

版本控制文档

宇宙无敌工作组



2015-4-11

# 文档更新记录

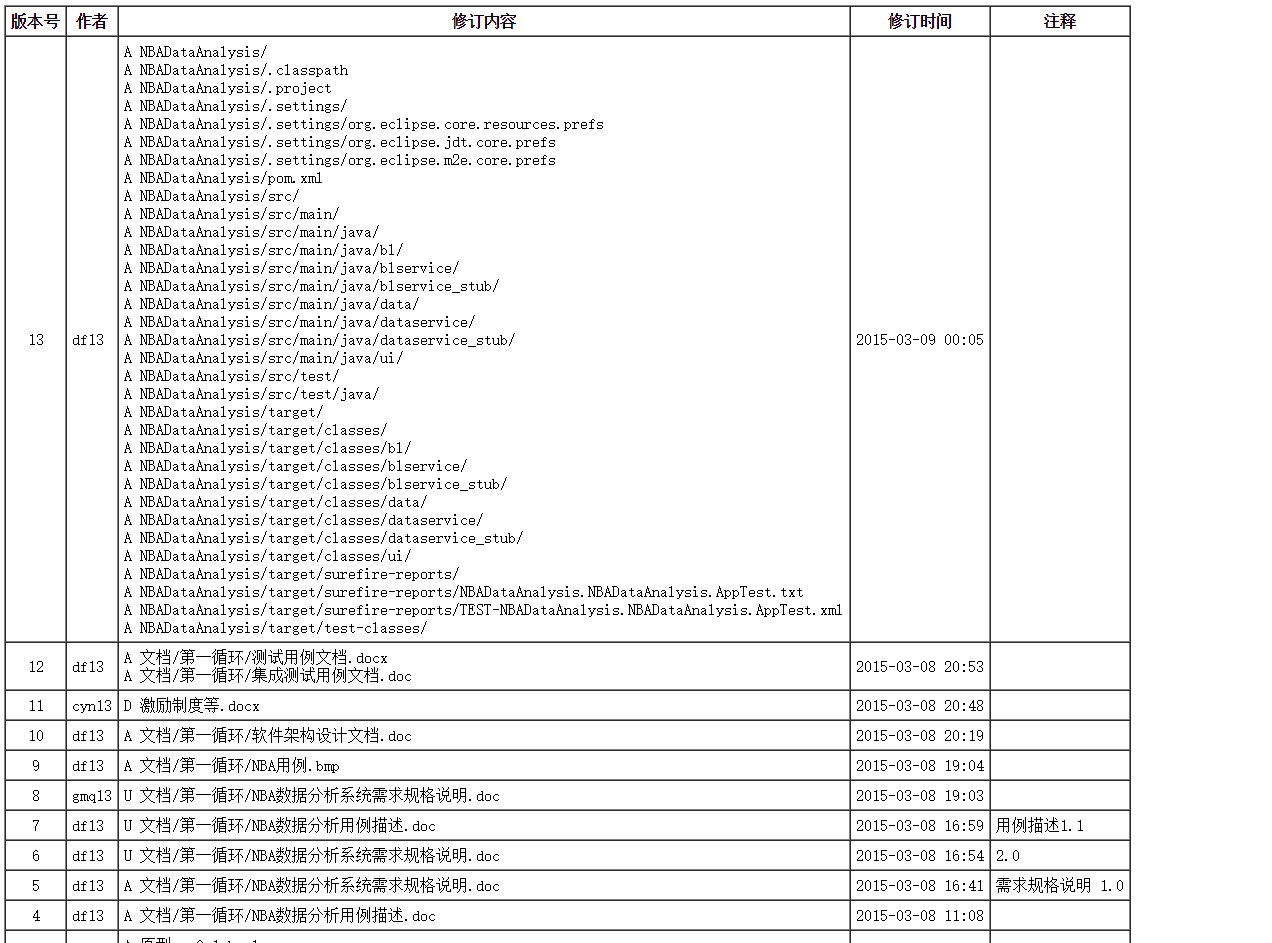
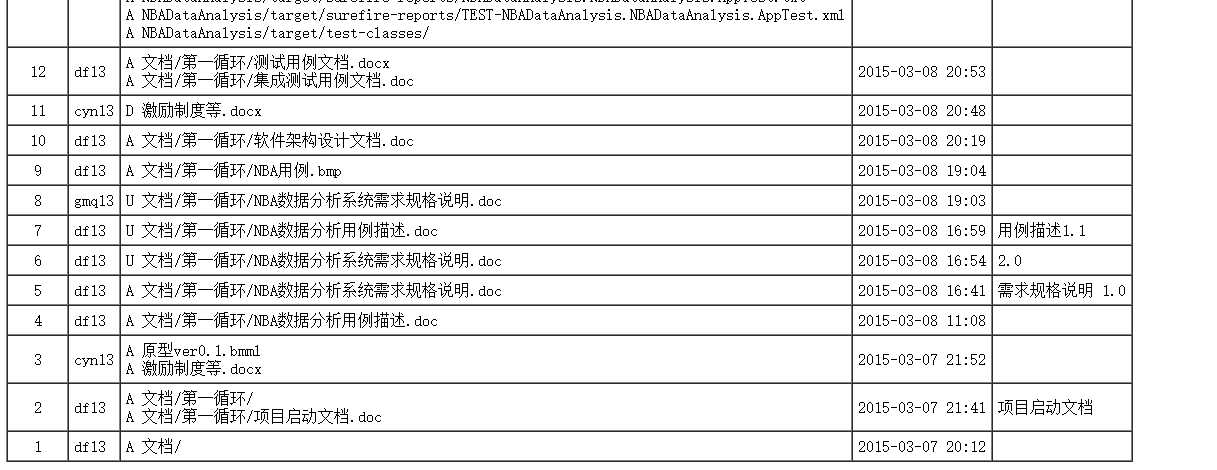
|  |  |  |  |
| --- | --- | --- | --- |
| 更新者 | 更新时间 | 更新内容 | 版本号 |
| 陈依娜 | 2015/4/11 | 建立版本控制文档，添加SVN部分的版本控制内容 | 草稿V1.0 |
| 陈依娜 | 2015/4/19 | 添加git部分的版本控制内容 | 正式版V1.0 |
| 陈依娜 | 2015/6/13 | 添加迭代三的版本控制内容 | 正式版V1.1 |

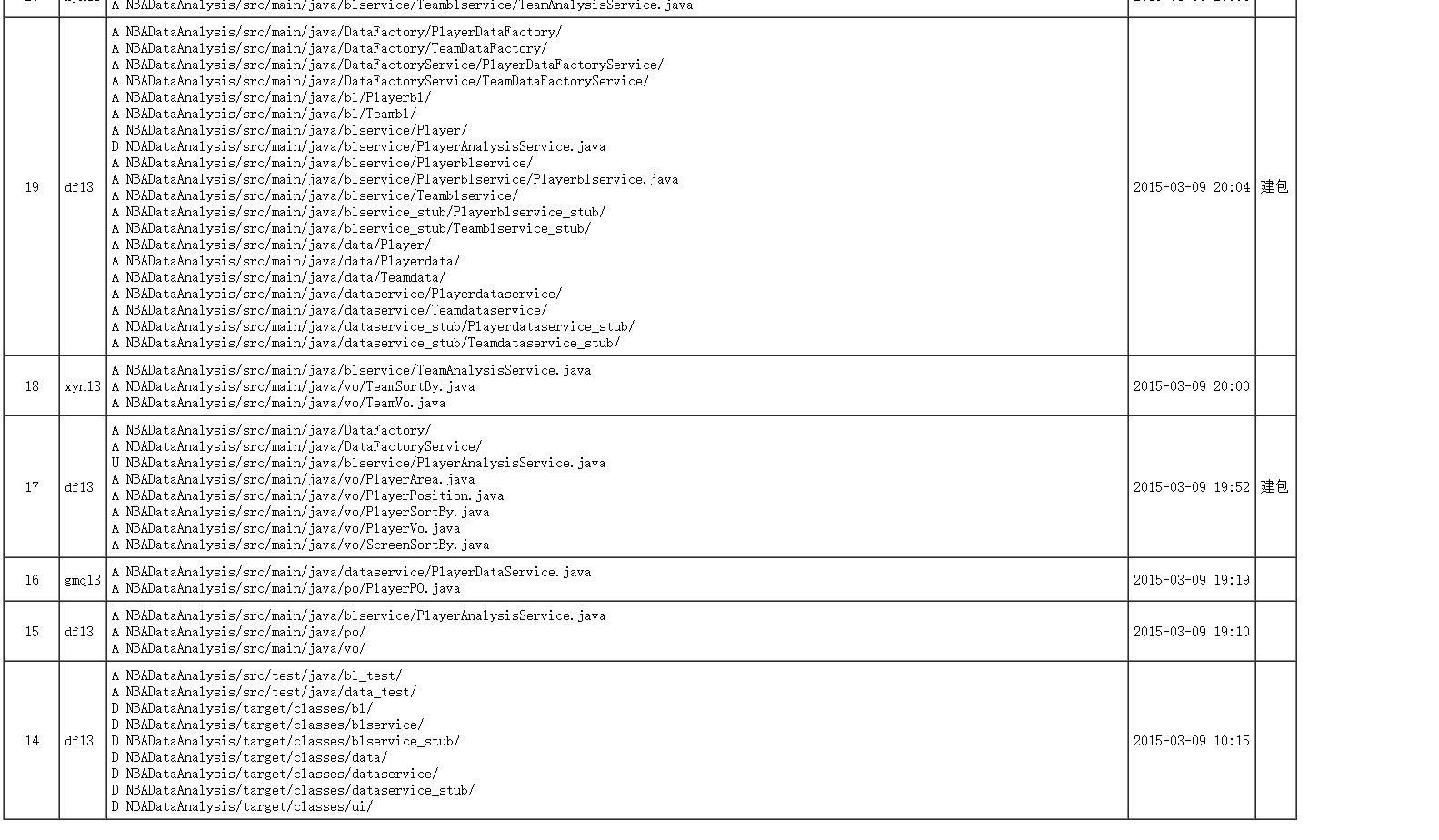
# 版本控制说明

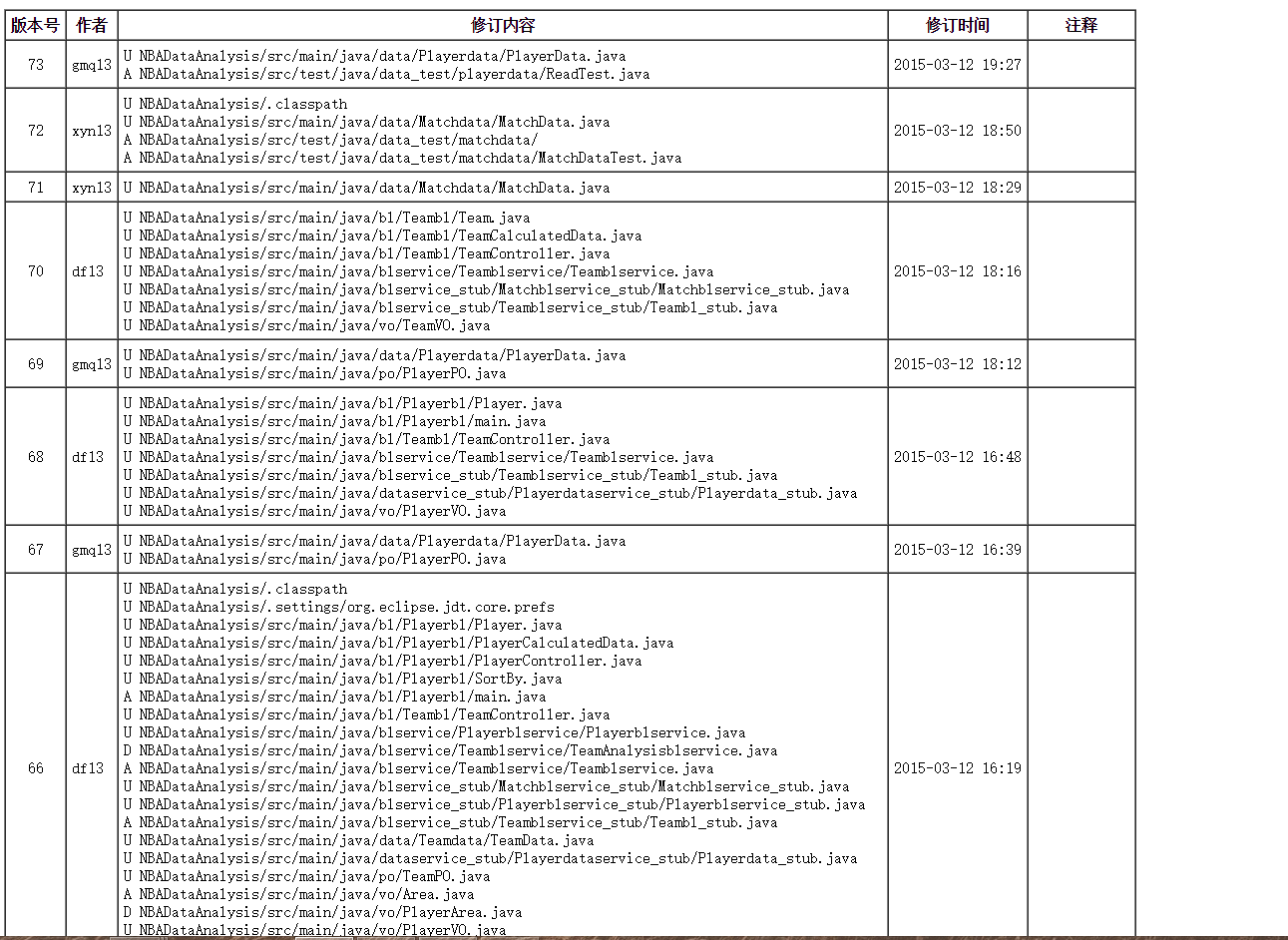
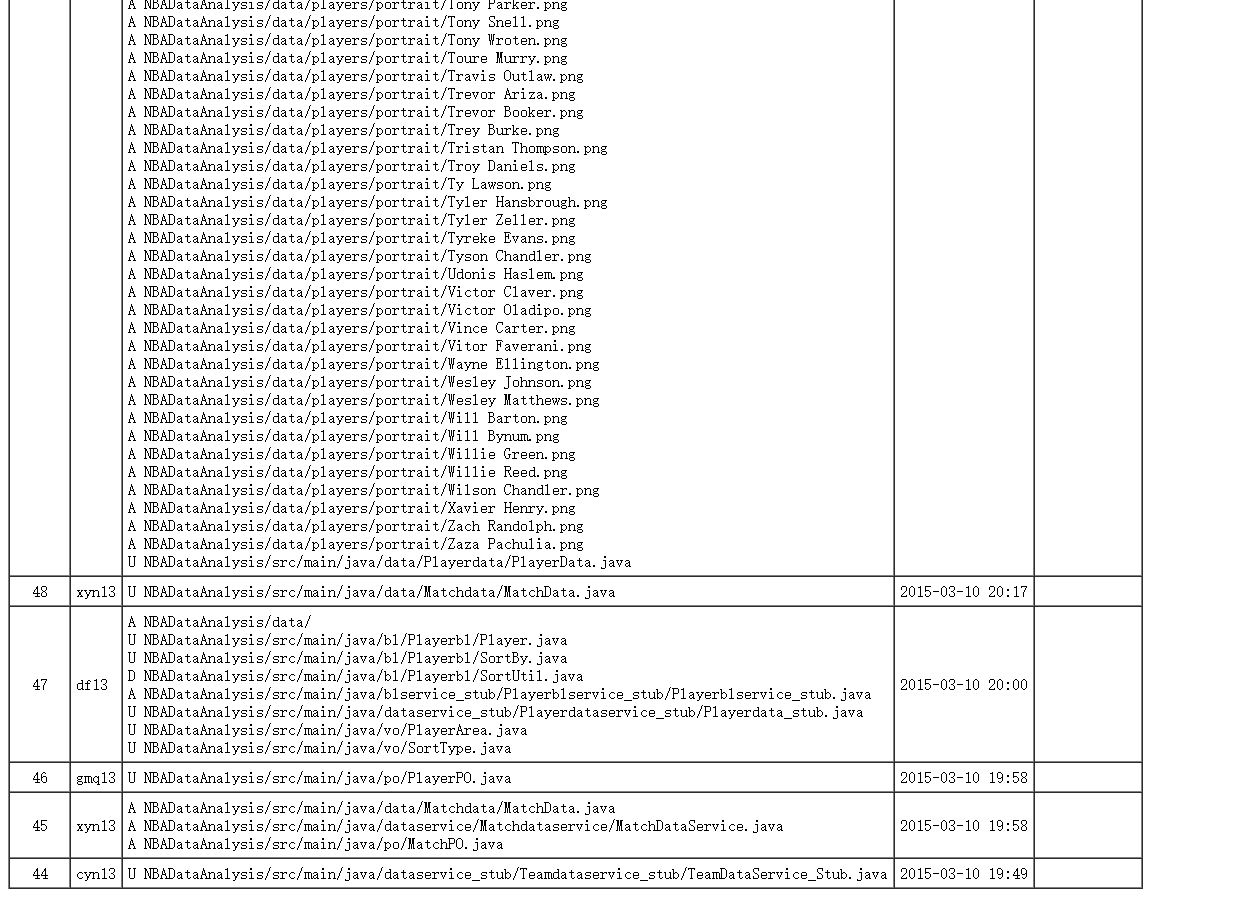
我们小组在迭代一阶段使用SVN作为版本控制工具，迭代二使用GitHub作为版本控制工具。这里会给出SVN(迭代一)的历史更新和截图和Github(迭代二)的历史更新与截图。

