Skip to content

Pose Operators

bpy.ops.pose.**armature_apply(*, selected=False)**

Apply the current pose as the new rest pose

PARAMETERS:

selected (*boolean, (optional)*) – Selected Only, Only apply the selected bones (with propagation to children)

bpy.ops.pose.**autoside_names(*, axis='XAXIS')**

Automatically renames the selected bones according to which side of the target axis they fall on

PARAMETERS:

axis (*enum in ['XAXIS', 'YAXIS', 'ZAXIS'], (optional)*) –

Axis, Axis to tag names with

- `XAXIS` X-Axis – Left/Right.
- `YAXIS` Y-Axis – Front/Back.
- `ZAXIS` Z-Axis – Top/Bottom.

bpy.ops.pose.**blend_to_neighbor(*, factor=0.5, prev_frame=0, next_frame=0, channels='ALL', axis_lock='FREE')**

Blend from current position to previous or next keyframe

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Factor, Weighting factor for which keyframe is favored more
- **prev_frame** (*int in [-1048574, 1048574], (optional)*) – Previous Keyframe, Frame number of keyframe immediately before the current frame
- **next_frame** (*int in [-1048574, 1048574], (optional)*) – Next Keyframe, Frame number of keyframe immediately after the current frame
- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*) –
  Channels, Set of properties that are affected
  - `ALL` All Properties – All properties, including transforms, bendy bone shape, and custom properties.
  - `LOC` Location – Location only.
  - `ROT` Rotation – Rotation only.
  - `SIZE` Scale – Scale only.
  - `BBONE` Bendy Bone – Bendy Bone shape properties.
  - `CUSTOM` Custom Properties – Custom properties.
- **axis_lock** (*enum in ['FREE', 'X', 'Y', 'Z'], (optional)*) –
  Axis Lock, Transform axis to restrict effects to
  - `FREE` Free – All axes are affected.
  - `X` X – Only X-axis transforms are affected.
  - `Y` Y – Only Y-axis transforms are affected.
  - `Z` Z – Only Z-axis transforms are affected.

bpy.ops.pose.**blend_with_rest(*, factor=0.5, prev_frame=0, next_frame=0, channels='ALL', axis_lock='FREE')**

Make the current pose more similar to, or further away from, the rest pose

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Factor, Weighting factor for which keyframe is favored more
- **prev_frame** (*int in [-1048574, 1048574], (optional)*) – Previous Keyframe, Frame number of keyframe immediately before the current frame
- **next_frame** (*int in [-1048574, 1048574], (optional)*) – Next Keyframe, Frame number of keyframe immediately after the current frame
- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*) –

- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*)

  Channels, Set of properties that are affected

  - `ALL` All Properties – All properties, including transforms, bendy bone shape, and custom properties.
  - `LOC` Location – Location only.
  - `ROT` Rotation – Rotation only.
  - `SIZE` Scale – Scale only.
  - `BBONE` Bendy Bone – Bendy Bone shape properties.
  - `CUSTOM` Custom Properties – Custom properties.

- **axis_lock** (*enum in ['FREE', 'X', 'Y', 'Z'], (optional)*) –

  Axis Lock, Transform axis to restrict effects to

  - `FREE` Free – All axes are affected.
  - `X` X – Only X-axis transforms are affected.
  - `Y` Y – Only Y-axis transforms are affected.
  - `Z` Z – Only Z-axis transforms are affected.

bpy.ops.pose.**breakdown(*, factor=0.5, prev_frame=0, next_frame=0, channels='ALL', axis_lock='FREE')**

Create a suitable breakdown pose on the current frame

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Factor, Weighting factor for which keyframe is favored more
- **prev_frame** (*int in [-1048574, 1048574], (optional)*) – Previous Keyframe, Frame number of keyframe immediately before the current frame
- **next_frame** (*int in [-1048574, 1048574], (optional)*) – Next Keyframe, Frame number of keyframe immediately after the current frame
- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*) –

  Channels, Set of properties that are affected

  - `ALL` All Properties – All properties, including transforms, bendy bone shape, and custom properties.
  - `LOC` Location – Location only.
  - `ROT` Rotation – Rotation only.
  - `SIZE` Scale – Scale only.
  - `BBONE` Bendy Bone – Bendy Bone shape properties.
  - `CUSTOM` Custom Properties – Custom properties.

