MapServer – Make beautiful maps

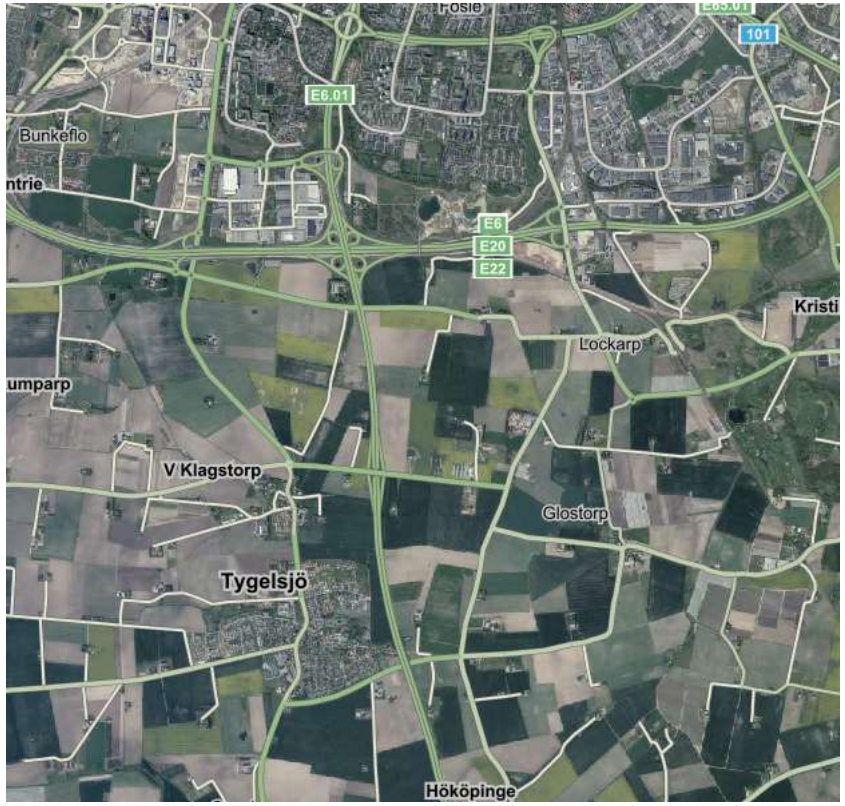
Lars Schylberg, PhD, Saab AB

FOSS4G Florence, 2022-08-25

Krokom
Hägra
Litsnäset
Indalsälven
Storsjön
Östersunds flygplats
Östersund
Staare
Brunflo
Fåker
E45
E14
340
339
321
87

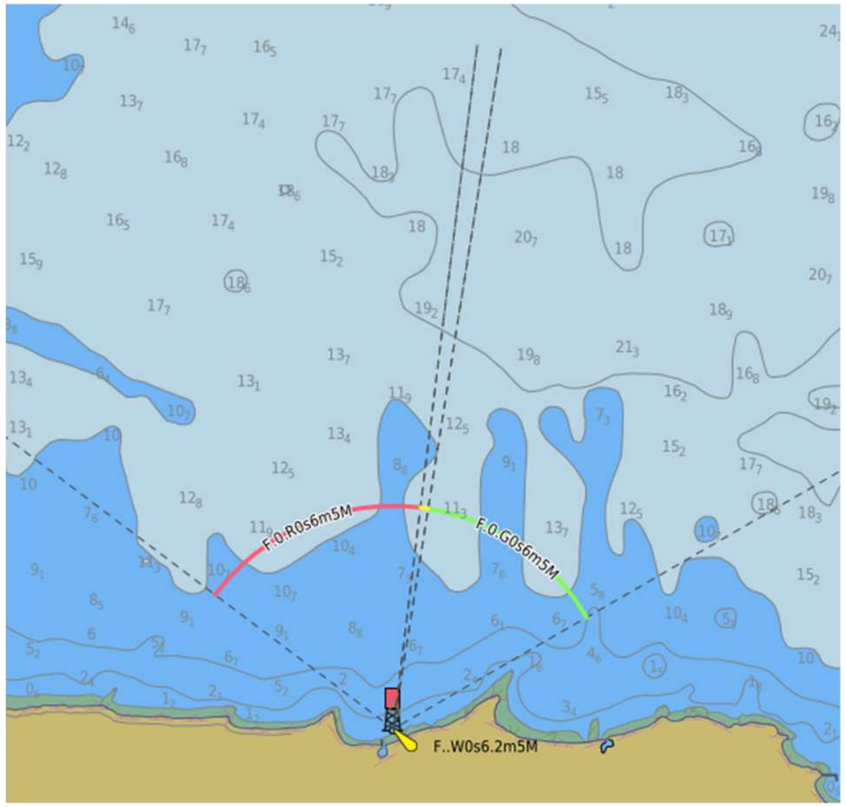
Outline – Some elements of beautiful maps

- Labels
  - Label placement
  - Label shadows that blends with background
- Shadows on objects
  - Polygons (lakes)
  - Point objects (houses)
  - and more …..
- Avoiding text and symbol overlap



My background

- Land Surveyor – PhD in Cartography
- GRASS user/dev 1987 – 1994
- Mapserver user 2001-2005
- Mapserver user again since 2012
- Senior Technical Fellow - Digital Maps at Saab
- SMAC-M on GITHUB – Sea Charts with Mapserver
- Love to author fast and beautiful WMS services

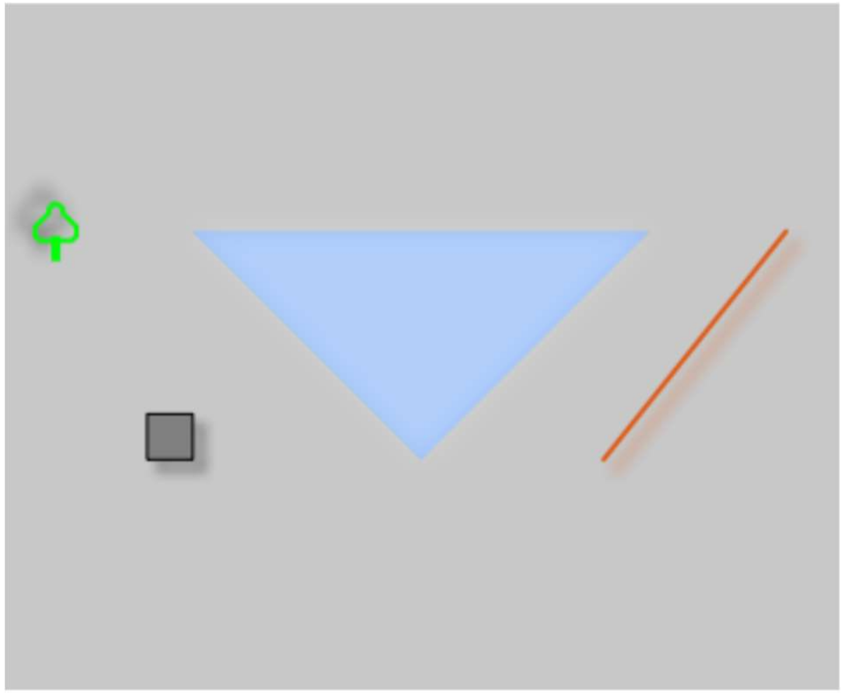


Tools in Mapserver

- Layer Composition Pipeline
- GEOMTRANSFORM "centerline".
- Transparency in text shadows
- Named Styles

Chainable Compositing Filters

- Was introduced in: RFC113 Chainable Compositing Filters (implemented by Thomas Bonfort)
- Was introduced already in MS 7.2
- An effort to add proper documentation was made this spring 2022.
- The primary purpose is to enable soft shadow and blurring effects, although other usages can exist or could be added in the future.
- One other usage is blend hillshades with background



Chainable Compositing Filters

- COMPFILTER [string]

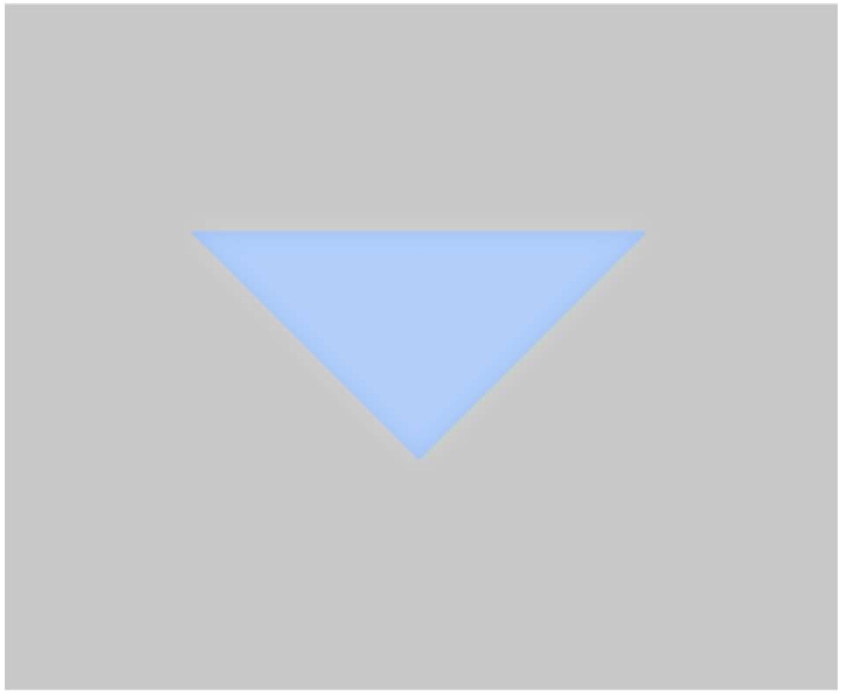
- The currently available filters are:
  - blur(integer)
  - translate(integer,integer)
  - grayscale()
  - blacken()
  - whiten()

