Mattermost’s Approach to Layered Extensibility in Open Source

Corey Hulen / Co-founder and CTO, Mattermost, Inc.
Feb 2019
Mattermost as an OS

- Mattermost as a platform from the very beginning

Mattermost Platform

AI
BI
Smart workflows
Salesforce
Real Estate
Finance
Healthcare
Etc
Etc
Embedding Applications inside of Mattermost
Mattermost has Layers!

- Slash Commands and Webhooks
- REST API/Bots
- Plugins
- Open Source
- Open Data
Mattermost is Open Source

- Typical open source goodness
- Fork, change what you want (heck change everything!)
- Hard, takes a lot of resources
- People love the idea but in reality very rarely happens
- 100% of what you see in the UI is accessible from our RESTful JSON web service API
Matterhorn is a terminal client for the Mattermost chat system. 
https://github.com/matterhorn-chat/matterhorn
Mattermost is Open Data

- You own 100% of your data
- Easy extraction and manipulation with SQL
- Easily chart, report, analyze
- No proprietary data models; nothing hidden
Developer Toolkit

- Incoming/Outgoing Webhooks
- Slash Commands
- Bots
- REST API
- Client-side Plugins
- Server-side Plugins

https://developers.mattermost.com
Webhooks

- **Incoming Webhooks**
  - Post messages to public, private and direct message channels
  - Simple HTTP POST request to a URL
  - Designed to easily allow you to post messages
  - Slack compatible

- **Outgoing Webhooks**
  - Post automated responses to posts made by your users
  - Uses trigger words
  - Simple HTTP POST request to a web service
  - Slack compatible
Webhooks Example

ALARM: "corp-mattermost-CPU-Utilization" in US East (N. Virginia)

You are receiving this email because your Amazon CloudWatch Alarm "corp-mattermost-CPU-Utilization" in the US East (N. Virginia) region has entered the ALARM state, because "Threshold Crossed: 1 datapoint [82.4641844956938 (03/11/18 18:40:00)] was greater than or equal to the threshold (50.0)." at "Saturday 03 November, 2018 18:45:57 UTC".

View this alarm in the AWS Management Console:
https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#s=Alarms&alarm=corp-mattermost-CPU-Utilization

Alarm Details:

github 07:10

PLT-6905 - Updating channel header design

#new-comment #6789 in mattermost/platform by Hanzei on 2017-07-06T11:09:35Z

On the Spinmint server clicking on "Recent Mentions" does nothing. Is this intended?
Slash Commands

- Easy way integrate external applications
- Similar to outgoing webhooks, but can be used in any channel
- Simple HTTP POST request to a URL, process response back into channel
- Both built-in and custom slash commands
- Slack compatible
Matterbuild example

`cpanato 21:18`

ok
for 5.5: pr-mattermost-server-9799
for master: pr-mattermost-server-9798

`/mb cut 5.5.0-rc1`

---

`Matterbuild BOT 11:12`  
Matterbuild  
Cut Release  
Release 5.5.0-rc1 is on the way.

---

`Mattermost Release BOT 11:15`  
Release 5.5.0-rc1 has started building.
Bots

- Official and community-built drivers for our web service API
  - Javascript
  - Golang
  - PHP
  - Python
  - [https://api.mattermost.com](https://api.mattermost.com)

- Golang bot sample code
  - Uses the go driver to interact with the Mattermost server
  - Login to the server, create a channel, modify a user, etc
  - [https://github.com/mattermost/mattermost-bot-sample-golang](https://github.com/mattermost/mattermost-bot-sample-golang)
Sample Bot Example

```go
func LoginAsTheBotUser() {
    if user, resp := client.Login(USER_EMAIL, USER_PASSWORD); resp.Error != nil {
        println("There was a problem logging into the Mattermost server. Are you sure ran the setup steps from the README.md?")
        PrintError(resp.Error)
        os.Exit(1)
    } else {
        botUser = user
    }
}

func SendMsgToDebuggingChannel(msg string, replyToId string) {
    post := &model.Post{
    post.ChannelId = debuggingChannel.Id
    post.Message = msg

    post.RootId = replyToId

    if _, resp := client.CreatePost(post); resp.Error != nil {
        println("We failed to send a message to the logging channel")
        PrintError(resp.Error)
    }
}
```
Hubot Example

Corey Hulen  9:34 AM
hubot lets ship it

hubot  9:34 AM

🔗 http://shipitsquirrel.github.io/images/ship%20it%20squirrel.png
Plugins Overview

- **Customize User Interfaces**
  - Write a web app plugin to add to the channel header, sidebars, main menu and more. Register your plugin against a post type to render custom posts or wire up a root component to build an entirely new experience. All this is possible without having to fork the source code and rebase on every Mattermost release.

