# User Manual for

# HDF5_Caller: A Cross-platform Plugin for Fast Access to HDF5-based Reference Genome Sequence Indexing and Querying

Last updated on January 01, 2017

## Preparation

1 You can run the self-installing executable file to unpack and install the JDK. As part of the JDK, this installation includes an option to include the Java Runtime Environment. (http://www.oracle.com/technetwork/java/javase/downloads/index.html)

2 You can download the Eclipse IDE for Java Developers. The download will be delivered as a compressed (i.e. a ".zip", or ".tar.gz") file. Decompress this file into the directory of your choice (e.g. "c:\eclipse" on Windows) and ensure you have full Read and Execute permissions.(http://www.eclipse.org/downloads/); You also can use NetBeans IDE for Java Develpers. (https://netbeans.org/).

3 You can download "**HRefAligner_plugin.jar**" from our website.

4 You can download HRefAligner "**testing data**" from our website.

5 You can download HRefAligner **"demo code"** from our website.

6 You can download HRefAligner "**multi-threaded demo**" from our website.

7 You can operate HRefAligner according to this **user's manual**.

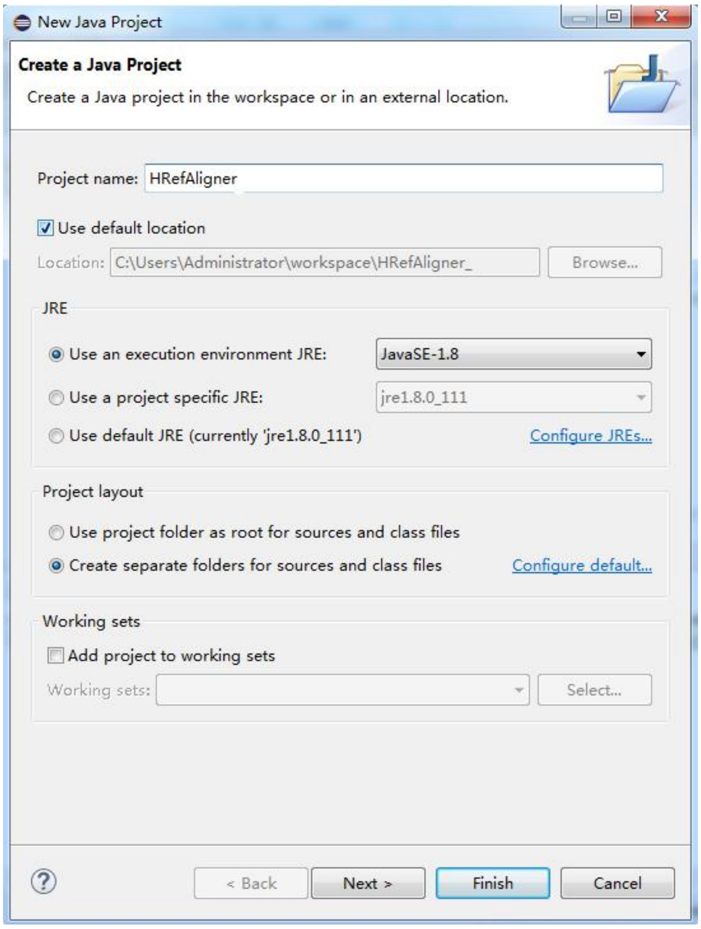
## I Creating a HRefAligner Project

In this section, you will create a new Java project. You will be using HRefAligner as your example project.

On this page(Fig.1),

- type "**HRefAligner**" in the Project name field.
- select "**JavaSE-1.8**" in the Use an execution environment JRE field.

Then click Finish.



New Java Project

**Create a Java Project**

Create a Java project in the workspace or in an external location.

Project name: HRefAligner

☑ Use default location

Location: C:\Users\Administrator\workspace\HRefAligner_ Browse...

JRE

● Use an execution environment JRE: JavaSE-1.8

○ Use a project specific JRE: jre1.8.0_111

○ Use default JRE (currently 'jre1.8.0_111') Configure JREs...

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files Configure default...

Working sets

☐ Add project to working sets

Working sets: Select...

