
regulations Documentation

Release

Author

Sep 01, 2017

1	Django Architecture	3
1.1	Generator	3
1.2	Views	4
1.3	Layers	4
2	regulations package	7
2.1	Subpackages	7
2.1.1	regulations.generator package	7
2.1.1.1	Subpackages	7
2.1.1.2	Submodules	13
2.1.1.3	regulations.generator.api_reader module	13
2.1.1.4	regulations.generator.generator module	13
2.1.1.5	regulations.generator.html_builder module	13
2.1.1.6	regulations.generator.label module	13
2.1.1.7	regulations.generator.link_flattener module	13
2.1.1.8	regulations.generator.node_types module	13
2.1.1.9	regulations.generator.notices module	13
2.1.1.10	regulations.generator.section_url module	13
2.1.1.11	regulations.generator.subterp module	13
2.1.1.12	regulations.generator.title_parsing module	13
2.1.1.13	regulations.generator.toc module	14
2.1.1.14	regulations.generator.versions module	14
2.1.1.15	Module contents	14
2.1.2	regulations.management package	14
2.1.2.1	Subpackages	14
2.1.2.2	Module contents	15
2.1.3	regulations.migrations package	15
2.1.3.1	Submodules	15
2.1.3.2	regulations.migrations.0001_initial module	15
2.1.3.3	regulations.migrations.0002_remove_failedcommentssubmission_files module	15
2.1.3.4	regulations.migrations.0003_delete_failedcommentssubmission module	15
2.1.3.5	Module contents	15
2.1.4	regulations.settings package	15
2.1.4.1	Submodules	15
2.1.4.2	regulations.settings.base module	15
2.1.4.3	regulations.settings.dev module	15

2.1.4.4	regulations.settings.production module	15
2.1.4.5	Module contents	15
2.1.5	regulations.templatetags package	15
2.1.5.1	Submodules	15
2.1.5.2	regulations.templatetags.dash_to_underscore module	15
2.1.5.3	regulations.templatetags.macros module	15
2.1.5.4	regulations.templatetags.render_nested module	15
2.1.5.5	regulations.templatetags.to_list module	15
2.1.5.6	regulations.templatetags.underscore_to_dash module	15
2.1.5.7	Module contents	15
2.1.6	regulations.tests package	15
2.1.6.1	Submodules	15
2.1.6.2	regulations.tests.api_reader_tests module	15
2.1.6.3	regulations.tests.apps_tests module	15
2.1.6.4	regulations.tests.base_template_test module	15
2.1.6.5	regulations.tests.diff_applier_tests module	15
2.1.6.6	regulations.tests.generator_section_url_tests module	16
2.1.6.7	regulations.tests.generator_subterp_tests module	16
2.1.6.8	regulations.tests.generator_tests module	16
2.1.6.9	regulations.tests.generator_toc_tests module	16
2.1.6.10	regulations.tests.generator_versions_tests module	16
2.1.6.11	regulations.tests.html_builder_test module	16
2.1.6.12	regulations.tests.label_tests module	16
2.1.6.13	regulations.tests.layers_appliers_test module	16
2.1.6.14	regulations.tests.layers_definitions_tests module	17
2.1.6.15	regulations.tests.layers_footnotes_tests module	17
2.1.6.16	regulations.tests.layers_formatting_tests module	17
2.1.6.17	regulations.tests.layers_internal_citation_tests module	17
2.1.6.18	regulations.tests.layers_interpretations_tests module	17
2.1.6.19	regulations.tests.layers_location_replace_tests module	17
2.1.6.20	regulations.tests.layers_paragraph_markers_tests module	17
2.1.6.21	regulations.tests.layers_toc_applier_tests module	17
2.1.6.22	regulations.tests.layers_utils_tests module	17
2.1.6.23	regulations.tests.link_flattener_tests module	18
2.1.6.24	regulations.tests.node_types_tests module	18
2.1.6.25	regulations.tests.notices_tests module	18
2.1.6.26	regulations.tests.partial_view_tests module	19
2.1.6.27	regulations.tests.sidebar_analyses_tests module	19
2.1.6.28	regulations.tests.sidebar_help_tests module	19
2.1.6.29	regulations.tests.templatetags_macros_tests module	19
2.1.6.30	regulations.tests.title_parsing_tests module	19
2.1.6.31	regulations.tests.tree_builder_tests module	19
2.1.6.32	regulations.tests.url_cache_tests module	21
2.1.6.33	regulations.tests.urls_test module	21
2.1.6.34	regulations.tests.views_breakaway_tests module	21
2.1.6.35	regulations.tests.views_chrome_tests module	21
2.1.6.36	regulations.tests.views_diff_tests module	21
2.1.6.37	regulations.tests.views_error_tests module	21
2.1.6.38	regulations.tests.views_landing_tests module	21
2.1.6.39	regulations.tests.views_navigation_tests module	21
2.1.6.40	regulations.tests.views_partial_definitions_tests module	21
2.1.6.41	regulations.tests.views_partial_interp_tests module	21
2.1.6.42	regulations.tests.views_partial_search_tests module	21
2.1.6.43	regulations.tests.views_preamble_tests module	21

