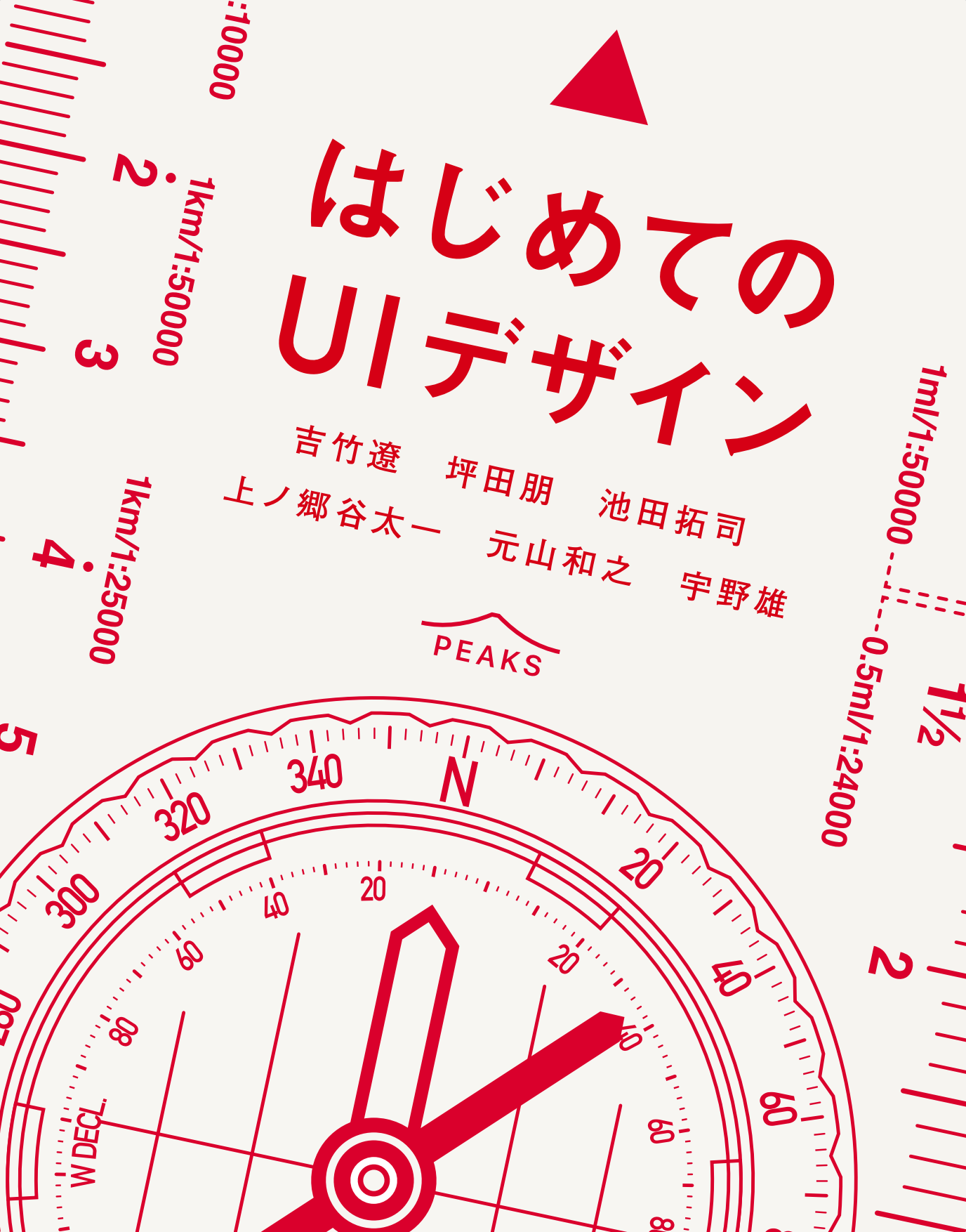
はじめてのUIデザイン

吉竹遼　坪田朋　池田拓司
上ノ郷谷太一　元山和之　宇野雄

PEAKS



1

はじめに

吉竹 遼

あなたはどんな思いでこの本を手に取りましたか？　最初の章ではこの本の簡単なイントロダクションと、UIデザインのこれまでの変遷、読み進めるために事前に知っておいてほしいことをまとめました。

この章のゴール

- この本の概要がわかる
- デジタルプロダクトにおける UIデザインの変遷を知る
- UIデザイナーが関わる人たちについて 理解する

1-1 なぜ今UIデザインなのか

なぜ、今、UI（ユーザーインターフェイス）デザインの本を出そうと思ったのか。きっかけは2つあります。

1つは、外部でUIデザインの講師を務める中で感じた「UIデザインを体系的に学ぶための本が少ない」という課題感でした。特に、スマートフォンアプリケーションなどのデジタルプロダクトをデザインするために必要な基礎知識や考え方がまとまっている本が必要だと感じました。

もう1つは「知り合いで良いUIデザイナーいませんか？」「仕事の依頼はたくさん届くんだけどUIデザイナーの数が足りなくて……」といった周りの声です。（肌感ですが）UIデザイナーになりたい人が増えてきて、必要とされる場面も増えてきたのに、初学者が学ぶ環境がまだまだ足りていない。それをどうにかしたい、と思ったのが執筆のきっかけです。

近年、SketchやAdobe XD、Figmaなど、UIを作るツールはとても手に入りやすくなりました。無料で使えるツールもあるので、環境さえ整えれば今すぐにでもUIを作り始めることができます。ですが、ツールは単にパーツをレイアウトできるだけで、「どのように全体像を考えるのか」「レイアウトするパーツの使いどころをどのように知るのか」は教えてくれません。

ではそういった内容をどこで学べるかというと、現状はほぼ独学に近い状況と言えます。インターフェイスデザインの専門書やAppleやGoogleが用意しているガイドラインなど、たしかに手がかりとなる情報はあります。ですが、昨日今日でUIデザインを学びたい、と思った人に差し出す最初の1歩としては、ちょっとハードルが高いです。一方でデザインツールと組み合わせやすい練習法としてUIトレースなどもありますが、基本理解がまだない状態でいきなりUIの具体化をおこなうのは少し危険でもあります。

この本は「はじめての」と付いているように、UIデザインを学びたい人を対象に、基本知識や考え方がひととおり身につくことを目指して書かれました。第一線で活躍されているデザイナーの方々にお声がけをし、みなさんの経験や考え方を交えつつ、各章を執筆していただきました。ぜひ、プロフェッショナルの生の考えや視点を取り入れてみてください。

最近はUIデザイナーが活躍する領域も拡がっており、特に海外では「プロダクトデザイナー」「デジタルプロダクトデザイナー」と呼ばれることも多くなってきました。UIデザインだけを勉強すれば万全、ということはありませんが、UIデザインに興味を持ってこの本を手にとったあなたが、UIデザインの楽しさを原動力に次なるステップに進むことを楽しみにしています。

1-2 この本で学べること

この本は

・コンポーネントの種類と役割について学ぶ（2章）
・情報設計の考え方と、可視化の方法（3章）
・各プラットフォームの特徴（4章）
・実務におけるUIの設計方法（5章）
・UIの検証（6章）
・UIデザインより前に考えるべきこと（7章）

を学ぶことができるように設計されています。英語の勉強に例えると、

・どのような英単語があるかを学ぶ（2章）
・文法を学ぶ（3章〜4章）
・単語と文法を組み合わせて文章の作り方を学ぶ（5章）
・文章がうまく伝わっているかを学ぶ（6章）
・「なぜ英語を勉強するのか」「勉強する前に準備することは何か」を考える（7章）

といった構造になっています。

読み終える頃には、UIを俯瞰して設計できる基礎知識や、UIデザインを始める前に必要な考え方や視点が身につくはずです。ですが、この本を読み終えたからといってあなたの学びがそこで終わるわけではありません。あくまでこの本は地図やコンパスのような、次の道を指し示す最初のアイテムと捉えて読んでください（そういえばこの本の表紙はコンパスですね！）。

この本のターゲット

先ほども書いたように、この本のターゲットは「UIデザインについて学びたいと考えている人」です。

もう少し具体化をすると、UIデザイナーを目指している人や、UIデザイナーとして入社したけれど知識や考え方に不安が残る人などを想定して書かれています。

この本で学べないこと

反対に、この本を読んでも学べないこともたくさんあります。ざっと挙げてみましょう。

・プログラミング、エンジニアリングを考慮した考え方
・デザインツールの細かい使い方
・グラフィックデザインの習得
・カッコいいUIの作り方
・認知心理学
・人間中心設計（HCD）
・アクセシビリティ
・リサーチ、インタビュー手法
・業態に合わせた適切な答え

上記は取り扱っていないか、扱っているとしても少し触れている程度となります。ただ、この本ではなるべく「他の本を参照したほうが学びが深くなる部分については積極的にリンクしよう」という考え方で書かれていますので、読み終えてもの足りないところが出てきたら参考図書にチャレンジしてみてください。

この本で扱うUIデザインの範囲

UIデザインとひとことで言ってもその対象は多岐に渡ります。この本では主に**デジタルプロダクト（スマートフォンアプリ／Webサービス）**のUIに焦点をあてています。

この本で扱わないUIデザインの範囲

デジタルプロダクトにも種類があります。すべてをカバーするのはページ数的にも難しいため、この本では次のカテゴリーは扱わないようにしています。

・デスクトップアプリ
・タブレット（の詳細）
・テレビ
・ウェアラブル
・音声
・AR
・VR

1-3 デジタルプロダクトにおけるUIデザインの変遷

今、私たちが触れている UI デザインは、ある日突然現れたわけではありません。過去の歴史があり、多くの人々が試行錯誤してきた積み重ねが今に至っています。デザインに関わらず、歴史を知ることは大切です。この節では、iPhone 登場以降の UI デザインの変遷を簡単に振り返ってみたいと思います。

