

1. Podstawy systemu App Inventor

1.1. Do czego i na co nam App Inventor?

Serwis i oprogramowanie App Inventor (w tym kursie dalej nazywane w skrócie AI2) służy do tworzenia aplikacji mobilnych na system Android ale jednocześnie jest doskonałym narzędziem do nauki programowania. W łagodny sposób wprowadza w sztukę tworzenia programów, w tym wypadku bez jednej linii kodu! Wszystko na bazie graficznych bloków, które w sensowny, spójny i logiczny sposób należy ze sobą wiązać aby stworzyć działający program. App Inventor to w sumie trzy elementy:

- serwis WWW w którym tworzymy aplikacje
- emulator smartfona z Androidem
- aplikacja na smartfon dla połączenia z serwisem i testowania tworzonych programów

Serwis WWW i jego usługi działają w chmurze tak więc wszystkie narzędzia programistyczne jak i nasze projekty przetrzymywane są tamże. Pozwala to pracować z dowolnego miejsca i komputera. Jedynym wymogiem jest posiadanie konta na GMail aby zalogować się do App Inventora.

Jeszcze jedna uwaga: w tym kursie używa się zamiennie słów aplikacja i program oraz tworzenie, budowanie, pisanie programu.

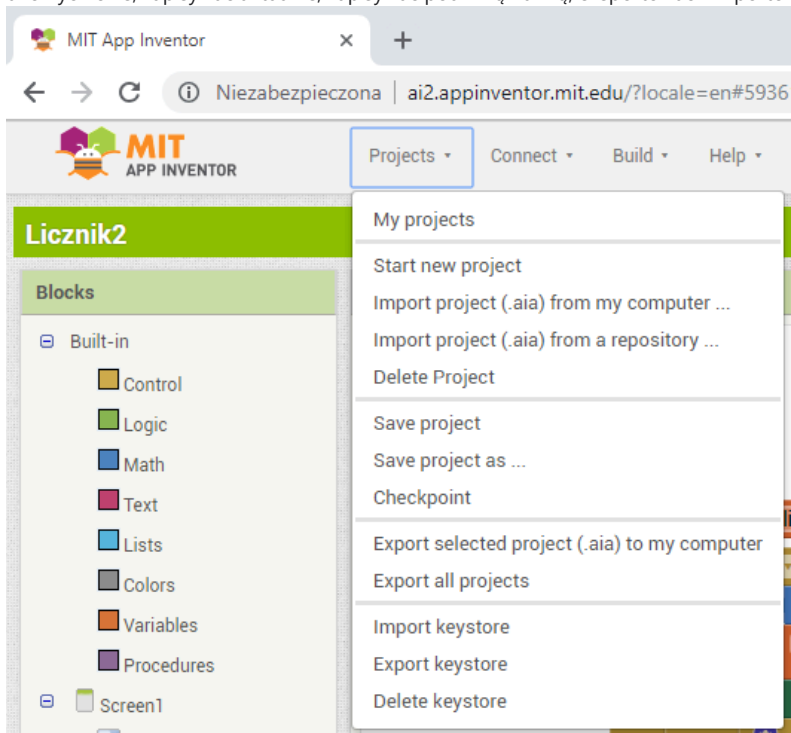
Przydatne linki:

<http://ai2.appinventor.mit.edu>

<http://www.appinventor.org>

1.2 Obsługa plików i projektów

Tworzone projekty domyślnie przechowywane są w "chmurze" (udostępnianej przez twórcę tej aplikacji uczelnię MIT) za pomocą menu Projects możemy tworzyć nowe, zapisywać aktualne, zapisywać pod inną nazwą, eksportować i importować do naszego komputera aby zarządzać nimi lokalnie.



Dla nas najważniejsze pozycje menu to

Projekty:

- Projects/My Projects: lista naszych projektów przechowywana na MIT
- Projects/Delete project: kasowanie projektu
- Projects/Save project: zapisanie w chmurze aktualnie otwartego projektu
- Projects/Save project as: zapisanie w chmurze aktualnie otwartego projektu pod nową nazwą (duplikacja np w celach testowych, rozbudowy programu zachowania starszej wersji)
- Projects/Export selected project to my computer: wyeksportowanie aktualnego projektu do naszego komputera (np. celem lokalnej archiwizacji lub przekazania tworzonej aplikacji innej osobie)
- Projects/Import project from my computer: import projektu z mojego komputera i zapisanie w chmurze MIT

