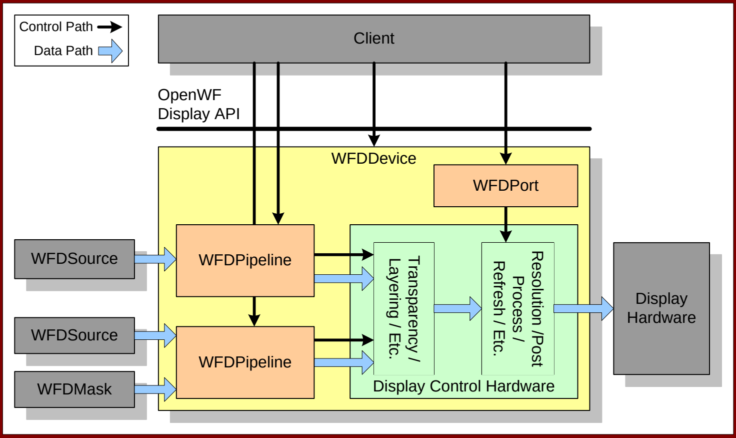
# OpenWF Display 1.0 API Quick Reference Card

**OpenWF Display®** is a standardized API for compositing and output to display. It serves as a low-level interface for two-dimensional composition used in embedded and/or mobile devices. Target users are windowing systems, system integrators etc. The API is implementable on top of a legacy display controller as well as specific hardware.
The header file to include is <WF/wfd.h>

- [n.n.n] refers to the section in the API Specification available at www.khronos.org/openwf/
- Blue are datatypes defined in the WFD spec
- (r/w) – read/writable (r) – read only
- Brown are constant values defined in the WFD spec
- *Italic* are parameter names in function declarations



## Errors [2.11] – of type WFDErrorCode

Error codes and their numerical values are defined by the WFDErrorCode enumeration and could be retrieved by the following function:
WFDErrorCode **wfdGetError**(WFDDevice *device*).

The possible values are as follows:

| | | | |
|---|---|---|---|
| WFD_ERROR_NONE | 0x0000 | WFD_ERROR_OUT_OF_MEMORY | 0x7510 |
| WFD_ERROR_ILLEGAL_ARGUMENT | 0x7511 | WFD_ERROR_NOT_SUPPORTED | 0x7512 |
| WFD_ERROR_BAD_ATTRIBUT E | 0x7513 | WFD_ERROR_IN_USE | 0x7514 |
| WFD_ERROR_BUSY | 0x7515 | WFD_ERROR_BAD_DEVICE | 0x7516 |
| WFD_ERROR_BAD_HANDLE | 0x7517 | WFD_ERROR_INCONSISTENCY | 0x7518 |

Functions that returns handles could return the following error:

**WFD_INVALID_HANDLE** [2.6]

## Device - A WFDDevice [3] is an abstraction of a display controller that supports one or more ports (WFDPort - display abstraction) and zero or more pipelines (a WFDPipeline – manipulates source images).

### Device Creation and Destruction [3.1], [3.2], [3.3]

WFDint **wfdEnumerateDevices**(WFDint **deviceIds*,
WFDint *deviceIdsCount*, const WFDint **filterList*)
Populate a list of available devices with respect to the filter-list (could be WFD_NONE).

WFCDevice **wfdCreateDevice**(WFDint *deviceId*,
const WFDint **attribList*)
Create a device with a known ID - could use WFD_DEFAULT_DEVICE_ID.

WFDErrorCode **wfdDestroyDevice**(WFDDevice *device*)
Delete a specific device.

### Commit modifications [3.4] Modifications are cached until commited.

void **wfdDeviceCommit**(WFDDevice *device*, WFDCommitType *type*,
WFDHandle *handle*)

Handle is a reference to the port or pipeline to commit – or WFD_INVALID_HANDLE when comitting the entire device.

