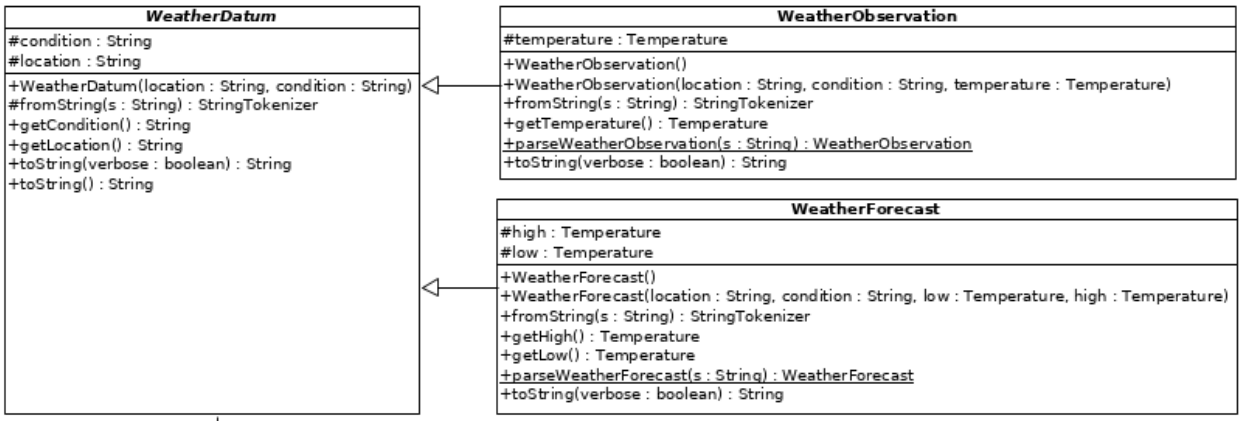
`weather` Package v1.0

Class Diagram

The relationships between the various classes in this package are illustrated in the following UML class diagram. (Note: Though it is not shown in the diagram, all of these classes must be in the `weather` package.)



The `StringTokenizer` class referred to in this diagram is in the Java API.

In addition to the specifications that are contained in this class diagram, the implementation must comply with the following specifications.

The `WeatherDatum` Class

A `WeatherDatum` object is an abstract piece of weather information.

Instance Variables

This class must, at a minimum, contain the following instance variables. It may contain other instance variables as well.

`location` A `String` containing the code for the observation site.

`condition` A `String` containing the condition (e.g., `"Sunny"`).

Methods

`fromString(final String s)`

A setter method that parses the terse `String` representation of a `WeatherDatum` and sets the attributes of the owning object accordingly. If the `String` representation contains too few fields then this method must leave the **remaining attributes** of the owning object unchanged and return `null`. Otherwise, it must return the `StringTokenizer` object that was used to tokenize the `String`.

`toString(final boolean verbose)`

Returns either a terse or verbose `String` representation of the owning object. The terse representation must consist of the `location`, followed by a comma and the `condition`. For example:

```
"PWW02,Sunny"
```

The verbose representation must consist of the `String` `"Location: "` (note there must be exactly one space after the colon), followed by the `location`, followed by a tab, followed by the `String` `"Condition: "` (note there must be exactly one space after the colon), followed by the `condition`. For example:

```
"Location: PWW02\tCondition: Sunny"
```

`toString()`

Must return a terse String representation of the owning object.

The `WeatherObservation` Class

An observation of the actual weather conditions at a particular location (at a particular time).

The Default Constructor

The default constructor must create an instance with a location of `"XXX"`, a `condition` of `"Unknown"`, and a default `temperature`.

Methods

```
fromString(final String s)
```

A setter method that parses the terse `String` representation of a `WeatherObservation` and sets the attributes of the owning object accordingly. If the `String` representation contains too few fields then this method must leave the **remaining attributes** of the owning object unchanged and return `null`. Otherwise, it must return the `StringTokenizer` object that was used to tokenize the `String`. If the `String` representation of the `temperature` attribute is invalid, it must change the `temperature` attribute to the default `Temperature`.

```
parseWeatherObservation(final String s)
```

Must create a `WeatherObservation` object from a terse `String` representation. It must conform to the specifications of the default constructor and the `fromString()` method.

```
toString(final boolean verbose)
```

Must return either a terse or verbose `String` representation of the owning object. The terse representation must consist of the terse representation of the parent class followed by a comma and the `String` representation of the `temperature`. For example:

```
"PWW02,Sunny, +86.7F"
```

The verbose representation must consist of the verbose representation of the parent class followed by a tab, followed by the `String` literal `"Temperature: "` (note there must be exactly one space after the colon), followed by the `String` representation of the `temperature`. For example:

```
"Location: PWW02\tCondition: Sunny\tTemperature:  +86.7F"
```

The `WeatherForecast` Class

A prediction/forecast of what the weather conditions will be at a particular location (during a particular interval time, like a day).

The Default Constructor

The default constructor must create an instance with a location of `"XXX"`, a `condition` of `"Unknown"`, and default `high` and `low` temperatures.

Methods

```
fromString(final String s)
```

A setter method that parses the terse `String` representation of a `WeatherForecast` object and sets the attributes of the owning object accordingly. If the `String` representation contains too few fields then this method must leave the **remaining attributes** of the owning object unchanged and return `null`. Otherwise, it must return the `StringTokenizer` object that was used to tokenize the `String`. If the `String` representation of the `low` attribute is invalid, it must change the `low` attribute to the default `Temperature`. Similarly, if the `String` representation of the `high` attribute is invalid, it must change the `high` attribute to the default `Temperature`.

```
parseWeatherForecast(final String s)
```

Must create a `WeatherForecast` object from a terse `String` representation. It must conform to the specifications of the default constructor and the `fromString()` method.

```
toString(final boolean verbose)
```

Must return either a terse or verbose `String` representation of the owning object. The terse representation must consist of the terse representation of the parent class, followed by a comma, followed by the `String` representation of the `low`, followed by the `String` representation of the `high`. For example:

```
"PWW01,Sunny, +86.7F, +91.3F"
```

The verbose representation must consist of the verbose representation of the parent class, followed by a tab, followed by the `String` literal `"Low: "` (note there must be exactly one space after the colon), followed by the `String` representation of the `low`, followed by a tab, followed by the `String` literal `"High: "` (note there must be exactly one space after the colon), followed by the `String` representation of the `high`. For example:

```
"Location: PWW02\tCondition: Sunny\tLow:  +86.7F\tHigh:  +91.3F"
```