



# Getting Started With Panl

A rather pleasing companion to the Apache® Solr® Faceted Search Engine

---

**Author:** Synapticloop

---

This book will get you up and running with a fully featured, SEO friendly, keyword searchable, faceted search engine with an in-built, example search page to test it all out.

---

*(Note: The most up-to-date version of this book can always be found here:  
<https://github.com/synapticloop/panl/tree/main/src/book>)*

## IMPORTANT:

Apache®, Solr® the names of Apache projects, and the multicolor feather logo are registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries.

The mention and references of any Apache projects, sub-projects, or resources in no way constitutes an endorsement for the Synapticloop Panl project.

# WHY?

Because...

---

```
/Caran+d'Ache/true/Black/bDW/
```

---

looks so much nicer than...

---

```
q=*:*&facet.mincount=1&rows=10&facet.field=lead_size_indicator&facet.field=grip_material&facet.field=colours&facet.field=nib_shape&facet.field=diameter&facet.field=cap_shape&facet.field=brand&facet.field=mechanism_type&facet.field=length&facet.field=hardness_indicator&facet.field=grip_type&facet.field=cap_material&facet.field=lead_grade_indicator&facet.field=tubing_material&facet.field=in_built_sharpener&facet.field=disassemble&facet.field=category&facet.field=body_shape&facet.field=clip_material&facet.field=mechanism_material&facet.field=lead_length&facet.field=body_material&facet.field=in_built_eraser&facet.field=grip_shape&facet.field=relative_weight&facet.field=name&facet.field=nib_material&facet.field=weight&facet.field=variants&facet=true&fq=brand:"Caran+d'Ache"&fq=disassemble:"true"&fq=colours:"Black"&q.op=AND
```

---

## TL;DR

If you are looking to get the quickest start on understanding how a Panl server is configured and up and running, then

1. Start with the [Quick Start - The 5 Steps](#) section to get the Solr and Panl servers up and running.
2. Look at the Panl Results Viewer  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/default/>
3. Work through the [A Working Example - The Book Store](#) section to gain a deeper understanding
4. Use your existing dataset with [Integrating An Existing Solr Schema](#)
5. Need help? See the [Panl Configuration](#) section

## QUICK TIP

If you need to restart the example Solr server installation - try these commands:

**Windows:**

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin\solr start -cloud -p 8983 -s "example\cloud\node1\solr"
bin\solr start -cloud -p 7574 -s "example\cloud\node2\solr" -z localhost:9983

**\*NIX:**

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr start -cloud -p 8983 -s "example/cloud/node1/solr"
bin/solr start -cloud -p 7574 -s "example/cloud/node2/solr" -z localhost:9983



**Tip:** Documentation for this book can also be read online, see  
<https://synapticloop.github.io/panl/>

## Copyright Notice

The data used in this book is based on 2mm mechanical pencil data hand curated by the Author. The data and associated Solr search results utilised in this book references manufacturers of writing instruments, specifically 2mm mechanical pencils. The manufacturer's name, the name of the pencil (and/or model number) is a trademark and/or is copyright of the manufacturer.

Additional datasets may have information that is copyright to the respective owners.

Whilst the project is released under the very liberal MIT licence, the sample mechanical data remains under the copyright of Synapticloop, and may not be used for commercial purposes.

~ ~ ~ \* ~ ~ ~

# TABLE OF CONTENTS

<b>README.1ST.....</b>	<b>13</b>
Welcome To Synapticloop Panl.....	13
Additional Panl Niceties.....	13
About This Book.....	14
Nomenclature Used Throughout This Book.....	15
Book Format Conventions.....	18
About Panl Server.....	20
How Many Facets Does Panl Support?.....	21
In-Built Results Viewer / Explainer.....	22
About Panl Generator.....	27
About Apache Solr.....	28
<b>Why Synapticloop Panl?.....</b>	<b>29</b>
What is a LPSE (pronounced 'lapse') code?.....	29
The three types of LPSE codes.....	30
URI Path Nomenclature.....	30
Is Synapticloop Panl For Me?.....	31
<b>What You Will Need.....</b>	<b>33</b>
Download The Resources.....	33
The Panl Directory Structure.....	33
<b>Panl Server Versions.....</b>	<b>36</b>
Solr-Panl-9-x.x.x.....	36
Solr-Panl-8-x.x.x & Solr-Panl-7-x.x.x.....	36
Solr-Panl-6-x.x.x and earlier version.....	36
<b>Quick Start - The 5 Steps.....</b>	<b>38</b>
A Note On Running The Commands.....	39
Windows Commands.....	39
*NIX Commands.....	41
<b>Getting Started.....</b>	<b>43</b>
Downloading the Resources.....	43
The Solr server.....	44
The Panl server.....	44
Creating and Starting a Solr Cloud Instance.....	44
Creating a collection.....	46
Indexing the Data.....	47

Starting The Panl Server.....	48
When Something Goes Wrong.....	49
Restarting the Solr Panl tutorial process.....	50
Stopping Solr.....	50
Deleting a collection.....	51
Re-Starting Solr.....	51
Worst-case scenario.....	52
<b>Additional Data.....</b>	<b>53</b>
All Data Panl Server.....	53
Simple Date.....	53
Creating and Indexing the Data.....	53
panl.properties File Additions.....	54
Mechanical Pencils OR Facet.....	54
Creating and Indexing the Data.....	55
panl.properties File Additions.....	55
Book Store.....	55
<b>A Working Example - The Book Store.....</b>	<b>57</b>
0. High Level Requirements.....	58
1. Understanding the Dataset.....	59
2. Configure the Solr Index.....	61
3. Configure the Panl Server.....	66
The Default panl.properties file.....	67
The Default Search Page Configuration.....	68
4. Determine the Web Pages to Render.....	84
Author and Author series.....	84
Author Listing.....	85
The Iterative Implementation Process.....	85
<b>Working with any Dataset.....</b>	<b>87</b>
Supported Solr Data Types.....	87
Unsupported/Partially Supported Solr Field Types.....	88
Facet and Field Types.....	89
Fields.....	89
Regular Facets.....	90
BOOLEAN Facets.....	90
RANGE Facets.....	91
DATE Range Facets.....	92
OR Facets.....	94
Other Facet Options.....	94
<b>A Brief Introduction To Solr.....</b>	<b>96</b>

Querying Data.....	96
The Solr Managed Schema.....	97
Determining the Appropriate FieldType and Attributes.....	101
The Solr Configuration File.....	105
Query Request Handler.....	105
Highlighting.....	107
<b>Panl Configuration.....</b>	<b>108</b>
The panl.properties file.....	108
The <panl_collection_uri>.panl.properties file.....	109
<b>The panl.properties Configuration File.....</b>	<b>111</b>
Configuring The SolrJ Connector.....	112
Setting The Solr Server URL(s).....	112
Enabling The Panl Results Testing URLs.....	114
The Panl Results Viewer.....	115
The Panl Results Explainer.....	116
Setting HTTP Status Message Verbosity.....	116
Binding Solr collections to Panl URI paths.....	117
<b>The &lt;panl_collection_uri&gt;.panl.properties Configuration File.....</b>	<b>119</b>
Parameter and Operand Definitions.....	119
panl.param.query.....	119
panl.param.sort.....	120
panl.param.page.....	120
panl.param.numrows.....	121
panl.param.query.operand.....	122
panl.param.passthrough.....	122
panl.param.passthrough.canonical.....	123
panl.form.query.respondto.....	125
panl.include.single.facets.....	125
panl.include.same.number.facets.....	126
solr.default.query.operand.....	127
solr.facet.limit.....	128
solr.facet.min.count.....	128
solr.numrows.default.....	128
solr.highlight.....	128
Facet Field Definitions.....	128
Properties Available for Facets.....	129
Regular Facet.....	131
Example.....	133
Decoding the URI.....	134

BOOLEAN Facet.....	134
Additional Properties.....	135
Example.....	136
Decoding the URI Without a Prefix or Suffix.....	136
Defining a Prefix and Suffix.....	137
Decoding the URI With a Prefix or Suffix.....	137
OR Facet.....	138
Additional Properties.....	138
The Difference Between Multi-valued Facets and OR Facets.....	138
Example.....	142
Decoding the URI.....	144
RANGE Facet.....	145
Additional Properties.....	146
Visualising the Properties.....	148
Decoding the URI - Range facet with an infix.....	149
Decoding the URI - Range facet without an infix.....	149
DATE Range Facet.....	150
Additional Properties.....	153
Example.....	153
Decoding the URI.....	154
Some Notes.....	155
About Value Replacements.....	155
On Field Validations.....	157
About Hierarchical Facets.....	157
About Sorting Facets.....	158
Putting It All Together.....	160
A Brief Architecture.....	160
Generating SEO Friendly URLs.....	161
Data Dependencies.....	161
Passthrough URIs.....	161
Handling Errors.....	162
Panl LPSE URI Paths Explained.....	163
Parameter Query.....	164
Parameter Query Operand.....	165
Parameter Number of Rows.....	165
Parameter Page Number.....	167
Parameter Pass Through.....	168
Parameter Sort Order.....	169
Facets.....	170

Explaining A More Complex Example.....	172
<b>Integration And The Panl Response Object.....</b>	<b>174</b>
Error Responses.....	174
Success Responses.....	175
The "active" JSON Object.....	176
Overall Integration and Implementation.....	178
The "active.facet" JSON Array (Regular facet object).....	179
Integration and Implementation.....	180
The "active.facet" JSON Array (BOOLEAN facet object).....	180
Integration and Implementation.....	181
The "active.facet" JSON Array (DATE Range facet object).....	182
Integration and Implementation.....	183
The "active.facet" JSON Array (OR facet object).....	183
Integration and Implementation.....	184
The "active.facet" JSON Array (RANGE facet object).....	184
Integration and Implementation.....	185
The "active.numrows" JSON Object.....	186
Integration and Implementation.....	187
The "active.page" JSON Object.....	187
Integration and Implementation.....	188
The "active.query" JSON Object.....	188
Integration and Implementation.....	188
The "active.query" JSON Object.....	189
Integration and Implementation.....	189
The "active.sort" JSON Array.....	189
Integration and Implementation.....	190
The "available" JSON Object.....	191
The "available.date_range_facets" JSON Array.....	192
Integration and Implementation.....	193
The "available.facets" JSON Array.....	193
Integration and Implementation.....	194
The "available.range_facets" JSON Object.....	195
Implementation and Integration.....	195
The "sorting" JSON Object.....	200
Implementation and Integration.....	201
<b>Integrating An Existing Solr Schema.....</b>	<b>206</b>
Using the Panl Generator.....	206
Running the Command Line Utility.....	206
Editing the Generated Files.....	209

Manually Creating the Configuration Files.....	210
<b>Properties Quick Reference.....</b>	<b>212</b>
Property References Format Explained.....	212
panl.bool.<lpse_code>.false.....	213
panl.bool.<lpse_code>.true.....	214
panl.collection.<solr_collection_name>.....	215
panl.date.<lpse_code>.days.....	216
panl.date.<lpse_code>.hours.....	217
panl.date.<lpse_code>.months.....	218
panl.date.<lpse_code>.next.....	219
panl.date.<lpse_code>.previous.....	220
panl.date.<lpse_code>.years.....	221
panl.facet.<lpse_code>.....	222
panl.facetsort.<lpse_code>.....	223
panl.field.<lpse_code>.....	224
panl.form.query.respondto.....	225
panl.include.same.number.facets.....	225
panl.include.single.facets.....	226
panl.lpse.ignore.....	226
panl.lpse.length.....	227
panl.lpse.order.....	227
panl.name.<lpse_code>.....	228
panl.or.facet.<lpse_code>.....	229
panl.param.numrows.....	229
panl.param.numrows.prefix.....	230
panl.param.numrows.suffix.....	230
panl.param.page.....	231
panl.param.page.prefix.....	231
panl.param.page.suffix.....	232
panl.param.passthrough.....	232
panl.param.passthrough.canonical.....	233
panl.param.query.....	233
panl.param.query.operand.....	234
panl.param.sort.....	234
panl.prefix.<lpse_code>.....	235
panl.range.facet.<lpse_code>.....	235
panl.range.infix.<lpse_code>.....	236
panl.range.max.<lpse_code>.....	236
panl.range.max.value.<lpse_code>.....	237

panl.range.max.wildcard.<lpse_code>.....	238
panl.range.min.<lpse_code>.....	238
panl.range.min.value.<lpse_code>.....	239
panl.range.min.wildcard.<lpse_code>.....	240
panl.range.prefix.<lpse_code>.....	240
panl.range.suffix.<lpse_code>.....	241
panl.results.fields.<field_set>.....	242
panl.results.testing.urls.....	242
panl.results.sort.fields.....	243
panl.status.404.verbose.....	243
panl.status.500.verbose.....	243
panl.suffix.<lpse_code>.....	244
panl.type.<lpse_code>.....	244
panl.when.<lpse_code>.....	245
solr.default.query.operand.....	245
solr.facet.limit.....	246
solr.facet.min.count.....	246
solr.highlight.....	247
solr.numrows.default.....	247
solr.search.server.url.....	247
solrj.client.....	248
<b>Design Considerations.....</b>	<b>249</b>
Collections And FieldSet URI Paths (CaFUPs).....	250
In-Built Panl Results Testing URLs.....	251
Panl Server Startup.....	252
Error Messaging.....	253
404 - Not Found.....	253
500 - Internal Server Error.....	254
Testing vs. Production.....	255
Quick Middleware.....	256
<b>Epilogue.....</b>	<b>257</b>
On The Tag-Line.....	257
On Documentation.....	257
Additional Functionality in the Pipeline.....	259
<b>Appendices.....</b>	<b>262</b>
Definitions.....	262
Command Line Options.....	267
Panl Server.....	267
Panl Generator.....	267

Usage Text.....	268
Sample .properties Files.....	270
The Sample panl.properties Files.....	270
The Sample <panl_collection_uri>.panl.properties Files.....	271
Solr Version 8 & 7 Integration Notes.....	273
Setting up a Solr 7 or 8 server.....	274
Additional Solr Version 7 Integration Notes.....	279
Solr Versions 6 and below.....	279

# README.1ST

---

## Welcome To Synapticloop Panl

Synapticloop Panl is an application server that sits between your web application and Solr search server instance(s) seamlessly converting human-readable, SEO friendly URLs into complex Solr search queries, and returning an enhanced JSON object for ease of integration and implementation.

It abstracts away the complexities of the Solr search parameters and building/translating of URLs so that you get the benefit of a human readable (and SEO friendlier) URL without having to understand the mechanics behind it.

Some examples contained in this book also contain the conversions that Panl performs, should you wish to delve deeper into the inner-workings of the Panl server's integration with Solr.

### Additional Panl Niceties

1. **MULTIPLE ways to 'SLICE and DICE'** - From one Solr collection, the Panl server can present the results and facets in multiple different ways, providing individual use cases for specific needs.
2. **PREFIXES and SUFFIXES** - Panl can be configured to add prefixes and suffixes to the values within the URI path to increase readability, for example,

The example LPSE URI path of `/Caran+d'Ache/true/Black/bDW/` could also have the `brand` Solr field prefixed with 'Manufactured By ' and suffixed by ' Company' to produce the URI path

`/Manufactured+By+The+Caran+d'Ache+Company/true/Black/bDW/`

3. **BOOLEAN field translations**, for any Solr field that is defined as a `solr.BoolField`, then an additional translation can be performed. 'True' and 'false' values can be replaced with arbitrary text, which will be transparently converted between Panl and Solr.

For the LPSE URI path of `/Caran+d'Ache/true/Black/bDW/` the true value (which is defined as whether the mechanical pencil can be disassembled) could be changed

to ‘Able to be disassembled’ for true values, and ‘Cannot be disassembled’ for false values. The above URI path would then become

```
/Caran+d'Ache/Able+to+be+disassembled/Black/bDW/
```

4. **FIELD VALUE validation** - By default, Solr will error when an invalid value is passed through - for example, if Solr is expecting a numeric value and it could not parse the passed in value. Panl protects against this by attempting to parse the value as best it can, and silently dropping the parameter if it cannot be sensibly<sup>1</sup> parsed. This is only done for numeric types (integer, long, float, and double).
5. **HIERARCHICAL facets** - Only show facets if a parent facet is currently selected, allowing you to lead users through the search journey.
6. **SORTED facets** - Each individual facet can be sorted by either the facet count (which is the default), or the facet value (e.g. alphabetic/numeric).
7. **SORTING options** - sort by any of the Solr fields, either ascending, or descending and with multiple sub-sorting available - e.g. sorting by a brand name, then the model number.
8. **PAGINATION** - easy to generate pagination URI paths giving you control of how you would like to implement it.
9. **STATIC SITE GENERATION** - With the exception of a query parameter, all available links for every conceivable URI path can be statically generated ahead of time, with canonical URLs.

Be warned that the number of possible pages that can be generated can quickly become incredibly large.

10. **STATELESS** - No state is stored in the Panl server, all of the state is from the URI path part that is passed through. No sessions, no memory.

---

<sup>1</sup> ‘Sensibly’ is a bit of a vague term... Panl will strip out any unexpected characters and ensure that it is valid. For example, if Solr (and therefore) Panl is expecting an integer parameter and the value 5gs6 is passed through, Panl will remove any non-numeric characters and parse the number - returning 56.

## About This Book

This book explains the functionality of the Panl server and how to configure it using the two main java properties files and how this impacts the generated URI paths.

To start with, this book will take you through setting up and running a new Solr instance in cloud mode, creating a new collection, indexing the included sample data, and seeing the results and facets through the in-built Panl Results Viewer web application.

The book then continues on to explain the configuration and integration of the Panl server, with the assumption that there is already a running Solr instance behind it.

This book is not designed to be an introduction into Solr configuration, administration, or schema design best-practices, however there are some hints and tips throughout the book. These hints and tips relate to items that will affect the results that you retrieve from the Panl server, Solr configuration and the integration and implementation.

## Nomenclature Used Throughout This Book

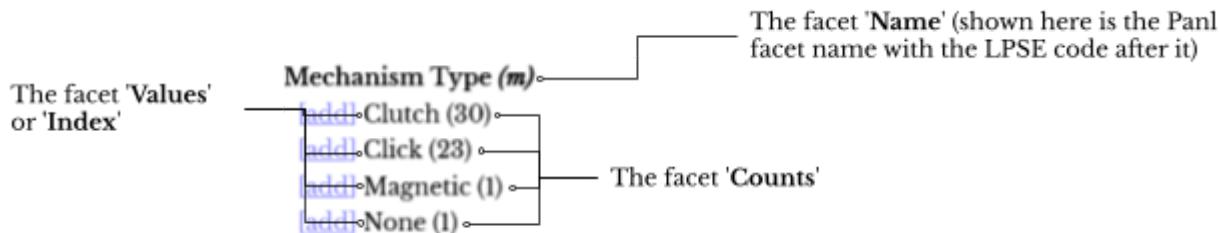
When implementing any faceted search interface to following three terms are the foundation of any search interface:

### Documents

*Solr nomenclature for the results that the Solr search server returns, these are a subset of the data that is indexed in the Solr search server. You can think of these as the rows of results that will be returned.*

### Facets

*Facets are specific categories/attributes/values that are extracted from the data and are attached to the index. Each of these attributes can then be used to filter the results such that only the documents that contain those attributes are returned. The image below shows the different parts of the facets.*



*Image: The parts of the returned Solr facet information*

In the above image:

- The Solr Facet name is `mechanism_type`, what is being rendered to the page is the Panl name - '`Mechanical Pencils`' , the `(m)` after the name is the Panl LPSE code
- The facet values represent the indexed data values that are attached to the document, they are:
  - `Clutch`
  - `Click`
  - `Magnetic`
  - `None`
- The facet counts represent the number of documents that contain this value, respectively they are:
  - `30`
  - `20`
  - `1`
  - `1`

### Search term

*The text (either a word or phrase) that is passed through from the web page through the Panl server to the Solr search server to search against the collections' indices. (also known as 'search query', 'keyword search', 'search phrase', or some other combination or name).*

Additional introductions to common words and phrases used throughout the book are below. Terms and names are defined where they first appear, for a full list - see the [Appendices - Definitions](#) at the end of the book.

### CaFUPs

*An acronym for Collection and FieldSets URI Paths which Panl uses to connect to the correct Solr collection of documents and return the specified fields.*

*CaFUPs allow you to configure multiple ways in which the search results and fields are returned for a Solr Collection.*

### **Collection(s)**

*Solr collections are an index of documents that can be filtered or searched upon. Panl collections are collections of URI paths and fieldsets*

### **LPSE code**

*The foundation of how the Panl server decodes and parses a URL to convert it to a form that the underlying Solr server can understand. A LPSE code is either a number, or an uppercase or lowercase letter of the alphabet (i.e. a-z, A-Z, 0-9).*

### **LPSE path**

*A LPSE path is a string of LPSE codes that is placed at the end of the URL which is used to decode the preceding path.*

### **Panl field**

*This is the field definition that contains the configuration of what parsing should be done on the incoming value. Additionally, it contains the configuration information as to how to pass this value through to the Solr search server.*

### **Panl generator**

*A stand-alone utility to quickly generate a `panl.properties` file and `<panl_collection_uri>.panl.properties` file from an existing Solr managed schema file that can be used as a starting point for configuring the Panl server.*

**NOTE:** *The generator does not interact or interfere with the Panl server and the generator codebase is not used when serving production content.*

### **Panl server**

*The server that handles the URLs, builds the Solr request object, connects to the Solr search server, executes the query, parses the results, builds the JSON response and passes it back to the caller.*

### **Solr field**

*The definition of the field in the `managed-schema.xml` Solr configuration file.*

### **Solr query**

*The query string that is sent to the Solr search server, an example of this is:*

```
q=*&fq.op=OR&facet.limit=100&fl=brand,name&facet.mincount=1&rows=10&facet.field=le  
ad_size_indicator&facet.field=colours&facet.field=brand&facet.field=mechanism_type&  
facet.field=id&facet.field=hardness_indicator&facet.field=lead_grade_indicator&face  
t.field=in_built_sharpener&facet.field=disassemble&facet.field=category&facet.field  
=lead_length&facet.field=in_built_eraser&facet.field=grip_shape&facet.field=weight&  
facet=true&fq=id:"53"&start=0
```

## Solr search server

*The Apache Solr search server that is queried for results.*

## Tokens

*The incoming LPSE code and any associated URI path values for each of the codes. Tokens will be parsed, prefixes and suffixes removed, and validation performed on the incoming value. If any parsing or validation fails, then the token will be marked as invalid, ignored, and not passed through to the Solr server.*

## Book Format Conventions

### Normal Text

Normal paragraph text is Libre Baskerville, 11pt, other formatting conventions are detailed below

### Footnotes<sup>2</sup>

Footnotes aren't used very often, but when they appear they can be safely ignored - these are more background thoughts as to why things were implemented the way they were. This will not impact the running of the Panl Server.

### Sidebars



**IMPORTANT:** Important notes are within a red side-bordered box, with an exclamation icon, and red background. Careful note should be made of the information contained within these boxes as this will affect the running of the Panl server, and may not be obvious.

---

<sup>2</sup> Thanks for checking out this footnote.



**Notes:** Notes are within a black side-bordered box, with a pencil icon, and grey background. This is something to look out for when you are reading the book, executing a task, or looking at an image or URL.



**Tips:** Notes are within a black bordered box, with a lightbulb icon, and white background. This is something which is a handy idea to know for the functioning or configuration of the Panl or Solr servers.

## Code Related Snippets

Inline code, or text related snippets are in monospaced text (`Inconsolata 11pt`), and highlighted in grey, for example: `/Caran+d'Ache/true/Black/bDW/`

This indicates that the text is exact and should be used as a reference.

For multi-line code related snippets they will be prefixed by a line number to be referenced with the text on a black-bordered grey box.

Note that within any line, there may be a line continuation character (`\`) which should not be included in the command. Unfortunately, for electronic viewers this means that it is a little more difficult to simply copy and paste the text - my apologies, however I chose readability and explanation of the text over cut-and-paste-ability.

```
01 # The text file that may be included, with some information or processing \
  directives which will not fit on one line, so it includes the continuation marker
02 # a line of commented text
03 # this is another line of text
```

## Commands

Any commands that should be run in your terminal or command line prompt will appear in a formatted table. Note the '`\`' character which means a line continuation and should not be included in the command. (As per above with the copy and paste-ability of the lines).

Command(s)
\the\command\that\needs\to\be\run -with "a parameter" -and-another ↵ "long parameter that may span many lines"

## Links

Links are designated by underlining the text of the link. If the text is underlined, it is either a link to another section of this book, or to an external website.

External links to websites (either local or remote) are in the standard blue underline and will **ALWAYS** match the URL that will open in your browser (i.e. The URLs are never truncated, even if they take multiple lines):

<http://localhost:8181/panl-results-viewer/mechanical-pencils/firstfive/Manufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/17+grams/bWGLw/>

Links within this documentation are bold, underlined, and in black text:

### **Integrating An Existing Solr Schema**

## About Panl Server

The Panl server is an interface into the Solr search server converting human-readable, SEO friendly URI paths into complex Solr search queries. Rather than adding query parameters to the URL, Panl automatically generates and returns complete URI path links that can be rendered by your web application.

The Panl server uses last path segment encoding (LPSE) to parse and decode the full URI path, converting a URI path from

```
/Manufactured+by+Koh-i-Noor+Company/Clutch/Green/bmWsb-sN-/
|-----LPSE PATH-----|LPSE code|
```

To a search query that will return a list of mechanical pencils that

- Are manufactured by Koh-i-Noor,
- Have a Clutch mechanism, and
- Are Green in colour

And, of the 8 results that are returned, the results will be sorted

- By brand name, then by
- Pencil name (i.e. model number)

Which is then passed through to the Solr search server as

```
q=*&*&q.op=OR&facet.limit=100&f1=brand.name&facet.mincount=1&rows=10&facet.field=lead_size
_indicator&facet.field=colours&facet.field=brand&facet.field=mechanism_type&facet.field=id
&facet.field=hardness_indicator&facet.field=lead_grade_indicator&facet.field=in_built_shar
pener&facet.field=disassemble&facet.field=category&facet.field=lead_length&facet.field=in_
built_eraser&facet.field=grip_shape&facet.field=weight&facet=true&fq=brand:"Koh-i-Noor"&fq
=mechanism_type:"Clutch"&fq=colours:"Green"&sort=brand+desc,name+desc&start=0
```

See the section on [Panl Server](#) command line options for full details on the options available.

## How Many Facets Does Panl Support?

The number of supported facets depends on the LPSE code length (which by default is 1). A LPSE code is a letter or number which maps to a parameter, operand, field, or filter. There are five mandatory (and one optional) LPSE codes:

1. The query parameter,
2. The page number,
3. The number of results per page,
4. The query operand,
5. The sort order, and
6. (*optionally*) The pass through parameter

The above configured LPSE codes cannot be registered as facet LPSE codes.

With a LPSE length of 1:

- With the five mandatory codes, Panl will support up to **57 facets**.
- With the five mandatory codes and the one optional code, Panl will support up to **56 facets**.

With a LPSE length of 2:

- With the five mandatory codes, Panl will support up to **3,249 facets**.

- With the five mandatory codes and the one optional code, Panl will support up to **3,186 facets**.

The formula for working out what the maximum number of supported facets for the LPSE code is the number of available LPSE codes to the power of the LPSE length:

- With the five mandatory codes<sup>3</sup>:

$$(62 - 5)^{\text{lpse\_length}} = 57^{\text{lpse\_length}}$$

- With the five mandatory codes and the one optional code<sup>4</sup>:

$$(62 - 6)^{\text{lpse\_length}} = 56^{\text{lpse\_length}}$$

A LPSE length of 2 should provide more than enough facets for the majority of implementations. Once the LPSE length gets above 2, the LPSE URI path becomes much longer, much more quickly, subtly negating the value of the encoding of the URI to be compact and readable.

Remember that you can define multiple Panl collections with CaFUPs for a Solr collection, and each of the CaFUPs can have different LPSE codes. You may have over 56 fields in your indexed Solr collection, but you may wish to have a LPSE length of 1 and just use a subset of the fields for each of the CaFUPs.

## In-Built Results Viewer / Explainer

For testing and debugging of the configured properties, Panl Results Viewer and Panl Results Explainer web apps are included in the panl release package. This surfaces all Panl functionality and allows integrators and implementers to understand and test the Panl configuration without having to integrate with a separate web application..



**Tips:** The recommendation is to either turn off the Panl Results Viewer / Explainer, or to not allow public access to these URLs.

## 'Simple' Panl Results Viewer Web App

What started as a relatively simple page for testing and debugging turned into a page that was a fully functional search page, able to highlight all of the functionality of the

---

<sup>3</sup> A LPSE length of 3 with the five mandatory codes would provide 185,193 facets, a length of 4 would provide 10,556,001

<sup>4</sup> A LPSE length of 3 with the five mandatory codes and one optional code would provide 175,616 facets, a length of 4 would provide 9,834,496

Panl server and surface most of the Solr search server functionality. It still remains an excellent way to test configuration options.

Below is a screenshot of the in-built Panl Results Viewer web app with all the features and functionality that you would expect from a search page implementation along with some additional features to make searching easier for you and your end-user.

When the Solr and Panl configuration is set up and enabled, it is accessible at:  
<http://localhost:8181/panyl-results-viewer/>

# Panl

*a very simple results viewer*

Available collections/fieldset URI Paths (CaFUPs):  
[\[/mechanical-pencils/default\]](#) [\[/mechanical-pencils/firstfive\]](#) [\[/mechanical-pencils/brandandname\]](#) 1

You are viewing collection/fieldset URI Path (CaFUP): /mechanical-pencils/brandandname 2

Canonical URI: /page-1/10-per-page/hexagonal/sb+pnq/ [\[explain\]](#) 3

**Search Query** 4

**Active Filters** 5

Query (*q*)  
[\[remove\]](#) hexagonal

Sorted by: Brand (*s*) [ASC]  
[\[Remove sort\]](#) [\[Change to DESC\]](#)

[\[Clear all sorting\]](#)

**Range Filters**

Weight (*w*): Range 6



from light to heavy pencils  
[\[Apply\]](#)

**Available Filters**

Brand (*b*):

- [\[add\]](#) Manufactured by Koh-i-Noor Company (9) 7
- [\[add\]](#) Manufactured by Caran d'Ache Company (4)
- [\[add\]](#) Manufactured by Faber-Castell Company (4)
- [\[add\]](#) Manufactured by Alvin Company (3)

**Search Results - Found 45 result(s) (exact)** 8

Page 1 of 5 Showing 10 results per page This page has result(s) 10

Sort by Brand: ASC [DESC](#) || Name: [ASC](#) [DESC](#)  
 Then sort by Name: [ASC](#) [DESC](#)

« PREV NEXT » 12

Solr: query time 95ms.  
 Panl: parse request 15ms, build request 36ms, send and receive request 145ms, parse response 1ms. Total time 198ms. 14

Show [3](#) [5](#) [10](#) per page 13

Name (name) Pro-Matic	15
Brand (brand) Alvin	
Name (name) Scott No. B/2	
Brand (brand) Alvin	
Name (name) Tech-DA	
Brand (brand) Alvin	
Name (name) Criterium	
Brand (brand) BIC	
Name (name) Fixpencil 22 (Smooth)	
Brand (brand) Caran d'Ache	

*Image:* The In-Build Panl Results Viewer web app

1. A list of available Collections and FieldSet URI Paths (CaFUPs) that Panl is configured to serve. CaFUPs enable different Solr fields, facets and documents to be returned from the same Solr collection.

2. A **textual representation of the CaFUPs** that the Panl Results Viewer web app currently is using.
3. **The canonical URI path** (which is returned with the Panl results JSON object) - An important part for search engines to de-duplicate URLs that return exactly the same information. Multiple Panl LPSE URI paths will return exactly the same results. You **SHOULD** use this link as either
  - The `rel="canonical"` link element in the HTML, or
  - The `rel="canonical"` link HTTP header

There is also an [\[explain\]](#) link that will take you to the Panl Results Explainer for this particular canonical URI.

4. **The search query box**, by default, Panl responds to the same parameter name as The Solr server - i.e. 'q'. This can be configured to be a different value through the Panl properties file.
5. **Active filters** - either queries, selected facets, or sorting options that are currently limiting the results - the [\[Remove\]](#) link is the URI path that will remove this query, facet, or sorting option from the results. If it is an active sorting filter, the [\[Change to DESC\]](#) or [\[Change to ASC\]](#) links will invert the sorting order without affecting any further sub-ordering.
6. **RANGE filters** - for facets that are defined as ranges - allowing end-users to select a range of values - the values are inclusive (i.e. include the minimum and maximum values).

**DATE Range filters (not shown<sup>5</sup>)** - Enabling searching on a range of dates (but not a specific date) in the form of:

`next/previous <any_integer> hours/days/months/years.`

For example:

`Last 30 days`  
`Previous 24 hours`  
`Next 30 days`

7. **Available filters** - additional facets that can further refine and limit the Solr search results.
8. **Number of results found**, and whether this is an exact match.

---

<sup>5</sup> Examples with specific dates are notoriously hard to put into examples as by the time you read this book, the example dates will be well out of range. There is an example data set (`simple-date`) which is included within the release package which has random dates spanning 2014 to 2032 which can be used to test out the features, however you will need to index the data set with separate commands.

9. **Query operand** - whether the Solr query is OR, or AND - this affects the search query, not the facetting - i.e. the Solr server `q.op` parameter.
10. **Page information**, the number of pages, how many results are shown per page, and how many results are shown on this page.
11. **Sorting options** - Whether to sort by relevance (the default) or by other configured sorting options with ascending and descending options available. Any Solr field can be configured to be used as a sorting option. And multi-sort orders are available, allowing sorting on more than one field.
12. **Pagination options** - the Panl server returns all information needed to build a pagination system, number of results, number of results shown per page and the current page number.
13. **Number of results per page**. Note: The values 3, 5, and 10 are just examples that are hard-coded into the viewer and can be implemented with any positive integer number.
14. **Timing information** about how long the Panl server took to build and return the results (including how much time the Solr server took to find and return the results).
15. **The results** - the fields that are returned with the documents and are shown in the results sections which are configured by the CaFUPs. Multiple field sets can be configured for the collection, allowing different groups of fields to be returned for different URI paths. In the image, only two fields are configured for this CaFUP, namely Brand, and Name.

## 'Simple' Panl Results Explainer

Again, this started as a relatively simple page for testing and debugging of the startup configuration options, rather than trawling through properties files and logs.

Below is a (cut-down) screenshot of the in-built Panl Results Explainer web app with explanations for canonical URI paths, the configuration of the Panl collection URI, and the individually configured properties for each of the fields and how this alters the Solr query. A useful page to see at-a-glance everything that a CaFUP is configured to do.

If enabled, available at <http://localhost:8181/panl-results-explainer/>

Panl  
a very simple results explainer

Available collections/fieldset URI Paths (CaFUPs):

- [/mechanical-pencils/default]
- [/mechanical-pencils/firstfive]
- [/mechanical-pencils/brandandname]

You are viewing collection/fieldset URI Path (CaFUP): /mechanical-pencils/brandandname

Enter the Canonical URI path below: 3

No URI path entered, nothing to explain.

## Request Token Explainer 4

[ 1 ] The length of the LPSE codes that are used for fields. The LPSE codes for Panl parameters is ALWAYS one (1).  
(set by the property 'panl.lpse.length')

... 5

## Configuration Parameters

[ 1 ] The length of the LPSE codes that are used for fields. The LPSE codes for Panl parameters is ALWAYS one (1).  
(set by the property 'panl.lpse.length')

... 5

## Field Configuration Explainer 6

In order of URI path

FIELD CONFIG [ PanlPassThroughField ] LPSE code 'z'.  
- This field is ignored by the Panl server and is not passed through to Solr query.

... 6

This page is designed to showcase the Panl LPSE encoding mechanism, with a minimal implementation. It is **NOT** a showcase of web technologies :)

Happy searching

*Image:* The In-Build Panl Results Viewer web app

1. A list of available Collections and FieldSet URI Paths (CaFUPs) that Panl is configured to serve. CaFUPs enable different Solr fields to be returned in the documents with the same search parameters.
2. A textual representation of the CaFUPs that the Panl Results Explainer web app is currently using.
3. The canonical URI path entry field allows you to enter any canonical URI path and have the parsing and tokenising explained to you, including whether the

parsed token was valid, the LPSE code found and the original value that Panl attempted to decode. **Note:** The CaFUP that the canonical URI path came from MUST match the CaFUP on the results viewer.

4. **The request token explainer** - for any canonical URI entered, this will list the parsing and decoding steps, with the following details
  - a. Whether the token is valid (if it is invalid, it will be ignored and not passed through to the Solr search server),
  - b. The type of token that was found,
  - c. The LPSE code,
  - d. The parsed value,
  - e. The original value, and
  - f. Where pertinent, additional information pertaining to the specific code.
5. **Configuration parameters** - parameters that are not fields or facets with information about the value, a description, and the property that set the value.
6. **Field configuration explainer** - for each of the fields or facets that are configured in the LPSE order an explanation of their configuration including:
  - a. The Java field type,
  - b. The LPSE code,
  - c. The Solr field name,
  - d. The Solr field type, the Panl field name, and
  - e. Additional configuration items which may include
    - i. Prefixes,
    - ii. Suffixes,
    - iii. Ranges,
    - iv. Facet type, or
    - v. Minimum/maximum values
  - f. Any configuration warning messages that were found whilst parsing the properties files.

## About Panl Generator

The Panl generator is a quick and interactive command line utility built into the Panl release package that, from a Solr managed schema file generates a default `panl.properties` and a `<panl_collection_uri>.panl.properties` files. This easily and quickly gets things up and running for your existing Solr schema from which you can iterate a solution from.

If you have an existing Solr schema and want to start testing the Panl server integration, then skip to the **Integrating An Existing Solr Schema** section. If you are skipping ahead and diving straight into the Panl configuration generator, the rest of the sections of the book will give understanding on how to configure the Panl server to suit the requirements of the search page implementation.

See the section on **Panl Generator** command line options for full details on the options available.

## About Apache Solr

From the Apache Solr website (<https://solr.apache.org/>)

---

Solr is the popular, blazing-fast, open source enterprise search platform built on Apache Lucene™

---

And:

---

Solr is highly reliable, scalable and fault tolerant, providing distributed indexing, replication and load-balanced querying, automated failover and recovery, centralized configuration and more. Solr powers the search and navigation features of many of the world's largest internet sites.

---

The Panl server abstracts away the complex Solr query options in both a developer and end-user friendly way, generating SEO friendlier URLs.

~ ~ ~ \* ~ ~ ~

## Why Synapticloop Panl?

---

Panl was designed to convert rather long and unfriendly (both in human readable and SEO terms) to shorter, nicer, and friendlier URI paths throughout the entire search journey.

Working with a Solr schema, the Panl configuration files translate unwieldy URI parameters into concise and precise URI paths.

Panl allows

1. Multiple ways to slice and dice the underlying Solr collection and present them with different URLs
2. Prefixes, and Suffixes for all facets, additional infixes for RANGE facets
3. Value replacements for Boolean data types
4. Field value validation to help avoid common errors
5. Hierarchical facets, only showing specific facets if another facet has already been selected
6. Sorting of facet values by either count (the default) or index.
7. Sorting by any of the fields or facets
8. Pagination - in built and ready to go
9. Static site generation - for small datasets with few options. Alternatively they can be seen as easily cacheable and canonical sources for the returned results.

## What is a LPSE (pronounced 'lapse') code?

The LPSE acronym stands for Last Path Segment Encoding, deriving its name from using the last part of the URI path as a code.

In effect the (29 characters) Panl LPSE query of

```
/Caran+d'Ache/true/Black/bDW/
```

For the above Panl URI path, the LPSE code is `bDW`, with the query of `/Caran+d'Ache/true/Black/`, this gets translated to

- The `brand` Solr facet field is set to `Caran d'Ache` (using LPSE code of `b`), and
- The `disassemble` Solr facet field is set to `true` (using LPSE code of `D`), and

- The `colours` Solr facet field is set to `Black` (using the LPSE code of `W`)

Contrast this with the equivalent<sup>6</sup> Solr query (828 characters) of

```
q=*&facet.mincount=1&rows=10&facet.field=lead_size_indicator&facet.field=grip_material&facet.field=colours&facet.field=nib_shape&facet.field=diameter&facet.field=cap_shape&facet.field=brand&facet.field=mechanism_type&facet.field=length&facet.field=hardness_indicator&facet.field=grip_type&facet.field=cap_material&facet.field=lead_grade_indicator&facet.field=tubing_material&facet.field=in_built_sharpener&facet.field=disassemble&facet.field=catalogue&facet.field=body_shape&facet.field=clip_material&facet.field=mechanism_material&facet.field=lead_length&facet.field=body_material&facet.field=in_built_eraser&facet.field=grip_shape&facet.field=relative_weight&facet.field=name&facet.field=nib_material&facet.field=weight&facet.field=variants&facet=true&fq=brand:"Caran+d'Ache"&fq=disassemble:"true"&fq=colours:"Black"&q.op=AND
```

## The three types of LPSE codes

1. **Filters** - codes that filter (and generally narrow<sup>7</sup>) the results - these are either the search term or a facet (Regular, Boolean, OR, DATE, or RANGE)
2. **Parameters** - codes that change the way the results are returned - these are the sorting orders, number of results per page, and page numbering. These do not change what results are returned.
3. **Operands** - codes that change the way in which the filters work, for Panl the only operand available is the Solr query operand.

## URI Path Nomenclature

As the design considerations were made mainly around URLs and the niceties that could be afforded to them, here is the nomenclature used throughout this book. For any URI path:

1. The the Web Server URI path, e.g.

```
/products-by-brands/Caran+d'Ache/b/
```

---

<sup>6</sup> This is probably not the fairest of comparisons, as a lot of the underlying Solr query implementation could be hidden behind the scenes anyhow. However, what Panl can do is automatically have CaFUPs for multiple fieldsets, facets, and queries which will automatically build the query, the returned facets, the fields, and more.

<sup>7</sup> The exception to this rule are any defined OR facets, which will increase the number of results that are returned.

2. Is then proxied through to the desired Panl CaFUPs URI path  
`/mechanical-pencils/brand-fields/Caran+d'Ache/b/`
3. Which will return the results from the `mechanical-pencils` Solr collections, only returning the fields in each document that are contained in the configuration of the `brand-fields`.

The Panl URI path is made up of the following parts

- `/mechanical-pencils/brand-fields/` This is the Collection and FieldSet URI Path (CaFUPs) the collection is `mechanical-pencils`, and the FieldSet is `brand-field`
- `/Caran+d'Ache/` This is the LPSE value part of the URI path
- `/b/` This is the LPSE code part of the URI path

When references are made throughout this book, they may be broken into the collection, FieldSet, LPSE value, and LPSE code.

## Is Synapticloop Panl For Me?

The tagline for Panl is "a rather pleasing companion to the Apache Solr Faceted Search Engine", and that is how it was designed and coded. Pleasing, as it is designed to be developer implementation friendly, makes Solr nicer to work with, and the URI paths much more friendly to read (both for humans and search engines).

