Discussion: what kind of language/syntax do we need?

- Most generic and economic ways to cover the widest collection of needs from the biggest variety of analyses?
- Objects, composite objects, event variables, definitions, selections, event weighting, …
- Intrinsic loops, reducers, optimizers, object combinations/permutations/partitioning, ternary operations, …
- Expressing the event input
- Expressing external functions

Consider

```
printf("Error 0x%04x: %s", id, errors[id]);
```

vs

```
std::cout << "Error 0x" << std::hex << std::setfill('0')
          << std::setw(4) << id << ": " << errors[id] << std::endl;
```

Consider

```
import pyspark
pyspark.sql("""
   SELECT CONCAT(first, " ", last) AS fullname, AVG(age)
      FROM my_table WHERE age BETWEEN 18 AND 24
      GROUP BY fullname
""")
```

vs

```
import pyspark.sql.functions as F
df = pyspark.read.load("my_table")
(df.withColumn("fullname",
       F.concat(F.col("first"), F.lit(" "), F.col("last")))
  .select("fullname", "age")
  .where(df.age.between(18, 24))
  .groupBy("fullname")
  .agg(F.mean("age")))
```

We want this
Select cut1
Select cut2

Vs

We don’t want this because it brings ambiguity
Function f (
If cut1
If cut2
)

Important to know who is in control: GPL or DSL?

```
run = adl.interpreter.Run("""
region "two muons": muons.size >= 2
{

  zmass := muons.distincts                                  # make pairs of distinct muons
                .map(pair => (pair[0] + pair[1]))           # add the Lorentz vectors
                .maxby(zcandidate => zcandidate.pt)         # find the maximum by pt
                .mass                                       # but compute and return mass

  count "zmass" by
     regular(60, 0, 120) <- zmass

}
""")
run(muons = muons)
run["two muons", "zmass"].plot()
```