igpp.docgen

Introduction with Examples

Todd King

Overview

The igpp-docgen tool is

- A document generator based on Apache Velocity.
- Can read PDS3 labels, text files containing keyword=value pairs, spreadsheets in CSV or TAB format and metadata in CDF files.
- The output format can be text files, including well-formed PDS3 or XML.

Available at: http://release.igpp.ucla.edu/igpp/docgen/

Apache Velocity

The Velocity Template Language (VTL) is embedded mark-up which can be included in text files and interpreted by the Velocity processor.

It allows the content of documents (like XML, web pages, etc.) to be parameterized.

Full documentation at:

http://velocity.apache.org/engine/devel/user-guide.html

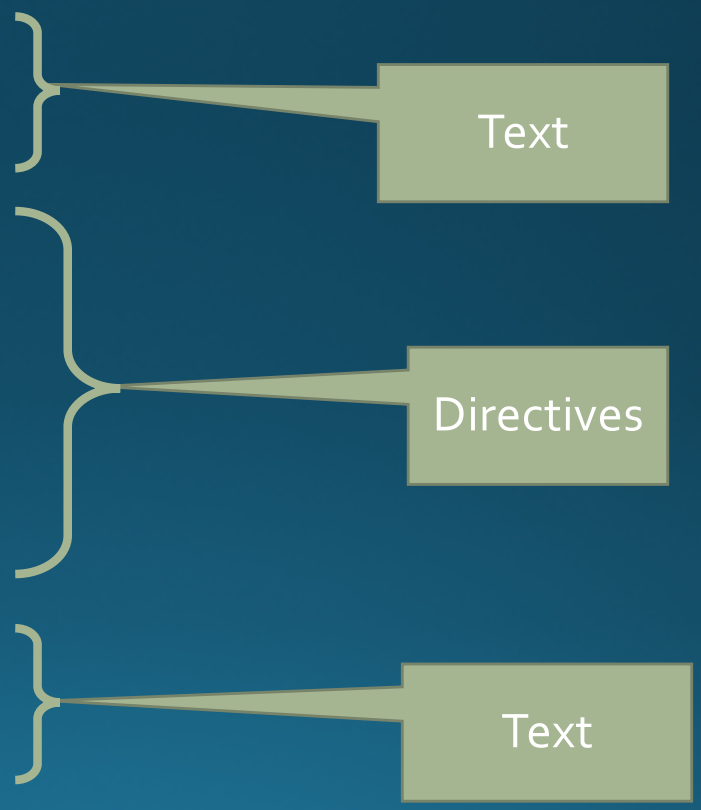
Velocity Template

- A velocity template is text plus velocity directives.

```
<html>
<body>
<p>My favorite classical
composer is

#if( $desire == 10 )
   Wolfgang Amadeus Mozart
#elseif( $desire > 5 )
   Ludwig van Beethoven
#else
   Johann Sebastian Bach
#end

</p>
</body>
</html>
```



Velocity Comments

Templates may have comments that are removed when processed.

Single line

```
## This is a comment
```

Multi-line

```
#*
This is a mult-line
comment.
*#
```

Javadoc style comments

```
#**
@author
@version 5
**#
```

Velocity Variables

- Variables can be set in the template or by the template processor (i.e. igpp.docgen).

  Example

```
#set($name = “something”)
#set( $list = ["name", "address"] )
```

- Variables are objects and can have properties and methods.
- Docgen sets several helper variables and sets variables based on inputs (more on this later)

Velocity Variable References

A variable consists of leading "**$**" followed a reference name of the variable. The name may be enclosed in curly braces {} for clarity.

Examples:

Simple variables

```
$name
${name}
```

Properties and Methods

```
$customer.Address
$purchase.getTotal()
```

For a literal $ escape with a \ (example \$name outputs $name)

Velocity Loops

Velocity can iterate over variables with multiple values.

```
<table>
#foreach( $customer in $customerList )
   <tr><td>$foreach.count</td><td>$customer.Name</td></tr>
#end
</table>
```

Loop properties

- $foreach.count : Index of current item
- $foreach.hasNext : Boolean if more items to process
- #break : exit a loop

Velocity Flow Control

Conditional processing is controlled with

```
#if( $desire == 10 )
   Wolfgang Amadeus Mozart
#elseif( $desire > 5 )
   Ludwig van Beethoven
#else
   Johann Sebastian Bach
#end
```

Operators

<, >, ==, !=, >=, <=, !, &&, ||

Other directives

- Don't parse me

  ```
  #[[ do not process ]]#
  ```

- Include

  ```
  #include(filename)
  ```

- Parse (include Velocity snippets)

  ```
  #parse(filename)
  ```

Math

- Simple math operations are supported

```
#set( $foo = $bar + 3 )
#set( $foo = $bar - 4 )
#set( $foo = $bar * 6 )
#set( $foo = $bar / 2 )
#set( $foo = $bar % 5 )
```

igpp.docgen features

igpp.docgen will process a Velocity Template and defines several utility contexts. Each context is a Java class library. Some are standard Java libraries, others are IGPP developed:

$Calc: Advanced math operations (igpp.util.Calc)

$Date: Date conversion and parsing (igpp.util.Date)

$File: Information about files (igpp.util.File)

