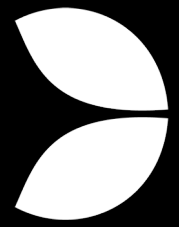
Evolution Engineering

Rat race continous

11.04.2022 Uldis Baurovskis, Armands Baurovskis

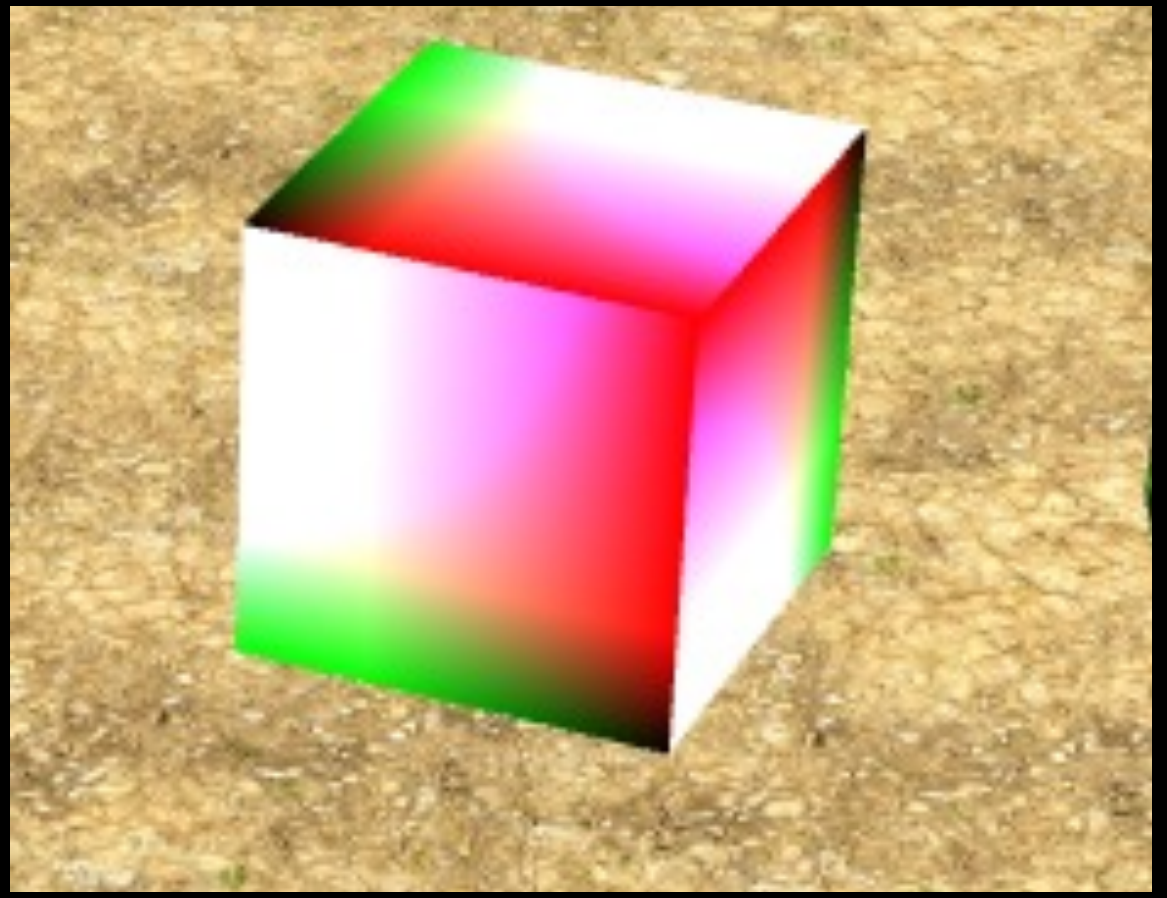
Agenda



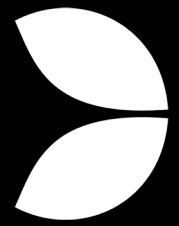
- Power up
- Shaders
- Create custom shader
- Shadow
- Decals
- Normal mapping
- Clone & Instances
- BabylonJS Inspector

Power up

- Boost up player speed
- Made from:
  - Cube as mesh
  - Custom material
  - Intersection for collision



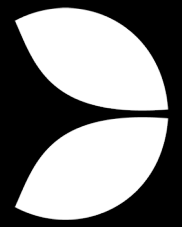
Power up (cont.)



