开源社
kaiyuanshe

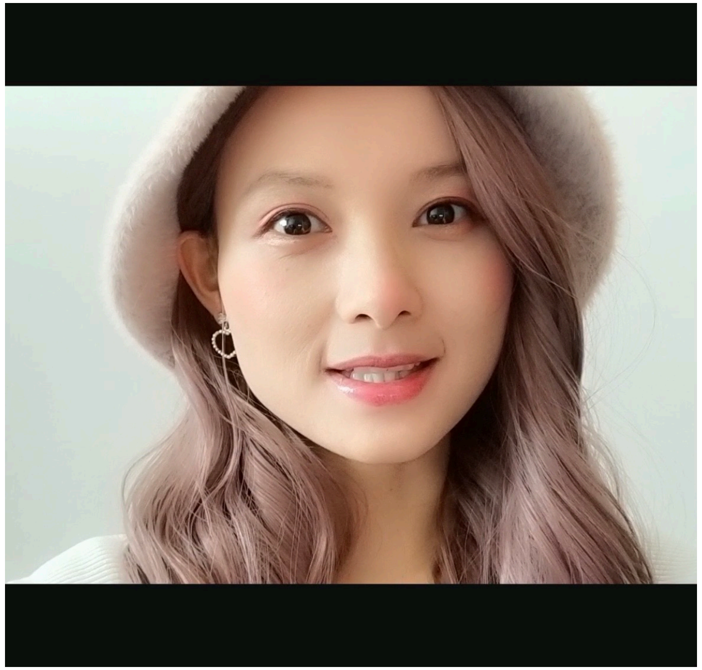
The Secret to Great Developer Experience is Killer Content

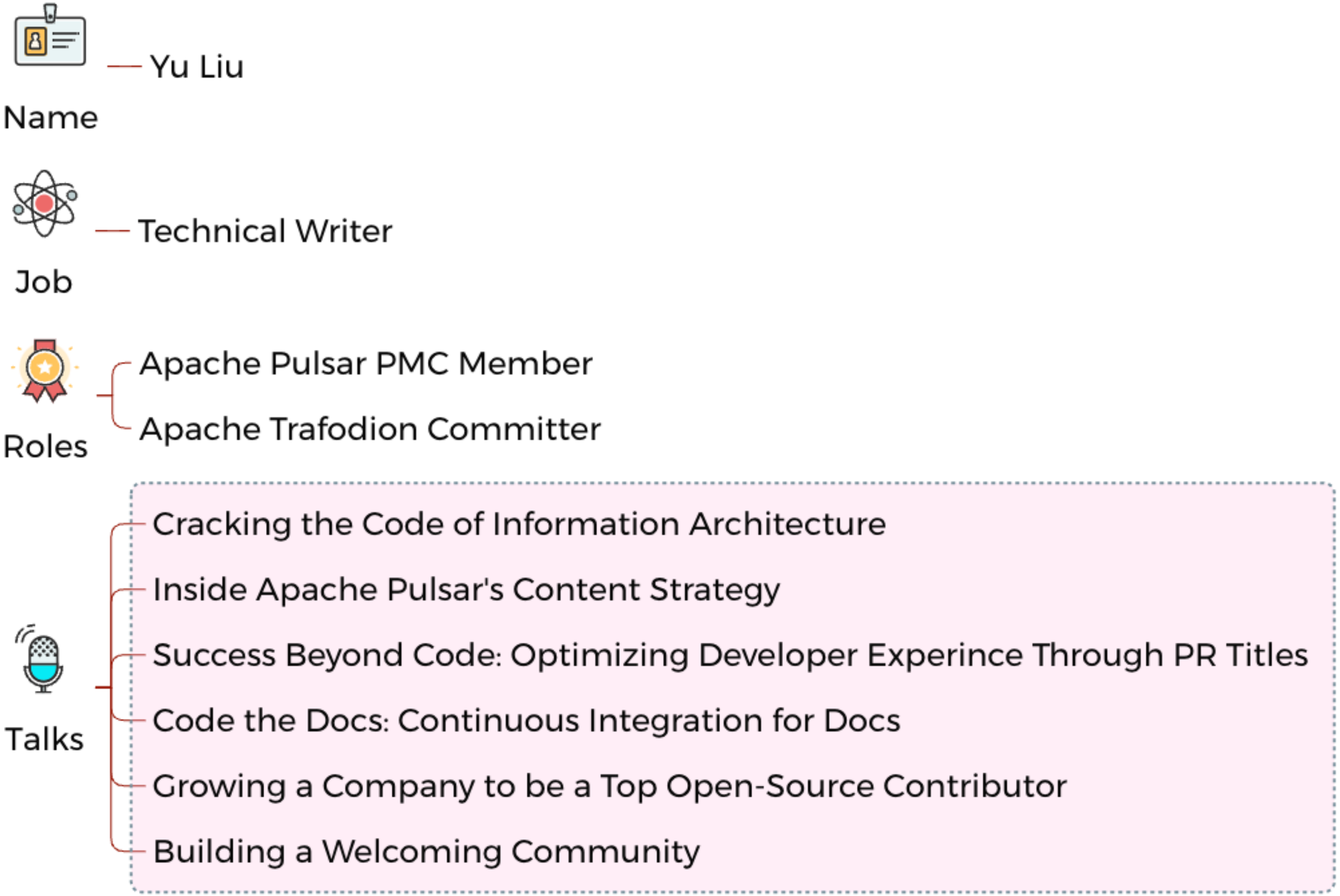
Yu Liu

Apache Pulsar PMC Member

Oct 28, 2023

Self intro





Name — Yu Liu

Job — Technical Writer

Roles
- Apache Pulsar PMC Member
- Apache Trafodion Committer

Talks
- Cracking the Code of Information Architecture
- Inside Apache Pulsar's Content Strategy
- Success Beyond Code: Optimizing Developer Experince Through PR Titles
- Code the Docs: Continuous Integration for Docs
- Growing a Company to be a Top Open-Source Contributor
- Building a Welcoming Community

Why should you attend this sharing?

- Shape system thinking

- Reuse efficient solutions

Agenda

1. What is DX (Developer Experience)?
2. Why Does DX Matter?
3. How to Design DX?
4. How to Evaluate DX?
5. Thoughts

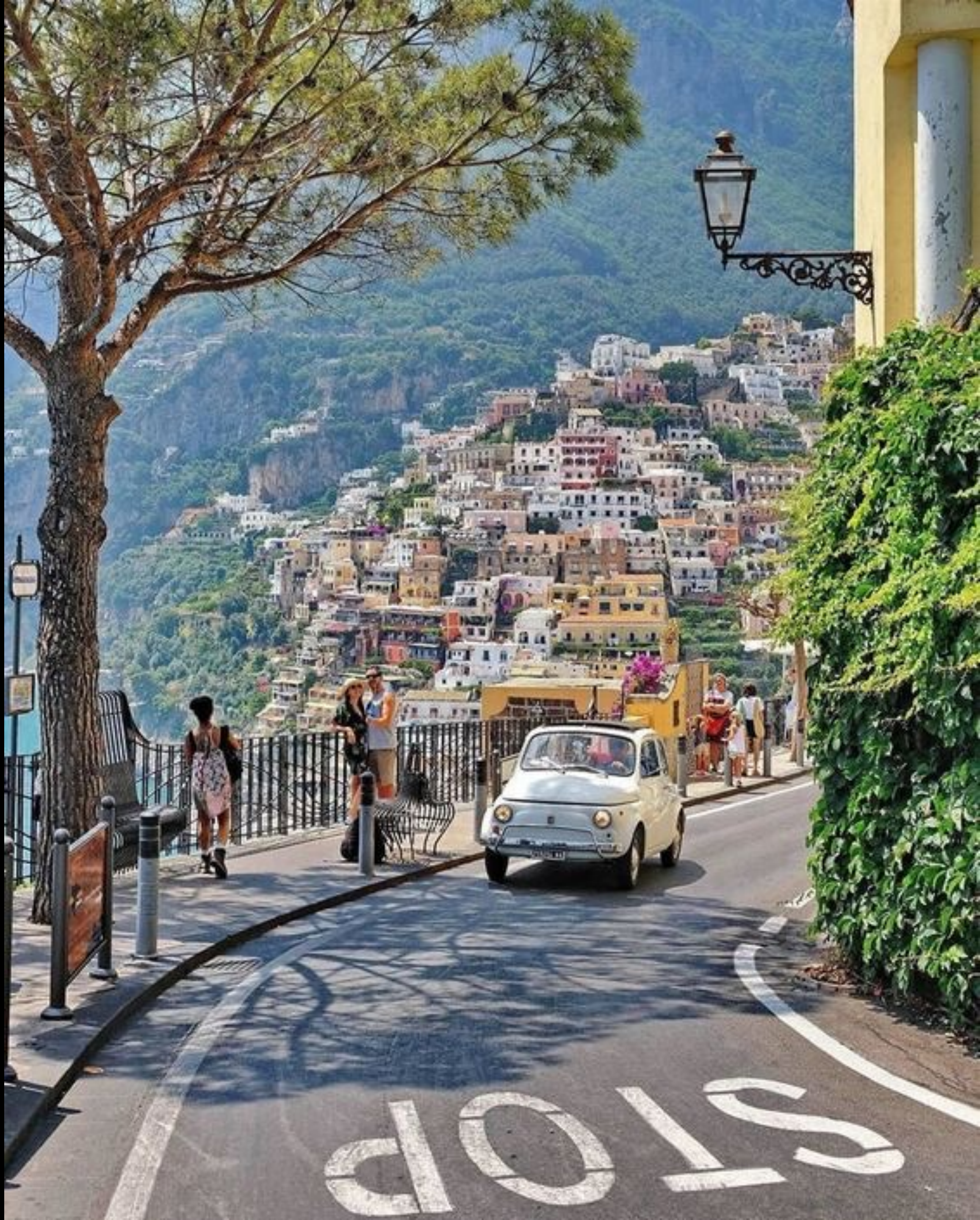
1. What is DX?

CULTURES, CUSTOMS AND TRADITIONS OF
SABAH
MALAYSIA
AN INTRODUCTION
VersaFine
CARTE
PARISIEN.

Comparison

Field	UX	DX
Audience	End users	Developers
Object	Consumer goods e.g., - iPhone 	Software products e.g., - **API (mostly used)** - SDK - Library - Framework
Goal	Use products e.g., use apps	Create software products e.g., create apps

2. Why Does DX Matter?

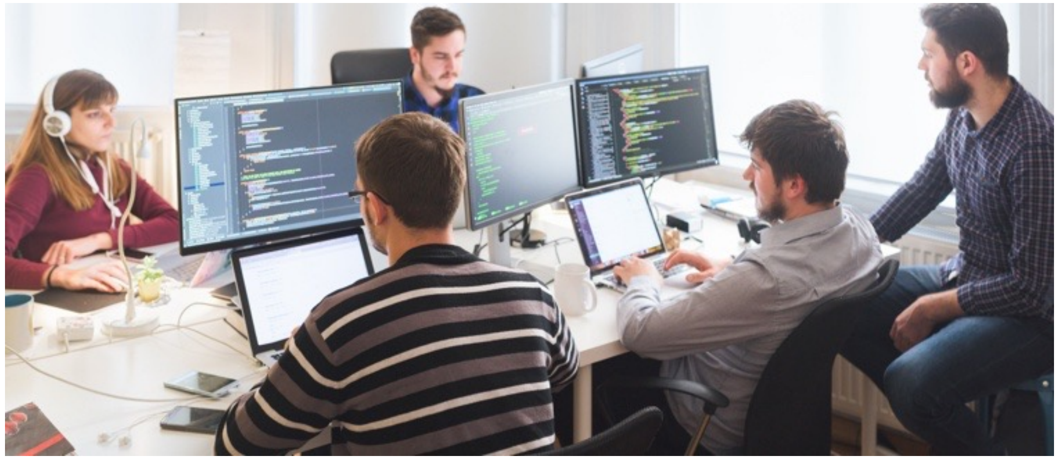


Developers tries, business buys

Business model

Business to Business (B2B)

Business to Developer (B2D)



CXOs

Developers (Devs)

Decision makers

Developers are rockstars of API economy

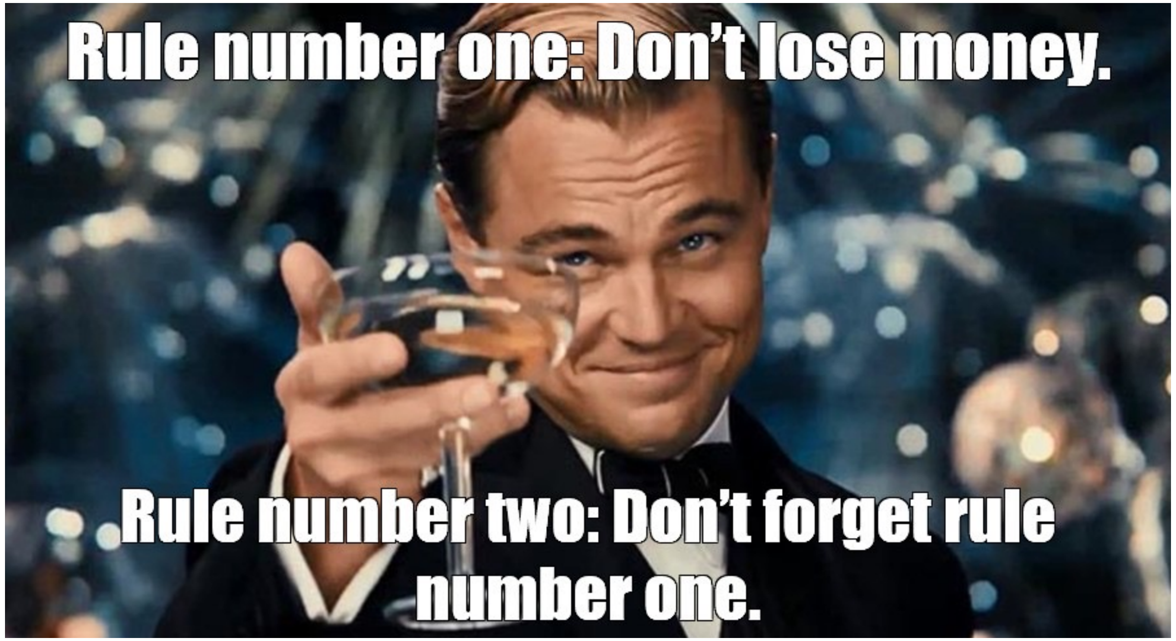
89 %

Of Developers Use APIs

(and there is 27 million of them)

Valued by API Market

$ **5.1** Billion


Rule number one: Don't lose money.
Rule number two: Don't forget rule number one.

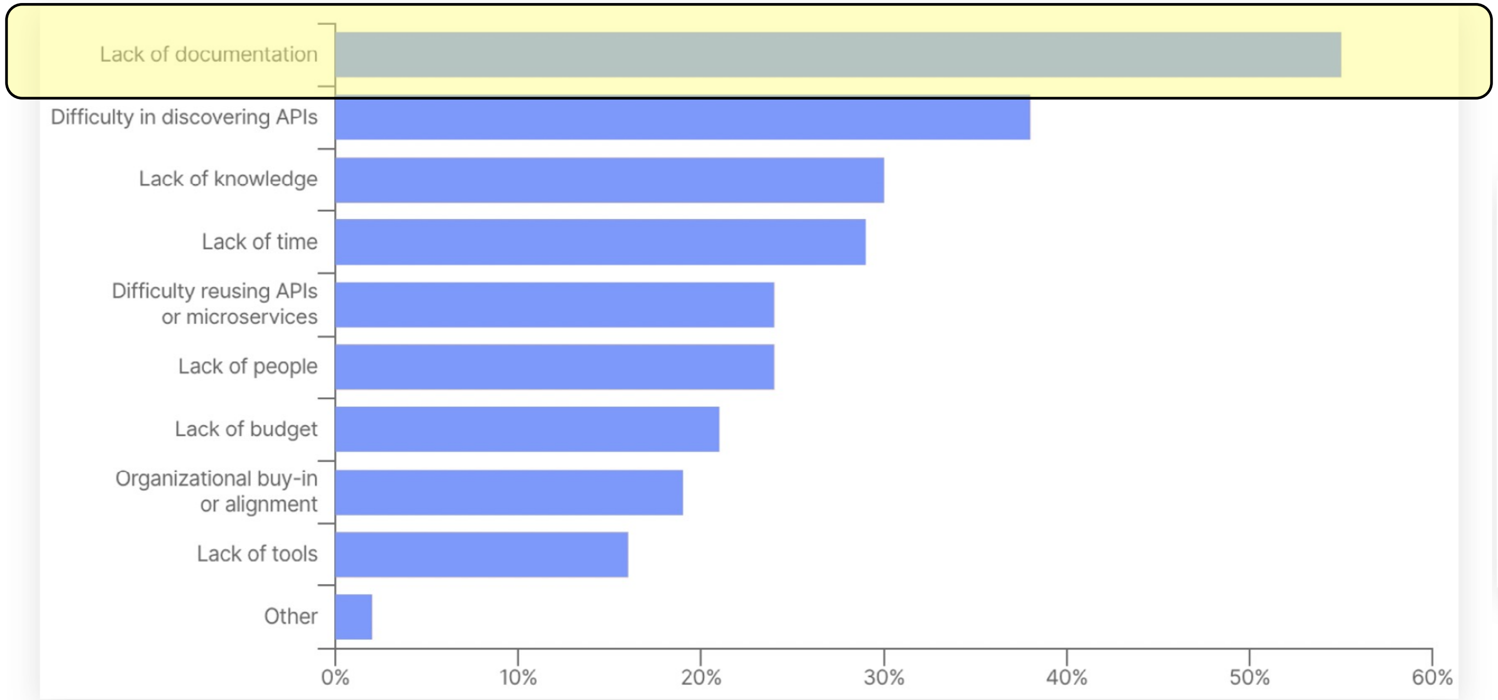
Source: DXHEROES

3. How to Design DX?

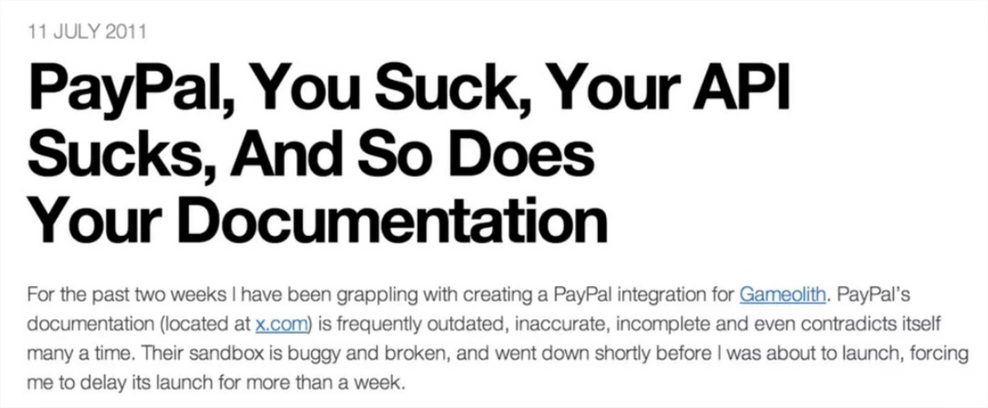
3.1 Issues

Issues from API content consumers

Obstacles to consuming APIs



Source: 2022 State of the API Report



11 JULY 2011

PayPal, You Suck, Your API Sucks, And So Does Your Documentation

For the past two weeks I have been grappling with creating a PayPal integration for Gameolith. PayPal's documentation (located at x.com) is frequently outdated, inaccurate, incomplete and even contradicts itself many a time. Their sandbox is buggy and broken, and went down shortly before I was about to launch, forcing me to delay its launch for more than a week.

Issues from API content producers

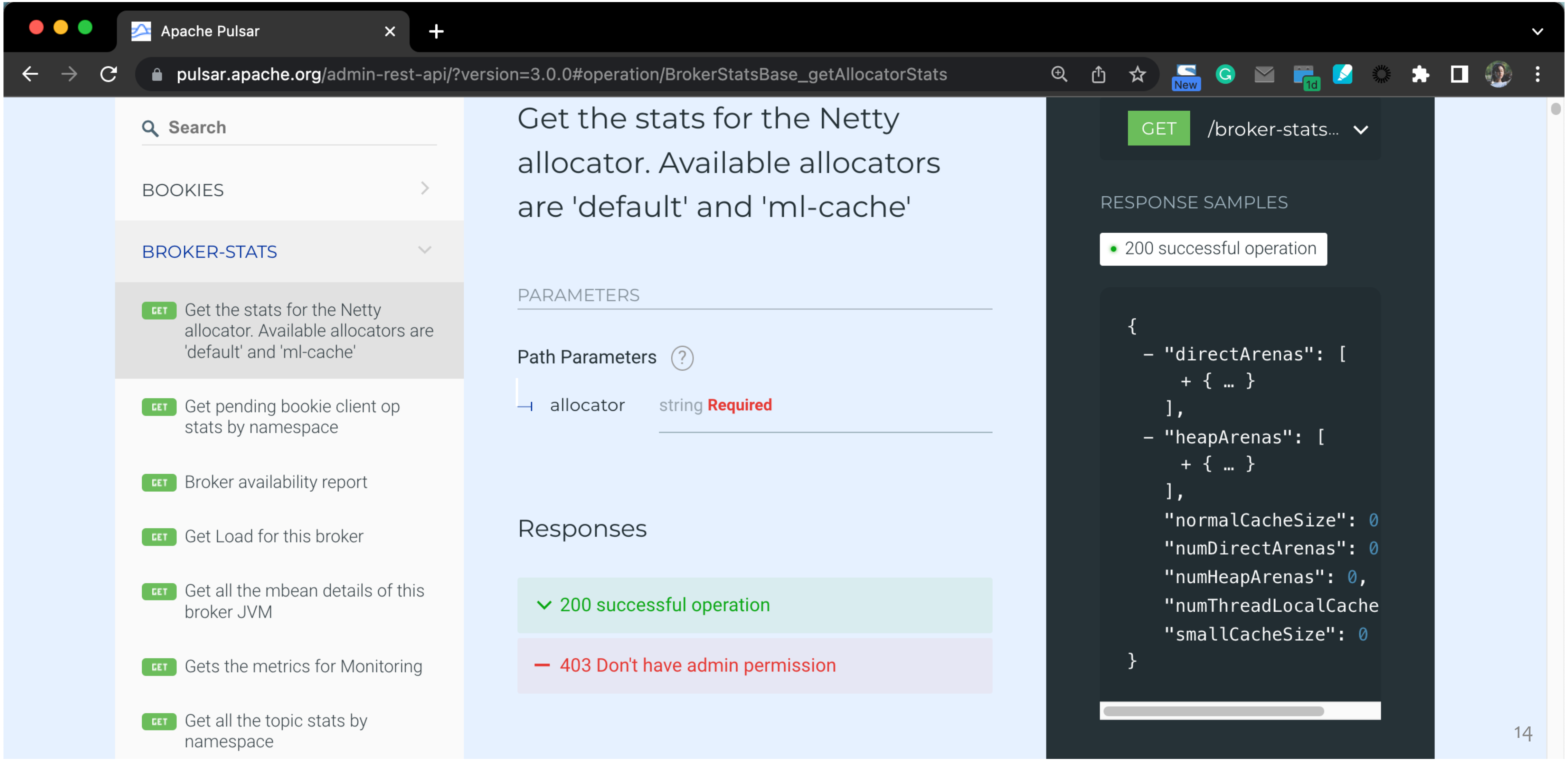
Stupid User Exception

Are you a stupid user?

Customers weren't technical enough.

We just couldn't get them to understand our APIs.

Pulsar API Reference



Apache Pulsar
pulsar.apache.org/admin-rest-api/?version=3.0.0#operation/BrokerStatsBase_getAllocatorStats
Search
BOOKIES
BROKER-STATS

GET Get the stats for the Netty allocator. Available allocators are 'default' and 'ml-cache'

GET Get pending bookie client op stats by namespace

GET Broker availability report

GET Get Load for this broker

GET Get all the mbean details of this broker JVM

GET Gets the metrics for Monitoring

GET Get all the topic stats by namespace

Get the stats for the Netty allocator. Available allocators are 'default' and 'ml-cache'

PARAMETERS

Path Parameters ?

allocator string **Required**

Responses

200 successful operation

403 Don't have admin permission

GET /broker-stats...

RESPONSE SAMPLES

● 200 successful operation

```
{
  - "directArenas": [
      + { … }
    ],
  - "heapArenas": [
      + { … }
    ],
    "normalCacheSize": 0
    "numDirectArenas": 0
    "numHeapArenas": 0,
    "numThreadLocalCache
    "smallCacheSize": 0
}
```

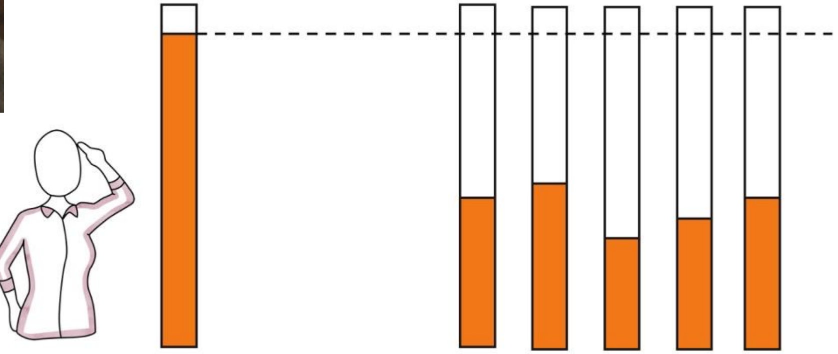
Context gap

API content producers

API content consumers

«

Curse of Knowledge



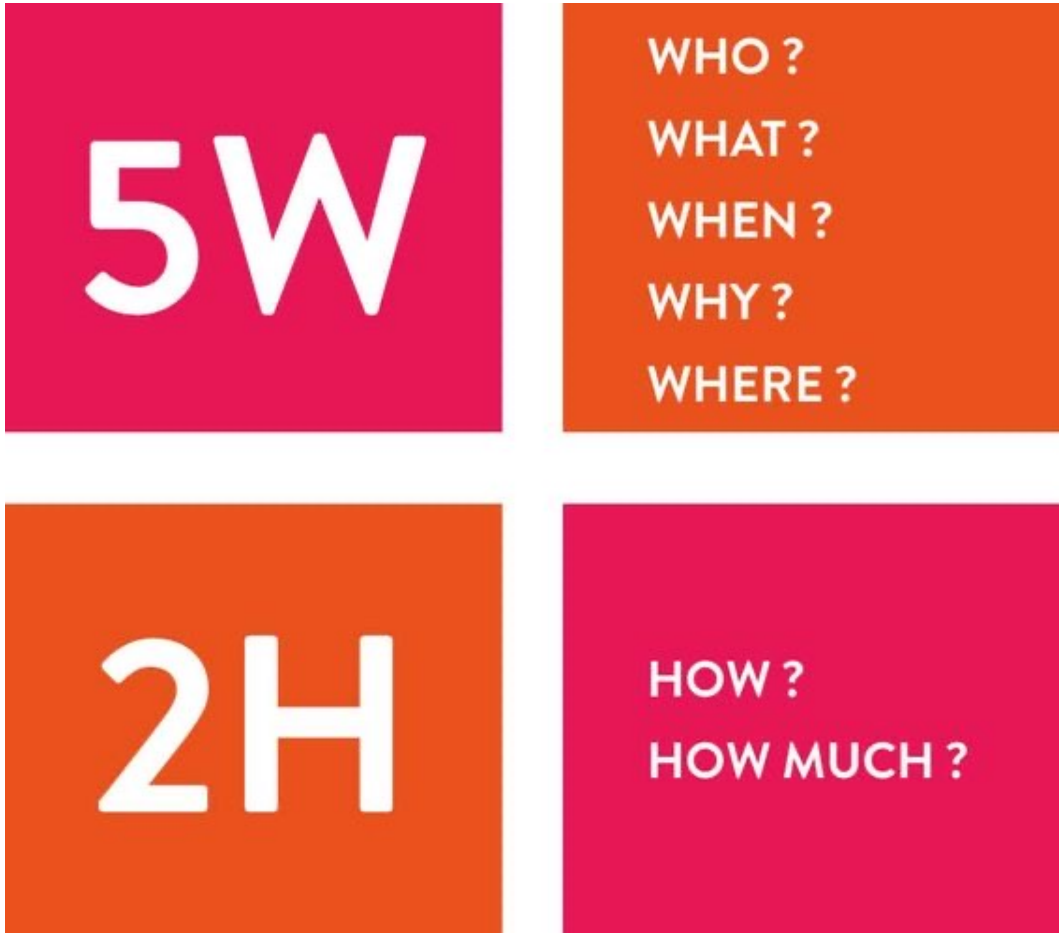
When you
know a lot...