Fig.1 Creating a HRefAligner Java Project

## II Loading HRefAligner_plugin.jar into project

1 In this section, you will create a “lib” folder inside the HRefAligner Project(Fig.2).

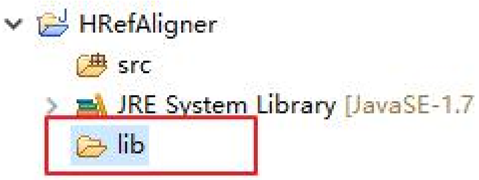


Fig.2 Creating a lib folder

2 You will load the HRefAligner_plugin.jar inside lib folder(Fig.3).

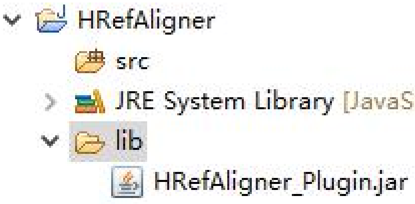


Fig.3 Loading HRefAligner_Plugin.jar

3 You will right click on HRefAligner_plugin.jar, and select Build path-> Add to Build Path (Fig.4).

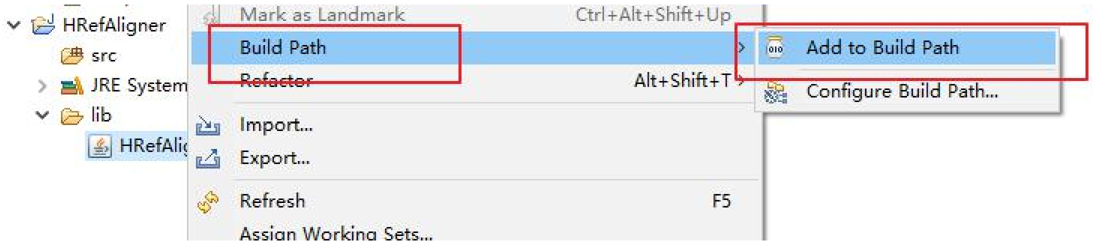


Fig.4 Setting JavaScript build path

Then the project runs fine when you see such package explorer (Fig.5).

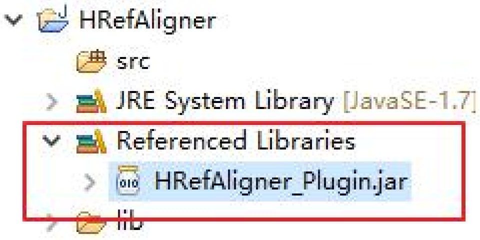


Fig.5 Package Explorer view

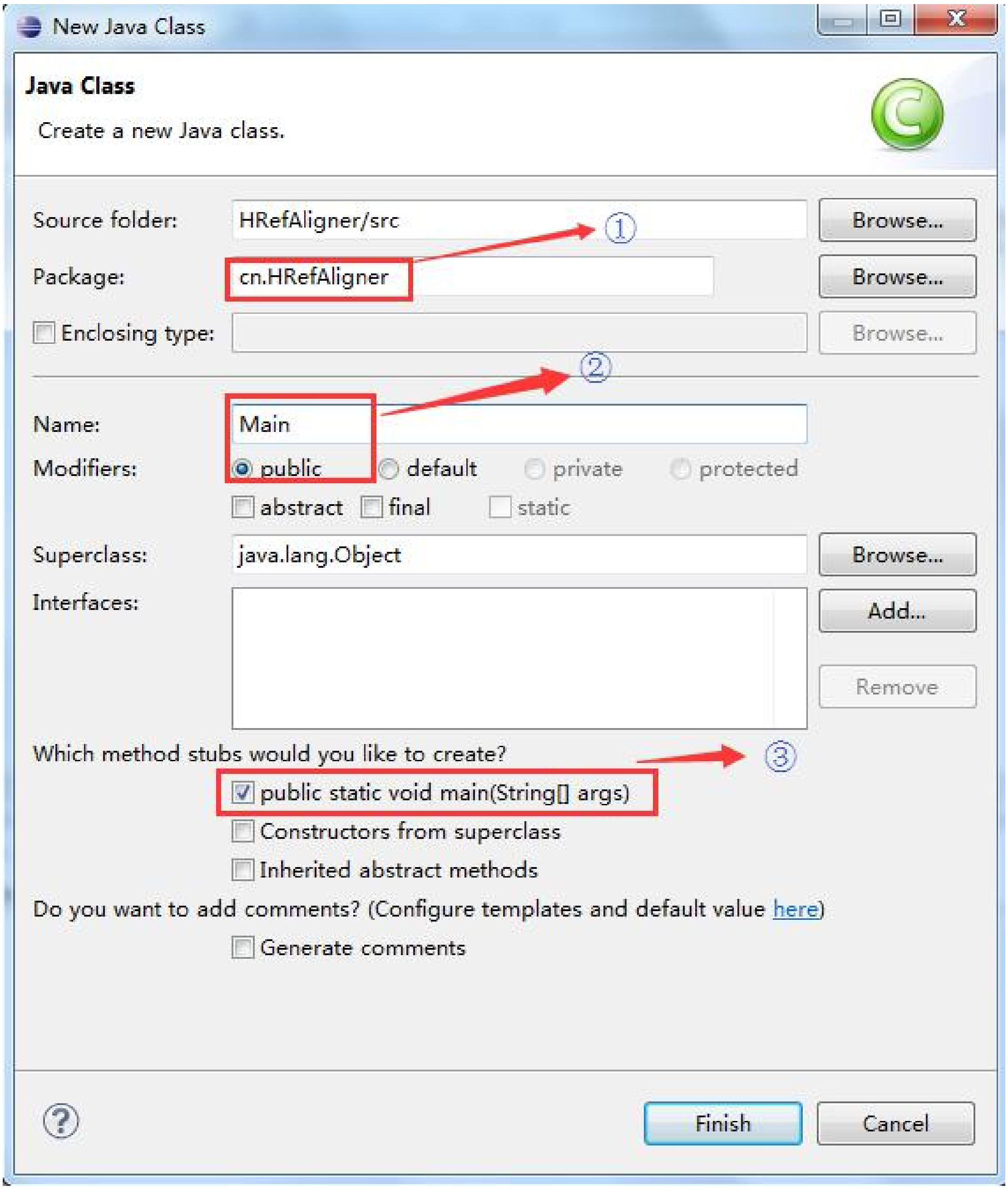
## III Creating a test class “Main” under src folder

1 You will create a new java class, called Main, under src folder.

On this page(Fig.6),

- type "**cn.HRefAligner**" in the Package field.
- type "**Main**" in the Name field.
- Choose “**public static void main(string[ ]args)**”.

