# Glossary

## Compute or Graphic Memory Allocated Object

A compute or graphic memory allocated object is an OpenCL or OpenGL object created by a computer application, which is allocated on the memory of the OpenCL or OpenGL device.

An allocated object can reside in the computer's main memory (as part of the driver's used virtual memory), in the device's internal memory, or in both. In multi-device OpenCL contexts and multi-renderer OpenGL contexts, an object's data can even reside in multiple devices at the same time. CodeXL currently supports the following allocated object types.

* OpenCL compute contexts
* OpenCL command queues
* OpenCL command events\*
* OpenCL Image objects
* OpenCL buffer objects
* OpenCL computation programs
* OpenCL kernel objects\*
* OpenCL sampler objects\*
* OpenGL render contexts
* OpenGL render contexts' static buffers
* OpenGL texture objects
* OpenGL render buffers
* OpenGL framebuffer objects (FBOs)\*
* OpenGL vertex buffer objects (VBOs)
* OpenGL shading programs\*
* OpenGL shaders
* OpenGL display lists
* OpenGL pixel buffer objects (PBuffers)\*

The memory consumption of these objects is insignificant, but their creation and deletion is still monitored by CodeXL.

**Function Types:** OpenCL functions can be divided into groups by their effect on the OpenCL implementation or by their outputs. Note that an OpenCL function can belong to more than one of these groups, or to none at all:

* Get Functions retrieve information from the OpenCL ICD.
* Buffer and Image Functions are functions related to buffer and image objects.
* Program and Kernel Functions are related to computation programs and kernels.
* Queue Functions are related to command queue objects.
* Synchronization Functions perform synchronization operations.

OpenGL functions can be grouped by their effect on the OpenGL implementation or by their outputs. Note that an OpenGL function can belong to more than one of these groups, or to none at all:

* Get Functions retrieve information from the OpenGL driver.
* State Change Functions change the values of OpenGL State Variables.
* Draw Functions can have a visible effect on the draw buffer.
* Raster Functions copy pixels from or to frame buffers.
* Programs and Shaders Functions are related to shaders and shading programs.
* Texture Functions create, delete, or manipulate textures.
* Matrix Functions are related to the matrix stacks (modelview matrix, projection matrix, color matrix, etc.).
* Name Functions are related to the OpenGL selection mode.
* Query Functions are related to OpenGL query items.
* Buffer Functions are related to OpenGL buffer objects.

To find out what groups a given function belongs to, look for its entry in the Function Calls Statistics tab (in the CodeXL Statistics view). The information is displayed under the "Function Type" column. The Statistics view's Function Types Statistics tab shows details of your API usage by these groups, as well as more detailed information about the use of each function type.

## Frames and Frame Terminators

**Computation Frame:** A computation frame is a set of OpenCL API calls, typically the largest set of calls an OpenCL compute context performs that can be considered a single logical operation. A computation frame is comparable to a render frame in a graphics engine. It is recommended for applications rendering graphics to define the computation frame as the set of commands used to calculate the data for a single render frame. Having a notion and boundary of what comprises a frame permits measurements such as frame times and frame rates, as well as API call statistics, which are useful in debugging and profiling.

**Frame Terminator:** Frame terminators are the functions that end your application computation and render frames. CodeXL uses frame terminators for frame-per-second calculations, statistics analysis, and other measurements, as well as for determining when to stop the execution after you press the Step Out button (Shift+F11). When choosing frame terminators for your project, ensure that at least one of the functions you choose is called for each frame you render. If you are using the OpenGL debug engine, you must select at least one render (OpenGL) frame terminator. A typical selection of Frame Terminators is clFinish for OpenCL and the SwapBuffers option for OpenGL.

## Vertex Batch

**Vertex Batch:** A vertex batch is a group of vertices sent to the graphic driver in a single API call. In OpenGL, this effectively means the number of vertices sent to the graphic pipeline with a single function call. For example, the function glVertex3f always generates a 1-vertex batch, while glDrawElements creates a batch of the size determined by its count parameter value. The number of vertex batches sent to the driver represents the **cost** of drawing, while the batch size is the **benefit** of each function call. Drawing 200 1-vertex batches has the same benefit as drawing a single 200-vertex batch, for 200 times the cost. For this reason, it is best to draw as many vertices using as few batches as possible.