Projekt TO
PetShop
Ślusarz Dominik

Wzorce:

Singleton- tworzony jest tylko jeden magazyn(jedna instancja klasy)

Multiton- uniemożliwia dodanie tego samego produktu(zwierzęcia) do klienta albo magazynu

Prototyp- dzięki niemu możemy klonować produkty, C# działa na referencjach, klonowanie jest niezbędne

Budowniczy- tworzenie obiektów klasy przez zarządce nie bezpośrednio przez konstruktor

Metoda wytwórcza- tworzenie obiektów klasy przez fabrykę

Dekorator- odpowiednie wyświetlanie każdego produktu(psa, kota)

Stan- klient może mieć aktywne konto i możliwość działania w systemie oraz nieaktywne

MVC- podział projektu na model widok kontroler

Obserwator- część MVC, model powiadamia widok/widoki o zmianie stanu

Strategia- część MVC, kontroler przyjmuje odpowiednią strategie, może mapować te same wejścia na inny sposób w zależności od wybranej strategii

Mediator- każdy klient przez odpowiednią klasę może wykonywać logi(poprzez zakup) w ten sam sposób

Fasada- przez nadanie odpowiedniej strategii kontrolerowi klient posiada dostęp tylko do części funkcjonalności systemu/podsystemów

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-02
 * Time: 13:47
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.Builder.Builders;
using PetShop.M.Classes.Product;


    /*
     * director to create dog : builder pattern
     */
    class DogBreeder{
        private DogBuilder dogBuilder;
        /*
         * getter and setter for dog builder
         */
        public DogBuilder DogBuilder{
            get { return dogBuilder; }
            set { dogBuilder = value; }
        }
        /*
         * getter for dog
         */
        public Dog Dog {
            get { return dogBuilder.Dog; }
        }
        /*
         * method to create dog by part
         */
        public void ConstructDog(int number, double price){
            dogBuilder.CreateNewDog();
            dogBuilder.BuildNumber(number);
            dogBuilder.BuildPrice(price);
            dogBuilder.BuildSpecies();
            dogBuilder.BuildRace();
        }
    }

    /*
     * director to create cat : builder pattern
     */
    class CatBreeder{
        private CatBuilder catBuilder;
        /*
         * getter and setter for cat builder
         */
        public CatBuilder CatBuilder{
```

```
            get { return catBuilder; }
            set { catBuilder = value; }
        }
        /*
         * getter for cat
         */
        public Cat Cat {
            get { return catBuilder.Cat; }
        }
        /*
         * method to create cat by part
         */
        public void ConstructCat(int number, double price){
            catBuilder.CreateNewCat();
            catBuilder.BuildNumber(number);
            catBuilder.BuildPrice(price);
            catBuilder.BuildSpecies();
            catBuilder.BuildRace();
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 12/21/2014
 * Time: 21:14
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using PetShop.M.Classes.Product;

namespace PetShop.Builder.Builders{
    /*
     * abstract Dog builder, base for creating dog : builder pattern
     */
    abstract class DogBuilder{
        protected Dog dog;

        public Dog Dog{
            get{ return dog; }
        }
        public void CreateNewDog() {
            dog = new Dog();
        }
        public abstract void BuildNumber(int number);
        public abstract void BuildPrice(double price);
        public abstract void BuildSpecies();
        public abstract void BuildRace();
    }

    /*
     * dog builders : builder pattern
     */
    class DogDogBuilder : DogBuilder{
        public override void BuildRace() {
            dog.Race = "Dog";
        }

        public override void BuildNumber(int number) {
            dog.Number = number;
        }

        public override void BuildPrice(double price) {
            dog.Price = price;
        }

        public override void BuildSpecies() {
            dog.Species = "Pet";
        }
    }

}
/*
```

```
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-02
 * Time: 13:46
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M.Classes.Product;

namespace PetShop.Builder.Builders{

    /*
     * abstract Cat builder, base for creating cat : builder pattern
     */
    abstract class CatBuilder{
        protected Cat cat;

        public Cat Cat{
            get{ return cat; }
        }
        public void CreateNewCat() {
            cat = new Cat();
        }
        public abstract void BuildNumber(int number);
        public abstract void BuildPrice(double price);
        public abstract void BuildSpecies();
        public abstract void BuildRace();
    }

    /*
     * cat builders : builder pattern
     */
    class CatCatBuilder : CatBuilder{
        public override void BuildRace() {
            cat.Race = "Cat";
        }

        public override void BuildNumber(int number) {
            cat.Number = number;
        }

        public override void BuildPrice(double price) {
            cat.Price = price;
        }

        public override void BuildSpecies() {
            cat.Species = "Pet";
        }
    }

}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 19:13
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M.Classes.Product;

namespace PetShop.Factory{
   public class Chicken : Farm{
      protected string race;

      public string Race{
         get {return race;}
         set {race = value;}
      }

      public Chicken(int number, double price, string species, string race){
         this.number = number;
         this.price = price;
         this.species = species;
         this.race = race;
      }

      public override string Name(){
         return "Chicken";
      }
   }
}
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 12/21/2014
 * Time: 21:16
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M.Classes.Product;

namespace PetShop.Factory{

   public class Cow : Farm{
      protected string race;

      public string Race{
         get {return race;}
         set {race = value;}
      }
```

```
        public Cow(int number, double price, string species, string race){
            this.number = number;
            this.price = price;
            this.species = species;
            this.race = race;
        }

        public override string Name(){
            return "Cow";
        }

    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 19:06
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.V;

namespace PetShop.Observer{
    public interface IObservable{
        void Attach(View view);
        void Detach(View view);
        void Notify();
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 19:06
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M;

namespace PetShop.Observer{
    public interface IObserver{
        void Update(Model model);
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 10:57
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M;
using PetShop.C.Strategy.StrategyInterface;
using PetShop.V;
namespace PetShop.C{

    public class Controller{
        IStrategy strategy;

          Model model;
        View view;
        //stratefy pattern
        public IStrategy Strategy{
            get { return strategy; }
        }

        public Model Model{
            get { return model; }
        }

        public View View{
            get { return view; }
        }

        public Controller(Model model, View view, IStrategy strategy){
            this.model = model;
            this.view = view;
            this.strategy = strategy;
        }

        public void Start(){
            strategy.InitModelAndView(model, view);
            strategy.Start();
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-10
 * Time: 21:46
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M;
using PetShop.C.Strategy.StrategyInterface;
using PetShop.V;

namespace PetShop.C.Strategy{
   public abstract class BaseStrategy : IStrategy{
      protected Model model;
      protected View view;

      public abstract void InitModelAndView(Model model, View view);
      public abstract void Start();

   }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 22:04
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.M;
using PetShop.V;

namespace PetShop.C.Strategy.StrategyInterface{
    public interface IStrategy{
        void InitModelAndView(Model model, View view);
        void Start();
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 22:13
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.Builder.Director;
using PetShop.Builder.Builders;
using PetShop.M;
using PetShop.M.Classes.Product;
using PetShop.V;

namespace PetShop.C.Strategy.Strategies{

   public class ConsoleStrategy : BaseStrategy    {

      public override void InitModelAndView(Model model, View view){
         this.model = model;
         this.view = view;
      }

      public override void Start(){
         while(true){
            view.DisplayMainOptions();
            try{
               switch(Convert.ToInt32(view.EnterOption())){
                  case(-1):
                     break;
                  case 0: // exit
                     view.DisplayMasage("Exit");
                     //view.WaitAndClear();
                     return;
                  case 1: // add animal to werehouse
                     view.DisplayAvailableAnimals();
                     switch(view.EnterAnimal()){
                        case("Dog"): // add dog
                           view.DisplayMasage("Enter number and price");
                           try{
                              // dog builder
                              DogBreeder dogBreeder = new DogBreeder();
                              dogBreeder.DogBuilder = new DogDogBuilder();
                              dogBreeder.DogBuilder.CreateNewDog();
                              dogBreeder.ConstructDog(Convert.ToInt32(view.EnterAnimalNumber
Werehouse()), Convert.ToDouble(view.EnterPrice()));

                              model.Warehouse.AddAnimalToWarehouse(dogBreeder.Dog.Race, dogB
reeder.Dog);
                              dogBreeder = null;
                           }catch(InvalidCastException){
```

```
                        }
                        break;
                    case("Cat"): // add cat
                        view.DisplayMasage("Enter number and price");
                        try{
                            // cat builder
                            CatBreeder catBreeder = new CatBreeder();
                            catBreeder.CatBuilder = new CatCatBuilder();
                            catBreeder.CatBuilder.CreateNewCat();
                            catBreeder.ConstructCat(Convert.ToInt32(view.EnterAnimalNumberW
erehouse()), Convert.ToDouble(view.EnterPrice()));

