

Programming Boot Camp Learning Phase #3

Bubble Basic #2

2024/11/23

事前準備

- 本日は前回作成したペットの健康管理アプリに、デザインやロジックを追加していきます
- その後、さらに API 連携やチーム開発といった応用についてもお話しします
- 本日の講義用に多少手を加えていますので、開始時点をそろえるため、こちら側で用意したアプリケーションを複製したものを利用してもらいます
- 複製したアプリケーションを配布しますので、@imahashi 宛てに、Bubble のアカウントを作成したメールアドレスを伝えてください。
- また、動作確認用に実際にペットを 5 件くらい登録しておいてください。



今日やること

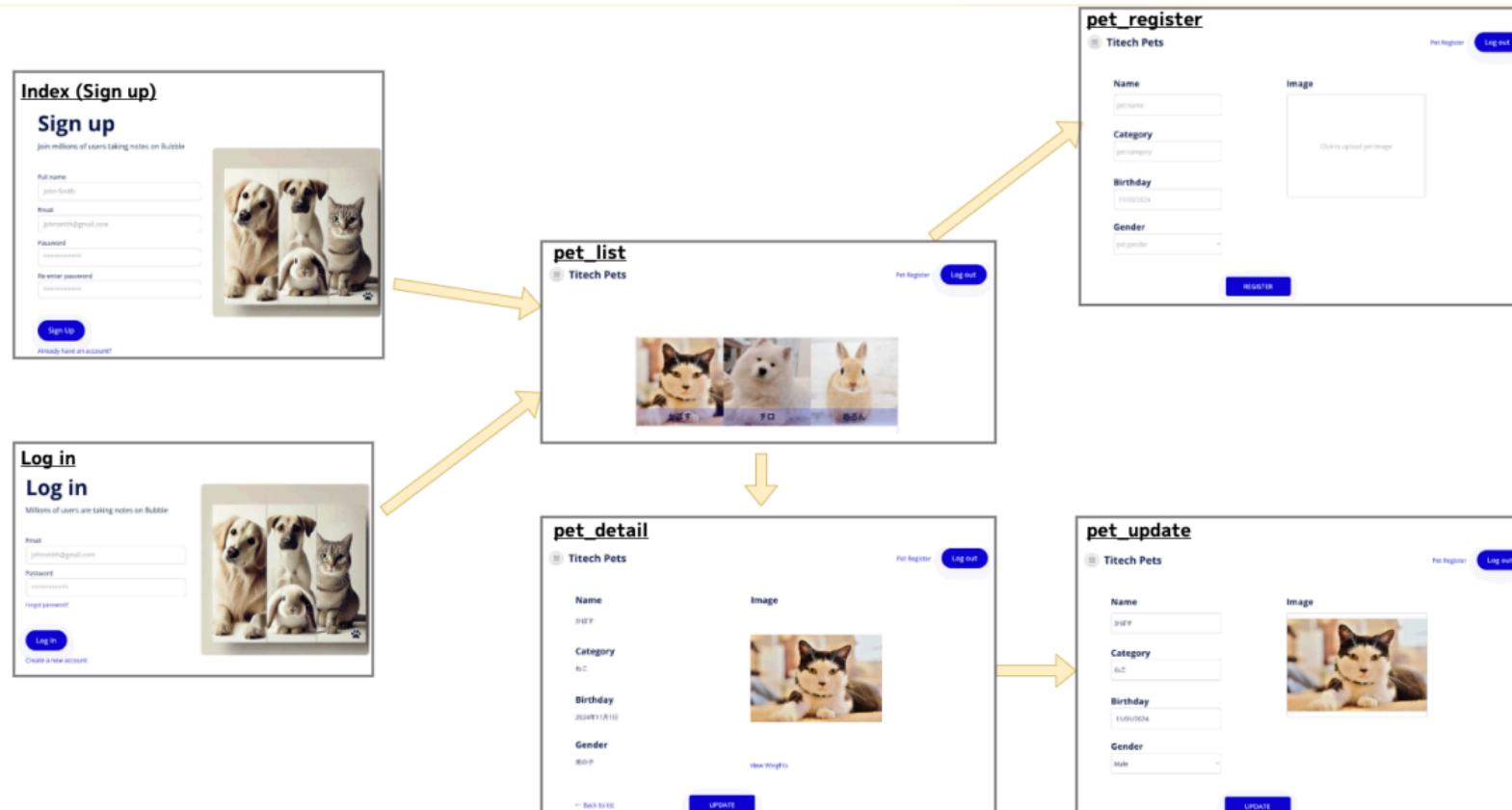
- 前回のおさらい
- デザインを作りこむ
- ロジックを作りこむ
- 外部システムと連携する
- チームで開発する

前々回のおさらい

- Bubble はビジュアルプログラミングツールとうたっており、画面からポチポチと操作して、見た目や動きをプログラミングできるツールです。
- Web アプリケーション前提であり、ディスプレイサイズにあわせた対応をいれスマホや PC に対応させます。
- 前回お休みの方はこちらの資料でキャッチアップしましょう
 - <https://github.com/GuildWorks/isct-pbc-2024/tree/main/docs/learning-phase-1/Bubble1/pdf>

前々回のおさらい

ペット管理アプリケーションのペットの登録、一覧、詳細、体重記録の画面を作りながら、Bubble の基本である、Design/Workflow/Data の使い方を学びました。



今日やること

まずは前回までのアプリケーションに対して、デザインやロジックをさらに作り込んでいきます

最終的にはこんな感じになります

ペットリスト (PC/スマホの表示切り替え)



PetLog



Log out

Register



Log out

Register



ペット詳細（ロジックを使ったデータ加工表示）



PetLog



Log out

← Back to List

[View Weights](#)

Consult

Name (Initial)

taro (T)

Image



Category

dog

Birthday

Age (as Dog/Cat)

2019/11/11

5 (36)

Gender

Latest Weight

女の子

10.5kg

アドバイザー用ペットリスト（ロジックを使った権限制御）

PetLog

Log out

Register

 taro	 taro	 tama
 mikan	 shiro	 aka

AI ペットアドバイザー (ChatGPT との連携)



Log out

← Back to Detail

Feel free to ask about anything.
A pet care advisor will help address your concerns.

あまりご飯をたべてくないのですが。。

Send

taroくん（2019年11月11日生まれの犬、最新体重10.5kg、計測日：2024年11月21日）についての相談ですね。食欲が落ちているとのこと、心配ですね。

まず、以下の点を確認してみることをお勧めします。

1. **フードの種類や質**: 餌の種類やブランドが同じであれば、たまには変更してみるのも良いかもしれません。特に嗜好性の高いフードやお肉系のものを試してみると食いつきが良くなることもあります。

2. **メンテナンス**: 定期的な運動や遊びが不足していると、食欲も落ちてしまうかもしれませんので、taroくんの生活に適度な活動を取り入れてあげると良いでしょう。

3. **健康状態の確認**: 食欲の低下は何らかの健康問題のサインの場合もありますので、特にこの状態が数日続くようであれば、獣医師に相談して健康状態をチェックしてもらうことをお勧めします。

4. **ストレス要因**: 新しい環境や変化（引越しや新しい家族の加入など）が影響している可能性があります。

動画検索 (Youtube との連携)



Log out

[Register](#)

search



Google ログイン（Google との連携）

Sign up

Join millions of users taking notes on Bubble

Login by Google

では、さっそくはじめていきましょう。

まずはデザインを作りこんでいきます

モバイルファーストでいこう

なお、今回はスマホの画面サイズをメインにしてつくっていきます。

モバイル端末に体験やデザインを最適化するアプローチを「モバイルファースト」と言います。

インターネットサービスの利用においてスマートフォンがPCを超えており、サービスや事業フェーズによっては、スマートフォンを主としたモバイル端末での利用を優先して、進めることも多くなっています。

モバイルファーストってどんなことをする？

- レスポンシブデザイン：スマホ向けのデザインを基に作り、PC やタブレットでの表示を調整する。
- タッチ操作の優先：マウスよりも指での操作が中心となるため、ボタンやリンクを押しやすくする。
- 高速ロード：モバイルデータ通信の速度や制限を考慮して軽量化を重視する。 など

Bubble でモバイルファーストはどうやる？

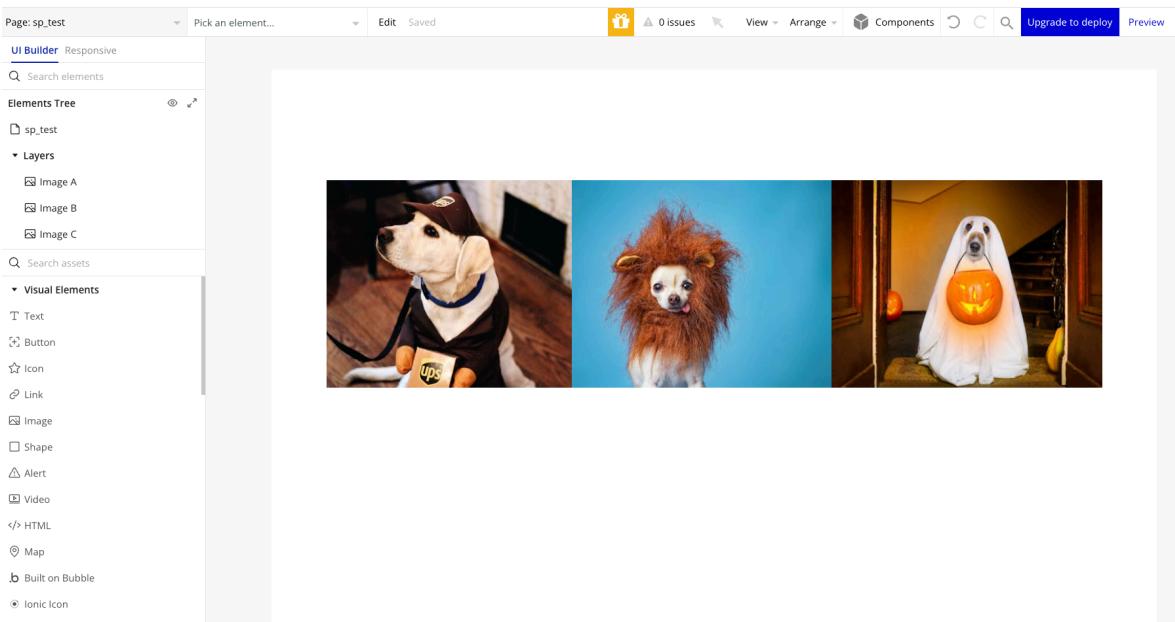
最も簡単にできる方法としては、開発時からページの幅をスマートフォンに合わせた幅にして開発を進めてあげることです。試して見ましょう。

まずは、空のページを立ち上げてください。

- 本日みなさんに配布したアプリを開く
- 画面左上の b ロゴの右側の `Page:xxxxx` となっている部分をクリック
- `Add a new page...` をクリック
- 適当な名前をつけて、`Create` をクリック。これで空のページができます。

初期のまま、単純に作ってしまうと PC に最適化される

- 前の会では意識しなかったかもしれません、この時点で画面幅は PC よりの幅になっています。
- このままの画面幅で単純にアプリケーションをつくっていくとスマートフォンでの表示には適さないアプリケーションになります。
- 試しに、大きめの画像を横に並べて配置してみましょう。



スマートフォンの画面幅では画像が切れてしまう

プレビューしてみましょう。

PC ではうまく表示出来ていると思いますが、スマートフォンだと見切れてしまします。

*PC のウィンドウサイズを狭めてみるとか、デモ表示の際の URL をスマートフォンに送ってスマートフォンで見てみてください。

bubble-2024-2-ongoing.bubbleapps.io/version-test/sp_test?debug_mode=true



My Page

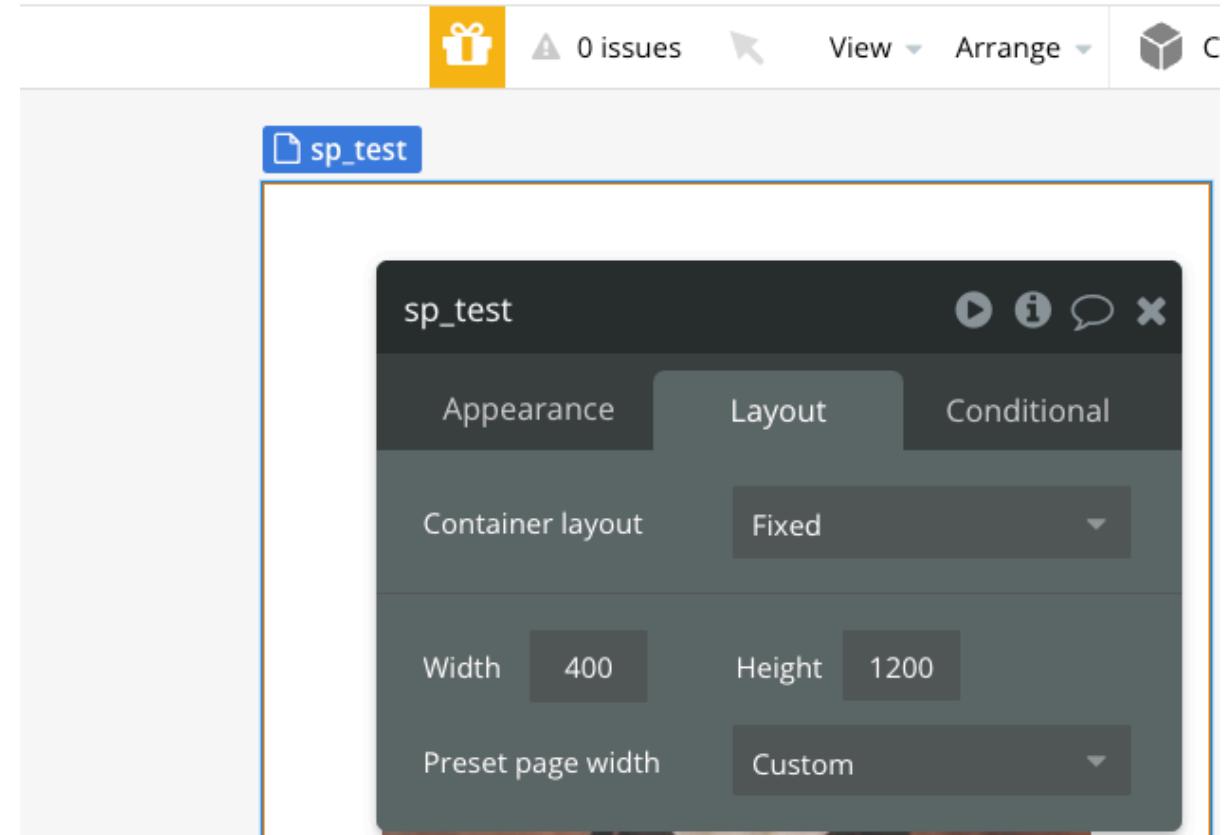


スマートフォンの画面幅を意識して開発しよう

そこでスマートフォンの画面幅を意識して広がりすぎないように作るのが最も単純な対応になりますが、そのままで開発しにくいので、開発時の画面幅をスマートフォンの画面幅に設定してしまいます。

スマートフォンの画面幅に設定する

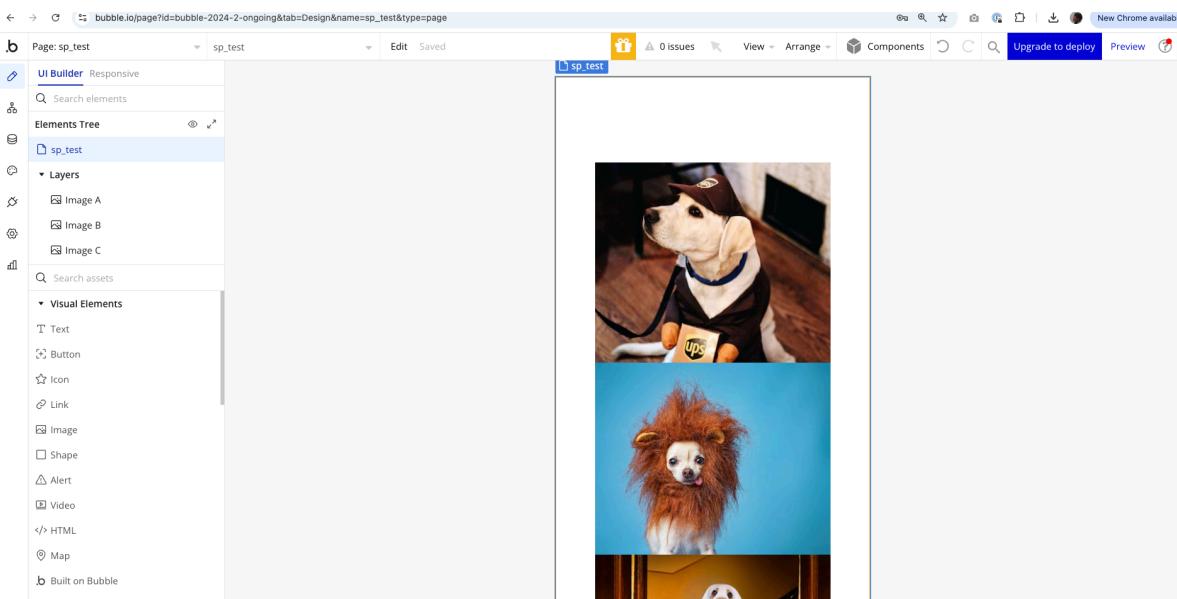
- 先ほど作った画面の Design ビューを開き（開いているはずです）
- 画面自体を選択するために、画面の外側の灰色の部分をクリックします
- Layout タブを開きます
- Width に 400 、 Height に 1200 を指定します。



スマートフォンの画面幅にあわせてレイアウトを整える

- 画像をその幅でいい感じにならぶように修正しましょう。
- 中央揃えでキレイに並べたいなら、Shift を押しながら全て選択し、右クリック（もしくはダブルタップ）して、`Center horizontally` を選択するとすべて中央そろえしてくれます。

こんな感じですね。



こんな感じになります

スマートフォンに最適化された表示になっています。

PC だと横幅がもったいないですが、PC でも表示はできていて利用可能です。モバイルを優先した最も単純かつ強力な対応で、スマートフォンを主としてよいフェーズでは強力です。



様々なディスプレイサイズへの対応について

なお、一概にモバイルといっても、ディスプレイサイズは様々です。

スマートフォンもあればタブレットもあります。また、単にスマートフォンといっても、iPhone Pro MAX のような大きな画面をもつスマートフォンがあったり Galaxy Fold のような広く広げられるものもあります。

ディスプレイサイズは様々なので、あらゆるディスプレイサイズに対して、もっと細やかに対応させていくことができます。

そのための方法がレスポンシブウェブデザインという方法になります。こちらは後ほどふれていくこととします。

モバイルファーストでいこう

今回、みなさんに配布している Bubble アプリは、前々回、京極がレクチャーした内容を先ほどのやり方で画面幅 400 でレイアウトしなおしたものになります。本日はこちらをつかっています。

では戻って、デザインの作り込みについて話していきます。

デザインの作り込みでやること

- ディスプレイサイズに合わせた画面をつくろう
 - レスポンシブウェブデザインという手法を使って、ディスプレイサイズに見た目を制御します
- Style を使ってみよう
 - Style を編集・追加したり、個別にスタイルをあてます

ディスプレイサイズに合わせた画面をつくろう

ディスプレイサイズに合わせた画面をつくろう

- Web アプリケーションは PC やタブレット、スマートフォンといった様々な端末で利用されます。
- 端末毎にディスプレイサイズが違うのですが、それらに対応する手法としてレスポンシブウェブデザインというデザイン手法があります
- 画面サイズに応じて、要素が伸びる／縮む、折り返す／折り返さない、表示する／表示しないといった見た目の切り替えが柔軟に行われる手法になります。
- 実現方法として、固定した配置やサイズを指定するのではなく、配置やサイズを決めるためのルールを指定します。
- Bubble は初期設定では、固定の配置やサイズが指定されるようになっていますが、各種ルールを指定することもできるようになっています。

よく使うルール

Bubble でレスポンシブデザインを実現するために、よく使うルールとして以下のようないわゆる「3大原則」があります。

1. 親要素内の配置ルール
2. 要素のサイズ決定ルール
3. 表示の有無ルール

これらのルールを組み合わせて、レスポンシブな画面デザインを実現します。

なお、これらのルールは Bubble に限らず、Web アプリケーション全般に通じる考え方でもあります。

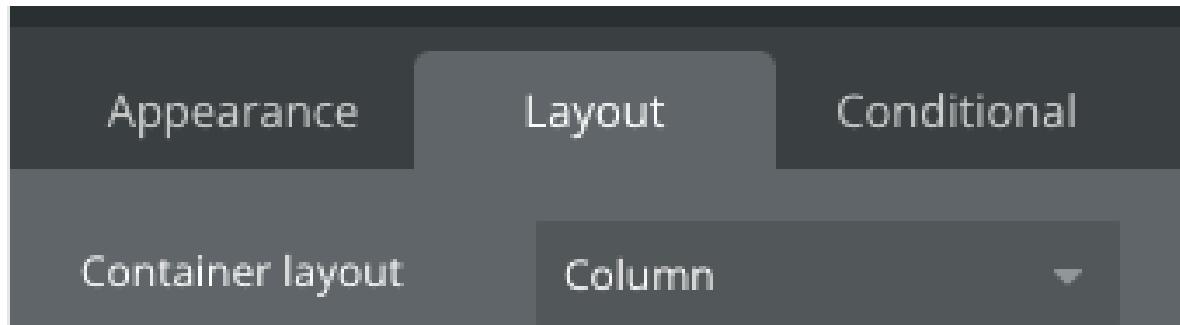
後ほど、一緒に画面に組み込んでいきますが、難しい考え方もあるので、まずは概要を説明していきます。

ルール1：親要素内の配置ルール

親要素の中でどう配置するかという、ルール指定になります。

リピーティンググループなどのグループやページ全体など個々の要素を囲むような親要素をBubbleではContainerと呼ばれています。

Containerでは中に含む子要素の配置ルールを指定できます。



子要素の配置ルールとしては以下の 4 つがあります。

- Fixed : 配置場所を固定で指定する
- Aligng to parent : 親要素に対する相対位置を指定する
- Row : 行方向（水平方向）に並べる
- Column : 列方向（垂直方向）に並べる

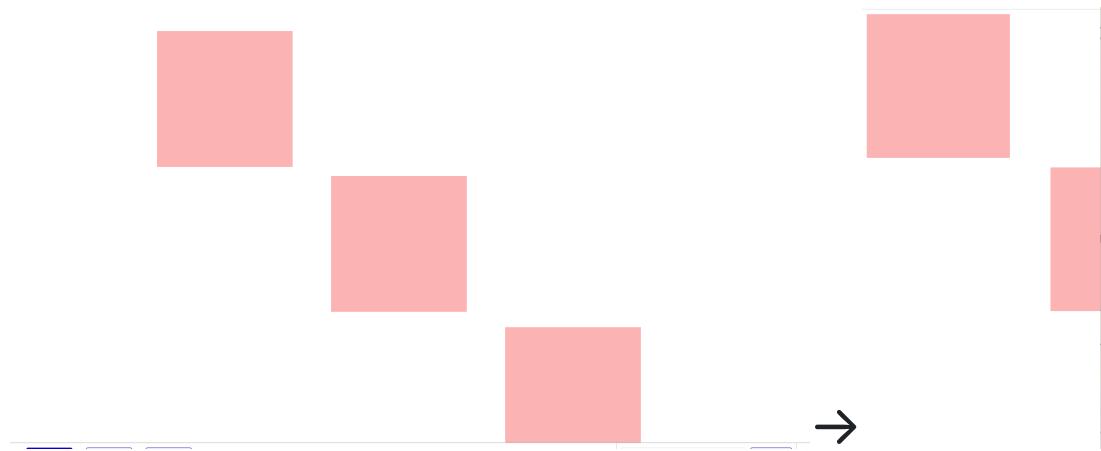
順に説明していきます。

Fixed：配置場所を固定で指定する

配置場所を固定で指定するルールです。配置場所をピクセル単位で指定します。

Bubble で親要素を置いた場合の初期設定となります。

固定で指定しているため、画面幅を変更しても指定した位置から変更されません。下の例では、画面の外にはみ出てしまっています。



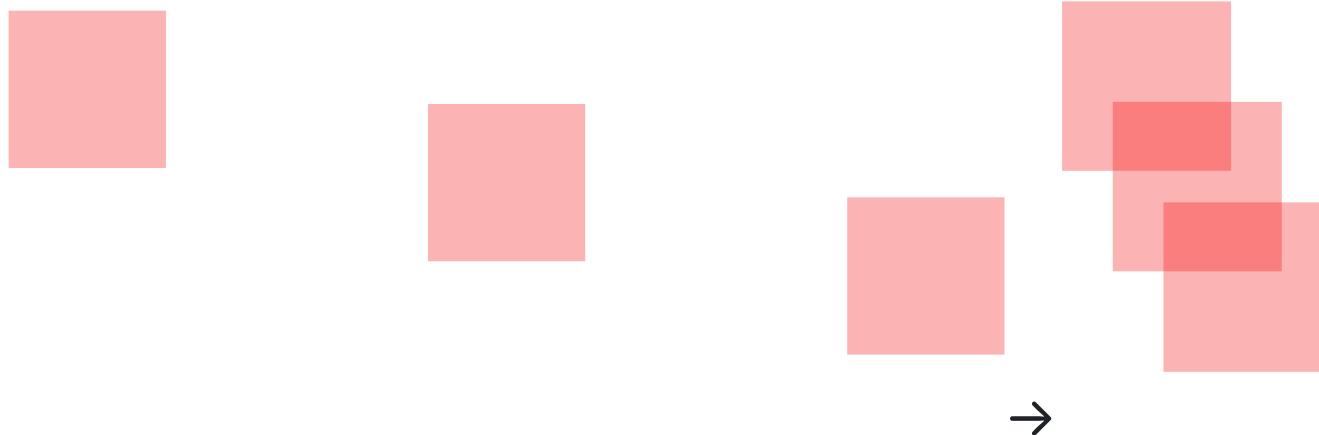
Alignng to parent : 親要素に対する相対位置を指定する

親要素に対する相対位置を指定するルールです。

Bubble では、9つのエリアから配置場所を指定できる。



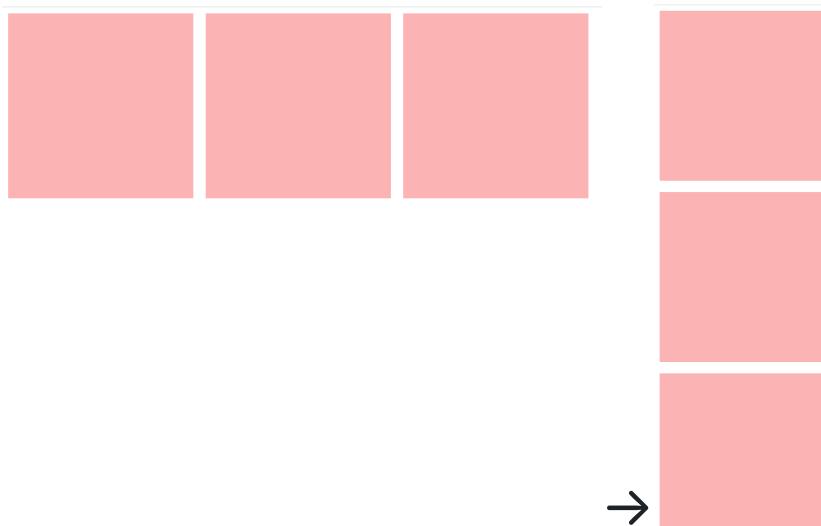
画面幅を変更したら、相対位置を維持して動く。下の例では、画面幅を狭めると、左上・中央・右下という相対位置を維持して動いている。



Row：行方向（水平方向）に並べる

行方向（水平方向）に並べるルールになります。行は自動で折り返します。

下の例では、画面幅を狭めると、行が折り返していき、結果として縦に並んでいます。



Row : 行の中での水平方向（左右）の配置を指定できる

要素ごとに行の中での垂直方向の配置を指定できます。



(Left-aligned)



(Centered)



(Right-aligned)



(Space-around)

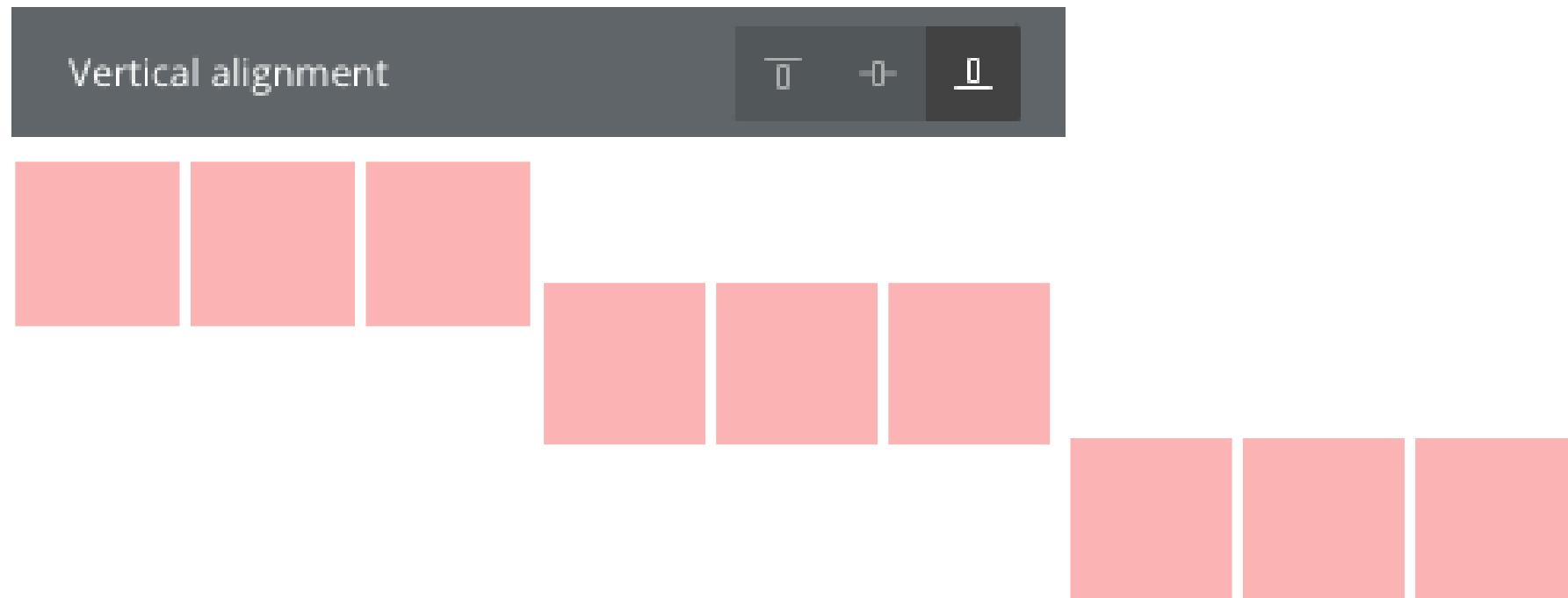


(Space-between)

Row : 行の中での垂直方向（上下）の配置を指定できる

行の中での垂直方向の配置を指定できる。

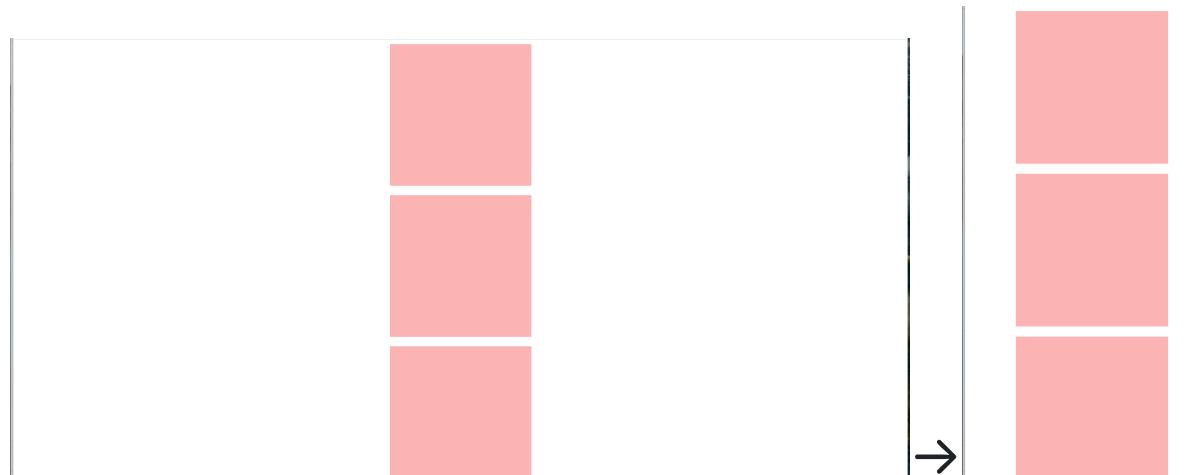
※これは親要素ではなく、子要素に対して指定するものになります。



Column：列方向（垂直方向）に並べる

列方向（垂直方向）に並べる。

下の例は左右中央寄せで縦に並べており、画面幅を小さくしても中央寄せを維持したまま縦に並んでいる。



Row と同様に水平、垂直の配置を指定できる。（水平・垂直で指定できる内容は反対）

ルール1(おさらい)：親要素内の配置ルール

親要素の中でどう配置するかという、ルール指定になります。
子要素の配置ルールとしては以下の4つがあります。

- Fixed：配置場所を固定で指定する
- Alignng to parent：親要素に対する相対位置を指定する
- Row：行方向（水平方向）に並べる
- Column：列方向（垂直方向）に並べる

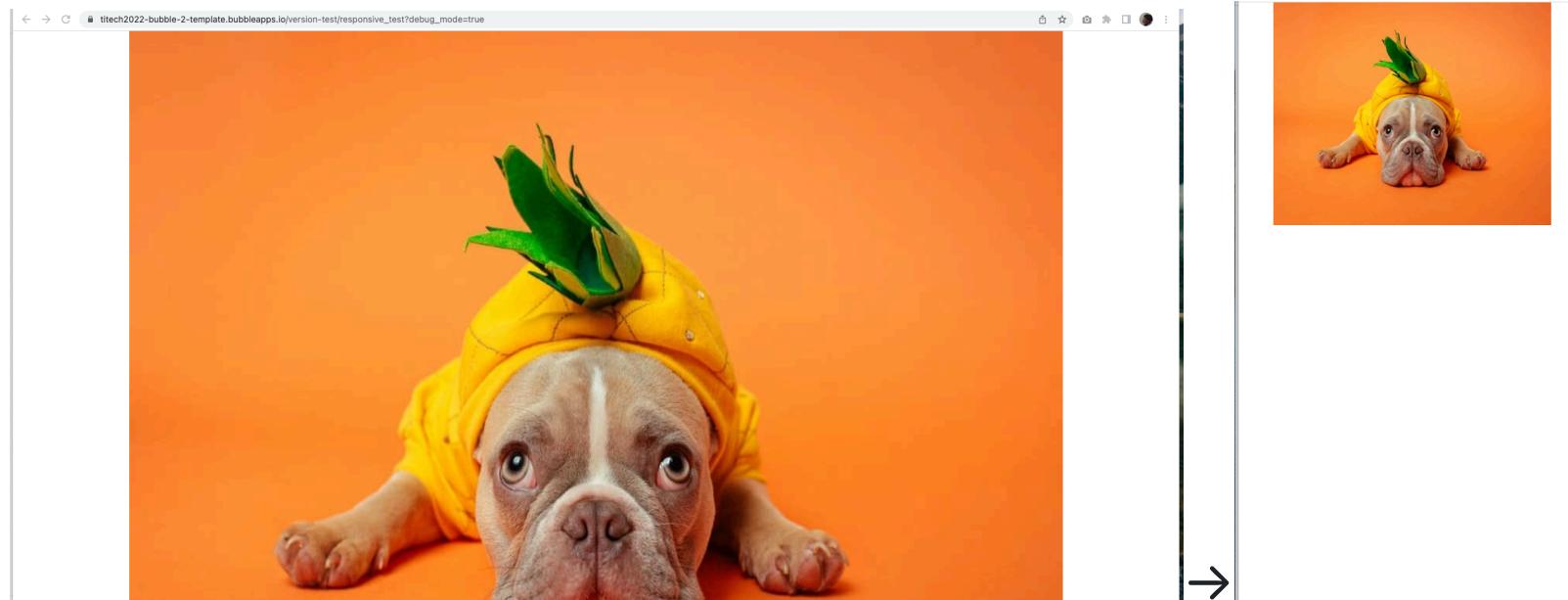
ルール 2：要素のサイズ決定ルール

画面幅によって、大きくなったり小さくなったりできるように、固定サイズを指定のではなく、サイズを決めるためのルールを指定します。主に以下の二つのいずれかを利用します。

- 親要素の大きさに対するパーセンテージを指定する
- 伸び縮みした場合の最大・最小サイズを指定する
 - 最大最小は指定せずに、無制限にすることも可能

パーセント指定

下の例は、画面に対して幅が 80%になるように指定しています。画面幅を小さくすると、80%の割合を維持しながら画像サイズも小さくなります。



下の例は、幅を最大 800px、最小 300px に指定しています。

300px から 800px の間で伸縮しますが、画面を広げきっても 800px 以上は大きくならない
ですし、逆に画面を狭めても 300px 以下にはなりません。

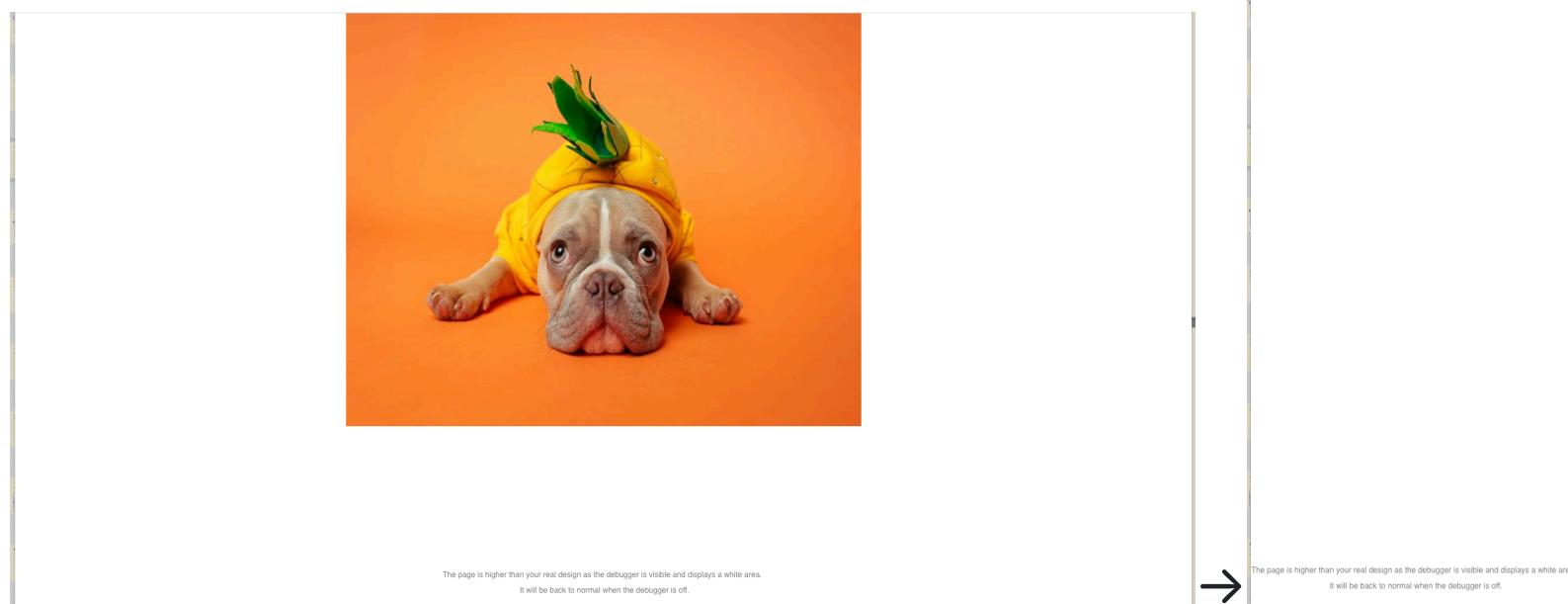


The page is higher than your real design as the debugger is visible and displays a white area.
It will be back to normal when the debugger is off.

ルール 3：表示の有無ルール

画面幅の下限や上限を指定して、画面内の要素の表示・非表示を切り替えることができます。

下の例は、画面幅が 992px 以下になった場合に画像を表示しないという指定をしています。



画面が大きい場合だけ情報を多く表示したいと言った場合によく使います。

よく使うルール(おさらい)

Bubble でレスポンシブデザインを実現するために、よく使うルールとして以下のようなものがあります。

1. 親要素内の配置ルール
2. 要素のサイズ決定ルール
3. 表示の有無ルール

では、実際に使ってみましょう。

(再確認) 事前準備

- 本日は前回作成したペットの健康管理アプリに、デザインやロジックを追加していきます
- 本日の講義用に多少手を加えていますので、開始時点をそろえるため、こちら側で用意したアプリケーションを複製したものを利用してもらいます。
- 複製したアプリケーションを配布しますので、@imahashi 宛てに、Bubble のアカウントを作成したメールアドレスを伝えてください。
- また、動作確認用に実際にペットを 5 件くらい登録しておいてください。



ペット一覧ページにレスポンシブデザインを適用します

ペット一覧ページにレスポンシブデザインを適用します。

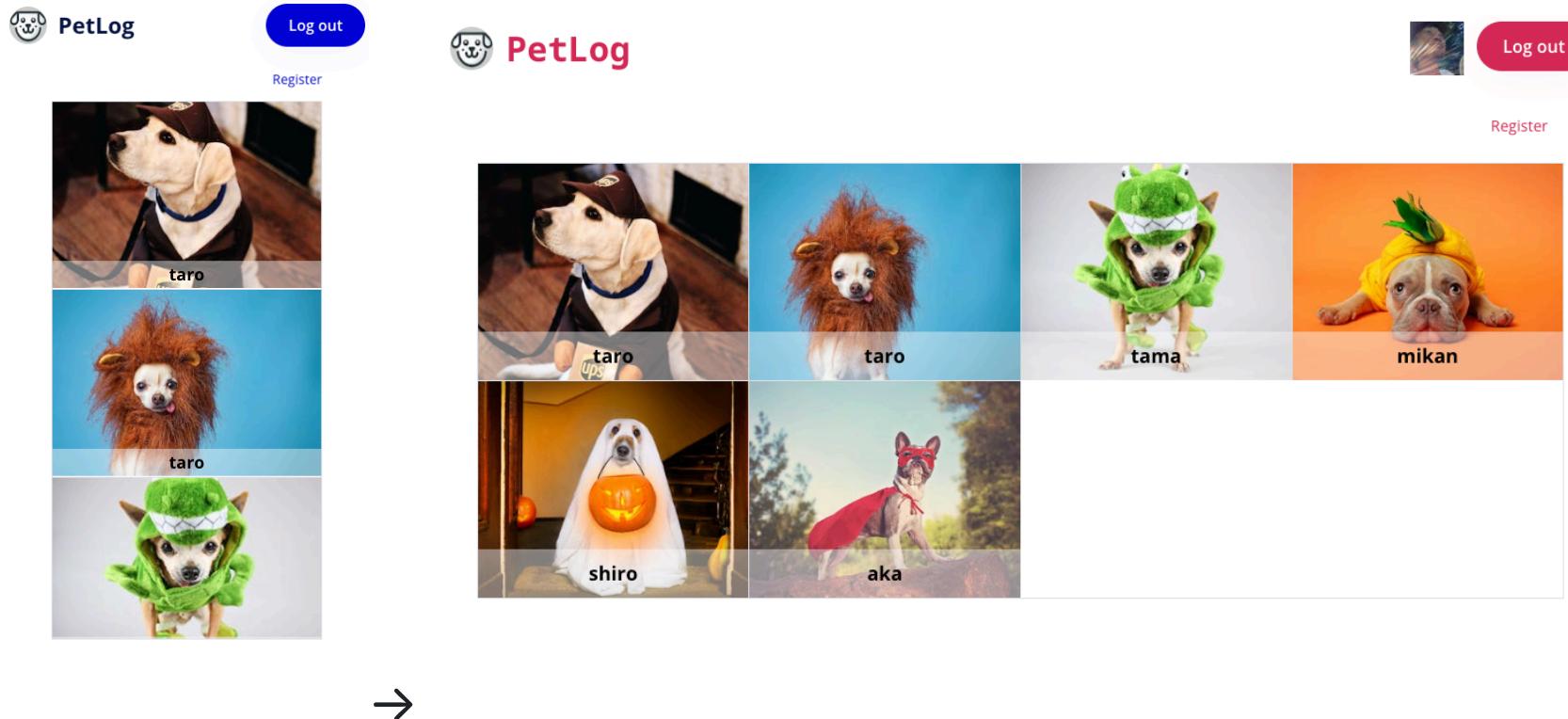
以下のルールを利用します。

1. 親要素内の配置ルール
2. 要素のサイズ決定ルール

またサイズの指定やリピーティンググループ特有のレスポンシブ対応用の設定も組み合わせていきます。

完成イメージ

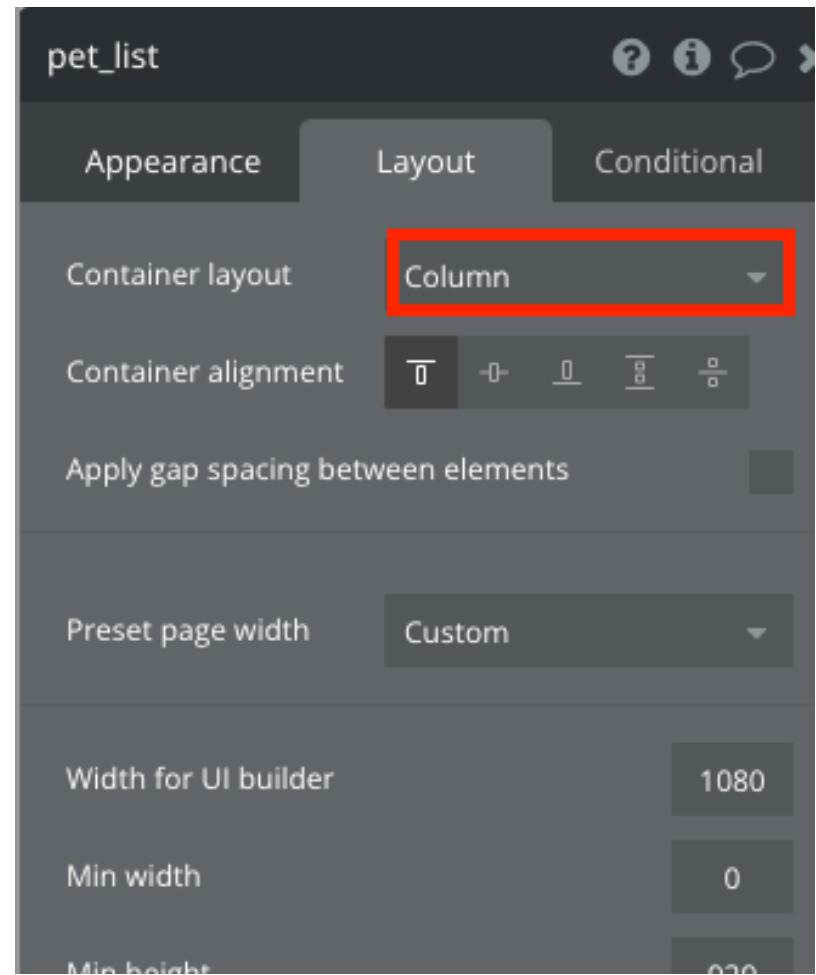
対応すると、画面幅に列数が柔軟に変わり、すべてのペットが一覧で見れます。

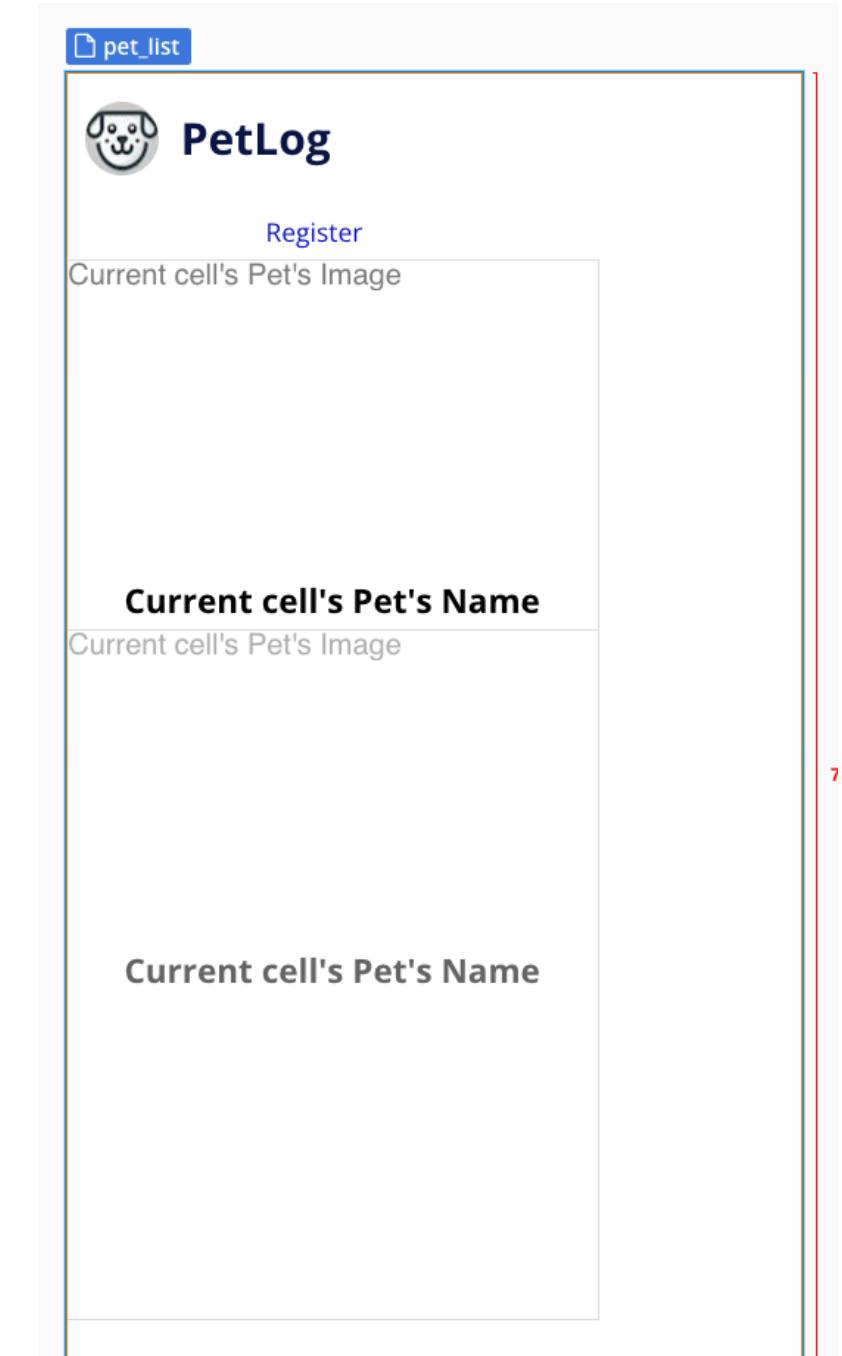


まずは、親要素となるページの
Container layout を変更しましょ
う。

今回は、 Column を指定します。

- pet_list の画面を開く
- ページの空白部分をダブルクリックして、ページ 자체の設定ウィンドウを開く
- 設定ウィンドウの Layout タブを指
定する
- Container layout を Column に
変更する





そうすると、勝手に画面内の要素が縦に
詰めて並ぶようになりましたね。

Column で設定された親要素の直下の
子要素は列方向（垂直方向）に勝手に整
列されるのです。この並びを基本とし
て、整えていきます。

続いて、リピーティンググループに対して設定をいれていきます。

- リピーティンググループ `pet list` の設定ウィンドウを開き、`Layout` タブに移動する
- 以下の設定をいれる
 - Horizontal alignment: `centered`

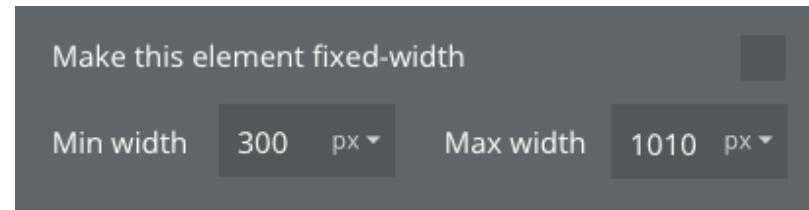


ペット一覧ページ直下の要素が列方向に並ぶようにしていますが、その際にこのリピーティンググループは左右中央寄せに整列されるようになります。

さらに、以下の設定をいれる

- Make this element fixed-width: チェックなし
- Min width: 300px
- Max width: 1010px

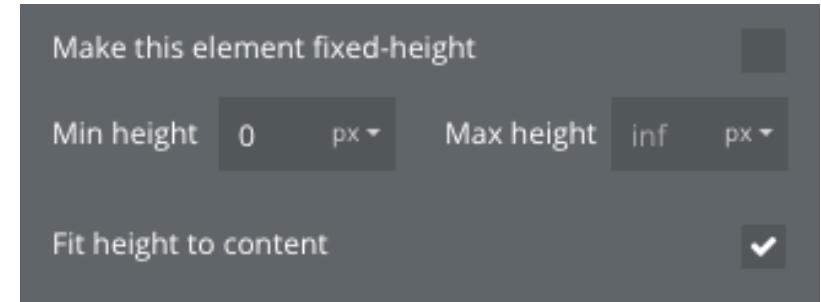
このリピーティンググループの幅は固定にせず、親要素（ここでは画面自体）の幅に応じて柔軟に伸縮します。ただし、極端なサイズだと見えにくいので、最小幅は 300px とし最大幅は 1010px までとしています。