Połączenia:

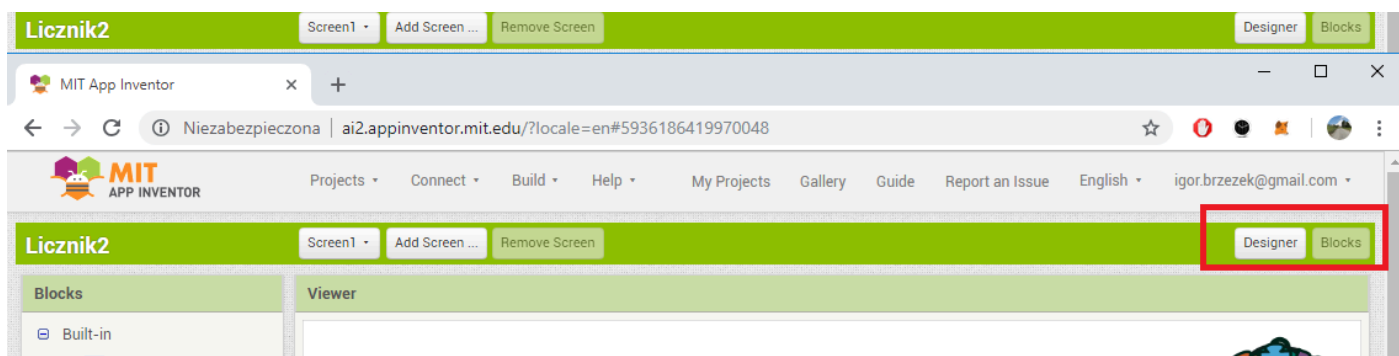
- Connect/AI Companion: połączenie ze smartfonem po WiFi
- Connect/Emulator: połączenie z emulatorem Androida pracującym na naszym komputerze (wymaga wcześniejszego włączenia programu aiStarter)
- Connect/Reset connection: zrestartowanie połączenia z emulatorem lub ze smartfonem (wymaga ponownego uruchomienie programu Companion na smartfonie)

Jeśli chcemy utworzyć aplikację instalacyjną na Androida lub na Google Play mamy do dyspozycji menu Build. Tak wygenerowany plik można skopiować do smartfona i zainstalować (musi być włączona opcja pozwalająca na instalację z niezaufanych źródeł).

1.3 Podstawowe widoki (okna)

Po wejściu do serwisu AI2 mamy do dyspozycji dwa główne okna:

