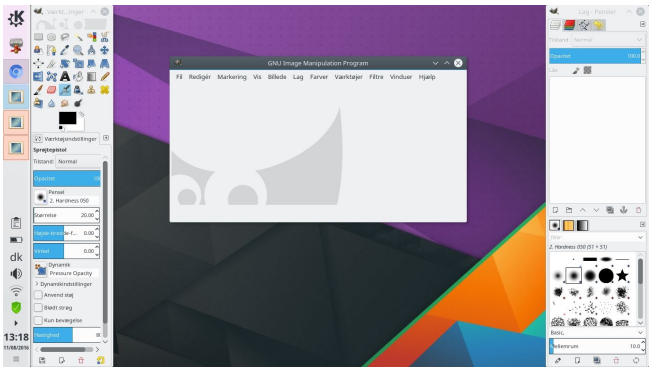
Digitale billeder 2016KomItC

Øvelse 1 - billedbehandling - Gimp

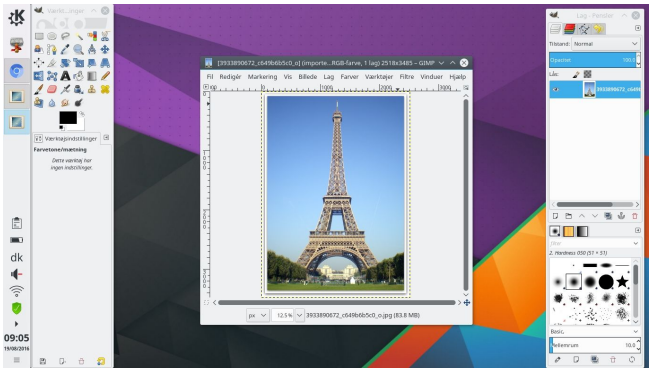
I denne øvelse har jeg lært nogle basiske funktioner i billedbehandlingsprogrammet **Gimp**. Vi skal have en gul luftballon til svæve ved siden af Le Tour Eiffel i Paris, Franking.

1. Le Tour Eiffel

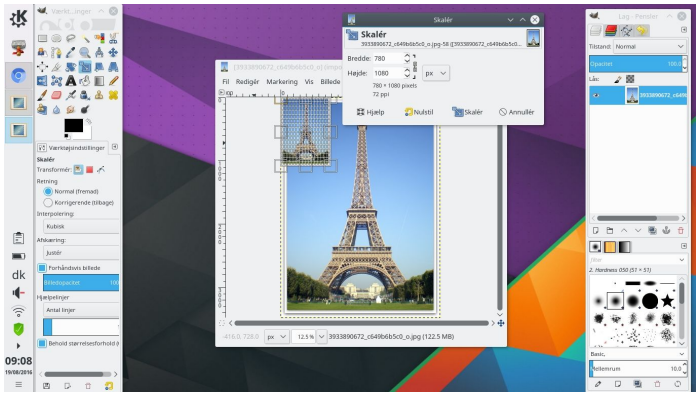
Vi starter med at starte Gimp op.

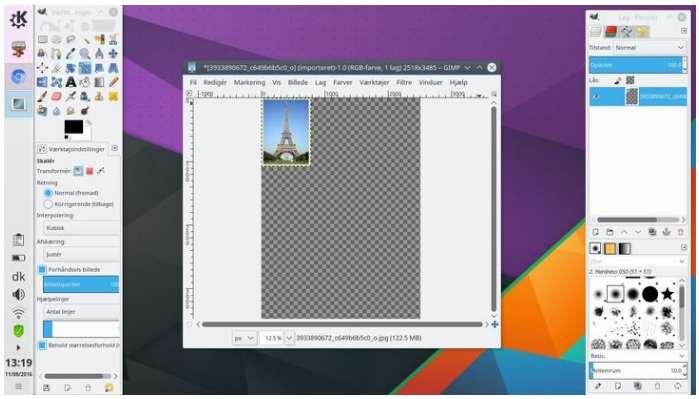


Først vil vi importere vores billede af Eiffel tårnet. *Fil -> Åbn… -> Vælg billede -> Åbn.*

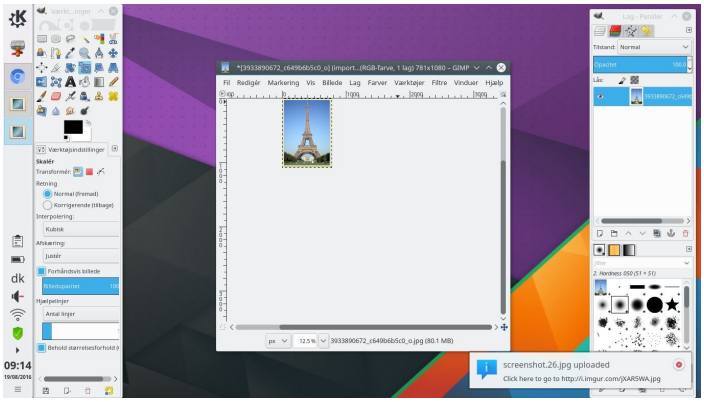


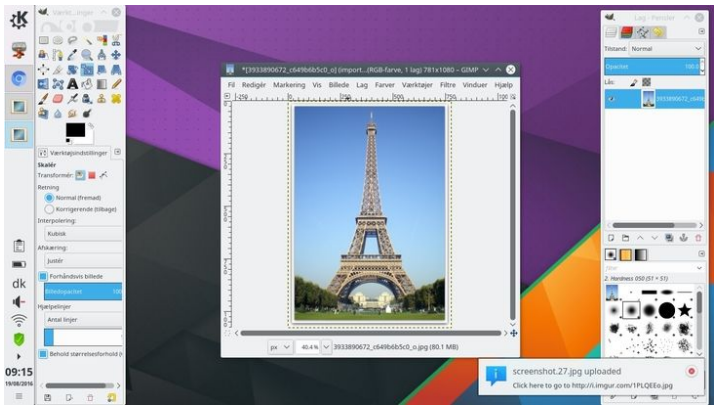
Siden vores billede er lidt stort (2518 x 3485), vil vi nedskalére det lidt. *Skalér -> (Lås ->) Vælg Højde/Bredde (780 x 1080) -> Skalér*





