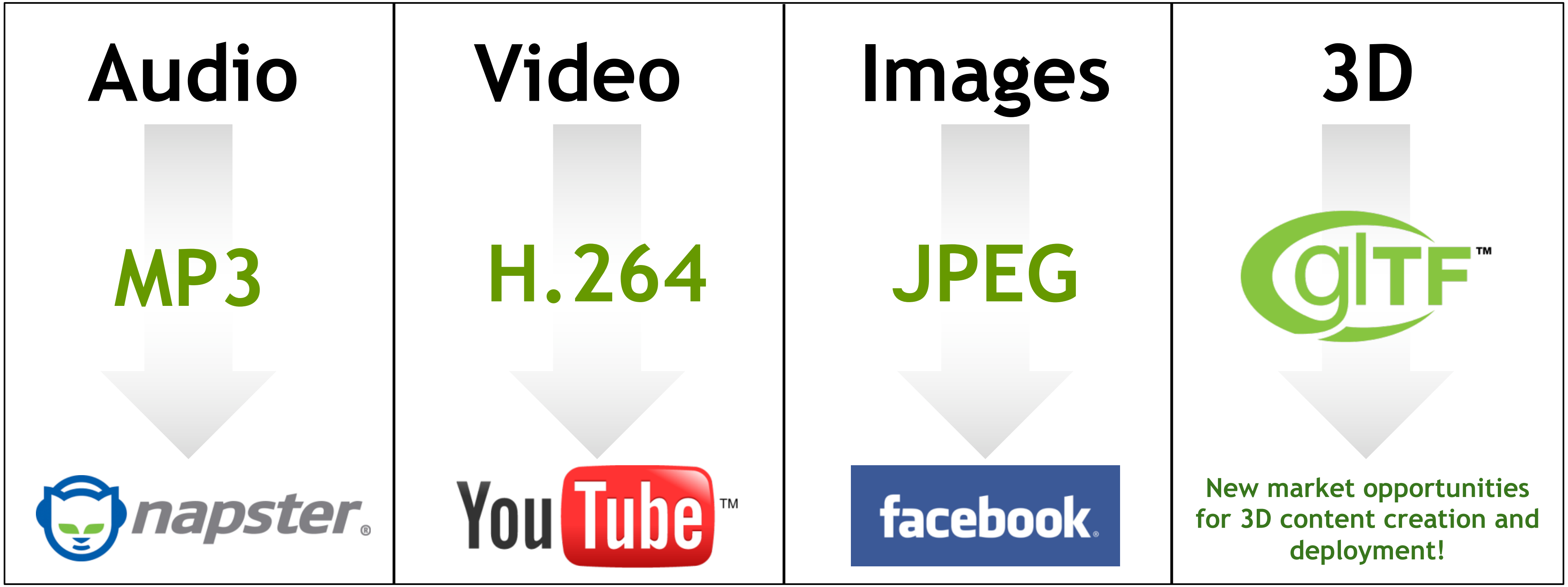
glTF 2.0: PBR Materials

GTC, May 2017

Saurabh Bhatia
Program Manager, Microsoft | glTF specification contributor
saurabh@microsoft.com | @iamSBTron

glTF - Runtime 3D Asset Delivery



Audio

MP3

Video

H.264

Images

JPEG

3D

New market opportunities for 3D content creation and deployment!

model/gltf+json MIME type Approved by IANA

glTF™

Compact to Transmit ✓

Fast to Load ✓

Describes Full Scenes ✓

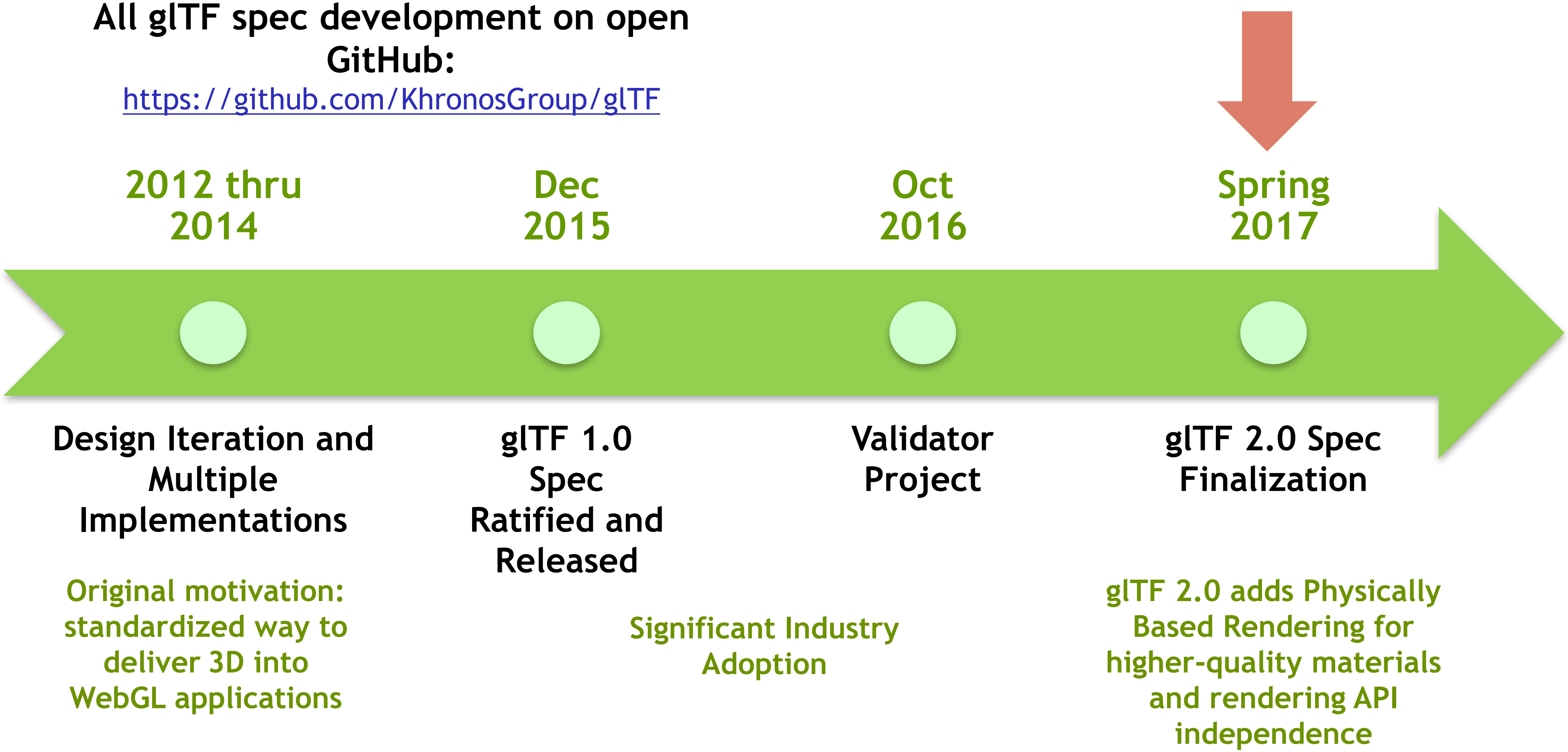
Runtime Neutral ✓

Extensible ✓

glTF Milestones

All glTF spec development on open GitHub:
https://github.com/KhronosGroup/glTF

We are here!