今回は触れませんが、さらに踏み込んだ歴史を学びたい方は次の記事を参考としてください。

2. GUI と CLI の歴史 - sfc-id2017
https://scrapbox.io/sfc-id2017/2._GUI と CLI の歴史

History of the graphical user interface – Wikipedia
https://en.wikipedia.org/wiki/History_of_the_graphical_user_interface

A History of the GUI ¦ Ars Technica
https://arstechnica.com/features/2005/05/gui/

2007年 iPhoneの登場とスキューモーフィックデザイン

スマートフォンの歴史は、2007 年に Apple が発表した初代 iPhone (iPhone 2G) から始まりました。

図 1-01 初代 iPhone (iPhone 2G)

それまでの携帯端末といえば、通話とメールが主な機能のフィーチャーフォンか、ビジネスユースに使われるPDAが主流でした。iPhoneが決定的に違ったのは、なめらかな操作感のタッチスクリーンと、サードパーティが参入できるApp Storeの存在でした（App Storeの開始は2008年）。タッチスクリーンはユーザーが画面上のオブジェクトを指で直接操作するという体験を、App Storeはさまざまなサービスをユーザーに提供できる販路を可能としました。

特にアプリケーションを指で直に触って操作する行為は、それまでの「マウスやキーボードを使った操作」とは一線を画すものでした。ユーザーの操作を助けるためにiPhone OS（現iOS）が用いた手段が、純正アプリケーションに代表される「物体らしさ」です。本物がそのまま画面に入っているかのようなコンパス、紙の質感が再現されているメモ、実際のカジノテーブルのような素材で構成されたGame Center、紙がめくれるような動きのページ送りなど、リアリスティックなUI表現はやがて「スキューモーフィック（現実の材質に似せた）デザイン」と呼ばれるようになりました。

図1-02 Game Center／コンパス／計算機

一方で、スキューモーフィックデザインには表現やレイアウトに物理的な制約が出てしまう側面もありました。Appleも開発者向けのドキュメント「Human Interface Guidelines（HIG）」（ヒューマンインターフェイスガイドライン）のiOS版で次のように言及しています。

たとえば、電話をかけるアプリケーションを例に考えてみます。アプリケーションが、キーパッドの代わりに、美しくリアルな回転式ダイヤルを表示していると想像してください。このダイヤルは細部に至るまで描かれていて、ユーザはその品質を評価すると同時に、ダイヤルの使い方をすぐに理解します。このダイヤルは実際のダイヤルと同じように動作します。ユーザは、ダイヤルを回すジェスチャと特徴

的な音に喜びます。しかし、「連絡先(Contacts)」に登録されていない電話番号にかけることが多いユーザにとって、この体験の最初の評価はまもなくフラストレーションに変わります。なぜなら、回転式ダイヤルはキーパッドを使用するよりもはるかに効率が悪いためです。ユーザが電話をかけやすいよう設計されているアプリケーションでは、こうした美しいカスタムUIが障害となります。

(iOS Human Interface Guidelines 2012-12-17版より)

後年になると端末の多解像度化が進み、グラフィック表現の多彩なアセットを対応させにくい、といった事態も起こるようになりました。

最終的には、Appleが2013年に発表したiOS 7でフラットデザインへの移行をおこなうことでその役目を終えることになりました。ですが2014年に登場したMaterial Designなど、他プラットフォームに与えた影響は大きいと言えるでしょう。

2008年 Androidの登場

iPhone登場の1年後にGoogleが発表したのが、Android OSです。

図1-03 最初に発売されたAndroid端末 T-Mobile G1

2

UIの見える部分を学ぶ

池田 拓司

この章では、コンポーネント、ビジュアル要素、アイコンといったUIをデザインするうえで基本となる見える部分について、私のこれまでの経験から重要だと感じているエッセンスを盛り込みながら、取り上げます。この章を読み終えると次の３つが学べます。

この章のゴール

- どのようなコンポーネントがあり、どんな場面で使えるかを理解する
- Androidアプリ、iOSアプリ、Webでコンポーネントの違いがあることを認識する
- UIデザインにおける色やフォント、レイアウトのポイントを理解する

2-1 コンポーネント

UIの大部分を占めるのが、コンポーネントと呼ばれる部品です。コンポーネントは役割に応じてさまざまな種類があり、ある程度分類されています。コンポーネントの種類を覚えることは英単語を覚えることに近いです。まずはどのようなコンポーネントがあり、それらをどう組み合わせると画面をレイアウトできるのかを学びましょう。これを学ぶと「こういう体験を作りたい」「こういう機能を作りたい」といった場合に「これならあのコンポーネントを使えばいいのでは？」というようにひらめくようになります。

また、この本ではスマートフォンを中心に解説をしていきますが、大きく分けてiPhone/iPadに使われているApple社のiOS、Pixel3/Galaxy/Xperiaなどで使われているGoogle社のAndroidと大きく分けて2つのプラットフォームがあります。そしてそれぞれ、iOSはHuman Interface Guidelines、AndroidはMaterial Designというように、プラットフォームごとにアプリのデザインに関するドキュメントが存在します（それぞれのガイドラインの詳細については4章にて解説します）。

この章では、コンポーネントの紹介とともに、それらが2つのガイドラインでどのように扱われているか、そしてそれに加えてWebでの利用事例を織り交ぜて解説しています。それぞれの解説がiOS、Android、Webのどれのことなのか、わかりにくく感じることがあるかもしれません。そんなときのために、この節の最後に各コンポーネントとそれぞれの名称の対応表（p.60）を用意しておりますので、合わせて読み進めてみてください。

Human Interface Guidelines
https://developer.apple.com/design/human-interface-guidelines/

Material Design
https://material.io/design/

iOS、Androidそれぞれのガイドラインはどちらもとても読み応えがあります。プラットフォームのガイドラインとしてだけではなく、UIデザインの考え方を知るうえでも、一読をおすすめします。

ナビゲーション

ナビゲーションは画面と画面を行き来するためのコンポーネントで、この**画面間の行き来のことを「画面遷移」「遷移」**などと言います。ほとんどのアプリやサービスには何らかのナビゲーションを配置します。タブバーを使って他のコンテキスト（文脈）の画面に移動する場合や、リストやバックボタンを使って異なる階層の画面に移動する場合などがあります。それぞれ、主流となるコンポーネントがあり、それらの使い方や動きを正しく理解することが、わかりやすいサービスを作るための近道です。

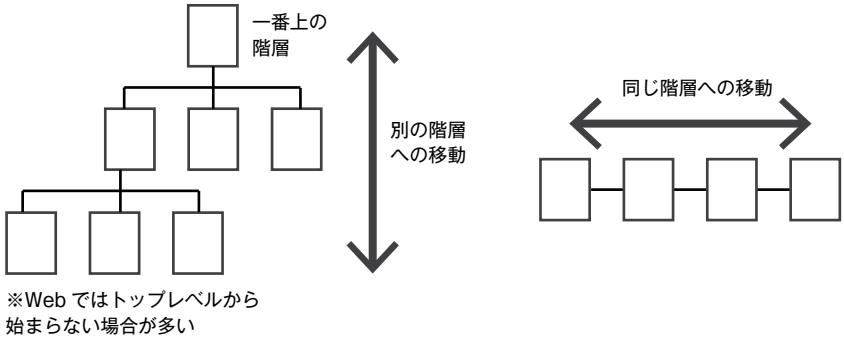


図 2-01 ナビゲーションの概念図

ヘッダー

ヘッダーの解説に入る前に、まず画面全体の大まかなエリアについて説明しておきます。PC やスマートフォンサイトでは大きく 3 つのエリアに分けられます。画面上部（ヘッダー）、画面下部（フッター）、そしてその間にあたる、画面のメインとなる部分です。

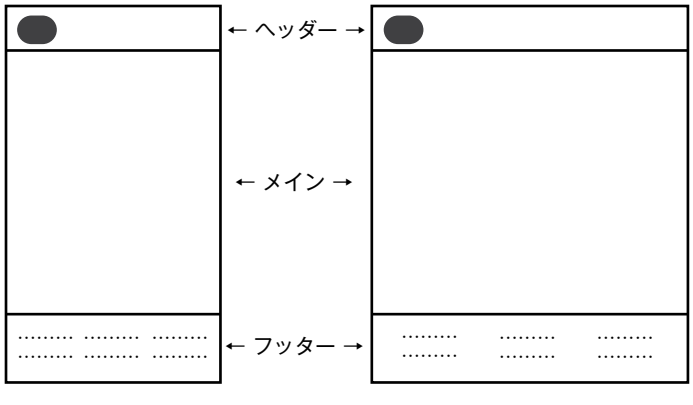


図 2-02 画面全体の大まかな構造図

スマートフォンアプリの場合、iOS アプリでは次の項目で説明するナビゲーションバー、Android アプリ

ではAppBarがヘッダーで利用する代表的なコンポーネントです。Webサイトでは主要コンテンツ、サービスを利用するうえでの基本アクション（ユーザー登録・ログインなど）や困ったときに見るページ（ヘルプ・FAQ）を配置することが多くあります。Webサイトの場合は検索エンジンなどから、トップページではなく下層のページに直接流入することもあり、サービス全体の主要な動線に直接遷移できるようにするためです。

サービスを運営していると、ヘッダーは画面の一等地とも言える場所のため要素が多くなってしまいがちです。何を載せるべきかは精査しましょう。

図2-03　Webサイトのヘッダーの例（GreenSnapのWeb）

PCでは画面サイズが広いため、ヘッダーに主要コンテンツへのナビゲーションを直接配置できますが、スマートフォンの場合は画面が狭いため、後述するドロワーを配置することがあります。

