Welcome!

Configura Internal Documentation

This documentation details all scripts used to power the Configura plugin. This includes the Configura Dock, loading in meshes, and setting up the UI. For basic uses, users will not have to touch these files.

Feel free to look through the documentation if you'd like to see under the hood. If you would like to add or edit the source code, you are welcome to fork the repository!

For information on uploading your 3D model(s) and creating the character creator scene, refer to Developer Documentation.

For a more detailed description of under-the-hood scripts, refer to Internal Documentation.

Version: Godot 4.6.2

⚙️ File Structure

Configura Internal Documentation

Godot Version: 4.6.2

Project File Hierarchy

- addons folder - Core functionality
    - Configura folder
        - data folder
            - options folder
                - animation_option.gd
                - blendshape_option.gd
                - bone_target.gd
                - color_option.gd
                - deform_modifier_3d.gd
                - deform_option.gd
                - mesh_swap_choice.gd
                - mesh_swap_option.gd
                - morph_option.gd
                - option_definition.gd
                - texture_atlas_option.gd
            - character_config.gd
            - character_load_row.gd
            - character_state.gd
        - editor folder
            - docks folder
                - configura_dock.gd
                - group_container.gd
                - option_row.gd
            - generators folder
                - animation_tree_builder.gd
                - load_generator.gd
                - scene_generator.gd
                - ui_generator.gd
            - inspectors folder
                - mesh_inspector.gd
        - icons folder
        - runtime folder
            - controls folder
                - anim_row.gd
                - color_row.gd
                - slider_row.gd
                - swap_group.gd
            - themes folder

            - creator_controller.gd
            - creator_loader.gd
            - creator_manager.gd
            - character_preview.gd
            - creator_ui.gd
        - character_state_applier.gd
        - configura_plugin.gd
- builds folder - Web executable
- meshes folder - Meshes and Assets

Data

📂 Options

animation_option.gd

animation_option Script

Overview

Drives the AnimationPlayer. Used to let the player choose an idle pose or expression preset.

Deprecated - animations were temporarily removed for v1.0, will be implemented in a future release.

Variables

Name	Type	Description
animation_player_path	NodePath	NodePath to the AnimationPlayer. Relative to SubViewport root
animation_name	String	The animation to play
loop_in_preview	Bool	Whether to loop the animation in preview
include_in_export	Bool	Whether to include animations in the exported CharacterState
tree_node_name	String	Used by CharacterExporter to set the PoseTransition transition_request parameter
is_default	Bool	Whether this animation is applied when the character creator first opens

Methods

Name	Description
to_string()	Returns a string with animation options settings
get_option_category()	Returns this option’s category
get_default_value()	Returns this option’s default value
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s animation to the character preview

apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option’s animation to the character
create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates the animation button row

blendshape_option.gd

blendshape_option Script

Overview

Drives a single blendshape weight via an HSlider. Maps to Blender shape keys.

Variables

Name	Type	Description
mesh_paths	Array[NodePath]	Array of NodePaths to the MeshInstance3D nodes that share this blendshape
editor_groups	Array[String]	Tracks all the mesh groups this shape appears in, used by the editor dock only (not read at runtime).
blend_shape_name	String	The exact string returned by get_blend_shape_name() for this shape

Methods

Name	Description
to_string()	Returns a string with blendshape options settings
get_option_category()	Returns this option’s category
get_default_value()	Returns this option’s default value
get_random_value()	Returns a random float value between min and max
get_mesh_paths()	Returns this option’s mesh paths
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s blendshapes to the character preview
apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option’s blendshape value to the character
get_editor_groups()	Returns this option’s editor groups

create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates the blendshape slider

bone_target.gd

bone_target Script

Overview

Stores information for bone deformations. Used in conjunction with deform_option. Holds information for single bone deformation.

Variables

Name	Type	Description
bone_name	String	The bone that is being deformed. Must match name in Skeleton hierarchy.
Export group: Positive Delta		Final transform for bone deformation. Applied at slider max (1.0). Used whether deform type is single or bidirectional.
max_position_offset	Vector3	Position offset applied when slider is at max (1.0).
max_rotation_degrees	Vector3	Rotation offset in degrees applied when slider is at max (1.0).
max_scale_multiplier	Vector3	Scale multiplier applied when slider is at max (1.0).
Export group: Negative Delta		Final transform for bone deformation. Applied at slider min (-1.0). Only used when deform type is bidirectional.
max_position_offset	Vector3	Position offset applied when slider is at min (-1.0).
max_rotation_degrees	Vector3	Rotation offset in degrees applied when slider is at min (-1.0).
max_scale_multiplier	Vector3	Scale multiplier applied when slider is at min (-1.0).

color_option.gd

color_option Script

Overview

Drives a color parameter on a material, producing a ColorPickerButton. Handles both ShaderMaterial custom uniforms and StandardMaterial3D built-in properties

Variables

Name	Type	Description
mesh_paths	Array[NodePath]	Array of NodePath to the MeshInstance3D whose material contains this parameter.
surface_index	int	Surface index on the mesh to target
shader_param	String	For ShaderMaterial: the uniform name as declared in the shader. For StandardMaterial3D: one of the recognised property names below
default_color	Color	The default color to use
apply_to_shared_material	Bool	Whether to update the material on any other MeshInstance3D nodes that share the same material resource
editor_groups	Array[String]	Tracks all the mesh groups this shape appears in, used by the editor dock only (not read at runtime).

