KhaledMostafaMe



Advanced Typed Decisions

Readme

Typed yes/no, category and scale decisions with calibrated probabilities - on the Bot Agent, or via Jev, Clef, ScaleDown, OpenAI or your own server.

Version 2.8.4-lite

11/11/2026

Table of Contents

1. Introduction 3
1.1 Overview 3
1.1.1 Sample input and output 3
1.2 Use cases 4
2. Requirements & Prerequisites 5
2.1 System Requirements 5
2.2 Prerequisites 5
3. Getting Started 6
3.1 Quick Start 6
3.1.1 Setup 6
3.1.2 Configuration and Use 6
3.1.3 Files included in this submission 7
3.1.4 Actions 8
4. Support & FAQs 13
4.1 Support 13
4.2 FAQs 13
4.2.1 Does any text leave the Bot Agent? 13
4.2.2 Where are the models? The package is only 24 MB. 13
4.2.3 Which languages are supported? 13
4.2.4 How many categories can a question have? 13
4.2.5 What does confidence mean? 13
4.2.6 The action says there is not enough memory. 14
4.2.7 Which model should I use? 14
4.2.8 Which engine should I use? 14
Appendix A: Record of Changes 15
Appendix B: References 16

1. Introduction

This document contains everything needed to install and use the Advanced Typed Decisions package: what it does, what it requires, how to set it up, how each group of actions is used, and its known limits.

1.1 Overview

Advanced Typed Decisions is an Automation 360 package that lets a bot ask typed questions about text and get back answers it can branch on directly, each with a calibrated probability. By default it runs open decision models entirely on the Bot Agent - Laya (General, Specialist, Multilingual) by Convai Innovations, which reproduces TypeSafe's Jev system-one API, and Strands Decider - with no account, no API key, no per-call fee, and the text never leaves the machine. Each action can instead use TypeSafe's Jev cloud API, Cloudflare's Clef models on Workers AI, ScaleDown's Constrained Decoding Model, or a self-hosted Jev-compatible server such as Strands Decider, Kev or Jeff; the outputs are the same whichever engine answers. Not affiliated with TypeSafe AI, Cloudflare or ScaleDown.

There are three kinds of question. A yes/no question returns the probability that the statement is true and a Boolean answer at a threshold you choose. A category question returns the best category and a probability for every category. A scale question returns the expected position on an ordered scale and the most likely level. Ask Questions answers any mix of these about the same text in one model pass.

Beyond text: on the Jev engine a question can look at a PNG or JPEG image, and Ask Questions supports Jev 1.14's box question type, which returns where something is in the image (top, bottom, left and right as fractions of the image); TypeSafe marks images and box as experimental. Cloudflare's Clef models read images too, Clef-omni also listens to audio and watches video, and ScaleDown reads a PDF or scanned image together with the text.

Every answer includes a confidence score (1 minus the normalized entropy of the answer distribution), and a `needs_review` flag that is true when confidence is below a percentage you set, so unclear cases can go to a person.

1.1.1 Sample input and output

> **Input text**
>
> *Subject: Refund not received. I cancelled my subscription two weeks ago and I still have not received my refund. This is the third time I am writing. If this is not resolved I will dispute the charge with my bank.*

Action	Question	Result
Choose Category	Which team should handle this ticket? categories billing / support / sales	answer = billing probability = 0.9342 `needs_review` = true at an 80% review threshold (confidence 0.7376)

Action	Question	Result
Rate on Scale	How urgent is this ticket? levels not urgent / somewhat urgent / urgent / critical	answer (expected level) = 1.4264 level = 2 `level_label` = urgent
Ask Yes/No	Is the customer likely to cancel or dispute?	answer = true probability = 0.8926

Values from the demo bot's run on a Windows Bot Agent.

1.2 Use cases

The key use cases include:

- **Support ticket triage**: route to a team, rate urgency and flag churn risk in one pass.
- **Email and case intake**: decide whether a message is a complaint, a request or spam before a bot acts on it.
- **Document checks**: ask yes/no questions about extracted text (for example, does the invoice include VAT) and send uncertain answers to review.
- Quality and compliance screening of free-text fields with a confidence-based human-in-the-loop step.

