WHAT KIND OF AI IS THAT?

The Types of AI Models, What Each Kind Does, and How to Tell Them Apart

Know what you're talking to.



Douglas B.

What Kind of AI Is That?

Know what you're talking to.

The Types of AI Models, What Each Kind Does, and How to Tell Them Apart

Douglas J. Brownfield

First edition — November 2026

Copyright

What Kind of AI Is That?

Copyright © 2026 Douglas J. Brownfield

All rights reserved. No part of this book may be reproduced, stored, or transmitted in any form or by any means — electronic, mechanical, photocopying, recording, or otherwise — without the prior written permission of the author, except for brief quotations used in reviews.

This book was written with the assistance of AI tools, working from the author's own questions and editorial direction, and from a research record built for this book from primary sources — the makers' own pages, papers, model cards, and repositories.

The models named in this book are stated as of November

each is dated where it stands. The full statement of how this book was checked, and what was left out for want of a source, is The Snapshot Statement in the back matter.

1. The kinds change slowly; the examples do not, and

Free editions of this book — EPUB, PDF, read-online, and audio — are published by the author at Douglas J. Brownfield Books.

The art in this book is the author's own painting, photographed in progression. The plates are cropped from the photographs with their colors unaltered; the cover setting and part plates derived from them are described in About the Art.

ISBN — [site edition; no KDP edition at this time]

Contents

Introduction

Part One — The Kinds

Chapter 1 — Large Language Models: The Talkers

Chapter 2 — Reasoning Models: The Kind That Became a Setting

Chapter 3 — Image Models

Chapter 4 — Video Models

Chapter 5 — Audio Models

Chapter 6 — Multimodal Models: The Kind Becoming a Property

Chapter 7 — Small and On-Device Models: The Pocket Kind

Chapter 8 — Embedding and Retrieval Models: The Librarians

Chapter 9 — The Frontier Shelf: Kinds Still Forming

Part Two — How to Read a New Model

Chapter 10 — The Five Questions

Chapter 11 — Names and Numbers

Chapter 12 — Benchmarks Without Tears

Part Three — Choosing

Chapter 13 — Match the Kind to the Job

Chapter 14 — The Layer: Loops, Tools, and Models in Combination

Back Matter

The Snapshot Statement

Sources

The Map on One Page

About the Art

About the Author

Also by Douglas J. Brownfield

Introduction

New AI models arrive every week. Each arrives with a name, an announcement, and a set of claims, and the name is the least informative part of the package. A name is marketing. It tells you who is selling and how the seller would like the model to feel — fast, smart, advanced, friendly. It does not tell you what the model is, what it takes in, or what it gives back. A reader who follows the names is following the one part of the field guaranteed to change.

The turnover is not an impression. It is the record. In the course of a single year, three of the best-known model names in the world ceased to exist. DALL·E, OpenAI's image model, was retired in May 2026. Imagen, Google's image model, was shut down that August. Sora, OpenAI's video model, closed in 2026 — the application in April, the developer service in September — and was not replaced. The work did not stop with the names. More images are generated now than when DALL·E was the household word, under names few people had heard a year earlier. Video generation continues without its most famous name. What ended was the names.

This book sorts by kind instead. A kind is determined by a short set of questions that can be asked of any model, whatever its name: What does it take in, and what does it give back? Where does it run — in a company's data center, or on a device in your hand? And can you hold it: are its weights published, under a stated license, or can it only be visited through the maker's service? Asked of a new announcement, these questions place the model on the map in about a minute. The answers change

slowly. The types in this book — language models, image models, the retrieval models that work quietly underneath search — have been recognizable for years, and they will still be recognizable when the current lineups have been replaced.

One distinction cuts across every kind and is treated throughout as what it is — a property of a model, not a kind of its own. Some models are published as weights a user can download and run; others exist only inside the maker's service. That difference decides who controls the tool, what it costs to keep using it, and whether it can be taken away on a schedule the user does not set. It is also a distinction with fine print: published weights come under named licenses — MIT, Apache 2.0, non-commercial terms, vendor-specific terms — that differ in what they permit. Wherever this book describes a model as open, it names the license, because "open" by itself does not say enough.

The map is honest about its own limits, because kinds are steadier than names, not frozen. Two chapters in Part One record kinds in the act of changing. Reasoning — models that work a problem in steps before answering — stood as its own kind while separate reasoning models were sold. By late 2026 it had been folded into the flagship language models as a setting, a matter of how hard the model is told to think. Multimodal ability, one model handling text, images, and sound natively, is further along the same path: less a shelf of its own, more a property every flagship is expected to have. Both chapters are kept, and both changes are stated where they happened, because how a kind behaves over time is part of what a reader needs to know. At the other end of the map, the final chapter covers kinds still forming, and says plainly where the record is thin.

Every named model in this book is a dated example, and the date is stated up front: the roster in these chapters was verified as of November 2026. A book about a field moving at this speed has two choices — pretend to be timeless, or date itself. This one dates itself. The kinds are the durable part; the examples are the perishable part, and they are labelled as such, each chapter closing with the shelf as it stood in November 2026. Printings stand as their dated selves. The online edition is updated in place as lineups change, each update carrying its version line and date, so a reader can always tell which snapshot they are holding.

Sorting by kind also pays at the one moment the names are most expensive: the moment of choosing. A name can be new, well reviewed, and entirely the wrong type for the job — a strong language model asked to search a private archive, a strong image model asked to keep figures consistent across a report. Most disappointments with these tools are mismatches of kind, not failures of quality. A reader who can name the kind a job needs, and recognize the kind a tool actually is, will choose better than a reader who knows every current name by heart — and will still be choosing well after those names are gone.

The arrangement follows from that. Part One is the map: one chapter per kind, what it is, what it is for, and what currently stands on its shelf. Part Two is the durable skill — how to read any new model announcement, its name, and its published scores, and place it yourself. Part Three matches kinds to jobs, and closes with the one thing that is not a kind at all: the layer of loops, tools, and permissions built around models, which makes them act rather than merely answer, and which is covered separately so the word *model* is never stretched to cover it.

This book is for anyone who uses these tools, chooses them, pays for them, or is simply tired of pretending to keep up with the names. It assumes no technical background and no patience for jargon; terms are explained where they are used. The title is the question the whole book teaches its reader to answer. Ask it of the next announcement — what kind of AI is that? — and the announcement becomes a good deal easier to read.

Part One — The Kinds



Part One plate — first washes. Art by Douglas John Brownfield; plate setting by the assistant.

Chapter 1 — Large Language Models: The Talkers

A large language model is a program that predicts text. Given a stretch of words, it produces the words most likely to come next, then the next, then the next, until the answer is complete. That is the whole mechanism. Everything these models do — answering questions, drafting letters, summarizing reports, writing code, holding a conversation — is that single operation, carried out at a scale where prediction becomes fluent.

The model learns to predict by training. During training it is shown an enormous body of text — books, articles, documentation, conversation, code — with parts hidden, and it is adjusted, over and over, until its guesses about the hidden parts are reliably good. What it stores from that process is not a library of sentences to look up. It stores patterns: grammar, facts, styles, the shapes of arguments and stories and programs, and the statistical relationships among all of them. When the model answers a question it was never specifically taught, it is assembling those patterns into new text. This is why a language model can be right about things in general and wrong about specifics in particular: it knows how answers of a given type usually go, which is a different thing from knowing.

Nearly every model in this chapter is built on one architecture, the transformer, introduced in a 2017 research paper. The transformer's central device is called attention: before producing each new piece of text, the model weighs which earlier pieces matter most to what comes next, and draws on those. Attention is what lets a model keep track of a subject across a long

paragraph, resolve a pronoun to the right name, or hold the thread of a contract clause from page one to page forty. The designs differ in detail from maker to maker, but a reader who understands that much — prediction, trained on text, attention holding the context — understands what kind of machine a large language model is.

"Large" refers to the model's parameters, the internal numbers set during training. Current flagships carry hundreds of billions of them, and the largest now pass a trillion. Many of the biggest models use a design called mixture of experts, in which only a fraction of the parameters are active for any given piece of text: Mistral's current flagship has one trillion parameters in total and uses 52 billion at a time, and DeepSeek's uses 49 billion of its 1.6 trillion. Total size describes what the model holds; active size is closer to what an answer costs.

What this type of model is for is, in practice, almost any work that arrives as words and leaves as words. It drafts, edits, summarizes, translates, explains, and answers. It writes and repairs computer code — code is treated in this book as the talkers at a trade, not a separate kind, because the models doing it are language models trained heavily on code. It is also the engine inside most products that appear to do something else: when a service answers in prose, a large language model is almost certainly doing the talking, whatever else is attached to it. Later chapters cover the models that work in other media, and Part Three covers the layer of tools and loops that lets a talker act. This chapter's kind is the foundation the rest of the map stands on.

A reader can recognize a large language model by its inputs and outputs. Text goes in; text comes out. It is reached through a chat window or through an API, the connection other software uses. It has no native grip on the world beyond its training and whatever a given product feeds it — no memory between separate conversations unless the product supplies one, and no ability to check a fact unless the product connects it to a source. Two properties that once defined separate shelves have been absorbed into this kind, and they get chapters of their own because the story matters: today's flagships reason, working a problem in steps before answering (Chapter 2), and the leading ones natively accept images alongside text, and in some cases audio and video as well (Chapter 6). But the base type — the thing being reasoned with and seen through — is the talker described here.

Families and tiers

No maker sells a single language model. Each sells a family: a set of models of different sizes and speeds, organized in tiers. The top tier is the most capable and the most expensive to run. The tiers below trade some capability for speed and price, down to small, fast models built for volume. A family shares a name and a generation number; the tier name says where a model sits inside it.

The tier names are each maker's own, and there is no standard across vendors. In November 2026, OpenAI's GPT-6 family runs, from the top, Astra, then Sol, Terra, and Luna. Under OpenAI's current naming scheme, the number marks the generation and the tier name marks the model's place in it, with each tier updated on its own schedule. Anthropic's ladder, from

the top, is Fable, Opus, Sonnet, and Haiku; Anthropic's own documentation names Opus the recommended default for most work, with the tier above it — Fable, new in 2026 — as its most capable widely released model. Alibaba's Qwen line marks its flagship tier with the word Max. The practical lesson is Part Two's subject, but it belongs to this chapter too: a tier name means something only inside its own family, and a model's full name — maker, family, generation, tier — is the smallest unit that actually identifies it.

Open weights and closed weights

One property cuts across every kind in this book, and it appears first here: whether a model's weights — the trained parameters themselves — can be downloaded. Closed models are available only through the maker's own products and API. Open-weight models can be downloaded and run on other hardware, under a license, and the license is part of the fact: this book names it, because "open" covers arrangements that differ a great deal.

The three largest American lines are closed. OpenAI's GPT models and Anthropic's Claude models are available only through those companies' APIs and products, and Google's Gemini flagships are likewise closed. The open shelf, as of this snapshot, is led from outside the United States. DeepSeek's V4 series is released under the MIT license, among the least restrictive licenses in use. Alibaba released the weights of its flagship Qwen3.8 model — a Max-class model, opened for the first time — under a custom Qwen license, while a smaller model in the same generation, Qwen3.8-27B, is under the Apache 2.0 license. Mistral, in France, published its flagship's weights under its own Mistral Ecosystem Open-Weight License,

a modified form of the MIT license under which running the model as a hosted service for others requires a separate commercial arrangement. One further caution belongs here: a downloaded model and the maker's hosted product of the same name are not always identical — Alibaba states that its hosted Qwen3.8-Max is its open model with additional features added.

The November 2026 shelf

OpenAI's current line is the GPT-6 family — Astra at the top, with GPT-6.1 Sol, GPT-6.1 Terra, and GPT-6 Luna below it — and the previous generation remains in service alongside it, with GPT-5.6 Sol still a documented flagship-tier model. The scheduled retirements tell the same story of turnover: the GPT-5 family is set to retire in December 2026, and the GPT-4o snapshots follow in January 2027.

Anthropic's shelf is the Claude 5.5 generation under a new top tier: Fable 5.1, released in September 2026, is the most capable model the company offers to the general market, with Opus 5.5 as its recommended default, then Sonnet 5.5 and Haiku 5.5. Anthropic also documents a gated counterpart, Mythos 5.1 — the same underlying model as Fable 5.1 with different safeguards — restricted to vetted organizations under a trusted-access program.

Google's flagship is Gemini 4 Argon, generally available since October 2026 across Google's API, cloud, search, and Gemini app. It natively accepts text, image, video, audio, and PDF input — the widest set among the flagships in this chapter. A second, stronger model, Gemini 4 Carbon, has been announced but is not generally available: as of this snapshot it is in pre-release testing with the United States government and trusted partners, a

government-first release pattern that is new for Google's main line.

Meta requires two sentences, because its position changed in 2026. Its open line is still Llama: Llama 4 Scout and Llama 4 Maverick, released in April 2025, remain the downloadable flagships, offered under Meta's own Llama Community License, which carries conditions — including a cap tied to monthly users — that standard open-source licenses do not. No Llama 5 has shipped. Meta's frontier, meanwhile, moved to a new family named Muse, developed by Meta Superintelligence Labs: Muse Spark, released in April 2026, is a natively multimodal model with closed weights, available through Meta's products rather than as a download. Open-weight models have since appeared under the Muse name as well, among them Muse Glimmer, a 30-billion-parameter model released under the Apache 2.0 license.

