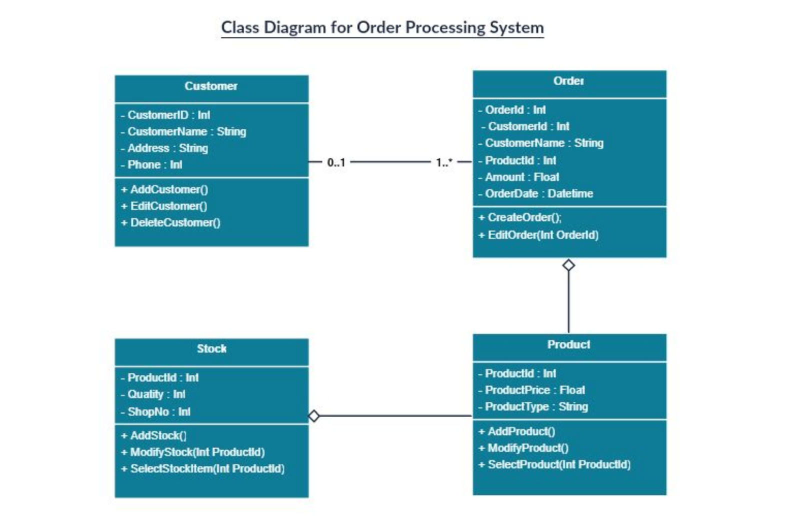
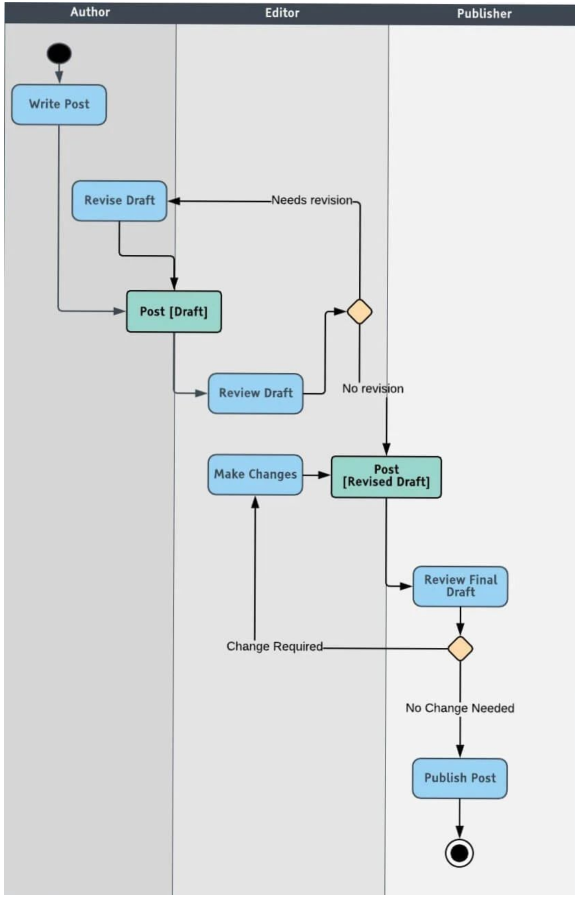
UML Diagrams