It's easy to think
everyone knows, too.

»

3.2 Design Thinking

What is context?



5W

WHO ?
WHAT ?
WHEN ?
WHY ?
WHERE ?

2H

HOW ?
HOW MUCH ?

All info that strengthens users' comprehension and prevents miscommunication, e.g.,

- High-level product info
- Hidden considerations
- Environment when using products

Content often fails to tell context

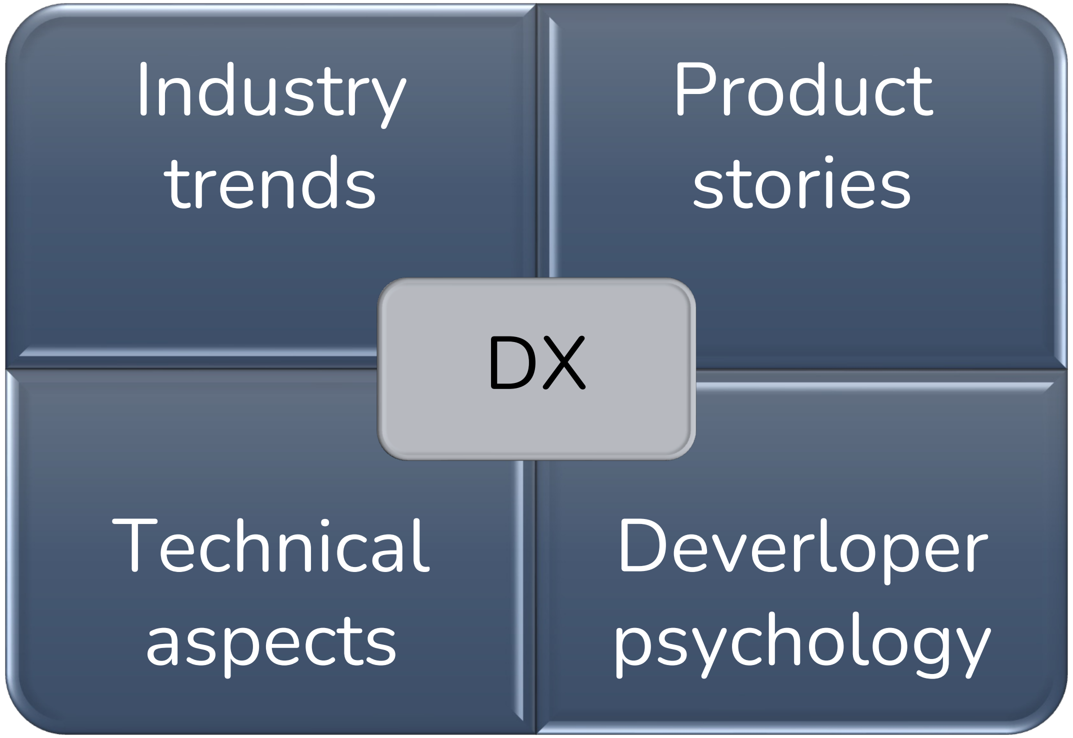
- Good at writing "How"

Requirement: Put the elephant into the fridge.
1. Open the door
2. Put it in
3. Close the door

- But fail to consider "Who/What/Why/When/How/How much"
  - Who cares about putting 🐘 into 🧊?
  - Who put?
  - Why put?
  - Which 🧊?
  - When put?
  - How much does put cost?

How to create context?

Building comprehensive context requires writers to understand



Industry trends

Product stories

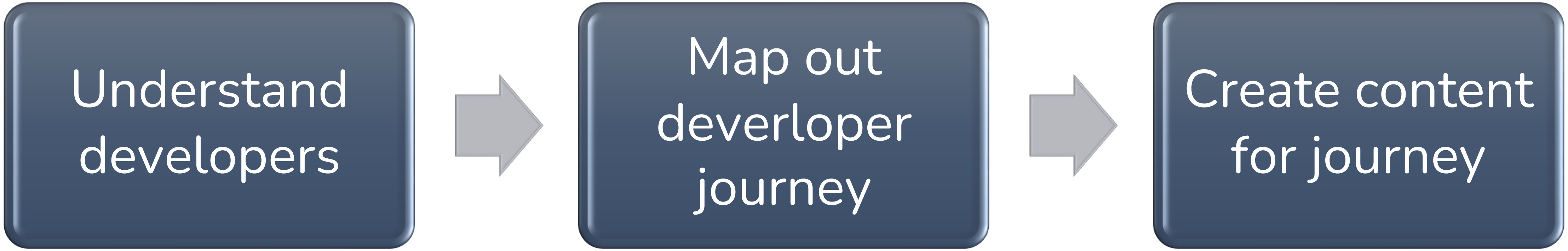
DX

Technical aspects

Deverloper psychology

3.3 Design Process

Overall steps



Understand developers

Map out deverloper journey

Create content for journey

3.3.1 Understand Developers

Developer decision-making unit

- Context (Who)

Roles	Job titles	What will they do?	Content needs	Content deliverables
Initiators	• Developers (coders) • DevOps	1. Begins the process by raising awareness internally, e.g., a developer who found your product via a Google search to an internal purchasing. 2. Use and interact with your product.	• Functionalities that answer specific needs • Customizations and customer-integration • Level and quality of technical support • Internal willingness to embrace	• Quick start • Playground / Free trial • Code samples / Tutorials • Use cases / Blogs • Trainings / Best practices • Knowledge base / Forums • Community / Events • Newsletters / Weekly reports
Influencers	• PoC / solution engineers • Customer support • Developer evangelists	1. Try your product firsthand and provide input. 2. Influence the overall decision though they do not have explicit decision-making authorities.		
Decision-makers	• CTOs • Architects • Tech leads • Product managers	1. Evaluate the technical aspects, such as compatibility with existing resources and the fit with the company's technical strategy, security, reliability, and so on.	• How it compares to other competitive choices in the market • Credibility & stability of the product & vendor	• Competitor comparisons • Success stories • Whitepapers • Release notes • Roadmaps
Budget holders	• CXOs	1. Evaluate the commercial aspects including pricing, business model, and ROI and gives approval to make the purchase.	• Overall ROI cases • Contractual terms & conditions • Reputations	• Pricing info • Terms of use • Usage policies • PR blogs

Developer mindset

Commonalities in characteristics

✓ Like facts not marketing

✓ Main motivator: I made it!

✓ Enjoy laziness

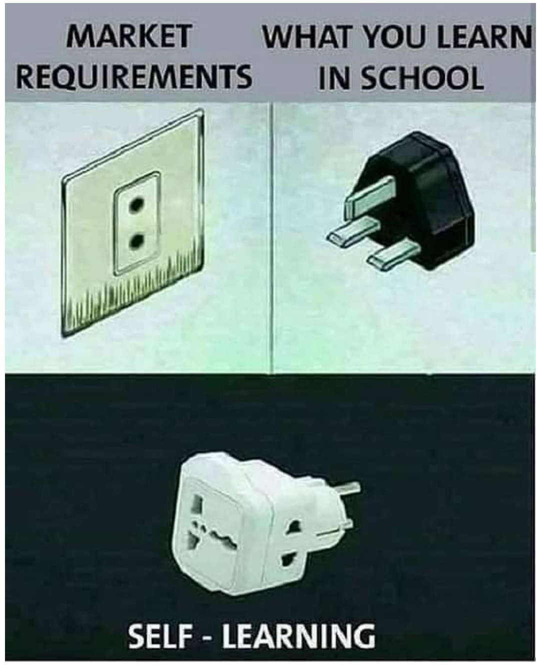
Developer mindset

Commonalities in learning

- ✓ Refer to docs when they have problems

When you start coding in a new language without reading the documentation
USCG

- ✓ Self study with code samples



- ✓ Kinesthetic learners

Developer archetype

- Who is reading your content?

	Systematic	Opportunistic	Pragmatic
Summary	✅ Gather needed info and then coding. Get a deep understanding of technology and read docs before using APIs.	✅ Gather needed info in parallel to coding. Solve problems and more willing to experiment with APIs without consulting docs.	✅ Falls in-between systematic and opportunistic. Use docs alongside exploration.
Learning habits	1) Review concepts, architecture, and features to understand the system and follow proposed suggestions closely. 2) Prepare dev environments. 3) Start tasks.	1) Search info in a very coarse-grained manner. (e.g., search for a specific piece of info and scroll briefly through some docs). 2) Check available solutions and tools. 3) Start tasks.	1) Learn just enough to start a task. 2) Refer to docs and other info resources to solve problems as they encounter them.

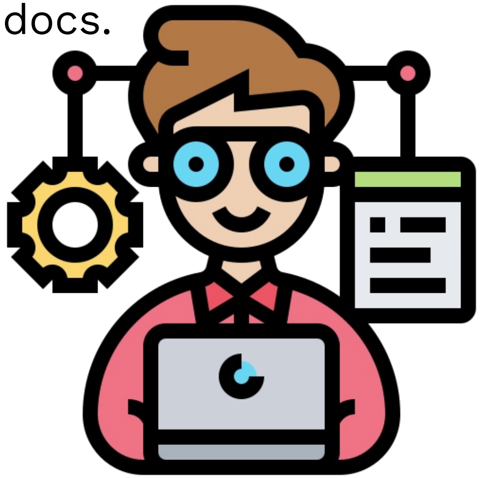
Developer archetype

- **Systematic**

Summary	Get a deep understanding of technology and read docs before using APIs.
Behavior	1. Gather needed info and then coding. Take some time to explore APIs and read docs and examples carefully. • Review concepts and architecture docs to understand the system as a whole. • Study the individual programming features to understand how pieces of the system work. • Follow proposed process and suggestions closely. • Form hypotheses about possible solutions, clarify terms they do not fully understand. • Notice docs that are not directly relevant to current task, however, still read it as a way to learn APIs. 2. Prepare a dev environment. 3. Start a task.

Developer archetype

- **Opportunistic**

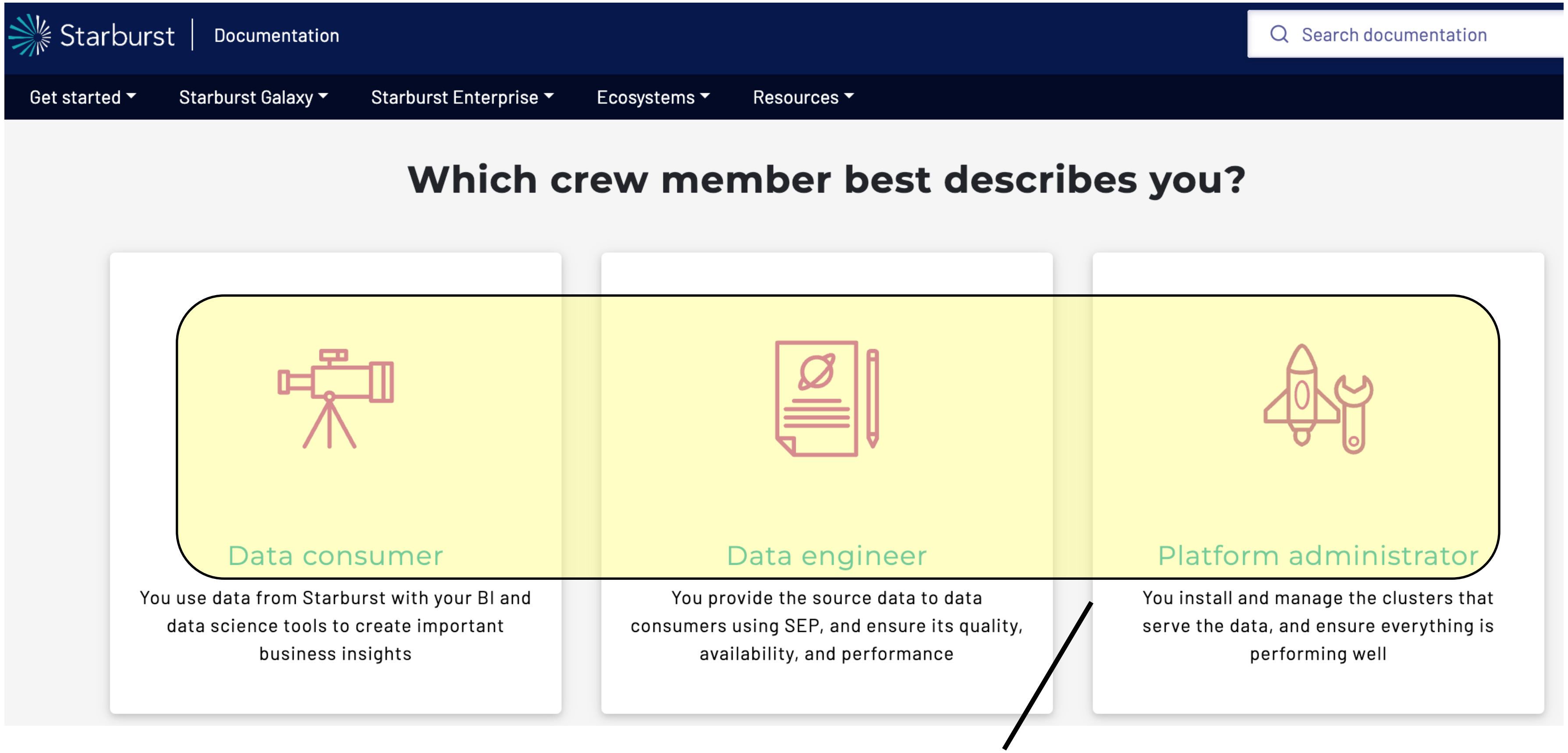
Summary	Solve problems and more willing to experiment with APIs without consulting docs.
Behavior	1. Gather needed info in parallel to coding. Search info in a very coarse-grained manner: • Do not take time to get a general overview of product. • Search the web to find answers rather than resorting to docs. • Search for a specific piece of info and scroll briefly through some docs. • Do a lot of searches while developing solutions and opening many browser tabs. 2. Check available solutions and tools. 3. Start a task. 

Developer archetype

- Pragmatic

Summary	Fall in-between systematic and opportunistic. Use docs alongside exploration.
Behavior	1. Learn just enough to start a task. 2. Refer to docs and other info resources to solve problems as they encounter them.

Implications for content design (all types)



Which crew member best describes you?

Data consumer

You use data from Starburst with your BI and data science tools to create important business insights

Data engineer

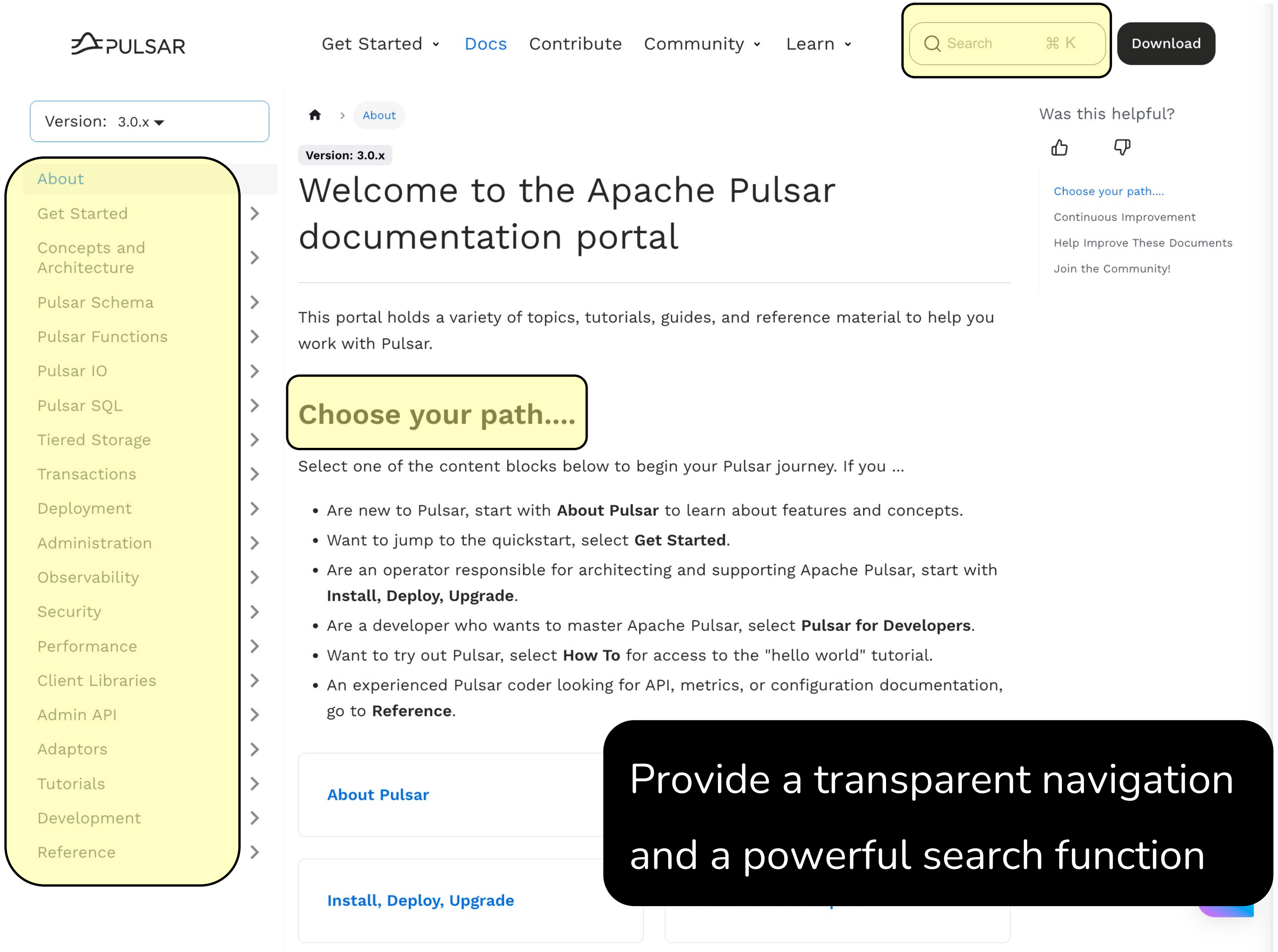
You provide the source data to data consumers using SEP, and ensure its quality, availability, and performance

Platform administrator

You install and manage the clusters that serve the data, and ensure everything is performing well

Design role-based learning paths

Implications for content design (all types)



Version: 3.0.x

Welcome to the Apache Pulsar documentation portal

This portal holds a variety of topics, tutorials, guides, and reference material to help you work with Pulsar.

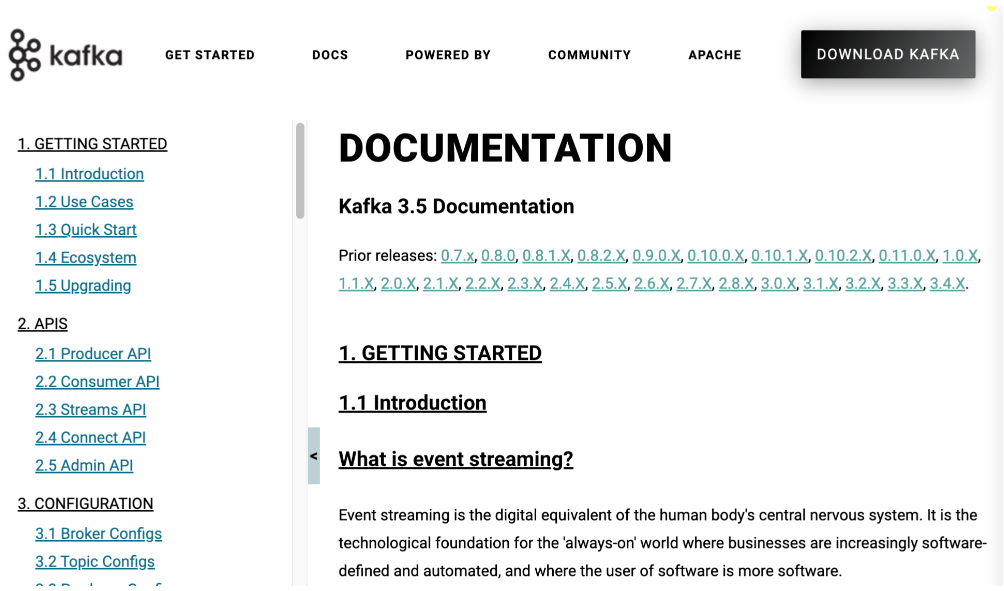
Choose your path....

Select one of the content blocks below to begin your Pulsar journey. If you ...

- Are new to Pulsar, start with **About Pulsar** to learn about features and concepts.
- Want to jump to the quickstart, select **Get Started**.
- Are an operator responsible for architecting and supporting Apache Pulsar, start with **Install, Deploy, Upgrade**.
- Are a developer who wants to master Apache Pulsar, select **Pulsar for Developers**.
- Want to try out Pulsar, select **How To** for access to the "hello world" tutorial.
- An experienced Pulsar coder looking for API, metrics, or configuration documentation, go to **Reference**.

About Pulsar

Install, Deploy, Upgrade



DOCUMENTATION

Kafka 3.5 Documentation

Prior releases: 0.7.x, 0.8.0, 0.8.1.X, 0.8.2.X, 0.9.0.X, 0.10.0.X, 0.10.1.X, 0.10.2.X, 0.11.0.X, 1.0.X, 1.1.X, 2.0.X, 2.1.X, 2.2.X, 2.3.X, 2.4.X, 2.5.X, 2.6.X, 2.7.X, 2.8.X, 3.0.X, 3.1.X, 3.2.X, 3.3.X, 3.4.X.

1. GETTING STARTED

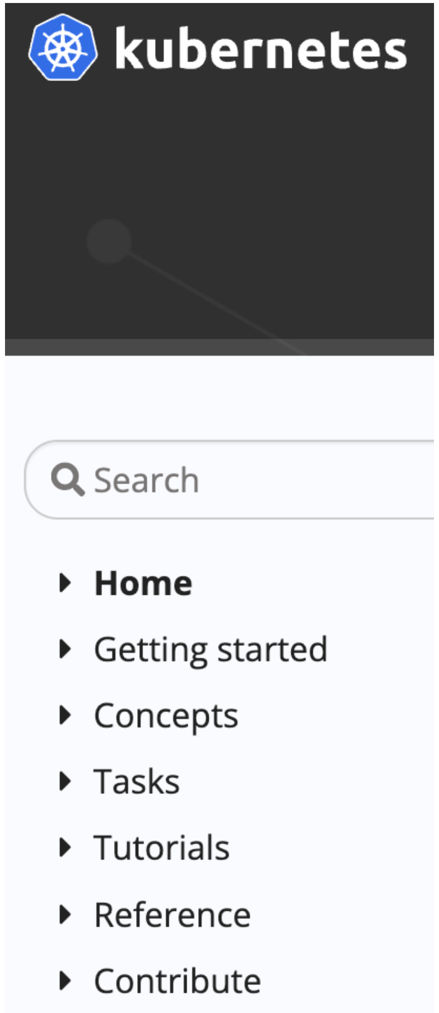
1.1 Introduction

What is event streaming?

Event streaming is the digital equivalent of the human body's central nervous system. It is the technological foundation for the 'always-on' world where businesses are increasingly software-defined and automated, and where the user of software is more software.

Provide a transparent navigation and a powerful search function

Implications for content design (Systematic)

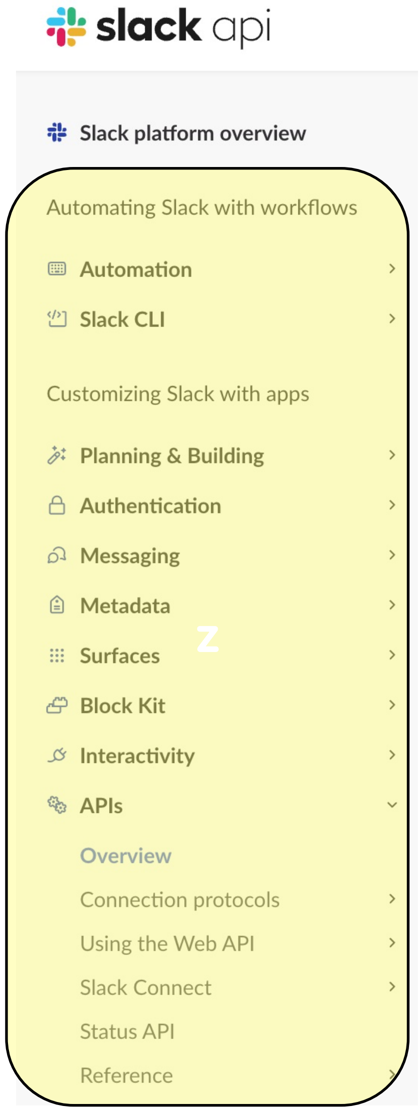


kubernetes

Search

- Home
- Getting started
- Concepts
- Tasks
- Tutorials
- Reference
- Contribute

Organize the content according to API functionality or content domain rather than info type



slack api

Slack platform overview

Automating Slack with workflows

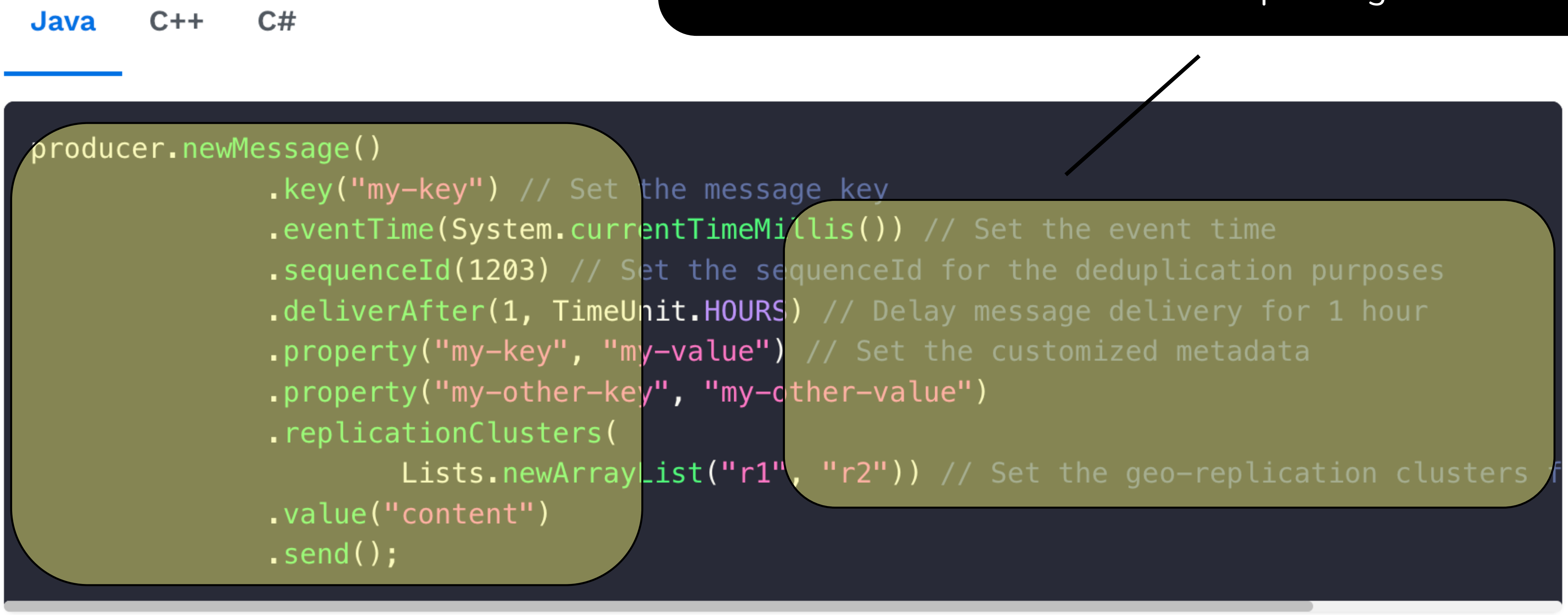
- Automation
- Slack CLI

Customizing Slack with apps

- Planning & Building
- Authentication
- Messaging
- Metadata
- Surfaces
- Block Kit
- Interactivity
- APIs
  - Overview
  - Connection protocols
  - Using the Web API
  - Slack Connect
  - Status API
  - Reference

