

Bezigon Permutations - Corner, Smooth and Cusp

Versions 4.2.0 and 2.2.0 · 16 October 2026

These tables apply to **both 4.2.0 and 2.2.0**. Every case was built with both libraries, and the handles, node types and flags agree in all 79 of them. Tangent nodes, which exist only in 4.2.0, are in *Permutations - Tangent*.

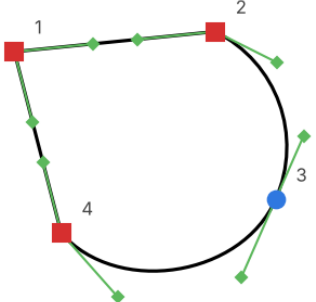
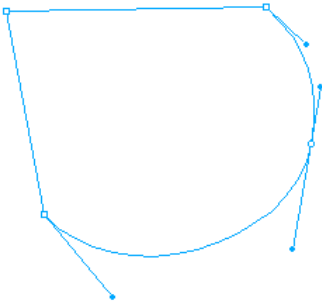
Markers: red square = Corner (a cusp is drawn the same way, as in the demo), blue circle = Smooth, green = handles. A grey ring marks the node an edit acts on. Node numbers are printed beside each node; where two nodes share a point both numbers are shown. PC and MC mean plain click and modifier click.

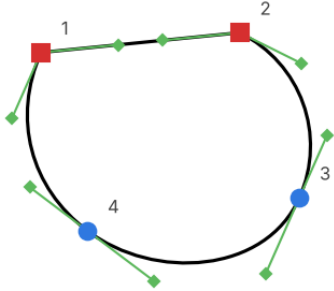
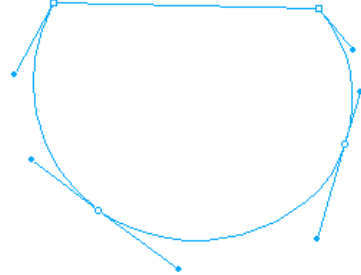
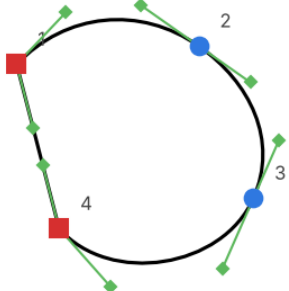
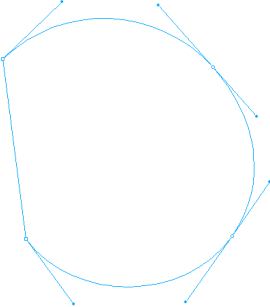
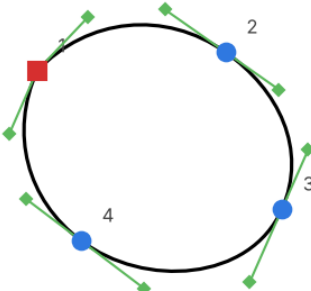
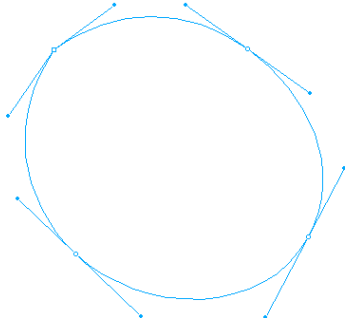
The pictures are drawn from the library's own output, styled like the demo. Table 1 carries over your FreeHand captures from the 4.1.0 tables: those sixteen shapes are unchanged in 4.2.0. Every other FreeHand column is blank for you to paste the matching capture.

In the code, node 1 is index 0. In 2.2.0, `set_node_smooth(path, i, true)` replaces `set_node_type(path, i, Smooth)`, and `false` replaces Corner.

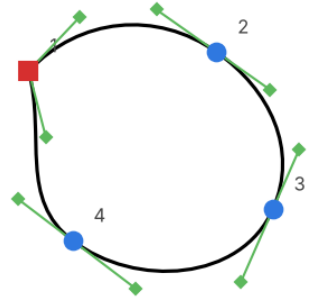
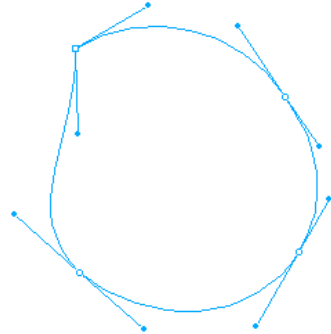
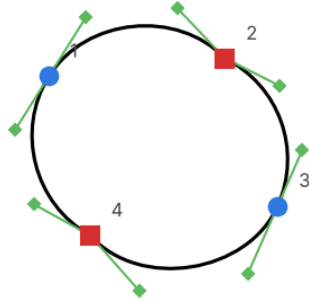
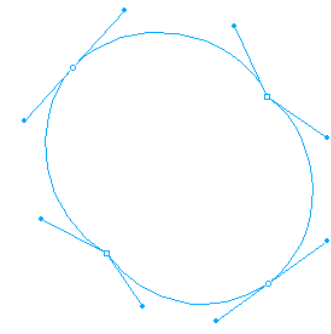
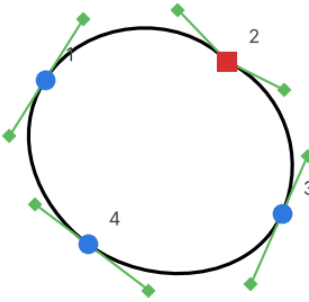
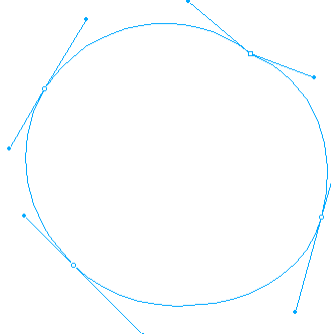
1. Four nodes

