
pypuppetdbquery Documentation

Release 0.9.0

Chris Boot

June 24, 2016

1	pypuppetdbquery package	3
1.1	Submodules	3
1.1.1	pypuppetdbquery.ast module	3
1.1.2	pypuppetdbquery.evaluator module	4
1.1.3	pypuppetdbquery.lexer module	5
1.1.4	pypuppetdbquery.parser module	6
2	Examples	11
3	Test Suite	13
4	Indices and tables	17
	Python Module Index	19

Contents:

pypuppetdbquery package

`pypuppetdbquery.parse(s, json=True, lex_options=None, yacc_options=None)`

Parse a PuppetDBQuery-style query and transform it into a PuppetDB “AST” query.

This function is intended to be the primary entry point for this package. It wraps up all the various components of this package into an easy to consume format. The output of this function is designed to be passed directly into `pypuppetdb`.

For examples of the query syntax see [puppet-puppetdbquery](#) and [node-puppetdbquery](#) by Erik Dalén.

Parameters

- **s** (*str*) – The query to parse and transform
- **json** (*bool*) – Whether to JSON-encode the PuppetDB AST result
- **lex_options** (*dict*) – Options passed to `ply.lex.lex()`
- **yacc_options** (*dict*) – Options passed to `ply.yacc.yacc()`

1.1 Submodules

1.1.1 pypuppetdbquery.ast module

Abstract Syntax Tree (AST) for the PuppetDBQuery language. These simple classes are used by the `pypuppetdbquery.parser.Parser` in order to represent the parsed syntax tree.

class `pypuppetdbquery.ast.AndExpression(left, right)`

Bases: `pypuppetdbquery.ast.BinaryExpression`

class `pypuppetdbquery.ast.BinaryExpression(left, right)`

Bases: `pypuppetdbquery.ast.Node`

class `pypuppetdbquery.ast.BlockExpression(expression)`

Bases: `pypuppetdbquery.ast.UnaryExpression`

class `pypuppetdbquery.ast.Comparison(operator, left, right)`

Bases: `pypuppetdbquery.ast.Expression`

class `pypuppetdbquery.ast.Date(value)`

Bases: `pypuppetdbquery.ast.Literal`

class `pypuppetdbquery.ast.Expression`

Bases: `pypuppetdbquery.ast.Node`

```
class pypuppetdbquery.ast.Identifier(name)
    Bases: pypuppetdbquery.ast.Node

class pypuppetdbquery.ast.IdentifierPath(components)
    Bases: pypuppetdbquery.ast.Node

class pypuppetdbquery.ast.Literal(value)
    Bases: pypuppetdbquery.ast.Node

class pypuppetdbquery.ast.Node
    Bases: object

class pypuppetdbquery.ast.NotExpression(expression)
    Bases: pypuppetdbquery.ast.UnaryExpression

class pypuppetdbquery.ast.OrExpression(left, right)
    Bases: pypuppetdbquery.ast.BinaryExpression

class pypuppetdbquery.ast.ParenthesizedExpression(expression)
    Bases: pypuppetdbquery.ast.UnaryExpression

class pypuppetdbquery.ast.Query(expression)
    Bases: pypuppetdbquery.ast.Node

class pypuppetdbquery.ast.RegexpIdentifier(name)
    Bases: pypuppetdbquery.ast.Identifier

class pypuppetdbquery.ast.RegexpNodeMatch(value)
    Bases: pypuppetdbquery.ast.Expression

class pypuppetdbquery.ast.Resource(res_type, title, exported, parameters=None)
    Bases: pypuppetdbquery.ast.Expression

class pypuppetdbquery.ast.Subquery(endpoint, expression)
    Bases: pypuppetdbquery.ast.Node

class pypuppetdbquery.ast.UnaryExpression(expression)
    Bases: pypuppetdbquery.ast.Node
```

1.1.2 pypuppetdbquery.evaluator module

```
class pypuppetdbquery.evaluator.Evaluator
    Bases: object

    Converts a pypuppetdbquery.ast Abstract Syntax Tree into a PuppetDB native AST query.

    DECAMEL_RE = re.compile("(?!^)([A-Z]+)")
        Regular expression used when converting CamelCase class names to underscore_separated names.

    evaluate (ast, mode='nodes')
        Process a parsed PuppetDBQuery AST and return a PuppetDB AST.

        The resulting PuppetDB AST is a native Python list. It will need converting to JSON (using
        json.dumps()) before it can be used with PuppetDB.

        Parameters

        • ast (pypuppetdbquery.ast.Query) – Root of the AST to evaluate
        • mode (str) – PuppetDB endpoint to target

        Returns PuppetDB AST

        Return type list
```


1.1.3 pypuppetdbquery.lexer module

exception `pypuppetdbquery.lexer.LexException` (*message*, *position*)

Bases: `Exception`

Raised for errors encountered during lexing.

