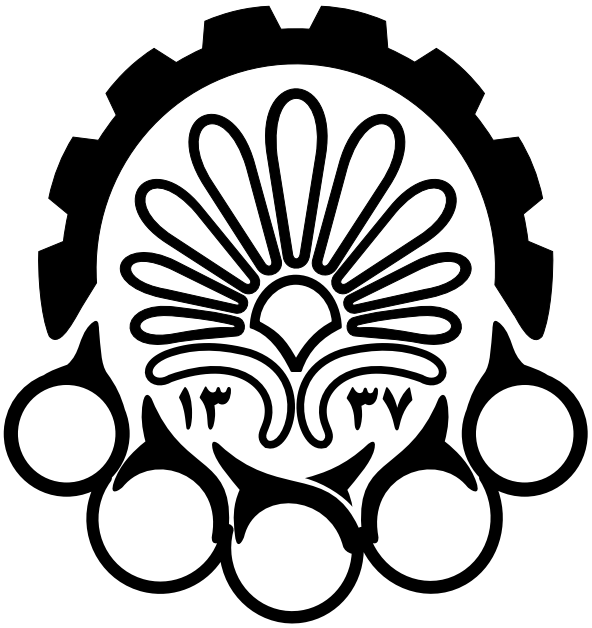

گزارش کار پروژه پروکسی

استاد صادقیان

شایان صورتگر ۹۷۳۱۰۹٤

مقدمه

در این پروژه یک پروکسی ساده همراه با قابلیت های کشینگ و فیلترینگ را با جاوا پیاده سازی کردیم.

همچنین برنامه دارای رابط کاربری گرافیکی برای فیلتر کردن آدرس های مختلف میباشد.

راه اندازی پروژه

پروژه با استفاده از java Main اجرا شده و روی پورت پیشفرض (1234) شروع به گوش کردن درخواست ها میکند.

همچنین میتواند با استفاده از اولی آرگیومنتی که هنگام اجرای برنامه به args میدهید پورت سوکت را مشخص کنید.

مثال: java Main 2525

معرفی کلاس ها

Main

متود main که وظیفه ساخت و اجرای پروکسی ما را دارد در این کلاس وجود دارد.

Proxy

گوش دادن در پورت مشخص و منتظر ماندن برای کانکشن های جدید و ساخت هندلر جدید و اجرای ترد ان کار این کلاس میباشد.

Request Handler

هندل کردن تمامی اطلاعات بین سرور و کلاینت توسط این کلاس انجام میشود.
همچنین کش کردن و چک کردن فیلتر نیز در این کلاس انجام میشود.

Admin Frame

گرافیک پروکسی که برای اضافه و حذف کردن فیتر ها ساخته شده در این کلاس پیاده سازی شده.

Data Forwarder

متود run نوشته شده در این کلاس در یک ترد جدا تمامی بایت های موجود داخل یک اینپوت استریم را به داخل یک اوتپوت استریم انتقال میدهد.

Cache

اطلاعات کشینگ در این کلاس ذخیره میشود.

Not Cached Exception

این اکسپشن زمانی که ارسال اطلاعات کش شده ناموفق شود پرتاب میشود

هندل کردن ریکوئست ها

بعد از اینکه کی درخواست جدید از سمت کلاینتی دریافت میشود در ابتدا توسط متود isFilter در کلاس Proxy بررسی میشود که فیلتر است یا خیر.

در صورتی که فیلتر بود درخواست ایگنور میشود و ترد بسته میشود.

در غیر این صورت نسبت به Http یا Https بودن درخواست ، آن را هندل میکنیم.

هندل کردن ریکوئست های HTTPS

اینکار در متود handleHttpsRequest در کلاس RequestHandler انجام میشود.

در صورتی که در سوکت پروکسی کلمه CONNECT نوشته شود این متود برای هندل کردن دیتا های بین سرور و کلاینت فراخوانی میشود.

بعد از باز کانکت شدن به سوکت سرور، برنامه با استفاده از کلاس Data Forwarder ورودی خروجی های سرور و کلاینت را به هم وصل میکند تا دیتای مورد نظرشان را به هم انتقال بدهند.

به این صورت که خروجی کلاینت به ورودی سرور و خروجی سرور به ورودی کلاینت وصل میشود و در دو ترد جدا فرایند انتقال داده شروع میشود.

هندل کردن ریکوئست های HTTP

اینکار در متود handleHttpRequest در کلاس RequestHandler انجام میشود.

در صورتی که در سوکت پروکسی کلمه CONNECT نوشته نشود این متود برای هندل کردن دیتا های بین سرور و کلاینت فراخوانی میشود.

متود در ابتدا در کش های موجود دنبال ریکوئست دریافت شده میگردد.

در صورتی که پیدا کرد سعی میکنم با پر کردن هدر If-None-Match از دریافت اطلاعات تکراری جلوگیری کند.

در صورتی که با فرستادن Etag ای که در کش پیدا شده موجود بود باز هم کد ۲۰۰ را در جواب بگیریم اکسپشن NotCachedException رخ میدهد.

در غیر این صورت (اگر کد ۳۰٤ دریافت کردیم) فایل کش شده ی قبلی را برای کلاینت ارسال میکند.

برای ارسال درخواست به سرور و ارسال پاسخ به کلاینت، تمامی بایت های خوانده شده از کلاینت را (که میتواند کاراکتر های UTF8 باشند یا عکس یا فایل های دیگر) برای سرور ارسال میکند.

سپس جواب سرور را برای کلاینت ارسال میکند.

در صورتی که توی header های جواب key برای Etag وجود داشت، جواب سرور را کش میکنیم