However, you may have a different point of view, so, before diving into it, you should evaluate whether this project is suitable for your needs. Panl is constrained by the following:

- You **MUST** be using a managed schema for Solr, Panl will **NOT** work with non-managed, or derived schemas (In fact Solr requires a managed schema for production deployments, so this will only affect the initial testing phase of your project).
- You **MUST** be using a Java version of at least 11.
- Panl **DOES NOT** include all of the options that are available to the Solr query parser, however it does cover all functionality required for a pleasant and powerful interface into the Solr search features.
- Panl **WILL NOT ALLOW** complex Solr query parameters (the Solr query parameter is passed through enclosed in double-quotes `"`), so if you need to do complex hand-crafted queries, these are not available, for example `"jakarta`

`apache"^-4 "Apache Lucene"`, or a query that relies on using any of the characters with a special meaning (+ - && || ! ( ) { } [ ] ^ " ~ \* ? : /).

- Panl IS NOT REALLY designed to be publicly accessible, instead it is designed to sit behind a proxy or service forwarder, so there is a requirement to be able to **configure a proxy or service forwarder**.
- Panl IS PERMISSIVE - there is no authentication, or access control lists. If you don't want an end-user to be able to query the Panl server, or restrict access to a certain subset of results, then you will need to implement your own authentication, authorisation and security layer (which comes with its own challenges).

~ ~ ~ \* ~ ~ ~

# What You Will Need

---

## Download The Resources

### 1. Java Runtime Environment

Download your favourite JRE of at least version 11. There are multiple locations to download the JRE and is left up to the reader to choose the implementation that you are comfortable with.

### 2. An Apache Solr instance

<https://solr.apache.org/downloads.html>

Ensure that you have downloaded and unzipped/un-tgzed the Apache Solr binary instance - in this book the referenced version is 9.6.0, however later versions should work especially with any 9.x.x Solr release. The version that is used with this book is the slim version - i.e. the binary release: `solr-9.6.0-slim.tgz`.

The location that you have unzipped/un-tgzed the Apache Solr package file is referred to as the `SOLR_INSTALL_DIRECTORY` and will be referenced throughout this book.

### 3. The Panl Server package

<https://github.com/synapticloop/panl/releases>

Download the Panl Server release package from github and unzip/un-tgz the file. The version that is used in this book is `solr-panl-9-1.0.0`, make sure that you download the binary package, not the source package

The location that you have unzipped/un-tgzed the Panl Server package file is referred to as the `PANL_INSTALL_DIRECTORY`, and will be referenced throughout this book.

## The Panl Directory Structure

The release package has the following directory structure, note that both the `sample` and `book` directories are not required (and they are not used) to run the panl server in production and may be safely deleted.

File Path	Explanation
/solr-panl-9-1.0.0/	The root install directory
bin/	The binary directories for starting the server
panl	The *NIX executable
panl.bat	The Windows executable
lib/	The library directory containing java archive library files to run the Panl server
book/	Where you will find this book - you may safely delete this directory (after reading and saving a copy of the book for safekeeping of course)
sample/	The directory containing sample data and configuration for the Panl server - you may delete this directory
data/	Directory for testing/example data
mechanical-pencils.json	Mechanical pencils example data
simple-date.json	Simple Date example data
book-store.json	Book Store example data
panl/	Directory for testing/sample Panl configuration
all/	The directory that contains the Panl configuration for all Solr collections and Panl collection URI bindings for testing across all CaFUPs.
mechanical-pencils/	<b>NOTE:</b> This includes configurations for the mechanical-pencils, simple-date, and book-store Solr collections.
simple-date/	The directory that contains the Panl configuration for the Simple Date Solr Panl collection URI bindings
book-store/	<b>NOTE:</b> To run examples in this book you <b>MUST</b> create and populate the Solr mechanical-pencils collection
mechanical-pencils-or/	The directory that contains the Panl configuration for the Mechanical Pencils Panl collection URI bindings with the brand Solr field configured to be an OR Facet
	The directory that contains the Panl configuration for the Simple Date Solr Panl collection URI bindings
	The directory that contains the Panl configuration for the Book Store Panl collection URI bindings.
	<b>NOTE:</b> Only brief sample data is included for this collection

<code>solr/</code>	Directory for testing/sample Solr configuration
<code>mechanical-pencils/</code>	The directory for the Solr configuration XML file and managed schema file for the Mechanical Pencils collection
<code>simple-date/</code>	The directory for the Solr configuration XML file and managed schema file for the Simple Date collection
<code>book-store/</code>	The directory for the Solr configuration XML file and managed schema file for the Book Store collection.
<b>NOTE:</b> Only brief sample data is included for this collection	

~ ~ ~ \* ~ ~ ~

# Panl Server Versions

---

Versions of the release package follow the Solr versions with the release versions that match the Solr version.



**IMPORTANT:**This book was written for the integration between Solr 9 and Panl 9.

For other versions - please refer to the Appendices .

If there are other integrations available with previous versions of Solr, then the Panl name will be suffixed with the major version number of the Solr release - i.e. `solr-panl-8` is the release for Solr version 8.x.x, compiled against the Solr binaries for version 8.

## Solr-Panl-9-x.x.x

The `solr-panl-9` designator is designed to integrate with Solr version 9.x.x.

This is the default integration and server available.

## Solr-Panl-8-x.x.x & Solr-Panl-7-x.x.x

The `solr-panl-8` designator is designed to integrate with Solr version 8.x.x.

The `solr-panl-7` designator is designed to integrate with Solr version 7.x.x.

There are differences in the available SolrJ connectors and the format for the JSON results returned between the various versions of Solr.

## Solr-Panl-6-x.x.x and earlier version

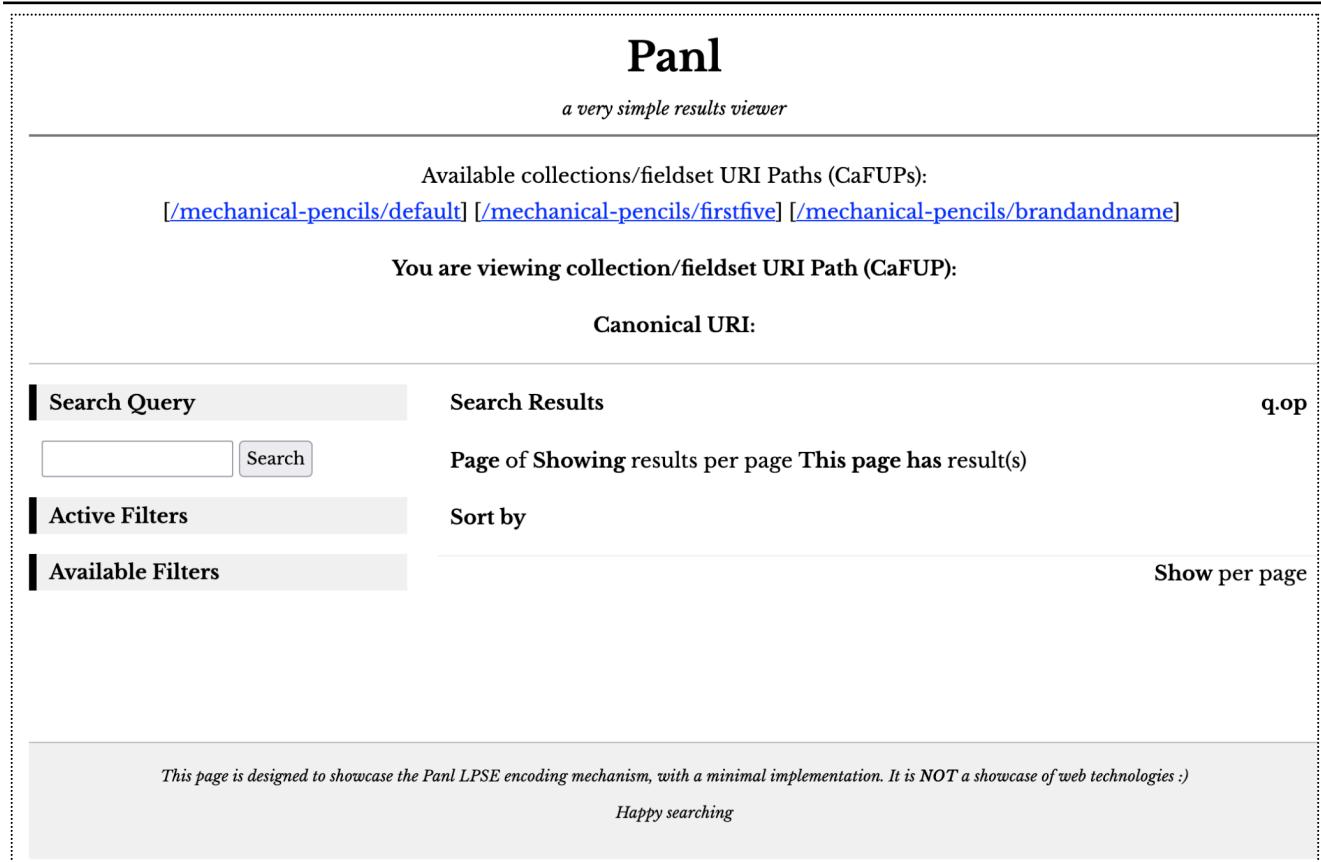
There are no pre-built integrations available for Solr version 6 and earlier, there are some hints and tips in the Appendices section, see the [Solr Versions 6 and below](#) section for more information

~ ~ ~ \* ~ ~ ~

# Quick Start - The 5 Steps

---

At the end of this chapter, you will have a web app up and running with the mechanical-pencils collection indexed and ready to search, sort, and facet on the URL:  
<http://localhost:8181/panl-results-viewer/>



*Image:* The In-Build Panl Results Viewer web app

---



**IMPORTANT:** Throughout the book the filesystem paths are described using the \*NIX nomenclature of a forward slash '/' between the directories, rather than the backslash of Windows systems '\'. So please take care when copying the commands.

## Step 0. Download Solr and Panl

Download the latest release of Synapticloop Panl - this book is using the `solr-panl-9.1.0.0` version:

<https://github.com/synapticloop/panl/releases>

Download the latest version of Apache Solr - this book is using the `9.6.0-slim` version:

<https://solr.apache.org/downloads.html>

## A Note On Running The Commands

These are all of the commands for either Microsoft Windows or \*NIX operating systems (Linux/Apple Macintosh). Should there be any errors - see the '[Getting Started](#)' section for a more in-depth explanation and approach.



**IMPORTANT:** You will need to replace the

`SOLR_INSTALL_DIRECTORY`

and

`PANL_INSTALL_DIRECTORY`

references in the commands for your particular setup.

## Windows Commands



**IMPORTANT:** Each of the commands - either Windows or \*NIX must be run on a single line - watch out for ↲ continuations.

### Step 1. Create an example cloud instance

This requires no interaction, will use the default setup, two replicas, and two shards under the 'example' cloud node.

Command(s)
<code>cd SOLR_INSTALL_DIRECTORY</code>
<code>bin\solr start -e cloud -noprompt</code>

### Step 2. Create the mechanical pencils collection

This will create and set up the mechanical pencil collection and schema so that the data can be indexed.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin\solr create -c mechanical-pencils -d PANL_INSTALL_DIRECTORY\sample\solr\mechanical-pencils\ -s 2 -rf 2

### Step 3. Index the mechanical pencils data

This will index the included sample mechanical pencil data into the Solr instance in the mechanical-pencils collection.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin\solr post -c mechanical-pencils < PANL_INSTALL_DIRECTORY\sample\data\mechanical-pencils\pencils.json

### Step 4. Start the Panl Server

This will start the server and be ready to accept requests.

Command(s)
cd PANL_INSTALL_DIRECTORY
bin\panl.bat -properties < PANL_INSTALL_DIRECTORY\sample\panl\mechanical-properties\panl.properties

### Step 5. Browse the in-built Panl Results Viewer web app

Open the link <http://localhost:8181/panl-results-viewer/> in your favourite browser, choose a collection/fieldset and search, facet, sort, paginate and view the results

Need help understanding what the Panl server is doing?

Open the link <http://localhost:8181/paml-results-explainer/> in your favourite browser, choose a collection and fieldset, paste the canonical URI path from the results viewer and an explanation of the parsing and decoding and the configuration will be presented.

## \*NIX Commands



**IMPORTANT:** Each of the commands - either Windows or \*NIX must be run on a single line - watch out for ↲ continuations.

### Step 1. Create an example cloud instance

This requires no interaction, will use the default setup, two replicas, and two shards under the 'example' cloud node.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr start -e cloud -noprompt

### Step 2. Create the mechanical pencils collection

This will create and set up the mechanical pencil collection and schema so that the data can be indexed.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr create -c mechanical-pencils -d ↲ PAML_INSTALL_DIRECTORY/sample/solr/mechanical-pencils/ -s 2 -rf 2

### Step 3. Index the mechanical pencils data

This will index the included sample mechanical pencil data into the Solr instance in the `mechanical-pencils` collection.

Command(s)
cd SOLR_INSTALL_DIRECTORY

**Command(s)**

```
bin/solr post -c mechanical-pencils <
PANL_INSTALL_DIRECTORY/sample/data/mechanical-pencils/mechanical-pencils.json
```

**Step 4. Start the Panl Server**

This will start the server and be ready to accept requests.

**Command(s)**

```
cd PANL_INSTALL_DIRECTORY

bin/panl -properties <
PANL_INSTALL_DIRECTORY/sample/panl/mechanical-properties/panl.properties
```

**Step 5. Browse the in-built Panl Results Viewer web app**

Open the link <http://localhost:8181/panl-results-viewer/> in your favourite browser, choose a collection/fieldset and search, facet, sort, paginate and view the results

**Need help understanding what the Panl server is doing?**

Open the link <http://localhost:8181/panl-results-explainer/> in your favourite browser, choose a collection and fieldset, paste the canonical URI path from the results viewer and an explanation of the parsing and decoding and the configuration will be presented.

**Something Went Wrong?**

Work your way through the next section, or have a look at the **When Something Goes Wrong** section.

~ ~ ~ \* ~ ~ ~

# Getting Started

---



**Notes:** If you have already got everything up and running from the [Quick Start - The 5 Steps](#) section, then you can skip this section as this expands on that section and goes into finer-grained detail with deeper explanations.

If something is not working, see the [When Something Goes Wrong](#) section.

For this book we are going to be looking at a faceted search engine based on mechanical pencils - in particular 2mm clutch type mechanical pencils.<sup>8</sup> The data used is included with the Panl release package and offers a good learning opportunity to understand

1. How to set up a Solr search server,
2. How to index data,
3. How to run a Panl server, and
4. What features are available through configuring the Panl server.

## Downloading the Resources

Apart from a Java version 11 or greater, you will need the following resources

1. Download the latest release of Synapticloop Panl - this book is using the `solr-panl-9.1.0.0` version:  
<https://github.com/synapticloop/panl/releases/>
2. Download the latest version of Apache Solr - this book is using the `9.6.0-slim` version:  
<https://solr.apache.org/downloads.html>

Now unzip/untar/untgz the resources into an installation directory. For my installation, I used the following:

---

<sup>8</sup> The example data ‘techproducts’ included with the Apache Solr instance is a reasonable test dataset, however, the way the schema and collections are designed places an emphasis more on testing ingestion and searching, rather than on a functional search set.

## The Solr server

On a Windows machine the installation directory was:

```
C:\java-servers\solr-9.6.0-slim\
```

whilst on a \*NIX the installation directory was:

```
/Users/synapticloop/java-servers/solr-9.6.0-slim/
```

Whichever operating system that you are using, throughout this book, this directory is referred to as the `SOLR_INSTALL_DIRECTORY`.

## The Panl server

On a Windows machine the installation directory was:

```
C:\java-servers\solr-panl-9-1.0.0\
```

whilst on a \*NIX<sup>9</sup> the installation directory was:

```
/Users/synapticloop/java-servers/solr-panl-9-1.0.0/
```

Whichever operating system that you are using, throughout this book, this directory is referred to as the `PANL_INSTALL_DIRECTORY`.

## Creating and Starting a Solr Cloud Instance

Run the following command which will start up Solr in cloud mode with default values i.e. two replicas and two shards per replica, with a default collection of ‘gettingstarted’

**Windows:**

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin\solr start -e cloud -noprompt

For example, the above command with the Windows installation directory replacement would translate to:

```
cd C:\java-servers\solr-9.6.0-slim\
bin\start -e cloud -noprompt
```

---

<sup>9</sup> I am using an Apple Mac system, but it is the same for Linux

## \*NIX:

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY  bin/solr start -e cloud -noprompt</pre>

For example, the above command with the \*NIX directory replacement would translate to:

```
cd /Users/synapticloop/java-servers/solr-9.6.0-slim
bin/start -e cloud -noprompt
```

This will start Solr in cloud mode with all of the defaults - you can check to see that it is up and running by going to the cloud admin console:

<http://localhost:8983/solr/#/~cloud>,

and you should be presented with a page similar to the following:

The screenshot shows the Solr Cloud Admin interface. On the left is a sidebar with various navigation options: Dashboard, Logging, Security, Cloud (selected), Nodes, Tree, ZK Status, Graph, Schema Designer, Collections, Java Properties, Thread Dump, Collection Selector, and Core Selector. The main content area displays a table of hosts. The table has columns for Host, Node, CPU, Heap, Disk usage, Requests, Collections, and Replicas. There are two rows in the table:

Host	Node	CPU	Heap	Disk usage	Requests	Collections	Replicas
<b>localhost</b> Windows 10 7.9Gb Java 17 Load: -1 <a href="#">show details...</a>	<b>7574_solr</b> Uptime: 0m <a href="#">show details...</a>	<b>8%</b>	<b>48%</b>	<b>138.0b</b>	RPM: 0.78 p95: 5172ms	gettingstarted	gettingstarted_s1r4 (0 docs) gettingstarted_s2r2 (0 docs)
	<b>8983_solr</b> Uptime: 0m <a href="#">show details...</a>	<b>18%</b>	<b>18%</b>	<b>138.0b</b>	RPM: 0.78 p95: 221ms	gettingstarted	<b>gettingstarted_s1r6</b> (0 docs) <b>gettingstarted_s2r1</b> (0 docs)

At the bottom of the page are links to Documentation, Solr Query Syntax, Community, Issue Tracker, Slack, and IRC.

*Image: The Solr Cloud Admin page*



**Notes:** In the above image, under the ‘Collections’ column in the table, you will see the value ‘gettingstarted’ - this is the default collection that is set up and can be safely ignored.

The above shows the two cloud Solr nodes up and running - ignore the values for the moment - now it is time to create the mechanical pencil collection and index the data.



**Notes:** The Solr admin pages off a lot of information for troubleshooting collections and indexes. It is worth your while getting to understand the functionality behind it, however it is beyond the scope of this book.

## Creating a collection

To create a collection, we will be using the schema and data included in the Panl Server Package, which resides in the `PANL_INSTALL_DIRECTORY/solr/mechanical-pencils/` directory.

Run the following commands to create a new collection named mechanical-pencils

**Windows:**

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY  bin\solr create -c mechanical-pencils -d &lt; PANL_INSTALL_DIRECTORY\sample\solr\mechanical-pencils\ -s 2 -rf 2</pre>

**\*NIX:**

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY  bin/solr create -c mechanical-pencils &lt; -d PANL_INSTALL_DIRECTORY/sample/solr/mechanical-pencils/ -s 2 -rf 2</pre>

This will create a collection called mechanical-pencils. You can confirm the creation of the new collection on the cloud admin console: <http://localhost:8983/solr/#/~cloud>, and you should be presented with a page similar to the following:

Host	Node	CPU	Heap	Disk usage	Requests	Collections	Replicas
<b>localhost</b> Windows 10 7.9Gb Java 17 Load: -1 <a href="#">show details...</a>	<b>7574_solr</b> Uptime: 1m <a href="#">show details...</a>	<b>1%</b>	<b>18%</b>	<b>276.0b</b>	RPM: 0.72 p95: 5172ms	gettingstarted mechanical-pencils	gettingstarted_s1r4 (0 docs) gettingstarted_s2r2 (0 docs) (2 more...)
	<b>8983_solr</b> Uptime: 2m <a href="#">show details...</a>	<b>9%</b>	<b>41%</b>	<b>276.0b</b>	RPM: 0.72 p95: 162ms	gettingstarted mechanical-pencils	<b>gettingstarted_s1r6</b> (0 docs) <b>gettingstarted_s2r1</b> (0 docs) (2 more...)

*Image: The Solr Cloud Admin page*



**Notes:** In the above image, two collections are now shown, 'gettingstarted' and 'mechanical-pencils', you can safely ignore the 'gettingstarted' collection.

## Indexing the Data

Now that the collection has been created, it is time to index the data. Data for the mechanical-pencils collection is included in the Panl Server package in the `PANL_INSTALL_DIRECTORY/sample/data/` directory.

To index the data:

**Windows:**

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

bin\solr post -c mechanical-pencils <
PANL_INSTALL_DIRECTORY\sample\data\mechanical-pencils.json
```

**\*NIX:****Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

bin/solr post -c mechanical-pencils <
PANL_INSTALL_DIRECTORY/sample/data/mechanical-pencils.json
```

## Starting The Panl Server

**Windows:****Command(s)**

```
cd PANL_INSTALL_DIRECTORY

bin\panl.bat -properties <
PANL_INSTALL_DIRECTORY\sample\panl\mechanical-properties\panl.properties
```

**\*NIX:****Command(s)**

```
cd PANL_INSTALL_DIRECTORY

bin/panl -properties <
PANL_INSTALL_DIRECTORY/sample/panl/mechanical-properties/panl.properties
```

<http://localhost:8181/panl-results-viewer/>

You will be presented with the following page:

Panl  
a very simple results viewer

Available collections/fieldset URI Paths (CaFUPs):  
[\[/mechanical-pencils/default\]](#) [\[/mechanical-pencils/firstfive\]](#) [\[/mechanical-pencils/brandandname\]](#)

You are viewing collection/fieldset URI Path (CaFUP):

Canonical URI:

<b>Search Query</b>	<b>Search Results</b>	<b>q.op</b>
<input type="text"/> <input type="button" value="Search"/>	Page of Showing results per page This page has result(s)	
<b>Active Filters</b>	Sort by	
<b>Available Filters</b>		Show per page

This page is designed to showcase the Panl LPSE encoding mechanism, with a minimal implementation. It is NOT a showcase of web technologies :)  
Happy searching

**Image:** The In-Built Panl Results Viewer web app showing the list of available collection/fieldset configurations

Select one of the collection/fieldset URLs - for example

<http://localhost:8181/panl-results-viewer/mechanical-pencils/firstfive>

and start searching, sorting, facetting and sorting.

## When Something Goes Wrong



**Tips:** Ensure that you have the latest version of this book. Whilst the commands shouldn't change, they might...

Alas, not everything always goes according to plan, when attempting to troubleshoot the instructions, the base advice is to start with a separate, clean Solr installation (as opposed to using an existing Solr installation) so that there are no errant configuration problems.

This will ensure that:

- a. All default ports are available for both the Solr (ports 8983, 7574, 9983) and Panl (port 8181) servers - which are referenced throughout this book, and
- b. No conflicting configuration is used from previous installations

## Restarting the Solr Panl tutorial process

### 1. Stop all running Solr instances

See the ‘Stopping Solr’ section below, make sure that you stop all instances, whether it is for this process, or any other pre-existing installations that you may have

### 2. Create separate Solr and Panl installations

Re-download the installation files if necessary and unzip/un-tgz to a new, clean installation directory

### 3. Re-run the tutorial, ensuring that you are using the correct values for

`SOLR_INSTALL_DIRECTORY`, and

`PANL_INSTALL_DIRECTORY`

## Stopping Solr

### Windows:

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY  bin\solr stop -all</pre>

### \*NIX:

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY  bin/solr stop -all</pre>

This will stop all instances of the Solr servers that are running in cloud mode.



**IMPORTANT:** If you have multiple Solr installations, you will need to make

sure that you run the stop command from the Solr installation directory that it was started from.

## Deleting a collection

**Windows:**

Command(s)
cd SOLR_INSTALL_DIRECTORY  bin\solr delete -c mechanical-pencils

**\*NIX:**

Command(s)
cd SOLR_INSTALL_DIRECTORY  bin/solr delete -c mechanical-pencils

You will then need to re-create the collection, and re-index the data.

## Re-Starting Solr

**Windows:**

Command(s)
cd SOLR_INSTALL_DIRECTORY  bin\solr start -cloud -p 8983 -s "example\cloud\node1\solr"  bin\solr start -cloud -p 7574 -s "example\cloud\node2\solr" -z localhost:9983

**\*NIX:**

Command(s)
cd SOLR_INSTALL_DIRECTORY  bin/solr start -cloud -p 8983 -s "example/cloud/node1/solr"

```
bin/solr start -cloud -p 7574 -s "example/cloud/node2/solr" -z localhost:9983
```

## Worst-case scenario

In the worst case scenario - you may have to look at the log files which can be found in the following directories:

- `SOLR_INSTALL_DIRECTORY/example/cloud/node1/logs`, and
- `SOLR_INSTALL_DIRECTORY/example/cloud/node2/logs`

If nothing else seems to work, delete the solr collection.

1. Ensure that Solr is stopped,
2. Delete the `SOLR_INSTALL_DIRECTORY/example/cloud/node1`, and `SOLR_INSTALL_DIRECTORY/example/cloud/node2` directories (including all of their contents<sup>10</sup>).
3. Re-create the Solr cloud instance,
4. Re-create the collection(s), and
5. Re-index the data

~ ~ ~ \* ~ ~ ~

---

<sup>10</sup> Commands weren't included in this as recursive force (i.e. `rm -rf` or `rmdir /S /Q`) deletion of directories can be a very dangerous thing.

# Additional Data

---

In some instances, this book also references additional datasets which the [Quick Start - The 5 Steps](#), and the [Getting Started](#) sections do not include instructions for setting up. Some references are made to the additional data whilst working through this book, whilst not necessary to have the datasets indeed, they showcase additional functionality which may be of use.

The steps are the same for any data set once the Solr search server has been set up, quick instructions are included for your reference.

## All Data Panl Server

There is an additional Panl configuration file located at `sample/panl/all/panl.properties` which, once all additional data has been indexed, will provide access to all CaFUPs. **Remember:** Panl was designed to have multiple CaFUPs connecting to multiple Solr collections.

## Simple Date

The `simple-date` sample collection includes randomly generated dates from NOW to approximately +/- 10 years from the date of writing this book to test the DATE range functionality and faceting.

The Simple Data dataset provides additional examples for the following Panl configuration functionality:

1. Hierarchical facet selection
2. DATE range facets

## Creating and Indexing the Data

If you come across any problems, the best thing to do is to reset the entire installation and start again, however this is generally not needed.

## Windows Commands

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

bin/solr create -c simple-date -d ←
PANL_INSTALL_DIRECTORY/sample/solr/simple-date/ -s 2 -rf 2

bin/solr post -c simple-date ←
PANL_INSTALL_DIRECTORY/sample/data/simple-date/simple-date.json
```

**\*NIX Commands****Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

bin\solr create -c simple-date -d ←
PANL_INSTALL_DIRECTORY\sample\solr\simple-date\ -s 2 -rf 2

bin\solr post -c simple-date ←
PANL_INSTALL_DIRECTORY\sample\data\simple-date\simple-date.json
```

**panl.properties File Additions**

You will need to include (or update) the following line in the `panl.properties` file.

```
panl.collection.simple-date=PANL_INSTALL_DIRECTORY/sample/panl/simple-date/simple-date.pan
l.properties
```

Panl will automatically add the new CaFUPS to the Panl Results Viewer and Panl Results Explainer web apps.

**Mechanical Pencils OR Facet**

The `mechanical-pencils-or` sample collection includes configuration that configures the brand facet as an OR facet so that more than one brand may be selected for the facet.

## Creating and Indexing the Data

This is a Panl configuration change, not a Solr configuration change and connects to the existing `mechanical-pencils` Solr collection and data which has been setup.

Nothing needs to be done for the Solr server configuration as it already holds the collection. This is just a simple example of how Panl can present different faceting information without having to re-index the data.

### panl.properties File Additions

You will need to include (or update) the following line in the `panl.properties` file.

```
panl.collection.mechanical-pencils=PANL_INSTALL_DIRECTORY/sample/panl/mechanical-pencils-or/mechanical-pencils-or.panl.properties
```



**Notes:** You may want to run both the configurations at the same time to see the differences, in which case the above configuration line would become:

```
panl.collection.mechanical-pencils=PANL_INSTALL_DIRECTORY/sample/panl/mechanical-pencils/mechanical-pencils.panl.properties,PANL_INSTALL_DIRECTORY/sample/panl/mechanical-pencils-or/mechanical-pencils-or.panl.properties
```

Panl will automatically add the new CaFUPS to the Panl Results Viewer and Panl Results Explainer web apps.

## Book Store

To help understanding, and to give a bit of practice, the adding and indexing of this dataset is left as an exercise for the reader. The next chapter will go through the complete process from idea, to data requirements, to indexing, and finally surfacing the data through Panl.



**Tips:** Work through the next section before indexing the data. The process and the background to the data provides insight into the decisions made with the data and will help you make better decisions when working through your own datasets.

~ ~ ~ \* ~ ~ ~

# A Working Example - The Book Store

---

At this point, you should

1. Have a functional Solr server with indexed data.
2. Have a brief understanding how to add data to the Solr server (the [Additional Data](#) section is a good place to start).
3. Have a Panl server instance available and have looked at the functionality provided.
4. Also have a good idea as to how Panl interacts with Solr and the options that are available.

Working through this chapter of the book will provide all of the thinking, design, and steps required to index the required dataset, configure the Panl server, and get it all up and running

This section runs through the setup and configuration of Panl and Solr for a distinct dataset showing the path from defining and setting up a Solr collection, to configuring the Panl server, to rendering the pages that are required for the user journey. This dataset will be based on a book store<sup>11</sup> with the underlying data containing 3 million records - this dataset is not included with the release packet (however a cut-down, cleaned version is included for testing). Running through this example will detail the decisions that are required and made and how the data was then formulated.

The process for pulling all of the components together is going to be the same regardless of the dataset that you are using.

1. Understand the dataset - *Knowing what data is available, what cleanup process may be required, what data is missing and whether data should be derived or included from a separate source.*
2. Configure the Solr index - *Knowing how the underlying data will be indexed and surfaced to the Panl server. In general one Solr collection, if set up correctly, will be able to drive multiple Panl configurations.*
3. Configure the Panl server - *This applies to both the panl.properties file and <panl\_collection\_uri>.panl.properties files. Remember: you may have multiple CaFUPs,*

---

<sup>11</sup> Historically, Java based examples for servers seem to have been based on the ubiquitous Pet Store, time for something new.

*so there may be multiple configuration files. Panl allows you to 'slice and dice' the underlying Solr search for specific use cases.*

Finally, using all of the above information, and the functionality that each decision will provide:

4. Determine any additional web pages to render - *Apart from the default (or main) search page which is the first page to be configured.*

This, most likely, will be an iterative process, with any of the steps requiring updates to the other steps and configuration. Having a good knowledge of the high-level requirements will enable you to quickly build up the configuration files for both the Solr and Panl servers.

For example when looking at the dataset, there are over 500,000 authors and I wanted to be able to have a page for each letter of the alphabet that would list the authors by their surname. The dataset doesn't directly support this, so derived fields will need to be added (i.e. a field that is computed/derived from the dataset) which can then be indexed by the Solr server.

## 0. High Level Requirements

From a very high level perspective

- For this particular implementation, the keyword search should act upon the
  - Title,
  - Description,
  - Author, and
  - Genre
- From the dataset, there will be hundreds of thousands of Authors, consequently immediately displaying all of the Authors within a search facet would not be useful.
- Have a unique link for every Author being able to see the if the author has any series, with links to books in the series
- Users should be able to facet on specific fields that make sense from a search perspective

**Note:** As the dataset is discovered and understood and the way in which users will interact with the data, this may lead to other use-cases and requirements. (see the section on [The Iterative Implementation Process](#) for more details).

## 1. Understanding the Dataset

Whether you are starting with an existing Solr managed schema, or you are looking to index a new dataset, understanding what the dataset contains is the **MOST** important part of the process. This will drive all other decisions - after all, without the correct data, you won't be able index it, search on it, facet upon it, or present it to the end-user.

The above may seem obvious, however, understanding the data, in conjunction with the desired pages that you want to render, will inform whether you can derive data from the underlying dataset, combine the dataset with other external datasets, or need to use a back-end datastore to generate pages with links.

In this example we will be looking at setting a separate dataset containing just over 3 million records based on fiction books.

The dataset that I have access to has duplicates, mis-spellings, missing information, and information that is just plain wrong. Before the indexing process, the data will need to be cleaned and any additional derived data generated. Ignoring all of the previous data problems, the dataset contains the following information, with some notes about we would want to search and facet the dataset

### Data

- **Author**
  - *Search on the author, or authors (the book may have a collaboration of authors)*
  - *Facet on the author, however with around 500,000 authors, we don't want to display all of the facets on the initial search page*
  - *See a page that lists all of the author's works (in order of date of publication) with links to the series that are available*
  - *See the Author in the returned documents*
- **Title**
  - *Search on the title*
  - *Will not be faceted, but will be displayed in the returned documents*
- **Description**
  - *Search on the description*
  - *Will not be faceted, but will be displayed in the returned documents*
- **Book Image**
  - *See the image of the book*

- *Will not be faceted, but will be displayed in the returned documents*
- **Buy URL**
  - *See the link to the URL to buy the book*
  - *Will not be faceted, but will be displayed in the returned documents*
- **Genre**
  - *Search on the genre*
  - *Be able to select multiple genres as a facet*
- **Number of Pages**
  - *See the number of pages in a book in the returned document*
  - *Will not be faceted, but will be displayed in the returned documents*
- **First published year**
  - *See the first published year of the book*
  - *Will be faceted, but as there are close to 100 years of books, we don't want to display every year on the initial search page*
- **Language**
  - *Select the language of the book*
  - *Be able to select only a single language*
- **Paperback/Hardcover**
  - *Select whether users want a hardcover, or paperback book*
- **Series**
  - *Search on the series name*
  - *Will be faceted, however as there will be many series of books, this should not appear with the initial results*
- **Price**
  - *Select a price range for books*
  - *Be able to sort by price range*

## Derived Data

- **ID**
  - *This is the required primary key for the Solr search server and the database primary key has been chosen for this purpose*
- **Author A-Z index** - derived from the first character of the 'Author' surname
  - *Used to present a list of pages by the first letter of the author's surname*
- **Decade published** - derived from 'Year published'
  - *Can be used to help narrow down the decade in which the book is published, i.e. the user should be able to select the decade first, and then be able to select the year within that decade*

- *Will be used as a facet*
- **Book type** - derived from the 'Number of pages' - one of 'Flash Fiction', 'Short Story' 'Novelette', 'Novella', or 'Novel'
  - *Will be used as a facet*
  - *Will not be returned with the search documents.*
- **Text** - A general purpose field that will have its contents analysed and used for search queries
  - *Will be used for the search query*
  - *Will not be returned in the search documents*
  - *Solr will generate this, rather than data being input.*

Now that the data types are known and how we are going to use them, let's determine how we are going to index them in the Solr search engine.

## 2. Configure the Solr Index

#TODO - rewrite

Looking at the data that you have indexed, or would like to index, the configuration of Panl is determined on two items:

1. Whether the indexed Solr field should be a facet, or just a field, and whether multiple values are available for the field, and
2. The data type of the Solr field, which will determine the configuration options that are available through the Panl server

### REMEMBER

- **Facets** will allow you to **filter** the results.
- **Fields** will be **returned** with the search documents - you cannot filter the results of the field.

Whether you choose a Solr field to be configured in Panl to be a Facet, or a Field, both

- Can be returned with the search documents
- Are able to be used for sorting options

The difference with Facets, is that in addition to the above, Facets

- Are able to be selected to filter the results and, depending on the data type, can be further configured for prefixes, suffixes, ranges and value replacements



**Note:** If in doubt as to whether the Solr field will ever need to be configured as a Facet in Panl, err on the side of yes (i.e. set `indexed="true"`). Remember that the Panl configuration can present a Solr field as either a Facet or a Field, however if it is not set to indexed in the Solr configuration, it can only ever be a Field for Panl.

Noting the following rules for Solr configuration:

1. If we want to be able to search, sort, or facet on the data, then it must be indexed.
2. If we want to see the results in the returned documents, then it must be stored
3. If the dataset field can only hold a single value, then multivalued is No, otherwise it is set to Yes.

Using information above and the requirements for the dataset, the Solr field definitions for importing and indexing is as follows:

Solr Field Name	Data Type	Analysed	Multivalued	Indexed	Stored
Author	String	No	Yes	Yes	Yes
Title	String	No	No	Yes	Yes
Description	String	No	No	Yes	Yes
Book Image	String	No	No	No	Yes
Buy URL	String	No	No	No	Yes
Genre	String	No	Yes	Yes	Yes
Number of Pages	Integer	No	No	No	Yes
First Published Year	Integer	No	No	Yes	Yes
Language	String	No	No	Yes	Yes
Paperback / Hardcover	Boolean	No	No	Yes	Yes
Series	String	No	No	Yes	Yes
Price	Float	No	No	Yes	Yes

Solr Field Name	Data Type	Analysed	Multivalued	Indexed	Stored
<i>Note: The following fields are derived from the dataset before being indexed by Solr</i>					
ID	Integer	No	No	Yes	Yes
Author A-Z index	String	No	No	Yes	No
Decade First Published	Integer	No	No	Yes	No
Book Length	String	Yes	No	Yes	No
Text	String	Yes	No	No	No

The above table would lead to the following snippet of the Solr managed schema file with the Solr field names and values where set to true highlighted.



**Remember:** The way that the Solr managed schema is configured can span across multiple CaFUPs and that you do not need to configure or include each facet or field for each of the pages that you want to render.

Configure the schema so that it will cover all requirements, and then let the Panl configuration define how the facets and results are returned.

If in doubt, you can always set up a separate Solr collection with a Panl configuration for a specific need or use case.

```

01 <schema name="book-store" version="1.6">
02   <field name="_version_" type="plong" indexed="false" stored="false"/>
03
04   <field name="id" type="string" stored="true" index="true" required="true" ✎
05     multiValued="false" />
06
07   <field name="author" type="string" indexed="true" stored="true" ✎
08     multiValued="true" />
09
10  <field name="title" type="string" indexed="true" stored="true" ✎
11    multiValued="false" />

```

```

08 <field name="description" type="pint" indexed="true" stored="true" ←
      multiValued="false" />
09 <field name="book_image" type="string" indexed="false" stored="true" ←
      multiValued="false" />
10 <field name="buy_url" type="string" indexed="false" stored="true" ←
      multiValued="false" />
11 <field name="genre" type="string" indexed="true" stored="true" ←
      multiValued="true" />
12 <field name="num_pages" type="pint" indexed="false" stored="true" ←
      multiValued="false" />
13 <field name="first_published_year" type="pint" indexed="true" stored="true" ←
      multiValued="false" />
14 <field name="language" type="string" indexed="true" stored="true" ←
      multiValued="false" />
15 <field name="is_paperback" type="boolean" indexed="true" stored="true" ←
      multiValued="false" />
16 <field name="series" type="string" indexed="true" stored="true" ←
      multiValued="false" />
17 <field name="price" type="pfloat" indexed="true" stored="true" ←
      multiValued="false" />
18
19 <field name="a_to_z_index" type="string" indexed="true" stored="false" ←
      multiValued="false" />
20 <field name="decade_published" type="pint" indexed="true" stored="false" ←
      multiValued="false" />
21 <field name="book_length" type="pint" indexed="true" stored="false" ←
      multiValued="false" />
22
23   <field name="text" type="text_general" indexed="true" stored="true" ←
      multiValued="true"/>
24
25 <uniqueKey>id</uniqueKey>
26
27 <copyField source="author" dest="text" />
28 <copyField source="title" dest="text" />
29 <copyField source="description" dest="text" />
30 <copyField source="genre" dest="text" />
```

```

31 <copyField source="series" dest="text" />
32 ...
33 ...
34 </schema>
```

Working through the schema:

**Line 1:**

This is the schema name which maps to the Solr collection name and will be used by Panl for the CaFUPs

**Line 2:**

The `_version_` field is required by a Solr Cloud deployment - this is an internal field, generated automatically by Solr and is used by the partial update procedure, the update log process. You may not need to have this field, however it is mandatory for a Solr Cloud instance

**Line 4:**

The `id` field for uniquely identifying a Solr document within the collection

**Lines 6 - 17:**

The fields that come directly from the book store dataset, note the values of the `xml` element for the attributes `multivalued`, `indexed`, and `stored`

**Lines 19-21:**

The fields that are derived from the data.

**Line 23:**

This is a field that is used as a storage area for every other field that needs to be searched on - see lines 27-31 below.

**Line 25:**

This element tells Solr what the unique key is for this collection

**Lines 27-31:**

This copies the values from the required fields so that they can be searched upon

### Line 33:

For clarity and space, the Solr field definitions and additional xml elements were not included and replaced by an ellipses.

### Line 35:

The end of the Solr schema definition



**IMPORTANT:** When defining the managed schema for a Solr collection, you need to consider \_\_ALL\_\_ of the use cases of the data and whether each field is going to be indexed and/or stored.

You can then configure the Panl server through the CaFUPs to facet and return just the individual facets and fields that you want.

## 3. Configure the Panl Server

Now that we understand the dataset, and the Solr search server is going to index the data, we can extend the Solr field definitions for the initial Panl configuration.



**Notes:** The following configuration is for the default search page, alternate configurations will be defined for further search page implementations.

Solr Field Name	Data Type	Facet or Field	Facet Type	Sortable	Additional information
Author	String	Facet	Regular	No	Hierarchical, Prefix/Suffix
Title	String	Field	N/A	No	
Description	String	Field	N/A	No	
Book Image	String	Field	N/A	No	

Solr Field Name	Data Type	Facet or Field	Facet Type	Sortable	Additional information
Buy URL	String	Field	N/A	No	
Genre	String	Facet	OR	No	Prefix/Suffix
Number of Pages	Integer	Field	N/A	No	
First Published Year	Integer	Facet	Regular	Yes	Hierarchical, Prefix/Suffix
Language	String	Facet	Regular	No	
Paperback / Hardcover	Boolean	Facet	BOOLEA N	No	Value replacement
Series	String	Facet	Regular	No	Hierarchical
Price	Float	Facet	RANGE	Yes	

**Note:** The following fields are derived from the dataset before being indexed by Solr

ID	String	Field	N/A	No	
Author A-Z index	String	Facet	Regular	Yes	Prefix/Suffix
Decade First Published	Integer	Facet	RANGE	No	
Book length	String	Facet	Regular	No	
Text	String	Ignored	N/A	No	

The above can be considered the default search page configuration, there will be other `<collection_uri>.panl.properties` files defined

## The Default panl.properties file

For all of the Book Store URI path parts, the `panl.properties` file will be the same, all comments have been stripped from the file and the only incomplete line is the last line `panl.collection.book-store=...` which will have the `<collection_uri>.panl.properties` files added once we have configured them.

```
01 solrj.client=CloudSolrClient
02 solr.search.server.url=http://localhost:8983/solr,http://localhost:7574/solr
03 panl.results.testing.urls=true
```

```

04 panl.status.404.verbose=true
05 panl.status.500.verbose=true
06 panl.collection.book-store=...

```

## The Default Search Page Configuration

To generate the default search page configuration the in-built Panl utility is the quickest way to get up and running fast:

For more information on the available options, see the section for the command line options for the [Panl Generator](#) in the appendices.



