FastLane

Description

FastLane is a command-line tool that is part of PitStop Server and that allows to extract information and attachments from a PDF file in a very fast way.

It is advised to have a look at the FastLane documentation before using this app, especially when using the direct query option (see further).

Compatibility

Switch 2022 Fall release. Windows or Mac OSX.

Compatibility third-party applications

This app relies on the presence of PitStop Server 2023 or higher. The presence of PitStop Server is automatically detected.

Connections

FastLane does not accept folders as input.

The app has outgoing traffic-light connections: Success, Warning and Error. The Warning connection is used when information is requested that is not present in the file, e.g. font information is requested, but the file contains no fonts. The Error connection is available but will very rarely be triggered.

Flow element properties

Name	FastLane
Description	
Operation mode	Extract information
Info dataset name	FastLane
Query mode	Query presets
Document info	Yes
Pages info	Yes
Units	mm
Page boxes info type	Coordinates
Fonts info	No
Layers info	No
Color info	Yes
Color info mode	Document colors
Document output intent info	No
Ink order info	No
Spectral data info	No
Form fields info	No
Annotations info	No
DParts info	No
XMP dataset name	None

Operations mode: the app has two operating modes, “Extract information” and “Extract attachments”.

These are the properties in the information extraction mode:

- **Info dataset name**: the name of the dataset into which all the requested information will be written. In this version it will always be a JSON dataset.

- **Query mode**: a dropdown with the values “Direct query” and “Query presets”. When choosing “Direct query” a couple of additional properties will be shown where you can specify most of the options available for the command line tool. When choosing “Query presets” several PDF properties will be shown for which you can choose to include them in the dataset or not. Using “Query presets” hides the necessity to work out the required patterns to achieve the desired result. Default: Query presets.

These are the properties in the direct query mode:

- **Pattern**: For more information on how to define a pattern, consult the FastLane documentation. On the command line the --pattern option must be enclosed in surrounding quotes, and it must be defined as a single string. Here the quotes are not required, and the pattern can be written on multiple lines which increases the readability. The default value contains a sample pattern.

- **JSON**: For more information on how to define a JSON structure, consult the FastLane documentation. The same comments apply as for the “Pattern” property: the surrounding quotes can be left out and the JSON structure can be defined on multiple lines. The default value contains a sample JSON structure that fits the pattern defined in the “Pattern” property.

- **Scope**: For more information on how to define a scope for the query, consult the FastLane documentation. The surrounding quotes can be left out, but as a scope is a simple string it cannot be specified on multiple lines. The default value contains a sample scope.

  There is no possibility to use Switch variables for any of the previous properties because this is not relevant.

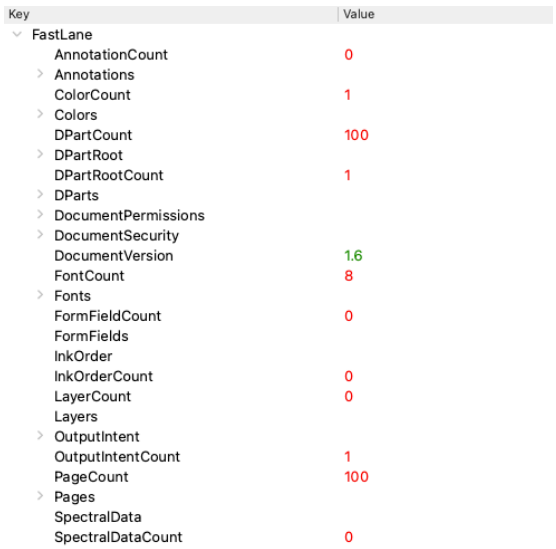
When choosing “Query presets” several PDF properties will be listed. You can choose to include them or not using a No-Yes dropdown. All requested properties will be added to the dataset. See below for details on what exactly is added to the dataset.

- **XMP dataset name**: the XMP section of a PDF is treated differently from the other PDF properties. As Switch supports datasets with the dataset model XMP, this section from the PDF is added exactly as it is. The information is the same as that available as the embedded metadata in the metadata dialog. In other words, it is not necessary to extract the XMP into a dataset for use in the definition of variables, but if you want to store the XMP section as a file this makes it easier. Default: None.

Query presets

Defining the correct values for the --pattern and –json parameters for the FastLane command line tool is not always easy. It requires a thorough understanding of the internal structure of a PDF file. Moreover, the information extracted from the PDF is not always easily interpretable. To make life a lot easier the app offers a set of presets that use predefined patterns and JSON structures for the output. The output as generated by FastLane is also postprocessed in most cases to make it more understandable/usable. For example, sizes extracted from a PDF will always use points as a unit because the internal values in the PDF are always in points, but it is often more convenient to have the values in millimeters or in inches. Similarly, in the PDF structure a page has no size; it is represented as an array of coordinates, but it will usually be more convenient to have a width and height.

