JERZY DĄBRÓWKA

DEMOKRATYZACJATECHNOLOGII.PL

@JDBRWK

Bubble
Dobre praktyki tworzenia aplikacji



PRAKTYCZNA LISTA
Przed Tobą pierwsza realizacja dla Klienta?
Skorzystaj z praktycznej ściągawki.

SPIS TREŚCI

O Autorze

O czym jest ta książka

Interfejs

Bazy danych

Workflows

Bezpieczeństwo i zgodność

SEO

Wtyczki

Deployment

Checklista przed wyjściem na produkcję

Bonus



O Autorze

CZEŚĆ! MAM NA IMIĘ JERZY!

Jestem developerem Bubble, autorem newslettera o nocode i trendach demokratyzujących technologię, nauczycielem Bubble w warszawskim Liceum. Dwa lata temu złapałem bakcyla nocode'owego na tyle mocno, że z czasem rzuciłem dobrze płatną, dyrektorską posadę na rzecz niepewnego i dynamicznego życia freelancera.

Wcześniej VP w Wirtualnej Polsce (Poczta/WP Pilot/Open FM), dyrektor w Operator Chmury Krajowej.

Serce mam produktowe i największą radość w Bubble przynosi mi budowanie oraz szybkość z jaką można dostarczać efekt. Czasami jednak warto zatrzymać się na chwilę i upewnić, że nasza aplikacja budowana jest na solidnych fundamentach. O tym będzie ta książka.

O tej książce



Jest to subiektywna lista porad dotyczących budowania aplikacji w Bubble. Nie utarła się jeszcze oficjalna ani promowana przez jakąś grupę lista dobrych praktyk, więc te wybrane są na podstawie moich własnych doświadczeń.

Pozycja jest przeznaczona dla osób które już mają zbudowane pojęcie o tworzeniu aplikacji w Bubble i chcą zadbać o jakość i bezpieczeństwo tworzonych rozwiązań.

Porady zebrałem na podstawie własnych obserwacji lub wielu wielu godzin spędzonych na czytaniu różnych źródeł. Dołożyłem staranności by w miarę możliwości przetestować wszystkie zalecenia, ale dopuszczam możliwość, że coś mi umknęło. Ponadto Bubble jest platformą dynamicznie rozwijaną w kierunku optymalizowania działania aplikacji - proponowane rozwiązania mogą być w trakcie lektury nieaktualne. Najlepszym rozwiązaniem zawsze jest samodzielne przetestowanie danego rozwiązania.

INTERFEJS

Style

Należy unikać tworzenia elementów których wygląd nie jest określony przez Style (ze względu na późniejsze utrzymanie takiego rozwiązania)

Korzystanie ze styli też pozytywni wpływa na szybkość działania aplikacji. Niezależne stylowanie każdego z elementów to po prostu więcej kodu który Bubble musi wyrenderować a przeglądarka pobrać.

Element tree i nazywanie elementów

Bubble domyślnie proponuje pewną konwencję nazywania elementów - guziki nazywa Button, grupy - Group etc. Z czasem zacząłem je skracać - zwłaszcza w wypadku Repeating Group do “rg” - dzięki temu więcej miejsca w inspektorze Elements Tree jest przeznaczone na funkcje elementu. Idąc dalej tym tropem, grupa to “g”, guzik to “btn”, checkbox to “chb”. Dokładna konwencja jest drugorzędna - istotne by trzymać się obranej ścieżki - to znacząco ułatwi czytanie aplikacji po przerwie lub osobom trzecim.

Element tree to kluczowy obszar edytora Bubble gdy zaczynasz prace z responsywnymi stronami - metoda nawigacji nauczona z layoutem "fixed" nie będzie się sprawdzała przy elementach wypełniających cały rozmiar "rodzica". Element Tree obsługuje drag’n’drop.

Bubble posiada dwie funkcje kopiowania. Kopiowanie elementów oraz kopiowanie elementów wraz z korespondującymi Workflows. Druga opcja dostępna jest w kontekstowym menu edycji.

