ZAAWANSOWANA BIOMETRIA (MOTOROLA)

PROJEKT

**ReGraph - digitalizacja wykresów**

*Autorzy: Prowadzący ćwiczenia:* Damian Janas Adam Szczepański

Marcin Jezierski

Kamil Dziedzic

Kraków, 3 września 2014

**SPIS TREŚCI**

1. [Opis aplikacji](#Opis)
2. [Pierwsze kroki - jak używać aplikacji](#pierwsze)
3. [Użyte technologie](#uyte)
   1. Wzorzec projektowy
   2. Biblioteki i frameworki
4. [Funkcjonalności](#funkc)
   1. Wczytywanie wykresu
   2. Rotacja obrazu
   3. Nadanie nazwy wczytanemu wykresowi oraz osiom
   4. Ustawienie zakresu osi
   5. Nadanie nazwy oraz koloru dla każdej linii na wykresie
   6. Ręczne dodanie brakujących lub źle wczytanych punktów
   7. Zapis powstałego wykresu
5. [Przykładowe rezultaty](#result)

**1. Opis aplikacji.**

Nasza aplikacja nazywa się ReGraph i służy do digitalizacji wykresów zarówno liniowych jak i punktowych. Dzięki zastosowaniu Windows Universal Apps nie musimy tworzyć oddzielnych aplikacji dla różnych typów urządzeń z Windows. Działa ona zarówno na smartfonach z Windows Phone, tabletach jak i komputerach z Windows, a nawet Xbox One.

Jeśli chodzi o funkcjonalności aplikacji, zostaną one przedstawione w dalszej części dokumentu, ale pokrótce są to:

- wczytywanie wykresów na 3 sposoby: z dysku w formie obrazu, przechwycenie z wbudowanego aparatu urządzenia oraz wczytanie z pliku CSV

- obrót wczytanego obrazu o zadany kąt

- nadanie nazw, zarówno wykresowi jak i osiom, na 2 sposoby: ręcznie oraz za pomocą OCR

- ustawienie zakresu osi

- ustawienie różnych kolorów i nazw dla każdego wykresu

- manualne dodanie brakujących lub źle wczytanych punktów

Aplikacja została stworzona według wzorca projektowego MVVM (Model-View-ViewModel) oraz korzysta z takich bibliotek oraz frameworków jak: Caliburn.Micro, Calliisto oraz WinRTXamlToolkit.

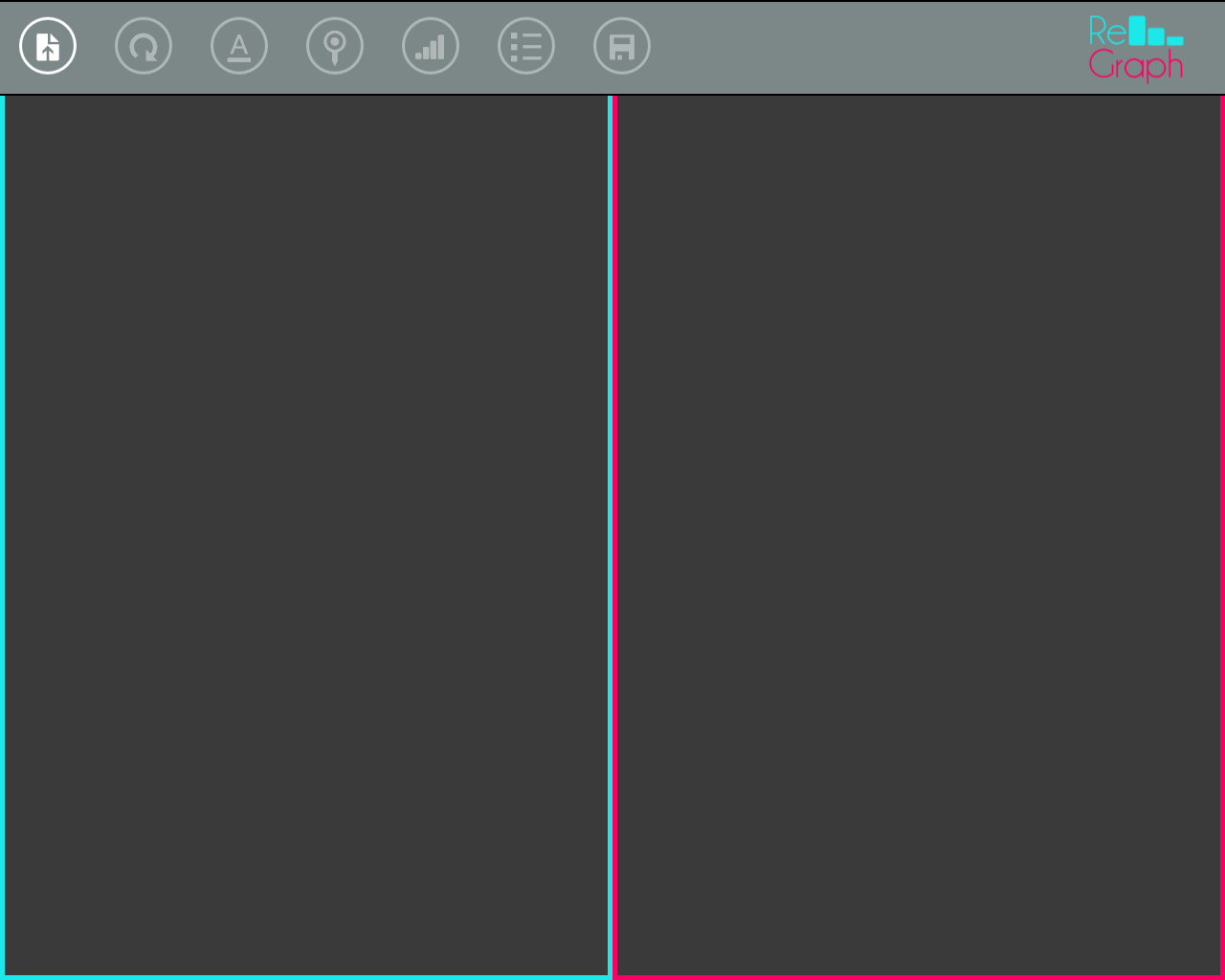
**2. Pierwsze kroki - czyli jak używać aplikacji.**

W tym punkcie postaramy się przedstawić w kilku krokach sposób w jaki należy korzystać z ReGraph.

