Facultatea de Electronică, Telecomunicații și Tehnologia Informației

Dezvoltarea aplicatiilor pe platforma Android

Coordonator: Studenți:

Conf. dr. ing. Ștefan Stăncescu Ionita Vlad-Cristian

Singureanu Liviu-Cristian

Grupa 433 A

**Cuprins**

1.Introducere ………………………………………………………………………3

1.1 Scurta introducere ………………………………………………………3

1.2 Istoric ……………………………………………………………………4

1.3 Caracteristici …………………………………………………………….5

2.Design ……………………………………………………………………………6

2.1 Scurta introducere ………………………………………………………6

2.2 Istoric ……………………………………………………………………7

2.3 Structura ………………………………………………………………...8

2.4 Sintaxa…………………………………………………………………...9

3.Instrumente pentru dezvoltatori …………………………………………….......10

4.Biblioteci support ………………………………………………………………11

5.Dezvoltarea aplicatiilor ……………………………………………………...…13

6.Distributia aplicatiilor ………………………………………………………….14

7. Impartirea Temei …………….………………………………………………...15

8. Bibliografie ……………………………………………………………………16

1. **Introducere**

**1.1 Scurta introducere**

Sistemul de operare Android furnizeaza un cadru de lucru bogat ceea ce permite programatorului dezvoltarea de aplicatii, jocuri pentru dispositive mobile(celulare, tablete) intr-un mediu de dezvoltare ce foloseste limbajul orientat pe obiecte, Java. [1]

Aplicatiile Android sunt construite ca fiind o combinatie de component distincte ce pot fi accesate individual si unul dintre cele mai importante lucruri este ca pot fi portabile, caracteristica a limbajului Java. Aceste aplicatii Android se pot adapta la orice fel de device-uri care suporta masina virtuala java, de exemplu de la un simplu smartphone cu diagonal mica pana la o tableta Android care are diagonal destul de mare pentru un device portabil, sau chiar Smart TV. Cu ajutorul limbajului orientat pe obiecte Java, o aplicatie Android se poate conecta la o baza de data externa daca dispozitivul Android are conexiune la internet. [1]

Unul dintre avantajele sistemului de operare Android este acela ca nu prezinta constrangeri in legatura multitudinea si diversitatea aplicatiilor ce pot fi rulate pe dispozitiv. [1]

Dupa cum a fost mentionat mai sus, portabilitatea este un punct forte al sistemului de operare, datorita limbajului Java. Astfel dupa creeare aplicatiei, aceasta poate fi instalata si rulata pe o multitudine de dispositive Android.[1]

O asemanare intre sistemul de operare IOS si Android este acela ca ambele folosesc un limbaj de programare orientat pe obiecte. Diferenta dintre cele doua sisteme de operare este data de limbajul de programare folosit: IOS a fost creat folosind Object C. Aceste nu detine o portabilitate la fel de mare precum Java dar este mai performant din puncutul de vedere al optimizarii, in sensul derularii mai eficinente si mai rapide a aplicatiilor. O alta diferenta o constituie faptul ca majoritatea aplicatiilor dezvoltate folosind platforma Android sunt de tip open source, contrar celor dedicate sistemului de operare IOS.[1]