以下の設定をいれる

- Make this element fixed-height:
チェックなし
- Min height: 指定なし
- Max height: 指定なし(inf) ※おそらく infinity の略
- Fit height to content: チェック

このリピーティンググループの高さは固定せず、中身に合わせてどこまでも伸縮するという設定です。



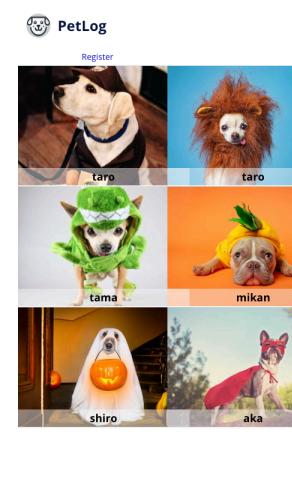
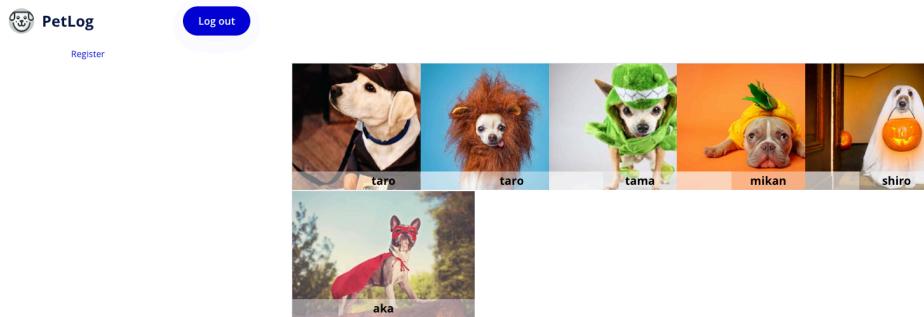
次はリピーティンググループの設定ウィンドウの Appearance タブに移ってください。
以下を設定していきます。

- Set fixed number of rows: チェックなし
- Min height of row: 200px
- Set fixed number of columns: チェックなし
- Min width of column: 250px

列数と行数は固定せず、行と列の最低幅を指定するようにしています。これによって、行数や列数が最低幅を維持しながら、表の大きさによって柔軟に切り替えられます。

では、プレビューを表示してみましょう。

まだ不細工ですが、画面の幅・表の幅に合わせて、列数・行数が変動するようになりました。



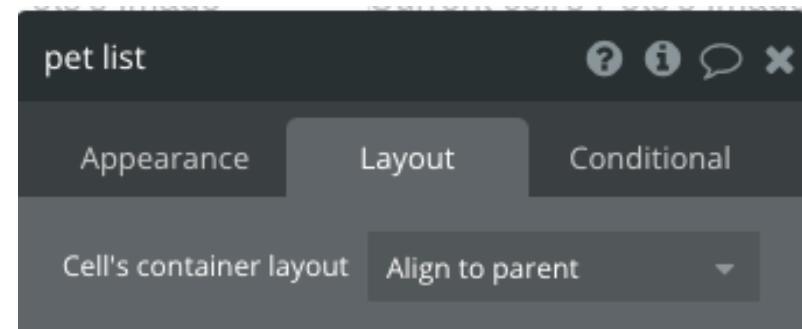
でも、セルの中で左によってしまっていたり、幅を小さくしていく際に意図しない余白ができててしまっていたりしますね。修正していきましょう。

またリピーティンググループの設定ウィンドウの Layout タブに移ってください。

以下を設定します。

- Cell's container layout: Align to parent

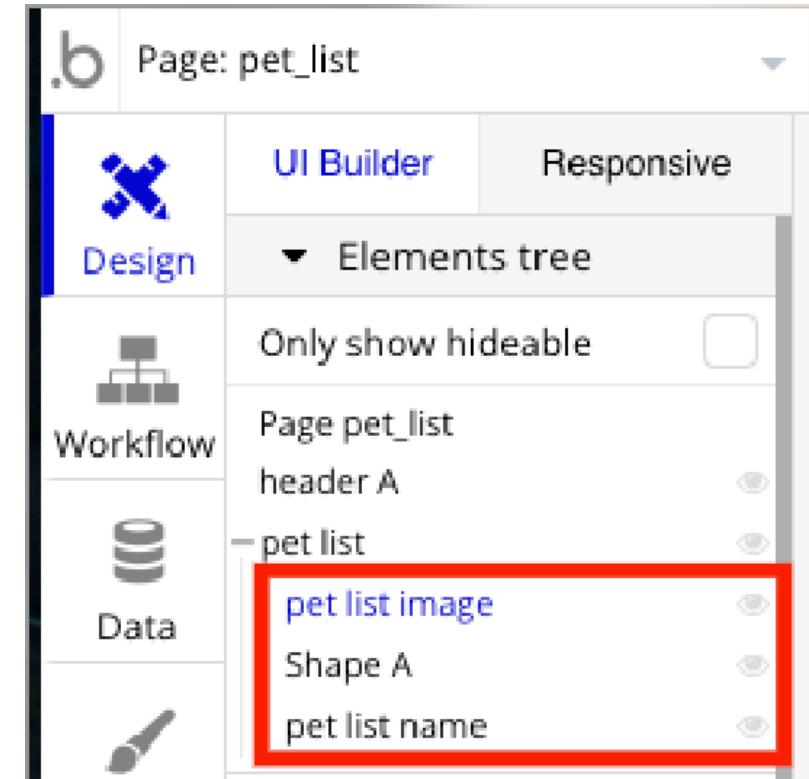
親要素（ここではリピーティンググループの一つ一つのセル）に対して、相対位置で要素を配置できるようになります。



次にセルの中身の要素のレイアウトを
ていきます。

要素が重なっていて選びにくいので、画
面左の **Elements tree** から指定しま
す。

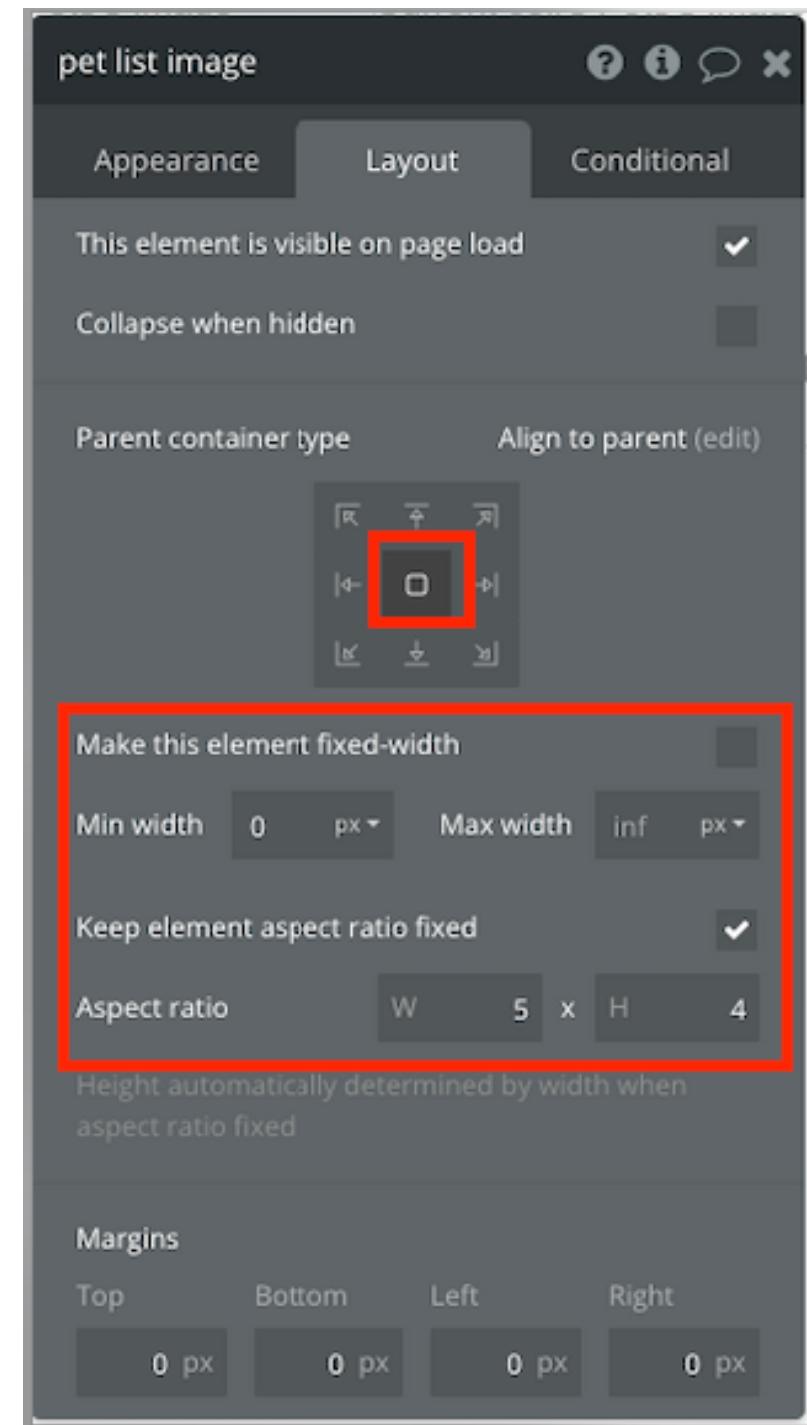
- Design メニューの **UI Builder** の
最上部に **Elements tree** という
部分があります。
- **Only show hideable** のチェック
がついていたら外してください。
- * **pet list** しか表示されていない
場合は、**+** をクリックしてツリー
を開いてください。

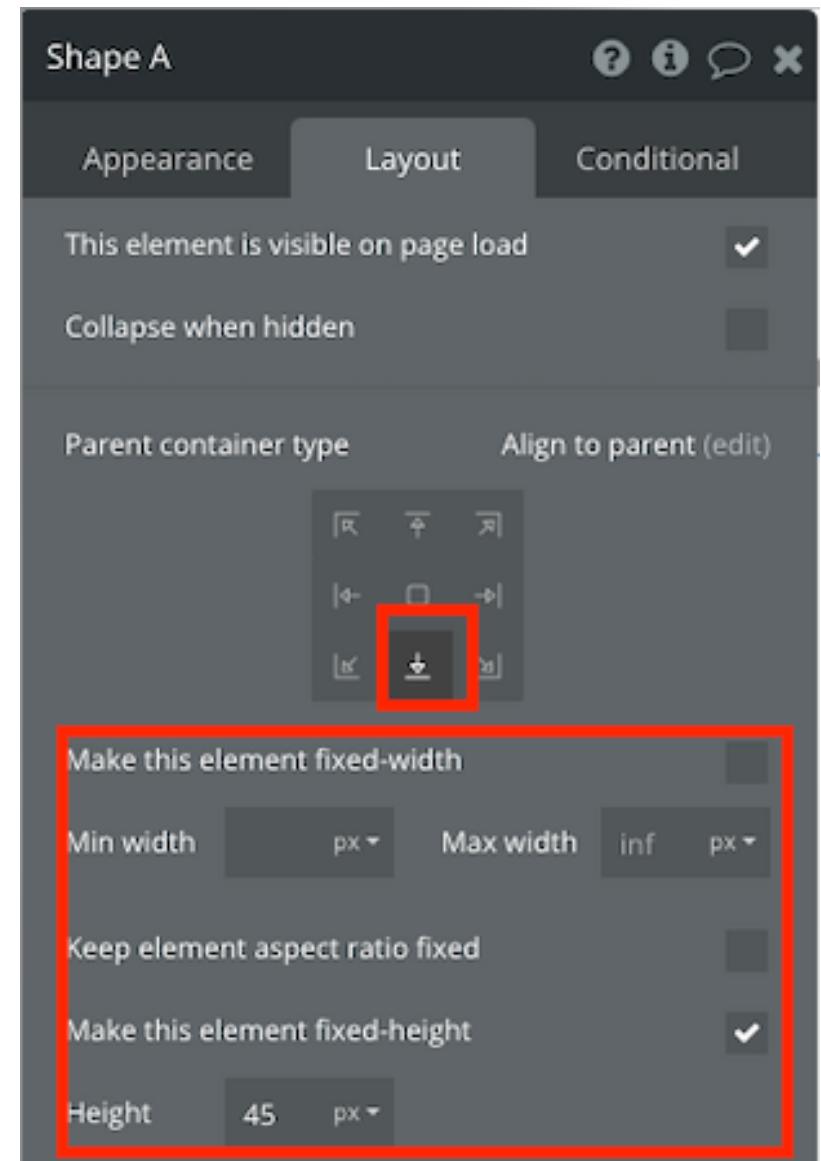


では、`pet list image` から順に設定していきます。

- `Elements tree` の `pet list image` をクリックして設定ウィンドウを開く
- `Layout` タブ内で右の画像と同様に設定してください。

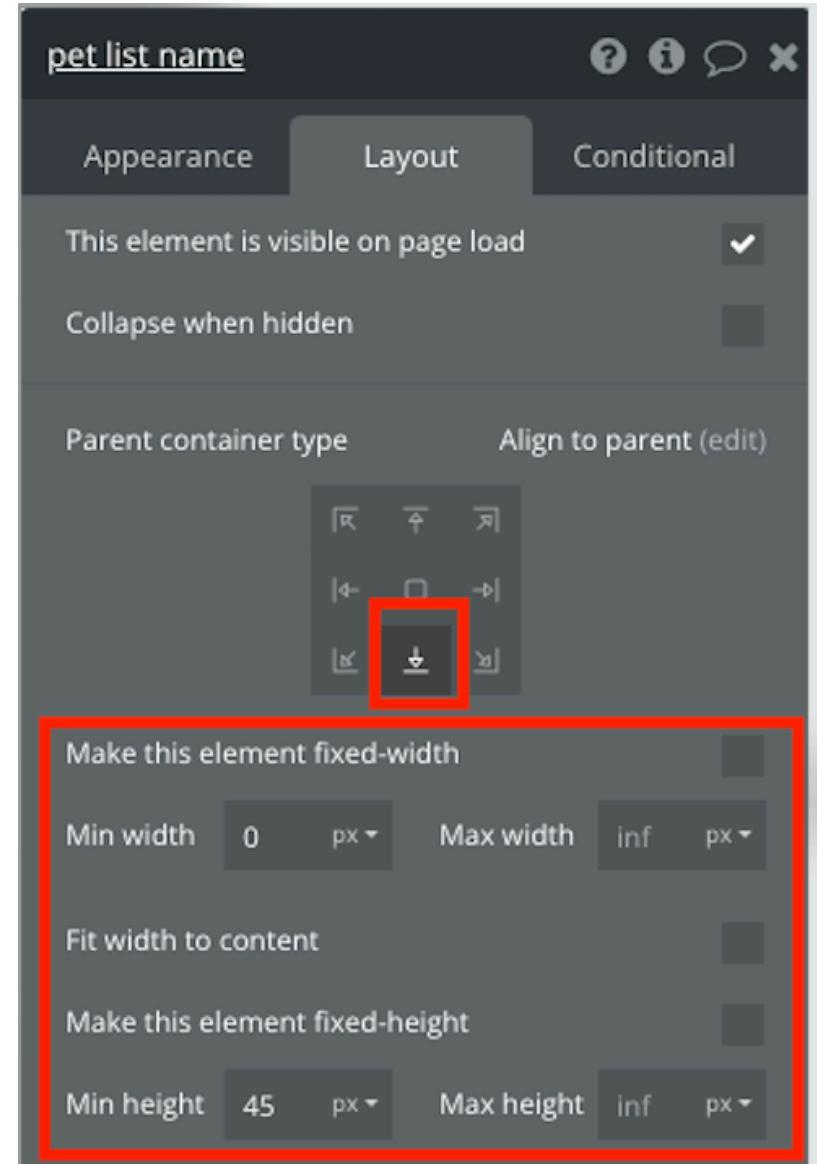
セル内の真ん中に配置し、縦横比を 4:5 に保ったままセル内いっぱいまで大きく表示する、という設定になります。





次は、Shape A の Layout に右の画像と同様の設定をいれます。

セル内の下部に配置し、高さ 45px を保ったまま左右いっぱいまで大きく表示する、という設定になります。

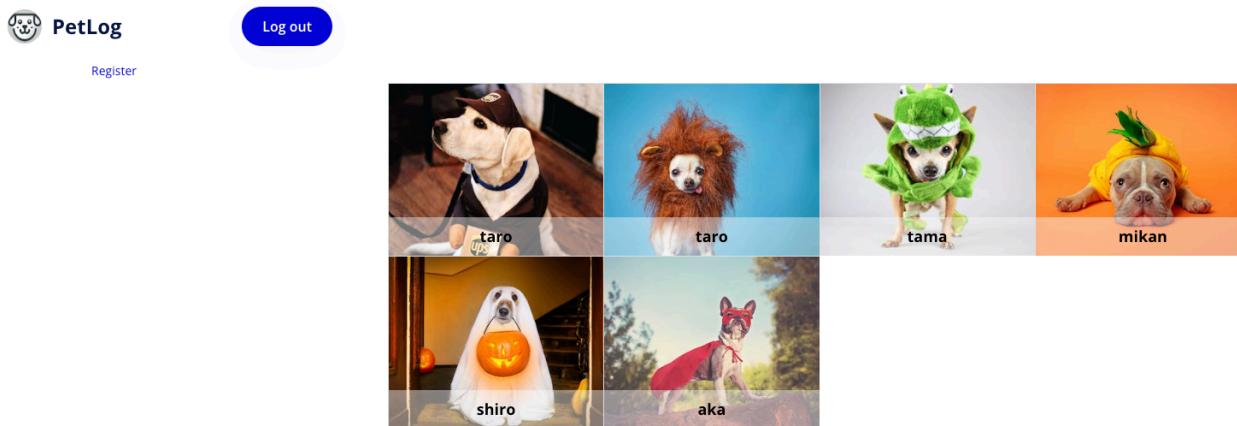


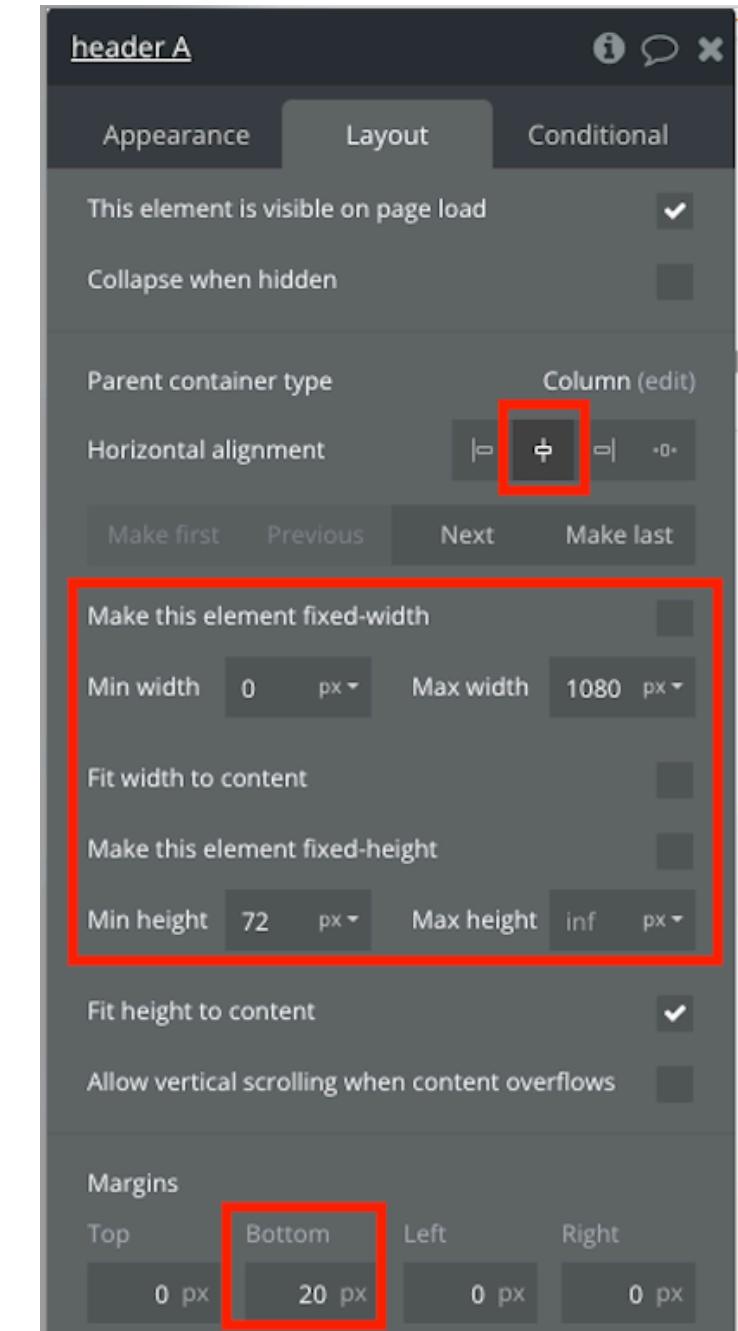
次は、`pet list name` の Layout に右の画像と同様の設定をいれます。

Shape A と同じ内容ですね。

プレビューします。

惜しいですね。あとはヘッダーの位置や余白が気になります。

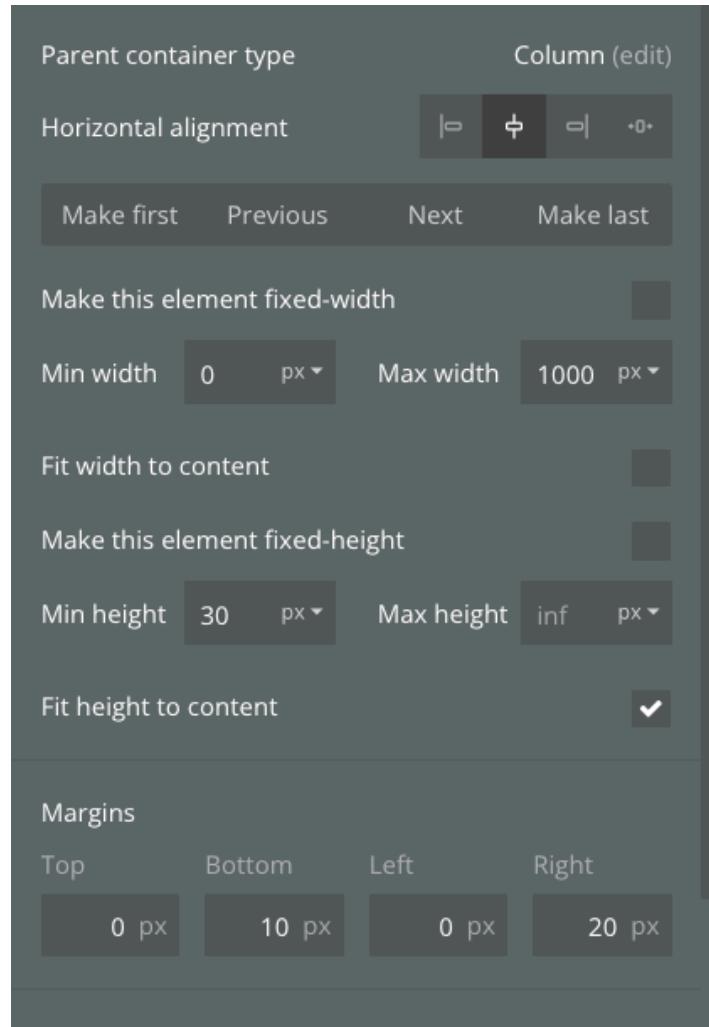




ヘッダーの設定ウィンドウを開き、
Layout に右の設定を入れます。

ページ内で列方向に並べられる際に、中央揃えで表示する。横幅は固定せずページ幅に応じて、最大 1080px まで伸びる。下側に 20px の余白をとる。という設定です。

つづいて、Register リンクの設定ウィンドウを開き、**Layout** に右の設定を入れます。

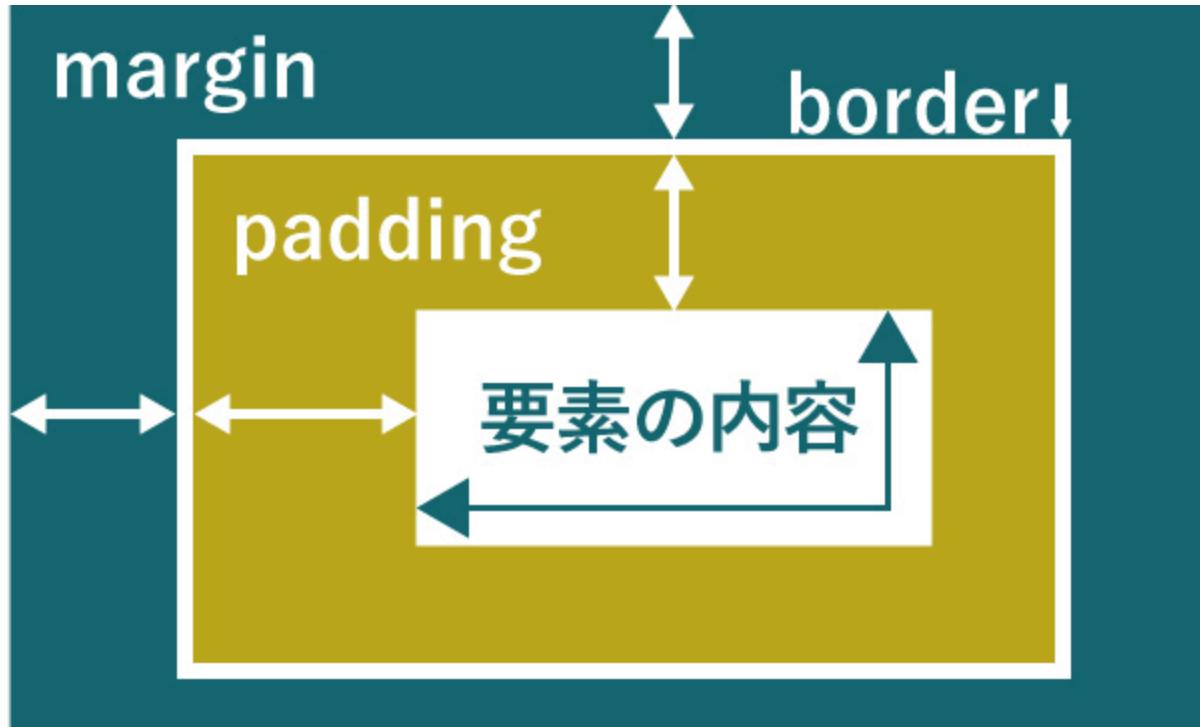


Margin と Padding について

Margin という言葉がはじめて出てきましたので、説明します。

余白を表す言葉に、Margin と Padding というものがあり、それぞれ以下を意味します。

- Margin : 要素の境界の外側の余白
- Padding : 要素の境界の内側の余白

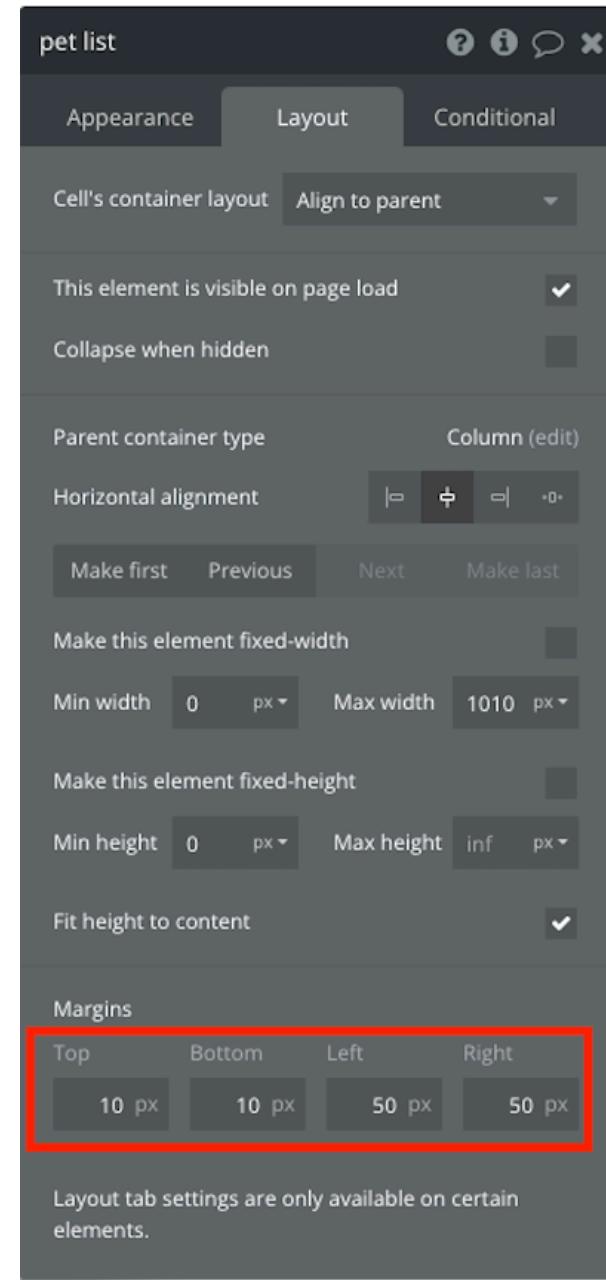


境界線の内と外とで、余白を使い分けたい時に意識してみてください。

ついでに、リピーティンググループにも余白をいれておきます。

リピーティンググループの Layout タブで以下を設定してください。

上下の余白に 10px、左右の余白に 50px をとるという設定になります。



では、プレビューしましょう。



PetLog



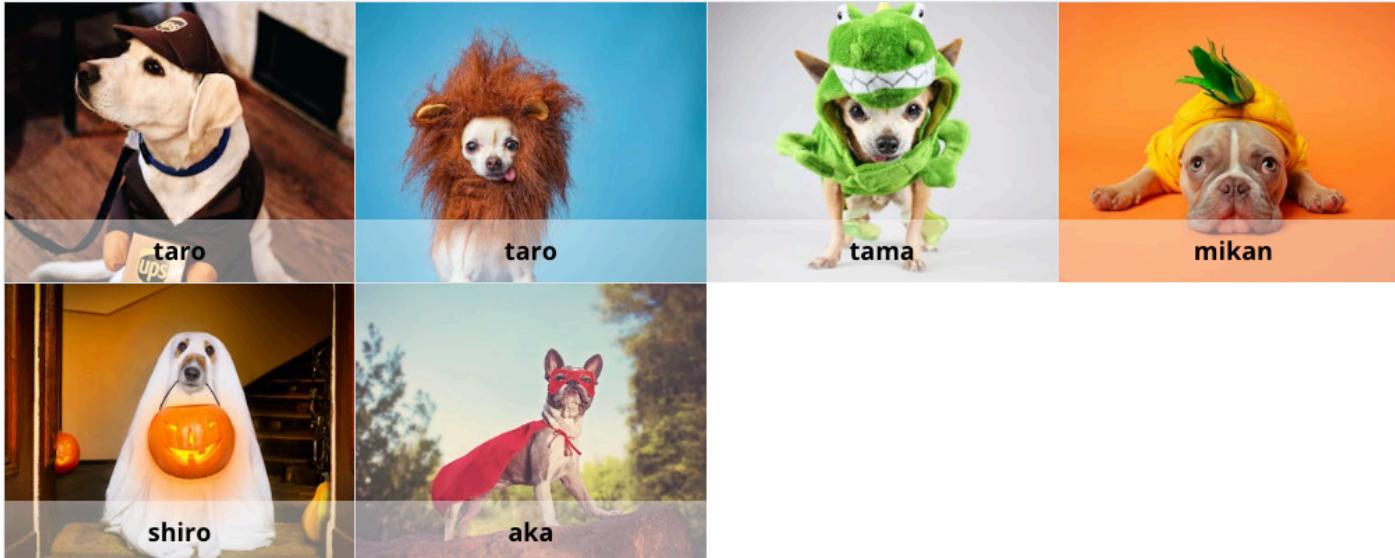
PetLog

Log out

Register

Log out

Register



やったー。

<Excercise>

ペット詳細画面にレスポンシブデザインを適用しよう

ペット詳細画面がせっかくの PC 画面の広さを活かせないようになっているので、レスポンシブ対応しましょう。

ステップ1：画面幅に応じて、伸縮させる ***難易度：中**

ヒント

- 親要素内の配置ルールを Column を利用して縦にならべる。
- 各要素を画面幅に応じて伸縮させる。

ステップ2：画面幅に応じて、段組を変える **※難易度：高**

ヒント

- ・子どもの要素を2つほどのかたまりにグルーピングする。
- ・グループを画面幅に応じて、縦にならべたり横にならべたりと、段組を切り替えるようする。（段組を切り替えたいグループを、さらにグループで囲って配置ルールをRowにする。）

よく使うルール(おさらい)

Bubble でレスポンシブデザインを実現するために、よく使うルールとして以下のようなものがあります。

1. 親要素内の配置ルール
2. 要素のサイズ決定ルール
3. 表示の有無ルール

これらを組み合わせて、レスポンシブデザインを適用しましょう。

Style を使ってみよう

Style を使ってみよう

- これまで Bubble が標準で用意してくれていたスタイルを利用してきました
- 実際のプロダクトでは、プロダクトにあったデザインコンセプトを描き適用していきます
- ここからスタイルを変更する方法を説明します

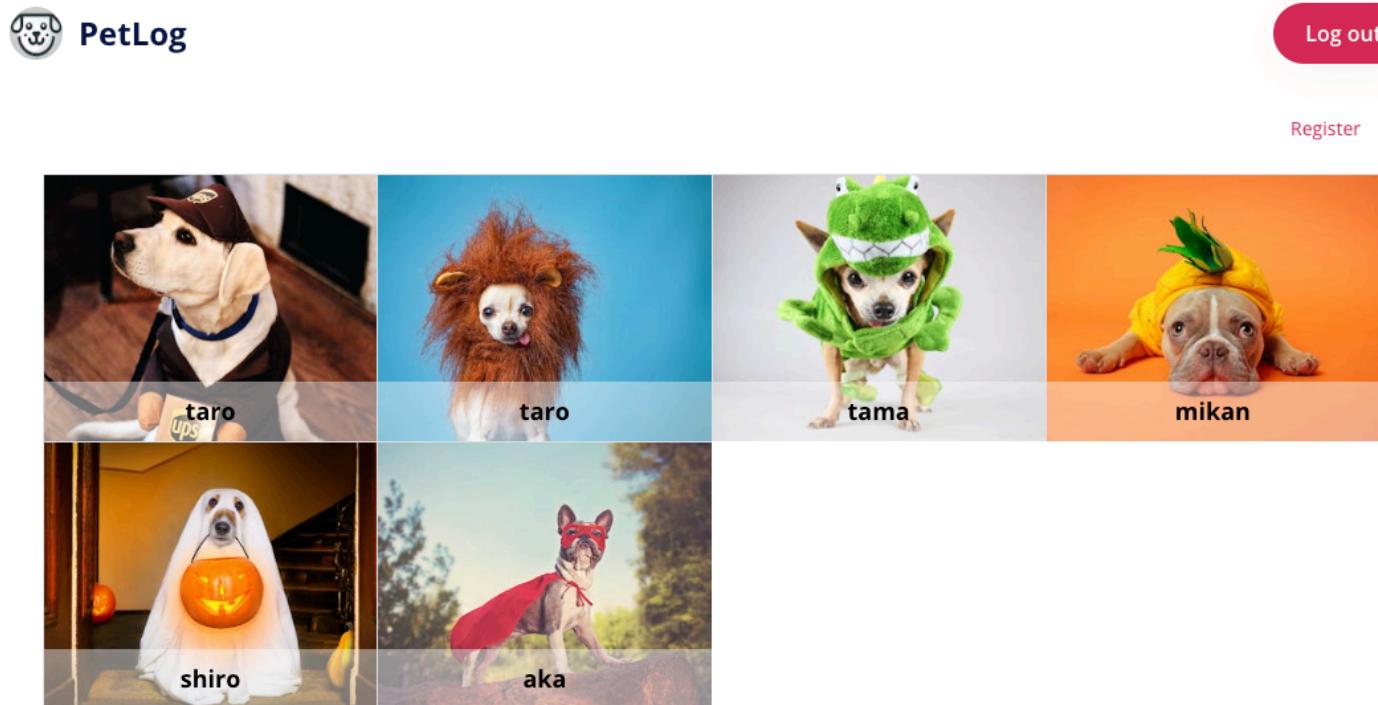
Style の適用方法には大きく 3 つあります

- 既存のスタイルを編集する
- 個別でスタイルを適用する
- 新しいスタイルを追加する

順にやっていきましょう

既存のスタイルを編集する

まずは、既存のスタイルを変更して、ボタンやリンクの色を変更したいと思います。



Style variables をつかってみよう

Bubble では、基本となる色やフォントが **Style variables** として設定されています。

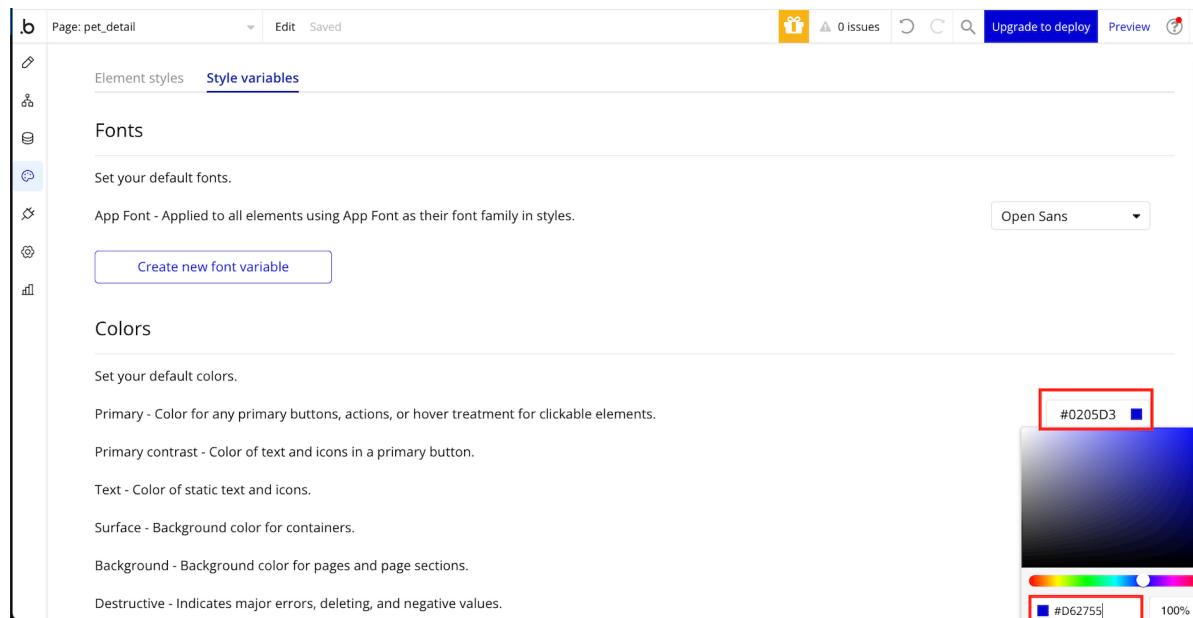
左メニューの **Styles** > 画面上部のタブで **Style variables** に移動してみてください。ここで指定された色は、スタイルの作成時や編集時に利用することができます。例えば、**Primary** という色設定は、基調色を意味し、Primary ボタンなどで利用されています。

The screenshot shows the Bubble app's styling interface. On the left is a sidebar with icons for pages, styles, fonts, colors, and more. The main area has tabs for 'Element styles' and 'Style variables', with 'Style variables' currently selected. Under 'Fonts', it says 'Set your default fonts.' and shows a dropdown menu set to 'Open Sans'. A button labeled 'Create new font variable' is visible. Under 'Colors', it says 'Set your default colors.' and lists several color variables with their hex codes and color swatches:

- Primary - Color for any primary buttons, actions, or hover treatment for clickable elements. Hex code: #0205D3 (dark blue)
- Primary contrast - Color of text and icons in a primary button. Hex code: #FFFFFF (white)
- Text - Color of static text and icons. Hex code: #091747 (dark teal)
- Surface - Background color for containers. Hex code: #FFFFFF (white)
- Background - Background color for pages and page sections. Hex code: #FFFFFF (white)

`Style variables` の `Primary` の設定を変更すると利用されているすべての箇所に変更が適用されています。

- 左メニューの `Styles` > 画面上部のタブで `Style variables` と選択
- `Primary` を変更する。(私は暗い目の赤にしたいので、`#D62755` で指定します。)



Primary ボタンのスタイルを確認すると、変更されていますね。

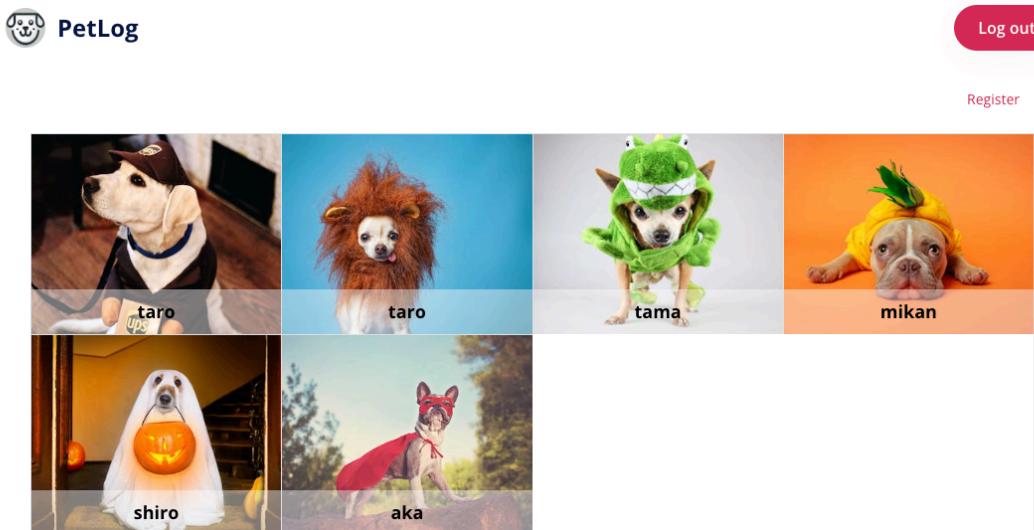
The screenshot shows a style editor interface with the following elements:

- Element styles** tab is selected.
- Style variables** tab is also present.
- Element type**: Button
- Search by name**: Primary Button
- Buttons sidebar**:
 - Button - Flat Button
 - Button - Outline Button
 - Button - Primary** (selected)
- Primary Button preview**: A red button with the text "...edit me...".
- Action buttons**:
 - Find all elements using this style
- Conditional panel**:
 - Appearance**, **Layout**, **Conditional (1)** tabs
 - OFF** state: remove condition
 - When**: This Button is hovered
 - Background color**: Primary (#D62755)
 - Select a property to change when true
 - + Define another condition

プレビュー

画面をプレビューしてみましょう。

ログイン／ログアウトボタンなど、基調となっていたボタンやリンクの色が変っています。



Style variables の使いどころ

このように、 Style variables を編集することで、標準の基調カラーを一括で変更できます。また、新たにスタイルを作成・編集する際も自分でルールを決めて Style variables を利用しておくことで、あとから一括で変更するなどメンテナンスが楽になります。

次に、個別でスタイルを指定してみましょう

ヘッダーロゴを基調色にあわせつつ、フォントももう少しかわいらしい感じにしたいと思います。

The screenshot shows the PetLog application interface. At the top left is the logo "PetLog" with a small dog icon. At the top right are "Log out" and "Register" buttons. Below the header is a grid of six dog photos:

- Top row: A white dog wearing a brown UPS delivery uniform labeled "taro".
- Top row: A small white dog with a lion mane costume labeled "taro".
- Top row: A small dog in a green dragon hooded onesie labeled "tama".
- Top row: A tan dog lying down in a yellow pineapple costume labeled "mikan".
- Bottom row: A white dog dressed as a ghost holding a glowing jack-o'-lantern labeled "shiro".
- Bottom row: A tan dog in a red superhero cape and mask labeled "aka".

b ロゴ右のメニューを開いて、header を選びます

The screenshot shows the Bubble platform's interface for managing pages and reusable elements. On the left, there's a sidebar with icons for search, edit, save, and various page types like index, pet_detail, etc. A blue-highlighted icon for a reusable element is selected, revealing a tooltip. The tooltip is titled "Reusable elements" and contains a card for a reusable element named "header". The card features a small dog icon and the text "PetLog". A red arrow points from the "header" entry in the sidebar towards this tooltip.

.b Search a page or a reusable... Edit Saved

Pages

index login

pet_detail pet_list

pet_register pet_update

pet_weight reset_pw

404

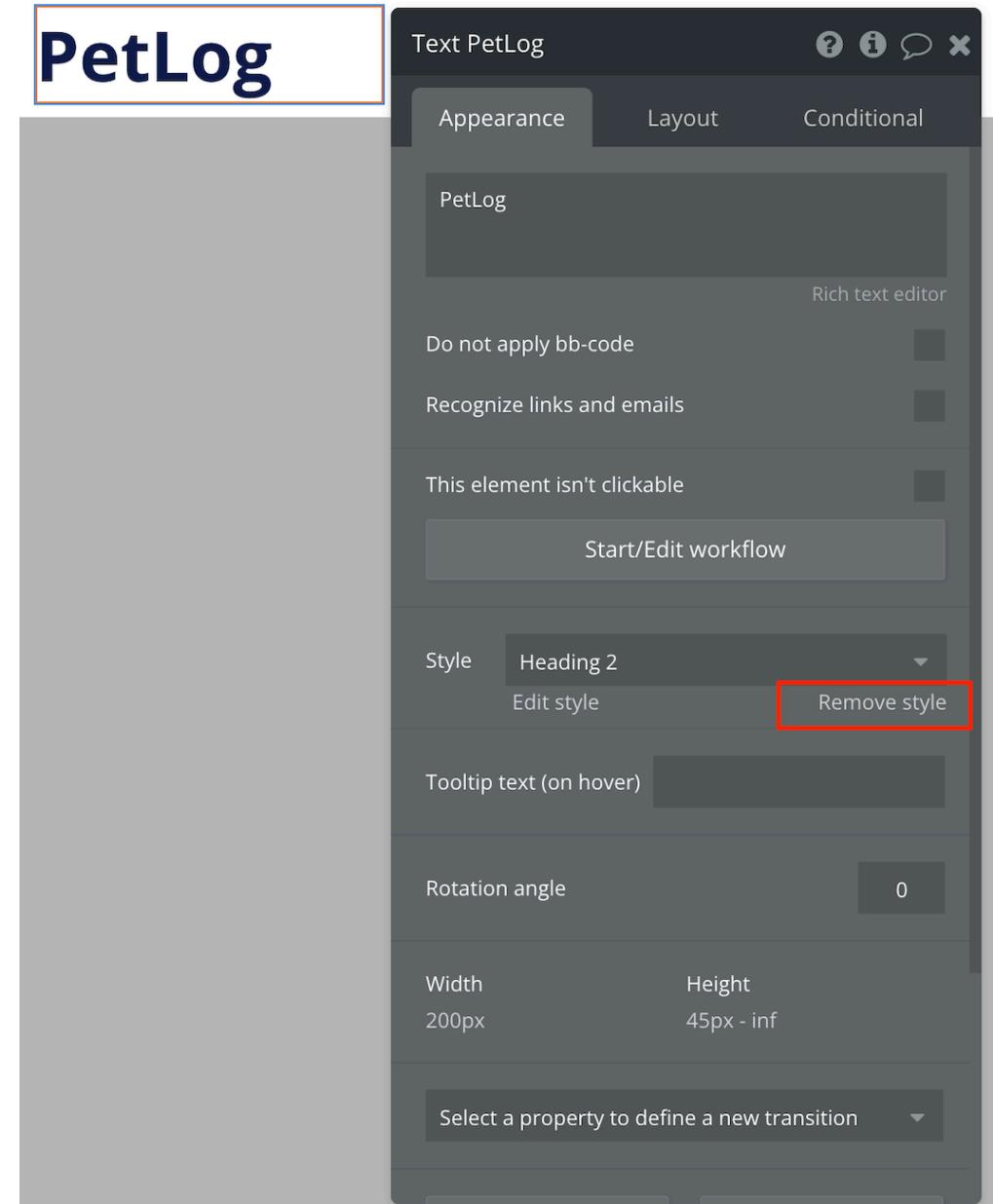
Add a new page... Add new page with AI BETA

Reusable elements

header PetLog

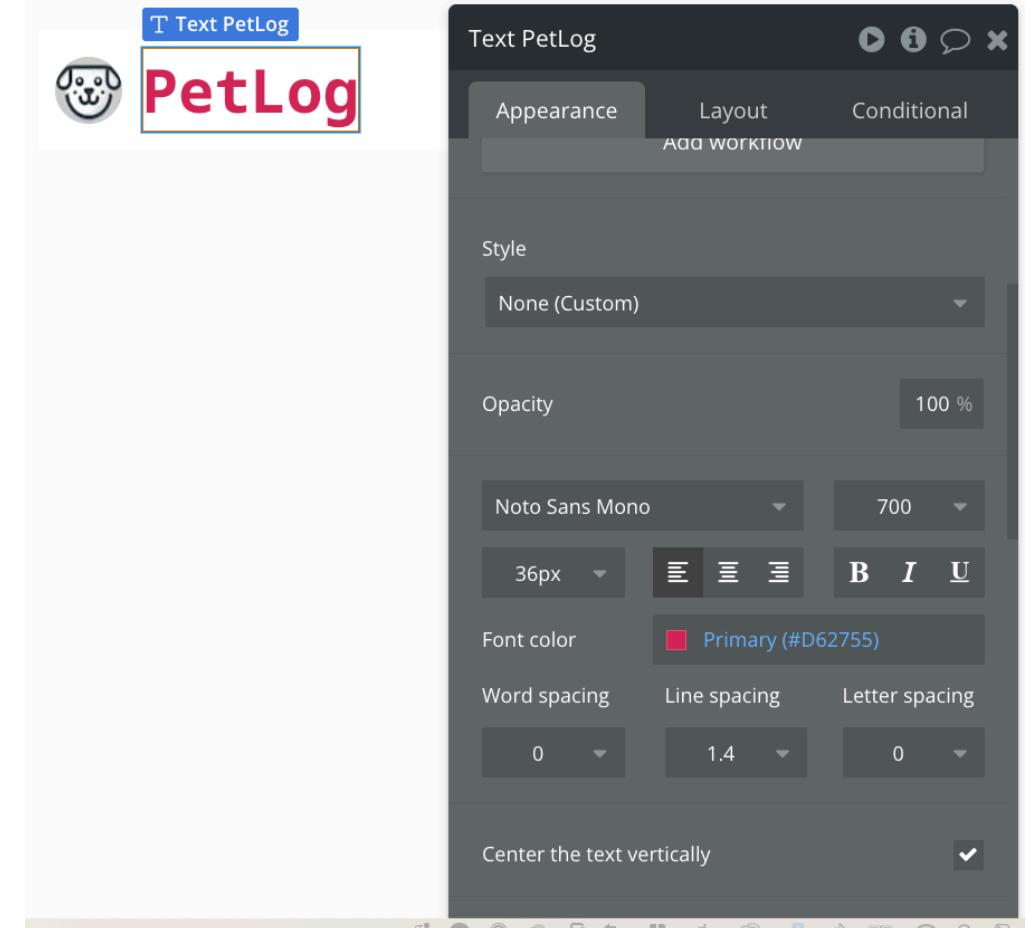
Add a new reusable element...