Such errors may include unknown tokens or unexpected EOF, for example. The position of the lexer when the error was encountered (the index into the input string) is stored in the *position* attribute.

class `pypuppetdbquery.lexer.Lexer` (***kwargs*)

Bases: `object`

Lexer for the PuppetDBQuery language.

This class uses `ply.lex.lex()` to implement the lexer (or tokenizer). It is used by `pypuppetdbquery.parser.Parser` in order to process queries.

The arguments to the constructor are passed directly to `ply.lex.lex()`.

Note: Many of the docstrings in this class are used by `ply.lex` to build the lexer. These strings are not particularly useful for generating documentation from, so the built documentation for this class may not be very useful.

input (*s*)

Reset and supply input to the lexer.

Tokens then need to be obtained using `token()` or the iterator interface provided by this class.

next ()

Implementation of `iterator.next()`.

Return the next item from the container. If there are no further items, raise the `StopIteration` exception.

t_asterisk = '*'

t_at = '@'

t_boolean (*t*)
true/false

t_dot = '\\.'

t_equals = '='

t_exported = '@@'

t_float (*t*)
-?d+.d+

t_greaterthan = '>'

t_greaterthaneq = '>='

t_hash = '[#]'

t_lbrace = '{'

t_lbrack = '\\['

t_lessthan = '<'

t_lessthaneq = '<='

```
t_LPAREN = '\\('
t_MATCH = '~'
t_NOTEQUALS = '!= '
t_NOTMATCH = '!~'
t_NUMBER(t)
    -?d+
t_RBRACE = '}'
t_RBRACK = '\\]'
t_RPAREN = '\\)'
t_STRING_bareword(t)
    [-w_:]+
t_STRING_double_quoted(t)
    "(\\.|[^\"])*"
t_STRING_single_quoted(t)
    '(\\.|[^\'])*'
t_error(t)
t_ignore = '\\t\\n\\r\\x0c\\x0b'
t_keyword(t)
    notlandlor
token()
    Obtain one token from the input.
tokens = ('LPAREN', 'RPAREN', 'LBRACK', 'RBRACK', 'LBRACE', 'RBRACE', 'EQUALS', 'NOTEQUALS', 'MAT
```

List of token names handled by the lexer.

1.1.4 pypuppetdbquery.parser module

exception `pypuppetdbquery.parser.ParseException` (*message*, *position*)

Bases: `Exception`

Raised for errors encountered during parsing.

The position of the lexer when the error was encountered (the index into the input string) is stored in the *position* attribute.

class `pypuppetdbquery.parser.Parser` (*lex_options=None*, *yacc_options=None*)

Bases: `object`

Parser for the PuppetDBQuery language.

This class uses `ply.yacc.yacc()` to implement the parser. In concert with `pypuppetdbquery.lexer.Lexer`, it produces an Abstract Syntax Tree (AST) as declared in `pypuppetdbquery.ast`.

Parameters

- **lex_options** (*dict*) – Passed as keyword arguments to `pypuppetdbquery.lexer.Lexer`
- **yacc_options** (*dict*) – Passed as keyword arguments to `ply.yacc.yacc()`

Note: Many of the docstrings in this class are used by `ply.yacc` to build the parser. These strings are not particularly useful for generating documentation from, so the built documentation for this class may not be very useful.