2.1.6.44	regulations.tests.views_redirect_tests module	21
2.1.6.45	regulations.tests.views_sidebar_tests module	21
2.1.6.46	regulations.tests.views_sxs_tests module	21
2.1.6.47	regulations.tests.views_universal_tests module	21
2.1.6.48	regulations.tests.views_utils_test module	21
2.1.6.49	Module contents	21
2.1.7	regulations.uitests package	21
2.1.7.1	Submodules	21
2.1.7.2	regulations.uitests.base_test module	21
2.1.7.3	regulations.uitests.comment_test module	21
2.1.7.4	regulations.uitests.definition_test module	21
2.1.7.5	regulations.uitests.diff_test module	21
2.1.7.6	regulations.uitests.interp_test module	21
2.1.7.7	regulations.uitests.navigation_test module	21
2.1.7.8	regulations.uitests.scroll_test module	21
2.1.7.9	regulations.uitests.subheader_test module	21
2.1.7.10	regulations.uitests.toc_test module	21
2.1.7.11	regulations.uitests.utils module	21
2.1.7.12	Module contents	23
2.1.8	regulations.views package	23
2.1.8.1	Submodules	23
2.1.8.2	regulations.views.about module	23
2.1.8.3	regulations.views.chrome module	23
2.1.8.4	regulations.views.chrome_breakaway module	23
2.1.8.5	regulations.views.diff module	23
2.1.8.6	regulations.views.error_handling module	23
2.1.8.7	regulations.views.navigation module	23
2.1.8.8	regulations.views.notice_home module	23
2.1.8.9	regulations.views.partial module	23
2.1.8.10	regulations.views.partial_interp module	23
2.1.8.11	regulations.views.partial_search module	23
2.1.8.12	regulations.views.partial_sxs module	23
2.1.8.13	regulations.views.preamble module	23
2.1.8.14	regulations.views.redirect module	23
2.1.8.15	regulations.views.reg_landing module	23
2.1.8.16	regulations.views.sidebar module	23
2.1.8.17	regulations.views.universal_landing module	23
2.1.8.18	regulations.views.utils module	23
2.1.8.19	Module contents	23
2.2	Submodules	23
2.3	regulations.all_urls module	23
2.4	regulations.apps module	23
2.5	regulations.context module	23
2.6	regulations.url_caches module	23
2.7	regulations.urls module	23
2.8	Module contents	23

3 Indices and tables 25

Python Module Index 27

Contents:

Django Architecture

Traditional Django apps contain models to store and retrieve data from a database, templates with which to convert these models into HTML, and thin views to connect the two. Generally, each request loads some subset of the models and shoves them through a template.

Regulations-site differs in some fundamental ways. It is model-less, at least in the Django sense; it loads data from an external API and represents the results as a `dict` (as opposed to converting them into objects). Rather than use a single template per request, the templating layer is used frequently and recursively; single requests may often trigger *dozens* (in some cases, *hundreds*) of templates to be processed. As a result, caching is critical to the application; we buffer AJAX calls in the browser, rendered templates, template file lookup, and API results.

Here, we'll dive into several of these components to get a sense of their general workings as well as history and context which led to their creation. We'll highlight the more abnormal bits, shining light on warts.

Generator

The eRegs UI was originally built as a simple HTML generator, rendering an *entire* regulation. As a result, much of the logic has lived in the `generator` module, which has largely no conception of the HTTP request/response life-cycle. Instead, it is aware of a connection to a backend API, how to associate the types of data served by that API with each other, and how to render the results as HTML.

The `HTMLBuilder` class is king, primarily due to its `process_node()` method, which takes “node” data (i.e. a plain text representation of a regulation, structured as a tree of nested paragraphs) and combines it with “layer” data (i.e. meta/derived data about the tree, such as citations, definitions, etc.) and converts them into HTML. For each node in the tree, layers are applied (see below) in sequence, each successively extending and replacing the node's “`marked_up`” field with HTML corresponding to the layer's updates. Each node (still represented as a `dict`) is also given extra attributes which will be used when rendering the Node in templates. To summarize, the `HTMLBuilder` effectively adorns Nodes with new fields, including one representing the Node's text, as HTML.

Within Django's views, the resulting Node structure is passed off to a template. This time the tree is walked *within the template*, such that each Node is converted into an appropriate chunk of HTML and concatenated with its siblings. Perhaps confusingly, templates are passed the Node data as a “skeleton” of a full regulation – the single section (or whatever component we care about) is “wrapped” with empty Nodes until it looks like a full regulation. This means

that, from the template perspective, there is largely only *one* entry point for views, regardless of whether that view is generating a section, a single paragraph, or an entire regulation. The practice no doubt stems from the original, full-regulation-generation functionality.

There's a tremendous amount of refactoring that should happen here. We shouldn't be walking the tree twice (once within `HTMLBuilder` and once within the templates) – it'd make more sense to remove the former altogether. Further, a conversion from the `dict` to a class would seem appropriate, to make it obvious where to look for functionality. Though the skeleton concept has merit, the hoops it causes us to jump through are rather strange. Perhaps a better solution would be to select an appropriate template automatically based on its type, position in the tree, etc.

Views

There are three primary categories for our views: “sidebar”, “partial”, and “chrome”. The first two stem from our AJAX needs; for browsers with the capability, we AJAX load in content as the user clicks around. The “partial” endpoints correspond to the center content of the page (e.g. a regulation section, search results, the diff view, etc.). When a user clicks to load a new section, their browser will make two AJAX requests, one for the center content and one for the sidebar content.

The “chrome” endpoints wrap these two other types of views with navigation, CSS includes, headers, etc. (i.e. the application's “chrome”). These endpoints are crucial for users without JavaScript (or modern implementations of the URL push API) and for initial loading (e.g. via hard refreshes, bookmarks, etc.).

We currently have far too many *different* views, despite them performing largely the same types of tasks. It would make more sense to combine all of the “node” views into a single class. Similarly, we *mirror* each “partial” view class with a “chrome” class; a more effective strategy would be to have a more generic `wrap_with_chrome` method and no distinct “chrome” classes. This should also remove the incredibly nasty manipulations of Django's request/response life cycle we're currently performing to populate the chrome version. Somewhat related, having a separate endpoint for the sidebar and a separate endpoint for partials didn't turn out as useful as we expected. It probably makes sense to combine them again.

Layers

We have a handful of layer generating classes, which know how to apply data from a layer API on to regulation text. While many of these classes correspond to a *single* data layer, this is not a hard rule. Indeed, we currently have *two* layer classes associated with the definition data – one handles when terms are *defined* while the other handles when they are *used*. As noted above, layers are applied within the `HTMLBuilder` and live inside the `generator` package. Which layers are used depends on the `DATA_LAYERS` setting. Individual requests can also request a subset of these, though that functionality is rarely used.

