Documentation Bootcamp

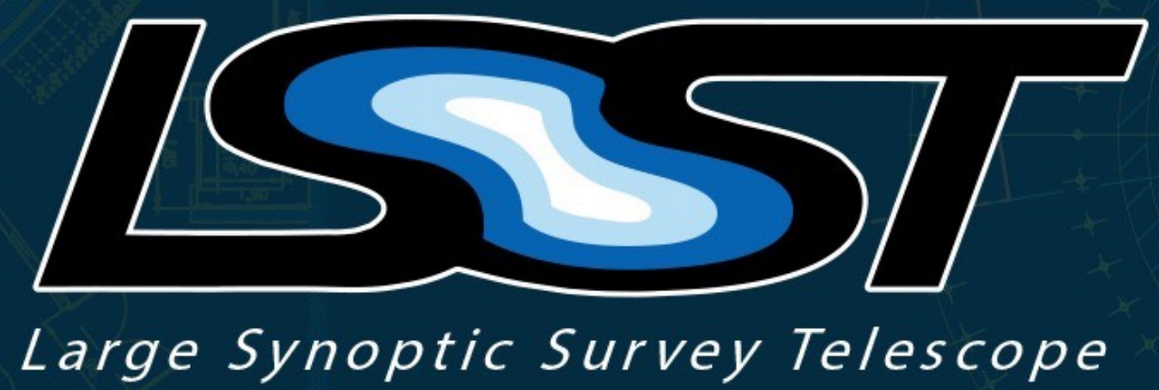
Finding it. Writing it.

Jonathan Sick, Documentation Engineer

LSST

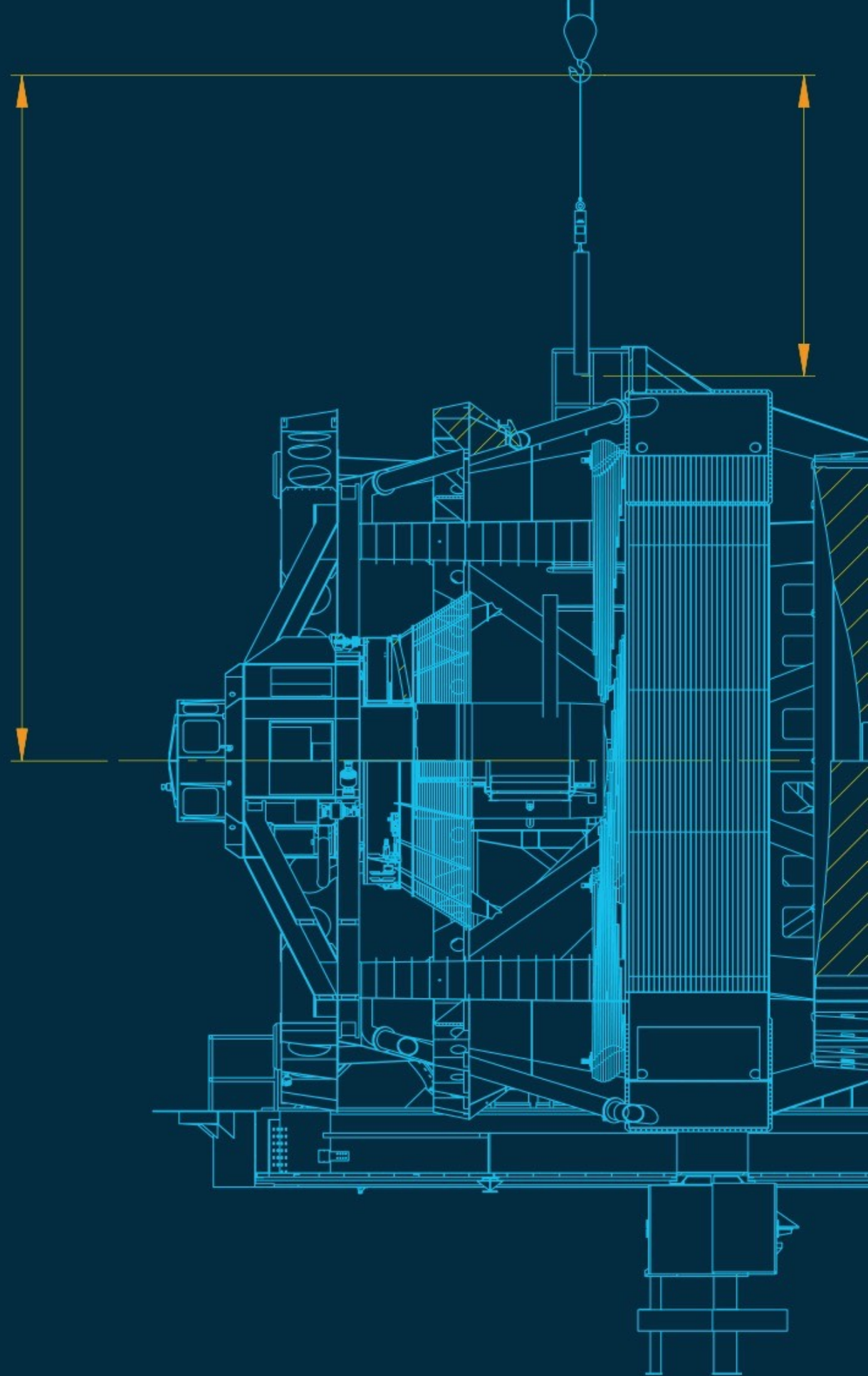
Large Synoptic Survey Telescope

Docs, or it didn't happen.



Outline

1. Tour of the docs
2. Starting a technote
3. ReStructuredText primer
4. Writing Python docstrings
5. Writing for users
6. Writing Pipelines docs