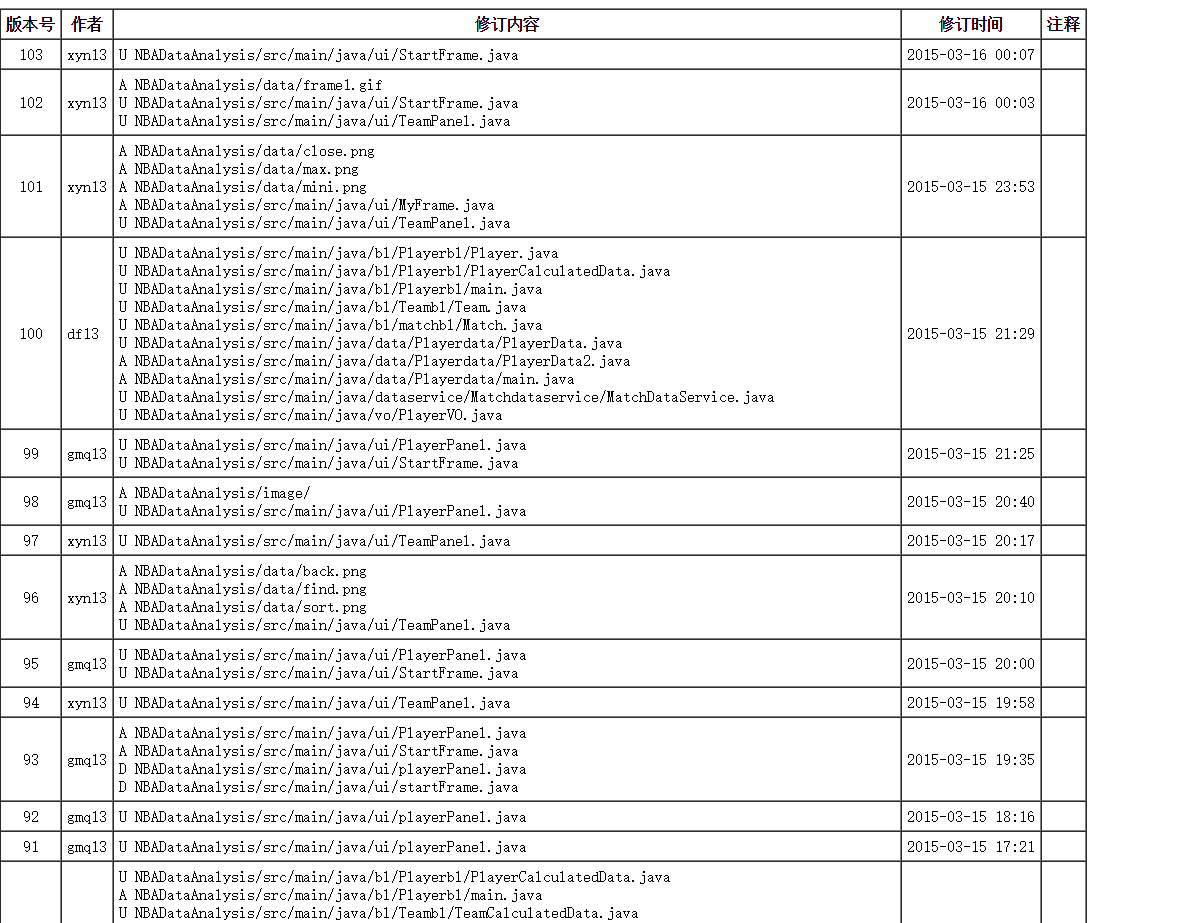
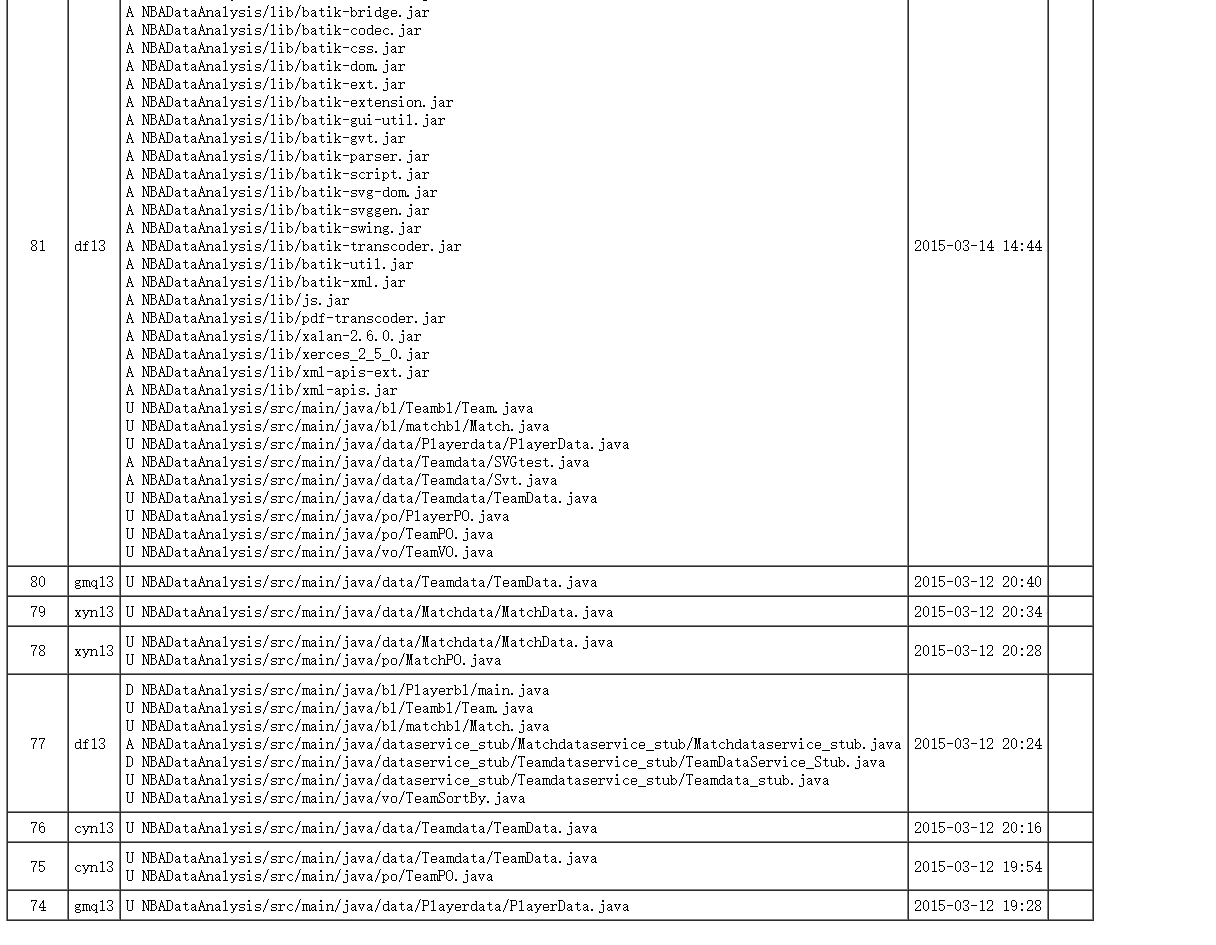
# 版本控制历史版本记录