Some info on UML Diagram concepts.
Sourced from: https://www.lucidchart.com/

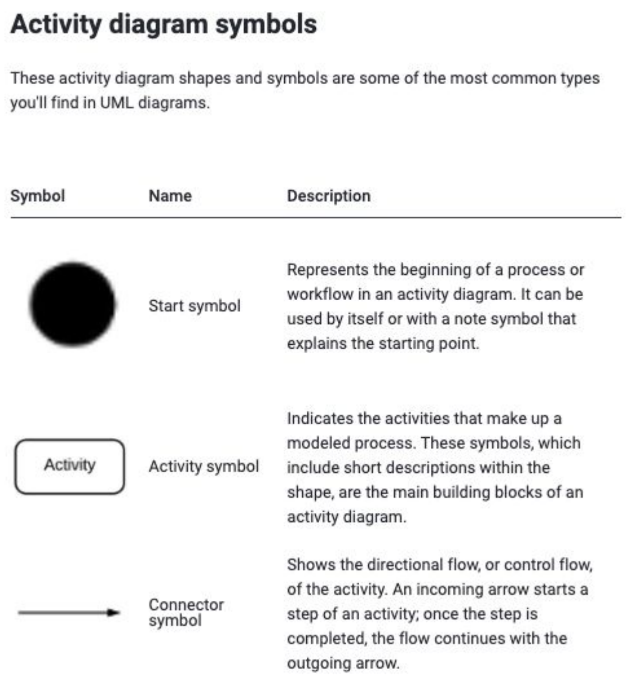


UML Activity Diagrams

Activity diagrams help people on the business and development sides of an organization come together to understand the same process and behavior.



UML Activity Diagrams - Basic Components

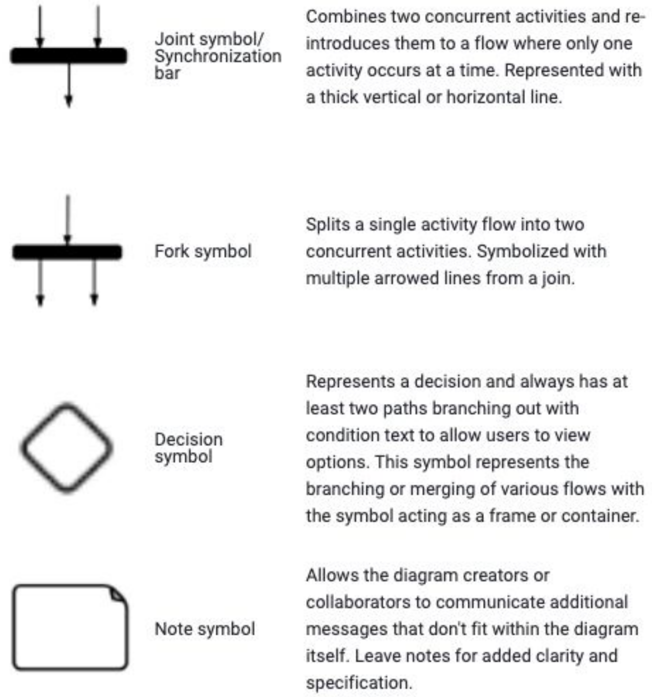


Activity diagram symbols

These activity diagram shapes and symbols are some of the most common types you'll find in UML diagrams.

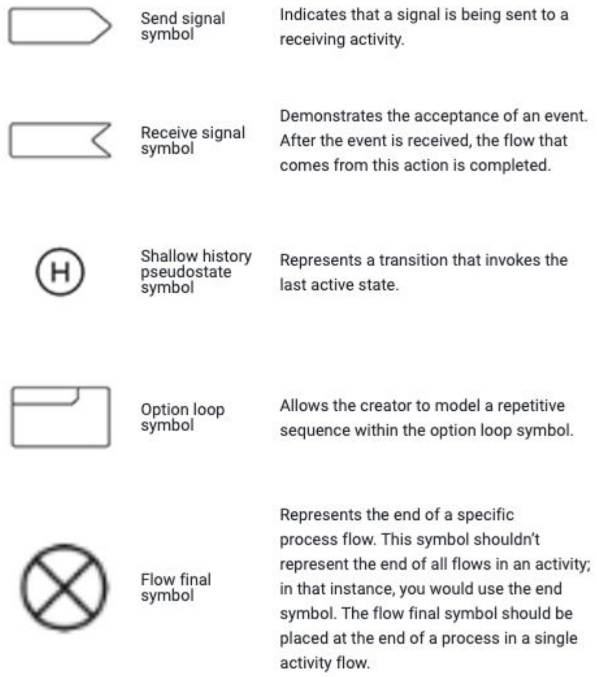
Symbol	Name	Description
●	Start symbol	Represents the beginning of a process or workflow in an activity diagram. It can be used by itself or with a note symbol that explains the starting point.
Activity	Activity symbol	Indicates the activities that make up a modeled process. These symbols, which include short descriptions within the shape, are the main building blocks of an activity diagram.
→	Connector symbol	Shows the directional flow, or control flow, of the activity. An incoming arrow starts a step of an activity; once the step is completed, the flow continues with the outgoing arrow.