- **Launch Tightly Integrated Services**
  - Launch and manage server plugins as services from your Mattermost server over RPC. Handle events via real-time hooks and invoke Mattermost server methods directly using a dedicated plugin API.

- **Extend the Mattermost REST API**
  - Extend the Mattermost REST API with custom endpoints for use by web app plugins or third-party services. Custom endpoints have access to all the features of the standard Mattermost REST API, including personal access tokens and OAuth 2.0.
Customize User Interfaces (Client Side)

- registerRootComponent(component)
- registerPopoverUserAttributesComponent(component)
- registerPopoverUserActionsComponent(component)
- registerLeftSidebarHeaderComponent(component)
- registerBottomTeamSidebarComponent(component)
- registerPostMessageAttachmentComponent(component)
- registerChannelHeaderButtonAction(icon, action, dropdownText, tooltipText)
- registerPostTypeComponent(type, component)
- registerMainMenuAction(text, action, mobileIcon)
- registerPostDropdownMenuAction(text, action, filter)
- registerPostDropdownMenuComponent(component)
- registerFileUploadMethod(icon, action, text)
- registerFilesWillUploadHook(hook)
- unregisterComponent(componentId)
- registerReducer(reducer)
- registerWebSocketEventHandler(event, handler)
- unregisterWebSocketEventHandler(event)
- registerReconnectHandler(handler)
- unregisterReconnectHandler()
- registerMessageWillFormatHook(hook)
Channel Header Button Example

- Add a button to the channel header
- If there is more than one button, a dropdown menu is created
- Great for things like video conferencing integration
- Our built-in Zoom integration is actually a plugin

```javascript
initialize(registry, store) {
  registry.registerChannelHeaderButtonAction(
    <Icon/>
  ,
    (channel) => {
      startMeeting(channel.id)(store.dispatch, store.getState);
    },
    'Start Zoom Meeting'
  );
}```
Zoom Integration adding a Header Button
Override Post Rendering

- Register a component to render a custom body for posts
- Great for rich integrations with custom applications
- More than simple text or a screenshot
- The Zoom plugin uses this to render a custom post

```javascript
initialize(registry, store) {
  registry.registerChannelHeaderButtonAction(
    <Icon/>,
    (channel) => {
      startMeeting(channel.id)(store.dispatch, store.getState);
      'Start Zoom Meeting';
    },
    'custom_zoom', PostTypeZoom);
}
```
Zoom Integration custom rendering
Render Component on the Team Sidebar

- Register a component fixed to the bottom of the team sidebar
- Used in our Github plugin

```javascript
class PluginClass {
  async initialize(registry, store) {
    registry.registerReducer(Reducer);
    await getConnected(true)(store.dispatch, store.getState);
    registry.registerLeftSidebarHeaderComponent(SidebarHeader);
    registry.registerBottomTeamSidebarComponent(TeamSidebar);
    registry.registerPopoverUserAttributesComponent(UserAttribute);
    static propTypes = {
      show: PropTypes.bool.isRequired,
      theme: PropTypes.object.isRequired,
    };

    render() {
      if (!this.props.show) {
        return null;
      }
      return (<SidebarButtons
        theme={this.props.theme}
        isTeamSidebar={true}
      />
    );
  }
}
```
Github Integration
Rewrite the Message Client Side

- Register a hook that will be called before a message is formatted into Markdown
- Used in the Walltime plugin

```javascript
export default class Plugin {
  // eslint-disable-next-line no-unused-vars
  initialize(registry, store) {
    registry.registerMessageWillFormatHook((post) => {
      const {message} = post;
      const state = store.getState();

      const {text} = nlpResult;
      newMessage = `${newMessage.replace(text, '\${text}\ *(${formattedDisplayDate})')}`;  
    });

