



# **Command Parser-1.1**

## **User Manual**

**Liubin Zhang, Yangyang Yuan, Wenjie Peng**

# 目錄

## 项目简介

Command Parser 简介	1.1
命令行界面	1.2
下载与安装	1.3
API 文档	1.4

## 快速入门

使用图形设计	2.1
启动图形设计界面	2.1.1
添加参数项	2.1.2
添加参数规则	2.1.3
设置解析器全局属性	2.1.4
参数解析测试	2.1.5
导出解析器	2.1.6
导入解析器进行重编辑	2.1.7
使用 Java 脚本设计	2.2
初始化解析器	2.2.1
设置解析器的全局属性	2.2.2
添加参数组	2.2.3
添加参数项	2.2.4
设置参数项属性	2.2.5
设置参数规则	2.2.6
格式化解析器	2.2.7
在本地项目中使用 CommandParser	2.3

## 项目实例

BGZToolkit	3.1
项目概述	3.1.1
API 方法	3.1.2
根据 API 设计解析器	3.1.3
设计主函数	3.1.4
创建 Jar 包	3.1.5
应用 BGZToolkit	3.1.6
HttpDownloader	3.2

## 添加参数项

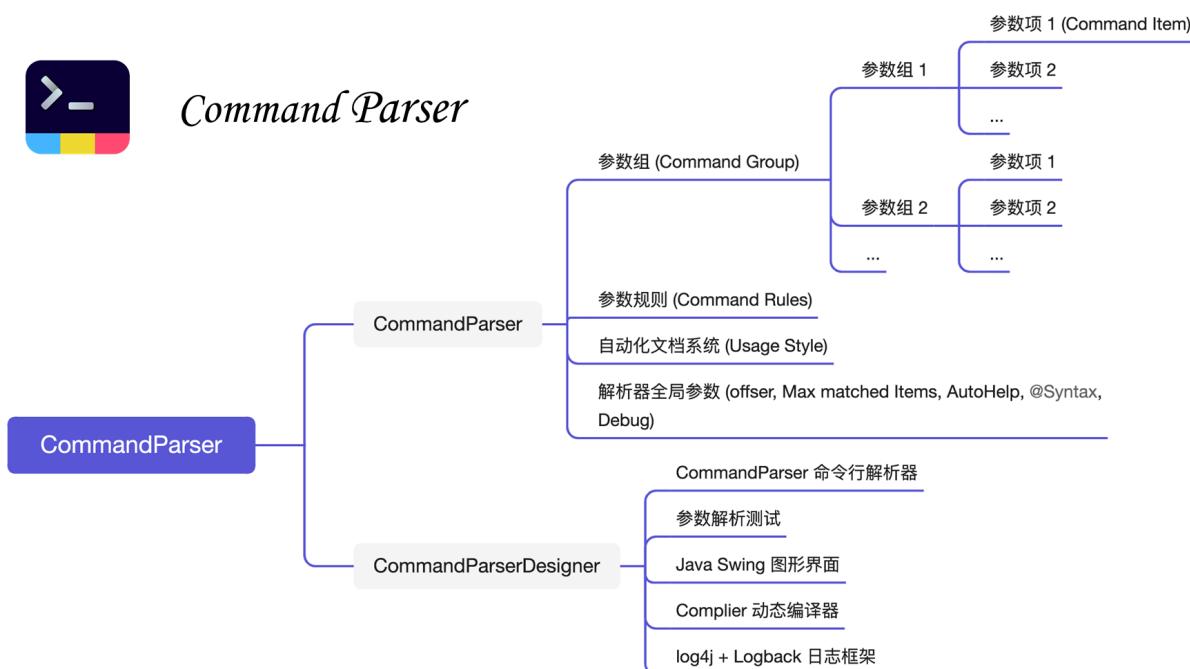
项目概述	3.2.1
API 方法	3.2.2
根据 API 设计解析器	3.2.3
设计主函数	3.2.4
创建 Jar 包	3.2.5
应用 HttpDownloader	3.2.6

## Command Parser 简介

CommandParser 是一个基于 Java 平台开发的轻量级框架，用于快速地开发、解析、管理命令行参数。它提供了一个基本的图形界面 (CommandParserDesigner)，用于可视化地管理、编辑命令项目。

CommandParser 具有如下优点：

- 跨平台：基于 Java8 开发，所有软件库均为标准库，能够在广泛的设备及 JDK 版本中运行；
- 图形化设计：commandParserDesigner-1.1.jar 提供了友好的图形化设计与管理功能，帮助开发者轻松预览、管理具有成百上千参数项的解析器；图形化设计的解析器可以直接导出 Java 脚本文件运行；
- 脚本设计：CommandParser 支持纯脚本开发，链式调用完成参数项注册；脚本文件也可以导入图形化设计程序进行可视化编辑、管理；
- 轻量级：commandParser-1.1.jar 仅 400+ KB，无外部依赖；核心程序包与图形界面程序分离，减小包大小；
- 单行解析：通过 `Parser.parse(...)` 完成字符串数组或文件的解析；
- 自动化文档及高度可定制化：自动化文档几乎满足所有的文档设计需求，开放接口设计允许用户自定义文档风格；
- 支持指令集之间的预定规则：在解析阶段完成指令互斥、依赖关系的校验，减少额外代码量；
- 开发模式：允许调试参数与用户参数在同一个脚本文件中进行开发，互不干扰。



[!COMMENT|label:联系开发者] 张柳彬, suranyi.sysu@gmail.com

## 致谢

- 感谢袁杨杨博士在本项目开发过程中的帮助，包括功能性测试、文档翻译及修订。
- 感谢李淼新教授对本项目图形界面交互提出的若干意见。
- 感谢彭文杰同学为本项目提供了第一版翻译稿。

## 命令行界面

命令行界面 (**Command-line interface, CLI**) 是一种基于文本的用户界面，用于运行程序、管理计算机文件以及与计算机交互。命令行界面与软件图形界面、Web 服务一样，都是用于实现程序操作的内部形式与人类可以接受的形式之间的转换。通常，命令行界面接受用户键盘输入的指令，并将指令解析为不同程序运行时的参数或设置，最终发起后端的计算任务。

在 Java 中将程序封装为可运行的 jar 包，能够方便不同平台之间传输、使用，不需要考虑 IDE 环境而进行繁琐的配置。通常 jar 包就包含了完成一套计算/分析/服务所需的全部代码，为了让 jar 包可以根据不同的输入参数执行不同的任务，需要对命令行进行高效地解析，而有效的命令行解析工具有助于提升开发效率、用户交互体验等。

```
java -jar bgzip.jar --level 6  
compress BGZToolkitParser.java  
--output BGZToolkitParser.java.gz
```

用户输入



```
int level = 6  
File compress = BGZToolkitParser.java  
File output = BGZToolkitParser.java.gz
```

参数解析

## 下载与安装

CommandParser 在 JDK 8 中开发完成，得益于 Java 跨平台及向下兼容的特性，它也可以在所有支持 Java8 语言的软件与硬件环境中运行。

- commandParser-1.1.jar: 用于导入其他项目中构建最小运行环境；
- commandParserDesigner-1.1.jar: 用于启动图形界面进行管理与开发。

资源类型	路径
软件包	<a href="http://pmglab.top/commandParser/commandParser-1.1.jar">http://pmglab.top/commandParser/commandParser-1.1.jar</a>
设计器图形界面	<a href="http://pmglab.top/commandParser/commandParserDesigner-1.1.jar">http://pmglab.top/commandParser/commandParserDesigner-1.1.jar</a>
源代码	<a href="https://github.com/Zhangliubin/CommandParser-1.1">https://github.com/Zhangliubin/CommandParser-1.1</a>
说明文档	<a href="http://pmglab.top/commandParser/">http://pmglab.top/commandParser/</a>
API 文档	<a href="http://pmglab.top/commandParser/api-docs/">http://pmglab.top/commandParser/api-docs/</a>
示例文件	<a href="http://pmglab.top/commandParser/bgzip/BGZToolkitParser.java">http://pmglab.top/commandParser/bgzip/BGZToolkitParser.java</a> <a href="http://pmglab.top/commandParser/downloader/HttpDownloaderParser.java">http://pmglab.top/commandParser/downloader/HttpDownloaderParser.java</a>

## 通过 wget 下载软件包

```
mkdir commandParser-1.1
cd commandParser-1.1
wget http://pmglab.top/commandParser/commandParser-1.1.jar
wget http://pmglab.top/commandParser/commandParserDesigner-1.1.jar
```

## 更新日志

[!UPDATE|label:2022/06/01]

- 发布 CommandParser 的第二个版本，版本号 1.1，Github 仓库地址：  
<https://github.com/Zhangliubin/CommandParser-1.1>
- 旧版本即日起停止更新和维护。

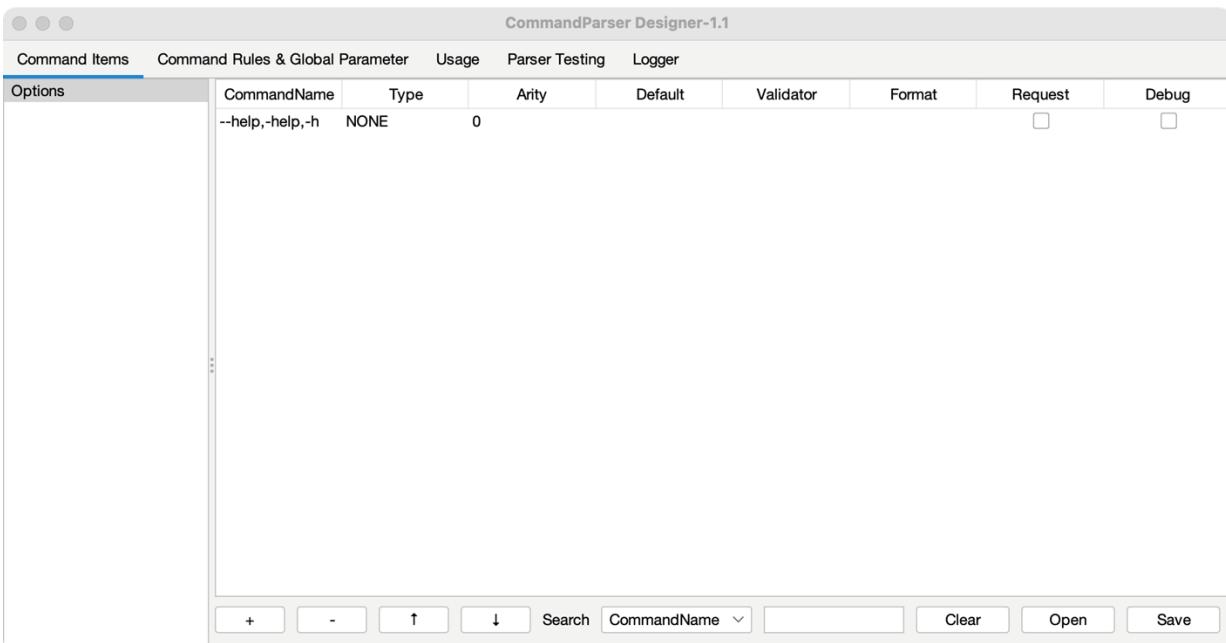
[!UPDATE|label:2021/11/23] 发布 CommandParser 的第一个版本，版本号 1.0，Github 仓库地址：

<https://github.com/Zhangliubin/CommandParser-1.0>

添加参数项

## 启动设计器图形界面

在终端中输入 `java -jar ./commandParserDesigner-1.1.jar` 指令或双击软件，启动图形界面。



[!TIP|label:设置快捷指令] Macos 或 Unix 系统可使用以下方式将 `java -jar ./commandParserDesigner-1.1.jar` 设置为快捷指令：

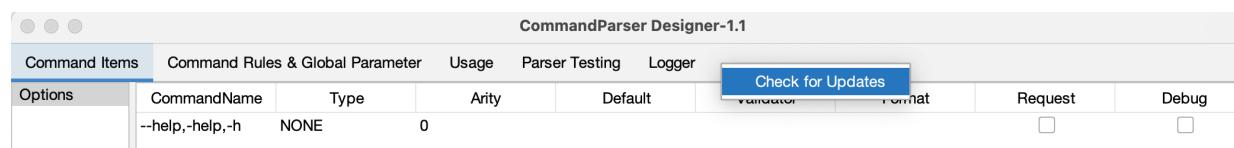
```
# 打开环境变量文件
vim ~/.zshrc

# 添加指令简写 (${path} 为 commandParserDesigner-1.1.jar 的绝对路径)
alias commandParser="java -jar ${path}"

# 按下 Esc, 输入 ":x" 并按回车, 保存并退出该文件
source ~/.zshrc
```

## 检查软件更新

在标签栏处点击鼠标右键，展开菜单。点击 "Check for Updates" 将检查 CommandParser 和 Command Parser Designer 是否有更新。



## 查看日志

在 Logger 标签页中，可以查看当前程序的工作日志，该日志系统基于 log4j + Logback 构建。

[!DANGER|label:Command Parser Designer 仅作为单独软件包使用] 将日志重定向到 JSwing 面板需要在日志系统加载前启动 GUI 界面。因此当用户导入 commandParserDesigner.jar 作为包使用时，无论是否主动调用图形界面，GUI 组件都会被加载，影响主营业务逻辑。

