

# Selecting a Caption Selects the Image

This document is provided to give a 2nd working copy of the same bug as reported in [Python3docs Bugs #2](#), with extra bugs & information supplied.

## Comment

The combo of report#1 + report#2 (below) were made at [BugZilla#164295](#) on Dec 12 2024. The bug was first documented at [GitHub Python3Docs Bug#2](#). After a fortnight there were 3 significant responses to the BugZilla report:

- a) The latest 25.2.0.0.beta1+ version is still affected by this bug  
Version 6.1.0.0.alpha0+ is *unaffected* by the bug
- b) It was introduced at version 6.5 on Fri Dec 6 14:04:55 2019 +0100
- c) Michael Stahl declared that *"this is working as designed"* (good grief)

Bugs #1 + #2 do NOT appear when using LibreOffice 6.1.0.3.  
Bugs #3 + #4 DO appear using 6.1.0.3.

I now understand that the reason that LibreOffice has been riddled with bugs for the last 5 years is *by design* rather than through ignorance, laziness or inability. There now follow the original + updated bug reports:-

*original bug report:*

Selecting the whole of an image caption text (eg for rename) also selects the image. If the caption is then deleted the image itself will also be deleted.

*update bug report 2:*

*If text exists immediately below the image-plus-caption, then selecting the top paragraph of text + it's Pilcrow ('¶' - EOP marker) will also select the image + caption above it. Deleting the text + paragraph will then delete both the paragraph + image + caption.*

*update bug report 3:*

*Placing the cursor behind the first Pilcrow ('¶') below the image+caption, then entering a command to create a new page, creates that page above the image rather than below. Doing the same within any other Pilcrow correctly creates a new page below the paragraph.*

*update bug report 4 (dupe of [bug03.odt](#)):*

*Placing Code Box 2 paragraphs below the image+caption, as in the final page (p10) of this bug-report, illustrates the way that the image inexplicably captures & integrates the 1<sup>st</sup> paragraph of text within itself + mangles it's border. The top-line of the Code Box 2 integral border is thus placed above the image, and the bottom spacing of the image frame (here 0.5 cm) is added above the paragraph text rather than between the two borders of image + text .*

## How to produce this document

1. This document created from scratch under 24.8.3.2.  
`menu:Help | About LibreOffice` shows: [Build](#)
2. All Styles within [Python3docs chapter\\_01.odt](#) then imported into this doc.
3. The 1<sup>st</sup> page from [bug02.odt](#) copied into this doc & later edited.
4. The image+caption “*Figure 2.1: IDLE’s Python Shell*” within [chapter\\_01.odt](#) pasted into a new page (p#7) of this doc (*image is anchored to a paragraph*).
5. 4 paragraphs from p8 of [chapter\\_01.odt](#) copied into another new page (p8).  
(*last 3 paragraphs are Code Box 2 paragraphs, which have an inline border*)
6. Inside yet another new page (p#9), a 2<sup>nd</sup> copy of the same image is pasted into the pilcrow at top of page, then a 2<sup>nd</sup> copy of the same 4 paragraphs pasted into the same pilcrow (which now is below the image).
7. Create a final new page (p#10), paste all the contents of p#9 into it, then remove (what used to be) the top paragraph of p8. That will leave the image+caption plus, below it, the 3 lines that are within a *Code Box 2* paragraph style.

## Demonstrations:

Refer to the final pages for a practical demonstration of this set of bugs.

LibreOffice 6.1.0.3 does NOT contain some of these bugs.

LibreOffice 3.3.0.4 & later all contain bug problem#4.

A set of old AppImages that include that version are available from [libreoffice.soluzioniopen.com](http://libreoffice.soluzioniopen.com). Refer to [AppImage Installation \(Python3docs on Github\)](#) for brief help on installing an AppImage.

*To easily demonstrate problem#1 (see page#7, #9 or #10):*

- i. Click within the Image caption
- ii. Press the <HOME> key  
(cursor should show at front of 'Figure')
- iii. Hold down <SHIFT> key and then press <END> key
- iv. Under LO v7 & later image + caption are now both selected  
(s/b just the caption)

*To easily demonstrate problem#2 (see page#9 or #10):*

- i. Do the same as (i) above, but this time start with the 1<sup>st</sup> letter of the 1<sup>st</sup> line on the paragraph that is below the Image.
- ii. Select to the end of the 1<sup>st</sup> line  
(no problem)
- iii. Select to the end of the whole paragraph  
(Under LO v7 & later the whole of the image above + caption + paragraph below has been selected)  
(that's greedy; deleting the paragraph will now delete everything)

*Note:* This only happens on pages below that contain both an image + text below the image.