UML Activity Diagrams - Basic Components



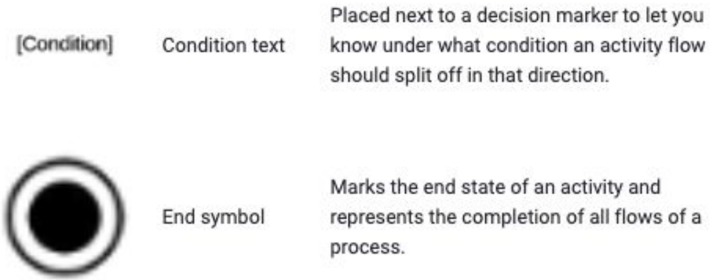
Joint symbol/ Synchronization bar	Combines two concurrent activities and re-introduces them to a flow where only one activity occurs at a time. Represented with a thick vertical or horizontal line.
Fork symbol	Splits a single activity flow into two concurrent activities. Symbolized with multiple arrowed lines from a join.
Decision symbol	Represents a decision and always has at least two paths branching out with condition text to allow users to view options. This symbol represents the branching or merging of various flows with the symbol acting as a frame or container.
Note symbol	Allows the diagram creators or collaborators to communicate additional messages that don't fit within the diagram itself. Leave notes for added clarity and specification.

UML Activity Diagrams - Basic Components



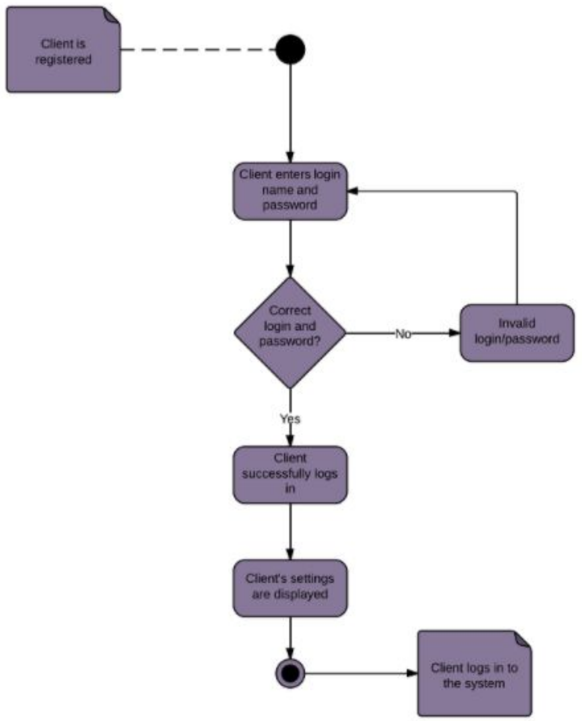
Send signal symbol	Indicates that a signal is being sent to a receiving activity.
Receive signal symbol	Demonstrates the acceptance of an event. After the event is received, the flow that comes from this action is completed.
Shallow history pseudostate symbol	Represents a transition that invokes the last active state.
Option loop symbol	Allows the creator to model a repetitive sequence within the option loop symbol.
Flow final symbol	Represents the end of a specific process flow. This symbol shouldn't represent the end of all flows in an activity; in that instance, you would use the end symbol. The flow final symbol should be placed at the end of a process in a single activity flow.

UML Activity Diagrams - Basic Components

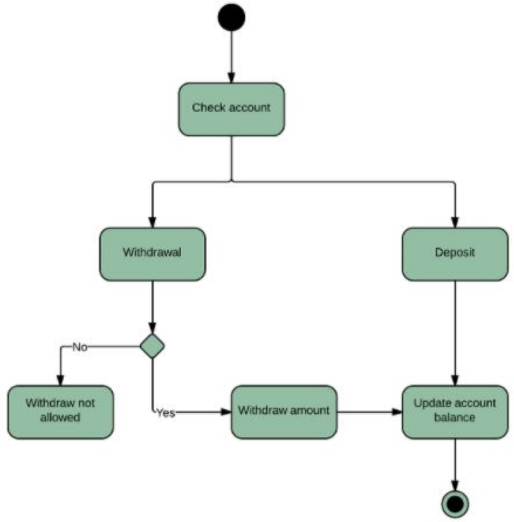


[Condition]	Condition text	Placed next to a decision marker to let you know under what condition an activity flow should split off in that direction.
◉	End symbol	Marks the end state of an activity and represents the completion of all flows of a process.