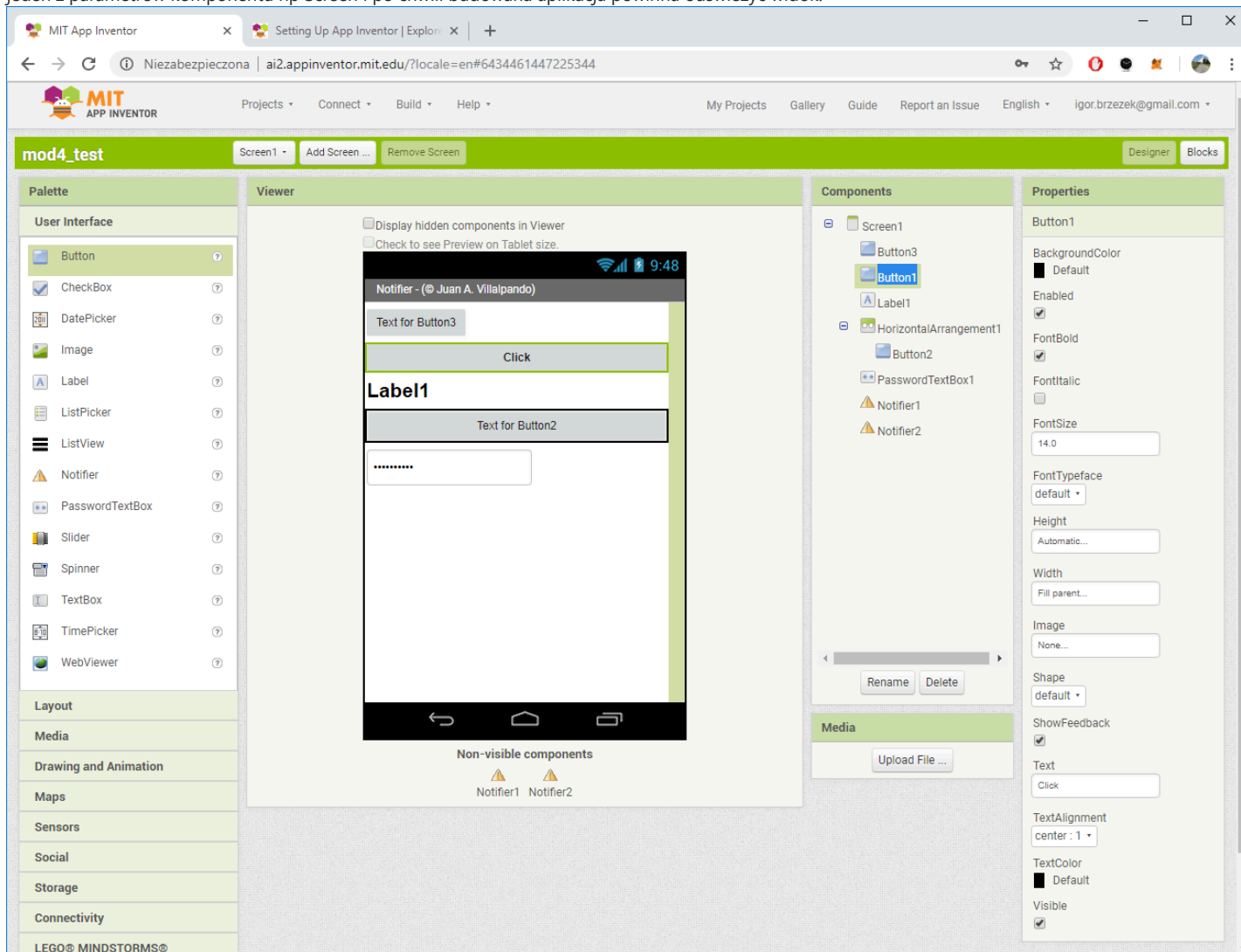
- Designer (projektant)
- Blocks (bloki kodu)



Okno *Designer* pozwala na projektowanie interfejsu użytkownika czyli wyglądu aplikacji. Okno *Blocks* pozwala na projektowanie działania (programowanie) aplikacji.