**1.2 Istoric**

Sistemul de operare Android a aparut de mai bine de sase ani. In acest timp se poate observa evolutia sistemelor de operare. Primul sistem de operare Android nu a avut un nume propriu, acesta fiind cunoscut simplu ca 1.0. Dupa un timp insa, producatorii au inceput sa le denumeasca, folosind nume de prajituri si incepand cu versiunea 1.5 aparuta la data de 30 aprilie 2009. Aceasta versiune s-a numit ‘*Cupcake*’. Versiunile sistemelor de operare au continuat sa avanseze, ultima versiune a sistemului de operare Android este 6.0 denumit ‘Marshmallow’. In tabelul urmator sunt prezentate versiunile Android din 30 aprilie 2009 pana in prezent. [2]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versiune | Nume de cod | Data lansarii | API level | Distributie |
| **6.0** | *Marshmallow* | prezent | 23 | 0.7% |
| **5.0** | *Lollipop* | Google I/O 2014 | 21 | 9% |
| **4.4** | *KitKat* | 31 Octombrie 2013 | 19 | 39.8% |
| **4.3** | *Jelly Bean* | 24 Iulie 2013 | 18 | 5.5% |
| **4.2.x** | *Jelly Bean* | 13 noiembrie 2012 | 17 | 18.1% |
| [**4.1.x**](http://ro.wikipedia.org/w/index.php?title=Jelly_Bean_%28sistem_de_operare%29&action=edit&redlink=1) | [*Jelly Bean*](http://ro.wikipedia.org/w/index.php?title=Jelly_Bean_%28sistem_de_operare%29&action=edit&redlink=1) | 9 iulie 2012 | 16 | 36.6% |
| [**4.0.3–4.0.4**](http://ro.wikipedia.org/w/index.php?title=Ice_Cream_Sandwich_%28sistem_de_operare%29&action=edit&redlink=1) | *Ice Cream Sandwich* | 16 decembrie 2011 | 15 | 5.3% |
| [**3.2**](http://ro.wikipedia.org/w/index.php?title=Honeycomb_%28sistem_de_operare%29&action=edit&redlink=1) | [*Honeycomb*](http://ro.wikipedia.org/w/index.php?title=Honeycomb_%28sistem_de_operare%29&action=edit&redlink=1) | 15 iulie 2011 | 13 | 0.1% |
| [**3.1**](http://ro.wikipedia.org/w/index.php?title=Honeycomb_%28sistem_de_operare%29&action=edit&redlink=1) | [*Honeycomb*](http://ro.wikipedia.org/w/index.php?title=Honeycomb_%28sistem_de_operare%29&action=edit&redlink=1) | |  |  | | --- | --- | | 10 mai 2011 |  | | 12 | 0% |
| [**2.3–2.3.2**](http://ro.wikipedia.org/w/index.php?title=Android_2.3&action=edit&redlink=1) | [*Gingerbread*](http://ro.wikipedia.org/w/index.php?title=Android_2.3&action=edit&redlink=1) | |  |  | | --- | --- | | 6 decembrie 2010 |  | | 9 | 5.7% |
| [**2.3.3–2.3.7**](http://ro.wikipedia.org/w/index.php?title=Android_2.3&action=edit&redlink=1) | [*Gingerbread*](http://ro.wikipedia.org/w/index.php?title=Android_2.3&action=edit&redlink=1) | |  |  | | --- | --- | | 9 februarie 2011 |  | | |  |  | | --- | --- | | 10 |  | | 5.7% |
| [**2.2**](http://ro.wikipedia.org/w/index.php?title=Android_Froyo&action=edit&redlink=1) | [*Froyo*](http://ro.wikipedia.org/w/index.php?title=Android_Froyo&action=edit&redlink=1) | 20 mai 2010 | 8 | 2.4% |
| [**2.0–2.1**](http://ro.wikipedia.org/w/index.php?title=Android_version_history&action=edit&redlink=1) | [*Eclair*](http://ro.wikipedia.org/w/index.php?title=%C3%89clair_%28operating_system%29&action=edit&redlink=1) | 26 octombrie 2009 | 7 | 0.3% |
| [**1.6**](http://ro.wikipedia.org/w/index.php?title=Donut_%28operating_system%29&action=edit&redlink=1) | [*Donut*](http://ro.wikipedia.org/w/index.php?title=Donut_%28operating_system%29&action=edit&redlink=1) | |  |  | | --- | --- | | 15 septembrie 2009 |  | | 4 | 0% |
| [**1.5**](http://ro.wikipedia.org/w/index.php?title=Cupcake_%28operating_system%29&action=edit&redlink=1) | [*Cupcake*](http://ro.wikipedia.org/w/index.php?title=Cupcake_%28operating_system%29&action=edit&redlink=1) | |  |  | | --- | --- | | 30 aprilie 2009 |  | | |  |  | | --- | --- | | 3 |  | | 0% |

