

Informatikos fakultetas

**T120B516 Objektinis programų projektavimas**

**Projekto ataskaita 2**

|  |  |  |
| --- | --- | --- |
|  | Studentai: | Ignas Savickas IFF-4/2 |
| Jonas Ausevičius IFF-4/2 |
|  |  |
| Dėstytojai: | Lekt. dr. Andrej Ušianov |
| Lekt. Kęstutis Valinčius |

Kaunas 2017

Turinys

[1. Projektavimo šablonai (3 laboratorinis) 3](#_Toc500277836)

[1.1. Template Method 3](#_Toc500277837)

[1.1.1. Klasių diagrama 3](#_Toc500277838)

[1.1.2. Esminis kodas 3](#_Toc500277839)

[1.1.3. Naudojimo pagrindimas 5](#_Toc500277840)

[1.2. Flyweight 5](#_Toc500277841)

[1.2.1. Klasių diagrama 6](#_Toc500277842)

[1.2.2. Esminis kodas 6](#_Toc500277843)

[1.2.3. Naudojimo pagrindimas 7](#_Toc500277844)

[1.3. State 7](#_Toc500277845)

[1.3.1. Klasių diagrama 7](#_Toc500277846)

[1.3.2. Esminis kodas 7](#_Toc500277847)

[1.3.3. Naudojimo pagrindimas 7](#_Toc500277848)

[2. Projektavimo šablonai (2 laboratorinis) 8](#_Toc500277849)

[2.1. Adapter 8](#_Toc500277850)

[2.1.1. Klasių diagrama 8](#_Toc500277851)

[2.1.2. Esminis kodas 8](#_Toc500277852)

[2.1.3. Naudojimo pagrindimas 9](#_Toc500277853)

[2.2. Prototype 9](#_Toc500277854)

[2.2.1. Klasių diagrama 9](#_Toc500277855)

[2.2.2. Esminis kodas 9](#_Toc500277856)

[2.2.3. Naudojimo pagrindinis 10](#_Toc500277857)

[2.3. Command 10](#_Toc500277858)

[2.3.1. Klasių diagrama 10](#_Toc500277859)

[2.3.2. Esminis kodas 11](#_Toc500277860)

[2.3.3. Naudojimo pagrindimas 12](#_Toc500277861)

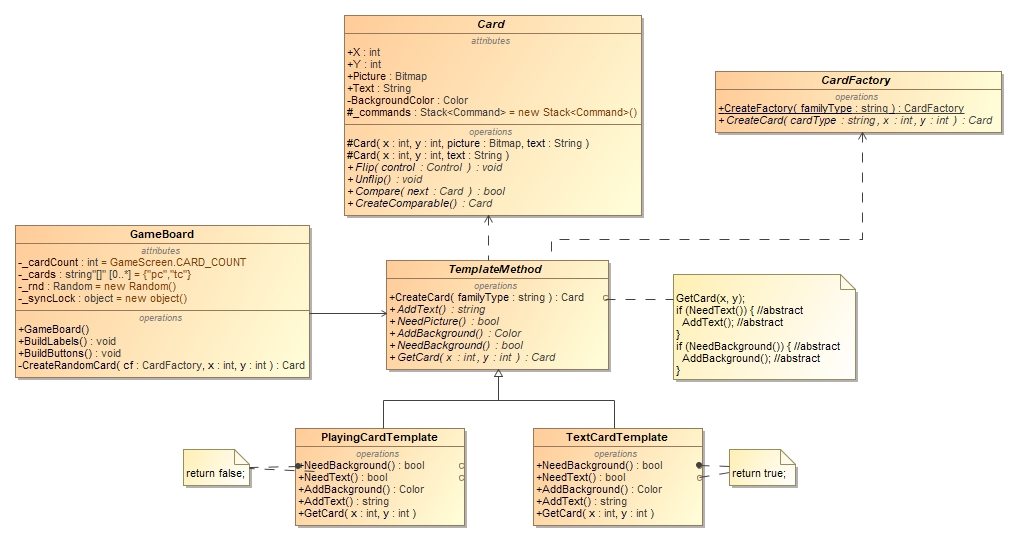
[3. Išvados 13](#_Toc500277862)

# Projektavimo šablonai (3 laboratorinis)

Kiekvienam laboratoriniam darbui buvo pasirinkti atskiri projektavimo šablonai. Pirmajam – Template Method, Flyweight, State.

