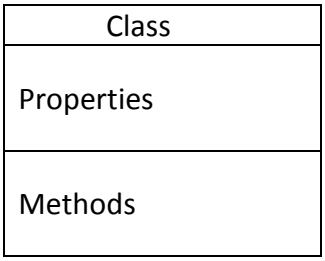
The Unified Modeling Language

When creating a diagram using the Unified Modeling Language (UML), there are two views that can be expressed. The first is the contents and components of an individual class itself, the internal view of the class. The second is the view reflecting relationships between classes, the external view of the class. Both parts are necessary for us to understand the internal structure and external relationships related to a class. Often, the two views will be shown separately, making it easier to focus on one or the other. In addition, this makes diagram creation simpler.

The Internal Class View

Here we define the components which represent the internal class view.

Symbolic Structure	**Corresponding Definition**
	A rectangular block is used to represent a class. The block contains three sections. The name of the class is listed at the top followed by the properties and methods being listed in the second and third sections. Sections are separated by a solid line.
<< ... >>	A doubled pair of less and greater than signs denotes a stereotype within the diagram, classifying elements of the diagram in a commented fashion.
+	A plus sign is used to denote that the property or method it is connected to is **public**.
–	A minus sign is used to denote that the property or method it is connected to is **private**.
#	A number sign is used to denote that the property or method it is connected to is **protected**.
Underlined Text	Underlined text denotes that the property or method underlined is **static**.
Italicized Text	Italicized text denotes that the class or method underlined is ***abstract***.
variableName: VariableType	Variables are declared by listing the variable name followed by a colon and then the variable type.
method(ParamList): ReturnType	Methods are declared by listing the method name followed by parenthesis containing a parameter list, a colon, and the return type of the method.
ParamList	A parameter list is a series of variable declarations within a comma separated list (varName1: VarType1, …, varNameN: VarTypeN).