Then click **Finish.**



New Java Class

Java Class

Create a new Java class.

Source folder: HRefAligner/src

Package: cn.HRefAligner

□ Enclosing type:

Name: Main

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

□ Constructors from superclass

□ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

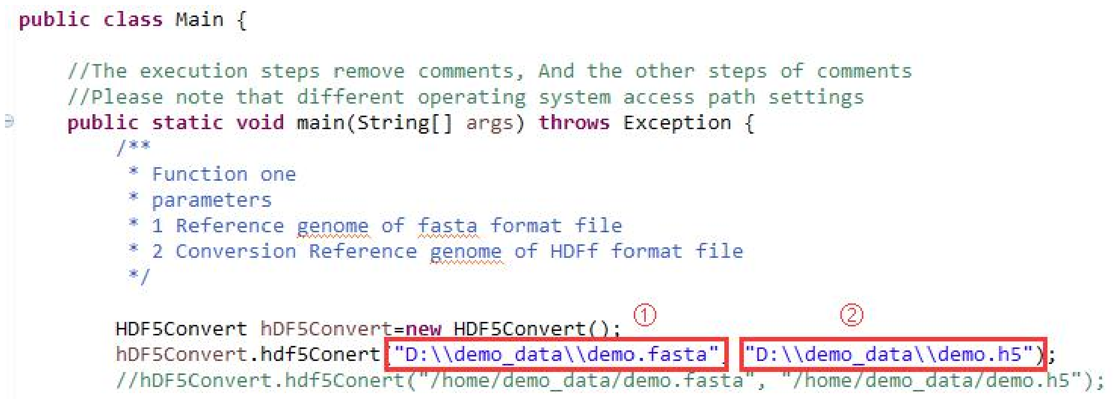
□ Generate comments

Fig.6 Creating a Main class

# IV Testing

1 You will change file format from .fasta to .hdf5(Fig.7).

- type the pathway of input file in ①, e.g. "demo.fasta"
- type the pathway of output file in ②, e.g. "demo.h5"

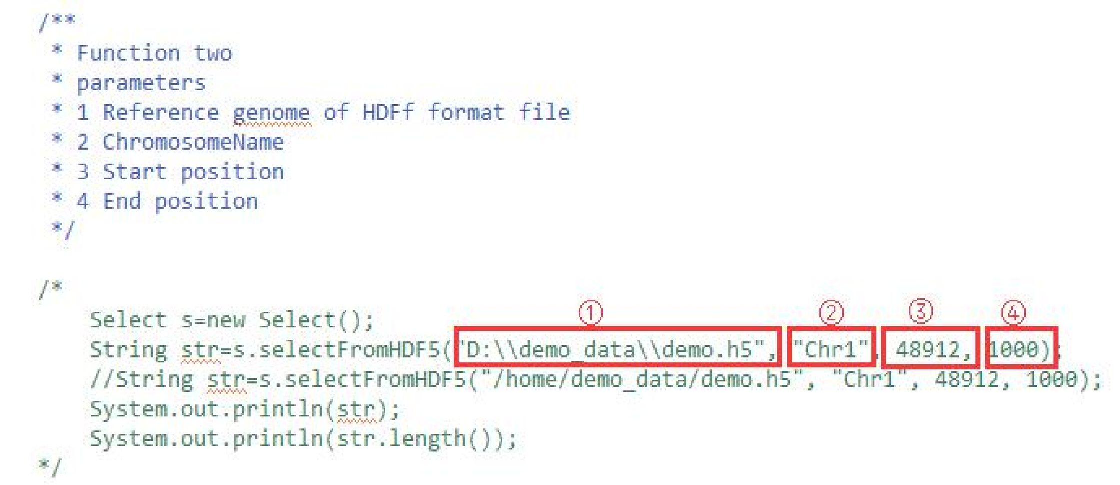


```
public class Main {

    //The execution steps remove comments, And the other steps of comments
    //Please note that different operating system access path settings
    public static void main(String[] args) throws Exception {
        /**
         * Function one
         * parameters
         * 1 Reference genome of fasta format file
         * 2 Conversion Reference genome of HDFf format file
         */

        HDF5Convert hDF5Convert=new HDF5Convert();
        hDF5Convert.hdf5Conert("D:\\demo_data\\demo.fasta", "D:\\demo_data\\demo.h5");
        //hDF5Convert.hdf5Conert("/home/demo_data/demo.fasta", "/home/demo_data/demo.h5");
```

Fig.7 Setting the input and output pathway

2 You will query arbitrary-length segments of .h5 data(Fig.8).

- type the pathway of .h5 file in ①.
- type chromosome name which you want to query in ②.
- type the starting position and the query length in ③,④ respectively.