- ロゴをダブルクリックして、設定を開く
- 設定ウィンドウの **Style** の部分に移動する
- プルダウンの右下あたりの **Detach style** をクリック
 - 定義された Style を適用するのではなく、個別で指定できるようになります



好きなスタイルを指定しましょう

- 私はフォントカラーは、 Style variables で指定した Primary カラー
- フォントは Noto Sans Mono が気に入ったので、それを指定します。
- サイズは 32px くらいにしておきます。
- フォントを変えると、ロゴが見切れたりすることがあるので、幅は適宜調整してください



プレビューしましょう

変わりましたね。



PetLog

Log out

Register



[← Back to List](#)[View Weights](#)

Name

tama

Image



Category

dog

Birthday

2024年6月18日

Gender

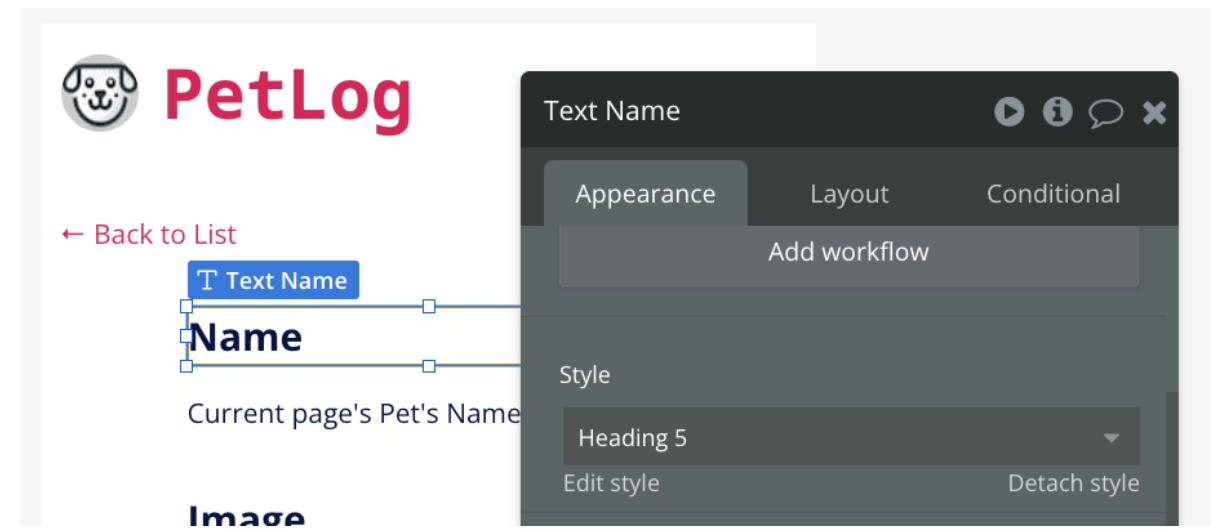
女の子

次は、新しいスタイルを追加してみましょう

ラベルが大きすぎて気になるので、ラベル用のスタイルを作っちゃおうと思います。

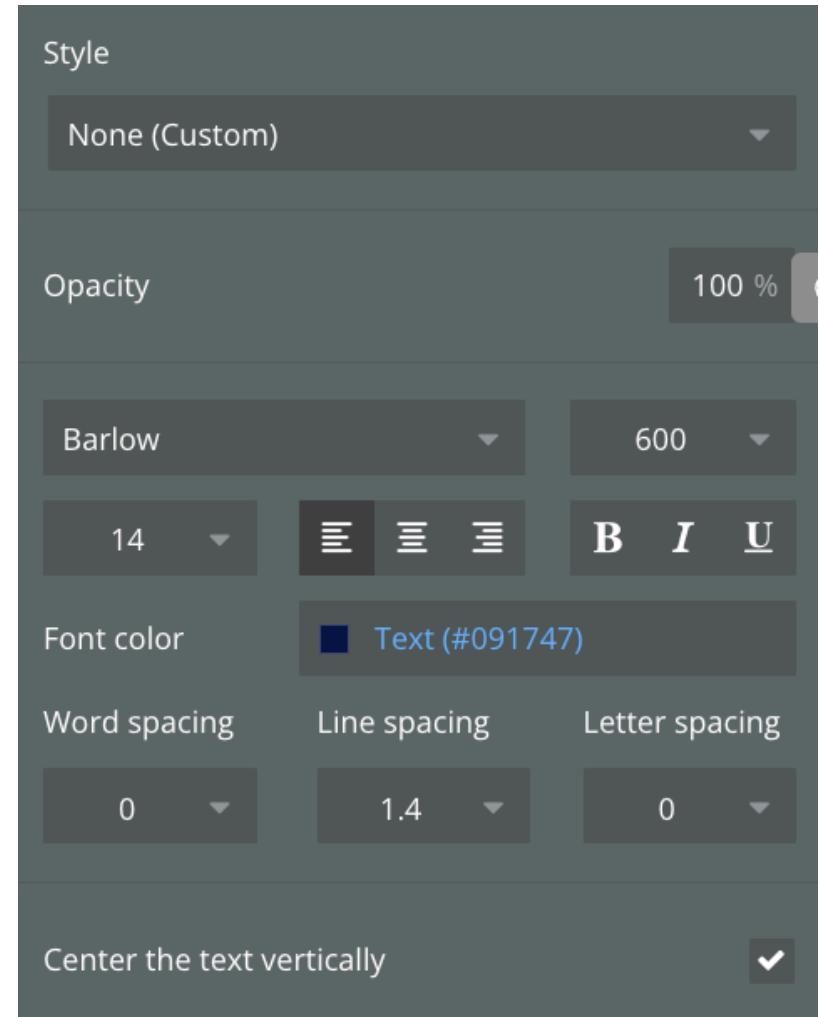
まずは個別にスタイルを指定します。

- pet_detail を開いて、 **Image** テキストをダブルクリックして、設定を開く
- **Style** のプルダウン右下の **Detach style** をクリック



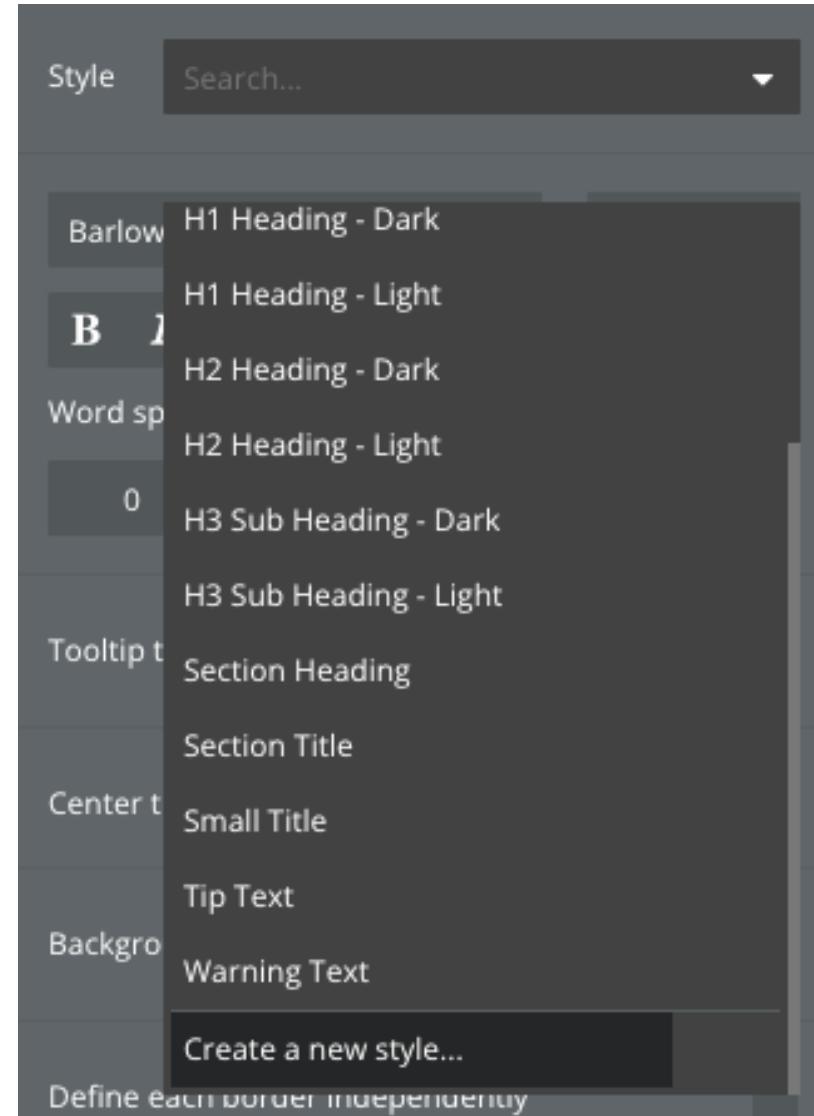
以下の設定にします

- フォントは Barlow
- フォントウェイトは 600
- フォントサイズは 14
- Center the text vertically
にチェック



続いて、指定した個別のスタイルを共通のスタイルとして定義します

- そのまま **Name** の設定の中で、**Style** のプルダウンを開く
- 一番下の **Create a new style..** をクリック



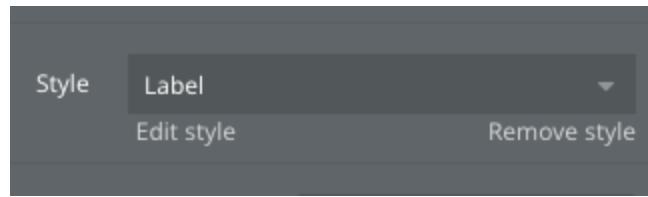
- Style 名に Label と入れる
- Element style は Text のまま、テキスト Element のスタイルであることを示しておく
- Use style of は Text Name のままで、Text Name を元としてスタイルをつくります

Create a new style

The dialog box has the following fields:

- Style name: Label
- Element style: Text
- Use style of: Text Name
- Buttons: CREATE (blue button), Cancel

スタイルに Label が指定されるようになっているはずです。



個別に指定したスタイルを共通のスタイルとして定義するのではなく、
先にスタイルを定義する方法もありますが、個別でスタイルを
指定したものを共通のスタイルとして設定する方が
デザインビューでイメージを確認しながらつくれるのでやりやすいです。

では、定義したスタイルを他のラベルにも適用していきましょう。

shift を押しながら要素を選択すると複数選択がでく、複数要素を一括で操作がでくます。

- pet_detail: Image, Category, Birthday, Gender
- pet_register: Name, Image, Category, Birthday, Gender
- pet_weight_register: Weight

ちょっと多くて手間ですね。

最初からスタイルを分けていたら、一力所のスタイルを変更するだけで済んだので、意味合
いが異なる画面要素がでてきたら、スタイル定義しておくということを意識しておくとよい
でくしょう。



PetLog

Log out

← Back to List

[View Weights](#)

Name

tama

Image



Category

dog

Birthday

2024年6月18日

Gender

女の子

プレビューしましよう

Style については以上です。

ロジックを作りこむ

ロジックを作りこむ

アプリケーションには様々なところにロジックが埋め込まれています。

- 画面操作に対するフィードバックを返す
- データを抽出、加工する
- 画面を権限によって切り替える など様々です

Bubble でも様々なところにロジックを埋め込めるので一緒にやっていきましょう。

画面操作に対するフィードバックを返す

画面操作に対するフィードバックを返す

Bubble では画面要素に対してロジックを埋め込みます。

画面操作に対するフィードバックを作り混むのに使えます。

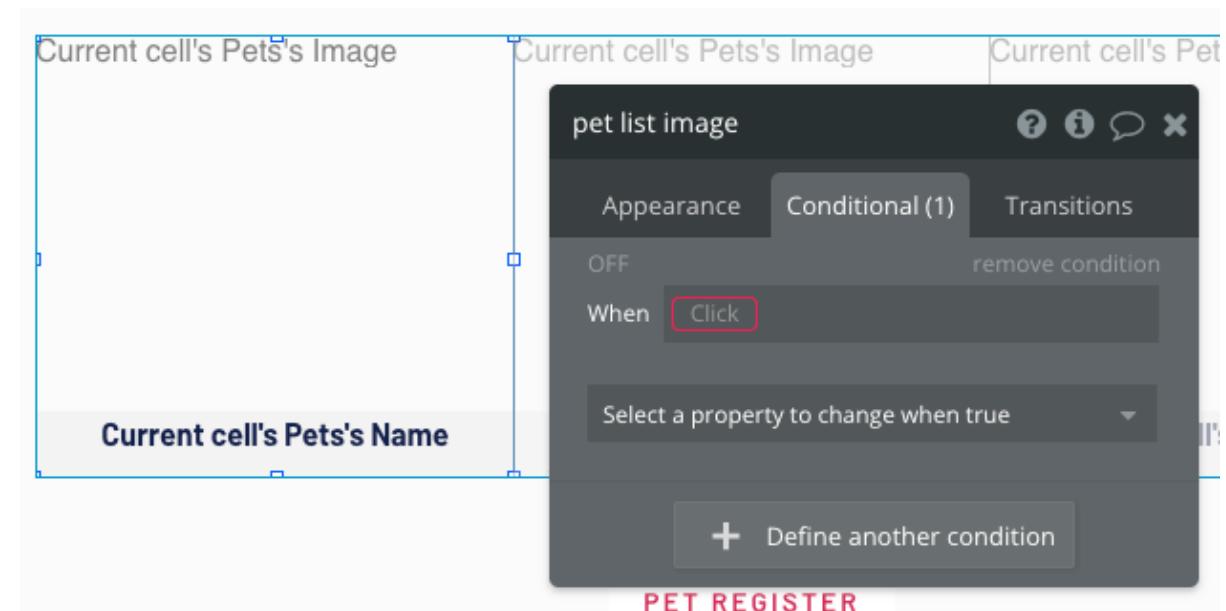
ペットリストにホバーしたら、赤枠が付くような動きをつけてみましょう

The screenshot shows a user interface for a pet log application. At the top left is a logo with a dog icon and the text "PetLog". At the top right are "Log out" and "Register" buttons. Below the header is a grid of six images of dogs. The first two images in the top row are highlighted with a red border, indicating they are being hovered over. Each image has a caption below it: "taro" for the first, "taro" for the second, "tama" for the third, "mikan" for the fourth, "shiro" for the fifth, and "aka" for the sixth. The background of the grid cells is light gray.

画像 Element にロジックを埋め込んで
いきます

- pet_list を開く
- ペット画像の画像 Element をクリックして、設定を開く
- タブから Conditional を指定する
- Define another condition ボタンをクリックする

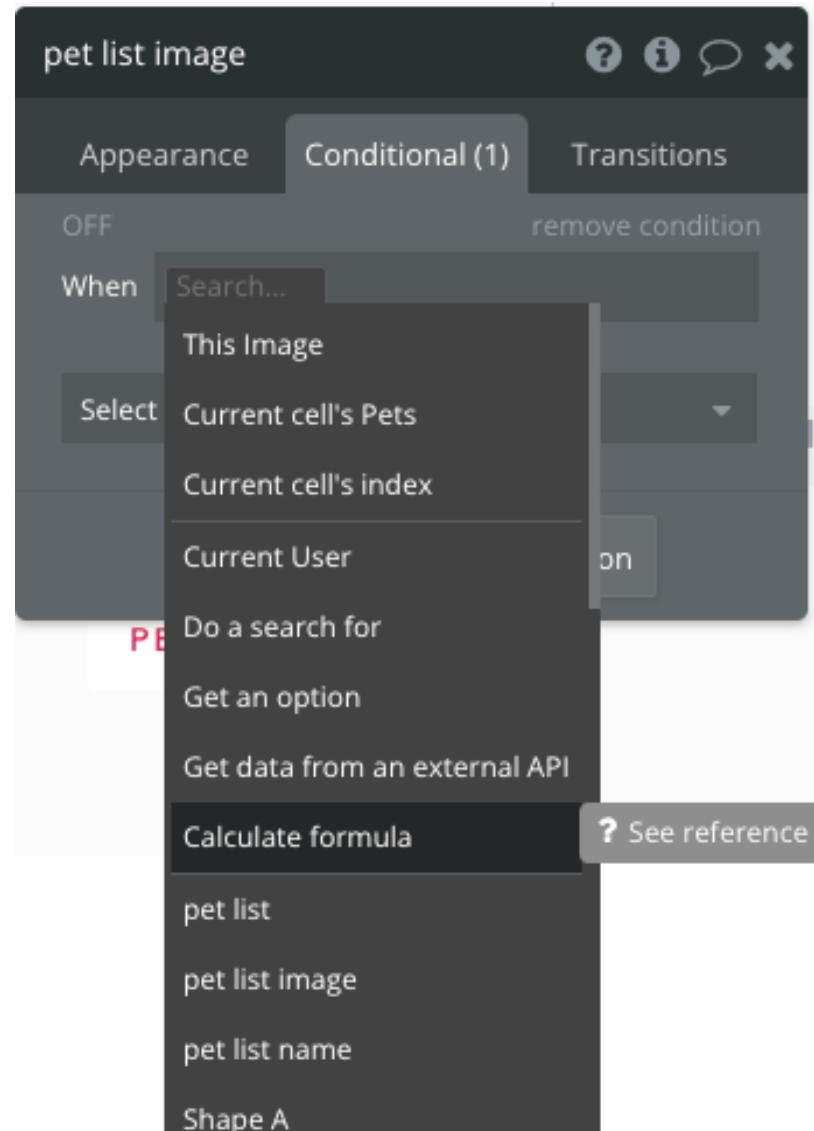
ここで条件と条件に合致した場合にプロパティをどう変更するかを定義できます。



まずは何に関する条件が指定できるか見てみましょう。

- 該当の画像 Element やその親要素や画面内の他の要素
- ログインユーザー
- 新規でのデータ検索
- 現在、日付や現在位置、ページ幅、スクロール位置など現在の状態

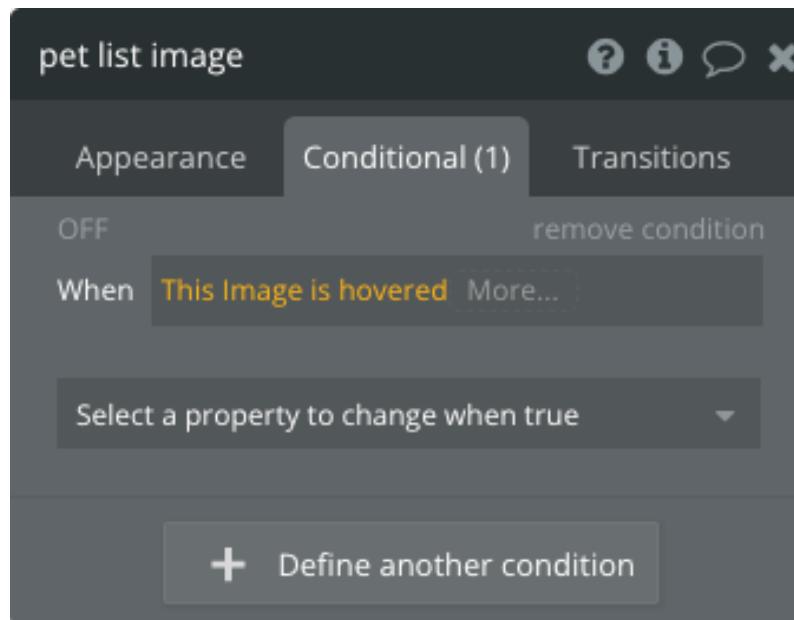
という具合に、なんかいろいろな条件が指定できそうなののがわかると思います。



今回は単純に該当の画像 `This Image` を選択しましょう。

すると、次は画像の状態がならびます。ここもいろいろありますが、今回は `is hovered` を選択してください。

これで、該当の画像がホバーされたらという条件になります。



次に条件に合致した場合に、どのプロパティをどう変更するかという指定をします。

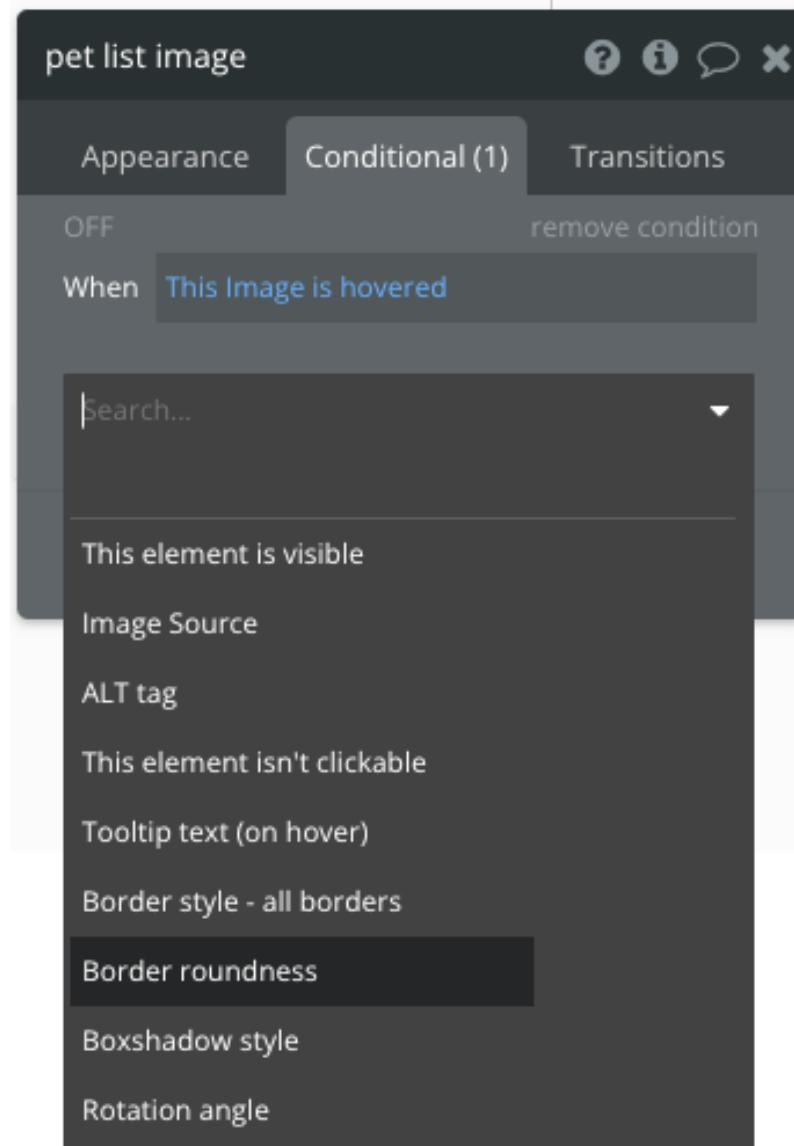
Select a property to change

when true をクリックして、中を眺めてみましょう。

- 画像のソース、alt 属性
- クリックできるか、ボーダーなど

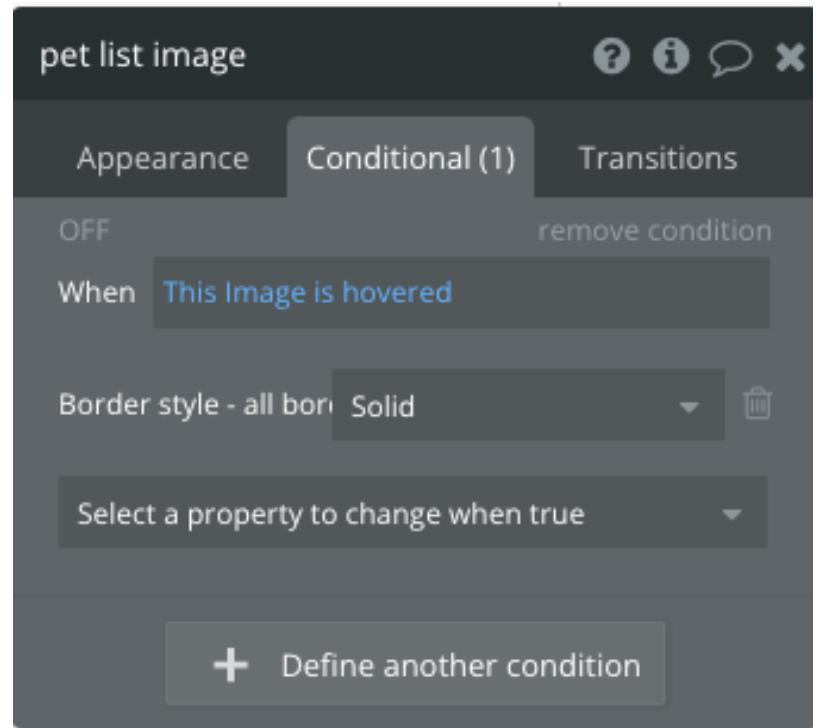
いろいろ変更出来そうだというのがわかります。

ここに並ぶものは Element の種類によって異なります。



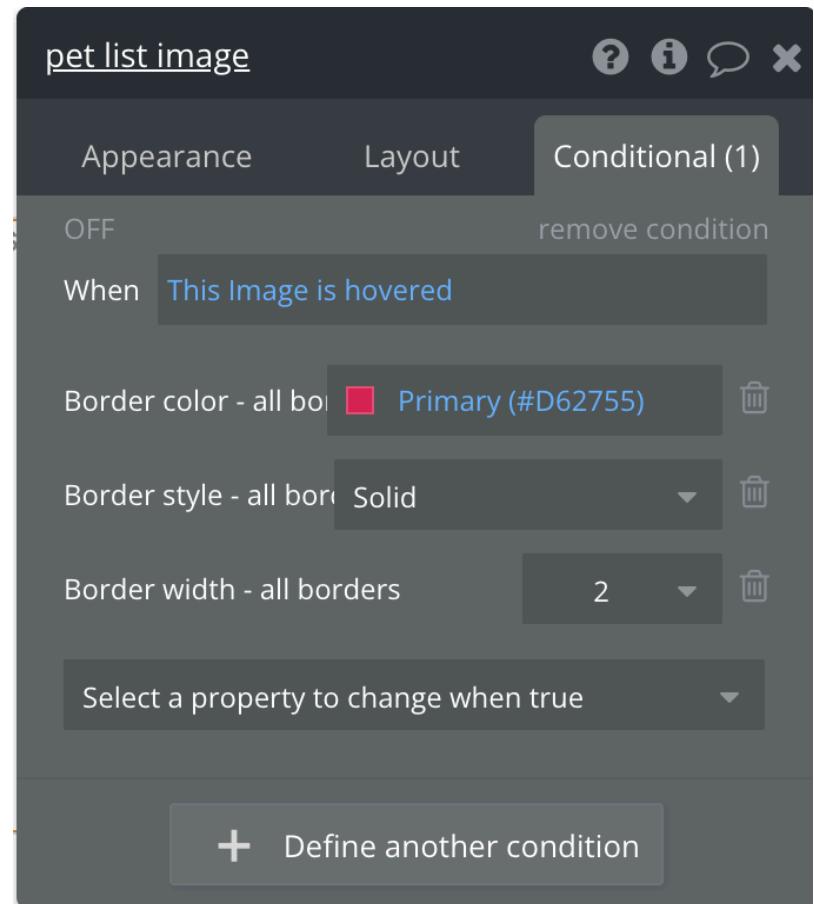
今回はホバーされたら、赤の枠線が付く
ようにします

- Border style - all borders をクリックする
- None となっている箇所を Solid に変更する
- 枠線が なし と指定されているのを 実線を表示 に変更しているという意味になります。



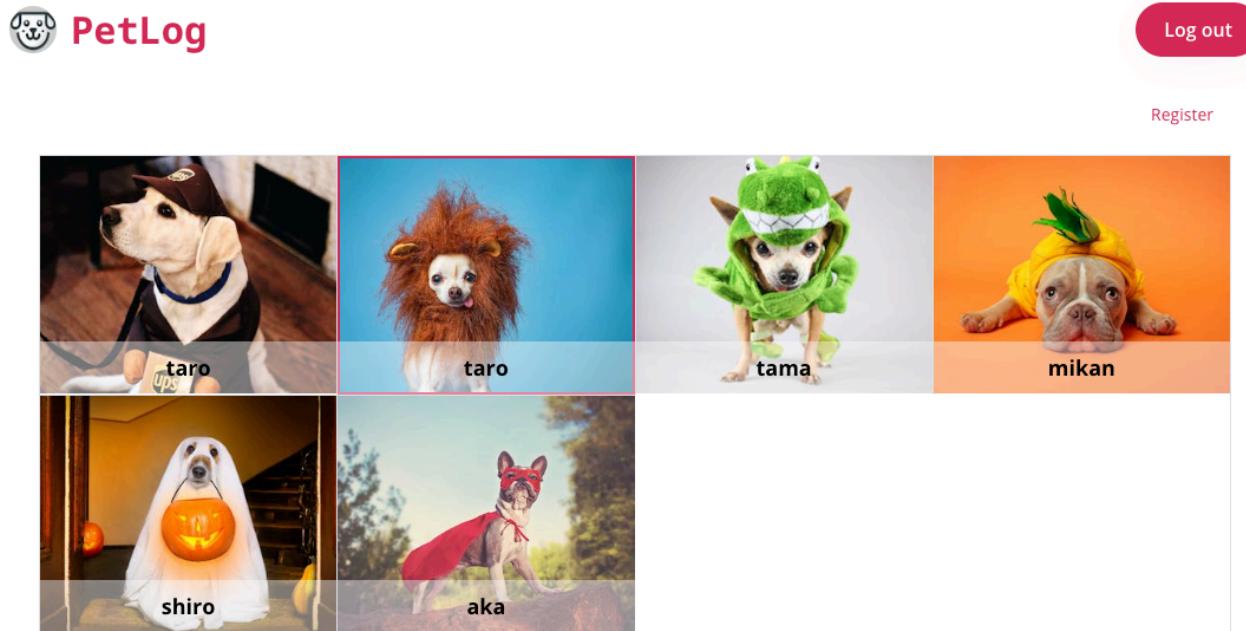
- Select a property to change when true を選ぶ
- Border color – all borders をクリックする
- 色を指定できるようになるので、定義してある Primary を選ぶ
- 同様にして次は、Border width – all borders を指定し、2 を設定する

以上で設定完了です。



プレビューしましょう

ホバーしたら、赤枠がでるようになりました。



こうやって、ユーザー操作に対してわかりやすいフィードバックを返したり、画面の装飾を状態によって切り替えたりといったロジックを埋め込んで、プロダクトを作りこんでいくことができます。

データを抽出、加工する

データを抽出、加工する

特定のデータだけを抽出したり、取得したデータを加工や計算して出したりすることができます。

ペットのイニシャル、年齢、最新の体重を表示するようにします。

Image



Name (Initial)

pochi(P)

Birthday

2020年8月18日

Age (as Dog/Cat)

2(24)

Gender

男の子

Latest Weight

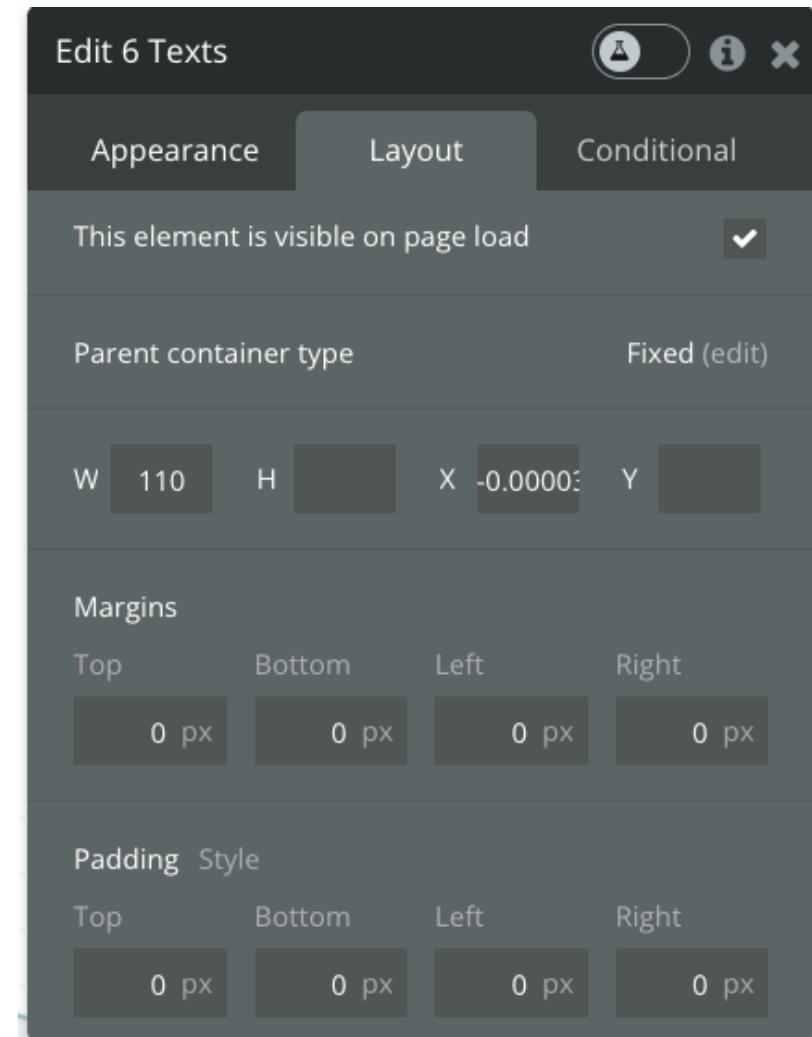
9kg



事前準備

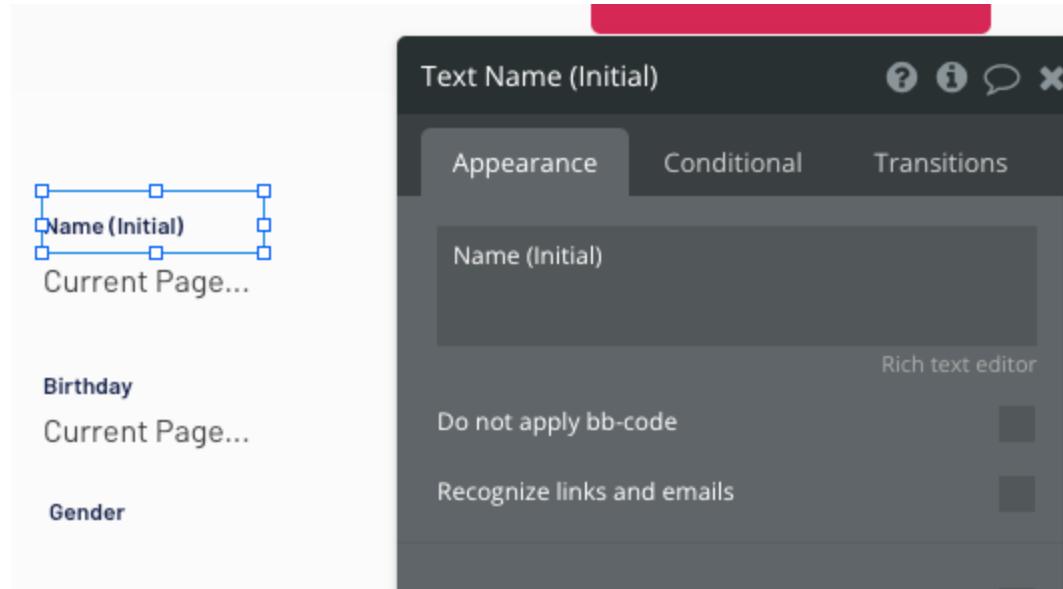
これから要素を追加していくので、
Name、Birthday、Gender の幅を縮め
ておきましょう。

- pet_detail で、Name をダブルクリックして設定を開く
- Shift を押しながら、Name のラベルから Gender のテキスト要素まで追加で選ぶ
- Layout タブを洗濯し Width と表示されている部分をクリックする
- W(幅)に 120 を指定する



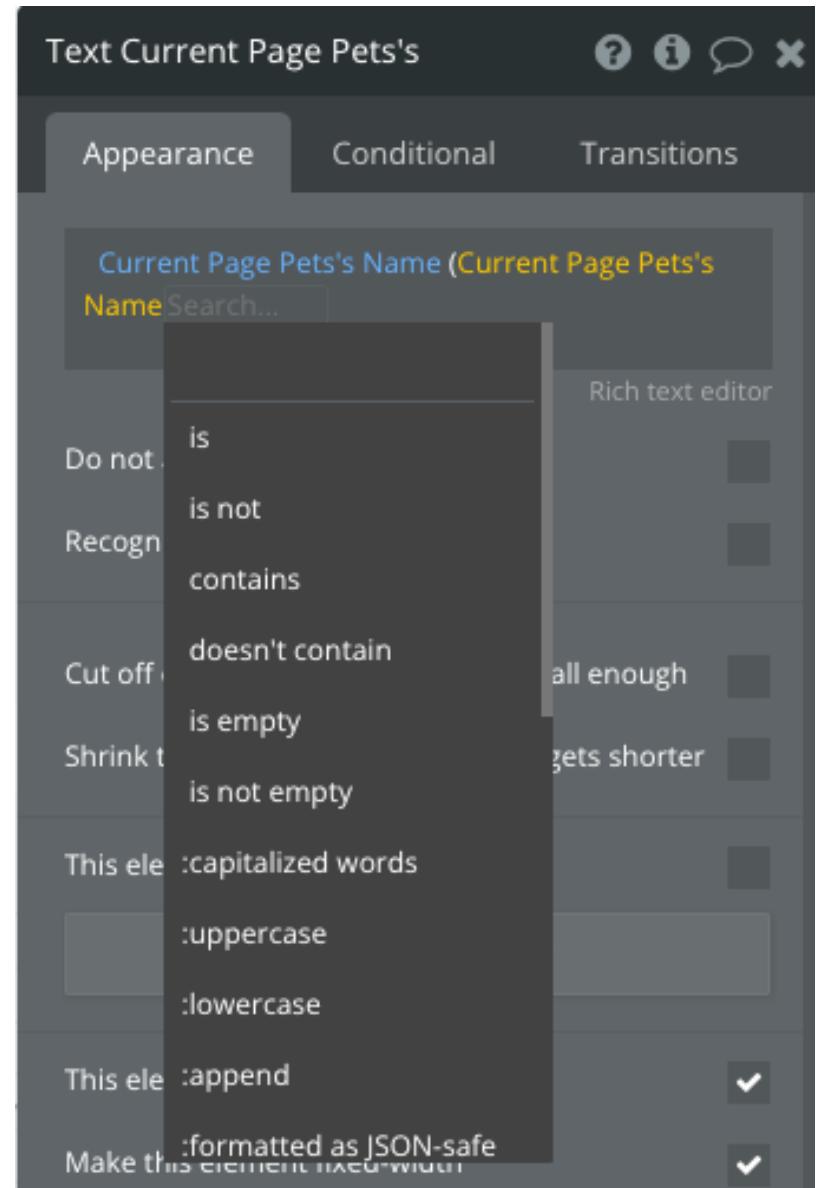
まずはイニシャルを表示します

- Name のラベルの中身を、イニシャルを含むというのがわかるように **Name (Initial)** に変更する

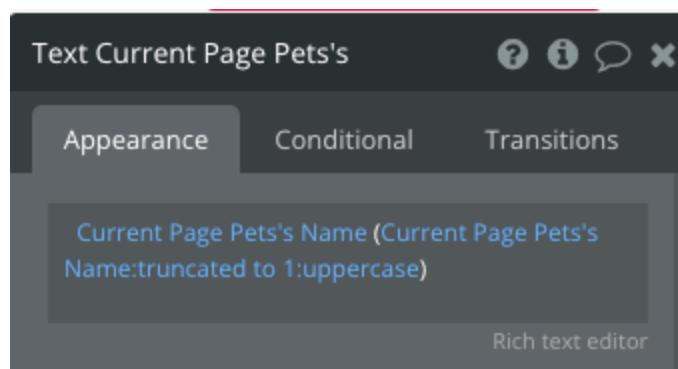


- Name の内容を出しているテキストを選択する
- テキスト入力欄の文字がない部分をクリックして、フォーカスをあてる
- (と入力する
- Insert dynamic data を選択する
- Current Page Pets > 's Name を選択する
- Search... となって続きが選択できるはずです。

ここで様々な加工方法が選択できます。
眺めて見ましょう。



- truncated to を選択する
 - これは指定された文字数までを切り取るという意味になります
- 1 と入力して、Enter キーで確定する
- また Search... と出る
- :uppercase を選択する
 - これは大文字に変換するという意味になります
 - (日本語名をつけている方にとっては意味がないですが)
- テキスト入力欄の文字がない部分をクリックして、) を入力する



プレビューしましょう

... - - - - -

Name (Initial)

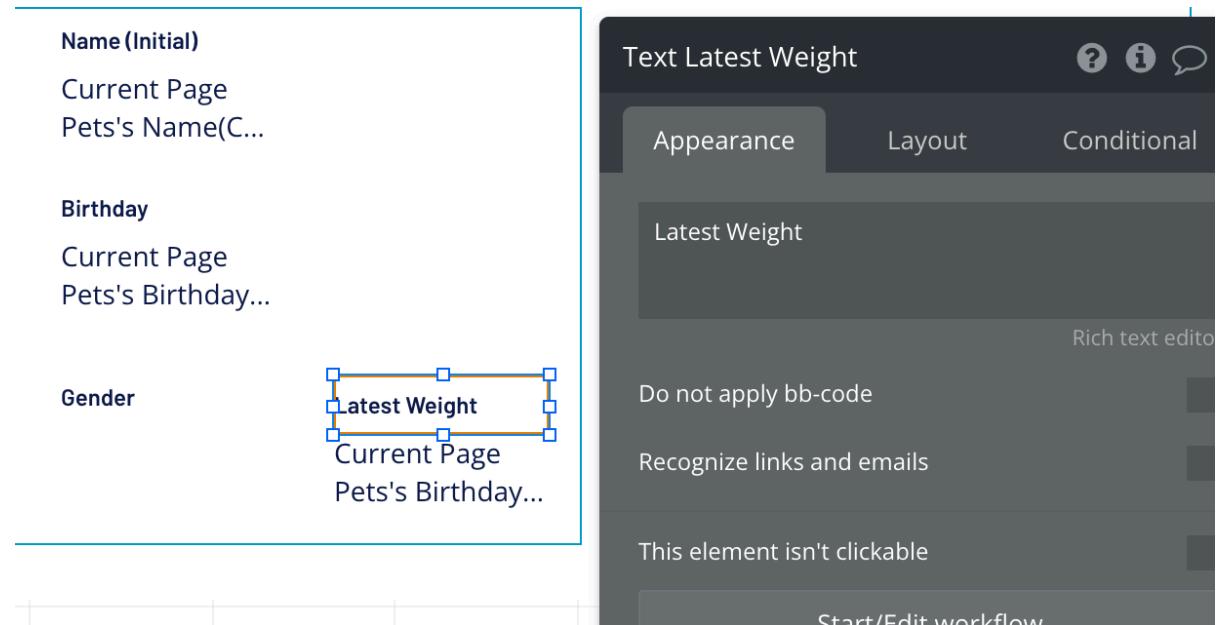
taro (T)

Image

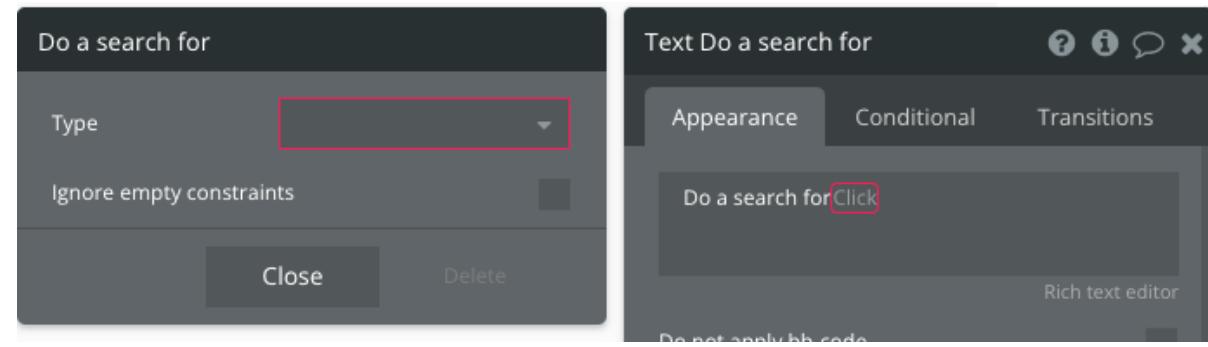


最新の体重を表示します

- Birthday のラベルとテキストをコピー&ペーストして配置しましょう
- ラベルを Latest Weight に変更しましょう



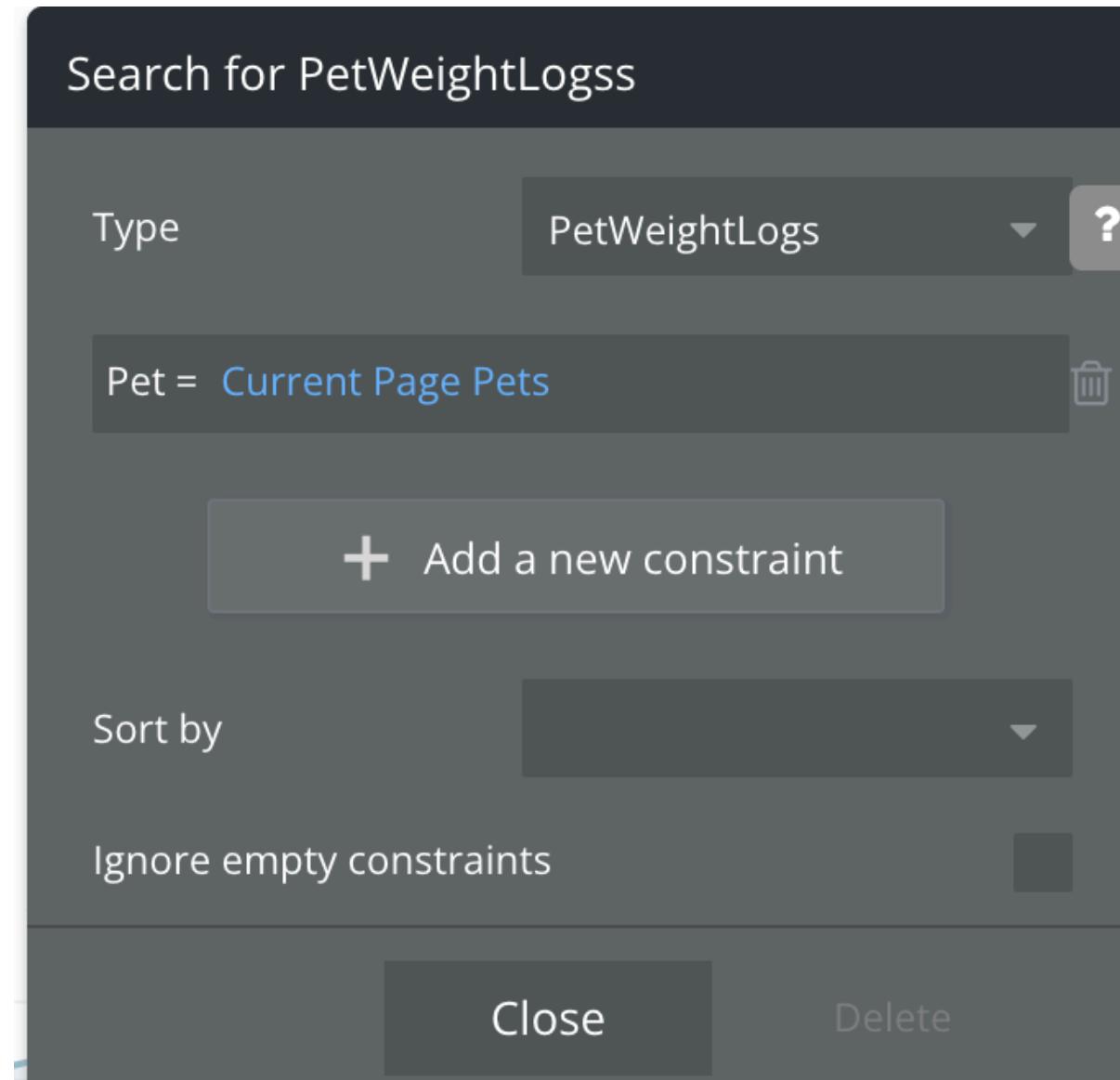
- Latest Wight の中身を出すテキストの設定を開いて、テキスト入力エリアを空にします
- フォーカスをあてて、 `insert dynamic data` をクリックします
- `Do a search for` をクリックします
 - データを検索するという意味ですね



現在ページで表示しているペットの体重を取得するという指定をします

- Type に PetWeightLogs を指定する
- Add a new constraint ボタンをクリックする
- 条件入力欄が現れるのでクリックして、Pet 、 = 、 Current Page Pets の順に指定する

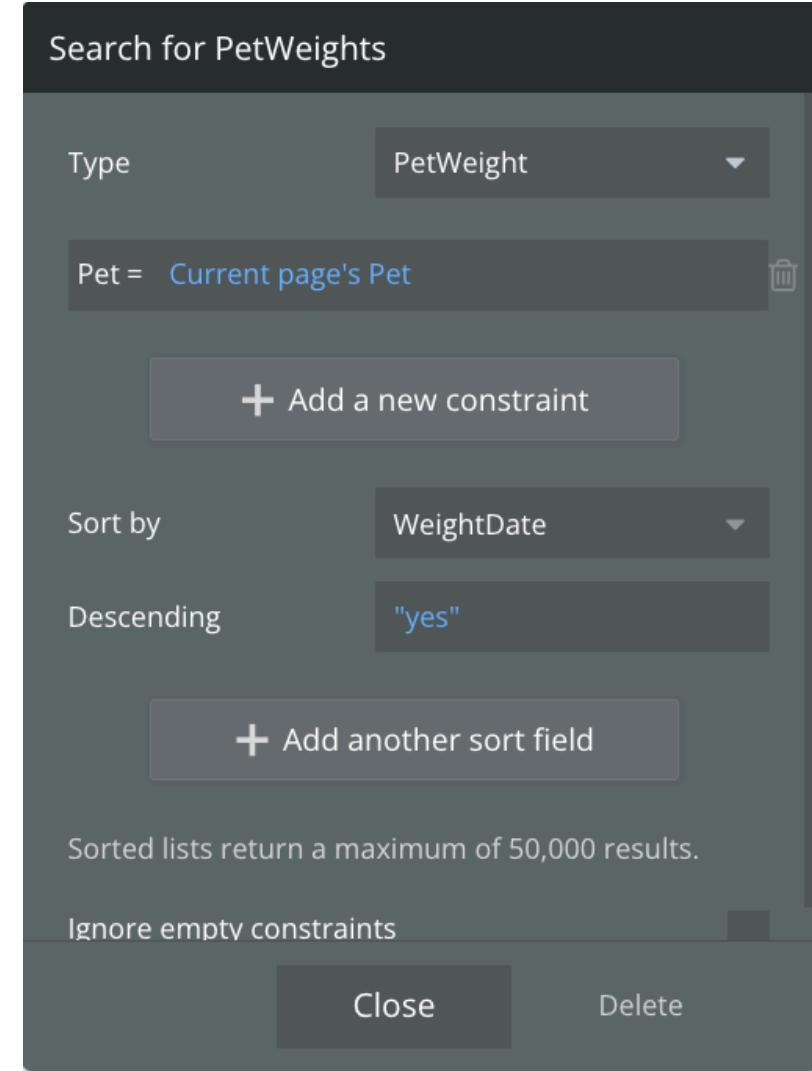
様々な条件で取得ができるので、どんな条件があるのか眺めておきましょう



作成日の降順、すなわち新しく作られた
もの順に並べるという指定をします

- Sort by に WeightDate を指定する
- Descending に yes を指定する
- Close する

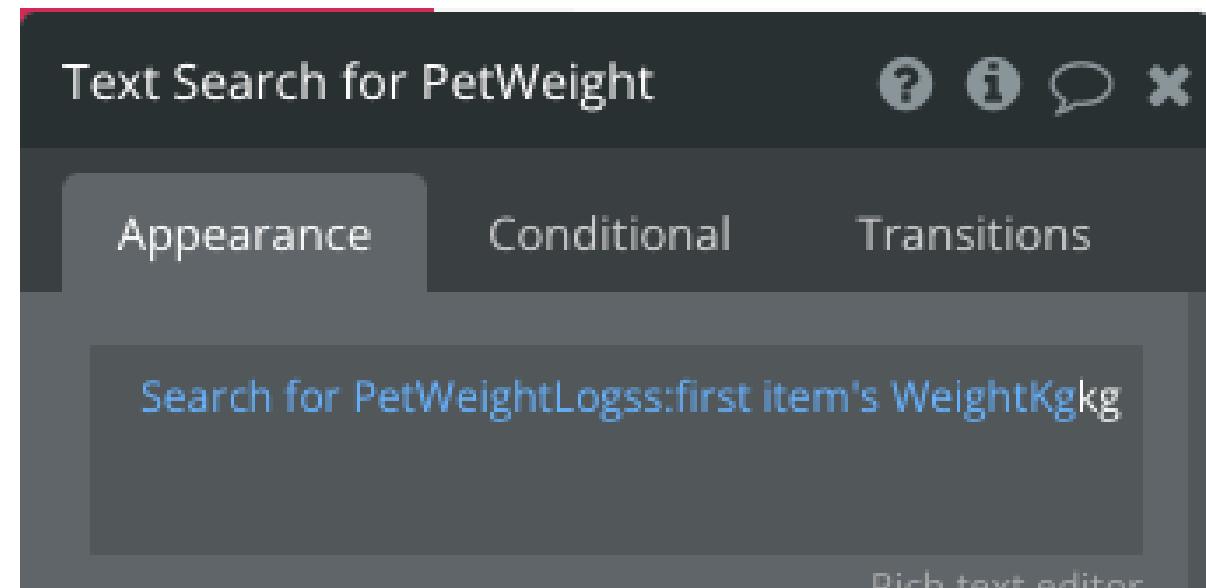
並び順の指定は忘れがちですが、重要な
ことがあります。



最新の1件の体重を表示します

- テキスト入力欄の `More` をクリックして、中身を眺めましょう
- 最初の1件目を取得したいので、`:first item` を指定する
- 続いて、`'s WeightKg` を指定する
- 空白部分をクリックして、`kg` と入力する

