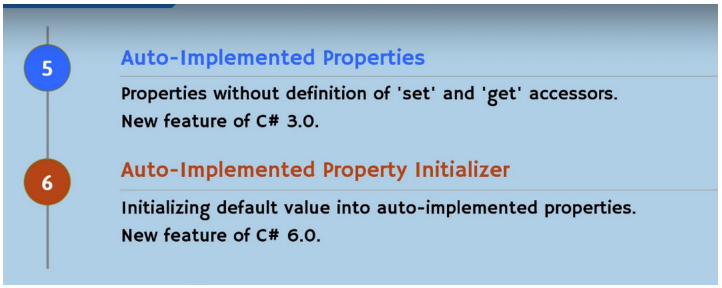
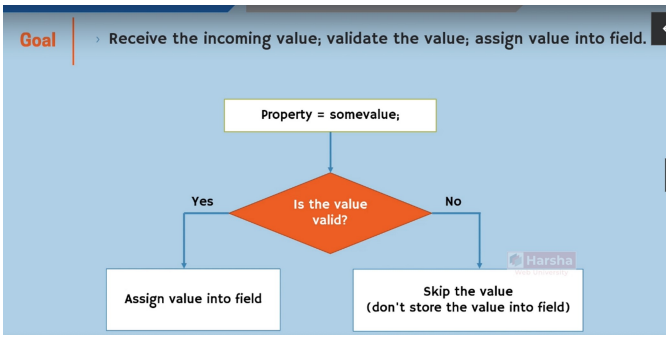
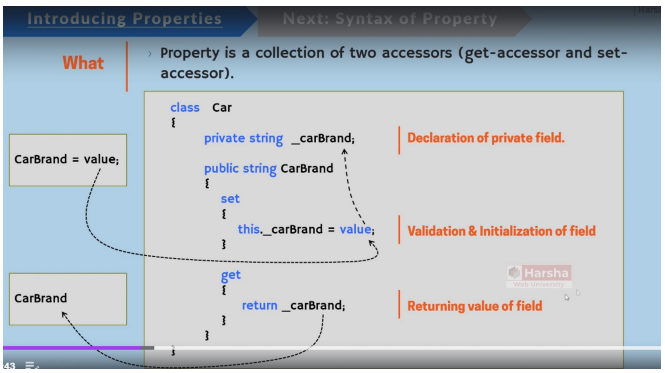
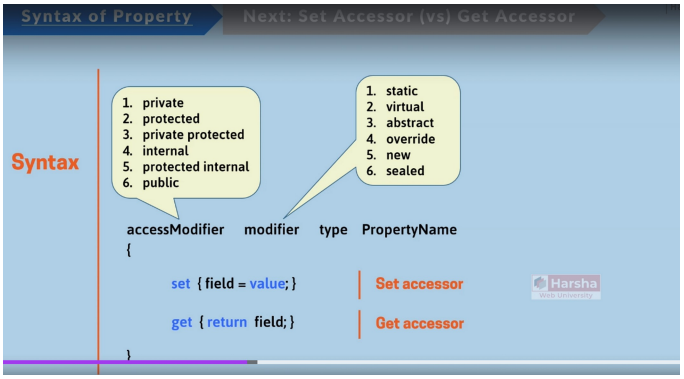
Section 9 : Properties & Indexers