2012 thru 2014

Design Iteration and Multiple Implementations

Original motivation: standardized way to deliver 3D into WebGL applications

Dec 2015

glTF 1.0 Spec Ratified and Released

Oct 2016

Validator Project

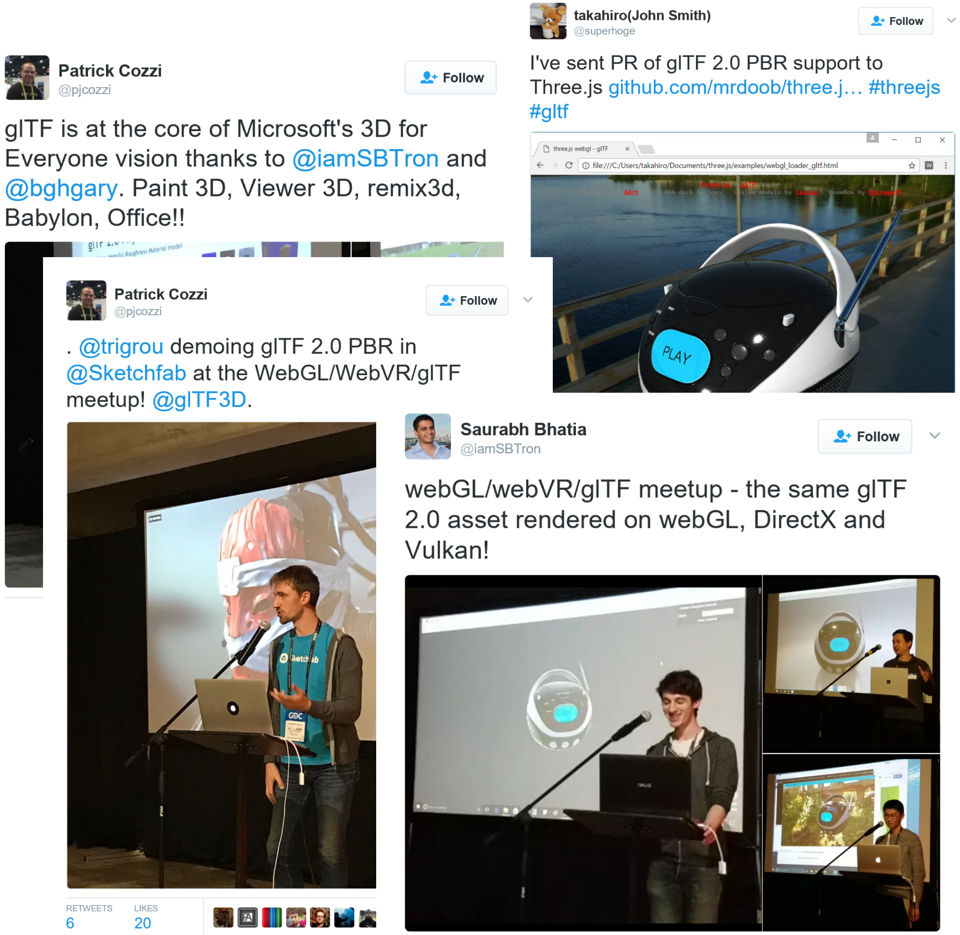
Significant Industry Adoption

Spring 2017

glTF 2.0 Spec Finalization

glTF 2.0 adds Physically Based Rendering for higher-quality materials and rendering API independence

Strong glTF Momentum



Patrick Cozzi @pjcozzi

glTF is at the core of Microsoft's 3D for Everyone vision thanks to @iamSBTron and @bghgary. Paint 3D, Viewer 3D, remix3d, Babylon, Office!!

takahiro(John Smith) @superhoge

I've sent PR of glTF 2.0 PBR support to Three.js github.com/mrdoob/three.j… #threejs #gltf

Patrick Cozzi @pjcozzi

. @trigrou demoing glTF 2.0 PBR in @Sketchfab at the WebGL/WebVR/glTF meetup! @glTF3D.

RETWEETS 6 LIKES 20

Saurabh Bhatia @iamSBTron

webGL/webVR/glTF meetup - the same glTF 2.0 asset rendered on webGL, DirectX and Vulkan!

Follow
three.js webgl - glTF
file:///C:/Users/takahiro/Documents/three.js/examples/webgl_loader_gltf.html
PLAY

glTF™

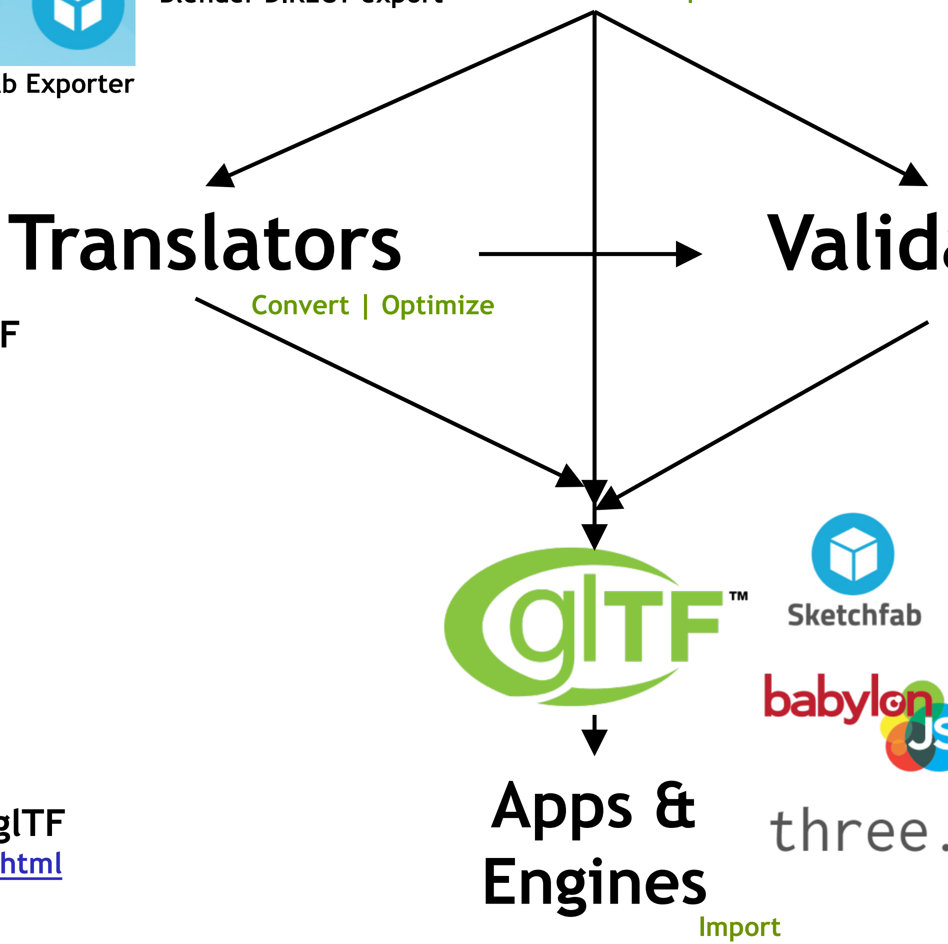
oculus
web3D CONSORTIUM
MPEG
Adobe
blender™
OGC
box
otoy
Fraunhofer
Microsoft
NVIDIA.
AREA Augmented Reality for Enterprise Alliance
Sketchfab
CESIUM
AUTODESK.
Google™

Publicly Stated Support for glTF

glTF Ecosystem

Unity to Sketchfab Exporter

Blender DIRECT export



KHRONOS GROUP

glTF open source Validator
(will support glTF 2.0)

AUTODESK

Drag and Drop FBX -> glTF
(coming soon)
http://gltf.autodesk.io/

Other Translators

AssImp
OBJ2GLTF
glTF Pipeline
COLLADA2GLTF
Cesium converter

COLLADA

Drag and drop COLLADA -> glTF
http://cesiumjs.org/convertmodel.html

Sketchfab

NVIDIA nvpro-pipeline

A-FRAME

babylonJS

CESIUM

three.js

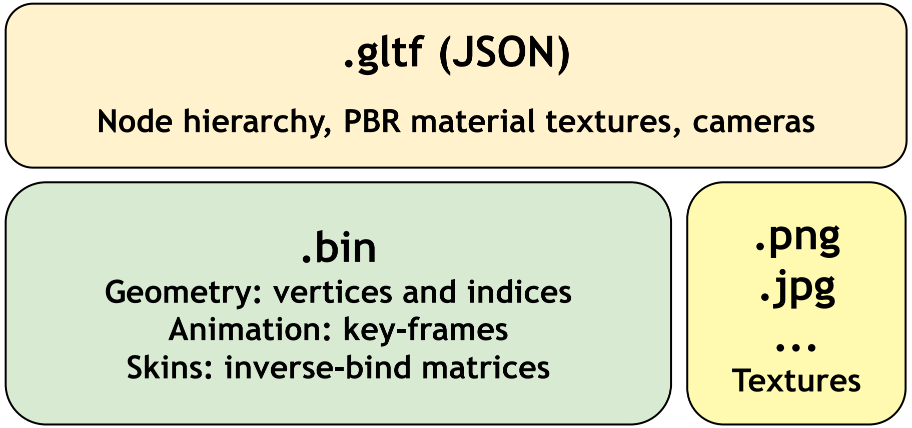
PEX

xeoEngine

What’s new in glTF 2.0

- **Physically Based Rendering (PBR) material definitions**
  - Material information stored in textures
- **Graphics API neutral**
  - GLSL materials moved to extension
  - Proven by implementations using WebGL, Vulkan and Direct3D
- **Morph Targets**
- **Improvements**
  - Binary glTF in core
  - Enhanced Performance

glTF 2.0 Scene Description Structure



.gltf (JSON)

Node hierarchy, PBR material textures, cameras

.bin
Geometry: vertices and indices
Animation: key-frames
Skins: inverse-bind matrices

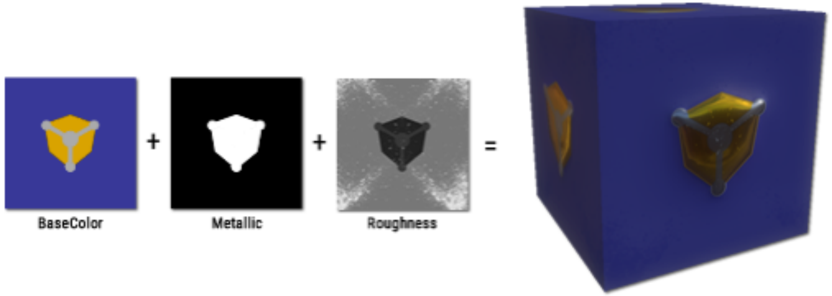
.png
.jpg
...
Textures

Geometry

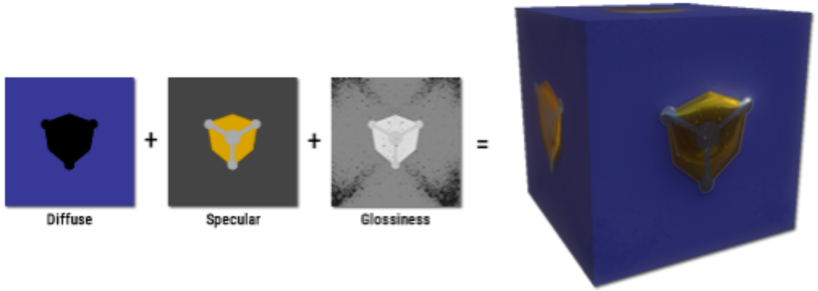
Texture based PBR materials

glTF 2.0 Physically Based Rendering

- **In Core: Metallic-Roughness Material model**
  - baseColor – base color
  - metallic – metalness
  - roughness – roughness