The top level of the JSON dataset will always be called “FastLane”. Each property for which information is requested will add one or more keys underneath the top level. Here is a screenshot of the result in the metadata window in Switch:



Document info

This adds the following properties to the JSON dataset:

- DocumentVersion: a string with the version number of the PDF.
- DocumentPermissions with the following sub-properties (all Boolean):
    - PrintLowQuality
    - PrintHighQuality
    - ModifyOther
    - Commenting
    - FillAndSign
    - DocAssembly
    - ExtractAccessibility
    - ExtractNonAccessibility
    - ModifySecurity
- DocumentSecurity:
    - Encrypted: Boolean
    - EncryptionType: string

Pages info

This adds the following properties to the JSON dataset:

- PageCount: a number with the number of pages in the document
- Pages: an array with one element per page containing the following properties:
    - MediaBox: depending on the chosen values for the dependent properties this will either be an array of 4 values with the coordinates of the page, or an object with the properties “width” and “height”. The units can also be chosen. Note that the values are always the effective values! In other words, the values have been transformed considering the page’s rotation and scaling (user units).
    - CropBox
    - BleedBox
    - TrimBox
    - ArtBox
    - Index: the page number (one-based)
    - Label: the label of the page. If the page does not have a label the value is the same as that of the index.
    - Orientation: one of Portrait, Landscape or Square.
    - Rotation: 0, 90, 180, 270. This is informational so it is possible to detect that a page has a rotation defined. The values of the different page boxes have already been calculated using the rotation.
    - UserUnit: an integer number. The same comments apply as for the Rotation property.

Fonts info

This adds the following properties to the JSON dataset:

- FontCount: an integer with the number of fonts used in the file
- Fonts: an array of objects with the following properties:

- FontName: for subsetted fonts the prefix to make the font name unique is removed.
- FontType: one of Type1, Type3, MMType1, TrueType, CompositeType1 or CompositeTrueType.
- FontEmbedState: one of NotEmbedded, Embedded or Subset.
- EmbedOpenType: Boolean.
- ItalicAngle: number. This property is not always present when the font is not italic.
- Flags: an object with more properties of the font. In PDF the font flags are represented by a number the individual bits of which represent Boolean values for FixedPitch, Serif, Symbolic, Script, Italic, AllCap, SmallCap, ForceBold.

Note that a PDF may define a font but that does not necessarily mean that it is being used.

Layers info

This adds the following properties to the JSON dataset:

- LayerCount: the number of layers in the file
- Layers: an array of objects with the following properties:
  - Name
  - Type (always OCG)
  - Usage: this property is not always present. If it is it has sub-properties for the viewing state and/or printing state of the layer.

Layers can be organized in groups and in a hierarchical manner. The array created by the app, however, shows a flat list of the layers, or to be more accurate, the optional content groups defined in the file. Note that there is no certainty that a defined layer has objects associated with it.

Color info

There is a dependent property that allows to define whether you just want a list of all the colorants in the document, or whether you also want it per page. The following properties are added “DocumentColors” and “PageColors” properties to the JSON dataset:

- DocumentColors: there is a sub-property “Colorants” which is an array of the names of the colorants defined in the document.
- PageColors: an array with an element per page which contains a “Colorants” property which in its turn is an array of the names of the colorants defined for that page.

Note that the fact that a colorant has been defined does not necessarily mean it is being used.

Document output intent info

The post-processing of the output intent analyzes the header of the included ICC profile and adds the following properties to the JSON dataset:

- OutputIntentCount: this will be 0 or 1.
- OutputIntent
    - Class: for an output intent this will usually be “Output device profile”.
    - ColorSpaceName: the value will usually be “CMYK”, but “Gray” and even “RGB” are also possible.
    - Colorants: an array with the name of the colorants defined in the ICC profile. For most output intents this list will be empty.
    - Description: a string describing the output intent
    - DeviceModel
    - Manufacturer
    - Version: a string with the version number of the ICC profile.

Ink order info

The root output intent dictionary can contain a list of inks used in the document in the order in which they should be printed. The name of the entry in the dictionary is “MixingHints”.

The following properties are added to the JSON dataset:

- InkOrderCount
- InkOrder: an array of names of the inks in order.

