Piotr Kisiel

Praktyczne aspekty nauki programowania w szkole średniej = TheNuts and Bolts of Programming Science in Secondary School

Dydaktyka Informatyki 13, 147-152

2018

Dydaktyka Informatyki 13(2018)
ISSN 2083-3156; e-ISSN 2543-9847
DOI: 10.15584/di.2018.13.19
http://www.di.univ.rzeszow.pl

Uniwersytet Rzeszowski

Piotr KISIEL

Dr inż., Uniwersytet Rzeszowski, Wydział Sztuki, ul. Kopisto 1; 35-959 Rzeszów, I Liceum Ogólnokształcące im. Juliusza Słowackiego w Przemyślu; e-mail: piotrkisiel@wp.pl

PRAKTYCZNE ASPEKTY NAUKI PROGRAMOWANIA W SZKOLE ŚREDNIEJ

THE NUTS AND BOLTS OF PROGRAMMING SCIENCE IN SECONDARY SCHOOL

Słowa kluczowe: programowanie, kompilacja, program, android, cpp, nauka programowania.
Keywords: Programming, compilation, program, android, cpp, programming learning.

Streszczenie

Korelacja wiedzy zdobytej w procesie edukacji z otaczającym światem ma kluczowe znaczenie. Artykuł przedstawia nie tyle modyfikację treści nauczania programowania na lekcjach rozszerzonej Informatyki, co poprzez dobór środowiska programowania, umożliwienie tworzenia atrakcyjnych i użytkowych aplikacji na urządzenia mobilne wykorzystywane przez uczniów na co dzień.

Abstract

Correlation of knowledge acquired in the education process with the surrounding world is crucial. The article presents not so much the modification of the teaching content of Programming Science, rather due to the selection of the programming environment, enabling the creation of attractive and utilitarian applications for modern devices.

Wstęp

„Obyś cudze dzieci uczył”,
bo rzeczywiście nie ma chyba większego wyzwania
ponad kształtowanie osobowości młodych ludzi,
wytyczanie szlaków ich wiedzy
i wpływanie na życiowe postawy.

J. Klejnocki[1]

[1] J. Klejnocki, *Koniec pewnych czasów*, http://www.twinpigs.zory.pl/dzien-nauczyciela-w--twinpigs-caly-dzien-za-10zl (dostęp: 27.12.2017 r.).

Nauczyciele informatyki z całą pewnością mogą zaświadczyć o tym, jak dynamicznie zmieniła się rzeczywistość na przełomie ostatniej dekady. Nie jest moim celem porównywanie treści programów nauczania z przedmiotów, np. matematyki czy fizyki w stosunku do informatyki. Jednak na kilka aspektów sposobu nauczania należy zwrócić uwagę chcąc zapewnić teraźniejszość i atrakcyjność prezentowanych uczniom treści.

Pedagodzy, którzy rozpoczynają naukę programowania, spotykają się z pytaniem: *Kiedy będziemy się uczyć pisać grę?*. Przyznam szczerze, że irytowało mnie to pytanie zadawane rokrocznie na początku zajęć przez uczniów. W końcu jednak postanowiłem głębiej przyjrzeć się temu problemowi. Problemowi, bo trudno w kilku słowach wyjaśnić „młodemu programiście" pełnemu zapału, że to nie takie proste, że najpierw trzeba… itd. Tymczasem „młody programista" pochodzi ze świata, gdzie wszystko jest „na już", w swoim smartfonie posiada informację zarówno o trzęsieniu ziemi sprzed pięciu minut na drugim końcu świata, ale i o pytaniach, które były na sprawdzianie w poprzedzającej jego lekcję grupie. Do wyboru ma też tysiące aplikacji i gier, a wszystko obudowane feerią barw i najważniejsze, to wszystko mieści się w kieszeni…

Zaczynając naukę programowania uczeń ma styczność z kompilatorami, które niezależnie od wpisywanych kodów wyświetlają efekt działania przeważnie w czarnym oknie z białym tekstem. Początkowo to wystarcza. Jest to zagadnienie nowe i absorbujące. I tak omawiając poszczególne typy zmiennych, pętli i tablic tworzymy programy wyliczające średnią spalania, BMI, ratę kredytu w banku, kalkulator, zamianę funkcji z ogólnej na kanoniczną oraz na odwrót, losowanie liczb, sortowanie bąbelkowe itd., itd.