2. Requirements & Prerequisites

2.1 System Requirements

- Automation 360 Control Room; Windows Bot Agent 20.11 or later.
- 8 GB RAM recommended on the Bot Agent (a loaded model holds about 1.7 GB).
- Internet access to huggingface.co on first use of each model (or a copied model file), and about 2.5 GB free disk space per model: the download plus the one-time unpack.

2.2 Prerequisites

- Permission to add packages in the Control Room (Manage > Packages).
- The default Built-in engine needs no credentials, lockers or external accounts.
- A cloud engine needs an account with that provider and its key in the Credential Vault: a TypeSafe API key (Jev), a Cloudflare account ID and API token made from the Workers AI template (Clef; the free plan includes a daily allowance), a ScaleDown API key, an OpenAI API key (OpenAI Decisions), or the key of the OpenAI-compatible chat API you use. A self-hosted server needs its URL and, if it asks for one, a key.

3. Getting Started

3.1 Quick Start

3.1.1 Setup

1. In the Control Room, go to Manage > Packages > Add package and upload the package jar (about 24 MB).
2. Enable the package and set it as the default version.
3. Open a bot in the editor: the actions appear under Advanced Typed Decisions in the Actions palette, in two groups, Decide and Model.

3.1.2 Configuration and Use

Engine

Every decision action has an Engine dropdown; the settings below it change with the engine. Cloud and self-hosted engines also take a Connection timeout and a Response timeout in seconds.

Engine	Settings
Built-in	Model: General, Specialist, Multilingual, Strands Decider 2B or Strands Decider 2B v1-2610. Runs on the Bot Agent.
TypeSafe Jev	API key; model (jev-latest, jev-preview or a pinned version such as jev-1.14.0) Optional: Image file, Long context
Cloudflare Workers AI	Account ID and API token; model (Clef-flash, Clef-omni, Clef, or Other with a newer model's name) Optional: Image file; Audio or video file for Clef-omni
ScaleDown CDM	API key; model (classify-1, classify, or Other) Optional: PDF, PNG, JPEG or TIFF document
OpenAI Decisions	API key; model (gpt-6-luna) Optional: organization ID, API URL (Azure OpenAI), image
OpenAI-compatible chat	API URL, model name; also returns a one-line rationale Optional: API key, image
Self-hosted	Server URL, model name Optional: key, image for vision servers

Box questions (Jev 1.14)

In Ask Questions, `{"button": {"type": "box", "instructions": "the Submit button in `image`"}}` returns button (a dictionary of top, bottom, left and right) plus `button_top` ... `button_right` and `button_confidence`. A box always comes back, so pair it with a yes/no question about the same thing and ignore the box when that answer is no.

Ask Yes/No

Give the text, the yes/no question and, optionally, what yes and no mean in your context. Set Answer yes at or above probability % (default 50). The result's answer is a Boolean and probability is P(true).

Choose Category

List the categories one per line as name | description (the description is optional). The result's answer is the chosen category, probabilities is a dictionary of every category's probability and ranked is a list, most likely first.

Rate on Scale

List the levels one per line, lowest first. The result's answer is the expected level (for example 1.43 on a 0-3 scale), level is the most likely level number and `level_label` its text.

Ask Questions

Give the questions as JSON in the Jev format: `{"name": {"type": "choice" | "score" | "noul", "instructions": "...", "criteria": ...}}`. Each answer comes back under its name (for example `$result{urgency}$`), with the other fields under `name_field` keys (for example `$result{urgency_confidence}$`).

Review threshold

Every decision action has Flag for review below confidence %. When an answer's confidence is below it, `needs_review` is true.

Load Model and Release Model

Load Model loads the model ahead of the first question; Release Model frees its memory. Both are optional: the first question loads the model automatically.

