SINGLE-SOURCE INTERACTIVE AND PRINTED CONTENT PUBLISHING USING THE DOCBOOK XML STANDARD

Dr. Derek Molloy

RINCE

Dublin City University, Ireland.

Introduction – Presentation of Course Content

- On-Line Format (HTML)
  - Allows interactive components
  - Allows personalized content
    - Through server-side applications (Servlet, EJB etc.)
  - Difficult to copyright protect content
  - Can be difficult to format for correct printing
- Print-Ready Format (PDF, Word, PowerPoint)
  - Can have large file size (Word/PowerPoint)
  - No interactive components
  - PDF content can be copyright protected

Introduction – Context of this implementation

- Module in object-oriented programming at DCU
  - With 80 full-time post-graduate students
  - With 40 part-time/on-line post-graduate students
- Full Course content must be available on-line
- Need content that is suitable:
  - for high-quality printing
  - for on-line viewing, to allow interactive components
  - for electronic lecture presentation
- Need a single document source
  - 3 versions too difficult to maintain as content is dynamic

Introduction – Introduction to DocBook

- DocBook XML Standard (OASIS Project)
  - Collection of standards & tools for publishing
  - Useful for highly structured content
  - Single source XML file can be transformed:
    - HTML – suitable for on-line presentation
    - PDF – suitable for printing
    - Rich Text – for possible later editing
  - Style is maintained independent of content