Vi vil nu tilpasse billedelaget til vinduet. *Billede -> Tilpas læred til lag. Shift + CTRL + J eller Vis -> Zoom (x%) -> Tilpas billede i vindue.*

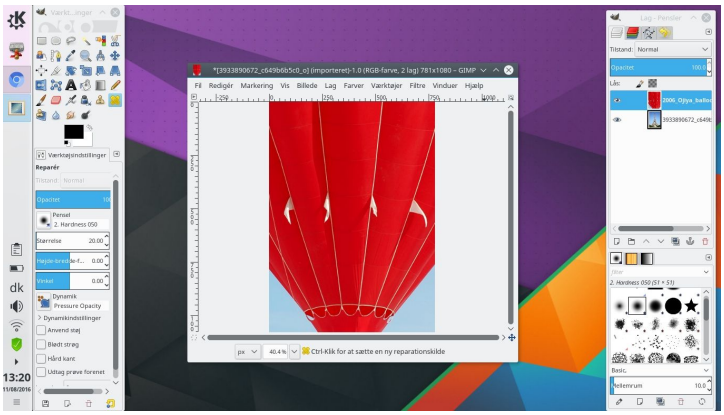




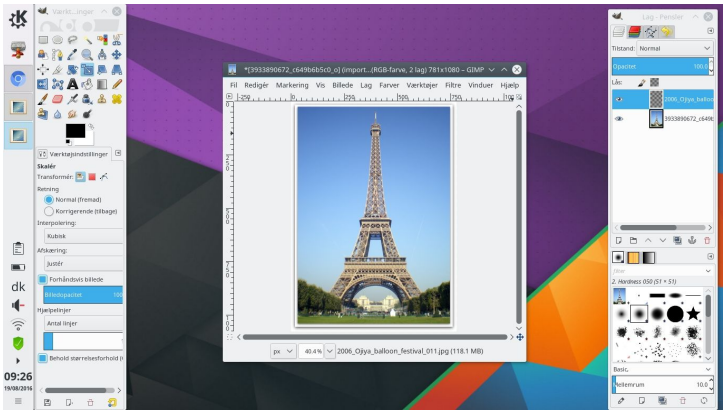
2. Luftballon

Import, nedskalér og flytte

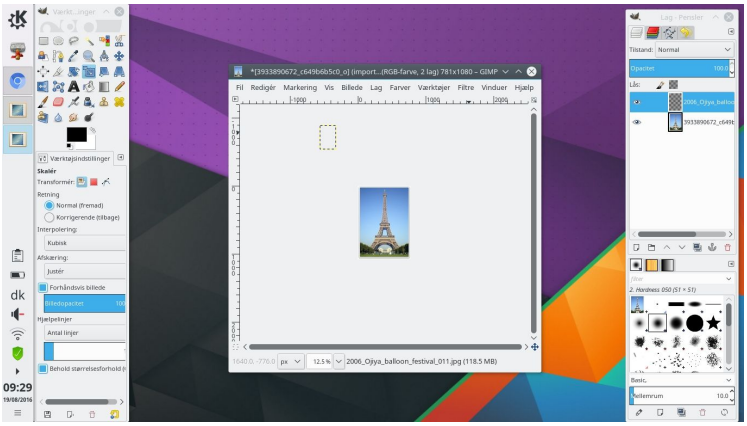
Nu er det tid til at tilføje vores luftballon. *Fil -> Åbn som lag…*

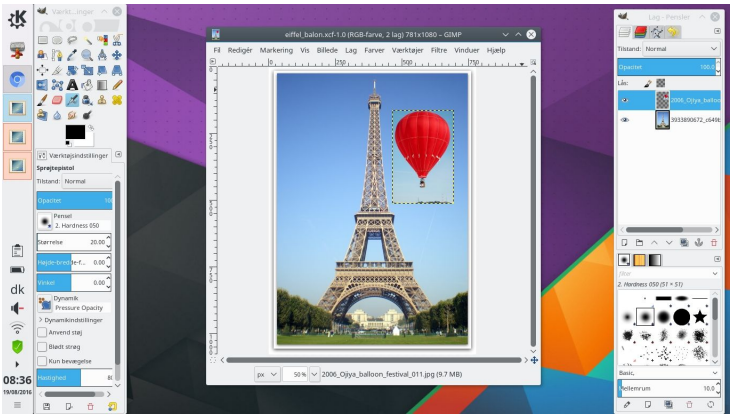


Præcis som vi nedskalerede vores Eiffeltårn, vil vi nedskalere vores luftballon. *Skalér -> (Lås ->) Vælg Højde/Bredde (233 x 350) -> Skalér.*



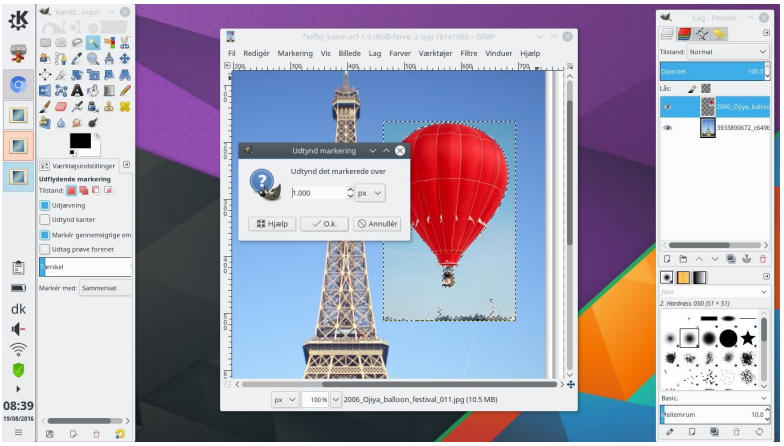
Det kan se ud som at billedet forsvandt, men hvis du zoomer ud (i bunden af vinduet) til fx. 12.5%, så kan du se den bare ligger oppe i hjørnet. Du kan flytte den med flytteværktøjet.



