Hammerspoonのご紹介

11/13 勉強会

目次

01　Hammerspoon とは
02　OS操作の入出力範囲がスゴい
03　激安左手デバイス作ってみた
04　その他設定例
05　大AI時代こそHammerspoonを！

はじめに：対応OSについて

macOS	Windows	Linux
対応	非対応	非対応

はじめに：対応OSについて

macOS

対応

Windows

非対応

Linux

非対応

Windows ユーザー向けの互換設定を用意

Hammerspoon とは：Lua による macOS 自動化ツール

設定ファイル
init.lua
Lua で記述

→

アプリ本体
Hammerspoon
hs.window などの
操作用関数

→

操作対象
macOS
ウィンドウ・アプリ・入力

誕生	2014年、同種のツール Mjolnir から派生して誕生
ライセンス	オープンソース・無料
導入	brew install --cask hammerspoon ＋ アクセシビリティ許可

Lua とは：組み込み用の軽量スクリプト言語

項目	内容
誕生	1993年、リオデジャネイロ・カトリック大学（ブラジル）
特徴	小さな文法、高速な処理系、C との連携が容易
採用例	Neovim / Roblox / World of Warcraft / nginx / Redis

例：⌘⌥H でアラートを表示

```
hs.hotkey.bind({"cmd","alt"}, "h", function()
hs.alert.show("Hello")
end)
```

入力：豊富なトリガー

入力：豊富なトリガー

キー入力

⌘⌥H などの同時押し
キーの長押し

`hs.hotkey`

マウス

クリック・スクロール

`hs.eventtap`

バッテリー

残量 20% 未満
電源アダプタの接続

`hs.battery.watcher`

USB 接続

キーボードの
接続・取り外し

`hs.usb.watcher`

Wi-Fi 変更

特定 Wi-Fi への
接続・切断

`hs.wifi.watcher`

スリープ/復帰

スリープ復帰
画面ロック

`hs.caffeinate`

アプリ起動・切替

特定アプリの起動
フォーカス

`hs.application`

ファイル変更

ダウンロードフォルダに
ファイル追加

`hs.pathwatcher`

ディスプレイ変更

外部モニターの接続
配置変更

`hs.screen.watcher`

タイマー

5秒ごと
毎日 18:00

`hs.timer`

URL を開く

ブラウザ等から
専用 URL を開く

`hs.urlevent`

ターミナル

コマンドから
処理を呼び出す

`hs.ipc`

出力：OS 操作に幅広く対応

出力：OS 操作に幅広く対応

ウィンドウ配置

最前面を
画面の左半分へ

`hs.window`

アプリ制御

VSCode を起動
前面化・非表示

`hs.application`

デスクトップ切替

右隣の
デスクトップへ移動

`hs.spaces`

キー・マウス操作

⌘V を自動入力、
指定位置をクリック

`hs.eventtap`

クリップボード

履歴から選んで
貼り付け

`hs.pasteboard`

シェル実行

brew upgrade、
結果を表示

`hs.task`

AppleScript

ミュージックを
一時停止

`hs.osascript`

アラート表示

画面中央に
「保存しました」

`hs.alert`

メニューバー

ON/OFF の状態を
アイコン表示

`hs.menubar`

選択リスト

定型文・URL を
一覧から選んで実行

`hs.chooser`

オーディオ制御

音量ミュート
出力先の切替

`hs.audiodevice`

スリープ抑止

発表中は画面を
スリープしない

`hs.caffeinate`

組み合わせ例 1/2

社内 Wi-Fi に接続 `hs.wifi.watcher`	→	音量をミュート `hs.audiodevice`
外部ディスプレイを接続 `hs.screen.watcher`	→	指定のウィンドウ配置とサイズに `hs.window`
スリープから復帰 `hs.caffeinate.watcher`	→	Slack を前面化 `hs.application`
毎日 18:00 `hs.timer.doAt`	→	日報リマインドを表示 `hs.alert`

