به نام خدا

عماد آقاجانی

۸۸۵۲۱۳۴۴

گزارش پروژه درس شبکه

انتقال و پخش همزمان دیتا

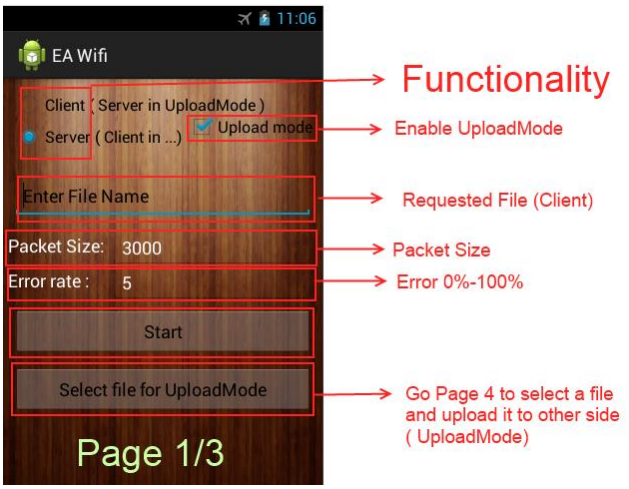
بین دو دستگاه آندروید

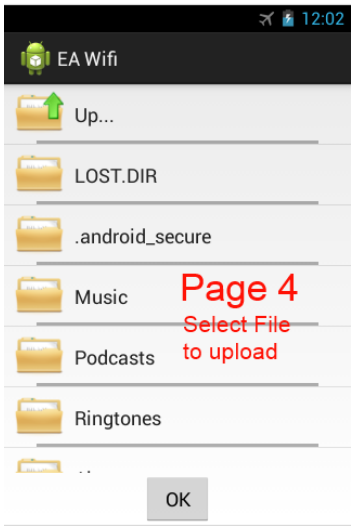
استاد درس: **دکتر مرتضی آنالوئی**

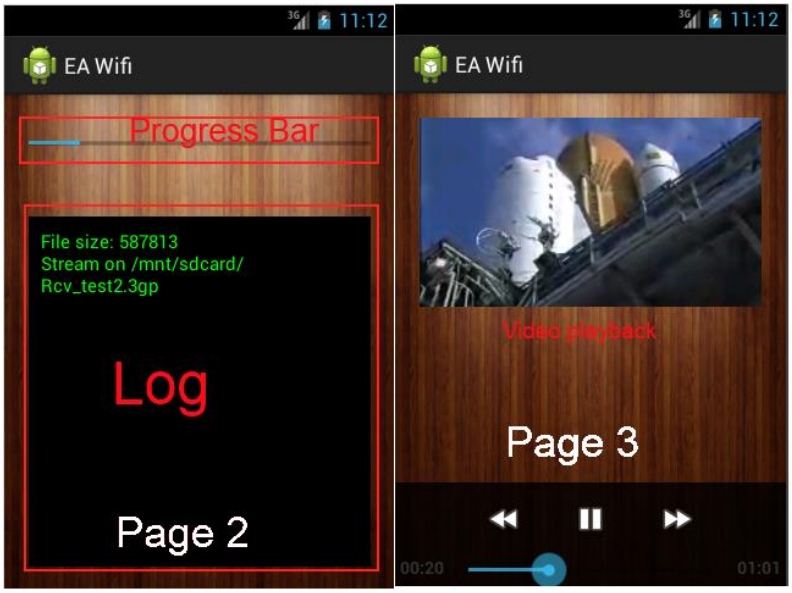
ظاهر برنامه

در ابتدا به توضیح UI برنامه میپردازیم. این برنامه از ۴ صفحه تشکیل شده که ۳ صفحه آن بصورت کشویی به یکدیگر متصل بوده و صفحه چهارم درواقع یک Activity برای انتخاب فایل میباشد.

