

卒業論文 2019年度（令和01年度）

Hyper Launcher
ホットキー型アプリケーションランチャーの
利便性を高める研究

慶應義塾大学 環境情報学部

佐藤 佑哉

増井俊之研究会

2020年1月

Hyper Launcher

ホットキー型アプリケーションランチャーの 利便性を高める研究

論文要旨

ホットキー型のアプリケーションランチャーは、特定のキーの入力のみでアプリケーションを起動することができるとても強力なユーティリティソフトウェアの一種であるが、使いこなすのが難しく広く普及しているとは言い難い。この形式のランチャーがあまり活用されていないのは、その設定の煩雑さや設定したキーの割り当てを記憶しなければならないというハードルの高さにある。これらの問題はアプリケーションの登録方法やその操作の柔軟性といったソフトウェアの工夫次第で改善できると考えられる。

「Hyper Launcher」はホットキーに対してアプリケーションを登録するという形式を採用することで、ユーザーの導入障壁を取り除き、単一のホットキーによる複数アプリケーションの操作を可能にした新しいランチャーシステムである。これにより既存の問題点が解消されるだけでなく、新しいランチャーの使い方が可能となる。本論文ではアプリケーションランチャーの種類や特徴についてまとめた上で、「Hyper Launcher」の設計や実装、その応用例について述べ、研究の発展性について考察する。

キーワード

ホットキー, ランチャー, インタフェース, ユーザビリティ

目次

第1章 序論	1
1.1 背景	2
1.1.1 形式と比較	2
1.1.2 既存のアプリケーション	4
1.1.3 運用例	5
1.1.4 問題点	5
1.2 本研究の目的	6
1.3 本論文の構成	6
第2章 システムの提案	7
2.1 設計	8
2.2 基本操作	8
2.2.1 登録と起動	8
2.2.2 登録の変更と解除	8
2.2.3 複数登録と操作	9
第3章 実装	12
3.1 システム構成	13
3.2 クライアント	13
3.3 データストア	14
第4章 実運用及び評価	16
4.1 筆者の運用	17
4.2 第三者の意見	17
4.2.1 登録について	17
4.2.2 覚えやすさについて	17
4.2.3 ホットキーの有用性について	17
4.2.4 問題点	18
第5章 議論	19
5.1 関連研究	20
5.2 展望	21
第6章 結論	22

謝辞	24
参考文献	25

目次

1.1	Dock	2
1.2	Dock 内のメニュー	3
1.3	Spotlight	4
1.4	PMenu	5
1.5	Snap	6
2.1	Hyper Launcher の起動画面	9
2.2	ドラッグアンドドロップによる登録	10
2.3	追加ボタンから登録	10
2.4	登録の変更	11
2.5	登録の解除	11
3.1	システムの構成	13

第1章 序論

本章では、本研究の背景と目的、及び本論文の構成について述べる。

1.1 背景

パーソナルコンピュータの普及に伴いアプリケーションの数はますます増加しており、個人が使うアプリケーションの数も自ずと増えてきた。数多くのアプリケーションの中から目的のアプリケーションを起動する方法としてアプリケーションランチャーは非常に有用であり、現在ではオペレーティングシステムにも標準で搭載されるようになっている。またサードパーティ製としても多くのソフトウェアが開発/公開されており、その形式は様々である。その中でも、ホットキーを利用したアプリケーションランチャーは、特定のキーの入力のみでアプリケーションを起動することができる強力なランチャーである。しかし未習熟者には使いこなすのが難しく、あまり普及していないのが現状である。

1.1.1 形式と比較

現在使用されているアプリケーションランチャーの形式を大きく4つに分類し、それぞれの利点/欠点を比較していく。なお分類については Wikipedia のランチャーについてのページ¹を参考にした。

(1) パレット型

画面上に固定されたパレット型のエリアにアプリケーションを登録し、マウスによるクリックで起動する形式のランチャー。macOS における Dock(図 1.1) がこれにあたる。デスクトップに常駐しているため誰でも簡単に使うことができる。しかしそのエリアは有限であり、多くのアプリケーションを登録するためには、ボタンの数を増やしたりそれぞれを小さく表示したりする必要がある。その数が増えれば増えるほど操作ステップが増えるため、数個から数十個のよく使うアプリケーションを登録して使用することが推奨される。

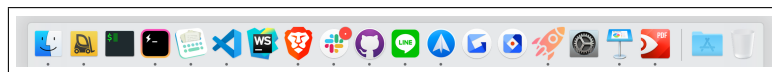


図 1.1: Dock

(2) メニュー型

Dock などにメニューを設け、マウスを使用して登録したアプリケーションを起動する形式のランチャー。ホットキーと組み合わせて使用されるものもある。複数のアプリケーションを一つにまとめることで場所を節約できるだけでなく、階層化によって自分の使いやすいように整理することもできる。しかし、項目や階層が増えれば目的のアプリケーションに辿り着くまでの操作ステップが増えることになるため、使用頻度の少ないアプリケーションを整理するのが一般的である。