- **axis_lock** (*enum in ['FREE', 'X', 'Y', 'Z'], (optional)*) –

  Axis Lock, Transform axis to restrict effects to

  - `FREE` Free – All axes are affected.
  - `X` X – Only X-axis transforms are affected.
  - `Y` Y – Only Y-axis transforms are affected.
  - `Z` Z – Only Z-axis transforms are affected.

bpy.ops.pose.**constraint_add(*, type='')**

Add a constraint to the active bone

PARAMETERS:

type (enum in Constraint Type Items, (optional)) – Type

bpy.ops.pose.**constraint_add_with_targets(*, type='')**

Add a constraint to the active bone, with target (where applicable) set to the selected Objects/Bones

PARAMETERS:

type (enum in Constraint Type Items, (optional)) – Type

bpy.ops.pose.**constraints_clear()**

Clear all constraints from the selected bones

bpy.ops.pose.**constraints_copy()**

Copy constraints to other selected bones

bpy.ops.pose.**copy()**

Copy the current pose of the selected bones to the internal clipboard

bpy.ops.pose.**flip_names(*, do_strip_numbers=False)**

Flips (and corrects) the axis suffixes of the names of selected bones

PARAMETERS:

do_strip_numbers (*boolean, (optional)*) – Strip Numbers, Try to remove right-most dot-number from flipped names.Warning: May result i incoherent naming in some cases

bpy.ops.pose.**hide(*, unselected=False)**

Tag selected bones to not be visible in Pose Mode

PARAMETERS:

unselected (*boolean, (optional)*) – Unselected

bpy.ops.pose.**ik_add(*, with_targets=True)**

Add IK Constraint to the active Bone

PARAMETERS:

with_targets (*boolean, (optional)*) – With Targets, Assign IK Constraint with targets derived from the select bones/objects

bpy.ops.pose.**ik_clear()**

Remove all IK Constraints from selected bones

bpy.ops.pose.**loc_clear()**

Reset locations of selected bones to their default values

bpy.ops.pose.**paste(*, flipped=False, selected_mask=False)**

Paste the stored pose on to the current pose

PARAMETERS:

- **flipped** (*boolean, (optional)*) – Flipped on X-Axis, Paste the stored pose flipped on to current pose
- **selected_mask** (*boolean, (optional)*) – On Selected Only, Only paste the stored pose on to selected bones in the current pose

bpy.ops.pose.**paths_calculate(*, display_type='RANGE', range='SCENE', bake_location='HEADS')**

Calculate paths for the selected bones

PARAMETERS:

- **display_type** (enum in Motionpath Display Type Items, (optional)) – Display type
- **range** (enum in Motionpath Range Items, (optional)) – Computation Range
- **bake_location** (enum in Motionpath Bake Location Items, (optional)) – Bake Location, Which point on the bones is used when calculating paths

bpy.ops.pose.**paths_clear(*, only_selected=False)**

Undocumented, consider contributing.

PARAMETERS:

only_selected (*boolean, (optional)*) – Only Selected, Only clear motion paths of selected bones

bpy.ops.pose.**paths_range_update()**

Update frame range for motion paths from the Scene's current frame range

bpy.ops.pose.**paths_update()**

Recalculate paths for bones that already have them

bpy.ops.pose.**propagate(*, mode='NEXT_KEY', end_frame=250.0)**

Copy selected aspects of the current pose to subsequent poses already keyframed

PARAMETERS:

- **mode** (*enum in ['NEXT_KEY', 'LAST_KEY', 'BEFORE_FRAME', 'BEFORE_END', 'SELECTED_KEYS', 'SELECTED_MARKERS'], (optional)*) –

  Terminate Mode, Method used to determine when to stop propagating pose to keyframes

  - `NEXT_KEY` To Next Keyframe – Propagate pose to first keyframe following the current frame only.
  - `LAST_KEY` To Last Keyframe – Propagate pose to the last keyframe only (i.e. making action cyclic).
  - `BEFORE_FRAME` Before Frame – Propagate pose to all keyframes between current frame and 'Frame' property.
  - `BEFORE_END` Before Last Keyframe – Propagate pose to all keyframes from current frame until no more are found.
  - `SELECTED_KEYS` On Selected Keyframes – Propagate pose to all selected keyframes.
  - `SELECTED_MARKERS` On Selected Markers – Propagate pose to all keyframes occurring on frames with Scene Markers after the current frame.