**IMPORTANT:** Be aware that everytime that you use the Panl generator, there is a chance that the generated files will change their LPSE codes. This will happen if the Panl generator has a LPSE code that clashes and needs to choose a random one.

### \*NIX command

Command(s)
<pre> cd PANL_INSTALL_DIRECTORY  bin/panl generate ^   -schema src/dist/sample/solr/book-store/managed-schema.xml ^   -properties src/dist/sample/panl/book-store/panl.properties </pre>

### Windows command

Command(s)
<pre> cd PANL_INSTALL_DIRECTORY </pre>

**Command(s)**

```
bin\panl.bat generate ^
    -schema src\dist\sample\solr\book-store\managed-schema.xml ^
    -properties src\dist\sample\panl\book-store\panl.properties
```

This will generate two files in the `src/dist/sample/panl/book-store/` directory named `panl.properties` and `book-store.panl.properties` the text of which is not included - however the complete file can be seen in the GitHub repository

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/book-store/book-store.panl.properties>

## General Properties Configuration

Skipping over the defaults values for the `panl.param.*` properties (and no prefixes or suffixes were added to either the `panl.param.page` or `panl.param.numrows` properties) the following properties were changed:

`solr.numrows.default=20`

This was changed from the default value of `10` as `20` results seems to be a good starting point for such a large collection

`solr.highlight=false`

Changed from `true`, no highlighting will be required on the Book Store collection

`panl.lpse.ignore=i`

We still want to be able to search on this field and be able to pull out the results, generally with a canonical URL - e.g.

`/Michael+Connelly+Harry+Bosh+Series+The+Black+Echo/1678/zi/`

And use the `id` Solr field (LPSE code '`i`') as the lookup key with the value `1678`, the rest of the URI path part will be ignored.

`panl.sort.fields=price,a_to_z_index,first_published_year`

These are the fields that are going to be able to be sortable - remember that relevancy always available as a sort order and is the default sort order if no other sort order is selected

## The Generated Fields and Facets Configurations

Comments providing information about the settings have been removed from the examples below.

### Solr Field 'id'

```

01 # <field "indexed"="true" "stored"="true" "name"="id" "type"="string" ←
      "multiValued"="false" "required"="true" />
02 panl.facet.i=id
03 panl.name.i=Id
04 panl.type.i=solr.StrField
05 #panl.prefix.i=prefix
06 #panl.suffix.i=suffix
07 #panl.when.i=
08 #panl.facetsort.i=index

```

This facet will be left as it is for the moment, but this will be ignored by the Panl server as the property `panl.lpse.ignore=i` has this LPSE code.

You can still return this as a field in the Panl results, so that if you need the unique id of the book for additional functionality (e.g. adding to a cart, linking to a separate page, looking up further details). See the `panl.results.fields.*` properties.

### Solr Field 'author'

```

01 # <field "indexed"="true" "stored"="true" "name"="author" "type"="string" ←
      "multiValued"="true" />
02 panl.facet.a=author
03 panl.name.a=Author
04 panl.type.a=solr.StrField
05 panl.prefix.a=Author
06 #panl.suffix.a=suffix
07 panl.when.a=q,A
08 #panl.facetsort.a=index

```

A prefix has been added on line 6 of 'Author' (note the ending whitespace).

There are too many authors to have this as a facet, and they will be ordered by the number of books that have been published, so this facet is configured to only appear if a search query is set, or if the first letter of the surname is selected. Consequently line 7 has been un-commented so that the `panl.when.a` property has a value. This facet will only appear if the search query (LPSE code '`q`') or the `a_to_z_index` facet (LPSE code '`A`') has been selected.

### Solr field 'title'

```

01 # <field "indexed"="true" "stored"="true" "name"="title" "type"="string" ←
      "multiValued"="false" />
02 panl.field.t=title
03 panl.name.t=Title
04 panl.type.t=solr.StrField
05 #panl.prefix.t=prefix
06 #panl.suffix.t=suffix
07 #panl.when.t=
08 #panl.facetsort.t=index

```

The generator has configured this Solr field as a Panl facet as it is both indexed and stored in Solr - this has been changed to a field, rather than a facet.

### Solr field 'description'

```

01 # <field "indexed"="true" "stored"="true" "name"="description" "type"="string" ←
      "multiValued"="false" />
02 panl.field.d=description
03 panl.name.d=Description
04 panl.type.d=solr.StrField
05 #panl.prefix.d=prefix
06 #panl.suffix.d=suffix
07 #panl.when.d=
08 #panl.facetsort.d=index

```

The generator has configured this Solr field as a Panl facet as it is both indexed and stored in Solr - this has been changed to a field, rather than a facet.

## Solr Field 'book\_image'

No configuration changes made

```

01 # <field "indexed"="false" "stored"="true" "name"="book_image" "type"="string" <
      "multiValued"="false" />
02 panl.field.b=book_image
03 panl.name.b=Book Image
04 panl.type.b=solr.StrField
05 #panl.prefix.b=prefix
06 #panl.suffix.b=suffix
07 #panl.when.b=
08 #panl.facetsort.b=index

```

## Solr Field 'buy\_url'

No configuration changes made

```

01 # <field "indexed"="false" "stored"="true" "name"="buy_url" "type"="string" <
      "multiValued"="false" />
02 panl.field.B=buy_url
03 panl.name.B=Buy Url
04 panl.type.B=solr.StrField
05 #panl.prefix.B=prefix
06 #panl.suffix.B=suffix
07 #panl.when.B=
08 #panl.facetsort.B=index

```

## Solr field 'genre'

```

01 # <field "indexed"="true" "stored"="true" "name"="genre" "type"="string" <
      "multiValued"="true" />
02 panl.facet.g=genre
03 panl.or.facet.g=true
04 panl.name.g=Genre

```

```

05 panl.type.g=solr.StrField
06 #panl.prefix.g=prefix
07 #panl.suffix.g=suffix
08 #panl.when.g=
09 #panl.facetsort.g=index

```

This will be an OR facet as it is configured with the property `panl.or.facet.g=true`, meaning that end users can select one or more of the facet values.

### Solr Fields 'num\_pages'

No configuration changes made

```

01 # <field "indexed"="false" "stored"="true" "name"="buy_url" "type"="string" ←
      "multiValued"="false" />
02 panl.field.B=buy_url
03 panl.name.B=Buy Url
04 panl.type.B=solr.StrField
05 #panl.prefix.B=prefix
06 #panl.suffix.B=suffix
07 #panl.when.B=
08 #panl.facetsort.B=index

```

### Solr Fields 'first\_published\_year'

```

01 # <field "indexed"="true" "stored"="true" "name"="first_published_year" ←
      "type"="pint" "multiValued"="false" />
02 panl.facet.f=first_published_year
03 panl.name.f=First Published Year
04 panl.type.f=solr.IntPointField
05 panl.prefix.f=First published in
06 #panl.suffix.f=suffix
07 panl.when.f=D
08 #panl.facetsort.f=index

```

Add in a prefix of 'First published in ' - Line 5 - and this will only appear when the decade\_published facet has been selected (LPSE code 'D') - Line 7.

### Solr Fields 'language'

No configuration changes made

```

01 # <field "indexed"="true" "stored"="true" "name"="language" "type"="string" <
      "multiValued"="false" />
02 panl.facet.l=language
03 panl.name.l=Language
04 panl.type.l=solr.StrField
05 #panl.prefix.l=prefix
06 #panl.suffix.l=suffix
07 #panl.when.l=
08 #panl.facetsort.l=index

```

### Solr field 'is\_paperback'

```

01 # <field "indexed"="true" "stored"="true" "name"="is_paperback" "type"="boolean" <
      "multiValued"="false" />
02 panl.facet.I=is_paperback
03 panl.name.I=Book Format
04 panl.type.I=solr.BoolField
05 #panl.prefix.I=prefix
06 #panl.suffix.I=suffix
07 panl.bool.I.true=Paperback
08 panl.bool.I.false=Hardcover
09 #panl.when.I=
10 #panl.facetsort.I=index

```

The display name has been changed to be 'Book Format' (Line 3) and a Boolean value replacement for both the true and false values (Lines 7 and 8).

Another way that this could have been index by Solr was to derive the data and store the book format as a string - i.e. `type="string"` with the values '`Paperback`' and '`Hardcover`', however keeping this as a boolean value with value replacements means that additional CaFUPs could be configured with different values for true and false if additional URLs were needed to be generated.

### Solr field 'series'

```

01 # <field "indexed"="true" "stored"="true" "name"="series" "type"="string" ←
      "multiValued"="false" />
02 panl.facet.S=series
03 panl.name.S=Series
04 panl.type.S=solr.StrField
05 #panl.prefix.S=prefix
06 #panl.suffix.S=suffix
07 panl.when.S=a
08 #panl.facetsort.S=index

```

This facet will only be passed through if an `author` facet (LPSE code '`a`') has been selected (Line 7).

### Solr field 'price'

```

01 # <field "indexed"="true" "stored"="true" "name"="price" "type"="pfloat" ←
      "multiValued"="false" />
02 panl.facet.P=price
03 panl.name.P=Price
04 panl.type.P=solr.FloatPointField
05 #panl.prefix.P=prefix
06 #panl.suffix.P=suffix
07 #panl.when.P=
08 panl.range.facet.P=true
09 panl.range.min.P=5
10 panl.range.max.P=100
11 panl.range.prefix.P=From
12 panl.range.infix.P=\ to

```

```

13 panl.range.suffix.P=\ dollars
14 panl.range.min.wildcard.P=true
15 panl.range.max.wildcard.P=true
16 #panl.facetsort.P=index

```

This facet is a RANGE facet (configured with the `panl.range.facet.P=true` property) - Lines 8 to 15. As an example, the configuration will generate the URI path part.

`/From+5+to+100+dollars/P/`

Additionally, with the wildcard properties set, it will generate a Solr query when the minimum or maximum values are passed through to use less than or greater than, respectively. I.e. if the URI path part was used, as they are both a minimum and maximum value, the query would prices between `5` or below and `100` and greater.

For a URI path part of

`/From+20+to+100+dollars/P/`

It would return books greater than `20` (even if they are greater than `100`)

For the URI path part of

`/From+45+to+50+dollars/P/`

It will only return values between `45` and `50` (inclusive)

### Solr field 'a\_to\_z\_index'

```

01 # <field "indexed"="true" "stored"="false" "name"="a_to_z_index" "type"="string" <
      "multiValued"="false" />
02 panl.facet.A=a_to_z_index
03 panl.name.A=Authors (A-Z)
04 panl.type.A=solr.StrField
05 #panl.prefix.A=prefix
06 #panl.suffix.A=suffix
07 #panl.when.A=
08 panl.facetsort.A=index

```

The Panl name of the facet has been changed to 'Authors (A-Z)' - Line 3.

Normally Solr will return facets in order of decreasing count, however by setting Line 8 to index, this will sort the facets by index order, i.e. by the alphabetical order, rather than the number of returned documents for the specific facet.

### Solr field 'decade\_published'

No configuration changes made

```

01 # <field "indexed"="true" "stored"="false" "name"="decade_published"   ←
      "type"="pint" "multiValued"="false" />
02 panl.facet.D=decade_published
03 panl.name.D=Decade Published
04 panl.type.D=solr.IntPointField
05 #panl.prefix.D=prefix
06 #panl.suffix.D=suffix
07 #panl.when.D=
08 #panl.facetsort.D=index

```

### Solr field 'text'

```

01 # <field "indexed"="true" "stored"="false" "name"="text" "type"="text_general" ←
      "multiValued"="true" />
02 #panl.facet.T=text
03 #panl.name.T=Text
04 #panl.type.T=solr.TextField
05 # The following two properties are optional and the values should be changed
06 #panl.prefix.T=prefix
07 #panl.suffix.T=suffix
08 #panl.when.T=
09 #panl.facetsort.T=index

```

This is an internal Solr field that is used as a multi valued text field to store all fields that need to be searched against. As such, it is not going to be used as a facet, or a field, so the entire entry has been commented out.



**IMPORTANT:** Ensure that you remove the 'text' field from all Panl configured FieldSets and LPSE orders, as the panl server will error on startup if it finds a field that it is not defined - see the following properties:

- panl.lpse.order
- panl.results.fields.\*

## Solr field 'book\_length'

No configuration changes made

```

01 # <field "indexed"="true" "stored"="false" "name"="book_length" "type"="string" <
      "multiValued"="false" />
02 panl.facet.L=book_length
03 panl.name.L=Book Length
04 panl.type.L=solr.StrField
05 #panl.prefix.L=prefix
06 #panl.suffix.L=suffix
07 #panl.when.L=
08 #panl.facetsort.L=index

```

## Panl LPSE Orders, FieldSets, and Sorting

The final part of the Panl configuration is the LPSE order, the FieldSets, and the available fields/facets to sort the results documents.

### The Panl LPSE Order

Going through the field and facet configuration items, two of the facets were changed to be fields, and one of the facets was removed (commented out), they were:

1. The Solr field title (LPSE code 't') is now a field, not a facet

2. The Solr field `description` (LPSE code 'd') is now a field, not a facet
3. The Solr field `text` (LPSE code 'T') was commented out

So the original LPSE order (strikethrough text below)

```

01 panl.lpse.order=z,\n
02 i,\n
03 a,\n
04 t,\n
05 d,\n
06 b,\n
07 B,\n
08 g,\n
09 N,\n
10 f,\n
11 l,\n
12 I,\n
13 S,\n
14 P,\n
15 A,\n
16 D,\n
17 T,\n
18 L,\n
19 q,\n
20 p,\n
21 n,\n
22 s,\n
23 o

```

The final version becomes

```

01 panl.lpse.order=z,\n
02 i,\n
03 a,\n
04 b,

```

```

05 B,\n
06 g,\n
07 N,\n
08 f,\n
09 l,\n
10 I,\n
11 S,\n
12 P,\n
13 A,\n
14 D,\n
15 L,\n
16 q,\n
17 p,\n
18 n,\n
19 s,\n
20 o

```

## The Panl FieldSets

Here, the only configured FieldSets is going to be the default, with no other FieldSets defined. I.e. the `panl.results.fields.firstfive` property has been removed.

```

01 panl.results.fields.default=id,\n
02 author,\n
03 title,\n
04 description,\n
05 book_image,\n
06 buy_url,\n
07 genre,\n
08 num_pages,\n
09 first_published_year,\n
10 language,\n
11 is_paperback,\n
12 series,\n
13 price,\n
14 a_to_z_index,

```

```

15 decade_published,\n
16 book_length,\n
17 text

```

Lines 14 to 17 have been removed. The id field (Line 1) was kept in as it may be useful to link to the database for other purposes. The final property looks thusly:

```

01 panl.results.fields.default=id,\n
02 author,\n
03 title,\n
04 description,\n
05 book_image,\n
06 buy_url,\n
07 genre,\n
08 num_pages,\n
09 first_published_year,\n
10 language,\n
11 is_paperback,\n
12 series,\n
13 price

```

## The Panl Sort Fields

To define the sort fields, use the `panl.sort.fields` property with a list of comma separated values. Each of the sort fields must match the Solr field name, NOT the Panl LPSE code as these are passed directly through to the Solr server.

```
panl.sort.fields=price,a_to_z_index,first_published_year
```



**Note:** There is only one sorting fields property for the file and spans across all FieldSets defined in this file. You may add as many sorting fields as you would like, you do not need to make the options available to the end user.

## Testing the Configuration

At this point (assuming that the data has been correctly added and indexed to the Solr Search server) you will be able to start the Panl server and view your single CaFUP on the Panl Results Viewer - <http://localhost:8181/panl-results-viewer/book-store/default/>.

## Configuration Change Summary

Panl Field Name	LPSE code	Changes
Author	a	Added hierarchy by setting <code>panl.when.a=q,A</code> Added prefix by setting <code>panl.prefix.a=Author</code> <i>(note the space at the end of the property)</i>
Title	t	Changed from a facet to a field - i.e. <code>panl.facet.t=title</code> to <code>panl.field.t=title</code> Remove field from <code>panl.lpse.order</code>
Description	d	Changed from a facet to a field - i.e. <code>panl.facet.d=description</code> to <code>panl.field.d=description</code> Remove field from <code>panl.lpse.order</code>
Book Image	b	No changes to panl field/facet configuration
Buy URL	B	No changes to panl field/facet configuration
Genre	g	Made this an OR facet by setting <code>panl.facet.or.g=true</code>
Number of Pages	N	No changes to panl field/facet configuration

Panl Field Name	LPSE code	Changes
First Published Year	f	<p>Added a prefix by setting  <code>panl.prefix.f=First published in</code>  <i>(note the space at the end of the property)</i></p> <p>Added to sort fields by adding to the property  <code>panl.sort.fields</code></p>
Language	I	No changes to panl field/facet configuration
Paperback / Hardcover	I	<p>Changed the Panl name by setting  <code>panl.name.I=Book Format</code></p> <p>Added BOOLEAN value replacement values by setting  <code>panl.bool.I.true=Paperback</code>  <code>panl.bool.I.false=Hardcover</code></p>
Series	S	<p>Added hierarchy by setting  <code>panl.when.S=a</code></p>
Price	P	<p>Made this a RANGE facet with value replacement by setting the following  <code>panl.range.facet.P=true</code>  <code>panl.range.min.P=5</code>  <code>panl.range.max.P=100</code>  <code>panl.range.prefix.P=From</code>  <i>(note the space at the end of the property)</i>  <code>panl.range.infix.P=\ to</code>  <i>(note the space at the end of the property)</i>  <code>panl.range.suffix.P=\ dollars</code>  <code>panl.range.min.wildcard.P=true</code>  <code>panl.range.max.wildcard.P=true</code></p> <p>Added to sort fields by adding to the property  <code>panl.sort.fields</code></p>

**Note:** The following fields are derived from the dataset before being indexed by Solr

id	i	Added to ignored facet by setting <code>panl.lpse.ignore=i</code>
----	---	--

Panl Field Name	LPSE code	Changes
Author A-Z index	A	Change the Panl field name by setting <code>panl.name.A=Authors (A-Z)</code> Removed field from <code>panl.results.fields.default</code> Added to sort fields by adding to the property <code>panl.sort.fields</code> Added sorting of the facet values by index, rather than count <code>panl.facetsort.A=index</code>
Decade First Published	D	Removed field from <code>panl.results.fields.default</code>
Book length	L	Removed field from <code>panl.results.fields.default</code>
Text	T	Commented out all properties for this field Remove field from <code>panl.lpse.order</code> Removed field from <code>panl.results.fields.default</code>

## 4. Determine the Web Pages to Render

In addition to the default search page with the default functionality, new page requirements are as follows.

1. SEO friendly URLs that list of all books published by Author (in order of publication) along with the ability to facet by any series that the Author has written.
2. SEO friendly URLs that lists all book series for Authors (in order of publication) that exist
3. A list of all Authors with their associated books and their series

### Author and Author series

Both of these pages can be generated with a single Panl Configuration (included as the `dist/sample/panl/book-store/author-alphabetical.panl.properties` file) For each of the links

to the authors, a link was generated in the format of `/Author+<author_name/a/` and then the Panl server was left to do its work.



**Tips:** The majority of these pages could also be directly generated through a database query, however you would also need to implement sorting, pagination, and any additional faceting as well, all of which Panl has built-in and ready to use.

## Author Listing

The indexing of the author listing page - i.e. a complete list of all authors within the dataset could not sufficiently be satisfied by the current dataset, so a new dataset was created and indexed by Solr (not included in the release package). I was then able to produce the pages that were required by passing it through the Panl server to utilise the searching, sorting, pagination, and hierarchical facets.

A single Panl solution may not fit all use cases so you may need to look at additional datasets, or simply by using pages generated from a database.

## The Iterative Implementation Process<sup>12</sup>

When testing the configurations, the original implementation didn't quite make sense, so

1. Books have more than one genre assigned to them, it didn't make sense to have the genre Panl field as an OR facet, so this was removed, after all users would want to narrow down by genre - say Murder AND Mystery, not increase the results by using Murder OR Mystery genres
2. The way the dataset was index by the Solr collection didn't suit all of the needs, so a separate collection was created to hold only an individual author with a multifield list of titles attached to it
3. Searching a book by decade didn't really make sense (or even having the hierarchical facet for `first_published_in`). They were removed as facets and made to be fields. The data was left in the Solr collection index, as they may be of use later

---

<sup>12</sup> Or, the mistakes that were made with the implementation.

4. The site that was generated was a joining of the web application server, the database, and then the Panl configuration. Some pages were generated by the database and served up by the web application which were then linked to the Panl implementation.
5. Any Solr fields that are of type float, when returned with the documents there may be storage errors, for example, each of the books are priced as a float as 19.99, when returned with the document, it comes back as 19.98999971118164 - which rounds to 19.99. Instead I derived another field to have it as an integer for price in cents, then on the front end, I just formatted it to the correct decimal place.

The changes made and implementation details have not been provided in the included sample dataset.

The Panl server runs purely on configuration, so any changes that are made to either of the configurations will be utilised at runtime. Provided that the Solr collection is set up to allow the broadest array of functionality, this becomes a very short iterative process.

~ ~ ~ \* ~ ~ ~

# Working with any Dataset

---

This section dives deeper into the supported Panl field configurations

The drivers behind the Panl configuration options are the Solr fieldType and whether it is configured to be a facet or a field. For example, the variants Solr field in the mechanical pencils Solr managed schema:

```
# <field "indexed"="true" "stored"="true" "name"="variants" "type"="string"
    "multiValued"="true" />
```

The `type` (i.e. the Solr fieldType) of `string` above is then referenced in the managed schema to

```
<fieldType name="string" class="solr.StrField" sortMissingLast="true" />
```

It is the `class` attribute above which is included in the `mechanical-pencils.panl.properties` file as a comment which then drives the configuration available:

```
# <field "indexed"="true" "stored"="true" "name"="variants" "type"="string"
    "multiValued"="true" />
panl.field.v=variants
panl.name.v=Variants
panl.type.v=solr.StrField
```

The `panl.type.v=solr.StrField` is the property value that Panl parses to determine the available configuration options.

## Supported Solr Data Types

The following Solr field types are explicitly supported in Panl, and the configuration options that are available are described.

All facets may be hierarchical, and can be sorted by either their count, or their value.

Solr Field Type	Prefix / Suffix	Range	OR	Boolean Value Replace	Date Range
solr.BoolField	YES	NO	NO	YES	NO
solr.DatePointField	NO	NO	NO	NO	YES
solr.DoublePointField	YES	YES	YES	NO	NO
solr.FloatPointField	YES	YES	YES	NO	NO
solr.IntPointField	YES	YES	YES	NO	NO
solr.LongPointField	YES	YES	YES	NO	NO
solr.StrField	YES	NO	YES	NO	NO
solr.TextField	YES	NO	YES	NO	NO

## Unsupported/Partially Supported Solr Field Types

Whilst the following fields aren't officially supported by Panl, they can still be returned within the results documents (i.e. configured to be fields). If they are configured to be facets, then the operation of Panl is undefined, however, they may work

- `solr.BBoxField`
- `solr.BinaryField`
- `solr.CollationField`
- `solr.CurrencyFieldType`
- `solr.DateRangeField`
- `solr.ExternalFileField`
- `solr.ICUCollationField`
- `solr.LatLonPointSpatialField`
- `solr.NestPathField`
- `solr.PointType`
- `solr.PreAnalyzedField`
- `solr.RankField`
- `solr.RptWithGeometrySpatialField`
- `solr.SortableTextField`

- `solr.SpatialRecursivePrefixTreeFieldType`
- `solr.UUIDField`

## Facet and Field Types

Each of the Panl fields can be defined as a Field, or a Facet. If the Panl field is configured to be a 'Field', then it will be returned with the documents, but no other configuration options are applicable. If it is configured as a 'Facet' then, depending on the Solr field type, the configuration properties available to Panl will vary.

### Fields

Fields are returned with the documents so that they may be rendered to the results page. They can be sorted on, but they **CANNOT** be faceted on. Any Solr field that is stored (i.e. `stored="true"` in the managed schema) may be a field, additionally any Solr field that is indexed (i.e. `indexed="true"` in the managed schema) may be set as a facet, or a field.

Multiple `<panl_collection_uri>.panl.properties` files can be defined with separate Panl properties files with different configurations of facets and fields all connecting to a single Solr search collection.

To configure any Solr field as a Panl field, use the `panl.field.<lpse_code>` property, rather than the `panl.facet.<lpse_code>` property.

```

01 # <field "indexed"="true" "stored"="true" "name"="diameter" "type"="pint" <
      "multiValued"="false" />
02 panl.field.d=diameter
03 panl.name.d=Diameter
04 panl.type.d=solr.IntPointField

```

There are no other panl configuration options available for a field and if any are set, will be silently ignored.

### Hints/Recommendations:

- Use fields for any Solr field that you want to be able to sort on, or be returned with the documents.
- Any facet can be configured to be a field - remember that you may have multiple CaFUPs configured using it as a field or a facet in different places.

- Any field can be configured to be returned or ignored with different Panl FieldSets.

## Regular Facets

If you are going to facet on a field, then the field type should be at least indexed and it is a good idea to have it stored as well, but not analysed. Multi valued fields are also good to use as facets as they will allow multiple choices for faceting the results, without the need for an OR facet..



**Note:** The reason behind not analysing the Solr field is that if the field is also analysed, then the facets that are returned will be broken up into their word forms.

### Hints/Recommendations:

- Regular facets are easy to set up, use, implement, offer prefixes and suffixes, and can be used as a sort order and do not have to be returned in the result documents.
- If they are multi-valued, the end user will be able to select more than one.
- They can be configured to be an OR facet if they are single valued, which will allow users to select more than one value.

## BOOLEAN Facets

BOOLEAN facets may only have one of two values, namely true or false and can have those values replaced by Panl from a more SEO friendly string to their underlying values.

The only Solr FieldTypes that allow true/false value replacement is the `solr.BoolField` and these can be set with the `panl.bool.<lpse_code>.true` and `panl.bool.<lpse_code>.false` properties.

You may still assign a prefix and suffix to the BOOLEAN facet. As an example, the `disassemble` Solr field from the mechanical pencils configuration has the following properties in the mechanical pencils configuration:

```

01 # <field "indexed"="true" "stored"="true" "name"="disassemble" "type"="boolean" <-
      "multiValued"="false" />
02 panl.facet.D=disassemble
03 panl.name.D=Disassemble
04 panl.type.D=solr.BoolField
05 panl.bool.D.true=able to be
06 panl.bool.D.false=cannot be
07 panl.suffix.D=\ disassembled

```

### Hints/Recommendations:

- For BOOLEAN facets, use the true and false value replacements where it makes sense.
- Not all BOOLEAN facets have to have the value replacement, if this field is not used often
- If you want to shorten the URI path part further, replace the true/false values with single characters - e.g. 1/0 or y/n

## RANGE Facets

RANGE facets allow the end user to filter the results of the facet by a range of values and have the most Panl configuration options available. Whilst ranges are available on String types of data in Solr, the main usage in Panl is with integer or floating point numbers.

RANGE facets will also return the individual values for each of the ranges which do not have to be rendered to the page.

```

01 # <field "indexed"="true" "stored"="true" "name"="weight" "type"="pint" <
      "multiValued"="false" />
02 panl.facet.w=weight
03 panl.name.w=Weight
04 panl.type.w=solr.IntPointField
05 panl.suffix.w=\ grams
06 panl.range.facet.w=true
07 panl.range.min.w=10
08 panl.range.max.w=50
09 panl.range.prefix.w=weighing from
10 panl.range.infix.w=\ to
11 panl.range.suffix.w=\ grams
12 panl.range.min.value.w=from light
13 panl.range.max.value.w=heavy pencils
14 panl.range.min.wildcard.w=true
15 panl.range.max.wildcard.w=true

```

## Hints/Recommendations:

- Use sparingly, and where it makes sense, ranges can filter the result set easily down to zero results returned if the range of values in the documents falls slightly outside.
- If there is a large number of disparate values then a range facet may be useful, if there are only a few values, then a regular facet may suffice.
- Derived fields and ranges can also be another option for a range facet, with the dataset being used to generate static ranges and then stored in the Solr field.

## DATE Range Facets

Solr stores a date field (of `fieldType DatePointField`) and stores the date as String representations expressed in Coordinated Universal Time (UTC - i.e. `YYYY-MM-DDThh:mm:ssZ`). An example value: `1972-05-20T17:33:18Z`.

This means that when you choose this for a facet, each of the fields will be returned to the exact second without being able to be rolled up to a day, month, or year. This leads to a *very* long list of facet values, one for each of the returned result documents. Consequently Panl will not return any facetting information from Solr, however it will

add information for the configured date range to the returned JSON object. This will allow the date range to be implemented on the front end.

```

01 # <field "indexed"="true" "stored"="true" "name"="solr_date" "type"="pdate" <
      "multiValued"="false" />
02 panl.facet.S=solr_date
03 panl.name.S=Solr Date
04 panl.type.S=solr.DatePointField
05 panl.date.S.previous=previous
06 panl.date.S.next=next
07 panl.date.S.years=\ years
08 panl.date.S.months=\ months
09 panl.date.S.days=\ days
10 panl.date.S.hours=\ hours

```

Both the `panl.date.<lpse_code>.previous` and `panl.date.<lpse_code>.next` properties must be set for the DATE Range facet to be active, however they do not have to be implemented on the front-end.



**IMPORTANT:** Panl will NOT request faceting on any Date field types which means that they will not be returned in the base Solr response object, however they can be returned in the field list of Solr document results.

However, Date field types that are defined as facets within the properties file can be used to return RANGE facet values from NOW +/- a specific period.

### Hints/Recommendations:

- If there is a Solr fieldType of then `solr.DatePointField` this will ALWAYS be configured to be a DATE Range facet
- If you want to have a date range of an arbitrary timeframe - say 3 months to 6 months ago, then you will need to derive a field based on that value and then set a range for that field. For example, if you wanted to range on months then derive a field of month and set it to year \* 12 + month - i.e. 1 would be January, year 0 whilst 24295 is July 2024.

## OR Facets

OR facets usually allow an end user to increase the number of results by choosing a single facet value OR another facet value for the same facets. If there are other facets available for this selection within this facet, then they will appear.

OR facets work in conjunction with each other, if you have multiple OR facets configured for a Panl collection then they work within their specific facet, not across facets.

#TODO example

```

01 # <field "indexed"="true" "stored"="true" "name"="brand" "type"="string" <-
      "multiValued"="false" />
02 panl.facet.b=brand
03 panl.or.facet.b=true
04 panl.name.b=Brand
05 panl.type.b=solr.StrField

```

### Hints/Recommendations:

- Use or facets to increase the number of results that are returned.
- Remember that OR facets only return more facets if there are additional values within the dataset.

## Other Facet Options

### Hierarchical Facets

Hierarchical facets allow one facet to only appear if another facet (or Panl parameter) has already been selected. In the Book Store example, The book series that an Author has published will only appear if an Author facet has been selected.

See the `panl.when.<lpse_code>` section for configuration options.

### Hints/Recommendations:

- Any facet can be made hierarchical.

- This is useful when you have a lot of facets and not all of them need to appear on the search page, or when you want to guide a user through the search results (for example force the user to select a year, then a month, then a day).

## Facet Sorting

By default Solr sorts the facet results by 'count' - i.e. the number of documents that have this facet value. This can be set to 'index' which will sort on the facet value.

This is a distinct property from the facets or fields that you would want to be able to sort the result documents on. This sorts a specific facet, not the documents.

For example, in the Book Store Panl configuration the `brand` facet is configured with `panl.facetsort.A=index` which will sort the returned facets by their facet values (i.e. index). Below is an image showing the difference between the facet sorting options.

Authors (A-Z) (4)	Authors (A-Z) (4)
[add] B (2)	[add] C (11)
[add] C (11)	[add] M (6)
[add] D (3)	[add] D (3)
[add] M (6)	[add] B (2)

*Image:* Images showing the difference between sorting on index (left), and count (right).

See the `panl.facetsort.<lpse_code>` section for configuration options.

~ ~ ~ \* ~ ~ ~

# A Brief Introduction To Solr

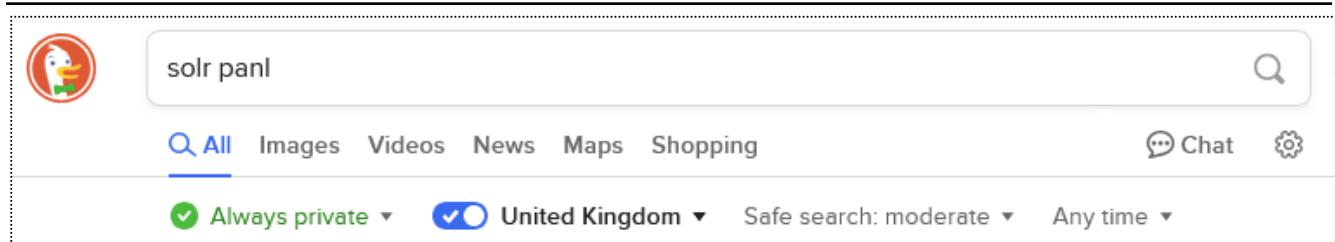
---

This chapter covers the minimal information that is required to understand the integration points between the Solr and Panl servers. Many books, articles, blog posts, sites, and information has been written and published on the Apache Solr project and server which are useful for understanding and fine-tuning the Solr search server, this chapter will not go into enough detail to even come close to replacing those resources.

## Querying Data

For any search engine, there are two ways to query the data, either through a keyword search, or through faceting.

Below is an image of the DuckDuckGo search engine which allows both a keyword search and some very simple faceting options.



*Image: A DuckDuckGo search with the keyword of 'Solr Panl' which will show relevant results and some simple faceting options.<sup>13</sup>*

Large scale web search engines attempt to index and make sense of the huge amount of data that is available on the myriad of websites.

Apart from the keyword search of 'solr panl', the following facets are available to a search:

- Country of origin (set to 'United Kingdom'),
- The safety of the search results (set to 'moderate'), and
- The time that the content was published (set to 'Any time')

---

<sup>13</sup> At the time of writing no useful results were returned by any of the large search engines for 'Solr Panl'.

The facets will help to guide the search results, but they are broad facets which are useful for this interface. **Note:** that there are advanced options to only search one site, or to search the file type as well.

With smaller and more well-defined search results, there is greater potential to add facets that help the user get to the correct details more quickly.

Individual sites use search keywords and faceting to help guide their users.

## The Solr Managed Schema

Included within the downloaded Panl release is an example managed schema file for the mechanical pencils collection.

PANL\_INSTALL\_DIRECTORY/sample/solr/mechanical-pencils/managed-schema.xml

For the Solr server version 9 - this file can also be viewed on the GitHub repository:

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/solr/mechanical-pencils/managed-schema.xml>

Whilst there are many configuration options within the schema file, this file has three major parts of interest to the Panl server, namely:

1. The schema name attribute on the top level element (`<schema name="mechanical-pencils" />`) which is configured to map to the Panl collection URI name,
2. The field definitions including the fields type (`<field />` elements) which may either be facets or fields, and
3. The field type definitions (`<fieldType />` elements) which drive validation and in place text replacements for the Panl server

In this file formatting added for readability, which means that the original schema file line numbers will not match the below:

```

01<?xml version="1.0" encoding="UTF-8" ?>
02<schema name="mechanical-pencils" version="1.6">
03  <field name="_version_" type="plong" indexed="false" stored="false"/>
04  <field name="id" type="string" indexed="false" stored="true" required="true" ←
05    multiValued="false" />
06  <field name="brand" type="string" indexed="true" stored="true" ←
07    multiValued="false" />
08  <field name="disassemble" type="boolean" indexed="true" stored="true" ←
09    multiValued="false" />
10  <field name="description" type="text_general" indexed="true" stored="true" ←
11    multiValued="false" />
12  <field name="manufacturer_link" type="string" indexed="false" stored="true" ←
13    multiValued="false" />
14  <field name="colours" type="string" indexed="true" stored="true" ←
15    multiValued="true" />
16
17  <!-- additional field definitions -->
18
19  <fieldType name="string" class="solr.StrField" sortMissingLast="true" />
20  <fieldType name="boolean" class="solr.BoolField" sortMissingLast="true"/>
21
22  <!-- additional field type definitions -->
23
24</schema>
```

**Line 1:**

Is the standard xml definition

**Line 2:**

The start of the schema definition for the mechanical-pencils data. **Note:** The schema's XML elements name attribute (i.e. `name="mechanical-pencils"`) will be used by the Panl generator as the filename for the `<panl_collection_uri>.panl.properties` file and for the property name. For the above example schema file, it will generate a properties file name `mechanical-pencils.panl.properties`, and place a property in the `panl.properties` file of

```
panl.collection.mechanical-pencils=mechanical-pencils.panl.properties
```



**IMPORTANT:** When creating a collection in the Solr server, (i.e. by having a `panl.collection.<solr_collection_name>=<panl_collection_uri>.properties_file_name` in the `panl.properties` file, the `<solr_collection_name>`) part of the property key **\_\_MUST\_\_** match the Solr collection name to connect to. The `<properties_file_name>`, may be any name, noting that the `<panl_collection_uri>` first part of the file name is the URI path that the Panl server will respond to, and **\_\_MUST\_\_** be unique amongst all URI paths registered by the Panl server.

For example, the following line in the `panl.properties` file:

```
panl.collection.mechanical-pencils=mechanical-pencils.panl.properties
```

Will be parsed as follows:

The Solr collection name of `mechanical-properties` will be taken from the property key: `panl.collection.mechanical-pencils`

- This is the Solr collection to query for results

Panl will read the configuration from the properties file

```
mechanical-pencils.panl.properties
```

and will use the first prefix of the filename to bind the Panl URI path to, for the above example the URI path would be `mechanical-properties/*`

For a property

```
panl.collection.mechanical-pencils=brands.panl.properties
```

The Panl server will still connect to the `mechanical-collections` Solr collection, but would be bound to the Panl URI path of

```
brands/*
```

Lines 3-9:

These are the field definitions, seven fields are defined in the above example, there are many more fields in the actual managed schema file.

1. `_version_` - required by Solr - this is an internal field that is used by the partial update procedure, the update log process, and by SolrCloud.
2. `id` - this is the identifier of the result, and must be unique across the collection.
3. `brand` - the field that stores the brand of the mechanical
4. `disassemble` - the field that stores whether the mechanical pencil can be easily disassembled.
5. `description` - the field that stores the description of the pencil
6. `manufacturer_link` - the field that stores the link to the manufacturer of the mechanical pencil
7. `colours` - the field that can store multiple colour values for the specific mechanical pencil.

### The impact of field definitions and field types on the Panl server

The two basic rules for Solr field definitions are:

1. If a field is indexed (`indexed="true"`) then you will be able to search on the field contents, sort by this field, and use the field contents as a facet, and
2. If a field is stored (`stored="true"`) then you will be able to return this field in the search results documents, and be able search on the field, but cannot be used as a facet (unless it is also indexed)

#### Line 11:

For brevity, additional field definitions were removed and replaced with a comment.

#### Lines 13-14:

Solr field type definitions, which the Panl generator will look for to determine how validation and prefix-suffix replacement will be done.

Note the `solr.BoolField` will also allow boolean value replacement (along with optional prefixes and suffixes).

#### Line 16:

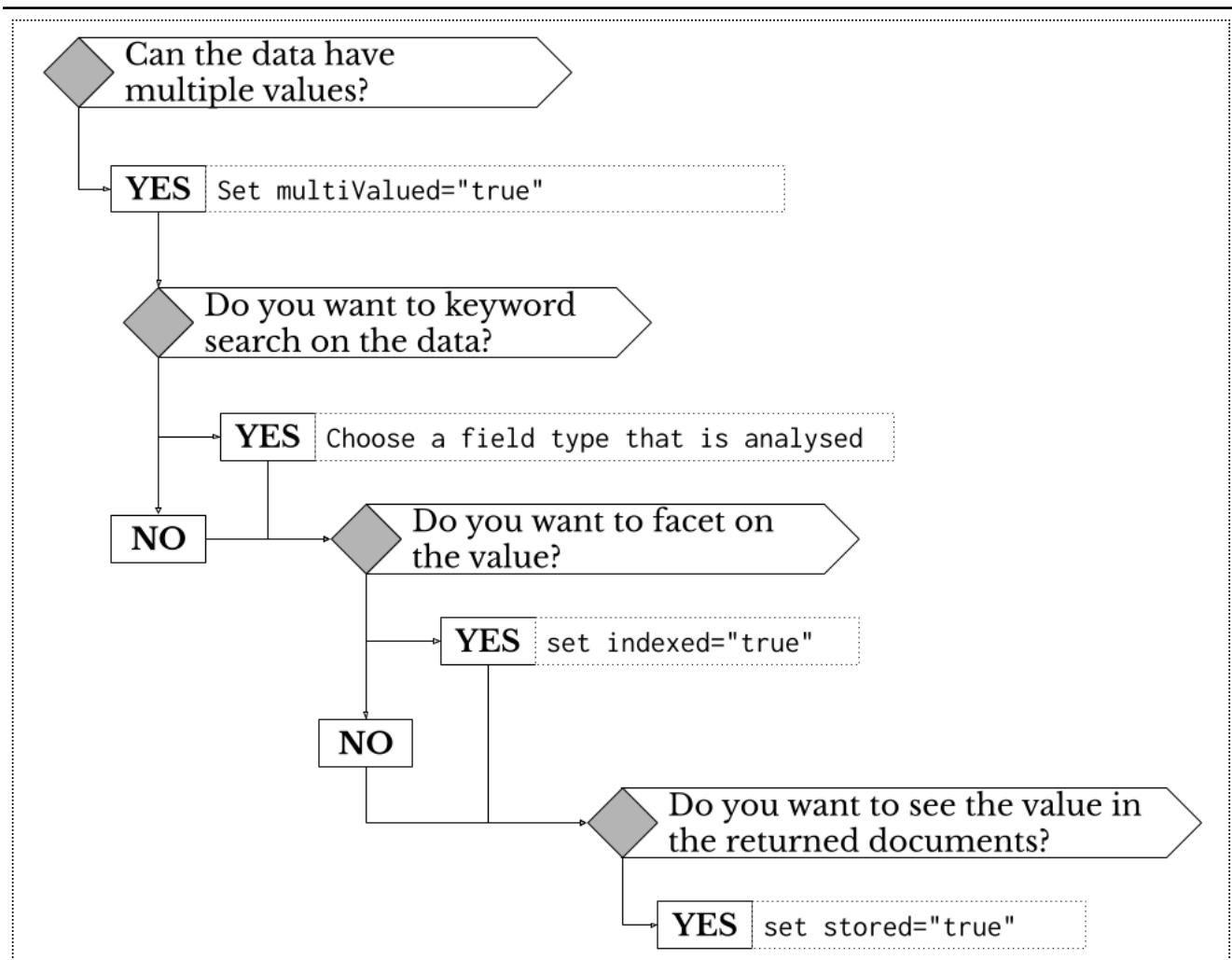
For brevity, additional field type definitions were removed and replaced with a comment.

#### Line 18:

The end of the Solr managed schema.