¹<https://ja.wikipedia.org/wiki/ランチャー#コンピュータ>

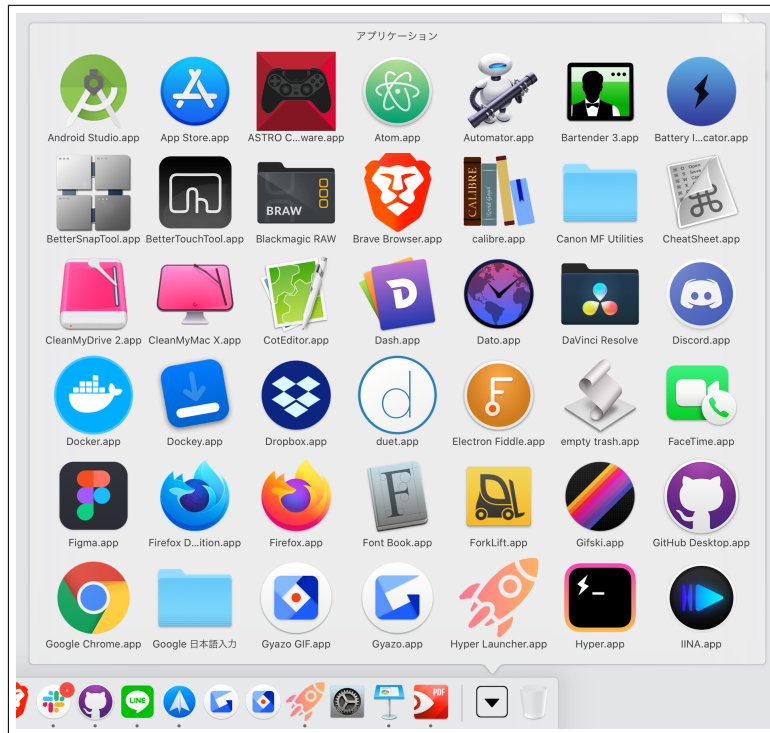


図 1.2: Dock 内のメニュー

(3) 検索型

アプリケーションの名前を入力することで対象のアプリケーションを検索し、起動する形式のランチャー。macOS における Spotlight がこれにあたる。検索メニューは使用するときのみ表示されるため使用していない時は場所をとらないことに加え、キーボードのみで操作が完結するようになっている。もちろんマウスと組み合わせて使用することもできる。また大抵の場合インクリメンタルサーチが導入されているため、アプリケーションの名前を完全に覚えていなくても起動することができる。しかし、汎用性が高い反面、毎回一定のキーボード入力が必要となってしまうため、素早く複数のアプリケーションを切り替えるような操作には不向きである。

(4) ホットキー型

単一のキーもしくは複数のキーの組み合わせを入力するだけで、登録したアプリケーションを起動できる形式のランチャー。他の形式のものとは違い、オペレーティングシステムに標準で搭載されていないことが多く、サードパーティ製ソフトウェアを自分で探す必要がある。普段は画面上に情報を表示する必要がないため場所をとらないが、その分どのホットキーにどのアプリケーションを登録したのか覚えておかなければならない。また、その設定画面の UI も使いやすいとは言えず、未習熟者には使いにくい形式だとされているのも事実である。一発で特定のアプリケー

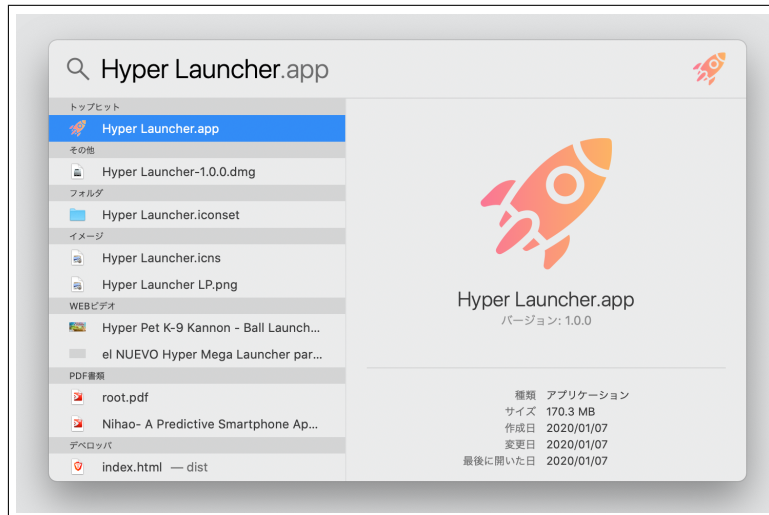


図 1.3: Spotlight

ションを起動できるというのはとても強力な機能ではあるが、あまり普及していないのはこれらの問題が原因だと考えられる。

1.1.2 既存のアプリケーション

先述した通り、ホットキーを利用したアプリケーションランチャーは標準で搭載されておらず、ユーザー自ら使いやすいものを見つける必要がある。しかしその種類は豊富とは言えず、突出した特徴があるわけでもない。例として筆者が日常的に使用してきた2つのアプリケーションランチャーを示す。

(1) PMenu

PMenu²(図 1.4) はホットキーとメニューを利用したランチャーで、アプリケーションだけでなく様々なファイルをコマンドで操作することができる。使い勝手はいたって普通で可もなく不可もなく言ったところである。自分の好きなホットキーを割り当てられるのは便利であるが、キーを登録するためのステップが多くとても煩雑で、こまめに切り替えるような使い方には向いていない。コンテキストメニューを利用することが主な方法とされているため、ホットキーに関しては最低限の機能が実装されているだけと言える。

(2) Snap

Snap³(図 1.5) はホットキーを利用したランチャーで、macOS の Dock を生かした機能が備わっている。Snap では Dock に登録されたアプリケーションに左から順に数字を割り当てていき、そ

