omigroup / **gltf-extensions** Public

<> Code   ⊙ Issues 8   Pull requests 4   **Discussions**   ⊙ Actions   ···

glTF Working Group Meeting 1/13/2022 #49

robertlong started this conversation in **General**

robertlong on Jan 10, 2022 Maintainer   edited ▾

OMI glTF Working Group Meeting 1/13/2022

This meeting will be on 1/13/2022 at 10:30 PM (UTC) / 2:30 PM (PST) in the OMI Discord within the #omi-weekly-meeting channel. During the meeting, we will be using the #omi-gltf-extensions channel to manage a speaker queue, post links, and for any sidebar discussions.

To be notified of this meeting and others, subscribe to the OMI Meetings and Events Calendar or add yourself to the @omi-gltf-subgroup role in the #roles channel of the OMI Discord.

Agenda

- OMI_audio_emitter
  - Discuss Explainer doc: **Finalize Explainer docs** #46
  - Blender Addon Updates
- Collider extension
- Animation retargeting for VRM workflow

To propose another item for the agenda, comment below (preferably before the meeting).

↑ 1

2 comments   **Oldest** Newest Top

robertlong on Jan 13, 2022 Maintainer Author   edited ▾

Notes:

- Discussed OMI_audio_emitter extension: Merged **Finalize Explainer docs** #46
- Discussed colliders: Added notes to **glTF Collision Extension** #50
- Animation Retargeting:
  - Webaverse's .metaverse files
    - Attaching objects to various bones
    - https://github.com/webaverse/potion/blob/main/.metaversefile#L8

↑ 1   0 replies

RangerMauve on Jan 13, 2022

Notes

OMI_audio_emitter

- Going to merge explainer doc. We can add contributers later if they aren't already in there.

Collider extension

- Mauve: Went over discussion **glTF Collision Extension** #50
- VRM springbone colliders? Should we consider them? https://vrm.dev/en/docs/vrm1/changed/#vrmc_springbone
- Should look at virtualcast https://github.com/virtual-cast/VCI/tree/master/Assets/VCI/UniVCI/Scripts/Format/GltfExtensions/NodeExtensions/glTF_VCAST_vci_colliders
- Robert: Could link to accessor within the extension to something like a trimesh while leaving it invisibile in default viewers.
- IFire: Wanted to degrade to regular GLTF,
- Robert: It should use the accessor so that it won't be visible
- Mauve: I think meshes can be refereced by their index in the array
- Robert: Might not need to even be a real mesh, but just the trimesh that's an array of vertecies. Wrote a trimesh extension for hubs for the terrain.
- IFire: Do we want to force it to only be convex shapes?
- Lyuma: Can we have convex trimeshes? Just don't want rigid bodies with concave
- IFire: One could always turn concave into convex
- Mauve: What's the reason not to support it?
- IFire: Wanted to know if we want it to present well on the engine side
- Lyuma: Worried about performance, could make really bad perf if you use concave meshes incorrectly. If you ahve lots of polys if they're in a small volume (like the face), would have lots of intersect tests and freeze the game. One *can* so people probably shouldn't use it
- IFire: The idea is it should be ready to use for runtime use.
- Mauve: I think it'd be useful for folks to at least have the option.
- IFire: TBH, I'd have it auto-converted to hull
- Lyuma: But you probably shouldn't convert it. E.g. a house might be better as a trimesh
- Koolala: Do many static boxes end up getting merged into one large static trimesh for collisions? Or is keeping around all the boxes actually better?
- Lyuma: Keeping boxes is better. 50k box colliders run fine even on iOS, very efficient.
- Mauve: Would you be okay with concave for edge cases?
- IFire: VRM only supports capsules for collision
- Lyuma: They support both with a bool
- IFire: Do we want to ban them anyway?
- Robert/Mauve/Lyuma: Not really, we want concave
- humbletim:Where do these collisions come from? Blender?
- IFire: You generate the shapes as part of a workflow.
- humbeltime: Then re-generate on changes?
- Mauve: Usually folks create empties in Blender and name them to be colliders for the Godot export
- humbeltime: How do the colliders get rendered?
- IFire: They usually get drawn in visibly
- Robert: Hubs actually supports having multiple ammo shapes. (link in dicsussion)