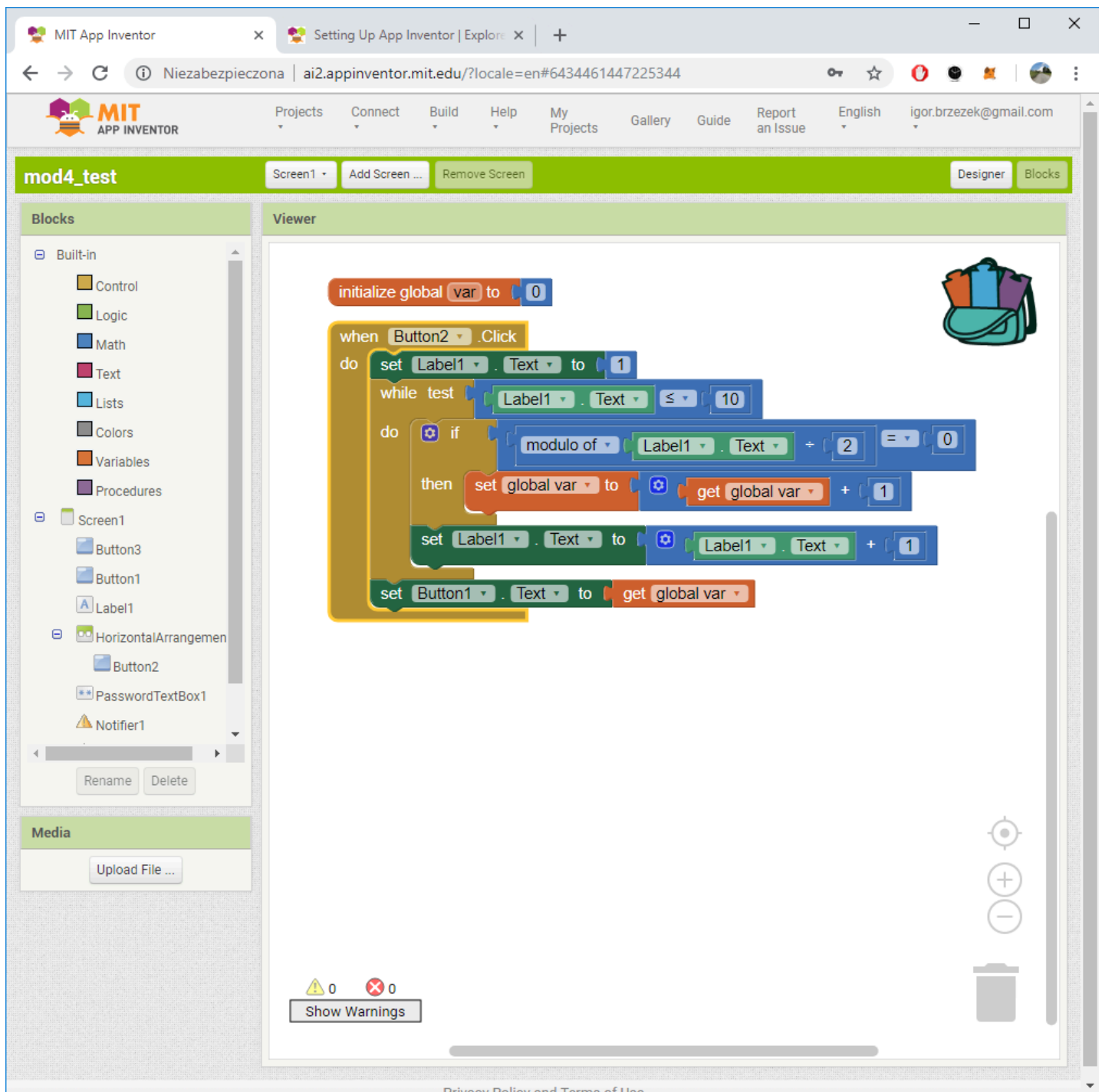
W oknie *Designer* budujemy interfejs użytkownika za pomocą przeciągania myszą komponentów (elementów interfejsu) z lewego panelu *Palette*. Panel podzielony jest na sekcje tematyczne związane z tym czego dotyczą poszczególne komponenty. Komponenty układa się w wymagany widok interfejsu w oknie *Viewer*. Zastosowane komponenty układane są w postaci widoku drzewa w oknie *Components*. Po kliknięciu dowolnego komponentu, w oknie *Properties* pojawiają się wszystkie jego właściwości. Właściwości to cechy charakterystyczne komponentu. Przykładowo przycisk będzie miał opis, rodzaj czcionki, komponent obrazka będzie miał link do pliku obrazu, orientację itd. Poniżej widok okna *Designer*.

Uwaga: jeśli w trakcie pracy na aplikacji dokonamy zmian w kodzie a na ekranie smartfonu/emulatora nic się nie zmienia można zmienić w oknie *Designer* jeden z parametrów komponentu np *Screen* i po chwili budowana aplikacja powinna odświeżyć widok.



