

Java



Introduction to annotations



HOME > GUIDES > JAVA > ANNOTATIONS > INTRODUCTION TO ANNOTATIONS >  
WHAT ARE ANNOTATIONS

# Understanding PDF annotations and their types



COPY PAGE



Content displayed as a PDF page isn't suitable for easy editing, but the PDF specification defines a comprehensive set of objects that can be added to PDF pages without changing the page content. These objects are called annotations, and their purpose ranges from marking up page content to implementing interactive features such as forms.

PDF viewers usually allow the creation and editing of various annotation types, e.g. text highlights, notes, lines, or shapes. Regardless of the annotation types that can be created, PDF viewers conforming to the PDF specification should also support rendering for all annotation types.

## PDF annotation types

The PDF specification defines two categories for annotations:

- ✦ **Markup annotations** — Primarily used to mark up the content of a PDF document
- ✦ **Non-markup annotations** — Used for other purposes, including interactive and multimedia.



ASK AI

## Text markup annotations

The simplest types of markup annotations are text markup annotations for marking up page text. These include text highlight, underline, or strikeout annotations.

Nutrient enhances these basic tools by allowing users to customize colors, adjust line thickness, and manage annotation styles through an intuitive interface.

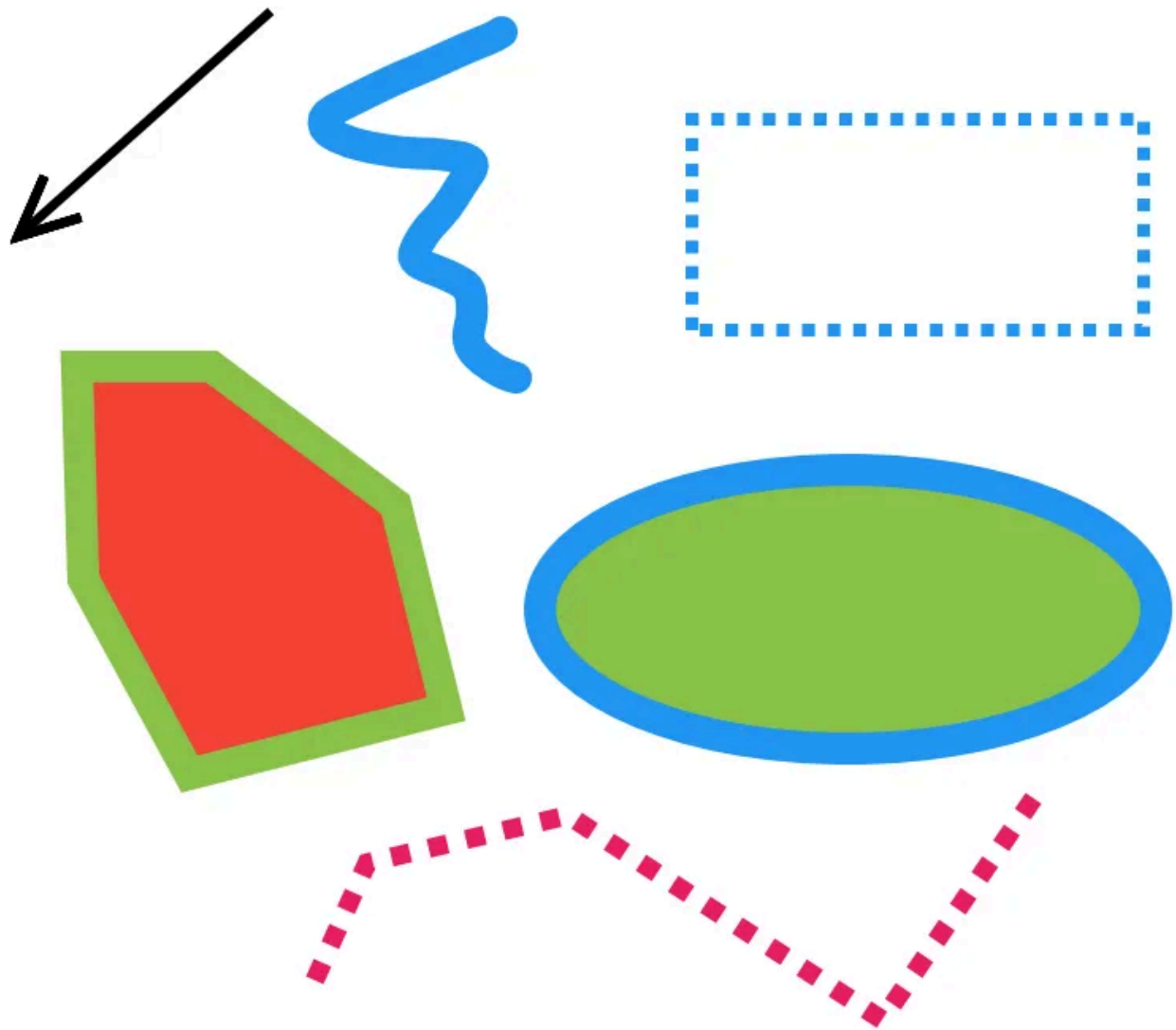
**Highlight annotations** (“Text markup annotations” in the PDF Reference) work on text rects and should thus only be created if there’s actual text on the PDF and not an image. PSPDFKit currently supports **Highlight**, Underscore and ~~Strikeout~~. Highlight annotations have both a boundingBox and an array of rects (the highlighted words). PSPDFKit has some **helpers** that will calculate the smallest fitting boundingBox for the array of rects.