組み合わせ例 2/2

USB キーボードを接続 `hs.usb.watcher`	→	入力ソースを英字へ `hs.keycodes`
Zoom を起動 `hs.application.watcher`	→	音楽を一時停止 `hs.eventtap`
ハイパーキー割り当て + Space `hs.eventtap`	→	定型文を選んで貼り付け `hs.chooser`
長いビルドが終了 `hs.ipc`	→	画面中央に「完了」と表示 `hs.application`

組み合わせ例 ぶっとび編

バッテリー残量 5% 未満

電源が落ちる直前を検知

`hs.battery.watcher`



アラート・全アプリに ⌘S を送信

`hs.eventtap`

組み合わせ例 ぶっとび編

バッテリー残量 5% 未満
電源が落ちる直前を検知
`hs.battery.watcher`

→ **アラート・全アプリに ⌘S を送信**
`hs.eventtap`

社内の Wi-Fi が切断
退勤した瞬間をネットワーク名で検知
`hs.wifi.watcher`

→ **全アプリを強制終了**
`hs.application`

組み合わせ例 ぶっとび編

バッテリー残量 5% 未満
電源が落ちる直前を検知

`hs.battery.watcher`

→

アラート・全アプリに ⌘S を送信
`hs.eventtap`

社内の Wi-Fi が切断
退勤した瞬間をネットワーク名で検知

`hs.wifi.watcher`

→

全アプリを強制終了
`hs.application`

Slack の未読が増える
Dock のバッジの数字を監視

`hs.axuielement`

→

画面全体がビカビカと光り始める
`hs.canvas`

Hammerspoonを使って
パワポ操作をしていました

MIDI とは：楽器間通信のための古き良き規格

「どのキーを、どの強さで押したか」送る

1 バイト目

0x90

押した合図

「ノートオン」と
チャンネル番号

2 バイト目

36

キーの番号

どのキー・パッドか
（0〜127）

3 バイト目

100

押した強さ

「ベロシティ」
（0〜127）