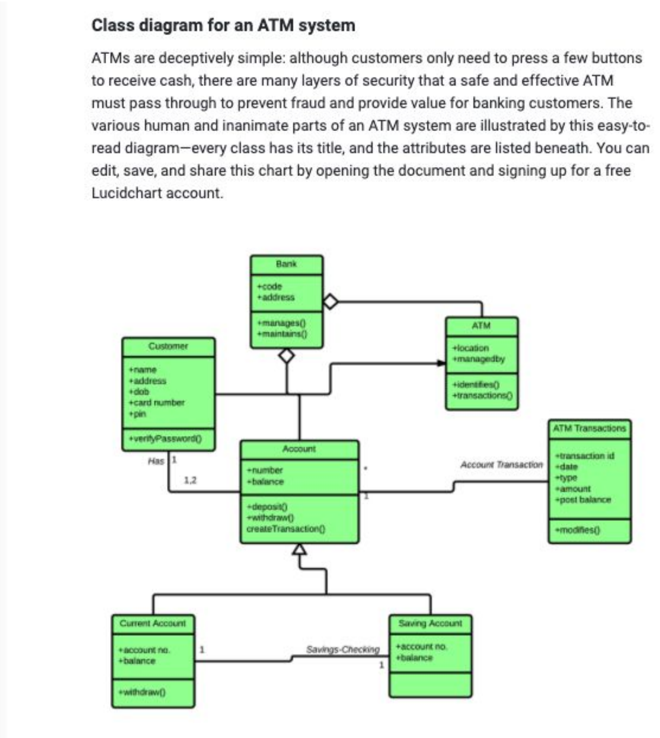
UML Activity Diagrams - Example


Client is registered
Client enters login name and password
Correct login and password?
No
Invalid login/password
Yes
Client successfully logs in
Client's settings are displayed
Client logs in to the system

