Lublin University of Technology

Electrical Engineering and Computer Science Faculty

Instytut informatyki

Institute of Computer Science

NATIVE AND CROSS-PLATFORM MOBILE DEVELOPMENT

Maria Skublewska-Paszkowska, Edyta Lukasik
Paweł Grzmil

Lublin University of Technology (POLAND)

Agenda

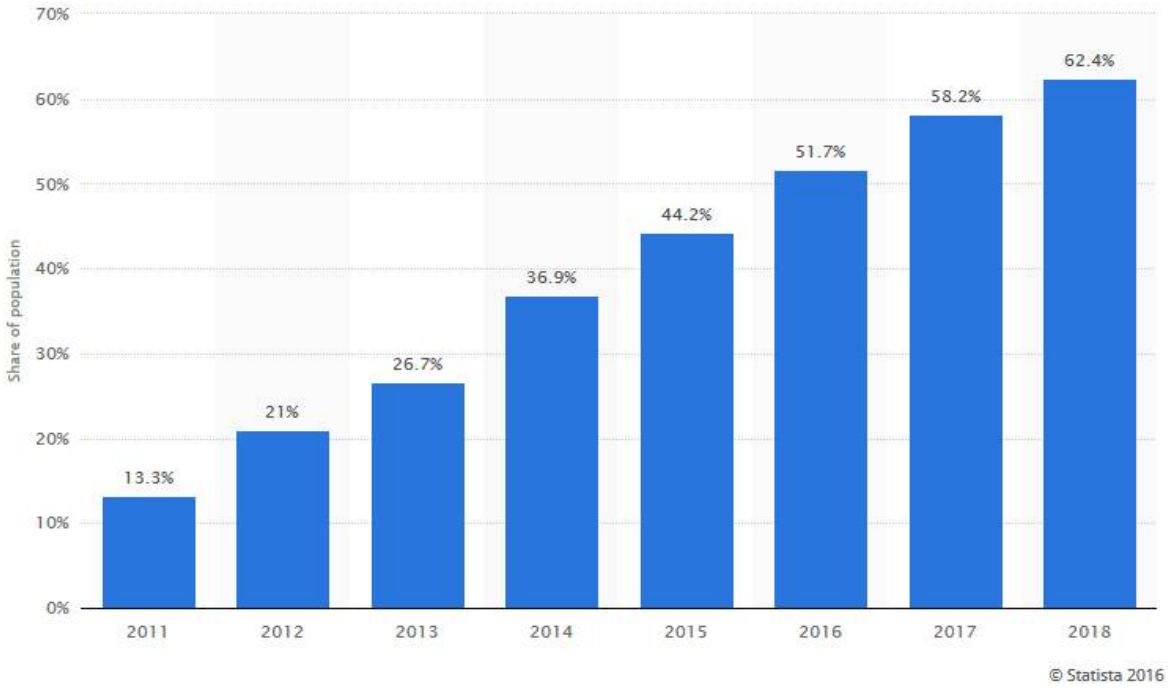
- Introduction
- Mobile applications and their development
- Modern operating systems for mobile devices
- Selected methods of creating mobile applications
- Analysis of the performance of native and cross-platform applications

Introduction

- The goal of every application developer***: to achieve the greatest possible market success***;
- To achieve this, the ***widest possible range of users*** should be reached;
- Leading mobile systems are not compatible with each other and there is a need for software development for each of them;
- It is an expensive and difficult process, which is why there are tools for producing ***multi-platform mobile applications***.
- The investment in a new technology platform is a challenge for enterprises. It is needed to check if it's worth creating cross-platform applications.

Mobile applications and their development

- The development of mobile devices and networks has caused that most of the telephone market is occupied by smartphones;



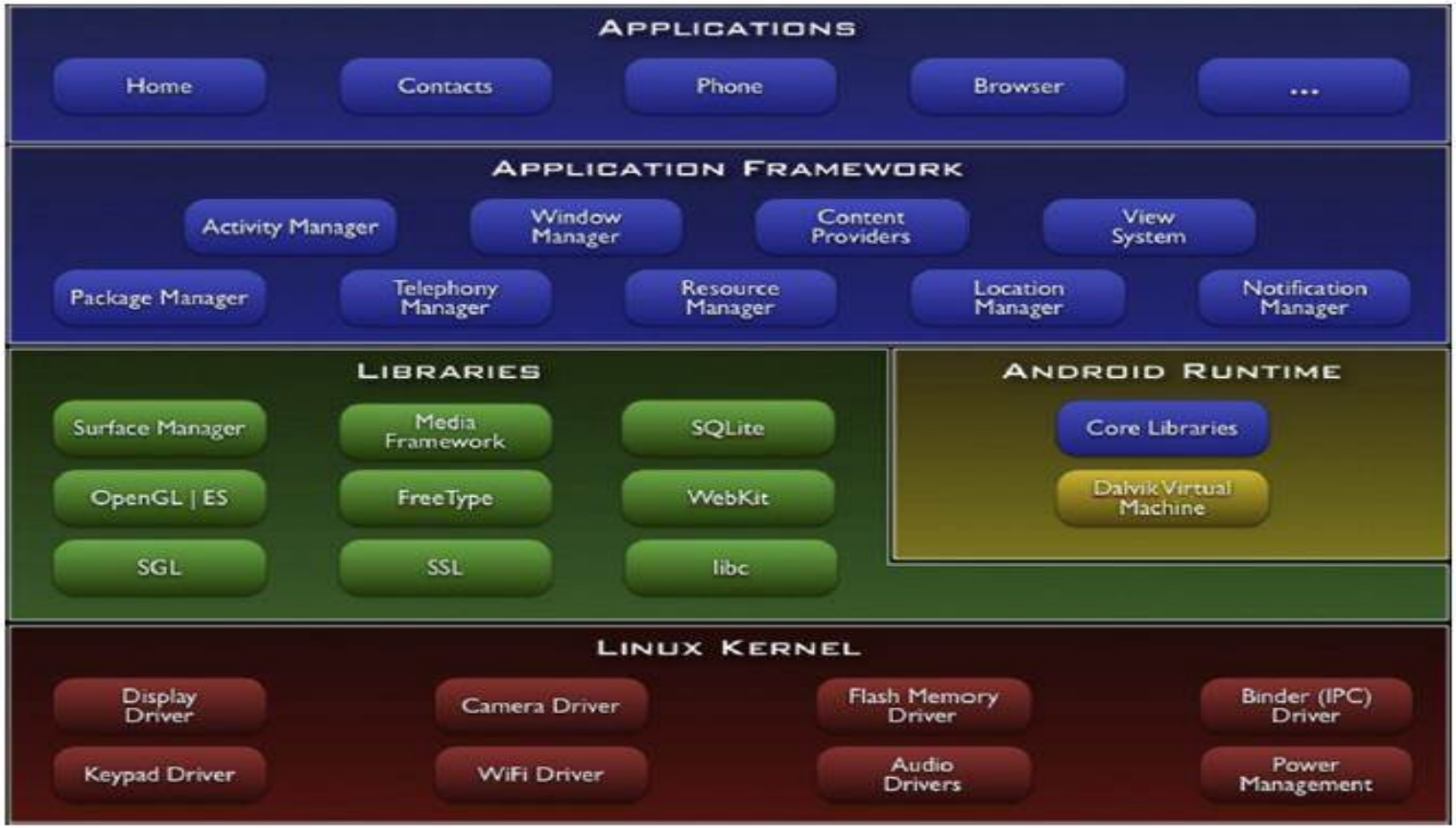
The percentage share of smartphones in the total number of telephones on the market in Central and Eastern Europe is presented