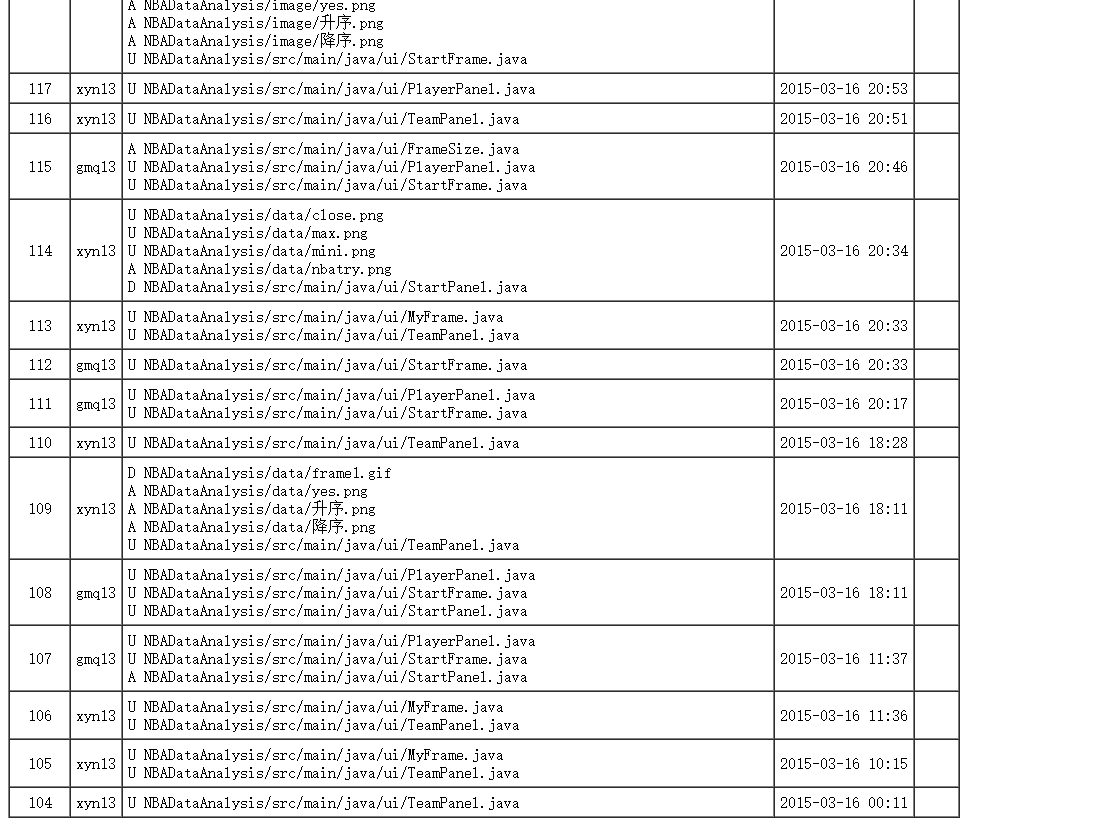
## SVN（迭代一）部分



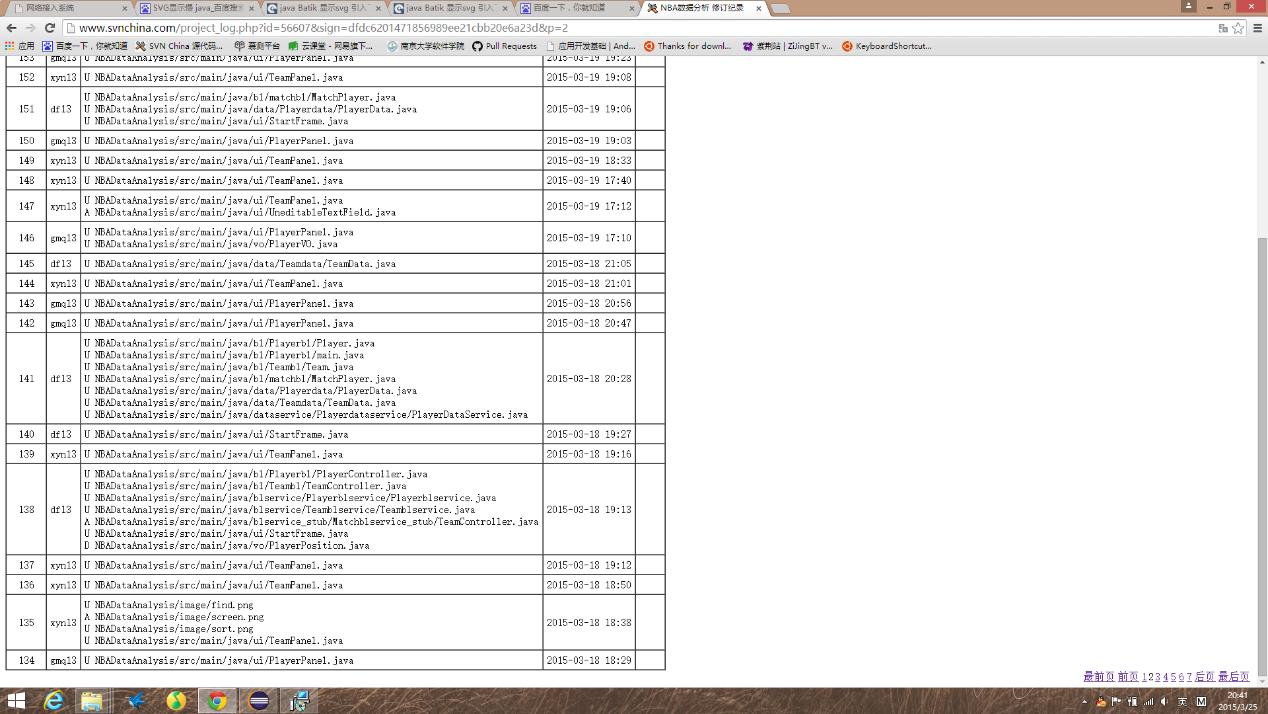


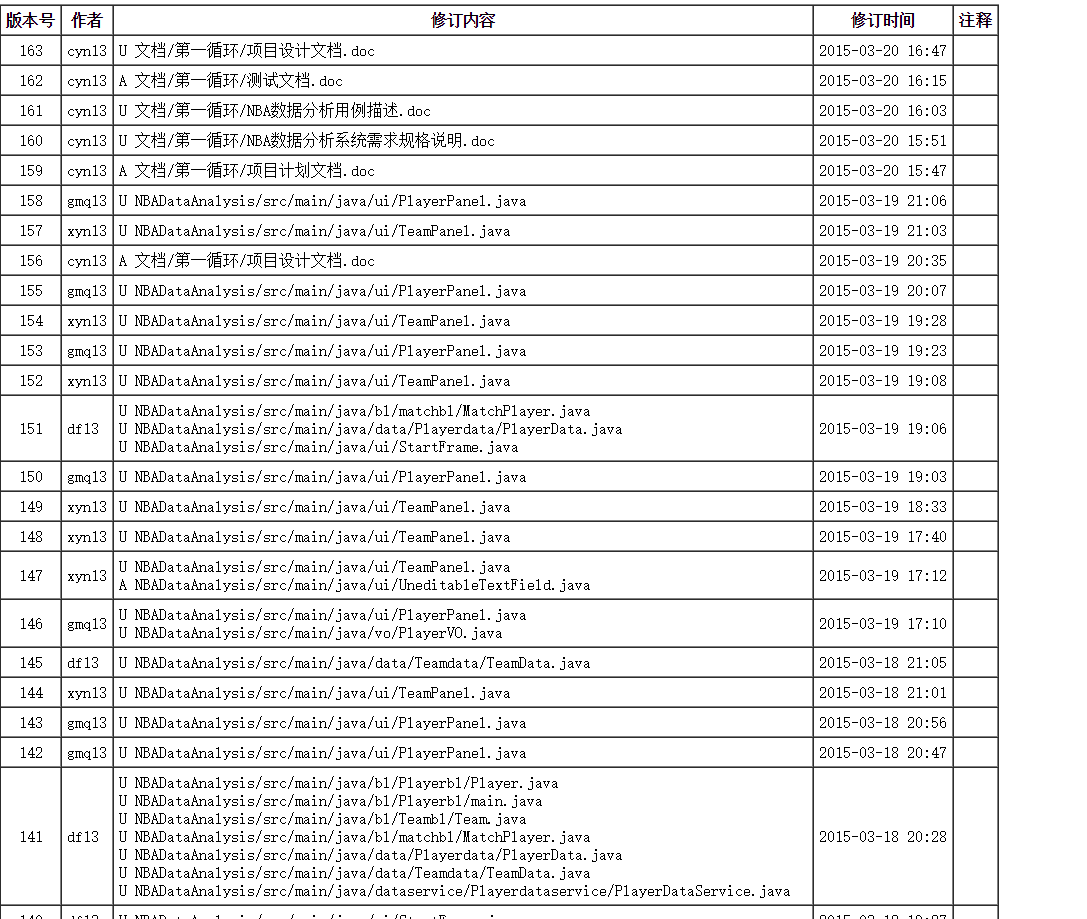


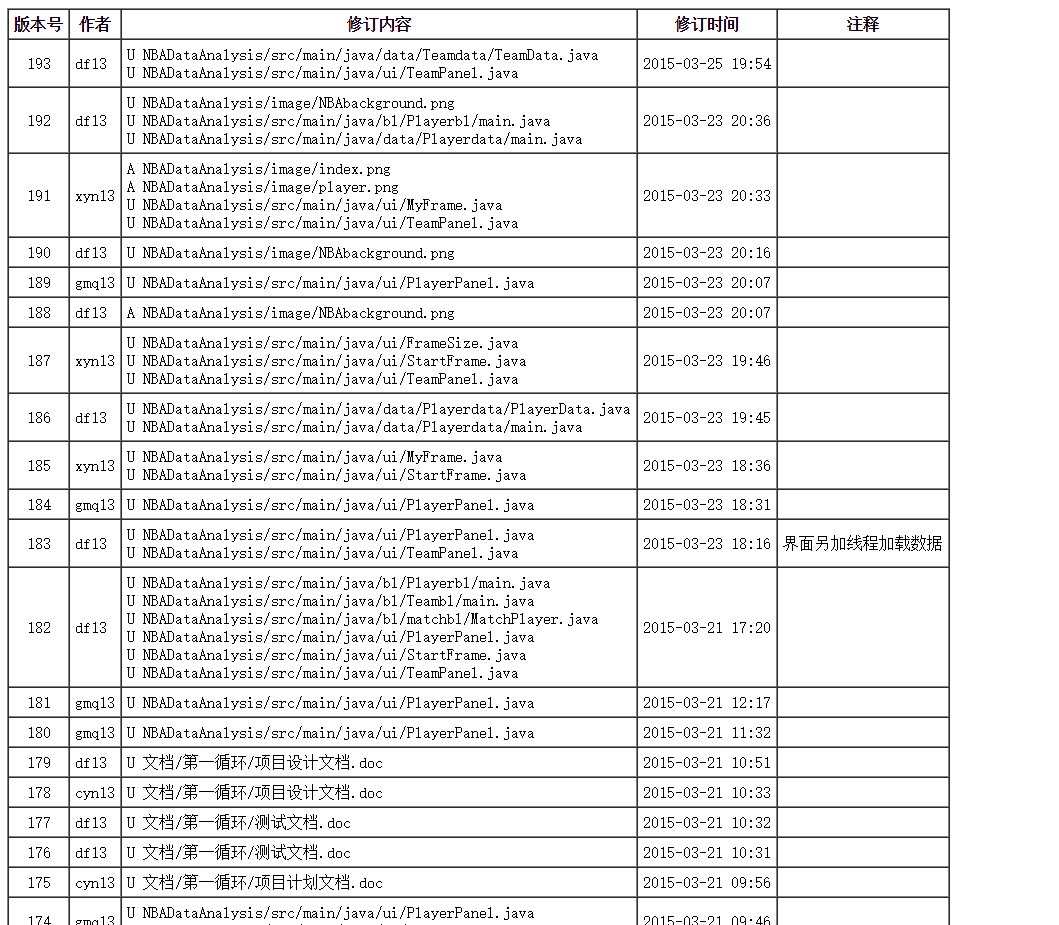




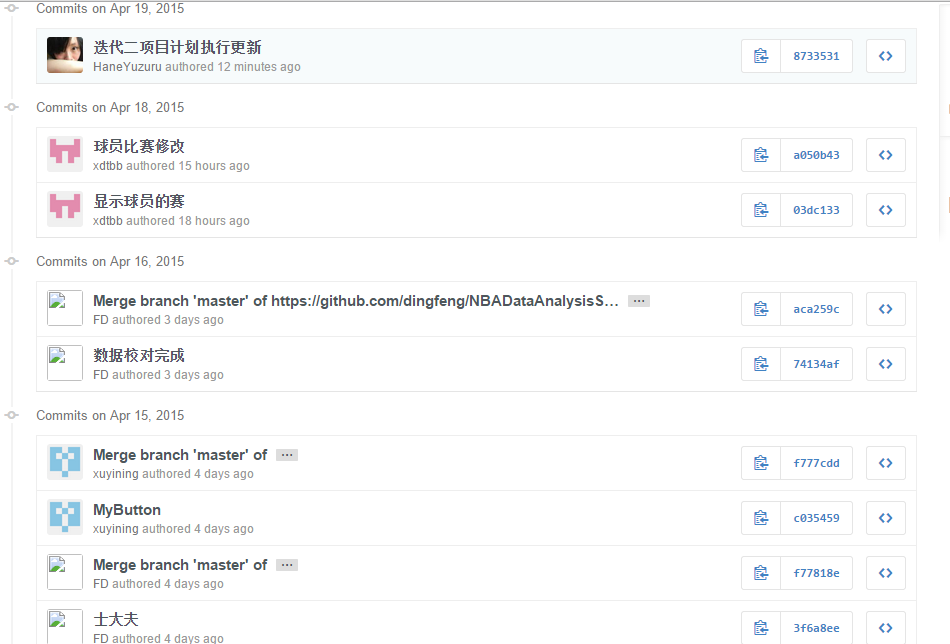


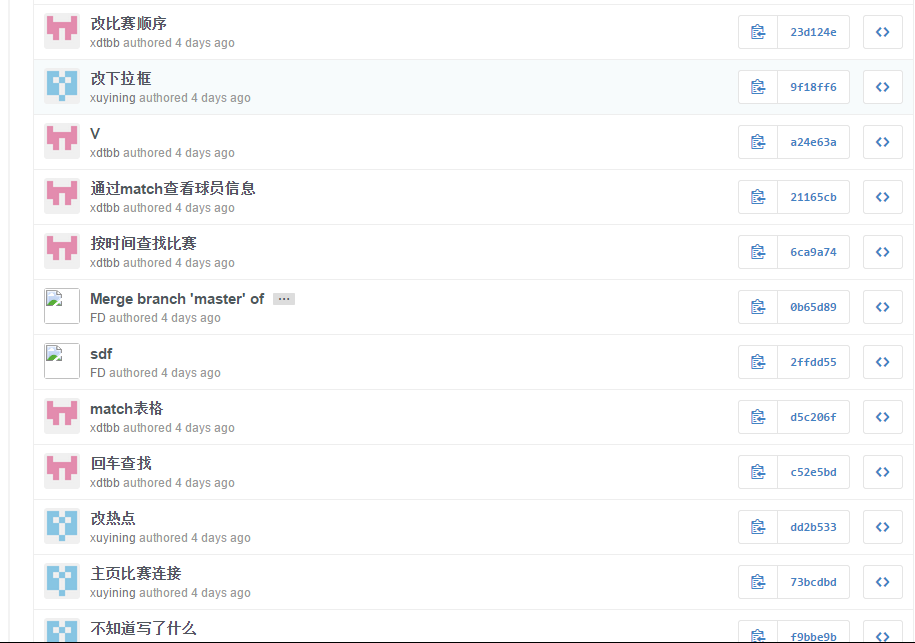


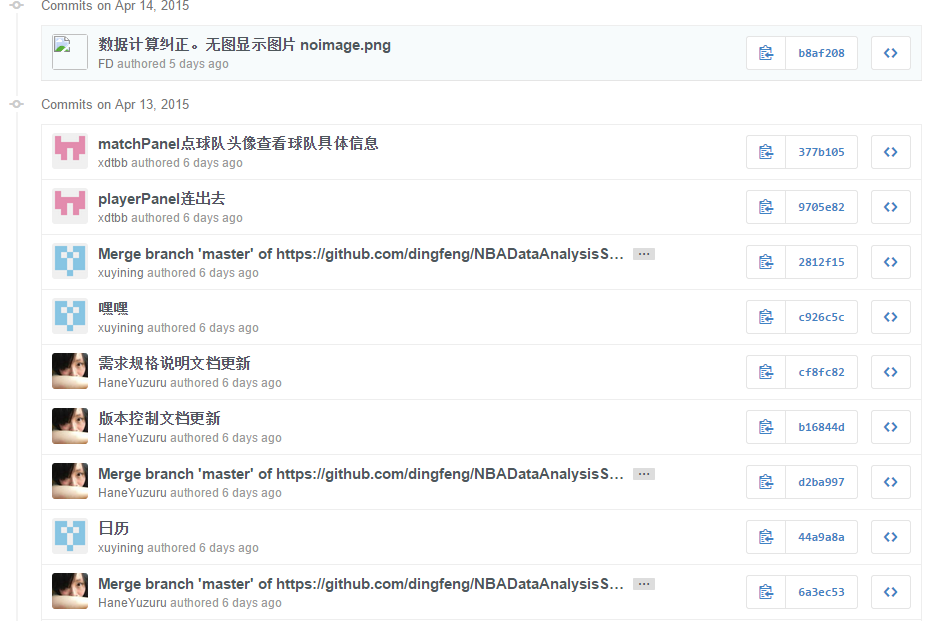


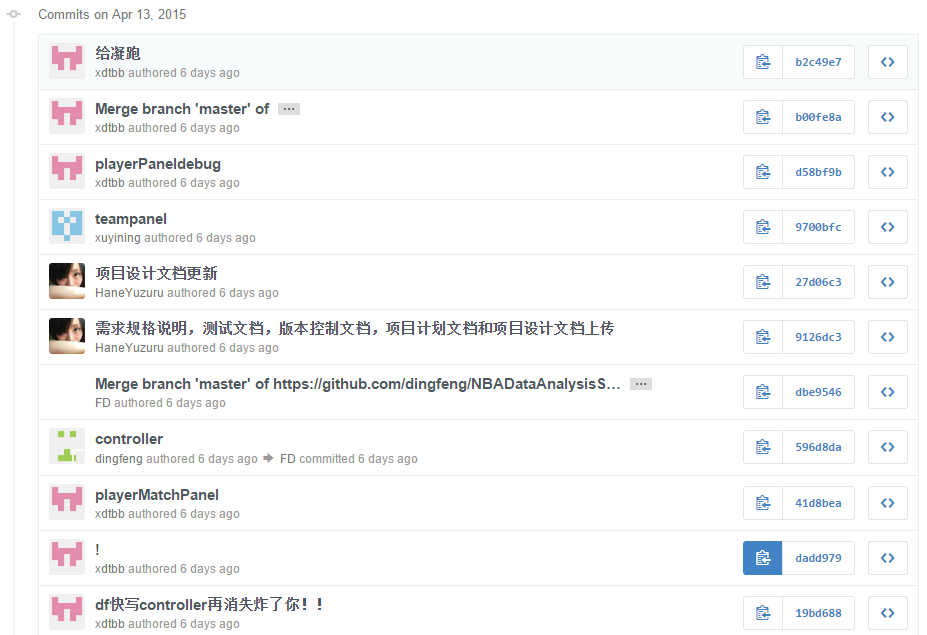


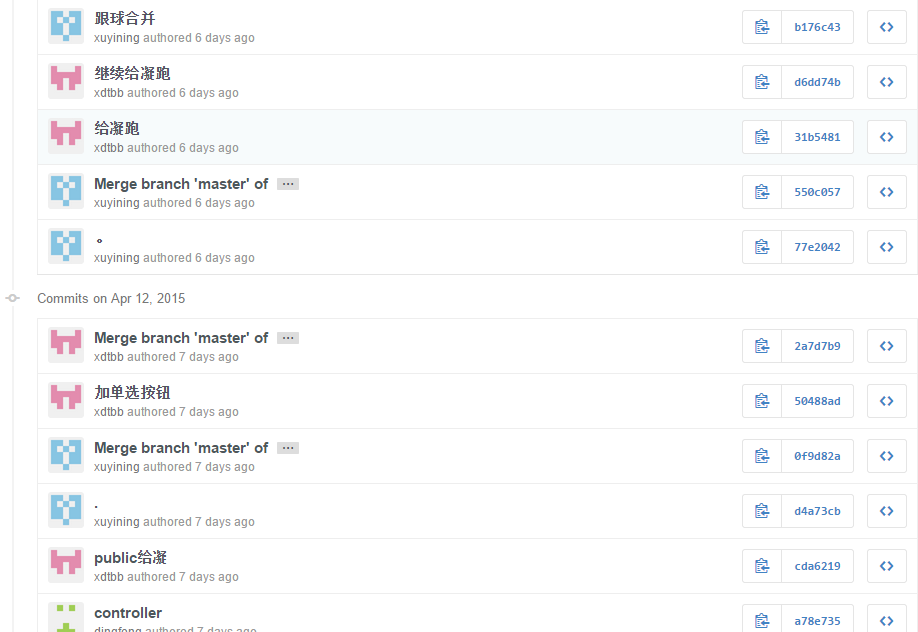
## Git（迭代二）部分

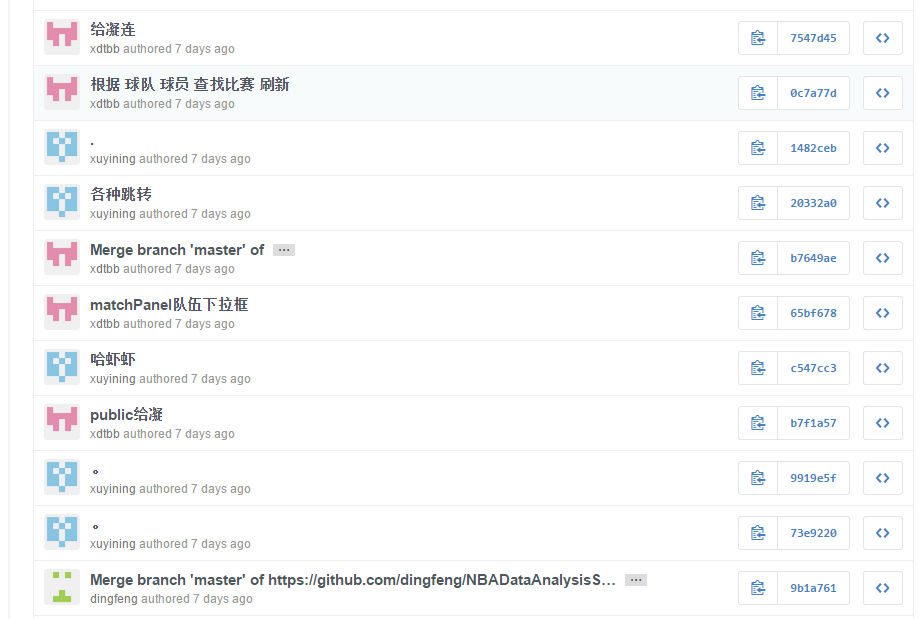


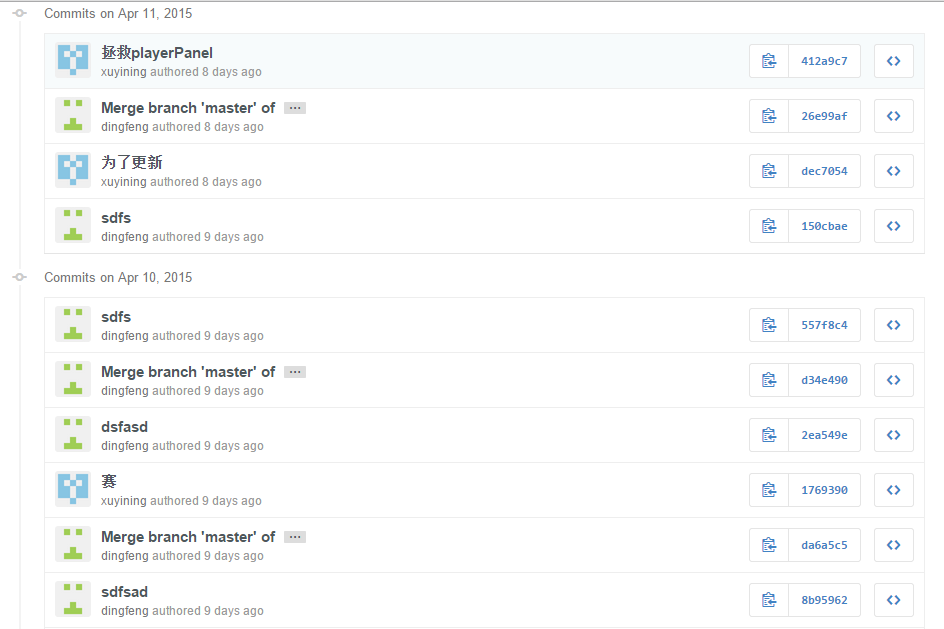


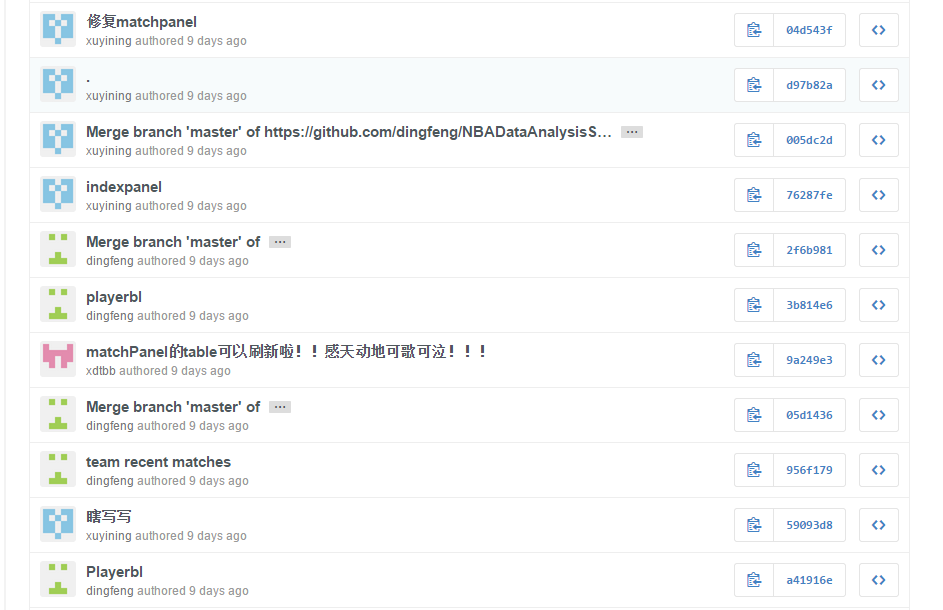


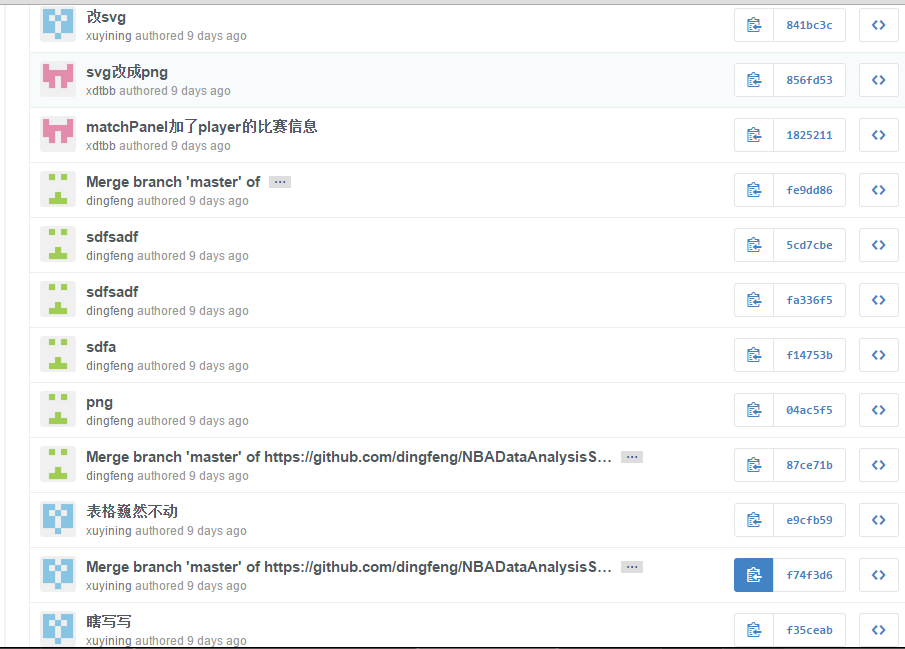


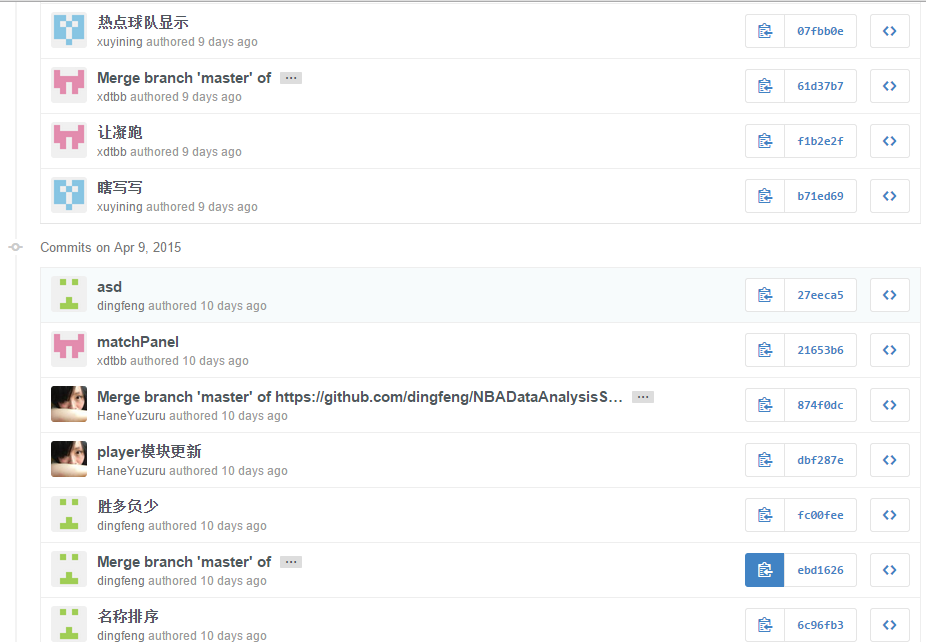


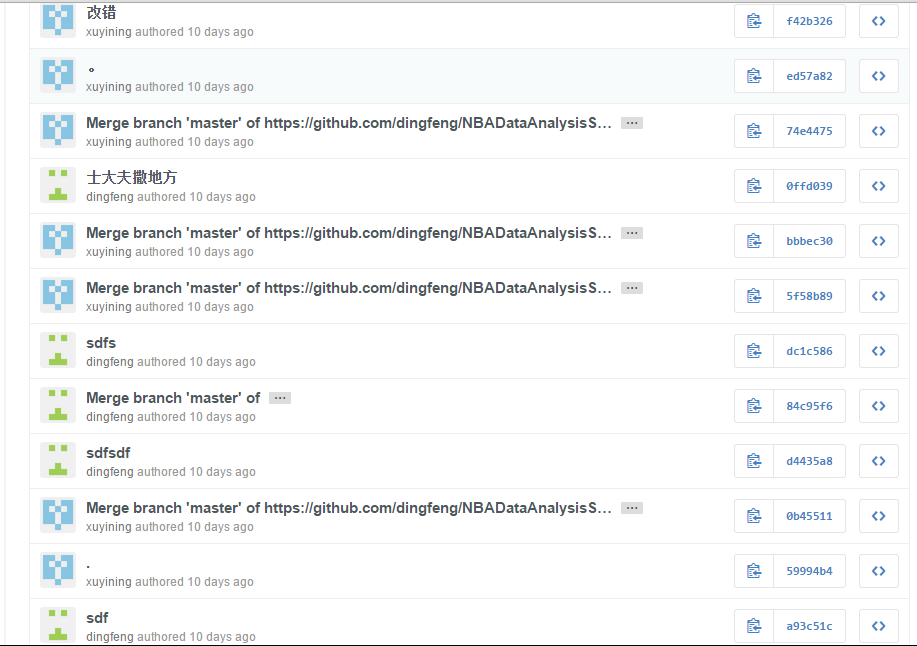


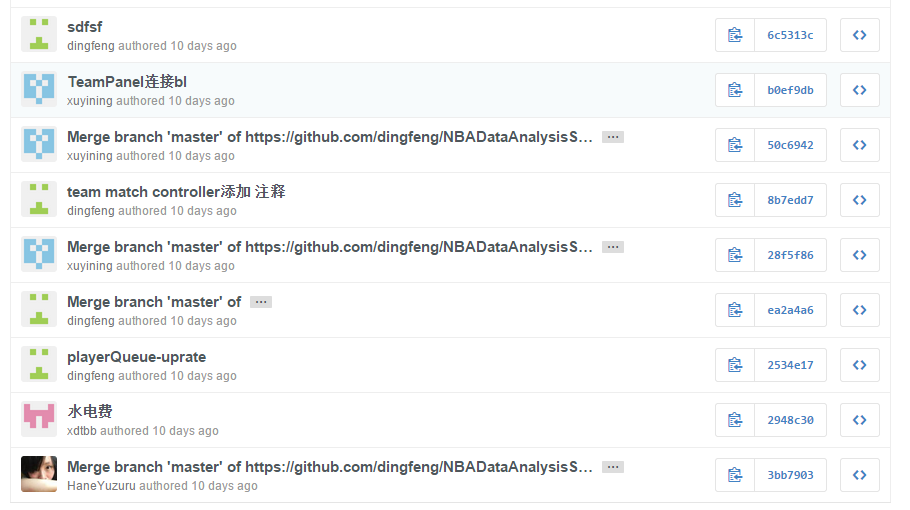


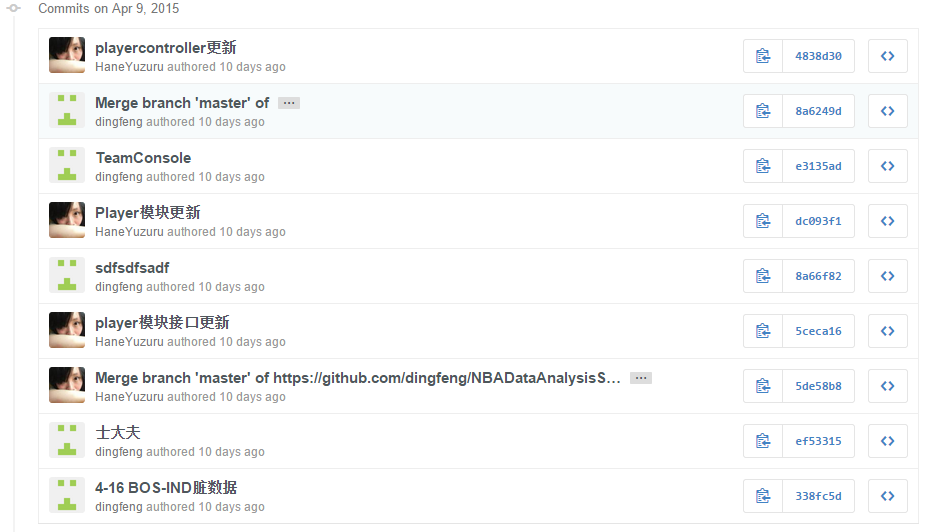


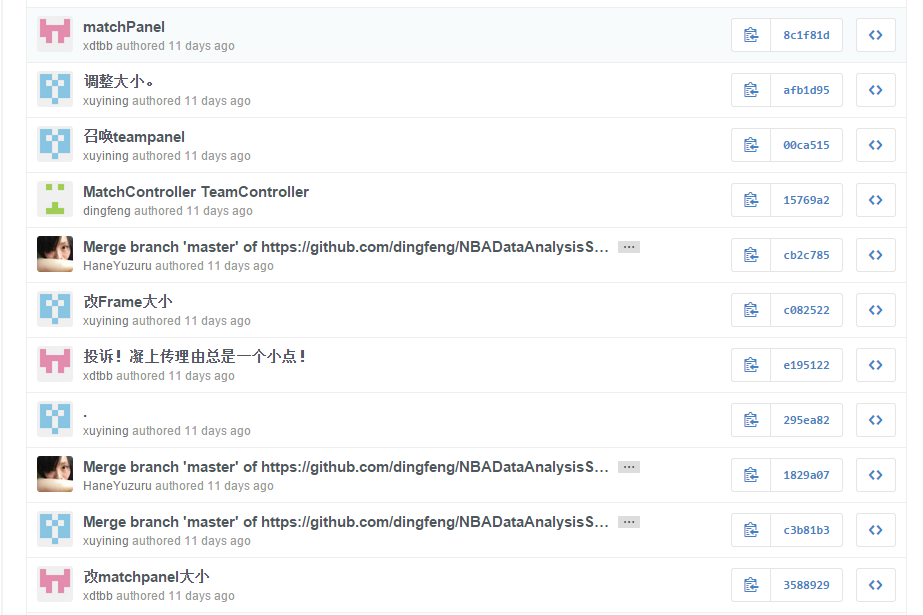


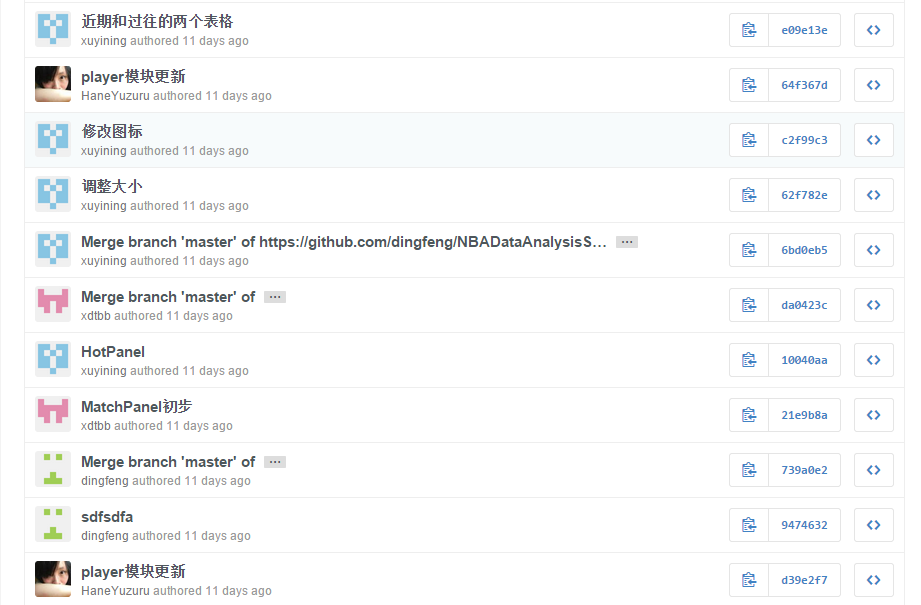


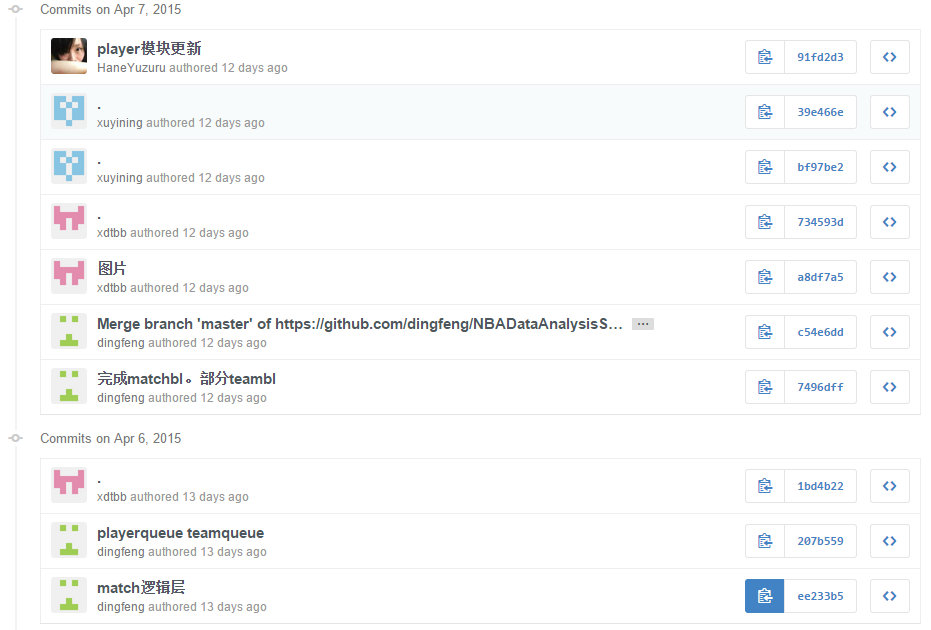


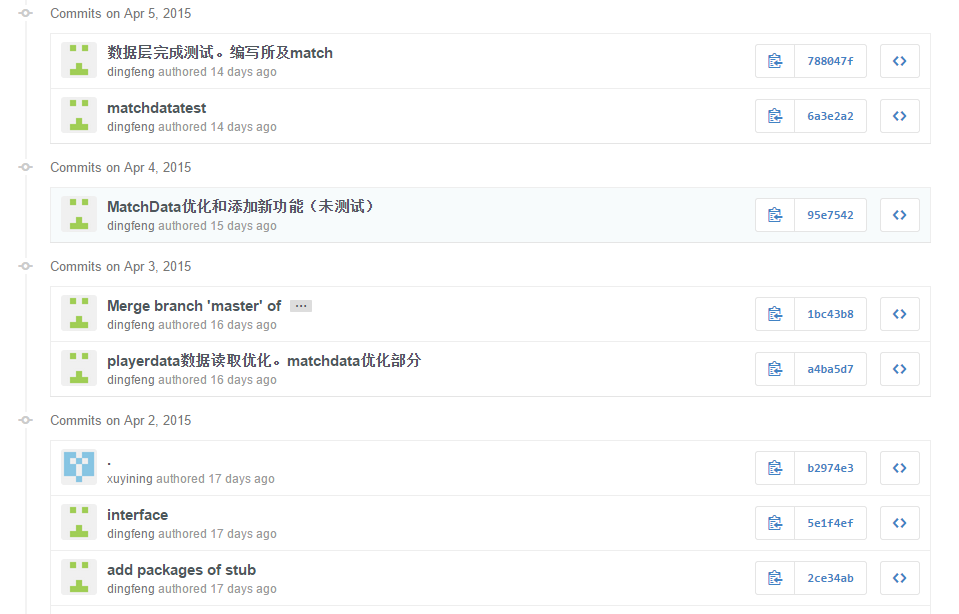


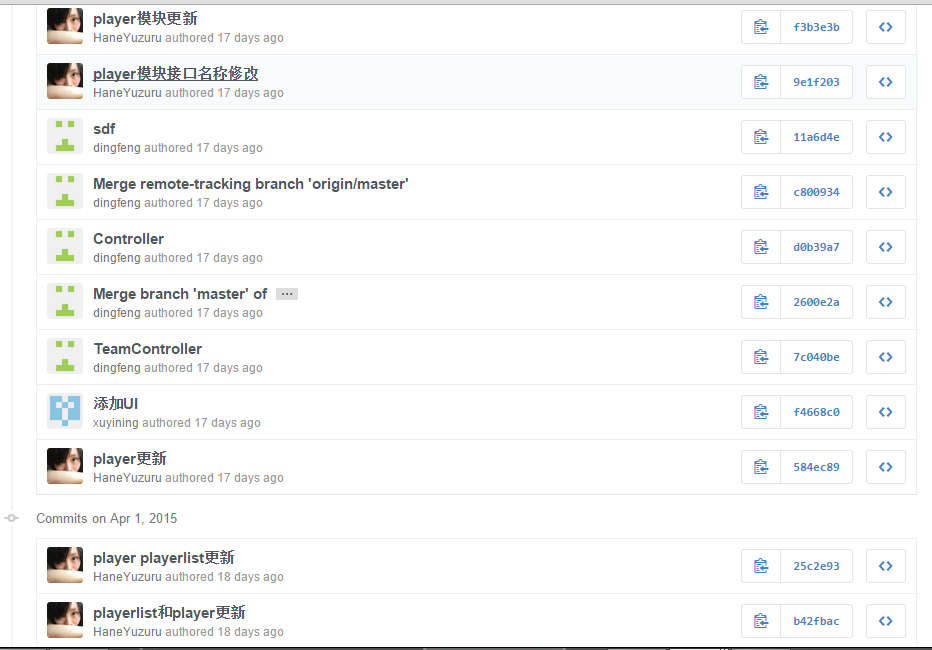


