appsembler

Build it with education.

The 5 Benefits of Developer Marketing

Table of Contents

appsembler
Build it with education.

Chapter 1

The Now-Famous "Ask Your Developer" Campaign

When Twilio launched their now-famous "Ask Your Developer" campaign in 2015, it was a landmark event that the technology industry will always remember as they saw Twilio's ads and high-visibility billboards displayed in urban centers like San Francisco and Boston.

What started as a marketing campaign intended to improve Twilio's product awareness with hard-to-reach decision makers, in retrospect, ended up being so much more.

The 3-word campaign was initially designed to help Twilio cut through the noise and to reach busy executives who didn't know about Twilio but were nonetheless making buying decisions that affected Twilio's market adoption. The idea behind the 3-word campaign was to *not* explain what Twilio does on a marketing campaign, but rather to prompt decision makers to ask their developers to do the explaining on behalf of Twilio. Developers were very familiar with Twilio through its tools and API — now Twilio was placing developers at the center of their marketing campaign.

What made the campaign special was that for the first time, a company was putting the developer first in their marketing. Twilio was purposefully championing the developer as a key influencer in a decision-making process usually driven by the commercial functions of a business. To be fair, Twilio wasn't the only company cultivating a developer community — events such as Google I/O, Facebook F8, Microsoft Build, and the long-tail of developer-centric Meetup events were staples at any tech hub's technology scene — but Twilio's "Ask Your Developer" campaign placed them at the forefront of what companies were doing on behalf of developers. The campaign was simple yet revolutionary, and little did they know, would help pave the way for the creation of the Developer Marketing category.

For the first time, a company was putting the developer first in their marketing.

Developer marketing departments to watch:

Fast forward several years and you have companies like Slack, Plaid, Docker, MongoDB, Kong, Redis, and Stripe who have reached billion-dollar valuations making great products supported by an effective developer marketing and developer advocacy strategy. Then there are stalwarts such as Oracle, SAP, IBM, Adobe, Amazon, InterSystems, Nutanix, General Motors, and Ford who have large and long-standing developer ecosystems. And finally, you have the beginnings of a new category taking shape with SlashData, the leading analyst firm in the developer economy; events like Developer Week, apidays, SlashData's Future Developer, and Evans Data's developer conferences; service providers such as Iron Horse, Catchy Agency, and BeMyApp; and venture capital firms like Unusual Ventures who are building in-house expertise to help portfolio companies with developer marketing initiatives.

Read the full story on Forbes

But the practice of developer marketing is still confined to the small chambers of its current practitioners, who typically reach their Developer Marketing role through three common routes:

- **A** Developers who found a knack for building developer communities and transitioned into an external, developer-centric role;
- **B** Engineers who had the combination of being technical and being great communicators, and so their day-to-day tasks pulled them more and more towards developer relations; and
- **C** Product-centric marketers who found themselves interacting with their company's technical audiences, such as external developers, their resellers' engineering staff, etc.

And while roles specifically tailored for Developer Marketing are starting to pop-up, they are still few and far between. Even for the drastic progress that Developer Marketing has made in the last few years, the number of Developer Marketing roles is still dwarfed by the number of "traditional" marketing roles such as Marketing Managers or quota-carrying roles such as Sales Engineers. So as far as the category has come, the broader world is still coming to terms with accepting and understanding what Developer Marketing exactly is and how a developer community affects a company's bottom line. In short, Developer Marketing is still in a nascent stage with significant room for growth, market education, and market adoption.

The broader world is still coming to terms with accepting and understanding what Developer Marketing exactly is and how a developer community affects a company's bottom line

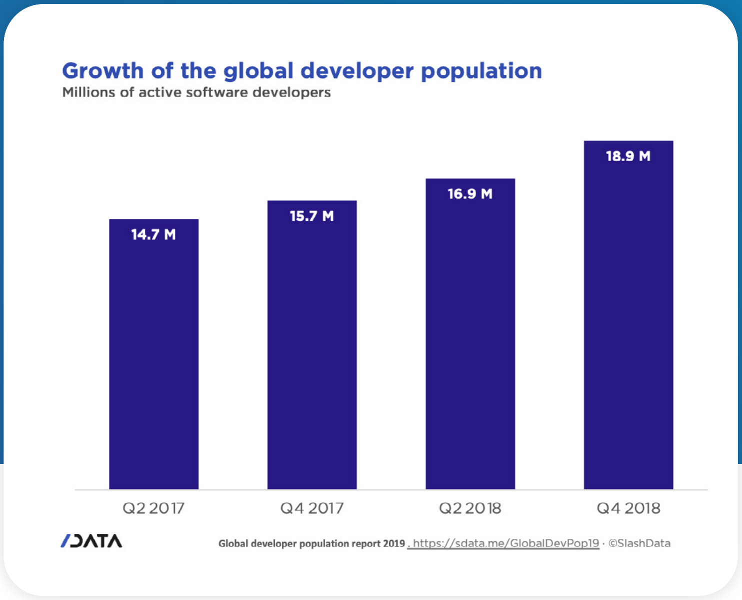
Despite its success, Developer Marketing is still in its nascent stage

From a random sample of 70 software companies:

- 100% had a marketing team
- 69% had a sales engineering team
- 99% had a product marketing team
- 43% had a developer marketing team