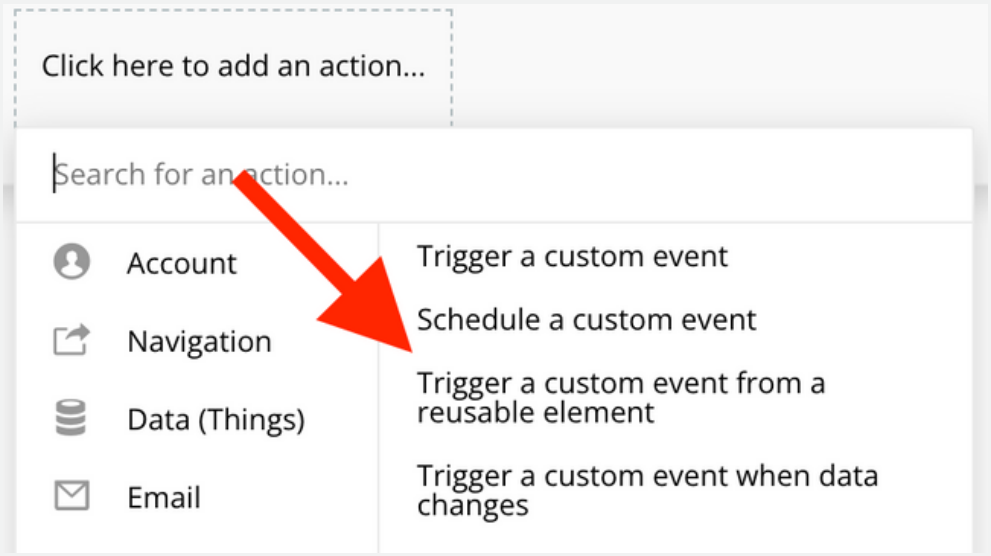
Reusable

Początkowo nie przepadałem za reusable - denerwowało mnie, że trudno “komunikuje się” między stroną a reusable. Ale w momencie gdy zacząłem jedynie korzystać z Parametrów URL w nawigacji to ta bariera przestała istnieć.

Reusable to bardzo potężna funkcja Bubble. Warto z nich korzystać gdy będziemy dany zestaw elementów wykorzystywać wielokrotnie na stronie. Polecam podzielenie aplikacji na “widoki” za pomocą Reusable. Każdy widok to odrębny Reusable.

Zwróć uwagę, że gdy dzielisz aplikację na reusable tworzysz niejako “mikroserwisy” bowiem każdy Reusable ma własne Workflows. Ułatwia to przyszłe debugowanie aplikacji - wiesz, że dane biznesowe Workflows będą w danym Reusable.

W oficjalnej dokumentacji Bubble sugeruje o tworzeniu "pustych" Reusable które zawierają jedynie Workflow i wstawianie ich na strony - dzięki temu osiągamy efekt "funkcji" znanej z tradycyjnego programowania.

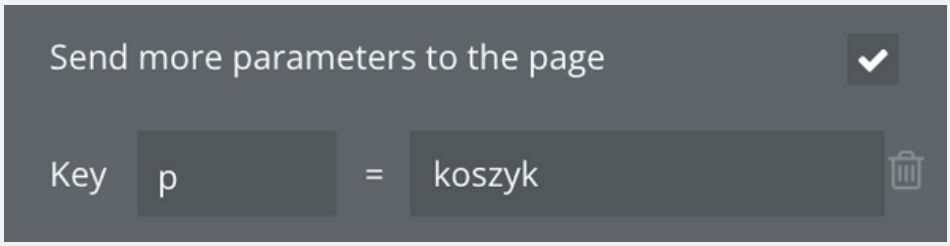


Parametry URL

Ukrywaj i wyświetlaj kluczowe elementy za pomocą parametrów URL.

Bubble nie uczy domyślnie nawigacji z wykorzystaniem parametrów URL ale jeśli chcemy budować Single Page App (SPA) to parametry są świetnym rozwiązaniem.

