AppInventor – pierwsze programy

rezultat projektu

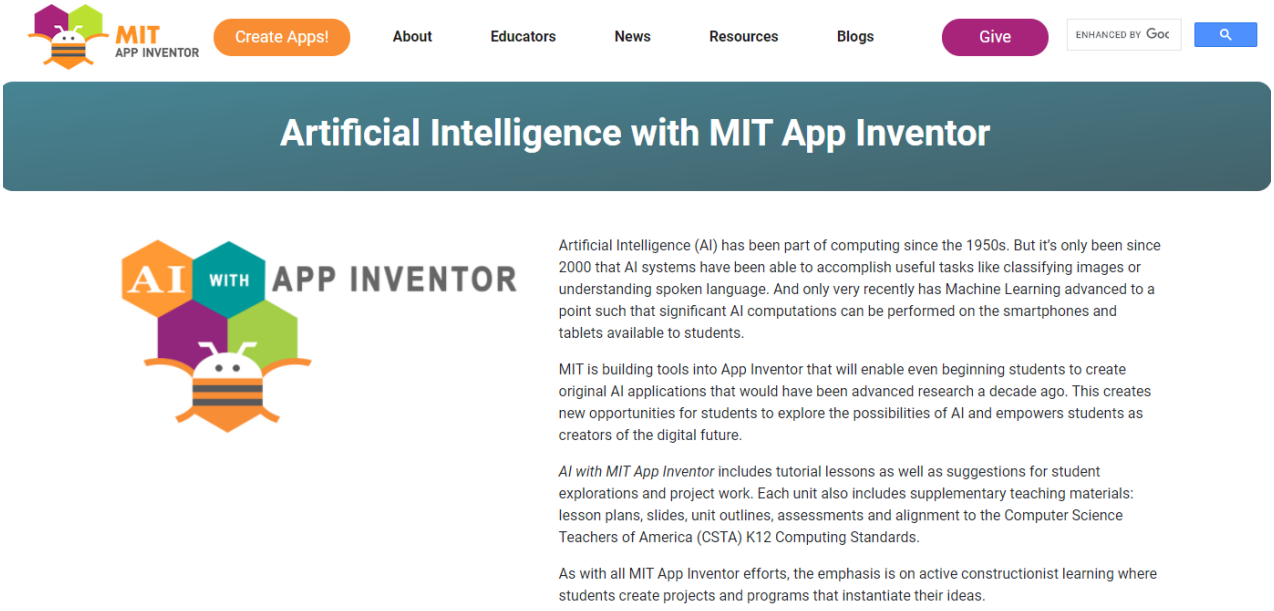
Nowe Technologie wsparciem dla edukacji

nr umowy - POWERSE-2018-1-PL01-KA101-049291

realizowanego ze środków POWER na zasadach programu Erasmus+

sektor Edukacja szkolna

„Ponadnarodowa mobilność kadry edukacji szkolnej”



Wprowadzenie

App Inventor jest narzędziem dostarczanym przez Google (obecnie projektem zajmuje się Massachusetts Institute of Technology MIT), służącym do tworzenia aplikacji na platformę Android. Tworzenie własnych aplikacji jest proste, nie wymaga znajomości języka programowania, nie wymaga instalowania dodatkowego oprogramowania na komputerze. Aby tworzyć własne aplikacje wystarczy przeglądarka internetowa.

App Inventor oferuje graficzny interface, podobny do aplikacji Scratch, która pozwala użytkownikowi budować własną aplikację z gotowych obiektów z wykorzystaniem mechanizmu “drag-and-drop”.

Tworzenie aplikacji w appInventorze następuje dwuetapowo: pierwszy etap polega na projektowaniu wyglądu aplikacji, wykorzystując gotowe kontrolki z programu App Inventor Design, a drugi etap, to tworzenie warstwy logicznej aplikacji z wykorzystaniem App Inventor Blocks Editor.

W jaki sposób rozpocząć pracę z AppInventorem? Są trzy możliwości korzystania z aplikacji.

Sposób 1 – (rekomendowany, z takiego będziemy korzystać). Potrzebujemy: konto na Google, komputer, telefon lub tablet z zainstalowanym systemem Android, sieć WiFi, do której są podłączone komputer i urządzenie mobilne (ważne - oba urządzenia muszą być podłączone do tej samej sieci WiFi). Na urządzeniu mobilnym instalujemy aplikację MIT AI2 Companion.

