卒業論文　2021年度（令和3年度）

ノーコードの先を行く「ノーデザイン」ツールの実現に関する研究

慶應義塾大学 環境情報学部

尾崎 正和

全世界インタフェースデザイン (増井研究会)

2022年1月

卒業論文　2021 年度（令和 3 年度）

ノーコードの先を行く「ノーデザイン」ツールの実現に関する研究

論文要旨

スマートフォンやパーソナルコンピュータの急激な普及につれ，様々な応用ソフト (アプリケーション) が開発され人々が毎日利用している．近年ではアプリケーション開発において十分な開発時間，人員，技術が確保することが難しく質の担保されたアプリケーションを設計し開発することが難しくなっている．そして個人開発者，スタートアップ企業，アイディアはあるが実装する力のない個人などではさらにこれを行うことが難しくなる．

その流れに伴い「ノーコード」と呼ばれるプログラミングやコンピュータサイエンスに精通していない人でも簡単にアプリケーションを開発できるサービスが登場した．しかしながら現在のノーコードにおいてもアプリケーションのユーザインタフェースデザインはユーザが行うか，テンプレートを利用する他ないためユーザインタフェースに精通していないユーザでは自由度の高く質の担保されたユーザインタフェースのアプリケーションを開発するのは困難である．

そこで本研究では，本来大規模な開発でしか行われない専門のユーザインタフェースデザインを自動化し，ユーザインタフェースに関する専門的な知識がなくても容易にユーザインタフェースを開発できるようにするためのシステムの開発をおこなった．ユーザインタフェースの設計の方法については確立された手法はなくデザイナのセンスと経験に委ねられていた部分が多かったが，既存のユーザインタフェースを分析していくことでユーザインタフェースは画面に表示されている要素の数と種類，各要素のグルーピング，画面内での要素がもつ優先度が決定されればインターフェースを制作できるのではないかという仮説を立てた．その仮説に基づき iOS 向けのネイティブアプリケーションのユーザインタフェースを専門的知識なしに制作できるシステムを開発し有用性を明らかにした．

キーワード

UI, ノーデザイン, ノーコード, プロダクト開発, UI 自動生成, インターフェースビルダー, 人間中心設計

慶應義塾大学 環境情報学部

尾崎 正和

Abstract Of Graduation Thesis Academic Year 2021

Research on the realization of "no-design" tools that go beyond no-codes

Summary

With the rapid spread of smart phones and personal computers, a variety of application software have been developed and are used by people every day. In recent years, it has become difficult to secure sufficient development time, manpower, and technology to design and develop applications with guaranteed quality. This is even more difficult for individual developers, start-up companies, and individuals who have ideas but lack the ability to implement them.

In response to this trend, a service called "no-code" has emerged, which allows people who are not familiar with programming or computer science to easily develop applications. However, even with the current "no-code" service, the user interface design of the application can only be done by the user or by using a template, making it difficult for users who are not familiar with user interfaces to develop applications with a high degree of freedom and guaranteed quality. Therefore, it is difficult for users who are not familiar with user interfaces to develop applications with high flexibility and quality.

Therefore, in this research, we developed a system to automate specialized user interface design, which is normally done only in large-scale development, so that users without specialized knowledge of user interfaces can easily develop user interfaces. There is no established method for designing user interfaces, and much of the work has been left to the designer's sense and experience. However, by analyzing existing user interfaces, we hypothesized that a user interface can be created if the number and type of elements displayed on the screen, the grouping of each element, and the priority of each element within the screen are determined. Based on this hypothesis, we developed a system that can create user interfaces for native applications for iOS without specialized knowledge, and clarified its usefulness.

Keywords

User Interface, No-design, No-code, Product Development, UI Generation, Interface Builder, Human Centered Design

Faculty of Environment and Information Studies

Keio University

Masakaz Ozaki

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景

近年，スマートフォンの急速な普及により数多くのアプリケーションが開発され，人々が毎日利用している，そのアプリケーションの開発には人々が感じている課題の洗い出し，適切な仕様の確定，使いやすいインタフェースの設計，実装など専門的で複雑な工程が多数存在する．中でもインタフェース設計とその実装は開発全体の中で設計段階で 45 %，実装段階で 50 %，保守段階で 37 %の時間を占めると言われている [2].

