

11.1 Self-constrained virtual class

Description

As a first example of a virtual class attribute, we extend the travel booking system introduced in section . We will encapsulate details about a travel booking in a local nested virtual class `TravelInfo` as shown in the example below:

```
class TravelBooking: Booking
  class TravelInfo:<
    source, destination: ref City
    asString -> s: var String:<
      inner(asString)
    theCarrier: ref Carrier
    theTravel: obj TravelInfo
    confirm::<
      confirmText := confirmText + theTravel.asString
    cancel:
      theTravel.theCarrier.notifyCancellation(bookingRef)
```

The symbol '`<`' specifies that class `TravelInfo` is virtual in the same way as for virtual methods. `TravelInfo` has the following attributes:

- The source and destination of the travel.
- A method `asString`, that returns a text about travel details that may be emailed to the customer.
-

The virtual class `TravelInfo` is an example of a *self-constrained* virtual class, which is a virtual class that is constrained by an object descriptor specifying the virtual class itself, and this object-descriptor includes declarations and/or statements. This is in contrast to a class-constrained virtual class, which is constrained by a class name.

The specification of `TravelInfo` may be extended in subclasses of `TravelBooking`, just like virtual methods may be extended. `TravelInfo` can only be extensions of the object descriptor specifying the virtual class. A class-constrained virtual class can be extended to any subclass of the virtual class constraining the virtual class.

Below, the specification of is extended in a way which is similar to the way a virtual method may be extended:

```
class FlightBooking: TravelBooking
  class TravelInfo::<
    fromAirport, toAirport: ref Airport
    row: var integer
    seat: var char
    asString::<
      s := theCarrier.asString +
        " from " + fromAirport.name +
        " to " + toAirport.name
      inner(asString)
```

The extension (also called further binding) of `TravelInfo` works as follows:

- The symbol '`::<`' specify that this declaration of `TravelInfo` extends ('`::`') the one from the superclass `TravelBooking`, and that it is still virtual ('`<`').
- The extension includes reference variables `fromAirport` and `toAirport`.
- It includes a row and seat for this booking.



Extending TravelInfo in subclass of TravelBooking

The further binding/extension of TravelInfo in FlightBooking implies that the complete definition of TravelInfo in FlightBooking is as follows:

```
class TravelInfo:<
```

```
  source, destination: ref City
```

```
  theCarrier: ref Carrier
```

```
  fromAirport, toAirport: ref Airport
```

```
  asString:<
```

```
    s := theCarrier.asString +
```

```
    " from " + fromAirport.name +
```

```
    " to " + toAirport.name
```

```
    inner(asString)
```

} properties from virtual defi

} properties from extension

Also, note, that we in FlightBooking do *not* need to extend the virtual method confirm, since the information needed for confirm is computed by asString in TravelInfo in TravelBooking.

We may make a similar extension of TravelInfo in TrainBooking.

```
class TrainBooking: TravelBooking
```

```
  class TravelInfo:<
```

```
    fromStation, toStation: ref TrainStation
```

```
    waggon, seat: var
```