Methods

Name	Description
to_string()	Returns a string with color options settings
get_option_category()	Returns this option’s category

get_default_value()	Returns this option’s default value
get_random_value()	Returns a random color value
get_mesh_paths()	Returns this option’s mesh paths
get_surface_index()	Returns this option’s surface index
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s color to the character preview
apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option’s color value to the character
get_editor_groups()	Returns this option’s editor groups
create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates the color slider

deform_modifier_3d.gd

deform_modifier_3d Script

Overview

SkeletonModifier3D that applies DeformOption resources to a Skeleton3D.

Variables

Name	Type	Description
_active_deforms	Dictionary[DeformOption, float]	Maps each active DeformOption to current deformation value.

Methods

Name	Description
set_deform_value(opt: DeformOption, value: float)	Sets current value of DeformOption. Called by CreatorManager.
_process_modification()	Called by SkeletonModifier3D during modification. Applies every active DeformOption in _active_deforms to Skeleton3D.

deform_option.gd

deform_option Script

Overview

Drives the deform options. Used for bone deformations. Creates one slider in character creator for all its bone targets.

Enums

DeformType
{
	SINGLE,
	BIDIRECTIONAL
}

Variables

Name	Type	Description
deform_type	DeformType	SINGLE: deform is driven by one transform (slider range: 0.0-1.0) BIDIRECTIONAL: deform is driven by two transforms (slider range: −1.0-1.0)
skeleton_path	NodePath	NodePath to the Skeleton node. Local path in character scene uploaded to dock.
bones	Array[BoneTarget]	Array that holds all the bones (BoneTargets) that this DeformOption affects.
default_value	float	The default slider value of this deform option
min_value	float	The min value threshold of the deform option
max_value	float	The max value threshold of the deform option

Methods

Name	Description

get_min_value()	Returns the min float value for this deform option (SINGLE: 0.0, BIDIRECTIONAL: -1.0)
get_max_value()	Returns the max float value for this deform option (Always 1.0)
apply()	For each bone target, call _lerp function and set bone pose
_lerp(rest: Transform3D, pos: Vector3, rot_deg: Vector3, scale_mult: Vector3, t: float) → Transform3D:	Interpolates a bone's rest transform toward the given deltas by factor [param t].
get_option_category()	Returns this option’s category
get_default_value()	Returns this option’s default value
get_random_value()	Returns a random float value between min and max values
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s deform to the character preview
apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option’s deform value to the character
get_editor_groups()	Returns this option’s editor groups
create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates the deform option
apply(skeleton: Skeleton3D, t: float)	Applies the bone deformations to the bone targets of the skeleton

mesh_swap_choice.gd

mesh_swap_choice Script

Overview

A single selectable choice owned by a [MeshSwapOption].

Variables

Name	Type	Description
include	Bool	Whether or not this option should be included as a selectable parameter in the UI
label	String	Label shown on the generated button
icon	Texture2D	Icon to display on mesh swap choice button
mesh_path	NodePath	NodePath to the MeshInstance3D for this choice
file_path	String	Stores the string path for this mesh swap's file. Loaded in at runtime when selected.

Methods

Name	Description
to_string()	Returns a string with the choice options settings

mesh_swap_option.gd

mesh_swap_option Script

Overview

Swaps between meshes by loading and unloading scenes as skeleton children. Used with [MeshSwapChoice].

Variables

Name	Type	Description
required	Bool	Whether at least one option must be selected, or this can be set as "None"
choices	Array[MeshSwapChoice]	The individual choices
default_choice	int	Index of the choice shown by default

Methods

Name	Description
to_string()	Returns a string with the mesh swap options settings
get_option_category()	Returns this option’s category
get_default_value()	Returns this option’s default value
get_random_value()	Returns random choice value
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s mesh swap to the character preview
get_editor_groups()	Returns this option’s editor groups
create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates this mesh swap’s option

option_definition.gd

option_definition Script

Overview

"Abstract" parent of Option classes: AnimationOption, BlendshapeOption, ColorOption, DeformOption, MeshSwapOption, TextureAtlasOption.

Variables

Name	Type	Description
display_name	String	Human-readable label shown in the generated UI
icon_to_display	Texture2D	Icon selected for option to be shown in generated UI
include	Bool	Whether this Option appears in the generated UI
group	String	Which tab this option appears under in the TabContainer

Methods

Name	Description
to_string()	Returns a string with the option settings
get_option_category()	Returns this option’s category
get_default_value()	Returns this option’s default value
get_random_value()	A method for child options to overwrite
get_mesh_paths()	Returns this option’s mesh paths
get_surface_index()	Returns this option’s surface index
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option’s value to the character preview
apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option’s value to the character
get_editor_groups()	Returns this option’s editor groups

create_editor_rows(content_vbox: VBoxContainer, group_ui: GroupContainer, group_name: String, group_has_atlas: bool, active_ui_controls: Array[Control], group_scene: PackedScene, row_scene: PackedScene)	Creates this option’s editor slider, button, etc.

texture_atlas_option.gd

texture_atlas_option Script

Overview

Drives texture atlas selection with buttons. Used by offsetting UV coordinates on target material.