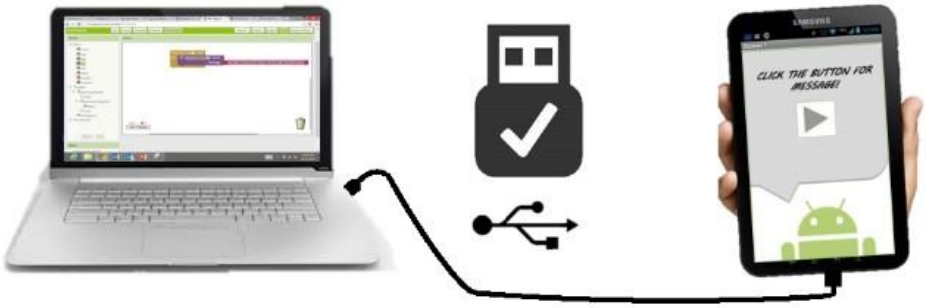
Tworzymy aplikację na komputerze

Testujemy aplikację na urządzeniu mobilnym

Sposób 2 – Jeżeli nie korzystamy z urządzenia mobilnego, możemy użyć emulatora

Tworzymy aplikację na komputerze, a następnie testujemy na emulatorze

Sposób 3 – komputer łączymy z urządzeniem mobilnym przez USB



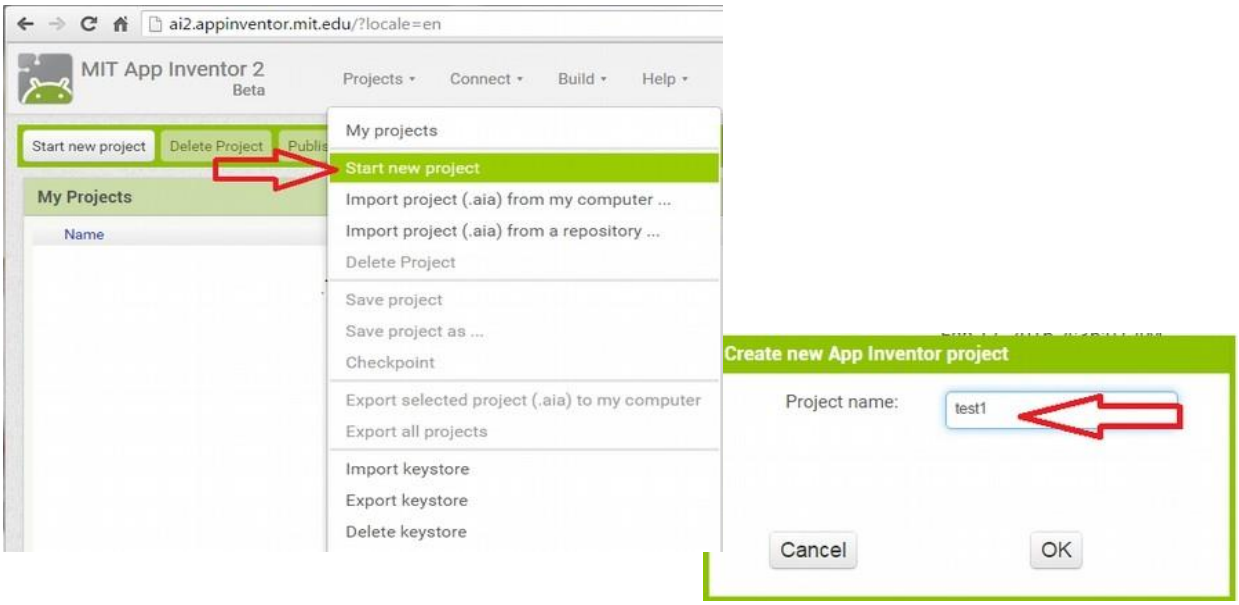
Budujemy aplikację na komputerze

Testujemy aplikację na urządzeniu mobilnym

Pierwszy projekt

Celem pierwszego projektu jest pokazanie kluczowych etapów budowania aplikacji oraz jej testowania na urządzeniu mobilnym, zapoznanie z podstawowymi obiektami palety komponentów oraz sposobem definiowania zdarzeń w trybie **Blocks**. Nasza aplikacja będzie zawierała button, po kliknięciu którego zmieni się kolor tła przycisku na niebieski.

- Otwieramy stronę http://ai2.appinventor.mit.edu/
- Projects->Start new projects ->wpisujemy nazwę projektu

