PODi

PPML 3.0

May 2010

Personalized Print Markup Language

Packaging PPML Datasets for Transport using ZIP files or Removable Media

PODi: the Digital Printing Initiative

1240 Jefferson Road, Rochester, New York 14623, USA

Tel: (585) 239-6014

Internet: **http://www.podi.org**

PODi the Digital Printing Initiative

Approval of a PODi standard requires acceptance by the members of PODi.

PODi is a not for profit industry consortium formed in 1996. Its charter is to foster the growth of the digital printing industry through market and standards development activities. PODi constantly monitors market and technology trends in the industry, and shares information through seminars, independent research, white papers, articles, and the web. PODi promotes interoperability through the PPML suite of open, XML based standards, test suites and certification.

PODi welcomes feedback on this specification, and offers the following services to support widespread adoption of the specification:

Specification Updates

> The PPML specification is distributed free of charge. Developers who are implementing the PPML standard are invited to subscribe to free PPML updates and the technical note service.

Developer Support web site

> Software and hardware developers interested in supporting PPML are invited to register for the PPML Developers discussion group.

To participate in the PPML initiative, send an email to ppmlinfo@podi.org.

Table of Contents

Table of Contents v
1 Scope 1
2 Normative references 1
3 Definitions 1
4 Rules for Files and Directories 2
4.1 Directory Structure 2
4.2 PPML File 2
4.3 Character Set 2
4.4 Case Sensitivity 3
4.5 Length Restrictions 3
5 Rules for URIs 4
5.1 Preserve Case 4
5.2 Use Escape Characters 4
Annex A (informative) Example 5
A.1 Directory listing of conforming ZIP file 6
Annex B (informative) Change History 7

Forward

PODi: the Digital Printing Initiative developed this packaging specification to enable interoperability in job submission among PPML Producers and Consumers.

This specification is a conformance statement defining a conformance subset of PPML. Readers are directed to the PPML Specification for the definition of the syntax and semantics of PPML.

The working group responsible for the current specification had the following membership:

PODi Senior Technologist: Dr. Paul Jones

PODi Director of Technology: James Mekis

Contributing working group members:

EFI: Boris Aronshtam, Reuven Ackner
Hewlett-Packard: Steve Hiebert
IBM: Hitesh Bhindi, Claudia Alimpich, Art Ford
Kodak Creo: Luci Wahrmann
Kodak NexPress: Tim Donahue
Kodak Versamark: Josh Howard, Pat McGrew
Konica Minolta: Darrell Hopp
Océ Printing Systems: Helmut Weiner
Pageflex: Peter Davis
Punch Graphix: Bart Wynants
Xerox: John Czudak

Send suggestions for improving this standard to PODi, 1240 Jefferson Road, Rochester, NY 14623, USA; e-mail: ppmlinfo@podi.org.

Introduction

MIME and other transport methods are extremely flexible, and provide a highly generalized interface between many kinds of systems. However, current practice often involves the simple case of creating an entire project on one machine and physically transporting it to another machine. For this case there is an easier method of transport. This document describes rules for reliably packaging a PPML dataset created on one machine for transport to another machine, where it can be unpacked so that all references will still function as expected.

In this workflow, all related resource files are typically within a single directory tree. This practice allows for a simple case for constructing the PPML references and packaging the project: the entire directory may be copied from removable media to a directory using compression software.

PKZIP is one such packing application. It is supported on many platforms (Windows, Macintosh, Unix/Linux) and is available as open source.

The rules described here for references and file naming allow for reliable transport of a PPML dataset from one machine to another, and between platforms.

PPML Consumers are not required to accept ZIP files. If a PPML Producer generates datasets and ZIP packages that conform to these rules, the receiving system can use any unzipping tools to unpack the package, and the references should work successfully.

Note: In the workflows where this method is intended to be used, unpacking ZIP files is a common practice. Nonetheless, Consumers that can directly read ZIP files will offer two competitive advantages: simpler workflow and savings of disk space. Since some datasets can be large, there can be a real advantage to building this functionality into the Consumer.

