BitstreamFormat Workbench

<?xml version="1.0" encoding="utf-8"?>
<html>
User's Guide for DSpace+1.6 BitstreamFormat+Workbench application.
This is a command-line-driven administrative tool that reports on
and also manipulates the format technical metadata in Bitstreams.

About BitstreamFormats

In DSpace+1.6, the file-format technical metadata is now stored in external *format registries* which are only referenced by the

```
BitstreamFormat
```

object in the DSpace data model.
For backward-compatibility there is a "DSpace" external registry that mostly duplicates the format model of versions up to DSpace 1.5.
For more details, see BitstreamFormat Renovation.

The Data Model

Every Bitstream refers to a *BitstreamFormat* (BSF) object, which defines its format technical metadata. The BitstreamFormat, in turn, refers to one or more entries in external *data format registries*, which have all of the actual technical metadata describing the format. A few useful pieces of metadata are cached in DSpace for each BSF: its descriptive proper *name*, its *MIME type*, etc.

The BitstreamFormat is identified by its integer database identifier (DB ID). That is the only identifier which *must* be unique; two BSFs may have identical names, although that should be discouraged.

Along with a reference to a BSF, each Bitstream has two related properties:

1. **Confidence** - a measure of the certainty with which its format was identified, which tells you how accurate it is *likely* to be.
2. **Format Source** – record of the entity or software module that assigned the format to this Bitstream.

You can use the *confidence* and *source* properties to pick out Bitstreams for further examination or reprocessing.

Each BSF is also *bound* to one or more *external format identifiers*, each of which indicates an entry in an external format registry. For example, the format-ID

```
PRONOM:x-fmt/384
```

represents the
PRONOM entry for a version of PostScript. If you look in the PRONOM registry you will find all sorts of other information, from signatures to help identify a file to standards documents describing the format in detail.

However, this Format Workbench utility is primarily concerned with the Bitstream's reference to its BitstreamFormat. It does not offer much control or information about the inner properties of the BSFs. The administrative GUI has pages to help you view and manage BSFs.

Format Workbench Functions

The Format Workbench has two primary functions: Reporting on formats, and manipulating (changing) format bindings. These are implemented by the following subcommands:

1. Reporting
   a. Report binding of Format to Bitstream - see

      ```
      -R
      ```

    b. Report Guessing new Format for Bitstream - see

    ```
    -R -g
    ```

    c. Histogram of BSF Usage - see

    ```
    -H
    ```

    d. List all BSFs - see

    ```
    -L
    ```

2. Manipulation
    a. Guessing (automatic format identification) format of Bitstreams - see

    ```
    -G
    ```

    b. Forcibly changing BSF of Bitstreams - see

    ```
    -C
    ```

The next sections gives details of how you can apply these subcommands to chosen sets of Bitstreams.

Invoking

```
Formats
```

Formats is a command-line Java application, which must be run on the DSpace server host.
Invoke it with *dsrun*, e.g.:

```
${DSPACE_HOME}/bin/dsrun org.dspace.administer.Formats options
```

Options are specified in the usual Unix-derived style: an option is a single dash followed by a single letter, or double-dash followed by a word, e.g.

```
-h
```

or

```
--help
```

. The *help* option
displays a summary of all options and the kinds of values they accept, although there is much more to know about the command than that.

Some options produce output on the "standard output" stream. In a Unix environment you can save this output by using your shell to redirect the standard output into a file. The standard error stream is only used for diagnostic messages.

Formats generates reports in a column-oriented format so they can be imported by spreadsheet and database programs, or processed by text manipulation tools like *awk*. The

```
-F
```

option changes
the character that
separates columns, which is comma (*

```
,
```

*) by default.
If the report data contains commas, you may need to change it to something else.

Global Options

These options apply to any subcommand chosen:

```
-v
```

Verbose

Setting the *verbose* option increases the level of detail in any printed report. The exact action depends on the command.
For example, reports which are normally summaries will include details about every selected Bitstream.

```
-n
```

Dry Run

Does a "dry run" of any operation that would normally change the state of the data model. For example, a command to re-identify a format would not make any changes. Helpful when testing a command to see what it *would* do.

Options to Select Bitstreams

Subcommands that operate on a *set of Bitstreams* can accept these options to choose their operands. These subcommands include

```
-R, -G, -C
```

.

```
-a
```

All

Operates on *all* Bitstreams in the archive, as modified by any filter options. This includes Bitstreams that do not belong to any Item, e.g. logo images for Collections.

```
-b
```

Bitstream

Chooses a specific Bitstream identified by its database identifier (i.e. the value of the

```
bitstream_id
```

column. "Filter options have
no effect on Bitstreams chosen by this option."

