The Vex Cookbook John W Austin

Table of Contents

1. Writing for DocBook using Vex

Problem

You want to author a DocBook document using Vex so that you can see what it will look like when you publish it

Solution

Edit a DocBook document using Vex.

Discussion

You must install Vex as described in Section 2, “Obtaining and Installing Vex” . You may then select one of the following options to begin editing with Vex:

1. Locate an XML document that uses a version of DocBook that is supported by the version of Vex you installed. Determine whether the DOCTYPE information in your document is compatible with Vex by importing the document into a project in your workspace and open it. You may see a dialog asking you to select a DTD. Choose a reasonable one for now and we will correct your document later.

2. Import the vex-sample documents as described in Section 3, “Importing the Vex Samples” and choose one of the DocBook examples. Open the document with the context menu (right-click) or double-click on the file in the Navigator view.

3. Create a new DocBook document as described in Section 5, “Creating a new DocBook document in

Vex”.

See Also

Section 6, “Publishing DocBook while using Vex”.

2. Obtaining and Installing Vex

Problem

You want to use Vex on a computer that doesn't have it installed.

Solution

You need to download and install the Vex package from the project web site at: `http://vex.sourceforge.net/`.

Discussion

Vex requires Java which must be equivalent to the Java 2 Software Development Kit (J2SDK) or Runtime Environment (J2RE) from Sun Microsystems: `http://java.sun.com/` at release 1.4 or better. If you think you may have Java installed, you can check by running the command: **java -version**.

The latest Vex software is available for download from the web site: `http://sourceforge.net/project/showfiles.php?group_id=67542`. Select the version of Vex applicable to your operating system.

Note

At 26Mb, Vex is a relatively large download. That's mostly due to the Eclipse platform on which Vex is based. However, once you've installed Vex, future updates are available through the Eclipse Software Update facility, and are therefore much smaller (~2Mb).

Unzip the archive to a directory on your hard disk and issue the command: **./vex** from within the directory.

As Vex is starting, it will prompt you for the location of your `workspace`. The workspace is a directory that holds the Vex documents on which you will be working. By default, Vex creates the `workspace` directory as a subdirectory of the installation directory. Once you have selected a workspace, you are presented with the Vex main window.

In Linux, you may add an icon for Vex to the Gnome panel and launch from there. To create this: Right-click Add to Panel->Launcher and fill out the form which appears. You will give the item a name, generic-name, comment, command, type and icon.

Name	A name that appears in a menu created for this launcher or on the Desktop. 'Vex 1.1.1' is an obvious choice.
Generic name	A name for the class of application to which this launcher belongs. 'XML Editor' is a good choice.
Comment	A short description of the launcher which appears in tooltip text when you point to the launcher icon on the panel. 'Visual Editor for XML (1.2.1)' is a good choice.

Command	A command to be executed by this launcher. You will use the Browse button to track down the `vex` executable inside your vex installation directory. For example: ${HOME}/vex-1.2.1/vex could be a reasonable location.
Type	Specify 'Application' because this launcher invokes Vex.
Icon	Choose an icon to represent the launcher. There are a few nice icons provided with Vex. I use `plugins/net.sf.vex_1.2.1/vex48.gif` where the name is relative to my vex installation directory.

See Also

See the Vex Project Home Page: `http://vex.sourceforge.net/`.

3. Importing the Vex Samples

Problem

You need to see working code.

Solution

Import the Vex Samples into your workspace.

Discussion

The *Vex Samples* are a sub-project of Vex and are built into the deliverable product. Anyone who installs Vex stand-alone, or runs the Vex plug-in under Eclipse, may access the samples from the menu bar. Simply import the samples with File -> Import and select 'Vex Samples' from the Import Wizard and press the 'Next' or 'Finish' button.

4. Creating a new Project

Problem

You need to start a new documentation project to organize your DocBook documents.

Solution

Create a new project in your workspace using File->New->Project .

Discussion

Vex is an Eclipse plug-in and follows the Eclipse conventions for organizing all development artifacts. In the case of Vex, development artifacts include XML documents, stylesheets and DTD files. In the case of DocBook, IBM DITA and XHTML 1.0 files, the DTD and stylesheets are built-in.