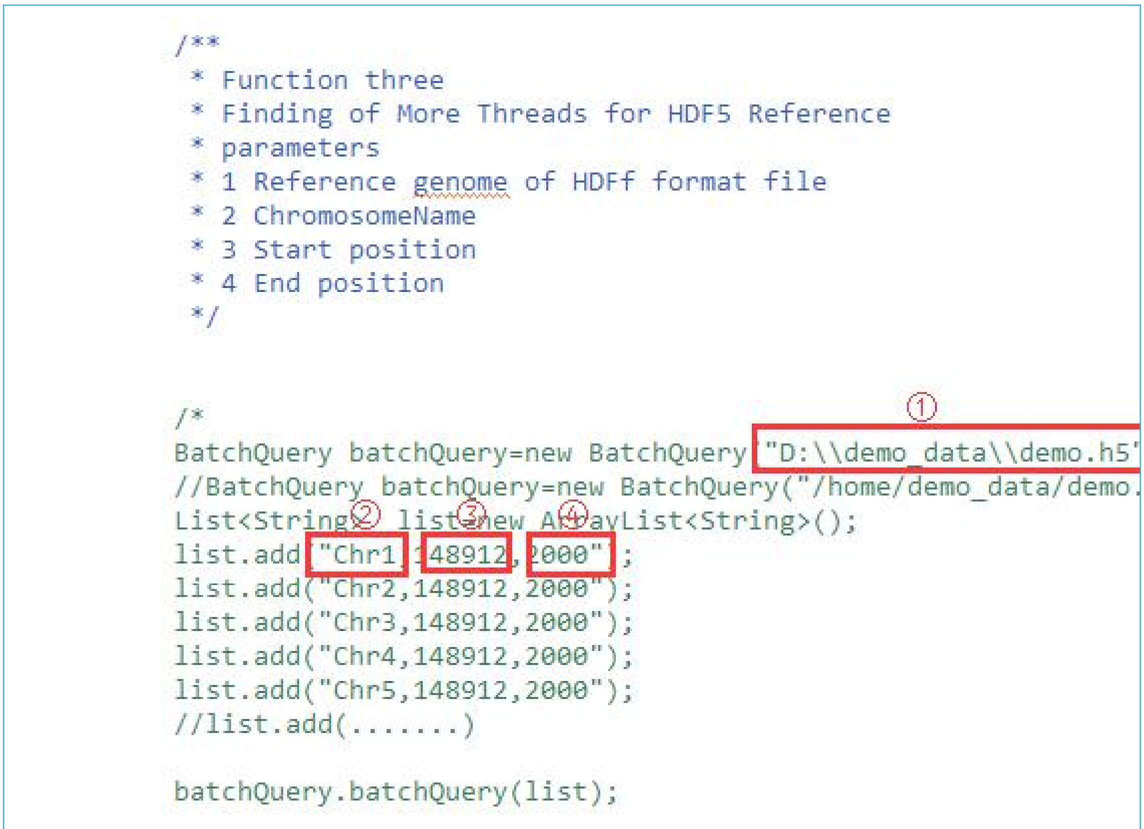


```
/**
 * Function two
 * parameters
 * 1 Reference genome of HDFf format file
 * 2 ChromosomeName
 * 3 Start position
 * 4 End position
 */

/*
    Select s=new Select();
    String str=s.selectFromHDF5("D:\\demo_data\\demo.h5", "Chr1", 48912, 1000);
    //String str=s.selectFromHDF5("/home/demo_data/demo.h5", "Chr1", 48912, 1000);
    System.out.println(str);
    System.out.println(str.length());
*/
```

Fig.8 Querying a segment of .h5 data

3 You will batch query in .h5 file(Fig.9).

- type the pathway of .h5 file in ①.
- type query list including reads with chromosome names, starting positions and query lengths in ②.
- type "batchQuery.batchQuery(list);" in ③ to batch query.

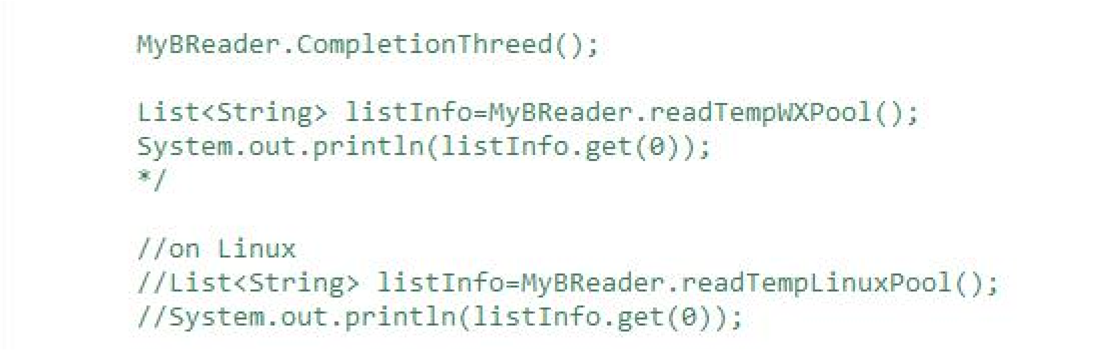


```
/**
 * Function three
 * Finding of More Threads for HDF5 Reference
 * parameters
 * 1 Reference genome of HDFf format file
 * 2 ChromosomeName
 * 3 Start position
 * 4 End position
 */


/*
BatchQuery batchQuery=new BatchQuery("D:\\demo_data\\demo.h5"
//BatchQuery batchQuery=new BatchQuery("/home/demo_data/demo.
List<String> list=new ArrayList<String>();
list.add("Chr1,148912,2000");
list.add("Chr2,148912,2000");
list.add("Chr3,148912,2000");
list.add("Chr4,148912,2000");
list.add("Chr5,148912,2000");
//list.add(.......)

batchQuery.batchQuery(list);
```