Chainable Compositing Filters

- COMPOP [string]
- Name of the compositing operator to use when blending the temporary image onto the main map image. See http://en.wikipedia.org/wiki/Blend_modes. The default compositing operator is “src-over”.
- The COMPOP values that can be used should be explicit and are listed here:
  - clear
  - color-burn
  - color-dodge
  - contrast*
  - darken
  - difference
  - dst
  - dst-atop
  - dst-in
  - dst-out
  - dst-over
  - exclusion
  - hard-light
  - invert*
  - invert-rgb*
  - lighten
  - minus*
  - multiply
  - overlay
  - plus
  - screen
  - soft-light
  - src
  - src-atop
  - src-in
  - src-out
  - src-over
  - xor

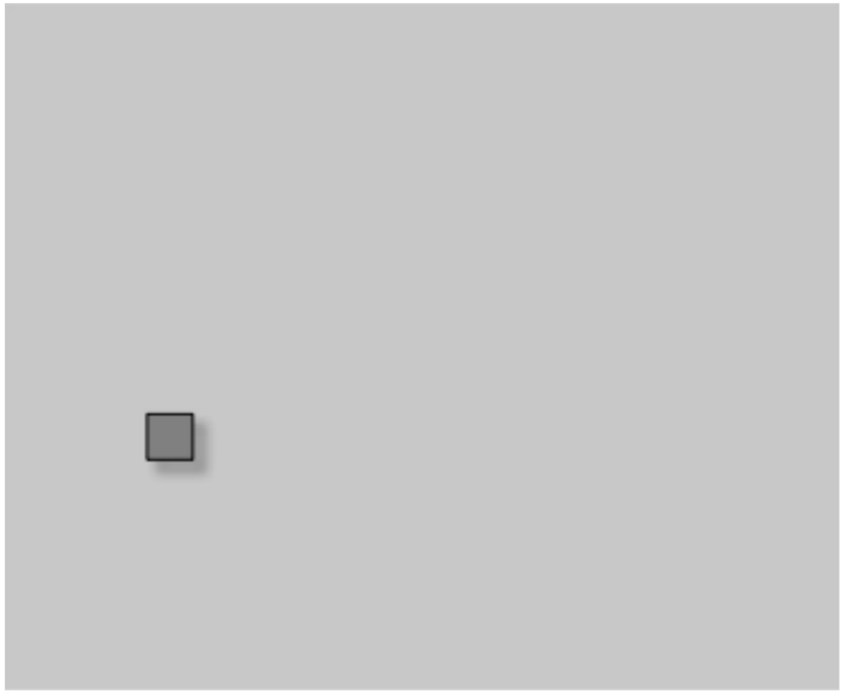
Inside shadows on a lake object

```
LAYER
    NAME "lake"
    TYPE POLYGON
    FEATURE POINTS 10 5 15 10 5 10 10 5 END END
    STATUS ON
    COMPOSITE
        #first we render normal water color
        # OPACITY 100
    END
    COMPOSITE
        COMPFILTER "whiten()"
        COMPFILTER "blur(10)"
        COMPOP "soft-light"
        OPACITY 50
    END
    CLASS
        STYLE
            COLOR 156 192 249  # watercolor
        END
    END
END # Layer
```



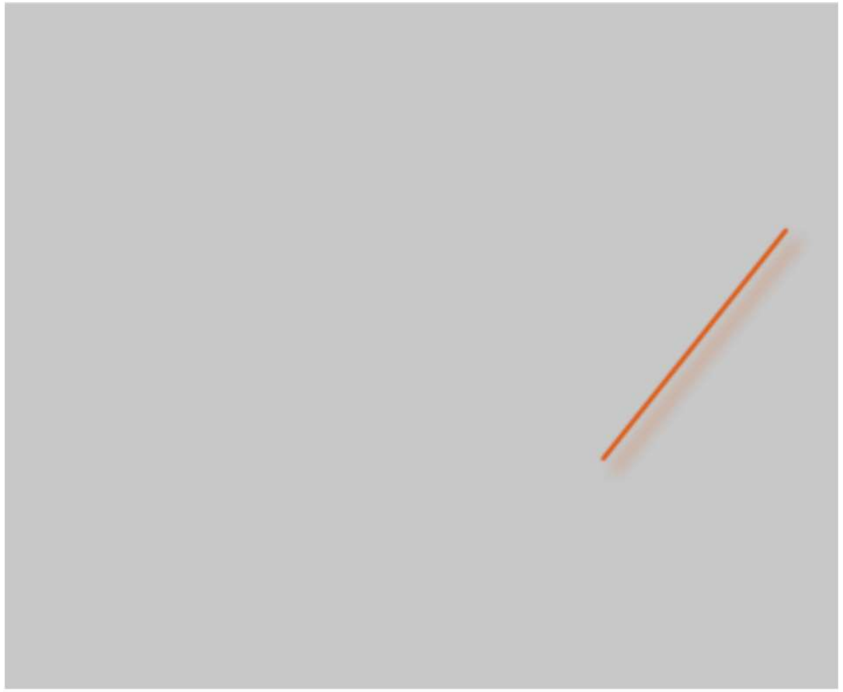
Soft shadow on house object

```
LAYER
    NAME "house"
    TYPE POLYGON
    FEATURE POINTS 4 5 4 6 5 6 5 5 4 5 END END
    STATUS ON
    COMPOSITE
        # create the shadow/blur effect by translating
        # a blurred version of the layer
        COMPFILTER "grayscale()"
        COMPFILTER "translate(5,5)"
        COMPFILTER "blur(4)"
        OPACITY 50
    END
    COMPOSITE
        #and render the buildings themselves
        OPACITY 100
    END
    CLASS
        STYLE
            COLOR 128 128 128
            OUTLINECOLOR 0 0 0
            WIDTH 1
        END
    END
END # Layer
```



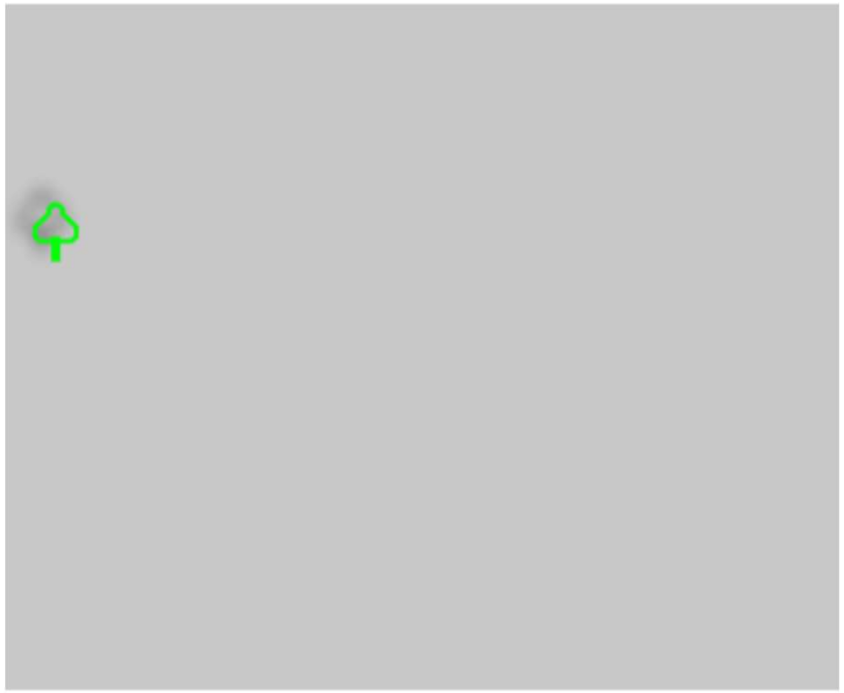
Shadow on a line object

```
LAYER
    NAME "line-test"
    TYPE LINE
    FEATURE POINTS 14 5 18 10 END END
    STATUS ON
    COMPOSITE
        # create the shadow/blur effect by
        #  translating a blurred version of the layer
        # COMPFILTER "grayscale()"
        COMPFILTER "translate(5,5)"
        COMPFILTER "blur(5)"
        OPACITY 50
    END
    COMPOSITE
        OPACITY 100
    END
    CLASS
        STYLE
            COLOR 225 95 31
            WIDTH 2
        END
    END
END # Layer
```



