betekenis van JDeveloper als ontwikkelgereedschap voor SQL en PL/SQL development. Ook kijken we naar enkele open source alternatieven voor ondermeer TOAD en PL/SQL Developer.

Technische documentatie

Vrijwel geen ontwikkelaar vindt het leuk om documentatie te schrijven bij zijn code en het gevaar dat code en documentatie niet meer synchroon zijn is altijd levensgroot. Als dan iedere ontwikkelaar zijn eigen aanpak en opmaak kiest voor documentatie en er ook nog eens geen tool-ondersteuning is (anders dan Word) om documentatie te maken, zijn de kansen op volledige en bruikbare technische documentatie van broncode wel erg klein. JavaDoc biedt daar voor Java-broncode in grote lijnen een oplossing voor.
Mijn eerste kennismaking met Java was waarschijnlijk door JavaDoc van een of ander Oracle-tool waar ik per ongeluk in belandde. JavaDoc is een begrip onder Java-ontwikkelaars: de technische documentatie van Java-code die op basis van de code – en kleine annotaties in de code – volledig voor je wordt gegenereerd in een lay-out die voor alle Java programmatuur is gestandaardiseerd. Al schrijf je geen regel commentaar in je code, dan nog genereert JavaDoc een zeer acceptabele set documentatie. De documentatie wordt geproduceerd in de vorm van HTML die in elke browser getoond kan worden en via webservers eenvoudig op internet kan worden gepubliceerd. Door het gestandaardiseerde formaat weet ook elke Java ontwikkelaar de weg in JavaDoc.
De meeste Java-ontwikkeltools zoals JDeveloper en Eclipse hebben JavaDoc geïntegreerd: een druk op een knop is voldoende om de documentatie opnieuw te genereren op basis van de programmacode. Het gevolg van het gemak en de kracht van JavaDoc is dat voor vrijwel ieder Java-project – hoe klein en onbeduidend ook - technische documentatie wordt opgeleverd. In die documentatie zijn dan in elk geval een gestructureerd overzicht van alle packages aanwezig, evenals classes, methodes en parameters, exceptions en return waarden.
Afhankelijkheden tussen code en referenties naar andere secties van de documentatie kunnen eenvoudig met hyperlinks worden geïmplementeerd. Je kunt HTML-opmaak gebruiken binnen commentaar in de code- bijvoorbeeld vet, onderstreept of zelfs een <table>. Je kunt ook eigen footers, headers en logo's opnemen in de gegeneerde documentatie. Met Cascading Stylesheets (css-files), kun je de look & feel nog verder naar je hand zetten.

Resources

PL/SQL heeft nooit een standaardisatie gekend voor wat betreft technische documentatie, laat staan dat er een tool vergelijkbaar met JavaDoc voor PL/SQL bestond. Je zou zelf natuurlijk iets kunnen ontwikkelen, bijvoorbeeld uitgaande van de PL/SQL source code die is terug te vinden op de volgende plaatsen:

- PL/SQL modules in Oracle Designer (Repository)
- PL/SQL objects in de Oracle Database (Data Dictionary) (USER_SOURCE en USER_ARGUMENTS views of met de dbms_metadata en dbms_utility.describe_procedure packages
- PL/SQL source files

Met name de eerste twee opties kunnen ook een in online modus worden gebruikt: de documentatie wordt steeds op aanvraag gegenereerd en is dus altijd actueel, dat wil zeggen: gebaseerd op de meest recente broncode. Op basis van de PL/SQL source code zijn dan ondermeer de volgende opties beschikbaar om tot JavaDoc-achtige documentatie voor PL/SQL te komen:

- Het freeware tool SchemaSurf (http://www.cobblesoft.com/schemasurf/) en de Repository Object Browser (onderdeel van Oracle Designer en Oracle SCM) - bieden beide browser-gebaseerde, online raadpleegfunctionaliteit, uitgaande van respectievelijk de data-dictionary en de Designer Repository.
- De (commerciële) plugin voor PL/SQL Developer (zie: http://www.allroundautomations.com/plsqldev.html)
- Het open source tool DDLWizard (http://www.ddlwizard.com/) dat DDL statements extraheert uit Database Export (.dmp) files en die desgewenst converteert in gekoppelde HTML pagina's
- Een zelfgebouwde oplossing, waarschijnlijk idealiter gebaseerd op XML/XSLT technologie; mijn collega Harm Verschuren heeft de aanzet voor zo'n soort implementatie gedemonstreerd voor PL/SQL definitions in Oracle Designer tijdens de ODTUG conferentie (zie ook zijn artikel op: http://www.amis.nl/tech_artikelen.php?id=117)

