POLITECHNIKA GDAŃSKA

Wydział Elektrotechniki i Automatyki

Kierunek: Automatyka i robotyka

Rok akademicki: 2016/2017

Studia stacjonarne, stopień: II, semestr: II

Data zdania projektu: 23.01.2017

**Projekt grupowy**

**Temat: Aplikacja mobilna wspomagająca wybór wydarzenia tanecznego**

Prowadzący:

dr inż. Daniel Wachowiak

Wykonali:

Jakub Adamczyk

Piotr Bzowski

Szymon Hełmecki

**GDAŃSK 2017**

Spis treści

[1. Założenia projektu 4](#_Toc472844325)

[2. Layout aplikacji 4](#_Toc472844326)

[2.1. Okno główne 4](#_Toc472844327)

[2.2. Opis wydarzenia 5](#_Toc472844328)

[3. Kod aplikacji 6](#_Toc472844329)

[3.1. Plik AndroidManifest.xml 6](#_Toc472844330)

[3.2. Klasa MainActivity 6](#_Toc472844331)

[3.3. Klasa Response 10](#_Toc472844332)

[3.4. Klasa LoadJSONTask 10](#_Toc472844333)

[3.5. Klasa EventDescription 12](#_Toc472844334)

[Klasa DescriptionActivity 13](#_Toc472844335)

[4. Struktura bazy danych 16](#_Toc472844336)

[5. Skrypt PHP obsługujący bazę danych 18](#_Toc472844337)

# Założenia projektu

Tematem projektu było utworzenie aplikacji mobilnej wspomagającej użytkownika w znalezieniu interesującego go wydarzenia tanecznego odbywającego się w jego pobliżu. Temat został uznany przez studentów za interesujący ponieważ na rynku trójmiejskim brakuje medium informacyjnego dostępnego dla każdej zainteresowanej osoby. Takie medium powinno spełniać kilka wymogów, między innymi:

* Prostota obsługi
* Prezentacja kompletnych danych na temat wydarzenia
* Element wspomagania decycji, w tym przypadku możliwość wyszukiwania wydarzeń ze względu na lokalizację użytkownika
* Niezależność od mediów społecznościowych, w których często pojawiają się tego typu ogłoszenia. Nie każdy zainteresowany posiada Facebook’a.

Mechanizm działania aplikacji opiera się na pobieraniu informacji o lokalizacji użytkownika i o promieniu, w którym chce on wyszukiwać wydarzenia. Lokalizacja pobierana jest za pomocą usługi Google Play Services, która udostępnia wiele najczęściej wykorzystywanych w aplikacjach mobilnych funkcji. Następnie dane te wysyłane są metodą GET na serwer, gdzie są przetwarzane i do użytkownika wysyłana jest tablica JSON z informacjami na temat wydarzeń znajdujących się w promieniu. Dalej, wydarzenia są wyświetlane w liście (Rysunek 2.1.). Po kliknięciu danego eventu otwiera się okno z jego szczegółowym opisem. Jeśli jakieś wydarzenie zainteresuje użytkownika może on zapisać je w kalendarzu swojego smartfona, co ustawi przypomnienie o nadchodzącym evencie.

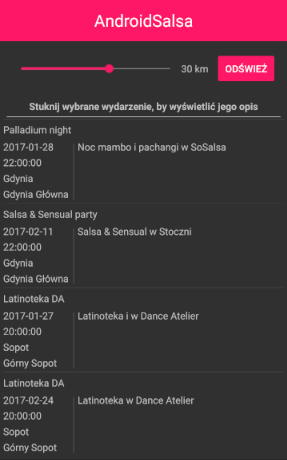
Postanowiono, że dla celów dydaktycznych wczesna wersja aplikacji powstanie tylko dla smartfonów z systemem operacyjnym Android. Aplikacja została napisana w języku Java przy wykorzystaniu środowiska Android Studio.

Aplikacja jest zgodna z systemami Android od 2.3 i wyżej. Do działania wymaga aktualizacji aplikacji Google Play Services i wyrażenia zgody na wykorzystanie lokalizacji użytkownika.

# Layout aplikacji

### Okno główne

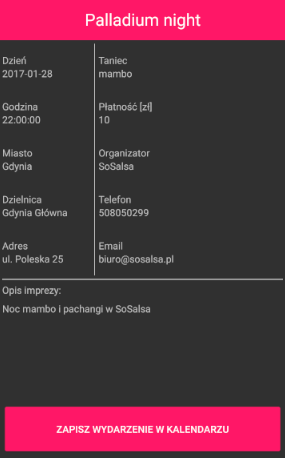
W oknie głównym (Rysunek 2.1.) znajduje się suwak do ustawiania promienia poszukiwań, przycisk „Odśwież” i lista znalezionych wydarzeń.



Rysunek 2. . Okno główne aplikacji

### Opis wydarzenia

W tym oknie (Rysunek 2.2.) wyświetlane są szczegółowe informacje na temat wybranego wydarzenia. Z poziomu tego okna można zapisać wydarzenie w kalendarzu telefonu za pomocą przycisku „Zapisz wydarzenie w kalendarzu”.



Rysunek 2. . Okno z opisem wybranego wydarzenia

# Kod aplikacji

### Plik AndroidManifest.xml

W manifestach przechowywane są ogólne informacje na temat aplikacji (np. jej nazwa) i uprawnienia. W wypadku tego projektu aplikacji nadano prawa do łączenia się z internetem i dostępu do lokalizacji użytkownika.