                            model.Warehouse.AddAnimalToWarehouse(catBreeder.Cat.Race,catBree
der.Cat);
                            catBreeder = null;
                        }catch(InvalidCastException){
                            view.DisplayError("It is not the number!");
                        }
                        break;
                    case("Cow"): // add cow
                        view.DisplayMasage("Enter number and price");
                        try{
                            // factory method
                            model.Warehouse.AddAnimalToWarehouse("Cow", Farm.FarmFactory
(Animal.Animals.Cow, Convert.ToInt32(view.EnterAnimalNumberWerehouse()),  Convert.ToD
ouble(view.EnterPrice())));
                        }catch(InvalidCastException){
                            view.DisplayError("It is not the number!");
                        }
                        break;
                    case("Chicken"): // add Chicken
                        view.DisplayMasage("Enter number and price");
                        try{
                            // factory method
                            model.Warehouse.AddAnimalToWarehouse("Chicken", Farm.FarmFact
ory(Animal.Animals.Chicken, Convert.ToInt32(view.EnterAnimalNumberWerehouse()),  Conve
rt.ToDouble(view.EnterPrice())));
                        }catch(InvalidCastException){
                            view.DisplayError("It is not the number!");
                        }
                        break;
                    default:
                        view.DisplayError("Wrong choise");
                        break;
                }
                model.Notify();
                //view.WaitAndClear();
                break;
            case 2: // Display warehouse
                view.DisplayWarehouseStatus(model.Warehouse.GetValues());
                //view.WaitAndClear();
                break;
```

```
                case 3: // remove number of animals form warehouse
                    view.DisplayWarehouseStatus(model.Warehouse.GetValues());
                    try{
                        view.DisplayMasage("Enter animals to remove");
                        model.Warehouse.RemoveAnimalFromWarehouse(view.EnterAnimal(), Con
vert.ToInt32(view.EnterAnimalNumberWerehouse()));
                    }catch(InvalidCastException){
                        view.DisplayError("It is not the number!");
                    }
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                case 4: // add to basket
                    view.DisplayWarehouseStatus(model.Warehouse.GetValues());
                    try{
                        view.DisplayMasage("What animal you want to buy and how many?");
                        model.Client.AddAnimalToClient(model.Warehouse, view.EnterAnimal(), C
onvert.ToInt32(view.EnterAnimalNumberClient()));
                    }catch(InvalidCastException){
                        view.DisplayError("It is not the number!");
                    }
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                case 5: // Display client's basket
                    view.DisplayClientStatus(model.Client.GetValues(), model.Client.Sum, model.C
lient.Credit);
                    //view.WaitAndClear();
                    break;
                case 6: // remove number of animals form client's basket
                    view.DisplayClientStatus(model.Client.GetValues());
                    try{
                        view.DisplayMasage("Chose animal and enter number of animals to remove");
                        model.Client.RemoveAnimalFromClient(model.Warehouse, view.EnterAnim
al(), Convert.ToInt32(view.EnterAnimalNumberClient()));
                    }catch(InvalidCastException){
                        view.DisplayError("It is not the number!");
                    }
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                case 7: // change price
                    view.DisplayWarehouseStatus(model.Warehouse.GetValues());
                    try{
                        view.DisplayMasage("Which animal you want to change price?");
                        model.Warehouse.ChangeAnimalPrice(view.EnterAnimal(), Convert.ToInt32
(view.EnterAnimalNumberWerehouse()));
                    }catch(InvalidCastException){
                        view.DisplayError("It is not the number!");
                    }
                    model.Notify();
                    //view.WaitAndClear();
```

```
                    break;
                case 8: // change number of animal
                    view.DisplayWarehouseStatus(model.Warehouse.GetValues());
                    try{
                        view.DisplayMasage("Which animal you want to change number?");
                        model.Warehouse.ChangeAnimalNumber(view.EnterAnimal(), Convert.ToIn
t32(view.EnterAnimalNumberWerehouse()));
                    }catch(InvalidCastException){
                        view.DisplayError("It is not the number!");
                    }
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                case 9: // buy animals, clear all basket
                    view.DisplayMasage("Buy all animals");
                    model.Client.BuyAllAnimals(model.Logs);
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                case 10: // change state
                    view.DisplayMasage("Change state of client: Active or Disactive?");
                    model.Client.State = view.GetState();
                    model.Notify();
                    //view.WaitAndClear();
                    break;
                default:
                    view.DisplayError("Wrong choise");
                    //view.WaitAndClear();
                    break;
            }
        }catch(InvalidCastException){
            view.DisplayError("It is not the number!");
            //view.WaitAndClear();
        }
        catch(FormatException){
            view.DisplayError("It is not the number!");
            //view.WaitAndClear();
        }
    }

  }
 }
}
/
```

```
*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 22:18
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Windows.Forms;
using PetShop.Builder.Director;
using PetShop.Builder.Builders;
using PetShop.M;
using PetShop.M.Classes.Product;
using PetShop.V;
using PetShop.V.WindowsApp;

namespace PetShop.C.Strategy.Strategies{

   public class WinAppStrategy : BaseStrategy{

      public override void InitModelAndView(Model model, PetShop.V.View view){
         this.model = model;
         this.view = view;
      }

      public override void Start(){
         // iniciation components
         view.InitComponent();
         // register event handler for button
         ((MainForm)view.Form).RegisterButton1EventHandler(new System.EventHandler(this.
Button1Click));
         ((MainForm)view.Form).RegisterButton2EventHandler(new System.EventHandler(this.
Button2Click));
         ((MainForm)view.Form).RegisterButton3EventHandler(new System.EventHandler(this.
Button3Click));
         ((MainForm)view.Form).RegisterButton4EventHandler(new System.EventHandler(this.
Button4Click));
         ((MainForm)view.Form).RegisterButton5EventHandler(new System.EventHandler(this.
Button5Click));
         ((MainForm)view.Form).RegisterButton6EventHandler(new System.EventHandler(this.
Button6Click));
         ((MainForm)view.Form).RegisterButton7EventHandler(new System.EventHandler(this.
Button7Click));
         ((MainForm)view.Form).RegisterComboBox1SelectionChanged(new System.EventHand
ler(this.ComboBox1SelectionChanged));

         view.StartApplication();
       }

      void Button1Click(object sender, EventArgs e){
         switch(view.EnterAnimal()){
```

```
            case("Dog"): // add dog
                try{
                    // dog builder
                    DogBreeder dogBreeder = new DogBreeder();
                    dogBreeder.DogBuilder = new DogDogBuilder();
                    dogBreeder.DogBuilder.CreateNewDog();
                    dogBreeder.ConstructDog(Convert.ToInt32(view.EnterAnimalNumberWerehous
e()), Convert.ToDouble(view.EnterPrice()));

                    model.Warehouse.AddAnimalToWarehouse(dogBreeder.Dog.Race, dogBreeder.Do
g);
                    dogBreeder = null;
                }catch(InvalidCastException){
                }
                break;
            case("Cat"): // add cat
                try{
                    // cat builder
                    CatBreeder catBreeder = new CatBreeder();
                    catBreeder.CatBuilder = new CatCatBuilder();
                    catBreeder.CatBuilder.CreateNewCat();
                    catBreeder.ConstructCat(Convert.ToInt32(view.EnterAnimalNumberWerehouse
()), Convert.ToDouble(view.EnterPrice()));

                    model.Warehouse.AddAnimalToWarehouse(catBreeder.Cat.Race,catBreeder.Cat);
                    catBreeder = null;
                }catch(InvalidCastException){
                    view.DisplayError("It is not the number!");
                }catch(FormatException){
                    view.DisplayError("It is not the number!");
                }
                break;
            case("Cow"): // add cow
                try{
                    // factory method
                    model.Warehouse.AddAnimalToWarehouse("Cow", Farm.FarmFactory(Animal.
Animals.Cow, Convert.ToInt32(view.EnterAnimalNumberWerehouse()), Convert.ToDouble(vie
w.EnterPrice())));
                }catch(InvalidCastException){
                    view.DisplayError("It is not the number!");
                }catch(FormatException){
                    view.DisplayError("It is not the number!");
                }
                break;
            case("Chicken"): // add Chicken
                try{
                    // factory method
                    model.Warehouse.AddAnimalToWarehouse("Chicken", Farm.FarmFactory(Anim
al.Animals.Chicken, Convert.ToInt32(view.EnterAnimalNumberWerehouse()), Convert.ToDoub
le(view.EnterPrice())));
                }catch(InvalidCastException){
                    view.DisplayError("It is not the number!");
```

```
            }catch(FormatException){
                view.DisplayError("It is not the number!");
            }
            break;
        default:
            view.DisplayError("Wrong choise");
            break;
    }
    model.Notify();
}

void Button2Click(object sender, EventArgs e){
    try{
        model.Warehouse.RemoveAnimalFromWarehouse(view.EnterAnimal(), Convert.ToInt32(view.EnterAnimalNumberWerehouse()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
    }catch(FormatException){
        view.DisplayError("It is not the number!");
    }
    model.Notify();
}

void Button3Click(object sender, EventArgs e){
    try{
        model.Warehouse.ChangeAnimalNumber(view.EnterAnimal(), Convert.ToInt32(view.EnterAnimalNumberWerehouse()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
    }catch(FormatException){
        view.DisplayError("It is not the number!");
    }
    model.Notify();
}

void Button4Click(object sender, EventArgs e){
    try{
        model.Warehouse.ChangeAnimalPrice(view.EnterAnimal(), Convert.ToDouble(view.EnterPrice()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
    }catch(FormatException){
        view.DisplayError("It is not the number!");
    }
    model.Notify();
}

void Button5Click(object sender, EventArgs e){
    try{
        model.Client.AddAnimalToClient(model.Warehouse, view.EnterAnimal(), Convert.ToInt32(view.EnterAnimalNumberClient()));
    }catch(InvalidCastException){
```