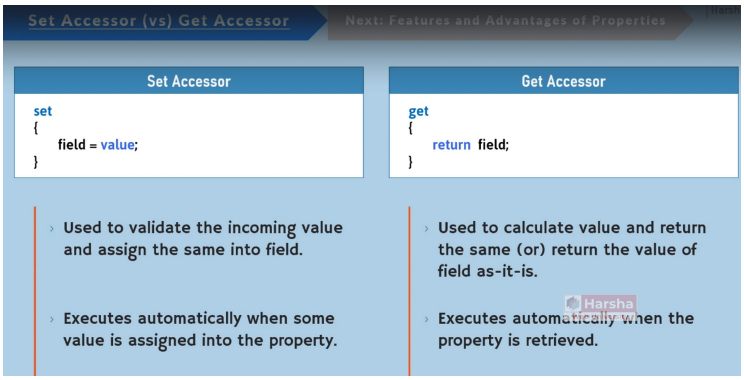
10 July 2024 21:44

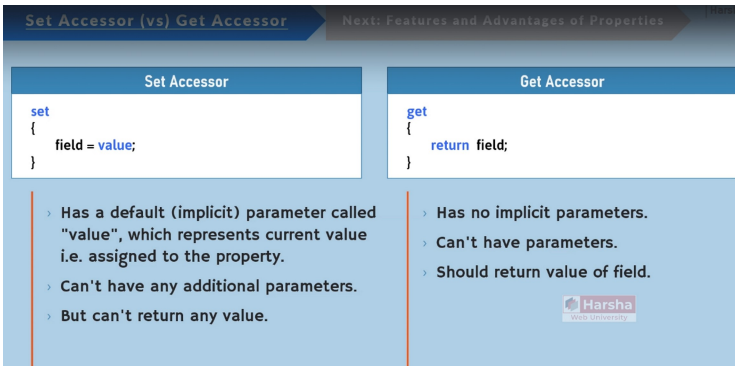




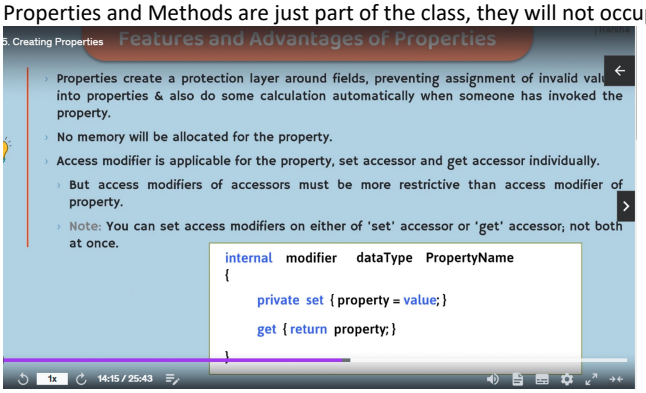




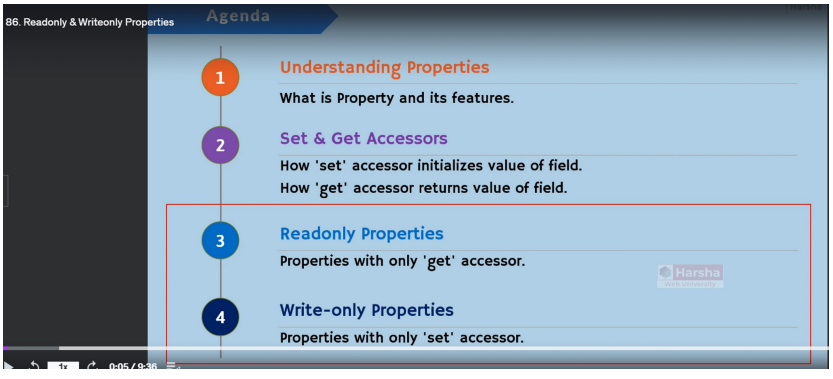




Properties and Methods are just part of the class, they will not occupy any memory.

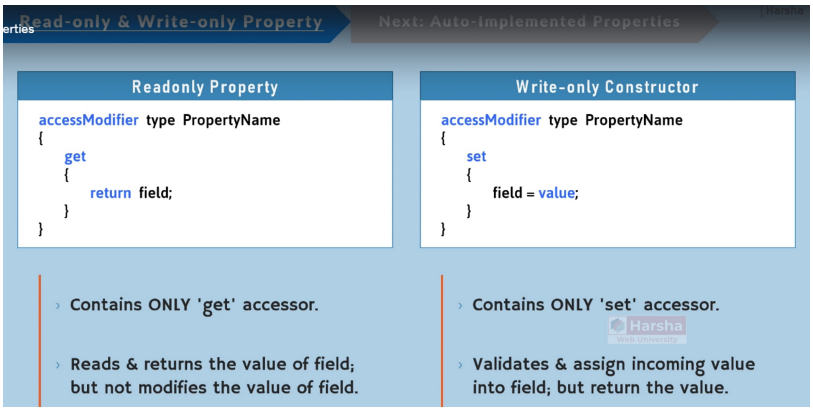


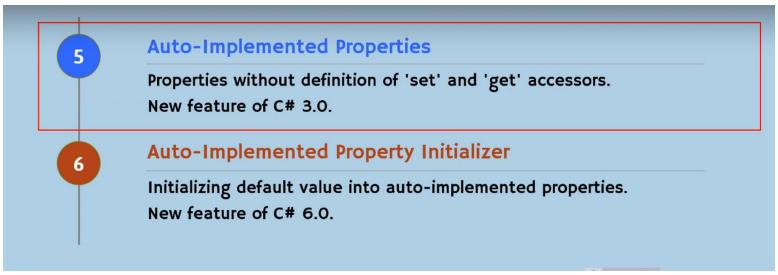
The Benefit of Property is that, we can add validation logic before assigning the value into the corresponding field.