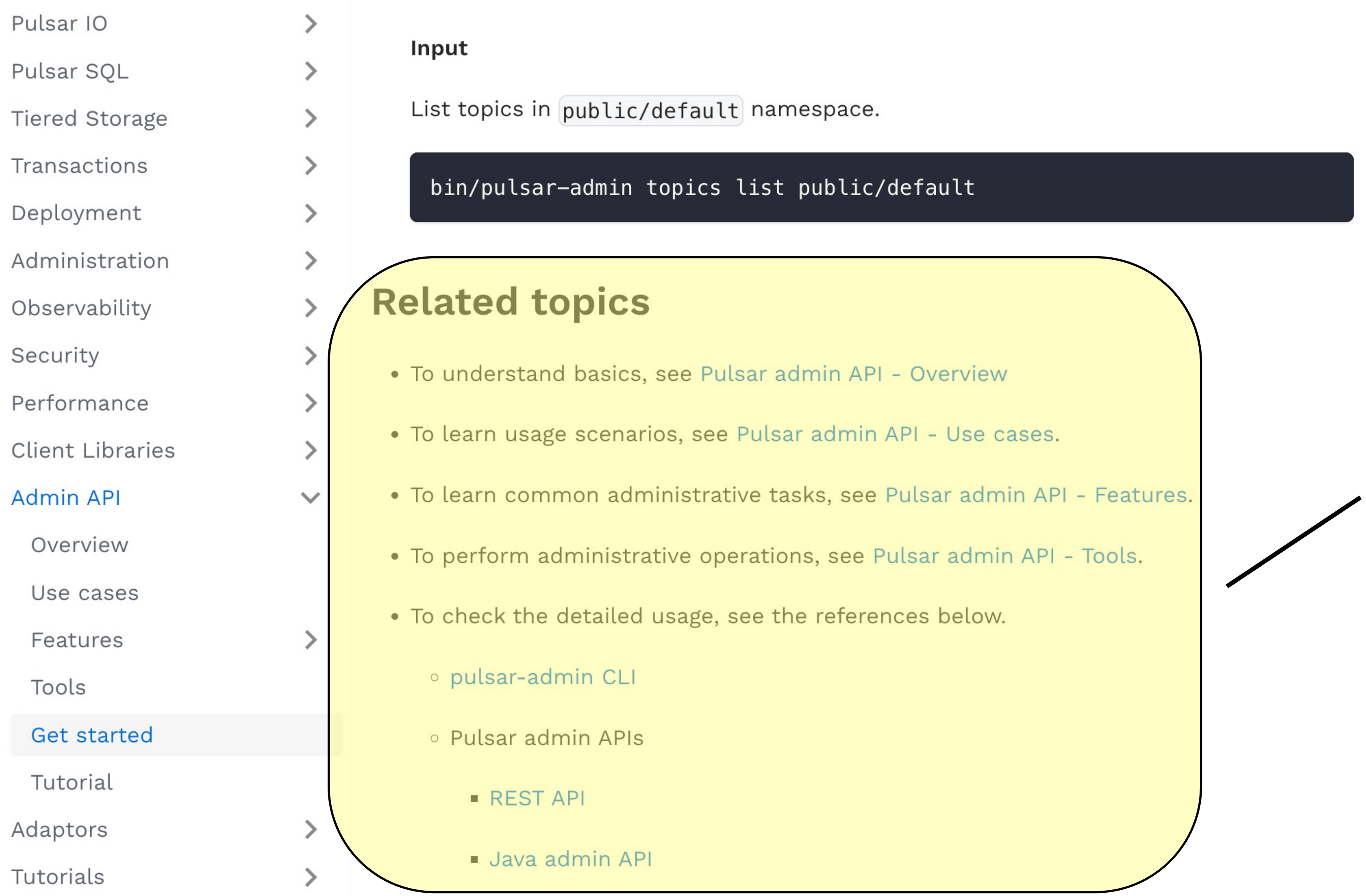
Implications for content design (Opportunistic)

- Present code examples in small chunks
- Integrate critical pieces of conceptual info into the code examples or source code with comments explaining what the code is doing



```
producer.newMessage()
              .key("my-key") // Set the message key
              .eventTime(System.currentTimeMillis()) // Set the event time
              .sequenceId(1203) // Set the sequenceId for the deduplication purposes
              .deliverAfter(1, TimeUnit.HOURS) // Delay message delivery for 1 hour
              .property("my-key", "my-value") // Set the customized metadata
              .property("my-other-key", "my-other-value")
              .replicationClusters(
                     Lists.newArrayList("r1", "r2")) // Set the geo-replication clusters f
              .value("content")
              .send();
```

Implications for content design (Opportunistic)



Input

List topics in `public/default` namespace.

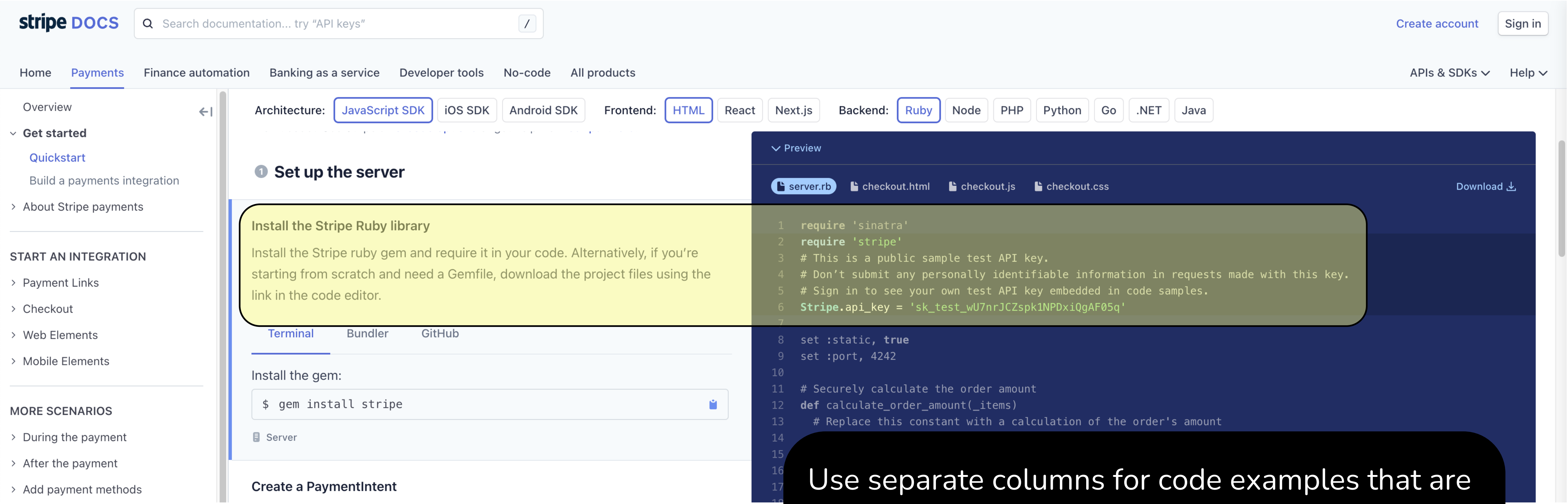
```
bin/pulsar-admin topics list public/default
```

Related topics

- To understand basics, see Pulsar admin API - Overview
- To learn usage scenarios, see Pulsar admin API - Use cases.
- To learn common administrative tasks, see Pulsar admin API - Features.
- To perform administrative operations, see Pulsar admin API - Tools.
- To check the detailed usage, see the references below.
  - pulsar-admin CLI
  - Pulsar admin APIs
    - REST API
    - Java admin API

- Provide important info redundantly
- Show domain-related background knowledge on-demand and integrate with the description of tasks and usage scenarios

Implications for content design (Opportunistic)



1 Set up the server

Install the Stripe Ruby library

Install the Stripe ruby gem and require it in your code. Alternatively, if you're starting from scratch and need a Gemfile, download the project files using the link in the code editor.

Install the gem:

```
$ gem install stripe
```

Create a PaymentIntent

```
require 'sinatra'
require 'stripe'
# This is a public sample test API key.
# Don't submit any personally identifiable information in requests made with this key.
# Sign in to see your own test API key embedded in code samples.
Stripe.api_key = 'sk_test_wU7nrJCZspk1NPDxiQgAF05q'

set :static, true
set :port, 4242

# Securely calculate the order amount
def calculate_order_amount(_items)
  # Replace this constant with a calculation of the order's amount
```

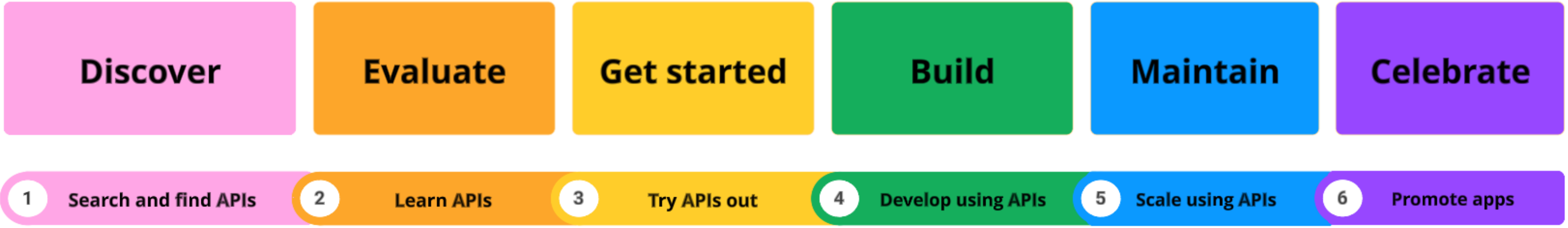
Signal text-to-code connections

Use separate columns for code examples that are aligned to the columns containing the text blocks referring to the code examples, making it easier to jump to relevant code examples directly.

3.3.2 Map out Developer Journey

Map out developer journey

Process: how developers use APIs



Discover

Evaluate

Get started

Build

Maintain

Celebrate

1 Search and find APIs

2 Learn APIs

3 Try APIs out

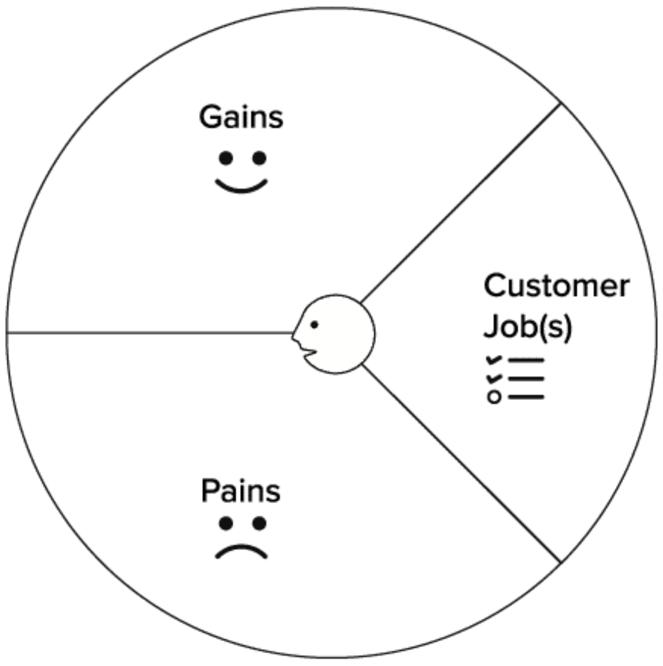
4 Develop using APIs

5 Scale using APIs

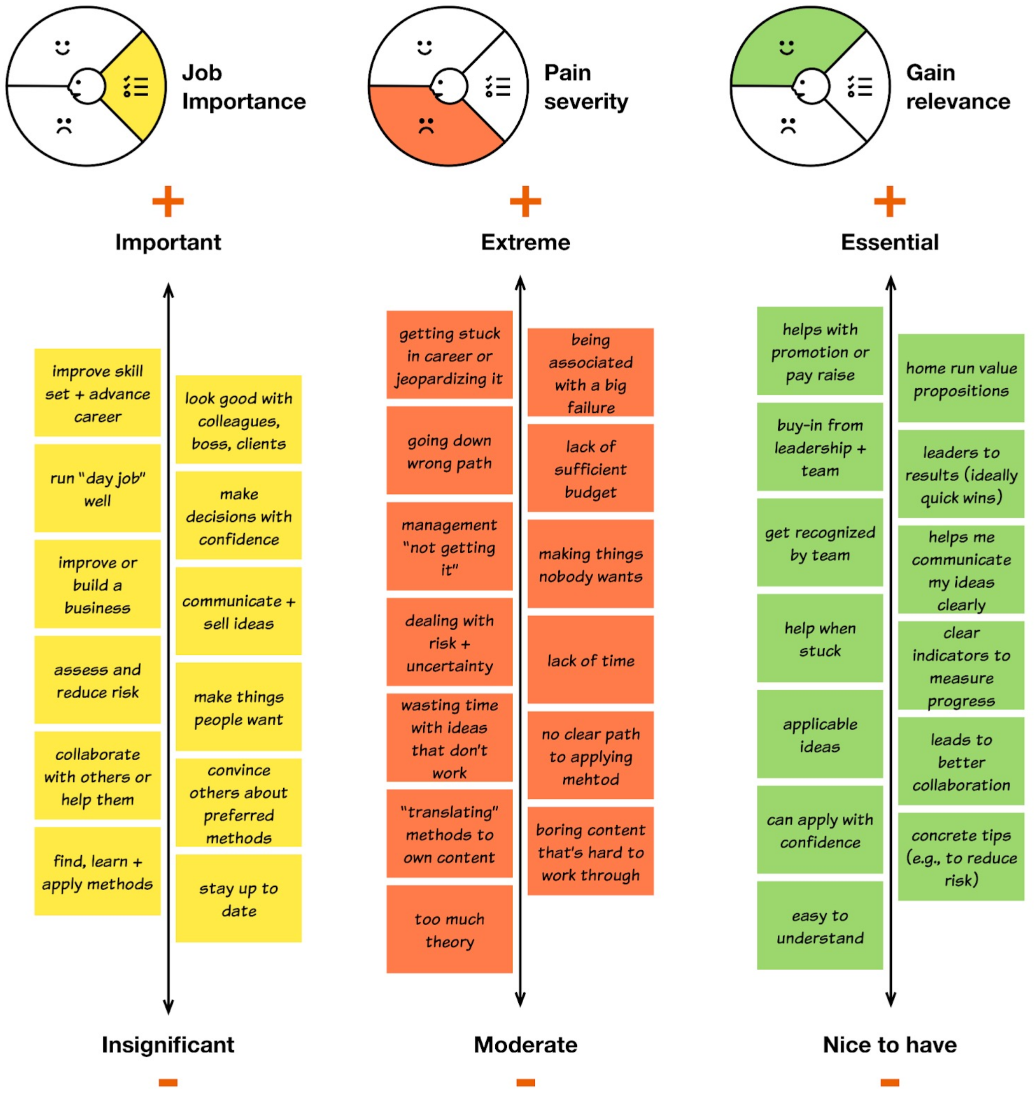
6 Promote apps

Analyze needs for each stage

Value Proposition Canvas



CUSTOMER PROFILE



Job Importance

+ Important

- improve skill set + advance career
- run "day job" well
- improve or build a business
- assess and reduce risk
- collaborate with others or help them
- find, learn + apply methods
- look good with colleagues, boss, clients
- make decisions with confidence
- communicate + sell ideas
- make things people want
- convince others about preferred methods
- stay up to date

Insignificant –

Pain severity

+ Extreme

- getting stuck in career or jeopardizing it
- going down wrong path
- management "not getting it"
- dealing with risk + uncertainty
- wasting time with ideas that don't work
- "translating" methods to own content
- too much theory
- being associated with a big failure
- lack of sufficient budget
- making things nobody wants
- lack of time
- no clear path to applying mehtod
- boring content that's hard to work through

Moderate –

Gain relevance

+ Essential

- helps with promotion or pay raise
- buy-in from leadership + team
- get recognized by team
- help when stuck
- applicable ideas
- can apply with confidence
- easy to understand
- home run value propositions
- leaders to results (ideally quick wins)
- helps me communicate my ideas clearly
- clear indicators to measure progress
- leads to better collaboration
- concrete tips (e.g., to reduce risk)

Nice to have –

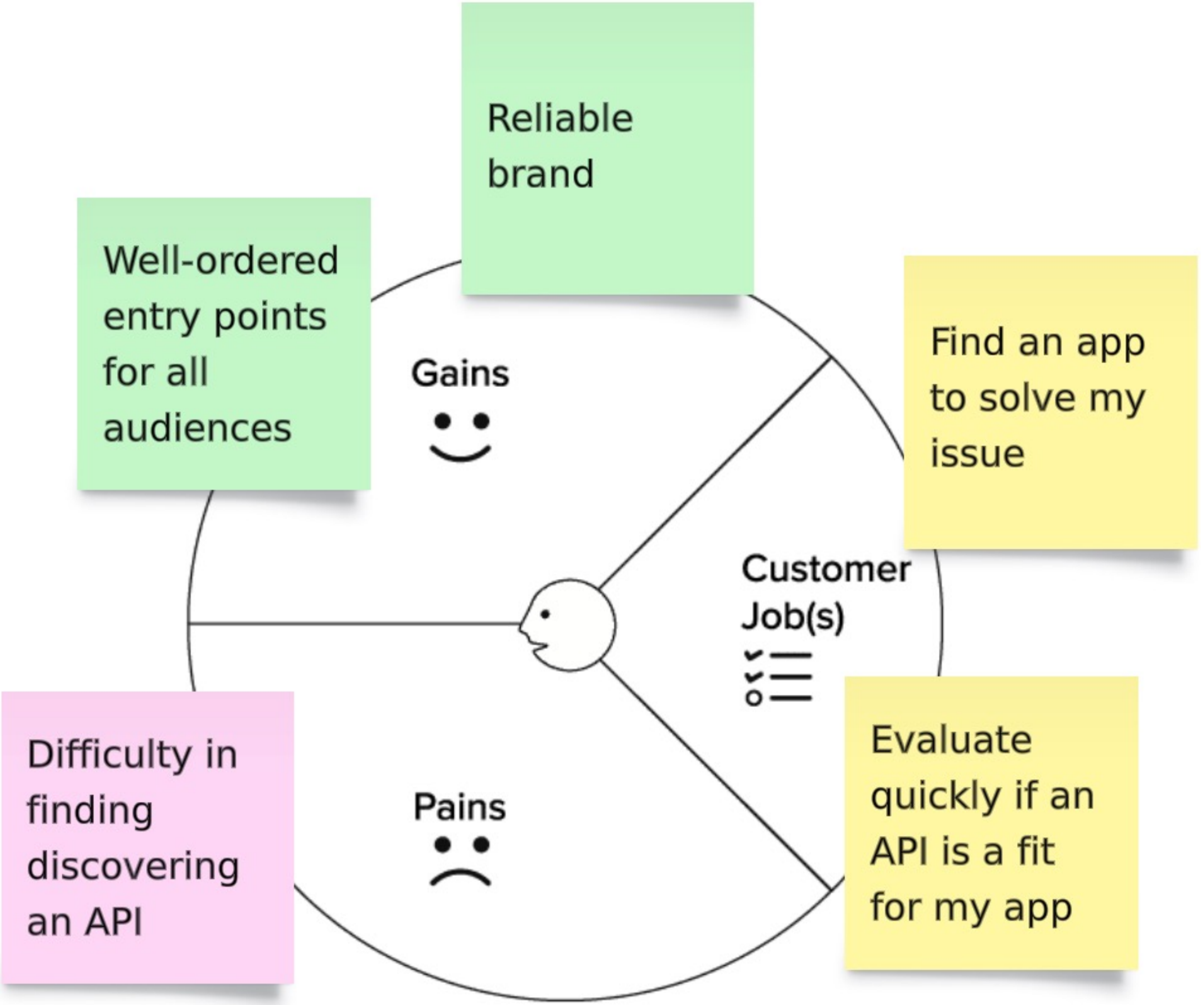
3.3.3 Create Content for Developer Journey

1. Discover
2. Evaluate
3. Get started
4. Build
5. Maintain
6. Celebrate

Discover + Evaluate

Developers' questions:

- How does this API solve my specific task?
- Can I trust it?
- Is pricing a barrier?



Reliable brand

Well-ordered entry points for all audiences

Gains

Find an app to solve my issue

Customer Job(s)

Evaluate quickly if an API is a fit for my app

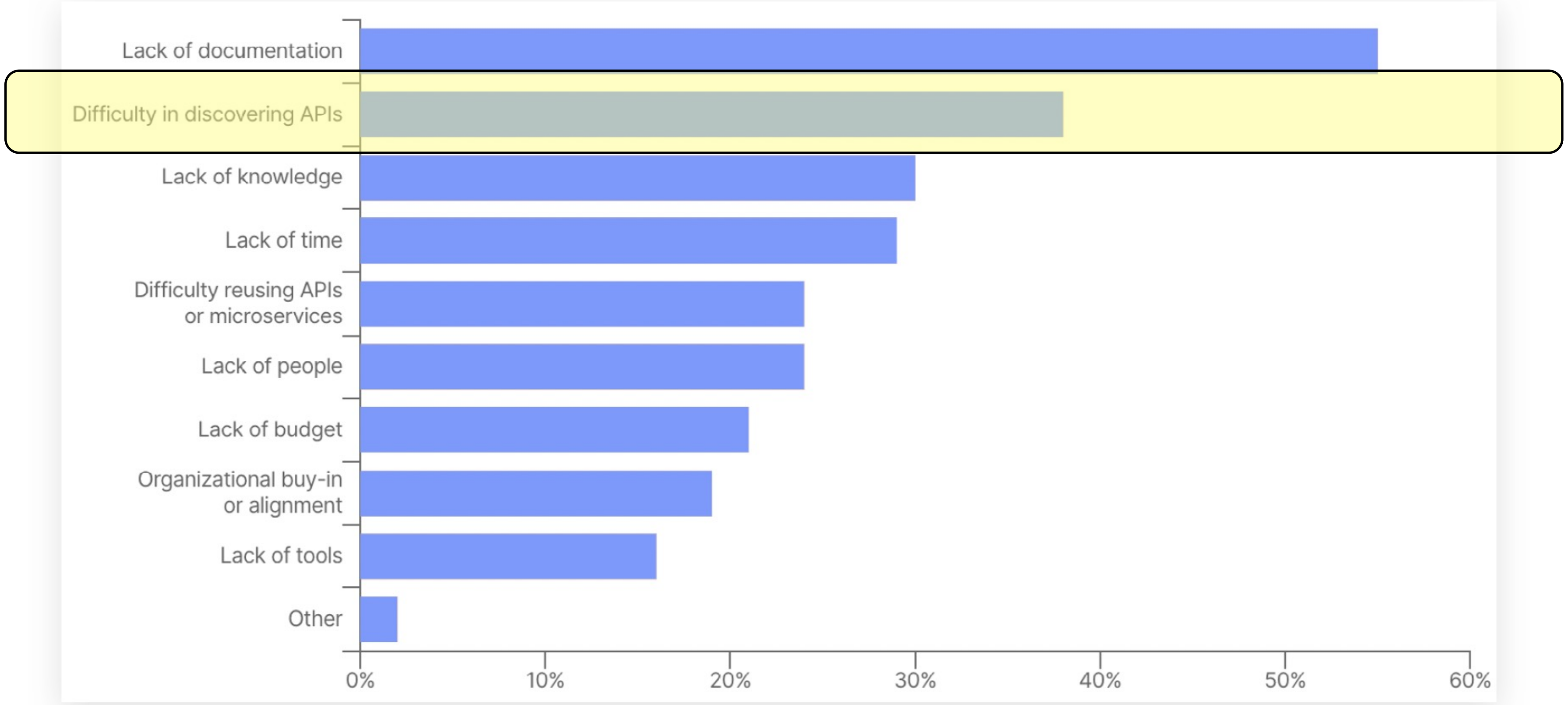
Difficulty in finding discovering an API

Pains

Discover + Evaluate

Issue

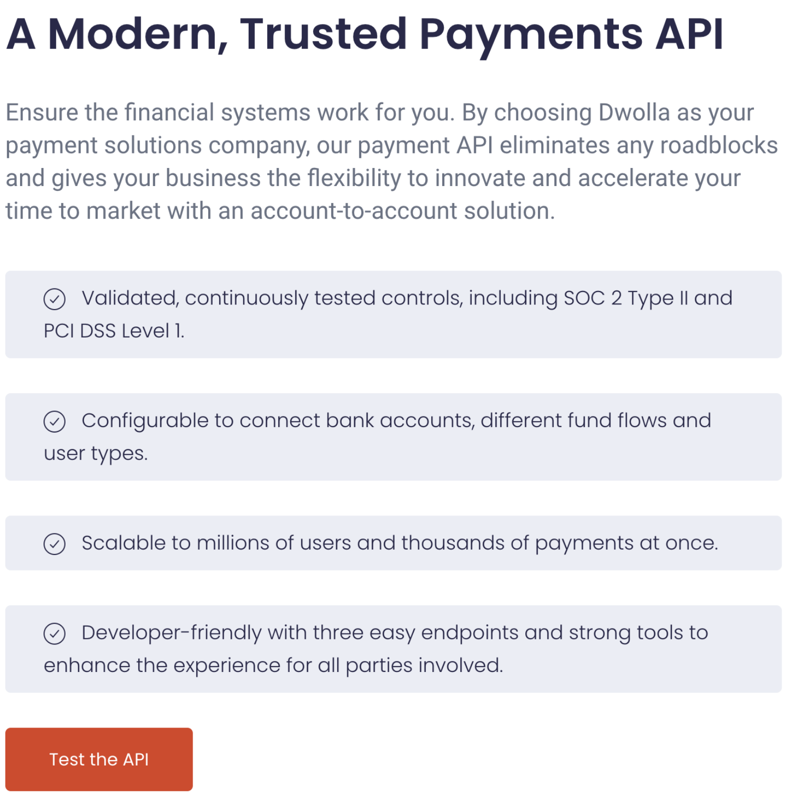
Obstacles to consuming APIs



Source: 2022 state of the API report

Discover + Evaluate

Landing page



A Modern, Trusted Payments API

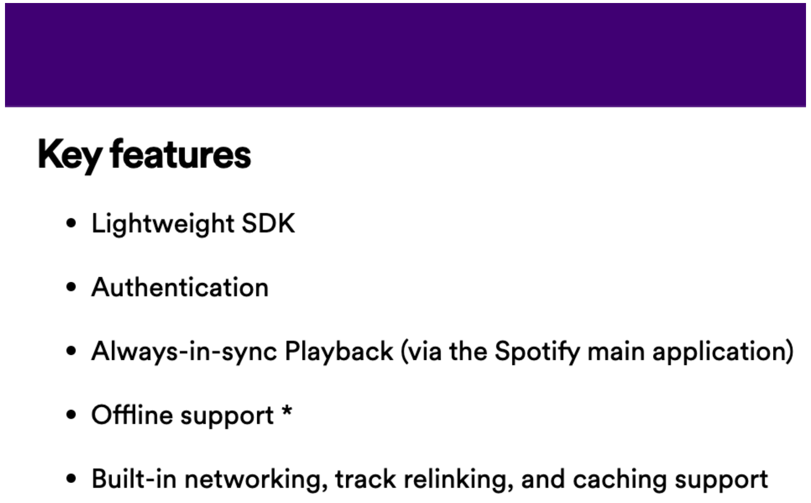
Ensure the financial systems work for you. By choosing Dwolla as your payment solutions company, our payment API eliminates any roadblocks and gives your business the flexibility to innovate and accelerate your time to market with an account-to-account solution.

✓ Validated, continuously tested controls, including SOC 2 Type II and PCI DSS Level 1.

✓ Configurable to connect bank accounts, different fund flows and user types.

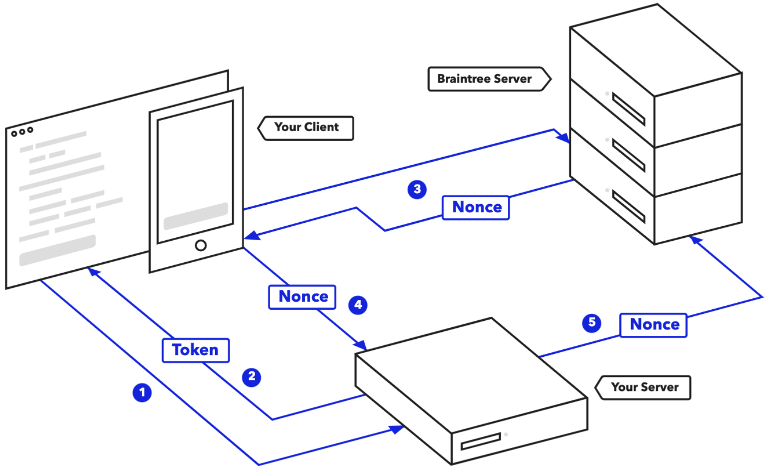
✓ Scalable to millions of users and thousands of payments at once.