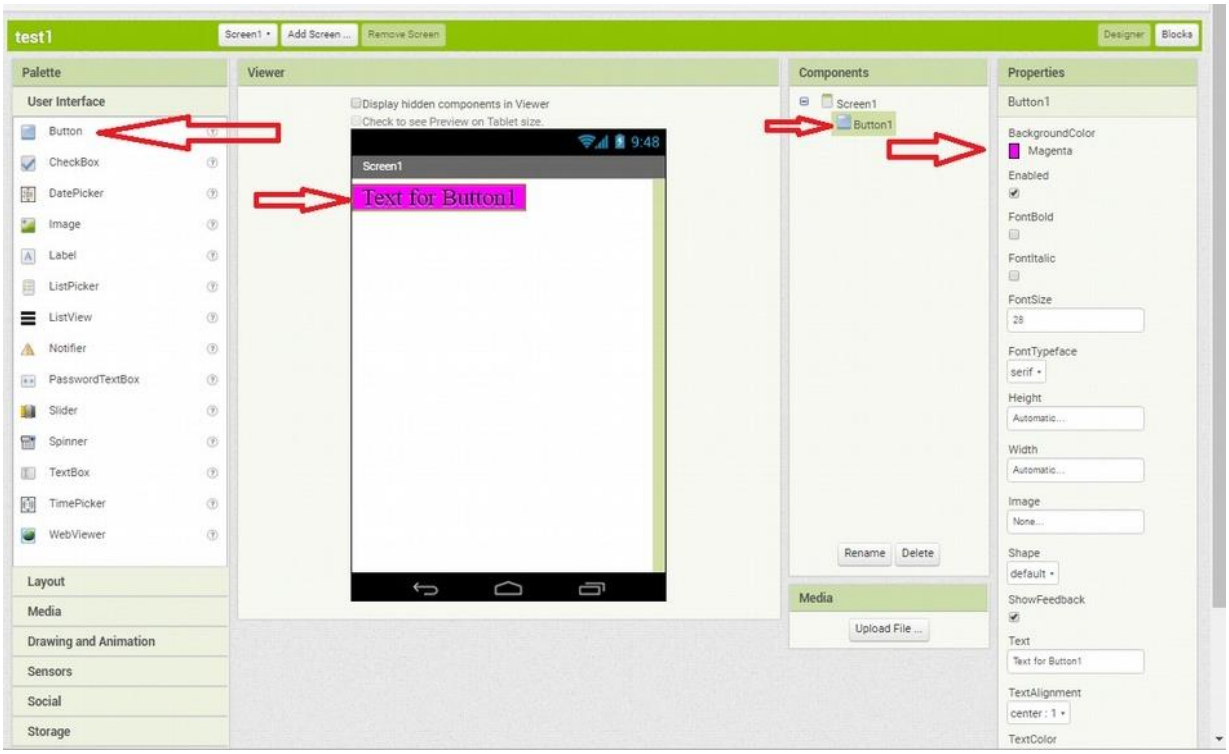


Tworzenie aplikacji w App Inventorze wymaga korzystania z trzech głównych elementów:

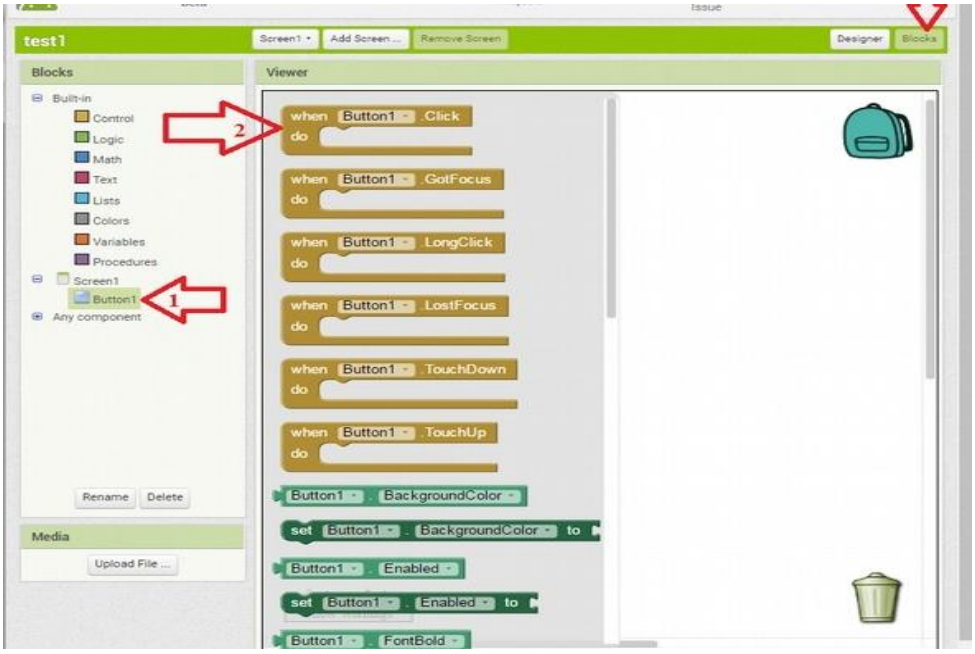
a) **Component Designer** (Rys.1). Mamy do dyspozycji paletę komponentów (lewa strona), które metodą „drag-and-drop" przenosimy na wirtualne urządzenie mobilne. Każdemu elementowi naszej aplikacji możemy dowolnie zmieniać wartości jego atrybutów.

b) **Blocks Editor** (Rys. 2). Klikając Buttons przełączamy się do okna programowania zdarzeń, które zachodzą w wyniku reakcji użytkownika programu (np. kliknięcie na button, Rys. 3, Rys.4)