- Shader material:
  - Base for custom materials
  - Allow to define own shader programs

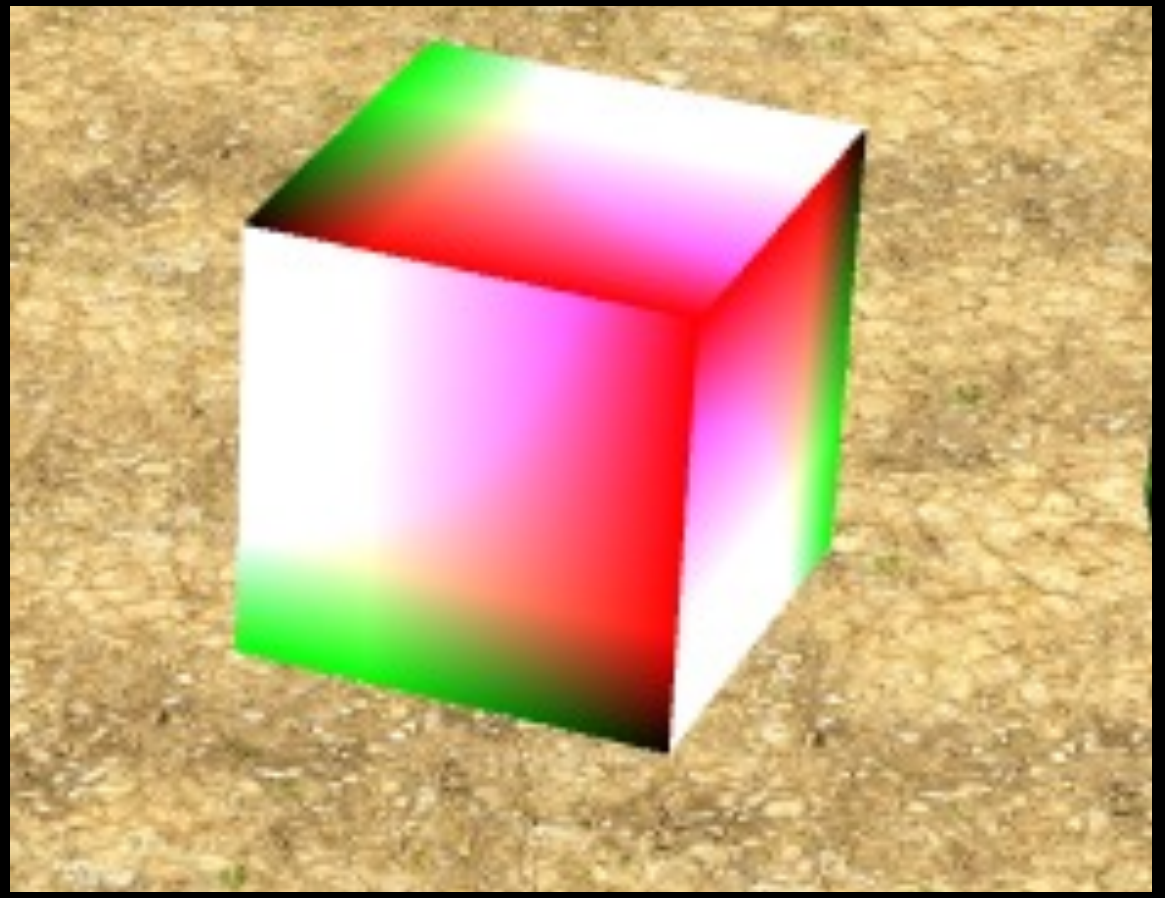
https://doc.babylonjs.com/advanced_topics/shaders/shaderMaterial

Power up (cont.)



