# SHADERS

## WHAT, WHERE, HOW

**TARU SIMONEN**

# AGENDA

- ME
- RIGHTWARE
    - sponsor company presentation
- SHADERS
    - Basics
    - Examples
    - Getting started right now

# FIRST, WHO?

## TARU SIMONEN - KITAI

- Aalto University fresman 2007
- Demoscener
- Technical Designer, Rightware 2014 -

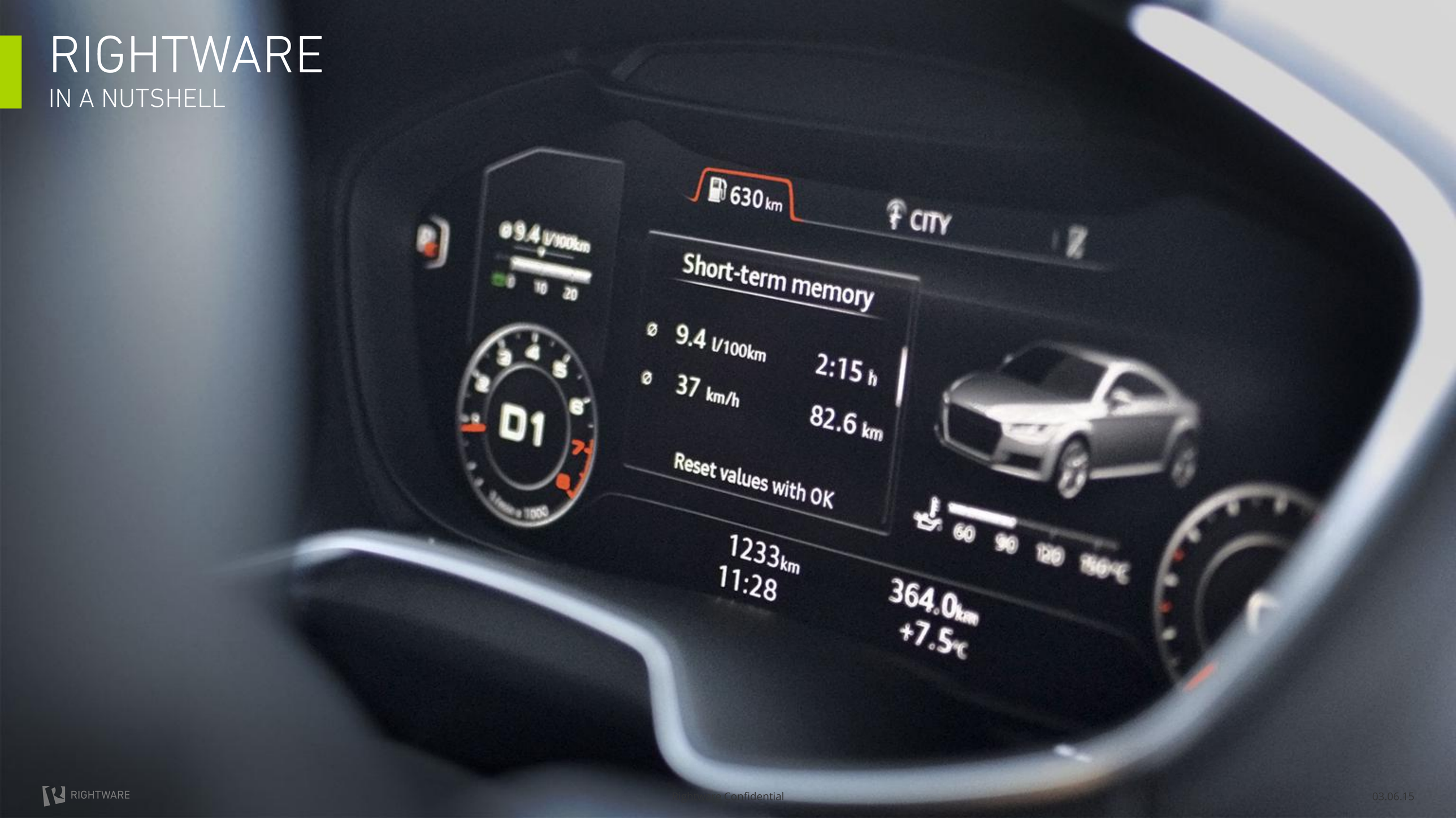
# RIGHTWARE

## LEADING UI SUPPLIER FOR AUTOMOTIVE INDUSTRY BY 2017

# EXECUTIVE SUMMARY

**EXECUTIVE SUMMARY** / Kanzi product family / Market / Organization

# RIGHTWARE
## IN A NUTSHELL



# RIGHTWARE
## IN A NUTSHELL

**KANZI**

User interface creation software

**40+**

**employees**
in Finland, US, Germany, China, Japan and Korea