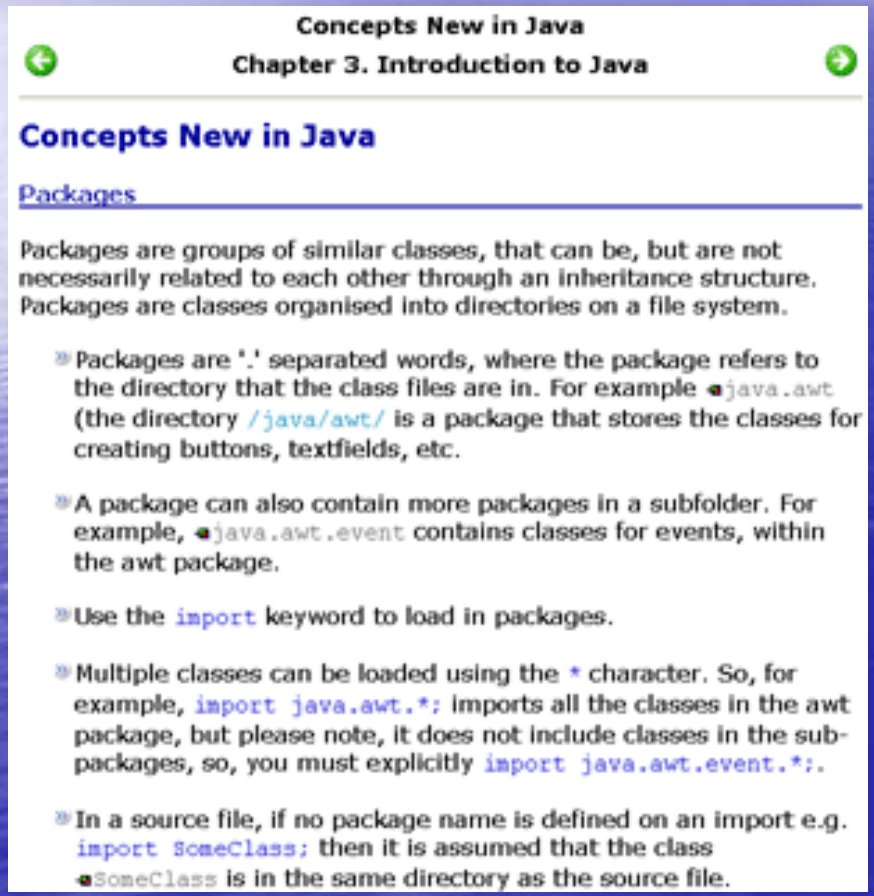
Implementation – DocBook Example Source

- Source Example:
  - Tags define content (e.g. <filename> <classname>)
  - and document structure (e.g. <para><sect2>)

```
<sect1><title>Concepts New in Java</title>
<sect2><title>Packages</title>
<para>
Packages are groups of similar classes, that can be, but are not necessarily related to each other
through an inheritance structure. Packages are classes organised into directories on a file system.
<itemizedlist>
    <listitem><para>Packages are '.' separated words, where the package refers to the directory
      that the class files are in. For example <classname>java.awt</classname> (the directory
      <filename>/java/awt/</filename> is a package that stores the classes for creating buttons,
      textfields, etc.</para></listitem>
    <listitem><para>A package can also contain more packages in a subfolder. For example,
      <classname>java.awt.event</classname> contains classes for events, within the awt package.
      </para></listitem>
    <listitem><para>Use the <literal>import</literal> keyword to load in packages.</para></listitem>
    <listitem><para>Multiple classes can be loaded using the <literal>*</literal> character. So,
      for example,<literal>import java.awt.*;</literal> imports all the classes in the awt package,
      but please note, it does not include classes in the sub-packages, so, you must explicitly
      <literal>import java.awt.event.*;</literal>.</para></listitem>
    <listitem><para>In a source file, if no package name is defined on an import e.g.
      <literal>import SomeClass;</literal> then it is assumed that the class
      <classname>SomeClass</classname> is in the same directory as the source file.</para></listitem>
</itemizedlist>
</para>
</sect2>
```

Implementation – DocBook Example Output

- Segment of HTML and PDF output



Concepts New in Java

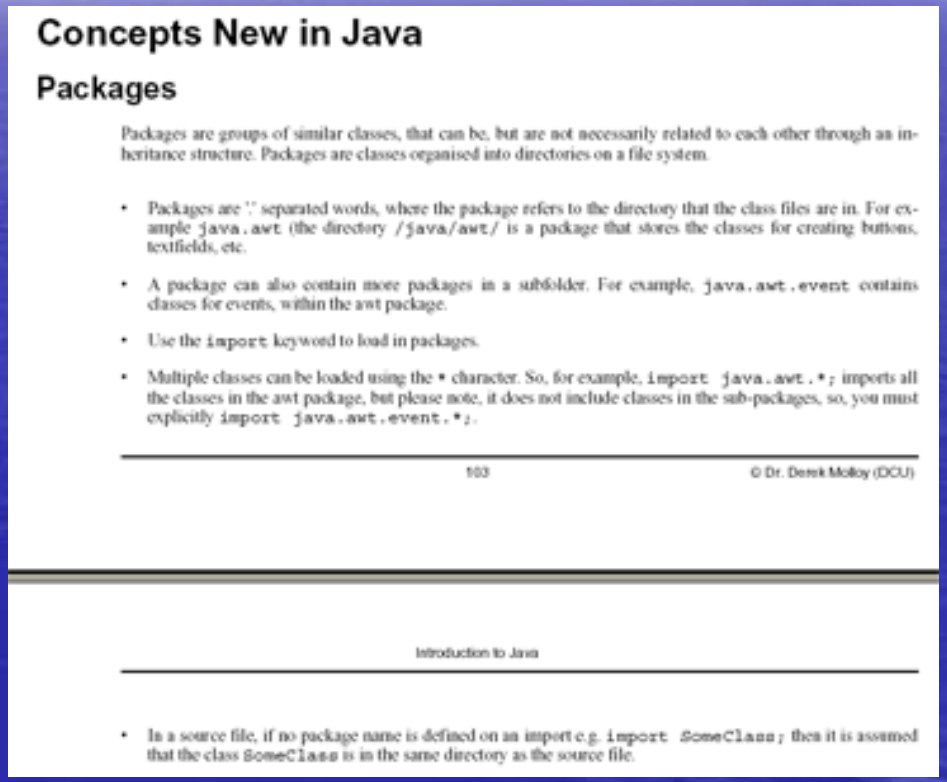
Chapter 3. Introduction to Java

Concepts New in Java

Packages

Packages are groups of similar classes, that can be, but are not necessarily related to each other through an inheritance structure. Packages are classes organised into directories on a file system.

- Packages are '.' separated words, where the package refers to the directory that the class files are in. For example `java.awt` (the directory `/java/awt/` is a package that stores the classes for creating buttons, textfields, etc.
- A package can also contain more packages in a subfolder. For example, `java.awt.event` contains classes for events, within the awt package.
- Use the `import` keyword to load in packages.
- Multiple classes can be loaded using the `*` character. So, for example, `import java.awt.*;` imports all the classes in the awt package, but please note, it does not include classes in the sub-packages, so, you must explicitly `import java.awt.event.*;`.
- In a source file, if no package name is defined on an import e.g. `import SomeClass;` then it is assumed that the class `SomeClass` is in the same directory as the source file.



Concepts New in Java

Packages

Packages are groups of similar classes, that can be, but are not necessarily related to each other through an inheritance structure. Packages are classes organised into directories on a file system.

