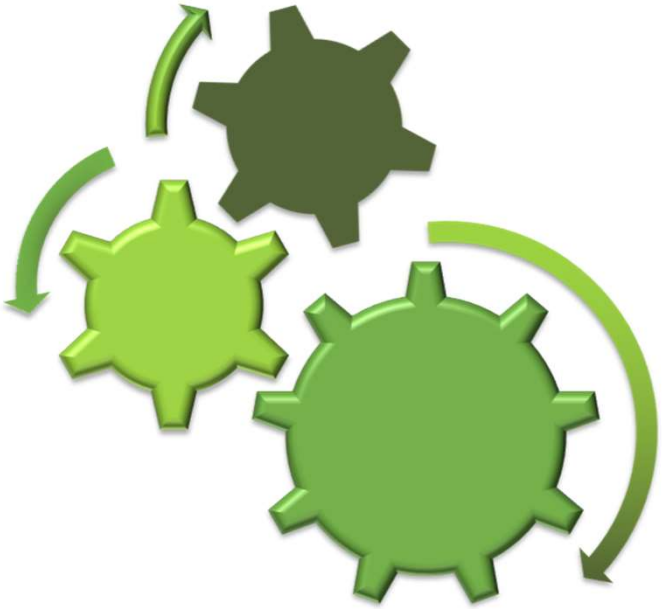


# OpenAPI Initiative

## API Descriptions make APIs usable

April 15, 2024

# Code of Conduct

The OpenAPI Initiative (OAI) is an open source Linux Foundation project and home of the OpenAPI Specification (OAS) released under the Apache 2.0 license. As contributors, maintainers, and participants in this project, we want to foster an open and welcoming environment. We pledge to make participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

## Our Standards

Examples of behaviors that contribute to

creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members
- Assuming the best intent from others

# Agenda

- **Introduction to OpenAPI and our mission.**
- **Current activities and focus areas.**
- **Summary of the soon to be published workflow specification.**
- **A peak into future releases.**

# What is OpenAPI Initiative?

- 40 member organizations across industries: e-commerce, tech, government, cloud, developer tools
- Promotes standards for describing APIs
- A vendor neutral description format that unites an industry


OpenAPI Specification (formerly known as Swagger)

# A Brief History of OAI & OpenAPI

**Started out in 2011 and was called *Swagger* for quite a while**

- **Competing approaches like *RAML* and *API Blueprint* exist**

***Swagger* was renamed to OpenAPI at the beginning of 2016**

- **Donated to the *OpenAPI Initiative* *(OAI)* (part of the *Linux Foundation*)**
- ***Swagger* now refers to a specific tool offered by *SmartBear***

***OpenAPI* has been the official name since 2016**

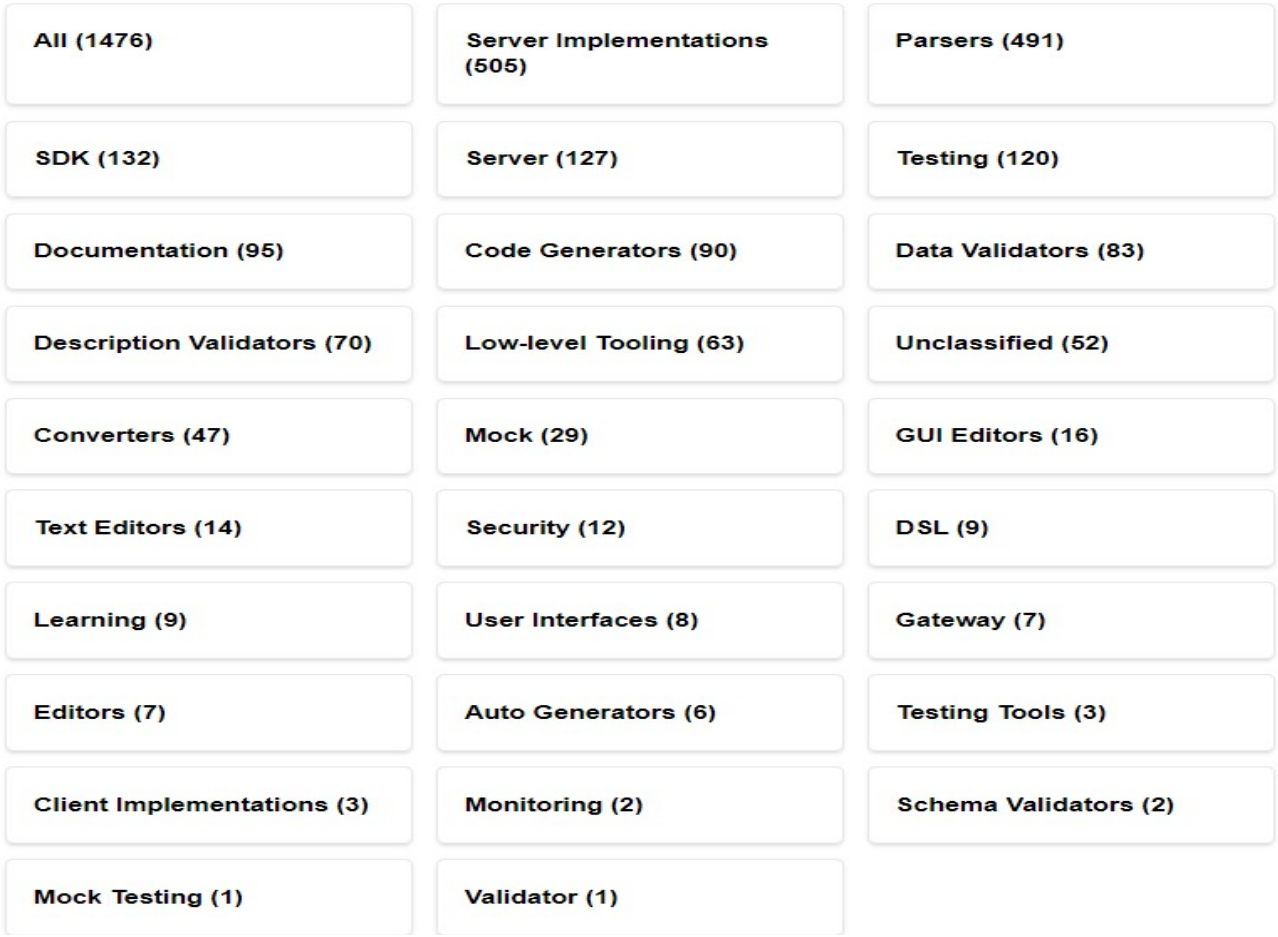
- ***OpenAPI 2.0* was the name after the rename (it's the same as *Swagger 2.0*)**
- ***OpenAPI 3.0* was released in July 2017**
- ***OpenAPI 3.1* was released in February 2021**

# What do we mean by a description document?

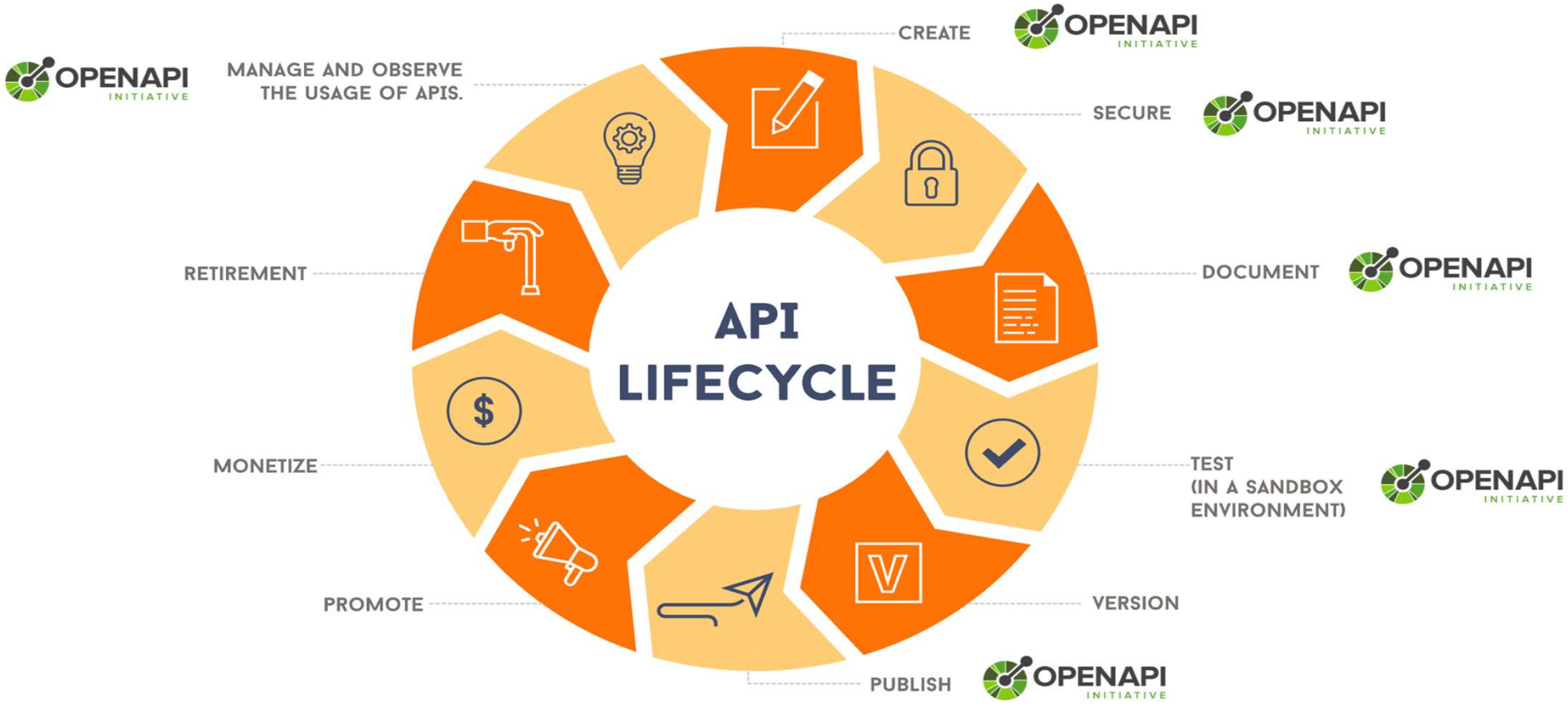
- The OpenAPI Specification (OAS) allows the description of a remote API accessible through HTTP or HTTP-like protocols
  - This description, which may be stored as one or more documents (such as local files or HTTP-accessible network resources), is called an OpenAPI Description (OAD)
  - An API description file (sometimes called contract) is a machine-readable specification of an API
    - It should strive to be as complete, and fully-detailed as possible
    - The more unambiguous it is, the more useful it becomes
- Its main advantage over documentation which only humans can read is that it enables automated processing
  - Documentation for humans including the list of available methods and their details can be easily generated from the API description file
  - Tools can use the API description to generate boilerplate code

# OpenAPI Tools

- You may be using the OpenAPI spec and not know it
- A collection of open-source and commercial tools for creating your APIs with OpenAPI
- Sourced from and published for the community
- You can add your tool to the list on the tool website

