# 

Implementacja standardu autoryzacyjnego

OAuth 2.0

Opracował:

Inż. Krzysztof Rybka

I rok mgr

Spis treści

[Wstęp 3](#_Toc497571829)

[Aktorzy w komunikacji 3](#_Toc497571830)

[Rejestracja aplikacji 3](#_Toc497571831)

[Rodzaje przepływów 4](#_Toc497571832)

[Implementacja protokołu OAuth 2.0 4](#_Toc497571833)

[Test aplikacji wykorzystującej implementację protokołu 4](#_Toc497571834)

[Podsumowanie 4](#_Toc497571835)

[Bibliografia 5](#_Toc497571836)

(informacje)

# Wstęp

OAuth 2.0

Standardowa komunikacja w aplikacji (bez OAuth)



Usługa zarządzająca danymi

Dane

Użytkownik

Komunikacja w aplikacji używającej OAuth :

Aplikacja kliencka to aplikacja która komunikuje się z zewnętrzną usługa która serwuje jej zasoby.



Aplikacja kliencka

Dane

Usługa zarządzająca danymi

Użytkownik

Użytkownik komunikuje się z aplikacją kliencką, natomiast aplikacja kliencka musi się wpierw przedstawić serwerowi z którego chce pozyskać dane.

## Aktorzy w komunikacji

Specyfikacja OAuth wyróżnia następujące podstawowe role:

* **Resource owner** – to zwyczajny użytkownik aplikacji która znajduje się przy komputerze i w danym momencie posługuje się aplikacją.
* **Client** – aplikacja kliencka która komunikuje się z zewnętrznym serwerem z zasobami.
* **Resource server** – usługa serwująca dane np. Dropbox REST API
* **Authorization server** – usługa pośrednicząca w przydzielaniu uprawnień aplikacji do zasobów konkretnego użytkownika

## Rejestracja aplikacji

Proces polega na rejestracji naszej aplikacji na serwerze z którym będziemy się komunikować. Podczas rejestracji potrzebne są nam informacje :

* **Client id** – login aplikacji klienckiej do API
* **Client secret** – hasło aplikacji klienckiej do API
* **Redirect URIs** – jest to element w którym konfigurujemy pod jakimi adresami nasza aplikacja kliencka jest dostępna
* **Scopes** - (parametr opcjonalny) – definiuje do jakich zasobów przydzielamy dostęp
* **Image** – obrazek, nie

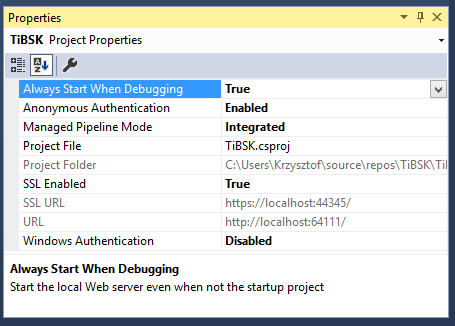
## Rodzaje przepływów

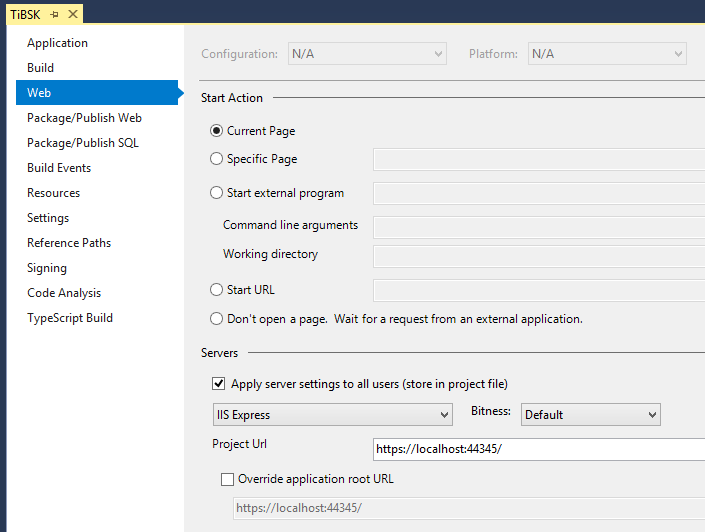
* **Authorization code** (najpopularniejszy)
* Implicit –
* ….
* ….
* … poboczny

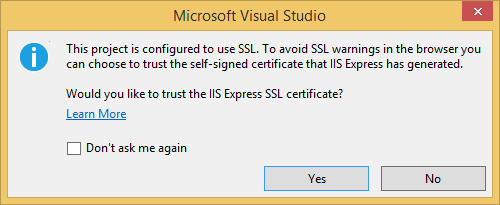
W…..