در شکل زیر نمایی از تمام صفحات و توضیحات هر قسمت آمده است:







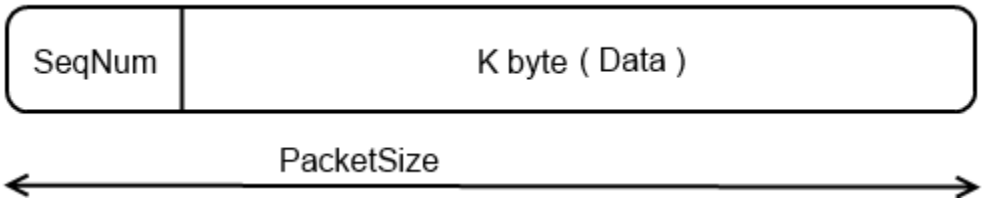
پروتکل ارتباطی

در این برنامه از پروتوکلی "مشابه" Go-Back-N سعی شد استفاده شود. بدین معنی که در سمت سرور یک پنجره ارسال تعریف شده است که توسط یک متغیر قابل تغییر است. همچنین برنامه طوری طراحی شده است که امکان تغییر داینامیک این مقدار در هنگام ارسال وجود دارد و میتوان براحتی مکانیزم هایی مشابه SlowStart را بدون زحمت اضافی پیاده سازی کرد.

در سمت کلاینت به ازی هر بسته ارسالی یک بسته حاوی شماره اولین بسته مورد انتظار کلاینت بعنوان Ack ارسال میگردد.

لازم به ذکر است که یک تایمر (مشابه GBN) برای آخرین بسته ارسالی و Ack نشده وجود دارد که درصورت رخداد تایمر تمامی بسته های پنجره جاری مجددا ارسال میگردد.

روند کار بدین شکل است که ابتدا سرور با زدن دکمه Start به حالت آماده باش در می آید. سپس از سوی کلاینت نام فایل درخواستی فرستاده میشود. در سوی سرور صحت درخواست و وجود فایل بررسی و آنالیز فایل آغاز میگردد و بسته ها بترتیب ارسال میگردد. بسته ها در فرم کلی بصورت زیر می باشند:



اولین بسته (بسته ۰ام) حاوی سایز کل فایل درخواست شده است. از این عدد برای پیشبینی تعداد کل بسته ها در سمت کلاینت استفاده میگردد (با علم به اینکه PacketSize و SeqNum در هردو سمت باید یکسان باشند. از بسته ۱ به بعد، بسته ها حاوی شماره بسته و اطلاعات فایل میباشد.

در آخر نیز یک بسته حاوی رشته "END"در بخش دیتا، به عنوان خاتمه از سمت سرور ارسال میشود.

نحوه کار برنامه

این برنامه در دو مد کاری زیر توانایی برنامه ریزی دارد:

۱. دریافت ویدیو با قابلیت پخش همزمان (مد اصلی) (Download)

۲. ارسال ویدیو با قابلیت پخش همزمان (Upload)

همچنین این برنامه قابلیت پخش هرگونه فایل دریافت شده را توسط یکی از ویژگی های اندروید دارا میباشد. در زیر به توضیح شماره ۱ میپردازیم و در انتها بصورت مختصر به نحوه مد دوم اشاره میکنیم.

نحوه دریافت (سمت کلاینت)

در سمت کلاینت پس از زدن دکمه Start نام فایل وارد شده توسط کابر توسط سوکتی به پورت ۷۰۰۰ فرستاده میشود.

سپس کلاینت در یک حلقه بینهایت منتظر دریافت بسته ها میگردد. ابتدا متغیر expected Packet Number برابر با صفر (اولین بسته قرار میگیرد) .

با آمدن هر بسته ابتدا بررسی میشود که آیا بسته مربوط به بسته "END" میباشد یا خیر. در این حالت برنامه از حلقه خارج شده و نوشتن در فایل خاتمه پیدا میکند. در سایر موارد، بسته ها حاوی یک شماره هستند (که این عدد بشکل رشته ای از کاراکتر ها ریخته نشده. این ویژگی کمک میکند که به ازای رزو n بایت برای شماره بسته $2^{n/8}$ شماره را بتوان پشتبانی کرد.) در یک حالت خاص و برای بسته شماره صفر، اطلاعات حجم فایل دریافت و فضای لازم برای دریافت سایر بسته ها اختصاص داده میشود.