**2.1. Krok 1: Uruchomienie aplikacji.**

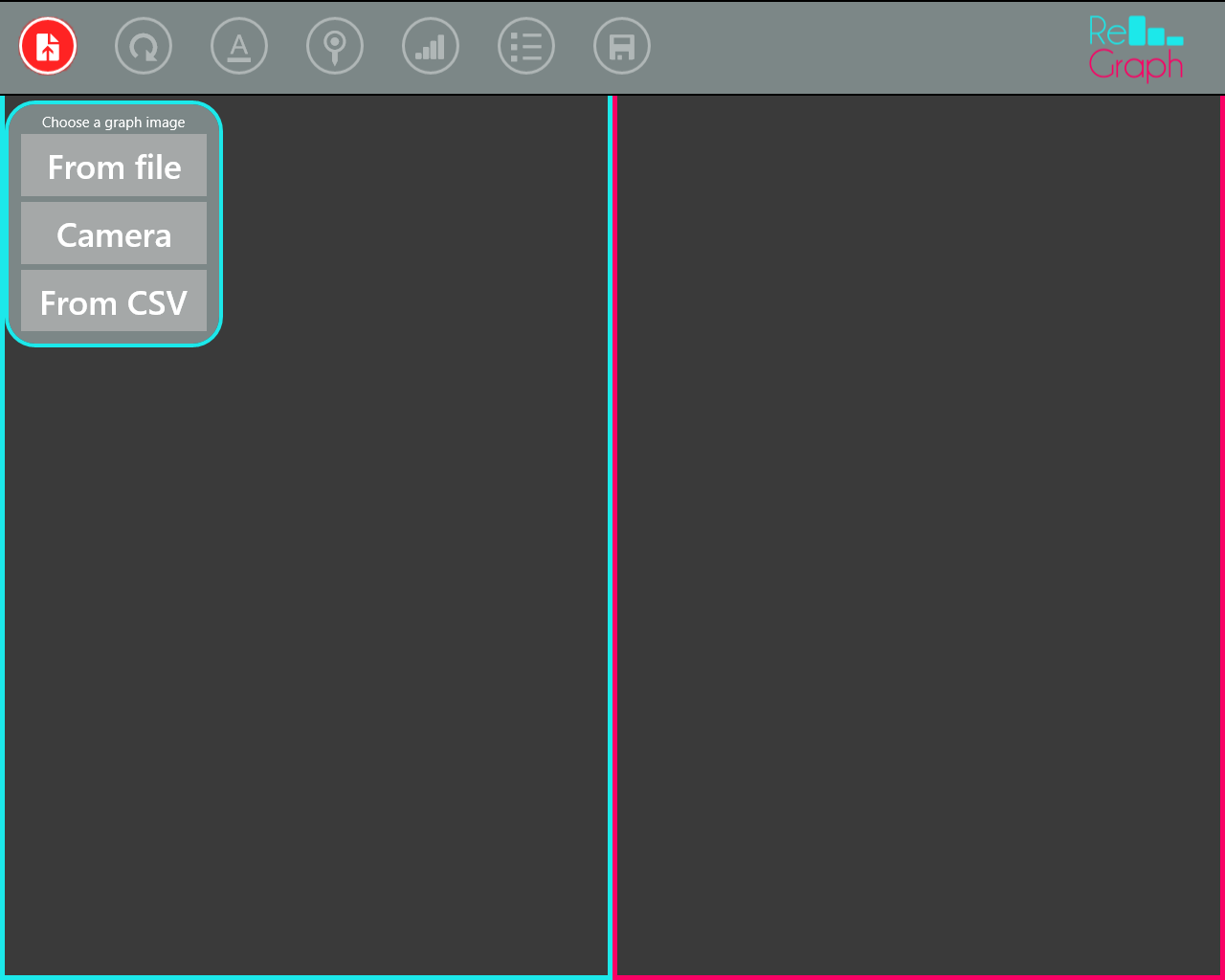
Na początku uruchamiamy naszą aplikację. Po uruchomieniu ukaże się nam główny wygląd aplikacji z dwoma panelami. Po lewej znajduje się panel do którego wczytujemy wykres w formie obrazu, a po prawej panel na którym będziemy rysować nasz wykres wyjściowy.

Klikamy prawy przycisk myszy, aby wysunąć menu. Wygląd głównego ekranu z wysuniętym menu.



