Chapter 9: `Rendering` 🎨📃

1. `vkAcquireNextImageKHR()`

- https://vkdoc.net/man/vkAcquireNextImageKHR
  - `.device` = ▨ *Same as* `SwapChain` 🧑‍🏫
    - *So, now you know which class this function has got to be inside* 😮
  - `.swapchain` = ▨ 🧑‍🏫
  - `.timeout` ⏱️ ⚡ `nanoseconds`
    - *specifies how long the function waits, in* ⏱️ ⚡ `nanoseconds`*, if no image is available.*

      ```
      uint64_t ns_per_second = 1'000'000'000;
      ```

  - `.semaphore` 🔗 *SubChapter 2*
  - `.fench` ▨ *ChapterZZZ*
  - `.pImageIndex` ↩️ 📦
    - *Well, this function doesn't return an* `VkImage` *but an index to it* 🧑‍🏫
- 📜 **REY_DOCs**
  - `VK_SUBOPTIMAL_KHR`
    - *if the window has been resized but the OS/platform's* `GPU-DriverImplementation` / `PresentationEngine` *is still able to scale the presented images to the new size to produce valid surface updates.*
    - *It is up to the application to decide whether it prefers to continue using the current swapchain in this state, or to re-create the swapchain to match resized window.*
  - `VK_ERROR_OUT_OF_DATE_KHR`
    - *the images in the swapchain no longer matches the surface properties (e.g., the window was resized)*
    - *and the presentation engine can't present them,*
    - *so the application needs to create a new swapchain that matches the surface properties.*
  - *REFs:- 1. minerva*

---

2. `VkSemaphore` ▨ ChapterZZZ

- https://vkdoc.net/man/VkSemaphore
  - *I wouldn't suggest reading it right now tho* 😅
  - *But, basically,*
    - `SemaPhore` *will be used to synchronize the rendering and presentation of images*

1. `VkSemaphoreCreateInfo`
   - *https://vkdoc.net/man/VkSemaphoreCreateInfo*
     - `.sType` = ▨ `VK_STRUCTURE_TYPE_SEMAPHORE_CREATE_INFO`
     - `.pNext` = 🪐 `NULL`
     - `.flags` = ▨ `0`

2. `vkCreateSemaphore`
   - *https://vkdoc.net/man/vkCreateSemaphore*
     - `.device`
     - `.pCreateInfo` ▨ 🧑‍🏫
     - `.pAllocator` ▨ *ChapterZZZ*
     - `.pSemaphore` ↩️ 📦

`📽️ So far, The result` 📝 **4.guide.chapter9.3.swapchain.hh**

3. Command Recording

1. `VkCommandBufferBeginInfo`
   - *https://vkdoc.net/man/VkCommandBufferBeginInfo*
     - `.sType` = ▨ `VK_STRUCTURE_TYPE_COMMAND_BUFFER_BEGIN_INFO`
     - `.pNext` = ▨ `NULL`
       - ∅ `VkDeviceGroupCommandBufferBeginInfo`
     - `.flags` 🔠 `VkCommandBufferUsageFlagBits`
       - *https://vkdoc.net/man/VkCommandBufferUsageFlagBits* | *ivirtex-github*
         - 🔠 `ONE_TIME_SUBMIT`
         - 🔠 `RENDER_PASS_CONTINUE` *[secondary command buffer]*
         - 🔠 `SIMULTANEOUS_USE`
     - `.pInheritanceInfo` ∅ *[secondary command buffer]*

2. `VkRenderPassBeginInfo`
   - *https://vkdoc.net/man/VkRenderPassBeginInfo*
     - `.sType` = ▨ `VK_STRUCTURE_TYPE_RENDER_PASS_BEGIN_INFO`
     - `.pNext` = ∅ `NULL`
     - `.renderPass` = ▨ 🧑‍💻
     - `.framebuffer` = ▨ 🧑‍💻
     - `.renderArea`
       - *https://vkdoc.net/man/VkRect2D*
     - `.pClearValues`
       - *https://vkdoc.net/man/VkClearValue*