## Template Method

### Klasių diagrama



pav. 1. Template Method projektavimo šablono pritaikymo klasė

### Esminis kodas

public abstract class TemplateMethod

{

public virtual Card CreateCard(int x, int y)

{

Card card = GetCard(x, y);

if (NeedText())

{

card.Text = card.Text + AddText();

}

if (NeedBackground())

{

card.BackgroundColor = AddBackground();

}

return card;

}

protected abstract Card GetCard(int x, int y);

protected abstract bool NeedText();

protected abstract string AddText();

protected abstract bool NeedBackground();

protected abstract Color AddBackground();

}

public class PlayingCardTemplate : TemplateMethod

{

private readonly string[] \_playingCards = { "h", "s", "d", "c" };

private readonly System.Random \_rnd = new System.Random();

private readonly object \_syncLock = new object();

protected override Card GetCard(int x, int y)

{

var cf = CardFactory.CreateFactory("pc");

int index;

lock (\_syncLock)

{

index = \_rnd.Next(0, \_playingCards.Length);

}

string cardType = \_playingCards[index];

return cf.CreateCard(cardType, x, y);

}

protected override bool NeedText()

{

return false;

}

protected override string AddText()

{

return null;

}

protected override bool NeedBackground()

{

return false;

}

protected override Color AddBackground()

{

return Color.White;

}

}

public class TextCardTemplate : TemplateMethod

{

private readonly string[] \_textCards = { "l", "n" };

private readonly System.Random \_rnd = new System.Random();

private readonly object \_syncLock = new object();

protected override Card GetCard(int x, int y)

{

var cf = CardFactory.CreateFactory("tc");

int index;

lock (\_syncLock)

{

index = \_rnd.Next(0, \_textCards.Length);

}

string cardType = \_textCards[index];

return cf.CreateCard(cardType, x, y);

}

protected override bool NeedText()

{

return true;

}

protected override string AddText()

{

return Random.GetLetter() + Random.GetNumber();

}

protected override bool NeedBackground()

{

return true;

}

protected override Color AddBackground()

{

return Random.GetColor();

