jsvelox — Complete Guide

Everything you need to know about using jsvelox

v2.2.0 — Codename: Quagga

github.com/yunus6141/jsvelox

// 01

What is jsvelox?

jsvelox is a lightweight JavaScript library that allows you to integrate AI providers — Claude, OpenAI, and Gemini — into your projects with just a few lines of code. Unlike heavy AI agent platforms that consume resources even for unused features, jsvelox follows a simple principle: only use what you need.

Built for developers who want AI integration without the overhead of visual pipeline builders or complex server setups. Install it, write a few lines, and your AI assistant is ready.

// 02

Problems jsvelox Solves

Too Complex	AI agent platforms require visual interfaces, complex configurations, and steep learning curves.	✔ jsvelox uses simple JavaScript objects — no drag-and-drop, no visual editors.
Too Heavy	Platforms like n8n consume RAM for every node, even unused ones.	✔ jsvelox only uses resources for what you actually call.
Too Many Files	Integrating AI usually means managing multiple API clients and authentication methods.	✔ One library, three AI providers, unified API.
Hard to Customize	Training an AI with your own data typically requires complex setups.	✔ systemPrompt and context let you train the AI in the same config object.

// 03

Code Structure

jsvelox is built around a single class: VeloxAI. Here is how it works:

Required Configuration

```
import VeloxAI from 'jsvelox'

const ai = new VeloxAI({
  provider: 'gemini',  // Required: claude | openai | gemini
  apiKey: 'your-key',  // Required: your API key
  memoryLimit: 10,      // Optional: max messages to remember
  timeout: 15000        // Optional: ms before timeout error
})

const reply = await ai.send('Hello!')
console.log(reply)
```

AI Training Configuration

```
const ai = new VeloxAI({
  provider: 'gemini',
  apiKey: 'your-key',
  systemPrompt: 'You are a shop assistant. Answer in English.',
  context: [
    { name: 'Product A', price: '$29', benefit: 'Boosts energy' },
    { name: 'Product B', price: '$49', benefit: 'Improves sleep' }
  ]
})
```

// 04

Available Methods

Method	Description
`send(message)`	Send a message and receive a response
`sendWithRetry(msg, n)`	Send with automatic retry on failure (default: 3 times)
`onSend(btn, input, res)`	Bind button/input/response in one line
`getMemory()`	Get full conversation history
`clearMemory()`	Clear conversation history

// 05

Common Errors & Solutions

ERROR	API key zorunlu! / API key required!
Cause:	You created VeloxAI without providing an apiKey.
Fix:	`Always pass apiKey in the config object:` `new VeloxAI({ provider: 'gemini', apiKey: 'your-key' })`

ERROR	Provider zorunlu! / Provider required!
Cause:	You did not specify which AI provider to use.
Fix:	`Add provider to config:` `new VeloxAI({ provider: 'gemini', apiKey: 'key' })`

ERROR	Gecersiz provider / Invalid provider
Cause:	You typed a provider name that is not supported. Example: provider: 'gpt'
Fix:	`Only use: claude | openai | gemini` `Note: jsvelox auto-converts to lowercase, so 'Claude' works too.`

ERROR	Mesaj bos olamaz! / Message cannot be empty!
Cause:	You called ai.send() with an empty string or whitespace.
Fix:	`Always check the message before sending:` `if (message.trim()) { await ai.send(message) }`

WARNING	Zaman asimi! / Timeout!
Cause:	The AI did not respond within the timeout period (default 30 seconds).

Fix:	`Increase timeout or use sendWithRetry:`
	`new VeloxAI({ timeout: 60000 })`
	`await ai.sendWithRetry(message, 3)`

ERROR	Claude hata: invalid x-api-key
Cause:	Your API key is wrong or expired.

Fix:	`Get a valid key from:`
	`- Claude: console.anthropic.com`
	`- OpenAI: platform.openai.com`
	`- Gemini: aistudio.google.com`

WARNING	**Quota exceeded**
Cause:	You have used all your free API quota for the day.
Fix:	`Wait for quota reset (usually midnight Pacific Time) or upgrade your plan. Use sendWithRetry to handle temporary quota errors.`

ERROR	**onSend: elements not found**
Cause:	The CSS selectors you passed to onSend() do not match any HTML elements.

Fix:	`Check your selectors:`
	`ai.onSend('#myBtn', '#myInput', '#myOutput')`
	`Make sure these IDs exist in your HTML.`

WARNING	**CORS Error (browser)**
Cause:	You are calling the API directly from a browser. Browsers block cross-origin requests to API servers.
Fix:	`Use jsvelox in a Node.js backend, not directly in the browser. A backend proxy will be supported in v3.0.`

// 06

Error Handling with try-catch

```
try {
  const reply = await ai.send('Hello!')
  console.log(reply)
} catch (err) {
  if (err.message.includes('Timeout')) {
    console.log('AI took too long, try again')
  } else if (err.message.includes('api-key')) {
    console.log('Check your API key')
  } else {
    console.log('Error:', err.message)
  }
}
```

// 07

When jsvelox Works Best

- ✓ Node.js backend applications
- ✓ Express.js or Fastify servers
- ✓ Next.js API routes
- ✓ CLI tools and scripts
- ✓ Any JavaScript/TypeScript project running on Node.js 18+

When jsvelox Does NOT Work

- ✗ Direct browser usage (CORS restriction) — coming in v3.0
- ✗ Deno or Bun environments (not tested)
- ✗ Node.js versions below 18 (fetch not built-in)

// 08

TypeScript Usage

```
import VeloxAI, { VeloxConfig } from 'jsvelox'

const config: VeloxConfig = {
  provider: 'gemini',
  apiKey: process.env.GEMINI_KEY!,
  memoryLimit: 20
}

const ai = new VeloxAI(config)
const reply: string = await ai.send('Hello!')
```