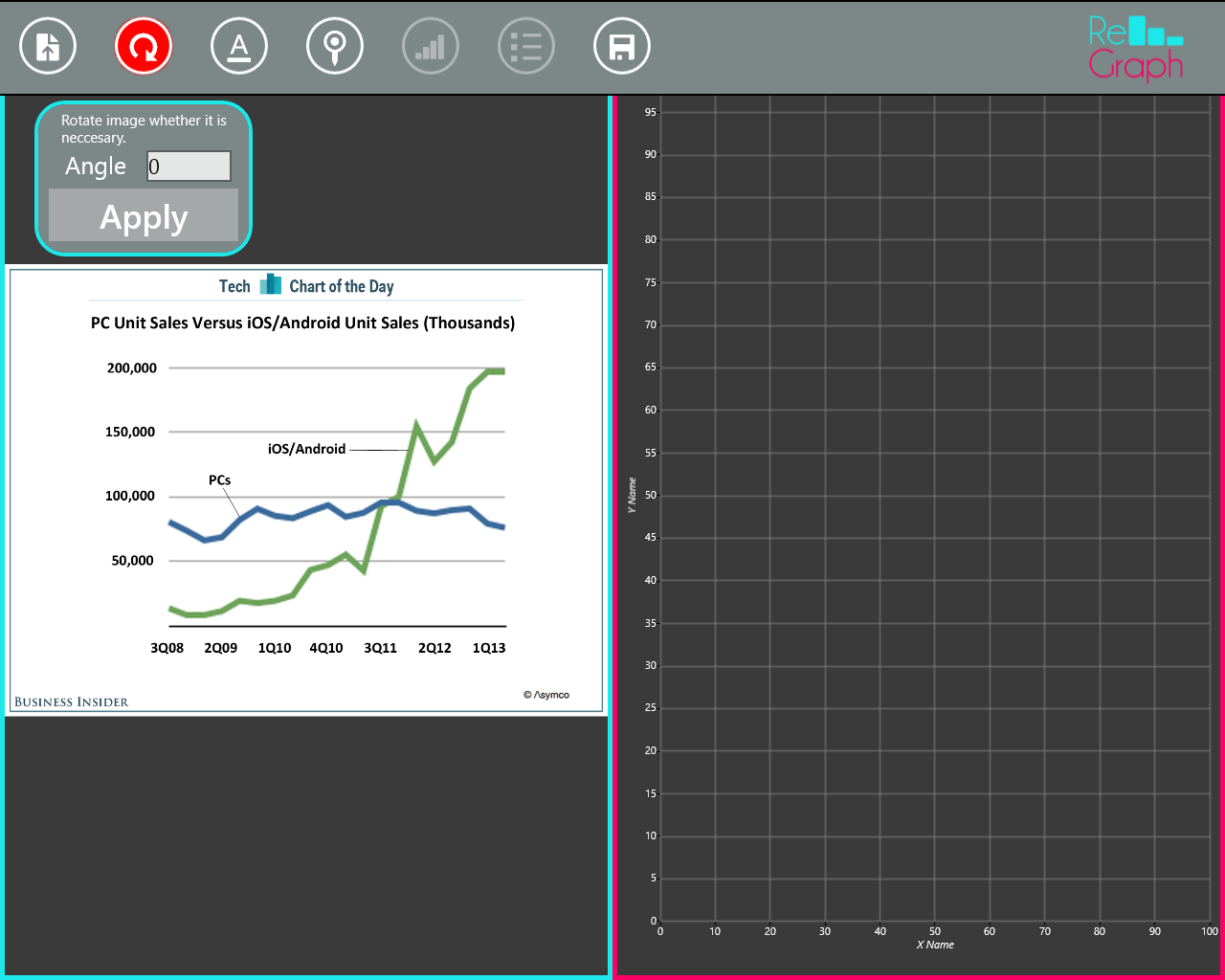
**2.2. Krok 2: Wczytanie wykresu, za pomocą jednego z trzech sposobów.**

Gdy uruchomimy naszą aplikację i wysuniemy pasek z menu, musimy wczytać nasz wykres do analizy za pomocą jednego z trzech sposobów. Aby to zrobić klikamy na poniżej zaznaczony przycisk i wybieramy jedną z opcji:



**2.3. Krok 3: Rotacja wczytanego obrazu (opcjonalnie).**

Po wczytaniu obrazu odblokowują nam się trzy kolejne przyciski. Jeśli wczytany obraz nie jest ustawiony prosto lub jest odwrócony, możemy obrócić go o zadany kąt. Aby to zrobić należy kliknąć na przycisk zaznaczony na obrazku poniżej, wpisać wartość kąta o jaki chcemy obrócić obraz i kliknąć Apply.



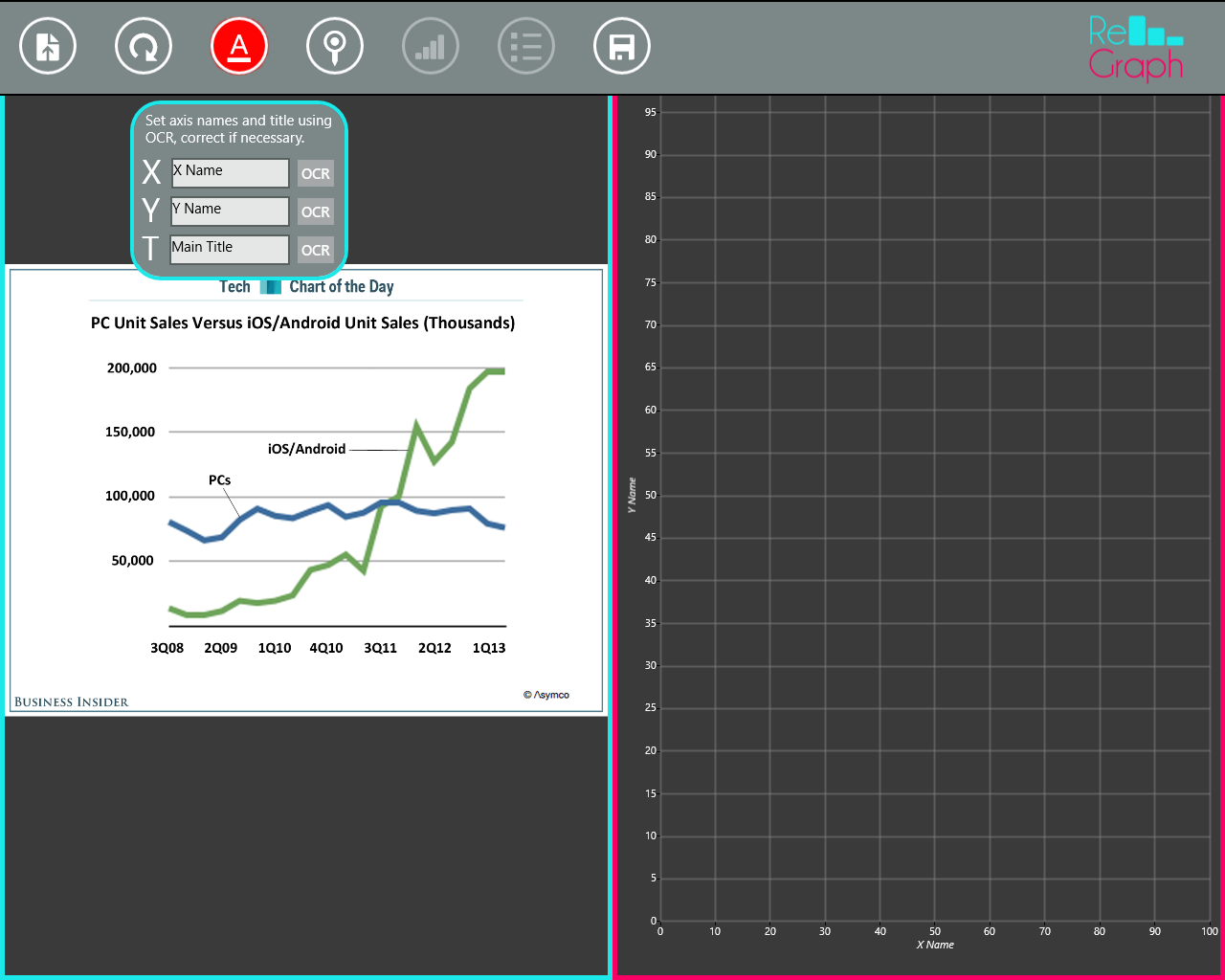
**2.4. Krok 4: Nadanie nazwy wykresowi oraz osiom (opcjonalnie).**