Mistral's flagship is Mistral Large 4: one trillion parameters, 52 billion active, natively multimodal, with its weights published in October 2026 under the license described above. DeepSeek's line is the V4 family — V4-Pro, generally available since August 2026, and the smaller V4.1-Flash — with open weights under the MIT license; DeepSeek's earlier, separate reasoning line was retired from its API in July 2026 and its capability folded into V4, which is Chapter 2's story. Alibaba's Qwen shipped four generations during 2026 and arrived at Qwen3.8, with the hosted Qwen3.8-Max at the top and, beside it, the first open release of a Max-class model's weights.

The pattern across the shelf is worth stating plainly, because it is the kind's current shape. The closed flagships are American; the

largest downloadable models now come from China and Europe. Every flagship reasons, and most take images as readily as text. And the shelf itself is perishable: several of the models named here will be replaced within the year. The kind — a trained predictor of text, working by attention over context — is what endures, and it is the kind the rest of this book builds on.

>

On the shelf, November 2026

- ***OpenAI (closed)*** *— GPT-6 family: Astra, 6.1 Sol, Terra, Luna. GPT-5.6 Sol still in service.*
- ***Anthropic (closed)*** *— Claude Fable 5.1, Opus 5.5, Sonnet 5.5, Haiku 5.5. Mythos 5.1 gated to vetted organizations.*
- ***Google (closed)*** *— Gemini 4 Argon, generally available. Gemini 4 Carbon announced, in gated pre-release testing.*
- ***Meta (split)*** *— Llama 4 Scout and Maverick, downloadable under the Llama Community License. Muse family at the frontier, mostly closed; Muse Glimmer open under Apache 2.0.*
- ***Mistral (open weights)*** *— Mistral Large 4, 1T parameters / 52B active, under the Mistral Ecosystem Open-Weight License.*
- ***DeepSeek (open weights)*** *— V4-Pro and V4.1-Flash, under the MIT license.*
- ***Alibaba Qwen (both)*** *— Qwen3.8-Max hosted; Qwen3.8-2.4T-A95B weights released under a custom Qwen license; Qwen3.8-27B under Apache*

2.0.

Chapter 2 — Reasoning Models: The Kind That Became a Setting

This chapter covers a kind that no longer exists as a kind. Two years ago, reasoning models were a separate shelf: distinct names, distinct prices, picked for distinct jobs. By November 2026, every major maker had folded reasoning into its flagship talkers, where it now lives as a mode — a setting, with a dial. The chapter stays in the book for two reasons. Reasoning was a real type, with a real difference in how the models worked. And its melting is the clearest case in this book of how kinds behave: a kind can be genuine, matter for a while, and still end as a feature of the kind next to it.

What they were

A plain large language model answers in one pass. It reads the question and writes the answer in a single forward run, the way a person blurts out a reply. That works well for most things. It works badly for problems that have to be carried in steps — a proof, a budget, a piece of code with moving parts — where the first sentence out is rarely the right place to start.

Reasoning models were trained to do the carrying first. Before answering, the model works through the problem in steps: it lays out sub-problems, tries an approach, checks it, backs up when a step fails, and only then writes the final answer. The working is usually hidden from the reader, or shown as a summary. What the reader gets is the answer at the end of it.

The trade behind this has a name worth knowing: **test-time compute**. An ordinary model spends nearly all of its computing

power in training, before it ever meets you; answering is the cheap part. A reasoning model spends computing power at answer time as well — the longer it is allowed to think, the more work it does on your particular problem. That is why these models were slower, why they cost more per answer, and why, on hard multi-step problems, they were often plainly better. You were not paying for a smarter memory. You were paying for time spent on the question.

Why they stood apart

In 2024 and 2025, reasoning was sold as a separate product line, not a setting. OpenAI kept its o-series — o1, o3, and their smaller variants — beside the GPT chat models, and pointed customers at one or the other depending on the job. DeepSeek shipped a pair: V3 as the general model and R1 as the reasoner, and R1's release in early 2025 was a genuine event — an open-weight reasoning model, MIT-licensed, that matched the closed labs' best. Anthropic offered reasoning as a named feature, Claude's "extended thinking." Google had Deep Think. A buyer in 2025 faced a real choice between two types of model, and the choice mattered: pick the talker for speed and price, the reasoner for problems with steps.

That is what made reasoning a kind in the first place. It was not a size or a brand tier. It was a different way of spending compute, trained into the model, and sold under its own names.

What happened

The capability did not disappear. The shelf did. The record, by maker:

OpenAI wound the o-series down. On **October 23, 2026**, o1, o1-pro, o3-mini, and o4-mini were shut down; o3 and o3-pro are scheduled to follow on December 11, 2026. In every case the listed replacement is a current GPT model with a reasoning setting. In the current GPT line, reasoning is a control — reasoning.effort, running from *none* through *low*, *medium*, and *high* to still higher settings on the top models.

DeepSeek retired R1 from its own API on **July 24, 2026**, days after its V4 line reached general release, and folded the capability into V4's thinking modes. V4-Pro offers three depths of thinking, on by default. R1 itself is not gone: its MIT-licensed weights remain downloadable, and people run it themselves. What ended was R1 as a separate hosted product. No R2 has been publicly released as of this snapshot.

Anthropic stopped treating thinking as a feature with a name. In the current Claude models, thinking is built in and on by default, with effort levels — the current defaults sit at *medium* on some models and *high* on others, and the user or the software can turn the dial up or down.

The pattern repeats at the smaller scale. Mistral's flagship offers reasoning as levels in the API, not as a separate model. Google describes its current flagship, Gemini 4 Argon, as "a frontier reasoning model," which is the melt stated in the maker's own words: the flagship is the reasoner now. Deep Think survives as a named mode, still presented by DeepMind on a Gemini 3.1 base.

What to take from the melt

First, learn the dial. Wherever you use a current flagship — OpenAI's, Anthropic's, DeepSeek's, Google's — there is an effort setting, whether or not the app in front of you shows it. It may be labeled *effort*, *thinking*, or a depth choice; it may be a fixed default chosen for you. The question "which model should I use for this?" has quietly become "how hard should this model think about this?" Those are different questions, and the second one is usually the one that decides the quality of the answer.

Second, match the thinking to the problem. High effort earns its cost on work with steps that depend on each other: mathematics, code that has to run, plans with constraints, arguments that have to hold together, and checking — asking a model to verify its own or another's work. It earns little on lookups, summaries, first drafts, and anything where a quick answer can be corrected in the next exchange. Paying for maximum thinking on a simple question buys delay, not quality.

Third, notice that the dial is already set. Because thinking is now on by default at the major makers, a reader can be paying for reasoning — in time and in tokens — without ever having chosen it. The defaults are sensible for general use, but they are choices made by the maker, not by you. For cheap, high-volume work, the low and *none* settings exist on purpose: a flagship told not to think is still a flagship, and it answers at talker speed. Knowing the dial exists is what turns a default into a decision.

Fourth, count the cost correctly. Thinking is billed and timed like output, because it is output — the model writes its working before it writes your answer, and you wait for both. A long think on a hard problem can take minutes and cost several times what the same model's quick answer costs. That is a fair trade when

the problem is hard. It is waste when it is not.

And last, the general lesson, which the rest of this book will use again: when you meet a new model name, ask whether it names a kind — a different way of working, trained in — or a mode, a tier, or a moment in a lineup. Reasoning spent two years as a kind and ended as a dial. The dial is worth more than the shelf ever was. But you have to know it is there.

>

>

> ***On the shelf, November 2026*** *There is no reasoning shelf. Reasoning lives inside the flagship talkers, as a mode with an effort dial:*
>
> - ***OpenAI*** *— the GPT line, with reasoning.effort from none to maximum settings; the o-series shut down October 23, 2026, with o3 scheduled to follow December 11, 2026.*
> - ***DeepSeek*** *— V4-Pro and V4.1-Flash, thinking on by default, three depths on the Pro model; open weights, MIT license. R1's weights remain downloadable; R1 left the hosted API July 24, 2026.*
> - ***Anthropic*** *— Claude models think by default, with effort levels; no separate reasoning product.*
> - ***Mistral*** *— Large 4, reasoning offered as API levels rather than a separate model.*
> - ***Google*** *— Gemini 4 Argon, billed by Google as a frontier reasoning model; Deep Think remains a*

named mode on a Gemini 3.1 base.

Chapter 3 — Image Models

An image model takes words in and gives a picture back. You describe what you want — a lighthouse at dawn, a product photograph of a red chair, a cartoon fox in a spacesuit — and the model produces an image of it. Most also work in the other direction on existing pictures: give one an image plus an instruction, and it returns an edited version, with an object removed, a background replaced, or a style changed. Of all the types in this book, this is the kind most people have already used, often without knowing its name.

What an image model is

Most image models of the last several years were built on a technique called diffusion. During training, the model is shown millions of images alongside text describing them, while those images are gradually buried in random noise, step by step, until nothing is left but static. The model learns to run that process in reverse. When you give it a prompt, it starts from a field of pure noise and removes the noise in stages, guided at every stage by your words, until a picture that matches the description emerges. Think of it as sculpting: the block is static, and the prompt tells the model what to carve away. Newer models vary the recipe — some generate the image in patches or in sequence, and some, like Google's current line, are built inside a general multimodal model rather than standing alone as a dedicated diffusion system. The techniques differ. The kind is defined by the exchange: words in, image out.

What it is for

The everyday uses are practical. Illustration for articles, books, and presentations. Product photographs and advertising drafts made without a camera or a studio. Concept art and storyboards — a quick look at an idea before anyone spends money on the real version. Design work where words have to appear inside the picture, such as posters, signs, and logos, which the better current models handle reliably after years of garbled lettering. And editing: extending a photograph beyond its edges, removing a stranger from a holiday picture, restyling an image you already own. This kind also does a great deal of quiet work inside other products, generating the pictures that talkers and apps hand back as part of a larger job.

One behavior follows from how this type works, and it surprises people the first time: the same prompt rarely gives the same picture twice. Because generation starts from random noise, each run carves a different image out of the static, and two people's identical prompts will differ as well. Skilled users treat this as part of the tool rather than a fault. They generate several versions, keep the one that works, and refine from there — or use the model's editing features to push a near-miss the rest of the way. A related habit separates the current models from their ancestors: keeping the same character, product, or style recognizable across a series of images, so a set of pictures can belong to one story or one brand. That consistency, more than raw picture quality, is what the recent generations have been competing on.

How to recognize one

The recognition test is the exchange. If the main thing going in is a text description and the main thing coming out is a picture,

you are looking at an image model, whatever the app around it is called. Editing features do not change the classification: an image plus an instruction in, an edited image out, is still this kind. The settings give it away too. Image models ask about size and aspect ratio, about style, and about how closely to follow the prompt — they do not carry on a conversation. If a service writes you three paragraphs about your picture instead of giving you the picture, a talker is doing the talking and an image model is working behind it. One more habit is worth knowing: the same name can cover several different models at once, sold in tiers. What matters is which tier you actually used, and when.

The November 2026 shelf

Two of the most famous names in this kind are gone, and the shelf is clearer for knowing it.

Midjourney remains the closed, hosted service it has been, and its current default model is V8.2, in place since July 24, 2026, with an updated editing model in testing behind it. Midjourney has never released weights; you use its models on its service or not at all.

OpenAI's image line is where the first famous name used to be. DALL·E was retired from OpenAI's API on May 12, 2026, and the name no longer exists as a product. Its replacement is the GPT Image family, and by November 2026 that line is already on GPT Image 2.5 — the first GPT Image version itself shut down in October 2026. “DALL·E” survives as a household word for the whole kind. It is no longer a model you can use.

Google's shelf turned over the same way. Imagen, Google's dedicated image line, shut down — Google's own migration

documentation set the final shutdown on the Gemini API for August 17, 2026. The successor is a Gemini-native family nicknamed “Nano Banana”: Nano Banana 2 and Nano Banana Pro, image models built inside the Gemini line rather than standing apart from it. Google's own naming churned during 2026, and the book reports the nickname because that is what the models are called in Google's documentation and in use. The pattern to notice: at both OpenAI and Google, the separate image brand was absorbed into the maker's main model family.

Ideogram split in two during 2026, and the split is the useful lesson. Ideogram 4.0, released in June 2026, is the company's first open-weight model — downloadable, and known for design work and accurate text inside images. Ideogram 4.5, released in September 2026, is the newer flagship, and it is closed: hosted only, including its precise-editing features. Same company, same year, two different arrangements. Which Ideogram a person means depends on which side of the split they are standing on.

FLUX, from Black Forest Labs, changed shape. In 2025, FLUX was the name most people meant by an open-weights image model. By November 2026, FLUX is a broader family — its newest generation is reported as one foundation spanning image, video, and audio — and its newest image model is closed. Open weights survive in the family's Klein tier and in FLUX.2 [dev], the established downloadable model. The name stayed; the arrangement underneath it moved.

Stable Diffusion, from Stability AI, is the shelf's quiet entry. Its current family is still SD 3.5, released in October 2024 — no new base image model has followed in over two years, and the

company has directed its recent work toward audio. The weights remain available and widely used. The famous open image model of 2022 to 2024 is alive; it is no longer where the frontier of this kind moves.