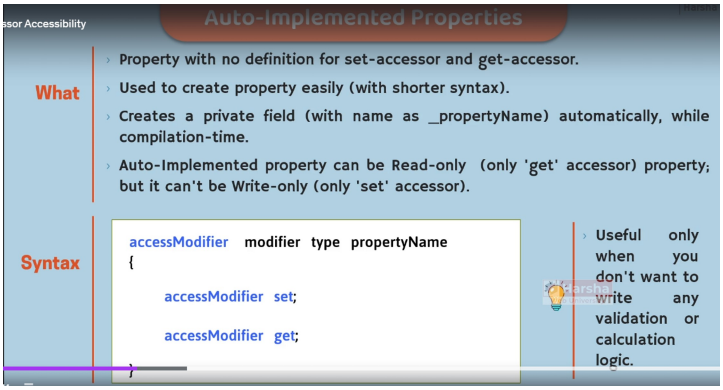


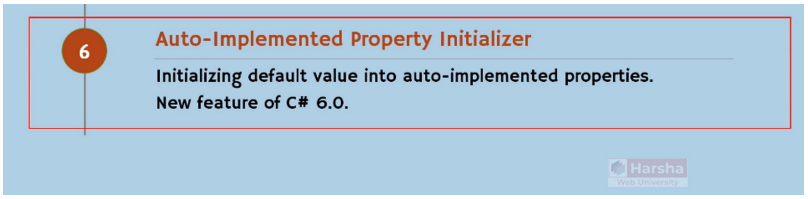
Use Readonly Properties when you want to allow the user to read the value of the field; but don't want to assign the value into the field.

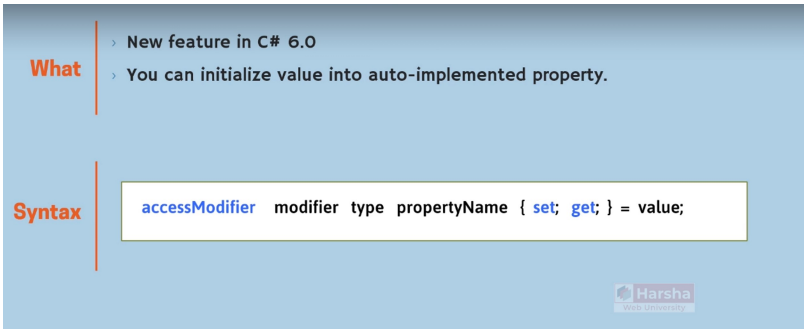
For example, in the Employee class, the salary of the employee can be read any no of times. But you don't want to allow of the user to modify the value of the Salary property of the Employee Class.