²<https://danadesign.ltd/>

³<https://apps.apple.com/jp/app/snap/id418073146>

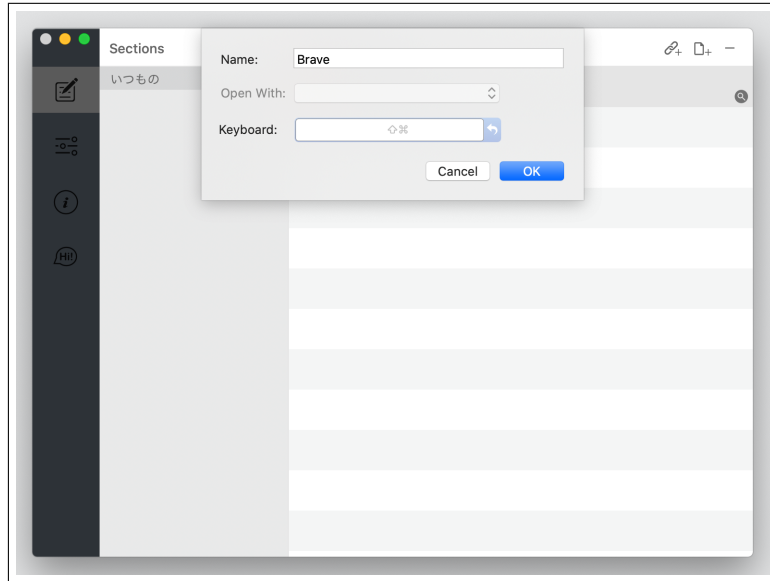


図 1.4: PMenu

の数字と設定したモディファイアキーとの組み合わせでアプリケーションを起動できるようになっている。これにより面倒な設定をすることなくホットキーの恩恵を授かることが可能となる。ただし Dock のサイズは有限であり画面上に場所もとるため、多くのアプリケーションを登録するというのはあまり現実的ではない。ホットキーの設定も可能ではあるが、PMenu と同様操作ステップが多く使いやすいという印象は受けない。

1.1.3 運用例

「macOS でディスプレイ 1 枚で作業する技術」⁴では、実際にホットキーを運用している例が紹介されている。本記事では、macOS の仮想デスクトップを 10 個に増やし、それぞれにアプリケーションとホットキーを割り当てることで、ランチャーと同等の機能を実現している。記事内では、ホットキーを覚えるのは大変だが、特定のキーを押すことで必ず指定されたアプリケーションが起動されるというのは安心感が得られると評価されている。また、デスクトップ上でアプリケーションを探す手間も解消されている。もちろん、他のホットキー型アプリケーションランチャーと同様の問題点はあるものの、その便利さがよく表れている例である。

1.1.4 問題点

ホットキー型のアプリケーションランチャーには、特定のアプリケーションをキーの入力のみで素早く起動できるという利点があるが、設定が煩雑であったり設定を覚える必要があったりするなど、使いこなせるようになるまでに大きな障壁がある。また、その機能についても新しい手法が開発されることは少なく、機能及び利便性の両面でまだまだ改善の余地があると考えられる。

