
GNDStk Documentation

Release 1.0

Martin Staley

Oct 21, 2021

1	INTRODUCTION & PRIMER	2
1.1	Introduction	2
1.1.1	Description	2
1.1.2	Background	2
1.1.3	Acknowledgements	2
1.2	Building GNDStk	2
1.2.1	Download	2
1.2.2	Build & Test	2
1.2.3	Summary	2
1.2.4	Your Own Application	2
1.2.5	Alternative: Bash Script	2
1.2.6	Header-Only Library	2
1.3	Tutorial	2
1.3.1	Basics + Core Interface	2
1.3.2	Read and Write GNDS	2
1.3.3	Data Structure “Direct”	2
1.3.4	Smart Query System	2
1.3.5	GNDS Creation	2
1.3.6	Advanced Examples	2
2	BASIC CONSTRUCTS	2
2.1	Primary Classes	2
2.1.1	Tree	2
2.1.2	Node	2
2.1.3	XML	2
2.1.4	JSON	2
2.2	Node: Major Capabilities	2
2.2.1	Query	2
2.2.2	Add Data	2
2.3	Functions	2
2.3.1	foo	2
2.3.2	bar	2
2.3.3	etc	2
2.4	Reading & Writing	2
2.5	Miscellaneous Utilities	2
2.5.1	Global Flags	2

2.5.2	Diagnostics	2
2.5.3	Other	2
3	CORE INTERFACE	2
3.1	Motivation	2
3.2	Query System, Part 1	2
3.2.1	Meta & Child	2
3.2.2	Operators	2
3.2.3	Query Metadata	2
3.2.4	Query Child Nodes	2
3.3	Query System, Part 2	2
3.3.1	Sequence Queries	2
3.3.2	Multi-Queries	2
3.3.3	Conversion & Filters	2
3.4	Creating Data	2
3.4.1	Direct	2
3.4.2	Using “Query” Objects	2
3.5	Conversion Scheme	2
3.6	Advanced Topics	2
4	HIGH-LEVEL INTERFACE	2
4.1	Component Base	2
4.1.1	Motivation	2
4.1.2	Capabilities	2
4.1.3	Usage Requirements	2
4.2	Main Structures	2
4.2.1	Examples	2
4.3	Field Concepts	2
4.3.1	Required	2
4.3.2	Optional	2
4.3.3	Defaulted	2
4.4	C++ Version-Specific	2
4.4.1	GNDS v1.9	2
4.4.2	GNDS v2.0	2
4.5	Python Bindings	2
5	SEARCH	2
6	REFERENCE	2
6.1	Core Classes	2
6.1.1	Tree	2
6.1.2	Node	2
6.1.3	XML	2
6.1.4	JSON	2
6.1.5	Meta	2
6.1.6	Child	2
6.1.7	KeywordTup	2
6.2	I/O and Related	2
6.3	Node: Major Capabilities	2
6.3.1	meta()	2
6.3.2	one() and many()	2
6.3.3	child()	2
6.3.4	operator()	2
6.3.5	operator[]	2
6.3.6	MetaRef & ChildRef	2

6.4	Meta & Child Operators	2
6.5	convert()	2
6.5.1	Tree/XML/JSON	2
6.5.2	For Metadata	2
6.5.3	For Child Nodes	2
6.6	Canned Keywords	2
6.6.1	For Metadata	2
6.6.2	For Child Nodes	2
6.6.3	Special cases	2
6.7	High-Level Support	2
6.8	High-Level Interface	2
6.8.1	GNDS Version 1.9	2
6.8.2	GNDS Version 2.0	2
6.9	Miscellaneous	2
7	INDEX	2

CHAPTER 1	CHAPTER 2
INTRODUCTION & PRIMER	BASIC CONSTRUCTS
<h2>1.1 Introduction</h2> <h3>1.1.1 Description</h3> <p>Los Alamos National Laboratory’s GNDS Toolkit, or GNDStk, has been designed first and foremost to provide a powerful, intuitive, and flexible C++ language API for interacting with Generalized Nuclear Database Structure data.</p> <p>We begin by providing basic and cleanly-designed classes in which GNDS data are stored. Next, we support a robust and flexible I/O system for reading from, and writing to, both the XML and JSON file formats. Support for more file formats is anticipated in the future, as GNDS becomes more widely used.</p> <p>While GNDStk is <i>one</i> library, from which you can use any functionality you wish to at any time, we consider it conceptually to consist of roughly three major parts: basic constructs and I/O; a “core” interface, and a higher-level interface that will also be equipped with Python bindings for users who wish to take advantage of them. Let’s say a bit more about all of these elements.</p> <h4>BASICS</h4> <p>Here we have the basic requisite data structures and functions, as well as flexible and easy-to-use GNDS file I/O capabilities. Along with these also come, of course, the numerous and sundry utilities needed for their implementation. Some of the utilities, e.g. those for generating diagnostic messages such as warnings and errors, may be of value in their own right to our users. We’ll therefore provide some documentation of how selected utility constructs work, without distracting us from our focus on GNDStk’s major, most interesting capabilities.</p> <h4>CORE INTERFACE</h4> <p>The heart of GNDStk lies in its Core Interface. Consider this interface to <i>include</i> the basics as described above, while adding to them a powerful, flexible, and highly user-programmable suite of <i>data query</i> and <i>creation</i> capabilities that can be used to great effect by themselves</p>	<h2>2.1 Primary Classes</h2> <h3>2.1.1 Tree</h3> <h3>2.1.2 Node</h3> <h3>2.1.3 XML</h3> <h3>2.1.4 JSON</h3> <h2>2.2 Node: Major Capabilities</h2> <h3>2.2.1 Query</h3> <h3>2.2.2 Add Data</h3> <h2>2.3 Functions</h2> <h3>2.3.1 foo</h3> <h3>2.3.2 bar</h3> <h3>2.3.3 etc</h3> <h2>2.4 Reading & Writing</h2> <h2>2.5 Miscellaneous Utilities</h2> <h3>2.5.1 Global Flags</h3> <h3>2.5.2 Diagnostics</h3> <h4>Notes</h4> <h4>Warnings</h4> <h4>Errors</h4> <h4>Context</h4> <h3>2.5.3 Other</h3>