

# Package: aisdk.providers (via r-universe)

June 9, 2026

**Title** Additional Model Provider Adapters for the 'aisdk' Toolkit

**Version** 0.1.0

**Description** Additional AI model provider adapters for the 'aisdk' toolkit, covering OpenAI-compatible and Anthropic-compatible services such as 'DeepSeek', 'Moonshot'/Kimi', 'Stepfun', 'Volcengine', 'AiHubMix', 'xAI', 'OpenRouter', 'Bailian', and 'NVIDIA'. Providers register themselves with the core 'aisdk' provider registry on load.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Imports** aisdk (>= 1.4.12), R6, rlang, base64enc, jsonlite, curl, utils

**Suggests** withr, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**URL** <https://github.com/YuLab-SMU/aisdk.providers>

**BugReports** <https://github.com/YuLab-SMU/aisdk.providers/issues>

**Depends** R (>= 4.1.0)

**Config/pak/sysreqs** libssl-dev

**Repository** <https://yulab-smu.r-universe.dev>

**Date/Publication** 2026-06-02 10:42:13 UTC

**RemoteUrl** <https://github.com/yulab-smu/aisdk.providers>

**RemoteRef** HEAD

**RemoteSha** 3cfb54e6300b717ff589bdca0f4288f006fa6ed7

## Contents

AiHubMixProvider . . . . .	2
BailianProvider . . . . .	4

create_aihubmix . . . . .	5
create_aihubmix_anthropic . . . . .	6
create_aihubmix_gemini . . . . .	7
create_bailian . . . . .	8
create_deepseek . . . . .	10
create_deepseek_anthropic . . . . .	12
create_kimi_code . . . . .	13
create_kimi_code_anthropic . . . . .	14
create_moonshot . . . . .	15
create_nvidia . . . . .	17
create_openrouter . . . . .	18
create_stepfun . . . . .	20
create_volcengine . . . . .	22
create_xai . . . . .	24
DeepSeekProvider . . . . .	25
MoonshotProvider . . . . .	26
NvidiaProvider . . . . .	28
OpenRouterProvider . . . . .	29
StepfunProvider . . . . .	30
VolcengineProvider . . . . .	32
XAIProvider . . . . .	33

<b>Index</b>	<b>36</b>
--------------	-----------

---

AiHubMixProvider	<i>AiHubMix Provider Class</i>
------------------	--------------------------------

---

## Description

Provider class for AiHubMix.

## Super class

`aisdk::OpenAIProvider` -> AiHubMixProvider

## Methods

### Public methods:

- `AiHubMixProvider$new()`
- `AiHubMixProvider$language_model()`
- `AiHubMixProvider$image_model()`
- `AiHubMixProvider$clone()`

**Method** `new()`: Initialize the AiHubMix provider.

*Usage:*

```

AiHubMixProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)

```

*Arguments:*

*api\_key* AiHubMix API key. Defaults to AIHUBMIX\_API\_KEY env var.

*base\_url* Base URL. Defaults to https://aihubmix.com/v1.

*headers* Optional additional headers.

*timeout\_seconds* Legacy alias for *total\_timeout\_seconds*.

*total\_timeout\_seconds* Optional total request timeout in seconds for API calls.

*first\_byte\_timeout\_seconds* Optional time-to-first-byte timeout in seconds for API calls.

*connect\_timeout\_seconds* Optional connection-establishment timeout in seconds for API calls.

*idle\_timeout\_seconds* Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
AiHubMixProvider$language_model(model_id = NULL)
```

*Arguments:*

*model\_id* The model ID (e.g., "claude-sonnet-3-5", "claude-opus-3", "gpt-4o").

*Returns:* An AiHubMixLanguageModel object.

**Method** `image_model()`: Create an image model.

*Usage:*

```
AiHubMixProvider$image_model(model_id = NULL)
```

*Arguments:*

*model\_id* The model ID (e.g., "gpt-image-2").

*Returns:* An OpenAIImageModel object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
AiHubMixProvider$clone(deep = FALSE)
```

*Arguments:*

*deep* Whether to make a deep clone.

---

BailianProvider	<i>Bailian Provider Class</i>
-----------------	-------------------------------

---

### Description

Provider class for Alibaba Cloud Bailian / DashScope platform.

### Super class

`aisdk::OpenAIProvider` -> BailianProvider

### Methods

#### Public methods:

- `BailianProvider$new()`
- `BailianProvider$language_model()`
- `BailianProvider$clone()`

**Method** `new()`: Initialize the Bailian provider.

*Usage:*

```
BailianProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

`api_key` DashScope API key. Defaults to DASHSCOPE\_API\_KEY env var.

`base_url` Base URL. Defaults to <https://dashscope.aliyuncs.com/compatible-mode/v1>.

`headers` Optional additional headers.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds for API calls.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
BailianProvider$language_model(model_id = NULL)
```

*Arguments:*

model\_id The model ID (e.g., "qwen-plus", "qwen-turbo", "qwq-32b").

*Returns:* A BailianLanguageModel object.

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*

```
BailianProvider$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

---

 create\_aihubmix

 Create AiHubMix Provider
 

---

**Description**

Factory function to create an AiHubMix provider.

AiHubMix provides a unified API for various models including Claude, OpenAI, Gemini, etc.

**Usage**

```
create_aihubmix(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

**Arguments**

api_key	AiHubMix API key. Defaults to AIHUBMIX_API_KEY env var.
base_url	Base URL for API calls. Defaults to <a href="https://aihubmix.com/v1">https://aihubmix.com/v1</a> .
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

**Value**

An AiHubMixProvider object.