This option may be repeated on the same command line, so you
can operate on a group of otherwise unrelated Bitstreams.

```
-i
```

Item

Chooses all Bitstreams in *all* Bundles under the specified Item. The Item maybe identified *either* by a Handle or by a database identifier. If any filter options are specified, they will act to restrict the set of chosen Bitstreams.

This option may repeated on the same command line, so you can operate on a group of otherwise unrelated Items.

```
-s
```

SequenceID filter

Restricts the operand Bitstreams to those whose SequenceID matches the given integer. This is mainly useful in conjuction with the

```
-i
```

option to select Bitstreams under an Item.

```
-S
```

Source filter

Filter that selects *only* those Bitstreams whose Format Source property matches the *regular expression* specified as the option argument.
For example,

```
-s '^TextHeuristic'
```

selects only those
Bitstreams with a source that starts with the word

```
TextHeuristic
```

.

```
-u
```

UserFormatDescription filter

Filter to select *only* those Bitstreams with a UserDefinedFormat set. Note that it does not match anything *in* the format name, it simply chooses the Bitstreams that have user-defined formats. (Perhaps searching on the text of the format name itself would be a useful extension.)

```
-L
```

List of BitstreamFormats

Generates a list of BitstreamFormat entries in the DSpace data model, one line for each unique BSF (after a heading identifying the columns). The columns are:

1. "BSF" to identify the type of data.
2. Database identifier of the BSF.
3. Proper name of the format.
4. MIME type
5. Support level (one of

   ```
   UNKNOWN, KNOWN, SUPPORTED
   ```

)
6. Canonical filename extension, if any
7. (Optional, when -v option specified) External Identifier

```
-R
```

Report Subcommand

Prints a report of formats bound to the chosen set of Bitstreams. It always includes a summary with the following columns:

1. "FORMAT-SUMMARY" to mark the type of data.
2. BitstreamFormat database identifier (integer)
3. BitstreamFormat proper name
4. Count of number of selected Bitstreams bound to this format
5. Format-source value from those Bitstreams
6. Format Confidence value from those Bitstreams

See the "Histogram" command for a slightly different kid of summary output.

With the

```
-g
```

(guess) Option

Attempts to re-identify the format of each selected Bitstream and adds the results to the report. This is a very effective way to evaluate new format identifier methods and test the effect of changing your configuration.

Displays addtional summary lines that give the number of Bitstreams counted for each "type" of identification result (e.g. one *type* is "a formerly unidentified Bitstream that now has a valid identification").
It contains the following columns:

1. "GUESS-SUMMARY" identifies the type of data
2. result-type, a number unique to this report, refereneced by the "primary" and "secondary" result-type columns in the verbose Bitstream lines
3. count of Bitstreams registering this kind of result – note that each Bitstream may register more than one type, so total of counts may be more than the total of Bitstreams
4. human-readable description of this result-type

With the

```
-v
```

(verbose) Option

The verbose option adds a data line for each selected Bitstream, as well, with the columns:

1. "BITSTREAM" to mark the kind of data
2. Handle of the Item that owns it, if known
3. Database identifier of the Bitstream
4. Name (i.e. filename) of the Bitstream
5. DB ID of its current format
6. Confidence (as a text word) of its current format identification

7. Format Source of its current format identification
8. (ONLY with "guess") primary result-type of guess
9. (ONLY with "guess") secondary result-type of guess
10. (ONLY with "guess") DB ID of newly-identified format
11. (ONLY with "guess") confidence of newly-identified format
12. (ONLY with "guess") format source of newly-identified format

```
-H
```

Histogram Subcommand

Generates a "histogram" report which shows usage counts of BitstreamFormats
and external format registries – i.e. a histogram of the number of Bitstreams
referencing each of them.

It prints two kinds of histogram lines: first, "BSF" lines listing
BitstreamFormats (perhaps broken down into subsets of a BSF at
different confidence levels, or combined with different format sources),
and second, a summary of references to each external Format Registry.