⁴<https://qiita.com/saboyutaka/items/d6cfd2a2b60f1a374d60>

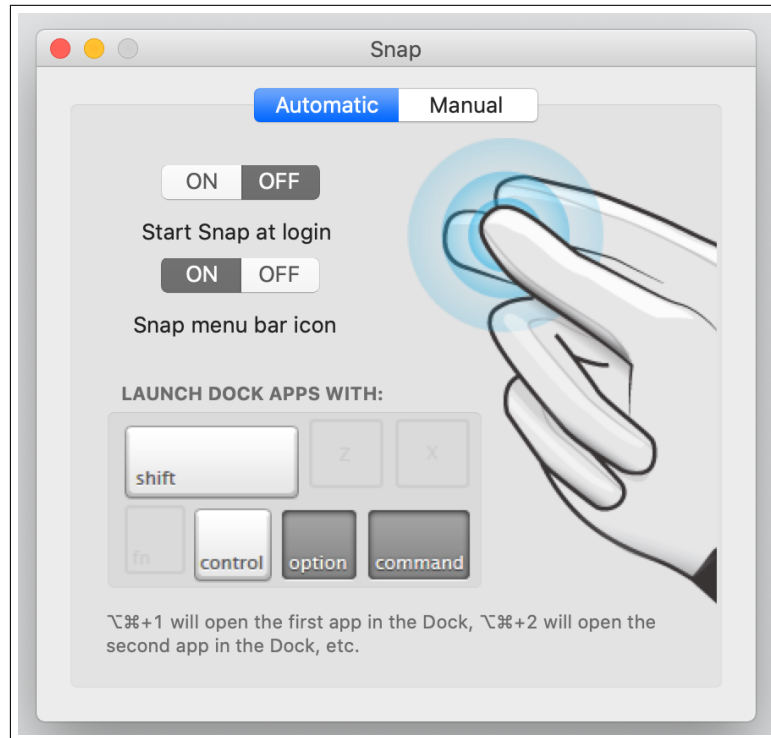


図 1.5: Snap

1.2 本研究の目的

既存のホットキー型アプリケーションランチャーにおける不便を解消し、より強力でより多くの人に使いやすいシステムを開発することが本研究の目的である。

1.3 本論文の構成

第1章では、本研究における背景と問題意識、目的について述べた。

第2章では、第1章で述べた問題意識を踏まえ、新しいホットキー型アプリケーションランチャーシステムを提案する。

第3章ではシステムの実装に関して述べ、第4章では実際に運用して得られた評価について述べる。

第5章では関連研究を交えシステムの議論を行い、第6で本研究を総括する。

第2章 システムの提案

本章では、アプリケーションランチャーについての背景を踏まえ、ホットキーを利用した新しいランチャーシステム「Hyper Launcher」を提案する。

2.1 設計

既存のホットキー型アプリケーションランチャーに見られる不便を解消するため、

1. アプリケーションの登録及びその変更が容易である
2. 設定を覚えるのが容易である
3. より強力な操作が可能である

という3つの目標を実現するランチャーシステム「Hyper Launcher」を設計した。具体的には

1. アプリケーションの登録や変更をドラッグアンドドロップによって簡単に行えるようにする
2. 使用するホットキーの組み合わせを9つに制限し、そこに対してアプリケーションを登録することで、可能な限り認知不可を下げられるようにする
3. 単一のホットキーの組み合わせに対して複数のアプリケーションを登録できるようにする

というアプリケーションを実装した。

2.2 基本操作

自身の開発環境を考慮し、アプリケーションは macOS 上でデスクトップアプリとして利用できることとする。起動画面を図 2.1 に示す。画面は9つのセクションに分けられており、予めそれぞれにホットキーが割り当てられている。

2.2.1 登録と起動

任意のアプリケーションをそれぞれのセクションにドラッグアンドドロップすることで簡単に登録することができる。また、タイトル横のプラスボタンを押すことでアプリケーション選択画面が出現し、そこからも登録することができる。後は指定されたホットキーを入力するだけでアプリケーションを起動することができる。(図 2.2, 2.3)

2.2.2 登録の変更と解除

アプリケーションに紐付けるホットキーを変更したい時は、対象のアプリケーションを目的のセクションへドラッグアンドドロップするだけで操作が完了する。またアプリケーションにマウスカーソルを乗せた際に右側に表れる削除ボタンを押すことで、登録を解除することができる。(図 2.4, 2.5)

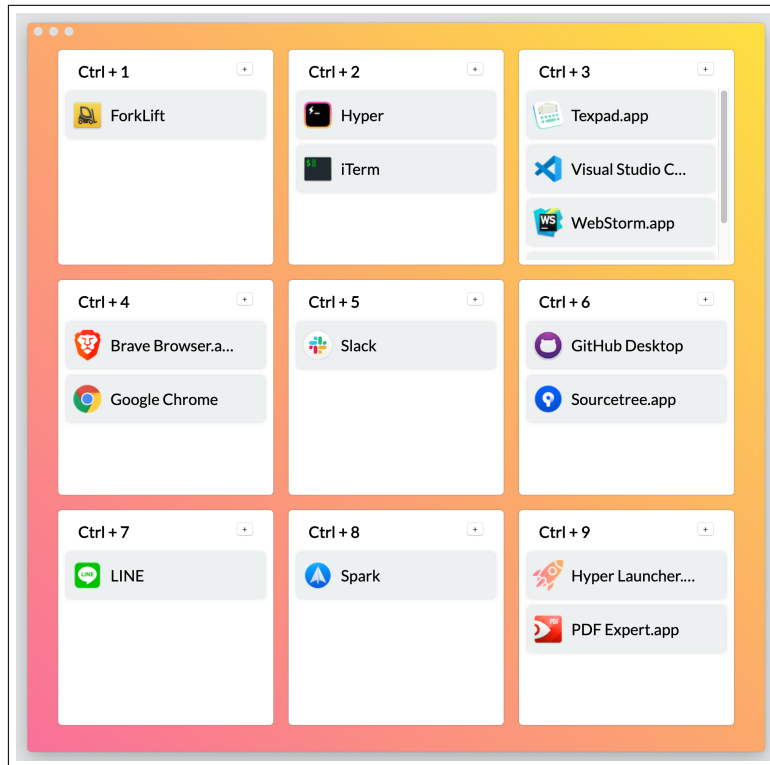


図 2.1: Hyper Launcher の起動画面

2.2.3 複数登録と操作

既にアプリケーションが登録されているセクションに、同じように新しいアプリケーションを追加するだけで複数のアプリケーションを登録することができる。アプリケーションは表示されている順番によって優先順位が定められており、キーを入力する度にその優先順位に基づき順番に起動するようになっている。その優先順位についてもドラッグアンドドロップするだけで変更することができる。