Open and closed, with the licenses named

This kind shows why this book names licenses instead of just saying “open.” Closed here means hosted only: Midjourney, GPT Image, Google's Nano Banana models, Ideogram 4.5, and FLUX's newest image model are all used on the maker's service, under the maker's terms, and can be changed or retired on the maker's schedule — as DALL·E and Imagen were. Open weights means the model file can be downloaded and run on your own hardware, but the license decides what you may do with it, and in this kind the licenses differ in ways that matter. Ideogram 4.0's weights are under the Ideogram Non-Commercial Model Agreement: free to download and use non-commercially, with a paid license required for commercial self-hosting. (The Apache 2.0 license attached to Ideogram's release covers its inference code, not the weights — the two are often confused.) FLUX.2 [dev] is under the FLUX Non-Commercial License. Within the FLUX Klein tier, terms differ by size: the FLUX.2 Klein 4B model is Apache 2.0, while the 9B is non-commercial. Stable Diffusion's SD 3.5 weights are under the Stability Community License. Two models can both be called “open” and permit entirely different things. Read the license, not the label.

>

On the shelf, November 2026

- ***Midjourney V8.2*** *— closed, hosted; the current default since July 2026.*
- ***OpenAI GPT Image 2.5*** *— closed; DALL·E retired May 12, 2026, and the GPT Image line replaced it.*
- ***Google "Nano Banana" (Nano Banana 2, Nano Banana Pro)*** *— closed, Gemini-native; Imagen shut down August 17, 2026.*
- ***Ideogram 4.0*** *— open weights under the Ideogram Non-Commercial Model Agreement.* ***Ideogram 4.5*** *— closed flagship.*
- ***FLUX*** *— newest image model closed; open weights in the Klein tier (terms vary by size) and FLUX.2 [dev] under the FLUX Non-Commercial License.*
- ***Stable Diffusion SD 3.5*** *— open weights under the Stability Community License; still the latest base model, unchanged since October 2024.*

Chapter 4 — Video Models

A video model generates moving pictures. You describe a shot in words — or hand it a still image and ask for motion — and it returns a few seconds of footage: a camera pushing in, a person crossing a room, rain on a window. The kind exists because a single image is often not the thing people actually want. They want the moment, not the frame.

What separates this type from the image models of the previous chapter is time. An image model has to get one frame right. A video model has to get every frame right and keep them consistent with each other — the same face, the same coat, the same light from the first frame to the last, while things move. That consistency problem is the technical heart of the kind. It is why video models arrived later than the other kinds, why they cost more to run, and why they still work in short clips rather than long scenes. Most current models produce footage measured in seconds. A finished minute of video is normally many clips, generated one at a time and edited together.

What it is for. Advertising shots and product clips. Storyboards and previsualization for film and television. Social-media video. Background and stock-style footage. A quick prototype of a scene that would be expensive to shoot. And, increasingly, editing: taking footage that already exists and changing it — removing an object, restyling a shot, extending what the camera captured. Generation and editing are the two halves of the kind, and some vendors sell them as separate models.

How you recognize one. Four signs:

- The output moves. If a tool returns a clip with a duration, it is a video model, whatever the marketing calls it.
- The unit of work is the shot, not the scene. Limits and prices are stated in seconds, resolutions, and frames per second.
- It usually accepts more than text: a still image to animate, reference images to hold a character or product steady, keyframes, or existing video to modify.
- Sound is a dividing line inside the kind. Some models generate audio together with the picture; others are silent and leave sound to the audio models covered in the next chapter. As of this snapshot, neither arrangement is standard. Check before you assume.

The limits you will meet. Because consistency across frames is the hard part, the failures of this type are consistency failures. Hands and faces drift. Text rendered inside a scene garbles as the camera moves. Objects appear, vanish, or change shape between frames; physical motion — pouring water, a bouncing ball — can look plausible at a glance and wrong on a second look. Length is the other limit: a model asked for one continuous minute will usually do worse than the same model asked for six ten-second shots. Working practice in this kind follows from those facts. Generate short. Generate more than you need and keep the best takes. Give the model reference images when a person or product has to stay recognizable, and expect a human edit at the end. None of this is a defect of one brand; it is the current shape of the type, and it is the honest baseline against which each new release should be measured.

The shelf, November 2026. Five working names and one empty chair. With one partial exception, everything on this shelf is

closed: rented from the maker through an app or an API, not downloaded and run on your own machine.

Runway is the specialist that stayed a specialist. Its current flagship is Gen-4.5, released in December 2025 — text- and image-to-video, picture only, with no native audio. Its editing half is a separate model, Aleph 2.0. One company, one kind, two models: the plainest illustration on the shelf of the generation/editing split described above.

Kling, from Kuaishou, is the fastest-moving name here — two full generations within a year. Kling 3.0 shipped in early 2026; Kling 4.0 became fully available in October 2026, with clips of up to thirty seconds and sound included. Where Runway separates its functions, Kling keeps adding them.

Luma fields the Ray family; its current flagship is Ray3.2, released in 2026, silent like Runway's models. Luma matters to this chapter for a second reason. In 2026 it rebuilt its product, Dream Machine, so that it now hosts other companies' models — Google's and Kuaishou's among them — alongside its own Ray line. One of the makers on this shelf has also become a storefront for its rivals. That is a distribution change, not a new kind, but a reader comparing tools will meet it and should know what it is.

Google keeps two names in this kind. Veo 3.1 is the current Veo. But Google now steers its users to a different model — Gemini Omni Flash — as the default video model in its API, and Omni has replaced Veo inside the Gemini app. No Veo 4 has been announced. The pattern will be familiar from the image chapter: a standalone media brand giving way to a model built natively into the maker's main line.

Pika covers both halves of the open-and-closed question in one company. Its own current model is Pika 2.5, closed like the rest of this shelf. Its notable 2026 release, though, is Pika H3 Flex: Pika took MiniMax's open H3 model, post-trained it for speed, and in October 2026 released the result as open weights under the MiniMax H3 Community License. That license is why this book names licenses instead of just saying "open": it excludes commercial use in the United States, the European Union, the United Kingdom, and South Korea. H3 Flex is the one entry on this shelf you can download — and its terms are a worked example of reading what "open" actually permits before you build on it. (The FLUX family, covered with the image models in Chapter 3, has likewise grown a video member; this kind is where it will be met in practice.)

Sora is the empty chair. Sora was OpenAI's video model and, through 2024 and 2025, the most famous name in this kind. It no longer exists. OpenAI announced its discontinuation in March 2026. The Sora app and website shut down on April 26, 2026. The Sora models and the Videos API were removed on September 24, 2026, with no replacement. As of November 2026, OpenAI offers no video generation product at all.

What the Sora exit teaches. Almost every model in this chapter is a rental. A rented model comes with the maker's roadmap and the maker's freedom to end the product, and Sora's users are the standing example: people who built workflows, habits, and in some cases businesses on the most prominent video model available had a few months' notice to move them or lose them. Nothing about the kind failed. Video generation is more capable and more widely available in November 2026 than it was when Sora launched. What ended was one company's product, on that

company's schedule.

The practical conclusions are plain. Keep your own copies of the things that are yours — the prompts, the reference images, the finished clips — rather than leaving them inside one vendor's app. Learn the kind, not just the tool: the skill of describing a shot transfers to every model on this shelf, while familiarity with one interface transfers to none of them. And treat moving your work between vendors as a normal, recurring cost of using this type of model, not as an emergency. A name is a lease. A kind is a category. This book sorts by the category.

>

On the shelf, November 2026

- ***Runway Gen-4.5*** *— generation;* ***Aleph 2.0*** *— editing. Closed.*
- ***Kling 4.0*** *(Kuaishou) — the shelf's longest clips, with sound. Closed.*
- ***Luma Ray3.2*** *— also sold from a storefront that hosts its rivals. Closed.*
- ***Google Veo 3.1*** *— current Veo;* ***Gemini Omni Flash*** *is now the default. Closed.*
- ***Pika 2.5*** *— the in-house line. Closed.* ***Pika H3 Flex*** *— open weights under the MiniMax H3 Community License, territory-restricted.*
- ***Sora*** *(OpenAI) — discontinued in 2026; no replacement. The chair is empty.*

Chapter 5 — Audio Models

Every other kind in this part of the book works in text or pictures. Audio models work in sound, and sound runs in more than one direction. That is the first thing to understand about this kind: it is really three types sharing one shelf.

The first type listens. Sound goes in, text comes out — a recording of a meeting becomes a transcript. The second type speaks. Text goes in, a voice comes out — a written page becomes narration. The third type makes music. A description goes in, and a finished song comes out, instruments and vocals included.

These are different jobs, done by different models, sold by different companies. A model that transcribes well does not speak well, and neither one writes songs. When someone says "audio AI," the useful question is: which direction?

Ears in: transcription

A transcription model takes recorded speech and returns written words. It is the least glamorous of the three types and probably the most used. Interviews, lectures, meetings, phone calls, podcasts — anywhere people talk and someone later needs the text, a model of this type is doing the work.

The work these models do is narrower than it sounds. A transcription model does not understand a conversation the way a language model does; it matches sound to words, adds punctuation and capital letters, and, in the better versions, marks where one speaker stops and another begins. Judged on accuracy, the differences between good ones are small and

mostly show up in hard cases: names, accents, technical vocabulary, and several people talking at once in a noisy room.

The standard example is Whisper, from OpenAI. Whisper was released as open weights, and the open release is the interesting part of its story. The downloadable models — large-v3 and its faster sibling, large-v3-turbo — are unchanged and remain free for anyone to run under the MIT license, one of the most permissive licenses there is. If you downloaded Whisper two years ago, the copy on your machine today is still current, still legal to use, and still entirely yours.

OpenAI's hosted service has moved on. In August 2026 the company deprecated the API models built on Whisper — whisper-1 and the transcribe models in the GPT-4o line — and set their shutdown for February 2027. The default transcription model in OpenAI's API is now called gpt-transcribe, a closed model from the GPT family. It is not a new version of Whisper. It is a different kind of model doing the same job, rented by the minute instead of owned.

Both facts belong in the book because together they teach something general. The name "Whisper" now refers to two different things: a finished, open artifact that anyone can keep, and a retired endpoint in a company's catalogue. The hosted service had a lifespan. The open weights do not — they last exactly as long as people care to run them. This pattern, an open release outliving the product it came from, appears in several kinds in this book. Audio is where it is easiest to see.

Voices out: speech synthesis

A speech model does the reverse of transcription. Text goes in; a spoken performance comes out. The modern versions do more than read words aloud in order. They handle pacing, emphasis, and emotion, and they can hold a consistent voice across a book-length narration or switch between speakers in a dialogue.

The leading name on this shelf is ElevenLabs. Its flagship model is Eleven v3, which became generally available in early 2026. Eleven v3 works in more than seventy languages, accepts directions inside the text itself — markers that tell the voice to whisper, to laugh, to pause — and handles conversations with several distinct speakers. It is built for quality rather than speed, and it is not the model ElevenLabs offers for live conversation.

That job belongs to a second model, Flash v2.5, which answers in roughly seventy-five milliseconds — fast enough that a person talking to it does not feel the delay. A third, Multilingual v2, remains the workhorse for long-form narration, the audiobook end of the trade. This division is typical of the kind: one model tuned to sound its best, another tuned to answer at once, a third tuned to read steadily for hours, and the buyer picks according to the work. ElevenLabs' older Turbo models, which once filled the fast role, are deprecated, with Flash named as their replacement.

ElevenLabs is a closed model. There are no weights to download; the voice exists only inside the company's service. It is also worth noticing, as a sign of where the kind is heading, that the same company now sells a transcription model and a music model alongside its voices. The three types on this shelf are separate kinds of model, but nothing stops one company from fielding all three.

Music

A music model takes a description — a genre, a mood, sometimes lyrics — and returns a complete recording: melody, arrangement, instruments, and usually a sung vocal. Of the three types in this chapter, it is the newest, and it is the one whose recent history has been decided as much in legal offices as in laboratories.

The leading name is Suno. The line moved quickly — v5 arrived in September 2025, and v5.5 followed in March 2026 with voice and custom-model features. Its current model is v6, released on September 9, 2026, together with two smaller variants, v6-wild and v6-mini. On the day v6 launched, Suno retired every older model it had — versions 4 through 5.5 stopped being available at once, although songs already made with them remain in their owners' libraries. That clean sweep is unusual, and it has a reason: v6 is Suno's first generation built together with partners from the music industry, including Warner Music Group, after Suno settled the lawsuits the record companies had brought against it. The model on the shelf in November 2026 is, in part, a product of those settlements.

The second name on the shelf shows the other way the same pressure can end. Udio was, for a time, Suno's closest rival. Its current model is still v1.5 Allegro, released in March 2025. Nothing newer has shipped. Udio also settled with the labels — Universal Music Group in October 2025, Warner a month later — and on the day the Universal settlement was announced, downloads were switched off across the whole platform: no audio files, no video, no stems. They remain off. The terms now limit what users make to personal, non-commercial use, and the

fully licensed platform Udio promised for 2026 had not launched as of this snapshot, with no date published.

So the music shelf in November 2026 holds one model that was rebuilt with the industry's participation and one that is waiting, intact but unable to hand its users their own songs. Neither outcome was decided by the quality of the models. It is a fair lesson in reading any new model: the technology is only part of what determines whether a kind thrives. Rights, licenses, and settlements are part of the shelf too.