The BSF lines have the following columns:

1. "BSF" to mark the type of data
2. Database ID of the BSF
3. Name of the BSF
4. Count of Bitstreams referring to the BSF described by this row. NOTE: On each row, this is the sum of other sub-rows with different confidence or source.
5. (optional, when selected) external format identifier
6. (optional, when selected) MIME type
7. One, or neither, of the following pairs:
    - Format Confidence:
        a. (optional, when selected) Confidence as symbolic value
        b. (optional, when selected) Count of Bitstreams with that Confidence
    - Format Source:
        a. (optional, when selected) Format Source value
        b. (optional, when selected) Count of Bitstreams with that Source

The per-Registry summary lines have the following columns:

1. "REGISTRY" to mark the type of data
2. Name of the registry
3. Count of BitstreamFormats with at least one external identifier from this registry – though note that each BSF is only counted once.
4. Count of Bitstreams whose BSF refers to to this Registry. As with the BSF, each Bitstream is only counted once, although it may count for more than one Registry.

```
-c
```

(columns) Option

Selects optional columns to include in the Histogram report.
Must be a comma-separated list of keywords (which may be abbreviated to
the first letter), including:

**E*xternal-Identifier - show all external identifiers bound to each BSF, listed on separate rows.
**M*IME-Type - add the MIME-type configured for each BSF.
**C*onfidence - add a separate row for each distinct value of Confidence in the Bitstreams referencing this BSF, and counts thereof.
**S*ource - add a separate row for each distinct value of Format Source in the Bitstreams referencing this BSF, and counts thereof.

```
-G
```

Guess Subcommand

Interactively "guesses" the format of each selected Bitstream using the
stack of format identifier methods. Displays a description of the Bitstream,
and then a list of the format hits in order of priority.
The user chooses a hit by answering with its index number, or chooses to skip
to the next Bitstream. Nothing is changed if the command is interrupted.

```
-y
```

(yes) Option

If the

```
-y
```

option is added to a Guess subcommand, the first
format hit is *always* accepted without question, and there is no
interactive question. The diagnostic and confirmation messages are
still shown, however.

```
-C
```

Change Subcommand

Forcibly change the BSF of indicated Bitstreams to the number given as the
value of the

```
-C
```

option. Use this *only* when you already
know the exact BSF to be assigned to a specific group of Bitstreams.

Usage Examples

How were formats assigned to an Item?

To learn how an Item's Bitstreams got their formats, get a format report
on the item with the verbose option. For example, given
Handle

```
123456789/8
```

, the command is:

```
dspace/bin/dsrun org.dspace.administer.Formats -R -v -i 123456789/8
```

..and observe the output. The Format Source and Confidence columns are your
best clues to where the format identification came from.
Bitstream 2 has "EditItemServlet" in the Source and a "MANUAL" confidence, so
its format appears to have been set explicitly through the JSP
administrative UI. The others appear to have been automatically
identified, since the sources are FormatIdentifier methods.

```
1. BITSTREAM, <Handle>, <DBID>, <Name>, <Current-Format>, <current-Confidence>, <current-Source>
   BITSTREAM, 123456789/8/1, 697, "foo.pdf", 15, POSITIVE_SPECIFIC, "org.dspace.content.format.DROIDIdentifier"
   BITSTREAM, 123456789/8/2, 698, "bar.ps", 16, MANUAL, "org.dspace.app.webui.servlet.admin.EditItemServlet"
   BITSTREAM, 123456789/8/3, 699, "license.txt", 2, HEURISTIC, "org.dspace.content.format.TextHeuristicIdentifier"
```

Testing Format Identification Changes

When you make changes to the

```
FormatIdentifier
```

plug-in stack
or change the configuration of format identifier plugins, it is important
to confirm that the change does what you expect with no unintended
consequences.

The easiest way to test format identification is to run *Formats* to
re-identify some Bitstreams and see if the performance is what you expected.
There are two subcommands that can do this, with subtle differences:

1. Bitstream Report with the "Guess" option

Invoke the Report subcommand with the

```
"-g"
```

option to show
what format would be automatically identified for each Bitstream, as
well as its existing format. For example, to show the results for
just one Item,

```
123456789/13
```

, issue:

```
dspace/bin/dsrun org.dspace.administer.Formats -R -v -g -i 123456789/13
```

The disadvantage of this approach is that it *only* shows the first
choice for each Bitstream; it does not show all possible format hits.
This may be inadequate if you are carefully tuning the identifier stack.

The advantage is that it outputs tabular data which is easy to analyze.
The summary lines also show how many formats would have been changed in
various ways.