Kolejnym funkcjonalnością która odblokowuje nam się po załadowaniu obrazu jest nadanie nazwy wykresowi oraz osiom. Aby tego dokonać klikamy na przycisk zaznaczony poniżej, a następnie wpisujemy nasze nazwy lub rozpoznajemy nazwy na obrazie za pomocą OCR ( OCR nie został zaimplementowany do końca, szczegóły w opisie funkcjonalności). Po wpisaniu wartości pojawią się one automatycznie na panelu z prawej strony. Jeśli pominiemy ten krok, zostaną użyte domyślne wartości, które są następujące:

- nazwa wykresu - "Main Title"

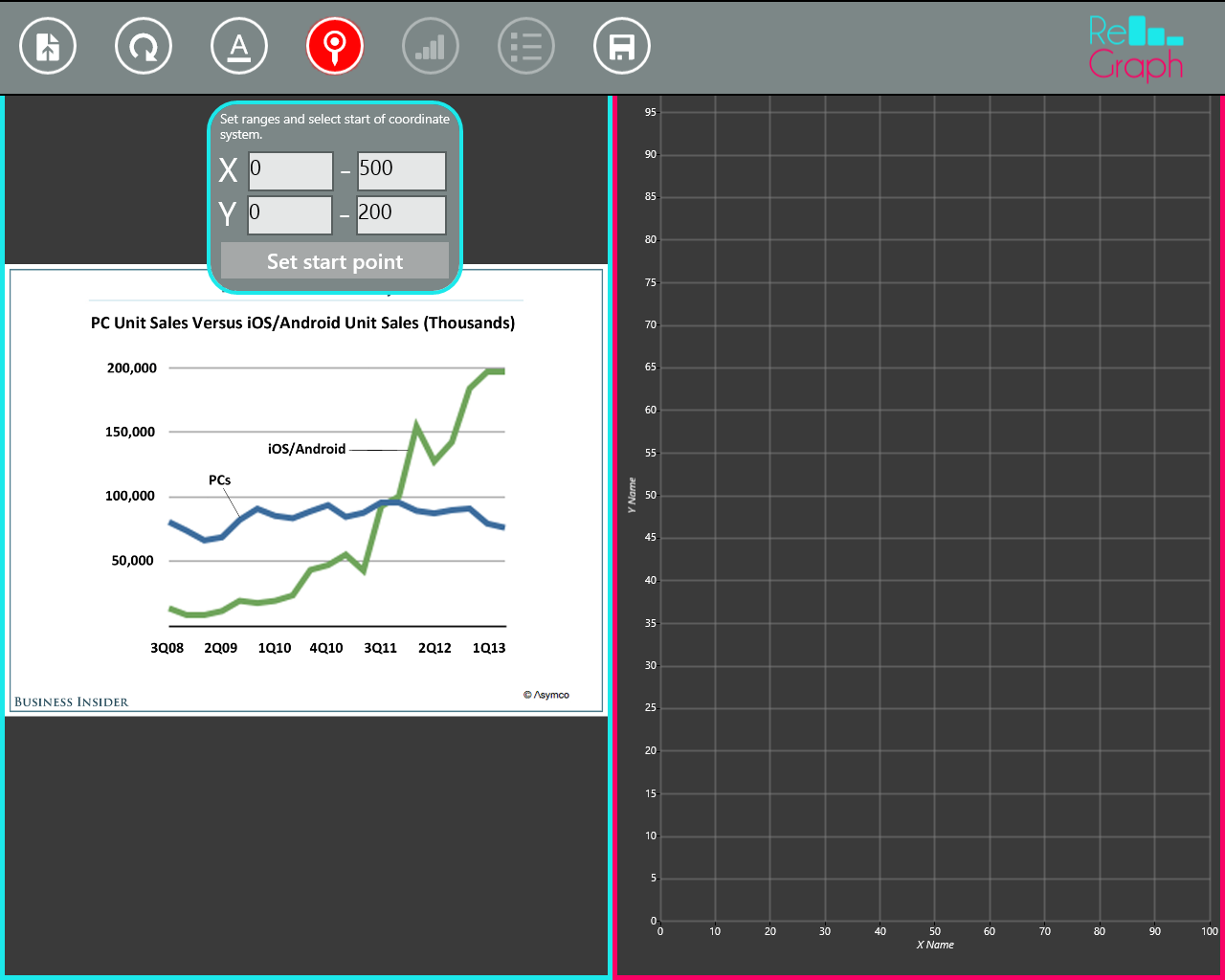
- oś X - "X Name"

- oś Y - "Y Name"



**2.5. Krok 5: Ustawienie zakresu osi oraz punktu początkowego.**

Kolejnym krokiem jest wpisanie wartości zakresów osi oraz ustawienie punktu początkowego wykresu (zazwyczaj jest to punkt początku osi układu współrzędnych). Aby to zrobić należy kliknąć na przycisk zaznaczony poniżej i wpisać interesujące nas wartości. Następnie klikamy przycisk "Set Start Point" i zaznaczamy nasz punkt początkowy na obrazie za pomocą prawego przycisku myszy.