Tabel 1. Versiuni sisteme de operare Android [1]

API Level – nivelul la care un programator poate realiza o comunicare cu functiile device-ului. Cu cat nivelul API este mai mare, cu atat functionalitatea creste.[3]

**1.3 Caracteristici**

* Stocarea datelor este realizata folosind software-ul de stocare SQLite.
* Suporta tehnologii de conectivitate incluzand GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth si Wi-Fi.
* Navigatorul de web disponibil este bazat pe platforma de aplicatii open source WebKit.
* Software-ul care este scris in Java poate fi compilat in cod masina Dalvik si se executata de masina virtuala Dalvik, aceasta este o implementare specializata de masina virtuala conceputa pentru utilizarea in dispozitivele mobile,desi, teoretic nu este o masina virtuala Java.
* Suporta urmatoarele formate media audio/video/imagine: MPEG-4,H.264,MP3,AAC,OGG,AMR,JPEG,PNG,GIF.
* Utilizeaza un support hardware aditional cum ar fi camera video/foto, touchscreen, GPS, accelerometru si grafica accelerate 3D.
* Piata Android (Android Market) este similar cu App Store-ul de pe sistemul de operare IOS, la inceput aplicatiile au beneficiat de gratuite, dar incepand cu anul 2009 unele aplicatii nu mai sunt gratuite.
* Mediu de dezvoltare include un emulator de dispozitive si unelte de depanare, plug-inuri pentru mediile de dezvoltare cum ar fi Eclipse sau Android Studio.
* Sistemul de operare beneficiaza de optiunea multi-touch. [4]

**2.Design**

**2.1 Scurta introducere**

Designul sistemului de operare Android este realizat in limbajul de marcare XML (Extensible Markup Language). Aceste este un meta-limbaj de marcare pentru crearea de alte limbaje de marcare, cum ar fi XHTML, RDF, RSS, MathML, SVG, OWL etc, care formeaza familia de limbaje XML.

Este o simplificare a limbajului SGML(de unde provine si HTML) si a fost dezvoltat in scopul transferului de date intre aplicatii pe internet. Recent a devenit si un model de stocare a datelor nestructurate si semi-structurate, in cadrul bazelor de date specific XML.

Acesta poate fi folosit chiar si in creearea documentelor de tip formular, care pot avea semnatura digitala, folosite des de contabili si pentru usurarea muncii de birotica.

Datele pot fi de asemenea utilizate si folosind limbajul HTML si permit o identificare rapida a documentelor, utilizand motoare de cautare. Folosind codurile javascript, php etc, fisierele XML pot si incluse in paginile web, cel mai concludent exemplu fiind sistemul RSS, ce foloseste un fisier XML pentru transportul informatiei de la o pagina web catre mai multe.

Limbajul de meta-marcare XML are anumite avantaje cum ar fi :

* Extensibilitate
* Valididate
* Editabilitate si modificabilitate
* Simplitate si accesibilitate [5]

**2.2 Istoric**

Limbajul de meta-marcare XML este descendent al SGML (Standard Generalized Markup Language), fiind un limbaj folosit in activitatea de marcare structural a documentelor. Specificatia sa a fost dezvoltata incepand cu 1996, in cadrul Cosortiului World Wide Web (W3C). Prima versiune de XML a fost standardizata in februarie 1998, ulterior acest standard fiind revizuit de doua ori in octombrie 2000 si respective in februarie 2004.

