# [PATCH] resurrection of # -*- warn_past_scope: true -*-

12/04/2017 10:37 AM - shyouhei (Shyouhei Urabe)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Open</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
</tbody>
</table>

## Description

I understand this feature was killed due to [Bug #10661].

However sometimes I **do** want to check variable name collisions.

Please consider this feature again, with default off; no warning shall be emitted unless there are the dedicated magic comment in the source code.

Signed-off-by: Urabe, Shyouhei <shyouhei@ruby-lang.org>

```diff
diff --git a/parse.y b/parse.y
index 03b9ed992e..a4a31af5e0 100644
--- a/parse.y
+++ b/parse.y
@@ -37,7 +37,7 @@
   #include "probes.h"
   #ifndef WARN_PAST_SCOPE
-#define WARN_PAST_SCOPE 0
+#define WARN_PAST_SCOPE 1
   #endif

 #define TAB_WIDTH 8
```

## Related issues:

Related to Ruby master - Bug #10661: The "possible reference to past scope" warning is quite frustrating and is forcing me to change my variable names from what I want added

#2 - 12/04/2017 10:49 AM - shyouhei (Shyouhei Urabe)

This patch works like this:

```bash
% ./ruby --disable-gems -w -ve '[1, 2, 3].sample.tap { |rand| puts "Random value: #{rand}" }; puts "Another random value: #{rand}"'
```

rub 2.5.0dev (2017-12-04) [x86_64-darwin15]
Random value: 3
Another random value: 0.25989520023218615

## History

- Related to Bug #10661: The "possible reference to past scope" warning is quite frustrating and is forcing me to change my variable names from what I want added

#2 - 12/04/2017 10:49 AM - shyouhei (Shyouhei Urabe)

This patch works like this:

```bash
% ./ruby --disable-gems -w -ve '# warn_past_scope: true' -e '[1, 2, 3].sample.tap { |rand| puts "Random value: #{rand}" }; puts "Another random value: #{rand}"'
```
I have no pro or con opinion so I will not comment on the suggestion.

I have one question though, apologies for a bit of side tracking:

- Can this be combined with other comment options such as "# frozen_string_literal: true"

Perhaps it may not apply as your example is only commandline but I wanted to ask just in case anyone may want to combine more than one option, for whatever reason, into the .rb file at hand.

Seems to me to be more appropriate as a Rubocop cop.

Yes. You can write multiple magic comments at once, like this:

```
# -*- coding: utf-8; frozen_string_literal: true; warn_indent: true; warn_past_scope: true -*-
```

or, line by line like this:

```
# coding: utf-8
# frozen_string_literal: true
# warn_indent: true
# warn_past_scope: true
```

At least, I don't like the name warn_past_code. It's not intuitive. And it may be better to be handled by Rubocop as Marc-Andre said.

Matz.

Are we relying on rubocop as ruby's official linter?

This question comes inline with my recent previous comment around core functionalities that I believe should be better managed as part of ruby API. Debugger, Coverage, Unit tests, Benchmarks, Performance tools and yes Linter. Missing Documentation here. Maybe something else?