p_block_expr (*p*)
 block_expr : LBRACE expr RBRACE

p_boolean (*p*)
 boolean : BOOLEAN

p_comparison_expr (*p*)
 comparison_expr : identifier_path comparison_op literal

p_comparison_op (*p*)
 comparison_op [MATCH]
 NOTMATCH
 EQUALS
 NOTEQUALS
 GREATERTHAN
 GREATERTHANEQ
 LESSTHAN
 LESSTHANEQ

p_empty (*p*)
 empty :

p_error (*p*)

p_expr (*p*)
 expr [resource_expr]
 comparison_expr
 subquery

p_expr_and (*p*)
 expr : expr AND expr

p_expr_identifier_path (*p*)
 expr : identifier_path

p_expr_not (*p*)
 expr : NOT expr

p_expr_or (*p*)
 expr : expr OR expr

p_expr_parenthesized (*p*)
 expr : LPAREN expr RPAREN

p_float (*p*)
 float : FLOAT

p_identifier (*p*)
 identifier [string]
 integer

p_identifier_path (*p*)
 identifier_path : identifier

p_identifier_path_nested (*p*)
 identifier_path : identifier_path DOT identifier

p_identifier_regexp (*p*)
 identifier : MATCH string

p_identifier_wild (*p*)
 identifier : ASTERISK

p_integer (*p*)
 integer : NUMBER

p_literal (*p*)
 literal [boolean]
 string
 integer
 float

p_literal_date (*p*)
 literal : AT string

p_query (*p*)
 query [expr]
 empty

p_resource_expr (*p*)
 resource_expr : string LBRACK identifier RBRACK

p_resource_expr_exported (*p*)
 resource_expr : EXPORTED string LBRACK identifier RBRACK

p_resource_expr_exported_param (*p*)
 resource_expr : EXPORTED string LBRACK identifier RBRACK block_expr

p_resource_expr_param (*p*)
 resource_expr : string LBRACK identifier RBRACK block_expr

p_string (*p*)
 string : STRING

p_subquery_block (*p*)
 subquery : HASH string block_expr

p_subquery_comparison (*p*)
 subquery : HASH string DOT comparison_expr

parse (*text*, *debug*=0)
 Parse the input string and return an AST.

Parameters

- **text** (*str*) – The query to parse
- **debug** (*bool*) – Output detailed information during the parsing process

Returns An Abstract Syntax Tree

Return type *pypuppetdbquery.ast.Query*

precedence = (('left', 'OR'), ('left', 'AND'), ('left', 'EQUALS', 'MATCH', 'LESSTHAN', 'GREATERTHAN'), ('right', 'MATCH'))

Precedence rules in lowest to highest order

start = 'query'

Non-terminal to use as the starting grammar symbol

Examples

Basic query for nodes using pypuppetdb:

```
import pypuppetdb
import pypuppetdbquery

pdb = pypuppetdb.connect()

pdb_ast = pypuppetdbquery.parse(
    '(processorcount=4 or processorcount=8) and kernel=Linux')

for node in pdb.nodes(query=pdb_ast):
    print(node)
```

Test Suite

```
class test_frontend.TestFrontend (methodName='runTest')
    Bases: unittest.case.TestCase

    Test cases targetting py puppetdbquery, and particularly py puppetdbquery.parse().

    test_empty_queries()
    test_simple_json()
    test_simple_raw()

class test_integration.TestIntegration (methodName='runTest')
    Bases: unittest.case.TestCase

    Test cases for the integrated combination of py puppetdbquery.lexer.Lexer,
    py puppetdbquery.parser.Parser, and py puppetdbquery.evaluator.Evaluator.

    setUp()

    test_boolean_values()
    test_capitalize_class_names()
    test_dates_in_queries()
    test_does_not_wrap_subquery_with_mode_none()
    test_double_quoted_strings()
    test_empty_queries()
    test_equals_operator()
    test_escape_non_match_parts_on_structured_facts_with_match_op()
    test_float_values()
    test_greater_than_eq_operator()
    test_greater_than_operator()
    test_less_than_eq_operator()
    test_less_than_operator()
    test_match_operator()
    test_negate_expressions()
    test_node_subqueries()
```

```
test_node_subqueries_with_block_of_conditions()
test_node_subqueries_with_fact_query()
test_node_subquery_fact_field()
test_not_equals_operator()
test_not_match_operator()
test_precedence_a()
test_precedence_b()
test_precedence_within_resource_parameter_queries_a()
test_precedence_within_resource_parameter_queries_b()
test_resource_queries_for_exported_resources()
test_resource_queries_for_exported_resources_with_parameters()
test_resource_queries_with_regexp_title_matching()
test_resource_queries_with_tags()
test_resource_queries_with_type_and_title()
test_resource_queries_with_type_title_and_parameters()
test_single_quoted_strings()
test_single_string_expressions()
test_structured_facts()
test_structured_facts_with_array_component()
test_structured_facts_with_match_operator()
test_structured_facts_with_wildcard_operator()

class test_lexer.TestLexer (methodName='runTest')
    Bases: unittest.case.TestCase
    Test cases for pypuppetdbquery.lexer.Lexer.
    setUp()
    test_all_tokens()
    test_empty_queries()
    test_invalid_input()

class test_parser.TestParster (methodName='runTest')
    Bases: unittest.case.TestCase
    Test cases for pypuppetdbquery.parser.Parser.
    setUp()
    test_boolean_values()
    test_dates_in_queries()
    test_double_quoted_strings()
    test_empty_queries()
    test_float_values()
```

```
test_invalid_input()
test_invalid_input_eof()
test_logical_operators()
test_negate_expression()
test_node_subqueries()
test_node_subqueries_with_block_of_conditions()
test_resource_queries_for_exported_resources()
test_resource_queries_for_exported_resources_with_parameters()
test_resource_queries_with_type_and_regexp_identifier()
test_resource_queries_with_type_title_and_parameters()
test_single_string_expressions()
test_structured_facts_with_wildcard_operator()
```

