Computer practicum

4. Vector graphics

Vector graphics

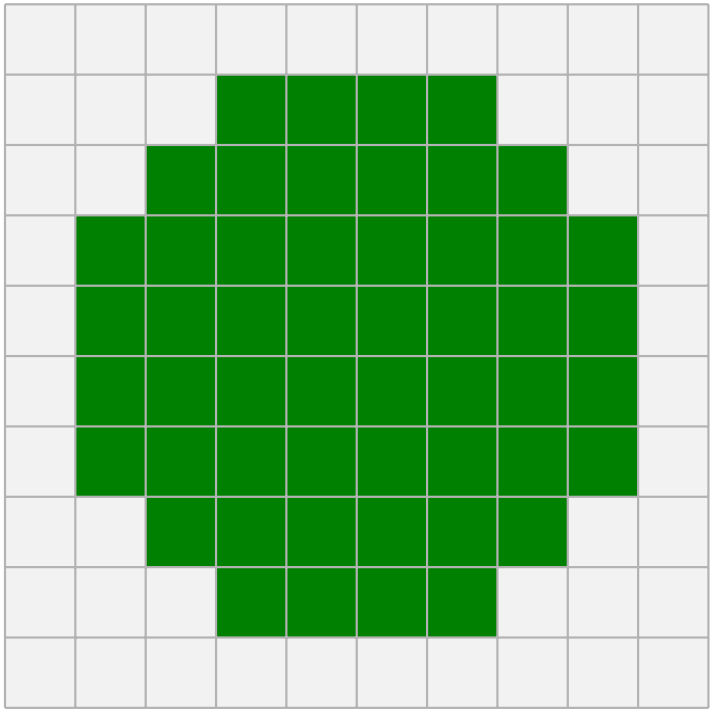
- **Vector graphics** is a type of computer graphics where images are created using various geometric shapes in the plane (2d graphics).
- For example: points, lines, curves, polygons.
- Related to vector graphics are hardware for vector images and printing, vector data models, as well as file formats and software based on these data models.

Where is vector graphics used?

- Graphic design and ilustration
- Geographics information systems (GIS)
- Computer aided design
- 3D computer graphics: including video games and animation.

Comparison with pixel graphics

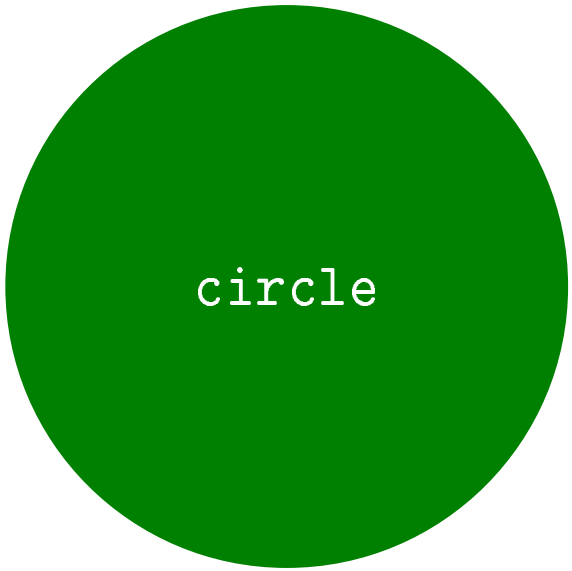
- The image is a **grid of pixels**
- Each pixel stores one colour
- The file is the list of all pixel colours
- Typical for photos



A “circle” made of pixels

Comparison with pixel graphics

- The image is a **description of shapes**
- Example instructions:
  - "circle, centre here, radius 5, green"
  - "line from A to B, black, 2 pt thick"
- The computer follows the recipe whenever it needs to display the image
- Examples: SVG, PDF, EPS



The same circle, as a shape

Comparison with pixel graphics

Vector is good for

- Diagrams and schematics
- Plots and charts
- Logos and icons
- Text and labels
- Anything that must be resized or edited later

Pixel is good for

- Photographs
- Microscopy images
- Gels, scans, screenshots
- Images with fine, irregular detail