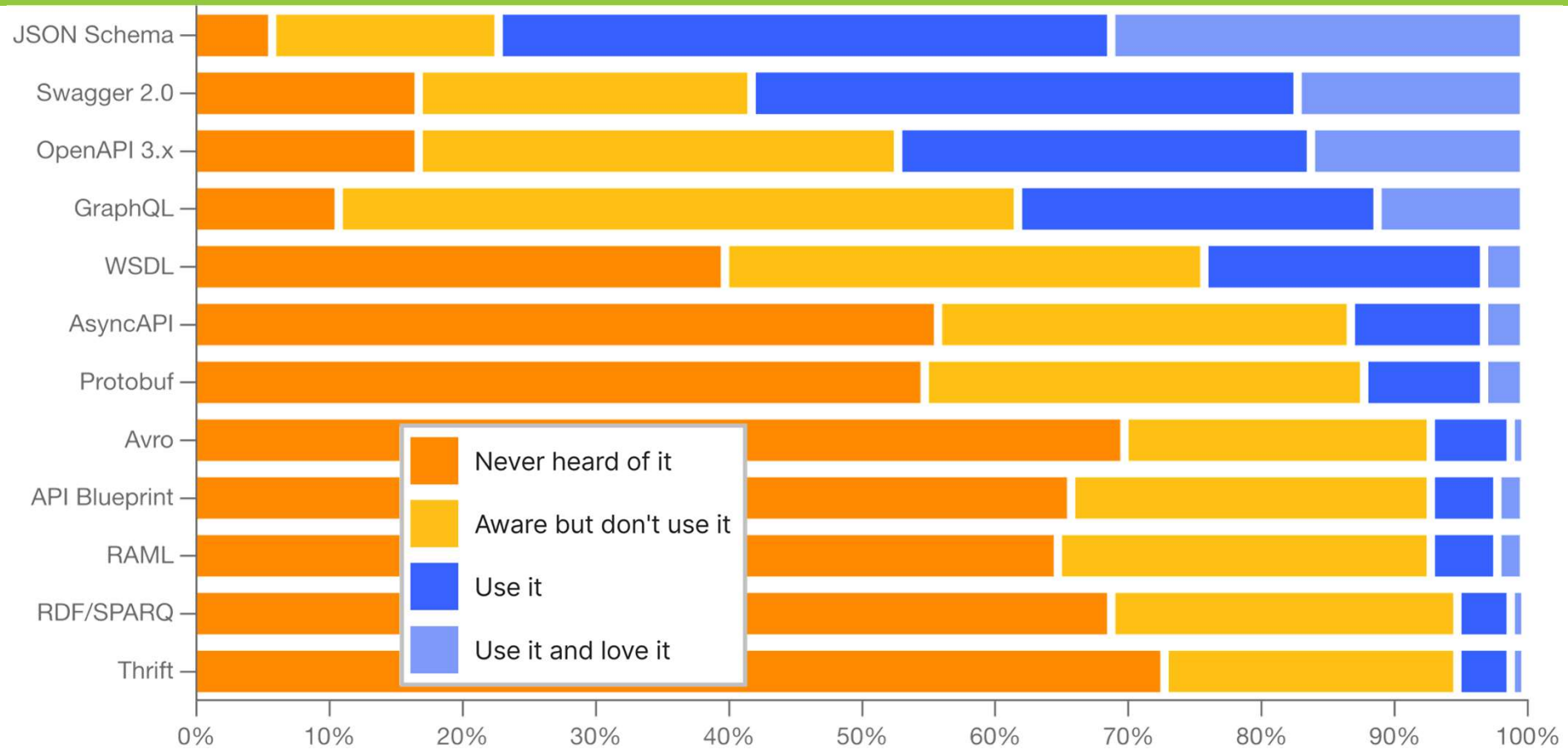


# The (Open)API Lifecycle




Source: WSO2

# API Technologies




Source: Postman 2023 State of API report

# Why does it matter? Travel Industry Example

- Travel APIs enable app developers to integrate directly with databases and systems containing travel-related data, such as hotel rooms, flights, rental cars, and more
  - ~450 airlines, dozens of major hotel (500k+ hotels in 200+ countries), cruise, rail and car companies, innumerable tour operators, restaurants, ground transportation, vacation rental, events, and local tourist boards
  - Online booking platforms utilize APIs for the lookup and reservation process. Annual travel API transactions are predicted to reach 3.6 billion by 2024, up from 1.5 billion in 2019 according to Phocuswright
- Non-standard APIs often suffer from inadequate or poorly documented specifications leading to higher labor costs
  - Lack of clear documentation forces them to guess endpoint URLs, payload structures, and available methods
    - Undocumented features, hidden side effects, authentication chaos, version chaos, security vulnerabilities

# Why does it matter? Travel Industry Example, continued

- Most API productivity tools are focused on the API deployer and not the API consumer
  - Across travel, 1000s of availability and booking APIs that do the same basic functions, each requiring unique code for the API consumer
  - In an industry where content is king, the vast majority of travel content is unreachable due to content acquisition costs
- Established standards ensure uniformity across APIs
  - Developers can rely on consistent patterns for requests, responses, and error handling
  - Consistency enables tooling for API deployment and consumption including code generation
  - Tooling and automation brings down labor costs
- Lower labor costs enables the acquisition of more content and revenue
  - Time to market also matters, **an unconsumed API does not generate revenue**

# How are we organized

- The OpenAPI Initiative (OAI) is an open-source Linux Foundation project
- Business Governance Board provides oversight and is composed of one representative appointed by each OAI Member
- Technical Steering Committee owns the specifications and is open to any developer, end user or subject matter expert that chooses to participate in the activities of OAI, regardless of whether the participant is employed by an OAI Member company
- Special Interest Groups (SIGs) are workgroups focused on a defined task and outcome
  - More on this in a minute
- Outreach committee, focused on community engagement

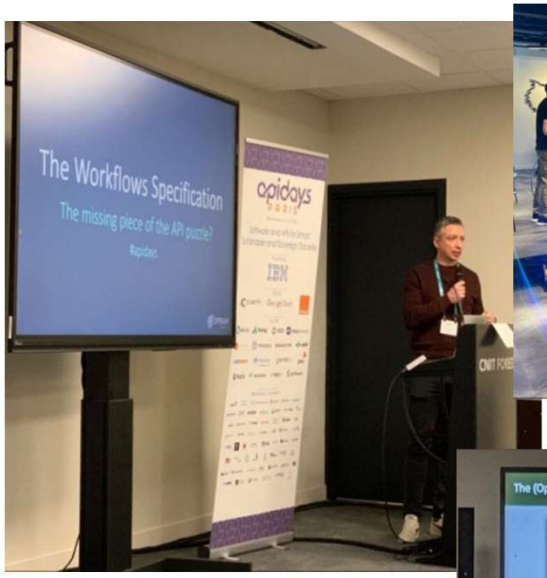
# Outreach Activities

- OpenAPI April Newsletter published April 4th
- Busy event calendar in April
  - OpenAPI Mini summit as part of LF Open-Source Summit week of April 15th
  - OpenAPI track as part of apidays Singapore April 17th-18th
  - OpenAPI track as part of JAX Mainz, April 22nd-26th
  - OpenAPI track as part of apidays New York April 30th – May 1st
- Education program under development
  - Educational material with an open license for individual use or professional offerings
    - Initial chapters at https://github.com/OAI/OAI-Courses

# API Days Series

**Taking the OpenAPI message to a wider audience thru an OpenAPI Track at partner conferences**

- **Thousands of attendees in 6 events.**
- **Lots of good discussions/meetings**
- **Great impressions of LinkedIn posts (> 3000 per post)**
- **Strong community engagement**


# Work Group Highlights

Special Interest Groups

# Special Interest Groups (SIGs)

Technical SIGs establishing and developing specific extensions to OAS

- Overlays (GitHub) (Slack) - Focused on defining overlays
- Security (GitHub) (Slack) - Focused on security
- Workflows (GitHub) (Slack) - Workflows with multiple APIs
- Service Level Agreement (GitHub) - SLA extensions
- Codegen (Slack) - Focused on code generation
- Formats (GitHub) - Formats registry effort.
- Lifecycle (GitHub) (Slack) - Defining the API lifecycle

Industry SIGs focused on the usage of OAS in a specific industry, which may eventually contribute changes, extensions, or overlays to OAS

- Travel (GitHub) (Slack) - Focused on using OAS within the travel industry
- Other industries under discussion

Jump into the discussions by using the links or go to https://github.com/OAI ! Membership not required

# Overlays SIG

- The Overlay specification defines a way of creating documents that contain information to be merged with an OpenAPI description at some later point in time, for the purpose of updating the OpenAPI description with additional information
  - Support multi-language API descriptions by using Overlays to contain language translations
  - Provide configuration information for different deployment environments
  - Allow separation of concerns for metadata such as gateway configuration or SLA information
  - Support a traits like capability for applying a set of configuration data, such as multiple parameters, or multiple headers to a targeted object
  - Provide default responses, or parameters where they are not explicitly provided
  - Apply configuration data globally or based on filter conditions
- Overlays support different views of the same OpenAPI documents

# Compliance Project SIG

- To create an OpenAPI Specification Compliance Parser that can be employed by tool providers
  - An oascomply command-line tool that takes an API description, as a single document or as several documents linked by JSON References ($ref) and indicates whether or not it is syntactically and semantically valid
  - Similar in approach to existing code and schema parsers
- The OpenAPI specification is semantically rich and allows complex relationships across the model where manual validation is not adequate
- Expected to become a baseline for API assessment much like language linting tools

# Travel SIG

- Industry SIGs are focused on the use on the OAI specifications in the context of an industry sector
  - Two-way communication with technical SIGs on industry requirements (inbound) and how the spec can solve real world issues (outbound)
- The Travel SIG will compile and present a collective representation of the API behaviors required by the travel industry documented in the form of workflow use cases
  - Being used by the workflow SIG to pressure test the spec
- The Travel SIG will work with the OpenTravel Alliance to publish reference implementations on the use of the spec to publish common use API sequences

opentravel
enabling the future

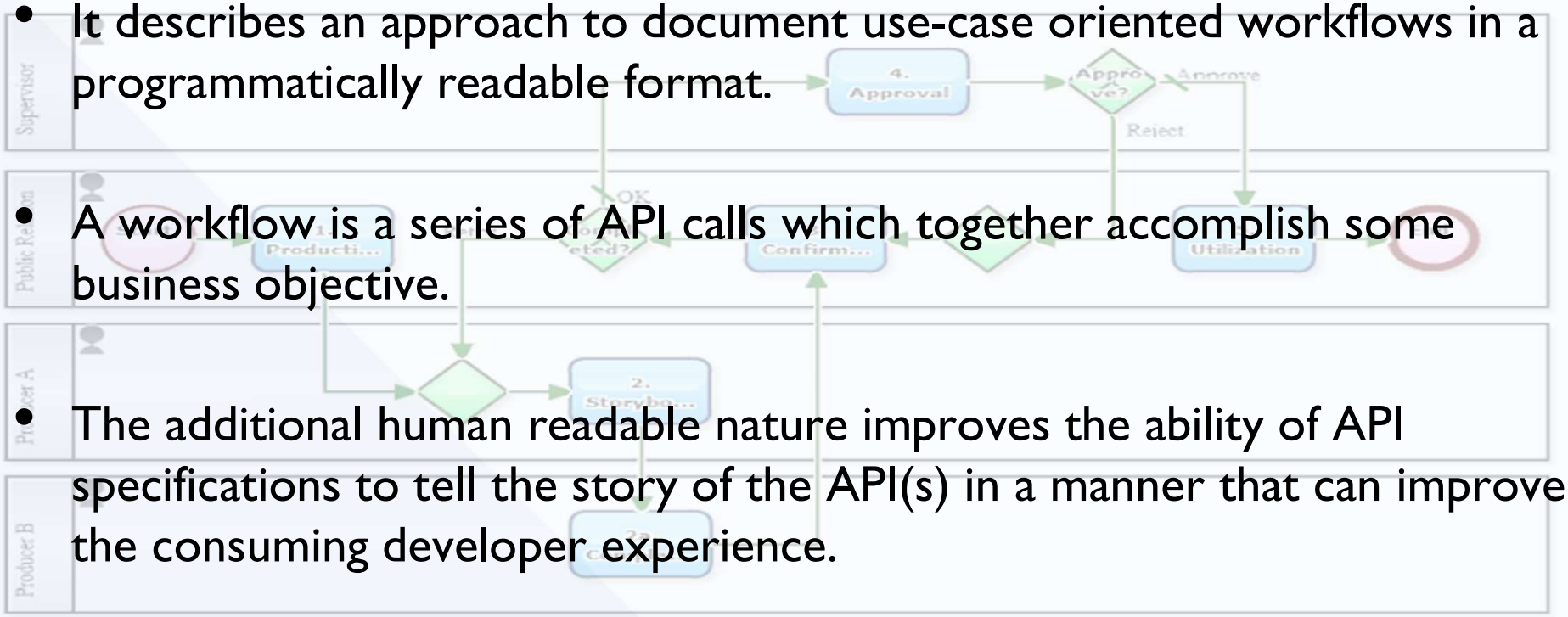
# Workflows

~~Deep~~ Shallow Dive

This just in, the name will be Tapestry

# What is the Workflows Specification?

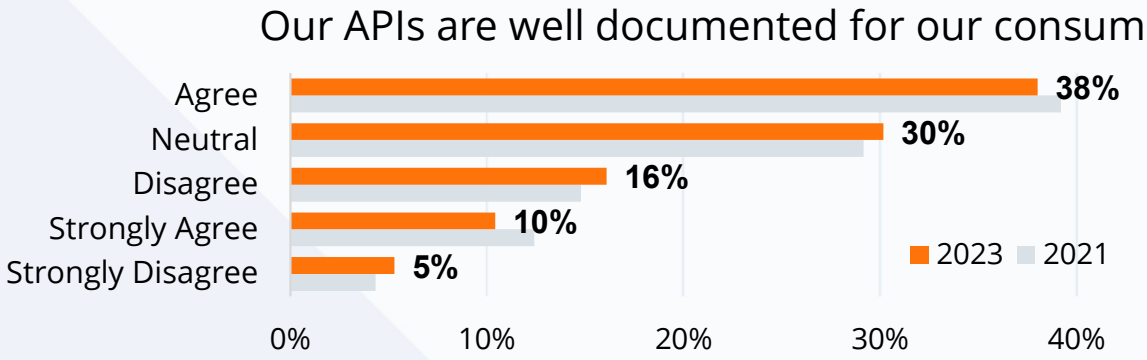
- It describes an approach to document use-case oriented workflows in a programmatically readable format.
- A workflow is a series of API calls which together accomplish some business objective.
- The additional human readable nature improves the ability of API specifications to tell the story of the API(s) in a manner that can improve the consuming developer experience.



