Image creation with PHP

- By Kore Nordmann
- PHP Unconference Hamburg
  - 25.04.08

About me

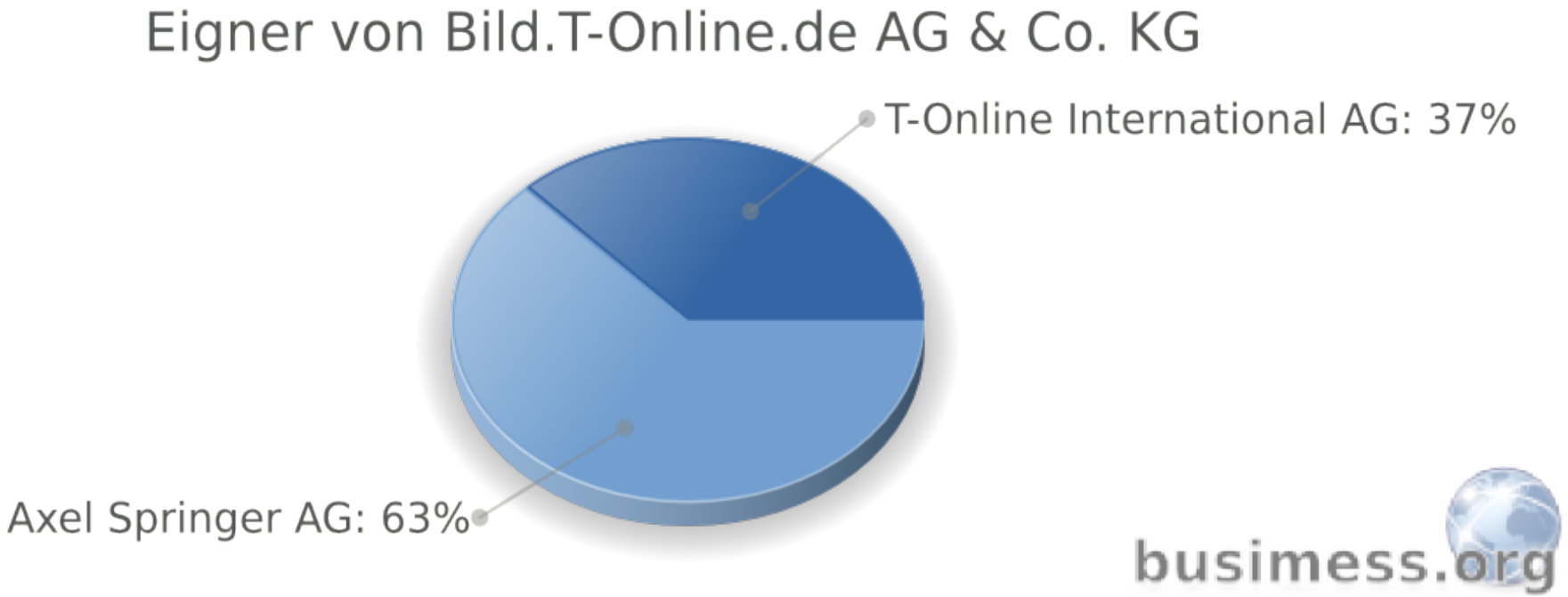
- Kore Nordmann
  - Studying computer science at the University Dortmund
  - Working for eZ systems on eZ components
  - Maintainer and / or Developer in multiple open source projects: Image_3D, KaForkl, ezcGraph, Torii, Busimess, PHPUnit, WCV, ...
    - Image related: Image_3D, KaForkl, ezcGraph

•Goal

- Write a wrapper to different image formats and libraries
- Shows common problems