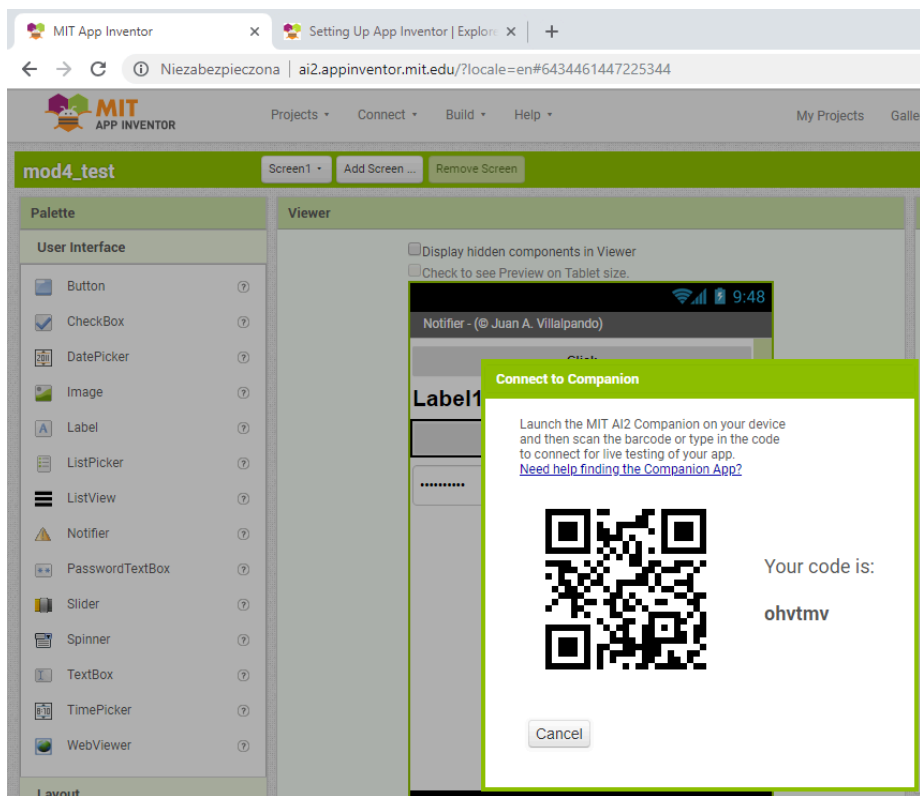
W oknie *Blocks* określamy sposób działania naszego programu. Wykonujemy to za pomocą przeciągania bloków z lewego okna *Blocks* do okna *Viewer*. W oknie *Blocks* mamy dwa rodzaje bloków: built-in wspólne dla wszystkich aplikacji oraz bloki związane z komponentami jakie umieściliśmy w interfejsie użytkownika. Za pomocą takiego zestawu narzędzi jesteśmy w stanie oprogramować działania naszej aplikacji. Bloki pełnią różne funkcje takie jak: obliczenia matematyczne, obsługa tekstów, wyrażenia logiczne, sterowanie kolorystyką, wyrażenia warunkowe, pętle i wiele innych, można by rzec instrukcji. Bloki można przyrównać do gotowych fragmentów, nie raz bardzo rozbudowanych, kodu w danym języku programowania. Inaczej mówiąc, stosując bloki tworzymy program komputerowy (bo smartfon dziś to komputer) nie pisząc ani jednej linii kodu.

Inaczej mówiąc okno *Designer* udostępnia komponenty celem utworzenia interfejsu aplikacji natomiast okno *Blocks* udostępnia swego rodzaju "kod" w postaci wizualnych bloków za pomocą których programujemy działania naszej aplikacji.

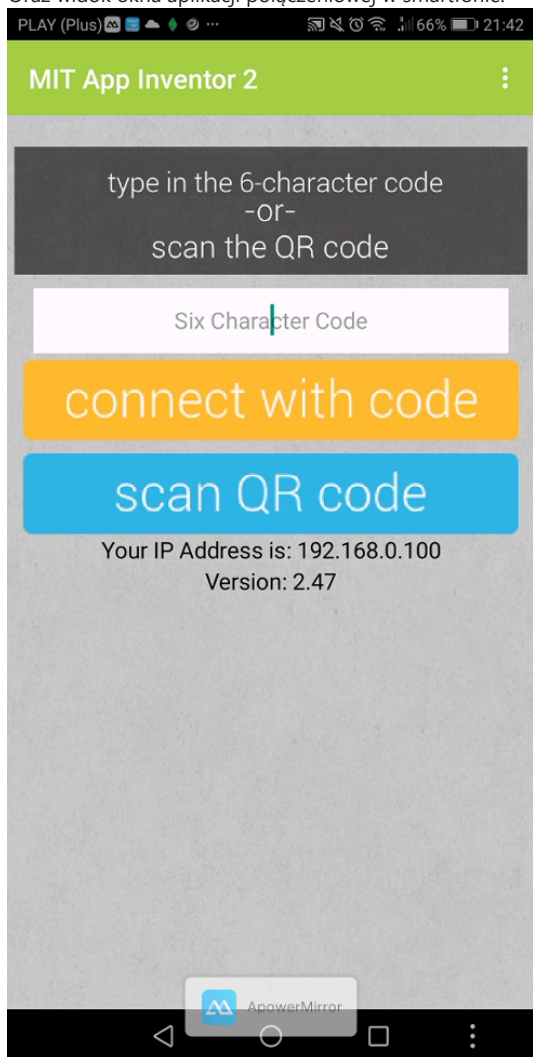


1.4 Podłączanie smartfona/tabletu

W celu automatycznego i w czasie rzeczywistym uruchamiania i testowania projektowanych aplikacji na smartfonie należy podłączyć go do AI2. Należy na nim zainstalować aplikację [MITAI2Companion](#), uruchomić ją i "sparować" ze swoim panelem AI2. Parowanie wykonujemy na dwa sposoby inicjując je z menu panelu AI2 Connect/AI Companion. Pojawi się okno, z którego możemy przepisać kod do aplikacji połączeniowej lub można zeskanować kod QR. Obie możliwości udostępnia zainstalowana na smartfonie aplikacja MIT. Po uzyskaniu połączenia wszystkie zmiany dokonane w projektowanej aplikacji są natychmiast widoczne na smartfonie. Poniżej widok okna połączenia w panelu AI2.