# Deeper dive on Workflows

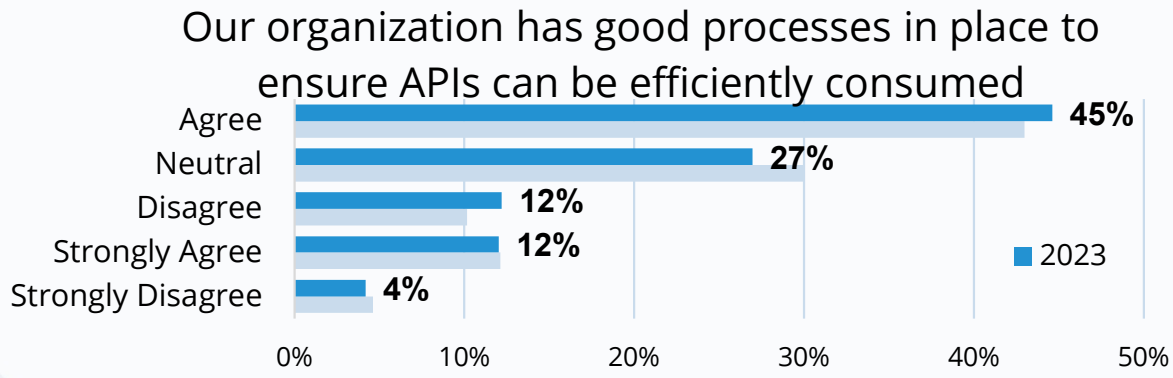
- APIs are typically accompanied by a reference guide; a piece of literature explaining to a developer how to use the API
  - In travel APIs, only if you are very lucky
- Unfortunately, everybody working on software development is familiar with one or more of the following problems:
  - Unclear documentation, leading to mistakes due to interpretation differences.
  - Incomplete or non-existing documentation.
  - Outdated information.
  - Information in a language the reader does not understand.
  - In these cases, to find the information they require developers might have to read source code (if available), debug programs or analyze network traffic, which are gigantic **time sinks**

# Alignment Gap




- Only 48% are confident their APIs are well documented and explain value for consumers

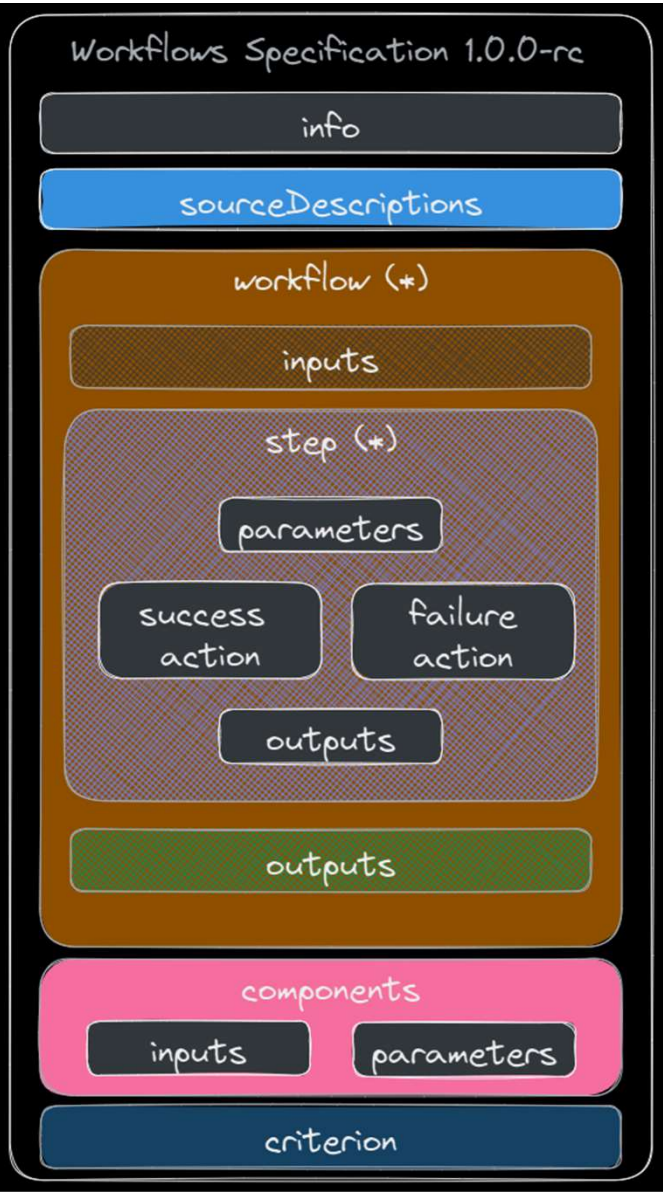
- ~43% unsure their processes will deliver appropriate developer experience (DX)




Source: SmartBear – State of Software Quality | API 2023

# Use cases

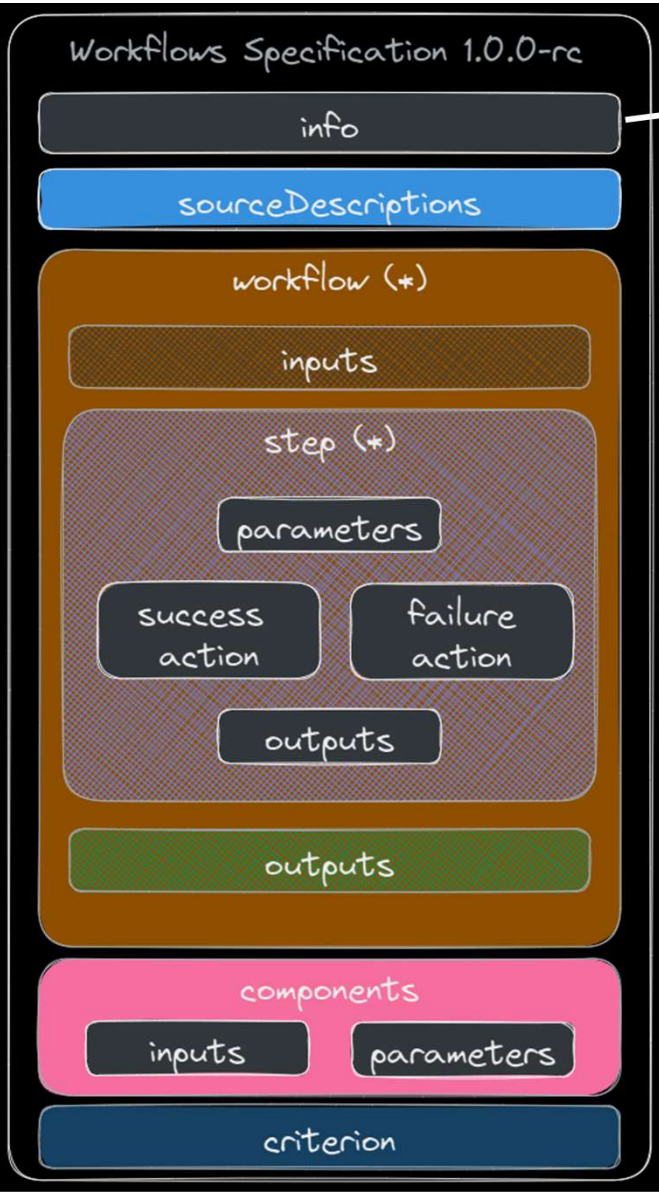
- Deterministic recipes for the use of APIs
  - Make sense of large unwieldy API description
  - Bridge the *gap* where business flows span more than one API description
  - Use case testing potential
- Living API documentation and improved consumer developer experience (DX)
- Assertable business value
- Targeted code generation tooling for APIs driven by use cases
- Improved regulatory checks via assertable workflows
- The AI potential for value orientated interaction with APIs




# Workflows Specification Structure

SMARTBEAR

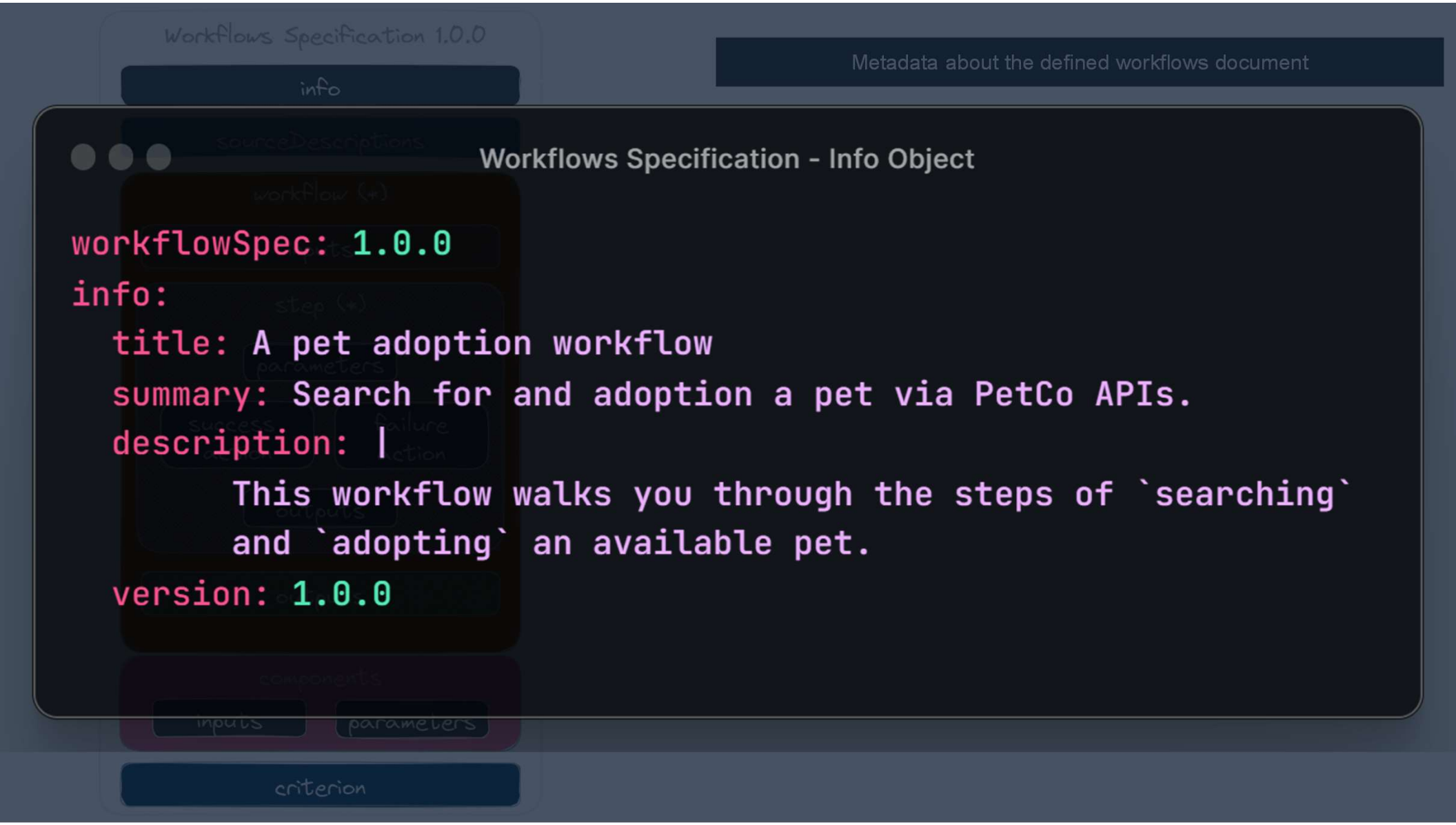
OPENAPI INITIATIVE




Metadata about the defined workflows document

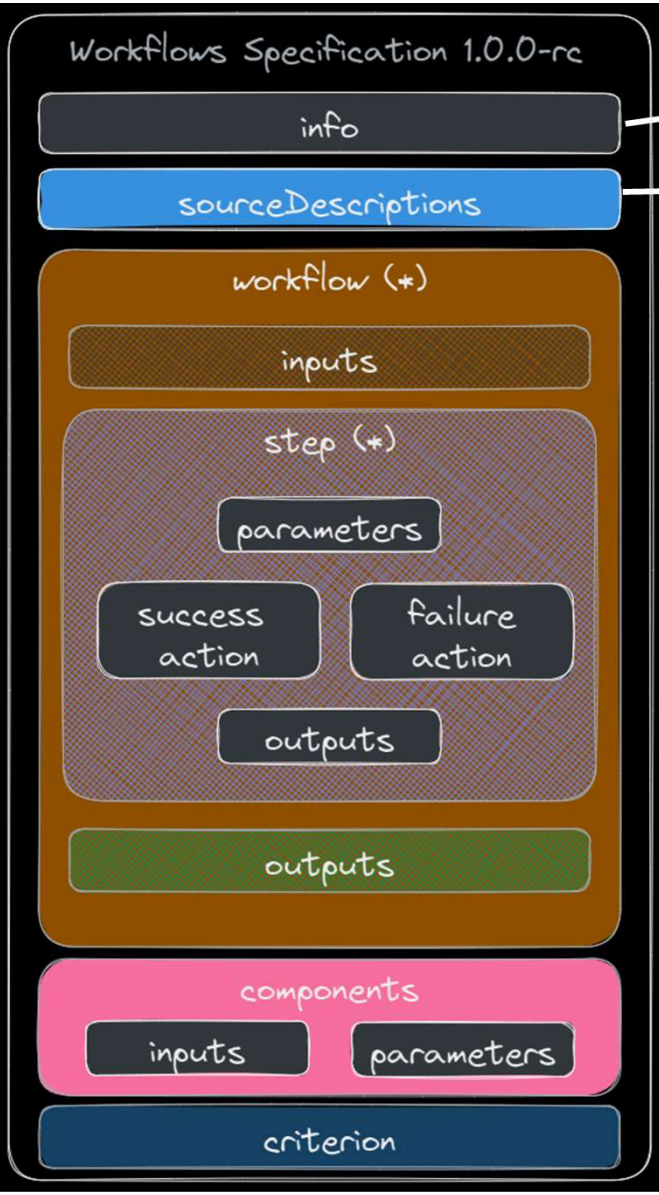
SMARTBEAR

OPENAPI INITIATIVE




Workflows Specification - Info Object

```yaml
workflowSpec: 1.0.0
info:
  title: A pet adoption workflow
  summary: Search for and adoption a pet via PetCo APIs.
  description: |
      This workflow walks you through the steps of `searching`
      and `adopting` an available pet.
  version: 1.0.0
```