Reading a new audio model, then, starts with direction. Ears in, voices out, or music — the three types are not interchangeable, and a launch announcement that does not say which one it means has not told you much. After direction comes the question this book asks of every kind: open or closed, and under what license? Whisper's weights can be downloaded and kept; ElevenLabs' voices and Suno's songs exist only inside their services, on whatever terms those services set — and the music shelf shows what can happen to the work when the terms change.

>

>

On the shelf, November 2026

- ***Ears in (transcription):*** *Whisper — open weights, unchanged, MIT license; a finished artifact anyone can run. OpenAI's hosted Whisper endpoints are deprecated, replaced in the API by the closed gpt-transcribe.*

- ***Voices out (speech synthesis):*** *ElevenLabs — Eleven v3 flagship for quality, Flash v2.5 for real-time conversation; the older Turbo models deprecated. Closed; service only.*
- ***Music:*** *Suno v6, released September 9, 2026, with all earlier versions retired the same day; built with music-industry partners after its label settlements. Udio remains on v1.5 (March 2025), downloads disabled since its October 2025 settlement, its licensed relaunch not yet shipped.*

Three directions, three types — and one name, Whisper, that now means both a model you can own and a service that has already ended.

Chapter 6 — Multimodal Models: The Kind Becoming a Property

A multimodal model is one model that works in more than one medium natively. It can take text, images, audio, or video as input — not through attachments or plug-ins, but as part of what the model itself was built and trained to handle. The word that does the work in that sentence is *natively*.

The alternative is assembly. For years, a system that seemed to see and hear was usually several single-medium models wired together: a speech recognizer turned talk into a transcript, a language model read the transcript and wrote a reply, and a voice model read the reply aloud. Systems like that work, and many are still in use. But each handoff is a translation, and translations lose things — tone of voice, emphasis, the pause before an answer. The language model in the middle never heard the question; it read a report of the question. A native multimodal model has no middleman. The same model that reasons about the words also processes the picture or the recording, so it can use what is in each medium when it thinks about the others.

That distinction is why multimodal models get a chapter of their own in this book — and the chapter comes with a warning label, because this kind is in the middle of disappearing into the others.

The omni era

For most of the field's short history, models were single-medium by default. Language models handled text. Image models handled pictures. A model that natively combined media was the

exception, and it looked like a new type of thing.

The model that made the point was OpenAI's GPT-4o, released in 2024. The "o" stood for "omni." One model took text, image, and audio input and could answer in text or in speech, holding the media together in a single system rather than passing them between specialists. Other makers fielded their own native multimodal models, and for a time "multimodal" — or "omni" — named a shelf of its own: a kind you chose on purpose, distinct from the plain talkers.

The state of it, November 2026

The shelf has emptied, and it emptied upward: the flagships absorbed the kind.

The clearest evidence is the fate of the model that defined it. GPT-4o is being retired. One of its dated snapshots was shut down on October 23, 2026, and the remaining snapshots are scheduled to retire on January 14, 2027, with GPT-6 Luna named as the replacement. A maker does not retire the model that founded a category unless the category has been folded into the main line — and it has been. The current GPT models take text and image input as a matter of course. There is no separate omni line to choose; the property comes with the flagship.

Google's Gemini 4 Argon, the current Gemini flagship, shows how far the absorption has gone. It natively accepts five types of input — text, image, video, audio, and PDF documents — in a single model, the deepest native set among the flagships surveyed for this book. Anthropic's current Claude models all accept text and image input. Mistral's flagship, Mistral Large 4, is described by its maker as natively multimodal, and its weights

are published under the Mistral Ecosystem Open-Weight License. Meta's Muse Spark is a natively multimodal reasoning model.

The property has also spread downward. DeepSeek's V4.1-Flash is its speed tier, not its flagship, and it gained native visual understanding in 2026. Alibaba's Qwen3.8-27B is a mid-size open model — its weights are published under the Apache 2.0 license — and it takes image and video input. When a capability reaches the cheaper and smaller members of a family, it has stopped being a specialty. It is part of what a model is now expected to do.

Stated plainly: in November 2026, a frontier model that handles only text is the exception. Multimodality at the top of the market is no longer a kind of model. It is a property the leading models have.

What still differs

A property can be standard and still vary. Three differences still separate one multimodal model from another, and they are the differences worth checking.

The first is *which senses*. The input lists are not the same. Gemini 4 Argon takes video and audio natively; the current Claude line takes text and image, not sound or motion. "Multimodal" on two labels can mean two different sets of media.

The second is *input against output*. Most of today's multimodal flagships take more in than they give out. They accept images, and in some cases audio and video, but they answer in text. The founding model was broader on this point: GPT-4o could

answer in speech as well as text. Its successors, as a group, take more media in than it did and send less out. Producing a picture, a voice, or a moving image is still mostly the work of the specialist types covered in the image, video, and audio chapters — or of a sibling model offered alongside the flagship in the same product. A model that can look at a photograph cannot necessarily draw one. When a label says multimodal, it is almost always describing the model's eyes and ears, not its hands.

The third is *depth*. Native handling is a spectrum. At the deep end, one model holds the media in a shared internal form, so a question about a diagram can draw on the diagram and the caption and the surrounding text at once. At the shallow end, media are converted to text summaries early and the model reasons over the summaries — closer to the old assembly line, wearing a native label. The deep end shows in work like computer use: OpenAI documents its GPT-6.1 Terra tier as handling visual reasoning and computer use — reading a screen as an image and acting on what it sees — which is only possible when the seeing is inside the model doing the reasoning. The label alone does not say where on that spectrum a given model sits; the maker's documentation, and testing on material like your own, does.

The lesson of a melting kind

This is the second chapter in this book about a kind that stopped standing on its own — reasoning did it first, becoming a mode of the talkers. Multimodal is doing it more slowly and more completely: not a mode, but a property, spreading until it belongs to nearly every model that matters.

The practical lesson is about labels. When a kind melts into a property, the word stops sorting. It still tells the truth — the model really does handle several media — but it no longer tells you which shelf a model belongs on, because the answer is all of them. The questions that sort are the narrower ones: Which media go in? Which media come out? How deeply are they joined? A reader who asks those three questions of any new model will know more about it than a reader who stops at the word "multimodal."

That is where this book leaves the word: accurate, and no longer load-bearing.

On the shelf, November 2026

- **Google Gemini 4 Argon** — closed weights. Takes text, image, video, audio, and PDF natively; answers in text. The deepest native input set among current flagships.
- **OpenAI GPT-6 family** — closed weights. Text and image input, text output. The GPT-4o line that defined the omni type is retiring; its last snapshots are scheduled to go on January 14, 2027.
- **Anthropic Claude (Fable 5.1, Opus 5.5, Sonnet 5.5, Haiku 5.5)** — closed weights. Text and image input, text output.
- **Mistral Large 4** — weights published under the Mistral Ecosystem Open-Weight License. Natively multimodal flagship.
- **Meta Muse Spark** — closed weights. A natively multimodal reasoning model; Meta's current frontier line.

- **Alibaba Qwen3.8-27B** — open weights under the Apache 2.0 license. Image and video input.
- **DeepSeek V4.1-Flash** — open weights under the MIT license. Native visual understanding added to the Flash line in 2026.

Chapter 7 — Small and On-Device Models: The Pocket Kind

Every other kind in this book is sorted by what a model does — talk, reason, draw, listen. This kind is sorted by where a model lives. A small model is one that runs on the device in your hand or on your desk: on the phone itself, not in a data center the request has to travel to. That single fact — where the work happens — produces a cluster of properties that no other kind can match, and it is why small is a kind of its own and not merely a size.

Start with what the location buys. **Privacy:** if the model runs on the device, the words, photos, and recordings it works on never have to leave the device. **Offline operation:** a model on the phone works in a tunnel, on a plane, in a dead zone — anywhere the phone works, because there is no server to reach. **Speed of response:** there is no round trip across a network, so answers start immediately. **Cost:** there is no per-call charge, because there is no call. The company that shipped the model does not meter your use of it, and using it a thousand times costs nothing beyond the battery.

Those four properties are the kind. A model earns its place on this shelf not by being under some parameter count but by being built to live under those conditions: limited memory, limited power, no connection assumed. A shrunken copy of a data-center model that still phones home for every answer is not this kind. A model designed from the start to work alone, in the pocket, is.

How a small model stays capable

The obvious question is what gets lost. A model with a few billion parameters cannot hold what a model with a trillion holds, and the honest answer is that small models know less and reason less deeply than the flagships of the earlier chapters. The work of the last several years has gone into narrowing that gap by three main routes.

The first route is better training material. Microsoft's Phi line is the standing example: its models are trained on carefully curated, textbook-quality data, on the theory that a small model fed well-organized material learns more per parameter than one fed the raw internet. The second route is distillation — a large model, the teacher, generates answers and worked examples, and a small model, the student, is trained on the teacher's output rather than on raw text. The student does not learn everything the teacher knows; it learns the teacher's way of answering, in a fraction of the space. The third route is sparsity, and it has produced the most striking design on the current shelf.

A sparse model does not use all of itself for every request. It carries a large total of parameters but activates only a small subset for any given question — the parts relevant to that question. Apple's current on-device line is the worked example. Its larger phone model, AFM 3 Core Advanced, carries 20 billion parameters in total but wakes only 1 to 4 billion of them per request. The memory the phone must hold is the full 20 billion; the computation the phone must perform is the small fraction. That is how a model of that total size can live on a phone at all, and it is why "how big is it" has become an ambiguous question for this type: the stored size and the

working size are now two different numbers.

On the shelf, November 2026

Five lines make up the current shelf, and they do not move at the same pace.

Microsoft Phi is the research line that defined the kind: small language models built on data quality. The current generation is still Phi-4 — no Phi-5 has been announced. The family includes the 14B Phi-4, the 3.8B Phi-4-mini, multimodal and reasoning members, and its newest general member, Phi-4-reasoning-vision-15B (March 2026), a 15B model that sees images and decides for itself when to think step by step and when to answer directly. The weights are published under the MIT license.

Google Gemma is Google's downloadable family, built from the same research as the closed Gemini line. Gemma 4 arrived in April 2026: Effective 2B and Effective 4B sizes built for phones and edge hardware — natively multimodal, and the two small sizes take audio input as well — alongside larger 26B and 31B members. The change worth recording is the license. Gemma 1 through 3 shipped under Google's custom Gemma Terms of Use; Gemma 4 ships under Apache 2.0, the standard license, with no custom terms layered on top.

Alibaba Qwen runs the fastest-moving open family, and its small shelf is a ladder: the Qwen3.5 small series — 0.8B, 2B, 4B, and 9B, released in March 2026 under Apache 2.0, natively multimodal like the rest of that generation. Qwen shipped two further generations during 2026, but their new sizes start at 27B, so the March small series remains the on-device line. (Qwen's

largest tiers — the hosted Max, Plus, and Flash services — are a different matter: those are API-only and belong with the talkers of Chapter 1.)

Apple Foundation Models are the kind in its purest commercial form: models that exist only inside a product, never downloaded, never sold separately. Apple's third generation of the family runs in two on-device tiers — AFM 3 Core, a 3B model, and the sparse AFM 3 Core Advanced described above. Where the work is too big for the device, Apple's Private Cloud Compute servers take the request; the on-device shelf is these two. One fact about their making belongs in the record: Apple states that this generation was custom-built in collaboration with Google. The weights are closed and unpublished; app developers reach the on-device model only through Apple's framework.

Meta Llama, at small sizes, is the line that stood still. Meta's small on-device offering remains the Llama 3.2 1B and 3B models of September 2024, published under the Llama Community License — a custom license, permissive for most uses but not a standard open-source license. No new small Llama appeared in 2025 or 2026, and Meta's newest frontier work has gone to a closed line instead. The 2024 models still work, and they are still the shelf entry; the record is simply that this seat has not been updated while the others moved.

What this kind is chosen for

Readers meet this type most often without choosing it: the model inside a phone's built-in assistant features, summarization, writing help, and photo tools is a small

on-device model, picked by the manufacturer for the four properties above. Where a person or a company does the choosing, the reasons are the same four. Choose the pocket kind when the material is private and should not travel; when the work must happen with no connection; when the answer is needed instantly, thousands of times a day, at no per-call cost; or when the job is narrow and constant — sorting, summarizing, transcribing, drafting — rather than open-ended. Choose a flagship from the earlier chapters when the question is hard, the knowledge required is wide, or the reasoning is long. The two kinds are not rivals. Most devices now run both: the small model on the spot, the large one a network away, and the craft is in knowing which of the two a given task is actually asking for.

On the shelf, November 2026

- ***Microsoft Phi-4 line*** *— including Phi-4-mini 3.8B and Phi-4-reasoning-vision-15B (Mar 2026). Open weights, MIT license.*
- ***Google Gemma 4*** *— Effective 2B / Effective 4B for phones; 26B and 31B above them (Apr 2026). Open weights, Apache 2.0.*
- ***Alibaba Qwen3.5 small series*** *— 0.8B / 2B / 4B / 9B (Mar 2026). Open weights, Apache 2.0.*
- ***Apple Foundation Models (3rd generation)*** *— AFM 3 Core 3B; AFM 3 Core Advanced, 20B sparse activating 1–4B. Closed; ships inside Apple products only.*
- ***Meta Llama 3.2 1B / 3B*** *— September 2024; still Meta's small line. Open weights, Llama Community License.*

Chapter 8 — Embedding and Retrieval Models: The Librarians

Every other kind in this book answers you. The talkers write, the image models draw, the audio models speak and listen. This type does none of that. An embedding model reads a piece of text — a sentence, a paragraph, a whole document — and hands back a list of numbers. That is all it produces, and it is enough to run most of the finding in modern software. If the talkers are the voices of AI, this is the index: quiet, invisible, and underneath nearly everything.

