7.5 .Netty 初认识 --netty 进行文件读取 开发示例

作者：289306290
原文链接：https://ld246.com/article/1583464207871
来源网站：链滴

实现功能:

命令行telnet 连接服务器,然后输入文件路径，打印文件内容


```
# apple @ wujingjian in ~ [11:04:28]
$ telnet localhost 8080
Trying ::1...
Connected to localhost.
Escape character is '^]'.

File not found :
/tmp/sangfordnscmp.txt
/tmp/sangfordnscmp.txt 592
An asterisk (*) denotes that a network service is disabled.
An is not a recognized network service.
** Error: The parameters were not valid.
iPhone USB
iPhone is not a recognized network service.
** Error: The parameters were not valid.
Bluetooth PAN
Bluetooth is not a recognized network service.
** Error: The parameters were not valid.
Thunderbolt Bridge
Thunderbolt is not a recognized network service.
** Error: The parameters were not valid.
USB Ethernet
USB is not a recognized network service.
** Error: The parameters were not valid.
Wi-Fi
There aren't any DNS Servers set on Wi-Fi.

File not found :
```

注意: 示例中仅仅简单实现，当进行大文件传输时候可能导致内存溢出,Netty提供了ChunkedWriteHandler来解决大文件或者码流传输过程中可能发生的内存溢出问题，具体可以再进行研究学习。

FileServer.java

```
package club.wujingjian.com.wujingjian.netty.file;

import io.netty.bootstrap.ServerBootstrap;
import io.netty.channel.ChannelFuture;
import io.netty.channel.ChannelInitializer;
import io.netty.channel.ChannelOption;
import io.netty.channel.EventLoopGroup;
import io.netty.channel.nio.NioEventLoopGroup;
import io.netty.channel.socket.SocketChannel;
import io.netty.channel.socket.nio.NioServerSocketChannel;
import io.netty.handler.codec.LineBasedFrameDecoder;
import io.netty.handler.codec.string.StringDecoder;
import io.netty.handler.codec.string.StringEncoder;
import io.netty.util.CharsetUtil;

public class FileServer {
    public void run(int port) throws Exception {
        EventLoopGroup bossGroup = new NioEventLoopGroup();
        EventLoopGroup workerGroup = new NioEventLoopGroup();
        try {
            ServerBootstrap b = new ServerBootstrap();
            b.group(bossGroup,workerGroup)
```

```
                .channel(NioServerSocketChannel.class)
                .option(ChannelOption.SO_BACKLOG,100)
                .childHandler(new ChannelInitializer<SocketChannel>() {
                    @Override
                    protected void initChannel(SocketChannel socketChannel) throws Exception {
                        socketChannel.pipeline().addLast(
                                new StringEncoder(CharsetUtil.UTF_8),
                                new LineBasedFrameDecoder(1024),
                                new StringDecoder(CharsetUtil.UTF_8),
                                new FileServerHandler()
                        );
                    }
                });
            ChannelFuture f = b.bind(port).sync();
            System.out.println("Start file server at port : " + port);
            f.channel().closeFuture().sync();
        } finally {
            bossGroup.shutdownGracefully();
            workerGroup.shutdownGracefully();
        }
    }

    public static void main(String[] args) throws Exception {
        int port = 8080;
        if (args != null && args.length > 0) {
            port = Integer.parseInt(args[0]);
        }
        new FileServer().run(port);

    }
}
```

FileServerHandler.java

```
package club.wujingjian.com.wujingjian.netty.file;

import io.netty.channel.ChannelHandlerContext;
import io.netty.channel.DefaultFileRegion;
import io.netty.channel.FileRegion;
import io.netty.channel.SimpleChannelInboundHandler;

import java.io.File;
import java.io.RandomAccessFile;

public class FileServerHandler extends SimpleChannelInboundHandler<String> {

    private static final String CR = System.getProperty("line.separator");
    @Override
    protected void messageReceived(ChannelHandlerContext ctx, String msg) throws Exception
{
        File file = new File(msg);
        if (file.exists()) {
            if (!file.isFile()) {
```

```
                ctx.writeAndFlush("Not a file : " + file + CR);
                return;
            }
            ctx.write(file + " " + file.length() + CR);
            RandomAccessFile randomAccessFile = new RandomAccessFile(msg, "r");
            FileRegion region = new DefaultFileRegion(randomAccessFile.getChannel(), 0, rando
AccessFile.length());
            ctx.write(region);
            ctx.writeAndFlush(CR);
            randomAccessFile.close();
        } else {
            ctx.writeAndFlush("File not found :" + file + CR);
        }
    }

    @Override
    public void exceptionCaught(ChannelHandlerContext ctx, Throwable cause) throws Excepti
n {
        cause.printStackTrace();
        ctx.close();
    }
}
```