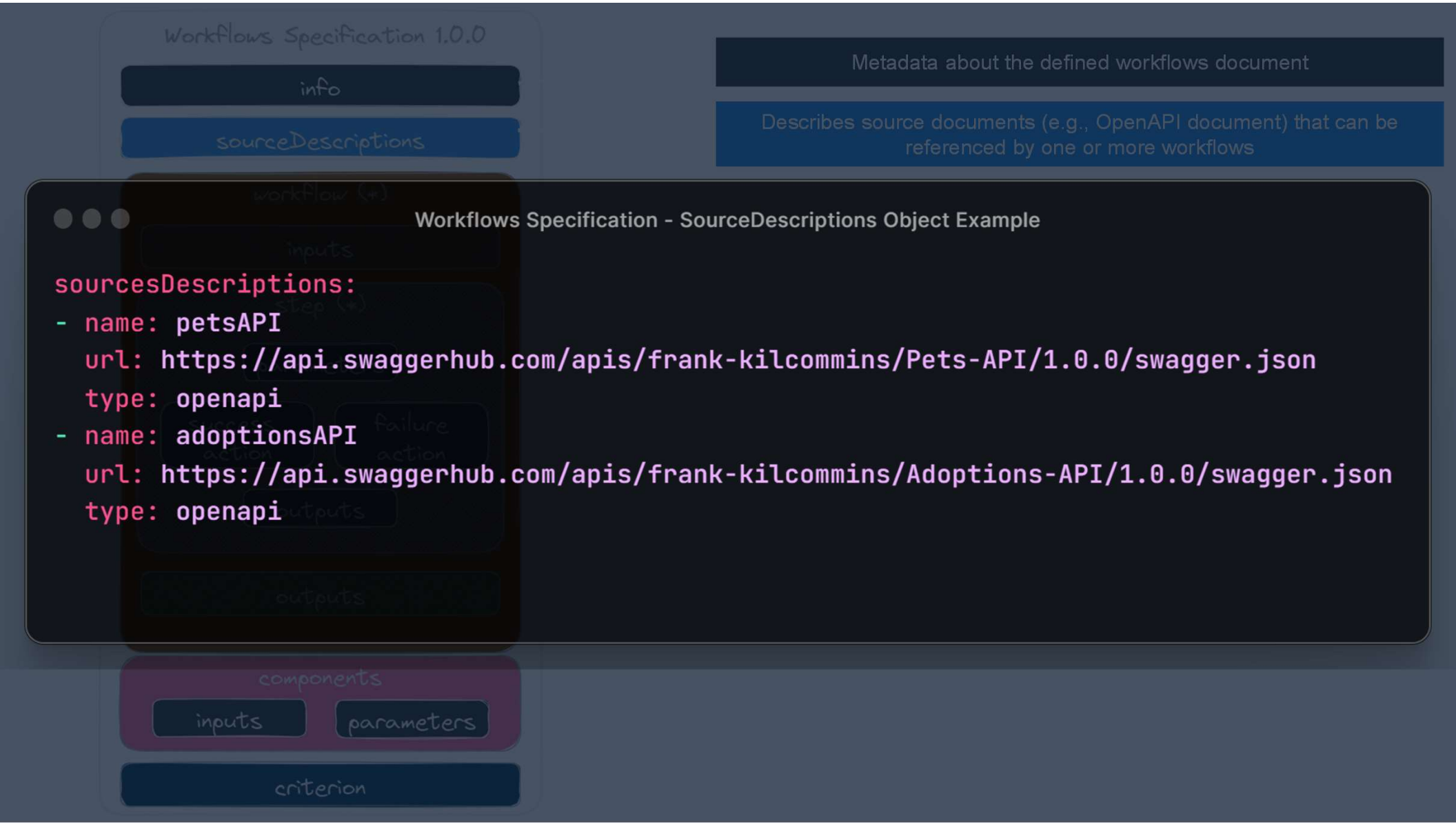


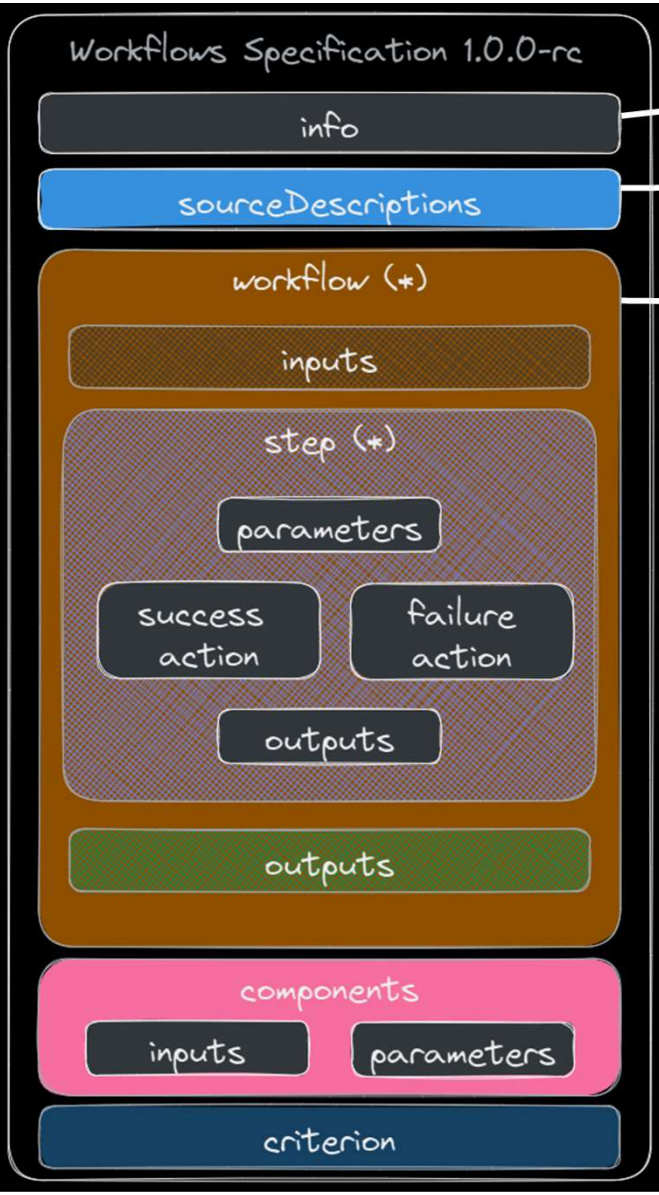

Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification 1.0.0
info
sourceDescriptions
components
inputs
parameters
criterion
Metadata about the defined workflows document
Describes source documents (e.g., OpenAPI document) that can be referenced by one or more workflows
Workflows Specification - SourceDescriptions Object Example
sourcesDescriptions:
- name: petsAPI
url: https://api.swaggerhub.com/apis/frank-kilcommins/Pets-API/1.0.0/swagger.json
type: openapi
- name: adoptionsAPI
url: https://api.swaggerhub.com/apis/frank-kilcommins/Adoptions-API/1.0.0/swagger.json
type: openapi




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

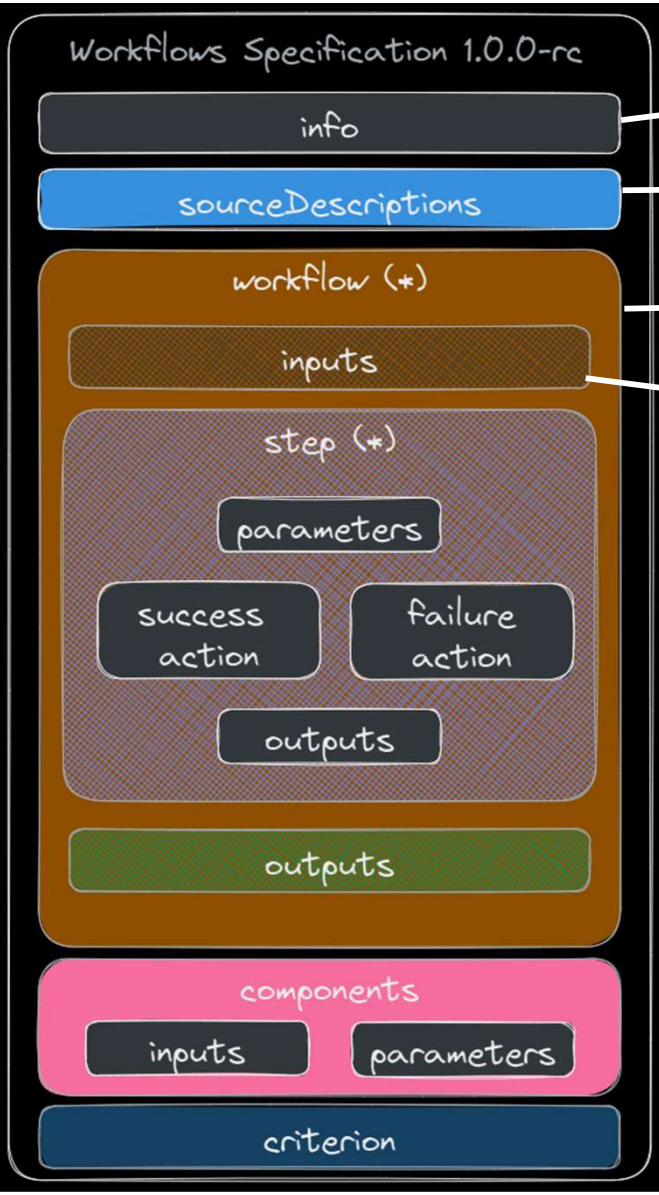
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

SMARTBEAR

OPENAPI INITIATIVE

Workflows Specification - Workflows Object

```yaml
workflows:
- workflowId: FindAndAdoptPet
  summary: Search for appropriate available pet and adopt it
  description: |
    This workflow lays out the steps and API calls
    needed to search for an adopt a Pet
  inputs: {}
  steps: []
  outputs: []
```




Metadata about the defined workflows document

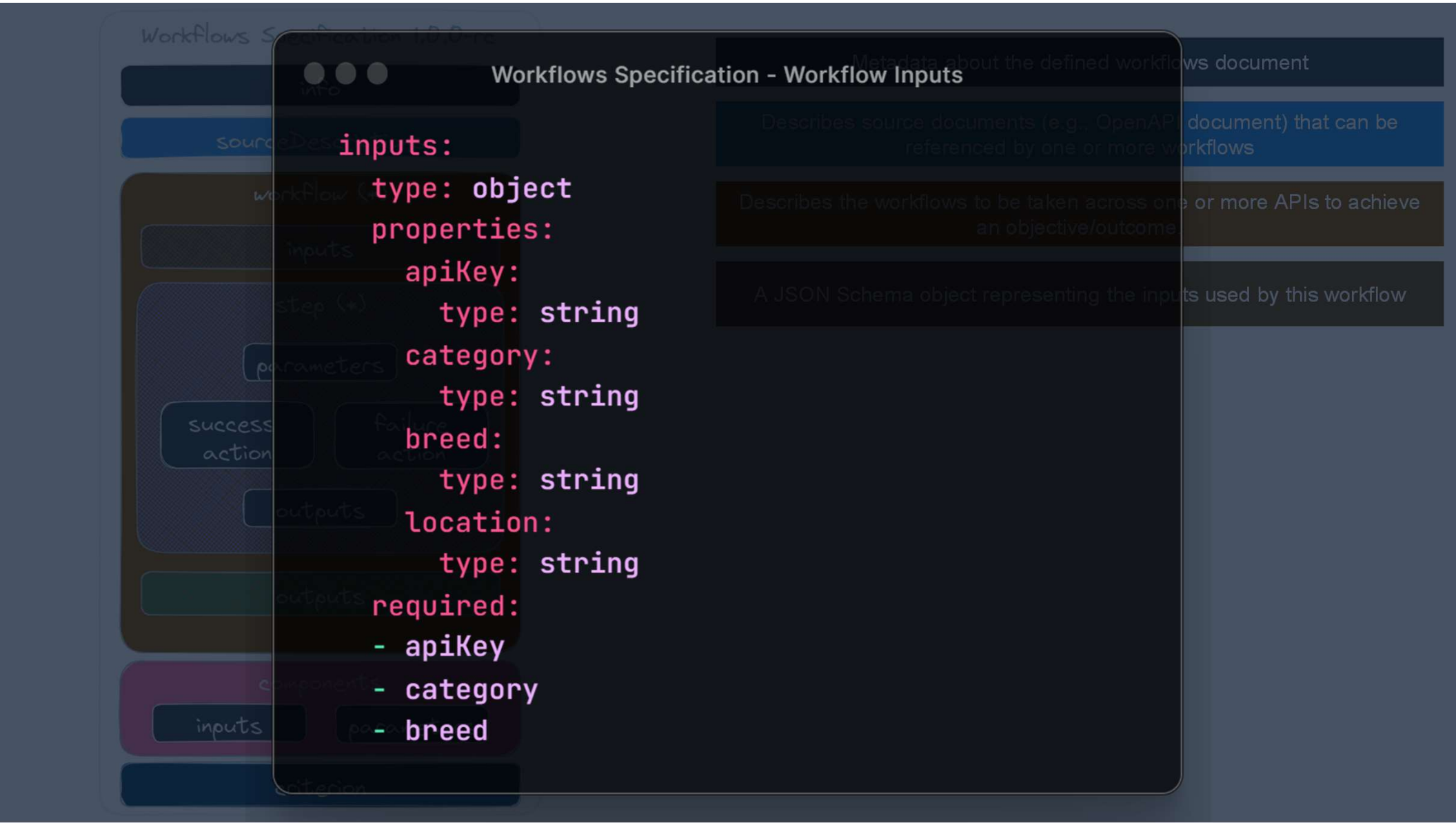
Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