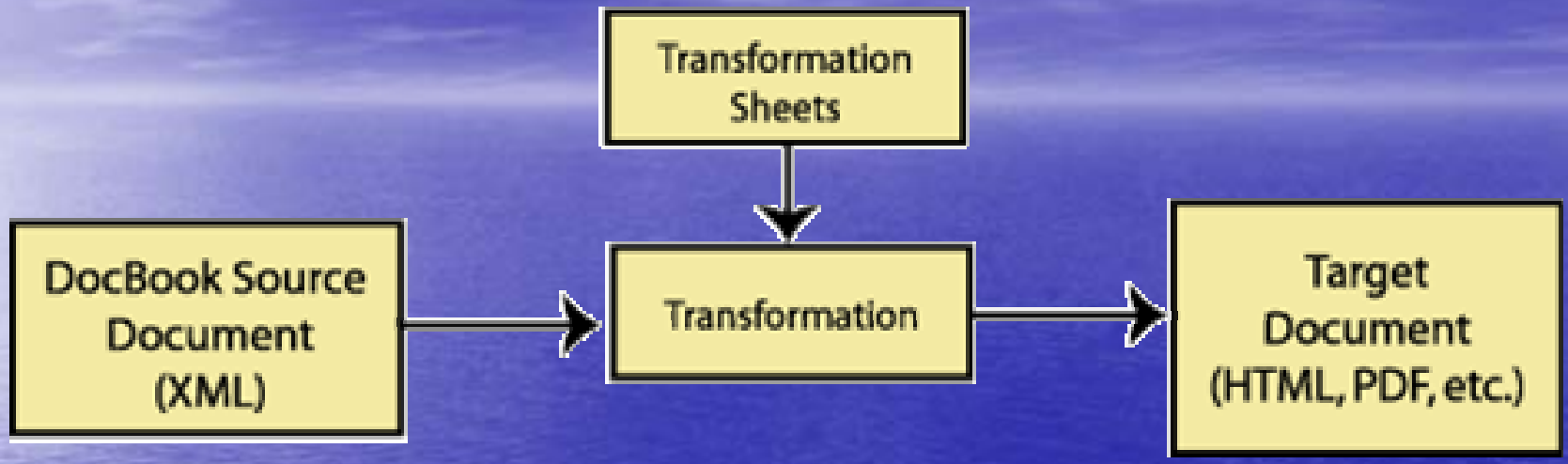
- Packages are '.' separated words, where the package refers to the directory that the class files are in. For example `java.awt` (the directory `/java/awt/` is a package that stores the classes for creating buttons, textfields, etc.
- A package can also contain more packages in a subfolder. For example, `java.awt.event` contains classes for events, within the awt package.
- Use the `import` keyword to load in packages.
- Multiple classes can be loaded using the `*` character. So, for example, `import java.awt.*;` imports all the classes in the awt package, but please note, it does not include classes in the sub-packages, so, you must explicitly `import java.awt.event.*;`.

103 © Dr. Derek Molloy (DCU)

Introduction to Java

- In a source file, if no package name is defined on an import e.g. `import SomeClass;` then it is assumed that the class `SomeClass` is in the same directory as the source file.

Implementation – DocBook Output Generation



- Transformation
  - Validation against DTD for document structure
  - Use of Xalan and FO for PDF conversion
  - Use of Xalan and XSL for HTML conversion
    - HTML is then combined with CSS for Presentation

Implementation – Advanced Conversion Features

- Profiling
  - Tag paragraphs with user-level details
    - E.g. <para userlevel="advanced"> text </para>
  - Allows generation of several parallel versions of document
    - Generate user-level “beginner” document for first pass
    - Generate user-level “advanced” document for second pass

Implementation – Advanced Conversion Features

- Interactive Content
  - Can add interactive content to HTML version
    - On-line self assessment questions (Servlet checked)
    - Interactive applet/flash movie
  - Provide suitable replacement for PDF version
    - Printed version of questions
    - Image instead of applet
  - Use rules in XML
    - E.g. <?dbhtml command1 ?><?dbfo command2 ?> where command1 is executed for html and command2 is executed for PDF version.

Implementation – Advanced Conversion Features

- Accessible Content (for disabled users)
  - Since content independent of Style
    - Can use high contrast conversion/CSS
    - Alternative image information presented
    - HTML contains content clues
    - Can provide XML source to VoxML in future?