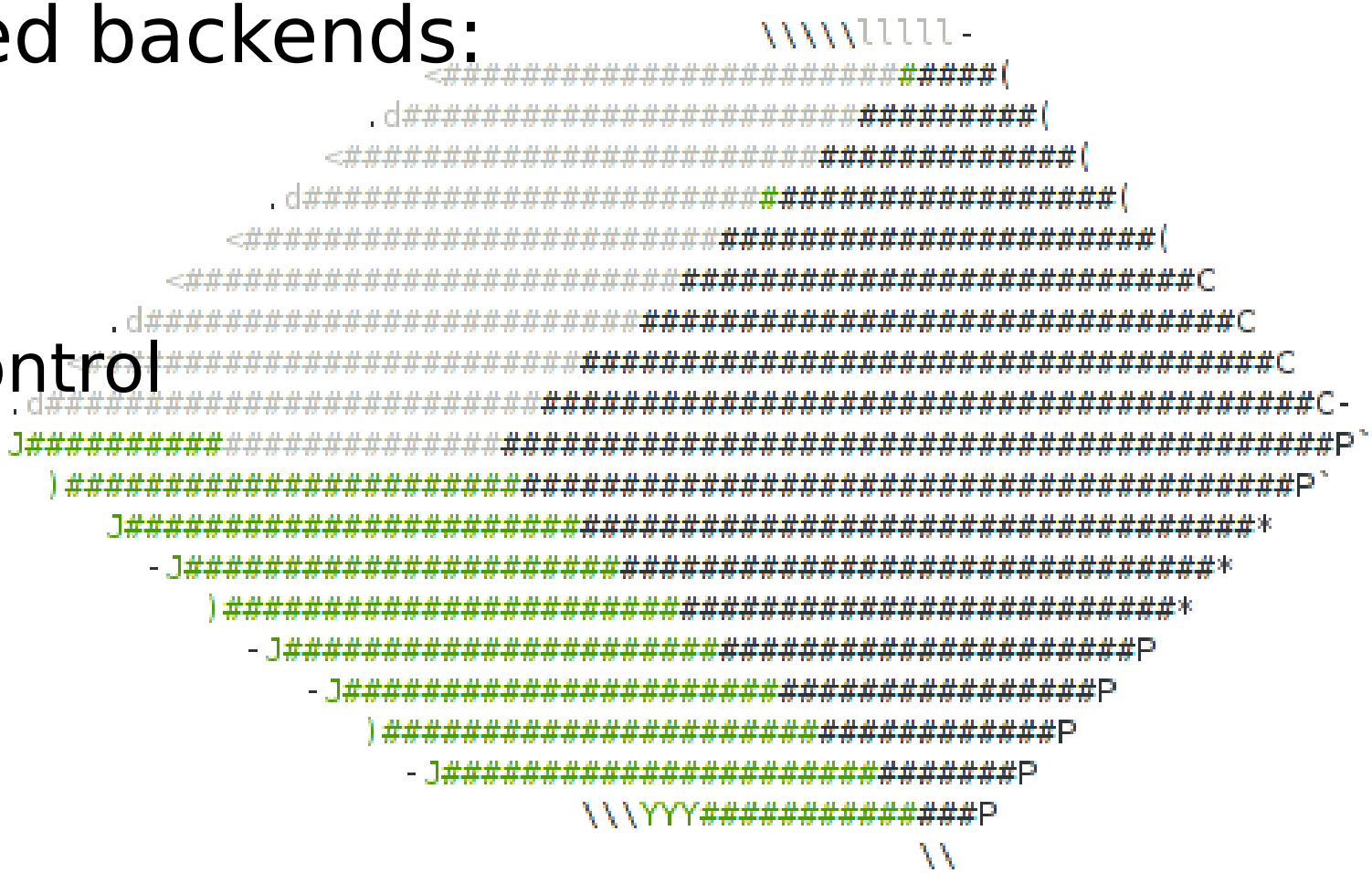
•Usage in ezcGraph

- Supported backends: Flash, GD, SVG, Cairo



•Usage in Image_3D

- Supported backends:
  - GD
  - SVG
  - SVG_Control
  - Canvas
  - ASCII



Terms

- Image
  - Used as a generalization:
- Picture
  - Images with natural contents, like photos or drawings.
  - Usually no clear borders
- Graphic
  - Computer generated graphics with technical illustrations or charts.

