

Selecting Image Caption also Selects the Image

This document is provided to give a 2nd working copy of the same bug as reported in [Python3docs Bugs #2](#), but with extra bugs highlighted + extra information provided. Note in advance that these are possibly Frame bugs rather than Image bugs.

These bugs do NOT appear within 6.1.0.3. See [BugZilla 164295](#).

The bugs are as follows:-

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. The 1st letter needs to be de-selected to prevent that.

update bug report 2:

In addition, selecting all the text (if it exists) within the 1st paragraph + Pilcrow ('¶' - EOL marker) immediately below the image-plus-caption will also select the image + caption above it. Deleting the text + paragraph will then delete both the paragraph + image + caption.

update bug report 3:

Selecting the first Pilcrow ('¶') below the image+caption, then entering Ctrl+Enter (command to create a new page) creates the new page above the image rather than below. Doing the same within any other Pilcrow correctly creates a new page below the paragraph. Yet another bug.

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

*Placing Code Box 2 paragraphs below the image+caption, as in the final page (p6) of this bug-report, illustrates the way that the image captures the text within the 1st paragraph. The top-line of the Code Box 2 integral border is then coalesced with the bottom frame of the image, and the bottom spacing (here 0.5cm) of that frame is added **above** the paragraph text rather than between the two borders.*

To produce this document the following steps were taken:

1. This document created from scratch under 24.8.3.2.
2. All Styles within [Python3docs chapter_01.odt](#) then imported into this doc.
3. The 1st 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 (p5 of this doc (*image is anchored to a paragraph*)).
5. 4 paragraphs from p8 of [chapter_01.odt](#) copied into another new page (p6). (*last 3 paragraphs are Code Box 2 paragraphs, which have an inline border*)
6. Inside yet another new page (p7), a 2nd copy of the same image is pasted into the pilcrow at top of page, then a 2nd copy of the same 4 paragraphs pasted into the same pilcrow (now below the image).
7. Create a final new page (p8), paste all the contents of p7 into it, then remove (what used to be) the top paragraph of p6. 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 these bugs. A set of old AppImages that include that version are available from libreoffice.soluzioniopen.com. Refer to [AppImage Installation \(Python3docs on Github\)](#) for brief help on installing an AppImage.

To easily demonstrate this problem (1):

- i. Click within an 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. Note that now both image + caption are both selected
(s/b just the caption)

To easily demonstrate this problem (2):

- i. Do the same as (i) above, but this time start with the 1st letter of the 1st line on the paragraph that is below the Image.
- ii. Select to the end of the 1st line
(no problem)
- iii. Select to the end of the whole paragraph
(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 this problem (3):

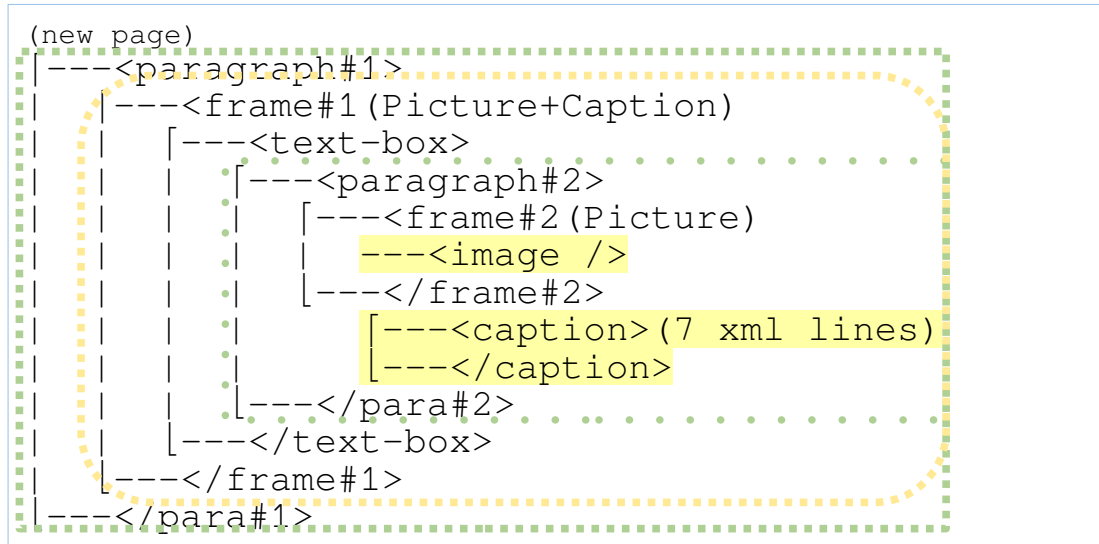
- i. Place your cursor to the left of the 1st Pilcrow ('¶') below any image+caption.
- ii. Enter the command to insert a new page
(this is either `menu:Insert | Page Break` or `kbd:Ctrl+Enter`). Note that the Page Break has been inserted *before* your current page & not *after* it.
- iii. Trying the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

To easily demonstrate this problem (4:)

- i. Make a visual inspection of the final page (p8) of the examples below.
- ii. Note that the image-frame bottom spacing (which is set at 0.5cm) is placed above the 1st line of text below it.
- iii. It's correct location should be between the bottom of the frame & the top of the text border. However, the top of the text border has been captured by the frame above it & is coalesced with the bottom border of that frame.