An embedding is a translation of meaning into position. The model converts a passage into a long list of numbers — hundreds or thousands of them — that together mark a point in an abstract space. The training behind it arranges that space so that passages with similar meanings land near each other and passages about different things land far apart. "How do I return a package?" and "Where is my parcel?" use almost no words in common, but they mean nearly the same thing, and a good model of this kind puts their number lists close together. "Return" in the sense of a tax return lands somewhere else entirely. Once meaning has an address, closeness becomes arithmetic: software can compare two lists in a blink and say how near they sit.

That one ability — measuring nearness of meaning — turns out to be the load-bearing piece in several systems a reader uses every day.

It is what makes search understand intent instead of matching keywords. A keyword search finds the documents that contain your words; a search built on embeddings finds the documents that mean what you meant, including the ones that never use your words at all.

It is what powers recommendations of the "more like this" kind: the article, product, or song nearest to the one you are looking at, measured in meaning rather than in tags somebody typed in.

And it is the first half of the arrangement called RAG — retrieval-augmented generation — which is how many talkers are made to answer from a specific body of material. Before the language model writes a word, a model of this type searches the library, ranks what it finds by nearness to the question, and fetches the few passages worth reading. The talker then answers from those pages instead of from memory alone. When a chatbot answers accurately about a company's own documents, the librarian did the finding and the talker did the talking. The two types are almost always found working as a pair, and only one of them gets the credit in conversation.

The same geometry does quieter work as well. A pile of documents can be grouped by topic simply by clustering the points that sit near each other. Near-duplicate pages reveal themselves as points that land almost on top of one another. Incoming items — support tickets, applications, reports — can be routed by which established category their position sits closest to. None of this requires the model to say anything; the arrangement of the points is the result.

Recognizing this kind is mostly a matter of noticing what it never does. It never talks back. You will not chat with an

embedding model, and no product puts one in front of you as a personality. Text goes in, numbers come out; the numbers are used to rank, sort, group, and retrieve, and then some other part of the system speaks. If a tool can pull the three most relevant pages out of a million in less than a second, a model of this type did that work, whatever the tool is called.

Judging these models is correspondingly plain. They are not rated on eloquence but on retrieval quality — whether the truly relevant items land at the top of the ranking — and the field keeps running comparisons for exactly that. The type also changes more slowly than the talkers. Its flagship names stayed put through 2026 while other categories turned over around them; the news in this kind was consolidation, not replacement.

The shelf, as of November 2026, looks like this.

OpenAI's pair, **text-embedding-3-small** and **text-embedding-3-large**, remains the default for much of the industry — closed models, reached through OpenAI's API. The small one represents each passage with 1,536 numbers, the large with 3,072, and both allow the list to be shortened when a coarser position is good enough. OpenAI shipped no new embedding model in 2026; the pair stays in place because it is the low-friction choice, even as rivals have taken the lead on quality in recent comparisons.

The clearest of those rivals is **Voyage AI**, which is no longer an independent company: MongoDB acquired it in February 2025, and its models now sit inside MongoDB's Atlas vector-search stack as "Voyage AI by MongoDB." The current family is the **Voyage 4 series**, active since January 15, 2026 by MongoDB's own lifecycle listing: voyage-4-large at the top, then voyage-4

and voyage-4-lite, all closed and API-served, all sharing a single embedding space so the model that positions a question and the model that positions documents can be mixed without the geometry breaking. Domain members followed — voyage-code-4 for code, voyage-context-4 for long material. The whole Voyage 3 series has been moved to Legacy status: still served, no longer the line being built on. One member of the new family breaks the closed pattern — **voyage-4-nano** is released as open weights under the Apache 2.0 license.

Google's entry, **Gemini Embedding 2**, also belongs on this shelf. It is a closed model, and by October 2026 published comparisons were placing it at or near the top of the retrieval rankings, alongside the Voyage 4 family.

The open flagship is still **BGE-M3**, from BAAI, released under the MIT license. It is a modest model by the standards of this book — 568 million parameters — that does three retrieval styles in one body: dense matching by overall meaning, sparse matching that still respects individual words, and multi-vector matching that compares passage piece by piece. It handles more than a hundred languages. Its November 2026 status is best stated plainly: alive, downloadable, widely deployed — and quiet. BAAI has shipped no new base embedding model in over a year, and the open-weights energy in this kind has lately sat with newer lines from other makers. BGE-M3 remains the standard open choice largely because, in its niche, nothing has replaced it.

That quiet is the honest summary of the whole kind. While famous generative names rose and died in a single year, the librarians kept their shelves: the same default pair at OpenAI,

one acquisition and one generational step at Voyage, a new Google entry at the top of the comparisons, and a 2024 open champion still holding its place. Readers looking for the type that will still be recognizable in the next edition of this book should start here.

On the shelf, November 2026

- **OpenAI text-embedding-3-small / text-embedding-3-large** — closed, API only. Unchanged in 2026; still the default pair.
- **Voyage 4 family** (voyage-4-large, voyage-4, voyage-4-lite; voyage-code-4, voyage-context-4) — closed, from Voyage AI by MongoDB. Voyage 3 series now Legacy.
- **Voyage-4-nano** — open weights, Apache 2.0 license. The small open member of the Voyage 4 family.
- **Google Gemini Embedding 2** — closed. Ranked at or near the top of October 2026 retrieval comparisons.
- **BGE-M3** (BAAI) — open weights, MIT license. The open flagship: dense, sparse, and multi-vector retrieval in one model, 100+ languages; no successor in over a year.

Chapter 9 — The Frontier Shelf: Kinds Still Forming

Every other chapter in this part of the book describes a kind that has settled. The models in those chapters can be bought, downloaded, compared, and argued about on the strength of a record. This chapter covers the kinds that have not settled yet. They are real — working models, from serious laboratories, that a reader can in some cases try today — but they are few, young, and still changing shape. They get a shelf of their own, and the shelf is labelled honestly: what follows is the state of two forming kinds as of November 2026, and parts of it will be out of date before any other chapter in this book.

Two types of model stand on the shelf. The first is the **world model**. A world model simulates an environment: not a picture of a place, and not a film of one, but a working model of how a place behaves, which holds together while someone acts inside it. The test that separates a world model from a video model is interaction. A video model produces a sequence of frames and the sequence is finished. A world model responds — turn left and the corridor is there, push an object and it moves as objects move — and it keeps the world consistent while the actions continue.

The second type is the **robot-action model**, usually called a VLA, for vision-language-action. Its inputs are what a robot has: camera images and an instruction in ordinary words. Its output is not text or pictures but physical action — motor commands that move a real machine. A talker answers a question; a VLA picks up the cup. These models are the current answer to an old

problem in robotics: instead of writing a program for every task a robot might be given, train one model on many robots doing many tasks, and let it generalise.

World models

The clearest statement of the kind is Google DeepMind's Genie line. **Genie 4**, announced on 27 October 2026 — a week before this book's snapshot — generates an explorable world in real time: the user moves through it with ordinary controls while the model invents the surroundings frame by frame and keeps them coherent. Genie 4 is the first real-time world model in the line to generate its audio natively — music, ambient sound, and spoken dialogue from characters inside the world arrive with the picture rather than being added to it. Prompts in natural language can steer a simulation while it runs, and a Gemini model works inside the loop as a guide, watching the scene and keeping longer sequences of events on track. Genie 4 is closed: no weights are published, and at the snapshot it is a research preview reaching trusted testers first.

World Labs, the company founded by Fei-Fei Li, takes the second approach: persistence. Its **Marble** model, launched in November 2025, turns text, images, video, or a rough layout into a three-dimensional environment that can be explored and exported — as data a game engine or simulator can use, not just as footage. A newer model, **Atlas**, entered early access on 1 September 2026 and is intended to power future versions of Marble. Both are closed, hosted products; the worlds are downloadable, the models are not. The company's circumstances are also part of the record: on 28 September 2026 AMD announced a definitive agreement to acquire World Labs for

approximately $8.2 billion in stock. The acquisition had not closed at the snapshot date. The most prominent independent world-model laboratory is, at the time of writing, in the process of becoming part of a chip company.

NVIDIA's **Cosmos** family is the third approach, and the open one. Cosmos models are built for physical AI — robotics and autonomous vehicles — and are meant to reason about the physical world, simulate it, and produce synthetic training data and robot actions. **Cosmos 3**, released in May 2026, is a single family on one architecture with two parts: a reasoner that takes text and vision in and puts text out, and a generator that takes text, vision, audio, and action in and puts vision, audio, and action out. It ships in three sizes — Super at 64 billion parameters, Nano at 16 billion, and Edge at 4 billion, the last intended to run on the robot hardware itself. Cosmos 3 is open: NVIDIA releases the source code and the model weights under the OpenMDW-1.1 license.

Meta's **V-JEPA 2** is the deliberate counter-argument. It generates nothing. Trained on over a million hours of video, it learns to predict what happens next in an abstract representation of a scene — an understanding of the world without a rendered picture of it. Given a small amount of robot data on top of that training, it can plan a robot arm's actions in environments it has never seen, without task-specific training. V-JEPA 2 was released in June 2025, and Meta published both the code and the model checkpoints for research and commercial use. Whether a world model must generate a world, or only predict one, is the live disagreement inside this kind, and both positions are held by laboratories with the resources to test them.

Robot-action models

DeepMind's **Gemini Robotics 2**, released on 30 July 2026, is a suite of three models: a planner (Gemini Robotics ER 2), the VLA itself, and a smaller on-device version that runs on the robot. The generation's step forward is scope. The 2025 models in this line manipulated objects with arms, on tables. Gemini Robotics 2 is the first in the family to control a whole humanoid body — balance, posture, and movement, in DeepMind's phrase "from feet to fingertips" — demonstrated on a partner company's humanoid robot. One model checkpoint drives more than one kind of robot body. The suite is closed and reaches early-access partners only.

Physical Intelligence, the best-known independent laboratory in this type, shows the open-to-closed pattern arriving in robotics. Its earlier models — **π0**, **π0-FAST**, and **π0.5** — are open: their base checkpoints can be downloaded from the company's openpi project, whose code is under the Apache 2.0 license. π0.5, from April 2025, added generalisation to places the robot had never been. The newer models, *π\0.6 **(November 2025) and** π0.7** (April 2026), are not in openpi. They were announced as research and, from October 2026, are offered through the company's Pi API beta. The older generation stays downloadable; the frontier of the line sits behind an API. Readers of the earlier chapters will recognise the shape of that arrangement.

NVIDIA's **GR00T** line is the open entry on this side of the shelf. GR00T models are cross-embodiment foundation models for humanoid robots: language, images, and the robot's own body state go in, motor actions come out, and the base model is then

post-trained onto a specific machine. The current downloadable release is **GR00T N1.7**, a 3-billion-parameter model whose weights are open under the NVIDIA Open Model License Agreement.

Where the shelves meet

The reason these two types share a chapter is that they are converging. GR00T N1.7's vision-language backbone is a Cosmos reasoner — a world-model component doing the seeing and thinking inside a robot-action model. Cosmos 3, from the other direction, takes action among its inputs and produces action among its outputs: a world model that already reaches into the VLA's job. One stack is forming in which a model simulates the world, plans in it, and acts in it. If that stack settles, a future edition of this book may give it one chapter with one name. It has not settled yet, and this book does not name kinds before they exist.

A closing word of calibration. The other chapters of this book describe kinds stable enough that their November 2026 shelves will still be recognisable in a year. This chapter carries no such expectation. Genie 4 was one week old at the snapshot; Atlas was two months into early access; the World Labs acquisition was announced and unclosed. The frontier shelf is where the map is drawn in pencil. It is included because a reader who meets these words — world model, VLA, physical AI — in the wild should be able to place them: what they take in, what they put out, and how young they still are.

>

>

On the shelf, November 2026 World models *— Genie 4 (Google DeepMind; closed, research preview; first real-time world model with native audio and dialogue). Marble and Atlas (World Labs; closed; persistent, exportable 3D worlds; AMD acquisition of World Labs announced at ~$8.2B in stock, not yet closed). Cosmos 3 (NVIDIA; open, OpenMDW-1.1 license; 64B / 16B / 4B). V-JEPA 2 (Meta; open code and checkpoints; predictive, non-generative).* ***Robot-action models (VLA)*** *— Gemini Robotics 2 (Google DeepMind; closed, early-access partners; whole-body humanoid control). π0, π0-FAST, π0.5 (Physical Intelligence; open checkpoints; π*0.6 and π0.7 API-only). GR00T N1.7 (NVIDIA; open weights, NVIDIA Open Model License).*

Part Two — How to Read a New Model



Part Two plate — the three-color field. Art by Douglas John Brownfield; plate setting by the assistant.

Chapter 10 — The Five Questions

Models are announced constantly, and the announcements are written to impress. A launch page leads with the largest number it can find, a name chosen by a marketing department, and a list of records the model has set. None of that tells you what the model is. Fortunately, sorting a new model does not take long. Five questions, asked in order, will place almost any model on the map in Part One — and the questions stay useful after the launch post is forgotten, because they ask about the model's shape rather than its publicity.

