import { useCallback, useState } from "react";

import { ReactFlow, applyEdgeChanges, applyNodeChanges } from "@xyflow/react";

import "[@xyflow/react/dist/style.css](mailto:@xyflow/react/dist/style.css)";

import initialNodes from "./Nodes.jsx";

import initialEdges from "./Edges.jsx";

function Flow() {

const [nodes, setNodes] = useState(initialNodes);

const [edges, setEdges] = useState(initialEdges);

const onNodesChange = useCallback(

(changes) => setNodes((nds) => applyNodeChanges(changes, nds)),

[setNodes]

);

const onEdgesChange = useCallback(

(changes) => setEdges((eds) => applyEdgeChanges(changes, eds)),

[setEdges]

);

return (

<ReactFlow

nodes={nodes}

edges={edges}

onNodesChange={onNodesChange}

onEdgesChange={onEdgesChange}

fitView

/>

);

}

export default Flow;

In the onNodesChange function, the changes parameter represents the latest changes to the nodes. This parameter is provided by the ReactFlow component when it calls the onNodesChange callback. Here's a step-by-step explanation of how changes gets the latest changes:

**1.**

**ReactFlow Component**:

The ReactFlow component is a library component that manages the state of nodes and edges in a flow diagram.

It provides various callback props, such as onNodesChange, which are called when the state of the nodes changes (e.g., when a node is moved, added, or deleted).

**2.**

**Callback Invocation**:

When a change occurs in the nodes (e.g., a user drags a node to a new position), the ReactFlow component detects this change and prepares a changes object that describes what has changed.

The changes object typically contains information about the type of change (e.g., position change, addition, deletion) and the affected nodes.

**3.**

**Passing Changes to Callback**:

The ReactFlow component then calls the onNodesChange callback, passing the changes object as an argument.

This is where the changes parameter in your onNodesChange function gets its value.

**4.**

**Handling Changes**:

Inside the onNodesChange function, you use the changes parameter to update the state of the nodes.

The applyNodeChanges function is a utility provided by the @xyflow/react library that takes the changes object and the current state of the nodes (nds), and returns the new state of the nodes after applying the changes.

UpdateNodeData() from useReactFlow helps update the current data for a given node

**How to get Handles and their Node Connections**

\*\*useHandleConnections: returns an array of connections on a specific handle or handle type, each with a source and target node id

\*\*useNodesData: returns the data object for a specific node id, if it exists in the nodes