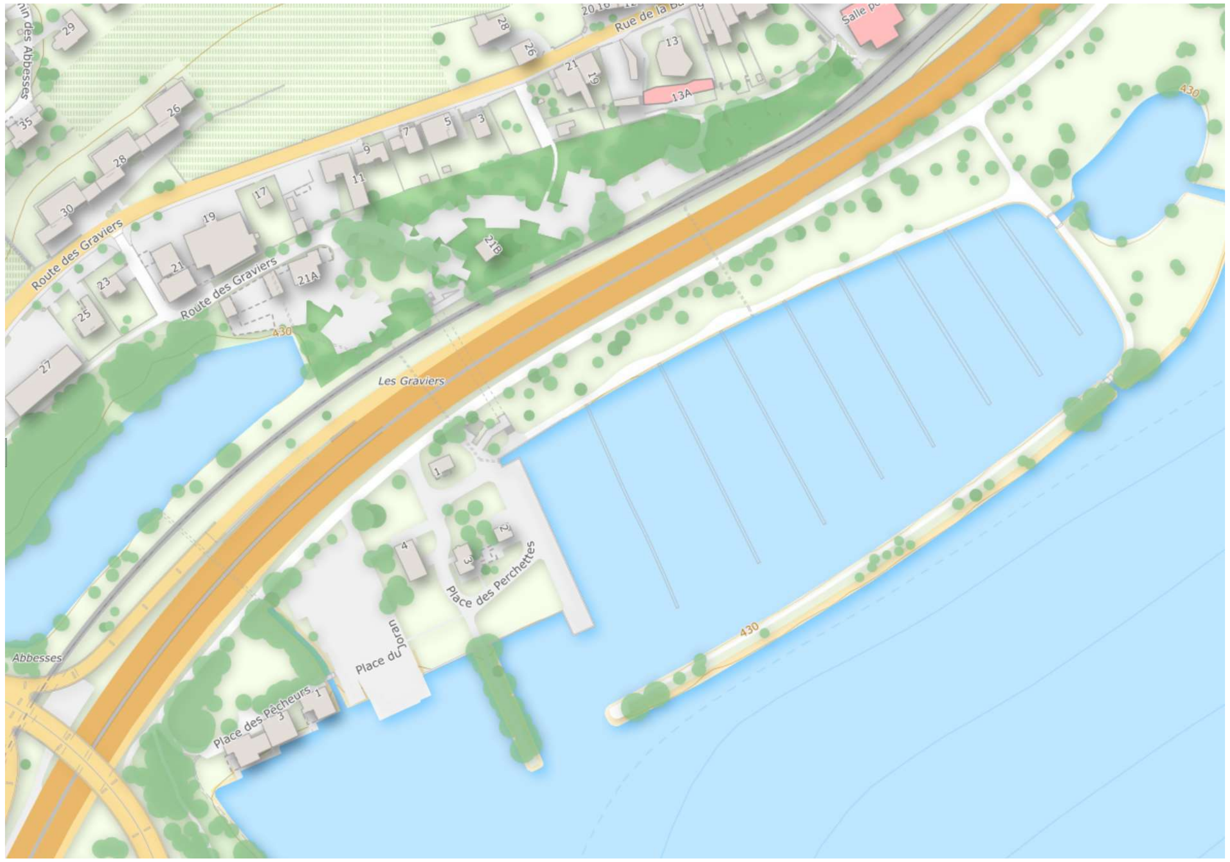
Shadow on a point object

```
LAYER
    NAME "point-symbol-test"
    TYPE POINT
    FEATURE POINTS 2 10 END END
    STATUS ON
    COMPOSITE
        # create the shadow/blur effect by
        #  translating a blurred version of the layer
        COMPFILTER "blacken()"
        COMPFILTER "translate(-6,-5)"
        COMPFILTER "blur(7)"
        OPACITY 50
    END
    COMPOSITE
        OPACITY 100
    END
    CLASS
        STYLE
            SYMBOL "Tree"
            SIZE 24
            COLOR 5 255 10
            WIDTH 2
        END
    END
END # Layer
```

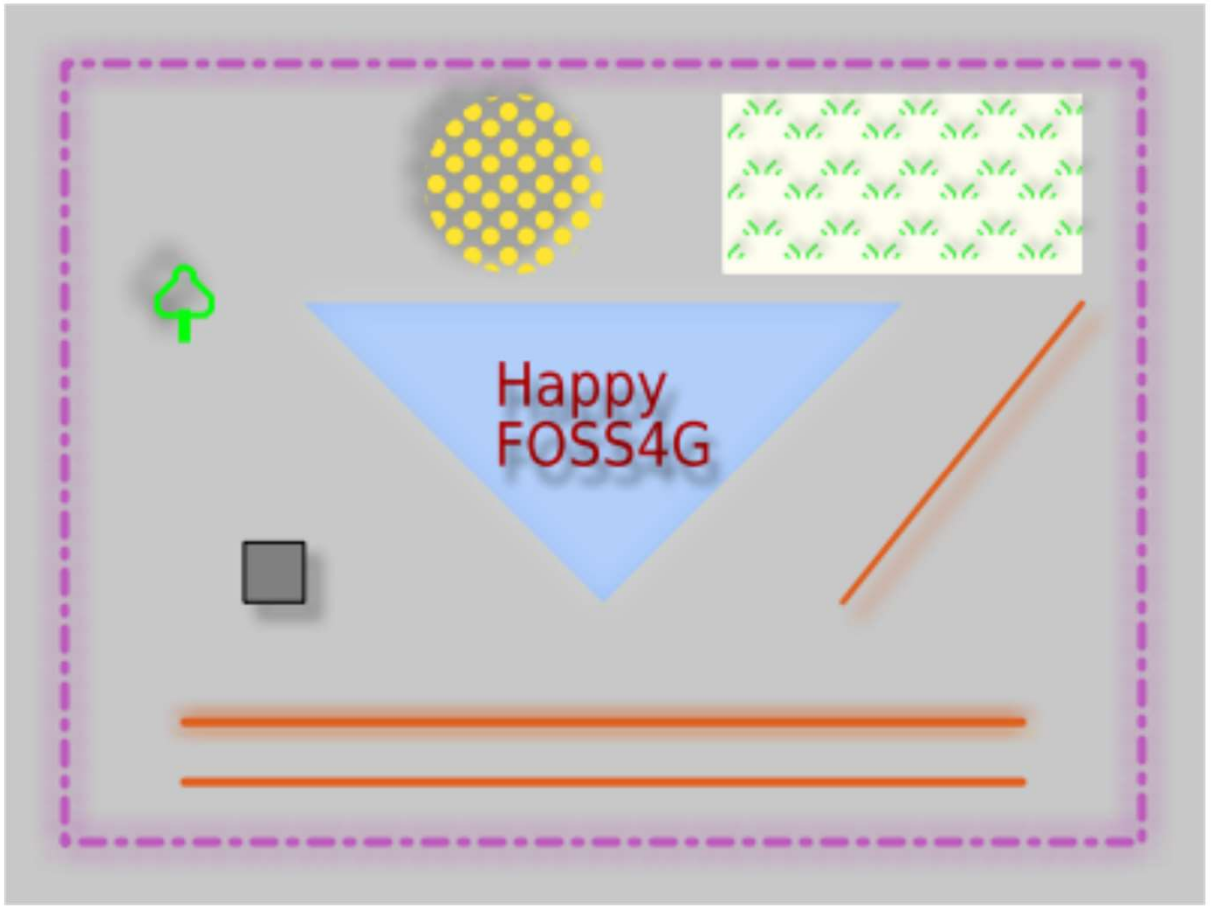


Favorite example: Chainable Compositing Filters

Géoportail du SITN

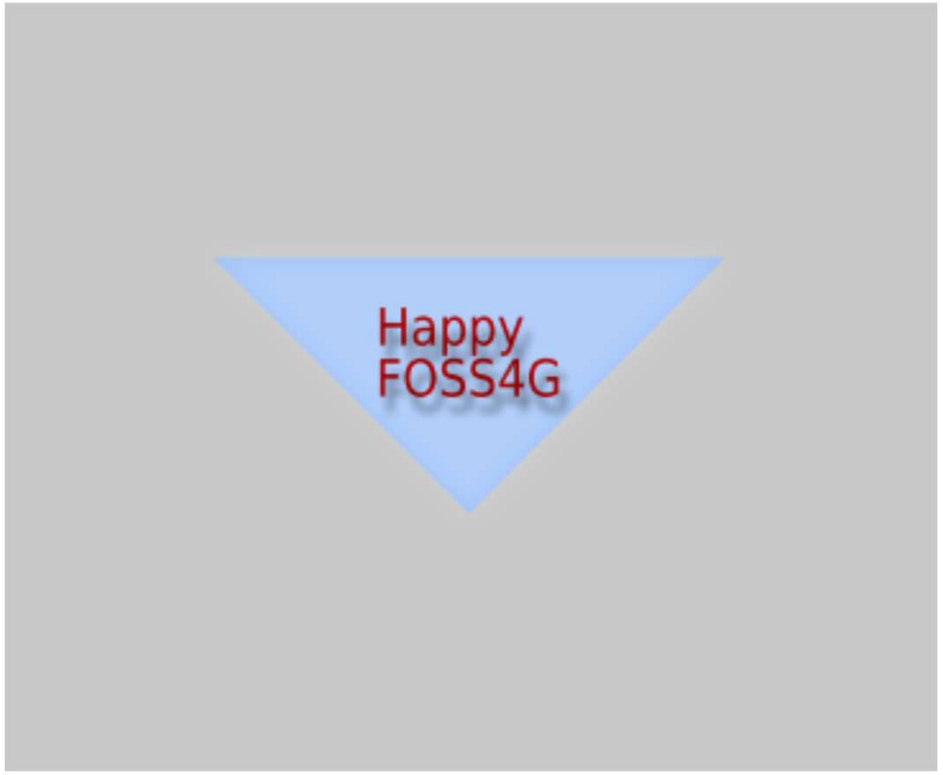


Well let's continue with more examples



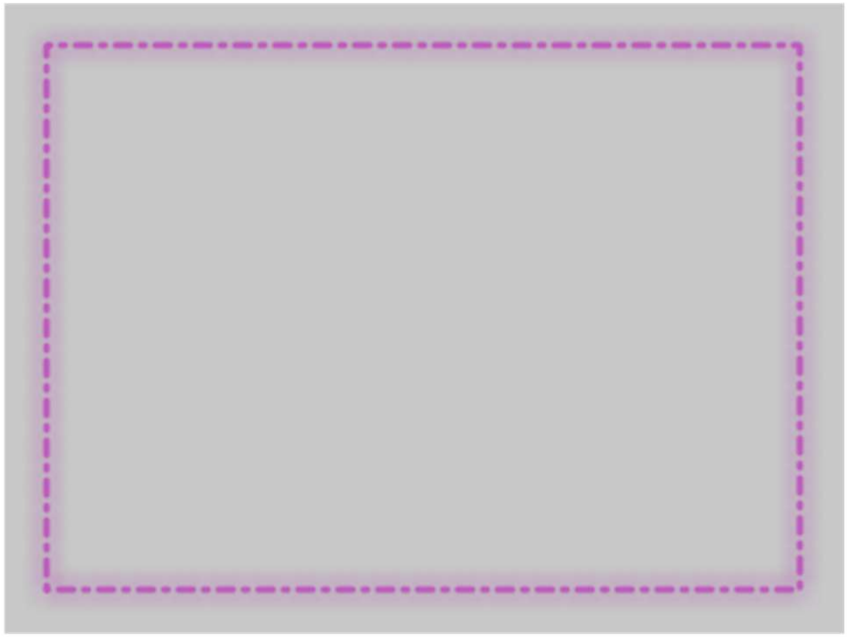
Text label shadow with labelcache off

```
LAYER
    NAME "text"
    TYPE POINT
    FEATURE
        POINTS
            10 8
        END
        TEXT "Happy FOSS4G"
    END
    STATUS ON
    LABELCACHE OFF
    COMPOSITE
        COMPFILTER "blur(3)"
        COMPFILTER "grayscale()"
        COMPFILTER "translate(3,3)"
        COMPOP "multiply"
        OPACITY 70
    END
    COMPOSITE
        OPACITY 100
        COMPOP "multiply"
        COMPFILTER "translate(0,-3)"
    END
    CLASS
        LABEL
            COLOR 255 0 0
            SIZE 15
            WRAP " "
        END
    END
END
```