- IFire: Found a convex decomposition thing
- Robert: Kevin from hubs did a lot of work on this.
- humbletim: Collision models from blender: https://www.katsbits.com/tutorials/blender/collision-models.php
- Robert: Auto-generating shapes works pretty well, but does break in a lot of instances, would want to manually define trimeshes as well. Would be good to have a default and the ability to override
- humbletim: In Hifi, we used fbx or obj for collisison model URLs. It was a pain in the butt, adding trimesh support was easier for creators. Led to performance bottlenecks
- Robert: Found a thing for splitting concave into convex. Seems like somethign that should be done upfront instead of on the fly. Might want to tweak it. Somtimes you'll put an entire environemnt, then auto-generate colliders and get messy colliders. Had this happen in hubs and it wasn't optimized.
- humbletim: To back up, what's the purpose of colliders?
- IFire: body, props, hair
- Lyuma: Static and dynamic bodies. Level/terrain. Rigid bodies is something you pick up.
- Rick: Might want to tie other aspecs to the data
- Lyuma: Should reccomend using different mesh data from rendering
- Mauve: I think it should be useful for collision like entering an area or pressing a button
- Lyuma: Trigger colliders? Some environments require those to be primitives
- Rick: Also blocking seeing through.
- Lyuma: Occlusion culling?
- Rick: You want something to be dropped in an enemy could not see through for ray casting. e.g. you can move through but it won't block the line of sight.
- IFire: Imagine a tree is a quad, and you're behind it. That kind of thing.
- Lyuma: Interacing with the behavior
- Lyuma: Probably way out of scope via stuff like layers and masks, quite relevant to these questions. e.g. visibility detection
- Rick: Good example, I'm working on a mech warrior game, like you can walk over some debris as a mech, can't as a pilot. Works with physics collision matrix
- Mauve: Could we table the physics and masking stuff as being something after we get colliders together?
- Lyuma: There's overlap in these concepts. There are some engines that will forbid you to use a trimesh for a rigid body. Regarding layers, usually there's a small number of layers.
- Should define mroe use cases and scoping
- Robert: Just having the collider and gemetry could be good to start with. Just knowing that would be good and then we can work on additional extensions like friction, dynamic rigid bodies, static bodies. Maybe we're just specifying the collision?
- Lyuma: Concern for OMI. If people implement these specs. WHat if they have something like a teacup that they'd want to pick it up. It could be useful as defining if something is static or not. So adding ones that don't block movement.
- IFire: want to preserve the use case of something being immovable while it's on the ground.
- Mauve: Might be useful to figure out after with physics
- Lyuma: Can we even get it to stage 4 if we don't have stuff visible?
- Mauve: Good point maybe we need to have physics discussion as part of colliders?
- Lyuma: Immediate payoff is level geometry
- Robert: Or objects with colliders you can grab. e.g hubs generate one for something like a duck, but others not at all.
- Mauve: Should we have something like static/dynamic/rigid in the parent?
- Lyuma: Yeah, maybe extend that with future types that default to "no collision"
- Mauve: Maybe have additional extensions on top of no-collision?
- Lyuma: e.g. ai-trigger is no-collision with another extension
- Robert: I wonder if we can do that at the application level. E.g. where it's brought it can determine if it's static or dynamic. Like a prop is probably dynamic. If it's environment probably mostly static. Maybe a small piece will be dynamic, but can we define the behaviors in the spec?
- antpb: Can we offer additional fields?
- Lyuma: That could be an extension. Dynamic things like probably shouldn't be allowed to use static. A map *could* have dynamic stuff. Also trigger areas. The application side is where I define what happens, but would want to include the area

in the area. Also kinematic?

- humbletim: Kinematic can be moving, but it's not part of the physics sim. Some engines have this disctinction
- IFire: Like not physically simulated, but animated
- Rick: Also talking about WASM. A holy grail could be an avatar with a GLTF with a WASM component, so behaviors associated with it.
- Mauve: That's a big use case I had in mind. E.g. having tigger boxes, then the GLTF listen for the trigger and set an animation or react in some other way. We can give people a lot of power for very little extra code.

Demo from Avaer and Jin

- Jin: We've got stuff like animation retargeting. Will screen share something like changing an avatar and metaverse files and how they work. Showed picking up a potion with an avatar for e.g.
- Avaer: In terms of metaverse files, it's like a GLTF extension, but we're putting it in a separate .metaverse file that gets put beside the GLTF and explains what the file does. We put it on top of the actual files we use. Part of a potions behavior is that it can be picked up, and we specify which bone you can attach to. For example when you're wearing it it's attached to your left hand (on a VRM).
- IFire: Is passing it to another person possible?
- Avaer: Every object has an "owner" and that's either a person or the world. So if you pick it up it gets owned by your character rather than the world and can be brought with you. Referencing the potion as a URL within the scene files.
- IFire: So moving from the world to you is parenting, can be tricky?
- Avaer: This is all ThreeJS based, so you're reparenting from the world to your avatar. Colliders are automatically generated using autotrimesh. I found in engines if you have gameplay you want, it's a matter of cusomtizing physics. WHich objects have them and which don't.And which kinds of colliders. Found most engines support the same sorts of primitives. So if you support those things, it's easy to write a plugin in any engine and get it in. After you need to do optimization of constants, but that probably shouldn't be in standards and isntead in games. But in the first place having physics properties for an object is important.
- IFire: Talked before about wanting something to be "movable" and this is required for this use case.
- humbletim: How did you get the positions in the metaverse file?
- Avaer: In the future we're hoping for tools in game
- humbletim: Is there a standard scheme?
- Avaer: one meter
- humbletim: Can you scale them?
- Avaer: You can scale them right now
- humbletim: One thing I saw is puffing out collision hulls, adding a bit of a buffer so it isn't going to go into the wall
- Avaer: THat's something in a physics engine, not something that should go into the engine.
- humbletim: Scale matters, and if we aren't assuming a cosmic scale, the same object might indicate different qualities in different worlds
- Koolala: If you can scale avatars, and avatars are the scenes, then being a 2x scale avatar might make stuff you pick up become 2x
- Lyuma: Fits into issues with trimeshes, they break down when you scale them. The way it's implemented is you check every tick which polygons intersect. If you have a lot of items in a room it'd be a project.
- Avaer: Most engines have a quantization process during the baking process. This model is very high quality, but the generated mesh is very efficient.
- Lyuma: If you're allowed to change this at runtime there's no problem, if you know the scale it shou--- (I lost track)
- Avaer: The potions for example ahve a different scale when you grab it
- Jin: Regarding animation, they don't come with the GLTF
- Avaer: Two schools of thought, manually animating each character, or retargetting one to many. We retarget at load time for a VRM rig. Where this plays into standards is the objects also have "use" properties which talk about which animations it works with. e.g. "drink". Each character knows how to drink.
- IFire: Curious about how you attach and how you attach to a skeleton for animation.
- Avaer: There's a regular Three skeleton and we attach (the potion) to the skeleton directly.