Layers fall into three categories:

- “inline”, where the layer defines exact text offsets in the Node's text. Internal citations (linking to another paragraph or section within the current regulation) are an example. They have data like:

```
{ "111-22-c": [{ "offsets": [[44, 52], ...], # string index into the text
                # Layer specific fields
                "citation": ["111", "33", "e"] },
  ... ] }
```

- “search-and-replace”, where the layer includes snippets of text (rather than offsets). External citations (linking to content outside of eRegs) are an example. They look like:

```
{
  "111-22-c": [
    {
      "text": "27 CFR Part 478", # exact text match
      "locations": [0, 2, 3], # skips the second reference
      # Layer specific fields
      "citation_type": "CFR",
      "components": {...},
      "url": "http://example.com/...",
    },
    ...
  ],
  ...
}
```

- “paragraph”, where the layer data is scoped to the full paragraph. The table-of-contents layer is an example here. All fields are specific to the individual layer. For example:

```
{
  "111-Subpart-C": [
    {
      "title": "Section 111.22 A Title",
      "index": ["111", "22"],
    },
    ...
  ],
  ...
}
```

The first two categories are needed when we want to modify some component of a Node’s text (e.g. a citation, definition, or formatting adjustment). In these scenarios, the generator provides the original text and the layer data to a corresponding template, which is then responsible for returning appropriate HTML. “Search-and-Replace” is the newer model, offering both better legibility of layer data as well as resiliency to minor errors at the cost of concision.

The “paragraph” layer types return a key and value which will be passed through to the template for rendering a full Node. These are largely used for “meta” data, such as the table of contents, section-wide footnotes, and data which would appear in the sidebar.

The main pain point here is the rather strange way that data is provided; the layer data structure points *into* the tree, spelling out specific chunks of text. An XML or similar structured document format would make much more sense. “Paragraph”-type layers could be attributes of the parent element or meta-data tags.

Subpackages

regulations.generator package

Subpackages

regulations.generator.layers package

Submodules

regulations.generator.layers.base module

class `regulations.generator.layers.base.InlineLayer`

Bases: `regulations.generator.layers.base.LayerBase`

Represents a layer which replaces text by looking at offsets

apply_layer (*text*, *label_id*)

Entry point when processing the regulation tree. Given the node's text and its *label_id*, yield all replacement text

attach_metadata (*node*)

Noop

inline_replacements (*text_index*, *original_text*)

Apply multiple inline layers to given text (e.g. links, highlighting, etc.)

replacement_for (*original*, *data*)

Given the original text and the relevant data from a layer, create a (string) replacement, by, for example, running the data through a template

class `regulations.generator.layers.base.LayerBase`

Bases: `object`

Base class for most layers; each layer contains information which is added on top of the regulation, such as definitions, internal citations, keyterms, etc.

attach_metadata (*node*)

Attach metadata to the provided node

data_source

Data is pulled from the API; this field indicates the name of the endpoint to pull data from

inline_replacements (*text_index, original_text*)

Return triplets of (original text, replacement text, offsets)

shorthand

A short description for this layer. This is used in query strings and the like to define which layers should be used

class `regulations.generator.layers.base.ParagraphLayer`

Bases: `regulations.generator.layers.base.LayerBase`

Represents a layer which applies meta data to nodes

inline_replacements (*text_index, original_text*)

Noop

class `regulations.generator.layers.base.Replacement` (*original, replacement, locations*)

Bases: `tuple`

locations

Alias for field number 2

original

Alias for field number 0

replacement

Alias for field number 1

class `regulations.generator.layers.base.SearchReplaceLayer`

Bases: `regulations.generator.layers.base.LayerBase`

Represents a layer which replaces text by searching for and replacing a specific substring. Also accounts for the string appearing multiple times (via the 'locations' field)

attach_metadata (*node*)

Noop

inline_replacements (*text_index, original_text*)

Entry point when processing the regulation tree. Given the node's `label_id`, attempt to find relevant layer data in `self.layer`

replacements_for (*text, data*)

Given the original text and the relevant data from a layer, create a (string) replacement, by, for example, running the data through a template. Returns a generator

regulations.generator.layers.defined module

regulations.generator.layers.definitions module

regulations.generator.layers.diff_applier module

class `regulations.generator.layers.diff_applier.DiffApplier` (*diff_json*, *label_requested*)

Bases: `object`

Diffs between two versions of a regulation are represented in our particular JSON format. This class applies that diff to the older version of the regulation, generating HTML that clearly shows the changes between old and new.

ADDED_OP = 'added'

DELETE = u'delete'

DELETED_OP = 'deleted'

EQUAL = u'equal'

INSERT = u'insert'

MODIFIED_OP = 'modified'

add_all (*text*)

Mark all the text passed in as deleted.

add_nodes_to_tree (*original*, *adds*)

Add all the nodes from new_nodes into the original tree.

apply_diff (*original*, *label*, *component='text'*)

Here we delete or add whole nodes in addition to passing to *apply_diff_changes* when text has been modified

apply_diff_changes (*original*, *diff_list*)

Account for modified text

deconstruct_text (*original*)

delete_all (*text*)

Mark all the text passed in as deleted.

delete_text (*start*, *end*)

get_text ()

classmethod has_moved (*label_op*, *seen_count*)

A label is moved if it's been deleted in one position but added int another

insert_text (*pos*, *new_text*)

is_child_of_requested (*label*)

Return true if the label is a child of the requested label.

relevant_added (*label*)

Get the operations that add nodes, for the requested section/pargraph.

remove_moved_labels (*label_ops*)

If a label has been moved, we will display it in the new position

set_child_labels (*node*)

As we display removed, added, and unchanged nodes, the children of a node will contain all three types. Pull the 'child_ops' data to derive the correct order of these combined children

tree_changes (*original_tree*)

Apply additions to the regulation tree.

regulations.generator.layers.external_citation module

regulations.generator.layers.footnotes module

class `regulations.generator.layers.footnotes.FootnotesLayer` (*layer, version=None*)

Bases: `regulations.generator.layers.base.ParagraphLayer`

Assembles the footnotes for this node, if available

attach_metadata (*node*)

Return a tuple of ‘footnotes’ and collection of footnotes. Footnotes are “collected” from the node and its children. .. note:

This does **not** handle the case where the same note reference **is** used **in** multiple children.

data_source = ‘formatting’

shorthand = ‘footnotes’

regulations.generator.layers.formatting module

regulations.generator.layers.graphics module

regulations.generator.layers.internal_citation module

regulations.generator.layers.interpretations module

regulations.generator.layers.key_terms module

regulations.generator.layers.layers_applier module

class `regulations.generator.layers.layers_applier.LayersApplier`

Bases: `object`

Most layers replace content. We try to do this intelligently here, so that layers don’t step over each other.