Variables

Name	Type	Description
mesh_paths	Array[NodePath]	Array of NodePaths to the MeshInstance3D nodes that this option controls.
surface_index	int	Surface index on the mesh to target
columns	int	Number of options in a horizontal row
rows	int	Number of rows for options
choice_labels	Array[String]	The labels for the UI buttons
required	bool	If true, at least one option must be selected
default_choice	int	The default texture atlas
apply_to_shared_material	Bool	If true, modifying this material alters all meshes sharing it
choice_icons	Array[Texture2D]	Icons shown on each choice button, displayed in the generated scene.
editor_groups	Array[String]	Tracks all the mesh groups this option appears in, not read at runtime.
shader_param	String	For ShaderMaterial: the uniform name for UV offset (usually a vec2 or vec3) For StandardMaterial3D: leave blank, it will automatically use uv1_offset

Methods

Name	Description
to_string()	Returns a string with the Texture atlas options settings
get_option_category()	Returns this option's category
get_default_value()	Returns this option's default value
get_random_value()	Returns a random value for this texture atlas
get_mesh_paths()	Returns this option's mesh paths
get_surface_index()	Returns this option's surface index
apply_to_preview(manager: CreatorManager, value: Variant, force_full_pass: bool = false)	Tells the Creator Manager to apply this option's texture atlas to the character preview
get_editor_groups()	Returns this option's editor groups
create_editor_rows(content_vbox, group_ui, group_name, group_has_atlas, active_ui_controls, group_scene, row_scene)	Creates the texture atlas option
apply_to_character(character_root: Node, skeleton: Node, value: Variant)	Applies this option's texture atlas to the character

character_config.gd

character_config Script

Overview

Source of truth for character creator configuration: which mesh scene to use, which options to expose, and how the runtime scene should behave.

Variables

Name	Type	Description
character_scene	PackedScene	The character scene to instantiate inside CharacterPreview
output_path	String	Output path for the generated scene and exported meshes.
options	Array[OptionDefinition]	All options the developer has chosen to expose to the player
skeleton_path	String	Path to the character's skeleton rig
allow_randomize	Bool	Whether or not to expose a Randomize button in the generated scene footer
save_state_on_confirm	Bool	Emit a CharacterState .tres file when the player confirms
state_save_path	String	Where to write the CharacterState .tres on confirm
preview_camera_distance	float	Distance of the preview camera from the character origin
preview_camera_height	float	Vertical offset of the preview camera's look-at target
ui_panel_width	float	Fraction of screen width the UI panel occupies
allow_preview_orbit	Bool	Whether the player can orbit the preview camera

theme_resource	Theme	Path for the theme resource
skybox_resource	ShaderMaterial	Path for the skybox resource

Methods

Name	Description
to_string()	Returns a string with a summary of Character config settings

Example

None

```
[gd_resource type="CharacterConfig" format=3]

[sub_resource type="BlendshapeOption" id="BlendshapeOption_1"]
display_name = "face_fat"
group = "Face"
mesh_path = NodePath("Character/Body")
blend_shape_name = "face_fat"
default_value = 0.0
min_value = 0.0
max_value = 1.0

[sub_resource type="MeshSwapOption" id="MeshSwapOption_2"]
display_name = "Hair"
group = "Hair"
...

[resource]
character_scene = ExtResource("hero_character.tscn")
options = [SubResource("BlendshapeOption_1"), SubResource("MeshSwapOption_2"),
...]
show_preview = true
allow_randomize = true
preview_camera_distance = 2.0
preview_camera_height = 0.8
```

character_load_row.gd

character_load_row Script

Overview

Extends Button,
The script all character load buttons have. It holds a variable for state which is then emitted via signal.

Variables

Name	Type	Description
no_icon	String	A placeholder Image, will be used if no image is found within the character state
state	CharacterState	The character state corresponding to this button

Methods

Name	Description
init(name:String,preview:Image,char_state:CharacterState)	Initializes all variables, sets the text and image of the button.
_on_pressed()	Emits the character state

character_state.gd

character_state Script

Overview

Saves all player's character creator selections. Holds no references to nodes, meshes, or materials. Can be applied to any character mesh with the same option names.

Variables

Name	Type	Description
values	Dictionary[String, Variant]	Key = option resource_name, value = choice
last_modified	int	Timestamp of the last modification. Useful for save slot UIs
active_meshes	Array[String]	All mesh scenes on character. Used by Character_Applicator for fast lookup of values.
icon	Image	Icon for the load tab
metadata	Dictionary	Arbitrary developer-defined metadata. Use this to attach things like character name, class, or faction without subclassing CharacterState

Methods

Name	Description
from_config(config: CharacterConfig)	Reads the options from the character config and returns a new Character state
record(option_id: String, value: Variant)	Writes a value into the correct dictionary based on its type. Called by CharacterExporter on every UI interaction to keep state in sync.
randomized(config: CharacterConfig)	Generates new values for each OptionDefinition, with constraints (include flag and value range).