### Determining the Appropriate FieldType and Attributes

A quick overview of the decisions around choosing the field type, and setting the :



*Image: The decision chart for determining the field types and attributes*

There attributes for every field definition in the Solr managed schema are as follows:

- `name` - the Solr field name for this piece of data
- `type` - the type of the data to be indexed, this will determine the configuration options that are available through the Panl server. **Note:** this 'type' is used to reference the Solr field type. It is this field type that determines how it is to be indexed by Solr.
- `indexed` - whether to index the data so that it may be searched/faceted upon
- `stored` - whether the data will be stored in the Solr index and available to be returned with the results documents.
- `multiValued` - whether this field may contain more than one value

### For example:

In the sample file, the `brand` field is configured as a type of `string`

```
01 <field name="brand" type="string" indexed="true" stored="true" ↵
    multiValued="false" />
```

Further down the Solr managed schema file are the definitions of the field types which match the above `type` attribute (I.e. the `type` of the field above, matches the name of the `fieldType` attribute below). For the above field, the matching `fieldType` definition is below.

```
01 <fieldType name="string" class="solr.StrField" sortMissingLast="true" />
```

This will return the facet values for the `brand` data as they are and stored. With **NO ANALYSIS** done on the fields.

They will be displayed on the Panl Results Viewer:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname>

And return the facets values in full (including the configured prefix and suffix):

### Brand (b)

- [\[add\]](#) Manufactured by Koh-i-Noor Company (11)
- [\[add\]](#) Manufactured by Caran d'Ache Company (4)
- [\[add\]](#) Manufactured by Faber-Castell Company (4)
- [\[add\]](#) Manufactured by Pacific Arc Company (4)

- [add] Manufactured by Alvin Company (3)
- [add] Manufactured by Kaweco Company (3)
- [add] Manufactured by Rotring Company (3)
- [add] Manufactured by Hightide Penco Company (2)
- [add] Manufactured by Kita-Boshi Company (2)
- [add] Manufactured by Küelox Company (2)
- [add] Manufactured by Mitsubishi Company (2)
- [add] Manufactured by OHTO Company (2)
- [add] Manufactured by Scrikks Company (2)
- [add] Manufactured by Staedtler Company (2)
- [add] Manufactured by BIC Company (1)
- [add] Manufactured by DEDEDEPRAISE Company (1)
- [add] Manufactured by Ito-Ya Company (1)
- [add] Manufactured by Mr. Pen Company (1)
- [add] Manufactured by Muji Company (1)
- [add] Manufactured by Redcircle Company (1)
- [add] Manufactured by Unbranded Company (1)
- [add] Manufactured by WSD Company (1)
- [add] Manufactured by YStudio Company (1)

In contrast, if the `brand` field was configured as `text_general`

```
01 <field name="brand" type="text_general" indexed="true" stored="true" ↵
    multiValued="false" />
```

The `fieldType` definition includes an analyser which will break up the text into individual words and lowercase them, and ignores stopwords:

```
01 <fieldType name="text_general" class="solr.TextField" positionIncrementGap="100">
02   <analyzer type="index">
03     <tokenizer name="standard"/>
04     <filter name="stop" ignoreCase="true" words="stopwords.txt" />
05     <!-- in this example, we will only use synonyms at query time
06     <filter name="synonymGraph" synonyms="index_synonyms.txt" ignoreCase="true" ↵
          expand="false"/>
07   <filter name="flattenGraph"/>
```

```

08    -->
09    <filter name="lowercase"/>
10  </analyzer>
11
12  <analyzer type="query">
13    <tokenizer name="standard"/>
14    <filter name="stop" ignoreCase="true" words="stopwords.txt" />
15    <filter name="synonymGraph" synonyms="synonyms.txt" ignoreCase="true"←
16      expand="true"/>
17    <filter name="lowercase"/>
18  </analyzer>
19</fieldType>

```

This will break up the facet values into individual, lower-cased words and return the analysed facet values for `brand` and display the following facet list: (Note: that this configuration is not included in the sample configuration)

### **Brand (b)**

- [\[add\]](#) i (11)
- [\[add\]](#) koh (11)
- [\[add\]](#) noor (11)
- [\[add\]](#) arc (4)
- [\[add\]](#) caran (4)
- [\[add\]](#) castell (4)
- [\[add\]](#) d'ache (4)
- [\[add\]](#) faber (4)
- [\[add\]](#) pacific (4)
- [\[add\]](#) alvin (3)
- [\[add\]](#) kaweco (3)
- [\[add\]](#) rotring (3)
- [\[add\]](#) boshi (2)
- [\[add\]](#) hightide (2)
- [\[add\]](#) kita (2)
- [\[add\]](#) küelox (2)
- [\[add\]](#) mitsubishi (2)
- [\[add\]](#) ohto (2)
- [\[add\]](#) penco (2)

- [\[add\] scrikks \(2\)](#)
- [\[add\] staedtler \(2\)](#)
- [\[add\] bic \(1\)](#)
- [\[add\] dededepraise \(1\)](#)
- [\[add\] ito \(1\)](#)
- [\[add\] mr \(1\)](#)
- [\[add\] muji \(1\)](#)
- [\[add\] pen \(1\)](#)
- [\[add\] redcircle \(1\)](#)
- [\[add\] unbranded \(1\)](#)
- [\[add\] wsds \(1\)](#)
- [\[add\] ya \(1\)](#)
- [\[add\] ystudio \(1\)](#)



**Tips:** In general, the only data that you want analysed are those fields that will be used as a keyword search.

Additionally, throughout this book, all Solr fields will use a copy field for text searching, rather than having Solr search on individual fields.

## The Solr Configuration File

The Solr configuration file (`solrconfig.xml`) with comments is over 1,300 lines long, there are two parts of the configuration file that may be of interest, namely the Query Request Handler and the Highlighting.

### Query Request Handler

This is highlighted on line 795

```

789 <!-- A request handler that returns indented JSON by default -->
790 <requestHandler name="/query" class="solr.SearchHandler">
791   <lst name="defaults">
792     <str name="echoParams">explicit</str>
793     <str name="wt">json</str>
794     <str name="indent">true</str>
```

```

795   <str name="df">text</str>
796 </lst>
797</requestHandler>
```

The default field that Solr will search against if no search field is set. Panl does not set the search field and relies on this default.

In the `managed-schema.xml` file for the mechanical pencils collection, all of the fields are copied to this `text` field, which can then be searched upon.

```

51 <copyField source="brand" dest="text" />
52 <copyField source="name" dest="text" />
53 <copyField source="mechanism_type" dest="text" />
54 <copyField source="nib_shape" dest="text" />
55 <copyField source="body_shape" dest="text" />
56 <copyField source="grip_type" dest="text" />
57 <copyField source="grip_shape" dest="text" />
58 <copyField source="cap_shape" dest="text" />
59 <copyField source="category" dest="text" />
60 <copyField source="nib_material" dest="text" />
61 <copyField source="mechanism_material" dest="text" />
61 <copyField source="grip_material" dest="text" />
63 <copyField source="body_material" dest="text" />
64 <copyField source="tubing_material" dest="text" />
65 <copyField source="clip_material" dest="text" />
66 <copyField source="cap_material" dest="text" />
67 <copyField source="colours" dest="text" />
68 <copyField source="variants" dest="text" />
69 <copyField source="description" dest="text" />
```



**IMPORTANT:** Panl does not include a way to set the search fields for a phrase query and relies on the default Solr field to search upon

## Highlighting

This section starts on line 1060 and runs for just over 100 lines, it is not included in this book for brevity. The highlighting component controls what text any highlighted words will be surrounded with when they are returned with the Solr results. This doesn't impact the Panl server in any way, however if you are going to use highlighting, then there are some important considerations in order to enable this.

If highlighting is enabled in the `<panl_collection_uri>.panl.properties` file via setting the `solr.highlight=true` property, then the following rules are applied:

Highlighting requires that you have a `uniqueKey` defined in your schema. - In the `mechanical-pencils` collection, the unique key is `id` (mapped to the LPSE code of `i`)

- Panl will pass through the `hl=true` Solr query parameter and the `hl.fl=*` parameter.
- Panl will use the unified highlighter
- In the Panl Results Viewer web app, the functionality for utilising the returned highlighted results has not been implemented. The implementation is upon you, the integrator, to determine the best way to display the results.

See the Solr documentation for in-depth information:

<https://solr.apache.org/guide/solr/latest/query-guide/highlighting.html>

~ ~ ~ \* ~ ~ ~

# Panl Configuration

---

Understanding the Panl server configuration is the most important part of this book, it drives all of the features for the connection to the Solr search server and the registration of the Panl collection URIs and how fields are to be processed.

The Panl server is driven by at minimum two configuration files (both of which are java based `.properties` files<sup>14</sup>), the `panl.properties` file and one or more of a `<panl_collection_uri>.panl.properties` file.

Within these files, properties that start with the prefix of `solr` relate to configuring the Solr results, properties that start with `panl` configure how Panl asks for and parses the returned results.

Of the two files mentioned, the `<panl_collection_uri>.panl.properties` file contains the most configuration.



**Notes:** The file names used within this book are examples only. Both of the filenames can be anything that you choose, but the format of the file must be a java `.properties` file format.

For clarity and understanding, it is suggested that you name the files as per the naming used in this book.

## The `panl.properties` file

The `panl.properties` file, configures the Panl Server to

1. Define how to communicate with the Solr Server,
2. Whether to use verbose server responses,
3. Whether to enable the in-built Panl Results Viewer and Explainer, and

---

<sup>14</sup> In some instances, the properties file layout would have been better suited to JSON, however, comments are not allowed in JSON files, which makes explaining the file a lot harder. Admittedly, HJSON could have been used, and parsed on the way into Panl, but this would reduce portability - sigh - these are the decisions which can reverberate through time and code.

4. The locations for the properties files for each Panl collection, and the Solr collection to which these properties files connect to.

Once configured, the `panl.properties` file should remain relatively static. The only changes that are required to this file is when a new Solr collection, or new Panl collection is required to be integrated.

If you use the generator functionality, the well-commented templates which drive the generation can be found in the Synapticloop Panl github repository:

The `panl.properties` template:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/panl.properties.template>

## The `<panl_collection_uri>.panl.properties` file

A `<panl_collection_uri>.panl.properties` file is defined for each individual Panl collection that

1. Defines the length of LPSE code,
2. Defines the LPSE codes for the parameters for
  - a. Query,
  - b. Sort order,
  - c. Pagination,
  - d. Number of results to return,
  - e. Passthrough codes, and
  - f. Individual facets
3. Defines which facets to include,
  - a. Whether to have suffixes/Prefixes for the facet
  - b. If it is a Boolean field, whether to replace the default true/false values
  - c. Whether to designate a facet as an OR, RANGE, DATE range, or Regular facet.
  - d. Whether this facet has a hierarchy.
  - e. The sorting order of the facet, either value, or count.
4. The order of the facets within the URI path
5. The available FieldSets for each collection

If you use the generator functionality, the well-commented templates which drive the generation can be found in the Synapticloop Panl github repository:

The `panl_collection_uri.panl.properties` template:

[https://github.com/synapticloop/panl/blob/main/src/main/resources/panl\\_collection\\_uri.panl.properties.template](https://github.com/synapticloop/panl/blob/main/src/main/resources/panl_collection_uri.panl.properties.template)

~ ~ ~ \* ~ ~ ~

# The panl.properties Configuration File

---

The `panl.properties` file controls

- which Solr server to connect to,
- which client to use, and
- the bindings of the Solr collections to Panl URI paths.

By default, upon startup, the Panl Server will look for a file named `panl.properties` in the same directory that the server was started. Alternatively, you can pass through a `-properties` command line argument with the value being the location of the properties file - this will then become the base directory for **ALL** referenced properties files in the `panl.collections.<field_set>` property.

The file is included in the release package, or, for an overly commented online format:

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/panl.properties>

The `panl.properties` file defines the following properties:

- `solrj.client`  
The SolrJ client to use to connect to the Solr server, the selection of client will impact whether the `solr.search.server.url` is a single URL, or a comma separated list of URLs, or even a zookeeper comma separated list of URLs
- `solr.search.server.url`  
The URL, or comma separated URLs for the panl server to connect to
- `panl.results.testing.urls`  
Whether to enable the testing URLs which will help you understand and debug the PANL LPSE URI paths
- `panl.status.404.verbose`  
Whether to enable a verbose 404 HTTP status error message
- `panl.status.500.verbose`  
Whether to enable a verbose 500 HTTP status error message
- `panl.collection.<solr_collection_name>`  
The list of properties files to load collections and field sets to serve

## Configuring The SolrJ Connector

Set the implementation of the SolrJ connection.

```
01 #solrj.client=Http2SolrClient
02 #solrj.client=HttpJdkSolrClient
03 #solrj.client=LBHttp2SolrClient
04 solrj.client=CloudSolrClient
```



**IMPORTANT:** This is a **MANDATORY** property, and the Panl server will not start if it is missing.

Panl uses the SolrJ connector to communicate with the Solr server, which **MUST** be one of the following implementations:

- `Http2SolrClient`
- `HttpJdkSolrClient`
- `LBHttp2SolrClient`
- `CloudSolrClient`

The property `solrj.client` controls which SolrJ implementation is to be used. There is a one to one mapping of property values to Solr client values in the package `org.apache.solr.client.solrj.impl` (**NOTE:** that some of the implementations are deprecated, and are not included in the list of possible values).

All values are included in the `panl.properties` file, with all but the last commented out.

## Setting The Solr Server URL(s)

Set the URL, or URLs for the Solr server

```
01 solr.search.server.url=http://localhost:8983/solr,http://localhost:7574/solr
```



**IMPORTANT:** This is a **MANDATORY** property, and the Panl server will not start if it is missing.

The `solr.search.server.url` contains the URL, or comma separated list of URLs for the underlying Solr server to connect to.

The defaults are set in the `panl.properties` file for the example cloud as below.

```
solr.search.server.url=http://localhost:8983/solr,http://localhost:7574/solr
```

For cloud based Solr server installations (including the one in this book), you may also use the zookeeper URLs.

Zookeeper URLs **MUST** start with a `zookeeper:` prefix (including the colon :). For the implementation contained in this book the property would be set as:

```
zookeeper:http://localhost:9983
```

The Solr documentation states that

---

The ZooKeeper based connection is the ***most reliable and performant means*** for CloudSolrClient to work. On the other hand, it means exposing ZooKeeper more broadly than to Solr nodes, which is a security risk. It also adds more JAR dependencies.

*(added emphasis by the author)*

---

Testing this configuration in this book has shown that this is slower by an order of magnitude. However, your implementation, hardware, and server setup may provide different results.



**IMPORTANT:** If the CloudSolrClient is configured as the SolrJ client with zookeeper URLs then you SHOULD test the speed of results generation with and without the zookeeper

In testing for this book, the zookeeper URLs were an order of magnitude slower than using the direct Solr URLs. Testing of the two configuration types is DEFINITELY recommended.

Zookeeper configuration does have the advantage of knowing which Solr instances are available and which ones to connect to, so choose wisely.

## Enabling The Panl Results Testing URLs

Controls whether the debugging and testing URLs are made available.

01 `panl.results.testing.urls=true`

The testing URLs provide the two simple web applications to help with understanding and debugging the Panl LPSE paths and configuration.

If enabled (i.e. set to `true`), two webapps will be made available:

1. <http://localhost:8181/panl-results-viewer/>, and
2. <http://localhost:8181/panl-results-explainer/>



**IMPORTANT:** If this property is set to true, then in the unlikely event that you have a collection named `panl-results-viewer`, or `panl-results-explainer`, the results may be indeterminate.

If this property does not exist, or does not exactly (case sensitive) equal the String value `true`, then the above URLs will not be registered to respond to any queries.



**IMPORTANT:** It is recommended to set this property to `false` for production use, and the only property value that will enable this is the case sensitive value

true.

## The Panl Results Viewer

A simple<sup>15</sup> results viewer allowing you to browse all features and functionality of the Panl server.

The screenshot shows a web-based search interface for the Panl server. At the top, it displays the title "Panl" and a subtitle "a very simple results viewer". Below this, it lists "Available collections/fieldset URI Paths (CaFUPs):" with links to [/mechanical-pencils/default], [/mechanical-pencils/firstfive], and [/mechanical-pencils/brandname]. It then indicates the current view: "You are viewing collection/fieldset URI Path (CaFUP):" followed by the canonical URI. The interface includes sections for "Search Query" (with a search bar and button), "Active Filters", and "Available Filters". On the right side, there are buttons for "q.op", "Page of Showing results per page", "This page has result(s)", "Sort by", and "Show per page". A note at the bottom states: "This page is designed to showcase the Panl LPSE encoding mechanism, with a minimal implementation. It is NOT a showcase of web technologies :)" and "Happy searching".

**Image:** The In-Built Panl Results Viewer web app showing the list of available collection/fieldset configurations

<sup>15</sup> So this started off as a simple way to test the Panl configuration and how it interacted with the Solr search server, over time, it became a little more complex. It also became incredibly useful in adding features to the returned JSON object so that integration and implementation became a more developer friendly experience.

The decision was made to enhance the ease of the developer integration experience at the cost of slightly increasing the returned payload.

This page will also `console.log` debugging information and the returned objects to the Browser console.

## The Panl Results Explainer

A simple<sup>16</sup> results explainer detailing how a Panl LPSE URI is parsed and decoded. It also provides an overview of the configuration for the specific collection.

Panl  
*a very simple results explainer*

Available collections/fieldset URI Paths (CaFUPs):  
[\[/mechanical-pencils/default\]](#) [\[/mechanical-pencils/firstfive\]](#) [\[/mechanical-pencils/brandandname\]](#)

You are viewing collection/fieldset URI Path (CaFUP):

Enter the Canonical URI path below:

Explain

*This page is designed to showcase the Panl LPSE encoding mechanism, with a minimal implementation. It is NOT a showcase of web technologies :)  
Happy searching*

**Image:** The In-Built Panl Results Explainer web app showing the list of available collection/fieldset configurations

## Setting HTTP Status Message Verbosity

Controls the message field of the JSON HTTP status response.

```
01 panl.status.404.verbose=false
02 panl.status.500.verbose=false
```

If set to false, the JSON object response will only ever have three keys:

1. `error` - will always be `true`
2. `status` - will be either `404`, or `500`

---

<sup>16</sup> Once again, a simple explainer turned into a more complex application as time and requirements became more involved.

3. `message` - will be either "Not found" or "Internal server error"

```

01 {
02   "error":true,
03   "status": <status_code>,
04   "message": <message>
05 }
```



**IMPORTANT:** It is recommended to set both of these properties to `false` for production use

## Binding Solr collections to Panl URI paths

The `panl.collection.<solr_collection>` property controls which Solr collections to bind and which Panl URI paths to serve them on.

<code>panl.collection.</code>	<code>&lt;solr_collection&gt;</code>	=	<code>path/to/file/</code>	<code>&lt;panl_collection_uri&gt;</code>	<code>.panl.properties</code>
This is the property key prefix	The Solr collection		File path	Filename prefix	The extension

- The property key prefix - The panl server will look for this prefix to load properties files
- The Solr collection is the search index to connect to on the Solr search server.
- File path - the file path is relative to the `panl.properties` file
- Filename prefix - This is the URI path that the Panl server will bind to
- The extension does not have to be `.panl.properties` - it may be anything provided that the file format is a Java `.properties` format

In the example `panl.properties` file, the property is set as:

```
panl.collection.mechanical-pencils=src/dist/sample/panl/mechanical-pencils.panl.properties
```

Which will serve up the results from the Mechanical Pencils Solr search collection on the Panl URI path of `/mechanical-pencils/*`.

~ ~ ~ \* ~ ~ ~

# The <panl\_collection\_uri>.panl.properties Configuration File

The `<panl_collection_uri>.panl.properties` file defines which Solr fields will be available on this URI path, how they are configured, sorted, and included in the facets and results.

The `mechanical-pencils.panl.properties` file is used as the reference material for the configuration options explained in this section. The file is included in the release package, or, for an overly commented online format:

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/mechanical-pencils.panl.properties>

 **Notes:** the `<panl_collection_uri>` part of the file name is the name of the Panl server URI path to bind to and the configuration properties for this URI path. In this book we will be using the Solr collection of `mechanical-pencils`, bound to a URI path of `/mechanical-pencils/*` consequently the properties file is named: `mechanical-pencils.panl.properties`.

## Parameter and Operand Definitions

### panl.param.query

The LPSE code for the search query from the user - i.e. for the HTML form below:

```

01 <form method="GET">
02   <label><input type="text" name="search" /></label>
03   <button type="submit">Search</button>
04 </form>

```

If the user enters a search query of `hexagonal` - this will be sent to the Panl server as

`/mechanical-pencils/brandandname/?search=hexagonal`

Which will generate a canonical URI path as - note the LPSE code is 'q'.

```
/hexagonal/q/
```



**Notes:** You can set the parameter key that Solr will use for the search query with the `panl.form.query.respondto` property (which is 'q' by default).

## panl.param.sort

The sort fields and the order for the results to be returned. The sorting of the results is all contained within the LPSE URI path part, and is encoded by:

1. The sort LPSE code - i.e. 's'
2. The facet or field LPSE code e.g. 'b', or 'N'
3. The sort order, '-' for descending, '+' for ascending

For the following URI path part

```
/mechanical-pencils/brandandname/sb-sN+/
```

The results will be sorted by:

1. `brand` (i.e. the 'b' LPSE code) in descending order (the '-' sort order code), then by
2. `name` (i.e. the 'N' LPSE code) in ascending order (i.e. the '+' sort order code).



**Note:** The sort LPSE code does not have a URI path part.

## panl.param.page

The page number of the results to be shown on the page, which also allows both a prefix and suffix. To define a prefix, configure the `panl.param.page.prefix` property, to define a suffix, configure the `panl.param.page.suffix` property.

With the following properties defined:

```

01 panl.param.page=p
02 panl.param.page.prefix=page-
03 panl.param.page.suffix=-shown

```

The generated URI path part will be generated as:

```
/mechanical-pencils/brandandname/page-1-shown/p/
```

The Solr search server does not have an implicit idea of pagination, it has a start parameter which is the offset of the number of documents and a numrows parameter which determines the number of documents to return for a query.

The Panl server translates this, in conjunction with the `panl.paramn.numrows` property, to the correct starting offset for the documents in the Solr Query. In effect the number of results per page (numrows) multiplied by the page number sets the start parameter for the Solr search query.

Additionally, the Panl server validates the passed in page number to ensure that it is a positive integer value. If the value cannot be parsed or validated, the page number is set to 1 (i.e. the Solr start query parameter is set to 0).

### **Example:**

The following will return all 55 results, on page 1 and showing 10 results per page

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/page-1/10-per-page/pn/>

With a Solr query start parameter of 0 (i.e. `start=0`) which is expected as the offset is 0 from the start of the results. For the second page

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/page-2/10-per-page/pn/>

The Solr query has a start parameter of 10 (i.e. `start=10`). In effect the start parameter is calculated by Panl as `(page_number - 1) * num_rows`.

### **panl.param.numrows**

The number of results shown on the page, which also allows both a prefix and suffix. To define a prefix, configure the `panl.param.numrows.prefix` property, to define a suffix, configure the `panl.param.numrows.suffix` property.

```

01 panl.param.numrows=n
02 panl.param.numrows.prefix=display-
03 panl.param.numrows.suffix=-per-page

```

The generated URI path part will be generated as:

```
/mechanical-pencils/brandandname/display-1-per-page/n/
```

### **panl.param.query.operand**

This LPSE code defines the query operand to be passed through to the Solr server, and only has one of two possible values. Like the sort LPSE code, this query operand LPSE code does not have a URI path part.

The query operand URL parameter that the Solr server responds to to determine whether the search term should be found in any one of the Solr indexed fields (OR), or all of the fields (AND).

The following URI Path part will search for the 'hexagonal' keyword and return results that are in in ALL of the indexed search fields

```
/mechanical-pencils/brandandname/hexagonal/qo+/
```

This URI path below will search for the 'hexagonal' keyword and return results that are in in ANY of the indexed search fields

```
/mechanical-pencils/brandandname/hexagonal/qo-/
```



**Note:** The query operand LPSE code does not have a URI path part

By default the query operand is set to OR as it will return the most results.

### **panl.param.passthrough**

This LPSE code is used purely for generating SEO friendly URLs and is not passed through to the Solr Search server. The passthrough parameter should be used with

other parameters that help to filter the results.

```
01 panl.param.passthrough=z
```

For example, the following URI path parts will return exactly the same results (in fact all of the results as there are no other facets applied):

```
/lots-of-mechanical-pencils/z/
```

```
/the+best+mechanical+pencils/z/
```

The above two paths would generate the same canonical URL of

```
/
```

However, you could also define multiple URI path parts to return pencils manufactured by BIC with the following URI path.

```
/the+finest+selection+of+BIC+mechanical+pencils/BIC/zb/
```

Note that the above URI path could also have been generated by using a prefix and suffix without the trailing /BIC/, however this would have meant that if this was set to an OR facet, then the length of the URI path would grow very quickly.



**Note:** The passthrough LPSE code will \_\_NOT\_\_ be included in the canonical URI generation unless the `panl.param.passthrough.canonical` is set to true. (see below).

## panl.param.passthrough.canonical

Whether the LPSE code for a passthrough parameter will be used to generate the canonical URL.

```
01 panl.param.passthrough.canonical=false
```

If set to `true`, the URI path value will still not be used as a filter for the Solr query, it will not be sent to the Solr server, and will therefore not affect the results, HOWEVER it will be part of the canonical URI path.

This allows the URI path to have arbitrary text for readability without affecting the outcome.

For example, if the value of this property (i.e. the LPSE code) is set to `z` then the following URI paths will return the **exact same** search results.

```
/cuddly-brown-teddy-bears/z/
```

```
/motor-vehicles-with-three-wheels/z/
```

For the above URI paths, the generated canonical URI paths will (respectively) be

```
/cuddly-brown-teddy-bears/z/
```

```
/motor-vehicles-with-three-wheels/z/
```

This can be useful when generating a SEO friendly URL which doesn't require any more facetting. For example if a URL is generated that references a unique ID of a product (which will only return one result) this mechanism can be used to ensure a friendly canonical URI path.

For example, the URI

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/53/i/>

Will return only one result - the Staedtler 925 pencil.

The URI path of `/53/i/` is not that exciting, so a passthrough parameter of `Staedtler-925-Metal-Pencil-in-blue,black,or silver colours` could be added for more informational URI paths.

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Staedtler-925-Metal-Pencil-in-blue,black,or+silver+colours/53/zi/>

```
/Staedtler-925-Metal-Pencil-in-blue,black,or+silver+colours/53/zi/
```

This would return the same result, however the canonical URI path would be:

```
/53/page-1/10-per-page/ipn/
```

With this `panl.param.passthrough.canonical` property set to `true`, the canonical URI path becomes

```
/Staedtler-925-Metal-Pencil-in-blue%2Cblack%2Cor+silver+colours/53/page-1/10-per-page/zipn  
/
```

This is done on a wide range of large sites, including e-commerce sites to boost SEO ratings.



**Tips:** It is recommended to have a separate CaFUPs file for each of these usages as there is no need to define or return any facets for this page as there is only one result.

## panl.form.query.respondto

When the user submits a search query, this is the URL parameter key that Panl will use as the search phrase. With the property set to '`search`' in the following line

```
panl.form.query.respondto=search
```

```
01 <form method="GET">  
02   <label><input type="text" name="search" /></label>  
03   <button type="submit">Search</button>  
04 </form>
```

For the above HTML form, Panl will use the URL parameter '`search`' value as the keyword search. This does not affect the LPSE code, which is set to '`q`' for URL generation.

## panl.include.single.facets

Facets that only include a single result to further refine the query will not be included by default. The reasoning behind this is that having a list of facets with only one result will not refine the query at all, it will simply return the same set of results, just with a longer LPSE URI. You may wish to include these results for a more verbose URL and possibly better search engine visibility.

The default value for this property is `false`, so you do not need to include it unless you wish to enable this feature, i.e.

```
panl.include.single.facets=true
```

### **panl.include.same.number.facets**

Facets that include the same number of results as the number of documents that are returned will not be included by default. The reasoning behind this is that if, by using this facet, you will get exactly the same results, then this is not a refinement of the query at all, it will simply return the same set of results, just with a longer LPSE URI.

The default value for this property is `false`, so you do not need to include it unless you wish to enable this feature, i.e.

```
panl.include.same.number.facets=true
```

#### **For example:**

The link:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Manufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/bWGL/>

Returns the following in the available filters section

#### **Colours (W)**

- [\[add\]](#) Purple (2)
- [\[add\]](#) Yellow (2)

**Note:** that there are no Blue or Red facet values returned



**Notes:** For the links below, this property is set to `false` in the provided example file, you will need to manually edit the `mechanical-pencils.panl.properties` file and set this to `true` for the example to work.

With this property set to true (i.e. `panl.include.same.number.facets=true`) then the results

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Manufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/bWGL/>

Show 3 results, and include within the available filters section

## Colours (*W*)

- [\[add\]](#) Blue (3)
- [\[add\]](#) Red (3)
- [\[add\]](#) Purple (2)
- [\[add\]](#) Yellow (2)

Note that the values of the Colours Blue (3) and Red (3) are both 3, which match the number of returned documents.

Clicking on either of those (or both in either order) will generate URI paths which return identical results:

- For Blue:  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma nufactured+by+Koh-i-Noor+Company/Green/Blue/Cylindrical+Grip/120mm/bWWGL/>
- For Red:  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma nufactured+by+Koh-i-Noor+Company/Green/Red/Cylindrical+Grip/120mm/bWWGL/>
- For both:  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma nufactured+by+Koh-i-Noor+Company/Green/Red/Blue/Cylindrical+Grip/120m m/bWWWGL/>

All of the above three links add additional LPSE path encodings, without filtering the results any further, thus offering little value to end-users who would like to refine their results further.

Should you wish the above scenario to be the case, then set this value to `false`.

## `solr.default.query.operand`

This maps to the Solr q.op parameter, by default it is OR which - i.e. the search query will need to be in one of the search fields. This is only used for keyword searches, not facets.

## **solr.facet.limit**

The maximum number of facet values to return for any particular facet.

## **solr.facet.min.count**

The minimum facet count to return, if the facet count is less than this, then the facet value will not be returned. By default, this is set to 1.

## **solr.numrows.default**

The number of results to return for the search. By default this is 10, but may be any positive integer.

## **solr.highlight**

Whether to return highlighting information in the JSON response object. If set to true an additional JSON Object will be added to the root JSON object named 'highlighting'. This object then has key value pairs keyed on the 'id' Solr field with the value of the Solr field where this highlighting occurs.

## **Facet Field Definitions**

Facets (along with the query parameter) refine the search results, however, unlike the query parameter, there are additional configuration items that will affect the URI path part that is displayed to the user.

A Panl facet field is configured by setting a key within the properties file and will \_\_ALWAYS\_\_ have the form of:

```
panl.facet.<lpse_code>=<solr_field_name>
```

Where:

- <lpse\_code> is the assigned LPSE code as either a letter or a number. The number of characters (either letters or numbers) in the <lpse\_code> \_\_MUST\_\_ match the value of the panl.lpse.length property.

For example: If the property panl.lpse.length=1, then each <lpse\_code> must be 1 alphanumeric character long, if panl.lpse.length=2, each <lpse\_code> must be 2 alphanumeric characters long.

- `<solr_field_name>` is the case-sensitive name of the field that is defined in the Solr search schema configuration. This **MUST** match the Solr field name.



**IMPORTANT:** Any property in the `<panl_collection_uri>.panl.properties` file that starts exactly with `panl.facet.` is configured to be a facet field in Panl.

Panl supports the following subtypes of facets

1. Regular Facet
2. BOOLEAN Facet
3. OR Facet
4. RANGE Facet
5. DATE Range Facet

## Properties Available for Facets

The following properties are available for any facet that is defined, however there are additional properties available depending on the type of the Solr field, and whether the field is configured to be a RANGE, OR, BOOLEAN, or DATE Range facet.

- `panl.facet.<lpse_code>=<solr_field_name>`

**MANDATORY** property to indicate to Panl that the named Solr field is to be used as a faceted result in Panl.

If this property does not exist, then this Solr field and LPSE code will not be registered, and all other properties will be ignored.

- `panl.name.<lpse_code>`

Optional property to display a nicer name than the Solr field name.

If this property does not exist, or is a blank or empty string, then the `<solr_field_name>` from the above property will be used instead.

- `panl.type.<lpse_code>`

Optional property of the base Solr field type, this property is set automatically by the Panl configuration generator.

This property drives the validation of incoming requests, and if the base Solr field type is `solr.BoolType`, this will enable the `panl.bool.<lpse_code>.true` and `panl.bool.<lpse_code>.false` property lookups.



**IMPORTANT:** This property should NEVER change, UNLESS the underlying Solr managed schema has changed.

- `panl.prefix.<lpse_code>`

Optional property of arbitrary text value to prefix the Solr field value with.

For example, if this property is `panl.prefix.<lpse_code>=This is the prefix` then the display of this value in the URI (encode) would be:

```
/This+is+the+prefix+<solr_field_value>/
```

This property value is also returned in the response object for optional use when displaying the list of active and available facets.

- `panl.suffix.<lpse_code>`

Optional property of arbitrary text value to prefix the Solr field value with.

For example, if this property is `panl.suffix.<lpse_code>=\ suffixed here` then the display of this value in URI encoded form would be:

```
/<solr_field_value>+suffixed+here/
```

This property value is also returned in the response object for optional use when displaying the list of active and available facets.

- `panl.or.facet.<lpse_code>`

Optional property which enables this facet to be an OR facet - i.e. multiple values can be selected for this facet.

This property is useful if you want users to be able to select a facet value, or another facet value.

For example, you may want to allow users to be able to see pencils that were manufactured by Caran d'Ache AND Faber-Castell. The normal functionality would be that the user sees either of the brands, not both.



**IMPORTANT:** Unlike other facet selections, this will **INCREASE** the number of documents returned from the Solr server.

- `panl.when.<lpse_code>=<comma_separated_list_of_lpse_codes>`

Referred to as a hierarchical facet, this property defines whether this property will be shown. This facet will only be returned by the Panl server if one of the LPSE codes in the `<comma_separated_list_of_lpse_codes>` has already been selected.

- `panl.facetsort.<lpse_code>=<index_or_count>`

By default Solr will return the facets sorted by the count, if you would like Solr to return the results by index (i.e. the value) then set this property.

## Regular Facet

A regular facet allows you to select a specific value to refine your search results. The returned results will include matches of this value In the Solr field.

Prefixes and suffixes are available for regular facets, with the following URI paths returning equivalent results, depending on whether a prefix, a suffix, neither, or both are configured.

- /Caran+d'Ache/b/ - Neither prefix nor suffix configured
- /Manufactured+by+Caran+d'Ache/b/ - Only a prefix is configured
- /Caran+d'Ache+Company/b/ - Only a suffix is configured
- /Manufactured+by+Caran+d'Ache+Company/b/ - Both a prefix and suffix is configured

For this example, the brand Solr field will be used with a LPSE code of b. The definition in the `mechanical-pencils.panl.properties` file is as follows:

```

01 # <field "indexed"="true" "stored"="true" "name"="brand" "type"="string" ←
      "multiValued"="false" />
02 panl.facet.b=brand
03 panl.name.b=Brand
04 panl.type.b=solr.StrField
05 panl.prefix.b=Manufactured by
06 panl.suffix.b=\ Company

```

## Example

---

**Brand (b)**

- [add] Manufactured by Koh-i-Noor Company (11)
- [add] Manufactured by Caran d'Ache Company (4)
- [add] Manufactured by Faber-Castell Company (4)
- [add] Manufactured by Pacific Arc Company (4)
- [add] Manufactured by Alvin Company (3)
- [add] Manufactured by Kaweco Company (3)
- [add] Manufactured by Rotring Company (3)
- [add] Manufactured by Hightide Penco Company (2)
- [add] Manufactured by Kita-Boshi Company (2)
- [add] Manufactured by Küelox Company (2)
- [add] Manufactured by Mitsubishi Company (2)

*Image:* The Panl Results Viewer web app showing the facet with the prefix and suffix applied.

---

The text **Brand (b)** is the name of the available facet, (derived from the `panl.name.b=Brand` property) and the `<lpse_code>`.

For each of the available values, the **[add]** link is displayed, followed by the prefixed and suffixed value, with the number of results in parentheses.

Clicking on any of the links will refine the search by only including those documents which have the particular brand.

The URI path

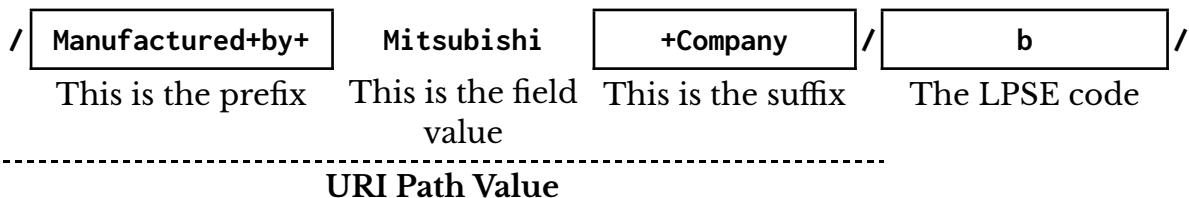
/Manufactured+by+Mitsubishi+Company/b/

Will have the prefix and suffix removed to return the result `Mitsubishi` which will then be passed on to the Solr search server.

## Decoding the URI

Without too many configuration options for a regular facet, the URI is straightforward to decode.

1. The LPSE code is looked up to resolve to the brand Solr facet field
2. The prefix and suffix is stripped from the URI path value
3. The field value is then passed through to the Solr query



**IMPORTANT:** If the prefix and/or suffix does not EXACTLY match the defined values, then this value is marked as invalid and IS NOT passed through to the Solr search query.

## BOOLEAN Facet

A BOOLEAN facet works exactly as per a regular facet, however, by definition, there are only two values that it can be, either true, or false. You select a value to refine your search results to return only those documents which contain this value. For this example, the brand Solr field will be used with a LPSE code of `D`. The definition in the `mechanical-pencils.panl.properties` file is as follows:

```

01 # <field "indexed"="true" "stored"="true" "name"="disassemble" "type"="boolean" <-
  "multiValued"="false" />
02 panl.facet.D=disassemble
03 panl.name.D=Disassemble
04 panl.type.D=solr.BoolField
05 panl.bool.D.true=able to be disassembled
06 panl.bool.D.false=cannot be disassembled

```

## Additional Properties

- `panl.bool.<lpse_code>.true`

Optional property that defines the replacement value for a 'true' value from the Solr server.

For example, if this property is `panl.bool.<lpse_code>.true=Has an extended warranty` and the returned Solr field value is 'true' then the display of this value in URI encoded form would be:

`/Has+an+extended+warranty/D/`



**IMPORTANT:** This property is only used if the property `panl.type.<lpse_code>=solr.BoolField`, in all other cases these fields will be ignored.

- `panl.bool.<lpse_code>.false`

Optional property that defines the replacement value for a false value from the Solr server.

For example, if this property is `panl.bool.<lpse_code>.false>No warranty` and the returned Solr field value is 'false' then the display of this value in URI encoded form would be:

`/No+warranty/D/`



**IMPORTANT:** This property is only used if the property `panl.type.<lpse_code>=solr.BoolField`, in all other cases these fields will be ignored.

## Example

Disassemble (D)
<a href="#">[add]</a> able to be disassembled (45)
<a href="#">[add]</a> cannot be disassembled (10)

*Image:* The Panl Results Viewer web app showing the BOOLEAN facet with value replacement.

The text **Disassemble (D)** is the name of the available facet, (derived from the `panl.name.D=Disassemble` property) and the `<lpse_code>`.

For each of the available values, the [\[add\]](#) link is displayed, followed by the boolean replacement value, with the number of results in parentheses.

The URI path

`/able+to+be+disassembled/D/`

Will look up the true/false values and if it matches, will pass on this true/false value on to the Solr search server.



**IMPORTANT:** If the value does not exactly match the true or false replacement values, then this will be marked as invalid and will not be passed through to the Solr search server.

## Decoding the URI Without a Prefix or Suffix

In the case of a true value, the URI path would be:

1. The LPSE code is looked up to resolve to the disassemble Solr facet field
2. The value is looked up and if it exactly matches
3. The field value is then passed through to the Solr query

/ **able+to+be+disassembled** / **D** /

This is a boolean replacement value that will be looked up - in this case, it is the true value

The LPSE code

#### URI Path Value

In the case of a false value, the URI path would be and would go through the same decoding process as the true value.

/ **cannot+be+disassembled** / **D** /

This is a boolean replacement value that will be looked up - in this case, it is the false value

The LPSE code

#### URI Path Value

## Defining a Prefix and Suffix

Prefixes and suffixes are also available for a boolean facet. Lines 5 and 6 of the above configuration (shown below)

```
05 panl.bool.D.true=able to be disassembled
06 panl.bool.D.false=cannot be disassembled
```

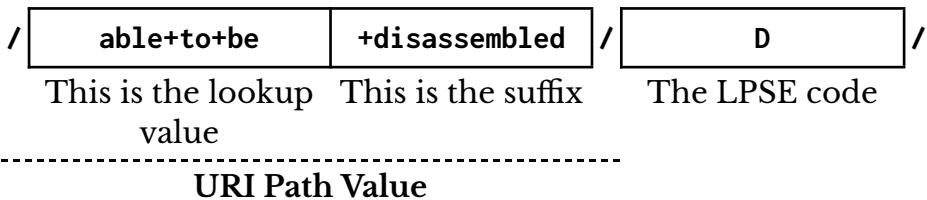
Could be replaced with the following three lines, and would have an identical effect, however the decoding of the URI path value would be different.

```
04 panl.bool.D.true=able to be
05 panl.bool.D.false=cannot be
06 panl.suffix.D=\ disassembled
```

## Decoding the URI With a Prefix or Suffix

1. The LPSE code is looked up to resolve to the disassemble Solr facet field
2. The suffix is removed from the URI path value
3. The lookup value is checked and if found, and exactly matches the defined property, then this token is marked as valid

- If the token is valid, then field value is then passed through to the Solr query



**IMPORTANT:** If the prefix and/or suffix and/or lookup for the boolean values do not exactly match the defined values, then this value is marked as invalid and not passed through to the Solr search query.

## OR Facet

An OR facet allows the user to select multiple values for the same facet, where a document only has a single value for this facet. This is different to a multivalued Solr field in a multi valued field, as the name would suggest, has multiple facet values for a specific document.

### Additional Properties

- `panl.or.facet.<lpse_code>=<true_or_false>`

This configures Panl to use this facet as an OR facet. By default it is set to '`false`', setting this property to '`true`' enables this facet as an OR facet.

## The Difference Between Multi-valued Facets and OR Facets

### Multi-values Facets

Multi-valued facets are Solr field definitions which may have multiple values for a specific field. When selected, the multi-valued facet field may still return values, provided that, within the selected documents, they also have other facet values that could further refine the results.