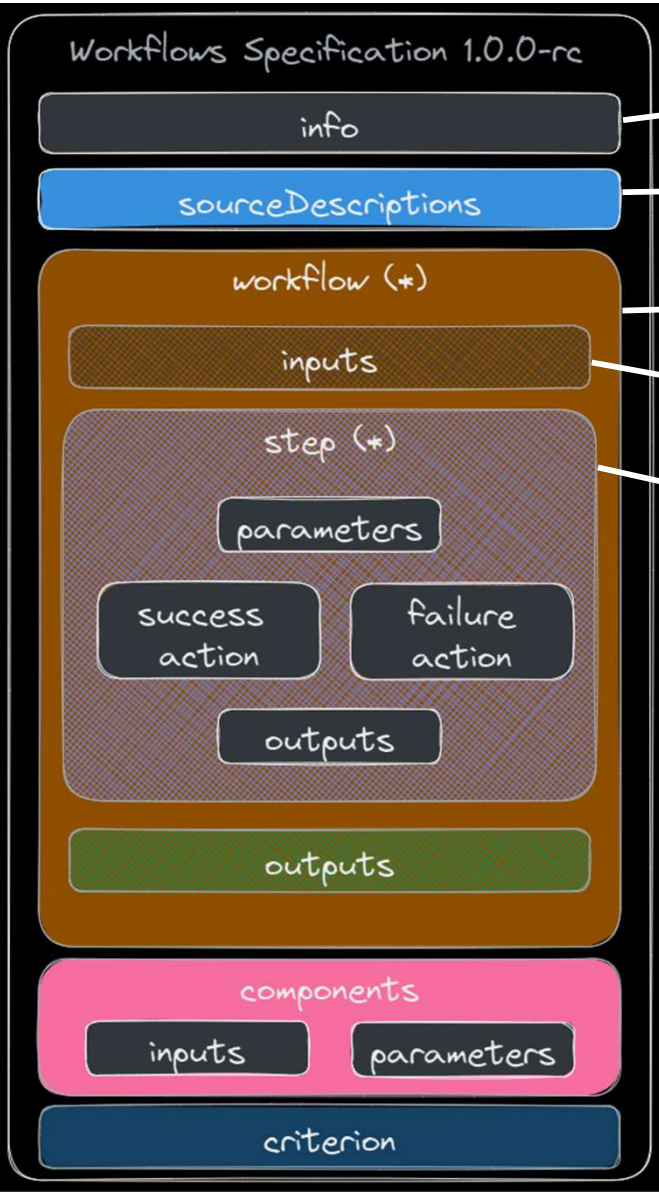
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

A JSON Schema object representing the inputs used by this workflow

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification - Workflow Inputs
inputs:
type: object
properties:
apiKey:
type: string
category:
type: string
breed:
type: string
location:
type: string
required:
- apiKey
- category
- breed
Metadata about the defined workflows document
Describes source documents (e.g., OpenAPI document) that can be referenced by one or more workflows
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome
A JSON Schema object representing the inputs used by this workflow




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

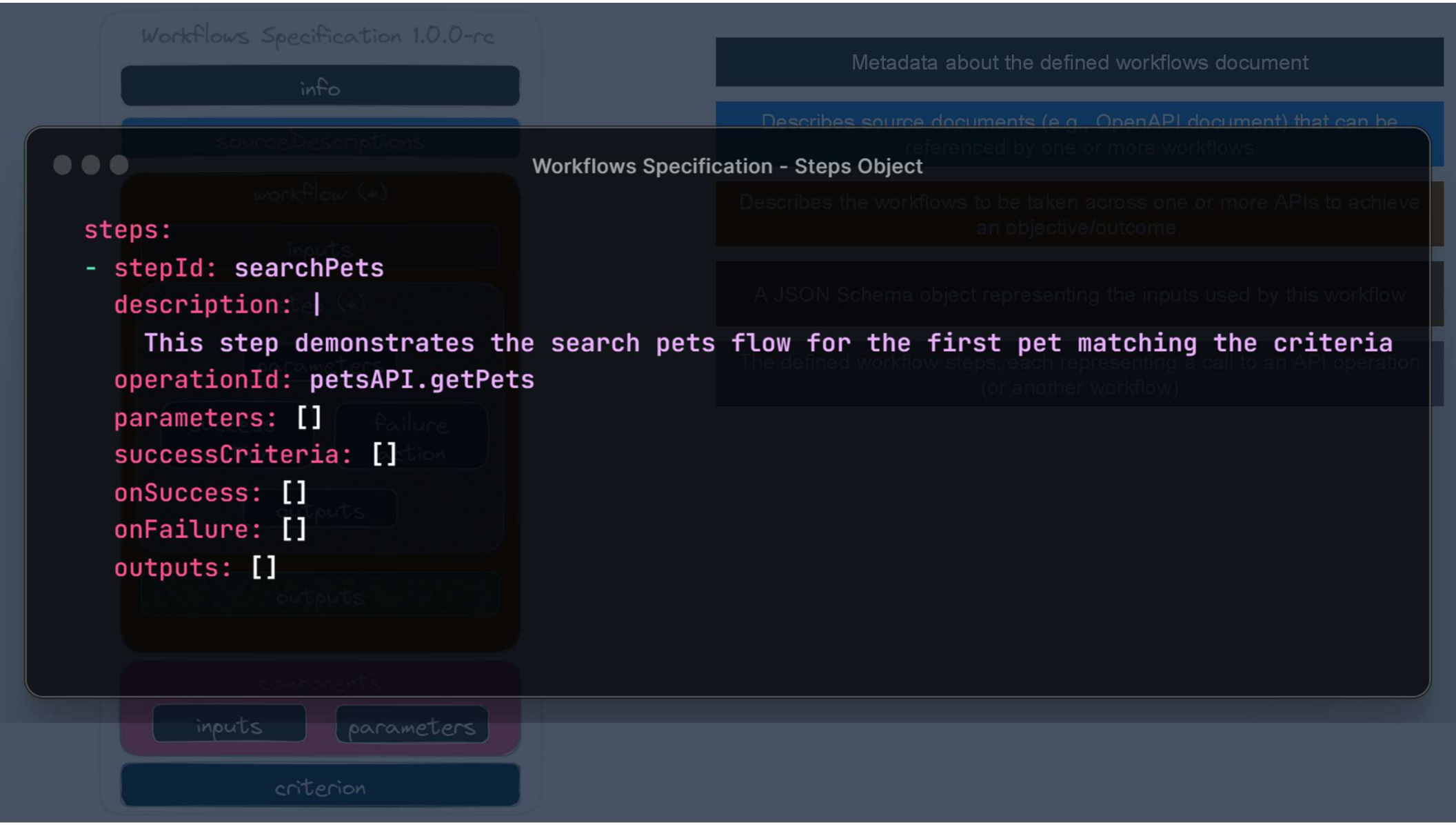
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

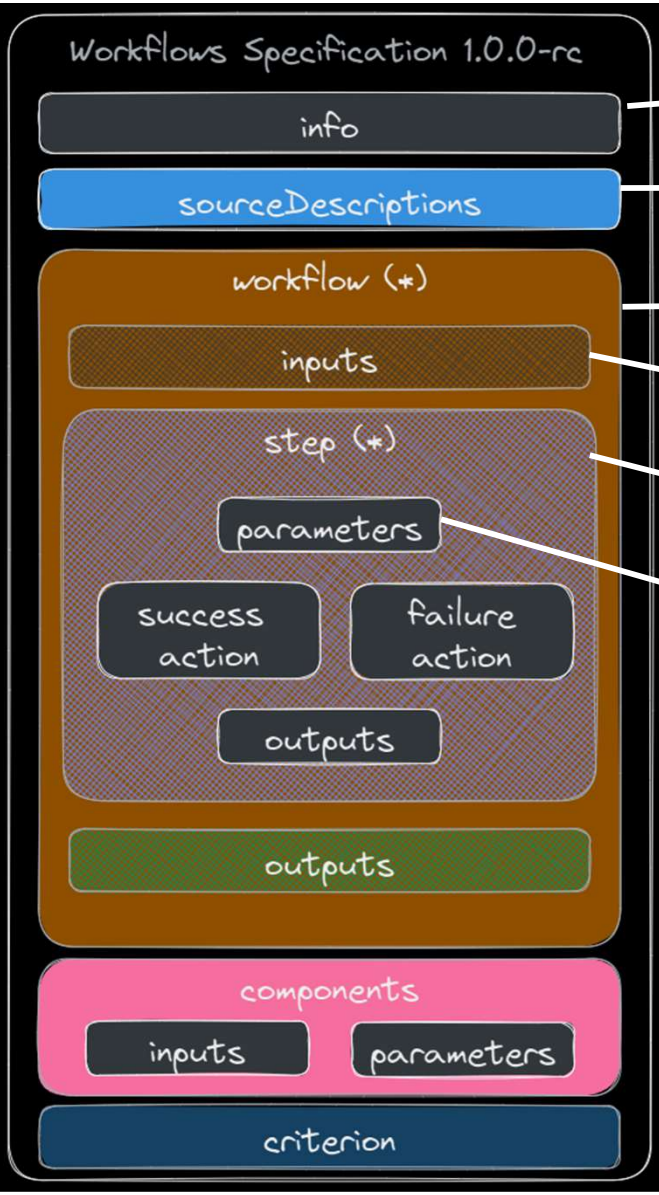
A JSON Schema object representing the inputs used by this workflow

The defined workflow steps, each representing a call to an API operation (or another workflow)

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification 1.0.0-rc
info
sourceDescriptions
workflow (+)
inputs
parameters
failure
outputs
outputs
components
inputs
parameters
criterion
Metadata about the defined workflows document
Describes source documents (e.g., OpenAPI document) that can be referenced by one or more workflows.
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.
A JSON Schema object representing the inputs used by this workflow
The defined workflow steps, each representing a call to an API operation (or another workflow)
Workflows Specification - Steps Object
steps:
- stepId: searchPets
description: |
This step demonstrates the search pets flow for the first pet matching the criteria
operationId: petsAPI.getPets
parameters: []
successCriteria: []
onSuccess: []
onFailure: []
outputs: []