Key Trends Driving the Need for Developer Marketing

The undercurrent of more developers coming:

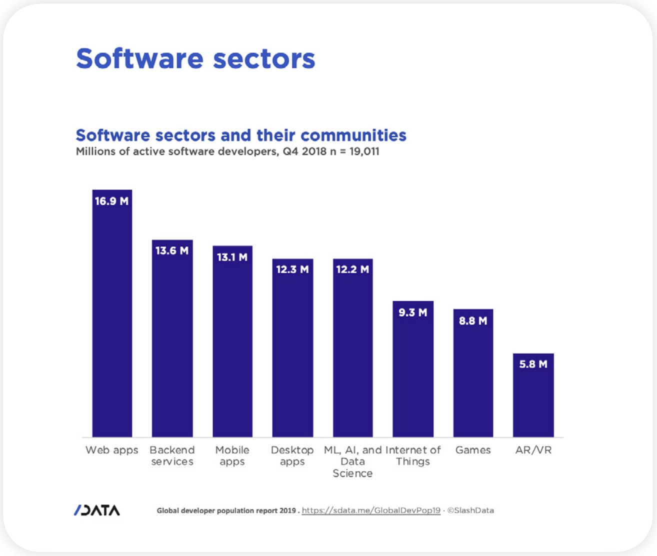


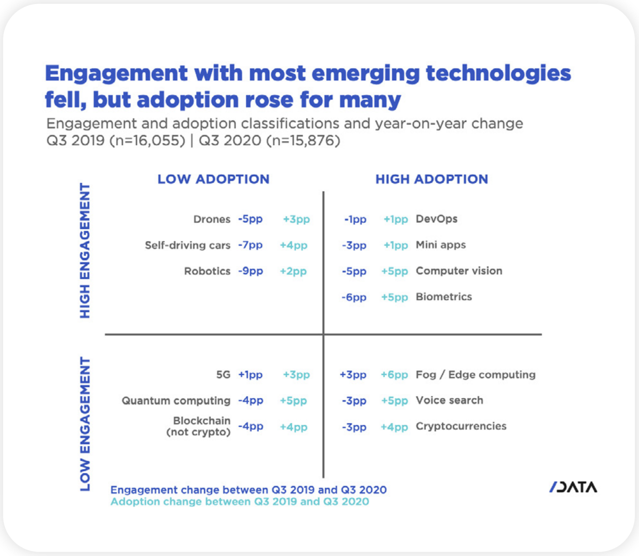
If we extrapolate the current growth rate outwards, we can expect to see at least 21 million developers by the end of 2019 and possibly upwards of 23 million.

45 million developers globally in 2030.

Source: 2019 State of Developer Economy Report by SlashData

The undercurrent of more innovation and software products coming:





Chapter 2

The Aches of Category Creation

While in its nascent stage, the Developer Marketing category has several areas that could be further refined and matured.

Several articles and books are written on this topic, namely SlashData's Developer Marketing: The Essential Guide, but in this eBook we'll categorize developer marketing's refinement areas into three (3) groups:

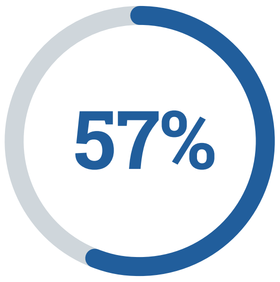
1. The practice of developer marketing - what do we do and what are our KPIs?
2. Improved stakeholder and organizational acceptance
3. Tools and technologies that practitioners need for success

In this eBook, we will concentrate on helping *current* developer marketers improve their stakeholder's understanding of the developer marketing role, specifically by helping current developer marketers articulate why developer marketing is important, the problems it solves, and the opportunities it helps organizations capture. We also introduce concepts and best practices on how current developer marketers are attempting to solve some of their most pressing problems today.

Why is developer marketing important?

Based on Stack Overflow's 2020 Developer Survey of 65,000 developers — and probably confirmed anecdotally by your Sales Team — today, 57% of developers exert a great deal of influence over technology purchases within their organization. And a 2017 Developer Marketing Survey by Evans Data places developers' purchase influence as high as 95%. Simply put, a majority of technology purchasing decisions are influenced or even made exclusively by a developer or a technical team. Long gone are the days when developers were expected to implement technologies after the buying and evaluation process were made. In fact, another survey by DeveloperMedia suggests that 60% of developers have the ability to approve or reject a technology purchase.

In short, developers are not only influencing buying decisions, but in many instances actually control the budget to make the purchase decision. So if you're a B2B software company looking to increase your product's adoption, it's simply not tenable to continue ignoring developers and technical SMEs during the marketing and selling process.



stackoverflow

65,000 developers stated that **57% of them** exert a great deal of influence over technology purchases within their organization

DeveloperMedia

748 developers stated that **60% of them** have the ability to approve or reject a technology purchase

And if you reflect on why this trend is happening, it makes sense. Why? Because after a buying decision is made, developers are typically the ones responsible for *implementing* the purchase decision. They are the ones that intimately understand how the purchased technology will fit within an organization's existing technology stack or how the technology will influence the end user's product experience. They are the people who take an organization's vision and build it into a reality.

Developers are the people who take an organization's vision and build it into a reality.