Border line with very soft shadow

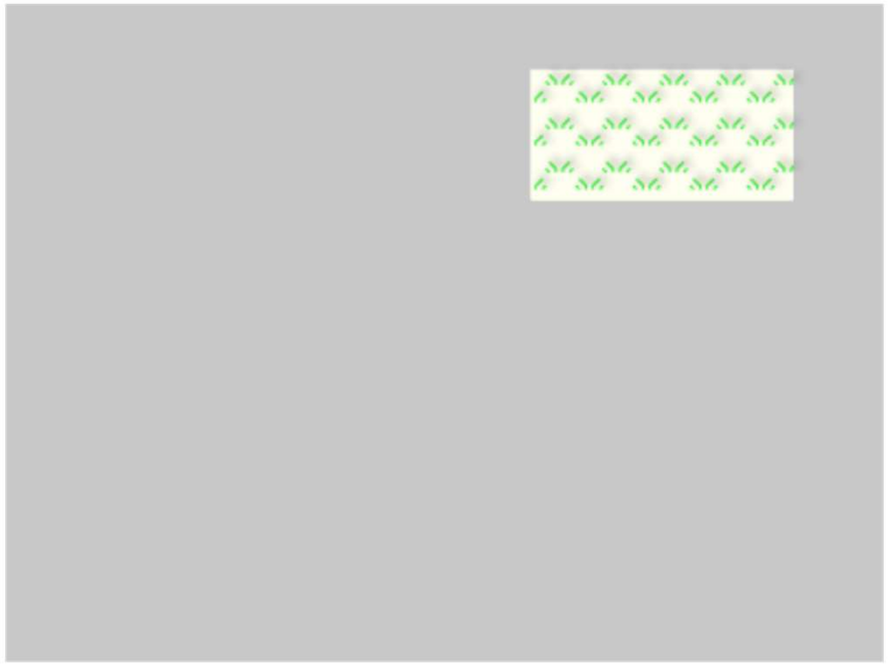
```
LAYER
    NAME "line-border"
    TYPE LINE
    FEATURE
        POINTS
            1 1
            1 14
            19 14
            19 1
            1 1
        END
    END
    STATUS ON
    COMPOSITE
        COMPFILTER "translate(4,4)"
        COMPFILTER "blur(5)"
        OPACITY 30
    END
    COMPOSITE
        COMPFILTER "translate(-4,4)"
        COMPFILTER "blur(5)"
        OPACITY 30
    END
    COMPOSITE
        COMPFILTER "translate(-4,-4)"
        COMPFILTER "blur(5)"
        OPACITY 30
    END
    COMPOSITE
        COMPFILTER "translate(4,-4)"
        COMPFILTER "blur(5)"
        OPACITY 30
    END
    COMPOSITE
        OPACITY 100
    END
    CLASS
        STYLE
            COLOR 190 90 190
            WIDTH 3
            PATTERN
                2 5
                7 5
            END
        END
    END
END
```



Shadows in polygon point patterns (vector symbol)

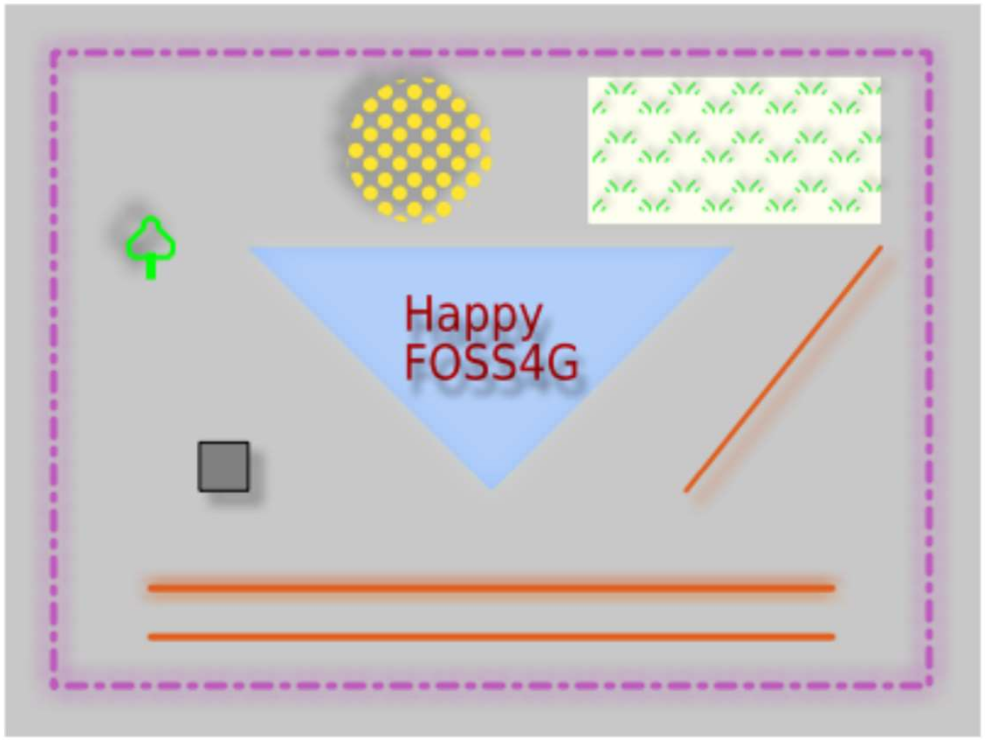
```
SYMBOL
    NAME "Grass"
    TYPE VECTOR
    POINTS
        0 2
        1 3
        -99 -99
        1 0
        3 2
        3 3
        -99 -99
        7 0
        5 2
        5 3
        -99 -99
        8 2
        7 3
        -99 -99
        8 10
        9 11
        -99 -99
        9 8
        11 10
        11 11
        -99 -99
        15 8
        13 10
        13 11
        -99 -99
        16 10
        15 11
        -99 -99
        17 13
    END
END
```

```
LAYER
    NAME "poly-fill-test"
    TYPE POLYGON
    STATUS ON
    FEATURE
        POINTS
            12 10.5
            12 13.5
            18 13.5
            18 10.5
            12 10.5
        END
    END
    COMPOSITE
        OPACITY 100
    END
    COMPOSITE
        COMPFILTER "grayscale()"
        COMPFILTER "blacken()"
        COMPFILTER "translate(2,-2)"
        COMPFILTER "blur(3)"
        OPACITY 50
    END
    CLASS
        STYLE
            COLOR 5 255 10
            WIDTH 0.8
            SYMBOL "Grass"
            SIZE 20
        END
    END
END
```