またデザインカンパニーである Goodpatch の社長である土屋は創業の原点について次のように述べており，デザインの重要性について述べている．

> Goodpatch は 2011 年に UI デザインに特化したデザイン会社としてスタートしました。2011 年に起業前に渡ったサンフランシスコでは創業期の Instagram や Uber、Airbnb など多くのスタートアップの共同創業者にデザイナーがおり、ベータ版から UI デザインに力を入れ、ユーザー体験を最初から考え、デザインを重要な差別化要素としてプロダクト開発を行っておりました。日本の環境と大きなギャップを感じた土屋は日本に帰り、創業したことが Goodpatch の原点です。[3]

大手企業の大規模開発では社内に専門知識を持ったデザイナが在籍していたり，Goodpatch のようなデザインカンパニーと共に設計を行うことが可能であるが専門のデザイナが在籍しないスタートアップ企業やインディーズ開発，個人開発ではこれらを行うのは非常に困難である．

また，日本ではデザインが装飾などの表層的なもののみを指すと誤解されていると指摘されている [3]．本来のデザインの役割とは，本質的な (何の？) 価値を見出し，価値を最大化させることである．Garret の UX 五段階モデル (図 1.1) は，具象と抽象を行き来しながら，戦略，要件，構造，骨格，表層全てを設計することがデザインであると示している．

1.1.1　システム開発におけるデザイナの役割と現状

資金や人的リソースが十分な大規模開発では役割としてのデザイナが参画している場合も多いが，前述のようなデザインの能力を有しているとは限らない．職業としてのデザイナを職業としている人の中でも Garret が述べている本来のデザインを忘れ，表層のデザインに注力してる場合が多く本来のデザインが一般に普及しているとは言えない．(←ソースは？

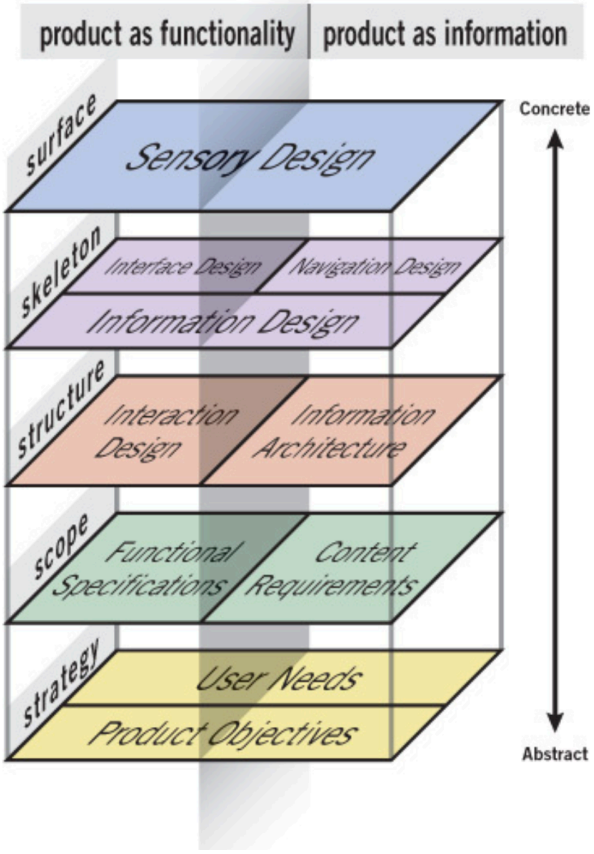

図 1.1: Garret の UX 五段階モデル [1]

また，デザイナよりもアプリに向き合っている時間が長いエンジニアの方がアプリのデザインについて専門的知識を持ち開発をおこなっている場合も多数ある．エンジニアの場合 Garret の UX 五段階モデルもさることながら，UI が実際に動く仕組みを理解している分，内部構造という面での専門知識を有してる場合も多い．実際に日本経済新聞電子版のメンバーによるデザインに関する登壇では 14 個中全ての登壇がエンジニア，又はエンジニア経験のある人の登壇となっている．[4]

個人開発アプリは大規模開発するほど収益が見込まれないものの，ニッチな消費者のニーズにあうさまざまなアプリケーションがアプリストアにリリースされている. しかしながら資金，時間，人的リソースが十分にない個人開発では開発者がデザインの重要性について理解した専門家がいないことにより使い勝手が悪いものや，表層のデザインにのみ注力したアプリケーションがあり，消費者のニーズに答えられずにいる．

以上のことから，(1) 現在は大規模な開発でしか行われていない専門性の高いユーザインタフェースデザインのハードルを下げること，(2) 表層のデザインにとらわれずに奥深くのデザインに注力することができる手法を開発することが求められる．

1.2 目的

本研究では (1) 現在は大規模な開発でしか行われていない専門性の高いユーザインタフェースデザインのハードルを下げ，(2) 表層のデザインにとらわれずに奥深くのデザインに注力することができるようにするシステムの開発を目指す．

現在の UI デザインには確立された設計手法はなく，デザイナの経験やセンスに頼っているものの，既存の UI を分析することにより UI 設計の手法を確立すれば簡易な UI 自動生成システムになり得る．また，(2) 注力されがちであった表層のデザインを自動化することでデザイナ，個人開発者が本来のデザインに向き合うことにも目を自然と向けられるようになることが可能になる．

