欢迎大家来到第五阶段课程

《分布式流媒体》实训项目

TNV DAY13

复习课

预习内容

HTTP服务器（4）

HTTP服务器（4）

业务服务类(service_c)的一级方法

知识讲解

- 处理GET方法：doGet
  - 从请求中提取资源路径
  - 从资源路径中提取路由
    - 有文件路由：处理文件路由
    - 无文件路由：响应请求异常(STATUS_BAD_REQUEST)
  - 发送响应
- 处理POST方法：doPost
  - 发送空响应
- 处理选项设置：doOptions
  - 设置响应选项
  - 发送空响应

业务服务类(service_c)的一级方法

知识讲解

- 处理错误：doError
  - 发送响应头
    - STATUS_BAD_REQUEST
  - 发送响应体
    - <root error='some error happened' />
- 处理其它：doOther
  - 发送响应头
    - STATUS_BAD_REQUEST
  - 发送响应体
    - <root error='unknown request method %s' />

业务服务类(service_c)的二级方法

知识讲解

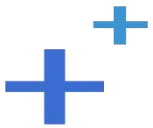
- 处理文件路由：files
  - 以与请求相同的连接模式回复响应
  - 从请求的资源路径中提取文件ID并检查之
  - 设置响应中的内容字段
  - 实例化客户机对象
  - 向存储服务器询问文件大小
  - 从存储服务器下载文件并发送响应
    - 初始化未下载字节数为文件大小
    - 若还有未下载数据则循环执行
      - 下载数据
      - 发送响应
      - 继续下载
  - 返回成功

附录：程序清单

TNV/src/06_http/05_service.cpp

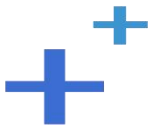
知识讲解

```
bool service_c::doGet(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res) {
        //从请求中提取资源路径
        acl::string path = req.getPathInfo();

        if (!path.ncompare(ROUTE_FILES, strlen(ROUTE_FILES)))
                //处理文件路由
                files(req, res);
        else {
                logger_error("unknown route, path: %s", path.c_str());
                res.setStatus(STATUS_BAD_REQUEST);
        }

        //发送响应
        return res.write(NULL, 0);
```



TNV/src/06_http/05_service.cpp

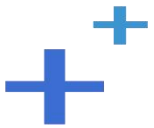

```
}

bool service_c::doPost(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res) {
        //发送响应
        return res.write(NULL, 0);
}

bool service_c::doOptions(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res) {
        res.setStatus(STATUS_OK)
           .setContentType("text/plain;charset=utf-8")
           .setContentLength(0)
           .setKeepAlive(req.isKeepAlive());
```



TNV/src/06_http/05_service.cpp

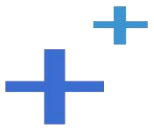
知识讲解

```
        //发送响应
        return res.write(NULL, 0);
}

bool service_c::doError(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res) {
        //发送响应头
        res.setStatus(STATUS_BAD_REQUEST)
           .setContentType("text/html;charset=");
        if (!res.sendHeader())
                return false;

        //发送响应体
        acl::string body;
        body.format("<root error='some error happened' />\r\n");
```



TNV/src/06_http/05_service.cpp

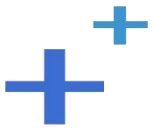

```
        return res.getOutputStream().write(body);
}

bool service_c::doOther(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res, char const* method) {
        //发送响应头
        res.setStatus(STATUS_BAD_REQUEST)
           .setContentType("text/html;charset=");
        if (!res.sendHeader())
                return false;

        //发送响应体
        acl::string body;
        body.format("<root error='unknown request method %s' />\r\n",
                method);
```



TNV/src/06_http/05_service.cpp

知识讲解

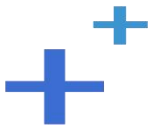
```
        return res.getOutputStream().write(body);
}

////////////////////////////////////////////////////////////////////////

//处理文件路由
bool service_c::files(acl::HttpServletRequest& req,
        acl::HttpServletResponse& res) {
        //以与请求相同的连接模式回复响应
        res.setKeepAlive(req.isKeepAlive());

        //从请求的资源路径中提取文件ID并检查之
        acl::string path = req.getPathInfo();
        acl::string fileid = path.right(strlen(ROUTE_FILES) - 1);
        if (!fileid.c_str() || !fileid.size()) {
```



TNV/src/06_http/05_service.cpp

```
            logger_error("fileid is null");
            res.setStatus(STATUS_BAD_REQUEST);
            return false;
        }
        logger("fileid: %s", fileid.c_str());

        //设置响应中的内容字段
        res.setContentType("application/octet-stream");
        acl::string filename;
        filename.format("attachment;filename=%s", fileid.c_str());
        res.setHeader("content-disposition", filename.c_str());

        client_c client; //客户机对象

        //向存储服务器询问文件大小
```

知识讲解



TNV/src/06_http/05_service.cpp

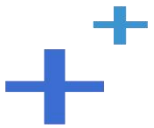
知识讲解

```
long long filesize = 0;
if (client.filesize(APPID, USERID, fileid, &filesize) != OK) {
        res.setStatus(STATUS_INTER_SERVER_ERROR);
        return false;
}
logger("filesize: %lld", filesize);

//从请求的头部信息中提取范围信息
long long range_from, range_to;
if (req.getRange(range_from, range_to)) {
        if (range_to == -1)
                range_to = filesize;
}
else {
        range_from = 0;
```

TNV/src/06_http/05_service.cpp

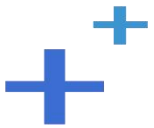
知识讲解

```
            range_to = filesize;
    }
    logger("range: %lld-%lld", range_from, range_to);

    //从存储服务器下载文件并发送响应
    long long remain   = range_to - range_from;          //未下载字节数
    long long offset   = range_from;                     //文件偏移位置
    long long size     = std::min(remain, FILE_SLICE); //期望下载大小
    char*     downdata = NULL;                          //下载数据缓冲
    long long downsize = 0;                              //实际下载大小
    while (remain) { //还有未下载数据
            //下载数据
            if (client.download(APPID, USERID, fileid.c_str(),
                    offset, size, &downdata, &downsize) != OK) {
                    res.setStatus(STATUS_INTER_SERVER_ERROR);
```



TNV/src/06_http/05_service.cpp


```
                return false;
            }
            //发送响应
            res.write(downdata, downsize);
            //继续下载
            remain -= downsize;
            offset += downsize;
            size    = std::min(remain, FILE_SLICE);
            free(downdata);
        }

        return true;
}
```



下节课见