以上で、最新の体重の設定は完了です。
データ抽出、リスト加工の方法として覚えて起きましょう。



Image



プレビューしましょう。

Category

dog

Birthday

2024/10/23

Gender

女の子

Latest Weight

10.5kg

< Advanced >

ちょっと横道にそれで、

数値の More、日付の More を眺めてみましょう。

Bubble では数値や日付についても、様々な加工・計算方法が提供されています。

< Advanced >

年齢を計算する

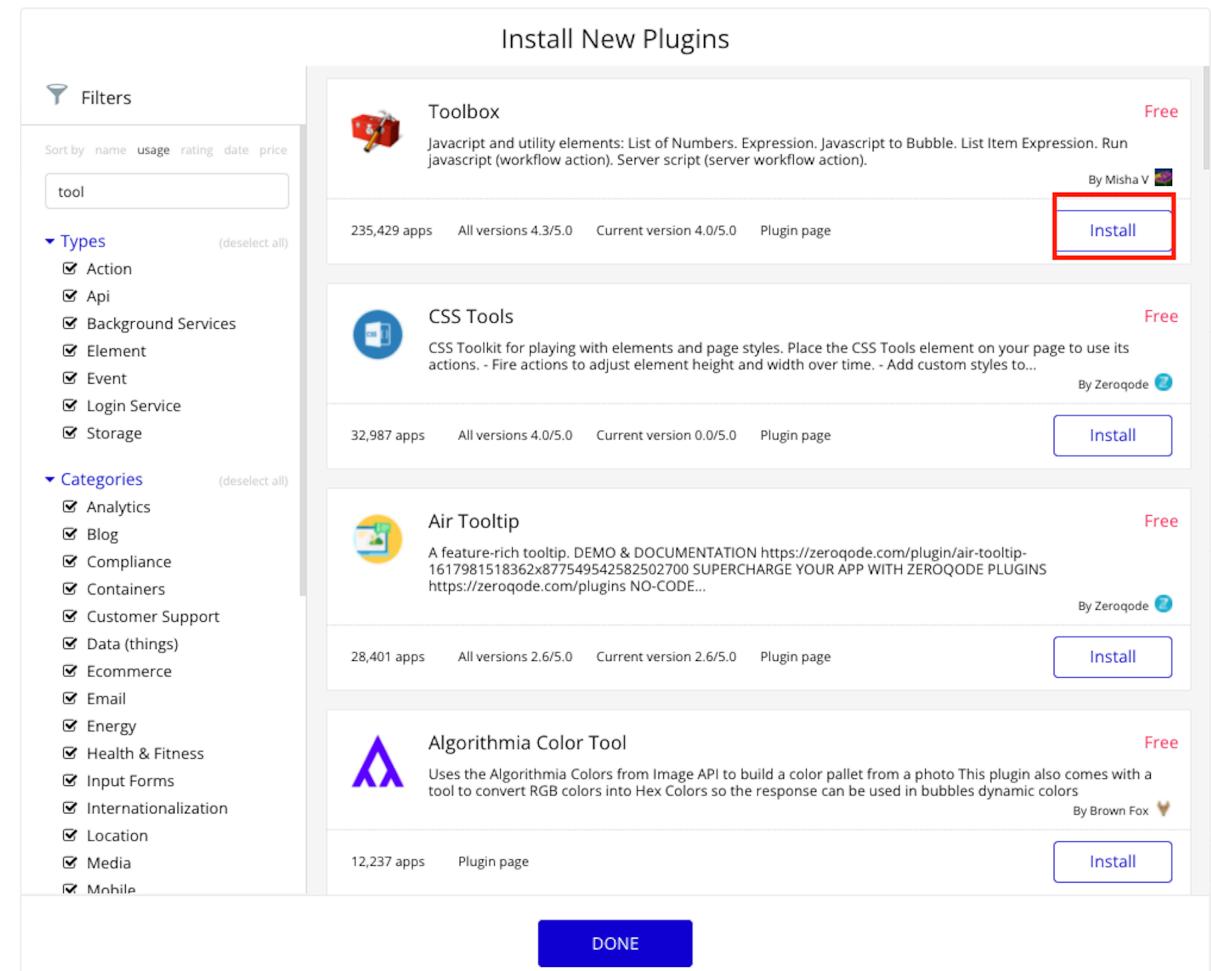
次は年齢を出します。先ほど見たとおり、数値や日付の加工・計算はできるのですが、ちょっと年齢計算は難しそうなので、直接コードを埋め込んで実現しようと思います。

Bubble では plugin を導入することで、javascript というプログラミング言語を使った簡易的な処理を動作させることができます。

< Advanced >

javascript のコードを埋め込むために
は、 Toolbox というプラグインを利用
します。
インストールしましょう。

- 左メニューの中の Plugins を指定
- 検索用のテキストボックスに
tool と入力（検索には少し時間
かかる）
- 検索結果の一番上にでてくる
Toolbox の Install ボタンを押
す



< Advanced >

Toolbox でのコードの埋め込み方は大きく 2 つあります、今回は以下の 2 通りを紹介します

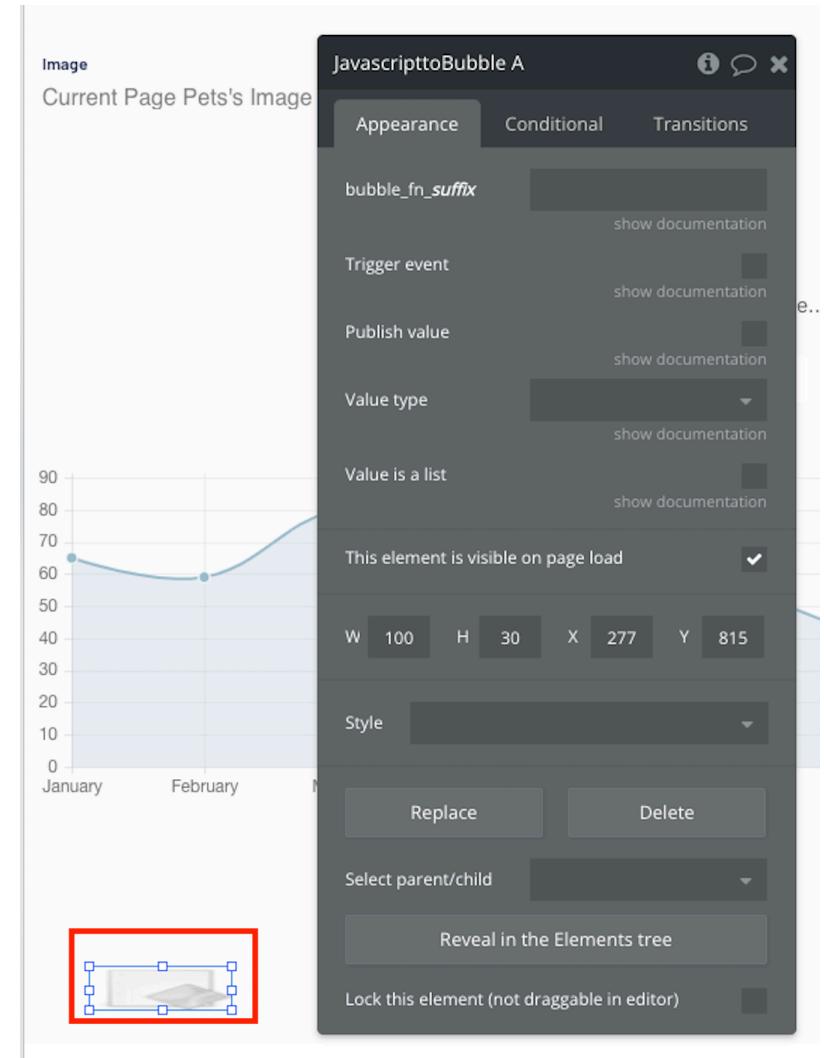
- Workflow 上の `Run javascript` で実行／Design 上の `Javascript to Bubble` で受取
 - 複数行にわたるような複雑な処理を使う
- Design 上の `Expression` で実行兼受け取り
 - 単発で終わるような処理を使う

< Advanced >

では、年齢を計算します。

Run javascript / Javascript to Bubbleで行います。まずは、pet_detail画面に、Javascript to Bubbleを仕込んでおきます。

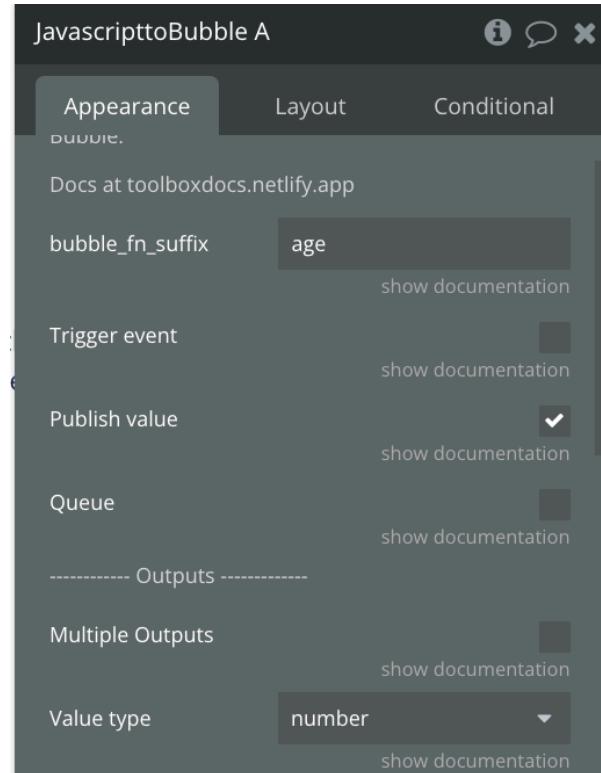
- 左メニューの Visual elements から javascript to Bubble を選択する
- 画面の最下部など邪魔にならない場所におく
- javascript の結果を受け取るためのものなので、プレビューなど実行時には表示されません



< Advanced >

- `bubble_fn_suffix` に `age` と指定する
- `Publish value` にチェックする
- `Value type` に `number` と指定する

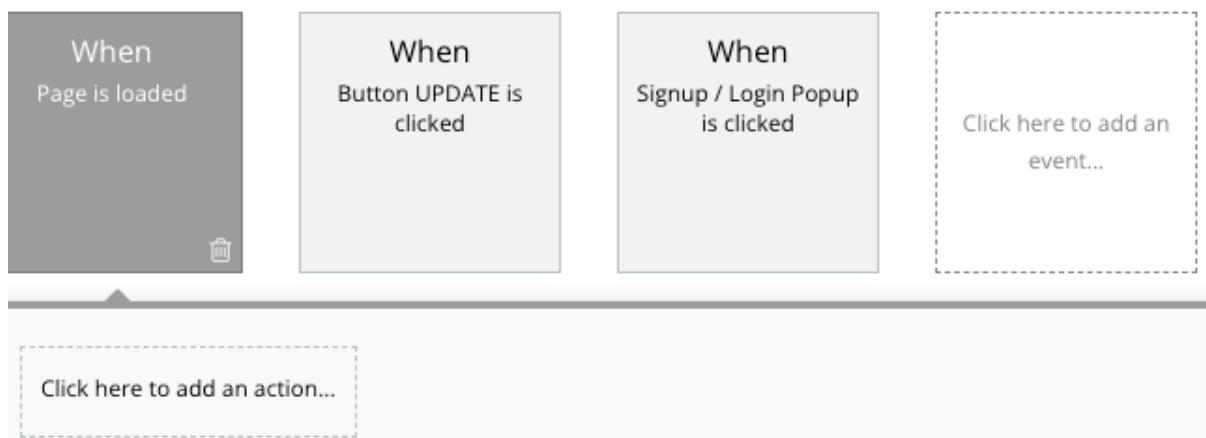
以上で、`bubble_fn_age` という関数（処理のかたまり）に javascript から値を渡すと、この画面要素で受け取れるようになります。



< Advanced >

次に javascript を実行する箇所を定義します。

- 左メニューから Workflow を選択する
- 正方形がならんでいるが、その一番右の **Click here to add an event...** を選択する
- **General > Page is loaded** を選択する



< Advanced >

- Click here to add an action... をクリックする
- Plugins > Run javascript をクリックする
- 設定が開くので、 Script の欄に次のページのコードを貼り付ける



The screenshot shows a 'Run javascript' dialog box. The title bar says 'Run javascript' with close and minimize buttons. The main area contains the following JavaScript code:

```
Script. To use a return value, use in conjunction with
//生年月日
const birthday = {
  year: ,
  month: ,
  date: 
};

function getAge(birthday){

  //今日
  let today = new Date();

  //今年の誕生日
  let thisYearsBirthday = new
Date(today.getFullYear(), birthday.month-1,
birthday.date);

  //年齢
  let age = today.getFullYear() - birthday.year;

  if(today < thisYearsBirthday){
    //今年まだ誕生日が来ていない
    age--;
  }

  return age;
}

bubble_fn_age(getAge(birthday));
```

At the bottom of the dialog, there are buttons for 'Rich text editor', 'show documentation', 'Asynchronous' (with a checked checkbox), and another 'show documentation' button.

< Advanced >

```
//生年月日
const birthday = {
  year: ,
  month: ,
  date: 
};

function getAge(birthday){

  //今日
  let today = new Date();

  //今年の誕生日
  let thisYearsBirthday = new Date(today.getFullYear(), birthday.month-1, birthday.date);

  //年齢
  let age = today.getFullYear() - birthday.year;

  if(today < thisYearsBirthday){
    //今年まだ誕生日が来ていない
    age--;
  }

  return age;
}

bubble_fn_age(getAge(birthday));
```

< Advanced >

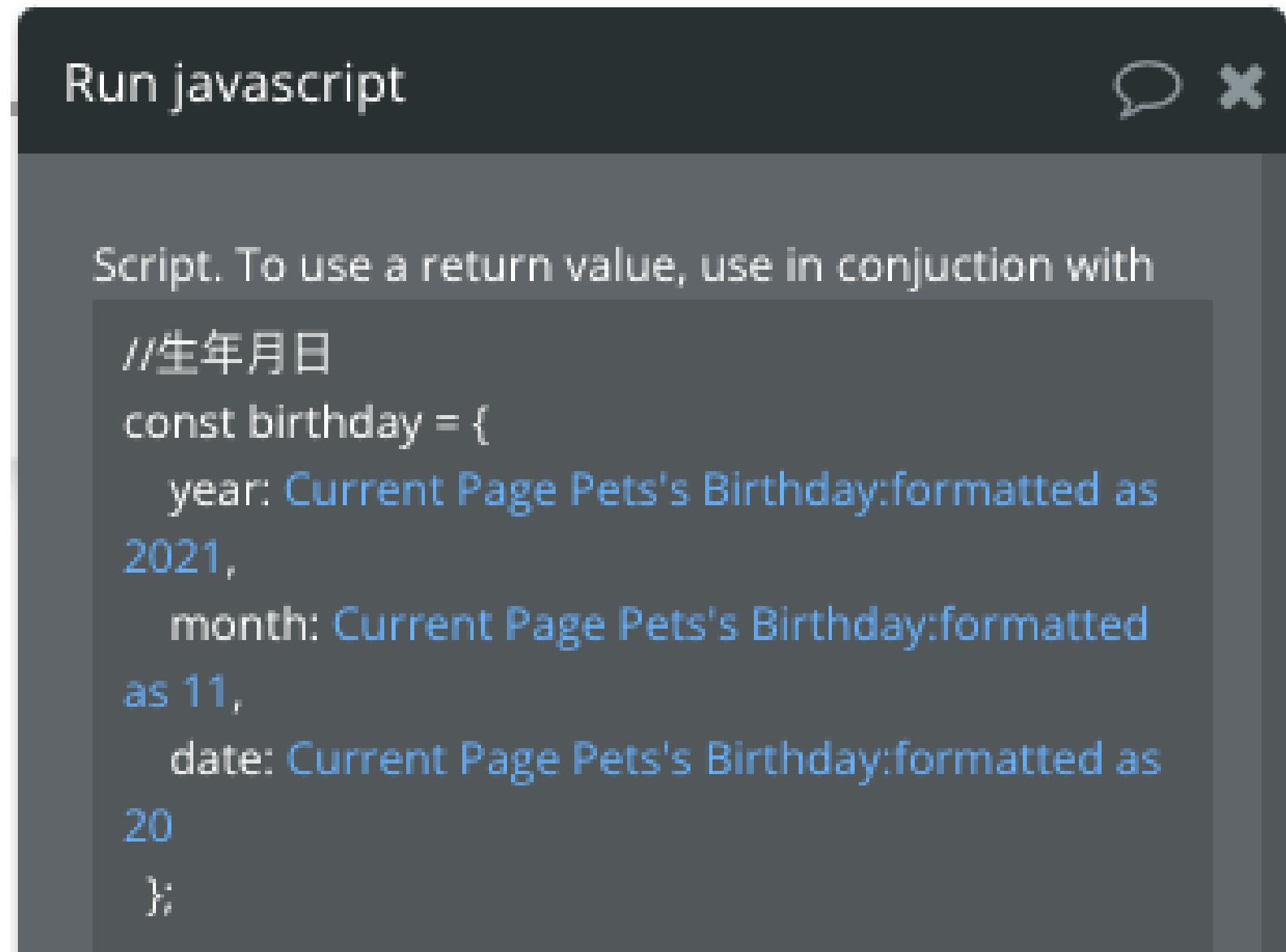
3行目から5行目の、`year:`、`month:`、`date:` の後ろに `insert dynamic data` で年月日を差し込む

- `year:` の後ろ (, の前) にカーソルを置く
 - `insert dynamic data` > `Current Page Pets` > 's Birthday
 - `More` > `formatted as 11/20/21`
 - `Format type` に `Custom` を指定する
 - `Custom format` に `yyyy` を指定する
- 同様に `month:` の後ろにも、`Custom format` を `m` にして差し込む
- 同様に `date:` の後ろにも、`Custom format` を `d` にして差し込む

※入力後イメージは次のページ

< Advanced >

入力後のイメージ



< Advanced >

表示するための画面要素を配置しましょう

- Birthday のラベルとテキストをコピー&ペースト
- ラベルを Age に変更
- テキストの中身を `JavascripttoBubble A > 's value` と指定する

[← Back to List](#)[View Weights](#)

Name (Initial)

taro (T)

Image



Category

dog

Birthday

2023/10/20

Age

1

Gender

女の子

Latest Weight

10.5kg

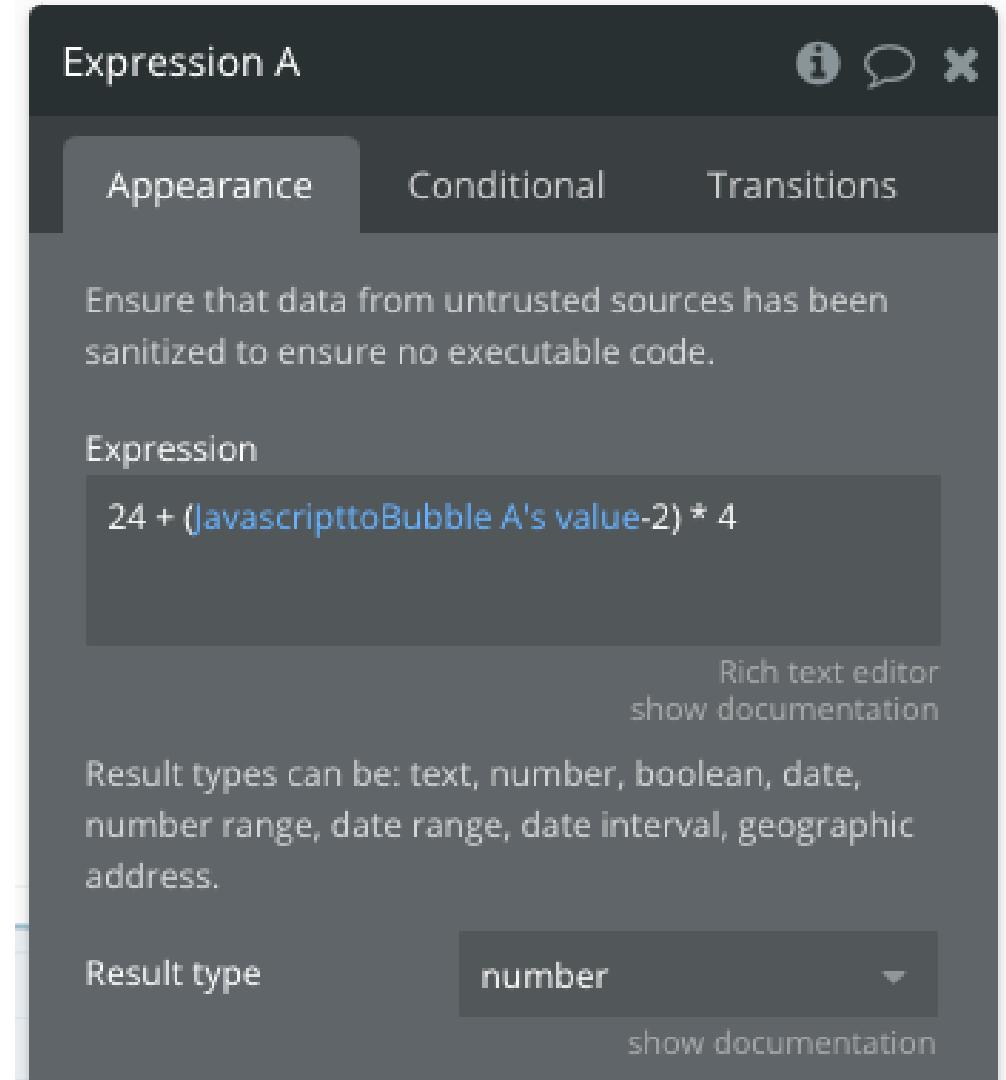
< Advanced >

プレビューしてみましょう

< Advanced >

犬や猫の年齢は

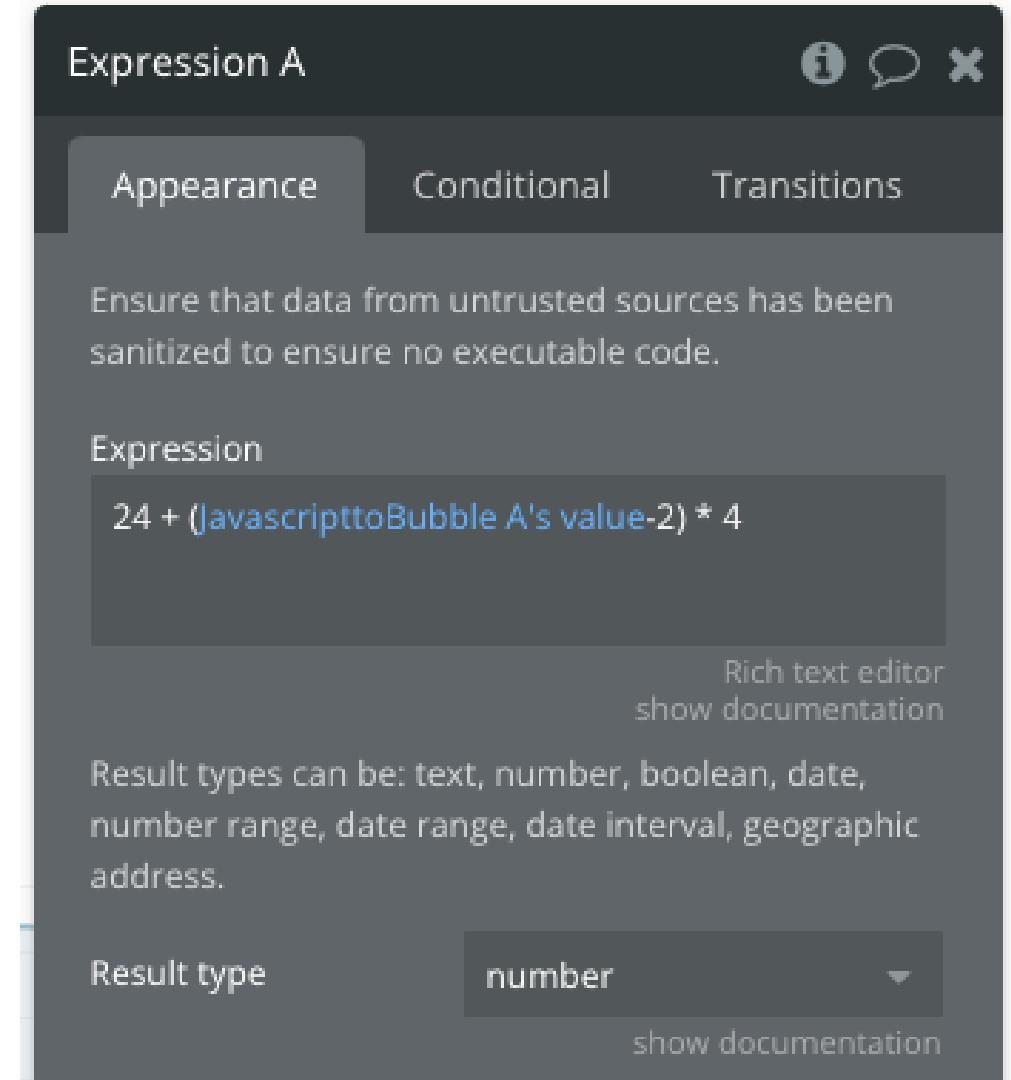
- Visual elements から Expression を選択して、先ほどの Javascript to Bubble の横辺りに配置する
- Expression に 24 + (と入力する
- insert dynamic data で JavascripttoBubble A > 's value を差し込む
- 続きに -2) * 4 と入力する
- Result type に number と指定する



< Advanced >

表示の設定をします。

- Age のラベルを犬猫年齢も含むことがわかりやすいように、
Age(as Dog/Cat) と変更する
- Age のテキストの中身に元々入力されているものの後ろに (と入力する
- insert dynamic data で Expression A > 's value を差し込む
-) と入力する



< Advanced >

なお、次に犬猫年齢の計算方法はサイズによっても違うようです。

今回利用した以下の年齢計算は3歳以上の小型犬の年齢計算をする場合の式となります。

$$\text{犬猫年齢} = 24 + (\text{人間年齢} - 2) \times 4$$

興味があれば、厳密に計算ができるよう作り込んでみましょう。

< Advanced >

プレビューしてみましょう

Name (Initial)

taro (T)

Image



Category

dog

Birthday

2019/11/11

Age (as Dog/Cat)

5 (36)

Gender

女の子

Latest Weight

10.5kg

画面を権限によって切り替える

画面を権限によって切り替える

ここまで、画面操作へのフィードバックやデータの抽出・加工など部分部分でのロジックの組み込みを説明してきました。

次は、複数機能にまたがるようなロジックをいれていくうと思います。

以下のことを行います。

- ユーザーをペットの飼い主（Pet Owner）とペットの飼育アドバイザー（Pet Advisor）に分ける
- 飼い主は今まで作ってきた画面・機能を利用できる
- アドバイザーはアドバイザー専用の画面・機能を利用できる

開発の流れとしては以下の順番に作り混んでいきます。

- ユーザー情報に飼い主かアドバイザーかを判別できるフィールドを追加する
- ユーザー登録時に、飼い主かアドバイザーかを選択して登録できるようにする
- アドバイザーの一覧画面、詳細画面を作成する
- 飼い主かアドバイザーかによって、ログイン後・サインアップ後の画面遷移先を切り替える

手順は多くかかりますが、複数のユーザー種別を扱うプロダクトはよくありますので、ぜひ身につけていきましょう

ユーザーを判別できるフィールドを追加する

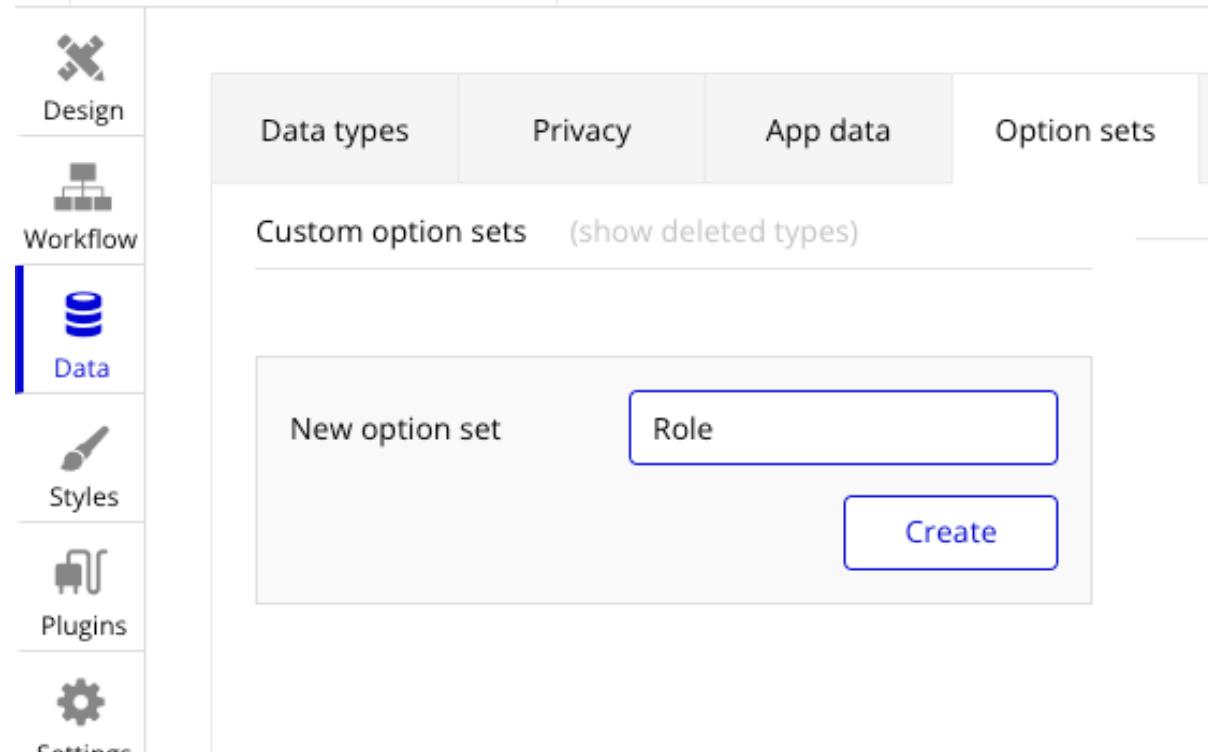
まずは飼い主かアドバイザーかという役割の違いをデータ上保持できるようにします。

ペットのオススメの時のようにテキストで保持してもよいのですが、決まった選択肢の中から指定するような値は、選択肢をあらかじめ定義して利用することで扱いやすくなります。

Bubble から Option set という仕組みが提供されているので、それを使ってみましょう。

Options を設定してみましょう

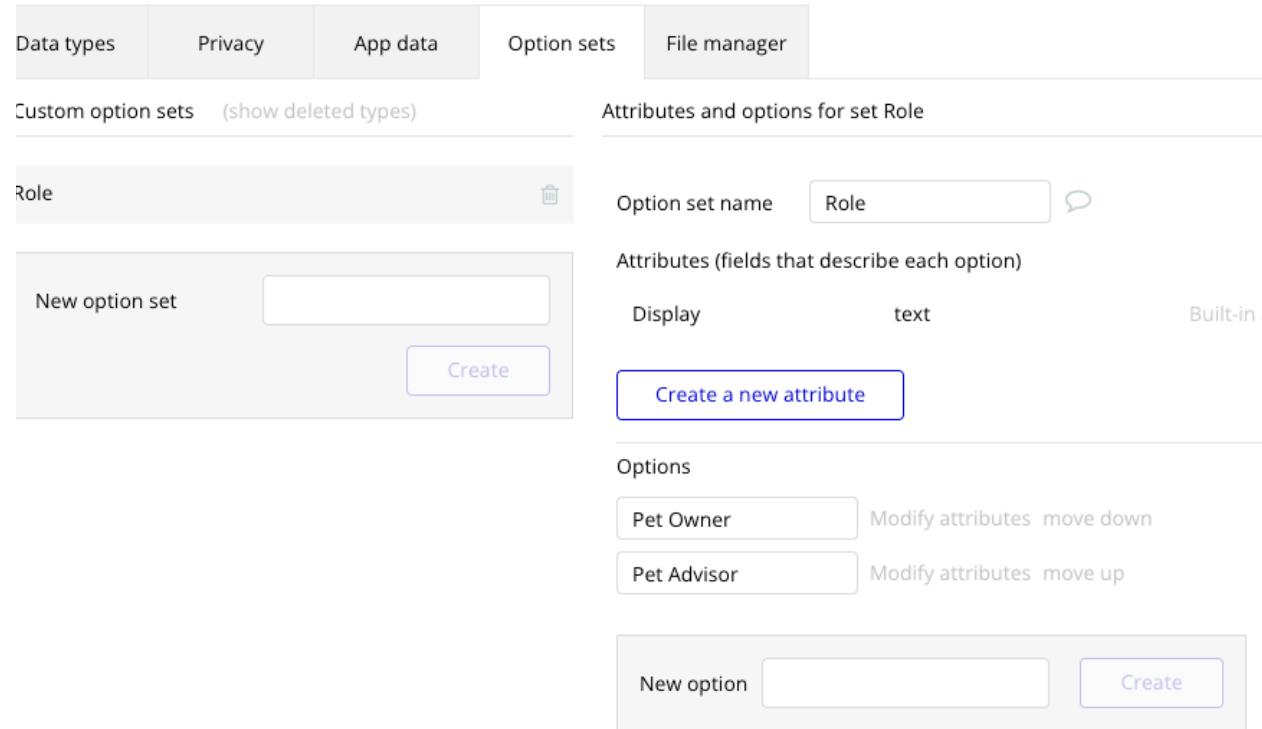
- 左メニューの **Data** > タブの **Option sets** と移動する
- New Option set に **Role** と入力して、Create ボタンを押す
- 新たな Option set として Role が作成されます



Role という Option set（選択肢のセット）に、具体的な Option（選択肢）を追加していきます。今回は、Pet Owner と Pet Advisor を作成します。

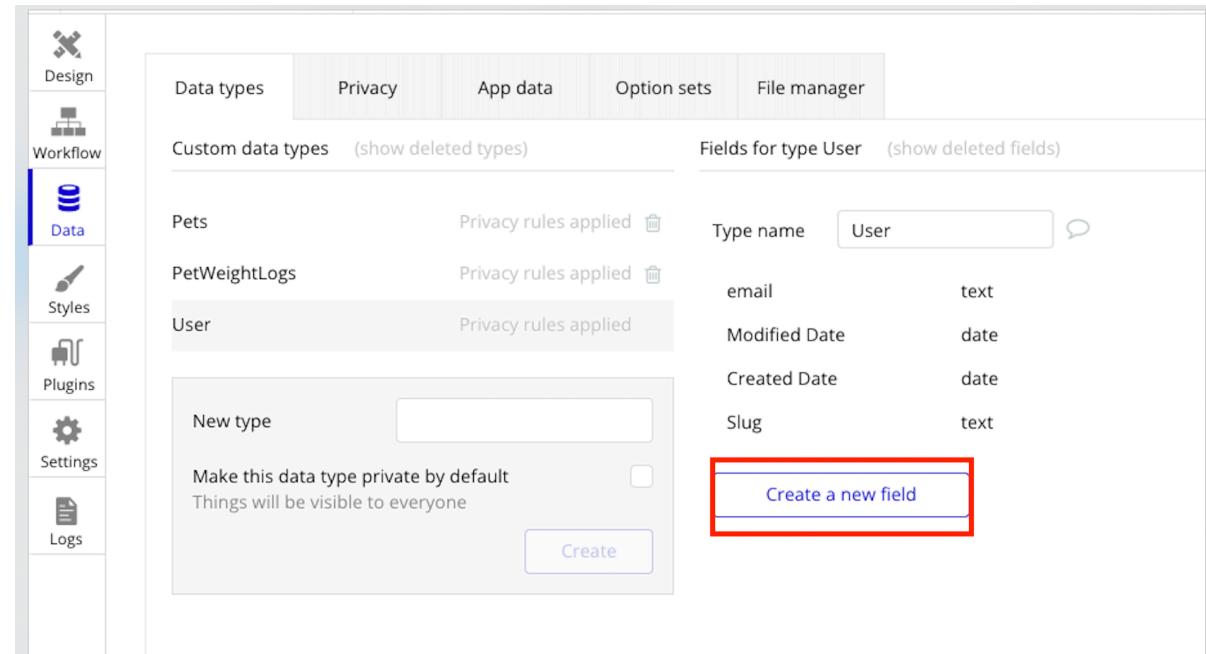
- 画面右下の New Option に Pet Owner と入力して、Create ボタンを押してください
- 同様に New Option に Pet Advisor と入力して、Create ボタンを押してください

以上で設定完了です



次はユーザーの属性として、ロールを追加しましょう

- 左メニューの **Data** > タブの **Data types** と移動する
- User** を選択する
- 画面右下の **Create a new field** ボタンをクリックする



- Field name に Role と入力する
 - これはわかりやすいものであれば、どういう名前でも構いません
- Field type に Role を選択する
 - ここで指定しているのは先ほど作成した Option set としての Role になります。
- Create ボタンを押す

Create a new field

Field name

Field type ▾

This field is a list (multiple entries)

CREATE Cancel

新しくフィールドを追加したので、既に作成してしまっているユーザーについては Role が空になってしまいます。後々、不整合を招きますので、既存データにパッチ（データ補正）をあてておきましょう。

- App Data タブ荷移動して、All Users を選択する
- 表が表示されるので、表の左端のペンアイコンをクリックして、1件ずつ編集する
 - 今作成されているユーザーはすべて飼い主のはずなので、Role に Pet Owner を指定する

Modify an existing database entry

Type of thing	User
Role	Pet Owner
Slug	
Email	kim+2@guildworks.jp
Unique id	1637274311987x540151212983663400
Created Date	Nov 19, 2021 7:25 am
Modified Date	Nov 19, 2021 7:25 am

SAVE **Cancel**

All PetWeightLogs Run as → kim@guildworks.jp Nov 16, 2021 7:19 pm Nov 19, 2021 6:25 am

Users の中のすべての行がの Role が Pet Owner になっていればOK

The screenshot shows a user interface for managing application data. On the left, there is a vertical sidebar with icons for Design, Workflow, Data (which is selected), Styles, Plugins, Settings, and Logs. The main area has tabs for Data types, Privacy, App data, Option sets, and File manager, with App data selected. Below this, it says "Database views" and "Application data - All Users - Development version". There are buttons for New view, Primary fields, Search, Delete (0), Upload, Modify, Export, Bulk, and New entry. A red box highlights the "Role" column in the data table. The table has columns for Email, Role, Created Date, and Modified Date. The data shows six entries, all of which have "Pet Owner" listed under the Role column.

		Email	Role	Created Date	Modified Date
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kim+2@guildworks.jp	Pet Owner	Nov 19, 2021 7:25 am	Nov 20, 2021 6:28 am
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kim+advisor2@guildworks.jp	Pet Owner	Nov 19, 2021 6:32 am	Nov 20, 2021 6:28 am
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kim+advisor@guildworks.jp	Pet Owner	Nov 19, 2021 5:35 am	Nov 20, 2021 6:28 am
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kim@guildworks.jp	Pet Owner	Nov 16, 2021 7:19 pm	Nov 20, 2021 6:28 am
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kyogoku+bubble_test2@guildw	Pet Owner	Nov 12, 2021 10:10 am	Nov 20, 2021 6:28 am
<input type="checkbox"/>	<input type="checkbox"/> Run as →	kyogoku+bubble_test@guildw	Pet Owner	Oct 31, 2021 8:51 pm	Nov 20, 2021 6:28 am

ユーザー登録時にロールを指定できるようにする

では、次はユーザー登録時に飼い主なのかアドバイザーなのか指定して登録できるようにします。

登録画面は Bubble が用意しているものを使い回してきましたが、そこに手を入れます。

- ログインページ `index` に移動する
- `Password` のラベルをコピーして、`Role` というラベルを配置する
- `Design` メニューの `Input forms` の中から `Dropdown` を選択して、パスワード入力欄の下に配置する

Sign up

Join millions of users taking notes on Bubble

Full name

Email

Role

Password

Re-enter password

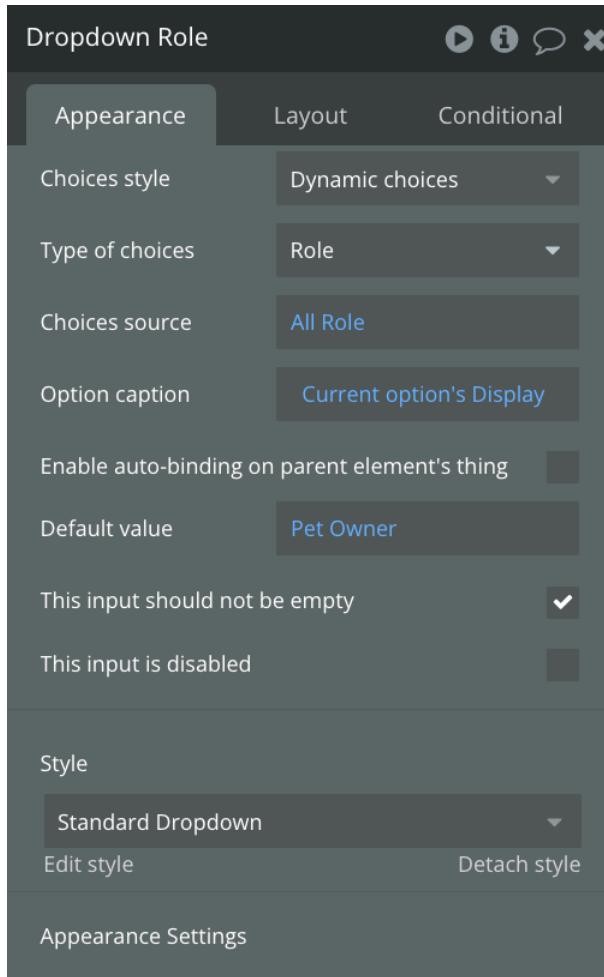
Sign Up

Already have an account?

- Dropdown の設定は以下のとおりとする
 - Element 名 : Dropdown Role
 - Placeholder: Choose a role...
 - Choice style : Dynamic choices
 - Type of choices : Role
 - Choices source : All Role
 - Option caption : Current option > 's Display
 - Default value : Pet Owner
 - This input should not be empty : チェック

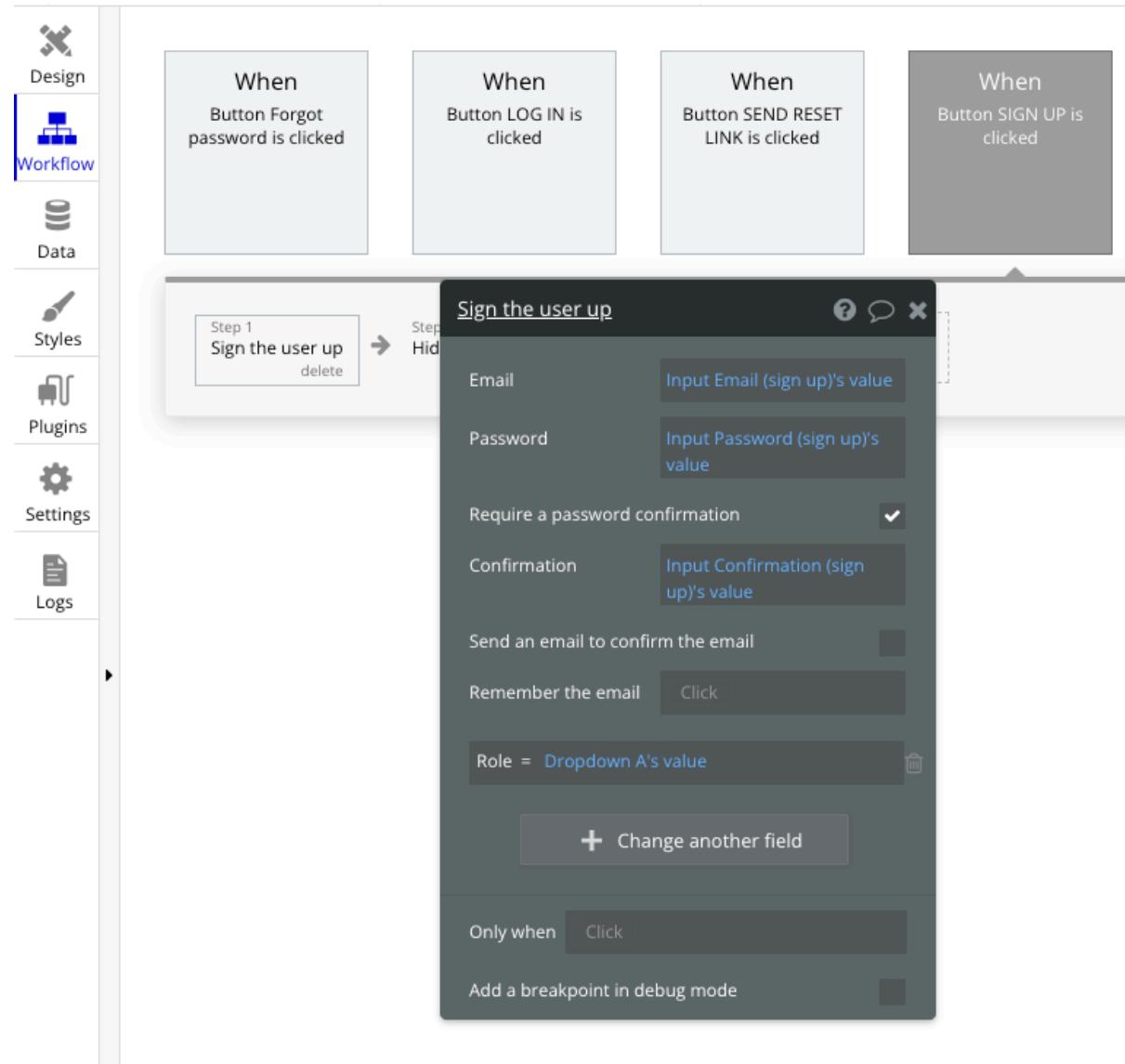
※画面イメージは次のページ

入力後イメージ



つづいて、入力された Role がユーザー登録時に設定されるようにします

- 左メニューから Workflow > 並んでいる正方形の中から Button Sign up is clicked > 並んでいる Action から Sign the user up の順に移動する
- Action の設定画面内の Change another field ボタンをクリックする
- 入力欄があるので、Role = Dropdown Role 's value と選択する



プレビューと動作確認をしましょう

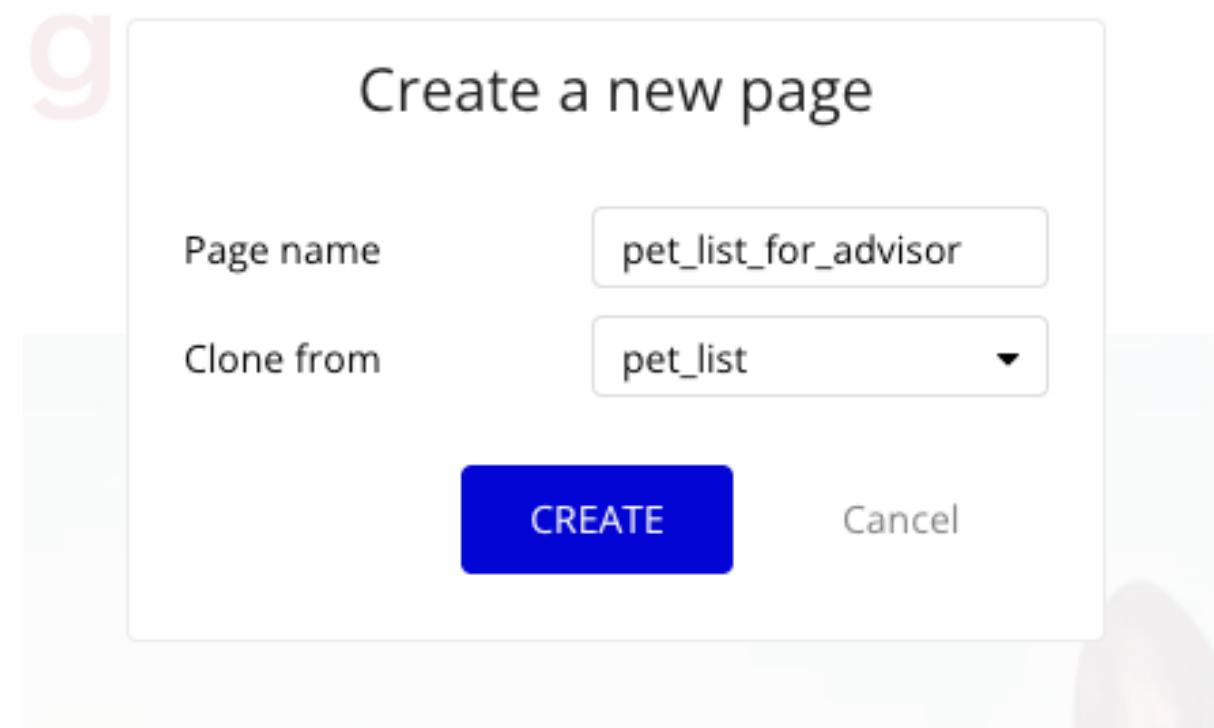
アドバイザーとして、アカウント登録して、データを確認してみましょう。

		Email	Name	Role
<input type="checkbox"/>	 Run as →	kim+advisor@guildworks.jp	Sanhe Kim (as advisor)	Pet Advisor

アドバイザーの一覧画面を作成する

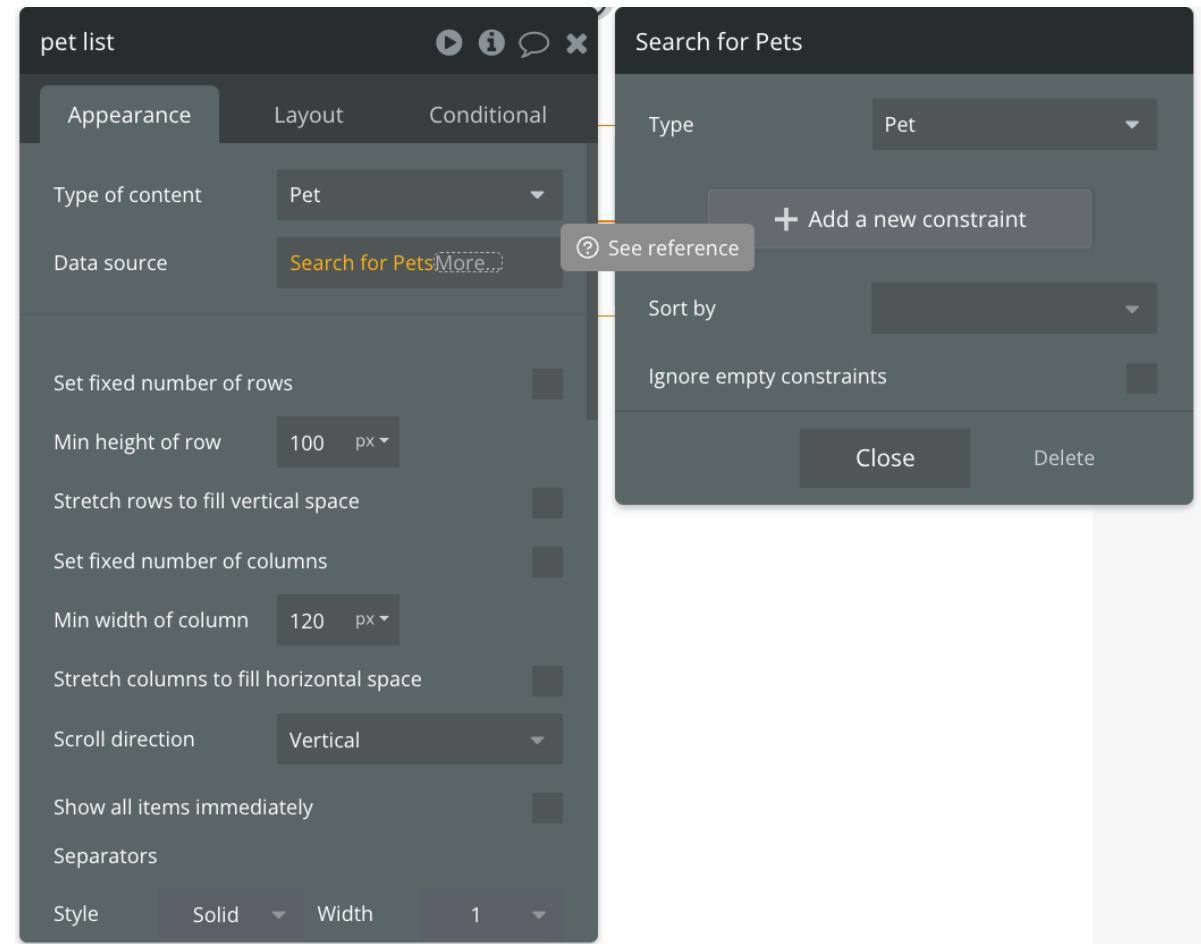
アドバイザーの一覧画面を作りましょう

- ロゴ横のメニューを開いて、
`Add a new page...`
- `Page name` に
`pet_list_for_advisor` と入力する
- `Clone from` で `pet_list` を選ぶ
- 新しく画面が作成される



アドバイザーは登録されたすべてのペットを参照できるようにします。

- もとの検索条件を削除する
 - pet list を選択して、Data source の Do search for をクリックする
 - Created By = Current User と条件指定されている部分をゴミ箱アイコンをクリックして削除
- アドバイザーは多くのペットを閲覧する必要があるので、セルのサイズは小さくする。
 - Min width に 120 と Min height に 100px を指定



プレビューしましょう

まだログイン後の遷移先は通常のペットリストになっているので、Preview ボタンから直接アドバイザー用のペットリストを開きます。

...何も出ない！？なんで？？



権限がないからです。

Bubble での権限制御

今まで意識しなくてよかったですが、
Bubble では Data へのアクセスが厳密
に制限されています。

左メニューの Data > タブの Privacy

と移動してください。

初期状態では、データは作成者しかア
クセスできなくなっています。

当然と言えば当然ですね。

The screenshot shows the Bubble Privacy settings page. At the top, there's a list of entities with their privacy status: Pets (Privacy rules applied), PetWeightLogs (Privacy rules applied), and User (Privacy rules applied). Below this, a specific rule is defined:

Name	Visible to creator	
When	This Pet's Creator is Current User	
Users who match this rule can...		
View all fields	<input checked="" type="checkbox"/> Find this in searches	<input checked="" type="checkbox"/> View attached files

Below the rule definition, there are sections for "Everyone else (default permissions)" and "Custom permissions". Under "Everyone else", "View all fields" is checked. Under "Custom permissions", several checkboxes are available: Birthday, Modified Date, Gender, Slug, Created By, Find this in searches, View attached files, and Allow auto-binding. A blue button at the bottom right says "Define a new rule".

では、アドバイザーだったら、すべてのデータを見れるという権限を追加していきます。

- Data タブから、さらに Privacy タブの中で、Pets を選択する
- Define a new rule ボタンをクリックする
- Rule name に Visible to advisor と入力する
- When に Current User 's Role is Pet Advisor と選択する
 - ユーザーがアドバイザーだった場合という条件です

これでアドバイザーだったら、すべてのペットのデータが見れます。

ルールごとに参照できるフィールドを限定することができますが、今回は利用しません。

*画面イメージは次のページ

Custom data types

Pets Privacy rules applied

PetWeightLogs Privacy rules applied

User Privacy rules applied

Data rules for type Pets

Name Visible to advisor

When Current User's Role is Pet Advisor

Users who match this rule can...

View all fields Find this in searches View attached files Allow auto-binding

Name Visible to creator

When This Pet's Creator is Current User

Users who match this rule can...