As an example,

- a pencil may come in a range of colours.
- it may a range of specific variants,

- it may be suitable for a wide range of applications

In the colours field definition for the mechanical-pencils schema is multivalued, each pencil can have one or more colours.

```
01 # <field "indexed"="true" "stored"="true" "name"="colours" "type"="string" <
      "multiValued"="true" />
```

Consider pencils that have a "Brown" colour:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Brown/W/>

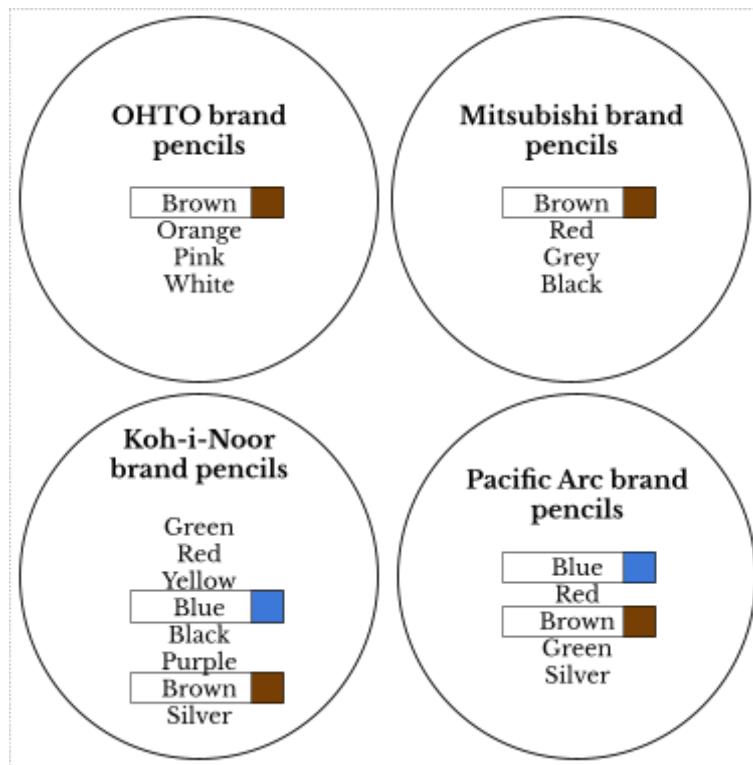
Which returns 4 results (Manufacturer // Pencil model):

1. OHTO // Maruta
2. Koh-i-Noor // 5217
3. Mitsubishi // Uni
4. Pacific Arc // Tech Pro

There are still additional colours to further facet by:

- Red (3)
- Black (2)
- Blue (2)
- Green (2)
- Orange (1)
- Pink (1)
- White (1)
- Yellow (1)

Which means that within the brands and model names of the pencils, some of the pencils within the four results also come in additional colours than just the 'Brown' that was selected.



**Image:** A selection of four brands and their associated colours - with at least one model within the Brand that comes in a 'Brown' colour

If the colour 'Blue' is then further selected:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Brown/Blue/WW/>

Only two results will be returned:

1. Koh-i-Noor // 5217
2. Pacific Arc // Tech Pro

Now we have a list of pencil models that come in both 'Brown' and 'Blue' colours (e.g. you could purchase a Koh-i-Noor pencil, model 5217 in one of the two colours).

Each selection of the multi-valued attributes on the document further refines the search criteria - the selection selects documents (i.e. mechanical pencils) that come in both 'Blue' AND 'Brown' colours.

Contrast this with an OR facet, which, as the name suggests would make this an OR query between the facet.

## OR Facets

OR facets are based on Solr field definitions which can only have singular values - i.e the field definition has the attribute of `"multiValued"="false"`. For example,

- a pencil can only be manufactured by a single company,
- a pencil will only weight one value,
- it will only be of a certain length,
- etc.

And then allow the user to select another value from this single-valued facet.



**IMPORTANT:** For the links in this section to work, you will also need to have configured the `panl.properties` file to include the `mechanical-pencils-or.panl.properties` in the sample directory.

```
01 # <field "indexed"="true" "stored"="true" "name"="brand" "type"="string" <
      "multiValued"="false" />
```

If you select a facet from a single value Solr field, then it will only return documents that contain this value. As it is not multi-valued, this facet will no longer be returned with any values from the Solr server.

For example, a pencil may only be manufactured by one company, so if you were to select the manufacturer 'Faber-Castell', only pencils that were manufactured by this brand would appear and no other brands would be returned in the facet, as a pencil may only be manufactured by a single company.

However, in some instances, you may want to increase the number of documents that are returned by allowing the user to select multiple manufacturers, this is where an OR

facet can be used.



**Notes:** The example below is not from the `mechanical-pencils.panl.properties` file, but is from a separate file, edited so that the prefix and suffix have been removed for clarity

If you want to allow users to be able to see pencils from multiple manufacturers - e.g. pencils that were manufactured by Caran d'Ache and Faber-Castell. (*The normal functionality would be that the user would only see either of the brands, not both.*)

```

01 # <field "indexed"="true" "stored"="true" "name"="brand" "type"="string" ←
      "multiValued"="false" />
02 panl.facet.b=brand
03 panl.or.facet.b=true
04 panl.name.b=Brand

```

## Example

<http://localhost:8181/panl-results-viewer/mechanical-pencils-or/firstfive>

A screengrab of the 'Brand' and 'Colours' facets are shown below. As the brand Solr field was configured to be an OR facet, the JSON object returns a key of `is_range_facet`, and the Panl web app is configured to suppress the rendering of the facet count for this facet.



**Notes:** The count of the Brand facets are still available, they are just suppressed in the rendering of the page by the Panl results viewer web application. This is done as when no brand is selected, the count of the values for the facet would normally indicate to the user how many pencils of each brand there are.

However, when one Brand facet is selected, the values for all other Brand facets will display 0 (zero) - which makes sense, as you cannot have a pencil that belongs to two Brands at once.

With no query or facets added, there are 55 results in the results set.

---

Search Results - Found 55 result(s) (exact).

<b>Brand (b)</b>	<b>Colours (W)</b>
[add] Koh-i-Noor	[add] Black (31)
[add] Caran d'Ache	[add] Blue (24)
[add] Faber-Castell	[add] Red (19)
[add] Pacific Arc	[add] Green (17)
[add] Alvin	[add] Silver (10)
[add] Kassco	[add] Yellow (10)

*Image:* The Brand and Colours facets, showing all Brands (without the facet count) and Colours.

---

### Selecting the first OR facet value

<http://localhost:8181/panl-results-viewer/mechanical-pencils-or/firstfive/Koh-i-Noor/b/>

Upon selecting a brand (in this instance 'Koh-i-Noor'), the Colours facet will now show the values for all mechanical pencils for this Brand - Note that the colour Black has now only 6 values (as opposed to the 31 available above).

There are only 11 results (out of the 55) that are of the selected brand.

---

Search Results - Found 11 result(s) (exact).

<b>Brand (b)</b>	<b>Colours (W)</b>
[add] Alvin	[add] Green (8)
[add] BIC	[add] Red (8)
[add] Caran d'Ache	[add] Yellow (8)
[add] DEDEDEPRAISE	[add] Blue (7)
[add] Faber-Castell	[add] Black (6)
[add] Huetastic Pencils	[add] Purple (5)

*Image:* Selecting the first facet 'Koh-i-Noor'

---

Normally, the 'Brand' facet would not be returned and consequently not displayed. As this is set to an OR facet, the user could now select another brand to broaden the search.

### Selecting the second OR facet value

<http://localhost:8181/panl-results-viewer/mechanical-pencils-or/firstfive/Koh-i-Noor/Alvin/bb/>

---

Search Results - Found 14 result(s) (exact).

<b>Brand (b)</b>	<b>Colours (W)</b>
[add] BIC	[add] Blue (10)
[add] Caran d'Ache	[add] Green (9)
[add] DEDEDEPRAISE	[add] Red (9)
[add] Faber-Castell	[add] Yellow (9)
[add] Hightide Penco	[add] Black (6)
[add] Koh-i-Noor	[add] Purple (5)

*Image: Selecting the second facet 'Alvin'*

---

Note that the number of results has increased (from 11 to 14) and the count of the facet values has also increased (the colour Black remains the same, however the Blue count has increased from 7 to 10).

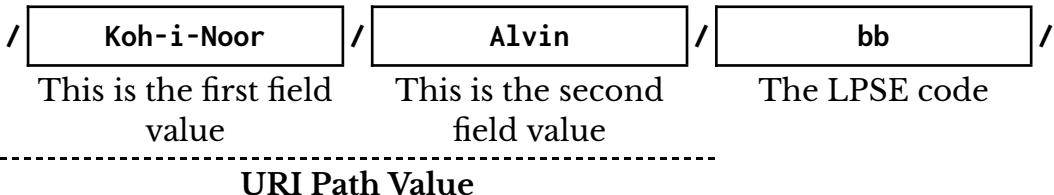
In effect, the query for Solr is requesting any result documents that have the Brand of 'Koh-i-Noor' OR 'Alvin'.

## Decoding the URI

/Koh-i-Noor/Alvin/bb/

1. The LPSE codes are looked up to resolve to the `brand` Solr facet field - there are two, both `b`
2. The first and second values are retrieved
3. The fields are passed through to the Solr server as an OR query:

`fq=brand:("Koh-i-Noor"+OR+"Alvin")`



Tips: See [The "available.facets" JSON Object](#) - Implementation notes section for details about rendering the information and generating URI paths

## RANGE Facet

RANGE facets are the most complex of facets with the most configuration options.

A range facet selects an inclusive range of values to refine the search results such that the results only return those documents which contain a value that lies within this range.



**Notes:** RANGE facets are ALWAYS returned in every query, additionally, the Panl server WILL return the regular facet for the range value if the regular facet is available

This allows you to choose whether to display the range facet or the regular facet.

For this example, the `weight` Solr field will be used with a LPSE code of `w`. The definition in the `mechanical-pencils.panl.properties` file is as follows:

```

01 # <field "indexed"="true" "stored"="true" "name"="weight" "type"="pint" <
02   "multiValued"="false" />
03 panl.facet.w=weight
04 panl.name.w=Weight
05 panl.type.w=solr.IntPointField
06 panl.suffix.w=\ grams
07 panl.range.facet.w=true
08 panl.range.min.w=10
09 panl.range.max.w=50
10 panl.range.prefix.w=weighing from
11 panl.range.infix.w=\ to
12 panl.range.suffix.w=\ grams
13 panl.range.min.value.w=from light
14 panl.range.max.value.w=heavy pencils
15 panl.range.min.wildcard.w=true
16 panl.range.max.wildcard.w=true

```

## Additional Properties

- `panl.range.<lpse_code>=<true_or_false>`

This property sets this facet as being a RANGE facet, the default property value for this is false.

Optional property which enables this facet (when set to 'true') to be a RANGE facet - i.e. the Solr search will include a range from a value, to another value.



**IMPORTANT:** Whilst Solr supports range facets on nearly all field types, only Solr field types of numbers are fully supported in Panl.

- `panl.range.min.<lpse_code>`

**MANDATORY IF** `panl.range.<lpse_code>=true` The minimum value to set the range slider to<sup>17</sup> - this must be a number (either integer or decimal)

- `panl.range.max.<lpse_code>`

**MANDATORY IF** `panl.range.<lpse_code>=true` The maximum value to set the range slider to<sup>18</sup> - this must be a number (either integer or decimal)

- `panl.range.prefix.<lpse_code>`

Optional property to prefix the range with, this is distinct from the `panl.prefix.<lpse_code>` property and changes the way in which the URI is generated.

- `panl.range.infix.<lpse_code>`

Optional property to place between the ranges, if set this will also affect the way

<sup>17</sup>Admittedly this is rather annoying having to know the value ranges ahead of time, however there are some niceties built into Panl to use the minimum and maximum values.

<sup>18</sup>Once again, annoying to have to know the values.

in which the LPSE URI path is generated. If this is not set it will default to the tilde character '~'



**IMPORTANT:** It is NOT recommended that a hyphen/minus sign (i.e. '-') is configured for the infix. Depending on the other configuration options, this may be confused for a negative value.

- `panl.range.suffix.<lpse_code>`

Optional property to suffix the range with with, this is distinct from the `panl.suffix.<lpse_code>` property and changes the way in which the URI is generated.

- `panl.range.min.value.<lpse_code>`

This property sets a replacement String for the minimum value of the range, which allows generating a friendlier URI. **Note:** You will still be able to use the minimum range value, this is just an additional nicety for SEO friendlier URLs.

- `panl.range.max.value.<lpse_code>`

This property sets a replacement String for the maximum value of the range, which allows generating a friendlier URI. **Note:** You will still be able to use the maximum range value, this is just an additional nicety for SEO friendlier URLs.

- `panl.range.min.wildcard.<lpse_code>`

Set whether to have a wildcard replace the minimum value when sending the query through to the Solr server. Setting this property to 'true' means that the minimum value also includes any values that are less than it.

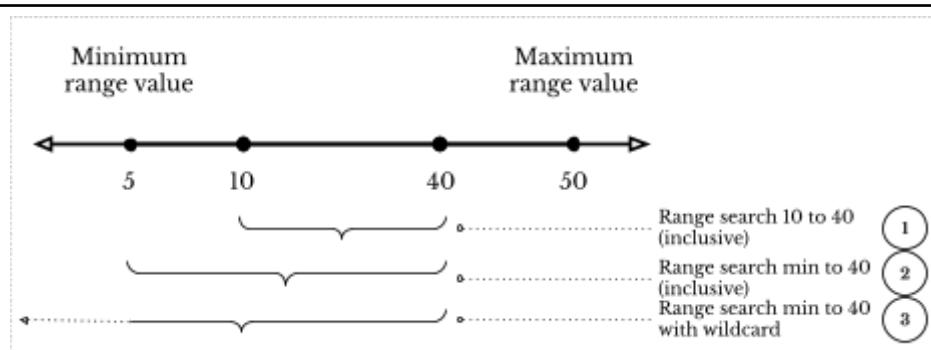
- `panl.range.max.wildcard.<lpse_code>`

Set whether to have a wildcard replace the maximum value when sending the

query through to the Solr server. Setting this property to 'true' means that the maximum value also includes any values that are greater than it.

## Visualising the Properties

Below is a diagram of what results will be included depending on the properties that are configured.



*Image: How the properties affect the values that the Solr search will include*

For the above range searches

1. For any default range search (in this case the search is 10 to 40), Panl will include both values - i.e. the search will select values greater than or equal to 10 and less than or equal to 40
2. For a search for the minimum value (`panl.range.min.value.<lpse_code>`) to 40, Panl will include all values greater than or equal to 5 and less than or equal to 40.
3. With the `panl.range.min.wildcard.<lpse_code>` property set, and the minimum value range search, Panl will return all results that are less than 40, with no lower bound.

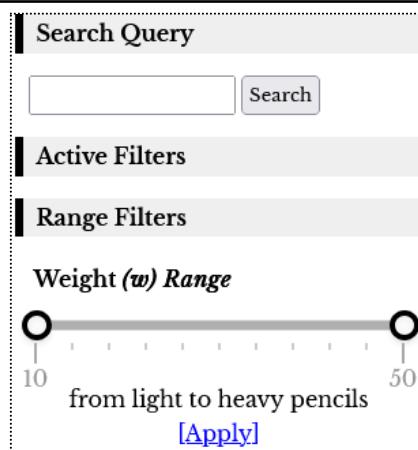
Having the minimum/maximum range properties allows documents that are added post configuration that would fall outside of the range to be included.

For example, if your min/max range properties were set to 5 and 100, without the wildcard properties, and a new document was added with the value 105, this document would never be able to be selected with the range filter. With the `panl.range.max.wildcard.<lpse_code>` property set, it would automatically be included.



**Tips:** It is prudent to set both the `panl.range.min.wildcard.<lpse_code>` and `panl.range.max.wildcard.<lpse_code>` properties to `true`.

When a RANGE facet is defined, a new heading of 'Range Filters' will appear in the Panl Results Viewer between the 'Active Filters' and 'Available Filters'.



*Image: The rendered RANGE facet*

The implementation in the Panl Results Viewer web app has all of the logic for the display of the range values including minimum and maximum value replacements.

### Decoding the URI - Range facet with an infix

/weighing+from+16+to+42+grams/w-w/

/weighing+from+	16	+to+	42	+grams	/	w-w	/
This is range prefix	This is the from value	This is the range infix	This is the to value	This is the range suffix	The LPSE code		
URI Path Value							

### Decoding the URI - Range facet without an infix

/weighing+from+	16	~	42	+grams	/	w+w	/
-----------------	----	---	----	--------	---	-----	---

This is range prefix	This is the from value	This is the to value	This is the range suffix	The LPSE code
<b>URI Path Value</b>				

Note that a RANGE facet with an infix uses the LPSE code `w-w`, whereas a RANGE facet without an infix uses the LPSE code `w+w` - note the plus versus minus signs.

## DATE Range Facet

Date range facets are only available on Solr field types of `solr.DatePointField` and for any of these field types, DATE Range facets will be enabled.



**IMPORTANT:** If the Solr field type is `solr.DatePointField`, then the **ONLY** type of facet that Panl supports is the DATE Range facet.

Additional fields were derived from the `solr_date` field to add additional options for end users to facet on, which are included in the XML example below. Descriptions for the derived fields are also explained in more detail. For the `simple-date` sample (in the `/sample/solr/simple-date/managed-schema.xml` file) the field definitions are:

```

01 <field name="solr_date" type="pdate" indexed="true" stored="true" ↵
      multiValued="false" />
02 <field name="text_date" type="string" indexed="true" stored="true" ↵
      multiValued="false" />
03 <field name="year" type="pint" indexed="true" stored="true" ↵
      multiValued="false" />
04 <field name="month" type="string" indexed="true" stored="true" ↵
      multiValued="false" />
05 <field name="day" type="pint" indexed="true" stored="true" ↵
      multiValued="false" />
06 <field name="day_of_week" type="string" indexed="true" stored="true" ↵
      multiValued="false" />
```

**Line 1:**

This defines the field `solr_date` to the type `type`, which is then matched to the `fieldType` `solr.DatePointField` - Panl can configure this field as a DATE range facet field.

#### Line 2:

This is the textual representation of the date, used to be displayed in the field sets that get returned

#### Line 3-6:

Instead of using the date field, additional fields are defined that use the date to enable faceting on the `year` (as an Integer), the `month` (as a String - e.g. January, February, etc.), the `date` (as an Integer), and the `day_of_week` (as a String - e.g. Monday, Tuesday, etc.) This allows the user to facet on the year, or day, or month, or day of week, whilst still allowing a DATE range. As an example:

<http://localhost:8181/panl-results-viewer/simple-date/default/next+12+months/S/>

Will show the following page.



**IMPORTANT:** The `simple-date` sample dataset and configuration was not set up in the instructions in the book, to see this example you will need to create the Solr collection and insert the data - see the `/sample/solr/simple-date/` and `/sample/panl/simple-date/` directories from the release package. Brief instructions are included in the [Additional Data](#) Chapter.

**Panl**  
*a very simple results viewer*

---

Available collections/fieldset URI Paths (CaFUPs):  
[\[/simple-date/default\]](#) [\[/simple-date/firstfive\]](#)

You are viewing collection/fieldset URI Path (CaFUP): /simple-date/default

Canonical URI: /next+12+months/1/10/Spn/

---

**Search Query**

Search

Search Results - Found 53 result(s) (exact)

[q.op AND](#) || [OR](#)

---

**Active Filters**

Solr Date (S)  
[\[remove\]](#) next 12 months

Page 1 of 6 Showing 10 results per page This page has 10 result(s)

Sort by Solr Date: [ASC](#) [DESC](#)

---

« PREV [NEXT](#) »

Show [3](#) [5](#) [10](#) per page

---

*Solr: query time 28ms.  
Panl: parse request 0ms, build request 40ms, send and receive request 76ms, parse response 0ms. Total time 117ms.*

**Range Filters**

Solr Date (S) Date Range

next ▾ 12 months ▾

[Apply range: next 12 months](#)

Text Date (text\_date)  
Friday 28 June 2024

Month (month)  
June

Year (year)  
2024

Solr Date (solr\_date)  
Fri Jun 28 12:26:01 AEST 2024

Text (text)  
Friday 28 June 2024

Day (day)  
28

Day Of Week (day\_of\_week)  
Friday

---

Year (y)  
[\[add\] 2024 \(27\)](#)  
[\[add\] 2025 \(26\)](#)

Month (m)  
[\[add\] April \(6\)](#)  
[\[add\] January \(5\)](#)  
[\[add\] July \(5\)](#)  
[\[add\] June \(5\)](#)  
[\[add\] November \(5\)](#)  
[\[add\] September \(5\)](#)  
[\[add\] August \(4\)](#)  
[\[add\] March \(4\)](#)  
[\[add\] May \(4\)](#)  
[\[add\] October \(4\)](#)  
[\[add\] December \(3\)](#)  
[\[add\] February \(3\)](#)  
Day (d)  
[\[add\] 11 \(5\)](#)

Text Date (text\_date)  
Monday 8 July 2024

Month (month)  
July

Year (year)  
2024

Solr Date (solr\_date)  
Mon Jul 08 12:26:01 AEST 2024

Text (text)  
Monday 8 July 2024

Day (day)  
8

Day Of Week (day\_of\_week)  
Monday

*Image:* The DATE range of 'next 12 months' with the returned facets

## Additional Properties

- `panl.date.<lpse_code>.previous=<string>`

This property sets the string that will be used to indicate that the DATE Range is for a previous length of time.

- `panl.date.<lpse_code>.next=<string>`

This property sets the string that will be used to indicate that the DATE Range is for a previous length of time.



**IMPORTANT:** You MUST set both the `panl.date.<lpse_code>.previous` and `panl.date.<lpse_code>.next` properties to enable the DATE Range facet. Whether you choose to allow users to be able to facet on either one or both is up to you.

- `panl.date.<lpse_code>.years=<string>`

This property sets the string that will be used as the range should be for years.

- `panl.date.<lpse_code>.months=<string>`

This property sets the string that will be used as the range should be for months.

- `panl.date.<lpse_code>.days=<string>`

This property sets the string that will be used as the range should be for days.

- `panl.date.<lpse_code>.hours=<string>`

This property sets the string that will be used as the range should be for hours.

## Example

The `simple-date.panl.properties` defines the `solr_date` Solr field as a DATE Range facet, the definition is below:

```

01 # <field "indexed"="true" "stored"="true" "name"="solr_date" "type"="pdate" <
      "multiValued"="false" />
02 panl.facet.S=solr_date
03 panl.name.S=Solr Date
04 panl.type.S=solr.DatePointField
05 panl.date.S.previous=previous
06 panl.date.S.next=next
07 panl.date.S.years=\ years
08 panl.date.S.months=\ months
09 panl.date.S.days=\ days
10 panl.date.S.hours=\ hours

```

The above configuration will allow the generation of URI paths such as:

- next 30 days  
/next+30+days/S/
- previous 10 hours  
/previous+10+hours/S/

Or any combination of previous/next with a value, followed by any one of years/months/days/hours

The format for a DATE Range facet is

<range\_identifier><value><range\_type>



**IMPORTANT:** If the range identifier or range type does not EXACTLY match the configured values, then the value will be ignored and not passed through the Solr server.

## Decoding the URI

The URI decoding is straightforward. Panl will know from the LPSE code that this is a DATE Range facet and will then parse the URI path value. Should the parsed value be valid (i.e. the range identifier and type are correct), then this will then be passed through to the Solr search server.

/	previous+	10	+hours	/	s	/
This is range identifier	This is value		This is the range type		The LPSE code	
URI Path Value						

A search on the simple date collection for the next 2 years

<http://localhost:8181/panl-results-viewer/simple-date/firstfive/next+2+years/S/>

Will pass through to Solr the following query:

```
q=*&*&q.op=OR&facet.limit=100&fl=solr_date,text_date,decade,year,month&facet.mincount=1&rows=10&facet.field=day&facet.field=month&facet.field=day_of_week&facet.field=decade&facet.field=text&facet=true&fq=solr_date:[NOW+TO+NOW%2B2YEARS]&start=0
```

In the above, the Solr range query is `fq=solr_date:[NOW+TO+NOW%2B2YEARS]` which, when URL decoded is `fq=solr_date:[NOW+TO+NOW+2YEARS]`.

## Some Notes

### About Value Replacements

This section covers prefixes, suffixes, infixes, and Boolean true/false value replacements.

Prefixes and suffixes can be added to any Panl field definition, irrespective of the field type. These will be transparently added and removed from the URI path when Panl processes it.



**IMPORTANT:** The decoded URI path part **\_\_MUST\_\_** exactly match the defined prefix/suffix/infix value. If it is not an exact match, then Panl will not pass the query value through to Solr.

### Defining a Prefix

```
panl.prefix.<lpse_code>=<your_prefix_here>
```

Prefixes are available on all facets, but are not used on fields.

In the case of a RANGE facet, these prefixes are still used if a singular value of this facet is selected - remember, Panl fields that are defined as RANGE facets return both range values and individual values.

If a range is selected, then this value is not used, instead the range prefixes and suffixes are used.

### Defining a Suffix

```
panl.suffix.<lpse_code>=<your_suffix_here>
```

Suffixes are available on all facets, but are not used on fields.

In the case of a RANGE facet, these suffixes are still used if a singular value of this facet is selected - remember, Panl fields that are defined as RANGE facets return both range values and individual values.

If a range is selected, then this value is not used, instead the range prefixes and suffixes are used.

### Defining an Infix

An infix is only available on RANGE facets and is set by the property

```
panl.range.infix.<lpse_code>
```

If this property is not set, then the tilde character '~' will be used as a default.

### Defining Boolean true/false replacement

These are only enabled if the Solr field type is solr.BoolField and

```
panl.bool.<lpse_code>.true=able to be disassembled
```

If the boolean field is true, then the above property will be used in the URI path part. Remember that a prefix and suffix may still be applicable on the boolean value, which will be prepended, appended to the above value.

```
panl.bool.<lpse_code>.false=cannot be disassembled
```

If the boolean field is false, then the above property will be used in the URI path part. Remember that a prefix and suffix may still be applicable on the boolean value, which will be prepended, appended to the above value.



**IMPORTANT:** If you wish to have whitespace before either the suffix (or prefix, although unlikely) you must backslash encode the whitespace. For example, the brand Solr field has both a prefix and suffix, with the suffix requiring a ‘space’ between the value and the text

```
panl.facet.b=brand
panl.name.b=Brand
panl.type.b=solr.StrField
panl.prefix.b=Manufactured by
panl.suffix.b=\ Company
```

Note the \ Company value above for the panl.suffix.b property

## On Field Validations

Field validations in Panl only support the integer and floating point types in Solr. The validations that occur within Panl simply strips out any non-matching characters and then pass it to the underlying Solr server.

The second type of validation is ensuring that the prefix/suffix and boolean value replacements are an exact match. If there is no match, the passed in request part is discarded and not passed through to the Solr search engine.

## About Hierarchical Facets

In the book store example, the first\_published\_year (LPSE code 'f') Solr field will only appear if the decade\_published (LPSE code 'd') Solr field has been selected. The property set on the first published Solr field:

```
panl.when.f=D
```

Shown below are the facets available with an empty search (left). Once a value for the 'Decade Published' facet has been selected. The 'First Year Published' facet is shown (right).

<b>Genre (g)</b> <a href="#">[add]</a> Fantasy (11) <a href="#">[add]</a> Young Adult (11) <a href="#">[add]</a> Mystery (9) <a href="#">[add]</a> Thriller (9) <a href="#">[add]</a> Crime (8) <a href="#">[add]</a> Detective (8) <a href="#">[add]</a> Romance (8) <a href="#">[add]</a> Children (3) <a href="#">[add]</a> Dystopia (3) <a href="#">[add]</a> Humour (3) <a href="#">[add]</a> Drama (2) <b>Authors (A-Z) (d)</b> <a href="#">[add]</a> C (11) <a href="#">[add]</a> M (6) <a href="#">[add]</a> D (3) <a href="#">[add]</a> B (2) <b>Decade Published (D)</b> <a href="#">[add]</a> 2000 (8) <a href="#">[add]</a> 1990 (6) <a href="#">[add]</a> 1980 (4) <a href="#">[add]</a> 2010 (3) <a href="#">[add]</a> 2020 (1) <b>Book Length (L)</b> <a href="#">[add]</a> Novel (16) <a href="#">[add]</a> Novella (6)	<b>Genre (g)</b> <a href="#">[add]</a> Fantasy (6) <a href="#">[add]</a> Young Adult (6) <a href="#">[add]</a> Romance (4) <a href="#">[add]</a> Crime (2) <a href="#">[add]</a> Detective (2) <a href="#">[add]</a> Dystopia (2) <a href="#">[add]</a> Mystery (2) <a href="#">[add]</a> Thriller (2) <b>First Published Year (f)</b> <a href="#">[add]</a> First published in 2008 (2) <a href="#">[add]</a> First published in 2009 (1) <a href="#">[add]</a> First published in 2002 (1) <a href="#">[add]</a> First published in 2003 (1) <a href="#">[add]</a> First published in 2006 (1) <a href="#">[add]</a> First published in 2007 (1) <a href="#">[add]</a> First published in 2009 (1) <b>Authors (A-Z) (d)</b> <a href="#">[add]</a> C (4) <a href="#">[add]</a> M (4) <b>Book Length (L)</b> <a href="#">[add]</a> Novel (6) <a href="#">[add]</a> Novella (2)
--	--

**Image:** The Panl Results Viewer web app showing the hierarchical facet of 'First Published Year' only appearing after the 'Decade' facet has been selected.



**Notes:** **ANY** Panl configured facet can have a hierarchy applied to it and the hierarchy can depend on any other facet, including Panl parameters (e.g. page, query string, number of pages, etc).

## About Sorting Facets

By default, Solr will return facets ordered by the number of documents which contain the specific facet. In the below image (taken from the Book Store collection), the books in the dataset with Authors having a surname starting with C is 11, D is 3.



**Note:** This is **NOT** the same as the sort fields defined by the property `panl.sort.fields`, which will sort the returned documents, this is for sorting the returned facets.

For some facets you may wish to order this in a more logical way for the end user - i.e. in alphabetical order. To do this, you can set the `panl.facetsort.<lpse_code>` property to `index`, rather than the default of `count` (if the property is not included).

**Authors (A-Z) (A)**

- [\[add\]](#) C (11)
- [\[add\]](#) M (6)
- [\[add\]](#) D (8)
- [\[add\]](#) B (2)

*Image:* The Panl Results Viewer web app showing the Authors A-Z facet ordered by 'count' which is the default.

By setting the `panl.facetsort.<lpse_code>` property to `index` the facet results will be sorted by their value, not their count.

**Authors (A-Z) (A)**

- [\[add\]](#) B (2)
- [\[add\]](#) C (11)
- [\[add\]](#) D (8)
- [\[add\]](#) M (6)

*Image:* The Panl Results Viewer web app showing the Authors A-Z facet ordered by 'index' or the value of the facet.

In the above image, the facet results are the same, however the resulting list of facets are now in alphabetical order - the books in the dataset with Authors having a surname starting with C is 11, D is 3 as was expected.



**Notes:** The facet sorting option can be applied to \_\_ANY\_\_ Panl configured facet.

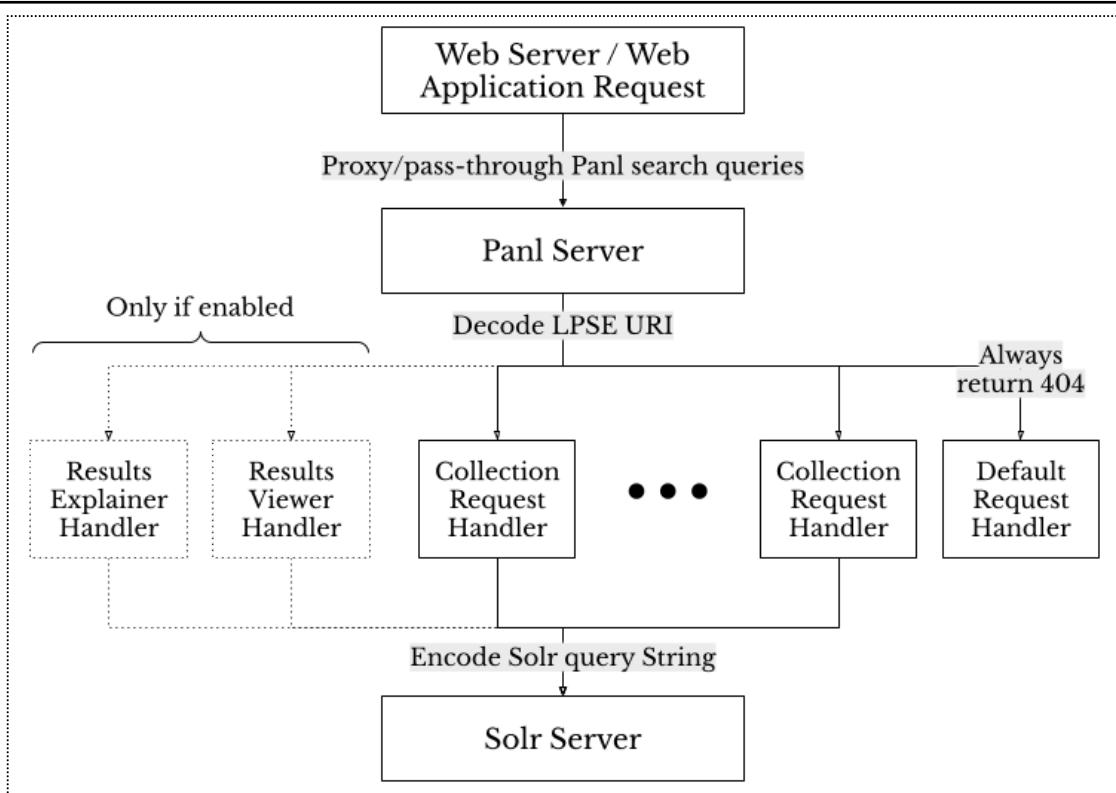
~ ~ ~ \* ~ ~ ~

# Putting It All Together

Panl was not specifically designed to be web facing, although it could be should you so choose. The recommended way is to hide the Panl implementation CaFUPs URL and proxy them through a web or application server.

A number of Panl servers can be spun up at the same time (on different ports of course) and load balanced between them. There is no state to be kept between requests, so it is lightweight and fast.

## A Brief Architecture



*Image: The recommended approach to integrating the Panl server*

In the above diagram, the request hits the front end web/app server, is proxied through to the correct Panl CaFUP which is then handled by a Collection Request Handler. The

URI path part is decoded and parsed, the query built, sent to the Solr server and the response built and returned.

You may wish to open the Panl server to external requests, however, be aware that there is no authentication or authorisation in the Panl server. See the section on [Is Synapticloop Panl For Me?](#) to inform your decision.

## Generating SEO Friendly URLs

Going beyond the in-built features of the Panl configuration, generating URLs - especially those with a passthrough parameter requires.

You could either go to the datasource (e.g. a database) and build the URLs programmatically, or let Panl generate all of the URI paths and, or a combination of both.

### Data Dependencies

Using the original datasource to generate links will create a dependency on the Panl configuration, and should the Panl configuration change, then the links may not still be valid.

For example, if you generated links for the mechanical pencils collection from a datasource as

```
/Manufactured+by+Koh-i-Noor+Company/b/
```

And the Panl configuration changed to generate the path:

```
/fine-pencils-by-Koh-i-Noor/b/
```

Then Panl would not return any results as the prefix and suffix did not match and this facet would not be passed through to the Solr server.

### Passthrough URIs

Passthrough URIs are a little more able to withstand Panl configuration changes as they are not linked to any Solr field value.

Passthrough URIs are also best used to link to a single document, or a subset of documents. For example, in the Book Store dataset, the following link uniquely identifies a book (using the ID field from Solr):

<http://localhost:8181/panl-results-viewer/book-store/default/4/i/>

And consequently any passthrough parameter will give the same result, such that

<http://localhost:8181/panl-results-viewer/book-store/default/Michael+Connelly+The+Last+Coyote+-Harry+Bosch+Series/4/zi/>, and

<http://localhost:8181/panl-results-viewer/book-store/default/This+is+not+a+link+to+a+book+but+returns+results/4/zi/>

Return the same result.

When using passthrough URIs, you may wish to configure a separate CaFUP just for these values, and be able to tweak other configured CaFUPs.

## Handling Errors

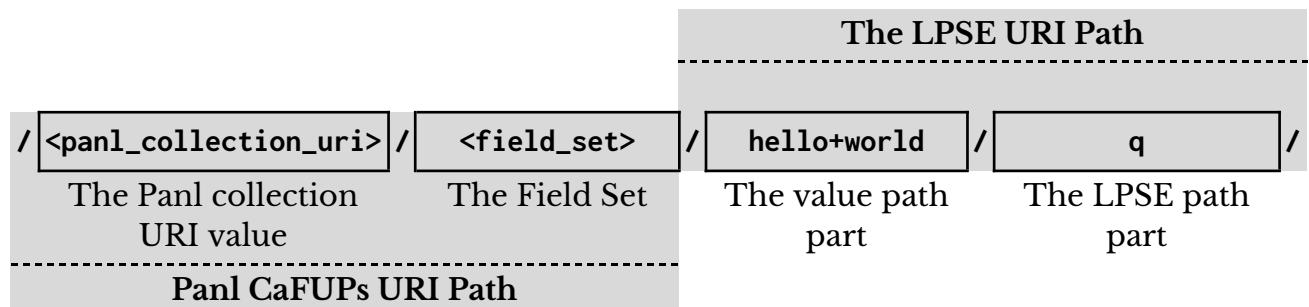
You may also wish to return a 404 HTTP status code if the number of results returned is zero (0). To indicate to users and search engines that are scraping your site that this URL is no longer available.

~ ~ ~ \* ~ ~ ~

# Panl LPSE URI Paths Explained

---

How to decode the URI parameters for the LPSE encoded URI. The nomenclature for the URI path is detailed below



For the information in this chapter, only the 'LPSE URI Path' will be discussed.

Each of the LPSE URI Path components will have the following information:

---

**Property** The property that configures this in the `<panl_collection_uri>.panl.properties` file.

---

**Value Path Part** Whether the value of this parameter or facet appears in the value path part, possible values are:

- **Always** - will always appear in the URI value path part
  - **Never** - will never appear in the URI valuepath part
  - **Conditional** - Depending on configuration options or values, may appear in the URI value path part
- 

**LPSE Path Part** Whether the value of this parameter or facet appears in the LPSE path part, possible values are:

- **Always** - will always appear in the LPSE path part
  - **Never** - will never appear in the LPSE path part
  - **Conditional** - Depending on configuration options or values, may appear in the LPSE path part
- 

**Canonical Value Path Part** Whether the value of this parameter or facet appears in

---

the canonical value path, possible values are:

- **Always** - will always appear in the canonical value path part
- **Never** - will never appear in the canonical value path part
- **Conditional** - Depending on configuration options or values, may appear in the canonical value path part

**Canonical LPSE Path part** Whether the value of this parameter or facet appears in the canonical URI value path, possible values are:

- **Always** - will always appear in the canonical LPSE path part
- **Never** - will never appear in the canonical LPSE path part
- **Conditional** - Depending on configuration options or values, may appear in the canonical LPSE path part

**Value Format** The format for the complete LPSE URI path.

**Default LPSE Code** The default LPSE code used in this book, or from the Panl generator utility.

**Notes** Any additional notes applicable to the parameter or facet LPSE code.

**See also** Other properties or headings that may influence the configuration of a parameter or facet

## Parameter Query

*The user entered search term (either a word or phrase) to search the Solr collections for a match.*

---

**Property** panl.param.query

---

**Value Path Part** Always

---

**LPSE Path Part** Always

---

**Canonical Value Path Part** Always

---

**Canonical LPSE Path part** Always

---

**Value Format** /<string>/<lpse\_code>/

---

**Default LPSE Code** q

---

**Notes**

---

**See also**

---

## Parameter Query Operand

*The parameter that controls whether to use an OR, or an AND query on the search term - this maps to the `q.op` query parameter for the Solr search server.*

---

**Property** `panl.param.operand`

---

**Value Path Part** **Never**

---

**LPSE Path Part** **Sometimes** - this will only appear in the URI path and URI Path part if it does not match the default query operand.

---

**Canonical Value Path Part** **Never**

---

**Canonical LPSE Path part** **Sometimes** - this will only appear in the URI path and URI Path part if it does not match the default query operand.

---

**Value Format** `/<lpse_code>(+/-)/`

---

**Default LPSE Code** `o`

---

**Notes**

1. The default query operand can be set in the `<panl_collection_uri>.panl.collection.properties` file.

---

**See also**

---

This parameter will rarely be used as it has a default property `solr.default.query.operand` which controls all of the collected CaFUPS. The default generated properties have a query operand of `-` (OR) which will give the greatest number of results. If this is set to `+` (AND) then the search term must match in all of the fields configured in the `solrconfig.xml` file.

## Parameter Number of Rows

*The number of rows of results to return from the Solr search server.*

---

**Property** `panl.param.numrows`

---

---

**Value Path Part** **Sometimes** - this will only appear in the URI path if it does not match the default number of rows property.

---

**LPSE Path Part** **Sometimes** - this will only appear in the URI Path part if it does not match the default number of rows property.

---

**Canonical Value Path Part** **Always**

---

**Canonical LPSE Path part** **Always**

---

**Value Format** /<prefix><integer><suffix>/<lpse\_code>/

---

**Default LPSE Code** n

---

**Notes** 1. If the value of this parameter is not set then the default is set to 10.

---

**See also**

---

The default `solr.numrows.default` when generated is 10 results per page

This parameter may have a prefix and or suffix configured, in the example `mechanical-pencils.panl.properties` file it is configured with the following properties

```
01 panl.param.numrows.prefix=
02 panl.param.numrows.suffix=-per-page
```

The LPSE URI path would be

/3-per-page/n/

If the properties were configured

```
01 panl.param.numrows.prefix=show-me
02 panl.param.numrows.suffix=-per-query
```

The LPSE URI path would be

/show-me-3-per-query/n/

## Parameter Page Number

*The parameter which sets the page number of the results*

---

**Property** `panl.param.page`

---

**Value Path Part** **Sometimes** - this will only appear in the URI Path part if it is not the first page.

---

**LPSE Path Part** **Sometimes** - this will only appear in the URI Path part if it is not the first page.

---

**Canonical Value Path Part** **Always**

---

**Canonical LPSE Path part** **Always**

---

**Value Format** `/<prefix><integer><suffix>/<lpse_code>/`

---

**Default LPSE Code** `p`

---

**Notes** For the first page, the results will be the same if this value is included or not. E.g. `/` is equivalent to `/page-1/p/`

---

**See also**

---

This parameter may have a prefix and or suffix configured, in the example `mechanical-pencils.panl.properties` file it is configured with the following properties

```
01 panl.param.page.prefix=page-
02 panl.param.page.suffix=
```

The LPSE URI path would be

`/page-1/p/`

If the properties were configured

```
01 panl.param.numrows.prefix=i-am-on-page-
02 panl.param.numrows.suffix=-of-the-results
```

The LPSE URI path would be

```
/i-am-on-page-4-of-the-results/p/
```

## Parameter Pass Through

*A parameter which is ignored by Panl and never sent through to the Solr query. This is useful for generating SEO friendly URLs.*

<b>Property</b>	<code>panl.param.passthrough</code>
<b>Value Path Part</b>	<b>Always</b>
<b>LPSE Path Part</b>	<b>Always</b>
<b>Canonical Value Path Part</b>	<b>Sometimes</b> - By default this will NOT appear in the LPSE path part unless the <code>panl.param.passthrough.canonical</code> property is set to <code>true</code> .
<b>Canonical LPSE Path part</b>	<b>Sometimes</b> - By default this will NOT appear in the LPSE path part unless the <code>panl.param.passthrough.canonical</code> property is set to <code>true</code> .
<b>Value Format</b>	<code>&lt;string&gt;/&lt;lpse_code&gt;/</code>
<b>Default LPSE Code</b>	<code>z</code>
<b>Notes</b>	1. This value will never be passed through to the Solr query
<b>See also</b> <a href="#">panl.param.passthrough.canonical</a>	

With the default for the property `panl.param.passthrough.canonical`, i.e.

```
01 panl.param.passthrough.canonical=false
```

The LPSE URI path would be

```
/some+sort+of+arbitrary+string/z/
```

And the canonical URI path would be

```
/page-1/10-per-page/pn/
```

**Note:** in the above URI path, the page number and number of results per page always appear, but the passthrough parameter does not.