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

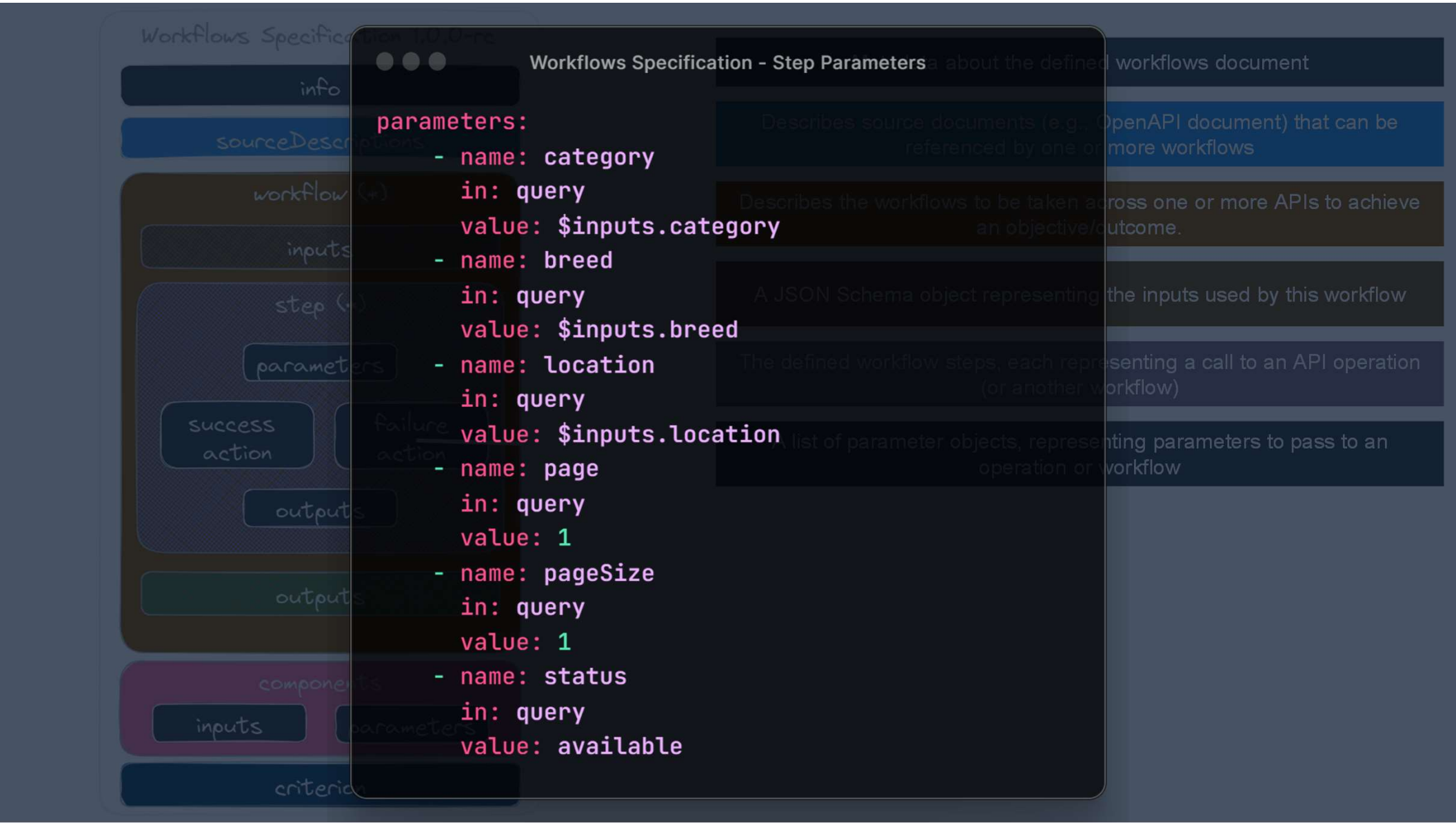
A JSON Schema object representing the inputs used by this workflow

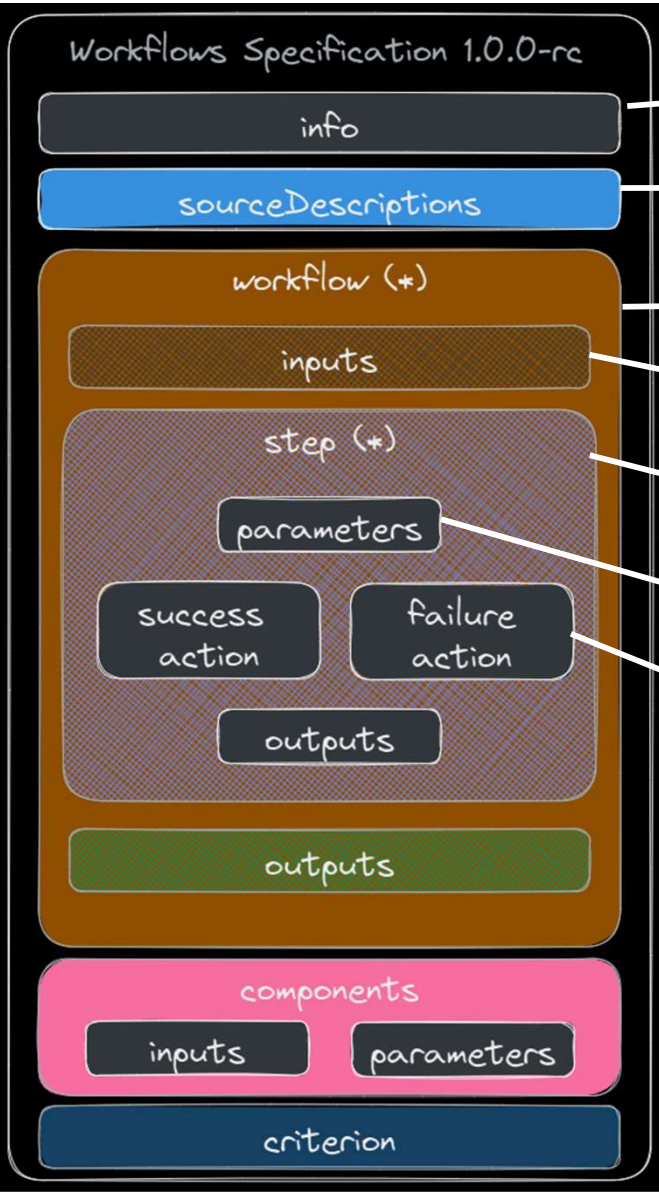
The defined workflow steps, each representing a call to an API operation (or another workflow)

A list of parameter objects, representing parameters to pass to an operation or workflow

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification - Step Parameters
parameters:
- name: category
in: query
value: $inputs.category
- name: breed
in: query
value: $inputs.breed
- name: location
in: query
value: $inputs.location
- name: page
in: query
value: 1
- name: pageSize
in: query
value: 1
- name: status
in: query
value: available
info
workflow
inputs
step
success action
outputs
inputs
criterion
workflows document
OpenAPI document) that can be referenced by one or more workflows
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.
the inputs used by this workflow
workflow steps, each representing a call to an API operation (or another workflow)
parameter objects, representing parameters to pass to an operation or workflow




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

A JSON Schema object representing the inputs used by this workflow

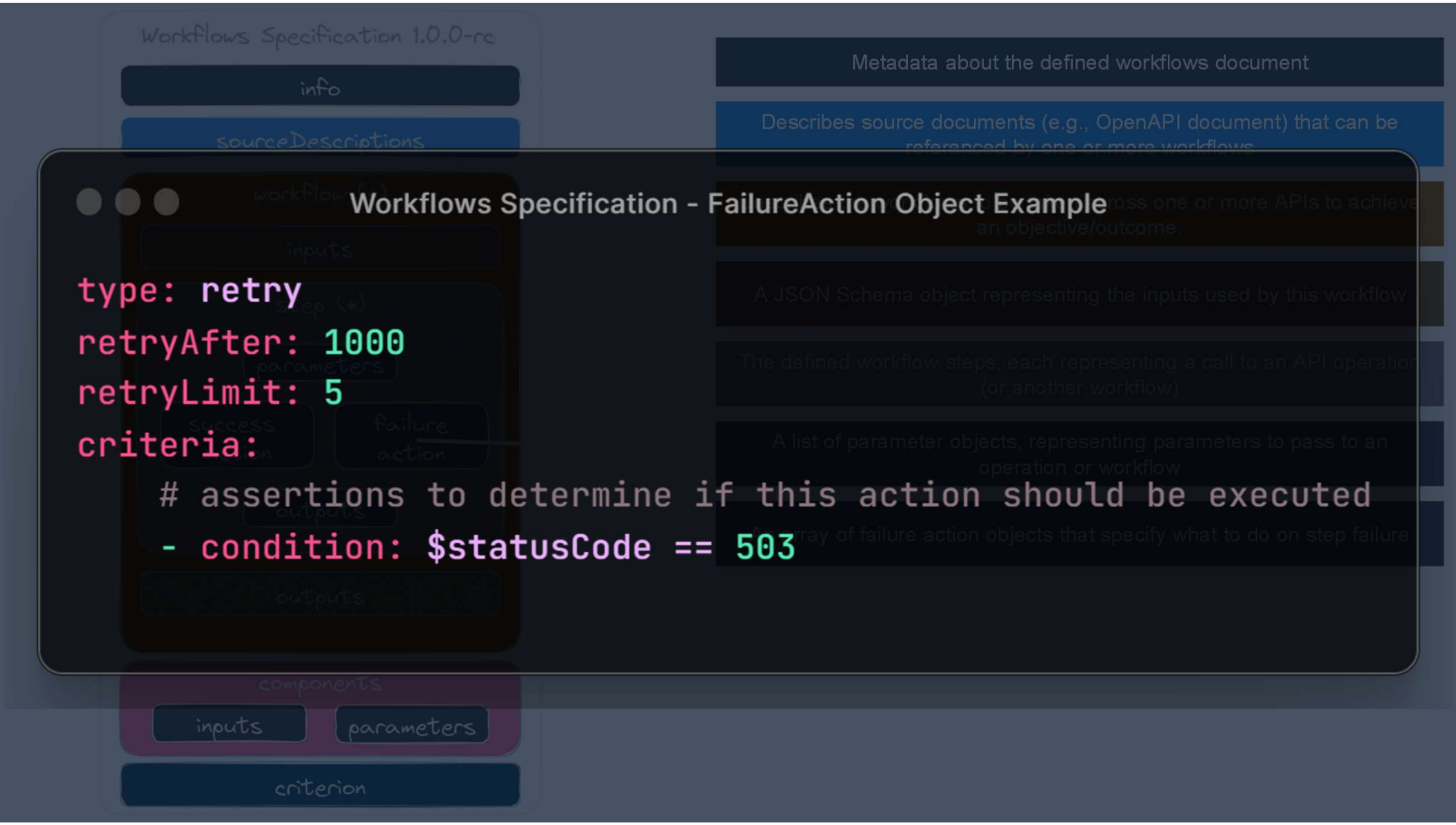
The defined workflow steps, each representing a call to an API operation (or another workflow)

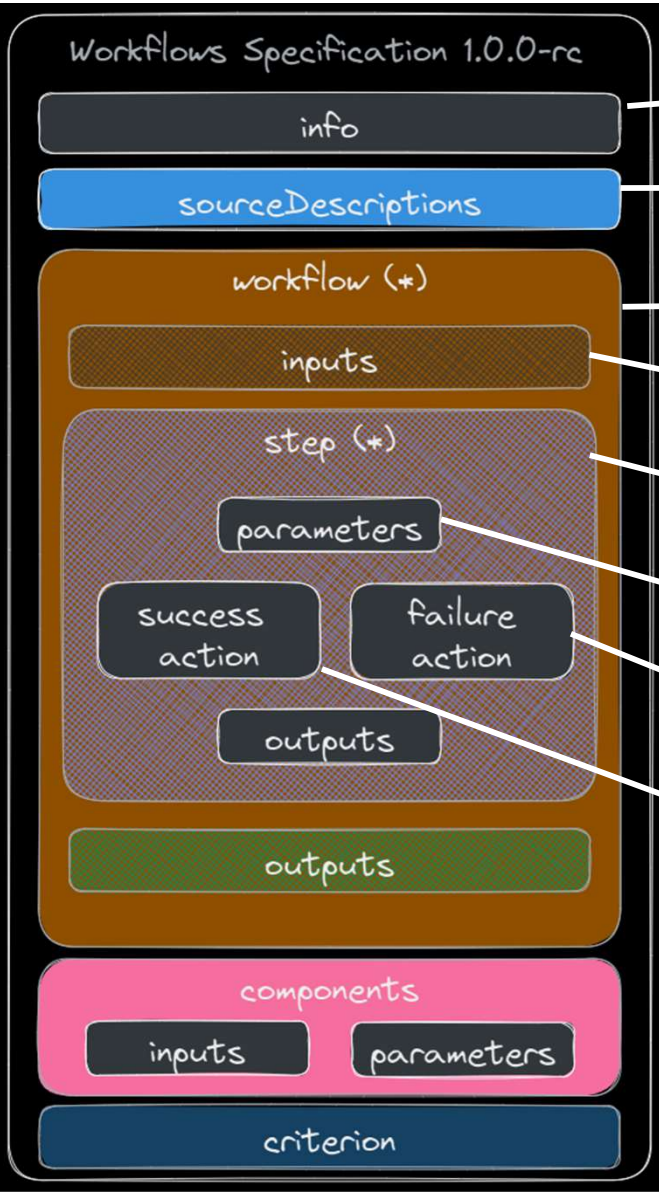
A list of parameter objects, representing parameters to pass to an operation or workflow

An array of failure action objects that specify what to do on step failure

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification 1.0.0-rc
info
sourceDescriptions
workflows
inputs
parameters
success action
failure action
outputs
components
inputs
parameters
criterion
Metadata about the defined workflows document
Describes source documents (e.g., OpenAPI document) that can be referenced by one or more workflows
A JSON Schema object representing the inputs used by this workflow
The defined workflow steps, each representing a call to an API operation (or another workflow)
A list of parameter objects, representing parameters to pass to an operation or workflow
An array of failure action objects that specify what to do on step failure
Workflows Specification - FailureAction Object Example
type: retry
retryAfter: 1000
retryLimit: 5
criteria:
# assertions to determine if this action should be executed
- condition: $statusCode == 503




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

A JSON Schema object representing the inputs used by this workflow

The defined workflow steps, each representing a call to an API operation (or another workflow)