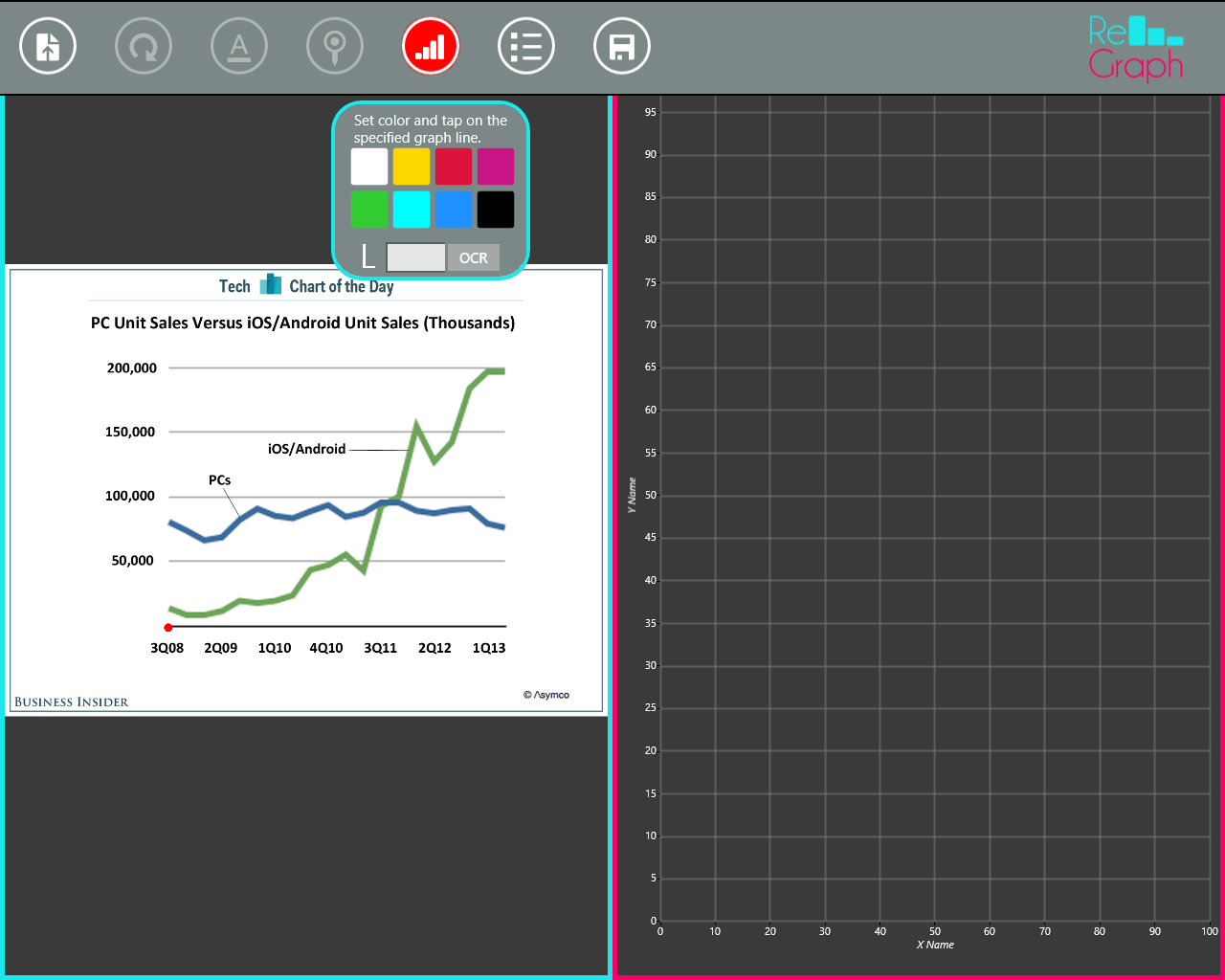


**2.6. Krok 6: Wybór koloru i kliknięcie na linię wykresu.**

Następną rzeczą jaką musimy zrobić to wybranie koloru, którym ma być rysowana linia wykresu, opisanie linii oraz zaznaczenie linii na wykresie znajdującym się na obrazie wejściowym.

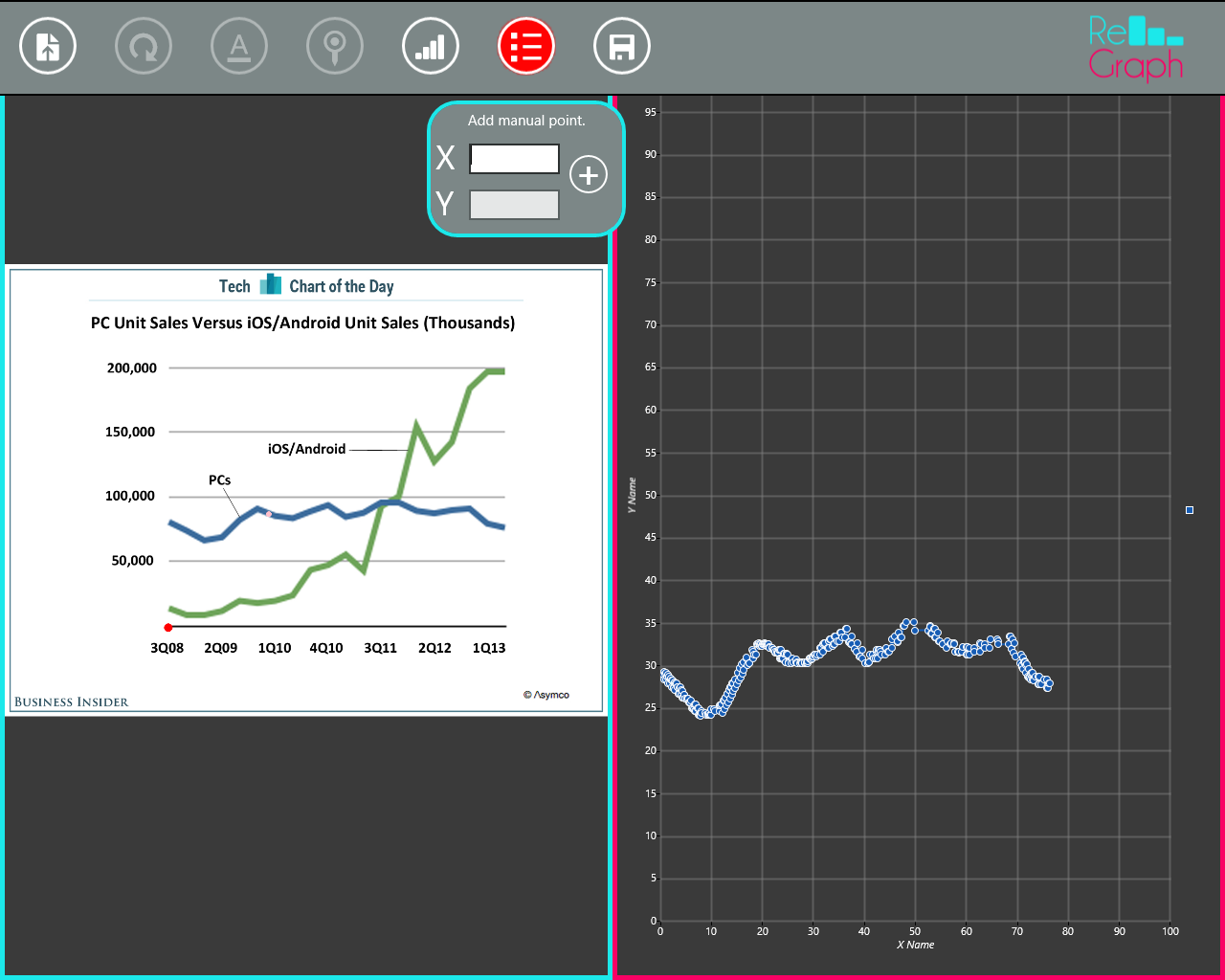
Aby to zrobić klikamy w przycisk zaznaczony poniżej, wybieramy kolor klikając na niego lewym przyciskiem myszy, wpisujemy nazwę dla naszej linii wykresu (opcjonalnie) i następnie klikamy prawym przyciskiem na obrazie wejściowym w punkt, przez który przebiega interesująca nas linia wykresu.