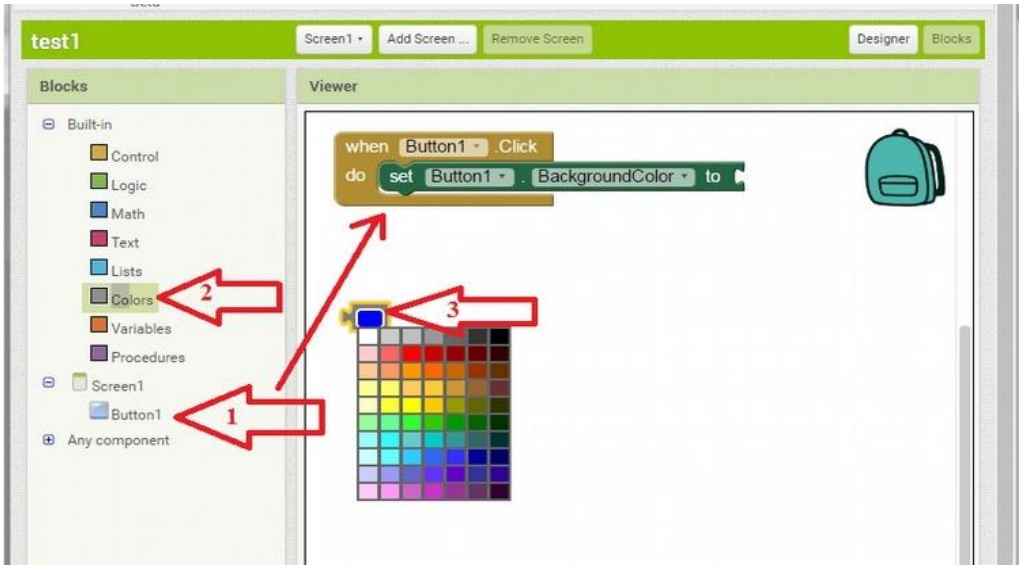
c) Urządzenie mobilne do testowania aplikacji.



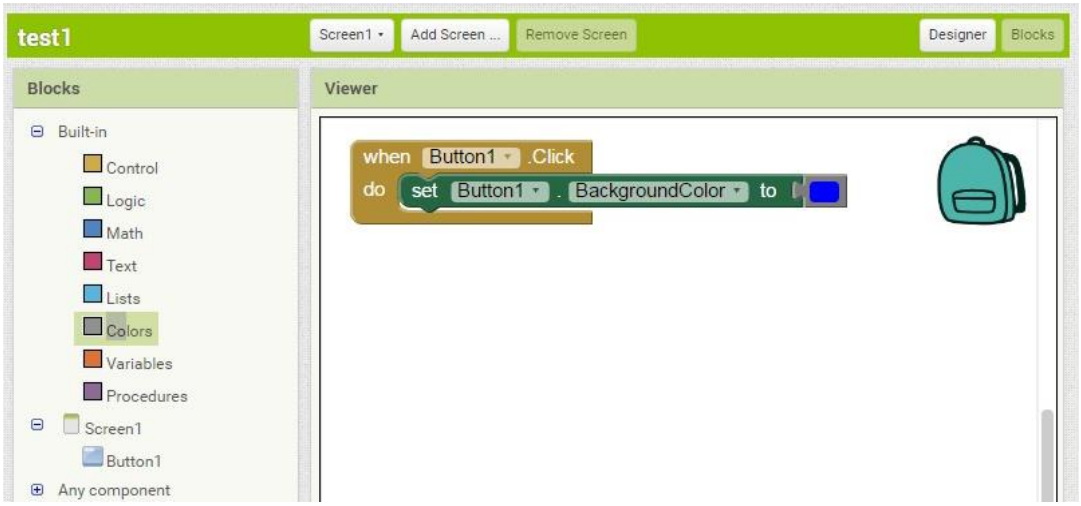
Rys. 1: Component Designer. Dowolnie ustawiamy własności komponentów



Rys. 2: Blocks Editor



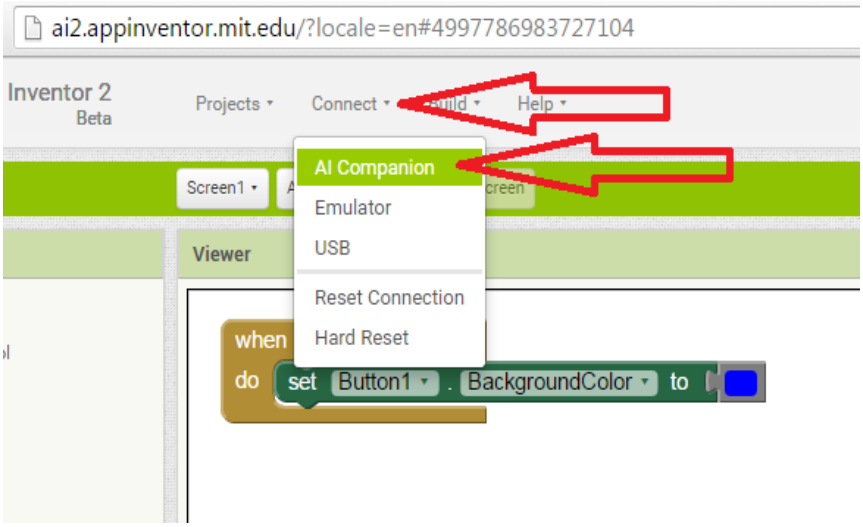
Rys. 3: Programowanie zdarzeń



Rys. 4: Po kliknięciu na button zmieni się jego kolor

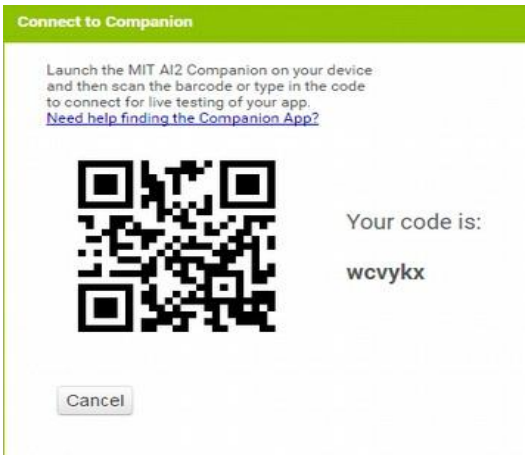
Uruchamiamy aplikację na urządzeniu mobilnym. W tym celu należy sprawdzić, czy oba urządzenia: komputer i urządzenie z Androidem są podłączone do tej samej sieci WiFi, następnie na urządzeniu mobilnym włączamy zainstalowaną wcześniej aplikację ***MIT AI2 Companion***.