If you wish to development new DocBook documents using Vex, you must provide a 'project' as a container in the workspace.

See Also

Section 1.5 in *Eclipse Cookbook* (O'Reilly)

5. Creating a new DocBook document in Vex

Problem

You need to create a new DocBook document using Vex.

Solution

Invoke the Vex New Document Wizard and complete a simple dialog.

Discussion

Vex provides a New Document Wizard that will assist the creation of a new document. You can invoke this wizard by selecting one of the following: (**CTRL-N**) and Other (**O**); (**SHIFT-ALT-N-O**) or File->New->Other followed by expansion of the Wizard's Vex folder and selection of Document.

The first page of the Vex New Document Wizard asks you to select from two combo boxes (selection lists) labelled: "Document Type" and "Root Element". These allow Vex to create the document by selecting a document type from the set of DTD's registered with Vex and a root element for the document which is a valid root element for the selecxted DTD.

The second page of the Wizard asks you to select a parent folder and a name for the new file. The parent folder is a subfolder of the current workspace. You may navigate by expanding folders in to a tree and selecting one folder to contain the new file. The file name may be any valid name for your operating system and should respect file naming practices for your environment. In my case, I use the filename extension: `xml`.

See Also

Section 1.8 in *Eclipse Cookbook* (O'Reilly)

6. Publishing DocBook while using Vex

Problem

You want to publish a DocBook document that you are editing with Vex.

“Hello.”

Solution

Use the DocBook XSL stylesheets and a shell script or the Cocoon servlet to produce output in one of the formats supported by Apache Fop.

Discussion

To publish your document, you may use the DocBook XSL style sheets described in Section 8, “Installing the DocBook Stylesheets for use by applications such as Cocoon.”. The style sheets are part

of a 'tool chain' which transforms your document into a publishing format such as a web page, Rich Text Format (RTF) or a PDF file. One example of a 'tool chain' is the XML tool set provided by The Apache Foundation. This includes Xalan, Fop, Cocoon and Tomcat which are all implemented in Java and should work in the same places as Eclipse (and Vex).

The style sheets may be applied to a DocBook source file using Xalan. This may be done through a stand-alone command line, a shell script, as an Ant build target or embedded in a larger application such as the Cocoon framework. Cocoon may be used as a stand-alone processor, a web server or packaged in a larger web application environment such as Tomcat. The Tomcat servlet container may run by itself or packaged within the context of a full-featured web server such as Apache.

The output of style sheet processing may used directly by the end-user as in the case of a web page or a text document. It may also be an intermediate format such as XSL Formatting Objects which requires further processing by a tool such as Apache Fop. Fop is used so often that it includes Xalan and XSL transformations as a form of input, is bundled with Cocoon and may therefore also be used from web applications running in Cocoon, Tomcat and Apache.

You may build a suitable publishing environment by installing Cocoon and connecting a Vex Plug-in Project to a directory within a Cocoon web application. For example, install Cocoon in your home directory as: `cocoon-2.1.6`. Build Cocoon here and place a new directory inside at `build/webapp/cocoon`. Call this directory `docbook` and place the following cocoon file: `sitemap.xmap` inside it.

Example 1. Cocoon `sitemap.xmap` for Vex DocBook publishing

```
<!-- Code example modified from "DocBook XML/SGML Processing Using OpenJade"
     Copyright (c) 2001 Saqib Ali -->

<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">

    <!-- use the standard components -->
    <map:components>
        <map:generators default="file"/>
        <map:transformers default="xslt"/>
        <map:readers default="resource"/>
        <map:serializers default="html"/>
        <map:selectors default="browser"/>
        <map:matchers default="wildcard"/>
    </map:components>

    <map:pipelines>
        <map:pipeline>

    <!-- respond to *.html requests with our docs processed by .xsl -->

            <map:match pattern="*.html">
                <map:generate src="{1}.xml"/>
                <map:transform
                     src="/usr/local/dbtools/docbook-xsl/html/docbook.xsl"/>
                <map:serialize type="html"/>
            </map:match>

            <map:match pattern="*.xhtml">
                <map:generate src="{1}.xml"/>
                <map:transform
                     src="/usr/local/dbtools/docbook-xsl/xhtml/docbook.xsl"/>
                <map:serialize type="html"/>
            </map:match>

    <!-- later, respond to *.pdf requests with our docs processed by .xsl -->
```

```
            <map:match pattern="*.pdf">
                <map:generate src="{1}.xml"/>
                <map:transform
                     src="/usr/local/dbtools/docbook-xsl/fo/docbook.xsl"/>
                <map:serialize type="fo2pdf"/>
            </map:match>

            <map:match pattern="*.xml">
                <map:generate src="{1}.xml"/>
                <map:serialize type="xml"/>
            </map:match>

           <map:match pattern="*.png">
               <map:read mime-type="images/png" src="{1}.png"/>
           </map:match>

        </map:pipeline>
    </map:pipelines>
</map:sitemap>
```