しかし、ドロワーを利用すると、重要なナビゲーションを隠してしまうことも考えられるため、PCとスマートフォン、それぞれ適切なナビゲーションになるよう注意して設計しましょう。

図2-04　スマートフォンのヘッダーにドロワーを利用した事例（GreenSnapのWeb）

ナビゲーションバー

ナビゲーションバーはヘッダーのコンポーネントの中でもiOSアプリでよく利用するコンポーネントです。中央に今表示されている画面のタイトルを、左上には前の画面に戻るバックボタン、画面を閉じるためのクローズボタン、または後述するドロワーを表示するためのボタンをよく配置します。

右上にもアイコンを配置して別の画面への動線を作ったり、後述するサーチバーなどに切り替えるアイコンを配置します。ほとんどの場合、ナビゲーションバーはどの画面でも表示するため、一貫したルールを作ること、常に表示するナビゲーションは何か、画面ごとに変えるナビゲーションは何かを明確に設計しましょう。

図2-05 ナビゲーションバーの利用事例（左：TwitterのiOSアプリ、右：RettyのiOSアプリ）

Androidでは、ナビゲーションバーは画面下部のバックボタンやホームボタンを含むデバイス全体の共通ナビゲーションのことを指すため、コミュニケーションでは注意が必要です。

Android上でのバックボタンのふるまいなど詳しくは4章で解説します。

図2-06 Androidのナビゲーションバー

Androidでは、ナビゲーションバーと同じような役割をTopAppBarが担います。画面タイトルを表示しドロワーやバックボタンなどを配置することが主目的になります。TopAppBarはナビゲーションとしてだけ

ではなく、ContextualActionBar として画面に対してのアクションを表示する役割も担います。

また、TopAppBar は画面上部だけでなく、画面下部でも利用できます（BottomAppBar）。デバイスが大きくなる傾向にある昨今、ヘッダーにあるナビゲーションを片手で操作することは困難です。画面下部に主要なナビゲーションがあるほうが使いやすいため、ナビゲーションバーの役割も今後変わってくることも考えられます。

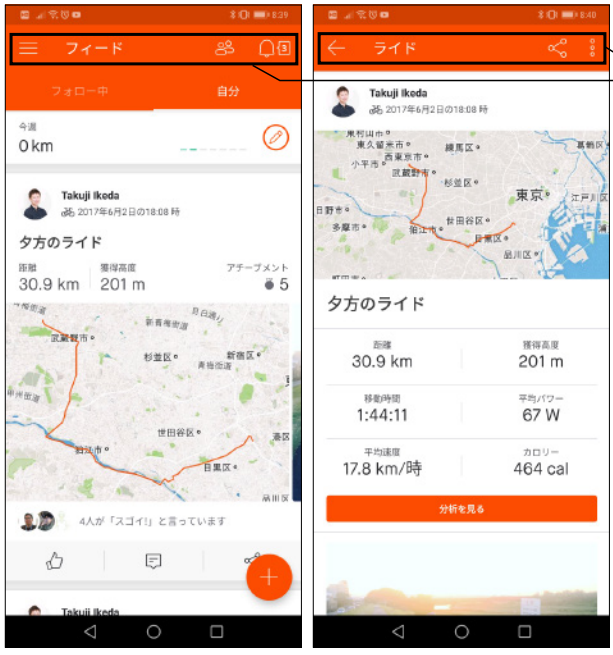


図 2-07　Android の TopAppBar の利用事例（Strava の Android アプリ）

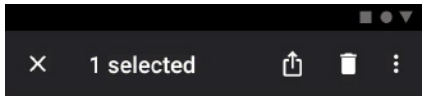


図 2-08　AppBar をツールバーのように利用した事例

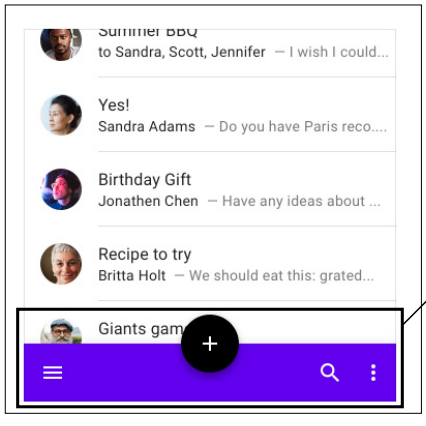


図 2-09　AppBar を画面下部に配置した（BottomAppBar）事例

3

UIの見えない部分を学ぶ

上ノ郷谷 太一

UIデザインは、目に見える要素を並べて終わりというものではありません。私たちがふだん目にしている画面の背後には、サービス全体を形づくる情報の流れが存在します。 この章ではビジュアルを考える前の段階である情報設計や全体の構造について紹介します。 この章を通して、デジタルプロダクトの骨格とも言える情報構造の設計について学びます。

この章のゴール

- 作るものの要件からUIデザインまでに必要なプロセスを知る
- 見た目から独立した情報の構造を設計できるようになる
- プロダクト開発におけるデザイナーの役割と関わり方を理解する

3-1 UIデザインの前に

ペルソナやユーザーの目的、目的の達成までのシナリオや成功指標といった要件を準備できていたとしても、すぐにUIデザインができるというわけではありません。では、どのようにすればUIにできるのか。この章では、アプリのペルソナを作った後から、UIデザインをおこなうまでに何が必要なのかを解説します。

もちろん、UIのラフスケッチをすぐに始めることもできます。アプリの改善であれば、そのアプリのデザインを参考にできますし、新規だとしてもさまざまなデザインパターンを参考にすれば、ある程度「らしい」ものになるでしょう。それも1つの方法で、作ろうとしているもののコアとなる機能（サービスを代表する機能）を切り出して形にしてみると、開発メンバーやステークホルダーとともに、作るものの解像度を高められる場合もあります。しかし、実際に目に見える形に落したものの印象は強いため、柔軟な考え方ができなくなってしまう可能性があります。

あくまで、画面として表れるものは最終的なアウトプットであり、アプリは目的達成の手段に過ぎません。ユーザーがわかりやすい、使いやすいと感じ、思いのままに目的を達成できるUIのデザインをおこなうためには、要件の定義や、構造の設計が必要です。デザイナーはこうしたプロセスを通じて、ユーザーがUI上でどのような情報を目にし、目的を達成するためにどのように行動するのかといった、本質に向き合う必要があります。

3-2 情報設計とは

情報設計はひとことで「わかりやすさを設計すること」と言えます。情報設計と聞いて、何をやればいいのかと感じるかもしれませんが、実はデザイナーが日々、仕事の中でおこなっていたりします。

たとえば、バナー画像なら限られた画像のサイズ内に掲載する情報の優先順位をつけ、把握してもらいたいことが一瞬で伝わるようにしたり、紙面であればレイアウトやコピーなどでメリハリをつけて視線の流れを作ります。プレゼンテーション資料なら、伝えたいことを相手に正確に伝えるだけではなく、興味を持ったり、持ち続けてもらえるように、各ページの構成や表記、全体の展開を考えます。このように、レイアウトや装飾、面の流れを目的に合わせてどのようにするのかを設計するのも情報設計です。

プロダクトの骨格を作る

情報設計は、あらゆるプロダクトの骨格を作ります。2章で触れているコンポーネント、視覚的要素やナビゲーションは、この骨格を基準に選択されます。どれだけ提供する価値が明確になっていたとしても、情報設計なしにおこなわれたUIデザインでは、アプリをうまくユーザーに使ってもらえないこともあります。アプリの目的や各画面の構成、画面のつながりが不明瞭になり、ユーザーはすぐに迷子になってしまうからです。ユーザーがアプリやサービスの本質的な価値をとらえ、迷うことなく操作できるようにするために情報設計が必要です。

適切に情報設計されたアプリは、ユーザーが自由に工夫して楽しく目的を達成できるといった、優れた体験を提供できます。優れた体験は、ユーザーに「アプリを自分の思いのままに操作できている」と感じさせたり、自分自身がレベルアップしたという感覚を生みます。そうした体験が、ユーザーにとってのわかりやすさや、使いやすさにつながります。また、情報設計はアプリのデータ構造の整理にも機能します。情報設計を適切に施すことでアプリ全体がシンプルな構成になり、使いやすさにとどまらず、アプリの拡張性にも影響します。

要件とUIデザインの架け橋

情報設計は作るものの要件とUIデザインをつなぐ架け橋となる存在です。例えると、ここまでの章で学んできた視覚的要素が「単語」だとすれば、要素を組み合わせて1つの画面にしたものが「文」、複数の画面で構

成されるプロダクト全体が１つの「物語」です。しっかりとした骨格を作れば、さまざまな物語、パターンに展開しても、重要な体験のブレを避けられます。見た目から独立した情報の構造を丁寧に設計する力がつけば、デザイナーはユーザーの創造性を刺激する、よりたくさんの表現を試せるようになるのです。

この章の流れと成果物

デザイナーが情報設計を含めたデザインプロセスでおこなうのは、どんな作業なのでしょうか。私はデザイナーの役割はデザインの検討とデザインの伝達だと考えます。ユーザーを目的達成に導くためのデザインの検討を進めるとともに、チームのメンバーやステークホルダーなど、プロダクトに関わるあらゆる人との共通言語を作るプロセスでもあります。

この章のプロセスに至るまでには、サービスコンセプトの設計や、機能仕様などの要件の準備とともに、ペルソナが作成されていると思います。それらを元にデザインを進めます。

ペルソナは、作ろうとしているアプリを使ってくれるであろうユーザーを見つけ、インタビューや行動の観察をおこなった結果、導き出されるターゲットユーザー像です。そのユーザーが達成したい目標は何か、そのためにどういった行動をとる人なのかなど、複数の情報を１人の架空の人格としてまとめます。ターゲットユーザー層が複数あるアプリでは、ペルソナを複数用意してもよいでしょう。