Karakteristieken van PLDoc

Home Page	http://sourceforge.net/projects/pldoc
Bestaat sinds	Oktober 2001
Meeste recente release	0.8.2 (17 Juli 2004) – ondersteunt PL/SQL 9i grotendeels
Status	Alpha
Onderliggende technologie	Java, XML en XSLT
Ontwikkelteam	3
Zip-grootte	2,7 MB
Licentie	LGPL (Lesser General Public License) (zie ook: http://www.opensource.org/licenses/lgpl-license.php)
Gerelateerde technologie en concepten	JavaDoc (zie http://java.sun.com/j2se/javadoc/writingdoccomments/index.html)

Het open source project PLDoc (zie: http://sourceforge.net/projects/pldoc) Met PLDoc is voor PL/SQL ontwikkelaars een tool beschikbaar met wel vrijwel dezelfde kenmerken en mogelijkheden als JavaDoc, gebaseerd op XML/XSLT en Java.

Meer over PLDoc

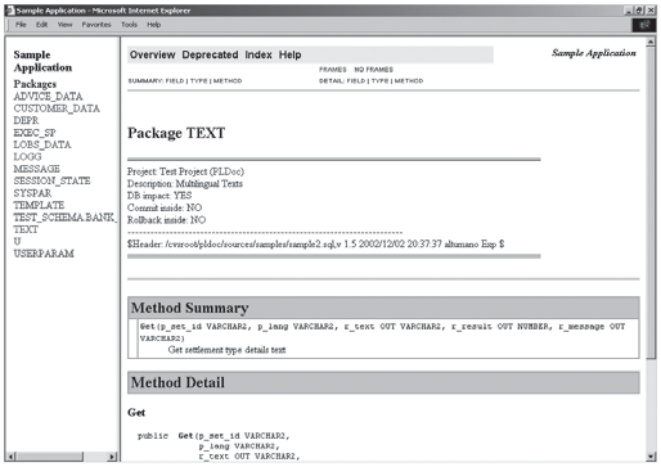
PLDoc is weer een open source project voor PL/SQL ontwikkelaars. Je kunt het terugvinden en downloaden op SourceForge, de grootste open source repository op internet: meer dan 1 miljoen ontwikkelaars met meer dan 100.000 open source projecten. SourceForge bestond in november 2004 precies tien jaar.

PLDoc kent een simpele installatie procedure: je moet een zip-file downloaden en deze naar een geschikte directory uitpakken. Om voor een set PL/SQL source-files de documentatie te genereren, moet je zorgen dat deze files samen in een directory staan. Vervolgens voer je het volgende command line commando uit:

```
pldoc.bat -d MyApp/doc MyApp/ddl/*.pck
```

-d geeft de doel-directory aan, waar de te genereren documentatie wordt weggeschreven; met andere parameters kan je PLDoc aangeven welk stylesheet moet worden gebruikt voor de generatie, of PL/SQL en SQL reserved words altijd in hoofd- of kleine letter moeten worden weergegeven et cetera. De laatste parameter in het command line commando geeft aan voor welke source files de documentatie moet worden gegenereerd[1].

De gegenereerde documentatie heeft dezelfde opbouw als JavaDoc: een overzicht van alle packages waarvandaan je kunt navigeren naar een specifiek package. Voor dat package wordt een lijst van alle procedures en functies getoond met de mogelijkheid daarop in te zoomen. PLDoc genereert al nette docu-



Afbeelding 1. Voorbeeld van PL/SQL documentatie in JavaDoc style, gegenereerd door PLDoc

mentatie puur op basis van de PL/SQL definities. Deze documentatie kan verder uitgebreid worden met commentaar dat de ontwikkelaar in de code toevoegt – voorafgaand aan iedere procedure of functie en bij voorkeur tussen /** en */.