1. What does it take in, and what does it give back?

This is the first question because it does most of the sorting. A model that takes in text and gives back text is a talker, whatever its name suggests. A model that takes in words and gives back pictures is an image model. A model that takes in sound and gives back text is doing transcription. The exchange — what crosses the boundary in each direction — decides the type before anything else about the model matters.

In practice, read the input list the way you would read the label on a tool. A short list means a specialist. A long list means the model is multimodal, a native of more than one medium. Google's Gemini 4 Argon, the flagship in this book's November 2026 snapshot, accepts text, images, video, audio, and PDF files as input. That single fact places it: it is not a plain talker with attachments but a model built from the start to work across media. The opposite shape is just as informative. A model that takes in text and gives back numbers — ordered lists of numbers that stand for meaning — is an embedding model, one of the

librarians. It will never write you a paragraph, and it is not supposed to.

2. Does it answer, or does it act?

A model, by itself, answers. It receives input and returns output, and then it stops. Acting — using a tool, changing a file, booking, buying, running a program, checking the result and going again — requires a loop around the model: tools it can call, memory of what it has done, and permission to proceed. That loop is a layer built on top of a model, not a kind of model, and the distinction is practical rather than pedantic.

When a product is described as an agent, this question separates the two parts so you can judge each one. The model inside may be an ordinary flagship talker. The acting belongs to the harness around it. OpenAI's agent mode, for example, was superseded in September 2026 by a system of always-on agents with their own cloud computer — the models underneath changed on their own schedule, while the layer above them was rebuilt entirely. If a demonstration shows software being operated on your behalf, you are watching a layer at work. Ask which model is inside it, and ask separately what the layer is permitted to do. Those are different questions with different risks.

3. Where does it run?

Some models run in a data center and reach you through an app or an API. Others run on the device in your hand. The answer changes what the model can offer you: a model on your device works without a connection, keeps its input on the device, and answers at the speed of the hardware you own. A model in the cloud can be far larger, and it changes whenever its maker

updates it — including, occasionally, being replaced or retired underneath the products built on it.

Size words in a model's name are usually a clue to its address. The small types in this book — Google's Gemma line, Microsoft's Phi line, Apple's on-device models — are built to live on phones and laptops. Apple's current generation is explicit about the arrangement: a compact core model runs on the device, sized in the low billions of parameters, with a larger sparse model beside it that wakes only a fraction of itself for any given task. When a name carries a word like *mini*, *nano*, or *lite*, the second question to ask is whether it names a smaller cloud model or a model meant to move in with you. The two are not interchangeable, and the difference shows up the first time the connection drops.

4. Can you have the weights, and under what license?

The weights are the model itself — the numbers that everything else is built around. If the maker releases them, you can download the model, run it on your own hardware, and keep the exact version you tested. If the maker does not, you have access to a service, and the service can change.

The word *open* is where careless reading costs the most, because in current practice it covers very different arrangements. DeepSeek's flagship models are released under the MIT license, which permits nearly any use, including commercial use. Meta's Muse Glimmer is released under Apache 2.0, a similarly permissive license. Ideogram shows why the license must be read rather than assumed: its 4.0 model was released as downloadable weights under a non-commercial license, while its 4.5 flagship remained closed. Same company, same year, two

opposite answers to this question. Other releases carry territory restrictions or vendor-specific terms. So the question is never simply whether a model is open. It is: can you have the weights, and what does the license that comes with them actually permit?

5. What is it dated?

Every claim about a model belongs to a month. Lineups turn over inside a year: versions are superseded, tiers are renamed, and models are retired on schedules that their own makers publish. A statement that was accurate when it was written — in a review, a tutorial, or an earlier edition of a book — may describe a model that no longer exists in that form, or at all.

The clearest case in this book's snapshot is Sora. OpenAI's video model was, through 2025, the most prominent name in its kind. The Sora app shut down on April 26, 2026, and the models were removed from the API on September 24, 2026, with no replacement; as of November 2026, OpenAI offers no video model. Nothing about the kind changed — video models remain a type, with other makers' models on the shelf. What expired was the name, on a date. This book handles the problem by carrying its snapshot openly: the examples in Part One are stated to be the roster as verified in November 2026, and the kinds are the part meant to last. A reader can apply the same discipline to anything written about models, including this book. Note the date of the claim, and weigh it accordingly.

The five questions, together

When the next model is announced, ask:

1. What does it take in, and what does it give back?

2. Does it answer, or does it act?
3. Where does it run?
4. Can you have the weights, and under what license?
5. What is it dated?

The answers will not tell you whether the model is good. That is a separate judgment, and the next two chapters take it up — first the names and numbers attached to a model, then the scores it is advertised with. But the five questions will tell you what kind of thing you are looking at, and most confusion about AI models is, at bottom, a failure to sort before judging.

Chapter 11 — Names and Numbers

A model name looks like a label. Read it as an advertisement that happens to contain some facts. Every name on the November 2026 shelf was assembled by a marketing department out of real parts — a family name, a generation number, a tier word, sometimes a size, sometimes a date. The parts are worth decoding, because each one tells you something true. The warning comes first, though, because it governs everything after it: **the name is the least reliable information about a model.** It was chosen to sell the model, not to describe it. Chapter 10's five questions — what kind is it, what goes in and out, open or closed, how big and what does it run on, who made it and when — will tell you more in a minute than the name tells you at all. Decode the name anyway. Just never let it outrank the questions.

The anatomy of a name

Most names on the shelf are built from the same kit, in roughly this order: **family, generation, tier, size, date.** Not every name carries every part, and vendors arrange them differently, but once you see the kit you can take almost any name apart.

The **family name** is the vendor's brand for a line of models: GPT, Claude, Gemini, Llama, Qwen, Mistral. It tells you who made it and which shelf it sits on, and it is the most durable part of the name — families last for years while everything after them churns. It tells you nothing about capability. A family name is a surname, not a description.

The **generation number** marks the family's big revisions: Gemini 4, Llama 4, Suno v6, Kling 4.0. Treat it as an edition number on a book. A higher number means a later, usually stronger revision of the same line — and nothing more precise than that. Generation numbers are not comparable across families (one vendor's 4 is another's 6, and both numbers were chosen by the vendor), and they are not even consistent within a family: the decimal after the generation — the 5.5 in Claude Opus 5.5, the 3.1 in Veo 3.1 — marks a smaller step, but how much smaller is the vendor's private judgment. Anthropic's shelf in November 2026 carries Claude Fable 5.1 above Opus 5.5 above Sonnet 5.5: three models whose numbers differ by tenths, released within weeks of each other, sitting in a strict capability order that the numbers alone would never tell you.

Then come the **tier words**, and these deserve their own section, because they are where names mislead most politely.

Tier words rank a family, not the field

OpenAI's current line is the cleanest worked example on the shelf. The GPT-6 family is sold in named tiers: **Astra** at the top, then **Sol**, then **Terra**, with **Luna** as the high-volume tier. None of those words describes what the model does. They are ranks — a ladder inside one family, in descending order of capability and price. Google does the same thing with different words: Gemini 4 **Argon** is the generally available flagship, and Gemini 4 **Carbon**, announced as the stronger model, is gated to government and trusted testers first. Anthropic's ladder is Fable, Opus, Sonnet, Haiku — poetry instead of astronomy, same device.

The older tier vocabulary — **Pro, Max, Ultra, Flash, Turbo, Mini** — works identically, and the shelf is covered in it: DeepSeek V4-Pro, Qwen3.8-Max, ElevenLabs' Flash v2.5, Kling's Turbo tier, the mini and nano models every vendor parks at the cheap end. Here is the rule that makes all of these words safe to read: **a tier word is a comparison inside one family and nowhere else.** "Max" means the top of *that* vendor's current line. It does not mean the strongest model of its type, it does not mean stronger than another vendor's plain model, and it does not stay true — when the next generation ships, last year's Max becomes this year's middle. "Flash" and "Turbo" promise speed relative to their siblings, not speed relative to the field. "Ultra" at one company can sit below "Pro" at another, and both companies are being honest by their own private scales. The words are not lies. They are local currency, and they do not exchange.

The numbers in open names

Closed models hide their size; open-weight models often print it in the name, and those numbers reward a closer read. When a name carries a figure like 3B or 9B, it is a parameter count — billions of parameters, roughly the model's size, and a fair guide to what hardware it needs. Alibaba's small Qwen3.5 series runs 0.8B, 2B, 4B, 9B, and Meta's small Llama 3.2 models are named 1B and 3B: in both cases the number is the point of the model, because it tells you the thing runs on a phone or a modest laptop.

Mixture-of-experts models carry **two** numbers, and confusing them is the most common honest mistake in reading a name. An MoE model holds a large total pool of parameters but activates only a fraction of them for any given piece of work. Mistral

Large 4 is the shelf's clear example: **1T total, 52B active** — one trillion parameters on the shelf, about fifty-two billion awake at a time. Alibaba's open Max-class release makes the same distinction inside the name itself: Qwen3.8-2.4T-A95B, total first, active after the A. Apple's on-device line shows why the distinction matters in practice: alongside a plain 3B model sits a 20B sparse model that activates only 1–4B parameters at a time, which is how something with 20B in its name runs on a phone. Read both numbers when a name offers both. The total tells you the model's capacity and its storage weight; the active figure tells you its running cost. A name that offers only one number is telling you only half of that.

Dates, snapshots, and the same name twice

The last part of the kit is the date, and it cuts two ways. Some names and identifiers carry dates outright: Anthropic's computer-use toolset is versioned computer_toolset_20260801 — the date is in the name, August 1, 2026, so a developer can pin an exact, unchanging thing. API model names commonly work the same way, with a dated snapshot suffix marking one frozen version. A dated name is the most honest name on the shelf: it promises that the thing behind it will not change.

The undated name promises the opposite. A bare name in an API is frequently an **alias** — a pointer the vendor can re-aim. OpenAI's gpt-5.6 alias routes to GPT-5.6 Sol; the name you call is not a model but an address, and what answers at that address is the vendor's current choice. The same effect arrives without aliases, too: a family word like Opus has now covered several distinct models across the 4 and 5 generations, each quietly replacing the last under a word that stayed still. So the rule from

Chapter 10 applies with full force here — *who made it, and when* is a question about a specific dated model, never about a name. When a name matters to your work, find its date. When it has no date, assume it is a moving target and plan accordingly.

The standing rule

None of this makes names useless. The family tells you the maker, the generation tells you the edition, the tier word tells you the rank at home, the printed numbers — read in pairs, on MoE models — tell you the size honestly, and a date, when present, pins the whole thing down. What a name cannot do is compare across vendors, promise a capability, or substitute for the five questions. Names are marketing first and description second, written by the seller, in the seller's private vocabulary, at the moment of sale. Read them the way you read a label on a market stall: useful for finding the stall again, no substitute for picking the goods up. The questions are the picking up.

Chapter 12 — Benchmarks Without Tears

Every model launch arrives with numbers. A new model is announced, and within the first paragraph there is a score: so many points on this test, a few points ahead of a rival on that one. The numbers look like verdicts. They are not. They are measurements, taken under particular conditions, and they become useful only when you know what was measured, when, and by whom. This chapter is about reading those numbers calmly.

What a benchmark is

A benchmark is a fixed test that many models take. The test might be a set of mathematics problems, a collection of coding tasks, a battery of questions drawn from school and professional examinations, or a series of instructions that human graders score against a rubric. The point of fixing the test is comparison: if every model answers the same questions under the same rules, the results can be set side by side.

That is all a benchmark score is. It is a record of how one model performed on one test, on one date, under one set of settings. Change any of those three — the test, the date, the settings — and the number can change with them. A score is not a property of a model in the way its size is. It is closer to a race time: real, recorded, and meaningful only alongside the course and the conditions.

What leaderboards measure, and what they cannot

A leaderboard gathers benchmark results into a ranking. Rankings are easy to read, which is exactly their danger. A leaderboard can tell you which models performed best on the tests it tracks, as those tests were run. It cannot tell you which model will write the clearest email for you, follow your house style, handle your spreadsheet, or stay accurate across a long document in your field. Your work is not on the test.

Two kinds of leaderboard are worth knowing apart. Some scores come from tests a laboratory runs on its own models — or on rivals' models — and publishes itself. Others come from standing public boards that run comparisons continuously and date what they publish. LMArena is the best known of the public kind: it pits models against each other in blind comparisons and ranks them by how people vote. Artificial Analysis is another standing public source, tracking models across quality, speed, and price, with its methods and dates published alongside the results. Neither board settles which model you should use. Both are useful for the same reason: they are third parties, they work to a stated method, and their results carry dates, so you can see whether a ranking is current or a season old.

Settings matter as much as the test itself. Many models can be run at different levels of effort — more time spent working through a problem, more attempts made before answering — and higher effort usually means a higher score at a higher cost. Two results on the same test are comparable only if the settings were comparable, and launch materials do not always say what settings were used. When they do not, read the score as the model's best case, not its everyday one.

Why vendors quote the tests they win

A launch post is an advertisement as well as a report. The laboratory chooses which benchmarks to feature, and it features the ones its model won. A model that leads on coding tests will be introduced as a coding model with the coding scores up front; a model that leads on examination questions will arrive wearing those. Nothing about this is dishonest by itself. The tests are usually real, and the scores are usually accurate as far as they go. But the selection is the vendor's, and a reader who sees only the selected tests has seen the model's best subjects, not its whole report card.