Note: In a closed, formal environment where the Producer and Consumer know each other's systems, they are free to use any conventions they want. However, datasets and packages that do not conform to these rules will not transport and unpack reliably on unknown receiving systems. For portability, or if the receiving system is unknown when the dataset is generated, the rules defined in this section should be obeyed.

Packaging PPML Datasets for Transport using ZIP files or Removable Media

1 Scope

This document specifies rules for naming the PPML file and its attachment files. It also prescribes the relative locations of such files. These rules implement the following requirements.

When the files are extracted in a single operation from the ZIP file on the target platform or copied from removable media to directory on a target platform:

- The PPML file can be located easily
- Each URI in the PPML file references the intended attachment file using the target platform's software for mapping a URI to a specific file.

Note that these rules describe the characteristics of the files on the *target* platform after they are extracted from a ZIP file or copied from a removable media. The files on the source platform are likely to have similar characteristics to simplify the zipping or copying operation, but they are not required to adhere to these rules.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

Network Working Group. Uniform Resource Identifiers (URI): Generic Syntax. August 1998. http://www.ietf.org/rfc/rfc2396.txt

Network Working Group. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. November 1996. http://www.ietf.org/rfc/rfc2045.txt

PKWARE, inc. ZIP File Format Specification Version 6.2.1. April 1, 2005. http://www.pkware.com/business_and_developers/developer/appnote/appnote.txt

Network Working Group/The Internet Society. Uniform Resource Identifiers (URI): Generic Syntax. 1998. http://www.ietf.org/rfc/rfc2396.txt

3 Definitions

For the purposes of this specification, the following definitions apply.

3.1 Conforming Package

All the files and directories in a ZIP file.

3.2 Conforming Package Extraction

All the files and directories on a Consumer's platform that come from a single Conforming Package, either by extracting them from a ZIP file, or by copying them from a removable media.

The rules are phrased in terms of the Conforming Package Extraction because it is easier to specify the files and directories on the Consumer's platform than to specify the representation of files and directories on a ZIP file. It is best to treat a ZIP file as a black box with add and extract APIs. The extract API produces the Conforming Package Extraction. A rule about a Conforming Package Extraction implies what must be in a Conforming Package in order for the ZIP extract operation or the removable-media copy operation to work properly.

4 Rules for Files and Directories

4.1 Directory Structure

A Conforming Package Extraction shall contain a single top level directory. All files and directories in the Conforming Package Extraction shall reside in that top-level directory, or in directories under it.

4.2 PPML File

A Conforming Package Extraction shall contain only one PPML file, whose name must conform to the rules in this Appendix, and whose suffix is "`.ppml`". The PPML file shall reside in the top-level directory.

4.3 Character Set

Each character of a file name or directory name within a Conforming Package Extraction must be a printable character from ISO 696 IRV (i.e. those in the range 32 to 126 inclusive) that is not one of the nine characters in the table below.

Note: These characters are excluded from PPML filenames because Windows does not allow them in filenames. Windows is the most restrictive. Mac OS9 excludes the colon ":" and Linux excludes only "/", both of which are excluded from Windows filenames.

34 ('"' double quote)	42 ('*' asterisk)	47 ('/' slash)
58 (':' colon)	60 ('<' less-than)	62 ('>' greater-than)
63 ('?' question mark)	92 ('\' backslash)	124 ('|' vertical bar)

The first character of a file name or directory name must not be '`.`' (dot).

Note: This is due to limitations of some Windows systems.

4.4 Case Sensitivity

Each directory within a Conforming Package Extraction on a case-sensitive platform (i.e. the top-level directory or any directories under it) shall not contain multiple files whose names are identical except for the case of one or more letters. For instance, a directory cannot contain both `test.eps` and `TEST.EPS`, or `test.eps` and `Test.eps`. Also, a directory cannot contain both `Foo` and `foo`, where each is either a directory or file.