**Examples**

```
if (interactive()) {  
  aihubmix <- create_aihubmix()  
  model <- aihubmix$language_model("claude-sonnet-3-5")  
  result <- generate_text(model, "Explain quantum computing in one sentence.")  
}
```

---

```
create_aihubmix_anthropic
```

*Create AiHubMix Provider (Anthropic API Format)*

---

**Description**

Factory function to create an AiHubMix provider using the Anthropic-compatible API. This allows you to use AiHubMix Claude models with the native Anthropic API format, unlocking advanced features like Prompt Caching.

**Usage**

```
create_aihubmix_anthropic(  
  api_key = NULL,  
  extended_caching = FALSE,  
  headers = NULL,  
  timeout_seconds = NULL,  
  total_timeout_seconds = NULL,  
  first_byte_timeout_seconds = NULL,  
  connect_timeout_seconds = NULL,  
  idle_timeout_seconds = NULL  
)
```

**Arguments**

api_key	AiHubMix API key. Defaults to AIHUBMIX_API_KEY env var.
extended_caching	Logical. If TRUE, enables the 1-hour beta cache for Claude.
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.

first\_byte\_timeout\_seconds  
Optional time-to-first-byte timeout in seconds for API calls.

connect\_timeout\_seconds  
Optional connection-establishment timeout in seconds for API calls.

idle\_timeout\_seconds  
Optional stall timeout in seconds for API calls.

## Details

AiHubMix provides an Anthropic-compatible endpoint at <https://aihubmix.com/v1>. This convenience function wraps `create_anthropic()` with AiHubMix-specific defaults.

## Value

An `AnthropicProvider` object configured for AiHubMix.

## Examples

```
if (interactive()) {  
  # Use AiHubMix via Anthropic API format (unlocks caching)  
  aihubmix_claude <- create_aihubmix_anthropic()  
  model <- aihubmix_claude$language_model("claude-3-5-sonnet-20241022")  
  result <- generate_text(model, "Hello Claude!")  
}
```

---

create\_aihubmix\_gemini

*Create AiHubMix Provider (Gemini API Format)*

---

## Description

Factory function to create an AiHubMix provider using the Gemini-compatible API. This allows you to use Gemini models with the native Gemini API structure.

## Usage

```
create_aihubmix_gemini(  
  api_key = NULL,  
  headers = NULL,  
  timeout_seconds = NULL,  
  total_timeout_seconds = NULL,  
  first_byte_timeout_seconds = NULL,  
  connect_timeout_seconds = NULL,  
  idle_timeout_seconds = NULL  
)
```

## Arguments

api_key	AiHubMix API key. Defaults to AIHUBMIX_API_KEY env var.
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

## Details

AiHubMix provides a Gemini-compatible endpoint at <https://aihubmix.com/gemini/v1beta/models>. This convenience function wraps `create_gemini()` with AiHubMix-specific defaults.

## Value

A GeminiProvider object configured for AiHubMix.

## Examples

```
if (interactive()) {  
  # Use AiHubMix via Gemini API format  
  aihubmix_gemini <- create_aihubmix_gemini()  
  model <- aihubmix_gemini$language_model("gemini-2.5-flash")  
  result <- generate_text(model, "Hello Gemini!")  
}
```

---

create\_bailian

*Create Alibaba Cloud Bailian Provider*

---

## Description

Factory function to create an Alibaba Cloud Bailian provider using the DashScope API.

## Usage

```
create_bailian(  
  api_key = NULL,  
  base_url = NULL,  
  headers = NULL,  
  timeout_seconds = NULL,  
  total_timeout_seconds = NULL,  
  first_byte_timeout_seconds = NULL,  
  connect_timeout_seconds = NULL,  
  idle_timeout_seconds = NULL  
)
```

## Arguments

<code>api_key</code>	DashScope API key. Defaults to <code>DASHSCOPE_API_KEY</code> env var.
<code>base_url</code>	Base URL for API calls. Defaults to <code>https://dashscope.aliyuncs.com/compatible-mode/v1</code> .
<code>headers</code>	Optional additional headers.
<code>timeout_seconds</code>	Legacy alias for <code>total_timeout_seconds</code> .
<code>total_timeout_seconds</code>	Optional total request timeout in seconds for API calls.
<code>first_byte_timeout_seconds</code>	Optional time-to-first-byte timeout in seconds for API calls.
<code>connect_timeout_seconds</code>	Optional connection-establishment timeout in seconds for API calls.
<code>idle_timeout_seconds</code>	Optional stall timeout in seconds for API calls.

## Value

A `BailianProvider` object.

