

<INNERMOST LOOP PER RENDER PASS>

VkRenderPassBeginInfo

```
sType = VK_STRUCTURE_TYPE_RENDER_PASS_BEGIN_INFO;
pNext = nullptr;
renderPass; ← VkRenderPass
framebuffer; ← VkFramebuffer
renderArea;
clearValueCount;
pClearValues;
```

typedef enum VkSubpassContents {

```
VK_SUBPASS_CONTENTS_INLINE = 0,
VK_SUBPASS_CONTENTS_SECONDARY_COMMAND_BUFFERS = 1,
} VkSubpassContents;
```

```
typedef struct VkClearDepthStencilValue {
    float    depth;
    uint32_t stencil;
} VkClearDepthStencilValue;
```

typedef union VkClearColorValue {

```
float    float32[4];
int32_t  int32[4];
uint32_t uint32[4];
} VkClearColorValue;
```

typedef union VkClearValue {

```
→VkClearColorValue    color;
→VkClearDepthStencilValue depthStencil;
} VkClearValue;
```

```
void vkCmdBeginRenderPass(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    const VkRenderPassBeginInfo* pRenderPassBegin,
    →VkSubpassContents    contents
);
```

```
void vkCmdBindPipeline(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    VkPipelineBindPoint pipelineBindPoint, ← VkPipelineBindPoint
    VkPipeline            pipeline ← VkPipeline
);
```

```
void vkCmdBindVertexBuffers(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    uint32_t             firstBinding,
    uint32_t             bindingCount,
    const VkBuffer*       pBuffers, ← VkBuffer
    const VkDeviceSize*  pOffsets
);
```

```
void vkCmdBindIndexBuffer(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    VkBuffer              buffer, ← VkBuffer
    VkDeviceSize          offset,
    VkIndexType           indexType
);
```

```
void vkCmdBindDescriptorSets(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    VkPipelineBindPoint pipelineBindPoint, ← VkPipelineBindPoint
    VkPipelineLayout      layout, ← VkPipelineLayout
    uint32_t             firstSet,
    uint32_t             descriptorSetCount,
    const VkDescriptorSet* pDescriptorSets, ← VkDescriptorSet
    uint32_t             dynamicOffsetCount,
    const uint32_t*       pDynamicOffsets
);
```

```
void vkCmdDrawIndexed(
    VkCommandBuffer      commandBuffer, ← VkCommandBuffer
    uint32_t             indexCount,
    uint32_t             instanceCount,
    uint32_t             firstIndex,
    int32_t              vertexOffset,
    uint32_t             firstInstance
);
```

```
void vkCmdEndRenderPass(
    VkCommandBuffer commandBuffer
);
```

typedef enum VkPipelineBindPoint {

```
VK_PIPELINE_BIND_POINT_GRAPHICS,
VK_PIPELINE_BIND_POINT_COMPUTE,
VK_PIPELINE_BIND_POINT_RAY_TRACING_KHR,
VK_PIPELINE_BIND_POINT_SUBPASS_SHADING_HUAWEI,
VK_PIPELINE_BIND_POINT_RAY_TRACING_NV,
} VkPipelineBindPoint;
```

typedef struct VkViewport {

```
float    x;
float    y;
float    width;
float    height;
float    minDepth;
float    maxDepth;
} VkViewport;
```

```
void vkCmdSetViewport(
    →VkCommandBuffer      commandBuffer,
    uint32_t             firstViewport,
    uint32_t             viewportCount,
    →const VkViewport*    pViewports
);
```

typedef struct VkRect2D {

```
VkOffset2D offset;
VkExtent2D extent;
} VkRect2D;
```

```
void vkCmdSetScissor(
    →VkCommandBuffer      commandBuffer,
    uint32_t             firstScissor,
    uint32_t             scissorCount,
    →const VkRect2D*      pScissors
);
```