JSD Workflow Guide

v2.84 | Stair creation, developments, handrails and balustrade

Start with the stair, create the developments you need, then add stringers, rails and infills. Make changes through the saved settings or the editable source geometry so the linked outputs can follow.

This guide covers v2.84. Batch capture, landing review and structural finish build-up are checked against the source and calculation tests; the complete editing and geometry cycle still needs an AutoCAD 2024 trial. The guide stops before fabrication drawings and export tools.

1. Choose how to make the stair

Before you start

- Work in millimetres: `INSUNITS` = 4. Use model space and the required drawing plane.
- Use StairPlan when you want the plugin to calculate the stair from dimensions. Use Manual Stair when you already have the tread outlines or surveyed edges.
- Keep the original linked geometry. Copying an output does not create a second independent live stair or development.

Option A — StairPlan: make the stair from dimensions

1. Run `JSDSTAIRPLAN`, or click Stair Plan. You do not need to select existing geometry first.
2. Choose the Stair Geometry Type: curved, curved with landing, straight, 180° return, 90° return, or 90° winders.
3. Enter the geometry and floor levels. Review the tread count, going, widths, direction and any landing or straight-flight settings.
4. Set the finishes, structure and reference paths described in section 2.
5. Accept the settings, then follow the placement prompt. A curved stair asks for its centre; the other builders use their own placement prompts.

To change the result later, run `JSDLIVESTAIRPLANEDIT` and select an original generated tread or reference from that stair. Change the saved settings and regenerate.

Option B — Manual Stair: full tread polylines

1. Draw one lightweight polyline around each tread. Use horizontal WCS XY outlines with normal +Z; inner and outer sides can contain arcs. The nosing and rear must each be a straight segment.

2. Run JSDPOLYLINESTAIR, or use Manual Stair → Polyline. Choose All and select every tread and landing polyline together, then Enter. Individual capture remains available for complex outlines.
3. From that selection, identify the lowest/first tread. Pick near the middle of its nosing segment, then near the inner end of that first nosing. The remaining treads and their boundaries are inferred from the connected geometry.
4. Review the landing prompt. Central goings over 400 mm are suggested: accept Suggested, choose Select and pick the actual landing outlines, or choose None. This is an overrideable suggestion. Open outlines are closed in the source; select 2 to 500 treads.
5. Choose the level method: drawn Z, equal rises between finished floor levels, or individual finished risers. In individual mode include the final rise to the upper floor.
6. In StairPlan → Polyline Source, explicitly confirm Finished treads and riser nosings or Structure treads and riser faces. Structural Z receives the tread finish depth; in plan the finished riser moves forward by riser finish thickness and the nosing adds its overhang beyond that. Floor levels and individual risers always describe finished levels.
7. Confirm the rear boundary: nosing-to-nosing outlines need rear overlap; complete tread outlines already include their depth. Set the overlap and whether to add the riser finish thickness.
8. StairPlan opens with the captured geometry. Check the floor levels and add finishes, structure and the reference paths you need, then generate.

Adjacent side boundaries must reach the next nosing within 0.1 mm in plan. The batch must form an unambiguous connected chain; duplicates, branches or an ambiguous final rear require correction or Individual capture. Use one outline per landing. Drawn-Z levels must increase uphill; flat drawings use floor levels or individual risers. Turning landing footprints with a zero-length side remain outside this capture model.

By default, v2.83 puts all captured tread polylines in one source block on `JSDLIVE`-SURVEY-SOURCES. `BEDIT` that block to adjust several treads together, then save and close once. The complete stair is checked before rebuilding. Invalid edits keep the last valid outputs and report the problem.

Keep each tread's vertex count and order when editing. Adding/removing vertices, changing the nosing/rear identities or inserting/reordering treads requires recapture. Existing individual blocks can use Edit Plan → Polyline Source → Combine into One Stair Block → Generate. Review landing flags with JSDSTAIRLANDINGS or Confirm / Edit Landing Treads in Polyline Source; the suggestion threshold is editable there.

Option C — Manual Stair: the existing edge method

Run `JSDSURVEYSTAIR` and choose Edges. For surveyed geometry at different Z levels, select all inner tread-edge pieces, then all outer pieces. Select an inner edge on the lowest tread and identify landing edges if needed. Matching inner/outer chains at each level form the treads. The flat-plan route captures inner and outer pieces per tread, from bottom to top. Use open horizontal lines, arcs or lightweight polylines; their chains must join.

2. What StairPlan controls

StairPlan is the saved stair model. It holds the geometry, floor levels, tread/riser build-up and the paths that downstream developments use. The manual stair route feeds captured geometry into the same system; its shape is controlled by the captured source rather than a calculated radius or sweep.