- Shader parts:
  - Vertex shader program
  - Fragment shader program
  - Data types
  - Uniforms and attributes

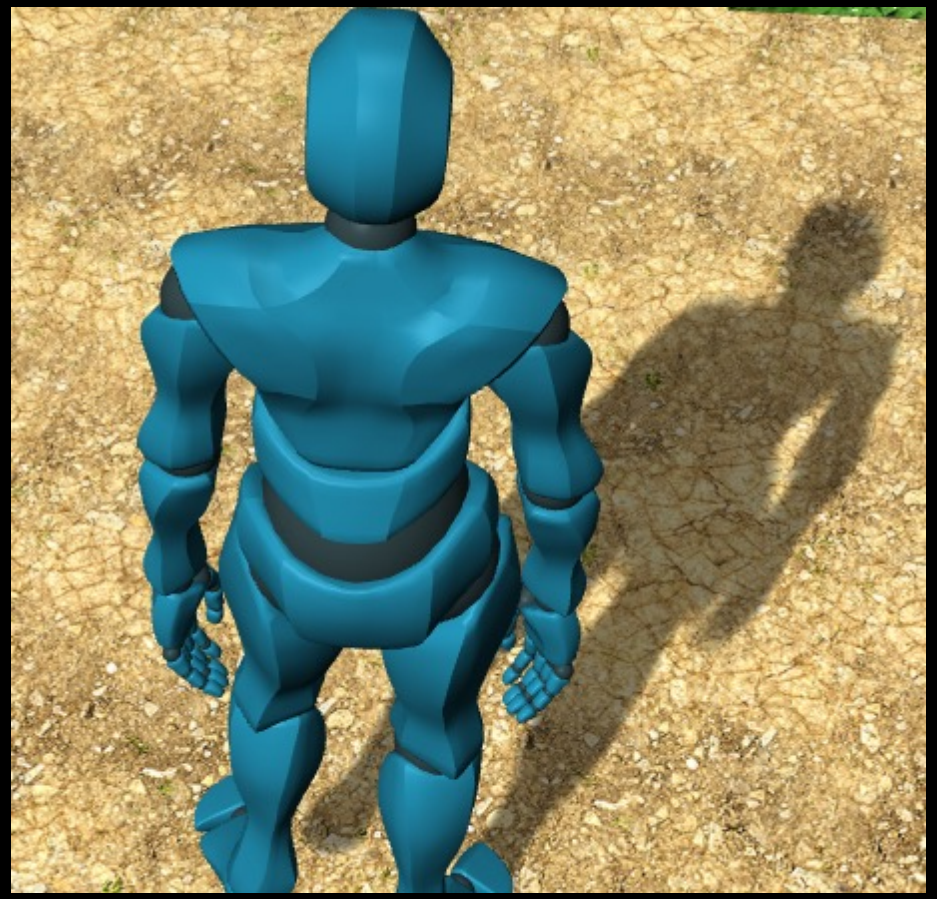
Power up (cont.)



- Intersection:
  - Way to check if mesh collides with other one
  - Need to be checked on some event (like enter frame)
  - Have 2 modes full check and bounding box check