3.1.3 Files included in this submission

- **AdvancedTypedDecisions-2.8.4-lite.jar** - the package; upload it under Manage > Packages.
- **AdvancedTypedDecisionsDemo_windows** - the demo bot, exercising every action with sample data.
- **README-AdvancedTypedDecisions-2.8.4-lite.pdf** - this document.

3.1.4 Actions

Ask Yes/No

Ask a yes/no question about text. Returns a calibrated probability and a true/false answer.

Inputs: Text to evaluate; Yes/no question; Response key; What 'yes' means; What 'no' means; Answer yes at or above probability %; Flag for review below confidence %; Engine; Connection timeout (seconds); Response timeout (seconds)

Returns: result {answer (Boolean), probability, confidence, `needs_review`, ...} to a dictionary variable.

Choose Category

Pick one category for text, with a probability for every category.

Inputs: Text to evaluate; Question; Response key; Categories; Or: categories as text, one per line (name | description); Flag for review below confidence %; Engine; Connection timeout (seconds); Response timeout (seconds)

Returns: result {answer, probability, probabilities, ranked, confidence, `needs_review`, ...} to a dictionary variable.

Rate on Scale

Rate text on an ordered scale. Returns the expected level and the most likely level.

Inputs: Text to evaluate; Question; Response key; Levels, lowest first; Or: levels as text, one per line, lowest first; Scale minimum; Scale maximum; Flag for review below confidence %; Engine; Connection timeout (seconds); Response timeout (seconds)

Returns: result {answer (expected level), level, `level_label`, probabilities, confidence, `needs_review`, ...} to a dictionary variable.

Ask Questions

Ask several typed questions (choice, score, yes/no) about the same text in ONE model pass. Takes the TypeSafe Jev / Laya questions JSON.

Inputs: Text to evaluate; Questions (JSON); Answer yes at or above probability %; Flag for review below confidence %; Engine; Connection timeout (seconds); Response timeout (seconds)

Returns: result {<name>, <name>_confidence, <name>_needs_review, ..., json} to a dictionary variable.

Download Model

Download a built-in model from its OFFICIAL publishers on Hugging Face (Convai for General, Specialist and Multilingual; Qwen + StrandsAgents for Strands Decider), at a pinned revision, and verify it. Run once per machine. Offline machines: download the same official file elsewhere and copy it into the models folder.

Inputs: Model; Models folder

Returns: result {model, path, bytes, `already_installed`, source} to a dictionary variable.

Load Model

Load the Laya model now (first run on a machine also unpacks it, ~10-30 s) so the first question is fast.

Inputs: Model

Returns: model info {model, `already_loaded`, `load_ms`, `cache_dir`, ...} to a dictionary variable.

Release Model

Free the memory of every loaded built-in model (~2 GB each). The next question loads its model again.

Inputs: none

Returns: true if a model was loaded to a boolean variable.

For Credential Vault

Only the cloud engines take credentials; the default Built-in engine and the demo bot need none. Create the credentials you need in any locker (the table uses one named Advanced Typed Decisions), give the bot runner's role Consumer access to the locker, and pick them in the action's key fields. Each key is sent only to its own provider's host.

Locker Name	Credentials Name	Attribute Name	Value
Advanced Typed Decisions	TypeSafe Jev	apiKey	TypeSafe API key (TypeSafe Jev engine)
Advanced Typed Decisions	Cloudflare Workers AI	accountId	32-character Cloudflare account ID (Clef engine)
Advanced Typed Decisions	Cloudflare Workers AI	apiToken	Cloudflare API token, Workers AI template (Clef engine)
Advanced Typed Decisions	ScaleDown	apiKey	ScaleDown API key (ScaleDown CDM engine)
Advanced Typed Decisions	OpenAI	apiKey	OpenAI API key (OpenAI Decisions engine)