Every combination of plain and modifier click for the start, second and fourth nodes, with the third always a modifier click so that each shape contains a curve, closed both ways. The nodes sit at (170,110), (470,80), (560,330), (240,380) in every row, so any difference comes from the clicks alone.

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
1	PC	PC	MC	PC	PC		Corner. In: along the chord to node 4. Out: along the chord to node 2.		The one major difference in our Bezigon algorithm is the corner node. FreeHand does not show handles for true corner nodes. Ours always does, treating every corner node as a cusp node.

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
2	PC	PC	MC	MC	PC		Corner. In: following node 4's arc. Out: along the chord to node 2.		
3	PC	MC	MC	PC	PC		Corner. In: along the chord to node 4. Out: following node 2's arc.		
4	PC	MC	MC	MC	PC		Corner. In: following node 4's arc. Out: following node 2's arc.		

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
5	PC	PC	MC	PC	MC		Corner. In: along the chord to node 4. Out: along the chord to node 2.		
6	PC	PC	MC	MC	MC		Corner. In: along the chord to node 4. Out: along the chord to node 2.		
7	PC	MC	MC	PC	MC		Corner. In: along the chord to node 4. Out: following node 2's arc.		

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
8	PC	MC	MC	MC	MC		Corner. In: along the chord to node 4. Out: following node 2's arc.		
9	MC	PC	MC	PC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
10	MC	PC	MC	MC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
11	MC	MC	MC	PC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
12	MC	MC	MC	MC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
13	MC	PC	MC	PC	MC		cuspl. In: along the chord to node 4. Out: along the chord to node 2.		

#	Start	2nd	3rd	4th	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
14	MC	PC	MC	MC	MC		cuspl. In: along the chord to node 4. Out: along the chord to node 2.		
15	MC	MC	MC	PC	MC		cuspl. In: along the chord to node 4. Out: along the chord to node 2.		
16	MC	MC	MC	MC	MC		cuspl. In: along the chord to node 4. Out: along the chord to node 2.		

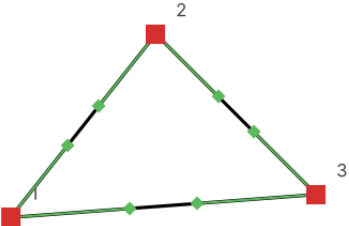
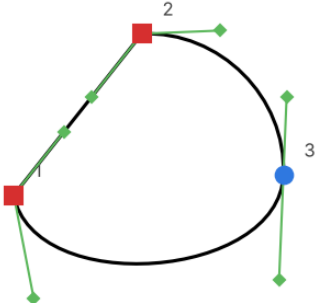
What the sixteen rows show

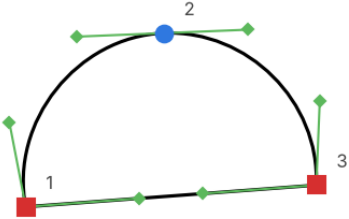
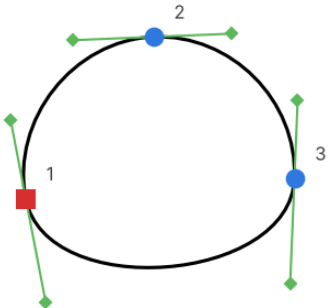
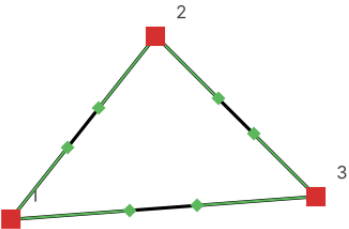
- Rows 1 to 8 start with a plain click, so the start node is a plain corner. Each handle lies along its chord, or follows the neighbour's arc when the neighbour is smooth.

