|  |  |  |
| --- | --- | --- |
|  |  |  |

Семинарска работа по предметот

Мрежен софтвер

**Андроид апликација за   
комуникација на локална мрежа**

**Ментор:**

проф. д-р. Аристотел Тентов

**Асистент:**

Маја Маленко

**Изработиле:**

Благоја Стојкоски 370/2009

Никола Коевски 194/2009

# СОДРЖИНА

[СОДРЖИНА 1](#_Toc348194117)

[1 ОПИС НА ПРОЕКТОТ 2](#_Toc348194118)

[2 ФУНКЦИОНАЛНОСТИ ПОНУДЕНИ СО РЕШЕНИЕТО 2](#_Toc348194119)

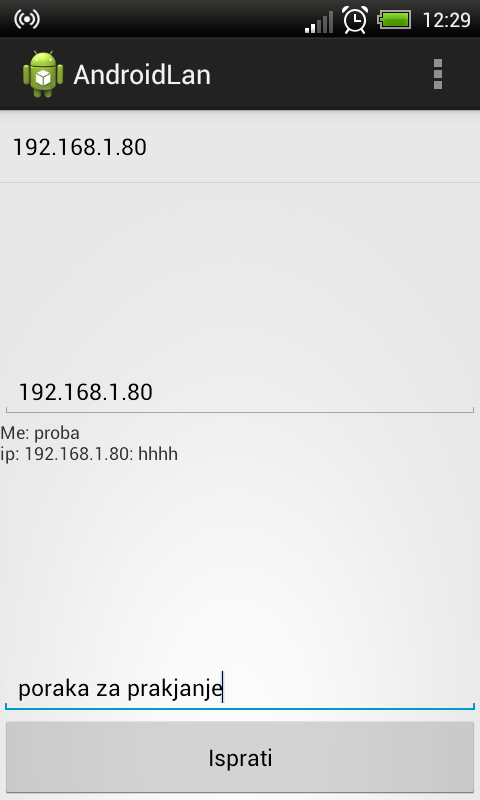
[3 СТРУКТУРА 3](#_Toc348194120)

# ОПИС НА ПРОЕКТОТ

Идејата позади нашата семинарска работа е изработка на мобилна апликација за комуникација на локална мрежа. Треба да се овозможи, при поврзување со рутер или Wireless Hotspot, комуникација преку локалната (lan) мрежа со користење на TCP/IP свитата на протоколи. Комуникацијата треба да се одвива преку внесување пораки преку одредена контрола. Притоа, апликацијата треба да имплементира приказ на достапните телефони (или други уреди) прикачени на мрежата. Од дадената листа на достапни уреди треба да може да се одбере до кој уред ќе се испрати пораката.

# ФУНКЦИОНАЛНОСТИ ПОНУДЕНИ СО РЕШЕНИЕТО

Мобилната апликација овозможува пронаоѓање на сите достапни телефони на локалната мрежа. Во моментов овозможено е скенирање само на адресите во опсегот 192.168.1.X, каде X е во опсег [0-255]. IP адресите на телефоните се прикажуваат во листа од која може да се одбере на кој телефон ќе се испрати порака. Исто така овозможено е и рачно внесување на IP адреса (преку текст контрола) доколку во мрежата се приклучил нов телефон по првичното пребарување за достапни телефони. Пораката се внесува преку текст контрола и преку притискање на копче се испраќа преку мрежата. За приказ на пораките имплементирана е текст контрола каде за секоја порака се прикажува и IP адресата од која е испратена пораката.



Нашето решение го тестиравме на три уреди, HTC One V, Sony Xperia Arc S и Sony Xperia Tipo. Сите овие уреди работат на оперативен систем Android верзија 4.0.3. Притоа беше тестирана комуникацијата помеѓу овие 3 уреди и успешно беа разменувани пораки помеѓу сите нив. За жал не бевме во можност да тестираме на уреди со постара верзија на Аndroid. Тестирањето беше вршено додека телефоните беа конектирани на разни бежични мрежи низ кампусот и на мрежи поставени на hotspot на секој од телефоните.