- **end_frame** (*float in [1.17549e-38, inf], (optional)*) – End Frame, Frame to stop propagating frames to (for 'Before Frame' mode)

bpy.ops.pose.**push(*, factor=0.5, prev_frame=0, next_frame=0, channels='ALL', axis_lock='FREE')**

Exaggerate the current pose in regards to the breakdown pose

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Factor, Weighting factor for which keyframe is favored more
- **prev_frame** (*int in [-1048574, 1048574], (optional)*) – Previous Keyframe, Frame number of keyframe immediately before the current frame
- **next_frame** (*int in [-1048574, 1048574], (optional)*) – Next Keyframe, Frame number of keyframe immediately after the current frame
- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*) –

  Channels, Set of properties that are affected

  - `ALL` All Properties – All properties, including transforms, bendy bone shape, and custom properties.
  - `LOC` Location – Location only.
  - `ROT` Rotation – Rotation only.
  - `SIZE` Scale – Scale only.
  - `BBONE` Bendy Bone – Bendy Bone shape properties.
  - `CUSTOM` Custom Properties – Custom properties.

- **axis_lock** (*enum in ['FREE', 'X', 'Y', 'Z'], (optional)*) –

  Axis Lock, Transform axis to restrict effects to

  - `FREE` Free – All axes are affected.
  - `X` X – Only X-axis transforms are affected.
  - `Y` Y – Only Y-axis transforms are affected.
  - `Z` Z – Only Z-axis transforms are affected.

bpy.ops.pose.**quaternions_flip()**

Flip quaternion values to achieve desired rotations, while maintaining the same orientations

bpy.ops.pose.**relax(*, factor=0.5, prev_frame=0, next_frame=0, channels='ALL', axis_lock='FREE')**

Make the current pose more similar to its breakdown pose

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Factor, Weighting factor for which keyframe is favored more
- **prev_frame** (*int in [-1048574, 1048574], (optional)*) – Previous Keyframe, Frame number of keyframe immediately before the current frame
- **next_frame** (*int in [-1048574, 1048574], (optional)*) – Next Keyframe, Frame number of keyframe immediately after the current frame
- **channels** (*enum in ['ALL', 'LOC', 'ROT', 'SIZE', 'BBONE', 'CUSTOM'], (optional)*) –

  Channels, Set of properties that are affected

  - `ALL` All Properties – All properties, including transforms, bendy bone shape, and custom properties.
  - `LOC` Location – Location only.
  - `ROT` Rotation – Rotation only.
  - `SIZE` Scale – Scale only.
  - `BBONE` Bendy Bone – Bendy Bone shape properties.
  - `CUSTOM` Custom Properties – Custom properties.

- **axis_lock** (*enum in ['FREE', 'X', 'Y', 'Z'], (optional)*) –

  Axis Lock, Transform axis to restrict effects to

  - `FREE` Free – All axes are affected.
  - `X` X – Only X-axis transforms are affected.
  - `Y` Y – Only Y-axis transforms are affected.
  - `Z` Z – Only Z-axis transforms are affected.

bpy.ops.pose.**reveal(*, select=True)**

Reveal all bones hidden in Pose Mode

PARAMETERS:

select (*boolean, (optional)*) – Select

bpy.ops.pose.**rot_clear()**

Reset rotations of selected bones to their default values

bpy.ops.pose.**rotation_mode_set(*, type='QUATERNION')**

Set the rotation representation used by selected bones

PARAMETERS:

type (enum in Object Rotation Mode Items, (optional)) – Rotation Mode

bpy.ops.pose.**scale_clear()**

Reset scaling of selected bones to their default values

bpy.ops.pose.**select_all(*, action='TOGGLE')**

Toggle selection status of all bones

PARAMETERS:

action (*enum in ['TOGGLE', 'SELECT', 'DESELECT', 'INVERT'], (optional)*) –

Action, Selection action to execute

- `TOGGLE` Toggle – Toggle selection for all elements.
- `SELECT` Select – Select all elements.
- `DESELECT` Deselect – Deselect all elements.
- `INVERT` Invert – Invert selection of all elements.

bpy.ops.pose.**select_constraint_target()**

Select bones used as targets for the currently selected bones

bpy.ops.pose.**select_grouped(*, extend=False, type='COLLECTION')**

bpy.ops.pose.**select_grouped(*, extend=False, type='COLLECTION')**

Select all visible bones grouped by similar properties

PARAMETERS:

- **extend** (*boolean, (optional)*) – Extend, Extend selection instead of deselecting everything first
- **type** (*enum in ['COLLECTION', 'COLOR', 'KEYINGSET'], (optional)*) – Type
  - `COLLECTION` Collection – Same collections as the active bone.
  - `COLOR` Color – Same color as the active bone.
  - `KEYINGSET` Keying Set – All bones affected by active Keying Set.

bpy.ops.pose.**select_hierarchy(*, direction='PARENT', extend=False)**

Select immediate parent/children of selected bones

PARAMETERS:

- **direction** (*enum in ['PARENT', 'CHILD'], (optional)*) – Direction
- **extend** (*boolean, (optional)*) – Extend, Extend the selection

bpy.ops.pose.**select_linked()**

Select all bones linked by parent/child connections to the current selection

bpy.ops.pose.**select_linked_pick(*, extend=False)**

Select bones linked by parent/child connections under the mouse cursor

PARAMETERS:

- **extend** (*boolean, (optional)*) – Extend, Extend selection instead of deselecting everything first

bpy.ops.pose.**select_mirror(*, only_active=False, extend=False)**

Mirror the bone selection

PARAMETERS:

- **only_active** (*boolean, (optional)*) – Active Only, Only operate on the active bone
- **extend** (*boolean, (optional)*) – Extend, Extend the selection

bpy.ops.pose.**select_parent()**

Select bones that are parents of the currently selected bones

bpy.ops.pose.**selection_set_add()**

Create a new empty Selection Set

FILE:

startup/bl_operators/bone_selection_sets.py:143

bpy.ops.pose.**selection_set_add_and_assign()**

Create a new Selection Set with the currently selected bones

FILE:

startup/bl_operators/bone_selection_sets.py:274

bpy.ops.pose.**selection_set_assign()**

Add selected bones to Selection Set

FILE:

startup/bl_operators/bone_selection_sets.py:190

bpy.ops.pose.**selection_set_copy()**

Copy the selected Selection Set(s) to the clipboard

FILE:

startup/bl_operators/bone_selection_sets.py:286

bpy.ops.pose.**selection_set_delete_all()**

Remove all Selection Sets from this Armature

FILE:

startup/bl_operators/bone_selection_sets.py:73

bpy.ops.pose.**selection_set_deselect()**

Remove Selection Set bones from current selection

FILE:

startup/bl_operators/bone_selection_sets.py:257

bpy.ops.pose.**selection_set_move(*, direction='UP')**

Move the active Selection Set up/down the list of sets

PARAMETERS:

direction (*enum in ['UP', 'DOWN'], (optional)*) – Move Direction, Direction to move the active Selection Set: UP (default) or DOWN

FILE:

startup/bl_operators/bone_selection_sets.py:122

bpy.ops.pose.**selection_set_paste()**

Add new Selection Set(s) from the clipboard

FILE:

startup/bl_operators/bone_selection_sets.py:298

bpy.ops.pose.**selection_set_remove()**

Remove a Selection Set from this Armature

FILE:

startup/bl_operators/bone_selection_sets.py:161

bpy.ops.pose.**selection_set_remove_bones()**

Remove the selected bones from all Selection Sets

FILE:

startup/bl_operators/bone_selection_sets.py:85

bpy.ops.pose.**selection_set_select(*, selection_set_index=-1)**

Select the bones from this Selection Set

PARAMETERS:

selection_set_index (*int in [-inf, inf], (optional)*) – Selection Set Index, Which Selection Set to select; -1 uses the active Selection Set

FILE:

startup/bl_operators/bone_selection_sets.py:235

bpy.ops.pose.**selection_set_unassign()**

Remove selected bones from Selection Set

FILE:

startup/bl_operators/bone_selection_sets.py:209

bpy.ops.pose.**transforms_clear()**

Reset location, rotation, and scaling of selected bones to their default values

bpy.ops.pose.**user_transforms_clear(*, only_selected=True)**

Reset pose bone transforms to keyframed state

PARAMETERS:

only_selected (*boolean, (optional)*) – Only Selected, Only visible/selected bones

bpy.ops.pose.**visual_transform_apply()**

Apply final constrained position of pose bones to their transform