https://doc.babylonjs.com/divingDeeper/mesh/interactions/mesh_intersect

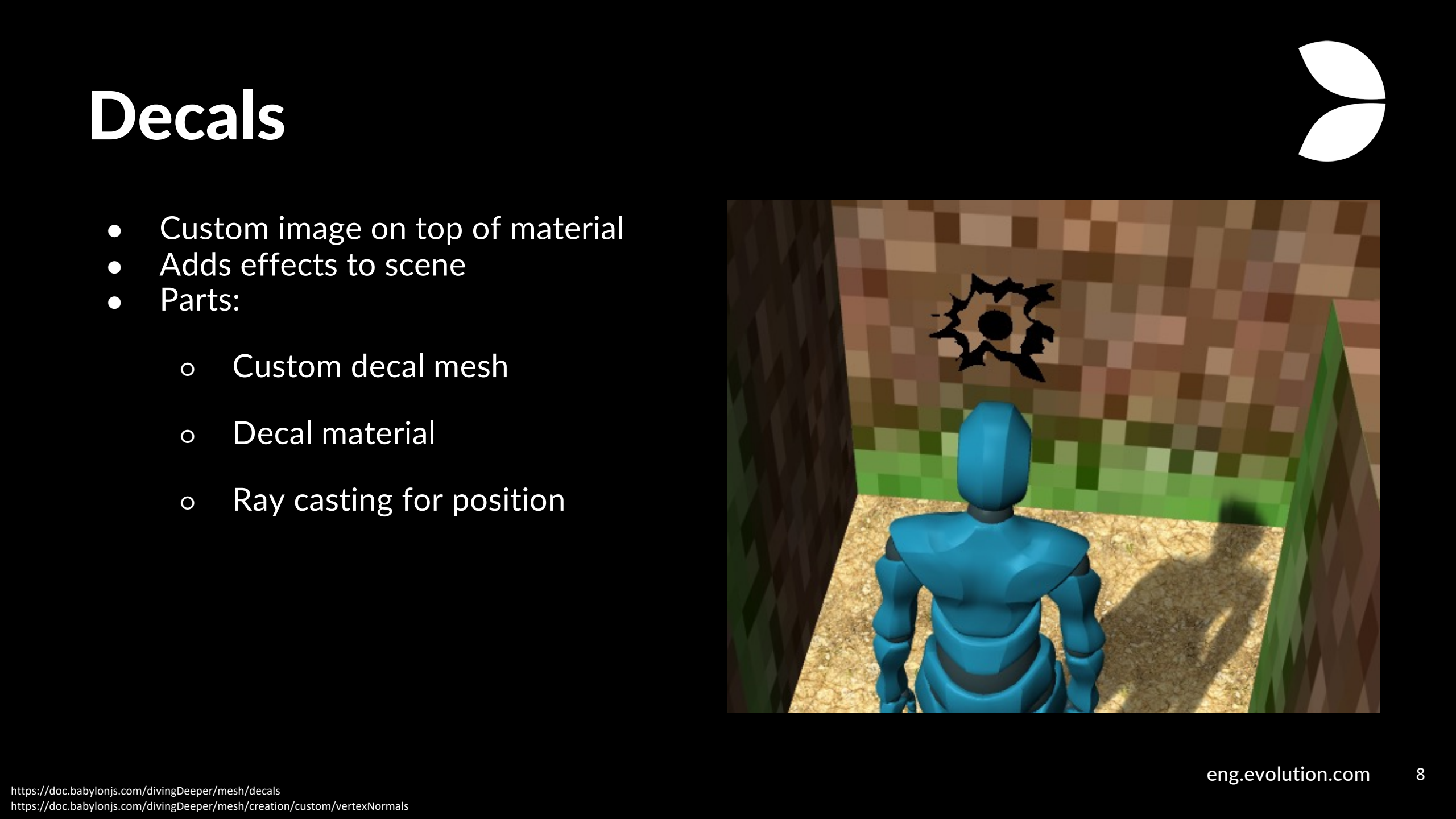
Shadows



- Add shadows for player
- On walls
- Parts:
  - Light that can emit shadows
  - List of shadow receivers
  - List of shadow casters
- Shadow filters

https://doc.babylonjs.com/divingDeeper/lights/shadows

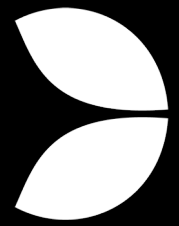
Decals



- Custom image on top of material
- Adds effects to scene
- Parts:
  - Custom decal mesh
  - Decal material
  - Ray casting for position

https://doc.babylonjs.com/divingDeeper/mesh/decals
https://doc.babylonjs.com/divingDeeper/mesh/creation/custom/vertexNormals

Decals (cont.)



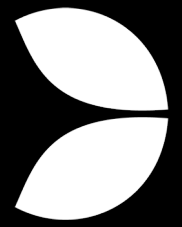
- Create decal mesh that will be added on top of other mesh

- size – size of decal mesh
- position – position on mesh where decal will be placed
- normal – normals used to deform decal, is necessary to place decal on complex meshes

```
const decal = MeshBuilder.CreateDecal(
        “decalName”
        targetMesh,
        {
          position: position,
          normal: normals,
          size: size
        }
);
```


https://doc.babylonjs.com/divingDeeper/mesh/decals

Decals (cont.)