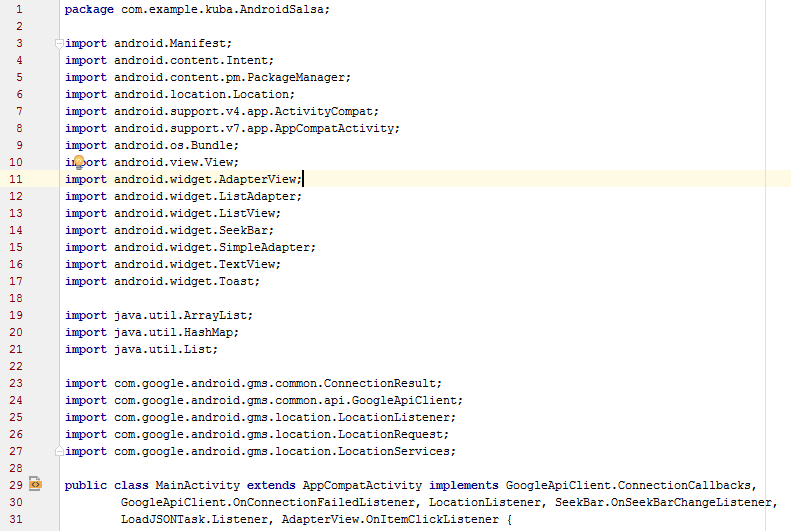
Listing 3. . Kod pliku AndroidManifest.xml