Następnie w App Inventorze, z górnego menu wybieramy *Connect->AICompanion* (Rys. 5)



Rys. 5: Kliknij Connect a następnie AI Companion

Pojawi się okno z kodem QR (Rys. 6), który należy przepisać lub zeskanować wykorzystując zainstalowany na urządzeniu mobilnym program ***MIT AI2 Companion***.

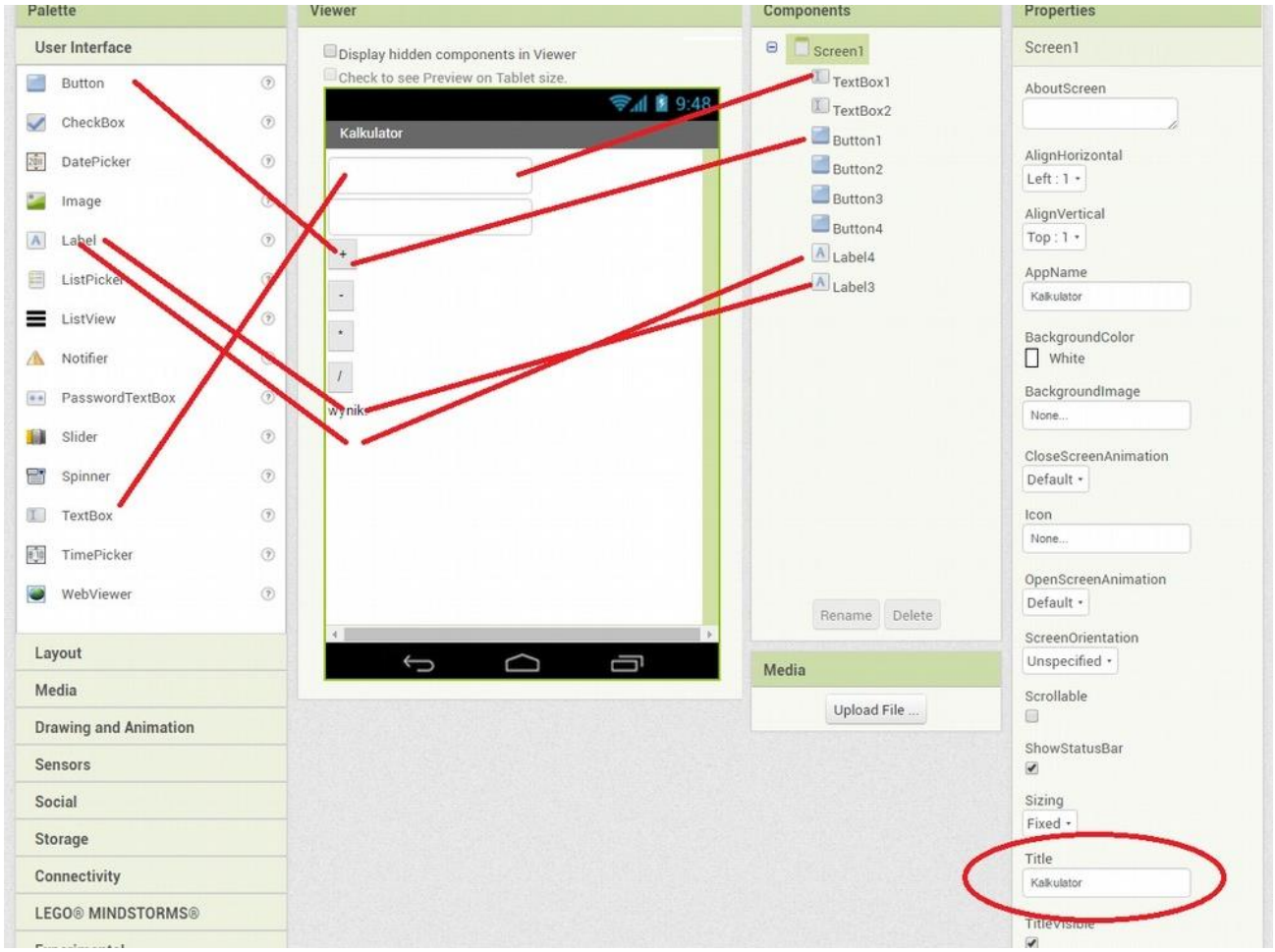


Rys. 6: Ten kod należy zeskanować

Aplikacja „kalkulator”