- **Extension: Specular-Glossiness Material model**
  - diffuse – reflected diffuse color
  - specular – specular color
  - glossiness – glossiness

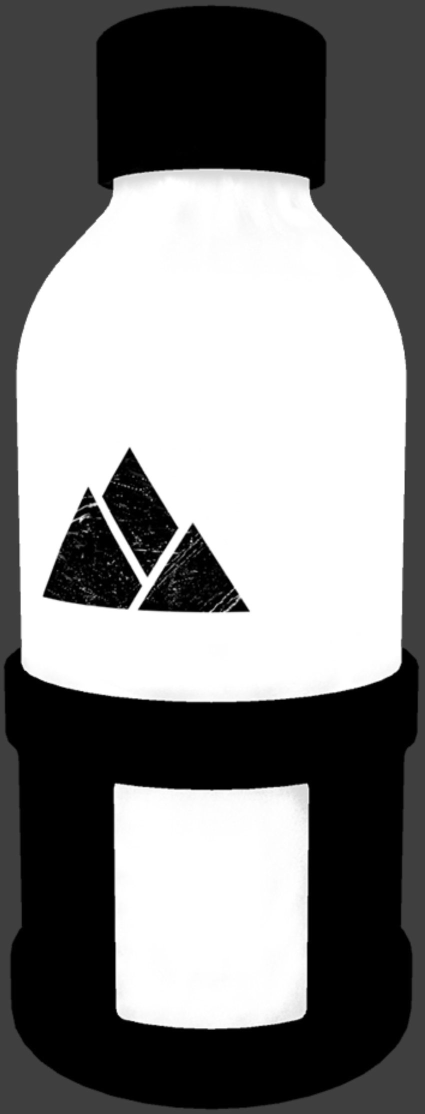


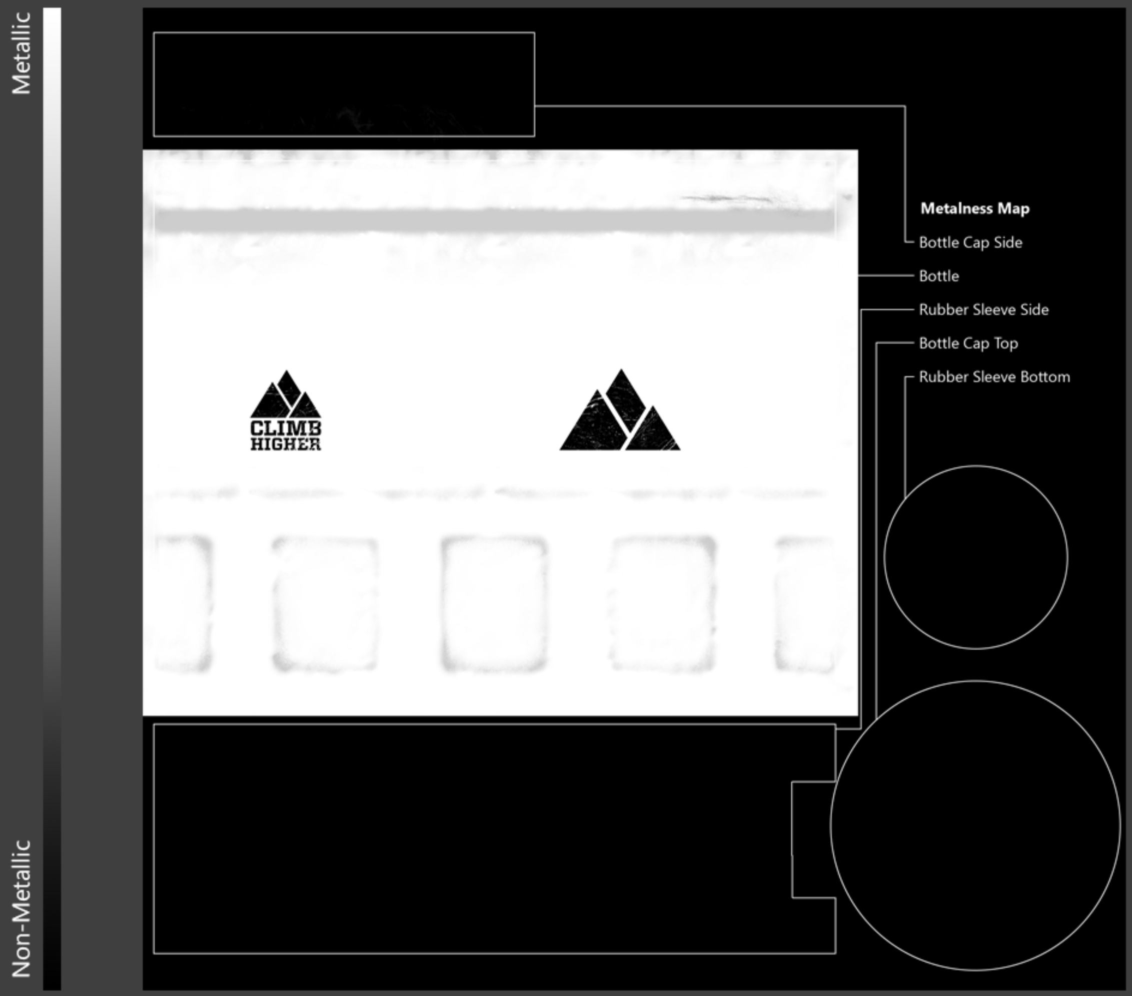
The two models can be combined

Metallic Roughness

PBR Materials

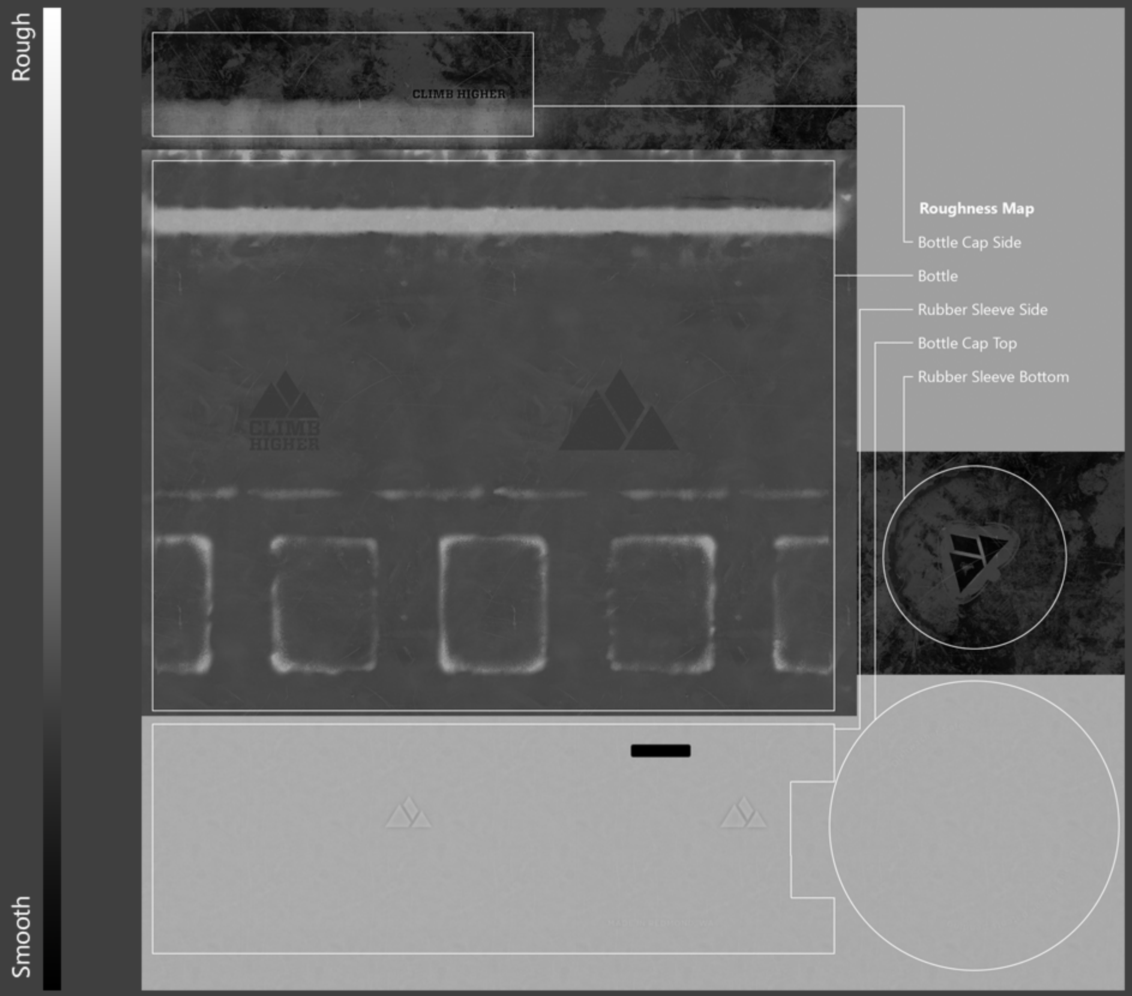
CLIMB HIGHER
2:10 pm · 61°
MADE IN REDMOND, WA



