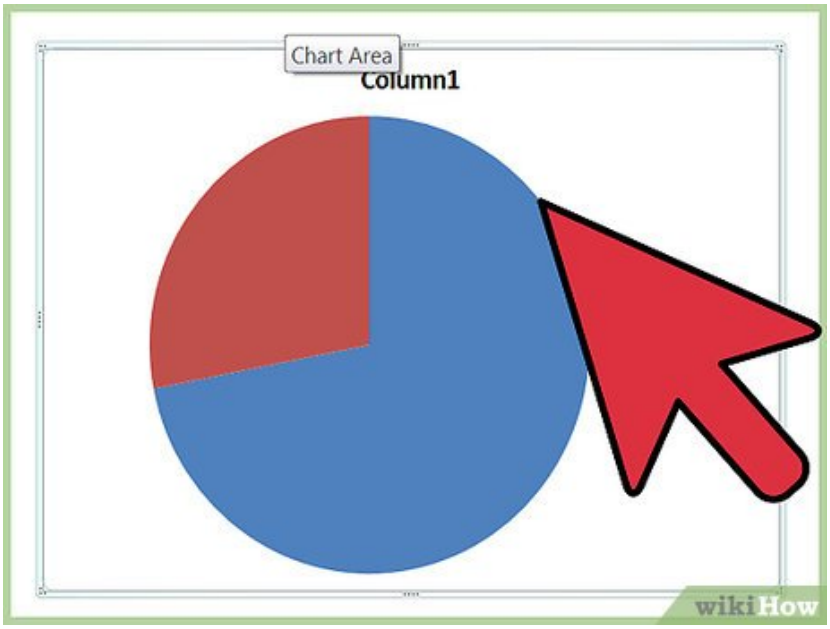
How to Design and Implement a Field Using C Sharp

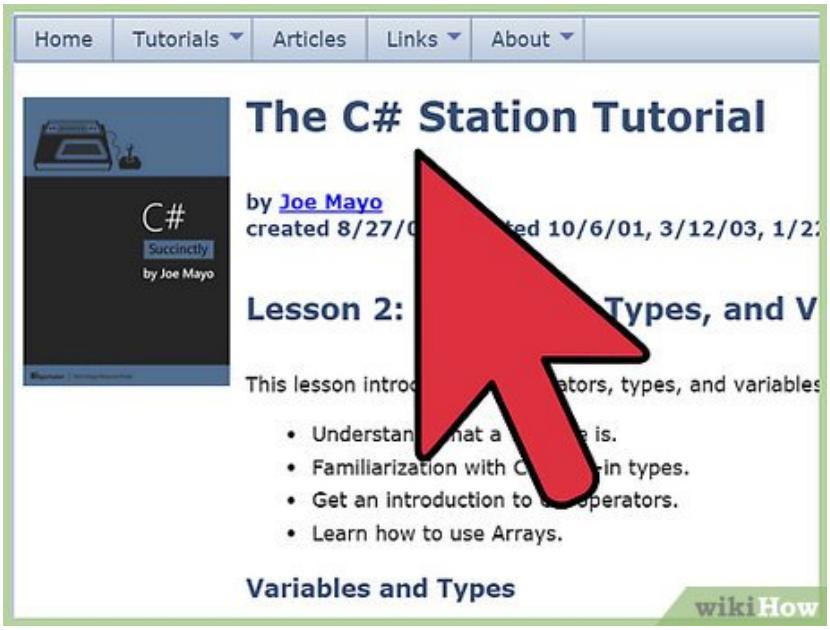
The first step in designing a class usually is to define its fields. Fields are the backbone that classes rely upon, and so, the process of designing and implementing fields is crucial to the overall process of designing classes. This...

Method 1 of 2:

Designing Fields of a Class

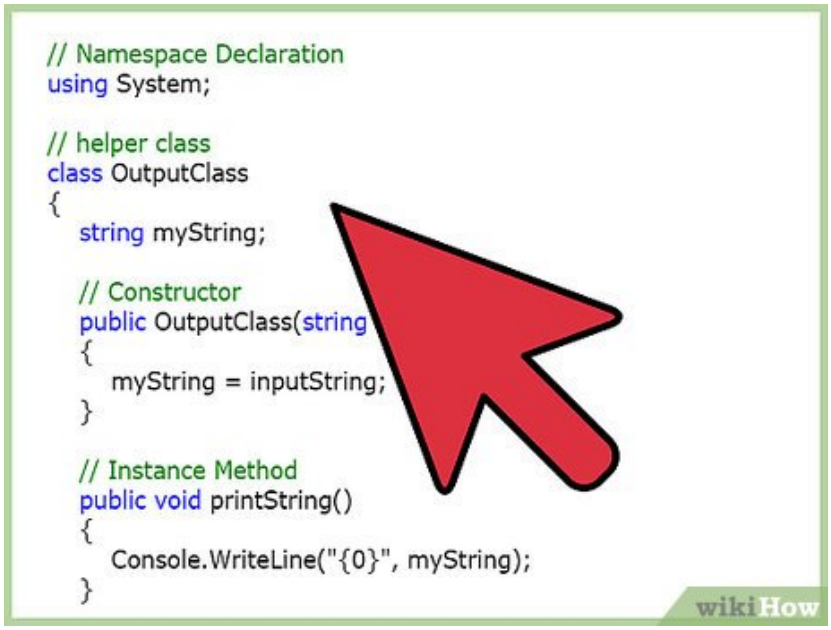
1. 