## 添加参数项

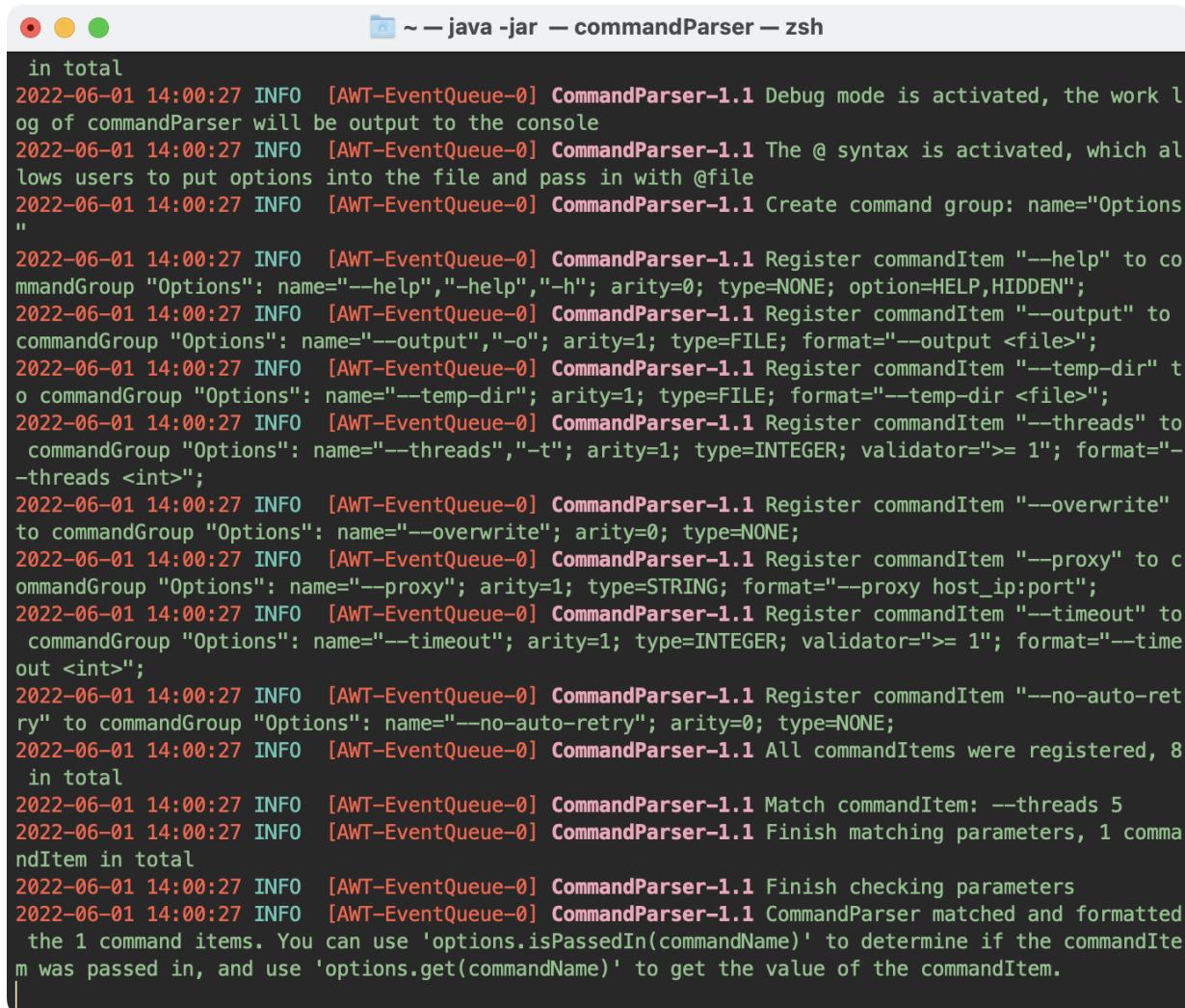


CommandParser Designer-1.1: BGZToolkitParser.java

Command Items    Command Rules & Global Parameter    Usage    Parser Testing    Logger

```
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--compress" to commandGroup "Mode": name= "--compress", -c; arity=1; type=FILE; validator="Exists,File"; format="--compress <file>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--decompress" to commandGroup "Mode": name= "--decompress", -d; arity=1; type=FILE; validator="Exists,File"; format="--decompress <file>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--concat" to commandGroup "Mode": name= "--concat"; arity=-1; type=FILE_ARRAY; validator="Exists,File"; format="--concat <file> <file> ...";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--md5" to commandGroup "Mode": name= "--md5"; arity=1; type=FILE; validator="Exists,File"; format="--md5 <file>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--md5-decompress" to commandGroup "Mode": name= "--md5-decompress", --md5-d; arity=1; type=FILE; validator="Exists,File"; format="--md5-decompress <file>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Create command group: name="Options"  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--help" to commandGroup "Options": name="--help", "-h"; arity=0; type=NONE; option=HELP,HIDDEN;  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--output" to commandGroup "Options": name="--output", "-o"; arity=1; type=FILE; format="--output <file>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--range" to commandGroup "Options": name="--range", "-r"; arity=1; type=LONG_RANGE; validator="> 0"; format="--range <long><long>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--level" to commandGroup "Options": name="--level", "-l"; arity=1; type=INTEGER; validator="0 ~ 9"; format="--level <int>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--threads" to commandGroup "Options": name="--threads", "-t"; arity=1; type=INTEGER; validator="> 1"; format="--threads <int>";  
2022-05-30 02:18:38 INFO CommandParser-1.1 Register commandItem "--no-log" to commandGroup "Options": name="--no-log"; arity=0; type=NONE;  
2022-05-30 02:18:38 INFO CommandParser-1.1 All commandItems were registered, 11 in total  
2022-05-30 02:18:38 INFO CommandParser-1.1 Match commandItem: --compress /Users/suranyi/Desktop/BGZToolkitParser.java  
2022-05-30 02:18:38 INFO CommandParser-1.1 Finish matching parameters, 1 commandItem in total  
2022-05-30 02:18:38 INFO CommandParser-1.1 Finish checking parameters  
2022-05-30 02:18:38 INFO CommandParser-1.1 CommandParser matched and formatted the 1 command items. You can use 'options.isPassedIn(commandName)' to determine if the commandItem was passed in, and use 'options.get(commandName)' to get the value of the commandItem.
```

通过终端启动时，终端处也会同步显示日志信息 (当终端不支持该字符集时，会显示乱码):



~ -- java -jar -- commandParser -- zsh

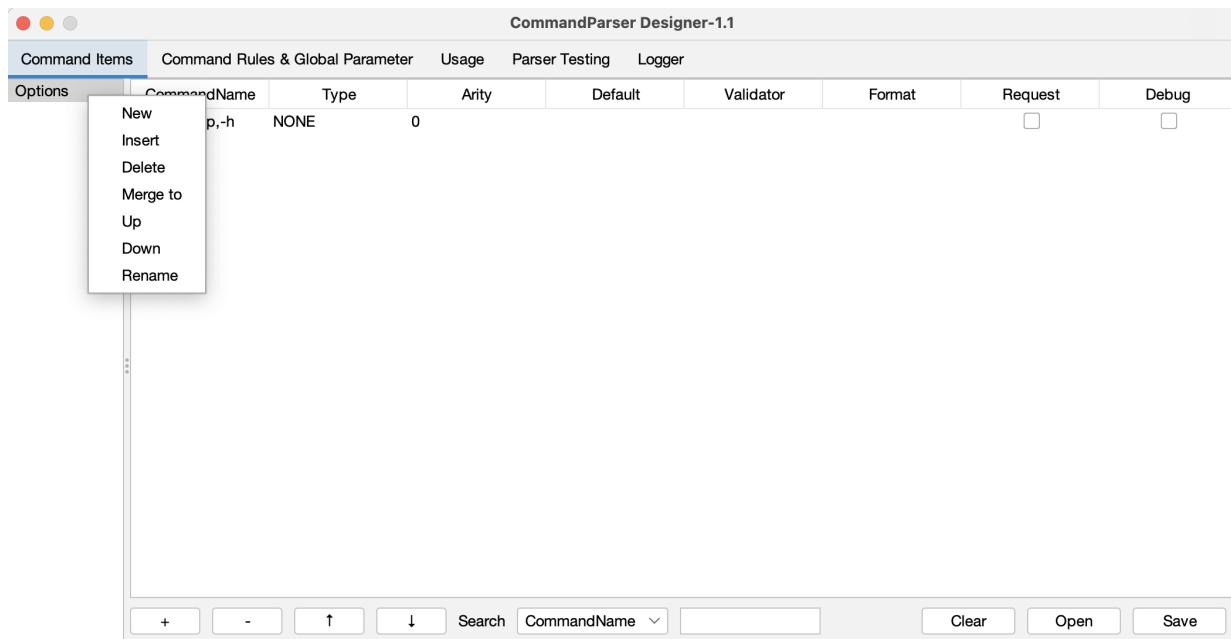
```
in total  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Debug mode is activated, the work log of commandParser will be output to the console  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 The @ syntax is activated, which allows users to put options into the file and pass in with @file  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Create command group: name="Options"  
  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--help" to commandGroup "Options": name="--help", "-h"; arity=0; type=NONE; option=HELP,HIDDEN;  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--output" to commandGroup "Options": name="--output", "-o"; arity=1; type=FILE; format="--output <file>";  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--temp-dir" to commandGroup "Options": name="--temp-dir"; arity=1; type=FILE; format="--temp-dir <file>";  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--threads" to commandGroup "Options": name="--threads", "-t"; arity=1; type=INTEGER; validator="> 1"; format="--threads <int>";  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--overwrite" to commandGroup "Options": name="--overwrite"; arity=0; type=NONE;  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--proxy" to commandGroup "Options": name="--proxy"; arity=1; type=STRING; format="--proxy host_ip:port";  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--timeout" to commandGroup "Options": name="--timeout"; arity=1; type=INTEGER; validator="> 1"; format="--timeout <int>";  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Register commandItem "--no-auto-retry" to commandGroup "Options": name="--no-auto-retry"; arity=0; type=NONE;  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 All commandItems were registered, 8 in total  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Match commandItem: --threads 5  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Finish matching parameters, 1 commandItem in total  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 Finish checking parameters  
2022-06-01 14:00:27 INFO [AWT-EventQueue-0] CommandParser-1.1 CommandParser matched and formatted the 1 command items. You can use 'options.isPassedIn(commandName)' to determine if the commandItem was passed in, and use 'options.get(commandName)' to get the value of the commandItem.
```

## 管理参数组 (Command Group)

通常按照参数的功能或属性可以将参数项粗分为几个大类，这个“大类”就是参数组。在 CommandParser 中，参数组是组织参数的基本单位。解析器初始化时，默认创建参数组 `Options`，以及该参数组中的第一个参数 `--help`, `-h`，用于呼出帮助文档。

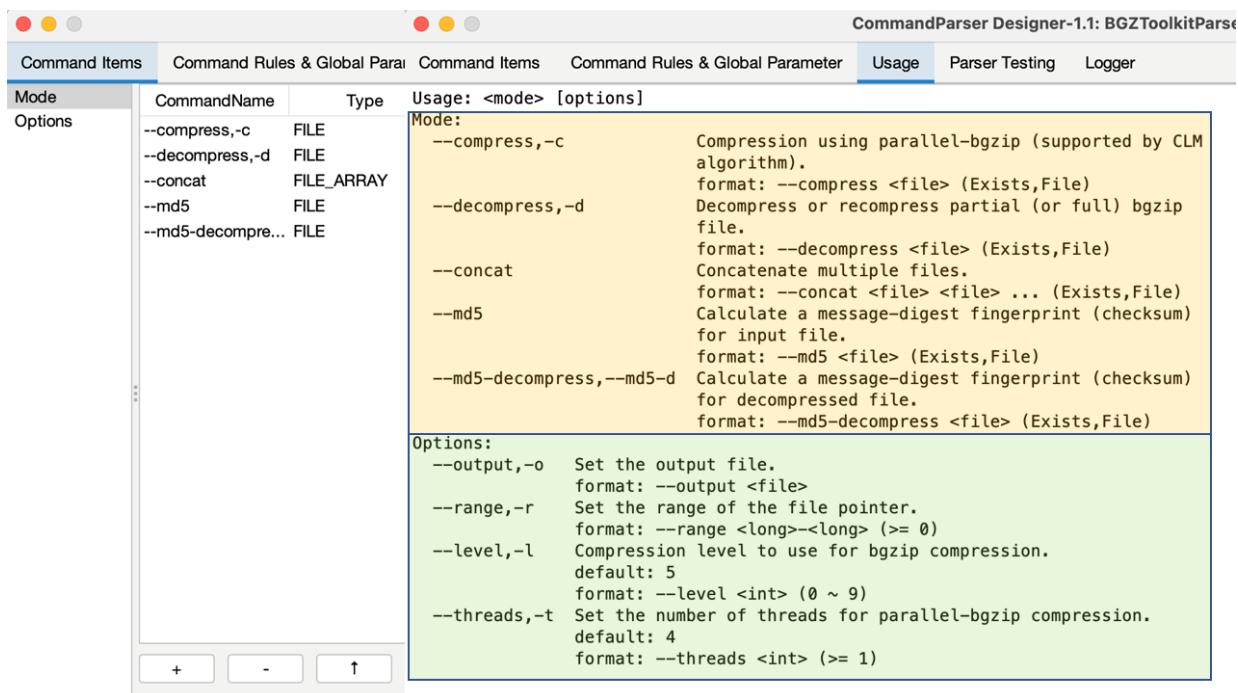
在 `Command Items` 标签页中，左侧为参数组面板。在参数组面板空白处或选中参数组点击鼠标右键，展开管理菜单。管理菜单包含以下 7 种操作：

- **New:** 创建新参数组
- **Insert:** 在当前位置插入新参数组
- **Delete:** 删除该参数组
- **Merge to:** 将该参数组合并至其他参数组
- **Up:** 上移该参数组
- **Down:** 下移该参数组
- **Rename:** 重命名参数组



参数组的顺序会影响自动化文档输出时的显示顺序：

## 添加参数项

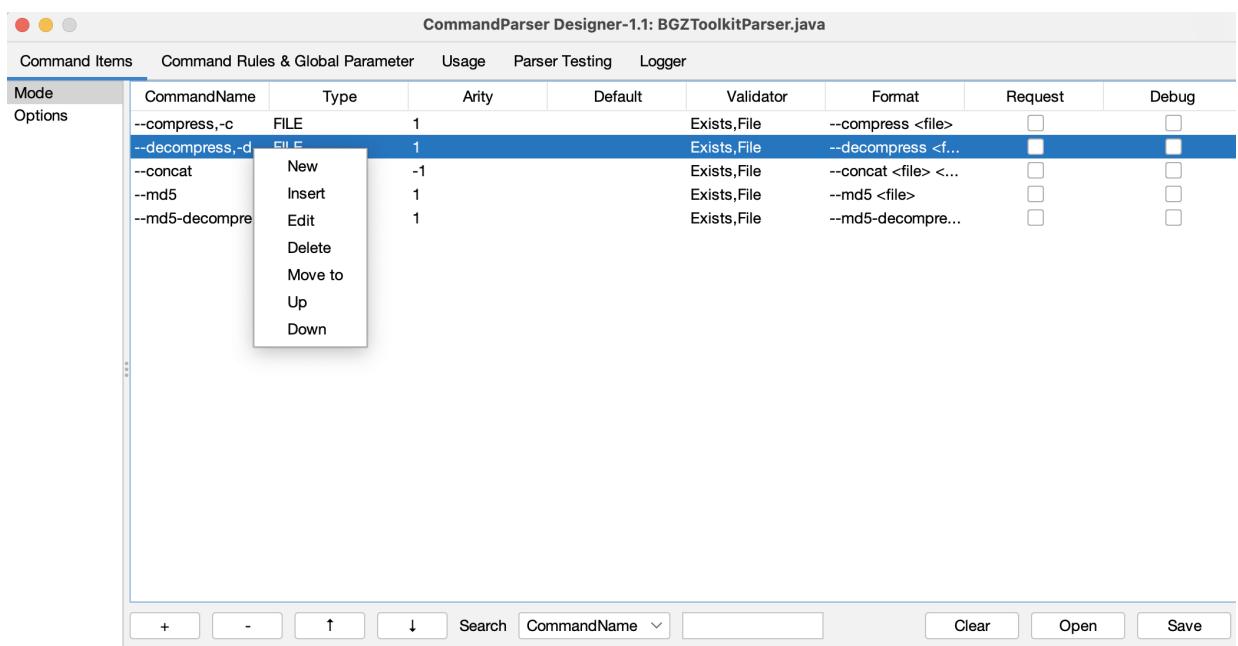


## 管理参数项 (Command Item)

在 Command Items 标签页中，右侧为参数项面板。在参数项面板空白处或选中参数项点击鼠标右键，展开管理菜单。管理菜单包含以下 7 种操作：

- **New:** 创建新参数项 (快捷键: Ctrl + N)
- **Insert:** 在当前位置插入新参数项
- **Edit:** 编辑该参数项 (快捷键: 双击参数项)
- **Delete:** 删除该参数项 (快捷键: Ctrl + Delete)
- **Merge to:** 将该参数项移动至其他参数组
- **Up:** 上移该参数项 (快捷键: Ctrl + U)
- **Down:** 下移该参数项 (快捷键: Ctrl + D)

下方菜单栏 “+” 对应 New 操作，“-” 对应 Delete 操作，“↑” 对应 Up 操作，“↓” 对应 Down 操作



## 搜索参数项

参数项面板下方搜索框处 (快捷键 Ctrl + F)，设置搜索的属性 (复选框) 及内容 (文本框)，按回车键查找符合的参数项并进行跳转、高亮显示。搜索内容忽略大小写。