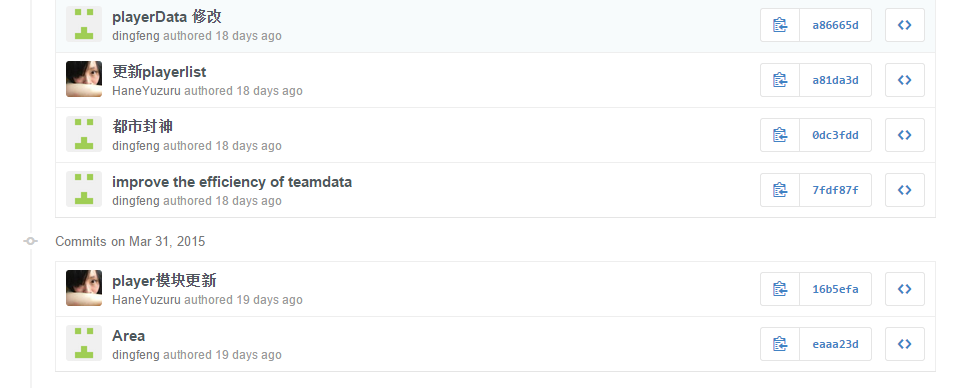


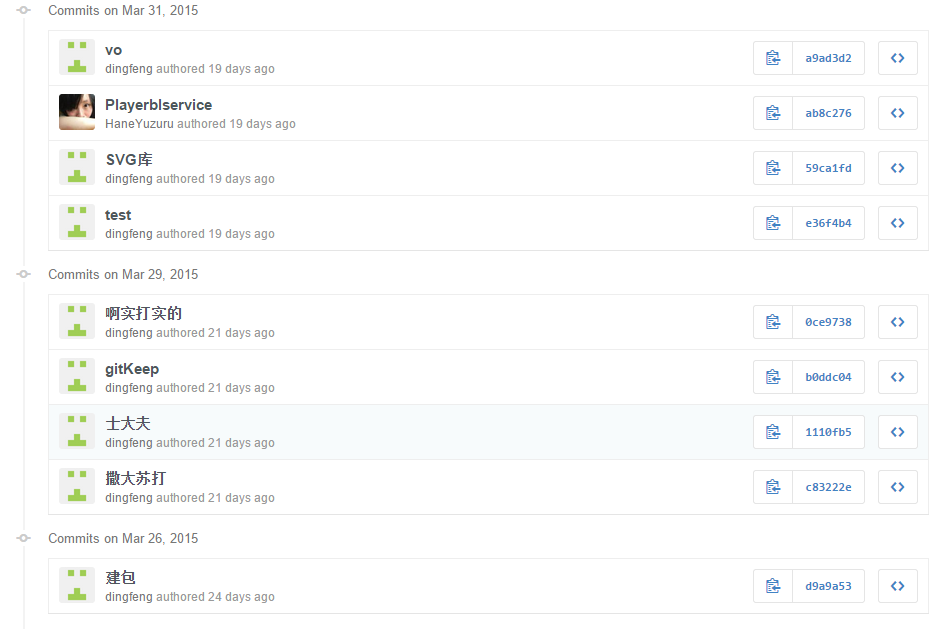




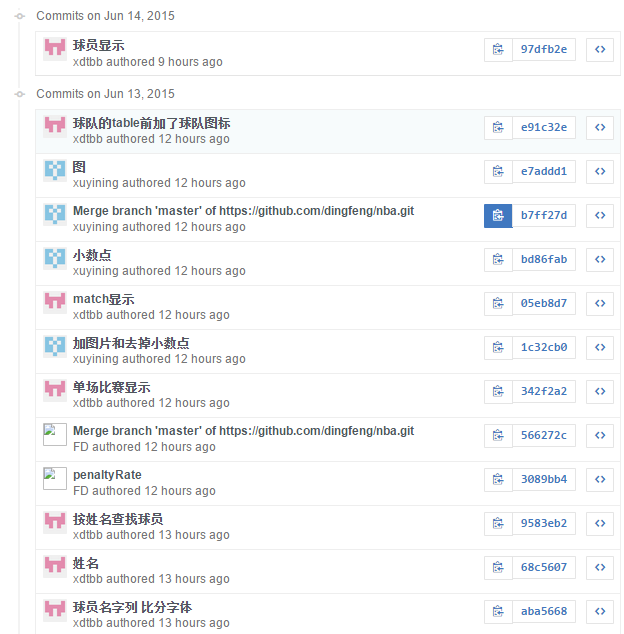


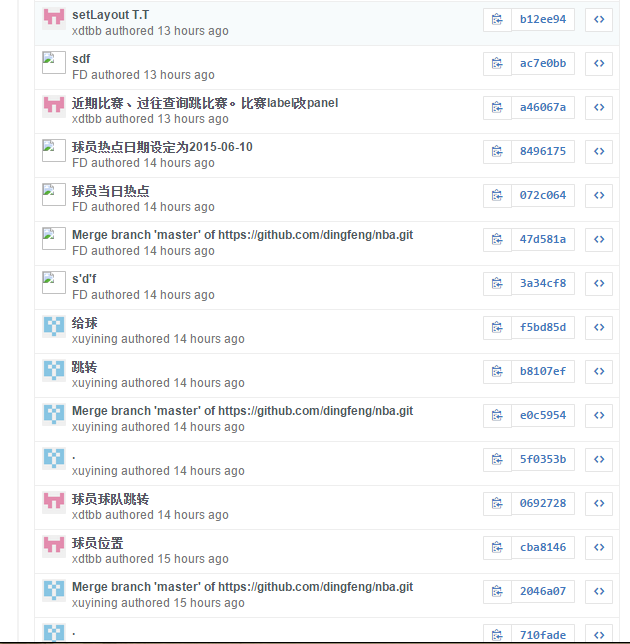


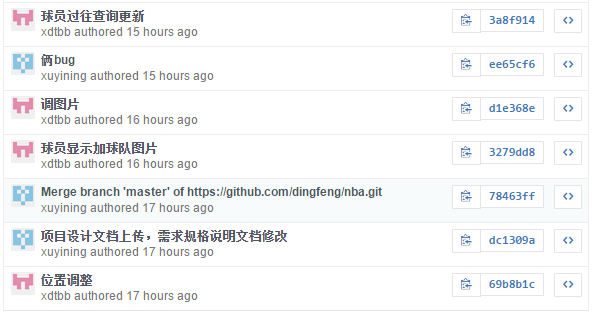


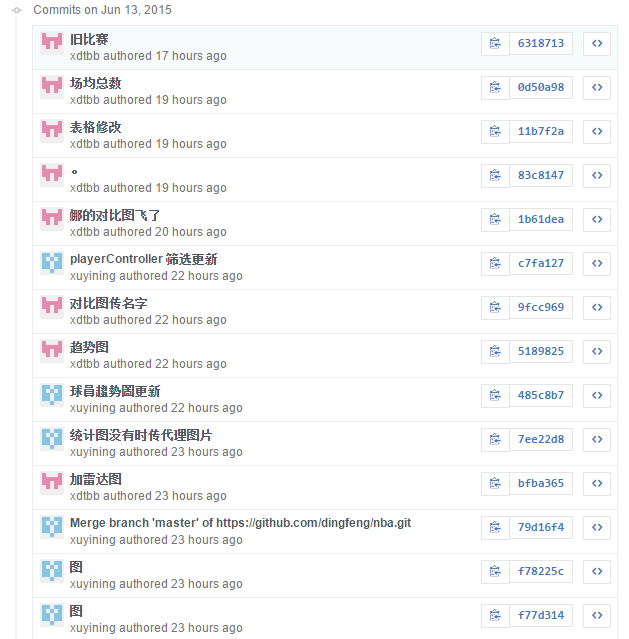


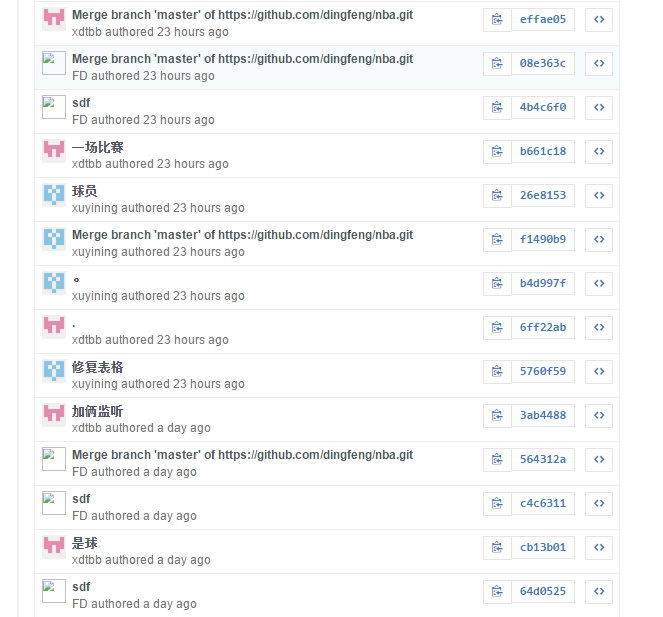
## Git（迭代三）部分

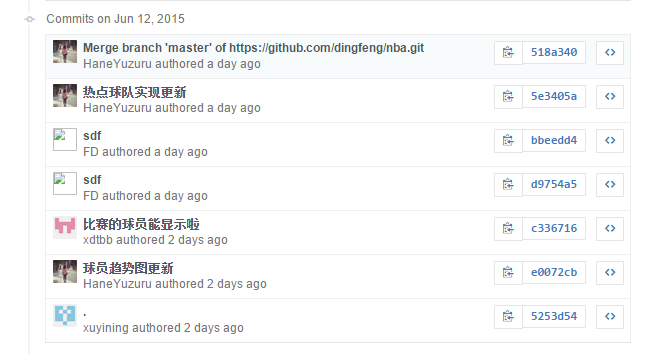




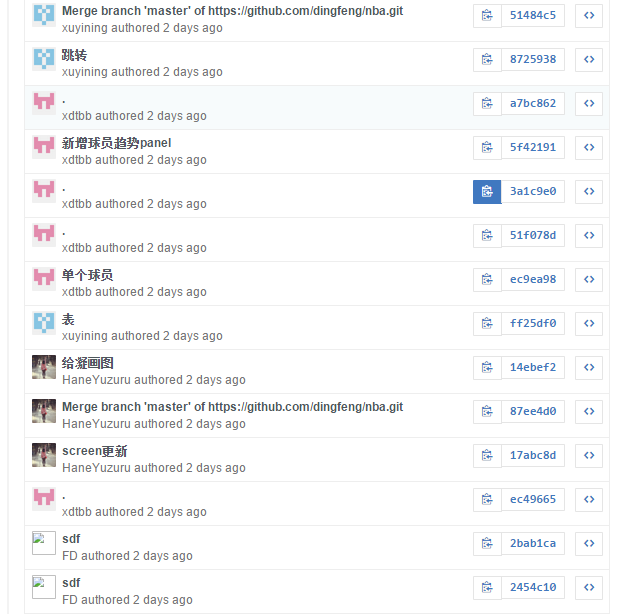


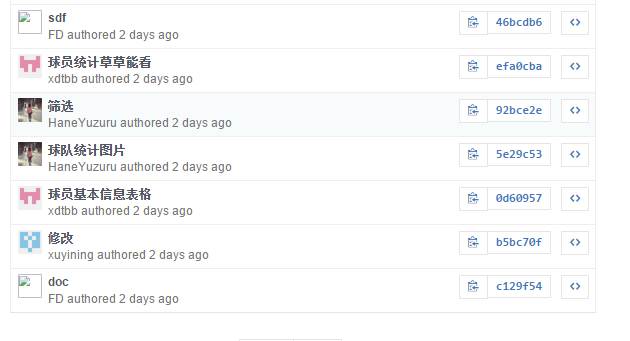


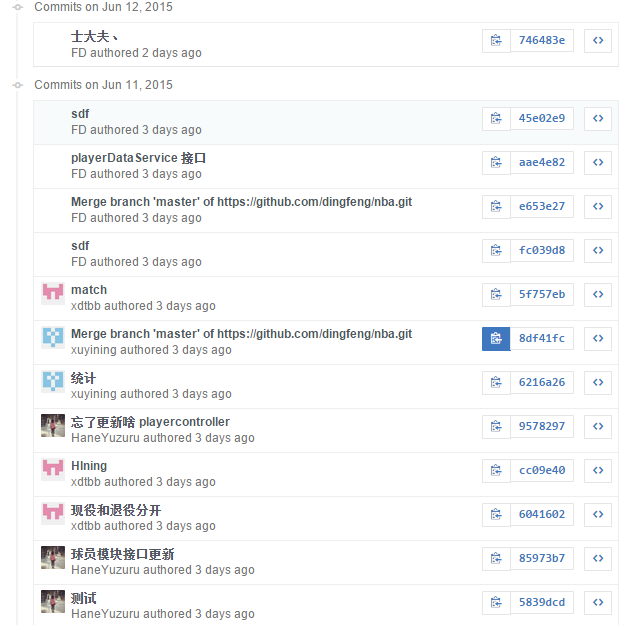


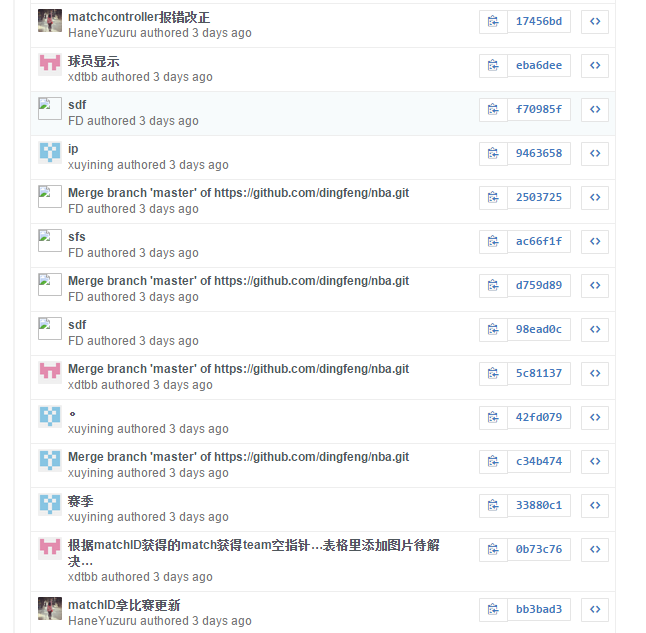


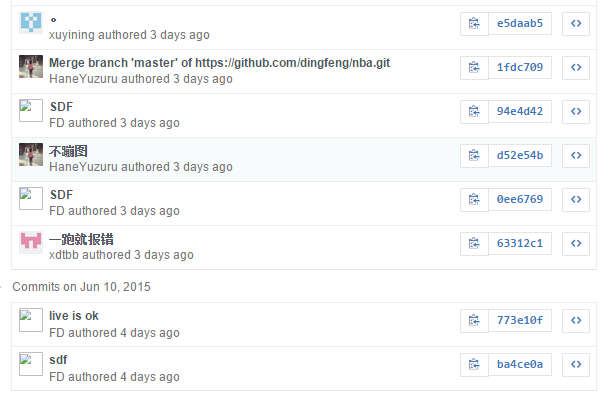


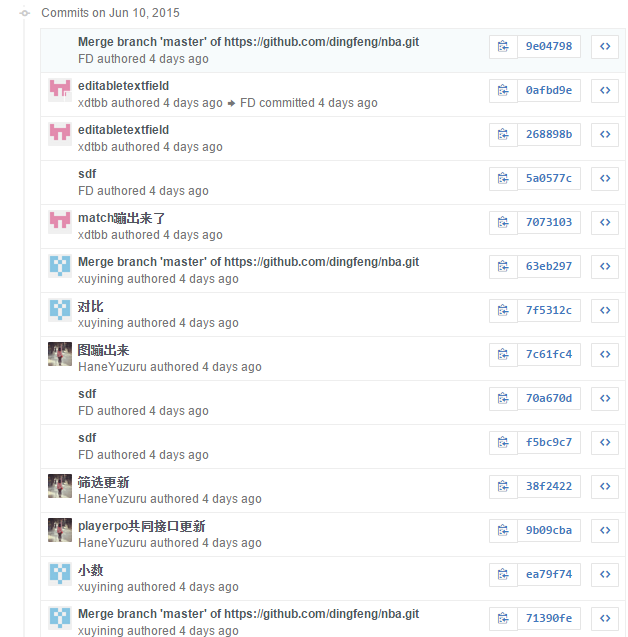


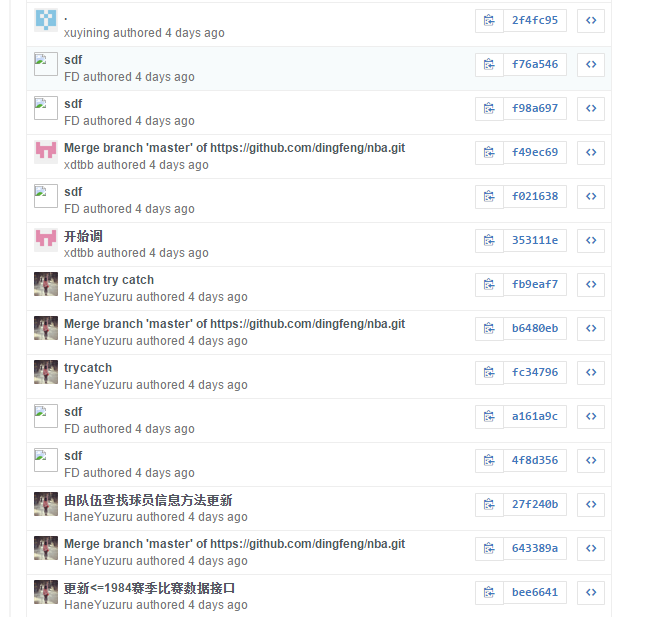


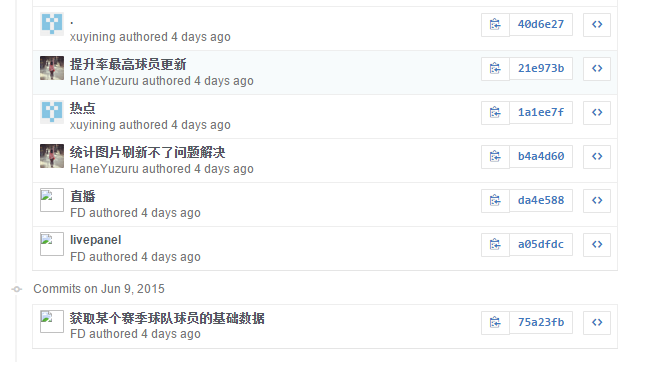


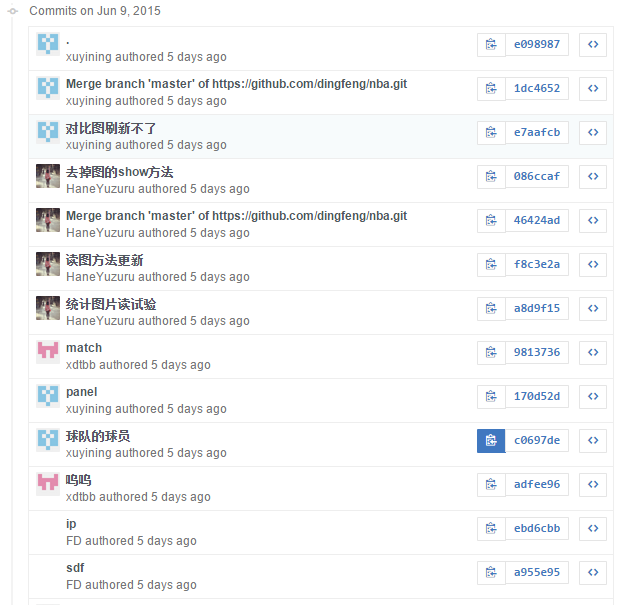






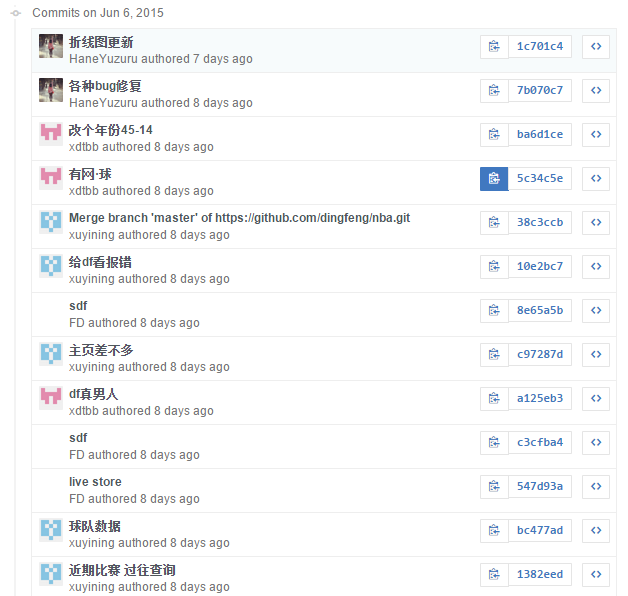


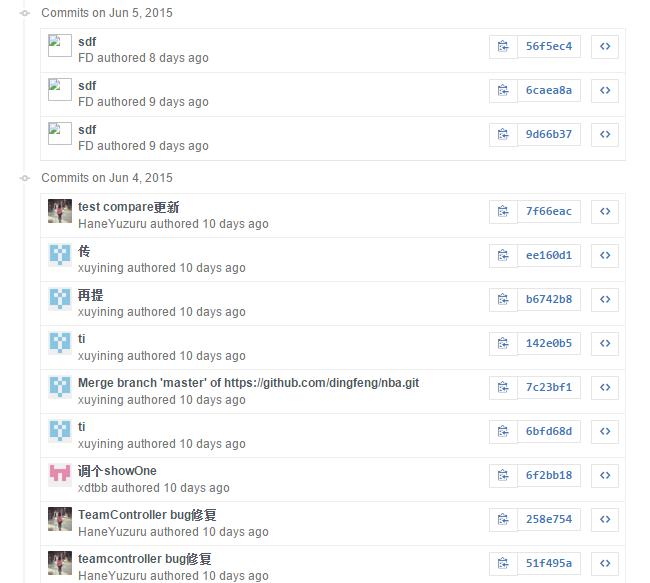


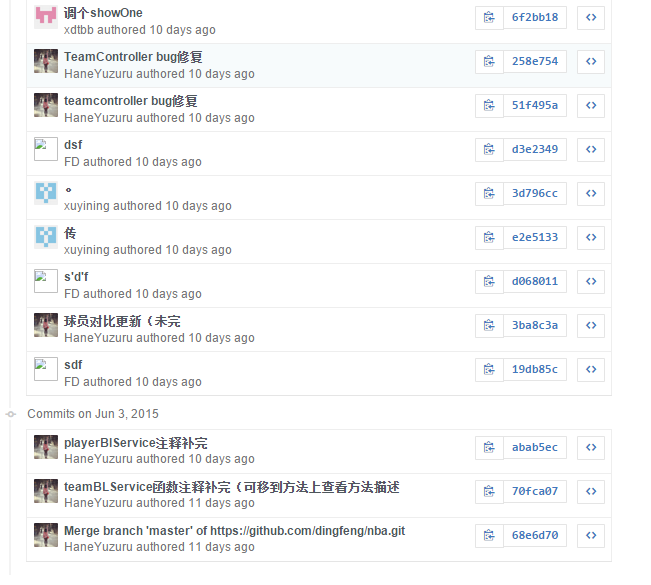


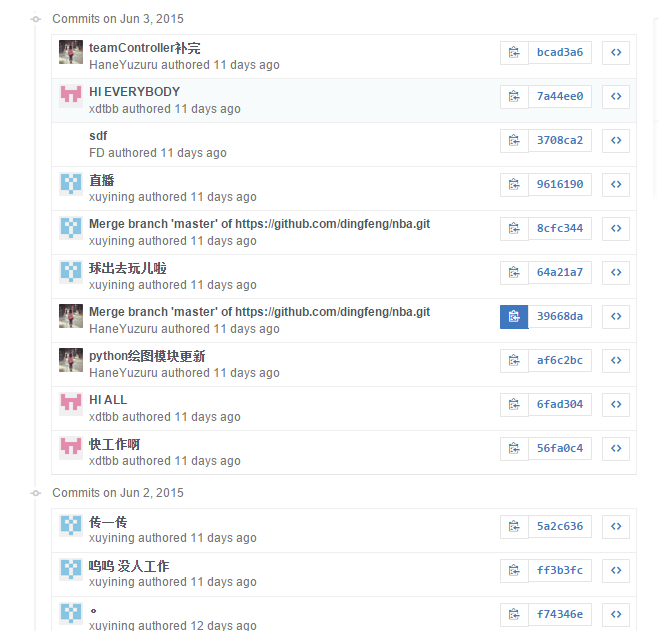


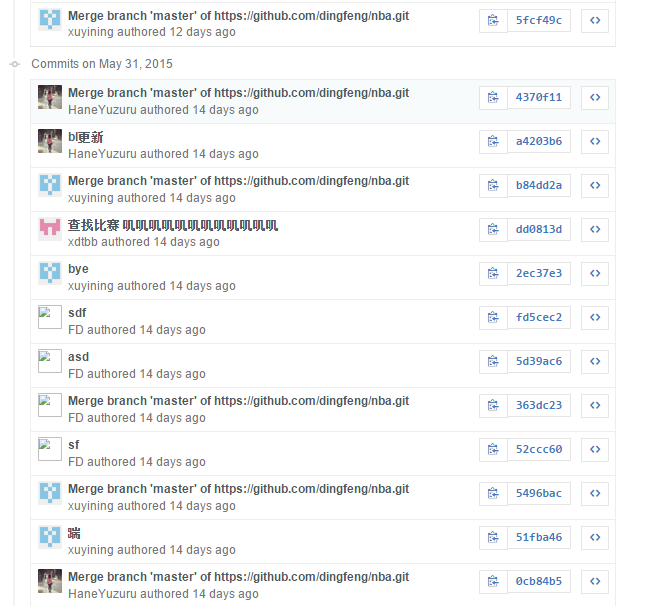


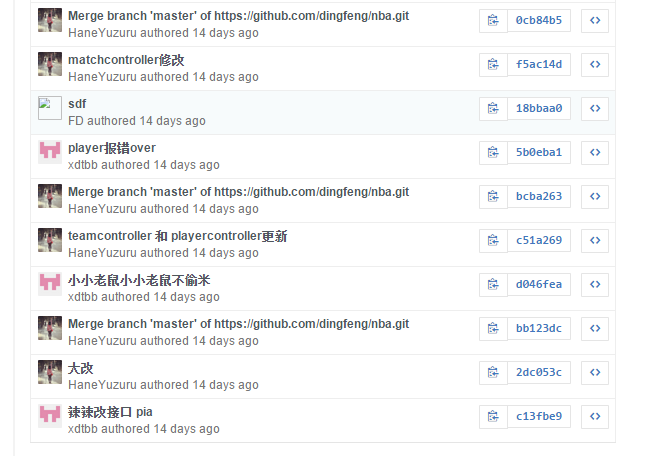