Separate the data requirements of your class in some kind of a diagram, paper, software or any form you find appropriate and comfortable to you. Prior knowledge of UML or ORM will be very useful in the design phase.

2. 

Decide whether each data item of the class is actual, calculated, or imaginary. Only actual data items are implemented as fields. The following information will help you decide the type of data item:

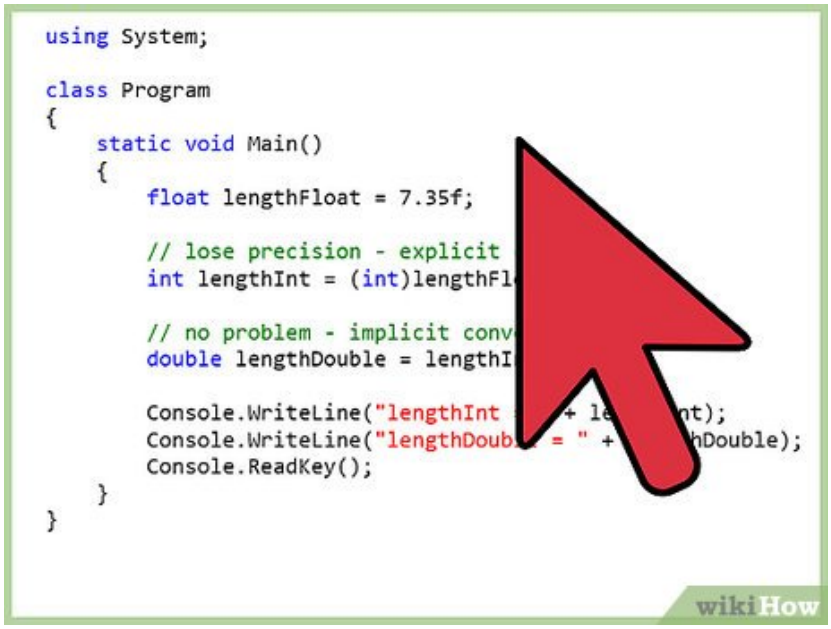
1. *Actual data items* are those items that can not be inferred from other fields and that are crucial for the class to function. For example, a Circle class will not be able to do its job without having information about its center and its radius. These two are actual data items.
2. *Calculated data items* are data items that can be calculated from other fields and need not be stored separately. For example, the area of a Circle class object can be calculated from the radius of that specific Circle object and need not be stored. Another less obvious example is the count of items in an array. This can be calculated by iterating through the array adding one to a counter (although usually, the count is stored for performance enhancement reasons).
3. *Imaginary data items* are data items that are not actually stored within the class objects. For example, the contents of a file in a "File" object. Actually, the contents of the file are not stored within the object, but rather, in the file. Whenever they are needed, they are retrieved from the file. Most imaginary data items are actually special types of calculated data items. This type of data items are usually implemented as a property, unless there are some performance or concurrency issues that makes it necessary to keep a copy of the data within the object itself. Another confusing example is when you have an object that has a reference to another object and one of the second object's data items is considered to be a data item in the first. In this case, the data is stored in the second object and so is imaginary in the first.

3. 

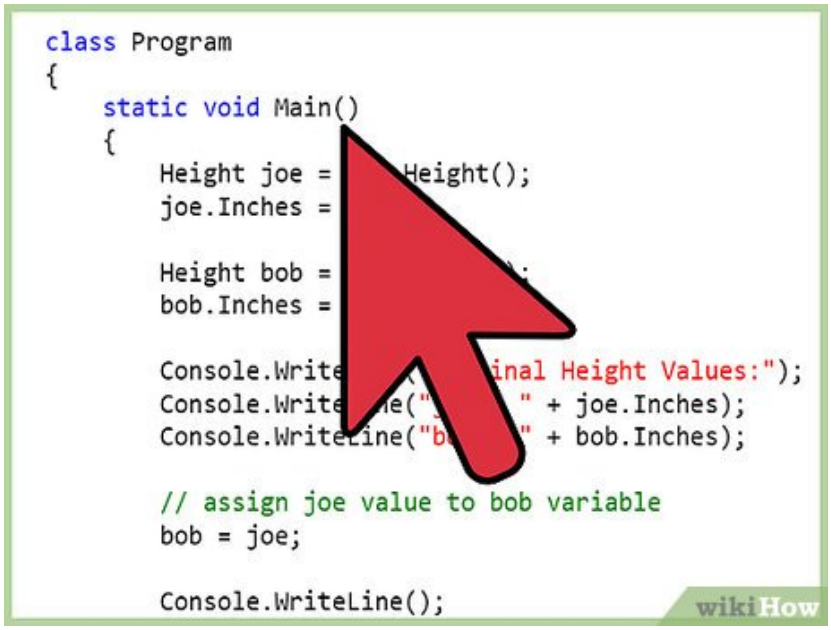
Recognize the data type of the data items that will be implemented as fields. Although this step is easy in many cases, it can be confusing in certain occasions. For example, let's suppose you are designing a "Contact" class that will store the name of the contact as a field. At first, string seems to be the right type for the name field. However, a thorough analysis might reveal that we need to store the first, middle and last name separately. This gives rise to the creation of another data type called "PersonName" to store the name (or, you might use three string fields instead and create a calculated field (property) "Name").