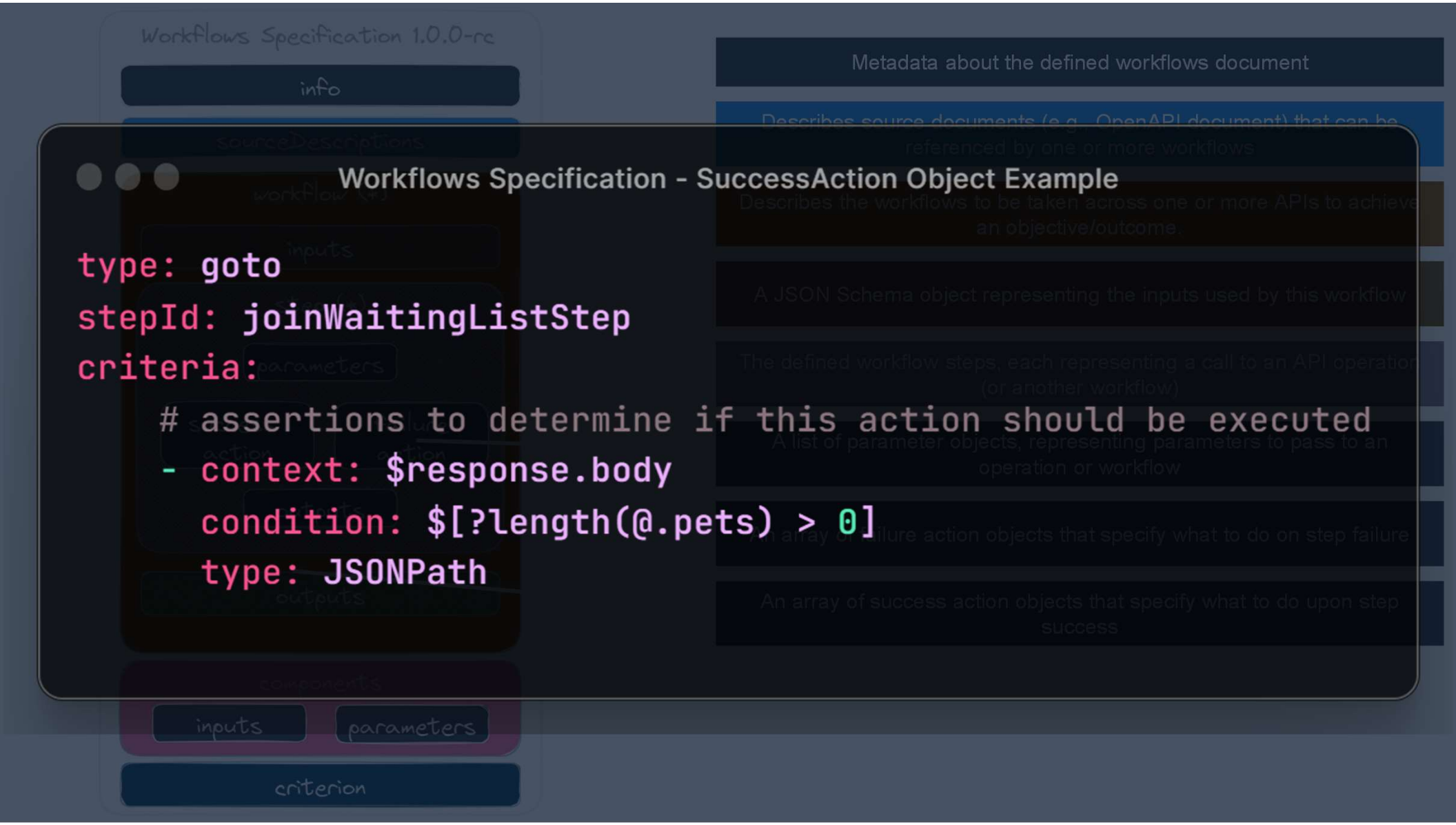
A list of parameter objects, representing parameters to pass to an operation or workflow

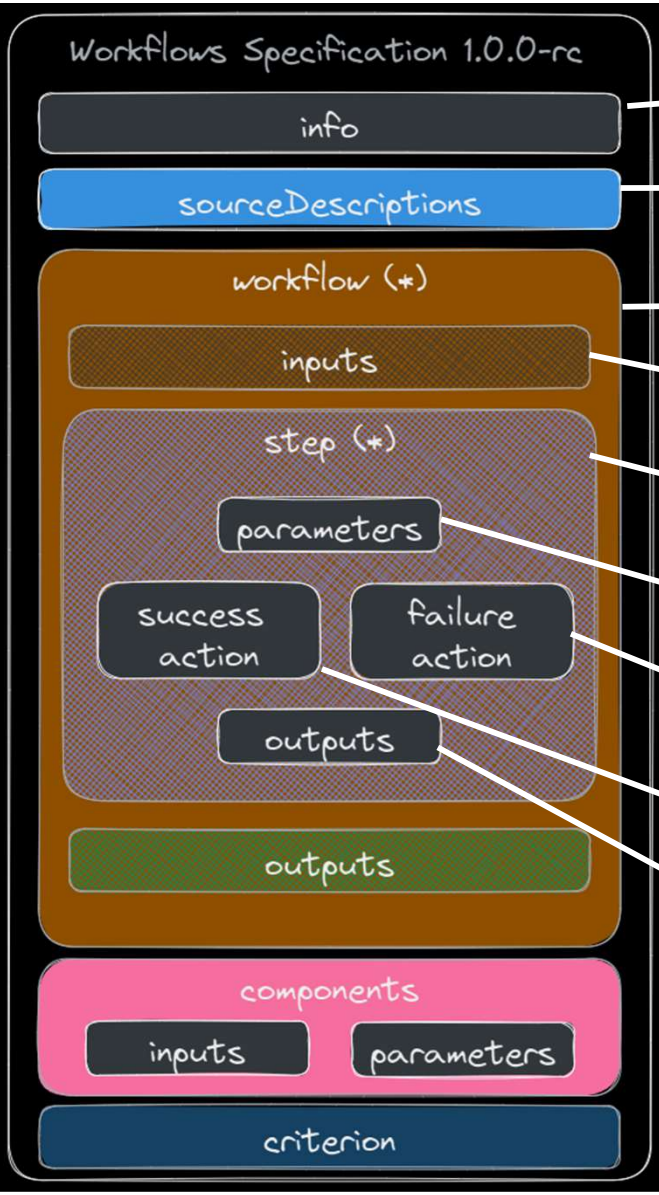
An array of failure action objects that specify what to do on step failure

An array of success action objects that specify what to do upon step success

SMARTBEAR

OPENAPI INITIATIVE


Workflows Specification 1.0.0-rc
info
Metadata about the defined workflows document
Describes source documents (e.g., OpenAPI document) that can be referenced by one or more workflows
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.
A JSON Schema object representing the inputs used by this workflow
The defined workflow steps, each representing a call to an API operation (or another workflow)
A list of parameter objects, representing parameters to pass to an operation or workflow
An array of failure action objects that specify what to do on step failure
An array of success action objects that specify what to do upon step success
inputs
parameters
criterion
Workflows Specification - SuccessAction Object Example
type: goto
stepId: joinWaitingListStep
criteria:
# assertions to determine if this action should be executed
- context: $response.body
condition: $[?length(@.pets) > 0]
type: JSONPath




Metadata about the defined workflows document

Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows

Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.

A JSON Schema object representing the inputs used by this workflow

The defined workflow steps, each representing a call to an API operation (or another workflow)

A list of parameter objects, representing parameters to pass to an operation or workflow

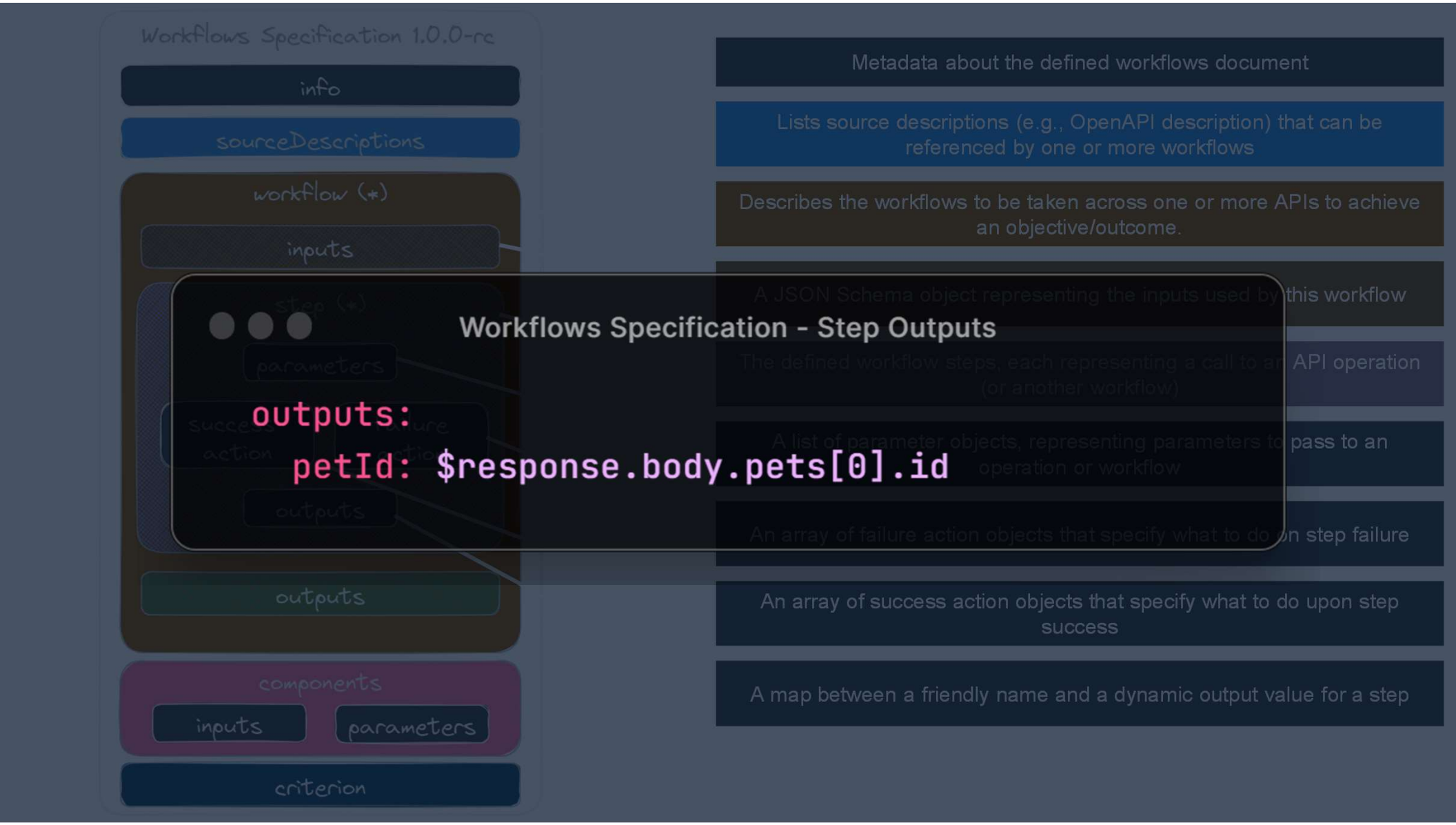
An array of failure action objects that specify what to do on step failure

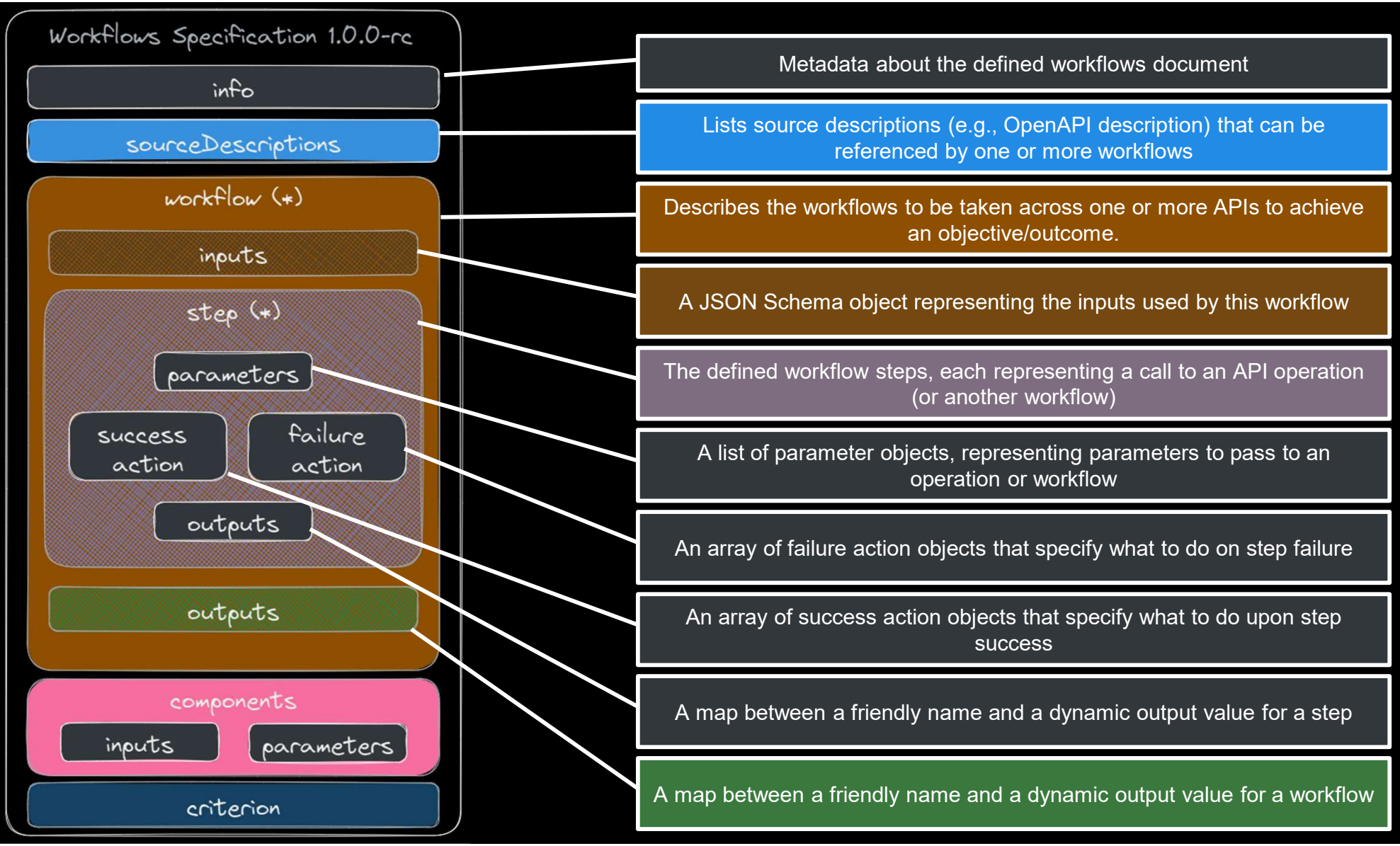
An array of success action objects that specify what to do upon step success