HTML_TAG_REGEX = `<_sre.SRE_Pattern object>`

apply_layers (*original_text*)

enqueue (*layer_element*)

enqueue_from_list (*elements_list*)

location_replace (*xml_node, original, replacement, locations*)

replace_all (*original, replacement*)

Replace all occurrences of original with replacement. This is HTML aware; it effectively looks at all of the text in between HTML tags

replace_at (*original, replacement, locations*)

Replace the occurrences of original at all the locations with replacement.

unescape_text ()

Because of the way we do `replace_all()`, we need to unescape HTML entities.

regulations.generator.layers.location_replace module

class `regulations.generator.layers.location_replace.LocationReplace`

Bases: `object`

Applies location based layers to XML nodes. We use XML so that we only take into account the original text when we're doing a replacement.

static `find_all_offsets` (*pattern, text, offset=0*)

Don't use regular expressions as they are a tad slow

location_replace (*xml_node, original, replacement, locations*)

For the *xml_node*, replace the *locations* instances of *original* with *replacement*. @todo: This doesn't appear to be used anymore?

location_replace_text (*text, original, replacement, locations*)

Given plain text, do replacements

update_offset_starter ()

As we're navigating the XML node, we need to keep track of how many offsets we've already seen.

update_offsets (*original, text*)

Offsets change everytime we replace the text, since we add more characters. Update the offsets.

regulations.generator.layers.meta module

class `regulations.generator.layers.meta.MetaLayer` (*layer_data*)

Bases: `regulations.generator.layers.base.ParagraphLayer`

attach_metadata (*node*)

Return a pair of field-name (meta) + the layer data

data_source = 'meta'

shorthand = 'meta'

regulations.generator.layers.paragraph_markers module

regulations.generator.layers.toc_applier module

regulations.generator.layers.tree_builder module

class `regulations.generator.layers.tree_builder.AddQueue`

Bases: `object`

Maintain a sorted list of nodes to add. This maintains a sorted queue of (label, node) tuples.

delete (*label*)

find (*label*)

insert (*item*)

insert_all (*items*)

sort ()

`regulations.generator.layers.tree_builder.add_child` (*parent_node, node*)

Add a child node to a parent, maintaining the order of the children.

`regulations.generator.layers.tree_builder.add_node_to_tree` (*node*, *parent_label*, *tree_hash*)

Add the node to the tree by adding it to its parent in order.

`regulations.generator.layers.tree_builder.all_children_are_roman` (*parent_node*)

Return true if all the children of the parent node have roman labels

`regulations.generator.layers.tree_builder.build_label` (*node*)

`regulations.generator.layers.tree_builder.build_tree_hash` (*tree*)

Build a hash map of a tree's nodes, so that we don't have to keep walking the tree.

`regulations.generator.layers.tree_builder.make_label_sortable` (*label*, *roman=False*)

Make labels sortable, but converting them as appropriate. Also, appendices have labels that look like 30(a), we make those appropriately sortable.

`regulations.generator.layers.tree_builder.parent_in_tree` (*parent_label*, *tree_hash*)

Return True if the parent of *node_label* is in the tree

`regulations.generator.layers.tree_builder.parent_label` (*node*)

This is not perfect. It can not handle children of subparts, for example

`regulations.generator.layers.tree_builder.roman_nums` ()

Generator for roman numerals.

regulations.generator.layers.utils module

`regulations.generator.layers.utils.convert_to_python` (*data*)

Convert raw data (e.g. from json conversion) into the appropriate Python objects

`regulations.generator.layers.utils.is_contained_in` (*child*, *parent*)

Return True if *child* is a child node of the *parent*.

Node labels are hierarchical paths, with segments separated by '-'. As an edge case, a node label is also a child of itself.

`regulations.generator.layers.utils.render_template` (*template*, *context*)

Module contents

regulations.generator.sidebar package

Submodules

regulations.generator.sidebar.analyses module

regulations.generator.sidebar.base module

regulations.generator.sidebar.diff_help module

regulations.generator.sidebar.help module

Module contents

Submodules

`regulations.generator.api_reader` module

`regulations.generator.generator` module

`regulations.generator.html_builder` module

`regulations.generator.label` module

`regulations.generator.link_flattener` module

`regulations.generator.link_flattener.flatten_links` (*text*)

Fix `<a>` elements that have embedded `<a>` elements by replacing the internal `<a>` element with its content.

`regulations.generator.node_types` module

`regulations.generator.node_types.label_to_text` (*label*, *include_section=True*, *include_marker=False*)

Convert a `label:list[string]` into a human-readable string

`regulations.generator.node_types.take_until_markerless` (*label_parts*)

`regulations.generator.node_types.to_markup_id` (*id_parts*)

Given the id parts from the JSON tree, convert to an id that can be used in the front end

`regulations.generator.node_types.type_from_label` (*label*)

Given a list of label parts, determine the associated node's type

`regulations.generator.notices` module

`regulations.generator.notices.add_depths` (*sxs*, *starting_depth*)

We use depth numbers in header tags to determine how titles are output.

`regulations.generator.notices.filter_labeled_children` (*sxs*)

Some children don't have labels. We display those with their parents. The other children are displayed when they are independently, specifically requested.

`regulations.generator.notices.find_label_in_sxs` (*sxs_list*, *label_id*, *fr_page=None*)

Given a tree of SXS sections, find a non-empty `sxs` that matches `label_id`. Some notices may have the same label appearing multiple times; use `fr_page` to distinguish, defaulting to the first

`regulations.generator.notices.non_empty_sxs` (*sxs*)

`regulations.generator.section_url` module

`regulations.generator.subterp` module

`regulations.generator.title_parsing` module

`regulations.generator.title_parsing.appendix_supplement` (*data*)

Handle items pointing to an appendix or supplement

`regulations.generator.title_parsing.section` (*data*)

Parse out parts of a section title.