Modern operating systems for mobile devices

Android

- an open-source operating system developed by Android Inc., which was later acquired by Google;
- Google develops the system implicitly, and publishes the code when the next version is released;
- System availability is based on the Apache Software License;
- Despite free and open access to the system code, Android is a trademark of Google, so device manufacturers and mobile operators are forced to sign license agreements to use it.

Modern operating systems for mobile devices



Android system architecture

Modern operating systems for mobile devices

- One of the most important parts of the system is the runtime environment in which applications and part of system services operate;
- For version 5.0 - applications were run on the Dalvik virtual machine;
- Applications are written in Java and compiled to a bytecode;
- It is compiled JIT (just-in-time) from the code to the machine code;
- While navigating the application, additional components are compiled and stored for reuse. This causes a drop in performance and loss of the application's fluency;
- Currently, the default environment for running is ART (Android Runtime) - the compilation to the native machine code occurs during the installation of the application;
- Excluding JIT compilation, resulting in lower CPU usage, which in turn allows smoother application operation and extends battery life.

Modern operating systems for mobile devices

- Android updates are not exclusive to the developer;
- Thanks to the openness of the system's sources, most telephone operators and equipment manufacturers adjust the system to their needs;
- Changes include adding custom applications, interface appearance themes, specific settings, etc;
- Often, phone makers, due to the high competition on the Android smartphone market, create overlays on the system that change the menu display, application management and access to user interface modification settings;
- Before accepting a system update, the user must add and test, on the new version of the system, all the elements they have changed, in the context of compatibility;
- Due to the additional costs, this process is ignored. This contributes to the large fragmentation of the Android smartphone market. This phenomenon is a significant obstacle in the development of mobile applications for this system.

Modern operating systems for mobile devices

- For a long time Android has stopped being a system only for smartphones and tablets;
- Google presented the variations of the system, specially adapted for:
  - TVs (Android TV),
  - car entertainment systems (Android Auto), smartwatches and wearable devices (Android Wear).
- There is a version for x86 processors that can be run on a traditional computer;
- Thanks to a very wide application, Android is the most popular system for mobile devices. In the third quarter of 2015, over 53 percent of mobile devices were controlled by the Google system.

Modern operating systems for mobile devices

iOS

- The premiere of the first iPhone - in June 2007
- It has started a small revolution on the smartphone market
  - a touch screen made in capacitive technology;
  - it enabled finger operation - without the use of a stylus;
- The way the device interacts is designed to maximize the use of natural movements such as scrolling, touching, rotating or expanding;
- With the launch of the phone, Apple released the first version of the operating system that controls the entire family of the company's products. The ***iPhone Operating System*** (the name was shortened to ***iOS***) was built on the basis of the 80's system – NeXTSTEP.