## Supported Models

- **qwen3.6-plus**: Latest Qwen flagship with 1M context. Strong reasoning and multimodal capabilities. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **qwen3.5-plus**: Balanced model with excellent reasoning at ultra-low cost. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **qwen-plus**: Most popular general-purpose model. Good balance of capability and cost. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **qwen-max**: Previous generation premium model for complex reasoning and analysis. (Reasoning, Vision, Tools, Structured) | ctx: 128k
- **qwen-turbo**: Fast and cost-efficient model for high-volume simple tasks. (Vision, Tools, Structured) | ctx: 128k

- **qwen-coder-plus**: Coding specialist optimized for agentic software engineering. (Reasoning, Tools, Structured) | ctx: 128k
- **qwq-32b**: 32B parameter reasoning model with strong math and logic capabilities. (Reasoning, Tools, Structured) | ctx: 128k
- **qwen-vl-plus**: Vision-language model for image understanding and visual reasoning. (Vision, Tools, Structured) | ctx: 32k
- **qwen3.5-omni-plus**: Multimodal model supporting text, image, video, and audio input. (Reasoning, Vision, Tools, Audio, Structured) | ctx: 32k
- **text-embedding-v3**: Text embedding model for similarity search and retrieval. | ctx: 8k

### Examples

```
if (interactive()) {
  bailian <- create_bailian()

  # Standard chat model
  model <- bailian$language_model("qwen-plus")
  result <- generate_text(model, "Hello")

  # Reasoning model (QwQ with chain-of-thought)
  model <- bailian$language_model("qwq-32b")
  result <- generate_text(model, "Solve: What is 15 * 23?")
  print(result$reasoning) # Chain-of-thought reasoning

  # Default model (qwen-plus)
  model <- bailian$language_model()
}
```

---

create\_deepseek

*Create DeepSeek Provider*

---

### Description

Factory function to create a DeepSeek provider.

### Usage

```
create_deepseek(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

**Arguments**

api_key	DeepSeek API key. Defaults to DEEPSEEK_API_KEY env var.
base_url	Base URL. Defaults to "https://api.deepseek.com".
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

**Details**

DeepSeek supports classic aliases plus newer model families such as DeepSeek V4.

Common model IDs include:

- **deepseek-chat**: Chat alias provided by DeepSeek
- **deepseek-reasoner**: Reasoning alias provided by DeepSeek
- **deepseek-v4\***: DeepSeek V4 family model IDs exposed by the API

Additional DeepSeek-specific request fields such as `thinking`, `thinking_budget`, and `reasoning_effort` are passed through when supplied to `$generate()` or `$stream()`.

**Value**

A DeepSeekProvider object.

**Supported Models**

- **deepseek-v4-flash**: DeepSeek V4 Flash hybrid-thinking model (Reasoning, Tools, Structured) | ctx: 1000k
- **deepseek-v4-pro**: DeepSeek V4 Pro hybrid-thinking model (Reasoning, Tools, Structured) | ctx: 1000k

**Examples**

```
if (interactive()) {
# Basic usage with deepseek-chat
deepseek <- create_deepseek()
model <- deepseek$language_model("deepseek-chat")
result <- generate_text(model, "Hello!")

# Using a reasoning-capable model
```

```

model_reasoner <- deepseek$language_model("deepseek-reasoner")
result <- model_reasoner$generate(
  messages = list(list(role = "user", content = "Solve: What is 15 * 23?")),
  max_tokens = 500,
  thinking = TRUE
)
print(result$text) # Final answer
print(result$reasoning) # Chain-of-thought reasoning

# Streaming with reasoning
stream_text(model_reasoner, "Explain quantum entanglement step by step")
}

```

---

```
create_deepseek_anthropic
```

*Create DeepSeek Provider (Anthropic API Format)*

---

## Description

Factory function to create a DeepSeek provider using the Anthropic-compatible API. This allows you to use DeepSeek models with the Anthropic API format.

## Usage

```

create_deepseek_anthropic(
  api_key = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)

```

## Arguments

<code>api_key</code>	DeepSeek API key. Defaults to DEEPSEEK_API_KEY env var.
<code>headers</code>	Optional additional headers.
<code>timeout_seconds</code>	Legacy alias for <code>total_timeout_seconds</code> .
<code>total_timeout_seconds</code>	Optional total request timeout in seconds for API calls.
<code>first_byte_timeout_seconds</code>	Optional time-to-first-byte timeout in seconds for API calls.
<code>connect_timeout_seconds</code>	Optional connection-establishment timeout in seconds for API calls.
<code>idle_timeout_seconds</code>	Optional stall timeout in seconds for API calls.

## Details

DeepSeek provides an Anthropic-compatible endpoint at <https://api.deepseek.com/anthropic>. This convenience function wraps `create_anthropic()` with DeepSeek-specific defaults.

Note: When using an unsupported model name, the API backend will automatically map it to `deepseek-chat`.

## Value

An `AnthropicProvider` object configured for DeepSeek.

## Examples

```
if (interactive()) {
  # Use DeepSeek via Anthropic API format
  deepseek <- create_deepseek_anthropic()
  model <- deepseek$language_model("deepseek-chat")
  result <- generate_text(model, "Hello!")