本研究ではこの手法になりうる仮説として，１画面に必要な要素一覧，各要素の相対的な優先度，要素同士のグルーピングから UI を自動で生成する手法を提案する．

そして提案した手法をもとに１画面に必要な要素一覧，各要素の相対的な優先度，要素同士のグルーピングの情報から iOS ネイティブアプリケーションのソースコード (SwiftUI) を自動で生成するシステムを開発した．そして，開発したシステムの有用性を示した．

1.3 本論文の構成

本論文の構成を示す．

第 1 章では本研究の背景について述べた．第 2 章では関連研究と諸概念を整理する．第**??**章では要素，優先度，グルーピングから UI を自動生成する手法を提案し，第**??**章ではその手法をもとに iOS 向けネイティブアプリケーションのコードを生成するシステムを提案す

る．そして，それらの有効性を示す．最後に，第 5 章の結論では本研究を総括し，考察と展望を述べる．付録として，本研究で行った実験で得られたデータを添付する．

第2章　関連研究と諸概念の整理

2.1　ユーザインタフェースのデザイン手法

2.1.1　大規模開発に置けるユーザインタフェースデザイン

2.1.2　個人開発に置けるユーザインタフェースデザイン

2.2　ノーコードでのシステム開発

大規模な開発を行うリソースがない企業や専門知識をあまり持たない個人でもアプリケーションやウェブサイトを作れるように近年，ノーコードと呼ばれるサービスが登場した．ノーコードとは文字通り，コーディングを行わずにアプリケーションやウェブサイトを作れるサービスである．

全体を通して，コードを書かないのでコードを書けばできることができない．

ここでは既存のノーコードのサービスの例をいくつかあげ，特徴と課題点を挙げていく．

2.2.1　STUDIO

STUDIO は STUDIO 株式会社が運営する

デザインの自由度が高いのがウリ．自由度の高いノーコードってイラレみたいで結局表層のビジュアルデザインに目を向けがち．web front-end の専門知識はなくても行えるが，デザインの専門知識は求められる．自由度の高いビジュアルデザインを売りにしているが，細かなインタラクションなどを追加でソースコードを書くことで補えないので STUDIO のできる範囲内で行わなければならない．素人ではなくある程度知識のある人が使うものであるにもかかわらず，STUDIO で制作した web サイトは一目でわかってしまうため，コードを書けないと自分で宣言しているようなものである．

2.2.2　Glide

Google Spread Sheets などをデータベースとして扱い，テンプレートベースの簡易なアプリを作れる．スケーラビリティが終わっているテンプレートベース社内ツールの自動化や小規模のもの向けでこれで作ったものをプロダクトとして出すことは難しい

2.2.3 Microsoft Power Apps

Glide と同じような感じであるが，若干のスケーラビリティがる．同じくビジネスツールなのでそれ自体をプロダクトにすることは難しい

2.3 デザインの自動化

デザインの専門知識がない人でもテンプレートに頼らず適切の UI を制作できるように UI の自動生成を行う研究が行われている

2.3.1 機械学習を用いた UI の自動生成

機械学習使うと次同じインプットでやった時にも同じのが出てくると限らないので適していない

(1) GAN を用いた自動生成

(2) GPT-3 を用いた自動生成

2.4 問題の所在

- デザインを表層のデザインだけだと誤解している
- 既存のノーコードではデザインは自分で行う又はテンプレートから選んだデザインをそのまま使用する必要がある
- 機械学習を用いた自動生成では一意性が担保できない．
- GAN で生成した UI は画像なのでアプリとして機能しない．

また，スマートフォンの UI は 2007 年の初代 iPhone から，パソコンについては 1984 年の初代 Macintosh から基本的な UI は全く変わっていないにもかかわらず，UI の生成の法則性の研究はあまり行われてこなかった．

以上のことから表層のデザインを自動化し，専門知識のない人でも

第3章　ノーデザイン: 要素，優先度，グルーピングの情報からのUI自動生成手法の提案

本章では，後述の iOS ネイティブアプリケーションのインタフェース自動生成システムの開発に先立ち，既存のインタフェースの分析から導きさした要素，優先度，グルーピングによりインタフェースの構築が行える法則について検討した．自動生成を行うアルゴリズムの提案行う．

3.1　既存インタフェースの分析

既存の iOS ネイティブアプリケーションの UI の分析をおこなっていく．画面内での UI は画面遷移を司り，画面内の体験には直接影響を与えないものが存在する．わかりやすくするために，画面遷移を司る UI とそうでない UI について分類して考えていく．

3.1.1　画面遷移を司る UI

HIG の hierarchical Navigation, FlatNavigation Content-Driven or Experience-Driven Navigation 大体こうなってる．

