

## Ruby master - Feature #4276

### Allow use of quotes in symbol syntactic sugar for hashes

01/13/2011 10:17 AM - tbenst (Tyler Benster)

<b>Status:</b>	Closed	
<b>Priority:</b>	Normal	
<b>Assignee:</b>	matz (Yukihiro Matsumoto)	
<b>Target version:</b>	2.6	
<b>Description</b>		
Current syntactic sugar allows this:		
<pre>hash = {Alabama: "AL"}</pre>		
This feature request is to also allow symbols delimited by quotes (and thus able to contain a whitespace) to use an equivalent syntactic sugar:		
<pre>hash2 = {"Rhode Island": "RI"}</pre>		
<b>Related issues:</b>		
Related to Ruby master - Feature #4801: Shorthand Hash Syntax for Strings	<b>Rejected</b>	<b>05/30/2011</b>
Related to Ruby master - Bug #10653: do-end block in ternary operator is synt...	<b>Closed</b>	
Has duplicate Ruby master - Feature #4935: Quoted Label Form for 1.9 Hashes	<b>Closed</b>	<b>06/27/2011</b>
Has duplicate Ruby master - Feature #9047: Alternate hash key syntax for symbols	<b>Closed</b>	<b>10/24/2013</b>

#### Associated revisions

##### Revision b0c03f63 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@47649 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

##### Revision 47649 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature #4276]

#### Revision 05f99bcb - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@47650 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

#### Revision 47650 - 09/20/2014 02:49 AM - nobu (Nobuyoshi Nakada)

NEWS: quoted symbol keys

- NEWS (Language changes): add quoted symbol keys. [Feature #4276]

## History

---

### #1 - 06/01/2011 11:59 PM - nobu (Nobuyoshi Nakada)

Hi,

At Thu, 13 Jan 2011 10:17:23 +0900,  
Tyler Benster wrote in [ruby-core:34453]:

Current syntactic sugar allows this:

```
hash = {Alabama: "AL"}
```

This feature request is to also allow symbols delimited by quotes (and thus able to contain a whitespace) to use an equivalent syntactic sugar:

```
hash2 = {"Rhode Island": "RI"}
```

I've forgotten to post the patch.

```
diff --git i/parse.y w/parse.y
index 06f96ce..0cacdd5 100644
--- i/parse.y
+++ w/parse.y
@@ -380,6 +380,8 @@ static NODE *ret_args_gen(struct parser_params*,NODE*);
 static NODE *arg_blk_pass(NODE*,NODE*);
 static NODE *new_yield_gen(struct parser_params*,NODE*);
 #define new_yield(node) new_yield_gen(parser, (node))
+static NODE *dsym_node_gen(struct parser_params*,NODE*);
+#define dsym_node(node) dsym_node_gen(parser, (node))
```

```

static NODE *gettable_gen(struct parser_params*, ID);
#define gettable(id) gettable_gen(parser, (id))
@@ -678,12 +680,12 @@ static void token_info_pop(struct parser_params*, const char *token);
    keyword__FILE__
    keyword__ENCODING__

-%token <id>    tIDENTIFIER tFID tGVAR tIVAR tCONSTANT tCVAR tLABEL
+%token <id>    tIDENTIFIER tFID tGVAR tIVAR tCONSTANT tCVAR tLABEL tLABEL_END
%token <node>   tINTEGER tFLOAT tSTRING_CONTENT tCHAR
%token <node>   tNTH_REF tBACK_REF
%token <num>   tREGEXP_END

-%type <node>  singleton strings string stringl xstring regexp
+%type <node>  singleton strings string stringl xstring regexp string_body
%type <node>   string_contents xstring_contents regexp_contents string_content
%type <node>   words qwords word_list qword_list word
%type <node>   literal numeric dsym cpath
@@ -3841,12 +3843,18 @@ string      : tCHAR
    }
    ;

-stringl      : tSTRING_BEG string_contents tSTRING_END
+string_body : tSTRING_BEG string_contents
    {
    /*%%*/
    $$ = $2;
    }
    ;
+
+stringl      : string_body tSTRING_END
+
+    {
+    /*%%*/
+    $$ = $1;
+    /*%
+    $$ = dispatch1(string_literal, $2);
+    $$ = dispatch1(string_literal, $1);
+    %*/
+    }
+
+    ;
@@ -4199,26 +4207,7 @@ dsym      : tSYMBEG xstring_contents tSTRING_END
    {
    lex_state = EXPR_END;
    /*%%*/
    if (!($$ = $2)) {
    $$ = NEW_LIT(ID2SYM(rb_intern("")));
    }
    else {
    VALUE lit;

    switch (nd_type($$)) {
    case NODE_DSTR:
    nd_set_type($$, NODE_DSYM);
    break;
    case NODE_STR:
    lit = $$->nd_lit;
    $$->nd_lit = ID2SYM(rb_intern_str(lit));
    nd_set_type($$, NODE_LIT);
    break;
    default:
    $$ = NEW_NODE(NODE_DSYM, Qnil, 1, NEW_LIST($$));
    break;
    }
    }
+
+    $$ = dsym_node($2);
+    /*%
+    $$ = dispatch1(dyna_symbol, $2);
+    %*/
@@ -4761,6 +4750,15 @@ assoc      : arg_value tASSOC arg_value
    $$ = dispatch2(assoc_new, $1, $2);
    /*%
    }
+
+    | string_body tLABEL_END arg_value
+
+    {
+    /*%%*/