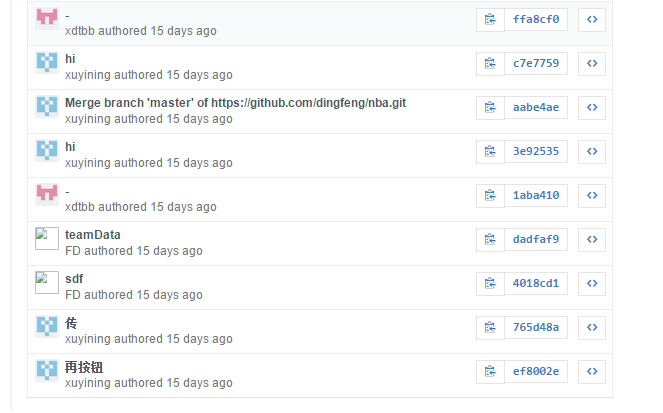


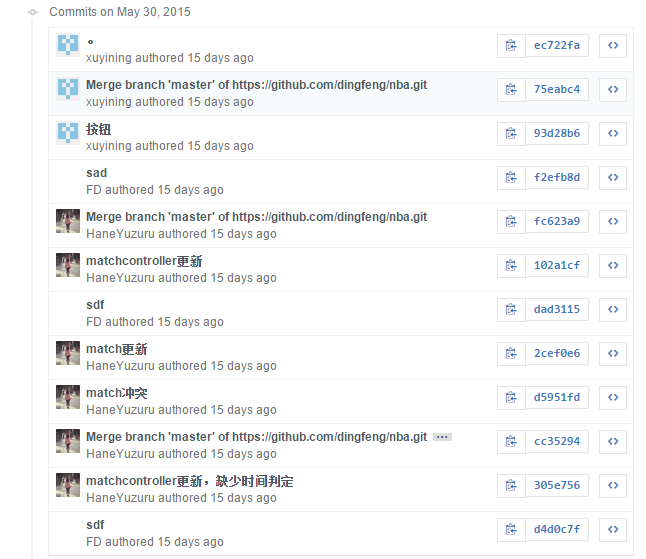


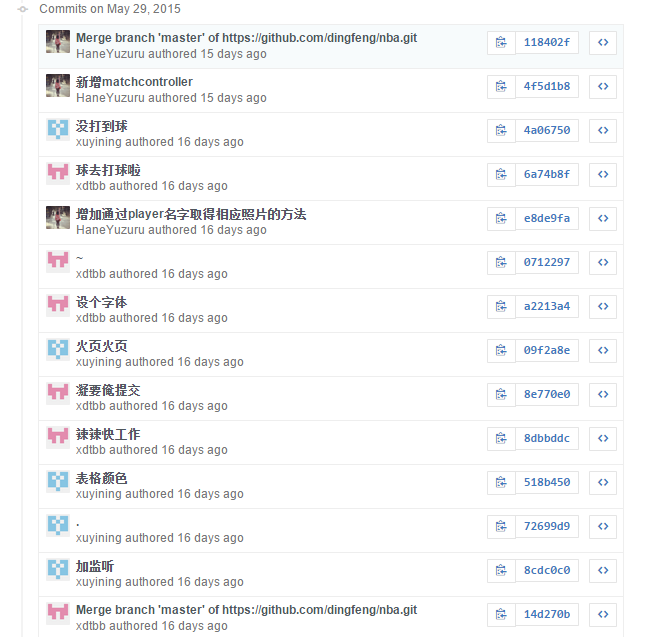






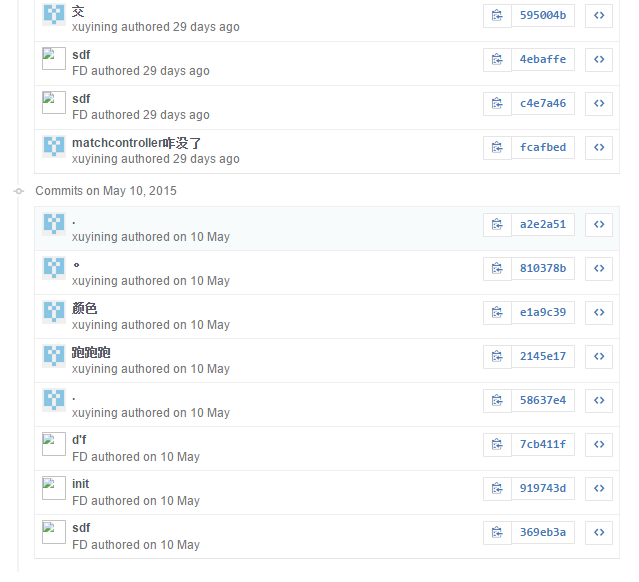






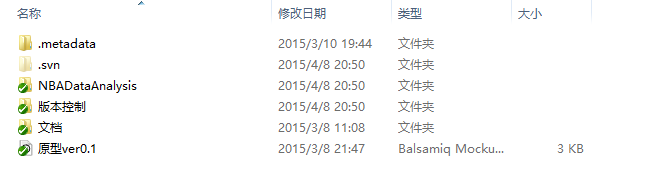






# 最新版本截图

## 迭代一部分截图



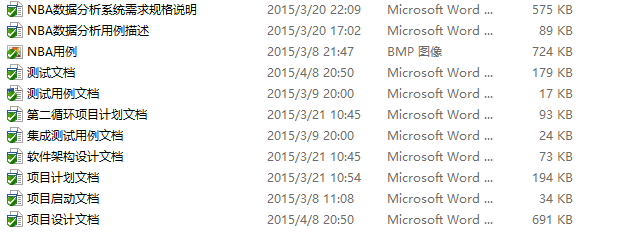
文件说明：

NBADataAnalysis：项目的Java工程，详细介绍请看后图。

版本控制：保存版本控制相关截图

文档：存放项目相关所有文档，详细介绍请看后图

原型ver0.1：项目原型设计草稿



文件说明：

NBA数据分析系统需求规格说明

NBA数据分析用例描述

NBA用例：NBA用例图

测试文档：结合单元测试用例，集合测试用例

测试用例文档：单元测试用例

第二循环项目计划文档

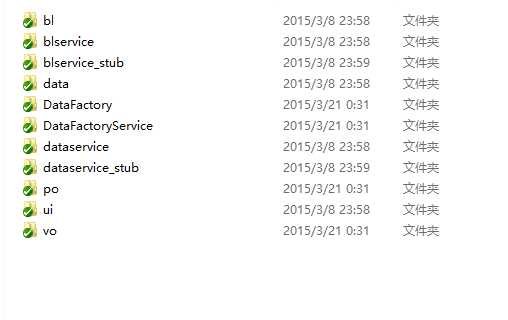
集成测试用例文档

软件架构设计文档

项目计划文档：包括第一循环计划文档，第一循环项目执行文档和第二循环计划文档

项目启动文档

项目设计文档



文件说明：

Bl：逻辑层代码

Blservice：逻辑层接口规范

Blservice\_stub：逻辑层的桩程序

Data：数据层代码

DataFactory：数据层服务的工厂

DataFactoryService：数据层服务工厂的接口规范

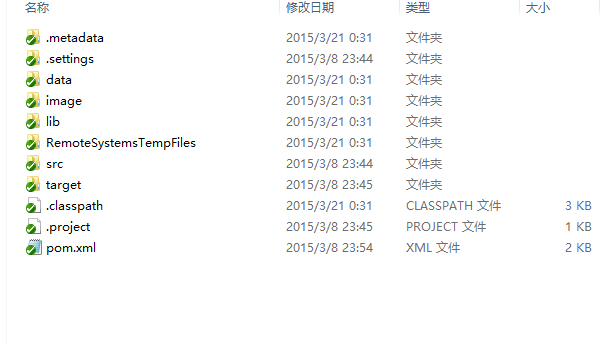
Dataservice：数据层的接口规范

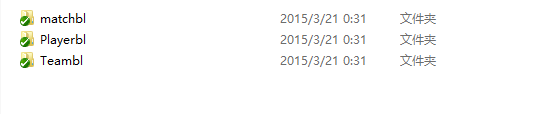
Dataservice\_stub：数据层的接口规范的桩程序

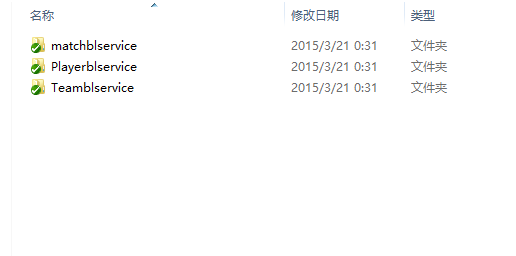
Po：永久化存储的对象

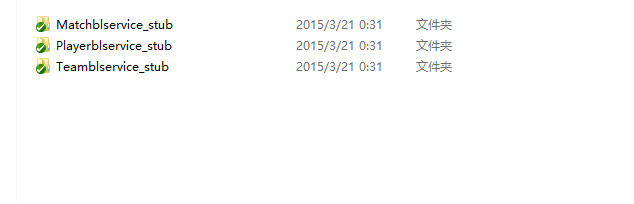
Ui：界面层的代码