Fig.9 Batch query reads of .h5 reference file

4 You can check the results of query as below codes(Fig.10).

Note: The comments presents query code in Linux.



```
MyBReader.CompletionThreed();

List<String> listInfo=MyBReader.readTempWXPool();
System.out.println(listInfo.get(0));
*/

//on Linux
//List<String> listInfo=MyBReader.readTempLinuxPool();
//System.out.println(listInfo.get(0));
```

Fig.10 Checking query results

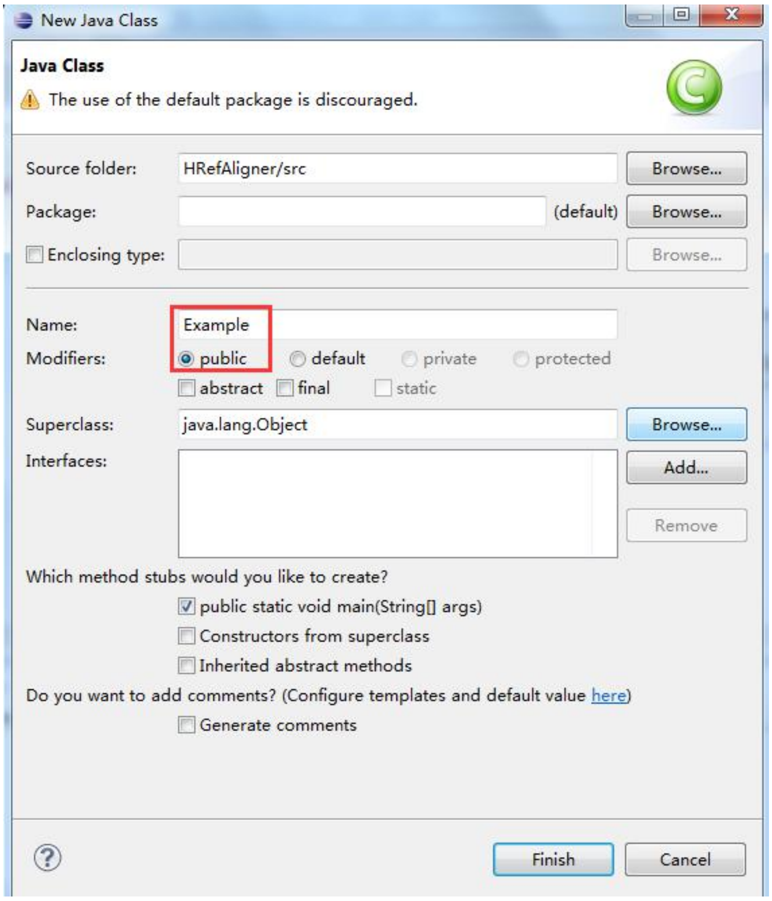
## V Multi-threaded demo-Creating a class to produce .sam file

1 You will create or open an “Example” class.

In this page(Fig.11),

- type "**Example**" in the Name field
- Choose “**public**” in the Modifiers field.

Then click **Finish**.



New Java Class

**Java Class**

⚠ The use of the default package is discouraged.

Source folder: HRefAligner/src [Browse...]

Package: (default) [Browse...]

□ Enclosing type: [Browse...]

Name: Example

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object [Browse...]

Interfaces: [Add...] [Remove]

Which method stubs would you like to create?

☑ public static void main(String[] args)

