### Submission: See course shell for how and when to submit

You will practice working with properties and using List.

# Creating a Pet class

Create the following class

|  |
| --- |
| **Pet**  Class |
| **Properties** |
| + «property setter absent» Name : **string**  + «property setter private» Owner : **string**  + «property setter absent» Age : **int**  + «property setter absent» Description : **string**  + «property setter private» IsHouseTrained : **bool** |
| **Methods** |
| + «constructor» Pet(  name : **string**,  age : **int**,  description : **string**)  + ToString() : **string**  + Train() : **void**  + SetOwner(newOwner : **string**) : **void** |

## Description of members:

##### Fields:

There are no fields.

##### Properties:

1. The properties are self-explanatory. The getter is public and the setter is mostly absent.

##### Constructor:

1. **public Pet(string name, int age, string description) –** This constructor takes three arguments and assigns them to the appropriate properties. It also initializes the fields owner to “no one” and **isHousedTrained** to **false**

Remember the ToString() method is needed to produce a sensible output on the screen

##### Methods:

1. **public override string ToString()** – This method returns a string fully describing this object.
2. **public void SetOwner(string owner)** – This method simply assigns the argument to the appropriate field.
3. **public void Train()** – This method sets the property IsHouseTrained to **true**.

## Test Harness

In your main method write the code to do the following:

1. Create four objects. You decide on the arguments
2. Create a List to store all the above objects.
3. Use some of the methods on some of the objects.
4. Using a suitable looping statement, display all the objects in the collection.
5. Prompt the user for an owner’s name and then display only the pets belonging to a particular person.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PetDemo2

{

class Program

{

static void Main(string[] args)

{

Pet pet = new Pet("Garfield", 5, "lazy cat");

Console.WriteLine(pet);

pet.SetOwner("Jon");

Console.WriteLine(pet);

pet.Train();

Console.WriteLine(pet);

Pet pet2 = new Pet("Beethoven", 2, "dog");

pet2.SetOwner("Jon");

Pet pet3 = new Pet("Ole Yeller", 2, "collie");

pet3.Train();

Pet pet4 = new Pet("Marmaduque", 4, "great dane");

//create a list of pet objects.

List<Pet> pets = new List<Pet>() { pet, pet2, pet3, pet4 };

Console.WriteLine("\nAll the pets:");

foreach (Pet p in pets)

{

Console.WriteLine(p);

}

string owner = "Jon";

Console.WriteLine($"All the pets owned by {owner}");

foreach (Pet p in pets)

{

if (p.Owner==owner)

{

Console.WriteLine(p);

}

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pet

{

class Pet

{

// FIELDS (INSTANCE VARIABLES)

private string name;

private string owner;

private int age;

private string description;

private bool isHouseTrained;

// CONSTRUCTOR

public Pet(string name, int age, string description)

{

this.name = name;

this.age = age;

this.description = description;

this.owner = "No one";

this.isHouseTrained = false;

}

// ToString Method

public override string ToString()

{

return string.Format("\n[Pet Information]" +

"\n- Name: " + name +

"\n- Owner: " + owner +

"\n- Age: " + age +

"\n- Description: " + description +

"\n- House Trained: " + isHouseTrained);

}

// PUBLIC Method

// This Method is needed for modifying owner

public void modifyOwner(string ownerName)

{

this.owner = ownerName;

}

// This Method is needed for picking particular owner up.

public string choiceOwner()

{

return owner;

}

}

}