Z czasem nasz „młody programista" nabywa podstawową wiedzę z zakresu programowania. Wyciągając jednak ukradkiem swój smartfon, żeby zobaczyć, co dzieje się na forum dyskusyjnym, zaczyna sobie zadawać pytanie, co to „programowanie" ma wspólnego z rzeczywistością?

Jako praktykujący nauczyciel programowania obserwuję od tego etapu edukacji coraz to większy rozdźwięk pomiędzy oczekiwaniami uczniów a tym co oferuje standardowy proces nauczania. Rosnący poziom trudności omawianych programów wraz z interfejsem z poprzedniej epoki i programy *.exe do wykonania lokalnie na komputerze stacjonarnym, to rozwiązanie, które w żaden sposób nie może być atrakcyjne. Sytuacje ratują tutaj rzesze zestawów, np. *Lego Mindstorms, Roborobo, Makeblock, Arduino* itp.

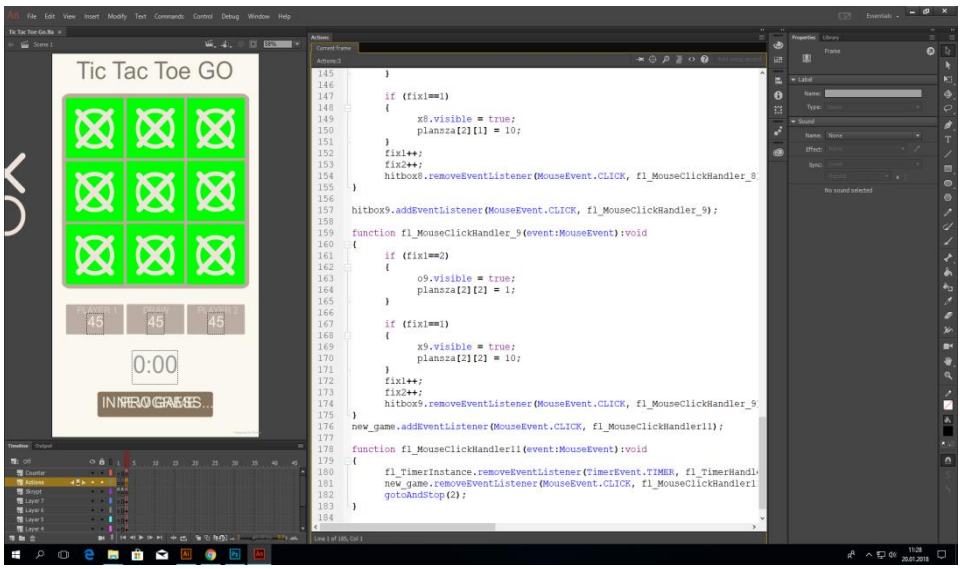
W dalszej części rozważań wrócę jednak do odpowiedzi na pytanie: *co z tymi grami?*.

Nowe środowisko, nowe możliwości

Jako nowe środowisko pracy został wybrany program Adobe Animate, będący składnikiem pakietu Adobe Creative Cloud for Education. Pakiet dostępny

jest dla szkół średnich na bardzo preferencyjnych warunkach i umożliwia on pracę z grafiką rastrową, wektorową, multimediami, technologiami internetowymi, na działaniach interaktywnych kończąc. Wpisuje się on doskonale jako narzędzie edukacyjne zapewniające realizację nowej podstawy programowej dla Informatyki w szkole średniej.

Aby lepiej zilustrować możliwości nowego środowiska pracy posłużę się przykładem realizacji na lekcjach programowania prostej gry „kółko i krzyżyk". Kod bazowy napisany w języku programowania C++ wstępnie został zaczerpnięty z gry Tic Tac Toe[2]. Ze względu na ograniczone miejsce w niniejszym opracowaniu nie przedstawię jego listingu, który jest zresztą dostępny pod wskazanym adresem. Mając na względzie „migrację" projektu do nowego środowiska programowania kod bazowy przepisany został na język ActionScript 3.0. Wprawdzie definicja języka ActionScript 3.0 może wydawać się nieco nietypowa z perspektywy programistów przyzwyczajonych do języków Java lub C++, jednak w praktyce definiowanie typów obiektów za pomocą klas ActionScript 3.0 odbywa się bardzo podobnie, jak w języku Java lub C++[3]. Kod został wzbogacony również o funkcje typowe dla środowiska AIR, umożliwiające intuicyjną pracę z własnoręcznie wykonanymi obiektami graficznymi.