A Vex Project can be connected to documents in this Cocoon subdirectory by adding a linked folder to a project in the Vex workspace. Use: File->New->Folder (**Shift-Alt-N**) to bring up a 'New Folder' dialog. Fill out the dialog that comes up. Select Advanced to expose a checkbox labeled: Link to folder in the file system and use the Browse button to find a folder to link.

Parent	A folder in the Vex project which will appear to contain the folder we are creating. Use `docs` in this case.
Linked folder	A folder in the file system that will contain resources used by the Vex project.

Linking files in your Vex project workspace to a directory in the Cocoon workspace will allow you to edit DocBook files in Vex and publish them with Cocoon.

Figure 1. This Vex Cookbook was written using Vex and is published as described herein.

You may start Cocoon as a stand-alone server (**./cocoon.sh servlet**) and use it to publish DocBook documents (such as this Cookbook) by pointing a web browser at your workstation or a Cocoon server: `http://localhost:8888/docbook/VexCookbook.pdf`.

This recipe links together software from several significant Open Source projects which are backed by substantial development and user communities.

Open Source Projects which contribute to this publishing solution:

Vex	Visual Editor for XML at *Sourceforge.net*
Eclipse	The Eclipse IDE at *Eclipse.org*
Apache Tools	Xerces, Xalan, Cocoon, Tomcat and Apache at *Apache.org*
DocVBook XSL	DocBook Open Repository at *SourceForge.netg*

See Also

Section 7, “Installing Cocoon to run on Linux”, Section 8, “Installing the DocBook Stylesheets for use by applications such as Cocoon.”. Chapter 11 in *Eclipse Cookbook* (O'Reilly).

7. Installing Cocoon to run on Linux

Problem

You need to install the Cocoon publishing framework on a Linux host to publish your XML documents.

Solution

Download, customize, build and install the Cocoon framework from the Apache XML web site.

Discussion

The Cocoon publishing framework is an Apache project and may be located at the web site: `http://cocoon.apache.org`. Locate the latest stable release of the project and download it to your machine. The download consists of source code, configuration files and build scripts in a compressed archive file. You must extract the files from the downloaded archive, optionally customize the configuration for your own purposes, build the project and either start the stand-alone server or deploy the cocoon.war into a servlet container such as Tomcat.

In my environment, the build did not succeed the first time. The build scripts were designed to work with older versions of Java and I wanted to use Java 1.5.0. After consulting the wisdom of the 'net', I was able to modify a single file and complete the build.

Example 2. Corrections to Cocoon 2.1.6 build files needed to build under Java 1.5.0

```
    The following changes were required in the file:
    cocoon-2.1.6/tools/targets/init-build.xml

    <!-- compile the ant tasks -->
    <mkdir dir="${tools.tasks.dest}"/>
    <javac srcdir="${tools.tasks.src}"
           destdir="${tools.tasks.dest}"
           debug="off"
           optimize="on"
           deprecation="on"
           target="1.5"    <=== changed from 1.3 in Cocoon 2.1.6
           nowarn="on"
           compiler="${compiler}"
           classpathref="tasks.classpath"/>

...

    <!-- compile the loader, used to change classpath especially for
           the CLI and Jetty -->
    <mkdir dir="${tools.loader.dest}"/>
    <javac srcdir="${tools.loader.src}"
           destdir="${tools.loader.dest}"
           debug="off"
           optimize="on"
```

```
deprecation="on"
target="1.5"    <=== changed from 1.3 in Cocoon 2.1.6
nowarn="on"
compiler="${compiler}"/>
```

Cocoon can be configured, built and tested in a few minutes. In some circumstances, you will make changes to the build configuration before you do this. In my case, I decided to disable all of the unstable optional 'blocks' in Cocoon and a few of the stable ones I didn't want to bother compiling. I did this by 'uncommenting' the definitions of variables in the local copy of the blocks.properties file.