今回の自動生成手法の提案ではこのような画面遷移を司る UI は切り離し，画面内のコンテンツの UI に注目して考えいく．実際の SwiftUI のコードでも画面遷移を司る UI, 画面のタイトルとコンテンツは別で実装されて画面遷移の上にコンテンツが乗る形式になる．のでこの方針は問題ない

3.1.2　画面の内容を表示する UI

前述では画面内の UI について画面の内容を表示する UI を画面遷移を司る UI が内包していると述べた．本セクションでは画面の内容を表示する UI のレイアウトは主に画面内に表示する要素 (UI パーツ), その要素同士の優先度，そして要素同士のグルーピングの 3 つのルールに則って構成されており，そのルールさえ示せば画面を構成できるのではないかという仮説を立てた．次にレイアウトを構成する 3 つのルールについて述べる．

(1) 要素

画面内にどの要素がいくつ存在するのかを定める必要がある．以下の図では既存の Instagram の画面における要素の抽出をおこなった．

(2) 優先度

画面内の要素の種類と個数がわかったら次に画面内でその要素の重要度を明確にする．例えば，SNS サービスであっても写真をメインとする Instagram と文字をメインとする Twitter では文字と画像の優先度の比重が変わってくる．以下の図では Instagram の画面における要素同士の優先度を抽出したものである．

(3) グルーピング

要素，優先度に加えて各要素同士でグループがある場合，グループを明確にする．以下の図では上述と同様に Instagram の画面における要素同士のグルーピングを抽出したものである．

3.2 アルゴリズムの構築

次にこの要素，要素同士の優先度，要素のグルーピングから 1 画面の UI を構成するアルゴリズムを検討した．

要素を優先度の高い順に画面内に縦に並べていくのが基本．グループは一つの要素としてカウントする．(優先度は中の要素の優先度の足し合わせ) 要素の中にはユーザが操作するもの，見るだけのものがあり，操作するものは全体に並ぶと良い．Group が入れ子になる場合は Hstack, Vstack を交互にすることによって大体の UI を構成できる．画面内の要素の優先度の基準を 10 とし，それを超える場合はスクロールするようにする．これで大体いい感じになる．Padding 等は適切につけていく．優先度が上がる度に文字は大きく，太く，ボタンもただの青文字ではなく，border がついたり，塗り領域が増えたりとしていく．

第4章　iOSネイティブアプリケーションのインタフェース自動生成システムの開発

前述の既存UIの分析と分析から導き出したアルゴリズムをもとにiOSのネイティブアプリケーションのインタフェース自動生成システムの開発を行った．

また，普段からiOSのネイティブアプリケーションを個人開発している14-17歳の男女を被験者とした本システムの有用性をはかる実験を行い，有用性を示すことができた．

本システムの設計は以下のとおりである．

4.1　システムの設計・開発

要素，優先度，グルーピング情報からの画面のコンテンツUI自動生成システムのプロトタイプとして主要な部分のみを実装した．本研究において画面のコンテンツUIとは実際画面に表示されるUIのうち，画面遷移を司どるUI，画面のタイトル表示を除いたものを指す．

本研究のUI自動生成において重要なのは画面内の要素，優先度，グルーピング，の情報のみでUIを自動生成できることである．この点を実現するためのシステムを開発した．

本システムで利用できる要素 (UIパーツ) の名称と機能は以下 (表4.1) とする．名称の命名に関しては基本的にSwiftUIでの名称を使用するが，SwiftUIではテキスト表示の行数による名称の違いがないため独自に一行表示をLabel, 複数行表示をTextViewとした．

本システムでの名称	UIKit での名称	SwiftUI での名称	ユーザの操作の有無	主な使用用途
Label	UILabel	Text	無	一行の文字列を表示するために使用する
TextView	UITextView	Text	場合によっては有	複数行の文字列を表示, 入力, 編集するのに使用する)
TextField	UITextField	TextField	有	一行の文字を入力するために使用する
Button	UIButton	Button	有	タップアクションを取得するために使用する
Toggle	UISwitch	Toggle	有	ON/OFF の切り替えのために使用する
Image	UIImageVIew	Image	無	画像表示のために使用する
Map	MKMapView	Map	有	地図の表示のために使用する
DatePicker	UIDatePicker	DatePicker	有	日付選択のために使用する
Slider	UISlider	Slider	有	特定の数値を調整するために使用する

表 4.1: 利用可能な要素一覧

4.1.1 基本的な配置アルゴリズム

前述のアルゴリズムより，基本的には要素を優先度順に縦に配置する．グループも一要素として扱い表示を行う．グループ内のレイアウトについては後述のグルーピングセクションで行う．

基本的な優先度に加え，UI の要素にはユーザからの操作を受け付けるものと受け付けないものがある．ユーザからの操作を受け付けるものに関しては指の届きやすい画面下に配置するのが適切であり，以下の表 4.2 のように各要素の種類ごとに配置係数を定め，後述の優先度の値と掛け合わせた値の昇順に配置する．