WFDCommitType – scope of the commit call

| | | |
|---|---|---|
| WFD_COMMIT_ENTIRE_DEVICE | 0x7550 | Commit device +attached ports & pipelines |
| WFD_COMMIT_ENTIRE_PORT | 0x7551 | Commit port +attached pipelines |
| WFD_COMMIT_PIPELINE | 0x7552 | Commit only specific pipeline |

## Events [3.6] - events are exposed per device.

WFDEvent **wfdCreateEvent**( WFDDevice *device*, const WFDint **attribList*)
Create an event container needed by a client to receive events (selected by the attribList) from a device. The created event is used in the rest of the event functions. WFD_EVENT_PIPELINE_BIND_QUEUE_SIZE = 0 disables events.

WFDEventAttrib

| | | |
|---|---|---|
| WFD_EVENT_PIPELINE_BIND_QUEUE_SIZE | 0x75C0 | (r/w) |
| WFD_EVENT_TYPE | 0x75C1 | (r) |
| WFD_EVENT_PORT_ATTACH_PORT_ID | 0x75C2 | (r) |
| WFD_EVENT_PORT_ATTACH_STATE | 0x75C3 | (r) |
| WFD_EVENT_PORT_PROTECTION_PORT_ID | 0x75C4 | (r) |
| WFD_EVENT_PIPELINE_BIND_PIPELINE_ID | 0x75C5 | (r) |
| WFD_EVENT_PIPELINE_BIND_SOURCE | 0x75C6 | (r) |
| WFD_EVENT_PIPELINE_BIND_MASK | 0x75C7 | (r) |
| WFD_EVENT_PIPELINE_BIND_QUEUE_OVERFLOW | 0x75C8 | (r) |

void **wfdDestroyEvent**( WFDDevice *device*, WFDEvent *event* )

WFDint **wfdGetEventAttribi**( WFDDevice *device*, WFDEvent *event*,
WFDEventAttrib *attrib* )

void **wfdDeviceEventAsync**( WFDDevice *device*, WFDEvent *event*,
WFDEGLDisplay *display*, WFDEGLSync *sync*)
Add or replace existing event subscription (use WFD_INVALID_SYNC to terminate existing subscription).

WFDEventType

| | |
|---|---|
| WFD_EVENT_INVALID | 0x7580 |
| WFD_EVENT_NONE | 0x7581 |
| WFD_EVENT_DESTROYED | 0x7582 |
| WFD_EVENT_PORT_ATTACH_DETACH | 0x7583 |
| WFD_EVENT_PORT_PROTECTION_FAILURE | 0x7584 |
| WFD_EVENT_PIPELINE_BIND_SOURCE_COMPLETE | 0x7585 |
| WFD_EVENT_PIPELINE_BIND_MASK_COMPLETE | 0x7586 |

WFDEventType **wfdDeviceEventWait**( WFDDevice *device*, WFDEvent *event*,
WFDtime *timeout* )
Blocking wait for event with timeout (could be WFD_FOREVER ).

void **wfdDeviceEventFilter**( WFDDevice *device*, WFDEvent *event*,
const WFDEventType **filter* )
Add a list of enabled events, terminated by WFD_NONE .

### Device Configuration [3.5] – currently only WFD_DEVICE_ID is defined in the spec.

WFDint **wfdGetDeviceAttribi**(WFDDevice *device,*
WFDDeviceAttrib *attrib*)

void **wfdSetDeviceAttribi**( WFDDevice *device*,
WFDDeviceAttrib *attrib*, WFDint *value*)

WFCint **wfcGetDeviceAttribi**( WFCDevice *dev*, WFCDeviceAttrib *attrib*)

# OpenWF Display 1.0 API Quick Reference Card

**Port** - A WFDPort[4] is the output from a WFDDevice (i.e. a display). It could be a CRT, a fixed LCD or an attachable TV for example. The API supports configuration of the display hardware.

WFDint **wfdEnumeratePorts**( WFDDevice *device*, WFDint **portIds*,
  WFDint *portIdsCount*, const WFDint **filterList* )

Retrieve a list of numbers and IDs of available ports of a device. Note that ports with detached display hardware [4.5.1.3] will still be listed and possible to create. If ID = WFD_INVALID_PORT_ID an unfiltered list is returned.

