Open Standard Print API (PAPI): Additions for Printer Capabilities API

Version 0.2 (DRAFT)

Open Standard Print API (PAPI): Additions for Printer Capabilities API: Version 0.2 (DRAFT)

Version 0.2 (DRAFT) Edition

Table of Contents

Chapter 1. Printer Capabilities

1.1. Introduction

In the context of this document, *printer capabilities* refers to information about the features, options, limitations, etc. of a print device (either an actual device, or an abstract device which may represent a group or pool of actual devices). This includes such information as:

- Does the printer support color printing?
- At what resolution(s) can the printer print?
- What input trays are present?
- What size media is loaded in each tray?
- Which trays are manual-feed and which are auto-feed?
- Can the printer print duplex output?
- What is the printable area on each of the loaded media?
- What output bins are present?
- What finishings (staple, punch, etc.) does the printer support?
- What combinations of features are not allowed together?
- What features should be presented on the print user interface?
- ...and many others...

The uses of printer capabilities by applications include:

1. To control how to display print options in a print UI dialog. Examples:
   - What values to put in the bin selection pull-down list
   - Whether or not to gray-out the duplex option when a particular output bin has been selected
   - Whether or not to display a color vs. back-and-white selection
2. To control how the print datastream is generated. Examples:
   - How large an image to draw to fill the printable area
   - How much to shift the image if "3-hole punch" finishing has been selected
   - How to request that the printer print on paper from the manual envelope feeder
3. To do job validation and printer selection. Examples:
   - Can I print this job with these options on this printer?
   - Find a printer which can print this job with these options.

1.2. Definitions

Driver:

In the context of this document, this is a software program that, possibly together with some external representation of printer capabilities, can translate generic graphic/drawing commands issued from an application into a printer-specific datastream which will render those commands on paper. The driver may also be able to transform graphic/drawing commands from an input datastream into a printer- specific output datastream (e.g. translate Postscript into raster images).

PPD (Postscript Printer Description) files:

Files which contain capabilities information in a special text format that was developed by Adobe for devices which include a Postscript interpreter. In addition to capabilities information, PPD files contain information about how to present capabilities to an end-user (e.g. in a GUI dialog) and how features can be selected and settings can be changed. Postscript drivers rely heavily on PPD files to generate the correct Postscript datastream. PPD files are heavily used on both Windows and Unix platforms, and on Linux they currently represent the primary repository for capabilities information. The specification of the PPD format can be found at http://partners.adobe.com/asn/developer/pdfs/tn/5003.PPD_Spec_v4.3.pdf.

UPDF (Universal Printer Description Format):

This is a relatively new, standard XML format for representing printer capabilities. UPDF is not tied to a particular printer datastream such as Postscript, and it is intended to support representation of dynamic printer capabilties better than PPD.

Constraint:

This is a restriction on the printer capabilities where some combination of two or more attributes/values are not allowed together. This may be due to printer hardware limitations or to the disallowing of combinations which do not make sense by the printer vendor or the print system administrator. An simple example constraint would be "transparencies cannot be selected when printing duplex".

1.3. Objectives

This section attempts to describe the objectives of the PAPI printer capabilities support. It is important to understand these objectives in order to understand why the support is structured the way that it is.

1.3.1. Standard printer capabilities API

There is no standard API which a Linux application can use to retrieve printer capabilities regardless of the device, the driver, and the print server being used. This makes it very difficult for application writers to support generating print data without writing multiple versions of the print logic or without tying the application to very specific print system environments. This specification provides the standard API, making applications which use it independent of the underlying print system.

1.3.2. Independent of underlying source of capabilities

The capabilities information returned to the application could come from many different sources and be in many different formats, including:

- PPD files
- UPDF database
- SNMP queries
- Device drivers

The API defined here must hide these differences so that the application is independent of which of the above implementation(s) are used.

1.3.3. Support returning information in context

The API must support a means for requesting capabilities information *in the context of* a particular set of job options. For example, a way is needed to request the printer capabilities given that medium and color/black-and-white selections have already been made.

1.3.4. Support returning constraints

The API must support a means for returning constraints on printer capabilities (see earlier definition of "constraint"). This allows applications to not submit jobs with disallowed combinations of options, and to display better print job dialogs (gray-out potentially conflicting options, highlight conflicting options that have been selected, display an error message when invalid combinations are submitted, etc.).

The constraints returned should allow some level of "boolean logic", including negation, to simplify the information returned. For example, to not allow doing finishing when transparencies are selected as the medium, it would be preferable if the constraints could express "(type = transparency) AND (finishing NOT= none)" instead of having to list a combination of "(type = transparency)" with every possible fininshing value other than "none".