CommandParser Designer-1.1: MainParser(1).java								
Command Items	Command Rules & Global Parameter		Usage	Parser Testing		Logger		
Options	CommandName	Type	Arity	Default	Validator	Format	Request	Debug
Inputs/Outputs	--info-cut	DOUBLE	1	0.0	0.0 ~ 1.0	--info-cut <doub...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Functions	--info-col	STRING	1	info		--info-col <string>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Utilities	--case-col,--cas...	STRING	1		AFF	--case-col <strin...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Quality Control	--control-col,--co...	STRING	1		UNAFF	--control-col <str...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--emic-plot-p	DOUBLE	1	0.0025	0.0 ~ 1.0	--emic-plot-p <d...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--calcu-selectivit...	NONE	0				<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--min-tissue	INTEGER	1	10		--min-tissue <int>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--ked-ref	FILE	1			--ked-ref <file>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--vcf-phased	NONE	0				<input type="checkbox"/>	<input type="checkbox"/>
	--expression-su...	STRING	1			--expression-su...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--no-gz	NONE	0				<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--hwe-control	DOUBLE	1	0.0	0.0 ~ 1.0	--hwe-control <d...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	<b>--hwe-case</b>	<b>DOUBLE</b>	<b>1</b>	<b>0.0</b>	<b>0.0 ~ 1.0</b>	<b>--hwe-case &lt;do...</b>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--seq-qual	DOUBLE	1	30.0		--seq-qual <dou...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--seq-mq	DOUBLE	1	20.0		--seq-mq <doub...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--seq-sb	DOUBLE	1	20.0		--seq-sb <doubl...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--seq-fs	DOUBLE	1	2.147483647E9		--seq-fs <double>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--max-allele	INTEGER	1	2	2 ~ 15	--max-allele <int>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--gtv-qual	DOUBLE	1	20.0	0.0 ~ 100.0	--gtv-qual <dou...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--gtv-sec-pl	INTEGER	1	20	0 ~ 100	--gtv-sec-pl <int>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	--atv-ap	FLOAT	1	0.6	0.0 ~ 1.0	--atv-ap <float>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

## 编辑参数项

解析器的工作单位是“参数项”，它声明了在海量词汇、符号组合中哪些字段是关键字？它之后有多少个参数是它的值？它捕获的值应该如何转为 Java 对象？它捕获的值进行什么样的检验？

新建或编辑参数项时，弹出参数项子面板，子面板有 12 个参数属性。`Command Name` 和 `Command Type` 定义了该参数项的关键字和数据类型，为必须设置的属性：

The screenshot shows the 'Edit Command' dialog box with the following steps highlighted:

- ① 设置参数名 (Set Command Name): A text input field containing '① 设置参数名'.
- ② 点击进行参数名规范检查 (Click to check name validity): A button labeled 'check'.
- ③-1 通过, 此时解锁其他控件 (Pass, unlock other controls): A green 'check' button.
- ④ 设置参数属性 (Set properties): A large text area for property settings.
- ⑤ 提交参数项 (Submit command): Buttons for 'Reset', 'Cancel', and 'Submit'.

其中，各个属性字段的含义如下：

参数属性	描述
Command Name 参数名	1. 参数具有多个参数名时，使用逗号分隔 (如: --output,-o)，第一个参数名作为主参数名; 2. 参数名格式: 0-9a-zA-Z+-_; 3. 参数名输入完成后点击 check 按钮进行检查 (检查是否符合格式、是否重名)，通过检查后的参数才能设置其他属性和提交。
Command Type 参数类型	1. 内置 None (无类型，仅验证是否被传入), Boolean, Byte, Short, Integer, Long, Float, Double, String, File 10 种基本类型; 2. 数据类型派生 Value, Array, Set, Range 等类型。
Validator 参数验证器	不同参数类型可用的验证器种类不同 (文件验证器、数值验证器等)，详见 <a href="#">参数验证器</a> 。
Default 默认值	该默认值需要经过 Command Type 进行格式转换及 Validator 参数值验证，输入格式与 Format 和 Arity 定义的格式一致。
Arity 参数长度	-1 表示不确定的长度(可以是 0 个参数)，直到下一个匹配的命令项的所有参数都被认为是这个命令项的值。
Format 参数的格式描述文档	默认根据参数类型进行设置 (通常为 主参数名 默认格式 )。
Description 参数的描述文档	参数的描述文档。
Help 是否识别为帮助指令	当标记为 Help 的参数项被传入时，解析器不会进行数值的格式转换和验证。
Request 是否为必备参数	没有传入 Help 标记的参数时，标记为 Request 的参数项必须传入。
Hidden 是否在文档中隐藏该指令	标记为 Hidden 的参数项不会在文档中显示。
Debug 该指令是否为调试指令	当解析器为 非 Debug 模式 时，标记为 Debug 的参数项将无法使用，相应的参数规则也会失效化，该参数项也不会在文档中显示。

## 参数类型输入格式

参数格式为： 主参数类型.派生参数类型 。主参数类型包括 IType.NONE (无类型，仅验证是否被传入), BOOLEAN, BYTE, SHORT, INTEGER, LONG, FLOAT, DOUBLE, STRING, FILE。派生参数类型包括以下 16 种类型。当类型被指定时，它被分配默认的格式文档 (i.e., `CommandName Format`)。带有 ... 标记的类型可以使用 arity 控制捕获的参数的个数。

Command Type 派生类型	默认格式 (Format)
VALUE	value
ARRAY	value value ...
ARRAY_COMMA	value,value,...
ARRAY_SEMICOLON	value;value;...
SET	value value ...
SET_COMMA	value,value,...
SET_SEMICOLON	value;value;...
MAP	key=value key=value ...
MAP_COMMA	key=value,key=value,...
MAP_SEMICOLON	key=value;key=value;...
RANGE	value-value
LABEL_RANGE	label:value-value label:value-value ...
LABEL_RANGE_COMMA	label:value-value,label:value-value,...
LABEL_RANGE_SEMICOLON	label:value-value;label:value-value;...
LABEL_ARRAY	label:value,value,... label:value,value,...
LABEL_ARRAY_SEMICOLON	label:value,value,...;label:value,value,...;...

## 参数验证器 (Validator)

不同类型的参数支持不同的验证器，如下所示：

Command Type	Validator 支持的类型
None, Boolean	不支持使用验证器。
Byte, Short, Integer, Long, Float, Double 数值类型	数值范围验证器： 1. 范围 (包含边界值): 最小值 ~ 最大值; 2. 指定最小值: $\geq$ 最小值。
String 字符串类型	限定值验证器 (只能取指定的元素): 多个限定值使用空格分隔, Ignore Case 忽略大小写, Index Access 允许使用索引访问 (0 代表第一个限定值...)。
File 文件类型	文件验证器 File Exists 文件路径必须存在; Single File 文件路径不能指向文件夹; Directory 文件路径必须指向文件夹; Inner Resource 优先识别当前运行环境资源 (允许访问 jar 包内部文件)。

## 管理参数规则 (Command Rule)

在 `Command Rules & Global Parameter` 标签页中添加参数间的指令规则。在参数规则面板空白处或选中参数规则点击鼠标右键，展开管理菜单。管理菜单包含以下 6 种操作：

- **New:** 创建新参数规则 (快捷键: Ctrl + N)
- **Insert:** 在当前位置插入新参数规则
- **Edit:** 编辑该参数规则 (快捷键: 双击参数规则)
- **Delete:** 删 除该参数规则 (快捷键: Ctrl + Delete)
- **Up:** 上移该参数规则 (快捷键: Ctrl + U)
- **Down:** 下移该参数规则 (快捷键: Ctrl + D)

下方菜单栏“+”对应 New 操作，“-”对应 Delete 操作，“↑”对应 Up 操作，“↓”对应 Down 操作

CommandNames	Rule_Type	Description
--vcf-ref, --ked-ref	AT_MOST_1	'--vcf-ref' + '--ked-ref' <= 1
--vcf-phased, --gene-feature-context	AT_MOST_1	'--vcf-phased' + '--gene-feature-context' <= 1
--rare-allele-freq, --allele-freq	AT_MOST_1	'--rare-allele-freq' + '--allele-freq' <= 1
-ignore-indel, --ignore-snv	AT_MOST_1	'-ignore-indel' + '--ignore-snv' <= 1
--candi-list, --candi-file	AT_MOST_1	'--candi-list' + '--candi-file' <= 1
--geneset-db, --geneset-file	AT_MOST_1	'--geneset-db' + '--geneset-file' <= 1
--db-gene, --regions-bed	AT_MOST_1	'--db-gene' + '--regions-bed' <= 1

## 搜索参数规则

参数规则面板下方搜索框处 (快捷键 Ctrl + F)，设置搜索的属性 (复选框) 及内容 (文本框)，按回车键查找符合的参数规则并进行跳转、高亮显示，搜索内容时忽略大小写。

## 添加参数项

The screenshot shows the CommandParser Designer application window titled "CommandParser Designer-1.1: MainParser(1).java". The main area is a table with columns: "CommandNames", "Rule\_Type", and "Description". The table lists various command-line options and their corresponding rule types and descriptions. At the bottom of the window, there are search and filter controls, and buttons for "Clear", "Open", and "Save".

CommandNames	Rule_Type	Description
--vcf-ref, --ked-ref	AT_MOST_1	'--vcf-ref' + '--ked-ref' <= 1
--vcf-phased, --gene-feature-context	AT_MOST_1	'--vcf-phased' + '--gene-feature-context' <= 1
--rare-allele-freq, --allele-freq	AT_MOST_1	'--rare-allele-freq' + '--allele-freq' <= 1
--ignore-indel, --ignore-snv	AT_MOST_1	'--ignore-indel' + '--ignore-snv' <= 1
--candi-list, --candi-file	AT_MOST_1	'--candi-list' + '--candi-file' <= 1
--geneset-db, --geneset-file	AT_MOST_1	'--geneset-db' + '--geneset-file' <= 1
--db-gene, --regions-bed	AT_MOST_1	'--db-gene' + '--regions-bed' <= 1

## 编辑参数规则

新建或编辑参数规则时，弹出参数规则子面板。无法为“Help”，“Request”类型参数设置规则。

② 勾选该参数规则适用的参数项

The image contains two side-by-side windows. The left window is titled "① 打开参数项选定面板" and shows a "Select Command Items" dialog with a "Rule Type" dropdown set to "AT\_MOST". It displays a message about specifying up to 1 item and provides a preview of the selected items. The right window is titled "③ 提交适用的参数项" and shows a table of command items with a "Selected" column where checkboxes are being checked. A red dashed box highlights the "Selected" column.

① 打开参数项选定面板

② 勾选该参数规则适用的参数项

③ 提交适用的参数项

④ 选择规则类型

⑤ 预览规则详情

⑥ 提交规则

## 参数规则类型

当参数规则选定了参数项  $\{p_1, p_2, \dots, p_n\}$  时，参数规则对应的含义如下：

添加参数项

参数规则类型	条件数	描述
AT_MOST	$k$	$p_1, p_2, \dots, p_n$ 至多传入 $k$ 个
AT_LEAST	$k$	$p_1, p_2, \dots, p_n$ 至少传入 $k$ 个
EQUAL	$k$	$p_1, p_2, \dots, p_n$ 需要传入 $k$ 个
MUTUAL_EXCLUSION 互斥	$k$	$p_1, p_2, \dots, p_k$ 与 $p_{k+1}, p_{k+2}, \dots, p_n$ 不能同时传入
SYMBIOSIS 依存	不支持	$p_1, p_2, \dots, p_n$ 同时传入或同时不传入
PRECONDITION 前置条件	不支持	当参数 $p_j$ 被传入时, $p_i (i < j)$ 都必须被传入

## 设置解析器全局属性

在 `Command Rules & Global Parameter` 标签页下部分设置解析器的全局属性。全局属性包含：

- **Program Name:** 程序名。
- **Usage Style:** 自动化文档的格式。双击复选框编辑格式，选择“...”创建新文档格式。
- **Offset:** 参数偏移量。
  - 跳过前面的 offset 个参数。
  - # offset = 3 时，传入的下列参数将跳过前 3 个参数，解析 “--level 5 -t 4 -o ~/test.gz”  
`bgzip compress <file> --level 5 -t 4 -o ~/test.gz`
- **Max Matched Items:** 最大匹配参数项个数，0 和 -1 表示不设置数量限制。
  - 传入的参数项达到最大个数时，后续的参数不再解析，而是作为最后一个匹配的参数项的值。
  - # maxMatchedItems = 1 时，下列参数只匹配 “bgzip”，剩下的参数 “compress <file> decompress <file>” 则作为 “bgzip compress <file> decompress <file>”
- **AutoHelp:** 在没有传入任何参数时，自动添加默认的 help 参数。
- **@Syntax:** @语法开关。
  - 在@语法下，识别 “@file” 类型参数，该参数被替换为 file 文件内容。
  - # file 文件内容为 compress <file> --level 5 -t 4，则下列语句解析值相同  
`bgzip @file -o ~/test.gz`  
`bgzip compress <file> --level 5 -t 4 -o ~/test.gz`
- **Debug:** 调试模式开关。标记为 “Debug” 的参数仅在调试模式下显示、可用；在调试模式下将显示解析器工作日志。



## 设置自动化文档的格式

在 `Command Rules & Global Parameter` 标签页双击 `Usage Style` 复选框或下拉选择 ...，打开文档格式编辑器。格式编辑器用于控制 `Usage` 标签页中的自动化文档格式（如左图），在 `Usage` 标签页中还可以进行复制、搜索（快捷键：Ctrl + F）。

## 添加参数项

Command Items   Command Rules & Global Parameter   Usage   Parser Testing   Logger

Usage: java -Xmx1g -jar kggsee.jar [options]  
Or: java -Xmx1g -jar kggsee.jar param.txt

Header Line 和 Sub Header Line

Options:

```
--help,-help,-h  
--emic-pfm-p  
    default: 2.5E-6  
    format: --emic-pfm-p <double>  
--lib-update  
--resource-update  
--web  
--sum-file,--pfile  
    path of the file storing GWAS summary statistics  
    format: --sum-file <file> (Exists,File)  
--vcf-phased  
--geneset-db  
    format: --geneset-db <string>  
--geneset-file  
    format: --geneset-file <string>  
--geneset-enrichment-test  
    default: 0.1  
    format: --geneset-enrichment-test <double>  
--qqplot  
    Draw quantile-quantile plot of p-values  
Inputs/Outputs:  
--chrom-col  
    column name of chromosome ID  
    default: CHR  
    format: --chrom-col <string>
```

Header Line Usage: `java -Xmx1g -jar kggsee.jar [options]`  
Sub Header Line Or: `java -Xmx1g -jar kggsee.jar param.txt`

Indent 1: 2   Indent 2: 6   Max Length: 80  
Request Mark: \*   Debug Mark: ^    Line Wrap after Command Name

OK   Cancel

Indent 1: 参数名前空格数  
Indent 2: 描述文档前空格数  
Max Length: 文档最大宽度 (超过宽度自动换行)  
Request Mark: 必备参数前添加的标记  
Debug Mark: 调试参数前添加的标记  
Line Wrap after Command Name: 参数名后换行

← Max Length →

## 参数解析测试

在 Parser Testing 标签页中可进行参数解析测试，Debug 模式下将会显示更详细的解析日志。在编辑框中输入参数，并点击 Parse 按钮，右侧将显示解析器对该次指令的参数捕获情况：