*To easily demonstrate problem#3:*

- i. Place your cursor to the left of the 1<sup>st</sup> Pilcrow ('¶') below any image+caption.
- ii. Enter the command to insert a new page  
(this is either `menu:Insert | Page Break` or `kbd:Ctrl+Enter`). Note that the Page Break has been inserted *before* your current page & not *after* it.
- iii. Trying the same command on a page that has more than one Pilcrow, after the 1<sup>st</sup> Pilcrow, will work as expected (which is Page Break *after* the current page).

*To easily demonstrate problem#4:*

- i. Make a visual inspection of the final page (p#10) of the examples below.
- ii. Note that the image+caption-frame bottom spacing (which here is set at 0.5 cm) is placed above the 1<sup>st</sup> line of text below it.
- iii. The text top-border correct location should be between the bottom of the frame & the top of the text border. However, the text has been captured by the image+caption, and the border has been placed above the frame. I believe that this may be a frame error, rather than an image error.

## Encapsulations:

### 1<sup>st</sup> Example Page

Below is a schematic layout of the encapsulations of the 1<sup>st</sup> Example page (page#7). It contains an Image + Caption only. The schema is revealed by `content.xml` within the ODT; see also [bug02+24.8-content.xml](#). A text-only schema for all pages is also within [bug02-info.txt](#).

*Note:* [7-Zip](#) can show + extract files contained within this ODT + other archive files.

```
(new page)
|---<paragraph#1>
| |---<frame#1 (Picture+Caption)
| | |---<text-box>
| | | |---<paragraph#2>
| | | | |---<frame#2 (Picture)
| | | | |---<image />
| | | | |---</frame#2>
| | | | |---<caption>(7 xml lines)
| | | | |---</caption>
| | | |---</para#2>
| | |---</text-box>
| |---</frame#1>
|---</para#1>
```

There is just an `image` + an embedded `caption` on page#7.

`paragraph#1` (`p1`) contains everything on the page. That starts with `frame#1` (`fr1`), and it frames the whole of the image + caption.

`frame#1` contains a `text-box` (`tb`), and the `text-box` contains yet another `paragraph` (`p2`) and that latter contains everything else on the page.

[The schema layout above became so complicated that I was forced to remove most of the colour highlighting (it was making it muddier, not clearer)]

Both the `image` and `caption` are contained within `paragraph#2` (`p2`). The `image` is contained within `frame#2` (`fr2`) and that frame is used to split the `caption` to be either above or (as in this case) below the `image`.

This is how the encapsulations each end up:

<code>image:</code>	<code>p1° fr1° tb° p2° fr2°</code>	<code>image</code>	<code>° fr2° p2° tb° fr1 ° p1</code>	[5 encapsulations]
<code>caption:</code>	<code>p1° fr1° tb° p2°</code>	<code>caption</code>	<code>° p2° tb° fr1 ° p1</code>	[4 encapsulations]

So, example page#7 is very simple (a single picture + it's caption) but that becomes a little complicated as we try to understand the setup details. Those details will become important as we try to analyse the reason for the bugs in later pages.

2<sup>nd</sup> Example Page

The second example page is equally as simple as the first: 4 lines of text. The only wrinkle is that the 1<sup>st</sup> line is a standard *Body Text / Text Body* paragraph, whereas the last 3 lines are *Code Box 2* paragraphs & thus have an embedded blue hairline border & `FreeMono` font. Apart from that they are perfectly normal (Monty Python reference).

The schematic is equally simple:

```
(new page)
<paragraph#3>
plain-text1
</para#3>
<paragraph#4>code-2-text2</para#4>
<paragraph#5>code-2-text3</para#5>
<paragraph#6>code-2-text4</para#6>
```

3<sup>rd</sup> Example Page

The final two pages are just a little bit more complicated, in that the text from the previous page is added to the bottom of the same image copied from page#7. There is an extra bug in this 3<sup>rd</sup> example page (page#9), and the schematic hints at that fact:

```
(new page)
[---<paragraph#1>
|   [---<frame#1 (Picture+Caption)
|   |   [---<text-box>
|   |   |   [---<paragraph#2>
|   |   |   |   [---<frame#2 (Picture)
|   |   |   |   |   ---<image />
|   |   |   |   |   ---</frame#2>
|   |   |   |   |   [---<caption>(7 xml lines)
|   |   |   |   |   [---</caption>
|   |   |   |   |   [---</para#2>
|   |   |   |   |   [---</text-box>
|   |   |   |   |   [---</frame#1>
|   |   |   |   |   [---</para#1>
|   |   |   |   |   ---plain-text1
|   |   |   |   |   ----</para#7>
|   |   |   |   |   ----<paragraph#4>code-2-text2</para#4>
|   |   |   |   |   ----<paragraph#5>code-2-text3</para#5>
|   |   |   |   |   ----<paragraph#6>code-2-text4</para#6>
```