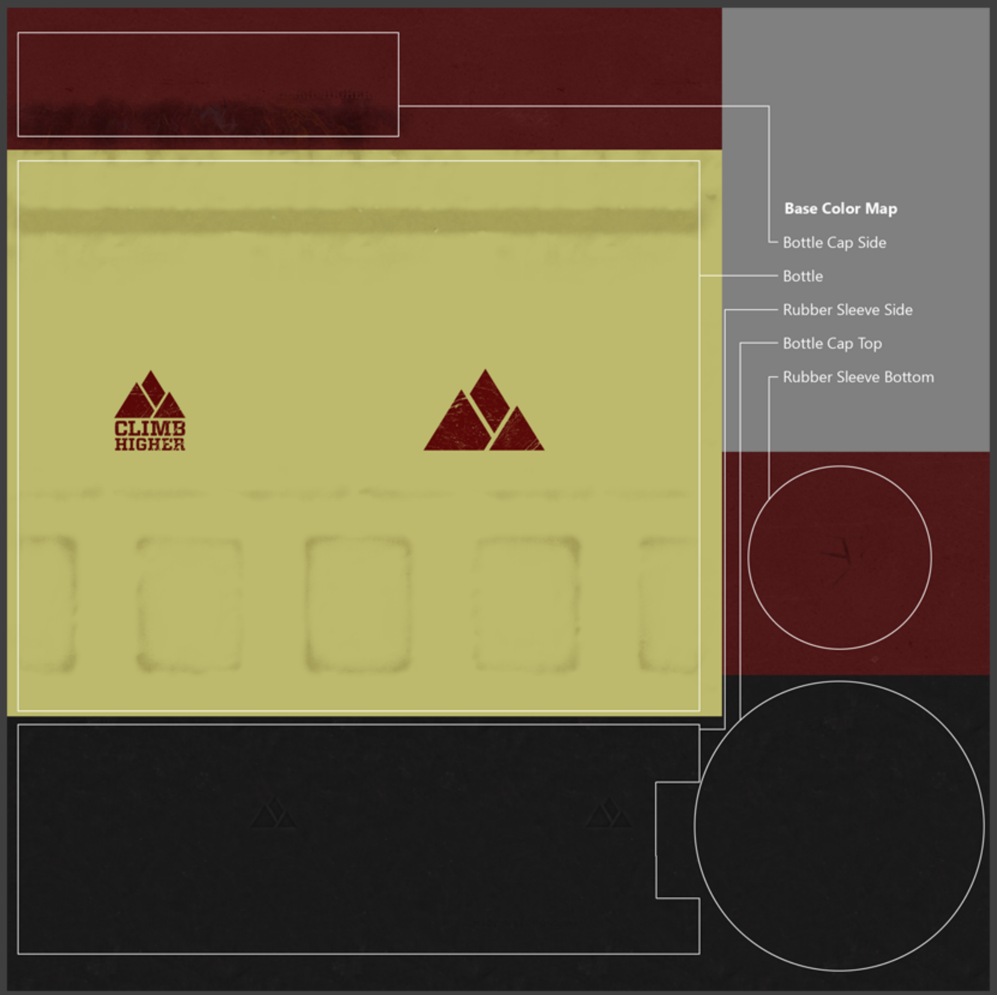


Metalness Map

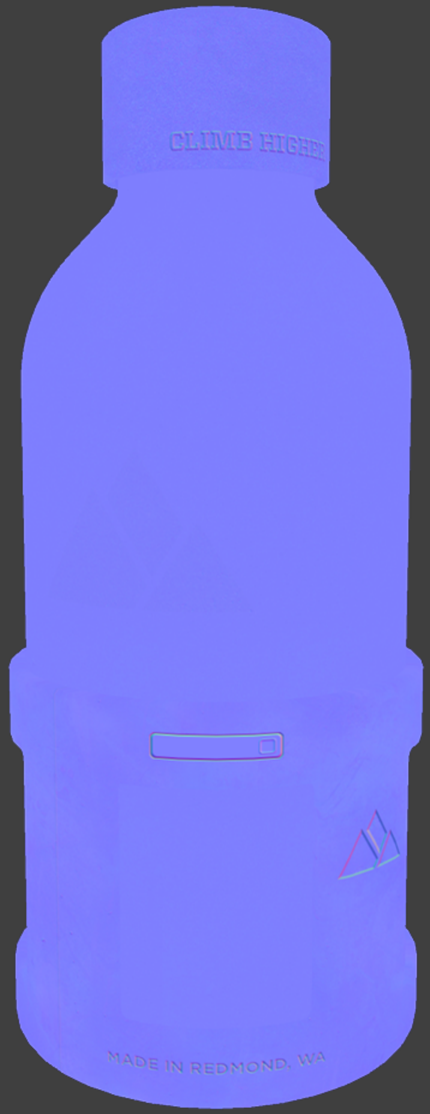
CLIMB HIGHER
MADE IN REDMOND, WA

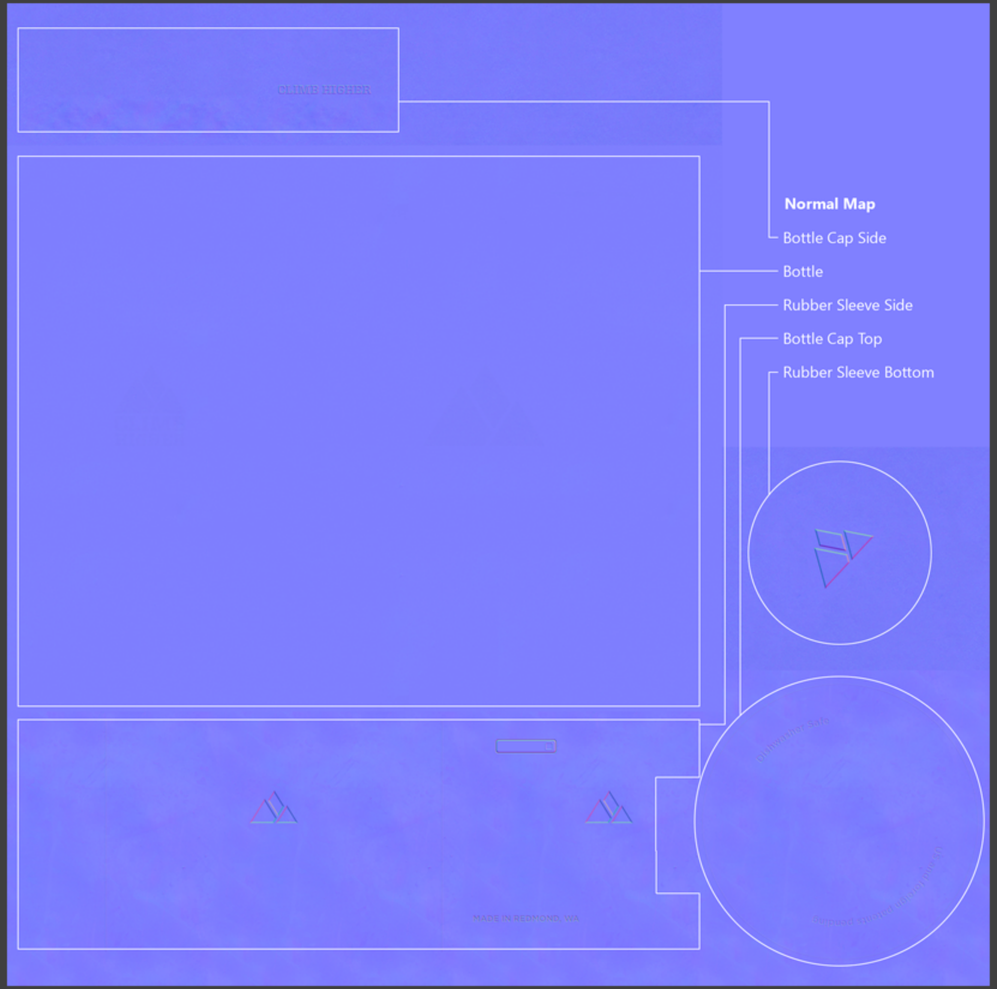


Roughness Map