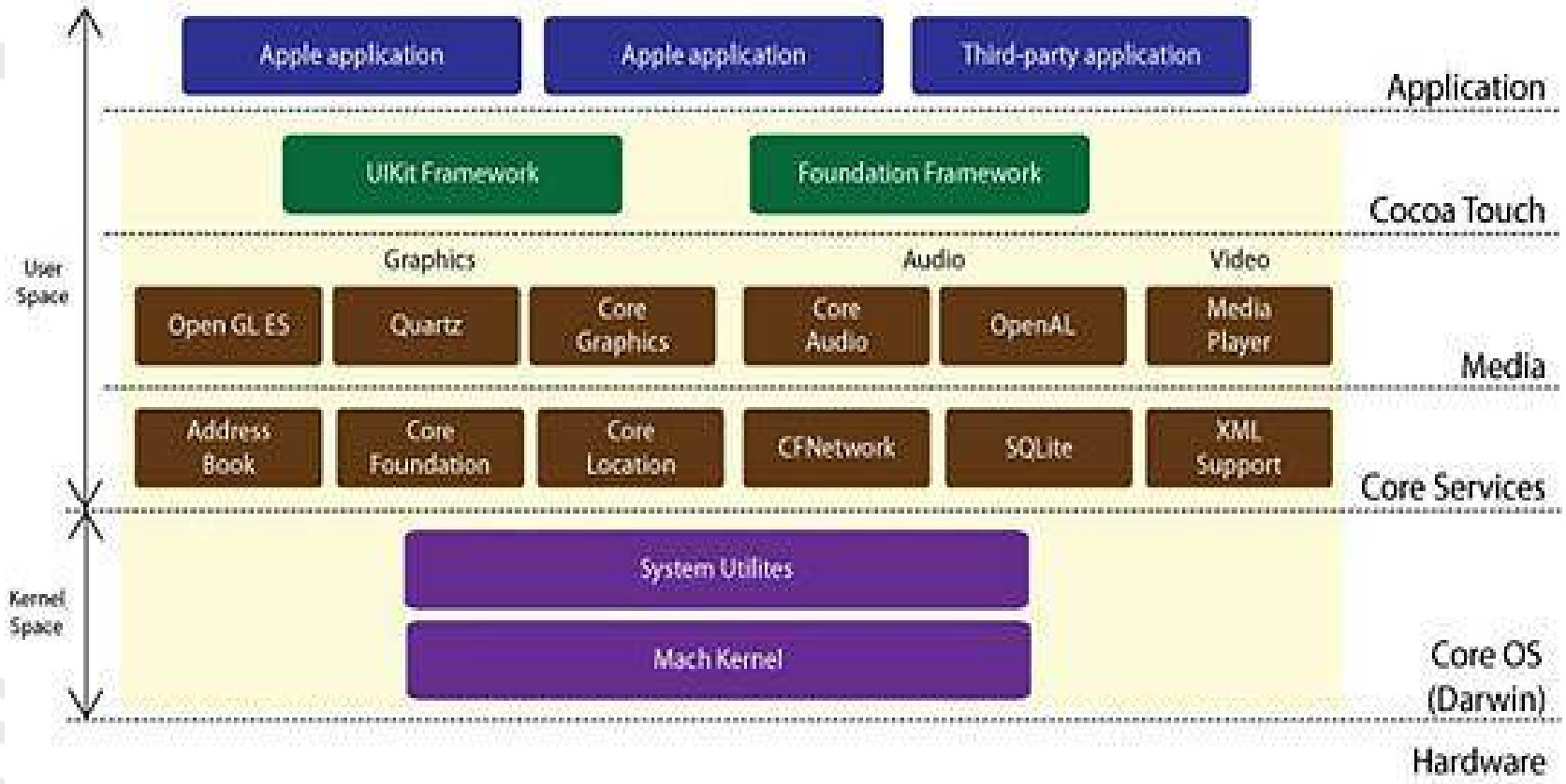
Modern operating systems for mobile devices

- In March 2008 - release a set of development libraries - ***iPhone SDK***;
- In June 2008, along with the release of the update to iOS 2.0, the *App Sto*re was launched;
- Providing the ability to create applications by independent developers and the provision of a centralized application distribution system has resulted in a huge market success;
- Within two months of the launch, applications from the store have been downloaded ***one hundred million*** times, and their number has increased almost ***six-fold*** from five hundred and fifty to over three thousand.

Modern operating systems for mobile devices

- To create an application for iOS: a computer with OS X, an installed Xcode programming environment;
- The programming languages are: Objective-C and Swift (presented in 2014);
- Frameworks have been prepared for the programmer, divided into layers:
  - Access to low-level operations in ***Core OS*** is blocked in the application - it is required to use libraries implementing particular functionalities of the operating system;
  - Libraries from the ***Core Services*** layer package basic system operations such as access to devices, files or network support;
  - Above is the ***Media Services*** layer, which is responsible for image and sound support in the application;
  - Layer of the highest in the hierarchy of abstraction is the ***Cocoa Touch*** framework - it supports, among others user interface, Push Notifications, and also contains classes to help you create games.

Modern operating systems for mobile devices



Modern operating systems for mobile devices

- Currently, iOS is the operating system for a whole range of portable devices from Apple:
  - iPod Touch,
  - iPhone smartphone,
  - iPad tablet,
  - Apple Watch smartwatch.
- iOS ranks second in terms of market share of the mobile devices market.