}

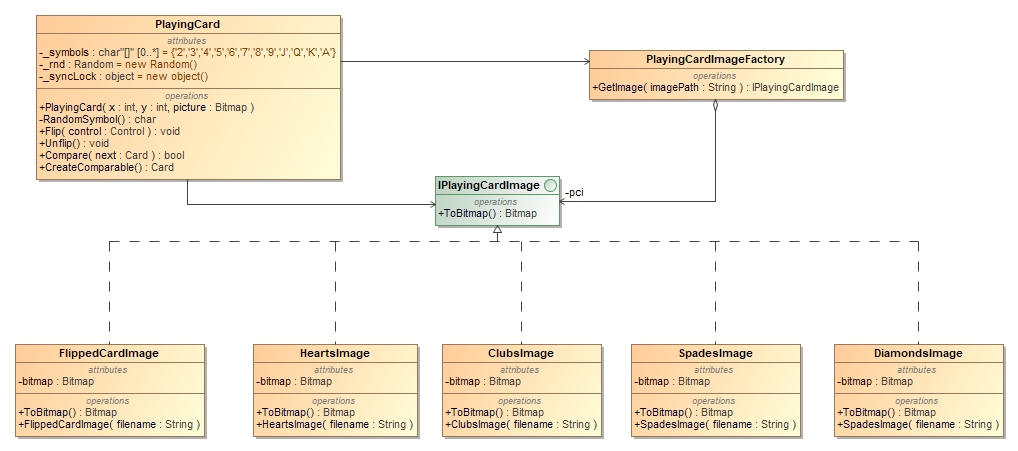
}

### Naudojimo pagrindimas

Sukuriamas bendras kortos kūrimo algoritmas, kurio žingsnių eiliškumas yra nekeičiamas. Kiekvienas žingsnis yra keičiamas klasėje, kuri paveldi abstrakčią klasę. Taip klientas negalės keisti kortos kūrimo algoritmo. Tam tikrą korta bus sukuriama pasinaudojant tam tikrą algoritmą.

## Flyweight

### Klasių diagrama



pav. 2. Flyweight projektavimo šablono pritaikymo klasės

### Esminis kodas

public class PlayingCardImageFactory

{

private static readonly Hashtable Hash = new Hashtable();

public static IPlayingCardImage GetImage(string image)

{

IPlayingCardImage pci = (IPlayingCardImage) Hash[image];

if (pci != null) return pci;

if (image.Equals("clubs"))

{

pci = new ClubsImage();

}

if (image.Equals("diamonds"))

{

pci = new DiamondsImage();

}

if (image.Equals("hearts"))

{

pci = new HeartsImage();

}

if (image.Equals("spades"))

{

pci = new SpadesImage();

}

if (image.Equals("flipped\_card"))

{

pci = new FlippedCardImage();

}

Hash.Add(image, pci);

return pci;

}

}

public interface IPlayingCardImage

{

Bitmap ToBitmap();

}

public class ClubsImage : IPlayingCardImage

{

private readonly Bitmap \_bitmap;

public ClubsImage()

{

\_bitmap = new Bitmap("../../Resources/clubs.png");

}

public Bitmap ToBitmap()

{

return \_bitmap;

}

}

Visos kitos klasės, kurios implementuoja IPlayingCardImage, yra labai panašios į ClubsImage klasė. Skiriasi tik paveiksliuko vieta.

### Naudojimo pagrindimas

Gaunant naują paveiksliuką programa atsimina jo vietą ir kai kitą kartą reikalingas toks pat paveiksliukas, tai programa nekuria naujo, o pasiima seną. Taip tausojama kompiuterio atmintis ir greičiau užkraunami paveiksliukai žaidimo paleidimo metu.

## State

### Klasių diagrama

pav. . State šablono pritaikymo klasių lentelės

### Esminis kodas

### Naudojimo pagrindimas

# Projektavimo šablonai (4 laboratorinis)

Antrajame laboratoriniame darbe buvo naudojami 4 projektavimo šablonai: Chain of responsibility, Interpreter, Mementor ir Null Object.

## Chain of responsibility

### Klasių diagrama

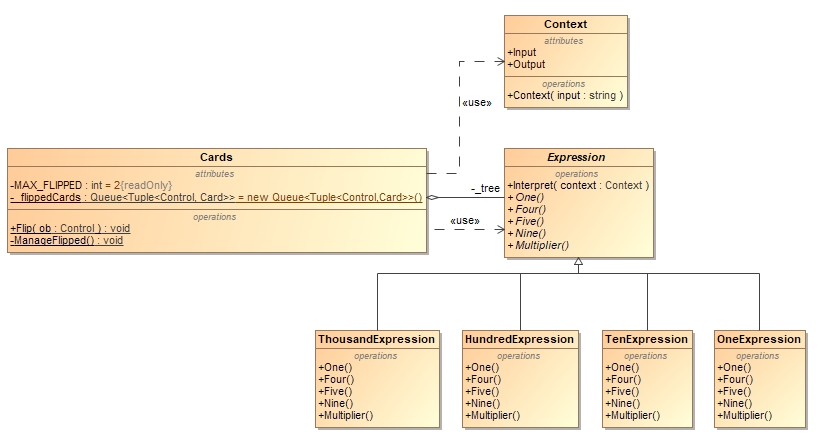
pav. . Chain of responsibility projektavimo šablono klasių diagrama