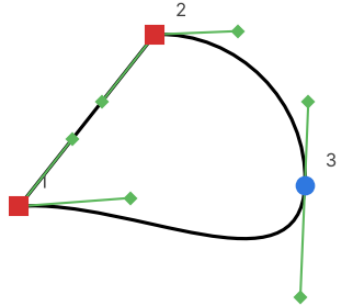
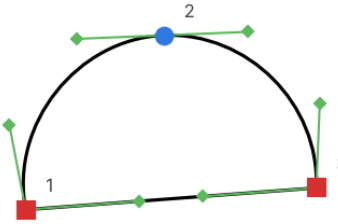
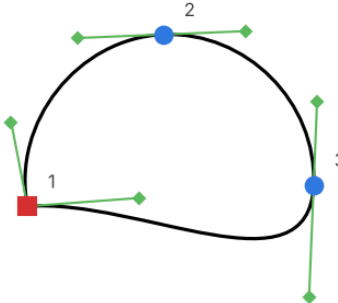
- Rows 5 to 8 close with the modifier. That sends the start node's in-handle straight back to node 4 whatever node 4 is doing: the one rule the close adds.
- Rows 9 to 12 start with a modifier click and close with a plain one, so the start node stays smooth.
- Rows 13 to 16 start with a modifier click and close with the modifier. The close turns the start node into a cusp: both handles lie along the chords.
- In every row both handles of the start node are present. A sharp node always carries handles along its chords, so it can be reshaped without being converted first. They do not change the drawn shape.

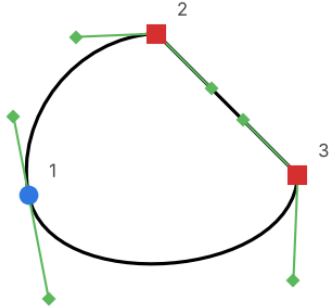
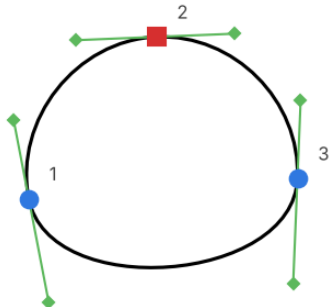
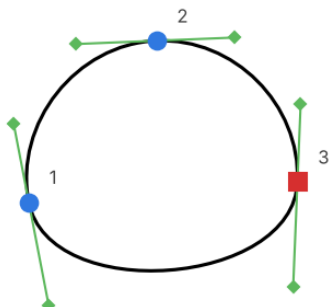
2. Three nodes

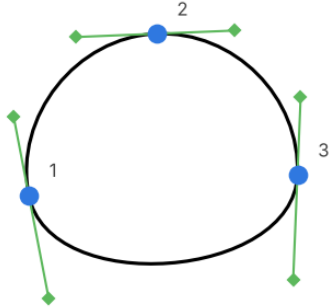
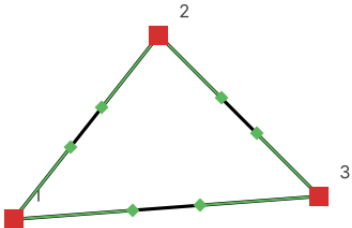
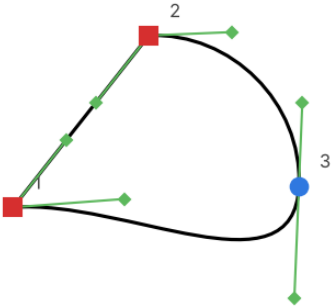
The smallest loop that can curve. Every combination of plain and modifier click for all three nodes, closed both ways. The nodes sit at (170,330), (360,90), (570,300).

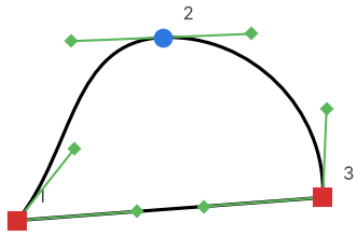
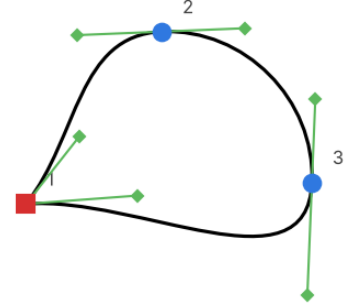
#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
1	PC	PC	PC	PC		Corner. In: along the chord to node 3. Out: along the chord to node 2.		
2	PC	PC	MC	PC		Corner. In: following node 3's arc. Out: along the chord to node 2.		

#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
3	PC	MC	PC	PC		Corner. In: along the chord to node 3. Out: following node 2's arc.		
4	PC	MC	MC	PC		Corner. In: following node 3's arc. Out: following node 2's arc.		
5	PC	PC	PC	MC		Corner. In: along the chord to node 3. Out: along the chord to node 2.		

#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
6	PC	PC	MC	MC		Corner. In: along the chord to node 3. Out: along the chord to node 2.		
7	PC	MC	PC	MC		Corner. In: along the chord to node 3. Out: following node 2's arc.		
8	PC	MC	MC	MC		Corner. In: along the chord to node 3. Out: following node 2's arc.		