Editor

Docks

configura_dock.gd

configura_dock Script

Overview

Phase 1: Mesh loading
- When the developer picks a file, run MeshInspector and populate the options list

Phase 2: Options list
- Each detected OptionDefinition gets its own OptionRow scene when building

Phase 3: Character Config
- Populates the character config with data from options list

Variables

Name	Type	Description
scene_generated	Signal	Which scene is generated from path
GroupContainerScene		Reference to the group_container scene
OptionRowScene		Reference to the option_row scene
_splitter	Regex	Used to split strings with a delimiter
_character_resource_picker	Container	Allows the user to select the character resource
_character_scene_container	HBoxContainer	Container for the selected character scene
_theme_resource_picker	Container	Allows the user to select the theme resource
_skybox_resource_picker	Container	Allows the user to select the skybox resource
_status_label	Inspector node: Label	Status Label
_categories_list	Inspector node: VBoxContainer	Categories list
_allow_random	Inspector node: Checkbox	Allow randomize
_save_state	Inspector node: Checkbox	Save state
_output_path	Inspector node: LineEdit	Output path

_output_dialog	Inspector node: FileDialog	Output dialog
_generate_btn	Inspector node: Button	Generate button
_bone_deforms_container	Foldable Container	Bone deformation controls for skeleton
_deform_list	FileDialog	One row per active DeformOption
_options_context	Label	Options Context
_current_character_scene	PackedScene	Current character scene
_detected_options	Array[OptionDefinition]	Array to hold detected options
_current_theme_resource	Theme	Theme resource
_current_skybox_shader	ShaderMaterial	Skybox shader resource
_active_ui_controls	Array[Control]	Array of controls
_base_model_file_path	String	Path to where the base model scene is located
_skeleton_path	String	Path to the character's skeleton rig
_deform_options	Array[DeformOption]	Current DeformOption resources for character

Methods

Name	**Description**
_ready()	On ready, connects resource signals
_on_theme_resource_selected()	When a theme resource is selected
_on_skybox_resource_selected()	When a skybox resource is selected
_on_input_resource_selected()	When the input resource button is selected
get_persistent_state()	saving and loading dock info
apply_persistent_state(state: Dictionary)	loading of saved information
_find_group_containers(root: Node)	helper for applying the states in each group

_apply_header_states(headers: Array)	apply header options
_on_confirm_button_pressed()	Reads the selected file and output path. Extracts the mesh assets and builds the base model scene. Calls to populate options list
_create_meshes_folder()	Helper method that checks if a meshes folder exists in the user provided output_path. Creates one if it doesn't
_rebuild_options_list()	Builds the options list UI by clearing the old UI, grouping all detected options by ‘group’ strings, and rebuilding the UI hierarchy dynamically
_build_base_scene(character_tscn: PackedScene)	Builds the base model scene
_find_skeleton_node(root: Node3D, base_scene_root: Node3D)	Helper method that recursively searches the character scene for the Skeleton3D node. Duplicates it for the base model scene and finds it's "metarig" parent node
_extract_meshes(character_tscn: PackedScene)	Extracts accessory and clothing meshes to their own separate scenes
_save_Node(root: Node, out_path: String)	Creates and saves a node scene to the out_path
_save_Mesh(root: MeshInstance3D, out_path: String)	Creates and saves a MeshInstance3D scene to the out_path
_on_add_deform_button_pressed()	Creates new DeformOption on button press.
_build_deform_row(opt: DeformOption, index: int)	Builds a single deform row with edit and remove buttons.
_on_output_button_pressed()	When the output button is pressed
_on_output_dialog_dir_selected(dir)	Reads the selected file and sets the output path text
_on_clear_output_button_pressed()	Clears the output path text
_on_generate_button_pressed()	On generate button pressed, the dock gathers the state of all rows into a CharacterConfig resource, then hands it to SceneGenerator

_slugify(opt: OptionDefinition)	Helper method to assign resource_name for options. Cleans up strings for safe dictionary keys, generates unique IDs based on the specific subclass

group_container.gd

group_container Script

Overview

In charge of setting up a group container's row of options.

Variables

Name	Type	Description
_include_group	Inspector node: Checkbox	Whether this group should appear in character creator
_group_field	Inspector node: LineEdit	Group name
_required_row	HBoxContainer	Container for required checkbox
_required	Inspector node: Checkbox	Whether this group is required (or can be set to None)
_default_row	HBoxContainer	Container for default options
_default_options	OptionButton	Lists all options in group for one to be selected as default
_option_row_containter	VBoxContainer	Option row container
_owned_rows	Array[OptionRow]	Keeps track of the rows of options in this group's container
_source_option	OptionDefinition	The source of the option
_curr_required	bool	Keeps track of whether required is set or not
_curr_default	int	Keeps track of which default index is set