UML Activity Diagrams - Example



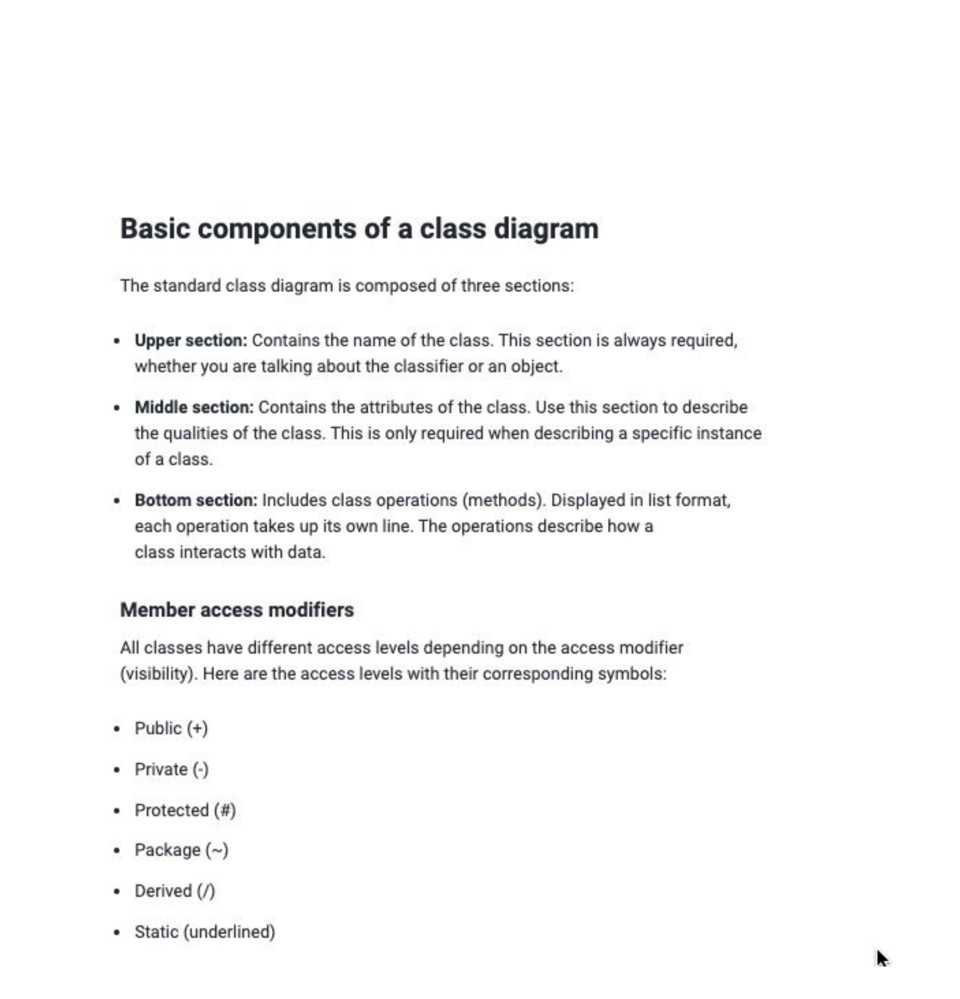
UML - Class Diagrams - Example



Class diagram for an ATM system

ATMs are deceptively simple: although customers only need to press a few buttons to receive cash, there are many layers of security that a safe and effective ATM must pass through to prevent fraud and provide value for banking customers. The various human and inanimate parts of an ATM system are illustrated by this easy-to-read diagram—every class has its title, and the attributes are listed beneath. You can edit, save, and share this chart by opening the document and signing up for a free Lucidchart account.

UML - Class Diagrams - Basic Components



Basic components of a class diagram

The standard class diagram is composed of three sections:

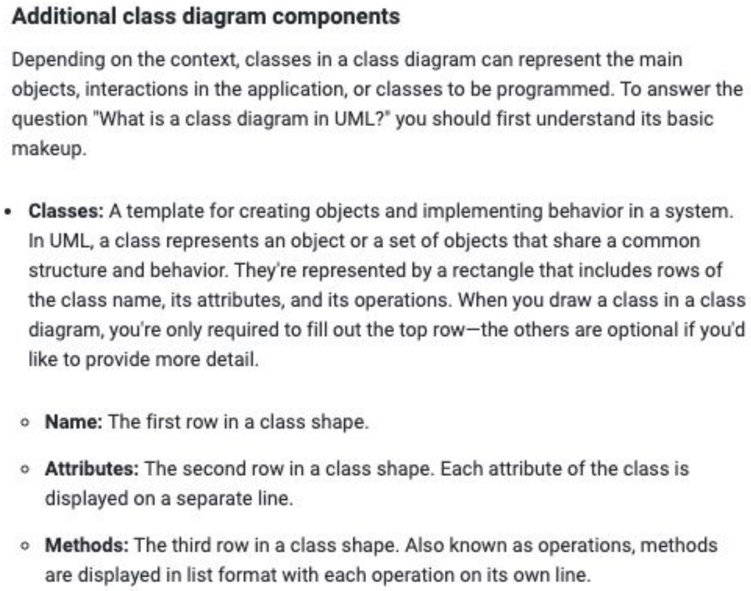
- **Upper section:** Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
- **Middle section:** Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
- **Bottom section:** Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.