Base Color Map

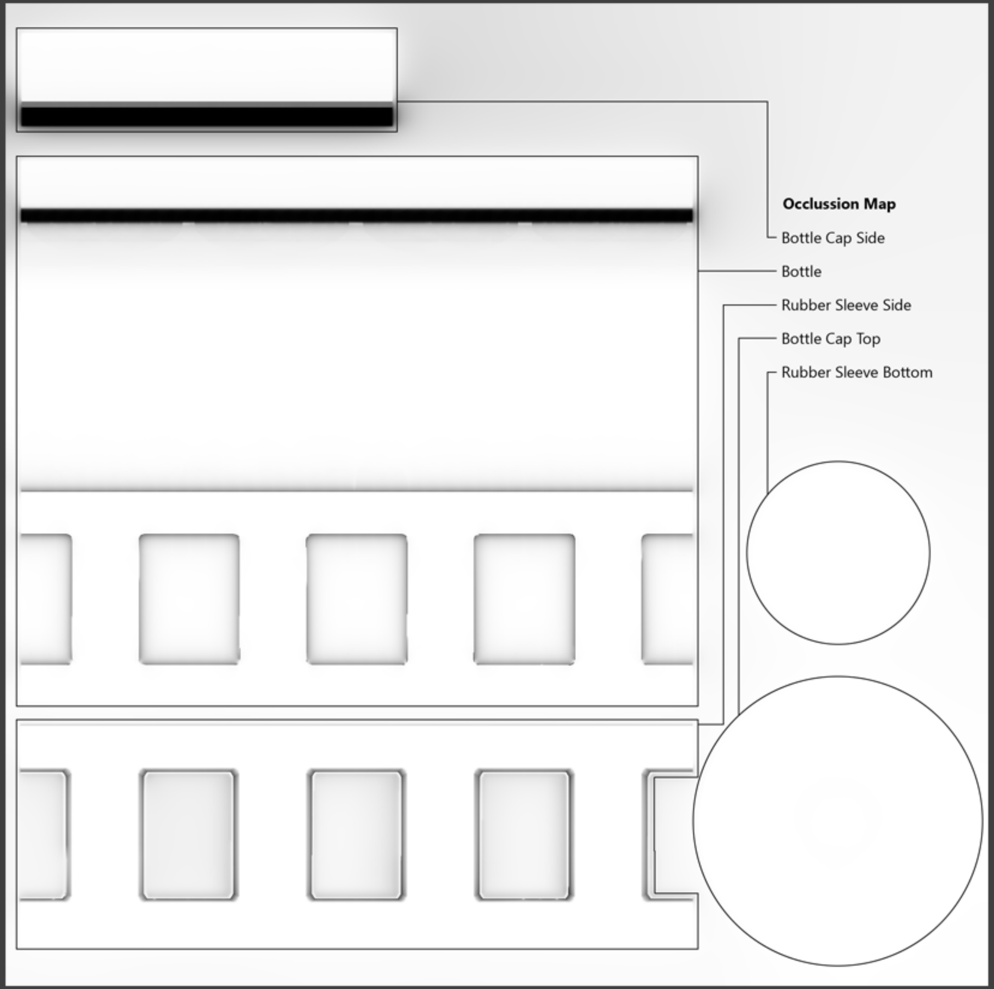




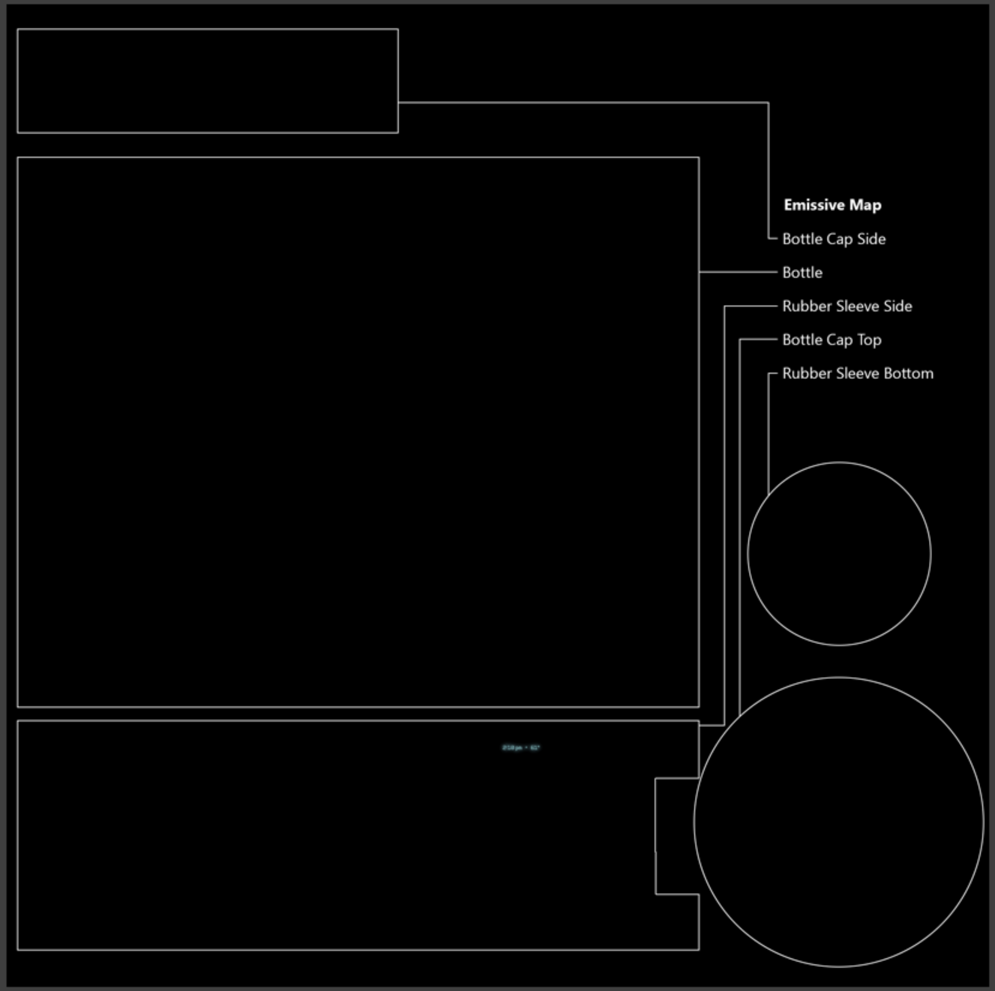
Normal Map

- Bottle Cap Side
- Bottle
- Rubber Sleeve Side
- Bottle Cap Top
- Rubber Sleeve Bottom