If a developer doesn't like your product, doesn't know who to contact if they need support, isn't well-trained in your product, or can't access your documentation — you're ignoring a key influencer in the buying decision and are ignoring the people responsible for ensuring your product's success. So getting developers onboard early, educating them on your product, and supporting them throughout your product's implementation are critical to ensuring that your product reaches its full potential.

And that's if you even reach the developers in the first place. As more software companies adopt a product-led growth strategy, they are discovering that their products need to be easier to try and use — which means having better self-service and try-before-you-buy programs that have less dependency on a live sales demo. And of course, developers are allergic to speaking with sales teams, so if you're trying to sell a product that will involve a developer in the purchasing decision, gating your product behind a sales call will alienate developers and ultimately stunt your product's growth potential. Finally, when you do make the sale, an unhappy and unengaged developer community will eventually stunt the growth of your product — and may even result in early customer churn.

As more software companies adopt a product-led growth strategy, they are discovering that their products need to be easier to try and use.

Now that we've established that developers are a growing and important step to your product's success, and introduced the pitfalls to ignoring developers in your marketing and sales efforts, in the next section we explore the common problem areas faced by developer marketers and how developer marketing is adding value in each situation.

Chapter 3

The 5 Benefits of Developer Marketing

1. Developer Adoption

In the most pragmatic interpretation of what developer marketing achieves, it is that it helps increase the number of developers using your product. Now we all know that developer marketing is more nuanced than just increasing the number of developer sign-ups, but to our stakeholders and our stakeholders' bosses, developer adoption is one KPI that they can understand and quantify. And so as much as we don't want to anchor the success of our developer marketing efforts on a single KPI like "# of new developers signed," ignoring and not improving this KPI (or not having a plan on how to improve the KPI) will ultimately reflect poorly on our developer marketing efforts.

Developer marketing helps increase the number of developers using your product.

Developer adoption can be interpreted as either the # of developers sign-ups, but it can also mean the level of which our product is *used* by our developer community. In this section, we'll refer to developer adoption as the # of developer sign-ups. In a later section, we'll cover developer usage.

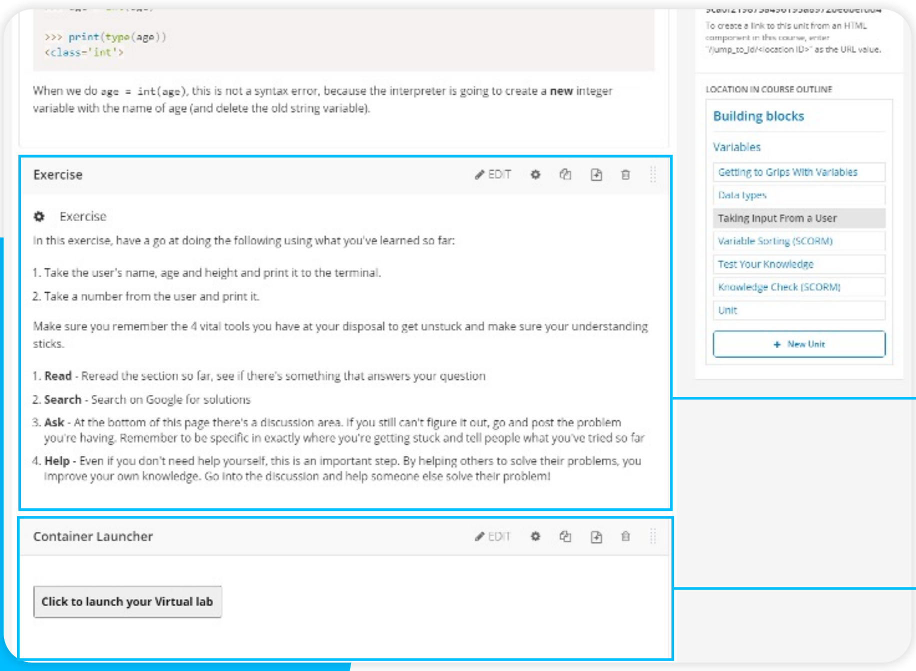
To increase developer adoption, there are **two main challenges we need to resolve:**

1. Reduce, as much as possible, the friction between developers and our product(s); and
2. Increase, as much as appropriate, the context that developers have when using or testing our product.

By friction, what we're referring to is the level of effort a developer needs to exert to work on our product. A developer should be able to reach a live, hands-on version of your product with as little effort as possible. This is especially an issue if your product requires a multi-step installation process or is a desktop-installed application, where by the time the developer has installed your product, their interest (initially high) has flattened. Or worse, the developer encounters an error during the installation process which obviously reflects poorly on your product (and may altogether prevent the developer community's broader adoption of your product). In either scenario, your installation process is creating friction between developers and your product.

For SaaS or web-based applications, this friction is lower because developers can launch your product directly from their web browser. But product trials are typically built for sales and marketing purposes, with non-technical audiences in mind, and without the technical walkthroughs that a developer may expect or need. So if your product is SaaS or web-based, developers are likely experiencing low levels of friction to reach your product — but have low- to zero-levels of context in terms of how to use your product.

Today, best-in-class developer marketers are providing their developer communities with educational product content that has easy-to-launch versions of their product embedded inside the educational experience. Their goal is to provide interactive product learning experiences that includes educational training content with immediate access to live, hands-on versions of the product. This way, developers are learning *and* using it simultaneously — leading to better rates of developer adoption.



```
>>> print(type(age))
<class='int'>
```

When we do `age = int(age)`, this is not a syntax error, because the interpreter is going to create a **new** integer variable with the name of age (and delete the old string variable).

