



Getting Started With Panl

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

Author: Synapticloop - Version 1.2.0

This book will rapidly 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 Walkthrough Example - The Book Store](#) section to gain a deeper understanding of the configuration options.
4. Use your existing dataset with [Integrating An Existing Solr Schema](#)
5. Need help understanding the options? 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/>, or to download the latest copy of this book, see <https://github.com/synapticloop/panl/releases/>.

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.

IMPORTANT: 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

| | |
|--|-----------|
| README.1ST..... | 14 |
| Welcome To Synapticloop Panl..... | 14 |
| Additional Panl Niceties..... | 14 |
| About This Book..... | 16 |
| Nomenclature Used Throughout This Book..... | 16 |
| Book Format Conventions..... | 20 |
| About Panl Server..... | 22 |
| How Many Facets Does Panl Support?..... | 23 |
| Panl URL Structure..... | 24 |
| In-Built Web Apps (Viewer / Explainer / Single Search Page)..... | 26 |
| About Panl Generator..... | 33 |
| About Apache Solr..... | 33 |
| Why Synapticloop Panl?..... | 35 |
| What is a LPSE (pronounced 'lapse') code?..... | 36 |
| The three types of LPSE codes..... | 37 |
| URL Path Nomenclature..... | 37 |
| Is Synapticloop Panl For Me?..... | 38 |
| What You Will Need..... | 40 |
| Download The Resources..... | 40 |
| The Panl Directory Structure..... | 40 |
| Panl Server Versions..... | 43 |
| Solr-Panl-9-x.x.x..... | 43 |
| Solr-Panl-8-x.x.x & Solr-Panl-7-x.x.x..... | 43 |
| Solr-Panl-6-x.x.x and earlier version..... | 44 |
| Quick Start - The 5 Steps..... | 45 |
| A Note On Running The Commands..... | 46 |
| Windows Commands..... | 47 |
| *NIX Commands..... | 48 |
| Next Steps..... | 50 |
| Getting Started..... | 51 |
| Downloading the Resources..... | 51 |
| The Solr server..... | 52 |
| The Panl server..... | 52 |
| Creating and Starting a Solr Cloud Instance..... | 52 |

| | |
|--|-----------|
| Creating a collection..... | 55 |
| Indexing the Data..... | 56 |
| Starting The Panl Server..... | 57 |
| When Something Goes Wrong. | |
| Checking the Logs..... | 59 |
| Deleting a collection..... | 59 |
| You will then need to re-create the collection, and re-index the data..... | 60 |
| Restarting the Solr Panl tutorial process..... | 60 |
| Stopping Solr..... | 60 |
| Re-Starting Solr..... | 61 |
| Worst-case scenario..... | 61 |
| Next Steps..... | 62 |
| Additional Data..... | 63 |
| All Data Panl Server..... | 63 |
| Simple Date..... | 64 |
| Creating and Indexing the Data..... | 64 |
| panl.properties File Additions..... | 65 |
| Mechanical Pencils OR Facet..... | 65 |
| Creating and Indexing the Data..... | 65 |
| panl.properties File Additions..... | 65 |
| Mechanical Pencils More Facets..... | 66 |
| Creating and Indexing the Data..... | 66 |
| panl.properties File Additions..... | 66 |
| Book Store..... | 67 |
| A Walkthrough Example - The Book Store..... | 68 |
| 0. High Level Requirements..... | 69 |
| 1. Understanding the Dataset..... | 70 |
| 2. Configure the Solr Index..... | 72 |
| 3. Configure the Panl Server..... | 77 |
| The Default Search Page Configuration..... | 78 |
| The Generated panl.properties File..... | 80 |
| The Generated <panl_collection_url>.panl.properties File..... | 80 |
| 4. Determine the Web Pages to Render..... | 96 |
| Author and Author Series..... | 96 |
| Author Listing..... | 96 |
| The Iterative Implementation Process..... | 97 |
| Working With Any Dataset..... | 98 |
| Supported Solr Data Types..... | 98 |
| Unsupported/Partially Supported Solr Field Types..... | 99 |

| | |
|--|------------|
| Facet and Field Types..... | 100 |
| Fields..... | 100 |
| Regular Facets..... | 101 |
| BOOLEAN Facets..... | 102 |
| RANGE Facets..... | 103 |
| DATE Range Facets..... | 104 |
| OR Facets..... | 105 |
| Other Facet Options..... | 106 |
| A Brief Introduction To Solr (And Faceted Search)..... | 108 |
| Querying Data..... | 108 |
| The Solr Managed Schema..... | 110 |
| Determining the Appropriate FieldType and Attributes..... | 115 |
| The Solr Configuration File..... | 119 |
| Query Request Handler..... | 120 |
| Highlighting..... | 121 |
| Panl Configuration..... | 123 |
| The panl.properties file..... | 123 |
| The <panl_collection_url>.panl.properties file..... | 124 |
| The panl.properties Configuration File..... | 126 |
| Configuring The SolrJ Connector..... | 127 |
| Setting The Solr Server URL(s)..... | 128 |
| Enabling The Panl Results Testing URLs..... | 129 |
| The Panl Results Viewer..... | 130 |
| The Panl Results Explainer..... | 131 |
| Select one of the available CaFUPs links, which will then present the configuration for that particular CaFUP..... | 132 |
| The Panl Single Page Search..... | 132 |
| Setting HTTP Status Message Verbosity..... | 133 |
| Binding Solr collections to Panl URL paths..... | 134 |
| The <panl_collection_url>.panl.properties Configuration File..... | 136 |
| Parameter and Operand Definitions..... | 137 |
| panl.param.query..... | 137 |
| panl.param.sort..... | 137 |
| panl.param.page..... | 138 |
| panl.param.numrows..... | 139 |
| panl.param.query.operand..... | 140 |
| panl.param.passthrough..... | 140 |
| panl.param.passthrough.canonical..... | 141 |
| panl.form.query.respondto..... | 143 |

| | |
|---|-----|
| panl.include.single.facets..... | 143 |
| panl.include.same.number.facets..... | 143 |
| solr.default.query.operand..... | 145 |
| solr.facet.limit..... | 145 |
| solr.facet.min.count..... | 146 |
| solr.numrows.default..... | 146 |
| solr.highlight..... | 146 |
| Field Definitions..... | 146 |
| Properties Available for Fields..... | 147 |
| Facet Definitions..... | 148 |
| Properties Available for Facets..... | 149 |
| Regular Facet..... | 151 |
| Example..... | 152 |
| Decoding the URL..... | 153 |
| BOOLEAN Facet..... | 153 |
| Additional Properties..... | 154 |
| Example..... | 155 |
| Decoding the URL Without a Prefix or Suffix..... | 155 |
| Defining a Prefix and Suffix..... | 156 |
| Decoding the URL With a Prefix or Suffix..... | 157 |
| OR Facet..... | 157 |
| Additional Properties..... | 157 |
| The Difference Between Multi-valued Facets and OR Facets..... | 158 |
| Example..... | 161 |
| Decoding the URL..... | 163 |
| RANGE Facet..... | 164 |
| Additional Properties..... | 165 |
| Suppressing Individual Facet Values..... | 167 |
| Visualising the Properties..... | 168 |
| Decoding the URL - Range facet with an infix..... | 170 |
| Decoding the URL - Range facet without an infix..... | 170 |
| DATE Range Facet..... | 170 |
| Additional Properties..... | 174 |
| Example..... | 174 |
| Decoding the URL..... | 175 |
| Some Notes..... | 176 |
| About Value Replacements..... | 176 |
| On Field Validations..... | 178 |
| About Hierarchical Facets..... | 178 |

| | |
|--|------------|
| About Sorting Facets..... | 179 |
| Putting It All Together (Technically)..... | 181 |
| A Brief Architecture..... | 181 |
| Production Readiness..... | 182 |
| Logging..... | 182 |
| Setting the Logging Levels..... | 184 |
| Generating SEO Friendly URLs..... | 184 |
| Data Dependencies..... | 184 |
| Passthrough URLs..... | 184 |
| Handling Errors..... | 185 |
| Panl LPSE URL Paths Explained..... | 186 |
| Parameter Query..... | 187 |
| Parameter Query Operand..... | 188 |
| Parameter Number of Rows..... | 188 |
| Parameter Page Number..... | 190 |
| Parameter Pass Through..... | 191 |
| Parameter Sort Order..... | 192 |
| Facets..... | 193 |
| Explaining A More Complex Example..... | 195 |
| Search Integration And The Panl Response Object..... | 197 |
| URL Bindings..... | 197 |
| Error Responses..... | 197 |
| Success Responses..... | 198 |
| The "active" JSON Object..... | 199 |
| Overall Integration and Implementation..... | 201 |
| The "active.facet" JSON Array (Regular facet object)..... | 202 |
| Integration and Implementation..... | 203 |
| The "active.facet" JSON Array (BOOLEAN facet object)..... | 203 |
| Integration and Implementation..... | 204 |
| The "active.facet" JSON Array (DATE Range facet object)..... | 205 |
| Integration and Implementation..... | 206 |
| The "active.facet" JSON Array (OR facet object)..... | 207 |
| Integration and Implementation..... | 208 |
| The "active.facet" JSON Array (RANGE facet object)..... | 208 |
| Integration and Implementation..... | 209 |
| The "active.numrows" JSON Object..... | 210 |
| Integration and Implementation..... | 210 |
| The "active.page" JSON Object..... | 210 |
| Integration and Implementation..... | 211 |

| | |
|---|------------|
| The "active.query" JSON Object..... | 211 |
| Integration and Implementation..... | 212 |
| The "active.query" JSON Object..... | 212 |
| Integration and Implementation..... | 213 |
| The "active.sort" JSON Array..... | 213 |
| Integration and Implementation..... | 213 |
| The "available" JSON Object..... | 214 |
| The "available.date_range_facets" JSON Array..... | 215 |
| Integration and Implementation..... | 216 |
| The "available.facets" JSON Array..... | 217 |
| Integration and Implementation..... | 218 |
| The "available.range_facets" JSON Object..... | 219 |
| Integration and Implementation..... | 219 |
| The "pagination" JSON Object..... | 226 |
| Integration and Implementation..... | 226 |
| The "sorting" JSON Object..... | 227 |
| Integration and Implementation..... | 229 |
| Single Search Page Integration And The Panl Response Object..... | 233 |
| URL Bindings..... | 233 |
| Error Responses..... | 234 |
| Success Responses..... | 234 |
| The "lpse_lookup" JSON Object..... | 235 |
| The "lpse_order" JSON Array..... | 236 |
| Generating the LPSE URL path..... | 236 |
| More Facets Integration And The Panl Response Object..... | 237 |
| URL bindings..... | 237 |
| Building the Request URL..... | 237 |
| Determining The Facet Limit..... | 238 |
| Error Responses..... | 240 |
| Success Responses..... | 240 |
| The "facet" JSON Object..... | 241 |
| Implementation notes..... | 241 |
| Integrating An Existing Solr Schema..... | 243 |
| Using the Panl Generator..... | 243 |
| Running the Command Line Utility..... | 243 |
| Editing the Generated Files..... | 247 |
| Manually Creating the Configuration Files..... | 247 |
| Properties Quick Reference..... | 249 |
| Property References Format Explained..... | 249 |

| | |
|---|-----|
| panl.bool.<lpse_code>.false..... | 250 |
| panl.bool.<lpse_code>.true..... | 251 |
| panl.collection.<solr_collection_name>..... | 252 |
| panl.date.<lpse_code>.days..... | 253 |
| panl.date.<lpse_code>.hours..... | 254 |
| panl.date.<lpse_code>.months..... | 255 |
| panl.date.<lpse_code>.next..... | 256 |
| panl.date.<lpse_code>.previous..... | 257 |
| panl.date.<lpse_code>.years..... | 259 |
| panl.decimal.point..... | 260 |
| panl.facet.<lpse_code>..... | 260 |
| panl.facetsort.<lpse_code>..... | 261 |
| panl.field.<lpse_code>..... | 262 |
| panl.form.query.respondto..... | 263 |
| panl.include.same.number.facets..... | 263 |
| panl.include.single.facets..... | 264 |
| panl.lpse.ignore..... | 264 |
| panl.lpse.length..... | 265 |
| panl.lpse.order..... | 265 |
| panl.mutlivalue.<lpse_code>..... | 266 |
| panl.name.<lpse_code>..... | 266 |
| panl.or.facet.<lpse_code>..... | 267 |
| panl.or.always.<lpse_code>..... | 268 |
| panl.param.numrows..... | 268 |
| panl.param.numrows.prefix..... | 269 |
| panl.param.numrows.suffix..... | 269 |
| panl.param.page..... | 270 |
| panl.param.page.prefix..... | 270 |
| panl.param.page.suffix..... | 271 |
| panl.param.passthrough..... | 271 |
| panl.param.passthrough.canonical..... | 272 |
| panl.param.query..... | 272 |
| panl.param.query.operand..... | 273 |
| panl.param.sort..... | 273 |
| panl.prefix.<lpse_code>..... | 274 |
| panl.range.facet.<lpse_code>..... | 274 |
| panl.range.infix.<lpse_code>..... | 275 |
| panl.range.max.<lpse_code>..... | 276 |
| panl.range.max.value.<lpse_code>..... | 276 |

| | |
|--|------------|
| panl.range.max.wildcard.<lpse_code>..... | 277 |
| panl.range.min.<lpse_code>..... | 278 |
| panl.range.min.value.<lpse_code>..... | 278 |
| panl.range.min.wildcard.<lpse_code>..... | 279 |
| panl.range.prefix.<lpse_code>..... | 280 |
| panl.range.suffix.<lpse_code>..... | 281 |
| panl.range.suppress.<lpse_code>..... | 281 |
| panl.results.fields.<field_set>..... | 282 |
| panl.results.fields.default..... | 282 |
| panl.results.fields.empty..... | 283 |
| panl.results.testing.urls..... | 283 |
| panl.results.sort.fields..... | 284 |
| panl.status.404.verbose..... | 284 |
| panl.status.500.verbose..... | 285 |
| panl.suffix.<lpse_code>..... | 285 |
| panl.type.<lpse_code>..... | 286 |
| panl.when.<lpse_code>..... | 286 |
| solr.default.query.operand..... | 287 |
| solr.facet.limit..... | 287 |
| solr.facet.min.count..... | 288 |
| solr.highlight..... | 288 |
| solr.numrows.default..... | 289 |
| solr.search.server.url..... | 289 |
| solrj.client..... | 289 |
| Design Considerations..... | 291 |
| Collections And FieldSet URL Paths (CaFUPs)..... | 292 |
| In-Built Panl Results Testing URLs..... | 293 |
| Panl Server Startup..... | 294 |
| Error Messaging..... | 295 |
| 404 - Not Found..... | 295 |
| 500 - Internal Server Error..... | 296 |
| Testing vs. Production..... | 298 |
| Quick Middleware..... | 298 |
| Afterword..... | 299 |
| On The Tag-Line..... | 299 |
| On Documentation..... | 299 |
| Additional Functionality in the Pipeline..... | 301 |
| Appendices..... | 305 |
| Panl URL Bindings..... | 305 |

| | |
|---|------------|
| User Defined URL Bindings..... | 305 |
| Panl Internal Functional URL Bindings..... | 306 |
| Internal Testing URL Bindings..... | 307 |
| Definitions..... | 308 |
| Command Line Options..... | 313 |
| Panl Server..... | 313 |
| Panl Generator..... | 313 |
| Usage Text..... | 314 |
| Sample .properties Files..... | 317 |
| The Sample panl.properties Files..... | 317 |
| The Sample <panl_collection_url>.panl.properties Files..... | 318 |
| Solr Version 8 & 7 Integration Notes..... | 319 |
| SolrJ..... | 319 |
| Solr Configuration files..... | 319 |
| JSON Response Object..... | 319 |
| Setting up a Solr 7 or 8 server..... | 320 |
| Additional Solr Version 7 Integration Notes..... | 325 |
| Solr Versions 6 and below..... | 326 |
| End Plate..... | 328 |

~ ~ ~ * ~ ~ ~

README.1ST

Welcome To Synapticloop Panl

Synapticloop Panl is a light-weight application server that is designed to sit 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) URLs without having to have a deep understanding of the mechanics behind it.

Some examples contained in this book also contain the conversions that Panl performs, and the Solr query that is executed, 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 URL path to increase readability, for example,

The LPSE URL 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 URL path

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

3. **BOOLEAN value 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 URL path of `/Caran+d'Ache/true/Black/bDW/` the true value (which is defined as whether the mechanical pencil can be disassembled) could be replaced

with ‘Able to be disassembled’ for true values, and ‘Cannot be disassembled’ for false values. The above URL 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, it will throw an exception. Panl protects against this by attempting to parse the value as best it can, and silently dropping the parameter if it cannot be sensibly¹ parsed. This is only done for numeric types (integer, long, float, and double).
5. **HIERARCHICAL facets** - Only show facets if a separate facet is currently selected, allowing you to lead users through the search journey, only displaying facets that help the user narrow down their results.

For example, you may be presenting a search page for Cars and you only want to show the car models once the brand of cars has been selected first.

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. **MORE facets** - Solr (and Panl) configures a limit for the maximum number of facet values that are returned, this functionality enables you to load more facet values if they are available but weren't returned with the results by default.
8. **RESULTS 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.
9. **PAGINATION** - Panl will return all of the variables required to easily generate pagination URL paths giving you options and control over your own implementation.

The returned variables are the number of pages of results, the number of results per page, the total number of results, the current page number and whether the returned results are an exact number.

¹ 'Sensibly' is a bit of a vague term... Panl strips out any unexpected characters and ensures 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. For values that cannot be converted, the value will be ignored and not passed through to the Solr server.

10. STATIC SITE GENERATION - With the exception of a query parameter, all available links for every conceivable URL 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.

11. STATELESS - No state is stored in the Panl server, all of the state is from the URL path part that is requested. No sessions, no memory, nothing to backup, easy to update and quick to start and restart.

12. TEXT CONFIGURATION - All configuration for Panl is based on text files (Java `.properties`) files so they can be stored in a source code management system. Additionally, upgrades to the Panl server are easy and with quick startup times, any configuration changes will be seen instantly.

About This Book

This book describes and explains the functionality of the Panl server, how to configure the server, and how this impacts the generated URL 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 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 the following terms are the foundation and are used throughout the book:

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.

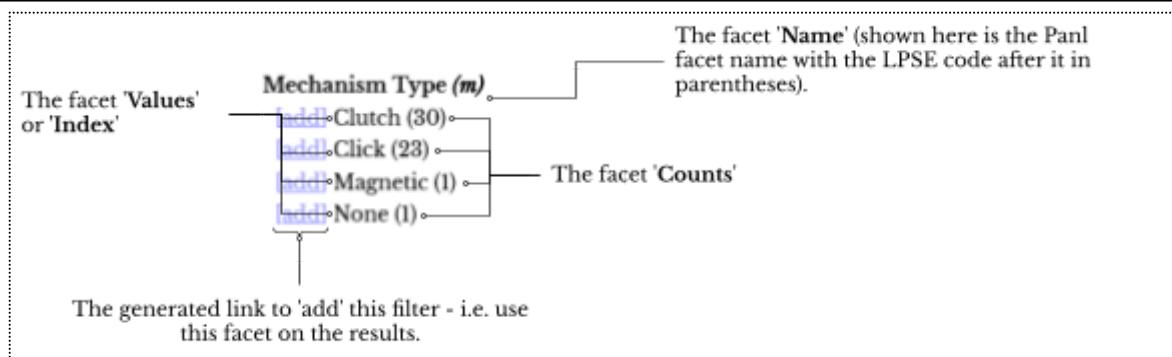


Image: A screenshot of the Panl Simple Results Viewer showing the 'Mechanism Type' facet and describing the parts of the returned Solr facet information

In the above image:

- The Solr field 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, which is output for reference in the Panl Results Viewer web app.
- The facet values represent the indexed attribute 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

- The [add] link is generated by the Panl Simple Results webapp from the returned JSON results object. This link is in the Panl LPSE form.

Search term

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

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 URL Paths - Panl allows many different groups of fields (the FieldSet) to be bound to a specific Collection which is a unique URL served by Panl.