名称	配置係数
Label	1.0
TextView	0.9
TextField	0.8
Button	0.5
Toggle	0.5
Image	2.0
Map	1.5
DatePicker	0.7
Slider	0.7

表 4.2: 利用可能な要素一覧

4.1.2 優先度

本システムにおいて優先度からは要素の視覚的な目立たせ具合，表示順序を決定する．優先度の値は一画面あたりの合計を 10 とし，それを超えた場合はスクロールする画面が提供される．優先度による各要素の目立たせ具合は以下の通りである．以後登場するスクリーンショットは全て本システムによって自動生成された UI である．

(1) Label

図 4.1 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

(2) TextView

図 4.2 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

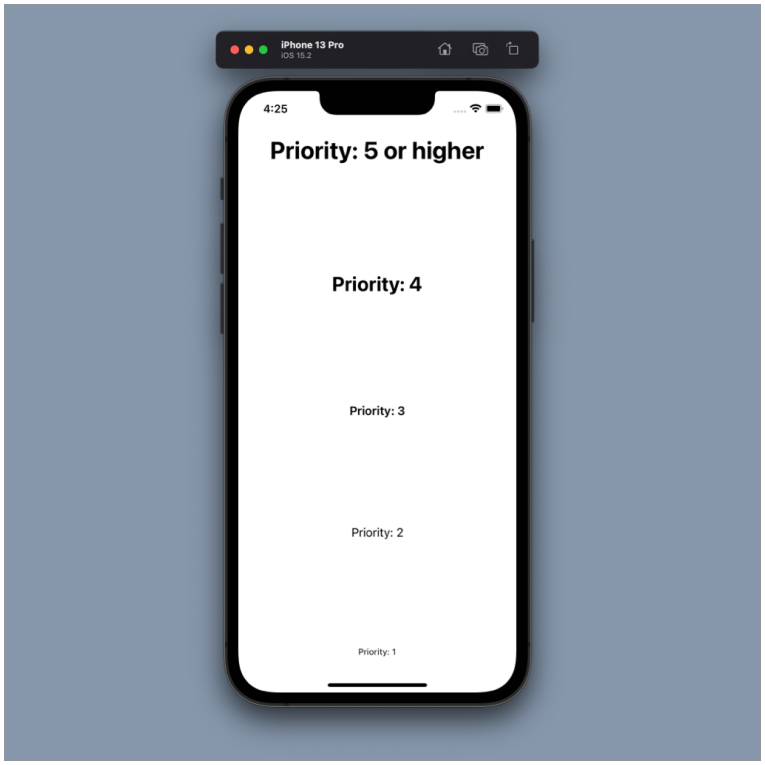

図 4.1: Label の優先度による視覚的目立たせ具合

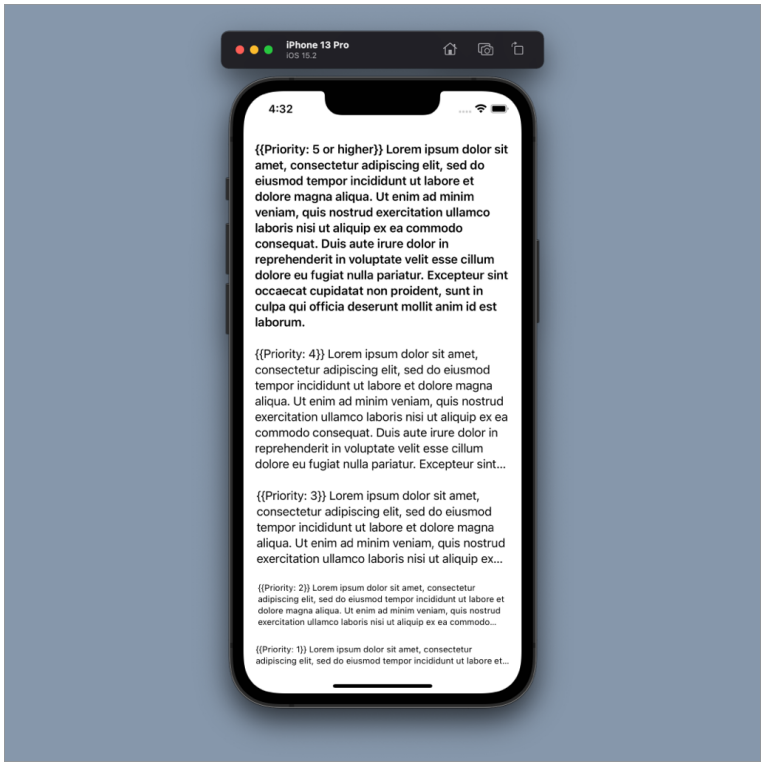

図 4.2: TextView の優先度による視覚的目立たせ具合

(3) Button

図 4.3 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

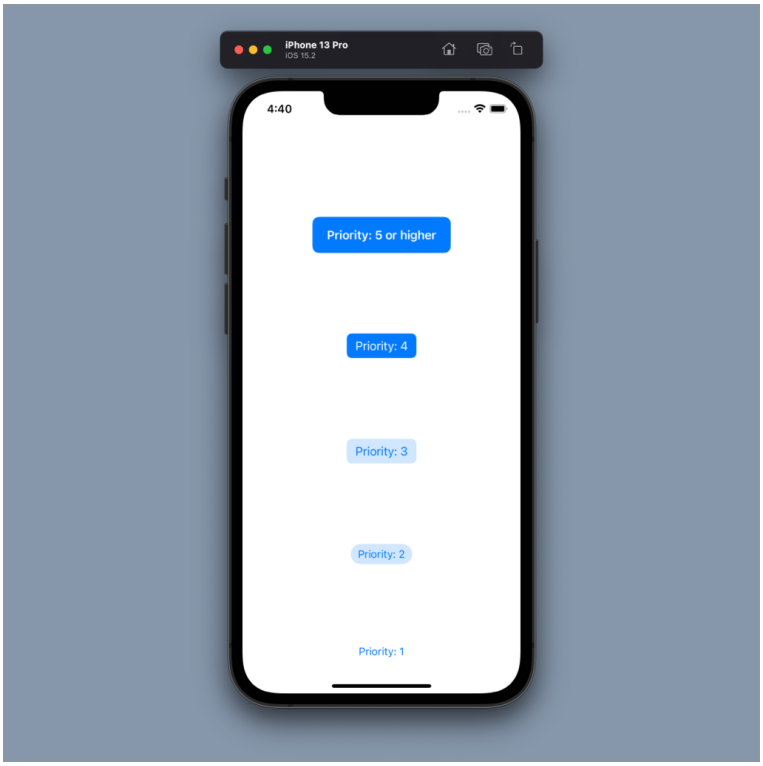