□ Constructors from superclass

□ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

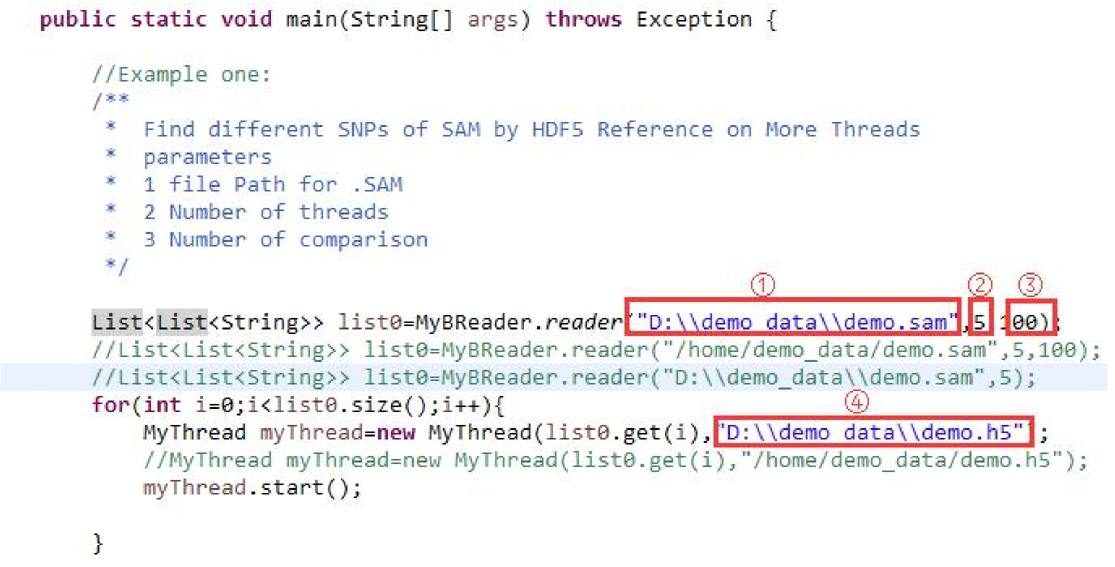
[Finish] [Cancel]

Fig.11 Creating an Example Java class

2 You will align .h5 reference sequence in multi-threaded mode(Fig.12).

- type the pathway of .sam file in ①.
- type the thread number which you want to use in ②.
- type the read start line number in ③.
- type the read end line number in ④.
- type the pathway of .h5 reference file in ⑤.

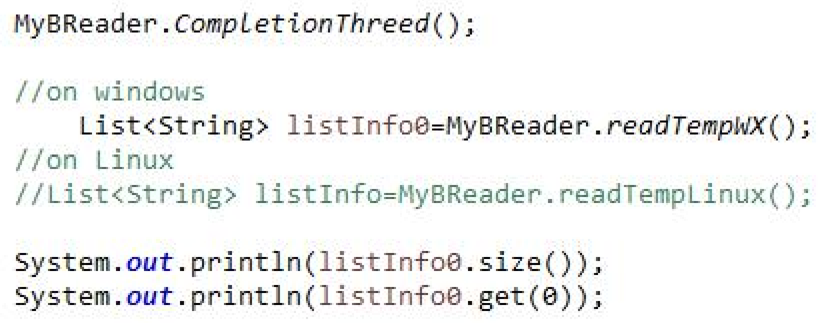
Note: The comments presents query code in Linux.



```
public static void main(String[] args) throws Exception {

    //Example one:
    /**
     *  Find different SNPs of SAM by HDF5 Reference on More Threads
     *  parameters
     *  1 file Path for .SAM
     *  2 Number of threads
     *  3 Number of comparison
     */

    List<List<String>> list0=MyBReader.reader("D:\\demo_data\\demo.sam",5,100);
    //List<List<String>> list0=MyBReader.reader("/home/demo_data/demo.sam",5,100);
    //List<List<String>> list0=MyBReader.reader("D:\\demo_data\\demo.sam",5);
    for(int i=0;i<list0.size();i++){
        MyThread myThread=new MyThread(list0.get(i),"D:\\demo_data\\demo.h5");
        //MyThread myThread=new MyThread(list0.get(i),"/home/demo_data/demo.h5");
        myThread.start();

    }
```

Fig.12 Multi-threaded query

3 You will check the results of muti-threaded query (Fig.13).



```
MyBReader.CompletionThreed();

//on windows
    List<String> listInfo0=MyBReader.readTempWX();
//on Linux
//List<String> listInfo=MyBReader.readTempLinux();

System.out.println(listInfo0.size());
System.out.println(listInfo0.get(0));
```