# СТРУКТУРА

|  |  |
| --- | --- |
| Класа | Објаснување |
| MainActivity.java | Ова е класата во која се исцртува корисничкиот интерфејс, секој засебен елемент од него се поврзува со функции кои ракуваат со соодветни ракувачи за нивните настани и се вклучува нитката во која работи серверот, објект од Server.java класата. Во onClick настанот за копчето за праќање, се креира и активира објект од класата NitkaZaPrakjanje кој ја зема пораката која е внесена во соодветното текстуално поле и ја праќа на селектираната адреса. Исто така главниот екран содржи листа во која се сместуваат IP адреси кои се конектирани на мрежата, и листа во која се сместуваат добиените и пратените пораки. |
| NitkaZaPrakjanje.java | Класа за испраќање на пораката. Ја користи АsyncTask која е обезбедена од Андроид API за полесна работа со нитки. Таа се користи за изведување на позадински операции врз Android UI нитката. Во конструкторот од класата се поставува контекстот (context) преку кој може да се пристапи до променливите од MainActivity класата како и пораката за праќање. Со функцијата pocni() се иницијализира објект од MyAsyncTaskPrakjanje класата. Во MyAsyncTaskPrakjanje класата дефинирана е doInBackground(Void... params) функцијата во која се поставува IP адресата на која се испраќа (klientIP). Доколку адресата не е поставена се печати порака за внесување на адреса. По поставувањето на адресата се иницијализира сокетот за испраќање (socketZaPrakjanje) и каналот за праќање (dataOutputStreamZaPrakjanje). Потоа пораката се испраќа со writeUTF(poraka). По завршување на испраќањето, со функцијата onPostExecute(Void result) се печати пратената порака преку функцијата refresh(dobienaPoraka) од MainActivity класата |
| NitkaZaPrimanje.java | Класа за примање на порака. Ја користи АsyncTask која е обезбедена од Андроид API за полесна работа со нитки. Таа се користи за изведување на позадински операции врз Android UI нитката. Во конструкторот од класата се поставува контекстот (context) преку кој може да се пристапи до променливите од MainActivity класата како и сокетот за примање на порака (socketZaPrimanje). Со функцијата pocni() се иницијализира објект од MyAsyncTaskPrimanje класата. Во MyAsyncTaskPrimanje класата дефинирана е doInBackground(Void... params) функцијата во која се поставува IP адресата од која се испраќа (cela). Потоа адресата се поставува во променливата klientIP од MainАctivity за да може да се прикаже и се додава во листата на адреси доколку не постои. По ова се иницијализира каналот за примање и се вчитува пораката со функцијата readUTF() во променливата dobienaPoraka. Потоа се печати испраќачката IP адреса и пораката. По завршување на примањето, со функцијата onPostExecute(Void result) се печати пратената порака преку функцијата refresh(dobienaPoraka) од MainActivity класата |
| NitkaZaSkeniranje.java | Класа за пребарување на достапни уреди во мрежата. Ја користи АsyncTask која е обезбедена од Андроид API за полесна работа со нитки. Таа се користи за изведување на позадински операции врз Android UI нитката. Во конструкторот од класата се поставува контекстот (context) преку кој може да се пристапи до променливите од MainActivity. Со функцијата pocni() се иницијализира објект од MyAsyncTaskAdresi класата. Во MyAsyncTaskAdresi класата дефинирана е doInBackground(Void... params) функцијата во која се дефинира типот на IP адреса според кој ќе се пребарува. Потоа за секоја се проверува дали таа е доделена на хост со функцијата isReachable(). Доколку постои хост на адресата, таа се додава во листата на достапни адреси (context.adresi) |
| Server.java | Како и другите класи, во Server класата се креира објект од AsyncTask класата обезбедена од Android АPI-то. Во овој објект се поставува серверски сокет кој слуша за конекции на порт 8888. Ова се прави со системски повик accept(), кој кога ќе врати одговор, се креира нов објект од NitkaZaPrimanje во кој се bind-ува нов сокет со кој ќе се врши примањето на пораката. Oваа нитка се активира во MainActivity класата со повикување на функцијата pocni() дефинирана во класата Server. |