`regulations.generator.title_parsing.try_split` (*text*, *chars*=(*u'\u2014', '-'*))

Utility method for splitting a string by one of multiple chars

regulations.generator.toc module

regulations.generator.versions module

Module contents

regulations.management package

Subpackages

regulations.management.commands package

Submodules

regulations.management.commands.cache_webpages module

regulations.management.commands.compile_frontend module

regulations.management.commands.eregs_cache module

regulations.management.commands.fetch_wkhtmltox module

Module contents

Module contents

regulations.migrations package

Submodules

regulations.migrations.0001_initial module

regulations.migrations.0002_remove_failedcommentsubmission_files module

regulations.migrations.0003_delete_failedcommentsubmission module

Module contents

regulations.settings package

Submodules

regulations.settings.base module

regulations.settings.dev module

regulations.settings.production module

Module contents

regulations.templatetags package

Submodules

regulations.templatetags.dash_to_underscore module

regulations.templatetags.macros module

regulations.templatetags.render_nested module

regulations.templatetags.to_list module

regulations.templatetags.underscore_to_dash module

Module contents

regulations.tests package

Submodules

regulations.tests.api_reader_tests module

regulations.tests.apps_tests module

regulations.tests.base_template_test module

2.1. Subpackages

regulations.tests.diff_applier_tests module

```
class regulations.tests.diff_applier_tests.DiffApplierTest (methodName='runTest')
    Bases: unittest.case.TestCase
```

```
build_tree()
create_diff_applier()
test_add_nodes_child_ops()
    If we don't know the correct order of children, attempt to use data from child_ops
test_add_nodes_empty_tree()
test_add_nodes_new_section()
test_add_nodes_to_tree()
test_apply_diff()
test_apply_diff_title()
test_child_picking()
test_create_applier()
test_deconstruct_text()
test_delete_all()
test_delete_text()
test_insert_text()
test_insert_text_at_end()
test_set_child_labels_reorder()
    Nodes which have been _moved_ should be ordered in their final resting position
test_tree_changes_new_section()
```

regulations.tests.generator_section_url_tests module

regulations.tests.generator_subterp_tests module

regulations.tests.generator_tests module

regulations.tests.generator_toc_tests module

regulations.tests.generator_versions_tests module

regulations.tests.html_builder_test module

regulations.tests.label_tests module

regulations.tests.layers_appliers_test module

```
class regulations.tests.layers_appliers_test.LayersApplierTest (methodName='runTest')
    Bases: unittest.case.TestCase
    test_enqueue()
    test_find_all_offsets()
    test_find_offsets_no_pattern()
    test_list_enqueue()
```

```

test_replace_all()
test_replace_at()
test_replace_at_case_sensitive()
test_replace_no_original()
test_replace_skip_location()
test_update_offset_starter()
test_update_offsets()

```

regulations.tests.layers_definitions_tests module

regulations.tests.layers_footnotes_tests module

```

class regulations.tests.layers_footnotes_tests.FootnotesLayerTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_multiple_children()
        test_single_note()
        test_sorted_multiple_notes()

```

regulations.tests.layers_formatting_tests module

regulations.tests.layers_internal_citation_tests module

regulations.tests.layers_interpretations_tests module

regulations.tests.layers_location_replace_tests module

```

class regulations.tests.layers_location_replace_tests.LayersLocationReplaceTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_location_replace_text()
        test_update_offsets_html()

```

regulations.tests.layers_paragraph_markers_tests module

regulations.tests.layers_toc_applier_tests module

regulations.tests.layers_utils_tests module

```

class regulations.tests.layers_utils_tests.LayerUtilsTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_convert_to_python()
        test_is_contained_in()

```

regulations.tests.link_flattener_tests module

```
class regulations.tests.link_flattener_tests.LinkFlattenerTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_embedded_link()
        test_multiple_level_embedded_links()
        test_multiple_serial_embedded_links()
        test_no_links()
        test_real_world_example()
        test_single_link()
        test_unembedded_links()
```

regulations.tests.node_types_tests module

```
class regulations.tests.node_types_tests.NodeTypesTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_change_appendix()
        test_label_to_text()
        test_type_from_label()
```

regulations.tests.notices_tests module

```
class regulations.tests.notices_tests.NoticesTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_add_depths()
        test_filter_children()
        test_filter_children_no_candidates()
        test_find_label_in_sxs_found()
        test_find_label_in_sxs_not_found()
        test_find_label_in_sxs_page()
        test_find_label_in_sxs_top_no_label()
        test_non_empty_sxs()
        test_non_empty_sxs_has_children()
        test_non_empty_sxs_no_paragraph()
```


`regulations.tests.partial_view_tests` module

`regulations.tests.sidebar_analyses_tests` module

`regulations.tests.sidebar_help_tests` module

`regulations.tests.templatetags_macros_tests` module

`regulations.tests.title_parsing_tests` module

```
class regulations.tests.title_parsing_tests.RegTest (methodName='runTest')
    Bases: unittest.case.TestCase
        test_appendix_supplement_ap()
        test_section()
        test_try_split()
```

`regulations.tests.tree_builder_tests` module

```
class regulations.tests.tree_builder_tests.TreeBuilderTest (methodName='runTest')
    Bases: unittest.case.TestCase
        build_tree()
        test_add_child()
        test_add_child_appendix()
        test_add_child_interp()
        test_add_child_odd_sort()
            Appendices may have some strange orderings. Make sure they keep order.
        test_add_child_root_appendix()
            Let's add an introductory paragraph child to a root interpretation node and ensure that the children are
            sorted correctly.
        test_add_child_root_interp()
            Let's add an introductory paragraph child to a root interpretation node and ensure that the children are
            sorted correctly.
        test_add_node()
        test_all_children_are_roman()
        test_build_tree_hash()
        test_make_label_sortable_not_roman()
        test_make_label_sortable_roman()
        test_parent_in_tree()
        test_parent_label()
        test_roman_nums()
```


