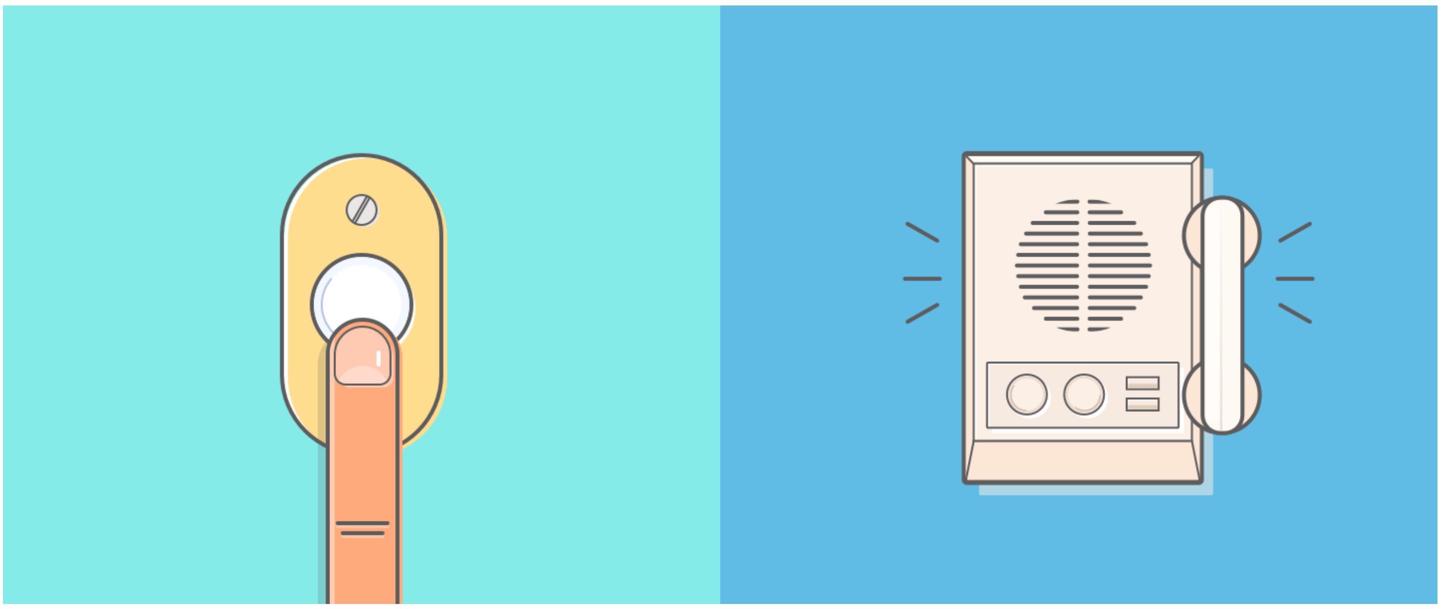
Chargebee Blog SUBSCRIBE →

GROWTH

What are Webhooks and Why You Can't Afford to Ignore Them

SADHANA BALAJI
~ 7 min read | May 26



Subscribe to get more such actionable insights for fast growth subscription businesses.

Let's play a small situational quiz to get your decision-making juices flowing:

You've always wanted to open a nice little lending library in your hometown, as a pet project, where only members are allowed to borrow books. You're finally able to realise that dream and your library is good to go.

BUT, (ah, there's always a "but") you're a busy person and hence you can't be expected to be physically present in the library 24×7. So you decide to get yourself an assistant, who will keep you regularly updated with the goings-on in the library.
Two people sign up for the interview; two *very* different people.

1. There's Mr. Andrew P. Isaac (aka API), a very efficient assistant, who gives you answers – when you ask him. So if you call him up and ask him "Hey! Has anyone subscribed today? Thanks!", you'll get the numbers that you're looking for.

2. Then there's Mr. Webster Hooks (aka Webhooks), who does things a bit differently. As soon as someone subscribes to the library membership, he automatically sends you a message saying, "Hey! Sam

> Harris from Florida has signed up for your Platinum plan. Here's more information about him! And oh, you're welcome!", without you asking for it.

The question: Whom will you hire?

The answer is quite obvious, isn't it? This post is all about reinforcing that answer and proving that you mustn't ignore this pretty cool feature, when you're looking for real-time notifications and data synchronization.

How do webhooks work?

Picture a doorbell system – with a button near the door, and a bell in your living room. When a visitor pushes the button, you hear the doorbell ring, which tells you that someone is at the door – meaning, a signal is sent from the button to the doorbell.

Webhooks work on the same concept. The entrance door is a third-party app/site (webhook provider) which sends a signal when a specified event occurs. And the bell in your living room is what is called the "**listener**" in the software world. The listener is the URL which receives the webhooks and performs a predefined action after that.