WFDPort **wfdCreatePort**( WFDDevice *device*, WFDint *portId*,
  const WFDint **attribList* )

If ID = WFD_DEFAULT_DEVICE_ID an integration specific default device is returned.

void **wfdDestroyPort**( WFDDevice *device*, WFDPort *port* )

## Port Modes [4.4] – one or more mode supported for attached display hardware

WFDPortModeAttrib [4.4.1]

| | | |
|---|---|---|
| WFD_PORT_MODE_WIDTH | **0x7600** | Resolution in pixels |
| WFD_PORT_MODE_HEIGHT | **0x7601** | Resolution in pixels |
| WFD_PORT_MODE_REFRESH_RATE | **0x7602** | In frames per second |
| WFD_PORT_MODE_FLIP_MIRROR_SUPPORT | **0x7603** | WFD_TRUE or WFD_FALSE |
| WFD_PORT_MODE_ROTATION_SUPPORT | **0x7604** | WFDRotationSupport in port |
| WFD_PORT_MODE_INTERLACED | **0x7605** | WFD_TRUE or WFD_FALSE |

WFDRotationSupport [4.4.1.4]

| | | |
|---|---|---|
| WFD_ROTATION_SUPPORT_NONE | **0x76D0** | No support |
| WFD_ROTATION_SUPPORT_LIMITED | **0x76D1** | 0, 90, 180, 270 degrees supported |

## Get/Set Port Modes & Attributes

WFDint **wfdGetPortModes(** WFDDevice *device*, WFDPort *port*,
  WFDPortMode **modes*, WFDint *modesCount* **)**

WFDPortMode **wfdGetCurrentPortMode(** WFDDevice *device*,
  WFDPort *port* **)**

void **wfdSetPortMode(** WFDDevice *device*, WFDPort *port*,
  WFDPortMode *mode* **)**

WFDint **wfdGetPortModeAttribi(** WFDDevice *device*, WFDPort *port*,
  WFDPortMode *mode*, WFDPortModeAttrib *attrib* **)**

WFDfloat **wfdGetPortModeAttribf(** WFDDevice *device*, WFDPort *port*,
  WFDPortMode *mode*, WFDPortModeAttrib *attrib* **)**

WFDPortConfigAttrib [4.5.1]

| | | |
|---|---|---|
| WFD_PORT_ID | **0x7620** | (r) from **wfdEnumeratePorts** |
| WFD_PORT_TYPE | **0x7621** | (r) WFDPortType |
| WFD_PORT_DETACHABLE | **0x7622** | (r) WFD_TRUE or WFD_FALSE |
| WFD_PORT_ATTACHED | **0x7623** | (r) WFD_TRUE or WFD_FALSE |
| WFD_PORT_NATIVE_RESOLUTION | **0x7624** | (r) array (width, height) in pixels |
| WFD_PORT_PHYSICAL_SIZE | **0x7625** | (r) array (width, height) in mm |
| WFD_PORT_FILL_PORT_AREA | **0x7626** | (r) If WFD_TRUE whole area must be filled |
| WFD_PORT_BACKGROUND_COLOR | **0x7627** | (r/w) (r,g,b) in float (0 - 1) |
| WFD_PORT_FLIP | **0x7628** | (r/w) Dependent of hw support |
| WFD_PORT_MIRROR | **0x7629** | (r/w) Dependent of hw support |
| WFD_PORT_ROTATION | **0x762A** | (r/w) in 90deg values if supported |
| WFD_PORT_POWER_MODE | **0x762B** | (r/w) current powermode |
| WFD_PORT_GAMMA_RANGE | **0x762C** | (r) array (**min**, **max**) |
| WFD_PORT_GAMMA | **0x762D** | (r/w) **min** ≤ value ≤ **max** |
| WFD_PORT_PARTIAL_REFRESH_SUPPORT | **0x762E** | (r) WFDPartialRefresh |
| WFD_PORT_PARTIAL_REFRESH_MAXIMUM | **0x762F** | (r) array (*width*, *height*) |
| WFD_PORT_PARTIAL_REFRESH_ENABLE | **0x7630** | (r/w) WFD_TRUE or WFD_FALSE |
| WFD_PORT_PARTIAL_REFRESH_RECTANGLE | **0x7631** | (r/w) (*offsetX*, *offsetY*, *width*, *height*) |
| WFD_PORT_PIPELINE_ID_COUNT | **0x7632** | (r) Nbr of pipelines |
| WFD_PORT_BINDABLE_PIPELINE_IDS | **0x7633** | (r) List of pipelines |
| WFD_PORT_PROTECTION_ENABLE | **0x7634** | (r/w) WFD_TRUE or WFD_FALSE |