View all fields Find this in searches View attached files Allow auto-binding

Everyone else (default permissions)

View all fields

Birthday	<input type="checkbox"/>	Gender	<input type="checkbox"/>	Image	<input type="checkbox"/>	Name	<input type="checkbox"/>	Created Date	<input type="checkbox"/>
Modified Date	<input type="checkbox"/>	Slug	<input type="checkbox"/>	Created By	<input type="checkbox"/>		<input type="checkbox"/>		<input type="checkbox"/>

Find this in searches View attached files Allow auto-binding

[Define a new rule](#)

Define a new rule

では、同じように PetWeight にもルールを追加してください。
これでアドバイザーはすべてのデータを見れるようになっているはずです。

プレビューしましょう

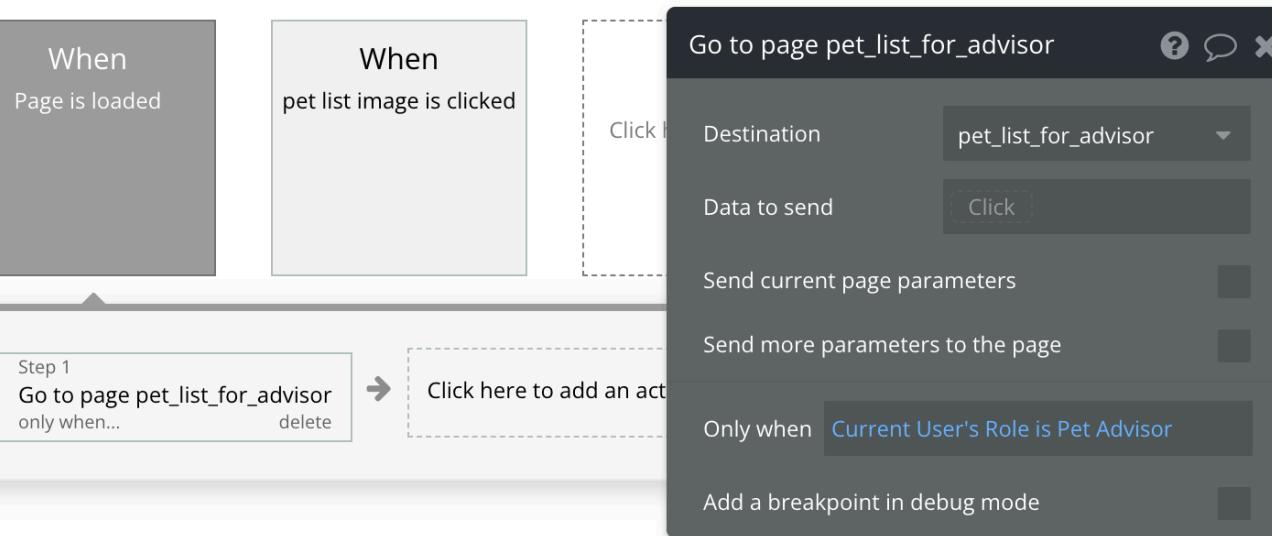
一覧に表示されましたか？他の Pet Owner のユーザーを作つてペットを登録してもこちらに表示されます。

[Log out](#)[Register](#)

次に、ログイン時の遷移先を制御します。

アドバイザーの場合は、`pet_list` に遷移した際に、`pet_list_for_advisor` に遷移するという Action を追加します

- pet_list ページの Workflow をひらく
- Click here to add an event.. をクリックする
- General > Page is loaded を選択
- Click here to add an action.. をクリックする
- Navigation > Go to page.. をクリックする
- 設定が開くので、Destination に pet_list_for_advisor を選択する
- Only when に Current User 'sRole is Pet Advisor を選択する



プレビュー&動作確認しましょう

アドバイザーでログインしたら



PetLog

Log out

Register



飼い主でログインしたら



Log out

Register



よしよし

< Advanced >

アドバイザーで勝手にアカウント作られて、

勝手に情報見られていの？

< Advanced >

システム管理者が承認しないと、利用開始出来ないようにしよう

< Advanced >

以下をやっていきます

- ユーザー情報にアドバイザーとして承認されているかを示すフィールドを追加する
- データへのアクセス権限はアドバイザーであることと同時に、承認されていることを条件に加える
- アドバイザー用のペットリストには、未承認だったら審査中ですというメッセージをだす

< Advanced >

ユーザー情報にアドバイザーとして承認されているかのフィールドを追加する

- 左メニューから Data > タブの Data types > 中の User とクリックしていく
- 画面右下の Create a new field ボタンをクリックする
- Field name に Approved As Advisor と入力する
- Field type に yes/no を選択する
- Create ボタンをクリック

Create a new field

Field name	Approved As Advisor
Field type	yes / no
This field is a list (multiple entries) <input type="checkbox"/>	
CREATE	Cancel

< Advanced >

追加されたフィールドに `default` という欄があるので、`no`（もしくは `いいえ`）を設定しておく。

作成されたタイミングでは、未承認状態となる。

The screenshot shows the 'Fields for type User' configuration page. On the left, there's a sidebar with 'Data types', 'Privacy', 'App data', 'Option sets', and 'File manager'. Below that are sections for 'Custom data types' (listing 'Pets', 'PetWeightLogs', and 'User') and a 'New type' input field with a 'Create' button. A note says 'Make this data type private by default' with a checkbox and a message 'Things will be visible to everyone'. On the right, under 'Fields for type User', there's a table with columns for 'Type name' (set to 'User'), 'Field name' (e.g., 'Approved As Advisor', 'Role', 'email', 'Modified Date', 'Created Date', 'Slug'), 'Type' (e.g., 'yes / no', 'Role', 'text', 'date', 'date', 'text'), and 'Default' (dropdown menu showing 'no'). The 'Approved As Advisor' row is highlighted with a red box around its entire row.

Type name	Field name	Type	Default
User	Approved As Advisor	yes / no	no
User	Role	Role	default
User	email	text	Built-in field
User	Modified Date	date	Built-in field
User	Created Date	date	Built-in field
User	Slug	text	Built-in field

< Advanced >

既存のユーザーについては、 Approved As Advisor はすべて no にしておく
(めんどうで、しょんぼり。。だけど、大事！)

	Email	Approved As Advisor	Role	
<input type="checkbox"/>	Run as → kim+advisor3@guildworks.jp	no	Pet Advisor	Ni
<input type="checkbox"/>	Run as → kim+2@guildworks.jp	no	Pet Owner	Ni
<input type="checkbox"/>	Run as → kim+advisor2@guildworks.jp	no	Pet Owner	Ni
<input type="checkbox"/>	Run as → kim+advisor@guildworks.jp	no	Pet Owner	Ni
<input type="checkbox"/>	Run as → kim@guildworks.jp	no	Pet Owner	Ni
<input type="checkbox"/>	Run as → kyogoku+bubble_test2@guildwo	no	Pet Owner	Ni
<input type="checkbox"/>	Run as → kyogoku+bubble_test@guildwo	no	Pet Owner	Oi

< Advanced >

データへのアクセス権限は承認されているかも見る

- 左メニューから Data > タブの Privacy > 中の Pets とクリックしていく
- Visible to advisor の When の条件が記載されている部分の末尾の Pet Advisor をクリックする
- More があらわれるので More をクリックする
- and Current User 's Approved As Advisor is "yes" と選択する
- PetWeight も同様にする

< Advanced >

動作確認してみよう

Approved As Advisor が no のユーザーでログインしてみる



Log out

Register

よし

< Advanced >

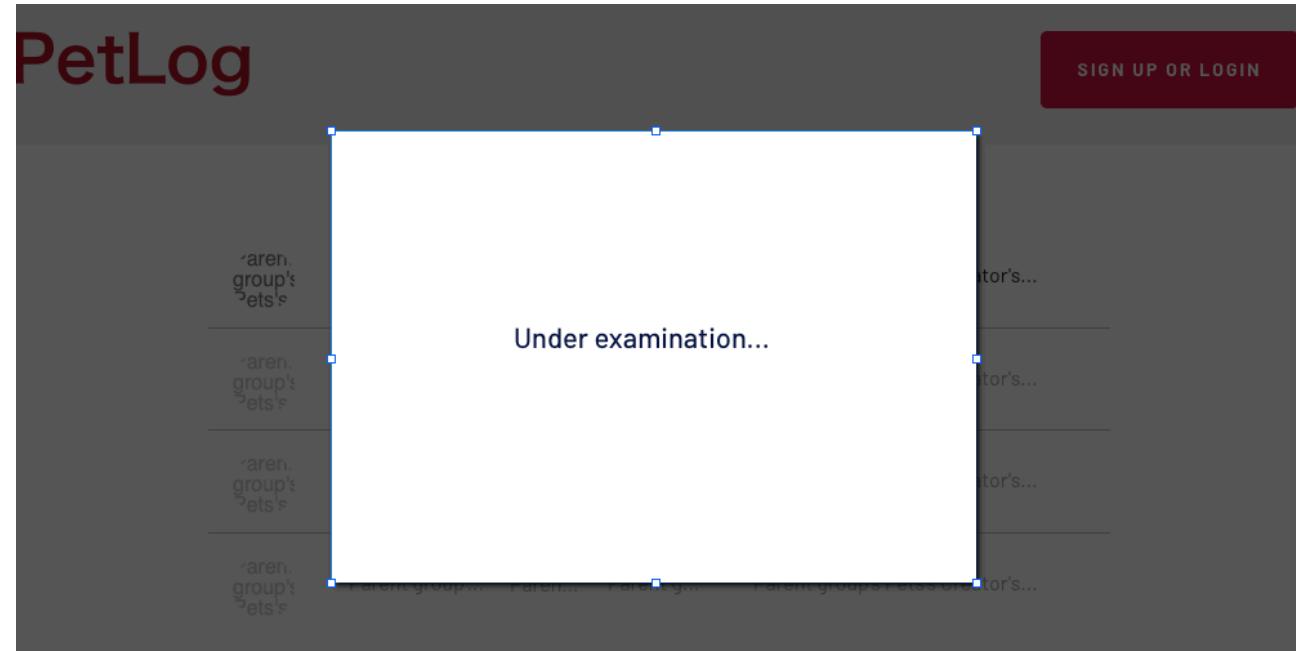
データを直接編集して Approved As Advisor が yes にしたら？

[Log out](#)[Register](#)

よし

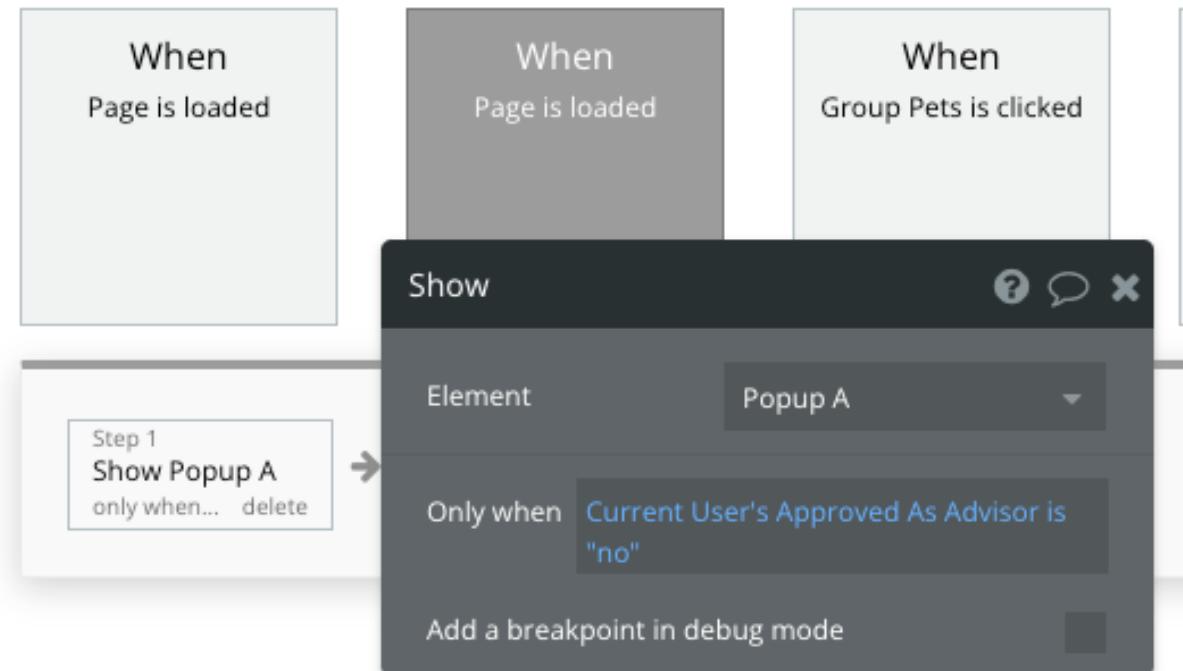
未承認だったら審査中ですと いうメッセージをだす

- `pet_list_for_advisor` の画面で
`Design` メニューを開く
- `Popup` を追加する
- `Layout` タブで、`Container`
`laout` に `Align to parent` を指
定する。
- `Popup` の上にテキスト Element を
追加する。
- 審査中である旨のメッセージを記
載する
- `Layout` タブで、親要素に対して
中央に表示されるように指定す



< Advanced >

- メニューから Workflow に移動する
- Click here to add an event... > Page is loaded とクリックする
- Click here to add an action... > Element Actions > Show とクリックする
- Element に Popup A を指定する
- Only when に、 Current User 's Approved As Advisor is "no" を指定する



< Advanced >

動作確認してみよう

yes のアドバイザーなら



Log out

Register

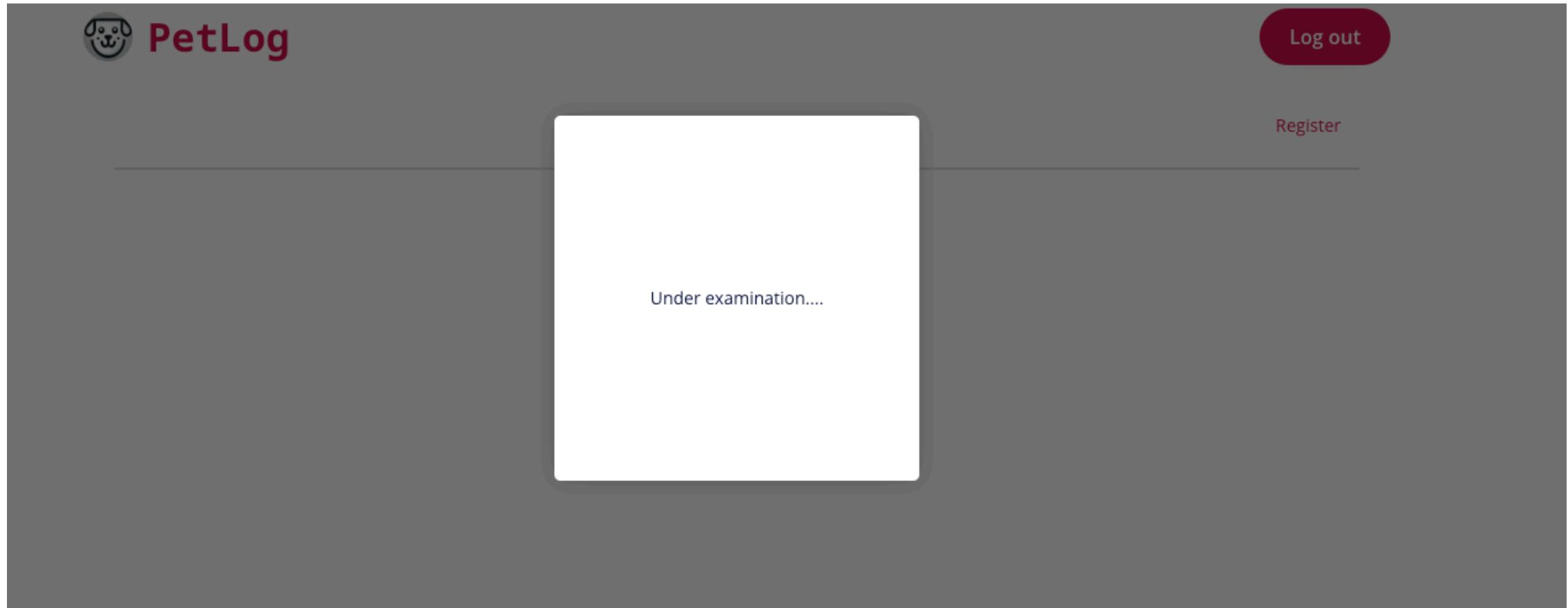


よし

170

< Advanced >

no なら



よし

< Advanced >

システム管理者はどうやって気づくの？

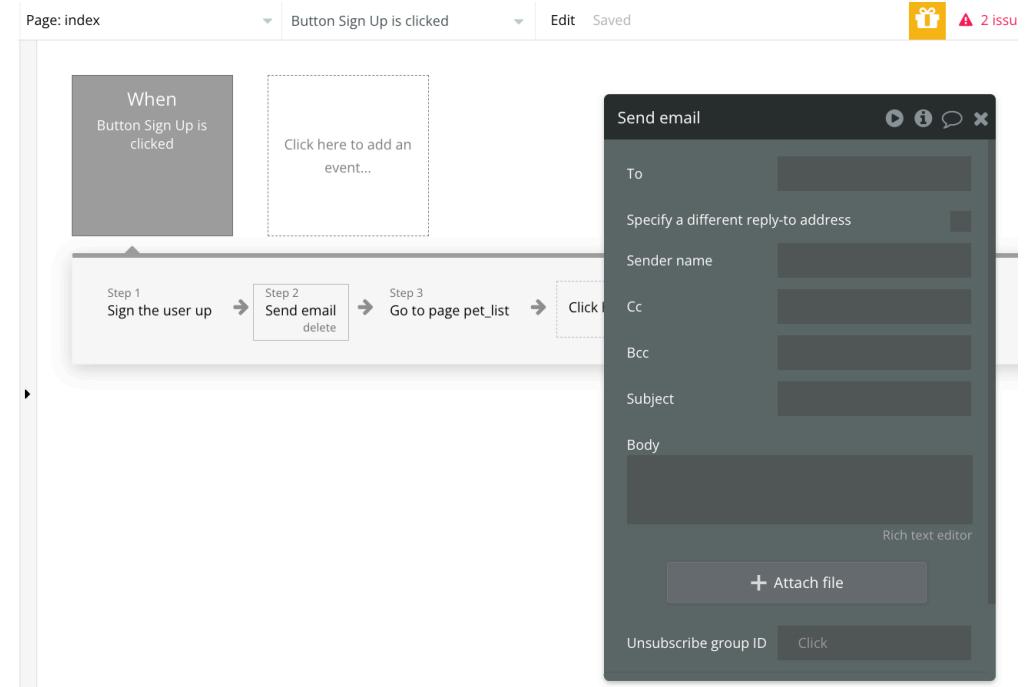
< Advanced >

アドバイザーの登録がされたら、システム管理者にメール通知されるようしょう

< Advanced >

システム管理者にメール通知 されるようにしよう

- index ページを開く
- メニューから Workflow に移動して、Button Sign up is clicked を選ぶ
- Click here to ad an action... > Email > Send Email とクリックする
- Action の位置を Go to page pet_list の前にドラッグして移動する

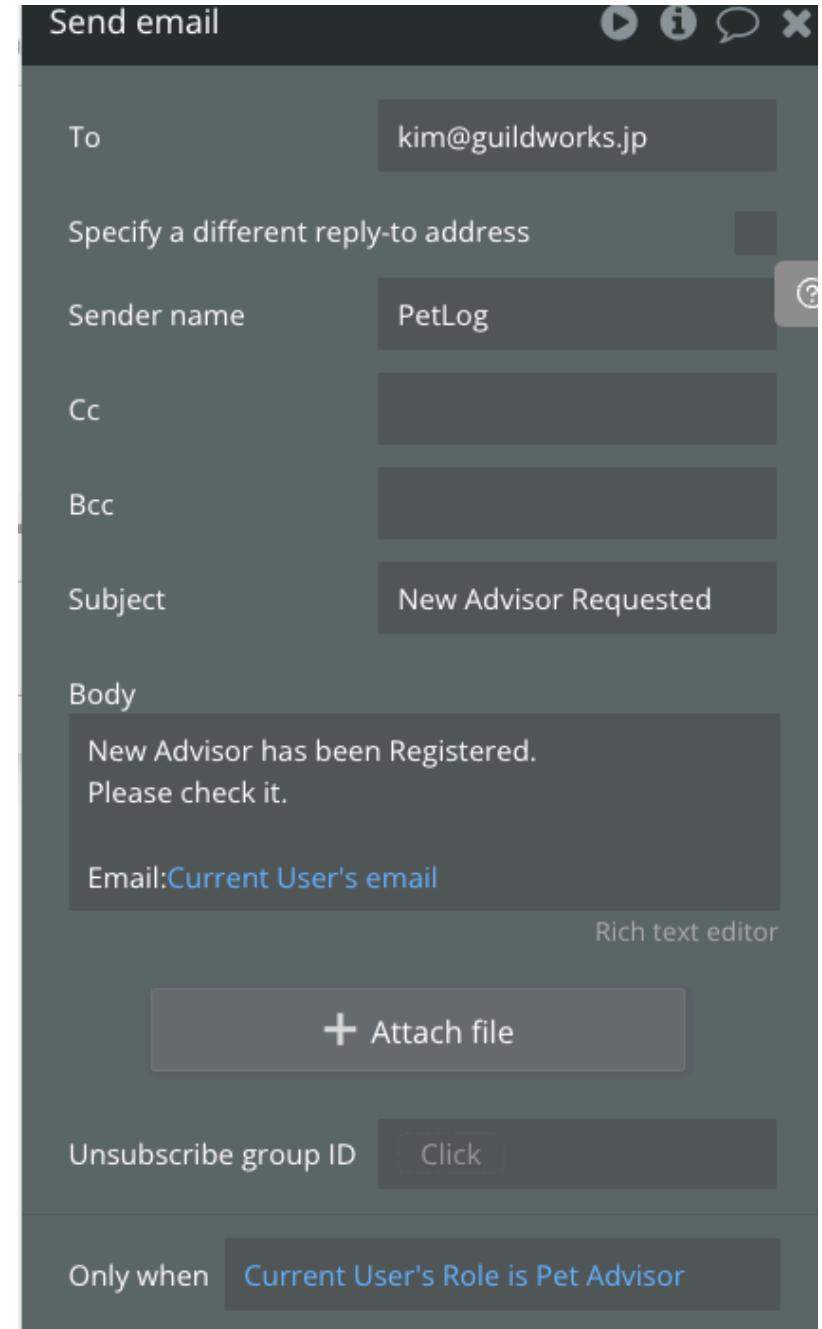


< Advanced >

- To に自分のメールアドレスを設定する
- Sender name は PetLog
- Subject は New Advisor Requested
- Body は以下の本文の末尾に dynamic data insert で Current User 's email を選択

New Advisor has been Registered.
Please check it.

Email:



< Advanced >

- Only when に、Current User's Role is Pet Advisor を指定する

< Advanced >

動作確認してみよう

アドバイザーでサインアップしたら

PetLog

New Advisor Registered

宛先: kim@guildworks.jp,

返信先: titech-bubble-2-suburi-2021119-no-reply@bubbleapps.io

New Advisor has been Registered.
Please check it.

Email: kim+advisor4@guildworks.jp

< Advanced >

飼い主なら

うん、こない。やったー

ここまでのおさらい

デザインを作り込みました

- ディスプレイサイズに合わせた画面をつくりました
 - レスポンシブウェブデザインという手法を使って、以下のようなルールを用いて、ディスプレイサイズに合わ見た目を制御しました。
 - 親要素内の配置ルール
 - 要素のサイズ決定ルール
 - 表示の有無ルール
- Style を使ってみました
 - Style を編集・追加したり、個別にスタイルをあてました

ロジックを作りこみました

- 画面操作に対するフィードバックを返しました
- データを抽出、加工しました
- 画面を権限によって切り替えました

Bubble で様々なところにロジックを埋め込めるのを見ていきました

< Excercise >

演習

ここまでで学んだ、デザインやロジックの作り込み方を使って、機能を追加してみよう。

例えば、アドバイザーから飼い主、もしくは飼い主からアドバイザーにアプローチできる機能をつくって見よう

- 例：飼い主にアドバイスを送れる
- 例：広告を掲載できる
- 例：アドバイザーに相談を投げられる などなど

ここまでのかえり

ここまで、Bubble 単体での講義は終わりになります。

触れられなかった機能もたくさんありますが、

Bubble はマニュアルとリファレンスが充実していますので、

もし Bubble を採用するならぜひ活用してください。

マニュアルはこちらです。

<https://manual.bubble.io/>

リファレンスは画面上でわからないものにカーソルを合わせると現れます。

だいたいの機能についてリファレンスへのリンクが浮き出できます。

このあとは Bubble を他のシステムと連携したりチームで開発することを学んでいきます。

外部システムと連携する

以下のように様々なシステムと連携して機能を作っていきます。

- ChatGPT を使って AI ペットアドバイザーを作成
- YouTube の動画一覧を表示
- Google アカウントを使ったソーシャルログイン
- YouTube の動画一覧を動的に検索

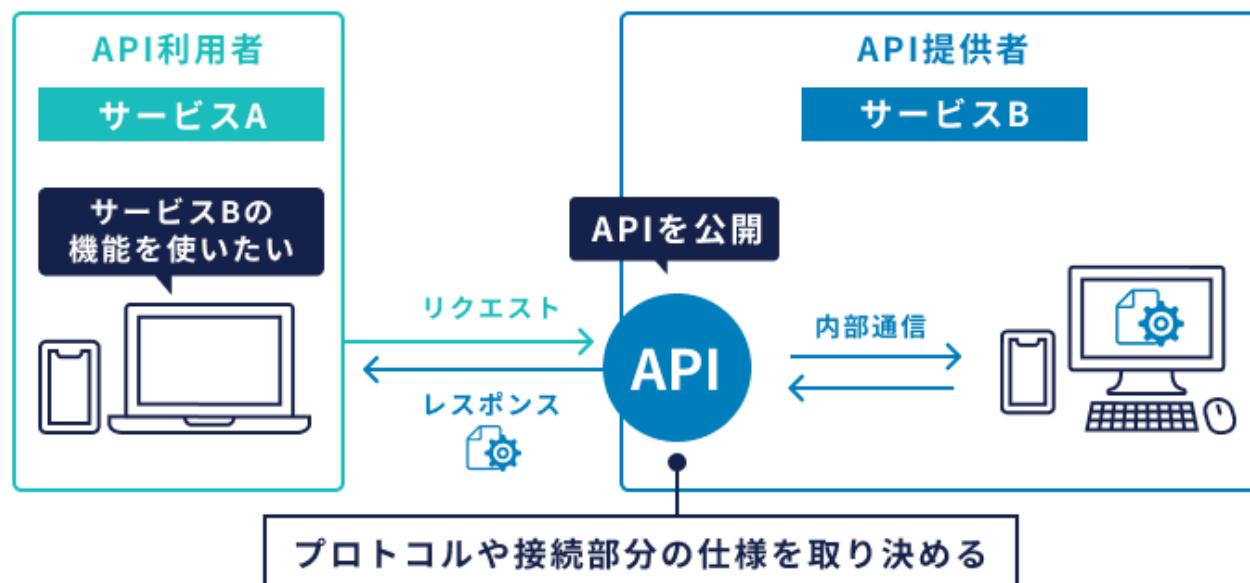
ChatGPT を使って AI ペットアドバイザーを作成

まず、一つ目のシステム連携では、ChatGPT と連携して、ペットの状態を考慮したアドバイスができる機能をつくっていきます。

ChatGPT との連携には、Open AI 社が提供している API を利用します。

API って何？

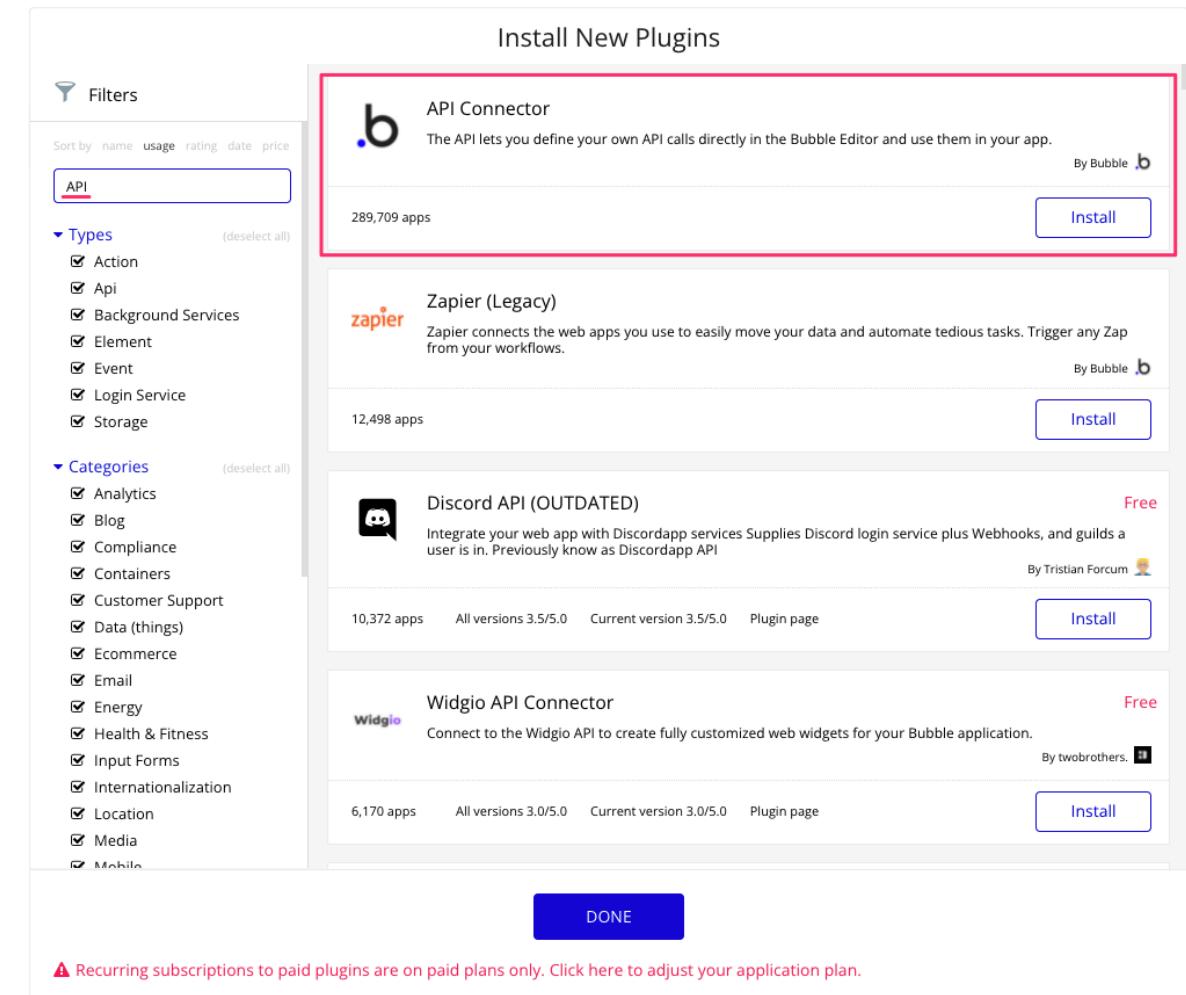
「API (Application Programming Interface)」のことで、ソフトウェアやアプリケーションが他のソフトウェアやサービスと通信するための方法です。API を通じて、アプリケーションはデータを共有したり、他のアプリケーションの機能を公開し、利用することができます。



Bubble での API 連携の準備

Bubble では、API と簡単に接続できる
ようにしてくれるプラグインが提供され
ています。準備していきましょう。

- 左メニューから Plugins を選択
し、Add plugins ボタンをクリック
- "API" で検索し、 API Connector
というプラグインをインストール
します



- このプラグインは Bubble のアプリから、世の中で公開されている API を使って、API の向こう側の情報と Bubble を繋げるためのプラグインとなります
- いくつかの Bubble のプラグインの中には、特定の API に特化させたものもあり、それらのプラグインはこの API Connector よりも設定が単純なものもあります
- 今回は一番の基本形となる API Connector プラグインを介して ChatGPT API や YouTube API を使い、機能を組み込んでいきます。

ChatGPT(Open AI) での API 連携の準備

Open AI の API を利用するには、以下の画面から Open AI のアカウントを作成し、支払方法の登録や API キーを発行をする必要があります。

<https://platform.openai.com/>

今回はそれらの手順は割愛して、事前に準備している API キーを Slack 共有しますのでそちらを利用していきましょう。

ただし、この場以外への公開やこの場でも使い過ぎないよう気を付けてください。（課金が爆発してしまうのでw）

参考 URL

ドキュメント

<https://platform.openai.com/docs/overview>

API リファレンス

<https://platform.openai.com/docs/api-reference/introduction>

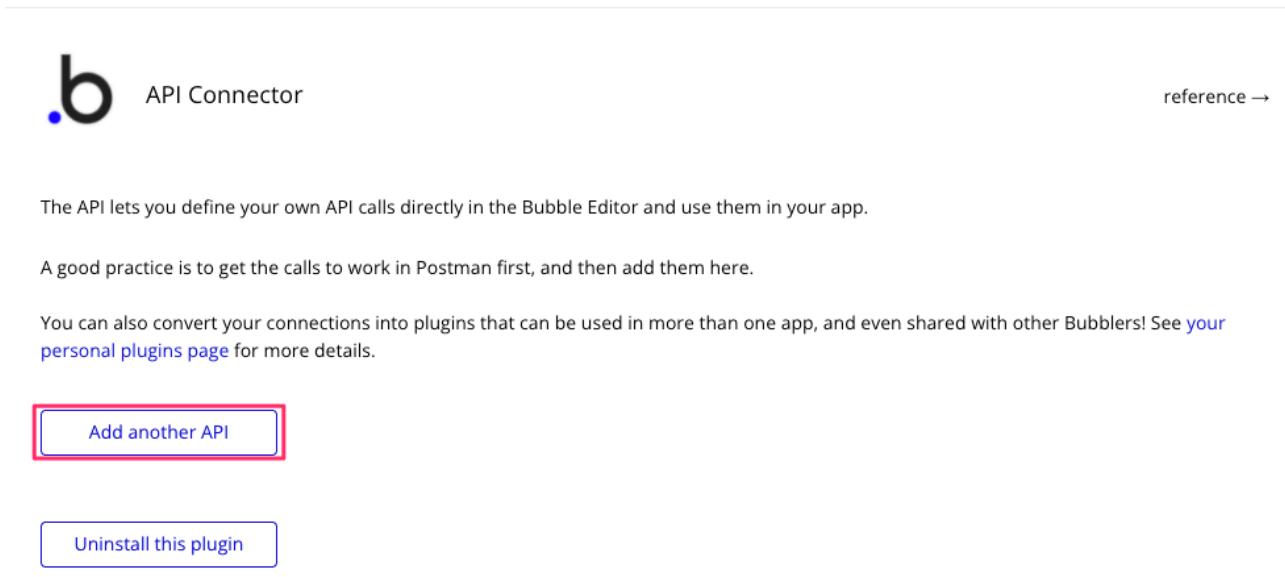
利用料金

<https://openai.com/api/pricing/>

管理画面

<https://platform.openai.com/settings/organization/projects>

- それでは ChatGPT API を使うための設定を行っていきます
- Plugins タブで Installed Plugins の中から "API Connector" を選択
- 右パネルにこのプラグインの概要が記載されているので、その下にある "Add another API" をクリックし、新しい API 連携の設定を開始します



- "Add another API" をクリックすると、このような API 設定部品が表示されると思います
- 各項目の簡単な説明と設定をしていきます

collapse

API Name Authentication delete

Shared headers for all calls

Add a shared header

Shared parameters for all calls

Add a shared parameter

expand

Name

Import another call from cURL

Add another call expand all calls

1. API Name: この API 連携のグループ名

- 今回は ChatGPT

The screenshot shows a configuration interface for an API group. The top section has an 'API Name' field (1) containing 'ChatGPT' and an 'Authentication' dropdown (2) set to 'None or self-handled'. A 'collapse' button is in the top right. Below this, there's a 'Shared headers for all calls' section (3) with a 'Add a shared header' button. Underneath is a 'Shared parameters for all calls' section (4) with a 'Add a shared parameter' button. At the bottom, there's a 'Name' field (5) containing 'chat/completions' with an 'expand' button to its right. A 'Import another call from cURL' section at the bottom left has an 'Add another call' button. An 'expand all calls' button is located at the bottom right.

1 API Name ChatGPT

2 Authentication None or self-handled

collapse

3 Shared headers for all calls

4 Shared parameters for all calls

5 Name chat/completions

expand

Import another call from cURL

Add another call

expand all calls

2. Authentication: このグループ配下の API で行う共通の認証方式

- 今回は "None or self-handled"

The screenshot shows the configuration interface for an API named "ChatGPT". The "Authentication" dropdown is set to "None or self-handled". Below it, there are sections for "Shared headers for all calls" and "Shared parameters for all calls", each with an "Add a shared header" and "Add a shared parameter" button respectively. A call definition for "chat/completions" is shown at the bottom. The interface includes "collapse" and "expand" buttons.

1 API Name ChatGPT

2 Authentication None or self-handled

3 Shared headers for all calls Add a shared header

4 Shared parameters for all calls Add a shared parameter

Name chat/completions

Import another call from cURL Add another call

expand

expand all calls

3. Shared headers for all calls: このグループ配下の API すべてで指定する共通のヘッダー情報

ChatGPT API では、ヘッダーに API キーをつけて接続する必要があります。キーを設定しましょう。

- Add a shared head をクリック
- Key に Authorization と入力
- Value に Bearer + (半角スペース)+ slack で共有した API キーを指定してください。

すべて入力したらこうなります。

API Name **1** ChatGPT Authentication **2** None or self-handled collapse

Shared headers for all calls

3 Key Authorization Value Bearer sk-proj-7zgAHEl4mtrLzZrjLF> remove

Add a shared header

Shared parameters for all calls

4 Add a shared parameter

Name **chat/completions** expand

Import another call from cURL

Add another call expand all calls

4. Shared parameters for all calls: このグループ配下の API すべてで指定する共通のパラメータ情報

- 今回は特に指定しません

The screenshot shows the configuration interface for an API named "ChatGPT". The interface includes fields for "API Name" (ChatGPT), "Authentication" (None or self-handled), and a "collapse" button. A "Shared headers for all calls" section contains a key-value pair: "Authorization" (Key) and "Bearer sk-proj-7zgAHEl4mtrLzZrjLF" (Value). A "Shared parameters for all calls" section contains a single entry: "chat/completions" (Name). Buttons for "Add a shared header" and "Add a shared parameter" are visible. At the bottom, there are buttons for "Import another call from cURL" and "Add another call".

API Name **1** ChatGPT Authentication **2** None or self-handled collapse

Shared headers for all calls

3 Key Authorization Value Bearer sk-proj-7zgAHEl4mtrLzZrjLF Delete

Add a shared header

Shared parameters for all calls

4 Add a shared parameter

Name chat/completions expand

Import another call from cURL

Add another call expand all calls

- 続いて、このグループに対して具体的な API の内容を設定していきます
- 先ほどの画面で "API Call" となっているブロックの右側に expand というリンクがあるのでそれをクリック
- すると、具体的な API の設定項目が表示されますので、簡単に説明します

The screenshot shows the configuration screen for a 'YouTube' API integration. At the top, there are fields for 'API Name' (set to 'YouTube'), 'Authentication' (set to 'None or self-handled'), and a 'collapse' button. Below these are sections for 'Shared headers for all calls' and 'Shared parameters for all calls', each with an 'Add a shared header' and 'Add a shared parameter' button respectively. A large section below is titled 'Name' with a field containing 'API Call'. To the right of this field is a red-underlined 'expand' link. At the bottom left is a 'Import another call from cURL' section with an 'Add another call' button, and at the bottom right is a 'expand all calls' link.

collapse

API Name YouTube Authentication None or self-handled

Shared headers for all calls

Add a shared header

Shared parameters for all calls

Add a shared parameter

Name API Call

expand

Import another call from cURL

Add another call

expand all calls

1. Name: 具体的な API の名前

- 今回は "chat/completions" と入力します

The screenshot shows a configuration interface for an API endpoint. The fields are labeled with red numbers:

- 1** Name: chat/completions
- 2** Method: POST
- 3** URL: https://api.openai.com/v1/chat/completions
- 4** Body (JSON object, use <> for dynamic values): A large text input field containing the number 1.

Other visible settings include "Use as Data", "Data type JSON", "Headers" (with an "Add header" button), "Body type JSON", and "Parameters" (with an "Add parameter" button).

2. Method: 接続時の http メソッド

- 今回は POST を選択します

The screenshot shows a configuration interface for an API endpoint. The fields are numbered as follows:

- 1 Name: chat/completions
- 2 Method: POST
- 3 URL: https://api.openai.com/v1/chat/completions
- 4 Body: (JSON object, use <> for dynamic values)

Below the main fields, there are sections for Headers (Add header), Body type (JSON), and Parameters (Add parameter). A large blue box highlights the Body input area.

3. Path: 具体的な API の URL

- 今回は `https://api.openai.com/v1/chat/completions` を入力します

1 Name chat/completions Use as Data Data type JSON

2 POST 3 https://api.openai.com/v1/chat/completions (use [] for params)

Headers Add header

Body type JSON

Parameters Add parameter

4 Body (JSON object, use <> for dynamic values)

1

4. Body: リクエストの本文です

- 今回は後述の内容をまず入力してください。

The screenshot shows a configuration interface for a request. The fields are labeled as follows:

- Name:** chat/completions (labeled 1)
- Method:** POST (labeled 2)
- URL:** https://api.openai.com/v1/chat/completions (labeled 3)
- Headers:** Add header
- Body type:** JSON
- Parameters:** Add parameter
- Body:** Body (JSON object, use <> for dynamic values) (labeled 4)

```
{  
  "model": "gpt-4o-mini",  
  "messages": [  
    {  
      "role": "system",  
      "content": "You are a helpful assistant."  
    },  
    {  
      "role": "user",  
      "content": "Hello!"  
    }  
  ]  
}
```

これは、ChatGPT API のドキュメントに記載されているサンプルリクエストの内容になります。

<https://platform.openai.com/docs/api-reference/chat>

すべて入力したらこうなります。

Name chat/completions Use as Data ▾ Data type JSON ▾

POST https://api.openai.com/v1/chat/completions See reference

Headers Add header

Body type JSON

Parameters Add parameter

Body (JSON object, use <> for dynamic values)

```
1 {  
2   "model": "gpt-4o-mini",  
3   "messages": [  
4     {  
5       "role": "system",  
6       "content": "You are a helpful assistant."  
7     },  
8     {  
9       "role": "user",  
10      "content": "Hello!"  
11    }  
12  ]  
13 }
```

Include errors in response and allow workflow actions to continue

Capture response headers

- 設定が終わったら、設定内容が正しいかを確認します
- 下部にある "Initialize call" ボタンをクリック
- 成功すると "Returned values - search" というポップアップが表示されるはずです

Returned values - chat/completions

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

id	chatmpl-AW3RSDySD5t7QzbYw7LCiswpUZywo	<input type="text"/>
object	chat.completion	<input type="text"/>
created	1732203090	<input type="number"/>
model	gpt-4o-mini-2024-07-18	<input type="text"/>
choices (list) (see fields below)		<input type="button" value="chat/completions choice"/>
index	0	<input type="number"/>
message role	assistant	<input type="text"/>
message content	Hello! How can I assist you today?	<input type="text"/>
message refusal		<input type="text"/>

SAVE

Cancel

- これは今回設定した "<https://api.openai.com/v1/chat/completions>" の API を実行した結果（レスポンス）の情報を表しています。
- Bubble の API Connector ではこのように、実際にリクエストを送信してみて、受け取った結果を解析してこのように Bubble 上で使いやすいように設定できるようになっています。

Returned values - search

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

kind	youtube#searchListResponse	<input type="text"/>
etag	mjrgGMgtIMWG6aVWolrNtjmgG8	<input type="text"/>
nextPageToken	CAUQAA	<input type="text"/>
regionCode	US	<input type="text"/>
pageInfo totalResults	946629	<input type="number"/>
pageInfo resultsPerPage	5	<input type="number"/>
items (list)	(see fields below)	<input type="text"/>
kind	youtube#searchResult	<input type="text"/>
etag	WahfIcVNgyipX-l3rjRiPUrHi10	<input type="text"/>
id kind	youtubee#channel	<input type="text"/>
id channelId	UCcRigzI_jBAZh_UySjv8xuw	<input type="text"/>
id videoId	S094aNshTSU	<input type="text"/>

SAVE **Cancel**

詳しくは割愛しますが、ポイントとしては下記 3 つです。

- `choices(list)` となっている部分が回答メッセージが格納されている一覧になります
- その型となるのが `chat/completions choice` という型となることを表しています
- 回答内容が `message content` となっている項目です

参考：chat completion object のリファレンス

<https://platform.openai.com/docs/api-reference/chat/object>

- 特に参照しない項目については紛らわしいのでプルダウンの中から **Ignore field** を選択しておきます
- こうすることで、Dynamic data として扱う時に、選択肢の中に表示しないようになり、設定項目を選ぶ際に悩まずに済みます
- 設定が完了したら "SAVE" をクリックして、設定内容を保存しておきます

Returned values - chat/completions

id	chatcmpl-AW3RSDySD5t7QzbYw7LCiswpUZywo	Ignore Field
object	chat.completion	Ignore Field
created	1732203090	Ignore Field
model	gpt-4o-mini-2024-07-18	Ignore Field
choices (list) (see fields below)		chat/completions choice
index	0	Ignore Field
message role	assistant	Ignore Field
message content	Hello! How can I assist you today?	text
message refusal (no sample data)		Ignore Field
logprobs (no sample data)		Ignore Field

続いて、リクエストの本文を指定していきます

リクエストは JSON という構造化された書式で記載されています。各項目は以下の意味を持ちます。

- `model` : 利用するモデルです。GPT はサイズや性能の異なる複数のモデルが提供されています。今回は軽量で安価な `gpt-4o-mini` を使って検証してみます。
- `messages` : GPT にコンテキストとして与えるメッセージ群になります。GPT は、大雑把に言うと文書の続きをとても上手く書くことができる言語モデルです。続きを元となるコンテキストを `role` (役割) と `content` (内容) のセットで記載して、連ねていくことでそのコンテキストに沿ったつづき(回答)を返してくれます。

与えるコンテキストはチャットに対する指示にあたるため、プロンプトと呼ばれています。

参考 : create chat completion のリファレンス

<https://platform.openai.com/docs/api-reference/chat/create>

今回はペットの登録内容と買い主の相談メッセージに基づいて、アドバイスを返してくれる機能を組み込みます。

Body に記載する内容は、`<name>` のように `<>` で括って名前をつけて埋め込んで置くことで、後ほど API を呼び出す際に Bubble から中身を埋め込むことができます。

以下のようにコンテキストを与えることとします。

```
{  
  "model": "gpt-4o-mini",  
  "messages": [  
    {  
      "role": "system",  
      "content": "ペットの飼育アドバイザーとして回答してください。  
      ペットの名前、生年月日、カテゴリー、最新の体重、最新の体重の計測日が  
      飼い主であるユーザーから伝えられます。また、さらに飼い主からペットに  
      対する相談内容も伝えられます。それらに基づいて、ペットの飼育への  
      アドバイスを回答してください。なお、回答の冒頭で確認のために、  
      与えられた情報である名前、生年月日、カテゴリー、最新の体重、  
      最新の体重の計測日、相談内容に触れるようにしてください."  
    },  
    {  
      "role": "user",  
      "content": "ペットの名前：<name>、ペットの生年月日：<birthday>、  
      ペットのカテゴリー：<category>、ペットの最新の体重：<weight>、  
      ペットの最新の体重の計測日：<weightDate>、相談内容：<content>"  
    }  
  ]  
}
```

*上記の内容は、改行をいれて整形しているため、そのまま貼り付けても利用できません。

以下を Body に貼り付けてください。

```
{
  "model": "opt-4o-mini",
  "messages": [
    {
      "role": "system",
      "content": "ペットの飼育アドバイザーとして回答してください。ペットの名前、生年月日、カテゴリー、最新の体重、最新の体重の計測日が飼い主であるユーザーから伝えられます。また、さらに飼い主からペットに対する相談内容も伝えられます。それらに基づいて、ペットの飼育へのアドバイスを回答してください。なお、回答の冒頭で確認のために、与えられた情報である名前、生年月日、カテゴリー、最新の体重、最新の体重の計測日、相談内容に触れるようにしてください。"
    },
    {
      "role": "user",
      "content": "ペットの名前: <name>、ペットの生年月日: <birthday>、ペットのカテゴリー: <category>、ペットの最新の体重: <weight>、ペットの最新の体重の計測日: <weightDate>、相談内容: <content>"
    }
  ]
}
```

すると、<>で囲んだ項目がBodyの入力エリアの下部に並びます。Privateのチェックをはずしてください。こうすることで、後ほどBubbleで呼び出す際に動的に指定できます。

Body (JSON object, use <> for dynamic values)

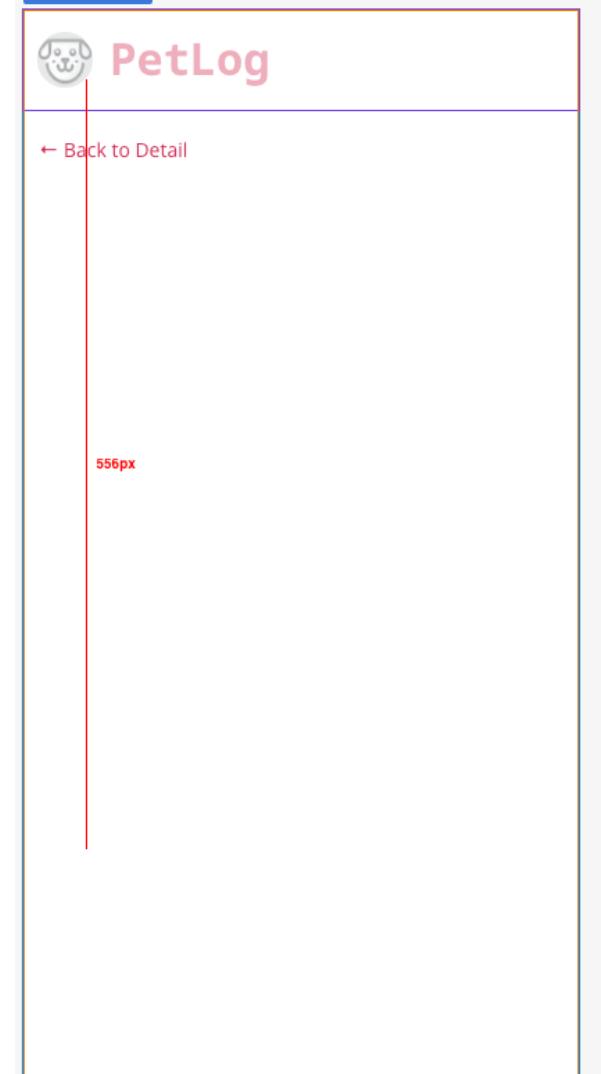
```
2 "model": "gpt-4o-mini",
3 "messages": [
4   {
5     "role": "system",
6     "content": "ペットの飼育アドバイザーとして回答してください。  
ペットの名前、生年月日、カテゴリー、最新の体重、最新の体重の計測日が  
飼い主であるユーザーから伝えられます。  
また、さらに飼い主からペットに対する相談内容も伝えられます。  
それらに基づいて、ペットの飼育へのアドバイスを回答してください。  
なお、回答の冒頭で確認のために、与えられた情報である名前、生年月日、  
カテゴリー、最新の体重、最新の体重の計測日、相談内容に  
触れるようにしてください。"
7   },
8   {
9     "role": "user",
10    "content": "ペットの名前 : <name>、ペットの生年月日 : <birthday>、  
ペットのカテゴリー : <category>、ペットの最新の体重 : <weight>、  
ペットの最新の体重の計測日 : <weightDate>、相談内容 : <content>"
11  }
12 ]
13 }
```

Key	Value	Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
name		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
birthday		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
category		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
weight		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
weightDate		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>
content		Private	<input type="checkbox"/>	Allow blank	<input type="checkbox"/>

画面に組み込んでみましょ
う。

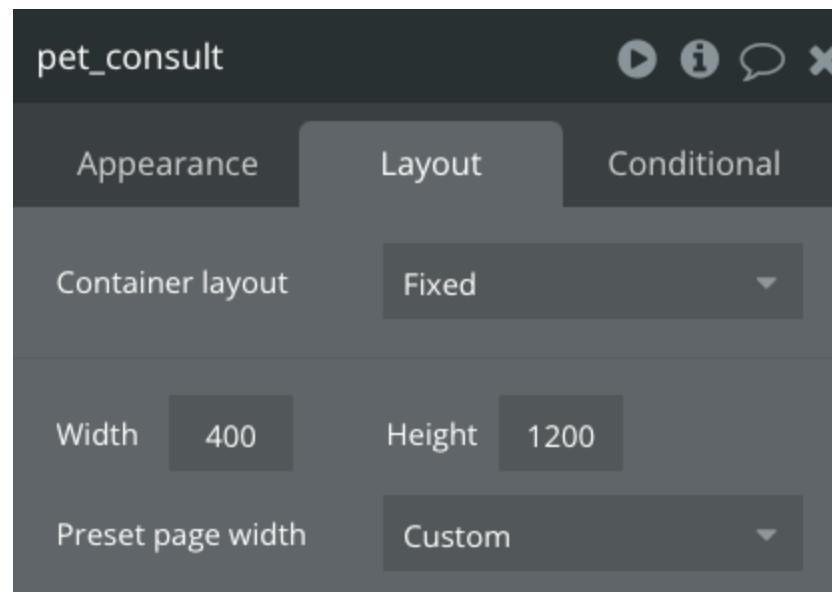
pet_weight 画面を元に新しくページを
作ります。

- b ロゴ右側のメニューを開き、
`Add a new page`
- `Page name` に `pet_consult` と入
力
- `Clone from` に `pet_weight` を選
択して、`Create` ボタンをおす。
- 画面が作成されたら、グラフ以下
の要素を全部削除する。



GPT の回答は長文になるので、全て表示しきれるよう画面の縦幅を大きめにとっておきましょう。（この画面はレスポンシブの設定をしていないので、伸縮してくれません）

- 画面の外枠をクリックして、`pet_consult` の設定ウィンドウを呼び出す。
- `Height` に `1500` を指定する。





[← Back to Detail](#)

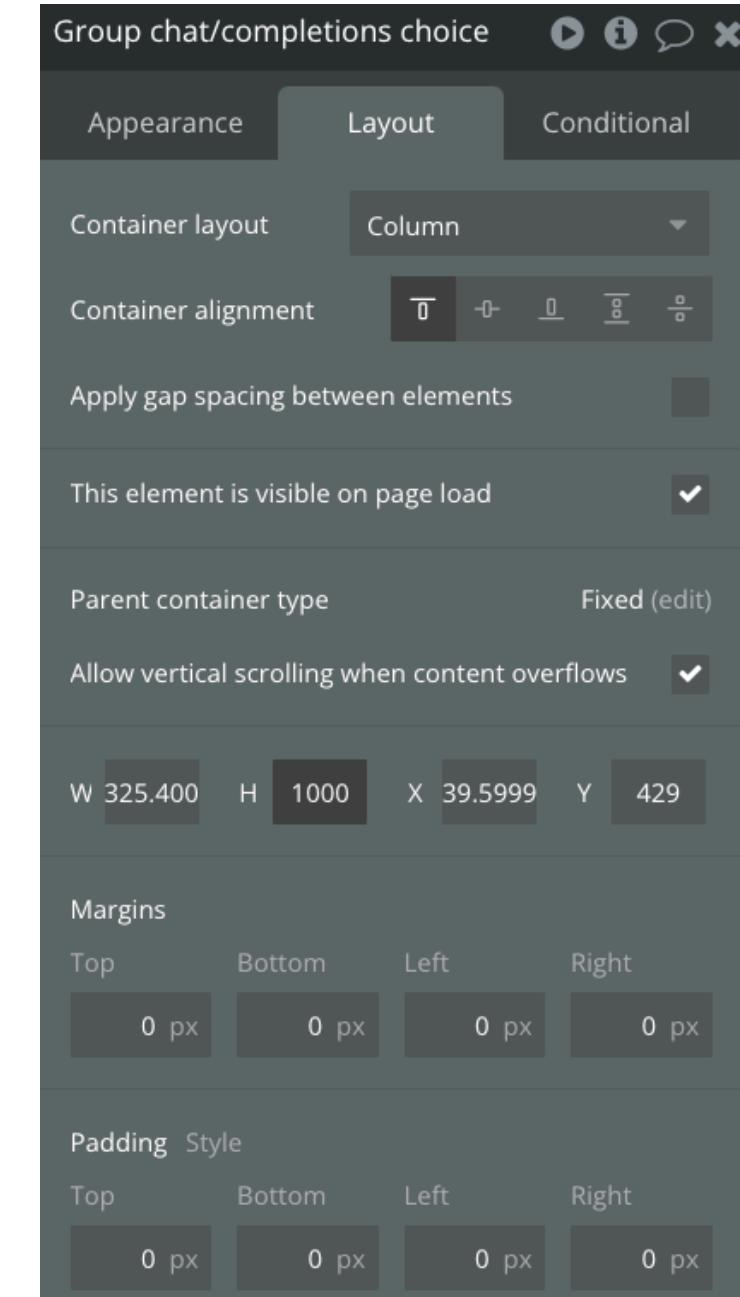
次に入力用の要素を配置していきます。
右のように要素を配置してみましょう。

- 説明文のテキスト
- メッセージ入力用のインプット
- 送信用のボタン
- 回答を表示するためのテキスト

やり方はわかりますね。

Feel free to ask about anything.
A pet care advisor will help address your concerns.