Ten krok możemy powtarzać wielokrotnie, dla różnych linii na wykresie z obrazu wejściowego.



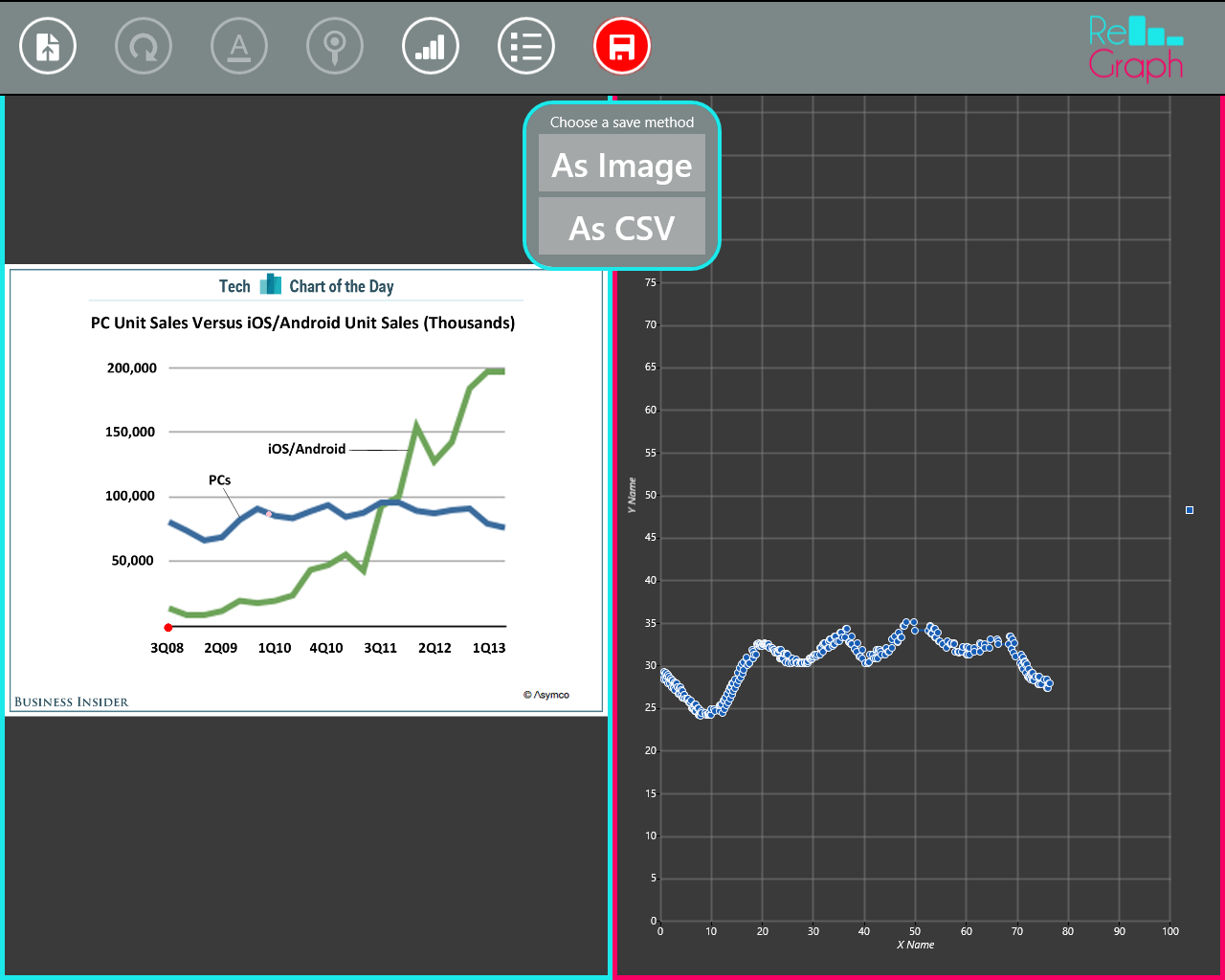
**2.7. Krok 7: Manualne dodanie brakujących punktów (opcjonalnie).**

W tym kroku mamy możliwość ręcznego dodania do wykresu dodatkowych punktów. Aby to zrobić należy kliknąć na przycisk zaznaczony poniżej, wpisać współrzędne naszego punktu i kliknąć "+". Po tych czynnościach nasz punkt automatycznie pojawi i dopasuje się do naszego narysowanego wykresu.



**2.8. Krok 8: Zapis powstałego wykresu.**

Ostatnim krokiem w działaniu naszej aplikacji jest zapis powstałego wykresu. Aby to zrobić należy kliknąć na przycisk zaznaczony poniżej. Mamy do dyspozycji 2 formaty zapisu wykresu: jako obraz lub jako plik CSV.



**3. Użyte technologie.**

**3.1. Wzorzec projektowy.**

W naszym projekcie użyliśmy wzorca projektowego MVVM (Model-View-ViewModel).

Umożliwia on oddzielenie warstwy prezentacji od warstwy logiki aplikacji. Dzięki temu można w łatwy sposób stworzyć aplikację która jest testowalna, prosta w rozbudowie i pozwala na uzyskanie wysokiej reużywalności kodu w przyszłych projektach.