Large Synoptic Survey Telescope

Types of documentation

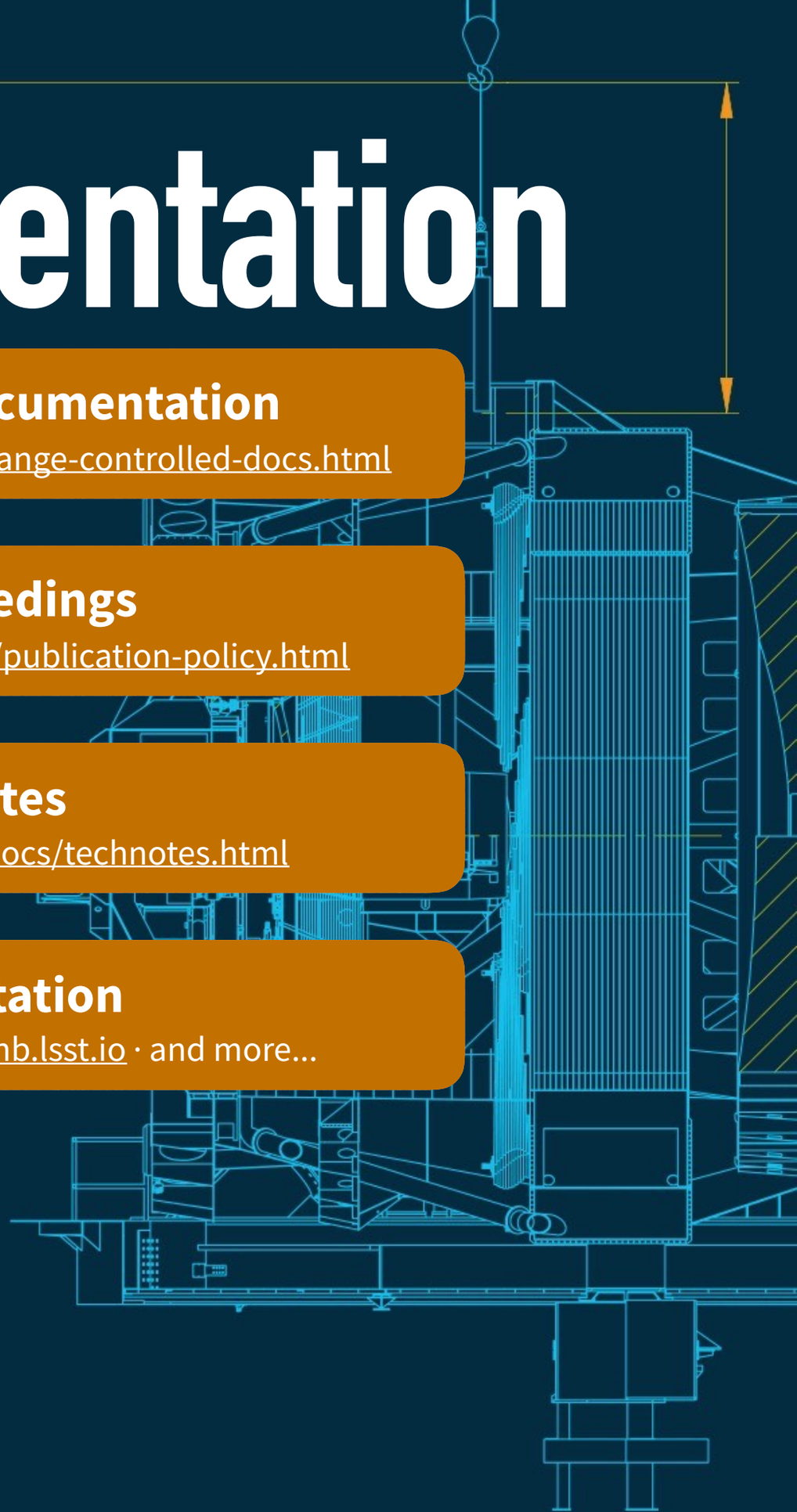
docs

change-controlled documentation
https://developer.lsst.io/project-docs/change-controlled-docs.html

papers & proceedings
https://developer.lsst.io/project-docs/publication-policy.html

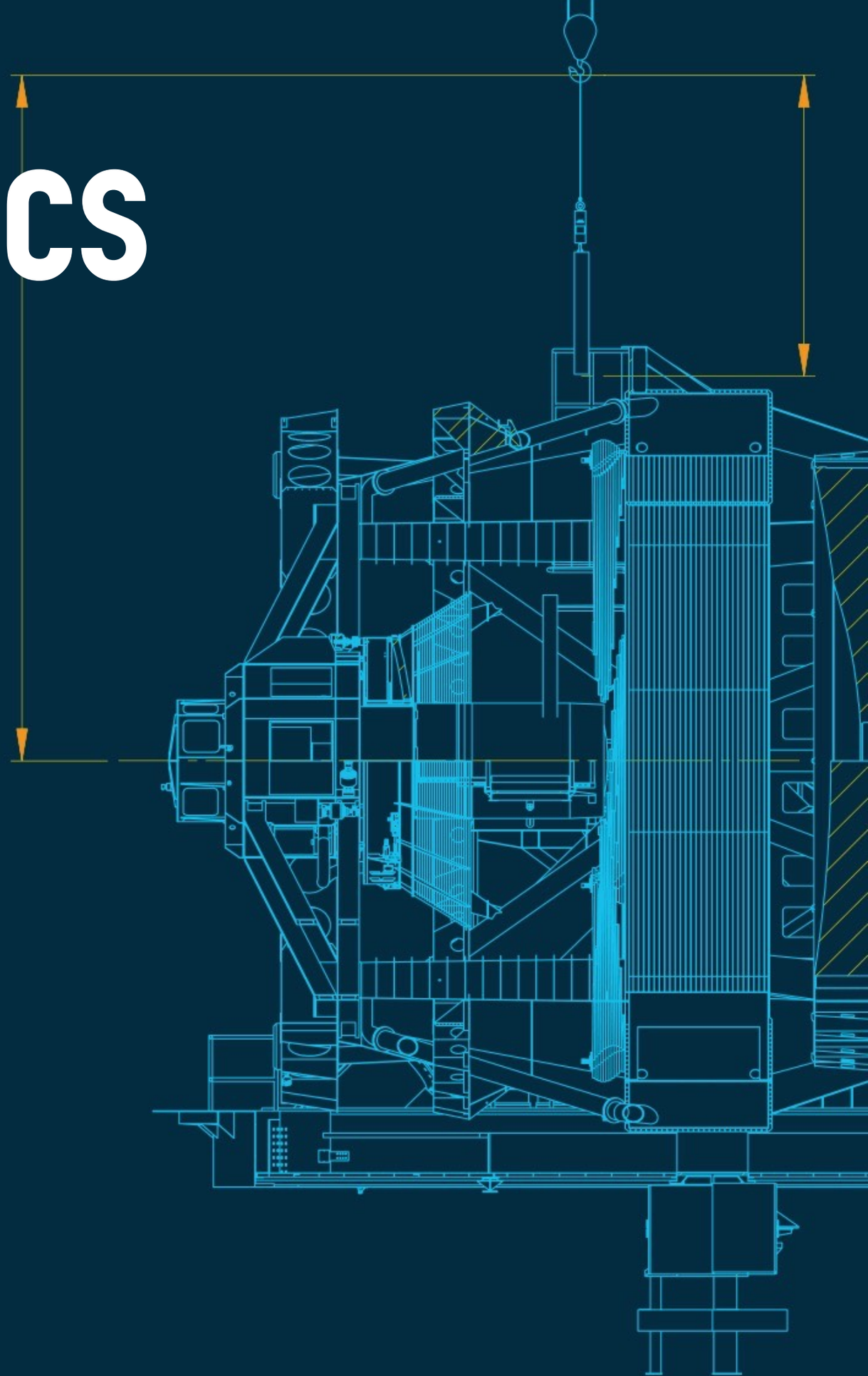
technical notes
https://developer.lsst.io/project-docs/technotes.html

user documentation
developer.lsst.io · pipelines.lsst.io · nb.lsst.io · and more...