2. Dry Run of the Interactive Format Identifier

Use the

```
"-G"
```

(Guess) subcommand to re-identify the selected
Bitstreams, but add the dry-run option

```
"-n"
```

to ensure no
formats are actually changed. For example, this command re-identifies
all Bitstreams under Item

```
123456789/13
```

.
Note that you can add the

```
"-y"
```

option to avoid responding to a question for each
Bitstream, since nothing will be changed in dry-run mode.

```
dspace/bin/dsrun org.dspace.administer.Formats -G -n -i 123456789/13
```

The disadvantage of this approach is that the format hits for each Bitstream are displayed as part of an interactive dialogue, so may be more difficult to pick out the data you are interested in. However, the advantage is that it shows the whole set of format hits, in order of preference, for each Bitstream.

Detecting and Evaluating User-Defined Formats

The DSpace data model allows each Bitstream to have a *"user-defined format".* This is an arbitrary string that is supposed to represent a data format not available in the (old, limited) format registry.

Hopefully, with the greater range of formats available from external registries, and with easier customization through the built-in Provisional registry, there is no longer any need to set a user-defined format on a Bitstream. Eventually, the concept of user-defined formats may even be phased out in a future release. It was judged to be too big a change to eliminate it in this release, and it would further complicate the conversion process which was already fearfully complex.

Many existing DSpace archives will contain at least a few Bitstreams with user-defined formats set. To find them, use the Report subcommand with

```
"-u"
```

filter option, selecting all Bitstreams, e.g.:

```
dspace/bin/dsrun org.dspace.administer.Formats -R -v -a -u
```

This will show only the Bitstreams which have a user-defined format set. Note how their BSF is set to the Unidentified format.

There is really no good reason to set user-defined formats any more, and a very good reason to avoid them: Bitstreams with user formats set will not benefit from any format-oriented preservation and reporting tools. Under the new format model, they are simply unidentified.

Eliminating User-Defined Formats

If you wish to try upgrading Bitstreams with user-defined formats to real format-registry entries, use the "Guess" subcommand to interactively identify them. This is only recommended if you are using a more extensive external format registry such as PRONOM; if your DSpace is configured with the old backward-compatible DSpace registry, it is unlikely the true formats of these Bitstreams will be detected.

```
dspace/bin/dsrun org.dspace.administer.Formats -G -a -u
```

Merge Two BitstreamFormats to correct Mistake

In this example, an extra BitstreamFormat entry was created by mistake. It occurred when an identifier method returned the second external format identifier, which had not yet been bound to any BitstreamFormat entry, so it created a new entry. Ideally, there should have been *one* BSF with two external identifiers. (This would happen automatically if the external registry lists the identifiers as synonyms, which is why it was a "mistake").

The solution to this unfortunate situation is to "merge" the BSFs by deleting one of them, and adding its external identifier(s) to the other. Then, any Bitstreams that were using the deleted format must be switched over to the merged one.

Here are the commands you would use to merge BSF 13 *into* BSF 3:

1. List affected Bitstreams, by DB ID

Get DB IDs of bitstreams using BSF 13:

```
% <u>dsrun org.dspace.administer.Formats -R -v -a | awk -F, '/^BITS/ && $5 h1. 13 {print $3}'</u>
```

Alternately, this command prints a string of

```
"-b"
```

options
ready to be appended to a Formats command:

```
% <u>dsrun org.dspace.administer.Formats -R -v -a | awk -F, '/^BITS/ && $5 13 {printf "-b %s ",$3}'</u>
-b 47 -b 346
```

2. Delete one BitstreamFormat

Delete BitstreamFormat #13 – but
FIRST, make a note of all of BSF 13's External Identifiers, since
we will need to add them to BSF 3.

Go into the administrative GUI, and delete BitstreamFormat number 13.
Note that this will force all Bitstreams using that format to the
Unidentified format. You can observe this with a Reporting command, e.g.

```
dsrun org.dspace.administer.Formats -R -v -b 47
```

3. Combine all External Identifiers in Target BSF

Add all of the External Identifiers that had been in BSF 13 to BSF 3.

4. Fix the Bitstreams that lost their formats

There are two ways to go about this: just use the built-in format identification, trusting it to work; or manually force all the Bitstreams to the merged format. The former technique is better if you ever periodically re-identify formats, since Bitstreams that have been forcibly set to a format will not be re-identified automatically – forcing sets the Confidence to MANUAL, the highest value.

1. *To re-identify the formats of the affected Bitstreams:*

   ```
   dsrun org.dspace.administer.Formats -G -b 47 ...
   ```

2. To forcibly set them to Format #3:

   ```
   dsrun org.dspace.administer.Formats -C 3 -b 47 ...
   ```

</html>