Tworzymy je za pomocą Workflow “Go to page...”, określamy Destination jako index (lub inną stronę która jest naszą główną stroną) i zaznaczamy checkbox “Send more parameters to the page”

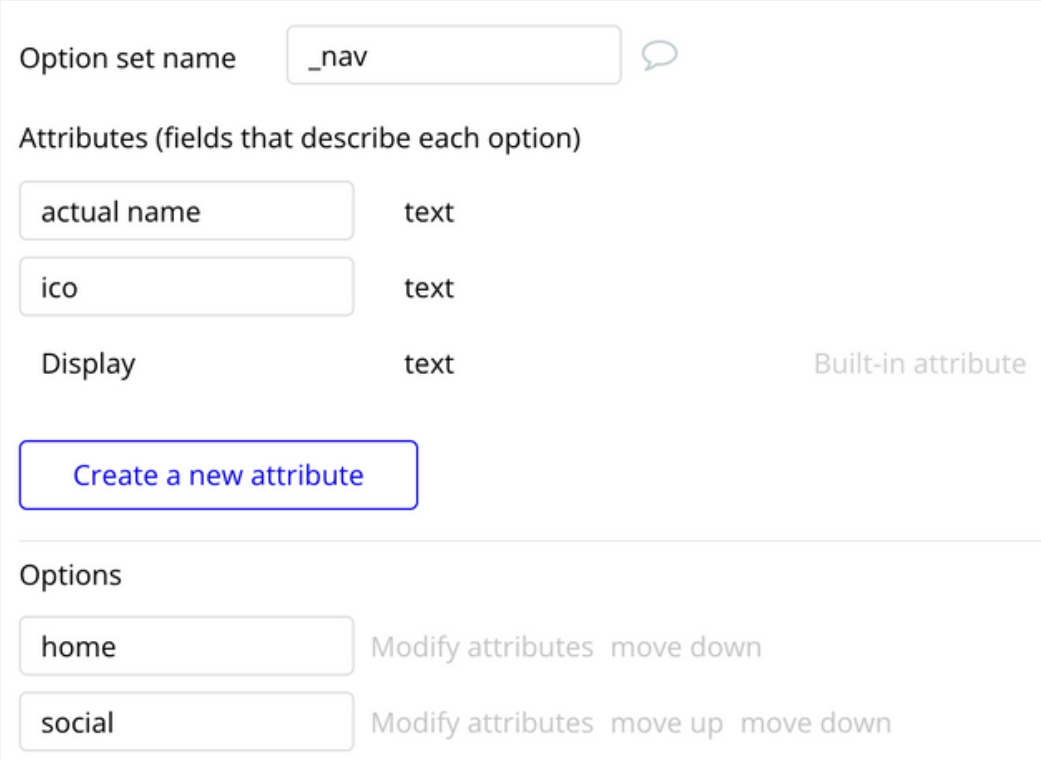


Jednak utrzymanie takiego rozwiązania jest trudne. Dlatego należy trzymać strukturę nawigacji w Option Set

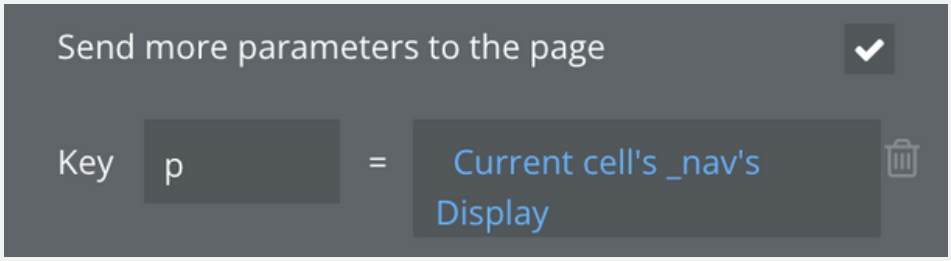
Nawigacja w Single Page Apps

Nawigacja oparta na Parametrach URL wykorzystywana w SPA opiera się na następujących komponentach