Example 3. Installation and testing of Cocoon 2.1.6 from a download file.

```
expand the downloaded material

[john@localhost john]$ tar zxf cocoon-2.1.6-src.tar.gz
[john@localhost cocoon-2.1.6]$ cd cocoon-2.1.6

review optional components:

[john@localhost cocoon-2.1.6]$ cp blocks.properties local.blocks.properties
[john@localhost cocoon-2.1.6]$ vi local.blocks.properties

correct a Java version problem described above

[john@localhost cocoon-2.1.6]$ vi tools/targets/init-build.xml

run the build scripts

[john@localhost cocoon-2.1.6]$ ./build.sh

I usually do the following in a CTRL-ALT-F1 tty

[john@localhost cocoon-2.1.6]$ ./cocoon.sh servlet
```

You should consider increasing the maximum heap size for Cocoon's JVM if you will be generating PDF documents that are more than a couple of hundred pages in length. The default value set in `cocoon.sh` is 512 megabytes. This will be sufficient for most documents. Please note that the performance of Fop is somewhat problematic as far as very large or complex documents are concerned.

See Also

DocBook XML/SGML Publishing using OpenJade by Saqib Ali (on the web).

8. Installing the DocBook Stylesheets for use by applications such as Cocoon.

Problem

You need to install the DocBook XSL stylesheets on your system so they can be used to generate high-quality material suitable for publishing,

Solution

Download and install the stylesheets where they can be used by applications programs on your system. You may refer to the excellent documentation for the style sheets if you wish to customize the output.

Discussion

Obtain the latest files from the SourceForge web site: `http://docbook.sourceforge.net/`. For example, the release at time of writing is 1.68.1 and the file I downloaded was `docbook-xsl-1.68.1.tar.bz2`.

As a suitably privileged user (e.g. *root*) create directories `/usr/local/dbtools/docbook-xsl` and expand the downloaded archive file.

Example 4. Installing the docbook-xsl style sheets.

```
[root@localhost root]$ mkdir -p /usr/local/dbtools
[root@localhost root]$ cp docbook-xsl-1.68.1.tar.bz2 /usr/local/dbtools
[root@localhost root]$ cd /usr/local/dbtools
[root@localhost dbtools]$ bunzip2 docbook-xsl-1.68.1.tar.bz2
[root@localhost dbtools]$ tar xf docbook-xsl-1.68.1.tar
[root@localhost dbtools]$ mv docbook-xsl-1.68.1 docbook-xsl
```

The expanded directory tree includes style sheets which will produce HTML, XSL Formatting Objects, XHTML and several other formats which are of more specialized nature (Eclipse Help, Help, HTML Help, Java Help and Unix Man pages).

See Also

DocBook XSL: The Definitive Guide by Bob Stayton (http://www.sagehill.net/).

9. Testing Vex in the Eclipse IDE

Problem

You have a platform that is not supported with an installable version of Vex or you need to use a feature that is only available in the cutting edge of Vex Development.

Solution

Install the Eclipse Integrated Development Environment from `Eclipse.org`, download the latest Vex code from the CVS Repository and run it under the Plug-in Development Environment.

Discussion

The Eclipse IDE can be obtained from `http://www.eclipse.org`, one of it's mirror sites or a Bit-Torrent server. See the web site for instructions and a list of current web, ftp or torrent sites and installation instructions. You must download and install the latest stable production release of the Eclipse platform. At time of writing, this is Eclipse 3.0.1.