Modern operating systems for mobile devices

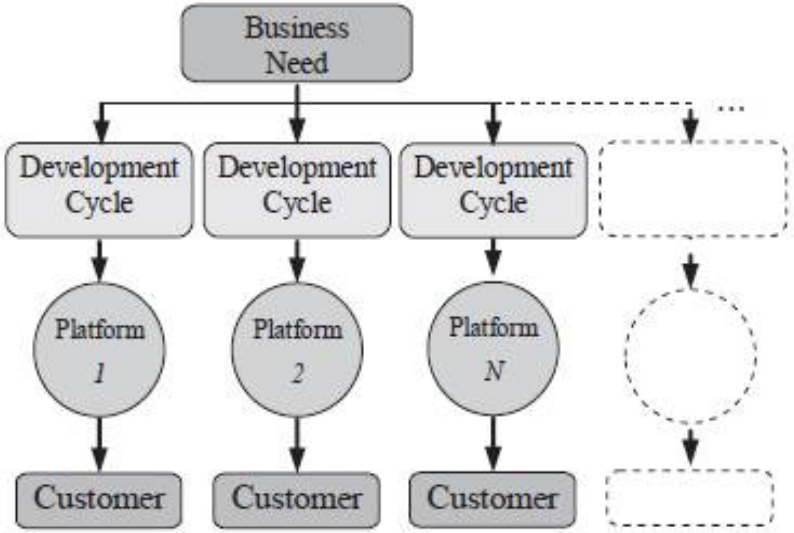
Windows

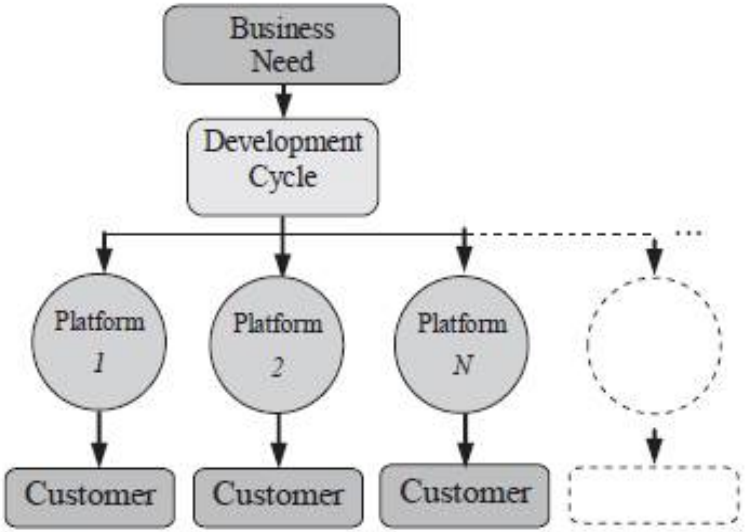
- The appearance of Windows Phone 7 was Microsoft's response to Apple's growing dominance in the mobile market;
- The system was based on previous versions of the Windows mobile system;
- The most characteristic feature of the new system was the user interface, which had little to do with solutions known from Android or iOS;
- Metro UI, later renamed to Modern UI, was characterized by a minimalist design;
- The start screen was a board of large colored tiles, which later could display additional information

Selected methods of creating mobile applications

- The goal: ***to provide the application to the largest possible groups of users***;
- Applications must be prepared separately for each platform:
  - due to the market structure of mobile systems;
  - the lack of compatibility between them.
- The programming teams must be divided into subteams responsible for creating individual versions of the program;
- This contributes to the increase in the number of developers developing the application, and this directly translates into an increase in project costs.

Selected methods of creating mobile applications





The model of producing a native application (on the left) and cross-platform (on the right)

Selected methods of creating mobile applications

- It is obvious that the business point of view, reducing costs and delivering the application in the shortest possible time is a priority in making decisions;
- To meet these needs, methods for generating ***multi-platform applications*** appeared ;
- Thanks to their use, the program is implemented once and can be run on many systems;
- This allows you to reduce costs by reducing production time and reducing the programming team.

Selected methods of creating mobile applications

- Typical problems of creating mobile applications:
  - ***limited resources***
    - mobile devices are becoming more efficient,
    - the screen resolution is growing,
    - maintain low weight and dimensions (thickness), the device has a significantly reduced size,
    - the process of creating a mobile application requires taking into account the optimization of energy use,
    - users are reluctant to use applications that significantly deplete the battery;
    - designing to a particular extent should include the planned operation of the application in the background, the method of communication with the server or limiting the processor's heavy tasks.

Selected methods of creating mobile applications

- Typical problems of creating mobile applications:
  - ***wireless comunication***
    - GSM telephone connections, connections to other devices via Bluetooth, the Internet, HSDPA, LTE and Wi-Fi network;
    - the implementation of the service of ***individual connections, depending on the requirements***, is just one problem;
    - wireless data transmission is characterized by interference, communication errors or large bandwidth differences;
    - therefore, a major challenge is to ***ensure the smooth operation of the application regardless of the type and quality of the connection***;
    - wireless networks are more susceptible to attacks - the medium is air, so potential eavesdroppers have easier access;
    - when designing applications that use sensitive data, ***special care should be taken to ensure encryption and authentication of connections***.

Selected methods of creating mobile applications

- Typical problems of creating mobile applications:
  - ***variety of devices***
    - there are many different mobile devices on the market;
    - Android and iOS support the entire range of size, aspect ratio and screen resolution (from four-inch phones from the lower shelf, to models with about seven-inch screens) with high pixel density, up to several-inch tablets, in which the resolution reaches up to the standard of 4K;
    - regardless of the type of device, the mobile application ***should provide the correct user experience***;
    - the set of internal devices of the phone or tablet is not standardized;
    - creating a mobile application using a specific sensor, it is also necessary to handle a scenario in which the sensor in the device is missing.

Selected methods of creating mobile applications

- Creating native mobile applications
  - native mobile application is created using the tools and programming languages provided by the platform for this purpose;
  - they can be run only using the system for which they were written;
  - applications have access to all functionalities that are available in the API and achieve the highest operational efficiency;
  - production of native software is expensive – a separate application must be created for each system platform;
  - each manufacturer of the system platform for mobile devices provides an integrated programming environment, programming languages and a set of libraries designed to develop applications;
  - each system platform has its own store distributing applications to users.

Selected methods of creating mobile applications

- Tools for creating native mobile applications:
  - ***Android studio***
    - is the official programming environment provided by Google as part of the Android SDK;
    - it is available free of charge, under the license of Apache License;
    - it can be installed on Linux, Windows and OS X;
    - The advanced emulator allows to test applications on different screen sizes and hardware configurations without the need for devices;
    - due to the long build times, the ***Instant run*** function has been introduced. It introduces changes to an already existing application, without the need to compile the whole project again;
    - it also includes tools for monitoring and analyzing the running application;
    - it is possible to check, live during operation, CPU load, memory usage.