Large Synoptic Survey Telescope

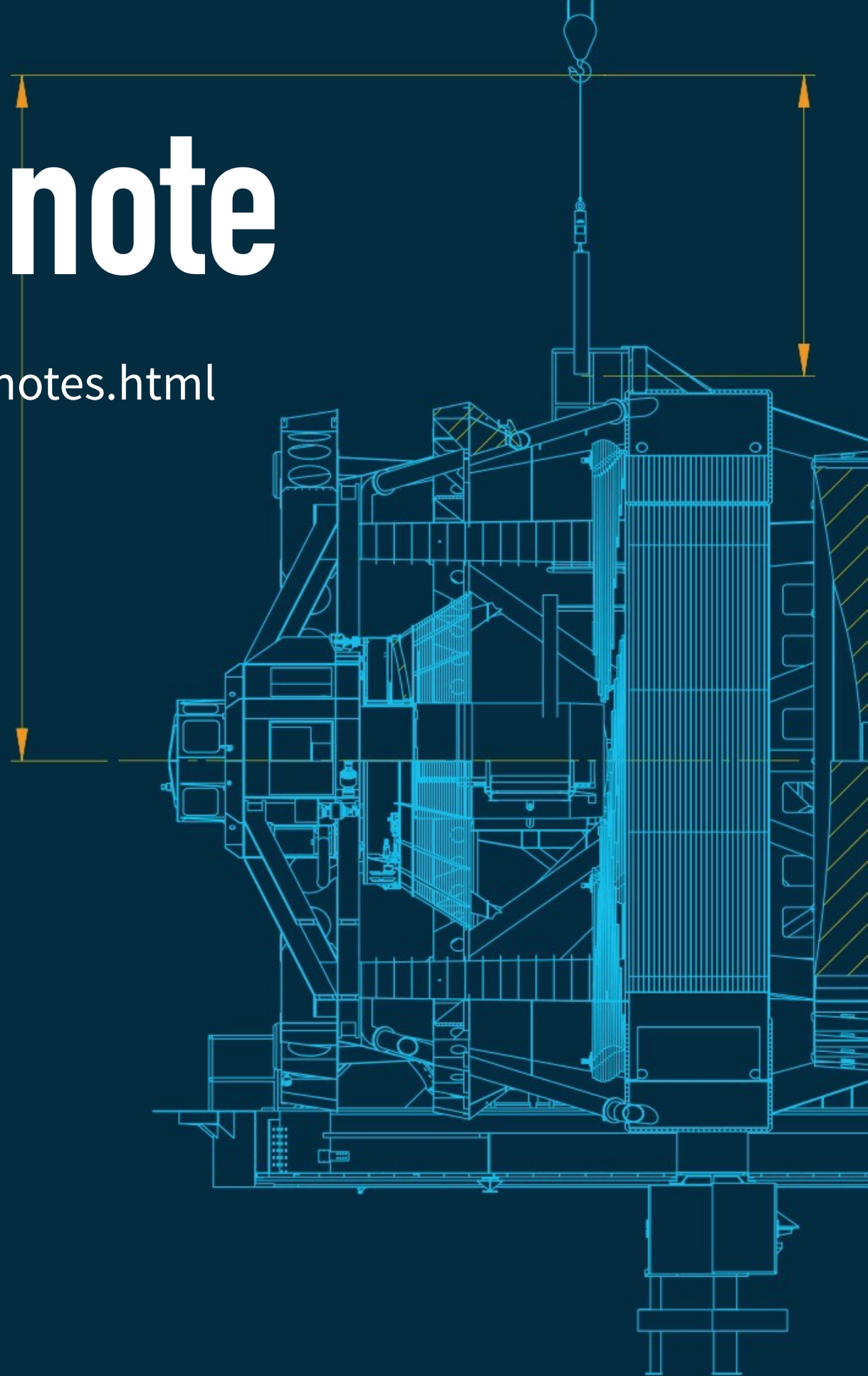
Tour of the docs



Large Synoptic Survey Telescope

Making a technote

https://developer.lsst.io/project-docs/technotes.html

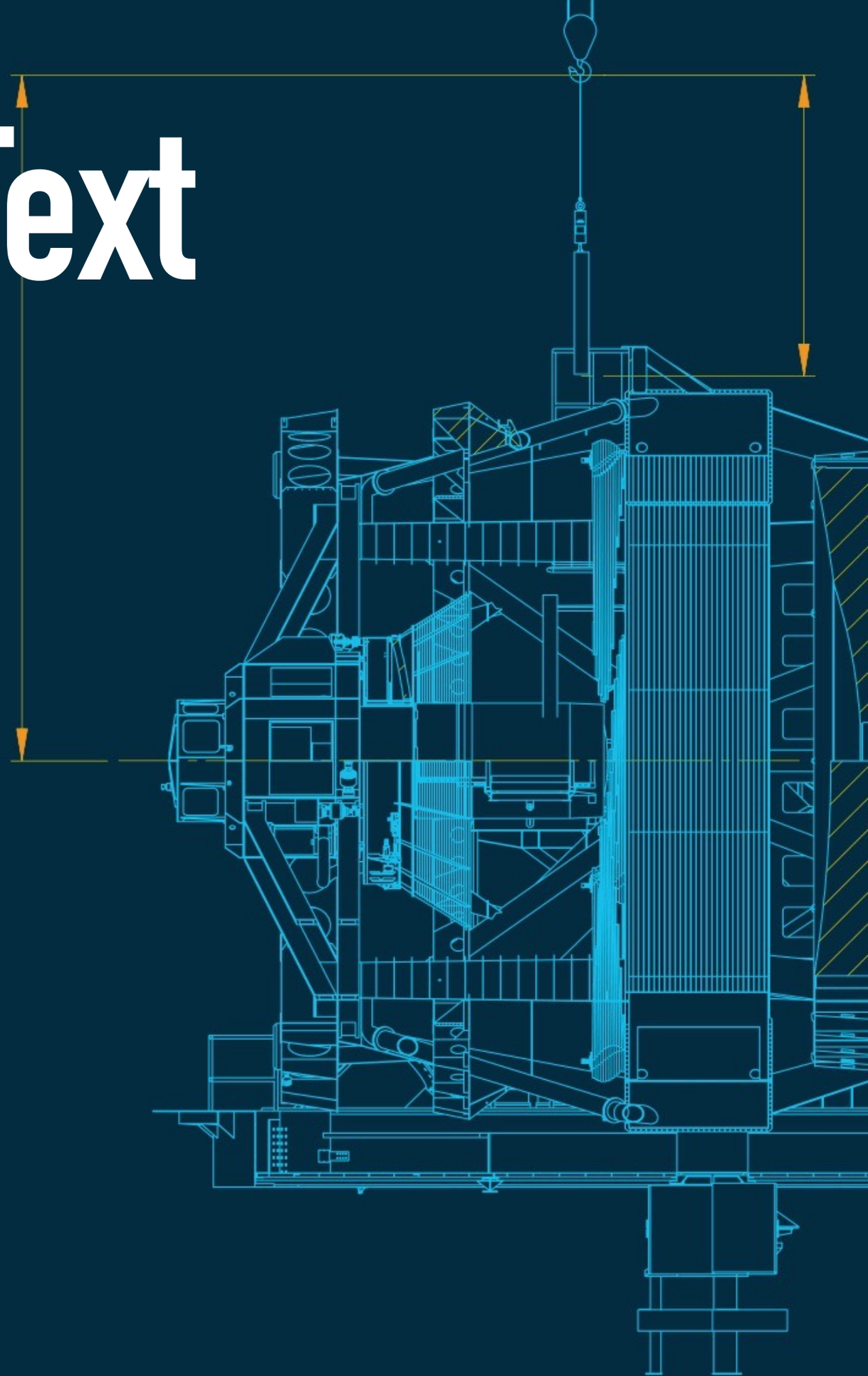


Large Synoptic Survey Telescope

reStructuredText

The reStructuredText format is central to many DM contexts:

- Python docstrings
- pipelines.lsst.io
- developer.lsst.io
- nb.lsst.io
- Technical notes



Large Synoptic Survey Telescope

ReStructuredText

Inline syntax

Bold

- **bold**

Italic

- *italic*

Monospace (inline code)

- ``monospace``

ReStructuredText

Formatting paragraphs

Put a single blank line between paragraphs.