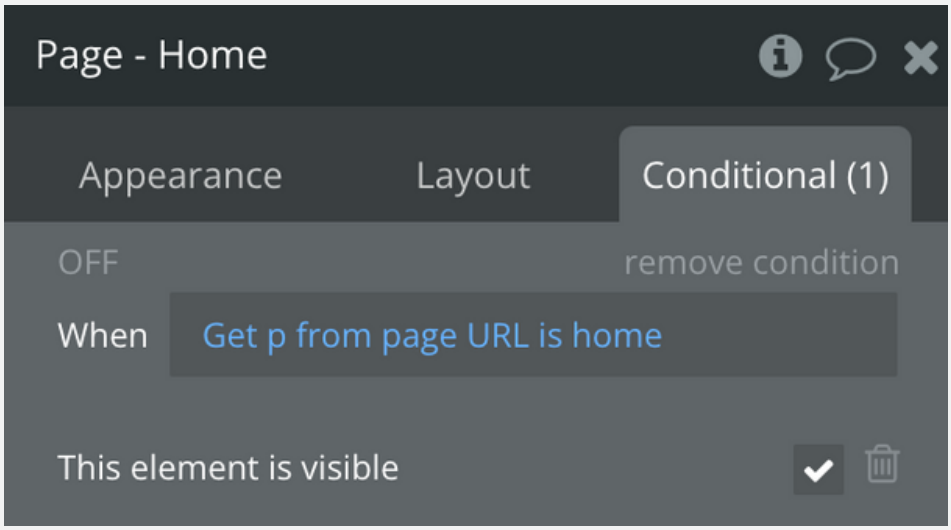
Lista stron zapisana w Option Set. By funkcja nawigacji działała wystarczy domyślny atrybut Display, ale możesz wykorzystać to rozwiązanie by upiec dwie pieczenie na jednym ogniu. W moim przypadku “Actual name” to ładna nazwa wyświetlana na guziku lub jako tytuł strony; “Ico” to tag do ikony w Iconify. Możesz tu dodać kolor asocjowany z daną sekcją, zdjęcie czy tłumaczenie tego klawisza.



Menu budowane jako Repeating Group z Data Source wskazującym Option Set. Komórka RG uruchamia Workflow “Go to page...” dodający parametry zgodny z Option Set



Element interfejsu - najlepiej Reusable całego widoku - domyślnie ukryty i collapsable z Conditionalem pokazującym ten element na podstawie parametru URL



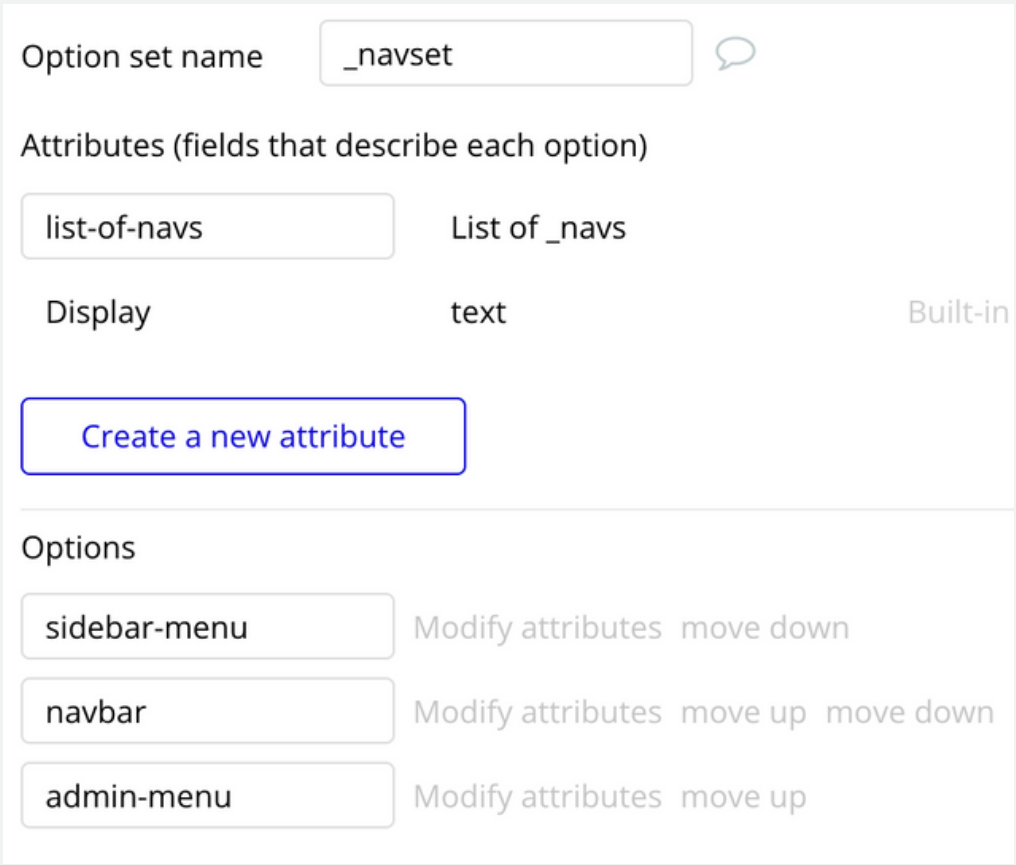
Reasumując, Option Sety trzymają słownik zapewniający, że nie popełnimy literówki wpisując słowo kluczowe w którymś z elementów. Trzymanie parametrów jako Option Set ułatwia też utrzymanie aplikacji - parametry te są w jednym miejscu. Jeśli chcemy parametr login zmienić na sign-in nie musimy szukać wszystkich linków menu oraz wszystkich conditionali pokazujących grupy które do tego parametru się odwołują - zmieniamy tylko wording w Option Set.