Selected methods of creating mobile applications

- Tools for creating native mobile applications:
  - ***Xcode***
    - is an integrated development environment developed by Apple;
    - it is intended for the production of applications for iOS and OS X;
    - it is provided free of charge via the Mac Apple Store;
    - a development package for iOS is provided with Xcode (simulators of Apple devices, class libraries used in applications, IDE has a lot of tools to help programmers work);
    - Interface Builder allows you to create a drag and drop user interface;
    - instruments is a set of advanced programs for monitoring, analysis and visualization of the application's operation;
    - it allow for measuring the parameters of the device as the use of the processor, power consumption;
    - allow you to test memory leaks or problems with thread synchronization.

Selected methods of creating mobile applications

- Tools for creating native mobile applications:
  - ***Visual Studio***
    - is an advanced Microsoft programming environment – Visual Studio 2017;
    - thee software is available for Windows, for Mac OS (App Center) and for Windows, Mac OS and Linux (Visual Studio Code);
    - The environment supports a wide spectrum of technologies, thanks to which it is possible to create software of any destination with it (programming embedded devices, mobile and desktop applications, internet services, to multi-layer systems and data warehouses);
    - it provides support for creating native mobile applications for Windows Phone, as well as multi-platform applications using ***Xamarin***;
    - it has advanced tools for debugging, testing and running applications;
    - built-in integration with Team Foundation Server provides access to tracking tools, task separation, version control and continuous integration directly in the environment.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - the mobile cross-platform application is designed to run on many mobile systems;
  - these types of applications are usually created using technologies that provide an additional layer of abstraction over the system API, uniform for each platform;
  - two types of frameworks can be distinguished on the market - launching ***an application in a system browser*** or compiling ***a cross-platform code*** for a native solution;
  - the biggest disadvantage of the first solution is low performance - applications are created in HTML5 and JavaScript, which are interpreted on the device;
  - access to the functionality provided in the system API is possible through specially prepared references on the JavaScript page;
  - applications run in the browser often have problems with ensuring the same good user interface experience as native solutions.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Apache Cordova (PhoneGap)***
    - is a technology for creating multi-platform mobile applications, made available as open-source;
    - it was decided to pass the code to the Apache Software Foundation in order to supervise the proper development of the project;
    - applications are created using HTML5, CSS and JavaScript;
    - the technology allows you to invoke system functions through a shared JavaScript API;
    - a native application is generated that contains only the browser component in which the application created in Apache Cordova is run;
    - the disadvantage of the solution is ***a noticeable difference in the speed of the application***;
    - the appearance of the user interface depends on the quality of HTML rendering by the system browser.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Corona***
    - is a set of developer tools designed to create mainly games for mobile systems;
    - applications created using the framework are written in Lua;
    - they are compiled to the code that runs natively on the device;
    - supports a large number of operating systems, among others iOS, Android, Windows Phone, tvOS, Amazon Kindle;
    - is available for free for use in applications;
    - it allows you to create applications using the shared abstraction layer above the system API;
    - the included simulator provides the ability to test without the use of devices;
    - the technology possibilities can be expanded by the plug-in available on the producer's website. Paid subscriptions provide the ability to use native libraries and compile without access to the Internet.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Marmalade***
    - is a collection of libraries and tools for programming multi-platform games and mobile applications;
    - the ability to create applications using HTML and JavaScript that will run the browser;
    - the second approach is to create games using an efficient 2D and 3D engine using C ++;
    - games created in this way are run natively in the system;
    - it supports the creation of applications for Android, iOS, BlackBerry, Tizen, Windows and Windows Phone;
    - games: Cut the Rope, Doodle Jump, Peggle, Plants Vs. Zombies;
    - the development kit includes a simulator for easy testing. A paid license also allows you to run and debug applications on the device;
    - Marmalade uses Visual Studio on Windows and Xcode on OS X.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Xamarin Platform (1)***
    - is a solution that allows creating cross-platform native applications using C#;
    - it has been developed by the creators of Mono, Mono Touch and Mono for Android - implementation of .NET Framework for Linux, iOS and Android, on which the whole environment is based;
    - Xamarin supports the production of applications for Android, iOS and Windows Phone;
    - In February 2016, Microsoft took over Xamarin and opened the source of the entire platform;
    - technology offers two ways to create cross-platform applications: ***Xamarin Native*** and ***Xamarin Forms.***

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Xamarin Platform (2)***
    - ***Xamarin Native*** is an approach that uses Android and iOS SDK that have been mapped in C# classes;
    - the developers boast about an almost one hundred percent coverage of the Android and iOS APIs;
    - thanks to this, everything in the native application can be implemented in Xamarin;
    - in this approach, the user interface is created separately for each platform, using the solution appropriate for each system.

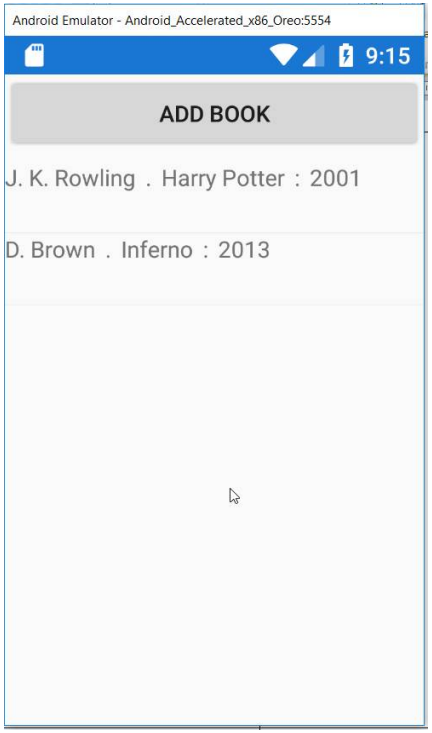
Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Xamarin Platform (3)***
    - ***Xamarin Forms –*** a library providing a programming interface independent of the system platform;
    - it is possible to achieve the maximum code sharing between applications;
    - the application interface is created, once for all platforms, in XAML language, using the controls provided in the library;
    - during startup, they are mapped to native components on a given system platform. This allows you to achieve native performance and quality of the user interface.

Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Xamarin Platform (4) - Xamarin Forms***

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:Book"
             x:Class="Book.MainPage">

    <StackLayout>
        <!-- Place new controls here -->
        <Button Text="Add book" Clicked="Button_Clicked"/>
        <ListView x:Name="bookList" ItemSelected="BookList_ItemSelected">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <StackLayout Orientation="Horizontal">

                            <Label Text="{Binding firstAuthor}"/>
                            <Label Text="."/>
                            <Label Text="{Binding title}"/>
                            <Label Text=":"/>
                            <Label Text="{Binding publishYear}"/>

                        </StackLayout>
                    </ViewCell>
                </DataTemplate>

            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>

</ContentPage>
```



Selected methods of creating mobile applications

- **Tools for creating cross-platform mobile applications**
  - ***Xamarin Platform (5)***
    - preparing the interface in Xamarin Native requires more time but also gives you full control over the structure of the views;
    - Xamarin Forms allows for much more sharing of code between platforms;
    - the developer recommends using the Native approach in applications that have complex interfaces and require the use of many functionalities related to the system API;
    - Xamarin Forms is proposed for use in situations when the time of product development and code sharing is more important than the extended interface.

Selected methods of creating mobile applications

- **Advantages and disadvantages of creating multi-platform applications (1)**
  - is ***shortening production time*** (faster delivery of applications to stores, and thus to users);
  - shortening the production time contributes significantly to ***reducing the costs*** of project implementation;
  - reduction in the number of teams also leads to ***savings*** (one team of programmers is enough);
  - one application means one set of technologies, tools and additional libraries used during the software development process. This ***simplifies change management*** at later stages of the application lifecycle;

Selected methods of creating mobile applications

- **Advantages and disadvantages of creating multi-platform applications (2)**
  - sharing code between applications makes it easier to provide ***the same functionality on all supported platforms***;
  - the app update should be done only once;
  - adding a supported platform, converting a hybrid application is much faster and easier than creating a native solution from scratch;
  - depending on the knowledge and experience of the team, implementation in multi-platform technology can be quite efficient (if programmers know .NET, it will be easier for them to get to know Xamarin than to learn Objective-C and iOS SDK);

Selected methods of creating mobile applications

- **Advantages and disadvantages of creating multi-platform applications (3)**
  - is ***lower efficiency*** compared to native methods (in the case of technologies based on launching applications in the browser, this problem is particularly noticeable);
  - cross-platform applications ***run longer*** due to loading of the runtime environment and numerous libraries at the start;
  - technologies that enable the creation of hybrid applications *provide their* ***own interface to wrap the functionality offered by the system***;
  - due to the differences in operating systems for mobile devices, often the API offers a set of functionalities being the common denominator of all supported platforms;

Selected methods of creating mobile applications

- ***Advantages and disadvantages of creating multi-platform applications (4)***
  - to achieve performance and aesthetics of the ***user interface similar to a native application,*** it may be necessary to adjust some elements especially for a given platform;
  - as with any new solution, developers must have ***time to become familiar with the technology*** used or ***training*** may be needed;
  - using hybrid methods, when creating an application, introduces dependence on the external library. This can cause ***difficulties with support or errors;***
  - you should also be aware that creating a cross-platform application uses technologies and languages not provided by the system manufacturer. Software created in this way may have ***unpredictable or difficult to detect errors***.

Analysis of the performance of native and hybrid applications

- In order to compare the performance of mobile applications created ***natively*** and ***hybridly***, a study was conducted to measure the time of execution of selected tasks;
- Android and iOS systems were selected for experiments because of the largest market share (altogether, both systems are installed on 95% of mobile devices in the world);
- The cross-platform application was created in the ***Xamarin*** environment.

Analysis of the performance of native and hybrid applications

- ***Methods***
  - the study was divided into four parts, measuring the task execution time;
  - this is the parameter that has the greatest impact on the user's feelings from using the mobile application;
  - each of the experiments concerns a different area of using the system's API:
    - numerical calculations,
    - access to the file,
    - handling of network connections,
    - use of the internal devices of the telephone - in this case GPS.

Analysis of the performance of native and hybrid applications

- ***Method***
  - six test applications were created for the purposes of the study - three for each system: native, multi-platform using the Xamarin Native approach and multi-platform using Xamarin.Forms libraries;