Voorbeeld

De broncode voor de supplied database packages – zoals dbms_output, dbms_utility en utl_file – is te vinden in de ORACLE_HOME\rdbms\admin\ van een Oracle database installatie. We gaan met PLDoc PL/SQL-documentatie genereren voor een aantal van deze packages. De package specificatie files voor een aantal packages – dbms_alert, dbms_utility, dbms_job, dbms_lob, dbms_output en utl_file - kopieer ik naar een aparte directory: PLDOC_HOME\SuppliedPackages.
Vervolgens genereer ik de documentatie met PLDoc:

```
pldoc -d .\SuppliedPackages\api .\SuppliedPackages\*.sql
```

Het resultaat is een keurige set HTML documenten die een uitstekend overzicht biedt in de functionaliteit van deze supplied packages. Zie http://www.amis.nl/files/technology/pldoc/SuppliedPackages/api/ voor een online demonstratie van deze documentatie. De Oracle-ontwikkelaars hebben redelijk wat commentaar toegevoegd in de package specificatie zodat goed leesbare documentatie is gecreëerd.
Overigens heb ik de source files licht moeten editen: mijn database is release 10g en niet alle 10g syntax wordt herkend door de PLDoc parser. Daarnaast kan PLDoc niet omgaan met meer elementen in een source-file dan alleen het package specificatie statement. Het is waarschijnlijk makkelijk om package specificaties uit de database te ' downloaden' met bijvoorbeeld TOAD of PL/SQL Developer of een zelfgeschreven tool. Een zelfde test met de Oracle Designer repository packages (ORACLE_HOME/repadm61/api en /jin op een Oracle Designer Client PC) levert ook prima resultaten, zie: http://www.amis.nl/files/technology/pldoc/OracleSCM/api/

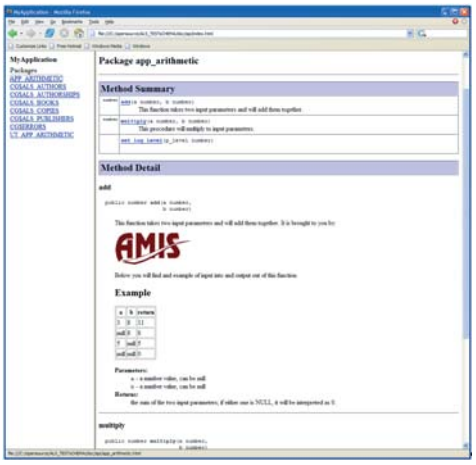
Speciale tags

PLDoc ondersteunt een aantal van de standaard JavaDoc tags. Deze worden gebruikt om de documentatie verder de standaardiseren en structureren. Deze tags kunnen worden opgenomen aan het eind van het commentaar. Het gaat om:
@throws – om aan te geven welke exceptions kunnen optreden in een procedure of functie
@return – om de waarde die een functie teruggeeft nader toe te lichten
@deprecated – geeft aan date en procedure of functie uitsluitend voor 'backward compatibility' is opgenomen en niet meer voor nieuwe aanroepen moet worden gebruikt
@param – om afzonderlijke input- en output-parameters toe te lichten