```

```

+     $$ = list_append(NEW_LIST(dsym_node($1)), $3);
+     /*%
+     $$ = dispatch1(dyna_symbol, $1);
+     $$ = dispatch2(assoc_new, $$, $3);
+     %*/
+     }
+
;

operation : tIDENTIFIER
@@ -5334,6 +5332,7 @@ rb_parser_compile_file(volatile VALUE vparser, const char *f, VALUE file, int st
#define STR_FUNC_QWORDS 0x08
#define STR_FUNC_SYMBOL 0x10
#define STR_FUNC_INDENT 0x20
+ #define STR_FUNC_LABEL 0x40

enum string_type {
    str_squote = (0),
@@ -5925,6 +5924,8 @@ parser_tokadd_string(struct parser_params *parser,

#define NEW_STRTERM(func, term, paren) \
    rb_node_newnode(NODE_STRTERM, (func), (term) | ((paren) << (CHAR_BIT * 2)), 0)
+ #define IS_LABEL_SUFFIX(n) (peek_n(':', (n)) && !peek_n(':', (n)+1))
+ #define MAYBE_LABEL() (IS_LABEL_POSSIBLE() ? STR_FUNC_LABEL : 0)

static int
parser_parse_string(struct parser_params *parser, NODE *quote)
@@ -5946,6 +5947,10 @@ parser_parse_string(struct parser_params *parser, NODE *quote)
    quote->nd_func = -1;
    return ' ';
}
+ if ((func & STR_FUNC_LABEL) && IS_LABEL_SUFFIX(0)) {
+     lex_state = EXPR_BEG;
+     return tLABEL_END;
+ }
+ if (!(func & STR_FUNC_REGEXP)) return tSTRING_END;
+     set_yylval_num(regx_options());
+     return tREGEXP_END;
@@ -6533,7 +6538,6 @@ parser_prepare(struct parser_params *parser)
#define IS_BEG() (lex_state == EXPR_BEG || lex_state == EXPR_MID || lex_state == EXPR_VALUE || lex_state == E
XPR_CLASS)
#define IS_SPCARG(c) (IS_ARG() && space_seen && !ISSPACE(c))
#define IS_LABEL_POSSIBLE() ((lex_state == EXPR_BEG && !cmd_state) || IS_ARG())
- #define IS_LABEL_SUFFIX(n) (peek_n(':', (n)) && !peek_n(':', (n)+1))

#ifdef RIPPER
#define ambiguous_operator(op, syn) ( \
@@ -6849,7 +6853,7 @@ parser_yylex(struct parser_params *parser)
    return '>';

    case '"':
-     lex_strterm = NEW_STRTERM(str_dquote, '"', 0);
+     lex_strterm = NEW_STRTERM(str_dquote | MAYBE_LABEL(), '"', 0);
    return tSTRING_BEG;

    case '`':
@@ -6868,7 +6872,7 @@ parser_yylex(struct parser_params *parser)
    return tXSTRING_BEG;

    case '\':
-     lex_strterm = NEW_STRTERM(str_squote, '\'', 0);
+     lex_strterm = NEW_STRTERM(str_squote | MAYBE_LABEL(), '\'', 0);
    return tSTRING_BEG;

    case '?':
@@ -8987,6 +8991,32 @@ new_args_gen(struct parser_params *parser, NODE *m, NODE *o, ID r, NODE *p, ID b
    ruby_sourceline = saved_line;
    return node;
}
+
+static NODE*
+dsym_node_gen(struct parser_params *parser, NODE *node)
+{
+     if (!node) {
+         node = NEW_LIT(ID2SYM(rb_intern("")));
+     }
}