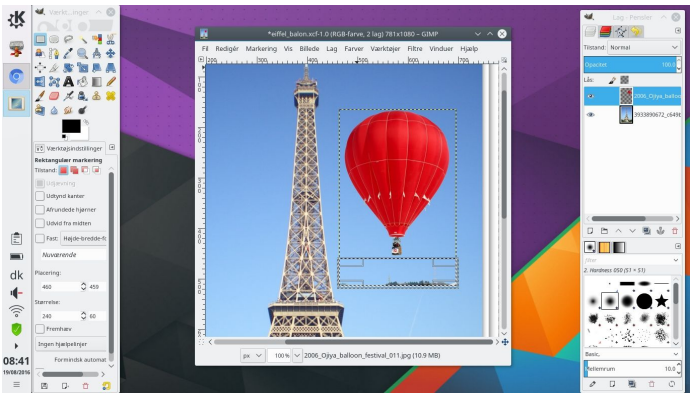


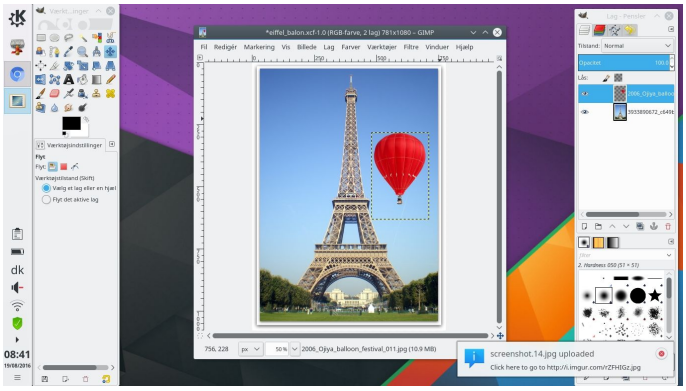
Baggrund- og farvebehandling

Vi starter med at fjerne baggrunden. Dette gør vi ved at trykke på tryllestaven -> vælge vores område og tryk “delete”. Hvis det skal se mindre “kantet” ud, kan du udtynde markeringen med 1 pixel eller 2.
Makering -> Udtynd…

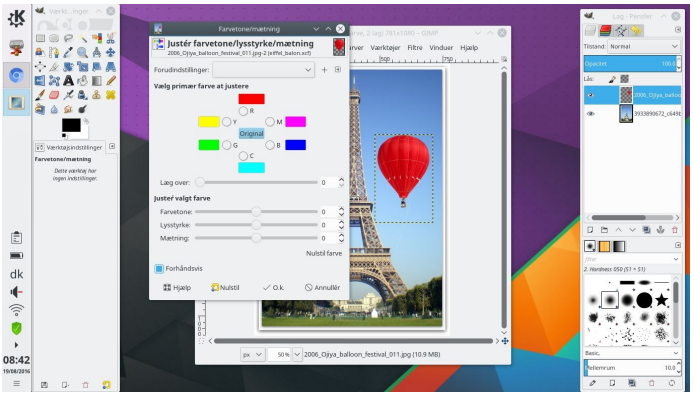


Hvis der stadig er lidt i bunden kan du evt. bruge den rektangulære markering -> markér og tryk delete.

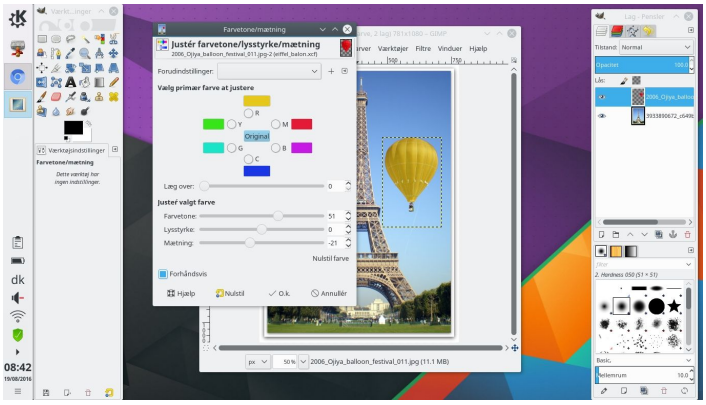




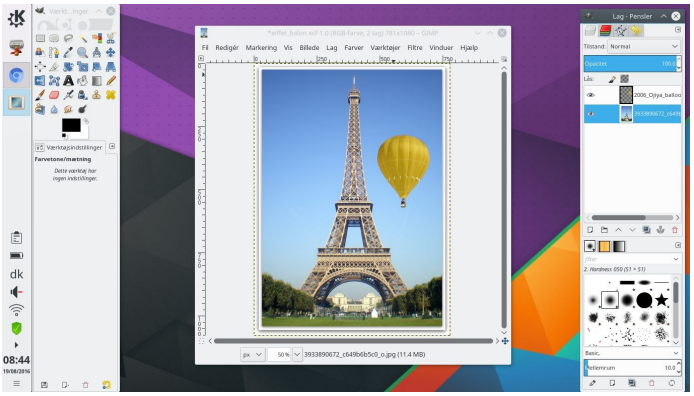
Nu er det tid til at ændre farven på ballonen. *Farver -> Farvetone/mætning…*



Her kan du justere farverne, så de passer til det du vil have. I vores tilfælde vil vi have en gul farve.



Tryk derefter O.k. og du er færdig.