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

Collection(s)

Solr collections are an index of documents that can be filtered or searched upon.

Panl collections are collections of URL paths and FieldSets

LPSE codes

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). These codes are placed in the last path part of the URL.

LPSE path

The LPSE path is a string of URL path values, that, in conjunction with the LPSE codes above is how the Panl server decodes the URL into a Solr server query.

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_url>.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 which determines what features can be used by the Panl server.

Solr query

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

```
q=*&q.op=OR&facet.limit=100&fl=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_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 URL 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

Footnotes²

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 there may be non-obvious side-effects.



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.

² Thanks for checking out this footnote.

For multi-line code related snippets the text appears in a black-bordered grey box prefixed by a line number so that they can be referenced within the description text.

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) |
|---|
| <code>\the\command\that\needs\to\be\run -with "a parameter" -and-another \ "long parameter that may span many lines"</code> |

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/paml-results-viewer/mechanical-pencils/firstfive/Manufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/17+grams/bWGLw/>

Links to other sections or chapters 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 URL paths into complex Solr search queries. Rather than adding query parameters to the URL, Panl automatically generates and returns complete URL 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 URL path, converting a URL 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 descending (sb-), then by
- Pencil Model (sN-)

Which is then passed through to the Solr search server as the following query:

```
q=*&fq.op=OR&facet.limit=100&fl=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
```



Tip: See the section on [Panl Server](#) for full details on the startup command line 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 **3,136 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 codes³:

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

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

- With the five mandatory codes and the one optional code⁴:

```
(62 - 6)^lpse_length = 56^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 URL path becomes much longer, more quickly, subtly negating the value of the encoding of the URL 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.

Panl URL Structure

Collection Request Handlers

The collection request handler responds with a JSON object that contains the results of a search query with all available facets and documents with the defined FieldSets

Within the Panl server the CaFUPs are defined within the configuration in the form

```
/<panl_collection>/<fieldset>/<lpse_path>/<lpse_codes>/
```

And will uniquely return facets and documents with the configured FieldSets. These URLs are configured through the `<panl_collection_url>.panl.properties` files and will serve as many URLs as configured.

Additionally, there are in-built Panl server URLs which provide additional functionality, the predefined URLs are as follows:

Panl Single Page Handler URLs

The Panl single page handler URLs are designed to return a JSON object that will allow building a single page search interface, they are bound to the following URLs:

```
/panl-single-page/<panl_collection>/
```

Where `<panl_collection>` is the Panl Collection that the single page search UI should be built from.

⁴ 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



Note: Do not confuse this handler and URL with the in-built Panl Single Page Search UI example bound to the `/panl-single-page-search/<panl_collection>/` URL,

Panl More Facets Handler URLs

The Panl more facets handler URLs are designed to return a JSON object that will provide more facet values for a specific facet and are bound to the following URLs:

```
/panl-more-facets/<panl_collection>/<fieldset>/<lpse_path>/<lpse_codes>/?code=<lpse_code>&
limit=<limit>
```

Where:

- `panl-more-facets` is the start of the URL that the Panl server has bound the 'More Facets' handler to
- `<panl_collection>` is the Panl collection
- `<fieldset>` is the FieldSet for the fields that will be return with the Solr documents
Note: This value is ignored and replaced by the More Facets handler with the 'empty' FieldSet as there is no need to return any other documents
- `<lpse_path>` is the encoded values for the facets
- `<lpse_codes>` are the LPSE codes for the `<lpse_path>` above
- `code=<lpse_code>` is the query parameter LPSE code for the facet that the additional facet values are requested
- `limit=<limit>` is the query parameter for the maximum number of facet values to return. **Note:** If this is set to -1, then all facets will be returned.

This will perform the search and only return the additional facet values for the Solr facet designated by the LPSE code `<lpse_code>` up to the limit designated by `<limit>` (or all if the limit is set to -1).



IMPORTANT: You CANNOT have a Panl collection start with the prefix '`panl`', it is reserved for internal use. If one is attempted to be registered, it will be rejected and the server WILL NOT start.

In-Built Web Apps (Viewer / Explainer / Single Search Page)

For testing and debugging of the configured properties, the Panl Results Viewer, Panl Results Explainer, and Panl Single Search Page 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 / Single Page Search, or to not allow public access to these URLs. This can be done by setting the `panl.results.testing.urls=false` property in the `panl.properties` file.

'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 faceted search interface, able to highlight all of the functionality of the 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 user.

When the Solr and Panl configuration is set up, the server is up and running, and the testing web apps enabled, it is accessible at:

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

Panl

a very simple results viewer (or [explainer](#) or [single page search](#))

Available collections/fieldset URI Paths (CaFUPs): 1

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

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

Search

Active Filters 5

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

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

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

Range Filters 6

Weight (*w*) Range

from light to heavy pencils
[\[Apply\]](#)
(Actual dynamic range: 4 to 43)

Available Filters 7

Brand (*b*)
[\[add\]](#) Manufactured by Koh-i-Noor Company (9)
[\[add\]](#) 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 10 result(s) 9 q.op AND || OR

Sort by Brand: ASC [DESC](#) || Pencil Model: [ASC](#) [DESC](#)
 Then sort by Pencil Model: [ASC](#) [DESC](#) 10

« PREV [NEXT](#) » 12

Solr: query time 89ms.
 Panl: parse request 4ms, build request 12ms, send and receive request 76ms, parse response 2ms. Total time 95ms. 13 Show [3](#) [5](#) [10](#) per page 14

| | | |
|---------------------|-----------------------|----|
| Pencil Model (name) | Pro-Matic | 15 |
| Brand (brand) | Alvin | |
| Pencil Model (name) | Scott No. B/2 | |
| Brand (brand) | Alvin | |
| Pencil Model (name) | Tech-DA | |
| Brand (brand) | Alvin | |
| Pencil Model (name) | Criterium | |
| Brand (brand) | BIC | |
| Pencil Model (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 URL Paths (CaFUPs) that Panl is configured to serve. CaFUPs enable different Solr fields and facets to be returned from the same Solr collection.
2. A textual representation of the CaFUP that the Panl Results Viewer web app currently is using.
3. The canonical URL 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 URL 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 URL.

4. The search query box, by default, Panl responds to the same URL 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 URL 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 users to select a range of values - the values are inclusive (i.e. include the minimum and maximum values). RANGE filters also include dynamic maximum and minimum values so that the range that is rendered can be automatically updated.

DATE Range filters (not shown⁵), enabling searching over a range of dates (but not a specific date) in the form of:

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

For example:

⁵ 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.

Last 30 days
 Previous 24 hours
 Next 3 years

7. **Available filters**, additional facets that can further refine and limit the Solr search results. This may also display a link to load more facets if the returned number of facets is not the complete set.
8. **Number of results found**, and whether this is an exact match.
9. **Query operand** - whether the Solr search term 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 progressive 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 Panl Results 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 URL paths. In the image, only two fields are configured for this CaFUP, namely Brand, and Pencil Model.

'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 URL paths, the configuration of the Panl collection URL, 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:

Explain

Request Token Explainer

No URI path entered, nothing to explain.

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')

...

Field Configuration Explainer

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.
```

...

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 URL Paths (CaFUPs) that Panl is configured to serve. CaFUPs enable different Solr fields to be returned in the

documents with the same search parameters. Clicking on these links will populate the 'Configuration Parameters' and 'Field Configuration Explainer' sections.

2. **A textual representation of the CaFUP** that the Panl Results Explainer web app is currently using.
3. **The canonical URL path entry field** allows you to enter any canonical URL 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 URL path came from MUST match the CaFUP on the results viewer.
4. **The request token explainer** - for any canonical URL 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.

'Simple' Panl Single Search Page Web App

Panl also binds a URL path to enable the building of a single search page interface, and binds a URL path to view a working example of what the single search page could look like.

When the Solr and Panl configuration is set up, the server is up and running, and the testing web apps enabled, it is accessible at:

<http://localhost:8181/panl-single-page-search/>

The screenshot shows the Panl Single Search Page interface for a collection of mechanical pencils. The interface is organized into several sections:

- Facets:**
 - Brand (Regular):** Includes options like Koh-i-Noor, Caran d'Ache, Faber-Castell, Pentel, Alvin, Kaweco, Rotring, Hightide Penco, Kita-Boshi, Keen, Mitsubishi, OHTO, Scrikks, Staedtler, BIC, DEDEPRAISE, Ito-Ya, Mr. Pen, Muji, Redcircle, Unbranded, WSD, YStudio.
 - Mechanism Type (Regular):** Includes options like Clutch, Click, Magnetic, None.
 - Colours (Regular):** Includes options like Black, Blue, Red, Green, Silver, Yellow, Purple, White, Brown, Grey, Orange, Wood, Pink, Brass, Gold.
 - Grip Shape (Regular):** Includes options like D5, Cylindrical, Hexagonal, Triangular.
 - Category (Regular):** Includes options like Entry Level, Consumer, Prosumer, Everyone.
 - Maximum Allowed Lead Length (Regular):** Includes options like 130, 120, 90.
 - Hardness Indicator (Regular):** Includes options like No, Yes - at top, Etched on body, Yes - at grip, Yes - on clip.
 - In Built Eraser (BOOLEAN):** [No selection]
 - Lead Size Indicator (Regular):** Includes options like No, Etched on body, Etched on cap.
 - In Built Sharpener (BOOLEAN):** Has in-built sharpener
 - Weight (RANGE):** A slider ranging from 10 to 50.
- Search Bar:** A button labeled "Search".
- Footer:** Includes a link to the Panl Results Viewer web app, a note about the Panl LPSE encoding mechanism, and a "Happy searching" message.

Image: The In-Build Panl Single Search Page interface web app for the mechanical pencils collection

1. A list of available example Single Search page interfaces that Panl is configured to serve. CaFUPs enable different Solr fields to be returned in the documents with the same search parameters. Clicking on these links will generate a working sample single search page.
2. The generated LPSE path that the selections from the search interface will apply.
3. All facets that can be selected, presented for the different types of facets, namely OR, RANGE, DATE Range, BOOLEAN, and regular facets .
4. The generated LPSE path that the selections from the search interface will apply.
5. The search button that will take you to the in-build Panl Results Viewer web app so that you can view the results instantly.

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 `<panl_collection_url>.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.



Tip: 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 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 URL paths throughout the entire search journey.

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

In addition to the Panl Niceties detailed in the previous section, Panl was designed to

1. Have SEO friendlier URL paths with much shorter URLs than traditional query parameters

This was the primary driver.

2. Abstract away the complexities of the Solr query string - being able to have a simple interface through the URL which could generate complex queries.

Not having to fully understand how Solr works in the back-end abstracts away the complexity of a front-end integrator and reduces the need to have the back-end and front-end understand each-other.

3. Be quick to start up and easy to configure

During development of a solution, being able to iterate over a solution, or change the way that Panl is configured is a must have. Additionally, being able to upgrade the Panl server and have the configuration files be automatically picked up and work without any changes is a plus.

4. Protect Solr from errant queries

Hiding the Solr implementation details from the end user and parsing, decoding, and validating the URL before passing the query through to Solr. Additionally, Solr has a tendency to return an internal server error when the query string is not as it expected, and this should not disturb the return of the results.

5. Be able to present the same Solr collection in multiple different ways

A single Solr collection should be able to serve up different fields and facets from the result documents without any back-end logic.

6. Have a configuration file drive the generation of the UI as much as possible

Rather than hard-coding facets and then determining how to display them, being able to have a returned JSON response which can be interrogated to determine how the facets should be displayed.

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 URL path as a code.

In effect the (29 characters) Panl LPSE query of

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

For the above Panl URL 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 equivalent⁶ 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=catalog&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
```

⁶ 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.

The effect of the two queries are identical, returning the same results, just with a much shorter URL.

As an example of a search performed on the Walmart site, the search URL of (281 characters)

https://www.walmart.com/search?q=mechanical+pancils&min_price=4&catId=1229749_9412206&facet=subscription_eligible%3ASubscription+item%7C%7Cexclude_oos%3AShow+available+items+only%7C%7Cpencil_lead_size_pencils%3A2.0+mm+to+3.0+mm%7C%7Ccount_per_pack_facet%3A1+-+4%7C%7Ccolor%3ABlack

If implemented with the Panl server could be the following URL of (196 characters):

[https://www.walmart.com/search/mechanical-pencils/from-4\\$-to-15\\$/pencils-and-pencils-sharpenerssubscription-eligible/in-stock/lead-size-from-2.0mm-to-3.0mm/4-items-per-pack/black/qp+pdsil+lrc/](https://www.walmart.com/search/mechanical-pencils/from-4$-to-15$/pencils-and-pencils-sharpenerssubscription-eligible/in-stock/lead-size-from-2.0mm-to-3.0mm/4-items-per-pack/black/qp+pdsil+lrc/)

Which is shorter, far more readable, and much more SEO friendly.

The three types of LPSE codes

1. **Filters** - codes that filter (and generally narrow⁷) the results - these are either the search term or a facet (Regular, Boolean, OR, DATE Range, or RANGE)
2. **Parameters** - codes that change the way the returned results are displayed - these are the sorting orders, number of results per page, and page numbering. These do not filter 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.

URL 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 URL path:

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

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

⁷ 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 CaFUP
`/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 URL path is made up of the following parts

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

When references to the URL path are made throughout this book, they refer to 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 URL 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 a 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 9.6.1, 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-9.6.1-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. You will need to replace this value with your own installation directory.

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-9-1.2.0`, 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-9-1.2.0/ | 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 URL bindings for testing across all CaFUPs. |
| mechanical-pencils/ | <p>NOTE: This includes configurations for the mechanical-pencils, simple-date, and book-store Solr collections. You will need to index the underlying datasets for this configuration to work.</p> <p>The directory that contains the Panl configuration for the Mechanical Pencils Panl collection URL bindings</p> |
| mechanical-pencils-or/ | <p>NOTE: To run examples in this book you MUST create and populate the Solr mechanical-pencils collection. The process for this will be explained in the following chapters.</p> <p>NOTE: This directory also contains an example implementation of the 'More Facets' functionality which is not registered in the panl.properties file.</p> <p>The directory that contains the Panl configuration for</p> |

| File Path | Explanation |
|----------------------------------|---|
| <code>simple-date/</code> | the Mechanical Pencils Panl collection URL bindings with the <code>brand</code> Solr field configured to be an OR Facet |
| <code>book-store/</code> | The directory that contains the Panl configuration for the Simple Date Solr Panl collection URL bindings |
| <code>solr/</code> | NOTE: Only brief sample data is included for this collection |
| <code>mechanical-pencils/</code> | Directory for testing/sample Solr configuration |
| <code>simple-date/</code> | The directory for the Solr configuration XML file and managed schema file for the Mechanical Pencils collection |
| <code>book-store/</code> | The directory for the Solr configuration XML file and managed schema file for the Book Store collection. |
| | NOTE: Only brief sample data is included for this collection |

~ ~ ~ * ~ ~ ~

Panl Server Versions

Naming of the versions of the release packages follow the Solr major release version appended with the Panl release version.

For example, the release package `solr-panl-9-1.2.0` is designed to integrate with Solr version `9.x.x` and the Panl release version is `1.2.0`.



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

For other versions and integration pointers - 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.

See the Appendices for [Setting up a Solr 7 or 8 server](#) for more information.

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/>



IMPORTANT: This is the Quick Start guide for Solr version 9, if you are using a previous version of Panl, ensure that you have the correct Panl release package and that you refer to the [Setting up a Solr 7 or 8 server](#) section in the Appendix for how to index data for those versions.

Panl

a very simple results viewer (or [explainer](#) or [single page search](#))

Available collections/fieldset URI Paths (CaFUPs):

mechanical-pencils - [\[default\]](#) - [\[firstfive\]](#) - [\[brandandname\]](#) - [\[empty\]](#)

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

Canonical URI: [\[explain\]](#)

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

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, with the mechanical pencils CaFUPs shown



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 Microsoft 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-9.1.2.0` version:

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

Download the latest version of Apache Solr - this book is using the `9.6.1-slim` version:

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



WARNING: Solr version 9.7.0 has changed command line options, and some of the in-built scripts are not fully working and there are changes required to the passed in parameters in these examples.

If you are using Solr version 9.7.0 then you will need to replace the command line option of `-noprompt` to `--no-prompt` when creating the initial Solr cloud example.

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) |
|-----------------------------------|
| 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 ↲ 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.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/panl-results-explainer/> in your favourite browser, choose a collection and fieldset, paste the canonical URL 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 <  
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.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 URL 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.

Next Steps

For indexing additional data, see the **Additional Data** section, for deeper understanding of how to set up a new collection and search page from scratch see the **A Walkthrough Example - The Book Store** 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.⁸ 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 configure and run a Panl server, and
4. What features are available through configuring the Panl server.



IMPORTANT: This is the Getting Started guide for Solr version 9, if you are using a previous version of Panl, ensure that you have the correct Panl release package and the you refer to the [Setting up a Solr 7 or 8 server](#) section in the Appendix for how to index data for those versions.

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-9-1.2.0` version:
<https://github.com/synapticloop/panl/releases/>

⁸ 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.

2. Download the latest version of Apache Solr - this book is using the 9.6.1-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:

The Solr server

On a Windows machine the installation directory was:

C:\java-servers\solr-9.6.1-slim\

whilst on a *NIX the installation directory was:

/Users/synapticloop/java-servers/solr-9.6.1-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-9-1.2.0\

whilst on a *NIX⁹ the installation directory was:

/Users/synapticloop/java-servers/solr-panl-9-1.2.0/

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) |
|---|
| <pre>cd SOLR_INSTALL_DIRECTORY bin\solr start -e cloud -noprompt</pre> |

⁹ I am using an Apple Mac system, but it is the same for Linux.

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

```
cd C:\java-servers\solr-9.6.1-slim\  
bin\start -e cloud -noprompt
```

*NIX:

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

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

```
cd /Users/synapticloop/java-servers/solr-9.6.1-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:

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

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/sample/solr/mechanical-pencils/` directory.

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

Windows:

| Command(s) |
|---|
| <code>cd SOLR_INSTALL_DIRECTORY</code> |
| <code>bin\solr create -c mechanical-pencils -d <PANL_INSTALL_DIRECTORY\sample\solr\mechanical-pencils\ -s 2 -rf 2</code> |

*NIX:

| Command(s) |
|---|
| <code>cd SOLR_INSTALL_DIRECTORY</code> |
| <code>bin/solr create -c mechanical-pencils <-d PANL_INSTALL_DIRECTORY/sample/solr/mechanical-pencils/ -s 2 -rf 2</code> |

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 |
|--|---|-----------|------------|---------------|--------------------------|--------------------------------------|---|
| localhost Windows 10 7.9Gb Java 17 Load: -1 show details... | 7574_solr Uptime: 1m show details... | 1% | 18% | 276.0b | RPM: 0.72 p95: 5172ms | gettingstarted mechanical-pencils | gettingstarted_s1r4 (0 docs) gettingstarted_s2r2 (0 docs) (2 more...) |
| | 8983_solr Uptime: 2m show details... | 9% | 41% | 276.0b | RPM: 0.72 p95: 162ms | gettingstarted mechanical-pencils | gettingstarted_s1r6 (0 docs) gettingstarted_s2r1 (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) |
|--|
| <code>cd SOLR_INSTALL_DIRECTORY</code> |

Command(s)

```
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 (or [explainer](#) or [single page search](#))

Available collections/fieldset URI Paths (CaFUPs):

[mechanical-pencils](#) - [\[default\]](#) - [\[firstfive\]](#) - [\[brandandname\]](#) - [\[empty\]](#)

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

Canonical URI: [\[explain\]](#)

| | | |
|-------------------|--|---------------|
| Search Query | Search Results | q.op |
| | Page of Showing results per page This page has result(s) | |
| Active Filters | Sort by | |
| Available Filters | | 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, faceting 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 check the logs and attempt to understand what the logging tells you, and if all else fails, 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:

- 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
- No conflicting configuration is used from previous installations

Checking the Logs

The Solr web app has an in-built administration console which contains logging. Whilst it can be confusing, it can give some hints as to what is happening and if there are any errors.