HTML-tags die in het commentaar worden opgenomen worden regelrecht overgenomen in de gegenereerde documentatie; daarmee kan dus in het commentaar de lay-out flink gestuurd worden. Een nuttig voorbeeld daarvan zijn referenties naar andere packages of procedures of functions in hetzelfde of andere packages (vergelijkbaar met de JavaDoc @see tag, die PLDoc nou net niet ondersteunt). Via de HTML <a> tag kan je al in het commentaar in de programmacode een hyperlink in de gegenereerde documentatie voorbereiden[2].
Een voorbeeld van gegenereerde documentatie bij toepassing van speciale tags is te zien in afbeelding 2. Het hierna volgende stuk PL/SQL-code bevat voorbeelden van het gebruik van specifieke tags en HTML binnen het code commentaar:

```
CREATE OR REPLACE package app_arithmetic
as

/**
  This function takes two input parameters and will add them together.
It is brought to you by:<p/><img src="http://technology.amis.nl/blog/
wp-content/images/AmisRed.png"/><p/>
  Below you will find and example of input into and output out of this
function
  <h2>Example</h2>
  <table border=1>
  <tr><th>a</th><th>b</th><th>return</th></tr>
  <tr><td>3</td><td>8</td><td>11</td></tr>
  <tr><td>null</td><td>8</td><td>8</td></tr>
  <tr><td>5</td><td>null</td><td>5</td></tr>
  <tr><td>null</td><td>null</td><td>0</td></tr>
  </table>
  <p/>
  @param a a number value, can be null
  @param b a number value, can be null
  @return the sum of the two input parameters; if either one is NULL,
it will be interpreted as 0.
  */
function add( a in number, b in number)
return number
;

/**
 This procedure will multiply to input parameters. If a parameter is
NULL, it will be interpreted as 0 (zero). Any
 call involving a NULL parameter will therefore result in 0 as return
value. Also see <a href="app_arithmetic.html#add(number,number)">See:
app_arithmetic.add(a,b)</a>

 @throws no exceptions
 @param a first factor in multiplication
 @param b second factor in multiplication
 @return the product (multiplication result) for the two input parame-
ters; if either one is NULL, the result will be 0


*/
function multiply( a in number, b in number)
return number
;

procedure set_log_level(p_level in number);

end;
```

Adv. Array Seminars



Afbeelding 2. Een voorbeeld van gegenereerde documentatie bij toepassing van speciale tags

Technische achtergrond

De eerste stap die PLDoc zet is het parsen van de PL/SQL Code die vanuit een file wordt gelezen. Aangezien PLDoc open source is en je dus de Java-code tot je beschikking hebt zou je eenvoudig kunnen ingrijpen in deze stap: de parser zou ook een rechtstreeks uit de database gelezen package specificatie kunnen gebruiken als bron; dit maakt de tussenstap van source files overbodig! Overigens: als je aanpassingen en verbeteringen maakt zou je die aan het PLDoc project moeten aanbieden – zo werkt open source!

De parser die wordt gebruikt is ontwikkeld op basis van JavaCC (zie: https://javacc.dev.java.net/). Dit is een open source tool dat source code parsers genereert. JavaCC wordt aangestuurd met een 'grammatica beschrijving' van de te parsen programmeertaal. Er wordt mij wel eens gevraagd of de grammatica van PL/SQL in het zogenaamde BNF (Backus-Naur Form) formaat. Dat ken ik niet, maar PLDoc wordt geleverd met een vrijwel equivalente beschrijving in JavaCC formaat. De parsestap levert hetzij een foutmelding hetzij een XML-document met daarin de geparste PL/SQL code structuur, met XML Elementen voor alle packages, procedures en functions, constants, comment et cetera. Met deze structuur zou je overigens nog veel meer doen dan alleen documentatie genereren. Denk bijvoorbeeld aan transformatie naar een andere programmeertaal, controle van PL/SQL programmeerstandaarden, upgrade naar een nieuwere versie van PL/SQL et cetera.

In de tweede stap wordt het XML Document door middel van een XSLT transformatie door de Open Source XSLT engine Apache Xalan gegenereerd tot documentatie. Het is betrekkelijk eenvoudig om de XSLT-stylesheets uit te breiden om de layout of inhoud van de gegenereerde documentatie verder naar je hand te zetten. Je kunt op die manier zelf ondersteuning voor de @see tag inbouwen of op een andere manier de gegeneerde documenten naar je hand zetten.

PLDoc wordt geleverd met zijn eigen ANT-task definitie (zie het vorige artikel uit deze serie voor meer achtergronden bij ANT). Door gebruik van deze taak kan de generatie van PL/SQL-documentatie eenvoudig onderdeel worden gemaakt van het nachtelijk build-proces. Het is betrekkelijk eenvoudig deze task vooraf te laten gaan door een task die de sources verzameld voor PLDoc, vanuit de database, een source code control systeem of ergens op het filesysteem.