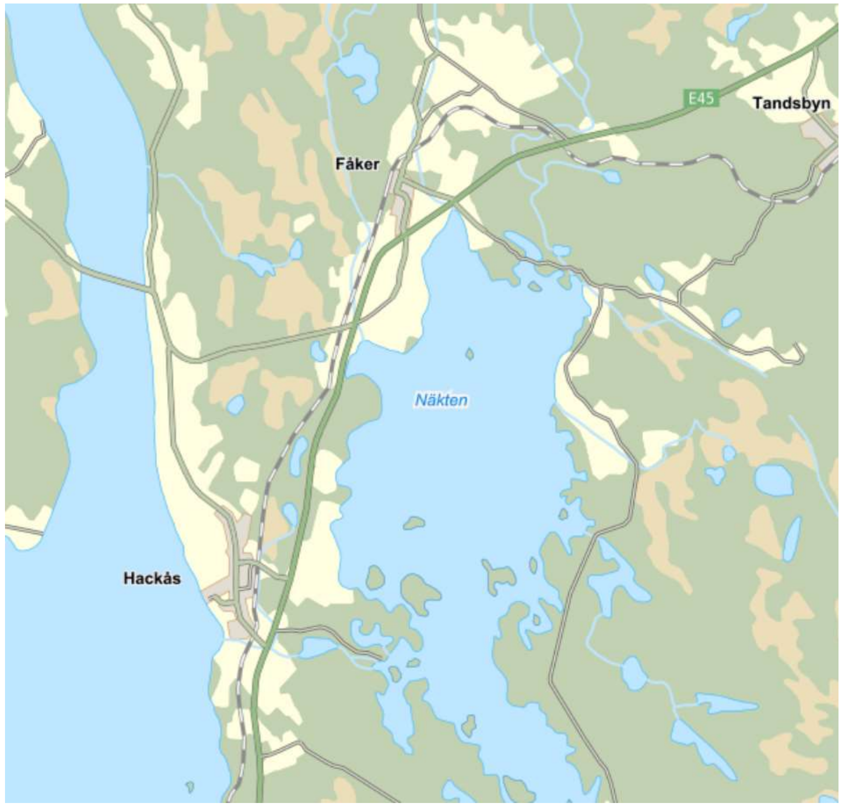
Chainable Compositing Filters - Summary

- Endless possibilities
- Let us as community explore



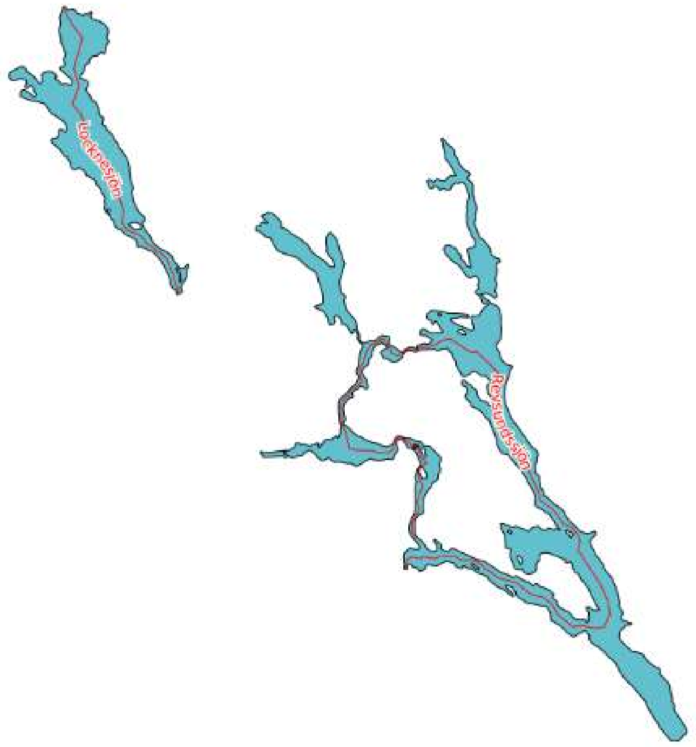
Make label text outline or shadow semitransparent in map

- You specify the color in hex: RGBA
- RGBA value (adding translucence): “#rrggbbaa”. To specify a semi-translucent magenta, the following is used:
- Example: COLOR "#FF00FFCC"
- Look at: https://mapserver.org/mapfile/style.html



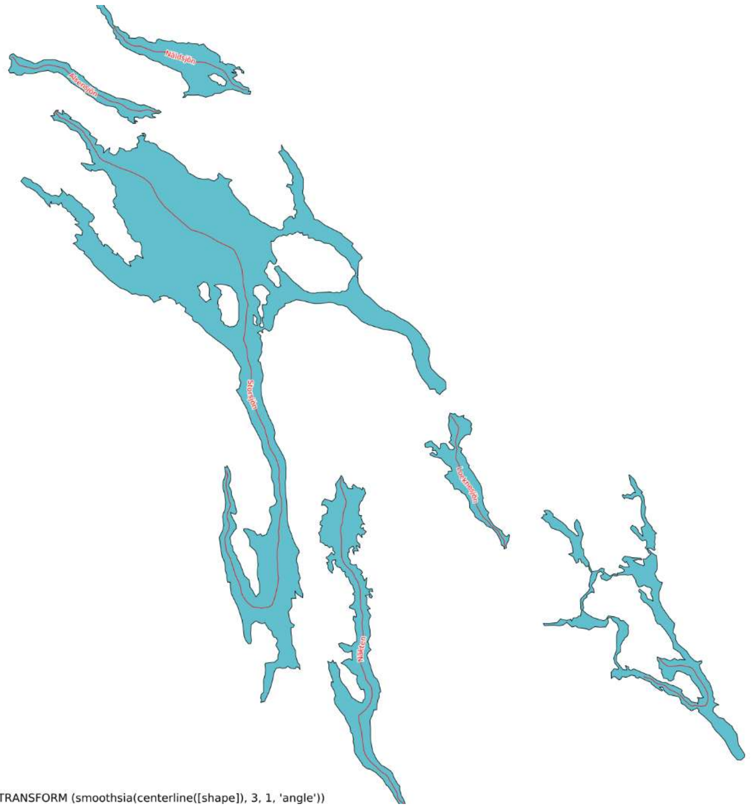
GEOMTRANSFORM - centerline

- *New in version 8.0:* centerline(), inner(), outer()
- Implemented by Steve Lime
- *(centerline([shape])*: Useful for labeling polygons, creates a centered line (*[shape]*) using a Voronoi diagram generated by GEOS and then additional simplification. Requires GEOS >= 3.5 Centerlines can *only* be computed for polygon shapes.



Tests with Swedish 1M data set

```
LAYER
  NAME "red-line"
  TYPE LINE
  STATUS ON
  CONNECTIONTYPE OGR
  CONNECTION "ZLakes_1M.db"
  DATA "lakes"
  LABELITEM "namn1"
  GEOMTRANSFORM (smoothsia(centerline([shape]), 3, 1, 'angle'))
  CLASS
    NAME "red"
    STYLE
      COLOR 255 0 0
    END #style
    LABEL
      COLOR 255 10 0
      OUTLINECOLOR 250 250 250
      OUTLINEWIDTH 2
      FONT sans
      TYPE truetype
      SIZE 10
      FORCE TRUE
      ANGLE FOLLOW
    END #label
  END #class
  PROJECTION
    "init=epsg:3006"
  END #projection
END #layer
```


GEOMTRANSFORM (smoothsia(centerline([shape]), 3, 1, 'angle'))

Tests with Swedish 250k data set

- Working good with:

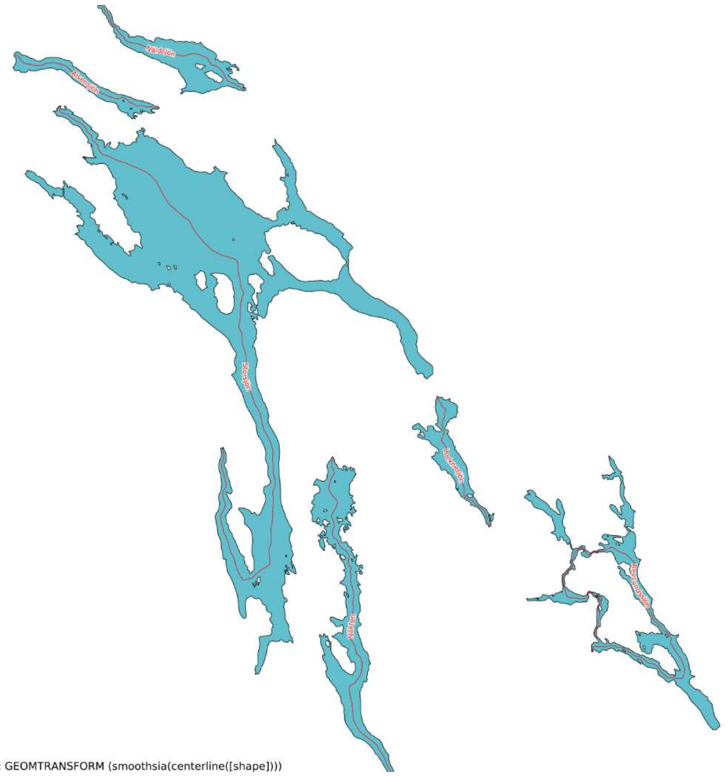
GEOMTRANSFORM(simplify(centerline([shape]),50))

GEOMTRANSFORM(smoothsia(centerline([shape])))

GEOMTRANSFORM(smoothsia(centerline([shape]), 3, 1, 'angle'))

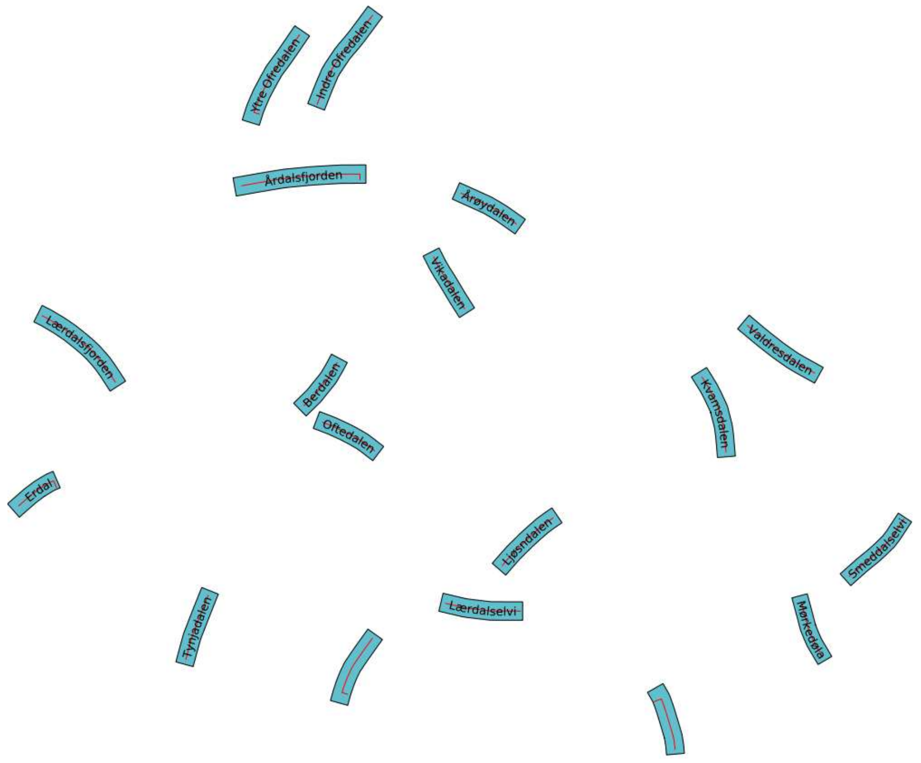
- Missed one lake with:

GEOMTRANSFORM(smoothsia(centerline([shape]), 3, 1, 'all'))

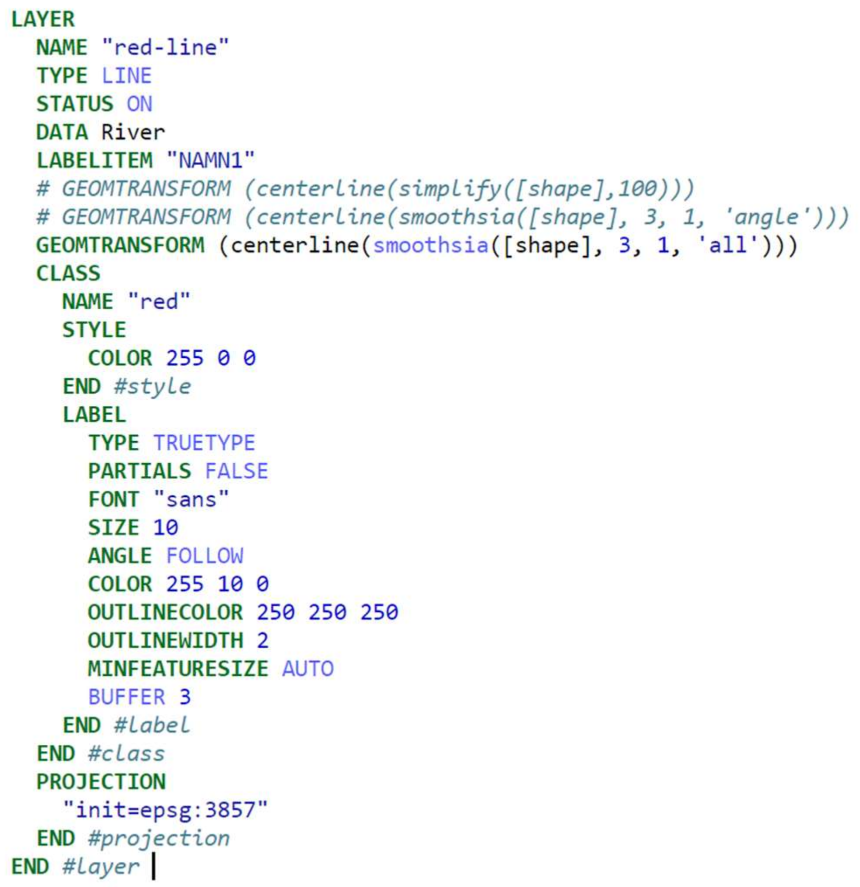

TEST: GEOMTRANSFORM (smoothsia(centerline([shape])))

Norwegian text labels in predefined boxes

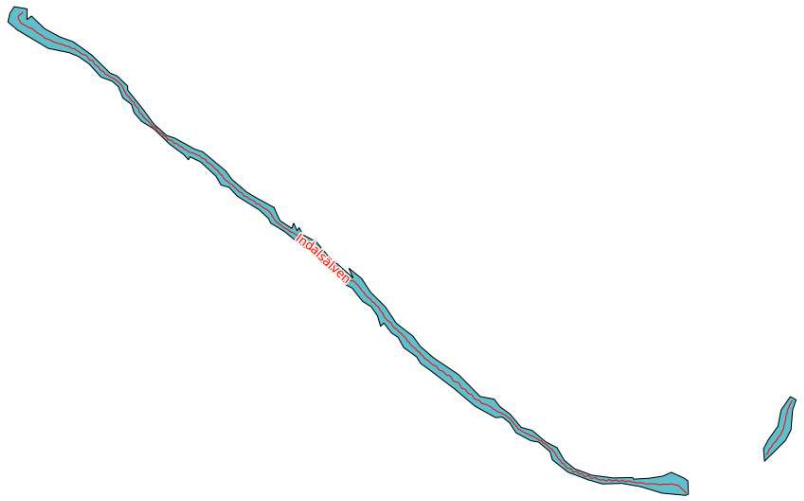
```
LAYER
  NAME "red-line"
  TYPE LINE
  STATUS ON
  CONNECTIONTYPE OGR
  CONNECTION "Norway_box_labels.db"
  DATA "labels"
  LABELITEM "text"
  GEOMTRANSFORM (centerline([shape]))
  CLASS
    NAME "red"
    STYLE
      COLOR 255 0 0
    END #style
    LABEL
      COLOR  0 0 0
      FONT sans
      TYPE truetype
      SIZE 10
      POSITION cc
      FORCE TRUE
      ANGLE FOLLOW
    END #label
  END #class
  PROJECTION
    "init=epsg:3006"
  END #projection
END #layer
```



River example – EGM data set

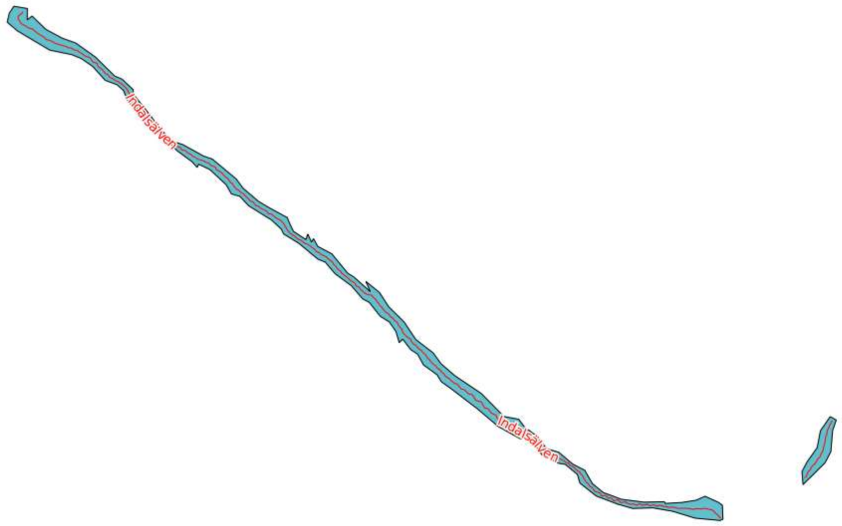


```
LAYER
  NAME "red-line"
  TYPE LINE
  STATUS ON
  DATA River
  LABELITEM "NAMN1"
  # GEOMTRANSFORM (centerline(simplify([shape],100)))
  # GEOMTRANSFORM (centerline(smoothsia([shape], 3, 1, 'angle')))
  GEOMTRANSFORM (centerline(smoothsia([shape], 3, 1, 'all')))
  CLASS
    NAME "red"
    STYLE
      COLOR 255 0 0
    END #style
    LABEL
      TYPE TRUETYPE
      PARTIALS FALSE
      FONT "sans"
      SIZE 10
      ANGLE FOLLOW
      COLOR 255 10 0
      OUTLINECOLOR 250 250 250
      OUTLINEWIDTH 2
      MINFEATURESIZE AUTO
      BUFFER 3
    END #label
  END #class
  PROJECTION
    "init=epsg:3857"
  END #projection
END #layer
```



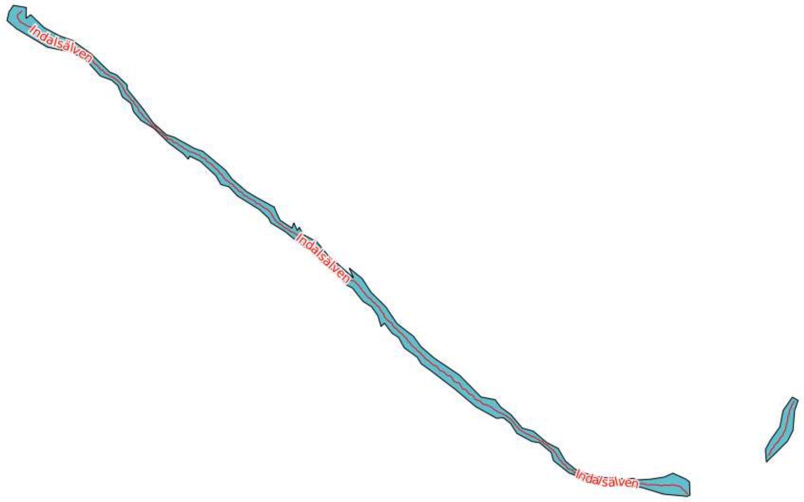
River example 2 – EGM data set add REPEATDISTANCE 400

```
LAYER
  NAME "red-line"
  TYPE LINE
  STATUS ON
  DATA River
  LABELITEM "NAMN1"
  GEOMTRANSFORM (centerline(smoothsia([shape], 3, 1, 'all')))
  CLASS
    NAME "red"
    STYLE
      COLOR 255 0 0
    END #style
    LABEL
      TYPE TRUETYPE
      PARTIALS FALSE
      FONT "sans"
      SIZE 10
      MINDISTANCE 200
      REPEATDISTANCE 400
      ANGLE FOLLOW
      COLOR 255 10 0
      OUTLINECOLOR 250 250 250
      OUTLINEWIDTH 2
      MINFEATURESIZE AUTO
      BUFFER 3
    END #label
  END #class
  PROJECTION
    "init=epsg:3857"
  END #projection
END #layer
```



River example 3 – EGM data set with smaller REPEATDISTANCE 300

```
LAYER
  NAME "red-line"
  TYPE LINE
  STATUS ON
  DATA River
  LABELITEM "NAMN1"
  GEOMTRANSFORM (centerline(smoothsia([shape], 3, 1, 'all')))
  CLASS
    NAME "red"
    STYLE
      COLOR 255 0 0
    END #style
    LABEL
      TYPE TRUETYPE
      PARTIALS FALSE
      FONT "sans"
      SIZE 10
      REPEATDISTANCE 300
      ANGLE FOLLOW
      COLOR 255 10 0
      OUTLINECOLOR 250 250 250
      OUTLINEWIDTH 2
      MINFEATURESIZE AUTO
      BUFFER 3
    END #label
  END #class
  PROJECTION
    "init=epsg:3857"
  END #projection
END #layer
```