✓ Developer-friendly with three easy endpoints and strong tools to enhance the experience for all parties involved.



Key features

- Lightweight SDK
- Authentication
- Always-in-sync Playback (via the Spotify main application)
- Offline support *
- Built-in networking, track relinking, and caching support



- Product intro
- Feature intro
- Implementation intro

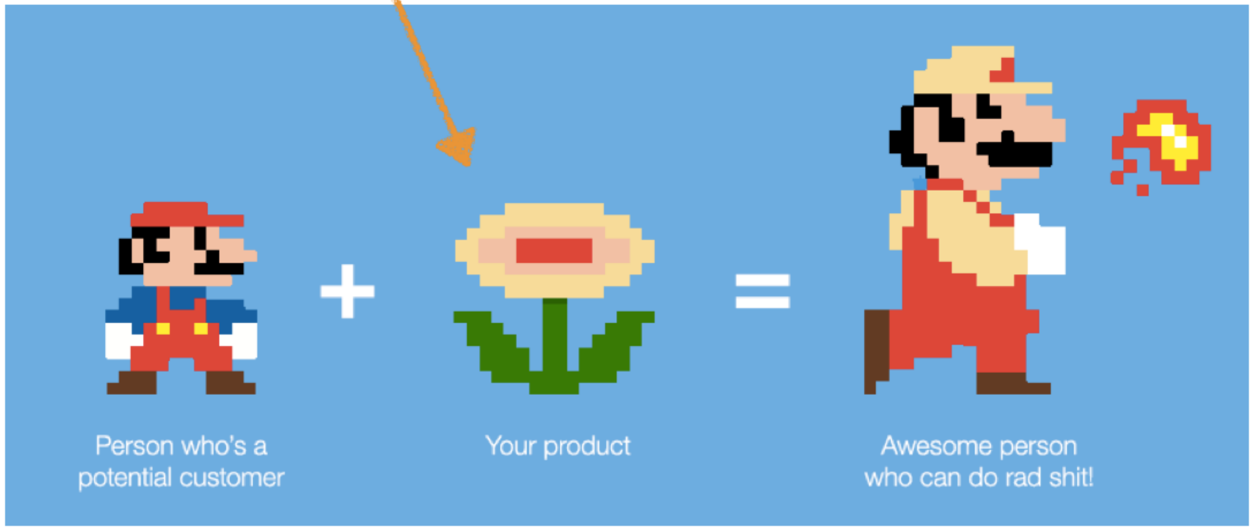
Discover + Evaluate

Q: Why do developers have difficulty in finding APIs?

A: Because your content does not contain the info that developers care most about, i.e., lack of contexts

- What developers' issues do this API fix?
- Why should developers use it?

This isn't what your business makes



+
=
Person who's a potential customer
Your product
Awesome person who can do rad shit!

Show benefits, not features

This is

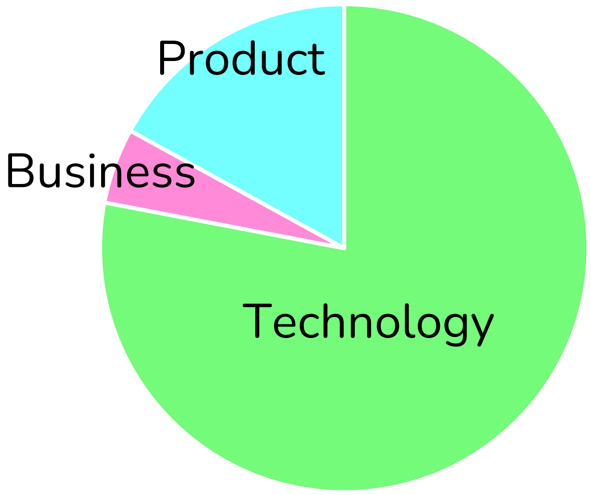
Discover + Evaluate

Content issues

- Focus most on the technical aspects (How)
- Not enough on the product's capabilities (What, Why)
- Business is always under-documented (What, Why)

Content solutions:

focus more on these



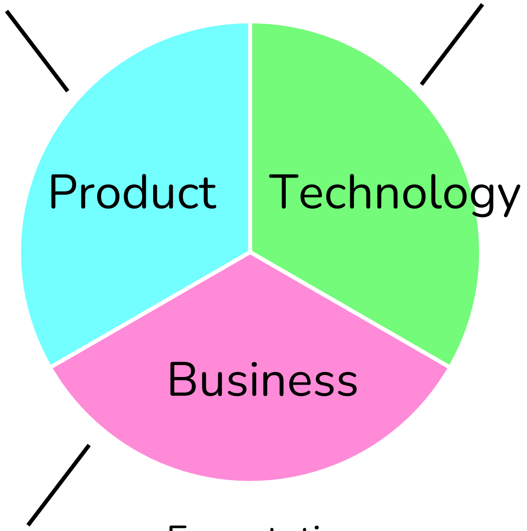
Reality

How deverlopers evaluate

APIs before using them

- What does it offer?
- What capabilities?
- What use cases does it cater to?

- How to call an API?
- How to implement it?
- API specs and paras



Expectation

- Business rules
- Pricing info
- Usage policy (SLA, security, legal, privacy, partner...)

Discover + Evaluate

Solution 1: developer messaging (Product "What")

It is different than messaging to consumers or businesses.

🔵 Keep messaging developer-friendly.

❑ Developers don't want to be marketed to.

❑ Messaging needs to be practical and speak to developers' needs.

❑ Developers want to understand very quickly if an API is for them.

🔵 Messaging should help developers decipher questions.

❑ What does API do?

❑ Why should a developer use it?

❑ What advantages does it have over competitive offers?

❑ How does it make a developer's life easier or better?

An easier way to develop!

Ineffective in enticing developers because it fails to answer many questions and brings up more questions 🤔

- What type of development is easier?
- Define easier?
- What part of the development process is easier, all of it?
- Why is it easier?

Discover + Evaluate

Solution 1: developer messaging (Product "What")

RULE OF THUMB

💡 **Best practice**

✅ Refrain from using superlatives around benefits.

✅ Messaging: 80% features + 20% benefits.

✅ Use "you" rather than "we".

Developers don't care how good you say you are. They care about what you can do for them.

- "We are trustworthy", "We have the best API", "We deliver..." ❌
- "You will be able to do xxx with our xxx API" ☑️

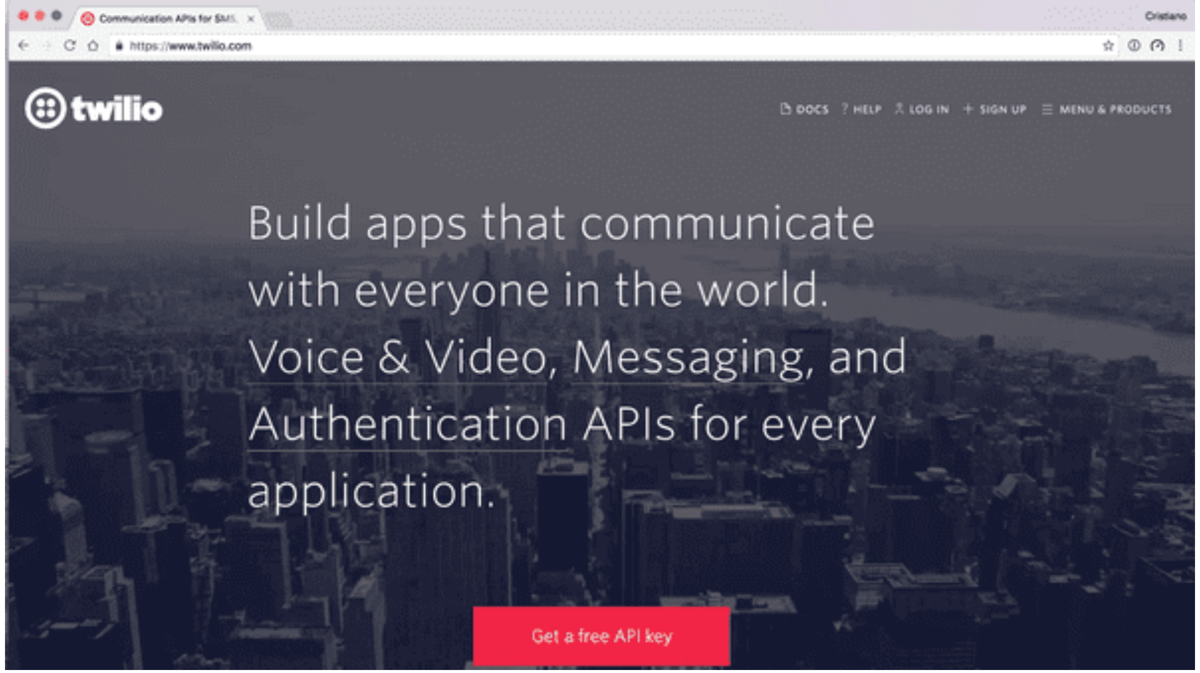
✅ Show your unique selling point (USP).

If product space is commoditized, try to differentiate on the **service** that is wrapped around your API.

e.g., great DX, comprehensive docs, timely technical support, or marquee customers.

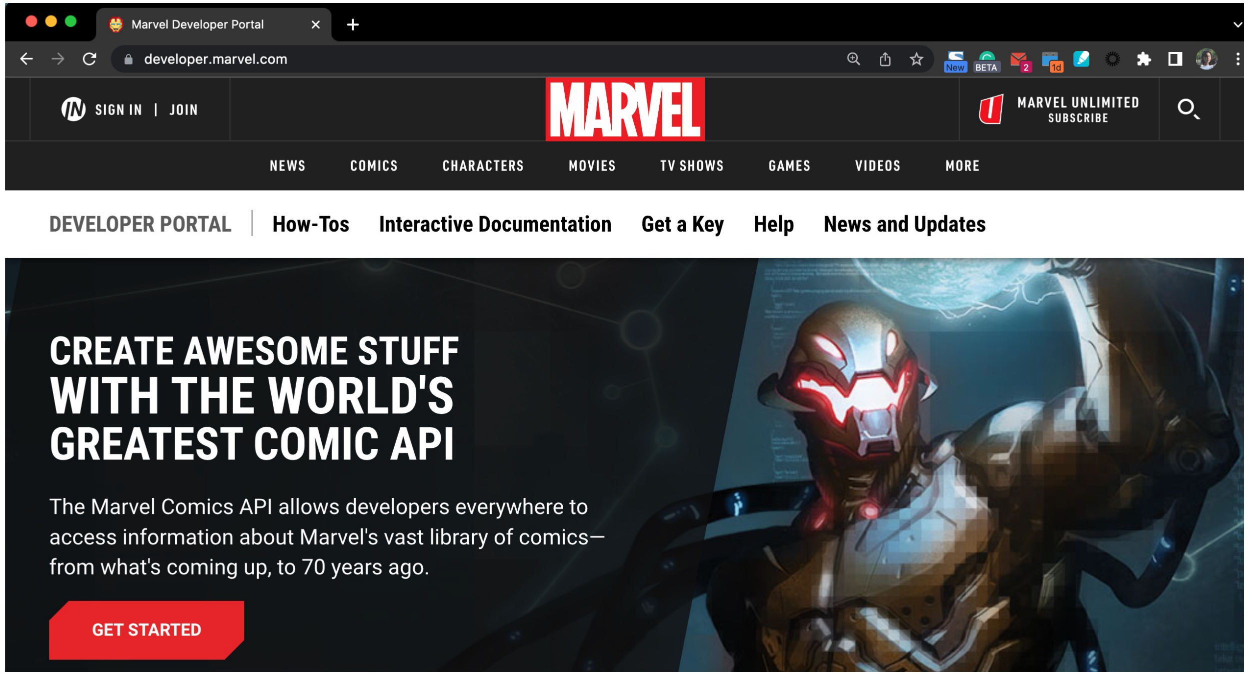
Discover + Evaluate

Example: developer messaging (Product "What")



Build apps that communicate with everyone in the world. Voice & Video, Messaging, and Authentication APIs for every application.

Twillo



CREATE AWESOME STUFF WITH THE WORLD'S GREATEST COMIC API

The Marvel Comics API allows developers everywhere to access information about Marvel's vast library of comics—from what's coming up, to 70 years ago.

Marvel

Discover + Evaluate

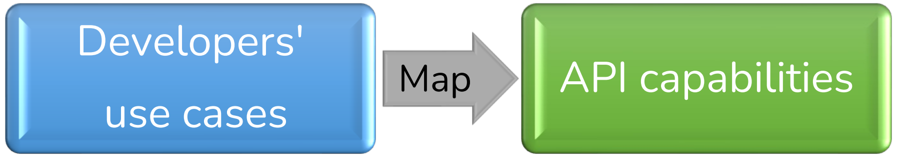
Solution 2: use case (Product "Why")

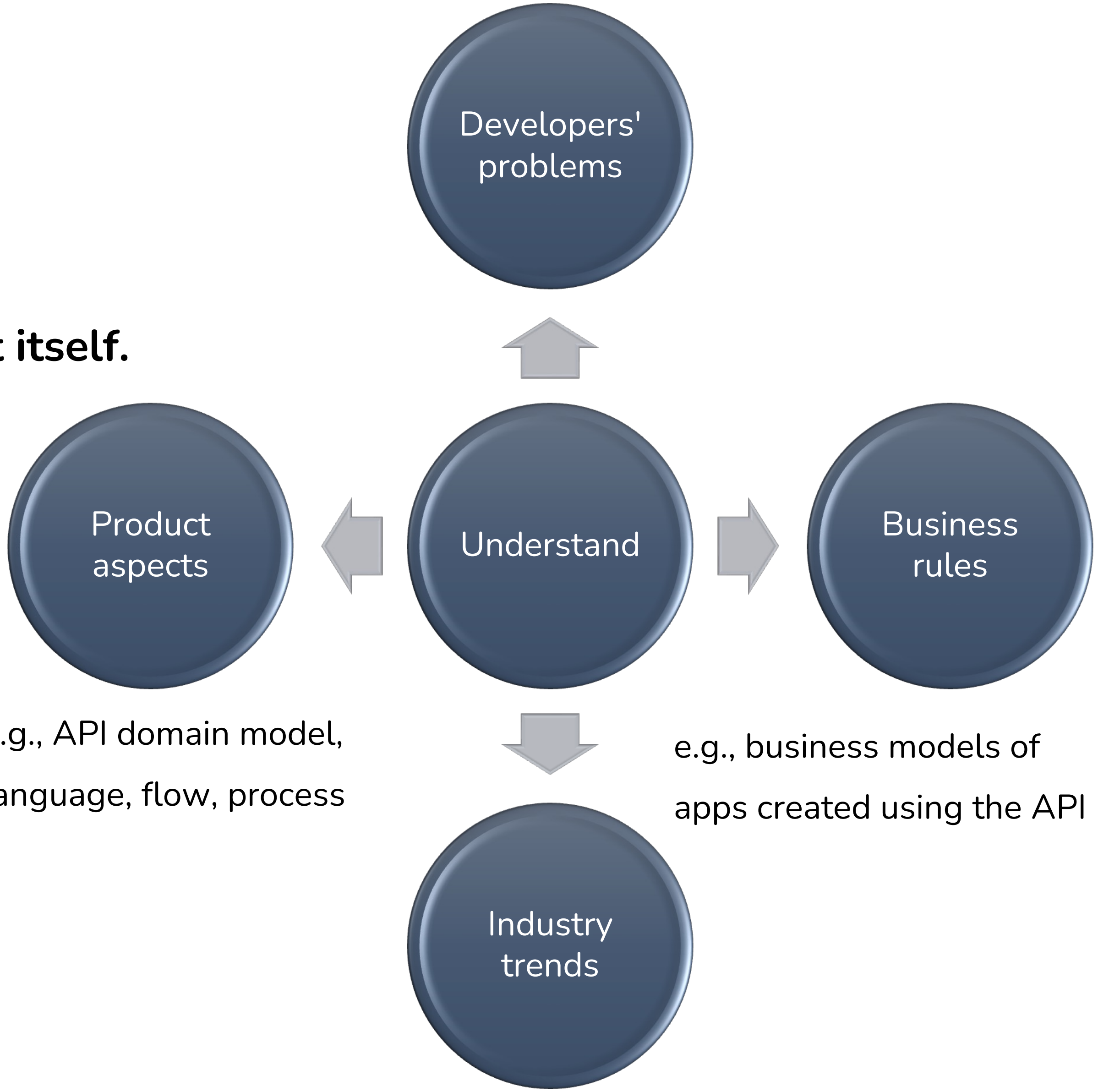
❑ **Developers care about use cases, not the product itself.**

Example

- Feature: Payment API
- Use case: collect money easily and securely

❑ Pre-analyze all for developers in advance.





e.g., API domain model, language, flow, process

e.g., business models of apps created using the API

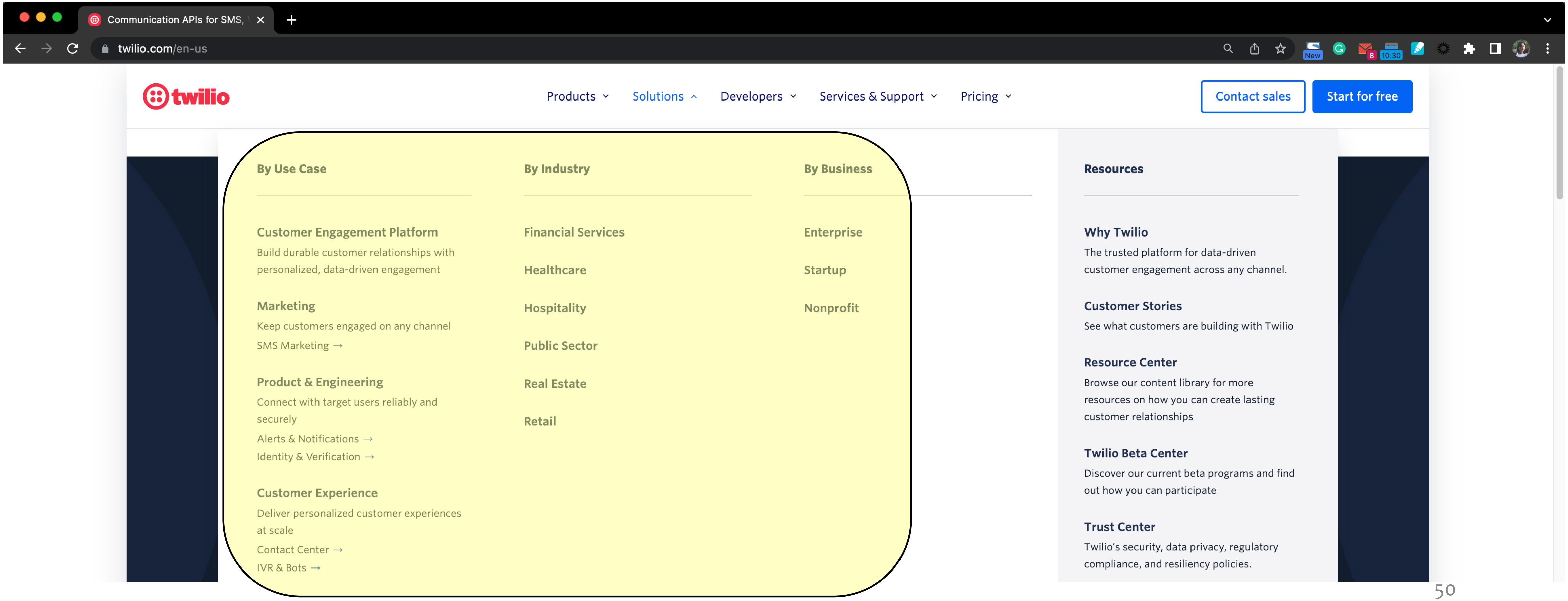
❑ Benefits of use cases

- Show business value directly
- Unlock developers' productivity
- Extend developers' imaginations

Discover + Evaluate

Example: solutions + use cases (Product "Why")

Twillo



By Use Case

Customer Engagement Platform
Build durable customer relationships with personalized, data-driven engagement

Marketing
Keep customers engaged on any channel
SMS Marketing →

Product & Engineering
Connect with target users reliably and securely
Alerts & Notifications →
Identity & Verification →

Customer Experience
Deliver personalized customer experiences at scale
Contact Center →
IVR & Bots →

By Industry

Financial Services
Healthcare
Hospitality
Public Sector
Real Estate
Retail

By Business

Enterprise
Startup
Nonprofit

Resources

Why Twilio
The trusted platform for data-driven customer engagement across any channel.

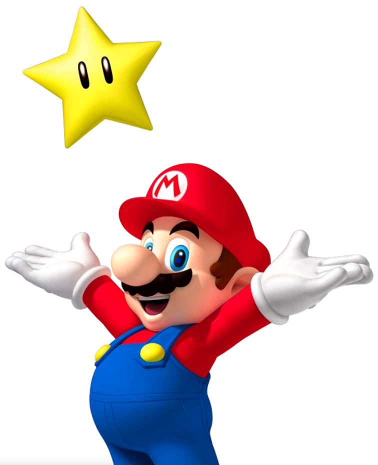
Customer Stories
See what customers are building with Twilio

Resource Center
Browse our content library for more resources on how you can create lasting customer relationships

Twilio Beta Center
Discover our current beta programs and find out how you can participate

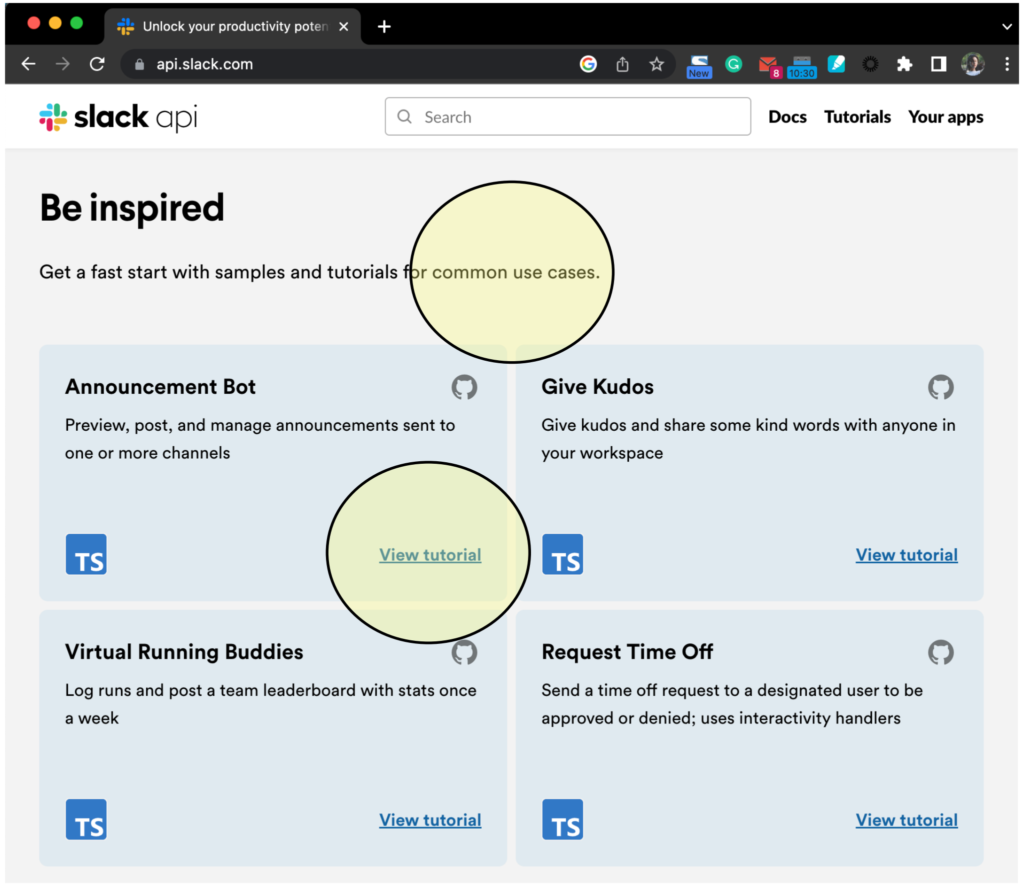
Trust Center
Twilio's security, data privacy, regulatory compliance, and resiliency policies.

Discover + Evaluate



Example: use cases tutorials (Product "Why")

Slack



Be inspired

Get a fast start with samples and tutorials for common use cases.

Announcement Bot

Preview, post, and manage announcements sent to one or more channels

TS View tutorial

Give Kudos

Give kudos and share some kind words with anyone in your workspace

TS View tutorial

Virtual Running Buddies

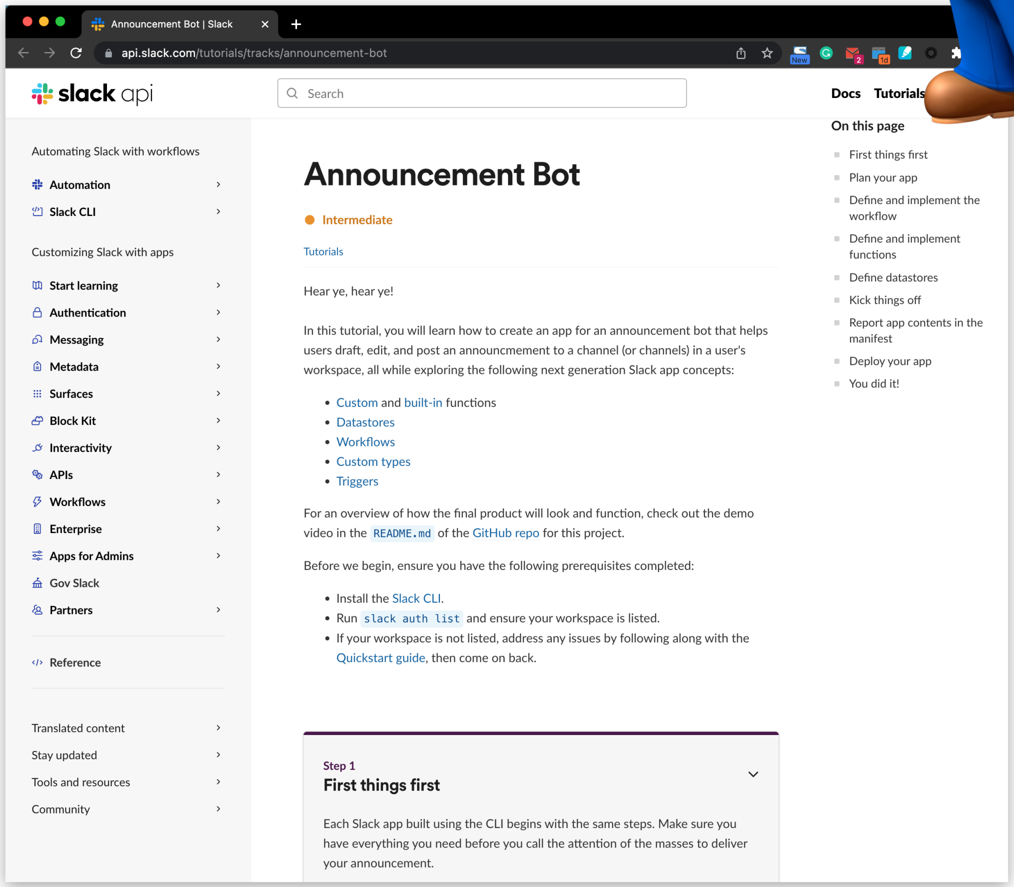
Log runs and post a team leaderboard with stats once a week

TS View tutorial

Request Time Off

Send a time off request to a designated user to be approved or denied; uses interactivity handlers

TS View tutorial



Announcement Bot

● Intermediate

Tutorials

Hear ye, hear ye!

In this tutorial, you will learn how to create an app for an announcement bot that helps users draft, edit, and post an announcmement to a channel (or channels) in a user's workspace, all while exploring the following next generation Slack app concepts:

- Custom and built-in functions
- Datastores
- Workflows
- Custom types
- Triggers

For an overview of how the final product will look and function, check out the demo video in the `README.md` of the GitHub repo for this project.

Before we begin, ensure you have the following prerequisites completed:

- Install the Slack CLI.
- Run `slack auth list` and ensure your workspace is listed.
- If your workspace is not listed, address any issues by following along with the Quickstart guide, then come on back.