```
            view.DisplayError("It is not the number!");
        }catch(FormatException){
            view.DisplayError("It is not the number!");
        }
        model.Notify();
    }

    void Button6Click(object sender, EventArgs e){
        try{
            model.Warehouse.RemoveAnimalFromWarehouse(view.EnterAnimal(), Convert.ToIn
t32(view.EnterAnimalNumberWerehouse()));
        }catch(InvalidCastException){
            view.DisplayError("It is not the number!");
        }catch(FormatException){
            view.DisplayError("It is not the number!");
        }
        model.Notify();
    }

    void Button7Click(object sender, EventArgs e){
        model.Client.BuyAllAnimals(model.Logs);
        model.Notify();
    }

    void ComboBox1SelectionChanged(object sender, EventArgs e){
        model.Client.State = view.GetState();
    }
  }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 12/16/2014
 * Time: 00:38
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using PetShop.Builder.Director;
using PetShop.Builder.Builders;
using PetShop.M;
using PetShop.M.Classes.Product;
using PetShop.C.Strategy;
using PetShop.Factory;
using PetShop.V;
using PetShop.V.WindowsApp;

namespace PetShop.C.Strategy.Strategies{
   public class WPFAppStrategy : BaseStrategy{
      public override void InitModelAndView(Model model, View view){
         this.model = model;
         this.view = view;
      }

      public override void Start(){
         view.InitComponent();

         ((WPFForm)view.Form).RegisterButton1EventHandler(new System.Windows.RoutedEv
entHandler(this.Button1Click));
         ((WPFForm)view.Form).RegisterButton2EventHandler(new System.Windows.RoutedEv
entHandler(this.Button2Click));
         ((WPFForm)view.Form).RegisterButton3EventHandler(new System.Windows.RoutedEv
entHandler(this.Button3Click));
         ((WPFForm)view.Form).RegisterButton4EventHandler(new System.Windows.RoutedEv
entHandler(this.Button4Click));
         ((WPFForm)view.Form).RegisterButton5EventHandler(new System.Windows.RoutedEv
entHandler(this.Button5Click));
         ((WPFForm)view.Form).RegisterButton6EventHandler(new System.Windows.RoutedEv
entHandler(this.Button6Click));
         ((WPFForm)view.Form).RegisterButton7EventHandler(new System.Windows.RoutedEv
entHandler(this.Button7Click));
         ((WPFForm)view.Form).RegisterComboBox1EventHendler(new System.Windows.Contr
ols.SelectionChangedEventHandler(this.ComboBox1Change));
         ((WPFForm)view.Form).RegisterComboBox2EventHendler(new System.Windows.Contr
ols.SelectionChangedEventHandler(this.ComboBox2Change));
```

```
        ((WPFForm)view.Form).RegisterComboBox3EventHendler(new System.Windows.Contr
ols.SelectionChangedEventHandler(this.ComboBox3Change));
        ((WPFForm)view.Form).RegisterComboBox4EventHendler(new System.Windows.Contr
ols.SelectionChangedEventHandler(this.ComboBox4Change));
        ((WPFForm)view.Form).RegisterComboBox5EventHendler(new System.Windows.Contr
ols.SelectionChangedEventHandler(this.ComboBox5Change));
        view.StartApplication();
    }

    void Button1Click(object sender, RoutedEventArgs e){
      switch(view.EnterAnimal()){
        case("Dog"): // add dog
          try{
            // dog builder
            DogBreeder dogBreeder = new DogBreeder();
            dogBreeder.DogBuilder = new DogDogBuilder();
            dogBreeder.DogBuilder.CreateNewDog();
            dogBreeder.ConstructDog(Convert.ToInt32(view.EnterAnimalNumberWerehous
e()), Convert.ToDouble(view.EnterPrice()));

            model.Warehouse.AddAnimalToWarehouse(dogBreeder.Dog.Race, dogBreeder.Do
g);
            dogBreeder = null;
          }catch(InvalidCastException){
          }
          break;
        case("Cat"): // add cat
          try{
            // cat builder
            CatBreeder catBreeder = new CatBreeder();
            catBreeder.CatBuilder = new CatCatBuilder();
            catBreeder.CatBuilder.CreateNewCat();
            catBreeder.ConstructCat(Convert.ToInt32(view.EnterAnimalNumberWerehouse
()), Convert.ToDouble(view.EnterPrice()));

            model.Warehouse.AddAnimalToWarehouse(catBreeder.Cat.Race,catBreeder.Cat);
            catBreeder = null;
          }catch(InvalidCastException){
            view.DisplayError("It is not the number!");
          }catch(FormatException){
            view.DisplayError("It is not the number!");
          }
          break;
        case("Cow"): // add cow
          try{
            // factory method
            model.Warehouse.AddAnimalToWarehouse("Cow", Farm.FarmFactory(Animal.
Animals.Cow, Convert.ToInt32(view.EnterAnimalNumberWerehouse()), Convert.ToDouble(vie
w.EnterPrice())));
          }catch(InvalidCastException){
            view.DisplayError("It is not the number!");
          }catch(FormatException){
```

```
                view.DisplayError("It is not the number!");
            }
            break;
        case("Chicken"): // add Chicken
            try{
                // factory method
                model.Warehouse.AddAnimalToWarehouse("Chicken", Farm.FarmFactory(Anim
al.Animals.Chicken, Convert.ToInt32(view.EnterAnimalNumberWerehouse()), Convert.ToDoub
le(view.EnterPrice())));
            }catch(InvalidCastException){
                view.DisplayError("It is not the number!");
            }catch(FormatException){
                view.DisplayError("It is not the number!");
            }
            break;
        default:
            view.DisplayError("Wrong choise");
            break;
    }
    model.Notify();
}

void Button2Click(object sender, EventArgs e){
    try{
        model.Warehouse.RemoveAnimalFromWarehouse(view.EnterAnimal(), Convert.ToIn
t32(view.EnterAnimalNumberWerehouse()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
    }catch(FormatException){
        view.DisplayError("It is not the number!");
    }
    model.Notify();
}

void Button3Click(object sender, EventArgs e){
    try{
        model.Warehouse.ChangeAnimalNumber(view.EnterAnimal(), Convert.ToInt32(view.
EnterAnimalNumberWerehouse()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
    }catch(FormatException){
        view.DisplayError("It is not the number!");
    }
    model.Notify();
}

void Button4Click(object sender, EventArgs e){
    try{
        model.Warehouse.ChangeAnimalPrice(view.EnterAnimal(), Convert.ToDouble(view.E
nterPrice()));
    }catch(InvalidCastException){
        view.DisplayError("It is not the number!");
```

```
        }catch(FormatException){
            view.DisplayError("It is not the number!");
        }
        model.Notify();
    }

    void Button5Click(object sender, EventArgs e){
        try{
        model.Client.AddAnimalToClient(model.Warehouse,
                                ((Animal)((WPFForm)view.Form).GetWGrid().SelectedItem).Name(),
                                Convert.ToInt32(view.EnterAnimalNumberClient()));
        }catch(Exception){
            view.DisplayError("Animal is not chose!");
        }
        model.Notify();
    }

    void Button6Click(object sender, EventArgs e){
        try{
            model.Client.RemoveAnimalFromClient(model.Warehouse, view.EnterAnimal(), Conv
ert.ToInt32(view.EnterAnimalNumberClient()));
        }catch(InvalidCastException){
            view.DisplayError("It is not the number!");
        }catch(FormatException){
            view.DisplayError("It is not the number!");
        }
        model.Notify();
    }

    void Button7Click(object sender, EventArgs e){
        model.Client.BuyAllAnimals(model.Logs);
        model.Notify();
    }

    void ComboBox5Change(object sender, EventArgs e){
        var comboBox = sender as ComboBox;
        if(comboBox.SelectedIndex == 0){
            model.Client.State = "Active";
        }
        else if(comboBox.SelectedIndex == 1){
            model.Client.State = "Disactive";
        }
        model.Notify();
    }

    void ComboBox1Change(object sender, EventArgs e){
        var comboBox = sender as ComboBox;
        if(comboBox.SelectedIndex == 0){
            ((WPFForm)view.Form).SpeciesComboBox.Visibility = Visibility.Hidden;
            ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Hidden;
            ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Hidden;
        }
```