1.3.5. Support returning display hints

The API should support a means for returning "display hints". This is information that the application can use to display print options in a print dialog that is easy to use. For example, returning information about which options should be displayed on the "main window", which should be displayed in an "advanced" dialog, and which should not be displayed at all.

1.3.6. Support logically grouping features

The API should support a means for returning logical groupings of printer features. This is information about combinations of lower-level features that can be displayed and selected as a group to make the user interface easier to use. For example, a group of features called "black-and-white-draft" could include a logical setting of the color, resolution, and print density options.

The feature group support should be an open, extendible way for printer vendors and print administrators to express logical and commonly used groupings of print options that make it easier for end-users to take advantage of lower-level printer features. They should *not* be used to blindly list all possible combinations of a set of options, whether or not all the combinations make sense.

1.4. Interface

1.4.1. Query Function

The API used by the application to retrieve printer capabilities is the papiPrinterQuery function. See the description of that function for further details.

1.4.2. Capabilities Attributes

In addition to the xxx-supported attributes defined by the IPP standard [RFC2911], this section defines new attributes needed to satisfy the objectives described earlier.

1.4.2.1. job-contraints-col

(1setOf collection) Constraints are expressed in the printer object's job-constraints-col attribute. This attribute is multivalued with each value having collection syntax. Each value is, in fact, an attribute list that represents *one* combination of job attributes/values which are not allowed for that printer. If an attribute in the collection does not have a value, then *all* values of that attribute are disallowed in this combination.

The set of values associated with job-constraints-col represents the complete set of job attribute constraints associated with the containing printer object.

The attribute values in job-constraints-col may also be in range syntax, if the corresponding job attribute has integer syntax. This represents the included (or excluded, if the attribute is named in job-constraints-inverted) range of values for that attribute within that constraint.

1.4.2.2. job-contraints-inverted

(1setOf type2 keyword) One attribute which may appear within the job-constraints-col collection is job-constraints-inverted. This attribute is used to list those attributes in the job-constraints-col value whose values are to be *excluded* ("no equal to" values) from the constraint. If an attribute name is not included in job-constraints-inverted attribute, then that attribute's values are to be included ("equal to" values) in the constraint.

You can think of the each attribute in a job-constraints-cols value as AND-ed together to express a disallowed combination of options: "(attr1 == values) AND (attr2 == values) AND ...". The job constraints-inverted attribute lists those attribute/value comparisons which are to be "!=" instead of "==".

1.4.2.3. Example

Here is an example of how the job-constraints-col attribute can be used to express various printer constraints. The example is expressed in pseudo-code with curly brackets enclosing each collection value and attributes within each collection are shown on separate lines with commas separating the values:

```
job-constraints-col =

  /* Constraint: no high print quality with 240 dpi resolution     */
  /*   (print-quality == high) AND (printer-resolution == 240dpi)  */
  {
    print-quality = high
    printer-resolution = 240dpi
  }

  /* Constraint: no transparency with duplex                        */
  /*  (sides != one-sided) AND (media == transparency)              */
  {
    job-constraints-inverted = sides
    sides = one-sided
    media = transparency
  },

  /* Constraint: no finishing with heavy-stock media                */
  /*  (finishings != none) AND (media == heavy-stock)               */
  {
    job-constraints-inverted = finishing
    finishings = none
    media = heavy-stock
  },

  /* Constraint: no duplex printing of A4 paper in landscape       */
  /*  (sides != one-sided) AND (media == A4) AND                    */
  /*  (orientation-requested == landscape)                          */
  {
    job-constraints-inverted = sides
    sides = one-sided
    media = A4
    orientation-requested = landscape
  },

  /* Constraint: no duplex printing of COM-10 envelopes            */
  /*  (sides != one-sided) AND (media == envelope) AND              */
  /*  (media-size == com10)                                         */
  {
    job-constraints-inverted = sides
    sides = one-sided
    media = envelope
    media-size = com10
  },

  /* Constraint: no stapling of greater than 50 sheets             */
  /*  (finishings == staple) AND (job-media-sheets > 50)            */
  {
    job-constraints-inverted = job-media-sheets
    finishings = staple
    job-media-sheets = 1-50
  }
```

```
};
```

1.4.3. Validation Function

The API used by the application to validate print job attributes against printer capabilities is the papiJobValidate function. See the description of that function for further details.

Appendix A. Change History

Version 0.2 (November 21, 2002)

- Added third capabilities usage to "Introduction".
- Added paragraph about boolean logic under "Support returning constraints" objective. Also clarified wording of how this can be used to improve print dialogs.
- Changed "Support returning composite features" to "Support logically grouping features" so that the objective does not imply a specific solution.
- Removed "Support Device Object" objective.
- Added "job-constraints-col" attribute.
- Added "job-constraints-inverted" attribute.

Version 0.1 (September 25, 2002)

- Original draft version

End of Document