図 4.3: Button の優先度による視覚的目立たせ具合

(4) Toggle

図 4.3 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

(5) Image

図 4.5 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

(6) Map

図 4.6 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

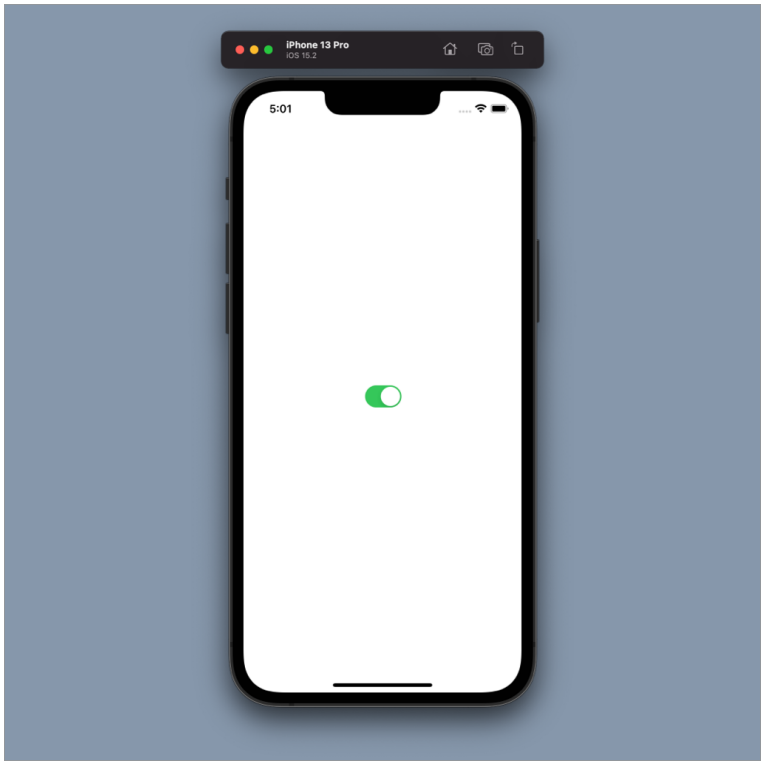

図 4.4: Toggle の優先度による視覚的目立たせ具合

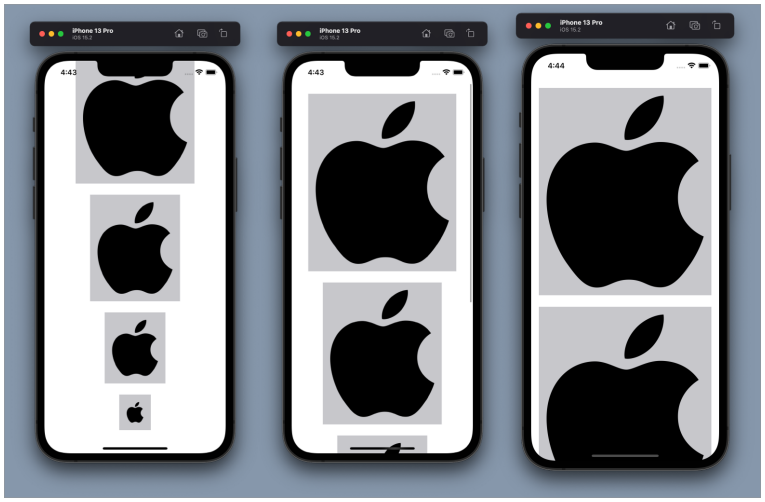

図 4.5: Image の優先度による視覚的目立たせ具合

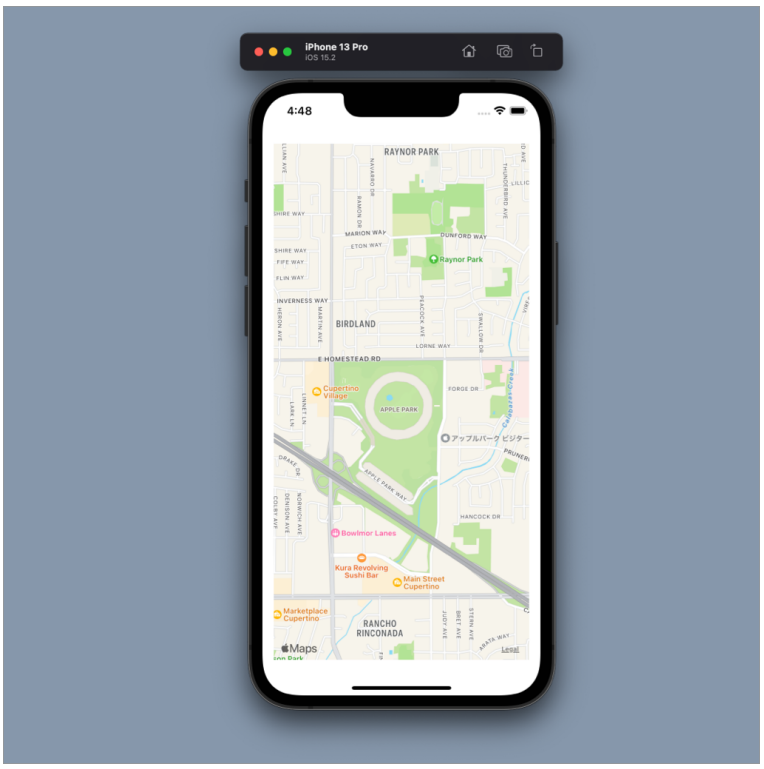

図 4.6: Map の優先度による視覚的目立たせ具合

(7) DatePicker

図 4.7 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

(8) Slider

図 4.8 のように実装した．優先度 5 以上でもこれ以上の大きさになることはなく，余白として扱われる．

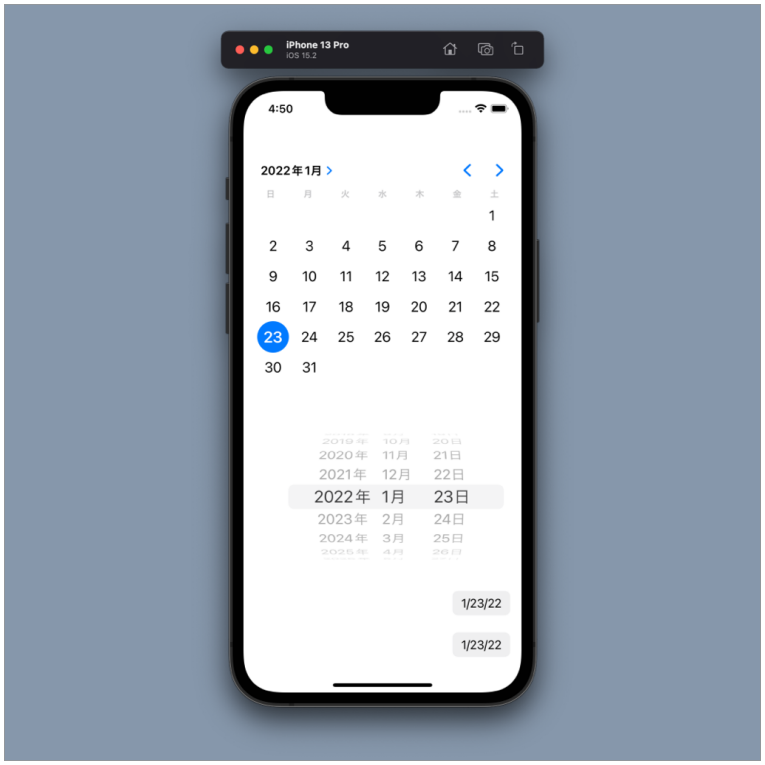

図 4.7: DatePicker の優先度による視覚的目立たせ具合