The Solr administration console can be viewed at:

<http://localhost:8983/solr/#/~logging>



The screenshot shows the Solr Cloud Admin's logging page. On the left, there is a sidebar with navigation links: Dashboard, Logging (which is selected), Level, Security, Cloud, Schema Designer, Collections, Java Properties, Thread Dump, Collection Selector, and Core Selector. The main area is titled "Log4j2" and displays a table of log entries. The columns are Time (Local), Level, Core, Logger, and Message. The log entries are as follows:

| Time (Local) | Level | Core | Logger | Message |
|------------------------|-------|-------|---------------------|---|
| 27/09/2024, 9:26:44 am | WARN | false | StartupLoggingUtils | Jetty request logging enabled. Will retain logs for last 3 days. See chapter "Configuring Logging" in reference guide for how to configure. |
| 27/09/2024, 9:26:44 am | WARN | false | Solr2kServer | Embedded Zookeeper is not recommended in production environments. See Reference Guide for details. |
| 27/09/2024, 9:26:44 am | WARN | false | ServerCnxnFactory | maxCnxns is not configured; using default value 0. |
| 27/09/2024, 9:26:45 am | WARN | false | NIOServerCnxn | Close of session 0x0 |
| 27/09/2024, 9:26:45 am | WARN | false | ClientCnxn | Session 0x0 for server 127.0.0.1/127.0.0.1:9983,8203; Closing socket connection. Attempting reconnect except it is a SessionExpiredException. |
| 27/09/2024, 9:26:47 am | WARN | false | ZKController | Contents of zookeeper /security.json are world-readable; consider setting up ACLs as described in https://solr.apache.org/guide/solr/latest/deployment-guide/zookeeper-access-control.html |
| 27/09/2024, 9:27:16 am | WARN | false | CoreContainer | Not all security plugins configured! authentication=disabled authorization=disabled. Solr is only as secure as you make it. Consider configuring authentication/authorization before exposing Solr to users internal or external. See https://s.apache.org/solrsecurity for more info |
| 27/09/2024, 9:27:17 am | WARN | false | MessagingBinders | A class jakarta.activation.DataSource for a default provider MessageBodyWriter<jakarta.activation.DataSource> was not found. The provider is not available. |

At the bottom of the table, it says "Last Check: 27/09/2024, 1:02:17 pm". To the right of the table, there are buttons for "Auto-Refresh" (with a checked checkbox) and "Show dates in UTC". Below the table, there are links to Documentation, Solr Query Syntax, Community, Issue Tracker, Slack, and JIRA. Two annotations are present: one pointing to the "Auto-Refresh" button with the text "Click on the 'Auto-Refresh' link to turn it 'Off'" and another pointing to the "i" icon in the log entries with the text "If there is more information to be displayed, click on the 'i' button".

Image: The Solr Cloud Admin's logging page

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.

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)

```
cd SOLR_INSTALL_DIRECTORY

bin\solr stop -all
```

***NIX:**

Command(s)

```
cd SOLR_INSTALL_DIRECTORY  
  
bin/solr stop -all
```

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.

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

If nothing else seems to work, delete the olr 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 contents¹⁰).
3. Re-create the Solr cloud instance,
4. Re-create the collection(s), and
5. Re-index the data

Next Steps

For indexing additional data, see the [Additional Data](#) section, for deeper understanding of how to set up a new collection and search page from scratch see the [A Walkthrough Example - The Book Store](#) section.

~ ~ ~ * ~ ~ ~

¹⁰ Commands weren't included in this as a 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 explicitly include commands and instructions for setting up. Some references are made to the additional data whilst working through this book, whilst not necessary to have the datasets indexed, 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 (including the Book Store Walkthrough Example), will provide access to all CaFUPs. **Remember:** Panl was designed to have multiple CaFUPs connecting to multiple Solr collections.

The 'all' `panl.properties` file binds the following Solr collections to the Panl CaFUPs:

| Solr Collection | Panl CaFUPs |
|---------------------------------|---|
| <code>simple-date</code> | <code>simple-date</code> <ul style="list-style-type: none"> - An example data set showing DATE Range facets |
| <code>mechanical-pencils</code> | <code>mechanical-pencils</code> <ul style="list-style-type: none"> - The default dataset based on the Mechanical Pencils data |
| | <code>mechanical-pencils-or</code> <ul style="list-style-type: none"> - The same dataset as above, but using an OR facet for the brand facet |
| | <code>mechanical-pencils-more</code> <ul style="list-style-type: none"> - The same dataset with the 'More Facet' links available |
| <code>book-store</code> | <code>book-store</code> <ul style="list-style-type: none"> - The data set for the walkthrough section in building your own collection and interface. |

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. 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) |
|--|
| <pre>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</pre> |

*NIX Commands

| Command(s) |
|--|
| <pre>cd SOLR_INSTALL_DIRECTORY bin\solr create -c simple-date -d ← PANL_INSTALL_DIRECTORY\sample\solr\simple-date\ -s 2 -rf 2</pre> |

Command(s)

```
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.panl.properties
```

Panl will automatically add the new CaFUPS to the Panl Results Viewer and Panl Results Explainer web apps. Additionally it will add the collection to the Single Search Page web app.

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/mec-
hanical-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.

Mechanical Pencils More Facets

The `mechanical-pencils-more` sample collection includes configuration that reduces the maximum number of facets to be returned to 10. This will enable the Panl Results Viewer to generate 'See all...' links for facets that have not returned the complete list of facets for the query.

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/m
echanical-pencils-more.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/mec-
hanical-pencils-more/mechanical-pencils-more.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 Walkthrough 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 store¹¹ with the underlying data containing 3 million records - this dataset is not included with the release package (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_url>.panl.properties` files. Remember: you may have multiple CaFUPs,*

¹¹ 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 collections 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 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 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, but only when an author has been selected*
- **Price**
 - *Select a price range for books*
 - *Be able to sort by price range*

Derived Data

- **ID** - derived from the database primary key
 - *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

Whilst we are defining how the data will be indexed, we need to keep in mind the configuration of the Panl server as well. 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 Panl to 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. **WARNING:** if the schema name starts with the string `panl-` then the Panl server will fail to start.

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 analysed by Solr and searched upon with a keyword or keywords.

Line 33:

For clarity and space, the Solr field definitions and additional XML elements were not included and replaced by 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 | |

| Solr Field Name | Data Type | Facet or Field | Facet Type | Sortable | Additional information |
|-----------------------|-----------|----------------|-------------|----------|-----------------------------|
| Description | String | Field | N/A | No | |
| Book Image | String | Field | N/A | No | |
| 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 `<panl_collection_url>.panl.properties` files defined

The Default Search Page Configuration

To generate the default search page configuration the in-built Panl generator 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 cannot be assigned from the first character of the Solr field name (either upper or lowercase) as they are already both in use and will then choose a random one.

*NIX command

Command(s)

```
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
```

Windows command

Command(s)

```
cd PANL_INSTALL_DIRECTORY

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 in this book - 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>

The Generated panl.properties File

The Panl generator utility will output the file to the path passed in through the `-properties` command line option.

For all of the Book Store URL path parts, the `panl.properties` file will be the same. Below (for clarity) all comments have been stripped from the file. Note the last line `panl.collection.book-store=book-store.panl.properties` has been automatically added to the file.

```

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.decimal.point=true
07 panl.collection.book-store=book-store.panl.properties

```



Note: The above is configured for testing purposes, with verbose error messaging and testing URLs live.

The Generated <panl_collection_url>.panl.properties File

The second file that the Panl generator utility will write is the `<panl_collection_url>.panl.properties` file. This file has three major parts (in order):

1. The general properties configuration,
2. The generated fields and facets configuration, and
3. The Panl LPSE order, FieldSets, and sorting

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
```

No changes for the default value of '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 URL 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 is always available as a sort order and is the default sort order if no other sort order is selected. The order of the value of this property is the order that Panl will return in the JSON response object.

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.multivalue.a=true
06 panl.prefix.a=Author
07 #panl.suffix.a=suffix
08 panl.when.a=q,A
09 #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 8 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.multivalue.g=true
07 #panl.prefix.g=prefix
08 #panl.suffix.g=suffix
09 #panl.when.g=
10 #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. **Note:** This is already a multivalued Solr field, however using this as an OR facet means that you can select books which are 'Sci-Fi' OR 'Horror', rather than a book that is 'Sci-Fi' AND 'Horror'

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 URL 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 URL 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 URL 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 URL 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. (Line 8)

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,\t
02 i,\t
03 a,\t
04 \t,\t
05 \d,\t
06 b,\t
07 B,\t
08 g,\t
09 N,\t
10 f,\t
11 l,\t
12 I,\t

```

```
13 S, \
14 P, \
15 A, \
16 D, \
17 T, \
18 L, \
19 q, \
20 p, \
21 n, \
22 s, \
23 o
```

The final version becomes

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

The Panl FieldSets

There are always going to be at least two FieldSets defined for any Panl collection, namely:

1. `default` - this is **ALWAYS** available, and if not set then it will return **ALL** fields in the Solr collection, whether you have defined them in the Panl configuration file or not. The recommendation is to either ignore this in your implementation, or edit this FieldSet to your purposes, as has been done below.
2. `empty` - this is **ALWAYS** available, and if it appears in the properties file, a warning will be printed and it will be ignored. This will return no fields for the document (i.e. no documents at all).

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,\n
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.or.facet.g=true</code> |
| Number of Pages | N | No changes to Panl field/facet configuration |
| First Published Year | f | Added a prefix by setting <code>panl.prefix.f=First published in</code> <i>(note the space at the end of the property)</i> Added to sort fields by adding to the property <code>panl.sort.fields</code> |
| Language | l | No changes to Panl field/facet configuration |
| Paperback / Hardcover | I | Changed the Panl name by setting <code>panl.name.I=Book Format</code> Added BOOLEAN value replacement values by setting <code>panl.bool.I.true=Paperback</code> <code>panl.bool.I.false=Hardcover</code> |
| Series | s | Added hierarchy by setting <code>panl.when.S=a</code> |

| Panl Field Name | LPSE code | Changes |
|-----------------|-----------|--|
| Price | P | <p>Made this a RANGE facet with value replacement by setting the following</p> <pre>panl.range.facet.P=true panl.range.min.P=5 panl.range.max.P=100 panl.range.prefix.P=From (note the space at the end of the property) panl.range.infix.P=\ to (note the space at the end of the property) panl.range.suffix.P=\ dollars panl.range.min.wildcard.P=true panl.range.max.wildcard.P=true</pre> <p>Added to sort fields by adding to the property</p> <pre>panl.sort.fields</pre> |

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

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

4. Determine the Web Pages to Render

In addition to the default (i.e. main) search page with its functionality, additional 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 Process¹²

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

1. 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.
2. 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.
3. 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.
4. 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 19.989999771118164 - 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.

~ ~ ~ * ~ ~ ~

¹² Or, the mistakes that were made with the implementation.

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. The value of the attribute is then used to lookup the Solr class value from the `fieldType` element in the managed schema file. This `fieldType` is then output to the property file.

```
# <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 | DATE Range | OR | BOOL Value Replace |
|-----------------------|-----------------|-------|------------|-----|--------------------|
| solr.BoolField | YES | NO | NO | NO | YES |
| solr.DatePointField | NO | NO | YES | NO | NO |
| solr.DoublePointField | YES | YES | NO | YES | NO |
| solr.FloatPointField | YES | YES | NO | YES | NO |
| solr.IntPointField | YES | YES | NO | YES | NO |
| solr.LongPointField | YES | YES | NO | YES | NO |
| solr.StrField | YES | NO | NO | YES | NO |
| solr.TextField | NO | NO | NO | NO | NO |
| solr.UUIDField | YES | NO | NO | YES | NO |



Notes: The `solr.TextField` above should only be used as a Panl field, not a facet. If it is configured as a facet, then every indexed word in all of the fields will be included as a facet.

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, they are just untested.

- `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`



IMPORTANT: The Panl generator will not generate any configuration for the above field types, you will have to manually configure them yourself as fields or facets.

Facet and Field Types

Each of the defined fields in the file can be defined as either a 'Field', or a 'Facet'. If the Panl field is configured to be a 'Field', then it will be returned with the documents, and 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 also indexed (i.e. `indexed="true"` in the managed schema) may be set as a facet, or a field.

Multiple `<panl_collection_url>.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

```

The only other configuration option for a field is the Panl field name - i.e. `panl.name.<lpse_code>` - which is a 'nicer' display name for the field.

Hints/Recommendations:

- Generally, fields that contain a lot of text are better configured as a Panl Field, unless you wish to have the words in the text as individual facets (similar to a word cloud).
- 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 Solr field, then the mapped field type should be at least indexed and it is a good idea to have it stored as well, but ensure that the type is not mapped to a Solr field type that is 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 replacements 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, or does not have value from an SEO perspective.
- If you want to shorten the URL path part further, replace the true/false values with single characters - e.g. 1/0 or y/n

- BOOLEAN replacement values are case-sensitive

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 as a Regular facet. If you do not want the Regular facet values to be returned as well then set the property `panl.range.suppress.<lpse_code>=true`.

```

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
16 panl.range.suppress.w=false

```

Hints/Recommendations:

- Use sparingly, and where it makes sense. Ranges can filter the results down to zero documents if the range of values in the documents falls slightly outside the provided values.
- 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`.

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 facetting 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.

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 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.

```

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

```

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.

Example:

If two facets are set as OR facets, for example Manufacturer and Mechanism Type. Multiple values for either of the two facets can be selected, provided that the Manufacturers that are selected also have documents with the Mechanism Type.

If you were to choose 'BIC' OR 'OHTO' as pencil manufacturers, and 'Click' OR 'None' then the query would of the form:

Select all pencils that are manufactured by BIC or OHTO AND have pencils that are Click or None mechanism types.

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.
- OR facets will not return additional facets if any separate facet is selected.

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 let the user 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>` property for configuration options.

~ ~ ~ * ~ ~ ~

A Brief Introduction To Solr (And Faceted Search)

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. The keyword matches text found within the indexed fields, whilst facets refine results by selecting matching document attributes.

For example, in the Mechanical Pencils dataset, the Faber-Castell Goldfaber wooden pencil has the following data:

```

01 {
02   "brand" : "Faber-Castell",
03   "name": "Goldfaber",
04   "mechanism_type": "None",
05   "nib_shape": "Tapered",
06   "body_shape": "Hexagonal",
07   "grip_type": "None",
08   "grip_shape": "Hexagonal",
09   "cap_shape": "None",
10   "category": "Everyone",
11   "length": "175",
12   "relative_length": "175",
13   "diameter": "8",
14   "weight": "4",
15   "relative_weight": "5",
16   "lead_length": "120",
17   "disassemble": false,
18   "nib_material": "Wood",

```

```

19 "mechanism_material": "N/A",
20 "grip_material": "Wood",
21 "body_material": "Wood",
22 "tubing_material": "None",
23 "clip_material": "None",
24 "cap_material": "None",
25 "hardness_indicator": "Etched on body",
26 "lead_size_indicator": "No",
27 "in_built_eraser": false,
28 "in_built_sharpener": false,
29 "variants": [ "Blue and gold pinstriping" ],
30 "colours": [ "Blue" ],
31 "description": "The basic, everyday pencil from Faber-Castell Goldfaber. A ↵
32   classic hexagonal wooden pencil with gorgeous blue and gold pinstriping. Not ↵
33   a mechanical pencil, but always worth including as a reference point.",
34 "id": 4,
35 "images": [ "400-goldfaber-blue.jpg" ]
36 }

```

The description field is used for the keyword search, whilst the other data is used as the attributes for faceting and fields. In the implementation used in the Panl example setup, other fields are copied to an additional Solr field named text, which is then used for the keyword to be searched against.

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

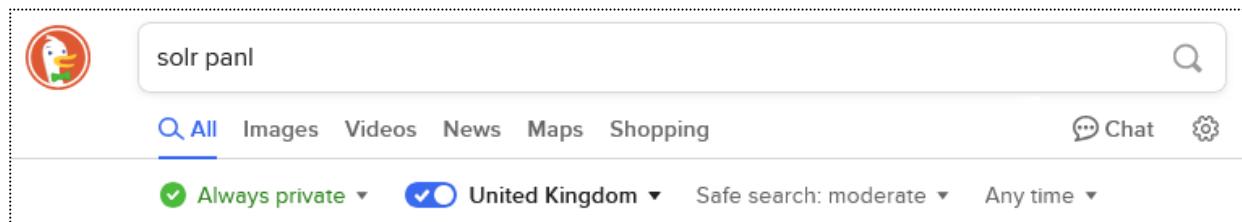


Image: A DuckDuckGo search with the keyword of 'Solr Panl' which will show relevant results and some simple faceting options.¹⁸

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')

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.

For our examples, using a smaller and more targeted search index, there is greater potential to add facets that help the user get to the correct details more quickly.

Individual sites use search keywords and more targeted and relevant 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

¹⁸ At the time of writing no useful results were returned by any of the large search engines for 'Solr Panl'.

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 URL 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_url>.panl.properties` file and for the URL that Panl will bind this collection to. 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_url>.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_url>` first part of the file name is the URL path that the Panl server will respond to, and **__MUST__** be unique amongst all URL 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 URL path to, for the above example the URL 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 URL 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 :

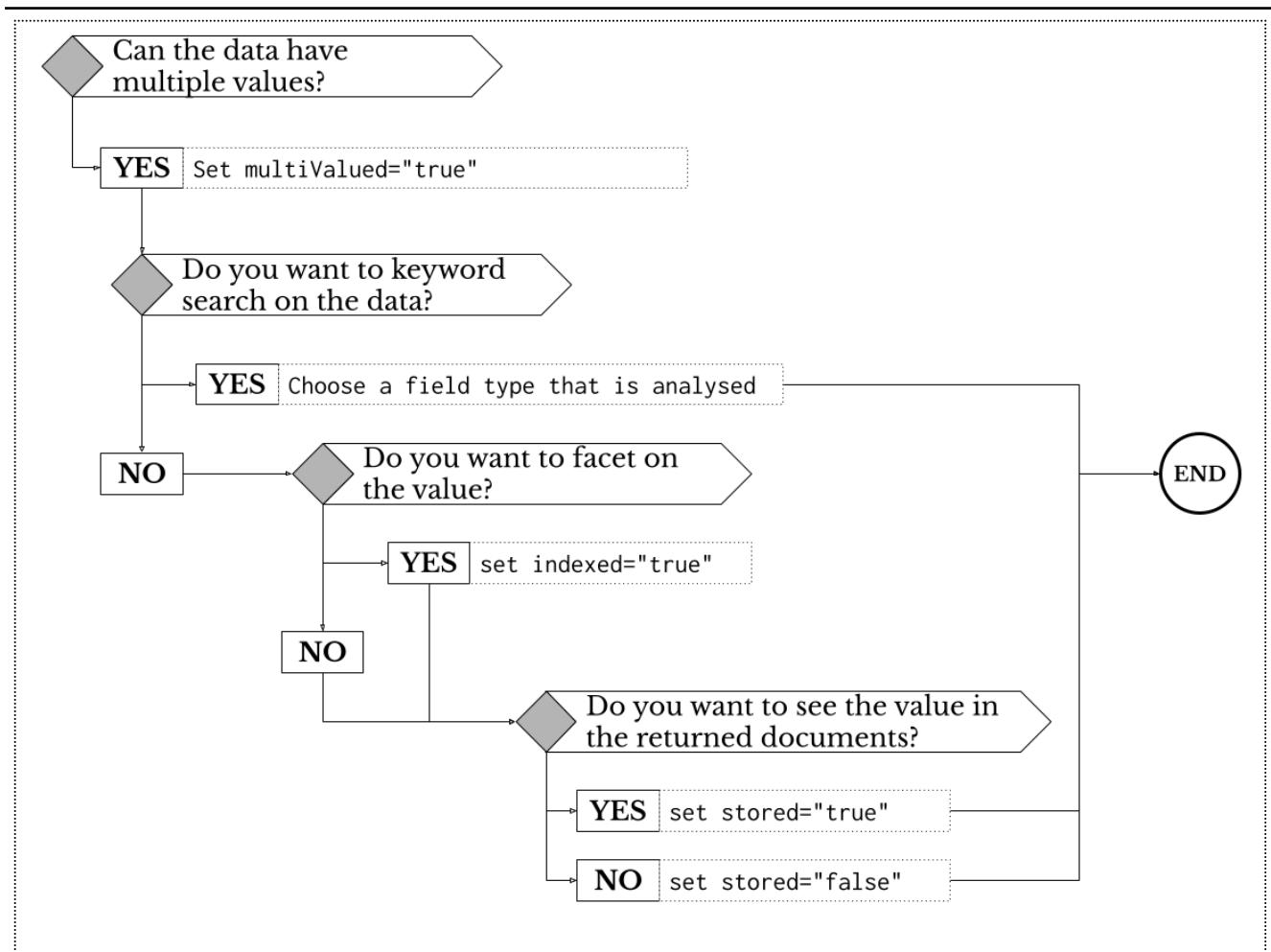


Image: The decision chart for determining the field types and attributes

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" />
```

The value of the `type` attribute is then mapped to a Solr class. 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`'s `name` 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.

