Node.js ke cluster module mein kuch key methods, properties, aur events hote hain jo aapko multi-core systems ke upar load distribute karne mein madad karte hain. Yahaan cluster ke **important methods and properties** ki list di ja rahi hai:

**🔧 Methods**

| **Method** | **Description** |
| --- | --- |
| cluster.fork([env]) | Ek naya worker process start karta hai. Optional env object pass kar sakte ho. |
| cluster.setupPrimary([settings]) | Primary process ke liye default settings set karta hai. |
| cluster.disconnect([callback]) | Sabhi workers ko gracefully shutdown karta hai. |
| cluster.worker.disconnect() | Current worker ko disconnect karta hai. |
| worker.send(message[, sendHandle[, options]][, callback]) | Worker ko message bhejne ke liye. |

**📦 Properties**

| **Property** | **Description** |
| --- | --- |
| cluster.isPrimary | true agar current process primary (master) hai. (Node 16+ me isMaster deprecated) |
| cluster.isWorker | true agar current process ek worker hai. |
| cluster.settings | Current settings (port, exec, args, env, etc.) batata hai. |
| cluster.workers | Object containing all active workers by their ID. |

**🔔 Events**

| **Event** | **Description** |
| --- | --- |
| 'fork' | Jab bhi naya worker fork hota hai. |
| 'online' | Jab worker ready hota hai. |
| 'listening' | Jab worker kisi port pe sunna start karta hai. |
| 'disconnect' | Jab worker disconnect hota hai. |
| 'exit' | Jab worker exit hota hai. |
| 'message' | Jab worker ya primary message bhejta hai. |

**🎯 Real-world usage pattern:**

* cluster.fork() ka use loop me hota hai to utilize all CPU cores.
* Primary/Worker logic ko if (cluster.isPrimary) aur else block me likhte hain.
* cluster.on('exit', ...) ka use crash hone par worker ko respawn karne ke liye hota hai.