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Proin facilisis pharetra neque, at semper nulla mattis auctor.
Proin semper mollis enim eget interdum.

Mauris eleifend eget diam vitae bibendum.
Praesent ut aliquet odio, sodales imperdiet nisi.
Nam interdum imperdiet tortor sed fringilla.
Maecenas efficitur mi sodales nulla commodo rutrum.
Ut ornare diam quam, sed commodo turpis aliquam et.
```

One-sentence per line formatting is strongly recommended for better Git diffs.

ReStructuredText

Sections

```
##########
Page title
##########

...

Section heading
===============

...

Subsection heading
------------------

...

Subsubsection heading
^^^^^^^^^^^^^^^^^^^^^
```

Page title

...

Section heading

...

Subsection heading

...

Subsubsection heading

ReStructuredText

Linking to sections using the ref role

```
.. _sec-a:

Section A
=========

See :ref:`sec-b`.

.. _sec-b:

Section B
=========

See the :ref:`previous
section <sec-a>`.
```

Section A

See Section B.

→ **Section B**

See the previous section.

ReStructuredText

Linking to other documents with the doc role

```
.
├── page-a.rst
├── page-b.rst
├── tasks
│   ├── task1.rst
│   └── task2.rst
└── tutorials
    └── tutorial-a.rst
```

tutorial-a.rst

```
:doc:`../page-a`
:doc:`/page-a`
:doc:`this task <../tasks/task1>`
```

→

Page A

Page A

this task

ReStructuredText

Hyperlinks

```
`GitHub <https://github.com>`_.

You can find the code on GitHub_.
```

`GitHub`.

You can find the code on `GitHub`.

ReStructuredText

Externally-defined hyperlinks

```
The `lsst-dm organization`_ on GitHub_.

.. _lsst-dm organization: https://github.com/lsst-dm
.. _GitHub: https://github.com
```

The lsst-dm organization on GitHub.

ReStructuredText

Multiple labels for externally-defined hyperlinks

```
The `lsst-dm organization`_ on GitHub_. `lsst-dm`_

.. _lsst-dm:
.. _lsst-dm organization: https://github.com/lsst-dm
.. _GitHub: https://github.com
```

The lsst-dm organization on GitHub. lsst-dm

ReStructuredText

Anonymous hyperlinks

```
- LDM-151 (`PDF <https://ls.st/ldm-151>`__)
- LDM-152 (`PDF <https://ls.st/ldm-152>`__)
```

two underscores
makes a link
"anonymous"

- LDM-151 (PDF)
- LDM-152 (PDF)

ReStructuredText

Link to Python APIs

```
Link to a function: `numpy.sin`.
Link to class: `astropy.table.Table`.
Link to a method: `astropy.table.Table.write`.
Link to a builtin: `list`.
```

Link to a function: numpy.sin.
Link to a class: astropy.table.Table.
Link to a method: astropy.table.Table.write.
Link to a builtin: list.

Because of "Intersphinx" you can link to not only LSST's APIs, but also the Python Standard library and most third-party packages.

ReStructuredText

Short-form API links

```
Link to a function: `~numpy.sin`.
Link to class: `~astropy.table.Table`.
Link to a method: `~astropy.table.Table.write`.
```

Link to a function: sin.
Link to a class: Table.
Link to a method: write.

ReStructuredText

Custom link text

```
`Link to a function <numpy.sin>`.
`Link to class <astropy.table.Table>`.
`Link to a method <astropy.table.Table.write>`.
```

Link to a function.
Link to a class.
Link to a method.

ReStructuredText

Linking to LSST tasks and configs

```
:lsst-task:`~lsst.pipe.tasks.calibrate.CalibrateTask`
:lsst-config-field:`~lsst.pipe.tasks.calibrate.CalibrateTask.photoCal`
```

CalibrateTask
photoCal

ReStructuredText

Lists: unordered

```
- First item.

- Second item.

- Third item.

  Another paragraph for third item.

- Fourth item.
```

ReStructuredText

Lists: ordered

```
#. First item.

#. Second item.

#. Third item.

   Another paragraph for
   third item.

#. Fourth item.
```

1. First item.
2. Second item.
3. Third item

   Another paragraph for third item.
4. Fourth item.

ReStructuredText

Lists: definition lists

```
Term A
    Description of A.

Term B
    Description of B:
    - nested
    - list

Term C
    Description of C.
```

Term A
Description of A.

Term B
Description of B:

- nested
- list

Term C
Description of C.

ReStructuredText

Showing code

```
.. code-block:: python
   :emphasize-lines: 2

   def hello():
       print('Hello world')
```

— or —

```
.. literalinclude:: example.py
   :languge: python
   :emphasize-lines: 2
```

ReStructuredText

Tables