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

The type attribute above is matched to the name attribute below

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

This is the Solr fieldType

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">
07      expand="false"/>
08    <filter name="flattenGraph"/>
09    -->
10    <filter name="lowercase"/>
11  </analyzer>
12
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>
18</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 \(1\)](#)
- [\[add\] wsd \(1\)](#)
- [\[add\] 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. This will allow values of non-analysed fields to be copied to this field, analysed, and queried.

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" />
62   <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 of the Solr managed schema file 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_url>.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>



IMPORTANT: Solr will only return fields for which are indexed. I.e. fields in the managed schema file with the `indexed` attribute set to `true`.

#TODO - example of returned results with highlighting

~ ~ ~ * ~ ~ ~

Panl Configuration

The configuration of the Panl server 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 URLs and how the Solr fields are to be processed.

The Panl server is driven by at minimum two configuration files (both of which are java based `.properties` files¹⁴), the `panl.properties` file and at least one (but maybe more) of a `<panl_collection_url>.panl.properties` file.

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

Of the two files mentioned, the `<panl_collection_url>.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, whilst you learn how to configure the Panl server, 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,

¹⁴ 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.

3. Whether to enable the in-built Panl Results Viewer, Explainer, and Single Search Page, and
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 is indexed, 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_url>.panl.properties` file

A `<panl_collection_url>.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 and/or 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 URL 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_url.panl.properties` template:

https://github.com/synapticloop/panl/blob/main/src/main/resources/panl_collection_url.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 Collection and FieldSet URL paths (CaFUPs).

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 - the location of this file 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>



Note: The above link is for the most up-to-date file, including new feature developments. To view the file for the version that you are using, use the tag that matches the version that you are using and browse the code. See <https://github.com/synapticloop/panl/tags> for the version that you are using, and <https://github.com/synapticloop/panl/tree/1.0.0> to view the source (This link will view the source at version 1.0.0).

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 (including options for zookeeper related connections)
- `panl.results.testing.urls`
Whether to enable the testing URLs which will help you understand and debug the Panl LPSE URL paths, and provide a single search page user interface
- `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:

```
solr.search.server.url=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 implementation 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 simple web applications to help with understanding and debugging the Panl LPSE paths and configuration.

If enabled (i.e. set to `true`), the following web apps will be made available:

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



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

generator, you will not be able to generate configuration files that start with the string `panl-`.

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 simple¹⁵ results viewer allowing you to browse all features and functionality of the Panl server.

¹⁵ 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 an incredibly useful tool when adding features to the returned JSON object so that integration and implementation became a more developer friendly experience.

Panl

a very simple results viewer (or [explainer](#) or [single page search](#))

Available collections/fieldset URI Paths (CaFUPs):

[mechanical-pencils - \[default \] - \[firstfive \] - \[brandandname \] - \[empty \]](#)

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

Canonical URI: [\[explain\]](#)

| | | |
|--|--|----------------------|
| Search Query | Search Results | q.op |
| <input type="text"/> <input type="button" value="Search"/> | Page of Showing results per page This page has result(s) | |
| Active Filters | Sort by | |
| Available Filters | | 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 and a working faceted search interface.

Select one of the available CaFUPs links, which will then allow you to search and facet and see the results.

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

The Panl Results Explainer

A simple¹⁶ results explainer detailing how a Panl LPSE URL is parsed and decoded. It also provides an overview of the configuration for the specific collection.

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

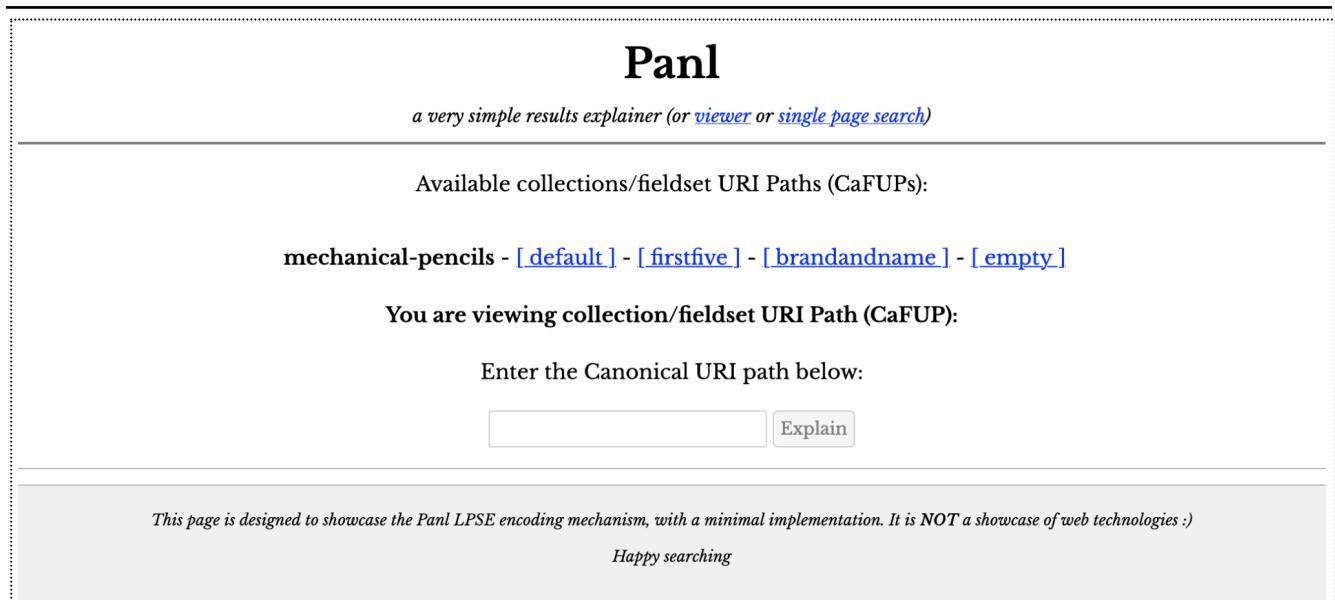


Image: The In-Built Panl Results Explainer web app showing the list of available collection/fieldset configurations

Select one of the available CaFUPs links, which will then present the configuration for that particular CaFUP.

The Panl Single Page Search

An example of a single page search interface which will allow you to see all facets, select them, and see the results in the results viewer.



Image: The In-Built Panl Single Page web app showing the list of available collections.

Select one of the available Collection links, which will then present all of the available facets in a single page search interface.

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

```

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

If this property is set to 'true' then the message will be more verbose, including a Java stack trace if there was a back end exception (i.e. an HTTP 500 status code). For 404 errors, the JSON response object will include a list of valid URLs.



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

Binding Solr collections to Panl URL paths

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

| | | | | |
|---|------------|--|--|--|
| <code>panl.collection. <solr_collection> = path/to/file/ <panl_collection_url> .panl.properties</code> | | | | |
| The property key prefix The Solr collection The file path The filename prefix and Panl URL path The extension | | | | |
| key prefix | collection | | | |

- 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 URL 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 `src/main/panl/mechanical-pencils/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 URL path of `/mechanical-pencils/*`, where the asterisk is the FieldSet to be returned.

~ ~ ~ * ~ ~ ~

The <panl_collection_url>.panl.properties Configuration File

The `<panl_collection_url>.panl.properties` file defines which Solr fields will be available on this URL path, how they are configured, sorted, and included in the facets and results. This file also defines the groupings of fields to be returned with the document results - these groupings are known as FieldSets.

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>



Note: The above link is for the most up-to-date file, including new feature developments. To view the file for the version that you are using, use the tag that matches the version that you are using and download and view the source code. See <https://github.com/synapticloop/panl/tags> for the version that you are using.



Notes: the `<panl_collection_url>` part of the file name is the name of the Panl server URL path to bind to and the configuration properties for this URL path. In this book we will be using the Solr collection of `mechanical-pencils`, bound to a URL 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 URL 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 URL path part, and is encoded by:

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

For the following URL 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 URL 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 URL 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/>

The Solr query is:

```
q=*&*&q.op=OR&facet.limit=100&fl=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=in_built_sharpener&facet.field=disassemble&fac
et.field=category&facet.field=lead_length&facet.field=in_built_eraser&facet.field=grip_sha
pe&facet.field=weight&facet=true&start=0
```

Note the 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/>

```
q=*&*&q.op=OR&facet.limit=100&fl=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=in_built_sharpener&facet.field=disassemble&fac
et.field=category&facet.field=lead_length&facet.field=in_built_eraser&facet.field=grip_sha
pe&facet.field=weight&facet=true&start=10
```

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 URL 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 URL 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 analysed fields (OR), or all of the analysed fields (AND).

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

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

Whilst this URL path below will search for the 'hexagonal' keyword and return results that are in ANY of the analysed search fields

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



Note: The query operand LPSE code does not have a URL 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 URL 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 URL path parts to return pencils manufactured by BIC with the following URL path.

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

Note that the above URL 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 URL path would grow very quickly.



Note: The passthrough LPSE code will `__NOT__` be included in the canonical URL 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 URL path value will 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 URL path.

This allows the URL 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 URL paths will return the **exact same** search results.

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

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

For the above URL paths, the generated canonical URL 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 URL path.

For example, the URL

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

Will return only one result - the Staedtler 925 pencil.

The URL 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 URL 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 URL path would be:

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

With the `panl.param.passthrough.canonical` property set to `true`, the canonical URL 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 URL. 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 URL.

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/Ma
ufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/bWGL/](http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma
ufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/bWGL/)

Shows 3 results and 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/Ma
ufactured+by+Koh-i-Noor+Company/Green/Cylindrical+Grip/120mm/bWGL/](http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma
ufactured+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 URL 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/bW WGL/>
- For Red:
<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/Ma nufactured+by+Koh-i-Noor+Company/Green/Red/Cylindrical+Grip/120mm/bW WGL/>
- 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/bWWGL/>

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

Should you wish the above scenario to be the case, then set this property 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 - By default this is set to `100`. If set to `-1`, then all facet values for all facets will be returned.

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. **Note:** When a Panl facet is defined as an OR facet, Panl will dynamically set the min count value.

solr.numrows.default

The number of documents to return for the search. By default this is 10, but may be configured to any positive integer. This default can be overridden through the interface by using the parameter and LPSE code value which allows users to dynamically select the number of results to see per page.

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 (i.e. the <uniqueKey /> XML element in the managed schema file) with the value of the Solr field where this highlighting occurs.

Field Definitions

Fields do not filter the search results, but are used to have them returned with each document. Additionally, field definitions allow Panl to be configured to use them as a sort order.

A Panl field is configured by setting a key within the properties file and will **ALWAYS** have the form of:

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

Where:

- **field** configures Panl to serve this as a Field
- **<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.

Properties Available for Fields

The following properties are available for any field that is defined

- `panl.field.<lpse_code>=<solr_field_name>`

MANDATORY property to indicate to Panl that the named Solr field is to be used as a field 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>=<nicer_name_for_the_field>`

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>=<truncated_solr_java_class_name>`

Optional property of the base Solr field type, this property is set automatically by the Panl configuration generator.

This property is not used by a Panl field, however it is a good idea to keep this property in case you wish to use it as a facet in a further iteration.



IMPORTANT: The `panl.type.<lpse_code>` property should **NEVER** change, **UNLESS** the underlying Solr managed schema has changed.

Facet 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 URL path part that is displayed to the user.

A Panl facet 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:

- **facet** configures Panl to serve this as a Facet
- **<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_url>.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>=<nicer_name_for_the_field>`

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>=<truncated_solr_java_class_name>`

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: The `panl.type.<lpse_code>` 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 URL (encoded) 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 URL 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 **OR** 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 facet 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 contain all documents that have this facet value.

Prefixes and suffixes are available for regular facets, with the following URL 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 URL path

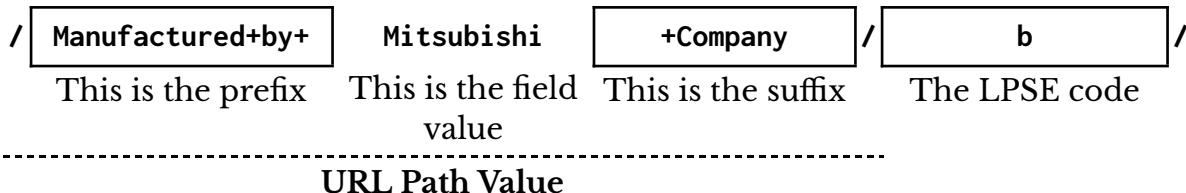
```
/Manufactured+by+Mitsubishi+Company/b/
```

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 URL

Without too many configuration options for a regular facet, the URL 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 URL 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" <
02 "multiValued"="false" />
03 panl.facet.D=disassemble
04 panl.name.D=Disassemble
05 panl.type.D=solr.BoolField
06 panl.bool.D.true=able to be disassembled
07 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 URL 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 URL 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) |
| [add] able to be disassembled (45) |
| [add] 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 URL 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 URL Without a Prefix or Suffix

In the case of a true value, the URL path would be:

1. The LPSE code is looked up to resolve to the `disassemble` Solr 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

URL Path Value

In the case of a false value, the URL 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

URL 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)

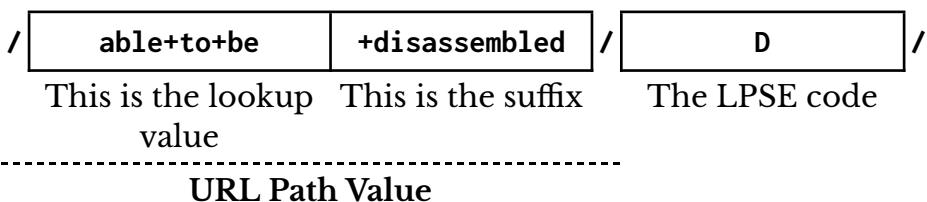
```
05 panl.bool.D.true=able to be disassembled
06 panl.bool.D.false=cannot be disassembled
```

Could be replaced with the following three lines, and would have an identical effect, however the decoding of the URL 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 URL 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 URL path value
3. The lookup value is checked and if found, and exactly matches the defined property, then this token is marked as valid
4. 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 as this field allows the user to select multiple facet values for a specific document, when the document has only one of them assigned..

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.

- `panl.or.always.<lpse_code>=<true_or_false>`

This configures Panl to always return the facet values for this specific facet. By default this is '`false`'. This will mean that by the user may be able to select a facet which will return no additional results if the combination of other selected facets would not return any results.

This may be useful in conjunction with the 'More Like This' functionality.

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.

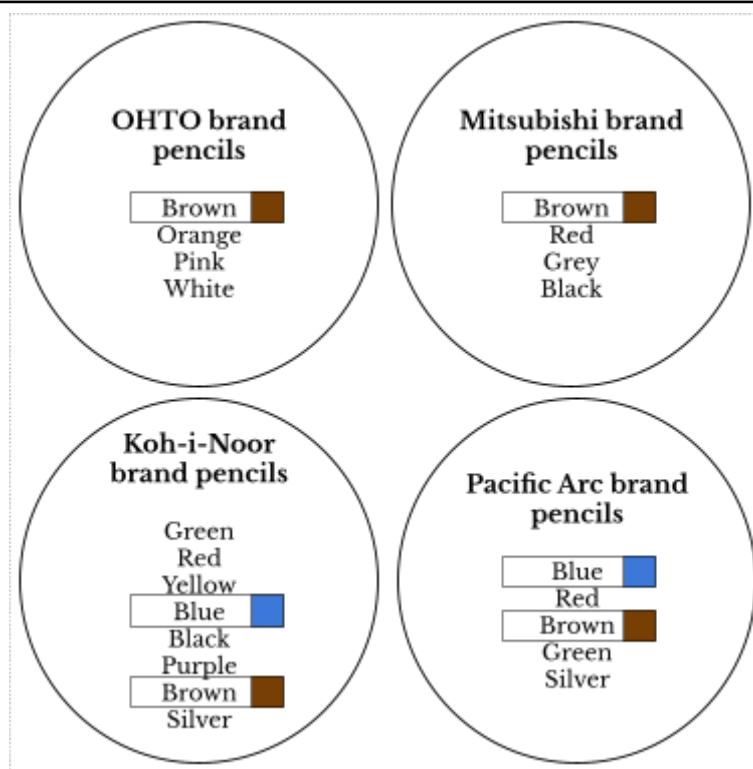


Image: A selection of four brands and their associated colours - with at least one model within the Brand that comes in a 'Brown' colour

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.



Tip: Multivalued Solr fields work best as Regular facets.

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" <br/>
     "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.

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 OR 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 Results Viewer 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 in the returned JSON Object, 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). | |
|--|--------------------|
| Brand (b) | Colours (W) |
| [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] Kaweco | [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).

| | |
|----------------------|--------------------|
| Brand (b) | Colours (W) |
| [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] Hightide Penco | [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).

| | |
|----------------------|--------------------|
| Brand (b) | Colours (W) |
| [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] Jiro Vya | [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). Or facets may also increase the selections for other facets.

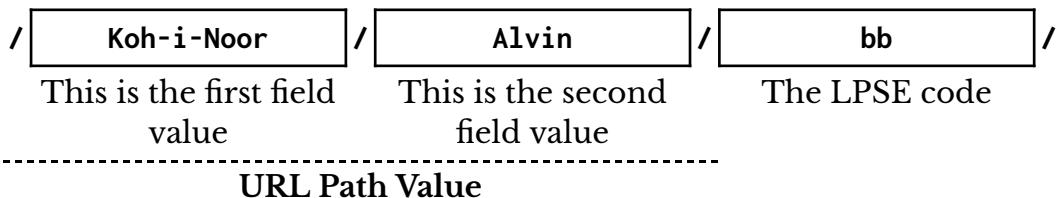
In effect, the query for Solr is requesting any result documents that have the Brand of 'Koh-i-Noor' OR 'Alvin'.

Decoding the URL

/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 URL 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: If a RANGE facet is configured, it will `ALWAYS` be returned in every query. Additionally, the Panl server `WILL` return the regular facet and values in the JSON response Object.

This allows you to choose whether to display the range facet or the regular facet, or both.