Step 1

First things first

Each Slack app built using the CLI begins with the same steps. Make sure you have everything you need before you call the attention of the masses to deliver your announcement.

On this page

- First things first
- Plan your app
- Define and implement the workflow
- Define and implement functions
- Define datastores
- Kick things off
- Report app contents in the manifest
- Deploy your app
- You did it!

Discover + Evaluate

Summary

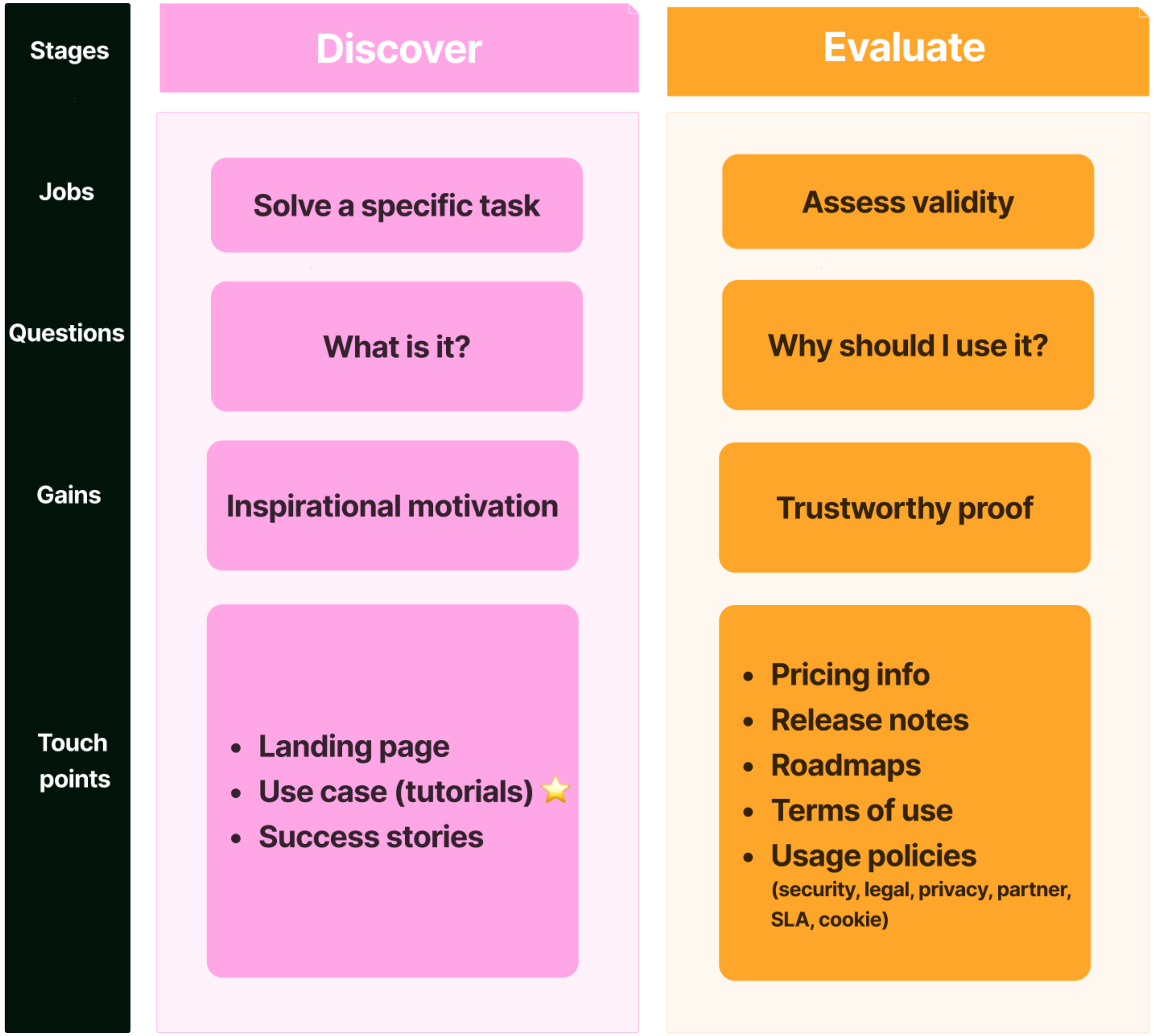
Best practice

Dos

- Ignite developers' imaginations to create and find commercial success.
- Focus on real-world problem-solving, not product promises.

Don'ts

- Outrageous marketing pitch.
- Exhaustive technical details.



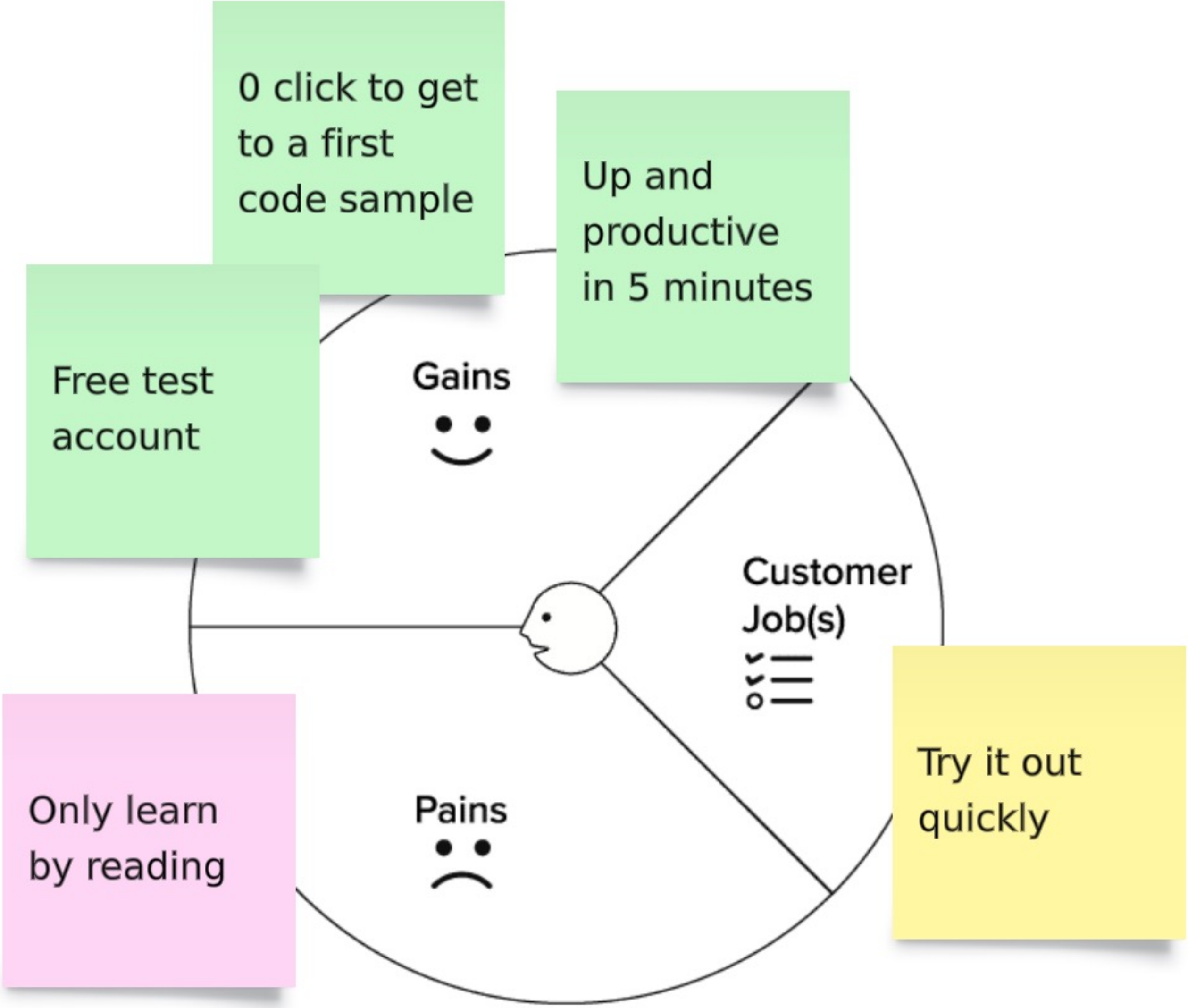
Stages	Discover	Evaluate
Jobs	Solve a specific task	Assess validity
Questions	What is it?	Why should I use it?
Gains	Inspirational motivation	Trustworthy proof
Touch points	• Landing page • Use case (tutorials) ⭐ • Success stories	• Pricing info • Release notes • Roadmaps • Terms of use • Usage policies (security, legal, privacy, partner, SLA, cookie)

1. Discover
2. Evaluate
3. **Get started**
4. Build
5. Maintain
6. Celebrate

Get started

Developers' questions:

- Where do I start?
- Does it provide fast try-out and test options?
- Can I get "Hello World" in 3 minutes?



Get started

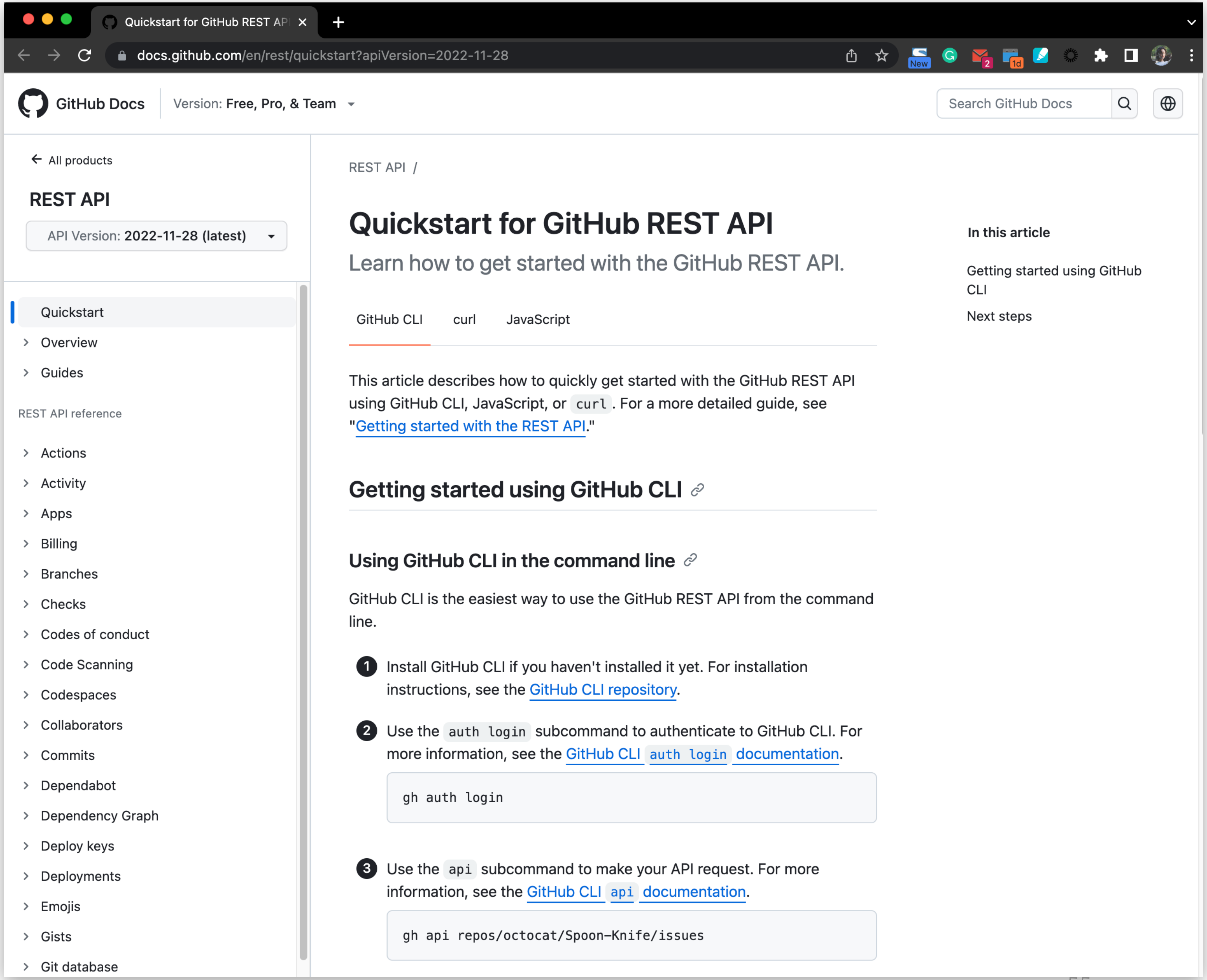
Issues

- Developers need to prepare

  test environments and materials

- Static learning

  Struggle to keep concentration



Quickstart for GitHub REST API

Learn how to get started with the GitHub REST API.

This article describes how to quickly get started with the GitHub REST API using GitHub CLI, JavaScript, or `curl`. For a more detailed guide, see "Getting started with the REST API."

Getting started using GitHub CLI

Using GitHub CLI in the command line

GitHub CLI is the easiest way to use the GitHub REST API from the command line.

1. Install GitHub CLI if you haven't installed it yet. For installation instructions, see the GitHub CLI repository.
2. Use the `auth login` subcommand to authenticate to GitHub CLI. For more information, see the GitHub CLI `auth login` documentation.

```
gh auth login
```

3. Use the `api` subcommand to make your API request. For more information, see the GitHub CLI `api` documentation.

```
gh api repos/octocat/Spoon-Knife/issues
```

Get started

Content solutions - design context "Where"

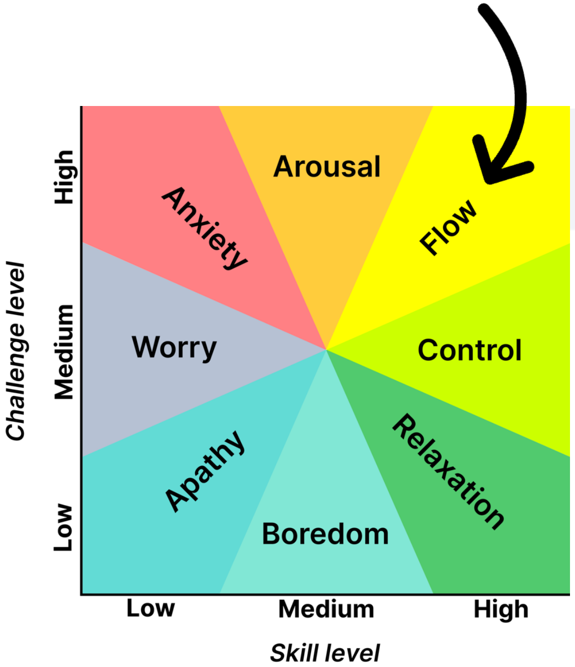
- Find developers' flow status
- Create an interactive learning environment

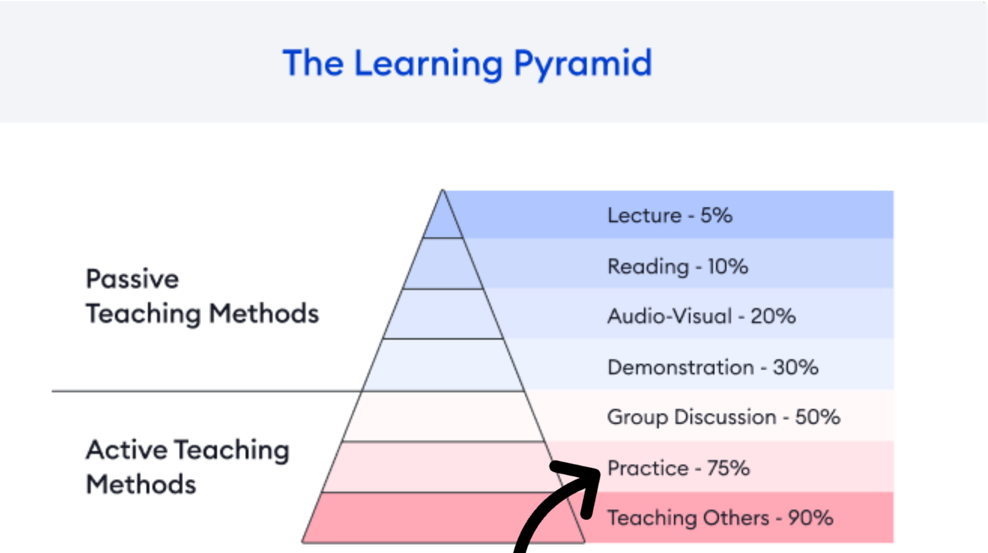
UX in games: satisfying

We like playing games because experiencing the flow of focus and stimuli creates a pleasant feeling.

Flow trigger

- Challenge–skill balance
- Clear goals & immediate feedback
- Concentration focused on one thing only
- Learn by doing





The Learning Pyramid

Passive Teaching Methods

Active Teaching Methods

Lecture - 5%

Reading - 10%

Audio-Visual - 20%

Demonstration - 30%

Group Discussion - 50%

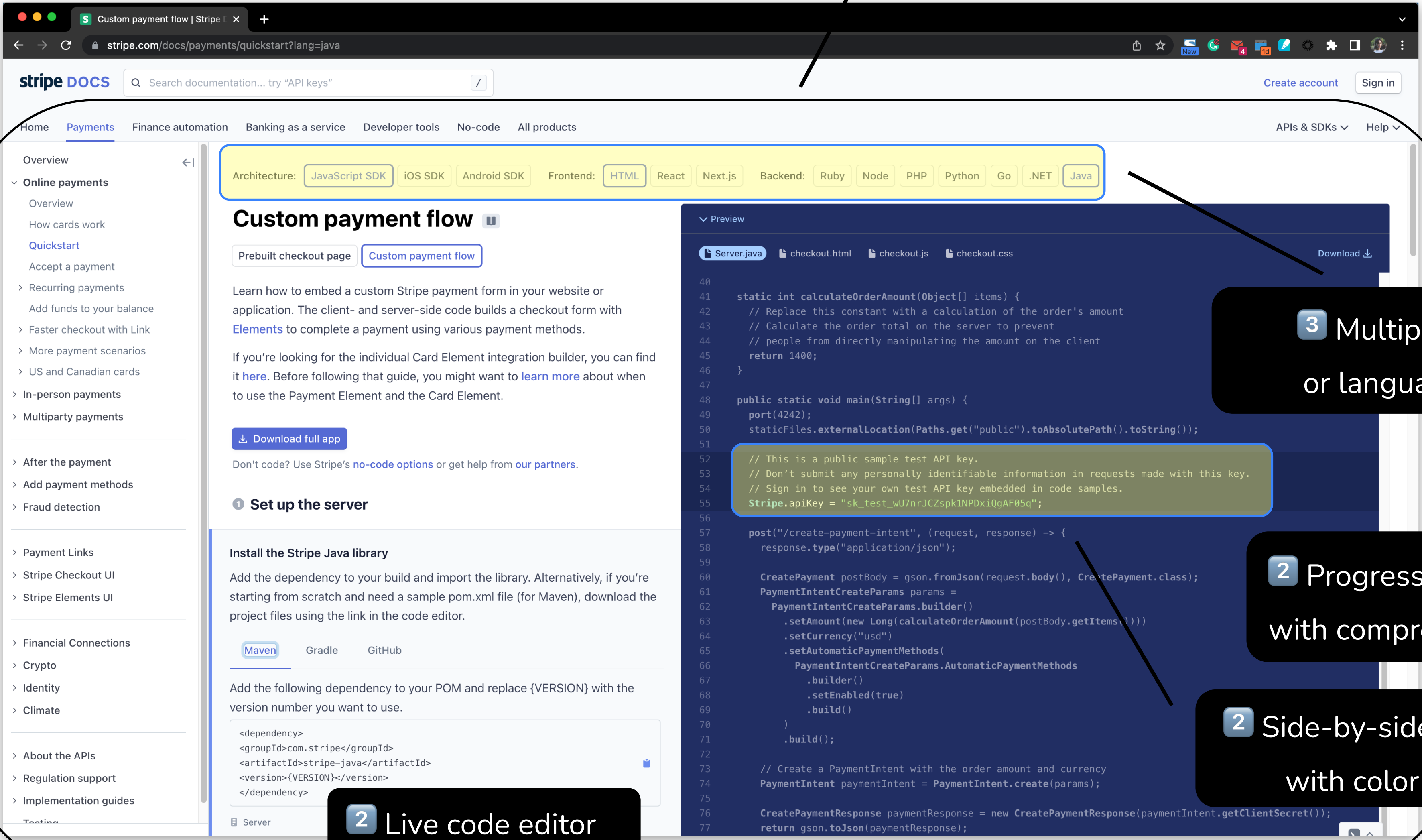
Practice - 75%

Teaching Others - 90%

Get started

Example

Stripe



Custom payment flow | Stripe D

stripe.com/docs/payments/quickstart?lang=java

stripe DOCS

Search documentation... try "API keys"

Create account · Sign in

Home · Payments · Finance automation · Banking as a service · Developer tools · No-code · All products · APIs & SDKs · Help

Overview
- Online payments
  - Overview
  - How cards work
  - Quickstart
  - Accept a payment
  - Recurring payments
  - Add funds to your balance
  - Faster checkout with Link
  - More payment scenarios
  - US and Canadian cards
- In-person payments
- Multiparty payments
- After the payment
- Add payment methods
- Fraud detection
- Payment Links
- Stripe Checkout UI
- Stripe Elements UI
- Financial Connections
- Crypto
- Identity
- Climate
- About the APIs
- Regulation support
- Implementation guides

Architecture: JavaScript SDK · iOS SDK · Android SDK   Frontend: HTML · React · Next.js   Backend: Ruby · Node · PHP · Python · Go · .NET · Java

Custom payment flow

Prebuilt checkout page · Custom payment flow

Learn how to embed a custom Stripe payment form in your website or application. The client- and server-side code builds a checkout form with Elements to complete a payment using various payment methods.

If you're looking for the individual Card Element integration builder, you can find it here. Before following that guide, you might want to learn more about when to use the Payment Element and the Card Element.

Download full app

Don't code? Use Stripe's no-code options or get help from our partners.

1 Set up the server

Install the Stripe Java library

Add the dependency to your build and import the library. Alternatively, if you're starting from scratch and need a sample pom.xml file (for Maven), download the project files using the link in the code editor.

Maven · Gradle · GitHub

Add the following dependency to your POM and replace {VERSION} with the version number you want to use.

```
<dependency>
<groupId>com.stripe</groupId>
<artifactId>stripe-java</artifactId>
<version>{VERSION}</version>
</dependency>
```

Server

Preview

Server.java · checkout.html · checkout.js · checkout.css · Download

```

  static int calculateOrderAmount(Object[] items) {
    // Replace this constant with a calculation of the order's amount
    // Calculate the order total on the server to prevent
    // people from directly manipulating the amount on the client
    return 1400;
  }

  public static void main(String[] args) {
    port(4242);
    staticFiles.externalLocation(Paths.get("public").toAbsolutePath().toString());

    // This is a public sample test API key.
    // Don’t submit any personally identifiable information in requests made with this key.
    // Sign in to see your own test API key embedded in code samples.
    Stripe.apiKey = "sk_test_wU7nrJCZspk1NPDxiQgAF05q";

    post("/create-payment-intent", (request, response) -> {
      response.type("application/json");

      CreatePayment postBody = gson.fromJson(request.body(), CreatePayment.class);
      PaymentIntentCreateParams params =
        PaymentIntentCreateParams.builder()
          .setAmount(new Long(calculateOrderAmount(postBody.getItems())))
          .setCurrency("usd")
          .setAutomaticPaymentMethods(
            PaymentIntentCreateParams.AutomaticPaymentMethods
              .builder()
              .setEnabled(true)
              .build()
          )
          .build();

      // Create a PaymentIntent with the order amount and currency
      PaymentIntent paymentIntent = PaymentIntent.create(params);

      CreatePaymentResponse paymentResponse = new CreatePaymentResponse(paymentIntent.getClientSecret());
      return gson.toJson(paymentResponse);
```

1 Interactive learning environment with sandbox

Frictionless onboarding experience

1 3-column layout

2 Live code editor

3 Multiple framework or language selectors

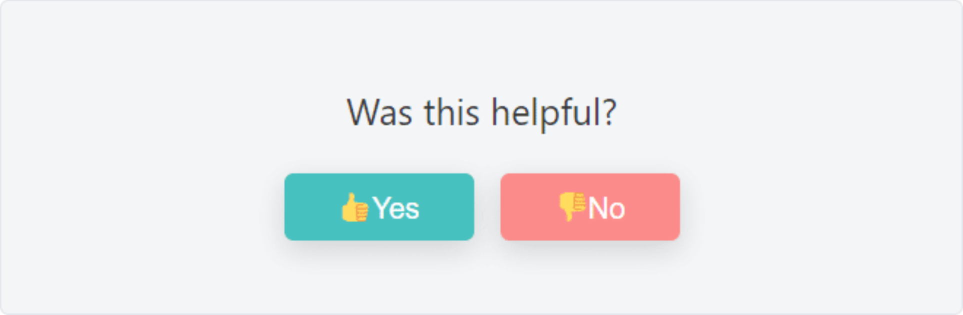
2 Progressive disclosure info with comprehensive contexts

2 Side-by-side code examples with color highlighted

Get started

Tip

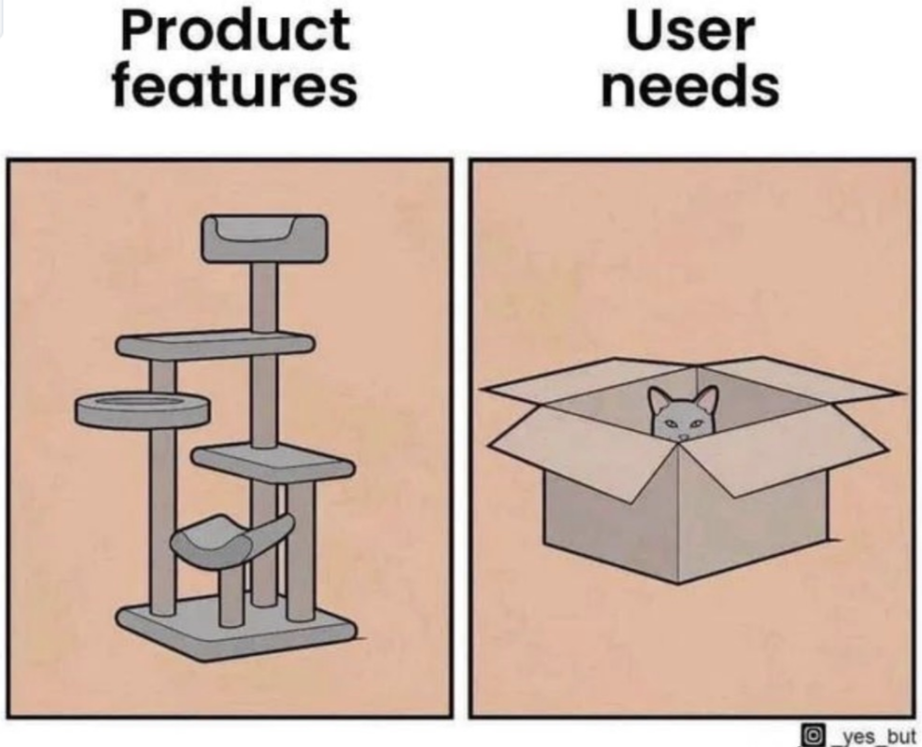
Many sites ask



Was this helpful?

How about asking:

- "Are you considering this API?"
- "Are you starting creating an app using this API?"