Because the above rule indirectly applies to a Conforming Package, the above rule implies that on a case-insensitive platform a Conforming Package Extraction contains exactly the same files and directories that are in the Conforming Package, e.g.. there won't be both a `test.eps` and `Test.eps`.

Note: A Producer creates a Conforming Package without the knowledge about whether the Consumer is on a case-sensitive or case-insensitive platform.

See rule 0 in the section entitled "A.4 Rules for URIs". It gives the corresponding rule for case sensitivity of URIs that reference files whose names are governed by this rule.

4.5 Length Restrictions

The maximum length for each name of a file or directory in a Conforming Package Extraction shall be 31 characters, including extension. Example: the name of the PPML file in the package must be limited to 26 characters plus 5 characters for "`.ppml`".

Note: Filenames in Macintosh OS9 are limited to 31 characters in length.

The maximum length for each path name of a file in a Conforming Package Extraction (relative to the top-level directory of the Conforming Package Extraction) shall be 127 characters, including slashes, period and file suffix.

Note: Windows is the motivation for this constraint. On Windows the maximum length of an absolute path is 254 characters; every file created in a Windows directory must result in an absolute path no more than 254 characters long, or the file creation will fail. Arbitrarily dividing 254 in half allows:

127 characters for the complete pathname of the top-level directory of the Conforming Package Extension, e.g. "`c:/projects/projectFoo`".

127 characters for the path of each file in the Conforming Package Extension relative to its top-level directory, e.g. "`images/sunset.gif`" is the relative path of the file (limit 127 characters) and "`c:/projects/projectFoo/images/sunset.gif`" is the complete pathname of the file (limit 254 characters).

These limits are considered reasonable for expected production needs, and they guarantee that any package can be unpacked on the most restrictive system (Windows), into any directory whose pathname doesn't exceed 127 characters, and the resulting absolute paths will never exceed the 254 character limit.

It is the responsibility of the person or program that picks the location of the top-level directory to ensure that the target directory's pathname is 127 characters or less. Again, this issue only exists on Windows systems; other systems have no practical limit, in the simple environments for which this specification is intended.

5 Rules for URIs

URIs in the PPML file of a Conforming Package Extraction (e.g. `EXTERNAL_DATA`) shall meet the requirements of either `rel_path` or `AbsoluteURI` in RFC 2396. Examples of a `rel_path` URI are: "`myfile.eps`", "`./myfile.eps`" and "`images/myfile.eps`". (URIs use slashes, not backslashes.) Examples of an `AbsoluteURI` URI are `"http://foo.com/test.eps"` and `"ftp://ftp.foo.com/test.eps"`.

Each `rel_path` URI must reference a file that is within the Conforming Package Extraction.

If absolute URIs are used, it is the responsibility of the Producer and Consumer to ensure that the Consumer can access all referenced data – it is not a function of this packaging specification. For instance, to access http://foo.com/test.eps, the Consumer must have HTTP support.

This rule prohibits `abs_path` URIs (e.g. "`/working/test1/test2.eps`") and platform centric URIs (e.g. "`file:///c:foo/bar.gif`").

5.1 Preserve Case

For each letter in a filename or directory of a Conforming Package Extraction, the corresponding letter in its referencing URI (in the PPML file) must be identical, including case. For example, file "`images/Foo.gif`" must be referenced in the URI as "`images/Foo.gif`" and not as "`images/foo.gif`", "`Images/foo.gif`" or "`IMAGES/FOO.GIF`". This rule ensures that each file in Conforming Package Extraction on a case-sensitive platform, such as Unix/Linux, can be referenced by its corresponding URI.