```
.. list-table:: Slack channels
   :header-rows: 1

   * - Channel
     - Purpose
   * - ``#dm``
     - General Data Management discussion.
   * - ``#dm-docs``
     - Documentation engineering and writing.
```

See also "csv-table"

Fig. 1 Slack channels

Channel	Purpose
#dm	General Data Management discusssion.
#dm-docs	Documentation engineering and writing.

ReStructuredText

Math

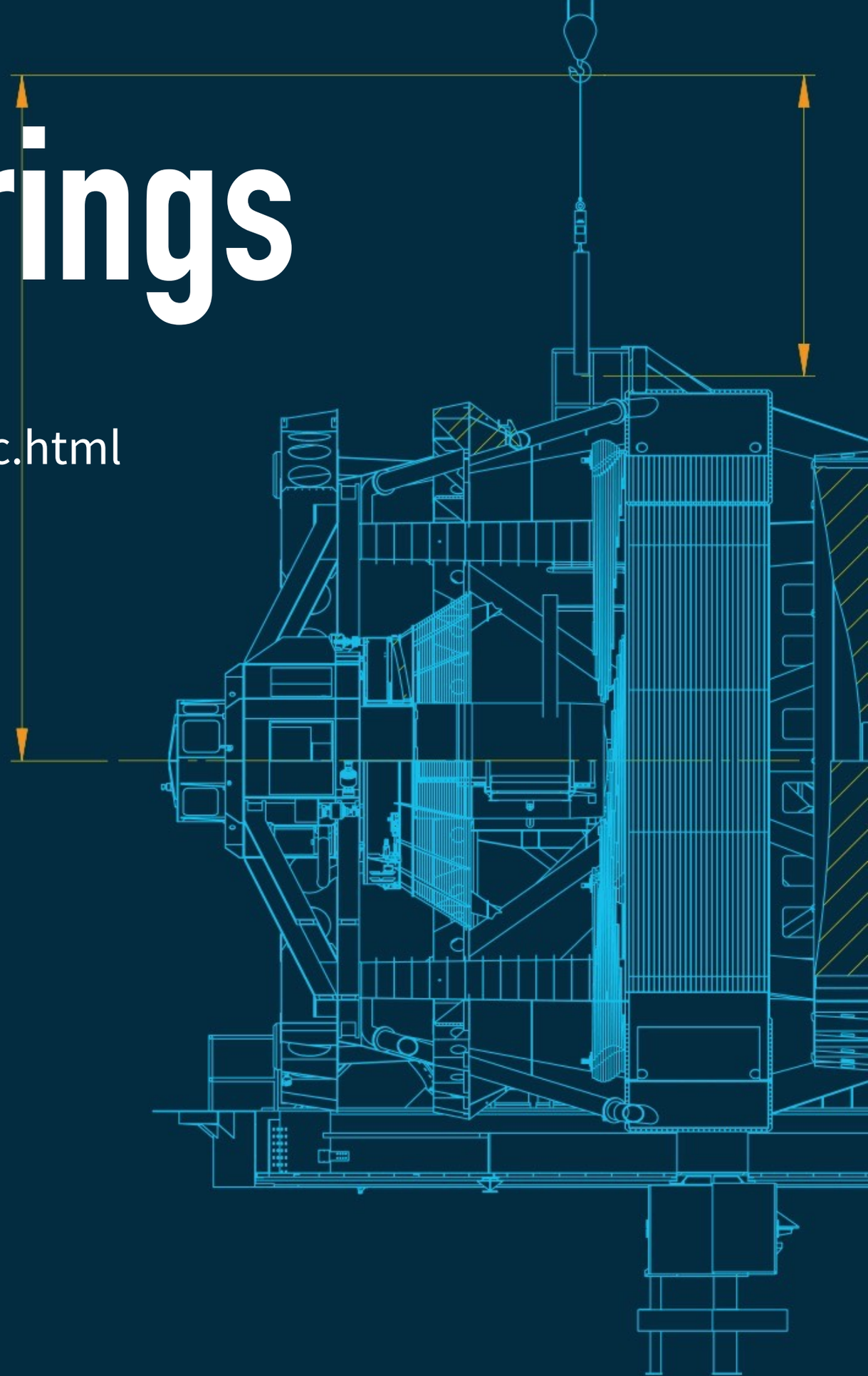
```
Write inline expressions: :math:`\sigma_\mathrm{mean} = \sigma /
\sqrt{N}`.
Or write block expressions:

.. math:: \sigma_\mathrm{mean} = \frac{\sigma}{\sqrt{N}}
   :label: sigma

:eq:`Link to equation <sigma>`.
```

Python docstrings

https://developer.lsst.io/python/numpydoc.html



Large Synoptic Survey Telescope

Docstrings

Basics

```
def upload():
    pass
```

Docstrings

Basics

```
def upload():
    """Upload content to the server.
    """
    pass
```

Use imperative mood for summary sentence.

~~`Uploads content to the server.`~~

Start and end the docstring with three double quotes. Put closing quotes on their own line.

Wrap after the 79th character column.

Docstrings

Function/method basics

```
def upload(body):
    """Upload content to the server.

    Parameters
    ----------
    body : `bytes`
        The content to upload, already encoded as
        bytes.
    """
    pass
```

② Type description. Use single-backticks to link type names.

① Variable name

③ Indent description by four spaces.

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    body : `bytes`
        The content to upload, already encoded as bytes.
    contentType : `str`, optional
        The media type of the ``body``. Common types:

        ``application/octet-stream``
            A binary file (default).
        ``application/text``
            Text content.
        ``application/json``
            A JSON-formatted document.
    """
    pass
```

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    [...]

    Returns
    -------
    url : `str`
        The canonical URL where the content can be accessed.
    """
    pass
```

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    [...]

    Returns
    -------
    url : `str`
        The canonical URL where the content can be accessed.
    remainingQuota : `int`
        The remaining available quota for the user account, in MB.
    """
    pass
```

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    [...]

    Returns
    -------
    [...]

    Raises
    ------
    lsst.example.UploadError
        Raised if a connection cannot be made with the server, or if the
        server returns a 500 error.
    lsst.example.AuthError
        Raised if the client cannot be authenticated or is not
        authorized.
    """
    pass
```

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    [...]
    Returns
    -------
    [...]
    Raises
    ------
    [...]
    Notes
    -----
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin facilisis
    pharetra neque, at semper nulla mattis auctor. Proin semper mollis enim eget
    interdum.

    Mauris eleifend eget diam vitae bibendum. Praesent ut aliquet odio, sodales
    imperdiet nisi. Nam interdum imperdiet tortor sed fringilla. Maecenas
    efficitur mi sodales nulla commodo rutrum. Ut ornare diam quam, sed commodo
    turpis aliquam et.
    """
    pass
```

Docstrings

Function/method basics

```
def upload(body, contentType='application/octet-stream'):
    """Upload content to the server.

    Parameters
    ----------
    [...]
    Returns
    -------
    [...]
    Raises
    ------
    [...]
    Notes
    -----
    [...]
    Examples
    --------
    To upload a JSON document:

    >>> import json
    >>> body = json.dumps({'hello': 'world'}).encode('utf-8')
    >>> upload(body, contentType='application/json')  # doctest: +SKIP
    ("https://example.com/uploads/1", 98)
    """
    pass
```

Docstrings

Class docstring

```
class Point:
    """A cartesian coordinate.

    Parameters
    ----------
    x : `float`
        The ``x``-axis coordinate.
    y : `float`
        The ``y``-axis coordinate.
    """

    def __init__(self, x, y):
        self.x = x
        self.y = y
```

Docstrings

Class attribute docstrings

```
class Point:
    """A cartesian coordinate.

    Parameters
    ----------
    x : `float`
        The ``x``-axis coordinate.
    y : `float`
        The ``y``-axis coordinate.
    """

    x = None
    """The ``x``-axis coordinate (`float`).
    """

    y = None
    """The ``y``-axis coordinate (`float`).
    """

    def __init__(self, x, y):
        self.x = x
        self.y = y
```

Docstrings

Class attribute docstrings, using properties

```
class Point:
    """A cartesian coordinate.

    Parameters
    ----------
    x : `float`
        The ``x``-axis coordinate.
    y : `float`
        The ``y``-axis coordinate.
    """

    @property
    def x(self):
        """The ``x``-axis coordinate (`float`).
        """
        return self._x


    def __init__(self, x, y):
        self._x = x
        self._y = y
```

Style guidelines for user documentation

https://developer.lsst.io/user-docs/index.html
https://developers.google.com/style

Large Synoptic Survey Telescope

User documentation style guidelines

Be conversational and friendly without being frivolous

❌ Please note that completion of the task requires the following prerequisite: executing an automated memory management function.