```

```

+   else {
+     VALUE lit;
+
+     switch (nd_type(node)) {
+     case NODE_DSTR:
+       nd_set_type(node, NODE_DSYM);
+       break;
+     case NODE_STR:
+       lit = node->nd_lit;
+       node->nd_lit = ID2SYM(rb_intern_str(lit));
+       nd_set_type(node, NODE_LIT);
+       break;
+     default:
+       node = NEW_NODE(NODE_DSYM, Qnil, 1, NEW_LIST(node));
+       break;
+     }
+   }
+   return node;
+}
#endif /* !RIPPER */

static void

```

---

Nobu Nakada

**#2 - 03/18/2012 07:01 PM - nahi (Hiroshi Nakamura)**

- Description updated
- Category set to core
- Status changed from Open to Assigned
- Assignee set to matz (Yukihiro Matsumoto)

**#3 - 11/20/2012 09:26 PM - mame (Yusuke Endoh)**

- Target version set to 2.6

**#4 - 12/01/2012 02:31 AM - bitsweat (Jeremy Daer)**

This would improve a lot of my code that's punctuated with { foo: bar, :'hoge' => piyo }. Switching syntax is a mental and visual interruption.

I hope Ruby 2 supports this quoted-symbol syntax from the first day it's released, so everyone can rely on it. Please consider this patch for preview2!

**#5 - 12/01/2012 04:41 AM - drbrain (Eric Hodel)**

Sorry, this feature is too late for 2.0.0

**#6 - 01/27/2013 11:26 AM - nobu (Nobuyoshi Nakada)**

One of the reason I didn't introduce this was, a few bundled scripts had compatibility issue when once I tried.

**#7 - 03/15/2014 02:21 AM - nobu (Nobuyoshi Nakada)**

- Has duplicate Feature #9047: Alternate hash key syntax for symbols added

**#8 - 03/15/2014 02:26 AM - nobu (Nobuyoshi Nakada)**

- Description updated

**#9 - 07/25/2014 08:36 AM - nobu (Nobuyoshi Nakada)**

Updated with the test in [Feature #4935]: <https://github.com/ruby/ruby/pull/684>

**#10 - 07/25/2014 12:41 PM - rosenfeld (Rodrigo Rosenfeld Rosas)**

Great, from the test it seems to allow interpolation. Could you please confirm I understood it correctly?

**#11 - 07/26/2014 08:11 AM - matz (Yukihiro Matsumoto)**

- Status changed from Assigned to Feedback

I am not against the idea, but I want to make sure that key will be symbol, since some may expect

```
{"foo bar": 12}
```

to be a hash with string key, especially who is familiar with JSON.

Matz.

#### #12 - 07/26/2014 01:54 PM - nobu (Nobuyoshi Nakada)

Rodrigo Rosenfeld Rosas wrote:

Great, from the test it seems to allow interpolation. Could you please confirm I understood it correctly?

Yes, it is allowed inside double quotes, but not single quotes.  
Other terminators (e.g., %(...)) cause syntax errors.

#### #13 - 08/11/2014 06:18 PM - Ajedi32 (Andrew M)

Yukihiro Matsumoto wrote:

I am not against the idea, but I want to make sure that key will be symbol,

Yes, I believe that was the intention. E.g.

```
# Currently, Hashes have syntactic sugar which allows:
{:key => "Value"} == {key: "Value"} #=> true

# This feature request is to allow:
{:some key" => "Value"} == {"some key": "Value"} #=> true
```

#### #14 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

OK, I commit the previous patch.  
Yell if you don't like it.

#### #15 - 09/20/2014 01:48 AM - nobu (Nobuyoshi Nakada)

- Status changed from *Feedback* to *Closed*  
- % Done changed from 0 to 100

Applied in changeset r47649.

---

parse.y: quoted ID key

- parse.y (assoc): allow quoted ID as a key of a hash literal. [ruby-core:34453] [Feature [#4276](#)]

#### #16 - 09/20/2014 02:27 AM - nobu (Nobuyoshi Nakada)

Just a note: last evening at RubyKaigi 2014, talked to matz about this issue, and got his approval to introduce this to see if anyone rants.

#### #17 - 12/04/2014 07:28 PM - etienne (Étienne Barrié)

Hi, any reason it doesn't work with inner hashes or arrays?

```
{ foo: {} } # => {:foo=>{}}
{ 'foo': {} }
# SyntaxError: (irb):2: syntax error, unexpected '['
{ 'foo': [] }
SyntaxError: (irb):1: syntax error, unexpected '['
```

#### #18 - 12/26/2014 08:01 AM - nobu (Nobuyoshi Nakada)

- Related to Bug #10653: do-end block in ternary operator is syntax error added