- **isPassedIn:** 是否传入该参数
- **MatchedParameters:** 该参数项捕获的参数值
- **Format:** 输入格式

双击参数项将跳转至 Command Items 标签页中的参数项位置。

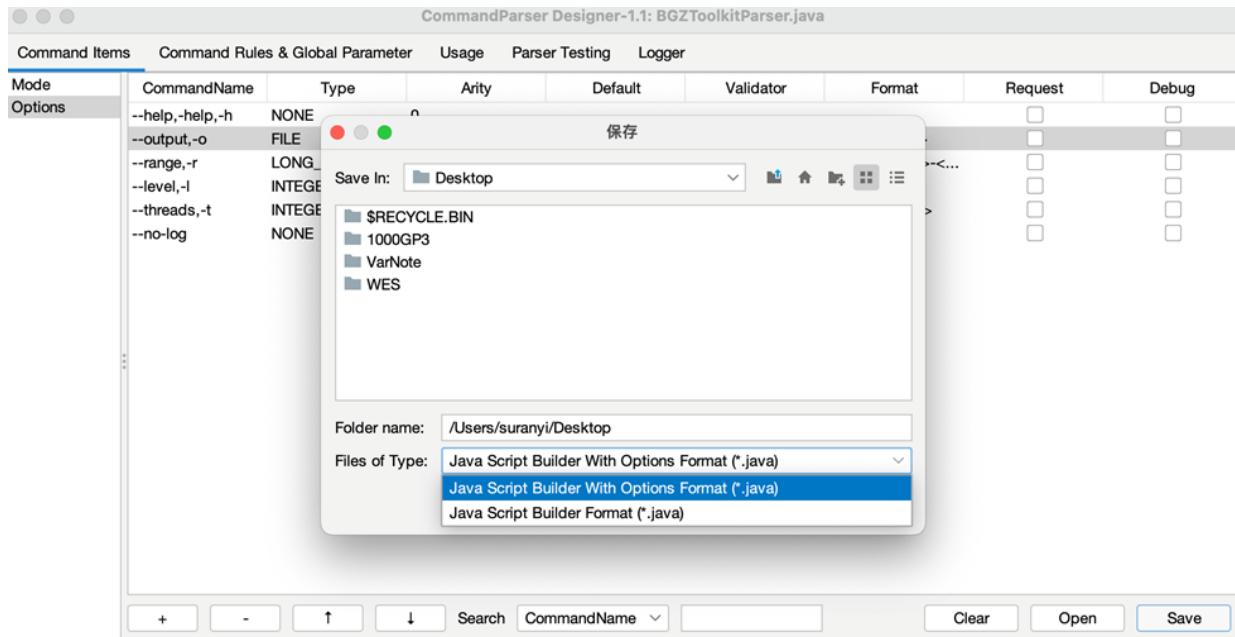
CommandName	isPassedIn	MatchedParameters	Format
--compress /Users/suranyi/Desktop/BGZT	<input checked="" type="checkbox"/>	/Users/suranyi/Desktop/BGZT...	--compress <file>
oolkitParser.java	<input type="checkbox"/>		--decompress <file>
--level 5	<input type="checkbox"/>		--concat <file> <file> ...
--output /Users/suranyi/Desktop/BGZToolk	<input type="checkbox"/>		--md5 <file>
itParser.java.gz	<input type="checkbox"/>		--md5-decompress <file>
--concat	<input type="checkbox"/>		
--md5	<input type="checkbox"/>		
--md5-decompress,--md5-d	<input type="checkbox"/>		
--help,-help,-h	<input type="checkbox"/>		
--output,-o	<input checked="" type="checkbox"/>	/Users/suranyi/Desktop/BGZT...	--output <file>
--range,-r	<input type="checkbox"/>		--range <long>--<long>
--level,-l	<input checked="" type="checkbox"/>	5	--level <int>
--threads,-t	<input type="checkbox"/>		--threads <int>
--no-log	<input type="checkbox"/>		

Parse    Open    Clear

## 导出解析器

在 `Command Items` 标签页或 `Command Rules & Global Parameter` 标签页中, 点击右下角 `Save` 按钮保存当前解析器 (全局快捷键: `Ctrl + S`)。提供两种保存格式:

- **Java Script Builder With Options Format:** 解析器及参数项构建文件
- **Java Script Builder Format:** 解析器文件



## Java Script Builder With Options Format 格式

`Java Script Builder With Options Format` 根据参数项的主参数名 (Command Name) 推断变量名 (驼峰命名法), 根据参数项的参数类型 (Command Type) 设置参数类型, 无需对变量进行类型转换。该格式支持的 API 方法如下:

API 方法	返回值类型	描述
<code>Parser options = Parser.parse(String[] args)</code>	<code>Parser</code>	解析指令
<code>Parser options = Parser.parse(File argsFile)</code>	<code>Parser</code>	解析文件中的指令
<code>Parser.usage()</code>	<code>String</code>	获取解析器文档
<code>Parser.getParser()</code>	<code>CommandParser</code>	获取解析器对象
<code>parser.getOptions()</code>	<code>CommandOptions</code>	获取解析参数值对象
<code>options.变量名.value</code>	<code>T (范形)</code>	获取参数值(未传入时, 该值为默认值)
<code>options.变量名.isPassedIn</code>	<code>boolean</code>	该参数项是否被传入
<code>options.变量名.matchedParameter</code>	<code>String</code>	捕获的参数值 (字符串原始格式)

在入口函数使用解析器桥接参数指令与业务逻辑:

添加参数项

```
public static void main(String[] args) {
    if (args.length == 0) {
        System.out.println(HttpDownloaderParser.getParser());
        return;
    }

    HttpDownloaderParser options = HttpDownloaderParser.parse(args);
    if (options.help.isPassedIn) {
        System.out.println(HttpDownloaderParser.getParser());
        return;
    }

    try {
        HttpDownloader2.instance(args[0])
            .setOutputFile(options.output.value)
            .setThreads(options.threads.value)
            .setPrintLog(true)
            .setTempDir(options.tempDir.value)
            .setProxy(options.proxy.value)
            .setTimeOut(options.timeout.value)
            .clean(options.overwrite.isPassedIn)
            .download();
    } catch (IOException e) {
        logger.error("{}: {}", e.getMessage());
    }
}
```

## Java Script Builder Format 格式

Java Script Builder Format 创建解析器单例, 通过参数名 (参数项的任一参数名) 访问参数解析信息 (值、是否被传入、捕获值), 在获取值时需要进行格式转换。该格式支持的 API 方法如下:

API 方法	返回值类型	描述
Parser.parse(String[] args)	CommandOptions	解析指令
Parser.parse(File argsFile)	CommandOptions	解析文件中的指令
Parser.usage()	String	获取解析器文档
Parser.getParser()	CommandParser	获取解析器对象
options.get(参数名)	Object	获取参数值(未传入时, 该值为默认值) 获得的参数类型是 Object类型, 需要手动转换格式
options.isPassedIn(参数名)	boolean	该参数项是否被传入
options.getMatchedParameter(参数名)	String	捕获的参数值 (字符串原始格式)

在入口函数使用解析器桥接参数指令与业务逻辑:

## 添加参数项

```
public static void main(String[] args) throws IOException {
    if (args.length == 0) {
        // 没有传入参数时，打印文档
        System.out.println(HttpDownloaderParser.usage());
        return;
    }

    CommandOptions options = HttpDownloaderParser.parse(args);
    if (options.isHelp()) {
        // 传入 help 指令时，打印文档
        System.out.println(HttpDownloaderParser.getParser());
        return;
    }

    // 业务逻辑
    try {
        HttpDownloader.instance(args[0])
            .setOutputFile((File) options.get("--output"))
            .setThreads((int) options.get("--threads"))
            .setPrintLog(!options.isPassedIn("--no-log"))
            .setTempDir((File) options.get("--temp-dir"))
            .setProxy((String) options.get("--proxy"))
            .setTimeOut((int) options.get("--time-out"))
            .clean(options.isPassedIn("--overwrite"))
            .download();
    } catch (IOException e) {
        logger.error("{}", e.getMessage());
    }
}
```

## 重编辑解析器文件

导出的解析器文件是 Java 源代码文件 (如下图), 用户可以直接修改 Java 源代码文件实现解析器的修改。此外,设计器图形界面也支持导入该源码文件 (拖拽文件到界面窗口、点击 Open 按钮或快捷键 Ctrl + O) 实现重编辑。

CommandParserDesigner 将解析器源代码文件动态编译为 class 文件, 并通过 .getParser() 获取解析器对象。最后, 设计器图形界面根据解析器对象的成员信息复现解析器。

```
parser.setProgramName("<mode>");

parser.offset(0);
parser.debug(false);
parser.usingAt(true);
parser.setMaxMatchedNum(-1);
parser.setUsageStyle(DefaultStyleUsage.UNIX_TYPE_1);

CommandGroup group001 = parser.addCommandGroup( groupName: "Mode");
group001.register(FILE.VALUE, ...commandNames: "--compress", "-c")
    .validateWith(FILE.validateWith( checkIsExists: true, checkIsFile: true))
    .setDescription("Compression using parallel-bgzip (supported by CLM algorithm).");
group001.register(FILE.VALUE, ...commandNames: "--decompress", "-d")
    .validateWith(FILE.validateWith( checkIsExists: true, checkIsFile: true))
    .setDescription("Decompress or recompress partial (or full) bgzip file.");
group001.register(FILE.ARRAY, ...commandNames: "--concat")
    .arity(-1)
    .validateWith(FILE.validateWith( checkIsExists: true, checkIsFile: true))
    .setDescription("Concatenate multiple files.");
group001.register(FILE.VALUE, ...commandNames: "--md5")
    .validateWith(FILE.validateWith( checkIsExists: true, checkIsFile: true))
    .setDescription("Calculate a message-digest fingerprint (checksum) for input file.");
group001.register(FILE.VALUE, ...commandNames: "--md5-decompress", "--md5-d")
    .validateWith(FILE.validateWith( checkIsExists: true, checkIsFile: true))
    .setDescription("Calculate a message-digest fingerprint (checksum) for decompressed file.");
```

## 初始化解析器

CommandParser 有 4 种构造器，参数 boolean init 代表是否创建初始的 help 指令 (默认为 true )；参数 String programName 代表解析器的程序名 (默认为 <main class> )：

- CommandParser parser = new CommandParser()
- CommandParser parser = new CommandParser(boolean init)
- CommandParser parser = new CommandParser(String programName)
- CommandParser parser = new CommandParser(boolean init, String programName)

```
CommandParser parser = new CommandParser(false);
```

## 设置解析器的全局属性

CommandParser 有 7 个全局属性，分别如下：

- parser.setProgramName(String programName)

设置程序名 (默认值: <main class> )。

- parser.offset(int length)

设置偏移量 (默认值: 0 )。

```
# offset = 3 时, 传入的下列参数将跳过前 3 个参数, 解析 "--level 5 -t 4 -o ~/test.gz"  
bgzip compress <file> --level 5 -t 4 -o ~/test.gz
```

- parser.debug(boolean enable)

是否为调试模式 (默认值: false )。

- parser.usingAt(boolean enable)

是否将 @ 识别为取地址符 (地址对应的文件内容作为参数传入) (默认值: true )。

- parser.setMaxMatchedNum(int length)

设置最大匹配参数项个数 (默认值: -1 )。传入的参数项达到最大个数时，后续的参数不再解析，而是作为最后一个匹配的参数项的值。

```
# maxMatchedItems = 1 时, 下列参数只匹配 "bgzip", 剩下的参数 "compress <file> decompress <file>" 则作为 "bgzip" 的参数  
bgzip compress <file> decompress <file>
```

- parser.setAutoHelp(boolean enable)

当没有指令被传入时，是否自动添加 help 指令 (默认值: false )。

- parser.setUsageStyle(IUsage usageStyle)

设置文档格式 (默认值: DefaultStyleUsage.UNIX\_TYPE\_1 )。

```
// DefaultStyleUsage 构造器:  
public DefaultStyleUsage(String before, String after, String subTitle, int indent1, int indent2, int maxLen
```

## 添加参数项

```
parser.setProgramName("<mode>");
parser.offset(0);
parser.debug(false);
parser.usingAt(true);
parser.setMaxMatchedNum(-1);
parser.setAutoHelp(true);
parser.setUsageStyle(DefaultStyleUsage.UNIX_TYPE_1);
```

## 添加参数组

CommandParser 通过以下两种方式添加参数组，并返回添加的参数组 (解析器中的同名参数组或新参数组):

- CommandGroup group = parser.addCommandGroup(String groupName)

添加组名为 `groupName` 的参数组。若该解析器中已存在同名参数组，则返回同名参数组，否则返回新参数组。

- CommandGroup group = parser.addCommandGroup(CommandGroup group)

若该解析器中已存在同名参数组，则将该参数组的所有参数注册到同名参数组中，否则将该参数组添加到解析器中。

```
CommandGroup group001 = parser.addCommandGroup("Mode");
CommandGroup group002 = parser.addCommandGroup("Options");
```

## 添加参数项

参数项可以通过参数组注入解析器:

- CommandItem item = group.register(IType type, String... commandNames)

主要方法。设置参数类型和参数名，注册参数项。

- CommandItem item = group.register(Class<?> tClass, String... commandNames)

设置参数类型和参数名，注册参数项。该方法的参数类型只能识别 VALUE 类型 (如: Integer.class) 和 ARRAY 类型 (如: int[].class)。

- CommandItem item = group.register(CommandItem commandItem)

若该解析器中已存在同名参数组，则抛出 `CommandParserException` 异常，否则将该参数项添加到解析器中。

直接通过解析器进行注册时，则向最后一次添加的参数组注册该参数项:

- CommandItem item = parser.register(IType type, String... commandNames)

设置参数类型和参数名，注册参数项。

- CommandItem item = parser.register(IType type, String... commandNames)

设置参数类型和参数名，注册参数项。该方法的参数类型只能识别 VALUE 类型 (如: Integer.class) 和 ARRAY 类型 (如: int[].class)。

- CommandItem item = parser.register(CommandItem commandItem)

若该解析器中已存在同名参数组，则抛出 `CommandParserException` 异常，否则将该参数项添加到解析器中。

## 添加参数项

[!NOTE|label:参数类型 主参数类型.派生参数类型 ]

主参数类型包括 IType.NONE (无类型, 仅验证是否被传入), BOOLEAN, BYTE, SHORT, INTEGER, LONG, FLOAT, DOUBLE, STRING, FILE。派生参数类型包括 [参数类型输入格式](#) 列出的 16 种类型。

# 设置参数项属性

参数项的属性设置都是返回参数项本身, 可以通过链式调用进行设置。属性的含义见[编辑参数项](#):

- CommandItem item = item.addOptions(String... options)  
添加参数 HIDDEN, HELP, REQUEST, DEBUG。
- CommandItem item = item.arity(int length)  
设置参数长度 只有不定长参数可以主动调用参数长度设置方法。
- CommandItem item = item.defaultValue(String... defaultValue)  
设置默认值 (使用字符串作为输入, 并按照格式转换器转为对应的值)。
- CommandItem item = item.validateWith(IValidator validator)  
设置参数验证器, 不同的参数类型适用不同的验证器方法。

Command Type	Validator 支持的类型
None, BOOLEAN	不支持使用验证器。
BYTE, SHORT, INTEGER, LONG, FLOAT, DOUBLE 数值类型	类型.validateWith(最小值, 最大值) 类型.validateWith(最小值)
STRING 字符串类型	STRING.validateWith(String... elements) STRING.validateWith(boolean ignoreCase, final boolean indexAccess, String... elements) 允许设置多个限定值, Ignore Case 忽略大小写, Index Access 允许使用索引访问 (0 代表第一个限定值...)。
FILE 文件类型	FILE.validateWith(boolean checkExists, boolean checkIsFile, boolean checkIsDirectory, boolean checkInnerResource)  checkExists 文件路径必须存在; checkIsFile 文件路径不能指向文件夹; checkIsDirectory 文件路径必须指向文件夹; checkInnerResource 优先识别当前运行环境资源 (允许访问 jar 包内部文件)。