- Stair geometry: floor-to-floor height, lower finished floor level, count, width, direction and the dimensions appropriate to the chosen stair type. Curved stairs use radius/sweep; return stairs use flight, well and landing controls.
- Straight treads and landings: lead-in/lead-out straight treads, their goings and the landing options available to that builder.
- Upper FFL tread: decide whether the last tread is at the upper finished floor. For a manual stair this uses the last selected tread, so recheck the number of rises.
- Tread finishes: finish depth, rear inset/extension, outline/solid display and the applicable riser finish settings.
- Risers: first/last riser inclusion, setback, finish thickness, solid/outline display and steel riser settings where enabled.
- Structure: steel tread/riser thicknesses and offsets, or concrete with its waist, landing and junction settings. Options depend on the stair type. Captured survey treads use separate plates rather than the standard folding workflow.
- Stringer references: inner/outer controls, widths, centreline or face positions, spine and one-sided options, plus additional paths where available. These locate future developments; configure the full stringer profile after developing.
- Handrail references: enable the inner and/or outer handrail centreline and set its side offset. These locate the future balustrade development; they do not by themselves create the raised handrail.
- Presets and preview: reuse saved build-up settings, then check the plan and section previews before generating. Cleat controls are available where supported by the selected steel setup.

Structure example: source riser face at station 0, 25 mm riser finish and 25 mm overhang gives finished riser at -25 and finished nosing at -50 mm. A 250 mm source bay with the default rear allowance gives a 300 mm finished tread. Drawn structural Z 500 plus 40 mm tread finish gives finished Z 540. The live source stays structural; steel/concrete use the structural footprint and their own controls. Rear allowances are corrected against the derived finished reference to avoid adding depth twice. Confirm Complete outline versus Nosing-to-nosing; the upper FFL tread has no automatic rear overlap.

3. Create developments from the stair

A development is the stair route laid out flat: distance along the route runs horizontally, and height runs vertically. Each saved development belongs to a particular route, side and, where applicable, flight. It supplies the editable flat geometry and the link back to the stair's 3D position.

1. Run `JSDLIVEDEVELOP`, or click Develop.
2. Select one original generated StairPlan tread or reference line. Do not select the manual stair's source block or an ordinary copied outline.
3. Choose the route: internal/external stringer, internal/external tread, or internal/external balustrade/handrail. Spine and additional stringer routes appear when that stair setup supports them.
4. Choose the flight where offered. For stringer/balustrade developments choose the viewing face and uphill direction as required.
5. Move the preview to a clear drawing area and click the lower-left placement point.

Only develop routes that were enabled in StairPlan. If the reference is missing, use Edit Plan to enable it and regenerate, then create the development. A stringer development and a balustrade development are separate saved sets, even when they follow the same side of the stair.

Dimensions and manual changes

- Enter controlling dimensions in the relevant settings: stair dimensions in StairPlan, stringer height/depth/thickness in Stringer settings, and rail heights/sections in the rail settings.
- Use `JSDLIVESTRINGER` or `JSDLIVESTRINGEREDIT` and select the original stringer development. Set its profile dimensions and enable the required 3D output. Stringer top height and depth below the pitch are separate controls.
- Grip edits to an editable developed stringer outline or rail centreline can be saved as profile overrides. Keep stringer outlines closed and rail centrelines open, in world XY at Z = 0. Split profiles need their settings or Local Radius rather than an arbitrary grip-edited replacement.
- The developed pitch line has its own manual override. Keep the required horizontal start/end stations and an increasing run; change the source stair or route settings when the overall run must change.
- Normal AutoCAD dimensions are measurement annotations here. Changing their displayed text does not drive the plugin geometry. Recheck dimensions after regeneration; do not assume dimensions attached to replaced entities remain associated.

4. Manual developments from your own route

Use `JSDMANUALDEVELOP` when you have a route drawn in plan and want a development without creating a StairPlan first. This is different from Manual Stair, which captures full tread footprints.

1. Draw one connected route using separate open arcs, lines or lightweight polylines. Select one side/path only, not both stair edges and not closed tread outlines.
2. Run `JSDMANUALDEVELOP`, choose a suitable join tolerance, then select the route curves. Selection order can be mixed.

3. Pick the lower/start end. If the start is ambiguous, pick inside the first curve to confirm the travel direction.
4. Give the development a name and choose Stringer, Balustrade/Handrail or Other.
5. Choose how to establish its rise: selected source Z levels, an overall rise along the route, equal risers, or individual rises. Enter the lower and upper end rises where applicable.
6. For Individual Rises, fill the Rises page. Each row is the rise after that selected segment; 0 means a level segment. Lower End Rise adds the rise before segment 1. Check the cumulative height.
7. Set the viewing direction and any lower landing, then place the development.