- Avaer: We have a case like wearing helmets where we warp to your character. If you don't get the morphs right, the helmet will poke through the hair. Much harder problem.
- IFire: Even attaching to the skin correctly needs to be the exact bone.
- Avaer: Hoping for tooling to do it automatically
- Evo: Is this working in a networked situation?
- Avaer: Everything under the hood uses CRDT deltas to compute the positions, changing replicated data and the single player world alsso reacts to it. Was using Y.js. Didn't support binary, rewrote to Y.js
- Evo: Same process for attaching to an avatar?
- Avaer: Correct that's why when a player enters they bring all their objects with them
- Robert: You mentioned the mxmo animation is done at load time. Is there a way to store that in the GLTF file that's already normalized to the skeleton?
- Robert: Proposing storing the animation in a separate GLTF file that are already ready to go for regargetting against a VRM skeleton
- Avaer: Two files?
- Robert: Two, yes
- IFIre: E.g. animation1, animation2
- Avaer: Seems like an abuse of GLTF
- Robert: GLTF is built as a container that's for any of it's sub nodes. Been hacking on just materials in a GLTF. The spec leaves it up to the engine in order to figure out how to load these resources from a GLTF.
- Jin: Reminds me of how previous gens used mpeg, VRML for avatars. Is GLTF our version of it?
- Nova: GLTF is a lot more like an SVG.
- Robert: If we can normalize for a single format, we can get rid of the bloat for needing e.g. FBX. Can package up stuff like animations, or potions could contain animations for drinking itself
- Jin: Reminds me of a cool collection of armor, all the GLTF files are attached to GLTF armatures in VRM files. E.g. props could be attached to armatures for exporting, can export with the JSON.
- Avaer: I wonder if it makes sense to have a set of standard animations, or whether everything should be in the object. I could see it getting heavy. E.g. 100 potions with their own animations. Or should drinking be a thing
- IFire; Hard to agree on what those are, everyone could make their own. At least some people will want extra animation sets just because
- Nova: For XR, a full body tracking extension would let you do this. Is it a bad thing if you have a bunch of use animations? THey're pretty cheap all things considered
- IFire: Cheap compared to images and audio and videos
- IFire: Do we want to specify a list of animation sets?
- Robert: Being able to reference them from separate files would also be useful. So if 30 files reference one, we can do that.
- Lyuma: Unclear, why would the potion be responsible
- Avaer: Currently isn't specifies it's drinkable. How would one reference the GLTF?
- Lyuma: If a character wants a custom one they can. There should be a platform default
- IFire: There's three things: Animation sets on the char or object, object having specific tag for the game platform
- Rick: Specifying "Drinkable" or "does damange" would be useful.
- Avaer: Would it be the chars or the objs?
- Lyuma: I don't think objects should. There have been cases. "Stations". Characters can play animations specified by the object you're itneracting with. e.g a dance station.
- Jin: Seen in MMD worlds
- Lyuma: Could see both sides. "Does damage" sounds like a tag. Application specific data. Protocols are an interesting concept for metaverse applications. Two content creators agreeing to a protocol without the platofrm agreeing. Person A says they're using a tag for a damange system, person B could implement it too.
- Nova: In minecraft they eventually had a standard "FE".
- Robert: What can we do to wrap up the animation retargetting?
- Avaer: What's the procedure for starting the standard.
- IFire: Gotta join W3C, then pull request

- antpb: COuld go in a discussion to talk about it before the PRs.
- Robert: starting with a discussion, then making a markdown doc
- IFire: Is there a specific one you want?
- Avaer: Getting the colliders definitions.

Wrap up:

Robert: Thanks for coming in. Will see you next week. If you're coming to the WASM meeting, it's 1 hour earlier than last week.

↑ 1

0 replies

Category

General

Labels

None yet

2 participants