#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
9	MC	PC	PC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
10	MC	PC	MC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
11	MC	MC	PC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		

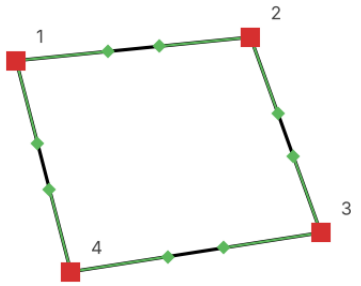
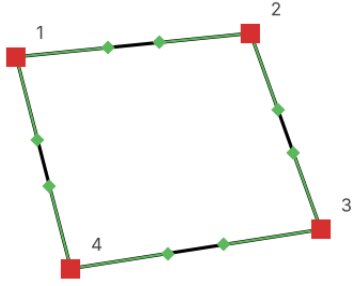
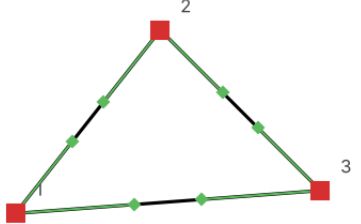
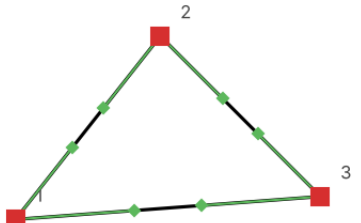
#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
12	MC	MC	MC	PC		Smooth, handles collinear. In: on the curve. Out: on the curve.		
13	MC	PC	PC	MC		cuspl. In: along the chord to node 3. Out: along the chord to node 2.		
14	MC	PC	MC	MC		cuspl. In: along the chord to node 3. Out: along the chord to node 2.		

#	Start	2nd	3rd	Close	Bezigon 4.2.0 / 2.2.0	How the start node ends	FreeHand	Notes
15	MC	MC	PC	MC		cusp. In: along the chord to node 3. Out: along the chord to node 2.		
16	MC	MC	MC	MC		cusp. In: along the chord to node 3. Out: along the chord to node 2.		

The start node follows exactly the pattern of the four-node table: a plain start is a plain corner, a modifier close reaches its in-handle straight back, and a modifier start with a modifier close is a cusp.

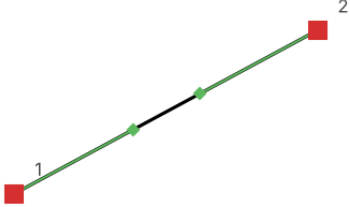
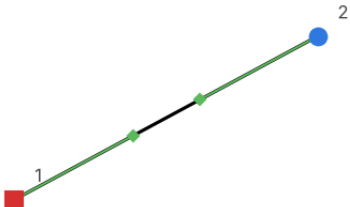
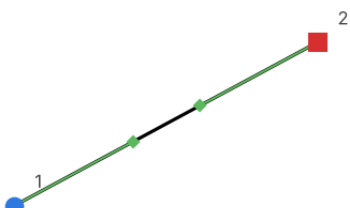
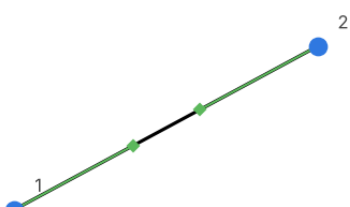
3. Polygons

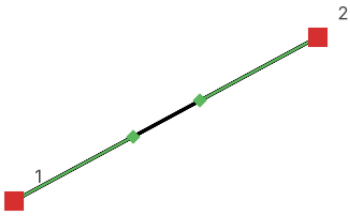
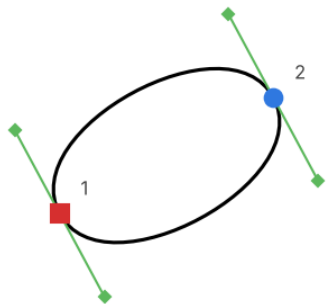
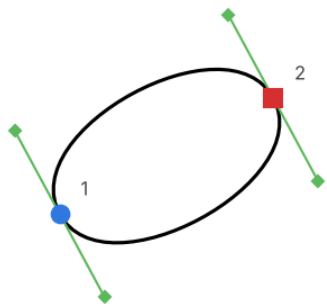
Every node a plain click: no node is smooth, so nothing curves.

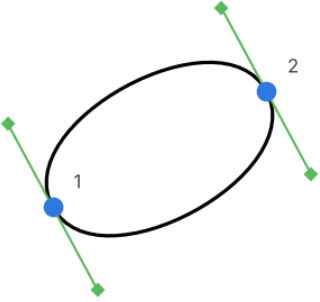
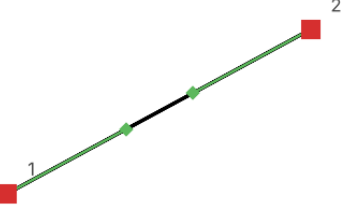
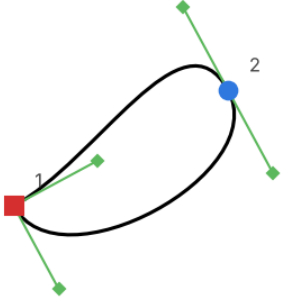
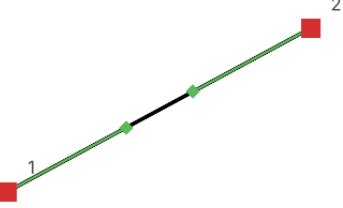
#	Clicks	Close	Bezigon 4.2.0 / 2.2.0	What it shows	FreeHand	Notes
1	PC, PC, PC, PC	PC		Four corners, plain close. Every handle lies along its chord, so every side is a straight line.		
2	PC, PC, PC, PC	MC		Four corners, modifier close. The same square: a start node drawn as a corner is not changed by the close.		
3	PC, PC, PC	PC		Three corners, plain close. A triangle; every side straight.		
4	PC, PC, PC	MC		Three corners, modifier close. The same triangle.		