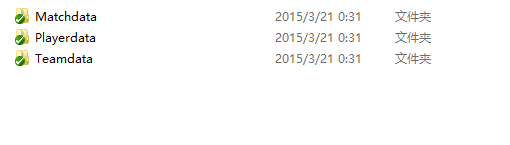
Vo：界面层与逻辑层交换数据的对象

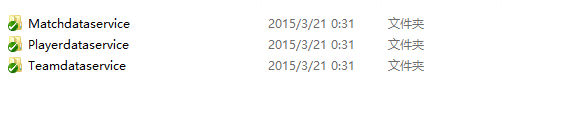


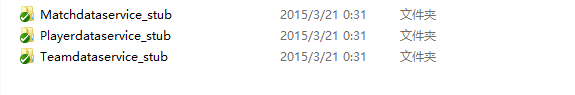


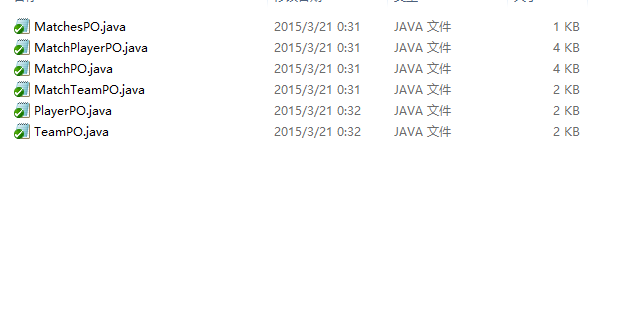








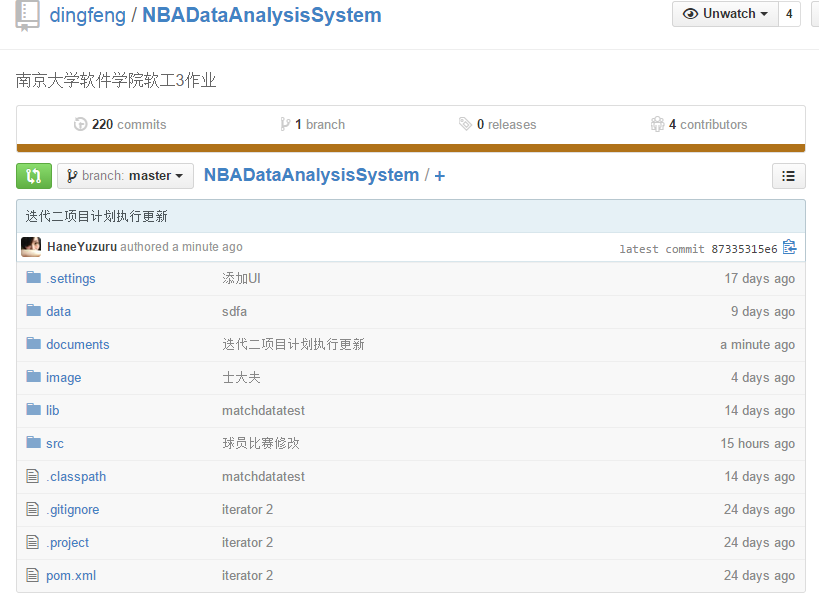








## 迭代二部分截图



文件说明：

Data：项目源数据

Documents：项目的相关文档

Image：项目中图片文件

Src：项目源代码

下面5张图为各个文件夹中文件的具体说明：



文件说明：

Datafactory：数据工厂，负责生成数据层服务

dataFactoryService：数据工厂，产生数据层服务的接口

bl：逻辑层实现

blservice：逻辑层接口

data：数据层实现

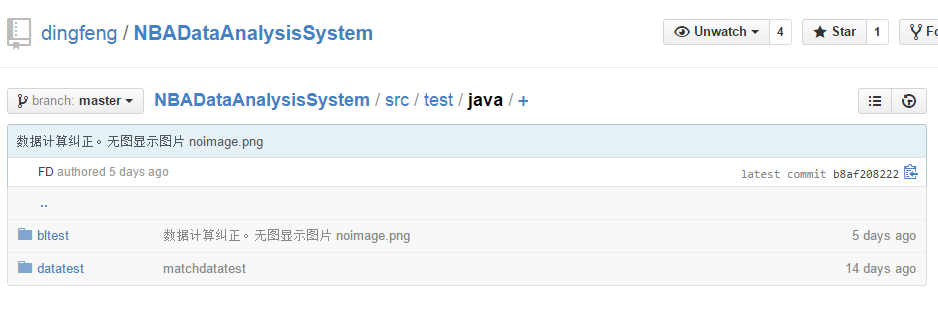
dataservice：数据层接口

po：持久化寸处对象

test：单元测试用例

ui：界面层代码

vo：逻辑层与界面层交换数据的对象

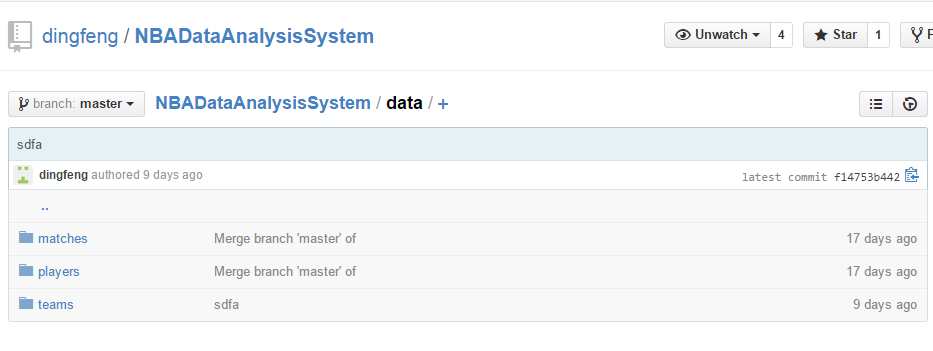


文件说明：

Bltest：逻辑层测试用例

Datatest：数据层测试用例



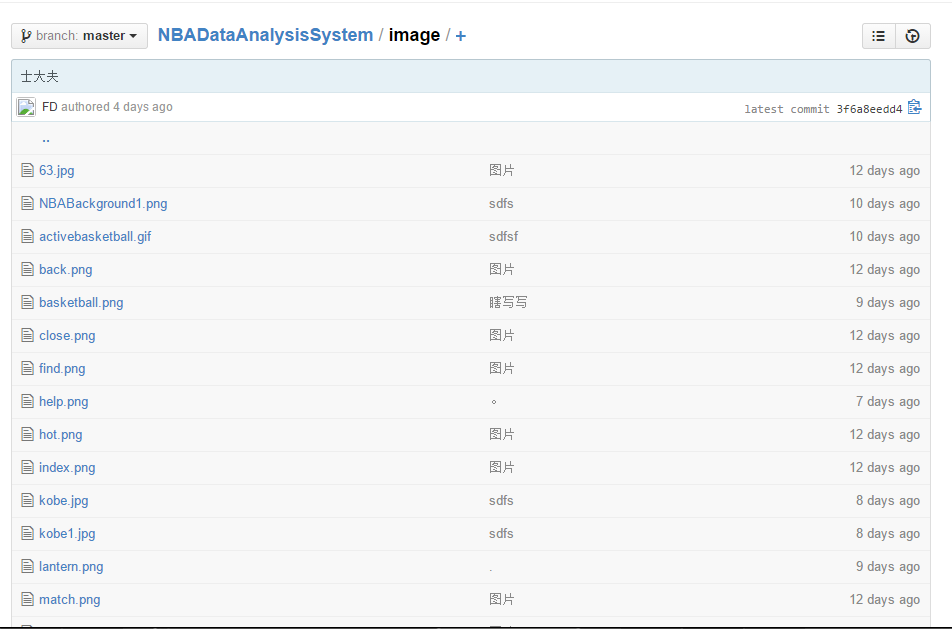


文件说明：

Matches：保存了比赛信息

Players：保存了球员信息

Teams：保存了球队信息



## 迭代三部分截图



文件说明：

Documents：所有项目相关文档

Doc：项目的API文档

Image：项目相关的图片

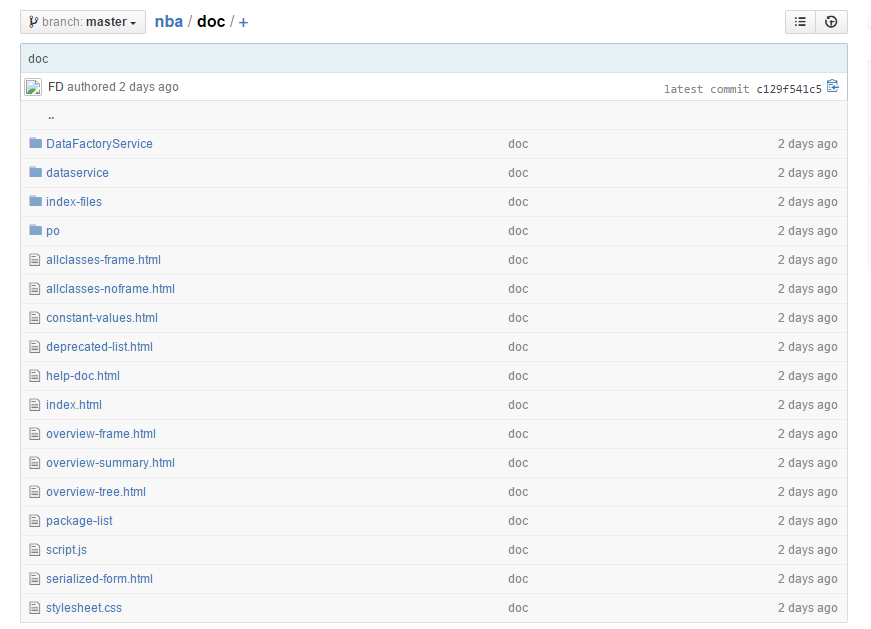
Lib：项目相关的lib

Playerdataservice：playerdataservice接口说明

Python：项目相关的python文件

Src/main/java：项目Java源代码

数据采集：项目数据采集模块的基本说明



文件说明：

项目的API文档