```
            else if(comboBox.SelectedIndex == 1){
                ((WPFForm)view.Form).SpeciesComboBox.Visibility = Visibility.Visible;
                ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Hidden;
                ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Hidden;
            }
            model.Notify();
        }

        void ComboBox2Change(object sender, EventArgs e){
            var comboBox = sender as ComboBox;
            if(comboBox.SelectedIndex == 0){
                ((WPFForm)view.Form).SpeciesComboBox.Visibility = Visibility.Hidden;
                ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Hidden;
                ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Hidden;
            }
            else if(comboBox.SelectedIndex == 1){
                ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Visible;
                ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Hidden;
            }
            else if(comboBox.SelectedIndex == 2){
                ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Visible;
                ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Hidden;
            }
            model.Notify();
        }

        void ComboBox3Change(object sender, EventArgs e){
            var comboBox = sender as ComboBox;
            if(comboBox.SelectedIndex == 0){
                ((WPFForm)view.Form).RacesPetComboBox.Visibility = Visibility.Hidden;
            }
            model.Notify();
        }

        void ComboBox4Change(object sender, EventArgs e){
            var comboBox = sender as ComboBox;
            if(comboBox.SelectedIndex == 0){
                ((WPFForm)view.Form).RacesFarmComboBox.Visibility = Visibility.Hidden;
            }
            model.Notify();
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 11:02
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.Observer;
using PetShop.M.Classes.Container;
using PetShop.M.Classes.Mediator;
using PetShop.M.Classes.Product;
using PetShop.V;

namespace PetShop.M{

    public class Model : IObservable {
        Client client;
        Warehouse warehouse;
        Logs logs;

        readonly List<View> views = new List<View>();

        public Model(){
            client = new Client();
            warehouse = Warehouse.InstanceWarehouse;
            logs = new Logs();
        }

        public Client Client{
            get { return client; }
            set { client = value; }
        }

        public Logs Logs{
            get { return logs; }
            set { logs = value; }
        }

        public Warehouse Warehouse{
            get { return warehouse; }
            set { warehouse = value; }
        }
        /*
         * Observator pattern
         */
```

```
        public void Attach(View view){
            views.Add(view);
        }
        public void Detach(View view){
            views.Remove(view);
        }
        public void Notify(){
            foreach (View view in views){
                view.Update(this);
            }
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 11:05
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.M.Classes.Product;

namespace PetShop.M.Classes.Container{
   public class ProductContainer{

      protected Dictionary<string, Animal> instance = new Dictionary<string, Animal>();

      public Dictionary<string, Animal> Instance{
         get { return instance; }
         set { instance = value; }
      }

      public void Add(string race, Animal animal){
         try{
            instance.Add(race, animal);
         }catch(ArgumentException){

         }
      }

      public Dictionary<string, Animal>.ValueCollection GetValues(){
         return instance.Values;
      }

      public Dictionary<string, Animal>.KeyCollection GetKeys(){
         return instance.Keys;
      }

      public List<T> getAnimal<T>(){
         List<T> list = new List<T>();
         foreach (KeyValuePair<string, Animal> animal in this.Instance){
            if (typeof(T) == animal.Value.GetType() ||
               typeof(T) == animal.Value.GetType().BaseType ||
               typeof(T) == animal.Value.GetType().BaseType.BaseType){
                  list.Add((T)(object)animal.Value);
            }
         }
         return list;
      }
```

```
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 11:06
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.M.Classes.Product;
using PetShop.M.Classes.Mediator;

namespace PetShop.M.Classes.Container{
   public class Client : ProductContainer{
      double sum;
      string state;
      double credit;

      public Client(){
         this.sum = 0;
         this.state = "Active";
         this.credit = 1000;
      }

      public double Credit{
         get { return credit; }
         set { credit = value; }
      }

      public double Sum{
         get { return sum; }
         set { sum = value; }
      }
      // state, state pattern
      public string State{
         get { return state; }
         set { state = value; }
      }
      /*
       * method to chack client can buy than amount of product with given price
       */
      private Boolean CanBuy(int number, double price){
         if(this.State.Equals("Active")){
            if(this.Credit < (number * price)){
               return false;
            }
         }
         return true;
```

```
}
/*
 * calculate sum, add credit when buy items
 */
public void CalculateSum(){
   if(this.State.Equals("Active")){
      this.sum = 0;
      this.credit = 1000;
      foreach(Animal animal in instance.Values){
         this.sum = this.sum + (animal.Number * animal.Price);
         this.credit -= (animal.Number * animal.Price);
      }
   }
}

/*
 *  Add animal to client's basket only if list don't have animal with this type : prototype pattern
 */
public void AddAnimalToClient(Warehouse warehouse, string key, int number){
   if(this.State.Equals("Active")){
      if(warehouse.Instance[key].Number >= number){
         if(this.CanBuy(number, warehouse.Instance[key].Price)){
            Animal animal = (Animal)warehouse.Instance[key].Clone();
            animal.Number = number;
            warehouse.Instance[key].Number -= number;
            if(this.Instance.ContainsKey(key)){
               this.Instance[key].Number += number;
            }
            else{
               this.Add(key, animal);
            }
            this.CalculateSum();
         }
      }
   }
}

/*
 * Remove item form client
 */
public void RemoveAnimalFromClient(ProductContainer pc, string key, int number){
   try{
      if(this.Instance[key].Number >= number){
         if (this.Instance[key].Number != 0){
            this.Instance[key].Number -= number;
            pc.Instance[key].Number += number;
         }
      }
   }catch(KeyNotFoundException){

   }
}
```

```
        /*
         * Create new list for client
         */
        public void BuyAllAnimals(Logs logs){
           if(this.State.Equals("Active")){
              // mediator pattern
              logs.addRegistry(new Registry(this.Instance.Values.ToList<Animal>(), this.Sum));
              List<string> list;
              list = new List<string>();
              foreach(string s in this.GetKeys())
                 list.Add(s);
              foreach(string s in list)
                 this.Instance.Remove(s);
              this.CalculateSum();
           }
        }
     }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 11:03
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.M.Classes.Product;

namespace PetShop.M.Classes.Container{
   public class Warehouse: ProductContainer {

      private static Warehouse instanceWarehouse;

      private Warehouse() {}
      // singleton pattern
      public static Warehouse InstanceWarehouse{
         get {
            if (instanceWarehouse == null){
               instanceWarehouse = new Warehouse();
            }
            return instanceWarehouse;
         }
      }

      /*
       *  Add animal to warehouse only if list don't have animal with this type
       */
      public void AddAnimalToWarehouse(string race, Animal animal){
         this.Add(race, animal);
      }

      public void AddNumberOfAnimal(string race, int number){
         try{
            this.Instance[race].Number += number;
         }catch(KeyNotFoundException){

         }
      }

      /*
       * Remove item form warehouse
       */
      public void RemoveAnimalFromWarehouse(string key, int number){
         try{
            if(this.Instance[key].Number >= number){
```

```
                if (this.Instance[key].Number != 0){
                    this.Instance[key].Number -= number;
                }
            }
        }catch(KeyNotFoundException){

        }
    }

    /*
     * Change price
     */
    public void ChangeAnimalPrice(string key, double price){
        try{
            this.Instance[key].Price = price;
        }catch(KeyNotFoundException){

        }
    }

    /*
     * Change namber of animal
     */
    public void ChangeAnimalNumber(string key, int number){
        try{
            this.Instance[key].Number = number;
        }catch(KeyNotFoundException){

        }
    }

  }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-02
 * Time: 13:36
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PetShop.M.Classes.Product{
    /*
     * Main abstract class, prototype pattern use
     */
    public abstract class Animal : ICloneable {
        protected int number;
        protected double price;

        public enum Animals { Dog, Cat, Cow, Chicken }

        public Animal(){ }
        public int Number{
            get {return number;}
            set {number = value;}
        }

        public double Price{
            get {return price;}
            set {price = value;}
        }

        public abstract string Name();
        // method to clone object
        public object Clone(){
            return this.MemberwiseClone();
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-02
 * Time: 13:38
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using PetShop.Factory;

namespace PetShop.M.Classes.Product{
   public abstract class Pet : Animal{
      protected string species;

      public Pet(){}

      public string Species{
         get {return species;}
         set {species = value;}
      }
   }


   public abstract class Farm : Animal{

      protected string species;

      public Farm(){}

      public static Farm FarmFactory(Animals farmType, int number, double price){
         switch(farmType){
            case(Animals.Cow):
               return new Cow(number, price, "Farm", "Cow");
            case(Animals.Chicken):
               return new Chicken(number, price, "Farm", "Chicken");
            default:
               break;
         }
         throw new System.NotSupportedException("The pizza type " + farmType.ToString() + "
is not recognized.");
      }

      public string Species{
         get {return species;}
         set {species = value;}
      }
   }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-02
 * Time: 13:38
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;

namespace PetShop.M.Classes.Product{
   public class Dog : Pet{
      protected string race;

      public Dog(){}

      public string Race{
         get {return race;}
         set {race = value;}
      }

      public override string Name(){
         return "Dog";
      }

   }

   public class Cat : Pet{
      protected string race;

      public Cat(){}

      public string Race{
         get {return race;}
         set {race = value;}
      }

      public override string Name(){
         return "Cat";
      }
   }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-15
 * Time: 15:55
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PetShop.M.Classes.Mediator{
   public class Logs {
      List<Registry> list;

      internal List<Registry> List{
         get { return list; }
         set { list = value; }
      }