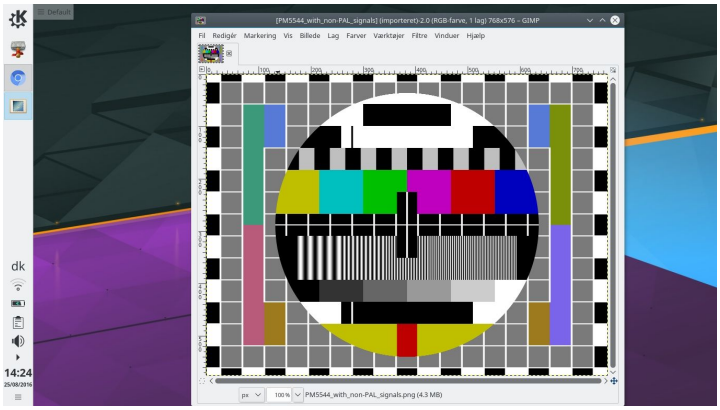


Øvelse 2 - Bits og bytes

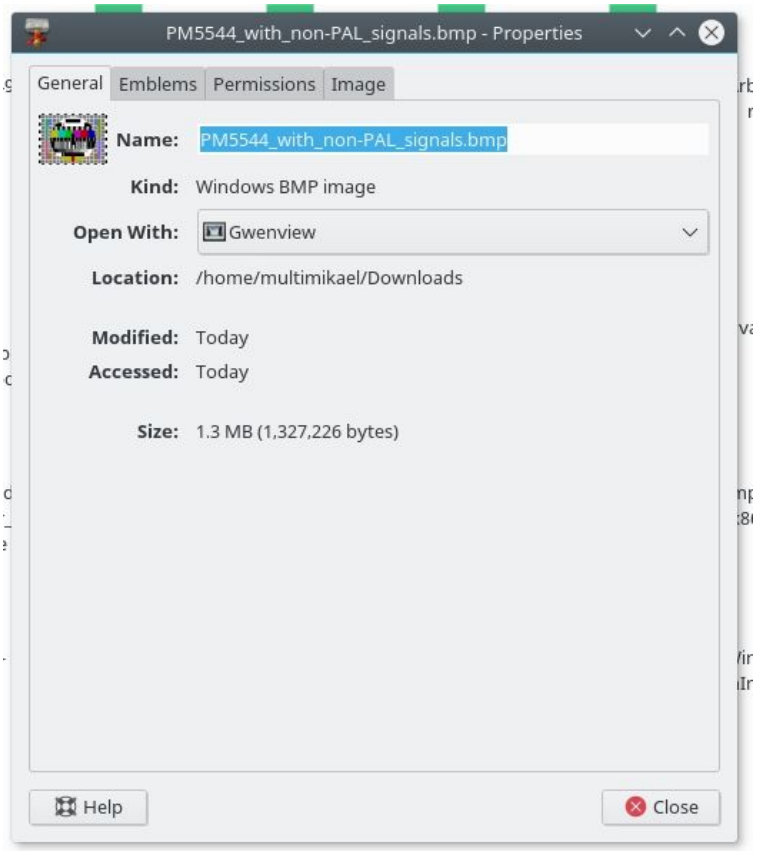
I computerens verden bruger vi 2 tals systemet “binær”. 1 Bit er et “tal” (0 eller 1). En bytes består af 8 bits, hvilket giver mulig for at have 2^8 eller 256 muligheder. Når vi gemmer et billede gemmer vi dens farvekoder i bytes. På 1 pixel har man typisk 3 kanaler med 1 byte hver. Hvis du altså har 8 pixels, ville det fylde 8*3 for hver pixel og billede ville fylde cirka 8*3*8 = 192 bytes.

Billedeksempel

Vi starter med importere billedet i Gimp. *Åbn... -> Vælg fil*



Når vi har importet fillen står lidt info i Gimp allerede til os. Vi kan se at billedet er 768 x 576 pixels. Vi kan nu begynde at udregne billedets størrelse i ukomprimeret størrelse. Hvis vi har 8*3 = 24 bits = 3 bytes per pixel, og vi har 768*576 pixel, kan vi regne det ud ved at side 768*576*3 = 1327104 bytes = 1327,104 kilobytes = 1,327104 megabytes.



Hvis vi så eksporterer billedet som en .bmp fil (ukomprimeret), og bruger en File Manager til at udregne størrelsen, kan vi se det bliver 1,327,226 bytes, hvor vi fik 1,327,104 bytes. Vi kan se at vi er 122 bytes fra det rigtige filstørrelse. Disse sidste bytes er metadata, det er data der bruges til at “fortælle” extra info. Det er data om data.

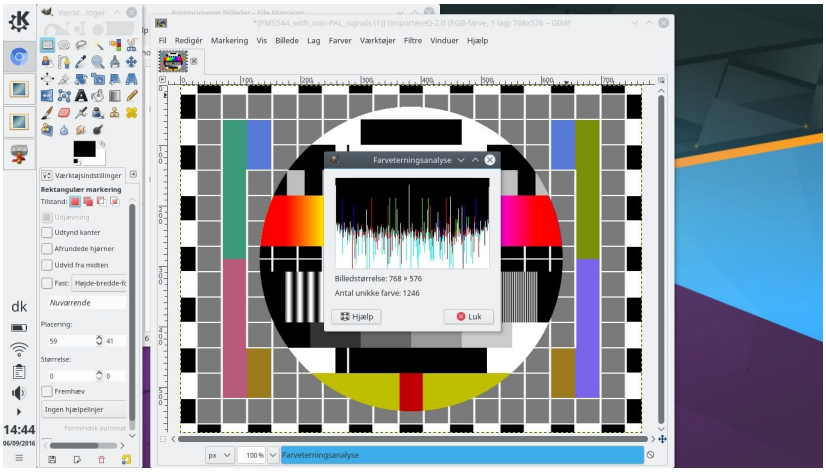
Videre udregning - Overførelseshastighed

Hvis nu vi vil regne lidt videre kan vi regne ud hvor lang tid det tager at overføre billedet med en downloadhastighed på 1 kilobit/s.

1. Vi starter med at omskrive vores bytes til kilobytes. 1.000 bytes = 1 kilobyte.
   1327104 bytes/1000 = 1327,104 kilobytes
2. Vi vil nu omskrive vores kilobytes til kilobits. 1 kilobyte = 8 kilobits.
   *1327,104 kilobytes * 8 = 10616,832 kilobits.*
3. Vi havde en downloadhastighed på 1 kilobit/s, så det tager:
   10616,832 kilobits / 1= 10616,832 sekunder at overføre billedet på 10616,832 kilobits.

Øvelse 3 - Komprimering

.jpg er et billedformat som kan bruges til at komprimere dine billeder. Hvis vi åbner vores billede i Gimp og går op i *farver -> info -> farveterningsanalyse.* Kan vi se hvor mange forskellige farver vores billede har.