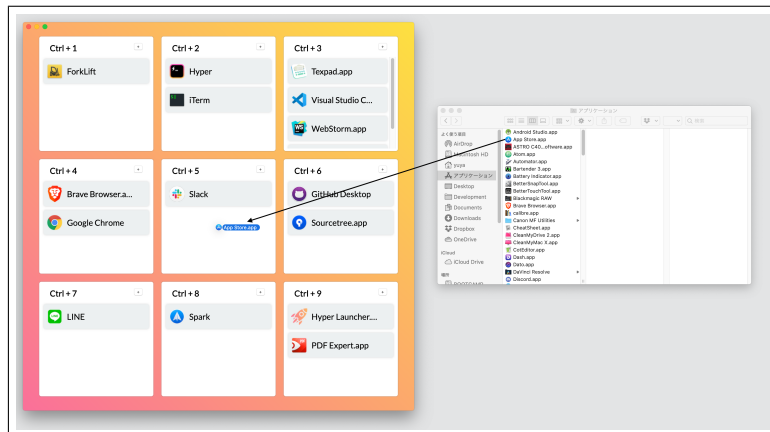


図 2.2: ドラッグアンドドロップによる登録

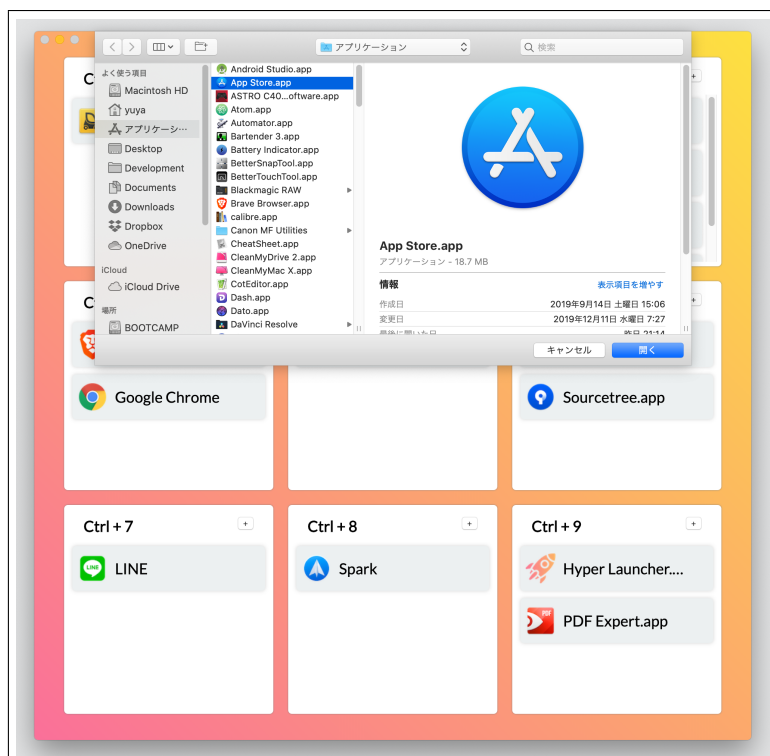


図 2.3: 追加ボタンから登録

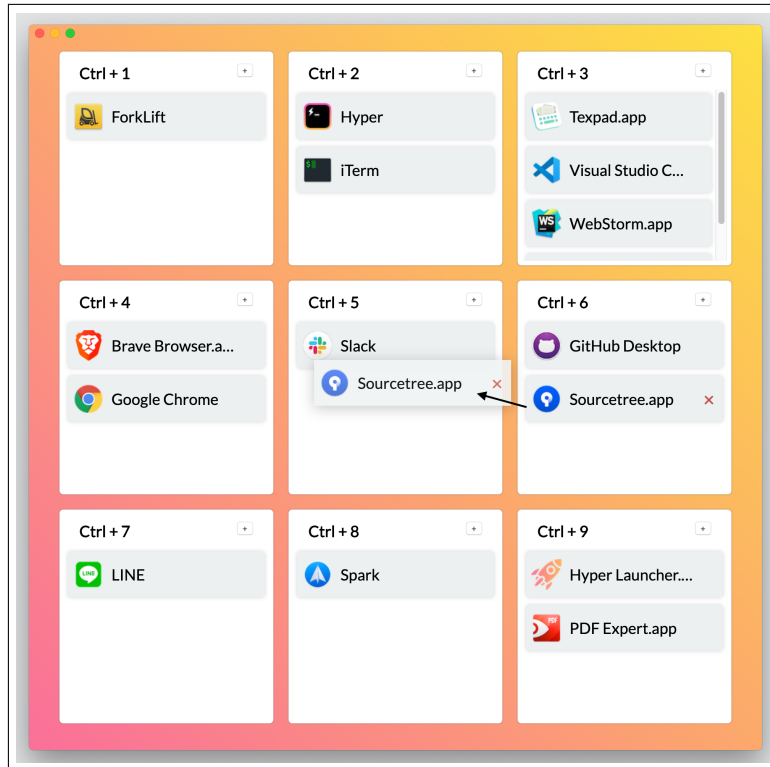


図 2.4: 登録の変更

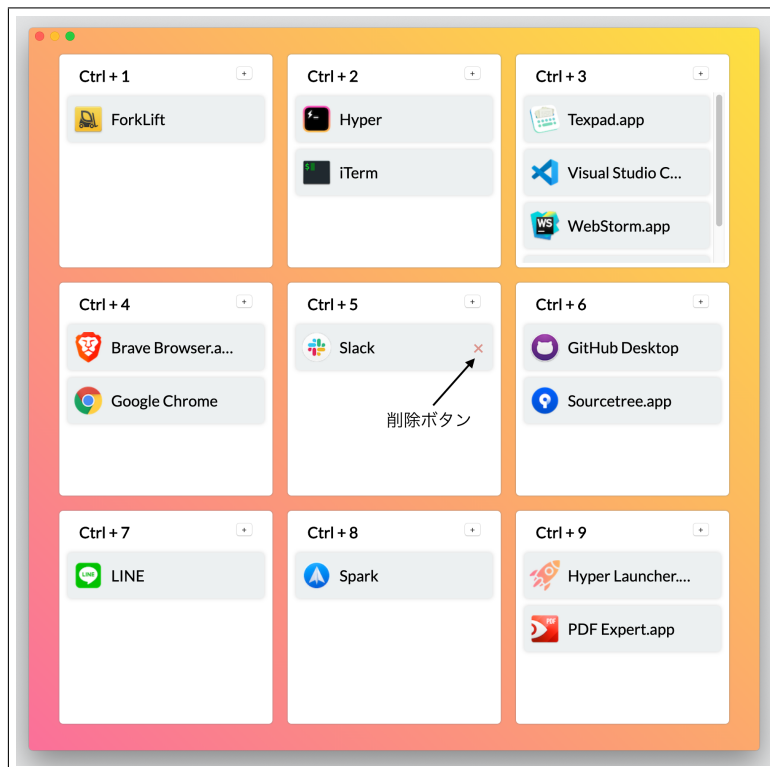


図 2.5: 登録の解除

第3章 実装

本章では、第2章で述べたシステムの設計を受け、Hyper Launcherの実装について述べる。

3.1 システム構成

Hyper Launcher はユーザーがアプリケーションを登録するためのクライアントと、登録したデータを保存しておくためのデータストアから構成される。構成図を図 3.1 に示す。

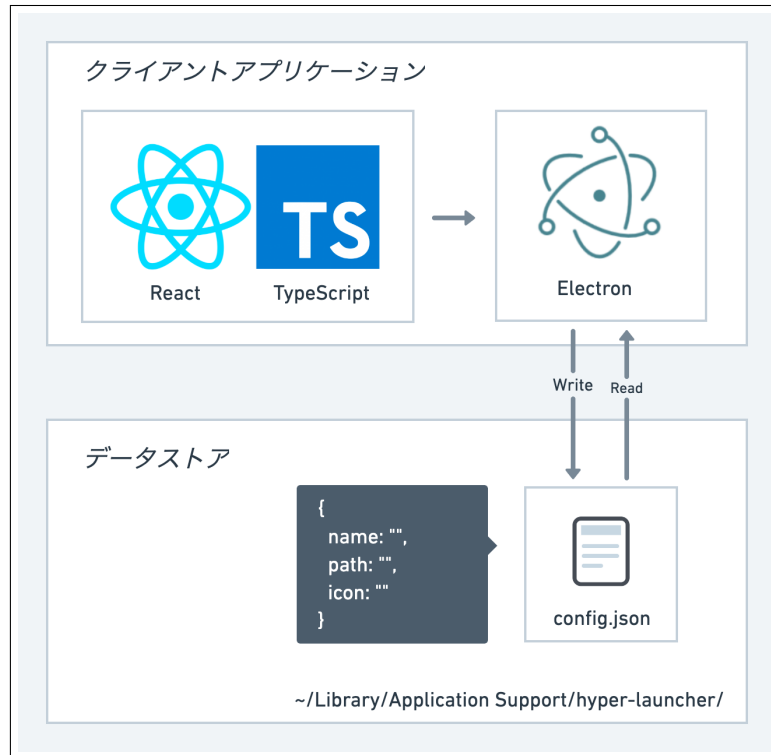


図 3.1: システムの構成

3.2 クライアント

クライアントは Electron¹や TypeScript²、React³といった Web 技術によって実装されており、macOS のデスクトップアプリケーションとして動作する。

(1) Electron

Electron は HTML、JavaScript、CSS を用いてクロスプラットフォームなデスクトップアプリケーションを作成することができるソフトウェアフレームワークである。Electron 自体は Chromium と Node.js を使用しており Web 技術のみで開発が完結するのが利点である。Hyper Launcher では Electron の `globalShortcut` という API を利用し、Hyper Launcher が非アクティブ状態でもホッ

¹<https://electronjs.org>