  # This is useful for tools that expect Anthropic API format
  # such as Claude Code integration
}
```

---

create_kimi_code	<i>Create Kimi Code Provider</i>
------------------	----------------------------------

---

## Description

Convenience wrapper for Kimi Code membership API. By default this uses the Anthropic-compatible endpoint because it works for self-built coding agents with their real User-Agent. Set `api_format = "openai"` when integrating with OpenAI-compatible tools that Kimi Code recognizes as coding agents.

## Usage

```
create_kimi_code(
  api_key = NULL,
  base_url = NULL,
  api_format = c("anthropic", "openai"),
  headers = NULL,
  prompt_cache_key = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

**Arguments**

api_key	API key. Defaults to MOONSHOT_API_KEY for the Kimi Open Platform, or KIMI_API_KEY / KIMI_CODE_API_KEY for Kimi Code.
base_url	Base URL for API calls.
api_format	API protocol to use: "anthropic" or "openai".
headers	Optional additional headers.
prompt_cache_key	Default prompt cache key for Kimi Code requests.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds.
first_byte_timeout_seconds	Optional time-to-first-byte timeout.
connect_timeout_seconds	Optional connection-establishment timeout.
idle_timeout_seconds	Optional stall timeout.

**Value**

A provider object configured for Kimi Code.

---

```
create_kimi_code_anthropic
```

*Create Kimi Code Provider (Anthropic API Format)*

---

**Description**

Convenience wrapper for Kimi Code's Anthropic-compatible endpoint. Use model ID `kimi-for-coding`. The public Kimi docs list the Anthropic base as `https://api.kimi.com/coding/`; aisdk's Anthropic provider appends `/messages` directly, so this wrapper normalizes to `https://api.kimi.com/coding/v1`.

**Usage**

```
create_kimi_code_anthropic(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

**Arguments**

api_key	API key. Defaults to MOONSHOT_API_KEY for the Kimi Open Platform, or KIMI_API_KEY / KIMI_CODE_API_KEY for Kimi Code.
base_url	Base URL for API calls.
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds.
first_byte_timeout_seconds	Optional time-to-first-byte timeout.
connect_timeout_seconds	Optional connection-establishment timeout.
idle_timeout_seconds	Optional stall timeout.

**Value**

An AnthropicProvider object configured for Kimi Code.

---

create_moonshot	<i>Create Moonshot / Kimi Provider</i>
-----------------	--

---

**Description**

Factory function to create a Moonshot provider. Use platform = "platform" for the pay-as-you-go Kimi Open Platform (<https://api.moonshot.cn/v1>) and platform = "coding" for Kimi Code membership API (<https://api.kimi.com/coding/v1>). The two platforms use separate API keys.

**Usage**

```
create_moonshot(
    api_key = NULL,
    base_url = NULL,
    platform = c("auto", "platform", "coding"),
    headers = NULL,
    prompt_cache_key = NULL,
    timeout_seconds = NULL,
    total_timeout_seconds = NULL,
    first_byte_timeout_seconds = NULL,
    connect_timeout_seconds = NULL,
    idle_timeout_seconds = NULL
)
```

**Arguments**

<code>api_key</code>	API key. Defaults to <code>MOONSHOT_API_KEY</code> for the Kimi Open Platform, or <code>KIMI_API_KEY</code> / <code>KIMI_CODE_API_KEY</code> for Kimi Code.
<code>base_url</code>	Base URL for API calls.
<code>platform</code>	API platform: "auto", "platform", or "coding".
<code>headers</code>	Optional additional headers.
<code>prompt_cache_key</code>	Default prompt cache key for Kimi Code requests.
<code>timeout_seconds</code>	Legacy alias for <code>total_timeout_seconds</code> .
<code>total_timeout_seconds</code>	Optional total request timeout in seconds.
<code>first_byte_timeout_seconds</code>	Optional time-to-first-byte timeout.
<code>connect_timeout_seconds</code>	Optional connection-establishment timeout.
<code>idle_timeout_seconds</code>	Optional stall timeout.

**Value**

A MoonshotProvider object.

**Supported Models**

- **kimi-k2.6**: Kimi flagship multimodal model for agentic coding, long-context reasoning, and general agent tasks (Reasoning, Vision, Tools, Structured) | ctx: 262k
- **kimi-k2.5**: Kimi K2.5 multimodal model for agent, coding, vision, thinking, and conversation tasks (Reasoning, Vision, Tools, Structured) | ctx: 262k
- **kimi-k2-0905-preview**: Kimi K2 preview model with enhanced agentic coding, frontend, and context understanding capabilities (Reasoning, Tools, Structured) | ctx: 262k
- **kimi-for-coding**: Stable Kimi Code membership model ID for coding agents; backend maps it to the latest coding model (Reasoning, Vision, Tools, Structured) | ctx: 262k
- **moonshot-v1-8k**: Moonshot V1 text generation model with 8k context (Tools, Structured) | ctx: 8k
- **moonshot-v1-32k**: Moonshot V1 text generation model with 32k context (Tools, Structured) | ctx: 33k
- **moonshot-v1-128k**: Moonshot V1 text generation model with 128k context (Tools, Structured) | ctx: 131k

**Examples**

```

if (interactive()) {
  moonshot <- create_moonshot()
  model <- moonshot$language_model("kimi-k2.6")
  result <- generate_text(model, "Hello", temperature = 1)

  kimi_code <- create_moonshot(platform = "coding")
  coding_model <- kimi_code$language_model()
  result <- generate_text(coding_model, "Review this function", prompt_cache_key = "task-1")
}

```

---

create\_nvidia

*Create NVIDIA Provider*


---

**Description**

Factory function to create a NVIDIA provider.

**Usage**

```

create_nvidia(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)

```

**Arguments**

api_key	NVIDIA API key. Defaults to NVIDIA_API_KEY env var.
base_url	Base URL. Defaults to "https://integrate.api.nvidia.com/v1".
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

**Value**

A NvidiaProvider object.

**Examples**