Principalele scopuri din spatele proiectarii XML au fost urmatoarele:

* Sinplitatea utilizarii pe Internet.
* Capacitatea de a suporta o gama larga de aplicatii.
* Combatibilitatea cu SGML.
* Usurinta scrierii de programe ce vor accesa documente XML.
* Reducerea numarului facilitatilor optionale din XML la minimum.
* Claritatea cat mai mare a documentelor XML, putand fi citite usor de catre utilizatori.
* Design-ul sa fie formal si concis.
* Redarea usoara a documentelor. [5]

**2.3 Structura**

Structura unui document XML este formata din doua elemente : marcaje(tag-uri) si date character.[5]

Marcajele au fost folosite initial pentru a descrie anumite adnotari (note marginale in cadrul unui text cu intentia de a indica tehnoredactorului cum trebuie listat un anumit pasaj). Marcajul poate fi definit ca orice actiune de interpretare explicita a unei portiuni de text. Aceste reprezinta un sir de caractere delimitat de “<” si “>”. Datele caracter reprezinta continutul marcajelor.[5]

In limbajul XML, marcajele nu sunt folosite pentru afisarea datelor, ci pentru : - Asigurarea unei sintaxe simple si standardizate ce poate fi folosita de analizoarele XML pentru a utiliza informatia stocata.

* Asigurarea unei metode de a descrie structura ierarhica a continutului prin impartirea informatiei in parti numite *elemente,* descries prin *atribute*. [5]

Un fisier XML este structurat astfel:

* Prolog
* Definitia tipului de document
* Elemental radacina

Exemplu:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE MAIL SYSTEM "MAIL.DTD">

<MAILid="10" date="14-05-2008">

<FROM>Exemplu@ex.ro</FROM>

<TO>Exemplu2@ex2.ro</TO>

<SUBJECT>TEST</SUBJECT>

<BODY>

Test

</BODY>

</MAIL> [5]

In codu de mai sus prologul este sintaxa ‘<?xml version="1.0" encoding="UTF-8"?>’. Aceasta ne specifica versiunea de XML folosita si tipul de codare.

Definiti tipului de document se face in sintaxa urmatoare ‘<!DOCTYPE MAIL SYSTEM "MAIL.DTD">’. In aceasta sintaxa se defineste un set de reguli.

Elementul radacina este dat de restul documentului.[5]

**2.4 Sintaxa[5]**

Un document XML poate contine urmatoarele tipuri de marcaje:

1. **Elemente**: Blocurile de baza ale unui document XML. Se pot folosi in scopul retinerii informatiei si definirea structurii.
2. **Atribute**: Au rolul de a descrie elementele. Sunt identice cu atributele din HTML. Sunt localizate in tag-ul de start al unui element, langa numele elementului si sunt urmate de semnul ‘=’, urmat de valoarea atributului, intre ghilimele.
3. **Comentarii**: Sunt secvente de caractere scrise de programator cu scopul de a descrie detalii legate de continutul si structura documentului. Nu sunt date caracter.
4. **Referinte la entitati**: sunt pointeri catre entitati. Entitatile in limbajul XML sunt unitati de text, ce pot varia de la un singur caracter la un intreg document.
5. **Instructiuni de prelucrare**: tip special de marcaje care contin informatii despre anumite aplicatii ce urmeaza a fi executate.
6. **Sectiuni CDATA**: utilizate pentru a include blocuri de text ce contin caractere care ar fi fost recunoscute ca marcaje.
7. **Declaratia tipului de document**: Marcaj special, inclus in documentele XML ce are rolul de a specifica locatia si definitia tipului de document.