Exercise

Exercise

In this exercise, have a go at doing the following using what you've learned so far:

1. Take the user's name, age and height and print it to the terminal.
2. Take a number from the user and print it.

Make sure you remember the 4 vital tools you have at your disposal to get unstuck and make sure your understanding sticks.

1. **Read** - Reread the section so far, see if there's something that answers your question
2. **Search** - Search on Google for solutions
3. **Ask** - At the bottom of this page there's a discussion area. If you still can't figure it out, go and post the problem you're having. Remember to be specific in exactly where you're getting stuck and tell people what you've tried so far
4. **Help** - Even if you don't need help yourself, this is an important step. By helping others to solve their problems, you improve your own knowledge. Go into the discussion and help someone else solve their problem!

Container Launcher

To create a link to this unit from an HTML component in this course, enter "/jump_to_id/<location ID>" as the URL value.

LOCATION IN COURSE OUTLINE

Building blocks

Variables

- Getting to Grips With Variables
- Data types
- Taking Input From a User
- Variable Sorting (SCORM)
- Test Your Knowledge
- Knowledge Check (SCORM)
- Unit

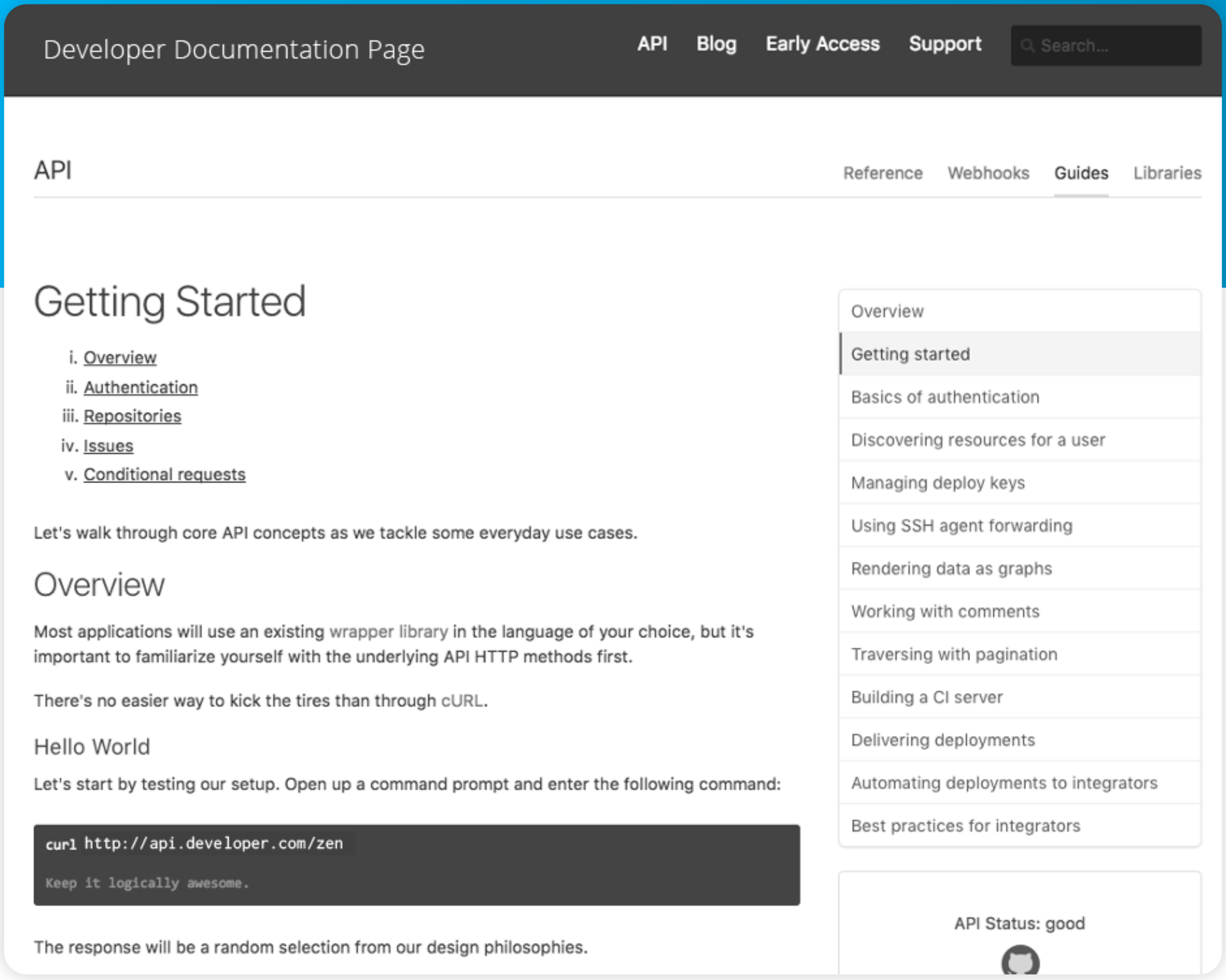
Educational content is given to the learner to provide context and learning principles about the product.