Vores har 1246 unikke farver.

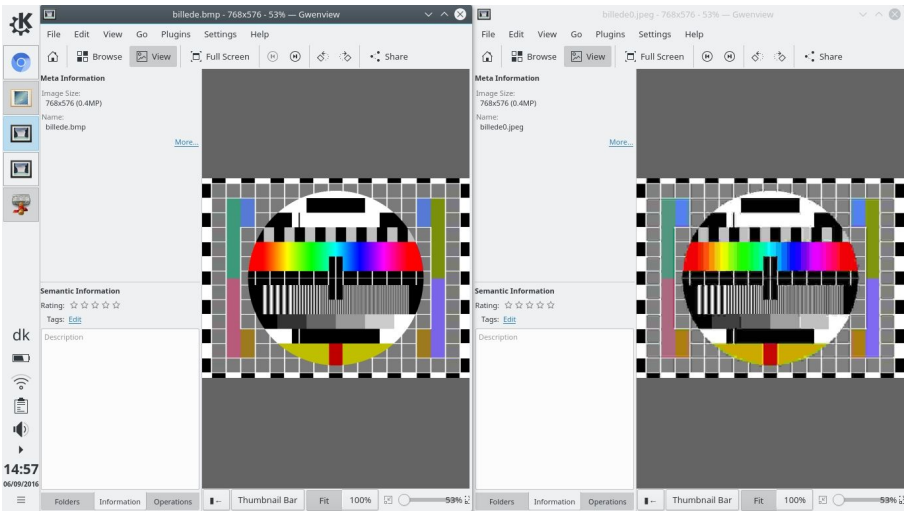
Eksportering af prøvebillede

Vi vil prøve at eksportere vores billede i flere forskellige formater; Ukomprimeret .bmp, .jpeg med 100 kvalitet, 50 kvalitet, 30 kvalitet, 0 kvalitet og .png med 9 compression level. *Fil -> Ekspotér som…*

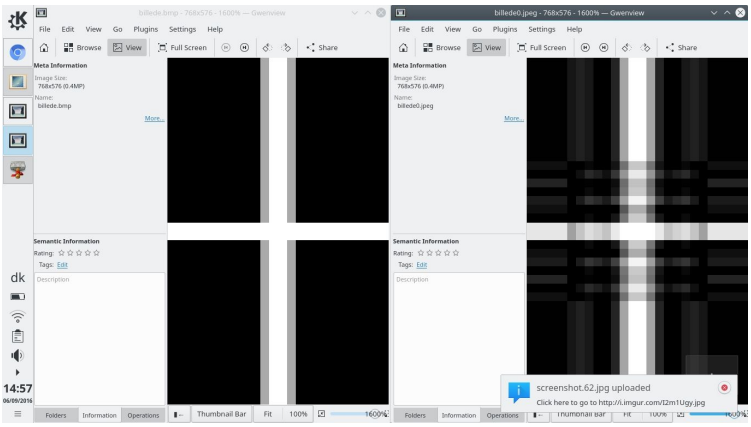
```
-rw-r--r-- 1 multimikael users   14388 Sep  6 14:48 billede0.jpeg
-rw-r--r-- 1 multimikael users  112094 Sep  6 14:47 billede100.jpeg
-rw-r--r-- 1 multimikael users   23105 Sep  6 14:48 billede10.jpeg
-rw-r--r-- 1 multimikael users   42054 Sep  6 14:47 billede50.jpeg
-rw-r--r-- 1 multimikael users 1327226 Sep  6 14:51 billede.bmp
-rw-r--r-- 1 multimikael users   25771 Sep  6 14:49 billede.png
```

Vi kan se at vores .jpeg med en kvalitets instilling på 0, fylder mindst. Derefter kommer vores .png med compression level på 9. Vores ukomprimeret fil fylder altså meget mere end de andre komprimeret. Men hvad er egentlig forskellen?

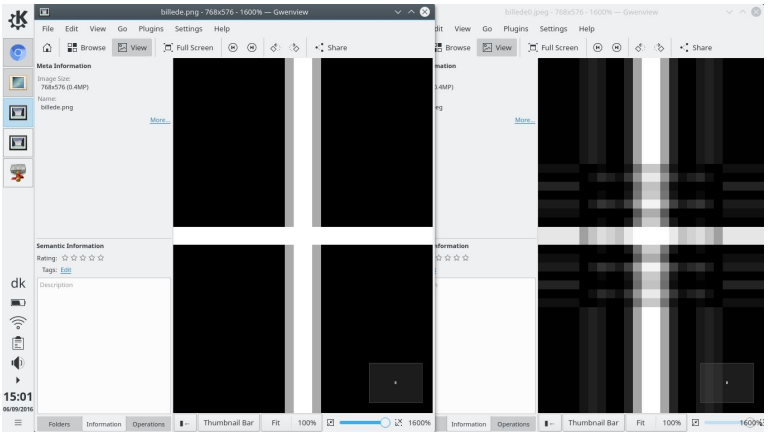
Billedkvalitet - .bmp eller .jpeg?



Hvis vi prøver at åbne vores .bmp (ukomprimeret, til venstre) og .jpeg (0 kvalitet, til højre), kan vi meget nem se forskel. Situationen ser ikke anderledes ud hvis vi zoomer ind.



Billedkvalitet - .jpeg eller .png?



Hvis vi prøver at åbne vores .png (Compression level 9, venstre) og en af vores .jpeg (kvalitet 0, højre). Det ser ud til at .png faktisk ser meget bedre ud tæt på, men hvis vi ser billederne langt væk fra kan vi ikke rigtig se nogen forskel.