Działanie komponentów przetestujemy tworząc prosty kalkulator. W tym celu:

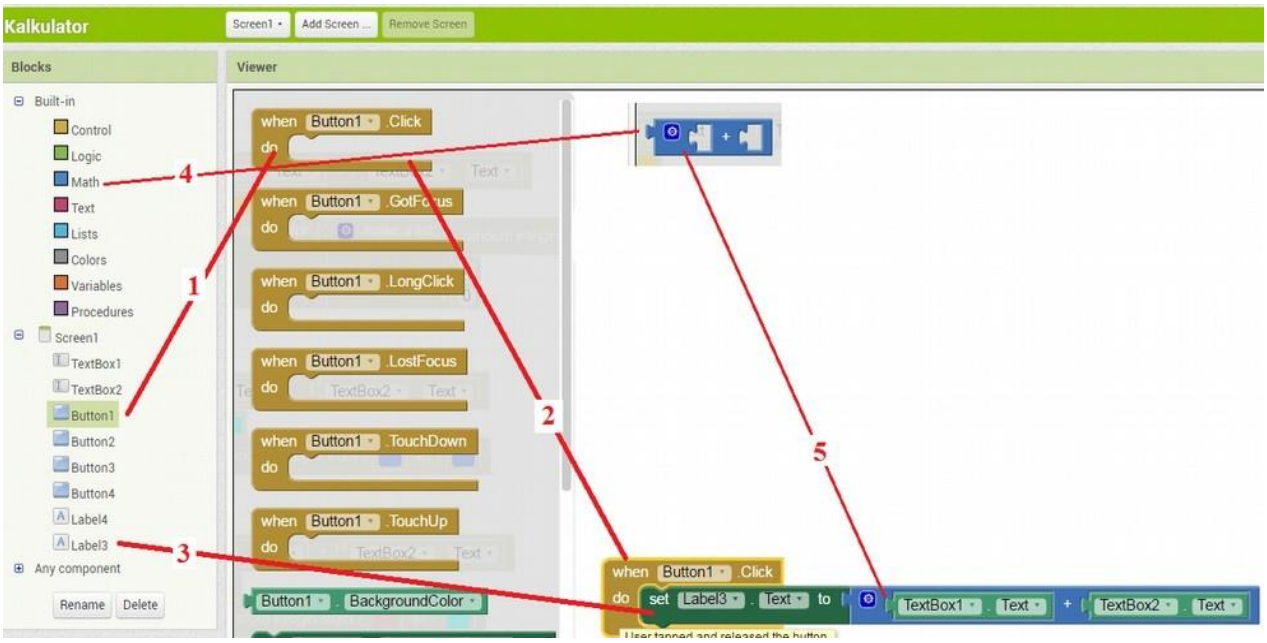
- Otwieramy stronę http://ai2.appinventor.mit.edu/
- Projects->Start new projects ->wpisujemy nazwę projektu, np. Kalkulator
- Przenosimy komponenty z części Palette do Viewer, tak jak na Rys. 7.



Rys. 7: Komponenty naszej aplikacji

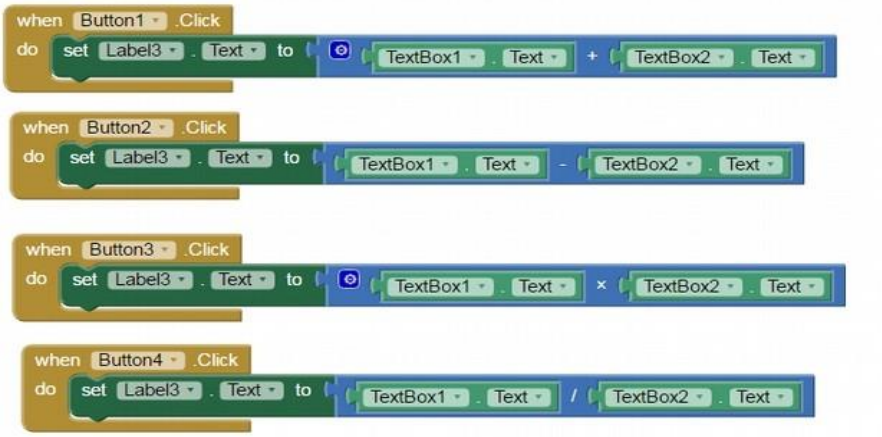
- Ustawiamy wartości dla własności komponentów: zaznaczamy komponent w części ***Components*** i zmieniamy wartość odpowiedniego atrybutu w części ***Properties***. Zaznaczamy komponent Screen1 i ustalamy wartość atrybutu „Title” na „Kalkulator” (rys. 7). Dla pozostałych komponentów ustalamy wartość atrybutu „text” na wartości, zgodnie z rys. 7.

Po pierwszej części- projektowanie interfejsu aplikacji, przechodzimy do zakładki Blocks, aby zająć się oprogramowaniem odpowiednich zdarzeń.