### Esminis kodas

### Naudojimo pagrindimas

## Interpreter

### Klasių diagrama



pav. 5. Interpreter projektavimo šablono pritaikymo klasės

### Esminis kodas

public class Context

{

public string Input { get; set; }

public int Output { get; set; }

public Context(string input)

{

Input = input;

}

}

public abstract class Expression

{

public void Interpret(Context context)

{

if (context.Input.Length == 0)

{

return;

}

if (context.Input.StartsWith(Nine()))

{

context.Output += 9 \* Multiplier();

context.Input = context.Input.Substring(2);

}

else if (context.Input.StartsWith(Four()))

{

context.Output += 4 \* Multiplier();

context.Input = context.Input.Substring(2);

}

else if (context.Input.StartsWith(Five()))

{

context.Output += 5 \* Multiplier();

context.Input = context.Input.Substring(1);

}

while (context.Input.StartsWith(One()))

{

context.Output += Multiplier();

context.Input = context.Input.Substring(1);

}

}

public abstract string One();

public abstract string Four();

public abstract string Five();

public abstract string Nine();

public abstract int Multiplier();

}

public class HundredExpression : Expression

{

public override string One()

{

return "C";

}

public override string Four()

{

return "CD";

}

public override string Five()

{

return "D";

}

public override string Nine()

{

return "CM";

}

public override int Multiplier()

{

return 100;

}

}

public class TenExpression : Expression

{

public override string One()

{

return "X";

}

public override string Four()

{

return "XL";

}

public override string Five()

{

return "L";

}

public override string Nine()

{

return "XC";

}

public override int Multiplier()

{

return 10;

}

}

Kitų dvi klasių principas panašus į dvi paskutines klases. Skiriasi tik tas, kad tūkstančių išraiškoje grąžinama tik vientas „M“ ir daugiklis 1000, o vieneto išraiškoje – vienetai romėniškoje numeracijoje.

### Naudojimo pagrindinis

Gaunant romėnišką skaičių programa paverčia skaičių į arabišką panaudodama šį šabloną. Nereikia naudoti išorinių bibliotekų. Atversti skaičiai sudedami ir atspausdinami komandinėje eilutėje.