Zarządzanie wieloma menu

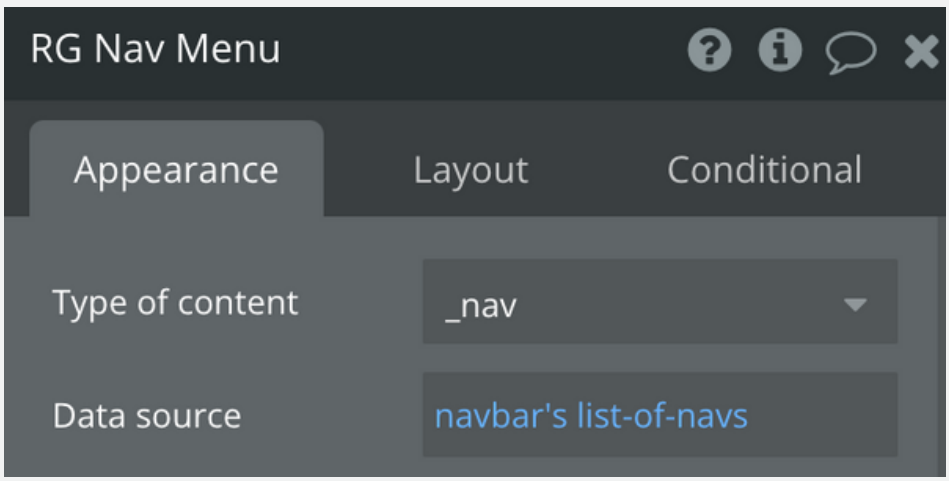
Czasami chcemy mieć więcej niż jedno menu na stronie. Czy to ze względu na projekt strony czy przez kwestię uprawnień użytkowników. Option Sety rozwiązują ten problem

Tworzymy nowy Option Set nazwany _navset a w nim atrybut list-of-navs (choć nazwa to naturalnie sprawa indywidualna) który jest listą Option Seta _nav.

Jako Opcje dodajemy różne Menu jakich potrzebujemy. Edytując (Modify attributes) je określamy jakie _navs mają w nich być. Jedynym minusem tego rozwiązania jest brak wygodnego zarządzania kolejnością dodanych tak _navs. Trzeba je skasować i dodawać od nowa w oczekiwanej kolejności.



Tworząc Menu oparte na Repeating Group jako Data Source wybieram nie _nav ale _navset



KONIEC BEZPŁATNEGO FRAGMENTU

Dziękuję za pobranie bezpłatnego fragmentu e-booka "Dobre praktyki projektów realizowanych w Bubble". Mam nadzieje, że informacje tu przedstawione były dla Ciebie interesujące.

Jeśli nie chcesz przegapić informacji o premierze pełnego wydania - zapraszam do zapisania się do newslettera na www.demokratyzacjatechnologii.pl