Natural Docs

Een ander open source project op het vlak van generatie van technische documentatie is Natural Docs. Het is met name interessant als je met meerdere programmeertalen werkt binnen een organisatie. Natural Docs werkt met een groot aantal programmeertalen, waaronder PL/SQL maar ook C#, C, Perl, Java, JavaScript, Python, en vrijwel alle andere talen die ik ken. Daarnaast kan je eenvoudig nieuwe talen toevoegen. Het sterke punt van Natural Docs is dat je voor alle programmeertalen dezelfde wijze van documenteren kunt hanteren: met hetzelfde

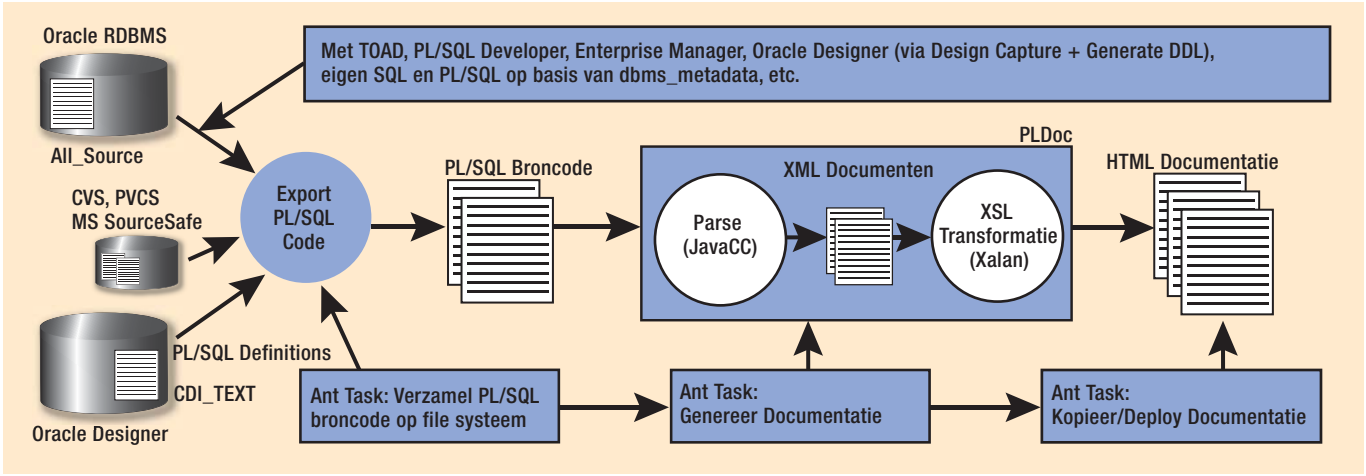
Supplied package dbms_metadata

Een eenvoudige manier om binnen SQL en PL/SQL aan de source code van ondermeer Packages in de database te komen is beschikbaar met behulp van het package dbms_metadata (vanaf Oracle 9i). Naast complete DDL scripts voor tabellen, views, types en andere database-objecten kun je met dit package ook het volledige 'create or replace package X as...' statements verkrijgen voor alle packages waar je tenminste execute rechten op hebt. Geen gehannes meer met ALL_SOURCE!

Een voorbeeld van hoe je de code verkrijgt – en afdrukt - van een package NUMBER_API in het SCOTT schema ziet er als volgt uit:

```
DECLARE
   c CLOB;
procedure pl(p_txt in clob)
is
begin
  dbms_output.put_line(substr(p_txt,1,255));
  if length(p_txt)> 255
  then
    pl(substr(p_txt, 256));
  end if;
end;

BEGIN
   c := DBMS_METADATA.get_ddl('PACKAGE', 'NUMBER_API','SCOTT');
   pl(c);
END;
```

Je kunt gebruik van dbms_metadata combineren met utl_file om de PL/SQL source files naar een directory op de database server te schrijven. Daar kan dan vervolgens PLDoc tegen gedraaid worden.



Afbeelding 3. Technische achtergronden van PLDoc

soort commentaar kan uit alle talen gelijksoortige documentatie worden gegenereerd. NaturalDocs kent een groot aantal keywords die je in het commentaar in de code kan gebruiken om in de gegenereerde documentatie bijvoorbeeld handige verwijsindexering of een trefwoordenregister op te nemen of de lay-out aan te scherpen. De standaard meegeleverde look & feel ziet er ook professioneel en wat verzorgder uit dan bij PLDoc.