Method 2 of 2:

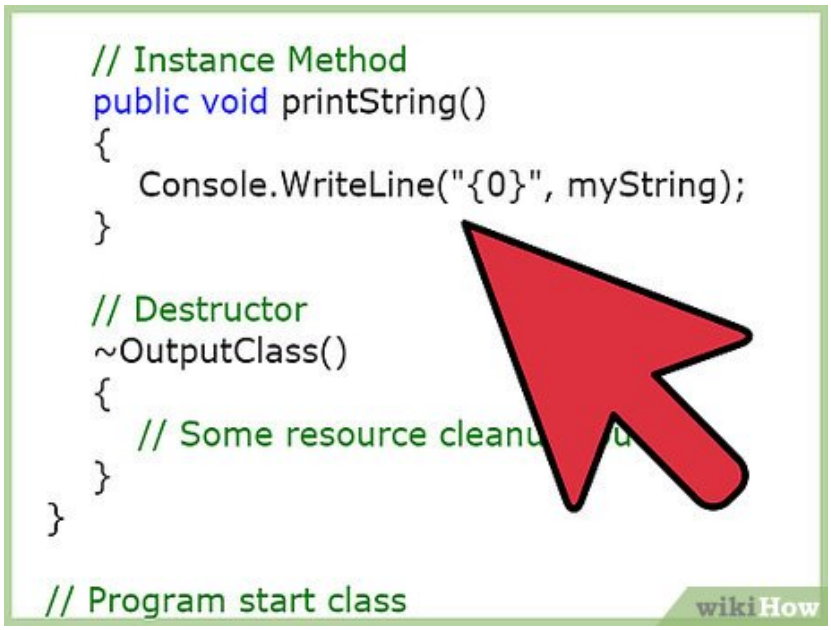
Implementing a Field

1. 

Decide the accessibility of your field (one of: public, protected, protected internal, internal, private).

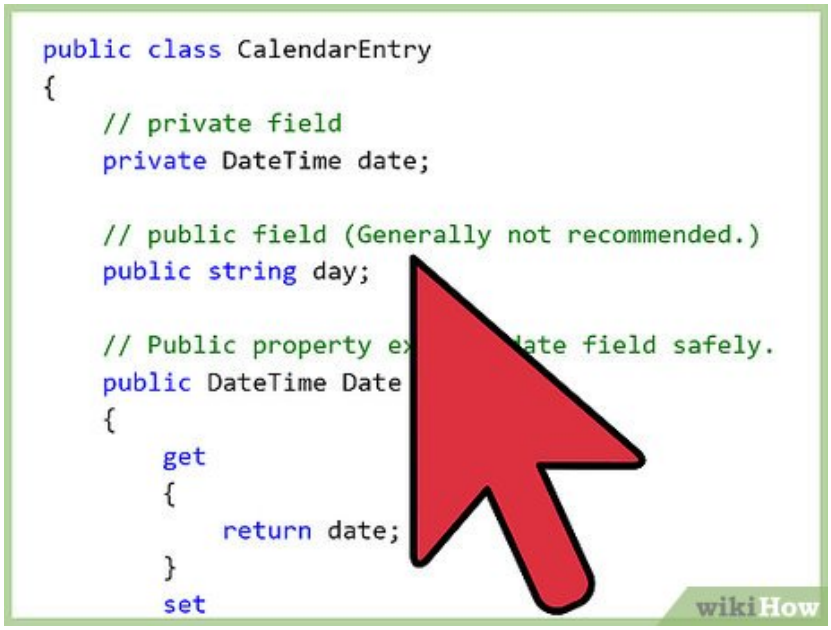
2. 

Decide the scope of your field (either static or instance scope).

3. 

Decide the changeability of your field (one of: const, readonly, or neither). The following points should help you decide:

1. Most fields are normal. That is, they are neither const nor read-only.
2. If your field is intended to hold a simple constant value and that value is a ValueType child that can be initialized immediately or a string value, you should declare the field as const. Constant (const) fields can be initialized only in the declaration of the field.
3. If your field is intended to hold a constant that is from a reference type (such as a Button, Student or any other class that is not a struct or enum), declare it as readonly. Read only fields can be initialized either in the declaration or within any constructor of the class and not anywhere else.

4. 

Declare the field using the following syntax: *accessibility* [*scope*] [*changeability*] *data-type* field-name [= initial-value];

You finished reading the article "**How to Design and Implement a Field Using C Sharp**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.