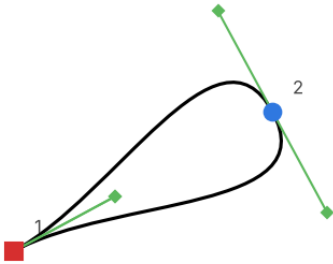
4. Two nodes

Two nodes left open, closed with a plain click and closed with the modifier. A closed path of two nodes has two segments, from node 1 to node 2 and back. The nodes sit at (150,330) and (520,130).

#	1st	2nd	Close	Bezigon 4.2.0 / 2.2.0	Shape	Node 1	Node 2	FreeHand	Notes
1	PC	PC	open		A straight line.	Corner. In: none. Out: along the chord to node 2.	Corner. In: along the chord to node 1. Out: none.		
2	PC	MC	open		A straight line.	Corner. In: none. Out: along the chord to node 2.	Smooth. In: along the chord to node 1. Out: none.		
3	MC	PC	open		A straight line.	Smooth. In: none. Out: along the chord to node 2.	Corner. In: along the chord to node 1. Out: none.		
4	MC	MC	open		A straight line.	Smooth. In: none. Out: along the chord to node 2.	Smooth. In: along the chord to node 1. Out: none.		

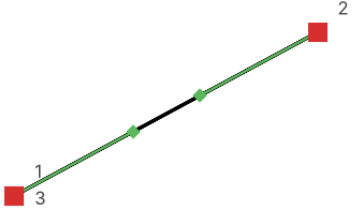
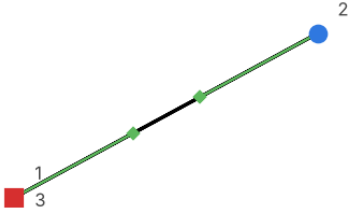
#	1st	2nd	Close	Bezigon 4.2.0 / 2.2.0	Shape	Node 1	Node 2	FreeHand	Notes
5	PC	PC	PC		A doubled straight line: both corners lie on the chord.	Corner. In: along the chord to node 2. Out: along the chord to node 2.	Corner. In: along the chord to node 1. Out: along the chord to node 1.		
6	PC	MC	PC		An ellipse. The corner continues the smooth node's arc.	Corner. In: square to the chord. Out: square to the chord.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.		
7	MC	PC	PC		An ellipse. The corner continues the smooth node's arc.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.	Corner. In: square to the chord. Out: square to the chord.		

#	1st	2nd	Close	Bezigon 4.2.0 / 2.2.0	Shape	Node 1	Node 2	FreeHand	Notes
8	MC	MC	PC		An ellipse, as in Xara.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.		
9	PC	PC	MC		A doubled straight line.	Corner. In: along the chord to node 2. Out: along the chord to node 2.	Corner. In: along the chord to node 1. Out: along the chord to node 1.		
10	PC	MC	MC		A teardrop. The modifier close sends the start node's in-handle straight back to node 2.	Corner. In: along the chord to node 2. Out: square to the chord.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.		
11	MC	PC	MC		A doubled straight line. The start node became a cusp, which lies on the chord.	cusp. In: along the chord to node 2. Out: along the chord to node 2.	Corner. In: along the chord to node 1. Out: along the chord to node 1.		

#	1st	2nd	Close	Bezigon 4.2.0 / 2.2.0	Shape	Node 1	Node 2	FreeHand	Notes
12	MC	MC	MC		A teardrop. The start node became a cusp; node 2 stays square to the chord.	cusp. In: along the chord to node 2. Out: along the chord to node 2.	Smooth, handles collinear. In: square to the chord. Out: square to the chord.		

5. Out and back

Paths whose end node sits exactly on the start node, left open, and one closed loop with two coincident interior nodes. The library treats these as ordinary nodes that happen to coincide. In the demo, releasing the end node on the start node merges the two and closes the path, which gives the two-node loops of section 4.

#	Clicks	Close	Bezigon 4.2.0 / 2.2.0	What happens	FreeHand	Notes
1	PC, PC, PC	open		The end node sits on the start node. Nothing special happens: a straight line out and back.		
2	PC, MC, PC	open		Node 2 is smooth, but its two neighbours are one point, so there is no circle through them. Its handles lie along the chords: still a straight line.		