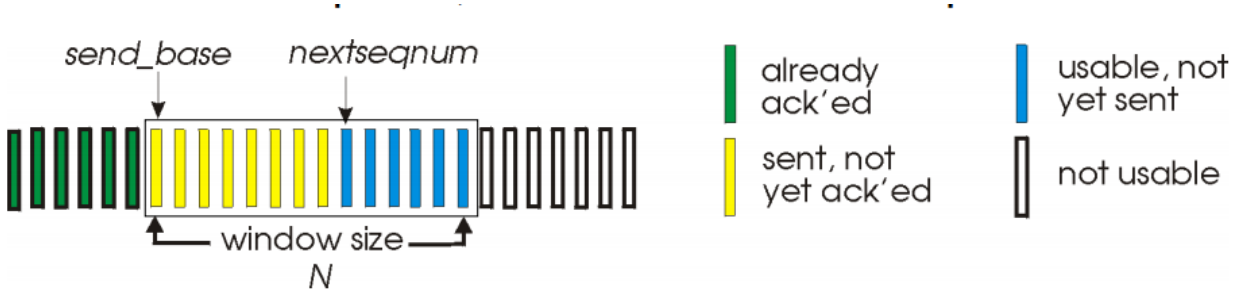
نحوه دریافت (سمت سرور)

در ابتدا بازدن دکمه Start سرور شروع به گوش کردن برای اولین درخواست قرار میگیرد. پس از دریافت درخواست، سرور صحت درخواست و وجود فایل بررسی و آنالیز فایل درخواست شده را آغاز میکند. در این مرحله یک آرایه از بسته ها بوجود می آید.

سپس ادامه روند برنامه از نخ(Thread) فعلی به دو نخ موازی انتقال میابد:

```
threadFunc_SendFile => { threadFunc_BGNSendData || threadFunc_BGNACK )
```

در `threadFunc_BGNSendData` بسته های تشکیل شده در مرحله قبل بترتیب و با یک فاصله زمانی کوتاه ارسال میشوند. سیستم Go-Back-N در اینجا همانند اسلاید های درس پیاده سازی شده است :



`threadFunc_BGNACK` پیدرواقع بعنوان دریافت کننده Ack ها قرار داده شده است. با دریافت Ack برای بسته N پنجره حرکت کرده و Base برابر N قرار میگیرد.

برای بحث Timeout برای آخرین بسته ارسال شده ولی Ack نشده نیز از یک تایمر که پس از گذشت `ackTimer_interval` ، `ackUpdate` را صدا میزند استفاده شده. وظیفه این تابع شفاف بوده و برای ارسال مجدد بسته های پنجره جاری تنها کافیست مقدار NextSeqNum را برابر Base قرار دهد.

نحوه کار مد ارسال (upload)

مد ارسال با بهره از پیاده سازی بخش دریافت، دارای یک تفاوت کوچک در روند میباشد. آن اینکه مرحله ارسال نام فایل حذف شده و کلاینت از ابتدا به دنبال دریافت بسته ای با شماره صفر برای آماده سازی دریافت سایر بسته های است.

نمایی از مرحله انتخاب فایل را در ابتدای این مستند مشاهده کردید (صفحه چهار)

مشکلات

ارسال در انجام این پروژه دو مشکل عمده ی پیش رو بشرح زیر بود:

۱. اتصال دو Emulator

برای حل این مشکل متوجه شدیم که باید بکمک اتصال به emulator ها جریان ورودی آنها را از OS به پورتی که روی آن receive انجام میدهند اصلاح کنیم. نمونه این کد:

telnet localhost 5554

redir add udp:[Port which other side send info]:[port we listen to]

۲. اجرای هزمان (Streaming)

یک مشکل نوع فایل های قابل اجرا بشکل Partial بود که در این آزمایش پسوند 3gp بعنوان یک نمونه خوب استفاده گشت.

مشکل دیگر عدم آگاهی از میزان قابل پخش از ویدیو دریافتی بود که همچنان به جواب کاملی جز کنترل آن توسط دستور زیر نرسیده ایم :

```
videoView.setOnErrorListener(new OnErrorListener () {
    @Override
    public boolean onError(MediaPlayer mp, int what, int extra) {
        Log.e(TAG, "Error playing video");
        return true;
    }
});
```