Programowanie komponentowe

Projekt semestralny

**NAWIGATOR   
BUDYNEK 34 SGGW**

Autorzy:

Natalia Walasik

Kamil Trybek

Paweł Wasil

Opis projektu

Aplikacja ułatwiająca studentom poruszanie się po budynku 34.

Użytkownik po uruchomieniu aplikacji ma do wyboru dwie opcje: podgląd mapy każdego z pięter lub wyznaczenie trasy z punktu do punktu. Wybierając pierwszą opcję, jest w stanie ocenić gdzie znajduje się jego cel i sam do niego trafić. Jeśli jednak napotka na trudności, określa swoje położenie- salę w pobliżu której się znajduje oraz salę którą chce odszukać. Program rysuje trasę   
i w przejrzysty sposób wyświetla ją użytkownikowi.

Aplikacja oferuję również szereg innych możliwości. Użytkownik posiada możliwość dodania spersonalizowanych notatek, przypisanych do danej sali np. godzinę o której ma zajęcia lub kolokwium. Następnie decyduje czy chce włączyć powiadomienia – aplikacja przypomni użytkownikowi, że zaraz zaczną się jego zajęcia.

Wygodny ekran ustawień pozwala użytkownikowi dostosować wygląd aplikacji do swoich potrzeb.

Aplikacja jest dostępna na platformy Windows (8.1 i nowszy), Windows Phone (8.1 i nowszy) oraz Android (4.4 i nowszy).

Link do Github’a

https://github.com/Trybek/Nawigator\_SGGW\_B34

IDL

module Drawing

{

public interface IDrawer

{

oneway void DrawPathFromRoomToCorridor(in Room which, in bool finish = false, in bool start = true);

oneway void DrawPathFromRoomToStairsAndReverse();

oneway void DrawArrow(in int x, in int y, in EnumPosition position, in bool start = true);

oneway void DrawPathBetweenFloors();

oneway void DrawPathStartHorizontal(in Room roomStart, in Room roomFinish, in bool start = true);

oneway void DrawPathStartVertical(in Room roomStart, in Room roomFinish, in bool start = true);

object[] DrawPath(in Room start, in Room finish);

};

struct Room {

attribute int ID;

attribute string Floor;

attribute string Name;

attribute double X;

attribute double Y;

};

enum EnumPosition {Up, Right, Down, Left};

}

module DataBase

{

public interface ISQLite

{

attribute string DBPath;

sequence <Room> ReadRooms();

sequence <Room> ReadRoomsOnFloor(in int floor);

sequence<Note> ReadNotes();

sequence <int> ReadFloors();

Room GetStairsByName(in string name);

Room FindRoomByID(in int ID);

Note FindNoteByID(in int ID);

string GetNameRoomByID(in string ID);

bool CheckTableExists();

oneway void CopyDatabase();

oneway void InsertNote(in Note note);

oneway void UpdateNote(in Note note);

oneway void DeleteNote(in int id);

};

}

module Notifications

{

public interface INotifications

{

attribute ISQLite DatabaseHelper;

attribute bool IsNotificationAllowed;

attribute bool RemoveOldNotes;

oneway void AddNotification(in Note note);

oneway void RemoveExpiredNotes(in sequence <Note> listOfNotes);

sequence <Note> GetListOfNotes();

};

struct Note {

attribute int ID;

attribute string TimeOfNote;

attribute int RoomID;

attribute string RoomName;

attribute string TextOfNote;

};

}

Diagramy

Diagram klas:

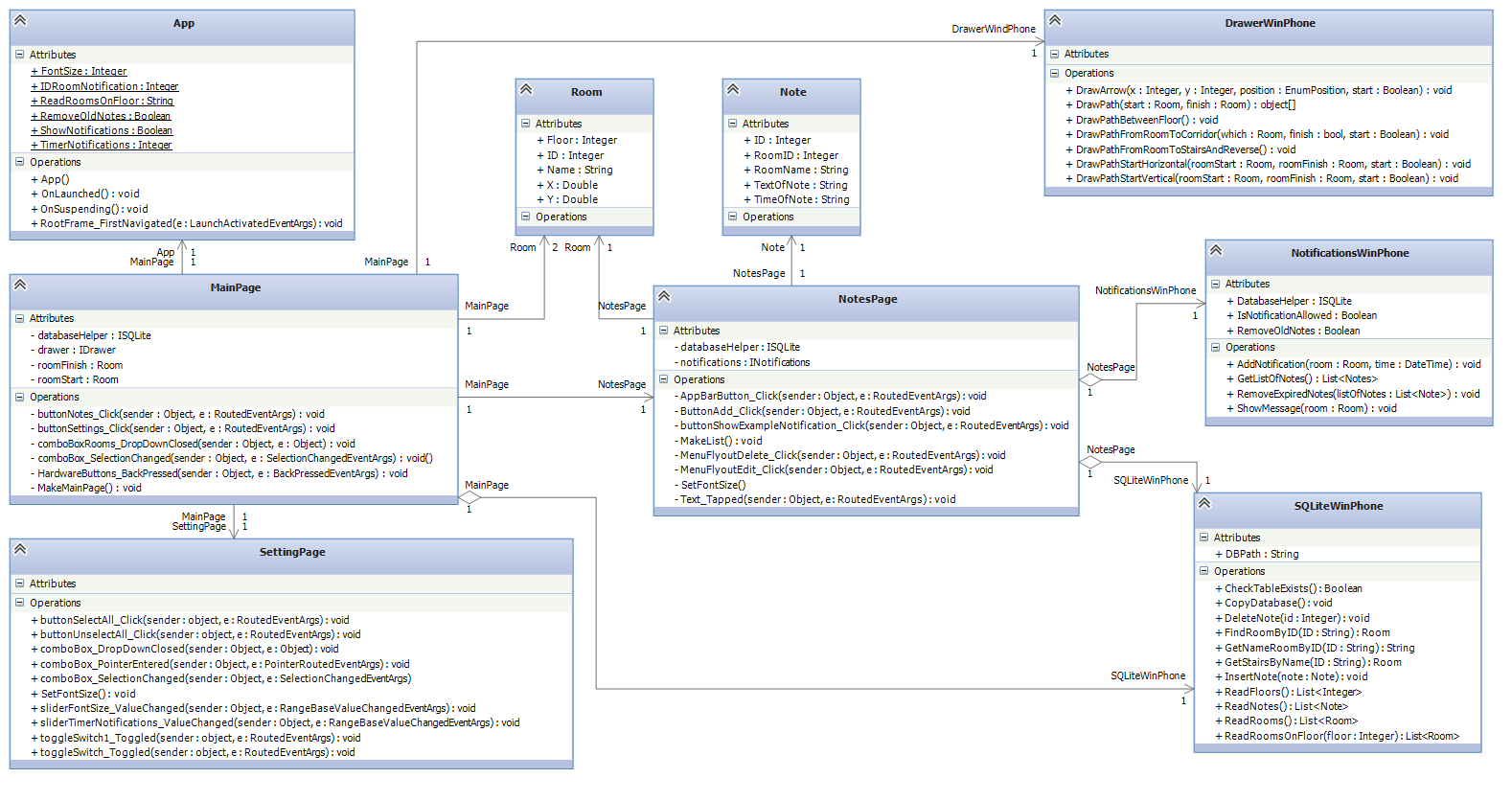


Diagram przypadków użycia (use case):