However, with the property set to true:

```
01 panl.param.passthrough.canonical=true
```

The LPSE URI path would be

```
/some+sort+of+arbitrary+string/z/
```

And the canonical URI path would be

```
/some+sort+of+arbitrary+string/page-1/10-per-page/zpn/
```

## Parameter Sort Order

*A parameter which defines how the results will be sorted.*

<b>Property</b>	panl.param.sort
<b>Value Path Part</b>	Never
<b>LPSE Path Part</b>	<b>Sometimes</b> - this will only appear if it does not match the default sort order (relevance descending).
<b>Canonical Value Path Part</b>	Never
<b>Canonical LPSE Path part</b>	<b>Sometimes</b> - this will only appear if it does not match the default sort order (relevance descending).
<b>Value Format</b>	/<lpse_code><sort_field_lpse_code>(+-)/
<b>Default LPSE Code</b>	o
<b>Notes</b>	If the value of this parameter
<b>See also</b>	

By default the Solr results return in search relevance order descending (there is no ascending order for search relevance), and Panl respects that.

The default sorting LPSE URI path is simply

/

And the canonical LPSE URI path for the sort order is the same

/

However, if there is a sort order set, then the LPSE URI path and the canonical LPSE URI paths will.

### Ordering by a Solr field.

Search ordering for fields is controlled by the `panl.order.fields` property in the `<panl_collection_uri>.panl.properties` file.

`/sb+/-` order the search results by `brand` in ascending order

`/sb-/-` order the search results by `brand` in descending order

If no Panl LPSE field definition is included, then it defaults to the relevance ordering. For any of the orderings, you have the option to replace the ordering with a new ordering, or add to the ordering with an additional ordering.

`/sb-sN+/-` order the search results by `brand` in descending order, then by `name` ascending.

## Facets

*Facets are the primary way of filtering results and are straightforward to decode.*

<b>Property</b>	<code>panl.facet.&lt;lpse_code&gt;</code>
<b>Value Path Part</b>	Always
<b>LPSE Path Part</b>	Always
<b>Canonical Value Path Part</b>	Always
<b>Canonical LPSE Path part</b>	Always
<b>Value Format</b>	<code>/&lt;prefix&gt;&lt;string&gt;&lt;suffix&gt;/&lt;lpse_code&gt;/</code> , or <code>/&lt;range_prefix&gt;&lt;string&gt;&lt;range_suffix&gt;/&lt;lpse_code&gt;/</code> , or <code>/&lt;min_value_replacement&gt;&lt;string&gt;&lt;max_value_replacement&gt;/&lt;lpse_code&gt;/</code>
<b>Default LPSE Code</b>	N/A
<b>Notes</b>	

---

### See also

---

For any facet, the URI path would be

```
/the+facet+value+with+prefix+or+suffix/?/
```

Where the `?` is the LPSE code, these are always passed through to the canonical URI path, as in

```
/the+facet+value+with+prefix+or+suffix/?/
```

For any URI path, the prefixes, suffixes and, in the case of a RANGE facet, the infix will be included in the URI. Multiple LPSE codes can be present in any order without affecting the returned results. For example, the URI path generated by Panl for this URL:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/firstfive/Red/Purple/Entry+Level/WWC/>

Is

```
/Red/Purple/Entry+Level/WWC/
```

with the canonical URI

```
/Purple/Red/Entry+Level/page-1/10-per-page/WWCpn/
```

**Note:** The Panl generated URI path is different from the canonical URI path, the canonical URI that is generated

- Always includes the page number and the number of results per page.
- Facets are placed in LPSE order
- Facets are ordered alphabetically

This ensures that there is only ever a single URI path for any set of results. If you were to move the LPSE codes and values around - e.g. putting the category value first (LPSE code 'C' as:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/firstfive/Entry+Level/Red/Purple/CWW/>

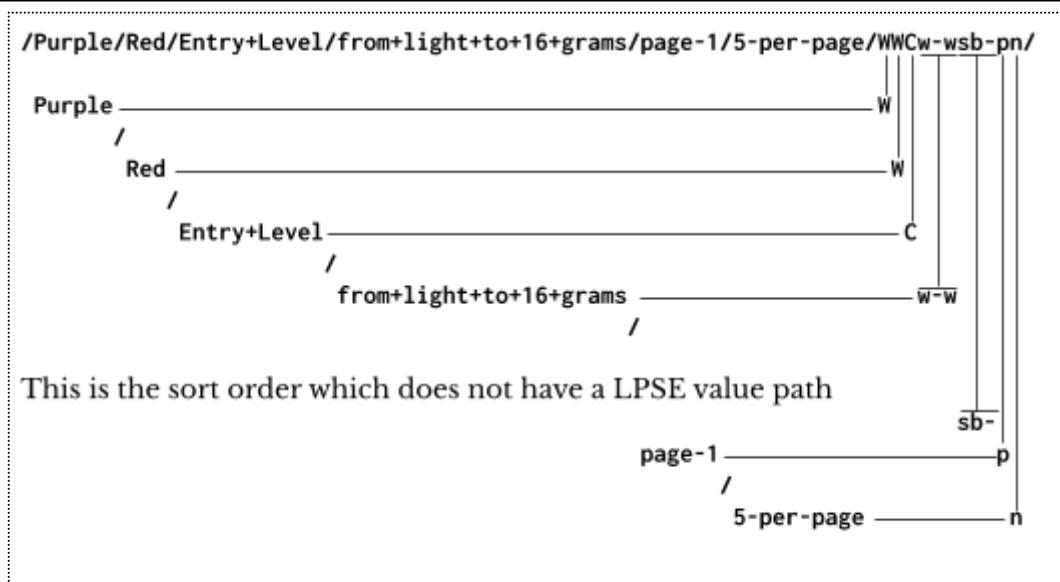
The results are the same, and the canonical URI remains as

/Purple/Red/Entry+Level/page-1/10-per-page/WWCpn/

## Explaining A More Complex Example

For the canonical URI path:

/Purple/Red/Entry+Level/from+light+to+16+grams/page-1/5-per-page/WWCw-wsb-pn/



*Image: Mapping of LPSE codes to LPSE values*

Decoding the above:

- 'Purple' is the Colour facet
- 'Red' is another Colour facet
- 'Entry Level' is the Category of pencil
- 'Weight' is from the minimum value to to 16 grams (inclusive)
- Sort order is by brand descending
- Page 1
- 5 results per page.

Or in more human readable terms:

*Select all entry level pencils that come in purple or red weighing anywhere from light to 16 grams. Sort the results by brand, showing the first page with up to 5 results.*

As you get more familiar with Panl visually decoding the URI path becomes easier, although once Panl is set up and running, the need for this diminishes.

**Remember:** that you can always use the Panl Results Explainer for additional information, for the above URI path - the results that the explainer gives are as follows:

### Request Token Explainer

```
PANL [ VALID ] <facet> LPSE code 'W' (solr field 'colours') with parsed value 'Purple', incoming value 'Purple'.
```

```
PANL [ VALID ] <facet> LPSE code 'W' (solr field 'colours') with parsed value 'Red', incoming value 'Red'.
```

```
PANL [ VALID ] <facet> LPSE code 'C' (solr field 'category') with parsed value 'Entry Level', incoming value 'Entry+Level'.
```

```
PANL [ VALID ] <facet (RANGE)> LPSE code 'w' (solr field 'weight') incoming value 'from+light+to+16+grams', parsed value 'RANGE(10:16) with infix'.
```

```
PANL [ VALID ] <sort> LPSE code 's' sort code 'b' (solr field 'brand'), sorted DESCending
```

```
PANL [ VALID ] <page_number> LPSE code 'p' using parsed value of '1'.
```

```
PANL [ VALID ] <num_rows> LPSE code 'n' original URI path value '5-per-page' using parsed value of '5'.
```

~ ~ ~ \* ~ ~ ~

# Integration And The Panl Response Object

---

The Panl response is always of the type JSON (with a MIME type of "application/json") with UTF-8 encoding.

To integrate your web app with the Panl response you may choose any language, and whilst the integration parts may seem daunting at first, the Panl Results Viewer Web App JavaScript file comes in at under 700 lines of code (not including the minified libraries). Provided that your language of choice can parse JSON (which almost all have libraries for, if not built-in already), the burden should not be too onerous.

## Error Responses

There will **ALWAYS** be an `error` key (with a boolean value of either `true` or `false`) on all responses which makes it easy to determine the best course of action. If there is an error (either 404, or 500), then the `error` key will be set to true, an integer status code keyed on `status`, and keyed on `message`, a human readable response.

An example 404 response (in non-verbose mode):

```

01 {
02   "error":true,
03   "status":404,
04   "message": "Not found"
05 }
```

An example 500 response (in non-verbose mode):

```

01 {
02   "error":true,
03   "status":500,
04   "message": "Internal server error"
05 }
```

## Success Responses

Rather than replacing the original Solr JSON response object, Panl appends a new JSON Object to the existing Solr JSON response object.

The Solr JSON response object has the following form:

```

01 {
02   "response": { ... },
03   "responseHeader": { ... },
04   "facet_counts": { ... },
05   "highlighting": { ... }
06 }
```

The Panl server adds two additional keys, namely a boolean valued key of `error`, and a JSON object keyed on `panl` to the root object. The response becomes (bold emphasis added):

The error key will **ALWAYS** be `false` for a successful return of the Panl Response object.

```

01 {
02   "response": { ... },
03   "responseHeader": { ... },
04   "panl": { ... },
05   "error": false,
06   "facet_counts": { ... },
07   "highlighting": { ... }
08 }
```

Within the above `panl` JSON object are the following keys:

```

01 {
02   "active": { ... },
03   "available": { ... },
04   "canonical_uri": "",
05   "fields": { ... },
06   "pagination": { ... },
07   "query_operand": { ... },
08   "query_respond_to": "",
09   "sorting": { ... },
10   "timings": { ... }
11 }
```

The following sections will go into greater detail for each of the JSON keys.

## The "active" JSON Object

This JSON object contains all active queries, sorting, pagination, operands, and BOOLEAN, DATE range, RANGE, OR, and Regular facets with links to remove the specific active filter.



**Notes:** The keys contained within the JSON object will only appear if the related query, facet, or parameter has been applied to the query.

```

01 {
02   "facet": [ ... ],
03   "numrows": { ... },
04   "page": { ... },
05   "query_operand": { ... },
06   "query": { ... },
07   "sort": [ ... ],
08   "sort_fields": { ... }
09 }
```

**Line 2:**

The JSON array of active facets - Regular, OR, RANGE, or DATE Range facets.

**Line 3:**

The JSON object of the selected number of rows to display per page

**Line 4:**

The JSON object of active page number

**Line 5:**

The selected query operand

**Line 6:**

The query phrase

**Line 7:**

The array or selected sorting in order of their selection

**Line 8:**

The sort fields lookup containing a key/value pair of the Solr sort field name and the Panl sort field name.

If there are no active filters for the particular type of filter, then the associated key will not be present. Each of the JSON arrays can hold one or more JSON objects. The `sort_fields` JSON object is added as a convenience lookup function of the active sort fields for use with the `available.sorting` integration and implementation

Each of the objects will contain at least the following keys and will not be repeated for the other JSON object explanations:

```

01 {
02 ...
03 "panl_code": "m",
04 "value": "Clutch",
05 "encoded": "Clutch",
06 "remove_uri": "/weighing+from+14+grams+to+46+grams/page-2/5-per-page/w-wsb+pno+/"
07 ...
08 }
```

**Line 3:**

The Panl LPSE code

**Line 4:**

The value of the Solr field

**Line 5:**

The value with any prefix or suffix which is URI encoded. For example the brand Solr field with Panl LPSE code `w` has a prefix of `Manufactured by+` and a suffix of `+Company`, the encoded key with a value of `OHT0` would look like the following:

`Manufactured+by+OHT0+Company`

**Line 6:**

The URI path to remove this facet from the search results.

## Overall Integration and Implementation

At a base level, implementation is straightforward with any of the active facets, although not all of the active filters may be rendered to the page. In the Panl Results Viewer, the only active facets that are rendered for removal are:

1. Any query phrase,
2. Any active facets, and
3. Any sort ordering

The following filters are not rendered to the page within the Active Filters section:

1. Query operand,
2. Page number, and
3. Number of results per page

This does not mean that your implementation cannot include them as well, it was just a choice that was made with the Panl Results Viewer.

The general rule for implementation of generating links is that the `remove_uri` value is used as the `href` attribute for the anchor tag and the `encoded` value used as the text - i.e.

```
<a href="/Clutch/page-2/msb+po+/">5-per-page</a>
```

```
<a href=" /Clutch/page-2/msb+po+/"> 5-per-page </a>
```

remove\_uri JSON key
 URL decoded
encoded JSON key



**Notes:** To render the text of the anchor tag, the `encoded` value will need to be URI decoded. In the `panl-viewer.js` file, additional processing is done thusly:

```
return(decodeURI(text).replaceAll("+", " ").replaceAll("%2B", "+));
```

The above line will replace the `+` character with spaces (which is not done with the javascript function `decodeURI()`) then any encoded `+` characters (i.e. `%2B`) are replaced with the `+` character.

## The "active.facet" JSON Array (Regular facet object)

A Regular facet contains the information to display the value and the removal URI.

```
01 {
02   "facet_name": "mechanism_type",
03   "name": "Mechanism Type",
04   "panl_code": "m",
05   "value": "Clutch",
06   "encoded": "Clutch",
07   "remove_uri": "/weighing+from+14+grams+to+46+grams/page-2/5-per-page/w-wsb+pno+/"
08 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section and are not repeated.

**Line 2:**

The name of the Solr field

#### Line 3:

The Panl name of the Solr field, i.e. the configured display name

### Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

### The "active.facet" JSON Array (BOOLEAN facet object)

A BOOLEAN facet contains the information to display the value and the removal URI and will have additional JSON keys `inverse_encoded`, `inverse_uri`, and `is_boolean_facet` - which is always set to `true`.

```

01 {
02   "facet_name": "disassemble",
03   "is_boolean_facet": true,
04   "inverse_encoded": "cannot+be+disassembled",
05   "inverse_uri": "/cannot+be+disassembled/D/",
06   "name": "Disassemble",
07   "encoded": "able+to+be+disassembled",
08   "panl_code": "D",
09   "remove_uri": "/",
10   "value": "true"
11 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section and are not repeated.

#### Line 2:

The name of the Solr field

#### Line 3:

Indicates that this is a BOOLEAN facet, with only two available values, true, or false.

#### Line 4:

This is the display for the inverse of the current value. In the above example, the value is set to "true", so this `inverse_value` displays the "false" value, with boolean value replacement.

#### Line 5:

This is the URI path to use to invert the currently selected value, if true, this is the false URI path, if false, this is the true URI path.

#### Line 6:

The Panl name of the Solr field, i.e. the configured display name

## Integration and Implementation

The BOOLEAN facet also includes two additional keys which can be used to render an 'inverse' link. So, in addition to the remove link, the inverse can be constructed using the `inverse_uri` value as the href attribute for the anchor tag and the `inverse_encoded` value used as the text - e.g. the URL:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/able+to+be+disassembled/D/>

The the inverse link is:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/cannot+be+disassembled/D/>

And an example anchor tag implementation for the inverse would be:

<pre>&lt;a href=" /cannot+be+disassembled/D /"</pre>	<pre>"&gt; cannot be disassembled &lt;/a&gt;</pre>
<code>inverse_uri</code> JSON key	URL decoded <code>inverse_encoded</code> JSON key

The rest of the implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

**Active Filters****Disassemble (D)**[\[remove\]](#) able to be disassembled[\[invert\]](#) cannot be disassembled

*Image:* The In-Build Panl Results Viewer web app showing the BOOLEAN filter with the invert link.

## The "active.facet" JSON Array (DATE Range facet object)

A DATE facet contains the information to display the value and the removal URI and will have an additional key of and `is_date_facet` - which is always set to `true`.



**IMPORTANT:** This example comes from the `simple-date` Solr collection and is see the [Additional Data](#) section for adding this sample data to Solr.

```

01 {
02   "facet_name": "solr_date",
03   "is_date_facet": true,
04   "name": "Solr Date",
05   "encoded": "next+10+months",
06   "panl_code": "S",
07   "remove_uri": "/",
08   "value": "10"
09 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section and are not repeated.

### Line 2:

The name of the Solr field

### Line 4:

Indicates that this is a DATE facet allowing a range of dates to be selected

#### Line 5:

The Panl name of the Solr field, i.e. the configured display name

## Integration and Implementation



**Notes:** This filter will also **ALWAYS** be returned in the available filters object, so you may choose not to display this in the active filters and use the available filters. Within the Panl Results Viewer web app, it displays both within the Active Filters and Range Filters.

The base implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text, however you may not want to to render this facet in the Active Filters, and just use the Range Filters section .

The screenshot shows a user interface for filtering search results. It features two main sections: "Active Filters" and "Range Filters".

- Active Filters:** Contains a single item: "Solr Date (S) [remove] previous 12 months".
- Range Filters:** Contains a heading "Solr Date (S) Date Range" followed by three buttons: "previous ▾", "12", and "months ▾". Below these buttons is a blue link "Apply range: previous 12 months".

*Image: The In-Build Panl Results Viewer web app showing the DATE filter in both the Active Filters and Range Filters sections.*

## The "active.facet" JSON Array (OR facet object)

An OR facet contains the information to display the value and the removal URI and will have an additional key of and `is_or_facet` - which is always set to `true`.

```

01 {
02   "facet_name": "solr_date",
03   "is_or_facet": true,
04   "name": "Solr Date",
05   "encoded": "next+10+months",
06   "panl_code": "S",
07   "remove_uri": "/",
08   "value": "10"
09 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section and are not repeated.

#### Line 2:

The name of the Solr field

#### Line 3:

Indicates that this is an OR facet allowing multiple selections of this facet

#### Line 4:

The Panl name of the Solr field, i.e. the configured display name

## Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

## The "active.facet" JSON Array (RANGE facet object)

A RANGE facet contains the information to display the value and the removal URI. A RANGE facet will have an additional key of `value_to`, and have two additional boolean keys, `has_infix`, and `is_range_facet` - which is always set to `true`.

```

01 {
02   "facet_name": "weight",
03   "has_infix": true,
04   "is_range_facet": true,
05   "name": "Weight",
06   "value_to": "45",
07   "encoded": "weighing+from+14+grams+to+45+grams",
08   "panl_code": "w",
09   "remove_uri": "/Manufactured+by+Koh-i-Noor+Company/b/",
10   "value": "14"
11 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section and are not repeated.

#### Line 2:

The name of the Solr field

#### Line 3:

Whether this field has an infix for the RANGE facet

#### Line 4:

Whether this is a RANGE facet

#### Line 5:

The Panl name of the Solr field, i.e. the configured display name

#### Line 6:

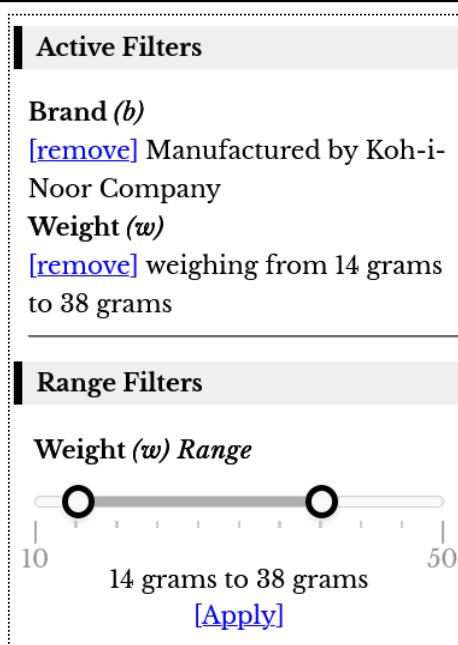
The ending range value for this range, used in conjunction with the `value` key of this JSON object.

## Integration and Implementation



**Notes:** This filter will also **ALWAYS** be returned in the available filters object, so you may choose not to display this in the active filters and use the available filters. Within the Panl Results Viewer web app, it displays both within the Active Filters and Range Filters.

The base implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text, however you may not want to to render this facet in the Active Filters, and just use the Range Filters section with the `value` and `value_to` already populated.



**Image:** The In-Build Panl Results Viewer web app showing the RANGE filter in both the Active Filters and Range Filters sections.

## The "active.numrows" JSON Object

The following URL:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Clutch/page-2/5-per-page/mpn/>

Has the following active filters set:

- **Mechanism type (m)** : Clutch
- **Page number (p)**: page 2

- **Number of rows per page (*n*):** 5 per page [This is the active filter to remove]

```

01 {
02   "remove_uri": "/Clutch/m/",
03   "panl_code": "n",
04   "value": "5",
05   "encoded": "5-per-page"
06 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section.

## Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

## The "active.page" JSON Object

The following URL:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Clutch/page-2/5-per-page/mpn/>

Has the following active filters set:

- **Mechanism type (*m*) :** Clutch
- **Page number (*p*):** page 2 [This is the active filter to remove]
- **Number of rows per page (*n*):** 5 per page

```

01 {
02   "remove_uri": "/Clutch/5-per-page/mn/",
03   "panl_code": "p",
04   "value": "2",
05   "encoded": "page-2"
06 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section.

## Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

## The "active.query" JSON Object

The following URL:

[http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/qo+/-](http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/qo+/)

Has the following active filters set:

- **Query (*q*)**: hexagonal
- **Query operand (*o*)**: q.op = AND [This is the active filter to remove]

```

01 {
02   "remove_uri": "/hexagonal/q/",
03   "panl_code": "o",
04   "value": "+",
05   "encoded": "%2B"
06 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section.

## Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

## The "active.query" JSON Object

The following URL:

[http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/qo+/?](http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/qo+/)

Has the following active filters set:

- **Query (*q*)**: hexagonal [This is the active filter to remove]
- **Query operand (*o*)**: q.op = AND

```

01 {
02   "remove_uri": "/o+/",
03   "panl_code": "q",
04   "value": "hexagonal",
05   "encoded": "hexagonal"
06 }
```

**Note:** Lines in *italics* above have already been described in the "active" JSON Object section.

## Integration and Implementation

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tags's `href` attribute and the `encoded` value for the anchor text.

## The "active.sort" JSON Array

The sorting active filter contains the information to display the value and the removal URI. The object will have additional keys of `inverse_uri` and `is_descending`.

The following URL:

[http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/sb-q/?](http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/hexagonal/sb-q/)

Has the following active filters set:

- **Query (*q*)** : hexagonal
- **Sorting (*s*)**: Sorting by brand descending [ This is the active filter to remove / invert ]

```

01 {
02   "facet_name": "brand",
03   "inverse_uri": "/hexagonal/sb+q/",
04   "is_descending": true,
05   "encoded": "Brand",
06   "name": "Brand",
07   "panl_code": "s",
08   "remove_uri": "/hexagonal/q/"
}

```

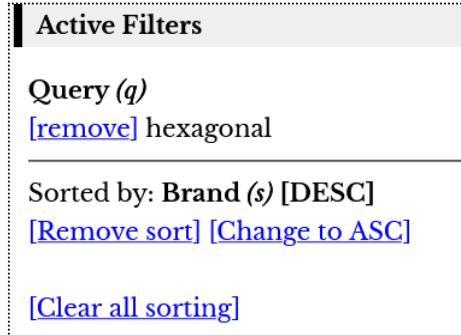
**Note:** Lines in *italics* above have already been described in the "active" JSON Object section.

## Integration and Implementation

Like the BOOLEAN facet, this has an `inverse_uri` which will allow you to generate a link that will invert the sort order, without interfering with any further sort orderings.

The `is_descending` key is a boolean value which can be used to generate the inverse link text.

The implementation details are as per the [Overall Integration and Implementation](#) details section for facets, i.e. render the `remove_uri` as the anchor tag's `href` attribute and the `encoded` value for the anchor text.



**Image:** The In-Build Panl Results Viewer web app showing a sort order of Brand descending, with the remove link, the change to ascending link, and the clear all sorting link.



**Notes:** The clear all sorting link above is generated from the `panl.sorting.remove_uri` JSON object path.

## The "available" JSON Object

This JSON object contains at maximum two JSON arrays (Note that the `range_facets` and `date_range_facets` JSON arrays will be empty if no facet was defined as a RANGE or DATE Range facet):

```

01 {
02   "date_range_facets": [ ... ],
03   "facets": [ ... ],
04   "range_facets": [ ... ]
05 }
```

### Line 2:

An array of facet objects that are available for DATE Range facets, this will be an empty array if no DATE Range facets have been configured.

**Line 2:**

An array of facet objects that are available for filtering the results

**Line 3:**

An array of facets that are configured as RANGE facets in the, this will be



**Notes:** The `date_range_facets` and `range_facets` key are **\_\_ALWAYS\_\_** available, irrespective of any other facets that have been chosen.

Additionally, any LPSE code that is defined as a RANGE facet will also be included in the facets array as a regular facet, if those facets are available.

## The "available.date\_range\_facets" JSON Array

For each of the DATE Range facet objects, the following information is available:

```

01 {
02   "next": "next+",
03   "uris": {
04     "before": "/",
05     "after": "/S/"
06   },
07   "designators": {
08     "hours": "+hours",
09     "months": "+months",
10     "days": "+days",
11     "years": "+years"
12   },
13   "previous": "previous+",
14   "facet_name": "solr_date",
15   "name": "Solr Date",

```

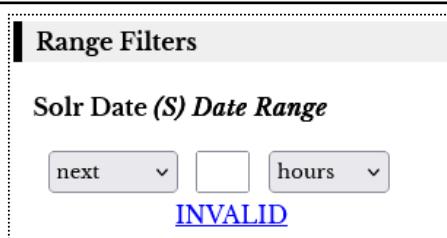
```

16    "panl_code": "S"
17}

```

## Integration and Implementation

In the Panl Results Viewer, this is implemented as two drop downs (one for each of the range indicator and range type) and a text field for the value.



*Image: The In-Build Panl Results Viewer web app showing the implementation of the DATE Range facet.*

Of course, you may implement your interface in any way that you desire.

To generate the URI,

1. Start with the `uris.before` value (**Line 4**)
2. Append the URL encoded `next` or `previous` value (**Line 2 or 13**)
3. Append the integer value that is entered
4. Append one of the designators (e.g. `designators.months`) (**Line 8 to 11**)
5. Append the `uris.after` value (**Line 5**)

Render the above value as the `href` attribute for the link.



**Note:** There can only ever be one DATE Range for the specified facet, and will ALWAYS appear in the `available.date_range_facets` array. The URI will always be a replacement URI not an additive URI - i.e. if you already have a DATE Range facet selected, the generated URI will replace the current one.

## The "available.facets" JSON Array

For each of the facet JSON objects in the `available.facets` JSON array

An example of the brand available facet

```

01 {
02   "facet_name": "brand",
03   "is_or_facet": false,
04   "is_range_facet": false,
05   "name": "Brand",
06   "panl_code": "b",
07   "uris": {
08     "before": "/",
09     "after": "/b/"
10   },
11   "values": [
12     {
13       "count": 5,
14       "value": "Faber-Castell",
15       "encoded": "Manufactured+by+Faber-Castell+Company"
16     },
17     ...
18   ]
19 }
```

This JSON array contains a list of all available facets (including names, encoded names, and counts) for the search, with links to add this facet to the search query.

## Integration and Implementation

To render the link on the search page, iterate through the values array (which is already sorted as per the Panl configuration) and generate the link with the following information.

1. Start with the `uris.before` value (**Line 8**)
2. Append the `values[i].encoded` value (**Line 15**)
3. Append the `uris.after` value (**Line 9**)

To render the value of the anchor tag use the encoded value and URL decode it.

## The "available.range\_facets" JSON Object

Range facts allow a variety of URI paths to be generated from the same dataset, the example included below is from the Mechanical Pencils collection.

```

01 {
02   "uris": {
03     "before": "/weighing+from+",
04     "before_min_value": "/from+light",
05     "has_infix": true,
06     "after_max_value": "heavy+pencils/w-w/",
07     "during": "+to+",
08     "after": "+grams/w-w/"
09   },
10   "min": "10",
11   "max": "50",
12   "prefix": "",
13   "range_min_value": "from+light",
14   "facet_name": "weight",
15   "name": "Weight",
16   "panl_code": "w",
17   "suffix": "+grams",
18   "range_max_value": "heavy+pencils"
19 }
```

## Implementation and Integration

RANGE facets have a variety of configuration options, and depending on the requirements of the URI path generation will depend on what properties need to be set.



**IMPORTANT:** Range facets will ALWAYS be returned in the results.

### Without an infix

Depending on the properties that are set, all of the following URI paths are equivalent and will return exactly the same results. without an infix (this defaults to the `~` character) are equivalent and will return exactly the same results:

- `/10~50/w+w/`
- `/10+grams~50+grams/w+w/`
- `/weight+10~weight+50/w+w/`
- `/weight+10+grams~weight+50+grams/w+w/`
- `/from+light~heavy+pencils/w+w/`
- `/from+light~50/w+w/`
- `/from+light~50+grams/w+w/`
- `/from+light~weight+50+grams/w+w/`
- `/from+light~weight+50/w+w/`
- `/10~heavy+pencils/w+w/`
- `/10+grams~heavy+pencils/w+w/`
- `/weight+10+grams~heavy+pencils/w+w/`
- `/weight+10~heavy+pencils/w+w/`

## With an infix

Depending on the properties that are set, all of the following URI paths with an infix (set to `+to+`) are equivalent and will return the exactly the same results:

- `/10+to+50/w-w/`
- `/weight+10+to+weight+50/w-w/`
- `/weight+10+grams+to+weight+50+grams/w-w/`
- `/10+grams+to+50+grams/w-w/`
- `/from+light+to+50/w-w/`
- `/from+light+to+weight+50/w-w/`
- `/from+light+to+weight+50+grams/w-w/`
- `/from+light+to+50+grams/w-w/`
- `/from+light+to+heavy+pencils/w-w/`
- `/10+to+heavy+pencils/w-w/`
- `/weight+10+to+heavy+pencils/w-w/`
- `/weight+10+grams+to+heavy+pencils/w-w/`
- `/10+grams+to+heavy+pencils/w-w/`
- `/weighing+from+10+to+50+in+grams/w-w/`
- `/weighing+from+10+to+50/w-w/`

- /weighing+from+10+to+weight+50/w-w/
- /weighing+from+10+to+weight+50+grams/w-w/
- /weighing+from+10+to+50+grams/w-w/
- /10+to+50+in+grams/w-w/
- /weight+10+to+50+in+grams/w-w/
- /weight+10+grams+to+50+in+grams/w-w/
- /10+grams+to+50+in+grams/w-w/
- /10+to+50+in+grams/w-w/

### The order of precedence with an infix set

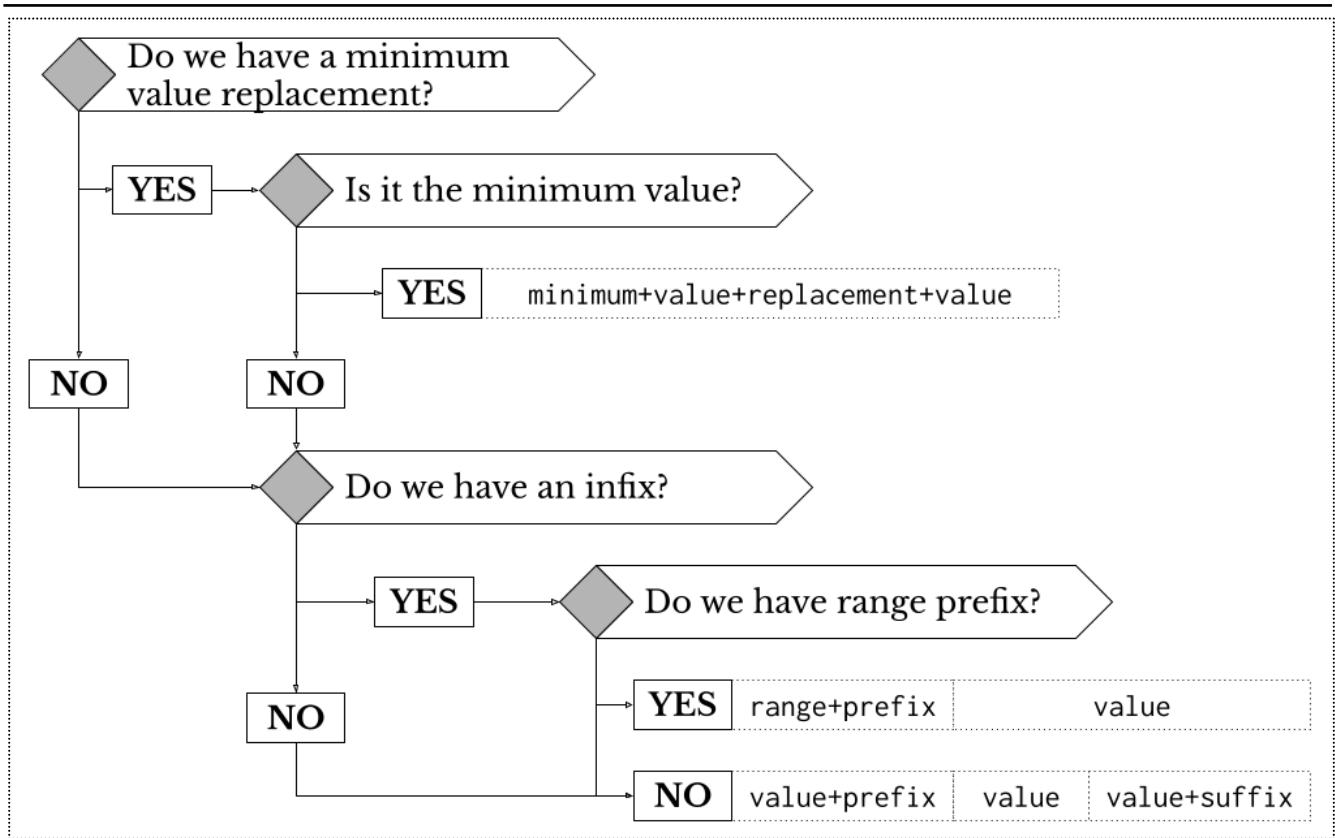
The order of precedence for the text **before** the infix is:

- If it exists, the minimum range value replacement  
~ **else** ~
- If there is a range prefix, the prefix + the 'from' value  
~ **else** ~
- If it a value prefix or value suffix exists, the value prefix + value + value suffix  
~ **otherwise** ~
- Just the value

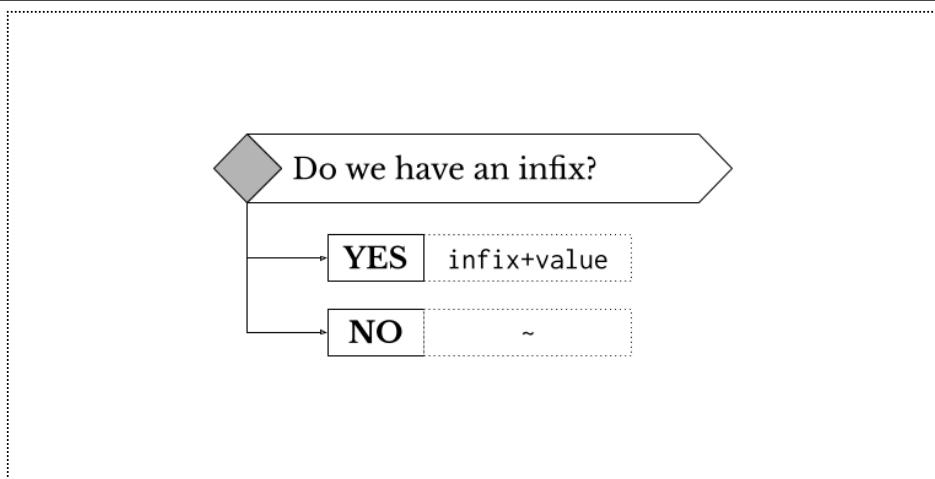
The order of precedence for the text **after** the infix is:

- If it exists, the maximum range value replacement  
~ **else** ~
- If there is a range suffix, the value + the range suffix  
~ **else** ~
- If it a value prefix or value suffix exists, the value prefix + value + value suffix  
~ **otherwise** ~
- Just the value

The flow chart below shows the logic for how the complete URI path for how a range facet is determined and generated



*Flowchart: Logic for generation of the URI path values for range prefixes*



*Flowchart: Logic for generation of the URI path values for range prefixes*

The order of precedence without an infix set

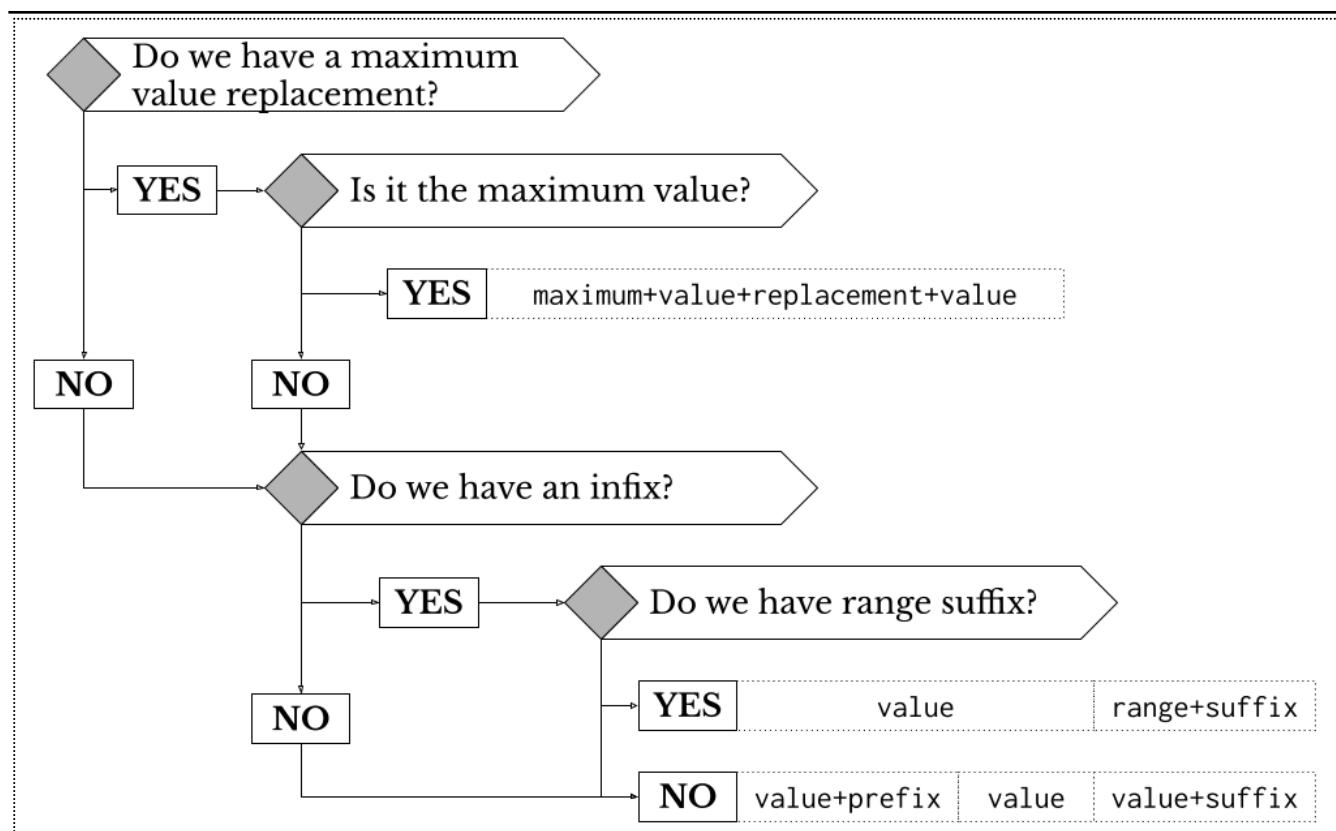
The order of precedence for the text without an infix set is:

- If it exists, the minimum range value replacement, or
  - If it exists, the value prefix
  - The value
  - If it exists, the value suffix

The order of precedence for the text after the infix is:

- If it exists, the maximum range value replacement, or
  - If it exists, the value prefix
  - The 'to' value
  - If it exists, the value suffix

The flow chart below shows the logic for how the complete URI path for how a range facet is determined and generated



*Flowchart: Logic for generation of the URI path values for range suffixes*

## The "sorting" JSON Object

The following URL will return the sorting JSON object listed below.

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/11+grams/hexagonal/wsN+q/>

The above query decodes to:

- A weight (w) of 11 (with a suffix of grams) - note the white space before grams),
- A search query of hexagonal, and
- Sorting by name (N) ascending (+)

```

01 {
02   "remove_uri": "/11+grams/hexagonal/wq/",
03   "fields": [
04     {
05       "name": "Brand",
06       "facet_name": "brand",
07       "set_uri_asc": "/11+grams/hexagonal/wsb+q/",
08       "set_uri_desc": "/11+grams/hexagonal/wsb-q/",
09       "add_uri_asc": "/11+grams/hexagonal/wsN+sb+q/"
10       "add_uri_desc": "/11+grams/hexagonal/wsN+sb-q/",
11     },
12     {
13       "name": "Name",
14       "facet_name": "name",
15       "set_uri_desc": "/11+grams/hexagonal/wsN-q/",
16       "set_uri_asc": "/11+grams/hexagonal/wsN+q/"
17     }
18   ]
19 }
```

**Looking at the response:**

If you wanted to remove all sorting:

- `remove_uri` is the URI path to reset to no sorting - which is the default Solr sort of relevance descending (**Line 2**)

If you wanted to replace the sorting - i.e. remove the `name` sorting, and replace it with `brand` sorting:

- `set_uri_asc` is the URI path for ascending (**Line 7**)
- `set_uri_desc` is the URI path for descending (**Line 8**)

If you wanted to add sorting by `brand` in addition to the current sort of `name` - i.e. sort first by `name` ascending then by `brand`:

- `add_uri_asc` is the URI path for sorting by the current sort order, then by brand ascending (**Line 9**)
- `add_uri_desc` is the URI path for sorting by the current sort order, then by brand descending (**Line 10**)

## Implementation and Integration

The screenshot shows a search interface with the following elements:

- Active Filters:**
  - Query (q)**: [remove] hexagonal
  - Weight (w)**: [remove] 11 grams
- Sort Options:**
  - Sort by Brand: ASC DESC || Name: ASC DESC** (highlighted with a red box labeled 1)
  - Then sort by Brand: ASC DESC** (highlighted with a red box labeled 2)
  - « PREV NEXT »**
  - Show 3 5 10 per page**
- Solr/Panl Metrics:**
  - Solr:** query time 62ms.
  - Panl:** parse request 0ms, build request 54ms, send and receive request 177ms, parse response 1ms. Total time 234ms.
- Facets:**
  - Name (name)**: 5218
  - Brand (brand)**: Koh-i-Noor
- Sorting Buttons:**
  - Sorted by: Name (s) [ASC]** (highlighted with a red box labeled 3)
  - [Remove sort] [Change to DESC]**
  - [Clear all sorting]**

**Image:** The In-Built Panl Results Viewer web app showing the rendering of sorting options, additional sorting options and active facets.