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" <
      "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

```

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>=<integer_or_decimal>`

MANDATORY IF `panl.range.<lpse_code>=true` The minimum value to set the

range slider to¹⁷ - this must be a number (either integer or decimal)

- `panl.range.max.<lpse_code>=<integer_or_decimal>`

MANDATORY IF `panl.range.<lpse_code>=true` The maximum value to set the range slider to¹⁸ - this must be a number (either integer or decimal)

- `panl.range.prefix.<lpse_code>=<text_to_prefix_with>`

Optional property to prefix the range with, this is distinct from the `panl.prefix.<lpse_code>` property and changes the way in which the URL is generated.

- `panl.range.infix.<lpse_code>=<text_infix_between_values>`

Optional property to place between the ranges, if set this will also affect the way in which the LPSE URL 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>=<text_to_suffix_with>`

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 URL is generated.

- `panl.range.min.value.<lpse_code>=<text_to_replace_minimum_value>`

This property sets a replacement String for the minimum value of the range,

¹⁷Admittedly 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.

¹⁸Once again, annoying to have to know the values.

which allows generating a friendlier URL. **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>=<text_to_replace_maximum_value>`

This property sets a replacement String for the maximum value of the range, which allows generating a friendlier URL. **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>=<true_or_false>`

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>=<true_or_false>`

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.

- `panl.range.suppress.<lpse_code>=<true_or_false>`

Set whether the individual range values should be suppressed. By default, Panl will return the available range facet, in the `available.range_facets` JSON array and also return the individual values in the `available.facets` JSON array. Setting this to '`true`' will not return the individual facets, only the range facets.

Suppressing Individual Facet Values

For the mechanical pencils data set, the weight facet is set to a range facet, and both the RANGE facet and individual facet values are returned. For the URL:

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

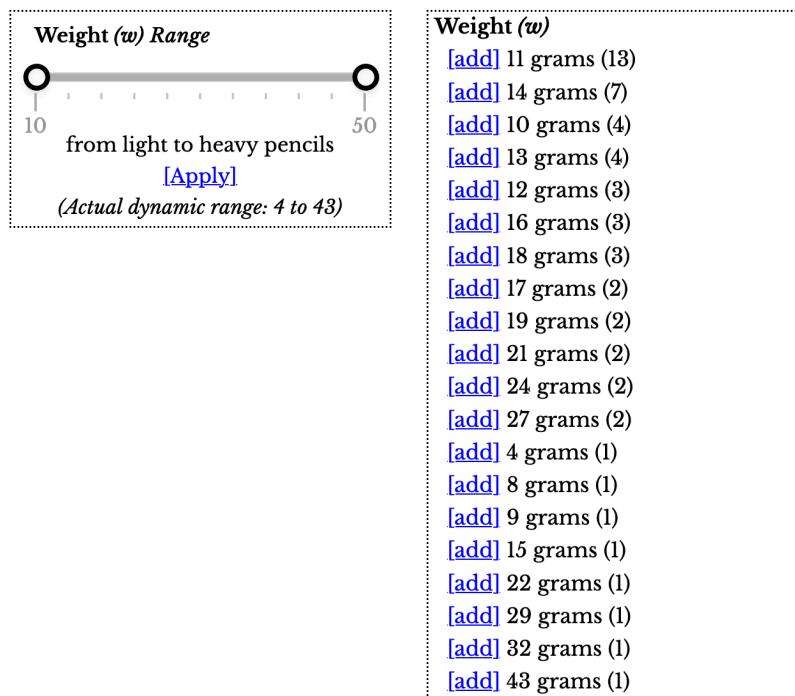


Image: On the left is the rendered RANGE facet interface, and on the right, the individual values for the facet. If the range suppress property is set to true, then the individual facets will not be returned.

Visualising the Properties

Below is a diagram of what results will be included depending on the properties that are configured.

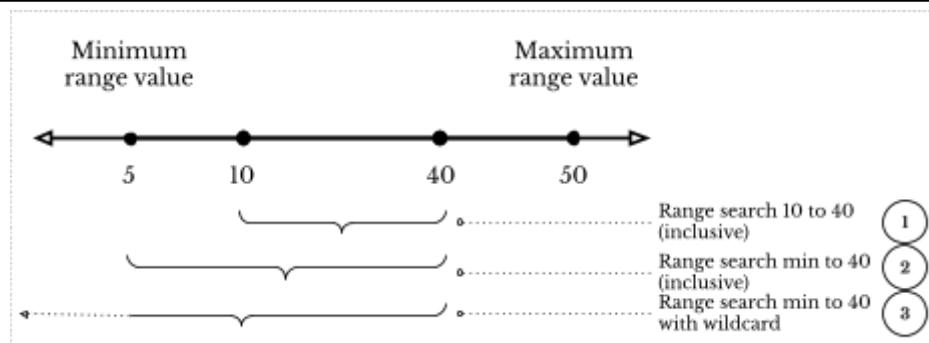


Image: How the properties affect the values that the Solr search will include

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 or equal to 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'.

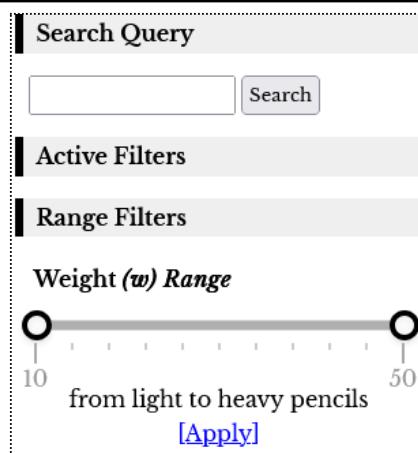


Image: The rendered RANGE facet

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 URL - 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 | URL Path Value | |
| ----- | | | | | | | |

Decoding the URL - 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 | URL Path Value | |
| ----- | | | | | | | |

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.

Because `solr.DatePointField` are always returned as DATE Range facets by Panl, new fields (namely, `year`, `month`, `day`, and `day_of_week`) were derived from the `solr_date` field for

users to facet on, these are included in the XML example below. 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 `pdate`, 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/pan1/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

Active Filters

Solr Date (S)
[\[remove\]](#) next 12 months

Range Filters

Solr Date (S) Date Range

next ▾ 12 months ▾

[Apply range: next 12 months](#)

Available Filters

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 \(5\)](#)

Search Results - Found 53 result(s) (exact)

q.op AND || OR

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.*

| | | |
|---|---|----------------------------|
| 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 | | |
| 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.

- `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.



IMPORTANT: You MUST set ALL of the above 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.

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 URL 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 URL

The URL decoding is straightforward. Panl will know from the LPSE code that this is a DATE Range facet and will then parse the URL 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 | |
| URL 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 facet definition, irrespective of the field type. These will be transparently added and removed from the URL path when Panl processes it. Prefixes and suffixes have no effect on Panl field definitions.



IMPORTANT: The decoded URL 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>=<your_infix_here>
```

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 will be ignored otherwise, example properties

```
panl.bool.<lpse_code>.true=able to be disassembled
```

If the boolean field value is true, then the above property will be used in the URL path part. Remember that a prefix and suffix may still be applicable on the boolean value, which will be prepended and/or appended to the above value.

```
panl.bool.<lpse_code>.false=cannot be disassembled
```

If the boolean field value is false, then the above property will be used in the URL path part. Remember that a prefix and suffix may still be applicable on the boolean value, which will be prepended and/or appended to the above value.



IMPORTANT: If you wish to have whitespace before 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, ensures that it is in the correct format for the field type and then passes 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).

| | |
|--|--|
| Genre (g) [add] Fantasy (11) [add] Young Adult (11) [add] Mystery (9) [add] Thriller (9) [add] Crime (8) [add] Detective (8) [add] Romance (8) [add] Children (3) [add] Dystopia (3) [add] Humour (3) [add] Drama (2) Authors (A-Z) (4) [add] C (11) [add] M (6) [add] D (3) [add] B (2) Decade Published (D) [add] 2000 (8) [add] 1990 (6) [add] 1980 (4) [add] 2010 (3) [add] 2020 (1) Book Length (L) [add] Novel (16) [add] Novella (6) | Genre (g) [add] Fantasy (6) [add] Young Adult (6) [add] Romance (4) [add] Crime (2) [add] Detective (2) [add] Dystopia (2) [add] Mystery (2) [add] Thriller (2) 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 2003 (1) [add] First published in 2006 (1) [add] First published in 2007 (1) [add] First published in 2009 (1) Authors (A-Z) (4) [add] C (4) [add] M (4) Book Length (L) [add] Novel (6) [add] 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 (3)
- [\[add\]](#) 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 (3)
- [\[add\]](#) 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 (Technically)

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

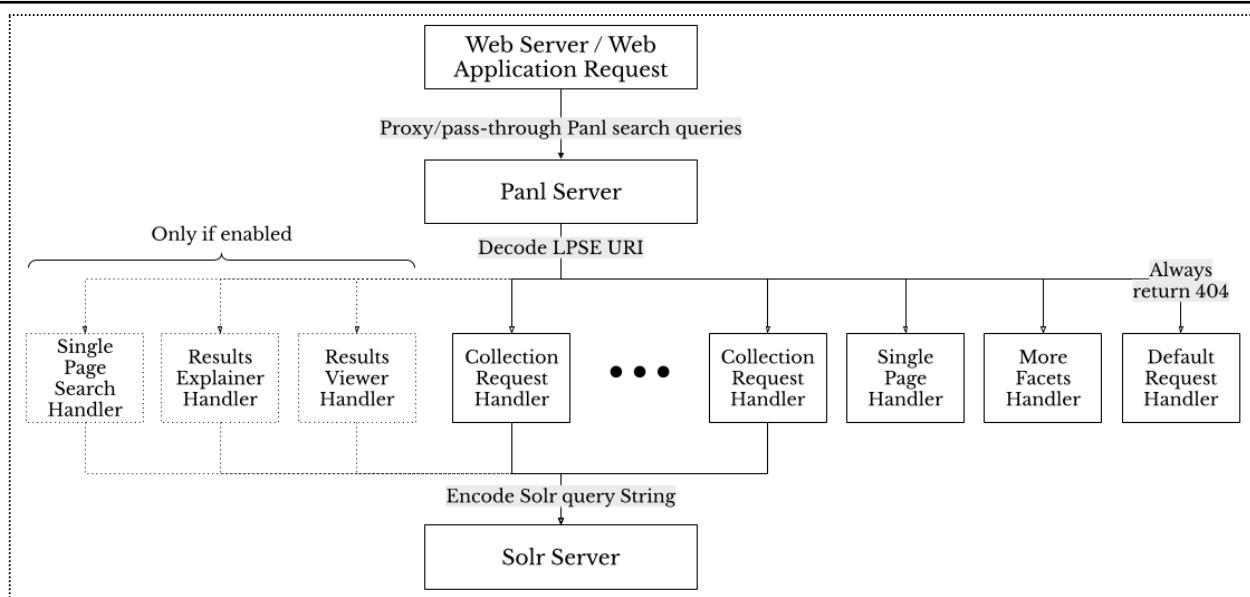


Image: The recommended approach to integrating the Panl server

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 URL 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.

Production Readiness

1. Turn off in-build Panl testing URLs,
Set the property `panl.results.testing.urls=false` in the `panl.properties` file
2. Turn off verbose 404 and 500 HTTP status responses,
`panl.status.404.verbose=false`
`panl.status.500.verbose=false`
3. Finally, check the logging requirements (see heading below).
Edit the `log4j2.xml` file to your requirements.

Logging

Panl utilises the log4j/slf4j framework for all its logging and does not log excessively. The parts of logging that Panl performs is:

- INFO level startup logging which logs informational messages about the Panl configuration and URL bindings at startup,
- DEBUG level logging for inbound Panl token decoding and validation,
- DEBUG level logging for the query string that is passed through to the Solr server, and
- Error level debugging for HTTP status 500 Internal Server Error (with stacktraces).

The most up-to-date logging file can always be viewed on the GitHub repository:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/log4j2.xml>

A copy of the file is included below:

```

01 <?xml version="1.0" encoding="UTF-8"?>
02 <!--
03   ~ This is the default logging configuration file for the Panl server which
04     ~ does not do too much.
05   ~

```

```

06 ~ Panl does not log too many things, just startup and the incoming requests
07 ~ and validation
08 ~
09 -->
10
11<Configuration status="WARN">
12 <!--
13 ~ This is the root logger which just logs the INFO messages to the
14 ~ console, which are the generic startup messages.
15 -->
16<Loggers>
17 <Root level="info">
18   <AppenderRef ref="console"/>
19 </Root>
20
21 <!--
22 ~ The Panl request handlers log on the debug level for the incoming panl
23 ~ token validation and the transposed Solr query that is executed. To
24 ~ remove this logging, comment out the following Logger, or just delete
25 ~ it.
26 -->
27 <Logger name="com.synapticloop.panl.server.handler" level="debug"
28   additivity="false">
29   <Appender-ref ref="console" />
30 </Logger>
31 </Loggers>
32
33<Appenders>
34   <Console name="console" target="SYSTEM_OUT">
35     <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %-5level %logger{2} - %msg%n"/>
36   </Console>
37 </Appenders>
38</Configuration>
```

Setting the Logging Levels

The logging configuration file is located in the release package with the name `PANL_INSTALL_DIRECTORY/lib/log4j2.xml`.

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 URL 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 URLs

Passthrough URLs are a little more able to withstand Panl configuration changes as they are not linked to any Solr field value.

Passthrough URLs 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 URLs, you may wish to configure a separate file and associated CaFUP just for these values, and be able to tweak other configured CaFUPs.

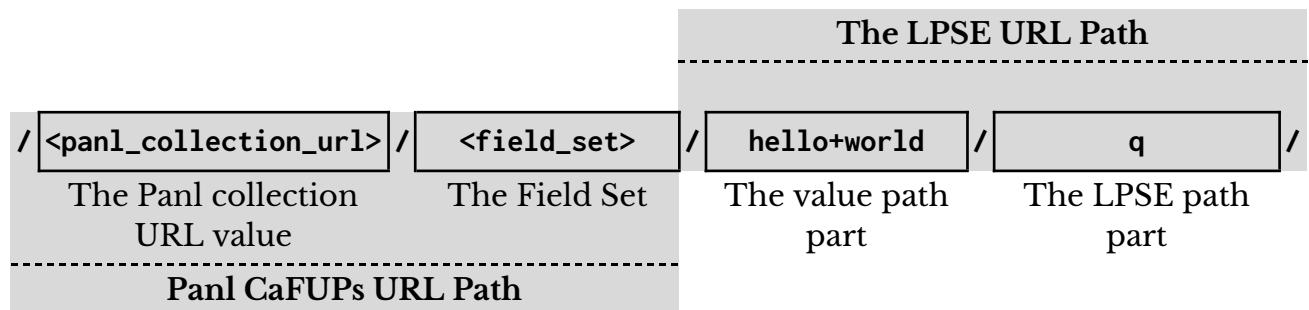
Handling Errors

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 URL Paths Explained

How to decode the URL parameters for the LPSE encoded URL. The nomenclature for the URL path is detailed below



For the information in this chapter, only the 'LPSE URL Path' will be discussed.

Each of the LPSE URL Path components will have the following information:

Property The property that configures this in the `<panl_collection_url>.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 URL value path part
 - **Never** - will never appear in the URL valuepath part
 - **Conditional** - Depending on configuration options or values, may appear in the URL 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 URL 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 URL 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 LPSE URL path.

Canonical Value Path Part **Never**

Canonical LPSE Path part **Sometimes** - this will only appear in the LPSE URL path 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_url>.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 search 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 LPSE URL path if it does not match the default number of rows

property.

LPSE Path Part **Sometimes** - this will only appear in the LPSE URL 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 URL 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 URL 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 LPSE URL Path part if it is not the first page.

LPSE Path Part **Sometimes** - this will only appear in the LPSE URL 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 URL 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 URL 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.

| | |
|--|--|
| Property | <code>panl.param.passthrough</code> |
| Value Path Part | Always |
| LPSE Path Part | Always |
| Canonical Value Path Part | Sometimes - 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> . |
| Canonical LPSE Path part | Sometimes - 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> . |
| Value Format | <code><string>/<lpse_code>/</code> |
| Default LPSE Code | <code>z</code> |
| Notes | 1. This value will never be passed through to the Solr query |
| See also panl.param.passthrough.canonical | |

With the default for the property `panl.param.passthrough.canonical`, i.e.

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

The LPSE URL path would be

```
/some+sort+of+arbitrary+string/z/
```

And the canonical URL path would be

```
/page-1/10-per-page/pn/
```

Note: in the above URL 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 URL path would be

```
/some+sort+of+arbitrary+string/z/
```

And the canonical URL 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.

| | |
|-----------------|-----------------|
| Property | panl.param.sort |
|-----------------|-----------------|

| | |
|------------------------|-------|
| Value Path Part | Never |
|------------------------|-------|

| | |
|-----------------------|--|
| LPSE Path Part | Sometimes - this will only appear if it does not match the default sort order (relevance descending). |
|-----------------------|--|

| | |
|----------------------------------|-------|
| Canonical Value Path Part | Never |
|----------------------------------|-------|

| | |
|---------------------------------|--|
| Canonical LPSE Path part | Sometimes - this will only appear if it does not match the default sort order (relevance descending). |
|---------------------------------|--|

| | |
|---------------------|---|
| Value Format | /<lpse_code><sort_field_lpse_code>(+-)/ |
|---------------------|---|

| | |
|--------------------------|---|
| Default LPSE Code | o |
|--------------------------|---|

| | |
|--------------|--------------------------------|
| Notes | If the value of this parameter |
|--------------|--------------------------------|

| | |
|-----------------|--|
| See also | |
|-----------------|--|

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 URL path is simply

/

And the canonical LPSE URL path for the sort order is the same

/

However, if there is a sort order set (or many sort orders set), then the LPSE URL path and the canonical LPSE URL paths will be generated.

Ordering by a Solr field.

Search ordering for fields is controlled by the `panl.order.fields` property in the `<panl_collection_url>.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.

| | |
|----------------------------------|---|
| Property | <code>panl.facet.<lpse_code></code> |
| Value Path Part | Always |
| LPSE Path Part | Always |
| Canonical Value Path Part | Always |
| Canonical LPSE Path part | Always |
| Value Format | <code>/<prefix><string><suffix>/<lpse_code>/</code> , or <code>/<range_prefix><string><range_suffix>/<lpse_code>/</code> , or <code>/<min_value_replacement><string><max_value_replacement>/<lpse_code>/</code> |
| Default LPSE Code | N/A |
| Notes | |
| See also | |

For any facet, the URL path would be

```
/the+facet+value+with+prefix+or+suffix/?/
```

Where the `?` is the LPSE code, these are always passed through to the canonical URL path, as in

```
/the+facet+value+with+prefix+or+suffix/?/
```

For any URL path, the prefixes, suffixes and, in the case of a RANGE facet, the infix will be included in the URL. Multiple LPSE codes can be present in any order without affecting the returned results. For example, the URL 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 URL

```
/Purple/Red/Entry+Level/page-1/10-per-page/WWCpn/
```

Note: The Panl generated URL path is different from the canonical URL path, the canonical URL 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 URL 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 URL remains as

```
/Purple/Red/Entry+Level/page-1/10-per-page/WWCpn/
```

Explaining A More Complex Example

For the canonical URL path:

```
/Purple/Red/Entry+Level/from+light+to+16+grams/page-1/5-per-page/WWCw-wsb-pn/
```

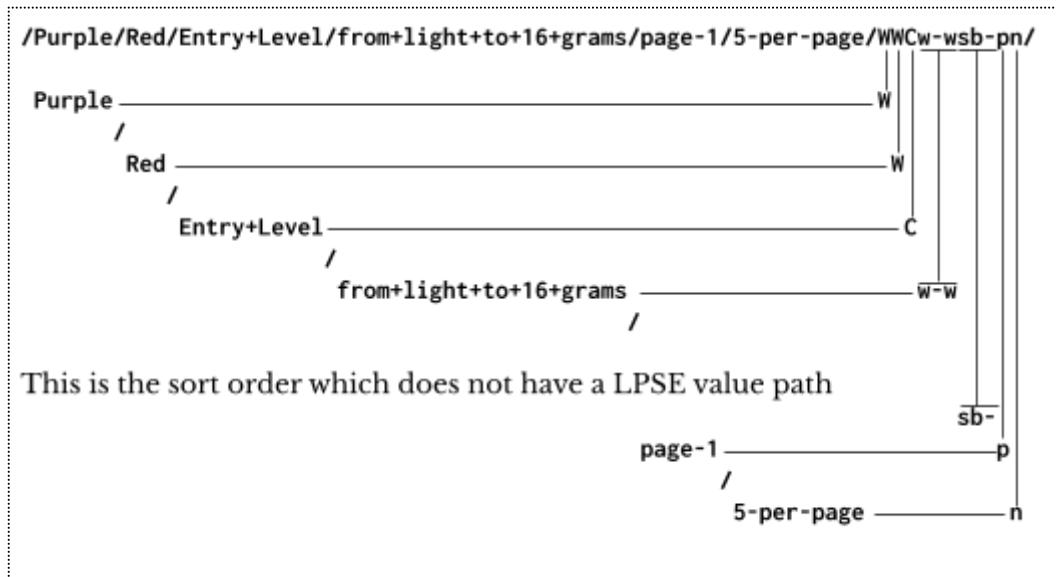


Image: Mapping of LPSE codes to LPSE values

Decoding the above:

- 'Purple' is the Colour facet (LPSE code 'W')
- 'Red' is another Colour facet (LPSE code 'W')
- 'Entry Level' is the Category of pencil (LPSE code 'C')
- 'Weight' is from the minimum value to to 16 grams (inclusive) (LPSE code 'W-W')
- Sort order is by brand descending (LPSE code 'sb-' - note that there is no LPSE value path part)
- Page 1 (LPSE code 'p')
- 5 results per page (LPSE code 'n')

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 URL 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 URL 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 URL path value '5-per-page' using parsed value of '5'.
```

~ ~ ~ * ~ ~ ~

Search 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.

URL Bindings

The Panl JSON responses are bound to the Panl URL path in the form of

`http://localhost:8181/<panl_collection>/<panl_FieldSet>`

For example, the `mechanical-pencils` Panl collection with the `default` FieldSet is bound to the Panl URL

<http://localhost:8181/mechanical-pencils/default/>

Which will return the JSON response detailed below.

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):



Note: 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 URL 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 URL 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>
```

| | | | |
|----------------------------------|---|-------------------------------|-------------------------|
| <code><a href="</code> | <code>/Clutch/page-2/msb+po+/"></code> | <code>5-per-page</code> | <code></code> |
| <code>remove_uri</code> JSON key | URL decoded | <code>encoded</code> JSON key | |



Notes: To render the text of the anchor tag, the `encoded` value will need to be URL decoded. In the `panl-viewer.js` file, additional processing is done thusly:

```
return(decodeURIComponent(text).replace(/\+/g, " ").replace(/\%2B/g, "+"));
```

The above line will replace the `+` character with spaces (which is not done with the javascript function `decodeURIComponent()`) 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 URL.

```

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 URL 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_booleans_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 URL path to use to invert the currently selected value, if true, this is the false URL path, if false, this is the true URL 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:

```
<a href="/cannot+be+disassembled/D"> cannot be disassembled </a>
    inverse_uri JSON key           URL decoded
                                inverse_encoded 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 URL 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 3:

Indicates that this is a DATE facet allowing a range of dates to be selected

Line 4:

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 do

render this facet in the Active Filters, and just use the Range Filters section .