❌ Then—BOOM—just garbage-collect (or *collecter des garbáge*, as they say in French), and you're golden.

✅ To clean up, call the `collectGarbage()` method.

The most straightforward sentence is often the best sentence.

Avoid sounding "clever."

Avoid jokes unless you know what you're doing.

Contractions are fine.

From https://developers.google.com/style/tone

User documentation style guidelines

Use second person

❌

In the previous tutorial in the series we used processCcd.py to calibrate a set of raw Hyper Suprime-Cam images. Now we'll learn how to use the LSST Science Pipelines to inspect processCcd.py's outputs by displaying images and source catalogs in the DS9 image viewer. In doing so, we'll be introduced to some of the LSST Science Pipelines' Python APIs.

✅

In the previous tutorial in the series you used processCcd.py to calibrate a set of raw Hyper Suprime-Cam images. Now you'll learn how to use the LSST Science Pipelines to inspect processCcd.py's outputs by displaying images and source catalogs in the DS9 image viewer. In doing so, you'll be introduced to some of the LSST Science Pipelines' Python APIs.

User documentation style guidelines

Use second person: speak directly to the user

❌ When setting up a product, a user may specify a version, or accept the current default.

✅ When setting up a product, you may specify a version, or accept the current default.

User documentation must be actionable for the user.

User documentation style guidelines

Use the active voice

❌ The service is queried, and an acknowledgment is sent.

Passive? If you can add "by monkeys" to a sentence, it's passive.

✅ Send a query to the service. The server sends an acknowledgment.

Active voice makes it clear who performs the action, which makes for more understandable documentation.

From https://developers.google.com/style/voice

User documentation style guidelines

Only use passive voice when the actor is unimportant to the user

✅ The database was purged in January.

✅ Over 50 conflicts were found in the file.

From https://developers.google.com/style/voice

User documentation style guidelines

Write in the present tense

❌ | Send a query to the service. The server will send an acknowledgment.

✅ | Send a query to the service. The server sends an acknowledgment.

❌ | You can send an unsubscribe message. The server would then remove you from the mailing list.

✅ | If you send an unsubscribe message, the server removes you from the mailing list.

From https://developers.google.com/style/tense

User documentation style guidelines

Don't anthropomorphize software and hardware systems

❌ The `doApCorr` configuration field tells the `ForcedPhotCcdTask` whether to apply aperture corrections.

✅ Set the `doApCorr` configuration field to `True` to enable aperture corrections.

Software doesn't:

- **tell**
- **know**
- **want**
- **…**

See https://developers.google.com/style/anthropomorphism

User documentation style guidelines

Use sentence case for section headlines

❌ LSST-Related Notebook Repositories

✅ LSST-related notebook repositories

❌ Getting Notebooks from GitHub

✅ Getting notebooks from GitHub

User documentation style guidelines

Don't style code keywords in headlines

❌ Creating an `lsst.afw.table.Table`

✅ Creating an lsst.afw.table.Table

User documentation style guidelines

Introducing a list

❌

Use the LSST Science Pipelines to:

- do forced photometry
- measure galaxy shapes
- apply photometric and astrometric calibrations

The introductory sentence must be complete. Don't complete it with list items.

✅

You can use use the LSST Science Pipelines any of the following purposes:

- To perform forced photometry of a set of images.
- To measure the shapes of galaxies.
- To apply photometric and astrometric calibrations.

See https://developers.google.com/style/lists

User documentation style guidelines

Introducing a list

❌

To set up the LSST Science Pipelines:

1. Open a terminal by clicking on the **+** button in the file browser.
2. Load the LSST environment:
   - `source /opt/lsst/software/stack/loadLSST.bash`
3. Set up the LSST Science Pipelines packages:
   - `setup lsst_distrib`

✅

To set up the LSST Science Pipelines, follow these steps:

1. Open a terminal by clicking on the **+** button in the file browser.
2. Load the LSST environment:
   - `source /opt/lsst/software/stack/loadLSST.bash`
3. Set up the LSST Science Pipelines packages:
   - `setup lsst_distrib`

See https://developers.google.com/style/lists

User documentation style guidelines

List tips

- ✓ Use parallel structure.
- ✓ Add a period if the item is a complete sentence.
- ✓ Don't add add punctuation if the item is a single word.
- ✓ Use the appropriate type of list:
  - Bullet lists for unordered items
  - Numbered lists for procedures.
  - Definition lists to pair terms with definitions or explanations.

See https://developers.google.com/style/lists

User documentation style guidelines

Introducing a code sample

❌

Run:

```
setup lsst_distrib
```

to set up the LSST Science Pipelines.

✅

Run this command to set up the LSST Science Pipelines:

```
setup lsst_distrib
```

See https://developers.google.com/style/lists

User documentation style guidelines

Introducing a code sample

❌

The following command shows how to set up the LSST Science Pipelines. For more information on the setup command, see [link]:

```
setup lsst_distrib
```

✅

The following command shows how to set up the LSST Science Pipelines. For more information on the `setup` command, see [link].

```
setup lsst_distrib
```

✅

The following command shows how to set up the LSST Science Pipelines:

```
setup lsst_distrib
```

For more information on the `setup` command, see [link].

See https://developers.google.com/style/lists

User documentation style guidelines

Avoid using code keywords as English verbs or nouns

❌ | `~ProcessCcdTask.run` the task.
↳ `run()` the task.

✅ | Execute the task by calling its `~ProcessCcdTask.run` method.
↳ Execute the task by calling its `run()` method.

❌ | Create a `~lsst.verify.Measurement`.
↳ Create a `Measurement`.

✅ | Create an instance of the `~lsst.verify.Measurement` class.
↳ Create an instance of the `Measurement` class.

See https://developers.google.com/style/code-in-text

User documentation style guidelines

Absolutely never inflect code keywords

❌ A `~lsst.verify.MeasurementSet` contains `~lsst.verify.Measurement`\s.

↳ A `MeasurementSet` contains `Measurements`.

