Flowchart (left)

```
┌─────────────────┐
│  page loads -   │
│   preloader?    │
└─────────────────┘
         │
         ▽
      ╱──────╲
    ╱  game is  ╲
   ◇ automatically ◇
    ╲  started  ╱
      ╲──────╱
         │
         │
┌──────────────┐     ╱─────────────────╲     ┌──────────────┐
│ Player 1 turn │◄───│  break when win  │───►│ Player 2 turn │
└──────────────┘     │  condition is met │    └──────────────┘
                      ╲─────────────────╱
                              │
                              ▼
                   ┌──────────────────┐
                   │ Winner announced, │
                   │ popup notification,│
                   │ game restarts when │
                   │   popup closes     │
                   └──────────────────┘
```

Flowchart (center)

```
┌──────────────┐
│  Game state  │
└──────────────┘
        │
        ▼
┌──────────────┐
│     true     │◄──────┐
└──────────────┘       │
        │              │
        ▼              │
┌──────────────┐       │
│     true     │       │
└──────────────┘       │
        │              │
        ▼              │
┌──────────────┐       │
│    false     │───────┘
└──────────────┘
```

Table (right, partially cut off)

Players take tu
Columns and R 7x6
game contain
win condition - 4 in a row horizontally/vert
finding column for piece t

rns	setCurrentPlayer(currentPlayer == "1" ? "2" : "1"
ows	7 arrays of 6 items column reverse
er	
s tically/diagonally	winning pieces light up when game is over - box inset glow

to be dropped	addPiece = (columnId) => {
	column = gamestate[columnId]
	piecePosition = column.indexOf(null)
	column[piecePosition] = currentPlayer;
	setGameState({
	...gameState,
	[columnId] : column)