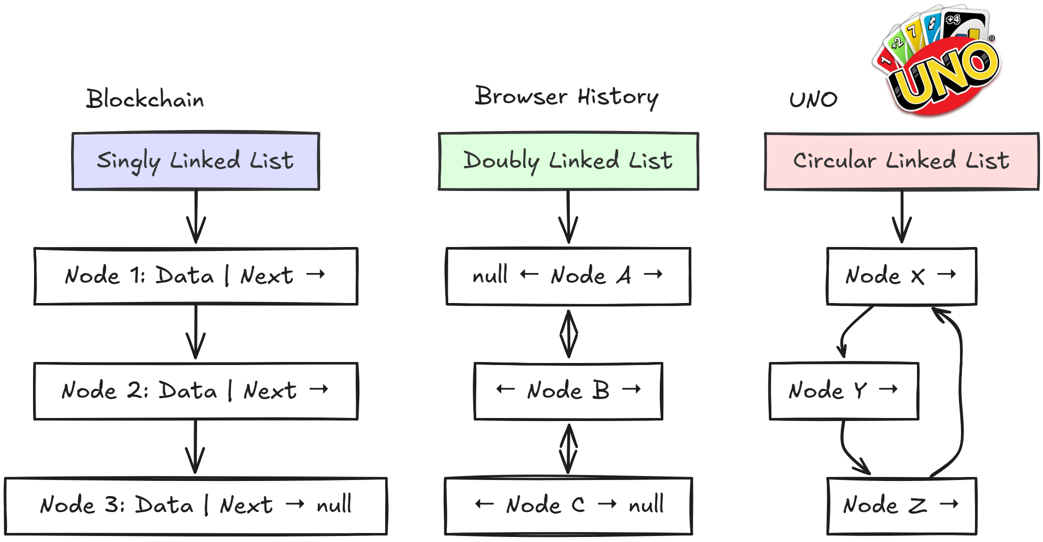


Blockchain

Singly Linked List

Node 1: Data | Next →

Node 2: Data | Next →

Node 3: Data | Next → null

Browser History

Doubly Linked List

null ← Node A →

← Node B →

← Node C → null

UNO

Circular Linked List

Node X →

Node Y →

Node Z →

🔗 1. Singly Linked List

Each node has two parts:

Data — the value (e.g., a name, a number)

Next — a pointer to the next node

➕ Key Points:

One-way direction only

Starts from the head and goes till null

Simple and memory efficient

💡 Example:

Think of a playlist where you can only skip to the next song. You can't go back, only forward.

🔄 2. Doubly Linked List

Each node has three parts:

Prev — pointer to the previous node

Data

Next — pointer to the next node

➕ Key Points:

You can move forward and backward

Needs more memory to store two pointers

Great for back-and-forth navigation

💡 Example:

A web browser history. You can go back to the last page or forward to the next one.

🔁 3. Circular Linked List

This one forms a loop — the last node doesn't point to null, it points back to the first node.

There are:

Singly Circular Linked Lists (one-directional)

Doubly Circular Linked Lists (two-directional)

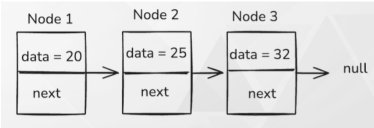
➕ Key Points:

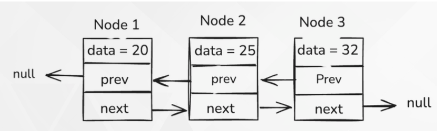
No true "end" — you can loop endlessly

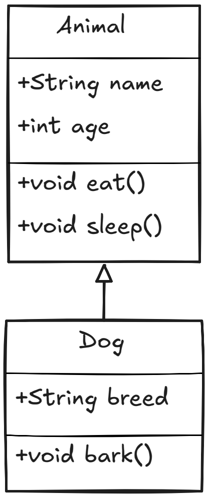
Great for round-robin scheduling

💡 Example:

A music playlist on repeat. After the last song, it goes back to the first automatically.







Animal

+String name

+int age

+void eat()

+void sleep()

Dog

+String breed

+void bark()