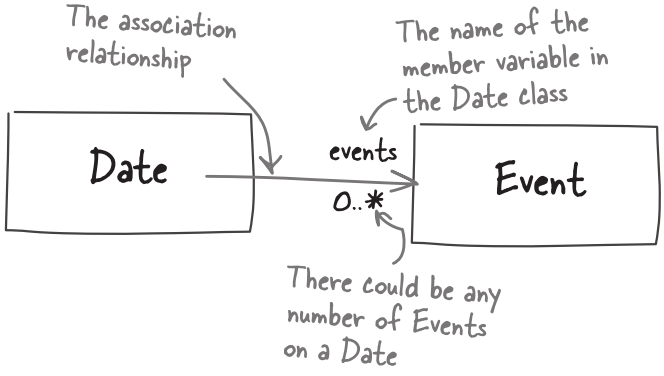
Class diagrams show relationships

Classes in your software don't exist in a vacuum, they interact with each other at runtime and have relationships to each other. In this book you've seen two relationships, called association and inheritance.

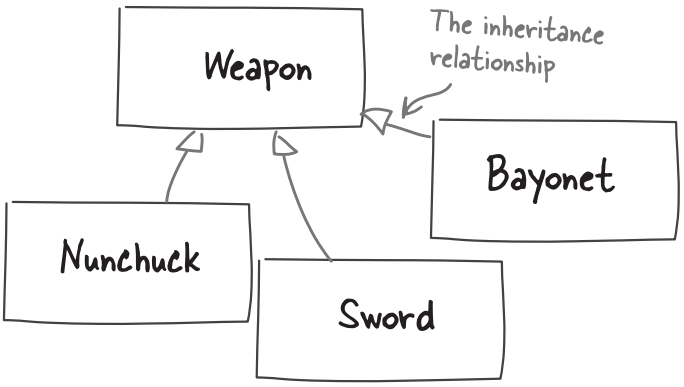
Association

Association is where one class is made up of objects of another class. For example, you might say "A Date is associated with a collection of Events."



Inheritance

Inheritance is useful when a class inherits from another class. For example, you might "A Sword inherits from Weapon."



there are no Dumb Questions

Q: Don't I need a big expensive set of tools to create UML diagrams?

A: No, not al all. The UML language was originally designed such that you could jot down a reasonably complex design with just a pencil and some paper. So if you've got access to a heavyweight UML modeling tool then that's great, but you don't actually need it to use UML.

Q: So the class diagram isn't a very complete representation of a class, is it?

A: No, but it's not meant to be. Class diagrams are just a way to communicate the basic details of a class's variables and methods. It also makes it easy to talk about code without forcing you to wade through hundreds of lines of Java, or C, or Perl.

Q: I've got my own way of drawing classes; what's wrong with that?

A: There's nothing wrong with your own notation, but it can make things harder for other people to understand. By using a standard like UML, we can all speak the same language and be sure we're talking about the same thing in our diagrams

Q: So who came up with this UML deal, anyway?

A: The UML specification was developed by Rational Software, under the leadership of Grady Booch, Ivar Jacobson, and Jim Rumbaugh (three really smart guys). These days it's managed by the OMG, the Object Management Group.

Q: Sounds like a lot of fuss over that simple little class diagram thing.

A: UML is actually a lot more than that class diagram. UML has diagrams for the state of your objects, the sequence of events in your application, and it even has a way to represent customer requirements and interactions with your system. And there's a lot more to learn about class diagrams, too.