Methods

Name	Description
setup(opt: OptionDefinition)	Set up this group's container. Accepts MeshSwapOption or TextureAtlasOption

register_rows(rows: Array[OptionRow])	Register rows option to _owned_rows
_set_rows_visible(visible: bool)	Sets the rows of options to visible
add_option_rows(row: OptionRow)	Adds OptionRow as child of option row container
get_config_option()	Returns MeshSwapOption or TextureAtlasOption depending on _source_option type
_harvest_rows()	Iterates through the _owned_rows array and sets up default choices
_get_swap_option()	Get mesh swap option
_get_atlas_option()	Get texture atlas option
_on_include_group_toggled(toggled_on: bool)	Toggles row visibility of group on include
_on_default_options_item_selected(index: int)	Tracks currently selected default option index
_on_required_toggled(toggled_on: bool)	Rebuilds default options dropdown to add/remove “None” based on whether Required is toggled or not
add_default_option(label: String)	Adds new entry to default options dropdown button
get_excluded_option()	Returns source option marked as excluded for groups left unchecked
get_group_state()	saving and loading of group containers
apply_group_state(state: Dictionary)	Applies saved group state, returned by get_group_state.

option_row.gd

option_row Script

Overview

In charge of setting up an option

Events/Listeners

Name	Type	Description
_display_name_changed(new_name: String)	Signal	Tells default dropdown when display name is edited to update dynamically

Variables

Name	Type	Description
_include_group	Inspector node: Checkbox	Include group
_display_name	Inspector node: LineEdit	Display name
_icon_container	Inspector node: PanelContainer	Input container
_icon_label	Inspector node: Label	Icon label
_color_picker	ColorPickerButton	Color picker
_icon_resource_container	ResourcePicker	Icon Resource Container
_mesh_path	Nodepath	Node path to this option's mesh
_file_path	String	String path to load this option's mesh

Methods

Name	Description
_ready()	On ready, set icon label and container visibility
setup(option: OptionDefinition)	Set up this row's option
setup_choice(choice: MeshSwapChoice, group: String)	Used by GroupContainer for individual swap choices

setup_atlas_choice(label: String, is_default: bool)	Atlas variant of setup_choice: no mesh path needed, badge reads "texture"
get_choice()	Returns this option’s Mesh swap choice
get_row_state()	saving and loading of row options
apply_row_state(state: Dictionary)	Restores this option row’s saved UI state from get_row_state.

📂 Generators

animation_tree_builder.gd

Animation_tree_builder Script

Overview

Builds the animation tree.

Variables

Name	Type	Description

Methods

Name	Description
build(character_root: Node, config: CharacterConfig, scene_root: Node)	Entry point to building the animation tree. Called by SceneGenerator after the character scene has been instantiated inside the SubViewport. Returns the built AnimationTree node, which SceneGenerator adds to the SubViewport and sets .owner on. Also mutates each MorphOption.tree_node_name and AnimationOption.tree_node_name in config.options so CharacterExporter has correct parameter paths at runtime.
_parse_animation_library(player: AnimationPlayer)	Parses the data from AnimationPlayer
_find_animation_player(root: Node)	Finds the Animation player
sanitize(name: String)	Cleans up the name of the animation tree
_sanitize(name: String)	Calls sanitize on AnimationTreeBuilder
_stamp_anim_option(config: CharacterConfig, display_name: String, node_name: String)	Write the resolved blend tree node name back onto the matching AnimationOption

load_generator.gd

load_generator Script

Overview

Helper class that generates the base load scene

Methods

Name	Description
build(root: VBoxContainer,scene_root: Node,theme: Theme)	Generates the base load control nodes.

scene_generator.gd

scene_generator Script

Overview

Acts as a bridge between the editor and runtime. It takes a finalized CharacterConfig and writes a fully-wired .tscn file to disk that the developer can drop into their project

Variables

Name	Type	Description

Methods

Name	Description
generate(config: CharacterConfig, out_path: String)	Generates the character scene from the CharacterConfig. Root node is constructed entirely in memory until _save() packs and writes it
_build_base_tree(config: CharacterConfig)	Creates the four runtime nodes and configures each from config 1. Create the root control of the character creator scene 2. Create the Character Creator UI 3. Create the CharacterPreview node (SubViewportContainer + Internals) 4. Create the CharacterExporter node
_generate_ui(root: Node, config: CharacterConfig)	Delegates to UIGenerator to build the tab/group structure and individual control nodes
_generate_load(root: Node)	Generates the Character Loader UI
_save(root: Node, config: CharacterConfig, out_path: String)	Serializes the CharacterCreator scene as a PackedScene. Save the config .tres file as well.
_validate(config: CharacterConfig)	Validates the config file before any construction begins
_unwrap_and_bind(node: Node, scene_root: Node, ignore_node: Node)	Recursively sets node owners to the scene_root and clears their scene_file_path for optimization

_set_owners(node: Node, scene_root: Node)	Walks a subtree and sets .owner on every node that doesn't already have one

ui_generator.gd

ui_generator Script

Overview

Runs exactly once when the developer clicks "Generate Scene" in the dock. Its job is to write a node tree into a .tscn file

Variables

Name	Type	Description
SliderRowScene		Control scene templates preloaded once at class level
SwapGroupScene		Control scene templates preloaded once at class level
ColorRowScene		Control scene templates preloaded once at class level
ANIM_ROW_SCENE		Control scene templates preloaded once at class level