Rys. 8: Oprogramowanie zdarzenia Button1.Click

Podobnie zdefiniujemy zdarzenia dla pozostałych działań matematycznych reprezentowanych przez odpowiednie buttony (Rys. 9)

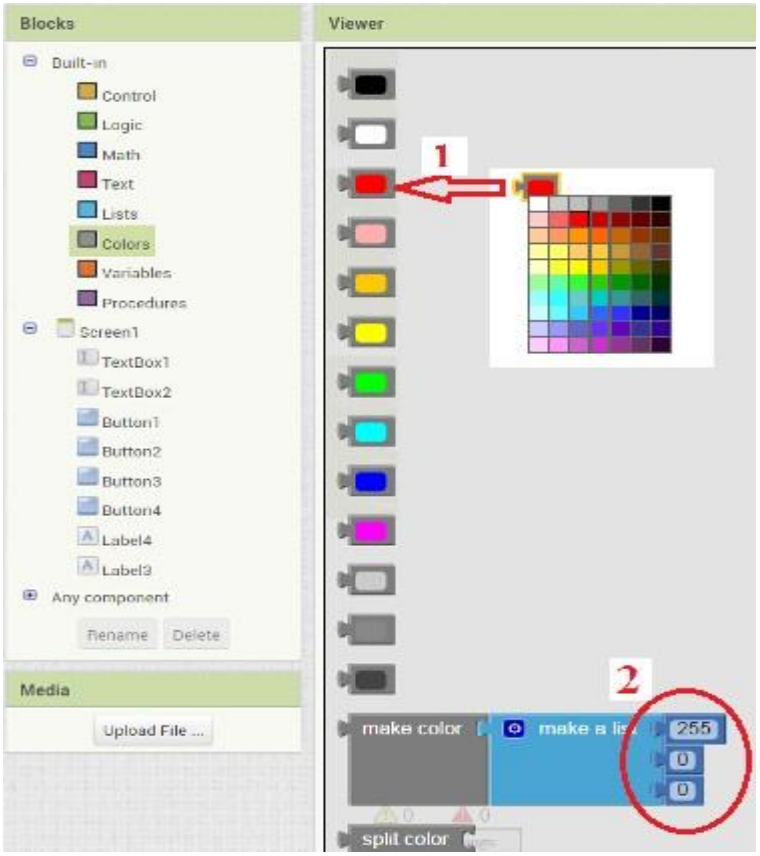


Rys. 9: Definicje zdarzeń obsługujących przyciski

Nasza aplikacja jest na tyle prosta, że bez utraty przejrzystości kodu możemy dołączyć dodatkowe funkcjonalności, dzięki czemu poznamy inne komponenty AppInventora.

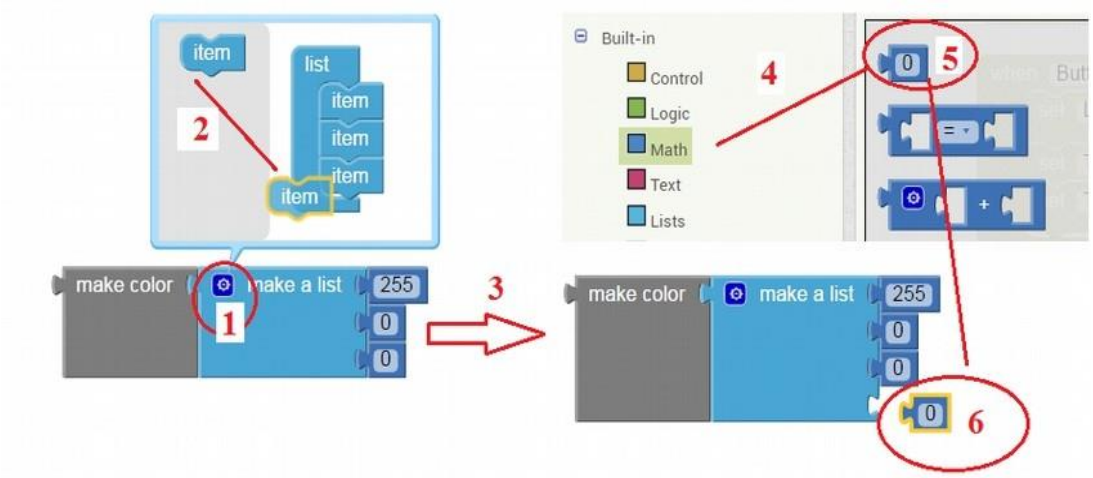
Opis wszystkich bloków znajdziemy na stronie:
http://appinventor.mit.edu/explore/ai2/support/blocks

W tej części budowy aplikacji zajmiemy się kolorami. Kolory są dostępne w części ***Blocks/Colors***. Kolory można wybrać z palety kolorów (Rys. 10). Po dwukrotnym kliknięciu na kolorowy prostokąt (1) można wybrać jeden ze zdefiniowanych kolorów. Kolory można wybierać definiując je w formacie RGB (red, green, blue) podając dla każdego koloru wartość całkowitą z zakresu <0; 255>. Wartości wpisujemy jako elementy listy (2).