Het grote nadeel van deze aanpak is dat de taalspecifieke ondersteuning vrijwel ontbreekt: als je NaturalDocs loslaat op code die niet specifiek gericht op NaturalDocs van commentaar is voorzien komt er geen bruikbare documentatie uit – dat in sterke tegenstelling tot PLDoc. Neem je de moeite om voor alle procedures commentaar toe te voegen – waarbij het commentaar ondermeer de naam van de procedure zelf moet bevatten omdat NaturalDocs geen PL/SQL Syntax kent- dan kan je een fraai eindresultaat bereiken.

Oracle Forms

Wat PLDoc kan doen voor PL/SQL broncode voor packages kan een ander open source tool – FoReDoclet – doen voor Forms modules. Dit omvat niet alleen de FMB maar ook de Object Library (OLB), Menu Module (MMB) en Program Library (PLL) files. FoReDoclet is geschreven in Java en maakt ook gebruik van XML-technologie. Het tool leest Oracle Forms modules via de Java Development API van Forms (beschikbaar sinds Forms 9i) en genereert HTML-documenten. Al heb je geen letter commentaar opgenomen in je Forms, dan nog levert FoReDoclet zeer waardevolle en leesbare documentatie op. Hierin zijn alle componenten van een Form hiërarchisch – drill down via hyperlinks – georganiseerd: Blocks, Items, Triggers, Program Units et cetera. De gegenereerde documenten doen weer sterk denken aan JavaDoc. Overigens geldt ook hier dat door het Cascading Stylesheet aan te passen de look & feel van de documentatie sterk beïnvloed kan worden.

Afbeelding 4 toont de documentatie die is gegenereerd voor het MULTISELECT90 form dat als sample met Forms wordt meegeleverd. Zie http://www.amis.nl/technology/foredoclet

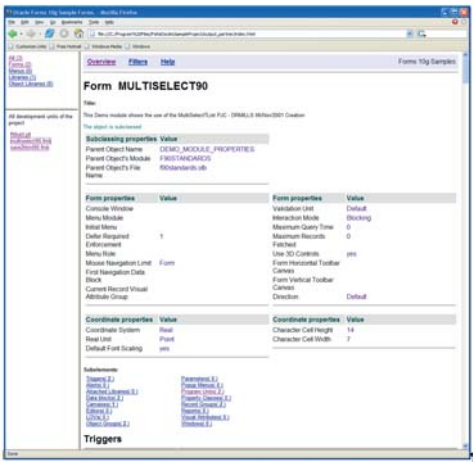
Karakteristieken van NaturalDocs

Natural Docs

Homepage	http://www.naturaldocs.org/ en http://sourceforge.net/projects/naturaldocs
Bestaat sinds	Mei 2003
Meeste recente release	1.32 (November 2004)
Status	Productie
Onderliggende technologie	(Opti)Perl
Ontwikkelteam	1 (geregistreerd), groot aantal gebruikers
Zip-grootte	330 Kb
Licentie	GPL (General Public License) (zie ook : http://www.opensource.org/licenses/gpl-license.php)
Gerelateerde technologie en concepten	Heeft Perl nodig (zie bijvoorbeeld http://www.activestate.com/Products/ActivePerl/)

Een open source tool: NaturalDocsWizard maakt het mogelijk om NaturalDocs via een Grafische interface te starten, in plaats van vanaf de commandline: zie http://www.codetools.com/tools/NaturalDocsWizard.asp

Voor organisaties die vrijwel uitsluitend met PL/SQL en eventueel Java werken zou ik PLDoc en voor Java uiteraard JavaDoc aanraden.



Afbeelding 4. De documentatie die is gegenereerd voor het MULTISELECT90 form dat als sample met Forms wordt meegeleverd

voor een uitvoeriger voorbeeld van gegenereerde Forms documentatie.