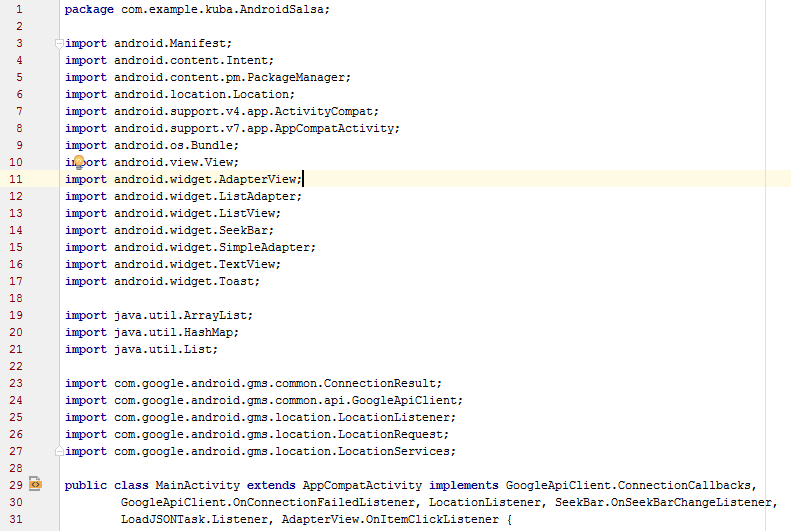
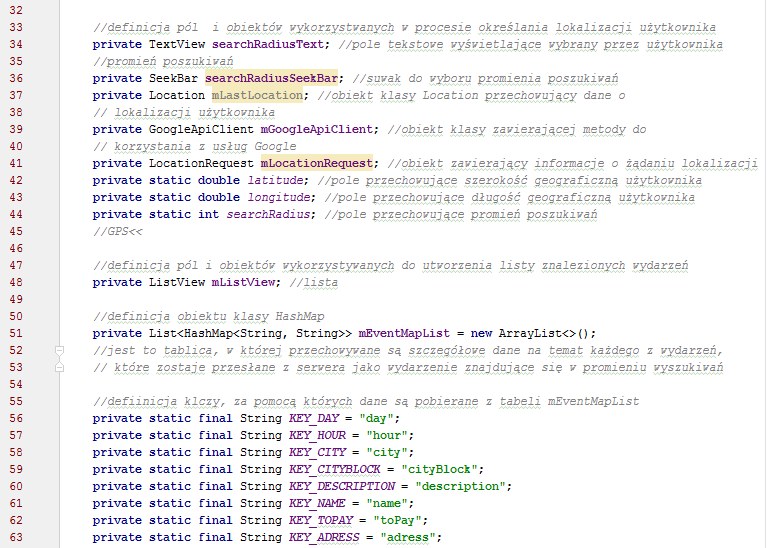
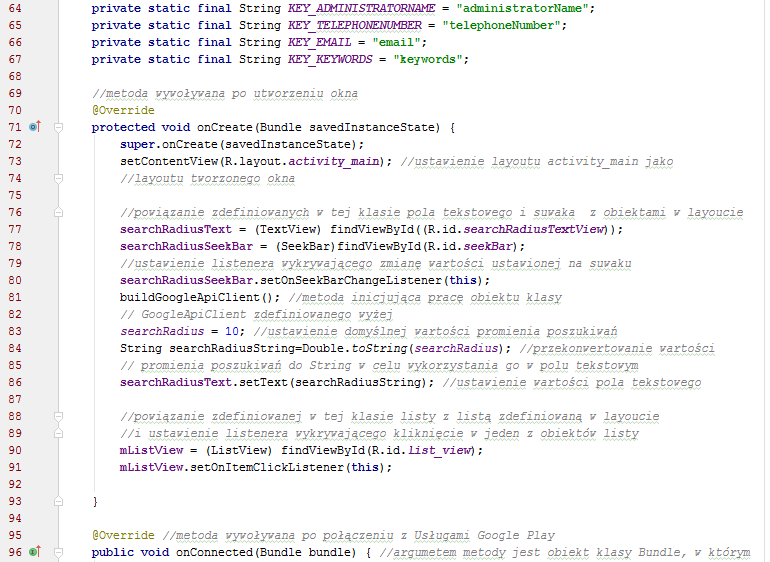
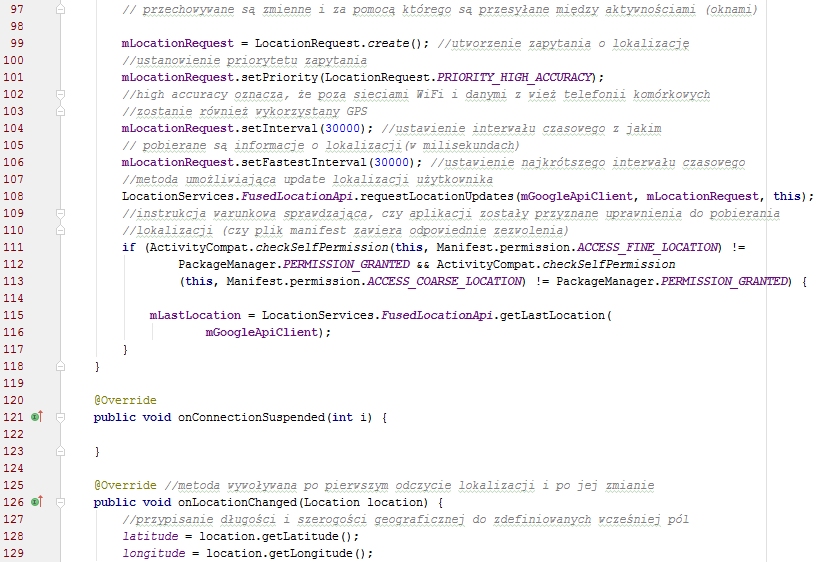
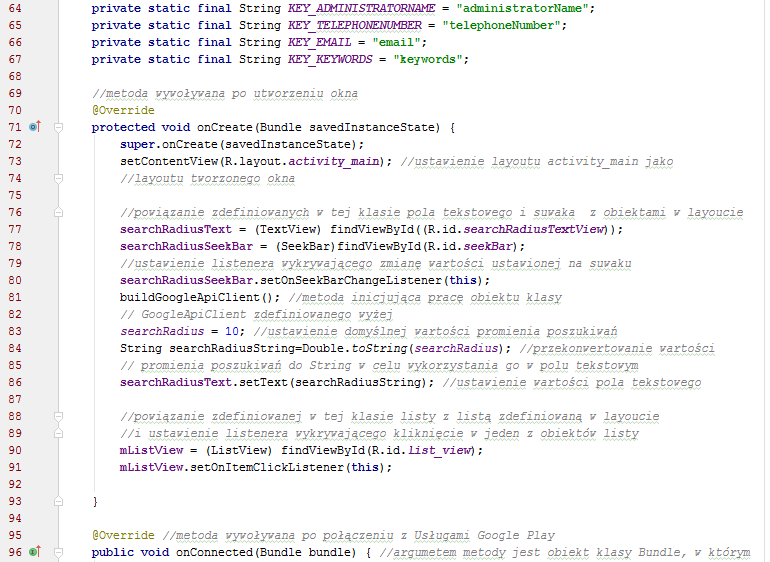