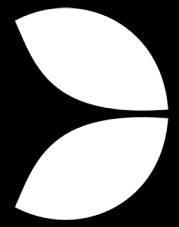


- Need to find a wall using ray casting
- To create a ray, need 3 parts:
  - Origin point
  - Direction vector
  - Distance

```
const decal = MeshBuilder.CreateDecal(
        “decalName”
        targetMesh,
        {
          position: position,
          normal: normals,
          size: size
        }
);
```


https://doc.babylonjs.com/typedoc/classes/babylon.ray

Decals (cont.)



- Use hit test to check if ray intersects with mesh

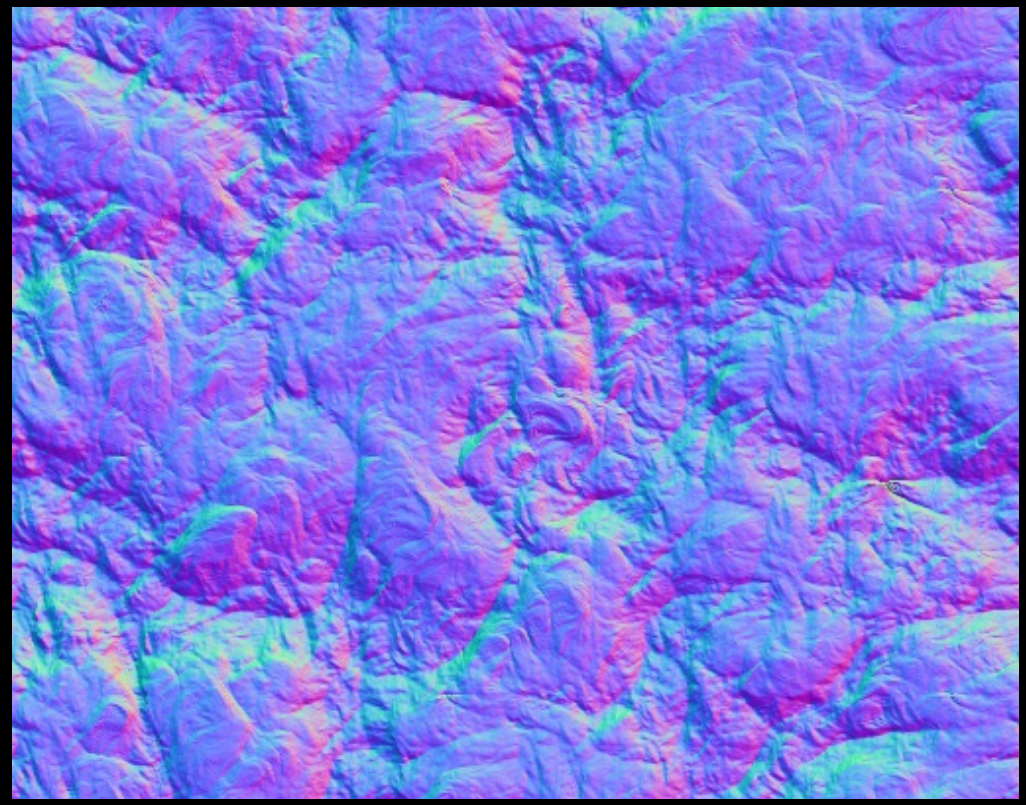
```
const ray = new Ray(origin, direction, rayDistance);

const hit = this.getScene().pickWithRay(ray, mesh => {
   return mesh.name.startsWith("Wall");
});
```