```
if (interactive()) {
  nvidia <- create_nvidia()
  model <- nvidia$language_model("z-ai/glm4.7")

  # Enable thinking/reasoning
  result <- generate_text(model, "Who are you?",
    chat_template_kwargs = list(enable_thinking = TRUE)
  )
  print(result$reasoning)
}
```

---

create\_openrouter      *Create OpenRouter Provider*

---

**Description**

Factory function to create an OpenRouter provider.

**Usage**

```
create_openrouter(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

**Arguments**

api_key	OpenRouter API key. Defaults to OPENROUTER_API_KEY env var.
base_url	Base URL for API calls. Defaults to https://openrouter.ai/api/v1.
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.

first\_byte\_timeout\_seconds

Optional time-to-first-byte timeout in seconds for API calls.

connect\_timeout\_seconds

Optional connection-establishment timeout in seconds for API calls.

idle\_timeout\_seconds

Optional stall timeout in seconds for API calls.

## Value

An OpenRouterProvider object.

## Supported Models

- **openai/gpt-5.5**: OpenAI's latest frontier model via OpenRouter. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **openai/gpt-5.4**: OpenAI's previous flagship with computer use and 1M context. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **anthropic/claude-opus-4-7**: Anthropic's most capable model via OpenRouter. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **anthropic/claude-sonnet-4-6**: Balanced Claude model with 1M context and strong coding. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **google/gemini-3.1-pro**: Google's most capable multimodal model via OpenRouter. (Reasoning, Vision, Tools, Audio, Structured, Search) | ctx: 1000k
- **google/gemini-2.5-flash**: Fast and cost-efficient Gemini with 1M context. (Reasoning, Vision, Tools, Audio, Structured, Search) | ctx: 1000k
- **xai/grok-4.1-fast**: xAI's fastest model with 2M context at aggressive pricing. (Reasoning, Vision, Tools, Structured, Search) | ctx: 2000k
- **xai/grok-4**: xAI frontier model with always-on reasoning. (Reasoning, Vision, Tools, Structured, Search) | ctx: 256k
- **deepseek/deepseek-chat**: DeepSeek's general-purpose chat model. (Vision, Tools, Structured) | ctx: 64k
- **deepseek/deepseek-reasoner**: DeepSeek's reasoning specialist (R1). (Reasoning, Tools, Structured) | ctx: 64k
- **meta-llama/llama-4-maverick**: Meta's open-weight frontier model with 10M context (Scout). (Reasoning, Vision, Tools, Structured) | ctx: 10000k
- **qwen/qwen3.6-plus**: Alibaba's latest Qwen flagship with 1M context. (Reasoning, Vision, Tools, Structured) | ctx: 1000k
- **moonshotai/kimi-k2**: Moonshot's agentic swarm model with 256K context. (Reasoning, Vision, Tools, Structured) | ctx: 256k
- **openai/gpt-4.1**: OpenAI's production workhorse with 1M context. (Vision, Tools, Structured) | ctx: 1000k
- **anthropic/claude-haiku-4-5**: Fast and cost-effective Claude for simple tasks. (Vision, Tools, Structured) | ctx: 200k

## Examples

```
if (interactive()) {
  openrouter <- create_openrouter()

  # Access any model via a unified API
  model <- openrouter$language_model("openai/gpt-4o")
  result <- generate_text(model, "Hello!")

  # Reasoning model
  model <- openrouter$language_model("deepseek/deepseek-r1")
  result <- generate_text(model, "Solve: 15 * 23")
  print(result$reasoning)
}
```

---

create\_stepfun

*Create Stepfun Provider*

---

## Description

Factory function to create a Stepfun provider.

## Usage

```
create_stepfun(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

## Arguments

api_key	Stepfun API key. Defaults to STEPFUN_API_KEY env var.
base_url	Base URL for API calls. Defaults to <a href="https://api.stepfun.com/v1">https://api.stepfun.com/v1</a> .
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.

connect\_timeout\_seconds

Optional connection-establishment timeout in seconds for API calls.

idle\_timeout\_seconds

Optional stall timeout in seconds for API calls.

### Value

A StepfunProvider object.

### Supported Models

- **step-1-32k**: Model: step-1-32k (Tools) | ctx: 32k
- **step-1v-32k**: Vision enabled model with 32k context (Vision, Tools) | ctx: 32k
- **step-1-8k**: Model: step-1-8k
- **step-1-256k**: Model: step-1-256k
- **step-1v-8k**: Model: step-1v-8k (Vision)
- **step-2-16k**: Model: step-2-16k
- **step-1x-medium**: Model: step-1x-medium
- **step-tts-mini**: Model: step-tts-mini (Audio)
- **step-2-16k-202411**: Model: step-2-16k-202411
- **step-asr**: Model: step-asr (Audio)
- **step-1o-vision-32k**: Model: step-1o-vision-32k (Vision)
- **step-2-mini**: Model: step-2-mini
- **step-2-16k-exp**: Model: step-2-16k-exp
- **step-1o-turbo-vision**: Model: step-1o-turbo-vision (Vision)
- **step-1o-audio**: Model: step-1o-audio (Audio)
- ... and 16 more models. Use `list_models("stepfun")` to see all.

### Examples