1. **Initial sorting options** - a list of all available initial sort fields with links to both ascending and descending.



**Note:** The order of the initial sorting options will match the order that is defined by the `panl.sort.fields` property

2. **Additional sorting options** - a list of additional sorting options available with links to both ascending and descending.



**Note:** The additional sorting options will only be rendered if an initial sort option has been selected.

3. **Active sorting options** - for sorting options that have been applied, they will be listed in order of application, including links to
  - a. Remove this sorting option (keeping any other sorting options that have been applied)
  - b. Invert the sorting option (i.e. if the sort order is ascending, change it to descending, and vice versa)
  - c. Clear all of the sorting options

## Rendering the Initial Sorting Options

1. Iterate through the `available.sorting.fields` JSON array
2. Render the value keyed on `name` ⇒ output `Name`
3. For the ascending link render the

```
<CaFUPs><set_uri_asc> ⇒ output  
/mechanical-pencils/brandandname/11+grams/hexagonal/wsb+q/
```

For the descending link render the

```
<CaFUPs><set_uri_desc> ⇒ output  
/mechanical-pencils/brandandname/11+grams/hexagonal/wsb-q/
```



**Note:** To determine whether the link is currently active, the `active.sort` array will need to be interrogated where `facet_name=<this_facet_name>` and then conditionally display the URI path dependent on the value of the `is_descending` key.

## Rendering the Additional Sorting Options

1. Iterate through the `available.sorting.fields` JSON array
2. If the `set_uri_asc` or `set_uri_desc` key exists then
  - a. Render the value keyed on `name` ⇒ output `Name`
  - b. For the ascending link render the

```
<CaFUPs><add_uri_asc> ⇒ output
```

```
/mechanical-pencils/brandandname/11+grams/hexagonal/wsN+sb+q/
```

For the descending link render the

```
<CaFUPs><add_uri_desc> ⇒ output
```

```
/mechanical-pencils/brandandname/11+grams/hexagonal/wsN+sb+q/
```

## Example of Multi Sorting Display



**Notes:** For this example, an additional Panl sort field was added, i.e. the property is now `panl.sort.fields=brand,name,weight`

**Search Query**

Search

**Active Filters**

**Range Filters**

**Search Query**

Search

**Active Filters**

Sorted by: Weight (s) [DESC] [\[Remove sort\]](#) [\[Change to ASC\]](#)

[\[Clear all sorting\]](#)

**Search Query**

Search

**Active Filters**

Sorted by: Weight (s) [DESC] [\[Remove sort\]](#) [\[Change to ASC\]](#)

Sorted by: Brand (s) [ASC] [\[Remove sort\]](#) [\[Change to DESC\]](#)

[\[Clear all sorting\]](#)

**Search Query**

Search

**Active Filters**

Sorted by: Weight (s) [DESC] [\[Remove sort\]](#) [\[Change to ASC\]](#)

Sorted by: Brand (s) [ASC] [\[Remove sort\]](#) [\[Change to DESC\]](#)

Sorted by: Name (s) [DESC] [\[Remove sort\]](#) [\[Change to ASC\]](#)

[\[Clear all sorting\]](#)

**Search Results - Found 55 result(s) (exact)**

q.op AND || 1

Page 1 of 6 Showing 10 results per page This page has 10 result(s)

Sort by Brand: [ASC DESC](#) || Name: [ASC DESC](#) || Weight: [ASC DESC](#)

« PREV NEXT »

Show 3 5 10 per page

**Search Results - Found 55 result(s) (exact)**

q.op AND || 2

Page 1 of 6 Showing 10 results per page This page has 10 result(s)

Sort by Brand: [ASC DESC](#) || Name: [ASC DESC](#) || Weight: [ASC DESC](#)

Then sort by Brand: [ASC DESC](#) || Name: [ASC DESC](#)

« PREV NEXT »

Show 3 5 10 per page

*Solr: query time 41ms.  
Panl: parse request 0ms, build request 79ms, send and receive request 156ms, parse response 1ms. Total time 236ms.*

**Search Results - Found 55 result(s) (exact)**

q.op AND || 3

Page 1 of 6 Showing 10 results per page This page has 10 result(s)

Sort by Brand: [ASC DESC](#) || Name: [ASC DESC](#) || Weight: [ASC DESC](#)

Then sort by Name: [ASC DESC](#)

« PREV NEXT »

Show 3 5 10 per page

*Solr: query time 39ms.  
Panl: parse request 0ms, build request 30ms, send and receive request 77ms, parse response 0ms. Total time 108ms.*

**Name (name)**  
Classic Revolve

**Search Results - Found 55 result(s) (exact)**

q.op AND || 4

Page 1 of 6 Showing 10 results per page This page has 10 result(s)

Sort by Brand: [ASC DESC](#) || Name: [ASC DESC](#) || Weight: [ASC DESC](#)

« PREV NEXT »

Show 3 5 10 per page

*Solr: query time 53ms.  
Panl: parse request 0ms, build request 48ms, send and receive request 114ms, parse response 0ms. Total time 163ms.*

**Name (name)**  
Classic Revolve  
**Brand (brand)**  
YStudio

*Image:* The In-Built Panl Results Viewer webapp showing the rendering of sorting options as they are selected

## 1. No Initial sorting options selected - URL:

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname>

- a. No active filters rendered
2. **Weight descending initial sorting option selected - URL:**  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/sw--/>
- a. `brand` and `name` additional sorting options are available
  - b. Active filters show sorting by
    - i. Weight descending
    - c. Active filters show clear sorting order link
3. **Brand ascending additional sorting option selected - URL:**  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/sw-sb+/>
- a. Only the `name` additional sorting option is now available
  - b. Active filters show sorting by
    - i. Weight descending, then
    - ii. Brand ascending
  - c. Active filters show clear sorting order link
4. **Name descending additional sorting option selected - URL:**  
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/sw-sb+sN-/>
- a. No additional sorting objects are available
  - b. Active filters show sorting by
    - i. Weight descending, then
    - ii. Brand ascending, then
    - iii. Name descending
  - c. Active filters show clear sorting order link

~ ~ ~ \* ~ ~ ~

# Integrating An Existing Solr Schema

---

There are two options to generate the Panl configuration file, namely:

1. Using the Panl Generator
2. Manually creating the configuration files



**Tips:** The Panl generator utility is the easiest and quickest method to generate a file.

## Using the Panl Generator

### Running the Command Line Utility

Included within the Panl package is an interactive quick start generator for working with existing managed-schema.xml files for Solr.

It quickly generates a `panl.properties` file and a `<panl_collection_uri>.panl.properties` file, and links the two.

This command line utility can be invoked with the following commands (ensure that you are in the `PANL_INSTALL_DIRECTORY` when running the commands).

#### \*NIX command

Command(s)
<code>bin/panl generate -schema src/dist/sample/solr/book-store/managed-schema.xml -properties src/dist/sample/panl/book-store/panl.properties</code>

#### Windows command

**Command(s)**

```
bin\panl.bat generate ←
  -schema src\dist\sample\solr\book-store\managed-schema.xml ←
  -properties src\dist\sample\panl\book-store\panl.properties
```

Running the command for your operating system will output the following to the console, prompting for input for the various properties that are required (with sensible defaults suggested).

**Note:** Output has had the logging timestamp removed and process designator removed.

```
01 INFO panl.Main -                                 --
02 INFO panl.Main - .-----.-.-----.|  |
03 INFO panl.Main - | _ | _ | || |
04 INFO panl.Main - | __|___.|__|__||__|
05 INFO panl.Main - |__| ... .-..
06 INFO panl.Main -
07 INFO panl.Main - ~ ~ ~ * ~ ~ ~
08 INFO panl.Main -
09 INFO panl.Main - Starting Panl generation with properties:
10 INFO panl.Main - -properties src/dist/sample/panl/book-store/panl.properties
11 INFO panl.Main - -schema ←
src/dist/sample/solr/book-store/managed-schema.xml
12 INFO panl.Main - -overwrite false
13 INFO panl.Main -
14 INFO panl.Main - ~ ~ ~ * ~ ~ ~
15 INFO panl.Main -
16 Enter the 1 character property value for 'panl.param.query' ←
(The search query parameter), default [q]:
17 Property 'panl.param.query' set to default value of 'q'
18 Enter the 1 character property value for 'panl.param.page' ←
(The page number), default [p]:
19 Property 'panl.param.page' set to default value of 'p'
20 Enter the 1 character property value for 'panl.param.numrows' ←
(The number of results to return per page), default [n]:
21 Property 'panl.param.numrows' set to default value of 'n'
```

```

22 Enter the 1 character property value for 'panl.param.sort' ↵
(The results sorting parameter), default [s]:
23 Property 'panl.param.sort' set to default value of 's'
24 Enter the 1 character property value for 'panl.param.operand' ↵
(The default query operand (q.op)), default [o]:
25 Property 'panl.param.operand' set to default value of 'o'
26 Enter the 1 character property value for 'panl.param.passthrough' ↵
(The URI path passthrough), default [z]:
27 Property 'panl.param.passthrough' set to default value of 'z'

```

This will output the file to the `src\dist\sample\panl\book-store\` directory and will generate two files, the `panl.properties` file and the `<panl_collection_uri>.panl.properties`. From here you may edit the files and set the configuration for your specific use case.

### Lines 9-12:

The Panl generator will output the passed in command line options so that you can confirm that the input and output files are correct

### Lines 16-27:

Will prompt for input from the console. This will check to ensure that

1. There is only one character for the Panl params
2. That the character has not been used before

If the enter key is pressed, then the default value will be used.

If there are any errors, the generator will print out the error and re-prompt to enter the command.

The Panl generator will output its working to the command line and highlight any warnings. A reduced version of the output is below:

```

01 INFO bean.PanlCollection - PanlCollection: book-store
02 INFO bean.PanlCollection - Have 17 panlFields, LPSE length is set to 1
03 INFO bean.PanlCollection - Assigned field 'id' to panl code 'i'
04 INFO bean.PanlCollection - Assigned field 'author' to panl code 'a'
05 INFO bean.PanlCollection - Assigned field 'title' to panl code 't'

```

```

06 INFO bean.PanlCollection - Assigned field 'description' to panl code 'd'
07 INFO bean.PanlCollection - Assigned field 'book_image' to panl code 'b'
08 INFO bean.PanlCollection - Assigned field 'buy_url' to panl code 'B'
09 INFO bean.PanlCollection - Assigned field 'genre' to panl code 'g'
10 INFO bean.PanlCollection - Assigned field 'num_pages' to panl code 'N'
11 INFO bean.PanlCollection - Assigned field 'first_published_year' to panl code 'f'
12 INFO bean.PanlCollection - Assigned field 'language' to panl code 'l'
13 INFO bean.PanlCollection - Assigned field 'is_paperback' to panl code 'I'
14 INFO bean.PanlCollection - Assigned field 'series' to panl code 'S'
15 INFO bean.PanlCollection - Assigned field 'price' to panl code 'P'
16 INFO bean.PanlCollection - Assigned field 'a_to_z_index' to panl code 'A'
17 INFO bean.PanlCollection - Assigned field 'decade_published' to panl code 'D'
18 WARN bean.PanlCollection - No nice panl code for field 'book_length', ←
  'b' and 'B' already taken
19 INFO bean.PanlCollection - Assigned field 'text' to panl code 'T'
20 INFO bean.PanlCollection - Assigned field 'book_length' to RANDOM panl code 'u'
21 INFO generator.PanlGenerator - Writing out file panl.properties
22 INFO generator.PanlGenerator - Done writing out file panl.properties
23 INFO generator.PanlGenerator - Writing out file book-store.panl.properties
24 INFO generator.PanlGenerator - Done writing out file book-store.panl.properties

```

### Line 18 and 20:

This is a warning from the generator that it could not automatically assign a LPSE code to the Solr field `book_length`. What the generator does is look at the first character of the Solr field, and attempt to assign it to that character as a LPSE code, first as a lowercase letter, then as an uppercase letter.

If neither are available, it will assign it a random LPSE code from the available codes - in this case `book_length` was assigned the LPSE code of '`u`'.

## Editing the Generated Files

The `panl.properties` file that is generated is well commented and you will definitely need to edit the `solrj.client` and `solr.search.server.url` properties.

Also, if running in production mode, it is recommended that you set all of the following properties to false:

- panl.results.testing.urls
- panl.status.404.verbose
- panl.status.500.verbose

Moving on to the `<panl_collection_uri>.panl.properties` file, use the properties quick reference section to edit the file to assign prefixes, suffixes, set the type of the facet to a field, or OR, BOOLEAN, RANGE etc.

## Manually Creating the Configuration Files

Whilst not recommended as it can become very time consuming, with a lot of cross-checking required to ensure that the Solr and Panl fields match to LPSE codes, including fields and sorting, you can use the existing files as a starting point, or look at the templates in the github repository.

**The example mechanical pencils panl.properties file:**

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/panl.properties>

The template upon which it was based:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/panl.properties.template>

**The example mechanical-pencils.panl.properties file:**

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/mechanical-pencils.panl.properties>

The template upon which it was based:

[https://github.com/synapticloop/panl/blob/main/src/main/resources/panl\\_collection\\_uri.panl.properties.template](https://github.com/synapticloop/panl/blob/main/src/main/resources/panl_collection_uri.panl.properties.template)



**IMPORTANT:** Ensure that you are using the correct version of the file for your installation. Whilst the properties files should be backwards compatible, there may be additional configuration items which are enabled in later versions and will have no effect on previous versions.

~ ~ ~ \* ~ ~ ~

# Properties Quick Reference

---

In alphabetical order. Properties that exist in the `panl.properties` file are:

- `panl.results.testing.urls`
- `panl.status.404.verbose`
- `panl.status.500.verbose`
- `panl.collection.<solr_collection_name>`
- `solrj.client`
- `solr.search.server.url`

All other properties are contained within the `<panl_collection_uri>.panl.properties` file.

---

## Property References Format Explained

Each of the properties referenced below will have a brief overview description, then a table that summarises the property, its values, and summarises its details.

Where the property key ends with `<lpse_code>` this is the LPSE code which **MUST** match the LPSE length that is set by the `panl.lpse.length` property.

Where the property key ends with `<field_set>` this is the field set .

---

### Scope One of Server, Collection, or Field

- **Server** - applies to the Panl server and every Panl collection URI that the server is registered to handle
  - **Collection** - applies to the Panl collection URI that the file registers (including the fieldsets that are registered)
  - **Field** - applies to a specific field with the registered LPSE code
- 

### Required Either Yes, No, or Optional

- **Yes** - if this property is missing then the Panl server will refuse to start and will exit with an exception, or the Field will not be registered and will not be active
  - **No** - The configuration item that the property controls will be set to a default value
  - **Optional** - Does not have a default value
-

---

**Value** One of Character, String, Integer, Decimal, Boolean, or List<Type>

- **Character** - a single alphanumeric character in the range a-z A-Z 0-9, or in some instances a + or -
- **String** - multi character string
- **Integer** - Any integer number
- **Decimal** - Any decimal number to any number of decimal places
- **Boolean** - may be either true or false, in some instances the value is case sensitive and must be the exact lowercase true or lowercase false.
- **List<Type>** - A comma separated list of values all of which are of type 'Type'. E.g. List<String> would be a comma separated list of Strings

---

**Default** If not set, this will be the default value, N/A indicates that there is no default.

---

**Location** Either

- `panl.properties` file - the server configuration file
- `<panl_collection_uri>.panl.properties` - the individual Panl collection URI file

---

**Notes** 1. An list of notes with brief explanations

---

**See also** Links to other section headings with properties that are of note for this particular property.

For some properties, a further explanation is included.

## **panl.bool.<lpse\_code>.false**

*Set the replacement text for a Solr boolean field for a 'false' value.*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** `false`

---

**Location** `<panl_collection_uri>.panl.properties`

- 
- Notes**
1. If the type of the field (i.e. the `panl.type.<lpse_code>` property value) is not `solr.BoolField`, then this property will be silently ignored.
  2. If the value of this URI path part does not exactly match the replacement value, then this token will be marked as invalid and

- 
- not passed through to the Solr server.
3. A prefix and suffix can also be applied to the field, which will be prepended/appended to this value respectively.

**See also** [BOOLEAN Facet](#)

[panl.type.<lpse\\_code>](#)  
[panl.bool.<lpse\\_code>.true](#)  
[panl.prefix.<lpse\\_code>](#)  
[panl.suffix.<lpse\\_code>](#)

## **panl.bool.<lpse\_code>.true**

*Set the replacement text for a Solr boolean field for a 'true' value.*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** true

---

**Location** <panl\_collection\_uri>.panl.properties

- Notes**
1. If the type of the field (i.e. the panl.type.<lpse\_code> property value) is not solr.BoolField, then this property will be silently ignored.
  2. If the value of this URI path part does not exactly match the replacement value, then this token will be marked as invalid and not passed through to the Solr server.
  3. A prefix and suffix can also be applied to the field, which will be prepended/appended to this value respectively.

**See also** [BOOLEAN Facet](#)

[panl.type.<lpse\\_code>](#)  
[panl.bool.<lpse\\_code>.false](#)  
[panl.prefix.<lpse\\_code>](#)  
[panl.suffix.<lpse\\_code>](#)

## panl.collection.<solr\_collection\_name>

The collection Panl property files either a single file path, or a comma separated list of file paths, to load configuration from that will return results from the <solr\_collection\_name>.

**Scope** Server

**Required** Yes - will produce an ERROR if no properties that start with panl.collection. are defined

**Value** List<String> - A comma separated list of property file names and paths which are relative to the panl.properties file that included it.

**Default** N/A

**Location** panl.properties

**Notes**

1. Multiple files should be comma separated and may be on new lines with the Java properties continuation of backslash \.
2. The base directory for relative files paths is the directory that the panl.properties file resides.

### See also

The file, or comma separated list of property files use the base directory of the panl.properties file as the starting the properties file for the collection to be registered with the Panl server. There is one property per collection that is to be served.

### Example:

If you have a directory structure with multiple properties files:

```
/panl
  /mechanical-pencils
    /mechanical-pencils.panl.properties
    /mechanical-pencils-or.panl.properties
  /products
    /products.panl.properties
  /server
  /panl.properties
```

And wanted to connect to two Solr connections (`mechanical-pencils` and `products`) with two separate CaFUPS then the `panl.properties` file would contain the following two properties:

```
panl.collection.mechanical-pencils=\n\n\n.../mechanical-pencils-or/mechanical-pencils-or.panl.properties,\n.../mechanical-pencils/mechanical-pencils.panl.properties\npanl.collection.products=\n\n.../products/base-products.panl.properties
```

If you start to server with the `-properties panl/server/panl.properties` command line option then the Panl server will read the `panl.properties` file, and then attempt to bind the Panl URI paths from reading the properties files relative to the current directory of the `panl.properties` file.

This would bind the following Panl collection URIs to the respective Solr collections

- Bound URI of `/mechanical-pencils/*` would connect to the `mechanical-pencils` Solr collection
- Bound URI of `/mechanical-pencils-or/*` would connect to the `mechanical-pencils` Solr collection
- Bound URI of `/products/*` would connect to the `base-products` Solr collection

## `panl.date.<lpse_code>.days`

*The suffix that configures the searched date range for this field which is then parsed by Panl to generate the Solr search query based on days. The query is ALWAYS from NOW - as in the current time.*

**Scope** Field

**Required** No

**Value** String

**Default** N/A

**Location** `<panl_collection_uri>.panl.properties`

- |              |  |
|--------------|--|
| <b>Notes</b> | <ol style="list-style-type: none"> <li>1. This works in with conjunction with the next and previous prefixes for DATE range facets</li> <li>2. If this property is not set then NO date range searches will be available for daily ranges</li> </ol> |
|--------------|--|

---

See also [DATE Range Facet](#)

[panl.date.<lpse\\_code>.next](#)  
[panl.date.<lpse\\_code>.previous](#)

[panl.date.<lpse\\_code>.days](#)  
[panl.date.<lpse\\_code>.months](#)  
[panl.date.<lpse\\_code>.years](#)

---

This property sets the suffix text that will indicate to Panl that this date range query is for daily date ranges. This creates a query that in the example URI paths below, the suffix is \ days (note the backslash encoded whitespace for the property). Below are using two example prefixes of `previous+` and `next+` to show the differences between the Solr query string that is generated - the %2B is a URL encoded + sign.

---

URI paths	/previous+5+years/S/ /next+5+years/S/
-----------	--

---

Canonical URI paths	/previous+5+years/S/ /next+5+years/S/
---------------------	--

---

Solr query string	fq=solr_date:[NOW-5YEARS+TO+NOW] fq=solr_date:[NOW+TO+NOW%2B5YEARS]
-------------------	--

---

## panl.date.<lpse\_code>.hours

*The suffix that configures the searched date range for this field which is then parsed by Panl to generate the Solr search query based on hours. The query is ALWAYS from NOW - as in the current time.*

---

Scope	Field
-------	-------

---

Required	No
----------	----

---

Value	String
-------	--------

---

Default	N/A
---------	-----

---

Location	<panl_collection_uri>.panl.properties
----------	---------------------------------------

---

Notes	<ol style="list-style-type: none"> <li>1. This works in conjunction with the next and previous prefixes for DATE range facets</li> <li>2. If this property is not set then NO date range searches will be available for hourly time ranges</li> </ol>
-------	---

---

---

See also [DATE Range Facet](#)

[panl.date.<lpse\\_code>.next](#)  
[panl.date.<lpse\\_code>.previous](#)

[panl.date.<lpse\\_code>.days](#)  
[panl.date.<lpse\\_code>.months](#)  
[panl.date.<lpse\\_code>.years](#)

---

This property sets the suffix text that will indicate to Panl that this date range query is for hourly time ranges. This creates a query that in the example URI paths below, the suffix is \ hours (note the backslash encoded whitespace for the property). Below are using two example prefixes of `previous+` and `next+` to show the differences between the Solr query string that is generated - the %2B is a URL encoded + sign.

---

URI paths	/previous+48+hours/S/ /next+48+hours/S/
-----------	--

---

Canonical URI paths	/previous+48+hours/S/ /next+48+hours/S/
---------------------	--

---

Solr query string	<code>fq=solr_date:[NOW-48HOURS+TO+NOW]</code> <code>fq=solr_date:[NOW+TO+NOW%2B48HOURS]</code>
-------------------	--

---

## panl.date.<lpse\_code>.months

*The suffix that configures the searched date range for this field which is then parsed by Panl to generate the Solr search query based on months. The query is ALWAYS from NOW - as in the current time.*

---

Scope	Field
-------	-------

---

Required	No
----------	----

---

Value	String
-------	--------

---

Default	N/A
---------	-----

---

Location	<panl_collection_uri>.panl.properties
----------	---------------------------------------

---

Notes	<ol style="list-style-type: none"> <li>1. This works in conjunction with the next and previous prefixes for DATE range facets</li> <li>2. If this property is not set then NO date range searches will be available for monthly date ranges</li> </ol>
-------	--

---

See also [DATE Range Facet](#)

---

---

`panl.date.<lpse_code>.next`  
`panl.date.<lpse_code>.previous`

`panl.date.<lpse_code>.hours`  
`panl.date.<lpse_code>.days`  
`panl.date.<lpse_code>.years`

---

This property sets the suffix text that will indicate to Panl that this date range query is for monthly date ranges. This creates a query that in the example URI paths below, the suffix is \ months (note the backslash encoded whitespace for the property). Below are using two example prefixes of `previous+` and `next+` to show the differences between the Solr query string that is generated - the %2B is a URL encoded + sign.

---

**URI paths** /previous+12+months/S/  
/next+12+months/S/

---

**Canonical URI paths** /previous+12+months/S/  
/next+12+months/S/

---

**Solr query string** fq=solr\_date:[NOW-12MONTHS+TO+NOW]  
fq=solr\_date:[NOW+TO+NOW%2B12MONTHS]

---

## **panl.date.<lpse\_code>.next**

*The prefix to indicate that Panl should interpret this as a date range query for the following (or next) date period. The query is ALWAYS from NOW - as in the current time.*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

1. This works in conjunction with the hours, days, months and years suffixes.
2. If this property is not set then NO date range searches will be available for time periods from now

---

**See also** [DATE Range Facet](#)

---

---

**panl.date.<lpse\_code>.previous**  
**panl.date.<lpse\_code>.hours**

**panl.date.<lpse\_code>.days**  
**panl.date.<lpse\_code>.months**  
**panl.date.<lpse\_code>.years**

---

This property sets the prefix text that will indicate to Panl that this date range query is for time ranges from NOW until the indicated suffix. This creates a query that in the example URI paths below, the prefix is `next` (note the whitespace for the property). Below are using the example suffixes of `\ hours`, `\ days`, `\ months`, and `\ years` to show the differences between the Solr query string that is generated - the `%2B` is a URL encoded `+` sign.

<b>URI paths</b>	/next+48+hours/S/ /next+30+days/S/ /next+12+months/S/ /next+5+years/S/
<b>Canonical URI paths</b>	/next+48+hours/S/ /next+30+days/S/ /next+12+months/S/ /next+5+years/S/
<b>Solr query string</b>	<code>fq=solr_date:[NOW+TO+NOW%2B48HOURS]</code> <code>fq=solr_date:[NOW+TO+NOW%2B30DAYS]</code> <code>fq=solr_date:[NOW+TO+NOW%2B12MONTHS]</code> <code>fq=solr_date:[NOW+TO+NOW%2B5MONTHS]</code>

---

## **panl.date.<lpse\_code>.previous**

*The prefix to indicate that Panl should interpret this as a date range query for the previous date period. The query is ALWAYS from NOW - as in the current time.*

<b>Scope</b>	Field
<b>Required</b>	No
<b>Value</b>	String
<b>Default</b>	N/A
<b>Location</b>	<code>&lt;panl_collection_uri&gt;.panl.properties</code>

---

- 
- Notes**
1. This works in conjunction with the hours, days, months and years suffixes.
  2. If this property is not set then **NO** date range searches will be available for time periods from now
- 

See also [DATE Range Facet](#)

[panl.date.<lpse\\_code>.next](#)

[panl.date.<lpse\\_code>.hours](#)

[panl.date.<lpse\\_code>.days](#)

[panl.date.<lpse\\_code>.months](#)

[panl.date.<lpse\\_code>.years](#)

---

This property sets the prefix text that will indicate to Panl that this date range query is for time ranges from the indicated suffix until NOW. This creates a query that in the example URI paths below, the prefix is `previous` (note the whitespace for the property). Below are using the example suffixes of \ hours, \ days, \ months, and \ years to show the differences between the Solr query string that is generated - the %2B is a URL encoded + sign.

---

<b>URI paths</b>	/previous+48+hours/S/ /previous+30+days/S/ /previous+12+months/S/ /previous+5+years/S/
------------------	---

---

<b>Canonical URI paths</b>	/previous+48+hours/S/ /previous+30+days/S/ /previous+12+months/S/ /previous+5+years/S/
----------------------------	---

---

<b>Solr query string</b>	fq=solr_date:[NOW-48HOURS+TO+NOW] fq=solr_date:[NOW-30DAYS+TO+NOW] fq=solr_date:[NOW-12MONTHS+TO+NOW] fq=solr_date:[NOW-5MONTHS+TO+NOW]
--------------------------	--

---

## **panl.date.<lpse\_code>.years**

*The suffix that configures the searched date range for this field which is then parsed by Panl to generate the Solr search query based on years. The query is ALWAYS from NOW - as in the current time.*

---

**Scope Field**

---

**Required** No

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

- Notes**
1. This works in conjunction with the next and previous prefixes for DATE range facets
  2. If this property is not set then **NO** date range searches will be available for yearly date ranges
- 

**See also** [DATE Range Facet](#)

[panl.date.<lpse\\_code>.next](#)  
[panl.date.<lpse\\_code>.previous](#)

[panl.date.<lpse\\_code>.hours](#)  
[panl.date.<lpse\\_code>.days](#)  
[panl.date.<lpse\\_code>.months](#)

---

This property sets the suffix text that will indicate to Panl that this date range query is for yearly date ranges. This creates a query that in the example URI paths below, the suffix is \ years (note the backslash encoded whitespace for the property). Below are using two example prefixes of `previous+` and `next+` to show the differences between the Solr query string that is generated - the %2B is a URL encoded + sign.

---

**URI paths** /previous+5+years/S/  
/next+5+years/S/

---

**Canonical URI paths** /previous+5+years/S/  
/next+5+years/S/

---

**Solr query string** fq=solr\_date:[NOW-5YEARS+TO+NOW]  
fq=solr\_date:[NOW+TO+NOW%2B5YEARS]

---

**panl.facet.<lpse\_code>**

*Register a Solr field as a short-hand Panl LPSE code that can be used to select facets for.*

---

**Scope** Collection

---

**Required** No

---

---

**Value Character** - see the LPSE length as this may be multiple characters

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

1. **MUST** match a Solr field named in the Solr managed schema XML file. This is not checked at instantiation, however will cause a runtime exception if the value does not match a Solr field.
  2. The number of characters the <lpse\_code> **MUST** match the LPSE length
- 

**See also** [panl.field.<lpse\\_code>](#)  
[panl.or.facet.<lpse\\_code>](#)  
[panl.range.facet.<lpse\\_code>](#)  
[panl.prefix.<lpse\\_code>](#)  
[panl.suffix.<lpse\\_code>](#)

---

This will configure Panl to serve this Solr field as a selectable facet. As a Regular facet Panl will automatically generate URI paths, including if this is a multi-valued field. There are additional configuration options which are dependent on the type of the Solr field (which would allow BOOLEAN and DATE Range facets), and whether this is configured to be an OR facet or a RANGE facet.

For regular facets selecting a multi-valued 'Colour' facet, firstly selecting a 'Blue' colour, then a 'Red' colour.

---

**URI paths** /Blue/W/  
/Blue/Red/WW/

---

**Canonical URI paths** /Blue/W/  
/Blue/Red/WW/

---

**Solr query string** fq=colours:"Blue"  
fq=colours:"Blue"&fq=colours:"Red"

---

## panl.facetsort.<lpse\_code>

*Set the way that Solr will sort the returned facets.*

---

**Scope** Collection

---

---

**Required** No

---

**Value** index or count

---

**Default** count

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also**

---

This affects the order in which the values and counts of a specific facet are returned. This does not affect the ordering of the LPSE code, or the order of the documents. If the value of this property is set to count, which is the default, then the facet is sorted by the number of documents that contain this facet value. If the property is set to index, then the facet is sorted by the value of the facet.

Below is an image showing the different sorting options for the facet.

Authors (A-Z) (4) <a href="#">[add]</a> B (2) <a href="#">[add]</a> C (11) <a href="#">[add]</a> D (3) <a href="#">[add]</a> M (6)	Authors (A-Z) (4) <a href="#">[add]</a> C (11) <a href="#">[add]</a> M (6) <a href="#">[add]</a> D (3) <a href="#">[add]</a> B (2)
--	--

*Image:* Images showing the difference between sorting on index (left), and count (right).

---

**panl.field.<lpse\_code>**

Register a Solr field as a Panl field that can be returned within the document results. This field will only be returned in the document if it is part of the FieldSets defined in the properties file.

---

**Scope** Collection

---

**Required** No

---

**Value** Character - see the LPSE length as this may be multiple characters

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

- 
- Notes**
1. **MUST** match a Solr field named in the Solr managed schema XML file. This is not checked at instantiation, however will cause a runtime exception if the value does not match a Solr field.
  2. The number of characters the <lpse\_code> **MUST** match the LPSE length
  3. This Solr field will not be returned as a facet, however it can be returned within the document results
- 

**See also** [panl.facet.<lpse\\_code>](#)  
[panl.results.fields.<field\\_set>](#)

---

Fields are never returned as the list of facets that are returned, however they can be returned in the resulting documents and used a sorting option.

## panl.form.query.respondto

*The default query URL parameter that the Panl server will use to generate the text or phrase query to send through to the Solr search server.*

---

**Scope** Collection

---

**Required** No

---

**Value** String

---

**Default** q

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also**

---

## panl.include.same.number.facets

*Whether to include facets in the available facet list if the value of the facet count is the same as the number of documents.*

---

**Scope** Collection

---

**Required** No

---

**Values** Boolean

---

---

**Default** `false`

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes**

---

**See also** [`panl.include.same.number.facets`](#)

---

## panl.include.single.facets

*Whether to include facets that only have a single value, i.e. for a specific facet value there is only one count for it.*

---

**Scope** Collection

---

**Required** No

---

**Value** Boolean

---

**Default** `false`

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes**

---

**See also**

---

If the property `panl.include.same.number.facets` is set to true then the facet values will not appear as singular facet.

If you have set `panl.include.same.number.facets=true` then this property takes effect.

## panl.lpse.ignore

*A comma separated list of LPSE codes to ignore when returning facets from the Solr search server.*

---

**Scope** Collection

---

**Required** No

---

**Value** List<Character> (or a list of LPSE codes)

---

**Default** N/A

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes**

1. This is a comma separated list of LPSE codes to be ignored

**See also**

Ignored LPSE codes will be ignored and not returned in either the active or available filters JSON object.

This is useful when you want to use a facet for a lookup, but not allow the facet to be removed or returned in the list of available or active facets. An example usage would be when generating a link to a single results (say with its id field) and the returned available facets would return all of the ids as facets, which would be unproductive.

## **panl.lpse.length**

*The length of the LPSE code for all facets and fields - this **DOES NOT** affect the length of Panl parameters or operands which are **ALWAYS** a length of 1.*

**Scope** Collection

**Required** No - a WARNING message will be printed to the console

**Values** Integer**Default** 1

**Location** <panl\_collection\_uri>.panl.properties

**Notes****See also**

The LPSE length will determine

- The number of facets that the Panl server is able to use when communicating to the Solr search server.
- The length of the LPSE code that is then placed in the URI path

## **panl.lpse.order**

*The order of the Panl LPSE codes to generate the URI path with.*

**Scope** Collection

**Required** Yes, the Panl server will ERROR if this property does not exist, and exit.

---

If one of the LPSE codes is contained within the list of values, but is not defined, then the Panl server will generate a WARNING message

---

**Value** List<String>

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

1. When the canonical URI path is being generated, the LPSE order is used.
2. The list of LPSE codes must match the defined LPSE codes for the panl.facet.<lpse\_code> definitions

**See also** [panl.lpse.length](#)

---

## panl.name.<lpse\_code>

*The display name for the Solr field that will be presented to the user.*

---

**Scope** Collection

---

**Required** No

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

1. If this is not included then the Solr field name will be used - i.e. the panl.facet.<lpse\_code>.

**See also** [panl.facet.<lpse\\_code>](#)

---

This is the name of the field, as opposed to the Solr field name. This name can be used as nicer text to be displayed on the search page. For example with the Book Store collection, the solr\_field is named first\_published\_year, however when rendered to the search page, the Panl name is used as 'First Published Year'.

**First Published Year (*f*)**

- [\[add\]](#) First published in 2008 (2)
- [\[add\]](#) First published in 2000 (1)
- [\[add\]](#) First published in 2002 (1)
- [\[add\]](#) First published in 2005 (1)
- [\[add\]](#) First published in 2006 (1)
- [\[add\]](#) First published in 2007 (1)
- [\[add\]](#) First published in 2009 (1)

*Image:* Showing the Panl field name that is rendered, which is distinct from the Solr field name.

**panl.or.facet.<lpse\_code>**

*Set this facet to be an OR facet, allowing multiple facet values to be selected for this facet.*

**Scope** Collection

**Required** No

**Value** Boolean

**Default** false

**Location** <panl\_collection\_uri>.panl.properties

**Notes**

**See also**

**panl.param.numrows**

*The LPSE code for the number of rows of documents to return with the search.*

**Scope** Collection

**Required** Yes

**Value** Integer

**Default** n

**Location** <panl\_collection\_uri>.panl.properties

**Notes** 1. On Panl Server property file generation, the default value is set

---

to 'n', however, it may be changed to another single character value.

2. For this to be active, this param must be set the in the comma separated list of the Panl LPSE order

---

**See also** [panl.param.numrows.prefix](#)  
[panl.param.numrows.suffix](#)  
[solr.numrows.default](#)

---

## panl.param.numrows.prefix

*The prefix to put before the numrows URI value.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also** [panl.param.numrows](#)  
[panl.param.numrows.suffix](#)  
[solr.numrows.default](#)

---

## panl.param.numrows.suffix

*The suffix to put after the numrows URI value.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also** [panl.param.numrows](#)

---

---

**panl.param.numrows.prefix**  
**solr.numrows.default**

---

## panl.param.page

*Set the page number for the results, the first page being page 1.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

1. On Panl Server property file generation, the default value is set to 'p', however, it may be changed to another single character value.
2. For this to be active, this param must be set the in the comma separated list of the Panl LPSE order

---

**See also** [panl.param.page.prefix](#)  
[panl.param.page.suffix](#)  
[panl.param.numrows](#)  
[panl.lpse.order](#)

---

## panl.param.page.prefix

*The prefix to place before the page number in the LPSE URI path.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also** [panl.param.page](#)  
[panl.param.page.suffix](#)

---

## **panl.param.page.suffix**

*The suffix to place before the page number in the LPSE URI path.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

See also [panl.param.page](#)  
[panl.param.page.prefix](#)  
[panl.lpse.order](#)

## **panl.param.passthrough**

*The LPSE code for URI values that are passed through the Panl server without any processing or sending through to the Solr search server - i.e. ignore it.*

---

**Scope** Collection

---

**Required** No

---

**Value** Character

---

**Default** z

---

**Location** <panl\_collection\_uri>.panl.properties

**Notes**

1. On Panl Server property file generation, the default value is set to 'z', however, it may be removed from the properties file, or changed to another single character value.
2. For this LPSE code to be active, this param must be set the in the comma separated list of the Panl LPSE order

See also [panl.lpse.order](#)  
[panl.param.passthrough.canonical](#)

## panl.param.passthrough.canonical

*Whether to include the pass through value in the canonical URI generation.*

---

**Scope** Collection

---

**Required** Optional

---

**Value** Boolean

---

**Default** `false`

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes**

<b>See also</b>	<a href="#"><u>panl.lpse.order</u></a>
	<a href="#"><u>panl.param.passthrough</u></a>

## panl.param.query

*The LPSE code for any user supplied queries (i.e. the text or phrase search query)*

---

**Scope** Collection

---

**Required** Yes

---

**Value** Character

---

**Default** `q`

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes**

- 1. On Panl Server property file generation, the default value is set to '`q`', however, it may be removed from the properties file, or changed to another single character value.
- 2. This will always be a single character. This DOES NOT conform to the LPSE length. If the LPSE length is set to 2, then this property value will still be one alphanumeric character long

---

<b>See also</b>	<a href="#"><u>panl.lpse.length</u></a>
-----------------	---

## panl.param.query.operand

*The LPSE code set set the Query Operand for the Solr search server.*

---

**Scope** Collection

---

**Required** Yes

---

**Value** Character

---

**Default** o

---

**Location** <panl\_collection\_uri>.panl.properties

---

- Notes**
1. On Panl Server property file generation, the default value is set to 'o', however, it may be removed from the properties file, or changed to another single character value.
  2. The default value is the one that is assigned by the Panl generator and used throughout the book.

---

**See also** [solr.default.query.operand](#)

---

## panl.param.sort

*Defines the LPSE code for the sorting of the documents*

---

**Scope** Server

---

**Required** Yes

---

**Value** Character

---

**Default** s

---

**Location** <panl\_collection\_uri>.panl.properties

---

- Notes**
1. On Panl Server property file generation, the default value is set to 's', however, it may be removed from the properties file, or changed to another single character value.
  2. The default value is the one that is assigned by the Panl generator and used throughout the book.

---

**See also**

---

## **panl.prefix.<lpse\_code>**

*Set the prefix for the facet value which is prepended to the value in the URI path.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also**

## **panl.range.facet.<lpse\_code>**

*Sets this Panl field to be a RANGE facet, allowing the user to select values that fall within an inclusive range.*

---

**Scope** Field

---

**Required** No

---

**Value** Boolean

---

**Default** false

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 3. RANGE facets also return the individual facets within the results.

---

**See also** RANGE Facet

[panl.range.prefix.<lpse\\_code>](#)

[panl.range.infix.<lpse\\_code>](#)

[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)

[panl.range.min.value.<lpse\\_code>](#)

[panl.range.min.wildcard.<lpse\\_code>](#)

---

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

---

## **panl.range.infix.<lpse\_code>**

*Set the infix that appears between the minimum and maximum values that a range can have*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** ~

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

- 4. If the panl.range.facet.<lpse\_code> is not set to true, then this field will be silently ignored.
- 5. If not included, is an empty or blank value, it will default to the tilde character.
- 6. This value CAN NOT be a hyphen/minus (i.e. '-') character.

**See also** [\*\*RANGE Facet\*\*](#)

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

---

## **panl.range.max.<lpse\_code>**

*Set the maximum selectable value for the range.*

---

**Scope** Field

---

---

**Required** Yes

---

**Value** Integer or Decimal

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

**See also** [RANGE Facet](#)

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.infix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

---

## panl.range.max.value.<lpse\_code>

*Set the replacement value if the range is the maximum value.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

**See also** [RANGE Facet](#)

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.infix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

---

---

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

---

## **panl.range.max.wildcard.<lpse\_code>**

*Set whether the range will include values greater than the maximum value if the maximum value is passed through.*

---

**Scope** Field

---

**Required** No

---

**Value** Boolean

---

**Default** false

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. If the panl.range.facet.<lpse\_code> is not set to true, then this field will be silently ignored.

**See also** [RANGE Facet](#)

---

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.infix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)

---

## **panl.range.min.<lpse\_code>**

*Set the minimum selectable value for the range.*

---

**Scope** Field

---

Required Yes

---

Value Integer or Decimal

---

Default N/A

---

Location <panl\_collection\_uri>.panl.properties

---

Notes 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

---

See also [RANGE Facet](#)[panl.range.prefix.<lpse\\_code>](#)[panl.range.infix.<lpse\\_code>](#)[panl.range.suffix.<lpse\\_code>](#)[panl.range.min.value.<lpse\\_code>](#)[panl.range.min.wildcard.<lpse\\_code>](#)[panl.range.max.<lpse\\_code>](#)[panl.range.max.value.<lpse\\_code>](#)[panl.range.max.wildcard.<lpse\\_code>](#)

---

**panl.range.min.value.<lpse\_code>**

*Set the replacement value if the range is the minimum value.*

---

**Scope** Field

---

Required Optional

---

Value String

---

Default N/A

---

Location <panl\_collection\_uri>.panl.properties

---

Notes 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

---

See also [RANGE Facet](#)[panl.range.prefix.<lpse\\_code>](#)[panl.range.infix.<lpse\\_code>](#)[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

## **panl.range.min.wildcard.<lpse\_code>**

*Set whether the range will include values less than the minimum value if the minimum value is passed through.*

---

**Scope**

---

**Required**

---

**Values**

---

**Default**

---

**Location** `<panl_collection_uri>.panl.properties`

---

**Notes** 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

---

**See also** [RANGE Facet](#)

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.infix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

## **panl.range.prefix.<lpse\_code>**

*Set the prefix for the range which will be prepended to the URI path part.*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

See also [RANGE Facet](#)

[panl.range.infix.<lpse\\_code>](#)  
[panl.range.suffix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)  
[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

## **panl.range.suffix.<lpse\_code>**

*Set the suffix for the range which will be appended to the URI path part.*

---

**Scope** Field

---

**Required** No

---

**Value** String

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. If the `panl.range.facet.<lpse_code>` is not set to `true`, then this field will be silently ignored.