Advantages and disadvantages of vector graphics

Advantages

- No quality loss when scaling
- Small files for simple drawings
- Every object stays separately editable
- Text remains real text

Limits

- Not suited for photos
- Very complex drawings get large and slow
- Needs to be converted to pixels to appear on a screen

Data models

- **Logic data models** of vector graphics rely on mathematics, in particular, geometry. Shapes are defined as a series of points in a two or three dimensional koordinate system. E.g.: $A = (x, y)$ or $B = (x, y, z)$.
- Since all the shapes are mathematically composed of an infinite amount of points, vector model defines a finite set of **primitive geometric shapes** that we can describe using a finite number of points or properties.
- **Example:** a square can be uniquely defined using three of its corner points from which we can extrapolate its boundary and inside area. We can also define a square using a single corner point, size and angle of rotation.

Data models

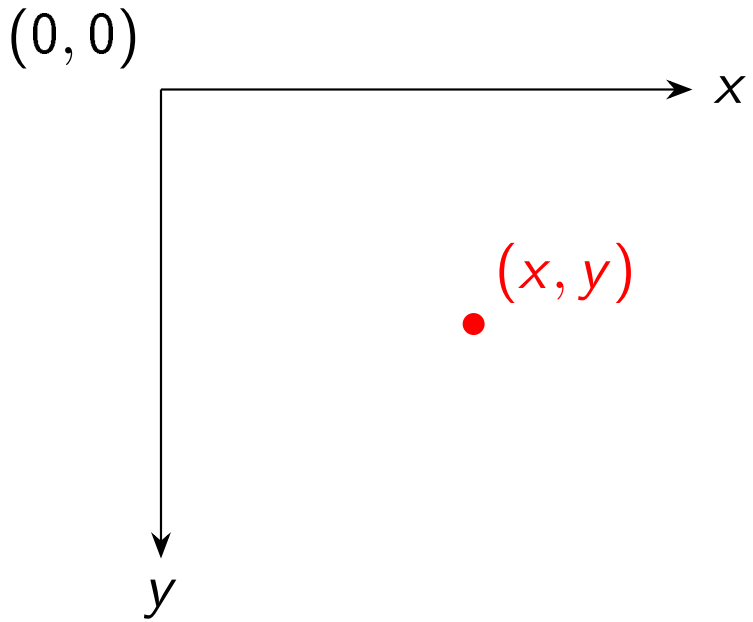
- Some primitive geometric shapes:
  - **Point**
  - **Line**, defined with two points
  - **Polygonal chain** (or polyline), defined with a sequence of points
  - **Polygon**, polyline with the same start and end points with clearly defined inner region.

Data models

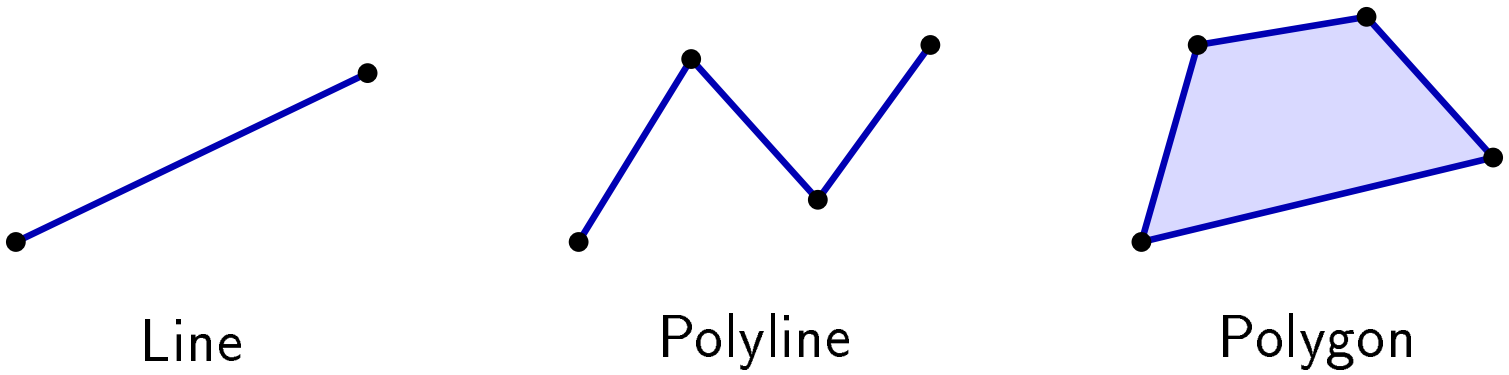
- There are also more complex shapes:
  - **Parametric curves**
  - **Standard parametric shapes** are: circle, ellipse, and many others
  - **Irregular 3D shapes**
  - **Fractals**
  - **Bézier curves**
  - **...**