The current source for Vex may be obtained from the CVS service on the SourceForge web site. From

within Eclipse you may access the CVS Repository Exploring Perspective using: Window->Open Perspective->CVS Repository Exploring (**Alt-WO**) . From the CVS Repositories View, right click to bring up the context menu, select New Repository Location and fill in the form as illustrated below. The anonymous password can be omitted or entered as 'anonymous'.

Figure 2. CVS Repository Exploring properties for the Vex connection.

When the Repository Location appears in the view, expand it and expand the HEAD branch that appears below it. This shows a number of Vex source directories. We are interested in a subset of these, which are listed below. Select the desired directories, (**Ctrl-Left-click**) and then use: Check Out (**Right-click**).

Vex Directories to be checked out of CVS.

Each the following produces an Eclipse plug-in that contributes functionality to the Vex project. The directories listed here represent the minimal supported configuration for Vex, even though it is theoretically possible to run Vex with only `vex-toolkit` and `vex-editor`. If you need to run such an absolute minimum configuration, you should be able to support it yourself.

`vex-docbook`	DocBook functionality.
`vex-editor`	Graphical User Interface components.
`vex-samples`	working samples of DocBook XML and XHTML documents.
`vex-toolkit`	core functionality: CSS support, DTD parsing, other basic infrastructure.
`vex-xhtml`	XHTML functionality.

Figure 3. CVS Repository containing Vex source components required for testing in the Eclipse IDE.

The Eclipse system will check the selected projects out of CVS, download them and build the Vex system for you. This process should produce a useable Vex plug-in for you. The Eclipse system includes a sophisticated testing facility known as the 'Run-time Workbench'. You may invoke this in two modes. Debug mode allows you to perform detailed debugging of the Java classes implementing both Eclipse and Vex. Run mode allows you to simply execute the Vex plug-in as if it were installed in Eclipse. Debug mode is documented elsewhere in the literature.

To execute Vex from Eclipse use: Run->Run As->Run-time Workbench. This will create a temporary Run Configuration and start up a fresh copy of Eclipse which includes your copy of the Vex Plug-in. This should behave identically to the packged version of Vex that users download from SourceForge. It may differ from the pre-packaged version if the code base has been updated by the development team.

See Also

See Recipe 1.1 "Getting Eclipse", Recipe 1.2 "Installing and Running Eclipse", Chapter 6: "Using Eclipse in Teams" and Recipe 12.4 "Testing Plug-ins with the Run-time Workbench" in *The Eclipse Cookbook* (O'Reilly) or the Eclipse.org web site: `http://eclipse.org`.

10. Publishing DocBook from the Command Line

Problem

You have to generate publishing-grade material from DocBook input in an environment which does not provide access to Eclipse or Tomcat and Cocoon.

Solution

Use the command line to invoke Apache Fop with input in DocBook XML, transformations in DocBook XSL and output to any one of the formats produced by Apache Fop. Note that limited customization of the DocBook XSL stylesheets is possible through the -param lines in the following example.

Discussion

If your system does not have Fop (**locate fop.jar**), you must download it from `http://xml.apache.org/fop` and install it **(su; mv fop-0.20.5-bin.tar.gz /opt; tar zxf fop-0.20.5-bin.tar.gz)**.

Use a command line or shell script containing the following:

Example 5. A shell script used to invoke Apache Fop from the command line.

```
#!/bin/sh

FILE=VexCookbook

FOP_HOME=/opt/fop-0.20.5
DBXSL_HOME=/usr/local/dbtools/docbook-xsl
VEXWS_HOME=${CATALINA_HOME}/webapps/cocoon/docbook

java -server -Xmx400m  -classpath \
   ${FOP_HOME}/lib/batik.jar:${FOP_HOME}/lib/xalan-2.4.1.jar \
   org.apache.xalan.xslt.Process \
   -param section.autolabel 1 \
   -param section.autolabel.max.depth 1 \
   -param generate.section.toc.level 1 \
   -param toc.section.depth 1 \
   -IN ${VEXWS_HOME}/${FILE}.xml \
   -XSL ${DBXSL_HOME}/fo/docbook.xsl \
   -OUT ${HOME}/${FILE}.fo

java -server -Xmx400m -classpath \
   ${FOP_HOME}/build/fop.jar:${FOP_HOME}/lib/batik.jar:${FOP_HOME}/lib/avalon-fram
   org.apache.fop.apps.Fop -fo ${HOME}/${FILE}.fo -pdf \
   ${HOME}/${FILE}.pdf

/bin/rm ${HOME}/${FILE}.fo

acroread ${HOME}/${FILE}.pdf

exit
```

See Also

The Apache Fop project: `http://xml.apache.org/fop/`.

