Aufgabe 2

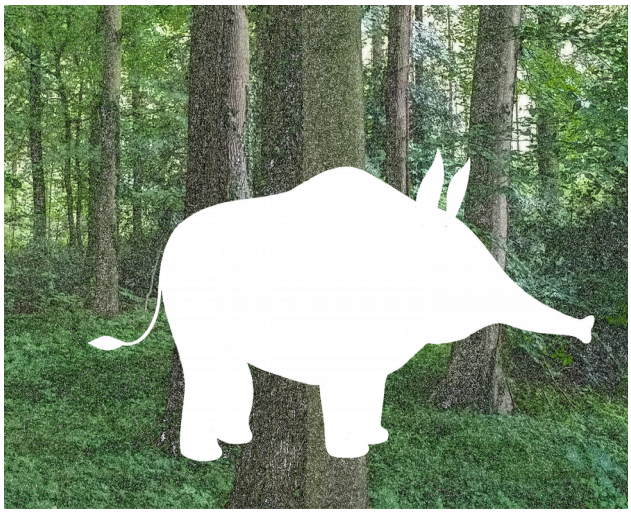
Lösungsidee

Bei dieser Aufgabe habe ich mich an KISS (keep it simple, stupid) gehalten. Ich gehe einfach jeden Pixel durch und überprüfe ob einer der Nachbarn die gleiche Farbe hat, wenn ja dann markiere ich den Nachbarn und den jetzigen Pixel (= füge sie in eine Liste hinzu). Am Ende werden alle markierten Pixel Weiß eingefärbt und das bearbeitete Bild wird in eine neue Datei geschrieben.

Umsetzung

Die Umsetzung wurde (wieder einmal) in Ruby geschrieben. Dieses Projekt ist älter als das andere, und benutzt deswegen noch nicht den Boilerplate Code welcher in der Lösung für Junioraufgabe 1 zu finden ist. Benutzt wurde ChunkyPNG als Library um die Bilder laden und bearbeiten zu können. Das Bild wird einfach geladen, die Pixel werden überprüft und das Bild wird danach ausgegeben. Dazu gibt es dann noch eine Ausgabe in dem Terminal welche einen darüber aufklärt wie lange der ganze Prozess gedauert hat.

Beispiele



Einmal ein Bild mit und ohne Rhinozelfant.

Wichtige Teile des Codes

```
def process name;
  @png = ChunkyPNG::Image.from_file name

  @width = @png.dimension.width
  @height = @png.dimension.height

  @marked = []


  def in_bounds x,y; (x >= 0 && x < @width-1 && y>=0 && y<@height-1) end

  def mark_neighbours x,y
    possibilities = [[x+1,y],[x+1,y+1],[x,y+1],[x-1,y],[x-1,y-1],[x,y-1]]
    possibilities.each do |po|
      if(in_bounds po[0],po[1] ) then
        if @png[x,y] == @png[po[0],po[1]] then
          @marked << Position.new(po[0],po[1])
        end
      end
    end
  end

  @png.height.times do |y|
    @png.row(y).each.with_index do |pixel, x|
      mark_neighbours x,y
    end
  end

  @marked.each do |pos|
     @png[pos.x,pos.y] = ChunkyPNG::Color.rgb(255,255,255)
  end

  @png.save "out.#{name}"
end
```

Diese Methode macht die ganze Arbeit.

Verwendung

Da dies (wieder einmal) Ruby ist muss auch hier wieder Ruby auf dem Host System vorhanden sein. Außerdem muss das ChunkyPNG gem installiert sein, welches einfach über *bundle install* (Falls Bundler installiert ist) oder mit *gem install chunky_png* installiert werden kann.

Sobald die Abhängigkeiten installiert sind kann das Skript wie folgt benutzt werden:

ruby main.rb <pfad_zur_datei>

Hierbei ist zu beachten, dass, falls von der Shell unterstützt (ich benutze ZSH), auch eine wildcard benutzt werden kann um die Dateien zu bearbeiten.

*ruby main.rb *.png*

Das würde alle .png Dateien bearbeiten wenn die Shell das so interpretiert, dass aus *.png eine liste aller Dateien werden soll die auf .png enden.