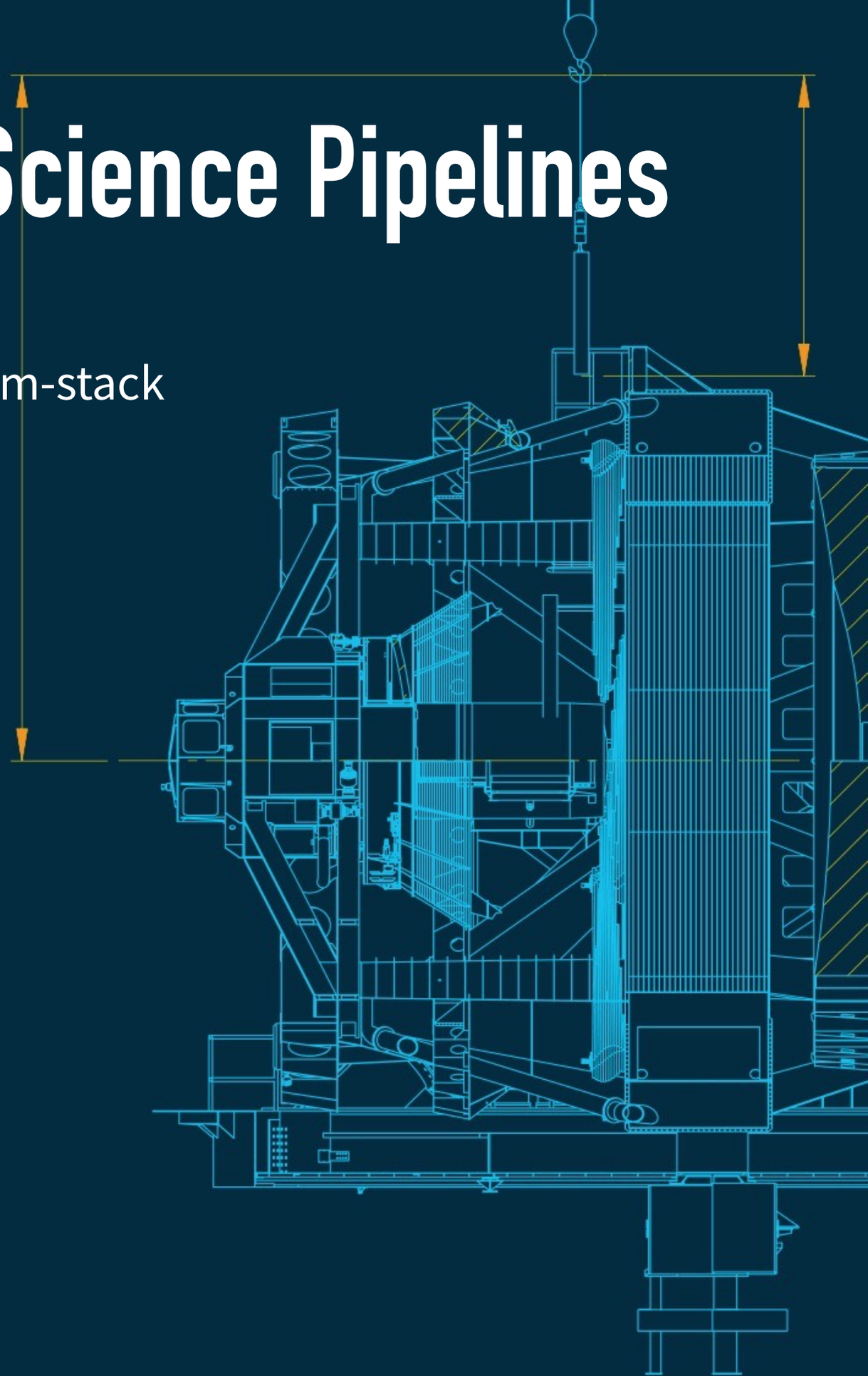
✅ A `~lsst.verify.MeasurementSet` contains `lsst.verify.Measurement` instances.

↳ A `MeasurementSet` contains `Measurement` instances.

See https://developers.google.com/style/lists

Writing user docs for Science Pipelines

https://developer.lsst.io/index.html#part-dm-stack



LSST

Large Synoptic Survey Telescope

Pipelines documentation

The doc/ directory

```
doc
├── SConscript
├── conf.py
├── doxygen.conf.in
├── index.rst
├── lsst.example
│   ├── index.rst
│   ├── scripts
│   └── tasks
└── manifest.yaml
```

https://developer.lsst.io/stack/module-homepage-topic-type.html

https://developer.lsst.io/stack/argparse-script-topic-type.html

https://developer.lsst.io/stack/task-topic-type.html

https://developer.lsst.io/stack/layout-of-doc-directory.html

Pipelines documentation

What type of documentation is it?

Tutorials

- Learning-oriented
- For “newcomers” to get started
- A self-contained lesson that the reader performs
- Provides an example

How–to guides

- Goal-oriented
- Shows how to solve a specific problem
- Series of steps

Concept guides

- Understanding-oriented
- Explains
- Provides background & context

Reference guides

- Information-oriented
- Accurate and complete
- Examples:
  - Numpydoc, Doxygen “docstrings”
  - Task reference pages

These types can be fractal (an example in a API reference).

Often just link from one page to another (we’re on the web).

Credit: https://www.divio.com/en/blog/documentation/

Pipelines documentation

What makes a good documentation page

- **Self-contained**

  No previous page, no next page.

  But lives within the site's web.

- **Specific and limited purpose**

  A topic (i.e., a page) should have a specific, well-defined purpose.

- **Conform to type**

  A type is like a template.

- **Link richly**

  Let the reader forage for information. Think of Wikipedia.

- **Establish context**

  Use the first paragraph to establish the topic's purpose and its relationships to other topics.

- **Assume the reader is qualified**

  Link to introductory topics, don't repeat introductory concepts.

- **Stay on one level**

  Don't evolve level of detail like a narrative.

Based on "Every Page is Page One" by Mark Baker

Pipelines documentation

Adding the page

```
doc
├── SConscript
├── conf.py
├── doxygen.conf.in
├── index.rst
├── lsst.example
│   ├── index.rst
│   ├── howto-get-butler-dataset.rst
│   ├── scripts
│   └── tasks
└── manifest.yaml
```

```
.. py:currentmodule:: lsst.example

.. _lsst.example:

############
lsst.example
############

.. _lsst.example-using:

Using lsst.example
==================


.. toctree::
   :maxdepth: 1

   howto-get-butler-dataset
```

Pipelines documentation

Establish context in the first paragraph

```
.. _howto-get-butler-dataset:

##################################
How to get a dataset with the Butler
##################################

With the LSST Science Pipelines, you don't
directly interact with data files, such as FITS.
Instead, datasets are managed by the Butler in a
repository. This page describes how to get a
dataset using the Butler Python class.

<the actual how-to goes here>
```

