String Formatting in Python 3

```
"(%d goals, $%d)" % (self.goals, self.penalties)
```

^ I know how to do this in Python 2

What is the Python 3 version of this?

I tried searching for examples online but I kept getting Python 2 versions

python python-3.x

asked Dec 19 '12 at 4:53
JoseBazBaz
340 ● 1 ● 4 ● 16

8 Note for the casual reader: The dollar sign has no special meaning. It is *just* a dollar sign here. (Thanks Martijn Pieters) – kevinarpe Oct 27 '14 at 15:23

3 Answers

Here are the docs about the "new" format syntax. An example would be:

```
"({:d} goals, ${:d})".format(self.goals, self.penalties)
```

If both `goals` and `penalties` are integers (i.e. their default format is ok), it could be shortened to:

```
"({} goals, ${})".format(self.goals, self.penalties)
```

And since the parameters are fields of `self`, there's also a way of doing it using a single argument twice (as @Burhan Khalid noted in the comments):

```
"({0.goals} goals, ${0.penalties})".format(self)
```

Explaining:

- `{}` means just the next positional argument, with default format;
- `{0}` means the argument with index `0`, with default format;
- `{:d}` is the next positional argument, with decimal integer format;
- `{0:d}` is the argument with index `0`, with decimal integer format.

There are many others things you can do when selecting an argument (using named arguments instead of positional ones, accessing fields, etc) and many format options as well (padding the number, using thousands separators, showing sign or not, etc). Some other examples:

```
"({goals} goals, ${penalties})".format(goals=2, penalties=4)
"({goals} goals, ${penalties})".format(**self.__dict__)

"first goal: {0.goal_list[0]}".format(self)
"second goal: {.goal_list[1]}".format(self)

"conversion rate: {:.2f}".format(self.goals / self.shots) # '0.20'
"conversion rate: {:.2%}".format(self.goals / self.shots) # '20.45%'
"conversion rate: {:.0%}".format(self.goals / self.shots) # '20%'

"self: {!s}".format(self) # 'Player: Bob'
"self: {!r}".format(self) # '<__main__.Player instance at 0x00BF7260>'

"games: {:>3}".format(player1.games)  # 'games: 123'
"games: {:>3}".format(player2.games)  # 'games:   4'
"games: {:0>3}".format(player2.games) # 'games: 004'
```

Note: As others pointed out, the new format does not supersede the former, both are available both in Python 3 and the newer versions of Python 2 as well. Some may say it's a matter of preference, but IMHO the newer is much more expressive than the older, and should be used whenever writing new code (unless it's targeting older environments, of course).

edited Mar 21 '14 at 14:22

answered Dec 19 '12 at 4:56
mgibsonbr
14.7k ● 4 ● 30 ● 68

9 You can also do `"({0.goals} goals, ${0.penalties})".format(self)` – Burhan Khalid Dec 19 '12 at 4:58

You need to take the '%' outside of the curly braces, e.g. {:.2f%} -> {:.2f}% – SuperElectric Apr 4 at 17:43

@SuperElectric you mean, in this line: `"conversion rate: {:.2%}".format(self.goals / self.shots)` ? Works fine for me as it is... (Python 3.4) Note that there's no `f` in it, I'm asking to format as a percentage, not as a floating point number. – mgibsonbr Apr 4 at 17:48

You're right; it seems to work in Python 3.4, so that's fine since the OP was asking about Python3. I just found it didn't work with python 2.7.6. `"{:.2f}%".format(float_num)` works fine for both. – SuperElectric Apr 4 at 17:53

@SuperElectric Are you sure? I tested both on my machine (Windows) and ideone (Linux, I think) and got the same result. The only issue I had was with `/`, that means "regular division" in Python 3.4 and "operands-compatible division" in Python 2.7 (so I got `0.00%` at first, since the operands were integers), but fixing that the format worked as intended. BTW, in your example you'd need to multiply the number to be formatted by `100` to achieve the same result. – mgibsonbr Apr 4 at 18:18

That line works as-is in Python 3.

```
>>> sys.version
'3.2 (r32:88445, Oct 20 2012, 14:09:29) \n[GCC 4.5.2]'
>>> "(%d goals, $%d)" % (self.goals, self.penalties)
'(1 goals, $2)'
```

answered Dec 19 '12 at 4:55
Mark Reed
34k ● 5 ● 42 ● 72

OK. COOL. After I posted the question, I was continuing my search and found that we have to do {%d} instead of just %d is that correct, too? Or is the Python2 way the only way to do it? – JoseBazBaz Dec 19 '12 at 4:58

2 Python 3 introduces a new syntax - that's the curly braces - but you're not required to use it. You can use either the old or the new. But they are different; if you're using curlies, you don't use percent signs. – Mark Reed Dec 19 '12 at 5:00

It was not deprecated, your code works fine. Read more here.

answered Dec 19 '12 at 4:57
yentup
2,371 ● 1 ● 10 ● 31