Client-side function

EngineApplyShaderToWorldTexture

This function applies a shader to one or more world textures.

> **Tip:**
>
> - The resource shader_tex_names can help in finding the names of world textures.
> - When replacing the texture for a ped using the CJ skin, set **textureName** to "CJ"
> - The shader inherits the render states of the original when it is drawn, so texture stage 0 will already be set to the original texture.
> - When using with a 'ped', ensure you have set 'ped' or 'all' in the elementTypes when calling dxCreateShader
> - CJ body parts textures can be replaced by using: "cj_ped_head", "cj_ped_hat", "cj_ped_torso", "cj_ped_legs", "cj_ped_feet", "cj_ped_glasses", "cj_ped_necklace", "cj_ped_watch" and "cj_ped_extra1". Latest version of shader_tex_names will show what is being used.

Syntax

```
bool engineApplyShaderToWorldTexture ( element shader, string textureName [, element targetElement = nil, bool appendLayers =
 true ] )
```

OOP Syntax Help! I don't understand this!

> **Method**: *shader:applyToWorldTexture(...)*

Required Arguments

- **shader:** The shader which is to be applied
- **textureName:** The name of the world texture to apply the shader to. Wildcard matching e.g. "ro?ds*" can be used to apply to more than one texture at a time.

Optional Arguments

- **targetElement:** The element to restrict applying the shader to. If this is not set the shader will be applied to everything using the texture name. Valid element types for targetElement are vehicles, objects and peds.
- **appendLayers:** allows two or more layered shaders to be applied in the same texture. You may want to modify the *DepthBias* in the technique pass to avoid Z-fighting artifacts when using this.

Returns

Returns *true* if the shader was successfully applied, *false* otherwise.