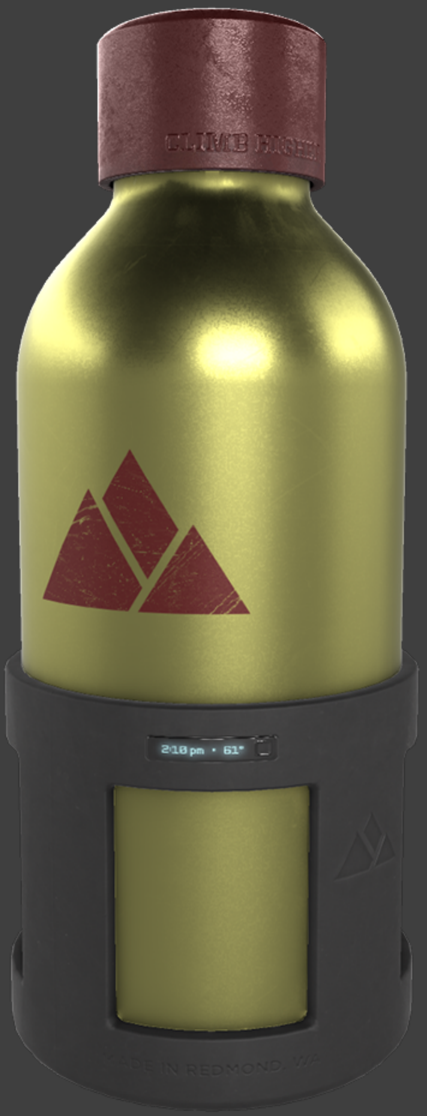
Normal Map



Occlusion Map



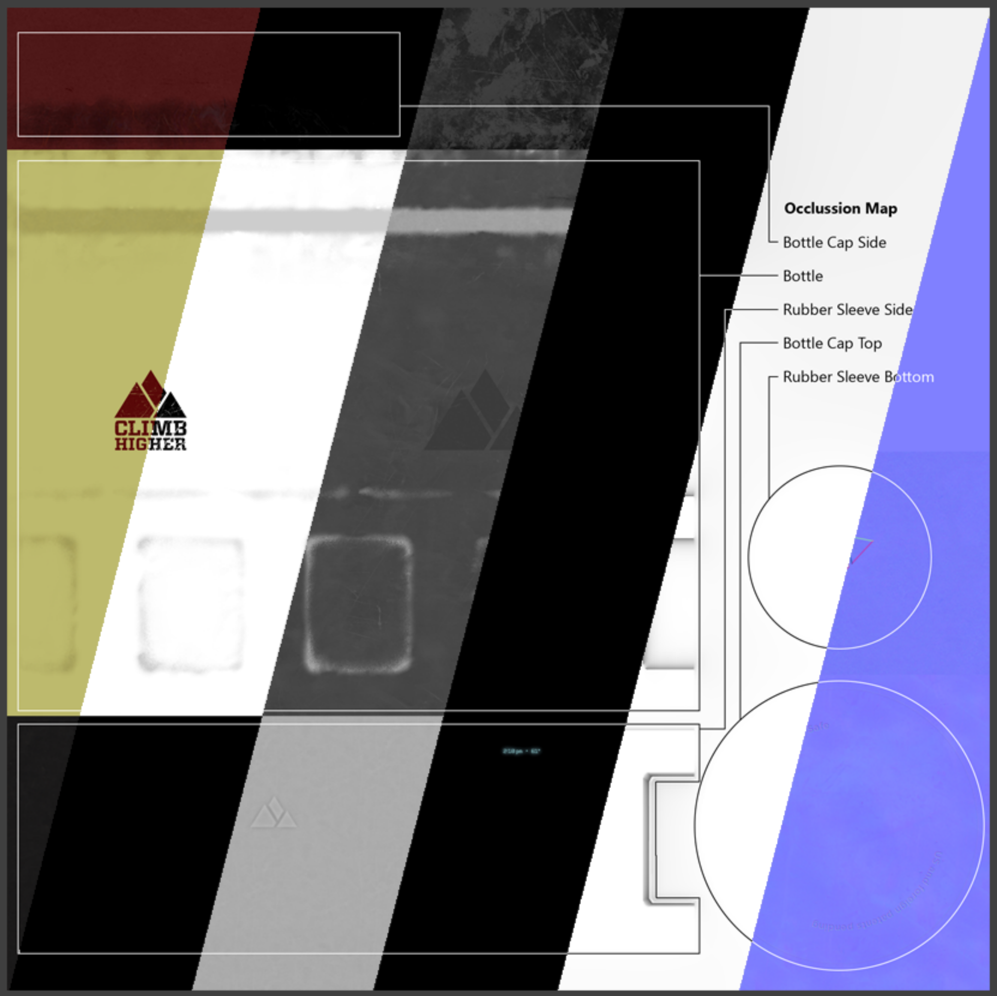
Emissive Map



Base Color

Metal

Rough



Occlussion Map

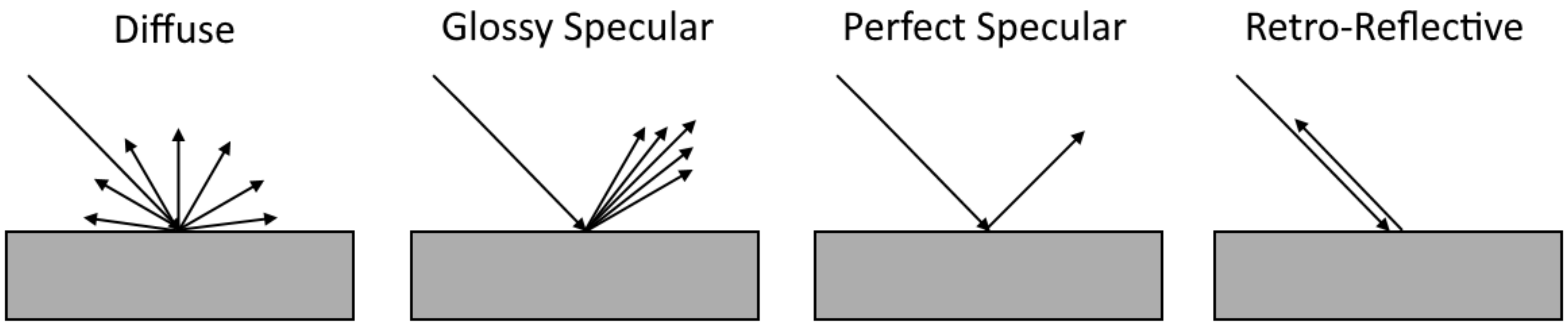
- Bottle Cap Side
- Bottle
- Rubber Sleeve Side
- Bottle Cap Top
- Rubber Sleeve Bottom

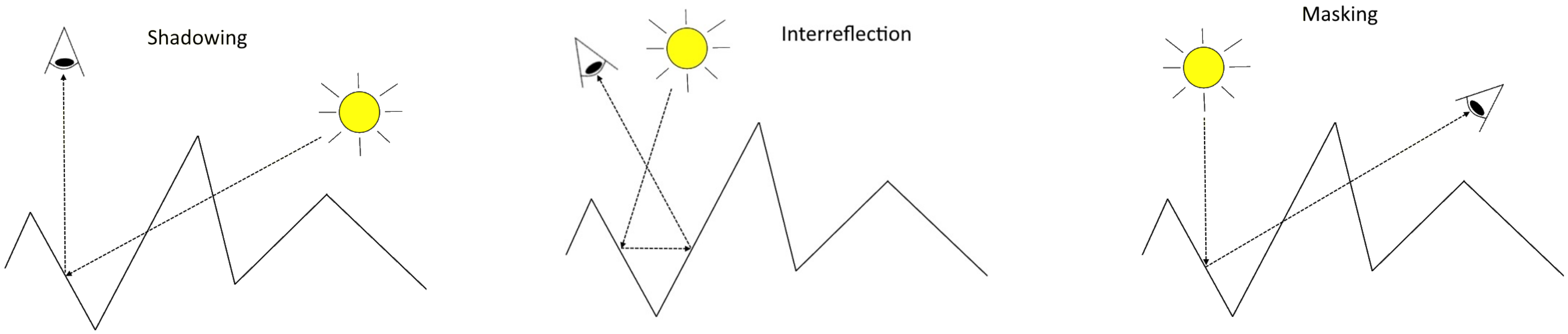
Emissive

Occlusion

Normal

Intro to PBR





https://github.com/moneimne/glTF-Tutorials/tree/master/PBR

BRDF Lighting Equation

$$f(l,v,h) = \underbrace{Diff(l,n)}_{\text{BRDF Diffuse}} + \underbrace{\frac{F(l,h)\,G(l,v,h)\,D\,(h)}{4(n*l)(n\ *v)}}_{\text{BRDF Specular}}$$

- l is the light direction
- v is the view direction
- h is the half vector
- n is the normal

The metallic-roughness material model is defined by the following properties:

- baseColor - The base color of the material
- metallic - The metalness of the material
- roughness - The roughness of the material

BRDF Diffuse

$$Diff(l,n) = (1\ - F(v * h))\frac{C_{diff}}{\pi}$$

Lambertian with energy conservation

C_{diff} is the diffuse reflected color. To conserve energy, the Fresnel term from specular component is subtracted from diffuse component.

```
const dielectricSpecular = rgb(0.04, 0.04, 0.04)
const black = rgb(0, 0, 0)
```

$C_{diff} =$ `lerp(baseColor.rgb * (1 - dielectricSpecular.r), black, metallic)`

BRDF Specular

$$f(l,v,h) = Diff(l,n) + \underbrace{\frac{F(l,h)\,G(l,v,h)\,D\,(h)}{4(n * l)(n\ * v)}}_{\text{BRDF Specular from Cook-Torrance}}$$

BRDF Specular : F

$$\frac{\boldsymbol{F}(\boldsymbol{l},\boldsymbol{h})\; G(l,v,h)\; D\;(h)}{4(n*l)(n\;\;*v)}$$

F is the Fresnel function used to simulate the way light interacts with a surface at different viewing angles.

$$F(l,h) = F_0 + (1 - F_0) * (1 - v * h)^5$$

Schlick Fernel model

F_0 is the specular reflectance at normal incidence

```
const dielectricSpecular = rgb(0.04, 0.04, 0.04)
```

$F_0 =$ `lerp(dieletricSpecular, baseColor.rgb, metallic)`

BRDF Specular : G

$$\frac{F(l,h)\,\boldsymbol{G(l,v,h)}\,D\,(h)}{4(n*l)(n\ *v)}$$

G is the geometric occlusion derived from a normal distribution function like Smith's function

$$G(l,v,h) = G_1(n,l)G_1(n,v)$$

$$G_1(n,v) = \frac{2\,(n*v)}{(n*v) + \sqrt{\alpha^2 + (1-\alpha^2)(n*v)^2}}$$

$$\alpha = \ (roughness)^2$$

BRDF Specular : D

$$\frac{F(l,h)\,G(l,v,h)\,\boldsymbol{D}\,(\boldsymbol{h})}{4(n*l)(n\ *v)}$$

D is the normal distribution function like GGX that defines the statistical distribution of microfacets.

$$\mathrm{D}(h) = \frac{\alpha^2}{\pi\,((n*h)^2(\alpha-1)+1)^2}$$

$$\alpha = \ (roughness)^2$$

Resources

- Demos
  - WebGL-PBR implementation https://github.com/moneimne/WebGL-PBR
  - glTF 2.0 Sample Models: https://github.com/KhronosGroup/glTF-Sample-Models