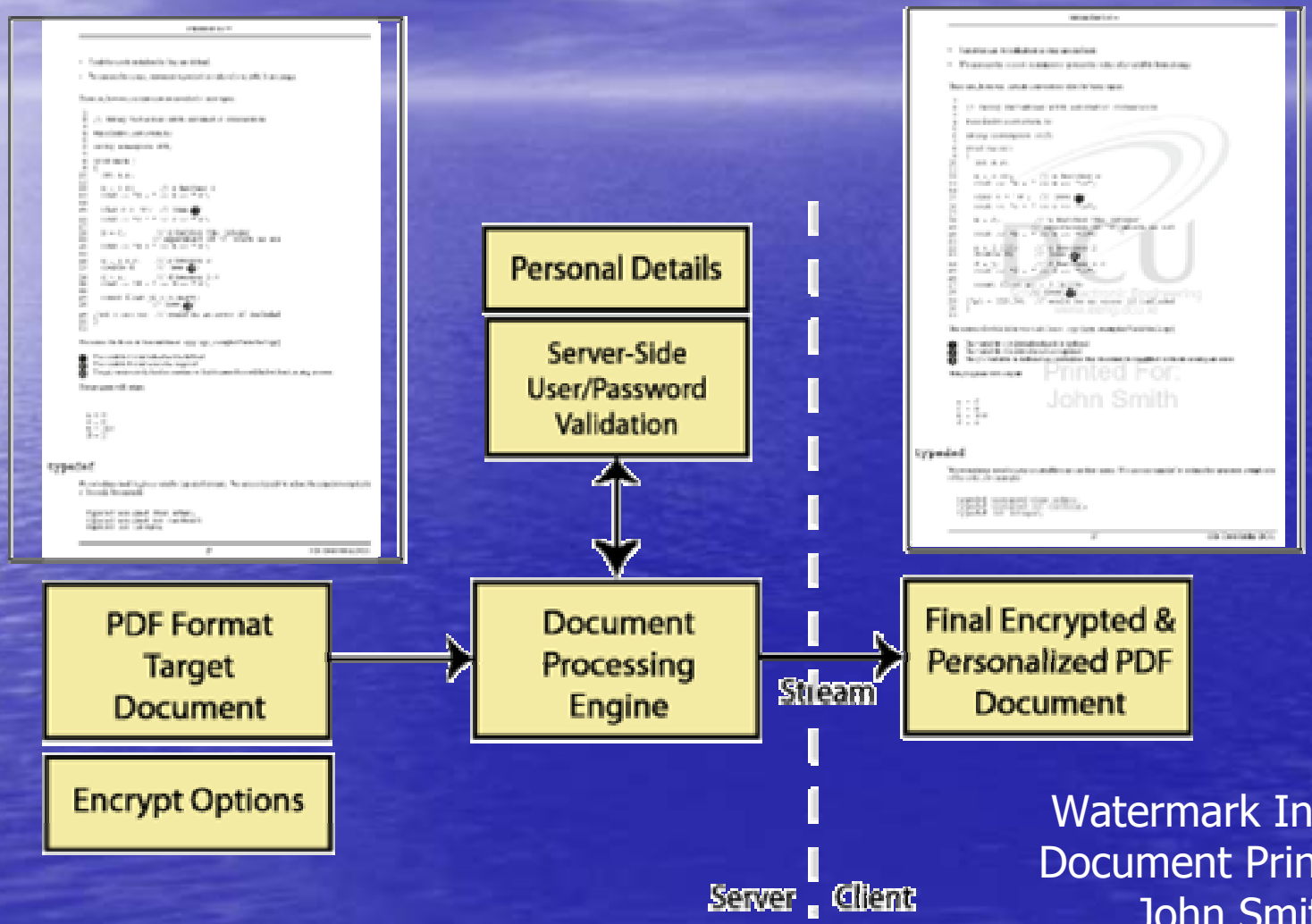
Implementation – Advanced Conversion Features

- Customization
  - Can modify and add own tags to suit content
  - E.g. <classname> tag can be modified to link to on-line Java API documentation
  - Modify using XSL conversion rules.
  - Can build in advanced conversion functionality

Implementation – Content Protection

- In this implementation
  - Additional Document Processing Engine Built
    - Using Java Servlets and iText (Java-PDF Library)
  - Generation of PDF takes 3-4mins
  - DPE takes generated PDF and customizes for user
    - Adds user information as Watermark
    - Creates personalized front cover
    - Encrypts document to maintain copyright (128bit)
    - Can prevent editing/cut-and-paste etc.

Implementation – Content Protection



Implementation – Content Protection

- PDF Version Protected
  - With personalized user details in watermark and on front cover
  - Encrypted with User password to open
    - Must consciously pass on information
  - Document properties can be altered
- HTML Version Protected
  - Difficult to download as Chunked
  - User personal details on each page
  - All links are static, not relative.

Conclusions – Problems with DocBook

- Problems with DocBook for On-Line Content
  - Difficult to set-up process, choice of tools
  - Suitable for Engineering/Computing content
    - May need many additions for other subject matter
  - No high-quality WYSIWYG Tools freely available
  - Set-Up time is significant

Conclusions – Final Conclusions

- Once in place changes to content easy to apply, Content easy to maintain
- Deployment of content can be scripted
- Once user-management is in place, access and control self-maintaining
- Further advanced features can be developed and easily integrated

Thank you for your patience