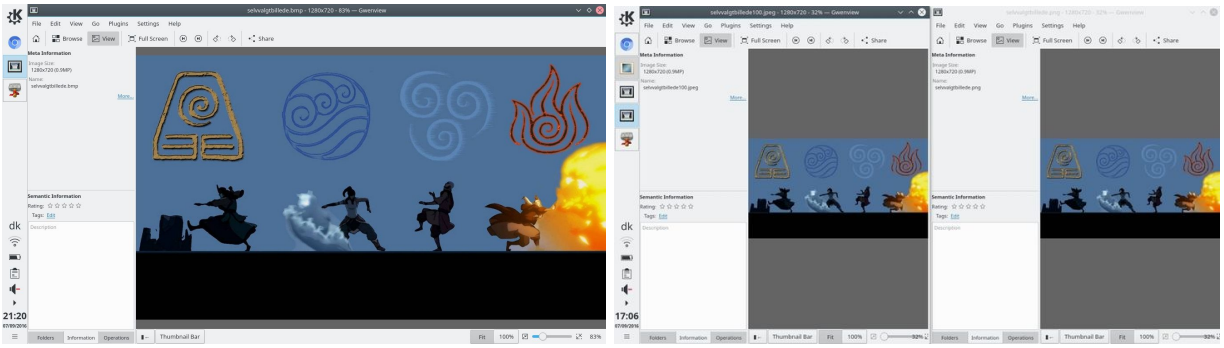
Eksportering af selvvalgt billede

Vi vil nu prøve at gøre det samme med et selvvalgt billede, som vi gjorde med prøvebilledet.

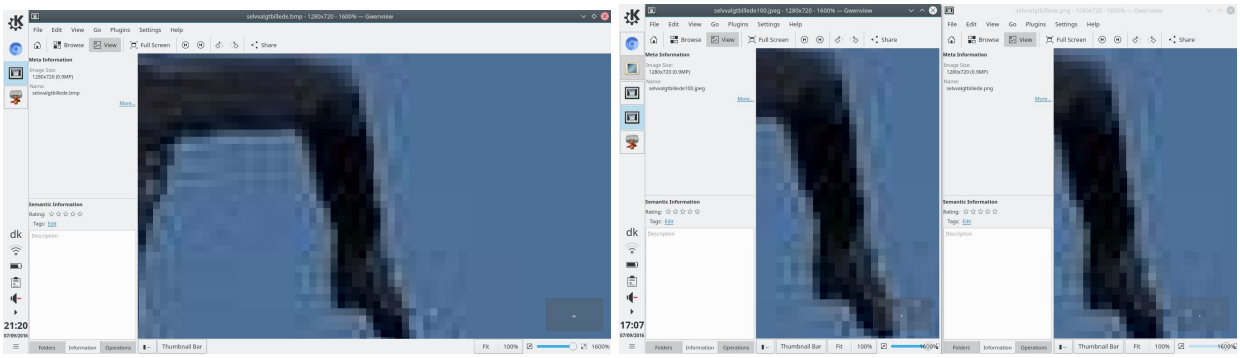
```
-rw-r--r-- 1 multimikael users    16142 Sep  7 16:56 selvvalgtbillede0.jpeg
-rw-r--r-- 1 multimikael users   261839 Sep  7 16:55 selvvalgtbillede100.jpeg
-rw-r--r-- 1 multimikael users    22697 Sep  7 16:56 selvvalgtbillede10.jpeg
-rw-r--r-- 1 multimikael users    60039 Sep  7 16:55 selvvalgtbillede50.jpeg
-rw-r--r-- 1 multimikael users  2764922 Sep  7 16:55 selvvalgtbillede.bmp
-rw-r--r-- 1 multimikael users   521837 Sep  7 16:55 selvvalgtbillede.png
```

Som vi kan se er vores komprimeret billede meget stort, men det er sjove er at vores .jpeg med 0 kvalitet faktisk blev næsten lige så stor som sidst. Billedet blev altså betydeligt mindre ved brug .jpeg komprimering. I dette tilfælde fylder vores .png faktisk mere end vores .jpeg med 100 kvalitet.

Billedkvalitet



Vores billeder ser faktisk meget ens ud, om det så er .bmp, .jpeg eller .png.



Hvis vi zoomer ind kan vi faktisk heller ikke se nogen forskel.

Filtyper

.jpeg kan i mange tilfælde blive komprimeret til en meget lavere filstørrelse end .png, men hvis kvaliteten skal se godt ud tæt på, vil .jpeg ikke være det bedste valg. I de fleste tilfælde vil jeg

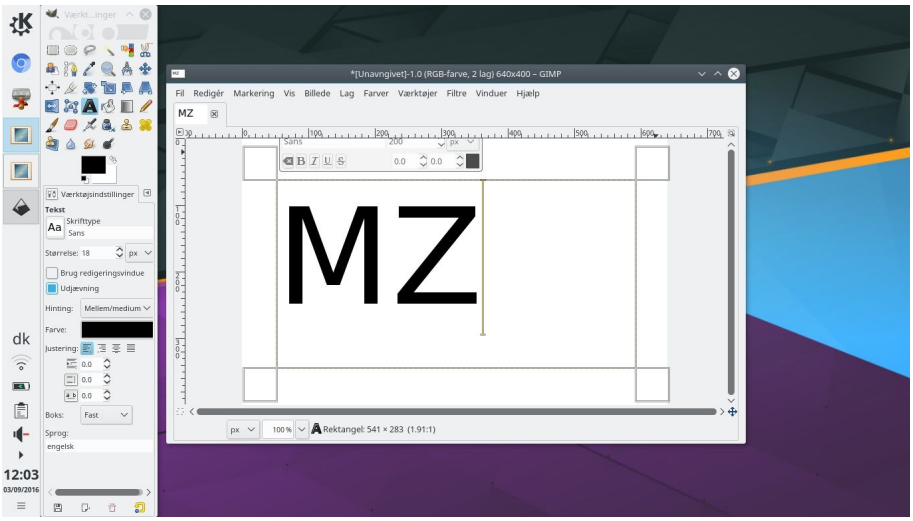
sige, at det ikke gør så stor forskel, men .png har også flere fordele. Hvis du har en .png fil, kan du også gøre baggrunden gennemsigtig, dette skyldes at den har en alphakanal, som gør det mulig at specificere “gennemsigtighed” for hver pixel, i stedet for kun farve. .bmp vil du nok ikke få brugt i din hverdag pga. filstrørrelsen, også derfor vil mange billedoploadingshjemmesider ikke tillade at oplade .bmp.