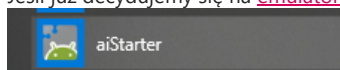


Oraz widok okna aplikacji połączeniowej w smartfonie.

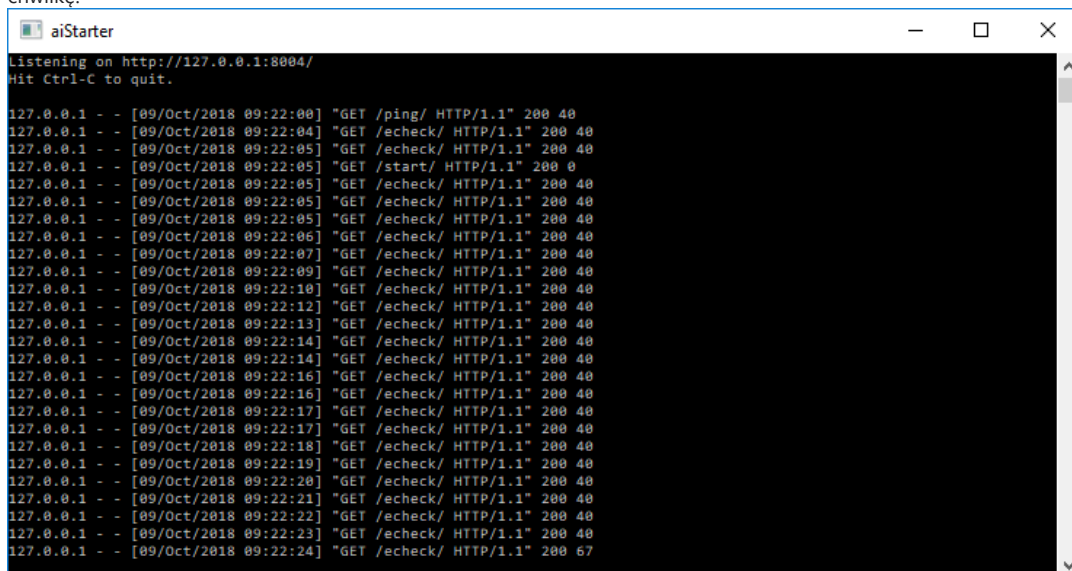


1.5 Testowanie aplikacji za pomocą emulatora programowego

Na wstępie należy nadmienić, że choć emulator działa poprawnie nie wszystko na nim sprawdzimy. Syntezator mowy działa dość kiepsko i tylko po angielsku, nie przetestujemy sensorów takich jak akcelerometr czy GPS. Najlepszy sposób testowania aplikacji to smartfon. Jeśli już decydujemy się na emulator najpierw należy pobrać i zainstalować (program aiStarter) następnie uruchomić.



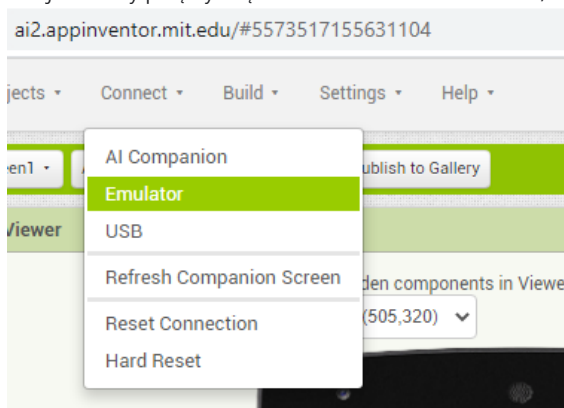
Pojawi się okno konsoli z informacjami o pracy emulatora (jest to tak naprawdę okno serwera świadczącego usługi dla emulatora), między innymi adres i port na jakim nasłuchuje żądań. Okna tego nie należy zamykać a jedynie zminimalizować - jest ono potrzebne podczas pracy emulatora, o którym za chwilę.