## Mementor

### Klasių diagrama

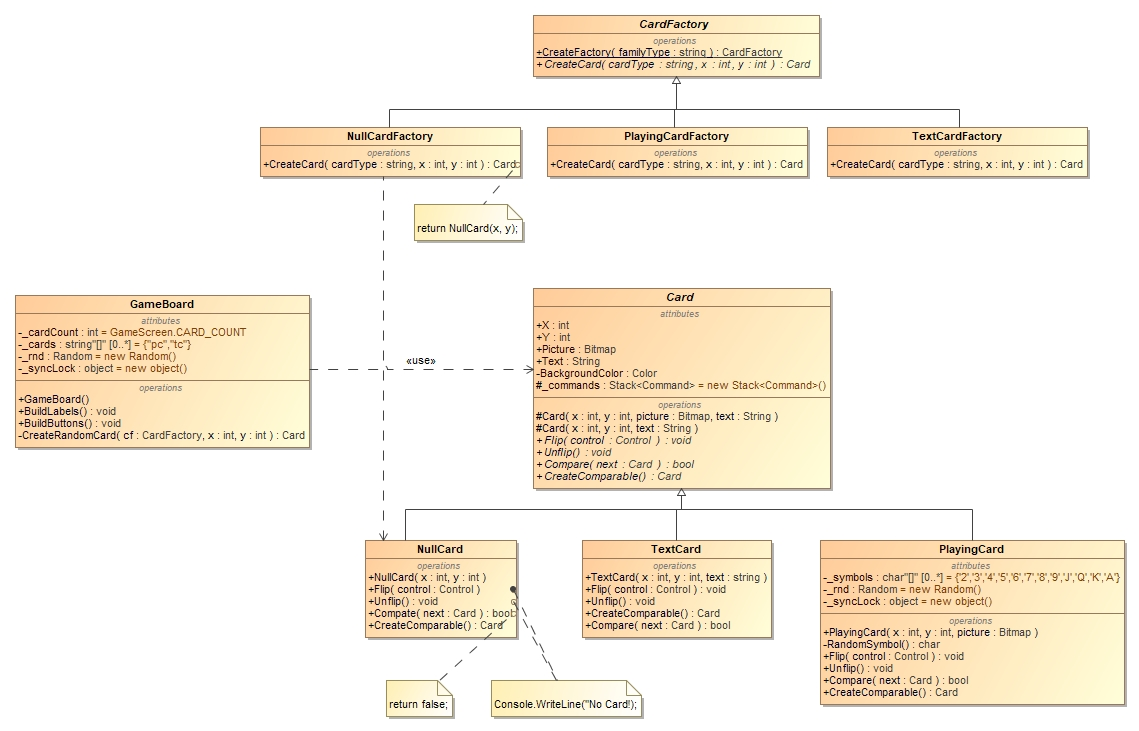
pav. . Mementor projektavimo šablono klasių diagrama

### Esminis kodas

### Naudojimo pagrindimas

## Null Object

### Klasių diagrama



pav. . Null Object projektavimo šablono pritaikymo klasės

### Esminis kodas

public abstract class CardFactory

{

public static CardFactory CreateFactory(string familyType)

{

if (familyType.Equals("pc"))

{

return new PlayingCardFactory();

}

if (familyType.Equals("tc"))

{

return new TextCardFactory();

}

return new NullCardFactory();

}

public abstract Card CreateCard( string cardType, int x, int y);

}

public class NullCardFactory : CardFactory

{

public override Card CreateCard(string cardType, int x, int y)

{

return new NullCard(x, y);

}

}

public abstract class Card

{

public int X { get; set; }

public int Y { get; set; }

public Bitmap Picture { get; set; }

public string Text { get; set; }

public Color BackgroundColor { get; set; }

protected readonly Stack<Command> \_commands = new Stack<Command>();

protected Card(int x, int y, Bitmap picture, string text)

{

X = x;

Y = y;

Picture = picture;

Text = text;

}

protected Card(int x, int y, string text)

{

X = x;

Y = y;

Text = text;

}

public abstract void Flip(Control control);

public abstract void Unflip();

public abstract bool Compare(Card next);

public abstract Card CreateComparable();

}

public class NullCard : Card

{

public NullCard(int x, int y) : base(x, y, Properties.Resources.\_null, "null")

{

}

public override void Flip(Control control)

{

Console.WriteLine(@"No card!");

}

public override void Unflip()

{

Console.WriteLine(@"No card!");

}

public override bool Compare(Card next)

{

return false;

}

public override Card CreateComparable()

{

return (Card)this.MemberwiseClone();

}

}

### Naudojimo pagrindimas

Jeigu atsitiktų taip, kad atitinkamas tekstas nebūtų sugeneruotas, tai būtų sukuriama tuščia gamykla, kurioje būtų sukuriama tuščia korta arba tiesiog tuščia korta būtų sukurta tam tikrose gamyklose. Taip atsitikus tokios kortos būtų pažymėtos raudonu „X“ ir nupieštos žaidime. Paspaudus ant tokios kortos, veiksmas būtų atšauktas ir komandinėje eilutėje būtų išvesta „No Card!”. Panaudojus šį šabloną, žaidimas nenustotų veikti, tik išvestų klaidą.

# Išvados