ペルソナにはこういった項目が含まれます。

・顔写真
・氏名
・性別
・年齢
・人物像
・１日の行動イメージ
・達成したい目標
・現状の課題

この章の具体的な流れは次のとおりです。

1. 行動、操作のシナリオの作成
ユーザーが目的達成するまでの理想的な状態の仮説を立てる

2. コンテンツの分類軸の設計
シナリオからユーザーの性質を理解し、それに合わせたコンテンツの分類をおこなう

3. UIモデルの設計

コンテンツの見え方ごとにユーザーが何を見てどうするのかを整理して、フロー図を作成する

4. レイアウトとインタラクションの設計

ペーパープロトタイプでレイアウトとインタラクションを設計する

このようなプロセスを進めていく中で、次のような成果物を準備していきます。

・ユーザーシナリオ

ユーザーシナリオは、作成したペルソナとペルソナが目的を達成した状態までをつなぐ、価値、シーン、行動、操作を具体的にするためのものです。

・コンテンツの分類軸

ユーザーはどのようにコンテンツを探すのかというユーザー視点の分類と、コンテンツはどのような属性を持っていて、どのようにカテゴライズすれば探しやすくなるのかをまとめたものです。

・フロー図

シナリオ、コンテンツの分類軸を元に、情報のまとまり単位でユーザーが「何を見てどうするか」を整理して、線でつないで図示化したものです。

・ペーパープロトタイプ

開発をおこなう前に、アプリがどのように見え、動作し、機能するのかをシミュレーションするツールです。紙は手書きですばやく作成できて、手元で操作もしやすいので、おすすめです。

成果物として準備するものは、プロダクトの規模やフェーズ、チームや組織の状況によって変わってきます。デザインの検討と伝達というデザイナーの役割を意識して、各ドキュメントに記載する内容の粒度を調整していくとよいでしょう。

以降では、ペルソナとして「音楽が好きなサービス開発会社勤務の39歳の男性デザイナー」といった人物像を設定し、音楽系のサービスを題材にして、情報設計を含めたデザインプロセスの流れを解説します。

3-3 ペルソナを動かすシナリオを作る

シナリオは、ペルソナが「何ができると目的を達成できるのか」や「それはどのように実行していくものなのか」といった目的達成までの道のりを具体的にするためのものです。

シナリオ作成は、ユーザーが目的を達成するまでの具体的な行動を理解する重要なプロセスです。なぜこのプロセスが重要なのかというと、ペルソナとしての具体的な人物像と、そのペルソナの目的を達成した状態（ゴール）だけでは、アプリの具体的なデザインを進めるのが難しいからです。シナリオを使ってペルソナに動きを与え、ペルソナが目的を達成するに至るまでの具体的な行動を理解できると、このあとのデザインを進めやすくなります。

注意しなければならないのは、シナリオは、ペルソナという特定のユーザー像の目的とゴールまでのタスクや行動を具体的にするためだけのものではないことです。導き出されたペルソナの認知や行動から、多くのユーザーが持つ認知や行動の特性を明らかにしていきます。抽象的な欲求から、ペルソナという形でユーザー像を具体的にして、目的達成までに必要な性質として抽象化するといった、抽象度の上げ下げを繰り返し、前後のプロセスを行き来しながら理解を深めていきます。

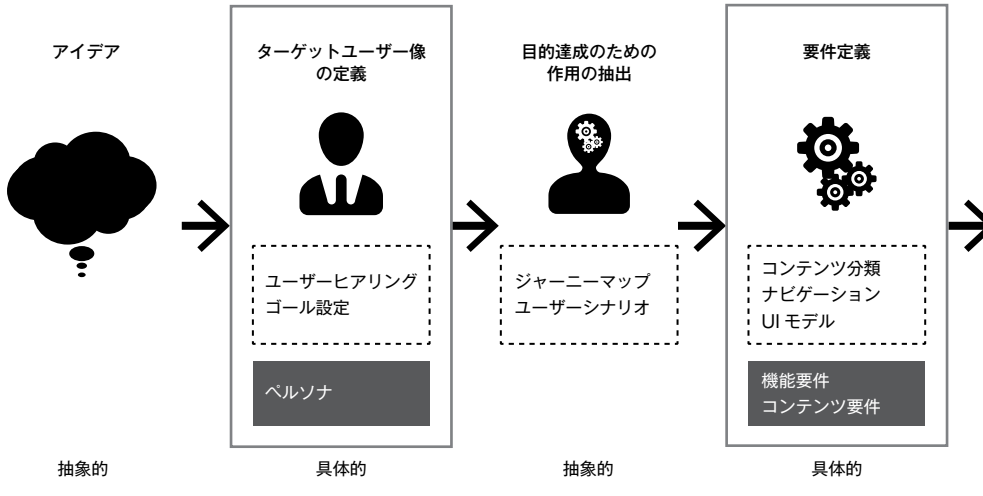


図 3-01　抽象度を上げ下げして要件定義を進める

またシナリオは、これから作ろうとしているものがユーザーにどのような価値を与えるものなのかを、開

4

UIが機能する環境を学ぶ

元山 和之

デジタルプロダクトにおいて、UIがデザインされる場所は主にアプリとWebの２つです。また、アプリには大きく分けてiOSとAndroidがあり、それぞれにお作法とも言えるガイドラインが存在しています。この章ではプラットフォームごとの特徴や気をつけるべき点について紹介します。この章を通して、プラットフォームごとに適したUIデザインについて学ぶことができます。

この章のゴール

- iOSとAndroidの基本的なデザイン・思想について理解する
- スマートフォンアプリ開発に必要な基本的なことについて知る
- Webサービスならではのデザインについて理解する

4-1 スマートフォンアプリのデザイン

サービスやメディアなどを作ろうと考えたときに、Webで作るのかそれともスマートフォンアプリで作るのかというのは最初に考えるポイントかもしれません。昨今ではWeb側の仕様や技術もかなり進み、アプリ並みのことが実現できるようになってきましたが、それでもスマートフォンアプリのほうが勝っている部分もまだまだ多くあります。

たとえば、カスタマイズも含めた高度なカメラコントロール、センサーの使用や、よりリッチなインタラクション、3Dタッチなども含めたさまざまなジェスチャー操作といった部分は、Webで構築するにはまだまだ難易度が高いです。

適切な通知や他のアプリ間との連携、スマートフォン・タブレット・ウォッチ・テレビといった各デバイスのユースケースに合ったサービスの提供という部分もアプリが優位に働くポイントでしょう。その分プラットフォームごとに合ったものを作らなければならず、コストが増えたり、難易度が上がる部分もあります。

まずはアプリケーションが動作する環境について基本的なことや、OSごとのプラットフォームの違いについて紹介します。

スマートフォンアプリの操作

スマートフォンはご存知のとおり画面を指で触って操作するデバイスです。単に画面を指で触るといっても、実はさまざまな操作方法が存在しています。ジェスチャー操作の中にはシステム的に利用されているものや、OSによって慣例となっている操作もあります。すでにわかりきった操作方法もあるとは思いますが、おさらいの意味も込めて改めて見直すつもりで紹介します。

タッチ、タップ（ダブルタップ）

タップはタッチパネルにおいて基本の操作で、PCでいうクリックにあたる操作です。iOSでは画面の位置などによってOS側での補正もあり、Androidに比べて、思っているとおりに押しやすいようになっています。タッチ操作を要求するボタンなどの要素は、基本を縦横44px、最低でも30px以上の大きさを確保しておいたほうがよいでしょう。

図4-01 タッチ、タップ（ダブルタップ）

タップは厳密には画面に触った「Touch Start」、画面に触れたまま動かしている状態でドラッグともいう「Touch Move」、画面から指を離した「Touch End」という操作に分解できます。インタラクションを考えるときなどは、タップしたときの挙動だけを考えるのではなくタッチしたときはどうなのか、離したときはどうなのか、ということも考えるとよいでしょう。

図4-02 ダブルタップ＆ドラッグ

他にも画面を2回続けてタップするダブルタップや、2回目のタップ時に指を離さずそのまま上下左右にドラッグするようなダブルタップ＆ドラッグという操作もあります。iOSのガイドラインには画像などの要素をズームインしたいときなどにダブルタップを用いると書かれています。

またダブルタップ＆ドラッグの例としては、マップアプリで縮尺を変更したいときに上にドラッグすると広域表示、下にドラッグすると詳細表示というような使い方があります。

タッチ&ホールド（プレス、ロングタッチ）

タッチ＆ホールドは画面の要素に触れて、そのまま触れ続ける操作方法でロングタッチとも呼ばれます。iOSではHome画面でアプリアイコンをタッチ＆ホールドすると編集モードに移行するという使い方がされています。

図4-03 タッチ＆ホールド

Androidではリストやタイル表示された写真などでタッチ＆ホールドすると、その要素が選択された状態になり、選択＆編集モード（要素の選択と選択された要素を一括で削除したりするモード）に移行するという使い方がされています。また、選択＆編集モードのあとも指を離さず、ドラッグすることで範囲選択のような操作もできます。

図4-04　左：iPhone Home画面のアプリ編集画面、右：Android Google フォト画像選択・編集画面

フリック、スワイプ、スクロール

フリックとスワイプはどちらも似たような操作で、画面にタッチしてから指を上下左右に振るような操作をいいます。iOSのガイドラインでは、スクロール操作や画面内の要素を起点にサッと細かく動かす操作をフリックといい、リスト要素を左右に動かして裏に隠れていたアクションボタンを表示したり、画面の端から左右に動かして前の画面に戻るような操作がスワイプとされています。このあたりは曖昧なところもあるので、厳密に考える必要はなさそうです。