Points

- A position is given by two numbers: x (across) and y (down)
- Like coordinates on a map or a position on a microscope stage
- The numbers need not be whole numbers: (3.7, 12.25) is fine
- On screens the origin $(0, 0)$ is usually the **top-left corner**



Line, polyline, and polygon



Line Polyline Polygon

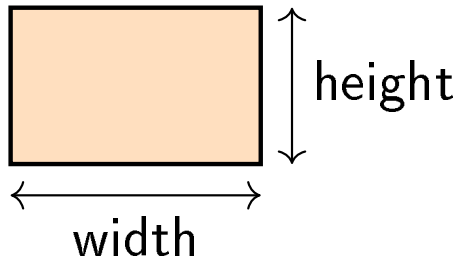
- **Line:** two points
- **Polyline:** several points connected one after another (an open chain)
- **Polygon:** a polyline that closes back to its start, so it can be filled

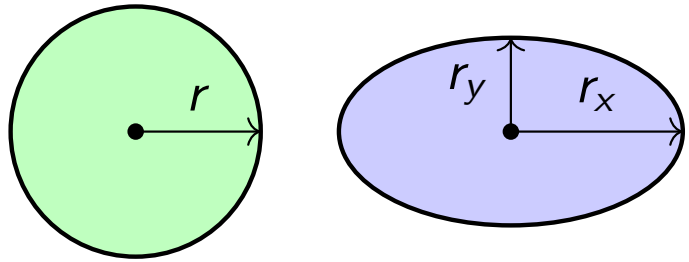
Rectangles, circles and ellipses

Described by only a few numbers:

- **Rectangle:** position of a corner, width, height
- **Circle:** position of the centre, radius
- **Ellipse:** centre, horizontal radius, vertical radius

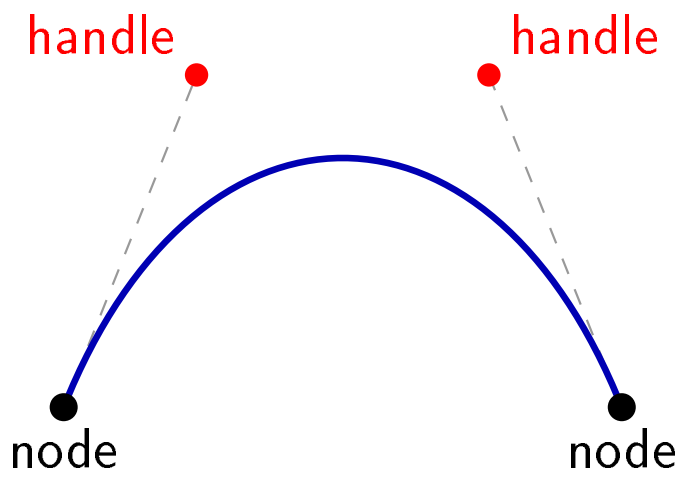
Very compact: a circle needs 3 numbers, whatever its size.





