

A Webhook also called a web callback or HTTP push API is a way for an app to provide other applications with real-time information. WebHooks allow you to integrate with external applications or trigger an external workflow when specific events happen in the publisher's application.

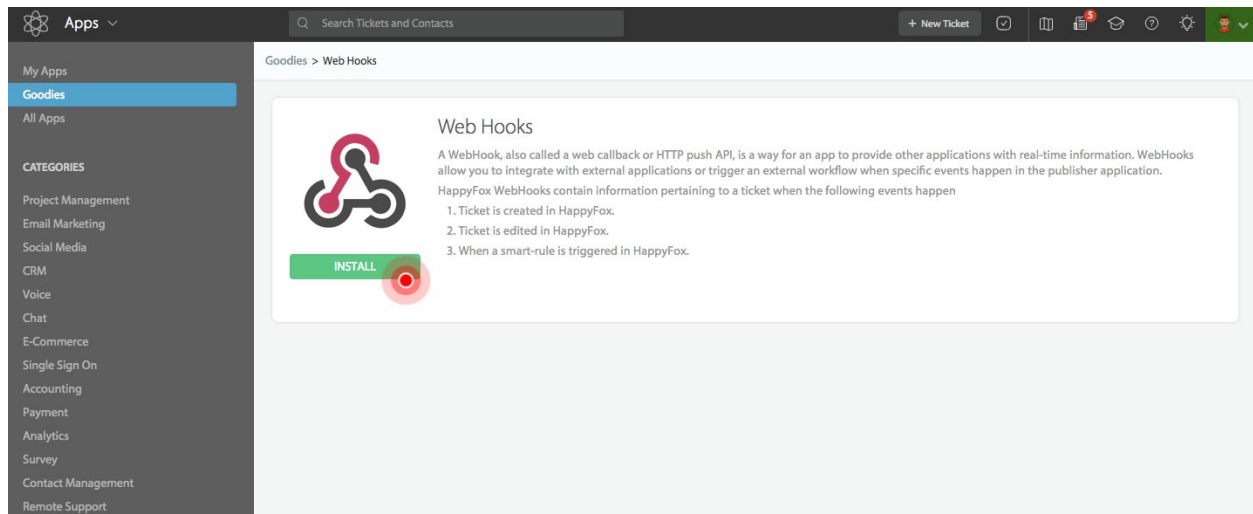
This feature is available for all plans.

HappyFox WebHooks contains information pertaining to a ticket when the following events happen:

- A ticket is created in HappyFox.
- A ticket is edited in HappyFox.
- When a smart-rule is triggered in HappyFox.

**To enable integration with Webhooks, do the following:**

- Log in to HappyFox.
- Go to Apps >> Goodies >> Webhooks.
- Click "Install"



- Make Sure " Webhook Active" drop-down is "**yes**". Also, note down your **webhook key**.

## Webhooks

### Basic Webhook Settings

Application wide Webhook settings (including smartrule actions)

Webhooks active:

No

Key:

09c34e32e75548d4a9310ebb52af5be6

### Setting up a Webhook and Trigger options:

While creating a Webhook URL, you can specify whether the webhook should be triggered during ticket creation or ticket update. Once defined and saved, a webhook URL will be triggered only when a ticket is updated or created or both (As Specified).

## Webhooks

### Basic Webhook Settings

Application wide Webhook settings (including smartrule actions)

Webhooks active:

Yes 

Key:

09c34e32e75548d4a9310ebb52af5be6

### Webhooks

Setup Webhooks here. Add. Delete. Modify.

Active

Name

URL

No webhooks created yet. [Create a new web hook](#)



Add new web hook

What will I receive at the remote end?

You will receive a JSON representation of the ticket details along with information about the type of event (Ticket created or ticket updated). You can receive the call at your end and perform an appropriate action.

## Webhooks and Smart rules

You can now even configure a webhook as a Smart rule action. This way, you can integrate with an external service whenever certain conditions are met on one or more tickets.

### Steps:

- Go to Automate >> Smart Rules.
- Click "Create a Smart Rule".
- Choose "Trigger Webhook" as the action to be performed by the Smart Rule.
- Enter the Webhook URL.
- Define conditions that govern when the Smart Rule should be triggered automatically.
- Enter name and description for the Smart Rule. Choose applicable categories and associate an appropriate Work Schedule.
- Click "Finish Creating a Smart Rule".

