# Introduction

SDF Map Tool it's a command line utility that will generate smooth, artist-authored shadow transitions for being used in our face shader.
The tool will take any number of images and generate a single greyscale image that will be then interpreted by the game shader as a 180 degree mapping of how the shadows should appear in relation to the light direction.

# Installation

This tool is bundled in the Modding Tools package next to the provided documentation. Both Windows and Linux builds are included. With every update of the Modding Tools, a newer version of Sdf Map Tool is bundled if available.

If you wish to manually update it, the GitHub repository is available at https://github.com/Circle2Labs/SdfMapTool. No installation required, just extract and run.

# Usage

First of all, you should prepare a set of images: these should match your base mesh skin UV map and be all of the same resolutions.

The image resolution will impact also the resolution of the shadows on the face. We recommend at least 1024x1024 as resolution.

The way you should paint them is by using a full black background and a full white brush. greyscales will be ignored. The first image will be mapped as light coming from full left, the last image from full front.

The more images you add to the count, the more this 90 degree angle mapping will be split in equal parts.

The opposite part will be treated as a mirror by the shader. Due to this limitation, you should aim to have a symmetrical UV unwrap for your face mesh.

As an example, with 3 images, you'd be having:

- the first being 0 degrees (from left side),
- the second would be 45 degrees,
- the third would be full front (90 deg).

Likewise, with 4 images:

- first is always 0 deg,

- second would be 30 deg,
- third would be 60,
- and 4th would be 90 as usual.

Assuming you got the images ready, let's look into how the tool works.

You'll need to use a command line for using this tool.
The command line for using it is as follows: `SDFMapTool -D <directory_with_images> -i "<image1.png;image2.png;...>" -o <output_file_path.png> ...`

If your terminal is opened in the same folder as where the tool has been extracted, and the images are together there, inside an " images" folder the command line to run would look like: `SDFMapTool -D "./images" -i "01.png;02.png;03.png;04.png" -o "./sdf.png"`

The available options for the command line application are as follows:

- `-d or --debug`
  - Enables debug mode.
  - Outputs multiple images at every step of the process.
  - Useful to report eventual bugs, should you find any.
- `-D or --imgDirectory`
  - Required, Specifies the folder where to look for the input image files.
  - If the folder is the same as where the executable is placed, either `./` or `.\` depending on your current OS, will work as path.
- `-i or --images`
  - Required. Images file names separated by ';' and in the correct order.
- `-o or --output`
  - Required. Output file path.
- `-b or --blur`
  - How many blur iterations to run.
  - More iterations will return a more homogeneous final result (requiring slightly longer process time)
  - Each blur iterations is repeated on the result from the previous one
- `-r or --radius`
  - Blur radius for each iteration.
  - Larger radius samples more pixels on every edge, altering larger parts of the image on every iteration.
- `-s or --step`
  - Blur step size.
  - How much the pixels will get moved from their original position
  - Bigger step sizes must be compensated by running more blur iterations in order to avoid harsh steppings to show up.

    - Smaller step sizes still benefit by higher blur iterations
- `--sigma`
    - Blur sigma value (weighting).
    - This value determines how much the original pixel will blend with the new pixel
- `--device`
    - Which device to use for computation.
    - Accepted values are: `CPU, OpenCL, CUDA`
    - When not specified the system "suggested highest performance" device will be chosen.
    - This allows to override the choice.
    - The calculations performed are exactly the same, there is no reason to run CPU if an OpenCL (AMD) or CUDA (nvidia) device is available as those are going to be an order of magnitude faster
- `--help`
    - Display a short help screen with all the parameters, basically a summary of this section.
- `--version`
    - Displays the version information.

When not specified, the blur values are set automatically to a known good setting which should work fine on 1024x1024 and bigger images.

We suggest using between 4 and 6 images for generating a good SDF map, as lower amounts of viewpoints will net potentially unexpected results and going higher will be a waste of time painting the map unless there are very specific transitions at some specific points required.

Beware that duplicating images to pad their duration will not work and that the movement across the different images should be smooth and accurate. As such, try to always start a new viewpoint by reusing the previous one.