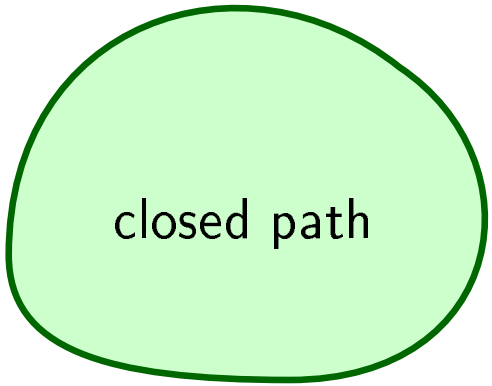
Curves: paths with handles

- Real shapes (cells, leaves, letters) are smooth and curved
- Vector programs describe curves with **nodes** and **handles**
- Nodes are the points the curve passes through
- Handles pull the curve in a direction (like bending a flexible wire)
- These are called **Bézier curves**: you do not need the maths, just the idea



Paths: the universal shape

- A **path** is a chain of nodes joined by straight or curved pieces
- Open path (a line, a curve) or closed path (a shape that can be filled)
- Any shape can be turned into a path — for example a circle becomes four curved pieces
- Letters in a font are closed paths, too!



Fractals: shapes made by repeating a simple rule

- A **fractal** looks similar at every zoom level (*self-similarity*)
- Built by applying one simple rule again and again
- A short rule gives endless detail, so the "recipe" stays tiny
- Found in nature (approximately): ferns, Romanesco broccoli, bronchial tree, blood vessels, neuron branches, coastlines

Style: outline and fill

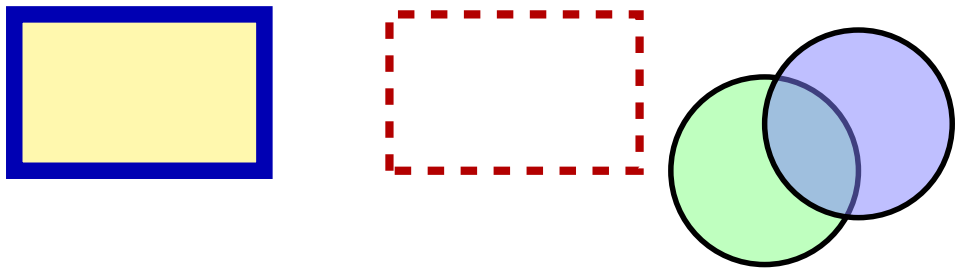
A shape is only geometry; its look comes from **attributes**:

Fill (the inside)

- Colour
- Transparency
- Gradient (colour fade)
- No fill

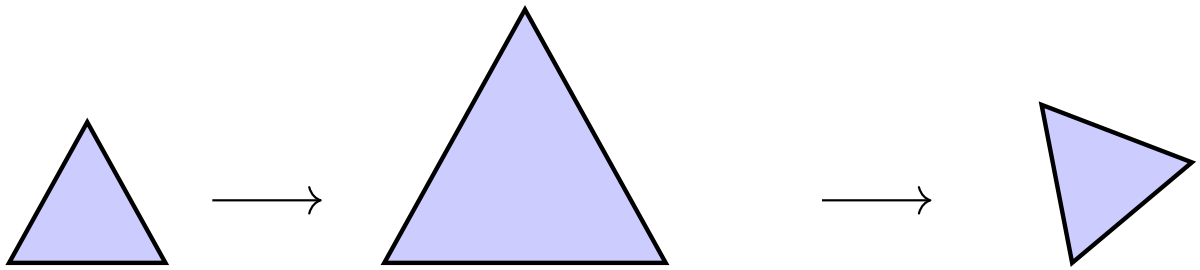
Stroke (the outline)

- Colour
- Thickness
- Dashed or solid
- Rounded or pointed corners



Moving, scaling and rotating

- Because the image is a list of shapes, editing is easy
- **Move:** shift all points by the same amount
- **Scale:** make everything larger or smaller
- **Rotate** and **flip**
- The program just recalculates the points — nothing is lost
- Contrast with a raster image: scaling needs to *invent* new pixels