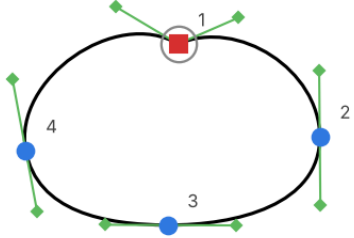
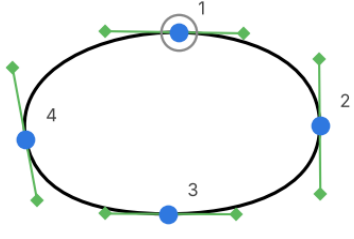
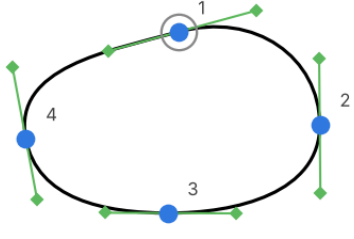
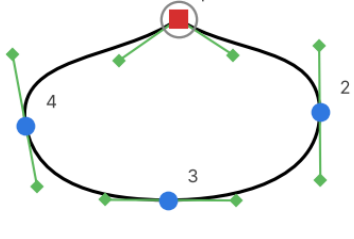
#	Clicks	Close	Bezigon 4.2.0 / 2.2.0	What happens	FreeHand	Notes
3	MC, PC, MC	open		As row 1, with smooth end nodes. An end node points along its one chord.		
4	MC, MC, MC	open		As row 2. Only a two-node loop takes the perpendicular; this path is open and has three nodes.		
5	PC, PC, MC, PC, MC	PC		A closed loop in which node 4 sits on node 2. Node 3's neighbours coincide, so its handles lie along the chords. The rest of the loop is unaffected.		

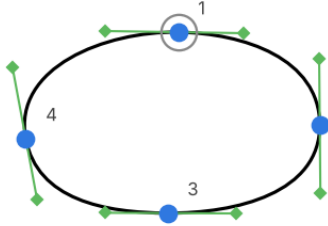
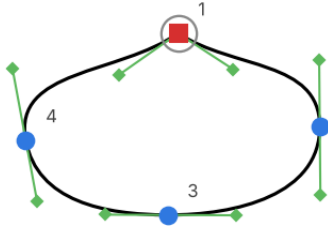
6. Edits after drawing

The tables above show shapes as drawn and closed. These sequences follow one path through later edits: conversions, dragged handles and released handles. Each row is the state after its action.

Sequence 1. A node drawn as a Corner between Smooth nodes

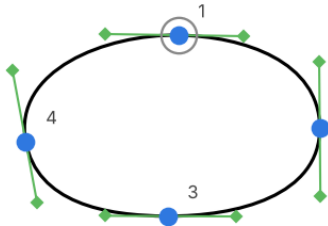
Your screenshots 1 to 3. Drawn with a plain click, node 1 is a plain corner and continues its neighbours' arcs, which makes the dip. Made smooth and then a corner again, it is a cusp, with both handles along its chords. From then on it toggles between smooth and the same cusp.

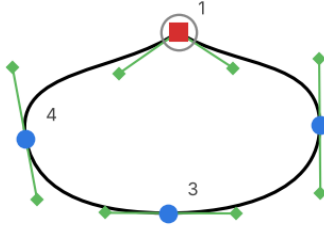
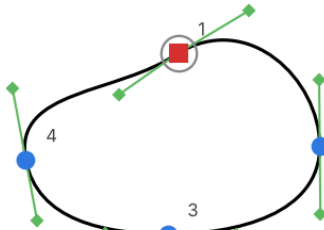
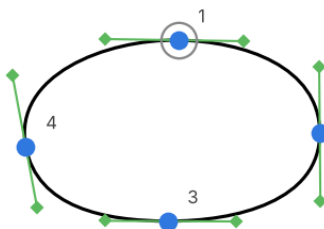
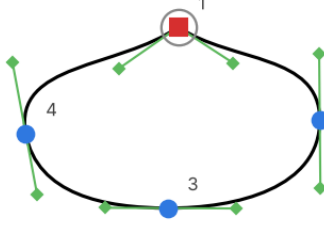
Step	Action	Call	Bezieron 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	Placed with a plain click, path closed with a plain click			Node 1: Corner. In: following node 4's arc. Out: following node 2's arc.		
2	Node 1 converted to Smooth	<code>set_node_type(path, 0, Smooth)</code>		Node 1: Smooth, handles collinear. In: on the curve. Out: on the curve.		
3	Node 1 out-handle dragged	<code>drag_handle(path, 0, false, x, y)</code>		Node 1: Smooth, handles collinear. In: dragged. Out: dragged.		
4	Node 1 converted to Corner	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
5	Node 1 converted to Smooth again	<code>set_node_type(path, 0, Smooth)</code>		Node 1: Smooth, handles collinear. In: on the curve. Out: on the curve.		
6	Node 1 converted to Corner again	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		