- CommandItem item = item.setFormat(String format)  
设置参数格式描述 (参数格式默认值见[参数类型输入格式](#), 默认为 主参数名 参数格式 )
- CommandItem item = item.setDescription(java.lang.String description)  
设置描述文档

```

group001.register(FILE.VALUE, "--compress", "-c")
    .validateWith(FILE.validateWith(true, true))
    .setDescription("Compression using parallel-bgzip (supported by CLM algorithm.)");
group001.register(FILE.VALUE, "--decompress", "-d")
    .validateWith(FILE.validateWith(true, true))
    .setDescription("Decompress or recompress partial (or full) bgzip file.");
group001.register(FILE.ARRAY, "--concat")
    .arity(-1)
    .validateWith(FILE.validateWith(true, true))
    .setDescription("Concatenate multiple files.");
group001.register(FILE.VALUE, "--md5")
    .validateWith(FILE.validateWith(true, true))
    .setDescription("Calculate a message-digest fingerprint (checksum) for input file.");
group001.register(FILE.VALUE, "--md5-decompress", "--md5-d")
    .validateWith(FILE.validateWith(true, true))
    .setDescription("Calculate a message-digest fingerprint (checksum) for decompressed file.");

group002.register(IType.NONE, "--help", "-help", "-h")
    .addOptions(HELP, HIDDEN);
group002.register(FILE.VALUE, "--output", "-o")
    .setDescription("Set the output file.");
group002.register(LONG.RANGE, "--range", "-r")
    .validateWith(LONG.validateWith(0L))
    .setDescription("Set the range of the file pointer.");
group002.register(INTEGER.VALUE, "--level", "-l")
    .defaultTo("5")
    .validateWith(INTEGER.validateWith(0, 9))
    .setDescription("Compression level to use for bgzip compression.");
group002.register(INTEGER.VALUE, "--threads", "-t")
    .defaultTo("4")
    .validateWith(INTEGER.validateWith(1))
    .setDescription("Set the number of threads for parallel-bgzip compression.");

```

## 设置参数规则

CommandParser 通过以下三种方式添加参数规则，规则的含义见[参数规则类型](#):

- parser.addRule(String ruleType, String... commands)  
添加参数规则，可选类型为 SYMBIOSIS 或 PRECONDITION。
- parser.addRule(String ruleType, int conditionalValue, String... commands)  
添加参数规则，可选类型为 AT\_MOST、AT\_LEAST、EQUAL、MUTUAL\_EXCLUSION。
- parser.addRule(CommandRule rule)  
添加参数规则。

```

parser.addRule(EQUAL, 1, "--md5-decompress", "--md5", "--compress", "--decompress", "--concat");
parser.addRule(MUTUAL_EXCLUSION, 1, "--md5", "--output", "--level", "--range", "--threads");
parser.addRule(MUTUAL_EXCLUSION, 1, "--md5-decompress", "--output", "--range", "--level", "--threads");
parser.addRule(MUTUAL_EXCLUSION, 1, "--concat", "--range", "--level", "--threads");

```

## 格式化解析器

按照标准模版格式化解析器后，该解析器可以直接拖入图形设计界面进行编辑、管理。图形设计界面可读要求：

- 类文件完整，可独立编译；
- 保留 package \${路径}; 字段；
- 类访问修饰符为 public；
- 具有静态方法 public static CommandParser getParser()。

## 添加参数项

CommandParserDesigner 将解析器源代码文件动态编译为 class 文件，并通过 .getParser() 获取解析器对象。最后，设计器图形界面根据解析器对象的成员信息复现解析器。

## 添加参数项

```
package edu.sysu.pmglab.bgztools;

import edu.sysu.pmglab.commandParser.CommandGroup;
import edu.sysu.pmglab.commandParser.CommandOption;
import edu.sysu.pmglab.commandParser.CommandOptions;
import edu.sysu.pmglab.commandParser.CommandParser;
import edu.sysu.pmglab.commandParser.types.FILE;
import edu.sysu.pmglab.commandParser.types.INTEGER;
import edu.sysu.pmglab.commandParser.types.IType;
import edu.sysu.pmglab.commandParser.types.LONG;
import edu.sysu.pmglab.commandParser.usage.DefaultStyleUsage;
import edu.sysu.pmglab.container.File;

import java.io.IOException;

import static edu.sysu.pmglab.commandParser.CommandItem.HELP;
import static edu.sysu.pmglab.commandParser.CommandItem.HIDDEN;
import static edu.sysu.pmglab.commandParser.CommandRule.EQUAL;
import static edu.sysu.pmglab.commandParser.CommandRule.MUTUAL_EXCLUSION;

public class BGZToolkitParser {

    /**
     * build by: CommandParser-1.1
     * time: 2022-05-31 18:06:01
     */
    private static final CommandParser PARSER = new CommandParser(false);

    private final CommandOptions options;
    public final CommandOption<File> compress;
    public final CommandOption<File> decompress;
    public final CommandOption<File[]> concat;
    public final CommandOption<File> md5;
    public final CommandOption<File> md5Decompress;
    public final CommandOption<?> help;
    public final CommandOption<File> output;
    public final CommandOption<long[]> range;
    public final CommandOption<Integer> level;
    public final CommandOption<Integer> threads;

    BGZToolkitParser(String... args) {
        this.options = PARSER.parse(args);
        this.compress = new CommandOption<>("--compress", this.options);
        this.decompress = new CommandOption<>("--decompress", this.options);
        this.concat = new CommandOption<>("--concat", this.options);
        this.md5 = new CommandOption<>("--md5", this.options);
        this.md5Decompress = new CommandOption<>("--md5-decompress", this.options);
        this.help = new CommandOption<>("--help", this.options);
        this.output = new CommandOption<>("--output", this.options);
        this.range = new CommandOption<>("--range", this.options);
        this.level = new CommandOption<>("--level", this.options);
        this.threads = new CommandOption<>("--threads", this.options);
    }

    public static BGZToolkitParser parse(String... args) {
        return new BGZToolkitParser(args);
    }

    public static BGZToolkitParser parse(File argsFile) throws IOException {
        return new BGZToolkitParser(CommandParser.readFromFile(argsFile));
    }

    /**
     * Get CommandParser
     */
    public static CommandParser getParser() {
        return PARSER;
    }

    /**
     * Get the usage of CommandParser
     */
    public static String usage() {
        return PARSER.toString();
    }
}
```

## 添加参数项

```
}

/**
 * Get CommandOptions
 */
public CommandOptions getOptions() {
    return this.options;
}

static {
    PARSER.setProgramName("<mode>");
    PARSER.offset(0);
    PARSER.debug(false);
    PARSER.usingAt(true);
    PARSER.setMaxMatchedNum(-1);
    PARSER.setAutoHelp(true);
    PARSER.setUsageStyle(DefaultStyleUsage.UNIX_TYPE_1);

    CommandGroup group001 = PARSER.addCommandGroup("Mode");
    group001.register(FILE.VALUE, "--compress", "-c")
        .validateWith(FILE.validateWith(true, true))
        .setDescription("Compression using parallel-bgzip (supported by CLM algorithm).");
    group001.register(FILE.VALUE, "--decompress", "-d")
        .validateWith(FILE.validateWith(true, true))
        .setDescription("Decompress or recompress partial (or full) bgzip file.");
    group001.register(FILE.ARRAY, "--concat")
        .arity(-1)
        .validateWith(FILE.validateWith(true, true))
        .setDescription("Concatenate multiple files.");
    group001.register(FILE.VALUE, "--md5")
        .validateWith(FILE.validateWith(true, true))
        .setDescription("Calculate a message-digest fingerprint (checksum) for input file.");
    group001.register(FILE.VALUE, "--md5-decompress", "--md5-d")
        .validateWith(FILE.validateWith(true, true))
        .setDescription("Calculate a message-digest fingerprint (checksum) for decompressed file.");

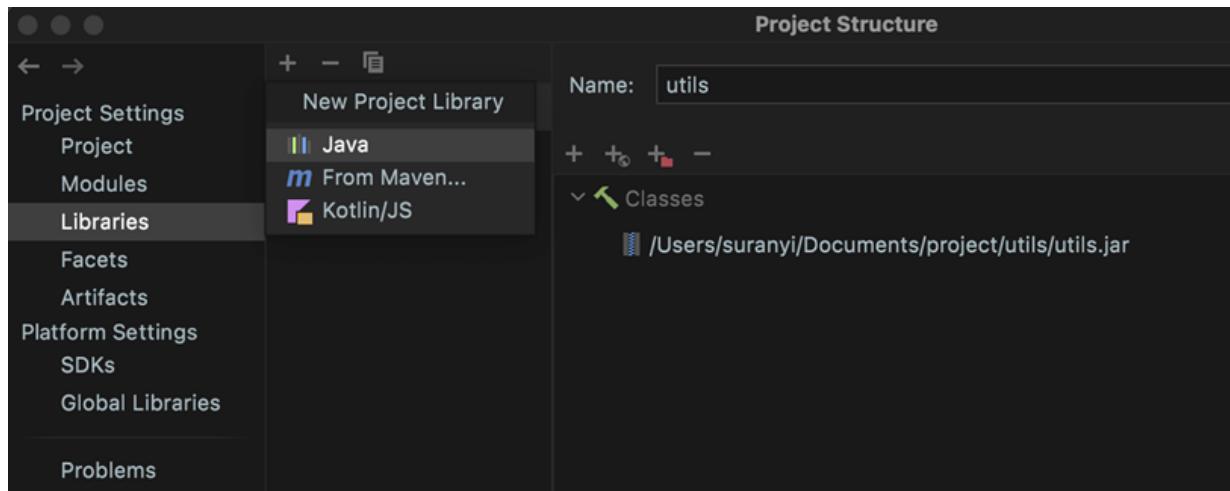
    CommandGroup group002 = PARSER.addCommandGroup("Options");
    group002.register(IType.NONE, "--help", "-help", "-h")
        .addOptions(HELP, HIDDEN);
    group002.register(FILE.VALUE, "--output", "-o")
        .setDescription("Set the output file.");
    group002.register(LONG.RANGE, "--range", "-r")
        .validateWith(LONG.validateWith(0L))
        .setDescription("Set the range of the file pointer.");
    group002.register(INTEGER.VALUE, "--level", "-l")
        .defaultTo("5")
        .validateWith(INTEGER.validateWith(0, 9))
        .setDescription("Compression level to use for bgzip compression.");
    group002.register(INTEGER.VALUE, "--threads", "-t")
        .defaultTo("4")
        .validateWith(INTEGER.validateWith(1))
        .setDescription("Set the number of threads for parallel-bgzip compression.");

    PARSER.addRule(EQUAL, 1, "--md5-decompress", "--md5", "--compress", "--decompress", "--concat");
    PARSER.addRule(MUTUAL_EXCLUSION, 1, "--md5", "--output", "--level", "--range", "--threads");
    PARSER.addRule(MUTUAL_EXCLUSION, 1, "--md5-decompress", "--output", "--range", "--level", "--threads");
    PARSER.addRule(MUTUAL_EXCLUSION, 1, "--concat", "--range", "--level", "--threads");
}
}
```

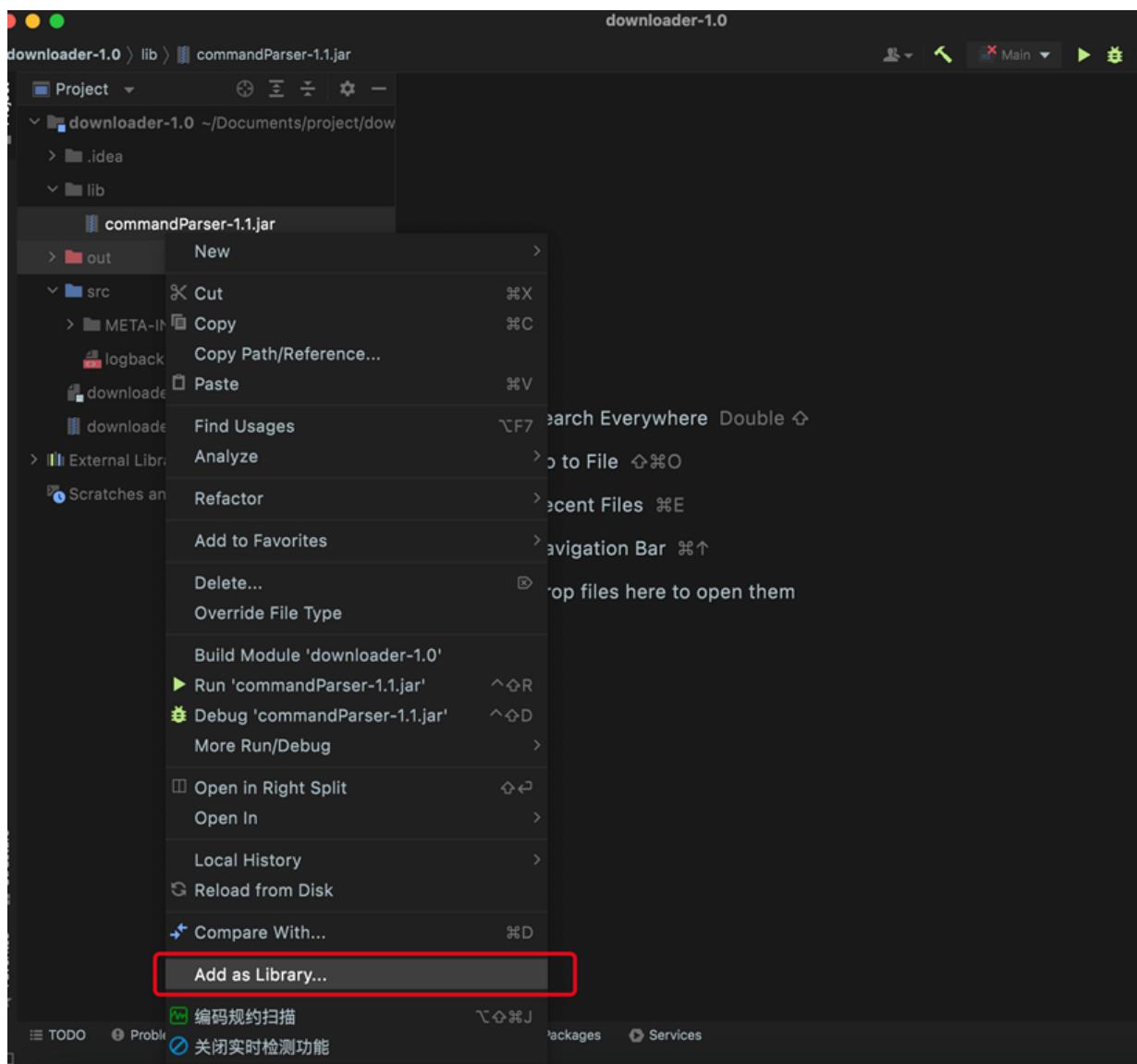
添加参数项

## 导入 CommandParser-1.1.jar

方式一: 在项目结构管理中选择“Libraries”，添加 commandParser-1.1.jar.



方式二: 在工程项目中创建 lib 文件夹, 添加 commandParser-1.1.jar, 并右键点击 “Add as Library...”.