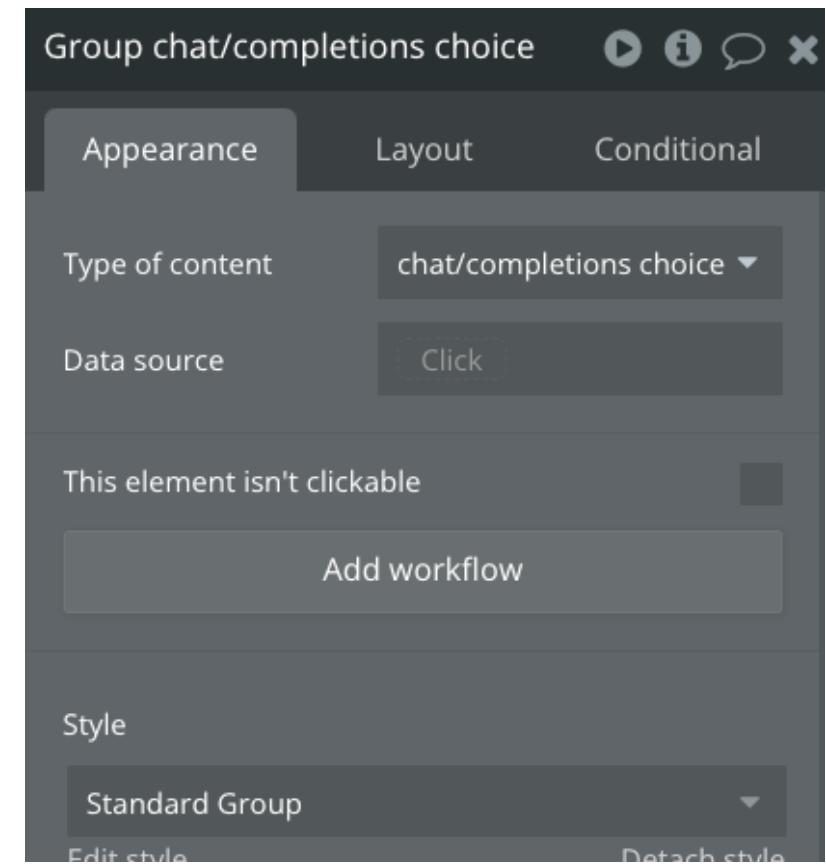
Send



また回答出力用の要素を配置していきます。回答を受けとるためには Type を定められる `Group` や `RepeatingGroup` といった Container 要素で受ける必要があります。

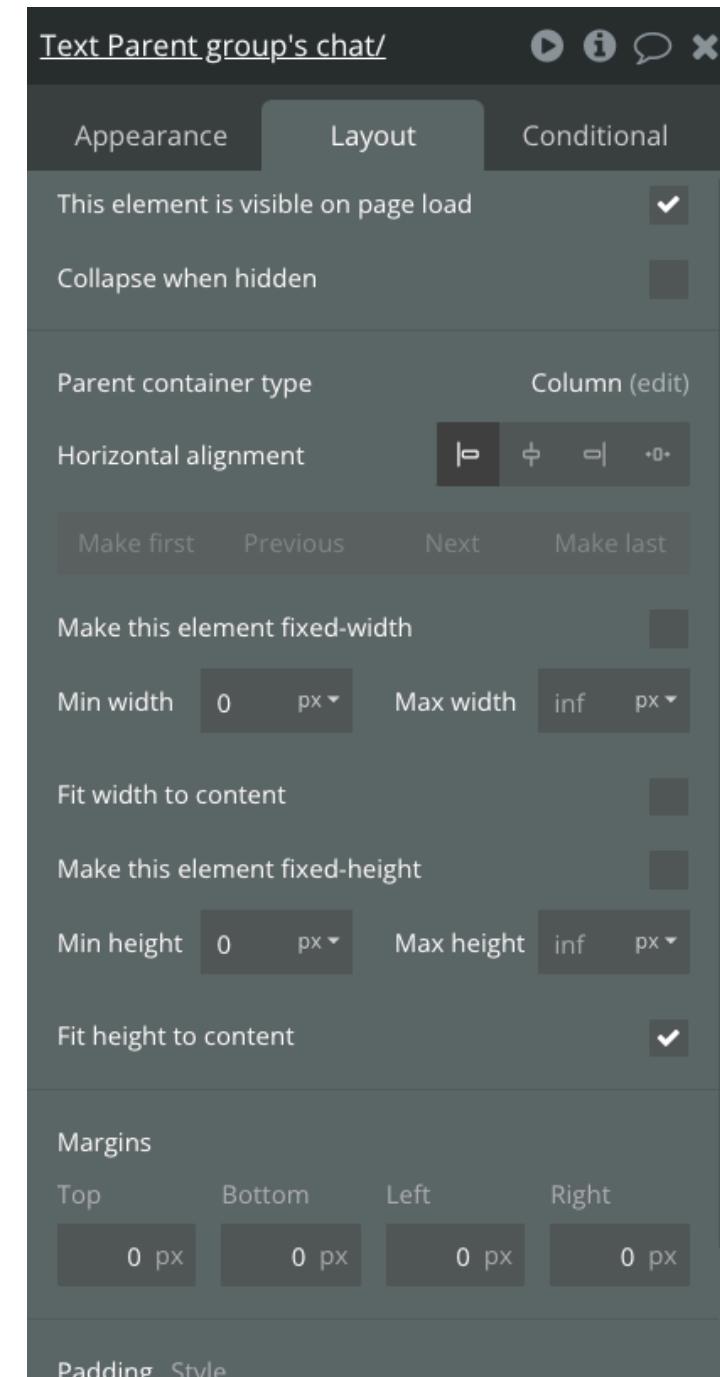
今回は繰り返し表示は必要ないので、`Group` を配置しましょう。長文の回答を表示しきれるように、`H` を `1000` にしておいてください。

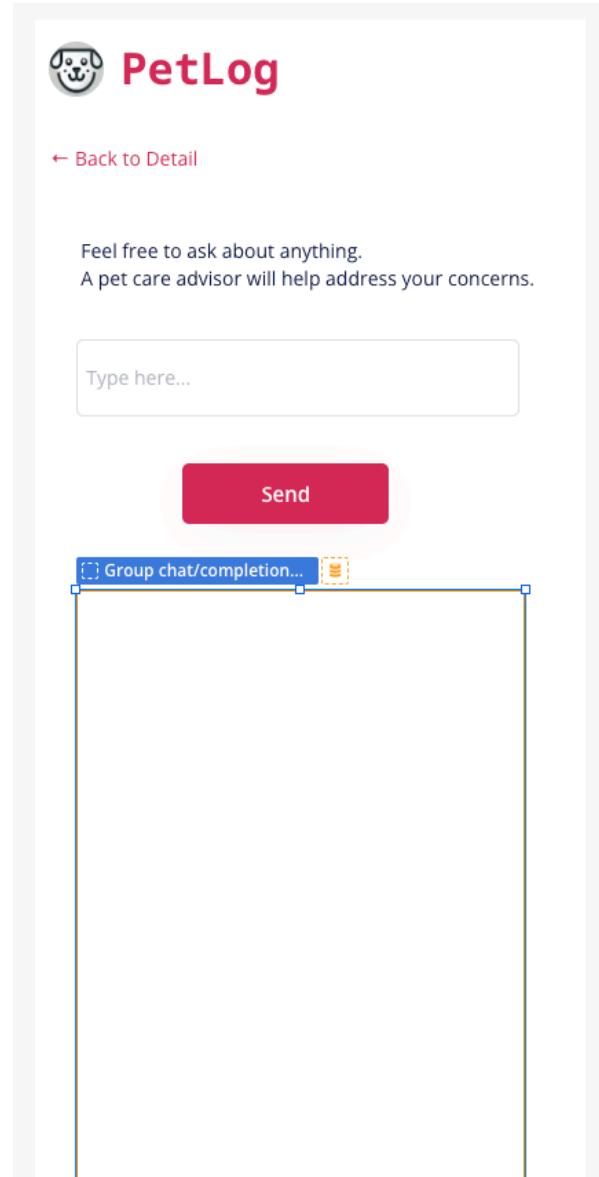
Type を指定しておいてください。先ほど API の設定で定めた、
chat/completions choice という
Type を指定します。



今度は回答のテキストを表示できるよう
に、作成した Group の中にテキストを
配置しましょう。Group のサイズ内い
っぱいままで表示できるよう、右のように
設定してください。

ポイントは、幅や高さは固定しない、高
さはコンテンツに合わせるという設定に
しているところになります。



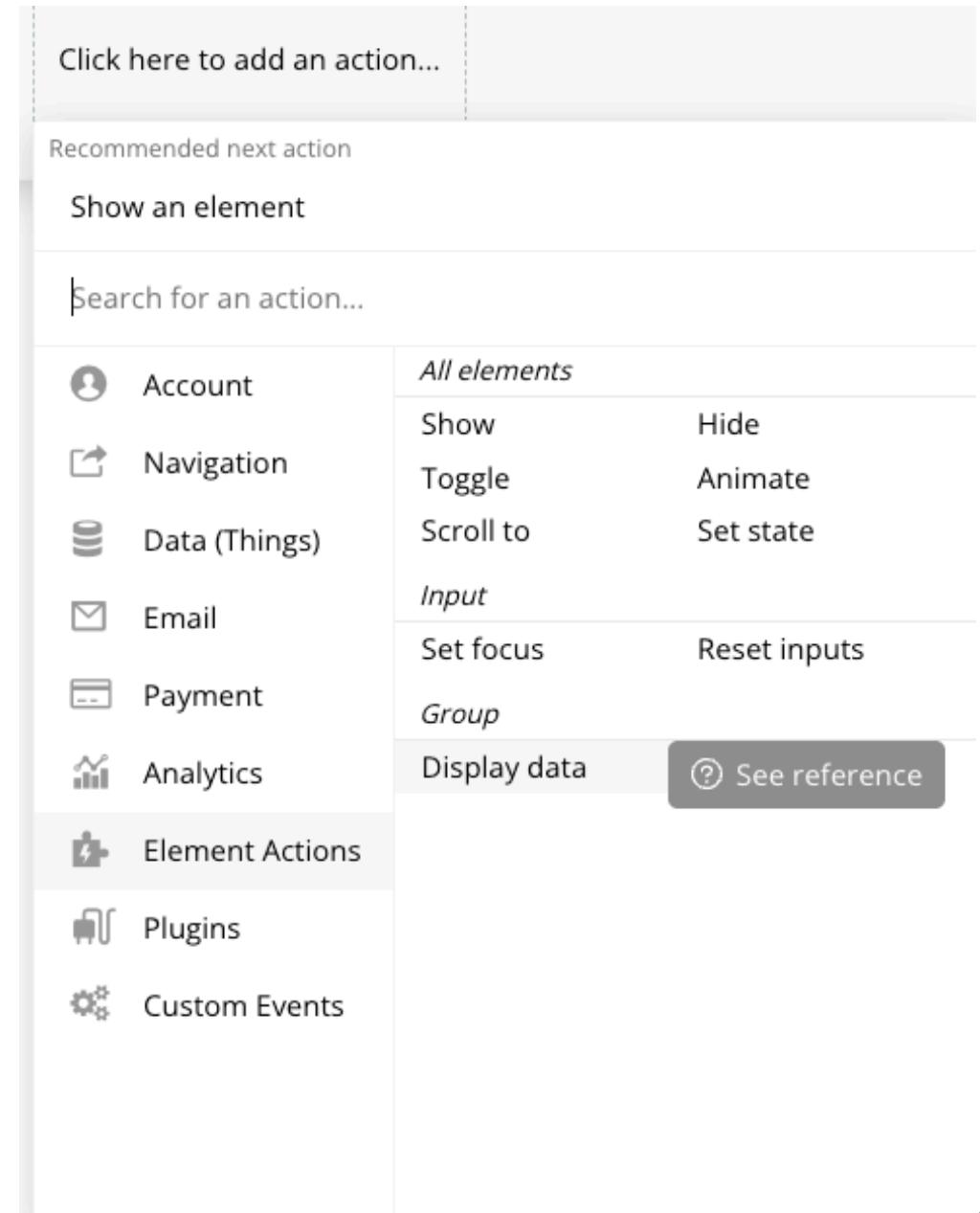


このような見た目になります。

では、ボタンを押したときの動作を設定しましょう。

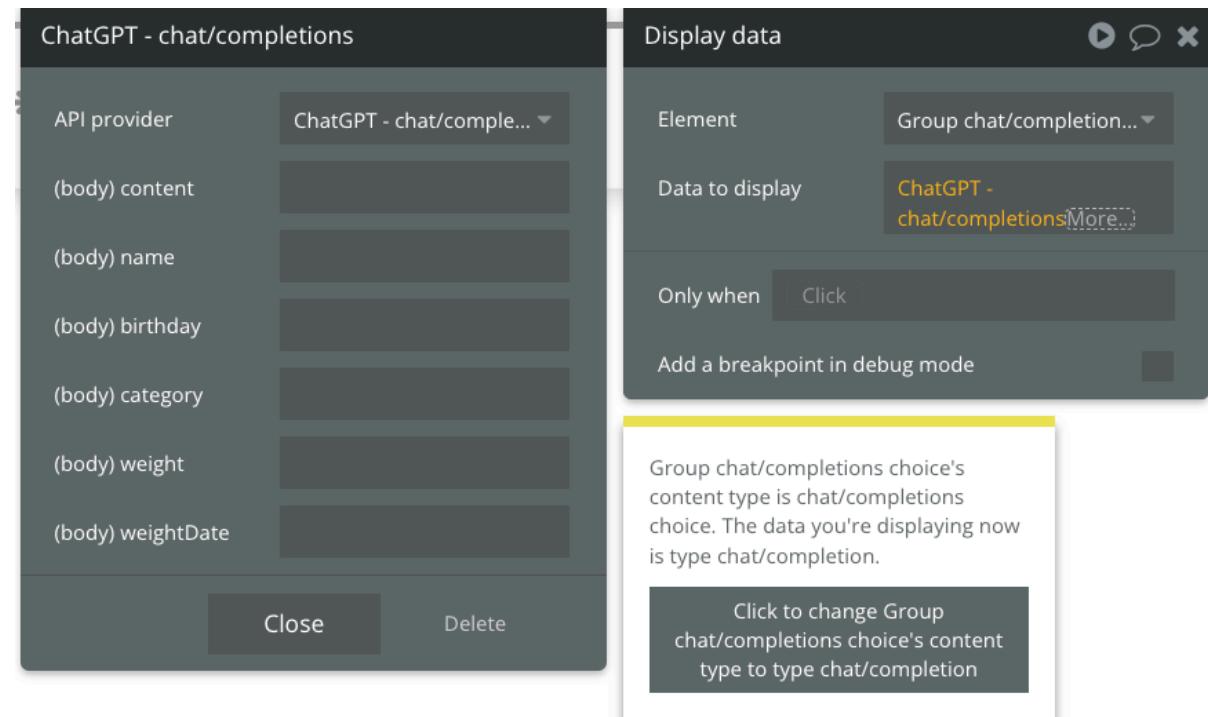
- Send ボタンをクリックし、設定ウィンドウを開く
- Appearance タブの中の Add workflow ボタンを押す
- Workflow の設定画面で Click here add an action... を押す
- Element Actions の Display data を選択する

要素に対して取得したデータを表示するという action になります。

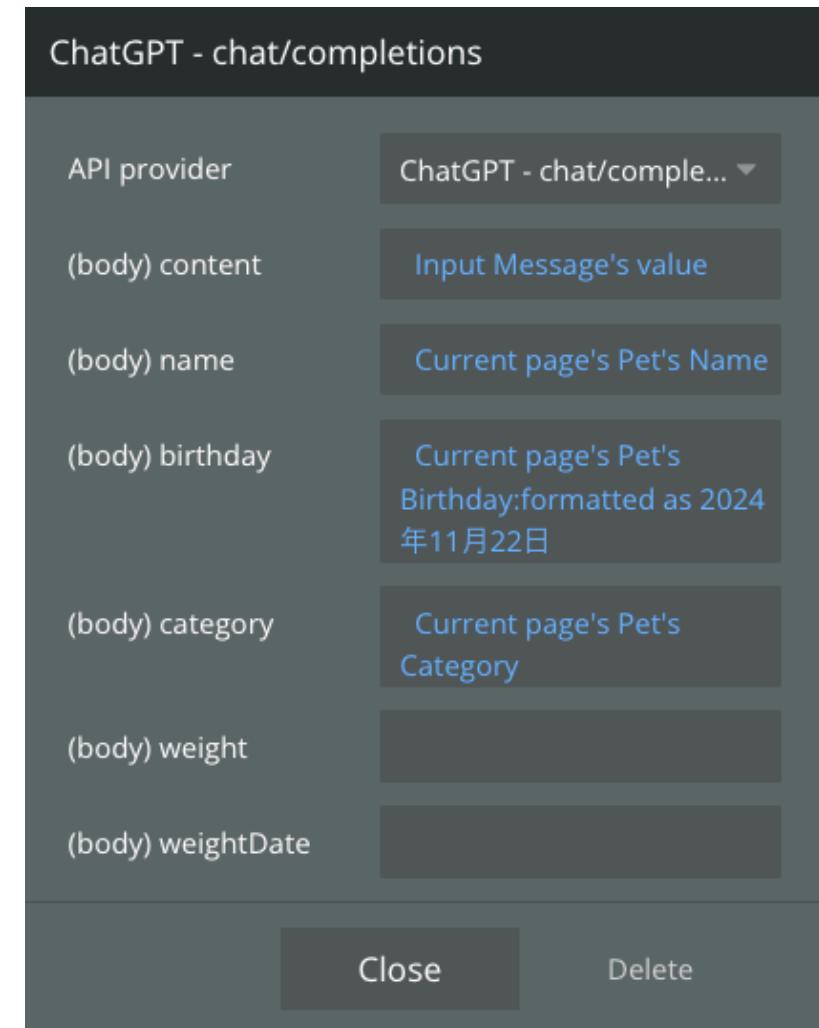


Display data にて、画面の内容を元に API から回答を取得する設定をいれていきます。

- Element に Group chat/completions choice を指定する。先ほど画面に配置した Group ですね。
- Data to display で Get data from an external API を指定する。外部 API からデータを取得するという指定になります。
- API provider に ChatGPT – chat/completions を指定する。前準備で API Connector で作成した設定ですね。

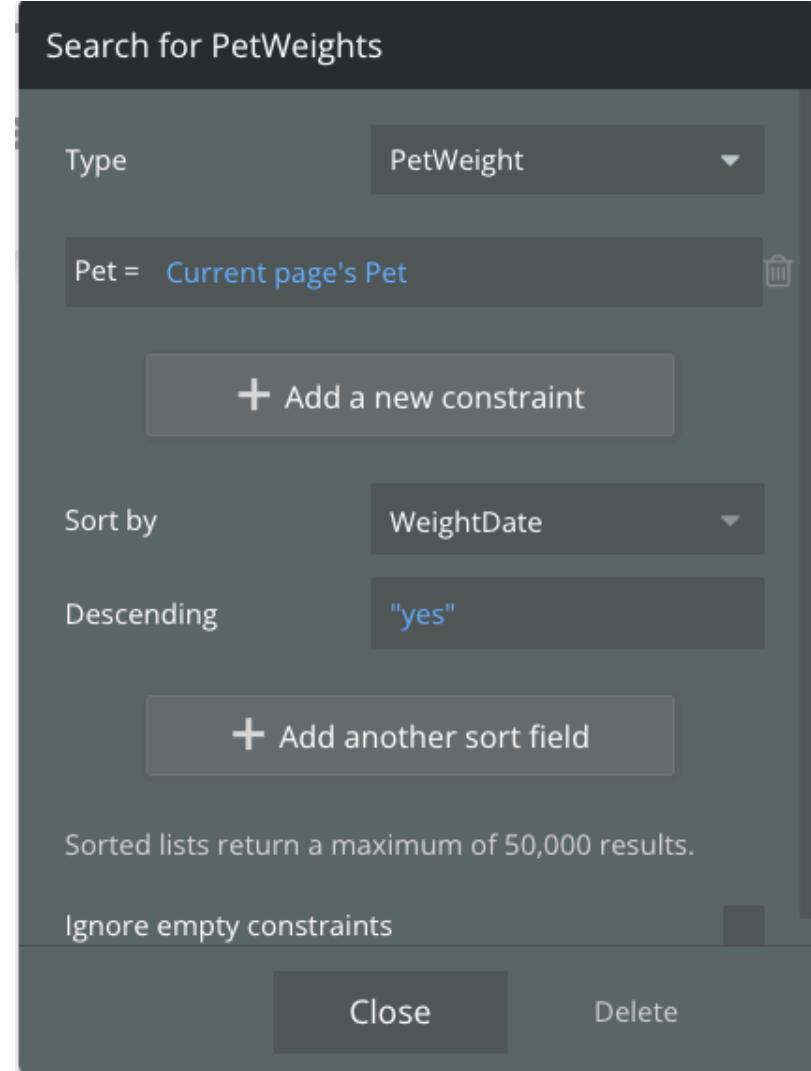


次に画面の情報を指定していきます。右の様に設定してください。



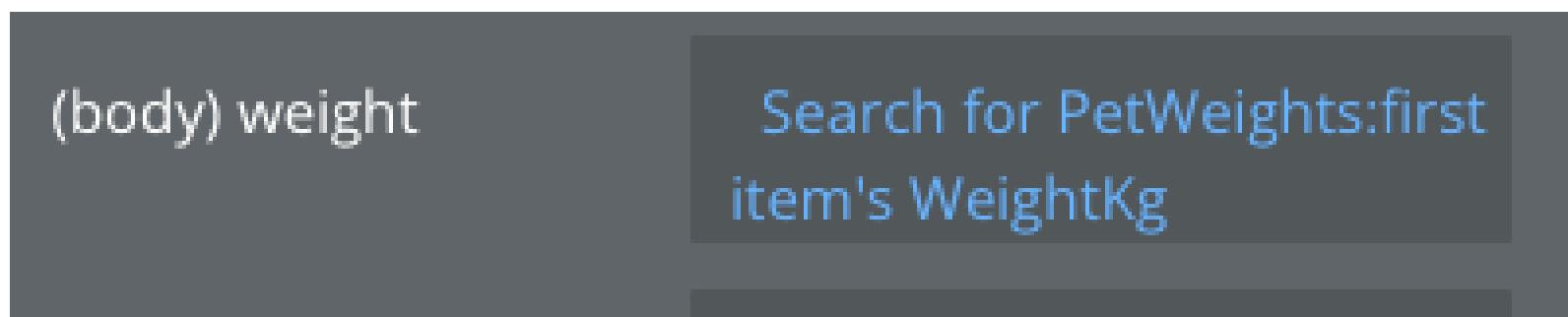
`weight` と `weightDate` については、
`PetWeight` のデータの中から検索して
くる必要があります。

- `weight` の欄で `Do a search for` を選択
- `Type` に `PetWeight` を指定
- `Add a new constraint` を押し、
`Pet``=``Current page's Pet` を指定。
- `Sort by` で `WeightDate` を指定し、`Descending` に `yes` を指定する。
- `Close` する。



表示しているペットの体重を新しい順に取得しているので、その中の最初の体重を採用する
ようにします。

- `:first item` を指定します
- `'s WeightKg` を指定します。



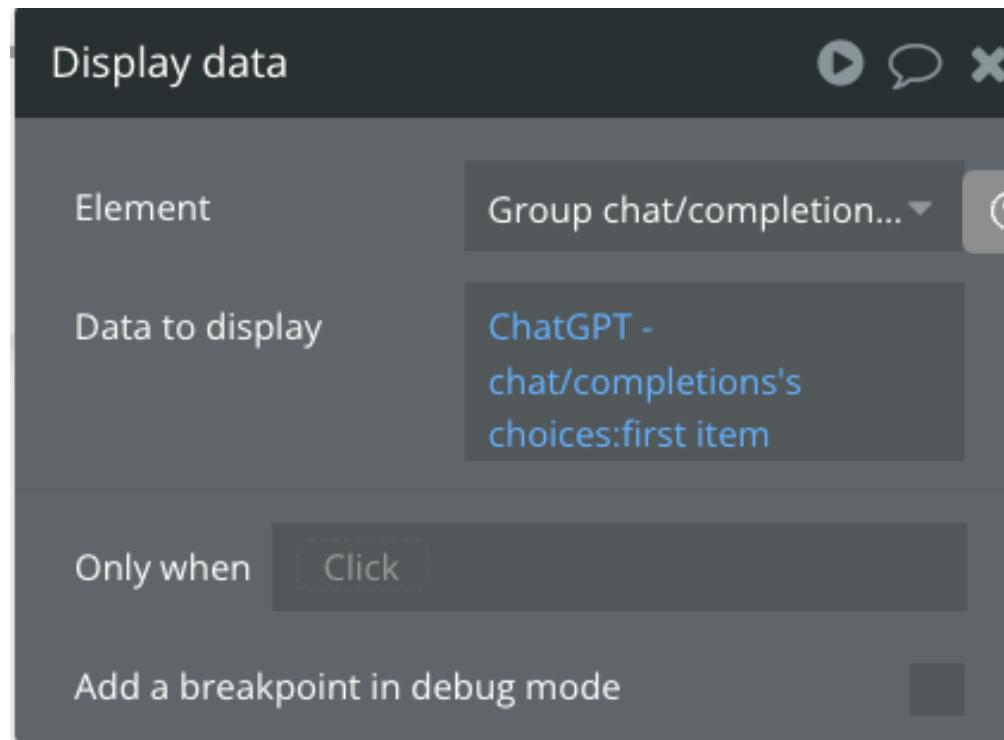
同様にして、`weightDate` は最新の体重の日付けを取得するように設定してください。

(body) `weightDate`

Search for PetWeights:first
item's WeightDate

Display data のウィンドウに戻り、「`s.choices``:first item」を指定します。

ChatGPT では、複数回答から選択することができるような使い方があるため複数とれるような構造になっていますが、今回の用とでは1つしか回答を取得しないので、1つ目を指定します。

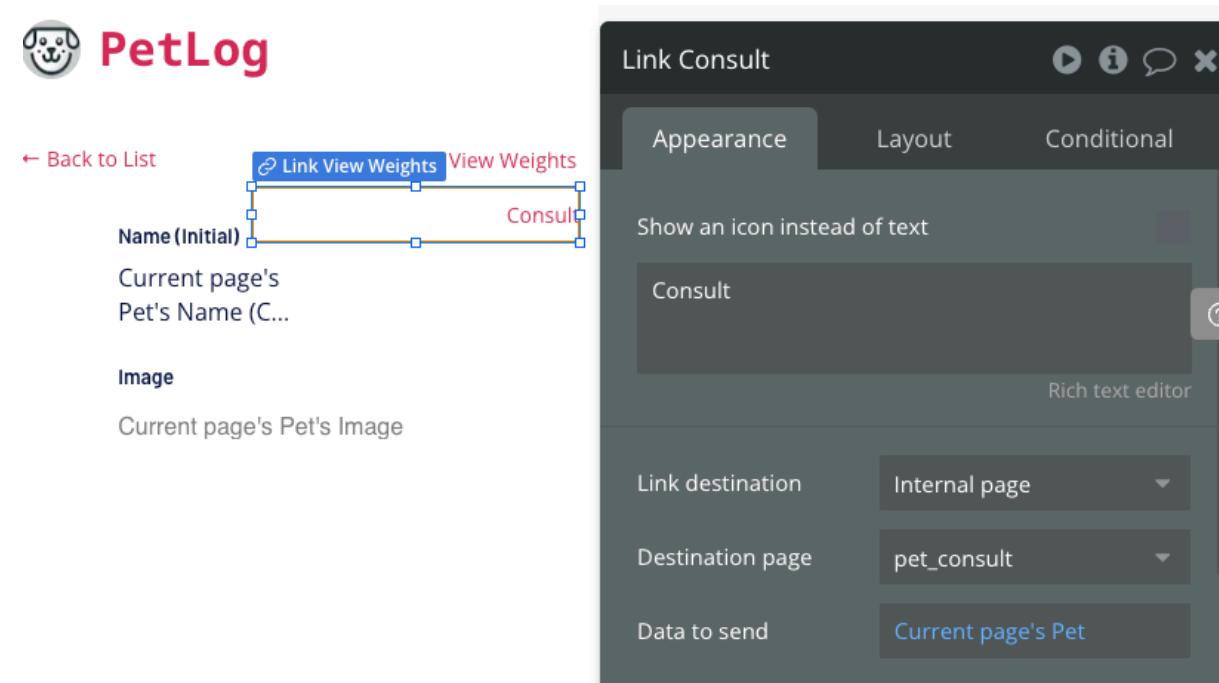


これで表示の準備は整いました。

最後に、ペット詳細からのリンクを配置しましょう。

- Design ビューで pet_detail を開く
- View Weights のリンクをコピーして下に配置し、リンク名を Consult にする。
- 遷移先を pet_consult に変更する

準備完了です。



[Log out](#)[← Back to Detail](#)

Feel free to ask about anything.
A pet care advisor will help address your concerns.

最近ご飯をあまり食べないのですが。。。

[Send](#)

動作確認してみましょう。

こんにちは！ご相談いただきありがとうございます。ペットの名前はtarōで、生年月日は2019年11月11日、カテゴリーは犬、最新の体重は10.5キログラム、体重の計測日は2024年11月21日ですね。また、「最近ご飯をあまり食べない」という相談内容ですね。

まず、食欲の低下はさまざまな理由が考えられます。以下の点を確認してみてください：

1. ****健康状態の確認**:** 最近の行動や体調に変化はありませんか？嘔吐、下痢、元気がないなど、健康に関する兆候が見られる場合は、獣医の診断を受けることをお勧めします。

2. ****食事の内容を見直す**:** 餌の種類やブランドを変えたことがありますか？新しい食事が好きではない場合や、古くなったフードを与えていた場合は、食べなくなることがあります。

3. ****環境の変化**:** 最近何か大きな環境の変化はありませんか？引っ越しや家族の変化、他のペットとの関係性、影響オブジェクトがおりません。

以上で、ChatGPTとの連携については完了です。

ChatGPTなどのLLMはプロンプトを工夫すれば様々な振る舞いができるので、ノーコードとツールとうまく組み合わせれば、簡単に強力な機能をつくれることがあります。

また、取得した回答をデータベースに格納して履歴参照したり、会話を続けたりといったこともできます。

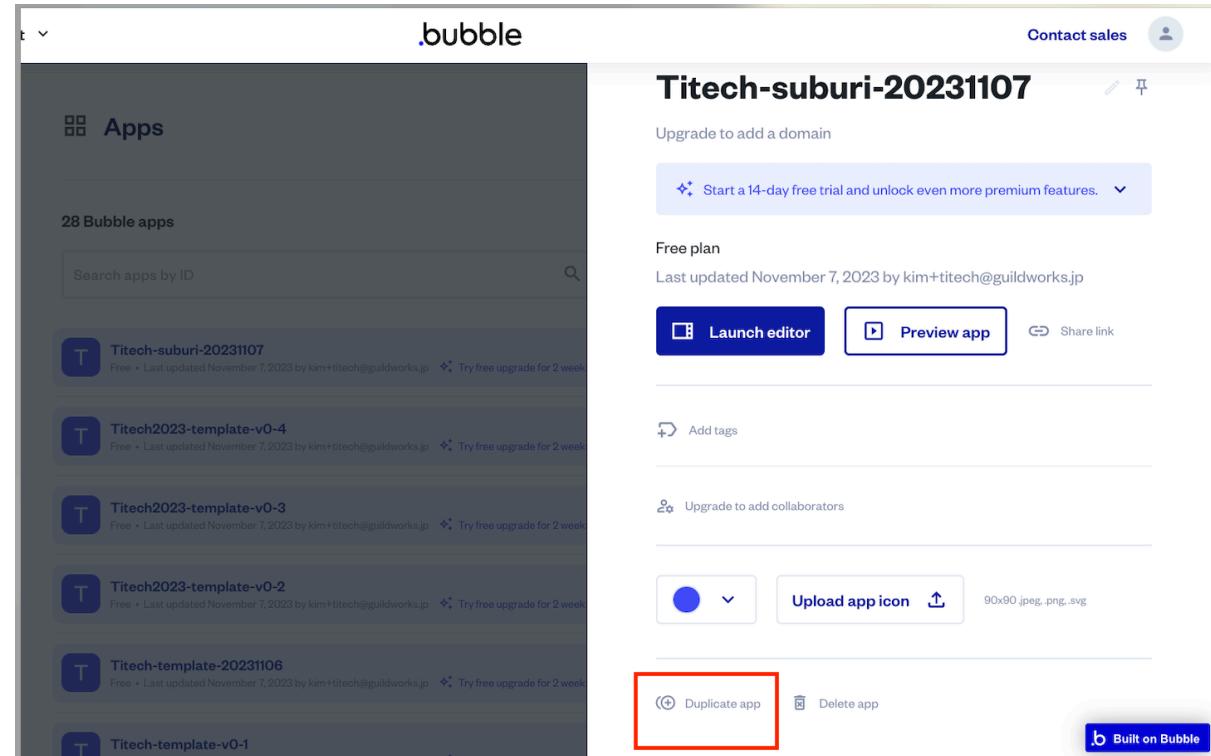
いろいろなアイデアが手軽につくれて楽しいので、ぜひやってみてください。

YouTube の動画一覧を表示

まずは土台となるアプリケーションを新たに用意

- 本日の前半で使ったアプリケーションをコピーして使い回したいと思います
- <https://bubble.io/home?tab=apps> にアクセス

- 各自のアカウントで作成している Bubble アプリの一覧が表示されます
- その中で、先ほどまで作ってきた アプリケーションを選択して、 ポップアップの最下部にある **Duplicate app** をクリック



- コピー後のアプリケーション名の入力ダイアログが表示されるので、適当な名前を入力
- 以前のデータは使い回さないので **Copy the application database content** のチェックは OFF のまま
- COPY をクリック

[Back to main menu](#)

Copy app

Name of this copy of titech-suburi-20231107

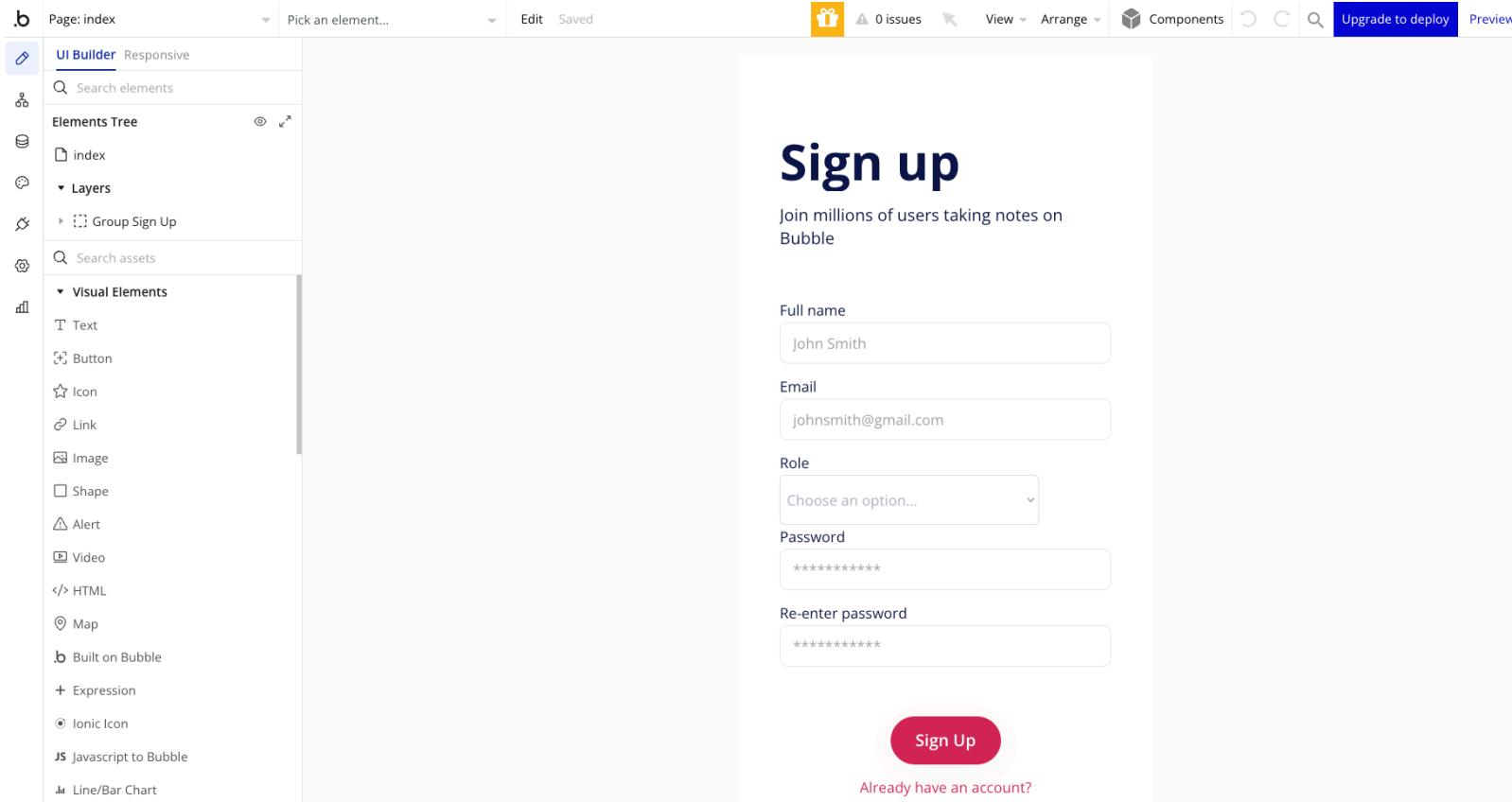
bubble-advanced

You can add a domain name later.

Copy the application database content

Copy

- コピーが終わると、コピーしたアプリケーションの編集画面（index）が表示されればOKです



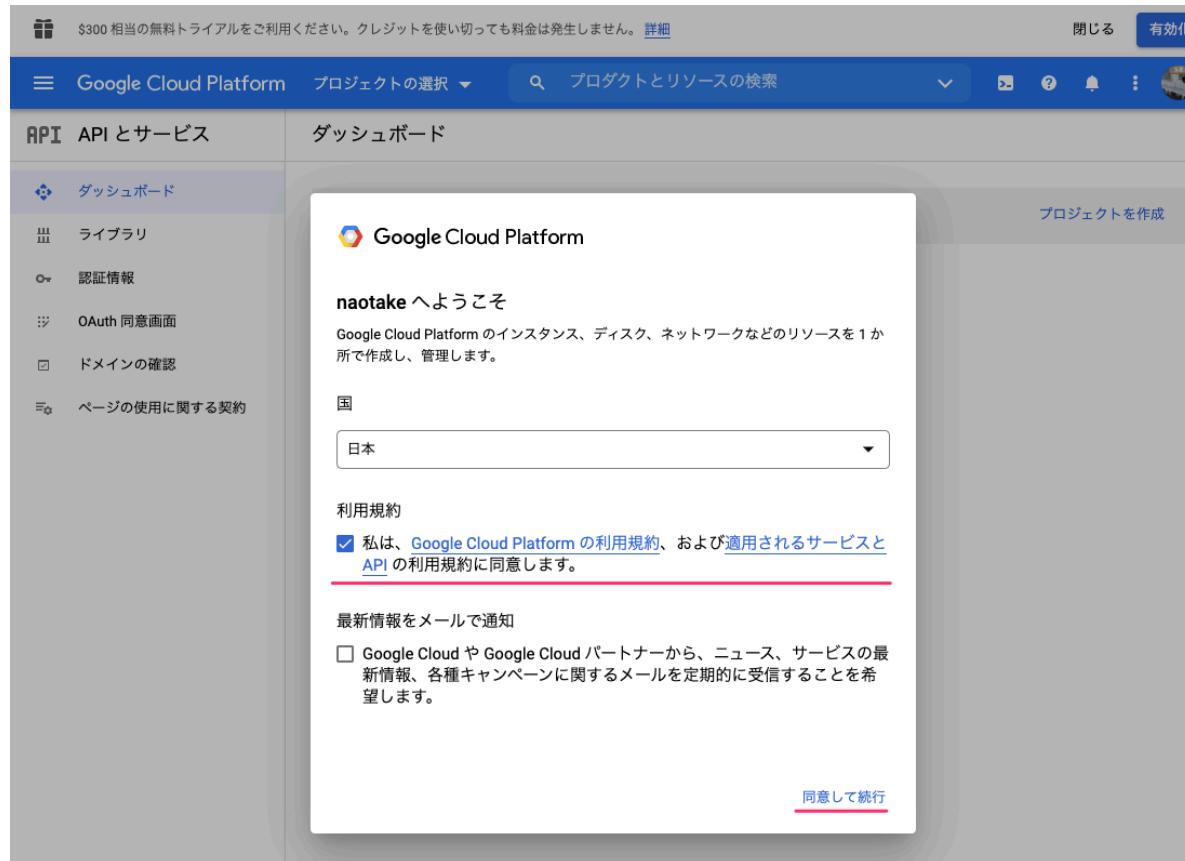
YouTube 連携の事前準備

- 続いて、YouTube の動画一覧を表示するために必要な下準備を行います
- 今回の講義では Google が提供しているクラウドサービス "Google Cloud Platform" に用意されている各種 API を使っていきます
 - コンピューティングやストレージ、データベース、ネットワークなど開発に必要なあらゆるツールやインフラがサービスとして提供されているプラットフォームです。
 - 他にも、Amazon が提供している "Amazon Web Service" (AWS)、Microsoft が提供している Azure といった有名なものがあります。
 - (余談ですが、これらのサービスが世に現れたことで自前のサーバーが要らなくなり、世の中のソフトウェア開発の敷居は一気に下がったのです)

- 下記 URL にアクセス
 - <https://console.developers.google.com/>
- ご自身の Google アカウントでログインしてください
 - もし、Google アカウントをお持ちでない方は、アカウントの作成をお願いします



- ログイン後、はじめて Google Cloud Platform(GCP) の画面にアクセスするとこのようなダイアログが表示されますので、国を選択し、利用規約にチェックを入れて「同意して続行」



- 同意すると、API とサービスのダッシュボードが表示されると思いますので、プロジェクトを作成のリンクを押下して今回の講義用のプロジェクト（箱）を作成します

The screenshot shows the Google Cloud Platform API & Services dashboard. On the left, there is a sidebar with the following items:

- ダッシュボード (selected)
- ライブラリ
- 認証情報
- OAuth 同意画面
- ドメインの確認
- ページの使用に関する契約

The main content area is titled "ダッシュボード" and contains the following message:

このページを表示するには、プロジェクトを選択してください。

[プロジェクトを作成](#)

- 新しいプロジェクトの作成画面が表示されるのでプロジェクト名に適当な名前を指定して作成

新しいプロジェクト

⚠ 割り当て内の残りのプロジェクト数は 12 projects 件です。プロジェクトの増加をリクエストするか、プロジェクトを削除してください。[詳細](#)

[MANAGE QUOTAS](#)

プロジェクト名 * ?

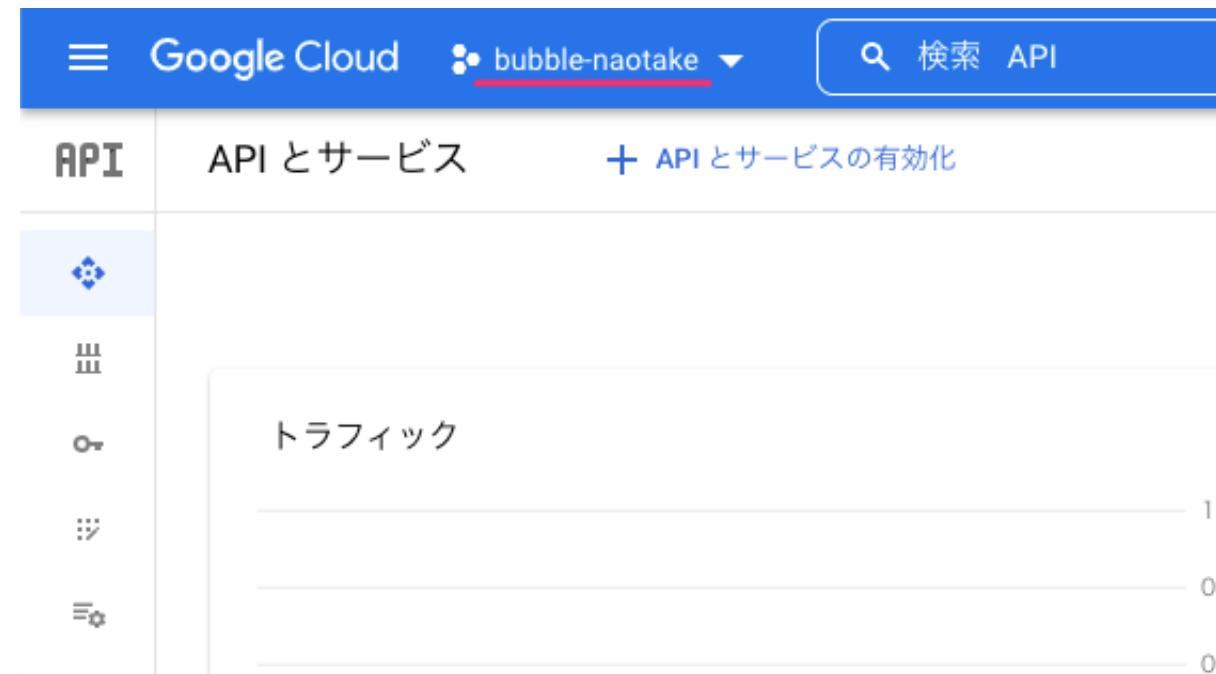
プロジェクト ID: **bubble-advance**。後で変更することはできません。 [編集](#)

場所 * 参照

親組織またはフォルダ

[作成](#) [キャンセル](#)

- 作成が完了すると、作成したプロジェクトのダッシュボードが表示されます
- 画面上部の Google Cloud の左隣に作成したプロジェクト名が表示されていれば OK



- 続いて YouTube の動画一覧を表示するために必要な認証キーを発行していきます
- API とサービスというメニューを開くため、画面上部の検索ボックスに "API" と入力
- サジェスト候補の中から "プロジェクトとページ" ブロックにある "API とサービス" を選択



- API とサービスページが開いたら、左メニューから「認証情報」を選択
- 右パネルが認証情報に変わるので、画面上部の「+ 認証情報を作成」をクリック
- ポップアップから「API キー」を選択

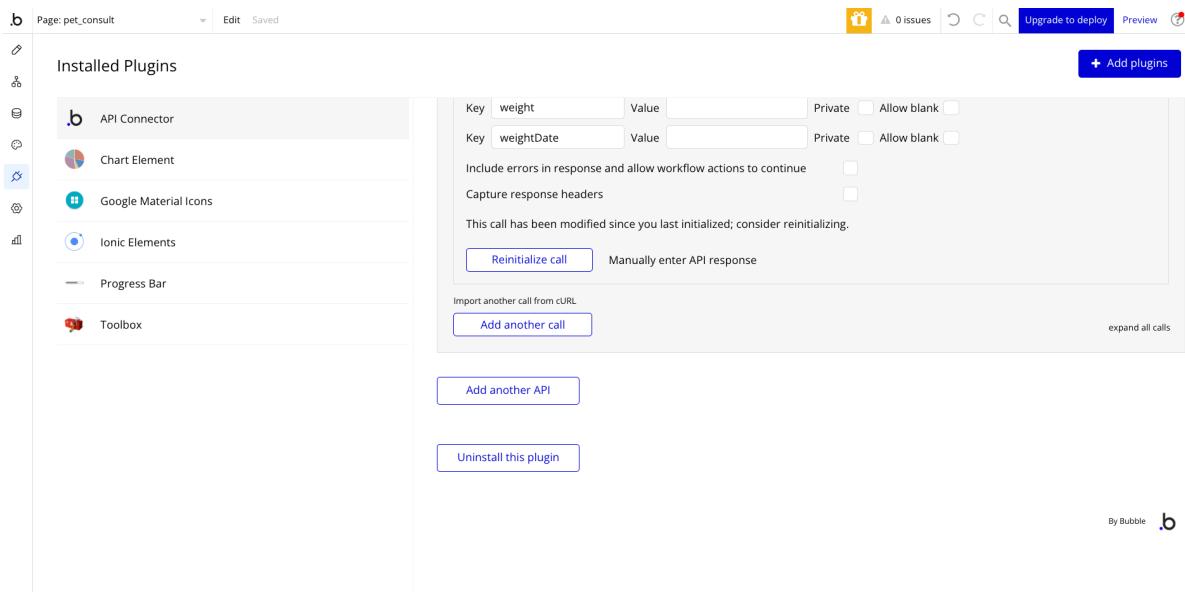
The screenshot shows the Google Cloud Platform interface for managing APIs and services. On the left, there's a sidebar with icons for '有効な API とサービス' (Enabled APIs & Services), 'ライブラリ' (Library), and '認証情報' (Authentication). The '認証情報' icon is highlighted with a red underline. The main content area has a header '認証情報' with a '認証情報を作成' button and a '削除' button. A modal window is open, titled 'API キー'. It contains three options: 'API キー' (selected, indicated by a red underline), 'OAuth クライアント ID', and 'サービス アカウント'. Below each option is a brief description. At the bottom of the modal, there's a '名前' (Name) input field and a '表示する API キー' (API keys to display) dropdown. The background shows a partially visible section for 'OAuth 2.0 クライアント ID'.