Sequence 2. A node drawn Smooth

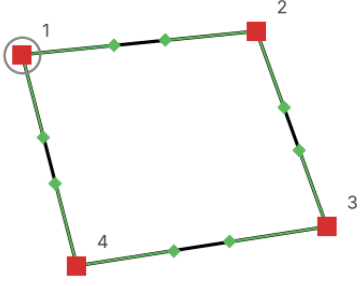
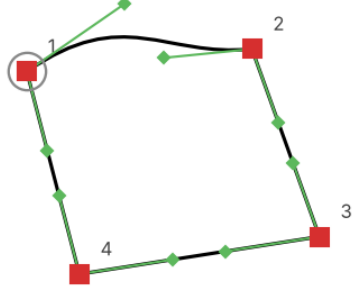
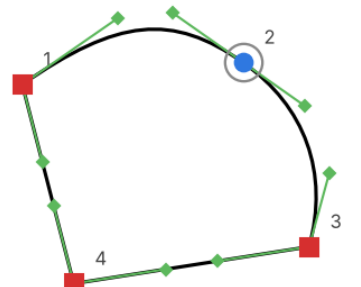
A cusp's two handles move independently. Converting to Smooth discards the dragged handle, and the cusp after it lies on its chords again.

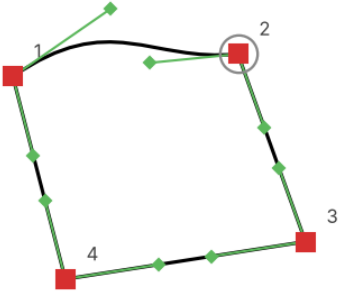
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	All four placed with a modifier click, plain close			Node 1: Smooth, handles collinear. In: on the curve. Out: on the curve.		

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
2	Node 1 converted to Corner	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		
3	Node 1 out-handle dragged	<code>drag_handle(path, 0, false, x, y)</code>		Node 1: cusp. In: along the chord to node 4. Out: dragged.		
4	Node 1 converted to Smooth	<code>set_node_type(path, 0, Smooth)</code>		Node 1: Smooth, handles collinear. In: on the curve. Out: on the curve.		
5	Node 1 converted to Corner again	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		

Sequence 3. Four corners, a dragged handle, then a conversion beside it

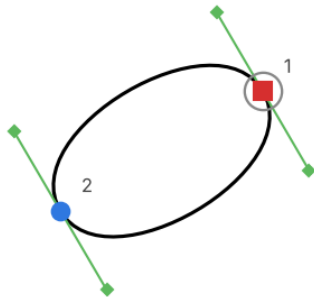
Node 1's dragged handle survives everything that happens to node 2. Node 2, made smooth and then a corner, is a cusp; beside corners a cusp looks the same as a plain corner, because there is no arc to follow.

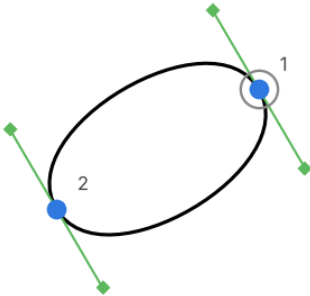
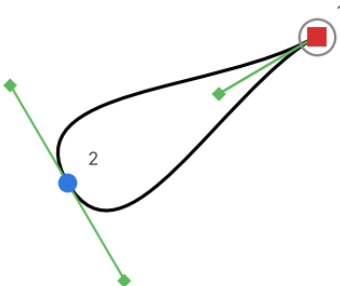
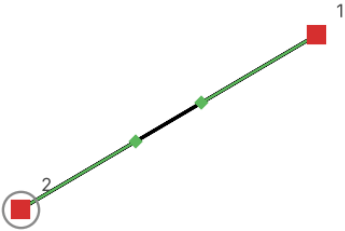
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	Four plain clicks, plain close			Node 1: Corner. In: along the chord to node 4. Out: along the chord to node 2. Node 2: Corner. In: along the chord to node 1. Out: along the chord to node 3.		
2	Node 1 out-handle dragged	<code>drag_handle(path, 0, false, x, y)</code>		Node 1: Corner. In: along the chord to node 4. Out: dragged. Node 2: Corner. In: along the chord to node 1. Out: along the chord to node 3.		
3	Node 2 converted to Smooth	<code>set_node_type(path, 1, Smooth)</code>		Node 1: Corner. In: along the chord to node 4. Out: dragged. Node 2: Smooth, handles collinear. In: on the curve. Out: on the curve.		

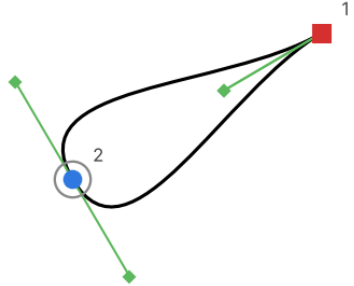
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
4	Node 2 converted to Corner	<code>set_node_type(path, 1, Corner)</code>		Node 1: Corner. In: along the chord to node 4. Out: dragged. Node 2: cusp. In: along the chord to node 1. Out: along the chord to node 3.		

