write down statements about things identified by URLs. But what do those statements mean? How should a computer go about interpreting the statements made? These questions are discussed in the next section where we introduce a schema language for RDF.

2.4 RDFS: Adding Semantics

RDF is a universal language that lets users describe resources using their own vocabularies. RDF does not make assumptions about any particular application domain, nor does it define the semantics of any domain. In order to specify these semantics, a developer or user of RDF needs to define what those vocabularies mean in terms of a set of basic domain independent structures defined by RDF Schema.

2.4.1 Classes and Properties

How do we describe a particular domain? Let us consider our domain of apartment rentals. First we have to specify the "things" we want to talk about. Here we make a first, fundamental distinction. On one hand, we want to talk about particular apartments, such as the Baron Way Apartment, and particular locations, such as Amsterdam; we have already done so in RDF.

But we also want to talk about apartments, buildings, countries, cities, and so on. What is the difference? In the first case we talk about *individual objects* (resources), and in the second we talk about *classes* that define types of objects.

A class can be thought of as a set of elements. Individual objects that belong to a class are referred to as *instances* of that class. RDF provides us a way to define the relationship between instances and classes using a special property rdf:type.

An important use of classes is to impose restrictions on what can be stated in an RDF document using the schema. In programming languages, *typing* is used to prevent nonsense from being written (such as $A + 1$, where A is an array; we lay down that the