**~30%**

**Market share** in digital clusters

Leading provider of Advanced UI Solution & Graphics Benchmarks for Automotive markets

Rightware selected as “Growth Builder 2015” by Finnish Venture Capital Association

# TARGET DE-FACTO HMI SOLUTION


# KANZI STUDIO

- User interface design tool
- Seamless 2D/3D authoring
- Build user interfaces without writing code
- Built-in scripting capabilities
- Compatible with all major design software

RIGHTWARE

# SHADERS

# SHADERS – WHAT?

ANY MODERN 3D RENDERING SOFTWARE USES SHADERS SOMEWHERE, WHETHER OR NOT USER KNOWS IT

## VERTEX SHADER

*Little program which is calculated for each vertex on the screen separately*

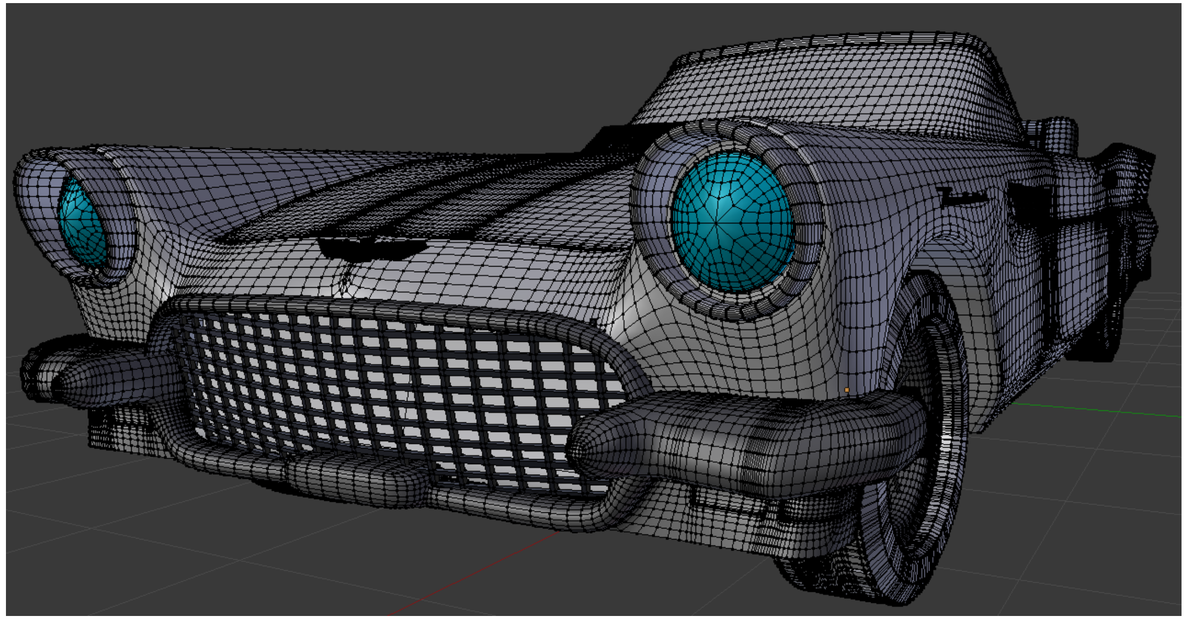
- GPU
- Optimized light calculations in real time
- Mesh morphing