_yes_but

Get started

Summary

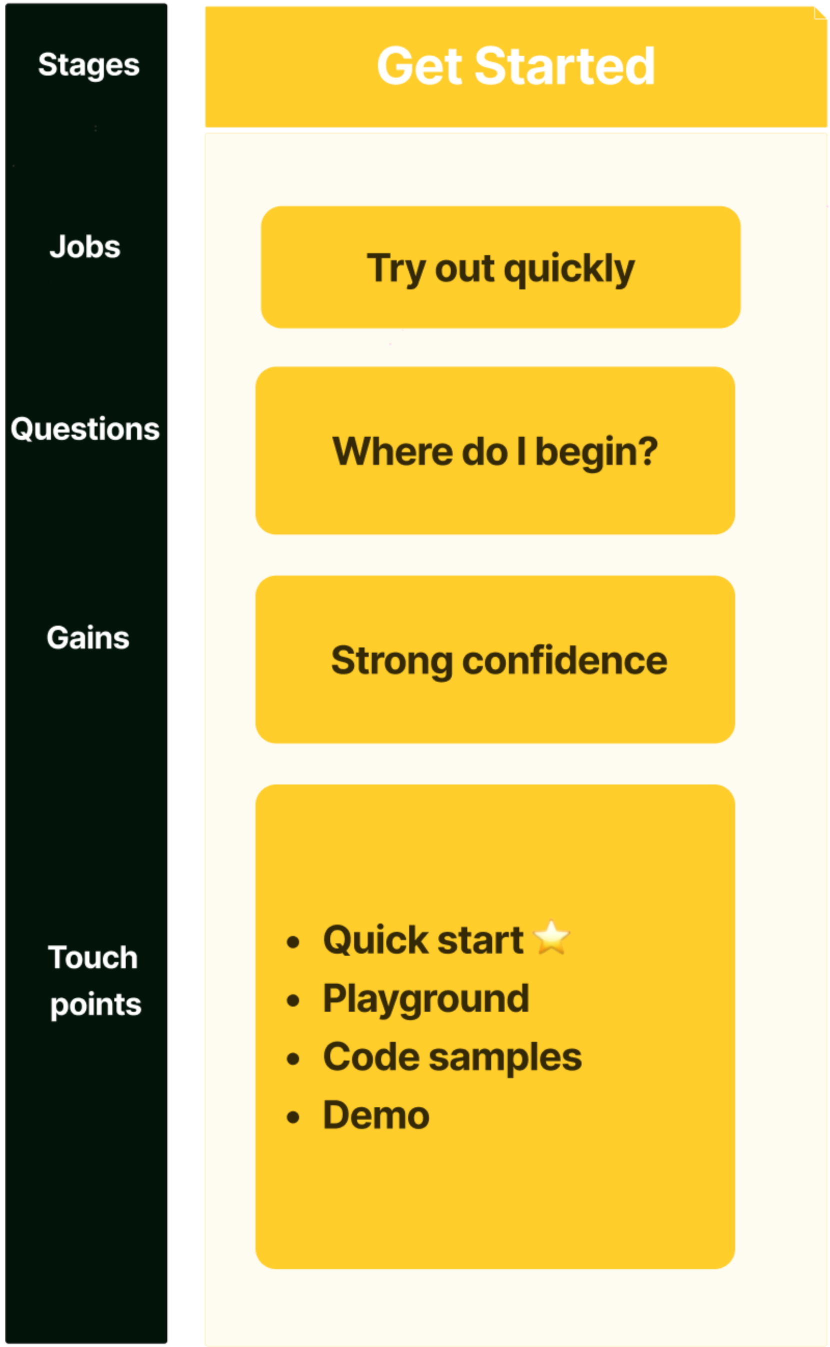
Best practice

Dos

- Get developers to "Hello World" as soon as possible with minimum steps and effort.
- Provide a sandbox environment to "kick the tire".

Don'ts

- Lengthy onboarding process with paid test accounts.
- Steep learning curve.



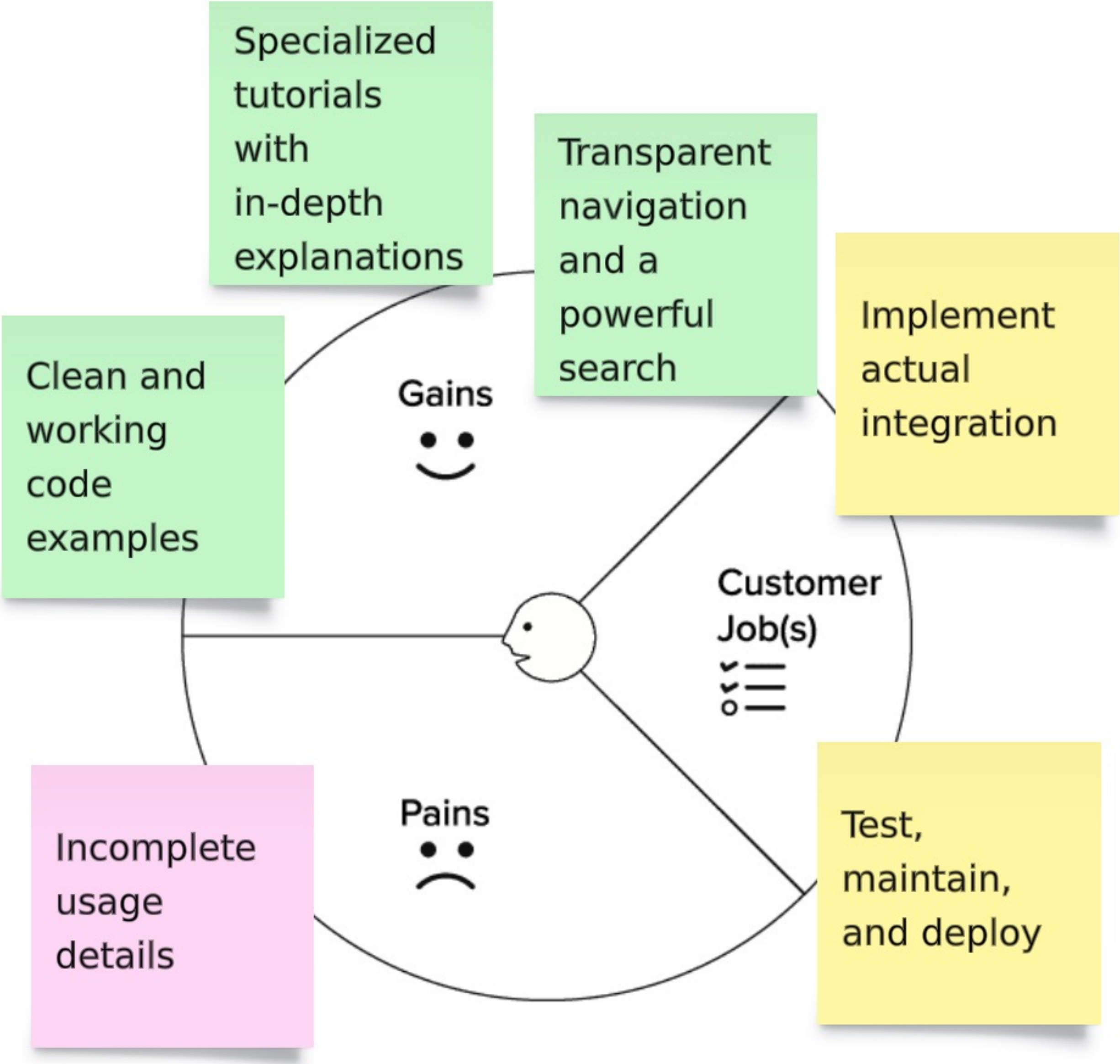
Stages	Get Started
Jobs	Try out quickly
Questions	Where do I begin?
Gains	Strong confidence
Touch points	• Quick start ⭐ • Playground • Code samples • Demo

1. Discover
2. Evaluate
3. Get started
4. Build
5. Maintain
6. Celebrate

Build + Maintain

Developers' questions:

- How to do X with Y?
- Is it easy to keep running?
- Is the support reliable?



Gains

Specialized tutorials with in-depth explanations

Transparent navigation and a powerful search

Clean and working code examples

Customer Job(s)

Implement actual integration

Test, maintain, and deploy

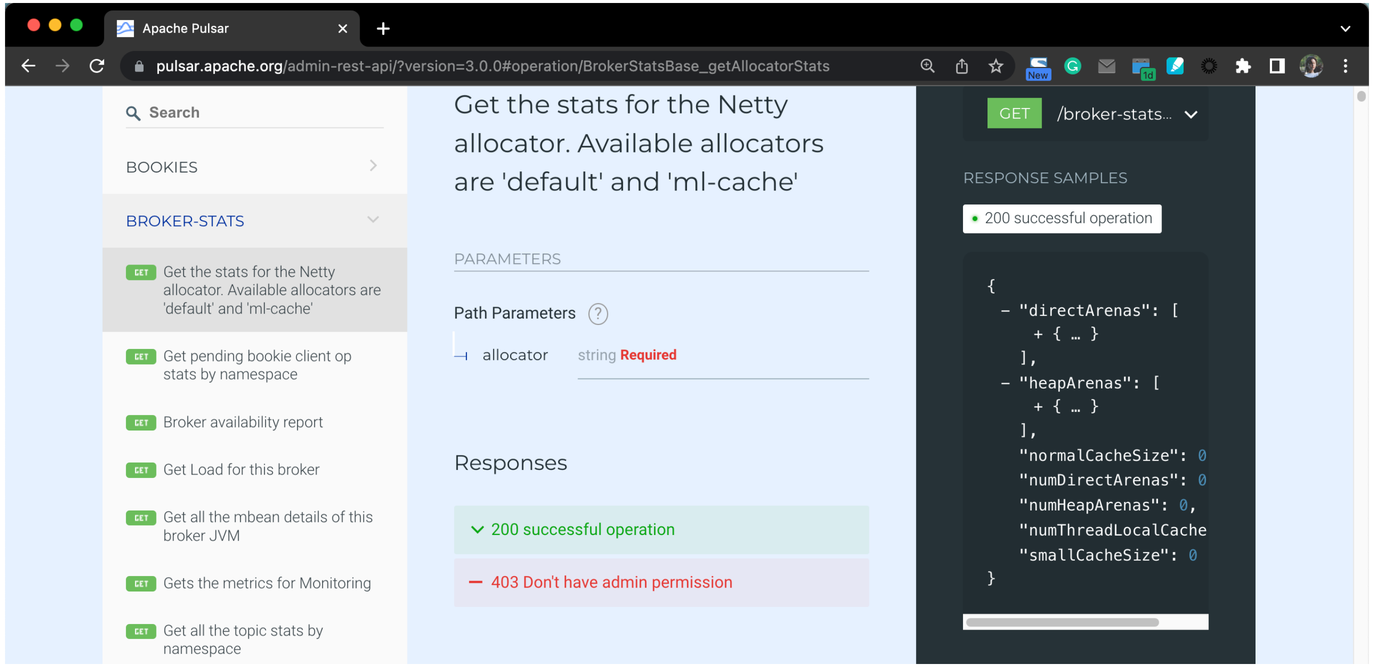
Pains

Incomplete usage details

Build + Maintain

Content issues

Lack usage details ("What, When, How")



Apache Pulsar
pulsar.apache.org/admin-rest-api/?version=3.0.0#operation/BrokerStatsBase_getAllocatorStats
Search
BOOKIES
BROKER-STATS
GET Get the stats for the Netty allocator. Available allocators are 'default' and 'ml-cache'
GET Get pending bookie client op stats by namespace
GET Broker availability report
GET Get Load for this broker
GET Get all the mbean details of this broker JVM
GET Gets the metrics for Monitoring
GET Get all the topic stats by namespace
GET
/broker-stats...

Get the stats for the Netty allocator. Available allocators are 'default' and 'ml-cache'

PARAMETERS

Path Parameters

allocator string Required

Responses

200 successful operation

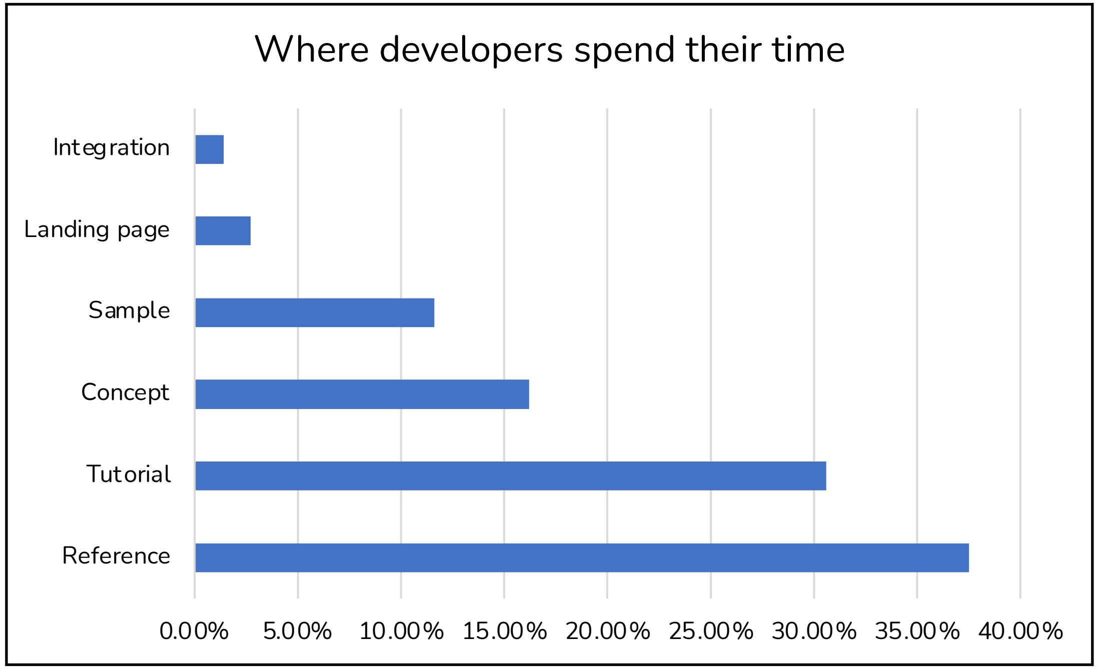
403 Don't have admin permission

RESPONSE SAMPLES

200 successful operation

```
{
  - "directArenas": [
      + { … }
    ],
  - "heapArenas": [
      + { … }
    ],
    "normalCacheSize": 0
    "numDirectArenas": 0
    "numHeapArenas": 0,
    "numThreadLocalCache
    "smallCacheSize": 0
}
```

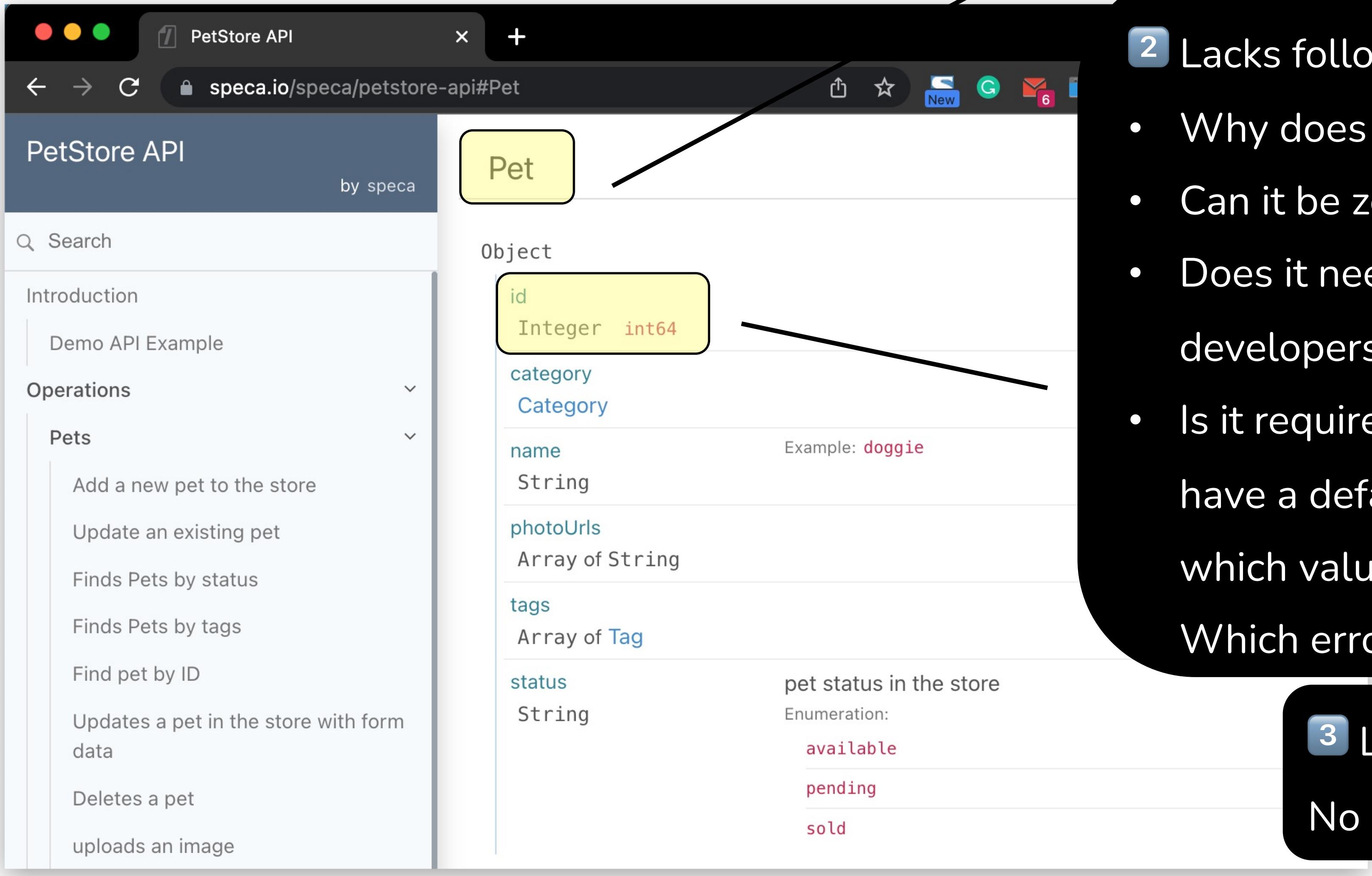
Pulsar API Reference



Source: 2019 API observation study

Build + Maintain

Issues – lack context "What, When, How"



PetStore API

speca.io/speca/petstore-api#Pet

PetStore API

by speca

Search

Introduction

Demo API Example

Operations

Pets

Add a new pet to the store

Update an existing pet

Finds Pets by status

Finds Pets by tags

Find pet by ID

Updates a pet in the store with form data

Deletes a pet

uploads an image

Pet

Object

id Integer int64	
category Category	
name String	Example: doggie
photoUrls Array of String	
tags Array of Tag	
status String	pet status in the store Enumeration: available pending sold

1️⃣ Lacks context "What"

What is it? Need to explain items that look "obvious".

Programming is nothing if not uncertainty.

2️⃣ Lacks following contexts:

- Why does the number need to be so large?
- Can it be zero or negative?
- Does it need to be unique? If yes, how do developers verify?
- Is it required or optional? If required, does it have a default value if is not specified? If yes, which value? If no, does it return an error? Which error code?

3️⃣ Lacks context "How"

No code examples

Build + Maintain

Solution: **thinking like a developer**

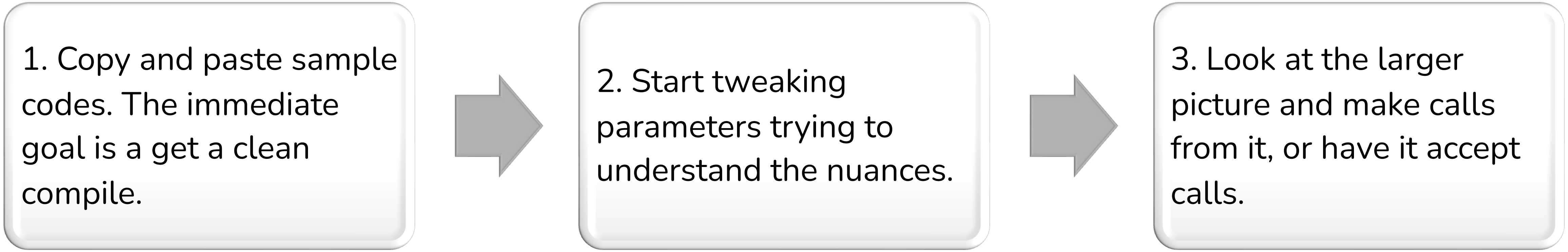
Understand what's important to developers,

anticipate their questions, and add those contexts.

Not thinking like a translator

Example

Process of playing with an API call



1. Copy and paste sample codes. The immediate goal is a get a clean compile.

2. Start tweaking parameters trying to understand the nuances.

3. Look at the larger picture and make calls from it, or have it accept calls.

Build + Maintain

Example

1 Simple description

Concise "What"

4 Code examples

Clear "How"

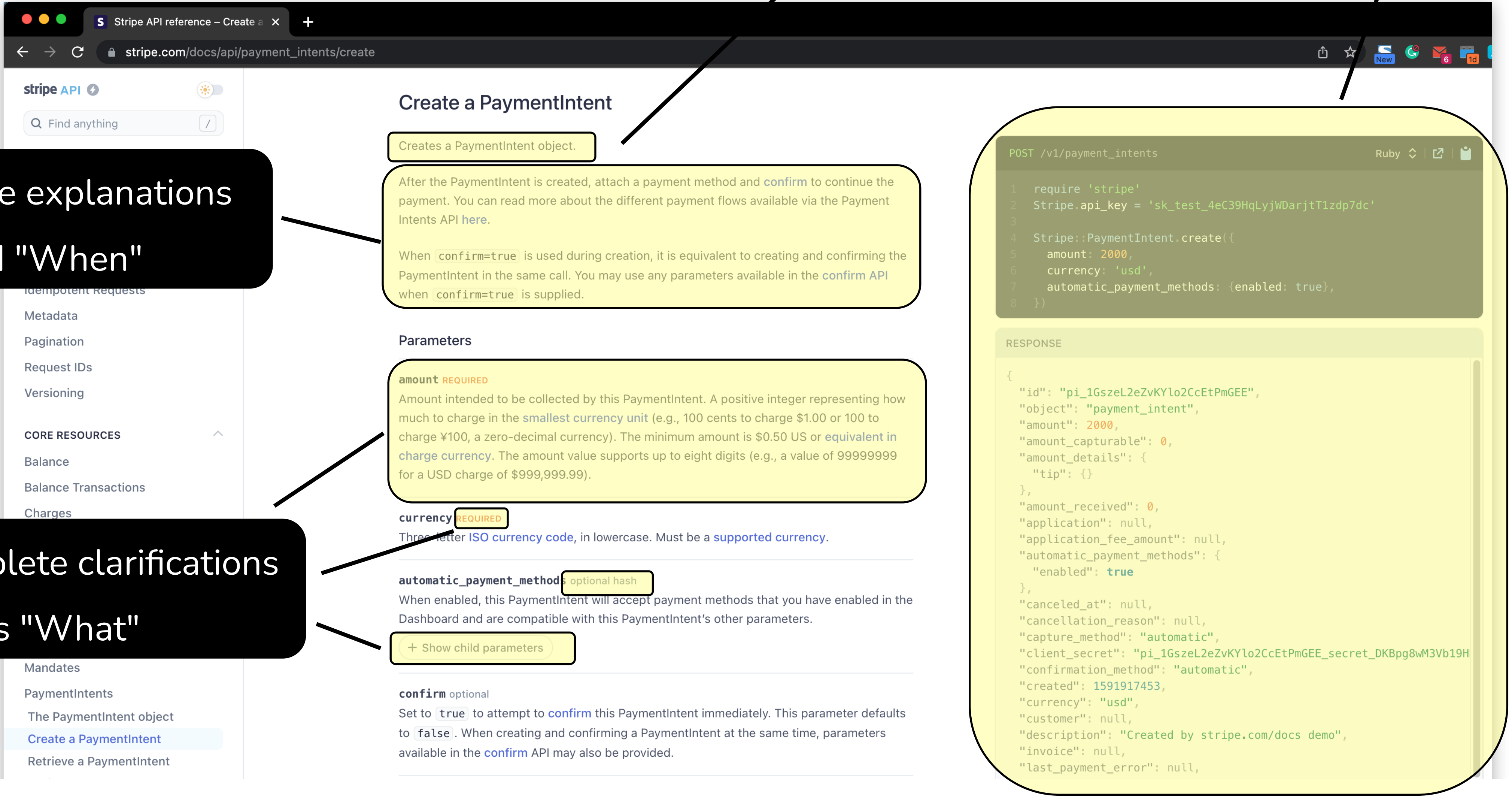
2 Usage explanations

Detailed "When"

Stripe

3 Complete clarifications

Rigorous "What"



Create a PaymentIntent

Creates a PaymentIntent object.

After the PaymentIntent is created, attach a payment method and confirm to continue the payment. You can read more about the different payment flows available via the Payment Intents API here.

When `confirm=true` is used during creation, it is equivalent to creating and confirming the PaymentIntent in the same call. You may use any parameters available in the confirm API when `confirm=true` is supplied.

Parameters

amount REQUIRED
Amount intended to be collected by this PaymentIntent. A positive integer representing how much to charge in the smallest currency unit (e.g., 100 cents to charge $1.00 or 100 to charge ¥100, a zero-decimal currency). The minimum amount is $0.50 US or equivalent in charge currency. The amount value supports up to eight digits (e.g., a value of 99999999 for a USD charge of $999,999.99).

currency REQUIRED
Three-letter ISO currency code, in lowercase. Must be a supported currency.

automatic_payment_methods optional hash
When enabled, this PaymentIntent will accept payment methods that you have enabled in the Dashboard and are compatible with this PaymentIntent's other parameters.

+ Show child parameters

confirm optional
Set to `true` to attempt to confirm this PaymentIntent immediately. This parameter defaults to `false`. When creating and confirming a PaymentIntent at the same time, parameters available in the confirm API may also be provided.

POST /v1/payment_intents   Ruby

```ruby
require 'stripe'
Stripe.api_key = 'sk_test_4eC39HqLyjWDarjtT1zdp7dc'

Stripe::PaymentIntent.create({
  amount: 2000,
  currency: 'usd',
  automatic_payment_methods: {enabled: true},
})
```

RESPONSE

```
{
  "id": "pi_1GszeL2eZvKYlo2CcEtPmGEE",
  "object": "payment_intent",
  "amount": 2000,
  "amount_capturable": 0,
  "amount_details": {
    "tip": {}
  },
  "amount_received": 0,
  "application": null,
  "application_fee_amount": null,
  "automatic_payment_methods": {
    "enabled": true
  },
  "canceled_at": null,
  "cancellation_reason": null,
  "capture_method": "automatic",
  "client_secret": "pi_1GszeL2eZvKYlo2CcEtPmGEE_secret_DKBpg8wM3Vb19H
  "confirmation_method": "automatic",
  "created": 1591917453,
  "currency": "usd",
  "customer": null,
  "description": "Created by stripe.com/docs demo",
  "invoice": null,
  "last_payment_error": null,
```

Build + Maintain

Summary

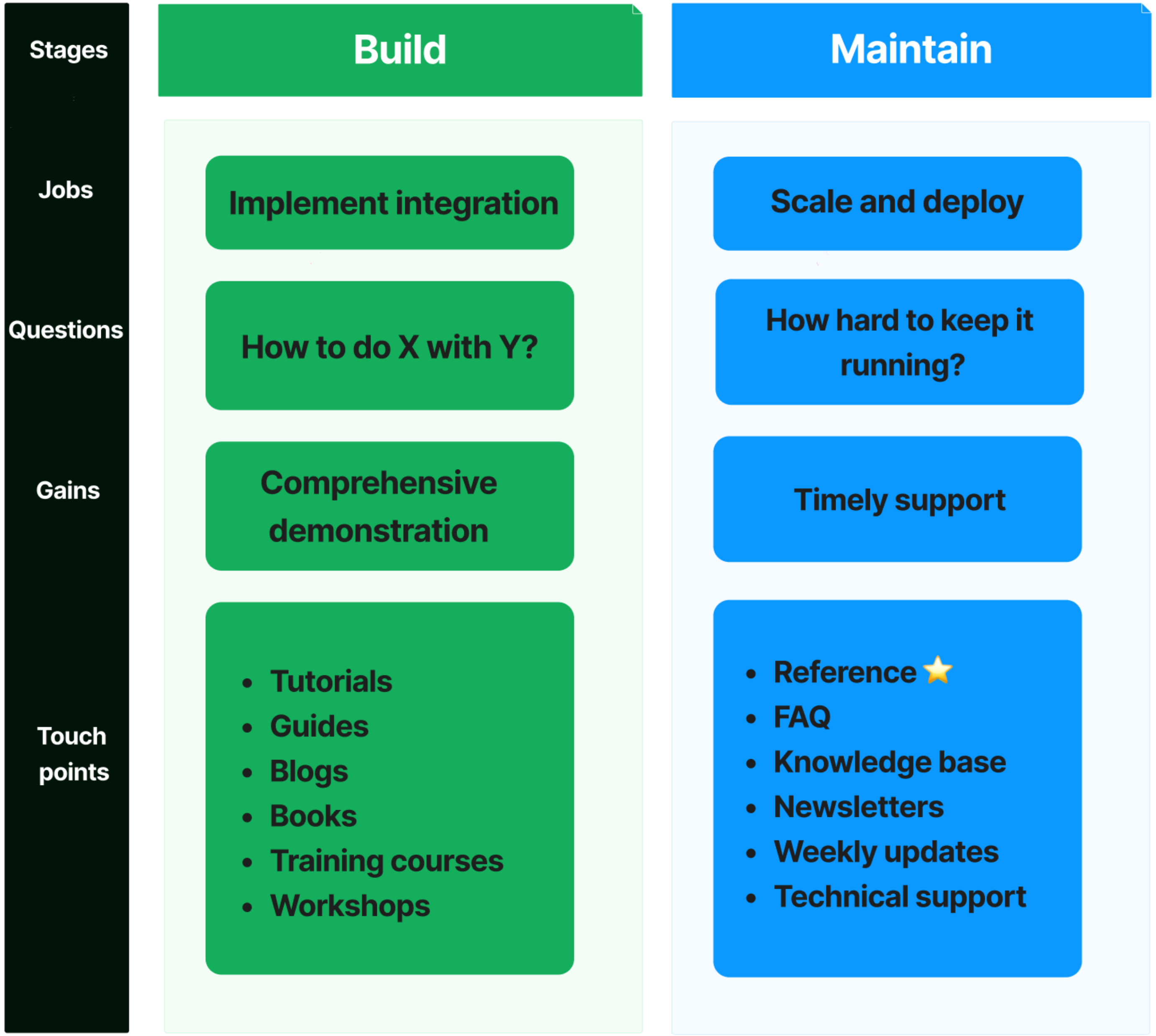
Best practice

Dos

- Provide specialized references, tutorials, and guides with comprehensive contexts and in-depth explanations.