Indices and tables

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

p

- `pypuppetdbquery`, 3
- `pypuppetdbquery.ast`, 3
- `pypuppetdbquery.evaluator`, 4
- `pypuppetdbquery.lexer`, 5
- `pypuppetdbquery.parser`, 6

t

- `test_frontend`, 13
- `test_integration`, 13
- `test_lexer`, 14
- `test_parser`, 14

A

AndExpression (class in pypuppetdbquery.ast), 3

B

BinaryExpression (class in pypuppetdbquery.ast), 3

BlockExpression (class in pypuppetdbquery.ast), 3

C

Comparison (class in pypuppetdbquery.ast), 3

D

Date (class in pypuppetdbquery.ast), 3

DECAMEL_RE (pypuppetdbquery.evaluator.Evaluator attribute), 4

E

evaluate() (pypuppetdbquery.evaluator.Evaluator method), 4

Evaluator (class in pypuppetdbquery.evaluator), 4

Expression (class in pypuppetdbquery.ast), 3

I

Identifier (class in pypuppetdbquery.ast), 3

IdentifierPath (class in pypuppetdbquery.ast), 4

input() (pypuppetdbquery.lexer.Lexer method), 5

L

Lexer (class in pypuppetdbquery.lexer), 5

LexException, 5

Literal (class in pypuppetdbquery.ast), 4

N

next() (pypuppetdbquery.lexer.Lexer method), 5

Node (class in pypuppetdbquery.ast), 4

NotExpression (class in pypuppetdbquery.ast), 4

O

OrExpression (class in pypuppetdbquery.ast), 4

P

p_block_expr() (pypuppetdbquery.parser.Parser method), 7

p_boolean() (pypuppetdbquery.parser.Parser method), 7

p_comparison_expr() (pypuppetdbquery.parser.Parser method), 7

p_comparison_op() (pypuppetdbquery.parser.Parser method), 7

p_empty() (pypuppetdbquery.parser.Parser method), 7

p_error() (pypuppetdbquery.parser.Parser method), 7

p_expr() (pypuppetdbquery.parser.Parser method), 7

p_expr_and() (pypuppetdbquery.parser.Parser method), 7

p_expr_identifier_path() (pypuppetdbquery.parser.Parser method), 7

p_expr_not() (pypuppetdbquery.parser.Parser method), 7

p_expr_or() (pypuppetdbquery.parser.Parser method), 7

p_expr_parenthesized() (pypuppetdbquery.parser.Parser method), 7

p_float() (pypuppetdbquery.parser.Parser method), 7

p_identifier() (pypuppetdbquery.parser.Parser method), 7

p_identifier_path() (pypuppetdbquery.parser.Parser method), 7

p_identifier_path_nested() (pypuppetdbquery.parser.Parser method), 8

p_identifier_regexp() (pypuppetdbquery.parser.Parser method), 8

p_identifier_wild() (pypuppetdbquery.parser.Parser method), 8

p_integer() (pypuppetdbquery.parser.Parser method), 8

p_literal() (pypuppetdbquery.parser.Parser method), 8

p_literal_date() (pypuppetdbquery.parser.Parser method), 8

p_query() (pypuppetdbquery.parser.Parser method), 8

p_resource_expr() (pypuppetdbquery.parser.Parser method), 8

p_resource_expr_exported() (pypuppetdbquery.parser.Parser method), 8

p_resource_expr_exported_param() (pypuppetdbquery.parser.Parser method), 8

[p_resource_expr_param\(\)](#) (pypuppetdbquery.parser.Parser method), 8
[p_string\(\)](#) (pypuppetdbquery.parser.Parser method), 8
[p_subquery_block\(\)](#) (pypuppetdbquery.parser.Parser method), 8
[p_subquery_comparison\(\)](#) (pypuppetdbquery.parser.Parser method), 8
[ParenthesizedExpression](#) (class in pypuppetdbquery.ast), 4
[parse\(\)](#) (in module pypuppetdbquery), 3
[parse\(\)](#) (pypuppetdbquery.parser.Parser method), 8
[ParseException](#), 6
[Parser](#) (class in pypuppetdbquery.parser), 6
[precedence](#) (pypuppetdbquery.parser.Parser attribute), 8
[pypuppetdbquery](#) (module), 3
[pypuppetdbquery.ast](#) (module), 3
[pypuppetdbquery.evaluator](#) (module), 4
[pypuppetdbquery.lexer](#) (module), 5
[pypuppetdbquery.parser](#) (module), 6

Q

[Query](#) (class in pypuppetdbquery.ast), 4

R

