An event is an object that communicates the status of OpenCL commands

- **Read Buffer dC**
- **Execute Kernel**
- **Write Buffer dB**
- **Write Buffer dA**

### From the OpenCL Notes:

1. Enqueue the Kernel Object for Execution

```c
size_t globalWorkSize[3] = { NUM_ELEMENTS, 1, 1 };  
size_t localWorkSize[3] = { LOCAL_SIZE, 1, 1 };  
status = clEnqueueNDRangeKernel( cmdQueue, kernel, 1, NULL, globalWorkSize, localWorkSize, 0, NULL, NULL );
```

### Creating an Event

```c
cl_event waitKernelA, waitKernelB, waitKernelC;  
status = clEnqueueNDRangeKernel( cmdQueue, kernel, 1, NULL, globalWorkSize, localWorkSize, 0, NULL, &waitKernelC );
```

### Waiting for Events from Previously-Executed Kernels

```c
cl_event dependenciesAB[2];  
dependenciesAB[0] = waitKernelA;  
dependenciesAB[1] = waitKernelB;  
status = clEnqueueNDRangeKernel( cmdQueue, kernelC, 1, NULL, globalWorkSize, localWorkSize, 2, dependenciesAB, NULL );
```
Creating the Full Execution Graph Structure

Creating the Full Execution Graph Structure

Placing a Barrier in the Command Queue

Placing a Barrier in the Command Queue

Placing an Event Marker in the Command Queue

Placing an Event Marker in the Command Queue

Waiting for Events Without Enqueuing Another Command

Waiting for Events Without Enqueuing Another Command

I Like Synchronizing Things This Way

I Like Synchronizing Things This Way
Getting Event Statuses Without Blocking

CL_EVENT_COMMAND_QUEUE
CL_EVENT_CONTEXT
CL_EVENT_COMMAND_TYPE
CL_EVENT_COMMAND_EXECUTION_STATUS

Specify one of these

cl_int eventStatus;
status = clGetEventInfo( waitKernelC,  CL_EVENT_COMMAND_EXECUTION_STATUS,  sizeof(cl_int),
                       &eventStatus,  NULL  );

Note that this a nice way to check on event statuses without blocking. Thus, you could put
this in a loop and go get some other work done in between calls.