The tell is absence. When a launch post is silent about a well-known test that earlier launches from the same company quoted proudly, the silence usually means the new model did not win it. You cannot learn a fact from an absence, but you can learn where to look next.

Contamination and teaching to the test

There is a second reason to hold benchmark scores loosely. A benchmark only measures well if the model has not seen the test before. Test questions circulate. They are quoted in articles, posted in forums, and folded into the enormous bodies of text that models are trained on. When a model's training data includes the test — or close copies of its questions — the model is no longer solving the problems cold. It is, in part, remembering them. This is called contamination, and it inflates scores in a way no reader can detect from the number alone.

Alongside contamination sits a plainer habit: teaching to the test. Once a benchmark becomes the industry's scoreboard, laboratories tune their models toward it, because the scoreboard is what gets quoted. The model genuinely improves at that test.

Whether it improves at everything the test was meant to stand for is a separate question, and usually nobody runs it. Old benchmarks also wear out: as models cluster near the top of a test, the test stops separating them, and the field moves on to a harder one. A benchmark's age is therefore part of its meaning. A high score on a saturated test from several years ago says less than a middling score on a current one.

The reader's working method

None of this requires you to ignore benchmarks. It requires you to read them the way you would read any claim: with four questions.

Check the date. Models turn over in months. A score from a year ago describes a model that may have been revised, replaced, or retired since. Dated results can be weighed; undated ones cannot.

Check who ran it. A result published by the model's own maker is a claim. The same result confirmed by a public board or an independent evaluator is evidence. Both can be true; they do not carry the same weight.

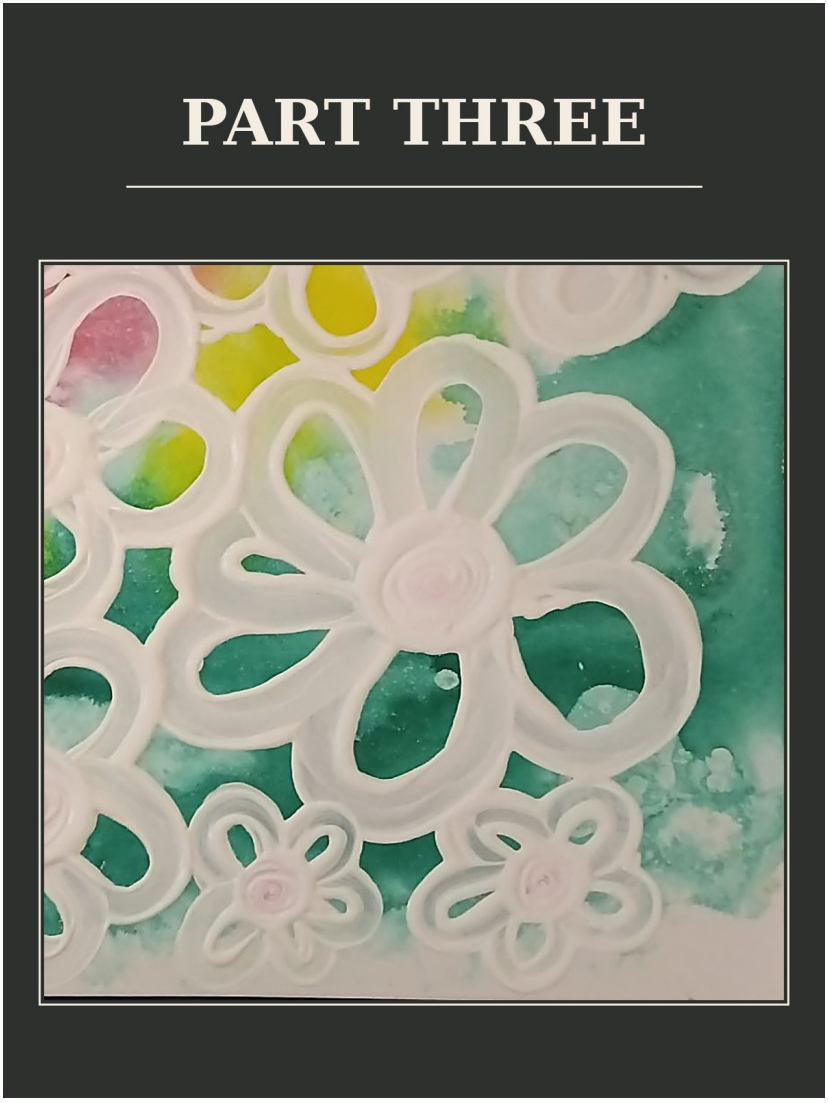
Prefer the test closest to your task. If you need a type of model for translation, a translation benchmark matters more than a mathematics one, however famous. If none of the standard tests resembles your work, the honest test is a small one of your own: a handful of real tasks, run through each candidate, judged by you.

Treat any undated score as advertising. A number with no test name, no date, and no source is not information. It is decoration on a claim, and it should move your judgment not at

all.

Read this way, benchmarks return to their proper size. They are a first filter — a quick way to see which models are in the leading group of their kind, and which claims deserve a closer look. The final judgment belongs to the only test that includes your work: yours.

Part Three — Choosing



Part Three plate — the finished piece, detail. Art by Douglas John Brownfield; plate setting by the assistant.

Chapter 13 — Match the Kind to the Job

Most people choose a model the way they choose a phone: by the ranking at the top of a chart. That is backwards. A leaderboard answers the question "which model scored highest on a test someone else designed?" The question that matters is narrower and plainer: "what work am I handing over, and what kind of model does that work?"

So start with the job, stated in ordinary words. Not "I need AI." Something like: draft a letter. Summarize a long report. Find the passage in my own library that mentions a contract date. Make a picture for a page. Read a chart and tell me what changed. Watch a counter and speak up when it moves. Each of those sentences already points at a kind, because each one fixes what goes in and what comes out. Text in and text out is a talker. A large library in and a short list of locations out is a librarian. Privacy-sensitive material that should never leave the room is a small on-device model. A description in and a picture out is an image type. The match is made by the shape of the work, before any brand name enters the conversation.

Worked examples, at the family level, as the roster stands in November 2026:

- **Drafting and summarizing** belong to the large language models, the talkers. This is the most served job on the shelf: the GPT-6 family, Claude's 5.5 family, Gemini 4 Argon, and the open-weight talkers from Mistral, DeepSeek, and Qwen all do it well. For ordinary drafting, the differences between

current flagships are smaller than the differences between a clear request and a vague one.

- **Finding things in a large collection** belongs to the embedding and retrieval models — the librarians. A talker asked to "find it" in ten thousand documents is being used as a very expensive librarian and will sometimes invent the shelf it claims to have checked. The librarian kind — OpenAI's text-embedding-3 pair, the Voyage 4 family, the open BGE-M3 — turns the collection into something searchable first; the talker then works on what was actually found.
- **Private material** belongs on the device. Medical notes, family finances, a diary, unreleased work: if the content should not travel, the model should not either. The small on-device types exist for exactly this — Gemma 4 under the Apache 2.0 license, Microsoft's Phi-4 line under MIT, Apple's on-device models, the small Qwen series. They are less capable than a flagship talker and that is the correct trade: the job's first requirement was privacy, and capability was second.
- **Pictures** belong to the image types: Midjourney, OpenAI's GPT Image line, Google's Gemini-native image family, the FLUX family. A talker with image output bolted on is a different tool from a model built for pictures, and the difference shows in editing and consistency across a set.
- **Reading a chart, a form, or a photographed page** belongs to the multimodal talkers. As Chapter 6 recorded, seeing is now a property of the flagships rather than a shelf of its own: Gemini 4 Argon takes text, image, video, audio, and PDF natively, and its peers do the same. The matching point is

simply to check that the chosen talker accepts the input the job actually has — a scanned document is not text until some model can see it.

- **Holding a conversation** belongs to a talker too, but latency picks the tier. A spoken or live exchange fails if each answer takes ten seconds, however good the answer is. The lighter, faster members of a flagship family — or a small model for narrow exchanges — serve conversation better than the largest model answering late.
- **Hard problems** are not a separate purchase. As Chapter 2 recorded, reasoning is no longer a kind of its own; it is a mode of the flagship talkers, usually an effort setting. The matching move is not to shop for a "reasoning model" but to raise the effort on the talker already chosen — and only for work that repays slow thinking: proofs, plans with many constraints, code that must be right. For a summary, high effort buys delay, not quality.
- **Watching a counter** — a price, a gauge, a status page — is a job of patience, not power. It wants a small, cheap model that can look often, or a librarian-style check against a threshold, with a talker called in only when something actually moved. Running a flagship continuously to watch a number is paying for eloquence that the job never uses.

Underneath all of these sit three costs, and they trade against each other in a fixed pattern. A bigger model is generally slower to answer, more expensive per answer, and further away — it runs in someone else's data center, under someone else's terms, on a connection that can fail. A smaller model is faster, cheaper, and can be near or on the device, at the price of capability. There is no direction that wins on all three. "Best" without a job

attached is a word without a referent; the honest question is which of the three costs the job can afford and which it cannot.

One caution completes the method. Prefer the kind over the vendor, and keep the work portable. A model name is a product, and products are withdrawn. Sora is the standing example in this book: the most publicized video model of its period was shut down in 2026 — the app in April, the API in September — with no replacement, and every workflow built on that name alone had to be rebuilt or abandoned. Workflows described in terms of kinds — "a video type that takes a still and a prompt" — survive the withdrawal of any one name, because another member of the kind can be put in its place. Names are how models are sold. Kinds are how work is planned.

Before choosing, run the short list:

1. State the job in one plain sentence. If the sentence will not come, the choice cannot either.
2. Name what goes in and what comes out. That pair selects the kind.
3. Ask where the material may travel. If it may not leave, the answer is on-device, whatever the leaderboards say.
4. Ask how often the job runs. Rare and hard argues for a flagship with effort raised; constant and simple argues for something small.
5. Ask what happens if this model is withdrawn next year. If the honest answer is "we start over," describe the job in kind terms now, while the description is cheap.

The next chapter takes up what happens when one kind is not enough — when a job needs a finder, a drafter, and a checker

working in sequence, held together by the layer this book set aside in Part One.

Chapter 14 — The Layer: Loops, Tools, and Models in Combination

This book has sorted models into kinds: talkers, image models, video models, and the rest. One phrase in common use names something real that does not belong on that list, and this closing chapter exists to say where it belongs instead. The phrase is *loop model*. People use it for systems that do not merely answer and stop — systems that take a task, act on it, look at what happened, and act again, sometimes for hours. The thing they are pointing at exists. It is simply not a type of model. It is a layer built around one, and the distinction matters for the same reason every distinction in this book matters: if you sort it wrong, you will shop for it wrong, price it wrong, and blame the wrong part when it fails.

A model, on its own, does one thing: it takes input and returns output. A talker given a question returns text. It does not open the file the answer depends on, it does not check whether the file changed since yesterday, and it does not notice that its first answer was wrong. Everything that turns a model into a worker is supplied by software around the model. That surrounding software is called a harness, and a system made of a model plus a harness is what the trade usually calls an agent. The model inside an agent is one of the kinds this book has already covered — most often a talker, sometimes a multimodal model. The loop is not in the model. The loop is the harness's way of calling the model, again and again, until the work is done or a limit is reached.

A harness has four working parts, and each one can be described plainly.

The first is the loop itself: act, observe, adjust. The model proposes an action — run this search, open that page, write this draft. The harness carries the action out and brings back the result. The model reads the result and proposes the next action, which may be a correction of the last one. A single pass through that cycle is a turn. A useful agent is mostly this cycle, repeated with judgment about when to stop: the task is finished, the evidence is sufficient, or further attempts are producing the same failure and a person should be asked.

The second part is tools. A tool is a defined action the harness will perform at the model's request: read a file, query a database, search the web, send nothing yet but prepare a draft, run a calculation, operate a browser. Tools are where a model's statements meet the world. They are also where the harness keeps control, because every tool is a door the harness's builder chose to install, with a defined shape — this tool can read but not write, that one can prepare a message but not send it.

The third part is memory. A model remembers nothing between calls; whatever continuity an agent has is stored by the harness and handed back to the model as context: the task list, the notes from earlier turns, the record of what has already been tried and what it produced. When an agent appears to remember your preferences from last week, it is the harness retrieving them, not the model recalling them.

The fourth part is permissions, and it is the part that decides how much any of this should be trusted. Permissions state what the system may do on its own, what it may do only with a person's

approval, and what it may not do at all. A mature harness treats sending, publishing, paying, and deleting as a different class of act from reading and drafting, and it routes the first class through a human being.

As of November 2026, the layer is not a research curiosity; it is shipping product, and it is moving at least as fast as the models are. Anthropic's computer use — a model seeing a desktop through screenshots and driving it with mouse and keyboard — has graduated from beta to general availability on its API, packaged as a dated toolset of seventeen member tools, alongside a separate toolset that acts on web pages directly rather than through pixels. OpenAI's agent mode, the task-at-a-time agent it shipped in 2025, was superseded in September 2026 by a product called dots: always-on agents, each with its own cloud computer and its own browser, intended to keep working toward a person's goals between conversations rather than starting cold at each request. Google's path ran the other direction. Its named agent experiment, Project Mariner, was shut down on 4 May 2026, and the capability did not disappear — it dissolved into Gemini Agent inside the Gemini app, and into agent features spread through Google's other products. The experiment ended because the layer had become a feature, not because it had failed.