Fig.13 Checking query results

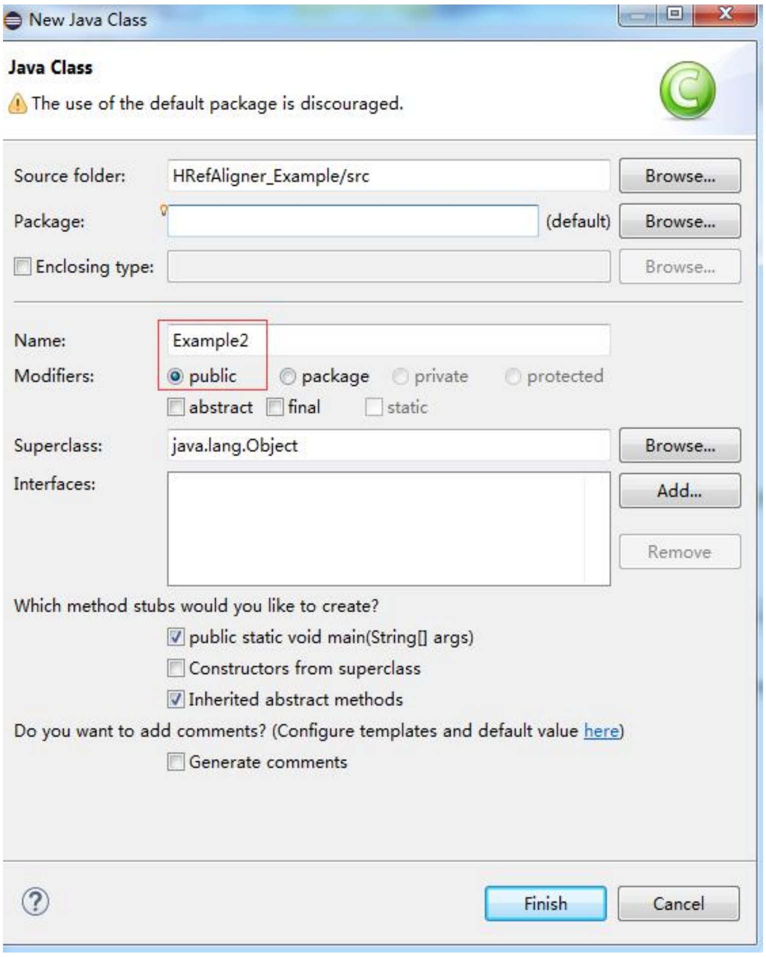
## VI Search Loci and create .sam files

1 You will create or open an “Example2” class.

In this page(Fig.11),

- type **"Example2"** in the Name field
- Choose **“public”** in the Modifiers field.

Then click **Finish**.



New Java Class

Java Class

The use of the default package is discouraged.

Source folder: HRefAligner_Example/src

Package: (default)

□ Enclosing type:

Name: Example2

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

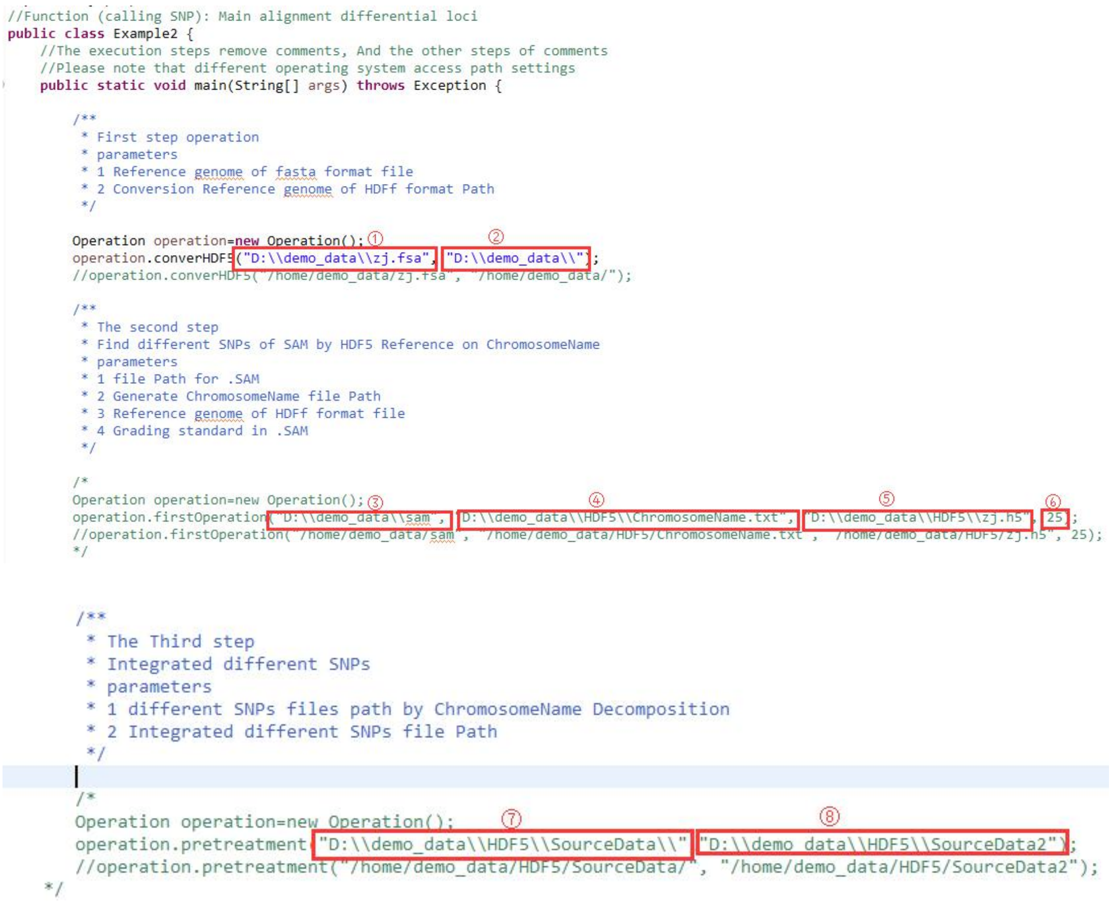
□ Generate comments

Fig.14 Creating an Example Java class

2 You will align .h5 reference sequence to Compare SAM file and Save different SNPs mode(Fig.12).

- The name of fasta reference files①.
- The path of creating responding hdf5 file ②.
- The path of .sam file③.
- Creating chromosome mapping file firstly④.
- The Reference genome of HDFf format File⑤.
- The grading standard of searching Loci ⑥.
- The path of creating .sam files according to chromosome⑦.
- The path of creating Loci file through integration .sam files⑧.

