Cheatography

# WebGL-BasicFunctions Cheat Sheet

by luzvare via cheatography.com/121842/cs/22441/

## Buffer Objects

**Creates a new buffer object:** Object createBuffer() Note: Corresponding OpenGL ES function is GenBuffers

**Binds an empty array buffer to the object:** void bindBuffer(enum target, Object buffer) target: ARRAY_BUFFER, ELEMENT_ARRAY_BUFFER

**Assings the passed values to the target:** void bufferData(enum target, Object data, enum usage) target and usage: Same as for bufferData above

## Uniforms and Attributes

**Index of the attribute**ulong getAttribLocation(Object program, string name)

**Index of the uniform variable:** uint getUniformLocation(Object program, string name)

**Passes the information from buffer to attribute:**void vertexAttribPointer(uint index, int size, enum type, bool normalized, long stride, long offset) type: BYTE, SHORT, UNSIGNED_{BYTE, SHORT}, FIXED, FLOAT index: [0, MAX_VERTEX_ATTRIBS - 1] stride: [0, 255] offset, stride: must be a multiple of the type size in WebGL

## Storage Qualifiers

**attribute** : Linkage between a vertex shader and OpenGL ES for per-vertex data

**uniform :** Value does not change across the primitive being processed, uniforms form the linkage between a shader, OpenGL ES, and the application

**varying :** Linkage between a vertex shader and fragment shader for interpolated data

## Programs and shaders

**Creates a shader object:** Object createShader(enum type) type: VERTEX_SHADER, FRAGMENT_SHADER

**Creates a program:**Object createProgram()

**Assigns a source program to the object:** ** void shaderSource(Object shader, string source)

**Attach the shader to the program:**void attachShader(Object program, Object shader)

**Compiles the shader:** void compileShader(Object shader)

**Links the attached shaders:**void linkProgram(Object program)

**Uses the combined shaders:**void useProgram(Object program)

## Writing to the draw buffer

**Draws the given array:** void drawArrays(enum mode, int first, long count) mode: POINTS, LINE_STRIP, LINE_LOOP, LINES, TRIANGLE_STRIP, TRIANGLE_FAN, TRIANGLES first: May not be a negative value.

**Draws the passed element** void drawElements(enum mode, long count, enum type, long offset) mode: POINTS, LINE_STRIP, LINE_LOOP, LINES, TRIANGLE_STRIP, TRIANGLE_FAN, TRIANGLES type: UNSIGNED_BYTE, UNSIGNED_SHORT

## Precision

**highp :** Satisfies minimum requirements for the vertex language. Optional in the fragment language.

**mediump:** Satisfies minimum requirements for the fragment language. Its range and precision is between that provided by lowp and highp.

**lowp:** Range and precision can be less than mediump, but still represents all color values for any color channel.