Methods

Name	Description
build(ui: VBoxContainer, options: Array[OptionDefinition], scene_root: Node, theme: Theme, randomize_option: bool)	Builds the UI with the finalized options, UI container box, and CharacterCreator root node
_build_tabs(ui: VBoxContainer, options: Array[OptionDefinition], scene_root: Node)	Builds the tabs for grouped attributes
_build_tab(tabs: TabContainer, group_name: String, options: Array, #[OptionDefinition], scene_root: Node)	Builds an individual tab
_set_owners(node: Node, scene_root: Node)	Walks a subtree and sets .owner on every node

_build_control(opt: OptionDefinition, scene_root: Node)	Builds the control for each OptionDefinition subtype
_build_slider_row(opt: BlendshapeOption)	Builds a slider option for blendshape
_build_swap_group(opt: MeshSwapOption, scene_root: Node)	Builds buttons for mesh swap options
_build_color_row(opt: ColorOption)	Builds a color option
_build_anim_row(opt: AnimationOption)	Builds an animation option
_build_deform_row(opt: DeformOption)	Builds slider for bone deformations per DeformOption
_build_atlas_group(opt: TextureAtlasOption, scene_root: Node)	Builds buttons for texture atlas options
_build_footer(ui: VBoxContainer, scene_root: Node)	Builds the footer section

📂 Inspectors

mesh_inspector.gd

mesh_inspector Script

Overview

Pure, stateless utility that takes a scene, walks the node tree and returns an Array[OptionDefinition]

Variables

Name	Type	Description
_splitter	Regex	Groups all meshes by the first token of their name, split on _ . , -
_atlas_regex	Regex	Matches the suffix _Atlas_<rows>_<cols> at the end of a material name

Methods

Name	Description
inspect(baseModel_packed_scene: PackedScene, orig_packed_scene: PackedScene)	Inspects the base model scene and original packed scene and instantiates it into a temporary node that lives only for the duration of the call. Calls to inspect swap slots, blendshapes and materials on meshes from both scenes
_inspect_swap_slots(meshes: Array[MeshInstance3D], root: Node)	Inspects mesh swaps. Instantiates MeshSwapChoice(s) and MeshSwapGroup(s)
_filter_meshes(root: Node)	Filter meshes that are not the base model meshes
_find_model_meshes(root: Node)	Finds only the base model meshes. Root node should be the root node of the base model scene
_inspect_blendshapes(mesh_node: MeshInstance3D, root: Node, global_seen: Dictionary, results: Array[OptionDefinition])	Inspects the mesh's blendshapes. If blendshape already exists, just add mesh to the result. Else, add the new blendshape.
_inspect_materials(mesh_node: MeshInstance3D, root: Node, global_colors:	Walks every surface on every MeshInstance3D and looks for color-type

Dictionary, global_atlases: Dictionary)	shader parameters on ShaderMaterial, falling back to standard albedo/emission on StandardMaterial3D. Also creates an option for texture atlas if found.
_inspect_pose_animations(root: Node)	Inspects the animation player and creates an animation option for each animation

Tool Scripts

resource_picker.gd

resource_picker Script

Overview

Creates resource picker for character creator dock

Variables

Name	Type	Description
picker_type	String	Type of resources that this picker selects
min_size	Vector2	Minimum size of UI
picker	EditorResourcePicker	This resource picker
selected_resource	Resource	Resource selected by this picker

Methods

Name	Description
_ready()	On ready, creates EditorResourcePicker with picker type and signal to emit when resource is selected

Runtime

Controls

anim_row.gd

anim_row Script

Overview

The control for animation.

Events/Listeners

Name	Type	Description
changed(option_id: String, value: Variant)	Signal	On changed delegate

Variables

Name	Type	Description
option_id	String	Option ID
animation_name	String	Animation name
_label	Inspector node: Label	Option label
_button	Inspector node: Button	Preview button

Methods

Name	Description
_ready()	On ready, connect the button’s functionality
_on_pressed()	Emit the change on button press
set_pressed_no_signal(_active: bool)	no state to restore for anim rows (preview-only)

color_row.gd

color_row Script

Overview

The control for color.

Events/Listeners

Name	Type	Description
changed(option_id: String, value: Variant)	Signal	On changed delegate

Variables

Name	Type	Description
option_id	String	Option ID
_label	Inspector node: Label	Option label
_picker	Inspector node: ColorPickerButton	Color Picker
_applying	Bool	A flag variable to block emission

Methods

Name	Description
_ready()	On ready, connect the button's functionality
_on_color_changed(color: Color)	Emit the change on button press
set_color_no_signal(color: Color)	ColorPickerButton has no set_color_no_signal. Block emission manually with a flag.
_on_color_changed_guarded(color: Color)	Emit the change when if _applying is false

slider_row.gd

slider_row Script

Overview

The control for a slider.

Events/Listeners

Name	Type	Description
changed(option_id: String, value: Variant)	Signal	On changed delegate

Variables

Name	Type	Description
option_id	String	Option ID
_label	Inspector node: Label	Option label
_slider	Inspector node: HSlider	Slider
_readout	Inspector node: Label	Readout

Methods

Name	Description
_ready()	On ready, connect the slider’s functionality
_on_value_changed(v: float)	On value change, read out the change and emit the change
set_value_no_signal(v: float)	Sets the Range's current value to the specified value, without emitting the value_changed signal

swap_group.gd

swap_group Script

Overview

Swap options have a variable number of choices. So the scene contains a fixed header and an empty container that UIGenerator populates.