MIDI とは：楽器間通信のための古き良き規格

「どのキーを、どの強さで押したか」送る

1 バイト目

0x90

押した合図

「ノートオン」と
チャンネル番号

2 バイト目

36

キーの番号

どのキー・パッドか
（0〜127）

3 バイト目

100

押した強さ

「ベロシティ」
（0〜127）

PC接続：USB-MIDI 対応機は macOS が標準で認識、ドライバ不要

Hammerspoon：MIDI入力に対応！キーの番号ごとに動作を割り当てられる！

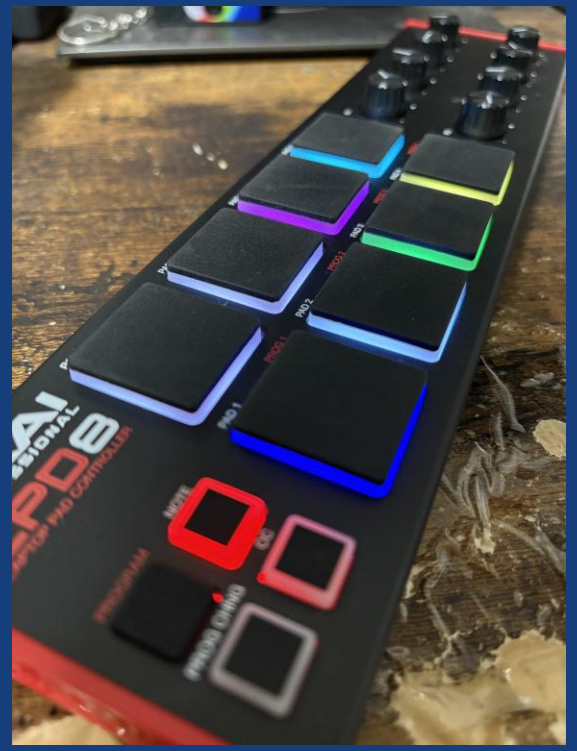
MIDI搭載の楽器を左手デバイス化できるのでは？

（しかもめちゃくちゃ安価に）

MIDI搭載の楽器を左手デバイス化できるのでは？

（しかもめちゃくちゃ安価に）

MIDIパッド（サンプラー）



指でボタンを叩く

↓

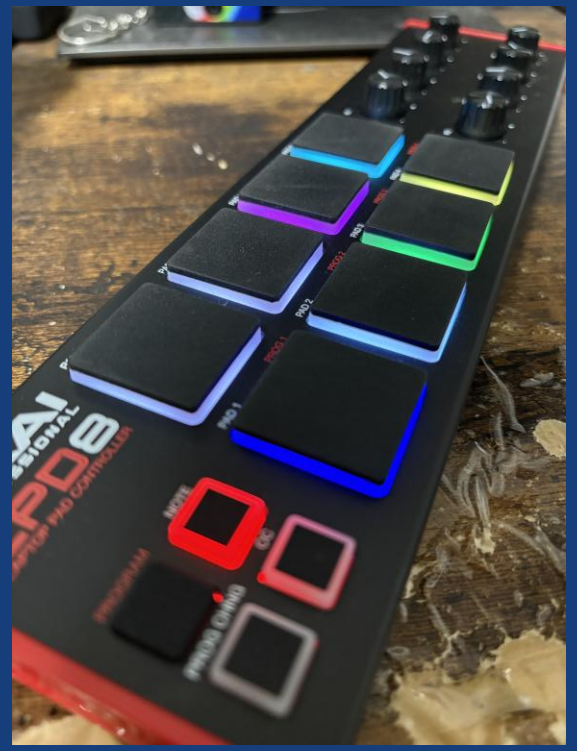
ドラムを演奏

曲をポン出し

MIDI搭載の楽器を左手デバイス化できるのでは？

（しかもめちゃくちゃ安価に）

MIDIパッド（サンプラー）



指でボタンを叩く

↓

ドラムを演奏

曲をポン出し

左手デバイスに使える楽器は意外とある

2オクターブの鍵盤

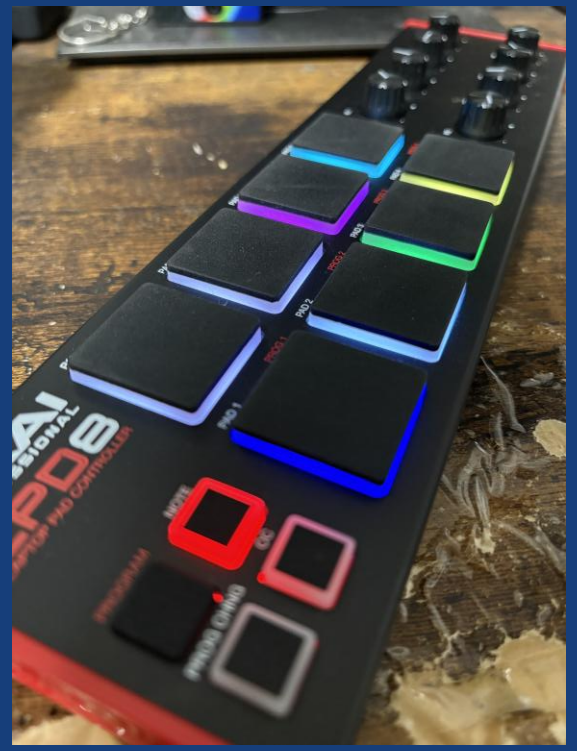
25鍵 ＝ ショートカット25個

C#（ド＃）弾いたらUnity立ち上げられる！！

MIDI搭載の楽器を左手デバイス化できるのでは？

（しかもめちゃくちゃ安価に）

MIDIパッド（サンプラー）



指でボタンを叩く

↓

ドラムを演奏

曲をポン出し

左手デバイスに使える楽器は意外とある

2オクターブの鍵盤

25鍵 ＝ ショートカット25個

C#（ド＃）弾いたらUnity立ち上げられる！！

押し入れから電子ピアノを出しましょう

電子ドラムをデスク左側に置きましょう

MIDIパッドに操作を割り当ててみる

項目	内容
機材	SMC-PAD Pocket
パッド数	4×4＝16（4プリセット）
価格	アリエクで3000円
割り当て	パッドの番号 （36〜51）ごとに動作
配置	下段：アプリ切り替え 上段：プレゼン・画面