11. Installing Vex Into Eclipse 3.0.1

Problem

You want to maintain program documentation in the development environment.

Solution

Install Vex into your Eclipse 3.0.1 IDE.

Discussion

Vex can be installed as a set of plugins into an existing Eclipse 3.0 environment by connecting to the Vex update site, as follows:

1. From Eclipse, select Help->Software Updates->Find and Install....
2. Select Search for new features to install and click Next.
3. Click New Remote Site.
4. Enter "Vex Update Site" for Name and "http://vex.sourceforge.net/update" for URI and click Ok.
5. Select Vex Update Site in the resulting list and click Next.
6. Select Vex XML Editor and click Next.
7. Read the license agreement. If you agree, select I accept the terms in the license agreement and click Next.
8. Click Finish.
9. You will be warned that the feature is not signed. Click Install to proceed with the installation.
10. When the installation is complete you will be prompted to restart Eclipse. I have successfully installed Vex this way without restarting, but you might want to restart if you want to be really safe.

See Also

The Vex home page includes up-to-date instructions for this and related tasks.

12. Upgrading Vex to a new Release

Problem

You have been using Vex for a while, a new version has been released and you want to use it.

Solution

If you have all ready installed Vex, you can upgrade to the latest version using the Eclipse Update facility.

Discussion

Use the Eclipse Update facility as follows:

1. Select Help->Software Updates->Find and Install...
2. Select Search for new features to install and select Next.
   Note

   The Vex upgrade may or may not be available under "Search for updates of the currently installed features" depending on the version of Vex you have installed, the version to which you are upgrading and your settings under Windows->Preferences->Install/Update.
3. Select Vex Update Site and Next.
4. Select Vex XML Editor and Next.
5. Read the license agreement. If you agree, select I accept the terms in the license agreements and Next.
6. Select Finish.
7. You will be warned that the feature is not signed. Select Install to proceed with the installation.
8. When the installation is complete you will be prompted to restart Eclipse.

See Also

The Vex home page includes up-to-date instructions for this and related tasks.

13. Extending Vex: Adding New Document Types and Styles to Vex

Problem

You have a Document Type that is not supported by Vex and you would like to edit files which use it.

Solution

You will need to add Document Type Definition (`dtd`) and Cascading Style Sheet (`css`) files to your Vex installation.

Discussion

Vex provides a Vex Plug-in Project nature to accomodate users who wish to extend Vex support for new document types. As an example, consider the SourceForge project: xmlresume. This project produced a

DTD which defines a very general format for the creation of resumes by job seekers. The project also produced some XSLT style sheets for the generation of resumes in print, word processing and hypertext formats. Some CSS style sheets were also produced to assist with the display of HTML formatted resumes on web browsers. The only new code required to support XML Resume in Vex is a `CSS` file which is available from the author.

Vex may be extended to support the XML Resume format as follows:

1. Create a Vex Plug-in project by invoking the Vex New Project Wizard by selecting one of the following: (**CTRL-N**); (**SHIFT-ALT-N-O**) or File->New->Other. This will open the New Wizard selection dialog. If necessary, expand the Vex entry and select Vex Plug-in Project.

2. The first page of the New Plug-in Project wizard asks you to name the new project and identify a directory which contains the projects resources. Every project requires a unique name in the workspace and the project contents are built up in the project directory within the workspace by default although they may exist elsewhere in the file system.

   In this case, you would simply create a new project directory in the Vex workspace.

3. Import the XML Resume DTD files from the installation directory of the XML Resume system. Use the command: File->Import to enter the Import Wizard and select the appropriate source for your files.