After reading the content above, the learner launches the product for hands-on learning.

2. Developer Success

Another area where developer marketing adds value is in helping developer communities be more successful with your product. Traditionally, this meant using products like GitHub Pages, Read the Docs, Tettra, Confluence or Apiary to host and distribute documentation. Documentation products are an essential component of any developer marketing initiative because they provide the basic foundation that developers use to begin understanding how to work with your product.

Documentation products provide the basic foundation that developers use to begin understanding how to work with your product.



Getting Started

i. Overview
ii. Authentication
iii. Repositories
iv. Issues
v. Conditional requests

Let's walk through core API concepts as we tackle some everyday use cases.

Overview

Most applications will use an existing wrapper library in the language of your choice, but it's important to familiarize yourself with the underlying API HTTP methods first.

There's no easier way to kick the tires than through cURL.

Hello World

Let's start by testing our setup. Open up a command prompt and enter the following command:

```
curl http://api.developer.com/zen

Keep it logically awesome.
```

The response will be a random selection from our design philosophies.

These documentation products, however, act more as Wikis or as a knowledge base — and don't have the feedback loops that provide developer marketing departments with deeper insight into how their developer communities are doing. For example, a simple documentation page won't help answer questions like:

1. What parts of my product are developers learning most about?
2. What parts of my product are developers ignoring?
3. What parts of my product are developers talking most about?

So while documentation tools provide a body of knowledge, they don't have the feedback loops that developer marketing teams need to be truly effective.

The next generation of developer documentation includes interactivity, thus providing developer marketing departments with the feedback loops to better understand how to help their developer communities succeed. These feedback loops are also used by product teams to inform them on how satisfied developers are with their product, areas of the product that the developer community is struggling or thriving with, and provides a channel for feature requests. Over time, this helps to drive the developer community's success with a company's product.

Ways to add interactivity to your developer documentation

- Surveys
- Short quizzes
- Multiple-choice questions
- Open-ended questions
- Community discussions
- Video
- Instructor grading
- Peer graded questions
- Hands-on product trials

3. Developer Usage

When it comes to developer adoption, some would say that increasing the # of developer sign-ups is a vanity metric — and that developer *usage* is what truly matters. In essence, that you would rather have 100 very active users instead of 1,000 inactive users. The reality is not a binary, one-or-the-other decision. In fact, increasing the # of developer sign-ups *and* improving developer usage are both important because they measure the success of different parts of a company's developer marketing program. In this section of the eBook, we discuss the obstacles inherent in improving developer usage and what some developer marketers are doing to circumvent these obstacles.

Increasing the number of developer sign-ups and improving developer usage are both important because they measure the success of different parts of a company's developer marketing program.

Once a developer is aware of your product and has signed up for access, you've succeeded in acquiring a developer sign-up, but the journey of driving usage has just begun. To drive developer *usage*, you must:

1. Reduce friction (once again);
2. Provide context (once again); and
3. Give incentives.

You'll recognize that obstacles 1 and 2 (reduce friction and provide context) are identical to the obstacles we encountered in the Developer Adoption section. These will be recurring themes throughout our quest to improve developer marketing. We've only skimmed the surface of these two themes in this document, but we'll do separate write-ups on both topics in follow-up pieces.

To reduce friction to drive developer sign-ups and increase developer usage, the solution is identical: you want to provide a version of your product that requires no installation and can launch in seconds. We won't repeat the discussion of that, just refer back to the Developer Adoption section in the eBook.

Key Takeaways from the Developer Adoption Section

1. The # of new developer sign-ups is a key metric for stakeholders
2. Two main drivers of developer sign-ups: reduce friction and increase context
3. Today, best-in-class developer marketers provide their communities with 1-click access to their products (reduces friction) within educational experiences (increasing context)

When increasing developer usage, however, the obstacle of providing context is slightly different when compared to providing context during developer adoption. In developer *adoption*, providing context means surrounding a hands-on product trial with educational content — this way, developers can learn and use your product simultaneously. In the context of developer *usage*, however, "provide context" means that when developers launch your product, the product should be in a state where the product's feature set and value are easy to experience.

TTF... what?

TTFHW means "time to first hello world." It is a common acronym used in developer marketing circles to refer to the time it takes for developers to experience a product's value. The lower the TTFHW, the better.

Today, when developers launch your product, they're often greeted with a blank slate — a product that has no data, no prior configuration, etc. Since developers are launching your product as a blank canvas, they need to upload data (to see reports and run queries) and configure the product (to have it perform appropriately) before they can experience your product's value and see it perform. Today, the most innovative developer marketing teams are providing versions of their product pre-configured and pre-populated with data — thus providing the context that places your product in its best light, improving developer usage, and reducing TTFHW times.

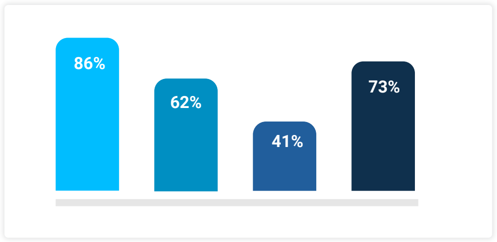
The third obstacle in driving developer usage is incentives. What this means is that developer marketing departments should give their developer communities nudges and incentives to continue increasing their product knowledge and product usage. These nudges are optional, but they do provide an additional, positive pressure for your developer community to deepen their understanding of your product.