$Text: Additional string operations (igpp.util.Text)

$Integer: Parsing and manipulation of integers (java.lang.Integer)

$Double: Parsing and manipulation of doubles (java.lang.Double)

$Long: Parsing and manipulation of big integers (java.lang.Long)

$String: Parsing and manipulation of strings (java.lang.String)

File Based Context

- igpp.docgen can generate a context based on the contents of a file. These are also implemented using Java class libraries. Supported context are:

  pds3: Scan a PDS3 label file and generates a list of values. Structure matches that of the label.

  list : Parse a text file with one keyword=value per line and generate a list of values.

  csv: Read a file containing a delimited table and for each row generate and entry.

  cdf: Parse a CDF file and extra all metadata.

igpp.docgen variables from files

- Igpp.docgen can create a context (group of variables) based on the contents of a file. The type of context is set by the filename extension.

  .txt : keyword=value. The value may be multivalued using {} and comma separated values. # indicates comment lines and are ignored.

  .csv, .tab: Tables. Field names are taken from first line. Values stored in an array named "record". # indicates comment lines and are ignored.

  .lbl: PDS3 label. Elements match label structure.

  .cdf: CDF file.

  .vm: Velocity template.

igpp.docgen command-line variables and context

Variables may be set on the command line with the syntax:

```
name=value
```

Context is "options"

Context name for parsed files is set with the syntax

```
context:filename.ext
-or-
context:format:filename.any
```

Examples:

```
label:product.lbl
label:pds3:product.lbl
```

igpp.docgen command line options

-f,--format <arg> Format. Format output with a given style. Allowed values are PDS3, XML, HTML and Plain. Default: XML

-h,--help Dispay this text

-i,--include <arg> Include Path. Path to look for files referenced with an INCLUDE or STRUCTURE pointer.

-o,--output <arg> Output. Output generated document to {file}. Default: System.out.

-s,--separator <arg> Separator. Pattern that separates values in tabular files. Default: is a tab

-t,--template <arg> Template. The template folder to search for templates file.

-v,--verbose Verbose. Show status at each step.

Give it a try.

Remember: verbose (-v) is your friend.

sample-label.vm

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="http://pds.nasa.gov/pds4/pds/v1/PDS4_PDS_${settings.schema_ver}.sch"
  schematypens="http://purl.oclc.org/dsdl/schematron"?>

<Product_Observational
  xmlns="http://pds.nasa.gov/pds4/pds/v1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
>
  <Identification_Area>
    <logical_identifier>urn:nasa:pds:$settings.bundle:$settings.collection:$options.lidprod</logical_identifier>
    <version_id>$settings.version</version_id>
    <title>$jove.getAttributeValue("Project") $jove.getAttributeValue("Instrument_type") located at $settings.location</title>
    <information_model_version>$settings.im_ver</information_model_version>
    <product_class>Product_Observational</product_class>
    <Citation_Information>
      <author_list>$jove.getAttributeValue("PI_name")</author_list>
      <publication_year>$File.getModificationDate("$jove.pathName").substring(0, 4)</publication_year>
      <description>
        $jove.getAttributeValue("Project") $jove.getAttributeValue("Instrument_type") located at $settings.location
      </description>
    </Citation_Information>
    <Modification_History>
      <Modification_Detail>
        <modification_date>$File.getModificationDate("$jove.pathName").substring(0,10)</modification_date>
        <version_id>$settings.version</version_id>
        <description>
          $settings.moddesc
        </description>
      </Modification_Detail>
    </Modification_History>
  </Identification_Area>

</Product_Observational>
```

settings.txt

```
# Settings for creating a RadioJove product label
bundle=RadioJove
collection=Florida
version=1.0
im_ver=1.5
schema_ver=1500
location=Florida State University
moddesc=Initial release
```

igpp.docgen

```
$ ../tools/igpp/docgen/bin/docgen -v lidprod=edr_sp2_300 settings:settings.txt \
        jove:radiojove_edr_sp2_300_201603201523_201603201537_v09.cdf simple-label.vm
```

radiojove_edr...v09.cdf

output.xml

```
<?xml-model href="http://pds.nasa.gov/pds4/pds/v1/PDS4_PDS_1500.sch"
  schematypens="http://purl.oclc.org/dsdl/schematron"?>
<Product_Observational
                    xmlns="http://pds.nasa.gov/pds4/pds/v1"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Identification_Area>
        <logical_identifier>urn:nasa:pds:RadioJove:Florida:edr_sp2_300</logical_identifier>
        <version_id>1.0</version_id>
        <title>RadioJOVE Spectrogram Receiver located at Florida State University</title>
        <information_model_version>1.5</information_model_version>
        <product_class>Product_Observational</product_class>
        <Citation_Information>
            <author_list>RadioJOVE Project</author_list>
            <publication_year>2016</publication_year>
            <description>
        RadioJOVE Spectrogram Receiver located at Florida State University
      </description>
        </Citation_Information>
        <Modification_History>
            <Modification_Detail>
                <modification_date>2016-04-26</modification_date>
                <version_id>1.0</version_id>
                <description>
          Initial release
        </description>
            </Modification_Detail>
        </Modification_History>
    </Identification_Area>

</Product_Observational>
```

Questions?