Don'ts

- Incomplete and inaccurate content.



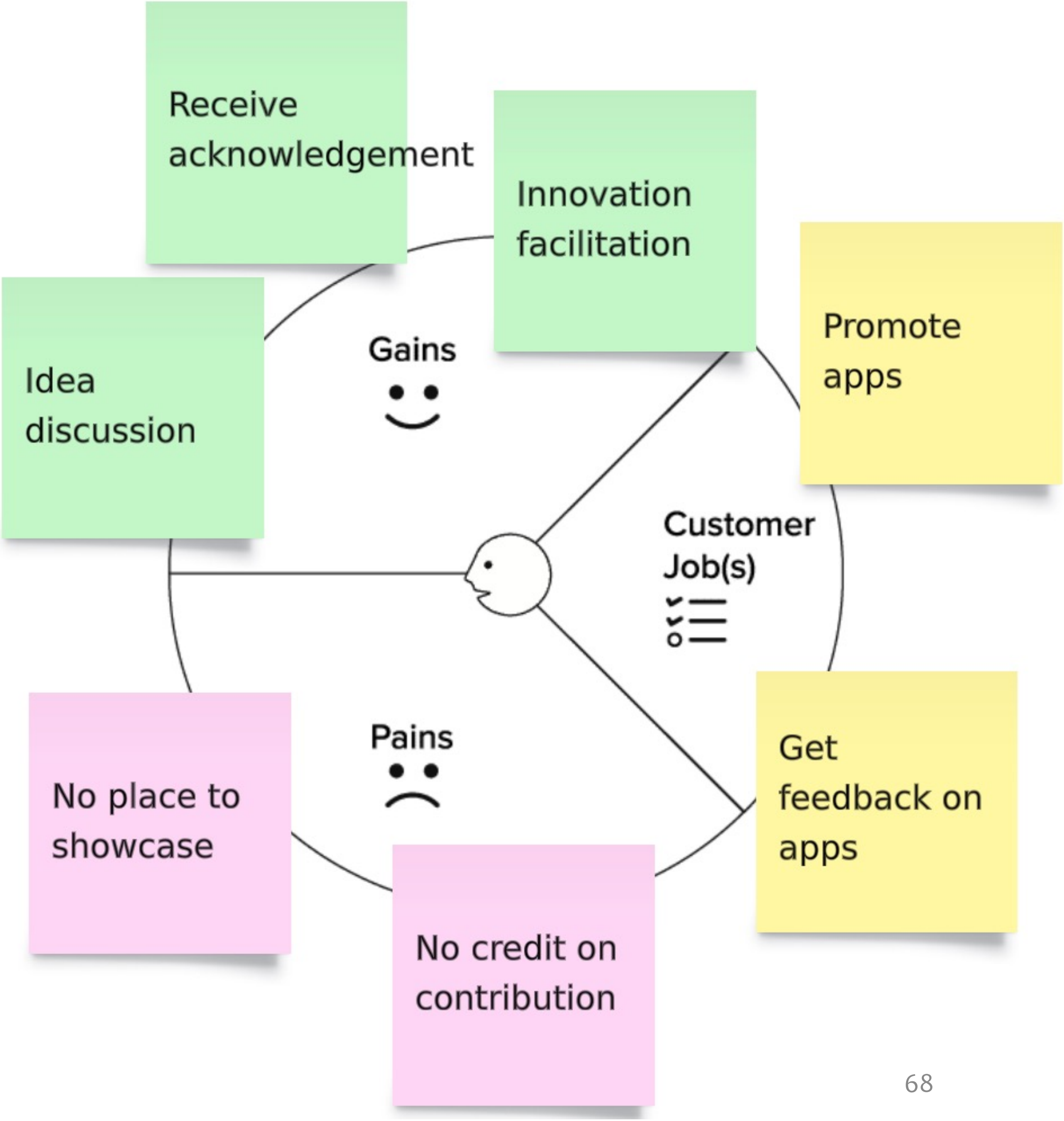
Stages	Build	Maintain
Jobs	Implement integration	Scale and deploy
Questions	How to do X with Y?	How hard to keep it running?
Gains	Comprehensive demonstration	Timely support
Touch points	• Tutorials • Guides • Blogs • Books • Training courses • Workshops	• Reference ⭐ • FAQ • Knowledge base • Newsletters • Weekly updates • Technical support

1. Discover
2. Evaluate
3. Get started
4. Build
5. Maintain
6. Celebrate

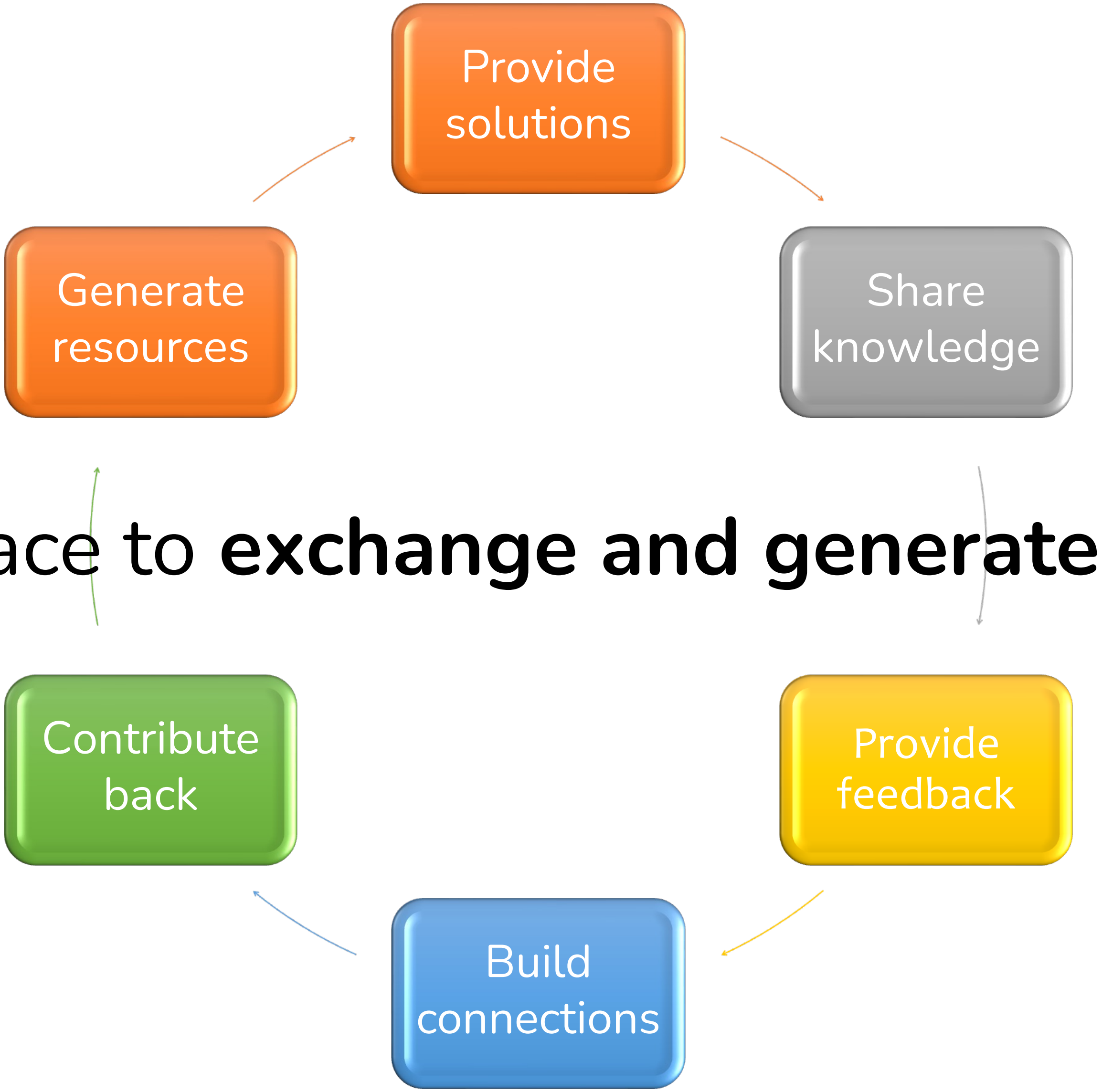
Celebrate

Developers' questions:

- How to promote it?
- Does anyone care about my work?
- How can I get effective feedback?



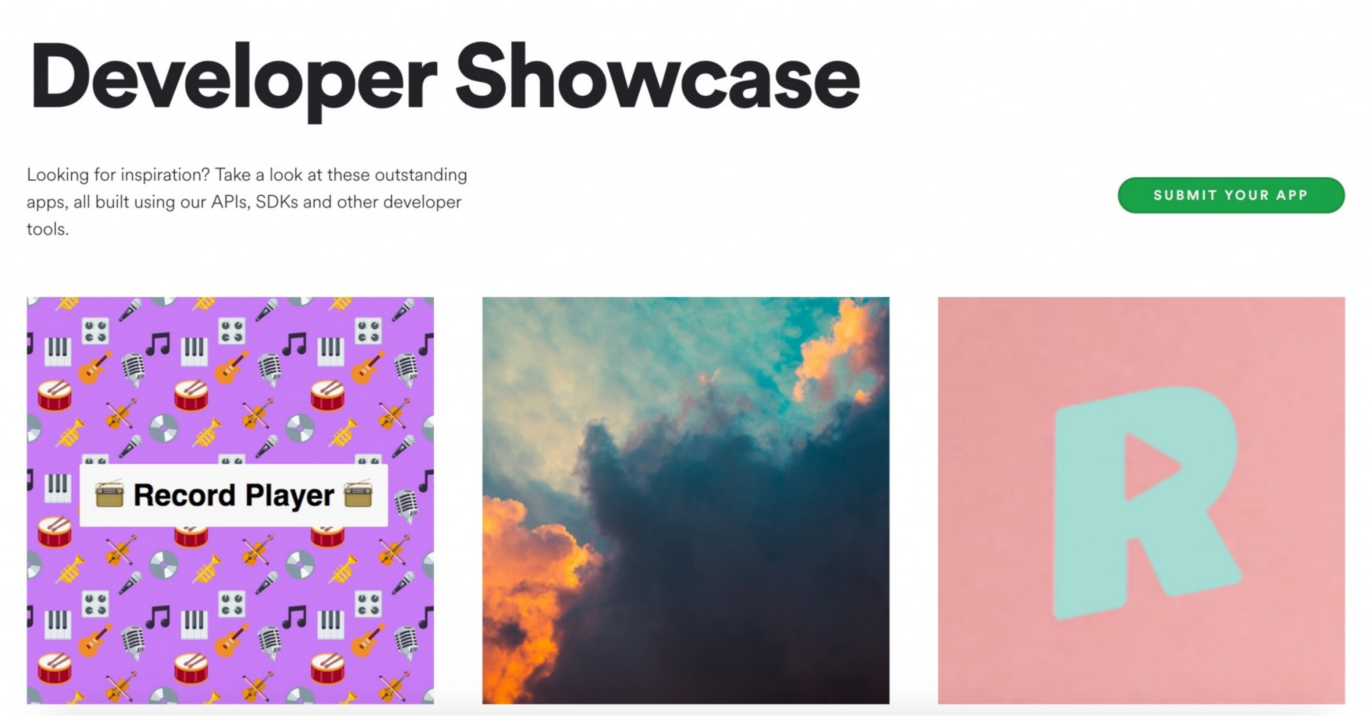
Celebrate



Build a place to **exchange and generate context**

Celebrate

Example

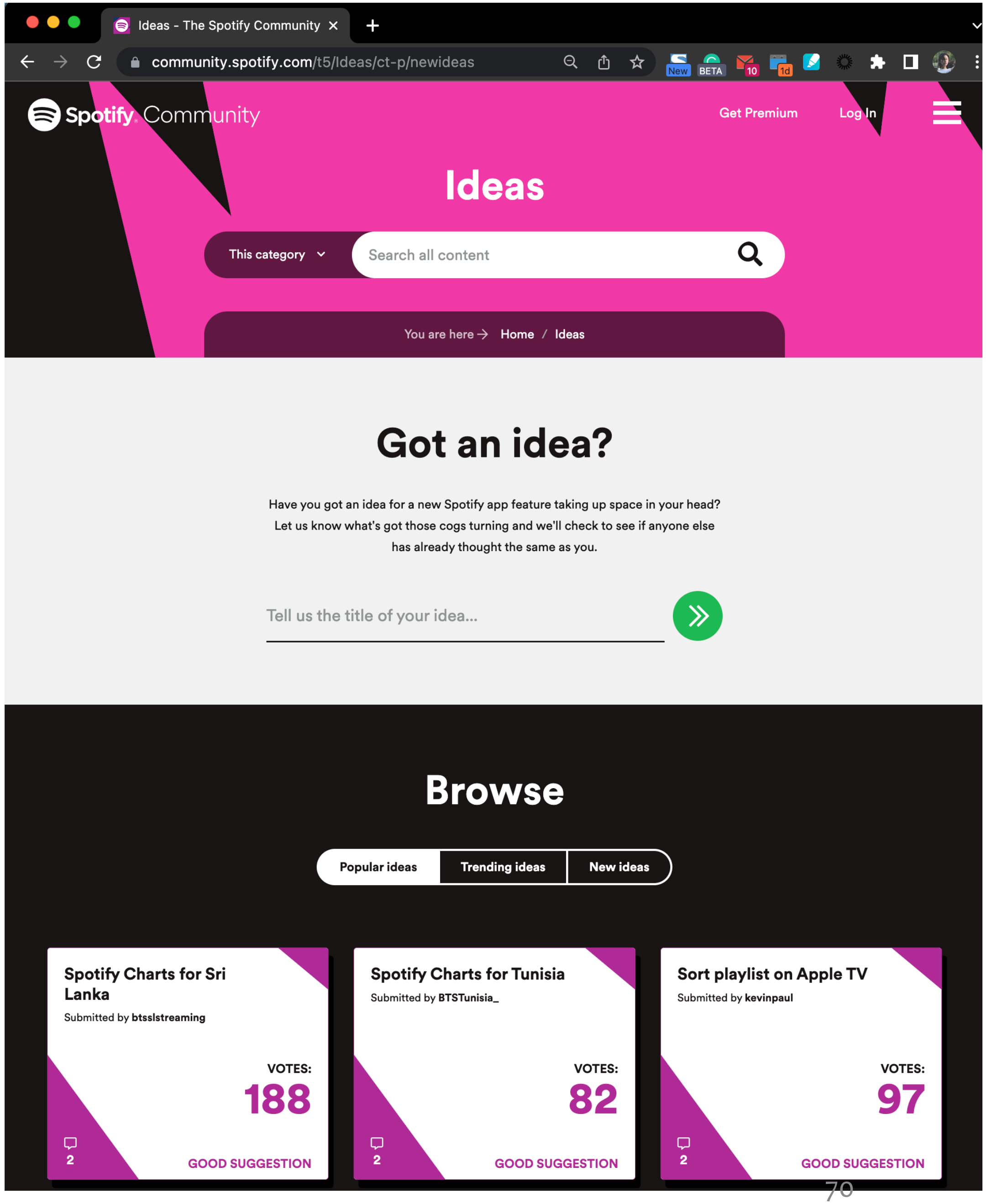


Developer Showcase

Looking for inspiration? Take a look at these outstanding apps, all built using our APIs, SDKs and other developer tools.

SUBMIT YOUR APP

Spotify developer community



Ideas - The Spotify Community

community.spotify.com/t5/Ideas/ct-p/newideas

Spotify. Community

Get Premium

Log In

Ideas

This category

Search all content

You are here → Home / Ideas

Got an idea?

Have you got an idea for a new Spotify app feature taking up space in your head? Let us know what's got those cogs turning and we'll check to see if anyone else has already thought the same as you.

Tell us the title of your idea...

Browse

Popular ideas | Trending ideas | New ideas

Spotify Charts for Sri Lanka
Submitted by **btsslstreaming**
VOTES: 188
2
GOOD SUGGESTION

Spotify Charts for Tunisia
Submitted by **BTSTunisia_**
VOTES: 82
2
GOOD SUGGESTION

Sort playlist on Apple TV
Submitted by **kevinpaul**
VOTES: 97
2
GOOD SUGGESTION

Celebrate

Summary

Best practice

Dos

- Provide a centralized place to build connections between developers and encourage them to exchange info and contexts.

Don'ts

- No place to showcase.
- No credit on contributions.

Stages	Celebrate
Jobs	Promote apps
Questions	Does anyone care about my work?
Gains	Respectful recognition
Touch points	• Community • Forum • 3rd party resources (SO, GitHub) • Social media (Hacker News, DZone, InfoQ) • Conferences • Hackathons

Developer Learning Journey

Summary

⭐ means MVD (minimum viable deliverables)

Stages	Discover	Evaluate	Get Started	Build	Maintain	Celebrate
Jobs	Solve a specific task	Assess validity	Try out quickly	Implement integration	Scale and deploy	Promote apps
Questions	What is it?	Why should I use it?	Where do I begin?	How to do X with Y?	How hard to keep it running?	Does anyone care about my work?
Gains	Inspirational motivation	Trustworthy proof	Strong confidence	Comprehensive demonstration	Timely support	Respectful recognition
Touch points	• Landing page • Use case (tutorials) ⭐ • Success stories	• Pricing info • Release notes • Roadmaps • Terms of use • Usage policies (security, legal, privacy, partner, SLA, cookie)	• Quick start ⭐ • Playground • Code samples • Demo	• Tutorials • Guides • Blogs • Books • Training courses • Workshops	• Reference ⭐ • FAQ • Knowledge base • Newsletters • Weekly updates • Technical support	• Community • Forum • 3rd party resources (SO, GitHub) • Social media (Hacker News, DZone, InfoQ) • Conferences • Hackathons

Content strategy comparison

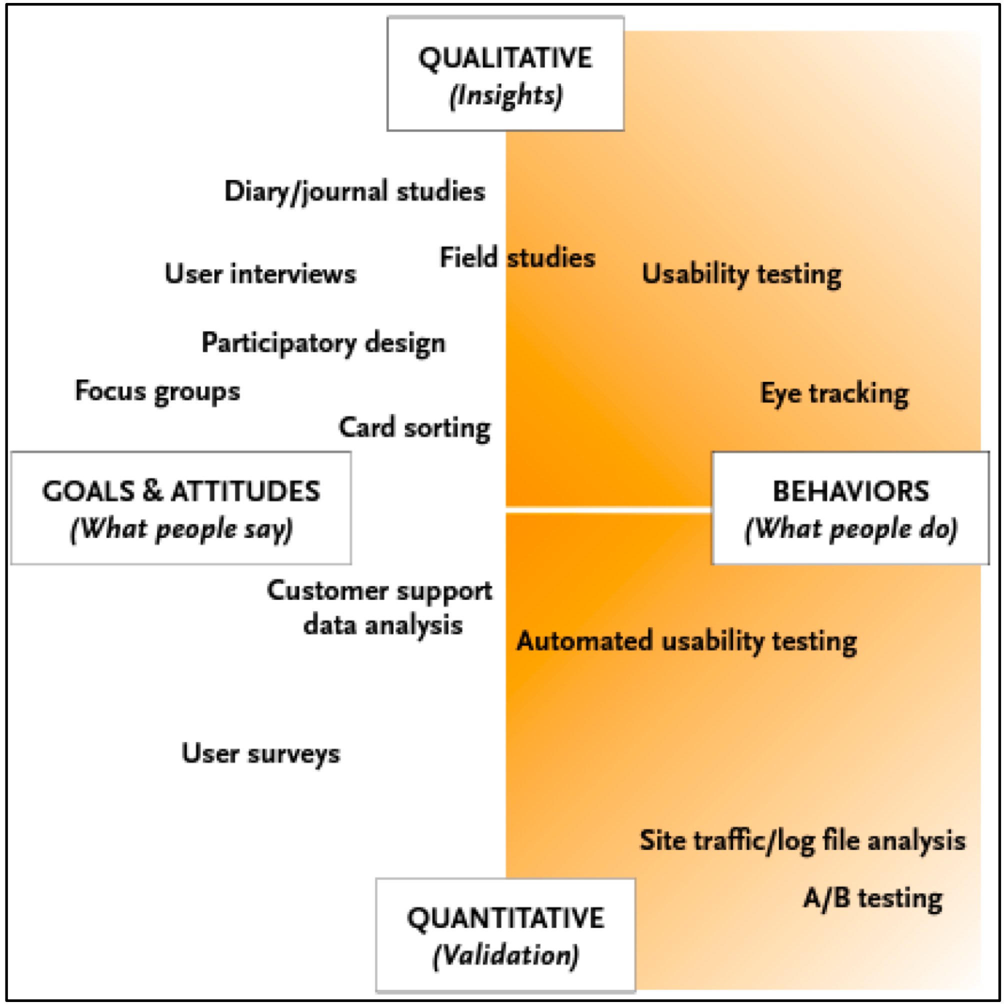
Design content for developers (OS vs. Commercial)

Content strategy	OS projects	Commercial products
Target user	Entry level • Basic • Intermediate	Advanced level
Content goal	Address access path for different skill levels	Address learning/knowledge gaps
Content positioning	Comprehensive references	Up-level skills
Content focus	Basics • Core fundamentals • How-to guides • References	Value-add content with detailed contexts (5W2H) • Use case tutorials • Case studies • Code examples • Deep dive blogs • Specialized trainings • Video courses
Content quality	Good enough	Production quality

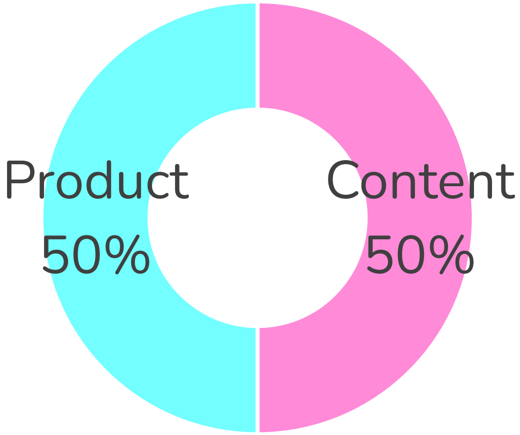
4. How to Evaluate DX?

Evaluate DX

UX Research Method Landscape



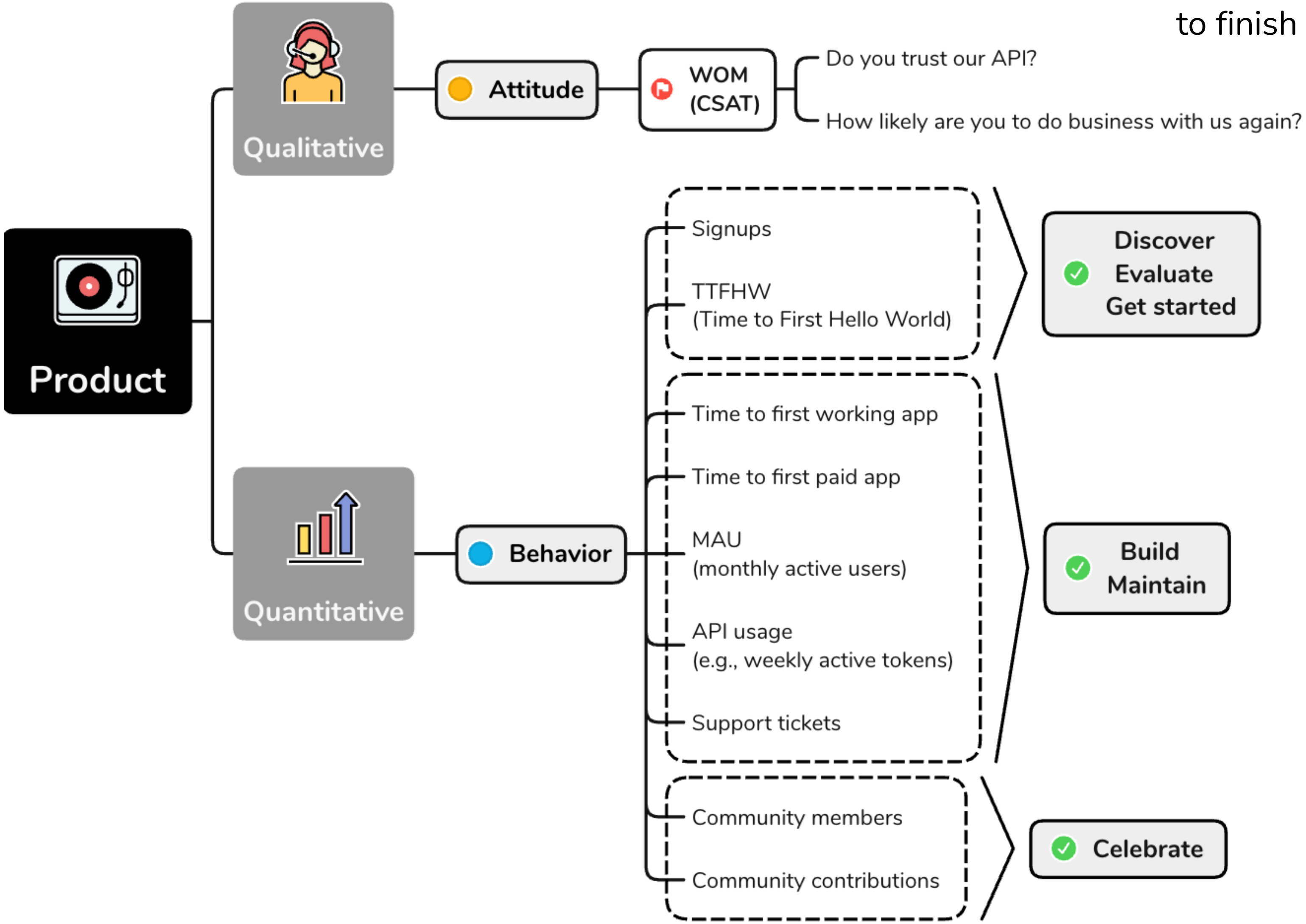
Evaluate both sides



Shared OKRs with both teams

Evaluate DX

Sometimes, UX
≠ spend less
time and effort
to finish tasks



Product

Qualitative

Attitude

WOM (CSAT)

Do you trust our API?

How likely are you to do business with us again?