4. `amVK_SurfacePresenter`

Can't have everything scatterred now, everything is getting too much sophisticating.... 🤔 🤦‍♀️ must *Refactor....*

Major Decision Change

Right now, `amVK_Surface::CTOR` creates `amVK_SurfacePresenter` . & `SwapChain, RenderPass, CommandPool` are supposed to be created from `amVK_SurfacePresenter` .

```
class amVK_Surface
    amVK_SurfacePresenter {
        create_SwapChain_interface()
            new amVK_SwapChain(this)
                this->CI.surface = PR->S->vk_SurfaceKHR;
                // later amVK_SwapChain::CreateSwapChain(void) uses this->PR->D->vk_Device
        create_RenderPass_interface()
            new amVK_RenderPass(this)
                this->PR = PR;
        create_CommandPool_interface()
            new amVK_CommandPool(this)
                this->CI.queueFamilyIndex = this->PR->D->amVK_1D_QCIs.ptr_Default()->queueFamilyIndex;
        create_FrameBuffers()
            new amVK_FrameBuffer(this)
                this->CI.renderPass = this->PR->RP->vk_RenderPass;
```

Problem #1:- I think this is just a little too much deep to handle....
Problem #2:- if `amVK_SwapChain.hh` included `amVK_SurfacePresenter.hh` , then the reverse can't happen. 💁‍♀️
Thus a lot of 1-liner functions would have to be put inside `.cpp` even tho i don't want it to.

1. `Problem #2:- in Details`

- *amVK_SurfacePresenter.hh#L37*
- *amVK_SwapChain.hh#L48*
- *The Solution*
    - `C1` *:- Don't include* `amVK_SurfacePresenter.hh` *in* `amVK_SwapChain.hh` *but rather inside* `amVK_SwapChain.cpp`
    - `C2` *:- Don't include* `amVK_SwapChain.hh` *in* `amVK_SurfacePresenter.hh` *but rather inside* `amVK_SurfacePresenter.cpp`
- `Case 1` :-
    - `amVK_SwapChain::CONSTRUCTOR`
    - `sync_SurfCaps()`
    - *both of these have to go inside* `amVK_SwapChain.cpp`
- `Case 2` :-
    - `amVK_SurfacePresenter::sync_SC_SurfCaps()`
    - `amVK_SurfacePresenter::synced_ImageExtent()`
    - *both of these (& as of my plan right now, heck ton of other 1 liner function) are gonna have to go inside* `amVK_SurfacePresenter.cpp`

2. Weeelll

- *There is one other solution.... That is to change the design.... Which is what I figured is should do.... Not everybody would want to use* `amVK_SurfacePresenter` *anyway* 💁‍♀️
- *2 Ways:-*

i. *Making* `amVK_SurfacePresenter` *Optional*
    a. *None of the other amVK_Class is gonna depend on this anymore*
    b. *amVK_SurfacePresenter serving as like a top level NODETREE system with extra PRESET Functions / soo. (If you are looking from a NodeEditor perspective)*
    c. *This is like having a BIG BAD NODE, and then connecting everything into it*
    d. *You can have anything you want in the header*

        e. *Let's try the other one and see what happens*

ii. *Making* `amVK_SurfacePresenter` *Code part*
    a. *EveryBody is gonna depend on this*
    b. *They are only gonna keep a pointer to this parent*
    c. *from this one, they are gonna get everything that they need*
    d. *even the VkDevice*
    e. *It's like having all the nodes inside a TOP LEVEL FRAME NODE*
    f. *Separating Code into .hh & .cpp is kinda crazy..... You basically can't have anything in the header....*
    g. *i already tried this*