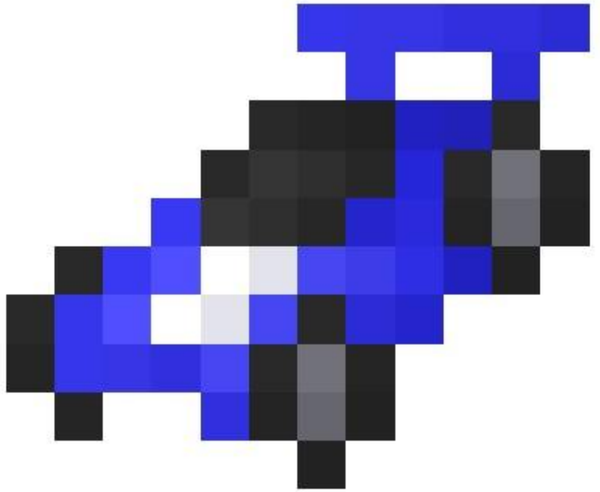
## PIXEL SHADER

Little program which is calculated for each pixel/fragment on the screen separately

- GPU
- Abstract visuals – demos
- Real looking objects – materials

# VERTICES? PIXELS?? FRAGMENTS???

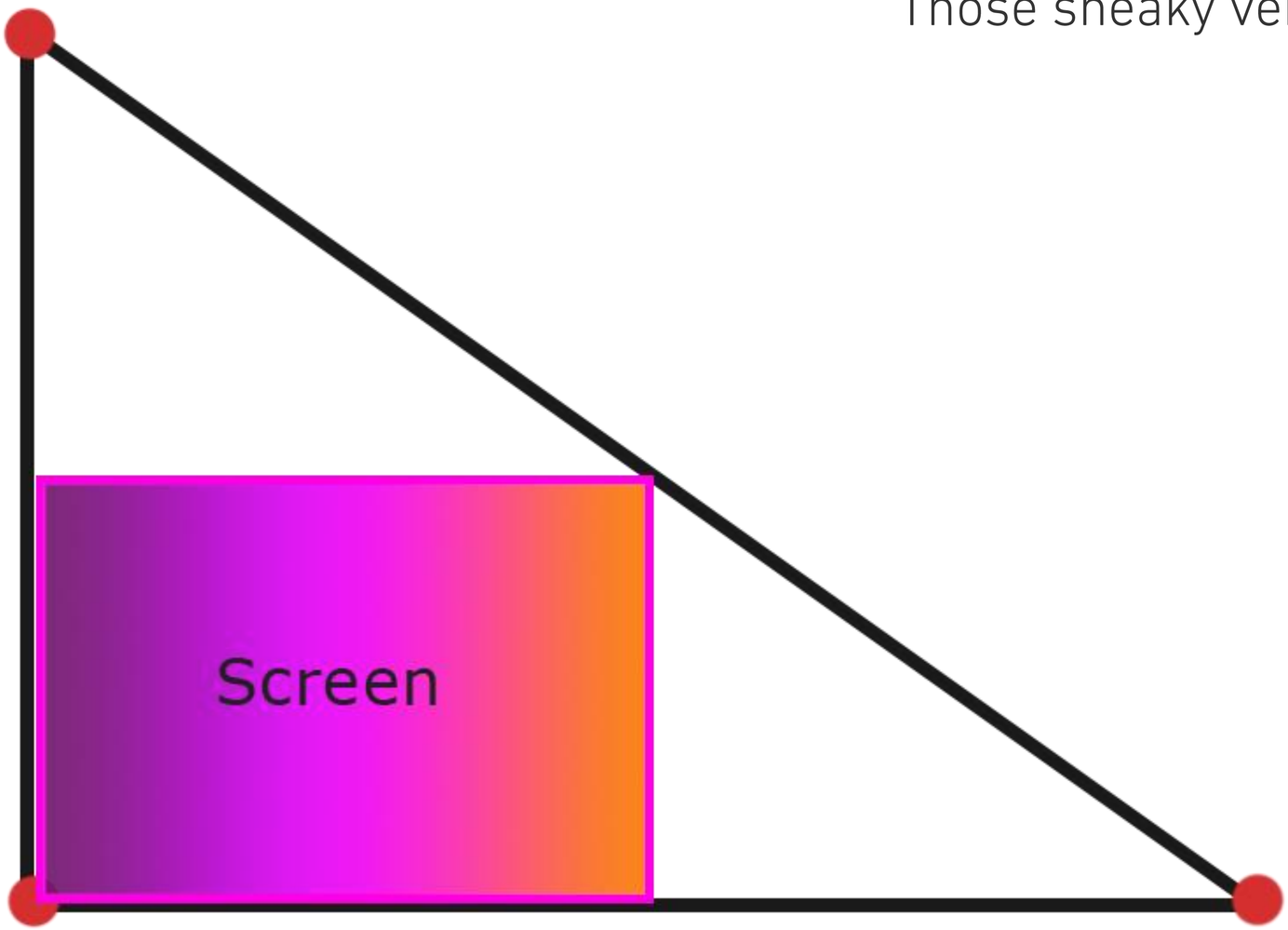




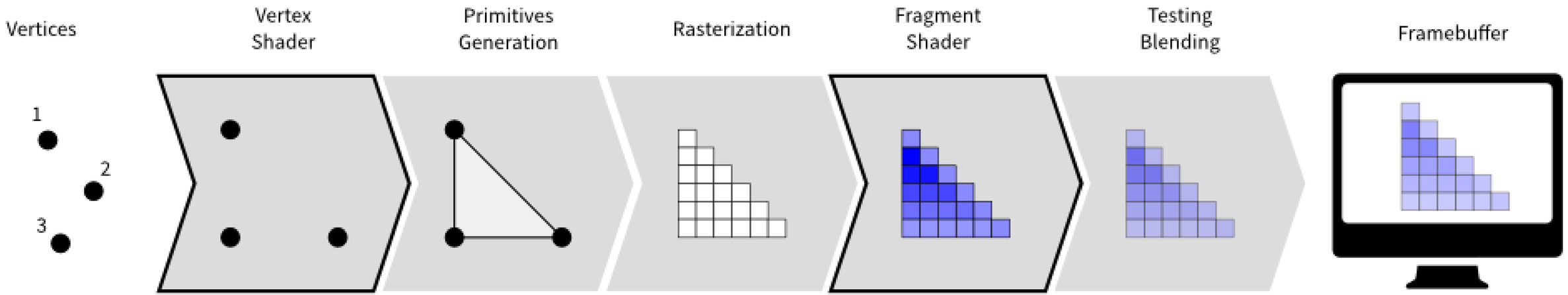
# "BUT I DONT WANNA PLAY WITH VERTICES..."

YOU ALWAYS HAV E AT LEAST 3 VERTICES ON YOUR SCREEN:

"Those sneaky vertices!!"