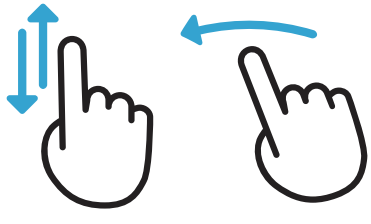


図4-05　フリック、スワイプ、スクロール

ちなみに、画面の端からスワイプする操作をエッジスワイプといい、iOSでは前の画面に戻る操作、Androidではドロワーの表示に利用されていますので、アプリ側で似たような操作を要求させようと思ったときは注意が必要です。

iOSでは一般的には左端からだと前の画面に戻る操作、右端からだと（ブラウザなどで）次の画面に進む操作、上端からだと通知の表示、下端からだとコントロールセンターの表示がされます。

Androidでは左端からのスワイプがナビゲーションドロワーの表示、上端からだと通知などの表示で、下端はバックボタンなどのナビゲーションバーがあるため実質的に利用不可となっています。

Androidの画面下部にナビゲーションバーがあるように、iOSのiPhone Xシリーズでは下部にHome Indicatorというホームボタンの代わりになるものが配置されています。そのため下端から上へのスワイプだけに限らず、下部付近で左右にスワイプするような操作はHome Indicatorの操作と重なって誤動作することがあります。そのため、画面下部でスワイプするような操作はさせないような設計にすることが重要です。

図4-06　左：iOS通知センター、中央：iOSコントロールセンター、右：Androidナビゲーションドロワー

ピンチ

2本の指で狭めたり、広げたりするような操作をピンチイン、ピンチアウトといいます。画像やマップを拡大したり、縮小したりといった操作に使われることが多く、直感的な拡大縮小を実現できます。

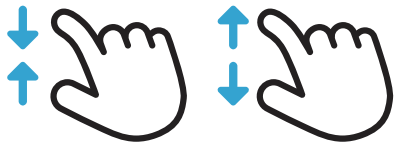


図4-07　ピンチイン、ピンチアウト

ドラッグ&ドロップ

iPadではアプリ間のデータの受け渡しが直感的にできる仕組みとしてドラッグ＆ドロップが利用できます。写真などの動かしたい要素をロングタッチすると動かせる状態になり、スプリットビューで表示したアプリや、Dockに表示したアプリにドロップできます。

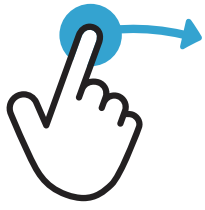


図4-08　ドラッグ＆ドロップ

写真やテキスト、ドローイングなどのデータを扱うiPadアプリではドラッグ＆ドロップの機能に対応しておくことで、ユーザーの利用の幅を広げることができます。

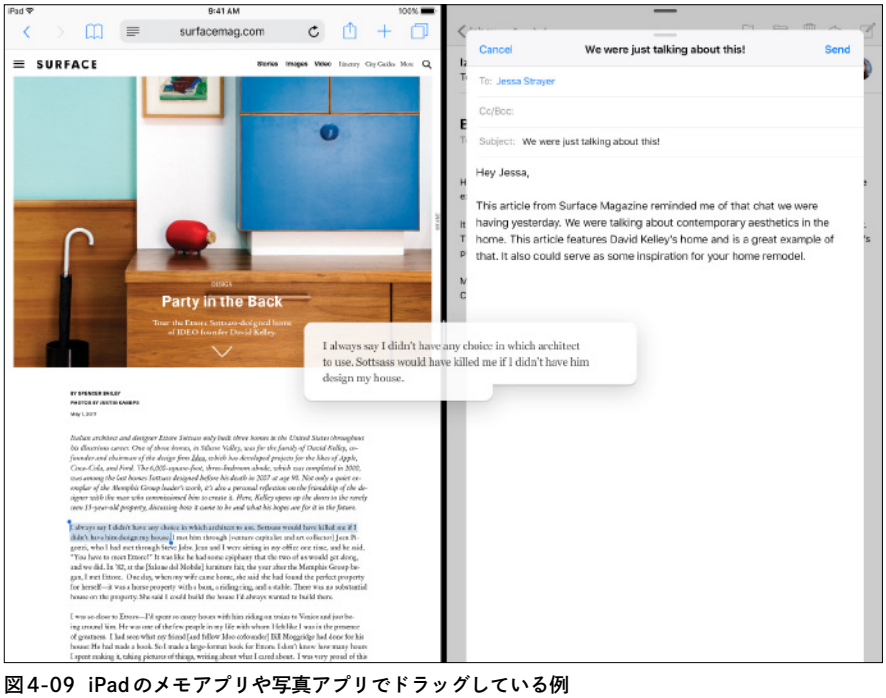


図4-09　iPadのメモアプリや写真アプリでドラッグしている例

5

UIデザインを作ってみよう

吉竹 遼

ここまで、2章ではコンポーネントの種類と役割を、3章ではUIの骨格とも言える情報設計を、4章では各プラットフォームのふるまいについて学んできました。この章では、デザインツールを用いてUIを具現化していくプロセスについて紹介します。

この章のゴール

・UIを作る前の情報整理について知る
・デザインツールの概要がわかる
・UIを作るプロセスを知る

5-1 UIを作る（1）
情報をUIに落とし込む

いよいよUIデザインを組み立てるプロセスについて解説します。

実際に画面に起こす場合、後ほど紹介するデザインツールを利用することになりますが、ツールを触る前に考えることはたくさんあります。つい**「UIデザイン＝ツールを使ってビジュアルを作ること」**と捉えがちですが、ツールを使ってUIを作ることは、**それまでに決めてきた「情報たち」を上手に落とし込む答え合わせのフェーズ**と言えます。

ではその「情報たち」とはどのようなものか。おおまかに3つに分けて紹介します。

現状の整理と目的の確認

現状の整理と目的の確認は、UIを作るうえでとても大切な判断材料となります。このプロセスを挟むことで、これまでたどってきた道と、これからたどる道を改めて理解することにつながります。

現状の整理

デザインツールを起動して、さぁUIを考えるぞ、となるまでに多くのことがチームの中で話されてきたはずです。何が決まっているのか、いないのか。何が足りているのか、いないのか。あるいは自分が理解していることと、していないことは何か。これらを言語化すると、UIを作るために必要な要素を把握できます。

また、システム要件も事前に整理する必要があります。新規プロダクトであれば「開発速度を優先して今回はこの仕様でいく」といった話が、既存プロダクトであれば「こういうことをやろうとすると改修にとてつもなく時間がかかるので、やるには検討が必要」といった話が出てくるものです。あるいはデザイナーが簡単にできると思っていることが、実は開発難易度が高い、といった話もよく出てきます。デザイナーの独断先行でUIを作ってしまって、実装の直前に開発からNOが出る、といった事態は避けましょう。

目的の確認

UIを作る目的は何でしょうか。なぜUIデザイナーは、コンポーネントを組み合わせてプロダクトの画面を作るのでしょうか。**この問いに対する答えを、UIデザイナーはしっかり言語化する必要があります**（あるいは他の人が用意しているかもしれませんが、どちらにしても理解は必須です）。

デジタルプロダクトは、ある意味で**「目的や意図の入れ子」**のようなものと言えるかもしれません。ボタン1つ取っても、色やサイズ、位置には目的や意図があり、そのボタンでおこなえるアクションにも目的と意図

があり、そのボタンが置かれている画面にも目的と意図があり、一連の画面の流れにも目的と意図があります。存在意義と言ってしまってもいいでしょう。

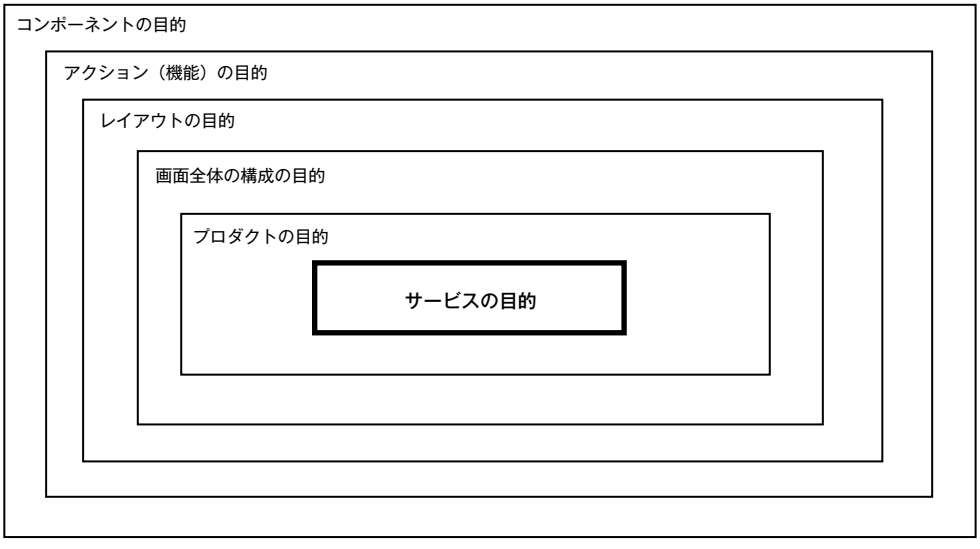


図5-01 コンポーネントの存在意義を繰り返し問うと、最終的にはサービスの目的にまで行き着く

現状を整理して把握し、目的と照らし合わせながらプロダクトのUIを考えることが、このフェーズでUIデザイナーが力を発揮する瞬間と言えます。

ここから先は、もう少し具体的に、UIの前段にあたる情報設計や画面構成、ビジュアルの落とし込みについてお話します。

情報をUIに落とし込む

UIを具現化するには、その前段にあたる情報設計が必要不可欠です。情報設計を骨組み、UIをその周りにある肉付きと考えるとわかりやすいです。シナリオ (p.86) やUI Flows (p.99) など、情報の流れが言語化されたドキュメントをもとに、UIの構成を考えていきましょう。