regulations.tests.url_cache_tests module

regulations.tests.urls_test module

regulations.tests.views_breakaway_tests module

regulations.tests.views_chrome_tests module

regulations.tests.views_diff_tests module

regulations.tests.views_error_tests module

regulations.tests.views_landing_tests module

regulations.tests.views_navigation_tests module

regulations.tests.views_partial_definitions_tests module

regulations.tests.views_partial_interp_tests module

regulations.tests.views_partial_search_tests module

regulations.tests.views_preamble_tests module

regulations.tests.views_redirect_tests module

regulations.tests.views_sidebar_tests module

regulations.tests.views_sxs_tests module

regulations.tests.views_universal_tests module

regulations.tests.views_utils_test module

Module contents

regulations.uitests package

Submodules

regulations.uitests.base_test module

regulations.uitests.comment_test module

regulations.uitests.definition_test module

regulations.uitests.diff_test module

regulations.uitests.interp_test module

regulations.uitests.navigation_test module

regulations.uitests.scroll_test module

regulations.uitests.subheader_test module

2.1. Subpackages

regulations.uitests.toc_test module

regulations.uitests.utils module

Module contents

regulations.views package

Submodules

regulations.views.about module

regulations.views.chrome module

regulations.views.chrome_breakaway module

regulations.views.diff module

regulations.views.error_handling module

regulations.views.navigation module

regulations.views.notice_home module

regulations.views.partial module

regulations.views.partial_interp module

regulations.views.partial_search module

regulations.views.partial_sxs module

regulations.views.preamble module

regulations.views.redirect module

regulations.views.reg_landing module

regulations.views.sidebar module

regulations.views.universal_landing module

regulations.views.utils module

Module contents

Submodules

regulations.all_urls module

regulations.apps module

regulations.context module

regulations.url_caches module

2.2. Submodules

regulations.urls module

CHAPTER 3

Indices and tables

- `genindex`
- `modindex`
- `search`

r

- regulations, [23](#)
- regulations.generator, [14](#)
- regulations.generator.layers, [12](#)
- regulations.generator.layers.base, [7](#)
- regulations.generator.layers.diff_applier,
[9](#)
- regulations.generator.layers.footnotes,
[10](#)
- regulations.generator.layers.layers_applier,
[10](#)
- regulations.generator.layers.location_replace,
[11](#)
- regulations.generator.layers.meta, [11](#)
- regulations.generator.layers.tree_builder,
[11](#)
- regulations.generator.layers.utils, [12](#)
- regulations.generator.link_flattener,
[13](#)
- regulations.generator.node_types, [13](#)
- regulations.generator.notices, [13](#)
- regulations.generator.sidebar, [13](#)
- regulations.generator.title_parsing, [13](#)
- regulations.management, [15](#)
- regulations.management.commands, [15](#)
- regulations.migrations, [15](#)
- regulations.settings, [15](#)
- regulations.templatetags, [15](#)
- regulations.tests, [21](#)
- regulations.tests.diff_applier_tests,
[15](#)
- regulations.tests.layers_appliers_test,
[16](#)
- regulations.tests.layers_footnotes_tests,
[17](#)
- regulations.tests.layers_location_replace_tests,
[17](#)
- regulations.tests.layers_utils_tests,
[17](#)
- regulations.tests.link_flattener_tests,
[18](#)
- regulations.tests.node_types_tests, [18](#)
- regulations.tests.notices_tests, [18](#)
- regulations.tests.title_parsing_tests,
[19](#)
- regulations.tests.tree_builder_tests,
[19](#)
- regulations.uitests, [23](#)
- regulations.uitests.utils, [21](#)
- regulations.views, [23](#)

A

[add_all\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9
[add_child\(\)](#) (in module regulations.generator.layers.tree_builder), 11
[add_depths\(\)](#) (in module regulations.generator.notices), 13
[add_node_to_tree\(\)](#) (in module regulations.generator.layers.tree_builder), 11
[add_nodes_to_tree\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9
[ADDED_OP](#) (regulations.generator.layers.diff_applier.DiffApplier attribute), 9
[AddQueue](#) (class in regulations.generator.layers.tree_builder), 11
[all_children_are_roman\(\)](#) (in module regulations.generator.layers.tree_builder), 12
[appendix_supplement\(\)](#) (in module regulations.generator.title_parsing), 13
[apply_diff\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9
[apply_diff_changes\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9
[apply_layer\(\)](#) (regulations.generator.layers.base.InlineLayer method), 7
[apply_layers\(\)](#) (regulations.generator.layers.layers_applier.LayersApplier method), 10
[attach_metadata\(\)](#) (regulations.generator.layers.base.InlineLayer method), 7
[attach_metadata\(\)](#) (regulations.generator.layers.base.LayerBase method), 8
[attach_metadata\(\)](#) (regulations.generator.layers.base.SearchReplaceLayer method), 8
[attach_metadata\(\)](#) (regulations.generator.layers.base.SearchReplaceLayer method), 8
[attach_metadata\(\)](#) (regulations.generator.layers.base.SearchReplaceLayer method), 8

[attach_metadata\(\)](#) (regulations.generator.layers.base.SearchReplaceLayer method), 10
[attach_metadata\(\)](#) (regulations.generator.layers.meta.MetaLayer method), 11

B

[build_label\(\)](#) (in module regulations.generator.layers.tree_builder), 12
[build_tree\(\)](#) (regulations.tests.diff_applier_tests.DiffApplierTest method), 15
[build_tree\(\)](#) (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19
[build_tree_hash\(\)](#) (in module regulations.generator.layers.tree_builder), 12

C

[convert_to_python\(\)](#) (in module regulations.generator.layers.utils), 12
[create_diff_applier\(\)](#) (regulations.tests.diff_applier_tests.DiffApplierTest method), 16