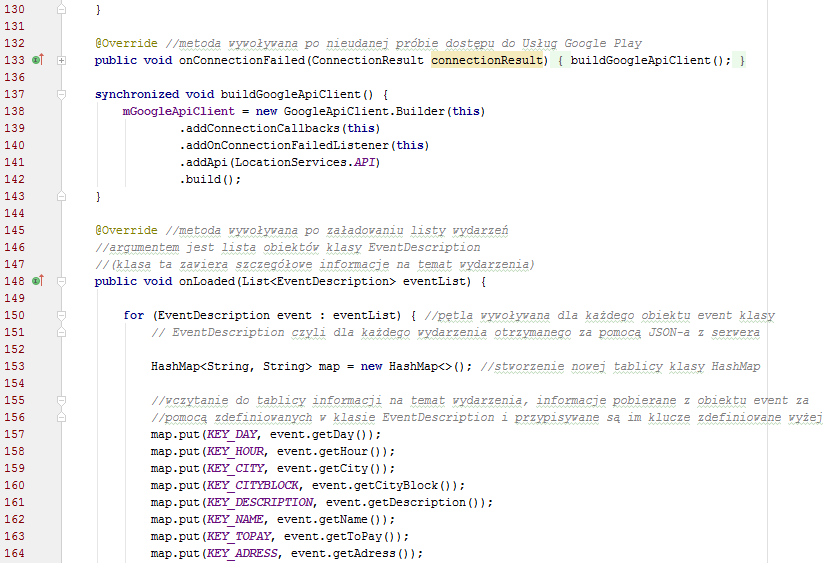
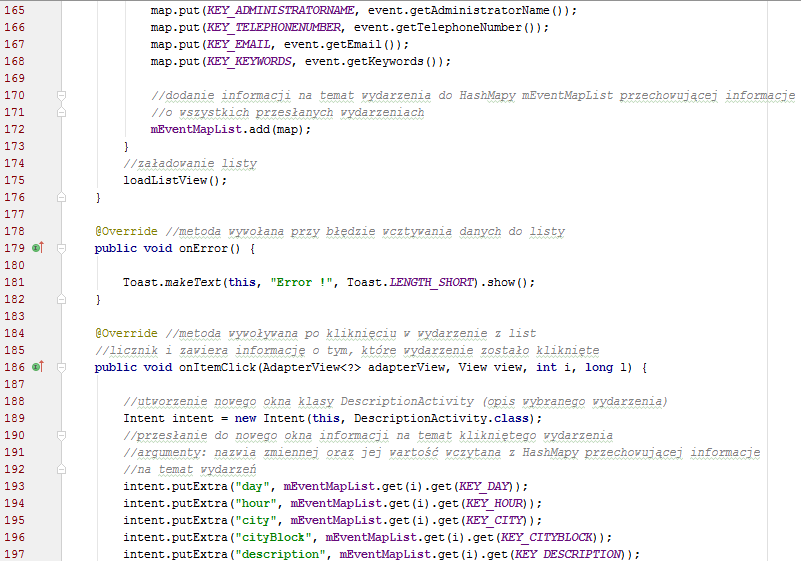
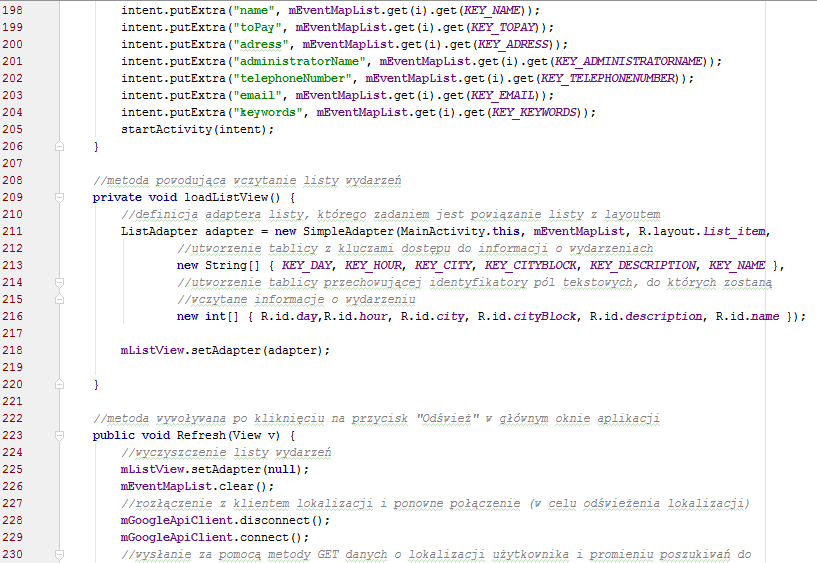
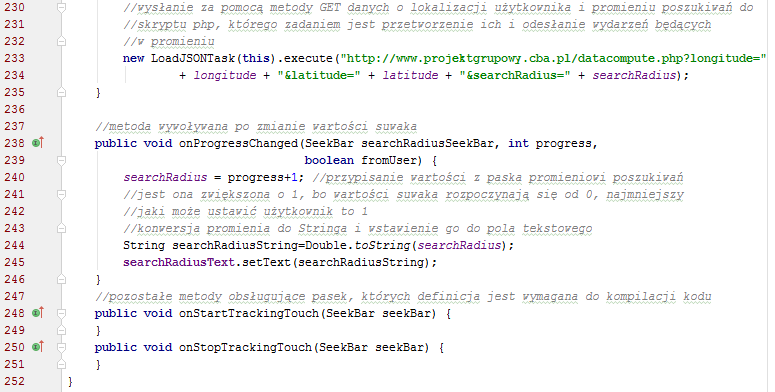