²<https://www.typescriptlang.org/>

³<https://reactjs.org>

トキーの入力を取得できるようになっている。アプリケーションの起動やアクティブ化なども標準の機能だけで実装できるため、ランチャーに適したフレームワークと言える。

(2) AppleScript

Web 技術のみでは実装が難しい部分については AppleScript を使用した。masOS には osascript と呼ばれるシェルスクリプトが存在し、これを利用することで Node.js⁴から AppleScript を実行することができる。AppleScript によって、起動中のアプリケーションや特定のアプリケーションがアクティブかどうかといった情報を取得することができ、これによってデスクトップアプリケーションとして十分な機能を実装することが可能となった。ただし実行する毎にプロセスを立ち上げるため、無闇に使用してしまうとラグが発生してしまう。また、AppleScript は macOS のみで使えるものであるため、Electron の利点であるクロスプラットフォーム対応はできなくなってしまった。

3.3 データストア

ユーザーが登録した情報は、Electron が定義するユーザーデータ領域 (/Library/Application Support/hyper-launcher/) に JSON ファイルとして保存されている。データのやり取りが全てローカルで完結するため、オフラインでも使用できるようになっている。データはホットキーの数字をキーにしアプリケーション情報の配列を値として持ったオブジェクトになっており、アプリケーション情報にはアプリケーションの名前、パス、そして base64 エンコードされたアイコンの 3 つが含まれている。これにより単一のキーに対して複数のアプリケーションを登録することが可能となっている。以下にその例を示す。ただし icon の base64 文字列と中間部分は長くなるため省略した。