  - due to differences in the system architecture and performance of cell components, the results obtained in the study will be compared only within the system platform.

Analysis of the performance of native and hybrid applications

- **Method**
  - in order to ensure the reliability of results, each test was carried out with care for repeatability of conditions;
  - applications have been launched on the same smartphones (Android: LG G4, and iOS iPhone: 6 16GB);
  - in order to use all the functions of the system, the applications were tested on system versions (Android: 6.0 Marshmallow, iOS: 9.3.2.);
  - the applications were built with the release setting and installed on the phone;
  - the startup was done without a debugger connected;
  - each test was performed ***20 times*** for each system and each application;
  - tests of using network connections have been performed on the same internet connection;
  - location tests were made from the same position.

Analysis of the performance of native and hybrid applications

- ***Numerical calculations - Android***
  - the time was calculated by calculating the number π to ten thousand decimal places

Test No.		1	2	3	4	5	6	7	8	9	10
Time [s]	Native	4.500	4.605	4.553	4.507	4.662	4.468	4.473	4.507	4.519	4.544
	Xamarin Native	21.358	20.968	21.536	21.546	21.182	22.309	22.624	22.492	22.270	22.254
	Xamarin Forms	21.285	21.412	21.759	21.474	21.149	21.751	21.584	18.905	19.236	19.472

Test No.		11	12	13	14	15	16	17	18	19	20
Time [s]	Native	4.490	4.443	4.496	4.468	4.492	4.576	4.527	4.496	4.420	4.551
	Xamarin Native	22.503	22.585	23.065	21.359	23.102	22.560	23.636	22.998	22.926	23.031
	Xamarin Forms	19.806	18.972	19.711	22.125	20.236	22.247	21.036	21.337	21.561	21.014

Analysis of the performance of native and hybrid applications

- ***Numerical calculations - iOS***
  - the time was calculated by calculating the number π to ten thousand decimal places

Test No.		1	2	3	4	5	6	7	8	9	10
Time [s]	Native	3.286	3.220	3.224	3.213	3.217	3.219	3.220	3.209	3.208	3.205
	Xamarin Native	5.118	5.121	5.107	5.120	5.123	5.115	5.112	5.133	5.117	5.098
	Xamarin Forms	5.132	5.160	5.099	5.101	5.120	5.114	5.112	5.118	5.143	5.124

Test No.		11	12	13	14	15	16	17	18	19	20
Time [s]	Native	3.213	3.204	3.208	3.213	3.200	3.207	3.204	3.205	3.207	3.212
	Xamarin Native	5.116	5.112	5.111	5.110	5.153	5.102	5.111	5.118	5.112	5.115
	Xamarin Forms	5.120	5.117	5.143	5.118	5.109	5.119	5.117	5.124	5.113	5.126

Analysis of the performance of native and hybrid applications

- For both systems, native applications achieved the best results;
- The results achieved by programs written in Xamarin in the case of Android proved to be several times slower than native ones;
- For iOS, the difference was not that big - the hybrid application was about 40% slower than the native one;
- Applications Xamarin Native was slower than Xamarin.Forms on Android by about 15%. In the case of iOS, both Xamarin applications achieved slightly different results.

Analysis of the performance of native and hybrid applications

- ***Writing and reading from a file***
  - the time of writing to the text file, the result of the calculation of ten thousand places after the comma π and the reading of this file was measured;
  - the data to write to the file has been calculated once and kept in memory;
  - the calculation time is not included in the experiment result.

Analysis of the performance of native and hybrid applications

- ***Reading from a file - Android***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [ms]	Native	24.121	27.822	27.016	23.798	26.764	28.115	22.242	26.051	27.232	24.003
	Xamarin Native	20.893	4.082	3.723	4.880	4.728	4.866	5.355	5.330	3.596	3.566
	Xamarin Forms	53.177	7.604	6.120	4.180	4.132	4.272	4.431	3.782	7.164	14.370

Test No.		11	12	13	14	15	16	17	18	19	20
Time [ms]	Native	23.653	29.146	24.164	31.021	27.115	29.488	26.376	26.178	23.938	31.797
	Xamarin Native	4.051	3.914	4.095	4.072	2.734	4.494	5.556	5.060	4.480	2.810
	Xamarin Forms	5.857	6.492	3.144	15.787	3.464	3.874	4.242	4.332	4.965	3.730

Analysis of the performance of native and hybrid applications

- ***Reading from a file - iOS***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [ms]	Native	0.712	0.802	0.698	0.705	0.702	0.699	0.693	0.711	0.703	0.725
	Xamarin Native	2.139	1.154	1.141	1.410	1.424	1.415	1.377	1.368	1.417	1.598
	Xamarin Forms	23.689	4.856	2.697	4.681	3.066	5.854	4.226	6.144	3.957	4.494

Test		11	12	13	14	15	16	17	18	19	20
Time [ms]	Native	0.691	0.694	0.701	0.495	0.695	0.893	0.696	0.702	0.807	0.747
	Xamarin Native	1.411	1.455	1.414	1.953	1.424	1.644	1.589	1.417	1.427	1.498
	Xamarin Forms	5.084	3.651	3.632	4.274	7.695	6.056	5.876	4.087	2.808	5.456

Analysis of the performance of native and hybrid applications

- ***Writing to a file - Android***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [ms]	Native	1.950	2.258	1.360	2.811	2.845	2.568	2.134	1.935	3.018	3.337
	Xamarin Native	32.786	3.317	13.114	4.876	5.502	4.271	4.205	5.535	5.447	5.103
	Xamarin Forms	17.284	28.862	13.677	12.894	6.573	8.500	11.478	8.855	12.138	10.252

Test No.		11	12	13	14	15	16	17	18	19	20
Time [ms]	Native	5.184	2.145	3.158	3.033	3.368	2.919	3.187	1.889	3.541	1.803
	Xamarin Native	5.606	5.108	4.077	3.240	6.192	5.884	5.316	5.205	6.552	3.693
	Xamarin Forms	8.290	6.054	7.721	6.009	6.956	12.145	7.966	5.610	8.090	6.832

Analysis of the performance of native and hybrid applications

- ***Writing to a file - iOS***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [ms]	Native	25.952	15.552	14.214	14.271	14.831	15.471	15.429	16.621	12.576	16.187
	Xamarin Native	18.550	16.420	20.570	16.232	17.154	18.105	16.725	17.920	18.673	18.601