[RegexIdentifier](#) (class in pypuppetdbquery.ast), 4
[RegexNodeMatch](#) (class in pypuppetdbquery.ast), 4
[Resource](#) (class in pypuppetdbquery.ast), 4

S

[setUp\(\)](#) (test_integration.TestIntegration method), 13
[setUp\(\)](#) (test_lexer.TestLexer method), 14
[setUp\(\)](#) (test_parser.TestParster method), 14
[start](#) (pypuppetdbquery.parser.Parser attribute), 9
[Subquery](#) (class in pypuppetdbquery.ast), 4

T

[t_asterisk](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_at](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_boolean\(\)](#) (pypuppetdbquery.lexer.Lexer method), 5
[t_dot](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_equals](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_error\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[t_exported](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_float\(\)](#) (pypuppetdbquery.lexer.Lexer method), 5
[t_greaterthan](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_greaterthaneq](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_hash](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_ignore](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_keyword\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[t_lbrace](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_lbrack](#) (pypuppetdbquery.lexer.Lexer attribute), 5

[t_lessthan](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_lessthaneq](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_lparen](#) (pypuppetdbquery.lexer.Lexer attribute), 5
[t_match](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_notequals](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_notmatch](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_number\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[t_rbrace](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_rbrack](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_rparen](#) (pypuppetdbquery.lexer.Lexer attribute), 6
[t_string_bareword\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[t_string_double_quoted\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[t_string_single_quoted\(\)](#) (pypuppetdbquery.lexer.Lexer method), 6
[test_all_tokens\(\)](#) (test_lexer.TestLexer method), 14
[test_boolean_values\(\)](#) (test_integration.TestIntegration method), 13
[test_boolean_values\(\)](#) (test_parser.TestParster method), 14
[test_capitalize_class_names\(\)](#) (test_integration.TestIntegration method), 13
[test_dates_in_queries\(\)](#) (test_integration.TestIntegration method), 13
[test_dates_in_queries\(\)](#) (test_parser.TestParster method), 14
[test_does_not_wrap_subquery_with_mode_none\(\)](#) (test_integration.TestIntegration method), 13
[test_double_quoted_strings\(\)](#) (test_integration.TestIntegration method), 13
[test_double_quoted_strings\(\)](#) (test_parser.TestParster method), 14
[test_empty_queries\(\)](#) (test_frontend.TestFrontend method), 13
[test_empty_queries\(\)](#) (test_integration.TestIntegration method), 13
[test_empty_queries\(\)](#) (test_lexer.TestLexer method), 14
[test_empty_queries\(\)](#) (test_parser.TestParster method), 14
[test_equals_operator\(\)](#) (test_integration.TestIntegration method), 13
[test_escape_non_match_parts_on_structured_facts_with_match_op\(\)](#) (test_integration.TestIntegration method), 13
[test_float_values\(\)](#) (test_integration.TestIntegration method), 13
[test_float_values\(\)](#) (test_parser.TestParster method), 14
[test_frontend](#) (module), 13
[test_greater_than_eq_operator\(\)](#) (test_integration.TestIntegration method),