ソースコード 3.1: config.json

```
1 {
2   "shortcut": {
3     "1": [
4       {
5         "name": "ForkLift",
6         "path": "/Applications/ForkLift.app",
7         "icon": "*****"
8       }
9     ],
10    "2": [
11      {
12        "name": "Hyper",
13        "path": "/Applications/Hyper.app",
14        "icon": "*****"
15      }
16    ],
17    ...
18    "9": [
19      {
20        "name": "Hyper_Launcher",
```

⁴<https://nodejs.org>

```
23     "path": "/Applications/Hyper_Launcher.app",
24     "icon": "*****"
25   }
26 ],
27 }
28 }
```

第4章 実運用及び評価

本章では、Hyper Launcher を実際に運用した結果および評価について述べる。

4.1 筆者の運用

筆者は本システムを開発段階を含め半年以上利用した。日常的にホットキー型のランチャーを使用しているため覚えやすさについての評価は出来ないが、その他については大きな利点を実感することができた。特に登録の作業は格段に便利になった。今まではアプリケーションを選択した後に割り当てるキーを選択する必要があったが、そのフローが一つにまとめられたことにより、余計なことを考える必要がなくなった。また一つのホットキーに対して複数のアプリケーションを登録できるため、これまで以上に柔軟な操作が可能となった。1から9のキーに対してカテゴリごとに分けてアプリケーションを管理し、状況に応じてそれらを使い分けたり、よく使うアプリケーションの組み合わせを一つのホットキーにまとめたりと、活用の幅がこれまで以上に広がった。ただし、複数登録したアプリケーションの操作についてはまだ最適とはいえず、まとめて表示したり、起動していない物も対象に含めたりできるようにすると、より便利に操作できるかもしれない。

4.2 第三者の意見

研究室のメンバーのうち、ホットキー型のランチャーをあまり使ったことのない2名に実際に利用してもらい、フィードバックをもらった。

4.2.1 登録について

アプリケーションの登録にドラッグアンドドロップを採用した点については好意的な意見が得られた。他のランチャーでは面倒な作業でも、Hyper Launcherでは楽しく設定できるようになっていると評価できた。また、事前にホットキーが指定されているということも、設定の敷居を下げている要因の一つになっている。ランチャーに限らずホットキーを登録するインターフェースは煩雑なものが多いため、これは大きなアドバンテージだと考えられる。

4.2.2 覚えやすさについて

設定自体は簡単になっていたが、覚えやすさという点ではまだまだ難しいという意見が得られた。事前にホットキーが固定されてはいるものの、そのホットキーとアプリケーションの対応を覚えなければならないことには変わらず、ホットキーの使用に慣れていない人にとっては依然として敷居が高いということがわかった。

4.2.3 ホットキーの有用性について

今回はホットキー型のランチャーをあまり使ったことのない人に試してもらうことができたが、しばらく使っていく中でその便利さを実感してもらうことができた。特にホットキーの入力一発

で素早く起動できるという点は好評だった。しかし今後も使用を続けるかとなるとまだ障壁があり、未習熟者でも段階的に使いやすくなっていくような仕組みが必要だと考えられる。

4.2.4 問題点

その他、細かい問題点についても意見が得られた。以下にそれを示す。

- 複数登録している場合の操作がわかりづらい。直感的でない。
- 削除時の確認はいらぬ。不可逆の変化ではないので簡単に消せたほうが良い。
- ドラッグアンドドロップの移動にバグがある。
- MacBook の場合小指で入力する必要があるため、Control キーの入力が難しい。

これ以外にも痒い所に手が届かないという指摘はいくつかあり、そういった部分については個別にカスタマイズできるような仕組みが必要である。

第5章 議論

本章では、Hyper Launcher についての議論を行う。

5.1 関連研究

Hyper Launcher を設計するにあたり、参考になった文献についてまとめる。

(1) FALCON

FALCON[4] はスマートフォン向けのアプリケーションランチャーで、ユーザーが次に使用する確率の高いアプリケーションをユーザーのいる場所や普段の傾向といったコンテキストから予測し、推薦してくれるというシステムである。予測をしてアプリケーションを並び替えて推薦するという手法は、スマートフォンのようにアプリケーションが格子状に並んでいる場合には効果的であるといえる。ホットキーと組み合わせると、一つのキーのみで全てのアプリケーションが操作できるようなインターフェースができるかもしれないと考えられる。しかし、完璧な予測モデルを作成するのは困難であり、今回は一つのキーに複数のアプリケーションを登録できるようにすることで、その機能に近づけられるよう工夫した。

(2) HotStrokes

HotStrokes[1] はホットキーの有用性を認めつつより習得しやすく且つ GUI メニューよりも効果的な手法として、トラックパッドを利用したジェスチャによるコマンド入力を提案している。確かに視覚的に記憶しやすいジェスチャをコマンドに対応させるという試みは興味深いだが、トラックパッドが使える環境は限られており、ホットキーに比べ汎用性に劣ると考えられる。インタフェースの大きな変革がない限り、ホットキーの習得をサポートするような手法を取り入れるほうが効果的だと考える。

(3) IconHK

IconHK[2] はツールバー上のボタンアイコンにホットキーに関する情報を埋め込むことで、ユーザーの覚えやすさを補助しようというシステムである。ホットキー型のアプリケーションランチャーの場合、画面に表示するものは基本的に何もないため、直接的に取り入れることはできなかったが、これを機にユーザーに寄り添うための機能が必要だと考え始めた。IconHK はその使いやすさとアプリケーションとしての見た目を損なわずに済むという利点を兼ね備えており、これはデスクトップ上に情報を埋め込むなどの方法で応用できるかもしれない。