- Articles
  - glTF PBR Tutorial: https://github.com/moneimne/glTF-Tutorials/tree/master/PBR
  - Substance PBR-guide: https://www.allegorithmic.com/pbr-guide
  - Moving Frostbite to PBR: http://www.frostbite.com/2014/11/moving-frostbite-to-pbr/
  - Good example values: https://seblagarde.wordpress.com/2014/04/14/dontnod-physically-based-rendering-chart-for-unreal-engine-4/

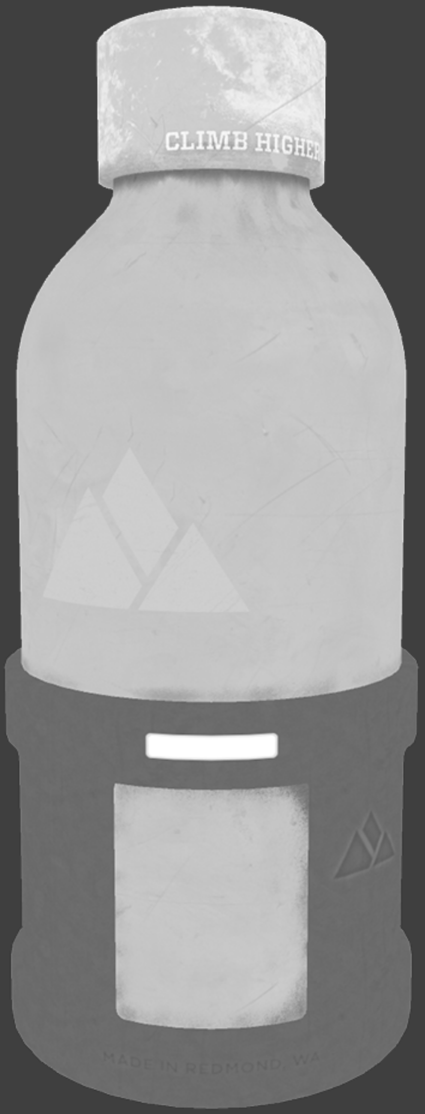
Calls to Action

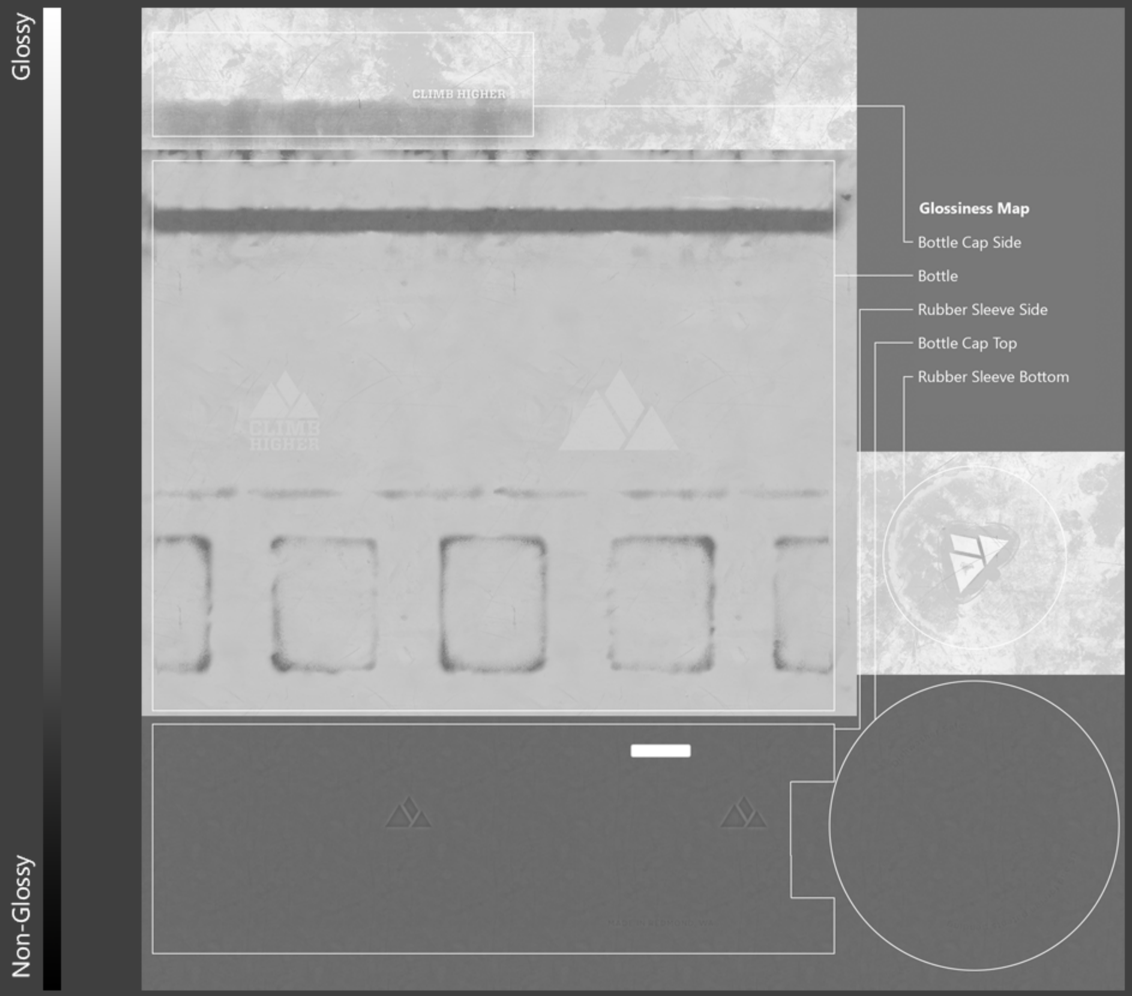
- **Implement glTF 2.0**
  - glTF 2.0 spec: https://github.com/KhronosGroup/glTF/tree/2.0/specification/2.0
- **Open Source Projects**
  - **https://github.com/KhronosGroup/glTF/issues/867**
- **Blender exporter**
  - https://github.com/KhronosGroup/glTF-Blender-Exporter
- **glTF Online Resources**
  - Github Page https://github.com/KhronosGroup/glTF
  - Resource Hub https://www.khronos.org/gltf/
- **Join Khronos!**
  - Get directly involved in the glTF Working Group