Webhooks are basically user defined HTTP callbacks (or small code snippets linked to a web application) which are triggered by specific events. Whenever that trigger event occurs in the source site, the webhook sees the event, collects the data, and sends it to the URL specified by you in the form of an HTTP request. You can even configure an event in one site to trigger an action in another site.

As you would've understood by now, the fundamental difference between API calls and webhooks is that while the former works on **request-based output mechanisms**, and the latter works on **event-based output mechanisms**.

Back to your lending library:

Only this time let's make it an online library. Assume that only your paid subscribers can view your publications, but every subscriber will get a one-month free trial. Suppose you have five individual webhooks setup for five different events as follows:

> **subscription_created:** Sent when a subscription is created

> **subscription_trial_ending:** Sent when a customer's trial period is about to end

card_added: Sent when a card is added for a customer

payment_succeeded: Sent when a payment is successfully collected

subscription_activated: Sent when a subscription is moved from the trial to active state

Let's see how they work in the following scenarios:

Day 1: A visitor X subscribes for a free trial. With the help of the webhook configured for the "subscription_created" event, the source site will send a HTTP POST request to a URL picked by you, with the details of the subscriber, and your customer database will automatically get updated.

Day 24: There's one week for the free trial of X to end. With the webhook for "subscription_trial_ending", you get notified about this. With a bit of coding and setting up, you can display a banner/alert in X's profile, informing him that his trial period is nearing its end and you can also give him a link to update his pricing plan and card details, which you'll be using to charge him at the end of the trial.

Now X really loves your service and doesn't want to stop using it. He promptly enters the card details. This will fire the "card_added" webhook, and the corresponding details will get updated in the customer database.

Day 32: The trial period is over and you charge the card, and voila! the transaction is a success! It's time for the "payment_succeeded" webhook to get triggered, which in turn changes the subscription from the trial state to the active state. This triggers the "subscription_activated" webhook, which then sends an email to X telling him that the payment was successful. While this happens, your customer database also gets updated accordingly.

So all of this will keep happening, with little or no intervention from your side, IF you automate the process with the help of a set of webhooks and codes.

Summing it up, you can use webhooks:

- to simply get to know that an event has taken place
- to ensure that data is in-sync across multiple web applications
- to customize/modify the applications and its functionalities based on needs
- to connect two or more applications where an event in one application would trigger an event in another application, so on and so

forth.

Real-life use cases of webhooks:

MailChimp:

MailChimp provides webhooks to the users, for events such as Subscribes, Unsubscribes, Profile updates, etc. So if you have a weekly newsletter, you'll be immediately notified when someone changes their email id, which you can then update in the subscribers' list in your CRM system.

Shopify:

Shopify provides its own set of useful webhooks that inform you when a product is updated, a cart is updated, a checkout is created, an order is paid for, a refund is initiated, and so on. You don't have to worry about missing an order and not shipping the products on time anymore.

For most aspects of life, the more the merrier. And webhooks are no exception to this statement. And many apps ensure that it gets as merry as possible for you.

Chargebee:

Take **Chargebee**'s "Multiple Webhooks" feature for instance. Need a webhook to notify an email automation tool like Klaviyo about the people who sign up with you? Set it up. Need another webhook to enable more features to users who upgrade their plans? Set it up as well.

Gone are the days where you have to sit and stare at your requirements and try to decide as to which need gets the webhook; now all of them can get a webhook of their own.

If you'd like to know more, here's a post on how different apps make use of the "webhooks magic".

Conclusion:

The benefits of webhooks in a nutshell – **instant, real-time notifications**.

A slightly longer version of that: In a webhook, the pre-requested information is passed to the other apps as and when an event happens, contrary to a typical API where you have to manually "poll" for new data at regular time intervals.

And this brings us to yet another underlying benefit of webhooks – higher efficiency, or in other words, lesser headaches.

You should care because webhooks will be ubiquitous. You should care because they're going to reshape the internet. You should care because webhooks are the next step in the evolution of communicationon the internet and nothing will be left untouched.

Timothy Fitz

I'm pretty sure that you do "care" by this point. And if you're buckling up to get started with "webhook-ing", here're two useful posts to brief you on the inner workings of webhooks illustrated with sample use cases:

1. A short post to get the basics straight.
2. A not-so-short post to ensure that you're doing whatever you're doing, right.

Sadhana Balaji

Marketer. Former staff writer and copy editor at Chargebee's SaaS Dispatch. Overthinker. Outgoing introvert. Finds solace in cocoa, words, and doodles perched in the pages of the notepad.

Webhooks – the Winning Recipe to Get Them Right

Make your data speak human with Metadata

Subscribe

Customer stories

All things subscriptions!