4. Select the imported DTD file in the Resource Navigator, right-click and select Properties. In the Property Sheet that appears, select the "Vex Document Type" item and fill in the appropriate values:

Name	XML Resume 1.5.1
Public Id	-//Sean Kelly//DTD XML Resume 1.5.1//EN
System Id	http://xmlresume.sourceforge.net/dtd/resume.dtd
Root Elements	A generated table of checkboxes containing potential root elements from the compiled DTD. The user is required to select one or more of these items for presentation to the user by the New Document Wizard as allowable root elements. The tag names: resume and resumes are valid selections for our purposes.

   Select 'Apply' or 'Ok' to save the values you wish to use.

5. Import the CSS style sheet file(s) from the file system.

6. Select the imported CSS file in the Resource Navigator, right-click and select Properties. In the Property Sheet that appears, select the "Vex Style" item and fill in the appropriate values:

Name	XML Resume 1.5.1
Document Types	A generated table of checkboxes containing document types registered with Vex. The user is required to select one (or more?) of these items so that the editor will be able to apply styling to documents when they are opened in the editor.

   Select 'Apply' or 'Ok' to save the values you wish to use.

See Also

XML Resume project at SourceForge.net.

14. Contributing to Vex: Changing a CSS file

Problem

You have discovered an error in one of the style sheets that came with Vex.

Solution

Set up a test environment and test your changes.

Discussion

The latest release of Vex includes a new configuration system that is supposed to simplify this process. For one reason or another, I was unable to modify the DocBook style sheet in my environment the way I thought I was supposed to. I did find another way to get this working and I thought it worth sharing as it demonstrates the new and old configuration systems.

I found myself dissatisfied with the DocBook style sheets that were supplied with Vex and decided to add some improvements. In order to outwit the Vex configuration system, I decided to remove the docbook-plain.css file from the configuration and roll my own using the new configuration system.

My first step was to remove docbook-plain.css from my development environment. I did this by opening the file: `vex-docbook/plugin.xml` and deleting the linkages between the DocBook DTD's and the style sheet. These linkages are in the form of `<doctypeRef>` tags contained with the style file's `<style>` tag.

My second step was to create a Vex Plug-in project and import a copy of the `docbook-plain.css` file into it. In my case, I decided to use the file name `docbook-test.css`. I made new links between the DocBook DTD's registered with Vex and this new style file by opening up the propertiies dialog for the new style file. Selecting the 'Vex Styles' dialog brought up the properties sheet for the new style. This was duly given a name and associated with the registered DocBook Document Types by selecting the appropriate check boxes in the property sheet.

To use this style sheet, I imported a DocBook document (the one you are reading now) and opened the style sheet file itself. I rearranged teh Eclipse Desktop so that the two editors were tiled side-by-side and began to make changes to the CSS file. Every time I saved a set of changes to the style sheet, they were applied (instantaneously) to the DocBook document before me.

I was so impressed by this that I decided to make sure that it stopped working for me. The original docbook-plain.css file is really quite simple and was probably developed while John's focus was on coding and testing the layout engine. I decided to start by breaking the file up into specialized subfiles and to add the usual Copyright statements to the code. This was possible because the CSS code that is used in Vex was written with support for the CSS @import statement. As I discovered specialized groups of statements int the main file, I refactored them into new files and added references to the new file to the original.

This had the effect of diminishing the instant effect of style changes on my visual editing environment. The imported subfiles are not yet supported as Eclipse Resources, so changes are not reported to the Workbench and updates are not reflected to the user. I find that I must make a trivial change to the main file (`docbook-test.css`) and save it to disk to trigger visual changes.

See Also

blah

15. Contributing to Vex: Changing a Java function

Problem

You have discovered a programming error in the implementation of Vex.

Solution

Set up a test environment and test your changes.

Discussion

You should download a copy of the latest Vex development software and test your changes thoroughly. Once you believe you have solved the problem, or reached a point at which the develiopment team will be interested in your contribution, you can generate a diff file (using Eclipse) and post it to the vex-developers mailing list on the sourceforge site.

Remember that your change may not meet everyone else's requirements, expectations and prejudices. You should be prepared to support your proposal in a critical review conducted through e-mail discussion.

See Also

The Vex home page includes up-to-date instructions for this and related tasks.