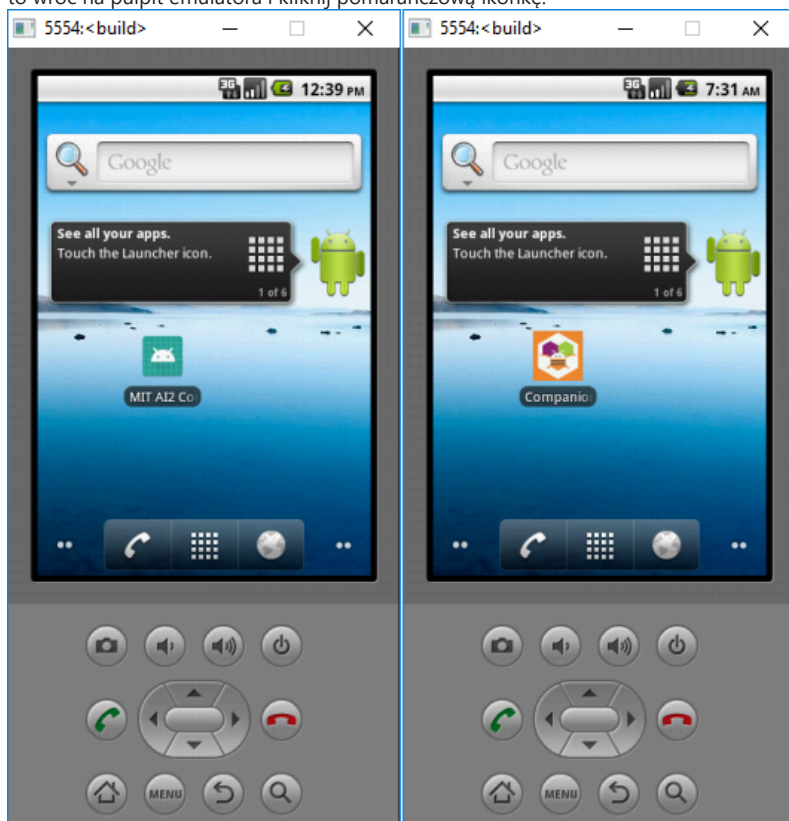
```
aiStarter
Listening on http://127.0.0.1:8004/
Hit Ctrl-C to quit.

127.0.0.1 - - [09/Oct/2018 09:22:00] "GET /ping/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:04] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:05] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:05] "GET /start/ HTTP/1.1" 200 0
127.0.0.1 - - [09/Oct/2018 09:22:05] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:05] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:05] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:06] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:07] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:09] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:10] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:12] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:13] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:14] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:14] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:16] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:16] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:17] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:17] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:18] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:19] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:20] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:21] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:22] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:23] "GET /echeck/ HTTP/1.1" 200 40
127.0.0.1 - - [09/Oct/2018 09:22:24] "GET /echeck/ HTTP/1.1" 200 67
```

Kolejno należy połączyć się z emulatorem z menu Connect/Emulator panelu AI2.



Emulator nie jest za szybki zwłaszcza na wolniejszych komputerach. W przypadku zawieszenia się emulatora wykonać należy akcje z menu panelu AI2: Connect/Reset Connection i połączyć się ponownie. Jeśli to nie działa należy zamknąć okno konsoli emulatora i ponownie uruchomić go poprzez program aiStarter. Jeśli aplikacja wewnątrz emulatora nie uruchamia się automatycznie należy dwa razy kliknąć pomarańczową zieloną czy inną (zależnie od wersji softu) ikonkę Companion widoczną na poniższym obrazie. Podobnie, jeśli przełączysz się w emulatorze na pulpit czy inny program (nie ma tam ich wiele) to wróć na pulpit emulatora i kliknij pomarańczową ikonkę.



Emulator pracuje także na systemach Linux i MAC. Jeśli pracujesz na systemie Linux zajrzyj [tutaj](#) a jeśli na MAC [tutaj](#).

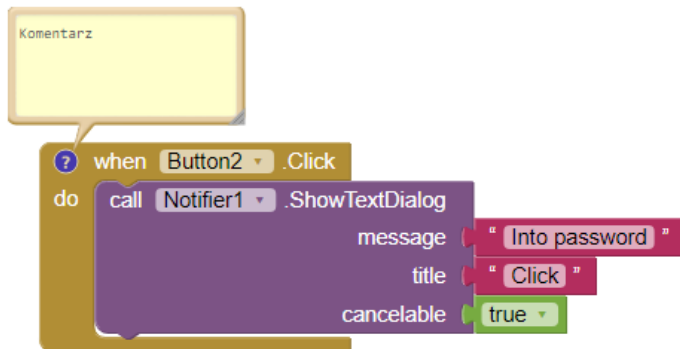
2 Obsługa bloków

W systemie AI2 bloki spełniają funkcję kodu. Zamiast pisać ogromną liczbę linii kodu w danym języku programowania mamy dostępnych wiele bloków wykonujących różne operacje takie jak instrukcje warunkowe, wyboru, pętle, obsługę komponentów i wiele innych.