Sequence 4. A two-node loop

Your screenshot 4. Node 1, drawn as a corner, continues node 2's arc, which in a two-node loop is the ellipse, so converting it to Smooth looks the same. Made a corner again it is a cusp along the chord, which turns the ellipse into a teardrop.

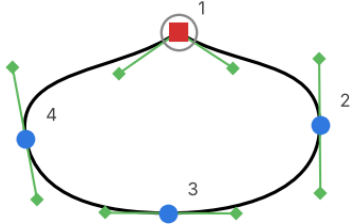
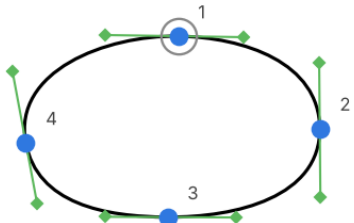
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	Plain click, modifier click, plain close			Node 1: Corner. In: square to the chord. Out: square to the chord. Node 2: Smooth, handles collinear. In: square to the chord. Out: square to the chord.		

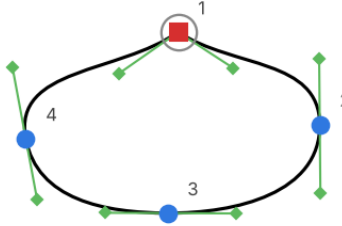
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
2	Node 1 converted to Smooth	<code>set_node_type(path, 0, Smooth)</code>		Node 1: Smooth, handles collinear. In: square to the chord. Out: square to the chord. Node 2: Smooth, handles collinear. In: square to the chord. Out: square to the chord.		
3	Node 1 converted to Corner	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 2. Out: along the chord to node 2. Node 2: Smooth, handles collinear. In: square to the chord. Out: square to the chord.		
4	Node 2 converted to Corner	<code>set_node_type(path, 1, Corner)</code>		Node 1: cusp. In: along the chord to node 2. Out: along the chord to node 2. Node 2: cusp. In: along the chord to node 1. Out: along the chord to node 1.		

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
5	Node 2 converted to Smooth	<code>set_node_type(path, 1, Smooth)</code>		Node 1: cusp. In: along the chord to node 2. Out: along the chord to node 2. Node 2: Smooth, handles collinear. In: square to the chord. Out: square to the chord.		

Sequence 5. The start node of a modifier close

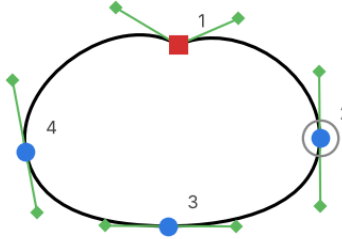
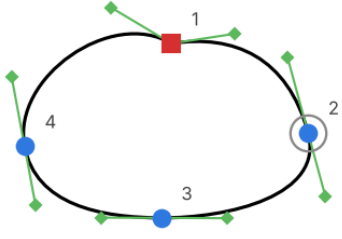
The modifier close makes the start node a cusp. It can be made smooth, which gives the shape of a plain close, and turned back into the same cusp.

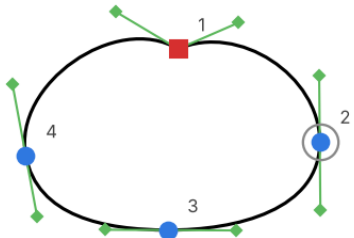
Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	All four modifier clicks, modifier close			Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		
2	Node 1 converted to Smooth	<code>set_node_type(path, 0, Smooth)</code>		Node 1: Smooth, handles collinear. In: on the curve. Out: on the curve.		

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
3	Node 1 converted to Corner	<code>set_node_type(path, 0, Corner)</code>		Node 1: cusp. In: along the chord to node 4. Out: along the chord to node 2.		

Sequence 6. A Corner follows its Smooth neighbour's dragged handle

Node 1 is a plain corner beside smooth node 2. When node 2's handle is dragged, node 1's handle follows at once, because `drag_handle` recomputes the path. Clearing the manual flags (the demo's R key) returns everything to the automatic shape.

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
1	Plain click then three modifier clicks, plain close			Node 1: Corner. In: following node 4's arc. Out: following node 2's arc. Node 2: Smooth, handles collinear. In: on the curve. Out: on the curve.		
2	Node 2 in-handle dragged	<code>drag_handle(path, 1, true, x, y)</code>		Node 1: Corner. In: following node 4's arc. Out: following node 2's arc. Node 2: Smooth, handles collinear. In: dragged. Out: dragged.		

Step	Action	Call	Bezigon 4.2.0 / 2.2.0	The nodes now	FreeHand	Notes
3	Dragged handles released	manual flags cleared, compute_handles		Node 1: Corner. In: following node 4's arc. Out: following node 2's arc. Node 2: Smooth, handles collinear. In: on the curve. Out: on the curve.		