For configuring the bot

INPUT VARIABLES				
Variable Name	**Type**	**Mandatory**	**Purpose**	**Example Input**
Text to evaluate	String	Yes	The text the question is about; plain text or a JSON record.	Subject: Refund not received...
Question	String	Yes	The yes/no, category or scale question.	How urgent is this ticket?
Categories / Levels	String	Yes (list or text)	A list with one row per category (name + description) or level (lowest first); or the same as text, one per line, to pass a variable.	billing | payments, refunds
Questions (JSON)	String	Yes (Ask Questions)	Several typed questions in Jev format.	`{"urgent": {"type": "noul", "instructions": "Is this urgent?"}}`
Engine	String (dropdown)	Yes (default Built-in)	Built-in, TypeSafe Jev, Cloudflare Workers AI, ScaleDown CDM, OpenAI Decisions, OpenAI-compatible chat model or Self-hosted.	Built-in
API key / token / account ID	Credential	Per engine	From the Credential Vault, for the cloud engines.	Cloudflare Workers AI > apiToken
Image / audio-video / document file	File	No	Image (Jev, Clef, OpenAI, chat model, Self-hosted), audio or video (Clef-omni), PDF or scan (ScaleDown).	C:\in\`invoice.png`
Flag for review below confidence %	Number	No	Sets `needs_review` when confidence is below it.	80
Response key	String	No	Names the question in the request and in the json output (default q).	department

INPUT VARIABLES				
Variable Name	**Type**	**Mandatory**	**Purpose**	**Example Input**
Scale minimum / maximum	Number	No	Rate on Scale: map the expected level onto this range as `scaled_score`.	0 / 100
Connection / Response timeout	Number (seconds)	No (20 / 120)	Cloud and self-hosted engines: how long to wait; HTTP 429/503 is retried twice.	120

OUTPUT VARIABLES				
Variable Name	**Type**	**Mandatory**	**Purpose**	**Example Output**
answer	Boolean / String / Number	-	Yes/no answer, chosen category, or expected scale level.	billing
probability	Number	-	P(true), or the probability of the chosen category or level.	0.9342
confidence	Number	-	1 - normalized entropy of the answer distribution (0 to 1).	0.7376
`needs_review`	Boolean	-	True when confidence is below the review threshold.	true
<name>_top ... _right	Number	-	Box questions only: the edges as fractions of the image, from the top-left corner.	0.44

OUTPUT VARIABLES				
Variable Name	**Type**	**Mandatory**	**Purpose**	**Example Output**
json	String	-	The full response in Jev's system-one JSON shape.	`{"model":"laya","answers":{...}}`

Running the sample bot

1. Import the demo bot zip from the submission into the Control Room (Bots > Import bots).
2. Open Bot Store > Advanced Typed Decisions-KhaledMostafaMe > AdvancedTypedDecisionsDemo_windows and run it on a Windows Bot Agent. It needs no configuration.
3. It loads the model, runs Ask Yes/No, Choose Category, Rate on Scale and Ask Questions on a sample support ticket, then releases the model.
4. Open `typesafe-demo-run.txt` in the Bot Agent's log folder (Advanced Typed Decisions-KhaledMostafaMe) to see every result.

4. Support & FAQs

4.1 Support

This is a free community package from KhaledMostafaMe.

- **Questions and issues**: the Automation Anywhere Pathfinder Community.
- **Package documentation**: this Read Me.
- Do not paste customer data into a public channel. Reduce the problem to a small synthetic example, and redact anything sensitive.

4.2 FAQs

4.2.1 Does any text leave the Bot Agent?

Not with the default Built-in engine: the models, tokenizer and runtime run in the Bot Agent's own process and no action makes a network call. Text (and any image, audio, video or document you attach) is sent out only when you set an action's Engine to a cloud or self-hosted engine: TypeSafe Jev (api.typesafe.ai), Cloudflare Workers AI (api.cloudflare.com), ScaleDown (api.scaledown.xyz), OpenAI Decisions (api.openai.com or your Azure OpenAI URL), the OpenAI-compatible chat API you configure, or your own server.