## Port Types [4.5.1.2] WFDPortType – type of display hardware

| | |
|---|---|
| WFD_PORT_TYPE_INTERNAL | **0x7660** |
| WFD_PORT_TYPE_COMPOSITE | **0x7661** |
| WFD_PORT_TYPE_SVIDEO | **0x7662** |
| WFD_PORT_TYPE_COMPONENT_YPbPr | **0x7663** |
| WFD_PORT_TYPE_COMPONENT_RGB | **0x7664** |
| WFD_PORT_TYPE_COMPONENT_RGBHV | **0x7665** |
| WFD_PORT_TYPE_DVI | **0x7666** |
| WFD_PORT_TYPE_HDMI | **0x7667** |
| WFD_PORT_TYPE_DISPLAYPORT | **0x7668** |
| WFD_PORT_TYPE_OTHER | **0x7669** |

## Power Mode [4.5.1.11] WFDPowerMode – indicated but maybe not possible for a specific display hardware . Recovery time to ON decreases from OFF to SUSPEND to LIMITED_USE, and the power consumption will increase..

| | | |
|---|---|---|
| WFD_POWER_MODE_OFF | **0x7666** | No power –frames lost |
| WFD_POWER_MODE_SUSPEND | **0x7667** | Faster recovery then OFF |
| WFD_POWER_MODE_LIMITED_USE | **0x7668** | Frames maintained in hw |
| WFD_POWER_MODE_ON | **0x7669** | Fully operational |

## Partial Refresh [4.5.1.13]

WFD_PORT_PARTIAL_REFRESH_SUPPORT indicates what mode the display hw supports. WFD_PORT_PARTIAL_REFRESH_MAXIMUM defines the max size of the rectangle – (*width*, *height*). WFD_PORT_PARTIAL_REFRESH_RECT defines the actual size (*offsetX*, *offsetY*, *width*, *height*) . WFD_PORT_PARTIAL_REFRESH_ENABLE activates the supported partial refresh mode from WFD_PORT_PARTIAL_REFRESH_SUPPORT.

WFDPartialRefresh – mode supported by the port

| | |
|---|---|
| WFD_PARTIAL_REFRESH_NONE | **0x7690** |
| WFD_PARTIAL_REFRESH_VERTICAL | **0x7691** |
| WFD_PARTIAL_REFRESH_HORIZONTAL | **0x7692** |
| WFD_PARTIAL_REFRESH_BOTH | **0x7693** |

Partial vertical – offsetY and height are used, partial horizontal – offsetX and width are used.

## Querying Port Attributes [7.3] integer or float, single value / vector of values

WFDint **wfdGetPortAttribi**( WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib* )

WFDfloat **wfdGetPortAttribf**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib* )

void **wfdGetPortAttribiv**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDint *count*, WFDint **value* )

void **wfdGetPortAttribfv**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDint *count*, WFDfloat **value* )

## Assigning Port Attributes [7.3] integer or float, single value / vector of values

void **wfdSetPortAttribi**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDint *value* )

void **wfdSetPortAttribf**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDfloat *value* )

void **wfdSetPortAttribiv**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDint *count*, const WFDint **value* )

void **wfdSetPortAttribfv**(WFDDevice *device*, WFDPort *port*,
  WFDPortConfigAttrib *attrib*, WFDint *count*, const WFDfloat **value*)

void **wfdBindPipelineToPort**(WFDDevice *device*, WFDPort *port*,
  WFDPipeline *pipeline*)

# OpenWF Display 1.0 API Quick Reference Card

## Pipelines [5] – is an abstraction of the hardware that transforms and blends source images into the final composited image on the display. Note that mask, rotation and scaling are optional to support.