Model:

Odpowiada w omawianym wzorcu za logikę biznesową aplikacji. Często jest tworzony przy użyciu obiektowych mapperów baz danych (np. nHibernate, Entity Framework, LINQ2SQL, WCF Services itp). Klasy w modelu najczęściej implementują interfejs INotifyPropertyChanged, który współpracuje z bindingiem wykorzystywanym w WPF oraz Silverlight.   
Korzystając z interfejsu IDataErrorInfo możemy zaimplementować sprawdzanie poprawności wprowadzanych danych przez użytkownika.  
Patrząc z perspektywy View i ViewModel, Model nie ma referencji do żadnej z tych warstw - jest od nich całkowicie niezależny.

View:

W idealnym rozwiązaniu kod widoku zawiera tylko konstruktor wywołujący metodę InitializeComponent(). Można spotkać się jednak z kodem który nie jesteśmy w stanie wyrazić za pomocą XAMLa np. skomplikowaną animacją.  
Widok porozumiewa się z modelem za pomocą ViewModel. Komponenty widoku są bindowane np. za pomocą komend z odpowiednimi metodami w kodzie ViewModel. ViewModel jest połączony z View za pomocą właściwości DataContext którą udostępnia View.

ViewModel:

Reprezentuje dane wysyłane do widoku - nie mając przy tym żadnej referencji do widoku. Widok za pomocą komend (command) oraz bindowania do właściwości odwołuje się do składników ViewModel. Takie rozwiązanie umożliwia testowanie ViewModelu odrębnie od części prezentacyjnej oraz modelu. Często w ViewModel-u wprowadza się dodatkową walidację czy też dodatkową logikę przedstawianą w warstwie prezentacji dla użytkownika.  
Jak rozpoznać czy daną operację umieścić w widoku czy też w ViewModel? Odpowiedź jest prosta:  
View - wszystko co jest związane z prezentacją danych użytkownikowi  
ViewModel - wszystko co związane z logiką i danymi

**3.2. Biblioteki i frameworki.**

Biblioteki i frameworki wykorzystane w naszym projekcie: WinRTXamlToolkit, Caliburn.Micro, Callisto.

**3.2.1. Caliburn.Micro**

Caliburn.Micro jest małym ale bardzo potężnym frameworkiem, przeznaczonym do budowania aplikacji na wszystkie platformy XAML. Z silnym wsparcie dla wzorca projektowego MVVM i innych sprawdzonych wzorców UI pozwala na szybką budowę aplikacji bez poświęcenia jakości kodu i testowalności.

**3.2.2. Callisto**

Callisto jest biblioteką używaną w aplikacjach XAML-owych w Windows Store. Dostarcza on dodatkowe funkcjonalności dla frameworka Windows.UI.Xaml

**3.2.3. WinRTXamlToolkit**

Jest to zestaw kontrolek, rozszerzeń i klas pomocniczych dla aplikacji Windows Runtime XAML.

**4. Funkcjonalności.**

**4.1. Wczytywanie wykresu.**

ReGraph umożliwia trzy sposoby wczytania wykresu:

1 .Wczytanie wykresu z dysku w formie obrazka. Po wybraniu tej opcji, ukazuje nam się okno FilePickera, w którym znajdujemy i wczytujemy nasz obraz.

2. Przechwycenie obrazu z podglądu kamery wbudowanej w urządzenie na którym aplikacja jest uruchomiona. Po wybraniu tej opcji, ukazuje nam się okno z podglądem wbudowanej kamery. Jeśli użyliśmy tej opcji pierwszy raz, musimy wyrazić zgodę na dostęp. Następnie możemy zrobić zdjęcie i przechwycić je do odczytu.

3. Wczytanie danych do wykresu w formie pliku CSV. Ostatnia opcja umożliwia nam wczytanie danych z pliku, również za pomocą FilePickera, w formacie CSV. Po otwarciu pliku, wykres rysowany jest automatycznie.

**4.2. Rotacja obrazu.**

Po wczytaniu obrazu mamy możliwość jego obrócenia o zadany kąt. Jeśli wczytany obraz jest obrócony, należy przywrócić go do prawidłowego ustawienia, za pomocą dostępnej rotacji.

W naszej aplikacji użyliśmy rotacji bez przycinania, aby nie tracić danych z obrazu.