Use `JSDMANUALDEVELOPEDIT` on an original manual development. Settings edits its dimensions; Add appends selected route curves; Reselect replaces the route selection. Review the rise table after changing the number of segments. Keep source curves in the drawing: they remain live inputs.

5. Add, remove and restore developments

To add another side, route or flight, run Develop again and place a separate development. To adjust a level lead-in or lead-out, use `JSDLIVEEXTENDPATH` on the saved development. For extra curves on a manual route, use Manual Development Edit → Add.

To remove a development properly, use `JSDLIVEDEVELOPREMOVE` and pick it in the drawing or choose it from the list. Review the affected infills, wraps and dependent rails before applying either option:

- Archive development and remove its generated outputs: removes the displayed development/output while retaining the saved record for restoration through `JSDLIVEDEVELOPREGISTER`.
- Keep geometry; remove Live links: leaves a static reference. It will no longer follow changes to the stair.

The stair plan and reusable infill definitions are retained. Other developments linked to a removed stringer may pause or lose their dependent bottom-rail link; read the removal summary. Use Development Register to manage archived items. Simply erasing a few lines is not the same as removing the saved development.

6. Add handrails and balustrade

Create the correct development first

In StairPlan, enable the required handrail centreline. Use Develop to create the Internal or External balustrade/handrail development for that side and flight. If the bottom rail is to follow Stringer Top, generate the matching stringer development and profile first. Otherwise choose Pitchline as its reference.

Configure the rails

1. Run `JSDLIVEBAL` and select the original balustrade development pitch line or rail. A stair-plan tread or a stringer-only development is not the correct selection for this step.
2. On Handrails, choose Configure Rails / Generate 3D. `JSDLIVEHANDRAIL` or `JSDLIVEBALRAILS` also opens these settings for the selected balustrade development.
3. Enable the handrail, top rail and bottom rail as needed. Set flight and landing heights, sections, junction radii and the available end/return options.
4. Choose the bottom rail reference: Stringer Top or Pitchline. Set its signed underside offset and separate sideways offset; landing FFL overrides are available.
5. Choose Auto Loft or Sections Only and the required 3D build method. Generate, then compare the developed rails with their 3D position.

Handrail heights are measured vertically to the top. Top/bottom rail clearances use their stated perpendicular references. Read the field labels rather than treating every offset as a vertical height.

Choose infills

- Section infills: use Choose Section Infill, or `JSDLIVEBALSECTION` on the balustrade development. Choose a round/rectangular section or a valid drawn closed section, then set spacing, end limits and fixing options. These infills are fitted between the chosen boundaries.
- Decorative infills: capture one complete infill drawn upright in the development plane, or load an infill DWG. Pick its bottom/fixing reference when prompted, then confirm its outside width.
- On Start & End, set the start/end references, offsets, centre or outside-edge alignment, maximum clear gap and any repeating A-B-A pattern. Widths and gaps are measured horizontally on the development; internal openings inside a decorative pattern are not checked by that gap control.
- Use `JSDLIVEBALEDIT` and select an original infill in the run to change that saved run. Use `JSDBALUSTERLENGTH` for the supported baluster/post length edits.

Use the libraries

- Section Library — `JSDLIVEPROFILES`: save actual rail cross-sections. Draw a closed lightweight polyline or circle in world XY, give it a unique name and use Add drawn profile. Apply it through the rail profile selector, checking rotation and offsets.
- Infill Library — `JSDINFILLS`: browse and insert reusable decorative infills. `JSDINFILLSAVE` saves selected infill geometry; the balustrade dialog also has capture, save and load controls.
- `JSDINFILLEDIT` opens a saved infill DWG for editing. Save it and refresh the library afterwards. Editing that library file does not automatically replace already placed instances; reload/reapply deliberately.

7. Manual balustrade and Wrap

Make an existing patterned run individually editable

Run `JSDLIVEBALMANUAL` and select an original patterned infill block in the run. The whole run becomes individually editable source blocks. Move or `BEDIT` those sources to adjust individual infills; the linked 3D follows without redistributing the gaps. This conversion is for patterned blocks, not the section-infill run.

Wrap infills you have drawn yourself

1. Create the correct live development first. Draw the manual infills flat at their required stations and heights on that development, using blocks, solids, circles or closed lightweight polylines.
2. Run `JSDLIVEWRAP`, or `JSDLIVEMANUALINFILLS`, and first select the original live development.
3. Select the developed original infill geometry. Do not select wrapped copies, plan 3D output, or profiles belonging to another development.
4. Choose the anchor: bottom centre for infill outlines or centre for section profiles. Enable extrusion and enter thickness for closed infill outlines if required. Loft needs at least two matching closed section profiles.
5. Generate the wrap and check its orientation and placement. Edit the developed source geometry to make changes; use `JSDLIVEWRAPEDIT` on an original wrapped output to change the saved wrap settings.