      public Logs(List<Registry> list){
         this.list = list;
      }

      public Logs(){
         this.list = new List<Registry>();
      }

      public void addRegistry(Registry registry){
         list.Add(registry);
      }
   }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-15
 * Time: 15:55
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.M.Classes.Product;

namespace PetShop.M.Classes.Mediator{
   public class Registry{
      List<Animal> list;
      double sum;
      DateTime date;

      public List<Animal> List{
         get { return list; }
         set { list = value; }
      }


      public DateTime Date{
         get { return date; }
         set { date = value; }
      }

      public double Sum{
         get { return sum; }
         set { sum = value; }
      }


      public Registry(List<Animal> list, double sum){
         this.list = list;
         this.date = DateTime.Now;
         this.sum = sum;
      }
   }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-03
 * Time: 10:57
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.Observer;
using PetShop.M.Classes.Product;
using PetShop.M;
using PetShop.M.Classes.Mediator;

namespace PetShop.V{

   public abstract class View : IObserver{

      public enum Animals { Dog, Cat, Cow, Chicken }

      protected Object form;

      public Object Form{
         get { return form; }
         set { form = value; }
      }

      public abstract void InitComponent();
      public abstract void StartApplication();

      public abstract void Update(Model model);

      public abstract void DisplayError(string error);
      public abstract void DisplayMasage(string msg);

      public abstract void DisplayWarehouseStatus(Dictionary<string, Animal>.ValueCollection
list);
      public abstract void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection
list, double sum, double credit);

      public abstract void DisplayLogs(Logs logs);

      public abstract void DisplayAvailableAnimals();
      public abstract void DisplayMainOptions();
      public abstract void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection list);

      public abstract string EnterOption();
```

```
        public abstract string EnterAnimalNumberWerehouse();
        public abstract string EnterAnimalNumberClient();
        public abstract string EnterAnimal();
        public abstract string EnterPrice();

        public abstract string GetState();
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 18:48
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.M.Classes.Mediator;
using PetShop.M.Classes.Product;
using PetShop.M;

namespace PetShop.V.Views{
   public class ConsoleView : View{

      public override void InitComponent(){ }

      public override void StartApplication(){
         Console.WriteLine("Start: Dominik Slusarz");
      }

      public override void Update(Model model){
         Console.WriteLine("View was updated");
         this.DisplayWarehouseStatus(model.Warehouse.GetValues());
         this.DisplayClientStatus(model.Client.GetValues(), model.Client.Sum,model.Client.Credit
);
         this.DisplayLogs(model.Logs);
         this.WaitAndClear();
      }

      private void WaitAndClear(){
         Console.WriteLine("Press any key");
         Console.ReadKey();
         Console.Clear();
      }

      public override void DisplayLogs(Logs logs){
         foreach(Registry registry in logs.List){
            Console.WriteLine("Logs:");
            foreach (Animal animal in registry.List){
               Console.WriteLine("I'm " + animal.GetType() + " number " + animal.Number + "
price " + animal.Price + "\n");
            }
            Console.WriteLine("Data "+ registry.Date);
         }
      }
```

```
public override string GetState(){
   return Console.ReadLine();
}

public override string EnterOption(){
   return Console.ReadLine();
}


public override string EnterAnimalNumberWerehouse(){
   return Console.ReadLine();
}

public override string EnterAnimalNumberClient(){
   return Console.ReadLine();
}

public override string EnterAnimal(){
   return Console.ReadLine();
}

public override string EnterPrice(){
   return Console.ReadLine();
}

public override void DisplayAvailableAnimals(){
   foreach(Animals element in Enum.GetValues(typeof(Animals))){
      Console.WriteLine(element.ToString());
   }
}

public override void DisplayMainOptions(){
   Console.WriteLine("0 - Exit\n" +
                     "1 - Add animal to warehouse\n" +
                     "2 - Show warehouse status\n" +
                     "3 - Remove from warehouse\n" +
                     "4 - Add to basket\n" +
                     "5 - Display basket\n" +
                     "6 - Remove from basket\n" +
                     "7 - Change price\n" +
                     "8 - Change number\n" +
                     "9 - Buy\n" +
                     "10 - Change state");
}

public override void DisplayError(string error){
   Console.WriteLine(" ---- " + error + " ----");
}

public override void DisplayMasage(string msg){
   Console.WriteLine(msg);
```

```
        }

        public override void DisplayWarehouseStatus(Dictionary<string, Animal>.ValueCollection
list){
            Console.WriteLine("Warehouse status:");
            int i = 0;
            foreach(Animal animal in list){
                string s = "I'm " + animal.GetType() +  " number " + animal.Number + " price
" + animal.Price + "\n";
                Console.WriteLine(i++ + "- " + s);
            }
        }

        public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection
list, double sum, double credit){
            Console.WriteLine("Client basket:");
            foreach(Animal animal in list){
                string s = "I'm " + animal.GetType() +  " number " + animal.Number + " price
" + animal.Price + "\n";;
                Console.WriteLine(s);
            }
            Console.WriteLine("Price " + sum);
        }

        public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection list){
            Console.WriteLine("Client basket:");
            foreach(Animal animal in list){
                string s ="I'm " + animal.GetType() + " number " + animal.Number + " price
" + animal.Price + "\n";
                Console.WriteLine(s);
            }
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 18:56
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using PetShop.M;
using PetShop.M.Classes.Product;
using PetShop.M.Classes.Mediator;
using PetShop.V;
using PetShop.V.WindowsApp;

namespace PetShop.V.Views{

   public class WinAppView : View{

      public override void InitComponent(){
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         this.Form = new MainForm();
      }

      public override void StartApplication(){
         Application.Run((MainForm)Form);
      }

      public override void Update(Model model){
         this.DisplayWarehouseStatus(model.Warehouse.GetValues());
         this.DisplayClientStatus(model.Client.GetValues(), model.Client.Sum, model.Client.Credi
t);
         this.DisplayLogs(model.Logs);
      }

      public override string GetState(){
         return ((MainForm)Form).GetState();
      }

      public override string EnterAnimalNumberWerehouse(){
         return ((MainForm)Form).GetNumber();
      }

      public override string EnterAnimalNumberClient(){
         return ((MainForm)Form).GetNumber();
```

```
    }

    public override string EnterAnimal(){
      return ((MainForm)Form).GetAnimal();
    }

    public override string EnterPrice(){
      return ((MainForm)Form).GetPrice();
    }

    public override void DisplayLogs(Logs logs){
      string s = "Logs:\n";
      foreach (Registry registry in logs.List){
        foreach (Animal animal in registry.List){
          s += animal.Name() + " number " + animal.Number + " price " + animal.Price + "\n";
        }
        s += "Suma = "+ registry.Sum +"\nData " + registry.Date + "\n--------------------\n";
      }
      ((MainForm)Form).SetTextLogs(s);
    }

    public override void DisplayError(string error){
      MessageBox.Show(error,
                "Error",
                MessageBoxButtons.OK,
                MessageBoxIcon.Exclamation,
                MessageBoxDefaultButton.Button1);
    }

    public override void DisplayMasage(string msg){
      MessageBox.Show(msg,
                "Message",
                MessageBoxButtons.OK,
                MessageBoxIcon.Exclamation,
                MessageBoxDefaultButton.Button1);
    }

    public override void DisplayWarehouseStatus(Dictionary<string, Animal>.ValueCollection
list){
      string s = "Warehouse status:\n";
      foreach(Animal animal in list){
        s += animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
      }
      ((MainForm)Form).SetTextWarehouse(s);
    }

    public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection
list, double sum, double credit){
      string s  = "Client basket:\n";
      foreach(Animal animal in list){
        s +=  animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
      }
```

```
            s +="Price " + sum;
            ((MainForm)Form).SetTextClient(s);
        }

        public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection list){
            string s  = "Client basket:\n";
            foreach(Animal animal in list){
                s +=  animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
            }
            ((MainForm)Form).SetTextClient(s);
        }

        public override void DisplayAvailableAnimals(){
            string s = "";
            foreach(Animals element in Enum.GetValues(typeof(Animals))){
                s += element.ToString();
            }
            MessageBox.Show(s,
                        "Avalible animals",
                        MessageBoxButtons.OK,
                        MessageBoxIcon.Exclamation,
                        MessageBoxDefaultButton.Button1);
        }

        public override void DisplayMainOptions(){    }

        public override string EnterOption(){
            return "";
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 12/16/2014
 * Time: 00:04
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using PetShop.M.Classes.Product;
using PetShop.M;
using PetShop.M.Classes.Container;
using PetShop.M.Classes.Mediator;
using PetShop.V.WindowsApp;
using PetShop.Factory;

namespace PetShop.V.Views{

   public class WPFAppView : View{
      public override void InitComponent(){
         this.Form = new WPFForm();
      }

      public override void StartApplication(){
         PetShop.App app = new PetShop.App();
         app.Run((WPFForm)this.Form);
      }

      public override void Update(Model model){
         this.DisplayWarehouseStatus(model.Warehouse.GetValues());
         this.DisplayClientStatus(model.Client.GetValues(), model.Client.Sum, model.Client.Credi
t);
         this.DisplayLogs(model.Logs);
         this.DisplayWGrid(model.Warehouse);
         this.DisplayCGrid(model.Client);
      }

      public void DisplayWGrid(Warehouse warehouse){
         switch(((WPFForm)Form).GetSelectedComboBox()){
            case 0:
```