D

[data_source](#) (regulations.generator.layers.base.LayerBase attribute), 8
[data_source](#) (regulations.generator.layers.footnotes.FootnotesLayer attribute), 10
[data_source](#) (regulations.generator.layers.meta.MetaLayer attribute), 11
[deconstruct_text\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9
[DELETE](#) (regulations.generator.layers.diff_applier.DiffApplier attribute), 9
[delete\(\)](#) (regulations.generator.layers.tree_builder.AddQueue method), 11
[delete_all\(\)](#) (regulations.generator.layers.diff_applier.DiffApplier method), 9

delete_text() (regulations.generator.layers.diff_applier.DiffApplier method), 9

DELETED_OP (regulations.generator.layers.diff_applier.DiffApplier attribute), 9

DiffApplier (class in regulations.generator.layers.diff_applier), 9

DiffApplierTest (class in regulations.tests.diff_applier_tests), 15

E

enqueue() (regulations.generator.layers.layers_applier.LayersApplier method), 10

enqueue_from_list() (regulations.generator.layers.layers_applier.LayersApplier method), 10

EQUAL (regulations.generator.layers.diff_applier.DiffApplier attribute), 9

F

filter_labeled_children() (in module regulations.generator.notices), 13

find() (regulations.generator.layers.tree_builder.AddQueue method), 11

find_all_offsets() (regulations.generator.layers.location_replace.LocationReplace static method), 11

find_label_in_sxs() (in module regulations.generator.notices), 13

flatten_links() (in module regulations.generator.link_flattener), 13

FootnotesLayer (class in regulations.generator.layers.footnotes), 10

FootnotesLayerTest (class in regulations.tests.layers_footnotes_tests), 17

G

get_text() (regulations.generator.layers.diff_applier.DiffApplier method), 9

H

has_moved() (regulations.generator.layers.diff_applier.DiffApplier class method), 9

HTML_TAG_REGEX (regulations.generator.layers.layers_applier.LayersApplier attribute), 10

I

inline_replacements() (regulations.generator.layers.base.InlineLayer method), 7

inline_replacements() (regulations.generator.layers.base.LayerBase method), 8

inline_replacements() (regulations.generator.layers.base.ParagraphLayer method), 8

inline_replacements() (regulations.generator.layers.base.SearchReplaceLayer method), 8

InlineLayer (class in regulations.generator.layers.base), 7

INSERT (regulations.generator.layers.diff_applier.DiffApplier attribute), 9

insert() (regulations.generator.layers.tree_builder.AddQueue method), 11

insert_text() (regulations.generator.layers.diff_applier.DiffApplier method), 9

is_child_of_requested() (regulations.generator.layers.diff_applier.DiffApplier method), 9

is_contained_in() (in module regulations.generator.layers.utils), 12

L

label_to_text() (in module regulations.generator.node_types), 13

LayerBase (class in regulations.generator.layers.base), 7

LayersApplier (class in regulations.generator.layers.layers_applier), 10

LayersApplierTest (class in regulations.tests.layers_appliers_test), 16

LayersLocationReplaceTest (class in regulations.tests.layers_location_replace_tests), 17

LayerUtilsTest (class in regulations.tests.layers_utils_tests), 17

LinkFlattenerTest (class in regulations.tests.link_flattener_tests), 18

location_replace() (regulations.generator.layers.layers_applier.LayersApplier method), 10

location_replace() (regulations.generator.layers.location_replace.LocationReplace method), 11

location_replace_text() (regulations.generator.layers.location_replace.LocationReplace method), 11

LocationReplace (class in regulations.generator.layers.location_replace), 11

locations (regulations.generator.layers.base.Replacement attribute), 8

M

make_label_sortable() (in module regulations.generator.layers.tree_builder), 12

MetaLayer (class in regulations.generator.layers.meta), 11

MODIFIED_OP (regulations.generator.layers.diff_applier.DiffApplier attribute), 9

N

NodeTypesTest (class in regulations.tests.node_types_tests), 18

non_empty_sxs() (in module regulations.generator.notices), 13

NoticesTest (class in regulations.tests.notices_tests), 18

O

original (regulations.generator.layers.base.Replacement attribute), 8

P

ParagraphLayer (class in regulations.generator.layers.base), 8

parent_in_tree() (in module regulations.generator.layers.tree_builder), 12

parent_label() (in module regulations.generator.layers.tree_builder), 12

R

RegTest (class in regulations.tests.title_parsing_tests), 19

regulations (module), 23

regulations.generator (module), 14

regulations.generator.layers (module), 12

regulations.generator.layers.base (module), 7

regulations.generator.layers.diff_applier (module), 9

regulations.generator.layers.footnotes (module), 10

regulations.generator.layers.layers_applier (module), 10

regulations.generator.layers.location_replace (module), 11

regulations.generator.layers.meta (module), 11

regulations.generator.layers.tree_builder (module), 11

regulations.generator.layers.utils (module), 12

regulations.generator.link_flattener (module), 13

regulations.generator.node_types (module), 13

regulations.generator.notices (module), 13

regulations.generator.sidebar (module), 13

regulations.generator.title_parsing (module), 13

regulations.management (module), 15

regulations.management.commands (module), 15

regulations.migrations (module), 15

regulations.settings (module), 15

regulations.templatetags (module), 15

regulations.tests (module), 21

regulations.tests.diff_applier_tests (module), 15

regulations.tests.layers_appliers_test (module), 16

regulations.tests.layers_footnotes_tests (module), 17

regulations.tests.layers_location_replace_tests (module), 17

regulations.tests.layers_utils_tests (module), 17

regulations.tests.link_flattener_tests (module), 18

regulations.tests.node_types_tests (module), 18

regulations.tests.notices_tests (module), 18

regulations.tests.title_parsing_tests (module), 19

regulations.tests.tree_builder_tests (module), 19

regulations.uitests (module), 23

regulations.uitests.utils (module), 21

regulations.views (module), 23

relevant_added() (regulations.generator.layers.diff_applier.DiffApplier method), 9

remove_moved_labels() (regulations.generator.layers.diff_applier.DiffApplier method), 9

render_template() (in module regulations.generator.layers.utils), 12

replace_all() (regulations.generator.layers.layers_applier.LayersApplier method), 10

replace_at() (regulations.generator.layers.layers_applier.LayersApplier method), 10