Smart Rules						
Search Smart Rule						
Filter						
Name - A to Z						
1 - 10						
ACTIVE	NAME	ACTION	WORK SCHEDULE	CATEGORIES	LAST UPDATED	
	+ Create a Smart Rule					
●	Auto Resolve on Time elapse	Set Status	Default Work Schedule	1	2 months ago	⋮
●	Auto Send email to Contractor if unresponded	Set Category		1	2 months ago	⋮
●	Auto Update Status if priority low	Set Status	Default Work Schedule	1	a month ago	⋮
●	Dynamically set last replied agent.	Set Assignee Dynamically		5	18 days ago	⋮
●	Dynamic assignment	Set Assignee Dynamically		5	a month ago	⋮
●	Priority low	Set Priority		5	a month ago	⋮
●	Re-Prioritise if Due in 12 days	Set Priority	Indian Work Schedule - North east	5	a month ago	⋮
●	Semi Urgent Automation AI	Set Priority		5	14 days ago	⋮
●	Set due date gone past by 11 days	Set Due Date	Support - Samoa Islands	5	a month ago	⋮
●	Status Change	Set Status		1	a month ago	⋮
						1 - 10

## Webhooks log

HappyFox also keeps a log of all the webhooks sent with the time stamps. You can see the webhook details in the log section too. In the event that some webhooks have failed to post, you can select them from the logs and resend them.

Webhook log showing 1 - 4 of 4 [Back to webhooks](#)

☐ [Resend selected](#) [Delete selected](#) --Status-- --Event type-- [Filter](#) 1-4 [◀](#) [▶](#)

Name	Event type	URL	Status	Event time	Last sent time
<input type="checkbox"/> requestbin	Ticket updated	http://requestbin.in/1ep3xce1	Pending	May 16, 2017, 3:09 p.m.	<a href="#">Hide</a>

**Signature** **Response code**

42dfe7c804a91c9b9cadd90be002e86ca12783d5 None

**Payload**

```
{
  "display_id": "#AIR00000061",
  "due_date": null,
  "uuid": "3a9c189cb65149ba87f9244ba2a61ce2",
  "event_type": "Ticket Updated",
  "status_name": "New",
  "subject": "lead ticket ",
  "update": {
    "priority_change": null,
    "duedate_change": null,
    "status_change": null,
    "contact_change": null,
    "category_change": null,
    "update_id": 594,
    "message": {
      "html text": "<html><head><meta charset=\"utf-8\"></head><body>sdfs fg</body></html>",
      "plain text": "sdfs fg\n\n"
    }
  }
}
```

*Sample webhook JSON data*

## Webhooks Key

In the webhook payload, a field called **"uuid"**, will be seen and this field is uniquely generated for every request. We (HappyFox) sign (HMAC-SHA1) every request payload (which contains this **uuid** field) with the key mentioned in the Apps>>Goodies>>Webhooks page, and we display the subsequent signature of each request with the associated payload in our Webhooks log page.

This can be used to verify that this webhook was in fact sent from HappyFox's server. No specific action is required for this with each request. As a side note, this **uuid** field from the request can be stored and compared to ensure that each request is processed only once, if required.

## Manage Webhooks:

Once you have created webhooks, you can choose to manage them.

- Go to Apps >> Goodies >> Webhooks.
- Click "Manage".
- Choose to "Activate/Deactivate", "edit" or "delete" individual webhooks that you have created.

The screenshot displays the 'Webhooks' management page. At the top, there are two buttons: 'Webhook log' and 'Back to integrations'. The main content is divided into two sections. The first section, 'Basic Webhook Settings', includes a toggle for 'Webhooks active' (currently set to 'No') and a 'Key' field with the value '58c4e6b843ad4304a52e3e7c69feb6cc'. The second section, 'Webhooks', contains a table with columns 'Active', 'Name', and 'URL'. One webhook is listed with the name 'Kinglo' and the URL 'https://hyperine.com/webhook/asdaosjdoasjdoaj'. Below the table is a button labeled '+ Add new web hook'. On the right side, there are three informational boxes: 'What are Webhooks?', 'When are Webhooks triggered?', and 'What will I receive at the remote end?'. The 'What are Webhooks?' box explains that webhooks allow configuration of URLs to receive JSON ticket details. The 'When are Webhooks triggered?' box states that webhooks can be triggered during ticket creation or update. The 'What will I receive at the remote end?' box mentions that a JSON representation of ticket details is sent.

**Webhooks**

Webhook log Back to integrations

**Basic Webhook Settings**  
Application wide Webhook settings (including smartrule actions)

**Webhooks active:** No **Key:** 58c4e6b843ad4304a52e3e7c69feb6cc

**Webhooks**  
Setup Webhooks here. Add, Delete, Modify.

Active	Name	URL
<input checked="" type="checkbox"/>	Kinglo	https://hyperine.com/webhook/asdaosjdoasjdoaj

+ Add new web hook

**What are Webhooks?**  
Webhooks allow you to configure URLs that will receive a json representation of the ticket details. If Webhooks are enabled and the respective WebHook URLs are active then a JSON representation of the ticket is sent whenever a ticket is created or updated. Information about the type of event is also sent along with the JSON representation. You can receive the call at your end and perform an appropriate action.

**When are Webhooks triggered?**  
While creating a Webhook URL, you can specify whether the webhook should be triggered during ticket creation or ticket updation. Once defined and saved, a webhook url will be triggered only when a ticket is updated or created or both.

**What will I receive at the remote end?**  
You will receive a JSON representation of the ticket details along with information about whether the JSON information is a