Framebuffers

- the *framebuffer* contains the memory resources needed to render an image
  - color buffer holds color data
  - depth buffer holds depth values
  - stencil buffer can be used to limit the area of rendering (potentially in conjunction with the depth buffer)
- the *default framebuffer* is used for the image that appears on the screen
  - associated with the graphics context
- additional framebuffers can be created and used for offscreen rendering
  - e.g. to render to a texture

Renderbuffers

- *renderbuffers* are memory that can be attached to a framebuffer for use as a color buffer, depth buffer, or stencil buffer
- for render-to-texture, the texture object is used as the color buffer
  - an additional renderbuffer is needed for the depth buffer if rendering texture from a 3D scene
  - (depth buffer is not needed for rendering 2D scenes)

Render to Texture

The idea –

- create a texture object, but without specifying image data
- create a framebuffer to render to
- attach the texture to the framebuffer as a color buffer
- create an additional renderbuffer for use as a depth buffer
- attach that renderbuffer to the framebuffer as a depth buffer
- draw to the framebuffer
- draw the scene using the generated texture

done in `initGL`

if depth buffer is needed

done in `draw`

Render to Texture

- create a texture object, but without specifying image data

```
texture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, texture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512,
                          0, gl.RGBA, gl.UNSIGNED_BYTE, null);
```

`texture` is the texture object (javascript variable)

dimensions (must be a power of 2)

`null` to not specify the image data (this will come from a framebuffer)

texture target is `gl.TEXTURE_2D` because we are making a single texture image

other parameters are the same as previously discussed

Render to Texture

- create a framebuffer to render to

```
framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER, framebuffer);
```

create the framebuffer (`framebuffer` is a javascript variable)

set the current framebuffer for subsequent operations

- attach the texture to the framebuffer as a color buffer
  - the desired framebuffer must be current (bound)

```
gl.framebufferTexture2D( gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0,
                         gl.TEXTURE_2D, texture, 0 );
```

attaches the texture object as the color buffer of the framebuffer

standard WebGL only supports a single color buffer, so always `gl.COLOR_ATTACHMENT0`

texture target (matches `gl.texImage2D`)

texture object

mipmap level (0 = render to texture image instead of one of its mipmap levels)

Render to Texture

If rendering a 3D scene for the texture –

- create an additional renderbuffer for use as a depth buffer

```
var depthBuffer = gl.createRenderbuffer();
gl.bindRenderbuffer(gl.RENDERBUFFER, depthBuffer);
gl.renderbufferStorage(gl.RENDERBUFFER, gl.DEPTH_COMPONENT16, 512, 512);
```

allocate memory

bind, to set the current buffer for subsequent operations

set up the renderbuffer

`gl.DEPTH_COMPONENT16` specifies that it will be used as a depth buffer with 16 bits per pixel

size must match the framebuffer size

- attach that renderbuffer to the framebuffer as a depth buffer
  - the desired framebuffer must be current (bound)

```
gl.framebufferRenderbuffer(gl.FRAMEBUFFER, gl.DEPTH_ATTACHMENT,
                           gl.RENDERBUFFER, renderbuffer);
```

attach renderbuffer as a depth buffer

`renderbuffer` is the variable for the buffer

Render to Texture

- draw to the framebuffer
  - rendering a 2D scene

```
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.useProgram(prog_texture); // shader program for the texture

gl.clearColor(1,1,1,1);
gl.clear(gl.COLOR_BUFFER_BIT);

gl.disable(gl.DEPTH_TEST); // framebuffer doesn't even have a depth buffer!
gl.viewport(0,0,512,512); // Viewport is not set automatically!

  .
  . // draw the texture image, which changes in each frame
  .
```

make the texture framebuffer the current one (`framebuffer` is the javascript variable for this framebuffer)

if using different shader programs for rendering texture vs rest of scene

set background color as desired

depth buffer not used for 2D drawing

viewport is not set automatically for other framebuffers (size must match size specified in `gl.texImage2D`)

  - for rendering a 3D scene, also clear `gl.DEPTH_BUFFER_BIT` and do not disable `gl.DEPTH_TEST`

Render to Texture

- draw the scene using the generated texture

```
gl.bindFramebuffer(gl.FRAMEBUFFER,null); // Draw to default framebuffer.
gl.useProgram(prog);  // shader program for the on-screen image
gl.clearColor(0,0,0,1);
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
gl.enable(gl.DEPTH_TEST);
gl.viewport(0,0,canvas.width,canvas.height);  // Reset the viewport!

.
.  // draw the scene
.   (use the texture object as usual)
```

set current framebuffer to the default framebuffer (for drawing to the display)

if using different shader programs for rendering texture vs rest of scene

set background color as desired

if depth test was previously disabled (for rendering 2D scene)

must restore the viewport manually

canvas is the javascript variable referring to the canvas on the web page