ドキュメントを理解する

ドキュメントの制作に自分が関わっていない場合、ドキュメントを読んでプロダクト全体の流れや画面やコンポーネントの重要度を理解する必要があります。ドキュメントを作った人と一緒に、認識違いがないか確認する時間を取るとよいでしょう。1つ前の項で書いたように、まずは現状を整理して、目的を確認するこ

とが大切です。作った人から一方的に内容を教えてもらうこともできますが、事前にドキュメントを共有してもらったうえで一度読み込み、疑問点をリストアップしてから確認作業に移ると、自分が理解している点と理解していない点の把握、そして理解していると思った点の認識が合っているかどうかをスムーズに話し合うことができます。ただし、ドキュメントの量が膨大で読み込みに時間がかかるようであれば、最初から一緒に話したほうが早いでしょう。

画面構成を考えながら具現化する

ドキュメントの理解が進んだら、コンポーネントを組み合わせた画面構成を考えていきます。これは、**言語化された情報を具現化していくプロセス**と言えます。

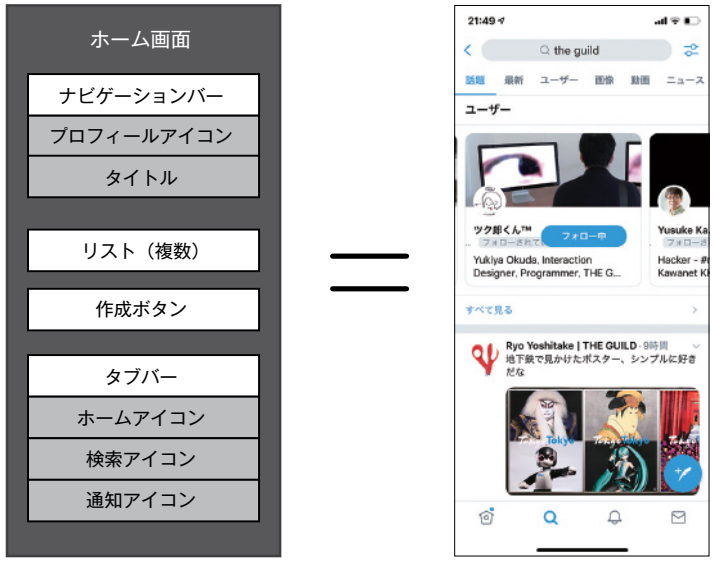


図5-02 言語化された情報を具現化する

UI Flowsやシナリオには「何を見てほしいか」「どのようなアクションを取ってほしいか」「最終的にどんな目的を達成してほしいか」が記載されていますが、それだけを見てUIを作ろうと考えても、目的に沿ったレイアウトにたどり着くのは難しいです。

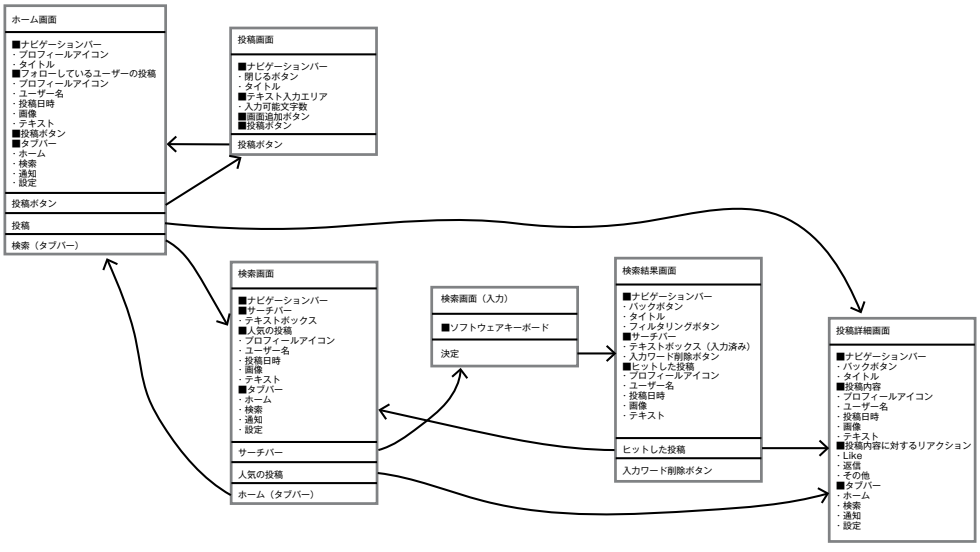


図5-03 Twitterアプリの一部をUI Flowsに起こしたもの

ここから具現化していく過程では、先ほどのドキュメントを理解するプロセスが助けとなります。**プロダクトのコンセプトや想定しているストーリーを反映することで、単なるコンポーネントの集合が意味あるものへと変化します。**具体的に、テキストが主コンテンツのTwitterと、画像が主コンテンツのInstagramを比較してみると、その差がはっきりとわかります。

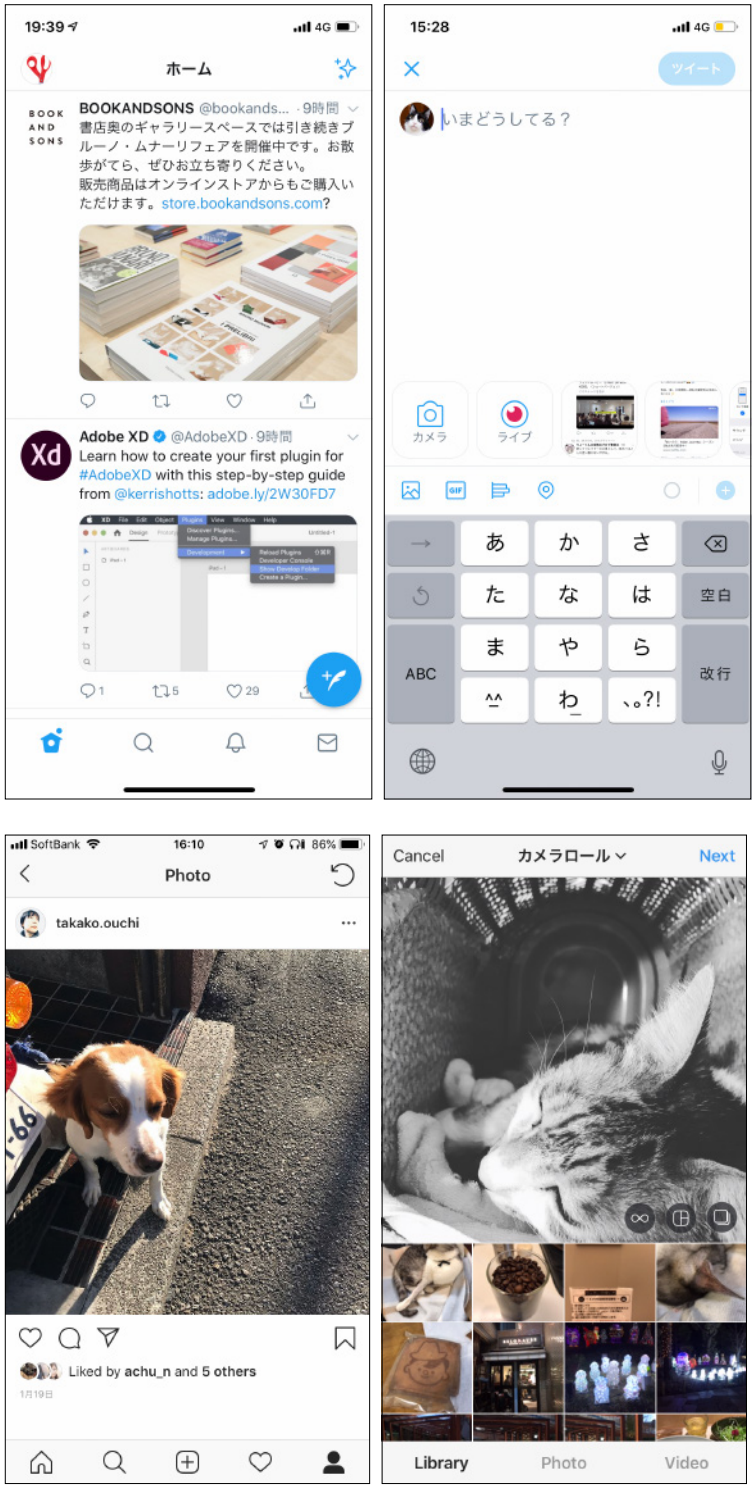


図5-04
Twitter（上）とInstagram（下）の比較。Twitterのタイムラインではテキストが先に来て、作成画面でもテキスト入力がデフォルトだがInstagramでは画像が大きく、作成画面ではまず写真を撮る／選ぶところから始まっている

6

UIデザインができたら

宇野 雄

5章で作り上げた美しいスタイルとレイアウトを、この章ではより使いやすく、よりわかりやすく愛されるものに磨きあげる方法を紹介していきたいと思います。Sketchなどで作ったデザインで満足してしまっていませんか？　実は、UIデザイナーの仕事としてはこれでようやく折り返し地点です。

「自分がいいと思った、チームのみんながいいと思った」デザインから、「使ってくれているユーザーがいいと思う、新しいユーザーが使いたいと思う」デザインへと昇華させるステップを学んでいきましょう。

この章のゴール

- ユーザー体験を意識したUIデザインの流れを理解する
- 目的に応じたユーザビリティの考え方を学べる
- 開発から運用までの注意点を学べる

6-1 体験をデザインする

必要なページのUIデザインができたので、より実際の画面に近い体験を作っていきます。

前章ですでにプロトタイプなどをおこなっていますが、次はより具体的な使い心地の検証を深めていく作業となります。

今までページ単位で、いわば「点」で考えていたデザインが、このフェーズを経て「線」でつながり、より体験として感じられるものになるのです。

ここでは大きく2つの作業を進めていきます。

・画面遷移にトランジションをつけ、ページの要素やレイアウトの検証をおこなう
・ユーザーのアクションに応え、よりわかりやすく使いやくするためのインタラクションを追加する