4.2.2 Where are the models? The package is only 24 MB.

This Lite edition carries no model weights. Add the Download Model action once (the demo bot does): it fetches the model from its publisher, Convai, on Hugging Face at a pinned revision, checks its SHA-256 and does nothing on later runs. On machines without internet, copy the file into the models folder the error message names, or use the Full edition (about 2.0 GB, all three Laya models inside) from the GitHub release.

4.2.3 Which languages are supported?

General and Specialist: English, first 512 and 1,024 tokens of the text. Multilingual: Convai's laya-multilingual model, 100+ languages (Arabic, French, Hindi, Chinese and more), first 1,024 tokens. Write the question in the same language as the text: on an Arabic ticket the English question scored 0.24 and the Arabic one 0.94. On English text use General - Multilingual scored lower there.

4.2.4 How many categories can a question have?

Up to 64 are accepted; the model works best with fewer than about 20. Long category lists are shortened to fit the model's option space.

4.2.5 What does confidence mean?

1 minus the normalized entropy of the answer's probability distribution: 1 means all probability on one answer, 0 means spread evenly. Use it with the review threshold to route unclear answers to a person.

4.2.6 The action says there is not enough memory.

The model needs about 2.3 GB of free RAM to load. Close other applications, run Release Model when a bot is done with questions, or use an 8 GB agent.

4.2.7 Which model should I use?

General for English text in general; Specialist for support tickets, invoices, security alerts and agent logs; Multilingual for other languages (ask the question in the text's language); Strands Decider 2B when English accuracy matters most and the agent has ~8 GB free RAM. Strands is never inside the package: run Download Model once per machine (4.6 GB from Qwen and StrandsAgents), or copy the four official files into the models folder offline.

4.2.8 Which engine should I use?

Measured on the same 200 typed decisions: built-in Specialist 0.735 (free, on the agent, about 7 s per 5-question case), TypeSafe Jev 0.715 (about 0.3 s, $0.042 per million tokens), Cloudflare Clef 0.710 (about 0.5 s, $0.24 per million; Clef-flash 0.665, Clef-omni 0.635), ScaleDown 0.640 (about 0.4 s, $0.05 per million) and built-in Strands Decider 0.56. Use Built-in when text must stay on the machine, Jev or Clef for speed, Clef-omni for audio or video, and ScaleDown for PDFs and scans. OpenAI Decisions returns OpenAI's own calibrated probabilities. The OpenAI-compatible chat engine adds a one-line rationale per answer, but its probabilities are the model's own estimate, not calibrated - use it when you must stay on your organisation's approved LLM or need the explanation, not where thresholds drive routing.

For questions relating to the Control Room, see the Automation 360 FAQs.

Appendix A: Record of Changes

No.	Version Number	Date of Change	Author	Notes
1	2.8.4	11/11/2026	KhaledMostafaMe	First public release. Seven actions: Ask Yes/No, Choose Category, Rate on Scale, Ask Questions, Download Model, Load Model and Release Model. Built-in engine (Laya General, Specialist and Multilingual, and Strands Decider, downloaded once from their publishers and checked by SHA-256), plus optional TypeSafe Jev, Cloudflare Workers AI, ScaleDown CDM, OpenAI Decisions, OpenAI-compatible chat and self-hosted engines.

Appendix B: References

No.	Topic	Reference Link
1	Laya model (Convai Innovations, Apache 2.0)	https://huggingface.co/convaiinnovations/laya
2	receptron/laya (reference implementation, MIT)	https://github.com/receptron/laya
3	ONNX Runtime	https://onnxruntime.ai/
4	TypeSafe Jev API (models, images, box)	https://docs.typesafe.ai/models
5	Cloudflare Workers AI - Clef models	https://developers.cloudflare.com/workers-ai/models/clef/
6	ScaleDown Decisions API	https://docs.scaledown.ai/api-reference/decisions-overview
7	Strands Decider (strands-labs, Apache 2.0)	https://github.com/strands-labs/strands-decider
8	OpenAI Decisions API	https://platform.openai.com/docs/guides/decisions