Composition & Aggregation

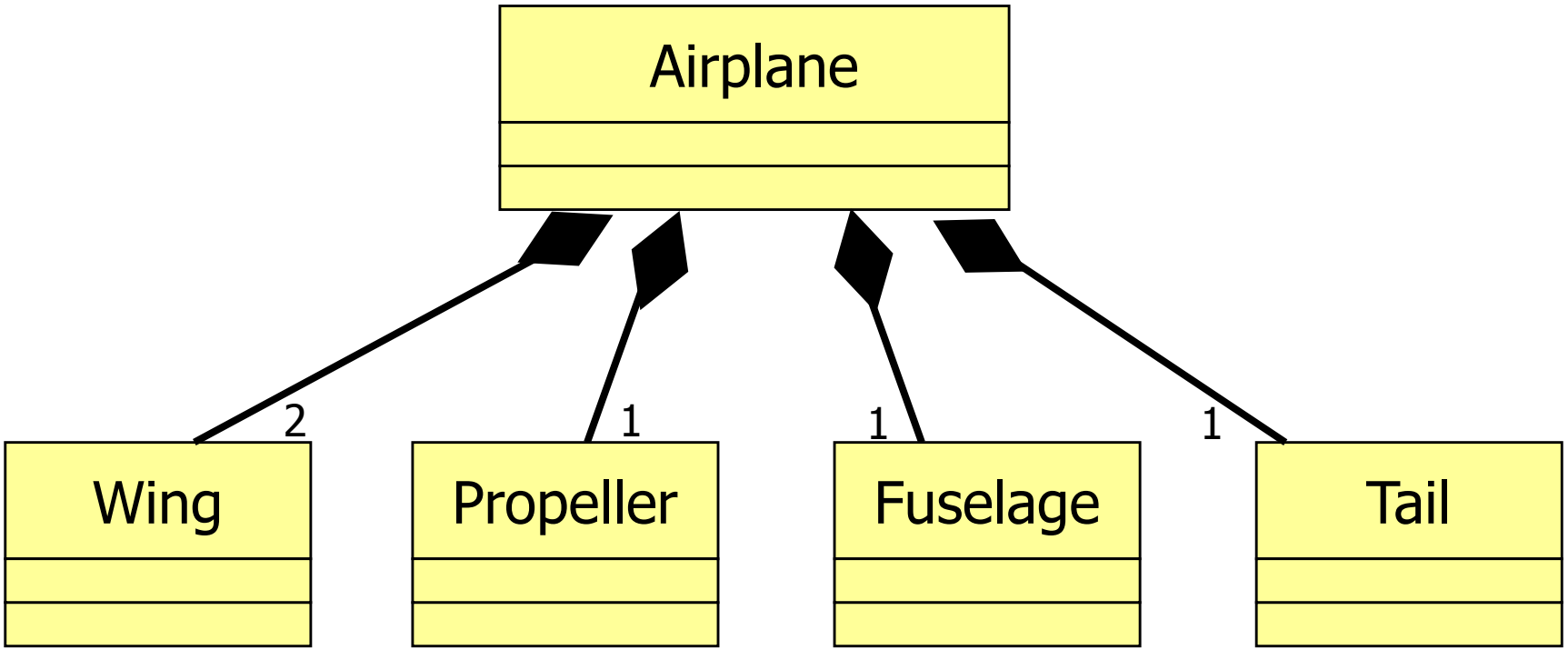
Composition

- Composition is a relationship between two classes where one class is a part of another (an ‘is a part of’ relationship)
  - If a class A is composed of classes B, C, and D, we may say that A has components B, C, and D
  - Compositions typically consist of a name, as well as the multiplicity of the component

Composition

- If a class ‘Car’ has a composition relationship with another class ‘Wheel’, we should agree that normally the multiplicity of Wheel is 4 (exactly)
- ‘Vehicle’, on the other hand, could have any number of wheels (0..*)
- Composition is a special form of association
  - It is important enough to have its own notation

Composition



Aggregation

- Aggregations is also a type of association, but again, a special one that is given its own notation
  - This is because, like composition, it is very common
- An aggregation is another relationship between classes which says one class ‘is a part of’ another class

Aggregation vs. Composition

- Both aggregation and composition represent the ‘is a part of’ relationship
- The main difference is what the part represents
  - In aggregation, the part (constituent) is meaningful without the whole (aggregate)
  - In composition, the part (component) is not meaningful without the whole (container)

Aggregation vs. Composition

- A side-effect of this difference is that a component is always a part of (at most) one container
  - However, a constituent may be a member of multiple aggregates

Aggregation vs. Composition

- In UML, composition is considered a special case of aggregation where constituents belong only to a single aggregate
  - This is why some tools (such as Rational Rose) only have symbols for aggregation, and not composition

Aggregation

- To illustrate an constituent, again, let's use a few examples:
  - A Forest is an aggregate of Trees
  - A Program is an aggregate of Statements
  - A Fleet is an aggregate of Ships
  - A Deck is an aggregate of Cards

Aggregation

- Parts of an aggregate are called constituents:
  - A Tree is an constituent of Forest
  - A Statement is an constituent of Program
  - A Ship is an constituent of Fleet
  - A Card is an constituent of Deck