    return newMessage;
  }
```
Walltime Plugin

Example

User in the timezone `America\Toronto` post the following message:

Let’s meet today at 10am

A user in another timezone, ie. `Europe\Berlin` will see:

Let’s meet today at 10am (Thursday, August 16, 2018 4:00 PM CEST)

Another example:

Let’s meet today at noon

the user reading the message in another timezone will see:

Let’s meet today at noon (Thursday, August 16, 2018 7:00 AM CEST)
Tightly Integrated Services (Server Side)

- `OnActivate()` error
- `Implemented()` (string, error)
- `OnDeactivate()` error
- `OnConfigurationChange()` error
- `ServeHTTP(c *plugin.Context, w http.ResponseWriter, r *http.Request)`
- `UserHasBeenCreated(c *plugin.Context, user *model.User)`
- `UserWillLogIn(c *plugin.Context, user *model.User)` string
- `UserHasLoggedIn(c *plugin.Context, user *model.User)`
- `MessageWillBePosted(c *plugin.Context, post *model.Post)` (*model.Post, string)
- `MessageWillBeUpdated(c *plugin.Context, newPost, oldPost *model.Post)` (*model.Post, string)
- `MessageHasBeenPosted(c *plugin.Context, post *model.Post)`
- `MessageHasBeenUpdated(c *plugin.Context, newPost, oldPost *model.Post)`
- `ChannelHasBeenCreated(c *plugin.Context, channel *model.Channel)`
- `UserHasJoinedChannel(c *plugin.Context, channelMember *model.ChannelMember, actor *model.User)`
- `UserHasLeftChannel(c *plugin.Context, channelMember *model.ChannelMember, actor *model.User)`
- `UserHasJoinedTeam(c *plugin.Context, teamMember *model.TeamMember, actor *model.User)`
- `UserHasLeftTeam(c *plugin.Context, teamMember *model.TeamMember, actor *model.User)`
Rewrite the Message Server Side

- Lots of opportunities to hook the message
- Before it enters into the database
- You can reject messages
- You can also send back ephemeral messages
- After it enters into the database

```go
func (p *Plugin) MessageWillBePosted(c *plugin.Context, post *model.Post) (*model.Post, string) {
    links := p.links.Load().([]*Autolinker)
    postText := post.Message
    offset := 0
    markdown.Inspect(post.Message, func(node interface{}) bool {
        switch node.(type) {
        // never descend into the text content of a link/image
        case *markdown.InlineLink:
            return false
        case *markdown.InlineImage:
            return false
        case *markdown.ReferenceLink:
            return false
        case *markdown.ReferenceImage:
            return false
        }
    }
}
```
Autolink Plugin

https://mattermost.atlassian.net/browse/MM-10001

Corey Hulen 9:50 AM

Write a message...
Implement a http.Handler on the Server

- ServeHTTP allows the plugin to implement the http.Handler interface.
- Requests destined for the /plugins/{id} path will be routed to the plugin.

```go
func (p *Plugin) ServeHTTP(c *plugin.Context, w http.ResponseWriter, r *http.Request) {
    var action *Action
    json.NewDecoder(r.Body).Decode(&action)

    if action == nil {
        encodeSphericalMessage(w, "WelcomeBot Error: We could not decode the action")
        return
    }

    data := &MessageTemplate{}
    var err *model.AppError

    func (p *Plugin) handleSupport(action *Action, data *MessageTemplate) {
        p.joinChannel(action, "peer-to-peer-help")
        p.joinChannel(action, "bugs")

        template, _ := template.New("Response").Parse(`
### Awesome, I have added you to the following support related channels!
~peer-to-peer-help - General help and setup
~bugs - To help investigate or report bugs
`)
    }
```
WelcomeBot Handling Custom Actions

In addition to the basics discussed above the Contributor Code of Conduct also applies.

Updated March 5, 2016

I can help you get started by joining you to a bunch of existing channels! Which types of channels would you like to join?

- I'm interested in support
- Developing on Mattermost
- I don't know?
Thank you
Mattermost is hiring!

https://mattermost.com/careers
Mattermost Mentor Program

● Shepherding
  ○ Helping someone get involved in the Mattermost community
  ○ Need help submitting your first pull request?
  ○ Want some general advice?
  ○ Two 1-1 sessions with a core contributor

● Mentorship
  ○ Focused on bringing more people into the Mattermost community
  ○ Application process
  ○ 3 month program
  ○ Minimum commitments must be met
  ○ Paired with a core contributor for the entire time
  ○ Weekly 1-1 sessions with a core contributor

Interested? Talk with me afterwards!
Questions?
How to reach me?

- Mattermost: @corey on https://community.mattermost.com
- Twitter: @corey_hulen
- Email: corey@hulen.com