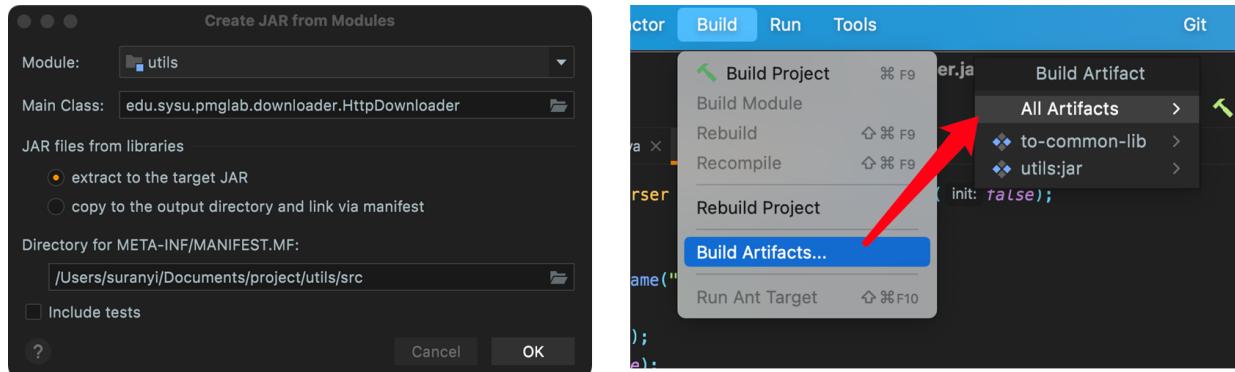


添加参数项

## 创建 Jar 包

创建解析器源代码文件、导入 commandParser-1.1.jar 包、编写入口函数后，就完成了完整的命令行程序开发。最后将 Java 工程项目打包为 jar 包（以 IDEA 为例）：

依次点击：Project Structure... > Artifacts > + > JAR > From modules with dependencies ... 显示左图窗口。在 Main Class 处选择入口函数位置，点击 OK。最后，点击 Build > Build Artifacts 构建 jar 包。



## BGZToolkit

BGZIP 是生物信息学领域的常用压缩工具，它将输入文件切分成近似 64KB 的小块，并单独应用 Deflater 压缩成一系列小的 BGZF 块，这使得可以针对压缩后的块构建索引信息，并用于快速地检索部分数据，而无需解压整个文件。

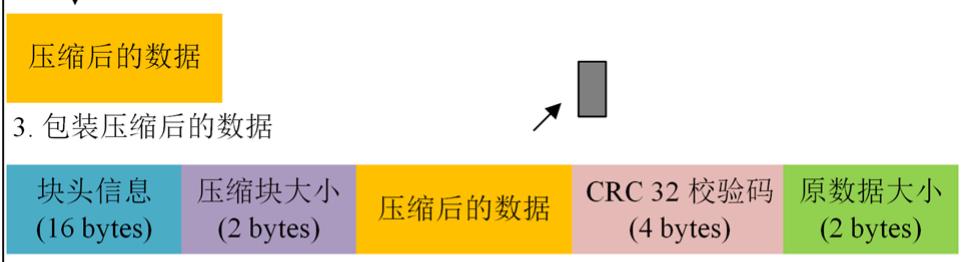
本项目使用纯 Java 脚本开发 BGZ 文件的操作方法，包括并行压缩、解压、拼接、文件校验等功能。

a.

1. 将原文件划分为近似 64KB 的块 (65498 bytes)



2. 每一个块使用 Deflater 算法 (zip 系列的核心算法) 压缩



4. 完成所有压缩后，在末尾追加一个空数据块 (长度为28 bytes)



资源类型	路径
软件包	<a href="http://pmglab.top/commandParser/bgzip/bgzip.jar">http://pmglab.top/commandParser/bgzip/bgzip.jar</a>
源代码	<a href="http://pmglab.top/commandParser/bgzip/BGZToolkit.java">http://pmglab.top/commandParser/bgzip/BGZToolkit.java</a>
解析器文件	<a href="http://pmglab.top/commandParser/bgzip/BGZToolkitParser.java">http://pmglab.top/commandParser/bgzip/BGZToolkitParser.java</a>

## API 方法

BGZIP 的主要 API 方法可以在 `edu.sysu.pmglab.bgztools.BGZToolkit` 中找到 (现已集成在 `commandParser` 包中)，分别为：

- **压缩文件:** `BGZToolkit.Compress.instance(File inputFile, File outputFile)`
  - 设置并行压缩线程数: `.setThreads(int nThreads)`
  - 设置压缩的文件范围: `.limit(long start, long end)`
  - 设置压缩级别: `.setCompressionLevel(int compressionLevel)`
- **提取文件:** `BGZToolkit.Extract.instance(File inputFile, File outputFile)`
  - 设置并行压缩线程数: `.setThreads(int nThreads)`
  - 设置读取的文件范围: `.limit(long start, long end)`
  - 设置输出文件格式: `.setOutputParam(BGZOutputParam outputParam)`
- **计算文件的 MD5 码:** `BGZToolkit.MD5.instance(File inputFile)`
  - 是否计算解压后文件的 MD5 码: `.setDecompression(boolean decompression)`
- **连接多个子文件:** `BGZToolkit.Concat.instance(File[] inputFiles, File outputFile)`

实例化任务完成后，通过 `.submit()` 提交相应的任务。

添加参数项

## 根据 API 设计解析器

BGZIP 工具集共有 5 个独立运行模式: 压缩、解压、连接、MD5 检查、MD5 检查(解压信息), 总共有包括线程数、文件指针范围、输出文件、压缩级别、是否打印日志在内的 5 个参数。创建步骤如下:

**Step1:** 创建参数组: Mode 组, Options 组;

**Step2:** 创建 Mode 组参数: --compress; --decompress; --concat; --md5; --md5-decompress;

**Step3:** 创建 Options 组参数: --output; --range; --level; --threads;

**Step4:** 设定参数规则:

- Mode 组参数 --compress; --decompress; --concat; --md5; --md5-decompress 必须传入一个;
- 选定了 --md5 或 --md5-decompress 模式时, 无法使用 --output; --range; --level; --threads 参数;
- 选定了 --concat 模式时, 无法使用 --range; --level; --threads 参数;

**Step5:** 设定程序名为: <`mode`> ;

**Step6:** 导出 Java Script Builder With Options Format 格式文件。

## 设计主函数

使用 CommandParser 桥接用户参数与业务逻辑, 将该文件命名为 `BGZToolkit.java`:

## 添加参数项

```
public static void main(String[] args) {
    try {
        BGZToolkitParser options = BGZToolkitParser.parse(args);
        if (options.getOptions().isHelp()) {
            System.out.println(BGZToolkitParser.usage());
            return;
        }

        if (options.compress.isPassedIn) {
            // 压缩文件
            long startTime = System.currentTimeMillis();
            File outputFile = options.output.isPassedIn ? options.output.value : options.compress.value.addExtension("gz");
            Compress task = Compress.instance(options.compress.value, outputFile)
                .setThreads(options.threads.value)
                .setCompressionLevel(options.level.value)
                .setPrintLog(true);

            if (options.range.isPassedIn) {
                task.limit(options.range.value[0], options.range.value[1]);
            }
            task.submit();
            logger.info("Compression completed. Total time: {} s; File size: {}", (System.currentTimeMillis() - startTime));
            return;
        }

        if (options.decompress.isPassedIn) {
            long startTime = System.currentTimeMillis();

            // 解压文件
            File outputFile;
            Extract task;
            if (options.output.isPassedIn) {
                outputFile = options.output.value;

                task = Extract.instance(options.decompress.value, outputFile)
                    .setThreads(options.threads.value);

                if (options.level.isPassedIn || options.output.value.withExtension(".gz")) {
                    task.setOutputParam(new BGZOutputParam(options.level.value));
                }
            } else {
                if (!options.level.isPassedIn) {
                    // 没有传入 level 时, 认为是解压文件
                    outputFile = options.decompress.value.changeExtension("", ".gz");
                    if (outputFile.equals(options.decompress.value)) {
                        logger.error("can't remove an extension from {} -- please rename", options.decompress.value);
                        return;
                    }
                } else {
                    // 传入了 level, 认为是提取文件子集
                    if (options.decompress.value.withExtension(".gz")) {
                        logger.error("missing required positional argument: --output");
                        return;
                    } else {
                        outputFile = options.decompress.value.addExtension(".gz");
                    }
                }
            }

            task = Extract.instance(options.decompress.value, outputFile)
                .setThreads(options.threads.value);
        }

        if (options.range.isPassedIn) {
            task.limit(options.range.value[0], options.range.value[1]);
        }

        task.submit();

        logger.info("Decompression completed. Total time: {} s; File size: {}", (System.currentTimeMillis() - startTime));
        return;
    }

    if (options.concat.isPassedIn) {
```

## 添加参数项

```
// 连接文件
long startTime = System.currentTimeMillis();
if (!options.output.isPassedIn) {
    logger.error("missing required positional argument: --output");
    return;
}

Concat.instance(options.concat.value, options.output.value).submit();

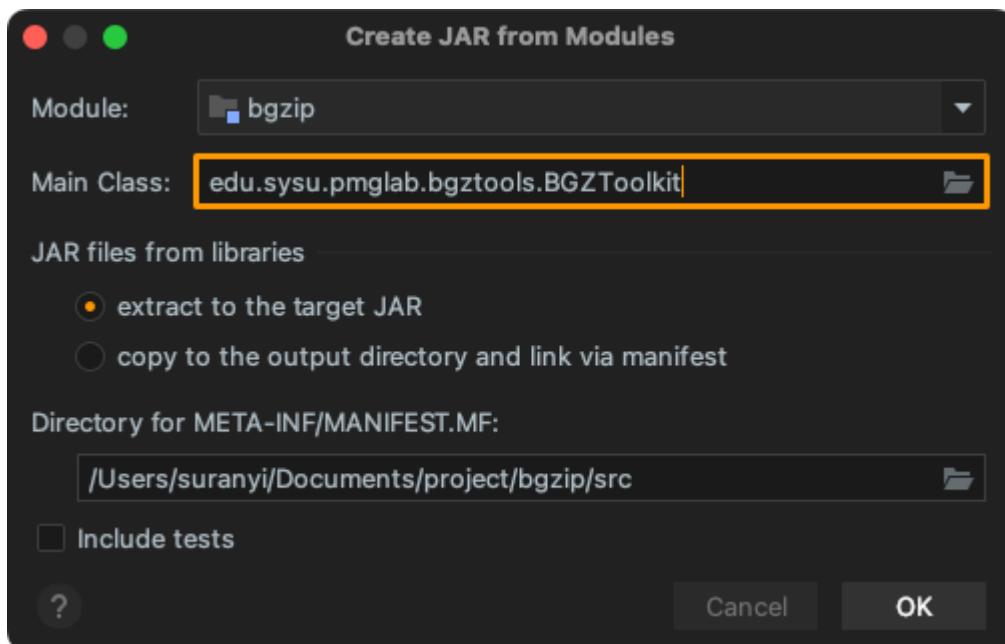
logger.info("Concat completed. Total time: {} s; File size: {}", (System.currentTimeMillis() - startTime));
return;
}

// md5 校验文件
if (options.md5.isPassedIn) {
    MD5 task = MD5.instance(options.md5.value);
    task.setDecompression(false);
    System.out.println("MD5 (" + options.md5.value + ") = " + task.submit());
    return;
}

if (options.md5Decompress.isPassedIn) {
    MD5 task = MD5.instance(options.md5Decompress.value);
    task.setDecompression(true);
    System.out.println("MD5 (" + options.md5Decompress.value + ") = " + task.submit());
    return;
}
} catch (Exception | Error e) {
    logger.error("{}: {}, e.getMessage(), e);
}
```

## 创建 jar 包

依次点击: Project Structure... > Artifacts > + > JAR > From modules with dependencies ..., 在 Main Class 处选择入口函数位置，并打包为 [bgzip.jar](#):



进入 `bgzip.jar` 所在路径，在控制台输入指令显示文档:

```
java -jar bgzip.jar
```

添加参数项

```
~/Documents/project/bgzip — suranyi@suranyi — ..project/bgzip — zsh
[(base) suranyi@suranyi ~] ~/Documents/project/bgzip > java -jar bgzip.jar
Usage: <mode> [options]
Mode:
--compress,-c           Compression using parallel-bgzip (supported by CLM
                        algorithm).
                        format: --compress <file> (Exists,File)
--decompress,-d          Decompress or recompress partial (or full) bgzip
                        file.
                        format: --decompress <file> (Exists,File)
--concat                Concatenate multiple files.
                        format: --concat <file> <file> ... (Exists,File)
--md5                   Calculate a message-digest fingerprint (checksum)
                        for input file.
                        format: --md5 <file> (Exists,File)
--md5-decompress,--md5-d Calculate a message-digest fingerprint (checksum)
                        for decompressed file.
                        format: --md5-decompress <file> (Exists,File)
Options:
--output,-o   Set the output file.
              format: --output <file>
--range,-r    Set the range of the file pointer.
              format: --range <long>-<long> (>= 0)
--level,-l    Compression level to use for bgzip compression.
              default: 5
              format: --level <int> (0 ~ 9)
--threads,-t   Set the number of threads for parallel-bgzip compression.
              default: 4
              format: --threads <int> (>= 1)
(base) suranyi@suranyi ~] ~/Documents/project/bgzip > |
```

## 应用 BGZToolkit

### 1. 多线程压缩单个文件

```
# 文件来源: http://pmglab.top/genotypes/#
java -jar bgzip.jar -c ~/Desktop/1000GP3/AMR/1kg.phase3.v5.shapeit2.amr.hg19.chr1.vcf -l 5 -t 8
```

```
~/Documents/project/bgzip — java -jar bgzip.jar -c -l 5 -t 8 — zsh
[(base) suranyi@suranyi ~] ~/Documents/project/bgzip > java -jar bgzip.jar -c ~/Desktop/1000GP3/AMR/1kg.phase3.v5.
._shapeit2.amr.hg19.chr1.vcf -l 5 -t 8
| Compressed: 5.6 GB / 9.3 GB (60.6 %); Speed: 1.1 GB/s; Time Left: 3.2 s|
```

### 2. 解压单个文件，并重新压缩为 bgzip 格式

```
# 提取前 1 GB 内容
java -jar bgzip.jar -d ~/Desktop/1000GP3/AMR/1kg.phase3.v5.shapeit2.amr.hg19.chr1.vcf.gz \
--range 0-1073741824 -o ~/Desktop/subFile.vcf.gz

# 解压上述文件
java -jar bgzip.jar -d ~/Desktop/subFile.vcf.gz
```

### 3. 检验文件 MD5 码

添加参数项

```
# 结果: 29ccebe9d2748328f6cc8b16c6563261
java -jar bgzip.jar --md5 ~/Desktop/subFile.vcf

# 结果: 29ccebe9d2748328f6cc8b16c6563261
java -jar bgzip.jar --md5-d ~/Desktop/subFile.vcf.gz
```

## 4. 连接多个文件

```
# 先拆分出 2 个文件
java -jar bgzip.jar -d ~/Desktop/1000GP3/AMR/1kg.phase3.v5.shapeit2.amr.hg19.chr1.vcf.gz \
--range 0-536870912 -o ~/Desktop/subFile1.vcf.gz
java -jar bgzip.jar -d ~/Desktop/1000GP3/AMR/1kg.phase3.v5.shapeit2.amr.hg19.chr1.vcf.gz \
--range 536870912-1073741824 -o ~/Desktop/subFile2.vcf.gz

# 拼接两个子文件
java -jar bgzip.jar --concat ~/Desktop/subFile1.vcf.gz ~/Desktop/subFile2.vcf.gz -o ~/Desktop/subFile.merge.vcf

# 校验 md5 码, 结果: 29ccebe9d2748328f6cc8b16c6563261
java -jar bgzip.jar --md5-d ~/Desktop/subFile.merge.vcf.gz
```

## HttpDownloader

生物信息学/医学信息学研究需要大量的公共数据资源支撑。然而，这些资源由于其数据量庞大（例如，dbNSFP 数据库达到 30 GB 以上，GnomAD 数据库达到 1TB 以上），在下载时伴随着诸多问题（例如重定向跳转、云端限速、断点续传等）。为了简化这些数据库的下载、更新流程，以及有效地将该流程作为 API 工具整合至其他的公共项目（这里主要针对 Java 开发），我们开发了HttpDownloader。此外，对于 FTP 的下载支持正在计划中。