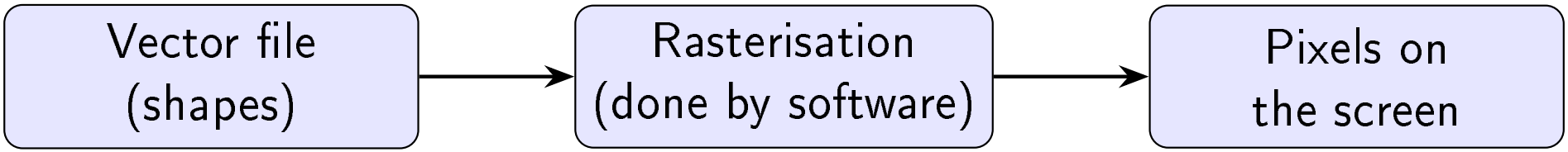


Layers and groups

- Objects are drawn **in order**: later objects cover earlier ones (stacking order)
- **Groups** treat several objects as one (move or scale a whole figure part together)
- **Layers** organise a figure: e.g. one layer for the background, one for labels
- Handy for scientific figures: switch labels on or off, or translate them to another language

The problem

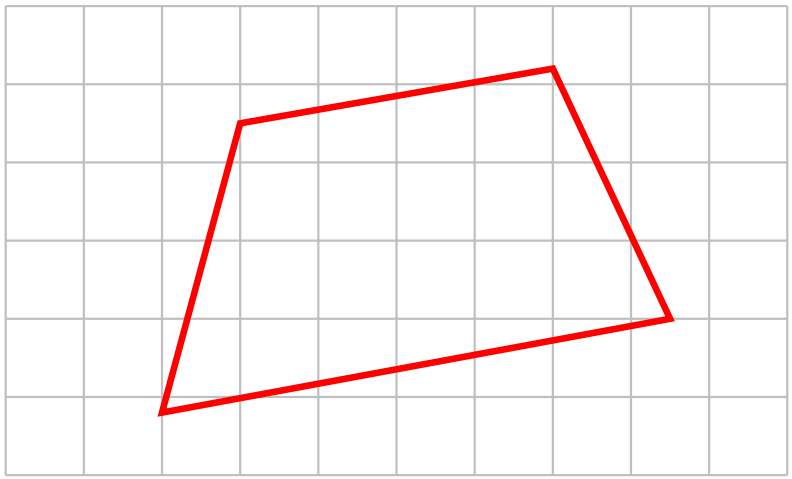
- A vector image is a list of shapes with exact positions
- A screen (or printer or projector) is **a grid of pixels**
- So every vector image must be converted to pixels before you can see it



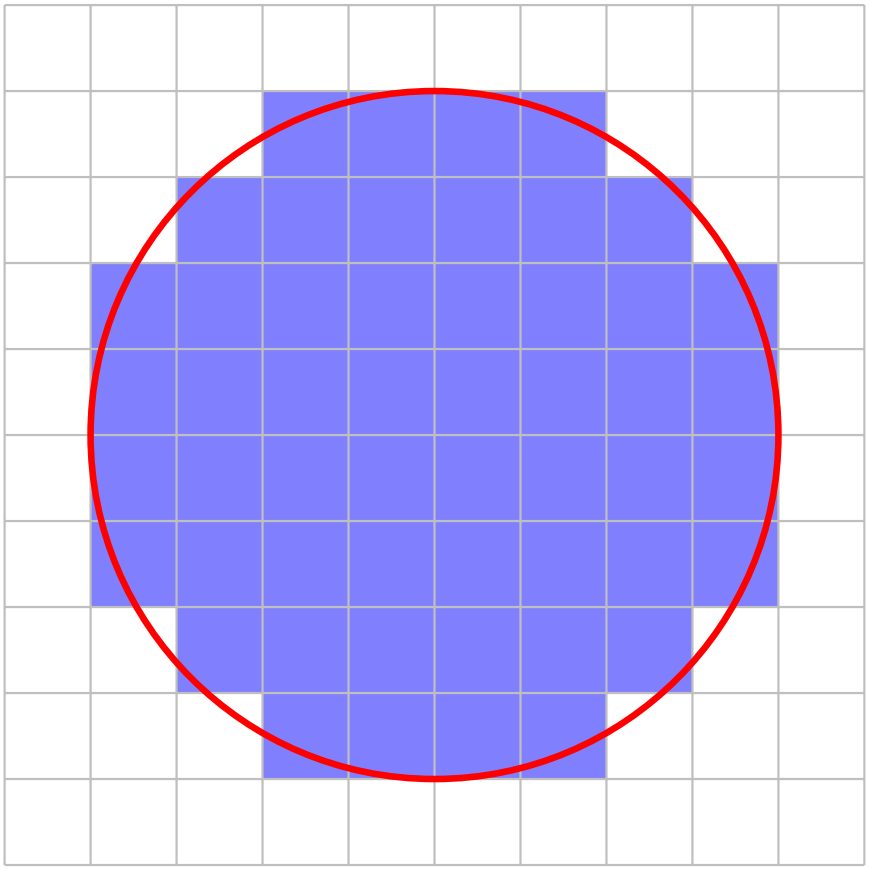
This conversion is called **rasterisation** (or *rendering*).

