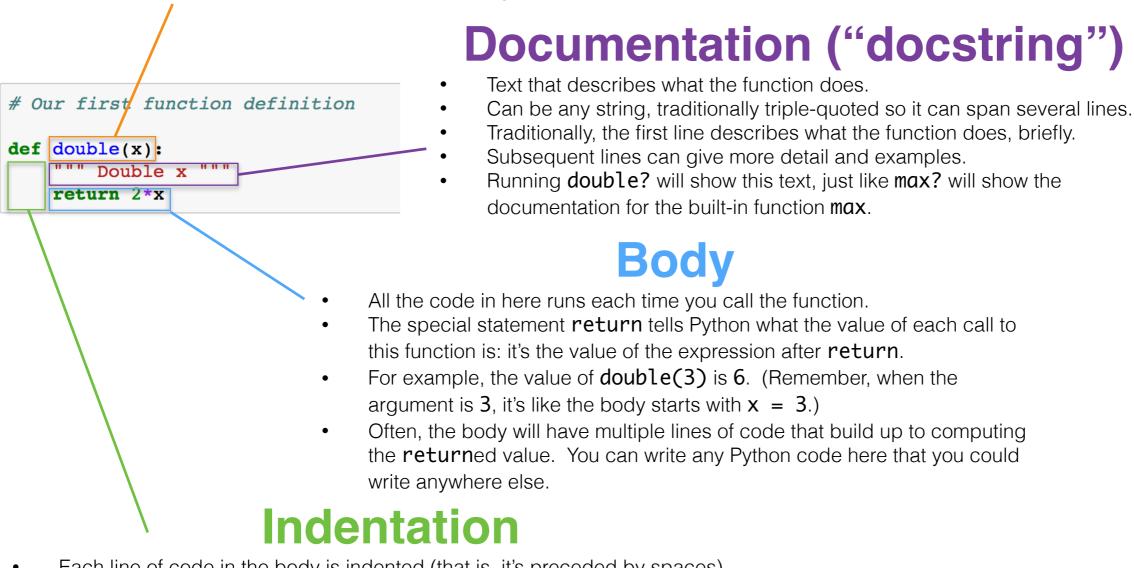# Signature

- Calls to the function will look like this (with the same name and number of arguments). Example: `double(3)`.
- When you call `double`, the argument can be any expression. (The name `x` doesn't affect calls.)
- In the body of the function, `x` is the name of the argument, as if the body included the code `x = <the first argument>`.

Documentation ("docstring")

- Text that describes what the function does.
- Can be any string, traditionally triple-quoted so it can span several lines.
- Traditionally, the first line describes what the function does, briefly.
- Subsequent lines can give more detail and examples.
- Running `double?` will show this text, just like `max?` will show the documentation for the built-in function `max`.

```
# Our first function definition

def double(x):
    """ Double x """
    return 2*x
```

Body

- All the code in here runs each time you call the function.
- The special statement `return` tells Python what the value of each call to this function is: it's the value of the expression after `return`.
- For example, the value of `double(3)` is `6`. (Remember, when the argument is `3`, it's like the body starts with `x = 3`.)
- Often, the body will have multiple lines of code that build up to computing the `return`ed value. You can write any Python code here that you could write anywhere else.

Indentation

- Each line of code in the body is indented (that is, it's preceded by spaces).
- Traditionally, we use 2 or 4 spaces. They only need to be consistent.
- This tells Python that those lines are part of the body.
- The function's body ends at any unindented line.