Agenda:

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

Formats

- Vector formats
  - SVG
  - Flash
  - PDF / PS / ...
- Bitmaps
  - GIF
  - JPEG
  - PNG
  - BMP / TIFF / ...

SVG (1)

- Scalable Vector Graphics
  - XML based W3C Standard
    - Large files
    - GZIP compressing
- Easy to generate / user readable
- May include other namespaces
- Scripting using ECMAScript

SVG (2)

- Subset of SVG: TinySVG
  - Common subset known by most clients / devices
- Text rendering is (normally) up to the client
  - No absolute text width / size estimation possible.

SVG (3)

- Pro
  - Open standard, common syntax (XML)
  - Lots of clients
- Cons
  - Slight differencies between clients
  - Font issues
- Common usage
  - Long term usable vector graphics

Flash (1)

- Closed standard by Adobe
  - Mainly for animated and interactive web graphics
  - Can be considered as a plain vector format in out case
- Only one real closed source client
  - No support for several platforms
  - Security and privacy issues

Flash (2)

- Pro
  - Rich format installed on a lot of system (98%)
- Con
  - Closed propretary format, clients are not available for all platforms.
  - Bad accessibility
- Common usage
  - Online marketing presentations with mainstream centric target group

PDF / PS / ...

- Also support for vector graphics
- Limited support of graphics features
- Completely different common target group

GIF (1)

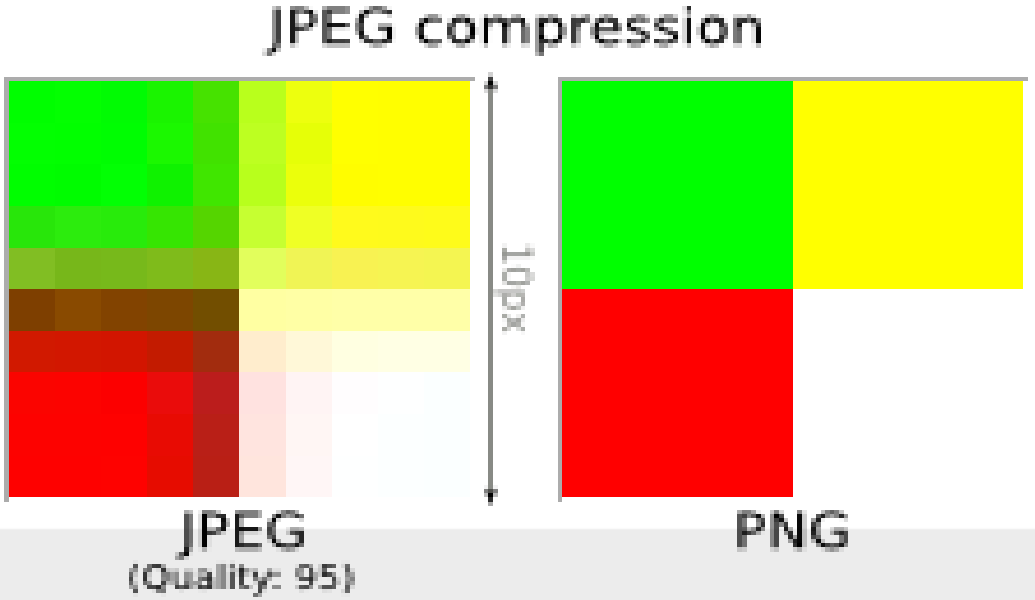
- Patent issues from time to time
- Limited color palette (256 colors)
- No support for semitransparent colors

GIF (2)

- Pro
  - Well known established image format
- Cons
  - Bad transparency support
  - Limited colorspace
- Common usage
  - Web applications, where the author did not know about PNG

JPEG (1)

- (Resolved) Patent issues
- Fourier-Tranformation based compression algorithm
  - Makes it less usable for images with hard edges
- No transparency support



JPEG (2)

- Pros
  - Good compression of pictures
- Cons
  - Bad compresssion and artifacts for graphics.
  - No transparency
- Common usage
  - Pictures

PNG (1)

- Developed and standardized by W3C to replace GIF
  - Ultimate format for graphics in the web
- Lossless compression
- Full RGB colorspace
- 128 alpha channels

PNG (2)

- Pro
  - Well known and established image format
  - Good compression of graphics
  - Full RGB with 128 alpha channels
- Cons
  - Limited support in one browser
- Common usage
  - Graphics in the web.