Before Commit:- https://github.com/REYNEP/amGHOST/blob/9cec3e58db123144bd8d88363ccf9a4a7ffc9edc/amVK/amVK_Surface.hh
Middle (Discarded) Commit:- https://github.com/REYNEP/amGHOST/blob/3be7cfcd154b383cd98783d302468f63fda0618b/amVK/amVK_SurfacePresenter.hh
Final Commit:- https://github.com/REYNEP/amGHOST/blob/7376cdb5c2c6eee19655dae436e6cf8edd02e1d5/amVK/amVK_SurfacePresenter.hh

`🎥 So far, The result` [🔗 `GITHUB`]

- 📁 `common`
    - 📝 *amVK.hh*
    - 📝 *amVK_ColorSpace.hh*
    - 📝 *amVK_Enum2String.cpp*
    - 📝 *amVK_Enum2String.hh*
    - 📝 *amVK_GPU.hh*
    - 📝 *amVK_RenderPass_Descriptors.hh*
    - 📝 *amVK_log.cpp*
    - 📝 *amVK_log.hh*
- 📁 `core`
    - 📝 *amVK_Instance.hh*
    - 📝 *amVK_Device.hh*
    - 📝 *amVK_DeviceQCI.hh*
    - 📝 *amVK_Surface.hh*
    - 📝 *amVK_SwapChain.hh*
    - 📝 *amVK_SwapChainIMGs.hh*
    - 📝 *amVK_RenderPass.hh*
    - 📝 *amVK_RenderPassFBs.hh*
    - 📝 *amVK_CommandPool.hh*
- 📝 amVK_SurfacePresenter.hh

- 📁 `extras`
    - 📝 *SCREENSHOT_STUDIO.hh*
    - 📝 *amVK_CommandBuffer.hh*
    - 📝 *amVK_FrameBuffer.hh*
    - 📝 *amVK_Image.hh*
    - 📝 *amVK_SemaPhone.hh*
- 📁 `guide`
    - *(Directory placeholder – add guide files here if any)*
- 📁 `impl`
    - 📝 *amVK_Device.cpp*
    - 📝 *amVK_Instance.cpp*
    - 📝 *amVK_InstanceProps.cpp*
    - 📝 *amVK_InstancePropsExport.cpp*
    - 📝 *amVK_InstancePropsExport_nloh...*
    - 📝 *amVK_Surface.cpp*
    - 📝 *amVK_SurfacePresenter.cpp*
    - 📝 *amVK_SwapChain.cpp*

5. `Back 2 Command Recording`

3. `vkBeginCommandBuffer()`
   - https://vkdoc.net/man/vkBeginCommandBuffer
     - `.commandBuffer` ▨ 🧑‍💻
     - `.pBeginInfo` ▨ 🧑‍💻
   - `</> TheCode`

```
amVK_CommandPool {
  public:
    REY_Array<VkCommandBuffer>      vk_CommandBuffers;
    REY_Array<VkCommandBuffer> AllocateCommandBuffers(void);

  public:
    VkCommandBufferBeginInfo BI = {
        .sType = VK_STRUCTURE_TYPE_COMMAND_BUFFER_BEGIN_INFO,
        .pNext = 0,
        .flags = VK_COMMAND_BUFFER_USAGE_ONE_TIME_SUBMIT_BIT,
        .pInheritanceInfo = nullptr
    };
    void BeginCommandBuffer(uint32_t CMDBUF_Index) {
        VkResult return_code = vkBeginCommandBuffer(vk_CommandBuffers[CMDBUF_Index], &BI);
        amVK_return_code_log( "vkBeginCommandBuffer()" );
    }
}
```