(4) ExposeHK

ExposeHK[3] はホットキーの使用を助けるシステムで、モディファイアキーを長押しすることで対象アプリケーションに割り当てられているホットキーをツールバー上にプレビューできるようになっている。これにより非習熟者であっても段階的にホットキーを覚えていくことが可能となる。このようなプレビュー機能については実際に使ってもらった中でも欲しいという意見があった。

Electron では長押しの実装ができず断念したが、この機能の重要性は高いと言える。プレビューの表示方法についても様々なバリエーションが考えられ、全てを羅列するだけでなく隣接するホットキーの情報を表示するなど Hyper Launcher に最適な方法を模索したい。

(5) FingerArc, FingerChord

FingerArc 及び FingerChord[5] はホットキーの習熟をサポートするシステムである。カメラを利用して指の位置やその動きを認識し、関連するホットキーを表示してくれるようになっている。これにより非習熟者でも視覚的なガイダンスを通じてホットキーのマッピングを学習できるようになっている。このような習熟をサポートするシステムは数多く提案されているが、カメラやセンサを使った比較的大掛かりな手法はあまり受け入れられないのではないかと考える。まずはホットキーの便利さを実感してもらえよう導入の敷居を下げ、そこから使いこなしていけるようにサポートするようなシステムが必要である。

5.2 展望

(1) ネイティブアプリケーション

今回は筆者のスキルの都合上 Electron を使用した開発になったが、ネイティブアプリケーションと比べると実装に限界を感じる部分があった。今後様々なインターフェースを試していくためには、Swift などを利用したより高度が開発が求められる。

(2) 習熟のサポート

実際に使用してもらった中で一番感じたのは、ソフトウェアが使用者に寄り添っていく必要があるということである。Hyper Launcher ではあまり革新的な手法を提案できなかったが、関連研究はたくさんあり、それらを組み合わせて様々な機能を試していきたい。

(3) 熟練者向けの機能

Hyper Launcher で実装した機能のうち、その操作性を向上させるものについては一定の効果が得られた。これらは熟練者にも有用なものであり、こういった機能もより拡充させていきたい。特に慣れていけばいくほど細かい好みが出てくるものなので、モディファイアキーの変更や複数アプリケーションの操作方法、新たなコマンドの登録など、日常的に使用している人たちにも、さらなる利便性を提供できるようにしていきたい。

第6章 結論

本章では、本研究で得られた成果について述べる。

本研究では、既存のホットキーを利用したアプリケーションランチャーの問題を明らかにし、新しいシステム Hyper Launcher を設計/開発した。その中で、新たな機能を取り入れ実際に運用することで、様々な利点/欠点を得ることができた。

特に

- ドラッグアンドドロップを利用した簡単な設定
- 単一のホットキーに対して複数のアプリケーションを登録できる機能

などについては、既存の不便を解消しより柔軟で強力で操作できるということが明らかになった。

その一方で

- 未習熟者へのホットキー利用をサポートする機能
- ホットキーの学習を促す機能

などについてはまだまだ改善の余地があるということがわかった。これらの機能はホットキーの障壁を無くす上でとても重要なものであり、Hyper Launcher の新しい強力な操作性と組み合わせることで、さらなる利便性の拡充を目指すことができると考えている。

Hyper Launcher のようにデザイン性にも富んだランチャーは少ないため、これを機にホットキー型のアプリケーションランチャーを使ってもらえるようこれからも開発を続けていきたい。

謝辞

本研究を進めるにあたり、学部2年次よりご指導いただきました増井俊之先生に深く感謝いたします。日頃の研究会を通して厳しく的確な意見をいただき、大変勉強になりました。

同じく、論文執筆に関して多大なアドバイスをいただいた田中優氏、実際に使用して感想をくださった早川匠氏と佐々木雄司氏、普段から相談に乗っていただいた左治木隆成氏、及び研究会メンバー、OB 諸氏のみなさんに感謝の意を表します。ありがとうございました。

最後に、筆者がホットキーとランチャーを使うきっかけとなった「macOS でディスプレイ 1 枚で作業する技術」の執筆者である Yutaka Tachibana 氏にもこの場を借りて感謝申し上げます。

参考文献

- [1] Wenzhe Cui, Jingjie Zheng, Blaine Lewis, Daniel Vogel, and Xiaojun Bi. Hotstrokes: Word-gesture shortcuts on a trackpad. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, New York, NY, USA, 2019. Association for Computing Machinery.
- [2] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. Iconhk: Using toolbar button icons to communicate keyboard shortcuts. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, p. 4715–4726, New York, NY, USA, 2017. Association for Computing Machinery.
- [3] Sylvain Malacria, Gilles Bailly, Joel Harrison, Andy Cockburn, and Carl Gutwin. Promoting hotkey use through rehearsal with exposehk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, p. 573–582, New York, NY, USA, 2013. Association for Computing Machinery.
- [4] David Chu Tingxin Yan, Aman Kansal Deepak Ganesan, and Jie Liu. Fast app launching for mobile devices using predictive user context. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, p. 113–126, New York, NY, USA, 2012. Association for Computing Machinery.
- [5] Jingjie Zheng, Blaine Lewis, Jeff Avery, and Daniel Vogel. Fingerarc and fingerchord: Supporting novice to expert transitions with guided finger-aware shortcuts. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, p. 347–363, New York, NY, USA, 2018. Association for Computing Machinery.