BMP / TIFF / ...

- Uncompressed, useless, raw formats
  - From a webdeveloper point of view

Agenda:

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

Libraries

- ext/gd
- pecl/cairowrapper
- ext/ming
- ext/dom
- pecl/imagick

ext/gd (1)

- Best known extension for image creation in PHP
  - Well documented
  - Lots of examples available
  - Installed nearly everywhere
- Bitmaps are rendered by the library
  - Image quality completely depends on it

ext/gd (2)

- Low quality rendering
  - No anti-aliasing (in most cases)
  - Broken transparency for ellipses (-sectors)
- No gradient support
- Nearly no anti aliasing
- It's damn slow

pecl/cairowrapper (1)

- Renders 2D vectorgraphics
- Used by
  - GTK (since 2.8.0)
  - Firefox (for SVG in 2.*, completely in 3.*)
- Multiple output formats
  - SVG, PDF, PNG, JPEG
  - XWindow, Win32, Glitz (OpenGL), Quartz

pecl/cairowrapper (2)

- Install
  - `pear install pecl/cairo_wrapper-beta`
  - http://pecl.php.net/package/cairo_wrapper
- Supports everything we want.
  - Full transpareny support
  - Excellent anti aliasing
- Really fast rendering

ext/ming

- Creates flash SWF files
- Alpha state since years
- Support for a subset of flash
  - Not matching any specific flash version
- Wrong or missing documentation is common

ext/DOM

- Implements a well known API for XML handling
- Enables you to modify any part of an document
  - xmlwriter may be faster, but harder to handle (forward only approach)

pecl/imagick

- Latest imagemagick/magicwand wrapper
- Exposes nearly the complete API
  - Mostly for image manipulation, not creation

Agenda:

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

Colors (1)

- Wrap the color definitions
  - Create from HEX or array definitions
- Specify the usage of transparency
  - Library dependent: Transparency vs. opacity
  - We define: 255 = full transparency

Colors (2)

- Source...

Coordinates

- Source?

The base class

- Source

Agenda

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

Surface

- Often used synonym for canvas in this context
  - Drawing area
- Implement the methods initialize() and save() for all backends

DOM & SVG – Initialization

- Creation of the initial XML document
  - Optionally extend an existing document
- Just store using DOMDocument::save()

GD & PNG – Initialization

- Create the ressource
  - Mind the transparency
  - Mind the supersampling
- Supports different output types
  - Merge / resize on save

Cairo & PNG – Initialization

- Create surface and context
  - Pretty straight forward

Ming & SWF – Initialization

- Create a new „movie“
  - Our image is a movie without animation

Imagick – Initialization

- Just create a new image object
- Merge drawing contexts on write

Agenda

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

The base class – extensions

- Add method headers to base class

Setting the style

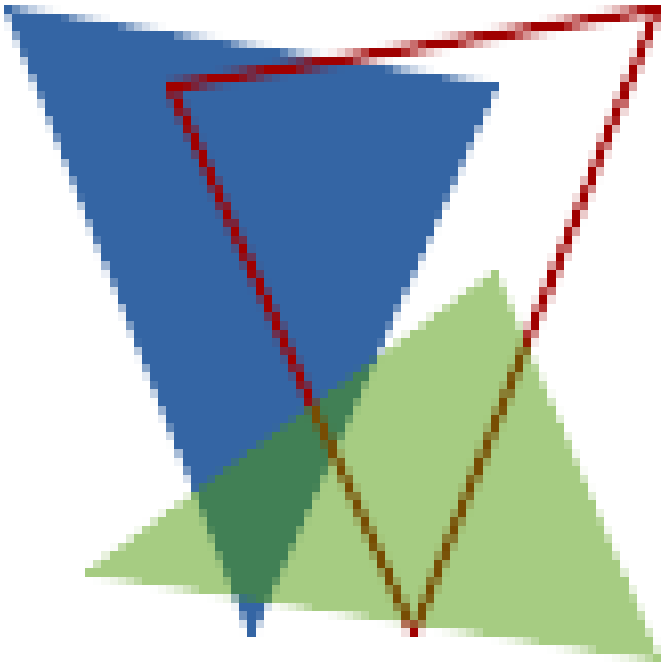
- SVG: Style attribute
- GD: Color and global states
- Flash: Set styles on current shape
- Cairo: Modify drawing context
- Imagick: Modify global drawing context