4. `vkCmdBeginRenderPass()`
   - https://vkdoc.net/man/vkCmdBeginRenderPass
     - `.commandBuffer` ▨ 🧑‍💻
     - `.pRenderPassBegin` ▨ 🧑‍💻
     - `.contents` 🔠 `VK_SUBPASS_CONTENTS_INLINE`
       - https://vkdoc.net/man/VkSubpassContents | ivirtex-github
         - 🔠 `INLINE`
         - 🔠 `SECONDARY_COMMAND_BUFFERS` *[secondary command buffer]*
         - 🔠 `INLINE_AND_SECONDARY_COMMAND_BUFFERS_KHR` *[VK_KHR_maintenance7]*
         - 🔠 `INLINE_AND_SECONDARY_COMMAND_BUFFERS_EXT` *[VK_EXT_nested_command_buffer]*

5. `vkCmdSetViewport()`
   - https://vkdoc.net/man/vkCmdSetViewport
     - `.commandBuffer` ▨ 🧑‍💻
     - `.firstViewport` ▨ `0`
     - `.viewportCount` ▨ `1`
     - `.pViewports` ▨ `VkViewport`
       - https://vkdoc.net/man/VkViewport
6. `vkCmdSetScissor()`
   - https://vkdoc.net/man/vkCmdSetScissor
     - `.pScissors` ▨ `VkRect2D`
       - https://vkdoc.net/man/VkRect2D
7. `vkCmdEndRenderPass()`
   - https://vkdoc.net/man/vkCmdEndRenderPass
     - `.commandBuffer` ▨ 🧑‍💻
8. `vkEndCommandBuffer()`
   - https://vkdoc.net/man/vkEndCommandBuffer
     - `.commandBuffer` ▨ 🧑‍💻

6. `Submit Command Buffer`

1. `VkSubmitInfo`
   - *https://vkdoc.net/man/VkSubmitInfo*
     - `.sType` ▨ `VK_STRUCTURE_TYPE_SUBMIT_INFO`
     - `.pNext` ⊘ `NULL`
     - `.pWaitSemaphores` 🔗 *Chapter9.1*
       - ▨ `amVK_SwapChain::AcquireNextImage_SemaPhore`
     - `.pWaitDstStageMask` ▨ `VK_PIPELINE_STAGE_COLOR_ATTACHMENT_OUTPUT_BIT`
     - `.pCommandBuffers` ▨ 🧑‍💻
     - `.pSignalSemaphores`
       - ▨ `amVK_SurfacePresenter::RenderingFinished_SemaPhore`

---

2. `vkQueueSubmit()`
   - *https://vkdoc.net/man/vkQueueSubmit*
     - `.queue` ▨ `GraphicsQueue`
     - `.submitCount` ▨ `1`
     - `.pSubmits` ▨ 🧑‍💻
     - `.fench` ▨ `VK_NULL_HANDLE`
3. `vkGetDeviceQueue()`
   - *https://vkdoc.net/man/vkGetDeviceQueue*
     - `.device`
     - `.queueFamilyindex` 🔗 *Chapter2.7*
       - `amVK_Device::amVK_1D_QCIs::select_QFAM_Graphics()`
     - `.queueIndex` 🔗 *Chapter2.4*
       - `VkDeviceQueueCreateInfo.queueCount`
     - `.pQueue` ↩ 📦

---

4. `VkPresentInfoKHR`
   - *https://vkdoc.net/man/VkPresentInfoKHR*
     - `.sType` ▨ `VK_STRUCTURE_TYPE_PRESENT_INFO_KHR`
     - `.pNext` ⊘ `NULL`
       - ⊘ *Maybe some interesting extensions, idk*
     - `.pWaitSemaphores` 🔗 *Chapter9.6*
       - ▨ `amVK_SwapChain::RenderingFinished_SemaPhore`
     - `.pSwapchains` ▨ 🧑‍💻
     - `.pImageIndices`
     - `.pResults`
5. `vkQueuePresentKHR()`
   - *https://vkdoc.net/man/vkQueuePresentKHR*
     - `.queue` ▨ 🧑‍💻
     - `.pPresentInfo` ▨ 🧑‍💻
6. `🎥 So far, The result` [🔗 `GITHUB`]
   - *(Adding after commiting and getting a hash....)*