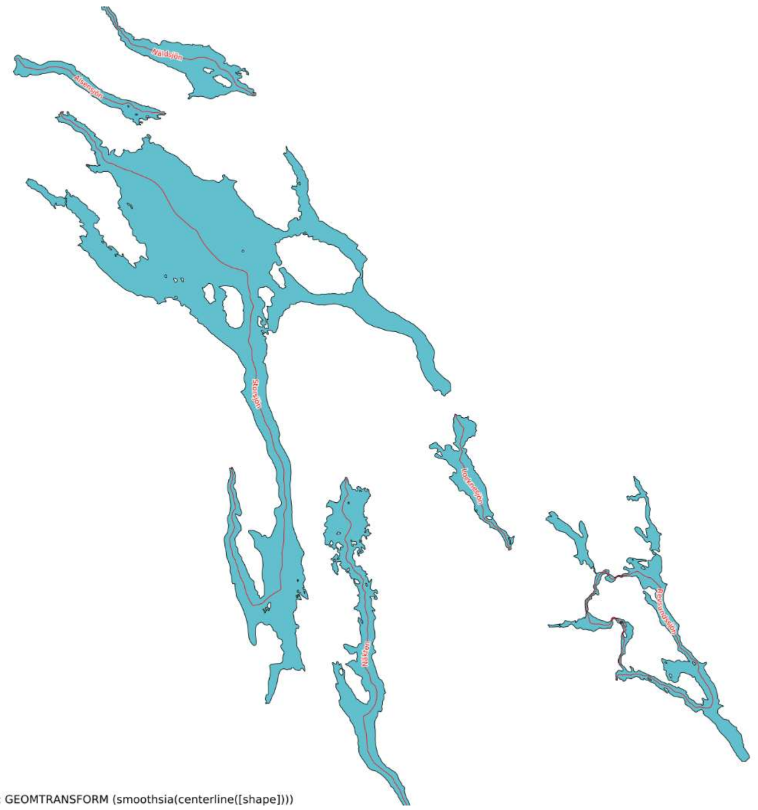
GEOMTRANSFORM – centerline - findings

- More experimentation needed to:
  - Find good parameters for different data sets
  - Figure out when preprocessing polygon is needed

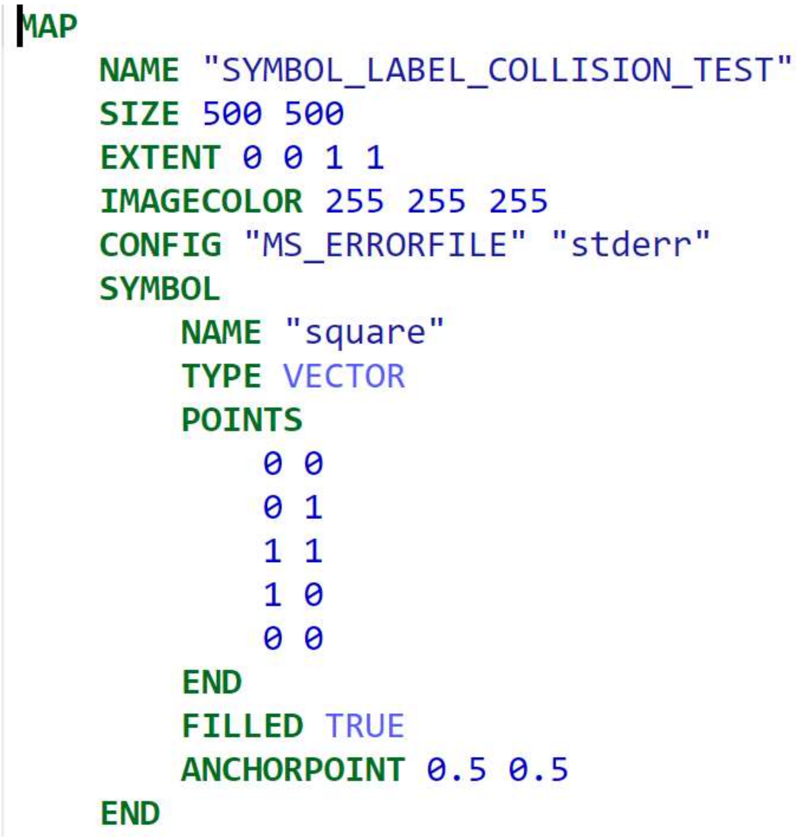
    (inside centerline function)
  - Figure out when postprocessing the line is needed

    (outside centerline function)

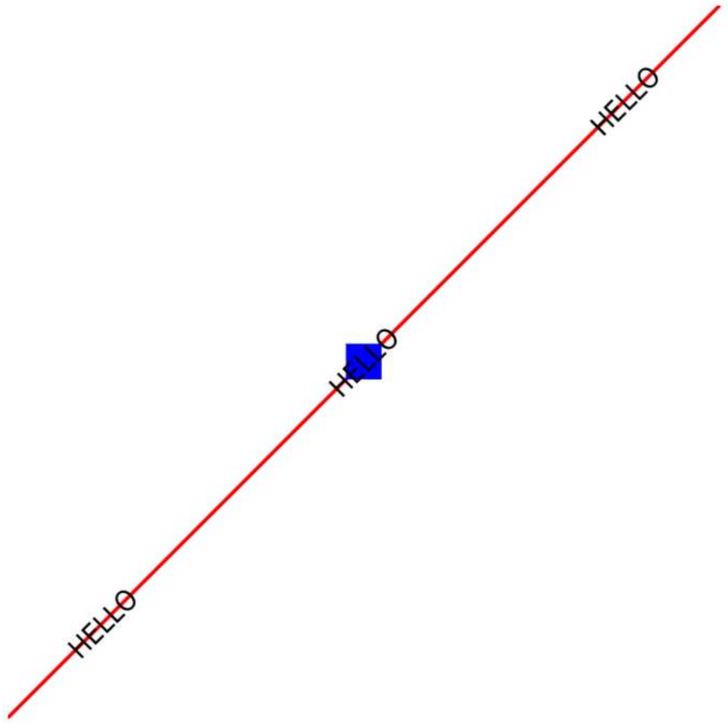
  Improve inner function with size argument


TEST: GEOMTRANSFORM (smoothsia(centerline([shape])))

Line text and symbol collision

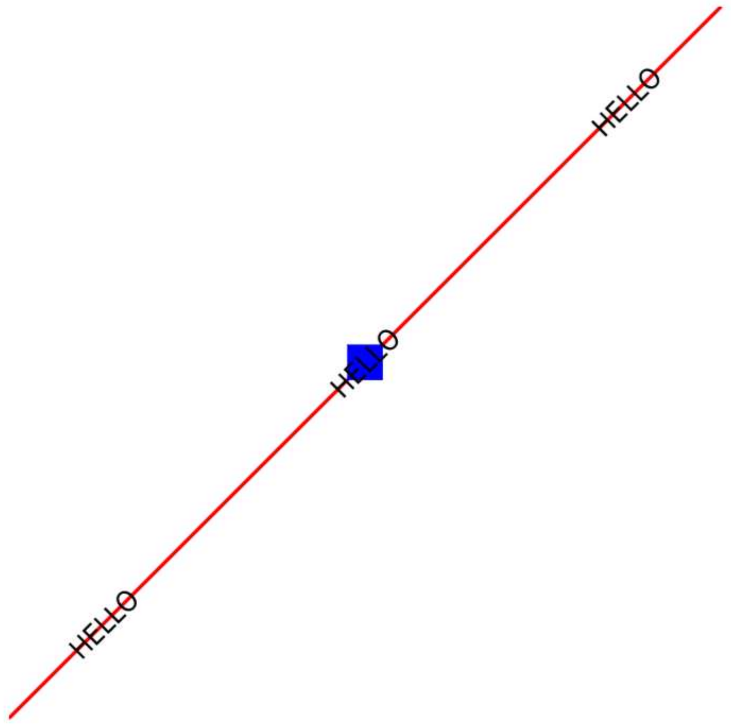


```
MAP
    NAME "SYMBOL_LABEL_COLLISION_TEST"
    SIZE 500 500
    EXTENT 0 0 1 1
    IMAGECOLOR 255 255 255
    CONFIG "MS_ERRORFILE" "stderr"
    SYMBOL
        NAME "square"
        TYPE VECTOR
        POINTS
            0 0
            0 1
            1 1
            1 0
            0 0
        END
        FILLED TRUE
        ANCHORPOINT 0.5 0.5
    END
```