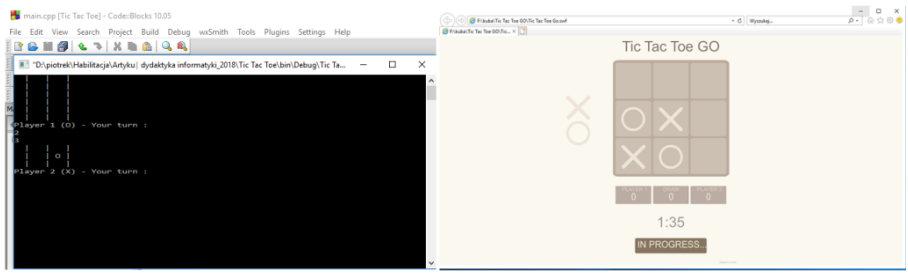


Rys. 1. Zrzut ekranu z programu Adobe Animate CC przedstawiający grę „kółko i krzyżyk"

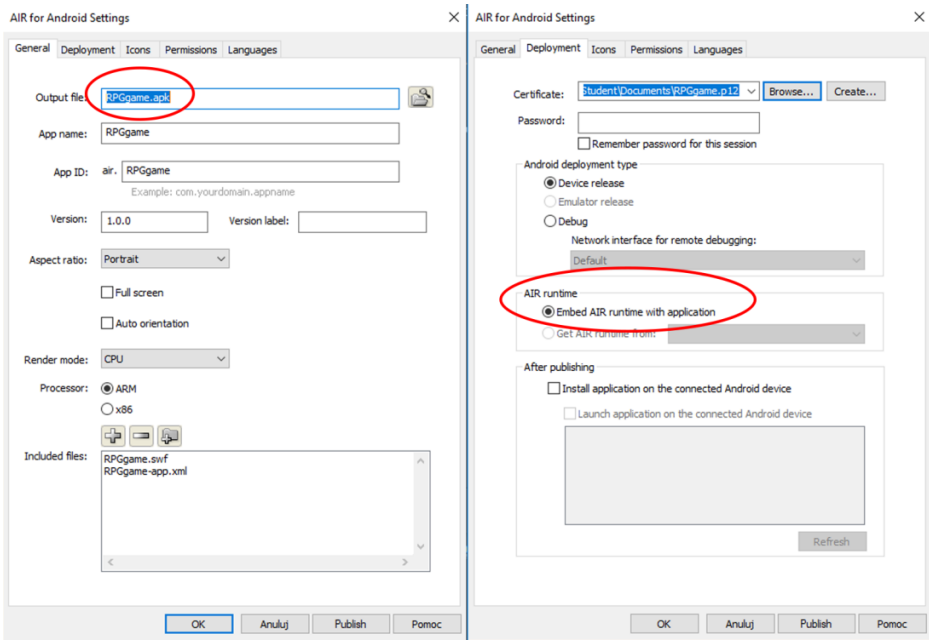
[2] http://cpp-programowanie.cba.pl/projekt-gra-kolko-i-krzyzyk/#more-751 (dostęp: 27.12.2017 r.).

[3] https://help.adobe.com/pl_PL/ActionScript/3.0_ProgrammingAS3/WS5b3ccc516d4fbf351e63e3d118a9b90204-7fa0.html (dostęp: 27.12.2017 r.).

Funkcjonalnie obie realizacje (zarówno w języku C++, jak i AS 3.0.) są tożsame, jednak ich kompilacje na platformie PC znacznie się różnią. Warto zwrócić uwagę na objętość aplikacji wynikowych. Program .exe zajmuje 918 KB podczas gdy program .swf – jedynie 213 KB, pomimo iż prócz części „logicznej" zawiera również treści graficzne. Jest to kolosalna różnica zwłaszcza w realizacji bardziej zaawansowanych projektów.