文件说明：

\_\_init\_\_.py：球员对比图

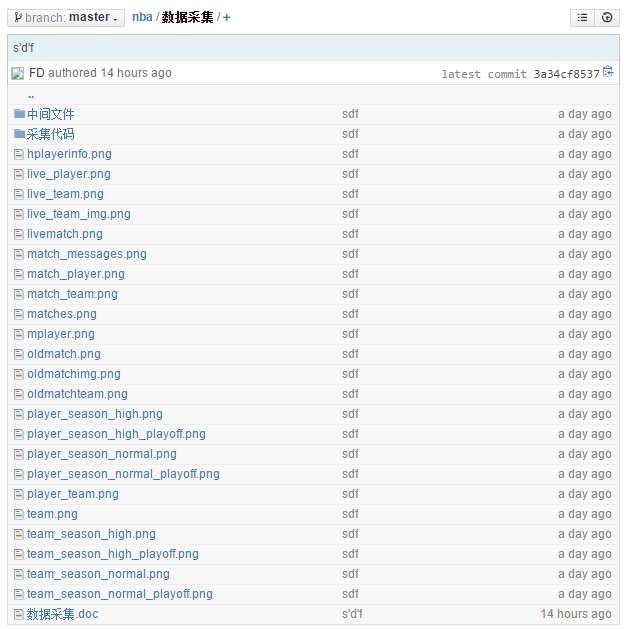
Errorbar.py：球员趋势图

playerBar.py：球员与所有球员场均数据对比图

radar.py:球员低阶数据雷达图

teamBar.py:球队与场均数据对比图

teamCompare.py：球队与球队对比图



文件说明：

数据采集.doc：详细说明数据采集的原理和数据库设计

中间文件：数据采集过程中的中间的代码

采集代码：数据采集实例



文件说明：

DataFactory:数据层工厂，负责初始化player，team，match的数据层服务

dataFactoryService:数据层工厂的接口

bl:逻辑层

blservice:逻辑层服务接口

dataservice:数据层服务接口

po:数据层数据持久化对象

test:所有JUnit测试用例

ui:界面层

vo:逻辑层与界面层交互的对象

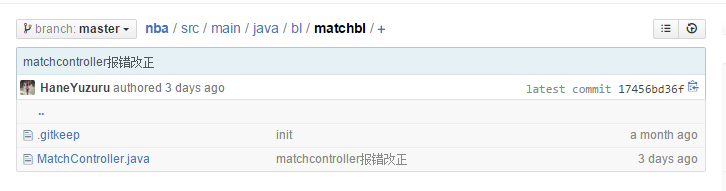


文件说明：

Matchbl：比赛逻辑层服务

Playerbl：球员逻辑层服务

Teambl：球队逻辑层服务



文件说明：

matchController：负责提供所有比赛的服务



文件说明：

playerController：负责提供所有球员的逻辑层服务

PlayerBlTest：球员逻辑层JUnit单元测试



文件说明：

TeamController：负责提供所有球队的逻辑层服务

teamTest：球队逻辑层JUnit单元测试



文件说明：

Matchblservice：比赛的逻辑层接口

Playerblservice：球员的逻辑层接口

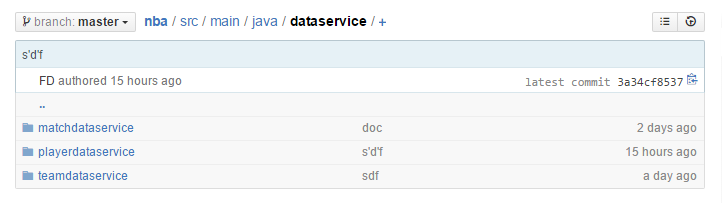
Teamblservice：球队的逻辑层接口

所有文件夹中均有一个对应接口java文件如下图：









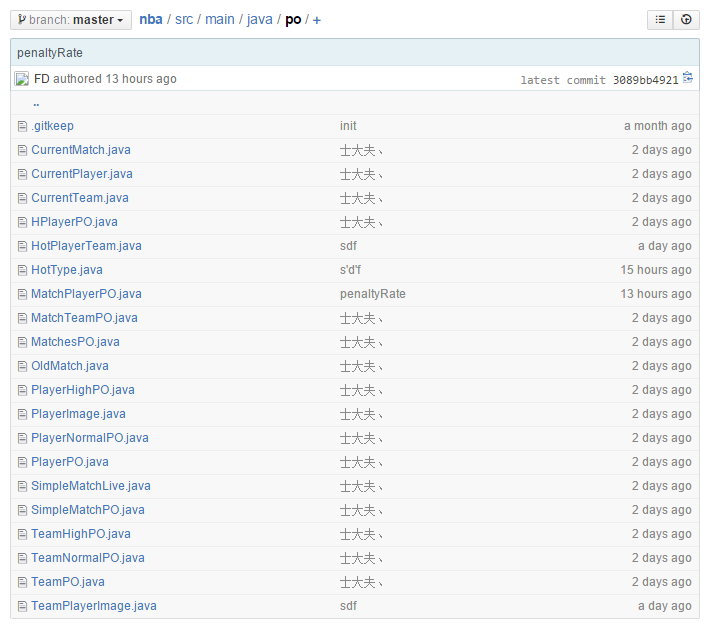
文件说明：

Matchdataservice：比赛的数据层接口

Playerdataservice：球员的数据层接口

Teamdataservice：球队的数据层接口

