Skip to content

BoneCollections(bpy_struct)

base class — `bpy_struct`

class bpy.types.**BoneCollections(bpy_struct)**

The Bone Collections of this Armature

active

Armature's active bone collection

TYPE:

`BoneCollection`

active_index

The index of the Armature's active bone collection; -1 when there is no active collection. Note that this is indexing the underlying array of bon collections, which may not be in the order you expect. Root collections are listed first, and siblings are always sequential. Apart from that, bor collections can be in any order, and thus incrementing or decrementing this index can make the active bone collection jump around in unexpected ways. For a more predictable interface, use `active` or `active_name`.

TYPE:

int in [-inf, inf], default 0

active_name

The name of the Armature's active bone collection; empty when there is no active collection

TYPE:

string, default "", (never None)

is_solo_active

Read-only flag that indicates there is at least one bone collection marked as 'solo'

TYPE:

boolean, default False, (readonly)

new(name, *, parent=None)

Add a new empty bone collection to the armature

PARAMETERS:

- **name** (*string, (never None)*) – Name, Name of the new collection. Blender will ensure it is unique within the collections of the Armatur
- **parent** (`BoneCollection`, (optional)) – Parent Collection, If not None, the new bone collection becomes a child of this collectio

RETURNS:

Newly created bone collection

RETURN TYPE:

`BoneCollection`

remove(bone_collection)

Remove the bone collection from the armature. If this bone collection has any children, they will be reassigned to their grandparent; in other words, the children will take the place of the removed bone collection.

PARAMETERS:

bone_collection (`BoneCollection`) – Bone Collection, The bone collection to remove

move(from_index, to_index)

Move a bone collection to a different position in the collection list. This can only be used to reorder siblings, and not to change parent-child relationships.

PARAMETERS:

- **from_index** (*int in [-inf, inf]*) – From Index, Index to move
- **to_index** (*int in [-inf, inf]*) – To Index, Target index

classmethod bl_rna_get_subclass(id, default=None)

PARAMETERS:

id (*str*) – The RNA type identifier.

RETURNS:

The RNA type or default when not found.

RETURN TYPE:

`bpy.types.Struct` subclass

classmethod bl_rna_get_subclass_py(id, default=None)

PARAMETERS:

id (*str*) – The RNA type identifier.

RETURNS:

The class or default when not found.

RETURN TYPE:

type

Inherited Properties

- `bpy_struct.id_data`

Inherited Functions

- `bpy_struct.as_pointer`
- `bpy_struct.driver_add`
- `bpy_struct.driver_remove`
- `bpy_struct.get`
- `bpy_struct.id_properties_clear`
- `bpy_struct.id_properties_ensure`
- `bpy_struct.id_properties_ui`
- `bpy_struct.is_property_hidden`
- `bpy_struct.is_property_overridable_library`
- `bpy_struct.is_property_readonly`
- `bpy_struct.is_property_set`
- `bpy_struct.items`
- `bpy_struct.keyframe_delete`
- `bpy_struct.keyframe_insert`
- `bpy_struct.keys`
- `bpy_struct.path_from_id`
- `bpy_struct.path_resolve`
- `bpy_struct.pop`
- `bpy_struct.property_overridable_library_set`
- `bpy_struct.property_unset`
- `bpy_struct.type_recast`
- `bpy_struct.values`

References

- `Armature.collections`