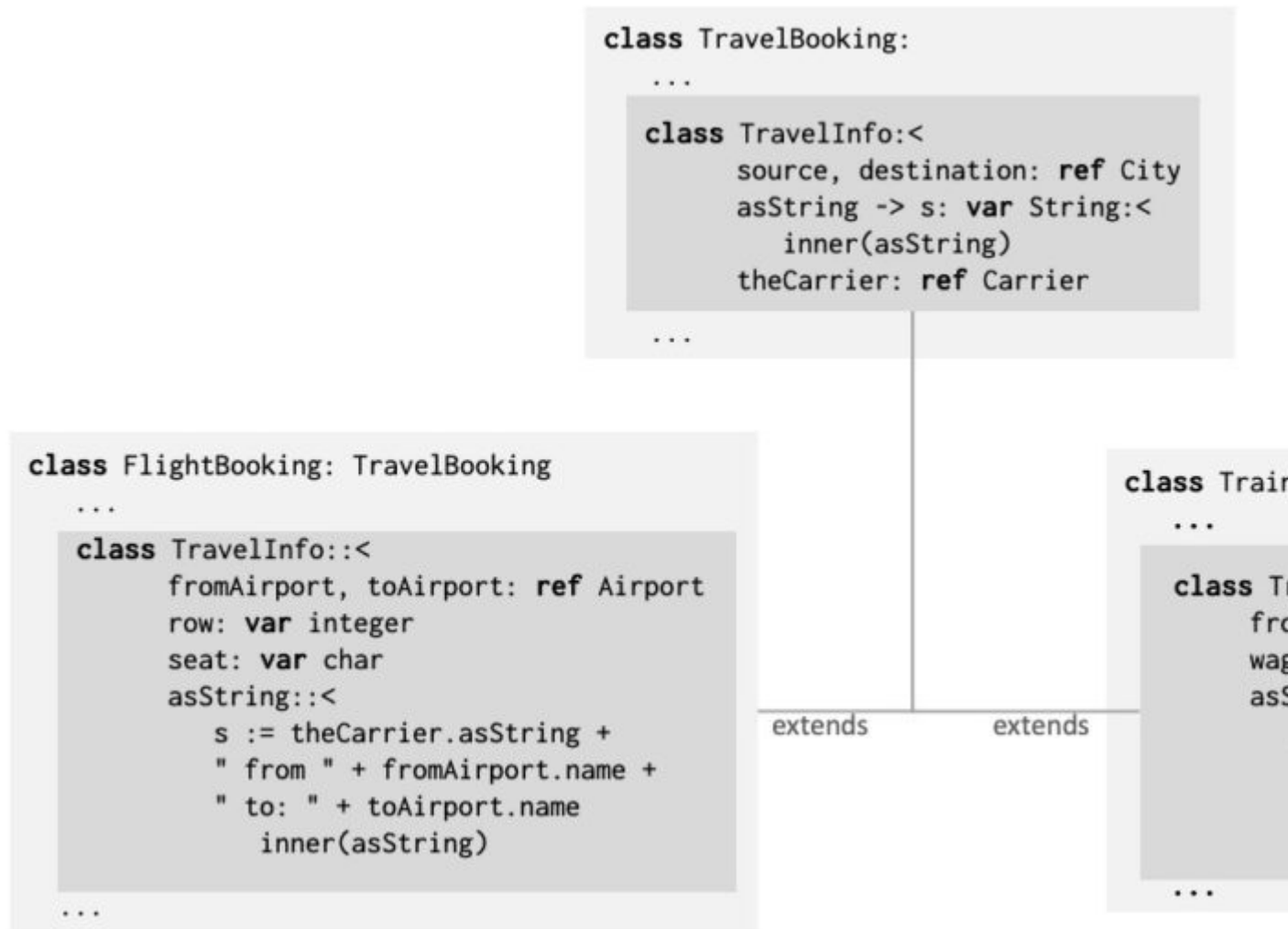
```
integer
```

```
  asString:<
```

```
    s := theCarrier.asString +
```

```
    " from: " + fromStation.name + " to: " + toStation.name
```

```
    inner(asString)
```