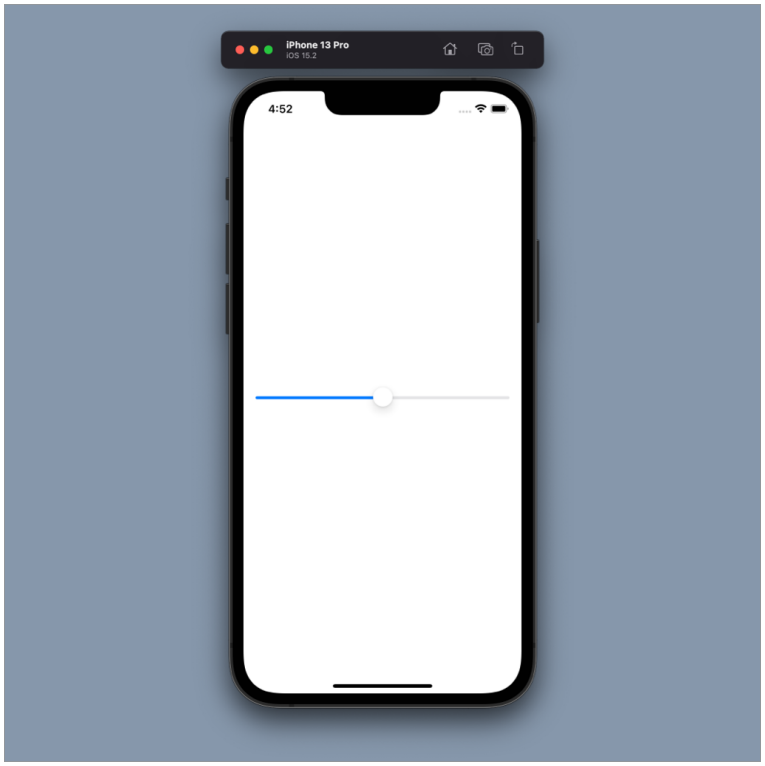

図 4.8: Slider の優先度による視覚的目立たせ具合

4.1.3　グルーピング

(1)　グルーピングの入れ子構造

4.1.4　表示レイアウトアルゴリズム

4.2　既存 UI の再現

4.3　システムの有効性についての実験

4.3.1　対象と手続き

4.3.2　結果と考察

第5章　結論

5.1　システムの提案

5.2　今後の課題

5.2.1　配色の自動生成

5.2.2　スライダー UI による表層デザインの自由化

5.2.3　統合的なシステムとしての展望

謝辞

本論文の作成にあたり，研究室聴講時代から終始適切な助言を賜り，また丁寧に指導してくださった増井俊之教授に深く感謝申し上げます．増井俊之教授には本研究のみならず，日頃からプロダクト，アイディアに対しても NOTA CTO としてのプロダクト開発の側面のご意見，研究者としての学術的なご意見を多数賜りました．感謝申し上げます．増井研究会では，博士課程の田中優氏，修士課程 OB の左治木隆成氏には様々なご意見をいただき充実した研究会活動をさせていただきました．卒業生の皆様，研究会メンバーの民様にも多くのご支援をいただきました．

また同輩であり，孫正義育英財団 [5] 正財団生であり，Bridge UI 株式会社 [6] 代表取締役社長兼 CTO の佐々木雄司氏には気の置ける友人として，日々的確で新しい意見をいただいた他，TEX の環境構築にも多大なるご支援をいただきました．

そして 1 年次よりデザインに関する終始適切な助言を賜り，丁寧に指導してくださった鳴川肇准教授に感謝申し上げます．また鳴川研究会メンバーの皆様にも日々たくさんのご支援をいただきました．

公私に渡りご指導，ご支援いただきました皆様に心から感謝申し上げます．

2022 年 1 月 吉日

尾崎 正和

参考文献

[1] Jesse James Garrett. *The elements of user experience : user-centered design for the Web and beyond*. New Riders, Berkeley, CA, 2010.

[2] Brad A Myers and Mary Beth Rosson. Survey on user interface programming. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 195–202, 1992.

[3] Goodpatch の原体験. `https://goodpatch.com/design`.

[4] Nikkei メンバーによる発表資料. `https://hack.nikkei.com/slides/?categories[]=` デザイン.

[5] 孫正義育英財団. 孫正義育英財団. `https://masason-foundation.org`.

[6] Inc BridgeUI. Bridge ui 株式会社. `https://bridgeui.co`.

付 録A　付録