The issue here is that the opening `<paragraph#3>` for `plain-text1` has disappeared entirely. Because of that, I've labelled the closing paragraph (which does not have an opening paragraph) as "`</para#7>`". The final 4 text lines are identical to their source. But remember: according to *Michael Stahl*, this is all "*working as designed*".

## Last Example Page

The only change between the previous example (page#9) & this one (page#10) is that the 1<sup>st</sup> text paragraph below the image+caption on page#9 has been deleted. That allows us to visually see the erroneous way that the image takes the top border of the Code Box 2 text which — remember — is actually supposed to be *below* the image, and places that border *above* the image. However, also remind yourself that coder *Stahl* has clearly stated that this is all “*working as designed*” (phew! thank goodness for that; to my ignorant mind it all looks as buggy as hell).

The schematic drawn from content.xml is below. If you compare it to the 3<sup>rd</sup> example page you will notice that an extra level of hell has been introduced, in that not only has ‘*plain-text1*’ + ‘</para#7>’ been removed (that is what we asked for) but also that ‘<paragraph#4>...</para#4>’ has been removed (did not ask for that) + ‘*code-2-text2*’ has been added *inside* ‘para#1’ (definitely did NOT ask for that).

We have thus ended up with some kind of strange abortion. But do not worry! “*This is working as designed*”™:

```
(new page)
|----<paragraph#1>
|  |----<frame#1 (Picture+Caption)
|  |  |----<text-box>
|  |  |  |----<paragraph#2>
|  |  |  |  |----<frame#2 (Picture)
|  |  |  |  |  |----<image />
|  |  |  |  |----</frame#2>
|  |  |  |  |----<caption>(7 xml lines)
|  |  |  |  |----</caption>
|  |  |  |----</para#2>
|  |  |----</text-box>
|  |----</frame#1>
|  |----code-2-text2
|----</para#1>
----<paragraph#5>code-2-text3</para#5>
----<paragraph#6>code-2-text4</para#6>
```

## Examples

(see next pages):



```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import SortedDict
>>> file_sizes = SortedDict.SortedDict(key=lambda x: x.lower())
>>> for name in os.listdir("."):
>>>     file_sizes[name] = os.path.getsize(name)

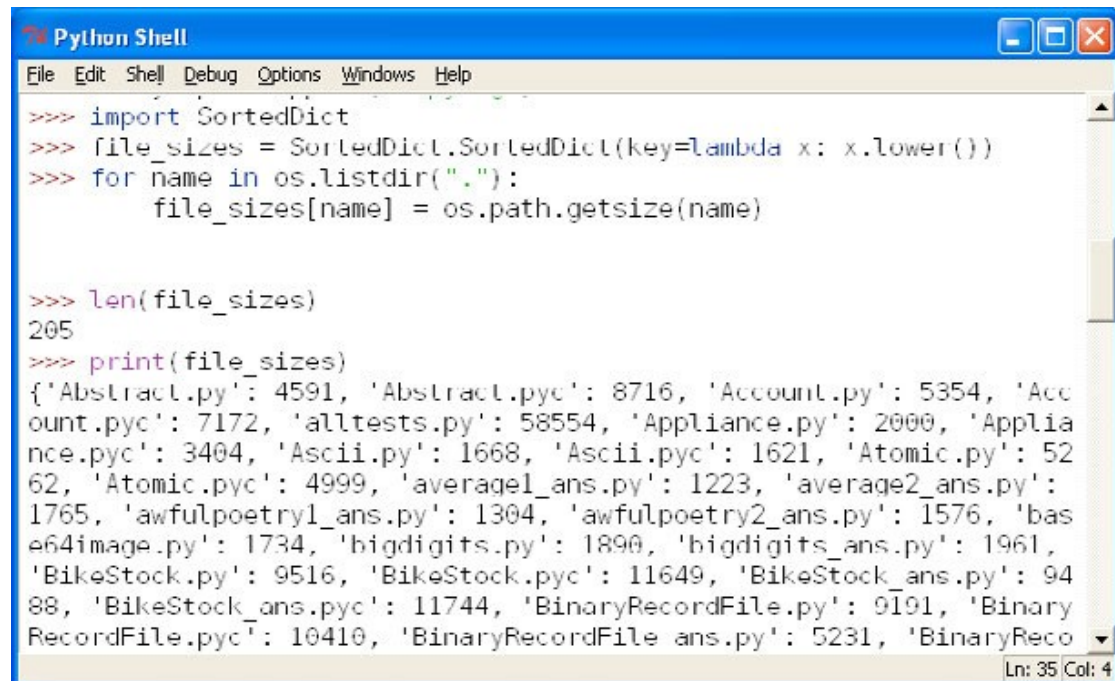
>>> len(file_sizes)
205
>>> print(file_sizes)
{'Abstract.py': 4591, 'Abstract.pyc': 8716, 'Account.py': 5354, 'Account.pyc': 7172, 'alltests.py': 58554, 'Appliance.py': 2000, 'Appliance.pyc': 3404, 'Ascii.py': 1668, 'Ascii.pyc': 1621, 'Atomic.py': 5262, 'Atomic.pyc': 4999, 'averagel_ans.py': 1223, 'average2_ans.py': 1765, 'awfulpoetry1_ans.py': 1304, 'awfulpoetry2_ans.py': 1576, 'base64image.py': 1734, 'bigdigits.py': 1890, 'bigdigits_ans.py': 1961, 'BikeStock.py': 9516, 'BikeStock.pyc': 11649, 'BikeStock_ans.py': 9488, 'BikeStock_ans.pyc': 11744, 'BinaryRecordFile.py': 9191, 'BinaryRecordFile.pyc': 10410, 'BinaryRecordFile_ans.py': 5231, 'BinaryReco
```