双方の相乗効果によって、より良いものを作り上げていくのが最終目的です。必ずしも明確なフェーズ分けが必要なわけではありませんが、それぞれのフェーズで利用するツールには一長一短がありますので、作りあげたい体験によって使い分けていくのがよいでしょう。

5章で紹介したデザインツールですが、ここでは、さらに役割ごとに分類をして私の使い方を紹介します。

画面遷移をつける

ページ単位でのデザインに「このボタンをタップ（クリック）したらこのページへ遷移する」というつながりをつける作業です。

・Sketchのプロトタイピング機能
・InVision
・Prott
・Figma

私の場合、Sketchを利用してデザインしたときは、あえて他のツールを使わずSketchのプロトタイピング機能で完結してしまうことも多いです。あくまでここは前後のページのつながりを明確にするための

フェーズですので、複雑な機能はあまり必要はなく、デザインの修正と行き来をしやすいことを優先しています。

インタラクションをつける

より完成品に近いアウトプットを作っていくための、インタラクションに特化したアプリです。

・Origami Studio
・Principle
・Framer X
・Flinto for Mac
・ProtoPie

これらのアプリは、より完成物に近づけたインタラクションが求められるときに利用します。私は単純な「Aのパターンから Bのパターンに変化する」といったものであればPrinciple、より複雑なユーザー操作に反応するインタラクションや繊細な調整をしたり、アニメーションを作成する際にはOrigami Studioをよく利用しています。最近はReact.jsが使えるようになったFramer Xも利用者が増えているようです。

各アプリの特性や使い方だけでも1冊の本が書けてしまいますので、アプリの詳細は割愛しますが、どれも実機での操作と確認ができることが最大の特徴です。それぞれ習得難易度とそれによる表現力が異なりますので、目的や好みでより使いやすく最適なものを選んでみるとよいでしょう。

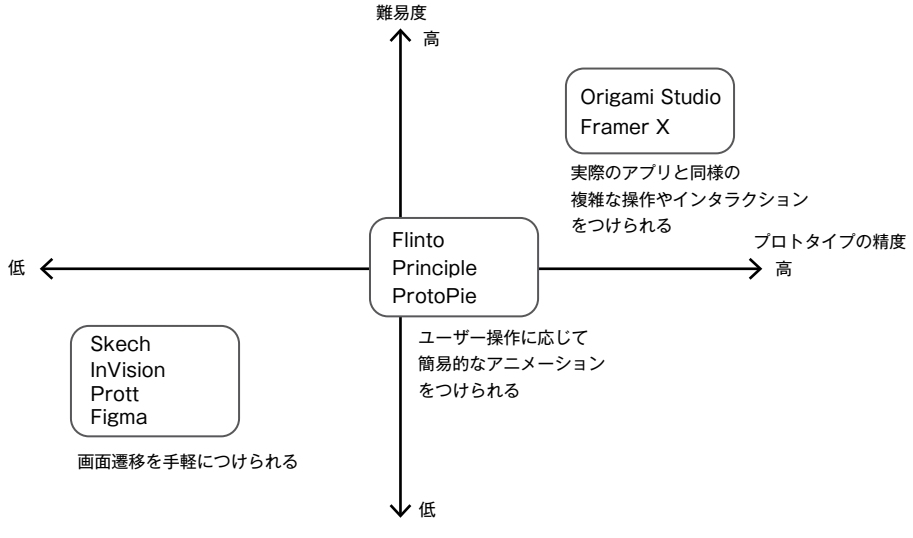


図6-01 各アプリの特性と習得難易度

すでにUIデザインの際に頭の中で「こんなトランジションやインタラクションをつけたらいいのではないか」なども浮かんでいる人も多いかと思います。UIデザインを細かに作り込もうという段階から、実機で確認しながら、画面遷移をつける作業を始めてしまうのも1つの手です。それを実際にやってみると「こんなはずではなかった」や「もっと違う見せ方ができるのではないか」という改善点も浮かんでくることも多くあります。

必要に応じてUIデザインのフェーズと行き来することが大事です。

画面遷移をつける作業に必要なこと

画面遷移をつける作業は、点と点をつなげて線にしたうえで、体験の設計として正しいのかを判断するのが目的です。

UIデザインをしていると、どうしても重要なのは1ページ単位（Sketchでいうと1つのArtboard）での出来栄えだと考えてしまいがちです。事前にペーパープロトタイピングなどをしていても、作り込みをする段階でその視野が狭まってきます。あくまで各モジュールやパーツは一連の体験を最適化するための手段にすぎません。それらの出来を俯瞰的に見るためにも、この作業は必要なフェーズとなるのです。

この段階で、前後のページとの不整合を見つけていきます。実はいらないというページや統合をしたほうが使いやすいページもあぶり出されてくるでしょう。5章で見てきたプロトタイピングはそこにある要素の過不足や流れを確認する作業でしたが、この章では、より作り込んだUIでトランジションをつけていきます。それにより、体験としての具体性を上げることができます。

逆に言うと、このフェーズでその大きな流れを決めておかないと、この後のフェーズでより細かな作り込みをすることで気づきにくくなってしまいます。「UIの細部」と「全体の流れ」という粒度の違う面を交互に見ることで総合的な完成度を上げていくことが大事です。

このフェーズでは何度でもやり直しが効きやすいので、納得のいくまで作り込んでいきましょう。

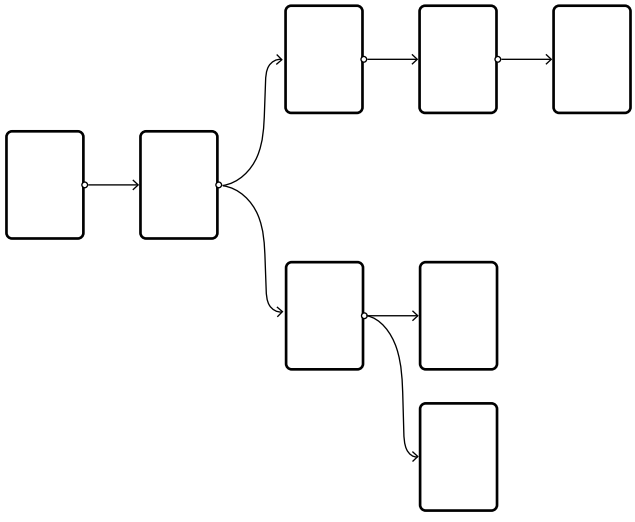


図6-02　ページ単体ではなく、ページとページのつながりが体験を生む

また、ハーフモーダルやトースト（p.46）などの、ページの一部分が書き換わったり要素が追加されたりする見せ方をしたい場合は、この段階でしっかりとそれを見据えておきましょう。

次の項でより細かなインタラクションはつけていきますが、「ページを切り替えるのか」「ハーフモーダルで見せるのか」「トーストで見せるのか」といった部分は大切な情報設計の指針です。それらをここで定めていくつもりで組んでいきましょう。

インタラクションをつける作業に必要なこと

より詳細な各ページの作り込みや特殊なトランジションを作っていきます。

このフェーズは必ずしも必要なわけではありません。特にアプリでは、4章で紹介しているOS標準のトランジションがありますので、先の画面遷移をつけるフェーズで完了してしまいます。本当にシンプルなアプリであれば、このフェーズはスキップしてもよいでしょう。

とはいえ、標準UIでのみ組み立てられたものはふだんから見慣れているため退屈な印象を受けることもあります。そのサービスの中で本当に大事にしたいコアな体験などには、そのサービスを特徴づけるマイクロインタラクション（とても小さな範囲でのフィードバックやアニメーション）をつけるだけでも、大きく印象が変わってきます。

Facebookの「いいね」やTwitterの「お気に入り」が象徴的でしょう。それらはとても細かい装飾でありますが、それぞれのサービスを象徴するアクションであり、強く印象づけ「押すと気持ちいい」というプラス α の体験を生み出しています。

図6-03　左：Twitterのいいね、右：Facebookのいいねのインタラクション

これらのiOSアプリでは共通してアニメーションと一緒にHaptic feedback（振動による触覚フィードバック）が用いられているのも特徴的です。これはiOSの標準機能（iPhone 7以降の機種で利用可能）ですが、ユーザーの行動に対して視覚だけではなく触覚に働きかけることができるため、非常に効果的なフィードバックといえるでしょう。常に使えるわけではありませんが、ときとしては音を利用するのもよいでしょう。（Facebookアプリでは「いいね！」をすると「ポン！」という音が鳴るのを知っていますか？）視覚だけではなく触覚や聴覚に働きかけるのもまたインタラクションの１つの形です。

文字のジャンプ率や色による強弱と同様に、こういったインタラクションを付与することでよりそのサービスの中で「大事であり特徴できない動作」であることがわかってくるでしょう。

すべてにこのインタラクションを付与してしまうと逆に動きがくどくなってしまいますが、ここぞという一番大切な場面でのUXは大事にしたいもの。ぜひともコアとなるようなアクションには細部へのこだわりを持って作り込んでみてください。

“完璧”を目指さない

ただ、私自身が全体を通して大事にしているのが、**完璧なものを作ろうとしないこと**です。

プロトタイプはあくまでプロトタイプにすぎません。特に、Origami StudioやFramer Xはほぼプログラミングをすることになりますので、極論的にはアプリと同じ表現をすることができてしまいます。その一方で、あくまでそれは表面上そのように見えているだけで、実際の実装とのさまざまな違い（プログラミング言語や通信処理、その他の裏側で動く処理）は確実にあります。