### Klasa MainActivity

W klasie tej zdefiniowane są pola i metody wykorzystywane do ustalenia lokalizacji użytkownika i wypełnienia listy (obiekt mListView klasy ListView) wyświetlającej znalezione wydarzenia.

Listing 3. . Kod klasy MainActivity

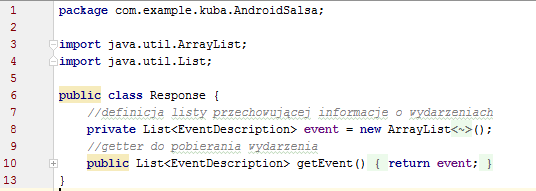


### Klasa Response

Ta klasa zawiera definicję listy wydarzeń (lista przechowuje obiekty klasy EventDescription, z których każdy zawiera komplet informacji na temat wydarzenia, patrz opis klasy EventDescription) i getter do pobierania wybranego wydarzenia. Obiekt tej klasy wykorzystywany jest przy odczytywaniu danych na temat wydarzenia otrzymanych w tablicy JSON (patrz opis klasy LoadJSONTask).

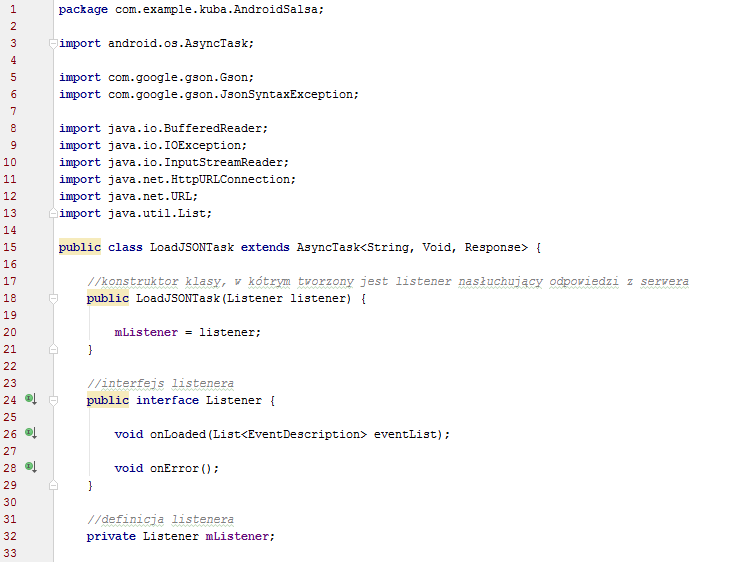
Listing 3. . Kod klasy Response

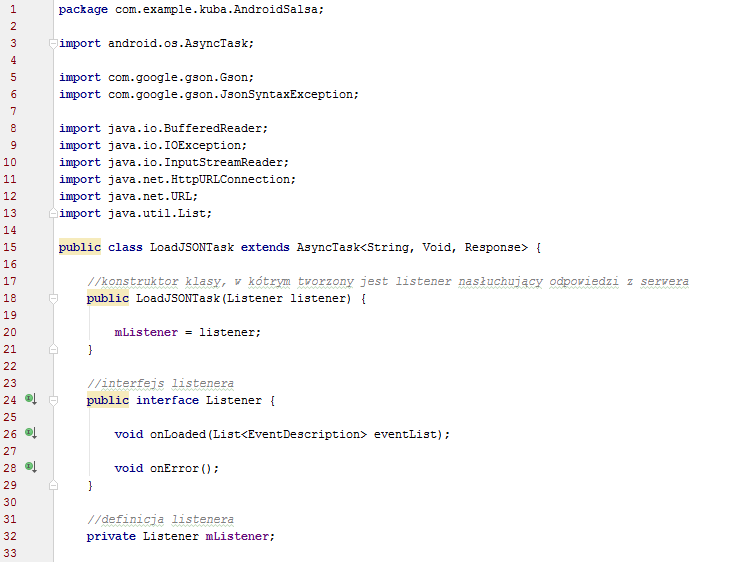
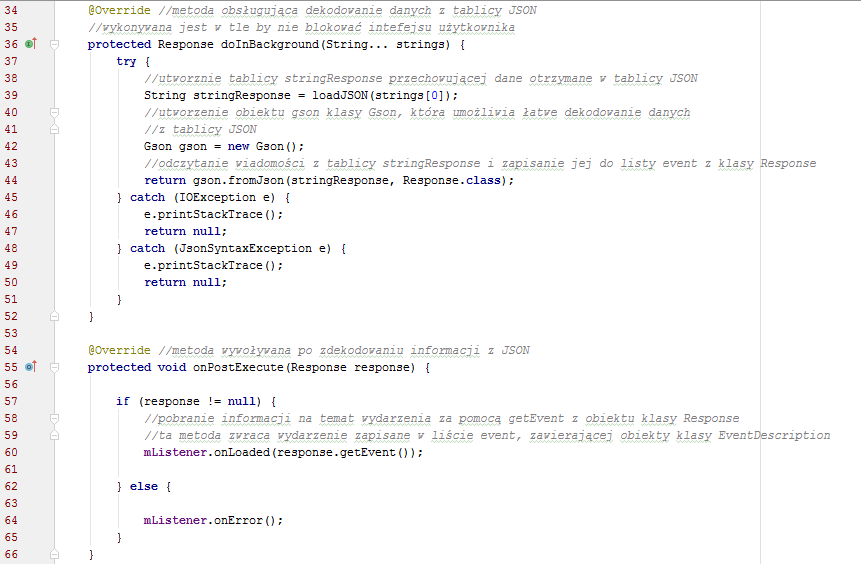
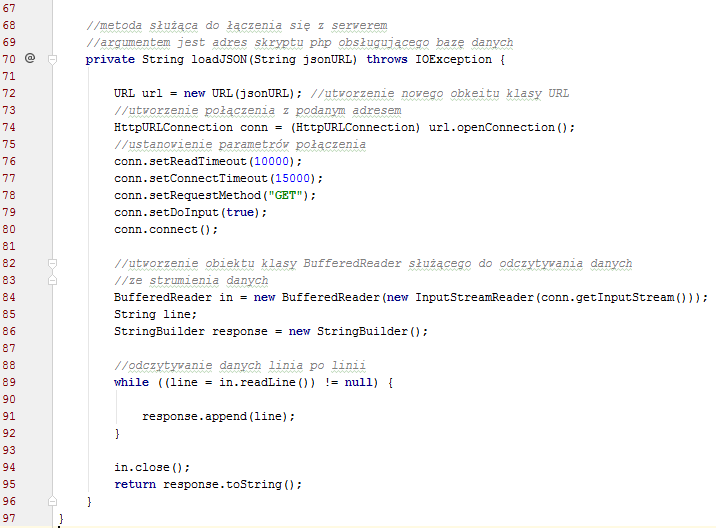


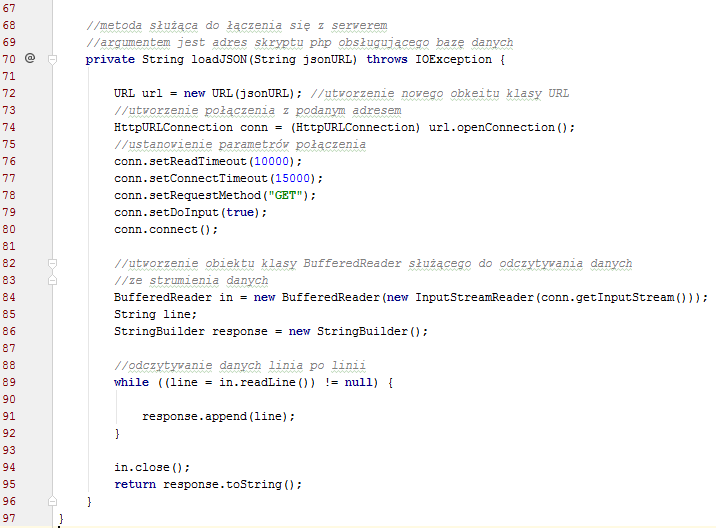
### Klasa LoadJSONTask

W tej klasie znajdują się metody wykorzystywane do dekodowania danych z tablicy JSON przesyłanej z serwera.

Listing 3. . Kod klasy LoadJSONTask



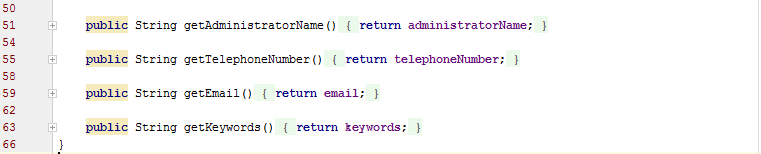
  



### Klasa EventDescription

W polach obiektów tej klasy znajdują się wszystkie informacje na temat wydarzeń. Dodatkowo klasa zawiera gettery pozwalające wykorzystać te informacje w innych obiektach.