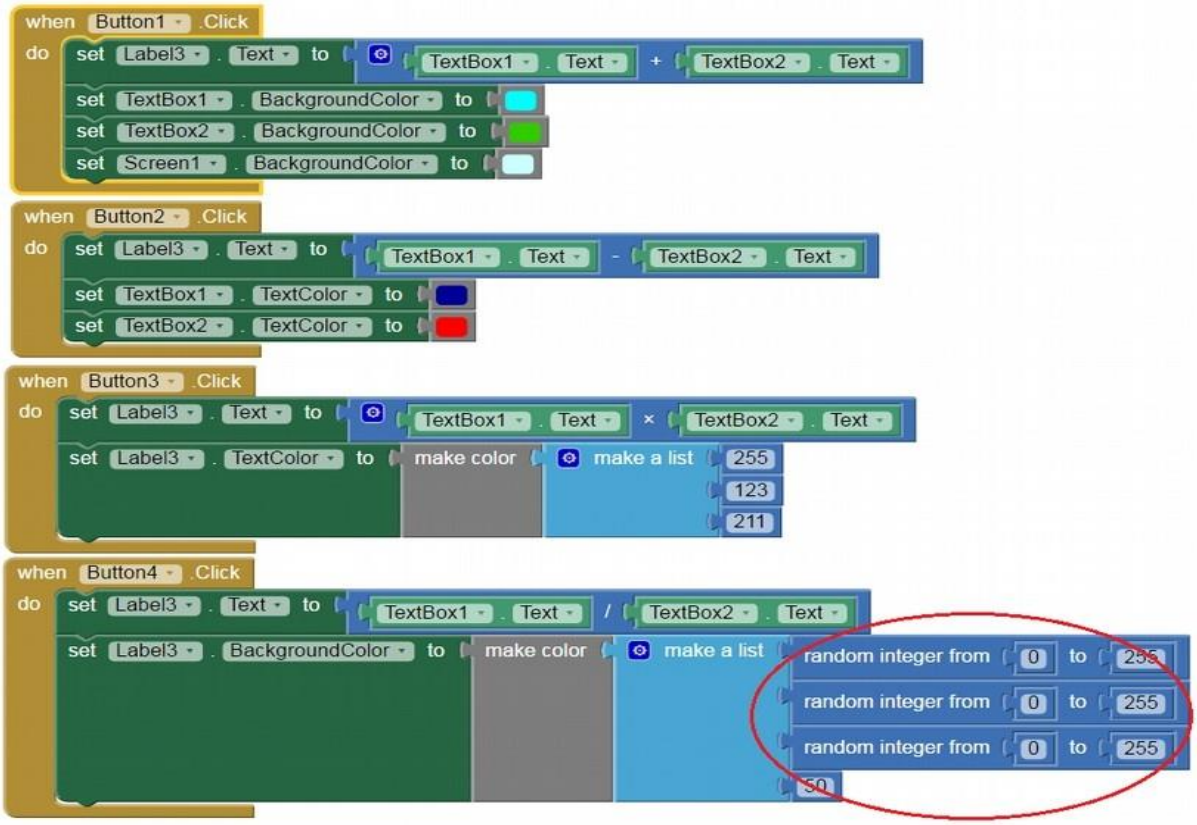
Rys. 10: Narzędzie do definiowania kolorów

Kolory możemy definiować w formacie RGBA (red, gree, blue, alpha) (Rys. 11). Kliknięcia na granatowy znak (1) umożliwi dodanie kolejnej pozycji do listy(2). W ten sposób utworzy nam się lista (3) z brakującym elementem. Brakującym elementem listy jest liczba, której dodanie umożliwi komponent z grupy Math (4). Przenosimy niebieski element (5) do listy (6). Wartość parametru alpha jest liczbą całkowitą z zakresu <0; 100> Wartość 0 (0%) oznacza, że piksel jest całkowicie przezroczysty, przyjmuje kolor tła. Wartość 100 (100%) oznacza, że piksel będzie całkowicie widoczny- jak w zwykłym obrazie cyfrowym bez kanału alpha. Wartości pośrednie powodują, że piksel będzie częściowo przezroczysty i będzie przepuszczał kolor tła.



Rys. 11: Definiowanie koloru w formacie RGBa

Zmienimy kolory tła oraz tekstu dla komponentów naszej aplikacji.



Rys. 12: Kolorujemy elementy

Uzupełnienia wymaga sposób definiowania koloru tła dla elementu Label3 Label3.BackgroundColor (Rys. 12). Paleta kolorów RGB powstaje poprzez losowy wybór liczb definiujących kolory. Uzyskanie całkowitych liczb losowych umożliwia blok **random integer from..** znajdujący się w grupie **Math**. W odpowiednie miejsce należy wpisać wartości graniczne **from** oraz **to**. Kolejność podania argumentów nie ma znaczenia.

Podsumowanie:

Poznalismy metody pobierania informacji od użytkownika, wykonania działań na liczbach rzeczywistych. Nauczyliśmy się wykorzystywać zdefiniowane kolory a aplikacji oraz poznaliśmy sposób definiowania własnych kolorów. Poznaliśmy metodę uzyskania liczb całkowitych losowych.

Zadanie:

Uzupełnić aplikację o możliwosc obliczania potęgi, pierwiastka, konwersji liczby z systemu dziesiątkowego do hexadecymalnego