## Drawing annotations

Various drawing annotations can be applied on top of a PDF page, including:

- ✧ **Square and circle annotations** — For drawing squares/rectangles and circles/ellipses.
- ✧ **Line annotations** — For drawing straight lines.
- ✧ **Polygon and polyline annotations** — For drawing polygonal lines.
- ✧ **Ink annotations** — For freeform drawing and connecting points into Bézier curves.

All of these support the usual drawing properties such as color, line thickness, fill color (for filled shapes), or line styles (e.g. dotted, dashed).



Nutrient's robust annotation system makes it easy to create and manipulate these shapes with customizable properties like color, line styles, and fill options.

## Stamp annotations

Stamp annotations can be used when simple shapes drawn with drawing annotations aren't sufficient — for example, when you want to draw a complex raster or vector image.

***APPROVED***

***WITNESS***



Nutrient even supports enhanced stamps that can be saved in the browser's local storage for repeated use, and that can be customized to include user-specific information like IDs or timestamps.

## Text annotations

Two annotation types can be used to add notes to the page:

- ✧ Text annotations for adding sticky notes
- ✧ Free-text annotations for adding a floating text box

**Free-text  
annotation**



Nutrient's annotation tools provide rich text formatting options, such as font selection, text rotation, and justification, allowing for greater flexibility and clarity in note-taking.

## Multimedia annotations

Multimedia annotations let you add interactive elements like audio, video, and even 3D models to your PDF documents. These annotations are crucial for creating engaging and informative content, particularly in educational or technical fields.

# Widget annotations

Widget annotations are used to implement interactive forms within PDFs. Nutrient supports a variety of form elements, including buttons, checkboxes, and combo boxes, enabling the creation of fully interactive PDF forms that can be easily filled out and submitted.

## The PDF annotation model

A PDF document is a collection of data objects organized into a tree structure. The document tree is formed from dictionaries of key-value pairs that can contain other dictionaries as values. An example of another PDF object is a content stream that contains lists of drawing operations or binary image data.

The top-level dictionary in a PDF document contains a list of pages in the document. Each page is represented by its own dictionary that contains separate entries for page content and for annotations associated with the page. Similarly, annotations are also represented by their own dictionaries. Each annotation dictionary contains at least two keys:

- ✧ **Rect** specifies the rectangle where the annotation is going to be positioned on the page. This rectangle is in the PDF page coordinate system, which has its origin in the bottom-left corner of the page, with the x axis pointing to the right and the y axis pointing up. Read more in our [coordinate space](#) guide.
- ✧ **Subtype** specifies one of the supported annotation types.

The annotation type specifies which additional keys can be present in the annotation dictionary and which keys are required.

Below is an example of a page dictionary:

```
46 0 obj
<<
  /Type      /Page          % Specifies that this dictionary defines a page.
  /Annots    [207 0 R, 208 0 R, 209 0 R] % Contains a list of references to annotation objects.
  /Contents  501 0 R         % Reference to page content stream.
  /MediaBox  [0, 0, 595, 842] % Page dimensions.
  % Other page properties
>>
```

Here's an example of a single annotation dictionary:

```
207 0 obj <<
  /Type      /Annot           % Specifies that this dictionary
  /Subtype   /Highlight       % Specifies annotation type.
  /Rect      [52.9556, 728.343, 191.196, 743.218] % Annotation bounding box in
  /C         [1.0, 1.0, 0.0]   % Annotation color.
>>
```

You can find all the details about how annotations are stored inside PDF documents in section 8.4 of [Adobe's PDF specification](#).

## Appearance streams

Annotations may contain properties that describe their appearance — such as annotation color or shape. However, these don't guarantee that the annotation will be displayed the same in different PDF viewers. To solve this problem, each annotation can define an appearance stream that should be used for rendering the annotation. An appearance stream is a set of drawing commands that tells a PDF viewer precisely how to render an annotation (independent of the visual properties defined in the annotation's dictionary).

---

## Was this helpful?

☒ YES

☐ NO

---