Rasterisation, step by step

1. Place the shape over the pixel grid at the current zoom level
2. For each pixel, decide: is it inside the shape?
3. Colour the pixels that are inside with the fill colour
4. Do the same for the outline (a thin shape around the edge)
5. Repeat for every object, from bottom to top of the stacking order



Pixels inside the shape get coloured

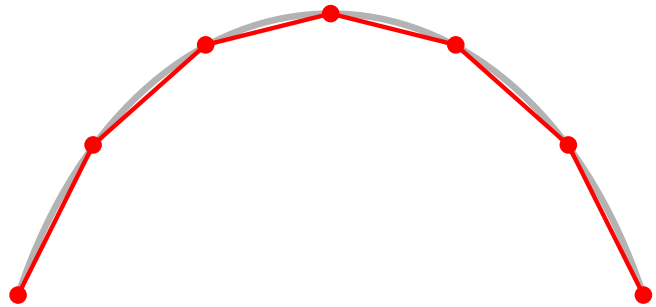


The red outline is the exact shape; the blue squares are what the screen can show.

Result: jagged, staircase-like edges.

Curves become short straight pieces

- Before filling, the software approximates a curve by many tiny straight segments
- Zoomed out: few segments are enough
- Zoomed in: more segments are used, so the curve stays smooth
- You never notice, because it is redone every time you zoom



Resolution: pixels per inch

- A vector image has no fixed resolution
- The resolution is chosen only when it is shown or exported
- The same vector figure can be rasterised for a small web icon *and* for a large poster
- For a raster image the resolution is fixed when you create it (e.g. camera)

Rule of thumb: keep the vector original; export raster copies as needed.

How is the vector file written down?

```
<svg xmlns="http://www.w3.org/2000/svg"
     width="200" height="120">
  <rect x="10" y="10" width="80" height="50"
        fill="lightblue" stroke="navy" stroke-width="2"/>
  <circle cx="140" cy="35" r="25" fill="orange"/>
  <line x1="10" y1="100" x2="190" y2="100"
        stroke="black" stroke-width="3"/>
  <text x="20" y="85">Hello</text>
</svg>
```

- It is just **text** you can open in a text editor
- One line per object: shape, position, colours
- Whole picture: a few hundred bytes

Common vector formats

Format	What it is good for
SVG	Open standard for the web; icons, diagrams, plots; readable text file
PDF	Documents and print; can contain vector and raster content; accepted by nearly all journals
EPS	Older print format; still requested by some journals
AI	Adobe Illustrator's own format
EMF/WMF	Microsoft Office/Windows formats; useful when exporting from Word/PowerPoint

SVG format

- **S**calable **V**ector **G**raphics: open standard based on XML (Extensible Markup Language)
- Supported by all web browsers, Inkscape, Illustrator, Affinity, and many more
- Can be edited in a graphics program *or* a text editor
- Can include text, gradients, transparency, even animation
- Can also contain embedded raster images
- Many plotting tools (R, Python/Matplotlib, GraphPad) can export SVG