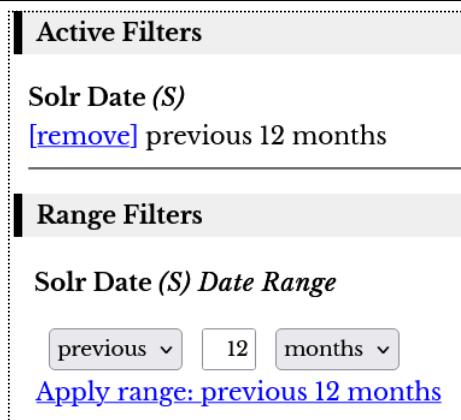


Image: The In-Build Panl Results Viewer web app showing the DATE filter in both the Active Filters and Range Filters sections.

The "active.facet" JSON Array (OR facet object)

An OR facet contains the information to display the value and the removal URL 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 URL. 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"
}

```

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.

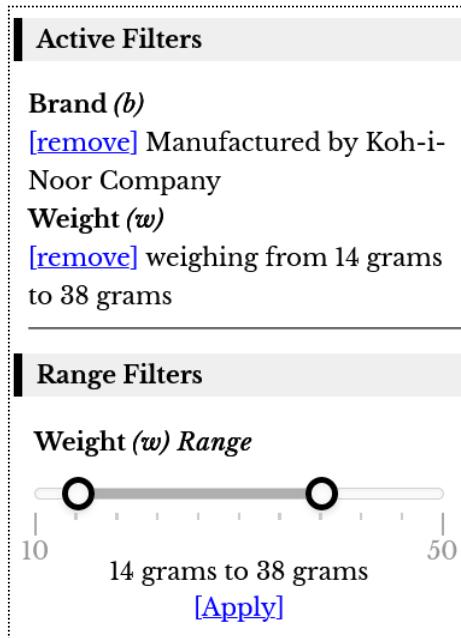


Image: The In-Build Panl Results Viewer web app showing the RANGE filter in both the Active Filters and Range Filters sections.

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/brandname/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+/>

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 URL. 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/>

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/"
09 }
```

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 tags's `href` attribute and the `encoded` value for the anchor text.

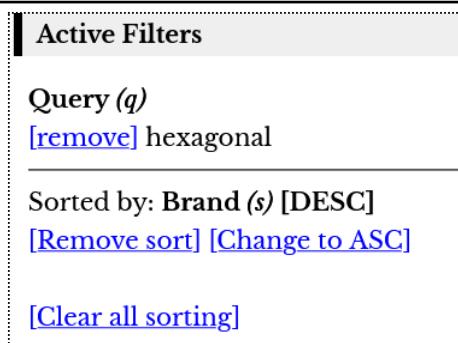


Image: The In-Build Panl Results Viewer web app showing a sort order of Brand descending, with the remove link, the change to ascending link, and the clear all sorting link.



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.

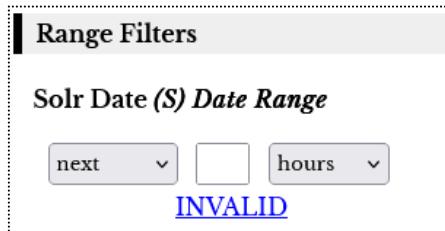


Image: The In-Build Panl Results Viewer web app showing the implementation of the DATE Range facet.

Of course, you may implement your interface in any way that you desire.

To generate the URL,

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 URL will always be a replacement URL not an additive URL - i.e. if you already have a DATE Range facet selected, the generated URL 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   "facet_limit": 100,
04   "is_or_facet": false,

```

```

05 "is_range_facet":false,
06 "is_multivalue": false,
07 "name":"Brand",
08 "panl_code":"b",
09 "uris":{
10   "before":"/",
11   "after":"/b/"
12 },
13 "values":[
14 {
15   "count":5,
16   "value":"Faber-Castell",
17   "encoded":"Manufactured+by+Faber-Castell+Company"
18 },
19 ...
20 ]
21 }

```

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 Solr 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 10**)
2. Append the `values[i].encoded` value (**Line 17**)
3. Append the `uris.after` value (**Line 11**)

To render the value of the anchor tag use the encoded value and URL decode it.

From version 9.2.0 there is an additional JSON key named `facet_limit` which is the integer maximum number of facets values that will be returned with the request. This value is set in the `<panl_collection_url>.panl.properties` file with the key `solr.facet.limit`.

If the size of the values array is greater than this number, then you may request additional facet values from the query by requesting more facets from the URL
`/panl-more-facets/<lpse_code>/<amount>`

Where `<lpse_code>` is the Panl LPSE code for the facet you are requesting and `<amount>` is the new facet limit to request.

When more facets are requested the key `facet_request` will be populated with the number of facets that were requested.

If, at any time the `facet_limit` is greater than the number of values then there is no need to request more facets.

If the `facet_request` is -1 then all facets have been returned.



Note: The `facet_limit` key is only available on Regular and OR facets.

The "available.range_facets" JSON Object

Range facts allow a variety of URL 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   "dynamic_max": 43,

```

```

11 "dynamic_min": 4
12 "min": "10",
13 "max": "50",
14 "prefix": "",
15 "range_min_value": "from+light",
16 "facet_name": "weight",
17 "name": "Weight",
18 "panl_code": "w",
19 "suffix": "+grams",
20 "range_max_value": "heavy+pencils"
21 }

```

Integration and Implementation

RANGE facets have a variety of configuration options, and depending on the requirements of the URL path generation will depend on what properties need to be set.



IMPORTANT: Range facets will **ALWAYS** be returned in the JSON response object.

However the UI component is generated, there are two options for generating the minimum and maximum value.

1. Use the hard-coded minimum and maximum values that are configured in the `<panl_collection_url>.panl.properties` file, or
2. Use the dynamic maximum and minimum that are passed through in the facet.

Using the hard coded values will enable you to also use the minimum and maximum value replacement, whilst using the dynamic values will not.

If you are using the hard-coded values, it is recommended that the minimum and maximum wildcard properties are set to true.

Example of hard-coded values

The Panl Results Viewer uses this method, setting the range slider to the values of a minimum of 5 and a maximum of 50. **Note** that the actual minimum value is 4, however

this is included in the minimum value as the minimum wildcard range search is enabled.

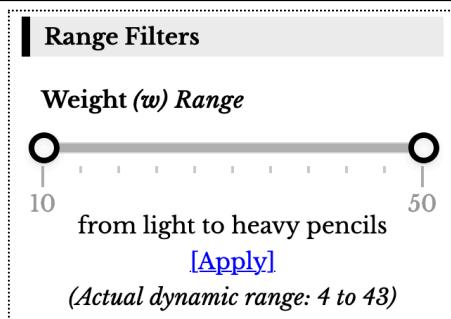


Image: The RANGE facet with hard-coded values with no selected range.

In the above image, no range was selected, in the below image a range of 17 to 40 grams was selected. Note that there are only values between 17 and 32 in the range, however the UI reflects the values chosen.

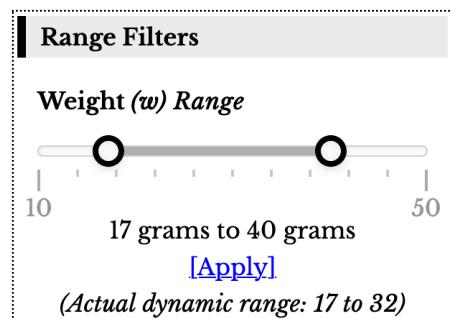


Image: The RANGE facet with hard-coded values and a selected range of 17 to 40 grams

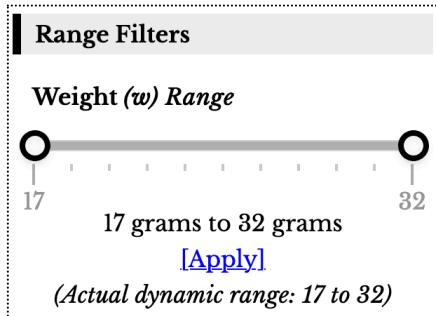


Image: The RANGE facet with dynamic range values and a selected range of 17 to 32 grams

In the above image, the same selection is made, and the range values have been updated to the maximum and minimum available (this UI example is not included in the Panl Results Viewer).

A dynamic range will allow a user to further refine the range to get a smaller subset of the range. The above example does not allow a user to expand the range, however you are free to implement the range facet in whichever way that you choose.

Defining a Range Without an infix

Depending on the properties that are set, all of the following URL 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/

Defining a Range With an Infix

Depending on the properties that are set, all of the following URL 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

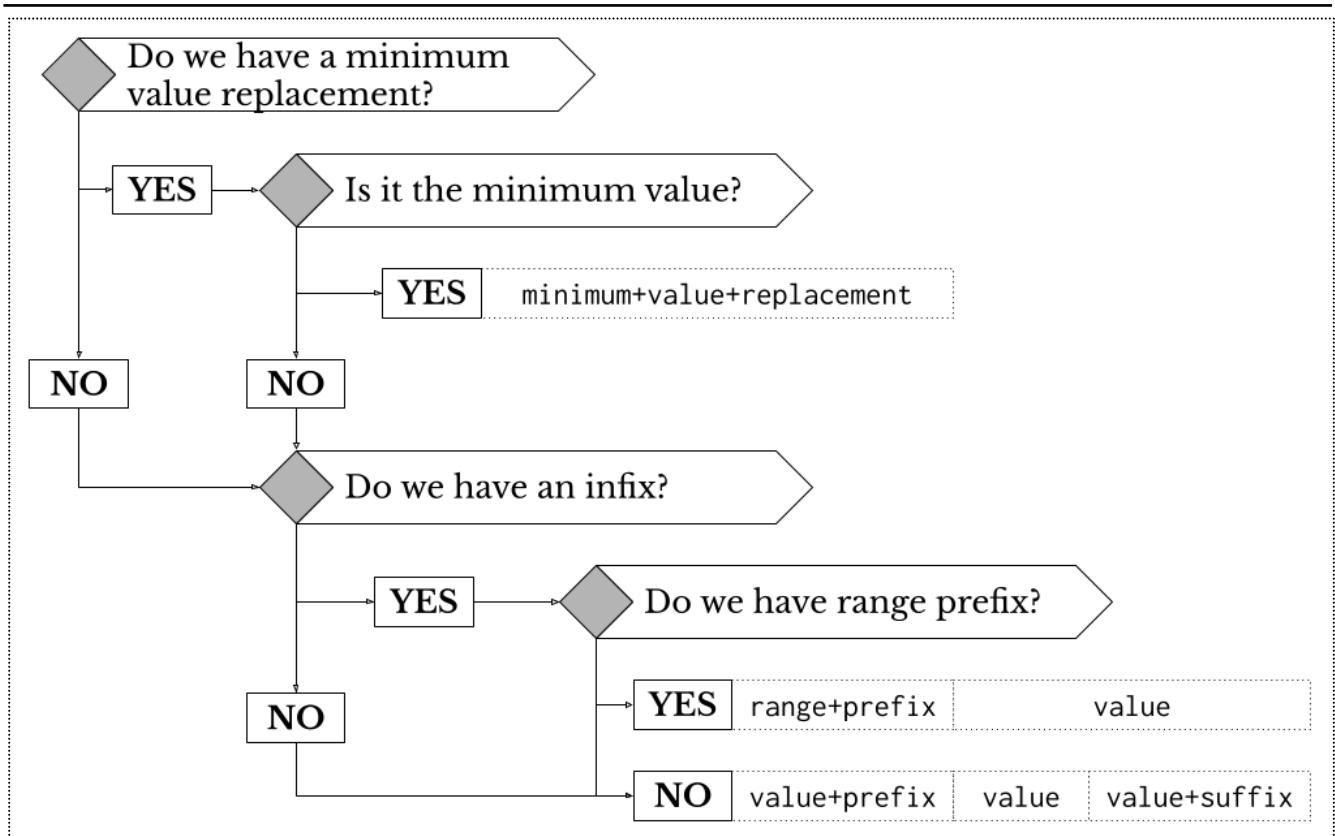
~ else ~

- If it a value prefix or value suffix exists, the value prefix + value + value suffix

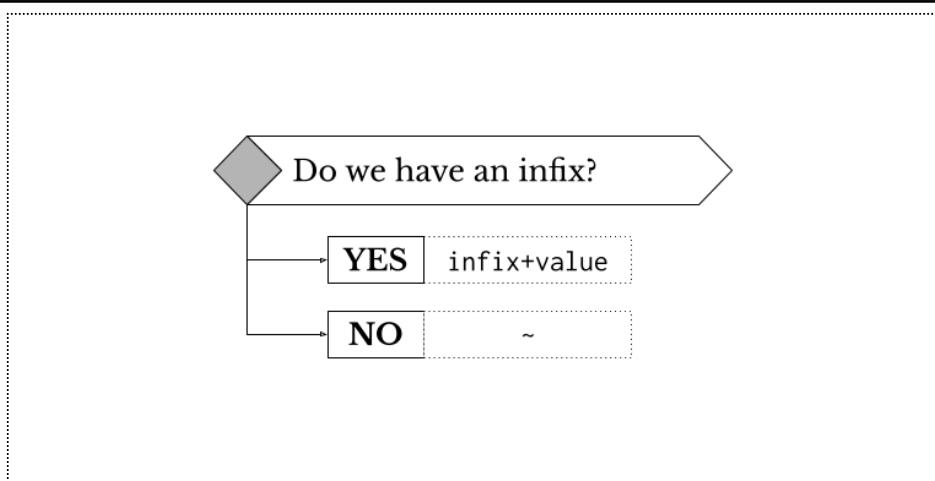
~ otherwise ~

- Just the value

The flow chart below shows the logic for how the complete URL path for how a range facet is determined and generated



Flowchart: Logic for generation of the URL path values for range prefixes



Flowchart: Logic for generation of the URL path values for range prefixes

The order of precedence without an infix set

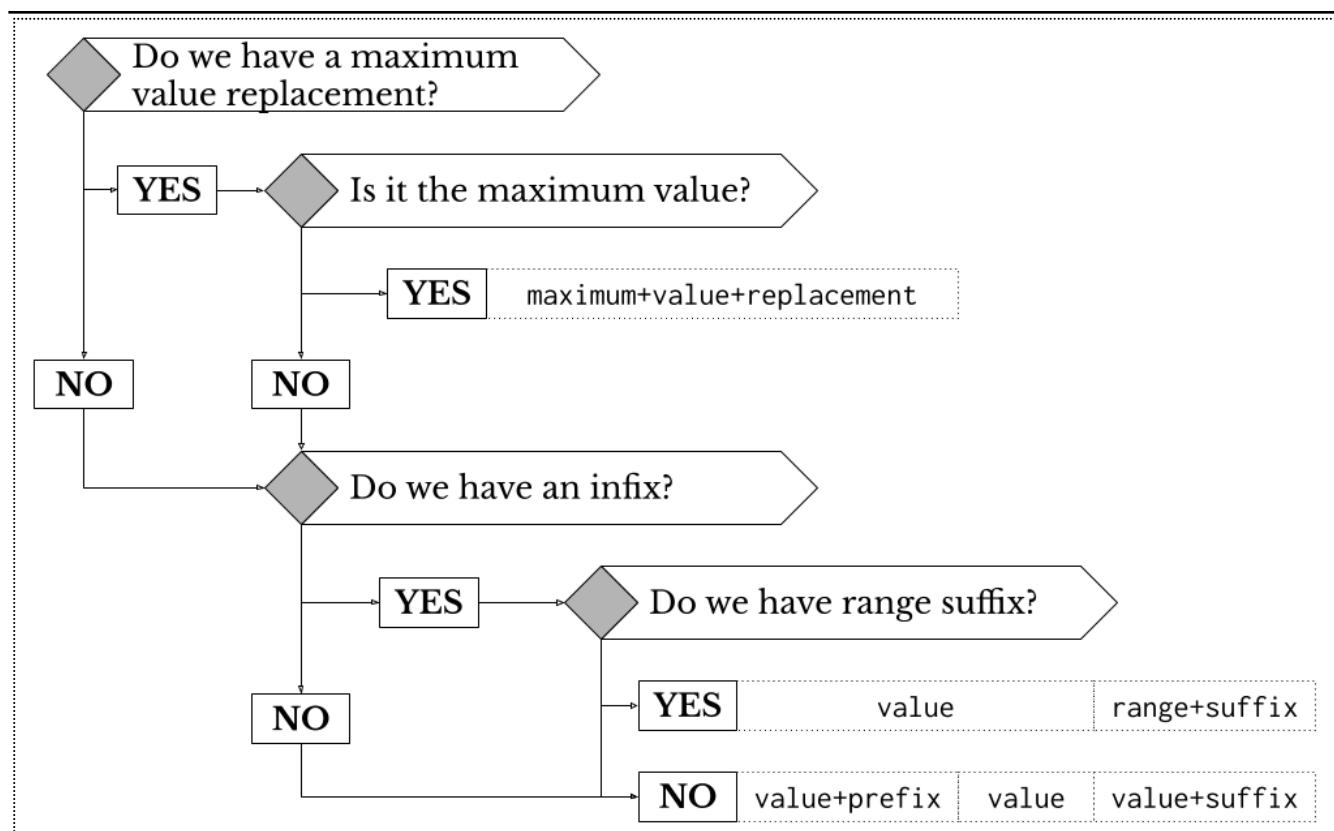
The order of precedence for the text without an infix set is:

- 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 URL path for how a range facet is determined and generated



Flowchart: Logic for generation of the URL path values for range suffixes

The "pagination" JSON Object

The following URL will return the pagination JSON object listed below.

