

MapReduce and DataSets

Module Objectives

In this module, you will look at:

- How to use DataSets with MapReduce
 - Declaring a DataSet
 - Injecting a DataSet
-

MapReduce and DataSets

Both Continuity Reactor `Mapper` and `Reducer` can directly read from a `DataSet` or write to a `DataSet` similar to the way a `Flowlet` or `Procedure` can

To access a `DataSet` directly in `Mapper` or `Reducer`, you need:

1. a declaration and
 2. an injection
-

MapReduce Declaration

1. Declare the DataSet in the MapReduce job's `configure()` method. For example, to have access to a DataSet named *catalog*:

```
public class MyMapReduceJob implements MapReduce {  
    @Override  
    public MapReduceSpecification configure() {  
        return MapReduceSpecification.Builder.with()  
            ...  
            .useDataSet("catalog")  
            ...  
    }  
}
```

MapReduce Injection

2. Inject the DataSet into the mapper or reducer that uses it:

```
public static class CatalogJoinMapper extends Mapper<byte[], Purchase, ...> {
    @UseDataSet("catalog")
    private ProductCatalog catalog;

    @Override
    public void map(byte[] key, Purchase purchase, Context context)
        throws IOException, InterruptedException {
        // join with catalog by product ID
        Product product = catalog.read(purchase.getProductId());
        ...
    }
}
```

Module Summary

You should be able:

- To use DataSets with MapReduce
 - Declare a DataSet
 - Inject a DataSet at runtime
-

Module Completed