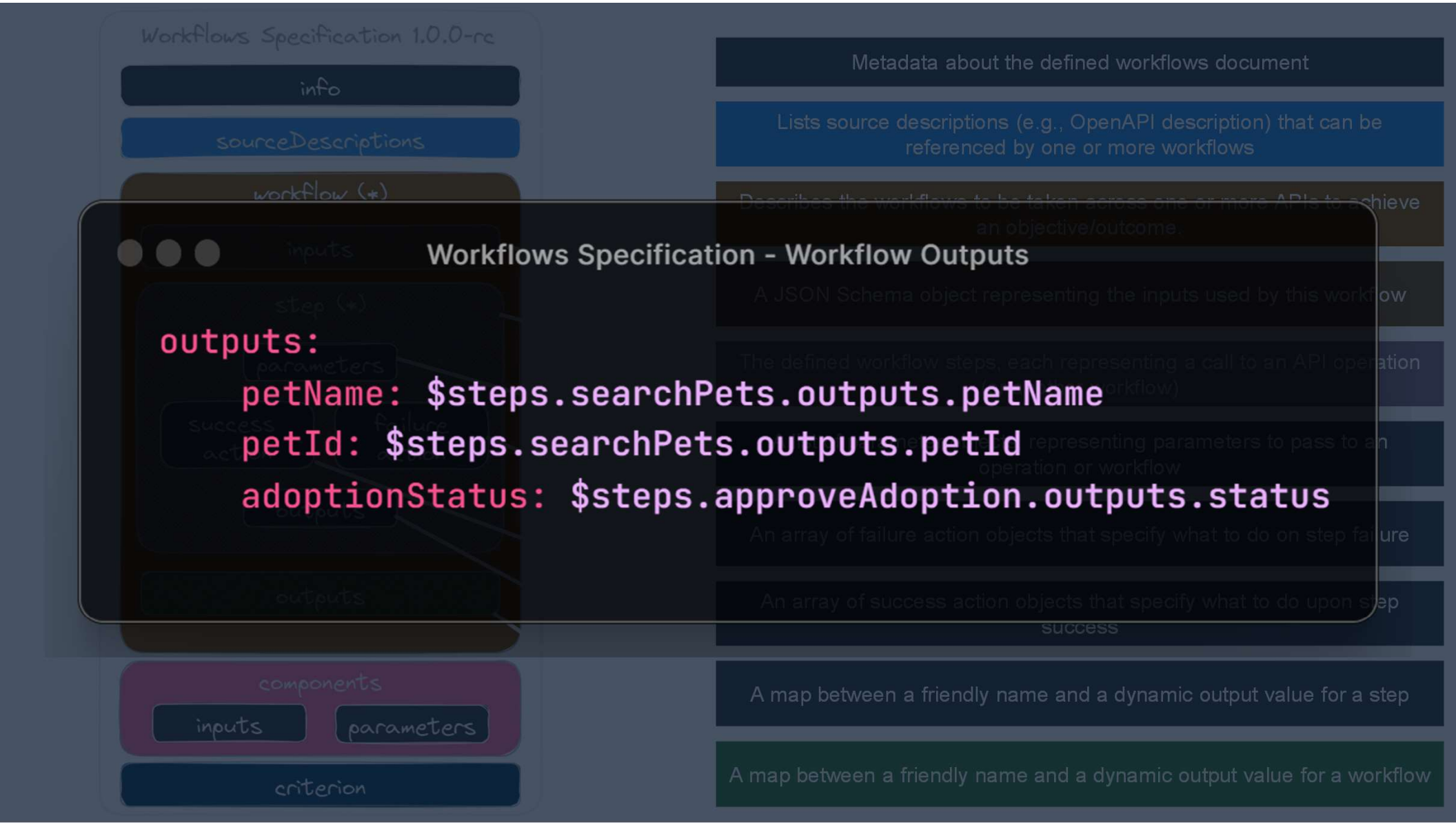
A map between a friendly name and a dynamic output value for a step

SMARTBEAR

OPENAPI INITIATIVE

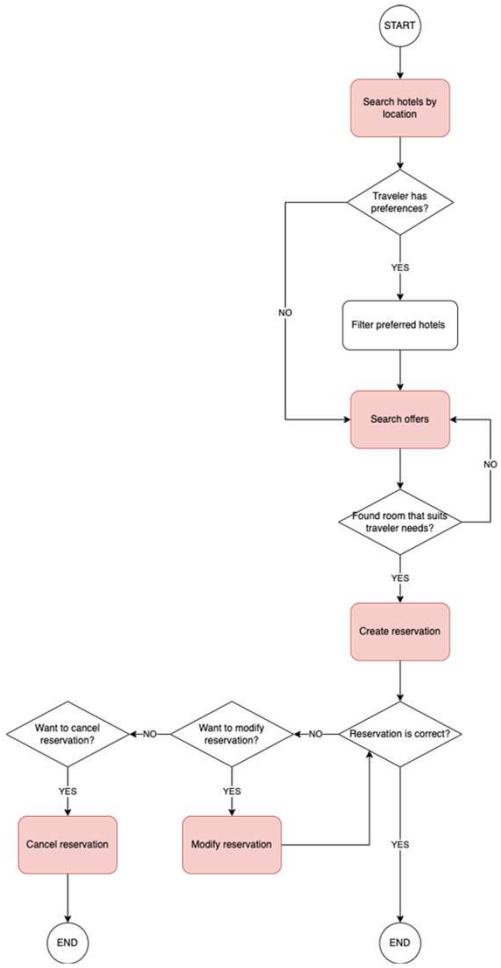

Workflows Specification 1.0.0-rc
info
sourceDescriptions
workflow (*)
inputs
parameters
outputs
components
inputs
parameters
criterion
Metadata about the defined workflows document
Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.
A JSON Schema object representing the inputs used by this workflow
The defined workflow steps, each representing a call to an API operation (or another workflow)
A list of parameter objects, representing parameters to pass to an operation or workflow
An array of failure action objects that specify what to do on step failure
An array of success action objects that specify what to do upon step success
A map between a friendly name and a dynamic output value for a step
Workflows Specification - Step Outputs
outputs:
petId: $response.body.pets[0].id


Workflows Specification 1.0.0-rc
info
sourceDescriptions
workflow (*)
inputs
step (*)
parameters
success action
failure action
outputs
outputs
components
inputs
parameters
criterion
Metadata about the defined workflows document
Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows
Describes the workflows to be taken across one or more APIs to achieve an objective/outcome.
A JSON Schema object representing the inputs used by this workflow
The defined workflow steps, each representing a call to an API operation (or another workflow)
A list of parameter objects, representing parameters to pass to an operation or workflow
An array of failure action objects that specify what to do on step failure
An array of success action objects that specify what to do upon step success
A map between a friendly name and a dynamic output value for a step
A map between a friendly name and a dynamic output value for a workflow


Workflows Specification 1.0.0-rc
info
sourceDescriptions
workflow (*)
components
inputs
parameters
criterion
Metadata about the defined workflows document
Lists source descriptions (e.g., OpenAPI description) that can be referenced by one or more workflows
Workflows Specification - Workflow Outputs
outputs:
petName: $steps.searchPets.outputs.petName
petId: $steps.searchPets.outputs.petId
adoptionStatus: $steps.approveAdoption.outputs.status
A map between a friendly name and a dynamic output value for a step
A map between a friendly name and a dynamic output value for a workflow

```yaml
workflowsSpec: 1.0.0-prerelease
info:
  title: LinksRez - Reservation process workflow
  summary: This workflow showcases how to create a hotel reservation through a sequenc
  description: |
    This workflow walks you through the steps of searching hotels, searching offers, c
  version: 1.0.0
sourceDescriptions:
- name: linksRezApiDescription
  url: ./linksrez.openapi.json
  type: openapi
workflows:
- workflowId: createReservation
  summary: User creates a hotel reservation for the desired room
  description: |
    This is how a user can:
    * search a hotel (or a list of hotels) near a desired location
    * list offers for the selected hotels
    * select one and
    * make a reservation for the selected room
    The workflow the hotel reservation confirmation number.
  inputs:
    $ref: "#/components/inputs/createReservationInputs"
  steps:
    - stepId: hotelsSearchStep
      description: Search hotels near a given location
      operationId: getHotelsByLocation
      parameters:
        - name: country
          in: query
```

0




**Workflow supported by LinksRez**

Describes all supported operations by LinksRez API. The boxes marked in red represent the endpoints provided by the API, the rest of logic must be implemented by the clients that consume the API.

1. Traveler searches for a list of hotels near a desired location. This search admits a few parameters: country, city, GPScoordinate (latitude, longitude), max distance from the provided GPS coordinate (or from the city center), etc.

2. If the traveler has preferences for a specific hotel or brand, the hotels list can be filtered.

3. Traveler can now start searching offers (availability and prices) for the desired hotels for a given reservation check-in and check-out date and number of guests (adults and children).

4. If a room that suits the traveler's needs is found, we can proceed with the reservation process. Otherwise, the traveler can modify the search parameters and try again until a room is found.

5. Traveler sends a reservation request containing all the required information: hotel, check-in, check-out, room type, number of guests, traveler information, payment details, etc. If the reservation is successfully processed by the hotel, a confirmation number is returned.

6. If the reservation is incorrect, either because the hotel wasn't able to confirm it, or the traveler made a mistake, the traveler can decide to modify or cancel the reservation.

**Proposed use case**

1. Search hotels

2. Select one hotel and search offers

3. No rooms founds

# SIG Efforts Combined

- OpenAPI Spec
  - Continual enhancement of API description that enables increased automation via tools
- Overlays
  - Ability to automate customizations of descriptions from a common source
- Workflows
  - Ability to combine and automate a sequence of API descriptions adding needed context of use
- Compliance
  - Ability to automate the validation of descriptions
- Industry SIGs
  - Ability to demonstrate relevancy and encourage adoption of the above efforts on an industry wide scale

# What’s Next?

Spec Upgrades

# What comes next for the OpenAPI Specification?

## Launching v4 "Moonwalk" by the end of 2024

- Semantics
  - It is not sufficient to describe the mechanics of an API without also describing its semantics
- Signatures
  - An API represents a set of functions, a function is identifiable by its signature
- Inclusion
  - Goal to describe all HTTP-based APIs, new or existing (ex: RPC APIs)
- Organization
  - Separation of concerns to allow modularization of different aspects of the API description
- Mechanical upgrading
  - Low impact upgrading from V3

https://www.openapis.org/blog/2023/12/06/openapi-moonwalk-2024

# What comes next for the OpenAPI Specification? continued

- There will be more releases before V4: 3.0.4, 3.1.1, and 3.2.0
  - Patch releases and minor function enhancements
- What is or is not included in each version is a community decision!!!
  - Don't sit there and gripe! Get involved!

- Suggested reading:
  - https://apisyouwonthate.com/blog/openapi-v4-project-moonwalk/
  - https://github.com/OAI/OAI-Courses
  - https://www.openapis.org/faq
  - https://learn.openapis.org/
- For travel industry digital retail, https://opentravel.org/

# Conclusions

- OpenAPI is driven by a dedicated community of companies and individuals working to improve API creation, deployment, and consumption thru machine readable documentation
  - Resulting automation provides higher quality and less labor required which drives down costs
  - A standard on how APIs are described and documented enables common use tooling
    - Tooling requirements unique to one company, government or even an industry is too small a marketplace to attract investment
    - Standards openly shared promote commonality in approach which provides users more choices in tool providers competing on the value they provide
- In travel, APIs costs (content acquisition) prevents the majority of travel offerings from reaching the digital marketplace
  - Standards, allowing mass automation, can drive down costs to make it affordable to get even a one-person tour operator's offering published and consumed

# Join the OAI/OpenAPI Community!

- **Join the community: *github.com/OAI/community***
- **All technical and special interest groups meetings: *openapis.org/calendar***
- **Upcoming events (including OAI Tracks): *openapis.org/events***


# Thank You!