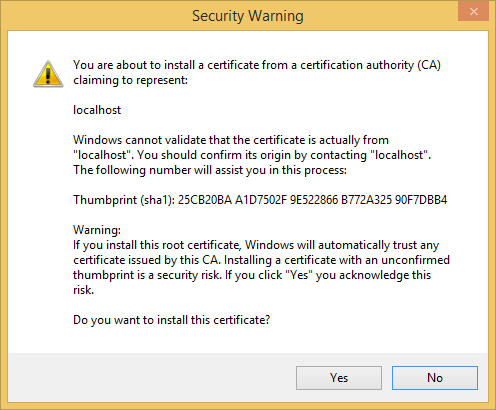
# Implementacja protokołu OAuth 2.0

Wybrana technologia/ język C#.





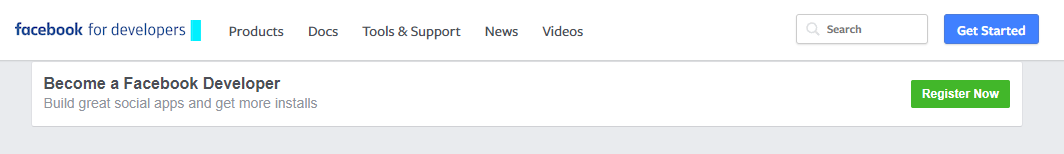




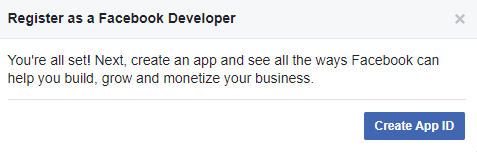


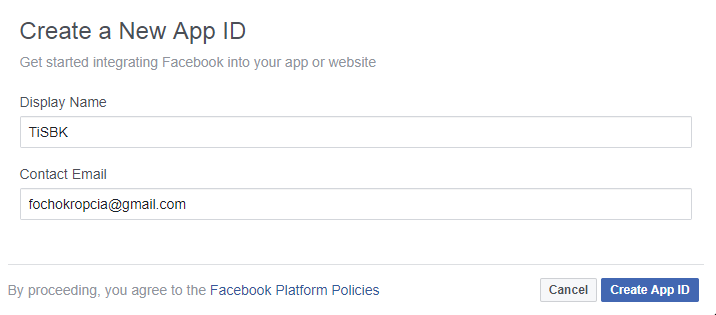
Rejestracja aplikacji w API

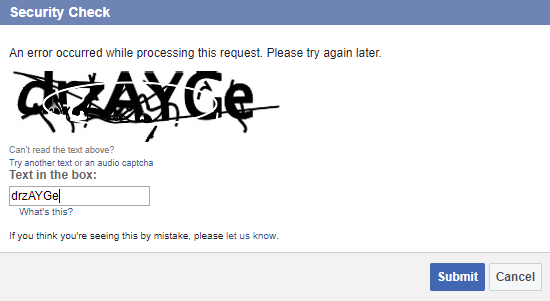
Przechodzimy w tym celu na stronę developerską danego serwisu, w przypadku Facebooka będzie to <https://developers.facebook.com/apps>