The layer also has an open shelf, and it matters for the same reason open weights matter: a harness you can read is a harness whose permissions you can verify. Browser Use, an open-source library that lets almost any model drive a real browser in a loop, is released under the MIT license and has become standard equipment that other agent products build on. OpenHands, which began as an open coding agent, has grown into Agent

Canvas — a self-hosted control center that runs many different agents, from several different makers, under one set of rules — also MIT licensed. That second fact is the layer's clearest self-portrait in 2026: the leading open agent project now describes itself as a panel for controlling other agents. Harnesses are being stacked on harnesses, which is only possible because the layer is a real, separable thing.

In practice, most serious work at the layer is not one model in a loop but several kinds in a chain. A retriever finds the relevant material and returns it with its sources. A talker drafts from that material. A checker — a second model call, a rules engine, or a person — tests the draft against the sources and against the facts it can verify independently. The loop carries the whole sequence, keeps the record of what was checked, and stops where it is told to stop. Each kind in the chain is doing the job its chapter in this book describes; the harness is what makes their separate outputs add up to finished work.

Consider one ordinary arrangement, described by its method only. A one-person operation — publishing, correspondence, a schedule of recurring products — is run day to day by one person working with an AI collaborator. Nothing in the arrangement is exotic. The collaborator works from written instructions and a standing record, so its context does not depend on anyone's memory of last month's conversation. Work runs on schedules: drafts are prepared before they are needed, never at the deadline. Every draft passes a verification gate before it reaches the person — figures are re-derived from their sources, names and dates are checked against the record, and anything that cannot be confirmed is flagged rather than smoothed over. The person approves what goes out; the system

does not publish, send, or commit on its own authority. When the person corrects the work, the correction is written into the standing record the same day, so the same mistake is not offered twice. The models inside such an arrangement are interchangeable — a better talker can be swapped in when one appears, and the schedules, gates, and records carry on unchanged. That interchangeability is the practical proof of this chapter's claim: the durable asset is the layer and its discipline, not any single model passing through it.

That is also where the person stands in the loop, and it is the right note to end on. A harness multiplies what a model can do, but it does not decide what should be done, and it does not carry responsibility for the result. Permissions are the formal expression of that fact: the wider the system's freedom to act, the more explicit its limits and its approval points have to be. The kinds of models will keep changing — this book's snapshot will age, some of its shelf entries will melt or be retired, and new types will form that have no chapter here. The layer will change with them. What does not change is the sorting habit this book has tried to teach: ask what kind of thing you are looking at, ask what is model and what is machinery around it, and keep the deciding — and the responsibility — with the person the work is for.

Back Matter

The Snapshot Statement

This book is a map of kinds, and a map is only as good as its date. The types themselves — talkers, image models, librarians, and the rest — change slowly. The examples do not. Every named model, version, license, and event in this book is therefore stated as of **November 2026**, and each was checked against the maker's own pages, papers, model cards, or repositories, where those exist.

Where no primary source could be reached, the rule was simple: the text says less rather than more. A few dates rest on consistent secondary reporting and are stated at the precision it supports — "early 2026," not a day. Items that could not be verified at all were left out, and three are worth naming here, because the leaving-out is part of the record:

- **A rumored "GPT-6.2 Astra"** — reported from dashboard sightings; it appears in no official OpenAI catalog and has not been announced.
- **A "Gemini 4 Deep Think"** — DeepMind's own page still describes Deep Think as built on Gemini 3.1 Pro. Whether it has moved to Gemini 4 is unconfirmed, so this book does not claim it.
- **Two world-model laboratories known only from roundup articles** — no paper, model card, or lab page could be checked, so the frontier chapter does not include them. A shelf in this book is earned by a source, not by coverage.

The online editions of this book are updated in place as the roster moves. Each carries a version line and a date; when a named model is retired, renamed, or replaced, the update says so, and the dated editions stand as the record of their moment.

Sources

Credit where the record came from. The roster in this book rests on the makers' own publications, grouped by kind. Secondary reporting was used only where noted in the text.

Large language models

- OpenAI, API documentation and model pages (developers.openai.com) — the GPT-6 and GPT-5.6 families and the deprecations schedule.
- Anthropic, Claude Platform documentation, models overview (platform.claude.com) — the Claude 5.5 family, its tiers, and thinking on by default.
- Anthropic, news index and the Sonnet 5.5 system card — release dates, and the gated Mythos 5.1.
- Google, "Google AI updates from October 2026" (blog.google) — Gemini 4 Argon's availability and Gemini 4 Carbon's gated testing.
- Meta AI, "Introducing Muse Spark" (ai.meta.com) — the frontier moving from Llama to Muse.
- Mistral AI, "Mistral Large 4" (mistral.ai) — scale, weights release, and license.
- DeepSeek, API documentation change log (api-docs.deepseek.com) and the V4 technical report on Hugging Face — the V4 releases; R1 and V3 retired from

the API.

- Alibaba Cloud, Qwen3.8 announcement (alibabacloud.com) — the hosted flagship and the first open Max-class weights.

Reasoning and multimodal

- OpenAI, deprecations schedule (developers.openai.com) — the o-series shutdown dates and their replacements.
- Google DeepMind, the Deep Think model page (deepmind.google) — Deep Think as built on Gemini 3.1 Pro, the checked state at the snapshot.
- The flagship pages above — reasoning as an effort setting, and native multimodal input, in each maker's own documentation.

Image models

- Midjourney, documentation version page — V8.2 as the default model.
- OpenAI, API reference for image generation — the GPT Image line that replaced DALL·E.
- Google, Firebase AI Logic migration documentation — Imagen's shutdown dates and its Gemini-native replacements.
- Black Forest Labs, FLUX documentation (docs.bfl.ai) — the FLUX.2 tiers and their licenses.
- Ideogram, licensing page and the 4.0 announcement (ideogram.ai) — the non-commercial weights agreement, distinct from the Apache 2.0 inference code.

Video models

- OpenAI, video-generation guide (developers.openai.com) — Sora's shutdown, stated by the maker and kept for historical reference.
- Runway, developer documentation (docs.dev.runwayml.com) — Gen-4.5 and the Aleph 2.0 editing model.

Audio models

- OpenAI, Whisper repository and model record — the open weights, unchanged under the MIT license.
- ElevenLabs, model documentation — Eleven v3, Flash v2.5, and the deprecated Turbo line.

Embedding and retrieval models

- MongoDB, Voyage AI model lifecycle documentation — the Voyage 4 family active, the Voyage 3 series legacy.
- BAAI, the BGE-M3 model card on Hugging Face — the open embedder's specification and standing.

Small and on-device models

- Microsoft Research, the Phi-4 reasoning-vision post, technical report, and Hugging Face card — the current Phi line, MIT-licensed.
- Google, the Gemma 4 announcement (blog.google) — the family, its sizes, and the move to Apache 2.0.
- Apple Machine Learning Research, the third-generation Foundation Models post and the 2025 technical report — the two on-device tiers.
- Qwen team, the Qwen3.5 repository on GitHub and the Qwen3 technical report (arXiv) — the small series and its

dates.

The frontier shelf

- Google DeepMind, the Genie 4 and Gemini Robotics 2 posts (deepmind.google) — the world model and the robot models.
- World Labs, the Atlas announcement (worldlabs.ai), and AMD's newsroom release — Atlas in early access; the acquisition agreement.
- NVIDIA, the Cosmos repository and the Cosmos 3 technical report — sizes, and the OpenMDW-1.1 license.
- Meta AI, the V-JEPA 2 post, paper on arXiv, code on GitHub — the predictive approach, openly released.
- Physical Intelligence, the company blog, the openpi repository, and the π0.5 paper (arXiv) — which checkpoints download and which models are API-only.
- NVIDIA, the GR00T N1.7 model card on Hugging Face and the GR00T white paper (arXiv) — the license, read from the card itself.

The layer

- Anthropic, the computer use tool documentation (platform.claude.com) — the toolset's general-availability state.
- OpenAI, "Introducing dots" (openai.com) — the current agent product and its guardrails.
- Google, the Gemini 3 announcement (blog.google) — Gemini Agent's launch in the Gemini app.

- Browser Use and OpenHands, project repositories on GitHub — the open harnesses, both MIT-licensed.

The Map on One Page

The nine kinds

1. **Large language models** — text in, text out: the talkers, and the kind everything else starts from.
2. **Reasoning models** — models that work a problem in steps before they answer.
3. **Image models** — pictures from words: generation, editing, and holding a subject consistent.
4. **Video models** — moving pictures from words and stills, a few seconds at a time.
5. **Audio models** — sound in three directions: speech out, transcription in, and music.
6. **Multimodal models** — native in more than one medium at once: they see, hear, and talk in a single model.
7. **Small and on-device models** — types built to run on a phone or a single machine: private, offline, and instant.
8. **Embedding and retrieval models** — the librarians: meaning turned into numbers, so anything can be searched by sense.
9. **The frontier shelf** — kinds still forming: world models that simulate a place, and robot-action models that move a body in one.

Two melts, recorded as of November 2026. Reasoning has dissolved from a kind into a *mode* — a setting inside the

flagship talkers, covered in its chapter as the record of that change. Multimodal is becoming a *property* — something every flagship now has, rather than a shelf of its own. Kinds can melt. The map records it when they do.

Not a kind: the layer. A loop — tools, memory, turns, and permissions wrapped around a model until it can do a job — is a system built *on* a model, not a type of model. It is covered in Part Three, where it belongs.

The five questions, for any new model: What kind is it? What goes in, and what comes out? Are the weights open or closed — and under what license? How big is it, and what does it run on? Who made it, and when?

Names will change by the next printing. The questions will still sort them.

About the Art

The art in this book is one painting, three times — and then once more, finished.

It is a canvas by the artist, photographed as it was made. First the color field: red and pink blooming at one corner, teal laid over the rest. Then yellow worked in across the top, until the three colors share the surface. Then the finish: white loop-flowers raised over the whole field, so the completed piece is loops, hundreds of them, standing on one shared ground.

The stages open the parts of the book in order — the first washes before Part One, the three-color field before Part Two, the finished surface before Part Three — and the finished piece is on the cover. Two closer photographs of the finished canvas, taken at an angle so the raised loops catch the light, appear inside.

The plates are cropped from the artist's photographs to the canvas only, with the colors unaltered. The cover and the part plates are settings made from those plates: the finished painting set full-width on a dark ground sampled and darkened from its own deep teal, the part plates cropped as bands and details, each in a thin cream frame on a ground sampled from its stage, with light sharpening only. The painting itself is nowhere retouched.

Art credit: the painting and the photographs are by Douglas John Brownfield. The cover setting and the part plates were made by the assistant from the artist's plates, as described above.



Stage one — first washes: red and pink blooming at one corner, teal over the rest. Art by Douglas John Brownfield.



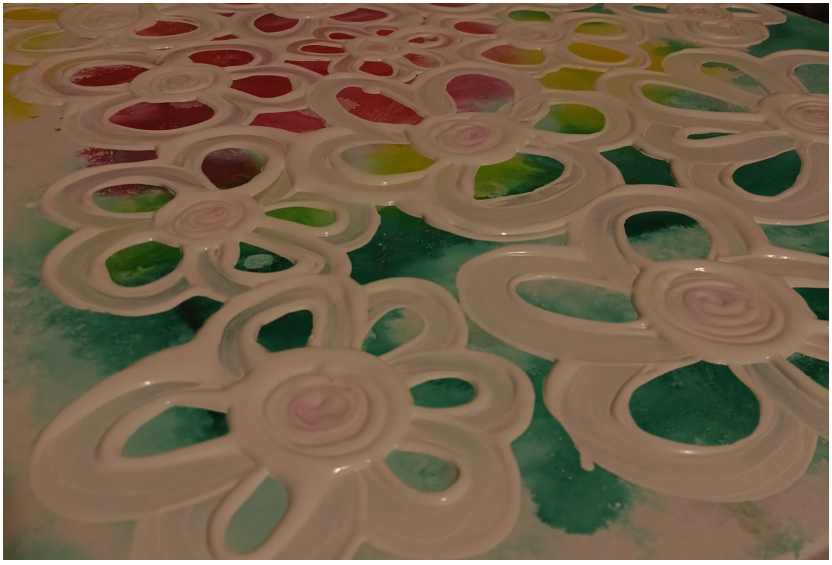
Stage two — yellow laid in across the top; the three colors share the surface. Art by Douglas John Brownfield.



Stage three — the finished piece: white raised loop-flowers over the whole color field. Art by Douglas John Brownfield.



Detail of the finished canvas — the near-edge loop-flowers. Art by Douglas John Brownfield.



Detail of the finished canvas — the flower field in raking light. Art by Douglas John Brownfield.

About the Author

Douglas J. Brownfield is an independent builder and artist based in America. The author writes plain-language books about the tools of the present moment — what they are, what they are for, and how to tell one from another — working from primary sources and dating every claim that can go stale.

This book began with a practical problem. New AI models are announced faster than anyone can follow, and the names turn over within the year, while the kinds underneath them hold still. The author went looking for the map — kinds first, current models as dated examples inside each kind — and wrote the one he had needed.

The artist's paintings carry the art of his books. The one in this book, loops over a shared field, is described in About the Art.

Also by Douglas J. Brownfield

Words to Prompt AI Well With — a field guide to asking well: how to prompt AI for better answers, in seventy-two short entries.

Before Q-Day — how to prepare Bitcoin and crypto holdings for the quantum change: what is settled, what is coming, and what to do in the meantime.