- 13
- test_greater_than_operator() (test_integration.TestIntegration method), 13
- test_integration (module), 13
- test_invalid_input() (test_lexer.TestLexer method), 14
- test_invalid_input() (test_parser.TestParster method), 14
- test_invalid_input_eof() (test_parser.TestParster method), 15
- test_less_than_eq_operator() (test_integration.TestIntegration method), 13
- test_less_than_operator() (test_integration.TestIntegration method), 13
- test_lexer (module), 14
- test_logical_operators() (test_parser.TestParster method), 15
- test_match_operator() (test_integration.TestIntegration method), 13
- test_negate_expression() (test_parser.TestParster method), 15
- test_negate_expressions() (test_integration.TestIntegration method), 13
- test_node_subqueries() (test_integration.TestIntegration method), 13
- test_node_subqueries() (test_parser.TestParster method), 15
- test_node_subqueries_with_block_of_conditions() (test_integration.TestIntegration method), 13
- test_node_subqueries_with_block_of_conditions() (test_parser.TestParster method), 15
- test_node_subqueries_with_fact_query() (test_integration.TestIntegration method), 14
- test_node_subquery_fact_field() (test_integration.TestIntegration method), 14
- test_not_equals_operator() (test_integration.TestIntegration method), 14
- test_not_match_operator() (test_integration.TestIntegration method), 14
- test_parser (module), 14
- test_precedence_a() (test_integration.TestIntegration method), 14
- test_precedence_b() (test_integration.TestIntegration method), 14
- test_precedence_within_resource_parameter_queries_a() (test_integration.TestIntegration method), 14
- test_precedence_within_resource_parameter_queries_b() (test_integration.TestIntegration method), 14
- test_resource_queries_for_exported_resources() (test_integration.TestIntegration method), 14
- test_resource_queries_for_exported_resources() (test_parser.TestParster method), 15
- test_resource_queries_for_exported_resources_with_parameters() (test_integration.TestIntegration method), 14
- test_resource_queries_for_exported_resources_with_parameters() (test_parser.TestParster method), 15
- test_resource_queries_with_regexp_title_matching() (test_integration.TestIntegration method), 14
- test_resource_queries_with_tags() (test_integration.TestIntegration method), 14
- test_resource_queries_with_type_and_regexp_identifier() (test_parser.TestParster method), 15
- test_resource_queries_with_type_and_title() (test_integration.TestIntegration method), 14
- test_resource_queries_with_type_title_and_parameters() (test_integration.TestIntegration method), 14
- test_resource_queries_with_type_title_and_parameters() (test_parser.TestParster method), 15
- test_simple_json() (test_frontend.TestFrontend method), 13
- test_simple_raw() (test_frontend.TestFrontend method), 13
- test_single_quoted_strings() (test_integration.TestIntegration method), 14
- test_single_string_expressions() (test_integration.TestIntegration method), 14
- test_single_string_expressions() (test_parser.TestParster method), 15
- test_structured_facts() (test_integration.TestIntegration method), 14
- test_structured_facts_with_array_component() (test_integration.TestIntegration method), 14
- test_structured_facts_with_match_operator() (test_integration.TestIntegration method), 14
- test_structured_facts_with_wildcard_operator() (test_integration.TestIntegration method), 14
- test_structured_facts_with_wildcard_operator() (test_parser.TestParster method), 15
- TestFrontend (class in test_frontend), 13
- TestIntegration (class in test_integration), 13
- TestLexer (class in test_lexer), 14
- TestParster (class in test_parser), 14
- token() (pypuppetdbquery.lexer.Lexer method), 6
- tokens (pypuppetdbquery.lexer.Lexer attribute), 6

U

[UnaryExpression](#) (class in `pypuppetdbquery.ast`), [4](#)