MIDIパッドに操作を割り当ててみる

スライド 戻る	スライド 進む	別モニター 転送	通知 センター
左半分	右半分	最大化	同一アプリ 巡回
Slack	Chrome	Claude	メモ
VSCode	PyCharm	cmux	Sourcetree

項目	内容
機材	SMC-PAD Pocket
パッド数	4×4＝16（4プリセット）
価格	アリエクで3000円
割り当て	パッドの番号 （36〜51）ごとに動作
配置	下段：アプリ切り替え 上段：プレゼン・画面

他にも設定作ってみた（hammerspoon-variants）

windows-compat

Windows互換

一時的に Mac を使う Windows ユーザー向け
Ctrl 系・Win キー・Alt+Tab を同じ位置のキーで

window-manager

ウィンドウマネージャ

キーボードで管理したい、自由に割り当てたい人
⌥+数字でアプリ切替、タイル型風の機能

シェルスクリプトで簡単適用

好きなものを選択・組み合わせ可能　/　オプションをつければ元に戻る

設定 1 / 2　windows-compat

Windows互換：Mac固有のキーをWindowsっぽく割り当て

左下隅
^ → Ctrl 系

Ctrl の右
⌥ → Win 系

スペースの横
⌘ → Alt 系

Macで押すキー	動作	Windowsでいう
^C / ^V / ^Z など	⌘ に変換	Ctrl+C など
⌘Tab / ⌘F4	ウィンドウ切替 / 閉じる	Alt+Tab / Alt+F4
⌥← / ⌥→ / ⌥↑	左半分 / 右半分 / 最大化	Win+← / → / ↑
⌥1〜⌥9	アプリを起動・前面化	Win+1〜9
⌥V	クリップボード履歴	Win+V

設定 2 / 2　window-manager

ウィンドウマネージャ：アプリ呼び出しとサイズ切替に特化

キー	動作
⌥1〜⌥8	アプリを起動 / 前面化。前面なら次のウィンドウへ（別デスクトップも対象）
⌥9 / ⌥0	左 / 右に配置。続けて押すと幅が 1/2 → 2/3 → 1/3
⌥= / ⌥-	最大化 / 動かす前の位置と大きさに戻す
⌥G	6×4 のグリッドを描画し、好きな位置、好きなサイズに切替
⌥M	Dockにしまう。もう一度押すと元に戻る

windows-compat と併用時は ⌥ → ^⌥ に自動で切り替え（例：^⌥1 で VS Code）

なぜ紹介したか：AI時代に相性バッチリだから

GUI 設定型ツール

設定：コンソール上でポチポチ操作
AI 利用：手順確認のみ
管理：プリセット機能等に依存？

Hammerspoon

設定：Lua で記述
AI 利用：AI がコードを直接生成
管理：git で履歴・共有

なぜ紹介したか：AI時代に相性バッチリだから

GUI 設定型ツール

設定：コンソール上でポチポチ操作
AI 利用：手順確認のみ
管理：プリセット機能等に依存？

Hammerspoon

設定：Lua で記述
AI 利用：AI がコードを直接生成
管理：git で履歴・共有

- コーディングエージェントの登場で
  Luaを覚えなくてはならない障壁が緩和！
- その気になればあらゆる便利ツールの
  再現、カスタマイズが可能！
- 是非その自由度の高さを体感して欲しい！

ご清聴ありがとうございました