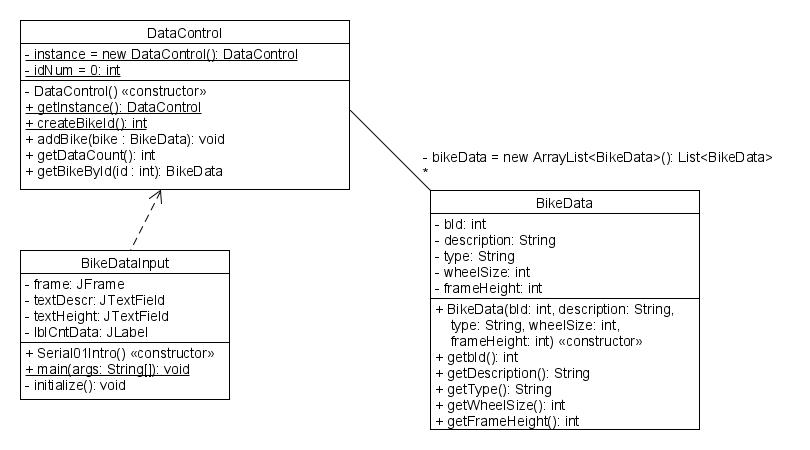
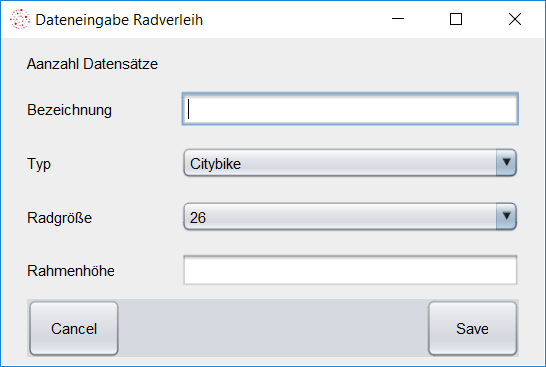
## Übergeordnete Aufgabenstellung

Wir haben ein Fenster zur Eingabe von Fahrradinformationen für ein Verwaltungsprogramm für einen Fahrradverleih erstellt. Dazu existiert der folgende UML-Entwurf:



Ein Swing-Fenster (BikeDataInput) als Application Window, eine Datenelementklasse für die Fahrraddaten (zunächst nur mit 4 Attributen) sowie eine eindeutige ID. Diese ID wird von einer Singleton-Instanz erzeugt, die auch die Verwaltung der Daten übernimmt. Da wir den Singleton über die statische Methode getInstance() aufrufen, können wir das an jeder beliebigen Stelle in BikeDataInput tun. Daher ist die Abhängigkeit zu DataControl explizit dargestellt.

Die Oberfläche soll in etwa so aussehen:

* zwei Textfelder für eine Bezeichnung oder den Hersteller sowie für die Rahmenhöhe;
* zwei Auswahlfelder für den Radtyp und den Laufraddurchmesser;
* Save speichert die eingegebenen Werte in einem neuen BikeData-Objekt und übergibt dieses Objekt an DataControl zum Ablegen in die Liste;
* Cancel bricht den Dialog ab und beendet das Fenster.

Im Verlauf einer Anwendungsentwicklung kommt es immer wieder vor, dass Änderungen vorgenommen werden oder Auswertungs-/Selektionsmethoden z. B. im Singleton hinzukommen.

So soll zunächst ein Filter erstellt werden, der alle Räder eines bestimmten Typs anzeigt.

**+ getBikesByType(type : String): List<BikeData>**

Es ist offensichtlich, dass es sehr lästig ist, für jeden Test einige Daten jedes Mal erneut einzugeben, vor allem, wenn das Programm beispielsweise aufgrund von Fehlern in der Programmierung abstürzt.

Besser ist es, wenn wir die Daten so speichern können, dass eingegebene Daten dauerhaft vorhanden bleiben, man spricht von persistenten Daten/Objekten.

Allerdings können wir Objekte nicht wie mit einer Druckform auf den Permanentspeicher schreiben. Darüber hinaus ist es wenig effizient, die Teile eines Objekts, das bei allen Objekten gleich sind (Methoden) ebenfalls abzuspeichern.

In den meisten Programmiersprachen hat man dazu einen Mechanismus realisiert, der unabhängig vom Übertragungsmedium immer gleich zu arbeiten scheint. Also egal, ob man es zu einem Permanentspeicher oder über eine Datenleitung senden möchte.

Als Hintergrundinformation liegt das Informationsblatt zur Datenübertragung vor.