HttpDownloader 是基于 Java 平台开发的简易 Http 下载器，旨在提升基础的资源下载功能。目前支持：断点续传、根据响应头的 Location 进行 URL 重定向、并行下载（需要服务器支持 chunked 或者发送分段请求后获得 206 响应码）、动态获取文件大小的资源下载、代理、迅雷链接下载。

资源类型	路径
软件包	<a href="http://pmglab.top/commandParser/downloader/downloader-1.0.jar">http://pmglab.top/commandParser/downloader/downloader-1.0.jar</a>
源代码	<a href="http://pmglab.top/commandParser/downloader/HttpDownloader.java">http://pmglab.top/commandParser/downloader/HttpDownloader.java</a>
解析器文件	<a href="http://pmglab.top/commandParser/downloader/HttpDownloaderParser.java">http://pmglab.top/commandParser/downloader/HttpDownloaderParser.java</a>

## API 方法

HttpDownloader 的主要 API 方法可以在 `edu.sysu.pmglab.downloader.HttpDownloader` 中找到（现已集成在 `CommandParser` 包中），分别为：

- 下载文件: `HttpDownloader.instance(String url)`
  - 设置并行下载线程数: `.setThreads(int nThreads)`
  - 设置输出文件名: `.setOutputFile(File outputFile)` 和 `.setOutputFile(String outputFileName)`
  - 设置临时文件夹: `.setTempDir(File tempDir)` 和 `.setTempDir(File tempDirName)`
  - 设置代理: `.setProxy(String host, String port)` 和 `.setProxy(String hostPort)`
  - 设置最长等待时间（单位: 秒): `.setTimeOut(int timeOut)`
  - 清除缓存数据: `.clean(boolean clean)`

实例化任务完成后，通过 `.download()` 执行下载任务。

## 根据 API 设计解析器

HttpDownloader 的第一个参数识别为 URL 地址，从第二个参数开始解析为参数项，设定 `offset=1`。解析器创建步骤如下：

**Step1:** 创建参数组：Options 组；

**Step2:** 创建 Options 组参数: `--output`; `--temp-dir`, `--threads`, `--overwrite`, `--proxy`, `--time-out`, `--no-auto-retry`;

**Step3:** 设定程序名为: `<URL>`，偏移量 `offset` 为: 1;

**Step4:** 设定 Usage Style 为 Unix\_Style\_3，并设置子标题信息如下:

About: Download file from an URL. The program supports breakpoints, parallel downloads, proxy IP settings and is suitable for downloading a wide range of http and thunder type urls.

**Step5:** 导出 Java Script Builder With Options Format 格式文件。

## 设计主函数

使用 CommandParser 桥接用户参数与业务逻辑，将该文件命名为 [HttpDownloader.java](#):

```
public static void main(String[] args) {
    try {
        if (args.length == 0) {
            System.out.println(HttpDownloaderParser.getParser());
            return;
        }

        HttpDownloaderParser parser = HttpDownloaderParser.parse(args);
        if (parser.help.isPassedIn) {
            System.out.println(HttpDownloaderParser.getParser());
            return;
        }

        do {
            try {
                HttpDownloader.instance(args[0])
                    .setOutputFile(parser.output.value)
                    .setThreads(parser.threads.value)
                    .setPrintLog(true)
                    .setTempDir(parser.tempDir.value)
                    .setProxy(parser.proxy.value)
                    .setTimeOut(parser.timeout.value)
                    .clean(parser.overwrite.isPassedIn)
                    .download();
            } catch (IOException e) {
                logger.error("{}" , e.getMessage());

                if (!parser.noAutoRetry.isPassedIn) {
                    logger.error("Download failed, resume download in 3 seconds.");

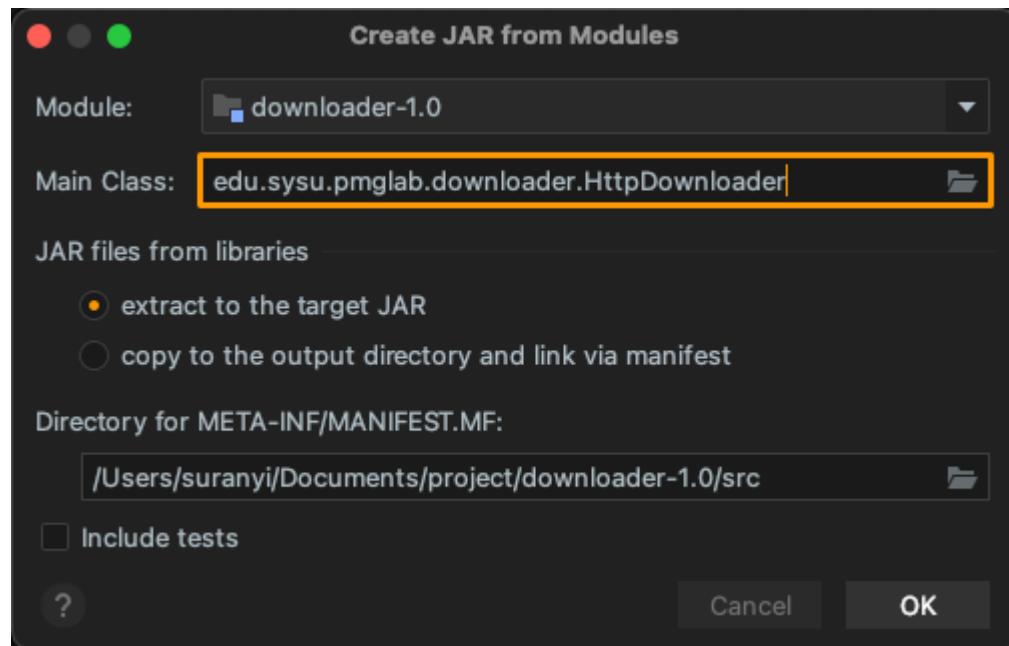
                    Thread.sleep(3000);
                    continue;
                }
            }

            break;
        } while (true);
    } catch (Exception | Error e) {
        logger.error("{}" , e.getMessage());
    }
}
```

## 创建 jar 包

依次点击: Project Structure... > Artifacts > + > JAR > From modules with dependencies ..., 在 Main Class 处选择入口函数位置，并打包为 [downloader-1.0.jar](#):

添加参数项



进入 `downloader-1.0.jar` 所在路径，在控制台输入指令显示文档：

```
java -jar downloader-1.0.jar
```

```
~/Documents/project/downloader-1.0 — suranyi@suranyi — ..ownloader-1.0 — zsh
[(base)] suranyi@suranyi ~] ~/Documents/project/downloader-1.0 > java -jar downloader-1.0.jar
Usage: <URL> [options]
About: Download file from an URL. The program supports breakpoints, parallel
       downloads, proxy IP settings and is suitable for downloading a wide
       range of http and thunder type urls.
Options:
--output,-o
  Set the output file name, the default is inferred from the URL address.
  format: --output <file>
--temp-dir
  Set the cache data path.
  format: --temp-dir <file>
--threads,-t
  Number of threads for parallel download.
  default: 5
  format: --threads <int> (>= 1)
--overwrite
  Overwrite existing files with --overwrite.
--proxy
  Set the proxy to access the target resource.
  format: --proxy host_ip:port
--timeout
  Disconnect if the server does not respond for more than --timeout
  second(s).
  default: 10
  format: --timeout <int> (>= 1)
--no-auto-retry
  Disable automatic reconnection on download failure.
```

## 应用 HttpDownloader

### 1. gnomAD 数据库下载

## 添加参数项

Genome Aggregation Database (简称gnomeAD)是由各国研究人员共同协作建立的一个基因组突变频率数据库，该数据库的目的是汇集和协调不同级别的大规模测序项目，包括全外显子与全基因组数据，为广泛的科学研究community汇总数据。目前该数据库包括125748个全外显子数据与15708个全基因组数据（v2版本），这些数据来源于各自不同的疾病研究项目与大型人口种群测序项目。该数据库包括之前常用的千人基因组数据、ESP数据库及绝大部分的ExAC数据库。

Amazon 的链接支持并行下载和断点续传，无需代理。

**Downloads**

See "What's the difference between gnomAD v2 and v3?" to decide which version is right for you.

gnomAD v2    gnomAD v2 liftover    **gnomAD v3**    ExAC

The gnomAD v3.1.2 data set contains 76,156 whole genomes (and no exomes), all mapped to the GRCh38 reference sequence.

**Summary**

- Variants
- Coverage
- HGDP + 1KG callset
- Mitochondrial DNA (mtDNA)
- Ancestry classification
- Local ancestry
- Short tandem repeats

**Variants**

**Note** Find out what changed in the latest release in the [gnomAD v3.1.2 blog post](#).

The variant dataset files below contain all subsets (non-cancer, non-neuro, non-v2, non-TOPMed, controls/biobanks, 1KG, and HGDP).

**Genomes**

- Sites Hail Table  
Show URL for Google / Amazon  
Copy URL for Google / Amazon
- chr1 sites VCF  
182.01 GiB, MD5: 65b21b95  
Download from Google / Amazon  
Download TBI from Google / Amazon
- chr2 sites VCF  
102.91 GiB, MD5: 5ad2d12001

在新标签页中打开链接  
在新窗口中打开链接  
在标签页组中打开链接 >  
下载链接文件  
下载链接文件为...  
将链接添加到书签...  
将链接添加到阅读列表  
0e3  
拷贝链接

```
java -jar ./downloader-1.0.jar https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz
```

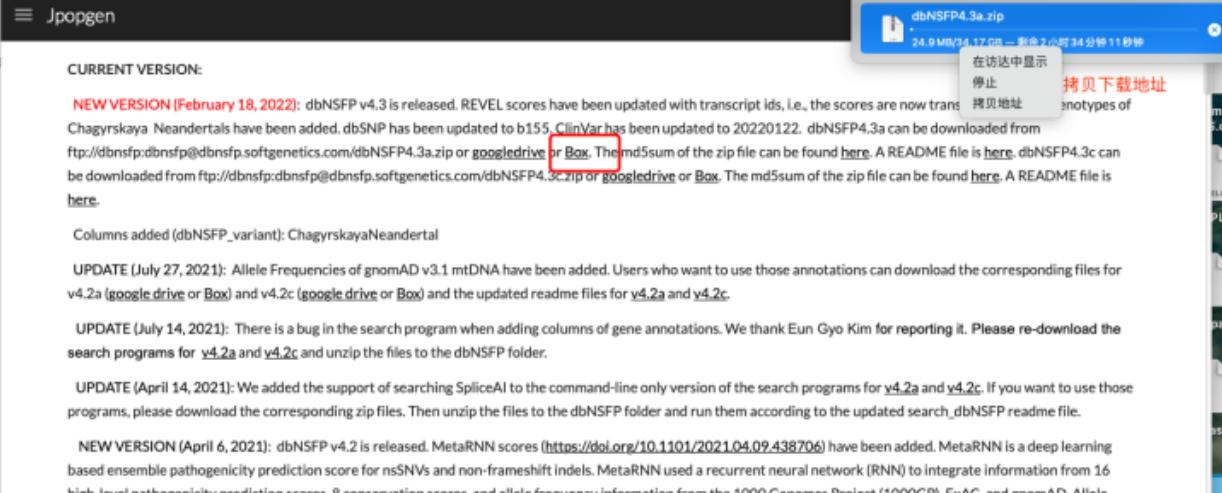
```
(base) suranyi@suranyi ~/Documents/project/downloader-1.0 ➤ java -jar ./downloader-1.0.jar https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz
2022-05-29 15:29:32 INFO [main] HttpDownloader Start download file /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz (from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz)
2022-05-29 15:29:32 INFO [ThreadPool-thread-3] HttpDownloader Download file from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz to /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz.~$temp/part2.~$temp (file pointer range: 78173652173 - 117260478258)
2022-05-29 15:29:32 INFO [ThreadPool-thread-1] HttpDownloader Download file from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz to /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz.~$temp/part0.~$temp (file pointer range: 0 - 39086826086)
2022-05-29 15:29:32 INFO [ThreadPool-thread-2] HttpDownloader Download file from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz to /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz.~$temp/part1.~$temp (file pointer range: 39086826087 - 78173652172)
[2022-05-29 15:29:32 INFO [ThreadPool-thread-5] HttpDownloader Download file from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz to /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz.~$temp/part4.~$temp (file pointer range: 156347304345 - 195434130433)
2022-05-29 15:29:32 INFO [ThreadPool-thread-4] HttpDownloader Download file from https://gnomad-public-us-east-1.s3.amazonaws.com/release/3.1.2/vcf/genomes/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz to /Users/suranyi/Desktop/gnomad.genomes.v3.1.2.sites.chr1.vcf.bgz.~$temp/part3.~$temp (file pointer range: 117260478259 - 156347304344)
| Downloaded: 239.1 MB / 182.0 GB (0.1 %); Speed: 2.2 MB/s; Time Left: 83308.1 s|
```

## 2. dbNSFP 数据库下载

dbNSFP是一个用于对人类基因组中所有潜在的非同义单核苷酸变异（nsSNV）进行功能预测和注释的数据库。其当前版本基于Gencode版本29/Ensembl版本94，包括总共84,013,490个nsSNV和ssSNV（拼接点SNV）。它汇编来自32个突变效应预测算法（包括SIFT、SIFT、SIFT4G、Polyphen2-HDIV、Polyphen2-HVAR、LRT、MutationTaster2、MutationAssessor、FATHMM、MetaSVM、MetaLR、MetaRNN、CADD、CADD\_hg19、

## 添加参数项

VEST4、PROVEAN、FATHMM-MKL、FATHMM-XF、fitCons x 4、LINSIGHT、DANN、GenoCanyon、Eigen、Eigen-PC、M-CAP、REVEL、MutPred、MVP、MPC、PrimateAI、GEogen2、BayesDel\_addAF、BayesDel\_noAF、ClinPred、LIST-S2、ALOFT) 的功能性预测结果、9个保守性评分 (如GERP++\_RS、phyloP100way、phastCons100way、bStatistic等) 和其他各类突变水平和基因水平的注释信息 (如等位基因频率、在ClinVar中的对应信息、基因表达、基因相互作用信息等)。



The screenshot shows a file download window from Jpopen. The file being downloaded is 'dbNSFP4.3a.zip' with a size of 24.8 MB/34.17 GB. The download progress is at 2 小时 34 分钟 11 秒。The window includes buttons for '显示' (Show), '停止' (Stop), '拷贝地址' (Copy Address), and '拷贝下载地址' (Copy Download Address). A red box highlights the '拷贝下载地址' button.

CURRENT VERSION:

**NEW VERSION (February 18, 2022):** dbNSFP v4.3 is released. REVEL scores have been updated with transcript ids, i.e., the scores are now trans. Chagyrskaya Neandertals have been added. dbSNP has been updated to b155. ClinVar has been updated to 20220122. dbNSFP4.3a can be downloaded from <ftp://dbnsfp:dbnsfp@dbnsfp.softgenetics.com/dbNSFP4.3a.zip> or [googledrive](#) or [Box](#). The md5sum of the zip file can be found [here](#). A README file is [here](#). dbNSFP4.3c can be downloaded from <ftp://dbnsfp:dbnsfp@dbnsfp.softgenetics.com/dbNSFP4.3c.zip> or [googledrive](#) or [Box](#). The md5sum of the zip file can be found [here](#). A README file is [here](#).