Solving the Blank Canvas Problem

Without pre-loaded data and prior configuration, your product's value is harder to experience. With pre-loaded data and proper configuration, however, it comes to life.

Before

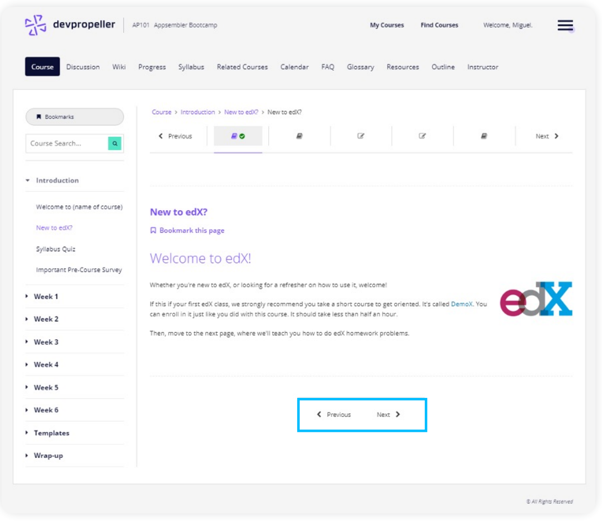
After



The 3 Types of Incentives: Gamification, Badges, and Certifications

Nudges and incentives can be performed and given through gamification, badges, and certifications. Gamification can be something as simple as having a progress tracker above a course to let developers know the time remaining before a course is completed. Badges are issued icons that developers receive upon completing a course — which they can then share on their social media profiles to showcase their competency level with your product. And finally, certifications are verifiable documents that prove a developer's accomplishment and are often associated with a longer, more rigorous learning experience.

Gamification



New to edX?

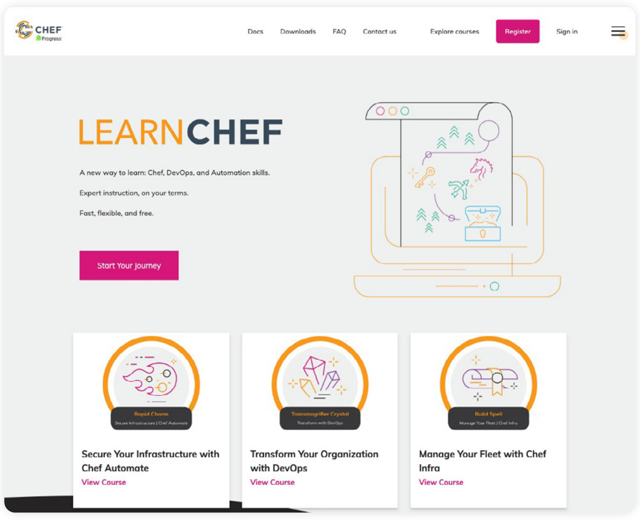
Welcome to edX!

Whether you're new to edX, or looking for a refresher on how to use it, welcome!

If this if your first edX class, we strongly recommend you take a short course to get oriented. It's called DemoX. You can enroll in it just like you did with this course. It should take less than half an hour.

Then, move to the next page, where we'll teach you how to do edX homework problems.

Badges



LEARNCHEF

A new way to learn: Chef, DevOps, and Automation skills.

Expert instruction, on your terms.

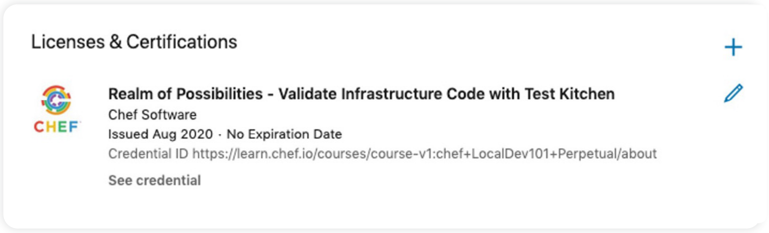
Fast, flexible, and free.