# SIMPLIFIED GPU PIPELINE



# GLSL VS HLSL

GLSL

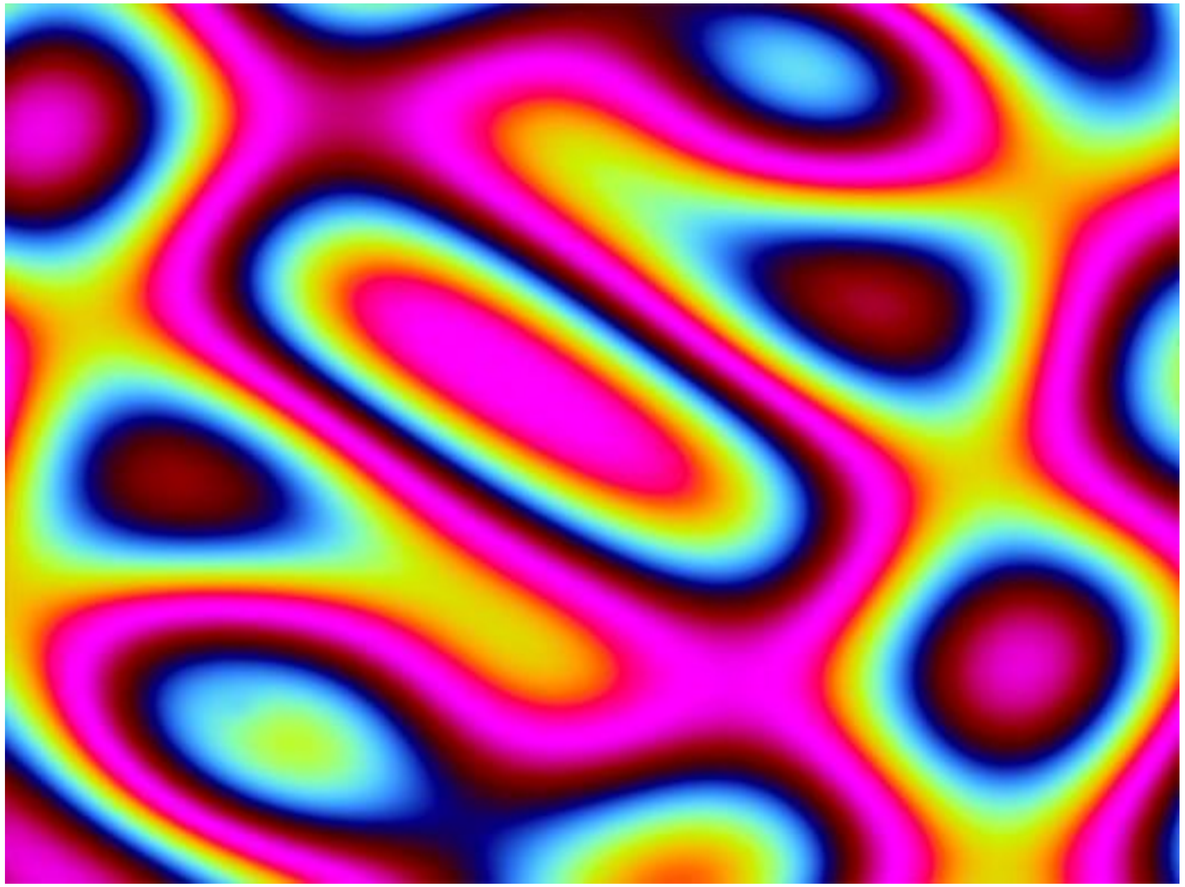
- OpenGL shading language
- Fragment shader
- Syntax based on C
- Multiplatform
- Processing has OpenGL integration
- Khronos

HLSL

- Direct3D shading language
- Pixel shader
- Microsoft

# HOW DO SHADERS LOOK LIKE?

## IT DEPENDS...



# HOW ON EARTH?

## LET'S PLAY AROUND WITH SHADERTOY...

# SO BASICALLY…

- Implement a function
- Feed your pixels' positions on the screen to your function
- Function patiently calculates a color for each pixel/fragment as you have defined

# PROFESSIONAL USE?

MATERIALS – SHADERS DEFINE HOW…

- Textures are wrapped around objects
- Light gets absorbed to objects surface
- Fine detail such as normal maps are interacting with light on top of mesh to add detail
- Shadowed parts look like
- Environment reflections (real time faked)
- Etc…

# "COOL! CAN I?"

## "YES, LET'S"

- Create a shader for example in Shadertoy
- Create a folder called "data" under your Processing project folder.
- Open notepad or such and paste your shaderprogram.
- Add uniform to the beginning of your file. You'll probably need at least these:
    - Uniform vec2 iResolution;
    - Uniform float iglobalTime;
- Save your shader to the data folder with .glsl ending.