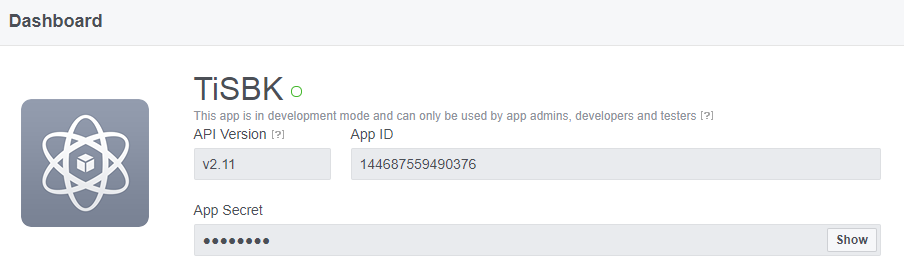


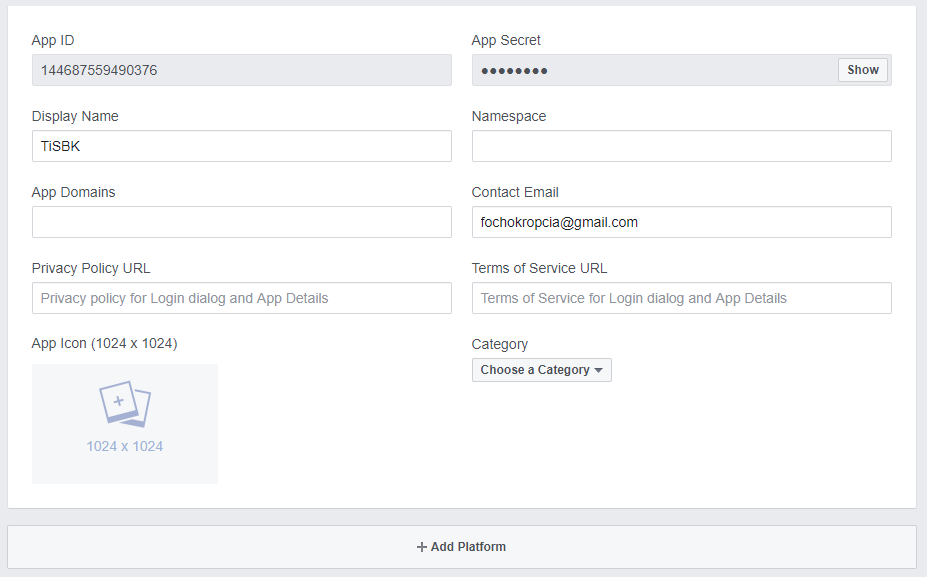


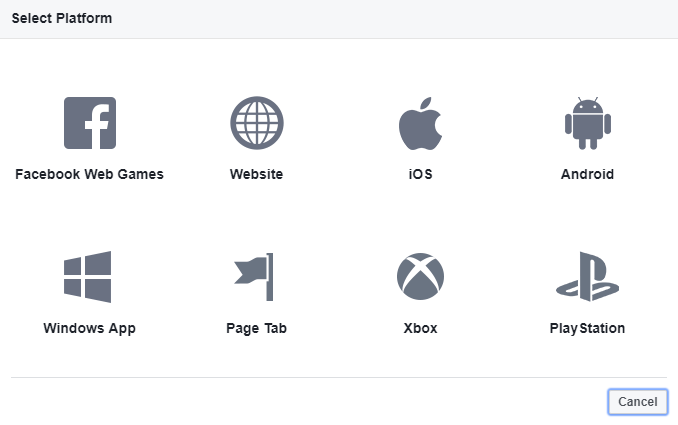


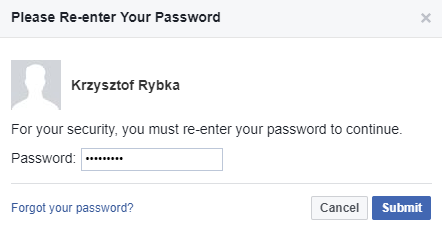


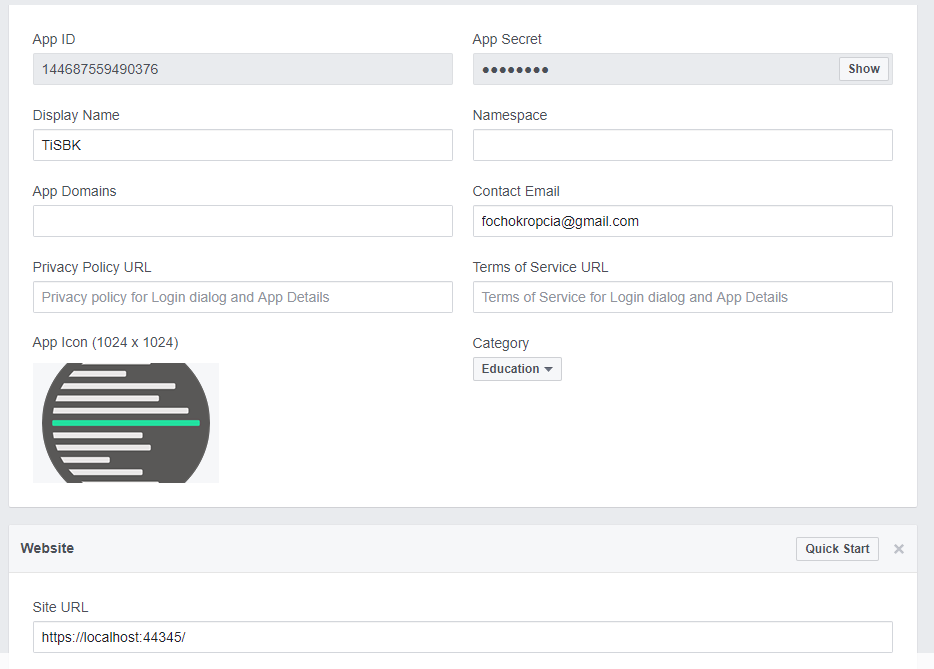


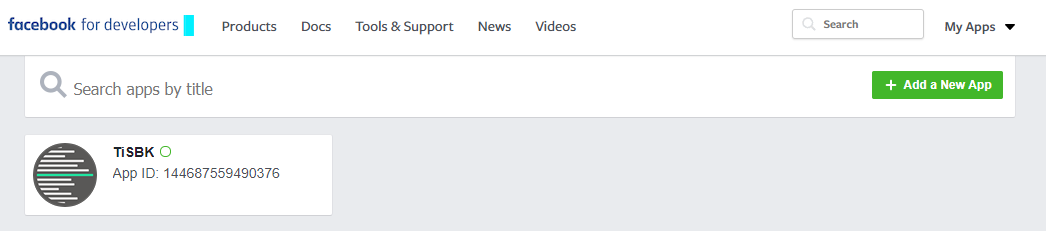


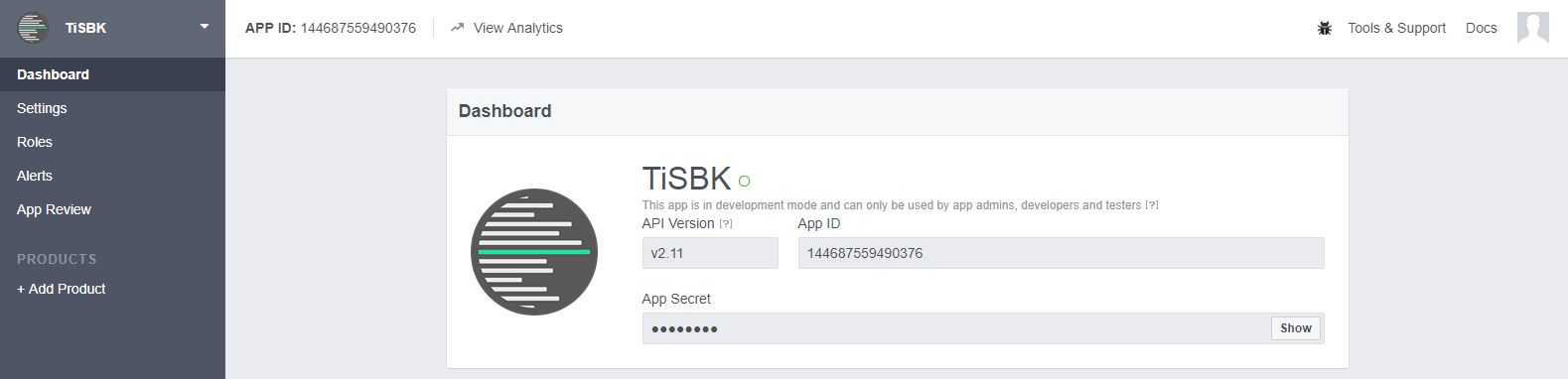












Aplikacja kliencka to ASP MVC 5. Najlepszym dla tego rodzaju projektu będzie wykorzystanie najpopularniejszego przepływu „*Authorization code*”.

# Test aplikacji wykorzystującej implementację protokołu

Hkahdkasdhkasjd

# Podsumowanie

OAuth nie służy do uwierzytelniania, nie jest to rzeczywista wymiana tożsamości tylko wymiana informacji o zasobach, gdzie zasobem jest informacja o użytkowniku. Jest on protokołem który, pozwala nam wydelegować autoryzację na serwer z zasobami ale nie jest on sam w sobie protokołem autoryzacji.

(1:02:00)Sporo pól pozwana nam na zdefiniowanie np. skąd mamy pobierać informacje, nie definiuje schematu tokenu, dla tego powinno się go moim zdaniem określać jako framework do tworzenia własnych protokołów.

# Bibliografia

1. Ryan Boyd, Getting Started with OAuth 2.0, wyd. O’REILLY, 2012 <http://profs.degroote.mcmaster.ca/ads/sartipi/courses/SC/w13/5.Resources/4.Security/Authorization/getting_started_with_oauth_2.0.pdf>