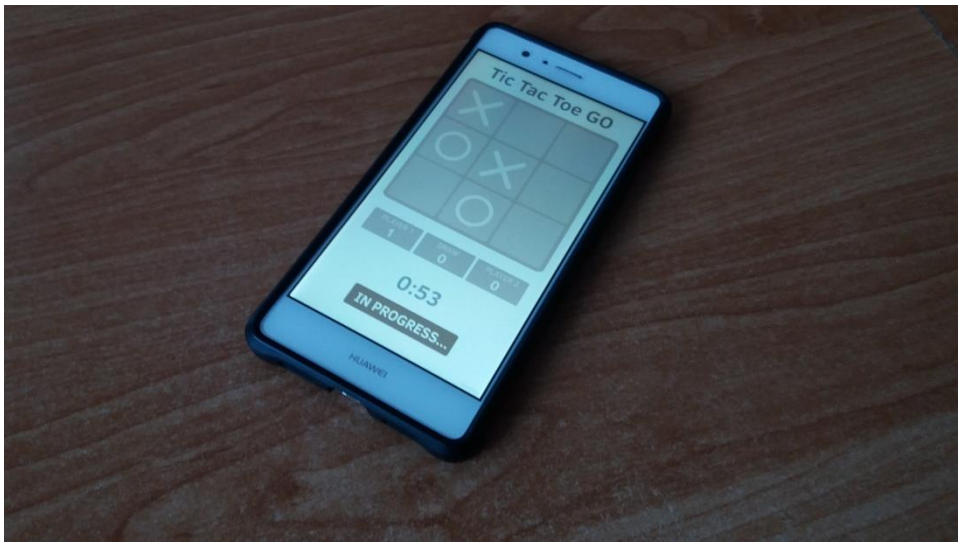
Rys. 2. Wynikowy program .exe kompilacji kodu w języku C++ (po lewej), wynikowy program .swf kompilacji kodu języka AS 3.0 (po prawej)



Rys. 3. Export aplikacji wraz ze środowiskiem wykonawczym dla systemu Android

Najważniejszą różnicą jest łatwość eksportu programu na urządzenia mobilne (plik o rozszerzeniu .apk) z systemem operacyjnym Android. Istnieje również możliwość eksportu aplikacji wraz ze środowiskiem wykonawczym AIR, co

w przypadku pierwszej instalacji tego typu oprogramowania na urządzeniu znacznie ułatwia i przyspiesza proces instalacji.



Rys. 4. „Kółko i krzyżyk" na urządzeniu mobilnym z systemem operacyjnym Android

Zakończenie

Koszula bliższa ciału

Większą uwagę i zainteresowanie poświęcamy sprawom, które nas bezpośrednio dotyczą. Trudno zatem się dziwić, że tworząc aplikacje na przedmioty nieodzowne w życiu młodego człowieka, znajdujące się w centrum jego zainteresowania, przykuwamy jego uwagę. Wykorzystanie programu Adobe Animete CC w nauce programowania daje wymierne efekty w postaci zaangażowania uczniów w realizację tematu.

Osobom, które w praktyce szkolnej nie posiadają dostępu do wyżej omawianego oprogramowania, polecam „Android Studio". Warto również zwrócić uwagę na kompilator „Eclipse" z pluginem „Android Developer Tools". Istnieje też wiele dróg „na skróty", które umożliwiają od razu przystąpienie do programowania gier[4], jednak ze względu na prawidłowy proces edukacyjny, nie polecam ich. Ważne jest, ażeby rozpocząć naukę programowania od podstaw w środowisku, które nie będzie rozpraszało uwagi. W moim przypadku stosuję darmowy Code::Blocks, na bazie którego omawiany jest język C++. Chcąc jednak podtrzymać zapał i zainteresowanie programowaniem wśród młodego pokolenia,

[4] "The Beginner's Guide to Android Game Development".

warto zejść co kilka lekcji z utartych ścieżek i poszerzyć omawianą tematykę i środowisko pracy.

Przedstawiona w artykule gra została zrealizowana przez uczniów klasy IID o profilu matematyczno-informatyczno-fizycznym I Liceum Ogólnokształcącego im. Juliusza Słowackiego w Przemyślu w roku szkolnym 2017/2018.

Bibliografia

Chambers M., Dura D., Hoyt K., Dragos G., *Adobe AIR dla programistów JavaScript.* Leksykon kieszonkowy (ebook), Helion, Gliwice 2012.

Underdahl B., *Flash Głębsze spojrzenie*, Helion, Gliwice 2002.

Underdahl B., *Flash Programowanie w języku ActionScript*, Helion, Gliwice 2002.

Netografia

Adobe Systems Incorporated *Programowanie w języku ADOBE ACTIONSCRIPT 3.0* San Jose, California 2008, https://help.adobe.com/pl _PL/ActionScript/3.0_ProgrammingAS3/flash_as3_ programming.pdf (dostęp czerwiec 2016 r.).

Klejnocki J., *Koniec pewnych czasów,* http://www.twinpigs.zory.pl/dzien-nauczyciela-w-twinpigs--caly-dzien-za-10zl (dostęp: 27.12. 2017).