Øvelse 4 - Vektorgrafik

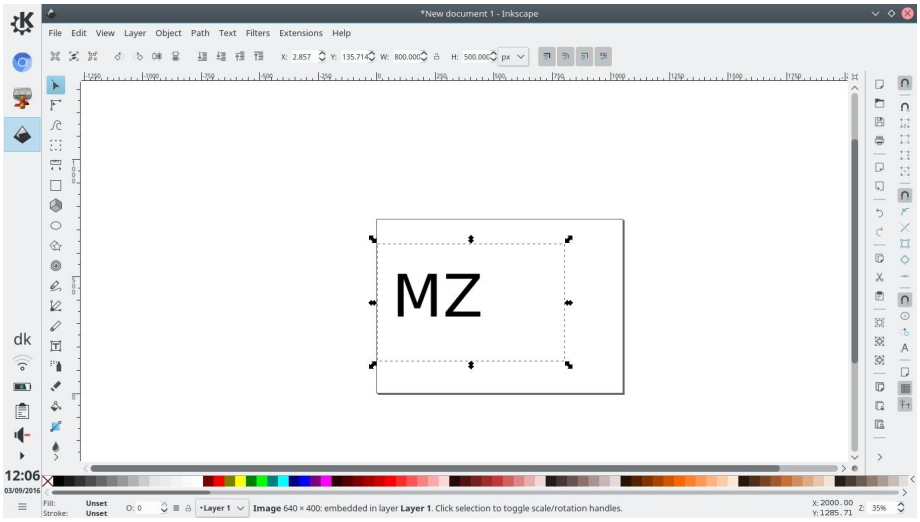
Vektor er det “omvendte” af raster. I raster har man data om hvor hver pixel skal være, mens i vektorgrafik er det lidt anderledes. I vektorgrafik kan vi bruge punkter og matematiske udregninger til at definere grafik, hvilket vil resultere i at grafiske billeder vil se godt ud lige meget hvor meget du zoomer ind.

1. Øvelse: fra raster til vektor

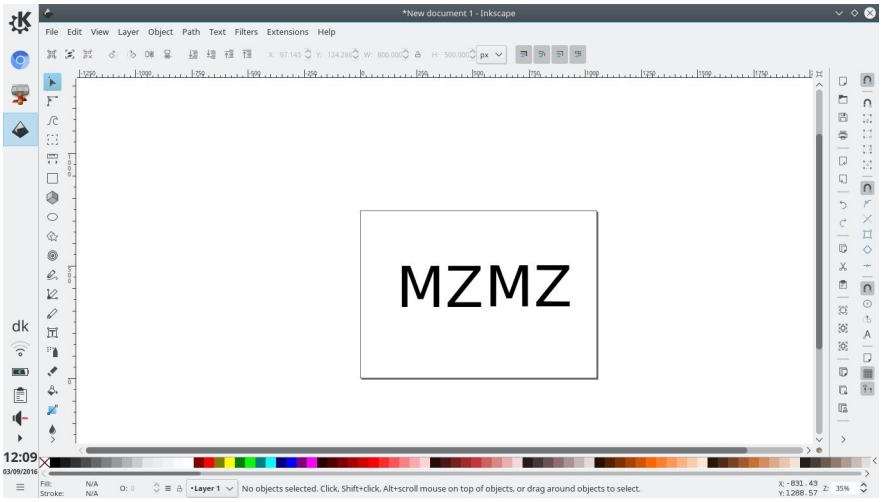
Vi starter med at skrive vores forbogstaver (Mikael Zhang = MZ) i Gimp og eksportere det som en .png fil. Det er en god idé at lave det i ok høj opløsning, da det vil blive nemmere for vores vektorprogram et udregne “punkter”.



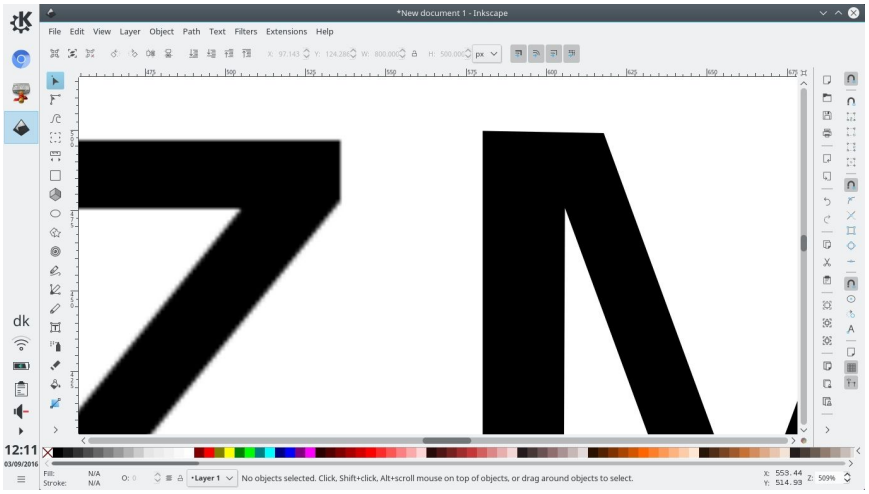
Nu åbner vi så Inkscape, som er lavet til at kunne lave vektorgrafik. Vi importere vores billede.
File -> Import… -> Vælg billede



Vi vil nu få Inkscape til finde punkter i vores raster og lave det om til vektor.
Path -> Trace Bitmap… Standard indstillinger burde være fine.