https://doc.babylonjs.com/divingDeeper/mesh/interactions/picking_collisions

Normal mapping

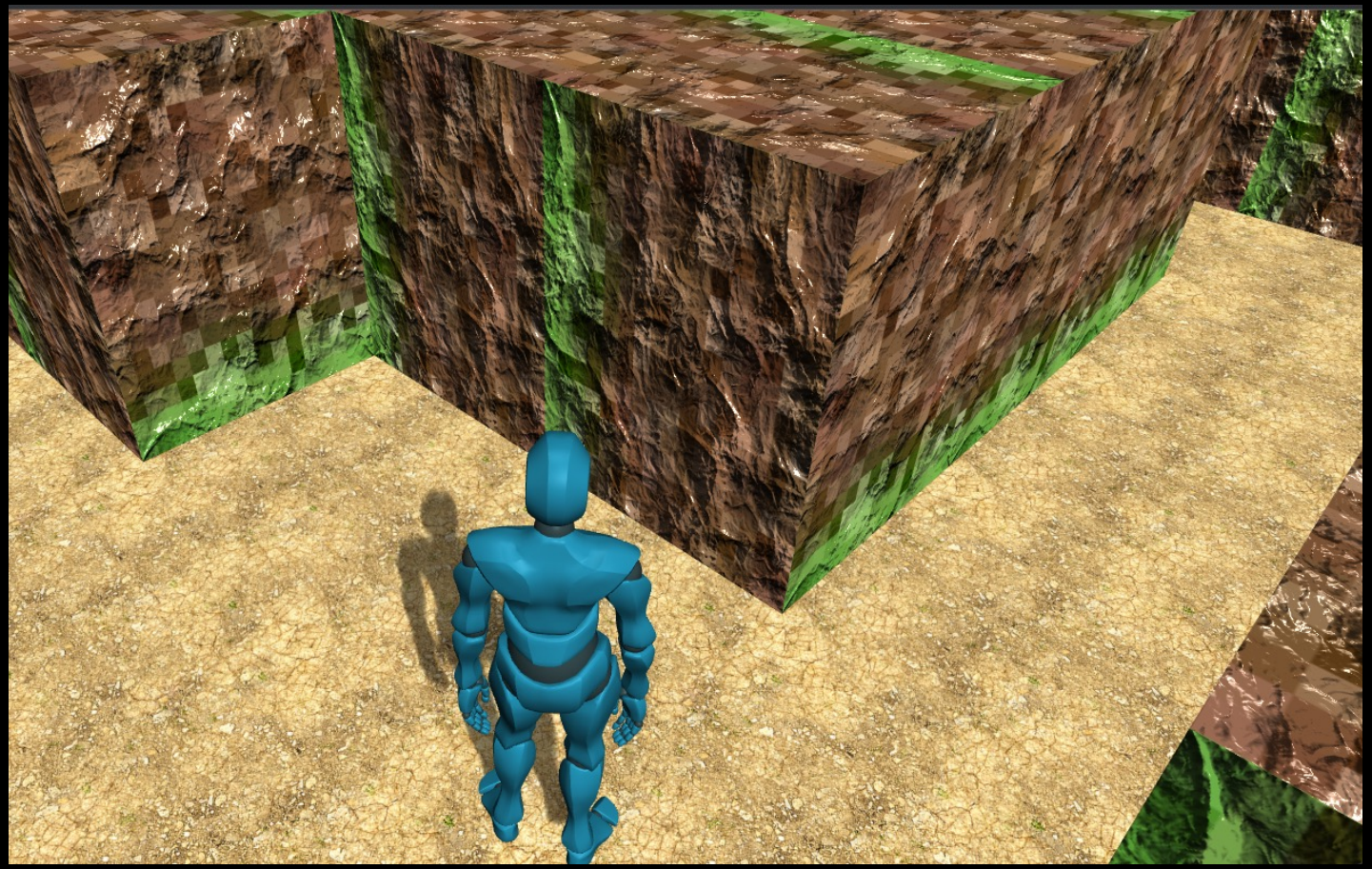
- Texture mapping technique for faking bumps and dents

```
material.bumpTexture =
scene.getTextureByName(TextureId.NormalMap);
```



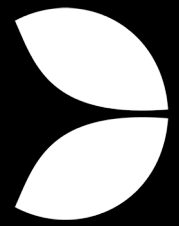
https://doc.babylonjs.com/divingDeeper/materials/advanced/normalMaps
https://learnopengl.com/Advanced-Lighting/Normal-Mapping

Normal mapping (cont.)