```
            break;
        case 1:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Animal>();
            break;
        case 2:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Pet>();
            break;
        case 3:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Farm>();
            break;
        case 4:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Cat>();
            break;
        case 5:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Dog>();
            break;
        case 6:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Cow>();
            break;
        case 7:
            ((WPFForm)Form).GetWGrid().ItemsSource = warehouse.getAnimal<Chicken>();
            break;
    }
}

public void DisplayCGrid(Client client){
    switch(((WPFForm)Form).GetSelectedComboBox()){
        case 0:
            break;
        case 1:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Animal>();
            break;
        case 2:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Pet>();
            break;
        case 3:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Farm>();
            break;
        case 4:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Cat>();
            break;
        case 5:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Dog>();
            break;
        case 6:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Cow>();
            break;
        case 7:
            ((WPFForm)Form).GetCGrid().ItemsSource = client.getAnimal<Chicken>();
            break;
    }
}
```

```
        public override string GetState(){
            return "";
        }

        public override string EnterAnimalNumberWerehouse(){
            return ((WPFForm)Form).GetNumberWerehouse();
        }

        public override string EnterAnimalNumberClient(){
            return ((WPFForm)Form).GetNumberClient();
        }

        public override string EnterAnimal(){
            return ((WPFForm)Form).GetAnimal();
        }

        public override string EnterPrice(){
            return ((WPFForm)Form).GetPrice();
        }

        public override void DisplayLogs(Logs logs){
            string s = "Logs:\n";

            foreach (Registry registry in logs.List){
                foreach (Animal animal in registry.List){
                    s +=  animal.Name() + " number " + animal.Number + " price " + animal.Price + "
" + "\n";
                }
                s += "Sum = "+ registry.Sum +"\nDate " + registry.Date + "\n--------------------\n";
            }
            ((WPFForm)Form).SetTextLogs(s);
        }

        public override void DisplayError(string error){
            MessageBoxResult result =
System.Windows.MessageBox.Show(error, "Error", MessageBoxButton.OK, MessageBoxImage.Er
ror);
        }

        public override void DisplayMasage(string msg){
            MessageBoxResult result =
System.Windows.MessageBox.Show(msg, "Massage", MessageBoxButton.OK, MessageBoxImage
.Information);
        }

        public override void DisplayWarehouseStatus(Dictionary<string, Animal>.ValueCollection
list){
            string s = "Warehouse status:\n";
            foreach(Animal animal in list){
                s += animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
            }
```

```
            ((WPFForm)Form).SetTextWarehouse(s);
        }

        public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection
list, double sum, double credit){
            string s  = "Credit = "+ credit +"\nClient basket:\n";
            foreach(Animal animal in list){
                s += animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
            }
            s +="Price " + sum;
            ((WPFForm)Form).SetTextClient(s);
        }

        public override void DisplayClientStatus(Dictionary<string, Animal>.ValueCollection list){
            string s  = "Client basket:\n";
            foreach(Animal animal in list){
                s += animal.Name() +  " number " + animal.Number + " price " + animal.Price + "\n";
            }
            ((WPFForm)Form).SetTextClient(s);
        }

        public override void DisplayAvailableAnimals(){
            string s = "";
            foreach(Animals element in Enum.GetValues(typeof(Animals))){
                s += element.ToString();
            }
        }

        public override void DisplayMainOptions(){
        }

        public override string EnterOption(){
            return "";
        }
    }
}
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-09
 * Time: 19:16
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Windows.Forms;

namespace PetShop.V.WindowsApp{

   public class MainForm : Form{

      public MainForm(){
         InitializeComponent();
      }
      private System.ComponentModel.IContainer components = null;

      protected override void Dispose(bool disposing){
         if (disposing){
            if (components != null){
               components.Dispose();
            }
         }
         base.Dispose(disposing);
      }

      private void InitializeComponent(){
         this.label1 = new System.Windows.Forms.Label();
         this.button1 = new System.Windows.Forms.Button();
         this.button2 = new System.Windows.Forms.Button();
         this.button4 = new System.Windows.Forms.Button();
         this.button3 = new System.Windows.Forms.Button();
         this.button5 = new System.Windows.Forms.Button();
         this.button6 = new System.Windows.Forms.Button();
         this.button7 = new System.Windows.Forms.Button();
         this.label2 = new System.Windows.Forms.Label();
         this.label3 = new System.Windows.Forms.Label();
         this.label4 = new System.Windows.Forms.Label();
         this.label5 = new System.Windows.Forms.Label();
         this.textBox1 = new System.Windows.Forms.TextBox();
         this.textBox2 = new System.Windows.Forms.TextBox();
         this.textBox3 = new System.Windows.Forms.TextBox();
         this.label6 = new System.Windows.Forms.Label();
         this.comboBox1 = new System.Windows.Forms.ComboBox();
         this.label7 = new System.Windows.Forms.Label();
         this.SuspendLayout();
```

```
//
// label1
//
this.label1.Location = new System.Drawing.Point(167, 61);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(307, 110);
this.label1.TabIndex = 0;
this.label1.Text = "Warehouse";
//
// button1
//
this.button1.Location = new System.Drawing.Point(16, 61);
this.button1.Name = "button1";
this.button1.Size = new System.Drawing.Size(145, 23);
this.button1.TabIndex = 1;
this.button1.Text = "Add animal to warehouse";
this.button1.UseVisualStyleBackColor = true;
//
// button2
//
this.button2.Location = new System.Drawing.Point(16, 90);
this.button2.Name = "button2";
this.button2.Size = new System.Drawing.Size(145, 23);
this.button2.TabIndex = 2;
this.button2.Text = "Remove animal from warehouse";
this.button2.UseVisualStyleBackColor = true;
//
// button4
//
this.button4.Location = new System.Drawing.Point(16, 148);
this.button4.Name = "button4";
this.button4.Size = new System.Drawing.Size(145, 23);
this.button4.TabIndex = 4;
this.button4.Text = "Change price";
this.button4.UseVisualStyleBackColor = true;
//
// button3
//
this.button3.Location = new System.Drawing.Point(17, 119);
this.button3.Name = "button3";
this.button3.Size = new System.Drawing.Size(145, 23);
this.button3.TabIndex = 5;
this.button3.Text = "Change number of animal";
this.button3.UseVisualStyleBackColor = true;
//
// button5
//
this.button5.Location = new System.Drawing.Point(16, 205);
this.button5.Name = "button5";
this.button5.Size = new System.Drawing.Size(146, 23);
this.button5.TabIndex = 6;
this.button5.Text = "Add to basket";
```

```
this.button5.UseVisualStyleBackColor = true;
//
// button6
//
this.button6.Location = new System.Drawing.Point(16, 234);
this.button6.Name = "button6";
this.button6.Size = new System.Drawing.Size(146, 23);
this.button6.TabIndex = 7;
this.button6.Text = "Remove from basket";
this.button6.UseVisualStyleBackColor = true;
//
// button7
//
this.button7.Location = new System.Drawing.Point(16, 263);
this.button7.Name = "button7";
this.button7.Size = new System.Drawing.Size(146, 23);
this.button7.TabIndex = 8;
this.button7.Text = "Buy";
this.button7.UseVisualStyleBackColor = true;
//
// label2
//
this.label2.Location = new System.Drawing.Point(168, 196);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(306, 81);
this.label2.TabIndex = 9;
this.label2.Text = "Client basket";
//
// label3
//
this.label3.Location = new System.Drawing.Point(16, 9);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(100, 23);
this.label3.TabIndex = 10;
this.label3.Text = "Animal";
//
// label4
//
this.label4.Location = new System.Drawing.Point(122, 9);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(100, 23);
this.label4.TabIndex = 11;
this.label4.Text = "Number";
//
// label5
//
this.label5.Location = new System.Drawing.Point(228, 9);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(100, 23);
this.label5.TabIndex = 12;
this.label5.Text = "Price";
//
```

```
// textBox1
//
this.textBox1.Location = new System.Drawing.Point(16, 35);
this.textBox1.Name = "textBox1";
this.textBox1.Size = new System.Drawing.Size(100, 20);
this.textBox1.TabIndex = 13;
//
// textBox2
//
this.textBox2.Location = new System.Drawing.Point(122, 35);
this.textBox2.Name = "textBox2";
this.textBox2.Size = new System.Drawing.Size(100, 20);
this.textBox2.TabIndex = 14;
//
// textBox3
//
this.textBox3.Location = new System.Drawing.Point(228, 35);
this.textBox3.Name = "textBox3";
this.textBox3.Size = new System.Drawing.Size(100, 20);
this.textBox3.TabIndex = 15;
//
// label6
//
this.label6.Location = new System.Drawing.Point(167, 299);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(265, 65);
this.label6.TabIndex = 16;
this.label6.Text = "Logs";
//
// comboBox1
//
this.comboBox1.FormattingEnabled = true;
this.comboBox1.Items.AddRange(new object[] {
                "Active",
                "Disactive"});
this.comboBox1.Location = new System.Drawing.Point(17, 177);
this.comboBox1.Name = "comboBox1";
this.comboBox1.Size = new System.Drawing.Size(121, 21);
this.comboBox1.TabIndex = 17;
//
// label7
//
this.label7.Location = new System.Drawing.Point(17, 299);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(120, 44);
this.label7.TabIndex = 18;
this.label7.Text = "label7";
//
// MainForm
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
```