Events/Listeners

Name	Type	Description
changed(option_id: String, value: Variant)	Signal	On changed delegate

Variables

Name	Type	Description
option_id	String	Option ID
_container	Inspector node: HFlowContainer	Button Container
_btn_group	ButtonGroup	A button group container
btn	Button	Buttons added to SwapGroup by UIGenerator

Methods

Name	Description
_ready()	On ready, connect the buttons' functionality
_on_button_toggled(active: bool, idx: int)	Emit the swap change
set_choice_no_signal(idx: int)	Changes the button_pressed state of the button, without emitting toggled. Use when you just want to change the state of the button without sending the pressed event

creator_controller.gd

creator_controller Script

Overview

Drops into a project where character creation happens. Wires its children together and exposes a two-signal public API to the rest of the game.

Enums

```
CompatResult
{
	COMPAT_FULL,
	COMPAT_PARTIAL,
	COMPAT_NONE
}
```

Events/Listeners

Name	Type	Description
character_confirmed(state: CharacterState)	Signal	Emitted when the player presses confirm. The developer connects to this and received the final CharacterState
character_cancelled()	Signal	Emitted when the player presses Cancel

Variables

Name	Type	Description
config	CharacterConfig	
ui	CreatorUI	
manager	CharacterManager	
preview	CharacterPreview	
loader	CharacterLoader	

Methods

Name	Description
_ready()	Where all the connections are made. UI changes applies to mesh immediately. Seeds the exporter with defaults and sync the UI to match. If a previous save exists and should be restored, load it now
_on_save()	Create a file dialogue pop-up for the user the save the character to their local data
get_file_count(path)	Opens file folder from config folder and returns how many files are within that directory
_on_toggle()	Toggle the load menu on and off
_on_load()	Create a file dialogue pop-up for the user to load a locally saved character
_on_randomize()	Randomize the character config
_try_restore_saved_state()	Checks for a previously saved state and restores it so returning players see their last character
_validate_state_compatibility(state: CharacterState)	Checks whether the saved state's keys still exist in the current config

character_loader.gd

character_loader Script

Overview

The script that handles everything pertaining to the LoaderUI control nodes within the constructed character creator scene. n

Events/Listeners

Name	Type	Description
back_pressed	Signal	Calls the on_toggle() function in character creator
character_selected(state:CharacterState)	Signal	Emits the selected character when a character_load_row is pressed.

Variables

Name	Type	Description
optionsList	VboxContainer	Options list
config	CharacterConfig	The character creator config

Methods

Name	Description
_ready()	Connects the signals and inits any required variables
load_saved_characters()	Opens the file directory specified in the config variable, will then make a character_load_row for each file it finds.
clearOptions()	Clears all children of optionsList
make_placeholder()	Makes placeholder panel if no characters are found.

creator_manager.gd

creator_manager Script

Overview

Serializes CharacterState and applies it to the live mesh. Translating abstract option changes into concrete mesh operations and maintaining the running CharacterState.

Variables

Name	Type	Description
_splitter	Regex	Used to split strings with a delimiter
config	CharacterConfig	
_current_state	CharacterState	
_character_root	Node	
_anim_tree	AnimationTree	
_ui	Control	
_option_map	Dictionary[String, OptionDefinition]	Key: resource_name Value: OptionDefinition
_mesh_cache	Dictionary[NodePath, MeshInstance3D]	
_blend_idx_cache	Dictionary[String, int]	
_player_cache	Dictionary[NodePath, AnimationPlayer]	
_mat_cache	Dictionary[String, Material]	
_skeleton_cache	Dictionary[Nodepath, Skeleton3D]	
_modifier_cache	Dictionary[NodePath, DeformModifier3D]	
_active_accessories	Dictionary[String, MeshInstance3D]	Keeps track of currently active accessories
_cur_mesh	MeshInstance3D	Stores the currently loaded mesh
_character_skeleton	Skeleton3D	A reference to the character’s

		skeleton rig

Methods

Name	Description
initialize(config: CharacterConfig, preview: CharacterPreview, ui: Control)	Called once by CharacterCreator._ready() before any player input arrives Does 3 things: - Builds the option map - Resolves and caches every node the apply methods will need - Applies defaults to the live mesh so the character appears correct from frame one
_warm_cache(opt: OptionDefinition)	Caches data from option for all subtypes
apply_option(option_id: String, value: Variant)	Exporter received every slider move, button press, and color pick input.
_apply_swap(opt: MeshSwapOption, choice_index: int, force_full_pass: bool = false)	Dynamically loads and unloads meshes when selected by player
apply_current_option_values(option_lists: Array[VBoxContainer], cur_group: String, key: NodePath)	Applies all current option values (blendshapes, atlas, colors) to the current newly loaded mesh
_create_opt_group_name(cur_group: String, opt_parts: Array[String])	Helper method to concatenate option group name parts together
apply_current_blendshape_value(blendshape_name: String)	Applies the current blendshape values to the newly loaded mesh
_apply_blendshape(opt: BlendshapeOption, value: float)	Caches the blend index so when the slider is dragged, the only work that is done is a dictionary lookup
apply_current_color_value(color_name: String)	Applies the current color values to the newly loaded mesh
_apply_color(opt: ColorOption, color: Color)	Applies color to the material(s) on the mesh
_write_color(mat: Material, param: String, color: Color)	Writes the color to the material shader
_apply_animation(opt: AnimationOption)	Applies animation to the character’s animation player