Encapsulations:

Following is a schematic layout of the encapsulations of the 1st page that contains an Image + Caption () as revealed by `content.xml` within this ODT. Note that [7-Zip](#) is an excellent program to extract files contained within this ODT + other archive files.



As you may tell, there is just an **image** + a **caption** on this page (p5). *paragraph#1* contains everything, starting with *frame#1* which contains everything else. Then, because that is not complicated enough, *frame#1* contains a *text-box*, and the *text-box* contains yet another *paragraph* (why?) and that contains everything else.

This picture is so complicated that I have been forced to stop highlighting with colours & designs to illuminate the design (it's making it worse).

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

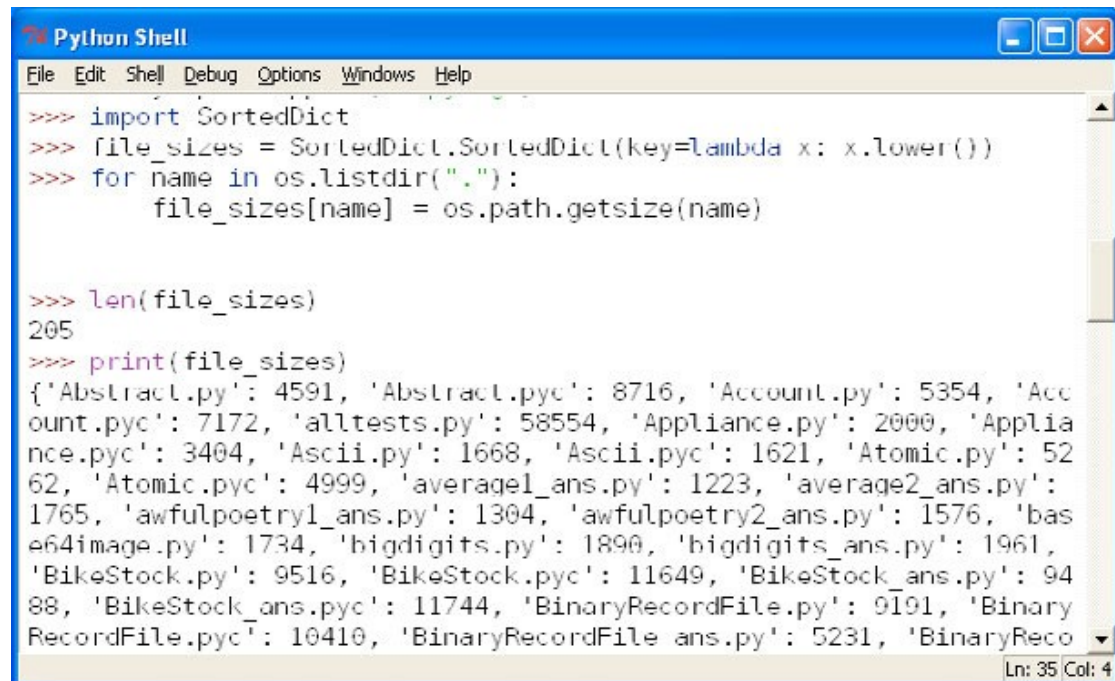
This is how the encapsulations each end up:

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

Last Page

The 1st line of the (added) Text paragraphs is inserted as 'bald' text into the very end of the image+caption (see also [bug02-info.txt](#)). Thus, '`x = "blue"`' is placed into the content *after* `</frame#1>` and *before* `</para#1>`. That line of text thus becomes an integral component of the entire image+caption. The author of that coding has stated that this is "by design". I believe it to be a bug.

Examples (next pages):

The image shows a screenshot of the IDLE Python Shell window. The window has a blue title bar with the text "Python Shell" and standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with options: File, Edit, Shell, Debug, Options, Windows, and Help. The main area of the window contains a text editor with Python code. The code defines a SortedDict, iterates over files in the current directory to calculate their sizes, and then prints the resulting dictionary. The output shows a list of files and their sizes in bytes. The status bar at the bottom right indicates "Ln: 35 Col: 4".

```
>>> 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
```

A screenshot of the IDLE Python Shell window. The window has a blue title bar with the text "Python Shell" and standard window controls. Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main area is a text editor with a white background and a vertical scrollbar on the right. It contains the following Python code:

```
>>> 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
```

The status bar at the bottom right shows "Ln: 35 Col: 4".

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
```

A screenshot of the IDLE Python Shell window. The window has a blue title bar with the text "Python Shell" and standard window controls. Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main area contains a Python script. The script imports SortedDict, creates a SortedDict named file_sizes with a key function that returns the lowercase name of each file, and then iterates over the current directory to populate the dictionary with file names and their sizes. Finally, it prints the length of the dictionary and the dictionary itself. The output shows a dictionary with 205 items, including file names and their sizes in bytes.

```
>>> 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
```