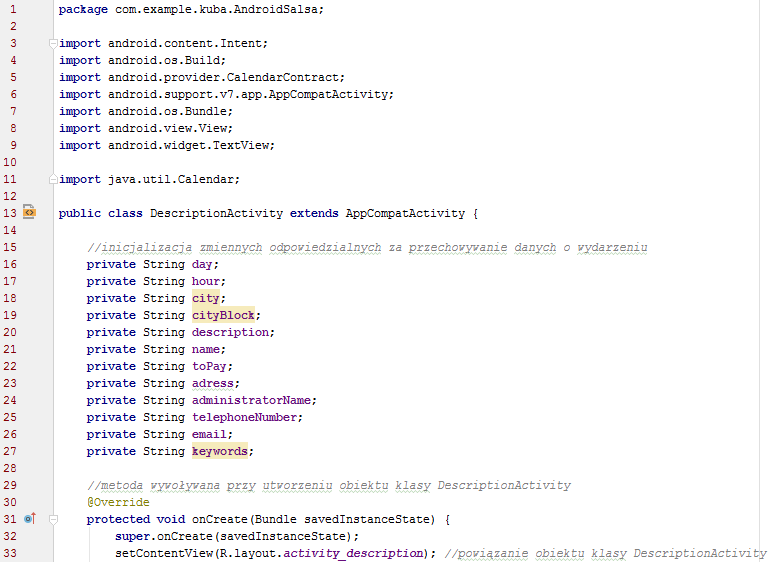
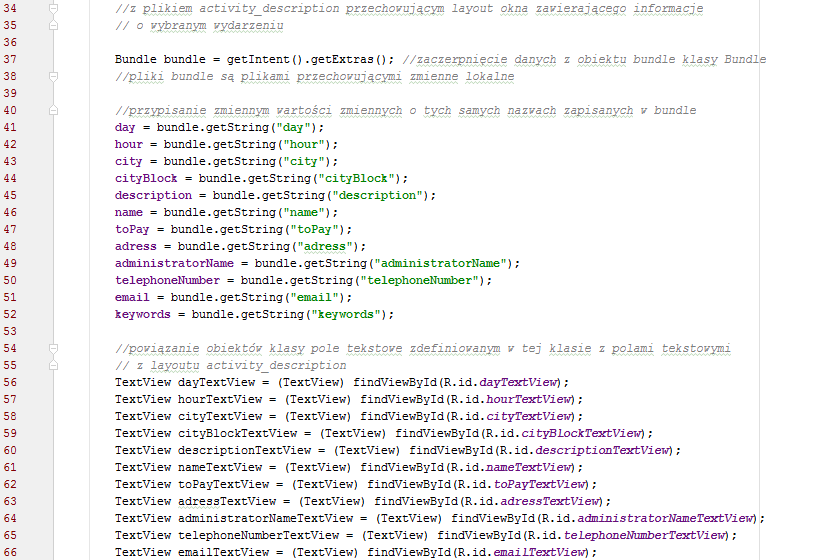
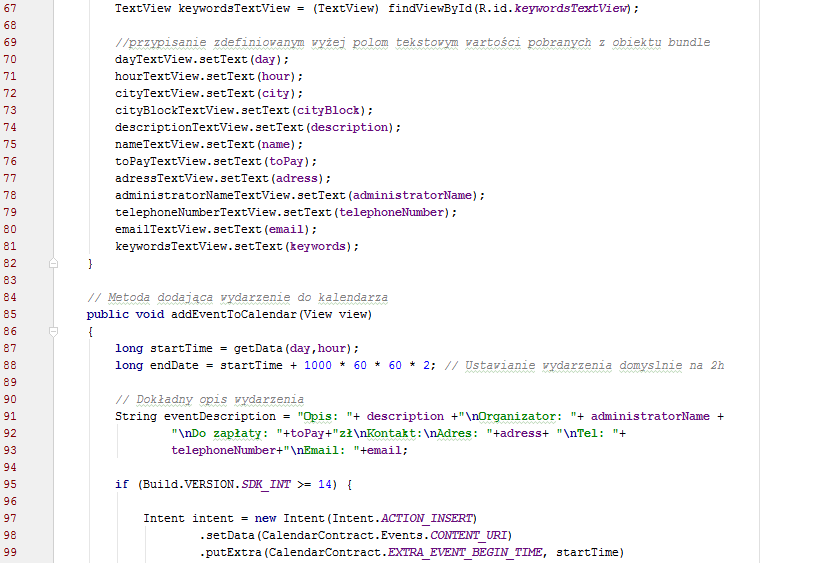
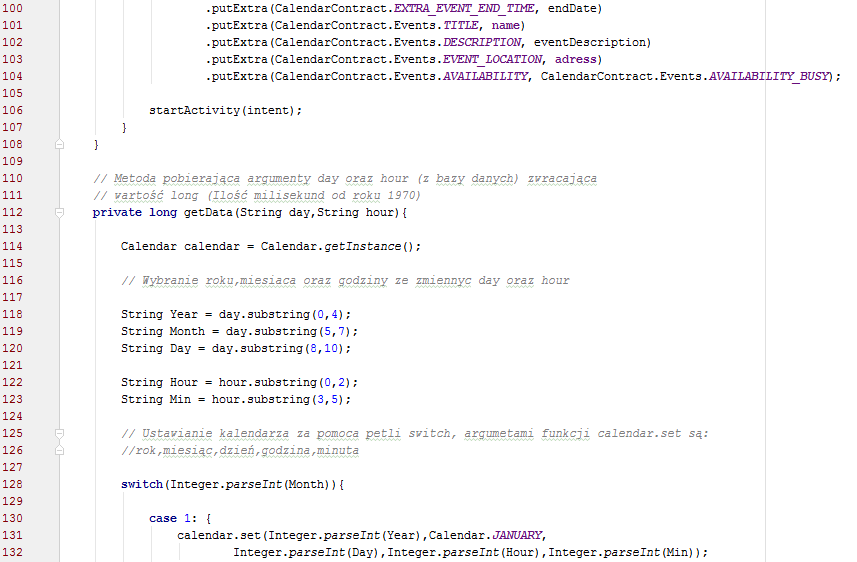
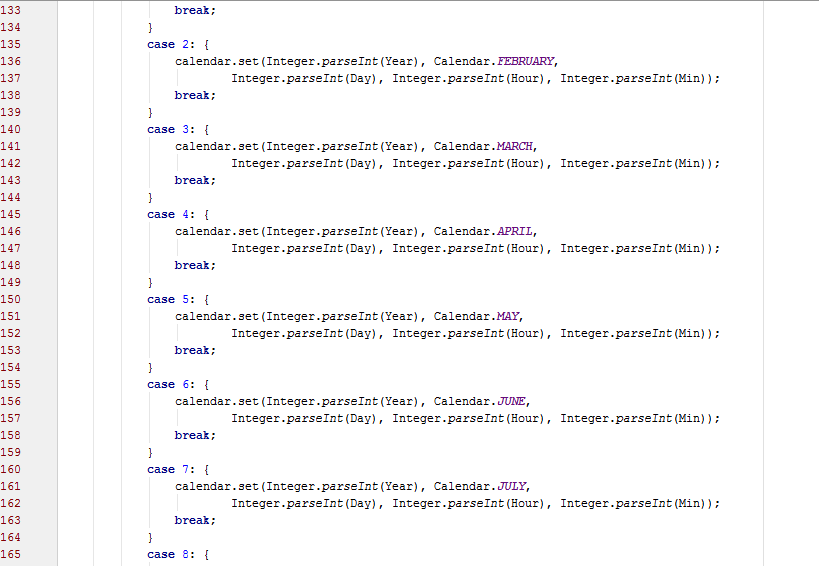
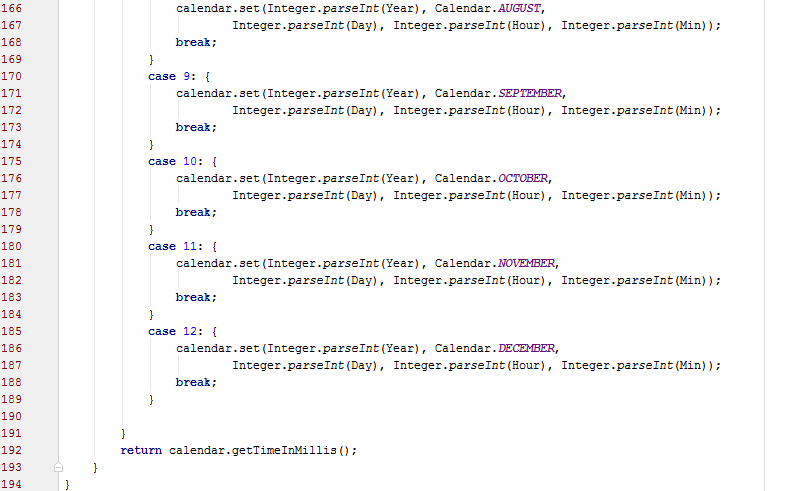
Listing 3. . Kod klasy EventDescription