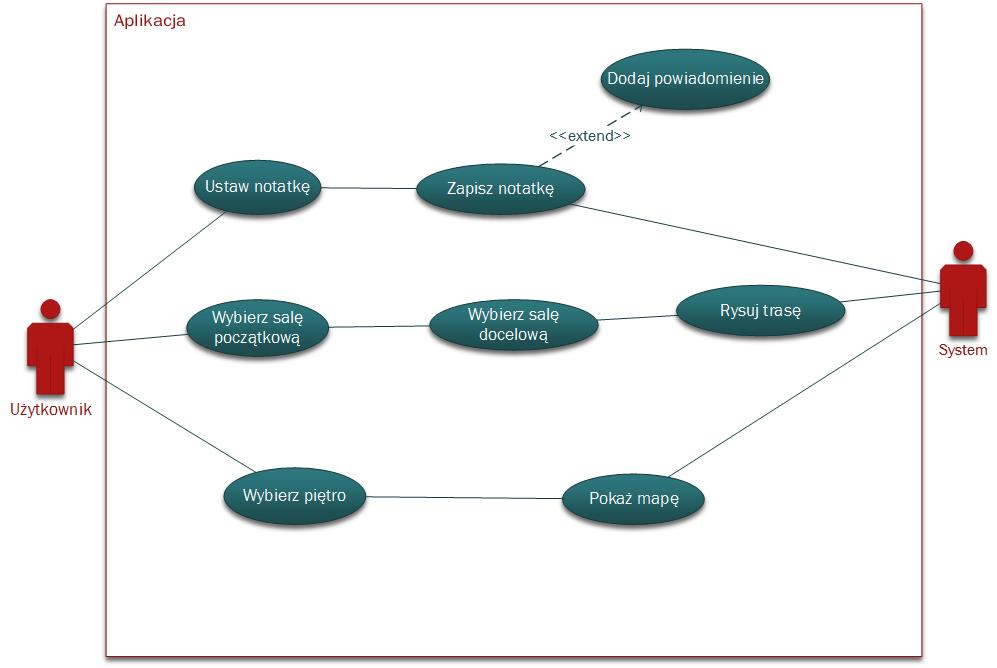
****

Diagram aktywności (activity):

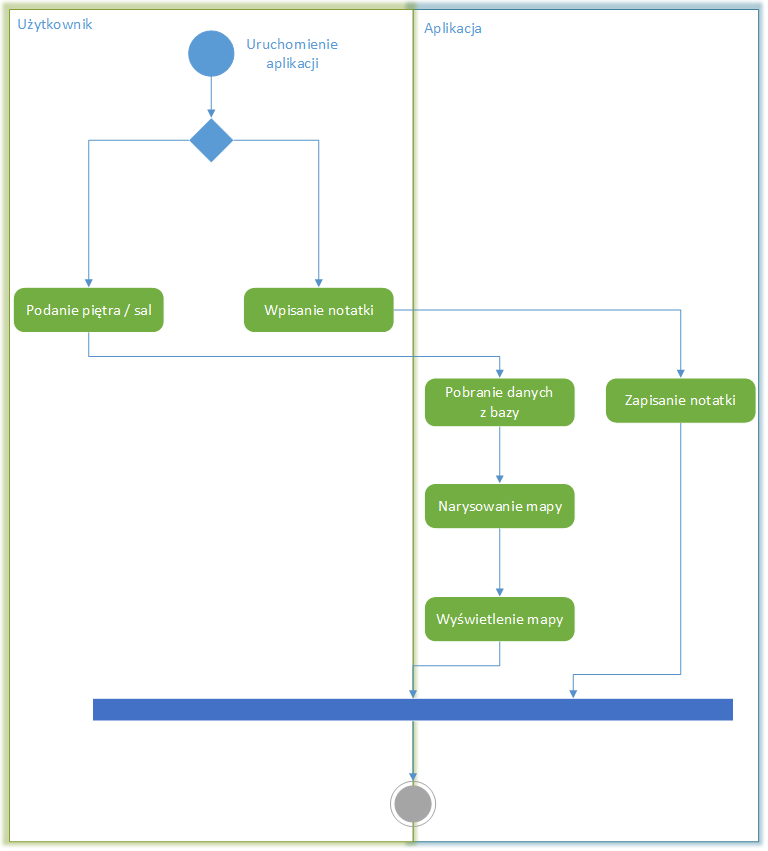


Diagram komponentów (component):

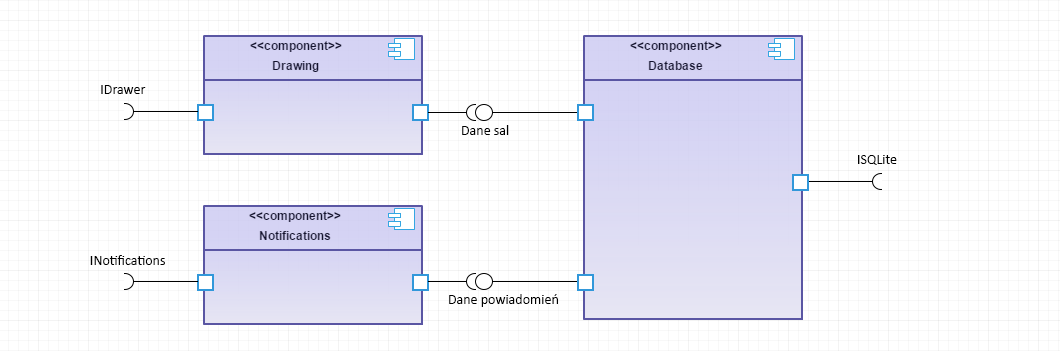


Diagram sekwencji

