Using org.lcsim to analyse the Stack data

First of all, you need the make the detector tag known to org.lcsim, which is done in

.lcsim/alias.properties

and add a line to it in the format detector tag: directory containing the geometry files (compact.xml, clichcalstack.heprep)

```
clichcalstack: file:///home/marcel/demo/stack
```

this allows you to use the correct geometry. Analysisng then is quite straightforward, the following driver shows some ideas

```
 import hep.aida.IHistogram1D;
import java.util.List;
import org.lcsim.event.EventHeader;
import org.lcsim.event.SimCalorimeterHit;
import org.lcsim.geometry.util.CalorimeterIDDecoder;
import org.lcsim.util.Driver;
import org.lcsim.util.aida.AIDA;


public class StackAnalysis extends Driver{
        String calcolname = "HCALCalorimeterHits";
        private AIDA aida = AIDA.defaultInstance();
        public StackAnalysis(){


        }
        protected void process(EventHeader event) {
                super.process(event);

                List<SimCalorimeterHit>calorimeterhits =
                        event.get(SimCalorimeterHit.class,calcolname);
// make a histogramm
                IHistogram1D hitlayer=aida.histogram1D("Layer Profile", 100, 0, 100);
                IHistogram1D hit_x=aida.histogram1D("Hit Pos X", 500, -500, 500);
                IHistogram1D hit_y=aida.histogram1D("Hit Pos Y",  500, -500, 500);
                IHistogram1D hit_z=aida.histogram1D("Hit Pos Z", 1000, 2000, 5000);
// loop over hits ...
                for (SimCalorimeterHit calhit: calorimeterhits) {
                        aida.cloud1D("Hit Energy").fill(calhit.getRawEnergy());
// get the CellID decoder
                        CalorimeterIDDecoder iddecoder = (CalorimeterIDDecoder)calhit.getIDDecoder();
                        iddecoder.setID(calhit.getCellID());
//get the layer
                        hitlayer.fill(iddecoder.getLayer());
// get x and y
                        hit_x.fill(calhit.getPosition()[0]);
                        hit_y.fill(calhit.getPosition()[1]);
                        aida.cloud2D("Hit x vs y").fill(calhit.getPosition()[0],calhit.getPosition()[1]);
                        hit_z.fill(calhit.getPosition()[2]);
                        aida.cloud2D("Hit z vs layer").fill(calhit.getPosition()[2],iddecoder.getLayer());
// get the timing
                        aida.cloud1D("Hit Timing").fill(calhit.getTime());
                }

        }
}
```