See also [RANGE Facet](#)

[panl.range.prefix.<lpse\\_code>](#)  
[panl.range.infix.<lpse\\_code>](#)

[panl.range.min.<lpse\\_code>](#)

---

[panl.range.min.value.<lpse\\_code>](#)  
[panl.range.min.wildcard.<lpse\\_code>](#)

[panl.range.max.<lpse\\_code>](#)  
[panl.range.max.value.<lpse\\_code>](#)  
[panl.range.max.wildcard.<lpse\\_code>](#)

---

## panl.results.fields.<field\_set>

*Define a FieldSet for the CaFUP which will configure the fields that are returned with the results.*

---

**Scope** Collection

---

**Required** No

---

**Value** List<String> - A comma separated list of Solr field names

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** There will **ALWAYS** be a FieldSet named '**default**' which, if not defined will include all of the fields.

---

**See also**

## panl.results.testing.urls

*Whether to enable the in-built Panl Results Viewer and Panl Results Explainer webapps.*

---

**Scope** Server

---

**Required** No

---

**Value** Boolean

---

**Default** false

---

**Location** panl.properties

---

**Notes**

---

**See also**

## panl.results.sort.fields

*Configure which fields are able to be used to sort the returned documents*

---

**Scope** Collection

---

**Required** Optional

---

**Value** List<String> - a list of Solr field names

---

**Default** N/A

---

**Location** panl.properties

---

**Notes**

1. The default sorting option in Solr is relevance descending, this will override the default sort
2. Multiple, multi-level sorts are available.

---

**See also**

## panl.status.404.verbose

*Whether to return a verbose error message if an HTTP 404 (Not Found) status code occurs.*

---

**Scope** Server

---

**Required** No

---

**Value** Boolean

---

**Default** false

---

**Location** panl.properties

---

**Notes**

---

**See also** [panl.status.500.verbose](#)

## panl.status.500.verbose

*Whether to return a verbose error message if an HTTP 500 (Internal Server Error) status code occurs.*

---

**Scope** Server

---

**Required** No

---

**Value** Boolean

---

**Default** `false`

---

**Location** `panl.properties`

---

**Notes**

---

**See also** [panl.status.404.verbose](#)

---

## **panl.suffix.<lpse\_code>**

*Set the suffix for the URI path value.*

---

**Scope** Field

---

**Required** Optional

---

**Value** String

---

**Default** N/A

---

**Location** `panl.properties`

---

**Notes**

---

**See also** [panl.suffix.<lpse\\_code>](#)

---

## **panl.type.<lpse\_code>**

*This is the Solr field type that is defined in the managed schema XML file.*

---

**Scope** Field

---

**Required** Yes

---

**Value** String

---

**Default** N/A

---

**Location** `panl.properties`

---

- 
- Notes**
1. This value is taken from the Solr managed schema XML file and **SHOULD NOT** be altered unless the underlying data type in Solr has also changed.
  2. This property drives configuration options for the available Panl field.

---

**See also**

---

## panl.when.<lpse\_code>

*Only display this facet if any of the LPSE code values have already been selected*

---

**Scope** Collection

---

**Mandatory** Optional

---

**Values** List<Character> - comma separated list of LPSE codes

---

**Default** N/A

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also**

---

## solr.default.query.operand

*The default Solr query operand that acts on the search query, either - for OR, or + for AND.*

---

**Scope** Collection

---

**Required** No

---

**Value** Character

---

**Default** +

---

**Location** <panl\_collection\_uri>.panl.properties

---

- Notes**
1. **MUST** be either a '-' (for OR) or '+' (for AND) character
  2. The Panl server will refuse to start if it is any other character is set

---

**See also**

---

## **solr.facet.limit**

*The maximum number of facets that will be returned for each individual facet.*

---

**Scope** Collection

---

**Required** No

---

**Value** Integer

---

**Default** 100

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. Will output a WARN error on Panl server startup, or if the value cannot be parsed to an integer

---

**See also**

## **solr.facet.min.count**

*The minimum count value for a facet to be returned with the results.*

---

**Scope** Collection

---

**Required** No

---

**Value** Integer

---

**Default** 1

---

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes** 1. Will output a WARN error on Panl server startup, or if the value cannot be parsed to an integer

---

**See also**

The minimum number that the count of results within a facet must contain to be returned with the results. For example, if search results are returned and there exist some facets where no results would exist if the facet was chosen

It is recommended to set this value to at least 1 (one), as setting it to zero would allow the selection of the facet to return 0 (zero) results which will not filter the results at all.

## **solr.highlight**

Whether to return highlighted information with the results JSON.

---

**Scope** Collection

**Required** No

**Value** Boolean

**Default** false

**Location** <panl\_collection\_uri>.panl.properties

---

**Notes**

---

**See also**

## **solr.numrows.default**

The default number of rows to return for this collection's search. This can be changed per search request by using the panl.param.numrows LPSE code in the URI path.

---

**Scope** Collection

**Required** No

**Value** Integer

**Default** 10

**Location** <panl\_collection\_uri>.panl.properties

**Notes** 1. Will output a WARN error on Panl server startup, or if the value cannot be parsed to an int

---

**See also** [panl.param.numrows](#)

## **solr.search.server.url**

Defines the URL(s) that the SolrJ client will use to connect to the Solr server instance(s).

---

**Scope** Server

---

**Required** Yes

---

**Value** List<String> - Either a single URL, or a comma separated list of URLs

---

**Default** N/A

---

**Location** panl.properties

---

**Notes** 1. The URL may also include a `zookeeper:` prefix if a connection to a zookeeper instance is required.

**See also** [solrj.client](#)

---

## solrj.client

Sets the SolrJ client that Panl will use to connect to the Solr server.

---

**Scope** Server

---

**Required** Yes

---

**Value** String

One of:

```
Http2SolrClient
HttpJdkSolrClient
LBHttp2SolrClient
CloudSolrClient
```

---

**Default** N/A

---

**Location** panl.properties

---

**Notes** 1. If not set, the Panl server will error and refuse to start  
2. The above values are relevant to Solr version 9.x.x, other versions of the Solr server may have different properties

**See also** [solr.search.server.url](#)

---

The above property will influence the values that are appropriate for the `solr.search.server.url` property.

~ ~ ~ \* ~ ~ ~

## Design Considerations

---

If you have gotten this far in the book, then you should have a firm understanding of how the Panl server is configured and how to understand, use and implement the Panl response JSON object. This chapter dives deeper into the considerations that drove the design of the Panl server so that should you wish to peruse the source code you will have a good functioning knowledge of how (and more importantly why) it all fits together.



**Notes:** At this point in the book, you should be comfortable with the features and functionality of the Panl server

This chapter can be skipped without losing any information, this is the background around the design thinking and consequently the implementation of the codebase.

Through writing this book and describing the features and functionality, it led to more of a focus on the way in which the end user would consume a search site and how developers would integrate and implement the Panl server, rather than any particular architectural purity.

---

*"End user ease and developer implementation and integration was at the forefront of the design, at the small expense of architecture and response payload"*

---

Both the code and the response payload have some interesting design choices. For example, on the code side, in the `CollectionReponseHandler` implementation, the constructor is passed in the Panl collection URI path, which it definitely does not need to know about. However, this provides information to better explain the results in the Panl results explainer, additionally, startup messages now are collection context aware.

On the response payload side, when implementing the Panl Results Viewer a new JSON object was added to the active JSON object of `sort_fields` allowing the developer to know which sort fields are active and the sort order for the field. This information is

contained within the `sort` JSON array on the active JSON object, however, being able to use a simple lookup, rather than iterating through an array was faster and easier to implement. Yes, the response payload increases slightly, but the ease of developer integration increases a lot more.

The One-Liner for the Panl server:

Panl was designed to convert Solr faceted search URLs to human readable/SEO friendlier URLs.

*(And be as helpful as possible as it does it)*

Additionally, during the development, the implementation, startup, and running of the Panl Server, it was designed to be as helpful as possible to ensure that results are returned for any search query (even if a human hand alters the URI path).

Finally the decision was taken to build into the Panl Server some niceties which helps with the development and debugging process, which, of course can be turned off in production mode.



**Notes:** See the section on '[Panl Configuration](#)' for a complete list of Panl Server options and their implications.

## Collections And FieldSet URI Paths (CaFUPs)

*The design consideration of the collections and fieldsets is to allow having multiple views over the same returned search data.*



**Notes:** For brevity of documentation, the Collection and FieldSet URI Paths are known as CaFUPs.

A Panl collection is a many-to-one mapping to a Solr collection. Within each of the

defined Panl collection URIs, one or more subset of the returned fields can be returned, these are known as FieldSets.

Each panl collection must have at least one FieldSet, and can have as many FieldSets as desired.

This was designed so that URLs could be set up which returned just the fields that are required to be displayed on a specific page - rather than retrieving all of the fields.

For example, the if you served up wanted the following URLs

- `/products-by-brands/Caran+d'Ache/b/` which would list all of the models that the brand Caran d'Ache has, and you could then return only a subset of the fields - say the link to an image with a description of the manufacturer.
- You may also have a specific page for the individual pencil, which may include all of the fields with the total amount of data.
- Should you implement a search lookahead, you may want to have a very small subset of the FieldSets.

It was also a very deliberate design decision to hide the CaFUPs URL from the front end user. The front end caller must know of the collection and FieldSet the request corresponds to, and proxy it through to the Panl server.

## In-Built Panl Results Testing URLs

*The design consideration for this was to have a way to view and test the Panl implementation as it was being developed, however it has uses beyond this for testing and showcasing the results that are returned by the Panl server. Note: that the implementation that was used was based on very rudimentary JavaScript plugins and there are much better ways to implement the search page.*

This results viewer simple web app is available on the URI path `/panl-results-viewer/` from there the collections and FieldSets can be viewed, along with all searching/faceting functionality.

It lists all of the available CaFUPs that are available, and it provides all search, faceting, sorting, and pagination available to any Panl web app.

The results explainer simple web app is available on the `/panl-results-explainer/` URI path which will show the Panl configuration and additional explanations for the given URI path.

Both of these in-built results web apps can be disabled with the following property in the `panl.properties` file.

```
panl.results.testing.urls=false
```



**Notes:** As the footer of the results viewer claims:

This page is designed to showcase the Panl LPSE encoding mechanism, with minimal implementation. It is NOT a showcase of web technologies :)

## Panl Server Startup

*The design consideration was to ensure that during both development and production, Panl will attempt to ensure that it is set up properly, outputting helpful messages where it cannot start, or something is configured which may not be ideal.*

On startup, the `panl.properties` file and any linked `<panl_collection_uri>.panl.properties` files are parsed for correctness, logging information will be output.

**INFO** log messages are for information only, detailing how the Panl Server has been configured.

Panl does not produce a large amount of logging as it is a type of proxy service to the Solr server, however the underlying technologies that Panl utilises may produce more verbose logging.

**WARN** log messages mean that Panl has had to alter the way in which the configuration works - this should be fixed before a production deployment. A helpful message on what to change will be printed.



**IMPORTANT:** You should fix any **WARN** logging errors before rolling out any production implementation, as this means that Panl is doing something implicitly, rather than explicitly.

**ERROR** log messages will cause the Panl Server to fail to start and will exit with a (hopefully) instructive message on what needs to be fixed.



**IMPORTANT:**If Panl cannot parse, interpret, or decode the properties, it will throw a `PanlServerException` and exit.

## Error Messaging

The design consideration for handling errors that occur either in Panl or the Solr server was to provide helpful messaging to troubleshoot the problem. In the case of a 404 status, the links that are available, for a 500 status code, what went wrong in the underlying call. This may lead to information leakage in a production environment, so both of these messages can be turned down to provide the simplest of responses (which do not provide much help).

### 404 - Not Found

Any non-mapped URLs will return a HTTP status code of 404 - NOT FOUND, with a list of valid collections that the Panl Server will listen to. For the exercises in this book, it will return a JSON body response along the lines of:

```

01 {
02   "error":true,
03   "status": 404,
04   "message":"Could not find a PANL request url, see 'valid_urls' array.",
05   "valid_urls":[
06     "/mechanical-pencils/*"
07   ]
08 }
```

Which can be seen on the URL <http://localhost:8181/>.

Additionally, if you were to use the valid URLs above, the 404 error message would be

```

01 {
02   "error":true,
03   "status":404,
04   "message":"Could not find a PANL request url, see 'valid_urls' array.",
05   "valid_urls": [
06     "/mechanical-pencils/default/",
07     "/mechanical-pencils/firstfive/",
08     "/mechanical-pencils/brandandname/"
09   ]
10 }

```

Which can be seen at the URL <http://localhost:8181/mechanical-pencils/>.

This verbose messaging can be turned off with the property in the `panl.properties` file:

```
panl.status.404.verbose=false
```

If this property is set to false, the following JSON message body will be returned:

```

01 {
02   "error":true,
03   "status": 404,
04   "message": "Not found"
05 }

```

## 500 - Internal Server Error

By default the error message will include the exception message that the Solr server returned. It is long, URL encoded, but useful for debugging.

The below message states that `Invalid Number: 1aa35 for field length`, which means that Solr was not able to parse the `1aa35` as a number. Additionally Panl has not been set up to ensure that the valid values are passed through to Solr. If Panl field validation is active, Panl would have picked up the incorrect value, parsed it and forwarded the correct value through to the Solr server.

```

01 {
02   "error":true,
03   "status": 500,
04   "message":"Could not query the Solr instance, message was: Error from server at
http://localhost:8983/solr/mechanical-pencils_shard1_replica_n4/select?q=*&fl=br
and%2Cname%2Ccategory%2Cmechanism_type%2Cnib_shape&facet.mincount=1&rows=10&facet.fi
eld=lead_size_indicator&facet.field=grip_material&facet.field=colours&facet.field=nib_
shape&facet.field=diameter&facet.field=cap_shape&facet.field=brand&facet.field=mec
hanism_type&facet.field=length&facet.field=hardness_indicator&facet.field=grip_type&
facet.field=cap_material&facet.field=lead_grade_indicator&facet.field=tubing_materia
l&facet.field=in_built_sharpener&facet.field=disassemble&facet.field=category&facet.
field=body_shape&facet.field=clip_material&facet.field=mechanism_material&facet.fiel
d=lead_length&facet.field=body_material&facet.field=in_built_eraser&facet.field=grip_
shape&facet.field=relative_weight&facet.field=name&facet.field=nib_material&facet.f
ield=weight&facet.field=variants&facet=true&fq=length%3A%221aa35%22&q.op=AND&_stateV
er=mechanical-pencils%3A5&wt=javabin&version=2: Invalid Number: 1aa35 for field
length"
05 }

```

This verbose messaging can be turned off with the property in the `panl.properties` file:

```
panl.status.500.verbose=false
```

Which will then return the following message:

```

01 {
02   "error":true,
03   "status": 500,
04   "message":"Internal server error"
05 }

```

## Testing vs. Production

*The design consideration was to have the default mode for the Panl server to be in testing mode with helpful output and messages, and be able to easily turn off the testing configuration.*

The default configuration that is included in the Panl Server distribution and the properties that are generated by the Panl Server Generator all include niceties which you may wish to turn off for a production installation. The properties are as follows:

```
panl.results.viewer.url=false  
panl.status.404.verbose=false  
panl.status.500.verbose=false
```

## Quick Middleware

*The design consideration was to have the Panl server startup in the shortest amount of time, with minimal dependencies and a subset of functionality.*

Allowing a quick start up time, with only configuration through properties file allows for small changes to be made with quick, iterative testing. This also led to not implementing any authentication, or security in the Panl server.

There is a use case where you would want a user to be able to only search their subset of search results, however the complexity of passing through a unique user ID which was not open to attack or alteration as a facet to the results was not deemed to be a priority.

~ ~ ~ \* ~ ~ ~

# Epilogue

---

Firstly, my sincerest thanks for getting this far through the book. I found it quite pleasurable (at most of the times) to write, and I hope that you have had the same experience reading it.

Although my most fervent wish is that you find the Panl project to be useful.

## On The Tag-Line

---

A rather pleasing companion to the Apache® Solr® Faceted Search Engine.

---

I wrote this line as a first draft with a definite understated tone. Although, as I progressed through writing the book it resonated more with me. Firstly, I do not like to emphasise my skills or work - believing the work should speak for itself, so it fits nicely with my personal views on life. Secondly, the Panl project implementation should be understated, something which sits in between a web app and the Solr Search server and just happily does its job, hidden between the layers.

I also like the name Solr Panl, it has a certain ring to it - the original tagline at project inception in 2008 was "*Panl - soaking up the Solr goodness*".

## On Documentation

My view on documentation is that it is something that everyone loves to read, but few people like to write. And when I write 'documentation', it comes in many sources, from the actual source code and tests of a project, to the generated documentation, the StackOverflow posts, the Search Engine searches, the blog posts, videos, and books.

The thing about writing documentation of any type, is that it makes you a better engineer/developer/coder and leads back to questioning your design principles and architecture. I have never had any hard and fast rules around architecture and some of the decisions made in coding the Panl project was to make the implementation and

integration easier for the engineer when it came to parsing the JSON results and debugging/testing through the Panl viewer and explainer web apps. This led to some design decisions which coupled to code far too tightly - which was a deliberate decision. Ease of understanding over architectural purity.

Documentation can be time consuming and difficult to do - perhaps this is why documentation can be such a low priority for people, and it can be a grind to get through, constantly writing and updating text and formatting, adding in new features and having to revise the entire book to ensure that everything gets updated and referenced properly.

Just by writing this book, by having to explain how it all fits together, new ideas and ways of doing things have come to my mind, which leads to even more edits of the book.

When you have to explain a decision to someone, or how something works, you are given a second chance to review what you have created, and have to put yourself in the shoes of the reader and ask yourself the same questions about what you are doing. Questions such as:

- Why did you decide to do things this way?
- Would I be able to change the way it is done?
- How can I configure it to do this?
- What about if I want my search results page to work like this?

It also means that you are not as easily able to hide functionality that should be there but isn't, by not including something you are publicly saying that you either

- a) Didn't know about it, or
- b) Couldn't be bothered to implement it, or
- c) Hoped that people wouldn't notice.

From this perspective I have changed the underlying code and features and functionality that is present within the Panl server. Some of the changes include

- Not having a configurable `/panl-results-viewer/` and `/panl-results-explainer/` URI path, instead you may either turn on or off this functionality. This would have been a straightforward change, but the benefits were slight, new properties would have to be added, and this only occurs if there is a collection named exactly panl-results-viewer, or panl-results-explainer.
- Implementation of pass-through (or ignored) URI paths - and the additional property for keeping this token in the canonical URI generation.

- Less verbose 404 and 500 error messages
- Translation of boolean fields from true or false to something more human readable
- Added in DATE Range facets
- Highlighting, although in this instance I deliberately chose a subset of functionality to implement
- Hierarchical facets, only displaying a facet and its value if another facet has already been selected.
- Better way of implementing OR facets in the way that they work.
- Sorting of facets by (to use Solr nomenclature) index or count.

All of this has made the product a better one, and if nothing else, I thoroughly recommend writing a book, or at the very least a HOW-TO on whatever project you are working on.

## Additional Functionality in the Pipeline

The codebase for this project started in 2008, now after languishing for a long time, 16 years later, it has come a long way. Not all features and functionality made it into the code base, some from time and effort of implementation, some from documentation, and finally, something just had to be produced and put out into the wide world.

In general it came down to drawing a line under the current functionality, after all...

Code is never complete, it is just abandoned.

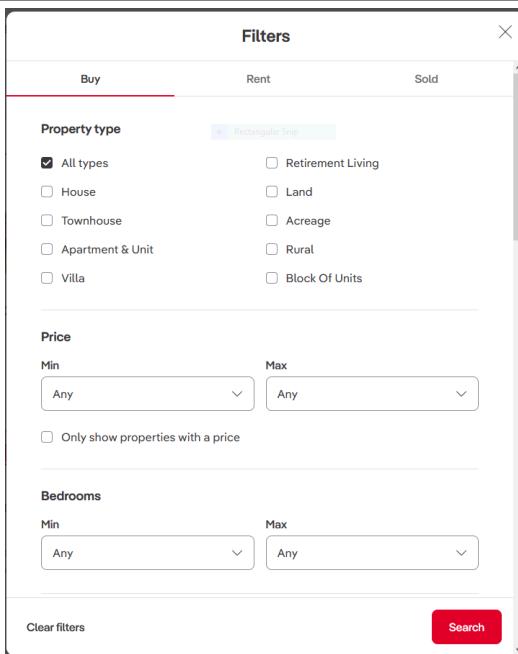
Not all of the features will make it into the next release and may be de-prioritised. The list is not in any particular order

- 1. Additional support for solr field types**

*Some will be implemented, some will probably be ignored (anything geospatial is probably not going to be included).*

- 2. Single search page**

*Being able to have a search landing page with all options available, with the ability to implement a single search page (the example below is a screenshot of the search page for <https://www.realestate.com.au>).*



*Image: A single search page showing all options - from <https://www.realestate.com.au>*

### 3. Hierarchical facets based on value

*Being able to only show facets if another facet has been selected with a specific value. This is probably not the best feature to include as it ties the configuration with the data values, which may change over time.*

### 4. Facet value replacement

*Being able to replace values for specific values of any facet, although this feature would tie the dataset and the panl configuration together more tightly than I would like.*

### 5. 'More Like This' functionality

*The ability to return 'more like this' results on a certain field, or fieldSets.*

### 6. Dynamic range functionality

*Dynamically generate the minimum and maximum value for a range for a facet value.*

### 7. Suppress range values

*For a RANGE facet, provide a configuration option to suppress the values that appear in the range as separate values.*

### 8. Returning more facets for a specific facet field

*By default, the facet limit is set to be 100 facet values per facet, in the instance where the returned number of facets are greater than this value, the facets will be truncated. The Panl server should be able to return the remaining facets with a simple query, without*

*returning any documents with it. This should be done on an individual facet and possibly have pagination.*

## 9. Default empty fieldSet

*In addition to the 'default' FieldSet, add another FieldSet always named 'empty' which will return no fields (this links in with the 'Returning more facets for a specific facet field' and 'Single search page' items).*

## 10. Internationalisation

*Floating point digits in particular suffer from using a full stop/period as a decimal place i.e. in the UK, a digit would be formatted e.g. 12,345,678.90, whilst most other European countries use a comma for the decimal place e.g. 12.345.678,90.*

## 11. DATE Range facet update

*As an extension to internationalisation, the DATE Range facet could do with an update to ensure that the SEO URL is better suited to international uses. For example, this facet will respond to <range\_identifier><value><range\_type> - e.g. 'previous 30 days' however for other languages this is not the most suitable - e.g. in French it might be '30 jours précédents' as in <value><range\_type><range\_identifier>.<sup>19</sup>*

## 12. Panl configuration editor

*A GUI to edit (and validate) the panl configuration files making it easier for a developer to get the configuration correct.*

~ ~ ~ \* ~ ~ ~

*In any case, I hope that you enjoy - and more importantly, implement and use - the Panl server for your Solr search server implementation.*

~ ~ ~ \* ~ ~ ~

---

<sup>19</sup> German users should be fine with the default implementation from Panl.

# Appendices

---

## Definitions

### #

**42:** *The meaning of life*

**404:** *An HTTP status code returned when a resource could not be located by the server.*

**500:** *An HTTP status code returned by the server when an error occurred that stopped the processing of a request.*

### A

**Apache:** *Refers to the Apache Software Foundation, the producers of an HTTP Server and a fine collection of software and libraries, also the author of the Solr search server.*

### B

**Boolean facet:** *A Solr field which can only be one of two values, 'true' or 'false'.*

**Boolean value replacement:** *For a 'true' or 'false' value from a Solr field can have their 'true' or 'false' value replaced with more meaningful text, which will be converted by the Panl server.*

### C

**CaFUPs:** *An acronym for Collection and FieldSet URI Paths. This is the URI path that the Panl server will handle to return results from the Solr collection with the configured field names.*

**Collection:** *A collection is either a single logical search index that uses a single Solr configuration file (`solrconfig.xml`), or a collection of properties and field sets for the Panl server. (See also: Solr collection, and Panl collection).*

### D

**DATE range facet:** *A Solr facet of type `solr.DatePointField` that can be used to filter results from the time period NOW to some arbitrary number of hours, days, months, or years, or from some arbitrary number of hours, days, months, years to NOW.*

**Document:** *A single result returned from the Solr search server.*

### E

### F

**FieldSet(s):** *A list of Solr field names that will be returned by the Panl server in the result documents.*

**Facet:** *Also referred to as a regular facet, a defined field in the Solr and Panl configuration that allows a user to filter the results by selecting a particular value for this field that the document must have. The returned facets have a Solr field name, a Panl field name, a value, and a count. (See also: Regular facets, OR facets, RANGE facets, DATE range facets, and Boolean facets)*

**Facet count:** *The number of documents which have a particular facet value assigned to them.*

**Facet index:** *Solr nomenclature for the value of a facet.*

**Facet value:** *The value of the Solr facet field as indexed by the Solr search server.*

## G

**Hierarchical facet:** *A facet that will only appear if another facet or parameter has already been selected. This is defined by the panl.when.<lpse\_code> property.*

## H

**Hierarchical facet:** *A facet that will only appear if another facet or parameter has already been selected. This is defined by the panl.when.<lpse\_code> property.*

**Highlighting:** *When configured, highlighting will return the matching text surround be markup (e.g. HTML) so that it may be rendered in a different fashion to bring attention to the text.*

## I

**Infix:** *Text that is placed between the range values. Note: this is only available for RANGE facets.*

## IJ

## K

## L

**LPSE code:** *The Last Path Segment Encoded string that the Panl server uses to parse the preceding URI path.*

## M

**Mechanical Pencil:** *(or a clutch pencil) is a pencil with a replaceable and mechanically extendable graphite core (or "lead"). The lead is not attached to the outer casing, and can be mechanically extended as its point is worn away from use.*

## N

**Number per page parameter:** *Defines the number of results to return per page from the Solr server. This may be set to any positive integer value greater than 0 (zero).*

## O

**OR facet:** *A configured facet that will allow returning multiple values from a facet that normally would allow the user to select only one.*

## P

**Page parameter:** *A parameter which defines the page number of results that the end-user will be shown.*

**Panl collection:** *A collection of properties and field sets that control how the facets are configured and how the search results are returned.*

**Panl collection URI:** *Unlike a Solr collection of documents, a Panl collection is a collection of field sets and configuration. There can be multiple field sets per configuration file, and multiple Collection URIs can be connected to a Solr search collection. The Panl collection URI is made up of the selected Panl collection and selected FieldSet.*

**Panl field:** *A one-to-one mapping between a Solr field to a Panl field which is referenced throughout the configuration.*

**Panl generator:** *A utility function within the Panl server that will generate a starting panl.properties file and a panl\_collection\_uri.panl.properties file from a managed-schema.xml Solr collection configuration file.*

**Panl server:** *The HTTP server that acts as the proxy between LPSE encoded URI parts and the underlying Solr Server*

**Panl release package:** *A binary release as a .zip or .tgz file that can be downloaded from the Github releases page for Synapticloop panl project.*

**Parameters:** *Parameters are specialised forms of LPSE codes that do not directly filter the results, but alter the way in which the results are returned.*

**Prefix:** *Text that is prepended to a facet value for the URI path, and optionally for the display of the value.*

## Q

**q:** *The query operator URL parameter that the Solr server responds to when searching text within the collection.*

**q.op:** *The query operand URL parameter that the Solr server responds to to determine whether the search term should be found in one of the fields (OR), or all of the fields (AND).*

**Query term:** *Text that is either a word or a phrase that is used to search across the Solr collection index. Unlike facets, this is not limited to the values of the Solr facets.*

**Query parameter:** *The parameter that is passed through as a URL parameter from a text input field by an HTML form.*

# R

**RANGE facet:** *A specialised form of facet which allows a search to be performed on a range of values. If the Solr document value matches between the minimum and maximum range values, then it will be included with the results.*

**Replacement value:** *A value that replaces another value, used in BOOLEAN facets to replace a "true" or "false" value with another more human-readable value, and in RANGE facets for minimum and maximum values.*

**Regular facet:** *The default type of facet that allows filtering on a particular value.*

# S

**SEO:** *An acronym for Search Engine Optimisation which is a set of practices designed to improve the appearance, positioning, and usefulness of content in the search results for search engines.*

**Solr collection:** *A collection of documents that Solr has indexed and is searched upon to return results. (See also: Panl collection)*

**Solr field:** *The name of a field that is defined within the managed-schema.xml file, which may be indexed, stored, and/or analysed.*

**Solr server:** *The Solr server*

**Solr schema:** *The XML file that defines the fields, their types, what*

**Sort order:** *The field, or fields of a document which will be either ascending or descending ordered.*

**Suffix:** *Text that is appended to a facet value for the URI path, and optionally for the display of the value. This also applies to a RANGE facet when appending the text to the end of a range query.*

**Synapticloop:** *The people behind the Panl server and generator.*

# T

**.tar:** *A bundle of files placed together into the Tape ARchive format*

**Token:** *Panl parses and decodes the incoming request URI path and turns each of the path parts into a token. If there was an error in either the parsing or decoding, then this token will be marked as invalid and will not be passed through to the Solr server.*

# U

**URI:** *Uniform Resource Identifier which is a unique sequence of characters that identifies an abstract or physical resource.*

**URI path part:** *A part of the path of a URI, in the examples in the book the path part is taken as everything after the hostname. A URI is of the form:*

*scheme ":" ["/" authority] path ["?" query] ["#" fragment]*

**URL:** *Uniform Resource Locator is a subset of URI which specifically references a webpage.*

## | V

## | W

**Wildcard:** *Designates that a value that is used will match all values - used by the Panl server within a range query to indicate that it should be any number either below or above the selected minimum/maximum value.*

## | X

**x:** *x marks the spot.*

## | Y

**y:** *Why not?*

## | Z

**Zip:** *A file compression format allowing multiple files and directories to be easily packaged into a single file.*

## Command Line Options

The Panl release package has two modes

1. Panl server - serving up the search results, and
2. Panl generator - to generate configuration files for an existing collection.

For both of the modes, the usage describes the invocation through the Java JRE, rather than the supplied executable files (i.e. bin/panl and bin\panl.bat), however the command line options are the same.

### Panl Server

The Panl server may be started by

```
bin\panl server
```

Which will start the server with default values:

- Looking for a `panl.properties` file in the directory that the command was executed
- Binding to the default port of `8181`.

To set the `panl.properties` file that will be referenced, use the `-properties` command line option.

To set the port that you want to bind the properties to, use the `-port` command line option.

### Panl Generator

The Panl generator can be invoked by

```
bin\panl generate -schema path/to/solr/managed-schema.xml
```

However you **MUST** at a minimum pass through the path to the Solr managed schema file with the `-schema` command line option.

The default output directory and file name is the directory where this command was invoked and the filename will be `panl.properties`. Should you wish to change this, use the `-properties` command line option with the filename. If the option points to a directory, the default filename of `panl.properties` will be used. **Note:** This will also be

the directory that the `<panl_collection_uri>.panl.properties` will be written to as well.

Panl generator will fail if the files already exist, however you may pass through the command line switch of `-overwrite` with a value of `true` to overwrite the files (the default is `false`).

## Usage Text

You can see the complete usage (and possibly updated) instructions on the GitHub repository:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/usage.txt>

```

01
02      .-----.
03      | _   | _   | _   |
04      | _ _|_ _|_ _|_ _|
05      |__|    ... .-..
06
07      ~ ~ ~ * ~ ~ ~
08
09 Usage:
10   java -jar panl.jar \
11     [command] \
12     [-properties properties_file_location] \
13     [-overwrite true_or_false] \
14     [-config solr_config_location] \
15     [-port port_number]
16
17 Where:
18   [command] is one of:
19     server - start the server, or
20     generate - generate an example panl.properties file from an existing Solr
21           managed-schema.xml file
22
23   To start the panl server
24

```

```

25  java -jar panl.jar \
26    server \
27    [-properties properties_file_location] \
28    [-port port_number]
29
30 To generate the panl.properties file for each collection:
31
32  java -jar panl.jar \
33    generate \
34    [-properties properties_file_location] \
35    [-overwrite true_or_false] \
36    -schema solr_schema_location
37
38 If you choose the 'server' command, the following command line options are
39 available:
40
41 [-properties properties_file_location] (optional - default panl.properties)
42   the properties file to load, if this property is not included, the default
43   panl.properties file will be used which __MUST__ reside in the same
44   directory as the server start command
45
46 [-port port_number] (optional) the port number to start the server on.  The
47   default port number is 8181
48
49 If you choose the 'generate' command, the following command line options are
50 available:
51
52 [-properties properties_file_location] (optional - default panl.properties)
53   the base properties file to write the generated configuration out to. If
54   this property is not included, the default panl.properties filename will
55   be used with each collection file that is generated named:
56     <collection_uri>.panl.properties
57   NOTE: If the files exist, the generation will TERMINATE, you will need to
58   remove those files before the generation - unless you have the
59   -overwrite true command line option present
60
61 [-overwrite true_or_false] (optional - default false)

```

```

62 -schema solr_schema_location(s) (mandatory) the managed-schema.xml
63   configuration file(s) to read and generate the panl.properties file from.
64   NOTE: For multiple files, use comma separated values
65
66

```

## Sample .properties Files

### The Sample panl.properties Files

There are multiple sample properties files that are included in the downloaded packaged and can be viewed online:

#### Mechanical pencils

The default collection for running the examples through the book.

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/panl.properties>

#### All

This file will start the Panl server with all included examples - however for it to display results correctly, the all datasets must be indexed.

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/all/panl.properties>

#### Book store

Just the book store demo dataset with hierarchical and facet ordering.

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/book-store/panl.properties>

#### Mechanical pencils OR

The mechanical pencils collection with the manufacturer as an OR facet

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils-or/panl.properties>

## Simple date

The simple date collection with DATE range facets.

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/simple-date/panl.properties>

The contents of these files are unlikely to change between publishing this book and updates to the codebase. To see the most up-to-date comments and instructions for usage, the template file that the Panl generator utility uses as a base can be found here:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/panl.properties.template>

## The Sample <panl\_collection\_uri>.panl.properties Files

The sample file for the mechanical pencils example used in this book can be viewed online:

### Mechanical pencils

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils/mechanical-pencils.panl.properties>

Additional properties files are also included in the download package

### Book store (with Hierarchical facets and facet ordering)

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/book-store/book-store.panl.properties>

### Mechanical pencils OR faceting

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/mechanical-pencils-or/mechanical-pencils-or.panl.properties>

### Simple date (with DATE range facets)

<https://github.com/synapticloop/panl/blob/main/src/dist/sample/panl/simple-date/simple-date.panl.properties>

The contents of this file is unlikely to change between publishing this book and updates to the codebase. To see the most up-to-date comments and instructions for usage, the template file that the Panl generator utility uses as a base can be found here:

[https://github.com/synapticloop/panl/blob/main/src/main/resources/panl\\_collection\\_urі.panl.properties.template](https://github.com/synapticloop/panl/blob/main/src/main/resources/panl_collection_urি.panl.properties.template)

## Solr Version 8 & 7 Integration Notes

The SolrJ connectors are changed, with the available list of SolrJ clients for version 8:

- HttpSolrClient
- Http2SolrClient
- LBHttpSolrClient
- LBHttp2SolrClient
- CloudSolrClient

And for version 7

- HttpSolrClient
- LBHttpSolrClient
- CloudSolrClient

The JSON response object has also changed

In Solr version 7 and 8:

```

01 {
02   "facet_counts": { ... },
03   "response": [ ... ],
04   "responseHeader": { ... }
05 }
```

**Note:** The `response` key is a **JSON array** of results (line 3)

Whilst in Solr 9:

```

01 {
02   "facet_counts": { ... },
03   "response": {
04     "docs": [ ... ]
05     "numFound": 55,
```

```

06     "start": 10,
07     "maxScore": 1,
08     "numFoundExact": true
09 },
10   "responseHeader": { ... }
11 }
```

**Note:** The `response` key is a JSON Object of results (line 3 to 9 in **bold** above) with additional details.

## Setting up a Solr 7 or 8 server

The main difference between Solr 7, 8 and Solr 9 is that the configuration **MUST** be uploaded to the zookeeper instance first, before the collection is created.

Additionally, there is no `bin\post` command for Windows machines so this is done through a Java command.

## Windows Commands



**IMPORTANT:** Each of the commands - either Windows or \*NIX must be run on a single line - watch out for ↲ continuations.

### Step 1. Create an example cloud instance

This requires no interaction, will use the default setup, two replicas, and two shards under the 'example' cloud node.

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY bin\solr start -e cloud -noprompt</pre>

### Step 2. Create the configuration for the mechanical pencils

This will create and set up the mechanical pencil schema so that a collection can be created and the data can be indexed.

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY
bin\solr zk upconfig -d ←
PANL_INSTALL_DIRECTORY\sample\solr\mechanical-pencils -n mechanical-pencils ←
-z localhost:9983
```

**Step 3. Create the mechanical pencils collection**

This will create and set up the mechanical pencil collection and schema so that the data can be indexed.

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY
bin\solr create -c mechanical-pencils -n mechanical-pencils ←
-s 2 -rf 2
```

**Step 4. Index the mechanical pencils data**

This will index the included sample mechanical pencil data into the Solr instance in the `mechanical-pencils` collection.

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

java -Dc=mechanical-pencils -Dtype=application/json -jar example\exampledocs\post.jar ←
PANL_INSTALL_DIRECTORY\sample\data\mechanical-pencils.json
```

**Step 5. Start the Panl Server**

This will start the server and be ready to accept requests.

**Command(s)**

```
cd PANL_INSTALL_DIRECTORY

bin\panl.bat -properties ←
PANL_INSTALL_DIRECTORY\sample\panl\mechanical-properties\panl.properties
```

## Step 6. Start searching and faceting

Open the link <http://localhost:8181/paml-results-viewer/> in your favourite browser, choose a collection/fieldset and search, facet, sort, paginate and view the results

For the simple-date dataset, the commands are almost identical, and, assuming that the Solr cloud is set up:

Command(s)
<pre>cd SOLR_INSTALL_DIRECTORY bin\solr zk upconfig -d ^ PANL_INSTALL_DIRECTORY\sample\solr\simple-date -n simple-date ^ -z localhost:9983  bin\solr create -c simple-date -n simple-date -s 2 -rf 2  cd SOLR_INSTALL_DIRECTORY  java -Dc=simple-date -Dtype=application/json -jar example\exampledocs\post.jar ^ PANL_INSTALL_DIRECTORY\sample\data\simple-date.json</pre>

## \*NIX Commands

The \*NIX commands are as per the windows section above with the file path delimiter changed from '\' to '/'



**IMPORTANT:** Each of the commands - either Windows or \*NIX must be run on a single line - watch out for '^' continuations.

## Step 1. Create an example cloud instance

This requires no interaction, will use the default setup, two replicas, and two shards under the 'example' cloud node.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr start -e cloud -noprompt

### Step 2. Create the configuration for the mechanical pencils

This will create and set up the mechanical pencil schema so that a collection can be created and the data can be indexed.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr zk upconfig -d ↵
PANL_INSTALL_DIRECTORY/sample/solr/mechanical-pencils -n mechanical-pencils ↵
-z localhost:9983

### Step 3. Create the mechanical pencils collection

This will create and set up the mechanical pencil collection and schema so that the data can be indexed.

Command(s)
cd SOLR_INSTALL_DIRECTORY
bin/solr create -c mechanical-pencils -n mechanical-pencils ↵
-s 2 -rf 2

### Step 4. Index the mechanical pencils data

This will index the included sample mechanical pencil data into the Solr instance in the `mechanical-pencils` collection.

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

java -Dc=mechanical-pencils -Dtype=application/json -jar example/exampledocs/post.jar <
PANL_INSTALL_DIRECTORY/sample/data/mechanical-pencils.json
```

**Step 5. Start the Panl Server**

This will start the server and be ready to accept requests.

**Command(s)**

```
cd PANL_INSTALL_DIRECTORY

bin/panl -properties <
PANL_INSTALL_DIRECTORY/sample/panl/mechanical-properties/panl.properties
```

**Step 6. Start searching and faceting**

Open the link <http://localhost:8181/panl-results-viewer/> in your favourite browser, choose a collection/fieldset and search, facet, sort, paginate and view the results

For the simple-date dataset, the commands are almost identical, and, assuming that the Solr cloud is set up:

**Command(s)**

```
cd SOLR_INSTALL_DIRECTORY

bin/solr zk upconfig -d PANL_INSTALL_DIRECTORY/sample/solr/simple-date -n simple-date <
-z localhost:9983

bin\solr create -c simple-date -n simple-date -s 2 -rf 2

java -Dc=simple-date -Dtype=application/json -jar example/exampledocs/post.jar <
PANL_INSTALL_DIRECTORY/sample/data/simple-date.json
```

## Additional Solr Version 7 Integration Notes

In the Panl response object, the `num_results_exact` (line 3 in **bold** below) will **ALWAYS** be `true`, as this version of Solr does not have this data available.

```

01 {
02   "num_pages": 6,
03   "num_results_exact": true,
04   "page_uris": {
05     "next": "/page-2/p/",
06     "before": "/page-",
07     "after": "/p/"
08   },
09   "page_num": 1,
10   "num_results": 55,
11   "num_per_page": 10,
12   "num_per_page_uris": {
13     "before": "/",
14     "after": "-per-page/n/"
15   }
16 }
```

## Solr Versions 6 and below

Unfortunately version 6 and earlier have no release packages to be built, the only way to have these built is to edit and compile the code from source.

<https://github.com/synapticloop/panl/>

Starting with the branch `solr-panl-7` is probably the wisest - these are the integration points that will need to be looked at.

1. Create a new branch with your version of Panl - e.g. Version 6 would be branched as `solr-panl-6`
2. Edit the `/build.gradle` file

- change the `distributionBaseName` - e.g. `solr-panl-6`

```
distributions {
    main {
        distributionBaseName = 'solr-panl-6'
    }
}
```

- Edit the Solrj dependencies, the dependency for SolrJ can be found on any maven repository:

<https://mvnrepository.com/artifact/org.apache.solr/solr-solrj>

```
dependencies {
    ...
    // solrj
    implementation 'org.apache.solr:solr-solrj:?.?.?'
    ...
}
```

- Edit the Solr schema configuration and managed-schema
  - Copy over the version schema from the Solr version that you are using. (this can be found in the `SOLR_INSTALL_DIRECTORY/example/files/conf` directory - or equivalent for your Solr version).
  - Edit the file to include the relevant fields that you want to index and search on
- Look at the `com.synapticloop.panl.server.client` package, adding in the Solr Clients that are applicable to your version
- Update the `com.synapticloop.panl.server.client.PanlClient` class to reference the correct clients.
- Update the  
`com.synapticloop.panl.server.handler.helper.CollectionHelper#getPanlClient()` factory method to return the correct client



**IMPORTANT:** The returned JSON object may have changed between versions which may cause problems with generation and returning the Panl response.

There may be other integration points for version 6.x.x and lower, however the instructions above where the changed integration points from version 9 to version 8 and 7.

~ ~ ~ \* ~ ~ ~