- すると API キー作成完了のポップアップが表示されるので、その値を控えておきます
- キーを制限しろと言われますが、講義の最後にこれらのキーを削除しますので一旦このまま行きます



- これで Bubble から YouTube の一覧を取得するためのキーが発行できたので Bubble 側の設定を行っていきます
- まずは Bubble から YouTube の一覧を API を介して取得するための準備を行います

- それでは YouTube API を使うための設定を行っていきます
- 先ほど ChatGPT API との連携で使用した、 "API Connector" を使用します
- **Plugins** のメニューから **API Connector** を選択し、画面下部にある "Add another API" をクリックし、新しい API 連携の設定を開始します



1. API Name: この API 連携のグループ名。今回は YouTube

The screenshot shows the configuration interface for a new API integration. The top section is titled 'API Name' with the value 'YouTube'. To its right is an 'Authentication' dropdown set to 'None or self-handled'. A red circle labeled ① points to the 'API Name' field. A red circle labeled ② points to the 'Authentication' dropdown. Below this, a red circle labeled ③ points to a section for 'Shared headers for all calls' with a 'Add a shared header' button. Another red circle labeled ④ points to a section for 'Shared parameters for all calls' with a 'Add a shared parameter' button. The bottom section is titled 'Name' and contains an 'API Call' input field. A red circle labeled ⑤ points to the 'Name' field. At the bottom left, there's a link to 'Import another call from cURL' and a blue 'Add another call' button. On the right side of the bottom section, there are 'expand' and 'expand all calls' buttons.

collapse

① API Name YouTube ② Authentication None or self-handled ▾

③ Shared headers for all calls

Add a shared header

④ Shared parameters for all calls

Add a shared parameter

expand

Name API Call

Import another call from cURL

Add another call

expand all calls

2. Authentication: このグループ配下の API で行う共通の認証方式

- 今回は "None or self-handled"

The screenshot shows the configuration interface for an API named "YouTube". The "Authentication" dropdown is set to "None or self-handled". There are sections for shared headers and parameters, each with an "Add a shared header" or "Add a shared parameter" button. A collapsed section for API calls is shown at the bottom.

① API Name: YouTube

② Authentication: None or self-handled

③ Shared headers for all calls

Add a shared header

④ Shared parameters for all calls

Add a shared parameter

Name: API Call

Import another call from cURL

Add another call

expand

collapse

expand all calls

3. Shared headers for all calls: このグループ配下の API すべてで指定する共通のヘッダー情報

- 今回は特に指定しません

The screenshot shows the configuration interface for an API group named "YouTube".

- ① API Name:** YouTube
- ② Authentication:** None or self-handled
- ③ Shared headers for all calls:** An "Add a shared header" button.
- ④ Shared parameters for all calls:** An "Add a shared parameter" button.

Below these sections is a collapsed section labeled "Name: API Call".

At the bottom left, there is a link to "Import another call from cURL" and a blue "Add another call" button. At the bottom right, there is an "expand all calls" button.

4. Shared parameters for all calls: このグループ配下の API すべてで指定する共通のパラメータ情報

- 今回は特に指定しません

The screenshot shows the configuration interface for shared parameters across all API calls. At the top, there are fields for 'API Name' (set to 'YouTube') and 'Authentication' (set to 'None or self-handled'). Below these are sections for 'Shared headers for all calls' and 'Shared parameters for all calls'. Each section has a 'Add a shared header' or 'Add a shared parameter' button. A large expandable box at the bottom contains an 'API Call' entry with a 'Name' field set to 'API Call'. There are also buttons for importing from cURL and expanding all calls.

collapse

① API Name YouTube ② Authentication None or self-handled ▾

③ Shared headers for all calls

Add a shared header

④ Shared parameters for all calls

Add a shared parameter

expand

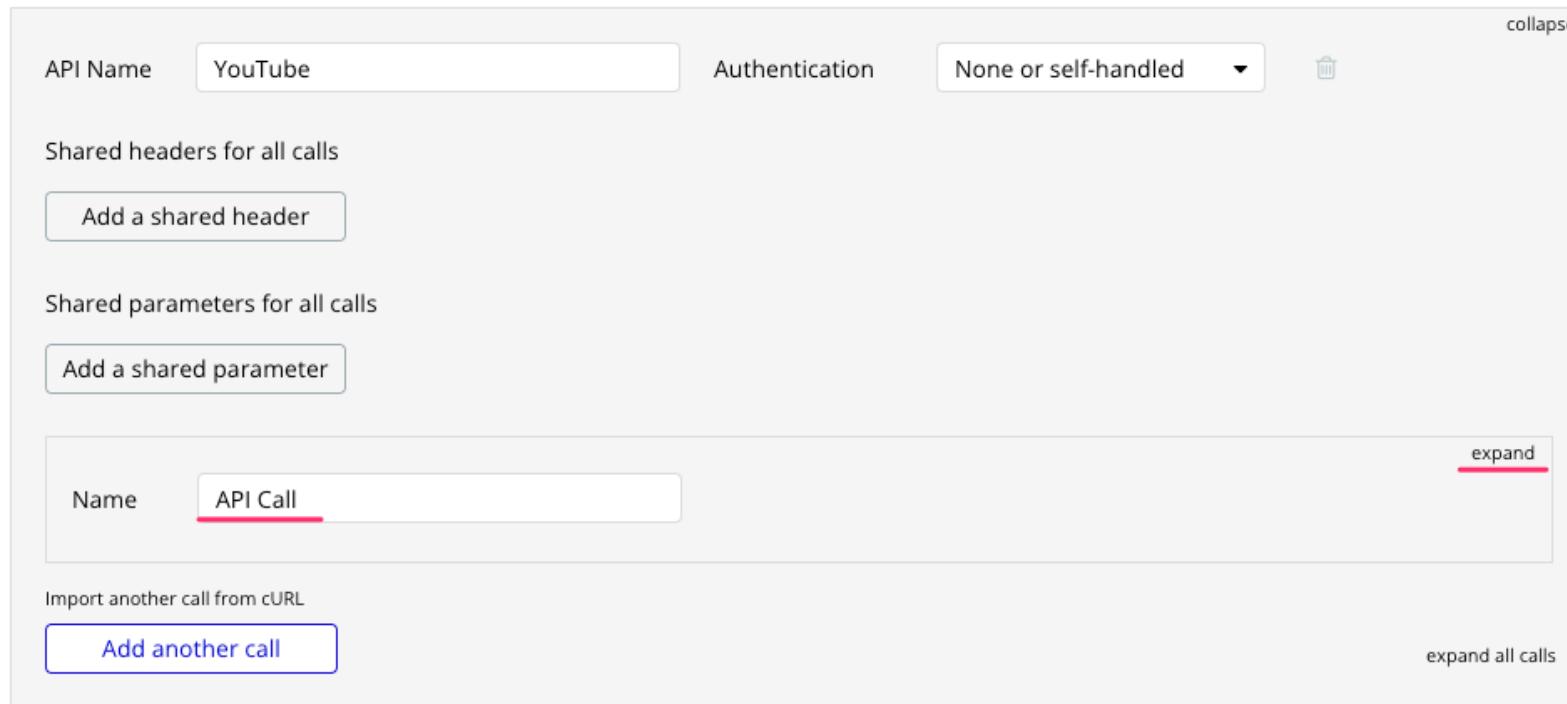
Name API Call

Import another call from cURL

Add another call

expand all calls

- 続いて、このグループに対して具体的な API の内容を設定していきます
- 先ほどの画面で "API Call" となっているブロックの右側に expand というリンクがあるのでそれをクリック



- すると、具体的な API の設定項目が表示されますので、簡単に説明します

collapse

Name API Call Use as Data Data type JSON (use [] for params)

GET Attempt to make the call from the browser

Headers Add header

Parameters Add parameter

Include errors in response and allow workflow actions to continue

Capture response headers

⚠ You need to initialize this call before it will work.

Initialize call Manually enter API response

1. Name: 具体的な API の名前

- 今回は "search" と入力します

The screenshot shows the configuration for a specific API call named 'search'. The configuration includes:

- Name:** search (marked with ①)
- Method:** GET (marked with ②)
- URL:** https://www.googleapis.com/youtube/v3/search
- Data type:** JSON
- Headers:** A section labeled ③ Headers with an 'Add header' button.
- Parameters:** A section labeled ④ Parameters with two entries:
 - Key: key, Value: AlzaSyAA3b34Zbc9r zx nw, Private: checked, Allow blank: unchecked, Optional: unchecked
 - Key: q, Value: おしゃべり唐揚げ, Private: checked, Allow blank: unchecked, Optional: unchecked
- Advanced Options:**
 - Include errors in response and allow workflow actions to continue:
 - Capture response headers:
- Initialization:** A warning message: **⚠ You need to initialize this call before it will work.** Below are two buttons:
 - Initialize call** (highlighted with a blue border)
 - Manually enter API response

2. Path: 具体的な API の URL

- 今回は `https://www.googleapis.com/youtube/v3/search` を入力します

The screenshot shows a workflow editor interface with the following configuration:

- Name:** search (marked with ①)
- Method:** GET (marked with ②)
- URL:** `https://www.googleapis.com/youtube/v3/search`
- Headers:** (marked with ③) - An "Add header" button is present.
- Parameters:** (marked with ④)
 - Key: key, Value: AlzaSyAA3b34Zbc9rznw, Private: checked, Allow blank: unchecked, Optional: unchecked
 - Key: q, Value: おしゃべり唐揚げ, Private: checked, Allow blank: unchecked, Optional: unchecked
- Include errors in response and allow workflow actions to continue:** An unchecked checkbox.
- Capture response headers:** An unchecked checkbox.
- Warning:** A red warning message: **⚠ You need to initialize this call before it will work.**
- Buttons:**
 - Initialize call (highlighted in blue)
 - Manually enter API response

3. Headers: この API 固有のヘッダー情報

- 今回は特に指定しません

The screenshot shows a configuration panel for an API call. The top section includes fields for Name (search), Use as (Data), Data type (JSON), and a URL (https://www.googleapis.com/youtube/v3/search). A 'collapse' button is in the top right. Below this, the 'Headers' section (marked ③) contains an 'Add header' button. The 'Parameters' section (marked ④) lists two parameters: 'key' with value 'AlzaSyAA3b34Zbc9r zx nw.' and 'q' with value 'おしゃべり唐揚げ'. Both parameters have 'Private' checked, 'Allow blank' unchecked, and 'Optional' unchecked. Buttons for 'Add header' and 'Add parameter' are also present. At the bottom, there are checkboxes for 'Include errors in response and allow workflow actions to continue' and 'Capture response headers', both of which are unchecked. A red warning message '⚠ You need to initialize this call before it will work.' is displayed. Finally, there are buttons for 'Initialize call' (highlighted in blue) and 'Manually enter API response'.

Name: search
Use as: Data
Data type: JSON
URL: https://www.googleapis.com/youtube/v3/search
(use [] for params)

③ Headers
Add header

④ Parameters

Key	Value	Private	Allow blank	Optional
key	AlzaSyAA3b34Zbc9r zx nw.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
q	おしゃべり唐揚げ	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Add parameter

Include errors in response and allow workflow actions to continue

Capture response headers

⚠ You need to initialize this call before it will work.

Initialize call Manually enter API response

4. Parameters: この API 固有のパラメータ情報

- 今回は下記 2 つのパラメータを指定します。Private / Allow blank / Optional は添付の通り
 - i. Key: `key` 、 Value: 先ほど Google アカウント側で発行した API キー
 - ii. Key: `q` 、 Value: YouTube で動作検索をする際のキーワード（ご自由に指定ください）

- すべて設定するとこんな感じです

collapse

Name	① search	Use as	Data	Data type	JSON	<input type="button" value="collapse"/>				
GET	② https://www.googleapis.com/youtube/v3/search	(use [] for params)				<input type="button" value="delete"/>				
③ Headers										
<input type="button" value="Add header"/>										
④ Parameters										
Key	key	Value	AlzaSyAA3b34Zbc9rznw.	Private	<input checked="" type="checkbox"/>	Allow blank	<input type="checkbox"/>	Optional	<input type="checkbox"/>	<input type="button" value="delete"/>
Key	q	Value	おしゃべり唐揚げ	Private	<input checked="" type="checkbox"/>	Allow blank	<input type="checkbox"/>	Optional	<input type="checkbox"/>	<input type="button" value="delete"/>
<input type="button" value="Add parameter"/>										
Include errors in response and allow workflow actions to continue <input type="checkbox"/>										
Capture response headers <input type="checkbox"/>										
A You need to initialize this call before it will work.										
<input type="button" value="Initialize call"/>				Manually enter API response						

- 設定が終わったら、設定内容が正しいかを確認します
- 下部にある "Initialize call" ボタンをクリック

- おそらく Status code 403 のエラーメッセージが返ってきたはずです
- 内容は書いてある通りですが、要約するとこんな感じです

YouTube Data API v3 はあなたのプロジェクトで無効になっています。
この URL にアクセスして有効にしてから再試行してください。

API Connector

There was an issue setting up your call.

Raw response for the API

Status code 403

```
{  
  "error": {  
    "code": 403,  
    "message": "YouTube Data API v3 has not been used in project 298727815636 before or it is disabled. Enable it by visiting https://console.developers.google.com/apis/api/youtube.googleapis.com/overview?project=298727815636 then retry. If you enabled this API recently, wait a few minutes for the action to propagate to our systems and retry.",  
    "errors": [  
      {  
        "message": "YouTube Data API v3 has not been used in project 298727815636 before or it is disabled. Enable it by visiting https://console.developers.google.com/apis/api/youtube.googleapis.com/overview?project=298727815636 then retry. If you enabled this API recently, wait a few minutes for the action to propagate to our systems and retry.",  
        "domain": "usageLimits",  
        "reason": "accessNotConfigured",  
        "extendedHelp": "https://console.developers.google.com"  
      }  
    ],  
    "status": "PERMISSION_DENIED",  
    "details": [  
      {  
        "@type": "type.googleapis.com/google.rpc.Help",  
        "links": [  
          {  
            "text": "View help center",  
            "url": "https://support.google.com/cloud/answer/6364746"  
          },  
          {  
            "text": "View developer guide",  
            "url": "https://cloud.google.com/youtube/v3/quickstart"  
          }  
        ]  
      }  
    ]  
  }  
}
```

- どうやら今回利用した YouTube Data API が各人の Google アカウント上で有効になっていないようです
- これは Google さんが全員が使わないであろう機能は最初から OFF にしていて、必要ある人だけ自分で ON にしてね、という状態になっています
- なので今回は YouTube Data API を使いたいので、メッセージに記載されている URL にアクセスして有効化しましょう

[https://console.developers.google.com/apis/api/youtube.googleapis.com/overview?
project={各人のプロジェクトID}](https://console.developers.google.com/apis/api/youtube.googleapis.com/overview?project={各人のプロジェクトID})

- すると親切に今回利用したい "YouTube Data API v3" の画面へ遷移したはずです
- 画面にあるとおり「有効にする」をクリックして有効にします



- しばらくするとこんな画面に遷移するはずです
- 画面上部の赤線部分が「API を無効にする」となっていれば有効化は成功です

The screenshot shows the Google Cloud Platform API library interface. On the left, there's a sidebar with navigation links: '有効な API とサービス' (selected), 'ライブラリ', '認証情報', 'OAuth 同意画面', and 'ページの使用に関する契約'. The main content area has a breadcrumb navigation: 'API / サービスの詳細' and a red underline over the link '■ API を無効にする'. Below this, there's a large thumbnail featuring the YouTube logo. The title 'YouTube Data API v3' is displayed, followed by a description: 'The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists, and channels.' It shows the service owner as 'Google'. A table provides detailed information: 'サービス名' (youtube.googleapis.com), 'タイプ' (公開 API), and 'ステータス' (有効). Buttons for 'LEARN MORE' and 'API EXPLORE' are present, along with a circular arrow icon. At the bottom, there are tabs for '指標', '割り当て', and '認証情報', and dropdown menus for 'グラフを選択' (4 個のグラフ) and '30 日'.

- YouTube Data API が有効化できたので、再度 Bubble の画面に戻り "Initialize call" ボタンをクリック
- 成功すると "Returned values - search" というポップアップが表示されるはずです

Returned values - search

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

kind youtube#searchListResponse	<input type="text"/>
etag bMKu0aiX56rekNwT5RVHZLOYR-o	<input type="text"/>
nextPageToken CAUQAA	<input type="text"/>
regionCode US	<input type="text"/>
pageInfo totalResults 1000000	<input type="number"/>
pageInfo resultsPerPage 5	<input type="number"/>
items (list) (see fields below)	<input type="text"/> search item
kind youtube#searchResult	<input type="text"/>
etag WahfcvNgYipX-i3rjRIPUrHi10	<input type="text"/>
id kind youtube#channel	<input type="text"/>
id channelId UCcRigzI_jBAZh_UySjv8xuw	<input type="text"/>
id videoId L2SaFPQl4_g	<input type="text"/>

- これは今回設定した
["https://www.googleapis.com/youtube/v3/search"](https://www.googleapis.com/youtube/v3/search) の API を実行した
 結果（レスポンス）の情報を表しています

Returned values - search

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

kind	youtube#searchListResponse	<input type="text"/>
etag	mjrgGMgtIMWG6aVWolrNtjmgtG8	<input type="text"/>
nextPageToken	CAUQAA	<input type="text"/>
regionCode	US	<input type="text"/>
pageInfo totalResults	946629	<input type="number"/>
pageInfo resultsPerPage	5	<input type="number"/>
items (list)	(see fields below)	<input type="text"/>
kind	youtube#searchResult	<input type="text"/>
etag	WahfIcVNgyipX-l3rjRiPUrHi10	<input type="text"/>
id kind	youtube#channel	<input type="text"/>
id channelId	UCcRigzl_jBAZh_UySjv8xuw	<input type="text"/>
id videoId	S094aNshTSU	<input type="text"/>

詳しくは割愛しますが、ポイントとしては下記 3 つです。

- `items(list)` となっている部分が検索結果の動画一覧が格納されている一覧になります
- その型となるのが `search item` という型となり、検索結果に含まれる複数系のデータであることを表しています
- そして、この後の設定で特に重要なのが `id videoId` となっている項目です

- YouTubeを見たことがある人なら分かると思いますが、このVideoIdというのが、それぞれの動画を一意に識別する情報となっており、画面表示するときにもこのIdが必要となります
- その他、特に参照しない項目については紛らわしいのでプルダウンの中から `Ignore field` を選択しておきます
- こうすることで、Dynamic dataとして扱う時に、選択肢の中に表示しないようになり、設定項目を選ぶ際に悩まずに済みます

- すべて設定するとこんな感じです
- 設定が完了したら "SAVE" をクリックして、設定内容を保存しておきます

Returned values - search

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

kind youtube#searchListResponse	Ignore field
etag mjrgGMgtIMWG6aVWolrNtjmgG8	Ignore field
nextPageToken CAUQAA	Ignore field
regionCode US	Ignore field
pageInfo totalResults 946629	Ignore field
pageInfo resultsPerPage 5	Ignore field
items (list) (see fields below)	search item
kind youtube#searchResult	Ignore field
etag WahfIcVNgyipX-l3rjRIPUrHi10	Ignore field
id kind youtube#channel	Ignore field
id channelId UCcRigzl_jBAZh_UySjv8xuw	Ignore field
id videoId SO94aNshTSU	text

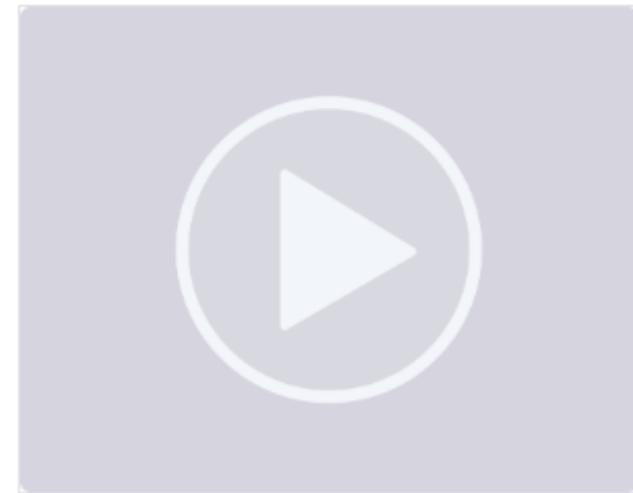
画面表示の準備

- Bubble の API 連携の準備が出来たので、いよいよ画面の設定をしていきます
- 画面の設定は前回と今回の講義でマスターしていると思いますので、概要だけお伝えして画面レイアウトを組み立ててみてください

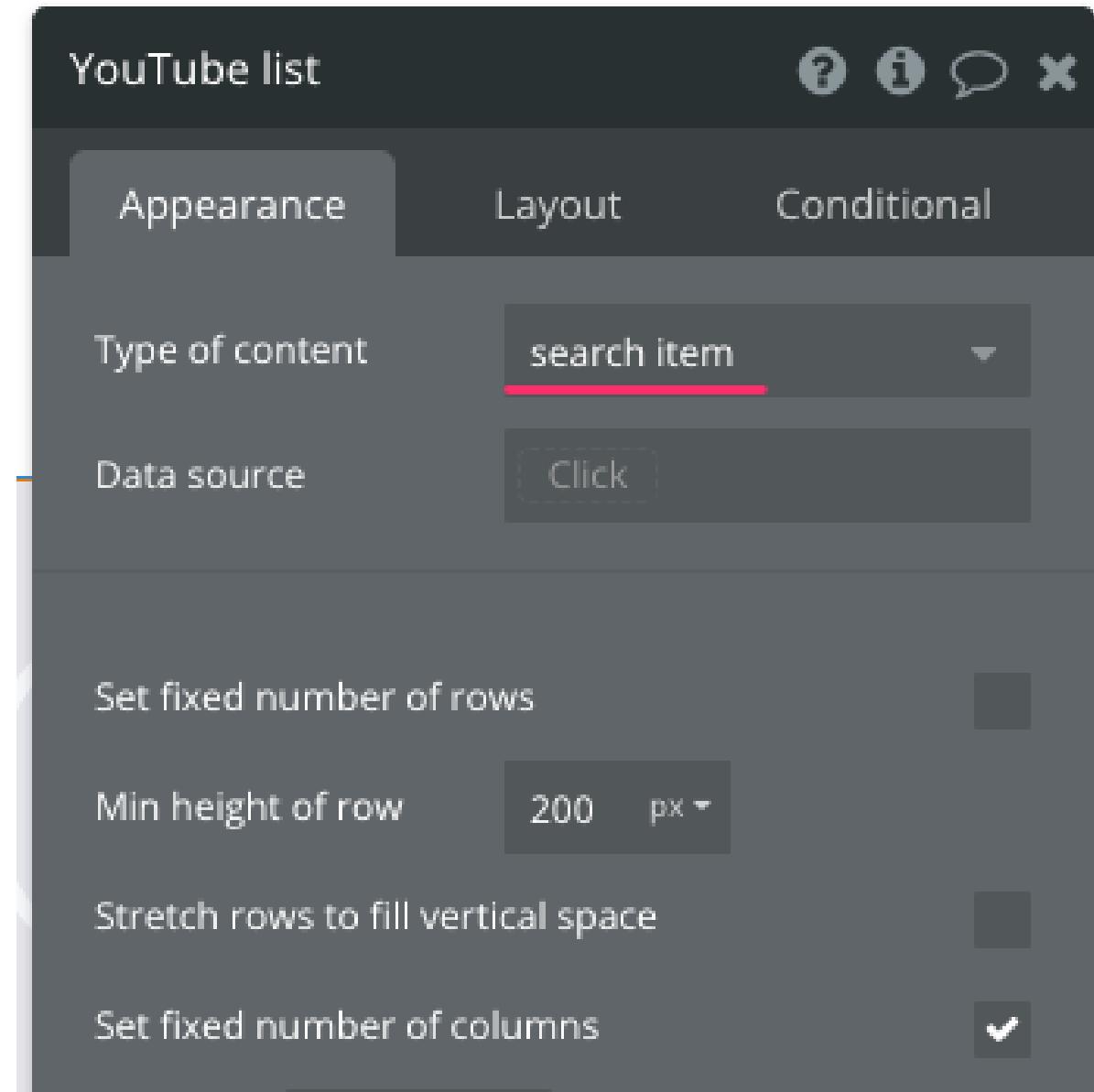


Register

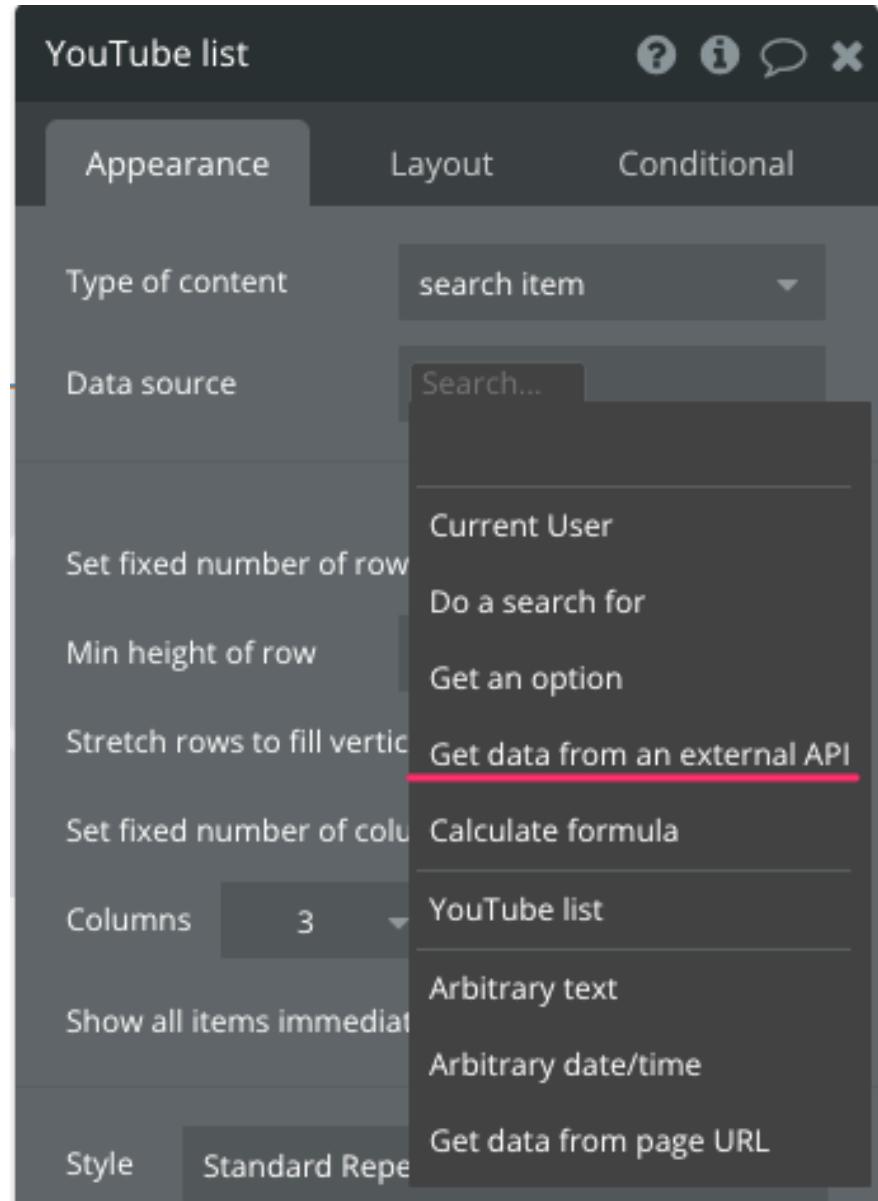
- pet_list ページを clone する形で "video_list" 画面を作成
- Repeating Group の各セルの中身 を全て削除
- 代わりに Visual elements の Video をセル内いっぱいになるよう設置
- こんな感じですね



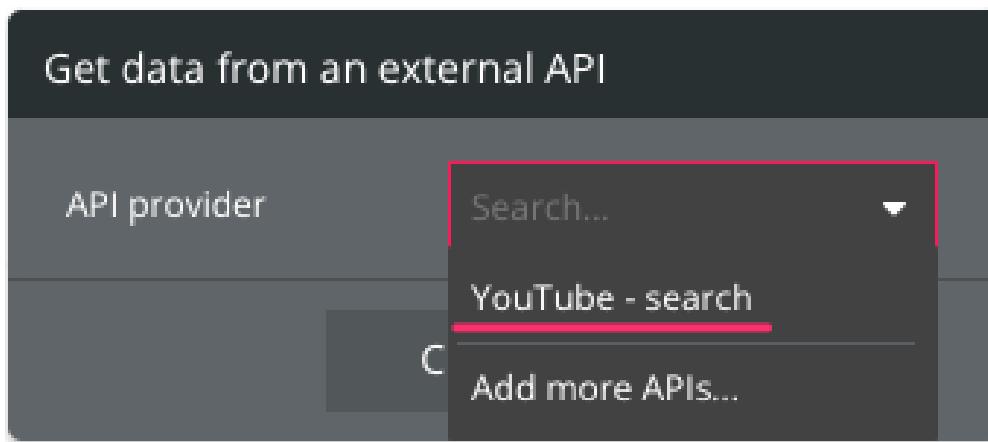
- レイアウトが出来たので実際に YouTube の一覧表示を設定していきます
- まず Repeating Group の Type of content は、先ほど API 実行時に確認した `search item` 型となります
- これで繰り返しするデータの種類が YouTube の動画 1つ1つとなります



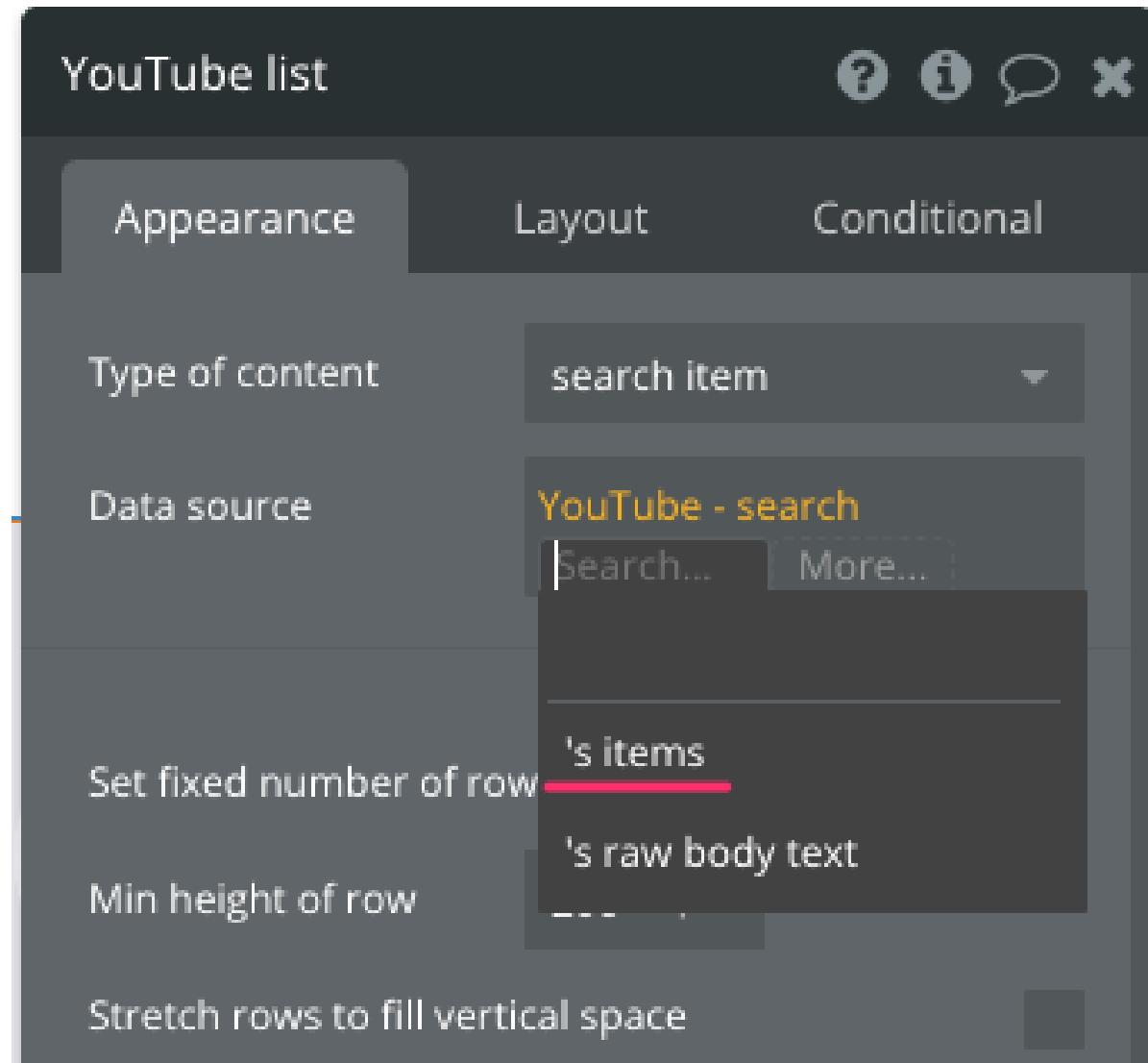
- 次に Data source ですが、今回は API から取得した YouTube の動画一覧を対象とします
- Click から候補を見てみると Get data from an external API という項目があると思うのでそちらを選択



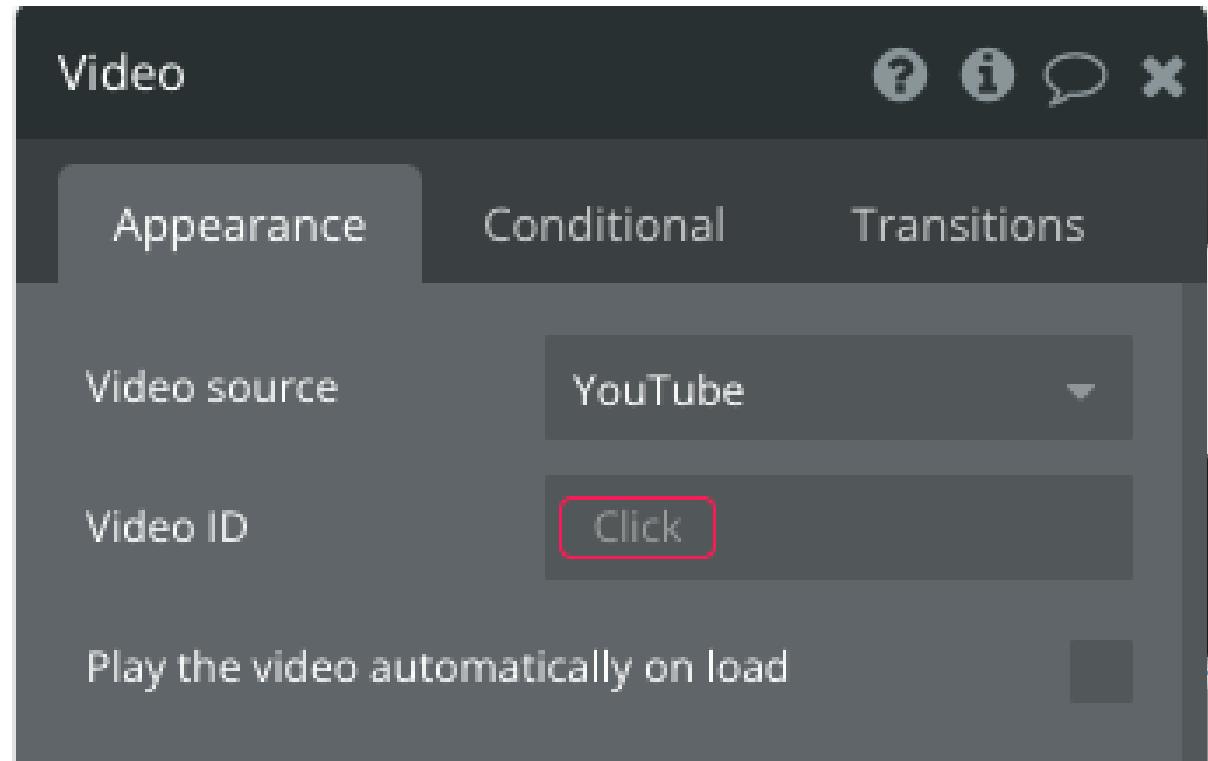
- 隣に Get data from an external API のポップアップが表示されるので、API provider の プルダウンを選んでください
- すると、プルダウンの中に先ほど設定した YouTube – search の候補があるのであると思いますのでそれを選択してください
 - ハイフンの前が API のグループ名で、後ろが具体的な API の名前です



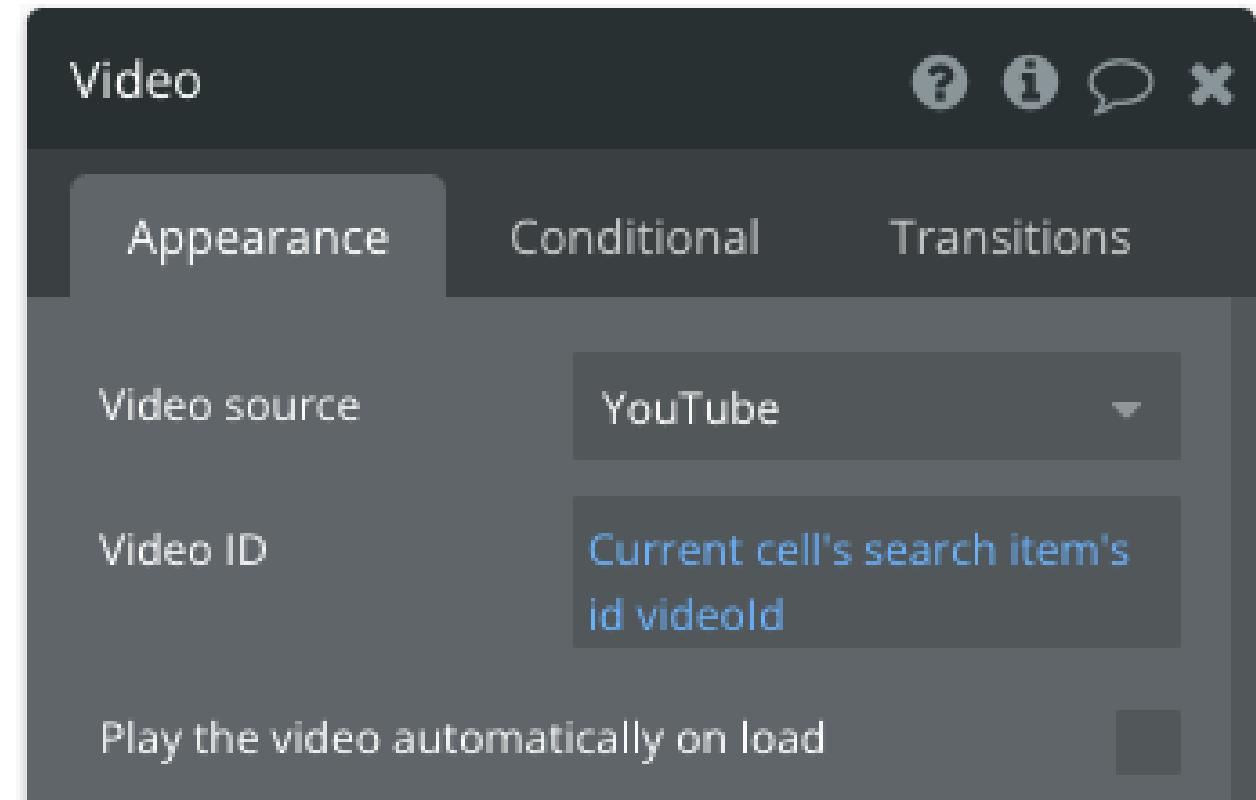
- API Provider が設定できたら、CLOSE で閉じます
- そして元の Repeating Group の Data source の選択として、さらに 's items を選択します
- これで、API の結果に含まれる動画一覧 (items) を繰り返し表示する、という指定ができました



- 次に各セルの設定をしていきますが、まずは皆さん設定してみましょう
- Hint 
 - Video source は YouTube
 - すると Video ID という項目が表示されるので、先ほど API 設定をしたときにみた Video Id を参照すればよさそうですね



- こんな感じですね



- これで設定完了です！さっそくプレビューしてみましょう！
- 事前に API で指定したキーワードの動画一覧が表示されましたか？



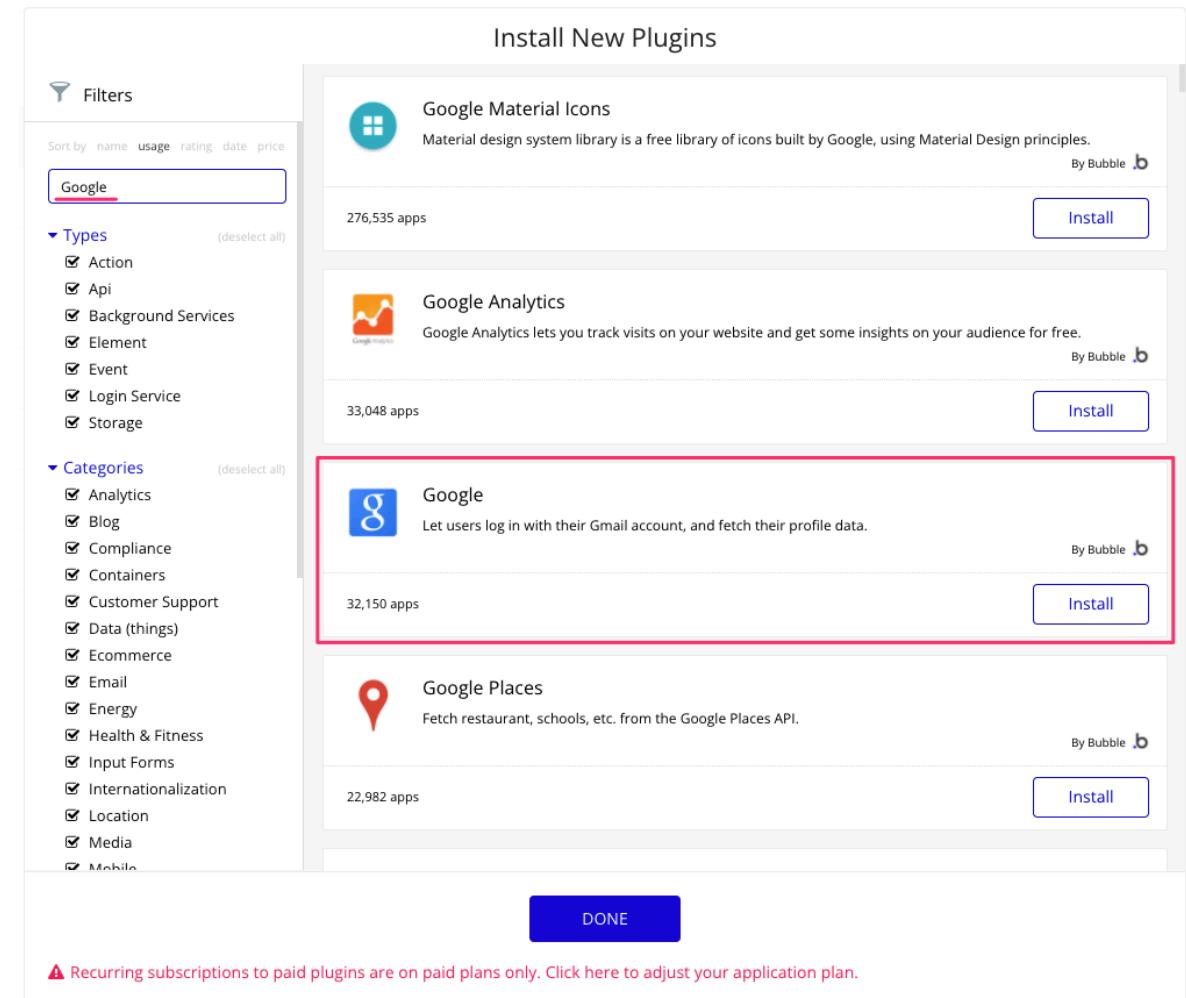
< Advanced >

Google アカウントを使ったソーシャルログイン

< Advanced >

Google ログインを試してみよう

- 続いて Bubble アプリへのログインを Google アカウントを使ってログインしてみましょう！
- まずは Google のプラグインをインストール



< Advanced >

- インストールした Google プラグインを選択
- 右パネルに表示されている `Use a generic redirect URL` の欄にチェックを入れる
 - この URL もメモしておくのですが、画面上で選択ができないため、下記 URL のアプリ名部分だけご自身のアプリ名に読み替えてメモしてください 😅
 - `https://{{Your App Name}}.bubbleapps.io/api/1.1/oauth_redirect`



Google

Service page →

Let users log in with their Gmail account, and fetch their profile data.

App Secret

App ID/API Key

Use a generic redirect URL (https://advanced-bubble-kyogoku.bubbleapps.io/api/1.1/oauth_redirect)



App Secret - dev.

(optional)

App ID/API Key - dev.

(optional)

Plugin content

This plugin is an authentication provider.

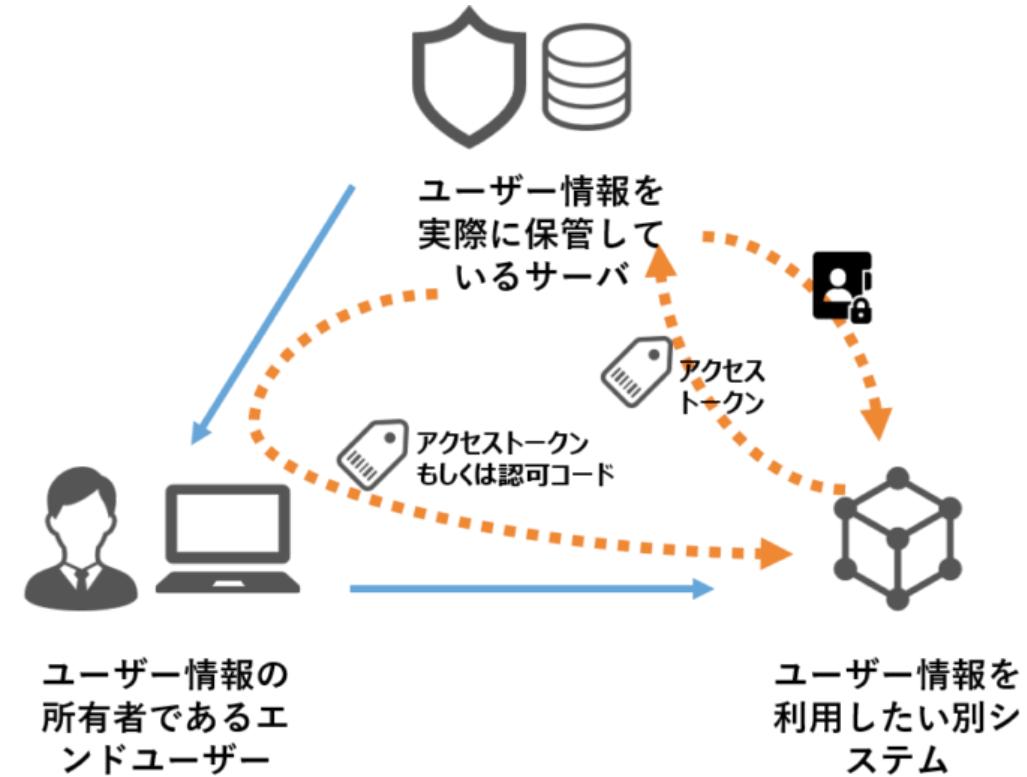
< Advanced >

OAuth とは何か

OAuth はインターネット上で、あるサービスが別のサービスにパスワードを明かすことなく、認証とリソースアクセスを可能にするための仕組みになります。あなたが他のサービス（例えばペット管理サービス）を使いたいけど、そのために新しいアカウントを作りたくないとき、OAuth を使って既存のアカウント（例えば Google や Facebook のアカウント）でログインできます。

どうやって動くのか

- ログインのリクエスト: ペット管理サービスに「Google アカウントでログイン」を選ぶ。
- 許可の確認: Google が「このサービスにあなたの情報の一部を使ってもいいか」と尋ねる。
- 「鍵」の渡し: 許可すると、Google はそのサービスに特別な「鍵」を渡す。すると、そのサービスはあなたの名前やメールアドレスなど、限られた情報にアクセスできるようになります。



< Advanced >

Googleへのログイン後にアプリに戻ってくるための認証準備

- Google Cloud Platform 画面にアクセス

<https://console.developers.google.com/>

- 先程と同じ API とサービスの画面を表示

< Advanced >

- プロジェクトを選択し、左メニューから "OAuth 同意画面" を選択

The screenshot shows the Google Cloud Platform interface. The top navigation bar includes the Google Cloud logo, a dropdown for the project ('bubble-naotake'), a search bar ('検索 プロダクト、リソース、ドキュメント...'), and various status indicators (1 notifications, help icon, etc.).

The left sidebar menu is titled 'API & サービス' and lists several options: '有効な API とサービス', 'ライブラリ', '認証情報', and 'OAuth 同意画面'. The 'OAuth 同意画面' option is highlighted with a red underline.

The main content area is titled 'OAuth 同意画面'. It contains a section for 'User Type' with two radio button options: '内部' (selected) and '外部'. Below each option is a detailed description and a link to 'ユーザーの種類の詳細'.

User Type: 内部
組織内のユーザーのみが使用できます。確認を受けるためにアプリを送信する必要はありません。[ユーザーの種類の詳細](#)

User Type: 外部
Google アカウントを持つすべてのテストユーザーが使用できます。アプリはテストモードで起動し、アプリを使用できるのは、テストユーザーのリストに追加されたユーザーに限られます。アプリを本番環境に移す準備ができたら、アプリの確認が必要となる場合があります。[ユーザーの種類の詳細](#)

A large blue '作成' (Create) button is located at the bottom left of the main content area.

At the bottom, a note states: 'Google の OAuth に関する [ご意見やご要望をお聞かせください](#)'.

< Advanced >

- User Type に "外部" を選択して作成をクリック

API API とサービス	OAuth 同意画面
<ul style="list-style-type: none">❖ 有効な API とサービス■ ライブラリ○ 認証情報☒ OAuth 同意画面≡ ページの使用に関する契約	<p>アプリをどのように構成および登録するか（ターゲット ユーザーを含む）を選択します。プロジェクトに関連付けることができるアプリは 1 つだけです。</p> <p>User Type</p> <p><input type="radio"/> 内部 <small>?</small></p> <p>組織内のユーザーのみが使用できます。確認を受けるためにアプリを送信する必要はありません。 ユーザーの種類の詳細</p> <p><input checked="" type="radio"/> 外部 <small>?</small></p> <p>Google アカウントを持つすべてのテストユーザーが使用できます。アプリはテストモードで起動し、アプリを使用できるのは、テストユーザーのリストに追加されたユーザーに限られます。アプリを本番環境に移す準備ができたら、アプリの確認が必要となる場合があります。 ユーザーの種類の詳細</p> <p>作成</p> <p>Google の OAuth に関する ご意見やご要望をお聞かせください。</p>

- 次ページで必要事項を入力していきます

< Advanced >

- アプリ情報
 - アプリ名: bubble で作成したアプリ名
 - ユーザーサポートメール: ご自身の Google アカウントのメールアドレス

アプリ登録の編集

① OAuth 同意画面 — ② スコープ — ③ テストユーザー — ④ 概要

アプリ情報

この情報は同意画面に表示されるため、デベロッパーのユーザー情報とデベロッパーへの問い合わせ方法をエンドユーザーが把握できます。

アプリ名 *
titech2022-advance-naotake

同意を求めるアプリの名前

ユーザー サポートメール *
kyougoku2bubble@gmail.com

ユーザーが同意に関して問い合わせるために使用

アプリのロゴ

参照

ユーザーがアプリを認識できるように、同意画面に 1 MB 以下の画像をアップロードします。使用できる画像形式は、JPG、PNG、BMP です。最適な結果を得るには、ロゴを 120 x 120 ピクセルの正方形にすることをおすすめします。

< Advanced >

- 承認済みドメイン
 - bubbleapps.io
- デベロッパーの連絡先情報
 - ご自身の Google アカウントのメールアドレス
- "保存して次へ"

承認済みドメイン ?

同意画面または OAuth クライアントの構成でドメインが使用されている場合は、ここで事前登録する必要があります。アプリの検証が必要な場合は、[Google Search Console](#) にアクセスして、ドメインが承認済みであるかどうかを確認してください。承認済みドメインの上限の[詳細](#)をご覧ください。

承認済みドメイン 1* —

[+ ドメインの追加](#)

デベロッパーの連絡先情報

メールアドレス * —

×

これらのメールアドレスは、プロジェクトの変更について Google からお知らせするために使用します。

[保存して次へ](#)

[キャンセル](#)

< Advanced >

- スコープは特に設定変更せず "保存して次へ"

< Advanced >

- テストユーザにはご自身の Google メールアドレスを入力しておく
- 設定したら保存して次へ

アプリ登録の編集

✓ OAuth 同意画面 — ✓ スコープ — 3 テストユーザー — 4 概要

テストユーザー

公開ステータスが「テスト中」に設定されている間は、テストユーザーのみがアプリにアクセスできます。アプリの確認前の許可済みユーザー数の上限は 100 で、この上限はアプリの全期間でカウントされます。[詳細](#)

+ ADD USERS

≡ フィルタ プロパティ名または値を入力 ?

ユーザー情報

kyougoku2bubble@gmail.com



保存して次へ

キャンセル

< Advanced >

- これで登録は完了したので続いて認証情報を発行します
- 左メニューから認証情報を選択して、「+ 認証情報を作成」をクリック
- サブメニューから OAuth クライアント ID を選択

The screenshot shows the Google Cloud Platform's API & Services page. On the left, a sidebar menu is open under 'API & Services' with the following options:

- 有効な API とサービス
- ライブラリ
- 認証情報** (highlighted with a red underline)
- OAuth 同意画面
- ページの使用に関する契約

The main content area is titled '認証情報' (Authentication Information). It contains three sections:

- API キー**: Describes using a simple API key to identify the project and grant access.
- OAuth クライアント ID**: Describes requesting user consent to access their data.
- サービス アカウント**: Describes enabling application-level authentication between servers.

At the top right of the '認証情報' section, there are two buttons: '+ 認証情報を作成' (Create Authentication Information) and '削除' (Delete). Below the sections, there is a heading 'OAuth 2.0 クラ' and a checkbox labeled '名前' (Name). At the bottom, it says '表示する OAuth クライアントがありません' (No OAuth clients displayed).