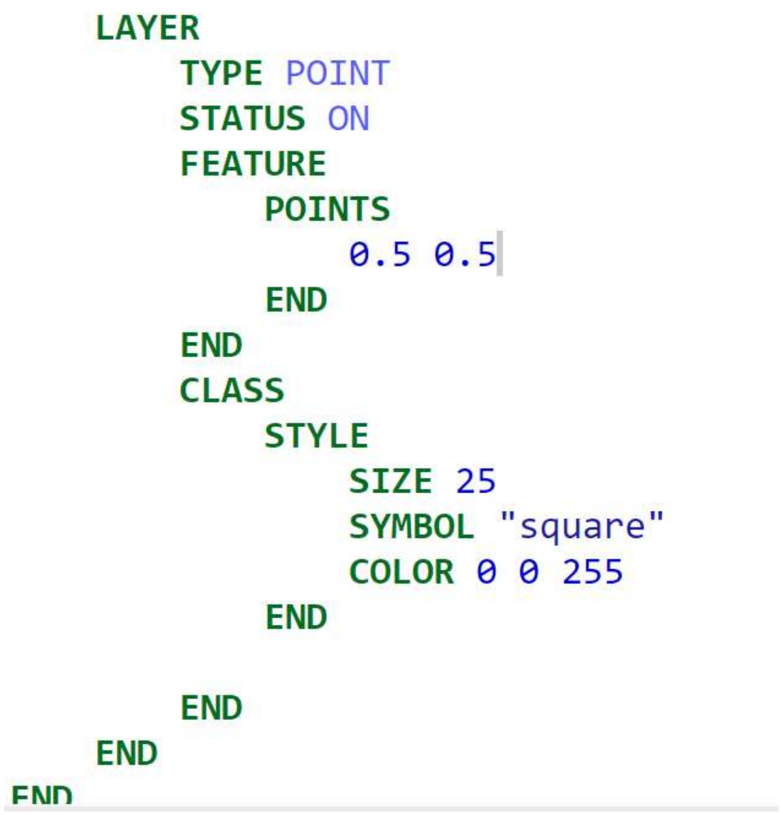


Line text and symbol collision (cont.)

```
LAYER
    TYPE LINE
    STATUS ON
    FEATURE
        POINTS
            0 0
            0.5 0.5
            1 1
        END
        TEXT "HELLO"
    END
    CLASS
        STYLE
            COLOR 255 0 0
            WIDTH 2.5
        END
        LABEL
            COLOR 0 0 0
            SIZE 15
            ANGLE FOLLOW
            POSITION AUTO
            REPEATDISTANCE 200
        END
    END
END
```

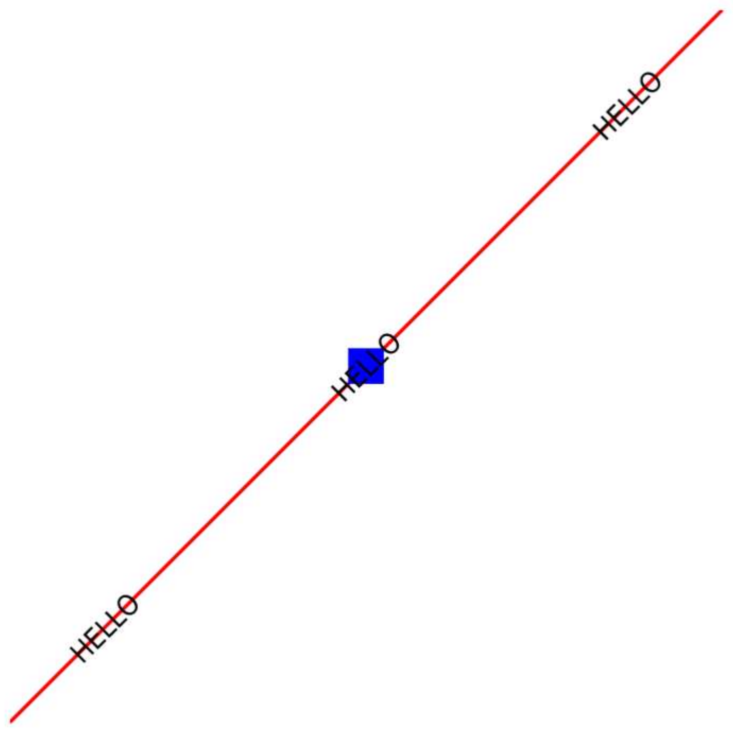


Line text and symbol collision (cont.)



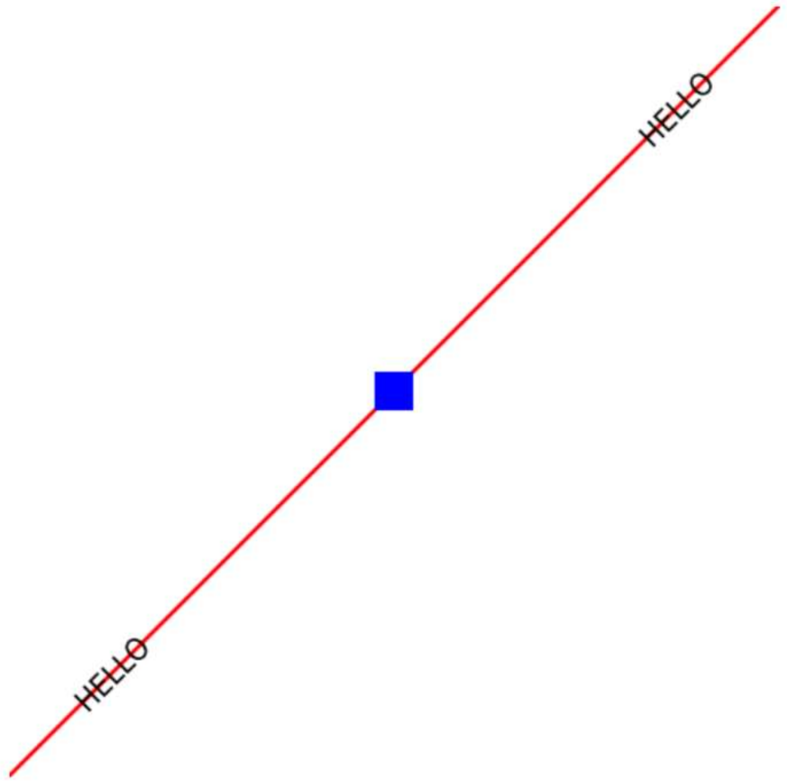
```
    LAYER
        TYPE POINT
        STATUS ON
        FEATURE
            POINTS
                0.5 0.5
            END
        END
        CLASS
            STYLE
                SIZE 25
                SYMBOL "square"
                COLOR 0 0 255
            END

        END
    END
END
```



Line text and symbol collision avoidance
Protect the symbol with a transparent label

```
    LAYER
        TYPE POINT
        STATUS ON
        FEATURE
            POINTS
                0.5 0.5
            END
        END
        CLASS
            STYLE
                SIZE 25
                SYMBOL "square"
                COLOR 0 0 255
            END
            LABEL
                TEXT "X"
                COLOR "#00000000"
                SIZE 8
                POSITION CC
            END
        END
    END
END
```

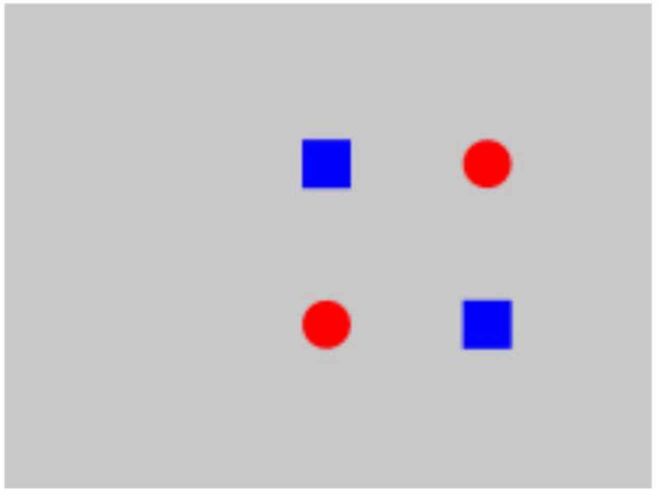


Named styles in Mapserver WMS

- Use LAYER CLASSGROUP and CLASS GROUP for style
- LAYER GROUP is for layer specification
- LAYER can also be an root layer when specified as MAP NAME
- An example can be found at:

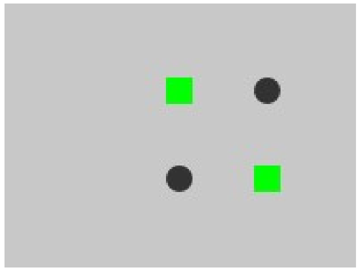
https://gist.github.com/LarsSchy/

It is called: style_test.map

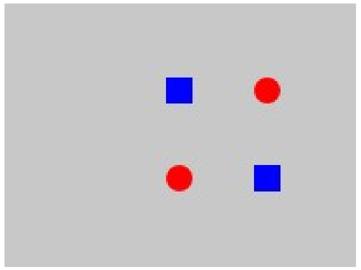


Named styles in Mapserver WMS

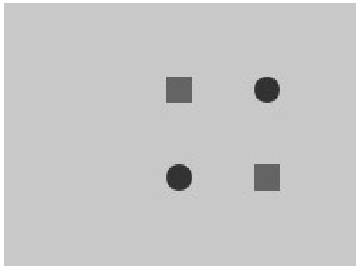
&LAYERS=square,circle&styles=green,grey



&LAYERS=ST&styles=color

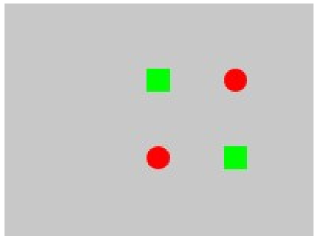


&LAYERS=ST&styles=grey



Some more examples

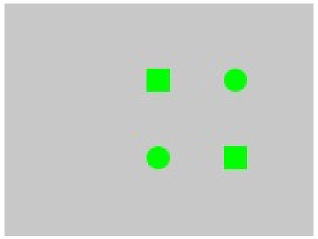
&LAYERS=square,circle&styles=green,color



&LAYERS=square,circle&styles=green,green

OR

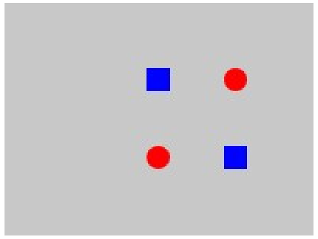
&LAYERS=ST&styles=green



&LAYERS=square,circle&styles=color,color

OR

&LAYERS=ST&styles=default



Some examples are available on GITHUB

https://github.com/LarsSchy

https://gist.github.com/LarsSchy

Thanks for your attention!