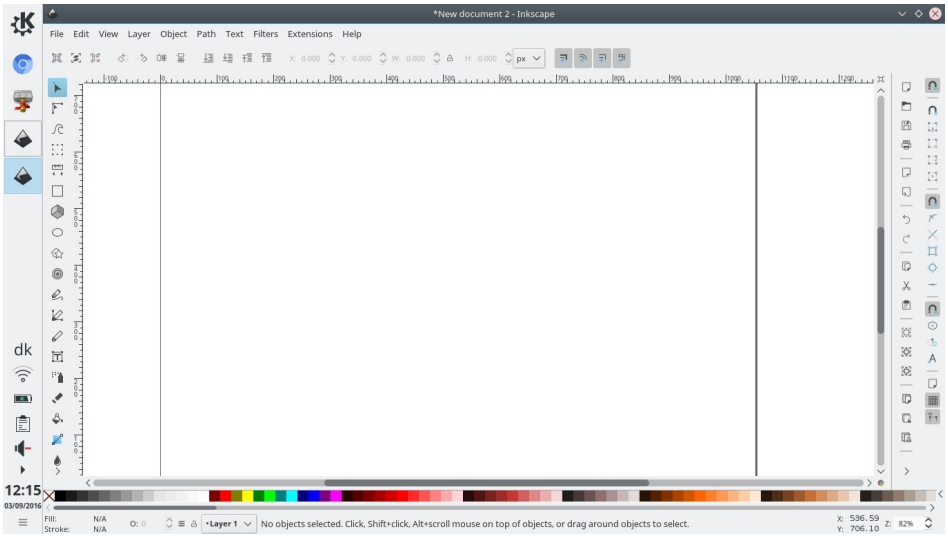
Den ligger et nyt lag ovenpå, som du bare kan rykke med dit flytteværktøj (klik og træk). Vi har nu raster grafiken til venstre og vektor til højre.



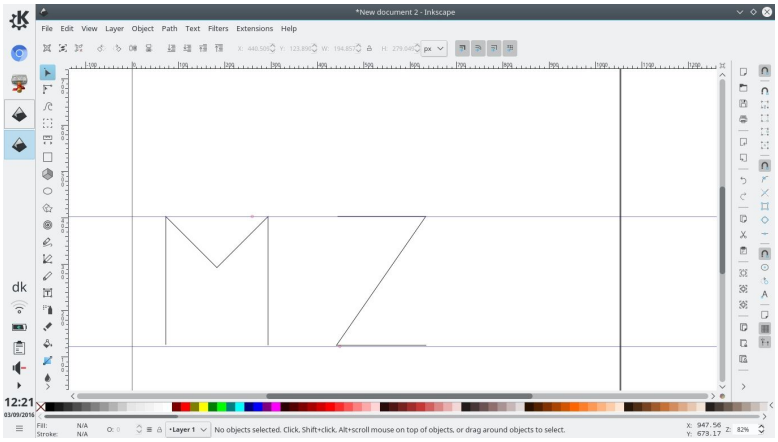
Hvis vi zoomer ind kan du se at vektorgrafikken ikke bliver kantet, mens rastergrafikken gør.

2. Øvelse: Tegn med vektorer

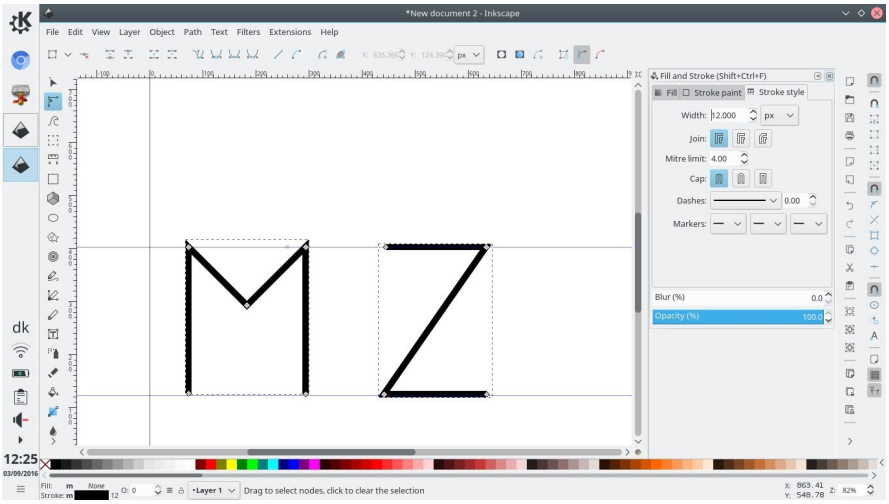
Vi vil nu prøve at tegne med vektorer. Vi vil prøve at tegne forskellige bogstaver (MZ) med Inkscape.

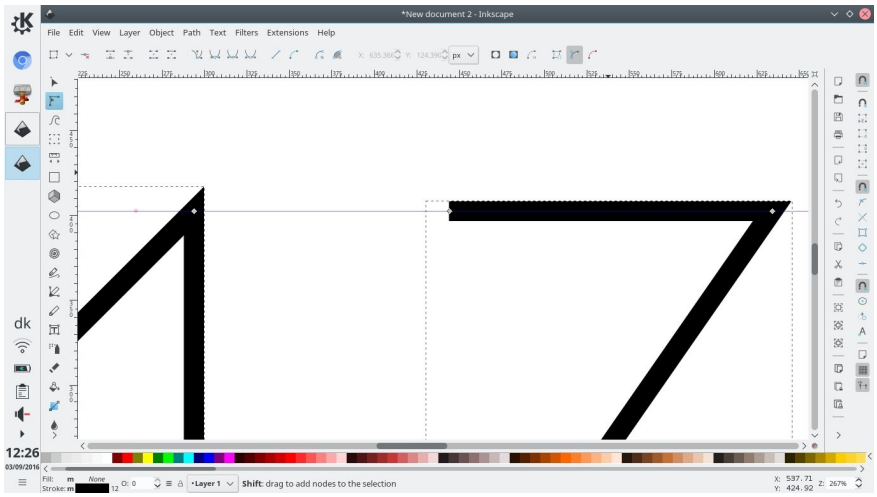


Vi starter med at lave et dokument. Vælger punktværktøjet og starter med lave de første punkter.



Med M og Z er det meget simpelt, men hvis nu du ville tegne noget med buer i, kan du punktjusteringsværktøjet; Først tryk og så træk i dem i retning og den bøje sig med.
Vi vil nu fylde vores streger med lidt mere. Marker dine tegn, *Object -> Fill and Stroke… -> Stroke style -> width -> vælg størrelse.* Her har jeg valgt en width på 12 pixels.





Hvis vi zoomer ind kan vi se at bogstaverne stadige er knivskarpe. Dette er fordelen ved at bruge vektorgrafik.