



Ref & PassRef

Efficient never-null pointers

Ref<T>

- Always points to an object
- Always holds a strong reference to that object
- Initialized with a T& or a PassRef<T>
- Use .get() to access the T&
- Doesn't need to do null checks in ctor/dtor

PassRef<T>

- Like PassRefPtr<T> but never null
- Used to initialize Ref<T>
- Must be sunk into a Ref<T> or explicitly dropRef()'ed
- Doesn't generate null checks or calls ~T() like PassRefPtr



RenderPtr<T>

Simple owning pointer for renderers

RenderPtr<T>

- Basically an `OwnPtr<RenderObject>`
- Calls `RenderObject::destroy()` instead of `~RenderObject()`
- Goal is to move the rendering code to using smart pointers for ownership clarity and general safety