PYTHON SNIPPETS

Python's format Function

Phil Best
2 min read · Sep 30

Hey there, and welcome to another short Python snippet post! This time we're taking a look at the `format` function.

I know what a lot of you are thinking: "Phil, format isn't a function. It's a method! If you're going to teach us stuff, at least use the right terminology..."

Well I'm here to tell you that there's a `format` method **and** a `format` function, and they do very different things!

Unsurprisingly, the `format` function is actually very closely related to our previous formatting posts, so you might want to familiarise yourself those before reading any further. You can find a couple of links below:

https://blog.tecladocode.com/python-snippet-20-formatting-numbers-for-printing/

https://blog.tecladocode.com/python-formatting-integers-in-different-bases/

In these posts we talk about using Python's Format Specification Mini

Language to format strings in various ways using some pretty arcane syntax. The `format` function is a way make use of this formatting syntax without using string interpolation. For example, you might use the `format` function in conjunction with string concatenation, or when printing a single value.

So what does the syntax look like?

```
format(12, "02x")  # 0c
```

We can see that `format` takes two arguments in this case. The first is a value to format, and this is actually the only required argument. We'll talk about this more in a little bit.

The second argument is a format specification, and for this we must pass in a string representing the formatting options we want to apply. When performing this kind of formatting as part of string interpolation, these options would come after a colon, but are otherwise unchanged:

```
f"{12:02x}"  # 0c
"{:02x}".format(12)
```

So, what exactly did this do? Well, if you've been following along with our formatting series, you probably recognise this code from our RGB to hexadecimal number converter. The `02x` tells Python we want the number represented as a 2 digit hexadecimal number, where any missing digits are padded with zeroes.

You can find an exhaustive list of formatting options in the Python documentation I linked earlier. There's far too much to cover in one little post!

Earlier I mentioned that `format` only has one required argument, so what happens when we don't provide a format specification as the second argument?

It's actually identical to passing the value to `str()`. You just get a plain string representation back for whatever value you passed in.

```python
example_tuple = 1, 2, 3
print(format(example_tuple))  # (1, 2, 3)
```

Wrapping up

That's it for the `format` function. I hope you learnt something new, and I'd recommend you take another look at Python's built in functions. There are a whole bunch of neat functions available that we almost never use.

If you're just starting out on your Python journey and you're a bit lost, check out out Complete Python Course. You should also sign up to our mailing list below, as we'll be sending coupons each month to give you the best deals on our Python courses!

Phil Best
I'm a freelance developer, mostly working in web development. I also create course content for Teclado!

Read More

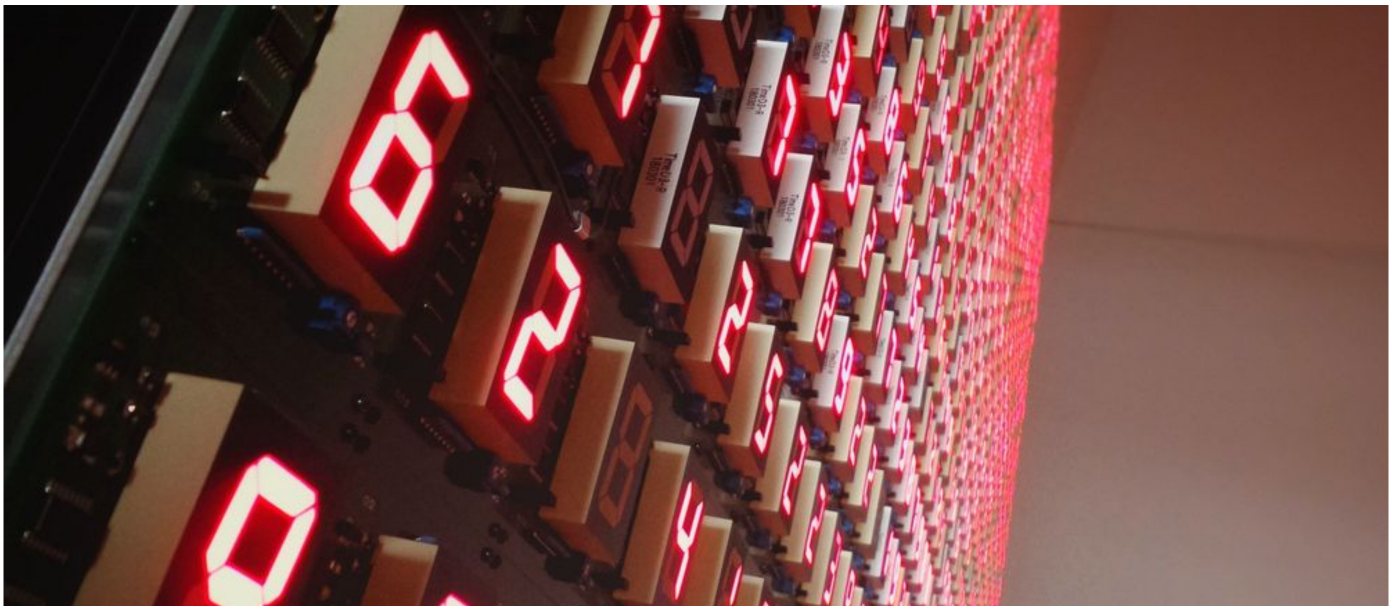
Python Snippets

Assignment expressions in Python →

Python's namedtuples →

Python's divmod Function →

→ See all 26 posts



LEARN PYTHON PROGRAMMING

Decimal vs float in Python

When working with fractions in Python, we tend to use floats, but we also have the decimal module. Learn how to use it, and its benefits over floats.

Phil Best
8 min read · Oct 3

Like what you see? Enter your e-mail to hear when new posts come out!

E-mail*

Name*

Receive our newsletter and get notified about new posts we publish on the site, as well as occasional special offers.

Sign Up