WFDint **wfdEnumeratePipelines** ( WFDDevice *device*,
WFDint **pipelineIds*, WFDint *pipelineIdsCount*
const WFDint **filterList* )

WFDPipeline **wfdCreatePipeline**( WFDDevice *device*,
WFDint *pipelineId*, const WFDint **attribList* )

void **wfdDestroyPipeline**( WFDDevice *device*, WFDPipeline *pipeline* )

| | | |
|---|---|---|
| WFD_PIPELINE_ID | 0x7720 | (r) |
| WFD_PIPELINE_PORTID | 0x7721 | (r) |
| WFD_PIPELINE_LAYER | 0x7722 | (r) |
| WFD_PIPELINE_SHAREABLE | 0x7723 | (r) |
| WFD_PIPELINE_DIRECT_REFRESH | 0x7724 | (r) |
| WFD_PIPELINE_MAX_SOURCE_SIZE | 0x7725 | (r) |
| WFD_PIPELINE_SOURCE_RECTANGLE | 0x7726 | (r/w) |
| WFD_PIPELINE_FLIP | 0x7727 | (r/w) |
| WFD_PIPELINE_MIRROR | 0x7728 | (r/w) |
| WFD_PIPELINE_ROTATION_SUPPORT | 0x7729 | (r) |
| WFD_PIPELINE_ROTATION | 0x772A | (r/w) |
| WFD_PIPELINE_SCALE_RANGE | 0x772B | (r) |
| WFD_PIPELINE_SCALE_FILTER | 0x772C | (r/w) |
| WFD_PIPELINE_DESTINATION_RECTANGLE | 0x772D | (r/w) |
| WFD_PIPELINE_TRANSPARENCY_ENABLE | 0x772E | (r/w) |
| WFD_PIPELINE_GLOBAL_ALPHA | 0x772F | (r/w) |

### Pipeline Layering [5. 9] - retrieves the pipeline layering order without having to bind the port and pipeline

WFDint **wfdGetPipelineLayerOrder**( WFDDevice *device*,
WFDPort *port*, WFDPipeline *pipeline* )
Returns the same value as for the WFD_PIPELINE_LAYER attribute on success.

### Get/Set Pipeline Attributes [5.7.2] & [5.7.3] integer or float, single value / vector of values

WFDint **wfdGetPipelineAttribi**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDPipelineConfigAttrib *attrib*)

WFDfloat **wfdGetPipelineAttribf**( WFDDevice *device*,
WFDPipeline *pipeline*, WFDPipelineConfigAttrib *attrib* )

void **wfdGetPipelineAttribiv**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDPipelineConfigAttrib *attrib*,
WFDint *count*, WFDint **value* )

void **wfdGetPipelineAttribfv**( WFDDevice *device*,
WFDPipeline *pipeline*, WFDPipelineConfigAttrib *attrib*,
WFDint *count*, WFDfloat **value* )

void **wfdSetPipelineAttribi**(WFDDevice *device,*
WFDPipeline *pipeline,* WFDPipelineConfigAttrib *attrib,*
WFDint *value* )

void **wfdSetPipelineAttribf**( WFDDevice *device,*
WFDPipeline *pipeline,* WFDPipelineConfigAttrib *attrib,*
WFDfloat *value* )

void **wfdSetPipelineAttribiv**(WFDDevice *device,*
WFDPipeline *pipeline,* WFDPipelineConfigAttrib *attrib,*
WFDint *count,* const WFDint **value* )

void **wfdSetPipelineAttribfv**(WFDDevice *device,*
WFDPipeline *pipeline,* WFDPipelineConfigAttrib *attrib,*
WFDint *count,* const WFDfloat **value* )

### Scaling [5.7.1.9]

WFDScaleFilter

| | |
|---|---|
| WFD_SCALE_FILTER_NONE | 0x7760 |
| WFD_SCALE_FILTER_FASTER | 0x7761 |
| WFD_SCALE_FILTER_BETTER | 0x7762 |

### Transparency [5.8]

WFDTransparency – bit field denoting possible combinations of supported transparency

| | |
|---|---|
| WFD_TRANSPARENCY_NONE | = 0 |
| WFD_TRANSPARENCY_SOURCE_COLOR | = (1 << 0) |
| WFD_TRANSPARENCY_GLOBAL_ALPHA | = (1 << 1) |
| WFD_TRANSPARENCY_SOURCE_ALPHA | = (1 << 2) |
| WFD_TRANSPARENCY_MASK | = (1 << 3) |

WFDint **wfdGetPipelineTransparency**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDbitfield **trans*,
WFDint *transCount* )
Query the supported transparency pixel formats.