### Klasa DescriptionActivity

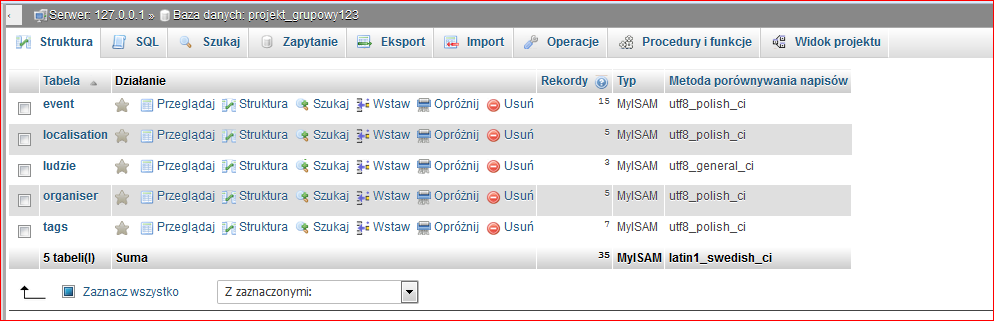
Obiekt tej klasy wykorzystywany jest przy otwieraniu okna z opisem wybranego wydarzenia. Dochodzi tu do zapełnienia pól tekstowych prezentujących informacje na temat eventu. W tej klasie jest również zdefiniowana metoda obsługująca zapisywanie wydarzenia do kalendarza.