```
    this.ClientSize = new System.Drawing.Size(489, 404);
    this.Controls.Add(this.label7);
    this.Controls.Add(this.comboBox1);
    this.Controls.Add(this.label6);
    this.Controls.Add(this.textBox3);
    this.Controls.Add(this.textBox2);
    this.Controls.Add(this.textBox1);
    this.Controls.Add(this.label5);
    this.Controls.Add(this.label4);
    this.Controls.Add(this.label3);
    this.Controls.Add(this.label2);
    this.Controls.Add(this.button7);
    this.Controls.Add(this.button6);
    this.Controls.Add(this.button5);
    this.Controls.Add(this.button3);
    this.Controls.Add(this.button4);
    this.Controls.Add(this.button2);
    this.Controls.Add(this.button1);
    this.Controls.Add(this.label1);
    this.Name = "MainForm";
    this.Text = "Dominik Slusarz";
    this.ResumeLayout(false);
    this.PerformLayout();
}
private System.Windows.Forms.Label label7;
private System.Windows.Forms.ComboBox comboBox1;
private System.Windows.Forms.Label label6;

private System.Windows.Forms.TextBox textBox3;
private System.Windows.Forms.TextBox textBox2;
private System.Windows.Forms.TextBox textBox1;
private System.Windows.Forms.Label label5;
private System.Windows.Forms.Label label4;
private System.Windows.Forms.Label label3;

private System.Windows.Forms.Label label1;
private System.Windows.Forms.Label label2;
private System.Windows.Forms.Button button7;
private System.Windows.Forms.Button button6;
private System.Windows.Forms.Button button5;
private System.Windows.Forms.Button button3;
private System.Windows.Forms.Button button4;
private System.Windows.Forms.Button button2;
private System.Windows.Forms.Button button1;

public void RegisterButton1EventHandler(System.EventHandler eventHandler){
    this.button1.Click += new System.EventHandler(eventHandler);
}

public void RegisterButton2EventHandler(System.EventHandler eventHandler){
    this.button2.Click += new System.EventHandler(eventHandler);
}
```

```
public void RegisterButton3EventHandler(System.EventHandler eventHandler){
   this.button3.Click += new System.EventHandler(eventHandler);
}

public void RegisterButton4EventHandler(System.EventHandler eventHandler){
   this.button4.Click += new System.EventHandler(eventHandler);
}

public void RegisterButton5EventHandler(System.EventHandler eventHandler){
   this.button5.Click += new System.EventHandler(eventHandler);
}

public void RegisterButton6EventHandler(System.EventHandler eventHandler){
   this.button6.Click += new System.EventHandler(eventHandler);
}

public void RegisterButton7EventHandler(System.EventHandler eventHandler){
   this.button7.Click += new System.EventHandler(eventHandler);
}

public void RegisterComboBox1SelectionChanged(System.EventHandler eventHandler){
   this.comboBox1.SelectedIndexChanged += new System.EventHandler(eventHandler);
}

public void SetTextWarehouse(string text){
   this.label1.Text = text;
}

public void SetTextClient(string text){
   this.label2.Text = text;
}

public void SetTextLogs(string text){
   this.label6.Text = text;
}

public void SetTest(string text){
   this.label7.Text = text;
}

public string GetAnimal(){
   return this.textBox1.Text;
}

public string GetNumber(){
   return this.textBox2.Text;
}

public string GetPrice(){
   return this.textBox3.Text;
}
```

```
        public string GetState(){
            return this.comboBox1.Text;
        }

    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<Window
    x:Class="PetShop.V.WindowsApp.WPFForm" xmlns="http://schemas.microsoft.com/winfx/2006
/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Dominik Slusarz"
    Height="700"
    Width="800" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://
schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">
    <Grid
        RenderTransformOrigin="0.5,0.5"
        Width="788.2"
        Height="669.1">
        <Label
            Grid.Column="0"
            Grid.Row="0"
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="8,8,0,0"
            Width="75"
            Height="27"
            Content="Animal"
            x:Name="AnimalLabel" />
        <Button
            Content="Add animal to warehouse"
            RenderTransformOrigin="0.5,0.5"
            x:Name="AddAnimalToWarehouseButton"
            x:FieldModifier="private"
            Grid.Column="0"
            Grid.Row="0"
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="8,85,0,0"
            Width="166"
            Height="23" />
        <TextBox
            x:Name="AnimalTextBox"
            x:FieldModifier="private"
            Grid.Column="0"
            Grid.Row="0"
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="8,35,0,0"
            Width="75"
            Height="28" />
        <Button
            Content="Remove animal from warehouse"
            x:Name="RemoveAnimalFromWarehouseButton"
            x:FieldModifier="private"
            Grid.Column="0"
            Grid.Row="0"
```

```
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="8,116,0,0"
    Width="166"
    Height="23" />
<Button
    RenderTransformOrigin="0.5,0.5"
    Content="Change number of animals"
    x:Name="ChangeNumberButton"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="8,147,0,0"
    Width="166"
    Height="23" />
<Button
    Content="Add to basket"
    x:Name="AddToBasketButton"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Bottom"
    Margin="8,0,0,299"
    Width="166"
    Height="23" />
<Button
    Content="Remove from basket"
    x:Name="RemoveFromBasketButton"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Bottom"
    Margin="8,0,0,268"
    Width="166"
    Height="23" />
<Button
    x:Name="BuyButton"
    Content="Buy"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Bottom"
    Margin="8,0,0,237"
    Width="166"
    Height="23" />
<Button
    RenderTransformOrigin="0.5,0.587"
```

```
    Content="Change price"
    x:FieldModifier="private"
    x:Name="ChangePriceButton"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="8,178,0,0"
    Width="166"
    Height="23" />
<Label
    x:Name="WarehouseLabel"
    Content="Warehouse"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    VerticalAlignment="Top"
    Height="120"
    Width="373"
    HorizontalAlignment="Left"
    Margin="182,63,0,0" />
<Label
    Content="Client's basket"
    x:FieldModifier="private"
    x:Name="BasketLabel"
    RenderTransformOrigin="0.5,0.5"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Stretch"
    VerticalAlignment="Top"
    Margin="272,327,8,0"
    Height="105"></Label>
<Label
    Content="Logs"
    x:FieldModifier="private"
    x:Name="LogsLabel"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Bottom"
    Margin="8,0,0,7.89999999999998"
    Width="320"
    Height="130.10000000000002" />
<Label
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="91,8,0,0"
    Width="75"
    Height="27"
    RenderTransformOrigin="0.5,0.5"
```

```
    Content="Number"
    x:Name="NumberLabel"
    x:FieldModifier="private" />
<Label
    Content="Price"
    x:Name="PriceLabel"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    VerticalAlignment="Top"
    Height="27"
    Width="75"
    HorizontalAlignment="Right"
    Margin="0,8,539,0" />
<TextBox
    x:Name="NumberTextBox"
    x:FieldModifier="private"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="91,35,0,0"
    Width="75"
    Height="28" />
<TextBox
    x:Name="PriceTextBox"
    x:FieldModifier="private"
    Text=""
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="174,35,0,0"
    Width="75"
    Height="28" />
<ComboBox
    x:Name="StateComboBox"
    BorderBrush="#FF707070"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Margin="8,319,0,0"
    Width="166"
    Height="20">
    <ComboBoxItem
        Content="Active"
        IsSelected="True"
        x:Name="Active" />
    <ComboBoxItem
        Content="Disactive"
        x:Name="Disactive" />
```

```
</ComboBox>
<ComboBox
    x:Name="AnimalComboBox"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Right"
    VerticalAlignment="Top"
    Margin="0,233,215,0"
    Width="120"
    Height="20">
    <ComboBoxItem
        Content=" "
        x:Name="EmptyCBI" />
    <ComboBoxItem
        Content="Animal"
        x:Name="AnimalCBI" />
</ComboBox>
<ComboBox
    x:Name="SpeciesComboBox"
    Grid.Column="0"
    Grid.Row="0"
    HorizontalAlignment="Right"
    VerticalAlignment="Top"
    Margin="0,253,215,0"
    Width="120"
    Height="20"
    Visibility="Hidden">
    <ComboBoxItem
        Content="Back"
        x:Name="BackSpeciesCBI" />
    <ComboBoxItem
        Content="Pet"
        x:Name="PetCBI" />
    <ComboBoxItem
        Content="Farm"
        x:Name="FarmCBI" />
</ComboBox>
<ComboBox
    x:Name="RacesPetComboBox"
    Grid.Row="0"
    HorizontalAlignment="Left"
    VerticalAlignment="Top"
    Width="120"
    Height="20"
    Visibility="Hidden"
    Margin="453,273,0,0"
    Grid.Column="0">
    <ComboBoxItem
        Content="Back"
        x:Name="BackRecesPetCBI" />
    <ComboBoxItem
        Content="Cat"
```