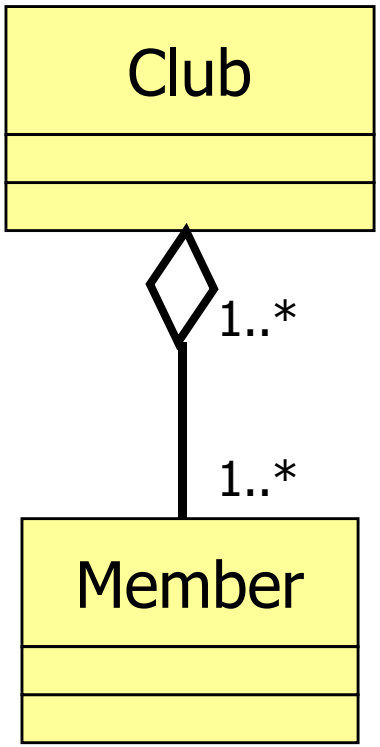
Aggregation

- These constituents share one common property:
  - The constituents of an aggregate are usually the same type
- Two other properties are also defined for aggregation:
  - An object *may* be a constituent of more than one aggregate simultaneously
  - It is *possible* to have an aggregate without any constituents

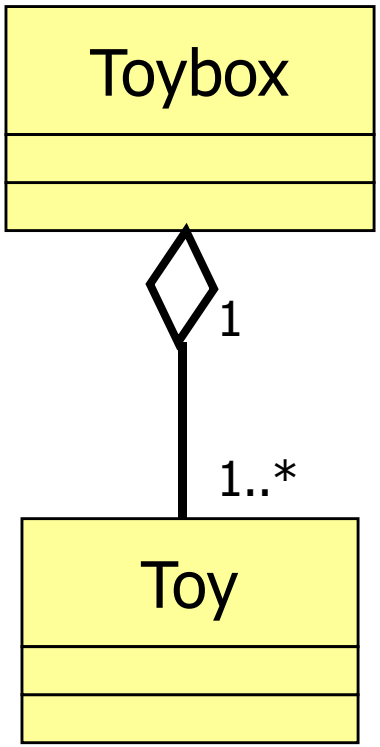
Aggregation

- An object *may* be a constituent of more than one aggregate simultaneously
  - This is identified by the multiplicity of the aggregate end of the association
- It is *possible* to have an aggregate without any constituents
  - This is identified by the multiplicity of the constituent end of the association

Aggregation



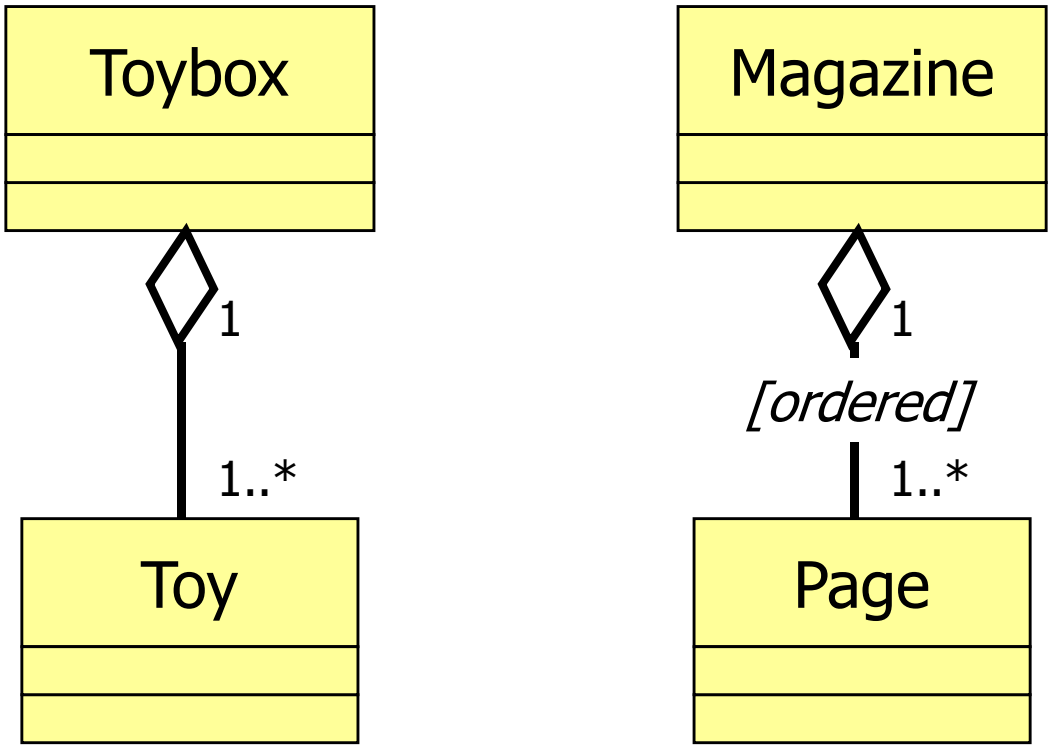
Aggregation



Aggregate Order

- Aggregates can be ordered or unordered
  - In ordered aggregates, the order of the constituents within the aggregate is important
  - In unordered aggregates, the order is unimportant
- Ordered aggregates are indicated using the symbol *[ordered]*
  - Unordered is the default for aggregates

Aggregate Order



Class Diagrams: An Example

- Say we are writing a word processor
  - A word processing document consists of a number of paragraphs
  - Each paragraph consists of a number of elements
  - An element can be a character or an image

Class Diagrams: An Example