FLOW

1. Xử lý Di Chuyển giữa các Ô (Arrow keys / Tab / Enter)

Mục tiêu:

- Khi nhấn **mũi tên**, **Tab** hoặc **Enter**, chọn ô khác tương ứng.
- Giống trải nghiệm di chuyển trong Excel/Google Sheets.

Cách xử lý:

a) Bắt sự kiện phím (`keydown`)

- Dùng `window.addEventListener('keydown', handleKeyDown)`.
- `handleKeyDown(event)` sẽ đọc `event.key`:
  - `"ArrowUp"`, `"ArrowDown"`, `"ArrowLeft"`, `"ArrowRight"`, `"Tab"`, `"Enter"`

b) Tính toán ô mới cần chọn

- Lấy vị trí ô hiện tại (`currentRow`, `currentCol`) từ `selectedCell`.
- Khi người dùng nhấn phím:
  - `ArrowUp`: `currentRow -= 1`
  - `ArrowDown`: `currentRow += 1`
  - `ArrowLeft`: `currentCol -= 1`
  - `ArrowRight`: `currentCol += 1`
  - `Tab`: `currentCol += 1`
  - `Enter`: `currentRow += 1`

> Ghi chú: Nếu qua biên (ví dụ: sang trái ở cột 0), cần giới hạn trong bảng.

c) Update ô mới

- Tìm `newCellId` theo hàng/cột mới.
- Gọi `selectCell(newCellId)`.
- Focus vào input để người dùng gõ ngay.

📌 2. Xử lý Copy - Paste (Ctrl+C / Ctrl+V)

2.1 Copy (Ctrl+C)

Mục tiêu:

- Khi người dùng nhấn **Ctrl+C**, lưu các giá trị ô đã chọn vào clipboard.

Cách xử lý:

a) Bắt sự kiện `keydown`

- Nếu `event.ctrlKey && event.key === 'c'` → chặn mặc định (`event.preventDefault()`).

b) Xây dựng dữ liệu để copy

- Với `selectedRange` (vùng nhiều ô được chọn), lặp qua tất cả các ô.
- Tạo một **chuỗi dạng bảng**:
  - Mỗi dòng là 1 hàng (`\n` ngắt dòng).
  - Mỗi ô trong dòng ngăn cách bằng `\t` (tab).

  Ví dụ:

  ```
  A1\tB1\tC1\n
  A2\tB2\tC2\n
  ```

c) Ghi vào clipboard

- Gọi `navigator.clipboard.writeText(formattedContent)`.

✅ **Sau đó**, người dùng có thể paste vào file text hoặc nơi khác.

2.2 Paste (Ctrl+V)

Mục tiêu:

- Khi nhấn **Ctrl+V**, lấy dữ liệu từ clipboard và dán vào bảng.

Cách xử lý:

a) Bắt sự kiện `keydown`

- Nếu `event.ctrlKey && event.key === 'v'` → chặn mặc định.

b) Đọc dữ liệu clipboard

- Gọi `navigator.clipboard.readText()`
- Kết quả là 1 chuỗi văn bản.

c) Parse dữ liệu

- Split chuỗi thành các dòng (`split('\n')`).
- Split mỗi dòng thành các ô (`split('\t')`).

d) Ghi dữ liệu vào bảng

- Với mỗi giá trị, ghi đè vào ô tương ứng:
  - Bắt đầu từ `selectedCell` hiện tại.
  - Tăng `row` / `col` theo dữ liệu paste.

e) Lưu trạng thái (undo)

- Trước khi paste, lưu bảng hiện tại vào `undoStack`.

🎯 Tóm tắt nhanh

Action	Diễn giải
Di chuyển	`keydown` → tính hàng/cột mới → `selectCell(newCellId)`
Copy (Ctrl+C)	Tạo chuỗi từ ô chọn → `navigator.clipboard.writeText()`
Paste (Ctrl+V)	`navigator.clipboard.readText()` → parse dữ liệu → ghi vào bảng

3. Xử lý Ctrl+Z (Undo) và Ctrl+Y (Redo)

3.1 Mục tiêu:

- **Ctrl+Z**: Quay lại trạng thái bảng trước đó (hoàn tác).
- **Ctrl+Y**: Làm lại hành động vừa hoàn tác (redo).

Tương tự Google Sheets, Excel!

3.2 Cấu trúc dữ liệu:

Cần 2 ngăn xếp (stack):

Stack	Ý nghĩa
`undoStack`	Lưu **các trạng thái trước đó** để undo
`redoStack`	Lưu **các trạng thái đã undo** để redo lại

3.3 Chi tiết xử lý

a) Khi người dùng thay đổi dữ liệu:

- **Trước khi thay đổi**, **copy bảng hiện tại** và **push vào** `undoStack`.
- **Sau khi thay đổi**, **clear** `redoStack` (vì có hành động mới rồi).

```
javascript

undoStack.push(cloneDeep(currentTableData));  // cloneDeep: sao chép sâu
redoStack = [];  // clear redo
```

---

b) Ctrl+Z (Undo)

- Bắt sự kiện `keydown`.
- Nếu `event.ctrlKey && event.key === 'z'`

- **Chặn mặc định** `event.preventDefault()`
- Kiểm tra `undoStack` có dữ liệu không?
  - Nếu có:
    1. Lưu bảng hiện tại vào `redoStack`.
    2. Lấy trạng thái gần nhất từ `undoStack.pop()`.
    3. Update lại bảng bằng dữ liệu đó.
  - Nếu không: không làm gì.

```
javascript

if (undoStack.length > 0) {
  redoStack.push(cloneDeep(currentTableData));
  const previousState = undoStack.pop();
  currentTableData = previousState;
  renderTable();
}
```

---

c) Ctrl+Y (Redo)

- Bắt sự kiện `keydown`.
- Nếu `event.ctrlKey && event.key === 'y'`
  - **Chặn mặc định** `event.preventDefault()`
  - Kiểm tra `redoStack` có dữ liệu không?
    - Nếu có:
      1. Lưu bảng hiện tại vào `undoStack`.
      2. Lấy trạng thái từ `redoStack.pop()`.
      3. Update bảng theo dữ liệu đó.
    - Nếu không: không làm gì.

```javascript
if (redoStack.length > 0) {
  undoStack.push(cloneDeep(currentTableData));
  const nextState = redoStack.pop();
  currentTableData = nextState;
  renderTable();
}
```

3.4 Lưu ý quan trọng

- **Mỗi lần thay đổi ô, paste, delete,...** đều phải lưu trước vào `undoStack`.
- **Redo chỉ hoạt động sau khi undo** (không có redo nếu chưa undo).
- **cloneDeep()** rất quan trọng → đảm bảo bạn lưu bản copy, không lưu tham chiếu!

🎯 Tóm tắt nhanh

Phím bấm	Ý nghĩa	Các bước làm
Ctrl+Z	Hoàn tác thay đổi	Lưu bảng hiện tại vào redo → lấy trạng thái undo → render lại
Ctrl+Y	Làm lại hành động vừa hoàn tác	Lưu bảng hiện tại vào undo → lấy trạng thái redo → render lại