WFDTSColorFormat – transparent source color type supported

| | |
|---|---|
| WFD_TSC_FORMAT_UINT8_RGB_8_8_8_LINEAR | 0x7790 |
| WFD_TSC_FORMAT_UINT8_RGB_5_6_5_LINEAR | 0x7791 |

void **wfdSetPipelineTSColor**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDTSColorFormat *colorFormat*,
WFDint *count*, const void **color* )
Set transparent color for the pipeline.

## Display Data [4.7]

WFDDisplayDataFormat – format types that could be supported

| | |
|---|---|
| WFD_DISPLAY_DATA_FORMAT_NONE | 0x76A0 |
| WFD_DISPLAY_DATA_FORMAT_EDID_V1 | 0x76A1 |
| WFD_DISPLAY_DATA_FORMAT_EDID_V2 | 0x76A2 |
| WFD_DISPLAY_DATA_FORMAT_DISPLAYID | 0x76A3 |

WFDint **wfdGetDisplayDataFormats**(WFDDevice *device*,
WFDPort *port*, WFDDisplayDataFormat **format*,
WFDint *formatCount* )
Check what dataformats the display supports.

WFDint **wfdGetDisplayData**( WFDDevice *device*, WFDPort *port*,
WFDDisplayDataFormat *format*, WFDuint8 **data*,
WFDint *dataCount* )
Retrieve display data in a specific format.

## Image Sources [5.5.1]

Content that can be used as input to display pipelines.

WFDSource **wfdCreateSourceFromImage**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDEGLImage image,
const WFDint **attribList* )

WFDSource **wfdCreateSourceFromStream**(WFDDevice *device*,
WFDPipeline *pipeline*,
WFDNativeStreamType *stream*,
const WFDint **attribList* )

For streams see also [2.8].

void **wfdDestroySource**( WFDDevice *device* WFDSource *source*)

WFDMask **wfdCreateMaskFromImage**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDEGLImage *image*,
const WFDint **attribList* )

WFDMask **wfdCreateMaskFromStream**( WFDDevice *device*,
WFDPipeline *pipeline*, WFDNativeStreamType stream,
const WFDint **attribList* )

void **wfdDestroyMask**( WFDDevice *device*, WFDMask *mask* )

void **wfdBindSourceToPipeline**( WFDDevice *device,*
WFDPipeline *pipeline,* WFDSource *source,*
WFDTransition *transition,* const WFDRect **region* )

Note – region is the "dirty region" for an EGLImage – should be 0 for stream sources.

void **wfdBindMaskToPipeline**(WFDDevice *device*,
WFDPipeline *pipeline*, WFDMask *mask*,
WFDTransition *transition* )

WFDRect – only relevant for EGLImage sources
(*offsetX, offsetY, width, height*)

WFDTransition

| | |
|---|---|
| WFD_TRANSITION_INVALID | **0x77E0** |
| WFD_TRANSITION_IMMEDIATE | **0x77E1** |
| WFD_TRANSITION_AT_VSYNC | **0x77E2** |

## Renderer and extension information [6]

WFDStringID – information about the runtime platform

| | |
|---|---|
| WFD_VENDOR | **0x7500** |
| WFD_RENDERER | **0x7501** |
| WFD_VERSION | **0x7502** |
| WFD_EXTENSIONS | **0x7503** |

WFDint **wfdGetStrings**(WFDDevice *device*, WFDStringID *name*,
const char ***strings*, WFDint *stringsCount* )

WFDboolean **wfdIsExtensionSupported**(WFDDevice *device*,
const char **string* )

The Khronos Group is an industry consortium creating open standards for authoring and acceleration of parallell computing. Graphics and dynamic media on a wide variety of platforms and devices.

See www.khronos.org/openwf to learn more about the Khronos Group. And OpenWF