AlternateScript\_UsingMapper

This is an alternate script to retrieve the same information in a different way

* Script Execution
* Introduction and purpose of my script
* Concept Implemented
* List of Objects

Script Execution:

This is a normal simple perl script which can be executed in the command line with the below parameters,

***perl xxxxx.pl arg1 arg2 arg3***

arg1->(1-22,x,y)

arg2->(Starting\_Coordinate)

arg3->(Ending\_Coordinate)

*Example : perl xxxxx.pl 12 63840524 80294266*

Introduction and purpose of my script:

The purpose of the script is to convert or map the GRch38 co-ordinates to the older assembly (GRch37) to its corresponding region. The perl script receives the co-ordinates in the above format and finally returns the mapped co-ordinates which refers to the older assembly (GRch37)

Concept Implemented:

This script works on the baseline with some of the Assembly Mapper APIs present in the registry.

Step 1: At first the Assembly Mapper and Coordinate system adaptors are created.

Step 2: Next is creating variables for the source and target coordinate systems.

Step 3: Creating Assembly mapper variable to define the mapping between (from - to) coordinate system

Step 4: Finally we use map() function to convert the coordinates

The output of this function has two modes which can either return a normal value or hash value which can be given in input argument as Boolean (0 or 1)

List of used objects :

*Adaptors* : AssemblyMapper , Coordsystem

*Methods* : fetch\_by\_name(), fetch\_by\_CoordSystems(), map()

*Array Variables* : @new\_coords, which holds the coordinate values (Hash or direct value)

Points Analysed:

* The output should only be stored in an array variable
* Hash value or original value can be changed in the parameter below

$asm\_mapper->map ($ARGV[0], $ARGV[1], $ARGV[2],1, $chr\_cs\_37, , , 1 );

* Fastmap(), map\_coordinates are similar methods used to retrieve coordinates
* There is a special slice argument present for map() method and it should be a target slice
* The slice and its argument is currently commented in my code
* There is also an Assembly projector from which we can very easily map the coordinates but DBSQL access required to do that
* For some coordinates method returns null value , but the same method successfully returns a hash value for all of the input coordinates
* The fact it fails with some coordinates is because it is unable to get the access get\_seq\_region\_id method
* I considered this script as an alternate way or an idea to get the coordinates and would need some more time to dig up the APIs to make it perfect