12.2.0 (2026-04-01)

Security

Prevent FITS decompression bomb

When decompressing GZIP data from a FITS image, Pillow did not limit the amount of data being read, meaning that it was vulnerable to GZIP decompression bombs. This was introduced in Pillow 10.3.0.

The data being read is now limited to only the necessary amount.

Fix OOB write with invalid tile extents

Pillow 12.1.1 added improved checks for tile extents to prevent an OOB write from specially crafted PSD images in Pillow >= 10.3.0. However, these checks did not consider integer overflow. This has been corrected.

Prevent PDF parsing trailer infinite loop

When parsing a PDF, if a trailer refers to itself, or a more complex cyclic loop exists, then an infinite loop occurs. Pillow now keeps a record of which trailers it has already processed. PdfParser was added in Pillow 4.2.0.

Integer overflow when processing fonts

If a font advances for each glyph by an exceeding large amount, when Pillow keeps track of the current position, it may lead to an integer overflow. This has been fixed.

Heap buffer overflow with nested list coordinates

Passing nested lists as coordinates to APIs that accept coordinates such as `ImagePath.Path`, `polygon()` and `line()` could cause a heap buffer overflow, as nested lists were recursively unpacked beyond the allocated buffer. Coordinate lists are now validated to contain exactly two numeric coordinates. This was introduced in Pillow 11.2.1.

API changes

Error when encoding an empty image

Skip to content

ode an image with zero width or height would previously raise a `SystemError`. That has now been changed to a `ValueError`.

This does not add any new errors. SGI, ICNS and ICO formats are still able to save (0, 0) images.

API additions

FontFile.to_imagefont()

`FontFile` instances can now be directly converted to `ImageFont` instances:

```
>>> from PIL import PcfFontFile
>>> with open("Tests/fonts/10x20-ISO8859-1.pcf", "rb") as fp:
...   pcffont = PcfFontFile.PcfFontFile(fp)
...   pcffont.to_imagefont()
...
<PIL.ImageFont.ImageFont object at 0x10457bb80>
```

ImageText.Text.wrap

`ImageText.Text.wrap()` has been added, to wrap text to fit within a given width:

```
from PIL import ImageText
text = ImageText.Text("Hello World!")
text.wrap(50)
print(text.text)  # "Hello\nWorld!"
```

or within a certain width and height, returning a new `ImageText.Text` instance if the text does not fit:

```
text = ImageText.Text("Text does not fit within height")
print(text.wrap(50, 25).text == " within height")
print(text.text)  # "Text does\nnot fit"
```

or scaling, optionally with a font size limit:

```
text.wrap(50, 15, "shrink")
text.wrap(50, 15, ("shrink", 7))
text.wrap(58, 10, "grow")
text.wrap(50, 50, ("grow", 12))
```

EXIF tag FrameRate

The EXIF tag `FrameRate` has been added.

Skip to content

Other changes

Support reading JPEG2000 images with CMYK palettes

JPEG2000 images with CMYK palettes can now be read. This is the first integration of CMYK palettes into Pillow.

Lazy plugin loading

When opening or saving an image, Pillow now lazily loads only the required plugin based on the file extension, instead of importing all plugins upfront. This makes `open` 2.3-15.6x faster and `save` 2.2-9x faster for common formats.

Thread safety for free-threaded Python

Critical sections are now used to protect FreeType font objects, improving thread safety when using fonts in the free-threaded build of Python.