_apply_deform(opt: DeformOption, value: float)	Applies bone deformation to mesh
_apply_texture_atlas(opt: TextureAtlasOption, choice_index: int)	Similar to _apply_color() method. Applies the texture atlas to the material shader
_apply_full_state(state: CharacterState)	When a CharacterState is loaded from disk and needs to be applied all at once. Ex: On session restore, after randomization, or when showing a preset
load_state(state: CharacterState)	Loads the CharacterState (full)
load_state_partial(state: CharacterState)	Loads the CharacterState (partial). When CharacterCreator._validate_state_compatibility() returns COMPAT_PARTIAL
get_current_state()	Gets the current CharacterState

character_preview.gd

character_preview Script

Overview

Sets up the camera controls for the character preview.

The `SubViewport` needs a handful of specific settings to behave correctly inside a UI panel. These are set in the `.tscn` rather than in script so they serialize cleanly and are visible in the editor inspector:

```python
# An isolated world keeps the preview self-contained regardless of what scene the developer embeds the creator into.
	own_world_3d:			true
# Lets the SubViewportContainer's own background show through where there is no geometry. This means the developer can style the preview area with a StyleBox on the container (ex: A dark panel, a gradient, a patterned background)
	transparent_bg:			true
# Forwards mouse events up to the main viewport
	handle_input_locally:	false
	size:					Vector2i(400, 600)
# Allows the SubViewport to render at a fixed internal resolution while the container scales
	size_2d_override_stretch: true
# Anti-aliasing
	msaa_3d:				MSAA_4X
```

Variables

Name	Type	Description
rotation_speed	float	Camera rotation speed
zoom_speed	float	Camera zoom speed
min_zoom	float	Camera min zoom threshold
max_zoom	float	Camera max zoom threshold
pan_speed	float	Camera pan speed

min_height	float	Camera min height threshold
max_height	float	Camera max height threshold
_zoom	float	Current camera zoom amount
_viewport	SubViewport	
_rig	Node3D	CameraRig
_pivot	Node3D	CameraPivot
_camera	Camera3D	Camera3D
_character	Node	Character
_world	WorldEnvironment	World environment
_dragging	Bool	If mouse input is dragging

Methods

Name	**Description**
_add_skybox_shader(shader: ShaderMaterial)	Set up the skybox
_ready()	Set up the camera rig
_input(event)	Handles mouse input events
update_camera_distance()	Updates camera zoom distance
get_character_root()	Gets the character root node

creator_ui.gd

creator_ui Script

Overview

CreatorUI is the generated control tree the player actually interacts with. It doesn't know what a blendshape is or how a mesh swap works. It only knows how to build, display, and emit changes from controls. The mapping between a control's value and a mesh operation lives entirely in CharacterExporter. CreatorUI's contract is simple: receive OptionDefinition resources, produce controls, emit signals when those controls change.

Events/Listeners

Name	Type	Description
option_changed(option_id: String, value: Variant)	Signal	Emitted whenever any control value changes
save_pressed()	Signal	Emitted when “Save” button pressed
load_pressed()	Signal	Emitted when “Load” button pressed
randomize_pressed()	Signal	Emitted when “Randomize” button pressed

Variables

Name	Type	Description
_control_map	Dictionary	map of Key: option_id Value: Control node (SwapGroup, SliderRow, ColorRow, AnimRow) Used by apply_state() to sync controls without re-emitting signals

Methods

Name	Description
_ready()	Bind the ui controls

_bind_controls(node: Node)	Recursively searches the tree for custom Option control nodes and sets up the connectivity
_bind_footer_buttons()	Finds the standard footer buttons and wires them up
hide_randomize_button()	Hide randomize button
apply_state(state: CharacterState)	Sync controls to a loaded state without triggering signals

character_state_applier.gd

character_state_applier Script

Overview

Applying state outside the creator scene to a character mesh elsewhere in the game. Ex: Somewhere in the game scene that spawns the player character.

Variables

Name	Type	Description

Methods

Name	Description
apply(state: CharacterState, config: CharacterConfig, character_root: Node)	Applies the options from the character config and character state to the character in the character root node.

configura_plugin.gd

configura_plugin Script

Overview

The actual plugin script.

Variables

Name	Type	Description
DOCK_SCENE		Configurar_dock scene
NAME	String	
_dock	ConfiguraDock	

Methods

Name	Description
_enable_plugin()	Place to add any autoloads
_disable_plugin()	Place to remove autoloads
_enter_tree()	Instantiate and add the dock to the editor UI and add any resource dependencies
_get_window_layout(configuration: ConfigFile)	Saves dock's persistent UI state to editor's window layout configuration.
_set_window_layout(configuration: ConfigFile)	Restores dock's persistent UI state from editor's window layout configuration.
_exit_tree()	Clean up plugin logic here