PDF format

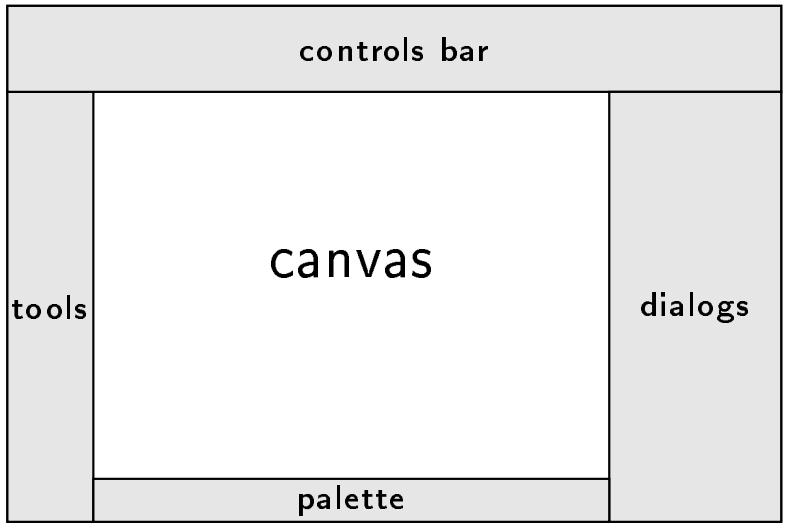
- **P**ortable **D**ocument **F**ormat, created by Adobe, now an open ISO standard
- Describes whole **pages** that look the same on every computer and printer
- A page can contain **text**, **vector shapes** and **raster images** together
- Fonts can be embedded, so text displays correctly even if the font is not installed
- Vector parts stay sharp at any zoom; embedded raster images do not
- Supports transparency, gradients, layers, and print colours (CMYK)
- Accepted by nearly all journals; the standard choice for print and posters
- Many tools export it: R, Python (Matplotlib), GraphPad, Word, PowerPoint, Inkscape
- Can be opened and edited in Inkscape (one page at a time)
- Check: zoom in to 800%. If text and plots stay sharp, they are vector; a screenshot saved as PDF is only a raster image inside a PDF

What is Inkscape?

- A free and open-source vector graphics editor
- Available for Windows, macOS and Linux: `inkscape.org`
- Uses SVG as its native format
- Alternative to Adobe Illustrator or Affinity Designer
- Good for: schematic figures, pathways, cell diagrams, posters, logos

The Inkscape window

- **Toolbox** (left): the tools
- **Canvas** (centre): your drawing and page
- **Controls bar** (top): options of the active tool
- **Colour palette** (bottom)
- **Dialogs** (right): Fill and Stroke, Layers, Align, ...



The most important tools

Tool	Key	Use
Selector	`S`	Select, move, resize, rotate objects
Node tool	`N`	Edit nodes and handles of a path
Rectangle / Ellipse	`R` / `E`	Draw basic shapes
Pen tool	`B`	Draw lines and curves point by point
Text tool	`T`	Add labels
Gradient tool	`G`	Colour fades

- `Ctrl+Shift+F`: Fill and Stroke `Ctrl+G`: group
- `Ctrl+Z`: undo — use it freely!

Useful operations

- **Fill and Stroke:** set colours, outline width, dashes
- **Align and Distribute:** line up and evenly space objects (great for figure panels)
- **Path → Union / Difference / Intersection:** combine shapes into new shapes
- **Object to Path:** turn a shape or text into an editable path
- **Layers:** keep a figure organised
- **Import:** place a microscopy image (PNG/TIFF) and annotate it with arrows, labels and scale bar
- **Trace Bitmap:** convert simple raster images to vector

Exporting from Inkscape

- **Save as...** choose the target format
  - **Inkscape SVG:** keeps everything for later editing
  - **Plain SVG:** for web or other programs
  - **PDF / EPS:** for print and journals
- **Export PNG** (`Ctrl+Shift+E`): choose the area (page, selection) and the DPI
- Tip: convert text to paths if the fonts will not be available elsewhere — but keep an editable copy first

Hands-on exercise (15 min)

1. Draw a simple cell: an ellipse for the cell, a circle for the nucleus
2. Give them fill colours and an outline
3. Add a label with the Text tool and an arrow with the Pen tool
4. Move the nucleus: notice the objects stay separate
5. Group everything, then scale the group up and down
6. Save as SVG; export as PNG at 300 DPI
7. Open the SVG in a text editor and find your ellipse and circle

Further reading

- Inkscape manual and tutorials: `inkscape.org/doc`