5.2 Use Escape Characters

According to RFC 2396, certain characters must not appear directly within a URI. Instead the character must be escaped by using a "%" followed the two digit hex value of the character. For example, the `rel_path` URI that references the file `"first time"` would be `"first%20time"` where `0x20` is the value of the space character. The excluded characters for the `abs_path` portion of the URI are the 10 characters space, "#", "%", "`;`", "`[`", "`]`", "^", "`'`","`{`" or "`}`". For the first segment of the `rel_path` URI (called `rel_segment`), the excluded characters are the same except that the ":" is an excluded character and the ";" is not. See RFC 2396 for the full grammar.

If a URI in a PPML file of a Conforming Package Extraction references a file whose name includes characters that cannot be directly represented in a URI as described above, then each such character must be escaped as described above.

Annex A
(informative)
Example

The following example shows a simple PPML dataset conforming to these rules: a PPML file that references three objects. Shown below are:

- The content of the PPML file, with the URIs highlighted.
- A Windows directory listing of the entire dataset: the PPML file and its three referenced files.
- A directory listing of a ZIP file created from this dataset.

The highlighting in these listings is not significant; it only draws attention to relevant information.

```
<?xml version="1.0" encoding="UTF-8"?>
<! DOCTYPE PPML PUBLIC
   "-//PODi//DTD PPML 3.00//EN" "http://www.podi.org/ppml/ppml300.dtd">
<PPML Creator="Test">
<DOCUMENT_SET Label="URI example">
<DOCUMENT Dimensions="594 840">
<PAGE Label="Page 1">
<MARK Position="0 800">
<VIEW/>
<OBJECT Position="0 0">
<SOURCE Format="application/postscript" Dimensions="40 40">
<EXTERNAL_DATA Src="object-1.eps"/>
</SOURCE>
<VIEW/>
</OBJECT>
         </MARK>
         <MARK Position="50 800">
            <VIEW/>
            <OBJECT Position="0 0">
               <SOURCE Format="application/postscript" Dimensions="40 40">
<EXTERNAL_DATA Src="object-2.eps"/>
               </SOURCE>
               <VIEW/>
            </OBJECT>
         </MARK>
         <MARK Position="100 800">
            <VIEW/>
            <OBJECT Position="0 0">
               <SOURCE Format="application/postscript" Dimensions="40 40">
<EXTERNAL_DATA Src="images/object-1.eps"/>
               </SOURCE>
               <VIEW/>
            </OBJECT>
         </MARK>
       </PAGE>
     </DOCUMENT>
</DOCUMENT_SET>
```



```
</PPML>
```

Windows directory listing of the files to be packaged:

```
Directory of E:\PPML
URIS~1   PPM           873  08-22-01  1:35p URIs.ppml
IMAGES         <DIR>        08-22-01  1:42p images
OBJECT-1 EPS       190,677  10-18-00 10:13a object-1.eps
OBJECT-2 EPS        37,122  10-17-00  8:14a object-2.eps

Directory of E:\PPML\images
OBJECT-1 EPS       193,366  10-16-00  1:39p object-1.eps
```

Note: The entire project, including the PPML file itself, was created within the directory E:\PPML, but that directory names never appear in the relative URIs in the PPML. In fact the project could have been created in any directory on any drive, and the PPML code and the ZIP package would look the same as shown here, because both the PPML and the ZIP package use relative locations.

A.1 Directory listing of conforming ZIP file

```
PKZIP(R)  Version 2.50  Compression Utility for Windows 95/NT  4-15-1998
 Viewing .ZIP: URItest.zip

  Length  Method    Size  Ratio   Name
  ------  ------    -----  -----   ----
     873 DeflatN      383 56.2%   URIs.ppml
       0 Stored         0  0.0%   images/
  190677 DeflatN    62047 67.5%   object-1.eps
   37122 DeflatN    14145 61.9%   object-2.eps
  193366 DeflatN    62483 67.7%   images/object-1.eps
  ------           ------ -----   ----
  422038           139058 67.1%      5
```

Annex B
(informative)
Change History

Version 3.0, May 2010

- **Updated revision to PPML 3.0**