Member access modifiers

All classes have different access levels depending on the access modifier (visibility). Here are the access levels with their corresponding symbols:

- Public (+)
- Private (-)
- Protected (#)
- Package (~)
- Derived (/)
- Static (underlined)

UML - Class Diagrams - Overview

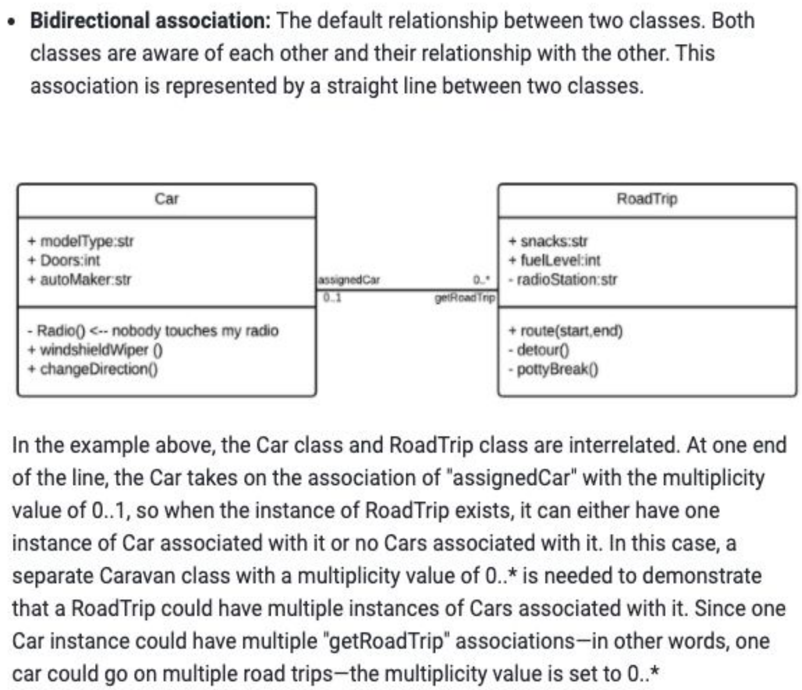


Additional class diagram components

Depending on the context, classes in a class diagram can represent the main objects, interactions in the application, or classes to be programmed. To answer the question "What is a class diagram in UML?" you should first understand its basic makeup.

- **Classes:** A template for creating objects and implementing behavior in a system. In UML, a class represents an object or a set of objects that share a common structure and behavior. They're represented by a rectangle that includes rows of the class name, its attributes, and its operations. When you draw a class in a class diagram, you're only required to fill out the top row—the others are optional if you'd like to provide more detail.
    - **Name:** The first row in a class shape.
    - **Attributes:** The second row in a class shape. Each attribute of the class is displayed on a separate line.
    - **Methods:** The third row in a class shape. Also known as operations, methods are displayed in list format with each operation on its own line.

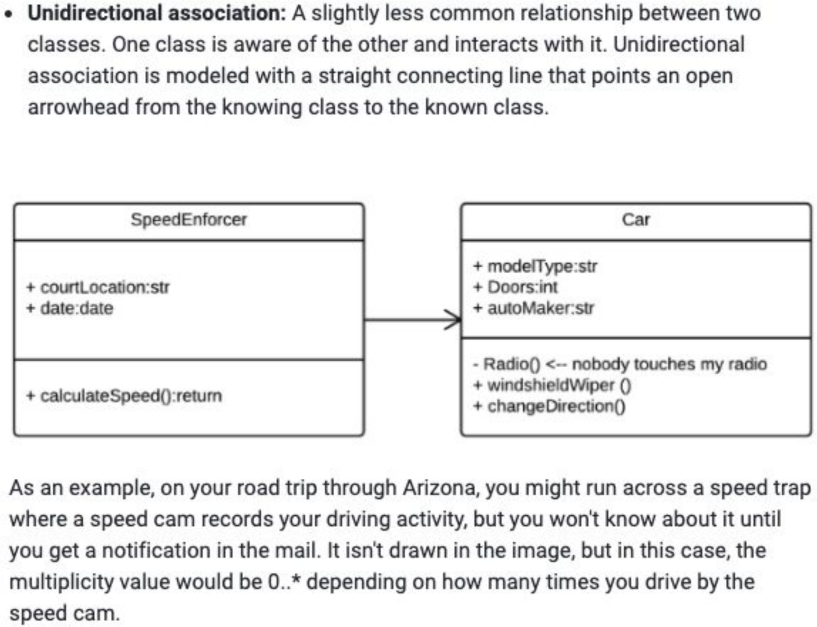
UML - Class Diagrams - Communication



- **Bidirectional association:** The default relationship between two classes. Both classes are aware of each other and their relationship with the other. This association is represented by a straight line between two classes.