Columns added (dbNSFP\_variant): ChagyrskayaNeandertal

UPDATE (July 27, 2021): Allele Frequencies of gnomAD v3.1 mtDNA have been added. Users who want to use those annotations can download the corresponding files for v4.2a ([google drive](#) or [Box](#)) and v4.2c ([google drive](#) or [Box](#)) and the updated readme files for [v4.2a](#) and [v4.2c](#).

UPDATE (July 14, 2021): There is a bug in the search program when adding columns of gene annotations. We thank Eun Gyo Kim for reporting it. Please re-download the search programs for [v4.2a](#) and [v4.2c](#) and unzip the files to the dbNSFP folder.

UPDATE (April 14, 2021): We added the support of searching SpliceAI to the command-line only version of the search programs for [v4.2a](#) and [v4.2c](#). If you want to use those programs, please download the corresponding zip files. Then unzip the files to the dbNSFP folder and run them according to the updated search\_dbNSFP readme file.

**NEW VERSION (April 6, 2021):** dbNSFP v4.2 is released. MetaRNN scores (<https://doi.org/10.1101/2021.04.09.438706>) have been added. MetaRNN is a deep learning based ensemble pathogenicity prediction score for nsSNVs and non-frame-shift indels. MetaRNN used a recurrent neural network (RNN) to integrate information from 16 high-level pathogenicity prediction scores, 8 conservation scores, and allele frequency information from the 1000 Genomes Project (1000GP), ExAC, and gnomAD. Allele

```
java -jar ./downloader-1.0.jar https://dl2.boxcloud.com/d/1/b1\!XancTvwk7LlsYg9p6aRo-sxtzNi73B978Cn0D_ZLbgPY78i
```

## 添加参数项

```
y8APjby0uPm4zC7fk1poSJvfltVcAxwMY9l7LiqCsvW5m4./download to /Users/suranyi/Desktop/dbNSFP4.3a.zip.~$temp/part0.~$temp (file pointer range: 0 - 6834123741)
2022-05-29 15:44:36 INFO [ThreadPool-thread-2] HttpDownloader Download file from https://dl2.boxcloud.com/d/1/b1!XancTvwk7LlsYg9p6aRo-sxtzNi73B978Cn0D_ZLbgPY78iR4ZVnMt4k2mG307f60KK7Q5iv0---ZroDnXSpMdjJeJtvWgU2HFHl86wDUL4BA17uXH2qpsJ-6u7mTw0qubNoh2ytZkqtMvw-V9VWfkZ5wBGbJyJ6B0PRfjsq6xD5vu3U1YUNYJLp0UzDfQhsBCmsTp7hSUS8H-dhaUEL6ldh_mKMANCQLyVb0ZKX0AbkoT_ICbuZQ_fKTqJF496KkyXEROkaYve0xWjI0h000-FYT451_f2gxkxLQxWC0QkTJ1jCgeFi_5LYSrCvfnQMpAoVedBAJwIuFTSKBDG1Y04JFDlChMt_000fYEpbP3L5-7HdR_qo@iwZea3oVq8K_zar7rIicC7aw4SAT0xGyg1v_h7qI0xWegHry76vytyu8J7d3rrsJE_0YwyQMrBhBa2BkvCS-0g_YhdRz820v8Zt3bcG4tk_EWhApicBBz_KB8Rym_boobyXzgbVM55XKHk5Kt13xYYTGK3E855DrsQd_ypxfx_qXyeZ0oSQuV6capn_eT-iu5zTwin8jRNMB0t8uKfDRV9RE1utF3vc3UfI6vovpSup737H3wQl2cYwKQh2axeYvHHRPbnRp4Gz2uShcVsiai15QzJxnhuesLkfuyK7iT5phQX_eDzzVuhsZ5dc9BEivr-Y_Pt7pDPRGftkFwug1pILMu_LyGS-9f103xfgVJXRvn0TnxzBpd1yoSa0G_RaMiv3zNksdso-o4a2EM4e8V2xJTC52k80v6fDmp0WPxjKaF5616qZoYbyXfdBMSL4bZvXTCL-vqkg7n9GpHQig0XrkJmdnke2CX0mPsHiTaLxg6qF9dg441hBeUY6wGRAj1tcw4VGv35LIM012H0ZV_353P9tostSh_3E0m7Z0kitxJss0UsI3VmgtPoWMF-XnifvWg1M-0qgPr91hV0Azixuuo5arqfSUf3k8zywvPL2Wxt_EwfZiTMBSQI8tzBIST4NdAs6LDccrPRUARIaQ0AOx9kqmLay-ydERaqYyCmH17jKd3unPcI31BFRm20L5FLaeShJz1oSi0iHTmg0UQR0W9jicpnKERcxXUNK16hOurquu6c1NGbwZic0c0908WTxadzYVfQdVtY1PgfzPozVqrzml-JPyvl5HiozmvGp12052k6HArP4fTOIUHQjT_qsab-CzaTAVibzdyB7fqFtTxS_xiae6LfHk12C8VnZ18pSEwtkICzNWidagPBENxtpp9yAl8L_3X_p4R0hH7FFv5a5Fm55LfhRkmExJ1lP4FokzYeWnYgv40Z9R3CDBle-IKFjWnhWty8APjby0uPm4zC7fk1poSJvfltVcAxwMY9l7LiqCsvW5m4./download to /Users/suranyi/Desktop/dbNSFP4.3a.zip.~$temp/part1.~$temp (file pointer range: 6834123742 - 13668247482)
2022-05-29 15:44:36 INFO [ThreadPool-thread-3] HttpDownloader Download file from https://dl2.boxcloud.com/d/1/b1!XancTvwk7LlsYg9p6aRo-sxtzNi73B978Cn0D_ZLbgPY78iR4ZVnMt4k2mG307f60KK7Q5iv0---ZroDnXSpMdjJeJtvWgU2HFHl86wDUL4BA17uXH2qpsJ-6u7mTw0qubNoh2ytZkqtMvw-V9VWfkZ5wBGbJyJ6B0PRfjsq6xD5vu3U1YUNYJLp0UzDfQhsBCmsTp7hSUS8H-dhaUEL6ldh_mKMANCQLyVb0ZKX0AbkoT_ICbuZQ_fKTqJF496KkyXEROkaYve0xWjI0h000-FYT451_f2gxkxLQxWC0QkTJ1jCgeFi_5LYSrCvfnQMpAoVedBAJwIuFTSKBDG1Y04JFDlChMt_000fYEpbP3L5-7HdR_qo@iwZea3oVq8K_zar7rIicC7aw4SAT0xGyg1v_h7qI0xWegHry76vytyu8J7d3rrsJE_0YwyQMrBhBa2BkvCS-0g_YhdRz820v8Zt3bcG4tk_EWhApicBBz_KB8Rym_boobyXzgbVM55XKHk5Kt13xYYTGK3E855DrsQd_ypxfx_qXyeZ0oSQuV6capn_eT-iu5zTwin8jRNMB0t8uKfDRV9RE1utF3vc3UfI6vovpSup737H3wQl2cYwKQh2axeYvHHRPbnRp4Gz2uShcVsiai15QzJxnhuesLkfuyK7iT5phQX_eDzzVuhsZ5dc9BEivr-Y_Pt7pDPRGftkFwug1pILMu_LyGS-9f103xfgVJXRvn0TnxzBpd1yoSa0G_RaMiv3zNksdso-o4a2EM4e8V2xJTC52k80v6fDmp0WPxjKaF5616qZoYbyXfdBMSL4bZvXTCL-vqkg7n9GpHQig0XrkJmdnke2CX0mPsHiTaLxg6qF9dg441hBeUY6wGRAj1tcw4VGv35LIM012H0ZV_353P9tostSh_3E0m7Z0kitxJss0UsI3VmgtPoWMF-XnifvWg1M-0qgPr91hV0Azixuuo5arqfSUf3k8zywvPL2Wxt_EwfZiTMBSQI8tzBIST4NdAs6LDccrPRUARIaQ0AOx9kqmLay-ydERaqYyCmH17jKd3unPcI31BFRm20L5FLaeShJz1oSi0iHTmg0UQR0W9jicpnKERcxXUNK16hOurquu6c1NGbwZic0c0908WTxadzYVfQdVtY1PgfzPozVqrzml-JPyvl5HiozmvGp12052k6HArP4fTOIUHQjT_qsab-CzaTAVibzdyB7fqFtTxS_xiae6LfHk12C8VnZ18pSEwtkICzNWidagPBENxtpp9yAl8L_3X_p4R0hH7FFv5a5Fm55LfhRkmExJ1lP4FokzYeWnYgv40Z9R3CDBle-IKFjWnhWty8APjby0uPm4zC7fk1poSJvfltVcAxwMY9l7LiqCsvW5m4./download to /Users/suranyi/Desktop/dbNSFP4.3a.zip.~$temp/part2.~$temp (file pointer range: 13668247483 - 20502371223)
2022-05-29 15:44:36 INFO [ThreadPool-thread-5] HttpDownloader Download file from https://dl2.boxcloud.com/d/1/b1!XancTvwk7LlsYg9p6aRo-sxtzNi73B978Cn0D_ZLbgPY78iR4ZVnMt4k2mG307f60KK7Q5iv0---ZroDnXSpMdjJeJtvWgU2HFHl86wDUL4BA17uXH2qpsJ-6u7mTw0qubNoh2ytZkqtMvw-V9VWfkZ5wBGbJyJ6B0PRfjsq6xD5vu3U1YUNYJLp0UzDfQhsBCmsTp7hSUS8H-dhaUEL6ldh_mKMANCQLyVb0ZKX0AbkoT_ICbuZQ_fKTqJF496KkyXEROkaYve0xWjI0h000-FYT451_f2gxkxLQxWC0QkTJ1jCgeFi_5LYSrCvfnQMpAoVedBAJwIuFTSKBDG1Y04JFDlChMt_000fYEpbP3L5-7HdR_qo@iwZea3oVq8K_zar7rIicC7aw4SAT0xGyg1v_h7qI0xWegHry76vytyu8J7d3rrsJE_0YwyQMrBhBa2BkvCS-0g_YhdRz820v8Zt3bcG4tk_EWhApicBBz_KB8Rym_boobyXzgbVM55XKHk5Kt13xYYTGK3E855DrsQd_ypxfx_qXyeZ0oSQuV6capn_eT-iu5zTwin8jRNMB0t8uKfDRV9RE1utF3vc3UfI6vovpSup737H3wQl2cYwKQh2axeYvHHRPbnRp4Gz2uShcVsiai15QzJxnhuesLkfuyK7iT5phQX_eDzzVuhsZ5dc9BEivr-Y_Pt7pDPRGftkFwug1pILMu_LyGS-9f103xfgVJXRvn0TnxzBpd1yoSa0G_RaMiv3zNksdso-o4a2EM4e8V2xJTC52k80v6fDmp0WPxjKaF5616qZoYbyXfdBMSL4bZvXTCL-vqkg7n9GpHQig0XrkJmdnke2CX0mPsHiTaLxg6qF9dg441hBeUY6wGRAj1tcw4VGv35LIM012H0ZV_353P9tostSh_3E0m7Z0kitxJss0UsI3VmgtPoWMF-XnifvWg1M-0qgPr91hV0Azixuuo5arqfSUf3k8zywvPL2Wxt_EwfZiTMBSQI8tzBIST4NdAs6LDccrPRUARIaQ0AOx9kqmLay-ydERaqYyCmH17jKd3unPcI31BFRm20L5FLaeShJz1oSi0iHTmg0UQR0W9jicpnKERcxXUNK16hOurquu6c1NGbwZic0c0908WTxadzYVfQdVtY1PgfzPozVqrzml-JPyvl5HiozmvGp12052k6HArP4fTOIUHQjT_qsab-CzaTAVibzdyB7fqFtTxS_xiae6LfHk12C8VnZ18pSEwtkICzNWidagPBENxtpp9yAl8L_3X_p4R0hH7FFv5a5Fm55LfhRkmExJ1lP4FokzYeWnYgv40Z9R3CDBle-IKFjWnhWty8APjby0uPm4zC7fk1poSJvfltVcAxwMY9l7LiqCsvW5m4./download to /Users/suranyi/Desktop/dbNSFP4.3a.zip.~$temp/part3.~$temp (file pointer range: 27336494965 - 34170618707)
2022-05-29 15:44:36 INFO [ThreadPool-thread-4] HttpDownloader Download file from https://dl2.boxcloud.com/d/1/b1!XancTvwk7LlsYg9p6aRo-sxtzNi73B978Cn0D_ZLbgPY78iR4ZVnMt4k2mG307f60KK7Q5iv0---ZroDnXSpMdjJeJtvWgU2HFHl86wDUL4BA17uXH2qpsJ-6u7mTw0qubNoh2ytZkqtMvw-V9VWfkZ5wBGbJyJ6B0PRfjsq6xD5vu3U1YUNYJLp0UzDfQhsBCmsTp7hSUS8H-dhaUEL6ldh_mKMANCQLyVb0ZKX0AbkoT_ICbuZQ_fKTqJF496KkyXEROkaYve0xWjI0h000-FYT451_f2gxkxLQxWC0QkTJ1jCgeFi_5LYSrCvfnQMpAoVedBAJwIuFTSKBDG1Y04JFDlChMt_000fYEpbP3L5-7HdR_qo@iwZea3oVq8K_zar7rIicC7aw4SAT0xGyg1v_h7qI0xWegHry76vytyu8J7d3rrsJE_0YwyQMrBhBa2BkvCS-0g_YhdRz820v8Zt3bcG4tk_EWhApicBBz_KB8Rym_boobyXzgbVM55XKHk5Kt13xYYTGK3E855DrsQd_ypxfx_qXyeZ0oSQuV6capn_eT-iu5zTwin8jRNMB0t8uKfDRV9RE1utF3vc3UfI6vovpSup737H3wQl2cYwKQh2axeYvHHRPbnRp4Gz2uShcVsiai15QzJxnhuesLkfuyK7iT5phQX_eDzzVuhsZ5dc9BEivr-Y_Pt7pDPRGftkFwug1pILMu_LyGS-9f103xfgVJXRvn0TnxzBpd1yoSa0G_RaMiv3zNksdso-o4a2EM4e8V2xJTC52k80v6fDmp0WPxjKaF5616qZoYbyXfdBMSL4bZvXTCL-vqkg7n9GpHQig0XrkJmdnke2CX0mPsHiTaLxg6qF9dg441hBeUY6wGRAj1tcw4VGv35LIM012H0ZV_353P9tostSh_3E0m7Z0kitxJss0UsI3VmgtPoWMF-XnifvWg1M-0qgPr91hV0Azixuuo5arqfSUf3k8zywvPL2Wxt_EwfZiTMBSQI8tzBIST4NdAs6LDccrPRUARIaQ0AOx9kqmLay-ydERaqYyCmH17jKd3unPcI31BFRm20L5FLaeShJz1oSi0iHTmg0UQR0W9jicpnKERcxXUNK16hOurquu6c1NGbwZic0c0908WTxadzYVfQdVtY1PgfzPozVqrzml-JPyvl5HiozmvGp12052k6HArP4fTOIUHQjT_qsab-CzaTAVibzdyB7fqFtTxS_xiae6LfHk12C8VnZ18pSEwtkICzNWidagPBENxtpp9yAl8L_3X_p4R0hH7FFv5a5Fm55LfhRkmExJ1lP4FokzYeWnYgv40Z9R3CDBle-IKFjWnhWty8APjby0uPm4zC7fk1poSJvfltVcAxwMY9l7LiqCsvW5m4./download to /Users/suranyi/Desktop/dbNSFP4.3a.zip.~$temp/part4.~$temp (file pointer range: 27336494964 - 20502371224)
| Downloaded: 15.3 MB / 31.8 GB (0.0 %); Speed: 3.8 MB/s; Time Left: 8632.1 s|
```