Replacement (class in regulations.generator.layers.base), 8

replacement (regulations.generator.layers.base.Replacement attribute), 8

replacement_for() (regulations.generator.layers.base.InlineLayer method), 7

replacements_for() (regulations.generator.layers.base.SearchReplaceLayer method), 8

roman_nums() (in module regulations.generator.layers.tree_builder), 12

S

scroll_to() (in module regulations.uitests.utils), 21

SearchReplaceLayer (class in regulations.generator.layers.base), 8

section() (in module regulations.generator.title_parsing), 13

set_child_labels() (regulations.generator.layers.diff_applier.DiffApplier method), 9

shorthand (regulations.generator.layers.base.LayerBase attribute), 8

shorthand (regulations.generator.layers.footnotes.FootnotesLayer attribute), 10

shorthand (regulations.generator.layers.meta.MetaLayer attribute), 11

sort() (regulations.generator.layers.tree_builder.AddQueue method), 11

T

<code>take_until_markerless()</code> (in module <code>regulations.generator.node_types</code>), 13	<code>test_change_appendix()</code> (regulations.tests.node_types_tests.NodeTypesTest method), 18
<code>test_add_child()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_child_picking()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
<code>test_add_child_appendix()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_convert_to_python()</code> (regulations.tests.layers_utils_tests.LayerUtilsTest method), 17
<code>test_add_child_interp()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_create_applier()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
<code>test_add_child_odd_sort()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_deconstruct_text()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
<code>test_add_child_root_appendix()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_delete_all()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
<code>test_add_child_root_interp()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_delete_text()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
<code>test_add_depths()</code> (regulations.tests.notices_tests.NoticesTest method), 18	<code>test_embedded_link()</code> (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18
<code>test_add_node()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_enqueue()</code> (regulations.tests.layers_appliers_test.LayersApplierTest method), 16
<code>test_add_nodes_child_ops()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_filter_children()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_add_nodes_empty_tree()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_filter_children_no_candidates()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_add_nodes_new_section()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_find_all_offsets()</code> (regulations.tests.layers_appliers_test.LayersApplierTest method), 16
<code>test_add_nodes_to_tree()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_find_label_in_sxs_found()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_all_children_are_roman()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_find_label_in_sxs_not_found()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_appendix_supplement_ap()</code> (regulations.tests.title_parsing_tests.RegTest method), 19	<code>test_find_label_in_sxs_page()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_apply_diff()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_find_label_in_sxs_top_no_label()</code> (regulations.tests.notices_tests.NoticesTest method), 18
<code>test_apply_diff_title()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16	<code>test_find_offsets_no_pattern()</code> (regulations.tests.layers_appliers_test.LayersApplierTest method), 16
<code>test_build_tree_hash()</code> (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	<code>test_insert_text()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
	<code>test_insert_text_at_end()</code> (regulations.tests.diff_applier_tests.DiffApplierTest method), 16

tions.tests.diff_applier_tests.DiffApplierTest method), 16	tions.tests.layers_appliers_test.LayersApplierTest method), 17
test_is_contained_in() (regulations.tests.layers_utils_tests.LayerUtilsTest method), 17	test_replace_at_case_sensitive() (regulations.tests.layers_appliers_test.LayersApplierTest method), 17
test_label_to_text() (regulations.tests.node_types_tests.NodeTypesTest method), 18	test_replace_no_original() (regulations.tests.layers_appliers_test.LayersApplierTest method), 17
test_list_enqueue() (regulations.tests.layers_appliers_test.LayersApplierTest method), 16	test_replace_skip_location() (regulations.tests.layers_appliers_test.LayersApplierTest method), 17
test_location_replace_text() (regulations.tests.layers_location_replace_tests.LayersLocationReplaceTest method), 17	test_roman_nums() (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19
test_make_label_sortable_not_roman() (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	test_section() (regulations.tests.title_parsing_tests.RegTest method), 19
test_make_label_sortable_roman() (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	test_set_child_labels_reorder() (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
test_multiple_children() (regulations.tests.layers_footnotes_tests.FootnotesLayerTest method), 17	test_single_link() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18
test_multiple_level_embedded_links() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18	test_single_note() (regulations.tests.layers_footnotes_tests.FootnotesLayerTest method), 17
test_multiple_serial_embedded_links() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18	test_sorted_multiple_notes() (regulations.tests.layers_footnotes_tests.FootnotesLayerTest method), 17
test_no_links() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18	test_tree_changes_new_section() (regulations.tests.diff_applier_tests.DiffApplierTest method), 16
test_non_empty_sxs() (regulations.tests.notices_tests.NoticesTest method), 18	test_try_split() (regulations.tests.title_parsing_tests.RegTest method), 19
test_non_empty_sxs_has_children() (regulations.tests.notices_tests.NoticesTest method), 18	test_type_from_label() (regulations.tests.node_types_tests.NodeTypesTest method), 18
test_non_empty_sxs_no_paragraph() (regulations.tests.notices_tests.NoticesTest method), 18	test_unembedded_links() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18
test_parent_in_tree() (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	test_update_offset_starter() (regulations.tests.layers_appliers_test.LayersApplierTest method), 17
test_parent_label() (regulations.tests.tree_builder_tests.TreeBuilderTest method), 19	test_update_offsets() (regulations.tests.layers_appliers_test.LayersApplierTest method), 17
test_real_world_example() (regulations.tests.link_flattener_tests.LinkFlattenerTest method), 18	test_update_offsets_html() (regulations.tests.layers_location_replace_tests.LayersLocationReplaceTest method), 17
test_replace_all() (regulations.tests.layers_appliers_test.LayersApplierTest method), 16	to_markup_id() (in module regulations.generator.node_types), 13
test_replace_at() (regulations.tests.layers_appliers_test.LayersApplierTest method), 16	tree_changes() (regulations.generator.layers.diff_applier.DiffApplierTreeBuilderTest method), 9
	TreeBuilderTest (class in regulations.tests.tree_builder_tests), 19

`try_split()` (in module `regulations.generator.title_parsing`),
[14](#)
`type_from_label()` (in module `regulations.generator.node_types`), [13](#)

U

`unescape_text()` (`regulations.generator.layers.layers_applier.LayersApplier`
method), [10](#)
`update_offset_starter()` (`regulations.generator.layers.location_replace.LocationReplace`
method), [11](#)
`update_offsets()` (`regulations.generator.layers.location_replace.LocationReplace`
method), [11](#)