Listing 3. . Kod klasy DescriptionActivity

# Struktura bazy danych

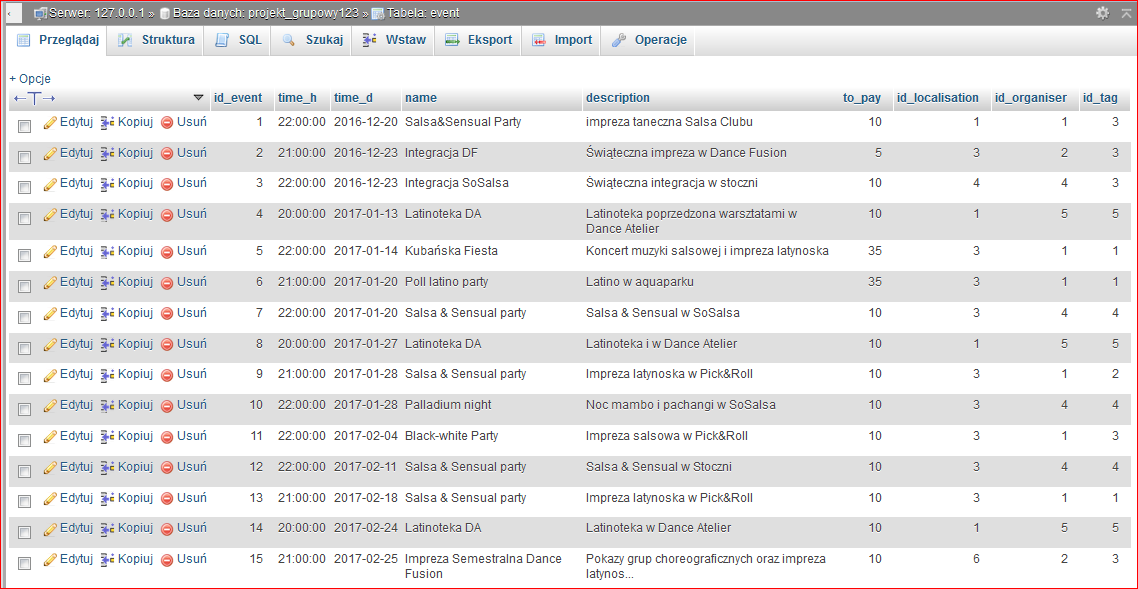
Dane znajdujące się na serwerze zostały podzielone na 5 tabel (Rysunek 4.1.). Zawartość poszczególnych tabel oraz ich powiązanie zostało opisane poniżej.



Rysunek 4. . Tabele bazy danych

Najważniejszą z tabel jest tabela EVENT (Rysunek 4.2.) zawierająca w sobie wydarzenia, których wyszukujemy w ramach działania aplikacji. Składa się z klucza głównego, kolumny z godziną rozpoczęcia wydarzenia (typ TIME) oraz kolumny z dniem rozpoczęcia wydarzenia (typ DATE).

Dodatkowo poszczególne rekordy posiadają nazwę oraz krótki opis (zmienne Char oraz text). Zmienną liczbową jest to\_pay która przedstawia cenę wstępu   
na wydarzenie. Można powiedzieć że jest to najważniejsza z tabel, gdyż zawiera ona klucze obce wiążące ją z innymi tabelami.



Rysunek 4. . Tabela event

Kolejną z tabel jest tabela localization (Rysunek 4.3.). Zawiera ona w sobie informacje dotyczące miejscowości, dzielnicy oraz dokładnego adresu organizowanego wydarzenia. Jej klucz główny jest kluczem obcym dla tabeli events. Dodatkowo tabela posiada współrzędne geograficzne danej lokalizacji dzięki czemu możliwe jest filtrowanie rekordów poprzez odległość od lokalizacji osoby wyszukującej wydarzenia (zostanie to opisane w kolejnym rozdziale).



Rysunek 4. . Tabela localization

Przedostatnią wykorzystywaną tabelą jest tabela organizer (Rysunek 4.4.) Zawiera ona w sobie kolumny ze zmiennymi tekstowymi zawierające dane organizatorów (nazwa, telefon oraz e-mail).



Rysunek 4. . Tabela organizer

Ostatnią z wykorzystywanych tabel jest tabela tags (Rysunek 4.5.) zawierająca w sobie słowa klucze, które mogą zostać wykorzystywane wraz z rozwojem aplikacji w celu filtracji wydarzeń lub personalizacji oczekiwań użytkowników. W chwili obecnej tagi pozwalają przedstawić podstawowe informacje dotyczące tematyki wydarzenia.



Rysunek 4. . Tabela tags

Dodatkowo wraz z rozwojem aplikacji wykorzystywana będzie tabela zawierająca dane użytkowników. W chwili obecnej nie jest ona jeszcze rozwinięta jednakże planowane jest przechowywanie posolonych haseł profili organizatorów (tworzących wydarzenia).

# Skrypt PHP obsługujący bazę danych

Na serwerze hostingowym został umieszczony skrypt (Rysunek 5.1.) którego zadaniem jest pobranie danych od użytkownika, dokonanie obliczeń oraz przesłanie odpowiednich rekordów z bazy danych.

Serwer najpierw pobiera dane z telefonu dotyczące położenia użytkownika (długość i szerokość geograficzna) oraz ustawionego w aplikacji promienia wyszukiwania wydarzeń. Operacja ta wykonywana jest w 3, 4 i 5 linijce skryptu za pomocą komendy $\_GET. Pobranie tych danych pozwala na dokonanie pomiaru odległości pomiędzy użytkownikiem, a lokalizacją poszczególnych wydarzeń znajdujących się w bazie danych. W linijkach 19-21 skryptu zostało przedstawione zapytanie skierowane do bazy danych w celu pobierania rekordów z wydarzeniami. Zapytanie   
to posiada 2 warunki. Pierwszym z nich jest warunek aby odległość obliczona była mniejsza od pobranego od użytkownika promienia poszukiwań. Drugim warunkiem jest wymaganie, aby użytkownik otrzymywał jedynie wydarzenia, które dopiero się wydarzą (w celu uniknięcia przesyłania nieaktualnych rekordów).

Dane pobrane za pomocą tego zapytania wysyłane są do użytkownika w tablicy zapisanej w formacie JSON.

Listing 5. . Kod pliku computedata.php