Spectral data info

As of PDF 2.0 the root output intent dictionary can also contain an entry called “SpectralData”. It is an array of spot colors for which the PDF file contains the spectral data information. This data is stored in the so-called Color Exchange Format, or CxF.

This adds the following properties to the JSON dataset:

- SpectralDataCount
- SpectralData: an array with the names of the inks

Form fields info

This adds the following properties to the JSON dataset:

- FormFieldCount: an integer with the number of form fields in the file.
- FormFields: an array of objects with the following properties:
    - Name
    - Type: one of Btn, Tx, Ch, Sig.
    - Value: the value depends on the type of field. For text fields this will a string with the contents of the field, for a checkbox it will be On/Off, etc.
    - Page: the zero-based index of the page on which the field is placed.
    - Position: an array of 4 coordinates with the position of the field. These values are not transformed and are in points!

Annotations info

There are quite a few objects in a PDF that fall under the category annotation. This property adds an object with a counter for the different annotation types:

- AnnotationCount
- Annotations
  - Text
  - Link
  - FreeText
  - Line
  - Square
  - Circle
  - Polygon
  - PolyLine
  - Highlight
  - Underline
  - Squiggly
  - StrikeOut
  - Caret
  - Stamp
  - Ink
  - Popup
  - FileAttachment
  - Sound
  - Movie
  - Screen
  - Widget
  - PrinterMark
  - TrapNet
  - Watermark
  - 3D
  - Redact
  - Projection
  - RichMedia

DParts info

PDF files can contain so-called Document Part Metadata. It is a construct that allows to insert additional information into the PDF for use by a consumer. The type of information that is inserted in this way is left open by the PDF specification. Therefore, the app adds the DPart information as it is stored in the PDF file without any post-processing.

The DPart information can be stored at two levels, at the root level of the document and at the page level.

The following properties are added to the JSON dataset:

- DPartRootCount
- DPartRoot
- DPartCount
- DParts: an array of objects with the properties “DPartPage”, the 0-based index of the page, and “DPart” with the complete content of the document part metadata.

Patterns

As a reference for those who want to exploit the possibilities of the FastLane tool further, here are the patterns used by the app to collect the different pieces of information.

Document version:

```
<</__pdf_version__ {version}>>
```

Document permissions:

```
<</__doc_permissions__
  << /__perm_print_lq__ {print_lq}
     /__perm_print_hq__ {print_hq}
     /__perm_modify_other__ {modify_other}
     /__perm_commenting__ {commenting}
     /__perm_fill_and_sign__ {fill_and_sign}
     /__perm_doc_assembly__ {doc_assembly}
     /__perm_extract_accessability__ {extract_accessability}
     /__perm_extract_non_accessability__ {extract_non_accessability}
     /__perm_modify_security__ {modify_security}
  >>
>>
```

Document security:

```
<</__doc_security__ {security}>>
```

Page information:

```
<</Type /Page /Rotate {rotation?}
              /UserUnit {userunit?}
              /__page_label__ {label}
              /__page_boxes__ <</__orientation__ {orientation}
                               /__media_box__ {mbox}
                               /__crop_box__ {cbox}
                               /__bleed_box__ {bbox}
                               /__trim_box__ {tbox}
                               /__art_box__ {abox}
                              >>
>>
```

Font information:

```
<</Type /Font
  /FontDescriptor {fontdescriptor?}
  /__font_info__ << /__embed_state__ {embedstate}
                    /__embed_opentype__ {embedopentype}
                    /__font_name__ {fontname}
                    /__font_type__ {fonttype}
                 >>
>>
```

Color information:

```
<</__color_info__ << /__colorant_names__ {colorants}>> >>
```

Layer information:

```
<</Type /OCG >>
```

Form fields information:

```
<</Type /Annot
  /Subtype /Widget
  /T {name}
  /FT {type}
  /V {value?}
  /AS {as?}
  /Rect {position}
  /P <</Type /__page_ref__ /__page_index__ {pageidx} >>
>>
```

Output intent:

```
<</DestOutputProfile <</__stream_data__ (${outputPath})>> >>
```

where ${outputPath} has to be substituted by the path where FastLane should write the contents of the output intent.

Printing order:

```
<</PrintingOrder {inkorder}>>
```

Spectral data:

```
<</Type /OutputIntent /SpectralData {spectraldata}>>
```

DPartRoot:

```
<</Type /DPartRoot /DPartRootNode <</DPM {dpartroot}>> >>
```

DParts:

```
<</Type /Page
  /DPart <</Type /DPart /DPM {dpart?} >>
  /__page_index__ {pageidx}
>>
```