**4.3. Nadanie nazwy wczytanemu wykresowi oraz osiom.**

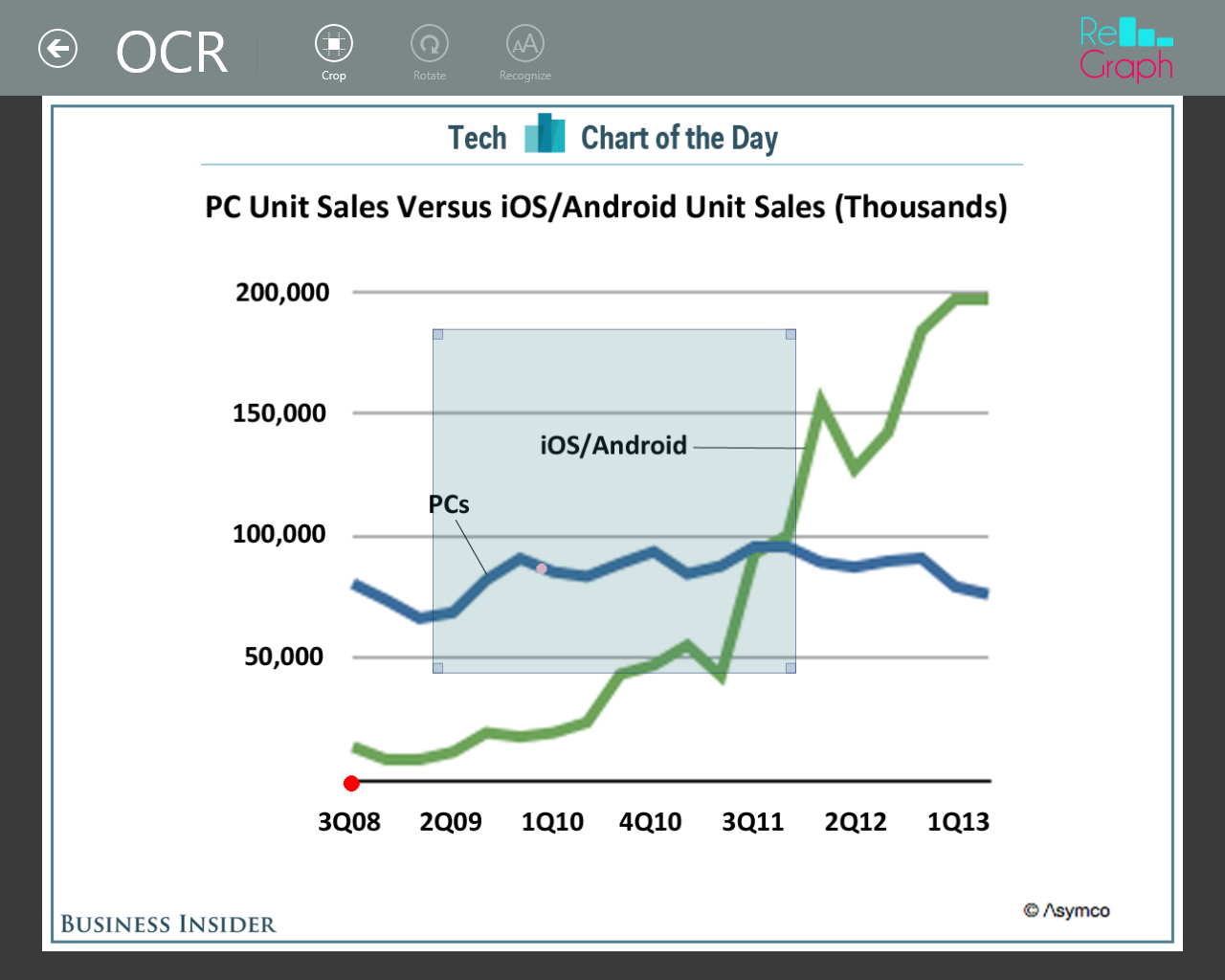
ReGraph umożliwia nadanie nazw dla rysowanego wykresu oraz jego osiom współrzędnych.

W założeniu powinno to realizowane na dwa sposoby.

Pierwszy - manualny - działa poprawnie i umożliwia ustawienie dowolnych nazw.

Drugi sposób to wycięcie z obrazu fragmentu zwierającego tekst i poddanie go działaniu zaimplementowanego OCR. Niestety algorytm OCR nie został przez nas dokończony.

Osoba odpowiedzialna za ten algorytm, głównie z powodu ważnych przyczyn osobistych, nie zdążyła dokończyć jego implementacji. Funkcjonalność rozpoznawania za pomocą OCR jest podpięta pod GUI i posiada osobne okno, które wygląda następująco:



Jak widać na załączonym obrazku składa się on z trzech etapów.

Pierwszy etap to przycięcie obrazu do obszaru, który przedstawia interesujący nas tekst. Możemy przesuwać i zmieniać rozmiar widocznego obszaru za pomocą myszki. Należy również zaznaczyć że po przycięciu obrazu, nie tracimy oryginału, tak samo przycięty obraz nie zastępuje oryginału na panelu w głównym oknie, lecz istnieje tylko na potrzeby OCR.

Drugi etap to rotacja obrazu po przycięciu, jeśli to konieczne. Mianowicie, jeśli nasz tekst na obrazie nie jest ustawiony prosto, należy go obrócić przed poddaniem do analizy.

Trzeci etap to samo rozpoznawanie. Jak już wspominaliśmy, nie jest dokończone i nie daje takich efektów, które umożliwiałyby korzystanie z OCR na potrzeby tej aplikacji. W chwili obecnej posiadamy za małą bazę oraz niewystarczający pre-processing. Możemy jedynie powiedzieć że algorytm rozpoznaje litery na podstawie następujących cech: stosunek wysokości do szerokości litery oraz stosunek czarnych do wszystkich pikseli w kilka obszarach. Możemy jeszcze dodać, że mamy zamiar rozwinąć ten algorytm w najbliższej przyszłości.

Wszystkie kroki w oknie OCR muszą być wykonywane w podanej kolejności aby można było wykonać rozpoznanie.

**4.4. Ustawienie zakresu osi.**

Mamy również możliwość ustawienia zakresów osi. Jest to konieczny etap, jeśli chcemy aby nasze wartości na wykresie były takie same jak na obrazie wejściowym.

Bezpośrednio po ustawieniu zakresów, musimy również ustawić punkt startowy na obrazie wejściowym, będący najczęściej początkiem układu współrzędnych.

**4.5. Nadanie nazwy oraz koloru dla każdej linii na wykresie.**

ReGraph umożliwia nam ustawienie koloru dla konkretnego wykresu oraz nadanie mu nazwy (legendy). Pozwala to na rozróżnienie narysowanych wykresów, jeśli jest ich wiele. Nadawanie nazwy, w założeniu, również mogło być realizowane za pomocą OCR (podpięty jest odpowiedni button).

**4.6. Ręczne dodanie brakujących oraz źle wczytanych punktów.**

Mamy również możliwość ręcznego dodawania punktów po automatycznym narysowaniu wykresu. Bardzo przydatna możliwość, w przypadku gdy algorytm pominął istotny dla nas punkt w procesie analizy wykresu. Samo dodanie punktu realizuje się poprzez wpisanie wartości jego współrzędnych i zatwierdzenie.

Istotną rzeczą jest fakt, że dodany punkt automatycznie dopasowuje się do najbliższej linii wykresu, zmieniając jej wygląd tak aby obejmowana ona dodany punkt.

**4.7. Zapis powstałego wykresu.**

ReGraph umożliwia nam zapis powstałego wykresu w dwóch możliwych postaciach.

Pierwszy z nich to zapis w postaci obrazu. Po wybraniu tej opcji pojawia się okno z przeglądem plików, znajdujemy interesującą nas lokalizację, wpisujemy nazwę dla naszego pliku i zapisujemy potwierdzając przyciskiem.

Druga możliwość to zapis do pliku CSV. Powstały plik CSV jest dość specyficzny, ponieważ różni się od tradycyjnych plików CSV tym, że w naszym wypadku przechowujemy również dodatkowe dane takie jak np. zakres wartości na osiach.

**5. Przykładowe rezultaty**