Bloki, w zależności od komponentu którego dotyczą, mają różne właściwości i możliwości ułatwiające tworzenie aplikacji. Ważniejsze z nich wraz z właściwościami opisano poniżej.

2.1 Komentarze

Wyjątkowo użyteczna funkcja, posiada ją chyba każdy język programowania. W AI2 klikamy prawym klawiszem myszki na bloku i z menu kontekstowego wybieramy *Add Comment*, w lewym górnym rogu bloku pojawi się niebieska ikona ze znakiem zapytania, klikając w ten znak otwiera się małe okienko gdzie możemy wpisać komentarz. Komentarze są dla programistów, nie mają żadnego wpływu na działanie programu.



2.2 Kopiowanie bloków

Czasem utworzony zostanie skomplikowany układ bloków i zaistnieje potrzeba powielenia go np w celu testu. Prawy klikamy na bloku i *Duplicate*. To samo zrobi Ctrl-C+Ctrl-V.

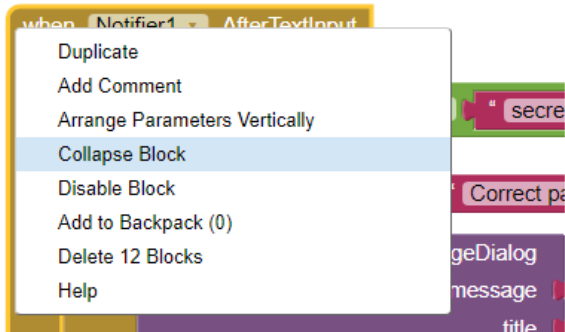
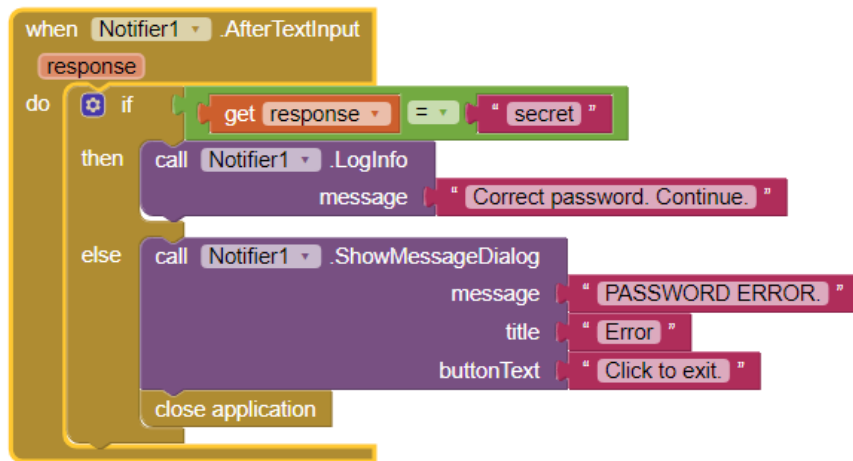
2.3 Wyłączanie bloków

Jeśli dany blok chcemy wyłączyć tak aby w trakcie programu nie działał ale nie chcemy go usuwać z ekranu mamy funkcje *Disable block/Enable block*. Blok wyłączony jest na ekranie ale NIE działa tak jak by go nie było. System informuje nas o tym szarym kolorem bloku.



2.4 Zwijanie bloków

Jeśli blok zajmuje dużo miejsca na ekranie projektowania możemy go "zwinąć" za pomocą pozycji menu *Collapse block/Expan block*, będzie nadal działał bez zmian ale jego widok będzie stanowiła jedna linia. Można także zwijać fragmenty większych bloków.



when Notifier1 .AfterTextl...

2.5. Zachowanie bloku do ponownego wykorzystania

Jeśli jakiś blok będziemy używać w przyszłości (tyczy się to raczej zbudowanych i skomplikowanych bloków) to możemy go zachować w "plecaku". W tym celu prawym na bloku i "Add to backpack". Blok pojawi się w plecaku.