Secure Your Infrastructure with Chef Automate

Transform Your Organization with DevOps

Manage Your Fleet with Chef Infra

Certification



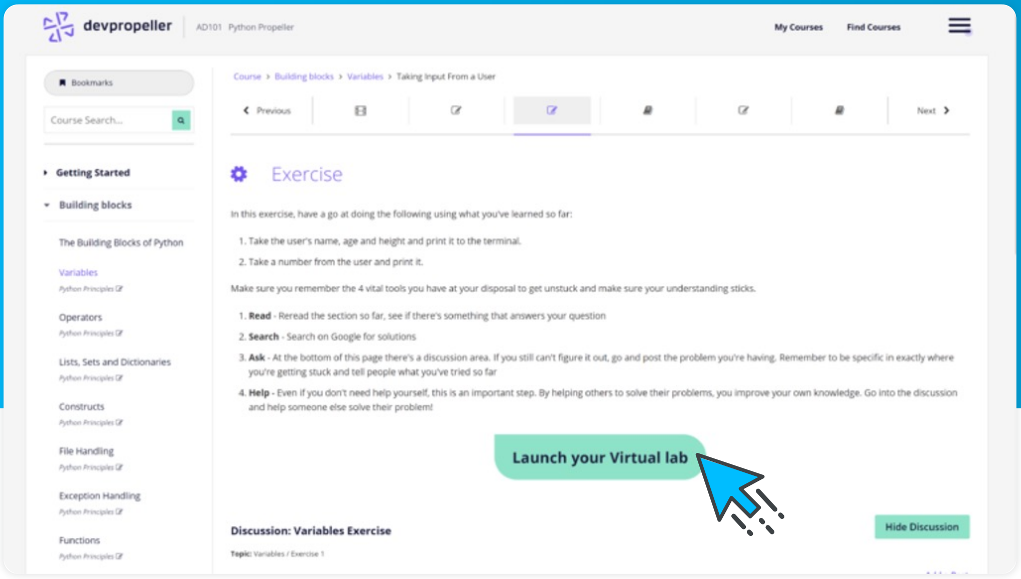
Licenses & Certifications

Realm of Possibilities - Validate Infrastructure Code with Test Kitchen
Chef Software
Issued Aug 2020 · No Expiration Date
Credential ID https://learn.chef.io/courses/course-v1:chef+LocalDev101+Perpetual/about
See credential

4. Lowering Support Costs

Just like your customers, your developer community will also require support. And when compared to traditional customer support questions (e.g. “I need to reset my password.”), your developer community’s questions will be more technical in nature (e.g. “The API isn’t returning the expected result.”).

Using customer support terminology, a developer community’s questions will require, on average, Tier 3 level support. And Tier 3 level support is traditionally provided by higher-cost resources such as sales engineers, product managers, or even developers. What this means is that as your developer community grows, the cost to support them will grow (and result in unexpected demands on critical resources like software engineers or product managers).



Exercise

In this exercise, have a go at doing the following using what you've learned so far:

1. Take the user's name, age and height and print it to the terminal.
2. Take a number from the user and print it.

Make sure you remember the 4 vital tools you have at your disposal to get unstuck and make sure your understanding sticks.

1. **Read** - Reread the section so far, see if there's something that answers your question
2. **Search** - Search on Google for solutions
3. **Ask** - At the bottom of this page there's a discussion area. If you still can't figure it out, go and post the problem you're having. Remember to be specific in exactly where you're getting stuck and tell people what you've tried so far
4. **Help** - Even if you don't need help yourself, this is an important step. By helping others to solve their problems, you improve your own knowledge. Go into the discussion and help someone else solve their problem!

Discussion: Variables Exercise

Topic: Variables / Exercise 1

Opens live version of a product

Since your product is constantly evolving and innovating, ongoing developer support costs will never be zero. But in an ideal scenario, once your developer advocacy team discovers a common issue that developers are having trouble with, they will build a hands-on course that addresses the issues that your developer community is having with your product/platform. For example, if developers are constantly asking questions on your reports API, your developer marketing team can create course content that educates them on your API’s reporting capabilities. For best practice, embed a hands-on virtual training lab (like in the example above) alongside the course content to increase learner retention and increase the probability of lowering your developer support costs.

5. Orchestrating Activities with Marketing

As developers adopt and use your product, orchestrating hand-offs and communications with your marketing and sales departments will become instrumental in driving developer success. For example, what communications should developers get from your company within 2 weeks of signing-up for your product? How do you get a single developer to evangelize your product within the rest of their organization? And when is the appropriate time for your sales team to reach out to the organization about a possible license purchase?

As developers adopt and use your product, orchestrating hand-offs and communications with your marketing and sales departments will become instrumental in driving developer success.

These are questions that are answered through the collaborative effort between developer marketers (who own the relationship with developers), traditional marketers (who own the relationship with the brand), and sales (who own the customer relationship). Every organization will send different material to their developers, but below is a checklist of typical items you may want to consider communicating to your developers.

Items to communicate to your developers:

- Product or platform training
- API status
- Updates to the product/platform
- Developer events
- Sample code
- Alpha or beta testing
- Available virtual training labs
- FAQs
- Changes to sandbox environments
- Product Roadmaps

If all goes as planned, your product's adoption within a customer's technical staff reaches a tipping point that it becomes appropriate for your sales team to contact the prospective customer about a potential license purchase. And similar to how every organization will send different marketing material to their developers, each organization will want to reach-out to prospective customers at different trigger points. Below are examples of potential triggers that may prompt your sales organization to contact a prospective customer for a license purchase, or to approach an existing customer for a license upgrade or training package purchase.