Figure 2.2: IDLE's Python Shell

Let's look at a few tiny examples, and then discuss some of the details:

```
x = "blue"  
y = "green"  
z = x
```





```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import SortedDict
>>> file_sizes = SortedDict.SortedDict(key=lambda x: x.lower())
>>> for name in os.listdir("."):
>>>     file_sizes[name] = os.path.getsize(name)

>>> len(file_sizes)
205
>>> print(file_sizes)
{'Abstract.py': 4591, 'Abstract.pyc': 8716, 'Account.py': 5354, 'Account.pyc': 7172, 'alltests.py': 58554, 'Appliance.py': 2000, 'Appliance.pyc': 3404, 'Ascii.py': 1668, 'Ascii.pyc': 1621, 'Atomic.py': 5262, 'Atomic.pyc': 4999, 'averagel_ans.py': 1223, 'average2_ans.py': 1765, 'awfulpoetry1_ans.py': 1304, 'awfulpoetry2_ans.py': 1576, 'base64image.py': 1734, 'bigdigits.py': 1890, 'bigdigits_ans.py': 1961, 'BikeStock.py': 9516, 'BikeStock.pyc': 11649, 'BikeStock_ans.py': 9488, 'BikeStock_ans.pyc': 11744, 'BinaryRecordFile.py': 9191, 'BinaryRecordFile.pyc': 10410, 'BinaryRecordFile_ans.py': 5231, 'BinaryReco
```

Figure 2.3: IDLE's Python Shell

Let's look at a few tiny examples, and then discuss some of the details:

```
x = "blue"
y = "green"
z = x
```



```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import SortedDict
>>> file_sizes = SortedDict.SortedDict(key=lambda x: x.lower())
>>> for name in os.listdir("."):
>>>     file_sizes[name] = os.path.getsize(name)

>>> len(file_sizes)
205
>>> print(file_sizes)
{'Abstract.py': 4591, 'Abstract.pyc': 8716, 'Account.py': 5354, 'Account.pyc': 7172, 'alltests.py': 58554, 'Appliance.py': 2000, 'Appliance.pyc': 3404, 'Ascii.py': 1668, 'Ascii.pyc': 1621, 'Atomic.py': 5262, 'Atomic.pyc': 4999, 'average1_ans.py': 1223, 'average2_ans.py': 1765, 'awfulpoetry1_ans.py': 1304, 'awfulpoetry2_ans.py': 1576, 'base64image.py': 1734, 'bigdigits.py': 1890, 'bigdigits_ans.py': 1961, 'BikeStock.py': 9516, 'BikeStock.pyc': 11649, 'BikeStock_ans.py': 9488, 'BikeStock_ans.pyc': 11744, 'BinaryRecordFile.py': 9191, 'BinaryRecordFile.pyc': 10410, 'BinaryRecordFile_ans.py': 5231, 'BinaryReco
```

Figure 2.4: IDLE's Python Shell

```
x = "blue"
y = "green"
z = x
```