In the example above, the Car class and RoadTrip class are interrelated. At one end of the line, the Car takes on the association of "assignedCar" with the multiplicity value of 0..1, so when the instance of RoadTrip exists, it can either have one instance of Car associated with it or no Cars associated with it. In this case, a separate Caravan class with a multiplicity value of 0..* is needed to demonstrate that a RoadTrip could have multiple instances of Cars associated with it. Since one Car instance could have multiple "getRoadTrip" associations—in other words, one car could go on multiple road trips—the multiplicity value is set to 0..*

UML - Class Diagrams - Communication



- **Unidirectional association:** A slightly less common relationship between two classes. One class is aware of the other and interacts with it. Unidirectional association is modeled with a straight connecting line that points an open arrowhead from the knowing class to the known class.

SpeedEnforcer

+ courtLocation:str
+ date:date

+ calculateSpeed():return

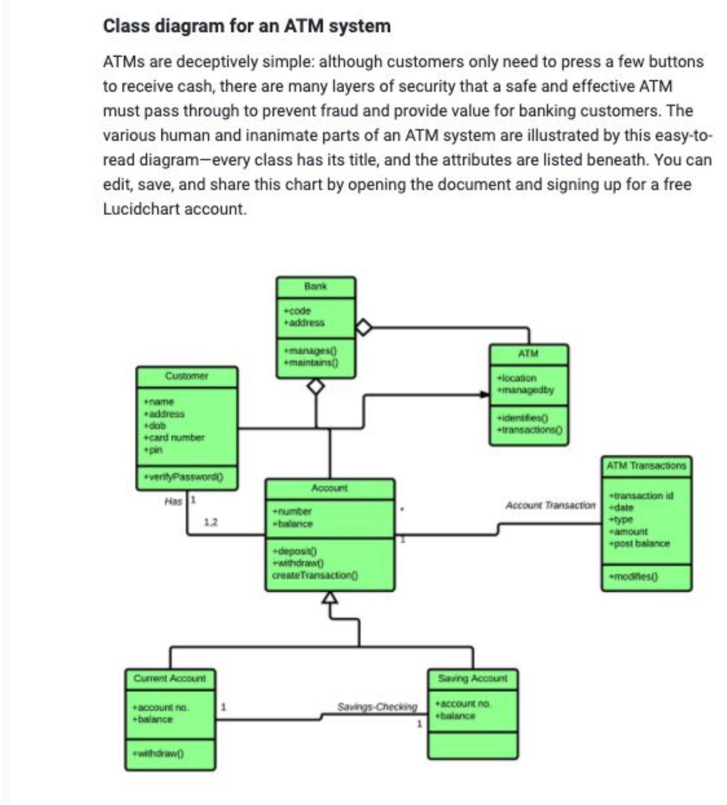
Car

+ modelType:str
+ Doors:int
+ autoMaker:str

- Radio() <-- nobody touches my radio
+ windshieldWiper ()
+ changeDirection()

As an example, on your road trip through Arizona, you might run across a speed trap where a speed cam records your driving activity, but you won't know about it until you get a notification in the mail. It isn't drawn in the image, but in this case, the multiplicity value would be 0..* depending on how many times you drive by the speed cam.

UML - Class Diagrams - Example



Class diagram for an ATM system

ATMs are deceptively simple: although customers only need to press a few buttons to receive cash, there are many layers of security that a safe and effective ATM must pass through to prevent fraud and provide value for banking customers. The various human and inanimate parts of an ATM system are illustrated by this easy-to-read diagram—every class has its title, and the attributes are listed beneath. You can edit, save, and share this chart by opening the document and signing up for a free Lucidchart account.

UML Diagrams: Useful Links

UML Activity Diagrams:

https://www.lucidchart.com/pages/uml-activity-diagram

UML Class Diagrams:

https://www.lucidchart.com/pages/uml-class-diagram

https://www.youtube.com/watch?v=UI6lqHOVHic