**3.Instrumente pentru dezvoltatori**

**Android SDK** (Software Development Kit) este un set de instrumente de dezvoltare folosite pentru a crearea aplicatiilor folosind platform Android. Aceste include urmatoarele:

* Bibliotecile necesare
* Debugger
* Emulator
* Documentatie relevanta pentru interfata de program Android (API)
* Fragmente de cod sursa
* Tutoriale pentru sistemul de operare Android[6]

De fiecare data cand Google lanseaza o noua versiune de Android, un SDK corespunzator este de asemenea lasnsat. Pentru a putea scrie programe ce folosesc cele mai noi proprietati, dezvoltatorii trebuie sa instaleze versiunea SDK pentru device-ul respective.[6]

Platformele de dezvoltate compativbile cu Android SDK include Windows(XP sau mai nou), Linux si MAC OS X. Componentele pentru Android SDK pot fi descarcate individual.[6]

Cea mai comuna metoda de a folosi Android SDK este printr-un mediu de dezvoldare integrat(IDE). Cel mai folosit este Eclipse, dar mai exista multe alte IDE-uri cum ar fi NerBeans sau IteelliJ, care functioneaza la fel de bine. Majoritatea IDE-urilor furnizeaza o interfata grafica, permitndu-le dezvoltatorilor sa isi duca la indeplinire munca intr-un timp cat mai scurt cu putinta. Deoarece aplicatiile Android sunt scrise in limbajul Java, utlizatorul trebuie sa aiba instalat kit-ul de dezvoltare Java( Java Development Kit / JDK).[6]

**4.Biblioteci suport**

Pachetul de biblioteci support este un set de biblioteci de cod care asigura versiuni compatibile de cadru de lucru Android si de asemenea asigura anumite caracteristici specific doar cu ajutorul bibliotecilor API. Fiecare biblioteca este compatibila cu o versiune specifica de Android API. [7]

Includedra bibliotecilor support este considerat un lucru de baza pentru dezvoltatorii de aplicatii, depinzand doar de plaja de versiuni tinta pentru aplicatia folosita si de API-urile pe care le foloseste. [7]

Fiecare biblioteca suport are in vedere un nivel API si fiecare furnizeaza un set diferit de caracteristici. Pentru folosirea lor eficienta, este importat de vazut ce caracteristici sunt necesare si de inteles ce caracteristici sunt suportate de fiecare librarie in parte, la fiecare nivel API.[7]

Cele mai importante biblioteci sunt urmatoarele:

* GSON
* RETROFIT
* EVENTBUS
* ACTIVEANDROID
* UNIVERSAL IMAGE LOADER[8]

**GSON**

Este o biblioteca Java folosita pentru a insera si a scoate obiecte Java din si in JSON. Acesta sarcina trebuie neaparat facuta daca se comunica cu APIs. Se foloste JSON deoarece este mult mai simplu decat XML.[8]

**RETROFIT**

Aceasta biblioteca este o solutie ideala pentru a organiza apelurile API intr-un proiect. Metoda de cerere si linkul relevant sunt adaugate prin adnotare, ceea ce face codul sa fie mai simplu si mai curat. Cu aceste adnotari, se poate manipula usor antetele si link-urile si se pot adaugata si corpuri de functie. Biblioteca RETROFIT foloseste GSON la initializare, deci nu este nevoie pentru cod suplimentar.[8]

**EVENTBUS**

Este o librarie care simplifica comunicatiile dintre diferite parti ale aplicatiei. De exemplu, se trimite ceva de la Activitate (clasa primara) la un serviciu care ruleaza, sau o simpla comunicatie intre fragmente. [8]

**ACTIVEANDROID**

Biblioteca ActiveAndroid este un ORM(Object Relational Mapping) pentru Android. Este o abstractizare SQLite care permite comunicatia cu baza de date de pe un device, fara a scrie cod SQL. Aceasta librarie simplifica mult codul, astfel nu mai este nevoie sa se scrie un cod de lungime mare, se scrie un cod de lungime mica si este mult mai usor de citit pentru programator.[8]

**UNIVERSAL IMAGE LOADER**

Aceasta biblioteca asigura in mod asincron incarcarea si stocarea in memoria cache a imaginilor.

In concluzie, folosirea bibliotecilor usureaza munca programatorilor si este mai rapid. Toate bibliotecile open-source sunt des testate si foarte simple de utilizat. In cele mai multe cazuri se pot importa in proiectele Android Studio.