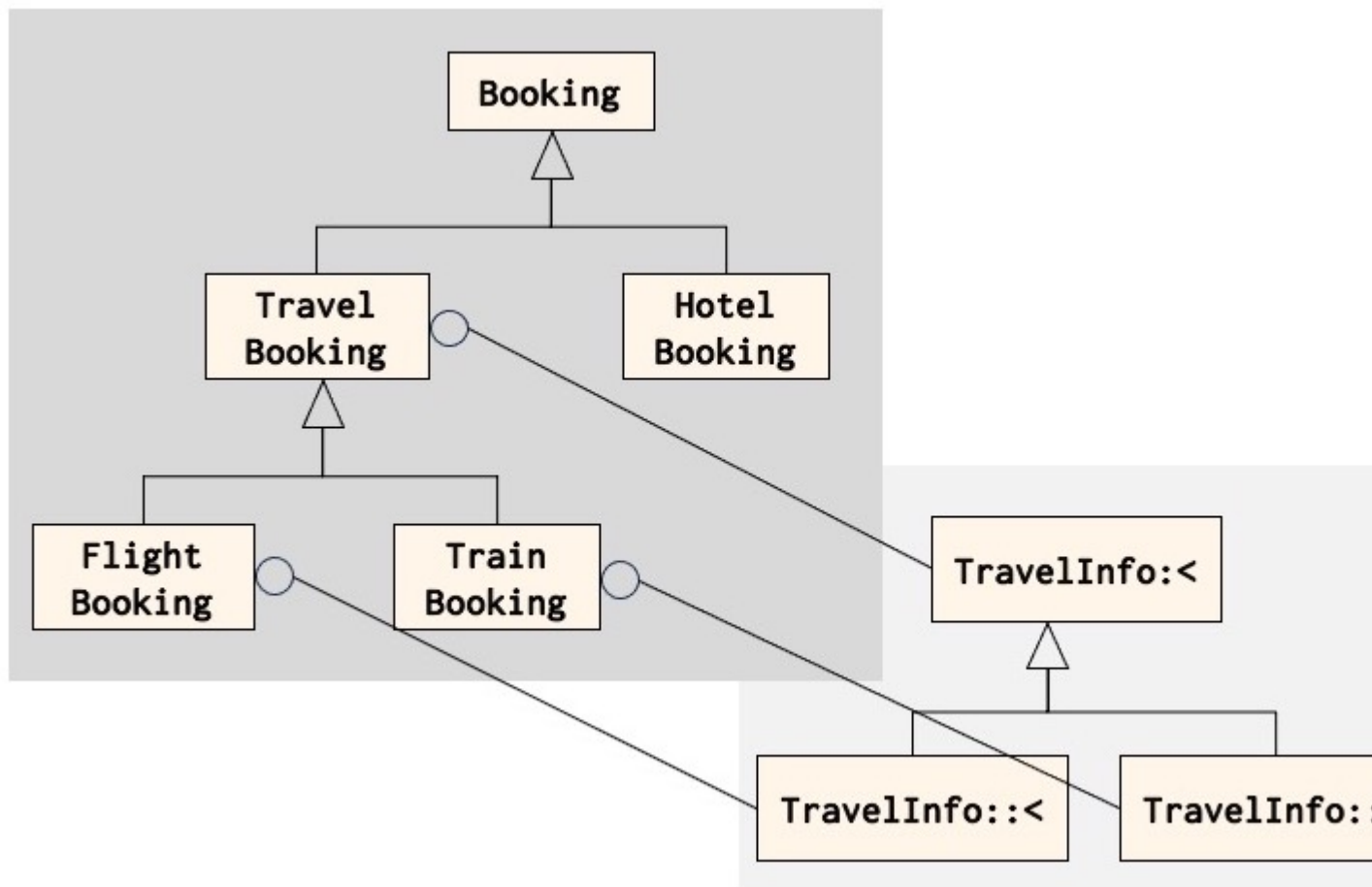


Extending TravelInfo in two subclasses of TravelBooking

In the above example, we use the class `Carrier`, which is only sketched below. We may make subclasses of `Carrier` representing airlines and railways as sketches below. The reference `theCarrier` in `FlightBooking` may then refer to an `Airline` object and in `TrainBooking` to a `Railway` object.

```

class Carrier:
    notifyCancellation(bookingRef: var String):
    ...
class Airline: Carrier
    ...
class Railway: Carrier
    ...
  
```



BookingAndTravelinfoHierarchyAndNesting/