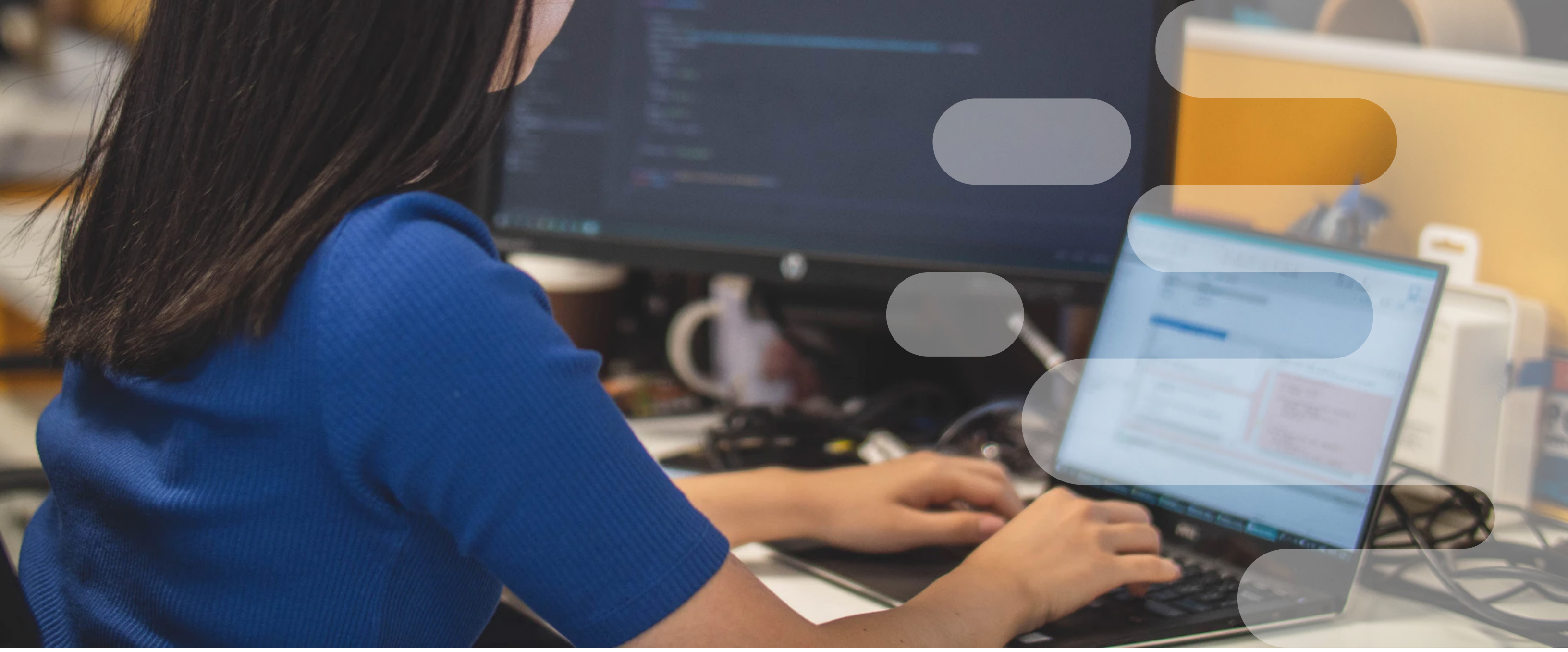
Each organization should want to reach out to prospective customers at different trigger points.

Potential triggers prompting a sales team's action

Different organizations will send different materials to their developers based on what their CRM, Developer Marketing Platform, and Marketing Automation Platform are saying. Below are 5 examples of events that may prompt your sales organization to contact a prospective customer.

CRM
Developer Marketing Platform
Marketing Automation Platform

1. More than 30% of an organization's developers are registered in your product university
2. 10% of an organization's development staff has attended a course in the last 90 days
3. An organization's development staff is regularly submitting support requests
4. 5% of an organization's developers are learning about your product's advanced features
5. An organization's development staff is in the top 25% of virtual lab usage



Chapter 4

For Another Day: Building Broader Market Understanding

In this eBook, we've covered Twilio's "Ask Your Developer" campaign, several facets of developer marketing, and introduced concepts on how developer marketers are advancing the practice of developer marketing today (namely through reducing friction and increasing context). One area that we did not cover in this eBook, however, is the challenge of building broader stakeholder understanding within an organization that currently *does not have* a Developer Marketing initiative. Today, these companies are probably using an ad-hoc team of sales engineers, support representatives, and product managers who collectively help developers become successful with the organization's product — but nobody actually owns the relationship with the developer community.

One area that we did not cover in this eBook, however, is the challenge of building broader stakeholder understanding within an organization that currently does not have a Developer Marketing initiative.

In the scenario outlined above, the company faces the challenge of describing, creating, and structuring the Developer Marketing role: what does the role do, how do you go about creating budget for the role, and what are typical KPIs that the role is held responsible for. Additional pieces and tools will be written and released on these topics in the months to come.

Looking to Learn More?

1. SlashData's Developer Marketing: The Essential Guide
2. The Complete Guide to Virtual Training Labs by Appsembler
3. The 3-Word Billboard That Launched One of America's Fastest Growing Tech Companies by Carmine Gallo, Senior Contributor at FORBES
4. Stack Overflow 2020 Developer Survey
5. Why Market to Developers? Developers Have Influence by Yolanda Fintschenko at DeveloperMedia
6. Why developers have influence today? by Matt Katz
7. Share Knowledge, Not Features: The Secret of Marketing to Developers is to Not Use Marketing by Adam DuVander
8. Your Developer Experience is Made of Content by Adam DuVander
9. First Step to Developer Marketing is to Stop Marketing by Adam DuVander
10. Creative Commons icons

Appsembler for Developer Marketing

Empower developers to build

Learn More

Appsembler is transforming how companies interact with their audiences and unleashes the impact of education to build a more empowered world. Our solutions are driving the world's knowledge transformation forward by uniting traditional learning experiences with the power of immersive, hands-on environments. **Appsembler for Developer Marketing** aligns marketing with the developer persona and enables organizations to build hands-on, educational product experiences for its developer community.