所有文件夹中均有一个对应接口java文件故不截图



文件说明：

CurrentMatch：正在直播的比赛

CurrentPlayer：正在直播的球员

CurrentTeam：正在直播的球队

HPlayerPO：历史球员基本信息对象

HotPlayerTeam：热点球员/球队的对象

HotType：热点球员/球队的排序依据

MatchPlayerPO：一场比赛中球员的比赛数值

MatchTeamPO：一场比赛中球队的比赛数值

MatchesPO：一场比赛的信息

OldMatch：1984年前的比赛对象

PlayerHighPO：球员的高阶数据

PlayerImage：球员的图片

PlayerNormalPO：球员的低阶数据

PlayerPO：现役球员的基本信息对象

SimpleMatchLive：简略的比赛直播

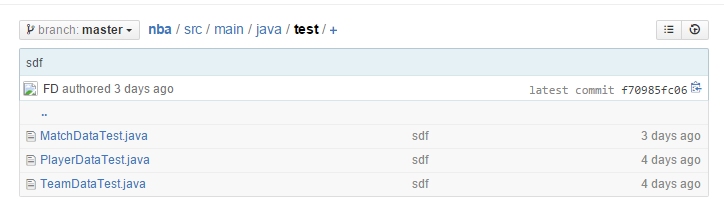
SimpleMatchPO：简略的比赛持久化对象

TeamHighPO：球队高阶数据持久化对象

teamNormalPO：球队的低阶数据持久化对象

teamPO：球队的基本信息持久化对象

TeamPlayerImage：球员球队的图片持久化对象

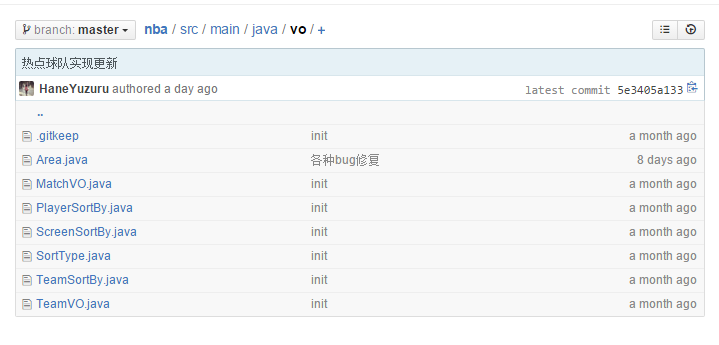


文件说明：

matchDataTest：比赛的数据层单元测试

playerDataTest：球员的数据层单元测试

TeamDataTest：球队的数据层单元测试

文件说明：

Area：分区的枚举类

MatchVO：比赛的vo对象

PlayerSortBy：球员排序依据的枚举类

ScreenSortBy：筛选的排序依据的枚举类

SortType：排序依据的枚举类

TeamSortBy：球队的排序依据的枚举类

TeamVO：一个球队的vo对象



文件说明：

Mainui：主界面

Playerui：球员界面

Statistics：统计界面

Teamui：球队界面

HotPanel：热点界面

IndexPanel：统计展示界面

LivePanel：直播界面

OneNowMatch：现在比赛界面

OneOldMatch：一个1984年前比赛的界面