https://doc.babylonjs.com/divingDeeper/materials/advanced/normalMaps

Clone & Instances



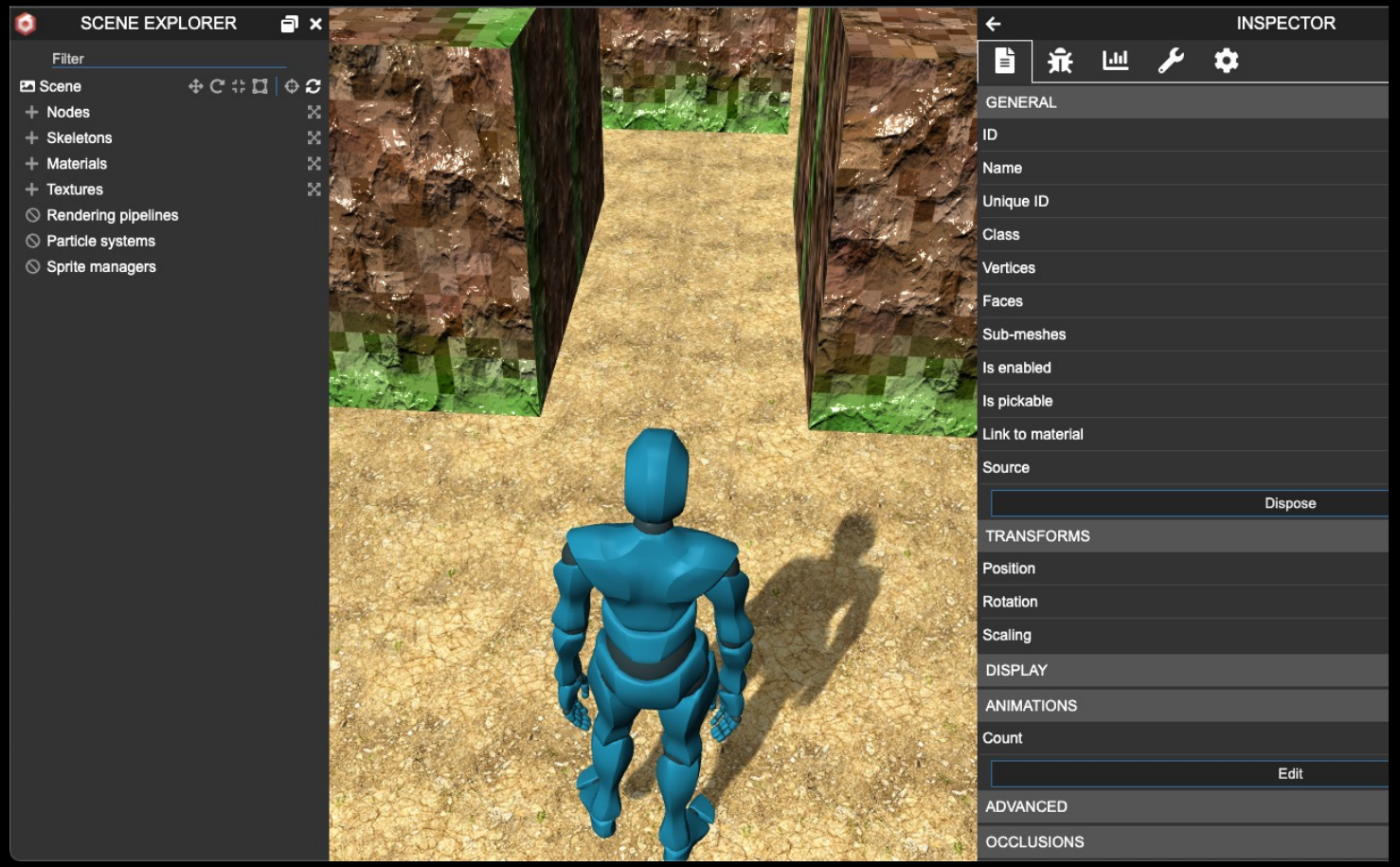
- Resource optimization techniques
- Instances:
    - mirrored meshes that can be rendered all together, but share same material so visually are the same
- Clones:
    - Share only geometry so can have individual material per mesh

https://doc.babylonjs.com/divingDeeper/mesh/copies/instances

BabylonJS Inspector

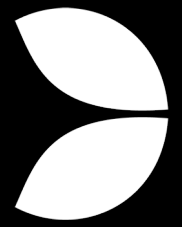
- How to use
- What features contains
- How to debug

- `this.scene.debugLayer.show();`



https://doc.babylonjs.com/toolsAndResources/tools/inspector

Homework



- Try out concepts learend in lecture
- Update game from day 2 with new materials
- Add own powere ups
- Share video / images in slack channel

Evolution Engineering

Q&A