そのため、あくまでそのゴールに向けて、プロダクトに関わるメンバーの共通認識を合わせる工程であり、そこに行き着くための手段として使うことが一番大事なことだと考えています。

また、何より、いきなり作り込みをしてしまうと、作ったものに対してどうしても執着が生まれます。「これが一番いい」という頭で作ってしまうと、他の人の意見をフラットに受け入れられません。そうなるとプロトタイプとしては失敗です。

壊すために作る、より気楽に動くものを作る、というような心持ちで望むほうがより柔軟なものづくりとコミュニケーションを生むことができるのではないでしょうか。

作り込みをしてはいけないということではありません。言葉では伝えられない複雑な動きやより繊細なタイミングの磨き込み、むしろそれが非常に大事であることも多くあります。しかし、あくまでページの構成や遷移などを確定させたうえで、その先のフェーズの作業として考えておくことが大事です。

7

UIをデザインする前の心得

坪田 朋

この章では「そもそも、なぜUIをデザインしなければならないのか」といった視点に立ち、UIをデザインする前に意識しておくことについて紹介します。ビジュアルデザインや情報設計のさらに前段階の心得として読んでください。

この章のゴール

- 新規サービス開発の流れを理解する
- チャレンジするモチベーションが湧く
- サービスを開発する精神状態になる

7-1 サービスを作る

何のためにサービスを作るのか

サービスは企業が売上や企業価値を上げるために作ります。**ユーザーに価値を与えた結果、利益を生み出す製品を作るのがデザイン**です。使いやすさだけではなく、売上・利益にも貢献する必要があることを覚えておきましょう。

サービス開発の流れを理解する

これはサービス開発時の流れの一例です。この章では、いくつかのフェーズで意識しておきたいことを紹介していきます。

- チーム内での役割のすり合わせ
- ビジョン、コンセプトの理解
- 情報収集
- ツール選定
- コンセプトデザイン
- MVP (Minimum Viable Product) 制作
- コンセプト評価
- スクラップビルドによるブラッシュアップ
- 製品版のUIデザイン／ロゴデザイン
- ベータテスト
- 改善
- リリース
- グロース

サービスを作る前の心得とチーム内での役割理解

サービス作りはグラフィックを爽快に描くイメージかもしれませんが、実際は**チームとコミュニケーショ**

ンをとりながらコンセプトを具現化していきます。期待されている立ち居振る舞いや成果物のすり合わせを意思決定者としっかりおこないましょう。

同じプランナーでも人によってスタイルが異なります。たとえば「実現したいゴールをテキストでまとめるので、実現方法やUI設計はお任せしたい」ケースと「自らワイヤーフレームを引いてUI設計の大枠は自分が考えるので、グラフィックをお任せしたい」ケースでは、デザイナーに期待されている役割が違います。**どちらがいいという話ではなく、人によってパスの出し方や受け方が異なるので、職種や先入観で判断せず、認識をすり合わせておきましょう。**相手のスタイルに合わせるのか、場合によっては自分がパフォーマンスを出しやすいスタイルを理解してもらうこともときに必要です。

私は仕事を請ける際は「ベストな実行プランと手段を提案したいので、実現したいことをシンプルにテキストでまとめて欲しいです」と伝えて、Dropbox Paperのヒアリングシートにテキストで記載してもらうようにしています。

コンセプトをインストールしてチームの一員になる

ゼロからアイデアを出して自分が意思決定者となる場合を除いて、**サービスや製品を作るときは必ず意思決定者がいます。**たとえばスタートアップなら社長がその役割を担うことが多いですし、メーカーや大企業なら事業部長や新規事業の責任者が意思決定者になります。

作る前に**意思決定者とビジョンのすり合わせをおこないましょう。そのサービスで何を実現したいのか、ユーザーにどのような価値を提供したいのかを正しく把握することでゴールに向かって自分で考えて行動**できるようになります。

間違った解釈でデザインを進めてしまうと成果物のミスリードを起こしてしまうので、方向性の理解はチーム開発に参加する最低条件です。ビジョンが確定せず、ふわっとしている現場は、認識のズレから品質劣化が起こりやすいので、ワークショップを実施してビジョンや方向性をすり合わせるのも重要です。

ほんの少しの掛け違いで設計がズレてしまうので、私は意思決定者から直接ヒヤリングして腹落ちするまでキャッチアップすることを心がけています。

情報収集することで判断する力を身にまとう

新規サービス作りは意思決定の連続です。**不確実性の高い状態で作り上げていくため仮説ベース**で進みます。

たとえば、メニューのレイアウトやテキストの言い回し1つとっても、ターゲットユーザーへのアプローチとして適切なのか、仮説を持って判断しなければなりません。

情報に基づく判断力を磨くことでサービスの成功確度をあげられます。「あの人はセンスや感覚が優れている」と呼ばれている人も、生まれつきの才能ではなく、その分野の歴史や知識などの情報と蓄積した経験からベストな判断に導いています。

以前、中国向けの育児サービスを作ったときは、現地の育児知識や文化を吸収するために中国の主要都市を回り、育児家庭を観察する「エスノグラフィックリサーチ」を実施したり、有識者へのインタビュー、中国の歴史、スタートアップの生態系、インターネットのインフラ状況を調査しました。そして歴史を踏まえて

今後どのように成長するかを考えながらサービスをデザインしました。

結果的に育児のHow to動画メディアを作ったのですが、そのプロセスがなければ現地ユーザーに受け入れられるコンテンツは作れなかったと思います。一見遠回りのようですが、**情報から得た知識は不確実性が高い状態においても適切な判断材料**となります。グラフィック知識だけではなく、関わる分野の知識を蓄えておくのも大事なスキルの1つです。

情報は幅広く収集する

関連情報として、UI設計の判断に必要な情報を網羅的に集めていきます。

- Pinterest、Behance、Dribbbleでビジュアル収集
- 競合サービス、類似サービスのエゴサーチ
- ユーザーヒアリングによるニーズ調査
- 有識者へのエキスパートインタビュー
- エンジニアとのフィージビリティ（実現可能性）調査

Pinterest、Behance、Dribbbleで関連するビジュアルイメージを集めてビジュアルの幅出し、類似サービスの利用ユーザーからニーズや行動観察、有識者のインタビューで表側からは見えない仕組みを理解し、さらにエンジニアと実現可能性のすり合わせを実施して、十分な知識を得てからUIをデザインしていきます。

プロジェクトを円滑に進められるツール選定

プロジェクト次第ですが、ツール選定はチームのコミュニケーションルールを決めるということでもあります。リリース後もサービスは生き続けるので、新メンバーが参加したり自分が抜けても作業しやすいようナレッジを可視化するなどデータ管理を意識しておくと、チームの生産性が向上します。

他職種とのコミュニケーションが円滑に進みやすい、次のようなツールを使うことが多いです。

- Slack：フロー型のコミュニケーション
- Notion：デザインスペックや仕様などストック情報の蓄積
- Dropbox：デザインデータの管理
- Sketch：UIデザインツール
- Abstract：Sketchのバージョン管理ツール
- Zeplin：インスペクタツール

チームの特性に合わせてデザインツール選定をおこないましょう。昨今は共同作業に強いUIデザインツールFigmaを採用する現場も増えてきている印象です。

デザインツールに関しては、こちらを参考にしてみるとよいと思います。

【2018年春】今チェックしておきたいデザインツール12選
https://uxmilk.jp/71315

2018年版：おすすめの人気UIデザインツール徹底比較
https://webdesign-trends.net/entry/6613

コンセプトをデザインする

サービスのコアとなるコンテンツから作っていきます。まずはビジュアルを作り込まずに、量を作って幅広い案を出したらそれを叩き台に方向性の議論、アイデアの追加削除を繰り返して納得のいくまでコンセプトを磨きます。

特に初期段階ではメンバー個々の理想が異なり、判断軸がブレるため、プレゼン型の一発レビューは避けましょう。デザインの初期段階から頻繁にシェアし、議論に巻き込むことでメンバーの認識も統一されていきます。

デザインはデザイナーだけで完結できません。開発メンバーとフィーリングを合わせていくことがサービス開発では重要です。

何をコアとするかはサービスによってさまざまですが、チャットサービスならトークリストからチャット画面への遷移、ライブ配信サービスなら演者と視聴者のコミュニケーション画面など、もっとも大事だと考えられるコンテンツを具現化することで、価値の実現手段を形にして機能に落とし込んでいきます。

以降では、私がデザインしたサービス事例を紹介します。

bosyuの事例

bosyuは、TwitterなどのSNSで簡単に募集することができるサービスです。募集タイトルと本文を入力するとSNSのタイムラインで表示されるOGP画像が生成され、URLをTwitterに貼り付けるとタイムラインに次のような画像が表示されます。

図7-01 bosyu

私自身、求人や何かしらで人を探すときにTwitterとFacebookを使うことが多かったのですが、課題に感じることも多く、それらの課題を解決するサービスを実現しました。

募集側の課題

・募集ツイート後、favが本気応募かノリか見分けがつかない
・DMが大量にきてしまい、全員に返事ができずに申し訳ない
・TwitterやFacebookのメッセージ欄が埋もれてしまう

応募側の課題

・リプライで返事をすると公開されてしまうのが嫌
・興味はあるけど、いきなりDMを送るのはハードルが高い
・勇気を出してDMを送ったが、返事がないと何だか切ない

bosyuではユーザーに一番多く見られる箇所のTwitterカードのデザインから作り始めました。**SNSを活用したサービスなので、どのように表現すればTwitterのタイムラインで興味を持ってもらえるかのポイントに絞って**デザインを考えました。

コアバリューを**SNSで同じ内容をつぶやくよりbosyuを使ったほうが応募が増えること**と設定したので、デザインの良し悪しは、それが達成できるか、余分な要素を盛り込まず最短で実現できるかで判断しました。