Ook FoReDoclet herkent HTML opmaak en JavaDoc tags (@return, @param, @throws) en verwerkt deze in de gegenereerde documentatie. Daarnaast kan via externe documenten en een parameterfile invloed op de gegenereerde documentatie worden uitgeoefend: stukken tekst en illustraties kunnen worden ingevoegd. Een mogelijke toepassing van deze functie is het opnemen van een screenshot van een form, een block of een canvas in de documentatie om deze toegankelijker te maken voor ontwikkelaars. Het nadeel daarvan is dat met iedere aanpassing het screenshot vervangen zou moeten worden.
De meest waardevolle toepassing van FoReDoclet en de technische documentatie van Forms ligt, denk ik, bij PLL's en Object Library's: van elementen die je als ontwikkelaar overweegt te hergebruiken – program units of objecten om van te subclassen – is het nuttig om aanvullende informatie te krijgen.
Documentatie van complete Forms is mooi en zal de QA-inspecteur tevreden stellen, maar is in de praktijk niet zo heel waardevol – hoogstens voor de testers. Aan de andere kant, met FoReDoclet is het zo eenvoudig op te leveren dat je het maar gewoon moet doen.

Installatie

Forms Builder 9i of 10g – of meer specifiek de Forms JDAPI (Java Development API) moet geïnstalleerd zijn om met FoReDoclet te werken. De development omgeving kan wel 6.0 of 6i zijn: fmb's of pll's ontwikkeld met 6.0 of 6i kunnen prima door FoReDoclet verwerkt worden. De documentatie voor FoReDoclet bestaat uit niet meer dan twee voorbeeldprojecten. Wel kun je bij Nostran Computing een complete set documentatie aanschaffen, voor $48,75.
De installatie van FoReDoclet is even zoeken aangezien er geen enkele documentatie bij wordt geleverd – tenzij je de commerciële documentatie aanschaft. Je kunt een executable jar-file downloaden (foredoclet-1.1-setup.jar). Deze moet je runnen: er wordt een installer gestart. Deze leidt je door een aantal wizard-pagina's waarna de daadwerkelijk installatie plaatsvindt. Onder FoReDoclet_HOME vind je de subdirectory sampleProject met daarin een aantal voorbeeld forms en library's alsmede een tweetal bat-files die kunnen worden gebruikt om FoReDoclet voor die forms en library's de documentatie te laten genereren. Door deze bat-files en de bijbehorende .fpp (parameter) files te editen voor je eigen Forms project kan je met FoReDoclet aan de slag gaan.

Karakteristieken van FoReDoclet

Homepage	http://foredoclet.sourceforge.net/ of http://sourceforge.net/projects/foredoclet/
Bestaat sinds	2002 (FoReDoclet is zijn leven als commercieel product begonnen en is in 2003 open source gemaakt)
Meeste recente release	1.1.6 (18 december 2003)
Status	Productie
Onderliggende technologie	Java, XML (Velocity, Avalon)
Ontwikkelteam	1
Zip-grootte	3,7 MB
License	GPL (GNU General Public License) (zie ook: http://www.opensource.org/licenses/gpl-license.php)
Gerelateerde technologie en concepten	JavaDoc (zie http://java.sun.com/j2se/javadoc/writingdoccomments/index.html), Velocity en Avalon (zie http://jakarta.apache.org)

Lucas Jellema is sinds 1994 werkzaam met Oracle- en later ook Java-technologie, eerst bij Oracle en sinds 2002 bij AMIS Services B.V. in Nieuwegein in de rol van Technisch Consultant. Met collega's onderhoudt hij een weblog (http://technology.amis.nl/blog/) rondom Oracle- en Java-technologie. Voor verdere vragen kan hij worden bereikt per e-mail: jellema@amis.nl.

1 PLDoc kijkt uitsluitend naar Package Specifications. Op dit moment wordt geen documentatie gegenereerd voor tabellen, views, package body's of andere objecten. De gegenereerde documentatie bevat ook alleen gegevens over 'publieke' procedures en functies, die opgenomen zijn in de package specificatie.
2 Bijvoorbeeld: Zie Package.Procedure(), meer concreet: Zie: cioapplication_system.sel(id,pl)

Adv. Quest