Note: The comments presents query code in Linux.



```
//Function (calling SNP): Main alignment differential loci
public class Example2 {
    //The execution steps remove comments, And the other steps of comments
    //Please note that different operating system access path settings
    public static void main(String[] args) throws Exception {

        /**
         * First step operation
         * parameters
         * 1 Reference genome of fasta format file
         * 2 Conversion Reference genome of HDFf format Path
         */

        Operation operation=new Operation();
        operation.converHDF5("D:\\demo_data\\zj.fsa", "D:\\demo_data\\");
        //operation.converHDF5("/home/demo_data/zj.fsa", "/home/demo_data/");

        /**
         * The second step
         * Find different SNPs of SAM by HDF5 Reference on ChromosomeName
         * parameters
         * 1 file Path for .SAM
         * 2 Generate ChromosomeName file Path
         * 3 Reference genome of HDFf format file
         * 4 Grading standard in .SAM
         */

        /*
        Operation operation=new Operation();
        operation.firstOperation("D:\\demo_data\\sam", "D:\\demo_data\\HDF5\\ChromosomeName.txt", "D:\\demo_data\\HDF5\\zj.h5", 25);
        //operation.firstOperation("/home/demo_data/sam", "/home/demo_data/HDF5/ChromosomeName.txt", "/home/demo_data/HDF5/zj.h5", 25);
        */


        /**
         * The Third step
         * Integrated different SNPs
         * parameters
         * 1 different SNPs files path by ChromosomeName Decomposition
         * 2 Integrated different SNPs file Path
         */

        /*
        Operation operation=new Operation();
        operation.pretreatment("D:\\demo_data\\HDF5\\SourceData\\", "D:\\demo_data\\HDF5\\SourceData2");
        //operation.pretreatment("/home/demo_data/HDF5/SourceData/", "/home/demo_data/HDF5/SourceData2");
    */
```

Fig.15 Compare SAM file to find different SNPs

# VII Demo Datas

Table 1 Arabidopsis Fasta Reference file format

| Fasta Reference |
|---|
| >Chr1 CHROMOSOME dumped from ADB: Jun/20/09 14:53; last updated: 2009-02-02 |
| CCCTAAACCCTAAACCCTAAACCCTAAACCTCTGAATCCTTAATCCCTAAATCCCTAAATCTTTAAATCCTACATCCAT |
| ATCGTTTTTATGTAATTGCTTATTGTTGTGTGTAGATTTTTTAAAAATATCATTTGAGGTCAATACAAATCCTATTTCT |
| >Chr2 CHROMOSOME dumped from ADB: Jun/20/09 14:54; last updated: 2009-02-02 |
| NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNGAATTCGTCGACCAGGACGGCGGAATG |
| CTCGACCAGGACGATGAATGGGCGATGAAAATCTATCGGGTTAGAGGAATGGTCGACCGGGTCCGAGAATTCGTCGACC |
| AGGACGAGGAGTGGTCGAGGATTTGTCGACCAGGAGTTGAAATCGTCGACCGGGTCCGAGAATTCGTCGACCAGGACGG |

Table 2 Arabidopsis SAM file format

```
@HD     VN:1.0  SO:unsorted
@SQ     SN:Chr1 LN:30427671
@SQ     SN:Chr2 LN:19698289
@SQ     SN:Chr3 LN:23459830
@SQ     SN:Chr4 LN:18585056
@SQ     SN:Chr5 LN:26975502
@SQ     SN:chloroplastLN:154478
@SQ     SN:mitochondria   LN:366924
@PG     ID:bowtie2    PN:bowtie2    VN:2.0.6
```

```
SRR388657.1    0   chloroplast   26003      42  20M  * 0 0 NGAATTCATTAAAGGAATGT   &4--/77777=2<<2FFF2F   AS:i:-1
SRR388657.2    0   Chr5          24549687   42  20M  * 0 0 NAGATGTTTTGTTCTTGTTG   ####################   AS:i:-1
SRR388657.3    4   *             0          0   *    * 0 0 NATTTAAGTTTTGAGATGCA   ####################   YT:Z:UU
SRR388657.4    0   Chr3          11242799   42  20M  * 0 0 NGCAAAATAATGAATATACT   ####################   AS:i:-1
SRR388657.5    16  Chr1          17061194   42  20M  * 0 0 AAAAGCTCCACTGTCACTGN   ####################   AS:i:-1
SRR388657.8    16  chloroplast   4634       42  20M  * 0 0 ATCCATTTTTTTTATGGCCT   GGA@4;B??>DAC20?C7??   AS:i:0
SRR388657.9    0   Chr2          11642498   42  20M  * 0 0 ACGAAATTTATTTGATATCT   HHHHBCHHHBGHCCHDGHFH   AS:i:0
SRR388657.10   16  mitochondria  55638      1   20M  * 0 0 TCACGTTCTGATACCTATAT   =AEABE*=EBDDF>FEF=FB   AS:i:-2
```