Wrap uses the linked straight and curved route. Source geometry remains editable and the mapped output is regenerated. Use the balustrade tool for automatic patterned runs; use Wrap when you want to control the individual source positions yourself.

8. Holes and the balustrade fixing link

Add, edit or remove holes and slots

1. Run `JSDLIVEHOLES` and select the original live 3D solids that should be drilled together, such as the stringer and the supported connected plates. Select the actual solids, not a 2D outline.
2. Pick the first hole centre with a face OSNAP. Enter diameter, round-hole or slot length, rows/columns, spacing and rotation.
3. Set the anchor offsets, drill direction and cut depth. Use Pick drilling axis when the face is angled. Enable simplified bolt display if wanted, then apply.

4. Use `JSDLIVEHOLESEDIT` to select a saved pattern and edit it, or choose Remove selected pattern and restore material. Removing the pattern regenerates the material while retaining other active fixing patterns.

Link fixing circles on a separate balustrade development

Use Link Balustrade Fixings — `JSDFABBALFIX` — for this relationship. Its command prefix contains FAB, but this guide only covers the link and fixing-circle workflow.

1. Generate a stringer with solid output enabled and a separate matching balustrade development. Run `JSDFABBALFIX` and select the target stringer development first.
2. Choose Draw to establish the link and make its fixing layer current. The tool can find a matching balustrade side/flight; otherwise select the separate balustrade development when prompted.
3. Draw `CIRCLE` fixing marks on that balustrade development using the layer the command sets. Switch back to your normal drawing layer when finished.
4. Alternatively choose Link and select existing original fixing circles to bring them into the watched layer. Use the source circles, rather than generated copies of fixing marks.
5. Moving or erasing the source circles updates the corresponding stringer fixings. Use Unlink to remove the relationship; existing manual stringer cutouts are retained.

This link requires supported solid stringer output; automatic fixing holes are not supported for the built-up box stringer option. Keep ordinary 3D hole patterns and balustrade-driven fixing circles distinct so you edit the correct source.

9. Apply a local radius

Use `JSDLIVERADIUS` after generating the stringer or rails. Select the original development, choose Stringer or Rails if both exist, then choose Edit.

- Stringer: apply the radius to the top edge or bottom edge.
- Rails: apply it to the handrail, top rail or bottom rail developed path.

Choose the edge, enter the radius and pick near the developed mitre junction. Radius 0 gives a sharp mitre. The saved junction setting is reused by the linked 3D output. The tool needs an actual change of direction; a straight edge has no junction to round.

Local Radius is for these developed paths, not a general tread-corner or hole-rounding tool. Applying it replaces the manual profile override for the affected edge/layer. Its Reset action clears the local radius overrides for the selected stringer or rail group; it does not perform Profile Reset.

10. Reset profiles after manual edits

Profile Reset — JSDLIVEPROFILERESET

Run the command and select the original development whose manual profile edits you want to clear. It removes all saved profile-shape overrides in that development and regenerates from the current settings: the stringer outline and the handrail, top-rail and bottom-rail developed centrelines.

Use it when grip-edited shapes have become awkward and you want the calculated profiles back. It clears the development's profile overrides together, not just the corner you click. Source tread/block edits, saved dimensions, pitch overrides and saved local-radius settings are not cleared by this reset.

Pitch Reset — JSDLIVEPITCHRESET

Use this separately to remove a selected development's pitch override. For a clean return from both kinds of manual development edits, reset the pitch as well as the profiles. Neither command restores an earlier version of the source stair.

11. A sensible order for a job

1. Create the stair or capture its tread outlines. Confirm finished floor levels, tread/riser build-up and rear allowances.
2. Enable the required stringer and handrail reference paths, then generate.
3. Develop the required stringer sides/flights and configure their profiles.
4. Develop the required balustrade sides/flights. Configure rails and their section profiles, then add section or decorative infills.
5. Use manual infill sources and Wrap where an automatic patterned run is not suitable.
6. Add the required holes and fixing links. Apply local junction radii and check the linked 3D result.
7. For changes, edit the correct saved settings or source geometry. Use Profile Reset or Pitch Reset only when you want to clear those particular development overrides.

If a change has not appeared: finish the active command or Block Editor session, clear selection/grips with Esc and check JSDLIVESTATUS. JSDLIVEPAUSE pauses updates; JSDLIVERESUME resumes and rebuilds. JSDLIVEUPDATE forces an update. Read any reported source or link error before changing more geometry.

Related reference: the JSD command-and-icon catalogue lists every current command, including aliases and the legacy LISP prototypes. This guide uses the current native workflow rather than those prototypes.

Command and icon catalogue:
https://docs.google.com/spreadsheets/d/1hNkCYBSWkuKG-eD_L9-aXTOpTraxuxMX/edit