```
if (interactive()) {
  stepfun <- create_stepfun()
  model <- stepfun$language_model("step-1-8k")
  result <- generate_text(model, "Explain quantum computing in one sentence.")
}
```

---

create_volcengine	<i>Create Volcengine/Ark Provider</i>
-------------------	---------------------------------------

---

## Description

Factory function to create a Volcengine provider using the Ark API.

## Usage

```
create_volcengine(  
    api_key = NULL,  
    base_url = NULL,  
    headers = NULL,  
    timeout_seconds = NULL,  
    total_timeout_seconds = NULL,  
    first_byte_timeout_seconds = NULL,  
    connect_timeout_seconds = NULL,  
    idle_timeout_seconds = NULL  
)
```

## Arguments

api_key	Volcengine API key. Defaults to ARK_API_KEY env var.
base_url	Base URL for API calls. Defaults to <a href="https://ark.cn-beijing.volces.com/api/v3">https://ark.cn-beijing.volces.com/api/v3</a> .
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

## Value

A VolcengineProvider object.

## Supported Models

- **doubao-lite-128k-240428**: Model: doubao-lite-128k-240428
- **doubao-pro-128k-240515**: Model: doubao-pro-128k-240515
- **doubao-lite-4k-240328**: Model: doubao-lite-4k-240328

- **doubao-lite-32k-240428**: Model: doubao-lite-32k-240428
- **doubao-pro-4k-240515**: Model: doubao-pro-4k-240515
- **doubao-lite-4k-character-240515**: Model: doubao-lite-4k-character-240515
- **doubao-embedding-text-240515**: Model: doubao-embedding-text-240515
- **mistral-7b-instruct-v0.2**: Model: mistral-7b-instruct-v0.2
- **doubao-pro-4k-character-240515**: Model: doubao-pro-4k-character-240515
- **doubao-pro-4k-functioncall-240515**: Model: doubao-pro-4k-functioncall-240515
- **doubao-lite-4k-pretrain-character-240516**: Model: doubao-lite-4k-pretrain-character-240516
- **doubao-pro-32k-character-240528**: Model: doubao-pro-32k-character-240528
- **doubao-pro-4k-browsing-240524**: Model: doubao-pro-4k-browsing-240524
- **doubao-pro-32k-functioncall-240515**: Model: doubao-pro-32k-functioncall-240515
- **doubao-pro-4k-functioncall-240615**: Doubao Pro function-calling model (4K); suited to low-latency function-calling. (Tools, Structured) | ctx: 4k
- ... and 95 more models. Use `list_models("volcengine")` to see all.

### API Formats

Volcengine supports both Chat Completions API and Responses API:

- `language_model()`: Uses Chat Completions API (standard)
- `responses_model()`: Uses Responses API (for reasoning models)
- `smart_model()`: Auto-selects based on model ID

### Token Limit Parameters for Volcengine Responses API

Volcengine's Responses API has two mutually exclusive token limit parameters:

- `max_output_tokens`: Total limit including reasoning + answer (default mapping)
- `max_tokens` (API level): Answer-only limit, excluding reasoning

The SDK's unified `max_tokens` parameter maps to `max_output_tokens` by default, which is the **safe choice** to prevent runaway reasoning costs.

For advanced users who want answer-only limits:

- Use `max_answer_tokens` parameter to explicitly set answer-only limit
- Use `max_output_tokens` parameter to explicitly set total limit

### Examples

```
if (interactive()) {
  volcengine <- create_volcengine()

  # Chat API (standard models)
  model <- volcengine$language_model("doubao-1-5-pro-256k-250115")
  result <- generate_text(model, "Hello")
}
```

```

# Responses API (reasoning models like DeepSeek)
model <- volcengine$responses_model("deepseek-r1-250120")

# Default: max_tokens limits total output (reasoning + answer)
result <- model$generate(messages = msgs, max_tokens = 2000)

# Advanced: limit only the answer part (reasoning can be longer)
result <- model$generate(messages = msgs, max_answer_tokens = 500)

# Smart model selection (auto-detects best API)
model <- volcengine$smart_model("deepseek-r1-250120")
}

```

---

create\_xai

*Create xAI Provider*


---

### Description

Factory function to create an xAI provider.

### Usage

```

create_xai(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)

```

### Arguments

api_key	xAI API key. Defaults to XAI_API_KEY env var.
base_url	Base URL for API calls. Defaults to https://api.x.ai/v1.
headers	Optional additional headers.
timeout_seconds	Legacy alias for total_timeout_seconds.
total_timeout_seconds	Optional total request timeout in seconds for API calls.
first_byte_timeout_seconds	Optional time-to-first-byte timeout in seconds for API calls.
connect_timeout_seconds	Optional connection-establishment timeout in seconds for API calls.
idle_timeout_seconds	Optional stall timeout in seconds for API calls.

**Value**

A XAIProvider object.

**Supported Models**

- **grok-4**: xAI frontier model with always-on reasoning and multi-agent architecture. (Reasoning, Vision, Tools, Structured, Search) | ctx: 256k
- **grok-4.1-fast**: High-throughput production model with 2M context. Best default for most workloads. (Reasoning, Vision, Tools, Structured, Search) | ctx: 2000k
- **grok-3**: Legacy enterprise-grade model optimized for data extraction and summarization. (Reasoning, Vision, Tools, Structured, Search) | ctx: 131k
- **grok-3-mini**: Lightweight low-latency option with reasoning capabilities. (Reasoning, Tools, Structured, Search) | ctx: 131k
- **grok-2-image**: Image generation model via xAI API. (Image-Out) | ctx: 33k

**Examples**