	Xamarin Forms	62.131	24.759	19.930	19.014	19.749	19.147	18.161	19.893	17.393	17.551

Test No.		11	12	13	14	15	16	17	18	19	20
Time [ms]	Native	20.722	17.198	17.763	16.623	12.781	19.894	16.727	20.479	18.463	17.146
	Xamarin Native	19.004	18.167	19.678	17.393	16.668	18.118	19.896	22.034	20.853	18.748
	Xamarin Forms	20.979	17.981	20.102	16.321	18.834	17.928	19.565	17.748	17.561	20.163

Analysis of the performance of native and hybrid applications

- ***Reading a file***
  - the Xamarin Native app got the best results on Android;
  - the slowest application turned out to be a native application - the file read time was more than five times slower than in the hybrid approach;
  - the native application was the fastest in iOS;
  - the longest reading of the file was carried out by Xamarin.Forms.
- ***Writing to a file***
  - native applications proved to be the fastest for both systems;
  - the slowest is Xamarin.Forms;
  - differences between the results of individual platforms were greater on Android than on iOS;

Analysis of the performance of native and hybrid applications

- ***Network support***
  - during the research, the time of downloading a large sized image file - about 7 MB - from the Internet was measured;
  - in the examination of each application, the same file placed on a publicly available server was used;
  - the smartphones were connected to the same internet connection via a Wi-Fi network.

Analysis of the performance of native and hybrid applications

- ***Network support - Android***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [s]	Native	6.834	6.875	6.811	6.667	6.949	6.844	6.945	6.840	6.803	6.864
	Xamarin Native	7.039	6.980	6.811	6.385	6.829	6.788	6.890	6.334	6.756	6.720
	Xamarin Forms	6.944	6.983	6.570	6.885	6.819	6.789	6.935	6.937	6.736	8.120

Test No.		11	12	13	14	15	16	17	18	19	20
Time [s]	Native	6.787	6.864	6.922	6.838	6.928	6.902	6.818	6.869	6.896	6.809
	Xamarin Native	7.003	6.252	6.729	6.284	6.718	6.772	6.282	6.410	6.657	6.702
	Xamarin Forms	6.709	6.884	7.034	6.627	6.689	6.691	6.948	6.586	6.871	10.018

Analysis of the performance of native and hybrid applications

- ***Network support - iOS***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [s]	Native	7.157	6.894	6.901	6.875	6.847	6.306	6.829	6.291	6.918	6.839
	Xamarin Native	6.391	6.325	6.391	6.328	6.358	6.400	6.988	6.875	6.807	6.865
	Xamarin Forms	7.096	6.419	6.389	6.852	6.502	6.596	7.049	6.969	7.123	6.681

Test No.		11	12	13	14	15	16	17	18	19	20
Time [s]	Native	6.868	6.901	6.789	6.829	6.347	6.856	6.493	6.303	6.322	6.960
	Xamarin Native	6.487	6.476	6.332	6.363	6.268	6.424	6.378	6.364	6.370	6.339
	Xamarin Forms	6.916	7.006	6.929	6.922	6.881	6.686	6.888	6.717	6.928	7.043

Analysis of the performance of native and hybrid applications

- ***Network support***
  - for both systems, the slowest applications were Xamarin.Forms;
  - the best results have been achieved by the Xamarin Native application;
  - it should be noted that the differences between the various types of applications are so small that they can be explained by temporary differences in bandwidth or other unforeseen factors, such as system tasks run in the background.

Analysis of the performance of native and hybrid applications

- ***Designation of the location***
  - the time of determining latitude and longitude was measured using sensors built into the phone;
  - in modern mobile devices the location can be determined using the GPS system using the location of BTS transmitters or based on available Wi-Fi networks;
  - due to the lack of choice of the location method in iOS, the test on both systems was carried out using the default system settings.

Analysis of the performance of native and hybrid applications

- ***Designation of the location - Android***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [s]	Native	2.530	1.956	3.372	5.694	3.577	9.632	2.154	2.775	2.965	3.019
	Xamarin Native	2.895	12.068	6.195	9.956	8.502	15.896	8.171	10.483	12.052	14.215
	Xamarin Forms	5.489	1.563	5.163	3.432	1.953	5.332	3.319	4.025	7.446	6.882

Test No.		11	12	13	14	15	16	17	18	19	20
Time [s]	Native	2.588	1.983	2.267	3.678	4.112	1.931	2.361	3.334	2.098	1.884
	Xamarin Native	12.006	8.856	2.706	2.994	15.914	9.556	11.256	13.554	8.561	6.511
	Xamarin Forms	3.129	5.951	7.953	8.802	4.653	2.287	3.336	2.952	4.748	3.372

Analysis of the performance of native and hybrid applications

- ***Designation of the location - iOS***

Test No.		1	2	3	4	5	6	7	8	9	10
Time [ms]	Native	9.926	10.343	4.691	15.997	17.808	9.897	6.073	9.202	16.378	11.324
	Xamarin Native	8.609	10.756	18.539	17.586	8.347	16.826	17.491	12.824	16.662	12.938
	Xamarin Forms	45.518	43.023	39.980	49.450	47.458	53.557	48.761	58.869	53.061	40.239

Test No.		11	12	13	14	15	16	17	18	19	20
Time [ms]	Native	34.035	9.691	16.054	6.046	6.235	9.240	11.340	10.363	13.026	14.508
	Xamarin Native	7.808	12.296	11.933	17.175	9.497	21.082	9.234	22.065	17.543	18.042
	Xamarin Forms	43.982	59.687	59.716	61.052	64.619	68.169	66.859	46.528	36.953	55.231

Analysis of the performance of native and hybrid applications

- ***Designation of the location***
  - the most striking difference is the size of the results of the study on Android and iOS;
  - in the first system, determining the location took an average of 3 to 9 seconds;
  - in contrast, in iOS it took on average from 12 to 52 milliseconds;
  - as in previous tests, on both systems, the native application works the best with the task;
  - the Xamarin Native app on Android was the longest position and on iOS Xamarin.Forms;
  - the difference between Xamarin Native and the native application was very small - on average about 2 milliseconds longer than native ones needed a cross-platform solution.

Thank You.

maria.paszkowska@pollub.pl
e.lukasik@pollub.pl