```
            x:Name="CatCBI" />
        <ComboBoxItem
            Content="Dog"
            x:Name="DogCBI" />
    </ComboBox>
    <ComboBox
        x:Name="RacesFarmComboBox"
        Grid.Column="0"
        Grid.Row="0"
        HorizontalAlignment="Right"
        VerticalAlignment="Top"
        Margin="0,273,215,0"
        Width="120"
        Height="20"
        Visibility="Hidden">
        <ComboBoxItem
            Content="Back"
            x:Name="BackRecesFarmCBI" />
        <ComboBoxItem
            Content="Cow"
            x:Name="CowCBI" />
        <ComboBoxItem
            Content="Chicken"
            x:Name="ChickenCBI" />
    </ComboBox>
    <DataGrid
        x:Name="WGrid"
        Grid.Column="0"
        Grid.Row="0"
        HorizontalAlignment="Left"
        VerticalAlignment="Top"
        Margin="8,231,0,0"
        Width="430"
        Height="80"
        x:FieldModifier="private" />
    <Label
        RenderTransformOrigin="0.5,0.5"
        x:FieldModifier="private"
        x:Name="Split"
        Content="----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------"
        Grid.Column="0"
        Grid.Row="0"
        HorizontalAlignment="Stretch"
        VerticalAlignment="Top"
        Margin="8,209,8,0"
        Height="24" />
    <TextBox
        x:Name="NumberClientTextBox"
        Grid.Column="0"
        Grid.Row="0"
```

```
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="182,350,0,0"
            Width="75"
            Height="28" />
        <Label
            RenderTransformOrigin="0.5,0.5"
            Content="Number"
            x:Name="NumberC"
            x:FieldModifier="private"
            Grid.Column="0"
            Grid.Row="0"
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="182,323,0,0"
            Width="75"
            Height="27" />
        <DataGrid
            Grid.Column="0"
            Grid.Row="0"
            HorizontalAlignment="Left"
            VerticalAlignment="Top"
            Margin="8,440,0,0"
            Width="430"
            Height="78"
            x:Name="CGrid"
            x:FieldModifier="private" />
        <Grid.ColumnDefinitions></Grid.ColumnDefinitions>
    </Grid>
</Window>
```

```
/*
 * Created by SharpDevelop.
 * User: Dominik
 * Date: 2014-12-15
 * Time: 16:16
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using PetShop.V.WindowsApp;
using PetShop.M.Classes.Product;

namespace PetShop.V.WindowsApp{
   public partial class WPFForm : Window{
      public WPFForm(){
         InitializeComponent();
      }

      public void RegisterButton1EventHandler(System.Windows.RoutedEventHandler
eventHandler){
         this.AddAnimalToWarehouseButton.Click += new System.Windows.RoutedEventHandler
(eventHandler);
      }

      public void RegisterButton2EventHandler(System.Windows.RoutedEventHandler
eventHandler){
         this.RemoveAnimalFromWarehouseButton.Click += new System.Windows.RoutedEventH
andler(eventHandler);
      }

      public void RegisterButton3EventHandler(System.Windows.RoutedEventHandler
eventHandler){
         this.ChangeNumberButton.Click += new System.Windows.RoutedEventHandler(eventHa
ndler);
      }

      public void RegisterButton4EventHandler(System.Windows.RoutedEventHandler
eventHandler){
         this.ChangePriceButton.Click += new System.Windows.RoutedEventHandler(eventHandl
er);
      }

      public void RegisterButton5EventHandler(System.Windows.RoutedEventHandler
eventHandler){
```

```
        this.AddToBasketButton.Click += new System.Windows.RoutedEventHandler(eventHandler);
    }

    public void RegisterButton6EventHandler(System.Windows.RoutedEventHandler eventHandler){
        this.RemoveFromBasketButton.Click += new System.Windows.RoutedEventHandler(eventHandler);
    }

    public void RegisterButton7EventHandler(System.Windows.RoutedEventHandler eventHandler){
        this.BuyButton.Click += new System.Windows.RoutedEventHandler(eventHandler);
    }

    public void RegisterComboBox1EventHendler(System.Windows.Controls.SelectionChangedEventHandler eventHandler){
        this.AnimalComboBox.SelectionChanged += new SelectionChangedEventHandler(eventHandler);
    }

    public void RegisterComboBox2EventHendler(System.Windows.Controls.SelectionChangedEventHandler eventHandler){
        this.SpeciesComboBox.SelectionChanged += new SelectionChangedEventHandler(eventHandler);
    }

    public void RegisterComboBox3EventHendler(System.Windows.Controls.SelectionChangedEventHandler eventHandler){
        this.RacesPetComboBox.SelectionChanged += new SelectionChangedEventHandler(eventHandler);
    }

    public void RegisterComboBox4EventHendler(System.Windows.Controls.SelectionChangedEventHandler eventHandler){
        this.RacesFarmComboBox.SelectionChanged += new SelectionChangedEventHandler(eventHandler);
    }

    public void RegisterComboBox5EventHendler(System.Windows.Controls.SelectionChangedEventHandler eventHandler){
        this.StateComboBox.SelectionChanged += new SelectionChangedEventHandler(eventHandler);
    }

    public void SetTextWarehouse(string text){
        this.WarehouseLabel.Content = text;
    }

    public void SetTextClient(string text){
        this.BasketLabel.Content = text;
    }
```

```
public void SetTextLogs(string text){
   this.LogsLabel.Content = text;
}

public string GetAnimal(){
   return this.AnimalTextBox.Text;
}

public string GetNumberWerehouse(){
   return this.NumberTextBox.Text;
}

public string GetNumberClient(){
   return this.NumberClientTextBox.Text;
}

public string GetPrice(){
   return this.PriceTextBox.Text;
}

public System.Windows.Controls.DataGrid GetCGrid(){
   return this.CGrid;
}

public System.Windows.Controls.DataGrid GetWGrid(){
   return this.WGrid;
}

public int GetSelectedComboBox(){
   if(this.AnimalComboBox.IsVisible){
      if(this.SpeciesComboBox.IsVisible){
         if(this.RacesPetComboBox.IsVisible){
            if(this.RacesPetComboBox.SelectedIndex == 1){
               return 4;
            }
            else if(this.RacesPetComboBox.SelectedIndex == 2){
               return 5;
            }
            else{
               return 2;
            }
         }
         else if(this.RacesFarmComboBox.IsVisible){
            if(this.RacesFarmComboBox.SelectedIndex == 1){
               return 6;
            }
            else if(this.RacesFarmComboBox.SelectedIndex == 2){
               return 7;
            }
            else{
               return 3;
```

```
                }
            }
            else{
                if(this.SpeciesComboBox.SelectedIndex == 1){
                    return 2;
                }
                else if(this.SpeciesComboBox.SelectedIndex == 2){
                    return 3;
                }
                else{
                    return 1;
                }
            }
        }
        else{
            if(this.AnimalComboBox.SelectedIndex == 1){
                return 1;
            }
        }
    }
    else{
        return 0;
    }
    return 0;
  }

 }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PetShop.C;
using PetShop.C.Strategy.StrategyInterface;
using PetShop.C.Strategy.Strategies;
using PetShop.M;
using PetShop.V;
using PetShop.V.Views;

namespace PetShop{

    /* Wzorce:
     * 1. Singleton - in Warehouse class, creating only one global warehouse
     * 2. Multiton - in ProducktContainter class, cant add existed product
     * 3. MVC - all project
     * 4. Strategy - in controller, controller can change strategy, 3 avaible strategy: Console, WPF, Form
     * 5. Observer - in model and view, model notify view when it change state, data
     * 6. Prototype - in Animal class, clone product couse c# work with references
     * 7. Builder - in Pet class, creating a Dog, Cat object
     * 8. Factory method - in Farm class, creating a Cow, Chicken object
     * 9. Mediator - in Client, sent a massage from client to logs about what he bought
     * 10. Decorator - in Animal and childs classes, decorating each animal
     * 11. Facade - teoretical: controller is facade; model, view is subsytems, client have access to method in strategy
     * 12. State - in Client, if client is disactive he cant do anything
     * Use Windows Form and WPF for windows
     */

    class Program{
        [STAThread]
        static void Main(string[] args){

            // declaration view, model, controller and strategy for controller
            View view;
            Model model;
            IStrategy strategy;
            Controller controller;

            // create view
            view = new ConsoleView();
            //view = new WinAppView(); // dont support all functionality
            //view = new WPFAppView();

            // create model
            model = new Model();

            // create strategy
            strategy = new ConsoleStrategy();
```

```
            //strategy = new WinAppStrategy(); // dont supported all functionality
            //strategy = new WPFAppStrategy();

            // create controller with model, view and strategy
            controller = new Controller(model, view, strategy);

            // attach view to model : observer pattern
            controller.Model.Attach(view);

            // start application
            controller.Start();
        }
    }
}
```