クライアント ID は、Google の OAuth サーバーで個々のアプリを識別するために使用します。アプリが複数のプラットフォームで実行される場合、それぞれに独自のクライアント ID が必要になります。詳しくは、[OAuth 2.0 の設定](#)をご覧ください。OAuth クライアントの種類の[詳細](#)

アプリケーションの種類* —

ウェブアプリケーション

名前* —

Bubble OAuth

OAuth 2.0 クライアントの名前。この名前はコンソールでクライアントを識別するためのみ使用され、エンドユーザーには表示されません。

● 下で追加する URI のドメインは、[OAuth 同意画面](#)に承認済みドメインとして自動で追加されます。

承認済みの JavaScript 生成元 [?](#)

ブラウザからのリクエストに使用します

[+ URI を追加](#)

承認済みのリダイレクト URI [?](#)

ウェブサーバーからのリクエストに使用します

URI 1* —

https://titech2022-advance-naotake.bubbleapps.io/api/1.1/oauth_redirect

[+ URI を追加](#)

注: 設定が有効になるまで 5 分から数時間かかることがあります

[作成](#)

[キャンセル](#)

< Advanced >

- OAuth クライアント ID の作成画面
が表示されるので下記の通り入力
し「保存」
 - アプリケーションの種類: ウェ
ブアプリケーション
 - 名前: **Bubble OAuth**

< Advanced >

- 承認済みのリダイレクト URI:
Bubble 側で Google プラグインインストール時に控えた URL

`https://{{Your App Name}}.bubbleapps.io/api/1.1/oauth_redirect`

`https://titech2022-advance-naotake.bubbleapps.io/api/1.1/oauth_redirect`

OAuth クライアント ID の作成

クライアント ID は、Google の OAuth サーバーで個々のアプリを識別するために使用します。アプリが複数のプラットフォームで実行される場合、それぞれに独自のクライアント ID が必要になります。詳しくは、[OAuth 2.0 の設定](#)をご覧ください。OAuth クライアントの種類の[詳細](#)

アプリケーションの種類*
ウェブアプリケーション

名前*
Bubble OAuth

OAuth 2.0 クライアントの名前。この名前はコンソールでクライアントを識別するためのみ使用され、エンドユーザーには表示されません。

下で追加する URI のドメインは、[OAuth 同意画面](#)に承認済みドメインとして自動で追加されます。

承認済みの JavaScript 生成元 [?](#)

ブラウザからのリクエストに使用します

[+ URI を追加](#)

承認済みのリダイレクト URI [?](#)

ウェブサーバーからのリクエストに使用します

URI 1*
https://titech2022-advance-naotake.bubbleapps.io/api/1.1/oauth_redirect

[+ URI を追加](#)

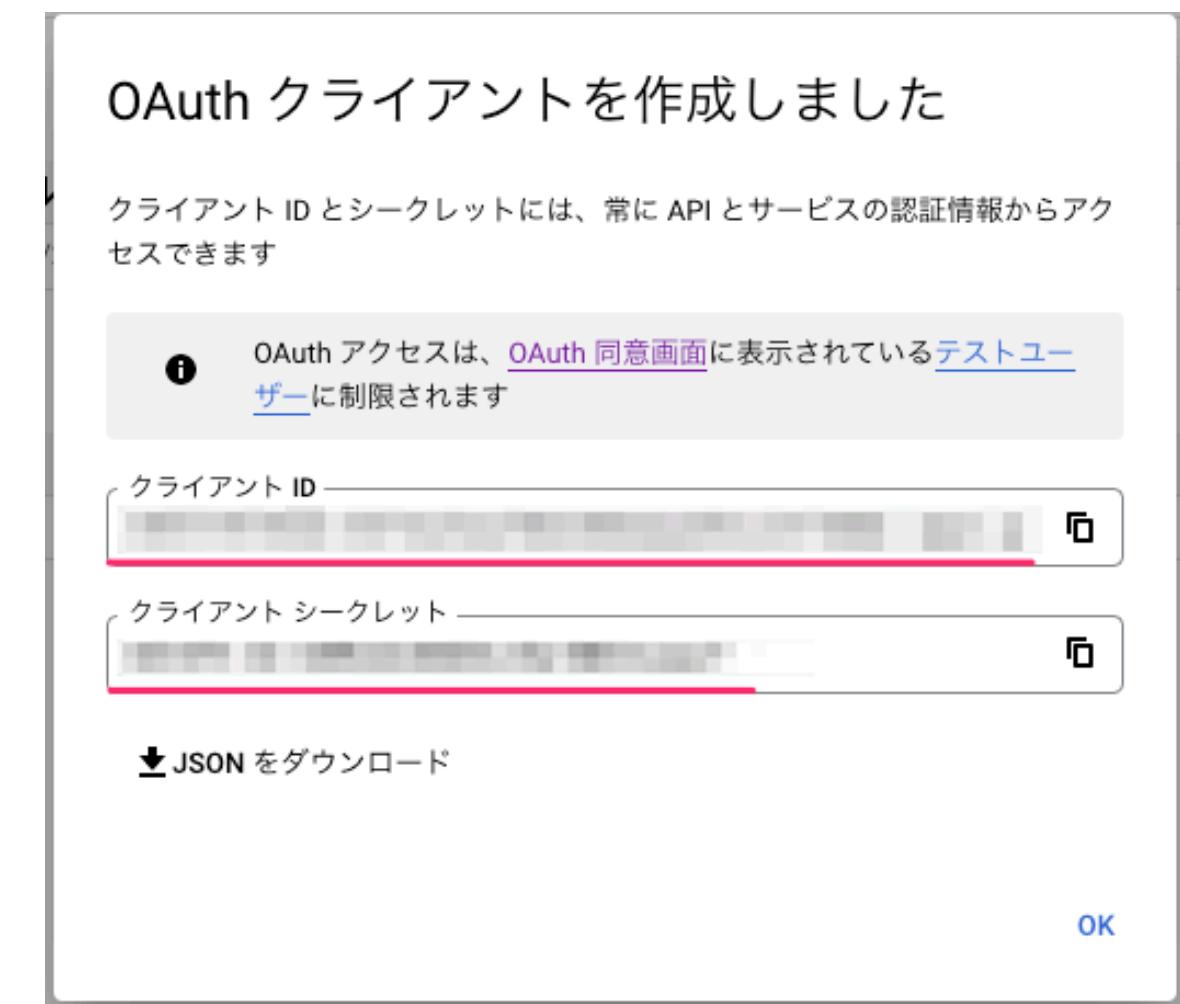
注: 設定が有効になるまで 5 分から数時間かかることがあります

作成

キャンセル

< Advanced >

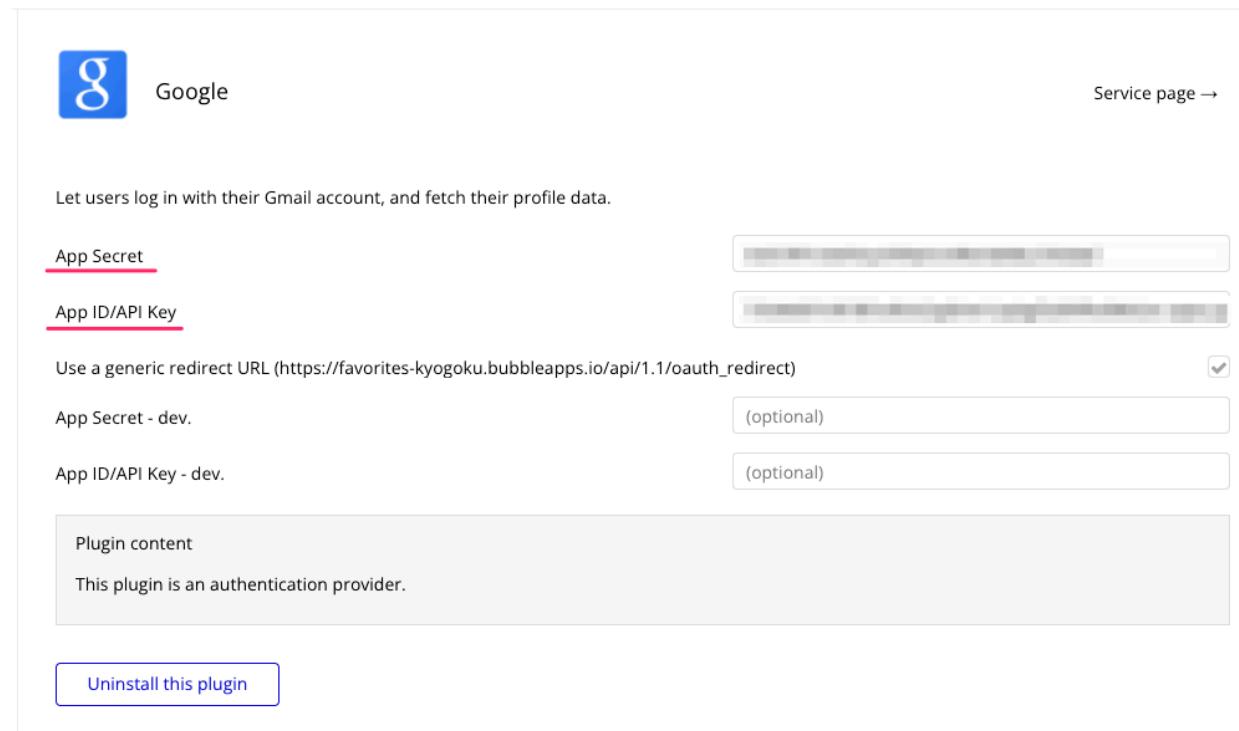
- "作成" クリック
- すると 「OAuth クライアントを作成しました」というポップアップが表示され、そこにクライアント ID とクライアントシークレットが表示されているので、それをメモしておく
 - メモだと忘れるかもしれない場合は JSON をダウンロードしても OK
- メモしたら OK でクローズ



< Advanced >

発行した認証情報を Bubble 側に設定

- Bubble の Google プラグイン画面に戻り、先ほど取得したキーをそれぞれの項目へ入力する
 - クライアント ID → AppID/API Key
 - クライアントシークレット → App Secret



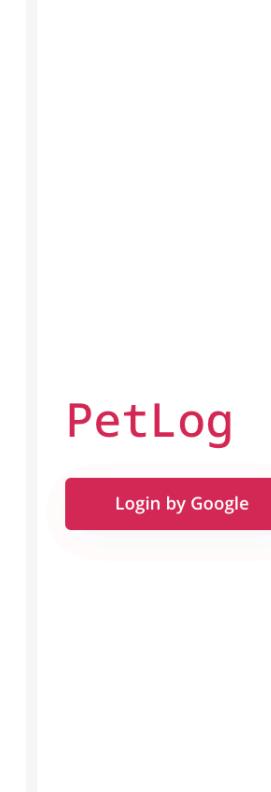
< Advanced >

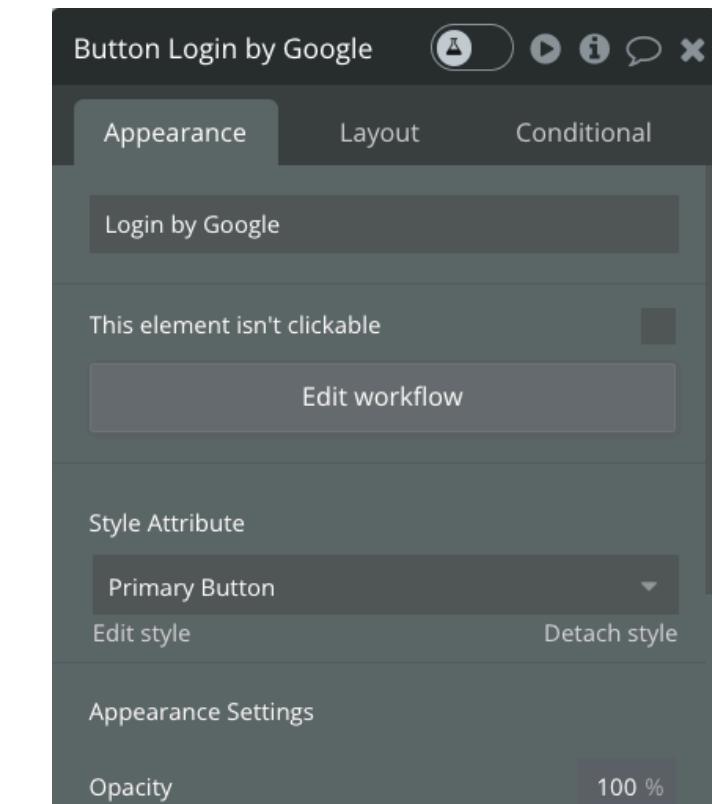
- これで事前設定は完了
- 続いてログイン機能を作り込んでいきます

< Advanced >

ログイン機構を Google に置き換える

- index 画面を開き、サインアップのテキストボックスやボタンをすべて削除しましょう
- 新たにボタンを置いて、`Login by Google` といったラベルに変更しましょう



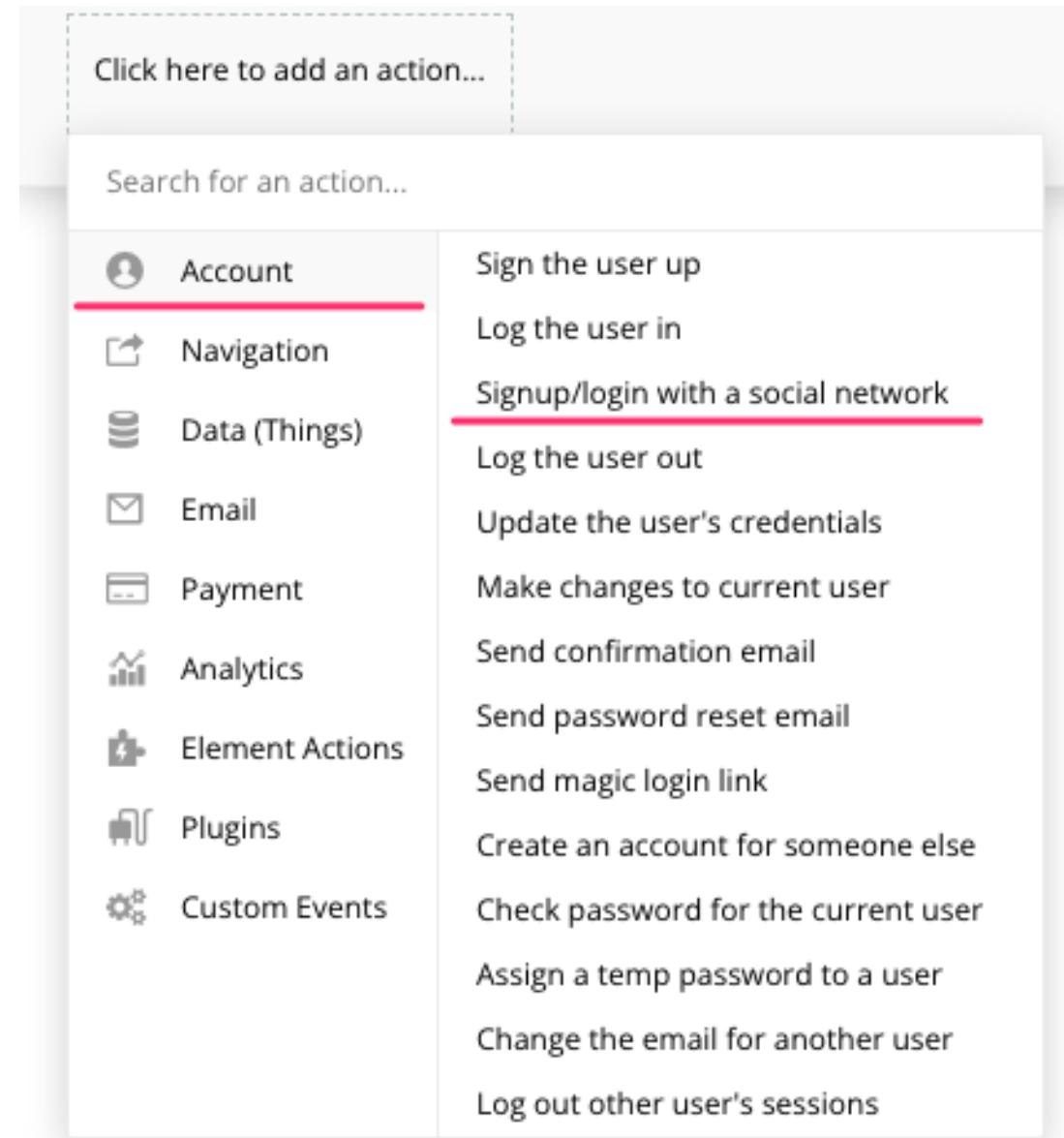


< Advanced >

- 設置したボタンの設定ウィンドウで `Edit workflow` をクリックして、ボタンクリック時のワークフロー作成を開始しましょう

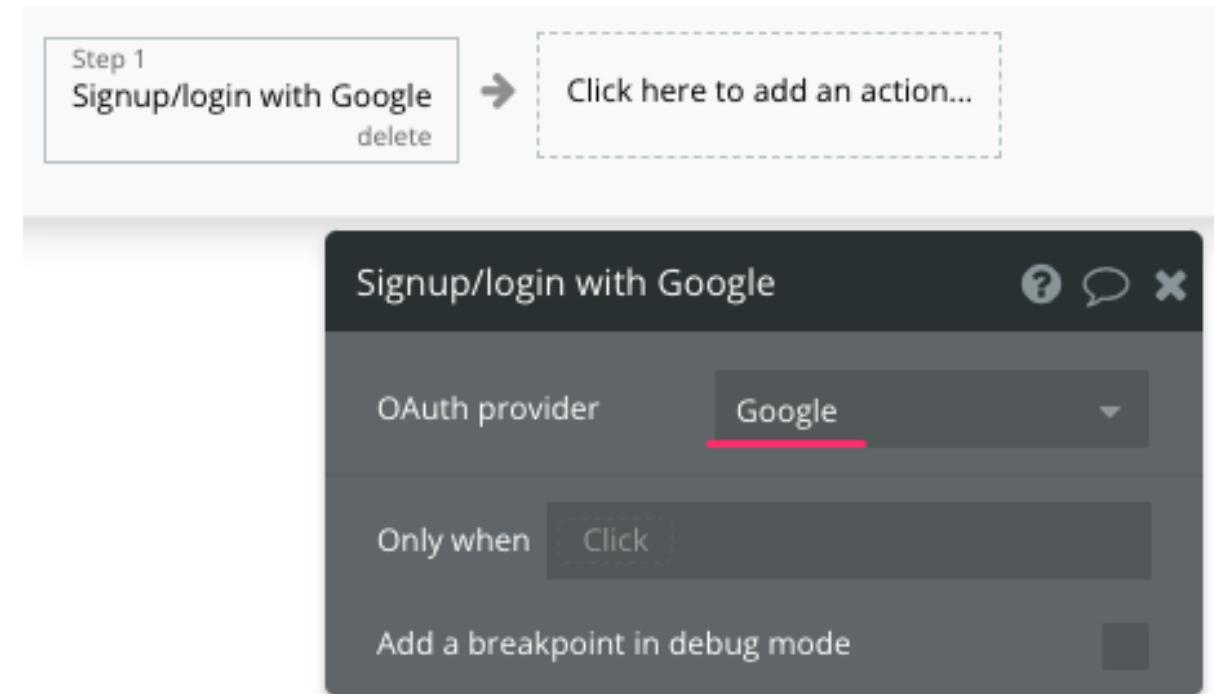
< Advanced >

- Google ログインのワークフローを設定していきます
- Click here to add an action... を選択
- Account から Signup/login with a social network を選択



< Advanced >

- Signup/login with Google のポップアップが表示されます
- OAuth provider に Google を選択



< Advanced >

ログインができた後に、ペットリストに遷移するように指定しましょう

- Click here to add an action... を選択
- Navigation > Go to page を指定する
- Destination に pet_list を指定する

これだけで Google アカウントでのログインができるようになりました。簡単ですね。

< Advanced >

- 折角なのでログイン成功したらヘッダー上部にアカウント名と画像を表示してみましょう



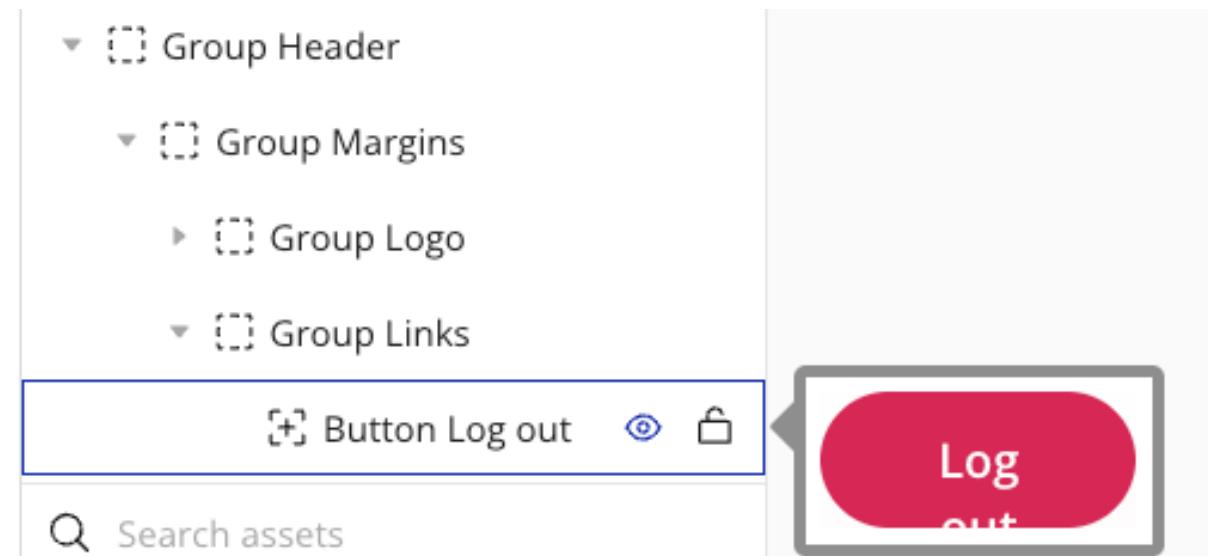
< Advanced >

- Reusable elements の header を開きます

The screenshot shows the Wagtail admin interface for managing reusable elements. At the top, there is a search bar labeled "Search a page or a reusable..." and a dropdown menu labeled "Pick an element...". To the right of these are "Edit" and "Saved" buttons. Below the search bar, there are two tabs: "Pages" and "Reusable elements". The "Pages" tab is selected, showing a list of pages: index, pet_list, pet_weight_register, reset_pw, and 404. Each page item has a small trash icon to its right. The "Reusable elements" tab is shown below, containing a single item: "header", which is underlined in red. There is also a "Add a new reusable element..." button at the bottom of this section.

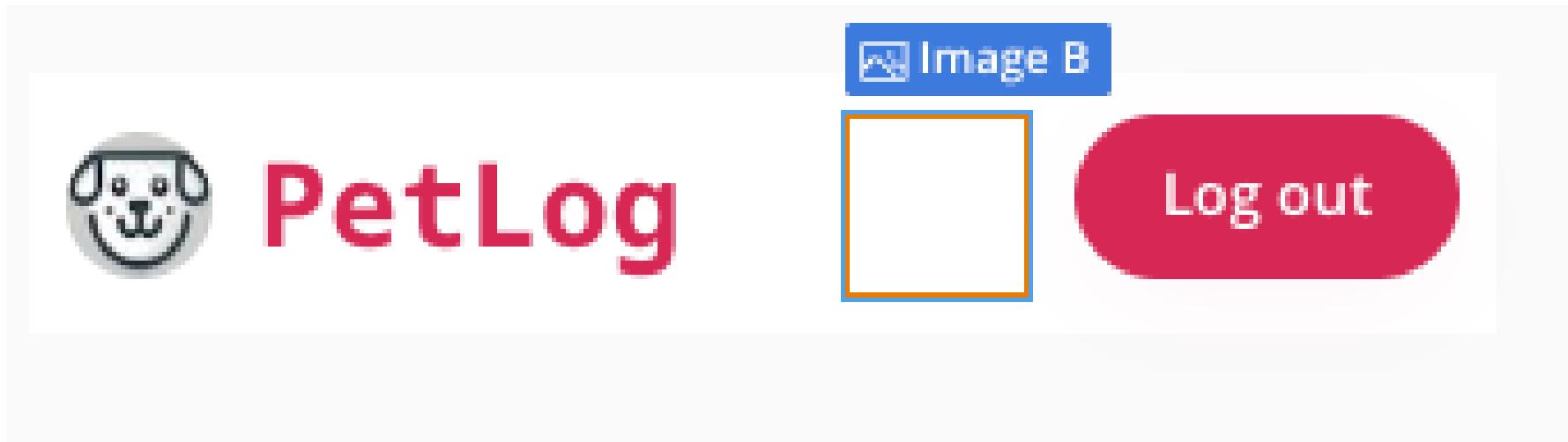
< Advanced >

- Design タブに切り替え、現在非表示になっている Log out ボタンのコンポーネントを Elements tree から表示しておきます
 - こうしておかないと、画像追加後のイメージが分かりづらいため



< Advanced >

- Visual elements から Image を選択し、"Log out" ボタンの左に画像を配置します
 - 縦横 50 x 50 の正方形にしておきましょう

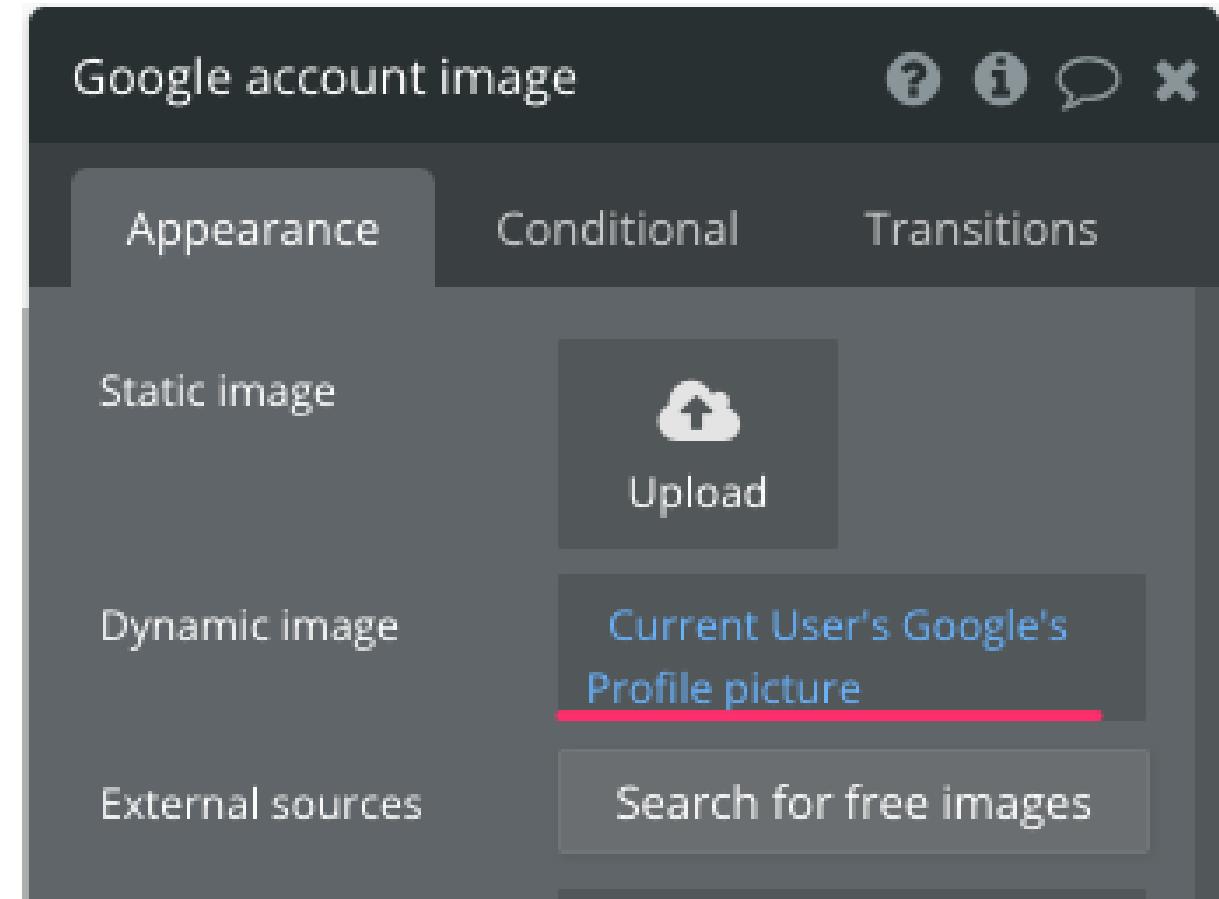


< Advanced >

- そして、画像の参照元を設定します
- Bubbler の皆さんなら設定方法はイメージつきますよね
- 動的な表示設定は Dynamic image
- 表示したいのは現在ログインしているユーザの Google アカウントのプロフィール画像

< Advanced >

- こんな感じですね



< Advanced >

- ではプレビューしてみましょう！
 - 事前に Data の User にこれから動作確認する際に使用する Google メールアドレスと同じメールアドレスのデータがある場合は削除しておいてください
- ログインボタンを押すと Google のログイン画面に遷移し、そこでログインをすると Bubble 側に戻ってきましたよね？
- ちなみに、ログインすると Bubble 側の User data にログインした Google アカウントのメールアドレス情報が登録されます
 - もちろんパスワードは保存されていないのでご安心を 😊

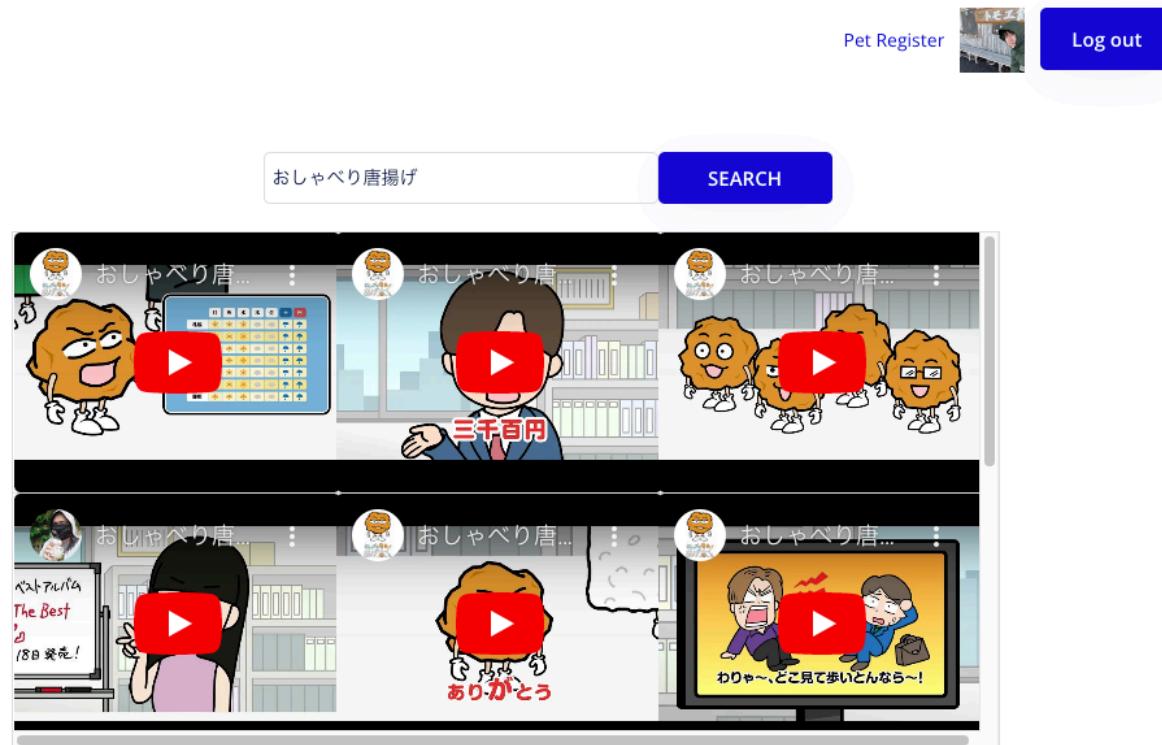


Log out

< Advanced >

YouTube の動画一覧を動的に検索

.bubble



< Advanced >

- 現状では YouTube API の認証キーと検索キーワードは固定になっています
- なので、これを下記のように変えてみましょう
 - 認証キーはログインしているユーザ情報から取得する（新たに field を追加する）
 - 検索キーワードは画面から入力・検索できるようにする

< Advanced >

次のページにヒントを書いておきますのでまずはトライしてみましょう！

< Advanced >

- まずは User に対して `key` という field を追加し、事前に設定しておく
 - 本来ここもユーザ自身に入力させるなどの導線が自然ですが今回は時間の都合上、API キーの設定画面は割愛します
- API のパラメータを処理の中で動的に設定する場合、まずは固定値を無くし `Private` のチェックを外すことで、DataSource などで API 呼び出しする際にパラメータを指定できます
- `video_list` ページに新たに検索用の要素（テキストボックスと検索ボタン）を配置し、検索ボタンがクリックされた時に、YouTube API 経由で検索を行い、結果を Repeating Group にセットしてあげます



< Advanced >

- 重要なポイントを解説します
- まずは User の型に対して `key` という field を追加します

Fields for type User

Type name	User	
key	text	default
email	text	Built-in field
Modified Date	date	Built-in field
Created Date	date	Built-in field
Slug	text	Built-in field

[Create a new field](#)

< Advanced >

- そして、今回取得した認証キー（API キー）を登録済みの User に事前にセットしておきます

The screenshot shows a user interface for managing application data. At the top, there are tabs: Data types, Privacy, App data (which is selected), Option sets, and File manager. Below the tabs, the title is "Database views" and the subtitle is "Application data - All Users - Development version". There are buttons for "New view", "Primary fields", "Search" (with a search input field), and "New entry". A search bar below the buttons contains the placeholder "Search view names or data types...". The main area displays a table with columns: a checkbox, a primary field icon, Email, and key. One row is visible, showing an email address and a redacted key value. At the bottom left, there is a link "All Users" and a "Create" button with a pencil icon.

		Email	key
<input type="checkbox"/>		kyougoku2bubble@gmail.com	AlzaSy [REDACTED]

< Advanced >

- 次に YouTube API 側の設定はこんな感じですね

The screenshot shows a configuration interface for a workflow action. At the top, there are fields for 'Name' (set to 'search'), 'Use as' (set to 'Data'), 'Data type' (set to 'JSON'), and a URL field containing 'https://www.googleapis.com/youtube/v3/search'. Below this, there are sections for 'Headers' (with an 'Add header' button) and 'Parameters'. The 'Parameters' section contains four entries:

Key	Value	Private	Allow blank	Optional
key	(redacted)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
q	(redacted)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
maxResults	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
type	video	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Below the parameters are two checkboxes: 'Include errors in response and allow workflow actions to continue' and 'Capture response headers'. A message at the bottom states 'This call has been modified since you last initialized; consider reinitializing.' At the very bottom are buttons for 'Reinitialize call' and 'Manually enter API response'.

< Advanced >

- ポイントとしては、動的に指定したい項目については Value の値を削除し、Private のチェックを外しました
- これにより、API を使う場面でこれらのパラメータの値を指定することができるようになります

The screenshot shows a configuration panel for a workflow action. At the top, there are fields for 'Name' (set to 'search'), 'Use as' (set to 'Data'), 'Data type' (set to 'JSON'), and a URL field containing 'https://www.googleapis.com/youtube/v3/search'. Below this, under 'Headers', there is a button labeled 'Add header'. Under 'Parameters', there are four entries: 'key' (Value field is redacted), 'q' (Value field is redacted), 'maxResults' (Value set to '10', Private checked, Allow blank unchecked, Optional unchecked), and 'type' (Value set to 'video', Private checked, Allow blank unchecked, Optional unchecked). There are also buttons for 'Add parameter', 'Include errors in response and allow workflow actions to continue' (unchecked), and 'Capture response headers' (unchecked). A message at the bottom states 'This call has been modified since you last initialized; consider reinitializing.' with buttons for 'Reinitialize call' and 'Manually enter API response'.

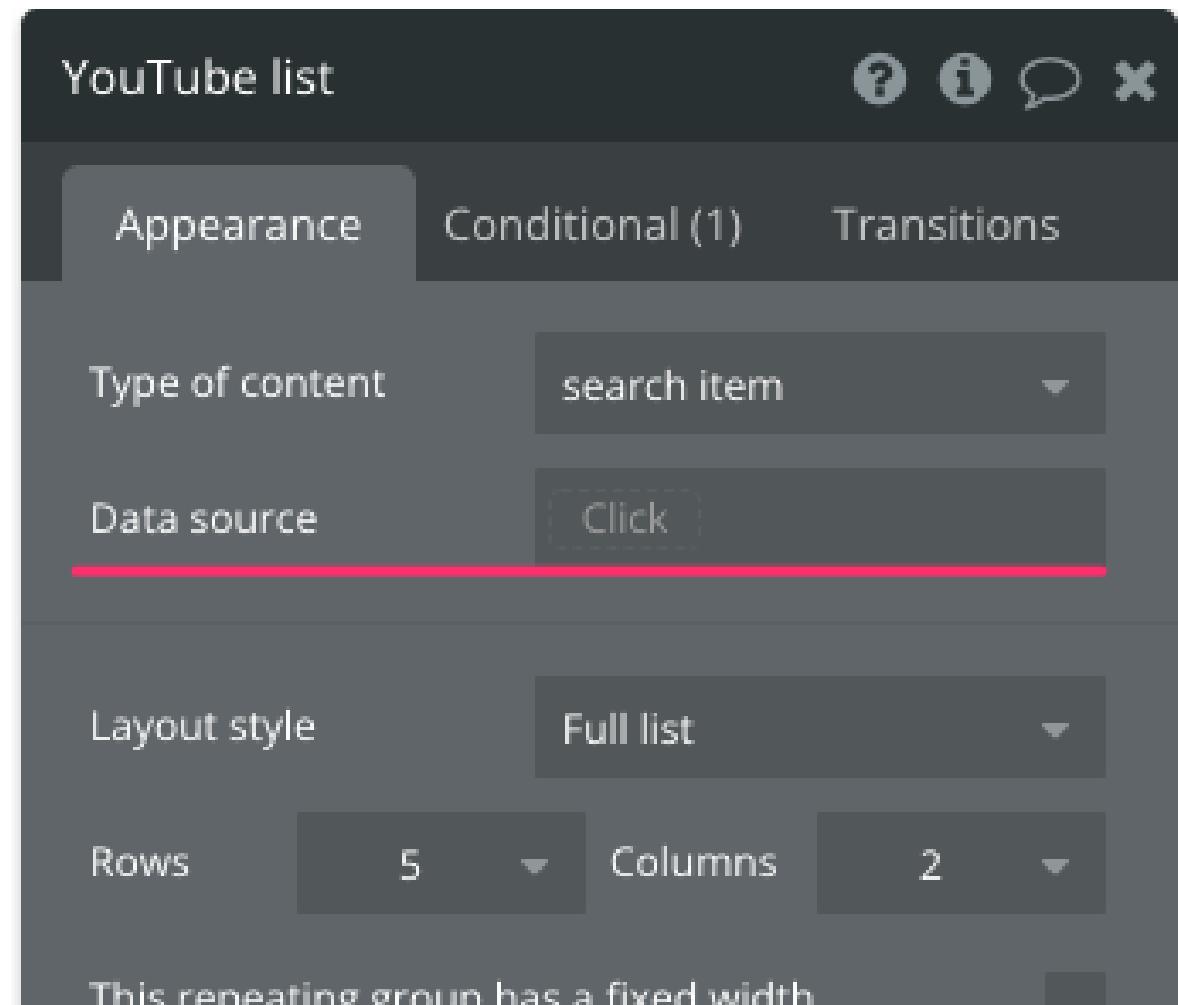
< Advanced >

- ついでに、検索のオプションを2つ追加しておきましょう
- key: maxResults、1回の検索で取得するデータの件数
 - デフォルトは5
- key: type、検索対象のデータ種類
 - video を指定することで、動画だけの検索が可能

The screenshot shows a configuration interface for an API call named 'search'. The 'Data type' is set to 'JSON'. The 'Method' is 'GET' and the 'URL' is 'https://www.googleapis.com/youtube/v3/search'. The 'Headers' section contains an 'Add header' button. The 'Parameters' section lists four entries: 'key' (Value: 'key', Private: checked, Allow blank: checked, Optional: checked), 'q' (Value: ' ', Private: checked, Allow blank: checked, Optional: checked), 'maxResults' (Value: '10', Private: checked, Allow blank: checked, Optional: checked), and 'type' (Value: 'video', Private: checked, Allow blank: checked, Optional: checked). Below the parameters are buttons for 'Add parameter', 'Include errors in response and allow workflow actions to continue' (unchecked), and 'Capture response headers' (unchecked). A message at the bottom states 'This call has been modified since you last initialized; consider reinitializing.' with buttons for 'Reinitialize call' and 'Manually enter API response'.

< Advanced >

- 続いて Repeating Group の Data source を空にします
- 表示する対象データの指定は、新たに設ける検索機能のワークフローの中で設定するため、ここが空でも問題ありません



< Advanced >

- 次にキーワード検索が行われた時のワークフローの設定となるため、新たに用意した検索ボタンに対してワークフローを設定します
- 今回は要素 (Repeating Group) に対するアクションとなるため "Element Actions" の中にある "Display list" を選択します

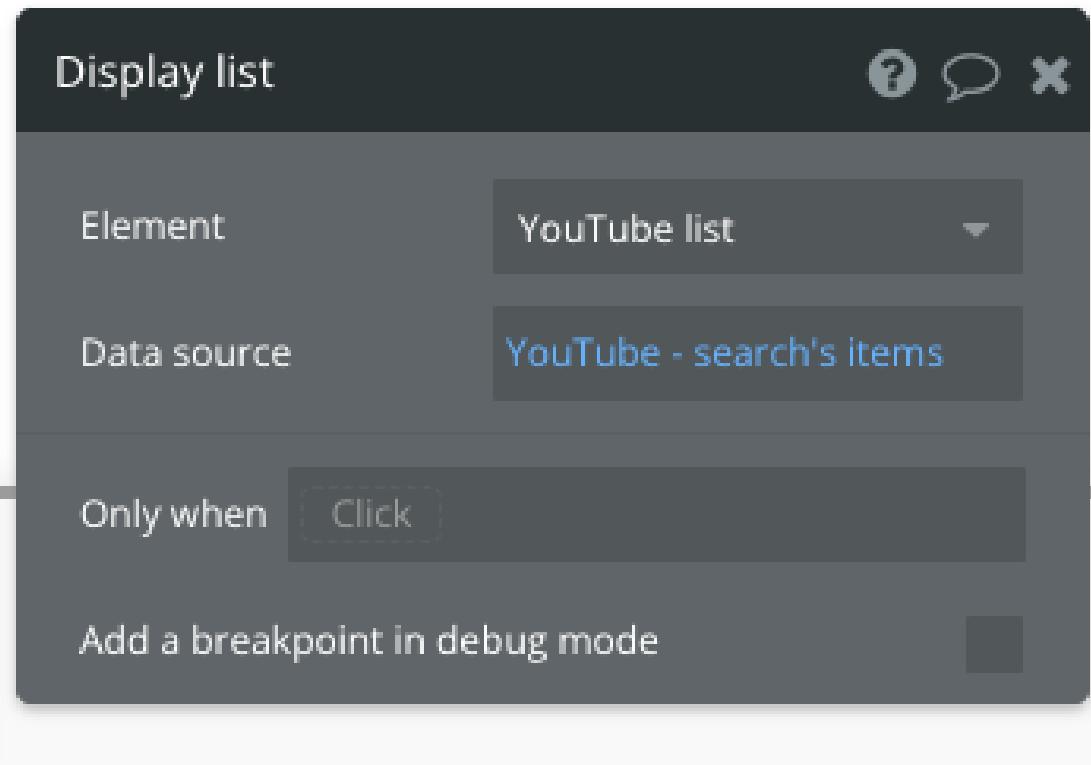
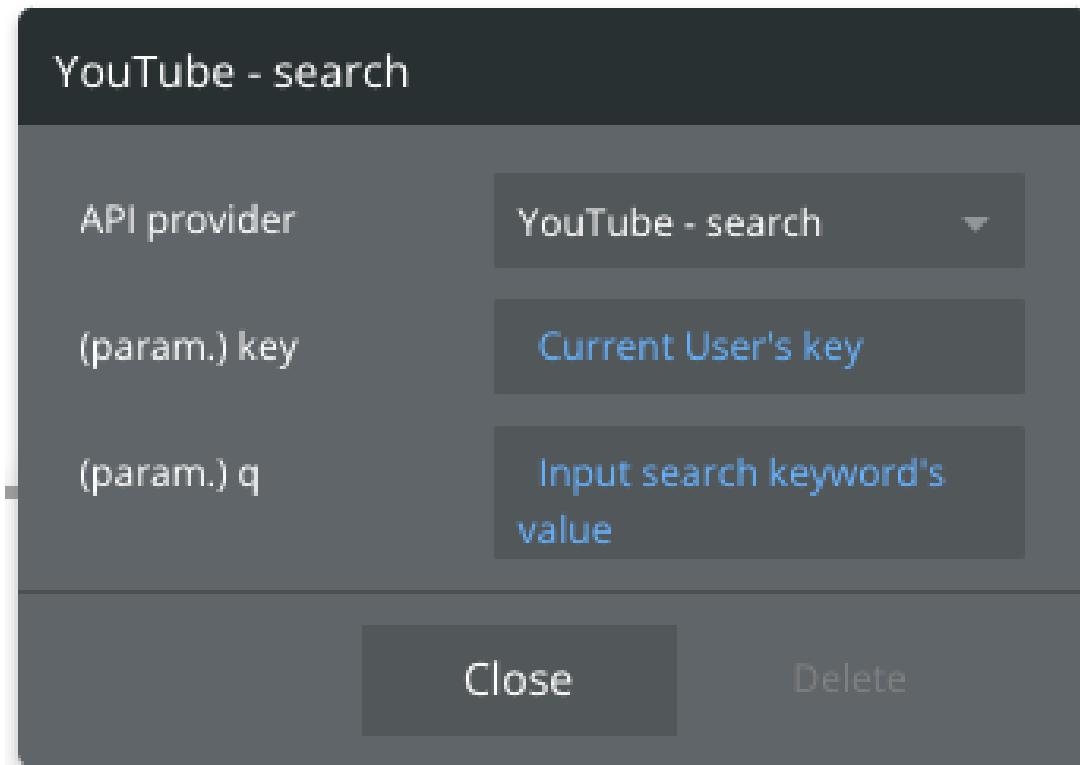
Search for an action...		
 Account	All elements	
 Navigation	Show	Hide
 Data (Things)	Toggle	Animate
 Email	Scroll to	Set state
 Payment	<i>Input</i>	
 Analytics	Set focus	Reset inputs
 Element Actions	<i>Group</i>	
 Plugins	Display data	Reset data
 Custom Events	<i>Repeating Group</i>	
	<u>Display list</u>	Show previous
	Show next	Clear list
	Go to page	Scroll to entry

< Advanced >

- Data source には Get data from an external API を選択し、API 経由のデータ表示とします
- そして API provider に YouTube – search を選択すると、先ほどはなかった 2 つの param が表示されるはずです
- これは先ほど YouTube API 側の設定で動的に指定したいパラメータについて Private のチェックを外したためとなります
- ここで指定する値はいつもの Dynamic data ですね

< Advanced >

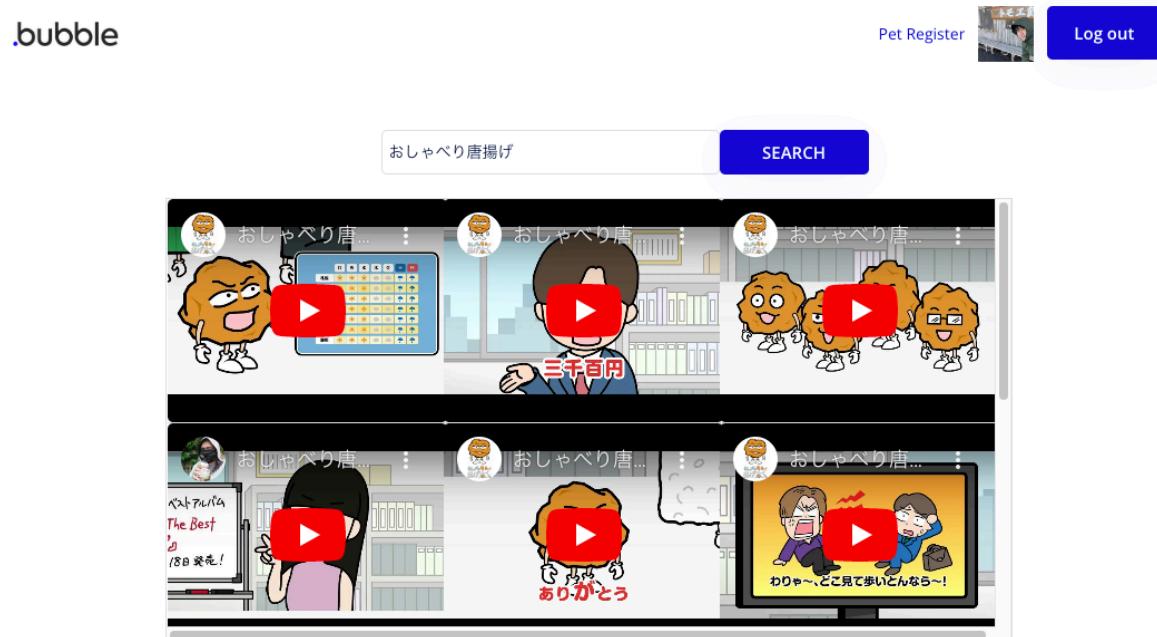
- 設定するとこんな感じです



< Advanced >

プレビューしてみましょう

- ・ログインしている状態で、キーワード欄に入力した動画一覧が表示されましたか？



< Advanced >

- いかがでしたでしょうか？
- API 連携を行うことで、Bubble で出来る幅がさらに広がったと思います！
- 合宿でも外部の API を使うチームがあるかもしれません、そのときは今日学んだことを活かして、API 連携を実践してみましょう！

チームで開発する

Bubble での共同作業の Tips

チームで Bubble を使って共同作業をする時に役立つ情報をお伝えします。

Bubble でのチームメンバーの招待方法

チームメンバーと一つのアプリを共同編集するには...

- チームで一つのアカウントを用意し、そのアカウントをみんなでログインして使いましょうことになります

- Bubble には正規の方法による共同編集機能はついているのですが... 💰💰💰

RECOMMENDED				
Free	Starter	Growth	Team	Enterprise
Best for learning how to use Bubble	Best for launching your app and testing	Best for growing your user base	Best for scaling your team and business	Best for maximum security and scale
\$0 / month	\$32 / month	\$134 / month	\$399 / month	Contact us
Try Free	Get started	Get started	Get started	Talk to sales
Great for: Projects that are under construction	Great for: MVPs and simple tools with small to moderate user bases	Great for: Consumer projects with complex functionality	Great for: Scaling projects with high usage	Great for: Internal tools and customer-facing apps
Free plan features:	Everything in Free, plus:	Everything in Starter, plus:	Everything in Growth, plus:	Everything in Team, plus:
<ul style="list-style-type: none"> ✓ Development version ⓘ ✓ API connector ⓘ ✓ Component library ⓘ ✓ 1 app editor ⓘ ✓ 50k workload units/mo ⓘ ✓ 6 hours of server logs ⓘ 	<ul style="list-style-type: none"> ✓ Live app ⓘ ✓ Custom domain ⓘ ✓ Recurring workflows ⓘ ✓ Basic version control ⓘ ✓ 175k workload units/mo ⓘ ✓ 2 days of server logs ⓘ 	<ul style="list-style-type: none"> ✓ 2 app editors ⓘ ✓ Premium version control ⓘ ✓ Two-factor authentication ⓘ ✓ 10 custom branches ⓘ ✓ 250k workload units/mo ⓘ ✓ 14 days of server logs ⓘ 	<ul style="list-style-type: none"> ✓ 5 app editors ⓘ ✓ Sub apps ⓘ ✓ 25 custom branches ⓘ ✓ 500k workload units/mo ⓘ ✓ 20 days of server logs ⓘ 	<ul style="list-style-type: none"> ✓ Choice of hosting location ⓘ ✓ Centralized admin ⓘ ✓ Dedicated server ⓘ ✓ Priority support ⓘ ✓ Enhanced security ⓘ ✓ Custom workload units ⓘ

Bubble での同時編集の注意点

- 同じエレメントに対する同時編集は避けましょう(後から行われた編集で上書きされます)
 - そのためには、画面ごとに担当者を決めて同じ画面の同時編集を避けるのがおすすめです。
- それ以外の注意点は特にありません。
- 編集は他の人の画面にリアルタイムで反映されます。
 - 画面の編集だけでなく、データベースや Workflow の編集もリアルタイムで反映されます。
- 画面ごとに作業分担すれば、難なく開発できそうです。

チーム開発演習

- DevelopmentPhase のチームで、新たに一つアプリを開発してみてください
 - DevelopmentPhase で開発予定のアプリとは異なるアプリにしましょう
- Adalo か Bubble のどちらを使うかは自由です
 - DevelopmentPhase で使う予定の方を選んでも良いですし、両方を選んでも良いです
- 全員が手を動かすようにしましょう

作業分担のやり方の例

- 画面や機能の洗い出しとデータベース設計は最初にみんなで一緒にやりましょう
- 画面ごとに担当者を決めましょう(登録画面担当、一覧画面担当、詳細画面担当、更新画面担当など)
- 複数人で一つの PC を使って作業をするというやり方もあります。その場合、PC 操作をする人と指示をする人は定期的に交代しましょう



演習結果の発表

チームごとに演習で作ったアプリについて発表してみましょう！

今日のレクチャーは以上になります。

最後に

- Google Cloud Platform で発行した API キーや OAuth の情報を削除しておきましょう

以上です！

お疲れさまでした！