Bonus

Specular Glossiness

PBR Materials

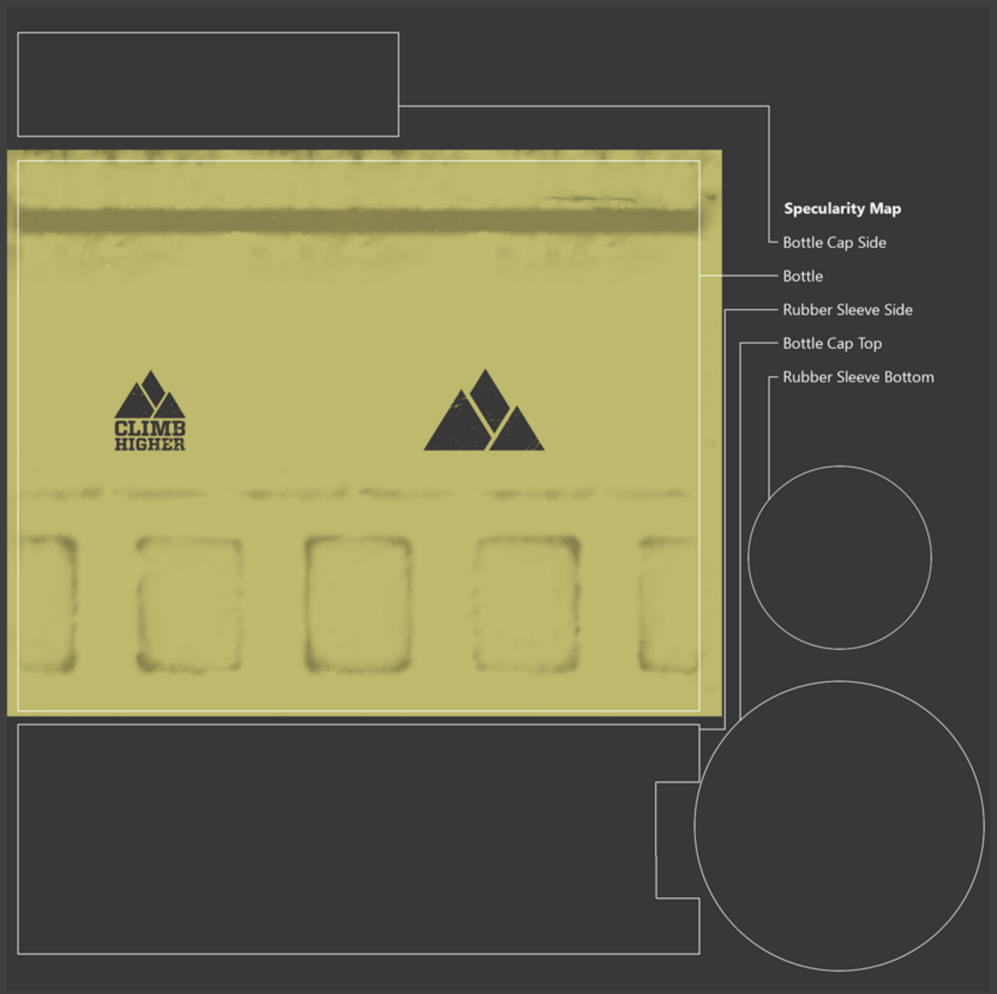




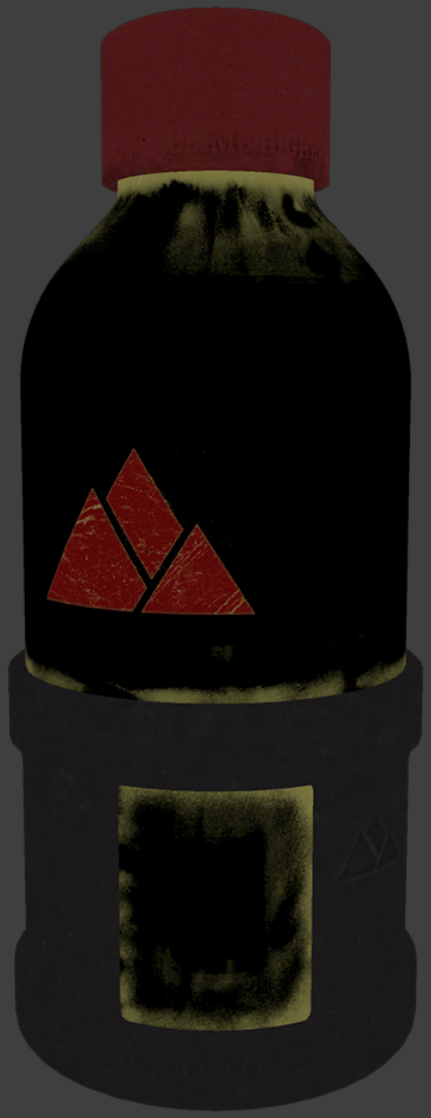
Glossiness Map

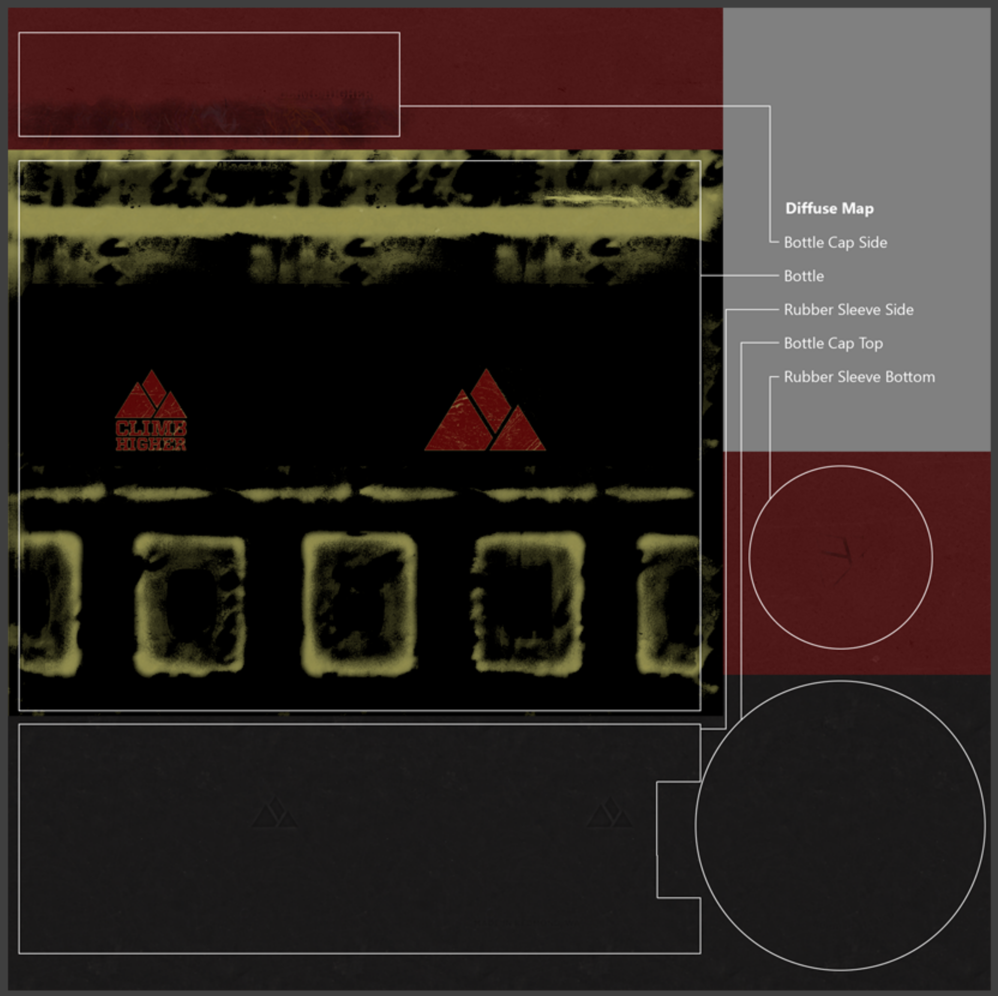
- Bottle Cap Side
- Bottle
- Rubber Sleeve Side
- Bottle Cap Top
- Rubber Sleeve Bottom

Glossiness Map



Specularity Map

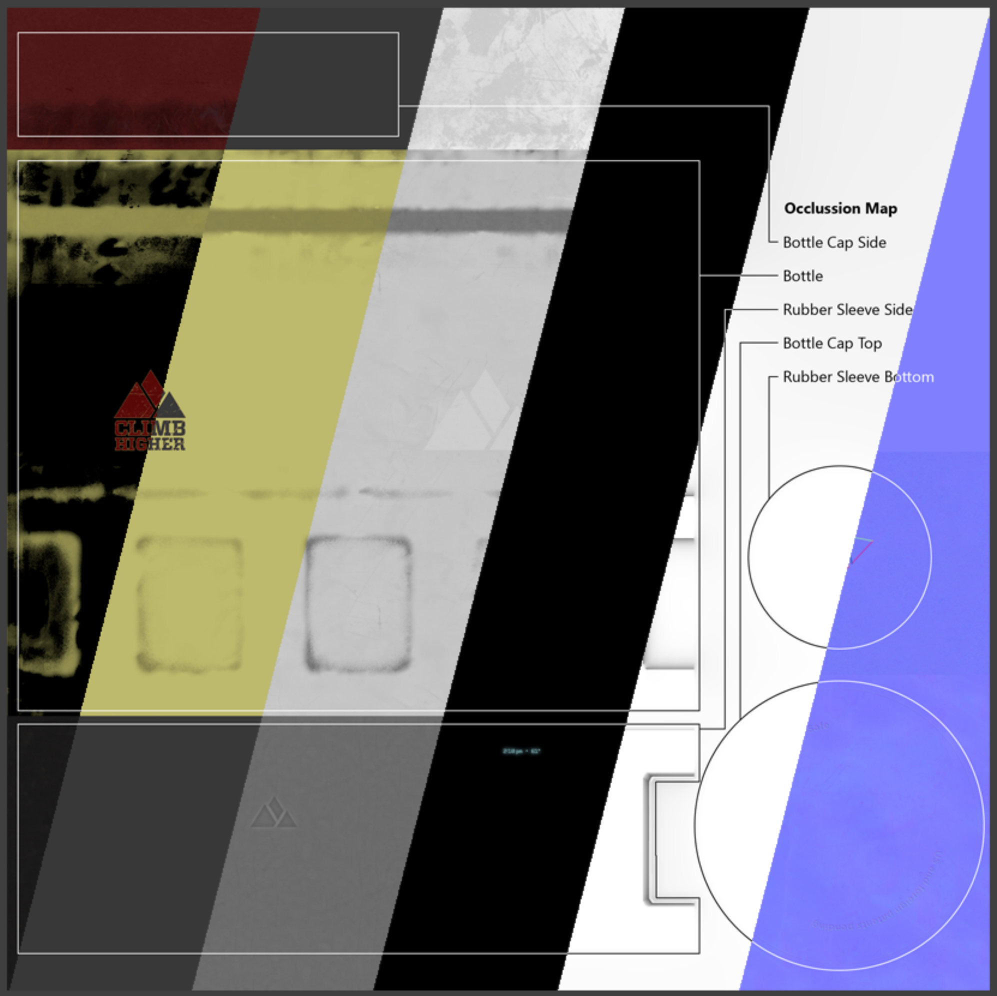



CLIMB
HIGHER
Diffuse Map
Bottle Cap Side
Bottle
Rubber Sleeve Side
Bottle Cap Top
Rubber Sleeve Bottom

Diffuse Map

CLIMB HIGHER
2:10 pm · 61°
MADE IN REDMOND, WA

Diffuse Specular Gloss



Occlussion Map
- Bottle Cap Side
- Bottle
- Rubber Sleeve Side
- Bottle Cap Top
- Rubber Sleeve Bottom

Emissive Occlusion Normal