```
if (interactive()) {
  xai <- create_xai()
  model <- xai$language_model("grok-beta")
  result <- generate_text(model, "Explain quantum computing in one sentence.")
}
```

---

DeepSeekProvider

*DeepSeek Provider Class*

---

**Description**

Provider class for DeepSeek.

**Super class**

[aisdk::OpenAIProvider](#) -> DeepSeekProvider

**Methods****Public methods:**

- [DeepSeekProvider\\$new\(\)](#)
- [DeepSeekProvider\\$language\\_model\(\)](#)
- [DeepSeekProvider\\$clone\(\)](#)

**Method** `new()`: Initialize the DeepSeek provider.

*Usage:*

```

DeepSeekProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)

```

*Arguments:*

`api_key` DeepSeek API key. Defaults to DEEPSEEK\_API\_KEY env var.

`base_url` Base URL. Defaults to https://api.deepseek.com.

`headers` Optional additional headers.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds for API calls.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
DeepSeekProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "deepseek-chat", "deepseek-reasoner", or a deepseek-v4\* model).

*Returns:* A DeepSeekLanguageModel object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
DeepSeekProvider$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

---

MoonshotProvider

*Moonshot / Kimi Provider Class*


---

**Description**

Provider class for Moonshot AI Kimi models.

**Super class**

[aisdk::OpenAIProvider](#) -> MoonshotProvider

## Methods

### Public methods:

- [MoonshotProvider\\$new\(\)](#)
- [MoonshotProvider\\$language\\_model\(\)](#)
- [MoonshotProvider\\$clone\(\)](#)

**Method** `new()`: Initialize the Moonshot provider.

*Usage:*

```
MoonshotProvider$new(
  api_key = NULL,
  base_url = NULL,
  platform = c("auto", "platform", "coding"),
  headers = NULL,
  prompt_cache_key = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

`api_key` API key. Defaults to `MOONSHOT_API_KEY` for the Kimi Open Platform, or `KIMI_API_KEY` / `KIMI_CODE_API_KEY` for Kimi Code.

`base_url` Base URL.

`platform` API platform: "auto", "platform", or "coding".

`headers` Optional additional headers.

`prompt_cache_key` Default prompt cache key for Kimi Code requests.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout.

`connect_timeout_seconds` Optional connection-establishment timeout.

`idle_timeout_seconds` Optional stall timeout.

**Method** `language_model()`: Create a language model.

*Usage:*

```
MoonshotProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID.

*Returns:* A `MoonshotLanguageModel` object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
MoonshotProvider$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

---

NvidiaProvider	<i>NVIDIA Provider Class</i>
----------------	------------------------------

---

**Description**

Provider class for NVIDIA.

**Super class**

`aisdk::OpenAIProvider` -> NvidiaProvider

**Methods****Public methods:**

- `NvidiaProvider$new()`
- `NvidiaProvider$language_model()`
- `NvidiaProvider$clone()`

**Method** `new()`: Initialize the NVIDIA provider.

*Usage:*

```
NvidiaProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

`api_key` NVIDIA API key. Defaults to `NVIDIA_API_KEY` env var.

`base_url` Base URL. Defaults to `https://integrate.api.nvidia.com/v1`.

`headers` Optional additional headers.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds for API calls.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
NvidiaProvider$language_model(model_id = NULL)
```

*Arguments:*

model\_id The model ID (e.g., "z-ai/glm4.7").

*Returns:* A NvidiaLanguageModel object.

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*

```
NvidiaProvider$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

---

OpenRouterProvider      *OpenRouter Provider Class*

---

**Description**

Provider class for OpenRouter.

**Super class**

[aisdk::OpenAIProvider](#) -> OpenRouterProvider

**Methods****Public methods:**

- [OpenRouterProvider\\$new\(\)](#)
- [OpenRouterProvider\\$language\\_model\(\)](#)
- [OpenRouterProvider\\$image\\_model\(\)](#)
- [OpenRouterProvider\\$clone\(\)](#)

**Method** new(): Initialize the OpenRouter provider.

*Usage:*

```
OpenRouterProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

api\_key OpenRouter API key. Defaults to OPENROUTER\_API\_KEY env var.

base\_url Base URL. Defaults to https://openrouter.ai/api/v1.

headers Optional additional headers.

timeout\_seconds Legacy alias for total\_timeout\_seconds.

total\_timeout\_seconds Optional total request timeout in seconds for API calls.

first\_byte\_timeout\_seconds Optional time-to-first-byte timeout in seconds for API calls.

connect\_timeout\_seconds Optional connection-establishment timeout in seconds for API calls.

idle\_timeout\_seconds Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
OpenRouterProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "openai/gpt-4o", "anthropic/claude-sonnet-4-20250514", "deepseek/deepseek-r1", "google/gemini-2.5-pro").

*Returns:* An `OpenRouterLanguageModel` object.

**Method** `image_model()`: Create an image model.

*Usage:*

```
OpenRouterProvider$image_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "openai/gpt-image-2").

*Returns:* An `OpenAIImageModel` object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
OpenRouterProvider$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

---

StepfunProvider

*Stepfun Provider Class*

---

## Description

Provider class for Stepfun.