<http://localhost:8181/panl-results-viewer/mechanical-pencils/brandandname/page-2/5-per-page/pn/>

The default query above returns all results (55 of them) with 5 results per page and is on page number 2:

```

01 {
02   "num_pages": 11,
03   "num_results_exact": true,
04   "page_uris": {
05     "next": "/page-3/5-per-page/pn/",
06     "previous": "/page-1/5-per-page/pn/",
07     "before": "/page-",
08     "after": "/5-per-page/pn/"
09   },
10   "page_num": 2,
11   "num_results": 55,
12   "num_per_page": 5,
13   "num_per_page_uris": {
14     "before": "/",
15     "after": "-per-page/n/"
16   }
17 }
```

Integration and Implementation

Unlike other returned JSON objects, the `pagination` object does not contain the `uris` key, instead it contains two new keys:

- `page_uris` - used to generate the pagination
- `num_per_page_uris` - used to generate the number of results to be shown per page

The 'page_uris' JSON Object

For convenience the `page_uris` JSON Object contains ready made links for the next and previous page. If this is the last page of the results, the `next` key will not be present, if this is the first page of the results, the `previous` key will not be present.

To render the full suite of pagination links to the pages of results, iterate from `1` to `num_pages` (inclusive) and

1. Start with the `page_uris.before` value (**Line 7**)
2. Append the integer page number value
3. Append the `page_uris.after` value (**Line 8**)

The 'num_per_page_uris' JSON Object

The `num_per_page_uris` will **ALWAYS** reset the page number back to the first page. To render the number per page link, use any positive integer value and

1. Start with the `num_per_page_uris.before` value (**Line 7**)
2. Append the integer page number value
3. Append the `num_per_page_uris.after` value (**Line 8**)

The "sorting" JSON Object

The following URL will return the sorting JSON object listed below.

<http://localhost:8181/paml-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`),
- Sorting by `name (N)` ascending `(+)`, and
- A search query of `hexagonal`

```

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 URL 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 URL path for ascending (**Line 7**)
- `set_uri_desc` is the URL 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 URL path for sorting by the current sort order, then by brand ascending (**Line 9**)
- `add_uri_desc` is the URL path for sorting by the current sort order, then by brand descending (**Line 10**)

Integration and Implementation

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 (1). Then sort by Brand: ASC DESC (2).
- Facets:** Name (name) 5218, Brand (brand) Koh-i-Noor.
- Logs:** Solr: query time 62ms. Panl: parse request 0ms, build request 54ms, send and receive request 177ms, parse response 1ms. Total time 234ms.
- Buttons:** « PREV NEXT » and Show 3 5 10 per page.
- Sorting Summary:** Sorted by: Name (s) [ASC] (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 in the Panl Results Viewer web app 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 URL 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`

1

| | | |
|--|--|---------------|
| Search Query | Search Results - Found 55 result(s) (exact) | q.op AND C |
| <input type="text"/> Search | Page 1 of 6 Showing 10 results per page This page has 10 result(s) | |
| Active Filters | Sort by Brand: ASC DESC Name: ASC DESC Weight: ASC DESC | |
| Range Filters | « PREV NEXT » Show 3 5 10 per page | |
| Search Query | Search Results - Found 55 result(s) (exact) | q.op AND C |
| <input type="text"/> Search | Page 1 of 6 Showing 10 results per page This page has 10 result(s) | |
| Active Filters | Sort by Brand: ASC DESC Name: ASC DESC Weight: ASC DESC | |
| Sorted by: Weight (s) [DESC] [Remove sort] [Change to ASC] | Then sort by Brand: ASC DESC Name: ASC DESC | |
| [Clear all sorting] | « PREV NEXT » Show 3 5 10 per page | |
| <i>Solr: query time 41ms. Panl: parse request 0ms, build request 79ms, send and receive request 156ms, parse response 1ms. Total time 236ms.</i> | | |
| Search Query | Search Results - Found 55 result(s) (exact) | q.op AND C |
| <input type="text"/> Search | Page 1 of 6 Showing 10 results per page This page has 10 result(s) | |
| Active Filters | Sort by Brand: ASC DESC Name: ASC DESC Weight: ASC DESC | |
| Sorted by: Weight (s) [DESC] [Remove sort] [Change to ASC] | Then sort by Name: ASC DESC | |
| Sorted by: Brand (s) [ASC] [Remove sort] [Change to DESC] | « PREV NEXT » Show 3 5 10 per page | |
| [Clear all sorting] | <i>Solr: query time 39ms. Panl: parse request 0ms, build request 30ms, send and receive request 77ms, parse response 0ms. Total time 108ms.</i> | |
| Name (name) Classic Revolve | Search Results - Found 55 result(s) (exact) | |
| Search Query | q.op AND C | 4 |
| <input type="text"/> Search | Page 1 of 6 Showing 10 results per page This page has 10 result(s) | |
| Active Filters | Sort by Brand: ASC DESC Name: ASC DESC Weight: ASC DESC | |
| Sorted by: Weight (s) [DESC] [Remove sort] [Change to ASC] | « PREV NEXT » Show 3 5 10 per page | |
| Sorted by: Brand (s) [ASC] [Remove sort] [Change to DESC] | <i>Solr: query time 53ms. Panl: parse request 0ms, build request 48ms, send and receive request 114ms, parse response 0ms. Total time 163ms.</i> | |
| Sorted by: Name (s) [DESC] [Remove sort] [Change to ASC] | Name (name) Classic Revolve | |
| [Clear all sorting] | Brand (brand) YStudio | |

2

3

4

Image: The In-Built Panl Results Viewer web app 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

~ ~ ~ * ~ ~ ~

Single Search Page Integration And The Panl Response Object

The Panl server also binds to a URL which will return the LPSE configuration as a JSON response to be used to generate a single search page - i.e. one that contains all of the facets that can be placed on a single page for users to select all values.



Tip: The single search page is useful where the number of facets are small and, irrespective of the combinations of facets, will always return search results. For some single search pages, a combination of selected facets may return no results, which should be avoided. Of course, a separate CaFUP can be configured to only provide a small subset of facets and then used for the single search page.

How you choose to implement the Single Page Search interface is up to you, as a starting point you can view the in-built implementation for the latest version on github:

The HTML index page:

<https://github.com/synapticloop/panl/blob/main/src/main/resources/webapp/single-page-search/index.html>

The JavaScript implementation

<https://github.com/synapticloop/panl/blob/main/src/main/resources/webapp/static/panl-single-page-search.js>

The key information that is required, is the LPSE order, and the LPSE facets that exist.

URL Bindings

The Panl JSON responses are bound to the Panl URL path in the form of

`http://localhost:8181/panl-single-page/<panl_collection>/`

For example, the `mechanical-pencils` Panl collection is bound to

<http://localhost:8181/panl-single-page/mechanical-pencils/>

Which will return the JSON response detailed below.

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.

Success Responses

Unlike the Panl Results Viewer and Explainer, this is a separate response with no Solr JSON response included.

The JSON response object has the following form:

```

01 {
02   "error": "false",
03   "panl": {
04     "lpse_lookup": { ... },
05     "lpse_order": [ ... ]
06   }
07 }
```

The Panl response only has two keys, namely a boolean valued key of `error`, and a JSON object keyed on `panl` to the root object.



Note: The `error` key will **ALWAYS** be `false` for a successful return of the Panl Response object.

Within the `panl` JSON Object key, there are two keys, `lpse_lookup` and `lpse_order` which are detailed below.

The "lpse_lookup" JSON Object

This JSON object is a simple lookup keyed on the LPSE code, with the value being the index of the LPSE path. All valid facets are contained within the lookup including LPSE parameters pagination, sorting, number of results per page and query operand.

```

01 {
02   "9": 12,
03   "b": 1,
04   "C": 6,
05   "D": 8,
06   "G": 5,
07   "h": 9,
08   "I": 10,
09   "L": 7,
10   "m": 3,
11   "N": 2,
12   "n": 16,
13   "o": 17,
14   "p": 15,
15   "q": 18,
16   "s": 14,
17   "W": 4,
18   "w": 13,
19   "z": 0,
20   "Z": 11
21 }
```

This lookup is used to build the correct ordering for the LPSE URL path, being able to use the facet LPSE code as a lookup for the index of the URL.

Note: The pagination, number of results, query operand and sorting are not implemented in the in-built web app.

The "lpse_order" JSON Array

This array contains the returned Solr facets in the LPSE order. The array element will be null if the facet is not available. The Objects are identical to the `panl.available` facet object, containing identical keys and values to an empty search. To implement the interface for any of these facets, use the same logic as detailed in the [Search Integration And The Panl Response Object](#) section.

Generating the LPSE URL path

Each of the facets is rendered in order of the LPSE path. On change of any of the values, the `panl_code` can be used as the lookup key on the `panl.lpse_order` object. This will return the order in the LPSE URL path that it should be placed.

~ ~ ~ * ~ ~ ~

More Facets Integration And The Panl Response Object

Where the number of available facets in the Solr search index is greater than the number of facets that Panl is configured to return (this is configured with the property `solr.facet.limit` in the `<panl_collection_url>.panl.properties` file), this handler allows you to retrieve an arbitrary number of facet values.

URL bindings

The Panl JSON responses are bound to the Panl URL path in the form of

```
http://localhost:8181/panl-more-facets/<panl_collection>/<fieldset>/<lpse_path>/<lpse_code
s>/?code=<lpse_code>&limit=<limit>
```

For example, the `mechanical-pencils` Panl collection is bound to

<http://localhost:8181/panl-more-facets/mechanical-pencils/>

It requires additional URL information and query parameters to work

Building the Request URL

To request more facet values for a specific LPSE code, the url would look like the following - the initial request:

<http://localhost:8181/panl-results-viewer/mechanical-pencils-more/brandname/Clutch/m/>

Is searching for mechanical pencils which have a 'Clutch' mechanism. The limit for the facets for this CaFUP configuration is set to 10 - i.e. `solr.facet.limit=4`. The image below shows the returned facets with the [See all...](#) link appearing at the bottom of the facets.



Note: This above request URL is for the CaFUP binding in the `sample/panl/mechanical-pencils/mechanical-pencils-more.panl.properties` file which is not included in the default mechanical pencils `panl.properties` file and will have to be enabled.

| Brand (b) |
|---|
| [add] Manufactured by Koh-i-Noor Company (10) |
| [add] Manufactured by Caran d'Ache Company (4) |
| [add] Manufactured by Alvin Company (3) |
| [add] Manufactured by Faber-Castell Company (3) |
| [add] Manufactured by Pacific Arc Company (3) |
| [add] Manufactured by Kuelox Company (2) |
| [add] Manufactured by Rotring Company (2) |
| [add] Manufactured by BIC Company (1) |
| [add] Manufactured by Mitsubishi Company (1) |
| [add] Manufactured by Staedtler Company (1) |
| See all... |

Image: The 'brand' facet with a limited number of facets and the 'See all...' link

Upon clicking on the [See all...](#) link a request is made to the URL

<http://localhost:8181/panl-more-facets/mechanical-pencils-more/brandandname/Clutch/m/?code=b&limit=-1>



Note: The ONLY change to the URL is swapping the `panl-results-viewer` part to `panl-more-facets`, and appending the `code` and `limit` query parameters.

The `code` query parameter is the LPSE code for the facet that you are requesting - in this case 'b' for 'brand' and the `limit` query parameter is set to `-1` - this will tell Solr to return ALL facet results. Alternatively the limit parameter could be set to any positive integer value and Solr will return up to that exact number of facet results (if they are available).

Determining The Facet Limit

Within the `available.facets` JSON array, for each of the objects contained within this array, there is a key of `facet_limit` which is the maximum number of facets that Solr has

been configured to return. In the case of the `mechanical-pencils-more` CaFUP, it is set to 10, and the available facet object returned is as follows:

```

01 {
02   "facet_limit":10,
03   "uris":{
04     "before":"/",
05     "after":"/Clutch/bm/"
06   },
07   "values":[ ... ],
08   "facet_name": "brand",
09   "name": "Brand",
10   "panl_code": "b"
11 }
```

To determine whether more facets are available, the logic is as follows:

If

```
facet_limit == -1
```

OR

```
facet_limit <= values.length
```

Then more facets are available.

Note: there is a case where the number of available facets are equal to the number of values that are returned, however there is no way for Solr to return the exact number of facets that are available. In this case the call to the Panl more facets service will not return any new results.

The logic in the Panl Results Viewer is

```

01 if(facet.facet_limit !== -1 && facet.facet_limit <= facet.values.length) {
02   // show the 'See all...' link
03 }
```

And the Panl Results Viewer always requests all remaining facets so that the call of:

<http://localhost:8181/panl-more-facets/mechanical-pencils-more/brandandname/Clutch/m/?code=b&limit=-1>

Returns the following JSON

```

01 {
02   "facet_limit": -1,
03   "uris": {
04     "before": "/",
05     "after": "/Clutch/bm/"
06   },
07   "values": [ ... ],
08   "facet_name": "brand",
09   "name": "Brand",
10   "panl_code": "b"
11 }
```

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.

Success Responses

Unlike the Panl Results Viewer and Explainer, this is a separate response with no Solr JSON response included.

The JSON response object has the following form:

```

01 {
02   "error": "false",
03   "panl": {
04     "facet": {
05       "facet_limit":10,
06       "uris":{ ... },
07       "values": [ ... ],
08       "facet_name": "brand",
09       "name": "Brand",
10       "panl_code": "b"
11     }
12   }
13 }
```

The Panl response only has two keys, namely a boolean valued key of `error`, and a JSON object keyed on `panl` to the root object.



Note: The `error` key will ALWAYS be `false` for a successful return of the Panl Response object.

Within the `panl` JSON Object key, there is only one key - `facet` which is detailed below.

The "facet" JSON Object

This JSON object is identical to the items in the "`available.facets`" JSON Array - see [The "available.facets" JSON Array](#) for implementation details.

To generate the URL for the more facets service, use the `facet_limit` JSON key to determine whether there are any more facets.

Implementation notes

1. You **MUST** send through the current query (i.e. LPSE path and LPSE codes) that the user is currently searching on. This will ensure that the additional facet values that are returned will be correct for this query.
2. This functionality is only available on Regular and OR facets.

3. Solr does not allow returning a subset of the facet values, it will always return the maximum number. For example, if the initial number of facets to return is set at 20, only 20 will be returned. When you request 40, all 40 will be returned - not the 21st to 40th result which means you should replace all of the values with the values returned by this query.
4. If you want to return all facets, then set the limit query parameter to -1.

~ ~ ~ * ~ ~ ~

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



IMPORTANT: You cannot generate the Panl configuration files if your Solr collection starts with the string `panl-`. This is a reserved collection prefix for the sole use of Synapticloop Panl.

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_url>.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) |
|---|
| <pre>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) |
|--|
| 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 URL 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_url>.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 with an empty response, 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 parameter.

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_url>.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 and error prone, 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_url.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_url>.panl.properties` file.

Property References Format Explained

Each of the properties referenced below will have a brief overview line, 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. This does not affect the `panl.param.*` properties which **ALWAYS** have a LPSe length of 1.

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 URL that the server is registered to handle
 - **Collection** - applies to the Panl collection URL 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** - Not required and 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, where noted 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. It is recommended that lowercase values are always used.
 - **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, List<Character> would be a comma separated list of Characters.
-

Default If not set, this will be the default value, N/A indicates that there is no default.

Location Either

- `panl.properties` file - the Panl server configuration file
 - `<panl_collection_url>.panl.properties` - the individual Panl collection URL file
-

Notes 1. An list of notes with brief explanations

See also Links to other section headings that are of note for this particular property.

For some properties, a further explanation is included, including examples.

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_url>.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 URL 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_url>.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 URL 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

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. This will produce an ERROR if no properties start with panl.collection. are defined
 2. The <solr_collection_name> MUST match the Solr collection that you want to connect this CaFUP to.
 3. Multiple files should be comma separated and may be on new lines with the Java properties continuation of backslash \.
 4. 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.../mechanical-pencils-or/mechanical-pencils-or.panl.properties,\n.../mechanical-pencils/mechanical-pencils.panl.properties\npanl.collection.products=\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 URL paths from reading the properties files relative to the current directory of the `panl.properties` file.

This would bind the following Panl collection URLs to the respective Solr collections

- Bound URL of `/mechanical-pencils/*` would connect to the `mechanical-pencils` Solr collection
- Bound URL of `/mechanical-pencils-or/*` would connect to the `mechanical-pencils` Solr collection
- Bound URL 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_url>.panl.properties`

- | | |
|--------------|---|
| Notes | <ol style="list-style-type: none"> 1. This property is ONLY available if the Solr field type is <code>solr.DatePointField</code>. 2. This works in with conjunction with the next and previous prefixes for DATE range facets |
|--------------|---|

-
3. If this property is not set then **NO** date range searches will be available for daily ranges
-

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 URL 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.

URL paths /previous+5+years/S/
/next+5+years/S/

Canonical URL 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_url>.panl.properties

Notes 1. This property is **ONLY** available if the Solr field type is solr.DatePointField.

-
2. This works in conjunction with the next and previous prefixes for DATE range facets
 3. If this property is not set then **NO** date range searches will be available for hourly time ranges
-

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 URL 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.

| | |
|------------------|--|
| URL paths | /previous+48+hours/S/ /next+48+hours/S/ |
|------------------|--|

| | |
|----------------------------|--|
| Canonical URL paths | /previous+48+hours/S/ /next+48+hours/S/ |
|----------------------------|--|

| | |
|--------------------------|--|
| Solr query string | fq=solr_date:[NOW-48HOURS+TO+NOW] fq=solr_date:[NOW+TO+NOW%2B48HOURS] |
|--------------------------|--|

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_url>.panl.properties |
|-----------------|---------------------------------------|

-
- | | |
|--------------|--|
| Notes | <ol style="list-style-type: none"> 1. This property is ONLY available if the Solr field type is <code>solr.DatePointField</code>. 2. This works in conjunction with the next and previous prefixes for DATE range facets 3. If this property is not set then NO date range searches will be available for monthly 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>.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 URL 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.

| | |
|------------------|--|
| URL paths | /previous+12+months/S/ /next+12+months/S/ |
|------------------|--|

| | |
|----------------------------|--|
| Canonical URL paths | /previous+12+months/S/ /next+12+months/S/ |
|----------------------------|--|

| | |
|--------------------------|--|
| Solr query string | <code>fq=solr_date:[NOW-12MONTHS+TO+NOW]</code> <code>fq=solr_date:[NOW+TO+NOW%2B12MONTHS]</code> |
|--------------------------|--|

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_url>.panl.properties |
|-----------------|---------------------------------------|

-
- | | |
|--------------|---|
| Notes | <ol style="list-style-type: none"> 1. This property is ONLY available if the Solr field type is <code>solr.DatePointField</code>. 2. This works in conjunction with the hours, days, months and years suffixes. 3. 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 URL 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.

| | |
|------------------|---|
| URL paths | /next+48+hours/S/ /next+30+days/S/ /next+12+months/S/ /next+5+years/S/ |
|------------------|---|

| | |
|----------------------------|---|
| Canonical URL paths | /next+48+hours/S/ /next+30+days/S/ /next+12+months/S/ /next+5+years/S/ |
|----------------------------|---|

| | |
|--------------------------|--|
| Solr query string | <pre>fq=solr_date:[NOW+TO+NOW%2B48HOURS] fq=solr_date:[NOW+TO+NOW%2B30DAYS] fq=solr_date:[NOW+TO+NOW%2B12MONTHS] fq=solr_date:[NOW+TO+NOW%2B5MONTHS]</pre> |
|--------------------------|--|

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.

Scope Field

Required No

Value String

Default N/A

Location <panl_collection_url>.panl.properties

- Notes**
1. This property is **ONLY** available if the Solr field type is `solr.DatePointField`.
 2. This works in conjunction with the hours, days, months and years suffixes.
 3. 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 URL 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.

| | |
|------------------|---|
| URL paths | /previous+48+hours/S/ /previous+30+days/S/ /previous+12+months/S/ /previous+5+years/S/ |
|------------------|---|

| | |
|----------------------------|---|
| Canonical URL paths | /previous+48+hours/S/ /previous+30+days/S/ /previous+12+months/S/ /previous+5+years/S/ |
|----------------------------|---|

| | |
|--------------------------|--|
| Solr query string | 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_url>.panl.properties

- Notes**
1. This property is ONLY available if the Solr field type is `solr.DatePointField`.
 2. This works in with conjunction with the next and previous prefixes for DATE range facets
 3. 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 URL 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.

URL paths /previous+5+years/S/
/next+5+years/S/

Canonical URL 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.decimal.point

Whether to use a decimal comma, or a decimal point as the separator between the integer and fractional point for decimal numbers (i.e. float or double).

Scope Server

Required No

Value Boolean

Default true

Location <panl_collection_url>.panl.properties

Notes 1. If missing, or not set, the default is to use the decimal point separator.

See also

Example:

The number

1,234,567.89

uses the decimal point '.' as the separator between the integer and the fractional part, whereas the number

1.234.567,89

uses the decimal comma ',' as the separator between the integer and the fractional part.

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_url>.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 URL 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.

| | |
|------------------|---------------------------|
| URL paths | /Blue/W/ /Blue/Red/WW/ |
|------------------|---------------------------|

| | |
|----------------------------|---------------------------|
| Canonical URL 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_url>.panl.properties

Notes 1. If this property is not set to index (case sensitive) then this property will default to the value count.

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) [add] B (2) [add] C (11) [add] D (3) [add] M (6) | Authors (A-Z) (4) [add] C (11) [add] M (6) [add] D (3) [add] 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_url>.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 as 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_url>.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_url>.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 there is only one value and associated count for it.

Scope Collection

Required No

Value Boolean

Default false

Location <panl_collection_url>.panl.properties

Notes

See also

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_url>.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 result (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

Values Integer

Default 1

Location <panl_collection_url>.panl.properties

Notes 1. a WARNING message will be printed to the console if this property is not set.

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 URL path

panl.lpse.order

The order of the Panl LPSE codes to generate the URL path with.

Scope Collection

Required Yes

Value List<String>

Default N/A

Location <panl_collection_url>.panl.properties

Notes 1. The Panl server will ERROR if this property does not exist, and

exit.

2. 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
3. When the canonical URL path is being generated, the LPSE order is used.
4. The list of LPSE codes must match the defined LPSE codes for the `panl.facet.<lpse_code>` definitions

See also [panl.lpse.length](#)

panl.multivalue.<lpse_code>

This property is generated from the Solr managed schema file and configures the Panl field to be multivalued.

Scope Field

Required No

Value Boolean

Default `false`

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 is used for single page search interface JSON response object.

See also

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_url>.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 Panl 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_url>.panl.properties

Notes 1. This value must be the lowercase value `true` for this property to be set.

See also [`panl.or.always.<lpse_code>`](#)

panl.or.always.<lpse_code>

Set this facet to always return

Scope Collection

Required No

Value Boolean

Default `false`

Location `<panl_collection_url>.panl.properties`

Notes 1. This value must be the lowercase value `true` for this property to be set.
2. This will only be applicable to OR facets

See also [`panl.or.facet.<lpse_code>`](#)

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_url>.panl.properties`

Notes 1. On Panl Server property file generation, the default LPSE code 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 number of rows URL value.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.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 number of rows URL value.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.panl.properties

Notes

See also [panl.param.numrows](#)
[panl.param.numrows.prefix](#)
[solr.numrows.default](#)

panl.param.page

Set the LPSE code for page number for the results, the first page being page 1.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.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 value in the LPSE URL path.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.panl.properties

Notes

See also [panl.param.page](#)

[panl.param.page.suffix](#)

panl.param.page.suffix

The suffix to place before the page number value in the LPSE URL path.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.panl.properties

Notes

See also [panl.param.page](#)
[panl.param.page.prefix](#)
[panl.lpse.order](#)

panl.param.passthrough

The LPSE code for URL 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_url>.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 when the canonical URL is generated.

Scope Collection

Required Optional

Value Boolean

Default false

Location <panl_collection_url>.panl.properties

Notes

See also [panl.lpse.order](#)

[panl.param.passthrough](#)

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_url>.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

See also [panl.lpse.length](#)

panl.param.query.operand

The LPSE code that sets the Query Operand for the Solr search server.

Scope Collection

Required Yes

Value Character

Default o

Location <panl_collection_url>.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_url>.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 of the facet in the URL path.

Scope Field

Required Optional

Value String

Default N/A

Location <panl_collection_url>.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_url>.panl.properties

Notes 1. 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.suppress.<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_url>.panl.properties

Notes

1. If the panl.range.facet.<lpse_code> is not set to true, then this field will be silently ignored.
2. If not included, is an empty or blank value, it will default to the tilde character.
3. 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.suppress.<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_url>.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.suppress.<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_url>.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.suppress.<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_url>.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.suppress.<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_url>.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.suppress.<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_url>.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.suppress.<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 Field

Required No

Value Boolean

Default `false`

Location <panl_collection_url>.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.suppress.<lpse_code>](#)

panl.range.prefix.<lpse_code>

Set the prefix for the range which will be prepended to the URL path part.

Scope Field

Required No

Value String

Default N/A

Location <panl_collection_url>.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.suppress.<lpse_code>](#)

panl.range.suffix.<lpse_code>

Set the suffix for the range which will be appended to the URL path part.

Scope Field

Required No

Value String

Default N/A

Location <panl_collection_url>.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.range.suppress.<lpse_code>](#)

panl.range.suppress.<lpse_code>

Sets whether to suppress the output of the range facet in the available filters response object

Scope Field

Required Optional

Value Boolean

Default false

Location <panl_collection_url>.panl.properties

Notes

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_url>.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.fields.default](#)
[panl.results.fields.empty](#)

panl.results.fields.default

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_url>.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.fields.<field_set>](#)
[panl.results.fields.empty](#)

panl.results.fields.empty

A FieldSet which always returns no documents.

Scope Collection

Required No

Value List<String> - A comma separated list of Solr field names

Default N/A

Location <panl_collection_url>.panl.properties

Notes There will **ALWAYS** be a FieldSet named '`empty`' which returns no fields from the Panl server (and therefore no documents). If this FieldSet is defined, then the overriding definition will be ignored and no Solr fields will be returned.

See also [panl.results.fields.<field_set>](#)
[panl.results.fields.default](#)

panl.results.testing.urls

Whether to enable the in-built Panl Results Viewer, Panl Results Explainer, and Single Page Search web apps.

Scope Server

Required No

Value Boolean

Default `false`

Location `panl.properties`

- Notes**
1. Setting this to `false` will disable Panl from serving up the testing web apps, i.e.:
 - a. Panl Results Viewer
 - b. Panl Results Explainer
 - c. Panl Single Page Search
 2. It is recommended that this is set to `false` for production use.
-

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 URL 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**
- 3. 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.
 - 4. 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_url>.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_url>.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_url>.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_url>.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_url>.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 URL path.

Scope Collection

Required No

Value Integer

Default 10

Location <panl_collection_url>.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 URL 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 URL 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 URL Paths (CaFUPs)

The design consideration of the collections and FieldSet is to allow having multiple views over the same returned search data.



Notes: For brevity of documentation, the Collections and FieldSet URL 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 URLs, 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 URL 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/` URL path which will show the Panl configuration and additional explanations for the given URL path.

The single search page interface web app is available on the `/panl-single-page-search/` URL path that shows a search interface with all of the configured facets. This includes a search button which will take the user through to the search results for the collection.

All 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: The footer of all of the in-built Panl web-apps notes:

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_url>.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 sensibly 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.fiel
d=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.

~ ~ ~ * ~ ~ ~

Afterword

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 use the word '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/` URL 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) URL paths - and the additional property for keeping this token in the canonical URL 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

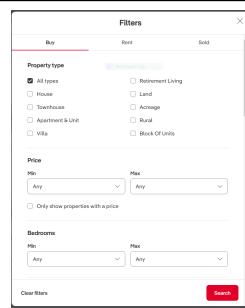
1. Additional support for Solr field types

Some will be implemented, some will probably be ignored (anything geospatial is probably not going to be included).

2. Single search page [Released in version 9.1.0]

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 [Released in version 9.1.0]

Dynamically generate the minimum and maximum value for a range for a facet value.

7. Suppress range values [Released in version 9.1.0]

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 [Released in version 9.2.0]

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 [Released in version 9.1.0]

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 [Released in version 9.1.0]

~~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>.¹⁹

12. Integrated typeahead

An example and implementation of type-ahead in the search results.

13. Additional text query options

Add in an LPSE code to be able to search on a specific field, fields, or all fields, rather than the default search field, or search for individual fields.

14. Panl configuration editor

A GUI to edit (and validate) the Panl configuration files making it easier for a developer to get the configuration correct.

15. Arbitrary Solr Query Addition

Being able to add arbitrary Solr query params to individual collections - there is some Solr functionality which does not require any configuration parameters to be surfaced through the Panl server.

16. Update RANGE LPSE URL encoding

Change RANGE LPSE URL from <lpse_code>(+/-)<lpse_code> to just <lpse_code>(+/-) - not really sure why this wasn't implemented in the initial release, will require backwards compatibility checking.

17. Always on OR facets [Released in version 9.2.0]

~~OR facets will not be presented if another facet has been selected, this will force the facet to always be returned, this will allow the results to continue to grow.~~

18. Arbitrary DATE Ranges

Being able to have a way of having arbitrary ranges - e.g. from 3 to 6 months, or 3 to 6 months before - this almost ties in with the DATE Range facet update element.

19. Arbitrary Separated Values for OR facets

Rather than having a prefix and/or suffix added to the Solr field value for each OR facet,

¹⁹ Although German users should be fine with the default implementation from Panl.

being able to have a separator character (or characters) between the values. E.g. for the following example URL:

```
/Manufactured+By+The+Caran+d'Ache+Company/Manufactured+By+The+BIC+Company/bb/
```

Should be able to be configured to be displayed as (or something along the lines of it):

```
/Manufactured+By+The+Caran+d'Ache, or+BIC+Company/b/
```

~ ~ ~ * ~ ~ ~

In any case, I hope that you enjoy - and more importantly, implement and use - the Panl server for your Solr search server implementation.

~ ~ ~ * ~ ~ ~

Appendices

Panl URL Bindings

The following is a list of URLs that Panl binds to upon start up

User Defined URL Bindings

Panl has internal URL bindings which will always exist and then binds any user configured CAFUPs to the root of the server, in this book the defined URLs are:

- Panl in-built URL bindings (These bindings are ALWAYS available)
 - <http://localhost:8181/mechanical-pencils/default/>
 - <http://localhost:8181/mechanical-pencils/empty/>
- User defined through the CAFUP configuration
 - <http://localhost:8181/mechanical-pencils/firstfive/>
 - <http://localhost:8181/mechanical-pencils/brandandname/>

Depending on the configuration of the Panl server, additional data sets may also have URL bindings:

- Book store
 - <http://localhost:8181/book-store/default/>
 - <http://localhost:8181/book-store/empty/>
- Mechanical Pencils (OR Facet)
 - <http://localhost:8181/mechanical-pencils-or/default/>
 - <http://localhost:8181/mechanical-pencils-or/firstfive/>
 - <http://localhost:8181/mechanical-pencils-or/brandandname/>
 - <http://localhost:8181/mechanical-pencils-or/empty/>
- Mechanical Pencils (More facets)
 - <http://localhost:8181/mechanical-pencils-more/default/>
 - <http://localhost:8181/mechanical-pencils-more/firstfive/>
 - <http://localhost:8181/mechanical-pencils-more/brandandname/>
 - <http://localhost:8181/mechanical-pencils-more/empty/>
- Simple Date
 - <http://localhost:8181/simple-date/default/>
 - <http://localhost:8181/simple-date/firstfive/>

- <http://localhost:8181/simple-date/empty/>

Panl Internal Functional URL Bindings

Additional functionality for a single search page implementation, along with retrieving more facets for a specific LPSE code are bound to URLs.

The Single Search Page JSON response is bound to the URL with the form

`https://localhost:8181/panl-single-page/<panl_collection>`

And the More Facets retrieval handler is bound to the URL with the form:

`https://localhost:8181/panl-more-facets/<panl_collection>/<fieldset>/<lpse_path>/<lpse_codes>/?code=<lpse_code>&limit=<limit>`

The following URLs (depending on the Panl configuration) will be bound for examples within this book:

- Single Search Page configuration information:
 - <http://localhost:8181/panl-single-page/mechanical-pencils/>
 - <http://localhost:8181/panl-single-page/mechanical-pencils-or/>
 - <http://localhost:8181/panl-single-page/mechanical-pencils-more/>
 - <http://localhost:8181/panl-single-page/book-store/>
 - <http://localhost:8181/panl-single-page/simple-date/>
- More facets
 - <http://localhost:8181/panl-more-facets/book-store/default/>
 - <http://localhost:8181/panl-more-facets/book-store/empty/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-or/default/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-or/firstfive/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-or/brandandname/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-or/empty/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-more/default/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-more/firstfive/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-more/brandandname/>
 - <http://localhost:8181/panl-more-facets/mechanical-pencils-more/empty/>
 - <http://localhost:8181/panl-more-facets/simple-date/default/>
 - <http://localhost:8181/panl-more-facets/simple-date/firstfive/>

- <http://localhost:8181/panl-more-facets/simple-date/empty/>

Internal Testing URL Bindings

The in-built example/testing functionality is bound to the following URLs:

- <http://localhost:8181/panl-results-viewer/>
- <http://localhost:8181/panl-results-explainer/>
- <http://localhost:8181/panl-single-page-search/>

The above URLs can be disabled by setting the `panl.results.testing.urls=false` property in the `panl.properties` file.

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 Collections and FieldSet URL Paths. This is the URL 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 surrounded by 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.*

| J

JSON: *JavaScript Object Notation, an open standard file and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute-value pairs.*

| K

| L

LPSE code: *The Last Path Segment Encoded string that the Panl server uses to parse the preceding URL 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 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 URL: *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 URLs can be connected to a Solr search collection. The Panl collection URL 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_url.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 URL 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 URL path, and optionally for the display of the value.*

Properties: *A text file format where each line defines a parameter which is stored as a pair of strings, one storing the name of the parameter (called the key), and the other storing 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 URL 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 URL 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.*

URL: *Uniform Resource Locator is a subset of URI which specifically references a webpage.*

URL path part: *A part of the path of a URL, in the examples in the book the path part is taken as everything after the hostname. A URL is of the form:*

scheme ":" ["/" authority] path ["?" query] ["#" fragment]

| 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.*

XML: *eXtensible Markup Language which is the format the Solr uses for the managed schema and configuration files.*

| 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.



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

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
```

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_url>.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 panl_properties_file_location] \
13     [-config solr_config_location] \
14     [-port port_number] \
15     [-overwrite true_or_false] \
16     [-schema solr_schema_location]
```

```

17
18
19 Where:
20   [command] is one of:
21     server - start the server, or
22     generate - generate an example panl.properties file from an existing Solr
23       managed-schema.xml file
24
25 [SERVER] To start the Panl server:
26
27   java -jar panl.jar \
28     server \
29     [-properties panl_properties_file_location] \
30     [-port port_number]
31
32
33           ~ ~ ~ * ~ ~ ~
34
35
36 [GENERATOR] To generate the panl.properties file for each collection
37
38   java -jar panl.jar \
39     generate \
40     [-properties panl_properties_file_location] \
41     [-overwrite true_or_false] \
42     -schema solr_schema_location
43
44
45           ~ ~ ~ * ~ ~ ~
46
47
48 If you choose the 'server' command, the following command line options are
49 available:
50
51   [-properties panl_properties_file_location] (optional - default
52     panl.properties) the properties file to load, if this property is not
53     included, the default panl.properties file will be used which __MUST__
```

```

54    reside in the same directory as the server start command
55
56 [-port port_number] (optional) the port number to start the server on. The
57 default port number is 8181
58
59
60      ~ ~ ~ * ~ ~ ~
61
62
63 If you choose the 'generate' command, the following command line options are
64 available:
65
66 [-properties panl_properties_file_location] (optional - default
67 panl.properties) the base properties file to write the generated
68 configuration out to. If this property is not included, the default
69 panl.properties filename will be used with each collection file that is
70 generated named:
71
72     <panl_collection_url>.panl.properties
73
74 NOTE: If the files exist, the generation will TERMINATE, you will need to
75 remove those files before the generation - unless you have the
76 -overwrite true command line option present
77
78 [-overwrite true_or_false] (optional - default false) - whether to overwrite
79 the file if it exists
80
81 -schema solr_schema_location(s) (mandatory) the managed-schema.xml
82 configuration file(s) to read and generate the panl.properties file from.

```

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>

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_url>.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 MORE facetting

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

Mechanical pencils OR facetting

<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_url.panl.properties.template

Solr Version 8 & 7 Integration Notes

For integrations of the Solr server with the Panl Server, there are differences between the commands used to index the data by the Solr server, the response payload from the Solr server, and the connection strings to the Solr server which is configured in the `panl.properties` file.

Apart from the above, the Panl response payload will be identical irrespective of the Solr server version used.

SolrJ

The SolrJ connectors are changed, with the available list of SolrJ clients for version 8:

- `HttpSolrClient`
- `Http2SolrClient`
- `LBHttpSolrClient`
- `LBHttp2SolrClient`
- `CloudSolrClient`
- `CloudHttp2SolrClient`

And for version 7

- `HttpSolrClient`
- `LBHttpSolrClient`
- `CloudSolrClient`

Solr Configuration files

The `managed-schema.xml` file has now been renamed to simply `managed-schema`.

The Solr configuration file `solrconfig.xml` files change between versions.

JSON Response Object

The JSON response object has also changed



IMPORTANT: The returned Solr JSON object has changed, however the returned Panl response JSON object has not. This will only ever affect your integration when dealing with the Solr JSON. The Panl Results Viewer web app

has taken this into account to support these versions.

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). In version 7 and 8 the `response` array contains the documents, without any further keys. Contrast this with the Solr 9 response which holds the documents in the `response.docs` JSON array and has additional keys containing information about the number of total documents:

```

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)

```
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\exemplar\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 facetting

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

cd SOLR_INSTALL_DIRECTORY

java -Dc=simple-date -Dtype=application/json -jar example\exampledocs\post.jar ←
PANL_INSTALL_DIRECTORY\sample\data\simple-date.json

```

***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) |
|---|
| <pre>cd SOLR_INSTALL_DIRECTORY bin/solr zk upconfig -d ← PANL_INSTALL_DIRECTORY/sample/solr/mechanical-pencils -n mechanical-pencils ← -z localhost:9983</pre> |

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) |
|--|
| <pre>cd SOLR_INSTALL_DIRECTORY bin/solr create -c mechanical-pencils -n mechanical-pencils -s 2 -rf 2</pre> |

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) |
|---|
| <pre>cd SOLR_INSTALL_DIRECTORY java -Dc=mechanical-pencils -Dtype=application/json -jar example/exemplledocs/post.jar ← PANL_INSTALL_DIRECTORY/sample/data/mechanical-pencils.json</pre> |

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/exemplar/docs/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

```

```

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
 - a. change the `distributionBaseName` - e.g. `solr-panl-6`

```

distributions {
  main {
    distributionBaseName = 'solr-panl-6'
  }
}
```

- b. 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:?.?.?'
    ...
}
```

3. Edit the Solr schema configuration and managed-schema
 - a. 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).
 - b. Edit the file to include the relevant fields that you want to index and search on
4. Look at the `com.synapticloop.panl.server.client` package, adding in the Solr Clients that are applicable to your version
5. Update the `com.synapticloop.panl.server.client.PanlClient` class to reference the correct clients.
6. Update the
`com.synapticloop.panl.server.handler.CollectionHelper#getPanlClient()`
factory method to return the correct client



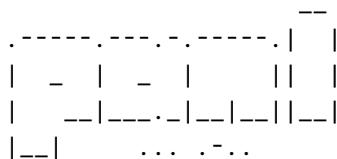
IMPORTANT: The returned JSON object may have changed between versions which may cause problems with generation and returning the Panl response.

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.

~ ~ ~ * ~ ~ ~

End Plate

~ ~ ~ * ~ ~ ~



~ ~ ~ * ~ ~ ~