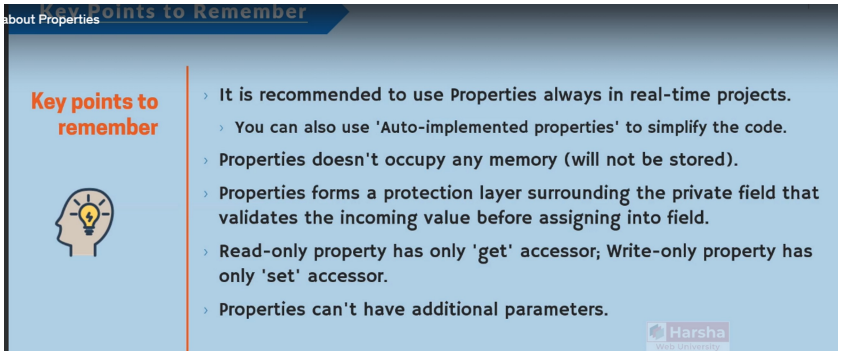




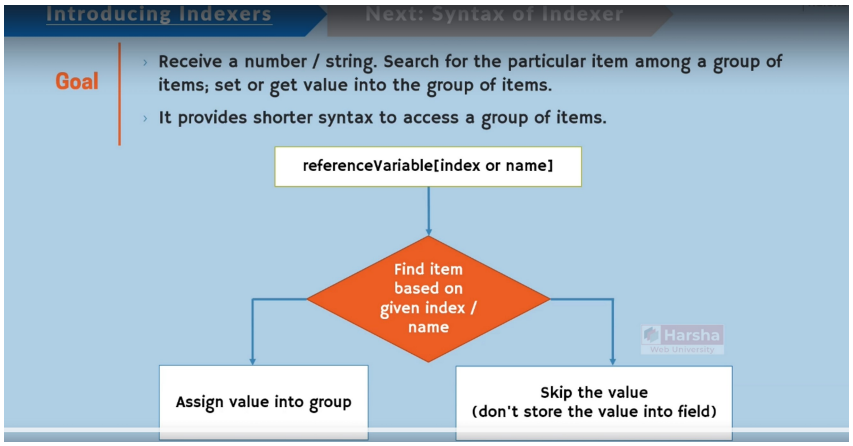


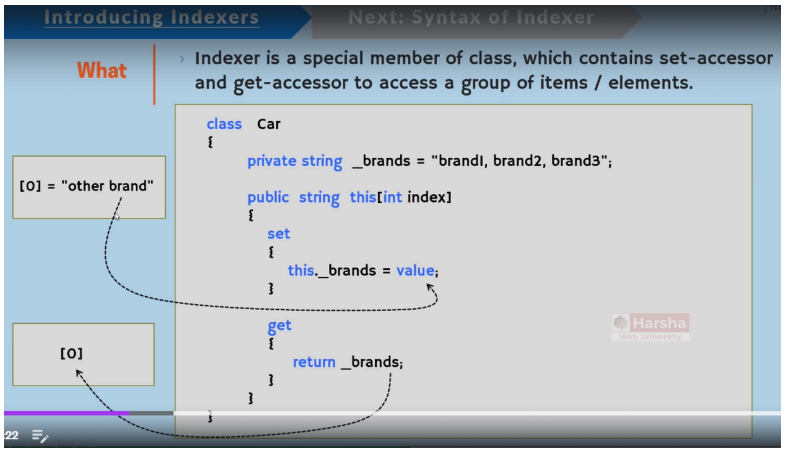


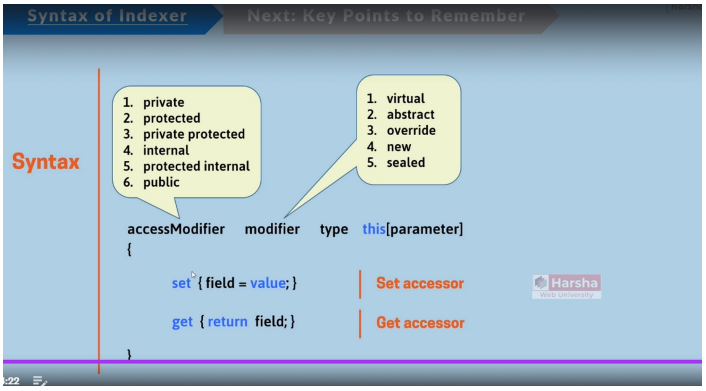


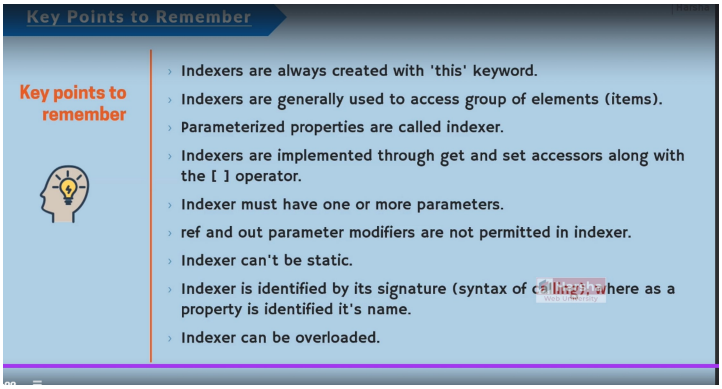


Rarely used in C#






Syntax of Indexer
Next: Key Points to Remember
Syntax
1. private
2. protected
3. private protected
4. internal
5. protected internal
6. public
1. virtual
2. abstract
3. override
4. new
5. sealed
accessModifier modifier type this[parameter]
{
set { field = value; }
Set accessor
get { return field; }
Get accessor
}


Key Points to Remember
Key points to remember
Indexers are always created with 'this' keyword.
Indexers are generally used to access group of elements (items).
Parameterized properties are called indexer.
Indexers are implemented through get and set accessors along with the [ ] operator.
Indexer must have one or more parameters.
ref and out parameter modifiers are not permitted in indexer.
Indexer can't be static.
Indexer is identified by its signature (syntax of calling) where as a property is identified it's name.
Indexer can be overloaded.