**5. Dezvoltarea Aplicatiilor**

Trebuiesc indeplinite doua procese pentru a crea o aplicatie, acestea sunt:

1. Creare Design-ului folosind XML in layout-uri (ferestre)
2. Creare cod Java

Primul pas, dupa cum am mentionat este crearea design-ului aplicatiei. Pentru fieare clasa Activity exista un layout ( fisier XML) specific acestuia. La fisierul XML toate butoanele, tabelele si in general elementele de care avem nevoie sunt obiecte si trebuiesc denumite pentru a le putea referi din codul Java. Atentie ! Daca in fisierul XML nu se inchide un caracter de tipul’>’ sau exista greseli in limbajul de marcare, nu se poate rula interfata aplicatiei pe masina virtuala SDK.

Dupa ce interfata XML a fost creata si nu exista erori, se creeaza codul Java a aplicatiei. Se creeaza astfel o clasa prin care facem referinta la obiectele din layout-ul XML creat. Se importa bibliotecile dorite in cod pentru a usura munca programatorului. Daca exista butoane, de exemplu, se fac referinta la acestea si se progrmeaza ca atunci cand se apasa sa se indeplineasca o anumita sarcina. Daca in aplicatia dorita, este nevoie de mai multe ferestre(layout-uri) pentru a indeplini sarcinile dorite, se declara in android-manifest.xml, unde MainActivity este clasa de baza, si MainLayout este interfata de baza. Compilatorul Java ne ajuta sa testam aplicatia, daca exista greseli in cod, acesta arata liniile unde codul este gresit. Daca in cod nu exista greseli atunci aplicatia poate fi rulata. Inainte de rulare trebuie creata o masina virtuala cu ajutorul Android-SDK, unde se poate crea virtual smartphone-ul dorit pentru testare. Se pot alege diferite attribute si modele pentru acesta, cum ar fi: memoria RAM, memoria card-ulu SD, procesorul acestuia, marimea ecranului, modelul smartphone-ului etc.

Daca s-au indeplinit conditiile de mai sus se poate rula aplicatia pe smartphone-ul virtual, si dupa accea, datorita portabilitatii oferite de Java, se poate instala aplicatia cu terminatia fisierului .apk pe alte device-uri Android.

**6.Distributia aplicatiilor**

Fiind o platforma deschisa, Android ofera mai multe variante pentru distributia aplicatiilor, in functie de preferintele dezvoltatorului. Procesul de dezvoltare si impachetare a aplicatiilor este acelasi pentru orice tip de distributie folosit. Acest lucru economiseste timp si permite modificarea dupa plac a etapelor procesului de distributie.

**Distributia prin Piata Android (App Market) :** este cea mai des folosita metoda de distributie a aplicatiilor, avand audienta globala.

**Distributia prin pagini web** : daca nu se doreste lansarea aplicatiei pintr-o piata de aplicatii, aceasta poate fi incarcata pe un site.

**6. Impartirea Temei**

1. Vlad Ionita-Cristian (pag 6-10 inclusiv, pag 14)
2. Singureanu Liviu-Cristian(pag 3-5 inclusiv, pag 11-13 inclusiv)

**7. Bibliografie**

1. <http://developer.android.com>
2. <http://arstechnica.com/gadgets/2014/06/building-android-a-40000-word-history-of-googles-mobile-os/>
3. <http://stackoverflow.com/questions/9057396/what-does-api-level-mean>
4. <http://www.wikipedia.com/Android>
5. <http://inf.ucv.ro/~mihaiug/courses/xml/IntroducereInXML.pdf>
6. <https://www.techopedia.com/definition/4220/android-sdk>
7. <http://developer.android.com/tools/support-library/index.html#overview>
8. https://infinum.co/the-capsized-eight/articles/top-5-android-libraries-every-android-developer-should-know-about