## Super class

`aisdk::OpenAIProvider` -> StepfunProvider

**Methods****Public methods:**

- [StepfunProvider\\$new\(\)](#)
- [StepfunProvider\\$language\\_model\(\)](#)
- [StepfunProvider\\$image\\_model\(\)](#)
- [StepfunProvider\\$clone\(\)](#)

**Method** `new()`: Initialize the Stepfun provider.

*Usage:*

```
StepfunProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

`api_key` Stepfun API key. Defaults to STEPFUN\_API\_KEY env var.

`base_url` Base URL. Defaults to <https://api.stepfun.com/v1>.

`headers` Optional additional headers.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds for API calls.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
StepfunProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "step-3.5-flash").

*Returns:* A `StepfunLanguageModel` object.

**Method** `image_model()`: Create an image model.

*Usage:*

```
StepfunProvider$image_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "step-1x-medium", "step-1x-edit").

*Returns:* A `StepfunImageModel` object.

**Method** clone(): The objects of this class are cloneable with this method.

*Usage:*

```
StepfunProvider$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

---

VolcengineProvider      *Volcengine Provider Class*

---

## Description

Provider class for the Volcengine Ark platform.

## Super class

`aisdk::OpenAIProvider` -> VolcengineProvider

## Methods

### Public methods:

- `VolcengineProvider$new()`
- `VolcengineProvider$language_model()`
- `VolcengineProvider$image_model()`
- `VolcengineProvider$clone()`

**Method** new(): Initialize the Volcengine provider.

*Usage:*

```
VolcengineProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

api\_key Volcengine API key. Defaults to ARK\_API\_KEY env var.

base\_url Base URL. Defaults to https://ark.cn-beijing.volces.com/api/v3.

headers Optional additional headers.

timeout\_seconds Legacy alias for total\_timeout\_seconds.

total\_timeout\_seconds Optional total request timeout in seconds for API calls.

first\_byte\_timeout\_seconds Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method** `language_model()`: Create a language model.

*Usage:*

```
VolcengineProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "doubao-1-5-pro-256k-250115" or "gpt-4o").

*Returns:* A `VolcengineLanguageModel` object.

**Method** `image_model()`: Create an image model.

*Usage:*

```
VolcengineProvider$image_model(  
  model_id = Sys.getenv("ARK_IMAGE_MODEL", "doubao-seedream-5-0")  
)
```

*Arguments:*

`model_id` The model ID (e.g., "doubao-seedream-5-0").

*Returns:* A `VolcengineImageModel` object.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
VolcengineProvider$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

---

XAIProvider

*xAI Provider Class*

---

## Description

Provider class for xAI.

## Super class

`aisdk::OpenAIProvider` -> XAIProvider

## Methods

### Public methods:

- [XAIProvider\\$new\(\)](#)
- [XAIProvider\\$language\\_model\(\)](#)
- [XAIProvider\\$image\\_model\(\)](#)
- [XAIProvider\\$clone\(\)](#)

**Method new():** Initialize the xAI provider.

*Usage:*

```
XAIProvider$new(
  api_key = NULL,
  base_url = NULL,
  headers = NULL,
  timeout_seconds = NULL,
  total_timeout_seconds = NULL,
  first_byte_timeout_seconds = NULL,
  connect_timeout_seconds = NULL,
  idle_timeout_seconds = NULL
)
```

*Arguments:*

`api_key` xAI API key. Defaults to XAI\_API\_KEY env var.

`base_url` Base URL. Defaults to <https://api.x.ai/v1>.

`headers` Optional additional headers.

`timeout_seconds` Legacy alias for `total_timeout_seconds`.

`total_timeout_seconds` Optional total request timeout in seconds for API calls.

`first_byte_timeout_seconds` Optional time-to-first-byte timeout in seconds for API calls.

`connect_timeout_seconds` Optional connection-establishment timeout in seconds for API calls.

`idle_timeout_seconds` Optional stall timeout in seconds for API calls.

**Method language\_model():** Create a language model.

*Usage:*

```
XAIProvider$language_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "grok-beta", "grok-2-1212").

*Returns:* A XAILanguageModel object.

**Method image\_model():** Create an image model.

*Usage:*

```
XAIProvider$image_model(model_id = NULL)
```

*Arguments:*

`model_id` The model ID (e.g., "grok-2-image").

*Returns:* A XAIImageModel object.

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

```
XAIProvider$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

# Index

AiHubMixProvider, 2  
aisdk::OpenAIProvider, 2, 4, 25, 26, 28–30,  
32, 33

BailianProvider, 4

create\_aihubmix, 5  
create\_aihubmix\_anthropic, 6  
create\_aihubmix\_gemini, 7  
create\_bailian, 8  
create\_deepseek, 10  
create\_deepseek\_anthropic, 12  
create\_kimi\_code, 13  
create\_kimi\_code\_anthropic, 14  
create\_moonshot, 15  
create\_nvidia, 17  
create\_openrouter, 18  
create\_stepfun, 20  
create\_volcengine, 22  
create\_xai, 24

DeepSeekProvider, 25

MoonshotProvider, 26

NvidiaProvider, 28

OpenRouterProvider, 29

StepfunProvider, 30

VolcengineProvider, 32

XAIProvider, 33