A DEVELOPER'S GUIDE TO MIGRATING APPS FROM HIPCHAT TO SLACK

WELCOME!

If you've built HipChat add-ons, you can recreate them in Slack with powerful custom apps. We're here to help! Here are some tips to get you started.

INTRODUCING SLACK APIs

While Slack's and HipChat's APIs have major similarities, the individual endpoints and features don't have a 1-to-1 mapping. That said, it's helpful to choose the APIs you need based on how your add-on (and soon-to-be app) behaves:

- **Web API** - composed of REST-like methods that allow the pull/push of information to Slack; used to interact with a workspace directly.
- **Events API** - similar to webhook subscriptions in Hipchat; listens for specific events in a workspace.
- **RTM API** - uses WebSockets to receive events and send basic messages to Slack; best if for apps limited to outgoing connections only.

CREATING A SLACK APP

Create a Slack app **on your apps page**; then you'll be presented with your **app configuration page**, the home for adding features, fetching tokens and credentials.

For the **Web API**, add scopes so your app has permission to call intended endpoints after install. **Add various scopes** to your app on the *OAuth & Permissions* page (under *Features*).

For the **Events API**, add subscriptions and endpoints on the *Events Subscriptions* page.

After configuration, install your app on your development workspace and begin coding!

RESOURCES FOR DEVELOPERS

Use **app blueprints** to get started. You'll need to customize, but they're a great place to begin: https://api.slack.com/best-practices/blueprints

Reference our **SDKs** to interact with Web API and Events API adapters. Available in both Node and Python: https://github.com/slackapi

If your app is using the **Hubot framework** with HipChat, fear not! We've maintained a Slack Hubot adapter with documentation detailing how to get up and running in Slack: https://slackapi.github.io/hubot-slack/

Additional tools can be found at **api.slack.com/tools**, providing outlets for sample code, app testing and links to community-built tools, such as Java and .NET wrappers around our APIs.

SUPPORT

Do not hesitate to reach out for help. **Email** developers@slack.com, where we have dedicated support agents on stand-by.

You can also contact us on **Twitter** @SlackAPI, or join the **bot developer hangout**: https://community.botkit.ai/

General support **documentation**, **changelog** and platform updates can always be found on api.slack.com.

SUPPLEMENTAL MATERIAL

Useful for admins or out-of-the-box app migration; complementary to materials focused on rebuilding custom apps.

Atlassian Migration Guide:
www.atlassian.com/partnerships/slack/migration

Getting Started: Slack + Atlassian:
https://slack.com/migration/atlassian

HELPFUL TIPS FOR COMMON FEATURES

Sending a message: You'll probably want to send notifications and messages to users in Slack. HipChat allows add-ons to send a message to a room by calling their REST API and configuring the `send_notification` permission. There are two different ways to post a message in Slack:

1. **Incoming webhooks** - generate a unique URL on your app configuration page that will let you post a message to a single channel.
2. **Web API** - the `chat.postMessage` method enables you to send a message into a specific channel; calling this method requires your app to have the `chat.write.user` permission scope.

Responding to user invocation: Your app will need to receive invocations from users. HipChat uses specified webhook triggers. In Slack, you can register **slash commands** by specifying a trigger in your app configuration. When a user is in Slack, they'll `/<trigger>` and your app will receive event information and respond appropriately.

Slack commands can be less intuitive for nontechnical users. **Actions** enable your app's users to turn a Slack message into a ticket, task or workflow: https://api.slack.com/actions

Gathering input: HipChat and Slack both offer **dialogs**, interactive modals that capture multiple pieces of information. You can use them to build a robust form inside Slack, enabling apps to quickly collect structured information. Check out a sample approval app we created using dialogs: api.slack.com/best-practices/blueprints/approval-workflows