TemporalField vs. Temporal Unit
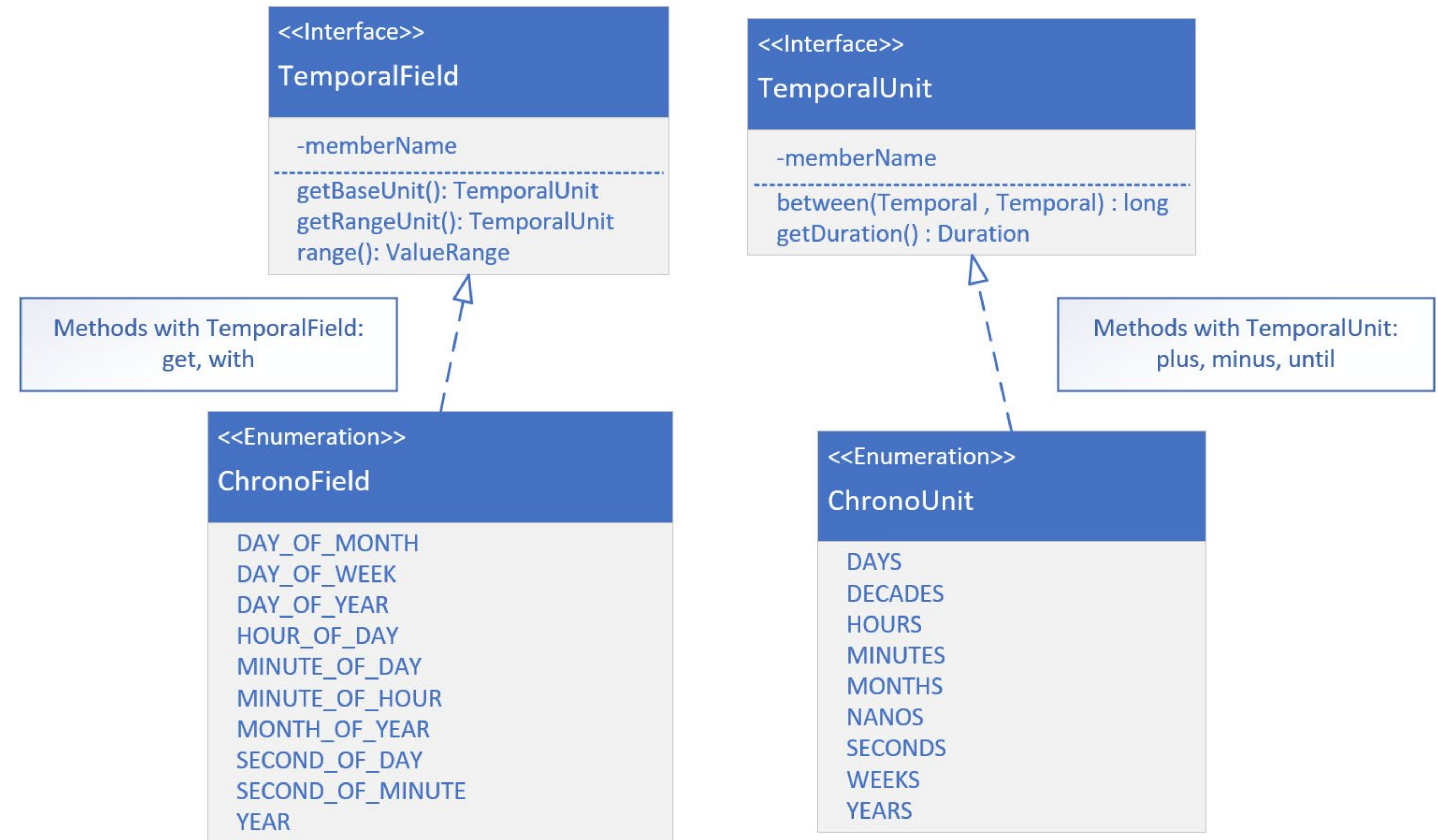
A temporal field represents **a field of date-time**, such as month-of-year, or minute-of-hour.

A temporal unit is **a measurement of time**, such as years, months, days, hours, minutes and seconds.

On this slide, I show both interfaces, and a couple of the methods that would be implemented.

<<Interface>>
TemporalField

-memberName
----
getBaseUnit(): TemporalUnit
getRangeUnit(): TemporalUnit
range(): ValueRange

Methods with TemporalField:
get, with

<<Enumeration>>
ChronoField

DAY_OF_MONTH
DAY_OF_WEEK
DAY_OF_YEAR
HOUR_OF_DAY
MINUTE_OF_DAY
MINUTE_OF_HOUR
MONTH_OF_YEAR
SECOND_OF_DAY
SECOND_OF_MINUTE
YEAR

<<Interface>>
TemporalUnit

-memberName
----
between(Temporal , Temporal) : long
getDuration() : Duration

Methods with TemporalUnit:
plus, minus, until

<<Enumeration>>
ChronoUnit

DAYS
DECADES
HOURS
MINUTES
MONTHS
NANOS
SECONDS
WEEKS
YEARS

{LP} LearnProgramming
.academy

TemporalField vs. Temporal Unit

I'm also showing you just some of the enum constants on ChronoField and ChronoUnit, ones that are pretty commonly used.

The get and with methods take a TemporalField as a parameter.

The plus, minus, and until methods, take a TemporalUnit.


```
<<Interface>>
TemporalField

-memberName
- - - - - - - - - - - - - - - - - -
getBaseUnit(): TemporalUnit
getRangeUnit(): TemporalUnit
range(): ValueRange
```

Methods with TemporalField:
get, with

```
<<Enumeration>>
ChronoField

DAY_OF_MONTH
DAY_OF_WEEK
DAY_OF_YEAR
HOUR_OF_DAY
MINUTE_OF_DAY
MINUTE_OF_HOUR
MONTH_OF_YEAR
SECOND_OF_DAY
SECOND_OF_MINUTE
YEAR
```

```
<<Interface>>
TemporalUnit

-memberName
- - - - - - - - - - - - - - - - - -
between(Temporal , Temporal) : long
getDuration() : Duration
```

Methods with TemporalUnit:
plus, minus, until

```
<<Enumeration>>
ChronoUnit

DAYS
DECADES
HOURS
MINUTES
MONTHS
NANOS
SECONDS
WEEKS
YEARS
```

{LP} LearnProgramming
.academy