link to a conceptual overview page

link to an API reference

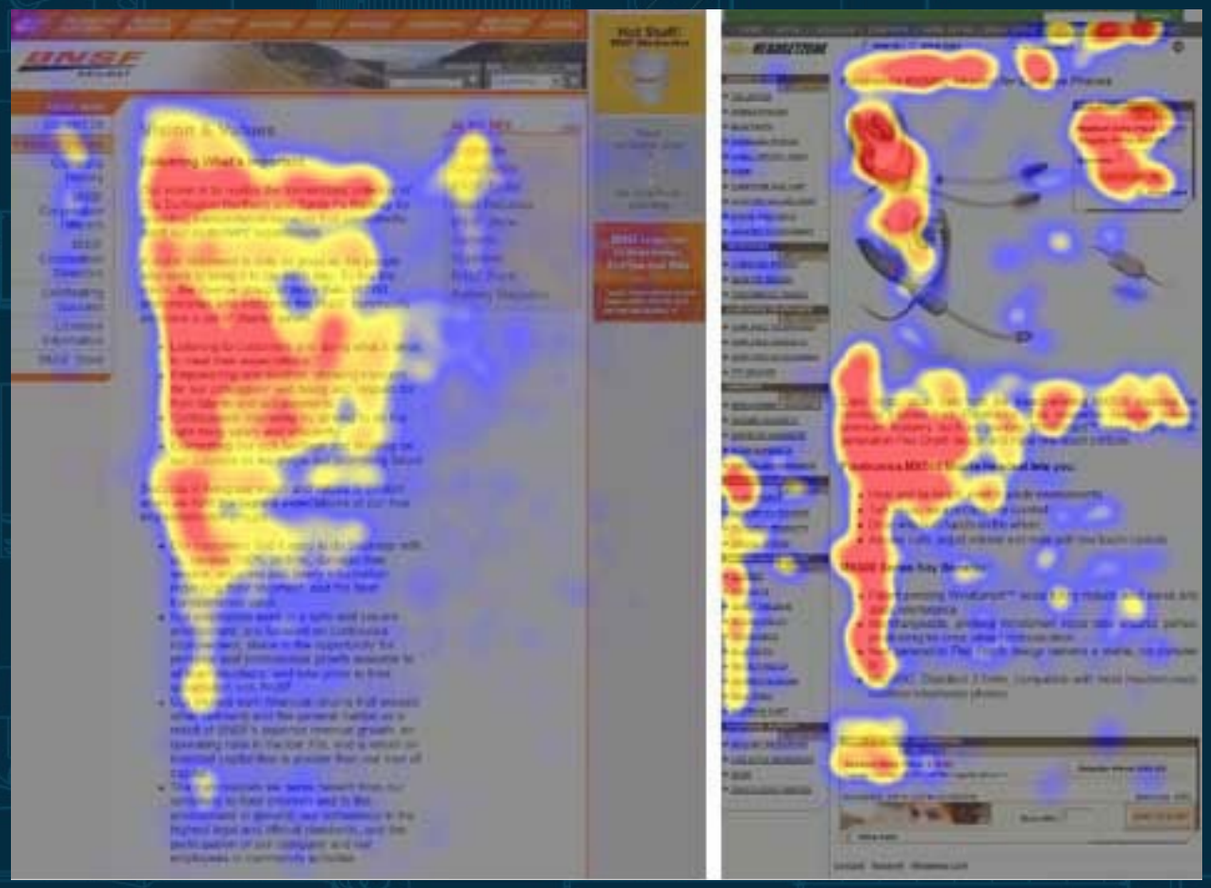
Pipelines documentation

Writing for the web

> Users won't read your text thoroughly in a word-by-word manner.

> The first two paragraphs must state the most important information.

> Start subheads, paragraphs, and bullet points with information-carrying words.



From: F-Shaped Pattern For Reading Web Content by Jakob Nielsen, http://ls.st/fzp.

Pipelines documentation

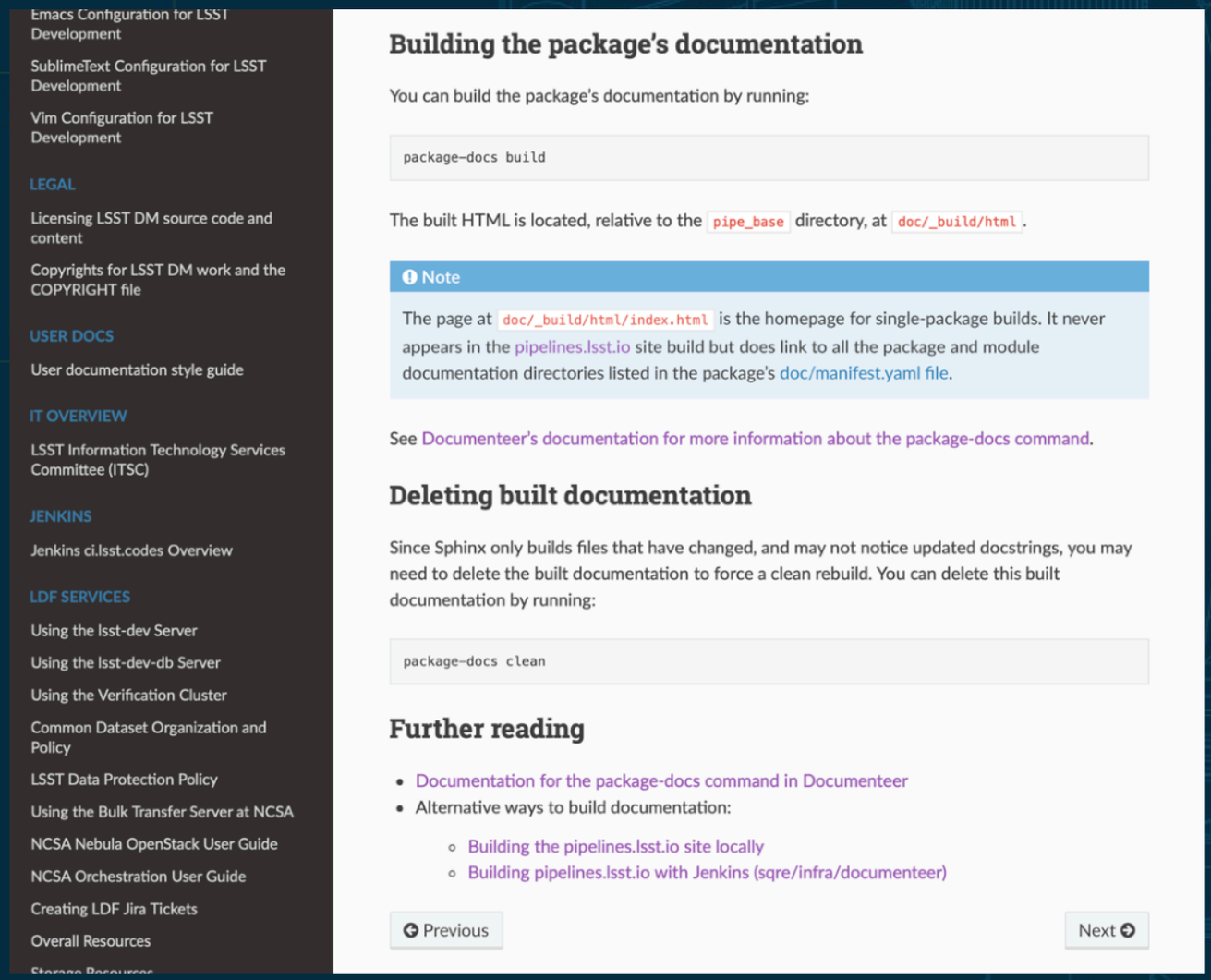
Writing for the web

- Most people don’t read the web sentence-by-sentence.
- Use lots of meaningful headers.
- 2–3 sentence paragraphs.
- First one or two words of a paragraph should convey meaning.
- Use lists and typographic elements for structure.

Pipelines documentation

Building the documentation locally

https://developer.lsst.io/stack/building-single-package-docs.html



Further reading

Project documents

https://www.lsst.io

User documentation style guide

https://developer.lsst.io/user-docs/index.html

ReStructuredText style guide

https://developer.lsst.io/restructuredtext/style.html

Docstrings

https://developer.lsst.io/python/numpydoc.html

Stack developer guide (including writing docs for pipelines.lsst.io)

https://developer.lsst.io/index.html#dm-stack

Writing technotes

https://developer.lsst.io/project-docs/technotes.html

Slack

#dm-docs channel for help writing docs