Quantitative

Behavior

Signups

TTFHW
(Time to First Hello World)

✓ Discover
Evaluate
Get started

Time to first working app

Time to first paid app

MAU
(monthly active users)

API usage
(e.g., weekly active tokens)

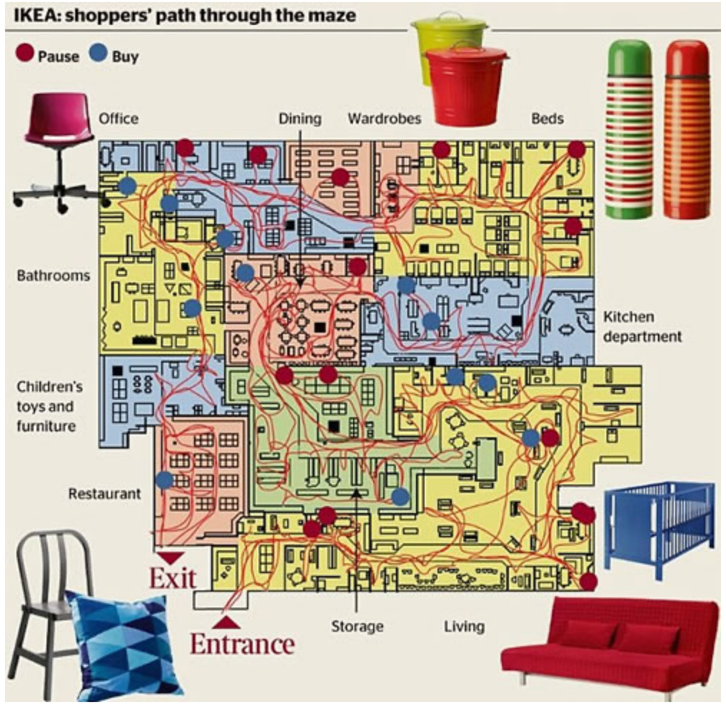
Support tickets

✓ Build
Maintain

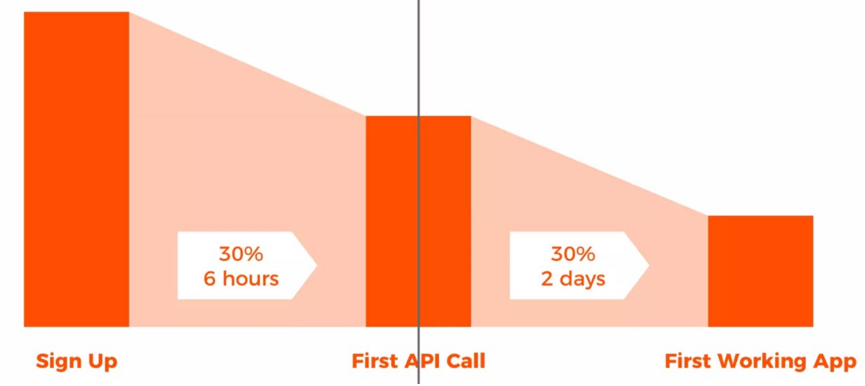
Community members

Community contributions

✓ Celebrate

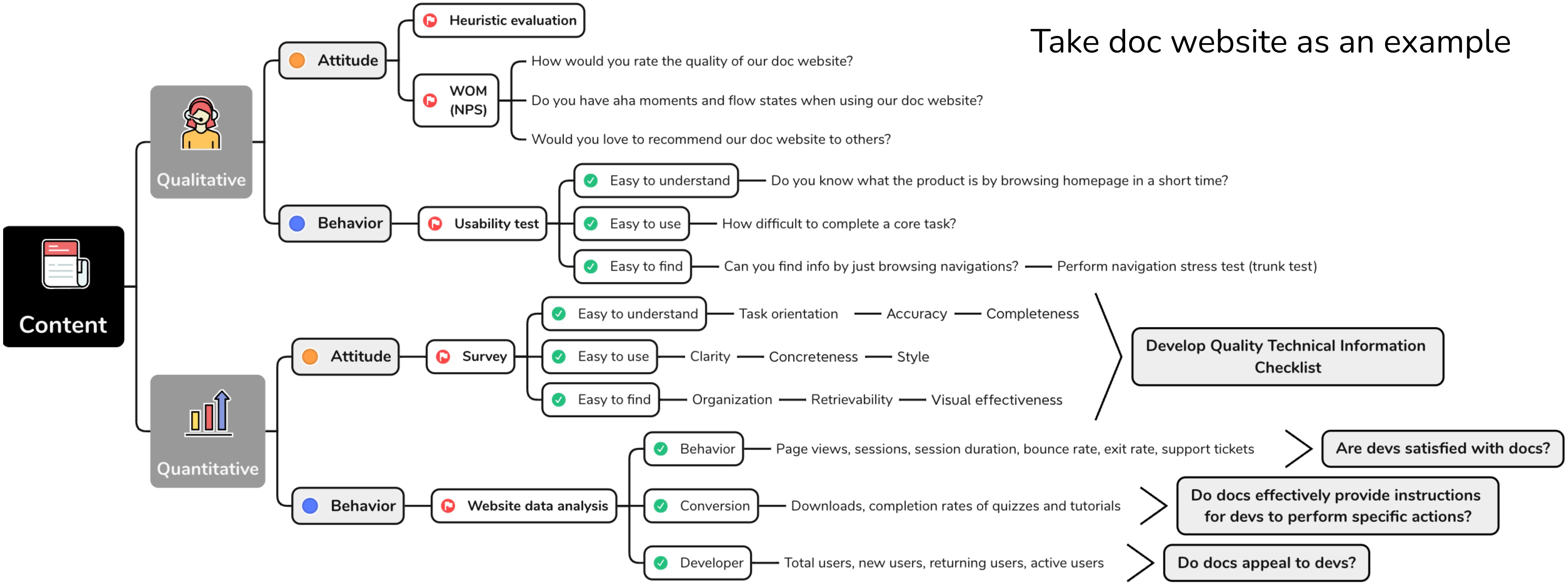


Track both conversion rate and time to next step



Evaluate DX

Take doc website as an example



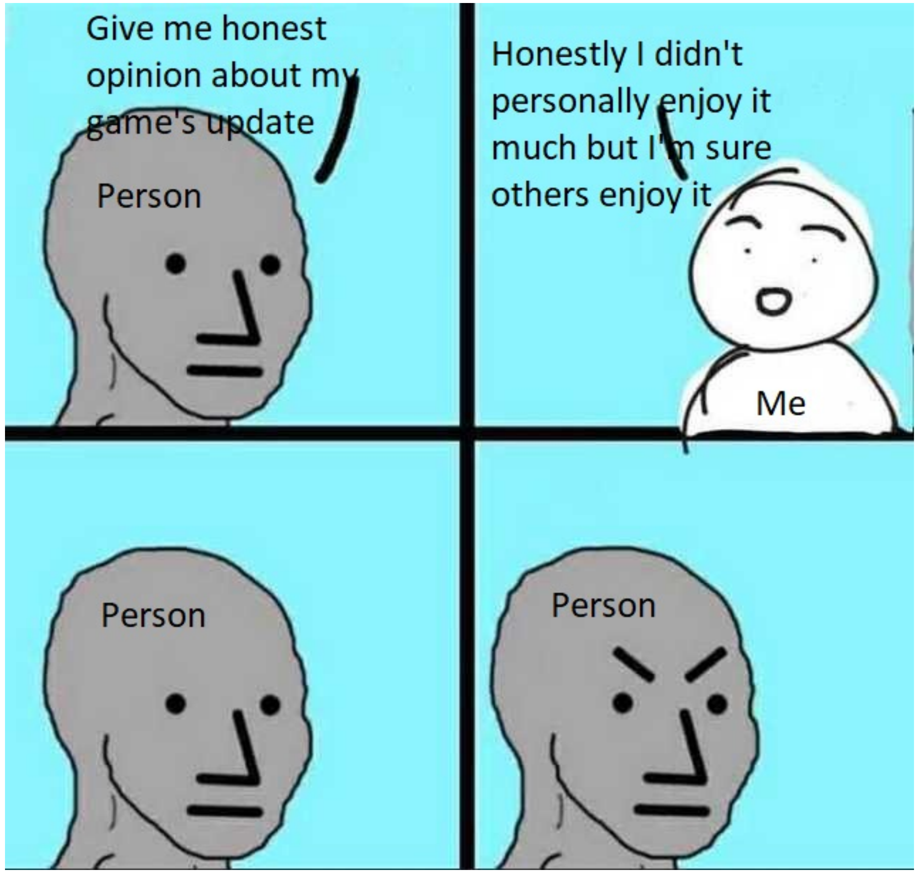
- Content
  - Qualitative
    - Attitude
      - Heuristic evaluation
      - WOM (NPS)
        - How would you rate the quality of our doc website?
        - Do you have aha moments and flow states when using our doc website?
        - Would you love to recommend our doc website to others?
    - Behavior
      - Usability test
        - Easy to understand — Do you know what the product is by browsing homepage in a short time?
        - Easy to use — How difficult to complete a core task?
        - Easy to find — Can you find info by just browsing navigations? — Perform navigation stress test (trunk test)
  - Quantitative
    - Attitude
      - Survey
        - Easy to understand — Task orientation — Accuracy — Completeness
        - Easy to use — Clarity — Concreteness — Style
        - Easy to find — Organization — Retrievability — Visual effectiveness
        - > Develop Quality Technical Information Checklist
    - Behavior
      - Website data analysis
        - Behavior — Page views, sessions, session duration, bounce rate, exit rate, support tickets > Are devs satisfied with docs?
        - Conversion — Downloads, completion rates of quizzes and tutorials > Do docs effectively provide instructions for devs to perform specific actions?
        - Developer — Total users, new users, returning users, active users > Do docs appeal to devs?

Evaluate DX

Qualitative + Quantitative = Sweet Success

✅ Reveal truth

✅ Get honest opinion


Give me honest opinion about my game's update
Person
Honestly I didn't personally enjoy it much but I'm sure others enjoy it
Me
Person
Person

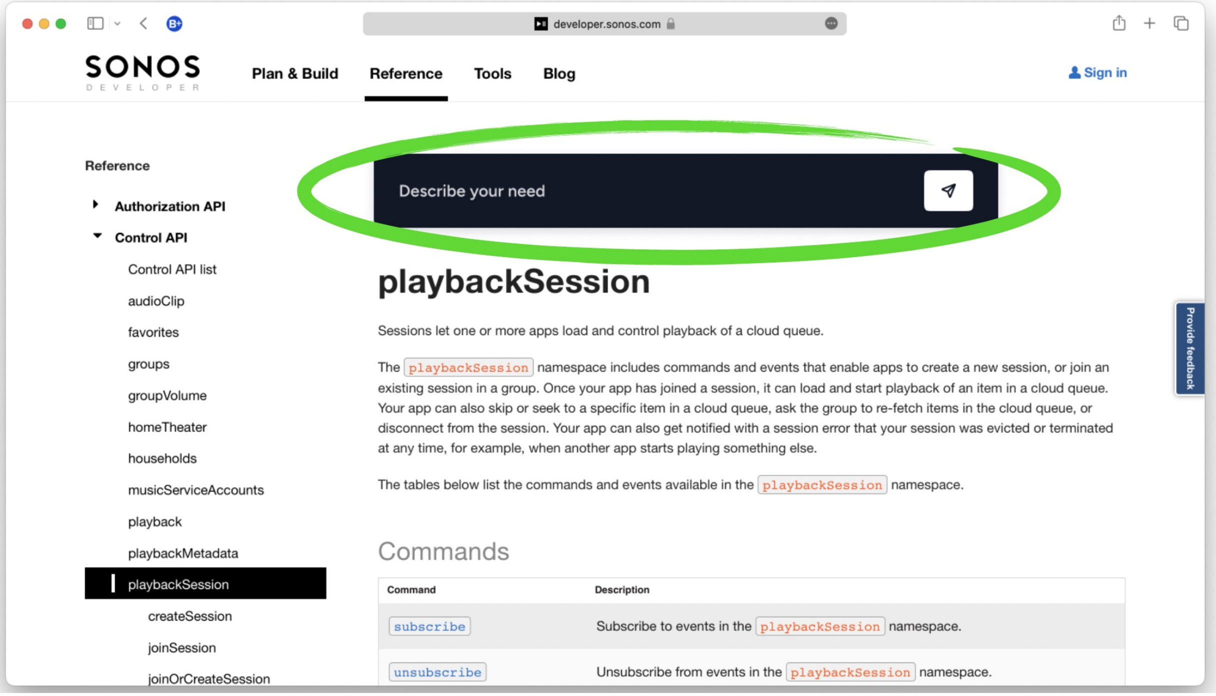
✅ Understand user feedback

5. Thoughts

Thoughts

AI is changing the way of producing and consuming next-generation API content.

How to level up DX with AI?



Describe your need

playbackSession

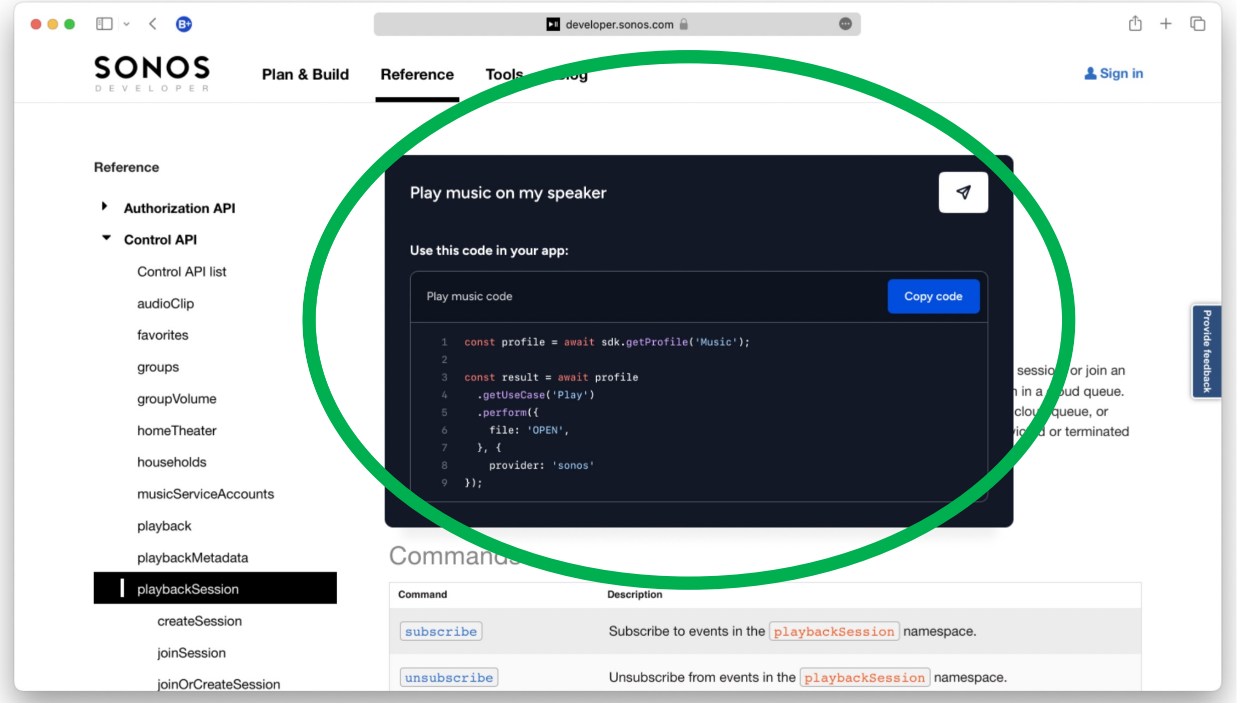
Sessions let one or more apps load and control playback of a cloud queue.

The `playbackSession` namespace includes commands and events that enable apps to create a new session, or join an existing session in a group. Once your app has joined a session, it can load and start playback of an item in a cloud queue. Your app can also skip or seek to a specific item in a cloud queue, ask the group to re-fetch items in the cloud queue, or disconnect from the session. Your app can also get notified with a session error that your session was evicted or terminated at any time, for example, when another app starts playing something else.

The tables below list the commands and events available in the `playbackSession` namespace.

Commands

Command	Description
`subscribe`	Subscribe to events in the `playbackSession` namespace.
`unsubscribe`	Unsubscribe from events in the `playbackSession` namespace.



Play music on my speaker

Use this code in your app:

Play music code

```
const profile = await sdk.getProfile('Music');

const result = await profile
  .getUseCase('Play')
  .perform({
    file: 'OPEN',
  }, {
    provider: 'sonos'
});
```

sessio or join an
n in a ud queue.
clou queue, or
vic d or terminated

Comman

Command	Description
`subscribe`	Subscribe to events in the `playbackSession` namespace.
`unsubscribe`	Unsubscribe from events in the `playbackSession` namespace.

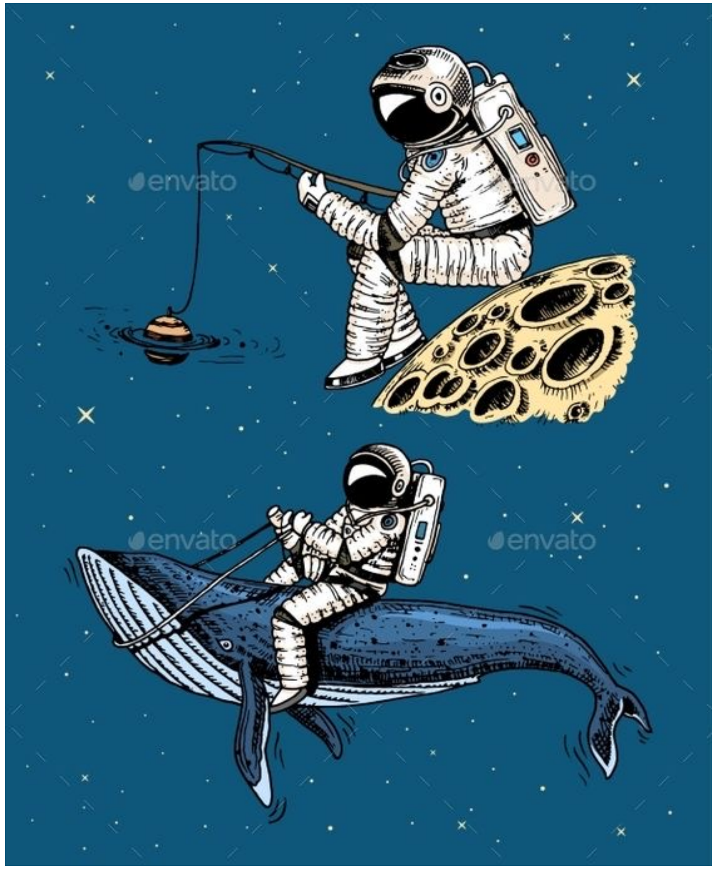
Thoughts

✅ Suggestion 1: Grow your mindset + Polish your creativity

AI won't take your job if you always think outside the box

GLASSES:
If glasses, Warby Parker.
If sunglasses, RayBan.
HOODIE:
Hoodie branded with the tech company you work for. Subtly says "I matter."
TEE:
A t-shirt from another startup that implements your API. *Can be exchanged for a button up on Thursdays.
WEARABLES:
A Pebble or FitBit, tracking your steps from the Mission to SOMA.
MESSENGER BAG:
Th

Some content is a XXXL t-shirt — everyone can wear it but not comfortable with it.



Developers were stereotyped as special nighttime dungeons-and-dragons playing populous who feed on beer and hackathons.

Instead, we should → think developers as

Developers are pioneers who are

- Looking for opportunities and ways to interact with business.
- Building new bridges for technology and business to adapt and grow.

Thoughts

✅ Suggestion 2: build more connections and contexts

AI can not do the full job of writers because

- AI produces content based on existing content.
- So much of what writers do isn't writing,

  it's relationship building:
  - The ability to empathize with users at every level.
  - The ability to build emotional intimacy and gain trust.
  - The ability to see the content as an entire product, not just discrete words, sentences, and topics.

Takeaways

✅ Design DX

1️⃣ Understand developers

- DDMU (Developer Decision-Making Unit): Initiators, Influencers, Decision-makers, Budget holders
- Archetypes: Systematic, Opportunistic, Pragmatic

2️⃣ Map out developer journey (6 stages)

- ▶️ Discover ▶️ Evaluate ▶️ Get started ▶️ Build ▶️ Maintain ▶️ Celebrate

3️⃣ Create content for developer journey

- Choose deliverables and prioritize tasks using the Value Proposition Canvas (Jobs, Gains, Pains)
- Provide MVD: use case tutorials, get started, and references
- Differentiate content strategies for open-source projects and commercial products

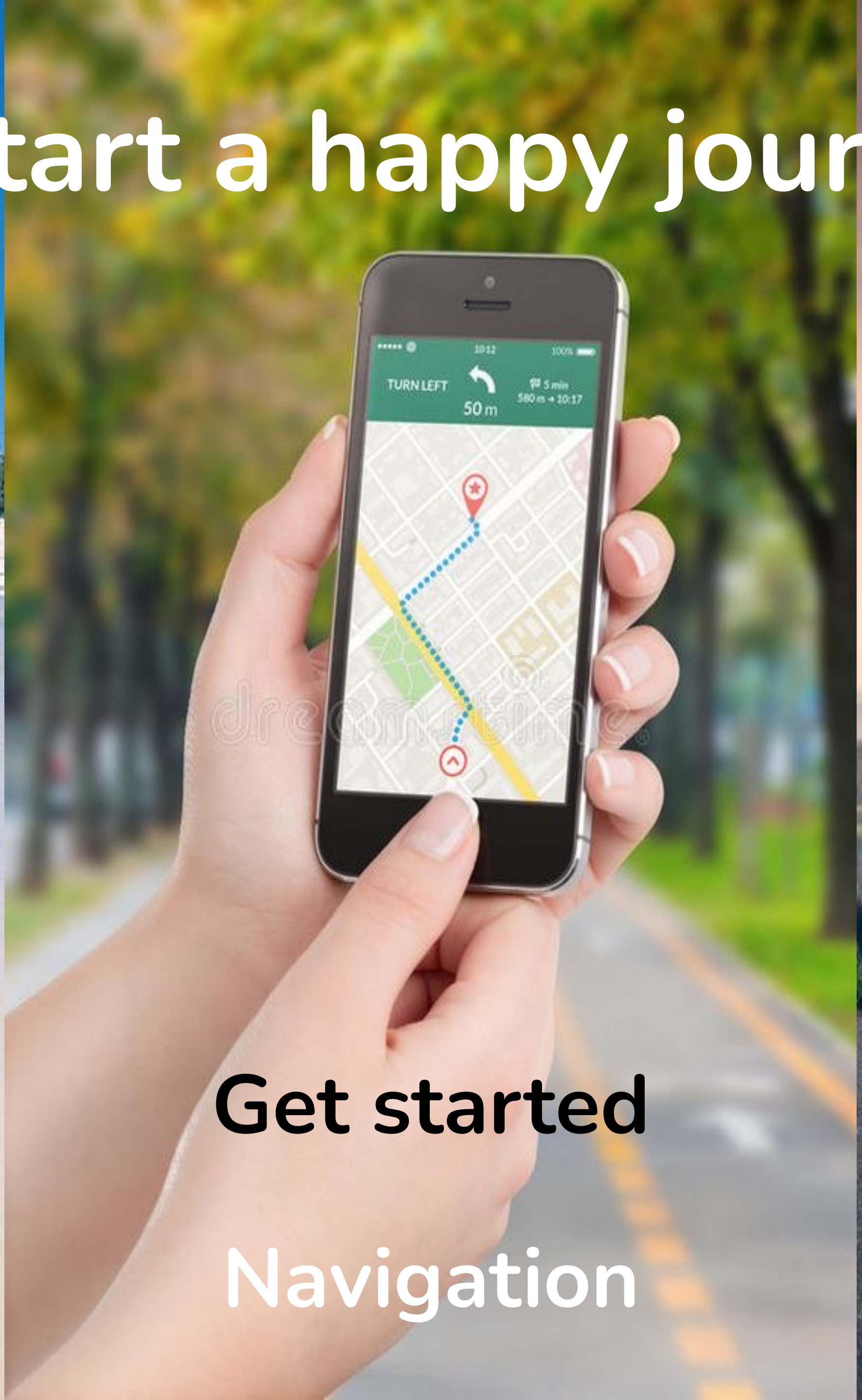
✅ Evaluate DX

- UX Research Method Landscape: qualitative + quantitative, analyze attitudes + observe behaviors
- DQTI (Develop Quality Technical Information): easy to understand, use, find

Developers start a happy journey 🎉

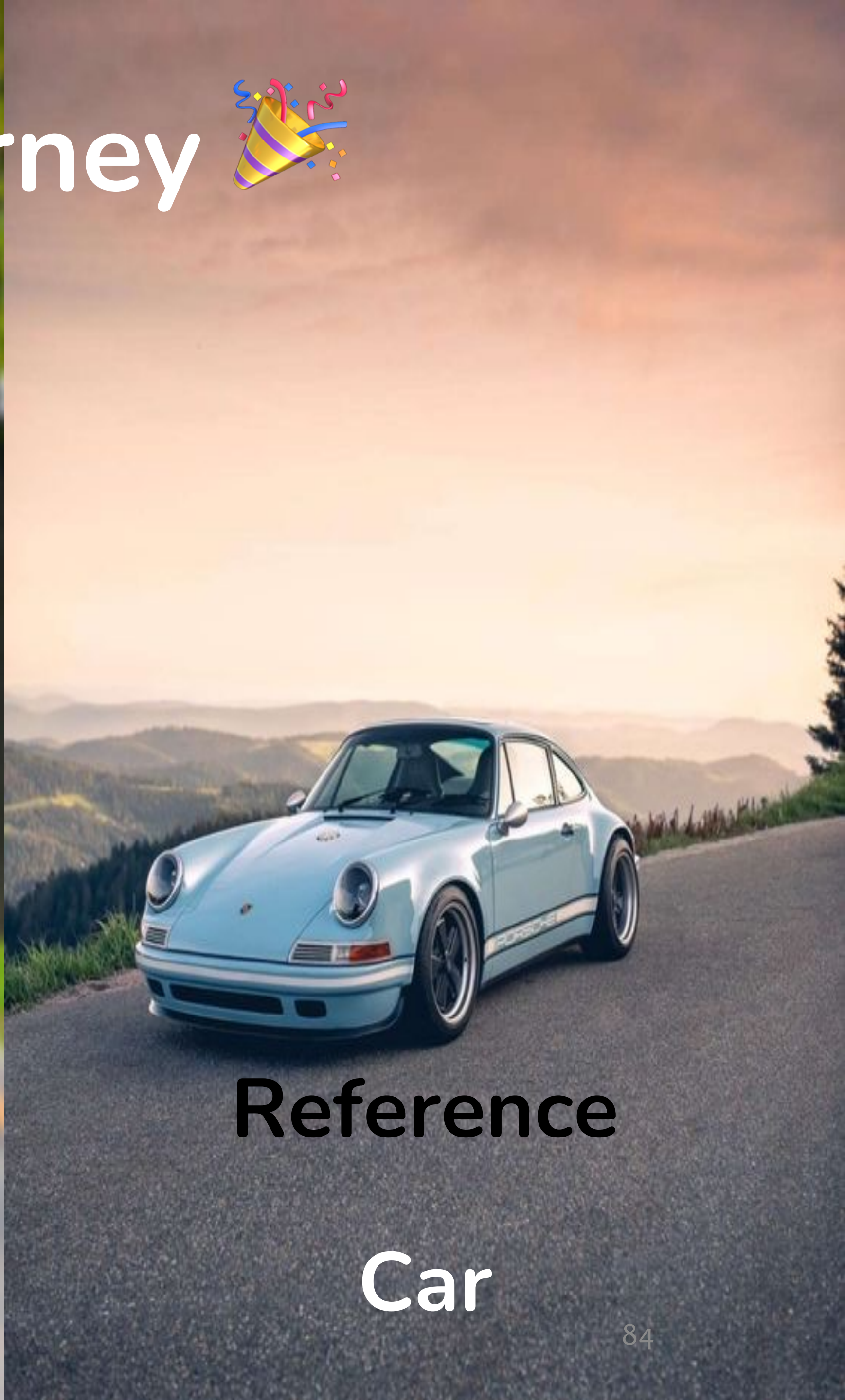
Use case tutorial

Destination



Get started

Navigation



Reference

Car

References

✅ Books

- Developer Relations: How to Build and Grow a Successful Developer Program
- Information Architecture for the World Wide Web
- The Elements of User Experience: User-Centered Design for the Web
- IBM Design Thinking

✅ Yu's talks (videos and slides are available)

- Cracking the Code of Information Architecture
- Inside Apache Pulsar's Content Strategy
- Success Beyond Code: Optimizing Developer Experience Through PR Titles
- Code the Docs: Continuous Integration for Docs
- Growing a Company to be a Top OS Contributor
- Building a Welcoming Community

Scan the QR code to add me on WeChat

开源社
kaiyuanshe

THANK YOU

QUESTIONS?

WeChat Official Account: 开源社KAIYUANSHE

WeChat Channels: 开源社KAIYUANSHE

Weibo: 开源社

Bilibili: 开源社KAIYUANSHE

Jianshu: 开源社

TouTiao: 开源社

Facebook: KaiyuansheChina

Twitter: 开源社KAIYUANSHE

Scan to Follow
KAIYUANSHE
WeChat Account

Scan to Contact with
COSCon Speakers