Create the polygon

- Just a styled path in:
  - SVG, Cairo, Flash, Imagick
- GD has different functions depending on the fill state

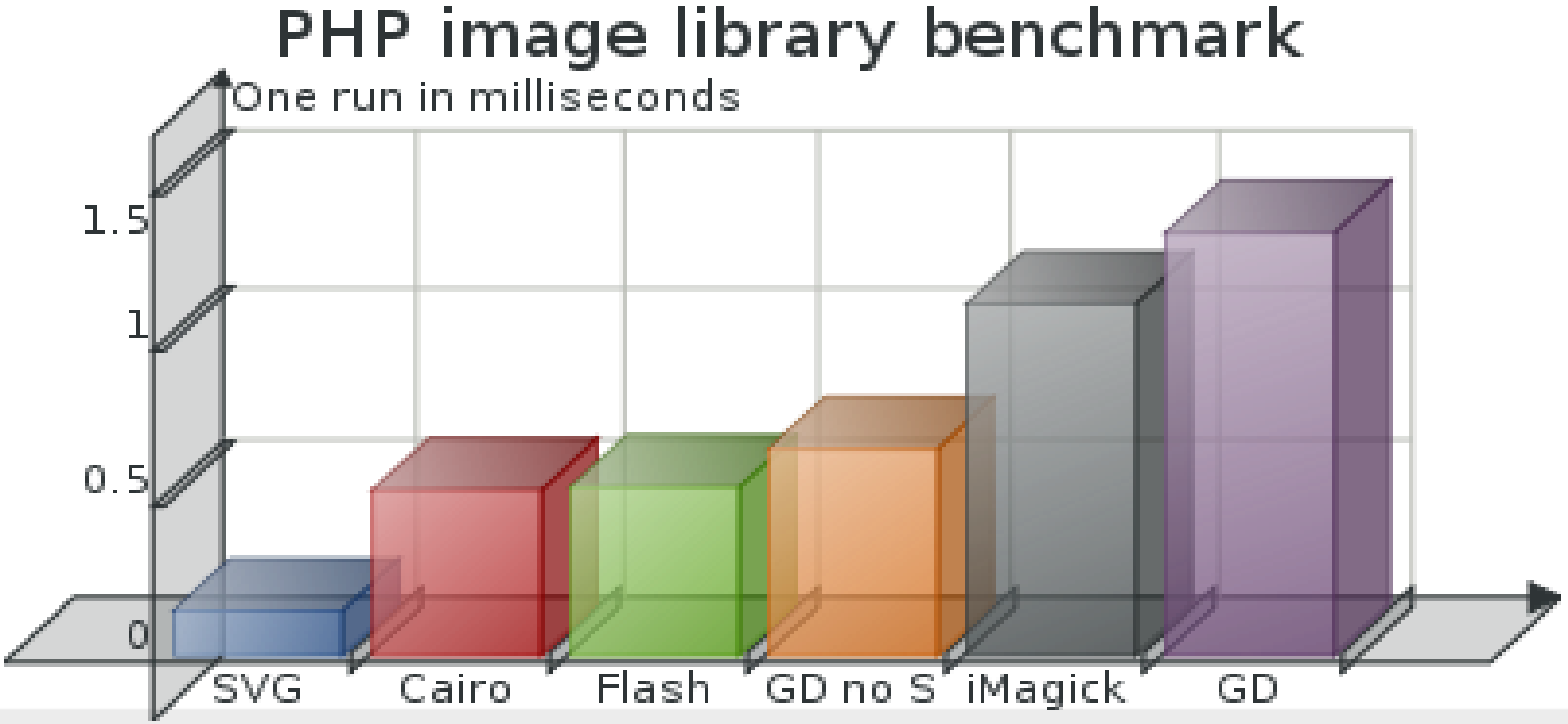
The results: Quality

- Quality:
  - Cairo > Imagick > GD
- Rendered by client:
  - SVG, Flash



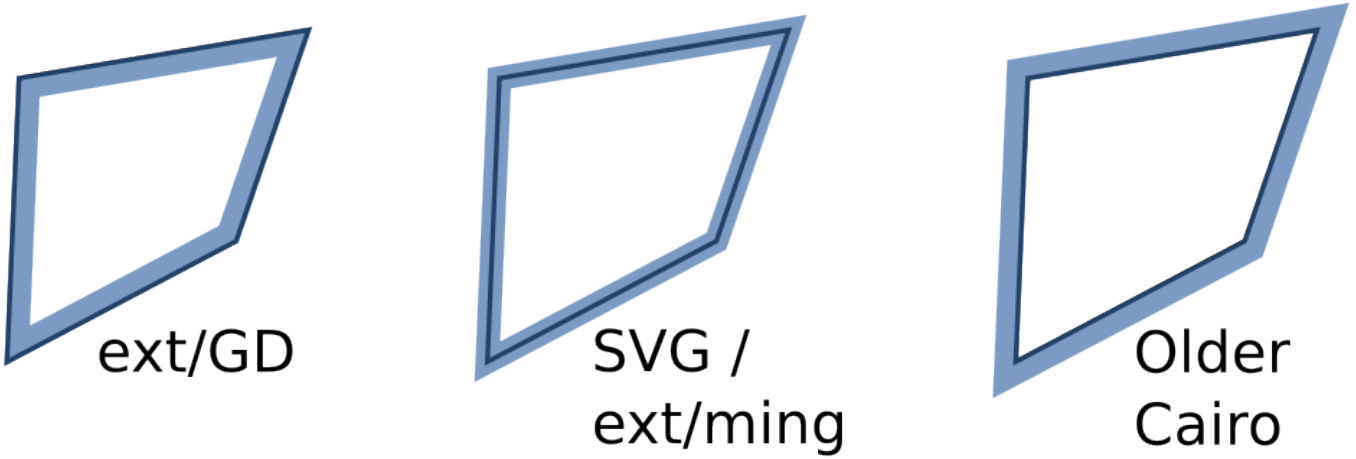
The results: Speed

- Cairo > Imagick > GD

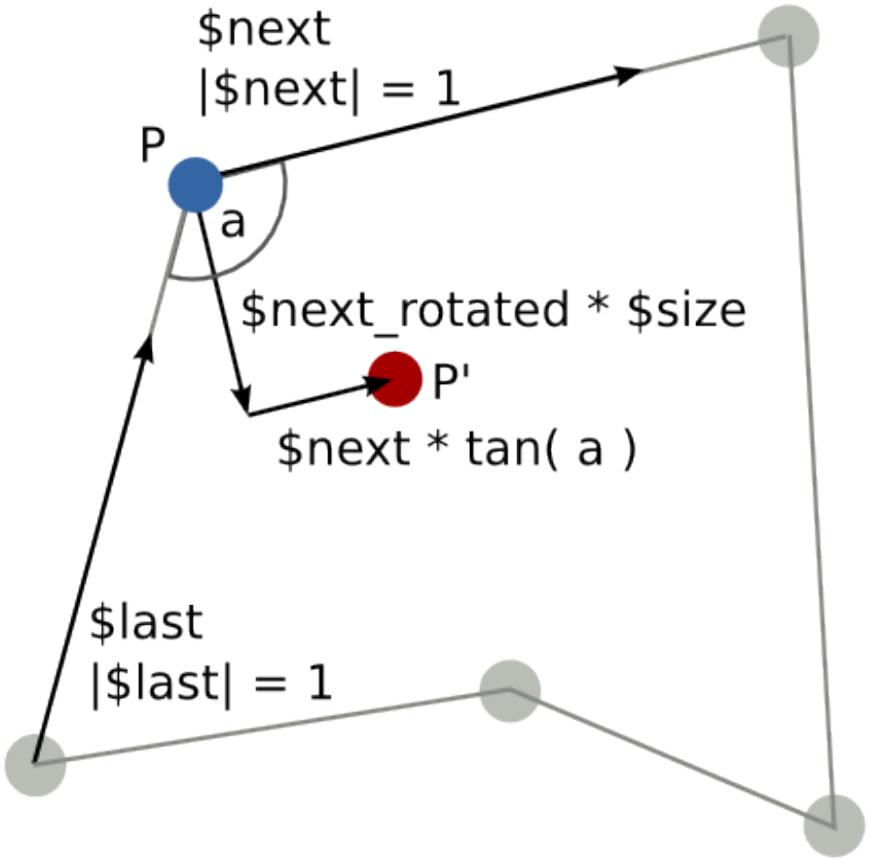


Different border placements

- Not yet noticeable, but: Different backends place borders at different positions