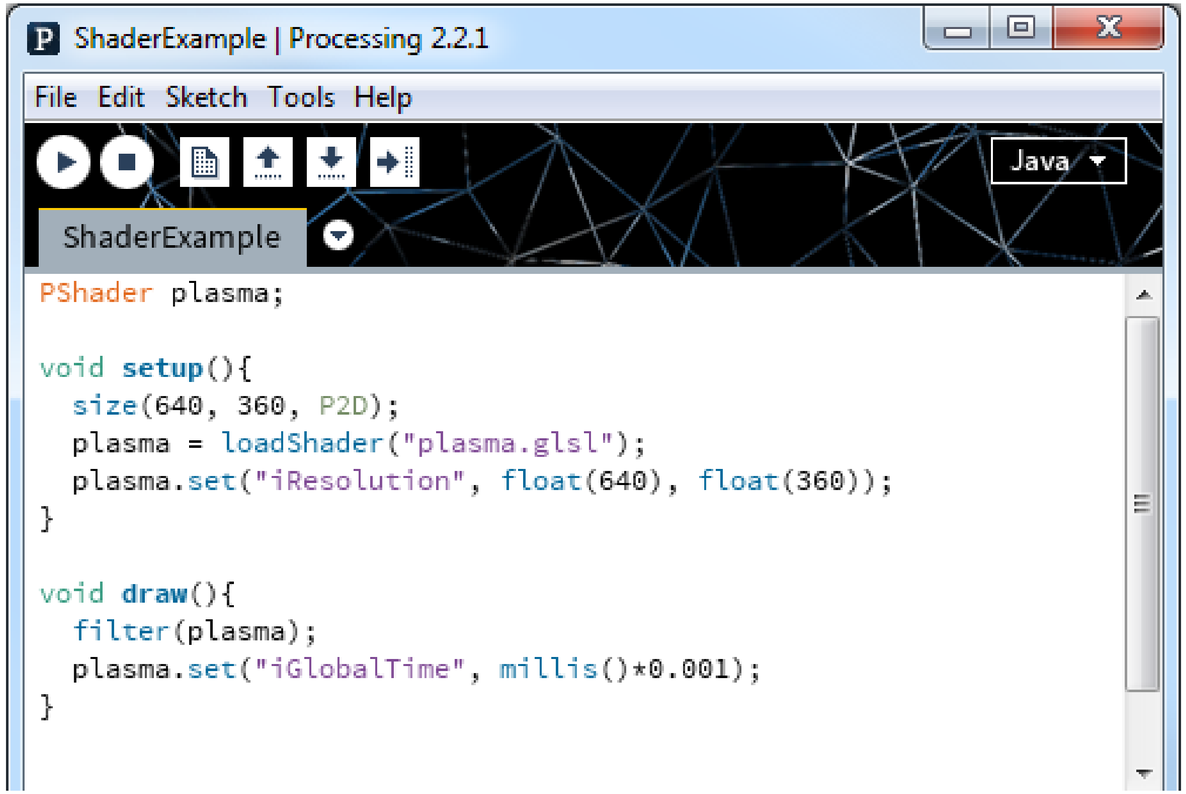
# COUPLE OF NAME CHANGES

## IMPORTING FROM SHADETOY.COM WON'T WORK RIGHT AWAY, CHANGE THESE FIRST:

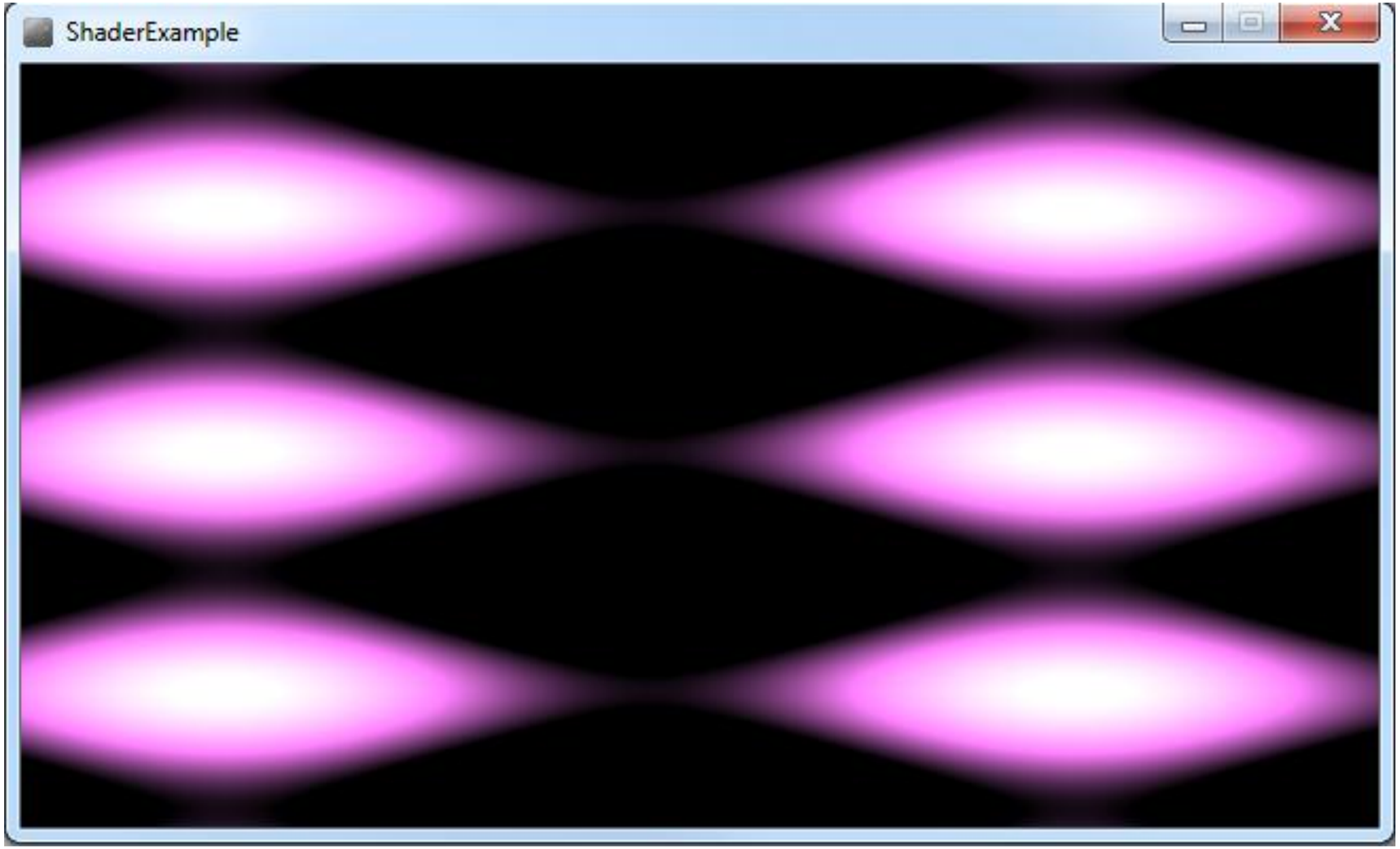
- " void mainImage( out vec4 fragColor, in vec2 fragCoord )" → " void main()"
- " fragCoord" → " gl_FragCoord"
- " fragColor" → " gl_FragColor"

# IN YOUR PROCESSING PROJECT

- Define name for your shader
- Use set() function to give values for your uniforms
- Set iGlobalTime inside of draw() function – Otherwise no movement!



# ENJOY!



# HOT WORDS TO KEEP ON LEARNING!

- SHADER
- GLSL
- HLSL
- OpenGL
- Directs3D
- Cubemap
- Normal map
- Phong
- Diffuse
- Fresnel
- Lambertian
- Raymarch
- Raytrace
- Physically based rendering
- Cook Torrance BRDF

# QUESTIONS?

# THANK YOU!

Taru Simonen
Rightware
Technical Designer

taru.simonen@rightware.com