  - User visible for big border widths
  - User visible for overlapping shapes



Different border placements – Theory



Conclusion

- Even the simplest shape has problems when it comes to abstraction
  - Imagine border size reducement for circle sectors
- You may brake down most shapes to polygones
  - Circles, Rectangles, ...
- ezcGraph also implements circles, ellipse sectors and lines

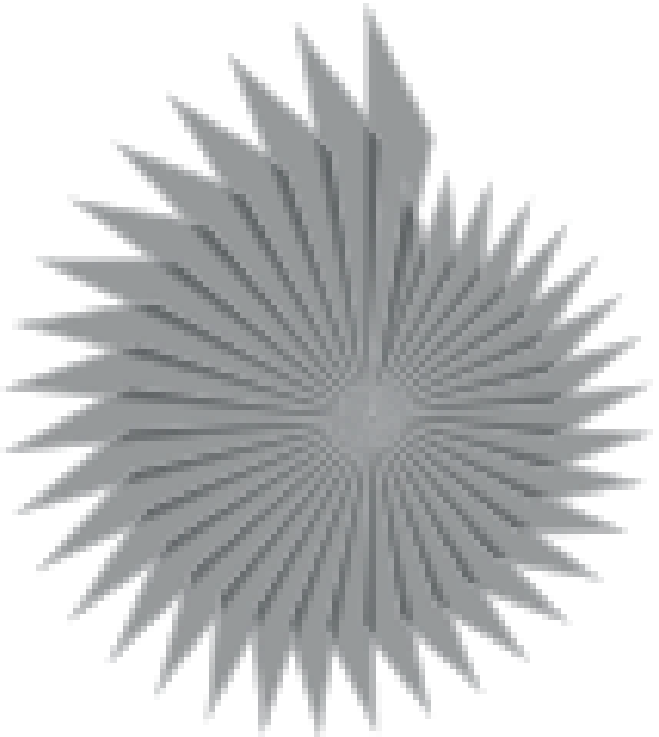
Agenda

- Formats
- Libraries
- Basic datastructures
- Creating the surface
- First shape: polygon
- Extensions

Implemented extensions

- Gradients
  - Radial / linear gradients
    - Not supported by GD and ~Imagick
    - Undocumented in Ming
    - Works fine with SVG and Cairo
- Text rendering
  - Inaccurate in SVG

The result



KoreNordmann

KoreNordmann

http://kore-nordmann.de
Contact: kore@php.net

Questions?

- Please ask!

The end

- Thanks for listening
- Ressources
  - http://kore-nordmann.de/blog/published_article_image_creation_with_php.html